Eight Semester Theory Examination 2009-10

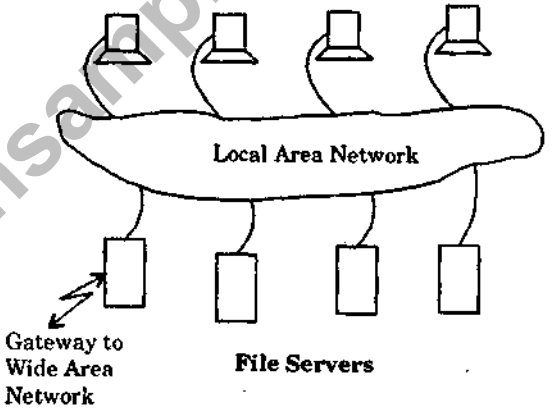
DISTRIBUTED SYSTEMS

*Note : (i) Attempt **all** questions.*

1. **Do *any four* parts of the following :**

Q. 1. (a) How the resource sharing done in distributed system ? Explain with an example.

Ans. It is the main motivating factor for constructing the distributed system. A computer in a distributed system environment can request for a remote service through the communication network. All the hardware and software resources are easily shared among all the workstations using this system. Such type of system is very useful for the organization having many departments communicating together and using the shared resources. In general, resource sharing in a distributed system provides mechanism for sharing files at remote sites, processing information in a distributed database, printing files at remote sites, using remote specialized hardware devices such as a high speed array processor, and performing other operations. **Example :** Network of workstations.



- Personal workstations + processors not assigned to specific users.
- Single file systems, with all files accessible from all machines in the same way and using the same path name.
- For a certain command the system can look for the best place to execute it.

Q. 1. (b) Discuss the limitation of distributed system.

Ans. 1. Data Security : It is a big problem due to sharing of data among various users of the communication network. The sharing may came access to the secret and confidential data by any user.

2. Absence of shared memory : All the computer systems in distributed environment have their own specific memory and so the entire system do not have the shared memory. Because of this limitation the current state of entire system is not provided to each user. The current state of a system tells about the system behaviour, fault tolerance, recovery from faults, etc. So the user in distributed system would not be aware of these specifications.

3. No Global Clock : When programs need to cooperate they coordinate their actions by exchanging messages. Close coordination often depends on a shared idea of the time at which the programs actions occur. But it turns out that there are limits to the accuracy with which the computers in a network can synchronize their clocks *i.e.*, there is no single global notion of the correct time. This is a direct consequence of the fact that the only communication is by sending messages through a network.

4. Independent failures : All systems can fail and it is the responsibility of system designers to plan for the consequences of possible failures. Faults in the network results in the isolation of the computers that are connected to it, but that doesn't mean that they stop running. In fact the programs on them may not be able to detect whether the network has failed or has become unusually slow. Similarly the failure of a computer or the unexpected termination of a program somewhere in the system is not immediately made known to other components with which it communicates. Each component of the system can fail independently, leaving the others still running.

Q. 1. (c) What do you mean by Global State of the distributed system ? Also explain the main features of consistent Global State.

Ans. "The global state of a distributed computation is the set of local states of all individual processes involved in the computation plus the state of the communication channels."

That is, $GS = \{LS_1, LS_2, \ldots . LS_n\}$ where n is the number of sites in the system.

Note that any collection of local states of sites need not represent a consistent global state. Consistency has a connotation that for every effect or outcome recorded in a global state, the cause of the effect must also be recorded in the global state.

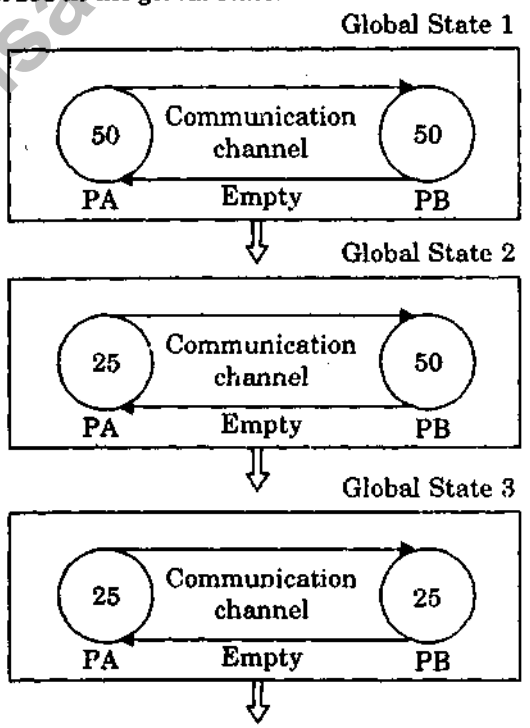


Fig. Global state within the ticket reservation system

Consistent global state : A global state $GS = \{LS_1, LS_2, \ldots LS_n\}$ is consistent if

$$V_i, V_j : \leq I, \leq n :: \text{inconsistent}(LS_i, LS_j) = \phi$$

Thus, in a consistent global state, for every received message a corresponding send event is recorded in the global state. In an inconsistent global state, there is at least one message whose receive event is recorded but its send event is not recorded in the correspond to consistent and inconsistent global state, respectively.

Q. 1. (d) Differentiate between Token based algorithm and non token based algorithm.

Ans. In **token based algorithm,** a unique token is shared among all sites. A site is allowed to enter its CS if it possesses the token. Depending upon the way a site carries out its search for the token there are numerous token based algorithms.

(i) Token based algorithms use sequence numbers instead of timestamps. Every request for the token contains a sequence number and the sequence numbers of sites advance independently. A site increments its sequence number counter every time it makes a request for the token. A primary function of the sequence numbers is to distinguish between old and current requests.

(ii) A correctness proof of token based algorithms to ensure that mutual exclusion is enforced is trivial because an algorithm guarantees mutual exclusion so long as a site holds the token during the execution of the CS.

The various token based algorithms are :

- Token ring algorithm.
- Ring-based algorithm.
- Suzuki-Kasami's Broadcast algorithm.
- Singhal's heuristic algorithm.

In **Non-token based algorithms**, a site communicates with a set of other sites to arbitrate who should execute the CS next. For a site S_i, request set R_i contains of all those sites from which site S_i must acquire permission before entring the CS.

Non token based algorithms timestamps to order requests for the CS and to resolve conflicts between simultaneous requests for the CS. In all these algorithms, logical clocks are maintained and updated according to Lamport's scheme. Each request for the CS gets a timestamp and smaller timestamp requests have priority over larger time-stamp requests.

The various algorithms are :

- Central coordinator algorithm.
- Ricart Agrawala algorithm.

Q. 1. (e) Explain the classification of distributed mutual exclusion.

Ans. The problem of mutual exclusion has received considerable attention and several algorithms to achieve mutual exclusion in distributed systems have been proposed. They tend to differ in their communication topology and in the amount of information maintained by each site about other sites. These algorithms can be grouped into two classes :

1. The algorithms in the first class are nontoken based. These algorithms require two or more successive rounds of message exchanges among the sites. These algorithms are assertion based because a site can enter its critical section (CS) when an assertion defined on its local variables

becomes true. Mutual exclusion is enforced because the assertion becomes true only at one site at any given time.

2. the algorithms in the second class are token-based. A unique token is shared among the sites. A site is allowed to enter its CS if it possesses the token and it continues to hold the token until the execution of the CS is over. These algorithms essentially differ in the way a site carries out the search for the token.

Q. 1. (f) Discuss the web challenges for implementing distributed system.

Ans. Although distributed systems are to be found everywhere, their design is quite simple and there is still a lot of scope to develop more ambitios and applications, but future designers need to be aware of them and to be careful to take them to account.

The construction of distributed system produces many challenges :

1. Heterogeneity : They must be constructed from a variety of different networks, O.S., hardware and programming languages.

2. Openness : Distributed systems should be extensible the first step is to publish the interfaces of the components but the integration of components written by different programmers is a real challenge.

3. Security : Encryption can be used to provide adequate protection of shared resources and to keep sensitive information secret when it transmitted in messages over a network.

4. Scalability : A distributed system is scalable if the cost of adding a user is a constant amount in terms of the resources that must be added. Frequently accessed data can be replicated.

5. Failure Handling : Any process or network may fail independently of the others. Therefore each component needs to be aware of the possible ways in which the components it depends on may fail and be designed to deal with each of those failures appropriately.

6. Concurrency : The presence of multiple users in a distributed system is a source of concurrent requests to its resources. Each resource must be designed to be safe in a concurrent environment.

7. Transparency : The aim is to make certain aspects of distribution invisible to the programmer so that they need only be concerned with the design of their particular application. For *e.g.*, they need not be concerned with its location or the details of how its operations are accessed by other components, or whether it will be replicated.

2. Attempt *any two* parts of the following :

Q. 2. (a) Define deadlocks. Differentiate between resource and communication Deadlocks. Discuss various deadlock handling strategies in detail.

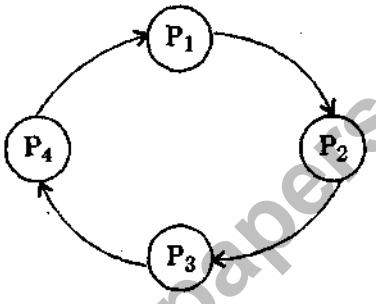
And. Deadlock : The deadlock is defined as the permanent blocking of the process. That is a set of process is waiting for an event that is never going to occur.

The problem of deadlock has been generally studied in distributed system by using the following model :

1. The system have only reusable resources.
2. Process are allowed only exclusive access to resources.
3. There is only one copy of each resources.

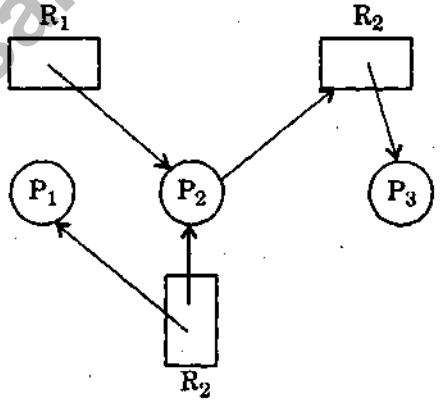
Communication Deadlock : When processes wait to communicate with other processes among a set of processes, a waiting process can unlock on receiving a communication from any one of these processes then communication deadlock occurred.

A set of process is communication deadlocked if each process in the set is waiting to communicate with another process in the set and no process in the set ever initiates any further communication for which it is waiting.



Resource Deadlock : Processes can simultaneously and cannot proceed until they have acquired all those resources.

A set of processes is resource deadlocked if each process in the set requests resources held by another process in the set and it must receive all of the requested resources before it became unblocked.



Deadlock handling strategies : These are 3 strategies to handle deadlocks :

1. Deadlock Prevention
2. Deadlock Avoidance
3. Deadlock Detection.

1. Deadlock Prevention : It is commonly achieved by either having a process acquire all the needed resources. Simultaneously before it begins execution or by pre-empting a process that holds the needed resources.

2. Deadlock Avoidance : In the deadlock avoidance approach to distributed systems, a resource is granted to a process if the resulting global system state is safe.

3. Deadlock Detection : If requires an examination of the states of process interactions for the presence of cyclical wait. Deadlock detection in D.S. has two favourable conditions :

(a) Once a cycle is formed in the WFG persists until it is detected and broken.

(b) Cycle detection can proceed concurrently the normal activities of a system.

Q. 2. (b) Write short notes on following :

(i) Wait for graph.

Ans. (i) Wait for Graph : A wait-for graph is a directed graph used for deadlock detection in operating systems and relational database systems.

In computer science, a system that allows concurrent operation of multiple processes and locking of resources and which does not provide mechanisms to avoid or prevent deadlock must support a mechanism to detect deadlocks and an algorithm for recovering from them.

One such deadlock detection algorithm makes use of a wait-for graph to track which other processes is currently blocking on. In a wait-for graph, processes are represented as nodes, and an edge from process P_i to P_j implies P_j is holding a resource that P_i needs and thus P_i is waiting for P_j to release it lock on that resource. A deadlock exists if the graph contains any cycles.

(ii) Atomic commit in distributed database systems.

Ans. In the problem of atomic, sites of a distributed database system must agree whether commit or about a transaction.

1. In the first phase of atomic commit, sites execute their part of a distributed transaction and broadcast their decision to all other sites.

2. In the second phase, each site based on what it received from other sites in the first phase, decides whether to commit or abort its part of the distributed computation since every site receives an identical response from all other sites, they can send conflicting response other site, causing them to make conflicting decision.

Since every site receives an identical response from all other sites, they will reach the same decision.

But if some sites behave maliciously, they can send a conflicting response to other sites, causing them to make conflicting decision. In this situation, we can use the Byzantine agreement to ensure that all non faulty processor reach a common decision about a distributed transaction.

Q. 2. (c) Explain Lamport-Shostak-Pease algorithm (Oral Message Algorithm) for $3m + 1$ or more processors where m is the no. of faulty processors.

Ans. The algorithm works when following is true :

$$n >= 3m + 1$$

where n = total number of processors.

m = number of faulty processors.

The algorithm works differently with faulty processors or without or faulty processors.

OM (0) : Algorithm (*i.e.*, $m = 0$)

Step 1 : Source processor sends its initial value to each and every processor.

Step 2 : Each processor uses this received value (if no value is received it uses a default value 0).

OM (m) Algorithm (*i.e.*, $m > 0$)

Step 1 : The source processor sends its value to every processor.

Step 2 : If a processor does not receive value it uses a default value of zero.

Step 3 : If a processor (p_i) receives a value vi from source, then it behave like source processor, and itself and the source from where it is receiving value).

Step 4 : If for a processor P_i, V_j, ($j \neq I$) is value received from P_j, the P_i uses the majority value as agreement value.

Complexity of Algorithm

1. The algorithm $OM(m)$ includes $n-1$ separate execution of algorithm $OM\ (m-1)$ each of which invokes $(n-2)$ execution of Algorithm $OM\ (m-2)$ and so on.

2. Hence, there are $(n-1)\,(n-2)\,(n-3)\ldots.(n-k)$ separate execution of algorithm $OM(m-k)$, $k = 1, 2, 3, m+1$ Hence.

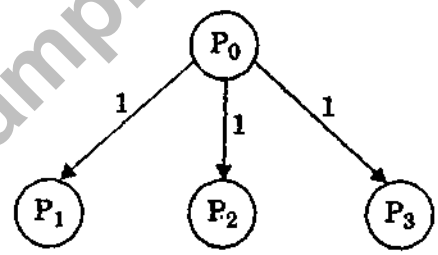
3. The message complexity : $O(nm)$

4. This algorithm will be very clear after discussing following examples.

5. Let there are four processor P_0, P_1, P_2, P_3 processor communicate with each other with value 0 and 1. P_0 is source processor and processor P_2 is faulty.

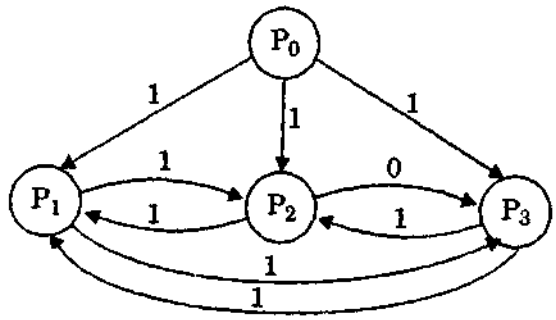
6. Processor $P0$ starts algorithm $OM(1)$ by sending value 1 to all other processors.

As in the given figure.



7. Next is execution of step 2 of algorithm $OM(1)$.

8. After receiving value 1 from source processor $P_1P_2P_3$ processors execute algorithm $OM(0)$. This is in Fig.



9. Processor P_1 and P_3 are Non-Faulty and send value 1 to (P_2, P_3) and (P_1, P_2) respectively.

10. P_2 send 1 to P_1 and 0 to P_3. Next is processor P_1, P_2, P_3 execute step 3 of $OM(1)$ for deciding majority value.

11. Following is the table of given value of their majority values

Process	Vector received	Majority value
P1	(1, 1, 1)	1
P2	(1, 1, 1)	1
P3	(1, 1, 0)	1

12. Hence all the three process agree to same value, *i.e.*, 1 and Byzanitine agreement problem is solved.

3. Attempt *any two* parts of the following :

Q. 3. (a) (i) What is the communication models proposed for the communication between the distributed objects ?

Ans. The two models have been widely used to develop ditributed applications for distributed systems :

1. Message Passing Method
2. Remote Procedure Calls.

Message Passing Method : The message passing method for communication depends on the two basic concepts for message transfer and these are SEND and RECEIVE processes.

The two are the main concept for message transfer among various sites of the distributed system and known as SEND primitive and the RECEIVE primitives.

Send primitive requires message and destination address. Receive primitive requires source address and buffer area.

Remote Procedure Calls :

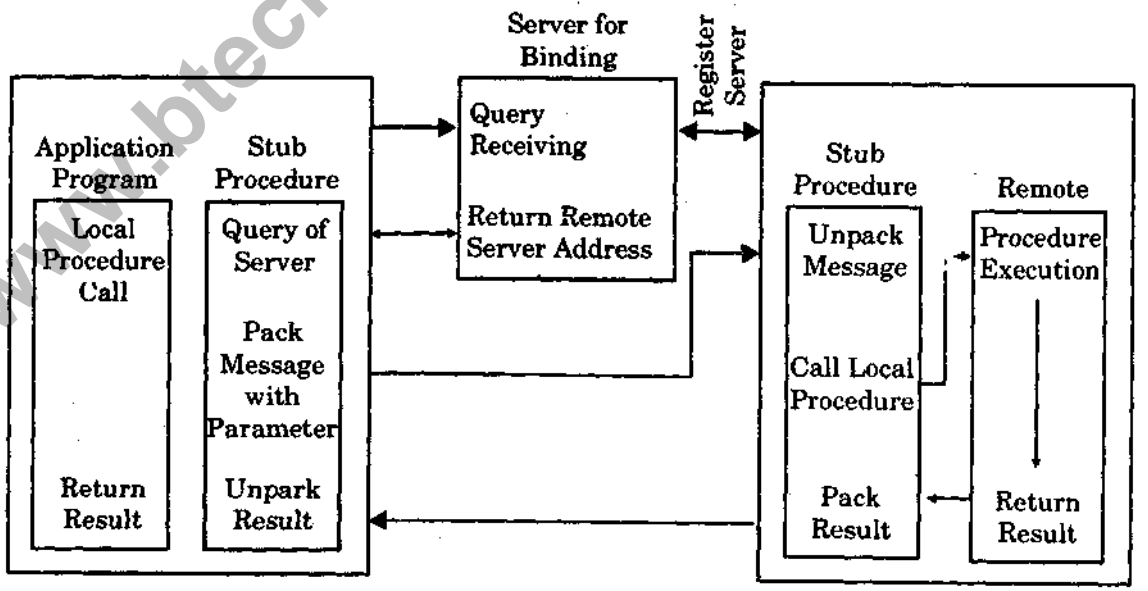
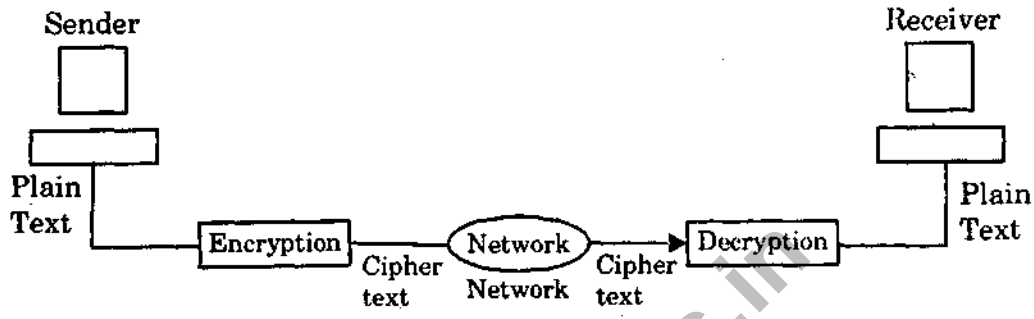


Fig. Generation of RPC in a Distributed System

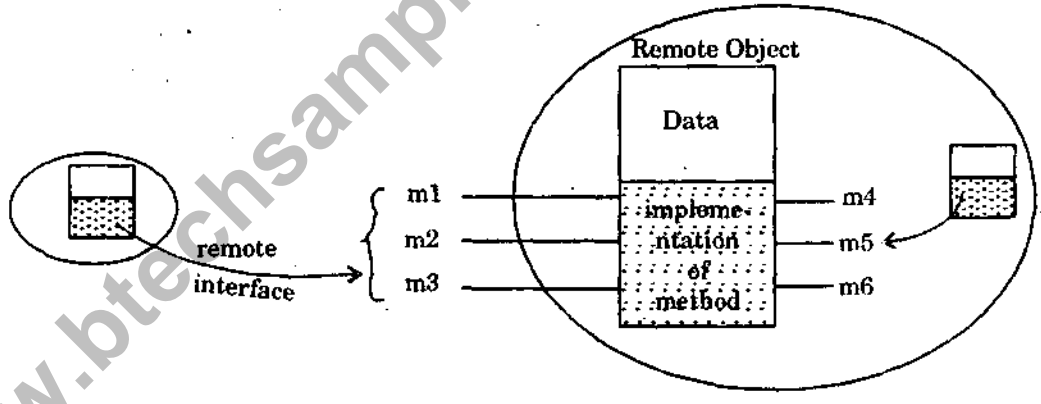


Q. 3. (a) (ii) Explain following with an example :

(A) Remote object reference.

(B) Remote interface.

Ans. The class of a remote object implements the methods of its remote interface. For example as public instance in Java Object in other process invoke only method that belong to its remote interface local object can invoke the method in the remote interface as well as other method implemented by a remote object.

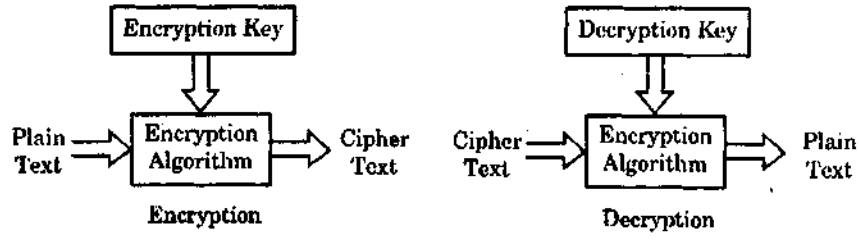


Q. 3. (b) What are the public and private keys ? List the key differences and issue in public keys cryptography and private key cryptography.

Ans. With public key cryptography, there are two different keys for encryption and decryption. Public key cryptography requires the surity that each used must have two keys :

(i) A public key : A public key is used by all the parties for encrypting the messages to be sent to that user.

(ii) A Private key : A private key is used for decrypting the message. It is kept secret.



Public Key Cryptography : Private key cryptography (as well as conventional cryptographic techniques requires the distribution of secret keys over the insecure communication network before secure communication can take place. This is called the *key distribution problem.* It is a boot-strap problem : a small secret communication (over an insecure communication network) is required before any further secret communication over the network can take place. A private courier or a secure communication channel is used for the distribution of keys over the network.

Public key cryptography solves this problem by announcing the encryption procedure E (and the associated key) in the public domain. However, decryption procedure D (and the associated key) is still kept secret. The crux of public key cryptography is the fact that it is impractical to derive the decryption procedure from the knowledge of the encryption procedure. The revolutionary concept was advocated by Diffie and Hellman [10]. Encryption procedure E and decryption procedure D must satisfy the following properties :

1. For every message M, $DE(M) = M$.
2. E and D can be efficiently applied to any message M.
3. Knowledge of E does not compromise security. In other words, it is impossible to derive D from E.

Public key cryptography allows two users to have a secure communication even if these users have not communicated before, because the encryption procedure used to encrypt message for every user is available in the public domain. If a user X wants to send a message M to another user Y, X simply uses Y's encryption procedure E_Y to encrypt the message. When Y receives the encrypted message $E_Y(M)$, it decrypts it using its decryption procedure D_Y.

Q. 3. (c) Write short notes on following :

(i) Architecture of distributed Event Notification.

Ans. In figure show an Architecture that specify the roles played by the object that participate in distributed event based system. The architecture is designed to decouple the publisher from the subscriber, allowing publisher to be developed independently of their subscriber.

- The main component is an event service that maintain a data base of published events and of subscriber interest. Event at an object of Interest Published at the event service.
- When the event occurs at an object of interest a notification sent to the subscribers to that type of events.

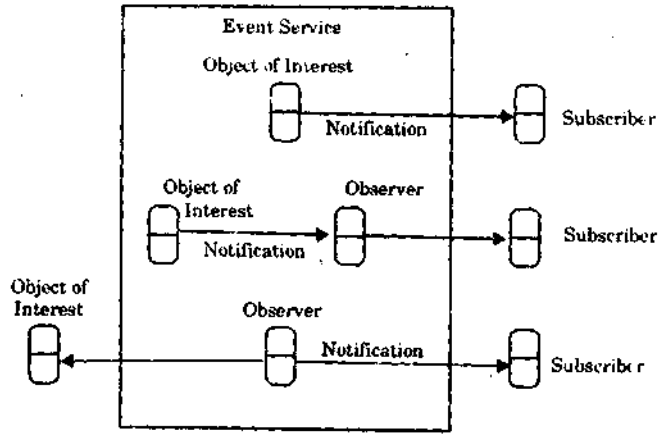


Fig. Architecture for distributed event notification

The role of participating object are as follows :

(a) **The object of interest :** This is an object that experience change of state as a result of its operation being invoked. Its change of state might be of interest to other objects. This description allows for events such as Person wearing an active badge entering a room.

(b) **Event :** An event at occurs at an object of interest as the result of completion of method execution.

(c) **Notification :** A Notification is an object that contain Information about an event. Typically it contain the type of the event and its attribute, which generally include the identity of the object of interest.

(d) **Subscriber :** A Subscriber is an object that has subscribed to same type of event in another object. It receives notification about such events.

(e) **Observer Object :** The main purpose of an observer is to decouple of an object of interest from its subscriber. An object of interest can have many different subscriber with different interest. For example, the subscriber may differ as to the type of events they are interested in, or those sharing the same requirement as to type may differ in the attribute values that are of interest.

(ii) Remote procedure call.

Ans. Remote Procedure call : Remote Procedure calls are the application through which the servers which are available at the remote servers in the forms of some subroutines and procedures are called at the certain work station through which the request is made.

For example : In yahoo website compose box while using graphical user interface, some fonts can be called at the client work station even that fonts are not available at the work station. Here certain RPCs are used in the coordinated network operating can be available at the work station.

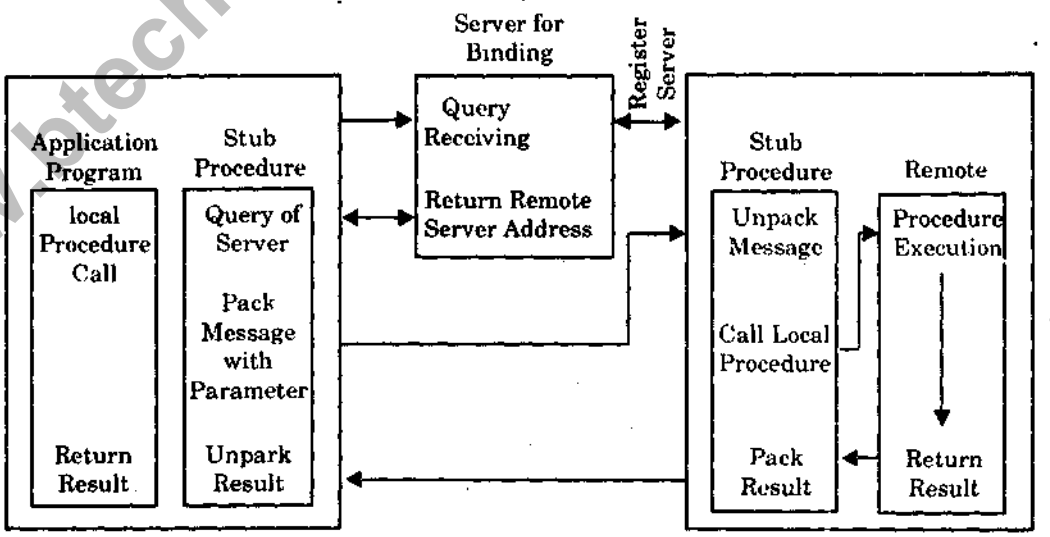


Fig. Generation of RPC in a Distributed System

4. Attempt *any two* parts of the following :

Q. 4. (a) Compare and contrast the methods of concurrency control for transactions. Explain the methods for concurrency control in distributed transactions.

Ans. Concurrency Control and Transactions

- At least one is a write.
- They both act on the same data.
- They are issued by different transactions.

$R_i(x)\ R_j(x)\ W_i(x)\ W_j(y)\ R_i(y)$ has $R_j(x)\ W_i(x)$ in conflict.

Definition : Two schedules are computationally equivalent if :

- The same operations are involved (possibly reordered).
- For every pair of operations in conflict, the same operation appears first in each schedule.

So, a schedule is serial sable if the schedule is computationally equivalent to a serial schedule.

Pessimistic and Optimistic Concurrency Control : Two operations conflict if they produce a change that is not possible in a serial transaction. If both operations are reads, they cannot conflict. Hence the possible conflicts are the read-write conflict and the write-write conflict. The basic techniques for concurrency control are synchronization by mutual exclusion or synchronization by timestamps. Concurrency control method can also be classified to pessimistc and optimistic approaches. In pessimistic approach, operations are scheduled so that conflict is not possible. In optimistic approach conflicts are allowed to happen but they are corrected afterwards.

Concurrency Control in Distributed Transactions : Each server manages a set of objects and is responsible for ensuring that they remain consistent when accessed by concurrent transactions. Therefore, each server is responsible for applying concurrency control to its own objects. The members of a collection of servers of distributed transactions are jointly responsible for ensuring that they are performed in a serially equivalent manner.

This implies that if transaction T is before transaction U in their conflicting access to objects at one of the servers then they must be in that order at all of the server whose objects are accessed in a conflicting manner by both T and U.

Q. 4. (b) What do you mean by two phase Locking ? How it is different from strict two phase Locking ? Explain.

Ans. Static Locking Scheme : According to static locking scheme, all the required data objects are locked before the execution of an action. All the data objects are predeclared. All these objects are unlocked by the transaction after the execution of all related actions. This type of locking is very simple, but useful for handling the concurrency in distributed database transaction. The drawback of this scheme is its requirement of previous knowledge about database.

Two Phase Locking Scheme : A method to prevent conflicts is to lock data items from other writes. Two-phase locking proceeds as follows :

1. When the scheduler receives an operation of T changing data item x, it checks if this operation conflicts some other operation that has already been granted a lock. In that case the operation is delayed. If there is no conflict, x is locked and the operation is passed to the data manager.

2. The lock for x is released when the operation setting the clock is completed.

- When a clock for transaction T is released, T is never more granted a lock.

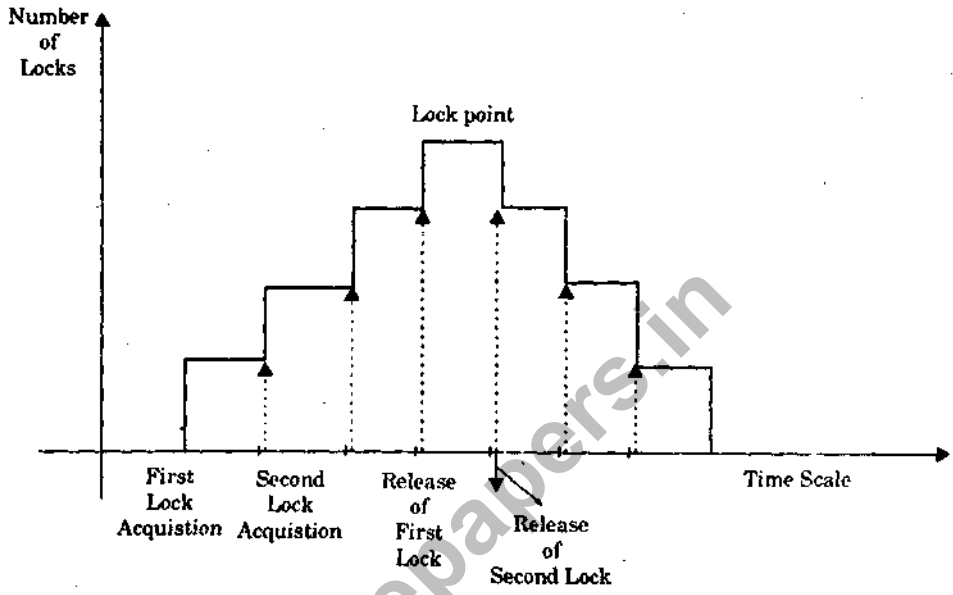


Fig. Two phase locking scheme

There is one lock point present in between these two phases. At the lock point stage, the transaction holds locks on all the needed data objects. Fig. shows a two phase locking scheme.

It has been proved that if all transactions use two-phase locking, all schedules interleaving them are serializable.

There are three ways to implement two-phased locking :

- In centralized 2PL a single site is responsible for granting and releasing locks.
- In primary 2PL each data item has a dedicated site to handle its locks.
- In distributed 2PL data is distributed and the schedule both takes care of locks and forwards the operation to the local data manager.

Q. 4. (c) Explain the following :

(i) Fault tolerant services.

Ans. For fault tolerant services, we provide a service that is correct despite of process failure by replicating data and functionality at replica managers. For the sake of simplicity we assume that communication remain reliable and that no positions occur. Each replica managers is assumed to behave according to the specification at the semantics of the objects it managers, when they have not crashed.

For example : A bank account specification would include an assurance that fund transformed between accounts can never disappear and that only deposits and withdrawls affect the balance at any particular account.

(ii) Highly available services.

Ans. Highly Available Services : Replication technique is used to make services highly available *i.e.*, clients access the services with reasonable response time. The several systems that provide the highly available services are :

1. Gossip
2. Bayou
3. Coda

5. Attempt *any two* parts of the following :

Q. 5. (a) Explain the term "routing". How routing problem can be classified ? Also Discuss the criterion for good routing algorithms.

Ans. Routing : A site is in distributed system is not generally connected directly to each and every site. A site send the packet to a destination directly to a subset of sites called neighbours. Routing is term to describe the decision procedure for which site is to be used to send the packet and on which way it will send to an ultimate destination. For this routing decision each and every site maintain information about routing table and topology of network. So the routing table must ensure the following type of information :

(a) **Table computation :**a The routing is computed when network is initialized and must be brought up to data if topology of network changes. So the information must be updated for source as well as destination.

(b) **Packet forwarding :** When a packet is to be sent to a particular destination so it is forwarded through a routing table.

Types of Routing : 1. Multicast Routing 2. Shortest Path Routing.

Criteria for Optimal Routing Algorithm : The network is represented as a graph where the nodes of the graph are the nodes of the network and there is an edge between two nodes if they are neighbours. The optimality of an algorithm depends on what is "best" path in the graph, there are several notions of what is "best" each with its own class of routing algorithms.

1. Minimum Hop : The cost of using a path is measured as the number of hopes (traversed channels or steps from node to node) of the path. A minimum hop routing algorithm uses a path with the smallest possible number of hopes.

2. Minimum Delay : Each channel is dynamically assigned a weight depending on the traffic on the channels. A minimum delay algorithm repeatedly revised the tables in such a way with a minimal total are always chosen.

3. Smallest Path : Each channel is statically assigned a weight and the cost of a path is measured as the sum of the weights of the channels in the path. A shortest part algorithm uses a path with lowest possible cost.

Q. 5. (b) (i) What are traversal algorithms ?

Ans. It is subclass of wave algorithm having following three properties :

(a) In each computation there is one initiation that starts the algorithm by sending out exactly one message.

(b) A process upon receipt of a message either sends out one message or decides.

(c) The algorithm terminates in the initiator and when this happens each process has sent a message at least once.

In each reachable configuration of a transversal algorithms there is either exactly one message in transit or exactly one process that has just received a message and not sent a message in reply.

Q. 5. (b) (ii) Explain Tarry's algorithm for traversing connected networks.

Ans. Tarry's Graph Traversal Algorithm : Graph traversal is a problem in which an initiator node sends out a token, each noninitiator receives the token and forwards to a neighbour, until the token returns to the initiator.

A correct traversal algorithm must guarantee that (1) the token visits every node in the graph and (2) eventually returns to the initiator, which signals terminates.

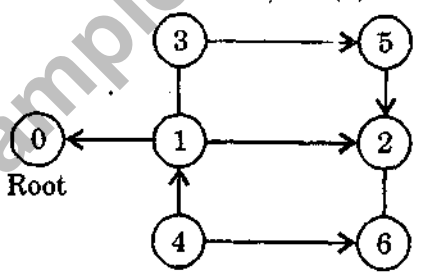
In 1895, Tarry proposed an algorithm that solves the traversal problem, and generates a spanning tree of the graph in that process. It is the oldest known traversal algorithm, and hence an interesting candidate for study. Define the *parent* of a node as one from which the token is received for the *first time*. All other neighboring nodes will simply be called *neighbors*. the root is the initiator, and it does not have a parent. The following two rules define the algorithm.

Rule 1. Send the token towards any neighbor exactly once.

Rule 2. If Rule 1 cannot be used to send the token, then send the token to its parent. When the token returns to the root, the entire graph has been traversed.

In the graph of Fig., a possible traversal route for the token is 0 1 2 5 3 1 4 6 2 6 4 1 3 5 2 1 0. Each edge is traversed twice, once in each direction, and that the directed edges connecting each node with its parent form a spanning tree. Tarry's algorithm does not guarantee that the traversal will follow the DFS order.

To prove that Tarry's algorithm is a traversal algorithm, we need to show that (i) a least one of the rules is applicable until the token returns to the root, and (ii) eventually every node is visited.



Q. 5. (c) Write short notes on following :

(i) CORBA services.

Ans. The CORBA has provide the following services :

1. Naming Service : It uses a naming context to lookup a name. The naming contexts can be linked, so that a name actually points to another naming context, allowing hierarchical names.

2. Event and notification Service : It provides for event management, including pattern matching for notification.

3. Security Service : It provides for management of authentication and access control, audit trails etc.

4. Trading Service : It allows services to be located by description, rather than name.

5. Transaction and Concurrency Control Service : It implements transactional mechanisms to provide concurrency control and ACID semantics to operations.

6. Persistent Object Service : It allows objects to be stored in passive form when not required and activated on demand.

(ii) Deadlock free packet switching.

Ans. Definition : Messages traveling through a packet switched communication network must be stored at each node before being forward to the next node on path of their destination. Each node of the network reserve some buffer space for this purpose, as the amount of buffer space is finite

in each node, situation may occur where no packet can be forwarded because all buffer in the next node are occupied.

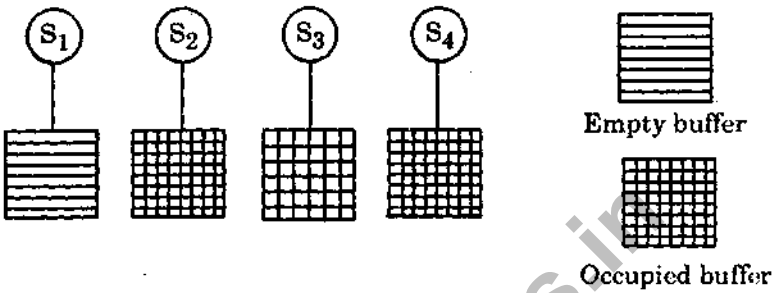


Fig. Dead lock free packet switching

Each of four node has buffer has B buffer, each capable of containing exactly one packet node s has sent B packet with destination v to t and node v has sent B packet with destination s to u. All buffer in u and v are now occupied and consequently none of the packet stored in t and u can be forwarded toward its destination.

Situation where group of packet can never reach their destination because they are all waiting for the use of a buffer currently occupied by another packet in the group are referred to as stored and forward deadlocks. An important problem in the design of packet switching network is how to deal with store forward deadlock.