Name - Suyash Vairagade
Roll No - 19115093

Sub - Distributed System
Branch & Sem - CSE, 7th Sem

Assignment - 1 & 2

Set - 6

Q.3)

Ans:) The messages that are to be forwarded to the migrant process's new location can be clarified into the following -

(a) Type 1 - The message received at the source node after process's execution stopped and did not started on destination node.

(b) Type 2 - The message that is received at the source node after the process's execution has started on its destination node.

(c) Type 3 - The message that are to be sent to the migrant process from any other node after it has started executing on the destination node.

Best message-forwarding mechanisms for following scenarios are -

(i) Transparency is the main goal -

Origin site mechanism can be used for it because process identifiers of these system has the processes origin site embedded on it. Therefore, a process's current location can be simply obtained by consulting its origin site. Link traversal mechanism can also be used for this case. Type 1 message can be

redirected using a message queue for the migrant process on its source code, and to redirect type 2 and type 3, a forwarding address known as link is left at the source node pointing to the destination node of migrant process. These two mechanisms are suitable when transparency is the main goal.

(ii) Reliability is the main goal -

If reliability is the goal then link update mechanism can be used because in this, processes communicate via location-independent links, which happened with the help of kernels controlling all the migrant processes communication partners. The messages sent to the migrant process on any of its links will be sent directly to the migrant processes new node.

(iii) Performance is the main goal -

If performance is the main goal, then origin site mechanism, link traversal mechanism and link update mechanism all can be used because they provide good performance. But in these three links, link update mechanism is best to use because link traversal has less reliability and efficiency in comparison and origin site mechanism is also not reliable and there is a continuous load on the migrant process' origin site even after the process has migrated from that node. So, its best to use link update mechanism.

(iv) Simple implementation is the main goal -

If simple implementation is the main goal then we can use mechanism of resending the message. In this mechanism type 1 and 2 are returned to the sender as not deliverable or simply dropped with the assurance that the sender of the message is storing a copy of data and is prepared to retransmit it and type 3 message are sent directly to the process's destination node in this mechanism.

Q. 2)

Ans.) Since load sharing algorithms are normally interested only in the busy or idle states of a node. Most of them employ the all-or-nothing strategy. This strategy uses the single threshold policy with the threshold value of all the nodes fixed at 1. That is, a node become a candidate for accepting a remote process only when it has no process and a node become a candidate for transfering a process as soon as it has more than one process.

But this strategy is not good in the sense that a node that becomes idle is unable to immediately acquire new processes to execute even though processes wait for service at other nodes, resulting in a loss of available processing power in the system.

To avoid this loss, anticipatory transfers to nodes that are not idle but are expected to soon become idle are necessary.

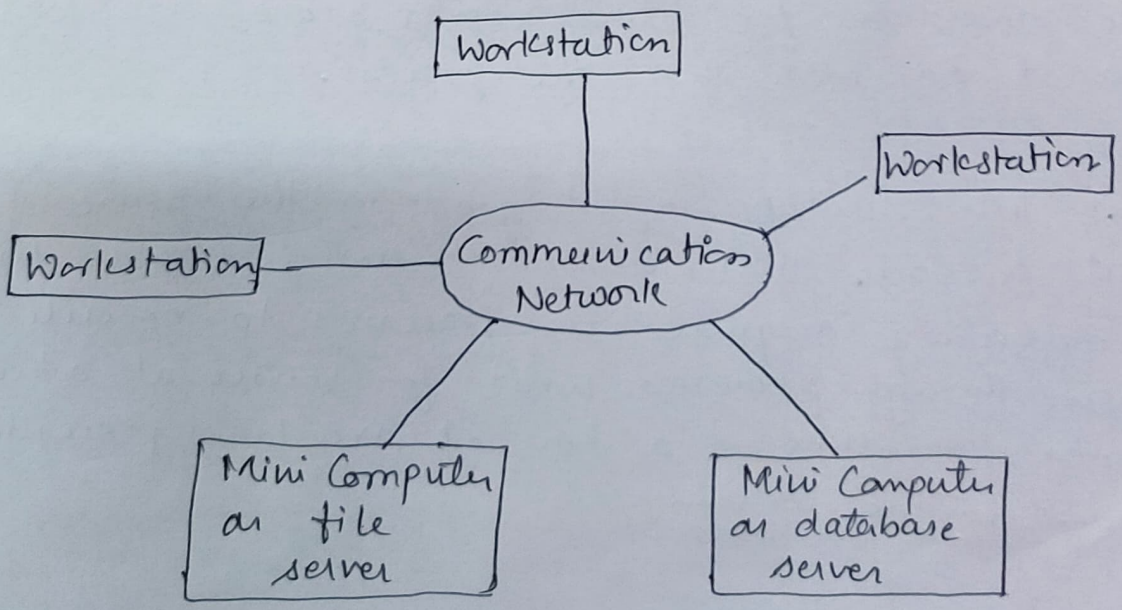
Thus, to take care of their loss, some load sharing algorithms use a threshold value of 2 instead of 1 so that after completing 1 processor node can acquire another immediately.

Q.1)

Ans:)

(a) For Workstation - Server Model -

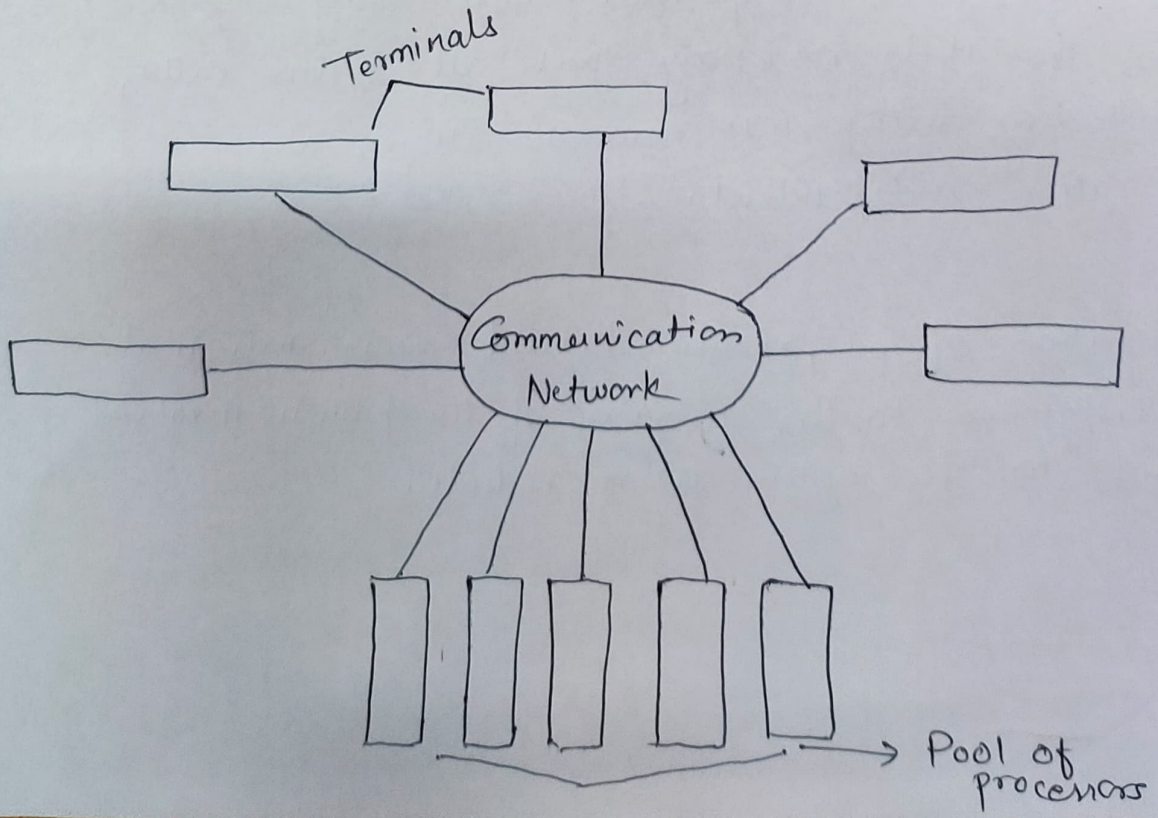
A workstation - server model contains workstation with disk and without disks and they also contains minicomputers which provide specific services to the workstation.



So for Workstation - Server model selfish priority assignment rule can be used because each workstation should provide the services to its user first and then it will do the remote process. All mini computers will have requests from the remote clients so there is no process for them, so selfish priority assignment will do fine for the workstation - server model.

(b) Processor - pool Model -

In Processor - poll model there is a pool of processors which are used to provide services to the user. It is good when user need a higher processing power for a short time processor pool is shared between the users.



In the processor pool model there is no concept of home machine so user is logged in to the system whole. So, we don't have to provide any kind of priority assignment rule and equal priority is given to the process of all the users.

Q.4)

Ans.) The process migration facility of a distributed system does not allow free migration of processes from one node to another but has certain restrictions regarding which node's processes can be migrated to which other nodes of the system. The reasons behind imposing such a restriction are -

(1.) A migrated process should not continue to depend upon its originating node after being migrated. For this it is necessary that all system calls, including interprocess communication, are location independent.

(2.) Migration of a process should cause minimal interference to the progress of the process involved and to the system as a whole.

(3.) A migrated process should not in any way continue to depend on its previous node once it has started executing on its new node since, otherwise, the following will occur.

(a) The migrated process continues to impose a load on its previous node, thus diminishing some of the benefits of migrating the process.

(b) A failure or reboot of the previous node will cause the process to fail.

Q.5)

Ans) Advantages of using full-file caching -

(1.) It provides fewer access to file server and it also reduces the network traffic.

(2.) It is very good for small-sized files.

(3.) It supports the heterogeneous environments.

Disadvantages of using full-file caching -

(1.) It is not suitable for diskless workstations.

(2.) Full-file caching is not suitable for large-sized files.

(3.) The network bandwidth and storage space of full-file caching are wasted if only a small part of the file is needed.

The advantages of block caching models for the data-caching mechanism of a distributed file system are -

(1.) The storage space in block caching models are saved.

(2.) It is suitable for diskless workstations.

The disadvantages of block caching models for the data-caching mechanism of a distributed file system are -

(1.) For large files in a block caching model, there is a need to make multiple requests for accessing the same file.

(2.) It increases the network traffic.

Q.6)

Ans.a) In Session Semantics multiple clients are allowed to perform both read and write operations concurrently on the same file. In this case, each client maintains its own image of the file. When a client closes its session, all other remote clients who continue to use the file are using a stale copy of the file. Client Disk is the suitable option for cache location because of large storage as in session semantics the whole file are carried over. Also, if the cached data is present on client disk, the data is still present even after crashes and no need to fetch it again from the server's node.

Ans.b) Session Semantics should be used only with those file systems that use the file level transfer model. In these models, the whole file is moved. These models are to be used because coupling the session semantics with caching parts of file may complicate matters, since a session is supposed to read the image of the entire file that corresponds to the time it was opened.

Ans. c) Write on close -

It is the most suited method for session semantics. In this method, the modification made to a cached data by a client are sent to the server when the corresponding file is closed by the client. However, it does not help much with reducing network traffic for those files that are open for very short period or are rarely modified. Furthermore, the close operation takes a long time because all modified data must be written to the servers before the operation completes. Therefore, this policy should be used only in cases in which files are open for long periods and are frequently modified.

Ans. d) For session semantics we can have the server store a list of client who have the file's data cached, whenever a server receives a request to close a file that has been modified, it notifies all the clients having that file data in their caches to discard their cached data and consider it invalid. Clients having the file open at that time discard their copy when the current session is over. Other clients discard their copies at once. In this method, the server need not be informed about opens of already cached files. If the client machine crashes, it is assumed that all its local files may be inconsistent.

Therefore, upon recovery from a crash, it generates a cache validation request for each file that is cached on its local disk.

Q.10)

Ans.) Yes, it is possible to have system-specific implementations of CORBA object references while still being able to exchange references with other CORBA-based systems because CORBA enables collaboration between systems on different operating systems, programming languages and computing hardwares.

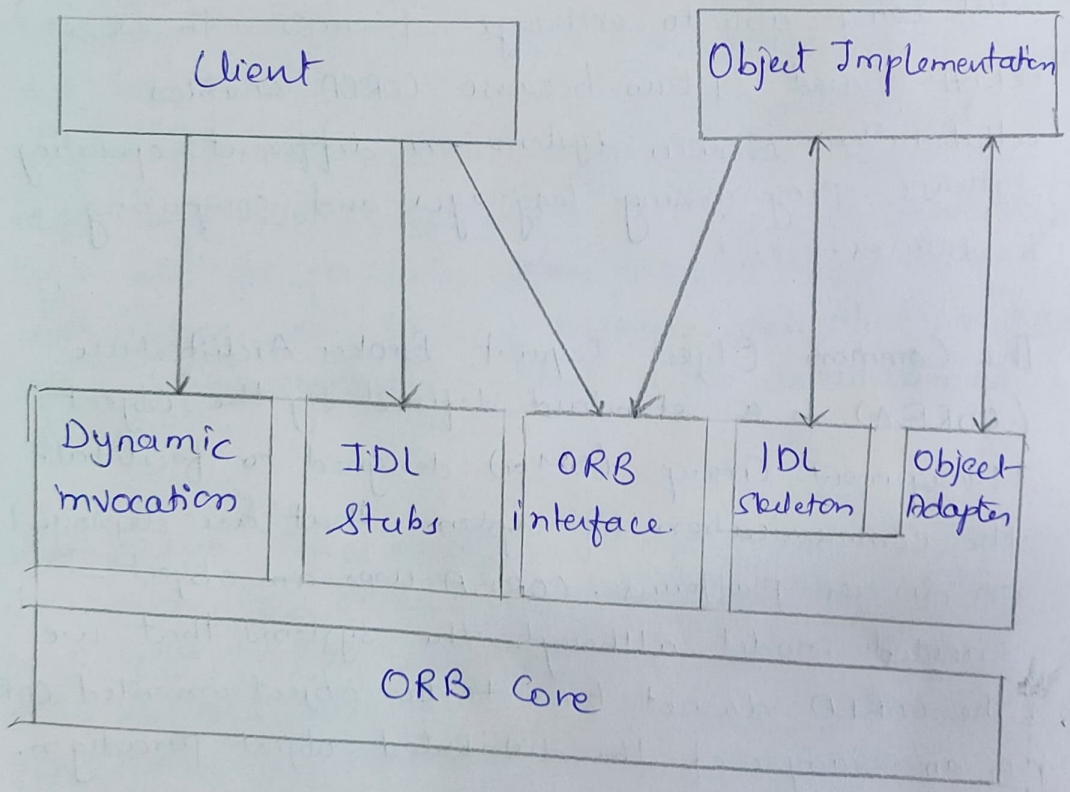
The Common Object Request Broker Architecture (CORBA) is a standard defined by the Object Management Group (OMG) designed to facilitate the communication of systems that are deployed on diverse platforms. CORBA uses an object-oriented model although the systems that use the CORBA do not have to be object oriented CORBA is an example of the distributed object paradigm.

The CORBA specification dictates there shall an ORB through which an application would interact with other objects. This is how it is implemented in practice-

(1.) The application initialises the ORB, and accesses an internal object adapter, which maintains things like reference counting, object (and reference) instantiation policies, and object lifetime policies.

(12)

(2.) The object adapter is used to register instances of the generated code classes. Generated code classes are the result of compiling the user IDL code, which translates the high level interface definition into an OS- and language specific class base for use by the user application.



CORBA Architecture.

Q. 9)

Ans.) The location for storing the login program and password file for the password mechanism can be different for different distributed system models. Best way to store login system file for different models are -

(a.) Workstation - Server model -

Workstation - server model with each workstation having a small hard disk about 20 megabytes capacity → for this we have two option. One is that we can make a minicomputer dedicated for the login file server and user can request to it for login and another method is that we can store the login file for particular ~~store~~ user on their workstation for efficiency because each of them has a amount of storage in which login detail can be saved.

(b.) Workstation - Server model in which some of the~~m~~ workstation are diskless and others have a small hard disk of about 20 megabytes capacity - for this ~~best~~ best solution is to make a dedication file server for password mechanism. We can also do that workstation that has disk can store login info of user to reduce the overhead but the former one is easy to implement in this case.

(c) The distributed system is based on the processor-pool model. Assume that any user is free to use any of the user terminals or workstation. In this scenario best is to provide access of login file to all the processors and user send their login request to some processor and any of the processor that are assigned to the request can verify the login of user because all of the processor has the access to login the file system.

Q.8)

(Ans-) Optimistic concurrency control is based on the observation that access conflicts in concurrency executing transaction are very rare. Therefore in this approach, transaction are allowed to process uncontrolled up to the end of the first phase. However, in the second phase, before a transaction is committed, the transaction is validated to see if any of its data items have been unchanged by any other transaction since it started. The transaction is committed if found valid otherwise aborted.

Let there be any two transaction T_i and T_j. The validation phase for T_i checks that for all transaction T_j one of the following below conditions must held by being validated or pass validation phase.

(i) Finish (T_i) < Start (T_j) : It means T_i is older transaction and it gets finished before T_j start. (no overlap)

(ii) Finish (T_i) < Validate (T_j) : This ensures actual write by T_i and T_j will not overlap.

(iii) Validate (T_i) < Validate (T_j) : It ensures that T_i has completed read phase before T_j completes read phase.

(9) Let us consider two transactions T_i and T_j such that TS(T_i) < TS(T_j) so that validation phase succeeds in schedule. A final write operation to the database are performed only after validation of both T_i and T_j.

Ex- Schedule A

T_i	T_j
R(x) // 12	
	R(x)
	x = x-10
	R(y) // y = 15
	y = y+10
	R(x)
<validate>	
print (x+y)	<validati>
	W(x)
	W(y)

Schedule A is succesfully validated as it satisfies condition validate (T_i) < Validate (T_j) and thus succesfully committed.

(b) Let us consider two transactions T_i and T_j such that $TS(T_i) < TS(T_j)$. In the transaction final write operation is performed before validation of both transaction T_i and T_j.

Ex- Schedule A

T_i	T_j
R(x) // 12	
	R(x)
	x = x-10
	R(y)
	y = y+10
	R(x)
<validate>	
W(x)	
W(y)	
	<validate>
	print (x+y)

Schedule A, Validation is unsuccesful an write operation is before the validation of T_i and T_j and thus the transaction is aborted.

Q.7)

Ans) A transaction is a computation consisting of a collection of operation that takes place indivisibly in there of failures and concurrent computations.

All are performed successfully or none of their effects prevail and other process executing concurrently cannot modify or observe intermediate state of computation.

Transaction have following essential properties -

1) Atomicity - This ensures to outside world, all operation of the world to be performed as indivisibly.

2) Serializibility - It ensures no concurrently executing Tx^n interference.

3) Premance - If tx^n becomes success, the result of its operation becomes permanent & cannot lost.

factors that threaten Atomicity are -

→ With respect to failure
→ With respect to concurrent access.

With respect to failure -

Failure atomicity ensures that if a transaction work is interrupted by failure, any partially completed results will be undone.

Failure atomicity is also known as all or nothing property because a tx^n is always performed either completing or not at all.

With respect to concurrent access -

Concurrent Atomicity ensures that while a tx^n is in progress, other processor executing concurrently with transaction cannot modify or observe intermediate states of transaction. Only the final state becomes visible to other processes after transaction complete.

Concurrency Atomicity is also known as consistency property because transaction moves the system from one consistent state to another.

Atomic Commit & Abort -

The atomic commit should meet the following requirements -

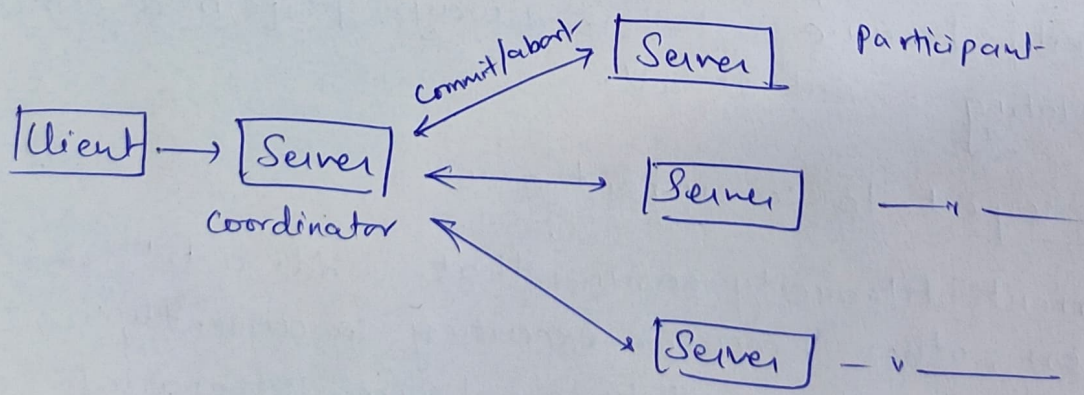
(i) All participant who make the choice, reach the same conclusion.

(ii) If any participant decides to commit then all other participants have voted yes.

(iii) If all participants vote yes & no failure then all participants decide to commit.

Distributed One Phase Commit -

A one phase commit protocol involves a coordinator who communicates with servers & performs each task regularly to inform them to perform or cancel actions in txn.



Distributed Two Phase Commit -

Two phase to work -

Phase 1 → Voting -

→ Prepare message is sent to each worker (participant).

→ Coordinator must wait until a response whether ready or not ready is received from each worker or a timeout occurs.

→ Worker must wait until coordinator sends the "prepare" message.

→ If a Txn is ready to commit then a "ready" message is sent to coordinator.

→ If a txn is not ready to commit then "no" message is sent to coordinator & results in abort of transaction.

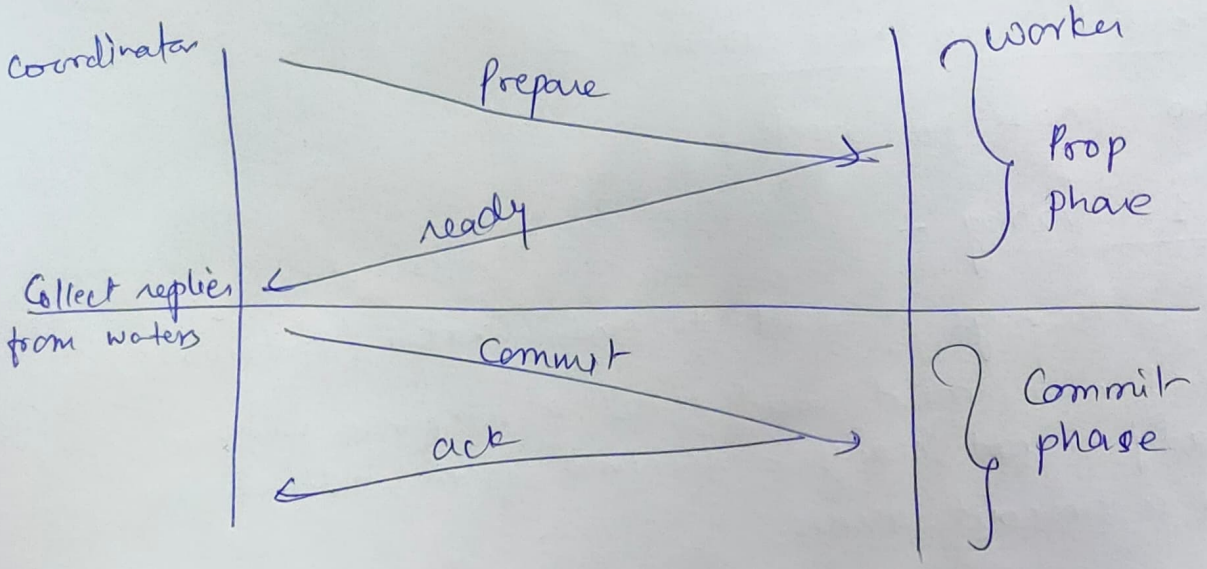
Phase 2 → Completion of Voting Result -

→ In this phase, coordinator checks "ready" message. If all sent "ready" then only "commit" is sent to each worker else "abort" is sent.

→ Now wait for acknowledgement until received from each worker.

→ Here in this place workers wait until coordinator sends "commit" or "abort" message then act according to menage received.

→ At last workers send an acknowledgement to coordinator.


coordinator
Prepare
ready
Collect replies from waters
Commit
ack
worker
Prep phase
Commit phase