Exam Roll - CSE 218056

Class Roll - 001710501029

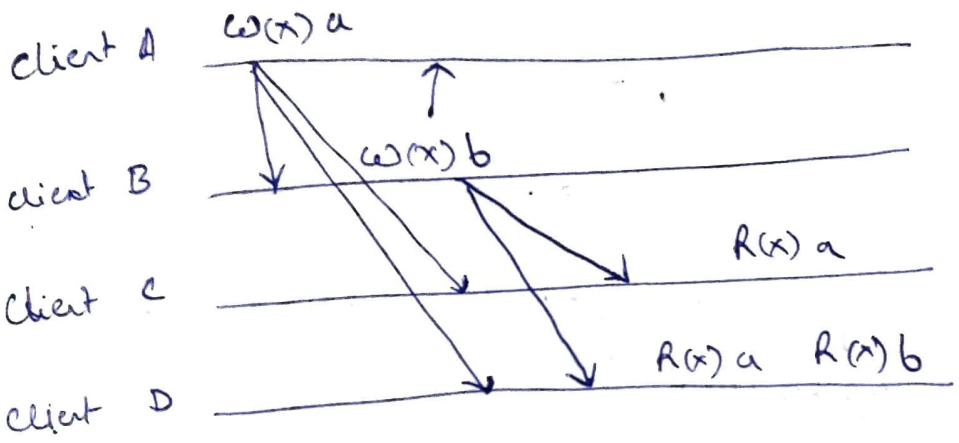
Semester - 2

Sub - Distributed Computing.

6. Strict consistency is the consistency model that requires all nodes to return only the most recent write on an item. This is how a system will act if it was running on a non-distributed setup.

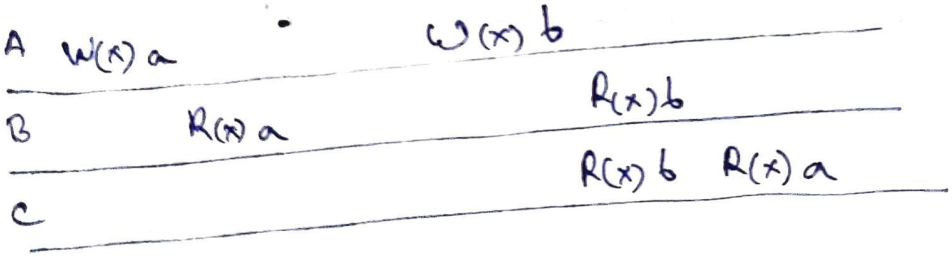
The interpretation on "most recent" will be inconsistent among different nodes of a distributed system because it requires the notion of an absolute global time, and achieving that is not possible. Also it requires that the writes are instantenous to all nodes, which again is physically impossible.

Linearizable consistency or sequential consistency model is a consistency model which drops the absolute global time requirement of the strict consistency model. It requires that all the nodes execute the operations in the same order. The order of executions of is decided according to the operation timestamps taken from involving client's loosely synced clock.



In the above diagram in all the nodes W(x) a occurs before W(x) b. Therefore it is sequentially consistent.

Given data store diagram :-



This given data store diagram is sequentially consistent. Both the writes occur on the same node. Therefore when they are propagated they will always be propagated to other nodes in the same order.

Release consistency model is a consistency model in which a client while entering a critical section in a node aquires a lock (a call to the aquire()). When one node has entered a critical section, no other nodes can enter the same critical section. Thus the data is upto date while entering.

While leaving the client calls the release() operation. During this operation all the updates are propagated to all other nodes.

The only difference between Release consistency & lazy consistency model is that in lazy consistency updates are propagated while the aquire() operation is performed on a node. This ensures that if a client enters a critical section multiple times, it requires only a single propagation update to all other nodes.

fig: release consistency model.

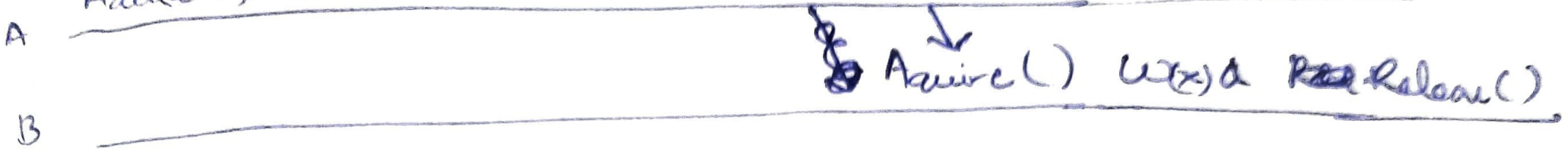


fig: Lazy consistency model.

8b

1. Disadvantages of of distributed computing systems :-

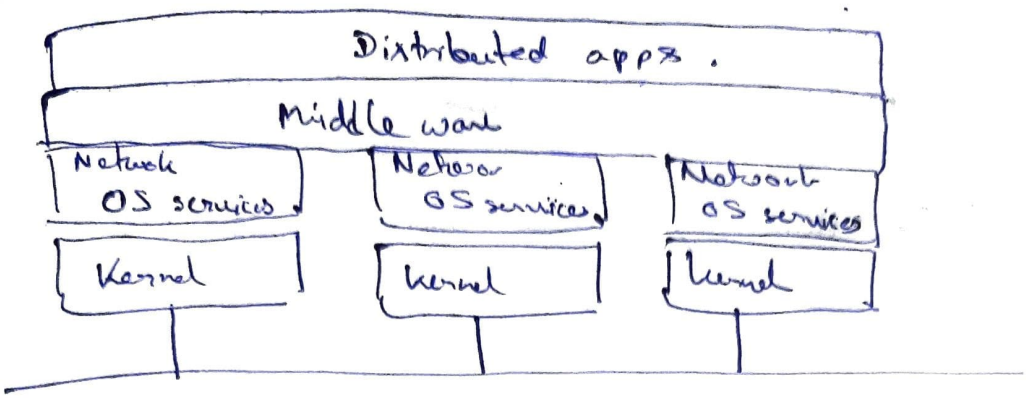
a) The nodes in the system are connected through a network. Therefore a reliable network is required. Also any error in networks slows down the system due to lag in communication between nodes. This also means that network latency is inavoidable.

b) Distributed systems require special software implimentations which are in general more complex than non-distributed counter parts.

c) Consistency among all nodes and syncronization among all nodes is a challenging task. This is a major obstacle for almost all distributed systems.

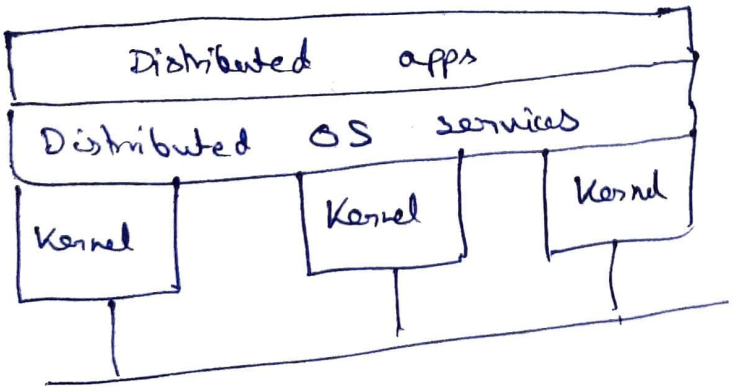
d). With a more complex algorithms, & more complex software etc. security & integrity also becomes more diffiant to achieve.

Middle ware programs are responsible for providing a single system image to the user.



The main objective of middle ware is to ~~achieve or~~ create ~~system independent~~ an enviroment for distributed application that hide the underlying ~~heterogenet~~ heterogeneity of ~~the~~ underlying machines.

In some cases the OS itself may also perform the task of a distributed middleware service. These distributed operating systems are built to provide services for distributed applications.

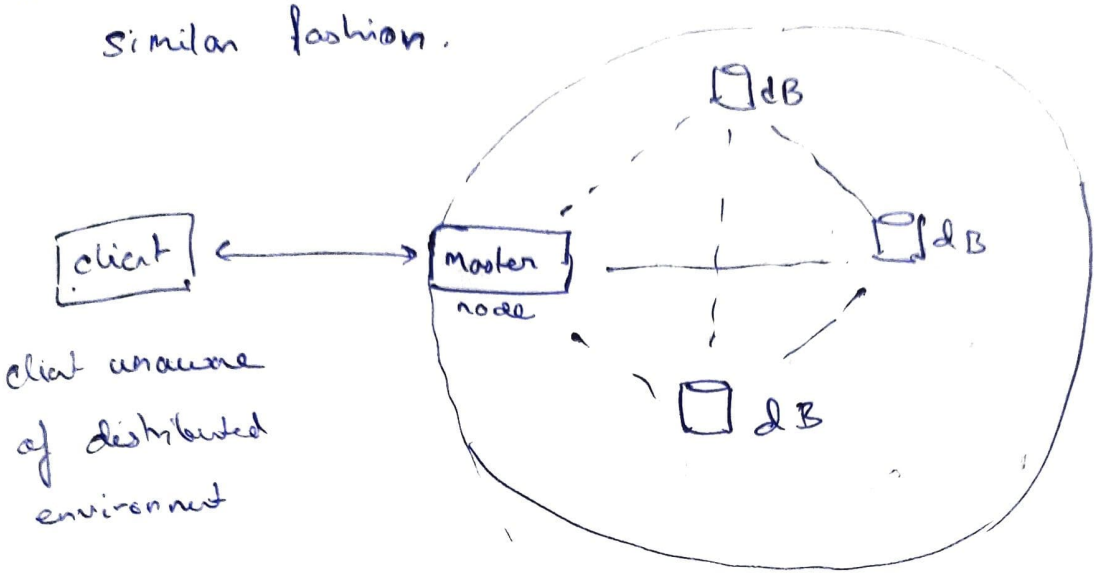


Various distributed applications ensure load balancing in various ways. In distributed database systems like DHT-CHORD, load balancing systems are implemented to ensure that all the nodes in the system store an approximately equal number of entries. If some node exits or fails, the data entries are reshuffled to ensure that the load is balanced. Similar operations are done when a new node enters the network.

In cluster computing systems, the RMS (Resource Management Scheduling) is implemented to ensure all the nodes in the cluster are load balanced.

Access transparency :-

Access transparency ensures that local & remote resources are accessed in similar fashion.



distributed data store.

Location transparency ensures that client are unaware of the location of the resource. like in the above diagram the client doesn't have to know where in the network the required data is stored.

Migration transparency ensures that resources can migrate with out a change in name or change in the way the client accesses them. Like in the above diagram if all the data in a DB node are transferred to other nodes, the client remains unaware and unaffected.

3.

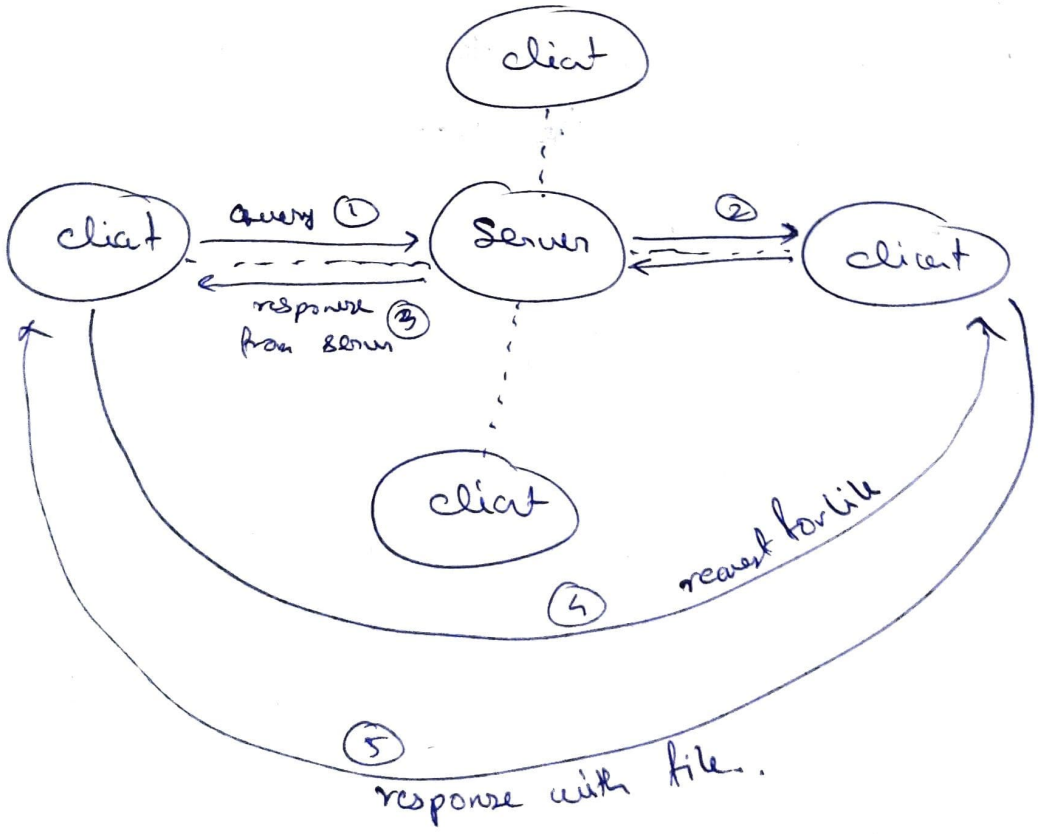
Napster :-

→ client searches for a certain file, it contacts the central server.

→ The server identifies which other client in the network has the required file and responds with the info of the client.

→ The client then requests the supplier client for the file.

→ Both requester and supplier clients update their statuses to the central serve.

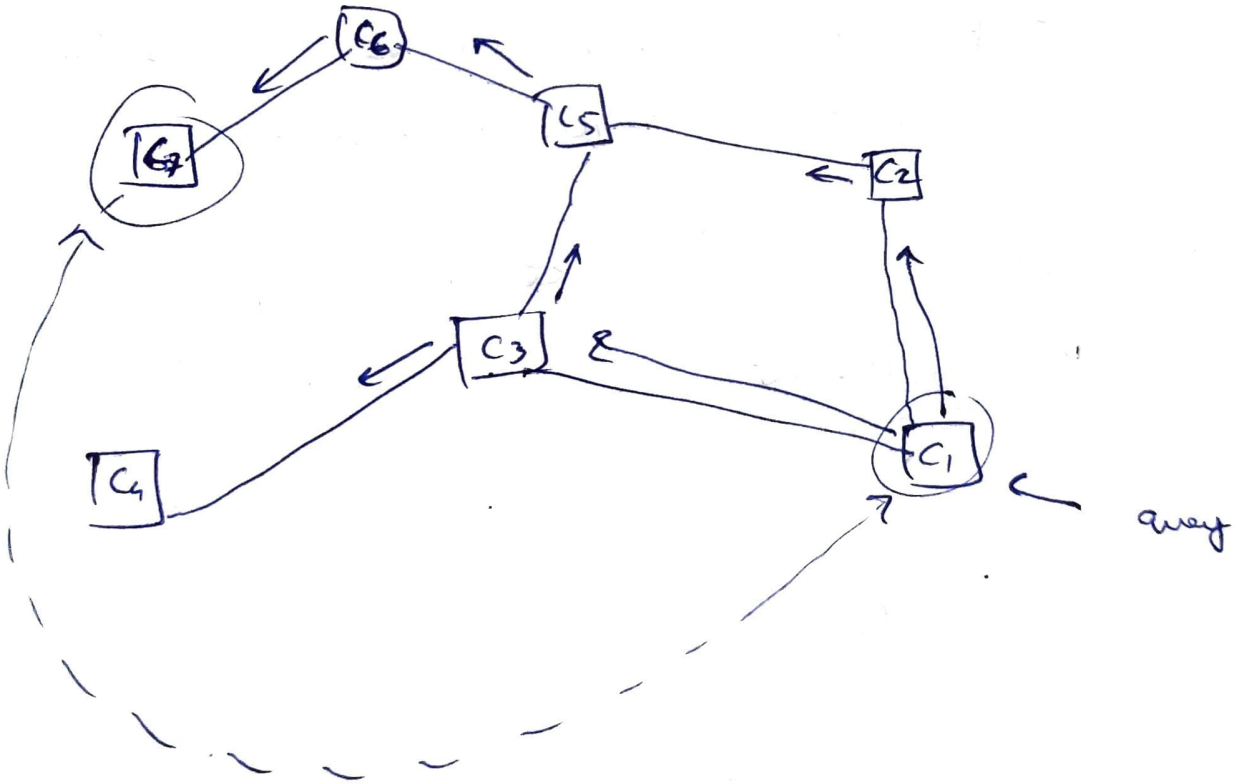


Gnutella :-

→ If a client requires a certain file, it sends request all the clients that it has connection to in the network.

→ The other clients further passes the message to their clients in the network till a client is found who has the required file.

→ Once the client who has the file is located, it contacts the requesting client & file transfer occurs between them.



(The file transfer btwn the c1 & c7 may occur on the same route that the request arrived on or on a different route).

KaZaA :-

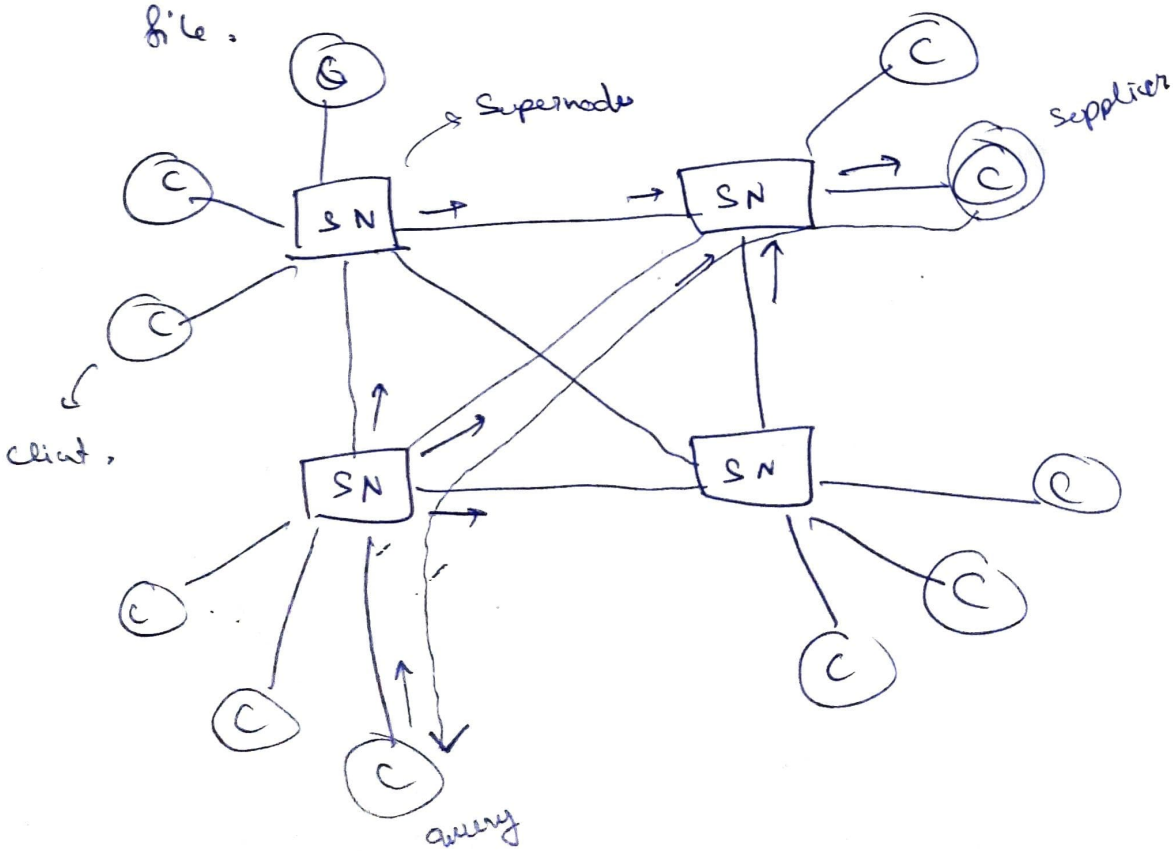
→ A client contacts its allocated ~~supernode~~ supernode for a file.

→ The super node contacts other supernodes asking for the file.

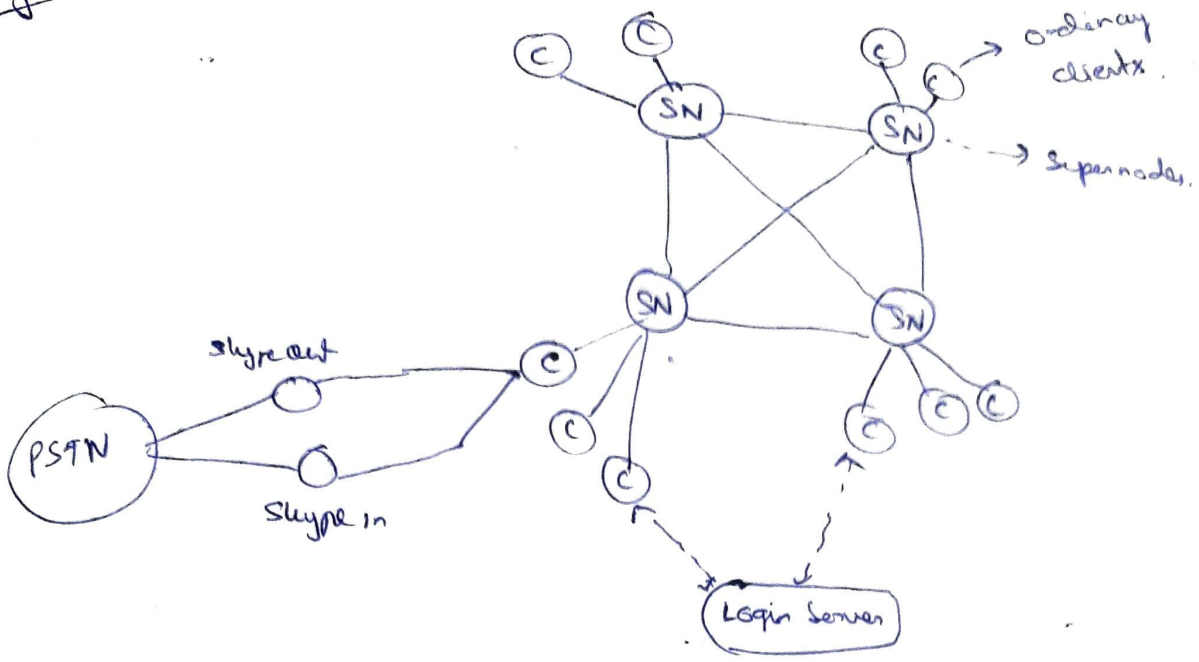
→ Each contacted supernode ~~asks~~ inquires its connected clients about the file.

→ If a file is found, the supernode sends the info to the requester supernode who intern sends the info to the requester client.

→ The client after receiving the info about the supplier client can contact the supplier client for the file.



Skype :-



→ Skype clients : nodes on user ends that starts or receives calls.

→ Supernode Nodes :- More powerfull nods then normal clients, that handle connection of clients over other supernodes.

→ login server :- Server that provides auth servis

→ Gateways (skypein & skypeout) :- servers that provid PC to PSTN & PSTN to PC bridging.

→ NAT :- Network Address translation

→ TCP signaling of both types i.e. UDP & HTTP is used for data transfer.

4.

A) Functions which are essential to achieve high availability in cluster computing environment are:-

a) Single I/O Space (SIOS):- any node can access any resource on the cluster without knowledge of physical location.

b) Single Process Space (SPS):- Any process created on any node is a cluster wide process and inter node communication occurs through signals or pipes.

c). Checkpoint and Process Migration :-

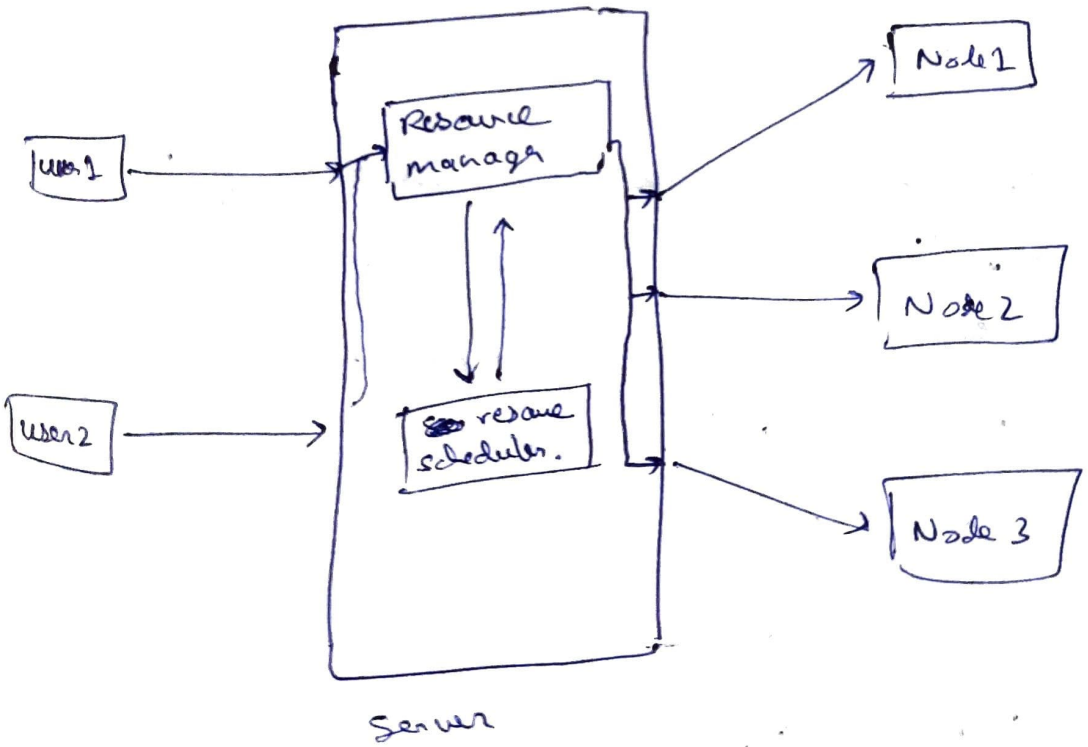
→ Checkpoint functionality saves a stable state of a process to the disk in case a rollback is required.

→ Process migration is required for load balancing among cluster nodes or in case a node fails.

RMS :→ Resource Management and Scheduling is the act of distributing jobs among cluster nodes in order to maximize throughput.

Resource manager :- locating & allocating computational resource, auth & creation & migration of processes.

Resource scheduler :- Queuing applications, resource locations and assignment, it instructs resource manager what to do and when.



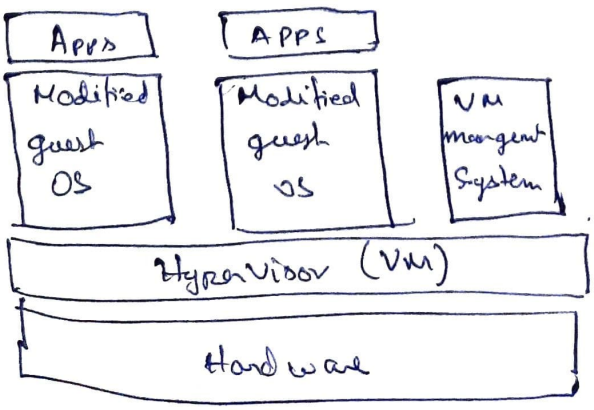
8) Types of full virtualization :-

i) Software enabled virtualization — The Host OS runs on the hardware platform & virtual machines run on top of the OS as separate software entities. The.

ii) Hardware enabled virtualization — The virtualization layer can directly access physical hardware and software virtual machines run on top of the virtualization layer.

Paravirtualization :-

A hypervisor runs on top of the hardware. The VM doesn't simulate hardware but provides an API for modifying the guest OS. The modifications allow the several guest OSes to share hardware resource.



Ex- XEN, KVM etc.

Benfits of para-virtualization over full-virtualization

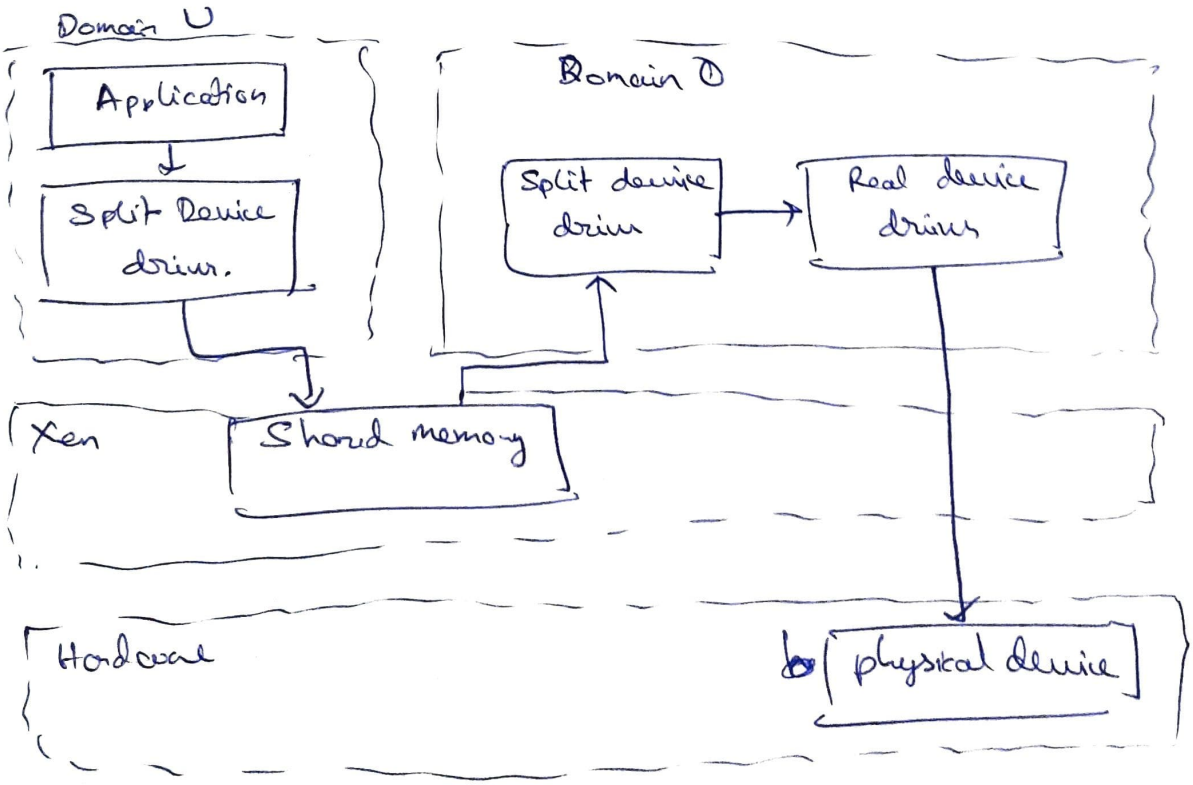
→ full virtualizon uses the concept of traps for execution of privileged instructions. Traps are expensive operations & reduce performance. Para-virtualization removes the need for traps by implementing hypercalls.

→ The hypervisor contains the device drivers in full virtualization, making in difficult for the user to install and use custom or other device drivers.
On the other hand para virtualization allows the use of modified drivers.

→ In para-virtualization both the PV OS and virtualizon layer runs at ring 0, reducing the need of context switching. and thus improving the performance.

XEN I/O Split Device Drive Model:-

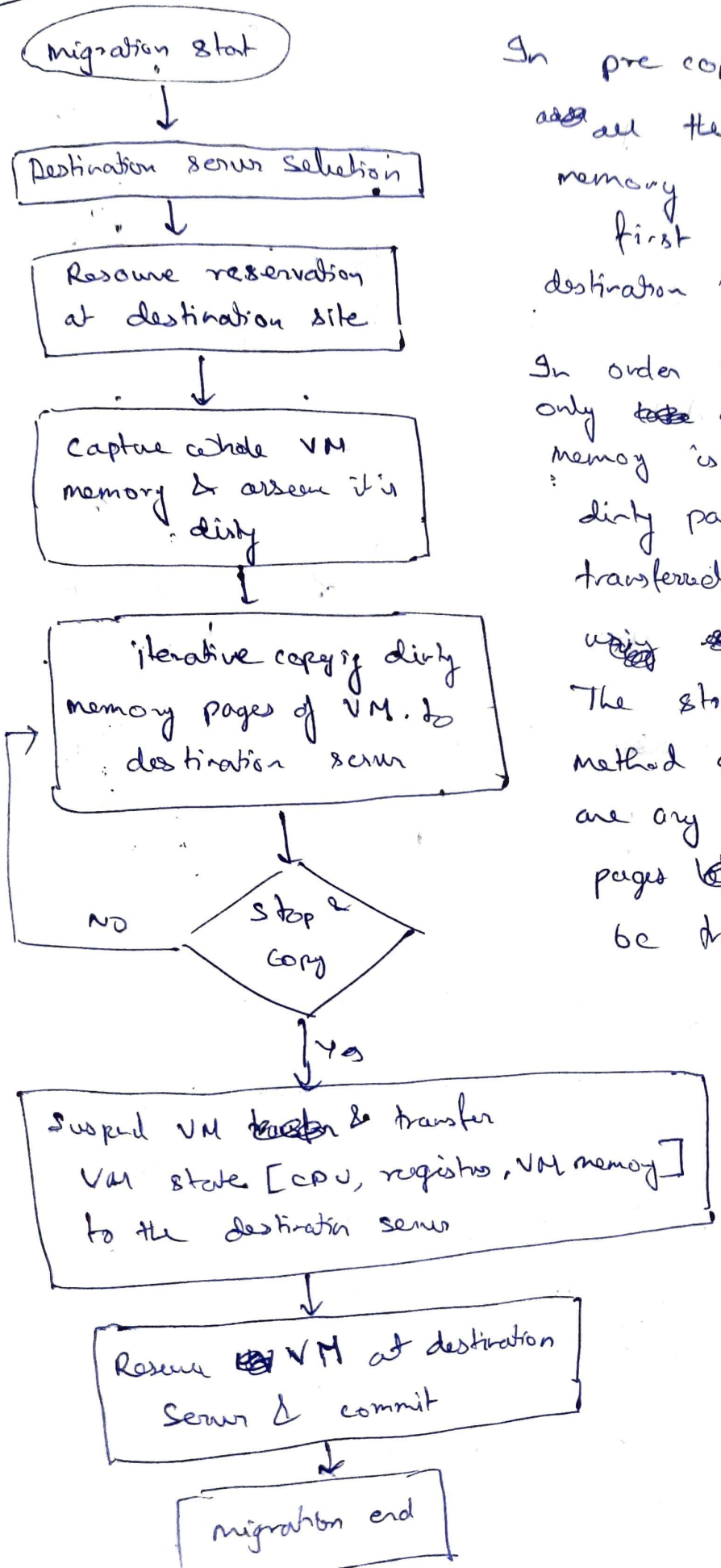
The device drivers are split into domains. Domain U is the device driver assumed by the guest OS & Domain 0 which is control domain of the VMs. When a guest application makes a request in the shared memory segment via its driver, domain 0 drivers reads the request. The request is then modified in accordance with the host device and the request is forwarded to the physical hardware. The response from the device follows a similar path. The domain 0 drivers receive the response from the drivers and puts it in the shared memory for the domain U drivers to access it.



7. Xen I/O virtualization technique of XEN:-

(refer to last part of 4.)

Pre-copy:-

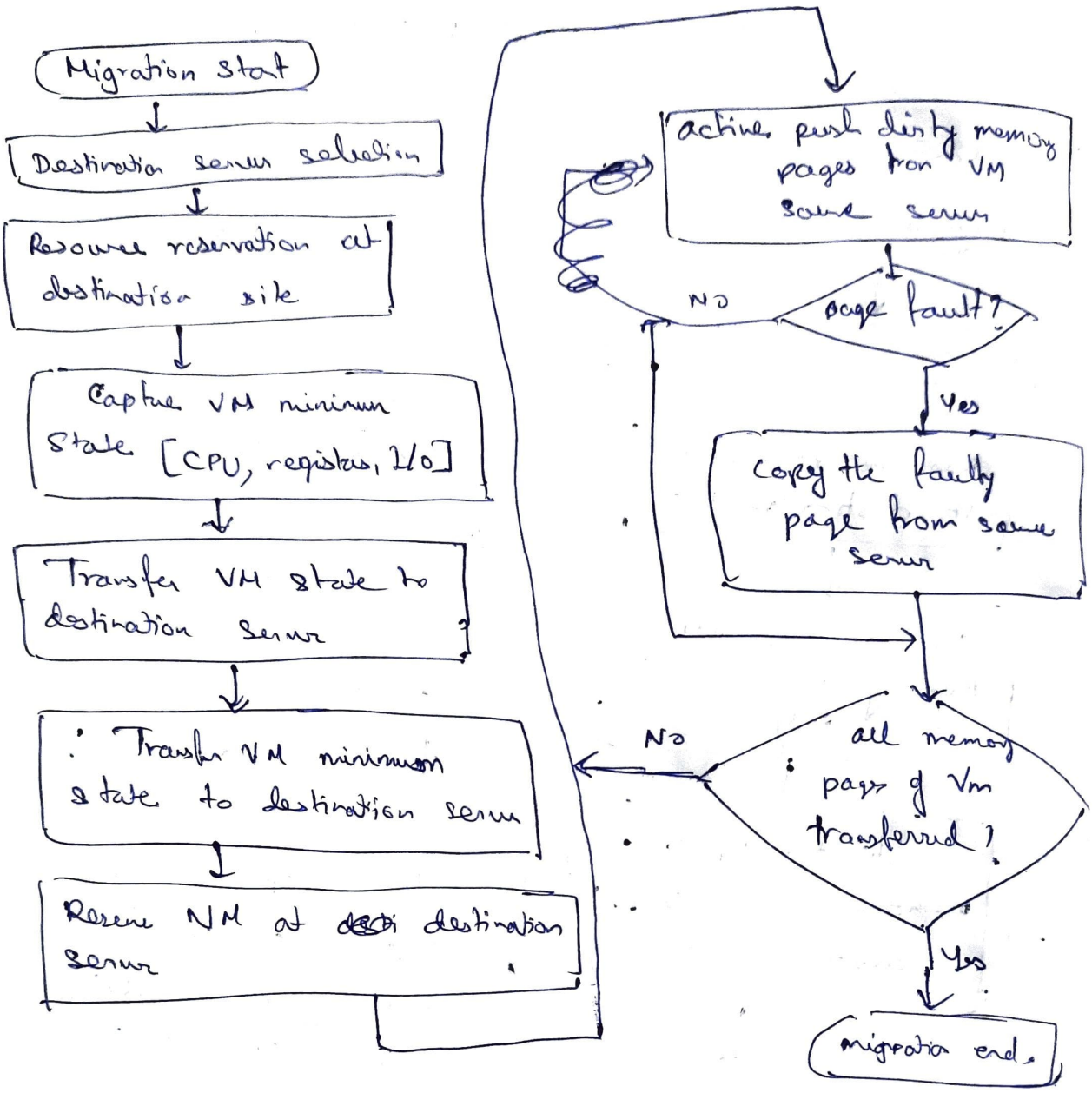


In pre copy, the migration all the source VM memory is transferred first to the destination VM memory.

In order to ensure that only most updated memory is transferred dirty pages are transferred iteratively.

The stop and copy method checks if there are any more dirty pages left to be transferred.

Post copy :-



In post copy, the VM state [CPU, registers, I/O] is transferred to the destination VM & execution is resumed on the destination VM.

If any page fault occurs on the memory, demand paging operation fetches the page from the same VM.

The migration is complete once all the pages are transfered to the destination VM.

Comparison of pre-copy & post copy :-

Performance metrics	Pre-copy technique	Post copy technique
Preparation time.	It includes all the modified pages in (8) iterative memory copying phases.	Negligible
Downtime	It includes transferring any remaining dirty pages	It includes transferring another minimal execution state
Resume time.	Re-Schedule the target VM at the destination server and remove the memory pages at the source server.	Majority of post-copy approach transfer the VM state and most of the memory pages in this period.
pages transferred	Transfer number of pages during preparation time period	Transfer number of pages during resume time period
Total migration time	More	less
Performance degradation	High due to the nature of dirtied pages for write intensive workloads	High due to servicing page faults.

Problems of virtualization :-

→ Each VM requires an OS.

→ Each OS requires license.

→ Each OS has its own compute and storage overhead.

→ Each OS needs maintains & updates.

Container based resource management :-

⇒ Multiple containers run on the same OS.
& Thus seperate OSes are not requires
⇒ no extra licenses required.

→ The containes run on the same OS and shares the resources.
The containes themselves are isolated and cannot interfere with each other.

→ Containers have all the good properties of VMs :-

→ comes with complete files ad data needed to run

→ Multiple copies can run on same machine

→ Same image can run on liffert hardwares.

→ can be stoped, saved or moved to seperate machines.

→ Unlike VM guest OS as containes & remove the need for maintainance & updates.

→ Only updating the Host OS can do the job.

→ Containers use the Host OS's resources, thus there is minimal compute & storage overhead.