DISTRIBUTED COMPUTING

SYLLABUS.

UNIT I Introduction.

Introduction - Definition, Relation to computer System Components motivation - Message passing System Versus shared Memory Systems - primitives for Distributed Communication - Synchronous Versus Asynchronous Executions - Design Issues and challenges : A model of Distributed computations : A Distributed program - A model of Distributed Executions - Models of communication Networks Global state of a Distributed System.

DISTRIBUTED COMPUTING.

Sub code: CS3551 UNIT I

Definition:-

A distributed system is one in which components located at networked computers communicate and co-ordinate their actions only by passing messages

The following features.

1. Concurrency.
2. No Global clock.
3. Independent failures.
4. Autonomy and heterogeneity.

Disadvantages of DS.

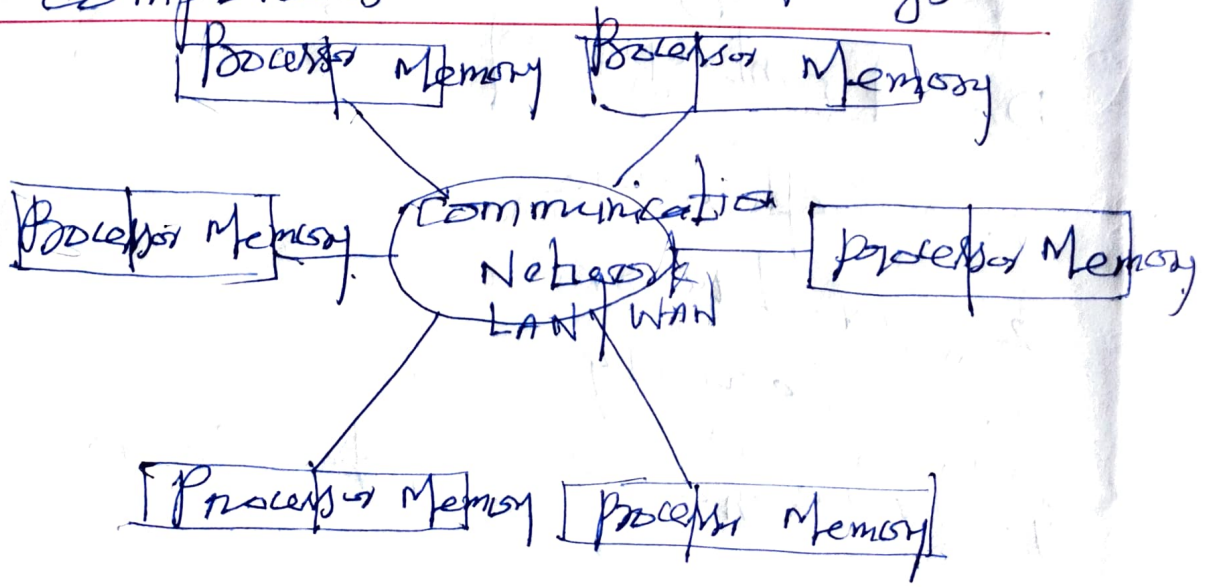
1. Software
2. Network
3. Security
4. Absence of global clock.

Difference between parallel computing and distributed Computing.

Sl. No.	parallel Computing.	distributed computing
1.	The interaction between processes is frequent.	Distributed computations usually long up time.
2.	Assumed to be reliable.	Assumed to be unreliable.
3.	It is fine grained with low overhead	It is heavier weight.

Relation to Computer system Components

Distributed System.



A Distributed System can consists of any number of possible configurations such as mainframes, personal computers, workstations, minicomputers and so on.

Motivation.

1. Economics
2. Speed.
3. Inherent Distribution.
4. Reliability.
5. Incremental growth.

Need of Distributed System.

- Resource sharing is main motivation of the distributed system.

• Resources may be the software resources or hardware resources Pointers, disks, CDROM and data are the example of software and hardware resources.

Primary requirements of distributed system are as follows:

1. Fault tolerance
2. Consistency of replicated data
3. Security
4. Reliability
5. Concurrent transactions.

Focus on Resource sharing.

Resources in a distributed system are encapsulated within the computer and can be

accessed from other computers by communication.

Types of resources

1. Hardware resource
2. Data
3. Service.

patterns of resource sharing

1. Search engine
2. Computer supported cooperative working (CSCW)

For effective sharing each resource must be managed by a program and offers a communication interface enabling the resource any updates reliably and consistently.

Hardware and Software Resource Sharing.

Hardware Resources.

1. CPU
2. Memory
3. Disk
4. Screen.
5. Printer.

Software Resources.

1. Web page
2. File
3. Object
4. Database
5. Newsgroup contents
6. Video / Audio stream.

Message passing Systems Versus Shared Memory Systems.

Message passing.

Two processes communicate with each other by passing messages. Message passing is direct and indirect communication. Indirect communication uses mail box for sending receiving message from other process.

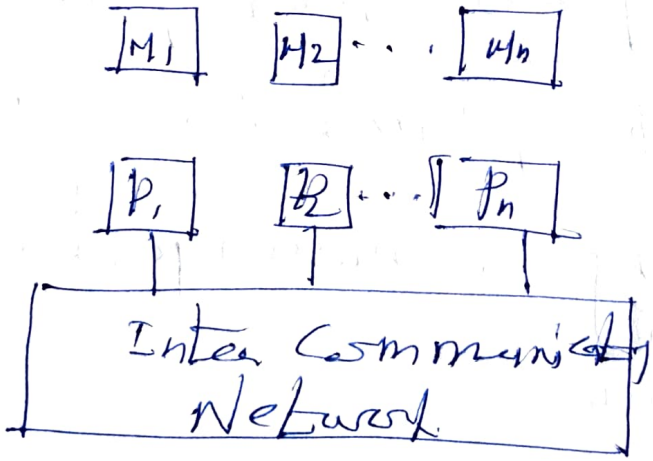
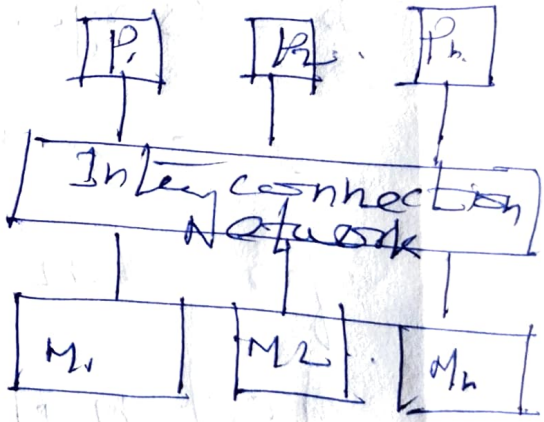
- send primitive is used for sending a message to destination.

- A process receives information for execution the receive primitive which indicates the source of the sending process

and the message.

Shared Memory:

In shared memory systems are those which there is common address space throughout the system.

Sl No.	Message passing system	Shared Memory system
		
	This technique can be used in heterogeneous computers	This cannot be used to heterogeneous computers

Emulating Message - Passing Systems on a Shared Memory Systems

Send and receive operations are implemented by using writing and reading the information from receiver and sender processor.

primitives and Distributed Communication Blocking / Non-Blocking, Synchronous / Asynchronous Primitives.

Message passing primitive Commands.

SEND [msg, dest)

RECEIVE [src, buffer]

- Send primitives uses two

options for sending data:

[Buffered and unbuffered]

Three combinations are possible using blocking and Nonblocking.

1. Blocking send, blocking receive
2. Nonblocking send, blocking receive
3. Nonblocking send, nonblocking receive

Processor Synchrony.

- indicates that all the processors execute in lock step with their clocks synchronized

Synchronous versus Asynchronous Executions.

Sl No.	Synchronous Execution	Asynchronous Execution
1.	means the first task in a program must finish processing before moving on to executing the next task.	means a second task can begin executing in parallel without waiting for an earlier task to finish.
2.	Transmitted messages are received within a known bounded time	No bound on message transmission delay.

Design Issues and challenges

Challenges from System Perspective

- Communication Mechanisms

- Processes
- Naming
- Synchronization
- Data storage and access
- Consistency and replication
- Distributed system security.

Challenges.

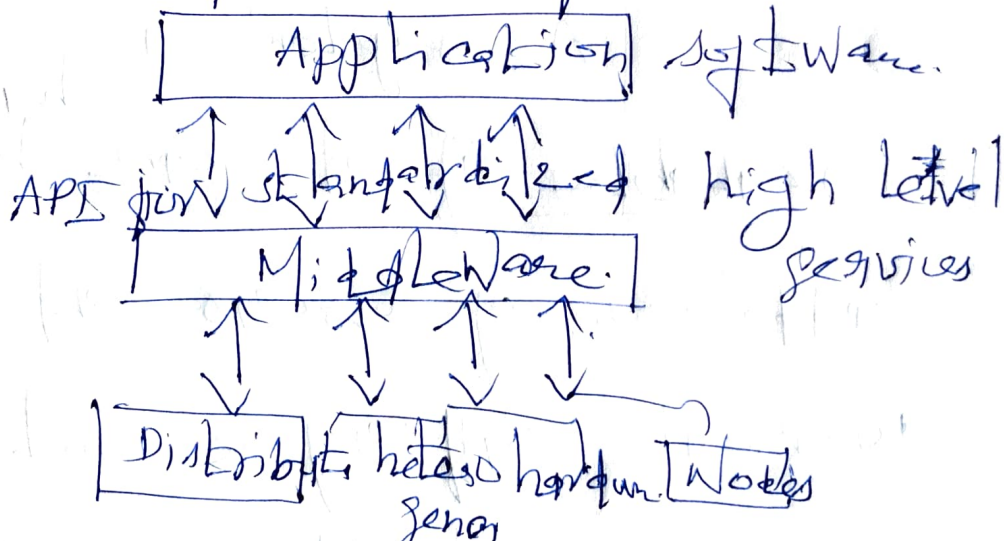
1. Heterogeneity
2. Openness
3. Security
4. Scalability
5. Failure Handling
6. Concurrency
7. Transparency.

Heterogeneity

1. It applies to the all
   - Computer Network
   - Computer hardware devices.
   - Operating Systems
   - programming languages.

Middleware

positions of Middleware.



Openness

— means that the system can be easily extended and

modified.

Security.

Three components.

1. Confidentiality.
2. Integrity
3. Availability.

Scalability.

A system is said to be handle scalable if it can ~~handle~~ the addition of users and resources without suffering a noticeable loss of performance.

Distributed system can be scalable because additional computers can be added in order to host additional components.

1. In size
2. In location
3. In administration

Scaling techniques.

1. Hiding communication latencies.
2. Distribution
3. Replication

Failure Handling.

Techniques for handling with failures.

1. Detecting failures
2. Masking Failure.
3. Tolerating failures
4. Recovering from failure
5. Redundancy

Detecting failure:

Not all failures are detected but some of the failures can detected.
For example: Corrupted data from file is detected by using checksum

Masking failures:

1. Messages could be retransmitted

Tolerating failures:

Recovery from failures.

Redundancy. – Services can be made to tolerate failures.

Concurrency:-

Components in distributed systems are executed in concurrent processes.

Transparency:

Concept: Hide different aspects of distribution from the client.

1. Location Transparency
2. Access "
3. Concurrency "
4. Replication "
5. Failure "
6. Mobility "
7. Performance "
8. Scaling "

Advantages of transparency:

- Easier for the user.
- Easier for the programmer

Disadvantages of transparency

- Underlying system can be very complete.

Application of Distributed computing and Challenges

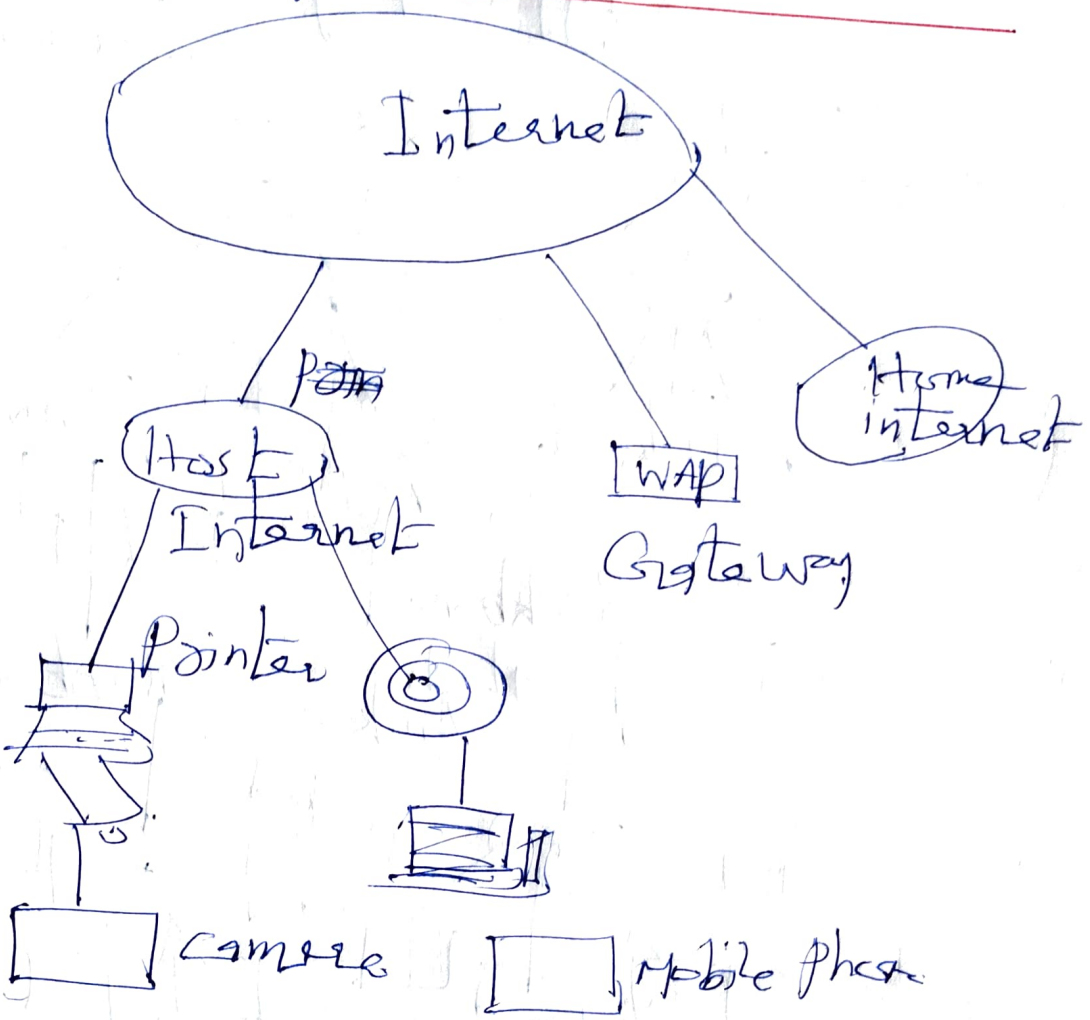
1. Mobile system

Mobile devices are

1. Laptop computer
2. Mobile computing
3. Ubiquitous computing

2. [illegible]

portable devices in DS

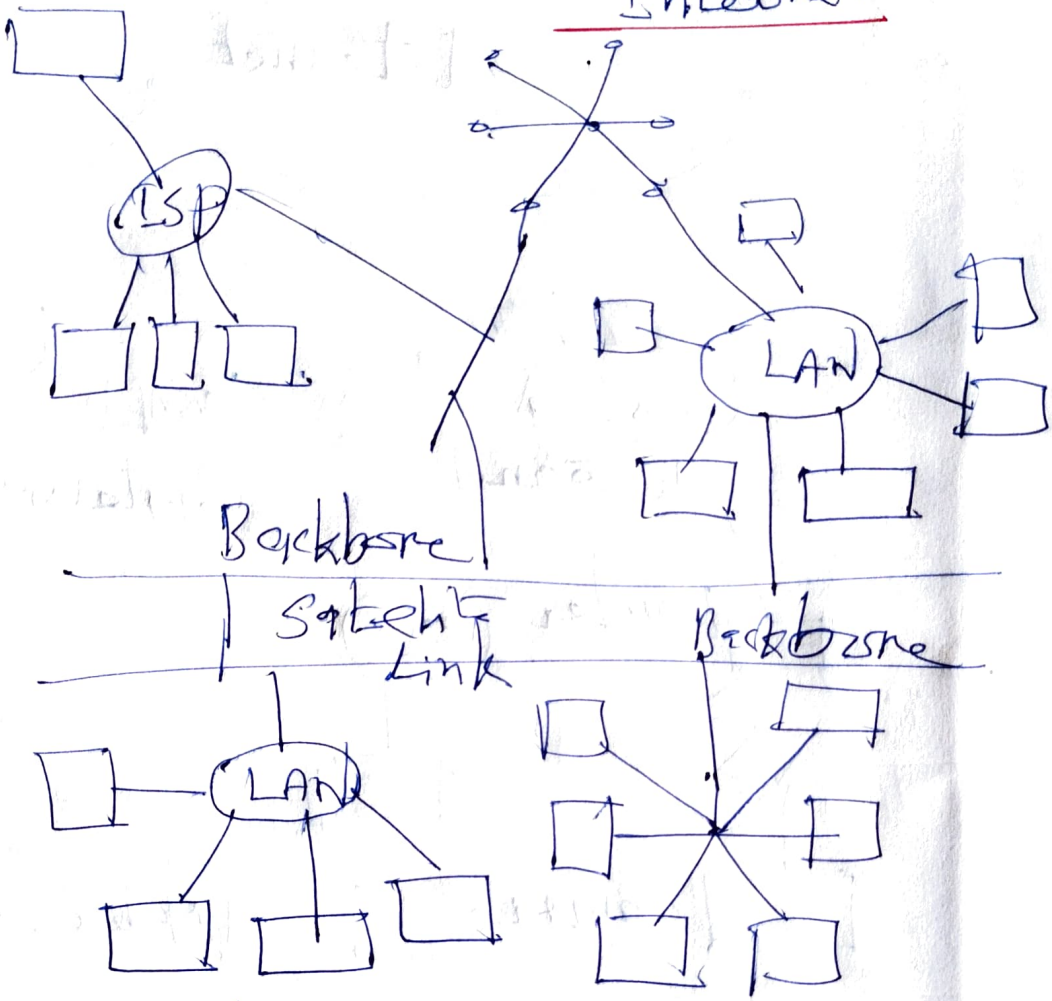


pervasive Computing

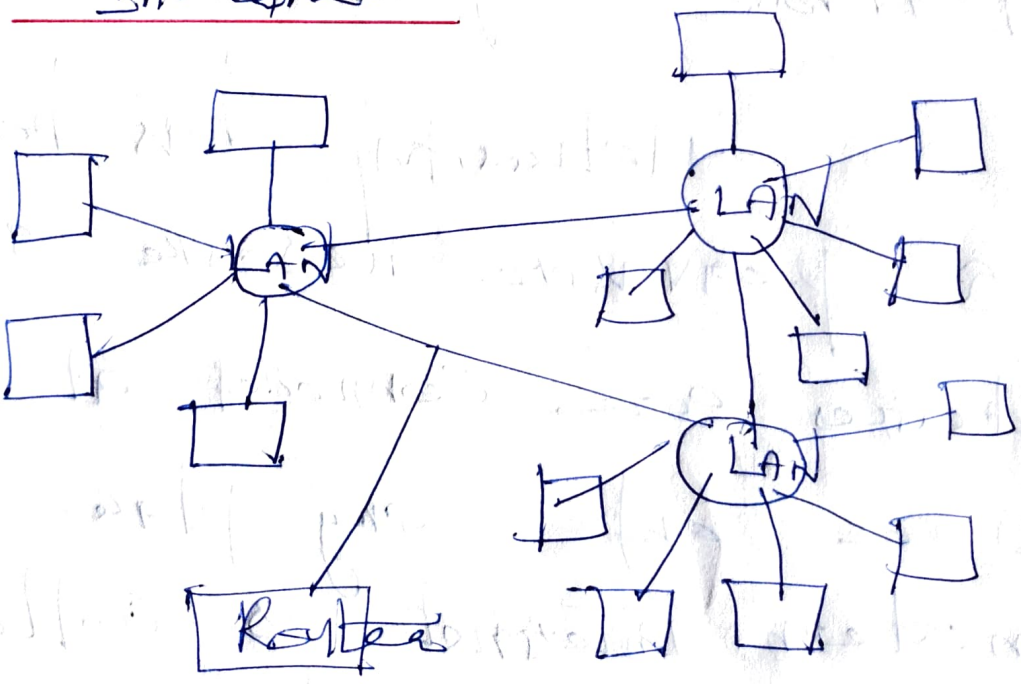
Networking has become a pervasive resource and devices are connected at any time and any place. The modern Internet is collection of various computer networks

Typical position of Internet

Internet



Intranet



3. Multimedia System

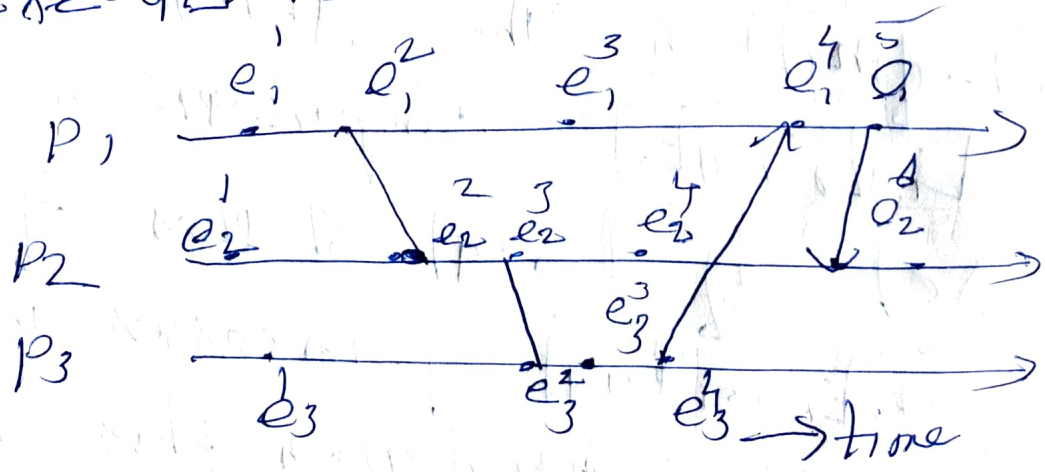
4. Web casting

— is an application of distributed multimedia technology. It broadcast continuous media over the internet. Webcasting is called push technology

A model of distributed computations:

A distributed program

A model of Distributed Executions



Causal precedence relation.

- Ordering of events for a single process is simple.

Logical Vs Physical Concurrency.

Two events are logically concurrent if and only if they do not causally affect each other

Models of Communication Networks.

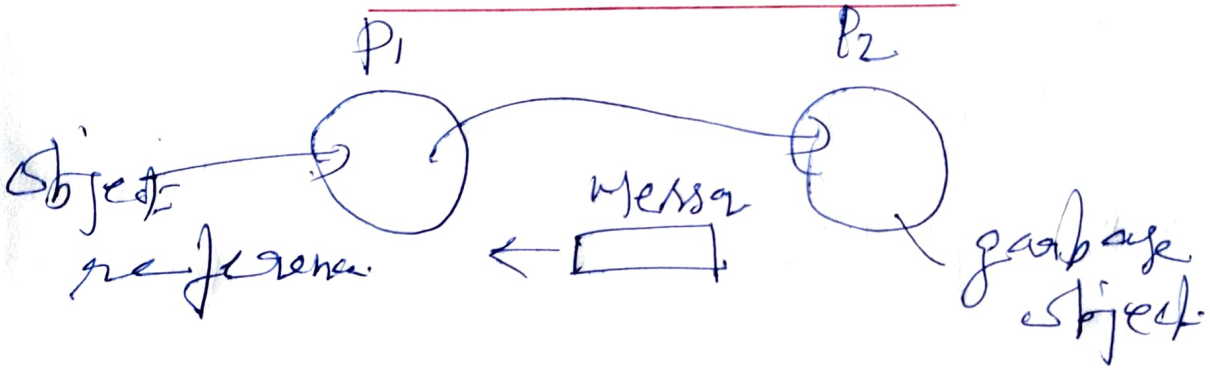
- Namely FIFO, Non FIFO and Causal Ordering

Global State of Distributed System

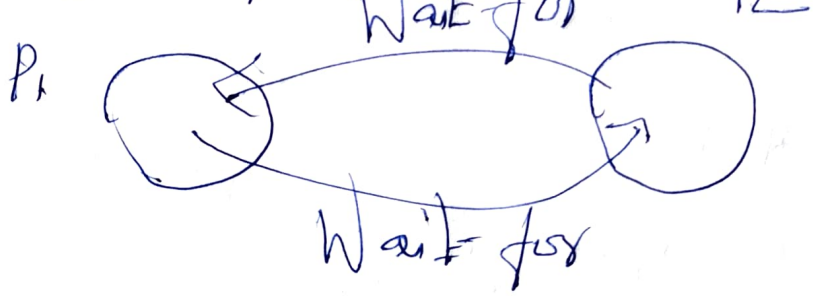
Definition: The global state of a distributed computation is the set of local states of all individual processes involved in the computation plus the state of the communication channels

Requirements of global state

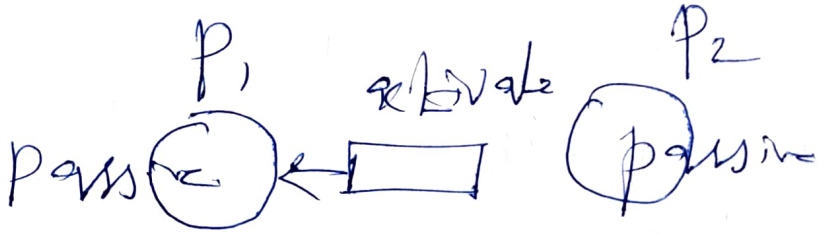
a. Garbage collection



b. Deadlock



c. Termination.



1. Distributed Garbage collection
2. Distributed deadlock detection
3. Distributed termination detection
4. Distributed debugging.

UNIT II LOGICAL TIME AND GLOBAL STATE

Logical Time: Physical clock synchronization: NTP – A framework for a system of Logical clocks Scalar Time, Vector-time: Message Ordering and Group communication Message ordering paradigms – Asynchronous Execution with Synchronous communication – Synchronous program order on Asynchronous system – Group communication – Causal order – Total order Global state and snapshot Recording Algorithms: Introduction System model and Definitions – Snapshot Algorithm for FIFO Channels.

UNIT II

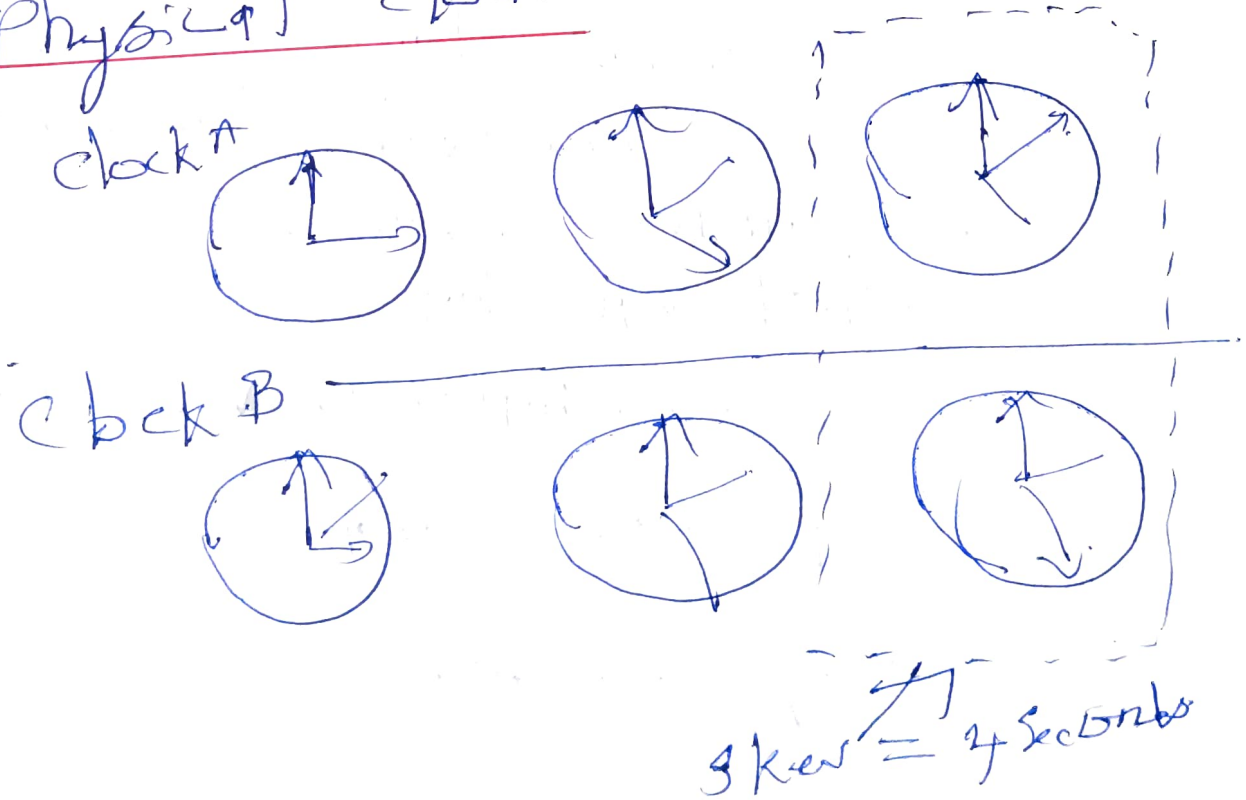
Logical Time and Global State

Clock events and process state

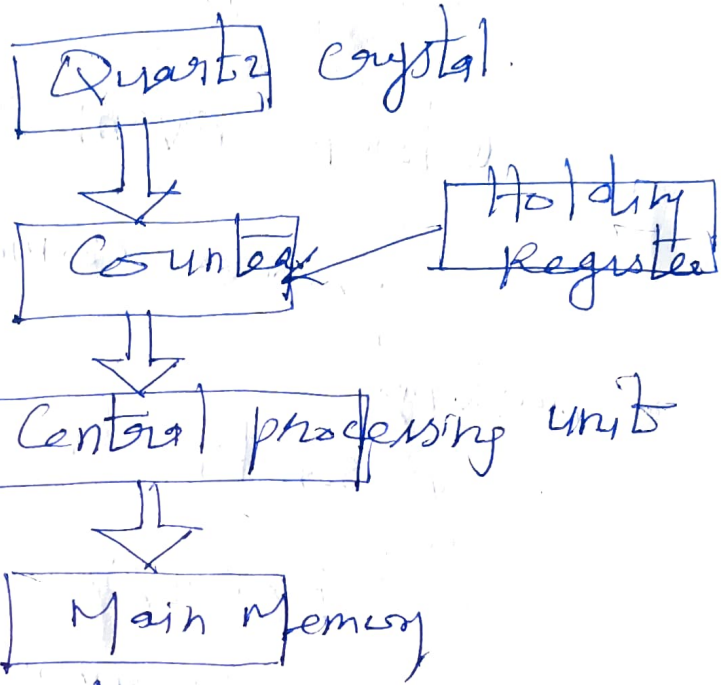
process communicate only by messages.

1. Sending a message
2. Receiving a message.
3. performing a computation that alters its state s;

Physical clock.



Working of Computer clock.

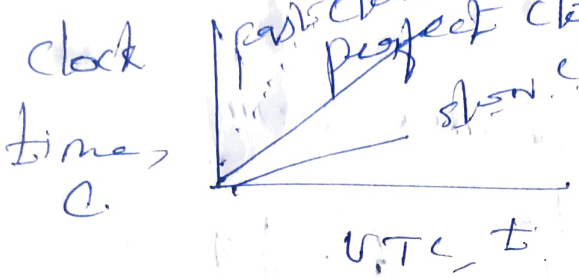


Some definitions.

1. Transit of the sun
2. Solar day.
   — Interval between two executive transits of the sun is called Solar day.
3. Coordinated Universal time.

Clock skew and Drift Compensating.

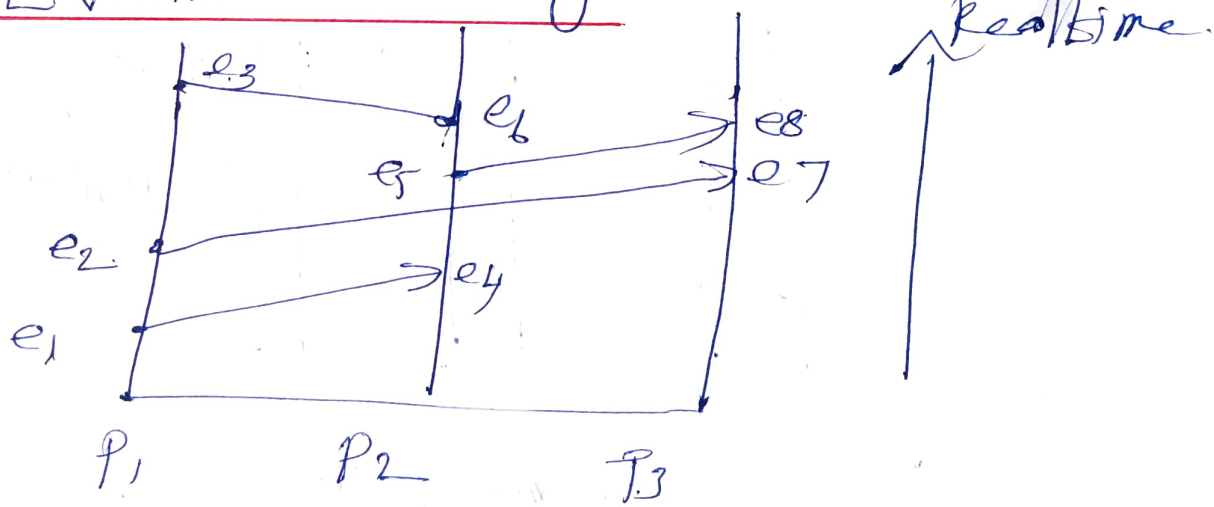
Drift rate of clocks.



Logical Time.

The convention in these algorithms is to speak of logical clocks.

Event ordering.

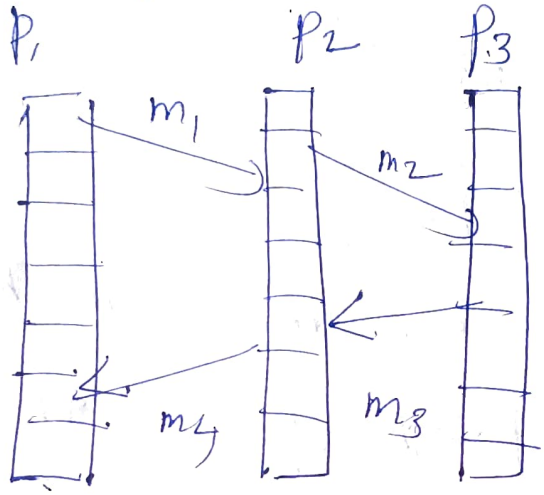


possible event ordering

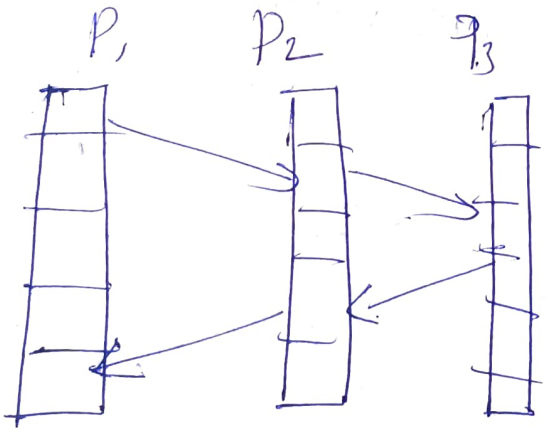
1. $e_1 \rightarrow e_2 \rightarrow e_3$
2. $e_1 \rightarrow e_4 \rightarrow e_5 \rightarrow e_6 \rightarrow e_3$
3. $e_2 \rightarrow e_7 \rightarrow e_8$

Lamport Timestamp

clock timestamp



processes with its own clock runs at different rate

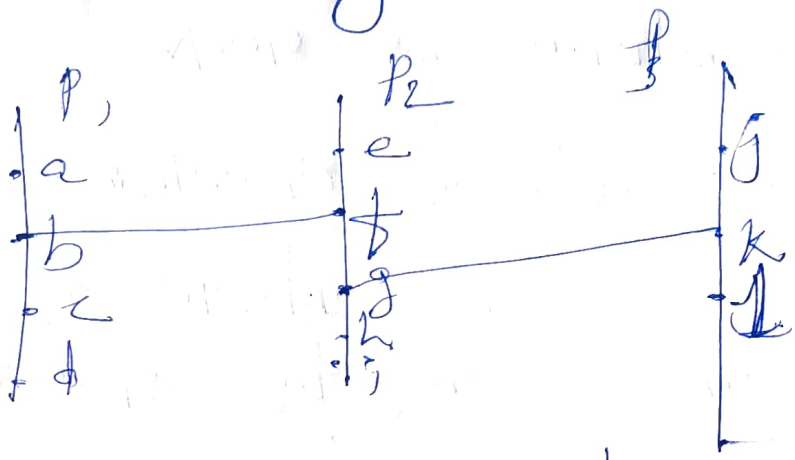


Lamport's Algorithm

Vector Timestamp

A Vector clock system is a mechanism that associates timestamps with events such that comparing two events timestamps

indicates whether those events are causally related.



Physical clock synchronization

Clock synchronization is done by two methods:

1. Internal synchronization
2. External synchronization

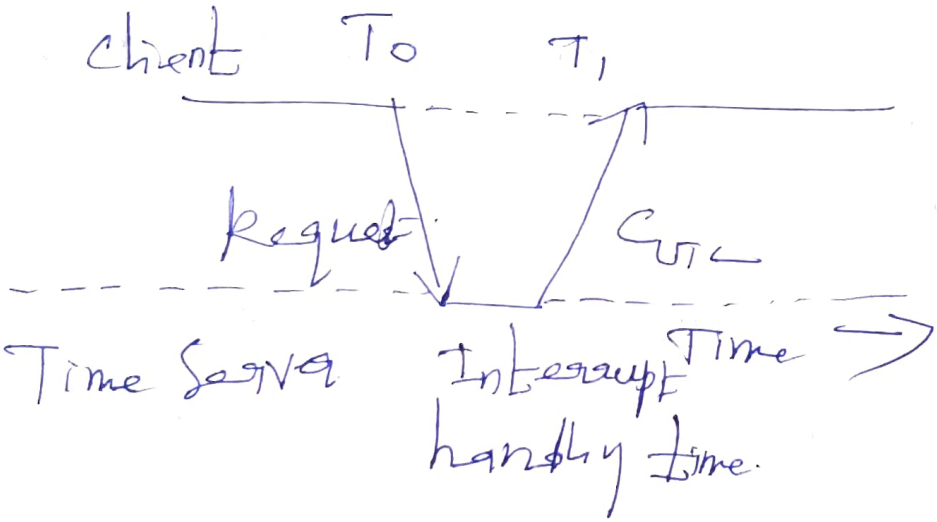
Synchronization in a Synchronous system

1. The time to execute each step of a process has known lower and upper bounds.

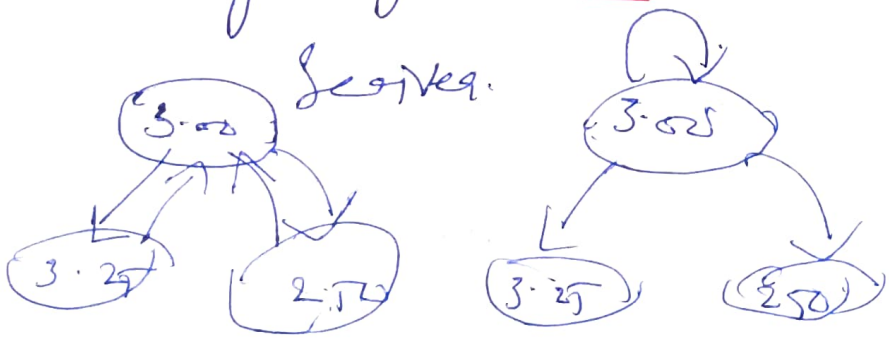
Christian's Method for Synchronizing clocks.

- Christian's Algorithm

- is centralized passive time server type algorithm.



Berkeley Algorithm



Network Time Protocol

Goals

Features of NTP

1. Multicast Mode
2. Procedure - Call Mode
3. Symmetric Mode

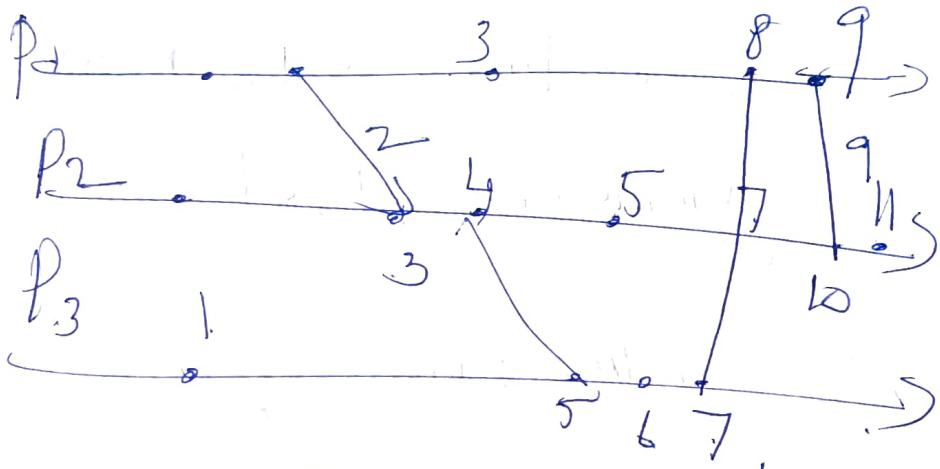
A Framework for a System of Logical Clocks

The logical time in distributed system is used to maintain the consistent ordering of events.

Relation $<$ is called the happened before or causal precedence.

Scalar time

Scalar time is designed by Lamport to synchronize all the events in distributed systems



Scalar time is designed by Lamport to synchronize all the events in distributed systems.

Basic Properties

1. Consistency property
   - Total ordering
   - Event counting

Vector time

Vector clocks are used in a distributed systems to determine whether pairs of events are causally related

Message ordering paradigm

Message delivery in orderly manner is important because it determines the messaging behaviour that can be expected by the distributed program.

Asynchronous Execution with Synchronous communication

Synchronous order.

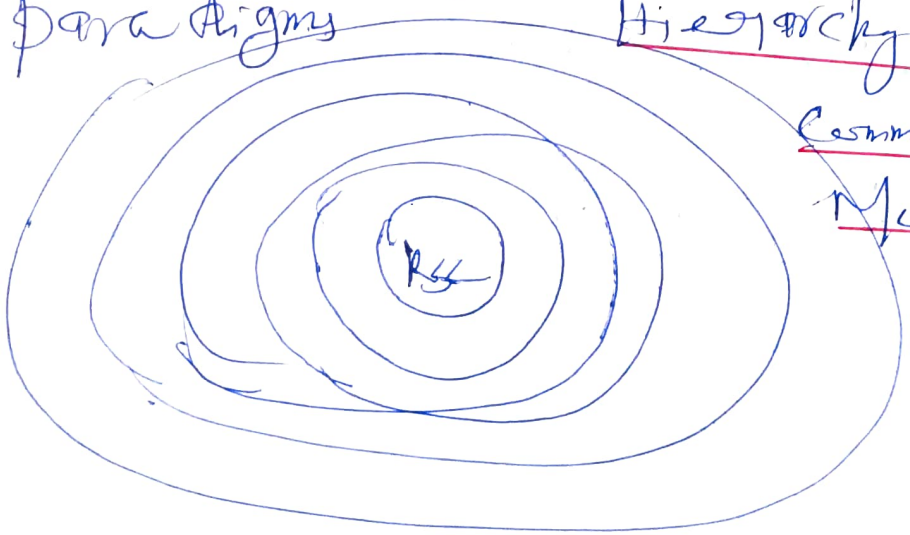
Process A	Process B
...	...
Send (B)	Send (A)
Receive (B)	Recev (A)

Execution Realizable with Synchronous Communication

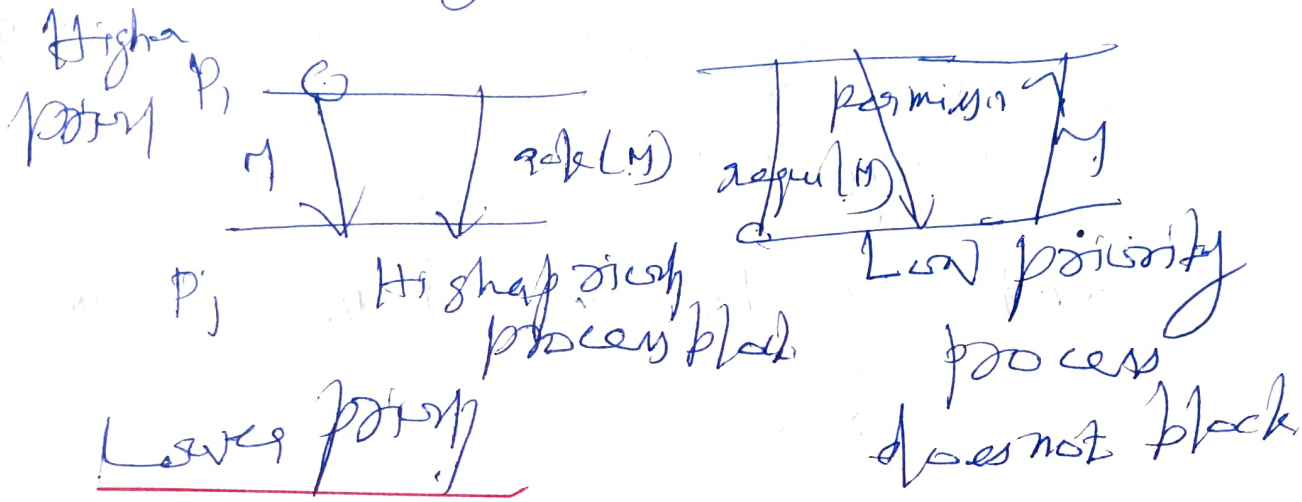
if each send event is immediately followed by its corresponding receive event

Hierarchy of Paradigms

Message ordering Hierarchy of Communication Models



Synchronous Program Order on Asynchronous System



Group Communication

1. One to many
2. Many to one
3. Many to many.

One to Many communication

- Message is sent one sender to multiple receivers

- Group Management
- Group addressing
- Buffered and Unbuffered multicast.

Send to all and Bulletin board Semantics

a. Send to all semantics
b. Bulletin-board Semantics

Atomic Multicast

— has an all or nothing property.

Why to use Communication

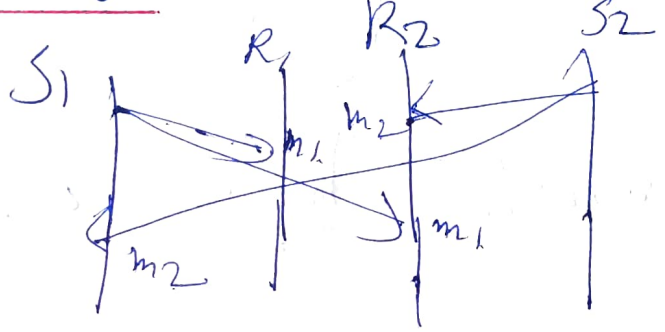
multiple servers send messages to a single server.

Many to Many Communication

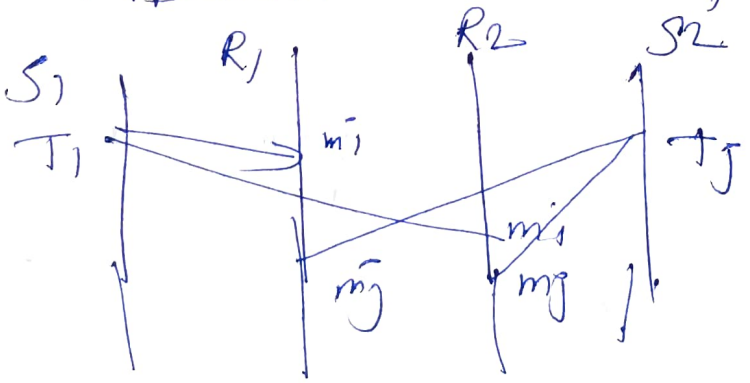
Multiple Senders send messages to multiple receivers

Message Ordering

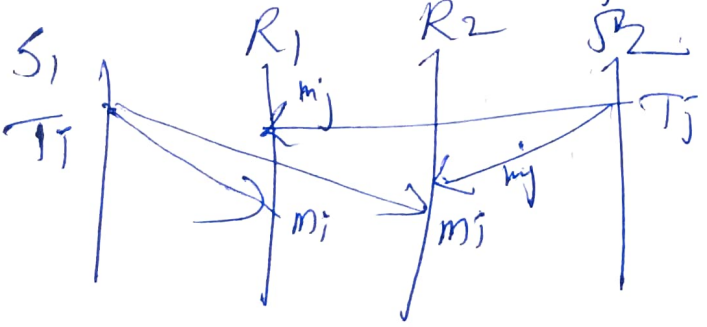
No ordering



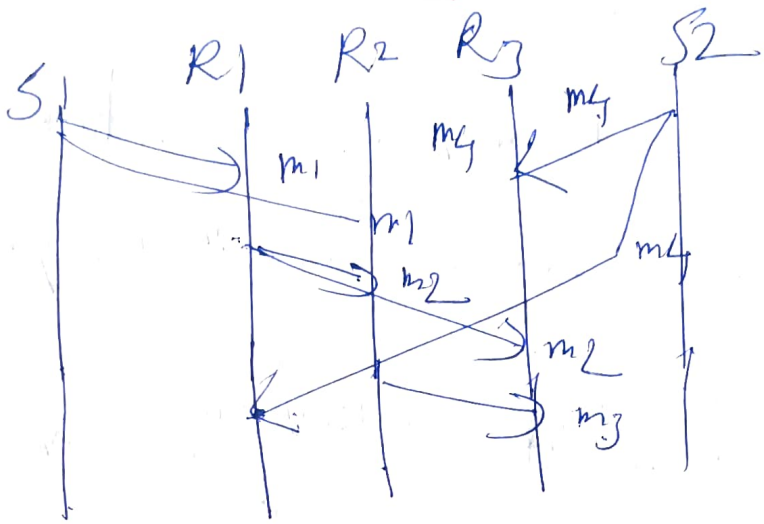
Absolute ordering



Consistent ordering



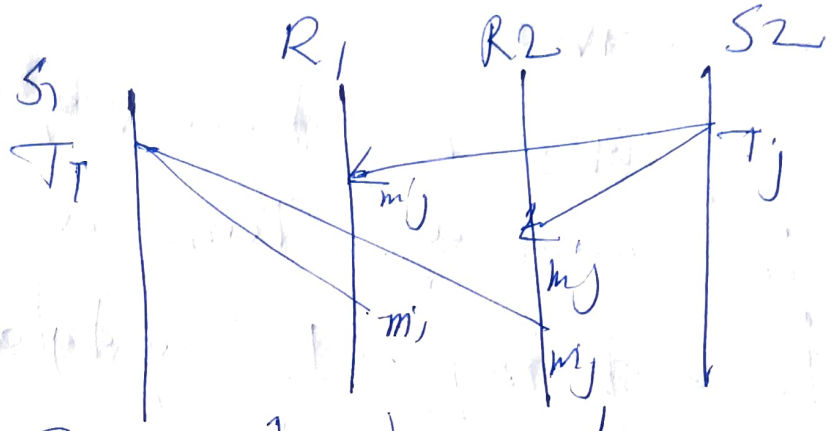
Causal order.



Raynal Schiper Toueg Algorithm

Every message transmitted by P_i is tagged with the contents of $sent_i$. Each process P_i also maintains an n-entry vector, $DELV_i$ to record the number of messages delivered to P_i from all others.

Total order



Consistant ordering.

Rule: Messages received in the same order regardless of their timestamp

Example: Replicated database updates.

Drawback: A centralized Algorithm

Three phase distributed Algorithm

Sending:

phase 1 - process multicast the message with a locally tag

and the local timestamp to the group members.

Receivers:

Phase 1: Receiver receives the message with a tentative timestamp. It updates the variable priority.

Global State and Snapshot Recording Algorithms.

Events are related: Events occurring at a particular process are totally ordered by their local sequence of occurrence, and receive event has a corresponding send event.

System Model

- A distributed computing system consists of spatially separated processes that do not share a common memory and asynchronously with each other by passing messages over communication channels.

Consistent Global State

The global state of a distributed system is a collection of the local states of the processes and the channels

Snapshot Algorithms for FIFO channels.

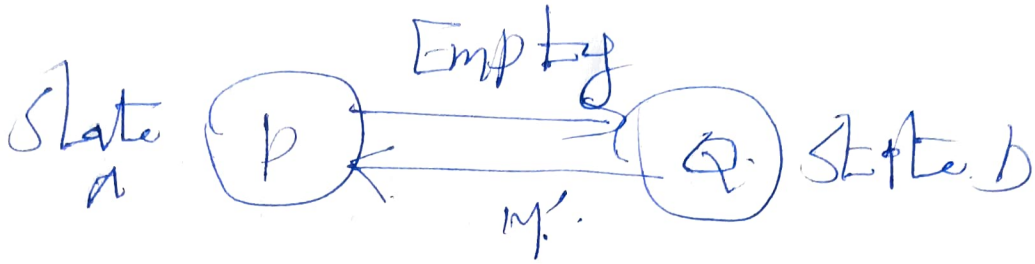
Chandy Lamport Algorithm

- records a set of process and channel states such that the combination of a consistent global state.

- communication channels assumed to be FIFO

- property of the Recorded Global state

- may not correspond to any of the global states that occurred during the computation



on receiving the marker p records the state of c' as the sequence consisting of the single Message M'.

The recorded Global State S^0 is shown above.

UNIT III DISTRIBUTED MUTEX AND DEADLOCK.

Distributed mutual Exclusion Algorithms: Introduction - preliminaries - Lamport's Algorithm - Ricart - Agrawala's Algorithm - Token based Algorithm - Suzuki - Kasami's Broadcast Algorithm, Deadlock Detection in Distributed systems- Introduction - System model, preliminaries- Models of deadlocks - Chandy - Misra - Haas Algorithm for the AND model and OR model.

UNIT III

Distributed Mutex and Deadlock.

Distributed Mutual Exclusion Algorithms: Introduction.

Mutual exclusion ensures that concurrent processes make a serialized access to shared resources or data.

Mutual exclusion

Entry section

Critical section.

Exit section

Remainder section.

Preliminaries

System Model.

Distributed mutual exclusion algorithms must deal with unpredictable message delays and incomplete knowledge of the system state.

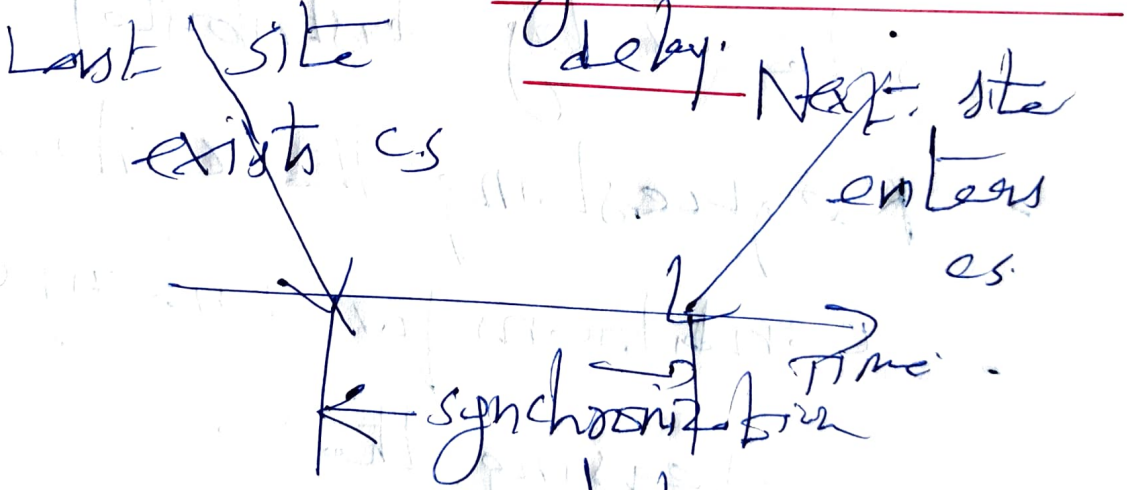
Requirement of Mutual Exclusion.

1. Freedom from deadlocks.
2. Freedom from starvation
3. Strict fairness
4. Fault tolerance.

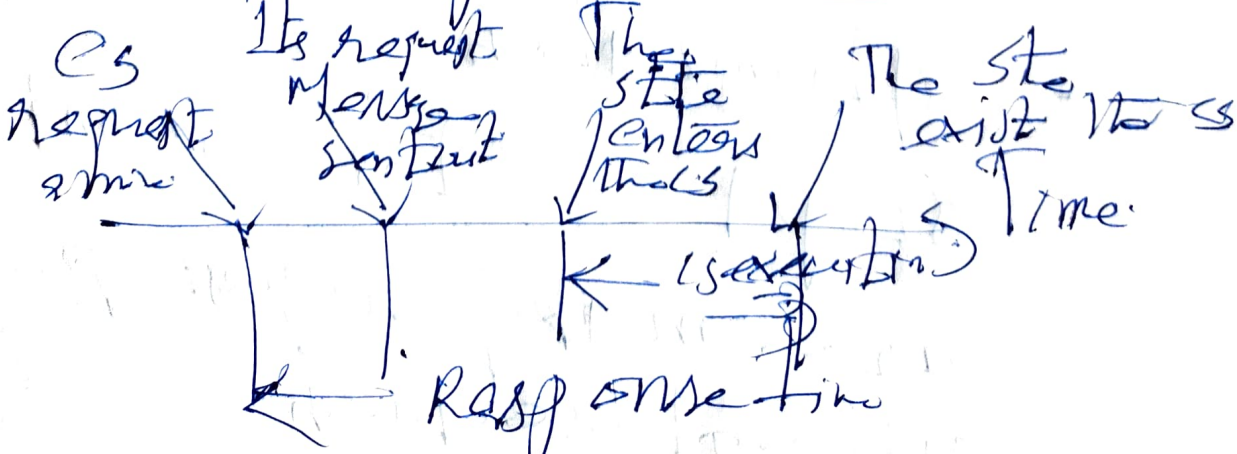
performance Metrics.

1. Message complexity.
2. Synchronization delay.
3. Response time.
4. System throughput.

Synchronization delay



Response time

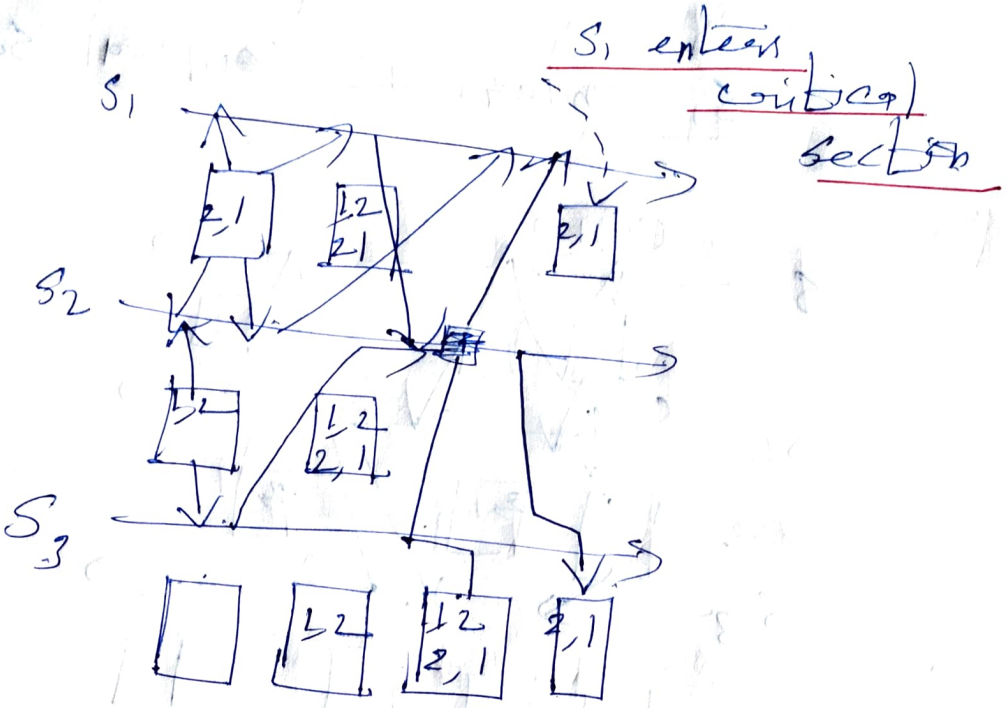


Lamport's Algorithm

Each process freely and equally competes for the right to use the shared resource requests are arbitrated by a central control site or by distributed agreement

- Requesting the critical section
- conditions for Entering CS
- Releasing the CS
- Correctness
- optimization

Step 1: Sites S_1 and S_2 are making request for critical section

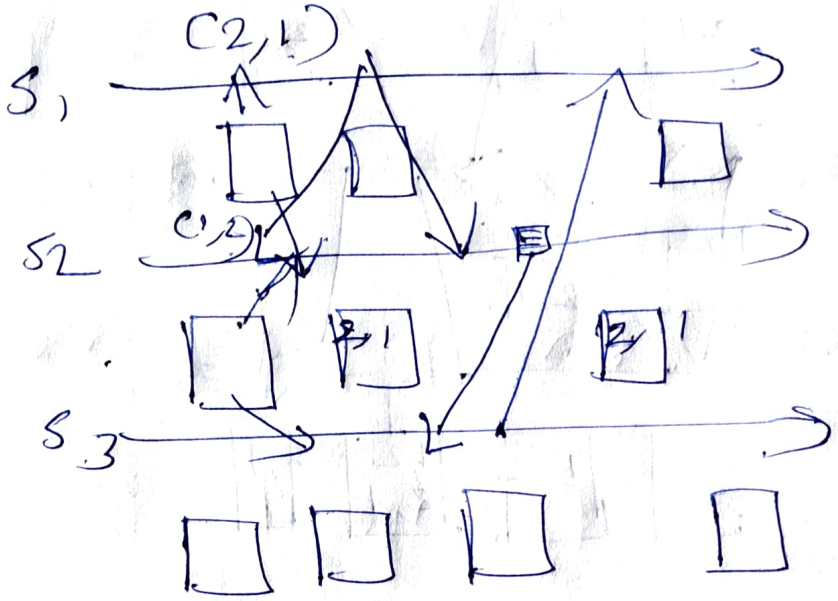


Ricart - Agarwala's Algorithm.

- Ricart Agarwala algorithm is an optimization on Lamport Algorithm.

- Releasing the critical section
- Optimization

S1 enters the critical section



Token Based Algorithms

Token based algorithms are Suzuki-Kasami algorithm and Raymonds tree Algorithm

- Major Design issues
  - Important Data structures

Algorithm:

- Requesting CS:
- Executing CS:
- Release CS

Performance:

1. The algorithm is simple and efficient. It requires 0 or N messages per CS invocations.
2. Synchronization delay:

0 or T. No messages is needed or the synchronization

delay if a site holds the idle token at the time of its request.

Deadlock Detection in Distributed systems: Introduction.

- A distributed system consists of a number of sites connected by a network. Each site maintains some of the resources of the system.

Deadlock occur in the following situation.

1. Process A at site 1 holds a lock on resource X.

2. Process A has requested but has not been granted resource Y at site 2.

3. Process B at site 2 holds a lock on resource Y

4. Process B has requested but has not been granted resource X at site 1.

Both processes are blocked by the other one. There is a global deadlock.

Necessary condition.

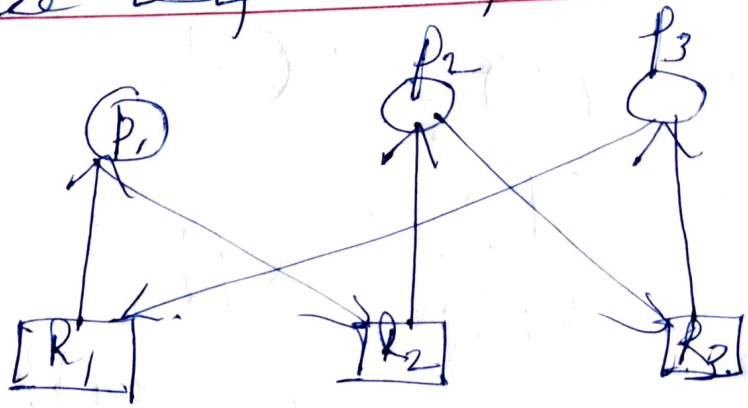
- A process can be in two states : Running or blocked

In the running state, a process has all the needed resources is either executing or is ready for execution.

Following conditions should hold simultaneously for deadlock to occur.

1. Mutual Exclusion.
2. No-Preemption.
3. Hold and wait
4. Circular wait.

circular wait by three deadlocked processes



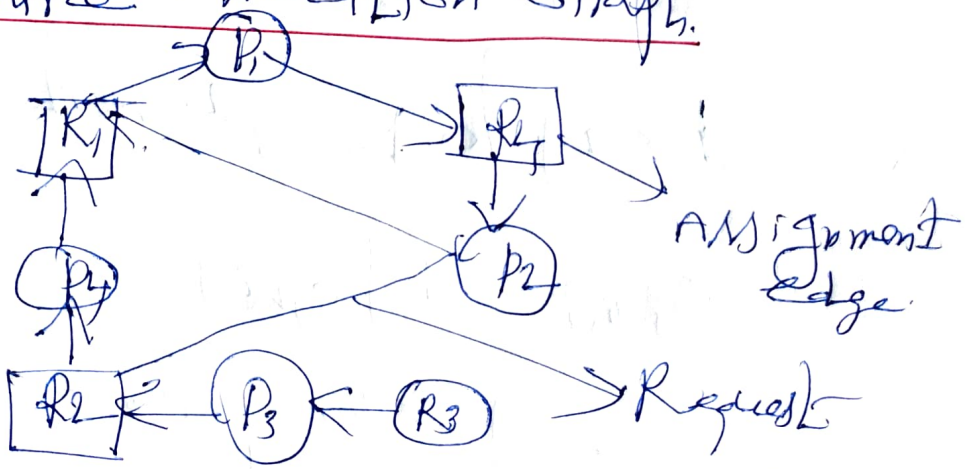
System Model

Each process utilizes a resource as follows

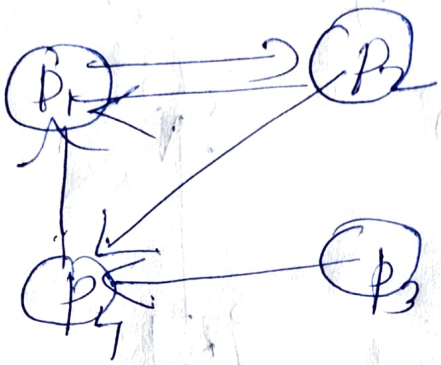
1. Request 2. Use 3. Release

Wait for Graph.

Resource Allocation Graph.



Wait for Graph.



Preliminaries: Deadlock Handling Strategies

Three strategies.

1. Deadlock prevention
2. Deadlock Avoidance
3. Deadlock detection

Deadlock prevention

First Method.

Prevent the circular wait condition by defining a linear

ordering of resource types

Second Method:

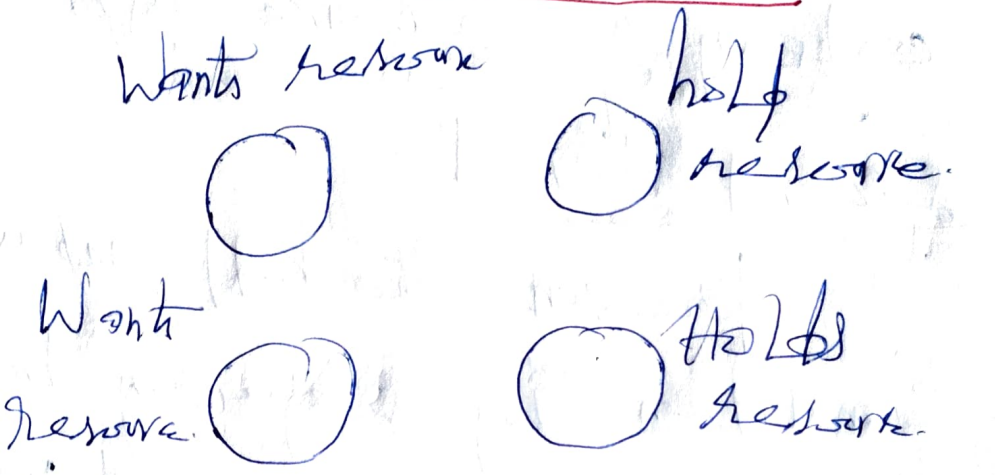
Prevent the hold and wait condition by requiring the process to acquire all needed resources before starting execution.

Third Method.

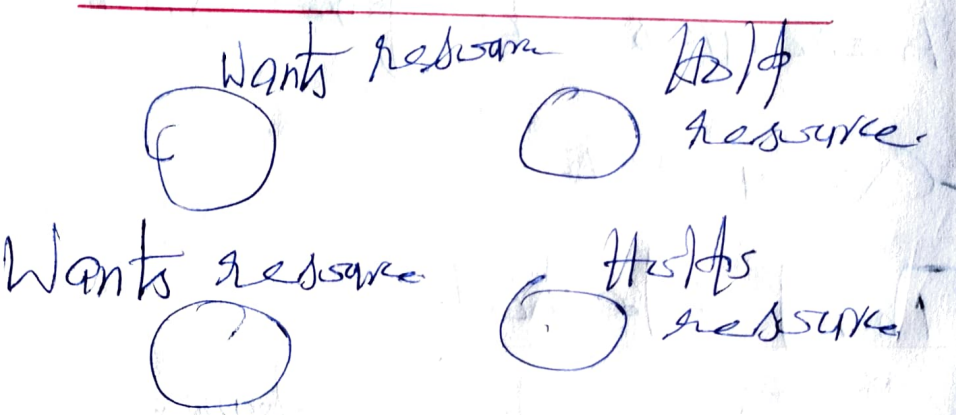
Use of time-stamps.

Use time-stamps for transaction to a database. Each transaction has the time stamp of its creation

Wait - Die Method



Wound - Die Method.



Deadlock Avoidance.

Decision made dynamically before allocation a resource, the resulting global system state is checked if it is safe state then allow for allocation.

Deadlock Detection

• principle of operation

Detection of cycle in WFG proceeds concurrently with normal operation.

- Resolution
- Observation

Models of deadlocks

1. Single Resource Model

- A process can have at most one outstanding request for only one unit of a resource.

The AND Model

A process that requires for execution can proceed when it has acquired all those resources

The OR Model

A process that requires resources for execution can proceed when it has acquired at least one of those resources

The AND OR Model

- a request may specify any combination of AND or in the resource request

Chandy Misra - Haas Algorithm for the AND model

It is considered one of the best deadlock detection Algorithm for distributed systems.

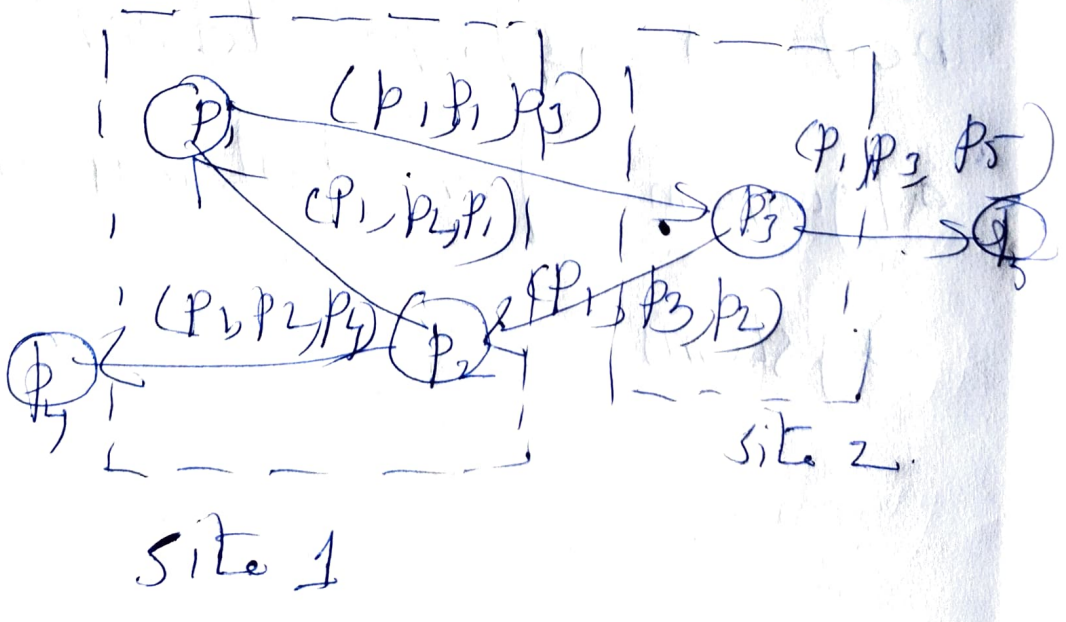
This is considered an edge - chasing probe - based Algorithm. It is also considered one of the best deadlock detection algorithm for distributed systems.

1. The id of the process that is blocked

2. The id of the process sending this particular version of the probe message and

3. The id of the process that should receive the probe message.

Deadlock example.



Advantages

1) It is easy to implement
2) Each probe message is of fixed length
3) There is little computation
4) There is little overhead
5) There is no need for special data structure

Chandy - Misra - Haas Algorithm for the OR model.

Chandy Misra Haas distributed deadlock detection algorithm for OR model is based on the approach of

diffusion - computation...

Two types of messages are used in a diffusion computation - query(i,j,k) and reply(i,j,k) denoting that they belong to a diffusion computation initiated by a process P_i and are being sent from P_j to process P_k.

When a blocked process P_k receives a query(i,j,k) it takes the following actions

1. If this is the first query message received by P_k for the deadlock detection initiated by P_i then it propagates the query to all the processes in its dependent set and sets a local variable num(k) to the number of query messages sent.

2. If this is not the engaging Query then P_k returns a reply message to it immediately provided P_k has been continuously blocked since it received the corresponding engaging query. Otherwise it discards the query.

UNIT IV

Consensus and Recovery.

Concensus and Agreement Algorithms. Problem definition - Overview of Results - Agreement in a Failure - Free System [Synchronous and Asynchronous] - Agreement in Synchronous Systems with Failures - Check pointing and Roll back Recovery Introduction - Background and definitions - Issues in failure Recovery Checkpoint based Recovery - Coordinated checkpointing Algorithm - Algorithm for Asynchronous checkpointing and Recovery.

4.1. Consensus and Agreement Algorithms: problem definition.

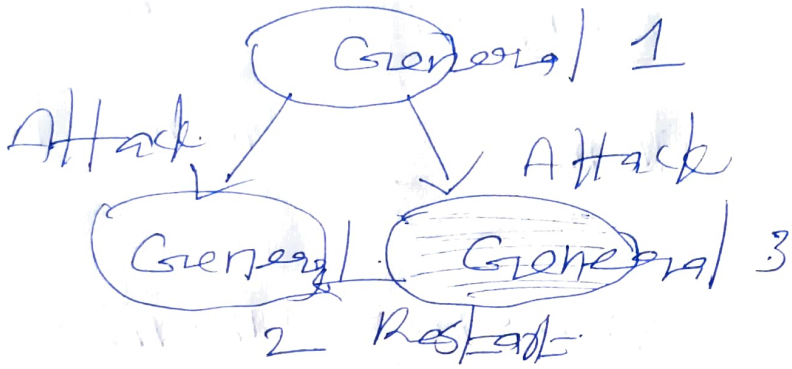
Agreeing whether to commit or to abort a transaction in distributed database Management system.

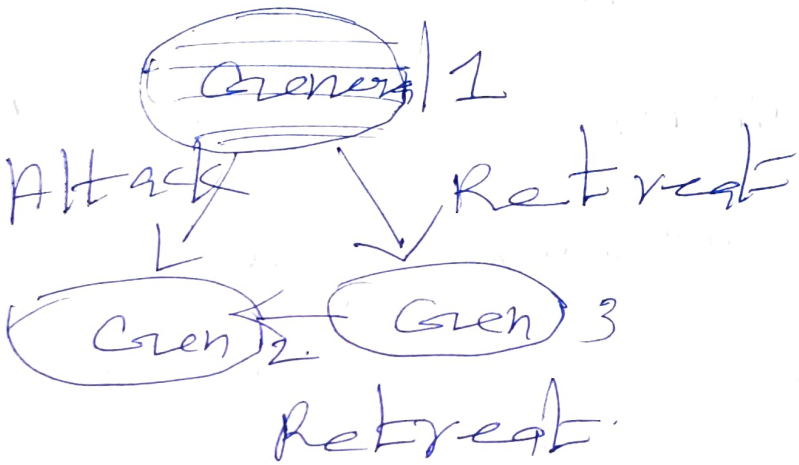
4.2: Byzantine Agreement problem.

Agreement:

All non faulty processors agree on the same value.

Validity: If source is non faulty, then the common agreed value must be the value supplied by the source processor.





4.2.1. Consensus Problem.

value agreed upon by faulty processors is irrelevant.

1. Agreement: All non-faulty processors agree on the same single value.

2. Validity: If initial value of every non-faulty processor is v, then all non-faulty processors

can agree on any common value. Value agreed upon by faulty processors is irrelevant.

4.2.2. Interactive consistency problem:

1. Every processor has its own initial value.

— Agreement is on a set of common values.

Overview of Results.

Sl.No	Failure Mode.	Synchronous system	Asynchronous System
1.	No failure.	Agreement attainable.	Agreement attainable.
2.	Crash failure.	Agreement attainable.	Agreement not attainable.

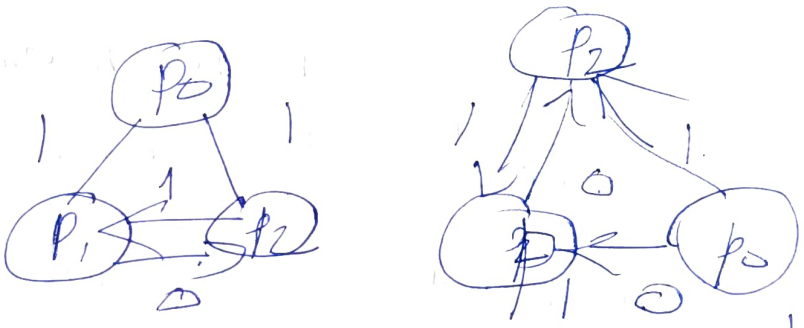
4.4. Solution to Byzantine Agreement Problem

1. All non faulty processors agree on the same value.

Validity: If source is nonfaulty then the common agreed value must be the value supplied by the source processor.

4.4.1. Impossible Scenario.

Two possibilities.



Case 1 - P_0 is nonfaulty

P_2 is faulty P_1 should agree upon 1 as the value. Not possible.

Case 2 - P_0 is faulty. P_1 may agree on 1 and P_2 on 0.

P_0 - nonfaulty.

P_1 - faulty.

4.4.2 Lamport - Shostak Pease Algorithm

Algorithm OM(0)

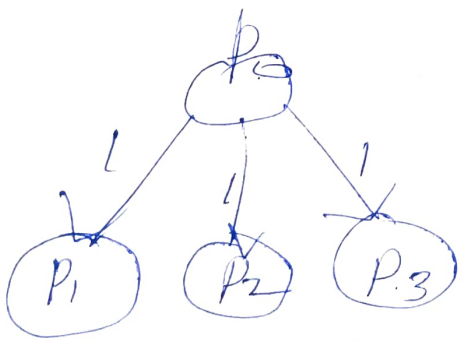
1. Source processor sends its values to every processor
2. Each processor uses the value it receives from source.

Lamports Algorithm : Example 1.

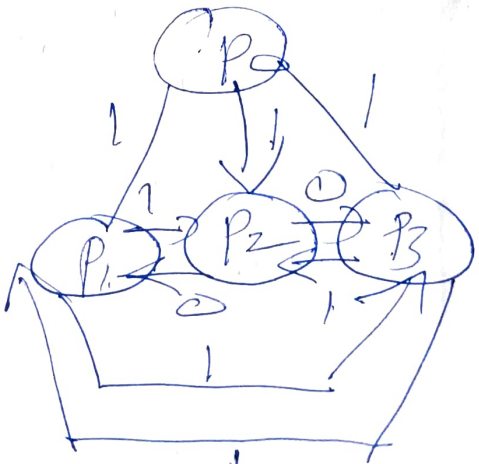
System with 4 processors

P_0, P_1, P_2, P_3, P_0 is source

P_2 is faulty.



processor P_0 executes the Algorithm OM(1)



processor $P_1 P_2 P_3$ executes the algorithm OM(0)

4.5 Agreement in a Failure free System

Failure free system

consensus can be reached by collecting information from the different processes arriving at a decision and distributing this decision in the system.

4.4: Agreement in Synchronous systems with failures.

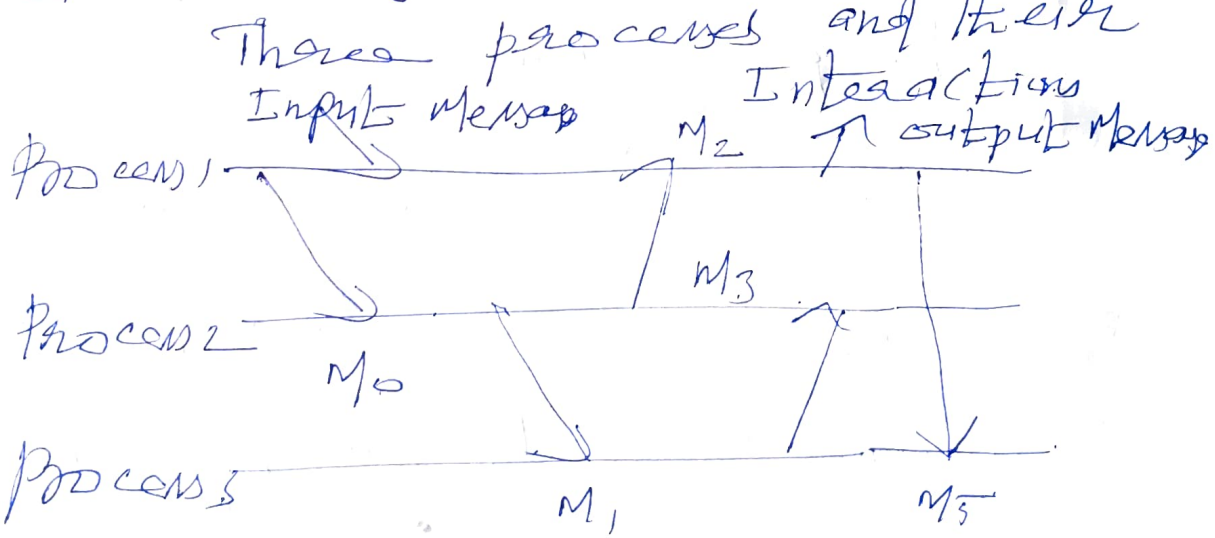
The Agreement condition is satisfied because in the $f+1$ rounds there must be at least one round in which no process failed.

Introduction of check pointing and Rollback Recovery.

: Rollback recovery treats a distributed system application as a collection of processes that communicate over a network.

4.8: Background and Definitions

4.8.1. System Model.



. The messages generated by the underlying distributed application are referred to computation messages.

4.8.2 Local checkpoint

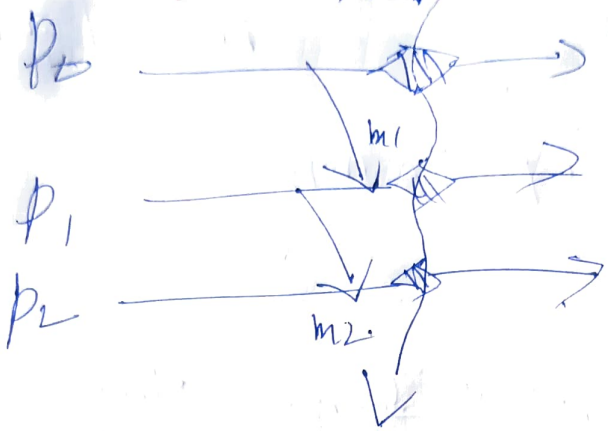
Local checkpoint is a snapshot of the state of the process at a given instance and the event of recording the state of a process is called local checkpointing.

4.9. Consistent set of checkpoints

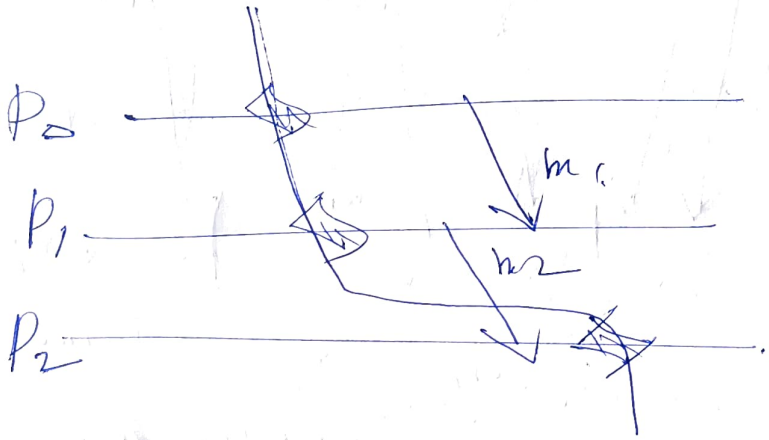
All the sites save their local states : local checkpoints

All the local checkpoints from each site collectively form a Global checkpoint.

Consistent State



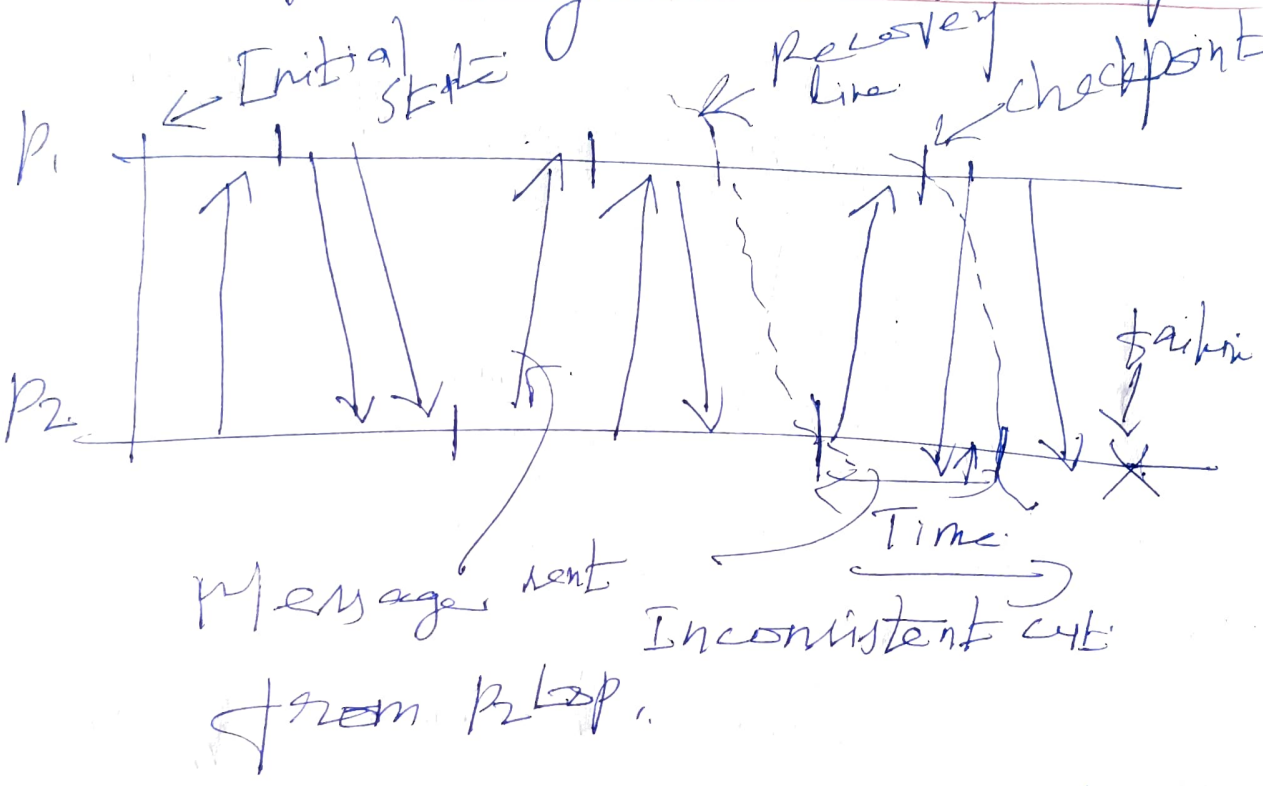
Inconsistent State



The domino effect is caused by orphan messages which in turn are caused by rollbacks.

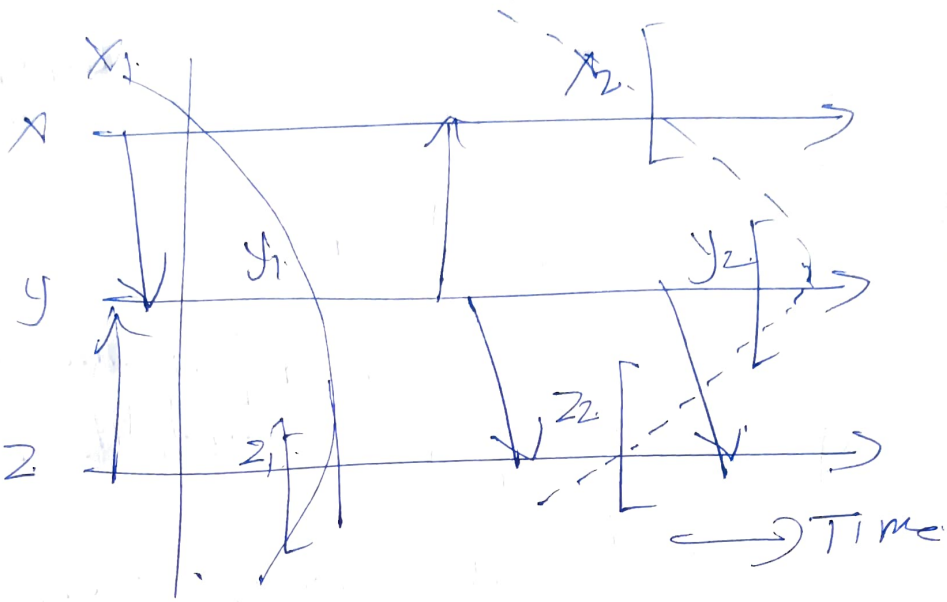
Strongly consistent set of checkpoints.

Recovery line and checkpoint



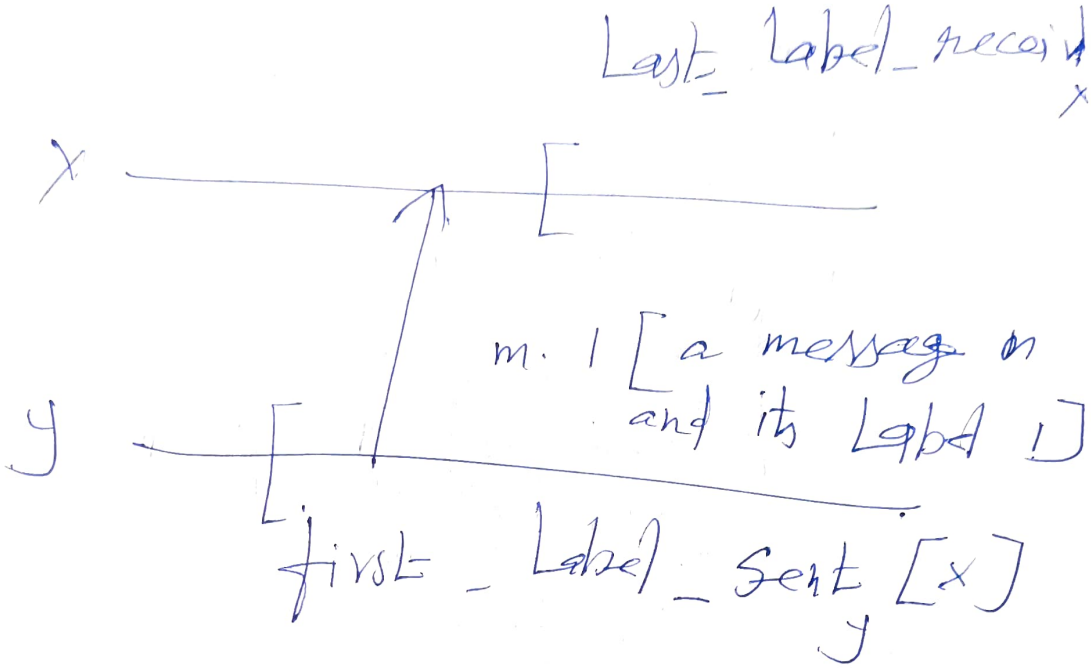
A consistent set of check points corresponds to a consistent global state.

Consistent set of checkpoint



- Set $[x_1, y_1, z_1]$ is a strongly consistent set of checkpoint
- Set $[x_2, y_2, z_2]$ is a consistent set of checkpoint.
- Consistent set of checkpoint
- Similar to the consistent Global state

checkpoint Notation



Each node maintains

1. Monotonically increasing counter with which each message from that node is labelled
2. Records of the last message from and the first message to all other nodes.

Phase two.

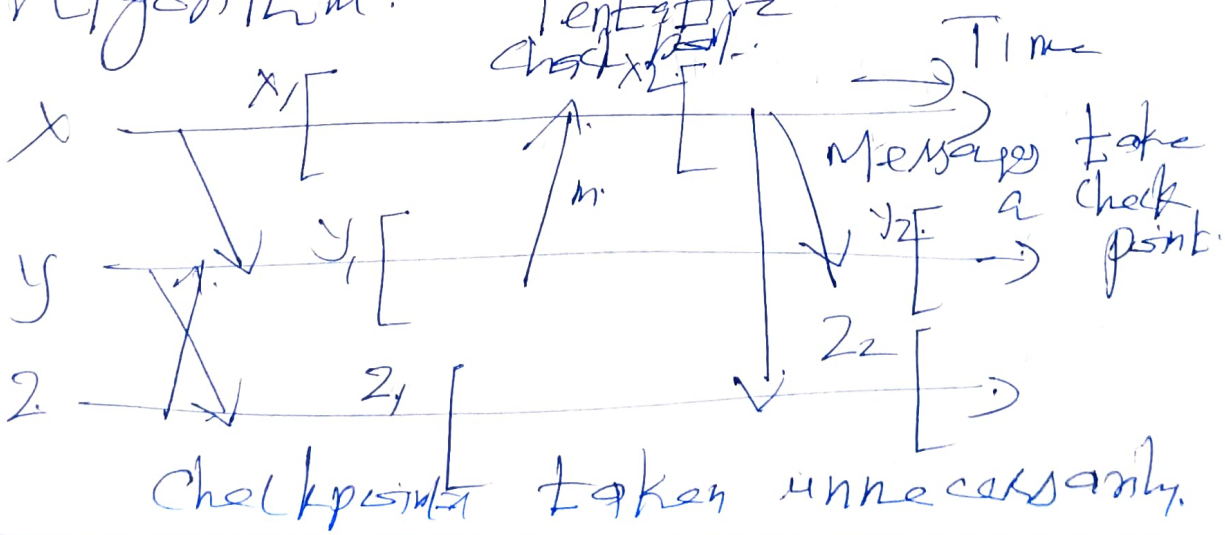
1. P_i propagates its decision to all processes.
2. On receiving the message from P_i, all processes act accordingly.

Synchronous Checkpointing

properties

- All or none of the processes take permanent checkpoints

Optimization of the checkpoint Algorithm.



Checkpoints taken unnecessarily.

Now $[x_2, y_2, z_2]$ forms a consistent set of checkpoints

$[x_2, y_2, z_1]$ also forms a consistent set of checkpoints.

Synchronizing Checkpoints

Disadvantages.

1. Additional Messages must be exchanged to coordinate checkpoints.
2. Synchronization delays are introduced during normal operations.

4.9.2. The Rollback Recovery Algorithm.

Phase one:

. Process P_i checks whether all processes are willing to restart from their previous checkpoints.

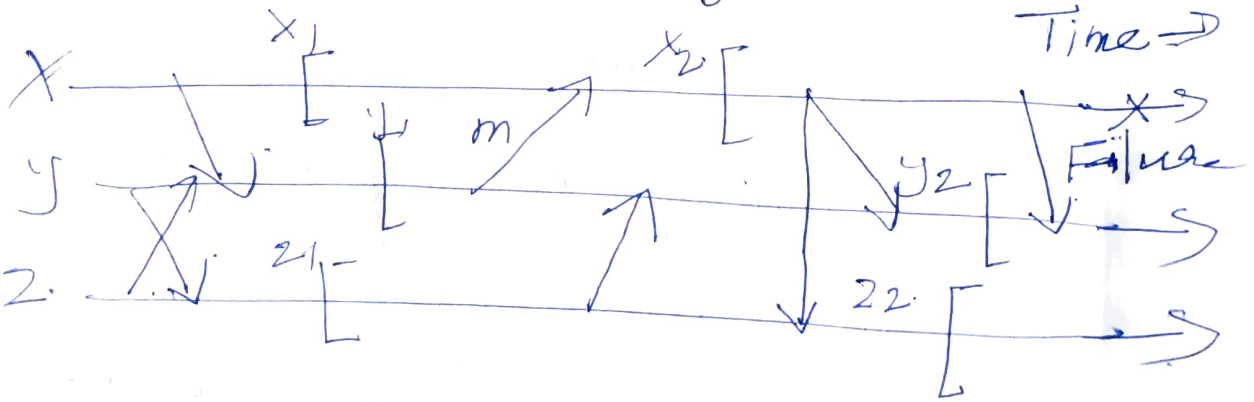
Phase two:

. P_i propagates its decision to all processes.

optimization:

. A minimum number of processes roll back

Unnecessary Rollback.



4.9.3 Message Types

- In-Transit Message
- Lost Messages.
- Delayed Messages.
- Orphan Messages
- Duplicate Messages.

4.10. Issues in Failure recovery

- Recovery refers to restoring a system to its normal operational state.

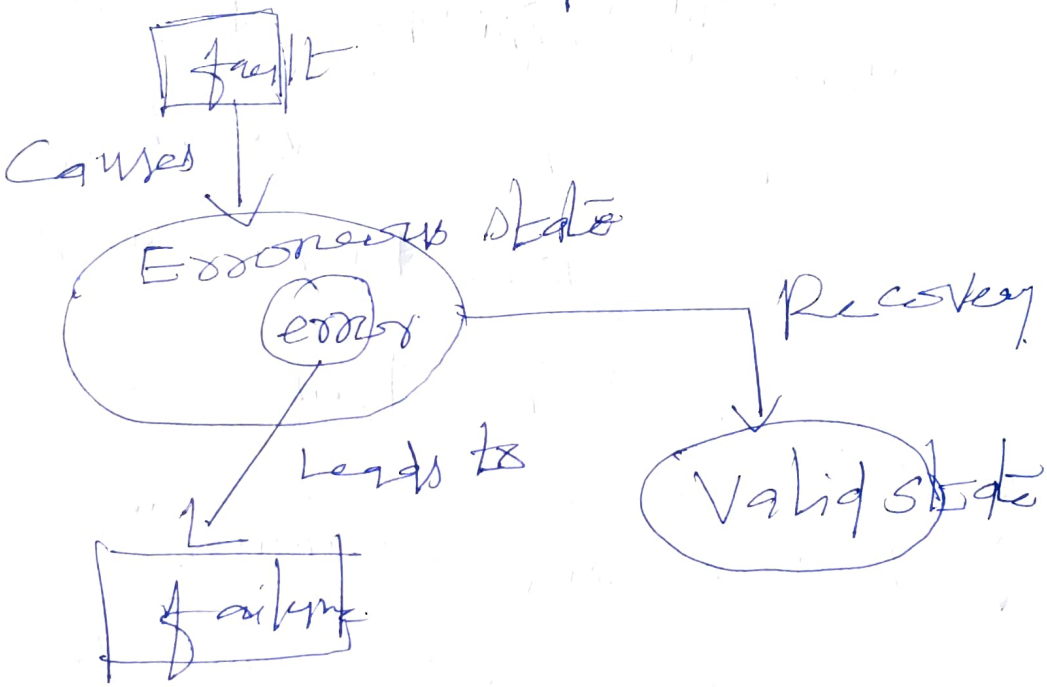
Some solution on process recovery.

1. Reclaim resources allocated to process
2. Undo modification made to database.
3. Restart the Process

4. restart process from point of failure and resume execution.

4.10.1. Basic concepts:

Concepts of recovery.



- System failure
- Erroneous system state
- Error
- Fault

4.11. Check-point based Recovery

Checkpoint recovery is the saving and restoration of system state.

The checkpointing mechanism takes a snapshot of the system state and stores the data on some non-volatile storage Medium.

1. Uncoordinated checkpointing

Advantages:- The lower runtime overhead during normal execution.

Disadvantages:

Domino effect during a recovery.

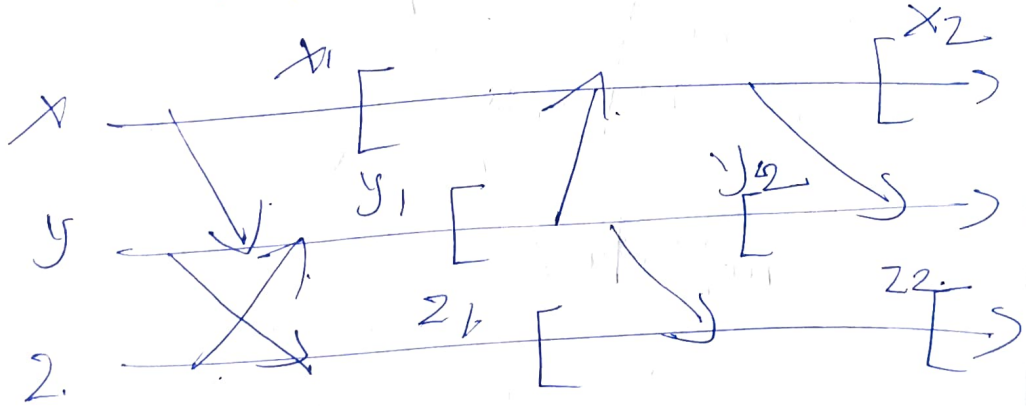
2. Coordinated checkpointing.

Two Approaches.

First - To minimize the number of Synchronization Messages and the number of checkpoints.

Blocking Checkpointing.

Blocking checkpointing



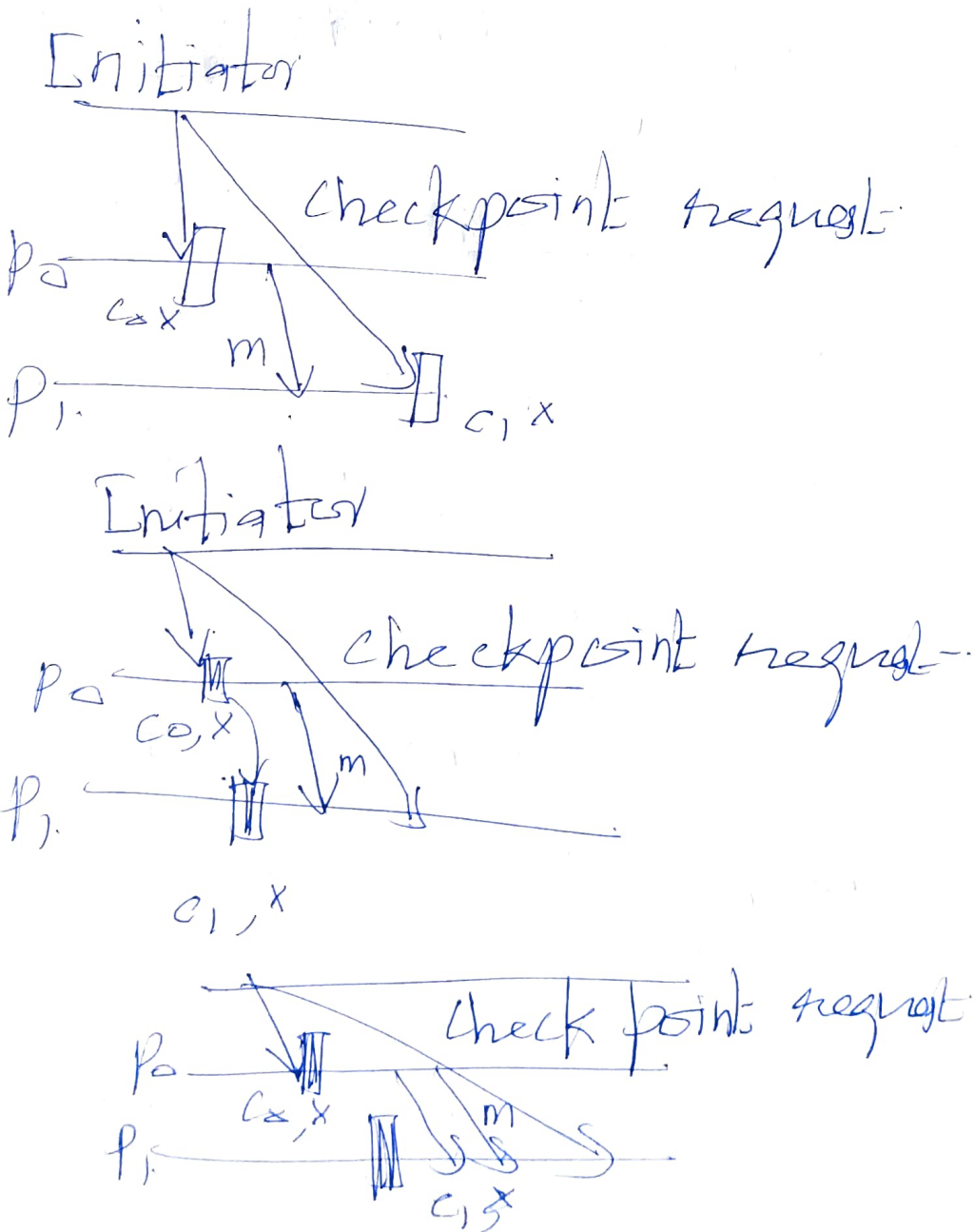
Coordinator takes a checkpoint, broadcasts a message to all processes

Disadvantages:- The computation is blocked during the checkpointing

Non-blocking checkpointing

- The processes need not stop their execution while taking checkpoints.

Non blocking checkpoint



Example of coordinated checkpointing

a) Checkpoint inconsistency

b) Solution with FIFO channels

3. Communication-induced checkpointing

Two types

1. Model based checkpointing
2. Index based communication.

4.11.1. Difference between Uncoordinated, Coordinated and Communication Induced checkpointing.

	parameters	Uncoordinated checkpoint	coordinated checkpoint	Communication induced checkpoint
No. of checkpoint		Many	One	Many

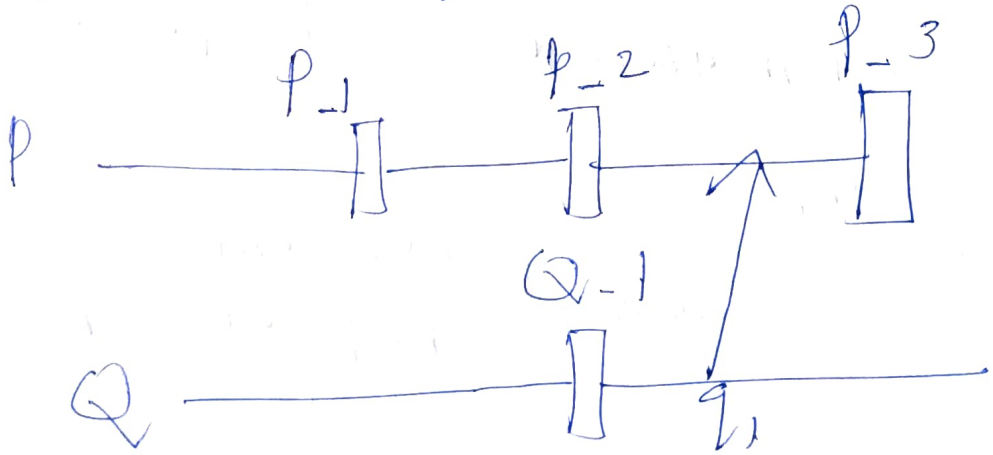
Domino Effect	possible	No	No
Orphan process	possible	No	No

4.12 Coordinated checkpointing Algorithm

1. The Koo-Toueg Algorithm.

2. An Algorithm which Staggers Checkpoint in time.

Koo-Toueg Algorithm



- An algorithm for coordinated checkpointing has two types of checkpoints
  - tentative and permanent.

4.13. Algorithm for Asynchronous checkpointing and Recovery.

Two types of log storage are maintained.

a. Volatile log: short time to access but lost if processor crash. Move to stable log periodically

b. Stable Log: Longer time to access but remains if crashed.

Asynchronous checkpointing:-

After executing an event the triplet is recorded without # any synchronization with other processes.

Recovery Algorithm.

Number of messages received by P_j from P_j from the beginning of computation to checkpoint

Idea: From the set of checkpoints find a set of consistent checkpoints. Using that based on the number of messages sent and received.

Unit V

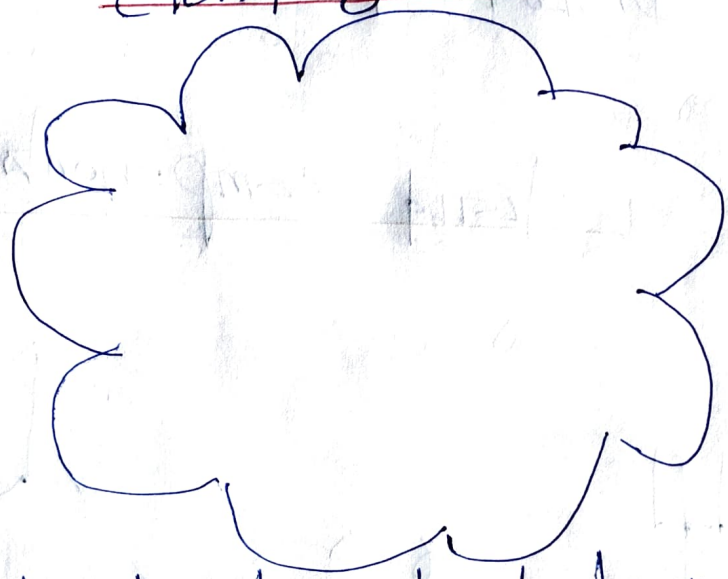
Cloud computing.

Definition of cloud computing - Characteristics of cloud - cloud development Models - cloud service models - Driving factors and challenges of cloud - Virtualization - Load Balancing Scalability and Elasticity - Replication - Monitoring - cloud services and platforms - compute Services - storage services - Application Services.

5.1 Definition of cloud computing.

cloud computing = Software as a service + Platform as a service + Infrastructure as a Service + Data as a service.

cloud symbol.



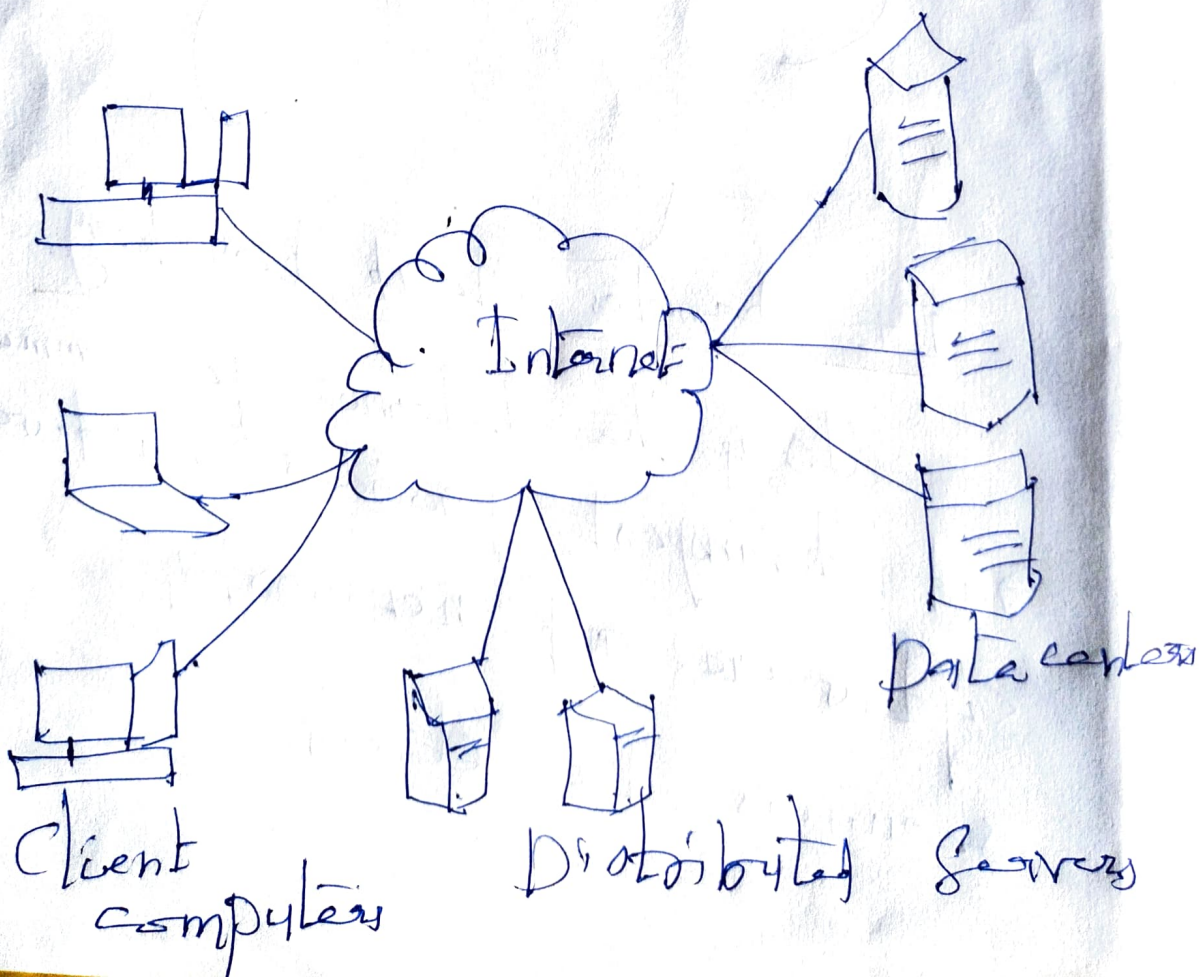
It denotes cloud boundary.

Using the Internet for communication and transport provides hardware software and networking services to clients.

5.1.1. Cloud Components.

1. clients
2. Benefits
3. Data centers
4. Virtualizing Servers
5. Distributed Servers

cloud components



5.1.2 Pros and Cons of cloud computing

Pros of cloud computing.

1. Lowers computer costs.
2. Improved performance.
3. Reduced Software costs.
4. Instant Software updates.
5. Improved document format compatibility.
6. Unlimited storage capacity.
7. Increased data reliability.
8. Universal document Access.
9. Latest version availability.
10. Easier group collaboration.
11. Device Independence.

Cons of cloud computing.

1. It requires a constant Internet connection.
2. Features might be limited.
3. Stored data might not be secure.
4. Does not work well with low speed connections

5 1.3. Application of cloud computing.

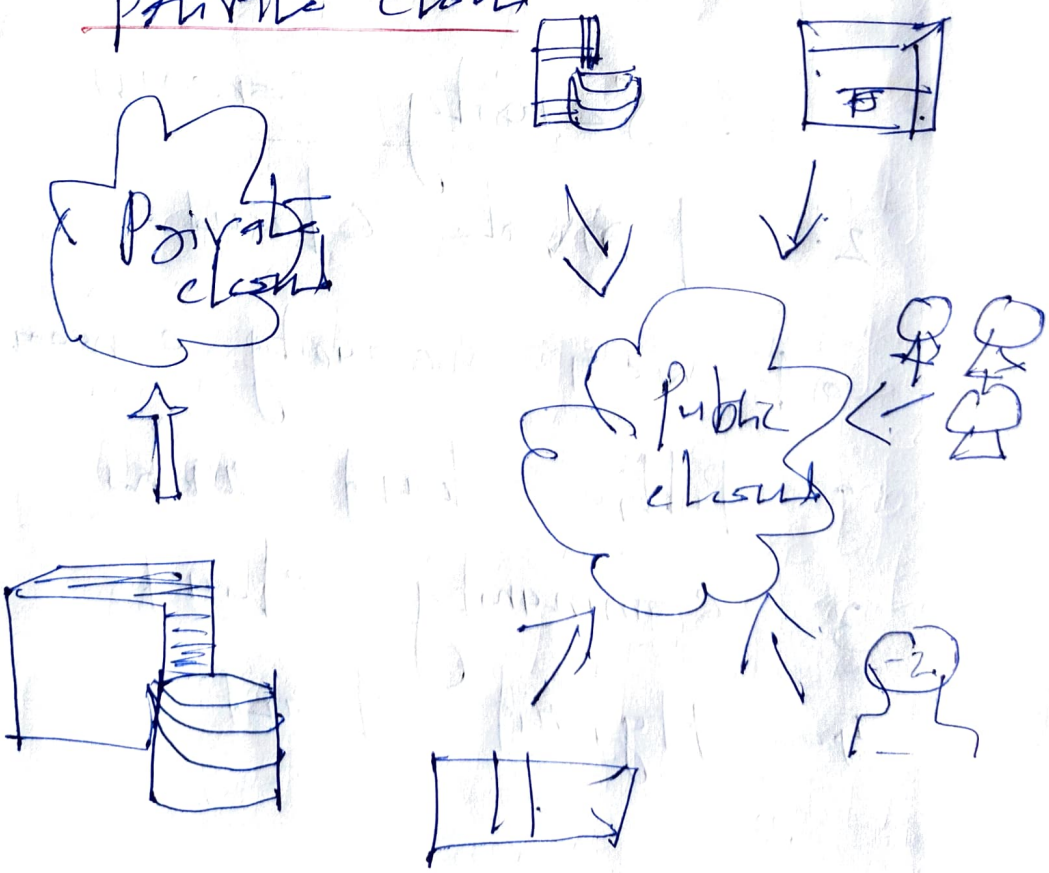
1. Through cloud cost & flexibility online marketplace gains access to more powerful analytics online.
2. Greater business scalability
3. Greater market adaptability
4. Market complexity.
5. context driven variability
6. Ecosystem con

5.2. Characteristics of cloud.

1. On-demand self-service.
2. Ubiquitous network access
3. Location-independent resource pooling.
4. Rapid elasticity.
5. Pay per use.

5.3. Cloud Deployment Models.

private cloud



1. Public cloud.

Examples - Facebook.

Google, Linkedin.

Public Cloud Benefits.

1. Low investment barrier.

Good test environments for applications that scale to many servers

Public cloud risks

1. Security concerns.

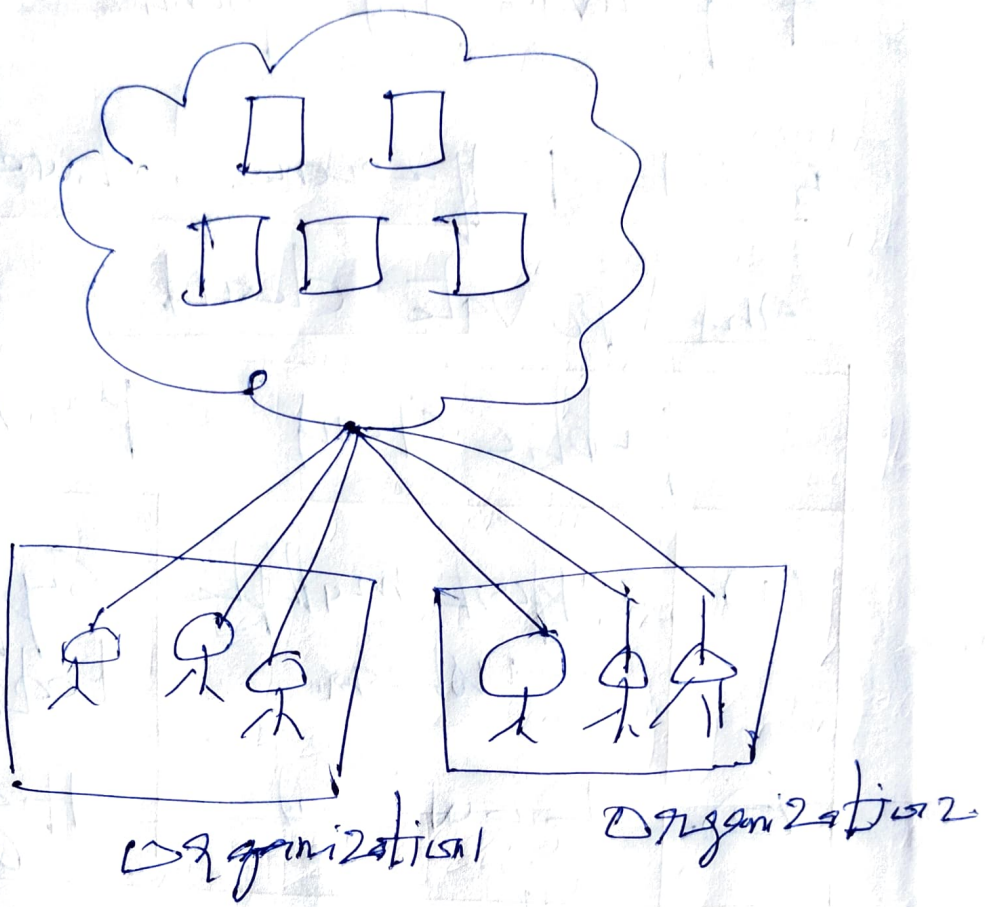
2. Private cloud

a) Fewer security concerns

- Public cloud risks

3. Community cloud

4. Hybrid cloud



Hybrid cloud benefits

a) Operational flexibility.

b) Scalability

Hybrid cloud risks

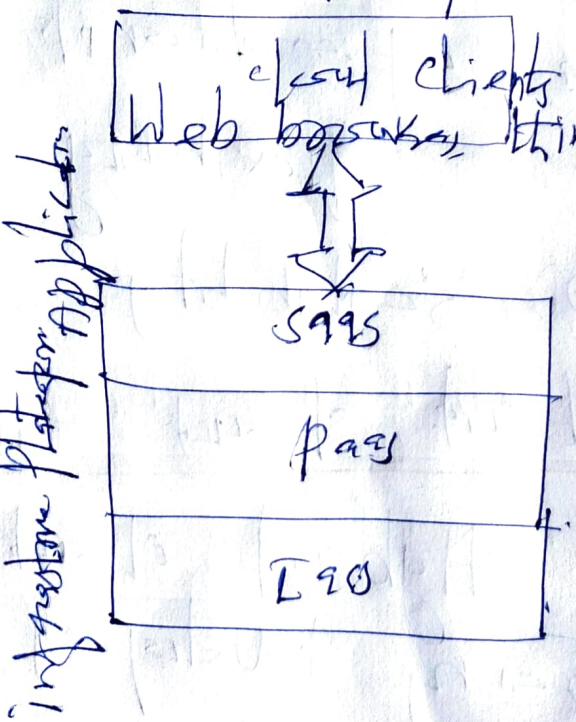
a) Hybrid clouds are still being developed, not many in real life.

6. control of security between private and public clouds

5.3. Difference between public and private cloud.

	public cloud	private cloud
1	support multiple customers.	Support dedicated customers
2.	Low cost	High cost.

5.4. Cloud Service Models Cloud computing stack.

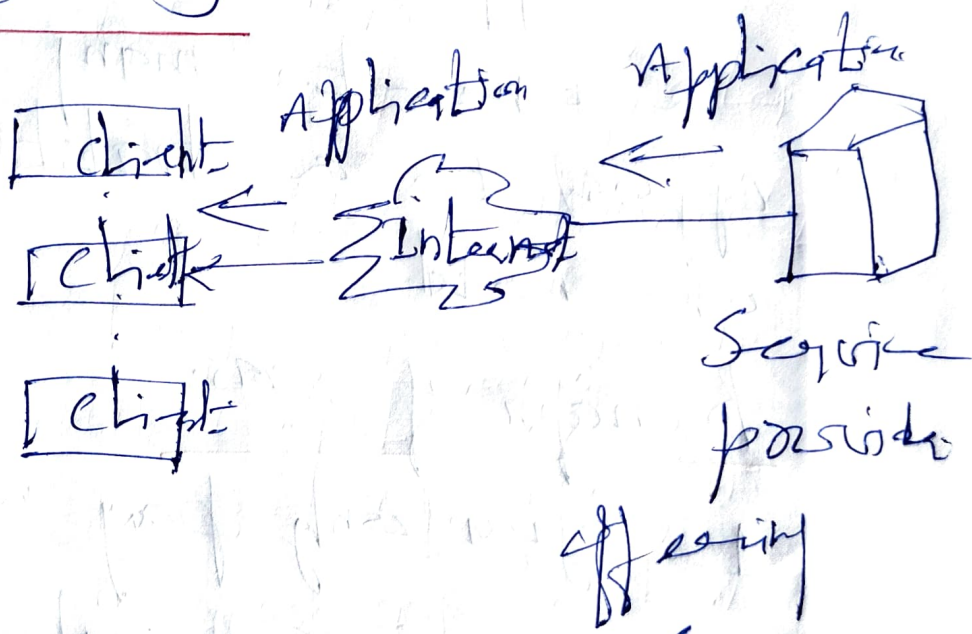


Flavors of cloud computing

1. saas designed for end-users
2. paas is to make coding
3. Iaas is the hardware and software

5.4.1. Software as a Service.

[saas]



The saas concept can be defined as providing robust Web based on demand software storage and various applications

Characteristics of SaaS.

1. Software applications or services are stored remotely.
2. A user accesses services via the internet.
3. Application delivery from a one-to-many model as opposed to a traditional one to one model.

Benefits of SaaS.

1. You only pay what you use.
2. Easier collaboration.
3. Compatibility

5.4.2 Platform as a Service (PaaS)

Platform as a service is another application delivery model also known as Cloud-ware

Service include: Application design, development, testing, deployment and hosting, team collaboration and versioning

Some examples - Google's App Engine or Force.com

PaaS include components:

1. pay contrary to billing
2. Browser based development studio:

Characteristics of PaaS

1. It support multi-tenant architecture.
2. It support for development & group collaboration.

Benefits of paas

1. Reduced Capital Expenditure
2. Reduced skill requirements
3. support of team collaboration

5.4.3

Infrastructure as a service

[IaaS]

IaaS

Service providers rents

these resources

1. Serving space.
2. Network equipment
3. Memory.

Examples - Amazon EC2, Google

Iaas Layer types

1. Physical Server
2. Dedicated virtual Server
3. Shared virtual Server

Advantages of Iaas

1. Reduced hardware cost.
2. Reduction of IT staff.

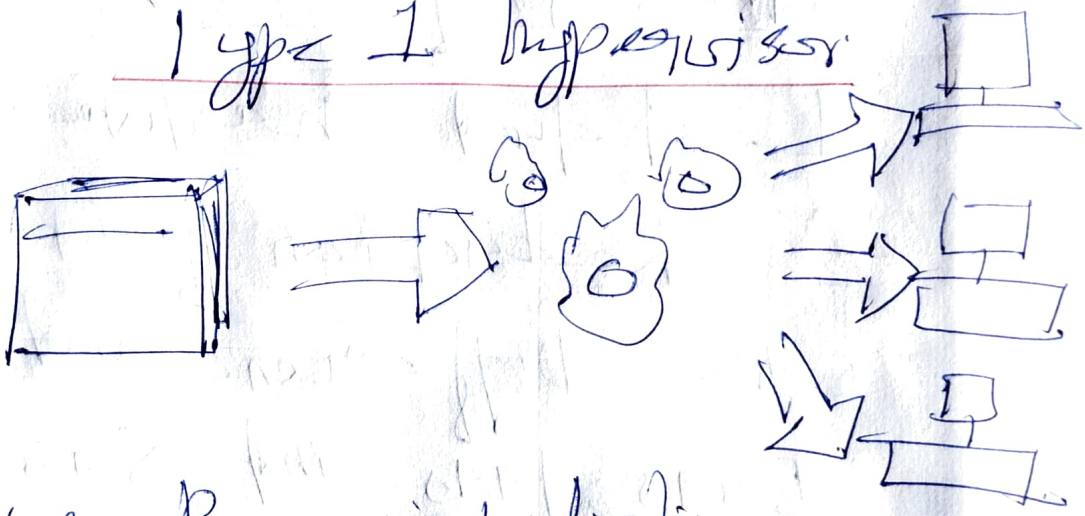
5 M. Difference between Iaas, Paas and Saas

Iaas	Paas	Saas
automated and scalable environments	Quickly developing and deploying applications	Applications available through the internet.

5.5. Driving factors of challenges of cloud.

1. Increased security vulnerabilities
2. Limited portability between cloud providers

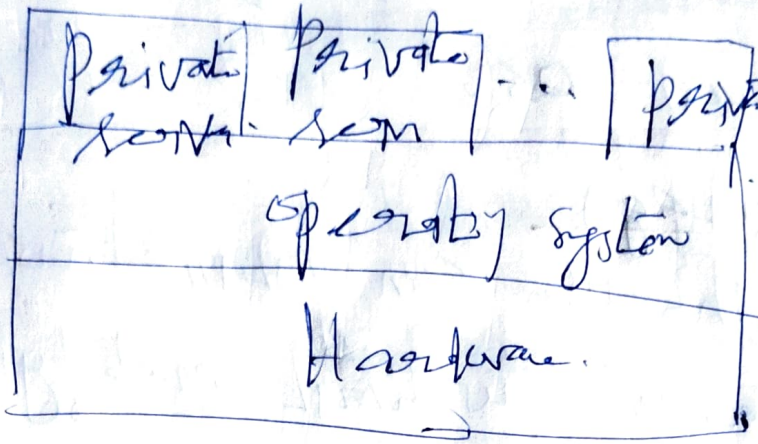
Type 1 hypervisor



5.6.2. Paravirtualization

- Helps to communication between the guest OS and the hypervisor to improve efficiency.

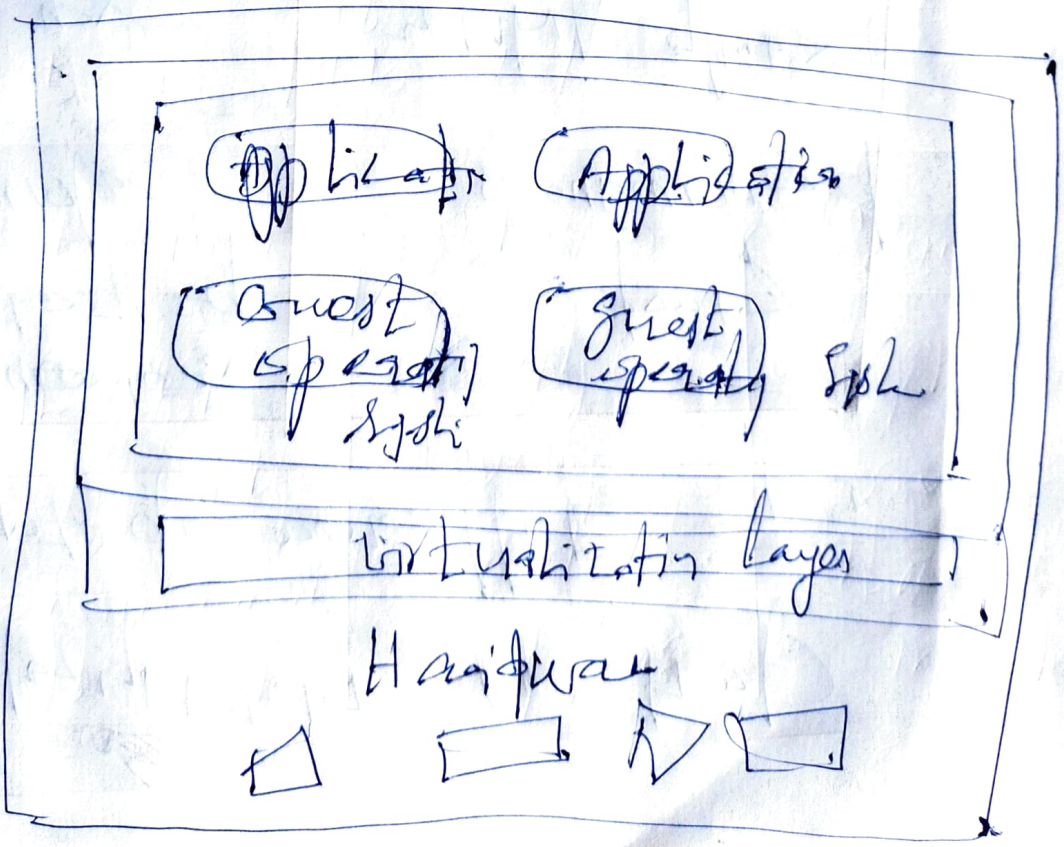
5.6.2. Para virtualization architecture



Problems with para virtualization

1. Para virtualized system won't run on native server

5.6.3. Full virtualization



it doesn't need to modify the host OS.

Advantages

1. The user can install this VM architecture without modifying the host OS.

5.6.4. Difference between Full virtualization and Para virtualization

Full virtualization	Para virtualization
Example VMWare	Xen architecture.
performance is slow.	performance is better in certain cases

5.6.5. Difference between Cloud and virtualization.

S.No	Virtualization	Cloud Computing
1	Instance storage is persistent	Instance storage is shortly lived

5.16. pros and cons of virtualization.

a) Pros.

1. Reduced cost.
2. Improved performance of IT resource

b) Cons.

1. Not all hardware and software can be virtualized.

5.17. Load Balancing

— means to achieve effective resources and utilization.

Load balancing can be divided into 3 types

1. Centralized approach
2. Distributed "
3. Mixed approach.

Metrics for Load balancing in Clouds

1. Throughput
2. Fault tolerance
3. Migration time
4. Response time.
5. Resource utilization
6. Scalability
7. performance.
8. Scalability and Elasticity

Scalability — is the ability of a system or network to handle increased load or usage

Cloud elasticity to manage available resources according to the current workload requirements dynamically.

5g. Replication.

refers to the process of replicating data from one premises storage of the cloud.

Types of replication
1. array based
2. Network based
3. host based

5.10 Monitoring

— services allow cloud users to collect and analyze the data on various monitoring Metrics

Device Types	Metric
CPU	CPU usage/Cpsec
Memory	Memory used

5.11. cloud services and platforms

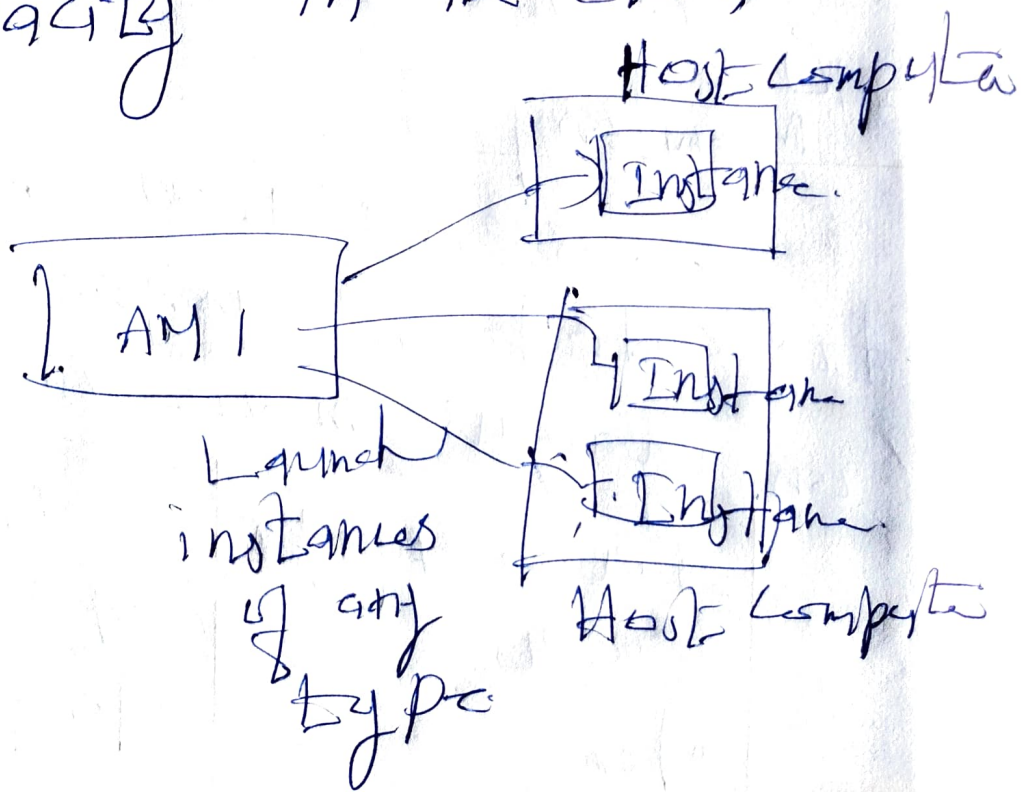
Compute services

Features

a. Scalable
b. Secure
c. cost effective.
d. Flexible.

5.11.1. Amazon Elastic Compute cloud.

is a web service that provides resizable compute capacity in the cloud



11.

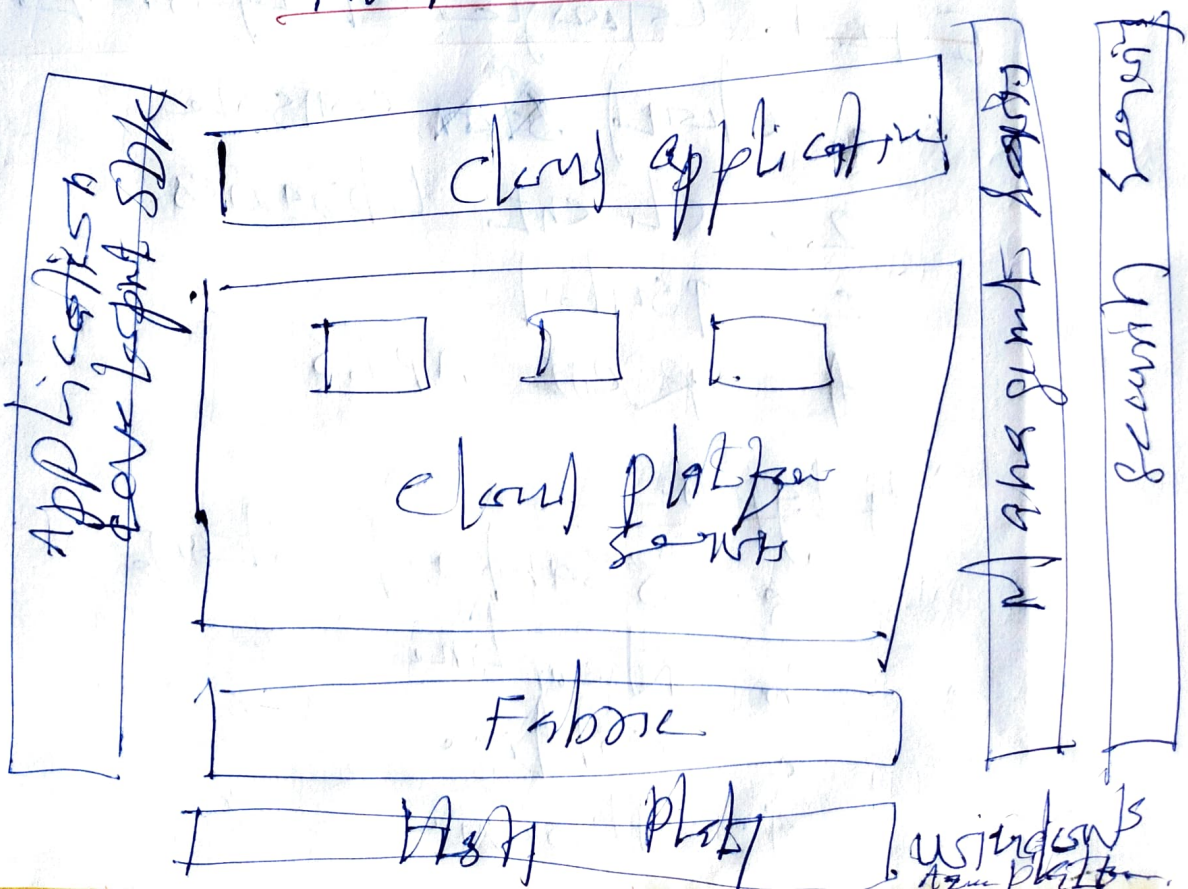
5.11.2. Windows Azure.

- is a cloud computing platform and infrastructure

Azure has three components

1. compute
2. Storage
3. App Fabric

Windows Azure platform Architecture



Advantages

1. Microsoft Azure offers high availability.
2. It is cost effective solution for an IT budget.

5.12 Storage Service.

Amazon S3 defines a bucket name as a series of one or more labels separated by periods

5.12.1. Google cloud storage

a. cloud console
2. client libraries.
c. gsutil
d. REST APIS

4 types.

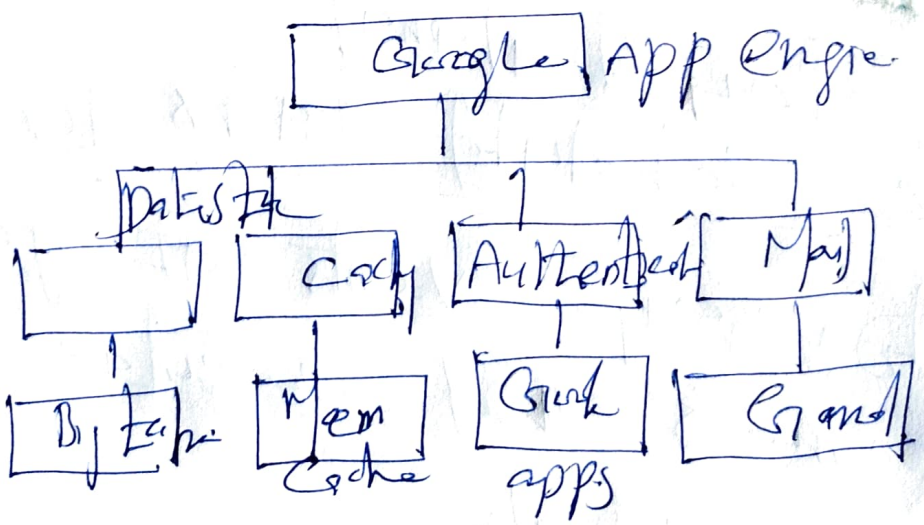
a. Standard Storage
b Nearline "
c. Coldline "
d. Archive "

5.13. Application Services
— Google Cloud Endpoints, Google Cloud SQL, Google Cloud Datastore, Google Cloud Storage, Google Prediction API, Google BigQuery, Google Cloud Messaging, Google Cloud Bigtable, Google Cloud Dataflow, Google Cloud Dataproc, Google Cloud Pub/Sub, Google Cloud Debugger, Google Cloud Trace, Google Cloud Monitoring, Google Cloud Logging, Google Cloud Deployment Manager, Google Cloud Test Lab, Google Cloud Source Repositories, Google Container Registry, Google Container Engine, Google Compute Engine, Google App Engine, Google Cloud Platform, Google Cloud Shell, Google Cloud Console, Google Cloud Mobile App, Google Cloud Machine Learning, Google Cloud Vision API, Google Cloud Speech API, Google Cloud Natural Language API, Google Cloud Translation API, Google Cloud Video Intelligence API, Google Cloud Jobs API, Google Cloud Dialogflow Enterprise Edition, Google Cloud Spanner, Google Cloud Memorystore, Google Cloud Firestore, Google Cloud Functions, Google Cloud Run, Google Cloud Build, Google Cloud Scheduler, Google Cloud Tasks, Google Cloud Pub/Sub, Google Cloud Storage, Google Cloud Filestore, Google Cloud Persistent Disk, Google Cloud Local SSD, Google Cloud Transfer Service, Google Cloud Interconnect, Google Cloud VPN, Google Cloud Router, Google Cloud Armor, Google Cloud IAM, Google Cloud Resource Manager, Google Cloud Billing, Google Cloud Support, Google Cloud Partner Interconnect, Google Cloud Dedicated Interconnect, Google Cloud CDN, Google Cloud DNS, Google Cloud Load Balancing, Google Cloud Network Service Tiers, Google Cloud Premium Network Tier, Google Cloud Standard Network Tier, Google Cloud Network Intelligence Center, Google Cloud Network Topology, Google Cloud Network Monitoring, Google Cloud Network Security, Google Cloud Network Management, Google Cloud Network Automation, Google Cloud Network Programmability, Google Cloud Network Virtualization, Google Cloud Network Functions Virtualization, Google Cloud Network Slicing, Google Cloud Network Orchestration, Google Cloud Network Analytics, Google Cloud Network Optimization, Google Cloud Network Planning, Google Cloud Network Design, Google Cloud Network Engineering, Google Cloud Network Operations, Google Cloud Network Support, Google Cloud Network Training, Google Cloud Network Certification, Google Cloud Network Partner Program, Google Cloud Network Marketplace, Google Cloud Network Solutions, Google Cloud Network Reference Architectures, Google Cloud Network Whitepapers, Google Cloud Network Blog, Google Cloud Network Community, Google Cloud Network Forum, Google Cloud Network Stack Overflow, Google Cloud Network GitHub, Google Cloud Network Slack, Google Cloud Network Twitter, Google Cloud Network LinkedIn, Google Cloud Network YouTube, Google Cloud Network Podcast, Google Cloud Network Webinar, Google Cloud Network Event, Google Cloud Network Conference, Google Cloud Network Meetup, Google Cloud Network User Group, Google Cloud Network Developer Group, Google Cloud Network Partner Group, Google Cloud Network Customer Group, Google Cloud Network Advisory Board, Google Cloud Network Steering Committee, Google Cloud Network Working Group, Google Cloud Network Special Interest Group, Google Cloud Network Task Force, Google Cloud Network Initiative, Google Cloud Network Program, Google Cloud Network Project, Google Cloud Network Service, Google Cloud Network Product, Google Cloud Network Feature, Google Cloud Network Capability, Google Cloud Network Benefit, Google Cloud Network Value, Google Cloud Network Differentiator, Google Cloud Network Advantage, Google Cloud Network Strength, Google Cloud Network Weakness, Google Cloud Network Opportunity, Google Cloud Network Threat, Google Cloud Network Risk, Google Cloud Network Mitigation, Google Cloud Network Contingency, Google Cloud Network Recovery, Google Cloud Network Continuity, Google Cloud Network Resilience, Google Cloud Network Reliability, Google Cloud Network Availability, Google Cloud Network Performance, Google Cloud Network Scalability, Google Cloud Network Flexibility, Google Cloud Network Agility, Google Cloud Network Innovation, Google Cloud Network Leadership, Google Cloud Network Vision, Google Cloud Network Strategy, Google Cloud Network Roadmap, Google Cloud Network Plan, Google Cloud Network Goal, Google Cloud Network Objective, Google Cloud Network Key Result, Google Cloud Network Metric, Google Cloud Network Indicator, Google Cloud Network Measure, Google Cloud Network Target, Google Cloud Network Baseline, Google Cloud Network Benchmark, Google Cloud Network Standard, Google Cloud Network Best Practice, Google Cloud Network Guideline, Google Cloud Network Policy, Google Cloud Network Procedure, Google Cloud Network Process, Google Cloud Network Workflow, Google Cloud Network Automation, Google Cloud Network Orchestration, Google Cloud Network Integration, Google Cloud Network Interoperability, Google Cloud Network Compatibility, Google Cloud Network Portability, Google Cloud Network Migration, Google Cloud Network Modernization, Google Cloud Network Transformation, Google Cloud Network Evolution, Google Cloud Network Revolution, Google Cloud Network Disruption, Google Cloud Network Innovation, Google Cloud Network Creativity, Google Cloud Network Imagination, Google Cloud Network Inspiration, Google Cloud Network Motivation, Google Cloud Network Passion, Google Cloud Network Commitment, Google Cloud Network Dedication, Google Cloud Network Loyalty, Google Cloud Network Trust, Google Cloud Network Reputation, Google Cloud Network Brand, Google Cloud Network Image, Google Cloud Network Identity, Google Cloud Network Culture, Google Cloud Network Values, Google Cloud Network Mission, Google Cloud Network Vision, Google Cloud Network Purpose, Google Cloud Network Goal, Google Cloud Network Objective, Google Cloud Network Key Result, Google Cloud Network Metric, Google Cloud Network Indicator, Google Cloud Network Measure, Google Cloud Network Target, Google Cloud Network Baseline, Google Cloud Network Benchmark, Google Cloud Network Standard, Google Cloud Network Best Practice, Google Cloud Network Guideline, Google Cloud Network Policy, Google Cloud Network Procedure, Google Cloud Network Process, Google Cloud Network Workflow, Google Cloud Network Automation, Google Cloud Network Orchestration, Google Cloud Network Integration, Google Cloud Network Interoperability, Google Cloud Network Compatibility, Google Cloud Network Portability, Google Cloud Network Migration, Google Cloud Network Modernization, Google Cloud Network Transformation, Google Cloud Network Evolution, Google Cloud Network Revolution, Google Cloud Network Disruption, Google Cloud Network Innovation, Google Cloud Network Creativity, Google Cloud Network Imagination, Google Cloud Network Inspiration, Google Cloud Network Motivation, Google Cloud Network Passion, Google Cloud Network Commitment, Google Cloud Network Dedication, Google Cloud Network Loyalty, Google Cloud Network Trust, Google Cloud Network Reputation, Google Cloud Network Brand, Google Cloud Network Image, Google Cloud Network Identity, Google Cloud Network Culture, Google Cloud Network Values, Google Cloud Network Mission, Google Cloud Network Vision, Google Cloud Network Purpose

5.13.1. Application Framework and Runtime — Google App Engine

— is a platform as a service cloud computing platform for developing and hosting web applications in Google Mange data centers

Major features of Google App Engine.



API access through services

1. Mail
2. Memcache
3. Image Manipulation

Key features of GAE

programming mode using java and Python.

1. Python.
2. Java

App Engine includes the features

a. Dynamic web serving
   Automatic scaling and load balancing
c. Scheduled tasks for triggering events at specified times and regular intervals

5.15.2. Queuing Service

Amazon Simple Queue Service

- is a fully managed message queuing service that enables users to decouple and scale microservices, distributed systems and serverless applications.

SQS offers two types of message Queues

1. Standard Queues offer maximum throughput, best effort ordering and

at least once delivery

- SQS FIFO Queues are designed to guarantee that messages are processed exactly once, in the exact order that they are sent.