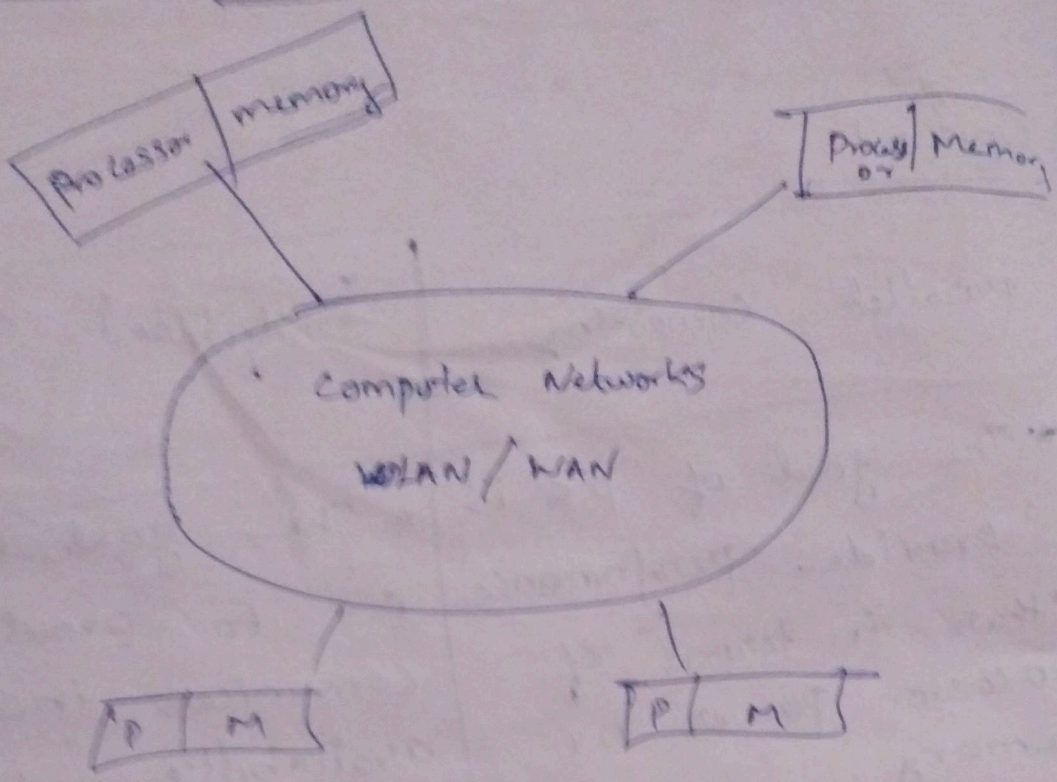
DC

Unit - 1

Parallel Computing	Distributed computing
* The goal of pc is to provide performance either in terms of processor power or memory.	* The goal of DC is to provide convenices includes availabilty, relabiliy and physical distribution
* In PC distribution is frequent	* In DC distribution is infrequent
* It is lower overhead.	* It is heavier weight
*ASSumed to reliable	*Assumed to unreliable
* Short time execution	* long time execution

2. Computer System Components.



3. Disadvantages of Distributed System.

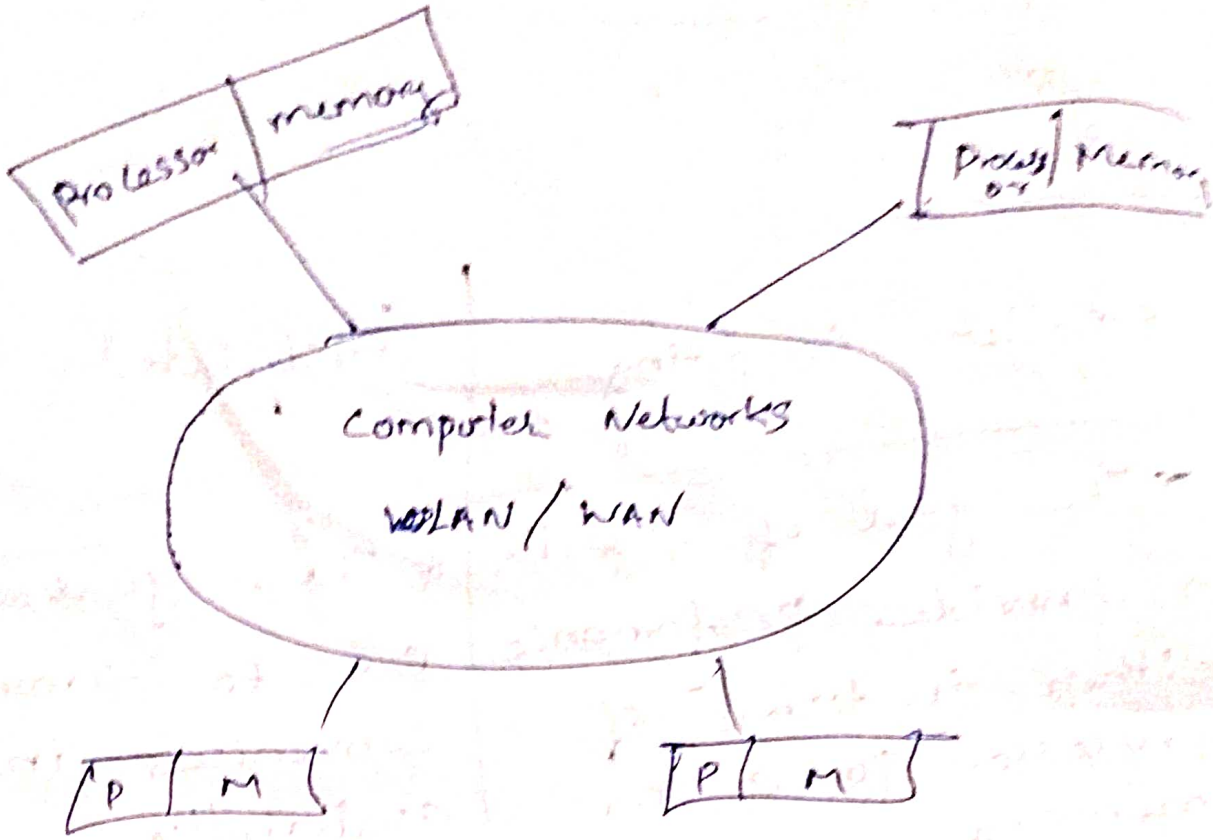
* Software : Difficult to develop Software for ds

* Network : Saturation, lossy transmission.

* Security : Easy to access the Secret data

* Absence of global clock.

2. Computer System Components.



3. Disadvantages of Distributed System.

* Software : Difficult to develop Software for ds.
* Network : Saturation, lossy transmission.
* Security : Easy to access the Secret data
* Absence of global clock.

4. Need of DS :

* Resource Sharing (eg): Printers, disks)
* Security
* Reliability
* consistency of replicated data
* fault tolerancy

5) Message - passing and shared memory :

* Message passing :

* Two processes communicates with each other by passing messages
* The messages can be both direct and indirect
* Indirect communications uses mailbox for sending and recieving Messages.
* Message passing used as a method of Communication in microkernels.

* Message passing System requires Synchronization and communication between the two processes.

* Messages Sent by Process can be either fixed or variable in size.

* The actual function of message passing is normally

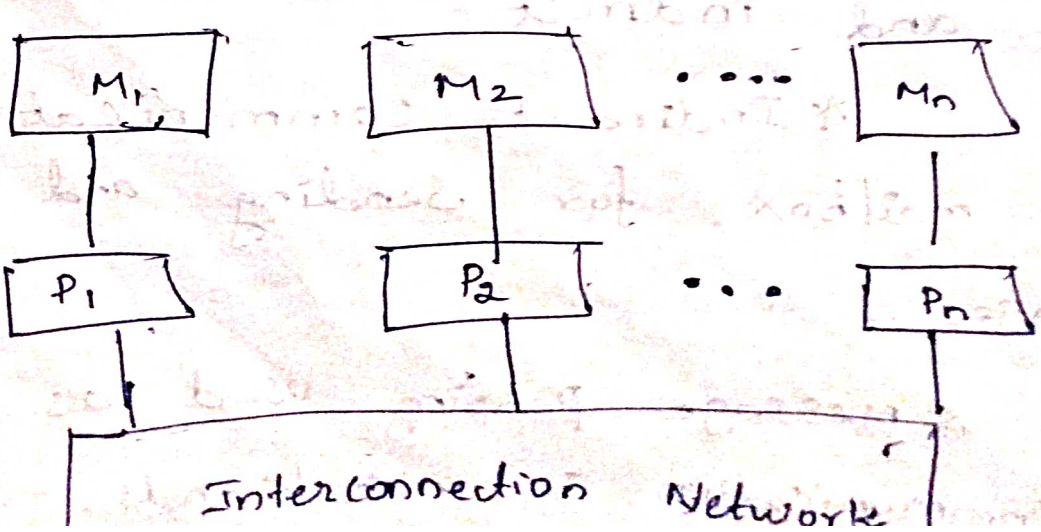
i) Send (destination_name, Msg)

ii) Recieve (source_name, msg)

* Send primitives is used for Sending a message to destination

* A process recieves information by executing the receive primitive

Diagram:



* This techinque can be used in heterogenous computers

Shared Memory :

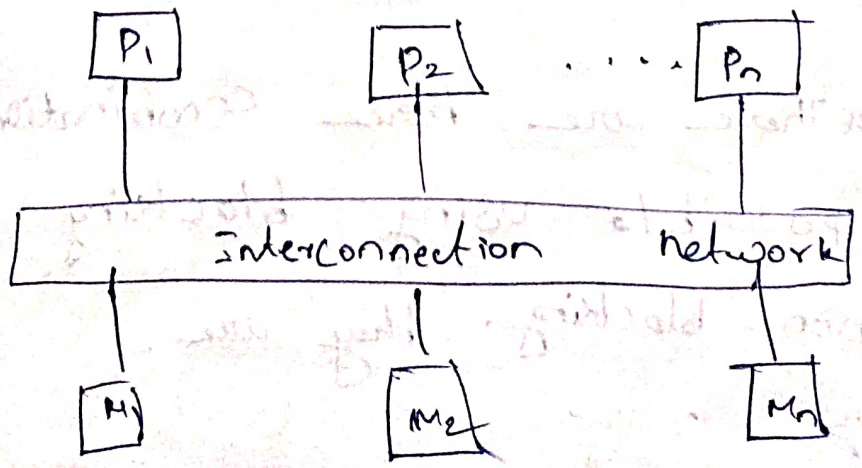
* Shared memory uses common Shared memory for data sharing

* communication takes place via Shared data variables

* It follows faster communication Strategy when compared to message passing technique.

* It is typically implemented using System calls.

* In Shared memory region, the information can by exchange by reading and writing data.



6. Primitives for DC:

* Message Communication primitives are denoted by Send() and receive().

* Message passing primitive commands

SEND (Msg, dest)

REICEIVE (src, buffer)

*Send primitives uses two options for sending data:

*Buffered

*unBuffered

* In buffered option, user data is copied in the kernel buffer. In unbuffered option data gets copied directly from the user buffer to the network

* There are three combinations are possible using blocking and non-blocking they are.

i) Blocking Send, Blocking Recieve

ii) Non Blocking Send, blocking Recieve

iii) Non Blocking Send, Non blocking Recieve

i) Blocking Send, Blocking Recieve:

* In this method both Sender() and Receiver() are blocked Untill the message is delivered.

* This is Called Rendezvous.

* This Combination allows for tight Sychronization between processes.

ii) Non Blocking Send, Blocking Recieve:

* Sender may continue on, the receiver is blocked untill message arrives.

iii) Non Blocking Send, Non-Blocking Receive:

Sending Process Send message and resume the operation. Receiver receives either a valid Message Or Null Message.

7. Synchronous Execution:

* Synchronous execution means the first task in a program must finish before processing the next tasks.

* Lower and upper bounds on execution time of processes can be set.

* Transmitted messages can be received within a known time bound.

* Drift rates between local clocks have a known bound

* In Synchronous distributed system there is a notation of global physical time.

* only Synchronous distributed systems have predictable behaviour in terms of time.

* In Synchronous distributed system it is possible to set timeouts in communication link and failure of process

* It is difficult and costly to implement Synchronous distributed systems.

Asynchronous Execution:

* Asynchronous execution means a second task can begin execution before the completion of first task

* No bounds on process execution time.

* No bound on message transmission delay

* No bounds on drift takes between local clocks

* No timeouts can be used

* Asychronous distributed systems are unpredicatable in terms of timing

* Asychronous are widely used for pratice and failure detection

8. Design Issues and Challages:

Challenges from System Perspective:

* communication mechanisms:

This task involves designing appropriate mechanism for communication among processes in the network.

* processes:

Issues involved are Code migration, thread management at client and Server.

Challenges :

1. Heterogeneity 2. openness 3. Security
4. Scalability 5. Failure handling
6. Concurrency 7. Transparency.

* Heterogeneity :

modern distributed systems are highly heterogeneous in many dimensions, including available bandwidth, processor speed, disk capacity, security, failure rate. It applies.

1. Computer Network - LAN, WAN, wireless Network, Satelite link
2. Computer hardware - Laptop, Computer, Mobile
3. Operating System : Linux, UNIX, Windows
4. programming language: C, C++, Java, PHP

Openness:

* openness means that the system can be easily extended and modified.

* openness means ability to plug and play.

* The integration of new components must have the ablity to communicate with already exist components in the system.

Security:

* Security becomes even more important in distributed systems.

* Authentication, authorization, digital signatures, encryption and privacy become major issues in the destributed system

* Scalability:

* A System is said to be Scalable if it can handle the addition of users and resource without buffering or loss of performance.

Scaling techniques

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

* Failure Handling:

* Failure in distributed System occurs when their is a fault in program
* Hardware, Software, and networks are not free of failures.

* Techniques for dealing with failure:

1) Detecting failure - finding
2. Masking failure - Hidden
3. Tolerating failure -
4. Recovering failure
5. Redudancy -

* Concurrency:

9. Global State of DS:

* Garbage collection
* Deadlock
* Termination
* Debugging