Unit - 1

i) Primitives of Distributed Computing

i) Definition

* Different communication primitives define how process interact, share and synchronize

ii) commands

* Send (msg, dest.)
* Recieve (src, buffer)

→ Buffered
Data copied in Kernel

→ unbuffered
Data copied from user Kernel

iii) Types of primitives

* Blocking
  * The sender waits until the process is completed before preceeding
  * process sends a message and wait for ACK.

  Eg: Mutex

* Non Blocking
  * Sender doesn't wait for the process to complete and can proceed with other task.
  * Sender continuing process and handle message response later

  Eg: Node.js

* Synchronous
  * Sender & reciever are synchronised
  * Sender waits for reciever to ACK the message
  * It is Two way Interaction.

  Eg: TCP

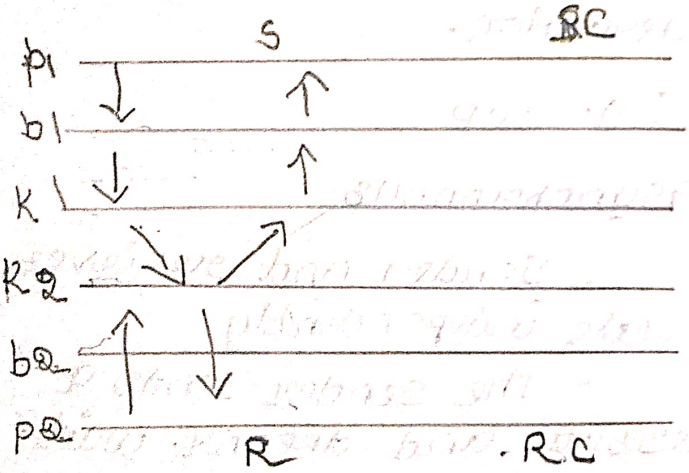
* Asynchronous
  * Sender and reciever operate independently
  * The sender sends a message and doesn't wait for ACK.
  * It continue processing with blocking
  * The reciever can response later.

  Eg: polling (Printer).

iv) Libraries and Standard

* MPI - Message passing Interface
* PVM - Parallel Virtual machine
* RPC - Remote procedure call

* DCE - Distributed computing environment
* RMI - Remote method Interface
* ROI - Remote object Invocation
* CORBA - common object Req Broker architecture
* DCOM - Distributed component Object model.



2) Design issues and challenges in Distributed System

i) Definition

Designing in DS doesn't come for free, some challenges need to overcome in order to build a ideal systems.

HOSS FCT

ii) Design issues

* Heterogeneity

Modern systems are heterogenous

Consist of diverse hardware, OS, network make difficult to ensure compatibility & seamless communication.

It is unavoidable.

Eg: E-Commerce platform

* openness

makes system easily extendable & modified

It achieves by using interface.

New components will integrate with old component

Eg: Social media

* Security

Important

Data is transferred over network in different Domains leads to data Breach, Man-in middle attacks, DOS.

Denial of service refers to the computer or resource is unavailable to users.

Eg: Online Banking

* Scalability

additional computer can be added in order to host additional components.

To control cost of physical resource.

Prevents software resource running out.

Eg: Netflix streaming

* Failure Handling

Operation continues even in failure called fault tolerant.

DS can maintain availability at low level of hardware, software.

It is difficult because failure is partial.

Eg: E-commerce
(@ Based Overflow)

* ~~Comp~~
* Concurrency

Components executed in concurrent process

Several clients will attempt to access shared resource.

each resource must be safe.

Eg: web server

* Transparency

Refers to concealment of complexity of system from user to app.

Make distributed system as a single unit.

Types:

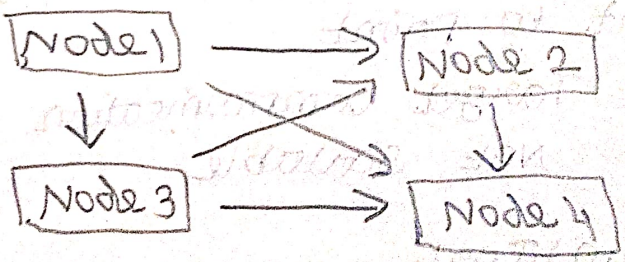
Access
location
replication
Concurrency
failure

Eg: cloud computing

3) Message passing system

i) Definition

It is communication medium where each node communicate with each other to coordinate, synchronize, data sharing



ii) Importance

It enables application to distribute workload, resource, synchronize actions

iii) Types of Message passing

* Synchronous
Interaction b/w Sender and reciever
Timely co-ordination, request / response pattern
Adv: Precise communication
Disadv: latency

* ASynchronous
Independently Sender and Reciever
Decoupled Timing, event driven model
Adv: Responsiveness
Disadv: Need additional mechanism

* Unicast
Single Sender to Single reciever
Direct communication, Point to Point
Adv: Target communication
Disadv: Not Scalable

* Multicast
Single Sender to Multiple reciever
Group Based
Adv: Reduce Traffic
Disadv: complex to implement

* Broadcast
Sending message to all nodes in the network
Wide coverage,
Adv: All recieve msg
Disadv: consumes network.

iv) Protocols

* TCP
Connection oriented
End-End
Data integrity, reliability

* UDP
connectionless
Lightweight
low latency.

* MQTT
M2M
Lightweight
Limited power, Bandwidth

* HTTP
Used for HTML
Essential for Web based communication

v) challenges

* Scalability
* Fault tolerance
* Security
* Message ordering

vi) Example

E-commerce website