Ch-2. System models in distributed systems

__/__/__

→ System that are intended for use in real-world environment should be designed to function correctly in the widest possible range of circumstances. Each type of model is intended to provide an abstract, simplified but consistent description of the relevant aspect of distributed system design.

Distributed system design :-

(i) physical model
(ii) Architectural design
(iii) fundamental model.

Physical model

A physical model is a representation of the underlying hardware elements of a distributed system that abstracts away from its specific details of the computer employed.

ⓐ Base line physical model
ⓑ Early distributed system
ⓒ Interenet scale distributed system
ⓓ Contemporary system
↳ The emergence of mobile and ubiquitous computing has led to a move from discrete nodes to architecture where computers are embedded in everyday objects in the surrounding environments.
Eg- washing machine, mixture

in everyday ...

Ex:- Washing Machine, Mixer

Comparison between generations of distributed system

D. S	Early	Internet-Scale	Contemporary
Scale	Small	Large	Ultra-Large
Heterogeneity	Limited	Significant in terms of platforms	Different styles of Architecture
Openness	Not a priority	Significant priority	Complex
Quality of Service	Small Services	Priority based significant services	High Quality

Architectural Elements

→ System Architectures

To understand fundamental building/model of distributed system we do the following case studies.

i) What are the entities that are communicating in distributed system

(Diagram)

ii) How do they communicate or which are algorithm they use.

iii) What growth and responsibility they have in the overall architecture.

iv) How they are mapped onto the physical distributed infrastructure

1. FIFO

Page Ref: 7, 1, 0, 3, 1, 2, 7, 4

Page frame size: 3.

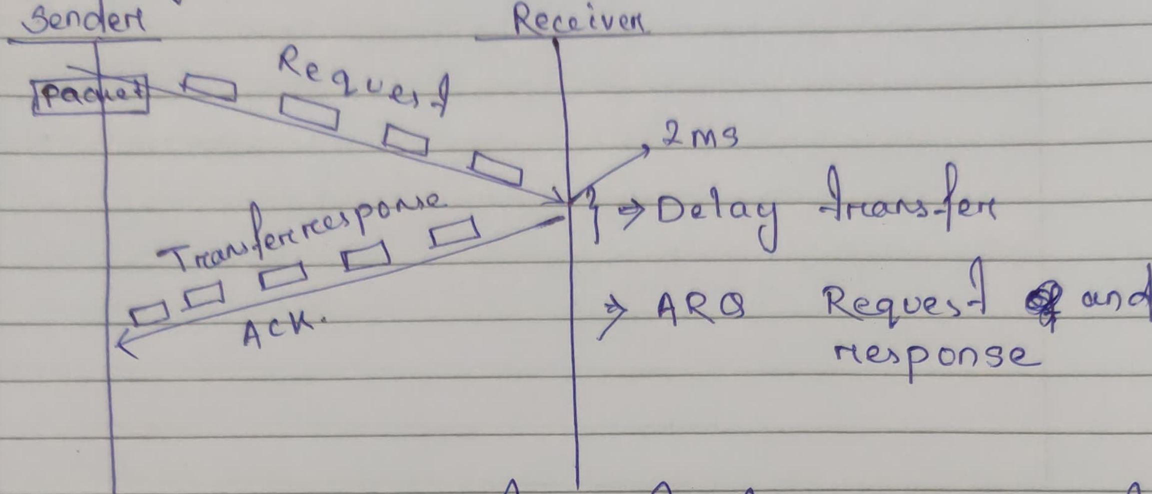
		0	0	0	0	7	7
	1	1	1	1	2	2	2
7	7	7	3	3	3	3	4
1	2	3	4		5	6	7
M	M	M	M		M	M	M

1, 3, 0, 3, 5, 6, 3

		0	0	0	0	3
	3	3	3	3	6	6
1	1	1	1	5	5	5
1	2	3		4	5	6

Fundamental models

Latency :-



Jitter - no. of packets sent at a time and that delay is called jitter.

Latency - Transmission delay b/w the request and transfer response of a process is called latency.

Bandwidth - Bandwidth of a computer system is a total amount of information that can be transmitted over it in a given time. When a large amount of communication channel are using the same network, they have to share the available bandwidth.

Jitter - It is the ~~variable~~ variation in time taken to deliver a series of message.

Eg - If consecutive samples of audio data are played with different intervals, the sound will be badly distorted.

These are all the key factors for communication channels ~~relating~~ ~~to~~ relative to their performance.

_/_/_

Interaction model

→ ds are composed of many process, interacting in complex ways for example (i) domain name system
(ii) voice concurrency system

→ There behaviour and state can be described by the distributed algorithm – A defination of steps to be taken by each of the process which the System is composed, including the transmission of messages b/w them.

→ messages are transmitted b/w process to transfere information b/w them and to coordinate their activity.

→ interacting process perform all the activity in a distributed system.

Note

(i) Communication performance is a limited feature.

(ii) It is impossible to maintain a single global notion of time.

Variants of interaction model

1) Synchronous distributed system
2) Asynchronous distributed system

Synchronous distributed system is defined as

- The time to execute each step of a process has known lower and upper bounds.
- Each message transmitted over a channel is received within a known bounded time.
- Each process has a local clock whose drift rate from real time has a known bound.

__/__/__

Asynchronous distributed system is defined as on the basis of –
(i) process execution speed
(ii) message transmission delay
(iii) clock drift rate

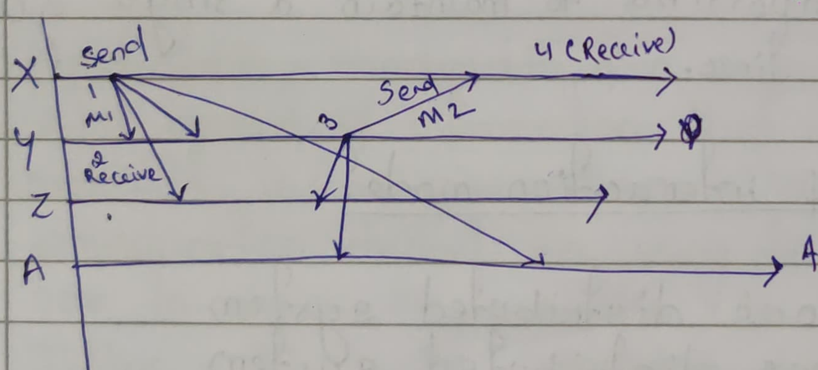
System model

Physical model
- → Baseline
- → Early distributed
- → Internet scale
- → contemporary.

Architectural

fundamental
- → Interaction
- → security
- → failure

Event ordering



Let at time t_1
X send msg (M_1) → Y
i.e.,
Y receives msg (M_1) ← X
and
Let at time t_2
Y send Msg(M_2) → X
A/c to real ordering of event
→ $\boxed{t_1 < t_2}$

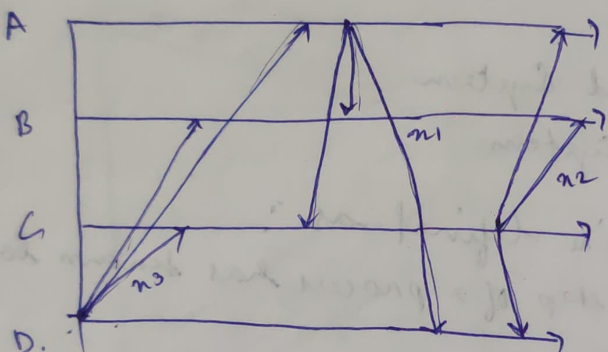
msg	sender	receiver
M_1	X	Y [t_1]
Re(M_1)	Y	X [t_1]

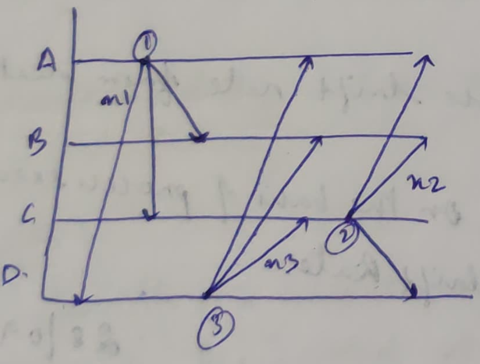
Msg	Sender	Receiver
m1	X	Y [t1]
Re(m1)	Y	X [t2]

Q.) On the basis of event ordering for given 4 senders A, B, C & D find the following relation. Events :-

1). A → C | C $\xrightarrow{x2}$ D | D $\xrightarrow{x3}$ B
t1: A → B | t2: C $\xrightarrow{x2}$ A | t3: D $\xrightarrow{x3}$ A
A → D | C $\xrightarrow{x2}$ B | D $\xrightarrow{x3}$ C.

$\boxed{T3 < T1 < T2}$





At t3 : D → B (x3)
At t2 : A → C (x1)
At t1 : C → D (x2)
i.e., t3 < t1 < t2

___/___/___

Failure model

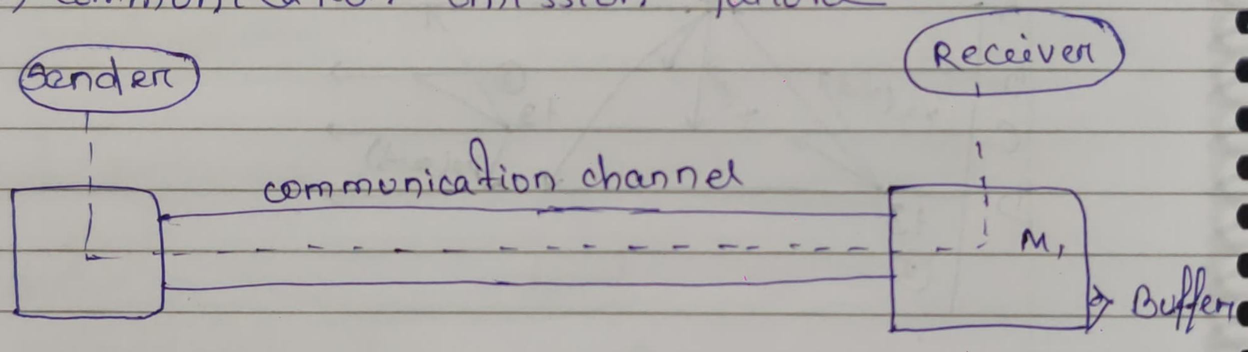
It defines the ways in which failure may occur in order to provide an understanding of the effects of the failure.

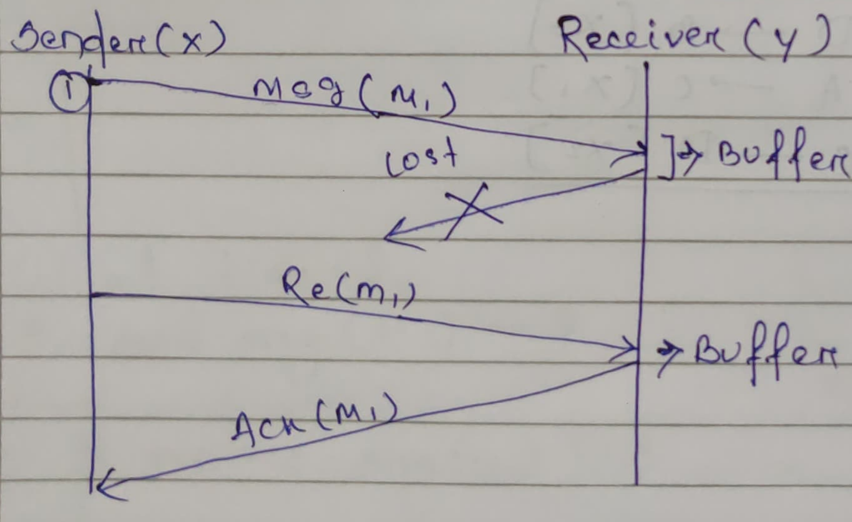
Types of failures—

I) Omission failures → Refer to cases when a process or communication channel fails to perform action that it is supposed to do.

(i) Process omission failure

(ii) Communication omission failure





A/c to diagram $M_1 : X \rightarrow Y$

But msg (M_1) lost

So, we send Re(M_1) : $X \rightarrow Y$

i.e., Y receives X and

X receives ack(M_1)

__/__/__

II) Arbitrary failures :-

An arbitrary failures of a process in which it arbitrary omits intended processing steps or takes unintended steps from these types of failures communication channel may suffer.

(a) fail-stop → process
(b) crash → process
(c) omission → channel
(d) send-omission → process
(e) Receive-omission → process

} Affected.

III) Timing failure :-

These are aplicable in synchronous distributed system where time limits are set on process execution time, message delivery time, clock draft rates.

Class of failures		Affects
clocks	→	process
performance of process	→	process
performance of message	→	channel

Reliability of one to one communication

validity → Any message in the outgoing message buffer is eventually delivered to incoming message buffer.

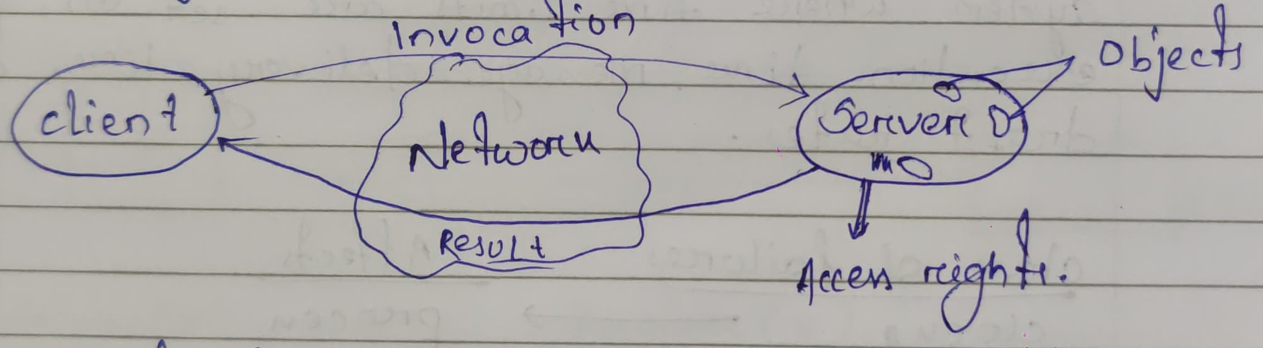
Integrity → The message received is identical to one sent, and no messages delivered twice.

__/__/__

The threats to integrity come from two independent sources:-

- Any protocol that retransmits messages but does not reject a message that arrives twices. Protocols can attach sequence numbers to messages so as to detect those that are delivered twice.
- Malicious users that may inject spurious messages, replay old messages or tamper with messages. security measures can be taken to maintain the integrity property in the face of such attacks.

III> Security model



Threats to process:-

a> servers
b> clients

Threats to channels:-

a> Attacks on secure channel
b> un authentication of a message