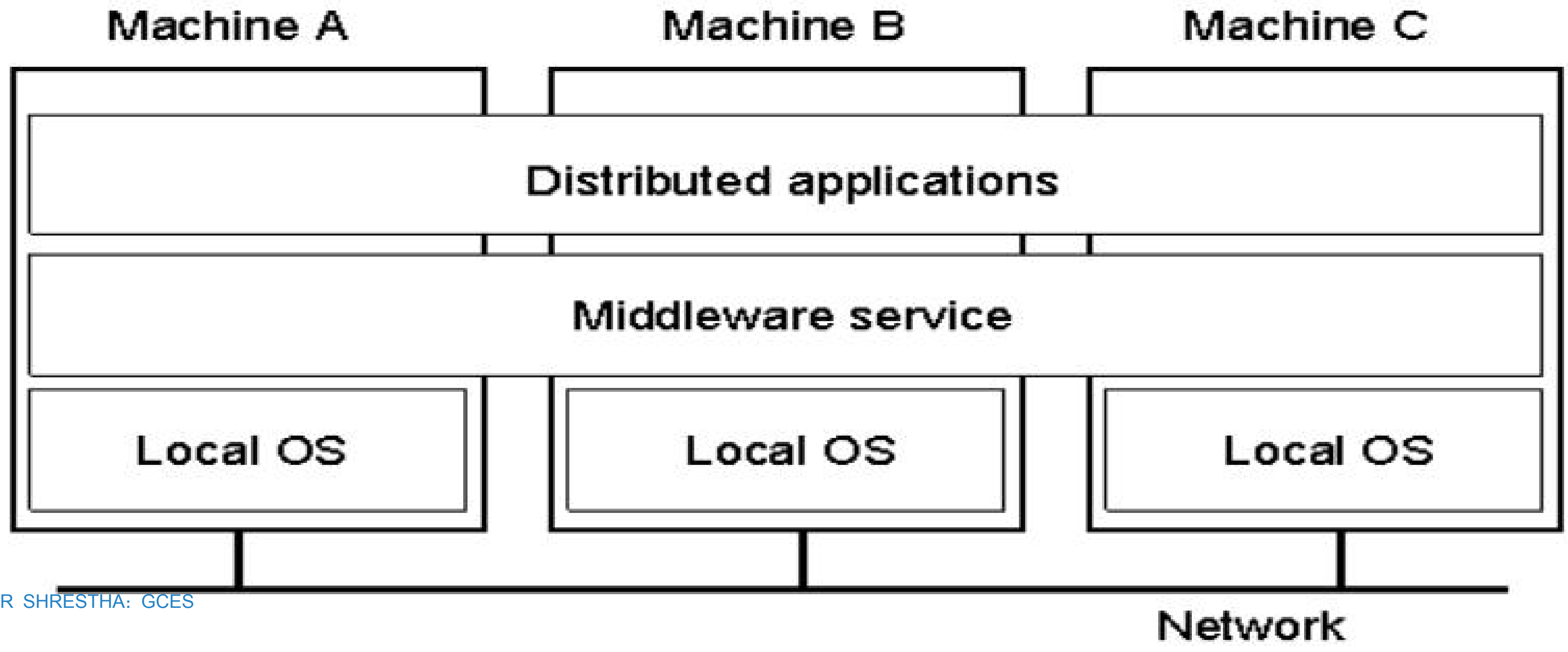
INTRODUCTION TO DISTRIBUTED SYSTEM

ER. DILIP KUMAR SHRESTHA

DISTRIBUTED SYSTEM



DEFINITIONS

A distributed system is a collection of independent computers that appears to its users as a single coherent system.

Aspects of definition

- Hardware: the machines are autonomous
- Software: the user think they are dealing with a single system

DEFINITIONS

Distributed system is a system in which hardware or software components located at networked computers communicate and coordinate their actions only by passing messages.

Key characteristics

- Concurrency: Several clients will attempt to access shared resources at the same time.
- No global clock: When programs need to cooperate, they coordinate their actions by exchanging messages.
- Independent failures: Each component of the system can fail independently, leaving the others still running.

KEY CHARACTERISTICS

- Resource sharing
- Flexibility
- Concurrency
- Reliability
- Fault Tolerance
- Speed

GOALS

1.Resource **sharing**

Transparency

Openness

Scalability

TRANSPARENCY

Access:

- Hide differences in data representation and how a resource is accessed

Location:

- Hide where a resource is located

Migration:

- Hide that a resource may move to another location

Relocation:

- Hide that a resource may be moved to another location while in use

Replication:

- Hide that a resource is replicated

Concurrency:

- Hide that a resource may be shared by several competitive users

Failure:

- Hide the failure and recovery of resource

SCALABILITY

By

- Size
- Geography
- Administrative

Problems

- Centralized services: A single server for all users
- Centralized data: A single on-line telephone book
- Centralized algorithms: Doing routing based on complete information

Scaling Techniques

- Hiding communication latencies
- Distribution
- Replication

DESIGN ISSUES

- Naming
- Communication
- Software Structure
- Openness
- Workload allocation
- Consistency maintenance

CONSISTENCY TYPES

Update Consistency:

consistency issues arise when several processes access and update the data concurrently. The modification of related set of data values is not possible instantaneously, but the changes by a given process should appear to all other process as though it was instantaneous.

Replication Consistency:

If data which is derived from a single source have been copied to several computers and subsequently modified at one or more of them, then there is a possibility of inconsistencies between the values of data items at different machines.

Cache Consistency:

It refers to the problem that arises when data values that have been cached by one client are updated by another.

Failure Consistency:

When a centralize computer system fails, all of the applications which are running on it fail simultaneously. But in case of distributed system, when one component fails, the others should continue to operate normally.

User interface consistency:

Whenever a user performs an input action such as key press or mouse click, in an interactive program, the screen becomes temporarily inconsistent while the processing of input goes on. This time delay should be acceptable.

Clock consistency:

Many of the algorithms used in applications and system software depend on the time stamps. In distributed system, clocks of the system can be synchronized by network communication

CHALLENGES

Heterogeneity

Openness

Security

Scalability

Failure handling

ADVANTAGES/DISADVANTAGES

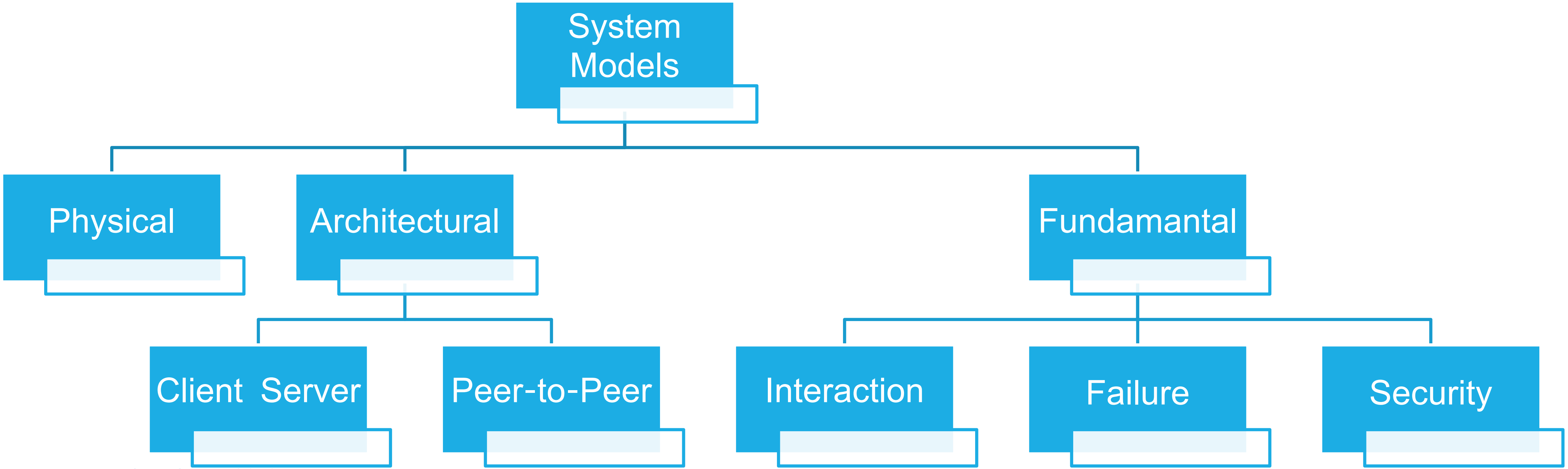
Advantages

- Sharing Data : There is a provision in the environment where user at one site may be able to access the data residing at other sites.
- Autonomy : Because of sharing data by means of data distribution each site is able to retain a degree of control over data that are stored locally.
- Availability : If one site fails in a distributed system, the remaining sites may be able to continue operating. Thus a failure of a site doesn't necessarily imply the shutdown of the System
- Give more performance than single system
- If one pc in distributed system malfunction or corrupts then other node or pc will take care of
- More resources can be added easily

Disadvantages

- Software Development Cost : It is more difficult to implement a distributed database system; thus it is more costly.
- Greater Potential for Bugs : Since the sites that constitute the distributed database system operate parallel, it is harder to ensure the correctness of algorithms, especially operation during failures of part of the system, and recovery from failures. The potential exists for extremely subtle bugs.
- Increased Processing Overhead : The exchange of information and additional computation required to achieve intersite co-ordination are a form of overhead that does not arise in centralized system.
- Security problem due to sharing
- Some messages can be lost in the network system
- Bandwidth is another problem if there is large data then all network wires to be replaced which tends to become expensive
- Overloading is another problem in distributed operating systems

DISTRIBUTED SYSTEM MODELS



System Models

Physical

Architectural

Fundamantal

Client Server

Peer-to-Peer

Interaction

Failure

Security

CONT…

Physical Models

- Physical models consider the types of computers and devices that constitute a system and their interconnectivity, without details of specific technologies.

Architectural Models

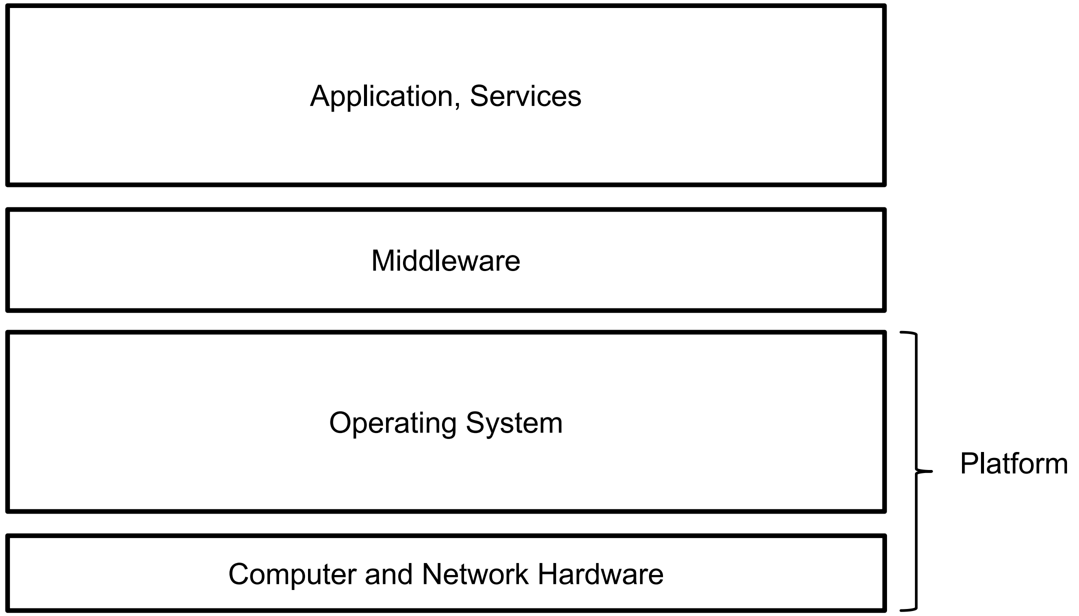
- Architectural models describe a system in terms of the computational and communication tasks performed by its computational elements; the computational elements being individual computers or aggregates of them supported by appropriate network interconnections.

ARCHITECTURAL MODEL

- **Software Layers**
- **Architectural Elements**
- **Communicating Entities**
- **Communication Paradigms**

SOFTWARE LAYERS

Architecture originally refers to structuring of software as layers or modules in a single computer and more recently in terms of services offered and requested between processes located in the same of different computers



ARCHITECTURAL ELEMENTS

What are the entities that are communicating in the distributed system?

How do they communicate, or, more specifically, what communication paradigm is used?

What (potentially changing) roles and responsibilities do they have in the overall architecture?

How are they mapped on to the physical distributed infrastructure (what is their placement)?

COMMUNICATING ENTITIES

- objects
- Components
- Web Services

COMMUNICATION PARADIGMS

Inter-process Communication

Remote Communication

Indirect Communication

- Group communication
- Publish-subscribe systems
- Message queues
- Tuple spaces
- Distributed shared memory

ARCHITECTURAL MODELS EXAMPLES

Distinct

- Client Server
- Peer-to-Peer

Variations

- Services provided by multiple servers
- Proxy servers and caches
- Mobile code
- Mobile agent
- Network computer
- Thin clients
- Mobile devices and spontaneous interoperation

FUNDAMENTAL MODEL

Interaction Model

- reflects the facts that communication takes place with delays that are often of considerable duration, and that the accuracy with which independent processes can be coordinated is limited by these delays and by the difficulty of maintaining the same notion of time across all the computers in a distributed system.

Failure Model

- defines the way in which failure may occur in order to provide an understanding of the effects of failure.

Security

- Defines the security methods to me adapted while designing the distributed system.

INTERACTION MODEL

Synchronous distributed system

- time to execute each step of a computation within a process has known lower and upper bounds
- message delivery times are bound to a known value
- each process has a clock whose drift rate from real time is bounded by a known value

Asynchronous distributed system

- no bound-on process execution times
- no bound-on message delivery times
- no bound-on clock drift rate

CONT...

Performance Characteristics of Communication Channels

- Latency
- Throughput
- Bandwidth
- Delay jitter

FAILURE MODEL

Omission Failures

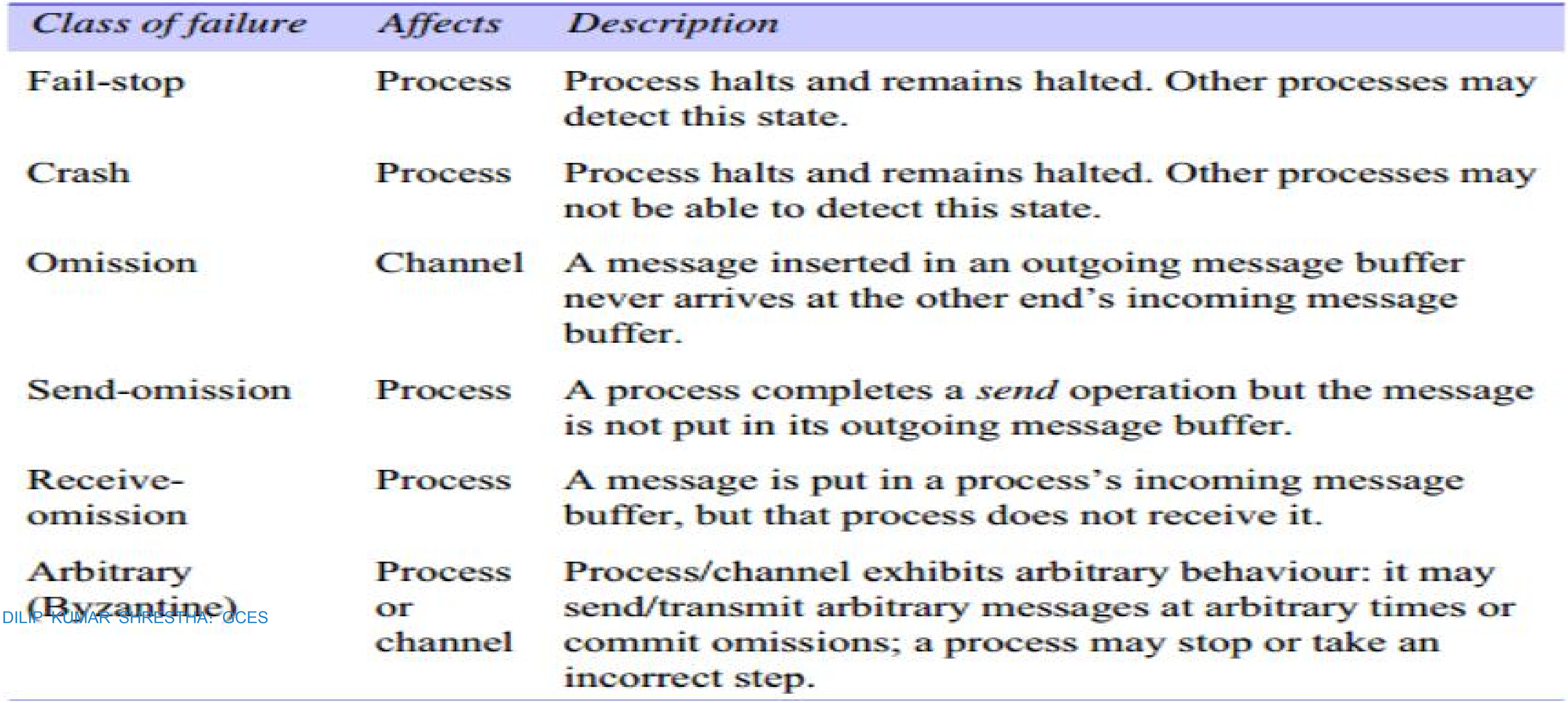
- process omission failures
- communication omission failures

Arbitrary Failures

- process: omit intended processing steps or carry out unintended ones
- communication channel: corruption or duplication etc.

Timing Failures

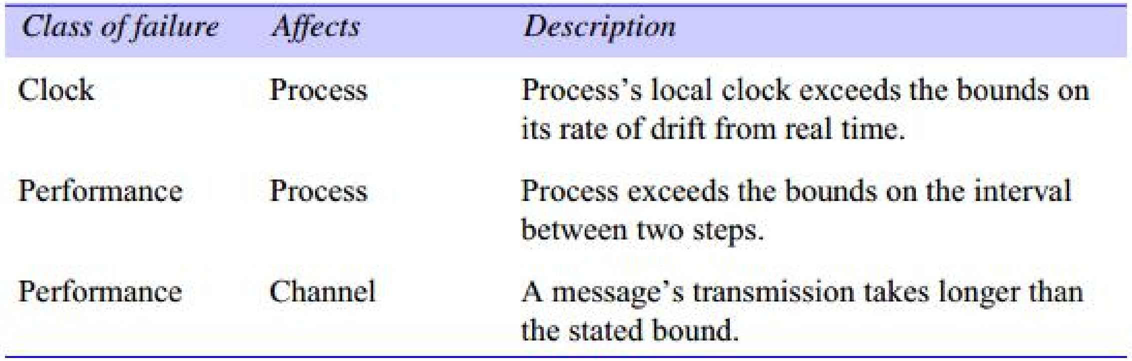
TYPES OF FAILURES



Class of failure	*Affects*	*Description*
Fail-stop	Process	Process halts and remains halted. Other processes may detect this state.
Crash	Process	Process halts and remains halted. Other processes may not be able to detect this state.
Omission	Channel	A message inserted in an outgoing message buffer never arrives at the other end's incoming message buffer.
Send-omission	Process	A process completes a *send* operation but the message is not put in its outgoing message buffer.
Receive-omission	Process	A message is put in a process's incoming message buffer, but that process does not receive it.
Arbitrary (Byzantine)	Process or channel	Process/channel exhibits arbitrary behaviour: it may send/transmit arbitrary messages at arbitrary times or commit omissions; a process may stop or take an incorrect step.

RELIABILITY OF ONE-TO-ONE COMMUNICATION

- ***Validity***: Any message in the outgoing message buffer is eventually delivered to the incoming message buffer.
- ***Integrity***: The message received is identical to one sent, and no messages are delivered twice



Class of failure	*Affects*	*Description*
Clock	Process	Process's local clock exceeds the bounds on its rate of drift from real time.
Performance	Process	Process exceeds the bounds on the interval between two steps.
Performance	Channel	A message's transmission takes longer than the stated bound.

SECURITY

problems

- Threats to processes - An attacker may send a request or response using a false identity (spoofing).
- Threats to communication channels - An attacker may eavesdrop (listen to messages) or inject new messages into a communication channel. An attacker can also save messages and replay them later.
- Denial of service - An attacker may overload a server by making excessive requests.

solutions

- Cryptography and authentication are often used to provide security. Communication entities can use a shared secret (key) to ensure that they are communicating with one another and to encrypt their messages so that they cannot be read by attackers.

THANK YOU

DILIP KUMAR SHRESTHA