

Lecture: 01

Introduction to Distributed Systems



Rajiv Misra

Dept. of Computer Science & Engineering
Indian Institute of Technology Patna
rajivm@iitp.ac.in

Preface

- The explosive growth of DCS makes understanding imperative yet difficult because of uncertainties introduced by asynchrony, limited local knowledge, and partial failures.
- The nature solves it perfectly such as flock of birds(mobile intelligent agents communicates to achieve common goal).
- The field of distributed computing provides theoretical underpinning for design and analysis of many DS such as- communication, coordination, synchronization and uncertainty to lower bound techniques.

Course Contents

- Introduction to Distributed Systems
- Basic Algorithms in Message Passing System
- Leader Election in Rings
- Distributed Minimum Spanning Tree
- Models of Distributed Computation
- Causality & Logical Time
- Global State and Snapshot Recording Algorithms
- Distributed Mutual Exclusion Algorithms
- Distributed Shared Memory
- Consensus and Agreement Algorithms
- Checkpointing & Rollback Recovery

Case Studies:

- Distributed Hash Table
- Peer to Peer Computing and Overlay Graphs
- Google File System
- HDFS and Map Reduce
- Introduction to Spark
- Introduction to Sensor Networks

Books

Text Books:

- Distributed Computing: Principles, Algorithms, and Systems-
Ajay D. Kshemkalyani and Mukesh Singhal
- Distributed Computing: Fundamentals, Simulations and Advanced Topics-
Hagit Attiya and Jennifer Welch

Reference Book:

- Distributed Algorithms-
Nancy Lynch

Distributed System: Definition

- A distributed system is
 - A collection of independent entities that cooperate to solve a problem that cannot be individually solved.
 - A collection of computers that do not share common memory or a common physical clock, that communicate by a messages passing over a communication network, and where each computer has its own memory and runs its own operating system.
 - Typically the computers are semi-autonomous and are loosely coupled while they cooperate to address a problem collectively.

Properties of Distributed Systems

- Heterogeneity
 - Systems consist of heterogeneous hardware and software components
- Concurrency
 - Multiple programs run together
- Shared data
 - Data is accessed simultaneously by multiple entities
- No global clock
 - Each component has a local notion of time
- Interdependencies
 - Independent components depend on each other

Relation to Computer System Components

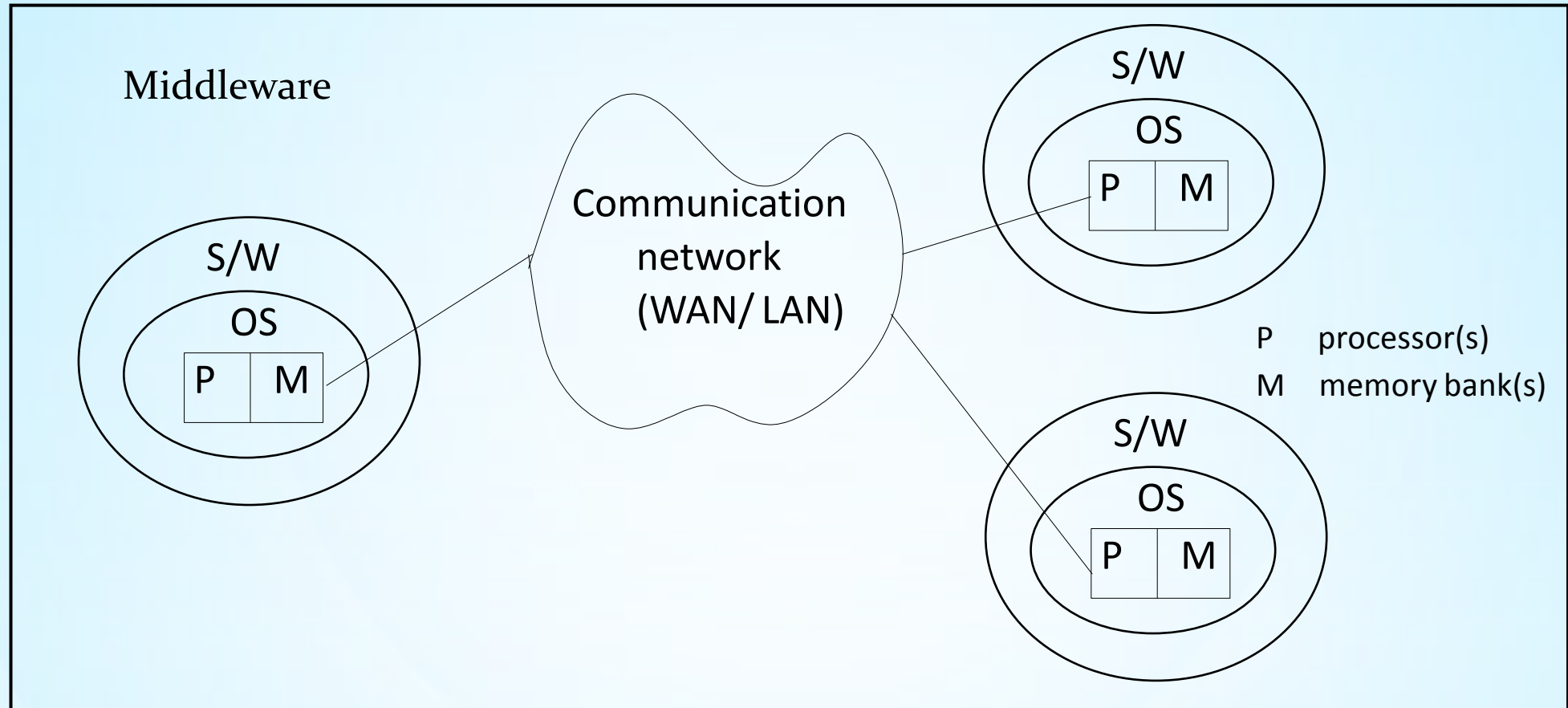


Figure 1.1: A typical distributed system

Relation to Computer System Components

- A DS connects autonomous processors by communication network which cooperate to run application network.
- The software components that run on each of the computers use the local operating system and network protocol stack for functioning.
- The distributed software is also termed as middleware.
- A distributed execution is the execution of processes across the distributed system to collaboratively achieve a common goal. An execution is also sometimes termed a computation or a run.

Layered Architecture

- The middleware(layered architecture) is the distributed software that drives the distributed system, while providing transparency of heterogeneity at the platform level.

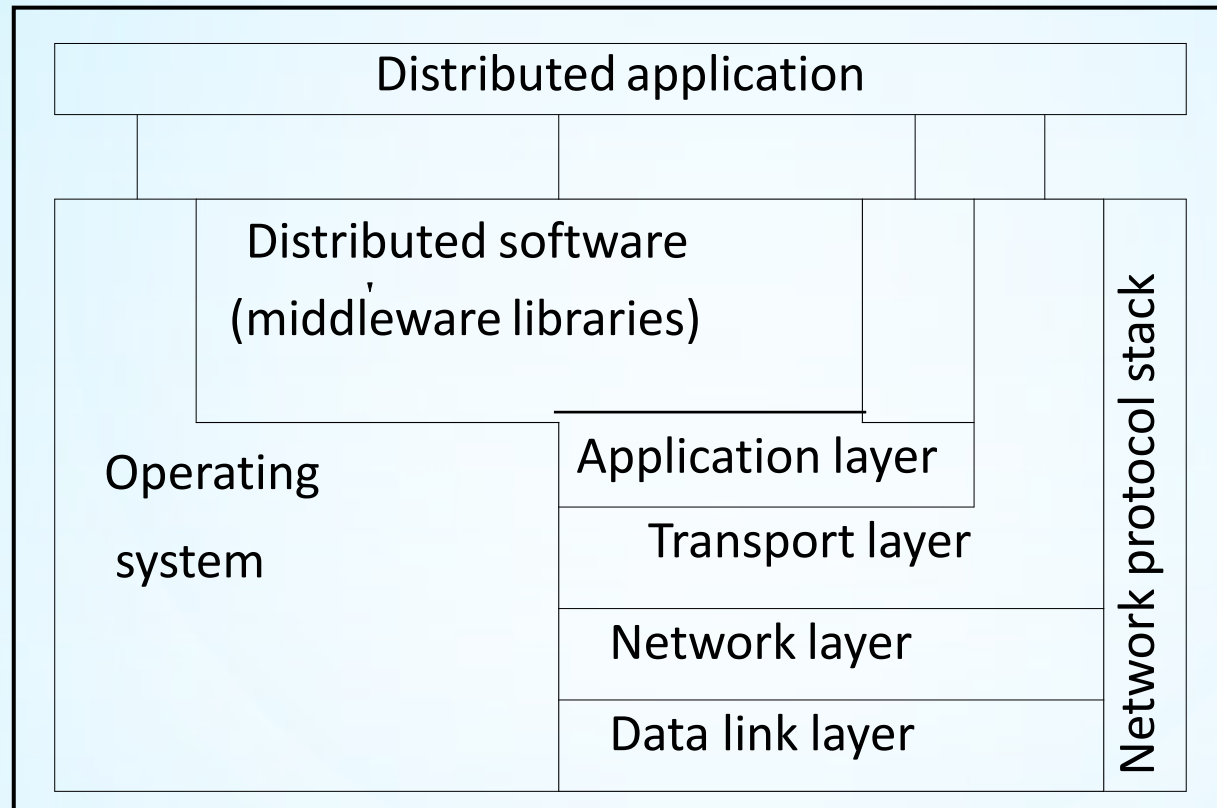


Figure 1.2: Layered Architecture of Distributed System Application

Several standards as :

- OMG (Object Management Group)
- CORBA (Common Object Request Broker Architecture)
- RPC (Remote Procedure Call)
- DCOM (Distributed Component Object Model)
- RMI (Remote Method Invocation)
- MPI (Message-Passing Interface)

Motivation for Distributed System

1. ***Inherently distributed computations:*** Many applications such as money transfer in banking, or reaching consensus among parties that are geographically distant, the computation is inherently distributed.
2. ***Resource sharing:*** *Sharing of* Resources such as peripherals, complete data sets in databases, special libraries, etc. It cannot be fully replicated at all the sites because it is often neither practical nor cost-effective.
3. ***Access to geographically remote data and resources:*** such as Bank databases, Supercomputers, resource-constrained mobile devices
4. ***Enhanced reliability:*** Possibility of replicating resources and executions to enhance reliability. The geographically distributed resources are not likely to crash/malfunction at the same time

Motivation for Distributed System

Reliability entails several aspects:

- **Availability**, i.e., the resource should be accessible at all times;
- **Integrity**, i.e., the value/state of the resource should be correct, in the face of concurrent access from multiple processors, as per the semantics expected by the application;
- **Fault-tolerance**, i.e., the ability to recover from system failures (defined as failure models)
- **Increased performance/cost ratio**: By accessing geographically remote data and resources sharing.

Other Advantages of Distributed System

- 5. ***Scalability***: Adding more processors to communication network does not pose a direct bottleneck to communication system.
- 6. ***Modularity and incremental expandability***: Heterogeneous processors may be easily added or replaced without affecting performance.

Design issues and challenges

1. Systems perspective of distributed system design
2. Algorithm perspective of distributed system design
3. Recent technology advances and/or Driven by new applications

1. Distributed systems design challenges from a system perspective

- **Communication:** This task involves designing appropriate mechanisms for communication among the processes in the network.
- **Processes:** Some of the issues involved are: management of processes and threads at clients/servers; code migration; and the design of software and mobile agents.
- **Synchronization:** Synchronization or coordination among the processes are essential. Mutual exclusion is the classical example of synchronization, but many other forms of synchronization, such as leader election, physical clocks, logical clocks and global state recording algorithms, all require different forms of synchronization.

1. Distributed systems challenges from a system perspective

- ***Fault tolerance:*** Requires maintaining correctness in spite of failures of links, nodes, and processes.
- Process resilience, reliable communication, distributed commit, check-pointing and recovery, agreement and consensus, failure detection, and self-stabilization are some of the mechanisms to provide fault-tolerance.

1. Distributed systems challenges from a system perspective

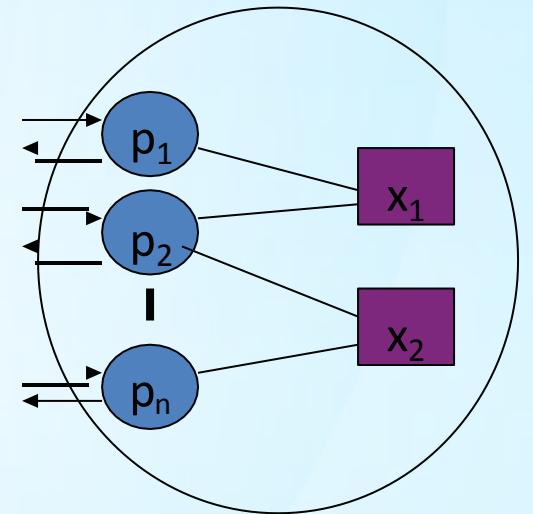
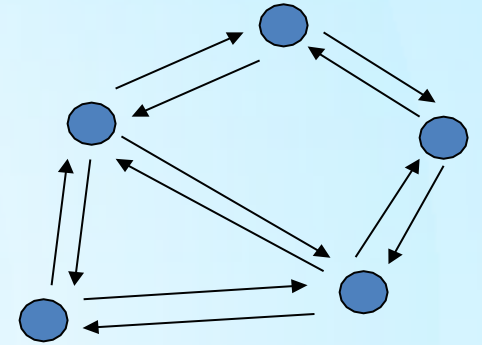
- **Transparency:** Hiding the implementation policies from the user can be classified as:
 1. Access transparency hides differences in data representation on different systems and provides uniform operations to access system resources.
 2. Location transparency makes the locations of resources transparent to the users.
 3. Migration transparency allows relocating resources without changing names.

Transparency contd..

4. Relocation transparency The ability to relocate the resources as they are being accessed is relocation transparency.
5. Replication transparency does not let the user become aware of any replication.
6. Concurrency transparency deals with masking the concurrent use of shared resources for the user.
7. Failure transparency refers to the system being reliable and fault-tolerant.

Distributed Algorithms

- In Distributed Systems, different complexity measures are of interest such as: time, space but now considered communication cost (no. of messages, size, no. of shared variables) and the number of faulty vs. non-faulty components.
- Because of complications faced by DS leads to increase scope of “negative results”, lower bounds and impossibility results.

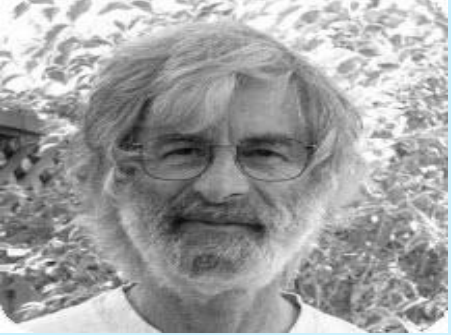


Distributed Algorithms

- Fundamental issues in design of distributed algorithms are factors such as : asynchrony, limited knowledge, and failures
- Asynchrony: Absolute & relative timings of events cannot always be known precisely.
- Local view: Computing entities can only be aware of information it acquires, so it has local views of global situation
- Failures: Computing entities can fail independently, leaving some components operational while others are not.

Distributed Computing Systems

- Studied since 1967, starting with Dijkstra and Lamport.
 - Edsger Dijkstra: 1972 Turing award winner
 - Leslie Lamport: 2013 Turing award winner



LESLIE LAMPORT

He devised important **algorithms** and developed formal modeling and verification protocols that improve the quality of real distributed systems.

Fundamental contributions to the theory and practice of distributed systems, notably the invention of concepts such as causality and logical clocks, safety and liveness, replicated state machines, and sequential consistency.

- Lamport was the winner of the 2013 **Turing Award** for **distributed computing** systems, in which several autonomous computers communicate with each other by passing messages.

2. Algorithmic challenges in distributed computing

1) *Time and global state in a distributed system:*

- The processes in the system are spread across three-dimensional physical space. Another dimension, time, has to be superimposed uniformly across space. The challenges pertain to providing accurate *physical time*, and to providing a variant of time, called *logical time*.
- *Logical time is relative time*, and eliminates the overheads of providing physical time for applications where physical time is not required. More importantly, logical time can (i) capture the logic and inter-process dependencies within the distributed program, and also (ii) track the relative progress at each process.

2. Algorithmic challenges in distributed computing

2) *Synchronization/coordination mechanisms:*

The processes must be allowed to execute concurrently, except when they need to synchronize to exchange information, i.e., communicate about shared data. Synchronization is essential for the distributed processes to overcome the limited observation of the system state. Following mechanisms are used for synchronization/coordination:

- Leader Election- Deals with asymmetry
- Mutual Exclusion- Access to critical resources has to be coordinated
- Termination Detection- Cooperation among processes to detect global state.
- Garbage Collection- Detecting garbage needs coordination

2. Algorithmic challenges in distributed computing

3) Reliable and fault-tolerant distributed systems:

A reliable and fault-tolerant environment has multiple requirements and aspects, and these can be addressed using various strategies:

1. Consensus algorithms
2. Replication and replica management
3. Voting and quorum systems
4. Distributed databases and distributed commit
5. Self-stabilizing systems
6. Check-pointing and recovery algorithms
7. Failure detectors

2. Algorithmic challenges in distributed computing

4) Group communication, multicast, and ordered message delivery:

A group is a collection of processes that share a common context and collaborate on a common task within an application domain. Specific algorithms need to be designed to enable efficient group communication and group management wherein processes can join and leave groups dynamically, or even fail. When multiple processes send messages concurrently, different recipients may receive the messages in different orders, possibly violating the semantics of the distributed program.

2. Algorithmic challenges in distributed computing

5) Distributed shared memory abstraction:

Under the covers in the middleware layer, the abstraction of a shared address space has to be implemented by using message-passing.

3. Applications of distributed computing and newer challenges

- 1) ***Mobile systems:*** Mobile systems typically use wireless communication which is based on electromagnetic waves and utilizes a shared broadcast medium.
- 2) ***Sensor networks:*** A sensor is a processor with an electro-mechanical interface that is capable of sensing physical parameters, such as temperature, velocity, pressure, humidity, and chemical
- 3) ***Ubiquitous or pervasive computing:*** Ubiquitous systems represent a class of computing where the processors embedded in and seamlessly pervading through the environment perform application functions in the background.

3. Applications of distributed computing and newer challenges

4) ***Peer-to-peer computing:*** Peer-to-peer (P2P) computing represents computing over an application layer network wherein all interactions among the processors are at a “peer” level, without any hierarchy among the processors. Thus, all processors are equal and play a symmetric role in the computation.

3. Applications of distributed computing and newer challenges

- 5) ***Distributed data mining:*** Data mining algorithms examine large amounts of data to detect patterns and trends in the data, to mine or extract useful information. A traditional example is: examining the purchasing patterns of customers in order to profile the customers and enhance the efficiency of directed marketing schemes.
- 6) ***Grid computing:*** Analogous to the electrical power distribution grid, it is envisaged that the information and computing grid will become a reality some day. Very simply stated, idle CPU cycles of machines connected to the network will be available to others.

3. Applications of distributed computing and newer challenges

7) *Security in distributed systems:* The traditional challenges of security in a distributed setting include: confidentiality (ensuring that only authorized processes can access certain information), authentication (ensuring the source of received information and the identity of the sending process), and availability (maintaining allowed access to services despite malicious actions).

Conclusion

- Distributed systems are having a wide variety of applications in real world scenario.
- To understand its contribution, it is required to be familiar with its fundamental principles.
- This lecture first characterized distributed systems and distributed algorithms by looking at various informal definitions, design issues and challenges based on theoretical & systems aspects.
- In upcoming lectures, we will try to give an insight on its detailed concepts that will give a good understanding of the further details.