Devi Ahilya University, Indore, India Institute of Engineering & Technology				III Year B.E. (Computer Engg.) (Full Time)			
Subject Code & Name	**Instructions Hours per Week**			**Credits**			
6CERC3 DISTRIBUTED SYSTEMS	**L**	**T**	**P**	**L**	**T**	**P**	**Total**
	3	**1**		**3**	**1**		**4**
Duration of Theory Paper: 3 Hours							

Prerequisite:

Operating Systems, Computer Networks, Data Structures & Algorithms, and proficiency in a programming language (Java/Python/C++).

Learning Objectives

Students will learn to:

- Understand distributed system architecture and communication models.
- Learn synchronization, consistency, and coordination techniques.
- Study distributed file systems, DSM, and fault tolerance.
- Understand modern frameworks like Hadoop, Spark, cloud & edge computing.
- Learn security mechanisms and design basic distributed applications.

Course Objectives –

- To provide foundational knowledge of distributed computing models and architectures.
- To understand communication, synchronization, and coordination in distributed environments.
- To study distributed file systems, shared memory, and fault tolerance techniques.
- To learn modern distributed platforms (Hadoop, Spark, cloud/edge computing).
- To develop the ability to design and implement distributed applications securely and efficiently.

Course Outcome:

CO No.	Course Outcome (CO)	Mapped PO(s)
CO1	Understand the fundamental concepts, architectures, and communication models of distributed systems.	**PO1, PO2**
CO2	Apply synchronization, coordination, and consistency techniques such as logical clocks, mutual exclusion, and replication.	**PO1, PO2, PO3**

CO No.	Course Outcome (CO)	Mapped PO(s)
CO3	Analyse the design and working of distributed file systems (NFS, AFS, HDFS) and distributed shared memory systems.	**PO2, PO3**
CO4	Evaluate distributed algorithms for consensus, fault tolerance, and reliability (2PC, 3PC, snapshots, and checkpoints).	**PO2, PO4**
CO5	Use modern distributed frameworks such as Hadoop, Map Reduce, Spark, and apply distributed programming concepts to solve real problems.	**PO3, PO5**
CO6	Identify security threats in distributed environments and recommend appropriate security mechanisms such as authentication and encryption.	**PO2, PO12**
CO7	Design and implement simple distributed applications demonstrating communication, synchronization, and basic fault tolerance.	**PO3, PO4, PO5**

CO-PO Relationship

Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6–PO11	PO12
CO1: Understand the fundamental concepts, architectures & communication models of distributed systems.	**3**	**2**	1	–	–	–	1
CO2: Apply synchronization, coordination, and consistency mechanisms.	**3**	**3**	**2**	2	1	–	–
CO3: Analyse DFS, DSM and distributed storage mechanisms.	2	**3**	**3**	2	1	–	–
CO4: Evaluate fault tolerance, consensus, and recovery algorithms.	2	**3**	2	**3**	1	–	–
CO5: Utilize modern distributed frameworks (Hadoop, Spark, MapReduce).	1	2	**3**	2	**3**	–	–
CO6: Identify and address security issues in distributed environments.	1	2	1	2	2	–	**3**
CO7: Design and implement distributed applications.	2	2	**3**	**3**	**3**	–	–

* CO (rows) mention nil/very small/insignificant contribution to the PO (column)
1→ relevant and small significance 2 → medium or moderate and 3 →strong

UNIT - I

1.1 Fundamentals

- Definition, characteristics, goals of distributed systems
- Types of distributed systems: Client-Server, Peer-to-Peer, Cluster, Cloud

1.2 Models of Distributed Systems

- Architectural models
- Fundamental models (interaction, failure, security models)

1.3 Communication Basics

- Inter-process communication (message passing, RPC, RMI)
- Sockets, communication channels, marshalling

1.4 Case Studies

- Google File System (GFS)
- Hadoop Distributed File System (HDFS)

UNIT – II

Synchronization, Coordination & Consistency

2.1 Synchronization

- Clock synchronization: NTP, Logical clocks (Lamport, Vector clocks)
- Global states, snapshots (Chandy–Lamport algorithm)

2.2 Coordination

- Mutual exclusion algorithms (Lamport, Ricart-Agrawala)
- Election algorithms (Bully, Ring)

2.3 Consistency & Replication

- Data-centric consistency models (strict, sequential, causal, eventual consistency)
- Client-centric consistency
- Replication techniques

UNIT – III

Distributed File Systems & Distributed Shared Memory

3.1 Distributed File Systems (DFS)

- Naming concepts
- File caching, replication
- Directory services

3.2 DFS Case Studies

- NFS (Network File System)
- AFS (Andrew File System)

3.3 Distributed Shared Memory (DSM)

- Concepts and architecture
- Memory coherence & consistency in DSM
- Implementation issues

3.4 Fault Tolerance Basics

- Fail-stop, crash recovery models
- Logging and check pointing

UNIT - IV

Distributed Algorithms & Fault Tolerance

4.1 Distributed Algorithms

- Consensus & agreement problems
- Two-phase commit (2PC)
- Three-phase commit (3PC)

4.2 Fault Tolerance

- Redundancy, replication
- Byzantine fault tolerance (simplified)
- Leader election, reliable group communication

4.3 Distributed Object Models

- CORBA basics

- Communication and invocation in distributed objects

UNIT- V

Cloud Computing, Edge Computing & Modern Distributed Systems

5.1 Cloud Computing Concepts

- Virtualization: types and role in distributed systems
- IaaS, PaaS, SaaS models
- Cloud storage and distributed databases

5.2 Edge & Fog Computing

- Architecture and use cases
- Low-latency distributed systems

5.3 Modern Distributed Frameworks

- Map Reduce Programming Model
- Apache Hadoop and Spark architecture

5.4 Security in Distributed Systems

- Threats and challenges
- Cryptographic basics, secure communication
- Access control

Learning Outcomes:

Upon completion, students will be able to:

- Understand distributed systems architecture and communication.
- Apply synchronization, consistency, and coordination mechanisms.
- Analyse distributed algorithms and fault tolerance.
- Use modern distributed frameworks (Hadoop/Spark).
- Address security challenges in distributed environments.
- Design and develop basic distributed applications.

Books

- **Distributed Systems: Principles and Paradigms** by Andrew S. Tanenbaum & Maarten Van Steen — Frequently recommended for academic courses. pce.ac.in+1
- **Distributed Computing: Principles, Algorithms, and Systems** by Ajay D. Kshemkalyani & Mukesh Singhal — Good for algorithms, concurrency and practical distributed system behaviour. Google Books+1
- For more practice and modern aspects: **Building Distributed Systems** by Ranjit Aneesh — emphasizes real-world design decisions and architecture (useful if you plan to work on projects or distributed systems in industry). BPB Online