Department of CSE
Details of Lesson Plan

S.No.	Particulars	Details
1.	Course Name	Distributed Systems
2.	Course Code	MCSE-104
3.	Academic Year	2024-2025
4.	Semester	1st
5.	Number of Lesson plans	38
6.	Faculty Assigned	Parul Sharma

Lesson Plan No. 1.1	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Introduction to Distributed Systems**
Objectives	At the end of the lesson, the student shall be able to: a. Explain the definition of distributed systems and how they relate to computer system components. b. Discuss the motivation behind using distributed systems and their advantages over centralized systems.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Distributed Systems (5 minutes)** Start by explaining the definition of distributed systems and their essential role in modern computing. Highlight the relationship between distributed systems and the core components of a computer system. • **Motivation and Benefits (10 minutes)** Discuss the reasons for using distributed systems, including scalability, fault tolerance, and resource sharing. Illustrate how these systems meet the demands of large-scale, high-availability applications. • **Comparison with Parallel Systems (10 minutes)** Explain the differences and relationships between distributed systems and parallel systems. Focus on how tasks are divided and managed in both systems and the implications for performance and complexity. • **Introduction to Message-Passing vs. Shared Memory Systems (10 minutes)** Delve into the two primary models of interaction in distributed systems: message-passing systems and shared memory systems. Describe each model's characteristics, usage scenarios, and how they manage data consistency and communication. • **Discussion and Wrap-Up (5 minutes)** End with a discussion that encourages students to think about scenarios where one might prefer distributed systems over centralized ones. Allow time for questions and provide a summary of the key points discussed.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Write a brief essay explaining the definition of distributed systems and discussing how these systems integrate with and differ from traditional computer system components. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.2	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Comparing Message-Passing and Shared Memory Systems**
Objectives	At the end of the lesson, the student shall be able to: a. Analyze the differences between message-passing systems and shared memory systems. b. Evaluate the pros and cons of each system type in terms of performance, scalability, and applicability.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Communication Models (5 minutes)** Begin the lecture by defining message-passing and shared memory systems as the two primary communication models in distributed systems. Briefly explain how each model facilitates inter-process communication. • **Detailed Analysis of Message-Passing Systems (10 minutes)** Discuss the architecture and operational mechanics of message-passing systems. Explain how processes communicate by explicitly sending and receiving messages, using real-world examples like MPI (Message Passing Interface). • **Exploration of Shared Memory Systems (10 minutes)** Shift focus to shared memory systems where multiple processes access a common memory space. Describe scenarios such as multithreaded programs on multicore processors where shared memory provides efficiency gains. • **Performance, Scalability, and Applicability (10 minutes)** Compare and contrast the performance implications of each model. Discuss scalability challenges and the typical applicability of each model in various computing environments, highlighting the benefits and limitations related to system design and maintenance. • **Q&A and Summary (5 minutes)** Conclude with a question-and-answer session to address any uncertainties or deeper inquiries from the students. Summarize the key distinctions and practical considerations when choosing between message-passing and shared memory systems.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Create a comparison chart that lists the differences between message-passing systems and shared memory systems, including the pros and cons of each in terms of performance, scalability, and applicability. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.3	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Primitives for Distributed Communication**
Objectives	At the end of the lesson, the student shall be able to: a. Describe the basic communication primitives used in distributed systems. b. Explore how these primitives enable synchronous and asynchronous executions in distributed environments.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Communication Primitives (5 minutes)** Start by defining communication primitives and their significance in the foundation of distributed systems. Provide an overview of the types of primitives commonly used, such as send, receive, and broadcast. • **Exploring Basic Primitives (10 minutes)** Delve into the basic communication primitives like sending and receiving messages. Explain their roles in facilitating communication between distributed processes, using examples to illustrate their implementation. • **Synchronous vs. Asynchronous Executions (10 minutes)** Discuss the concepts of synchronous and asynchronous communication. Describe how synchronous communication requires a handshake between sender and receiver, whereas asynchronous communication allows processes to continue without waiting for message acknowledgments. • **Impact of Communication Primitives on Distributed Systems (10 minutes)** Analyze how the choice of communication primitives affects the design and performance of distributed systems. Consider factors like latency, throughput, and fault tolerance in different communication models. • **Discussion and Practical Application Scenarios (5 minutes)** End the session with a discussion encouraging students to share their thoughts on scenarios where certain primitives might be more effective. Provide a brief recap of the key points discussed and suggest further reading or projects to deepen understanding.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Draft a report describing various communication primitives used in distributed systems and illustrate with examples how these primitives facilitate both synchronous and asynchronous executions. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.4	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Design Issues and Challenges in Distributed Systems**
Objectives	At the end of the lesson, the student shall be able to: • Identify key design issues when developing distributed systems. • Discuss the challenges faced by developers, such as fault tolerance, consistency, and scalability.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Design Issues in Distributed Systems (5 minutes)** Begin by discussing the complexity of designing distributed systems. Outline the session's focus on key design issues that arise from the distributed nature of these systems. • **Key Design Issues (10 minutes)** Delve into the key design issues developers face when building distributed systems. These include data distribution strategies, service orchestration, and ensuring effective load distribution across nodes. • **Fault Tolerance (10 minutes)** Explain the concept of fault tolerance and its critical role in distributed systems. Discuss various strategies to achieve fault tolerance, such as redundancy, replication, and failover mechanisms, providing examples of how these are implemented in industry-standard solutions. • **Consistency and Scalability Challenges (10 minutes)** Address the challenges of maintaining consistency in data across multiple nodes while managing the scalability of the system. Discuss consistency models such as eventual consistency and strong consistency, and how they impact system design. • **Discussion and Practical Challenges (5 minutes)** Encourage an interactive discussion where students can share their insights or ask questions about the practical challenges faced in real-world distributed systems. Discuss how theoretical approaches are adapted in practical applications to handle these challenges. •

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Prepare a detailed analysis on the key design issues and challenges (such as fault tolerance, consistency, and scalability) when developing distributed systems, and suggest potential solutions or strategies to overcome these challenges. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.5	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Models of Distributed Executions and Programs**
Objectives	At the end of the lesson, the student shall be able to: a. Introduce a model of distributed executions that describes how distributed programs operate. b. Explain different models of communication networks and their impact on the behavior and performance of distributed applications.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Models of Distributed Executions (5 minutes)** Start the lecture by introducing the concept of distributed executions. Define what is meant by a "model" in this context and why it is crucial for understanding distributed programs. • **A Model of Distributed Programs (10 minutes)** Explain how distributed programs are modeled, focusing on aspects like process coordination, task distribution, and resource management. Use diagrams to illustrate how different components of a distributed program interact. • **Models of Communication Networks (10 minutes)** Introduce various models of communication networks such as client-server, peer-to-peer, and publish-subscribe models. Discuss how these models influence the design and functionality of distributed systems. • **Impact on Behavior and Performance (10 minutes)** Explore the impact of different execution and communication models on the behavior and performance of distributed applications. Discuss how choices in model design affect scalability, latency, and throughput. • **Q&A and Recap of Key Concepts (5 minutes)** Finish the session with a Q&A period, allowing students to clarify doubts and deepen their understanding of the material covered. Recap the key models discussed and their implications for the design and operation of distributed systems.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. **Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Research and describe a model of distributed executions and explain how it influences the operation of distributed programs. Include an analysis of different models of communication networks and their impact on system behavior and performance. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.6	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Understanding Global States and Cuts in Distributed Systems**
Objectives	At the end of the lesson, the student shall be able to: a. Define what global state and cuts are in the context of distributed systems. b. Analyze how past and future cones can be used to understand the dependencies and causality of events.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Global State and Cuts (5 minutes)** Begin by defining what global state is in the context of distributed systems. Explain the concept of cuts as a snapshot of the system state at a given point in time across multiple processes. • **Detailed Exploration of Global States (10 minutes)** Discuss how global states are utilized in distributed systems to monitor and debug system performance. Use examples to show how capturing global states can help in resolving issues like deadlocks and resource contention. • **Understanding Cuts and Their Types (10 minutes)** Elaborate on different types of cuts, such as consistent and inconsistent cuts, and their significance in distributed computing. Provide visual aids to help illustrate how cuts are made across process timelines. • **Past and Future Cones of an Event (10 minutes)** Explain the concepts of past and future cones in distributed systems. Discuss how these cones help understand the dependencies and causality of events, aiding in system diagnosis and recovery strategies. • **Discussion and Interactive Analysis (5 minutes)** Engage students in an interactive discussion where they can analyze given scenarios involving global states and cuts. Encourage them to identify potential issues that might arise from different types of cuts. •
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:**

	• Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Submit a written explanation of what global states and cuts represent in distributed systems, and analyze how understanding past and future cones of an event can aid in managing dependencies and event causality. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.7	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Logical Time and Clock Synchronization**
Objectives	At the end of the lesson, the student shall be able to: a. Introduce the concept of logical clocks and explain how they help manage time in distributed systems. b. Discuss physical clock synchronization techniques, with a focus on the Network Time Protocol (NTP).
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Logical Clocks (5 minutes)** Start the session by introducing the concept of logical clocks and their importance in managing time across distributed systems where physical clock synchronization is not feasible. • **Framework for Logical Clocks (10 minutes)** Discuss the framework for logical clocks, including how they are implemented and used in distributed systems to order events and ensure consistency. • **Scalar and Vector Clocks (10 minutes)** Introduce scalar time and extend the discussion to vector clocks. Explain how vector clocks allow a system to more accurately track the causal relationships between events across different processes. • **Physical Clock Synchronization (10 minutes)** Shift focus to physical clock synchronization, particularly the Network Time Protocol (NTP). Describe how NTP works to synchronize clocks in a distributed environment and why precise time synchronization is crucial for certain distributed applications. • **Q&A and Practical Implications (5 minutes)** Conclude with a Q&A session, allowing students to ask detailed questions about logical and physical time management in distributed systems. Recap the key points and discuss practical implications of time management strategies in real-world systems. •
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these.

	2. Suggested Reading books: • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Write a summary on the concept of logical clocks and their application in distributed systems. Additionally, discuss how physical clock synchronization, particularly the Network Time Protocol (NTP), is implemented and its importance. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.8	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Vector Clocks and Event Ordering**
Objectives	At the end of the lesson, the student shall be able to: a. Explain the concept of vector clocks and how they extend scalar logical clocks to capture causality between distributed events. b. Explore how vector clocks are used to order events across different processes in a distributed system.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Vector Clocks (5 minutes)** Begin by explaining the concept of vector clocks. Describe how they build upon scalar logical clocks to provide a comprehensive method for tracking the sequence and causality of events in distributed systems. • **How Vector Clocks Work (10 minutes)** Dive into the mechanics of vector clocks, explaining how each component of the system maintains its own clock that is updated upon sending or receiving messages. • **Using Vector Clocks to Order Events (10 minutes)** Discuss the use of vector clocks in ordering events across different processes. Provide examples that illustrate how vector clocks resolve ambiguities in event sequences that are not possible with scalar clocks. • **Challenges and Limitations of Vector Clocks (10 minutes)** Address the challenges and computational costs associated with maintaining vector clocks, especially in systems with a large number of processes. Discuss strategies to mitigate these issues. • **Interactive Session and Conclusion (5 minutes)** Engage students in an interactive session where they can suggest scenarios and discuss how vector clocks would handle the ordering of events. Summarize the key benefits of vector clocks in maintaining distributed system consistency and conclude the session.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these.

	2. **Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Compose an essay on vector clocks, detailing how they extend scalar logical clocks to better capture causality between distributed events. Discuss how vector clocks are used to order events across different processes in a distributed system. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.1	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Message Ordering Paradigms**
Objectives	At the end of the lesson, the student shall be able to: a. Explain the different paradigms of message ordering such as FIFO, causal, and total ordering. b. Discuss the importance of each ordering paradigm in ensuring consistency in distributed systems.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Message Ordering Paradigms (5 minutes)** Start by defining message ordering and its critical importance in distributed systems. Explain the concept of ordering paradigms such as FIFO (First In, First Out), causal, and total ordering. • **Explaining Each Ordering Paradigm (15 minutes)** Delve into each type of message ordering paradigm, explaining how they work, their applications, and their importance in ensuring consistency across distributed systems. Use examples to illustrate the differences and how they affect system operations. • **Applications and Implications (15 minutes)** Discuss real-world applications of different message ordering paradigms. Evaluate the implications of selecting one ordering system over another in terms of system performance, complexity, and reliability. • **Discussion and Conclusion (5 minutes)** Encourage an open discussion about the challenges of implementing these paradigms in modern distributed systems. Wrap up by summarizing the key points and takeaways from the session.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994

	- Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: - Write a comparative analysis of FIFO, causal, and total ordering paradigms. Include specific examples where each paradigm would be most effective in maintaining consistency in distributed systems. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.2	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Asynchronous Execution with Synchronous Communication**
Objectives	At the end of the lesson, the student shall be able to: a. Describe how asynchronous systems can implement synchronous communication methods. b. Analyze the challenges and benefits of combining these two approaches in distributed environments.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Asynchronous Systems and Synchronous Communication (5 minutes)** Explain the fundamental differences between asynchronous execution and synchronous communication. Discuss why integrating synchronous communication in an asynchronous environment might be necessary. • **Mechanics of Integrating Synchronous Communication (15 minutes)** Detail the techniques and technologies used to implement synchronous communication in asynchronous systems. Discuss scenarios where this integration is beneficial and necessary. • **Challenges and Benefits (15 minutes)** Analyze the challenges such as potential performance bottlenecks and benefits like improved data consistency that come with integrating synchronous communication in asynchronous systems. • **Q&A and Practical Insights (5 minutes)** Open the floor for questions and provide practical insights based on current technologies and trends. Conclude with a brief recap of the session.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994

	- Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: - Develop a brief report on how asynchronous systems can benefit from implementing synchronous communication methods. Discuss the potential challenges and advantages of this integration, citing examples from case studies or theoretical models. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.3	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Synchronous Program Order on an Asynchronous System**
Objectives	At the end of the lesson, the student shall be able to: a. Explore how synchronous program orders can be enforced in inherently asynchronous distributed systems. b. Evaluate the impact of enforcing synchronous order on system performance and reliability.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Synchronous Order in Asynchronous Systems (5 minutes)** Begin with an explanation of what synchronous program order means and its relevance in an asynchronous distributed environment. • **Enforcing Synchronous Order (15 minutes)** Discuss methods and protocols used to enforce a synchronous order, such as barrier synchronizations and clock synchronization techniques. • **Impact on System Performance (15 minutes)** Evaluate how enforcing synchronous order affects performance, scalability, and fault tolerance of an asynchronous system. • **Wrap-Up and Future Trends (5 minutes)** Conclude with a discussion on future trends in this area and open up for questions, summarizing the key points discussed in the lecture.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**:

	• Investigate a case study or academic paper where synchronous program order was enforced in an asynchronous system. Summarize the findings, focusing on the impact on system performance and reliability, and provide your critical analysis of the approach. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.4	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Group Communication in Distributed Systems**
Objectives	At the end of the lesson, the student shall be able to: a. Introduce the concept of group communication and its mechanisms in maintaining consistency and managing group dynamics. b. Discuss the implementation of group communication protocols and their role in fault tolerance and message dissemination.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Basics of Group Communication (5 minutes)** Define group communication and its fundamental importance in distributed systems. Introduce concepts like multicast and group consensus. • **Mechanisms and Protocols (15 minutes)** Discuss various mechanisms and protocols that facilitate group communication, such as reliable multicast and consensus algorithms. Include practical examples to illustrate these mechanisms. • **Challenges and Strategies (15 minutes)** Explore challenges in group communication such as network partitions and message loss. Discuss strategies to overcome these challenges and ensure robust group communication. • **Interactive Session (5 minutes)** Engage students in an interactive session where they can suggest scenarios or ask questions about group communication. Summarize the session with key takeaways.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**:

	• Create a presentation outlining different mechanisms of group communication in distributed systems. Discuss how these mechanisms aid in maintaining consistency, managing group dynamics, and ensuring fault tolerance. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.5	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Causal and Total Ordering of Messages**
Objectives	At the end of the lesson, the student shall be able to: a. Explain causal ordering and its requirement for preserving the causality of events in distributed systems. b. Define total order and discuss how it can be achieved in distributed environments, including the algorithms used to ensure total ordering.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Understanding Causal and Total Ordering (5 minutes)** Introduce the concepts of causal and total ordering. Explain how these ordering mechanisms are crucial for maintaining consistency across distributed applications. • **Implementing Causal Ordering (15 minutes)** Detail the implementation of causal ordering with practical examples. Discuss vector clocks and their role in maintaining causal relationships between events. • **Achieving Total Order (15 minutes)** Explain the methods to achieve total ordering in distributed systems, such as sequencers or consensus protocols. Discuss the trade-offs involved in implementing total order. • **Discussion and Application Scenarios (5 minutes)** Provide a platform for students to discuss application scenarios and potential challenges with causal and total ordering. Conclude by recapping the major points.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994

	- Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: - Design a simple scenario or use a diagram to illustrate how causal and total ordering of messages is implemented in a distributed system. Discuss the scenarios where each ordering is crucial and how it helps preserve the causality of events. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.6	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Global State and Snapshot Recording Algorithms**
Objectives	At the end of the lesson, the student shall be able to: a. Introduce the concepts of global state and the significance of snapshot recording in distributed systems. b. Describe various snapshot algorithms for FIFO channels and their applications in capturing consistent global states without halting the entire system.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	• **Introduction to Global State and Snapshots (5 minutes)** Define global state and the importance of snapshot recording in distributed systems. Explain how snapshots help in fault recovery and system diagnosis. • **Snapshot Algorithms for FIFO Channels (15 minutes)** Describe snapshot algorithms specifically designed for FIFO channels, such as the Chandy-Lamport algorithm. Provide step-by-step explanations of how these algorithms capture global states. • **Applications and Challenges (15 minutes)** Discuss the applications of snapshot algorithms in real-world systems and the challenges faced during their implementation. • **Q&A and Closing Thoughts (5 minutes)** End with a question-and-answer session, allowing students to explore further the concepts discussed. Summarize the session and highlight key areas for further study.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994

	- Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: - Conduct a research on various snapshot algorithms used for FIFO channels in distributed systems. Write a summary of each algorithm, focusing on how they capture global states and the implications of their use in real-world systems. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.1	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Introduction to Distributed Mutual Exclusion**
Objectives	At the end of the lesson, the student shall be able to: a. Understand the concept of mutual exclusion in distributed systems and its importance in ensuring consistent data. b. Explore various scenarios and challenges that necessitate the use of distributed mutual exclusion algorithms.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Distributed Mutual Exclusion (5 minutes)** • Start with a brief explanation of mutual exclusion and its role in distributed systems to ensure data consistency. **Scenarios Necessitating Mutual Exclusion (10 minutes)** • Discuss real-world scenarios such as online banking and airline reservation systems where distributed mutual exclusion is critical. • Explain the challenges faced in these scenarios, such as network delays, partitioning, and concurrent access. **Transition to Preliminaries of Distributed Systems (10 minutes)** • Shift focus to the structure underlying systems that manage mutual exclusion. • Introduce the basic components and architecture of distributed systems, emphasizing their role in supporting mutual exclusion. **Safety, Liveness, and Fairness in Mutual Exclusion (10 minutes)** • Define the properties of safety (no two processes enter a critical section simultaneously), liveness (every request for entry into a critical section is eventually granted), and fairness (no process is indefinitely delayed from entering its critical section). • Provide examples to illustrate how these properties are maintained in well-designed systems. **Conclusion and Q&A (5 minutes)** • Summarize the key points covered in the session. • Open the floor for any questions from the students to clarify doubts and deepen understanding.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Identify a real-world system that utilizes mutual exclusion, describe how it implements this concept, and discuss its impact on system performance. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.2	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Preliminaries of Distributed Systems**
Objectives	At the end of the lesson, the student shall be able to: a. Identify the basic components and architecture of distributed systems that support mutual exclusion. b. Discuss the fundamental properties and requirements (safety, liveness, and fairness) essential for mutual exclusion in distributed systems.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Distributed Systems (5 minutes)** • Provide a brief overview of distributed systems, defining what they are and the importance of their architecture in supporting various functionalities including mutual exclusion. **Components of Distributed Systems (10 minutes)** • Detail the primary components of a distributed system, such as nodes, communication links, and data stores. • Explain how these components interact within the system to facilitate operations across different physical and virtual environments. **Architecture Supporting Mutual Exclusion (10 minutes)** • Discuss different architectural models that support distributed mutual exclusion, such as client-server and peer-to-peer models. • Highlight the role of middleware in managing resource access and synchronization across distributed applications. **Safety, Liveness, and Fairness (10 minutes)** • Define critical properties needed for mutual exclusion in distributed systems: ○ **Safety**: ensuring that only one process can enter the critical section at a time. ○ **Liveness**: guarantees that requests for entering critical sections are eventually granted. ○ **Fairness**: ensuring that no process is indefinitely postponed from entering a critical section.

	- Provide examples of scenarios where each property is upheld or violated, discussing the consequences of such events. **Conclusion and Interactive Discussion (5 minutes)** - Recap the key concepts covered about the architecture and essential properties of distributed systems. - Engage with the students through a question-and-answer session to clarify any doubts and deepen their understanding of the material discussed.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** - Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. - Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 - Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: - Create a diagram of a basic distributed system architecture supporting mutual exclusion, labeling key components and their roles. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.3	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Lamport's Algorithm for Mutual Exclusion**
Objectives	At the end of the lesson, the student shall be able to: a. Describe the mechanism of Lamport's timestamp-based algorithm for achieving mutual exclusion. b. Analyze the efficiency and limitations of Lamport's algorithm in different distributed system environments.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Lamport's Algorithm (5 minutes)** • Start with a brief introduction to Leslie Lamport and his contributions to distributed systems. • Explain the significance of Lamport's algorithm in the context of mutual exclusion in distributed systems. **Core Principles of Lamport's Algorithm (10 minutes)** • Discuss the use of logical clocks to order events and resolve conflicts in a distributed environment. • Explain how these clocks help maintain the consistency of distributed processes. **Detailed Walkthrough of Lamport's Algorithm (10 minutes)** • Provide a step-by-step explanation of how the algorithm works, including the generation and use of timestamps to request and release access to a critical section. • Use diagrams to illustrate how different nodes interact using these timestamps. **Analysis of Lamport's Algorithm (10 minutes)** • Evaluate the efficiency of the algorithm in terms of message complexity and fault tolerance. • Discuss scenarios where Lamport's algorithm is particularly effective, and where it might face challenges. **Conclusion and Open Discussion (5 minutes)** • Recap the main points about Lamport's algorithm, emphasizing its role in ensuring mutual exclusion and the importance of logical clocks.

	- Invite questions from the students to address any uncertainties or deeper inquiries into the algorithm’s application and impact.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** - Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. - Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 - Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: - Compare Lamport’s algorithm to another mutual exclusion algorithm in terms of message complexity and fault tolerance. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.4	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Ricart-Agrawala Algorithm for Mutual Exclusion**
Objectives	At the end of the lesson, the student shall be able to: a. Explain the operation of the Ricart-Agrawala algorithm and its improvement over Lamport's algorithm. b. Evaluate the message complexity and performance of the Ricart-Agrawala algorithm under various conditions.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Ricart-Agrawala Algorithm (5 minutes)** • Begin with an overview of the Ricart-Agrawala algorithm, developed by Glenn Ricart and Ashok Agrawala, highlighting its significance in the field of distributed systems. • Mention how this algorithm improves upon earlier solutions like Lamport's algorithm by reducing the number of required messages. **Core Principles of Ricart-Agrawala Algorithm (10 minutes)** • Explain the fundamental concept of using timestamps for requesting access to a critical section similar to Lamport's algorithm but with enhancements in message efficiency. • Discuss the mutual exclusion properties ensured by the algorithm, emphasizing how it guarantees that only one process can enter a critical section at any given time. **Detailed Walkthrough of Ricart-Agrawala Algorithm (10 minutes)** • Provide a step-by-step breakdown of how processes communicate to request and release the critical section. • Use diagrams to illustrate the flow of request and reply messages between nodes, detailing how decisions are made regarding who enters the critical section. **Analysis of Ricart-Agrawala Algorithm (10 minutes)** • Assess the performance of the Ricart-Agrawala algorithm in terms of message complexity and response time.

	• Consider real-world scenarios where this algorithm would be particularly beneficial, and discuss any potential limitations in terms of scalability and network load. **Conclusion and Interactive Discussion (5 minutes)** • Summarize the key aspects of the Ricart-Agrawala algorithm, emphasizing its role in efficient mutual exclusion in distributed environments. • Encourage an open discussion, inviting questions and comments from students to deepen their understanding of the algorithm's mechanics and applications.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Analyze a case where the Ricart-Agrawala algorithm would improve system efficiency, detailing the scenario and expected outcomes. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.5	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Maekawa’s Algorithm for Mutual Exclusion**
Objectives	At the end of the lesson, the student shall be able to: a. Outline Maekawa's algorithm and its unique approach using a quorum-based system. b. Assess how Maekawa’s algorithm addresses scalability and message overhead issues.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Maekawa's Algorithm (5 minutes)** • Begin with a brief introduction to Maekawa’s Algorithm, detailing its development by Keiichi Maekawa as a solution to the mutual exclusion problem in distributed systems. • Highlight the uniqueness of the algorithm, which uses a quorum-based approach to decide which process can enter a critical section. **Core Principles of Maekawa’s Algorithm (10 minutes)** • Explain the concept of a quorum and how it is used in Maekawa’s algorithm to reduce the number of messages compared to other mutual exclusion algorithms. • Discuss the voting system, where each process must receive permission from a majority of a predefined subset of processes (a quorum) before it can enter the critical section. **Detailed Walkthrough of Maekawa’s Algorithm (10 minutes)** • Provide a detailed step-by-step explanation of the algorithm, focusing on the processes of sending requests, receiving votes, and entering/exiting the critical section. • Use diagrams to visually represent how requests are sent to quorum members and how votes are collected. **Analysis of Maekawa’s Algorithm (10 minutes)** • Evaluate the strengths of Maekawa's algorithm, particularly its efficiency in managing the number of messages that need to be exchanged. • Discuss potential issues such as deadlocks and starvation, explaining the conditions under which they may occur and how they can be mitigated.

	Conclusion and Interactive Discussion (5 minutes) • Summarize the main points about Maekawa's algorithm, emphasizing its role in providing an efficient solution for mutual exclusion in distributed systems. • Open the floor for questions and foster a discussion to clarify any complex aspects of the algorithm or to explore deeper into how it can be implemented in real-world systems.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Evaluate a distributed system scenario where Maekawa's algorithm could be applied, and discuss its advantages over other algorithms. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.6	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Suzuki–Kasami's Broadcast Algorithm**
Objectives	At the end of the lesson, the student shall be able to: a. Discuss the Suzuki-Kasami algorithm's broadcast system for requesting and granting access to a critical section. b. Evaluate the benefits and drawbacks of using broadcast mechanisms in mutual exclusion algorithms.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Suzuki–Kasami's Broadcast Algorithm (5 minutes)** • Start by introducing the Suzuki–Kasami algorithm, developed by Ichiro Suzuki and Tadao Kasami, which is a token-based broadcast solution for managing mutual exclusion in distributed systems. • Highlight the key feature of this algorithm, which is its efficient use of a token to control access to a critical section, reducing the need for continuous message exchanges. **Core Principles of the Suzuki–Kasami Algorithm (10 minutes)** • Explain the role of the token in this algorithm, which contains the information necessary for deciding which process gets access to the critical section next. • Discuss how the broadcast mechanism works, where a process sends a request to all other processes and how the token is passed among the processes. **Detailed Walkthrough of Suzuki–Kasami's Algorithm (10 minutes)** • Provide a detailed, step-by-step explanation of how a process requests access to a critical section, how it receives the token, and what happens when the critical section is released. • Use diagrams to visually demonstrate the flow of requests, the token's movement, and the actions taken by each process during these stages. **Analysis of Suzuki–Kasami's Algorithm (10 minutes)**

	• Evaluate the effectiveness of the Suzuki–Kasami algorithm in terms of message overhead and response time to requests for entering the critical section. • Discuss the advantages of using a token-based system and the potential challenges it faces, such as the token becoming a single point of failure and issues related to its loss or duplication. **Conclusion and Interactive Discussion (5 minutes)** • Summarize the key aspects of the Suzuki–Kasami broadcast algorithm, emphasizing its efficiency and practicality for distributed systems requiring frequent and fair access to a shared resource. • Encourage questions and open a discussion for students to explore further nuances or hypothetical scenarios where the algorithm might be implemented or adapted.
Closure	**1.** Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Research and describe a scenario where the Suzuki-Kasami broadcast algorithm is particularly effective or ineffective, explaining why. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.7	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Introduction to Deadlock Detection in Distributed Systems**
Objectives	At the end of the lesson, the student shall be able to: a. Define deadlocks in the context of distributed systems and explain their implications on system performance and stability. b. Review different types of deadlocks and the conditions leading to their occurrence.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Deadlocks (5 minutes)** • Begin the lesson by defining what a deadlock is, specifically within the context of distributed systems. • Highlight the importance of understanding deadlocks due to their potential to significantly impact system performance and stability. **Characteristics of Deadlocks (10 minutes)** • Explain the four necessary conditions for a deadlock to occur, often referred to as the Coffman conditions: mutual exclusion, hold and wait, no preemption, and circular wait. • Discuss how these conditions manifest uniquely in distributed environments where resources are spread across multiple systems. **Types of Deadlocks in Distributed Systems (10 minutes)** • Delve into the various types of deadlocks that can occur in distributed systems, such as communication deadlocks and resource deadlocks. • Use diagrams to illustrate scenarios in which these different types of deadlocks may arise, helping to visualize the complexities involved. **Conditions Leading to Deadlock Occurrence (10 minutes)** • Detail the specific conditions and sequences of events that can lead to deadlocks in distributed systems. • Discuss real-world examples of deadlocks, possibly incorporating case studies or anecdotal accounts from well-known software issues or system failures. **Conclusion and Recap (5 minutes)**

	• Summarize the main points discussed during the lesson, emphasizing the critical nature of detecting and preventing deadlocks in distributed systems. • Open the floor for questions, allowing students to clarify their understanding or explore more deeply into any of the topics covered.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Identify a type of deadlock specific to a known distributed system and explain the conditions that lead to its occurrence. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.8	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Algorithms for Deadlock Detection and Resolution**
Objectives	At the end of the lesson, the student shall be able to: a. Examine Knapp's classification and various algorithms for detecting and resolving deadlocks in single and multiple resource models. b. Analyze the complexity and feasibility of implementing these algorithms in real-world systems.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Deadlock Detection Algorithms (5 minutes)** • Start with an overview of why deadlock detection and resolution are critical for maintaining system reliability and performance in distributed systems. • Briefly introduce the types of algorithms that will be covered, focusing on Knapp's classification. **Knapp's Classification of Deadlocks (10 minutes)** • Explain Knapp's method of classifying deadlocks into various types, each with distinct characteristics and detection strategies. • Provide a detailed overview of how this classification helps in understanding and applying appropriate detection algorithms. **Algorithms for Single and Multiple Resource Models (10 minutes)** • Discuss specific algorithms used for detecting and resolving deadlocks in systems with single and multiple resource models. • Use examples or case studies to illustrate how these algorithms function in practical scenarios, detailing the process of monitoring, detecting, and resolving deadlocks. **Complexity and Feasibility of Implementation (10 minutes)** • Analyze the computational and operational complexities associated with implementing these algorithms in real-world systems. • Discuss factors that influence the choice of an algorithm, such as system architecture, resource type, and the expected frequency of deadlock occurrences.

	Conclusion and Discussion (5 minutes) • Recap the key points from the lesson, emphasizing the importance of selecting the appropriate algorithm based on Knapp's classification and the specific system requirements. • Encourage questions and open a discussion for students to share their thoughts on the practical challenges they might anticipate when implementing these algorithms in their future careers or projects.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Choose one algorithm for deadlock detection and resolution, and analyze its application in a multi-resource model with specific examples. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.9	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Algorithms for Deadlock Detection: Single Resource, AND, and OR Models**
Objectives	At the end of the lesson, the student shall be able to: a. Describe the algorithms used for deadlock detection in systems with single resource models, highlighting their operational mechanisms and conditions under which they effectively prevent or resolve deadlocks. b. Analyze the complexities and implementation considerations for deadlock algorithms in the AND and OR models, comparing their effectiveness in different system architectures and scenarios.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Deadlock Detection in Single Resource Models (5 minutes)** • Begin by explaining what single resource models are and why deadlock detection is crucial in such environments. • Introduce the basic algorithm used for deadlock detection in single resource models, emphasizing its operational mechanism. **Detailed Explanation of Single Resource Model Algorithms (10 minutes)** • Describe the specific steps and conditions under which the single resource model algorithm detects and resolves deadlocks. • Use a simple diagram or flowchart to illustrate how the algorithm monitors and manages resource allocation to prevent deadlock situations. **Introduction to AND and OR Models for Deadlock Detection (5 minutes)** • Explain the concepts of AND and OR models in deadlock detection, including how these models differ from single resource models. • Outline the scenarios where each model is typically applied in distributed systems. **Algorithms for AND and OR Models (10 minutes)** • Dive into the specific algorithms used for deadlock detection in AND and OR models.

	• Discuss the operational mechanisms of these algorithms, using examples to show how they handle deadlock detection and resolution. **Complexities and Implementation Considerations (5 minutes)** • Analyze the complexities involved in implementing these algorithms in real-world systems. • Discuss the factors that affect the selection of an appropriate deadlock detection algorithm, such as system size, resource type, and the frequency of transactions. **Conclusion and Comparative Analysis (5 minutes)** • Summarize the key points covered in the lesson, highlighting the differences in effectiveness of deadlock detection algorithms across single resource, AND, and OR models. • Encourage students to ask questions and discuss the practical challenges they might face when implementing these algorithms in various system architectures.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Analyze and describe the application of a deadlock detection algorithm for a chosen system, focusing on its operational mechanism, conditions for effectiveness, and implementation complexities Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.1	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	Introduction to Checkpointing and Rollback Recovery
Objectives	At the end of the lesson, the student shall be able to: a. Understand the fundamental concepts and significance of checkpointing and rollback recovery in distributed systems. b. Explore the basic types and functions of recovery mechanisms that ensure system reliability and data integrity.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Checkpointing and Rollback Recovery Concepts (5 minutes)** • Start by defining checkpointing and rollback recovery, emphasizing their critical roles in ensuring data consistency and system reliability within distributed systems. • Highlight how these mechanisms prevent data loss and enable systems to recover quickly from hardware failures or software errors. **Fundamental Types of Recovery Mechanisms (10 minutes)** • Explore the various types of checkpointing methods such as full checkpointing, incremental checkpointing, and differential checkpointing. • Discuss rollback recovery techniques and how they interact with different types of checkpointing to restore system state. **Practical Applications and Benefits (10 minutes)** • Illustrate with real-world scenarios where checkpointing and rollback recovery are employed, such as in database management systems and cloud computing services. • Discuss the benefits of each type in terms of recovery time objectives (RTO) and recovery point objectives (RPO). **Challenges in Implementing Recovery Mechanisms (10 minutes)** • Analyze the challenges and limitations of implementing checkpointing and rollback recovery in distributed systems, including issues of storage overhead, performance trade-offs, and complexity of coordination. • Provide examples of common pitfalls and how they can be mitigated through proper system design and periodic testing. **Conclusion and Interactive Discussion (5 minutes)**

	• Recap the main points covered, stressing the importance of checkpointing and rollback recovery in maintaining high availability and data integrity. • Open the floor for questions and encourage discussion about the students' experiences or hypothetical application of these recovery strategies in different technological environments.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Research and write a brief report on how checkpointing and rollback recovery are implemented in a well-known distributed system, such as a major cloud platform. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.2	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Background and Definitions of Recovery Strategies**
Objectives	a) At the end of the lesson, the student shall be able to: b) Define key recovery strategies including checkpointing and logging. c) Establish the historical context and technological advancements that have shaped current recovery techniques.
Teaching Aids (if any)	a) PPTs. b) Green board (Chalk and Talk). c) Video Lectures by NPTEL / Youtube d) Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Recovery Strategies in Computing (5 minutes)** • Begin with an overview of recovery strategies, defining key concepts like checkpointing, logging, and replication. • Provide a brief historical perspective on how these strategies have evolved over time, highlighting major technological milestones. **Definitions and Key Components (10 minutes)** • Define and explain the critical components of recovery strategies, including physical logging, logical logging, and state snapshotting. • Distinguish between synchronous and asynchronous strategies, explaining the implications of each for system performance and data consistency. **Technological Advancements in Recovery Strategies (10 minutes)** • Discuss the impact of advancements in storage technologies, network speed, and computing power on the design and efficiency of recovery strategies. • Explore modern approaches such as distributed snapshots and log-structured merge-trees used in high-performance systems. **Case Studies and Real-World Applications (10 minutes)** • Present case studies illustrating the implementation of sophisticated recovery strategies in large-scale distributed systems like financial trading platforms and global e-commerce sites. • Analyze how these strategies contribute to business continuity and disaster recovery planning. **Conclusion and Q&A (5 minutes)**

	• Summarize the session by highlighting the importance of understanding both the theoretical and practical aspects of recovery strategies. • Invite questions from students, fostering a deeper discussion on how recovery strategies can be tailored to meet specific system requirements and business objectives.
Closure	**1.** Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2.** **Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Prepare a timeline that highlights key milestones in the development of recovery strategies in computing, focusing on the evolution from early backup systems to modern distributed recovery techniques. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.3	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Analyzing Issues in Failure Recovery**
Objectives	At the end of the lesson, the student shall be able to: a) Identify common problems and challenges associated with failure recovery in distributed systems. b) Discuss potential impacts of these issues on system performance and stability.
Teaching Aids (if any)	a) PPTs. b) Green board (Chalk and Talk). c) Video Lectures by NPTEL / Youtube d) Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Failure Recovery Challenges (5 minutes)** • Begin by explaining the significance of robust failure recovery mechanisms in distributed systems. • Highlight how failure recovery is crucial for maintaining system integrity and availability. **Common Problems in Failure Recovery (10 minutes)** • Identify and explain common problems such as loss of data consistency, recovery latency, and partial failures in distributed systems. • Discuss how these issues complicate the recovery process, potentially leading to cascading failures. **Challenges in Implementing Recovery Mechanisms (10 minutes)** • Dive into the technical challenges like network partitioning, synchronization overhead, and state inconsistency during recovery. • Illustrate with examples the impact of these challenges on recovery time and system throughput. **Impact on System Performance and Stability (10 minutes)** • Discuss the potential adverse effects of poor recovery strategies on system performance, such as increased downtime and reduced throughput. • Analyze how stability issues can arise from incomplete or incorrect recovery, potentially leading to repeated failures. **Conclusion and Discussion (5 minutes)**

	• Summarize the key issues and challenges discussed, emphasizing the importance of designing distributed systems with robust and efficient failure recovery mechanisms. • Open the floor for a discussion, encouraging students to ask questions or share their insights on improving failure recovery in distributed systems.
Closure	**1.** Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Identify a recent real-world example of a failure recovery issue in a distributed system, describe the problem, and discuss its impact on system performance and stability. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.4	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Implementing Checkpoint-based Recovery**
Objectives	At the end of the lesson, the student shall be able to: a) Examine different methods of checkpointing and their applicability in various system architectures. b) Evaluate the effectiveness and efficiency of checkpoint-based recovery systems.
Teaching Aids (if any)	a) PPTs. b) Green board (Chalk and Talk). c) Video Lectures by NPTEL / Youtube d) Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Checkpoint-based Recovery (5 minutes)** • Start by defining checkpoint-based recovery and its role in maintaining system state consistency across distributed systems. • Outline different types of checkpoints, such as full, incremental, and differential. **Methods of Checkpointing (10 minutes)** • Explore various methods of checkpointing, explaining how each works and their suitability for different distributed system architectures. • Provide examples of systems where each checkpointing method might be preferred, highlighting the reasons for their selection. **Applicability in System Architectures (10 minutes)** • Discuss the application of checkpoint-based recovery in various architectures, including client-server, peer-to-peer, and cloud-based systems. • Highlight considerations such as resource availability, network topology, and system scale that influence the choice of checkpointing method. **Evaluating Effectiveness and Efficiency (10 minutes)** • Evaluate the effectiveness of checkpoint-based recovery systems, focusing on metrics such as recovery time, data integrity, and operational overhead. • Discuss the efficiency of different checkpointing methods, considering factors like storage requirements, system impact during checkpointing, and ease of restoration. **Conclusion and Interactive Discussion (5 minutes)**

	• Recap the main points covered about implementing checkpoint-based recovery, stressing the importance of selecting the right method based on specific system requirements. • Encourage an interactive discussion where students can ask questions or propose scenarios where different checkpointing methods might be applied.
Closure	**1.** Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Choose a specific distributed system architecture and propose an appropriate checkpointing method for it, justifying your choice based on the architecture's characteristics and needs. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.5	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Log-based Rollback Recovery Techniques**
Objectives	At the end of the lesson, the student shall be able to: a) Describe the process and advantages of log-based rollback recovery in maintaining system state after failures. b) Analyze case studies to illustrate the practical application and challenges of log-based recovery.
Teaching Aids (if any)	a) PPTs. b) Green board (Chalk and Talk). c) Video Lectures by NPTEL / Youtube d) Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Log-based Rollback Recovery (5 minutes)** • Define log-based rollback recovery and its role in maintaining system state after failures. • Outline the basic components of this recovery method, including logging actions and recovery points. **Detailed Process of Log-based Recovery (10 minutes)** • Describe the process of capturing logs, including what data is logged and how these logs are maintained to ensure they are failure-resilient. • Explain the steps involved in rolling back to a previous state using these logs when a failure occurs. **Advantages of Log-based Recovery (10 minutes)** • Discuss the key benefits of log-based recovery, such as minimal downtime, precise state restoration, and the ability to handle multiple types of failures. • Highlight scenarios where log-based recovery is particularly effective, including complex transaction systems. **Case Studies and Practical Challenges (10 minutes)** • Analyze case studies from sectors like banking and telecommunications that utilize log-based rollback recovery. • Discuss the challenges encountered, such as managing large log volumes, ensuring log integrity, and optimizing recovery times. **Conclusion and Discussion (5 minutes)** • Recap the major points about log-based rollback recovery, stressing its importance in ensuring robust system recovery.

	• Open the floor for discussion, encouraging students to ask questions and consider how log-based recovery might be implemented in their projects or future work environments.
Closure	**1.** Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Analyze a case study where log-based rollback recovery was used, focusing on how the recovery process was managed and the challenges encountered during implementation. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.6	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Coordinated Checkpointing Algorithms**
Objectives	At the end of the lesson, the student shall be able to: a) Explain the necessity and operation of coordinated checkpointing in complex distributed systems. b) Assess the synchronization requirements and challenges involved in implementing coordinated checkpointing.
Teaching Aids (if any)	a) PPTs. b) Green board (Chalk and Talk). c) Video Lectures by NPTEL / Youtube d) Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Coordinated Checkpointing (5 minutes)** • Explain what coordinated checkpointing is and why it is necessary in complex distributed systems. • Discuss the primary goal of coordinated checkpointing—to create a consistent snapshot of the system across all nodes. **Operation of Coordinated Checkpointing (10 minutes)** • Describe the operational mechanism of coordinated checkpointing, including the initiation of checkpoints and the coordination among nodes. • Illustrate with a simple diagram how a coordinated checkpointing algorithm ensures all nodes are synchronized during the checkpoint process. **Synchronization Requirements (10 minutes)** • Delve into the synchronization aspects, explaining how time-stamping or sequencing can be used to manage state consistency across distributed components. • Discuss the challenges related to network delays, message ordering, and fault tolerance in the synchronization process. **Assessment of Implementation Challenges (10 minutes)** • Assess the common challenges in implementing coordinated checkpointing, such as handling node failures during checkpointing, minimizing the performance overhead, and dealing with network partitions. • Provide strategies to overcome these challenges, using examples from existing implementations.

	Conclusion and Interactive Discussion (5 minutes) • Summarize the importance and complexities of implementing coordinated checkpointing in maintaining data consistency. • Invite questions and discussion on the application of coordinated checkpointing in real-world systems, encouraging students to consider both theoretical and practical aspects.
Closure	**1.** Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Explain the role of coordinated checkpointing in a distributed system of your choice and discuss the synchronization challenges that might arise in that specific context. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.7	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Consensus and Agreement in Distributed Systems**
Objectives	At the end of the lesson, the student shall be able to: a) Define the problem of achieving consensus in distributed systems, particularly in the presence of system failures. b) Review key agreement algorithms and their effectiveness in ensuring system consistency and reliability in both failure-free and failure-prone environments.
Teaching Aids (if any)	a) PPTs. b) Green board (Chalk and Talk). c) Video Lectures by NPTEL / Youtube d) Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Defining the Consensus Problem (5 minutes)** • Start with a definition of consensus in distributed systems and explain its critical role, particularly in the presence of system failures. • Outline the primary objectives of consensus algorithms: agreement, reliability, and termination. **Overview of Agreement Algorithms (10 minutes)** • Review various agreement algorithms, highlighting how they work in failure-free environments and their adaptation to handle failures. • Discuss popular algorithms like Paxos and Raft, noting their use cases and operational principles. **Agreement in Synchronous Systems with Failures (10 minutes)** • Explain how consensus algorithms operate in synchronous systems and address challenges introduced by node and network failures. • Provide examples of how these systems ensure consistency and reliability even when components fail. **Effectiveness of Agreement Algorithms (10 minutes)** • Analyze the effectiveness of these algorithms in ensuring system consistency and reliability, discussing both their strengths and limitations. • Compare how these algorithms perform in both failure-free and failure-prone environments, using case studies or theoretical examples. **Conclusion and Q&A Session (5 minutes)**

	- Recap the key points on consensus and agreement in distributed systems, emphasizing the importance of these algorithms in maintaining system integrity. - Open the floor for questions, allowing students to explore the complexities and nuances of achieving consensus in real-world distributed systems.
Closure	**1.** Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** - Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. - Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 - Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: - Evaluate the effectiveness of a chosen consensus algorithm in a known distributed system, particularly focusing on its performance during system failures. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.1	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Introduction to Peer-to-Peer Computing and Overlay Graphs**
Objectives	At the end of the lesson, the student shall be able to: a. Understand the fundamental concepts of peer-to-peer (P2P) networks and the use of overlay graphs in these networks. b. Explore the architectural design and topological structure of various P2P systems and how they enhance connectivity and data distribution.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Understanding the Basics of Peer-to-Peer Networks (5 minutes)** • Start by defining what peer-to-peer (P2P) computing is and differentiate it from client-server models. • Highlight the decentralized nature of P2P systems and their importance in modern computing. **Exploring Overlay Graphs (10 minutes)** • Explain the concept of overlay graphs and how they are used to organize and manage connections in a P2P network. • Show examples of overlay graphs and how they enhance data routing and resilience in network disruptions. **Architectural Design and Topological Structure (10 minutes)** • Discuss the architectural designs commonly seen in P2P systems such as structured, unstructured, and hybrid overlays. • Analyze how different topological structures impact data distribution and network scalability. **Benefits and Challenges of P2P Systems (10 minutes)** • Review the main benefits of P2P networks, including scalability, fault tolerance, and reduced dependence on centralized resources. • Address the challenges involved, such as security, data redundancy, and maintaining consistency. **Conclusion and Open Discussion (5 minutes)** • Summarize the key aspects of P2P computing and overlay graphs.

	• Encourage questions and promote a discussion on potential innovations or future trends in P2P technology.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Write a summary on the role of overlay graphs in enhancing connectivity in a chosen P2P network example. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.2	**Course Name: Distributed Systems**	**Course No.: MCSE 104**

Topics	**Data Indexing and Overlays in P2P Systems**
Objectives	At the end of the lesson, the student shall be able to: a. Discuss the methods and importance of data indexing in managing and retrieving data efficiently in P2P networks. b. Examine different overlay network designs and their impact on the performance and scalability of P2P systems.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Data Indexing in P2P Networks (5 minutes)** • Begin with an explanation of data indexing and why it is critical for efficient data retrieval in distributed networks. • Introduce various indexing techniques used in P2P systems. **Overlay Network Designs (10 minutes)** • Describe different types of overlay networks such as ring, tree, and mesh structures. • Discuss how these overlays support indexing and affect the performance and scalability of P2P networks. **Data Indexing Methods and Their Impact (10 minutes)** • Explore specific data indexing methods like Distributed Hash Tables (DHTs) and their role in data localization and retrieval. • Use case studies or examples to illustrate how these methods are implemented in real-world P2P systems. **Performance and Scalability Considerations (10 minutes)** • Analyze how effective data indexing contributes to the overall performance of a P2P network. • Evaluate scalability challenges in P2P networks and how different data indexing and overlay designs help address these issues. **Conclusion and Group Discussion (5 minutes)** • Recap the importance of efficient data indexing and well-designed overlay networks in P2P systems. • Open the floor for a discussion on potential improvements or emerging technologies in data indexing and network overlays.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Research and describe a specific overlay network design and discuss its impact on data indexing and retrieval efficiency. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.3	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	Chord Protocol in Peer-to-Peer Networks
Objectives	At the end of the lesson, the student shall be able to: a. Explain the Chord protocol and its role in facilitating efficient data location and retrieval in a decentralized environment. b. Analyze the Chord protocol's algorithmic underpinnings, including its hash function and consistent hashing for node organization.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to the Chord Protocol (5 minutes)** • Start with a brief explanation of the Chord protocol, detailing its purpose and fundamental operation within P2P networks. • Highlight the protocol’s role in facilitating efficient data location and retrieval across distributed nodes. **Understanding Chord’s Algorithmic Foundations (10 minutes)** • Dive into the specifics of the Chord protocol, including its use of a consistent hashing mechanism to distribute data among nodes. • Explain how Chord manages node joins and departures, ensuring minimal disruption in the network. **Chord's Hash Function and Node Organization (10 minutes)** • Discuss the hash function used by Chord to assign keys to nodes and how it ensures a balanced load distribution. • Illustrate with diagrams the ring structure typical of Chord, showing how nodes and data are organized. **Evaluating Chord’s Efficiency and Scalability (10 minutes)** • Analyze the efficiency of Chord in terms of lookup speed and network traffic. • Evaluate the scalability of the Chord protocol, particularly how it handles large-scale networks and high churn rates. **Conclusion and Interactive Discussion (5 minutes)** • Recap the main features of the Chord protocol and its importance in the landscape of distributed computing.

	• Invite questions and facilitate a discussion about the potential enhancements or alternatives to Chord in modern distributed systems.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Analyze the implementation of the Chord protocol in a real-world application, focusing on its efficiency in data location and retrieval.. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.4	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Content Addressable Networks**
Objectives	At the end of the lesson, the student shall be able to: a. Describe the structure and functioning of Content Addressable Networks (CAN) and their application in P2P systems. b. Evaluate the advantages and challenges associated with CAN, particularly in terms of scalability and fault tolerance.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Content Addressable Networks (CAN) (5 minutes)** • Begin by defining Content Addressable Networks and their primary function in P2P computing. • Outline the basic principles of CAN, emphasizing its decentralized design and self-organizing nature. **Structure and Functioning of CAN (10 minutes)** • Describe the multi-dimensional Cartesian coordinate space used by CAN to partition and store data. • Show how data is placed, located, and retrieved through the coordinate space, enhancing understanding through visual aids. **Advantages of Using CAN (10 minutes)** • Discuss the key advantages of CAN, such as robustness against node failures, efficient routing, and load balancing. • Provide examples of applications where CAN's unique characteristics provide significant benefits over other network designs. **Challenges Associated with CAN (10 minutes)** • Identify and analyze the challenges and limitations of CAN, such as scalability issues related to space partitioning and the overhead of maintaining the coordinate routing table. • Discuss potential solutions to these challenges, referencing current research and technological advances. **Conclusion and Group Discussion (5 minutes)** • Summarize the significant aspects of Content Addressable Networks, reinforcing their role in distributed data management.

	• Encourage an open discussion on the future directions of CAN and how emerging technologies might influence its development and application in distributed systems.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Evaluate the use of a Content Addressable Network in a specific case study, highlighting its scalability and fault toleranc. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.5	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Tapestry Overlay Network**
Objectives	At the end of the lesson, the student shall be able to: a. Define the Tapestry network and its unique approach to routing and data placement in distributed environments. b. Assess how Tapestry handles dynamic network changes and maintains efficient routing and data lookup.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Tapestry (5 minutes)** • Begin with an explanation of the Tapestry network, focusing on its origins as an overlay for scalable, distributed applications. • Highlight the principles of location-based routing and object location that define Tapestry's functionality. **Understanding Tapestry’s Network Design (10 minutes)** • Explore the architecture of the Tapestry network, emphasizing its decentralized, scalable, and self-stabilizing nature. • Explain the use of unique identifiers for nodes and objects within the network to facilitate efficient routing and data location. **Dynamic Network Adaptations in Tapestry (10 minutes)** • Discuss how Tapestry handles dynamic changes in the network, such as node additions and departures, without significant disruption. • Illustrate the process of maintaining routing information in the face of network changes, using detailed diagrams and examples. **Efficiency and Fault Tolerance in Tapestry (10 minutes)** • Analyze the efficiency of Tapestry in terms of network traffic and speed of data retrieval. • Evaluate Tapestry’s fault tolerance capabilities, detailing how the network ensures data availability and reliability despite potential node failures. **Conclusion and Interactive Discussion (5 minutes)** • Recap the critical features and benefits of the Tapestry overlay network.

	- Encourage questions and facilitate a discussion on the potential for integrating Tapestry into current and future distributed systems, exploring innovative uses and improvements.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** - Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. - Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 - Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: - Explore and document how the Tapestry network adapts to dynamic changes and maintains data consistency in a distributed environment. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.6	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Introduction to Distributed Shared Memory (DSM)**
Objectives	At the end of the lesson, the student shall be able to: a. Introduce the concept of distributed shared memory, its abstraction, and the advantages it provides in distributed computing environments. b. Explore the basic operational principles of DSM and how it simulates a coherent memory system on non-shared architectures.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Conceptual Overview of DSM (5 minutes)** • Start by defining Distributed Shared Memory and its role in abstracting a coherent memory model across computers in a distributed system. • Highlight the advantages of using DSM, such as ease of programming and increased flexibility in data sharing across distributed applications. **Operational Principles of DSM (10 minutes)** • Delve into the operational mechanisms of DSM, explaining how it simulates shared memory on systems that are physically distributed. • Discuss the types of DSM systems, categorized based on their consistency models and synchronization mechanisms. **Benefits of DSM in Distributed Computing (10 minutes)** • Review the key benefits of DSM, focusing on how it enhances collaboration and resource sharing among distributed nodes. • Provide examples from industry or research that illustrate successful implementations of DSM and the impact on system performance and scalability. **Challenges and Limitations of DSM (10 minutes)** • Identify the challenges associated with implementing DSM, such as maintaining consistency, managing latency, and ensuring memory coherence. • Discuss strategies used to overcome these challenges, including various consistency protocols and synchronization techniques. **Conclusion and Group Discussion (5 minutes)**

	• Summarize the critical aspects of Distributed Shared Memory and its significance in modern distributed systems. • Open the floor for discussion, encouraging students to share their thoughts on how DSM could be implemented in their projects or potential areas of research.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Provide an analysis of a distributed system using DSM, focusing on how it simulates a coherent memory model across distributed nodes. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.7	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	**Memory Consistency Models in DSM**
Objectives	At the end of the lesson, the student shall be able to: a. Define various memory consistency models used in DSM systems, such as sequential consistency, causal consistency, and eventual consistency. b. Discuss the implications of each consistency model on the development and performance of distributed applications.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Memory Consistency Models (5 minutes)** • Begin by explaining what memory consistency models are and why they are critical for the correct operation of distributed shared memory systems. • Outline the role these models play in determining the order and visibility of memory operations across different nodes in a distributed system. **Exploring Different Consistency Models (10 minutes)** • Delve into various consistency models used in distributed systems, such as sequential consistency, causal consistency, and eventual consistency. • Discuss the theoretical foundations of each model and provide examples to illustrate how they ensure memory operations appear consistent across nodes. **Implications of Consistency Models on Application Development (10 minutes)** • Explore how different consistency models affect application development, focusing on the trade-offs between consistency, availability, and partition tolerance. • Examine the impact of choosing a particular model on the complexity and performance of distributed applications. **Challenges in Implementing Consistency Models (10 minutes)** • Discuss the technical challenges associated with implementing these models in real-world distributed systems, such as synchronization overhead and latency issues.

	• Analyze case studies where specific consistency models either succeeded in providing the necessary guarantees or led to problems due to their limitations. **Conclusion and Interactive Discussion (5 minutes)** • Recap the importance of understanding and selecting appropriate memory consistency models for distributed shared memory systems. • Invite questions and encourage a discussion on potential areas of research or innovation within memory consistency models, asking students to consider their implications in emerging technologies.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , “ Distributed Operating Systems: Concepts and Design”, Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., “Advanced concepts in operating systems”, McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., ” Distributed Systems: Principles and Paradigms”, 2nd edition, 2007. **Home work**: • Compare and contrast two memory consistency models in DSM, illustrating their implications on system performance with real-world examples. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.8	Course Name: Distributed Systems	Course No.: MCSE 104

Topics	Shared Memory Mutual Exclusion in DSM
Objectives	At the end of the lesson, the student shall be able to: a. Discuss the importance of mutual exclusion mechanisms in maintaining consistency and order in access to shared memory in DSM. b. Examine techniques and algorithms like locks, semaphores, and barriers that are implemented to ensure mutual exclusion in distributed shared memory environments.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube d. Use of Nearpod / Mentimeter / Kahoot tools.
Teaching Development	**Introduction to Mutual Exclusion in DSM (5 minutes)** • Start with a definition of mutual exclusion and its necessity in the context of distributed shared memory to prevent concurrent access conflicts. • Highlight the relevance of mutual exclusion mechanisms in maintaining data integrity and system stability. **Mechanisms for Ensuring Mutual Exclusion (10 minutes)** • Describe various techniques and algorithms used to achieve mutual exclusion in distributed shared memory environments, such as locks, semaphores, and barriers. • Provide a detailed explanation of how these mechanisms work, including their operational details and usage scenarios. **Evaluating Mutual Exclusion Techniques (10 minutes)** • Compare different mutual exclusion techniques in terms of efficiency, scalability, and fairness. • Discuss the advantages and limitations of each technique, focusing on their suitability for different types of DSM architectures and application requirements. **Case Studies and Practical Applications (10 minutes)** • Analyze real-world implementations of mutual exclusion in distributed systems, focusing on how these implementations address common challenges like deadlock and starvation.

	• Examine specific case studies to illustrate the effectiveness of mutual exclusion strategies in maintaining system coherence and consistency. **Conclusion and Group Discussion (5 minutes)** • Summarize the critical aspects of implementing mutual exclusion in distributed shared memory systems. • Open the floor for discussion, encouraging students to propose ideas for innovative mutual exclusion mechanisms or improvements to existing techniques, fostering a deeper understanding of the complexities involved.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. **2. Suggested Reading books:** • Pradeep K Sinha , " Distributed Operating Systems: Concepts and Design", Prentice Hall of India, 2007. • Mukesh Singhal and Niranjan G. Shivaratri., "Advanced concepts in operating systems", McGraw-Hill, Inc., 1994 • Tanenbaum A.S., Van Steen M., " Distributed Systems: Principles and Paradigms", 2nd edition, 2007. **Home work**: • Research and explain how mutual exclusion is maintained in a distributed shared memory environment using a specific technique or algorithm. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents