CS6601 DISTRIBUTED SYSTEMS

UNIT I
INTRODUCTION

SYLLABUS: Introduction – Examples of Distributed Systems–Trends in Distributed Systems – Focus on resource sharing – Challenges. Case study: World Wide Web

COURSE OBJECTIVE: Understand foundations of Distributed Systems

PART A

1. Define distributed systems?

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

(Or)

A *distributed system* is one in which components located at networked computers communicate and coordinate their actions only by passing messages. This definition leads to the following characteristics of distributed systems:

- Concurrency of components
- Lack of a global 'clock'
- Independent failures of components

2. Give examples of distributed systems.

- Intranet and Local Area Network
- Internet/World-Wide Web
- Mobile and Ubiquitous Computing
- Database Management System
- Automatic Teller Machine Network
- E-mail
- Newsgroups and bulletin boards
- FTP, Telnet
- W WW
- Distributed computing
- Electronic payment
- Distributed real-time processing

3. Write the following
(i)HTTP (ii) HTML (iii) URL

The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, hypermedia information systems.[1] HTTP is the foundation of data communication for the Web. Hypertext is structured text that uses logical links (hyperlinks) between nodes containing text. HTTP is the protocol to exchange or transfer hypertext.

(ii) HTML is the standard markup language used to create web pages.[1]W~~eb browsers can~~ read HTML files and render them into visible or audible web pages. HTML describes the structure of a website semantically along with cues for presentation, making it a markup language, rather than a programming language

(iii) URL is an acronym for Uniform Resource Locator and is a reference (an address) to a resource on the Internet. A **URL** has two main components: Protocol identifier: For the **URL**

http://example.com, the protocol identifier is http. Resource name: For the **URL** http://example.com, the resource name is example.com.

4. Why Web Services are used?

- Interoperability
- Deployability - Web Services are deployed over standard Internet technologies.
- Low Cost of Communication
- Usability
- Reusability

5. Define heterogeneity.

This term means the diversity of the distributed systems in terms of hardware, software, platform, etc. Modern distributed systems will likely span different:

- Hardware devices: computers, tablets, mobile phones, embedded devices, etc.
- Operating System: MS Windows, Linux, Mac, UNIX, etc.
- Network: Local network, the Internet, wireless network, satellite links, etc.
- Programming languages: Java, C/C++, Python, PHP, etc.
- Different roles of software developers, designers, system managers

6. What are the characteristics of heterogeneity?

- Portability, interoperability
- Mobile code, adaptability (applets, agents)
- Middleware (CORBA etc.)
- Degree of transparency, Latency, Location-based services

7. What is the purpose of heterogeneity mobile code?

- The term **mobile code** refers to code that can be sent from one computer to another and run at the destination – Java applets are an example.
- The concept of a **virtual machine** provides a way of making code executable on any hardware. The compiler generates code for a virtual machine rather than for a specific processor. E.g. the Java compiler generates code for the Java Virtual Machine (JVM).

8. Why we need openness?

If the well-defined interfaces for a system are published, it is easier for developers to add new features or replace sub-systems in the future. Example: Twitter and Facebook have API that allows developers to develop their own software interactively.

9. How we provide security?

- Concealing the contents of messages: security and privacy
- Identifying a remote user or other agent correctly (authentication)
- New challenges:
- Denial of service attack
- Security of mobile code

10. Define scalability.

Distributed systems must be scalable as the number of user increases.

A system is said to be scalable if it can handle the addition of users and resources without suffering a noticeable loss of performance or increase in administrative complexity

Example: the Internet

11. What are the types of transparencies?

Distributed systems designers must hide the complexity of the systems as much as they can.

Transparency	**Description**
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may be moved to another location while in use
Replication	Hide that a resource may be copied in several places
Concurrency	Hide that a resource may be shared by several competitive users
Failure	Hide the failure and recovery of a resource
Persistence	Hide whether a (software) resource is in memory or a disk

12. Define transparencies.

- How to achieve the single-system image, i.e., how to make a collection of computers appear as a single computer.
- Hiding all the distribution from the users as well as the application programs can be achieved at two levels:
    1. hide the distribution from users
    2. At a lower level, make the system look transparent to programs.

13. Write the consequences of distributed system?

- Requirement for advanced software to realize the potential benefits.
- Security and privacy concerns regarding network communication
- Replication of data and services provides fault tolerance and availability, but at a cost.
- Network reliability, security, heterogeneity, topology
- Latency and bandwidth
- Administrative domains

14. Define Middleware.

The term **middleware** refers to a software layer that provides a programming abstraction as well as masks the heterogeneity of the underlying networks, hardware, operating systems and programming languages.

Thus if an application developer uses middleware (such as RPC) then the developer does not need to know the network protocol details and the Socket API.

Both CORBA and Java RMI are examples of middleware. RMI is Java specific while CORBA is language-neutral.

15. Define Interoperability

The ability of two different systems or applications to work together

16.What are we trying to achieve when we construct a distributed system?

- Certain common characteristics can be used to assess distributed systems
- Heterogeneity
- Openness

- Security
- Scalability
- Failure Handling
- Concurrency
- Transparency

17. List down the Types of Distributed Systems

Distributed Computing Systems

- Clusters
- Grids
- Clouds

Distributed Information Systems

- Transaction Processing Systems
- Enterprise Application Integration

Distributed Embedded Systems

- Home systems
- Health care systems
- Sensor networks

18. What is the difference between Network OS and Distributed OS?

FEATURES	NOS	DOS
multiplicity of machines	Users are aware of multiplicity of machines	Usersare**not**awareof multiplicity of machines
Architecture model	Employs a **client-server** model	Employs a **master-slave** model.
Data access	Run process remotely, and needs to transfer all the data **to the server for processing.**	Run process remotely, rather than transfer all data **locally.**
Process Migration	Execute an entire process, or parts of it, at **the remote server.**	Execute an entire process, or parts of it, at **different sites.**
Computation and Data Migration	Transfer the data, to and from the remote server and only the server performs the all or most of the computation.	Transfer the computation, rather than the data, across the system.
Resource Access	Access to resources of various machines is done explicitly by remote logging into the appropriate remote machine or transferring data from remote machines to local machines, via the File Transfer Protocol (FTP) mechanism	Access to remote resources similar to access to local resources.

19. List down Metrics of the Distributed Computer System
(or)
List out the characteristics of performance of DS?

The following performance characteristics relating to latency, bandwidth and jitter.

a. **Latency:** The delay between the start of a message's transmission from one process and the beginning of its receipt by another is referred to as latency.

b.**Bandwidth:** The bandwidth of a computer network is the total amount of information that can be transmitted over it in a given time. When large number of communication channels are using the same network, they have to share the available bandwidth.

c. **Jitter.** Jitter is the variation in the time taken to deliver a serious of messages. Jitter is relevant to multimedia data. For example, if consecutive samples of audio data are played with differing time intervals, the sound will be badly distorted.

- **Granularity** – relative size of units of processing required. Distributed systems operate best with coarse grain granularity because of the slow communication compared to processing speed in general
- **Processor speed** – MIPS, FLOPS
- **Reliability** – ability to continue operating correctly for a given time
- **Fault tolerance** – resilience to partial system failure
- **Security** – policy to deal with threats to the communication or processing of data in a system
- **Administrative/management** domains – issues concerning the ownership and access to distributed systems components

20. .What are the significance of distributed system?

a. Concurrency of computers.
b. No global clock.
c. In dependent failures.

21. Define the following:

i. Backbone ii. Firewall

A **backbone network** or **network backbone** is a part of computer network infrastructure that interconnects various pieces of network, providing a path for the exchange of information between different LANs or sub networks.

A backbone can tie together diverse networks in the same building, in different buildings in a campus environment, or over wide areas. Normally, the backbone's capacity is greater than the networks connected to it

Firewall

A firewall is a system or group of systems (router, proxy, or gateway) that implements a set of security rules to enforce access control between two networks to protect "inside" network from "outside" network.

22. Give the difference between Synchronous &asynchronous distributed system

Synchronous communication works much like a phone call. The Receiver (callee) must be available, otherwise the conversation cannot occur.

Asynchronous communication works much like the postal system: An application creates a message (that's a piece of data such as the text String "Order 1000 barrels crude oil", or an XML expression), and labels the message with a destination address (that's typically the logical name of a "mail box", and not an IP address). The message is passed to the messaging middleware system. Now the sender application proceeds happily, without needing to wait for the message to be delivered.

23. List down the Trends in Distributed Systems
Distributed systems are undergoing a period of significant change and this can be traced back to a number of influential trends:

- The emergence of pervasive networking technology.
- The emergence of ubiquitous computing coupled with the desire to support user mobility in distributed systems.
- The increasing demand for multimedia services.
- The view of distributed systems as utility.

24. What is resource sharing in distributed systems?

The motivation for distributed systems stems from the need to share resources, both hardware (disks, laser printers etc.), software (programs), and data (files, databases and other data objects).

- Ability to use any hardware, software or data anywhere in the system.
- Resource manager controls access, provides naming scheme and controls concurrency.
- Resource sharing model (e.g. client/server or object-based) describing how resources are provided, they are used and provider and user interact with each other.

25.**List the Limitations of distributed systems.(May-2016)**

- Security problem due to sharing
- Some messages can be lost in the network system
- Bandwidth is another problem if there is large data then all network wires to be replaced which tends to become expensive
- Overloading is another problem in distributed operating systems
- If there is a database connected on local system and many users accessing that database through remote or distributed way then performance become slow
- The databases in network operating is difficult to administrate then single user system

26. **Name some services and examples of Middleware**.**(May-2016)**

Services that can be regarded as middleware

- **enterprise application integration**,
- **data integration**,
- **message oriented middleware** (MOM),
- **object request brokers** (ORBs),
- **enterprise service bus** (ESB).

Examples of Middleware

- Database access technology - e.g ODBC (Open Data Base Connectors)
- Java's database connectivity API : JDBC
- Remote computation products - e.g ONC RPC, OSF RPC and RMI (Java Remote Method Invocation)
- Distributed Computing Environment (DCE) products, Common Object Request Broker Architecture (CORBA), Distributed Component Object Model (DCOM)

27. Write the technological components of web?

* HTML
* HTTP-request-reply protocol
* URL's

PART - B

1.Identify the various challenges in the distributed system in detail**. (AP)**

2.What does transparency mean? What are the different kinds of transparencies in a distributed system? Explain with suitable examples. **(U)**

3.What do you think about Distributed system? What are the pros and cons of Distributed system over centralized system**. (U)**

4. Explain the recent trends in distributed system with suitable real time example. **(An) (May – 2016)**

5.Prepare a neat sketch of resource sharing and explain with suitable example**. (C)**

6. Utilize the World Wide Web as an example to illustrate the concept of resource sharing, client and server. **(U)** (**May – 2016)**

7. Enlighten the examples of distributed system. **(An) (May – 2016)**

COURSE OUTCOME:

- Discuss trends in Distributed Systems
- Demonstrate an understanding of the challenges faced by future distributed systems

UNIT II
COMMUNICATION IN DISTRIBUTED SYSTEM

SYLLABUS: System Model – Inter process Communication - the API for internet protocols – External data representation and Multicast communication. Network virtualization: Overlay networks. Case study: MPI Remote Method Invocation and Objects: Remote Invocation – Introduction - Request-reply protocols - Remote procedure call - Remote method invocation. Case study: Java RMI - Group communication - Publish-subscribe systems - Message queues - Shared memory approaches -Distributed objects - Case study: Enterprise Java Beans -from objects to components.

COURSE OBJECTIVE

- Understand the network virtualization, remote method invocation and objects and RPC.

PART - A

1. **What is meant by inter process Communication? (May-2016)**
   Inter process communication (IPC) is a set of programming interfaces that allow a programmer to coordinate activities among different program processes that can run concurrently in an operating system. This allows a program to handle many user requests at the same time.

2. **What is the difference between RMI and RPC?**

- RPC is language neutral while RMI is limited to Java.
- RPC is procedural like in C, but RMI is object oriented.
- RPC supports only primitive data types while RMI allows objects to be passed as arguments and return values. When using RPC, programmer must split any compound objects to primitive data types.
- RMI is easy to program that RPC.
- RMI is slower than RPC since RMI involves execution of java byte code.
- RMI allows usage of design patterns due to the object oriented nature while RPC does not have this capability.

3. **What is the role of Proxy server and mobile code.(May-2016)**

proxy server is a dedicated computer or a software system running on a computer that acts as an intermediary between an endpoint device, such as a computer, and another **server** from which a user or client is requesting a service.

The term mobile code is used to refer to program code that can be transferred from one computer to another and run at the destination.

Code suitable for running on one computer is not necessarily suitable for running on another because executable programs are normally specific both to the instruction set and to the host operating system. **Example: Java applets.**

4. **Define Datagram.**

The term ‘datagram’ refers to the similarity of this delivery mode to the way in which letters and telegrams are delivered. The essential feature of datagram networks is that the delivery of each packet is a ‘one-shot ‘process; no setup is required, and once the packet is delivered the network retains no information about it.

5. **What is the use of UDP?**

UDP uses a simple connectionless transmission model with a minimum of protocol mechanism.
UDP provides checksums for data integrity,
port numbers for addressing different functions at the source and destination of the datagram.

6. **What are the methods provides by datagram socket?**

- bind(SocketAddress ,addr)
- close()
- connect(InetAddress address, int port)
- disconnect()
- get Broadcast()
- get Port()
- get LocalPort()
- get TrafficClass()

7. **What are the characteristics of network hidden by stream abstraction?**

- Message sizes
- Lost messages
- Flow control
- Message duplication and ordering
- Message destinations.

8. **What is the use of remote object references?**

he RMI (Java Remote Method **Invocation**) system is a mechanism that enables an object on one Java virtual machine to invoke methods on an object in another Java virtual machine. Any object whose methods can be invoked in this way must implement the java.rmi. Remote interface.

9. **What is meant by client server communication?**

A **client-server** network is a central computer, also known as a **server**, which hosts data and other forms of resources. **Clients** such as laptops and desktop computers contact the **server** and request to use data or share its other resources with it.

10. What is meant by group communication?

- group is a collection of processes that act together in some system or user specified way.
- The key property that all groups have that when a message is sent to the group; itself all the members of the group receive it.
- It is a form of one to many communication

10. What is the use of RMI registry?

The RMI registry is used to store a list of available services. A client uses the registry to make it's proxy object, and the Registry is responsible for giving appropriate information to the client so that it can hook up with the server that implements the service.

11. What is meant by distributed garbage collection?

Distributed garbage collection (DGC) in computing is a particular case of **garbage collection** where references to an object can be held by a remote client. DGC uses some combination of the classical **garbage collection** (GC) techniques, tracing and reference counting..

12. Explain the use of reflection in RMI?

- Determine the class of an object.
- Get information about a class's modifiers, fields, methods, constructors, and super classes.
- Find out what constants and method declarations belong to an interface.
- Create an instance of a class whose name is not known until runtime.
- Get and set the value of an object's field, even if the field name is unknown to your program until runtime.
- Invoke a method on an object, even if the method is not known until runtime.
- Create a new array, whose size and component type are not known until runtime, and then modify the array's components.

13. Define overlay Networks.

An **overlay network** is a computer **network** that is built on top of another **network**. Nodes in the **overlay network** can be thought of as being connected by virtual or logical links, each of which corresponds to a path, perhaps through many physical links, in the underlying **network**.

14. Mention the advantages of overlay networks

- In telecommunication
- Support for virtual machine (VM) mobility independent of the physical network.
- Ability to manage overlapping IP addresses between multiple tenants. Support for multi-path forwarding within virtual networks.

Disadvantages

□ Slow in spreading the data.
□ Long latency.
□ Duplicate packets at certain points.

15. Define indirect communication

Indirect communication is defined as communication between entities in a distributed system through an intermediary with no direct coupling between the sender and the receiver(s).

16. Mention the properties of indirect communication

Space uncoupling, in which the sender does not know or need to know the identity of the receiver(s), and vice versa

Time uncoupling, in which the sender and receiver(s) can have independent lifetimes.

17. What are the properties of ordered multicast communication?

FIFO ordering: First-in-first-out (FIFO) ordering- is concerned with preserving the order from the perspective of a sender process

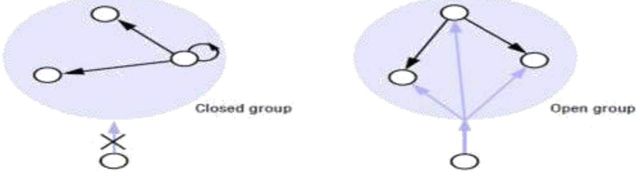
Causal ordering: Causal ordering takes into account causal relationships between messages,

Total ordering: In total ordering, if a message is delivered before another message at one process, then the same order will be preserved at all processes.

18. Define open and closed groups.

A process in a closed group delivers to itself any message that it multicasts to the group.

A group is *open* if processes outside the group may send to it.



19. Mention the tasks of group membership management.

- *Providing an interface for group membership changes*
- *Failure detection*
- *Notifying members of group membership changes*
- *Performing group address expansion*

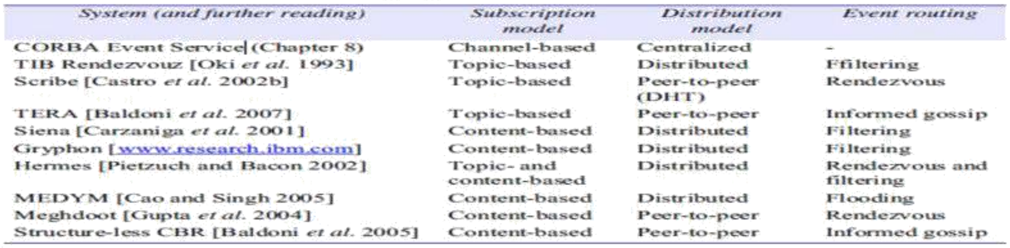
20. Give some Examples of Building blocks.

- Message Dispatcher
- RPC Dispatcher
- Notification Bus

21. Mention the Applications of publish-subscribe systems.

- financial information systems;
- other areas with live feeds of real-time data (including RSS feeds);
- support for cooperative working, where a number of participants need to be
- informed of events of shared interest;
- support for ubiquitous computing, including the management of events emanating
- from the ubiquitous infrastructure (for example, location events);
- a broad set of monitoring applications, including network monitoring in the Internet

22. Give some examples of publish –subscribe systems



System (and further reading)	*Subscription model*	*Distribution model*	*Event routing*
CORBA Event Service (Chapter 8)	Channel-based	Centralized	-
TIB Rendezvouz [Oki *et al.* 1993]	Topic-based	Distributed	Ffiltering
Scribe [Castro *et al.* 2002b]	Topic-based	Peer-to-peer (DHT)	Rendezvous
TERA [Baldoni *et al.* 2007]	Topic-based	Peer-to-peer	Informed gossip
Siena [Carzaniga *et al.* 2001]	Content-based	Distributed	Filtering
Gryphon [www.research.ibm.com]	Content-based	Distributed	Filtering
Hermes [Pietzuch and Bacon 2002]	Topic- and content-based	Distributed	Rendezvous and filtering
MEDYM [Cao and Singh 2005]	Content-based	Distributed	Flooding
Meghdoot [Gupta *et al.* 2004]	Content-based	Peer-to-peer	Rendezvous
Structure-less CBR [Baldoni *et al.* 2005]	Content-based	Peer-to-peer	Informed gossip

23. Define Message Queues

Message queues are also referred to as Message-Oriented Middleware. This is a major class of commercial middleware with key implementations including IBM’s Web Sphere MQ, Microsoft’s MSMQ and Oracle’s Streams Advanced Queuing (AQ).

24. Define the role of JMS

- A JMS *Java Messaging Service* client is a Java program or component that produces or consumes messages, a JMS producer is a program that creates and produces messages and a JMS consumer is a program that receives and consumes messages.
- A JMS provider is any of the multiple systems that implement the JMS specification.
- A JMS message is an object that is used to communicate information between
- JMS clients (from producers to consumers).

25. Outline an implementation of a publish/subscribe system based on a message-queuing system like that of IBM MQ Series.

Such an implementation can be accomplished by using a message broker. All publish/subscribe messages are published to the broker. Subscribers pass their subscriptions to the broker as well, which will then take care to forward publish messages to the appropriate subscribers. Note that the broker makes use of the underlying message-queuing network.

26. What is a distributed shared memory?

- The distributed shared memory (DSM) implements the shared memory model in distributed systems, which have no physical shared memory
- The shared memory model provides a virtual address space shared between all nodes
- The overcome the high cost of communication in distributed systems, DSM systems move data to the location of access

27. What is a tuple space?

tuple space is an implementation of the associative memory paradigm for parallel/distributed computing. It provides a repository of **tuples** that can be accessed concurrently.

28. What is an IDL?

An interface description language or **interface definition language** (**IDL**), is a specification language used to describe a software component's application programming interface (API)..

29. Define Physical model.

Physical models are the most explicit way to describe a system; they capture the hardware composition of a system in terms of the computers (and other devices, such as mobile phones) and their interconnecting networks.

30. Define Architectural model.

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

31. Define Fundamental model.

Fundamental models take an abstract perspective in order to examine individual aspects of a distributed system.

Fundamental models examine three important aspects of distributed systems:

- interaction models - which consider the structure and sequencing of the communication between the elements of the system
- failure models - which consider the ways in which a system may fail to operate correctly
- Security models - which consider how the system is protected against attempts to interfere with its correct operation or to steal its data.

32. What are the difficulties faced by the designer of distributed system?

- Widely varying modes of use.
- Wide range of system environments.
- Internal problems.
- External threats.

33. Define Request reply protocols.

- The client *sends* are quest message
- the server *executes* the requested operation
- the server *responds* with a reply message
- Most widely used request/reply communication model is Remote Procedure Call (RPC)

34. Define Publish-subscribe systems.

Publish-subscribe systems share the crucial feature of providing an intermediary service that efficiently ensures information generated by producers (or publishers) is routed to consumers (or subscribers) who desire this information. Publish-subscribe systems offer a one-to-many style of communication.

35. Define Message Queues.

Message queues offer a point-to-point service whereby producer processes can send messages to a specified queue and consumer processes can receive messages from the queue or be notified of the arrival of new messages in the queue. Queues therefore offer an indirection between the producer and consumer processes.

36. What do you mean by Tuple Space?

A **tuple space** is an implementation of the associative memory paradigm for parallel/distributed computing. It provides a repository of **tuples** that **can** be accessed concurrently. ... Producers post their data as **tuples** in the **space**, and the consumers then retrieve data from the **space** that match a certain pattern.

37. **<u>Differentiate marshaling and un marshalling.</u>**

Marshalling	**Unmarshalling**
process of taking a collection of data items and assembling them into a form suitable for	process of disassembling them on arrival to produce an equivalent collection of data items

transmission in a message	at the destination
consists of the translation of structured data items and primitive values into an external data representation	consists of the generation of primitive values from their external data representation and the rebuilding of the data structures

38. What is Remote Procedure Call?

Remote Procedure Call (**RPC**) is a protocol that one program can use to request a service from a program located in another computer on a network without having to understand the network's details. A **procedure call** is also sometimes known as a function **call** or a subroutine **call**..

39. What is Remote Method Invocation?

The Java **Remote Method Invocation** (**RMI**) system allows an object running in one Java virtual machine to invoke **methods** on an object running in another Java virtual machine. **RMI** provides for **remote** communication between programs written in the Java programming language.

40.What is the difference between RMI and RPC?

RPC	**RMI**
RPC is a mechanism that enables calling of a procedure on a remote computer.	RMI is the implementation of RPC in java.
RPC is language neutral but only supports primitive data types to be passed.	RMI is limited to Java but allows passing objects.
RPC follows traditional procedural language constructs.	RMI supports object-oriented design.

PART B

1. Illustrate with necessary diagrams explain in detail about Fundamental Model, Architectural Model. (**Ap**)
2. Discuss about the characteristics of inter process communication. (**U**)
3. Create API for internet protocols and its addressing with suitable diagrams. **(C)**
4. Create API for internet protocols and its addressing with suitable diagrams. **(C)**
5. Describe the Communication between distributed objects. (**U**)
6. Describe about the client server communication. (**U**)
7. Describe in detail about Events and Notifications. (**R**)
8. Explain in detail about Remote Procedure call with a case study. **(An)**
9. Describe java RMI and explain it with suitable program**. (R) (May – 2016)**
10. Describe group communication with neat sketch**. (R)**
11. Explain TCP and UDP datagram communication with suitable examples. **(U) (May – 2016)**

COURSE OUTCOME:

- Able to learn and apply the concept of network virtualization and remote method invocation
- Apply network virtualization in Real time systems like **skype**.

UNIT III
PEER TO PEER SERVICES AND FILE SYSTEM

SYLLABUS : Peer-to-peer Systems – Introduction - Napster and its legacy - Peer-to-peer – Middleware - Routing overlays. Overlay case studies: Pastry, Tapestry- Distributed File Systems – Introduction - File service architecture – Andrew File system. File System: Features-File model -File accessing models - File sharing semantics naming: Identifiers, Addresses, Name Resolution – Name Space Implementation – Name Caches – LDAP.

COURSE OBJECTIVE: Introduce the idea of peer to peer services and file system

PART A

1. What is peer to peer system?

A **peer-to-peer** (**P2P**) **network** is created when two or more PCs are connected and share resources without going through a separate server computer. A **P2P network** can be an ad hoc connection—a couple of computers connected via a Universal Serial Bus to transfer files..

2. What are Peer-to-peer middleware?

Peer-to-peer middleware systems are emerging that have the capacity to share computing resources, storage and data present in computers 'at the edges of the Internet' on a global scale. They exploit existing naming, routing, data replication and security techniques in new ways to build a reliable resource-sharing layer over an unreliable and untrusted collection of computers and networks.

3.What are characteristics of Peer-to-peer systems?(May2016)

Peer-to-peer systems share these characteristics:

- Their design ensures that each user contributes resources to the system.
- Although they may differ in the resources that they contribute, all the nodes in peer-to-peer system have the same functional capabilities and responsibilities.
- Their correct operation does not depend on the existence of any centrally administered systems.
- They can be designed to offer a limited degree of anonymity to the providers and users of resources.
- A key issue for their efficient operation is the choice of an algorithm for the placement of data across many hosts and subsequent access to it in a manner that balances the workload and ensures availability without adding undue overheads.

4. What are three generations of Peer-to-peer systems?

Three generations of peer-to-peer systems:

1. Napster music exchange | relied in part on a central server
2. File sharing systems | with greater fault tolerance and no reliance on a central server, examples include: Gnutella, Direct Connect, Kazaa, Emule, Bittorrent, FreeNet
3. The emergence of middleware layers for peer-to-peer systems making possible the application independent provision of resources.

5. Discuss about Napster.

- Napster was an early offering in peer-to-peer style systems
- Offering the ability for users to share data files quickly became popular with those sharing music files

- However Napster was shut down as a result of:
- People sharing copyrighted music
- This lead the owners of the copyrighted material to instigate legal proceedings against the Napster service operators
- This in turn caused the Napster service to be shut down

6. Write down the steps executed in Napster when a user request for a particular file.

When a user had a request for a particular file the following steps where executed:

1. A file location request is made by a user to the centrally managed Napster index
2. The Napster server responds to the request with a list of peers who have the requested file available
3. The user then requests that file from one of the list of peers
4. The peer from whom the file is requested then delivers the file directly to the requesting user, without central server intervention
5. Finally, once the requested file is received by the user it informs the centrally managed Napster server such that the index of files may be updated. That is, the requesting user now has the particular file

7. What are Functional requirements of peer to peer middleware?

- Functional requirements:
  - Enable clients to locate and communicate with any individual resource
  - Add and remove nodes
  - Add and remove resources
  - Simple API

8. List out the Non-Functional requirements of peer to peer middleware?

- global scalability,
- load balancing,
- accommodating to highly dynamic host availability
- , trust,
- anonymity,
- deniability…

9. Write about Routing Overlays

- A distributed algorithm known as routing overlay.
- Routing overlay locates nodes and objects. It is middleware layer responsible for routing requests from clients to hosts that holds the object to which request is addressed.
- Main difference is that routing is implemented is in application layer (besides the IP routing at network layer)

10. What are the main tasks of the Routing Overlay?

Routing of Requests to Objects: A client wishing to perform some act upon a particular object must send that that request, with the GUID attached, through the routing overlay

Insertion of Objects: A node wishing to insert a new object, must compute a new GUID for that object and announce it to that routing overlay such that that object is available to all nodes

Deletion of Objects: When an object is deleted the routing overlay must make it unavailable for other clients

Node addition and removal: Nodes may join and leave the service at will. The routing overlay must organize for new nodes to take over some of the responsibilities of other (hopefully nearby) nodes. When a node leaves, the routing overlay must distribute its responsibility to remaining nodes

11. List out the operations involved in distributed Hash tables.

A distributed hash table has three operations:
1. put (GUID; data): stores data at all nodes responsible for the object identified by GUID
2. Remove (GUID): deletes all references to GUID and the associated data
3. Get (GUID): retrieves the data associated with GUID from one of the nodes responsible for it.

12. Mention two case studies for routing overlay.

Pastry, Tapestry

13. Compare structured and unstructured peer to peer system

Structured peer to peer system	**Unstructured peer to peer system**
Advantages 1. Guaranteed to locate (existing) objects 2. Relatively low message overhead	Advantages 1. Self-organizing and naturally resilient to node failure 2. Different versions of software can often interoperate with little engineering effort
Disadvantages 1. Need to maintain complex overlay structures 2. Slow to adapt to highly dynamic networks 3. Software is dicult to upgrade if it updates the distributed data structures used	Disadvantages 1. Offers no guarantees on locating objects even if they exist 2. Can generate large amounts of messaging overhead

14. What is Distributed File System (DFS)?

Distributed File System (DFS) allows administrators to group shared folders located on different servers and present them to users as a virtual tree of folders known as a namespace. Namespace provides numerous benefits, including increased availability of data, load sharing, and simplified data migration.

15. What are file system modules?

Directory module: relates file names to file IDs

File module: relates file IDs to particular files

Access control module: checks permission for operation requested

File access module: reads or writes file data or attributes

Block module: accesses and allocates disk blocks

Device module: performs disk I/O and buffering

16. List out types of storage system.

Main memory - RAM

File system -UNIX file system

Distributed file system- Sun

NFS Web - Web server

Distributed shared memory - Ivy

Remote objects (RMI/ORB) - CORBA
Persistent object store - CORBA Persistent State Service
Peer-to-peer storage system –Ocean Store

17. What does the term "Meta data" refer?

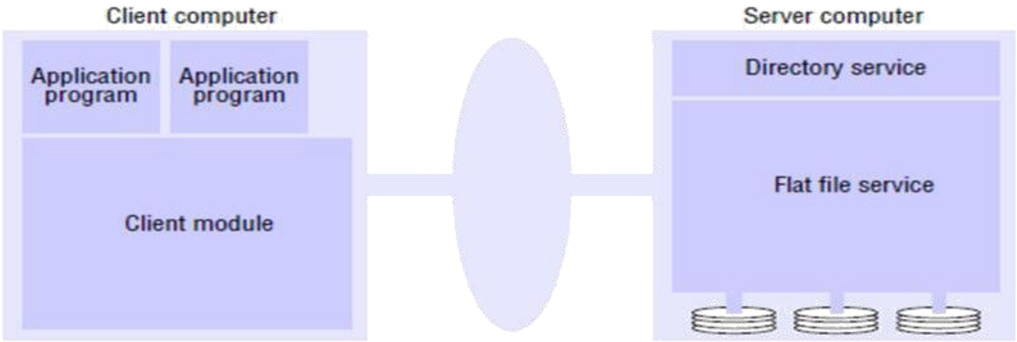
The term metadata is often used to refer to all of the extra information stored by a file system that is needed for the management of files. It includes file attributes, directories and all the other persistent information used by the file system.

18. What are the requirements of DFS?

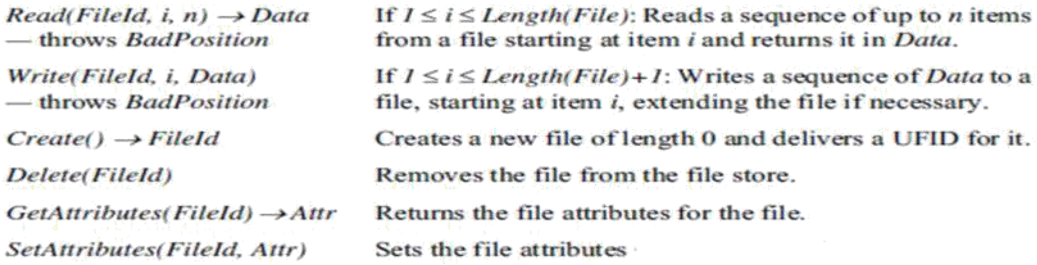
Transparency
Concurrent file updates
File replication
Hardware and operating system heterogeneity
Fault tolerance
Consistency
Security
Efficiency

19. What is file service architecture?

An architecture that offers a clear separation of the main concerns in providing access to files is obtained by structuring the file service as three components – a flat file service, a directory service and a client module.

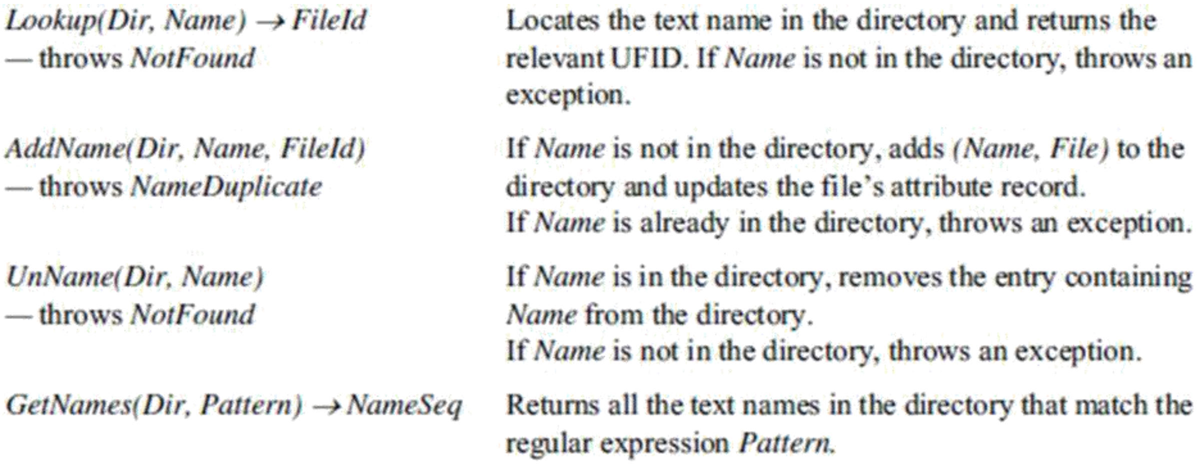


20. What are flat file service operations?



Read(FileId, i, n) → Data — throws *BadPosition*	If $1 \leq i \leq Length(File)$: Reads a sequence of up to *n* items from a file starting at item *i* and returns it in *Data*.
Write(FileId, i, Data) — throws *BadPosition*	If $1 \leq i \leq Length(File)+1$: Writes a sequence of *Data* to a file, starting at item *i*, extending the file if necessary.
Create() → FileId	Creates a new file of length 0 and delivers a UFID for it.
Delete(FileId)	Removes the file from the file store.
GetAttributes(FileId) → Attr	Returns the file attributes for the file.
SetAttributes(FileId, Attr)	Sets the file attributes

21. What are Directory service operations?



Lookup(Dir, Name) → FileId — throws *NotFound*	Locates the text name in the directory and returns the relevant UFID. If *Name* is not in the directory, throws an exception.
AddName(Dir, Name, FileId) — throws *NameDuplicate*	If *Name* is not in the directory, adds *(Name, File)* to the directory and updates the file's attribute record. If *Name* is already in the directory, throws an exception.
UnName(Dir, Name) — throws *NotFound*	If *Name* is in the directory, removes the entry containing *Name* from the directory. If *Name* is not in the directory, throws an exception.
GetNames(Dir, Pattern) → NameSeq	Returns all the text names in the directory that match the regular expression *Pattern*.

22. Write about Andrew File System.

AFS provides transparent access to remote shared files for UNIX programs running on workstations. Access to AFS files is via the normal UNIX file primitives ,enabling existing UNIX programs to access AFS files without modification or recompilation.

23. Mention two unusual design characteristics of AFS.

Whole-file serving: The entire contents of directories and files are transmitted to client computers by AFS servers (in AFS-3, files larger than 64kbytes are transferred in 64-kbyte chunks).

Whole-file caching: Once a copy of a file or a chunk has been transferred to a client computer it is stored in a cache on the local disk. The cache contains several hundred of the files most recently used on that computer. The cache is permanent, surviving reboots of the client computer. Local copies of files are used to satisfy clients' open requests in preference to remote copies whenever possible.

24. What is name space?

A name space is the collection of all valid names recognized by a particular service.

25. What is Naming domain?

A naming domain is a name space for which there exists a single overall administrative authority responsible for assigning names within it.

26. What are Naming Services?

In a Distributed System, a Naming Service is a specific service whose aim is to provide a consistent and uniform naming of resources, thus allowing other programs or services to localize them and obtain the required metadata for interacting with them**.**

27. Discuss about LDAP(May- 2016)

LDAP (Lightweight Directory Access Protocol) is a software protocol for enabling anyone to locate organizations, individuals, and other resources such as files and devices in a network, whether on the public Internet or on a corporate intranet.

PART B

1. Distinguish between IP and overlay routing for peer-to-peer applications**. (An)**
2. Explain in detail about Napster and its legacy. May – 2016) (**U**)
3. Discuss in detail about Peer to peer middleware. (May – 2016). (**An**)
4. With neat sketch explain Routing overlay in detail. **(Ap**). (May – 2016)
5. Write in detail about i. Pastry .(**U**) ii. Tapestry.(**U**)
6. Describe file service architecture**. (U)** (May – 2016)
7. What judgement would you make about Sun network File system and Andrew File system**. (E)**
8. Explain about name services and the domain name system. (**U)**
9. What conclusion can you draw about directory services. (**An)**
10. Write about the global name service. **(R)**

COURSE OUTCOME:

- Able to analyze the mechanism of peer to peer systems, DFS and DNS.

UNIT IV
SYNCHRONIZATION AND REPLICATION

SYLLABUS: Introduction - Clocks, events and process states - Synchronizing physical clocks- Logical time and logical clocks - Global states – Coordination and Agreement – Introduction - Distributed mutual exclusion – Elections – Transactions and Concurrency Control– Transactions - Nested transactions – Locks – Optimistic concurrency control - Timestamp ordering – Atomic Commit protocols -Distributed deadlocks – Replication – Case study – Coda.

COURSE OBJECTIVE: Methods of understanding clock synchronization protocols &replication

PART - A

1. What is transparency?

Distributed systems should be perceived by users and application developers as a whole rather than as a collection of cooperating components. Transparency hides the distributed nature of the system from its users and shows the user that the system is appearing and performing as a normal centralized system.

2. What are the different types of transparency?

- Network Transparency
- Name Transparency
- Location Transparency
- Semantic Consistency
- Access Transparency
- Execution Transparency
- Replication Transparency
- Performance Transparency
- Configuration Transparency

3. Define cut, consistent cut and inconsistent cut.

- **Cut is a subset of the global history that contains an initial prefix of each local state**

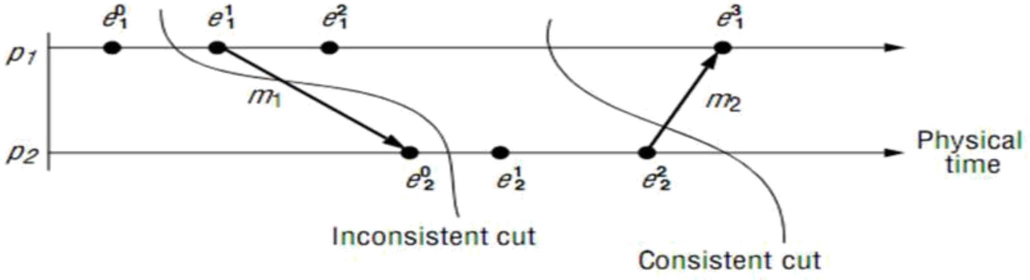
A cut of the system's execution is a subset of its global history that is a union of prefixes of process histories:

$$C = h_1^{c_1} \cup h_2^{c_2} \cup \ldots \cup h_N^{c_N}$$

A cut C is consistent if, for each event it contains, it also contains all the events that happened-before that event:

For all events $e \in C$, $f \rightarrow e \Rightarrow f \in C$

Consider the events occurring at processes p1 and p2 shown in Figure. The figure shows two cuts, one with frontier $<e_1^0, e_2^0>$ and another with frontier $<e_1^2, e_2^2>$. The leftmost cut is inconsistent. This is because at p2 it includes the receipt of the message m1, but at p1 it does not include the sending of that message.



4. What is a clock?

- A clock C_i for each process $P_i$to be a function which assigns a number $C_i(a)$ *to any* event a in that process.
- Logical Clock (C_i)has no relation with the Physical Clock.

5.What is the condition for a clock?

For any event a, b:

If a→b, then C (a) <C(b)

Clock Condition is satisfied if:

C1. If *a* and *b* are events in process P_i, and a comes before *b*, then $C_i(a) < C_i(b)$.

C2. If *a* is the sending of a message by process P_i and *b* is the receipt of that message by process P_j, then

$C_i(a) < C_j(b)$

6. Define logical clock.

In distributed systems, it is not possible to have a global physical time. So a logical clock allows global ordering on events from different processes in such systems.

In logical clock systems each process has two data structures: *logical local time* and *logical global time*. Logical local time is used by the process to mark its own events, and logical global time is the local information about global time. A special protocol is used to update logical local time after each local event and logical global time when processes exchange data.

7. Define clock skew.

Computer clocks tend not to be in perfect agreement .The instantaneous difference between the readings of any two clocks is called clock skew.

8. Define clock drift.

The crystal-based clocks used in computers are subject to clock drift, which means that they count time at different rates, and so diverge.

9. Define External synchronization.

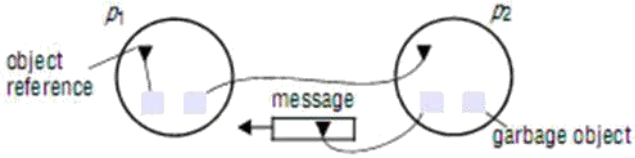
In order to know at what time of day events occur at the processes in distributed system „– it is necessary to synchronize the processes' clocks, Ci, with an authoritative, external source of time. This is external synchronization.

10. Define internal synchronization.

If the clocks Ci are synchronized with one another to a known degree of accuracy, then we can measure the interval between two events occurring at different computers by appealing to their local clocks, even though they are not necessarily synchronized to an external source of time. This is internal synchronization.

11. When an object is considered to be a garbage?

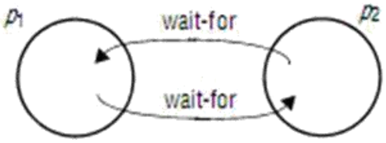
An object is considered to be garbage if there are no longer any references to it anywhere in the distributed system. The memory taken up by that object can be reclaimed once it is known to be garbage.



12. What is distributed dead lock ?

A distributed deadlock occurs when each of a collection of processes waits for another process to send it a message, and where there is a cycle in the graph of this 'waits-for' relationship.

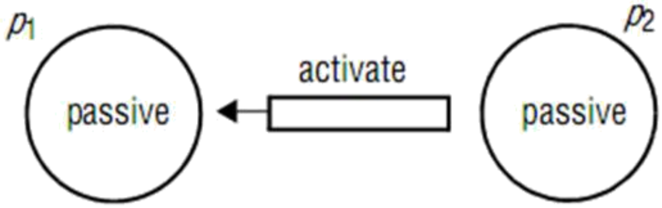
Figure shows that processes p1 and p2 are each waiting for a message from the other, so this system will never make progress



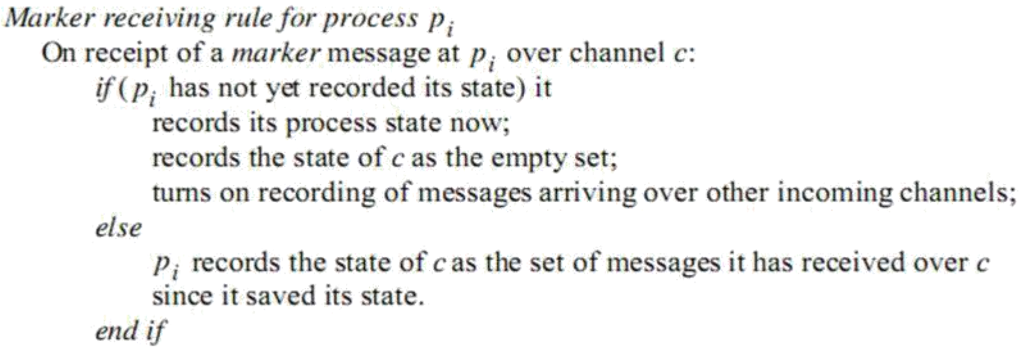
13. What do you meant by Distributed termination detection?

A distributed algorithm is considered to be executed by two processes p1 and p2, each of which may request values from the other. Instantaneously, a process is either active or passive – a passive process is not engaged in any activity of its own but is prepared to respond with a value requested by the other. Suppose process p1 is passive and that p2 is passive, it is not possible to conclude that the algorithm has terminated.

Consider the following scenario: when p1 is tested for passivity, a message was on its way from p2, which became passive immediately after sending it. On receipt of the message, p1 became active again – after p1 found to be passive. The algorithm had not terminated.

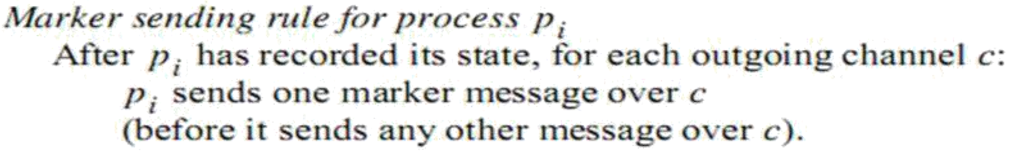


14. Define marker receiving rule.



```
Marker receiving rule for process $p_i$
    On receipt of a marker message at $p_i$ over channel $c$:
        if ($p_i$ has not yet recorded its state) it
            records its process state now;
            records the state of $c$ as the empty set;
            turns on recording of messages arriving over other incoming channels;
        else
            $p_i$ records the state of $c$ as the set of messages it has received over $c$
            since it saved its state.
        end if
```

15. Define marker sending rule.



```
Marker sending rule for process $p_i$
    After $p_i$ has recorded its state, for each outgoing channel $c$:
        $p_i$ sends one marker message over $c$
        (before it sends any other message over $c$).
```

16. What are the approaches to implement DSM?

The approaches to implement DSM are

- Hardware
- Paged virtual memory
- Middleware

17. List the options to update DSM consistency.

- Write update
- Write invalidate

18. Define Transaction.

Transaction is a unit of program execution that accesses and possibly updated various data items.

19. What are the properties of transaction?(ACID properties)

The properties o f transactions are:

- Atomicity
- Consistency
- Isolation
- Durability

21. Define concurrency control in distributed transactions.

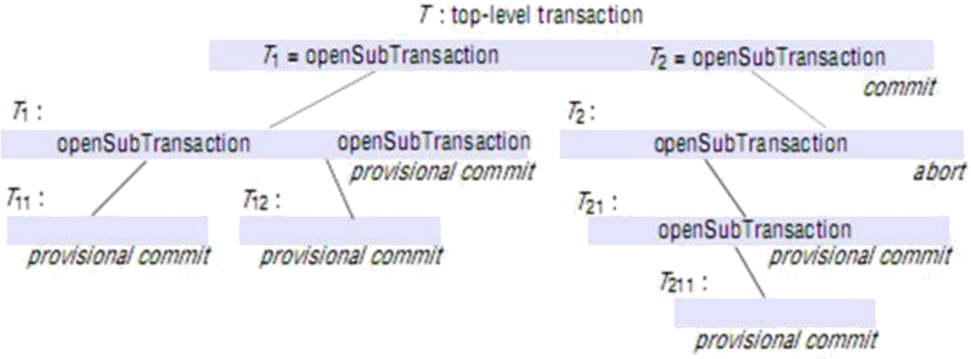
Distributed concurrency control is the concurrency control of a system distributed over a computer network

concurrency control ensures that correct results for concurrent operations are generated, while getting those results as quickly as possible.

22. What is meant by nested transaction?

Several transactions may be started from within a transaction, allowing transactions to be regarded as modules that can be composed as required.

The outermost transaction in a set of nested transactions is called the top-level transaction. Transactions other than the top-level transaction are called sub transactions. For example, in Figure, T is a top-level transaction that starts a pair of sub transactions, T1 and T2. The sub transaction T1 starts its own pair of sub transactions, T11 and T22. Also, sub transaction T2 starts its own sub transaction, T21, which starts another sub transaction, T211.



23. Define strict two phase locking.

A transaction that needs to read or write an object must be delayed until other transactions that wrote the same object have committed or aborted. To enforce this rule, any locks applied during the progress of a transaction are held until the transaction commits or aborts. This is called strict two-phase locking. The presence of the locks prevents other transactions reading or writing the objects.

24. Define distributed deadlock.

In a distributed system involving multiple servers being accessed by multiple transactions, a global wait-for graph can be constructed from the local ones. There can be a cycle in the global wait-for graph that is not in any single local one –there can be a distributed deadlock. There is a deadlock if and only if there is a cycle in the wait-for graph.

25. Write about validation phase and update phase.

Validation phase: When the close Transaction request is received, the transaction is validated to establish whether or not its operations on objects conflict with operations of other transactions on the same objects. If the validation is successful, then the transaction can commit. If the validation fails, then some form of conflict resolution must be used and either the current transaction or, in some cases, those with which it conflicts will need to be aborted.

Update phase: If a transaction is validated, all of the changes recorded in its tentative versions are made permanent. Read-only transactions can commit immediately after passing validation. Write transactions are ready to commit once the tentative versions of the objects have been recorded in permanent storage.

26. Define time stamp ordering.

Each transaction is assigned a unique timestamp value when it starts. The timestamp defines its position in the time sequence of transactions. Requests from transactions can be totally ordered according to their timestamps.

27. What is two phase commit protocol?

The **two phase commit protocol** is a distributed algorithm which lets all sites in a distributed system agree to **commit** a **transaction**. The **protocol** results in either all nodes **committing** the **transaction** or aborting, even in the case of site failures and message losses..

28. Define Edge chasing.

A distributed approach to deadlock detection uses a technique called edge chasing or path pushing. In this approach, the global wait-for graph is not constructed, but each of the servers involved has knowledge about some of its edges.

The servers attempt to find cycles by forwarding messages called probes, which follow the edges of the graph throughout the distributed system. A probe message consists of transaction wait-for relationships representing a path in the global wait-for graph.

29. Write the definition for sequential consistency.

A replicated shared object service is said to be sequentially consistent if for any execution there is some interleaving of the series of operations issued by all the clients that satisfies the following two criteria:

• The interleaved sequence of operations meets the specification of a (single)
correct copy of the objects.

• The order of operations in the interleaving is consistent with the program order
in which each individual client executed them.

30. Define Latency.

Latency is the time required for an individual data element to move through a stream from the source to the destination. This may vary depending on the volume of other data in the system and other characteristics of the system load. This variation is termed jitter – formally, jitter is the first derivative of the latency.

31. Define Mobile IP.

Mobile IP (or MIP) is an Internet Engineering Task Force (IETF) standard communications protocol that is designed to allow **mobile** device users to move from one network to another while maintaining a permanent **IP** address

32. Define CORBA.

The **Common Object Request Broker Architecture** (**CORBA**) is a standard **defined** by the Object Management Group (OMG) designed to facilitate the communication of systems that are deployed on diverse platforms.

33.**Distinguish between physical clock and Logical clocks(May - 2016)**

A **physical clock** is a **physical** process coupled with a method of measuring that process to record the passage of time. For instance, the rotation of the Earth measured in solar days is a **physical clock**. Most **physical clocks** are based on cyclic processes (such as a celestial rotation).

Logical clocks

Logical clocks are useful in computation analysis, distributed algorithm design, individual event tracking, and exploring computational progress. Some noteworthy logical clock algorithms are:

- Lamport times tamps, which are monotonically increasing software counters.
- Vector clocks, that allow for partial ordering of events in a distributed system.
- Version vectors, order replicas, according to updates, in an optimistic replicated system.
- Matrix clocks, an extension of vector clocks that also contains information about other processes' views of the system.

34.**Define happened before relation (May - 2016)**

happened-before relation (denoted: → {\display style \to \;}) is a **relation** between the result of two events, such that if one event should **happen before** another event, the result must reflect that, even if those events are in reality executed out of order (usually to optimize program flow).

35.**Define Mutual Exclusion.(MAY/JUNE2012)**

Each process accessing the shared data excludes all others from doing simultaneously called as Mutual Exclusion.

PART - B

1. Explain about clocks, events and process state. Explain snapshot algorithm. **(An)**
2. Explain about synchronization of physical clock. (**An**)
3. Discuss in brief about Distributed mutual exclusion. **(U)**
4. Describe briefly about network time protocol. **(U)**
5. Demonstrate Election algorithm with example. (**Ap)**
6. What is atomic commit protocol? Explain the different types of protocol in atomic commit protocol**. (R)**
7. Describe about nested transaction. **(R)**
8. Generalize the concept of locks with suitable example. **(C)**
9. Define optimistic concurrency control. Explain in detail. (**U)** (**May – 2016**)
10. Explain in detail about time stamp ordering. **(E)**
11. Describe about Maekawa’s Voting Algorithm. (**U)** (**May – 2016**)
12. Discuss on Nested Transactions. **(U)** (May – 2016**)**
13. Describe in detail about distributed deadlock. **(U)**

COURSE OUTCOME:

- Able to understand key mechanisms and models for distributed systems including logical clocks, causality, distributed mutual exclusion, distributed deadlocks..

UNIT V
PROCESS & RESOURCE MANAGEMENT

SYLLABUS: Process Management: Process Migration: Features, Mechanism - Threads: Models, Issues, Implementation. Resource Management: Introduction- Features of Scheduling Algorithms – Task Assignment Approach – Load Balancing Approach – Load Sharing Approach.

COURSE OBJECTIVE: Understand the issues involved in studying process and resource management

PART - A

1.How to create a new process in thread?

Process:

- Process (slightly oversimplified): "An execution stream in the context of a particular process state."
    - Execution stream: a sequence of instructions (only one thing happens at a time).
    - Process state: everything that can affect, or be affected by, the process: code, data, call stack, open files, network connections, etc.

The Process of assignment of the CPU to the first process on the ready list is called as Dispatching.

Threads

- Most modern operating systems also support threads: multiple execution streams within a single process
    - Threads share process state such as memory, open files, etc.
    - Each thread has a separate stack for procedure calls (in shared memory)
    - Thread is unit of sequential execution

2. What are the different operations that can be performed on a process?

- Create a process
- Destroy a process
- Change a process's priority
- Wakeup a process
- Enable a process to communicate with others
- Suspend a process
- Resume a process
- Block a process

3. What is meant by Context Switching?

A **context switch** (also sometimes referred to as a process **switch** or a task **switch**) is the **switching** of the CPU (central processing unit) from one process or thread to another. A process (also sometimes referred to as a task) is an executing (i.e., running) instance of a program

4.What are benefits of Multiprogramming?

- Responsiveness
- Resource Sharing
- Economy
- Utilization of multiprocessor architectures.

5. Define process management

Process management is the ensemble of activities of planning and monitoring the performance of a business **process**. The term usually refers to the **management** of business **processes** and manufacturing **processes**. Business **process management** (BPM) and business **process** reengineering are interrelated, but not identical.

6. What is process migration?

- **Process migration** is a specialized form of process management whereby processes are moved from one computing environment to another.
- This originated in distributed computing, but is now used more widely.
- Process migration happens as a standard part of process scheduling, and it is quite easy to migrate a process within a given machine, since most resources (memory, files, sockets) do not need to be changed, only the execution context

5. List out the advantages of process migration.

- Reducing average response time of processes
- Speeding up individual jobs
- Gaining higher throughput
- Utilizing resources effectively
- Reducing network traffic
- Improving system reliability

7. What are types of process migration?

There are two types of process migration

- Non-preemptive process migration
- Preemptive process migration

non- preemptive process migration

Process migration that takes place before execution of the process starts. This type of process migration is relatively cheap, since relatively little administrative overhead is involved. **preemptive process migration**

Process migration whereby a process is preempted, migrated and continues processing in a different execution environment. This type of process migration is relatively expensive, since it involves recording, migration and recreation of the process state as well as the reconstructing of any inter-process communication channels to which the migrating process is connected.

6. Define threads

- A Thread is a piece of code that runs in concurrent with other threads.
- Each thread is a statically ordered sequence of instructions.
- Threads are used to express concurrency on both single and multiprocessors machines.

7. Difference between processes and threads

Processes	threads
A process consists of an execution environment together with one or more threads.	A thread is the operating system abstraction of an activity
Heavyweight processes	Lightweight processes
Processes are expensive to create	Threads are easier to create and destroy

8. What are the models of threads?

- Dispatcher-workers model
- Team model
- Pipeline model

9. What are the advantages of using threads?

- Useful for clients
- Handle signals, such as interrupts from the keyboard
- Producer-consumer problems are easier to implement using threads
- It is possible for threads in a single address space to run in parallel, on different CPUs.

10. List out the thread packages.

There are two thread packages

Static thread

- The choice of how many threads there will be is made when the program is written or when it is compiled. Each thread is allocated a fixed stack. This approach is simple, but inflexible

Dynamic thread

It allows threads to be created and destroyed on-the-fly during execution

11. What are issues in designing a thread package?

- Threads creation
- Thread termination
- Threads synchronization
- Threads scheduling

12. Define resource management

Resource manager are to control the assignment of resources to processes. Resources can be logical (shared file) or physical (CPU).

13. What are the types of resource management?

There are three types

- Task assignment approach
- Load-balancing approach
- Load-sharing approach

14. What is task assignment approach?

Finding an optimal assignment to achieve goals such as the following:

- Minimization of IPC costs
- Quick turnaround time of process
- High degree of parallelism
- Efficient utilization of resources

15. What is load balancing approach?

Load balancing in grid is a technique which distributes the workloads across multiple computing nodes to get optimal resource utilization, minimum time delay, maximize throughput and avoid overload.

16. What is load sharing approach?

Load-sharing is much simpler than load-balancing since it only attempts to ensure that no node is idle when heavily node exists.

17. What are the different process states available?

- Running, if it currently has the CPU
- Ready, if it could use a CPU if one were available
- Blocked, if it is waiting for some event to happen before it can proceed

18. Define Process.

Process is defined as

- Program in execution
- A synchronous activity.
- The "animated spirit" of a procedure
- The "locus of control of a procedure in execution which is manifested by the existence of a "process control block" in the operating system
- That entity to which processors are assigned the dispatch able unit

19.**Write any two advantages of Process Migration(May - 2016)**

- Load sharing
- Communications performance
- Availability
- Utilizing special capabilities

20. List the issues in designing load balancing algorithms(May - 2016)

- **Load estimation policy**
- **Process transfer policy**
- **Location policy**
    - *Threshold method*
    - *Bidding method*
    - *Pairing*
    - **State information exchange policy**

PART B

1. Explain Process and threads with neat diagram. **(E)**

2. Describe in detail about Process migration. (**R)**

3. Show the various mechanisms available for thread concepts and how it is applied. Explain with example. (**Ap)**

4. Describe various thread models with an example. **(R)**

5. Describe about managing resource in resource management. (**U)**

6. Compare and classify various scheduling algorithms with an example. (**An)**

7. What is task assignment? Explain the approaches we used to assign it. **(R) (May – 2016)**

8. Explain in detail about load balancing algorithm with an example. (**U)** (**May – 2016)**

9. Explain in load sharing scheduling algorithm with an example. (**U)** (**May – 2016)**

10. Illustrate the features and mechanism of Process migration with suitable examples.(**Ap)** (**May – 2016)**

COURSE OUTCOME: Students are able to Design process and resource management systems.

.