ECE 677: *Distributed Computing Systems*

Salim Hariri

High Performance Distributed Computing Laboratory

University of Arizona

Tele: (520) 621-4378

Nsfcac.arizona.edu

Fall 2013

Outline

- Motivation and objectives
- Distributed System Characteristics
- Distributed System Evolutions
- Distributed System Design Framework

- **Office :** Room ECE 459, Tel: 621-4378
- **Office Hours:** By Appointment, questions via email are encouraged.
- **Prerequisites:** ECE 578, CS 552.
- Grading: Class Participation 10%, Homework 20%, Project 35%, Term Paper 35%
- **Recommended Readings: Research Papers, text books on distributed computing systems, IEEE Tutorials, and Conference Proceedings.**

- It is important to check the class website
  - Homework assignments, announcements, etc. will be posted at class website
- **Recommended Textbooks/References:**
  - High Performance Distributed Systems: Network, Architecture, and Programming, *in preparation*,
  - **Tools and Environments for Parallel and Distributed Computing,** Salim Hariri and Manish Parashar, John Wiley & Sons, Inc., ***2004***.
  - Cluster Computing: The Journal of networks, software tools and applications, Editor-in-chief, Salim Hariri, Springler
  - Journals and Conference Proceedings (Symposium on High Performance Distributed Computing, Cluster Computing Journal, IEEE Transactions on Parallel and Distributed Systems, Journal of Parallel and Distributed Computing)

Term Paper : Every Student Must Submit a term paper that addresses important issues of designing distributed computing systems.

Programming Project : Every student or a group (typically two) is required to develop a project that applies distributed computing to solve computationally intensive applications across a cluster of workstations. The project should be developed using any message passing tool such as MPI, JAVA, Globus, RPC, Socket, etc.

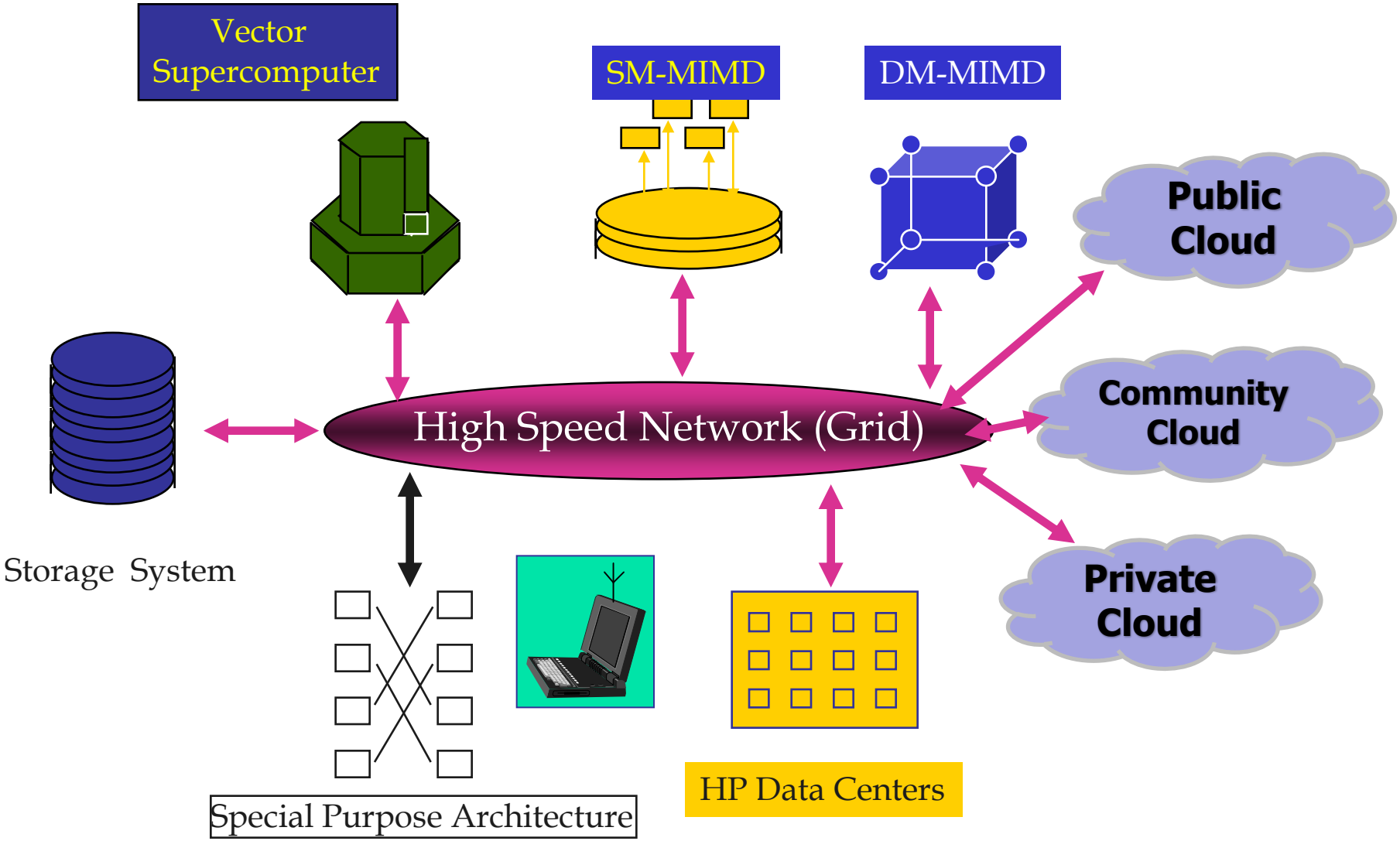
Lectures	Topics
1, 2	Introduction and overview of distributed system design issues
3, 4	Distributed programming paradigms: Message Passing, RPC
5 , 6	Distributed programming paradigms: Distributed Shared Memory
7 , 8	Distributed operating systems
9 , 10	Distributed file services
11 , 12, 13, 14	Transaction, recovery, and concurrency control
15, 16, 17, 18	Resource allocation and load balancing
19, 20, 21, 22	High speed communication protocols and Host-network Interface Designs
23, 24	Cloud computing environment
25, 26	Autonomic Computing
27, 28, 29	Term paper presentations

- **Term Paper**: Every student must submit a term paper that addresses issues of some aspects of distributed computing systems.
- **Programming Project**: Every student or a group of two maximum is required to develop a project that applies distributed computing to solve a computationally intensive application a cross a cluster of workstations. The project could be developed using any message-passing tool of your choice such as MPI, PVM, P4, or RPC.
- **Important Dates:**
  - **Paper abstract** : Sept. 30 Outline: October 28 Full Paper : Dec.8
  - **Project abstract**: Sept 30 Outline: October 28 Project Report: Dec. 8

High Performance Distributed Systems

- 2000's will be the decade of high performance computing on the Internet/Grids/Clouds/
- Advances in Computing and Communications Technology and Software will drive the move toward HPDS
- What is a Distributed System?
  - It has been around for more than three decades
  - It is a collection of computers working together on some common applications
- What is a High Performance Distributed System?
  - Include supercomptuers, massively parallel computers, and high performance data centers or HPC Clouds
  - Network should be of low latency and high throughput

High Performance Distributed System



Distributed system concepts and architecture

- Goals
- Transparency
- Services
- Architecture models
- Network Communication protocols
- Design issues
- Distributed Computing Environment

Characteristics of DS

- Multiplicity
  - multiple users
  - concurrent processes
  - replication of resources
- Dispersion
  - distributed resources
  - decentralized control

Characteristics of DS (cont...)

- Non-negligible communication delay
- Lack of global information
- Failures

Goals

- Efficiency
  - communication delay
    - data propagation
    - communication protocols
    - load distribution (congestion, bottleneck)
  - distributed processing and load balancing
- Flexibility
  - user's viewpoint--friendliness, freedom
  - system's viewpoint--evolve and migration
    - modularity, scalability, portability and interoperability

Goals (cont...)

- Consistency
  - lack of global information, replication and partitioning of data, component failures, complexity of interaction
  - consistency control in data and files--crucial issue in DS
- Robustness
  - failures in communication links, processing nodes, client-server processes, security for users and systems

Transparency

- Access
  - local and remote
- Location
  - name transparency, logical names (object)
- Migration
  - location independence, objects
- Concurrency
  - sharing of objects, time-sharing

Transparency(cont...)

- Replication
  - consistency among multiple instances
- Parallelism
  - parallel activity, specified by user
- Failure
  - fault-tolerance
- Performance
  - consistent and predictable performance, independence of remote on delays

Transparency(cont...)

- Size
  - modularity and scalability
- Revision
  - vertical growth

DS Design Issues

- Communication, synchronization, distributed algorithms
  - interaction and control transparency
- process scheduling, deadlock handling, load balancing
  - performance transparency
- resource scheduling, file sharing, concurrency control,
  - resource transparency

DS Design Issues (cont)

- Failure handling, configuration, redundancy
  - failure transparency

Services

- Primitive services
  - communication, synchronization, processor multiplexing--provided by kernel
  - send and receive primitives
- System services
  - name or directory
    - name: to locate users, processes or machines
    - directory: files and communication ports
    - address and location--communication path

Services (cont...)

- Network
  - need broadcast or multicast server in OS
- Time
  - clocks-- synchronization and scheduling
  - physical clock
    - real time
  - logical clocks
    - total order of event occurrences

Services (cont.)

- File
  - sharing of system resources
  - directory sub-server--access control
  - security sub-server--authentication
- Migration
- Authentication

Characteristics of Distributed Systems

- Geographic separation
- Level of cooperation

Level of Cooperation	Geographical Separation	
	Close	Distant
None	Centralized independent processors	Decentralized system
Modest	Local area network	Distributed network
Close	Locally distributed multiple processor system	Geographically distributed multiple processor system
Intimate	Multiprocessor	None

Basic Design Concepts

- Many names given to Dist. Systems:

  grid computing, net-centric systems, Data Centers, Cloud Computing, etc.

  distributed data processing,

  distributed multiple computer systems,

  distributed database systems,

  network-based computing, supercomputing

  cooperative computing,

  client-server systems,

  geographically distributed multiple computer systems

- Distributed systems can be defined in terms of their characteristics and the provided services or transparencies

Basic Design Concepts

Three characteristics: logical units, distribution, and transparency

- ***Logical:*** Distributed systems have been defined as `` a collection of logical units that are interconnected logically and physically with an agreeable protocol for executing distributed programs."
- ***Distribution***: Four components of a system might be distributed:
  - hardware or processing logic
  - data
  - the processing itself
  - control (e.g., message passing software, operating system)
- ***Transparency:*** The degree to which the system looks like a single, integrated system to users and applications

Basic Concepts

- ANSA defines eight different types of transparency:
  1. Access Transparency: allows local and remote resources to be accessed using same set of operations.
  2. Location Transparency: allows objects to be accessed without knowing their actual locations.
  3. Concurrency Transparency: enables multiple users or distributed applications to run concurrently without any conflict.
  4. Replication Transparency: allows several copies of files and data to exist in order to increase reliability and performance without users knowing/requesting the replication

Basic Concepts

5. Failure Transparency: allows the concealment and tolerance of faults; i.e. it enables distributed applications to run to completion despite some failures in hardware and/or software
6. Migration Transparency: allows system components (processes, threads, files, etc.) to move within the system without affecting the operation of users or application programs
7. Performance Transparency: provides the system with the ability to be reconfigured to improve performance as loads and other system dynamics change
8. Scaling Transparency: allows scaling the system and applications without the need to change the system structure or the application algorithm

Basic Concepts

- Our Definition of HPDS:

``networked (loosely coupled) system of independent computing/storage resources with adequate software structure to enable the integrated use of these resources toward a common goal"

  - The computing resources: workstations, PC's, minicomputers, mainframes, supercomputers, special purpose parallel computers, etc.
  - The underlying network interconnecting the distributed resources: LAN's, MAN's and even WAN's, can have different topologies (bus, ring, full connectivity, random interconnect, etc.)

Evolution of Distributed Computing Systems

- 1st Generation - Remote Execution Systems (RES)

  Spans 1970's era users mainly doing remote login, remote data entry and job execution, in addition to file transfer capability.

- 2nd Generation - Distributed Computing Systems

  Spans the 1980s where significant advances occurred at three fronts: the computing technology, network technology experienced the availability of software tools and environments

- 3rd Generation - High-Performance Distributed Systems (Metacomputing, Grid Computing):

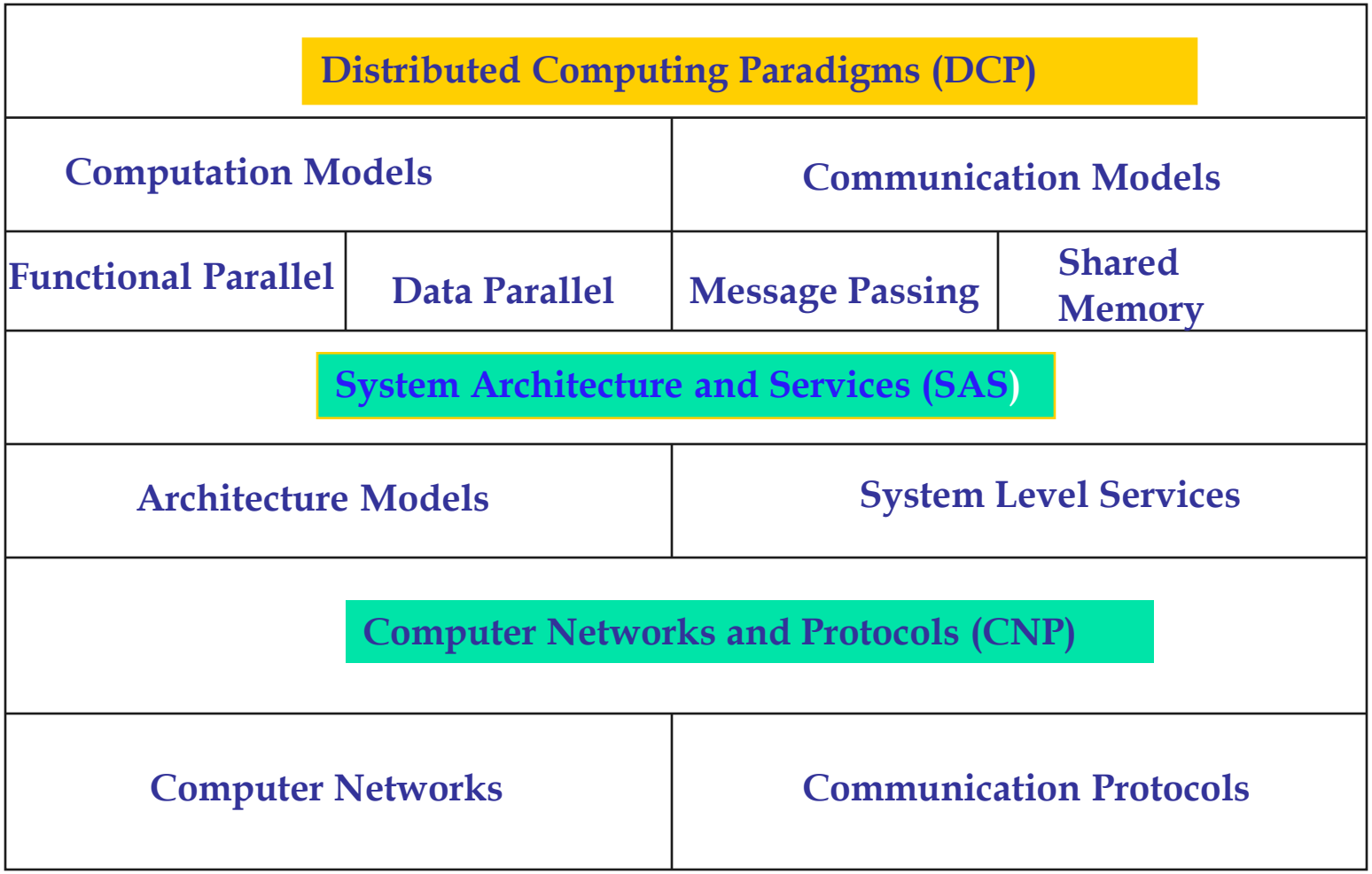
  span the 1990's which will be the decade of where distributed systems will be used in industry, health care, finance, science, education and military

- 4th Generation – Autonomic Computing – The Next Era of Computing

Evolution of Distributed Computing Systems

Dist. System Generation	Computing Resources	Network Resources	Software /Applications
Remote Execution Systems (RES)	Mainframe, Minicomputers: Centralized:Expensive	Packet Switched networks; Slow WAN(2400-9600bps) Few networks	Terminal emula. Remote login adata entry
Distributed Computing Systems (DCS)	Workstation & PCs Mainframes, Minicomputers: Distributed: Not expensive	Fast LANs & FDDI,DQDB; MANs 10-100Mb/s, Fast WANs(T1(1.5Mbps) Large number	Net File Systems (NFS) Message-passing tools (PVM,P4 ,ISIS) onLine Transaction Systems - Airline reservations
High-Performance Distributed Systems (HPDS)	Workstations & PCs; Parallel/supercomputer	High-Speed LANs, MANs,WANs, Fast Gigabit Ethernet, ATM Explosive number	Fluid turbulence Video-On-Demand; Parallel/Dist . Computing
Cloud/Autonomic Computing	HP computers & PCs Smart devices	+ wireless networks	Ecomerce, online, HPC, transactions

Distributed Systems Design Framework



Communication Network Architecture: Layer 1

- Distributed systems rely entirely on computer networks for the communication of data and control information, require high network performance and reliability
- Low-bandwidth and high-latency in computer networks represent main obstacle in developing high-performance dist. systems
- Computer networks have evolved significantly

  8 s to transmit 1 Kbytes data over 1 kbps networks -> only 8 μs to transmit the same data over 1 Gbps
- Network design issues involve three sub-layers:

  - network type (LAN, MAN, WAN, DAN (Desk Area Network), protocols, network interface.

Network Type

- Characterizes network type as local area network (LAN), wide area network (WAN), metropolitan area network (MAN), or personal area network (PAN); based on the geographical area they cover
- Also defines the topology, the type of medium (e.g., coaxial cable, fiber optics, wireless, satellite, etc.), the medium speed, and the type of network access protocols and services
- **WAN**s or *long haul networks*

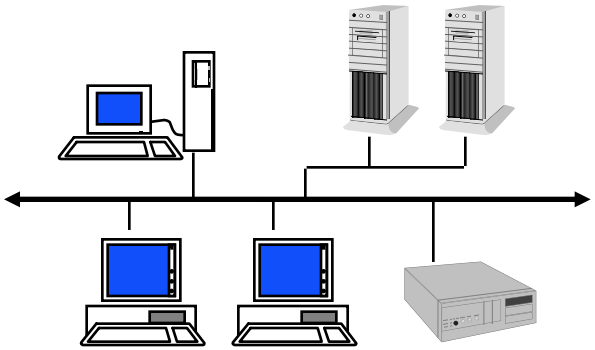
  - Intended for use over large distances

  - Typically operate at slower speeds than other technologies, have high communication delays
- **MAN**s

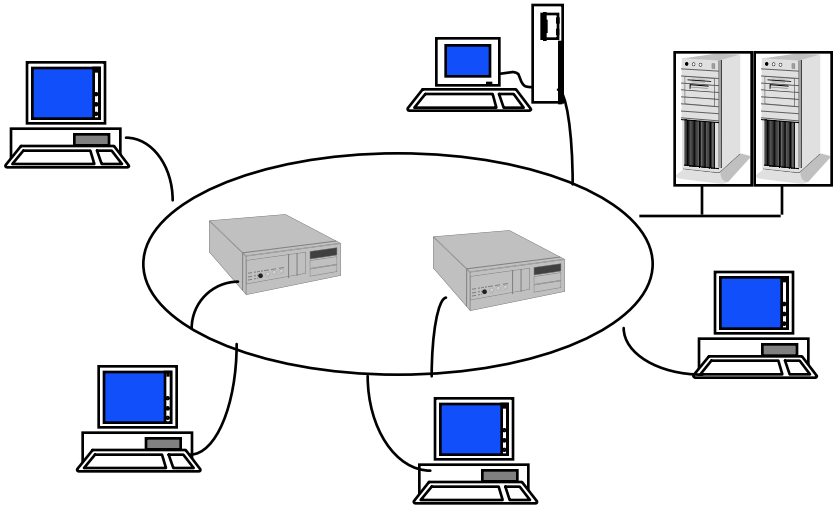
  - Spans intermediate distances

  - operates at medium-to-high speeds; 56Kbps to 100 Mbps

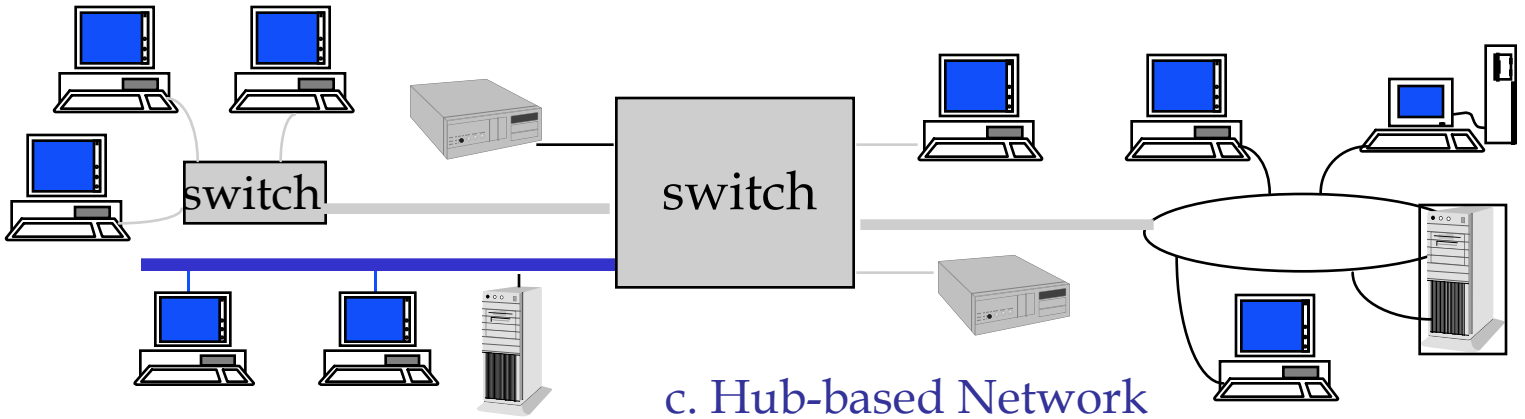
Network Technologies



a. BUS Network



b. RING Network



c. Hub-based Network

Topologies

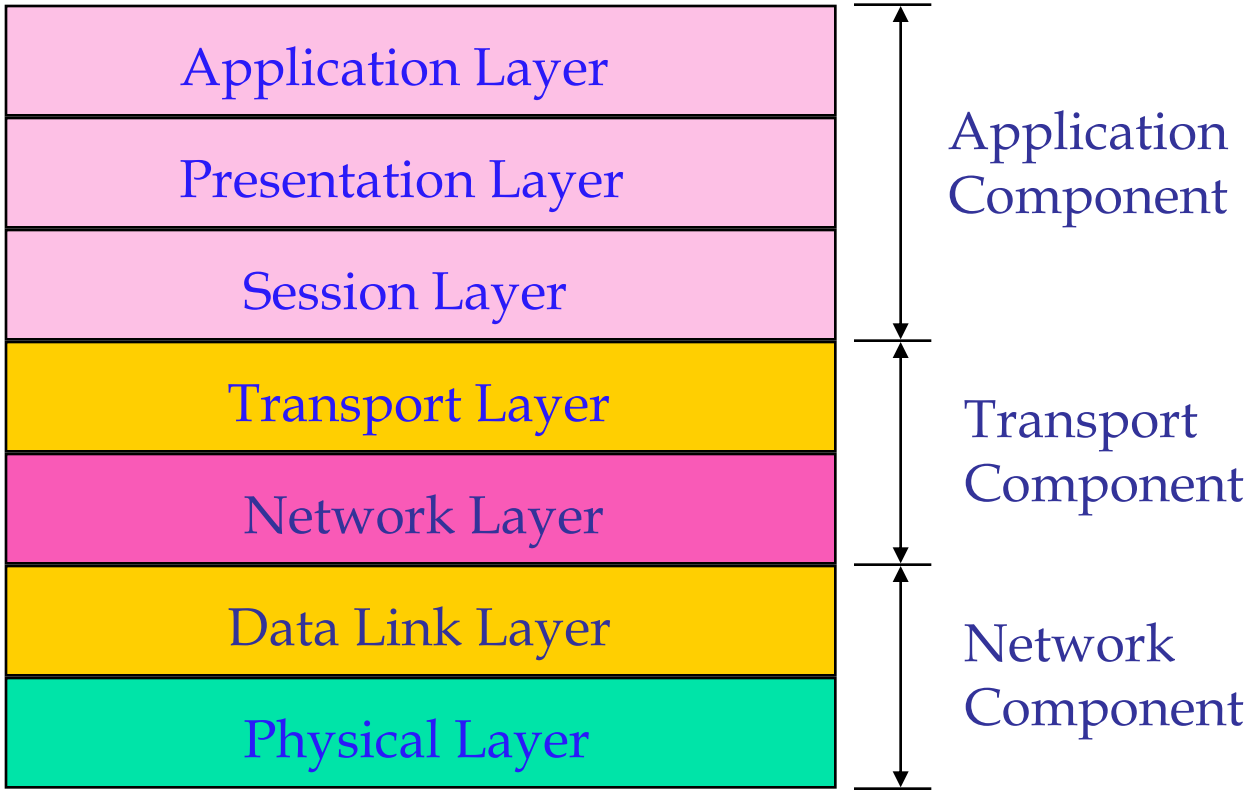
- The topology of computer networks can broadly divided into six types: bus, loop, star, hierarchical, fully connected and random.
- Bus-based Networks
  - bus is usually time-shared among computers
  - control may either be centralized or distributed
  - main limitation is its scalability to accommodate large number of connections
- Loop-based Networks
  - computers are connected to the ring nodes who are connected using point-to-point communication links
  - main advantages: simplified routing scheme, fast connection setup, cost proportional to number of users and interfaces, and provide high throughput
  - main limitation of loop topology is its reliability
- Star-based (switched-based or hub-based) Networks
  - central switch acts as master routing device
  - can be made hierarchical where a slave computer can act as a master switch for another cluster and so on
  - expected to be an attractive topology in designing high speed networks (e.g. ATM)

Switching Techniques

- Packet Switching
  - packet carries source and destination addresses
  - intermediate nodes store and forward the packets
  - not reliable, may disorder on the receive side.
  - delay is not guaranteed
- Circuit Switching
  - telephone
  - 3 phases, setup connection, data transmission, and release connection.
  - reliable, received in order.
  - delay is fixed
- Variations
  - Message Switching: variable message size as the packet length.
  - Virtual Circuit: like circuit switching, but shared circuits.

Communication network protocol

- Open System Interconnection (OSI) from International Standards
- Transmission Control Protocol/Internet Protocol (TCP/IP) from Department of Defense (DoD)
- User Datagram Protocol (UDP)


Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer
Application Component
Transport Component
Network Component

TCP/IP protocol suite

- Interconnect networks
- TCP--transport layer--TP4
- IP--network layer
- Transport layer
  - connection-oriented and connectionless
  - virtual circuit and datagram
- TCP and User datagram protocol (UDP)

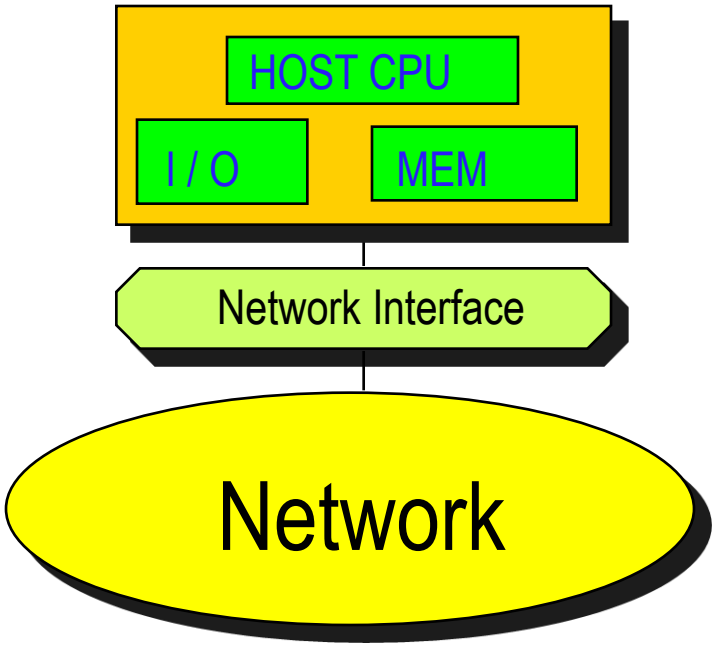
TCP/IP protocol suite(cont.)

- Process id
- Ports
- Internet address
  - network address
    - network, sub-network addresses--domain
  - host address
- socket--abstraction of network I/O
  - read and write operations

TCP/IP protocol suite(cont.)

- Socket descriptor
- system calls--socket, connect, bind
- connectionless
  - sendto, recfrom

Network Interface



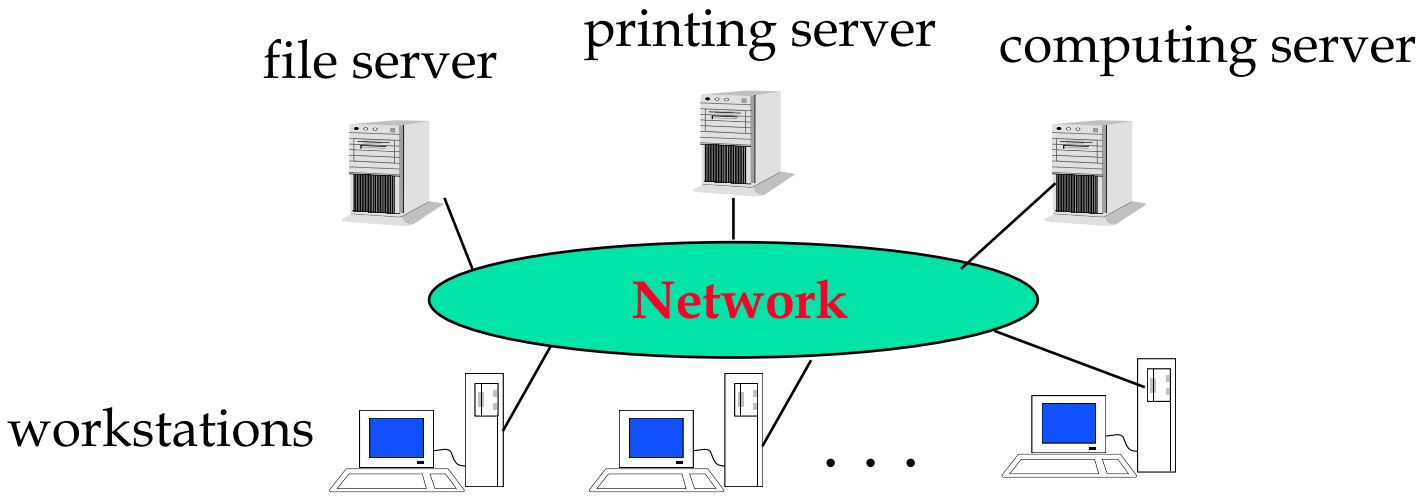
- Responsible for transferring data from host memory to the communication medium and vice versa
- Perform functions related to message assembly, formatting, routing and error control

Tradeoff:

- more functions allocated to the network interface, the less load imposed on the host to process network functions
- however, the cost of the network interface will increase

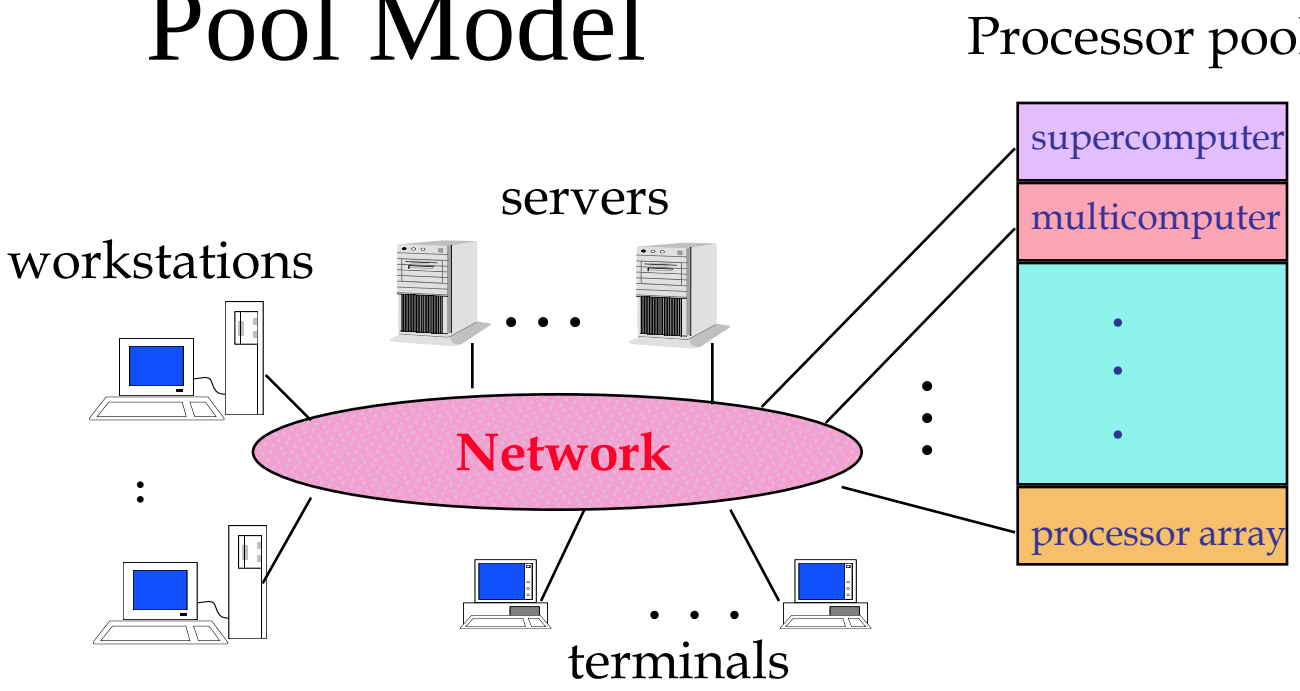
Distributed Systems Architectural Models

- **Server Model or client/server model**



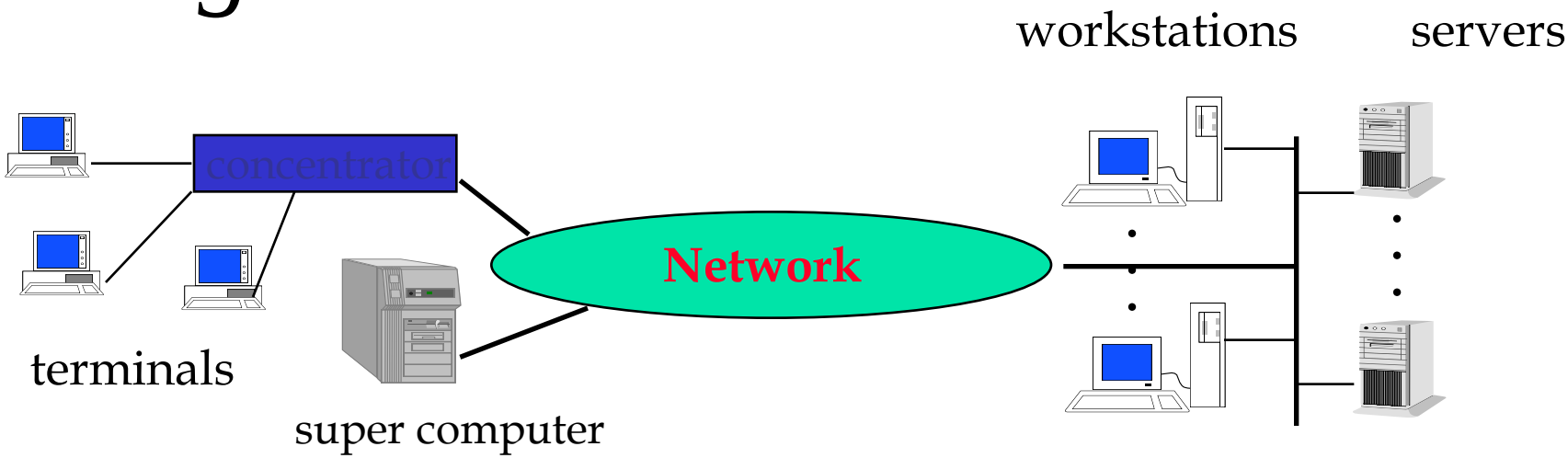
- Majority of distributed systems are based on this model
  - Most computations are done on the client side
  - Share data between users and applications
  - Share file servers and directory servers
  - Share expensive peripheral equipment

Pool Model



- Rack full of CPUs which can be dynamically allocated to users on demand
- Users given high-performance graphics terminals, or smart devices
- All the processors belong equally to everyone whenever they are needed
  - Advantage: better utilization of resources
  - Disadvantages: increased communication between users and pool resources

Integrated Model



- Each computer performs both the role of a server and the role of a client
- Computing resources managed by a single distributed operating system that makes them appear to the user as a single image system
-

Hybrid Model

System Level Services: Operating System

Centralized Operating Systems

- Allocates computer resources (memory, CPU, I/O devices, files, etc.) to processes
- Gives users a command language to invoke OS facilities
- Command language to access editor, compiler, utilities, and other OS resources.
- Functions performed by an OS:

  - Receive, parse, and interpret the user commands

  - Authenticate the user request for proper authority

  - Allocate the resources needed to execute the user commands if the user is allowed to access the resources

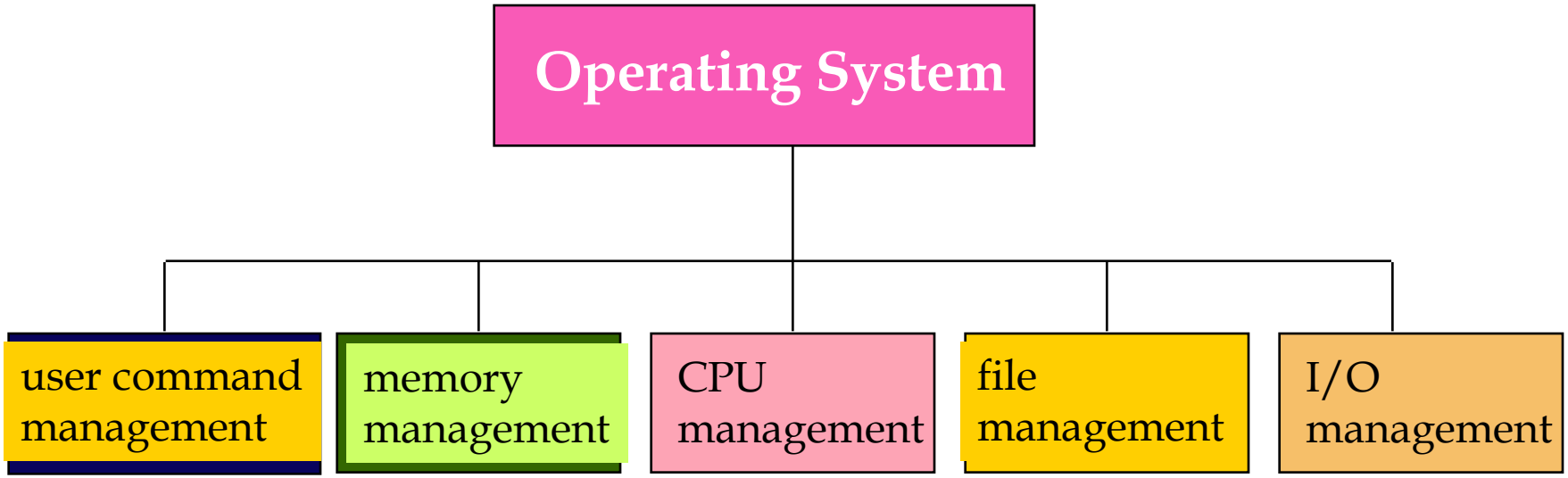
  - Schedule the user request as process

  - Monitor the processes in the system

  - Free the resources held by the processes at termination

  - Display results if the user session with information about resource utilization, etc.

Centralized Operating System



Functional View of Operating Systems

Distributed Operating Systems

- DOS handles its users very much like a centralized OS but runs on multiple, interconnected computers.
- Functions performed by a DOS:

  - Receive, parse and interpret the user command from any computer

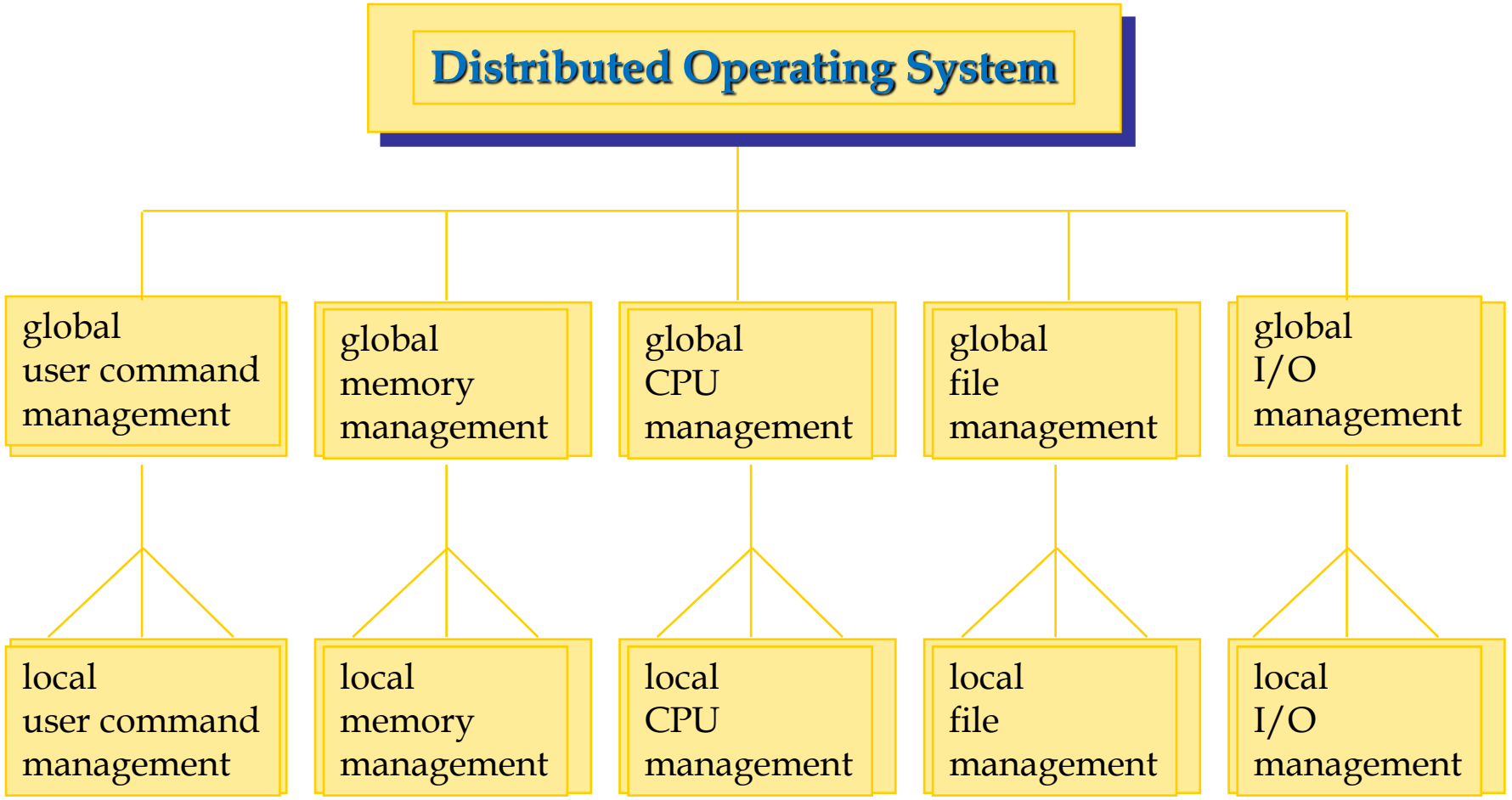
  - Authenticate the user request for proper authority.

    * Global security tables need to be created, maintained, and accessed to handle security across multiple computers

  - Allocate resources

    * This provides the major source of transparency because the resource (files, programs, directories, I/O devices, memory units, CPU cycles) may be at any machine.

    * Global information about all resources in the network need to be created, maintained, and accessed for global resource allocation



A functional view of distributed operating systems

Distributed Operating Systems

- Design Issues in Distributed Operating Systems

  - Communication of processes between different computers

  - Naming services for object across multiple computers

  - Global resource management (allocation, scheduling, de-allocation)

  - Scalability of services from a few to thousands of workstations

  - Fault management in distributed environments

  - Security and protection across computers

- Other Issues

  - Load Balance, File Systems, Concurrency Control, and Redundancy Management, etc.

Distributed Computing Paradigms: Layer 3

- Layer 2 describes the architectural models, component properties and services
- It describes what is required to build a distributed system
- Layer 3, it describes how you program distributed applications; what techniques (models) do you use?
- Also, it describes the types of tools that can be used to implement the applications

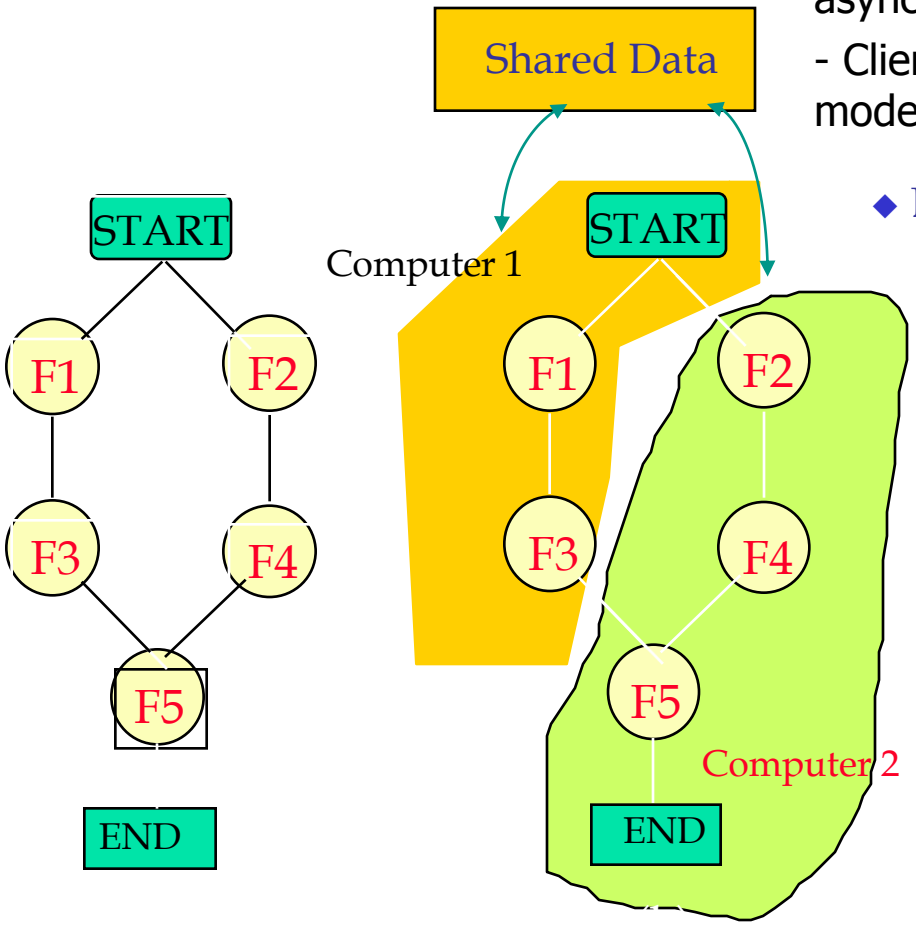
There are two sub-layers:

Computational Model

- **Communication Model**

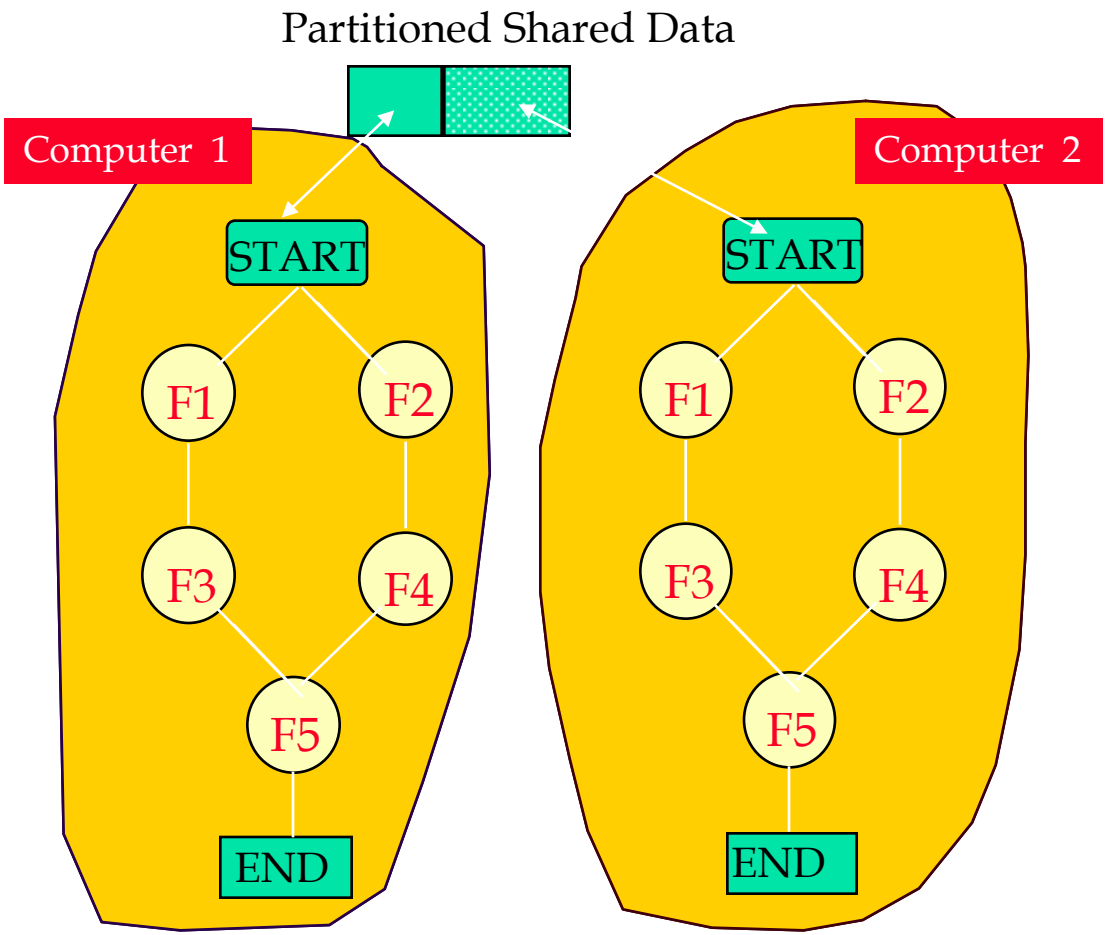
Computational Model: Functional Model

- Functional Parallel Model
  - Computers execute different threads of control,
  - It is referred to as control parallelism, asynchronous parallelism
  - Client-Server Models are variations of this model



- Limitations of Functional Parallelism
  - Asynchronous interactions could lead to data race conditions
  - If the application has large number of parallel tasks, it is difficult to achieve good load balancing

Computational Models: Data Model



- Large number of problems can be solved using this model
- It is easier to develop applications

- Amount of parallelism in functional parallelism is fixed
- Amount of parallelism in data parallelism scales with the data size
- Generally speaking, efficient distributed applications should exploit both types of parallelism

Communications Models

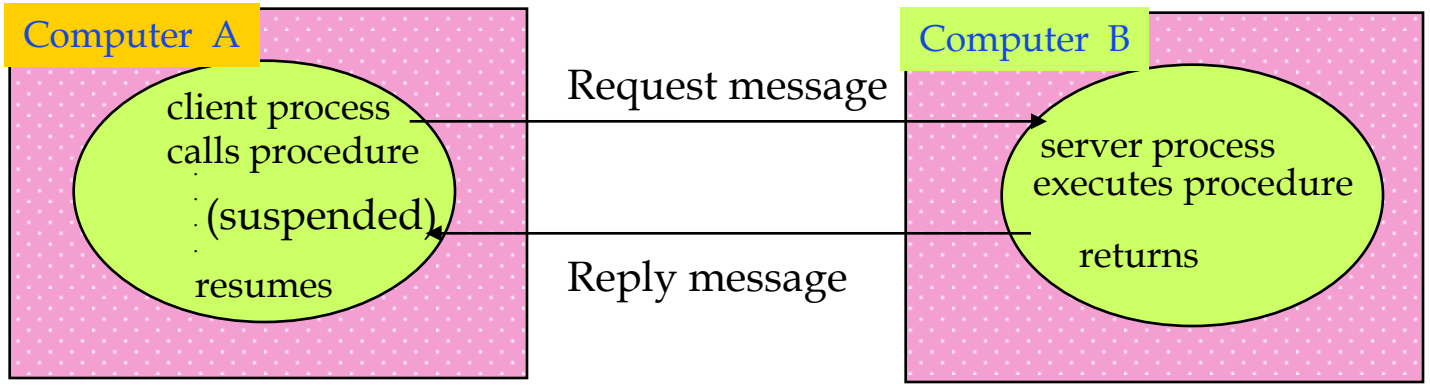
Message passing model

- Messages are used to exchange information between local and remote processes as well as between processes and the operating system
- Application developers need to explicitly involved in writing the communication and synchronization routines
- Users use two basic communications primitives: SEND and RECEIVE
- SEND and RECEIVE primitives have different implementations depending on whether or not they are blocking or Nonblocking, synchronous or asynchronous.
- The main limitations are:

  * synchronizing request and response messages
  * handle data representations
  * machine addresses
  * handle system failures that could be related to communications network or compute failures
  * debugging and testing is difficult

Communication Models: Remote Procedure Calls:

- It alleviates some of the difficulties encountered in message passing model
- Procedure call mechanism within a program is a well understood technique to transfer control and data between the calling and called programs
- RPC technique is an extension of this concept to cover networked computers
- RPC system hides all the details related to transferring the control and the data to give the illusion of calling a local procedure
- RPC model provides a methodology for communication between the client and server parts of a distributed application

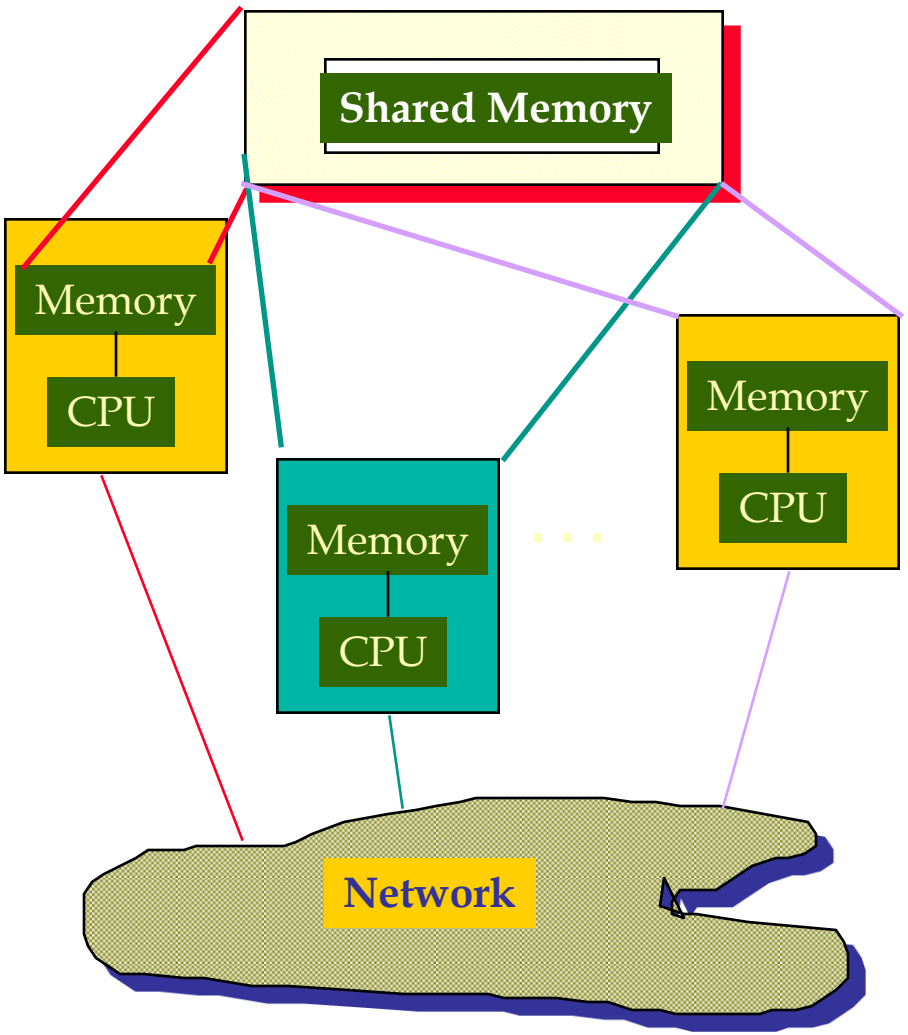


Remote Procedure Calls (cont.)

- *Main difference between ordinary procedure calls and RPC*
  - ***called and calling processor usually run on different computers***
    - ***arguments and results are sent in messages***
    - ***prone to network and remote computer failures***
    - ***different delay times***

Distributed Shared Memory

- In message passing model, the communication between processes is highly controlled by a protocol and involves explicit cooperation between processes
- Direct shared memory communication is not explicitly controlled and requires the use of a global shared memory
- Message communication resembles the operation of postal service in sending and receiving mail
- The shared memory scheme can also be compared to a bulletin board, found in a grocery store or supermarket; it is a central repository for existing information that can be read or updated by anyone



Advantages of Distributed Shared Memory:

- easy to program
- easy to transfer complex data structures
- no data encapsulation is required
- portability (program written for multiprocessor systems can be ported easily to this environment
- Message communication needs to pass all the information; we need to handle issues like buffer management, allocation, acknowledgment, routing, flow control, and error control