



CSC801

Distributed Computing

Mrs Sunita Sahu
Assistant Professor,
Computer Engineering

Program Structure for Fourth Year Computer Engineering

Course Code	Course Name	Teaching Scheme (Contact Hours)		Credits Assigned		
		Theory	Pract. Tut.	Theory	Pract.	Total
CSC801	Distributed Computing	3	--	3	--	3
CSDC 801X	Department Level Optional Course -5	3	--	3	--	3
CSDC 802X	Department Level Optional Course -6	3	--	3	--	3
ILO 801X	Institute Level Optional Course -2	3	--	3	--	3
CSL801	Distributed Computing Lab	--	2	--	1	1
CSDL 801X	Department Level Optional Course -5 Lab	--	2	--	1	1
CSDL 802X	Department Level Optional Course -6 Lab	--	2	--	1	1
CSP801	Major Project 2	--	12 [#]	--	6	6

Course Objectives:

1. To provide students with contemporary knowledge in distributed systems.
2. To explore the various methods used for communication in distributed systems.
3. To provide skills to measure the performance of distributed synchronization algorithms.
4. To provide knowledge of resource management, and process management including process migration.
5. To learn issues involved in replication, consistency, and file management.
6. To equip students with skills to analyze and design distributed applications

Course Outcomes:

At the end of the course students will be able to

1. Demonstrate the knowledge of basic elements and concepts related to distributed system technologies.
- 2 Illustrate the middleware technologies that support distributed applications such as RPC, RMI and Object-based middleware.
- 3 Analyze the various techniques used for clock synchronization, mutual exclusion and deadlock.
- 4 Demonstrate the concepts of Resource and Process management.
- 5 Demonstrate the concepts of Consistency, Replication Management and fault Tolerance.
- 6 Apply the knowledge of Distributed File systems in building large-scale distributed applications.

Syllabus- MODULE I

Introduction to Distributed Systems - 4 hrs

I.1	Characterization of Distributed Systems: Issues, Goals, Types of distributed systems, Grid and Cluster computing Models, Hardware and Software Concepts: NOS, DOS.
I.2	Middleware: Models of middleware, Services offered by middleware.

Contents

- ❑ Definition, Issues, Goals,
- ❑ Types of distributed systems,
- ❑ Grid and Cluster computing Models,
- ❑ Distributed System Models,
- ❑ Hardware concepts, Software Concept,
- ❑ NOS, DOS
- ❑ Models of Middleware,
- ❑ Services offered by middleware

Definition

- ❑ A distributed system is a collection of independent computers, interconnected via a network, capable of collaborating on a task.
- ❑ A distributed system can be characterized as collection of multiple autonomous computers that communicate over a communication network and having following features:
 - ❑ No common Physical clock
 - ❑ Enhanced Reliability
 - ❑ Increased performance/cost ratio
 - ❑ Access to geographically remote data and resources
 - ❑ Scalability

Overview...

- Distributed system connects autonomous processors by communication network.
- The software component that run on each of the computers use the local operating system and network protocol stack.
- The distributed software is termed as middleware.
- The distributed execution is the execution of the processes across the distributed system to collectively achieve a common goal.

Definition of a Distributed System

By Tannenbaum:

A Distributed System is -

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

Definition of a Distributed System

By Lamport :

A Distributed System is -

One in which the failure of a
computer you didn't even know
existed can render your own
computer unusable

Motivation for Distributed system

- **Inherently distributed computation** that is many applications such as money transfer in the banking, or reaching a consensus among the parties that are geographically distant, the computation is inherently distributed.
- **Resource sharing** the sharing of the resources such as peripherals, and a complete data set and so on and so forth.
- **Access the geographically remote data and resources**, such as bank database, supercomputer and so on.
- **Reliability** enhanced reliability possibility of replicating the resources and execution to enhance the reliability.

Definitions of a Distributed System-more specific :

- One important characters of DS is that differences between the various computers and the way in which they **communicate** are hidden from users. **(Transparency)**
- Users and applications can interact with a distributed system in a **consistent and uniform** way, regardless of where and when interaction take lace.**(Coherency)**
- D.S should also be easy to **expand or scale**. **(Scalability)**
- A DS will normally be **continuously** available, although perhaps certain parts may be temporarily out of order. **(Reliability)**

Types of interconnected system

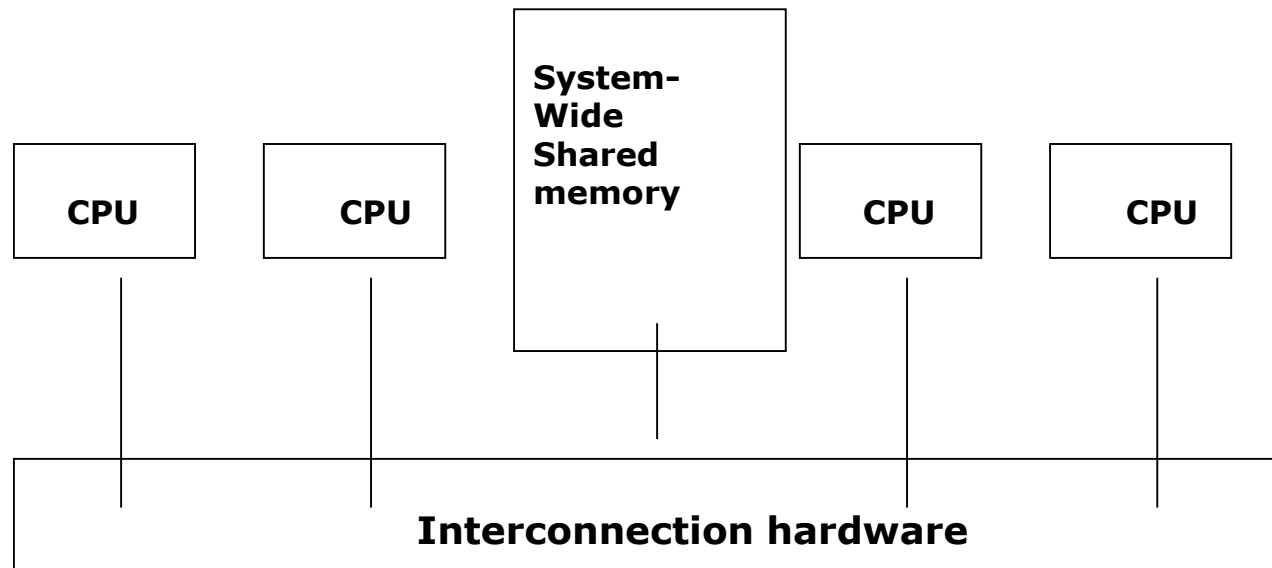
Computer architectures consisting of interconnected, multiple processors are basically of two types:

- 1). Tightly coupled system
- 2). Loosely coupled system

TIGHTLY COUPLED SYSTEMS

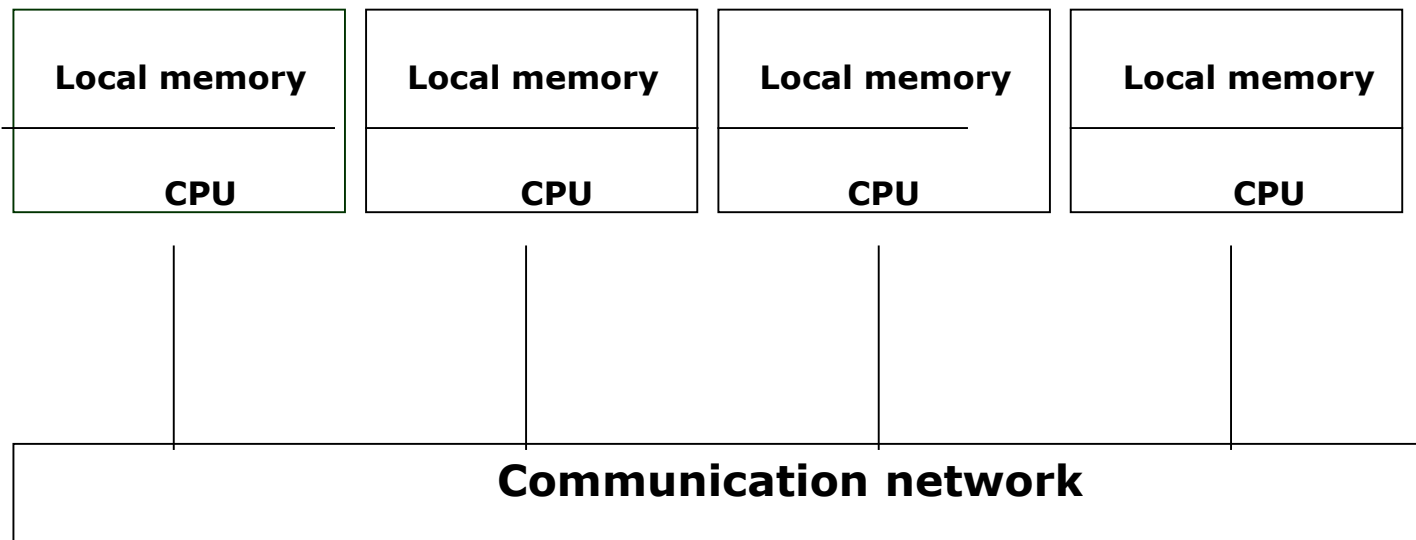
In these systems, there is a single system wide primary memory (address space) that is shared by all the processors .

Usually tightly coupled systems are referred to as parallel processing systems.



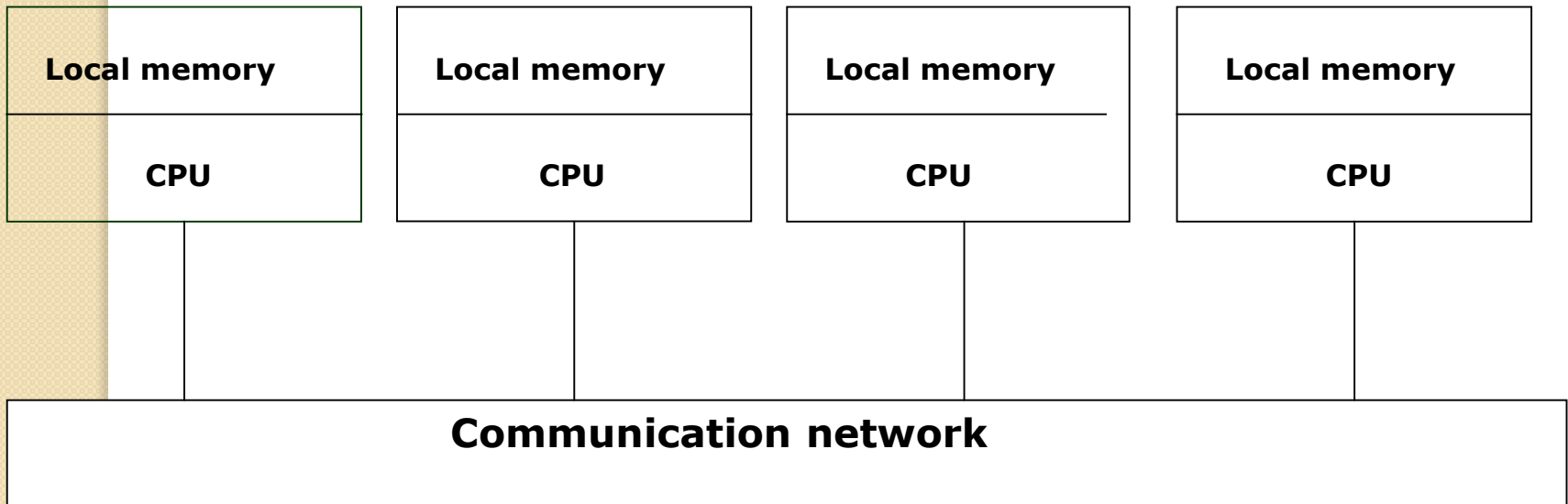
LOOSELY COUPLED SYSTEMS

- In these systems, the processors do not share memory, and each processor has its own local memory. Loosely coupled systems are referred to as distributed computing systems, or simply **distributed systems**



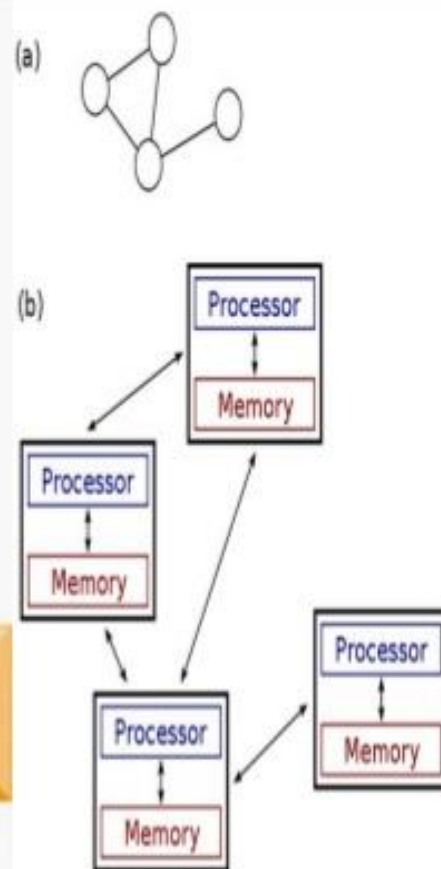
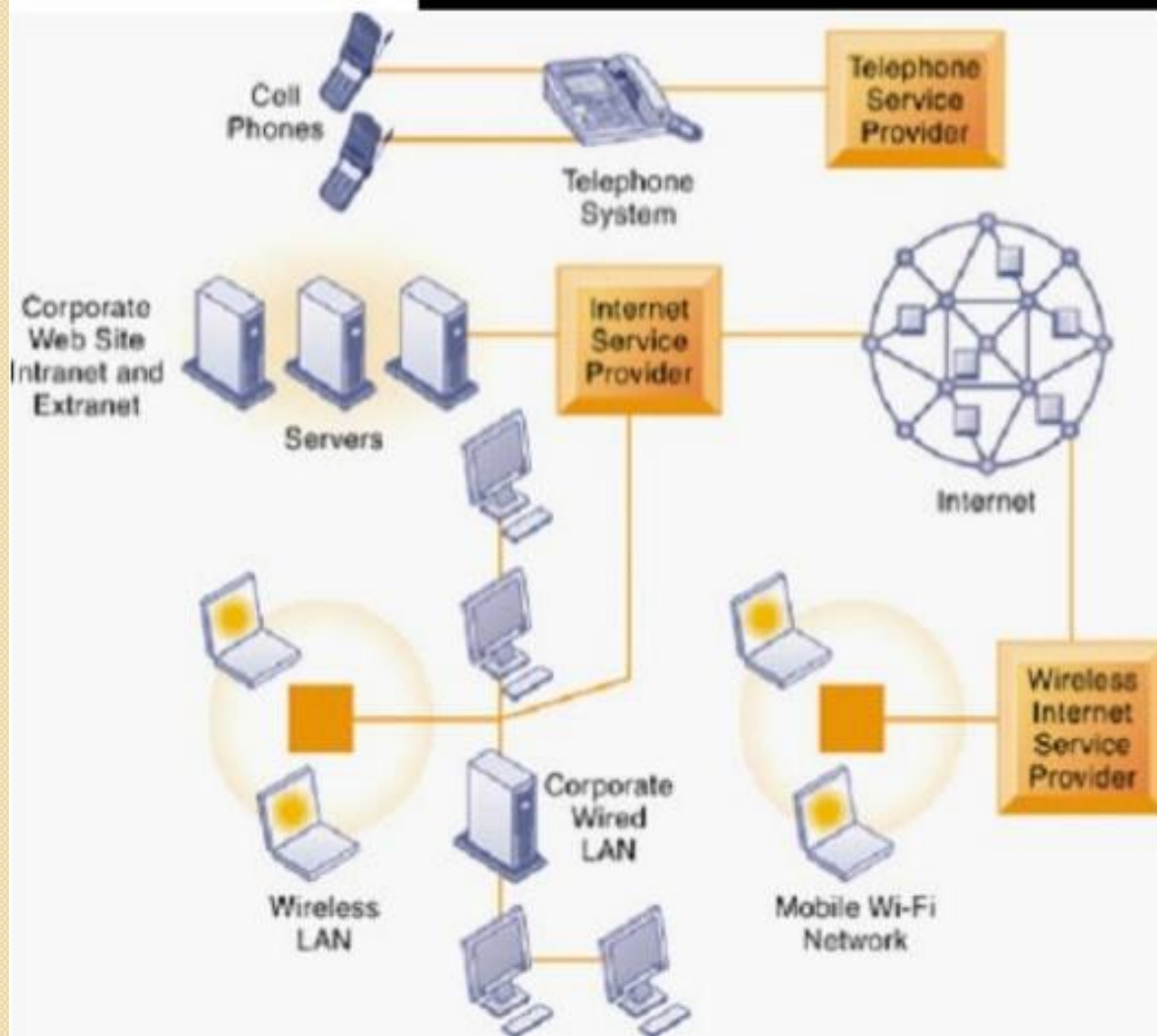
LOOSELY COUPLED SYSTEMS

- In these systems, the processors do not share memory, and each processor has its own local memory. Loosely coupled systems are referred to as distributed computing systems, or simply **distributed systems**





A Distributed System

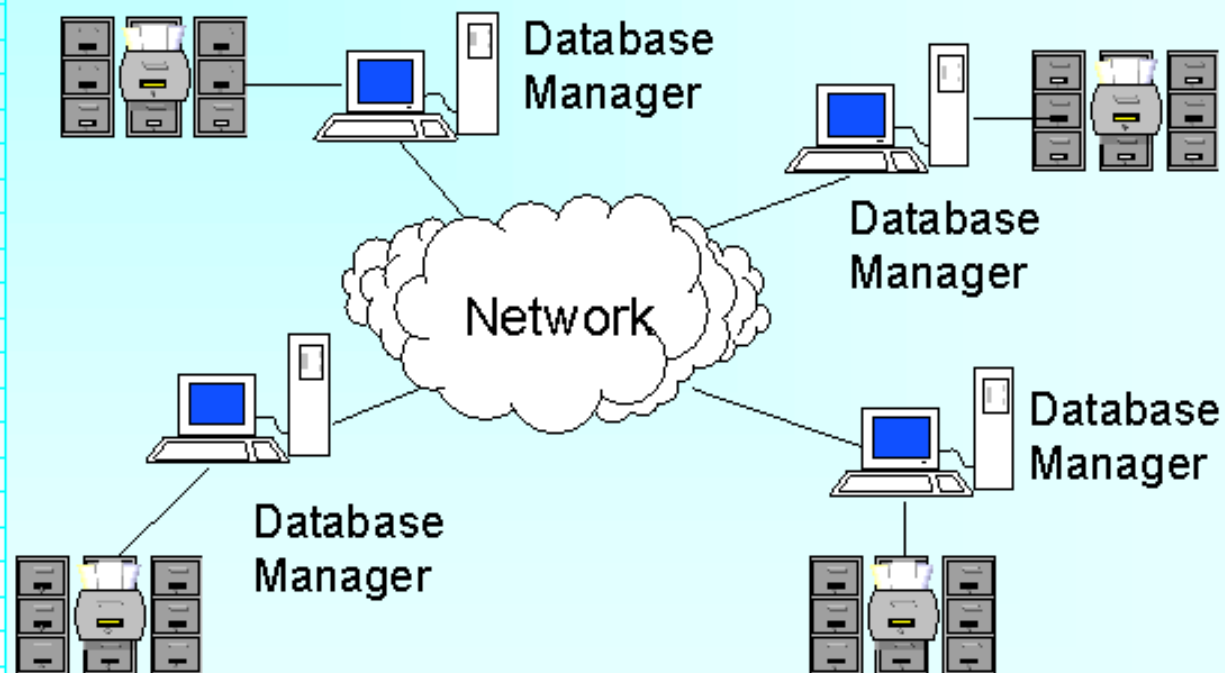


Characteristics Of Distributed System

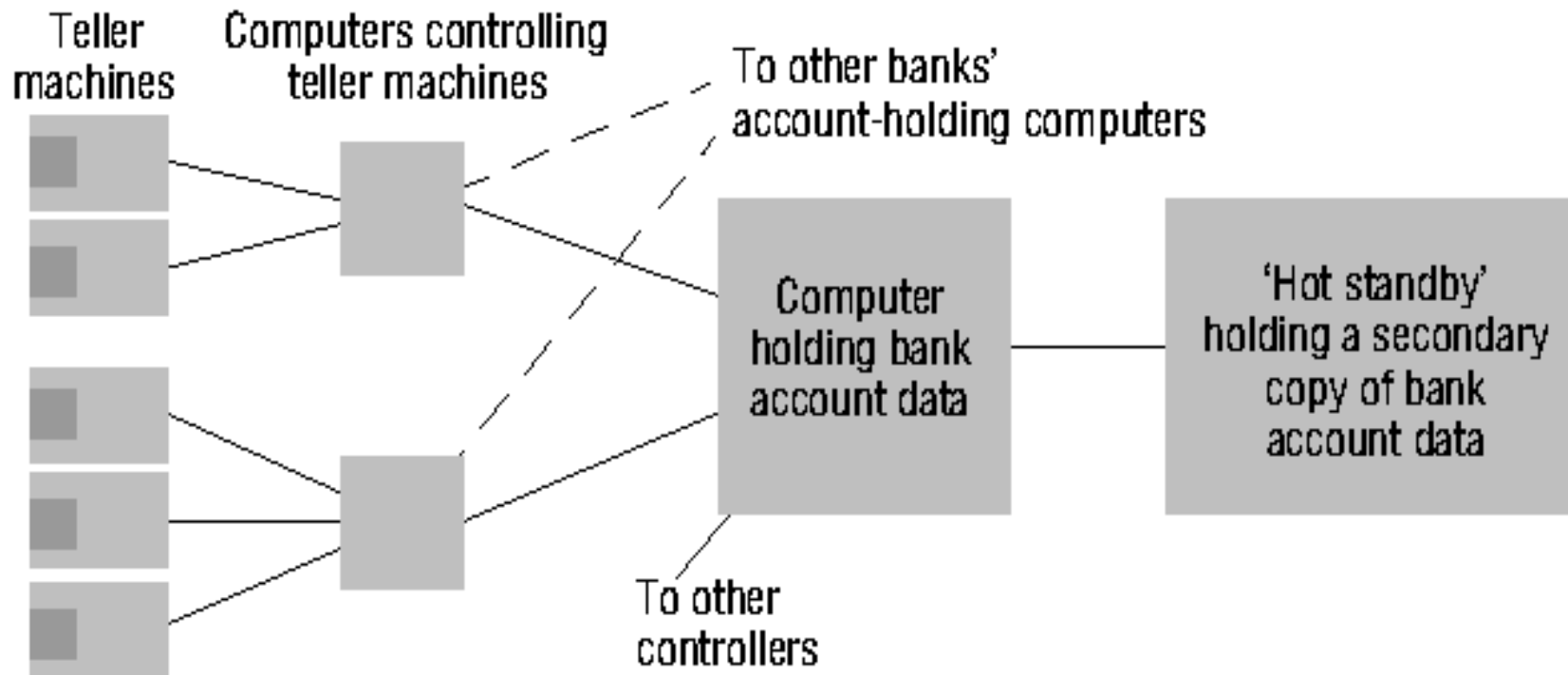
- ☐ Concurrency
- ☐ No global clock
- ☐ Independent failures
- ☐ More reliable
- ☐ Fault tolerant
- ☐ Scalable

Database Management System

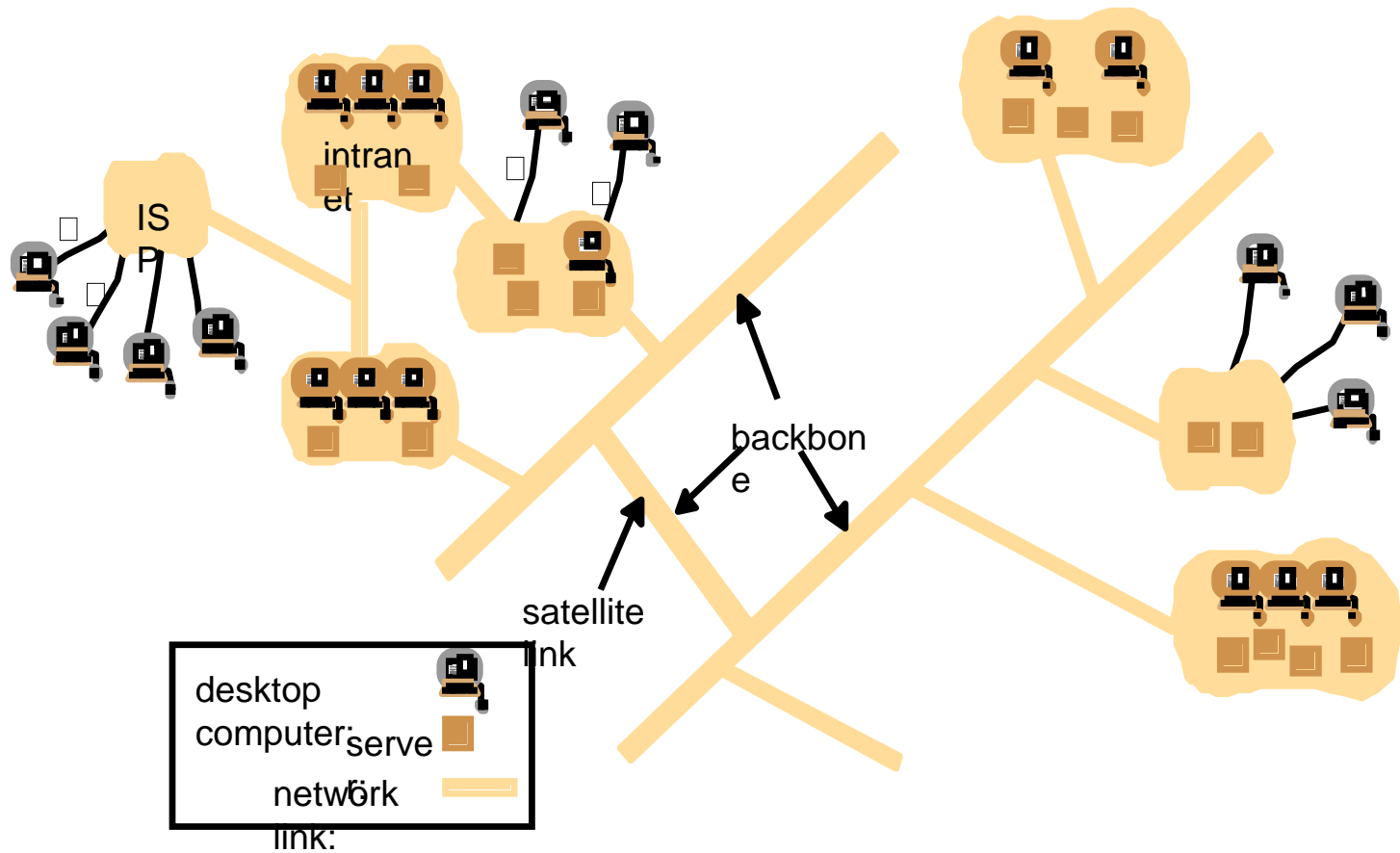
A Distributed Database



Automatic Teller Machine Network



INTERNET



WORLD-WIDE-WEB

Uppsala universitet - Windows Internet Explorer


http://www.uu.se/


File Edit View Favorites Tools Help

Google Search Bookmarks Check Translate

Windows Live Bing What's New Profile Mail Photos Calendar MSN Share

Uppsala universitet


 UPPSALA
UNIVERSITET



In English Karta Personal Enhet
Lyssna Fritext
RSS Sök

Startsida **Utbildning** Forskning Samverkan Om UU Bibliotek Kontakt

Välkommen till Uppsala universitet!



Rektor Anders Hallberg:
"Vid Uppsala universitet arbetar vi målmedvetet och långsiktigt för att alltid kunna erbjuda de bästa förutsättningarna för vår utbildning och forskning." Välkommen till Uppsala!


Nyheter

Ännu fler miljoner i anslag till Hans Ellegren
FN:s Karen AbuZayd årets Dag
Hammar skjöld föreläsare

Genvägar

- » Alumn
- » Doktorand
- » Fakulteter och enheter
- » Internt för anställda
- » Lediga anställningar
- » Pressinformation
- » Student
- » Stöd Uppsala universitet

Blåsenhus




Blåsenhus - Uppsala universitets nyaste campusområde för nya

Aktuellt

Om Influenta A (H1N1)

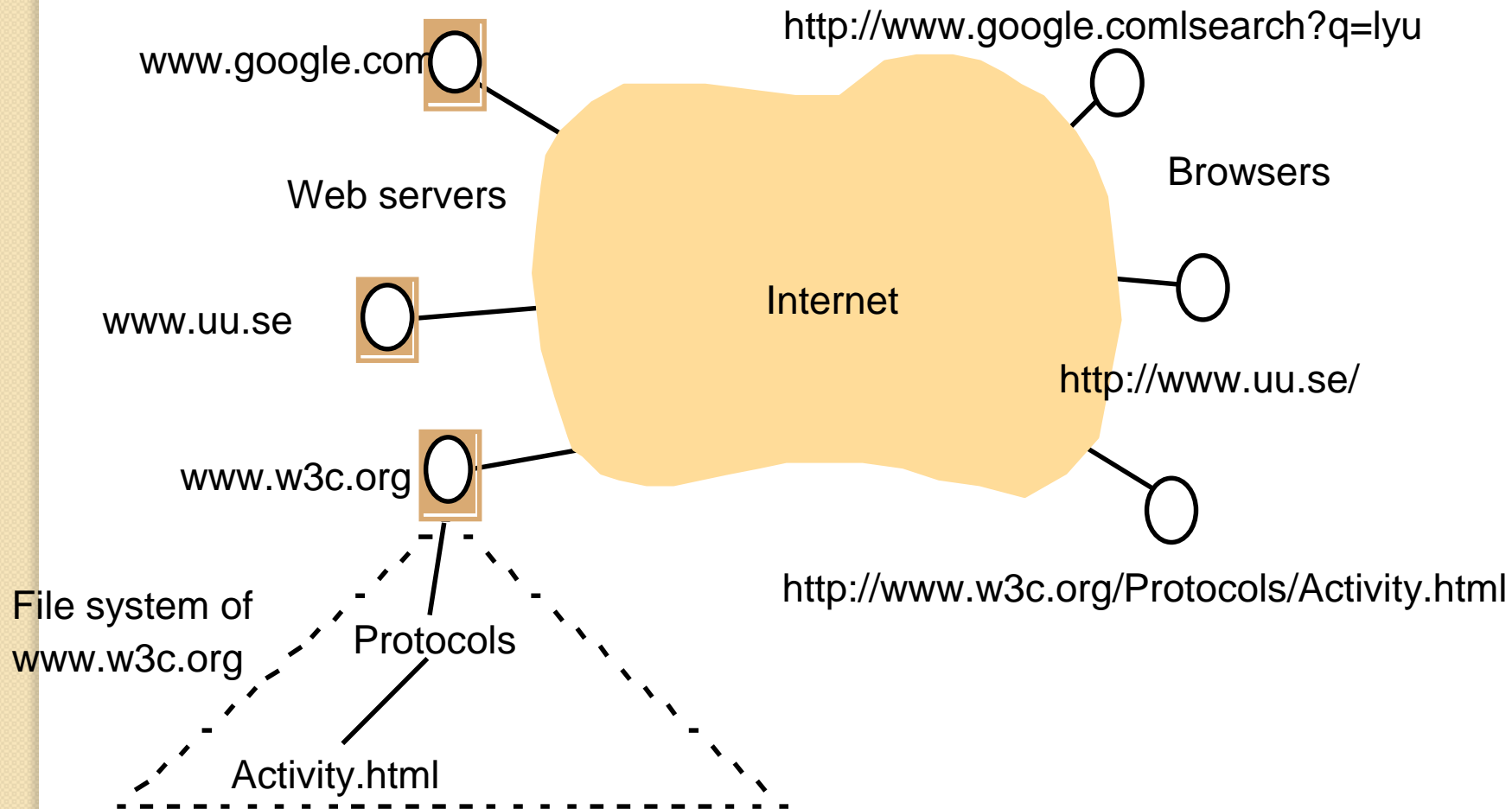
Här kan du vaccinera dig

Expedition Antarktis

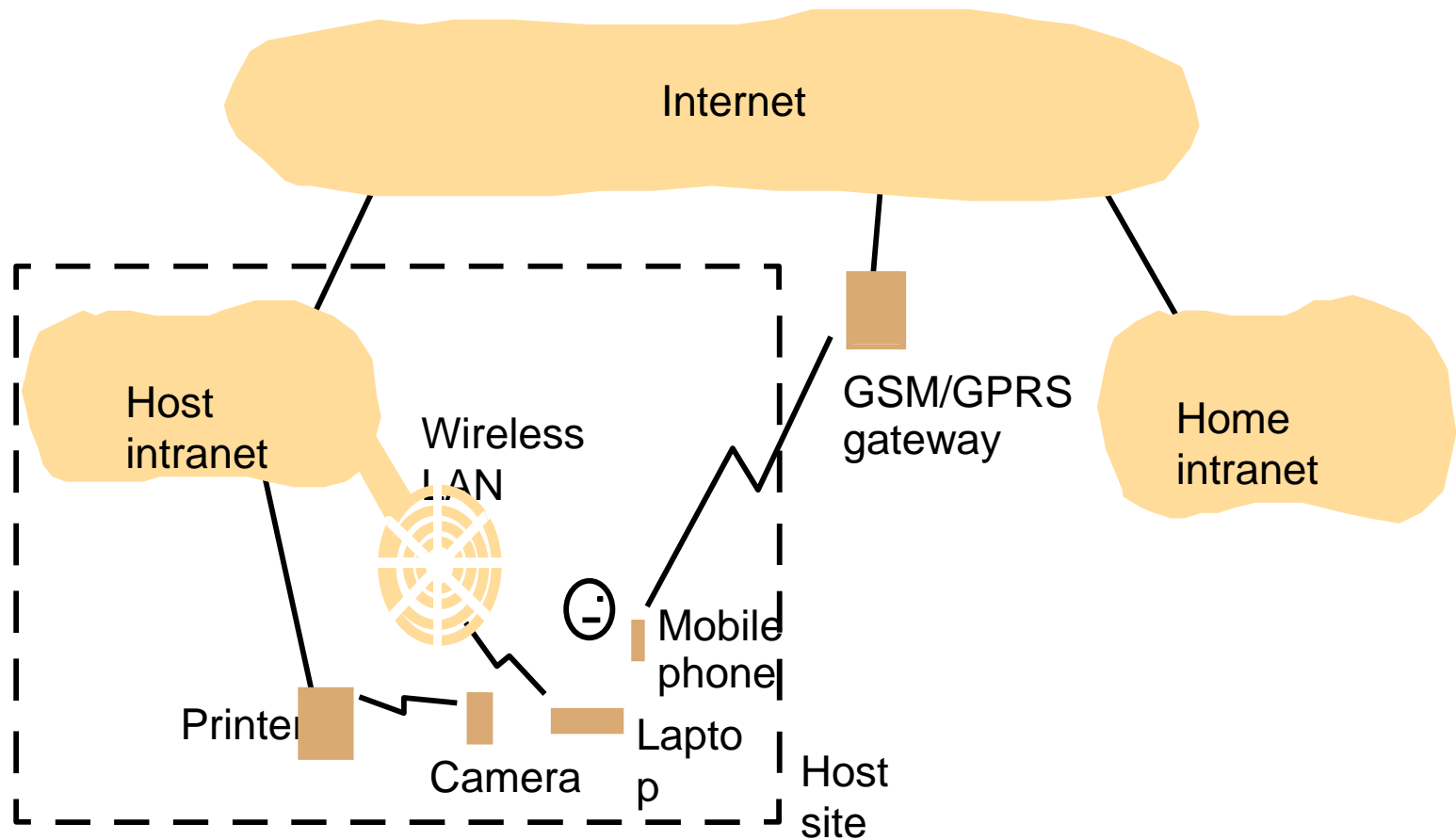


Forskningsexpedition till Antarktis om influensavirus och antibiotikaresistens. Följ forskningsresan på professor Björn Olsens blogg.

WEB SERVERS AND WEB BROWSERS



MOBILE AND UBIQUITOUS COMPUTING



Examples of distributed systems

Finance and commerce	eCommerce e.g. Amazon and eBay, PayPal, online banking and trading
The information society	Web information and search engines, ebooks, Wikipedia; social networking: Facebook and MySpace
Creative industries and entertainment	online gaming, music and film in the home, user-generated content, e.g. YouTube, Flickr
Healthcare	health informatics, on online patient records, monitoring patients
Education	e-learning, virtual learning environments; distance learning
Transport and logistics	GPS in route finding systems, map services: Google Maps, Google Earth
Science	The Grid as an enabling technology for collaboration between scientists
Environmental management	sensor technology to monitor earthquakes, floods or tsunamis

Distributed System

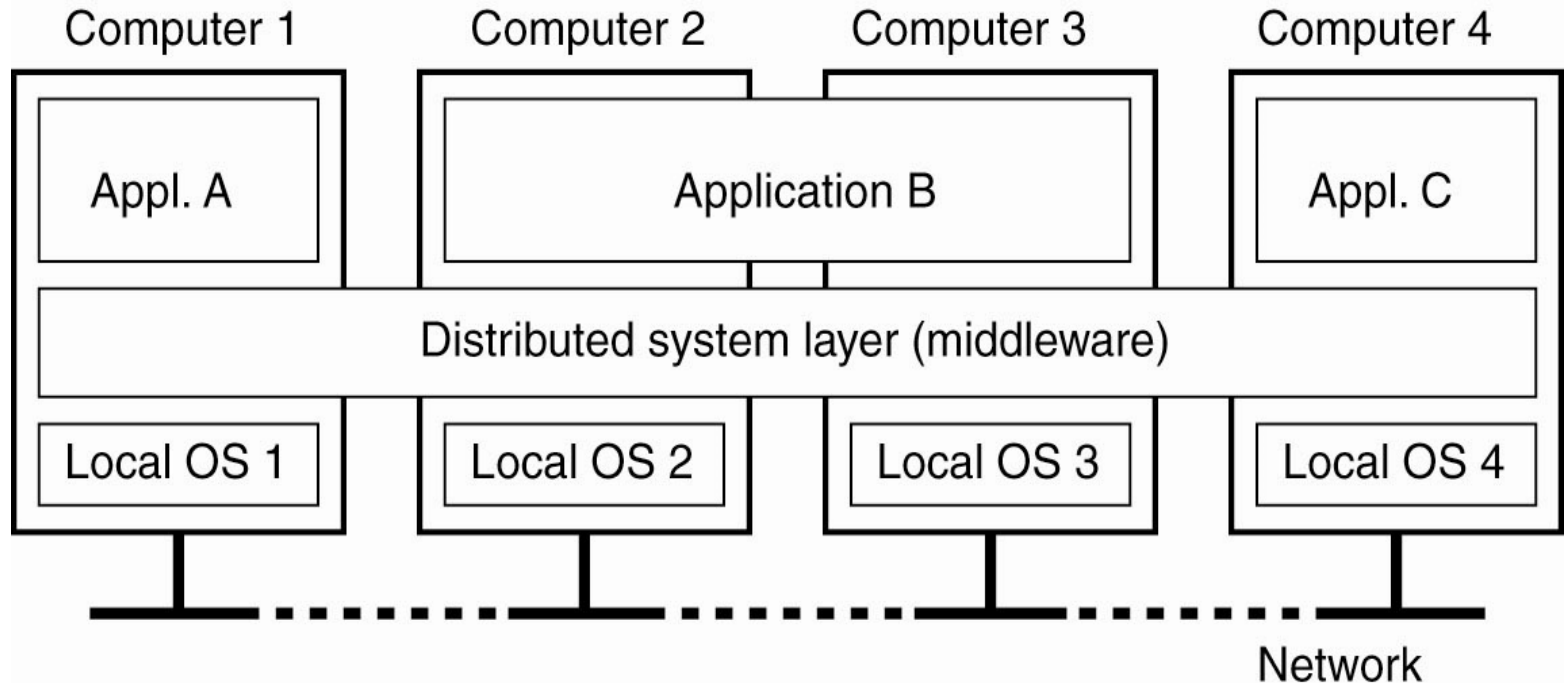
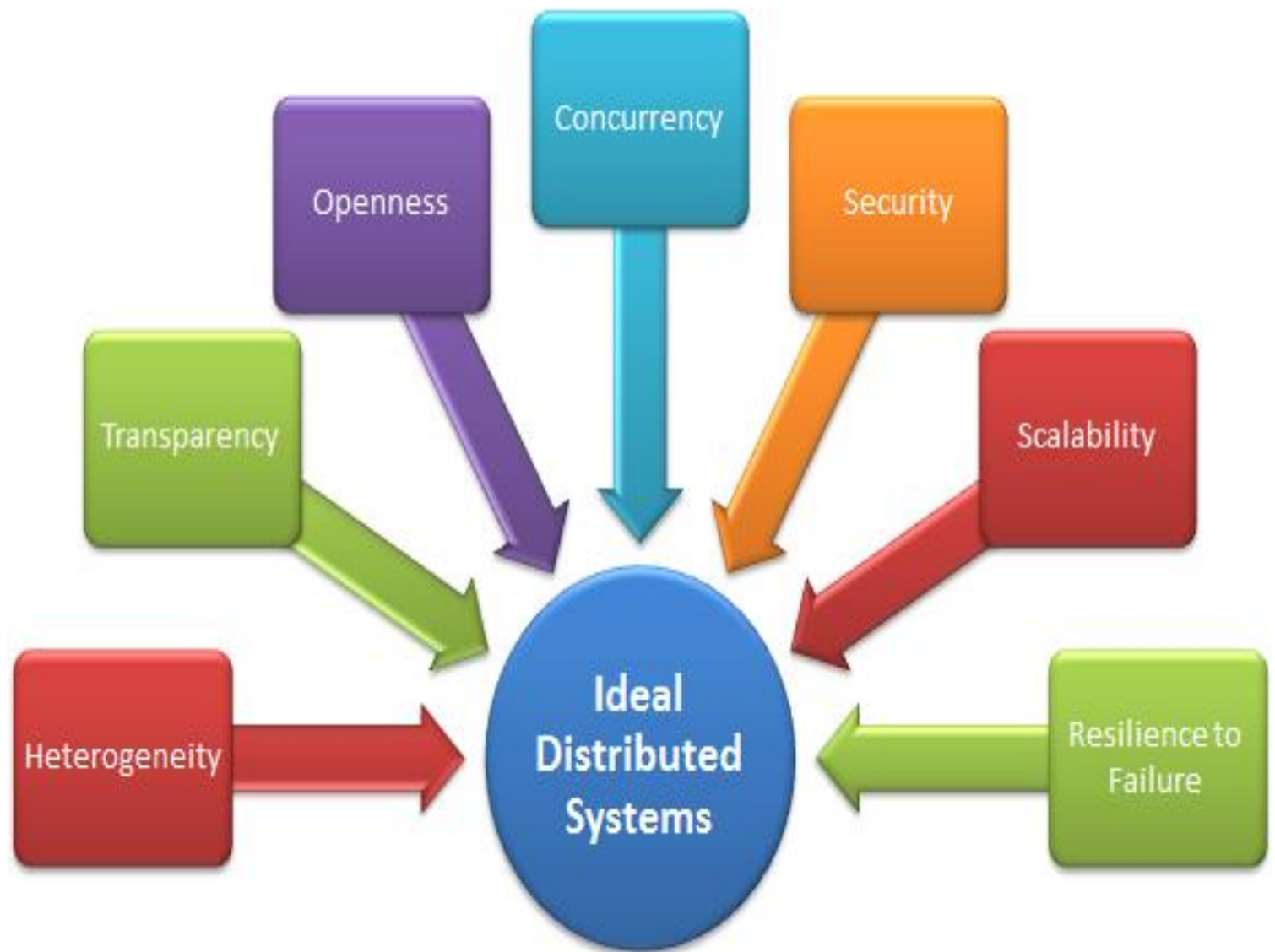


Figure 1-1. A distributed system organized as middleware. The middleware layer extends over multiple machines, and offers each application the same interface.

GOALS:COMMON CHARACTERISTICS

- ❑ Making resources accessible
- ❑ Openness
- ❑ Transparency
- ❑ Security
- ❑ Scalability
- ❑ Failure Handling
- ❑ Concurrency
- ❑ Heterogeneity



Making resources accessible

- The main goal of a distributed system is to make it easy for the users (and applications) to access remote resources, and to share them in a controlled and efficient way.
- Resources can be just about anything, but typical examples include things like printers, computers, storage facilities, data, files, Web pages, and networks,

Reasons to share resources.

- Economics.

What are the resources?

- Hardware
 - – Not every single resource is for sharing
- Data
 - – Databases
 - – Proprietary software
 - – Software production
 - – Collaboration
- Different resources are handled in different ways, there are however some generic requirements:
 - – Namespace for identification
 - – Name translation to network address
 - – Synchronization of multiple access

OPENNESS

- ❑ An open distributed system is a system that offers services according to standard rules that describe the syntax and semantics of those services.
- ❑ Detailed interfaces of components need to be published.
- ❑ New components have to be integrated with existing components. An open distributed system should also be extensible.
- ❑ Differences in data representation of interface types on different processors (of different vendors) have to be resolved.

TRANSPARENCY

- Distributed systems should be perceived by users and application programmers as a whole rather than as a collection of cooperating components.
- Ability to hide the fact that process and resources are distributed .
- Transparency has different aspects.
- These represent various properties that distributed systems should have.

Transparency in a Distributed System

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 is replicated
Concurrency	Hide that a resource may be shared by several competitive users
Failure	Hide the failure and recovery of a resource

ACCESS TRANSPARENCY

- **Enables local and remote information objects to be accessed using identical operations.**
- **Example: File system operations in NFS.**
- **Example: Navigation in the Web.**
- **Example: SQL Queries**

LOCATION TRANSPARENCY

- **Enables information objects to be accessed without knowledge of their location.**
- **Example: File system operations in NFS**
- **Example: Pages in the Web**
- **Example: Tables in distributed databases**

CONCURRENCY TRANSPARENCY

- **Enables several processes to operate concurrently using shared information objects without interference between them.**
- **Example: NFS**
- **Example: Automatic teller machine network**
- **Example: Database management system**

REPLICATION TRANSPARENCY

- Enables multiple instances of information objects to be used to increase reliability and performance without knowledge of the replicas by users or application programs
- Example: Distributed DBMS
- Example: Mirroring Web Pages.

FAILURE TRANSPARENCY

- **Enables the concealment of faults**
- **Allows users and applications to complete their tasks despite the failure of other components.**
- **Partial failure transparency is achievable but complete failure transparency is not possible**
- **Example: Database Management System**

MIGRATION TRANSPARENCY

- Allows the movement of information objects within a system without affecting the operations of users or application programs
- **Relocation Transparency:**
- Situation in which resources can be relocated *while they are being* accessed without the user or application noticing anything. In such cases, the system is said to support relocation transparency.

PERFORMANCE TRANSPARENCY

- **Allows the system to be reconfigured to improve performance as loads vary.**
- **Load should be evenly distributed among all the machines.**

SCALING TRANSPARENCY

- Allows the system and applications to expand in scale without change to the system structure or the application algorithms.
- Example: World-Wide-Web
- Example: Distributed Database

HETEROGENEITY

- **Variety and differences in**
 - Networks
 - Computer hardware
 - Operating systems
 - Programming languages
 - Implementations by different developers

SECURITY

- **In a distributed system, clients send requests to access data managed by servers, resources in the networks:**
 - Doctors requesting records from hospitals
 - Users purchase products through electronic commerce
- **Security is required for:**
 - 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

FAILURE HANDLING (FAULT TOLERANCE)

- **Hardware, software and networks fail!**
- **Distributed systems must maintain *availability* even at low levels of hardware/software/network *reliability*.**
- **Fault tolerance is achieved by**
 - recovery
 - redundancy

CONCURRENCY

- Components in distributed systems are executed in concurrent processes.
- Components access and update shared resources (e.g. variables, databases, device drivers).
- Integrity of the system may be violated if concurrent updates are not coordinated.

SCALABILITY

- Scalability of a system can be measured along at least three different dimensions
- **scalability with respect to size:** meaning that we can easily add more users and resources to the system.
- **geographically scalable :** system is one in which the users and resources may lie far apart.
- **Administratively scalable:** meaning that it can still be easy to manage even if it spans many independent administrative organizations.

SCALING TECHNIQUES

□ **Hiding communication latencies**

- Asynchronous communication
- Allocate more job to client machine

□ **Distribution**

- Distribution involves taking a component, splitting it into smaller parts, and subsequently spreading those parts across the system. An excellent example of distribution is the Internet Domain Name System (DNS)

□ **Replicate**

4. BASIC DESIGN ISSUES

□ **Specific issues for distributed systems:**

- Naming
- Communication
- Software structure
- System architecture
- Workload allocation
- Consistency maintenance

NAMING

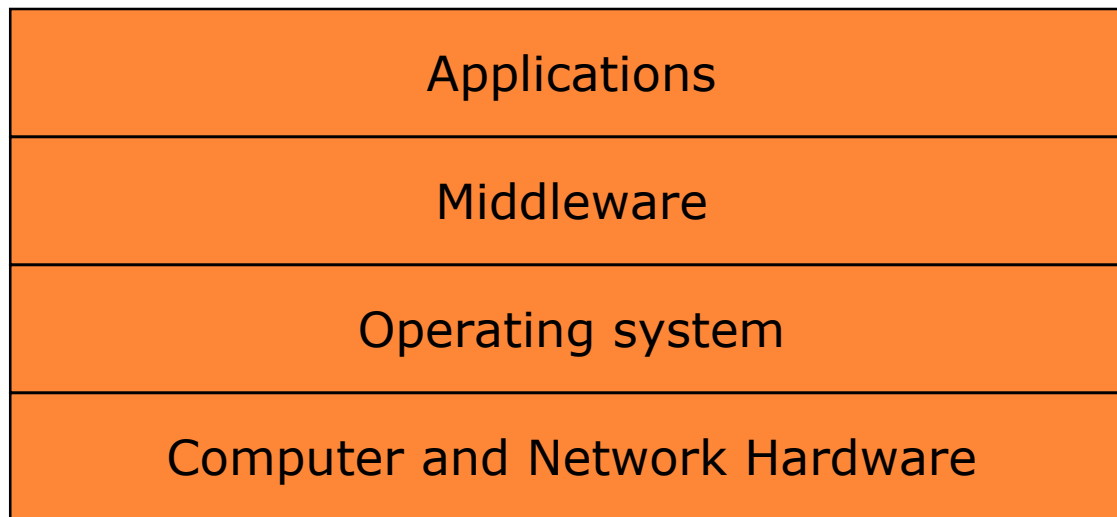
- **A name is resolved when translated into an interpretable form for resource/object reference.**
 - Communication identifier (IP address + port number)
 - Name resolution involves several translation steps
- **Design considerations**
 - Choice of name space for each resource type
 - Name service to resolve resource names to comm. id.
- **Name services include naming context resolution, hierarchical structure, resource protection**

COMMUNICATION

- **Separated components communicate with sending processes and receiving processes for *data transfer* and *synchronization*.**
- **Message passing: *send* and *receive* primitives**
 - synchronous or blocking
 - asynchronous or non-blocking
 - Abstractions defined: channels, sockets, ports.
- **Communication patterns: client-server communication (e.g., RPC, function shipping) and group multicast**

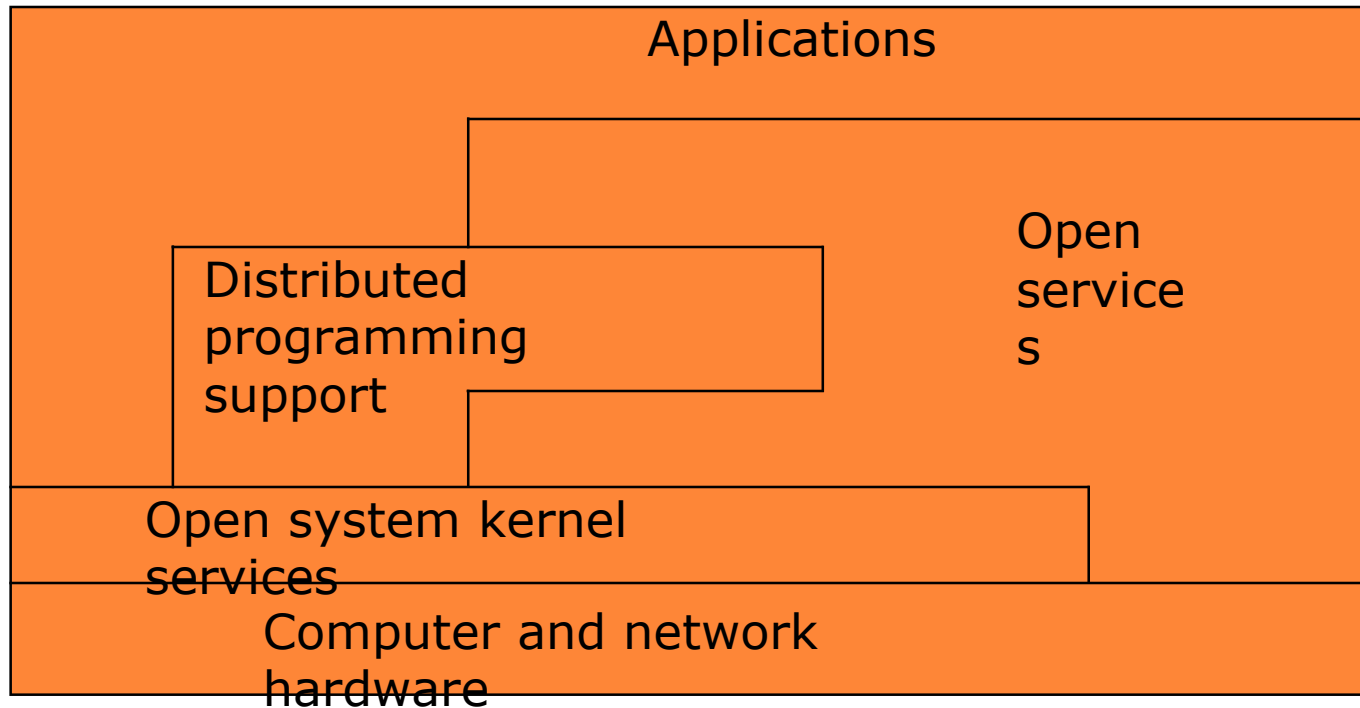
SOFTWARE STRUCTURE

- **Layers in centralized computer systems:**



SOFTWARE STRUCTURE

- **Layers and dependencies in distributed systems:**



Challenges

- Performance
- Concurrency
- Failures
- Scalability
- System updates/growth
- Heterogeneity
- Openness
- Multiplicity of ownership, authority
- Security
- Quality of service/user experience
- Transparency

ADVANTAGES OF DISTRIBUTED SYSTEM

- ❑ Information Sharing among Distributed Users
- ❑ Resource Sharing
- ❑ Extensibility and Incremental growth
- ❑ Shorter Response Time and Higher Output
- ❑ Higher Reliability
- ❑ Better Flexibility's in meeting User's needs
- ❑ Better price/performance ratio
- ❑ Scalability
- ❑ Transparency

DISADVANTAGES OF DISTRIBUTED SYSTEM

- Difficulties of developing distributed software
- Networking Problem
- Security Problems
- Performance
- Openness
- Reliability and Fault Tolerance

TYPES OF DISTRIBUTED SYSTEMS

1. Distributed computing systems,
2. Distributed information systems,
3. Distributed Pervasive systems.

DISTRIBUTED COMPUTING SYSTEMS

- Used for high-performance computing tasks.
- **Two subgroups**
 - **In cluster computing** the underlying hardware consists of a collection of similar workstations or PCs, closely connected by means of a high speed local-area network. each node runs the same operating system.
 - **grid computing**. This subgroup consists of distributed systems that are often constructed as a federation of computer systems, where each system may fall under a different administrative domain, and may be very different when it comes to hardware, software, and deployed network technology.

DISTRIBUTED INFORMATION SYSTEMS

Integrate applications into an enterprise-wide information system

- Transaction Processing Systems
- Enterprise Application Integration:

The more applications became decoupled from the databases they were built upon, the more evident it became that facilities were needed to integrate applications independent from their databases.

Application components should be able to communicate directly with each other and not merely by means of the request/reply behavior that was supported by transaction processing systems.

DISTRIBUTED PERVASIVE SYSTEMS

- **In distributed pervasive systems, are often characterized by being small,** battery-powered, mobile, and having only a wireless connection, although not all these characteristics apply to all devices.
- In this type of system instability is the default behavior.
- An important feature is the general lack of human administrative control. At best, devices can be configured by their owners, but otherwise they need to automatically discover their environment

Three requirements for pervasive applications:

1. Embrace contextual changes.
2. Encourage ad hoc composition.
3. Recognize sharing as the default.

Examples

- Home Systems
- Electronic Health Care Systems
- Sensor Networks

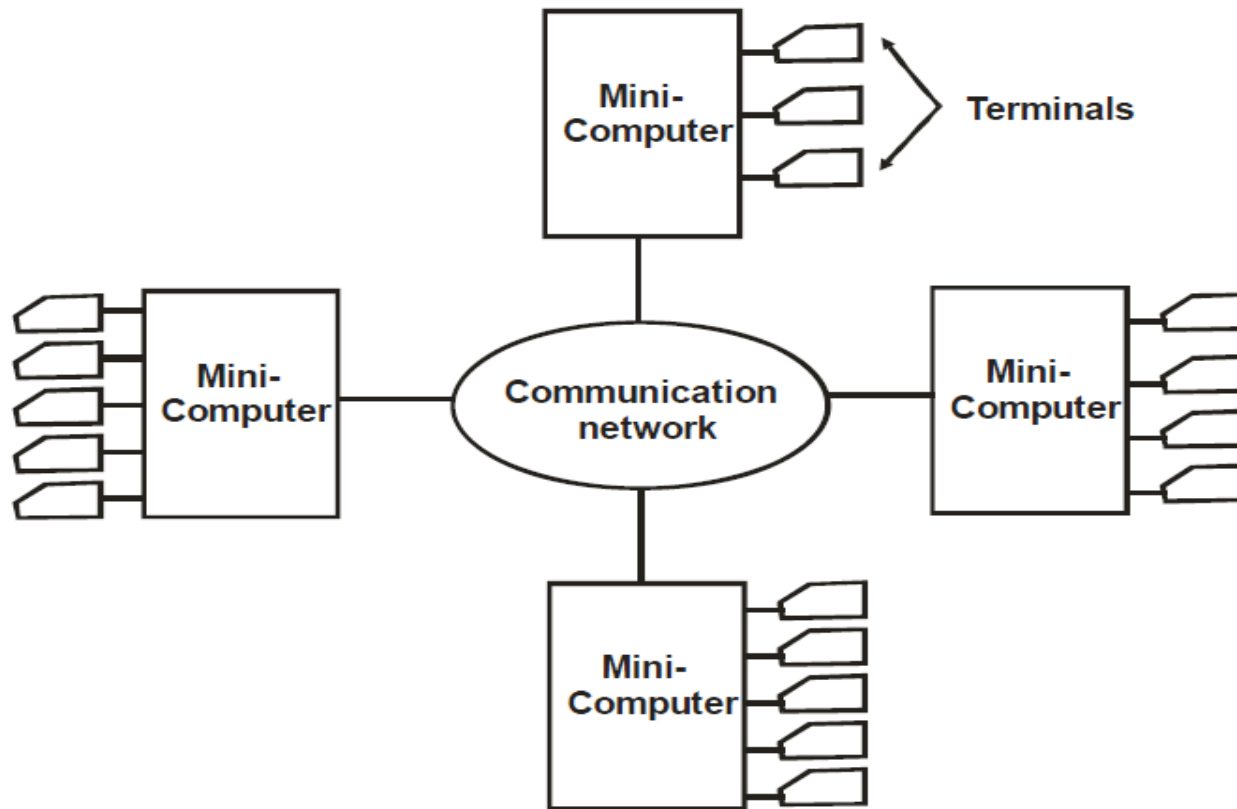
DCS MODELS

- **Minicomputer Model**
- **Workstation Model**
- **Workstation Server Model**
- **Processor Pool Model**
- **Hybrid Model**

MINICOMPUTER MODEL

- **Extension of the centralized time sharing system.**
- **few minicomputers, many terminals**
- **Remote access to other minicomputers.**
- **resource sharing with remote users is desired.**
- **Example: Early ERPAnet**

Minicomputer Model (Cont. ...)

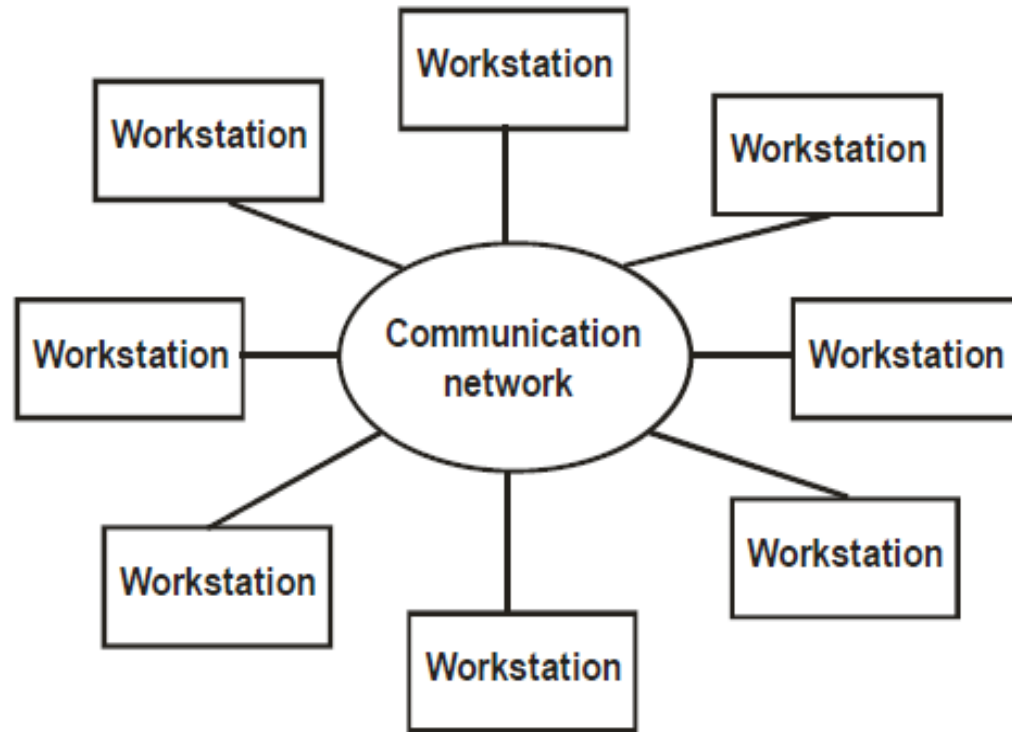


The distributed system based on minicomputer model

WORKSTATION MODEL

- Consists of several workstations.
- The workstations are independent computers with memory, hard disks, keyboard and console.
- A company's office or a university department may have several workstations scattered throughout a building or campus.
- User logs onto one of the workstations called his or her *home workstation*

Workstation Model (Cont. ...)



A distributed system based on the workstation model

Workstation Model (Cont. ...)

This model is not so simple to implement because several issues must be resolved. These issues are as follows:

- Finding Idle workstation
- How is a process transferred from one workstation to get it executed on another workstation
- a workstation that was idle until now and was being used

WORKSTATION-SERVER MODEL

- **few minicomputers and several workstations interconnected**
 - Diskfull Workstation
 - Diskless Workstation

- **types of services, such as database service and print service**

WORKSTATION-SERVER MODEL

(CONT. ...)

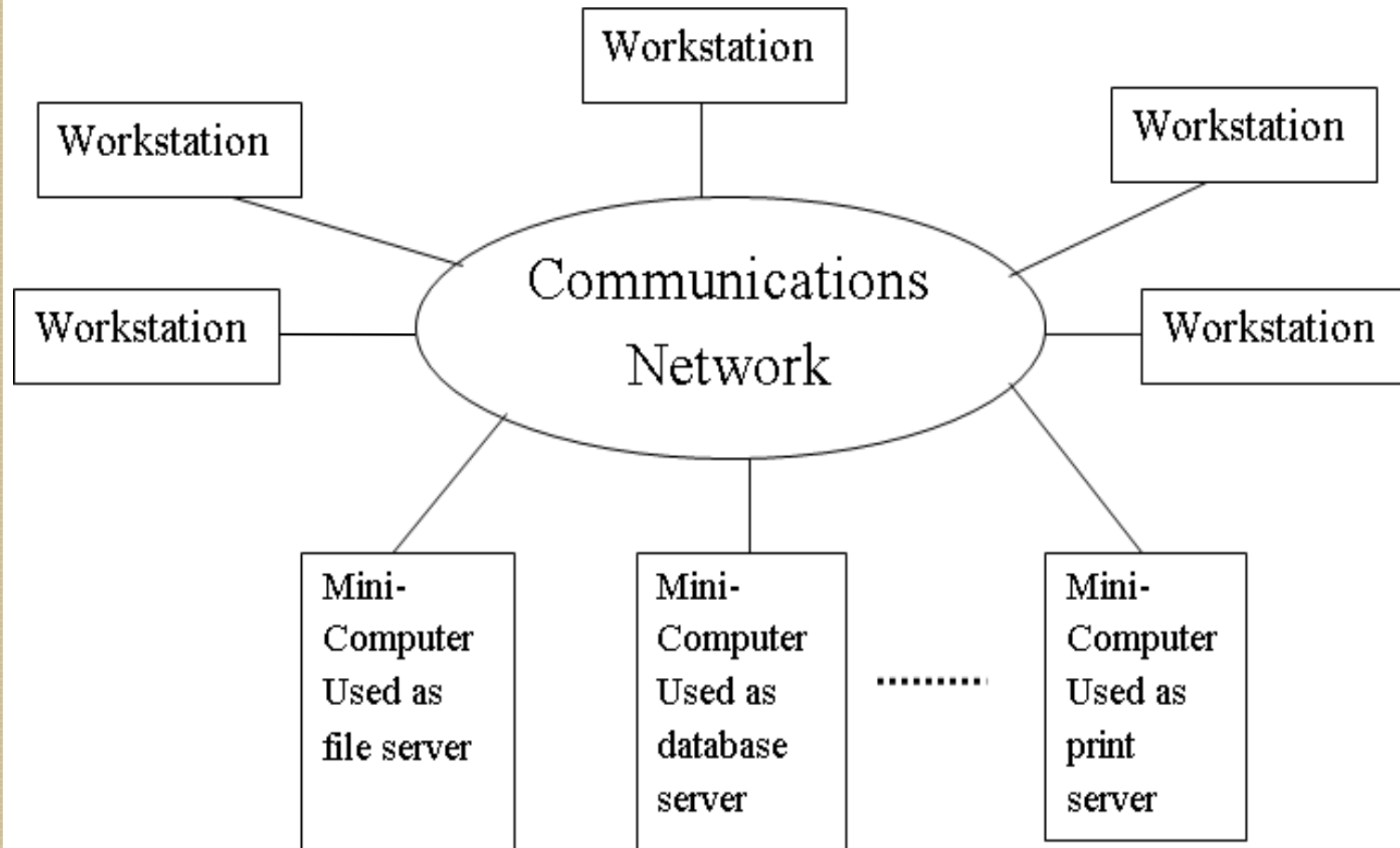


Fig 3: A Distributed System based on the workstation-server model

Workstation-Server Model (Cont. ...)

ADVANTAGES WORKSTATION SERVER MODEL

- ❑ **Cheaper**
- ❑ **Diskless workstations are also preferred to dishful workstations from a system maintenance point of view.**
- ❑ **Users have the flexibility to use any workstation**
- ❑ **The request-response protocol**
- ❑ **Guaranteed response time .**

PROCESSOR POOL MODEL

- large number of microcomputer and minicomputers
- based on the observation that most of the time a user does not need any computing power
- A user submits a Job for computations are temporarily assigned to the job by the run server
- the entire processing power is available for use by the currently logged-users

PROCESSOR POOL MODEL

(CONT. ...)

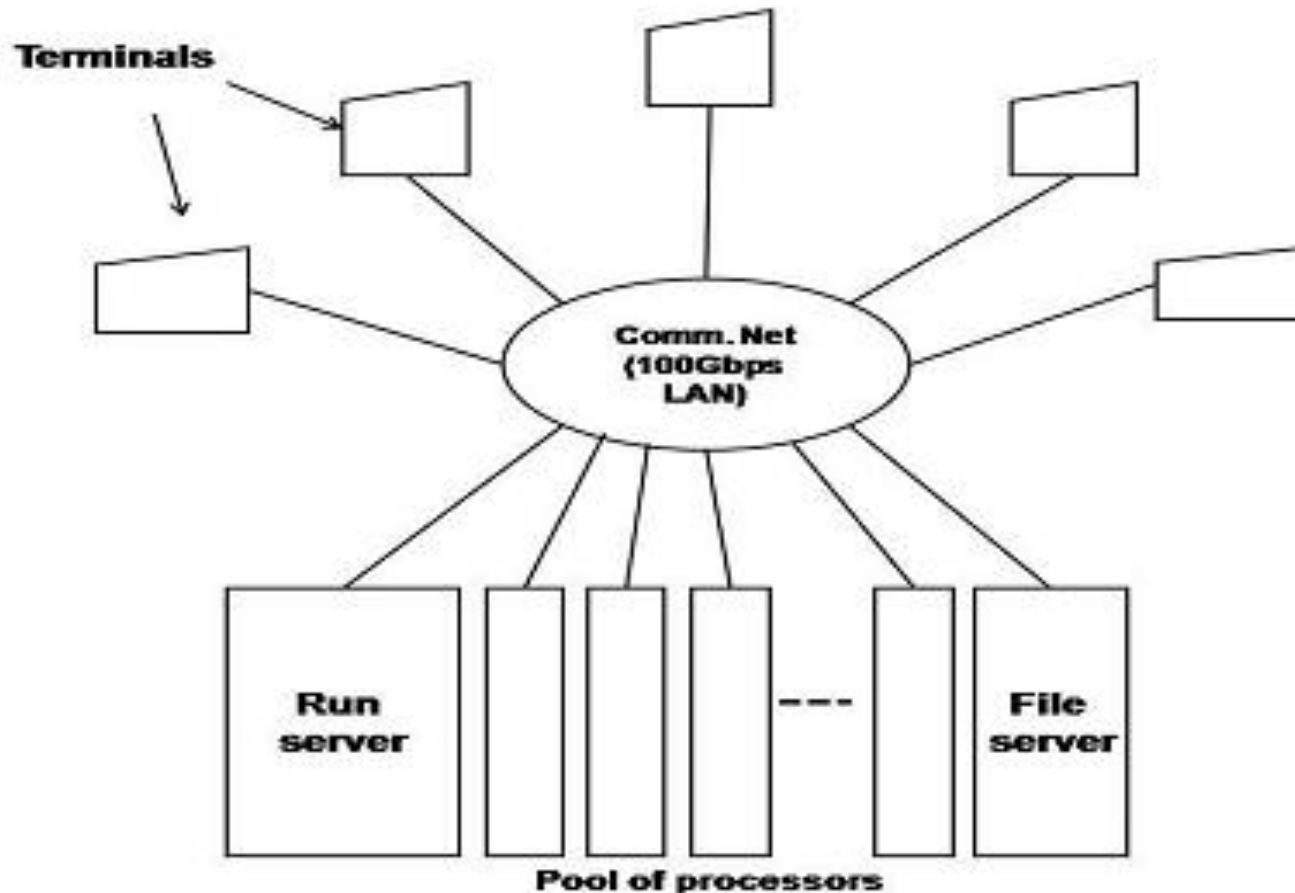


Figure 4: A distributed computing system based on processor-pool model

PROCESSOR POOL MODEL (CONT. ...)

□ **Advantages:**

- Better utilization of processing power
 - Better flexibility
 - More Scalable
-
- **Not suitable for interactive application**

HYBRID MODEL

- ❑ Based on the workstation-server model but with the addition of a pool of processors.
- ❑ Requires several computers concurrently for efficient execution.
- ❑ Gives guaranteed response to interactive jobs by allowing them to be processed on local workstations of the users.
- ❑ Process allocated dynamically for the computations.
- ❑ More expensive.

Thank You