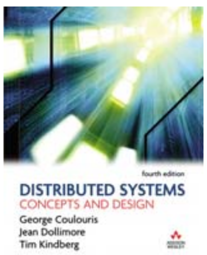
Introduction to Distributed Systems

Dr. Xiaobo Zhou



Distributed Systems:
Concepts and Design

Edition 4, © Addison-Wesley 2005

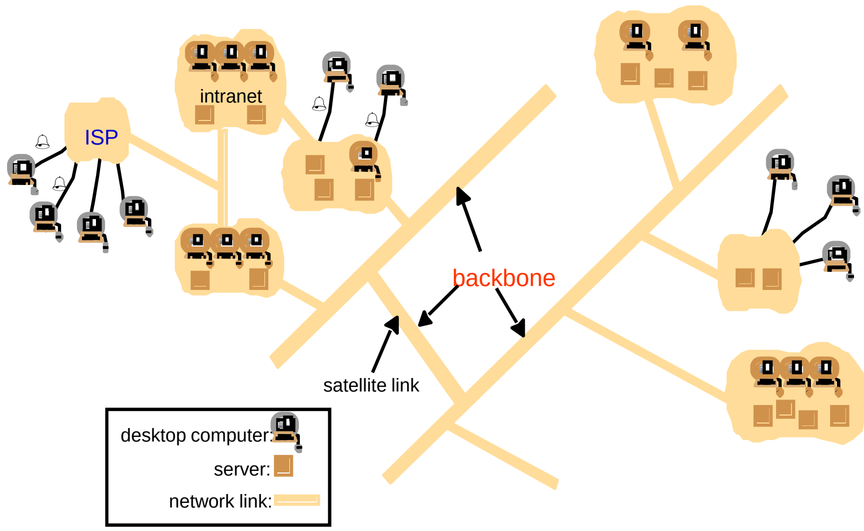
Definition of a Distributed System

- **G. Coulouris, J. Dollimore, T. Kindberg: A system in which hardware and software components located at networked computers communicate and coordinate their actions only by message passing**

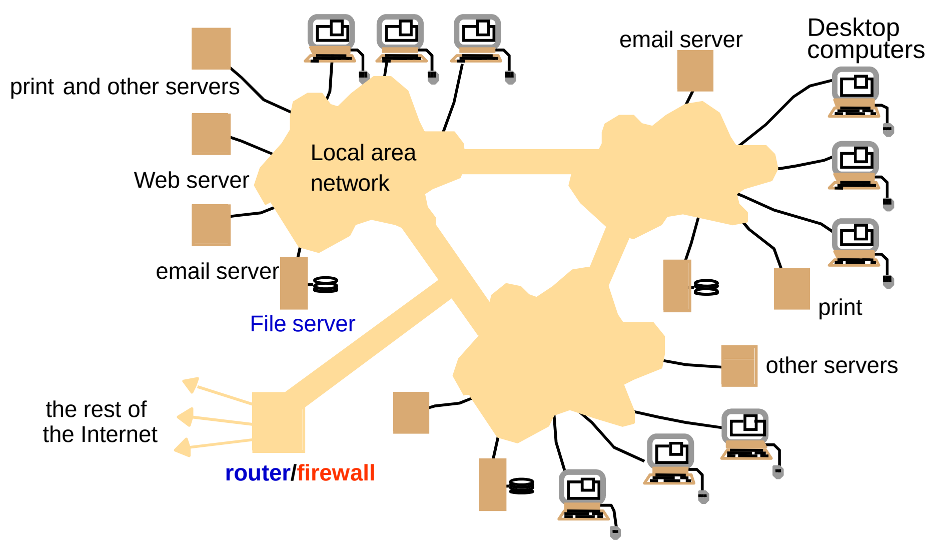
- **A. Tanenbaum and M. Steen: a collection of independent computers that appears to its uses as a single coherent system**

Core Objective:
Resource Sharing

The Internet: fostering distributed computing

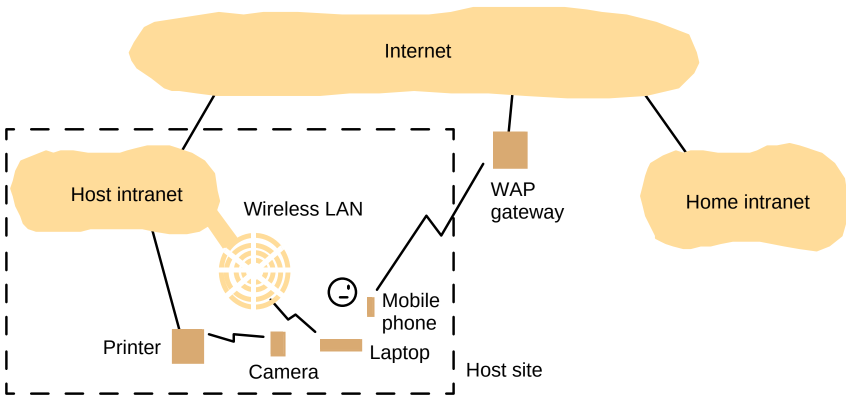


A Typical Intranet



Mobile Computing

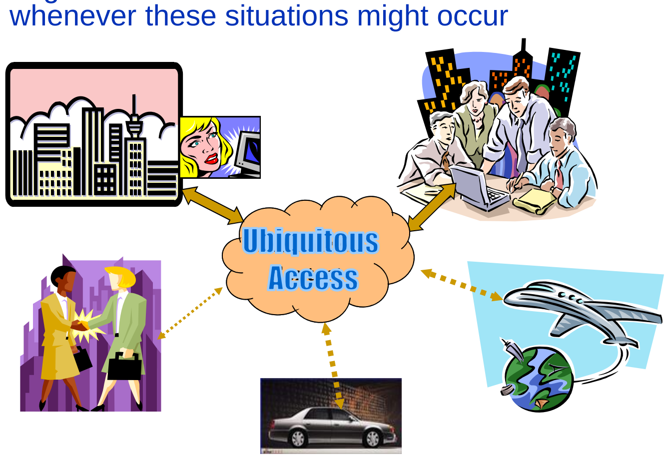
- **Mobile computing: increasing the capability of moving computing services with us; computing model does not change**



Portable and handheld devices in a distributed system

Ubiquitous/Pervasive Computing

- Anytime/Anyplace Computing
  - help organize and mediate social interactions wherever and whenever these situations might occur

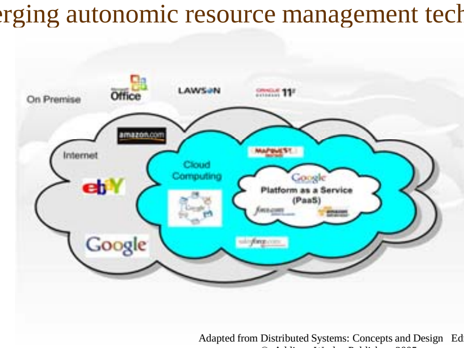


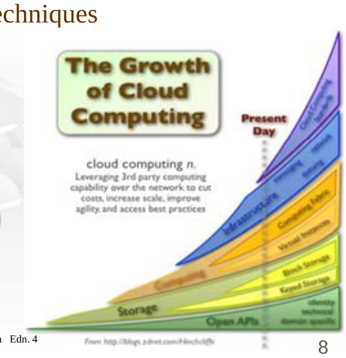
History and Evolution

- Changing Forces
  - Wireless telecommunication capabilities
  - Open networks
  - Continued increases in computing power
  - Improved battery technology
  - Emergence of autonomous software infrastructures
- Mobile Computing + Embedded Computing
  - Mobile computing: increasing the capability of moving computing services with us; computing model does not change
  - Embedded computing: increasing the capability of obtaining information from its environment and utilize it to dynamically build computing models, less level of mobility

Cloud Computing

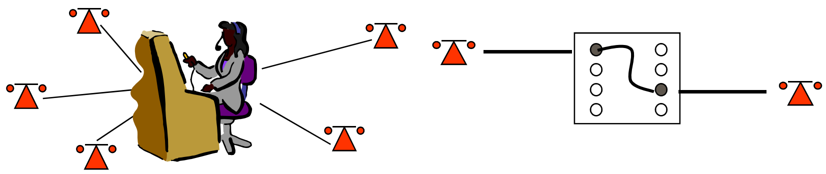
- Aims to offer IT capabilities over the Internet as an on-demand, pay-per-use service.
  - Data centers / high-density servers
  - Server virtualization
  - Service-oriented architecture
  - Pay-as-you-go business model
  - Emerging autonomic resource management techniques





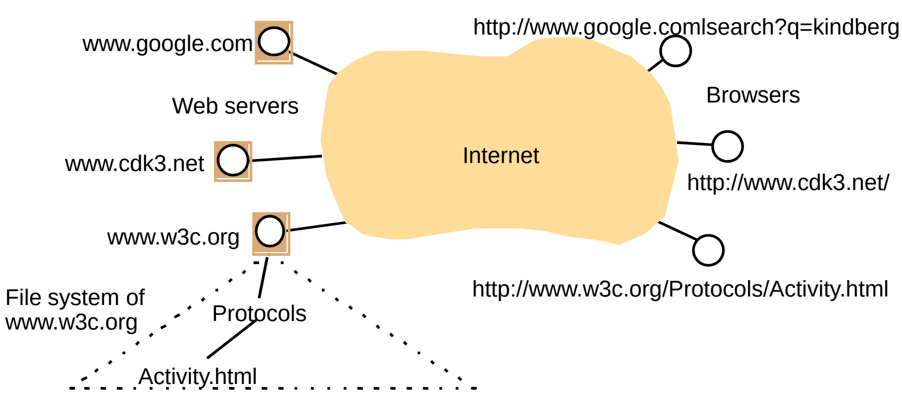
Autonomic Computing

- Seeks to improve computing systems with an aim of decreasing human involvement
  - Computing systems have reached a level of complexity where the human effort required to get the systems up and running and keeping them operational is getting out of hand. [ACM Computing Surveys 2008]
  - Think about the telephony in 1920s. Automatic branch exchanges were introduced to eliminate the need for human intervention.



Resource Sharing: Web Servers and Web Browsers

- **WWW: an evolving open system for publishing and accessing resources and services across the Internet**



Key Characteristics of Distributed Systems

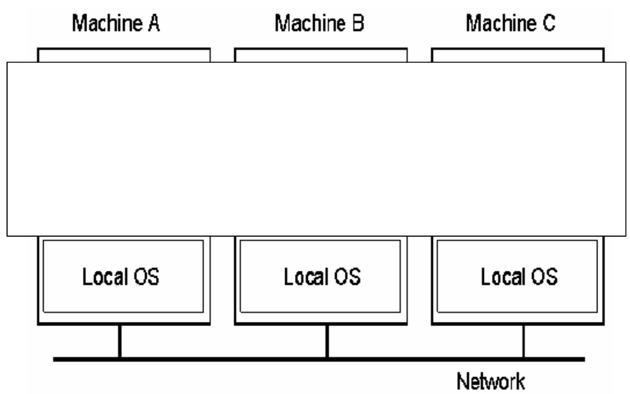
- **Raise three key characteristics of distributed systems**
- **Concurrency: concurrent program execution and resource access**
  - How to coordinate actions?
- **No global clock: the limits to the accuracy with which the computers in a network can synchronize their clocks**
  - The only communication is messaging passing
- **Independent failures: each component of the system can fail independently, leaving the others still running**
  - How to tell if the network has failed or become unusually slow?
  - How to know if a remote server crashed immediately?

Heterogeneity

- **Applies to all of the following:**
  - **Networks**
    - Internet protocols mask the differences between networks
  - **Computer hardware**
    - e.g. data types such as integers can be represented differently
  - **Operating systems**
    - e.g. the API to IP differs from one OS to another
  - **Programming languages**
    - data structures (arrays, records) can be represented differently
  - **Implementations by different developers**
    - they need agreed standards so as to be able to interwork

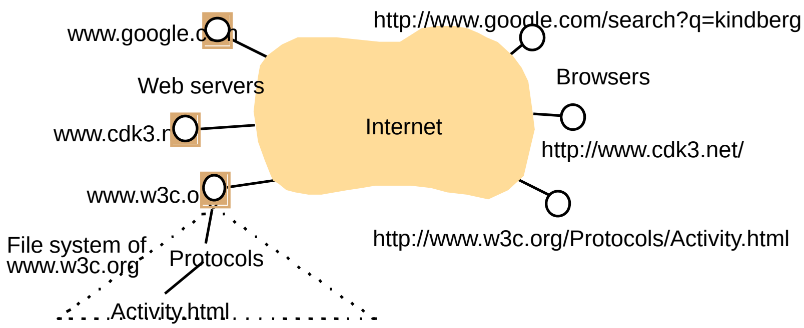
Challenges and Issues: Heterogeneity

- **Middleware: a software layer that provides a programming abstraction as well as masking the heterogeneity of the underlying networks, hardware, OS, and programming languages.**
  - Example: CORBA, Java RMI



Challenges and Issues: Openness

- **Openness: whether the system can be extended and re-implemented in various ways and new resource-sharing services can be added and made available for use by a variety of client programs**
  - specification and documentation of key software interface must be published
  - Example: Web, which can be extended and implemented in new ways without disturbing its existing functionality; an open distributed system



Challenges and Issues: Security

- **Resource (information) sharing leads to security issue**
  - Confidentiality (protection against unauthorized individuals)
  - Integrity (protection against alternation)
  - Availability (protection against interference with resource access)
  - Working encryption techniques for authentication
  - Still challenges: DoS/DDoS, Mobile code, Flash Crowds

Challenges and Issues: Scalability

- **Scalable: remain effective when there is a significant increase in the number of resources and the number of users**

Date	*Computers*	*Web servers*
1979, Dec.	188	0
1989, July	130,000	0
1999, July	56,218,000	5,560,866
2003, Jan	171,638,297	35,424,956

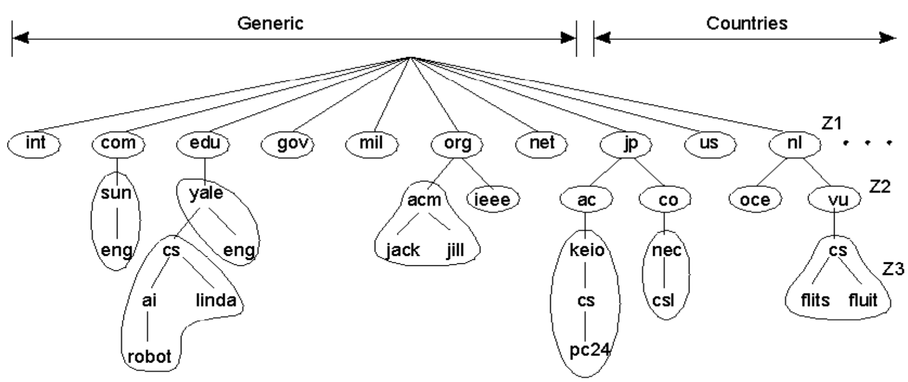
Date	*Computers*	*Web servers*	*Percentage*
1993, July	1,776,000	130	0.008
1995, July	6,642,000	23,500	0.4
1997, July	19,540,000	1,203,096	6
1999, July	56,218,000	6,598,697	12
2003, July		42,298,371	

Scalability Problems

Concept	Example
Centralized services	A single server for all users
Centralized data	A single on-line telephone book
Centralized algorithms	Doing routing based on complete information

- Controlling the cost of physical resources: linear O(n)?
- Controlling the performance loss: centralized data →hierarchic O(*log* n)
- Preventing software resources running out: IP address 32-bit →128-bit
- Avoiding performance bottlenecks: centralized vs. decentralized.

Scaling Techniques: Centralized vs Decentralized



An example of dividing the DNS name space into hierarchical zones

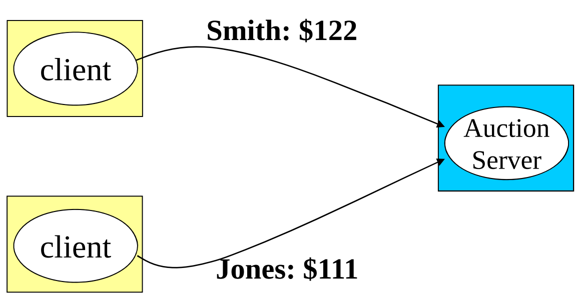
Raise more example(s) from network routing?

Challenges and Issues: Failure Handling

- Partial / independent failures
  - some components fail while others continue to function
- Detecting failures:
  - hard job, even impossible, why?
  - how to behave under suspected (not detected) failures?
- Masking failures: hid the detected failures
  - message retransmitting; data replication
- Tolerating failures and redundancy
  - multi-path routing in the Internet
  - replication in the Internet servers
- Failure recovery
  - consistency due to incomplete actions?

Challenges and Issues: Concurrency

- Resources are shared concurrently by clients
  - An object's operations must be synchronized so that its data remains consistent
    - E.g., Semaphores



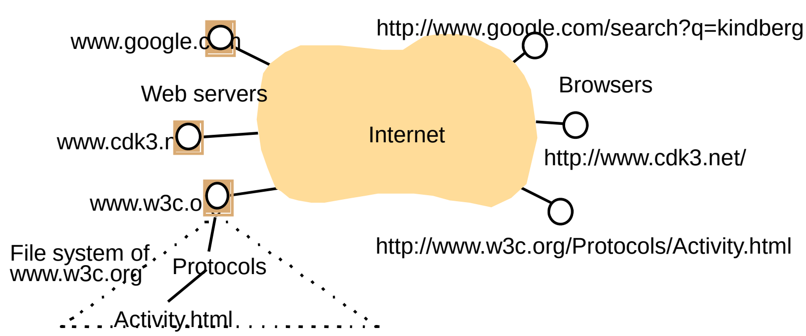
Challenges and Issues: Transparencies

System is perceived as a whole rather than a collection of independent components

Access transparency: enables local and remote resources to be accessed using identical operations.
Location transparency: enables resources to be accessed without knowledge of their location.
Concurrency transparency: enables several processes to operate concurrently using shared resources without interference between them.
Replication transparency: enables multiple instances of resources to be used to increase reliability and performance without knowledge of the replicas by users or application programmers.
Failure transparency: enables the concealment of faults, allowing users and application programs to complete their tasks despite the failure of hardware or software components.
Mobility/migration transparency: allows the movement of resources and clients within a system without affecting the operation of users or programs.
Performance transparency: allows the system to be reconfigured to improve performance as loads vary.
Scaling transparency: allows the system and applications to expand in scale without change to the system structure or the application algorithms.

Transparencies of URLs

- **Location transparency?**
- **Mobility transparency?**



Distributed vs. Centralized Systems

- Advantages of Distributed Systems
  - Resource sharing
  - Reliability
  - aggregate computing power
  - scalability / openness
- Disadvantage of Distributed Systems
  - Security
  - computing power per node is limited

What is a Parallel System?

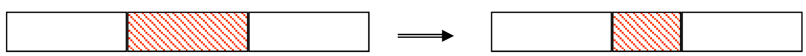
- A parallel system is a collection of ***processing elements*** that cooperate to solve large problems fast
- Some broad issues
  - resource allocation
    - How large a collection (how many elements)?
    - How powerful are the processing elements
  - data access, communication and synchronization
    - How do the elements cooperate and communicate?
    - How are data transmitted between processors?
    - What are the abstractions and primitives for cooperation?
  - Performance and Scalability
    - How does it all translate into performance
    - How does it scale?

Speedup - Amdahl's Law

ExTime after improvement = ExTime unaffected + Extime affected / amount of improvement

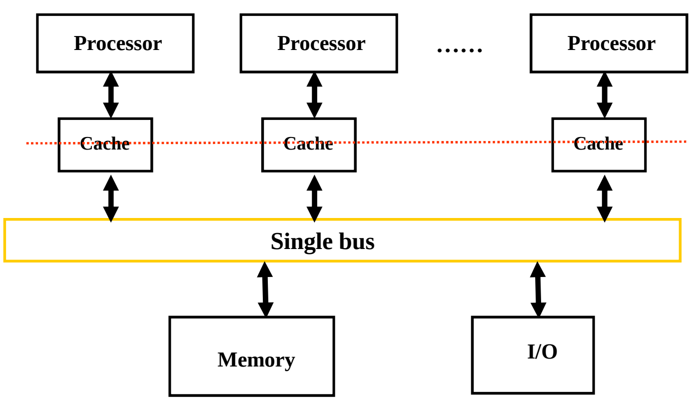
Speedup due to enhancement E:

$$\text{Speedup(E)} = \frac{\text{ExTime w/o E}}{\text{ExTime w/ E}} = \frac{\text{Performance w/ E}}{\text{Performance w/o E}}$$



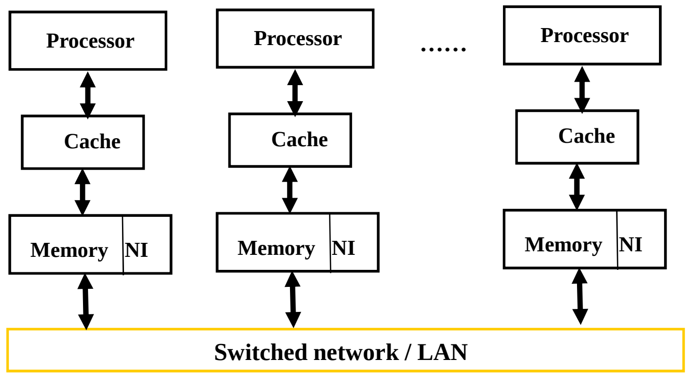
Shared-Memory Architectures

⌘Examples: Cray C90, SGI Power Challenge



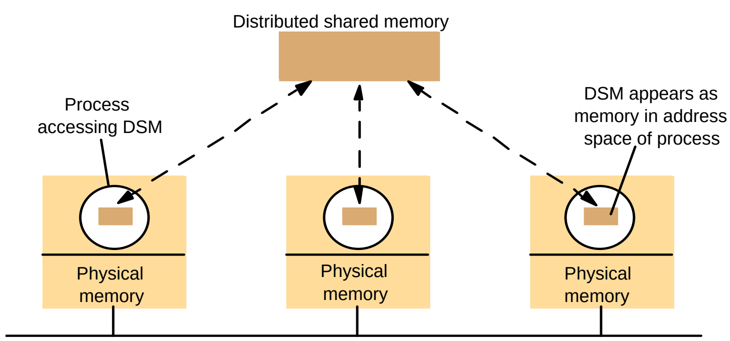
Distributed (Multi-Private) Memory Architectures

- Examples:
  - NUMA: Cray T3E , SGI Origin 2000 (connected on memory bus)
  - UMA: Sun Enterprise 10000 (connected on memory bus)
  - Clusters: IBM SP-2, UC Berkeley NOW (connected on I/O bus)



Distributed Shared Memory

- Build a "virtual" memory address over distributed multiple physical memories; an abstraction for data sharing between computers that do not share physical memory
  - Load/store vs. send/receive message passing
  - Run-time system support for transparent accesses

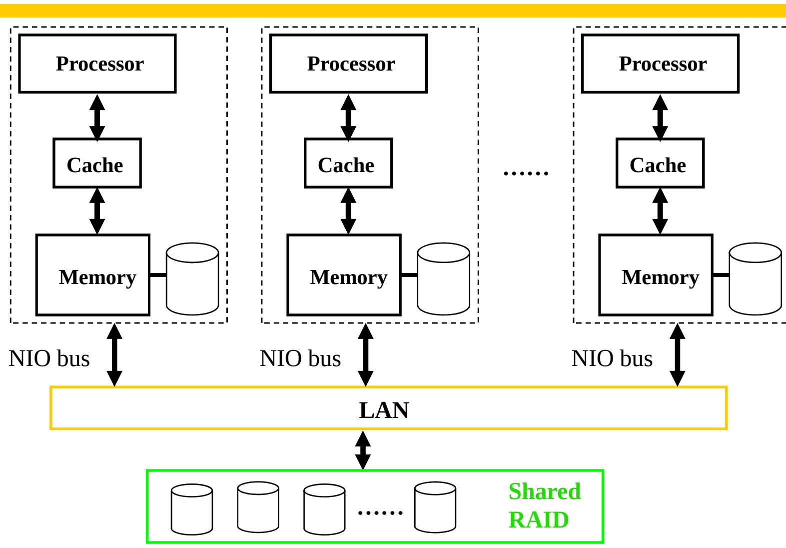


Modern Data Centers

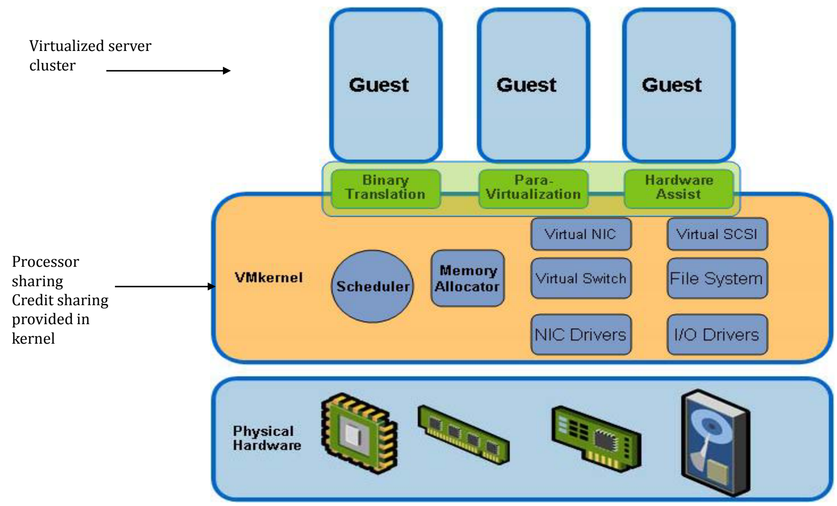
- Data centers are the next computing platform
  - Power consumption, distribution and cooling are major components in a data center's cost breakdown.

In 2009, we built a university prototype data center for research with a $1.25M equipment grant from the Air Force. The expenditure in racks, 24 HP G6 blade servers, 40 TB HP EVA storage area network with 10Gbps Ethernet and 8Gbps Fibre/iSCSI dual channels and the VMware licenses was about $0.6 million, while the expenditure in three APC InRow RP Air-Cooled and UPS equipments for maximum 40 kWs in the n+1 redundancy design was about another $0.6 million.

Cluster Architectures



Virtualized Server Clusters



Courtesy: adapted from VMware.com

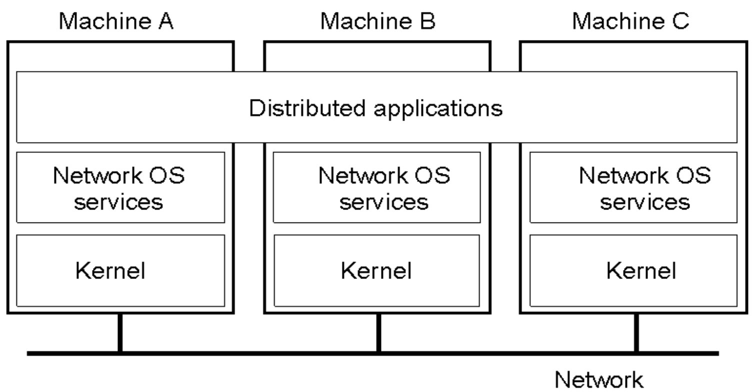
Distributed Operating Systems

- **Requirements**
  - **Provide user with a single coherent computer system**
  - **Hide distribution of resources**
  - **Mechanisms for resource protection**
  - **Secure communication**

- **Definition of Distributed OS**
  - **Look to user like ordinary centralized OS, but runs on multiple and independent CPUs**
    - **use of multiple processors is invisible**
    - **user views system as a virtual uniprocessors**

Network Operating System

- A collection of OSs of computers connected through a network incorporating modules to provide access to remote resources
  - Users are aware of file locations; remotely log-in



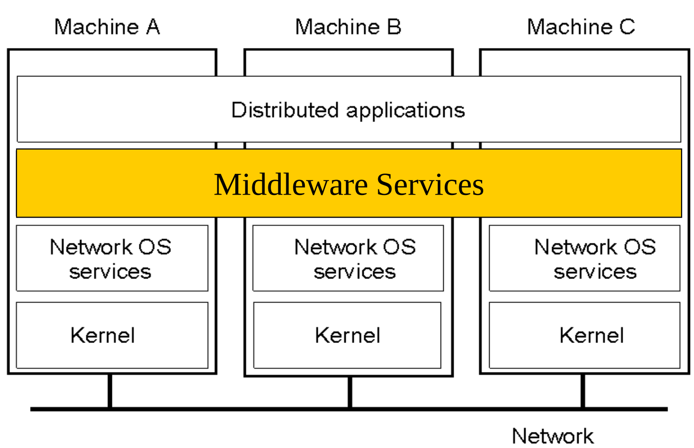
1/20/2011

Adapted from Distributed Systems: Concepts and Design Edn. 4
© Addison-Wesley Publishers 2005

33

Distributed OS: Positioning Middleware

- General structure of a distributed system as middleware.



1/20/2011

Adapted from Distributed Systems: Concepts and Design Edn. 4
© Addison-Wesley Publishers 2005

34

Distributed vs. Networked Operating Systems

- Transparency
  - how aware are users of the fact that multiple computers are being used?
- Network OS
  - Users are aware where resources are located
  - Network OS is built on top of centralized OS
  - Handles interfacing and coordination between local OSs
- Distributed OS
  - Designed to control and optimize operations and resources in distributed system, but giving a virtual single system to users

Learning Summary

- the key characteristics of distributed systems
  - concurrency
  - independent failure of components
  - lack of a global clock
- the key challenges in distributed systems
  - heterogeneity
  - openness
  - security
  - scalability
  - failure handling
  - concurrency
  - transparency

More reading

- **Chapter 1**
  - Coulouris, Dollimore and Kindberg, Distributed Systems: Concepts and Design, Edition 4, © Addison-Wesley 2005