CS550 “Advanced Operating Systems”

Instructor: **Professor Xian-He Sun**

- Email: sun@iit.edu
- Office: SB235C
- Class time: Monday, Wed., 3:15pm-4:30pm, **HH MEZZANINE**
- Office hour: Monday, Wednesday, 4:45-5:45pm
- http://www.cs.iit.edu/~sun/cs550.html

- TA: Mr. Hua Xu, Email: hxu40@hawk.iit.edu
- Office Hour: 11am - 12pm, Tuesday
- meet.google.com/kfp-pysg-cat
- Office Hour: 12pm - 1pm, Tuesday & Thursday
  meet.google.com/bnn-eqao-htg

- Blackboard*:*
  - http://blackboard.iit.edu
- Substitute lecturer:
  - Anthony Kougkas, assistant research professor
  - akougkas@hawk.iit.edu

Research Related Term Projects (see BB)

- Keith Bateman kbateman@hawk.iit.edu
- Jaime Cernuda Garcia jcernudagarcia@hawk.iit.edu
- Neeraj Rajesh nrajesh@hawk.iit.edu
- Meng Tang mtang11@hawk.iit.edu
- Jie Ye jye20@hawk.iit.edu
- Izzet Yildirim iyildirim@hawk.iit.edu

Storage systems, Parallel IO, IO & Machine Learning, Database Buffering

- **Luke Logan llogan@hawk.iit.edu**
  - Advanced OS

- **Xiaoyang Lu** xlu40@hawk.iit.edu
  - Heterogeneous Memory System

- **Any other** distributed system related topics

Term Project

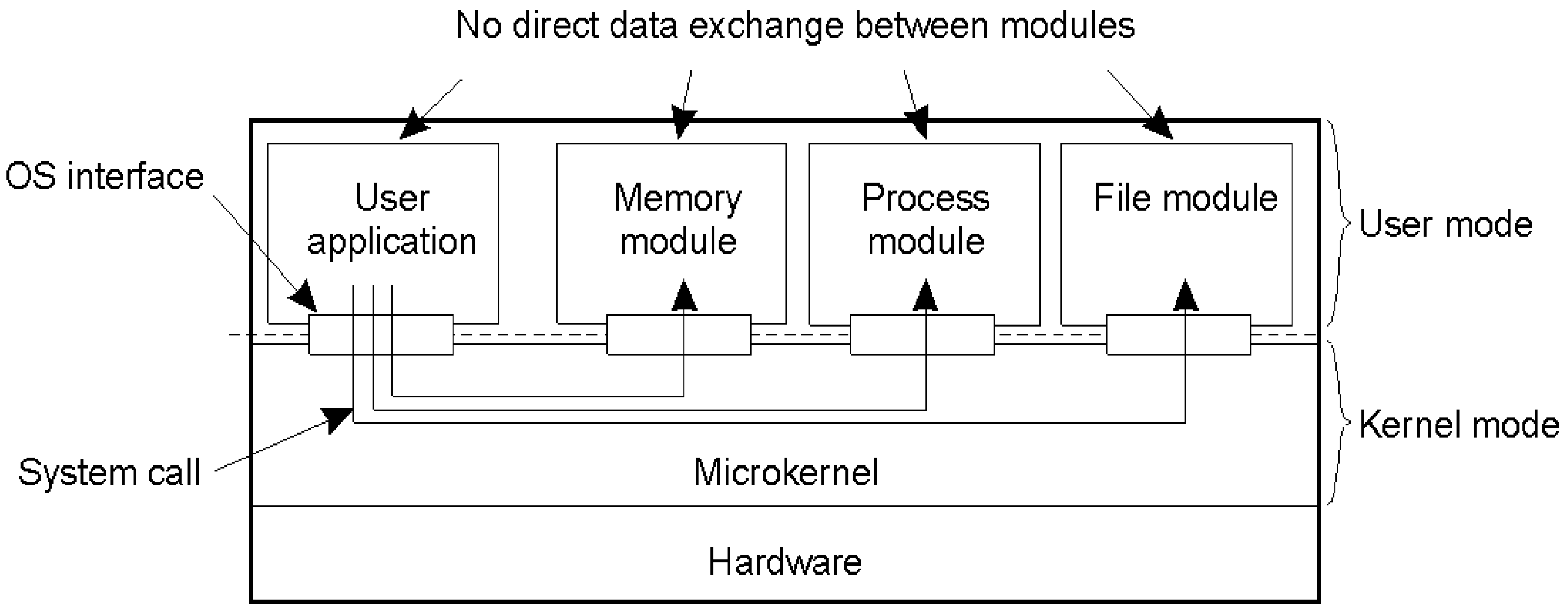
- See http://www.cs.iit.edu/~sun/html/report2.html
- A **two-page** project proposal due by **Jan. 29, 2024** (format and examples)
- Final project report is due on April 25, 2024

- Example topics
  - Study and practice of some middleware programming-environment, software packages, applications.
  - Study and analyze some distributed environment, architectures, and network structures.
  - Study the distributed solution of certain application package, algorithm, and system software.
  - Performance metric, measurement, and benchmark.
  - Study and practice of some visualization tools.
  - Survey of certain topics.
  - Any other topics that are relevant to this course.

Uniprocessor Operating Systems

Microkernel architecture

- Small kernel
- user-level servers implement additional functionality

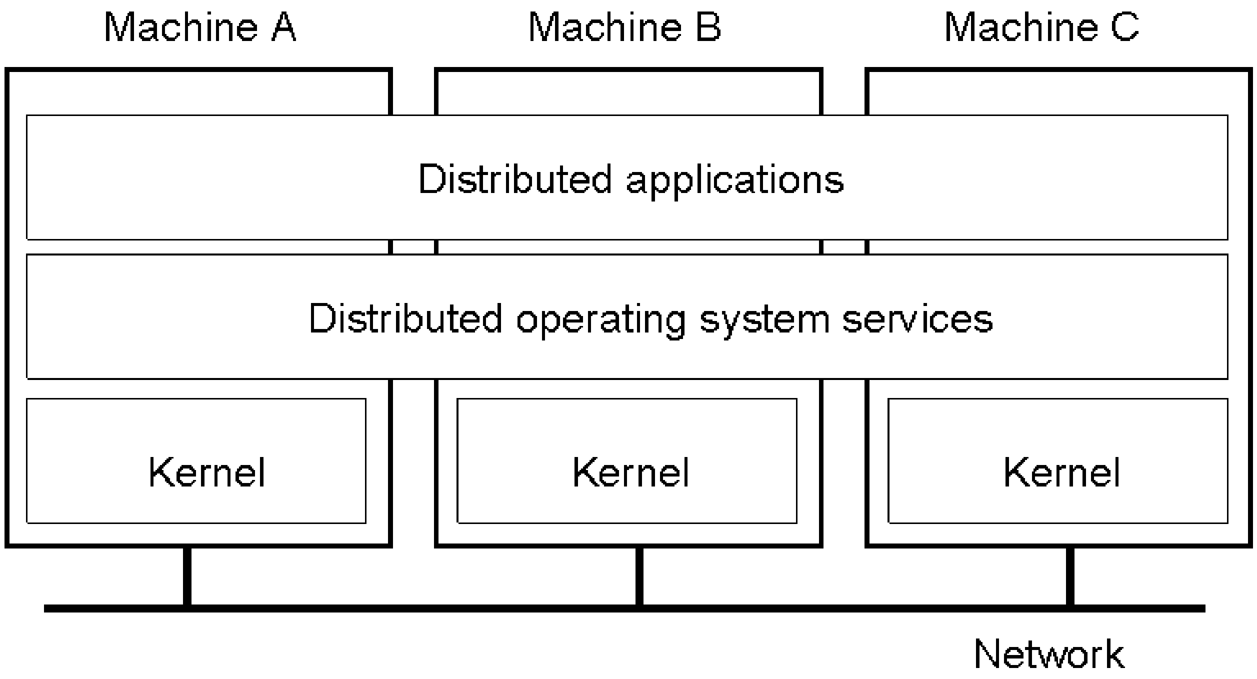


Multiprocessor Operating Systems

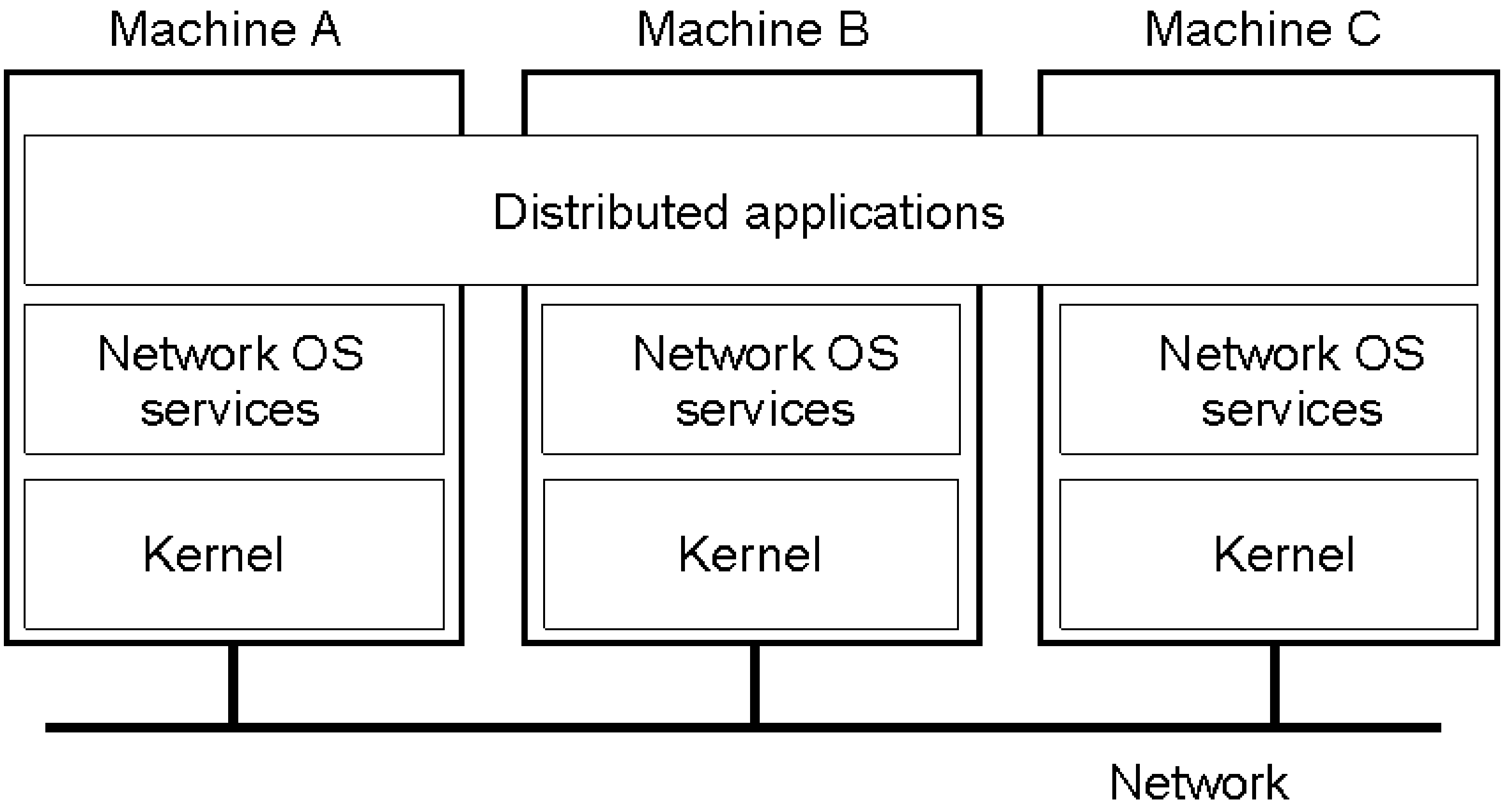
- Like a uniprocessor operating system
- Manage multiple CPUs transparently to the user
- Each processor has its own hardware cache
  - Maintain consistency of cached data
  - Scalability issues
- Shared variable versus message passing

Multicomputer Operating Systems

- More complex than multiprocessor OS
  - Because communication must be through explicit message passing

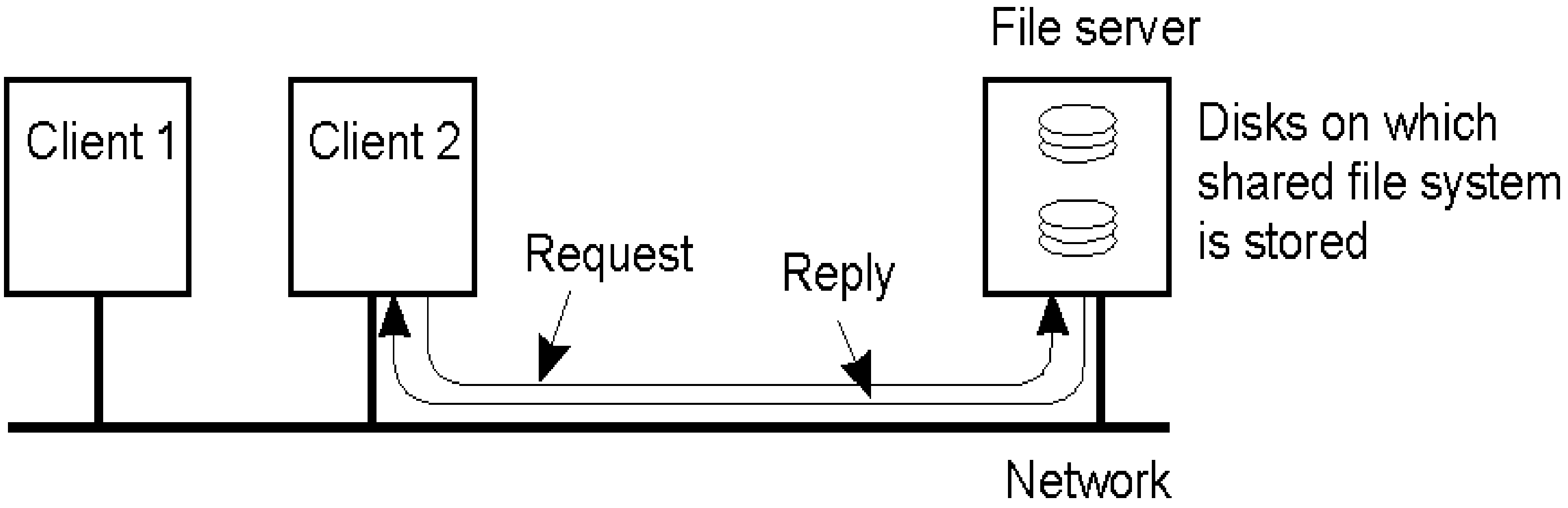


Network Operating System



Network Operating System

- Employs a client-server model
  - Minimal OS kernel
  - Additional functionality as user processes



Network-Operating Systems

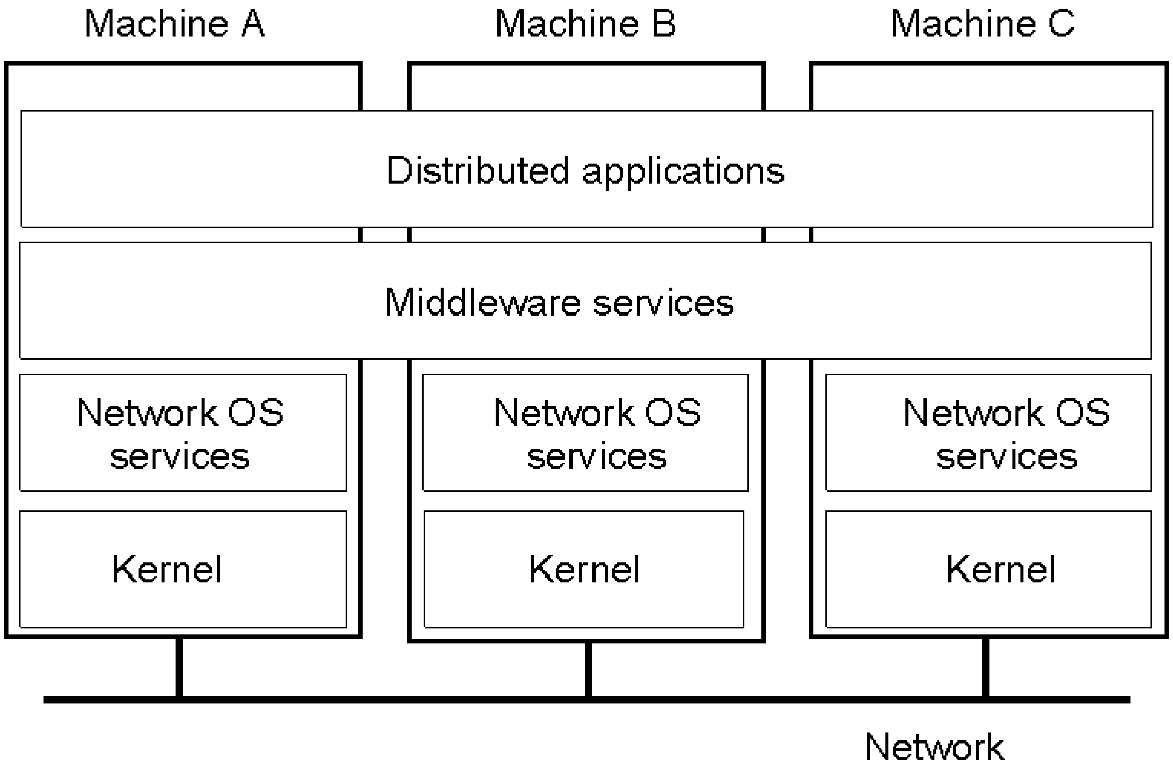
- Users are aware of multiplicity of machines. Access to resources of various machines is done explicitly by
  - Remote logging into the appropriate remote machine.
  - Transferring data from remote machines to local machines, via the File Transfer Protocol (FTP) mechanism.

Distributed Operating System

- Users not aware of multiplicity of machines.
- Manages resources in a distributed system
  - Seamlessly and transparently to the user
- Looks to the user like a centralized OS
  - But operates on multiple independent CPUs
- Provides transparency
  - Location, migration, concurrency, replication,…
- Presents users with a virtual uniprocessor

Middleware-based Systems

- General structure of a distributed system as middleware.



Any Questions?

Distributed Systems

- Definition of distributed systems
- Design goals
- Examples

Distributed Systems

- What is a distributed system?

"A distributed system is a collection of autonomous computing elements that appears to its users as a single coherent system."

-From our textbook, A. Tanenbaum

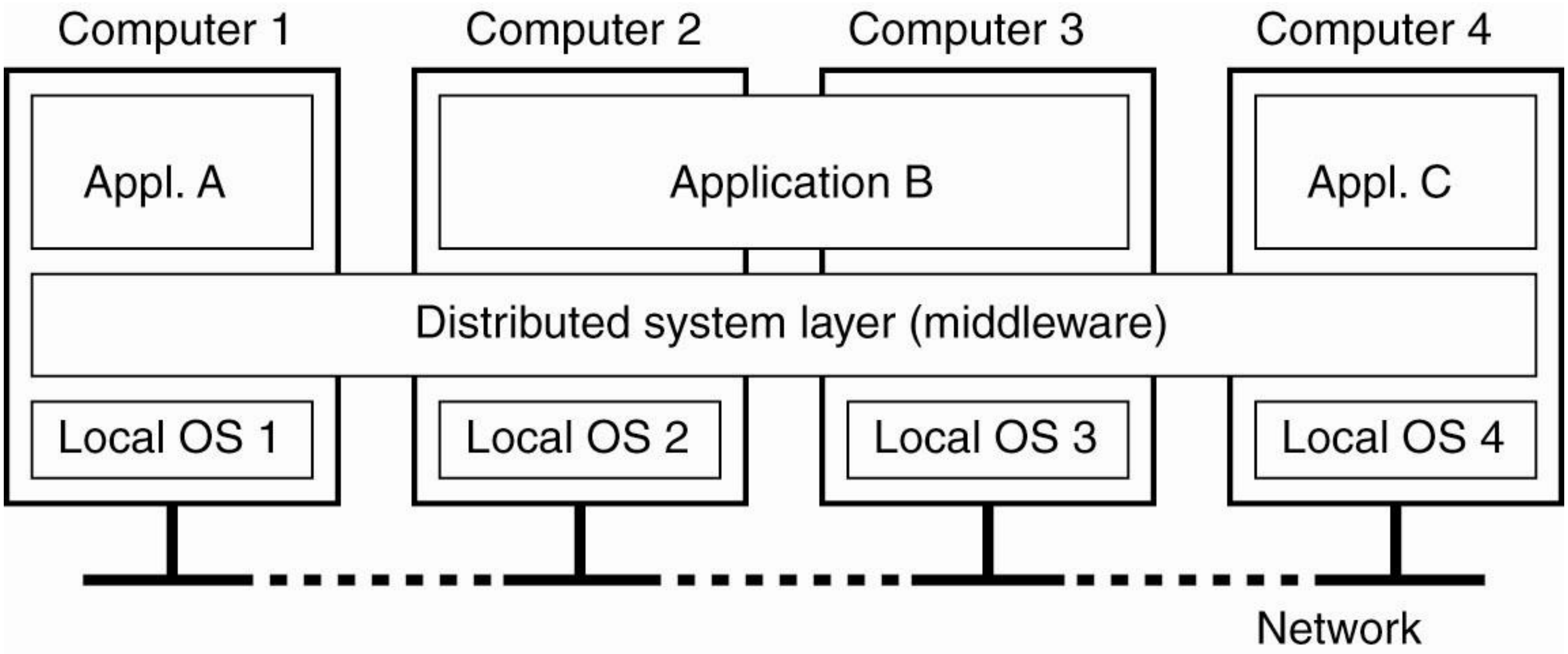
Distributed Systems

- Collection of autonomous computing elements:
  - Nodes can act **independently** from each other
  - Each one will have its own notion of **time** (lack of global clock)
  - Managing **group membership** can be exceedingly difficult
    - Open group: any node can join the distributed system
    - Closed group: only the members of that group can communicate with each other, and a separate mechanism is needed to let a node join or leave the group
  - Organization of the collection:
    - Overlay network
      - Structured and Unstructured
      - Always connected
      - e.g., Peer-to-peer (P2P) networks

Distributed Systems

- Single coherent system:
  - Appears as **one**
    - I.e., in a single coherent system the collection of nodes operates the same, no matter where, when, and how interaction between a user and the system takes place.
  - Distribution **transparency**
    - I.e., The end user would not be able to tell exactly on which computer a process is currently executing, or even perhaps that part of a task has been spawned off to another process executing somewhere else.

Distributed Systems



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

Key Characteristics

- Support for resource sharing
- Openness
- Concurrency
- Scalability
- Fault tolerance (reliability)
- Transparency

Resource Sharing

- Share hardware,software,data and information
- Hardware devices
  - Printers,disks,memory,…
- Software sharing
  - Compilers,libraries,toolkits,…
- Data
  - Databases,files,…

Openness

- Definition?
- Hardware extensions
  - Adding peripherals,memory,communication interfaces…
- Software extensions
  - Operating systems features
  - Communication protocols

Concurrency

- In a single system several processes are interleaved
- In distributed systems: there are many systems with one or more processors
  - Many users simultaneously invoke commands or applications
  - Many server processes run concurrently, each responding to different client request

Scalability

- Scale of system
  - Few PCs servers ->dept level systems->local area networks->internetworked systems->wide are network…
  - Ideally, system and application software should not change as systems scales

- Scalability depends on all aspects
  - Hardware
  - Software
  - networks

Fault Tolerance

- Definition?
- Two approaches:
  - Hardware redundancy
  - Software recovery

- In distributed systems:
  - Servers can be replicated
  - Databases may be replicated
  - Software recovery involves the design so that state of permanent data can be recovered

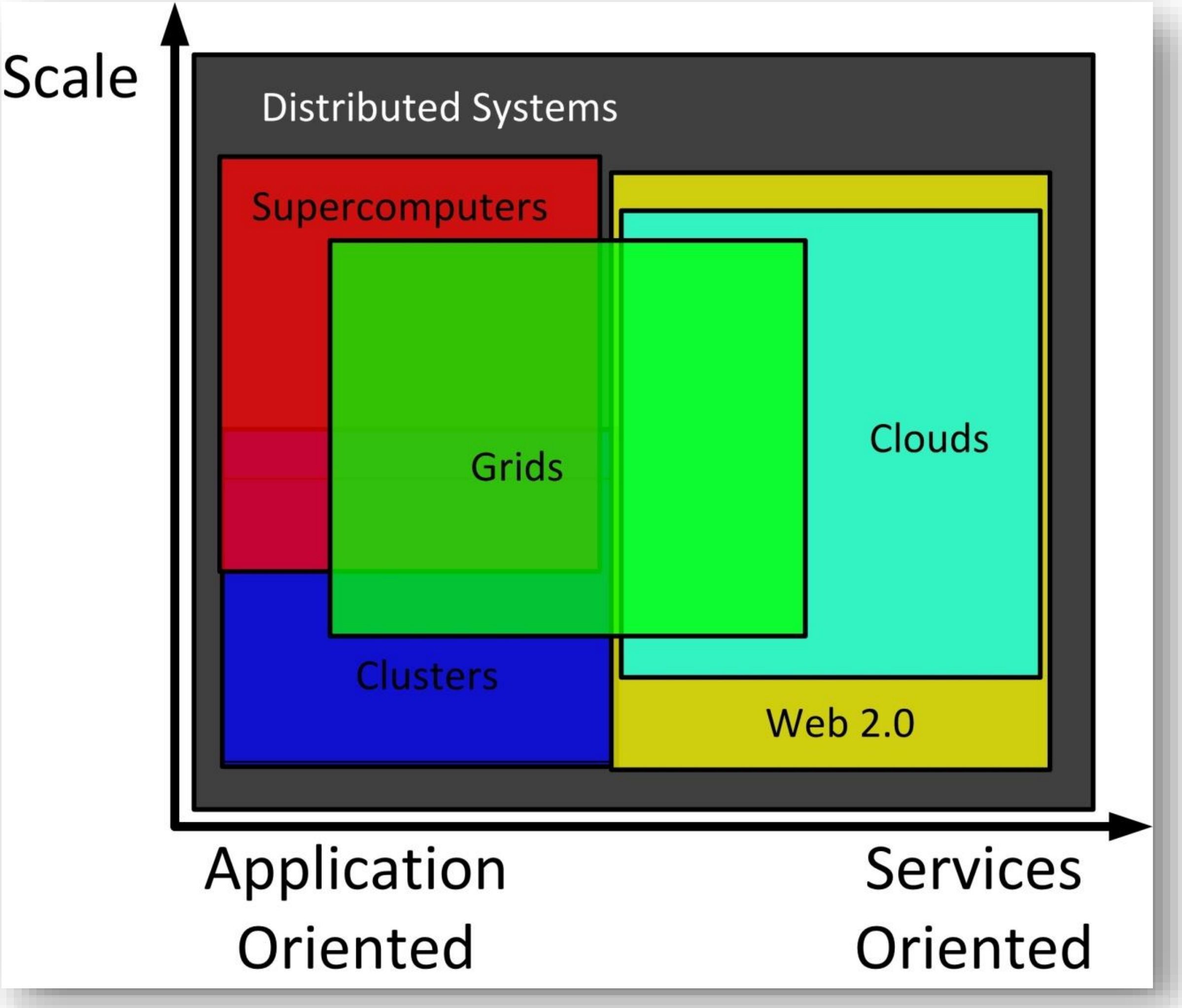
Transparency in a Distributed System

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	Means that users do not know whether a replica or a master provides a service.
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 on disk

Pitfalls When Developing Distributed Systems

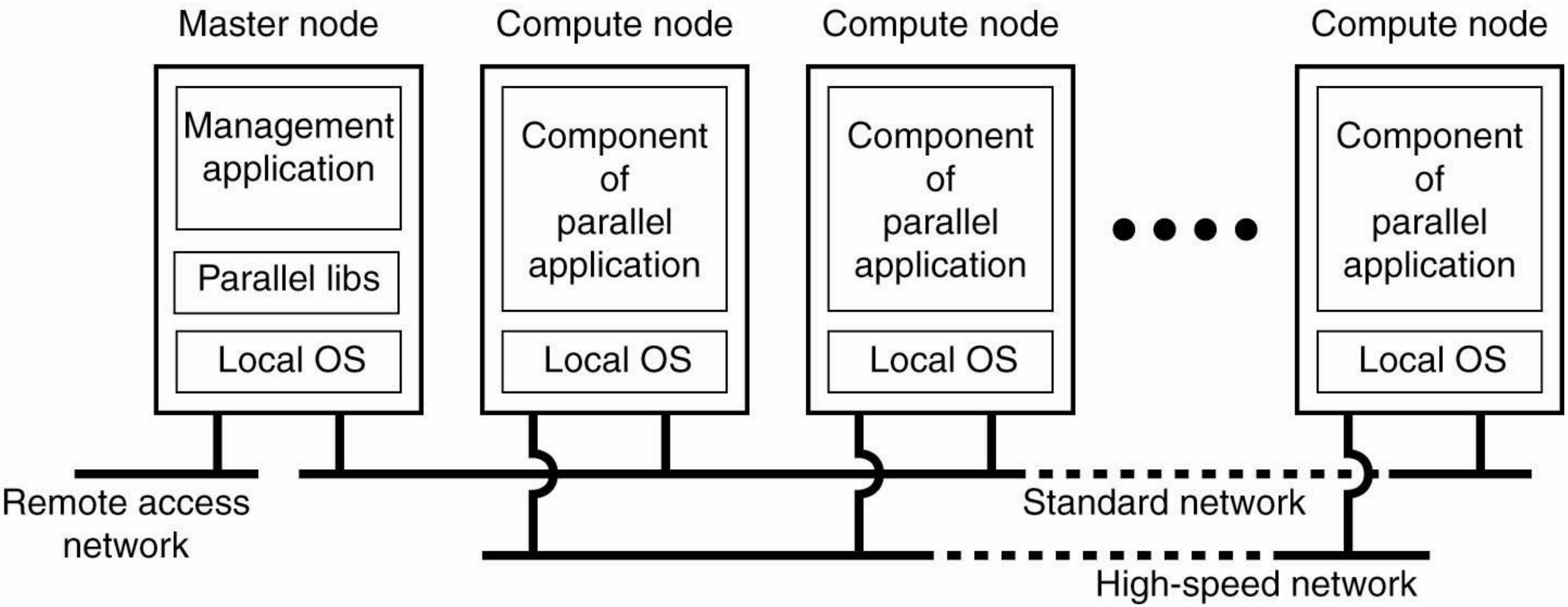
- False assumptions made by first time developer:
  - The network is reliable.
  - The network is secure.
  - The network is homogeneous.
  - The topology does not change.
  - Latency is zero.
  - Bandwidth is infinite.
  - Transport cost is zero.
  - There is one administrator.

Distributed Systems:
Supercomputers, Clusters, Grids, Clouds

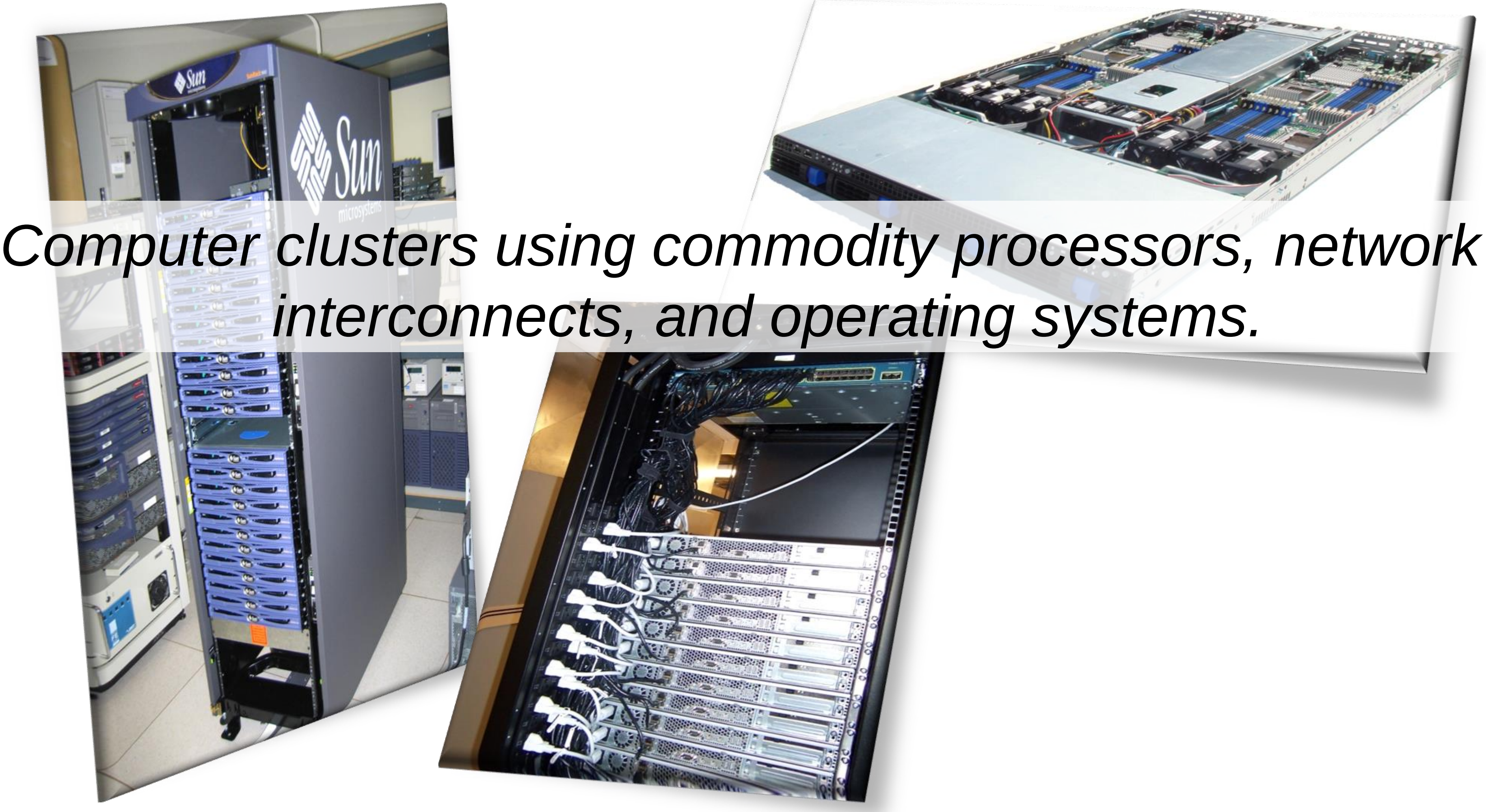


[GCE08] "Cloud Computing and Grid Computing 360-Degree Compared"

Cluster Computing Systems



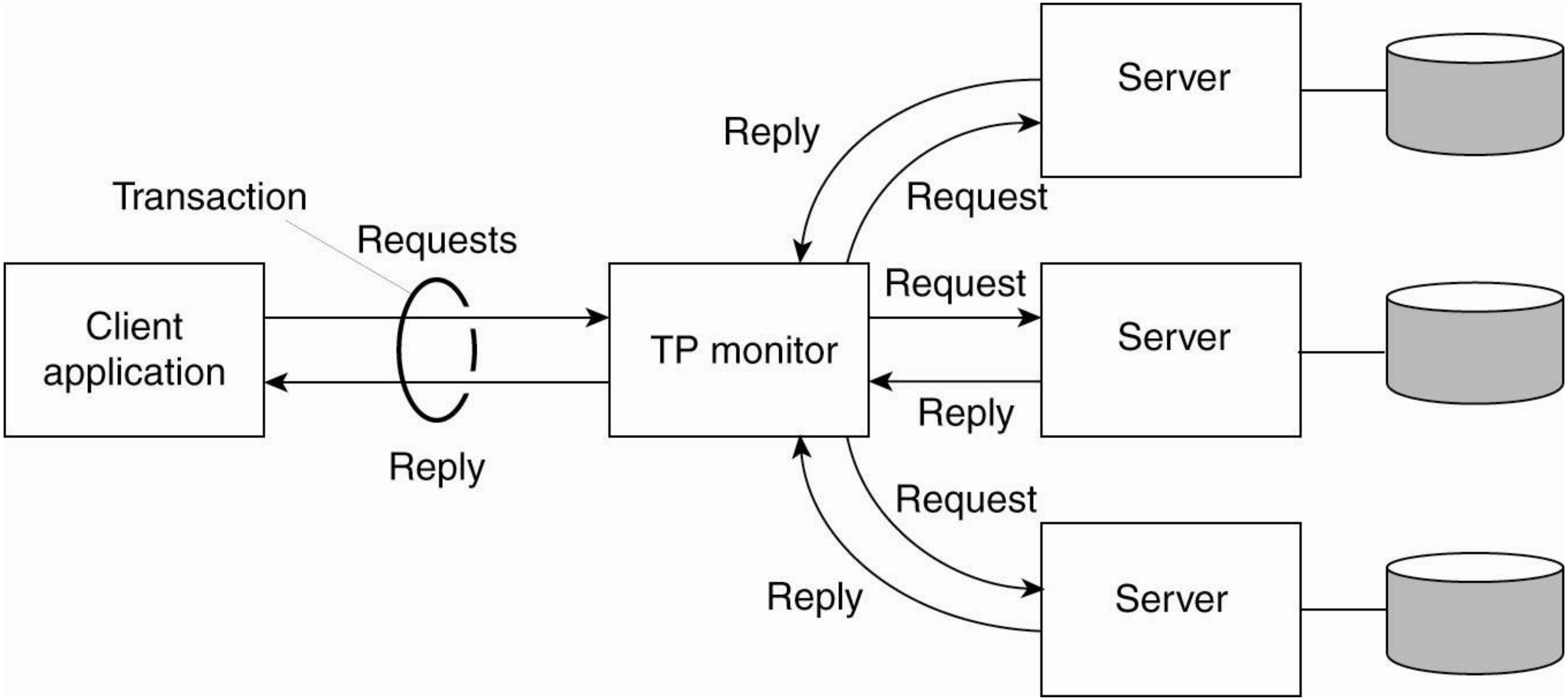
Cluster Computing Systems



Computer clusters using commodity processors, network interconnects, and operating systems.

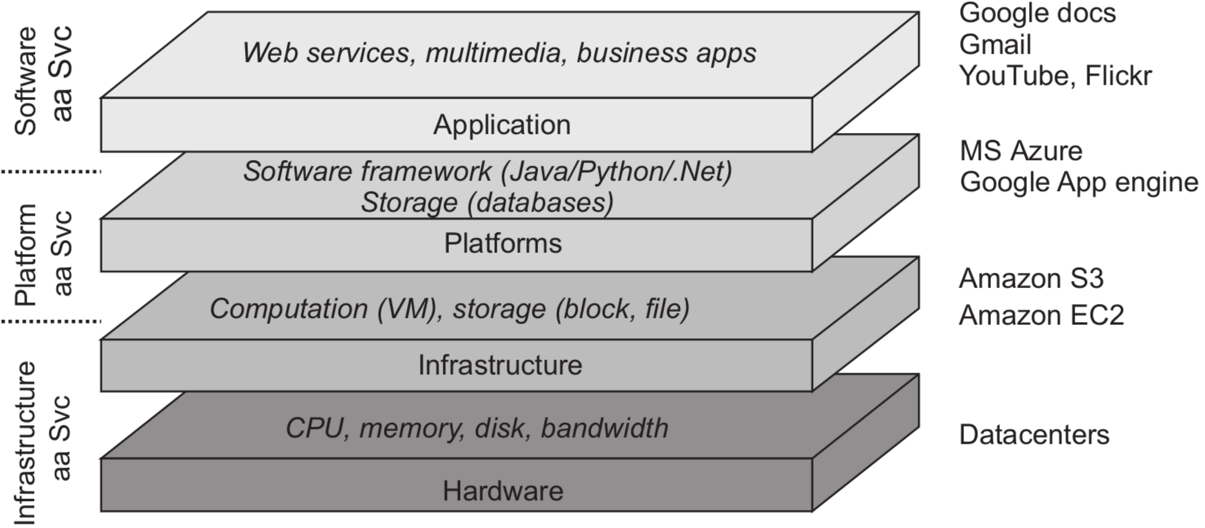
Distributed Information Systems

- Transaction processing systems



Cloud Computing Systems

- Organization of clouds



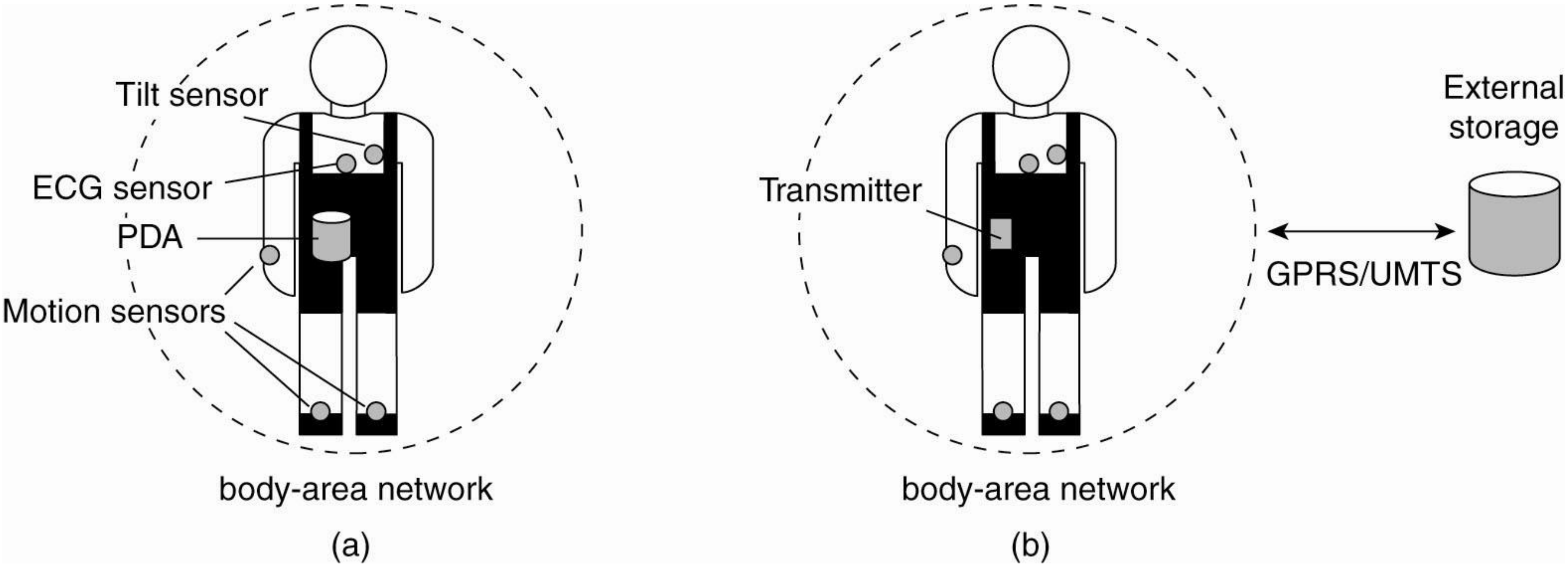
Cloud computing is characterized by an easily usable and accessible pool of virtualized resources.

□ Cloud services:

- □ **Infrastructure-as-a-Service** (IaaS) covering the hardware and infrastructurelayer
- □ **Platform-as-a-Service** (PaaS) covering the platform layer
- □ **Software-as-a-Service** (SaaS) in which their applications are covered

Distributed Pervasive Systems

- Electronic health care systems



Monitoring a person in a pervasive electronic health care system, using (a) a local hub or (b) a continuous wireless connection.

Distributed vs. Single Systems

- Data sharing
  - Multiple users can access common database, data files,…
- Device/resource sharing
  - Printers, servers, CPUs,….
- Communication
  - Communication with other machines…
- Flexibility
  - Spread workload to different & most appropriate machines
- Extensibility
  - Add resources and software as needed

Disadvantages of Distributed Systems

- Software
  - Little software exists compared to PCs
- Networking
  - Still slow and can cause other problems (e.g. when disconnected)
- Security
  - Data may be accessed by unauthorized users

What is Middleware

- Software above the operating system but below the application program
- Middleware refers to the software that is common to multiple applications and builds on the network transport services to enable ready development of new applications and network services
- DCOM, Java RMI, Cloud

Middleware Layer

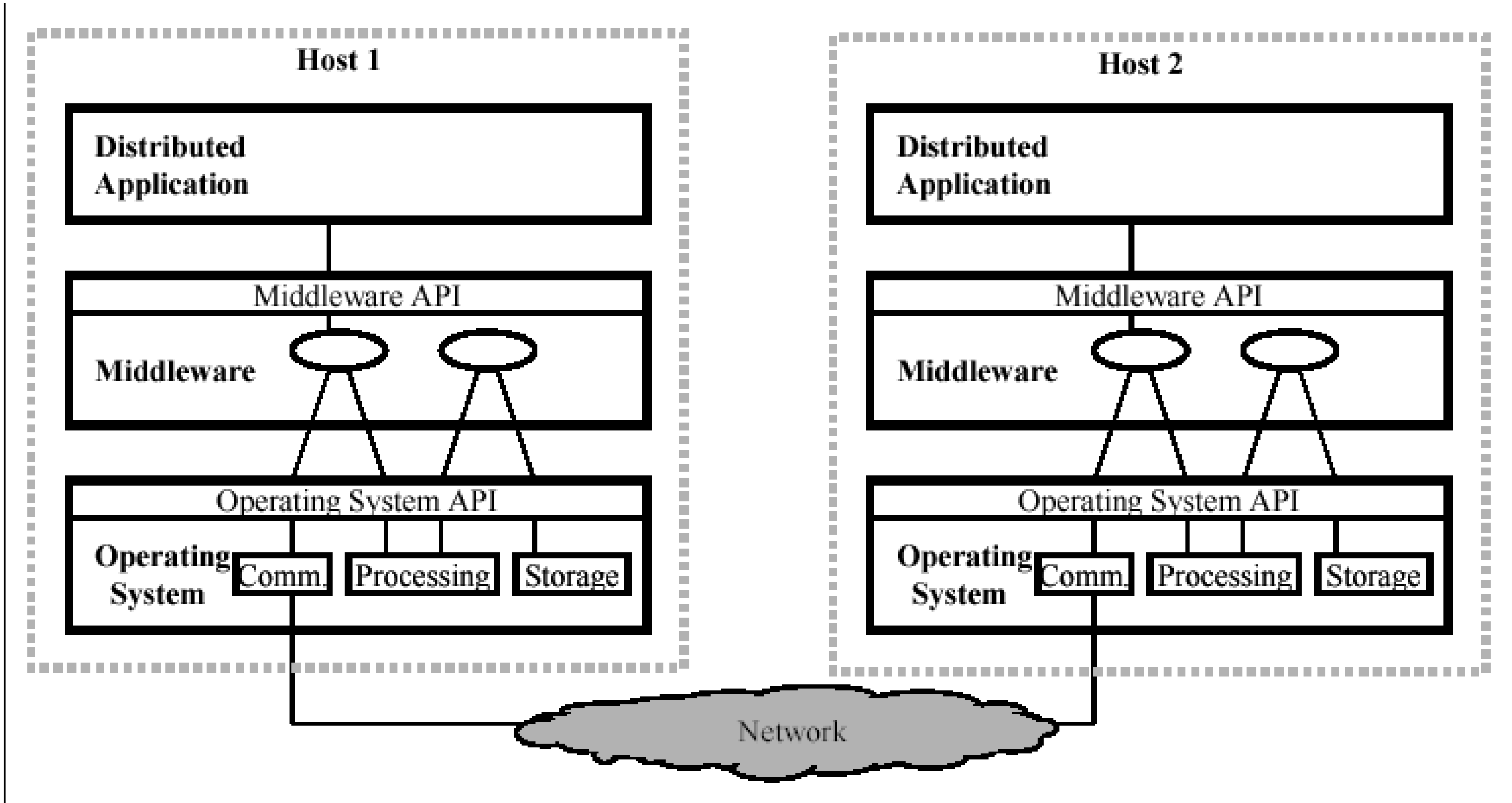
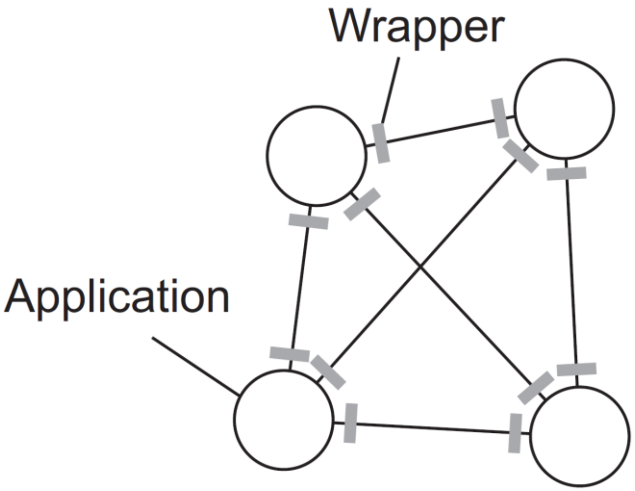
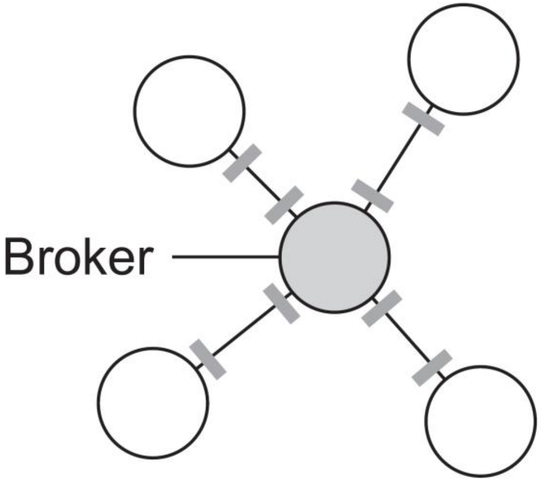


Figure 1: Middleware Layer in Context

Middleware Organization

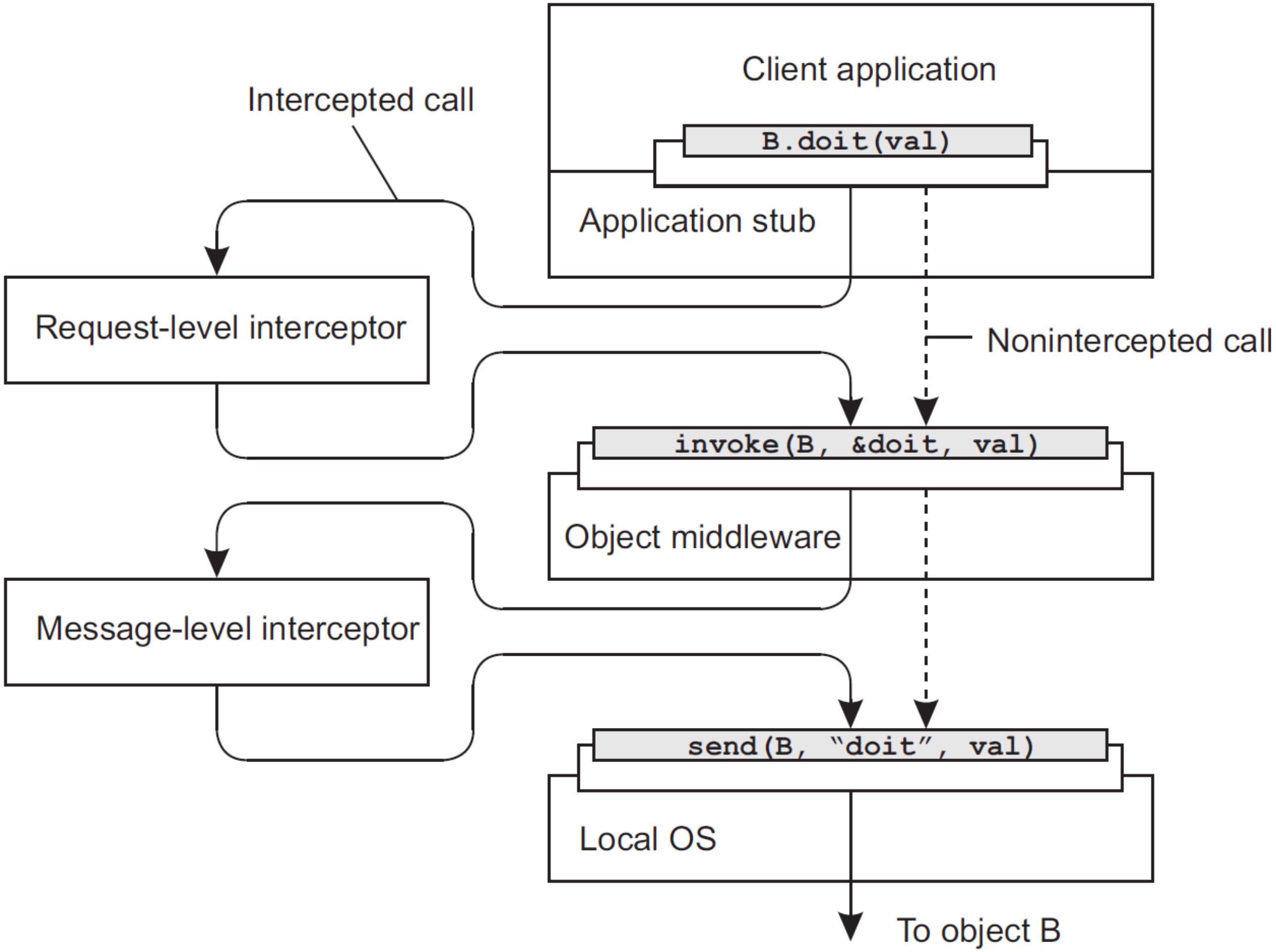
- Wrapper or Adapter





Middleware Organization

- Interceptor



Technical Challenges

- **The changing environment**
  - Computing world has changed, and middleware must adapt to this ever-changing environment
- **Architecture**
  - New technological advances impose changes in established middleware architecture
- **Dynamic configuration**
  - Dynamic changes in system configuration will be inherent characteristics of future computing environments.

Client/Server Organization

- Server: a software module manages a set of resources of a particular type using certain policies and methods.
  - Servers may be run in different machines
    - Mail server, http server
  - A machine can maintain more than one server
- Client: a software module requests services from servers.
- Centralized server versus by distributed servers
  - centralized server: e.g. printer and mail
  - distributed servers: e.g. file servers
- Proxy server and caches: middleman between origin server and clients

Peer-to-Peer Organization

- All processes play similar roles, interacting cooperatively as peers to perform a distributed activity or computation without any distinction between clients and servers.
- Fully distributed and parallel
- For Example
  - Remote memory access
  - Process migration
  - P2P file exchange

Mobile Code Organization

- Mobile codes
  - Programs that function as they are transferred from one host to the other. Instead of sending requests associated with input data to a server for processing, the mobile code approach uploads codes to the server for execution
  - E.g. Javascript code, Java Applets

- Mobile agent
  - can travel from host to host autonomously, carrying their code as well as running state.
  - Itinerary mobility (proactive mobility)
  - Security in mobile agents
    - Server protection
    - Agent protection

Summary

- Course information

- Definition of distributed systems
- Design goals
- Examples

- Readings
  - Chapter 1 of textbook

Any Questions?

Questions?

- What is the difference between operating system and (software) system?
- What is the difference between Network OS and Distributed OS?
- What is the difference between Distributed OS and Distributed (software) system?
- What is middleware?
- What is the difference between middleware and distributed (software) system?