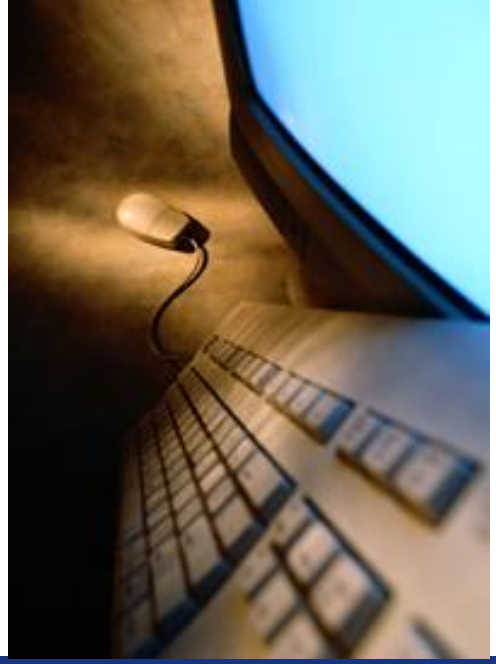
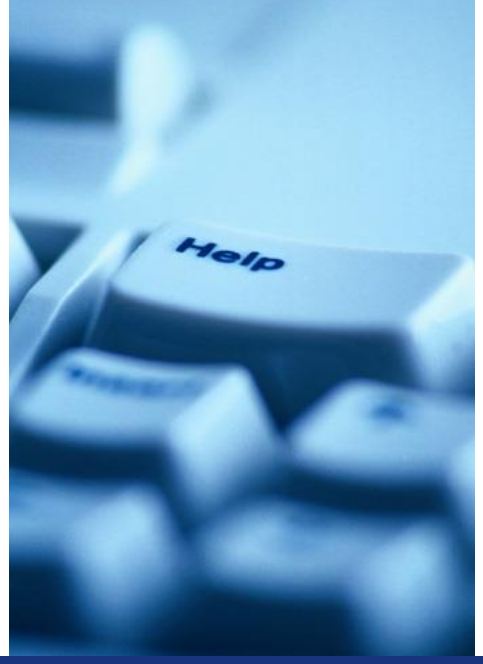



并行与分布式计算

Parallel & Distributed Computing

陈鹏飞
数据科学与计算机学院
2018-05-30
chenpf7@mail.sysu.edu.cn

Lecture 12 — Characterization of Distributed Systems

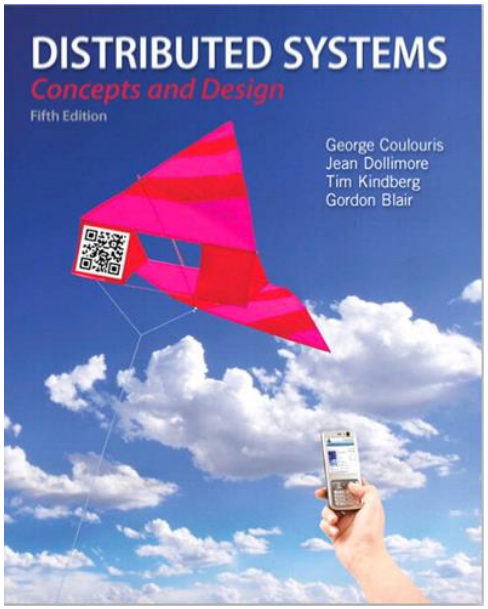
Pengfei Chen

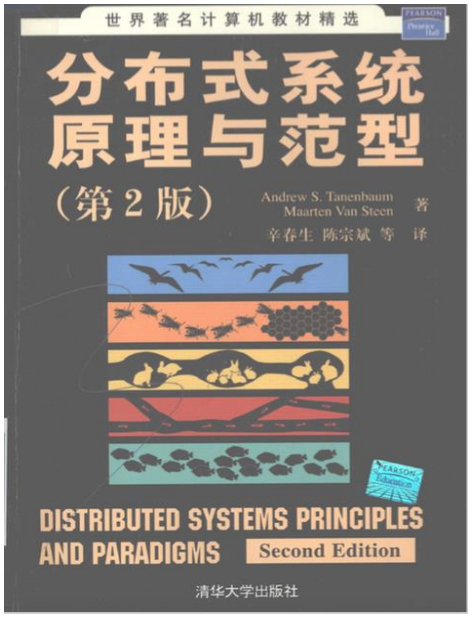
School of Data and Computer Science

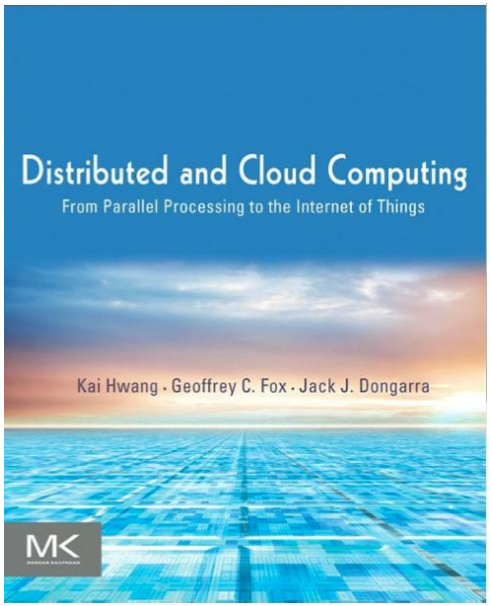
May 30, 2018

chenpf7@mail.sysu.edu.cn

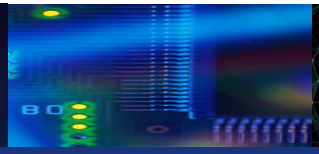
Text books







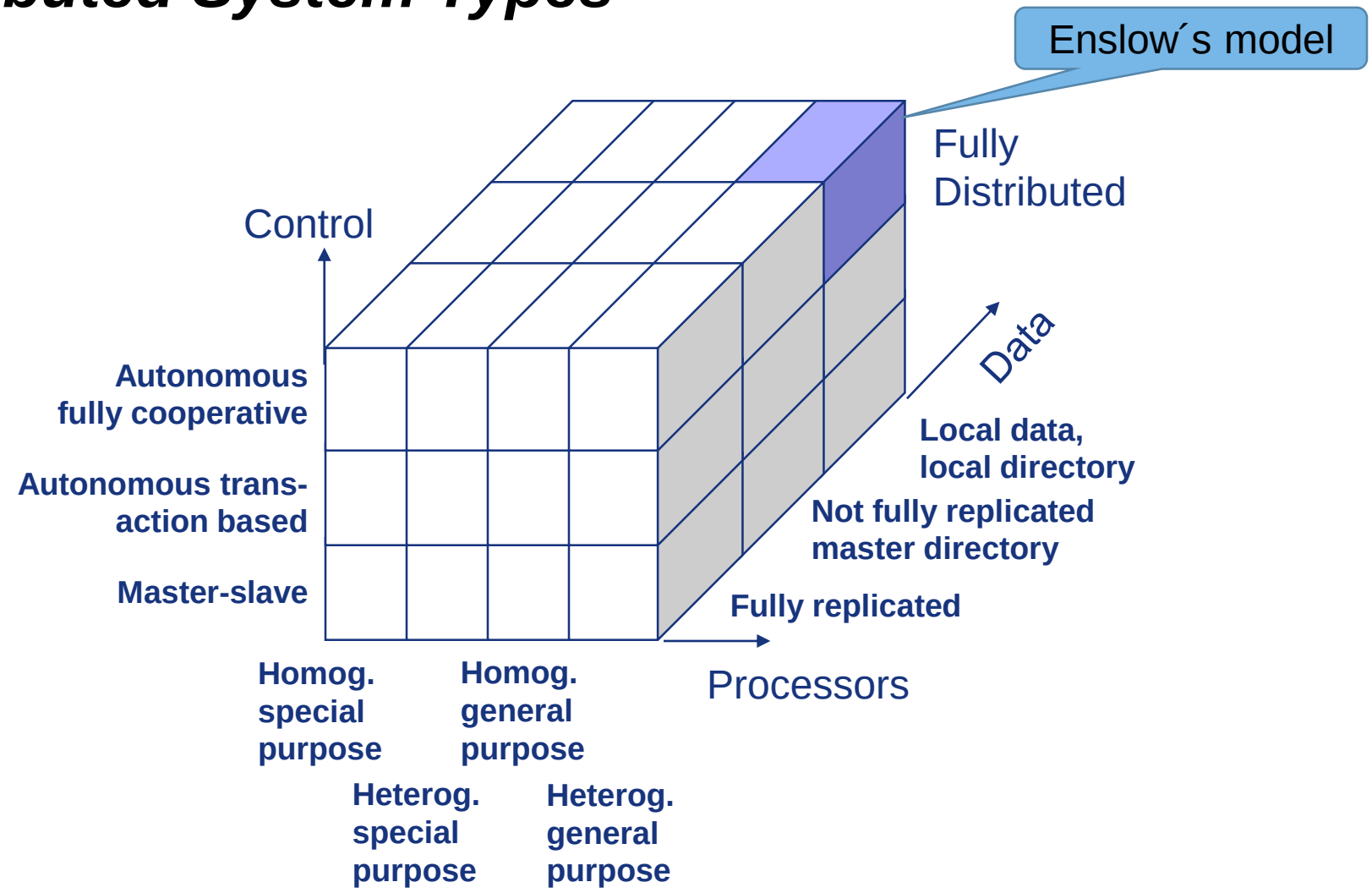
Resources: http://www.cdk5.net/wp/

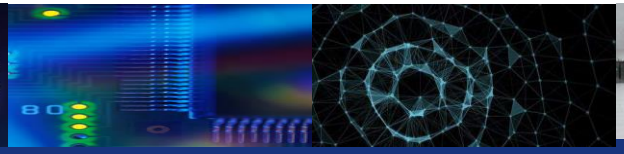


Outline:

- What is a Distributed System
- Examples of Distributed Systems
- Common Characteristics
- Basic Design Issues
- Summary

Distributed System Types

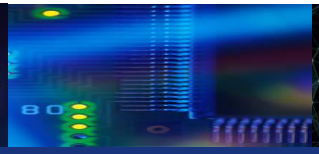




What is a Distributed System?

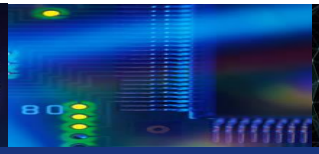
Definition: A *distributed system* is one in which components located at **networked computers** communicate and coordinate their actions only by **passing messages**. This definition leads to the following characteristics of distributed systems:

- concurrency of components
- lack of a global clock
- independent failures of components



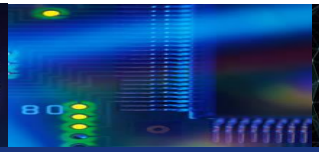
Centralized System Characteristics

- One component with non-autonomous parts
- Component shared by users all the time
- All resources accessible
- Software runs in a single process
- Single point of control
- Single point of failure



Distributed System Characteristics

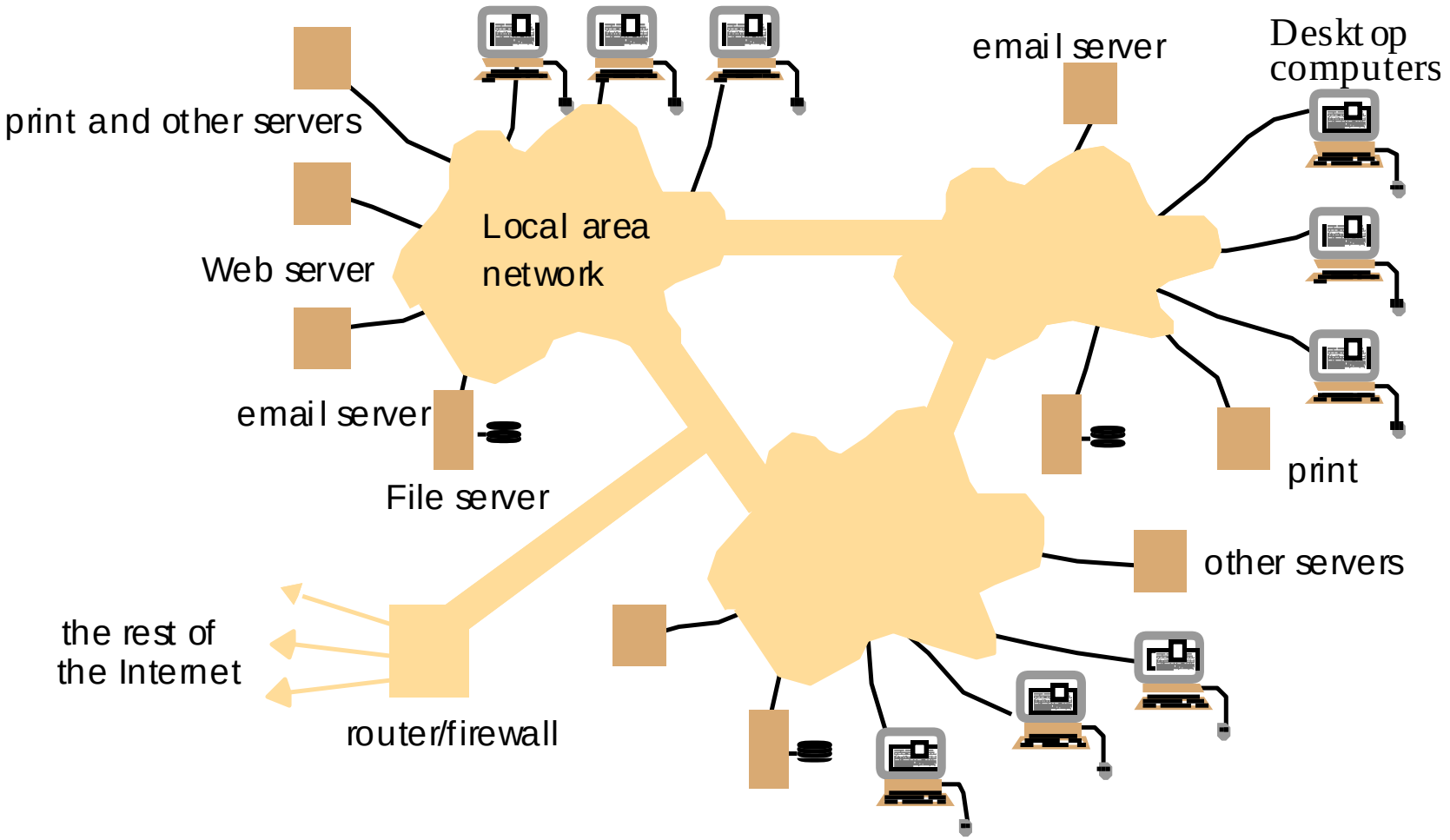
- Multiple autonomous components
- Components are not shared by all users
- Resources may not be accessible
- Software runs in concurrent processes on different processors
- Multiple points of control
- Multiple points of failure



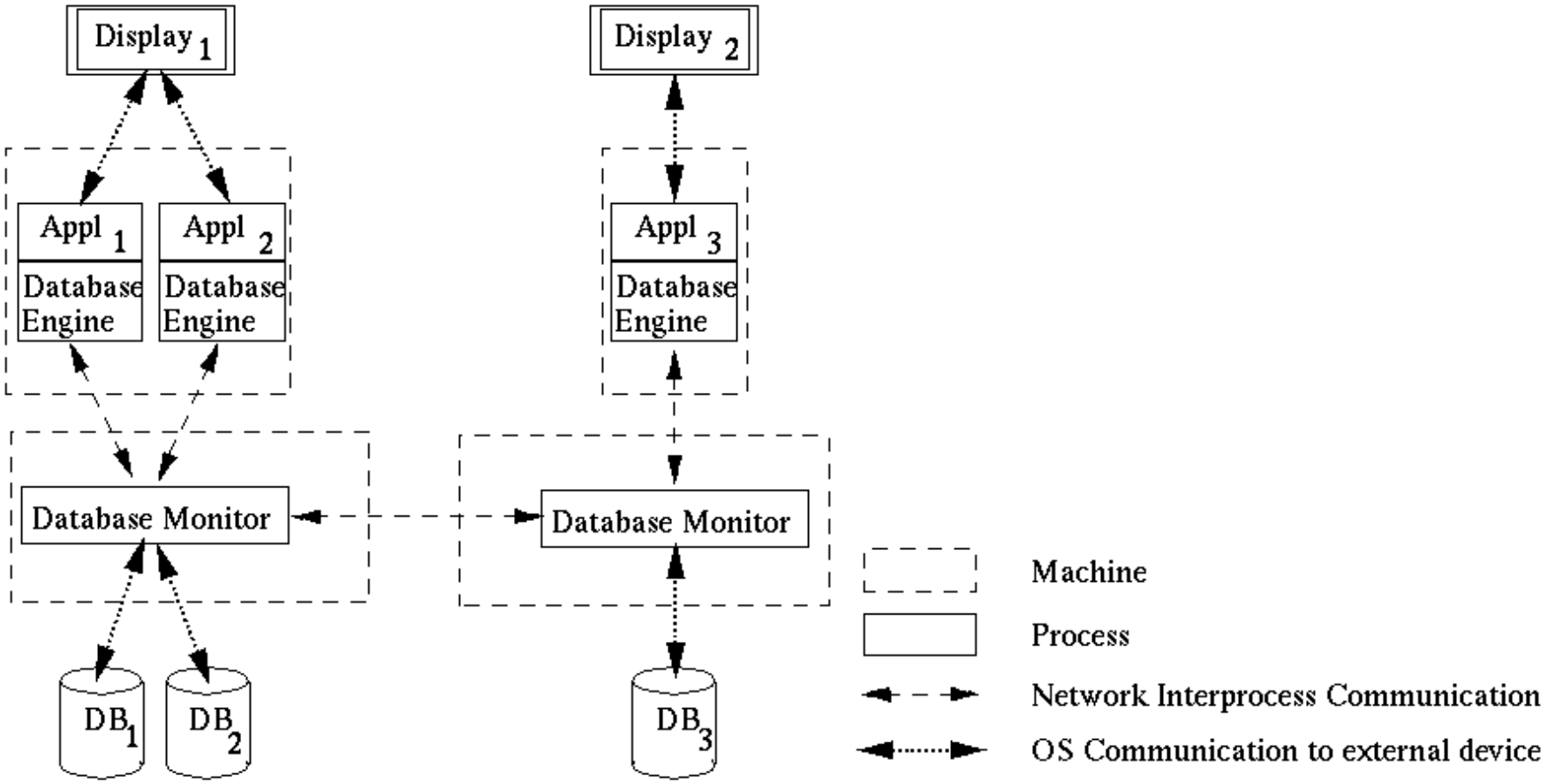
Examples of Distributed Systems

- Local Area Network and Intranet
- Database Management System
- Automatic Teller Machine Network
- Internet/World-Wide Web
- Mobile and Ubiquitous Computing

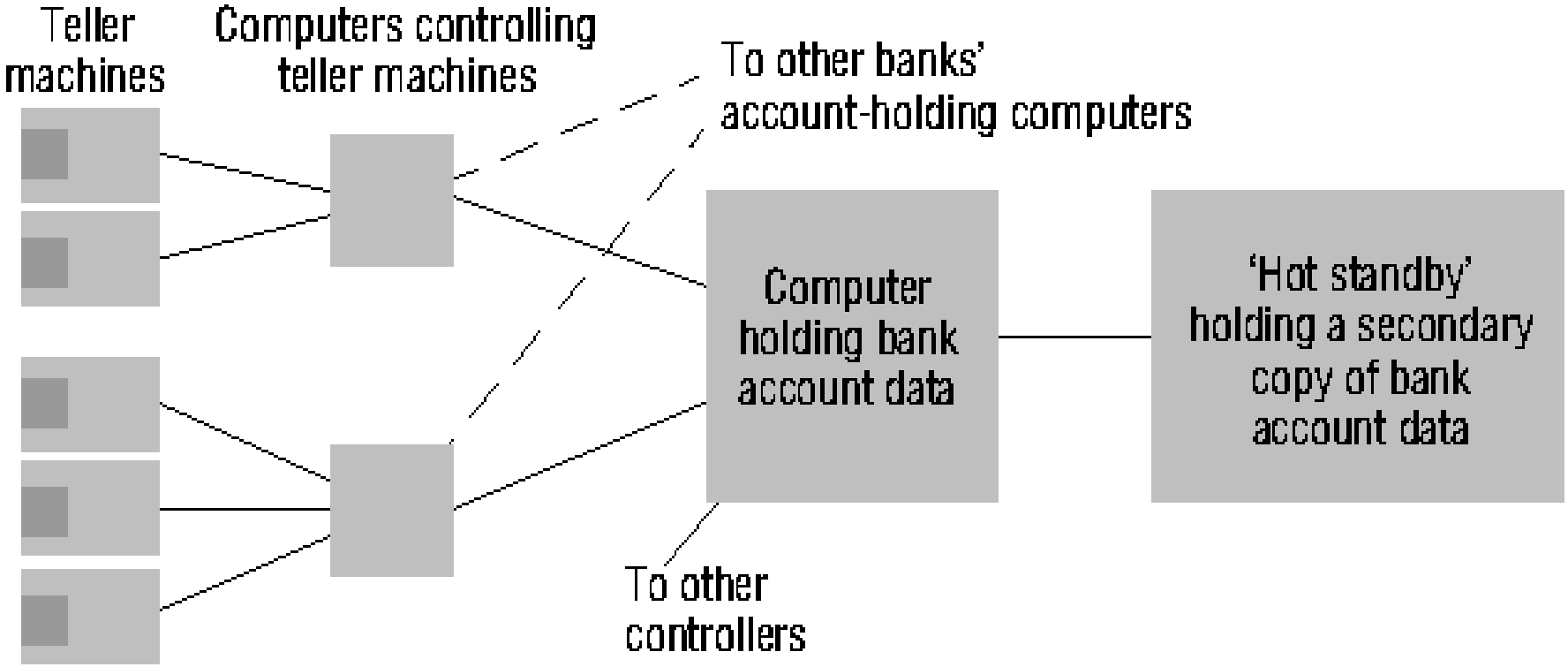
Local Area Network



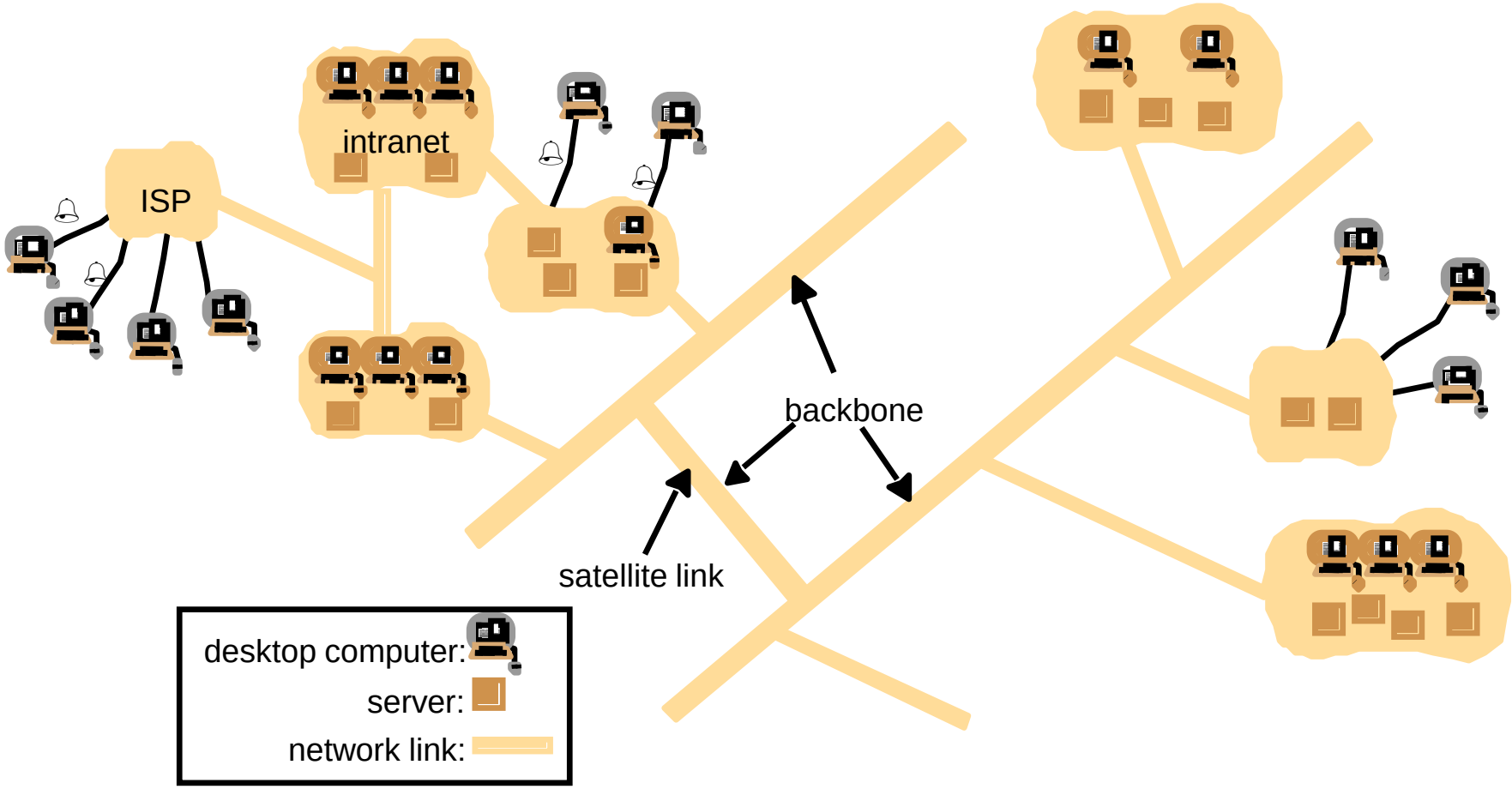
Database Management System (DBMS)



Automatic Teller Machine Network



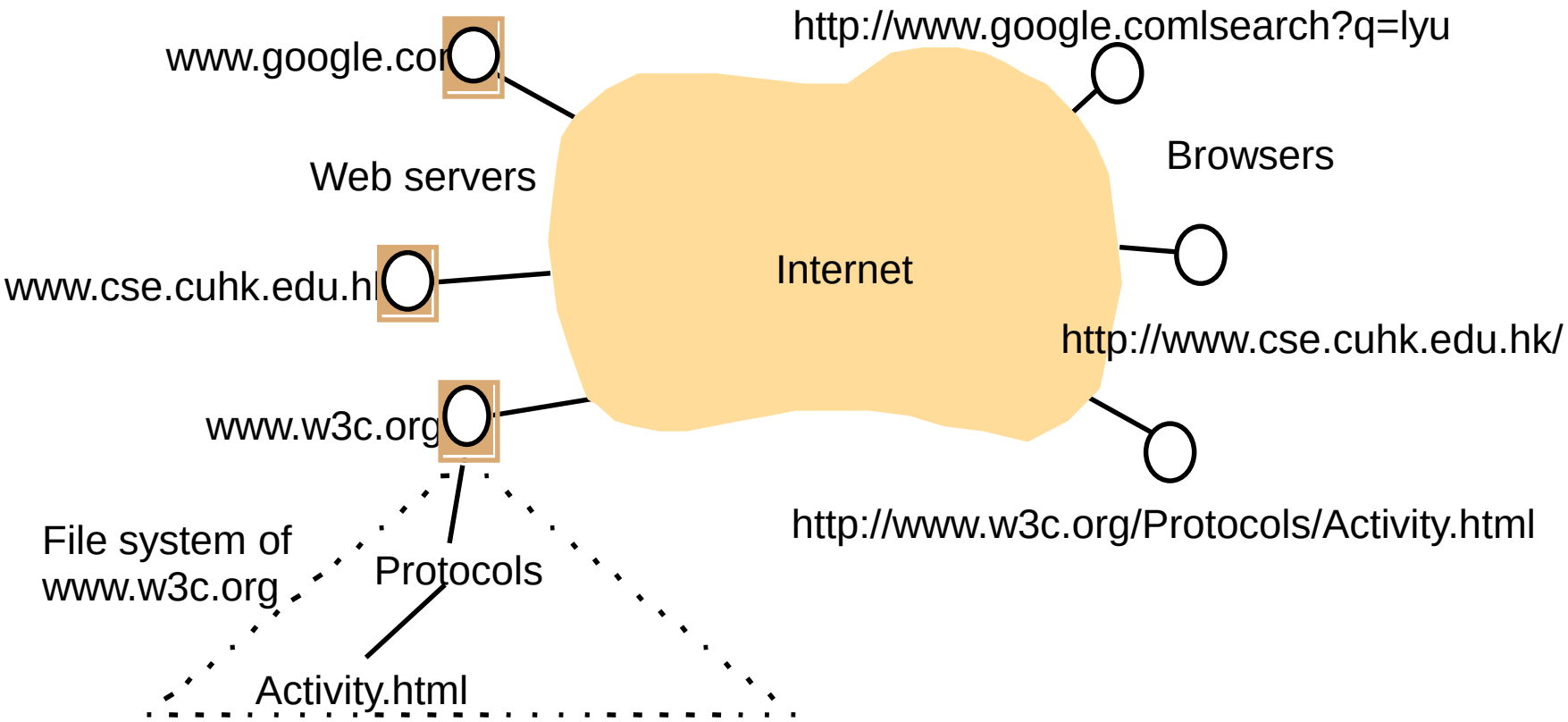
Internet


intranet
ISP
backbone
satellite link
desktop computer:
server:
network link:

Internet

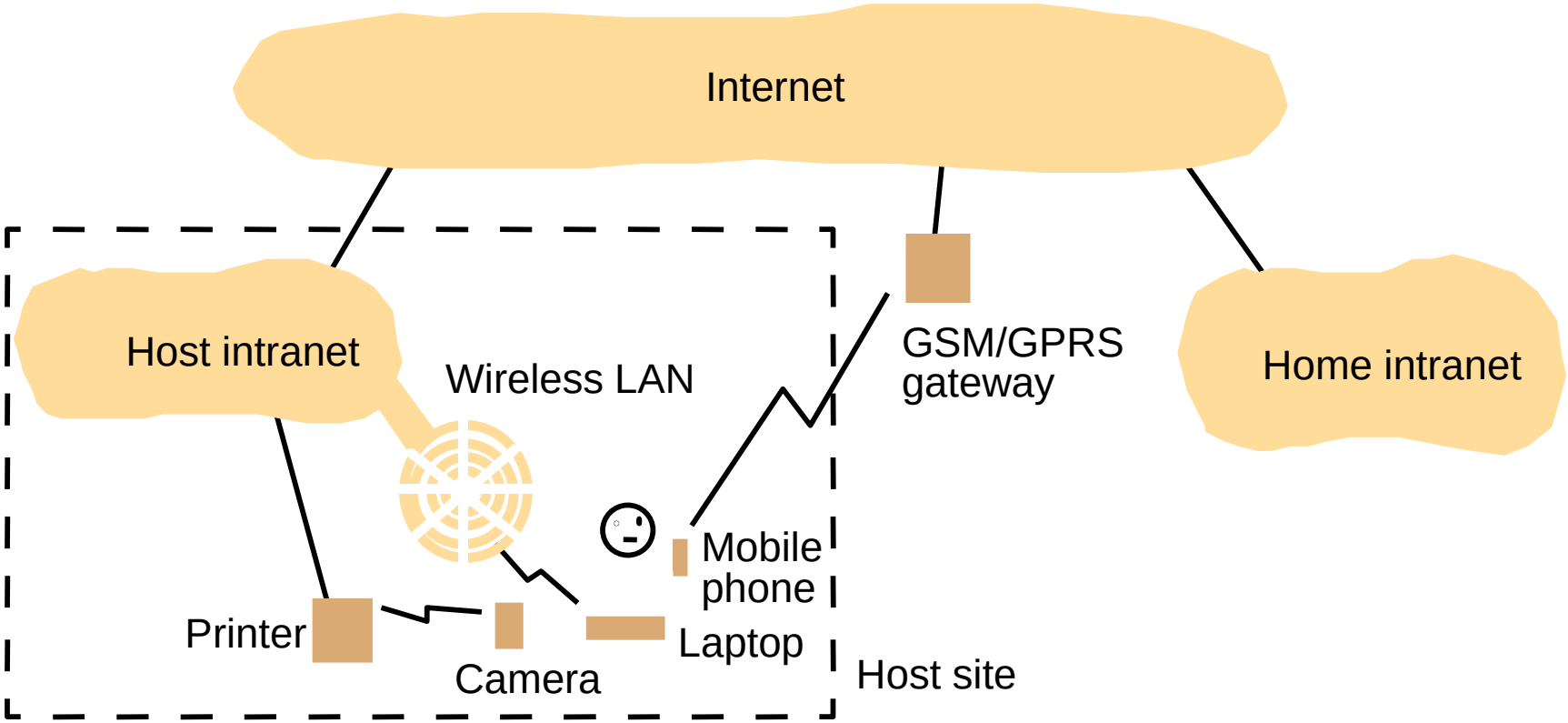

中山大學
SUN YAT-SEN UNIVERSITY
English Version
中大搜索
搜索
学校概况
机构设置
学科与师资
教育教学
科学研究
招生与就业
图书馆
人才招聘
校友与基金
走进中大
本科教育 | 研究生教育 | 国际教育 | 继续教育
扎根中国大地 建设一流大学
博學 審問 愼思 明辨 篤行
中大新闻 | NEWS & EVENTS
进入>>
"太平洋世界的环境史"国际学术会议在我... 05-28
校党委理论学习中心组开展"推进马克思主... 05-28
我校承办中国密码学会2018年密码算法... 05-28
中国统一战线理论研究会港澳和海外统战工... 05-29
科研专栏
进入>>
肿瘤防治中心徐瑞华教授团队牵头构建食道... 05-24
肿瘤防治中心张力教授团队发现共存突变对... 05-22
电子与信息工程学院陈钰杰副教授、余思远... 05-18
用户信息导航
我的中大 My SYSU
大学服务中心 USC
学生 | 教职工 | 校友 | 考生及访客

Web Servers and Web Browsers



The performance and security problems of DNS server?

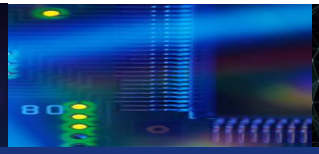
Mobile and Ubiquitous Computing (普适计算)



Edge Computing?

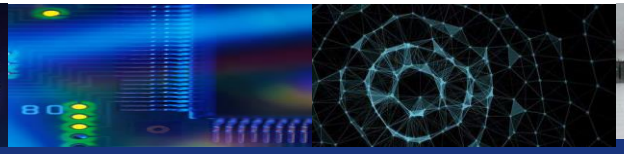
Common Characteristics

- What are we trying to achieve when we construct a distributed system? Certain common characteristics can be used to assess distributed systems
  - Heterogeneity （异构性）
  - Openness
  - Security
  - Scalability
  - Failure Handling
  - Concurrency （多用户）
  - Transparency



Heterogeneity

- Variety and differences in
  - Networks
  - Computer hardware
  - Operating systems
  - Programming languages
  - Implementations by different developers
- *Middleware* as software layers to provide a programming abstraction as well as masking the heterogeneity of the underlying networks, hardware, OS, and programming languages (e.g., Web service).
- *Mobile Code* to refer to code that can be sent from one computer to another and run at the destination (e.g., Java applets, Java *virtual machine, Apps*) （可移植性）.



Openness

- Openness is concerned with extensions and improvements of distributed systems.
- Detailed interfaces of components need to be published.
- New components have to be integrated with existing components.
- Differences in data representation of interface types on different processors (of different vendors) have to be resolved.

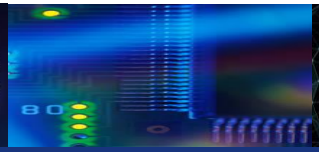
Security

- 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 (隐藏) 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 or apps

Scalability

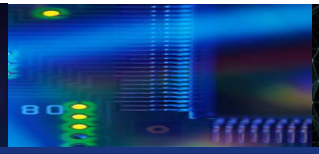
- Adaptation of distributed systems to
  - accommodate more users
  - respond faster (this is the hard one)
- Usually done by adding more and/or faster processors.
- Components should not need to be changed when scale of a system increases. (解耦)
- Design components to be scalable!

Scale out / in ?



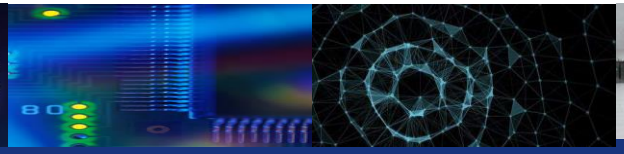
Concurrency

- 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.
  - Lost updates
  - Inconsistent analysis



Transparency

- Distributed systems should be perceived by users and application programmers as a whole rather than as a collection of cooperating components.
- Transparency has different aspects. （自主）
- These represent various properties that distributed systems should have.

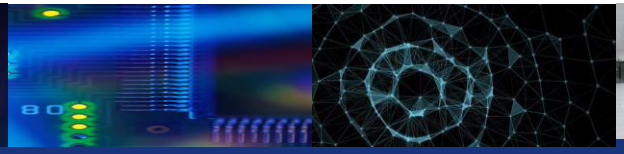


Access Transparency

- Enables local and remote information objects to be accessed using identical operations.
- Example: File system operations
- Example: Navigation in the Web
- Example: Database queries.

Location Transparency

- Enables information objects to be accessed without knowledge of their location.
- Example: File system operations
- Example: Pages in the Web
- Example: Tables in distributed databases (How about NoSQL DB?)



Concurrency Transparency

- Enables several processes to operate concurrently using shared information objects without interference between them.
- Example: File system operations
- Example: Automatic teller machine network
- Example: Database Management System (DBMS)

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

HDFS ?

Failure Transparency

- Enables the concealment of faults
- Allows users and applications to complete their tasks despite the failure of other components.
- Example: Database Management System (DBMS)
- Example: Cloud Computing – Amazon Web Service

Mobility Transparency

- Allows the movement of information objects within a system without affecting the operations of users or application programs
- Example: Network file service
- Example: Web Pages

Performance Transparency

- Allows the system to be reconfigured to improve performance as loads vary and parallelism can be explored.

- Example: Hadoop which implements MapReduce.

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

Design Issues

- Specific issues for distributed systems:
  - Naming
  - Communication
  - Software structure
  - System architecture
  - Workload allocation
  - Consistency maintenance

Naming

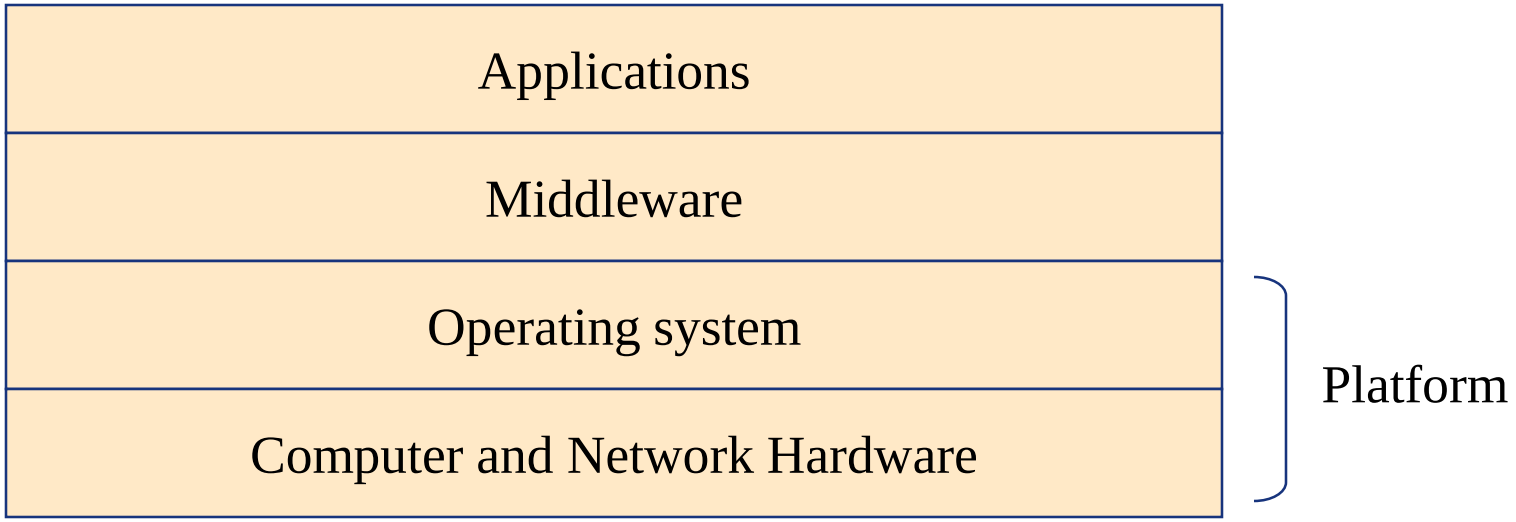
- A name is resolved when translated into an interpretable form for resource/object reference.
  - Communication identifier (IP address + port number)
  - Name resolution involves several translation steps
- Design considerations
  - Choice of name space for each resource type
  - Name service to resolve 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

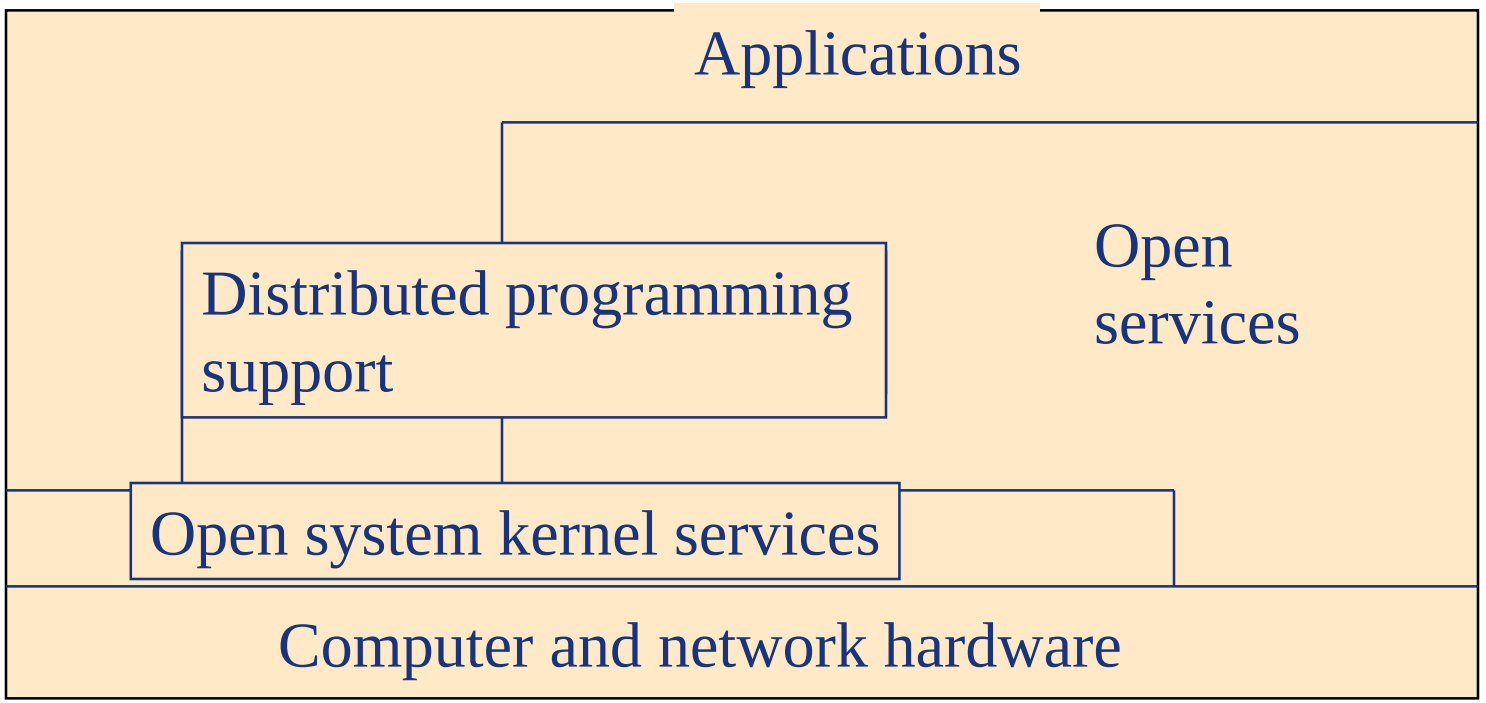
Software Structure

- Layers in centralized computer systems:



Software Structure

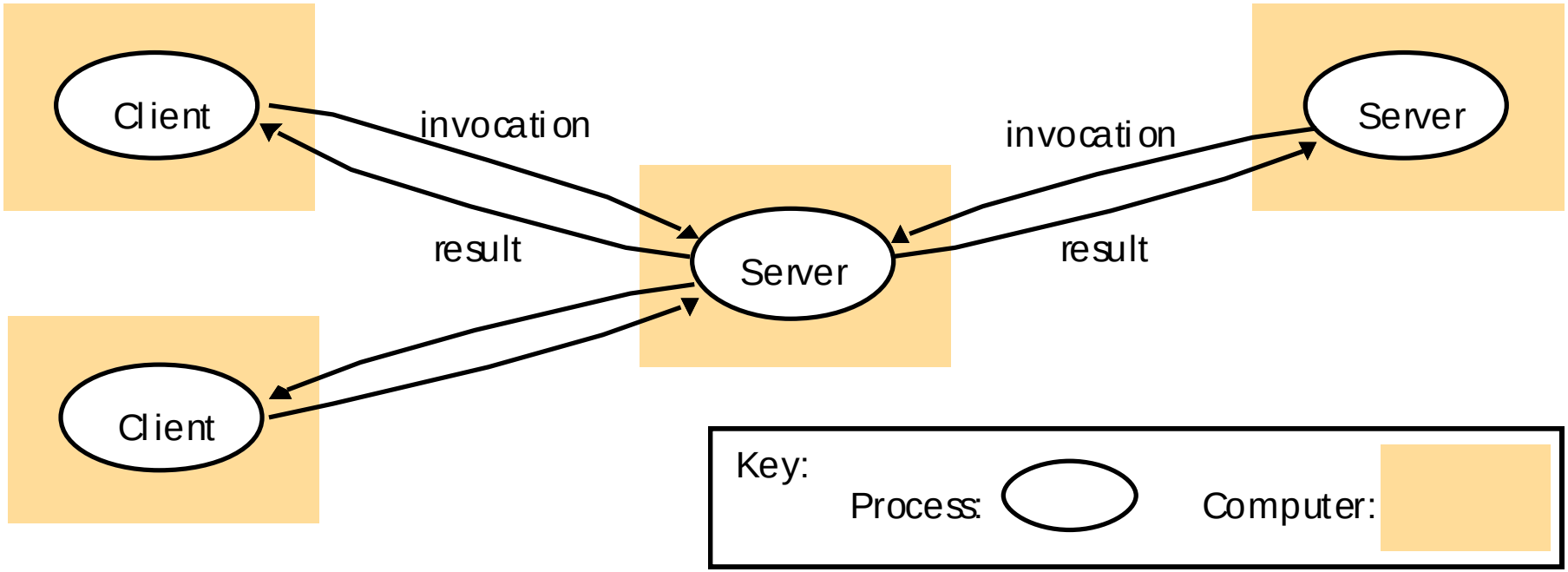
- Layers and dependencies in distributed systems:



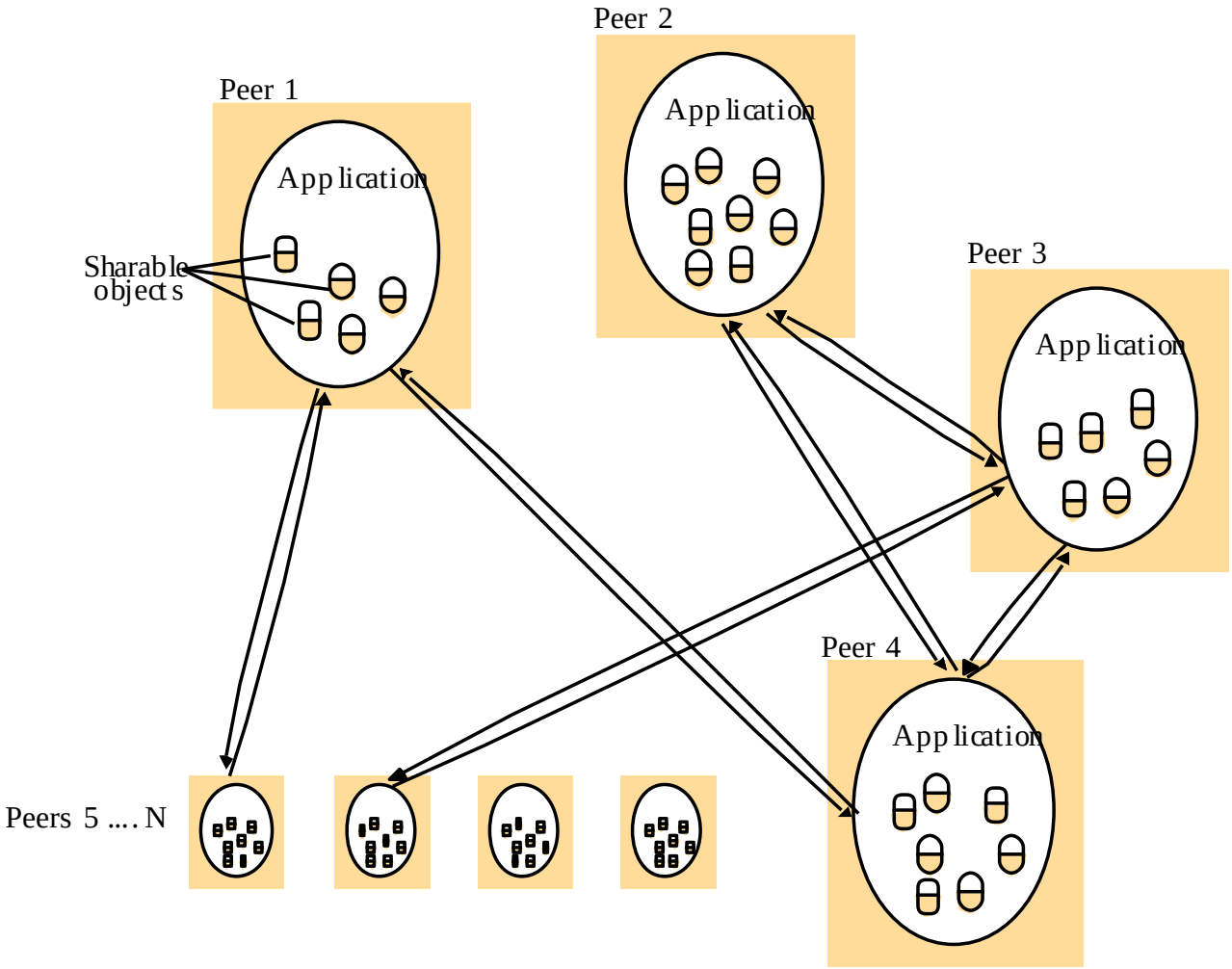
System Architectures

- Client-server
- Peer-to-peer
- Services provided by multiple servers
- Proxy servers and caches
- Mobile code and mobile agents
- Network computers
- Thin clients and mobile devices

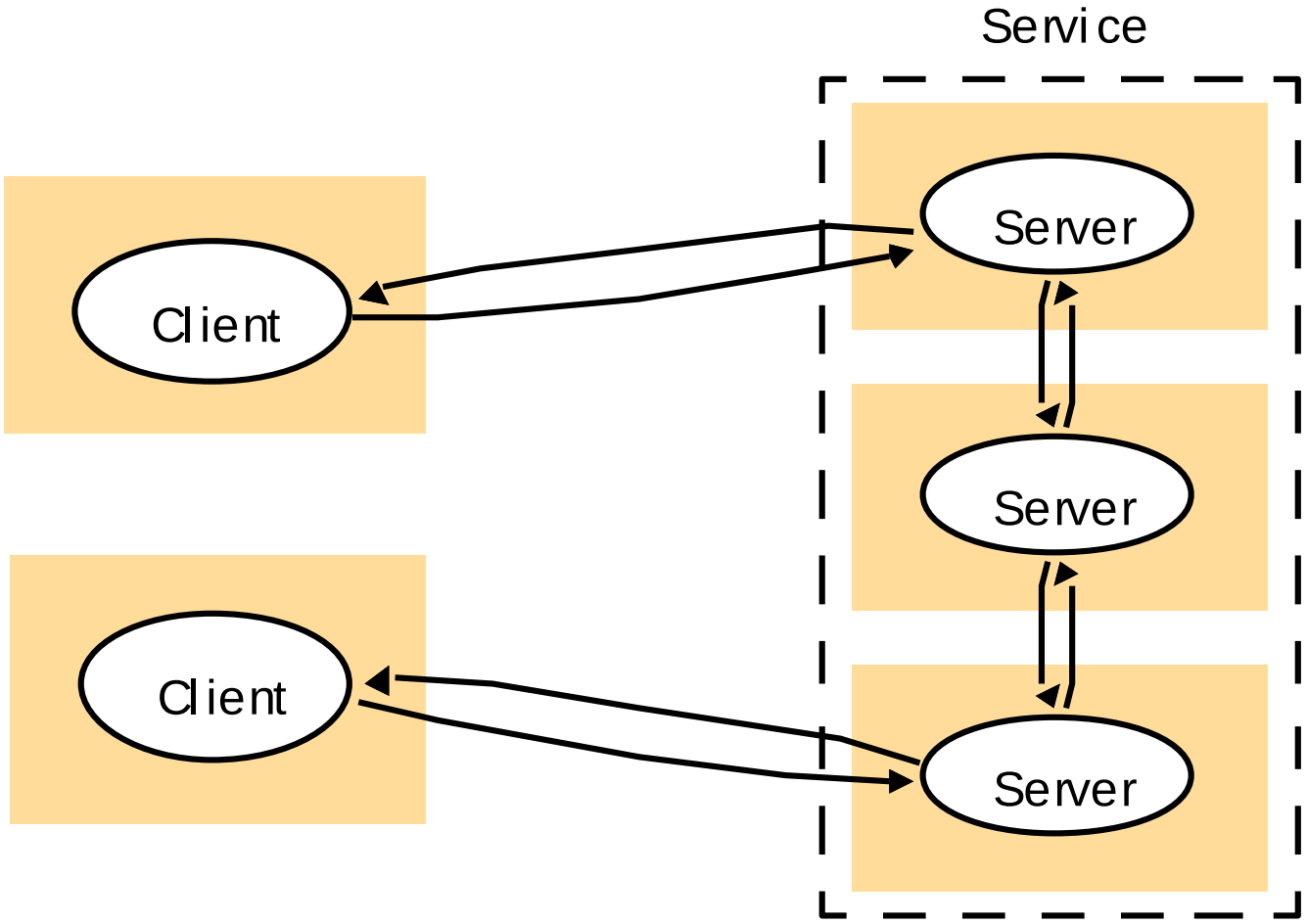
Clients Invoke Individual Servers



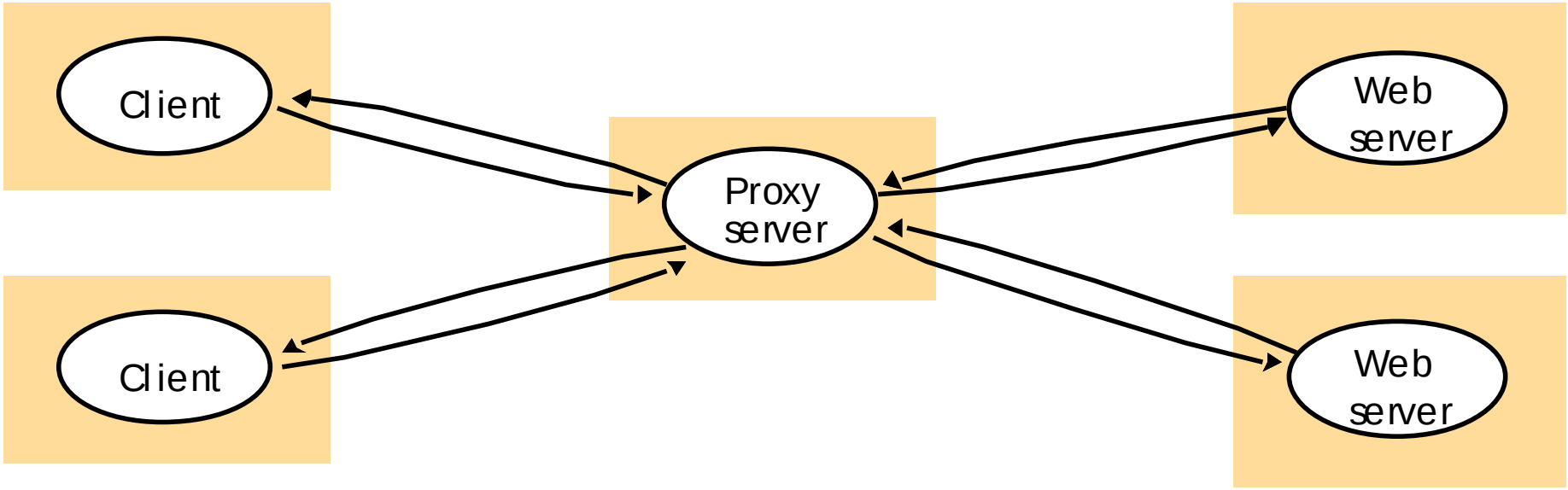
Peer-to-peer Systems



A Service by Multiple Servers

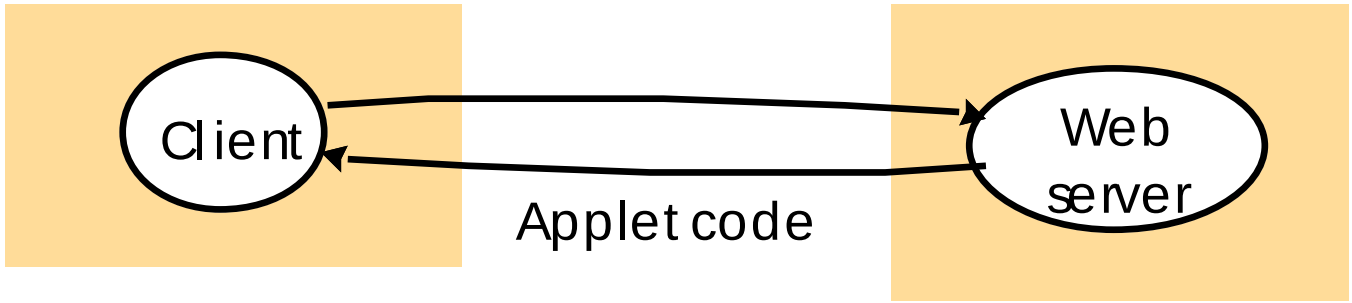


Web Proxy Server



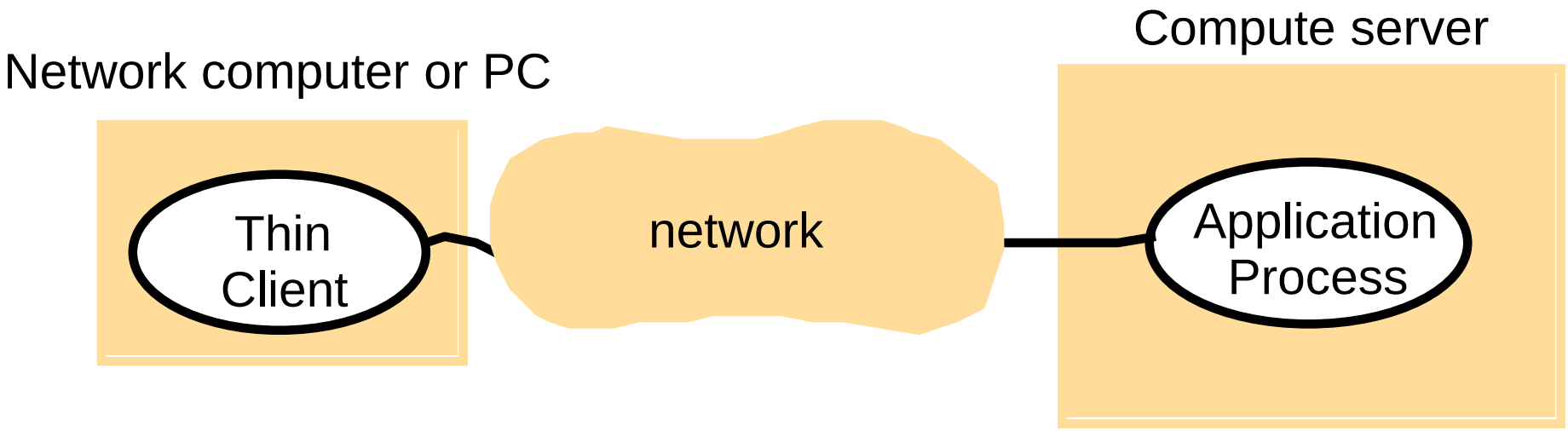
Web Applets

a) client request results in the downloading of applet code



b) client interacts with the applet

Thin Clients and Compute Servers



Summary

- Definitions of distributed systems and comparisons to centralized systems.
- The characteristics of distributed systems.
- The eight forms of transparency.
- The basic design issues.
- Read Chapter 1 and Chapter 2 of the textbook.

Thank You !