DISTRIBUTED SYSTEMS

MODULE IV

Introduction

A distributed system is one in which components located at networked computers communicate and coordinate their actions only by passing messages.

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

significant characteristics of distributed systems:

- concurrency of components,
- lack of a global clock and
- independent failures of components.

The Prime motivation is *to share resources*

Example: Internet

Examples

1. Web Search

- 10 billion searches per calendar month
- 63 billion pages in the web
- One trillion unique web addresses
- Example: Google search engine (largest and most complex distributed system infrastructure)

2. Massively multiplayer online games (MMOGs)

- Large numbers of users interact through the Internet with a persistent virtual world
- 50k simultaneous online players

Challenges:

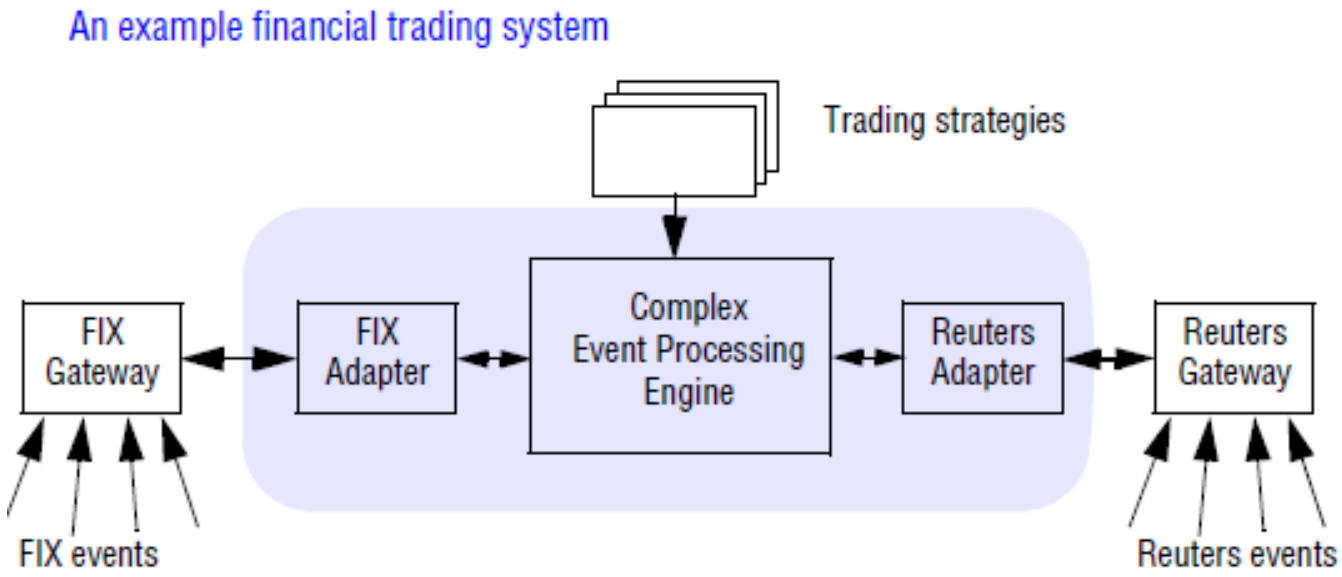
- Need for fast response times to preserve the user experience of the game
- Real-time propagation of events to the many players and maintaining a consistent view of the shared world

Preferable architecture: Client-server or distributed

Examples

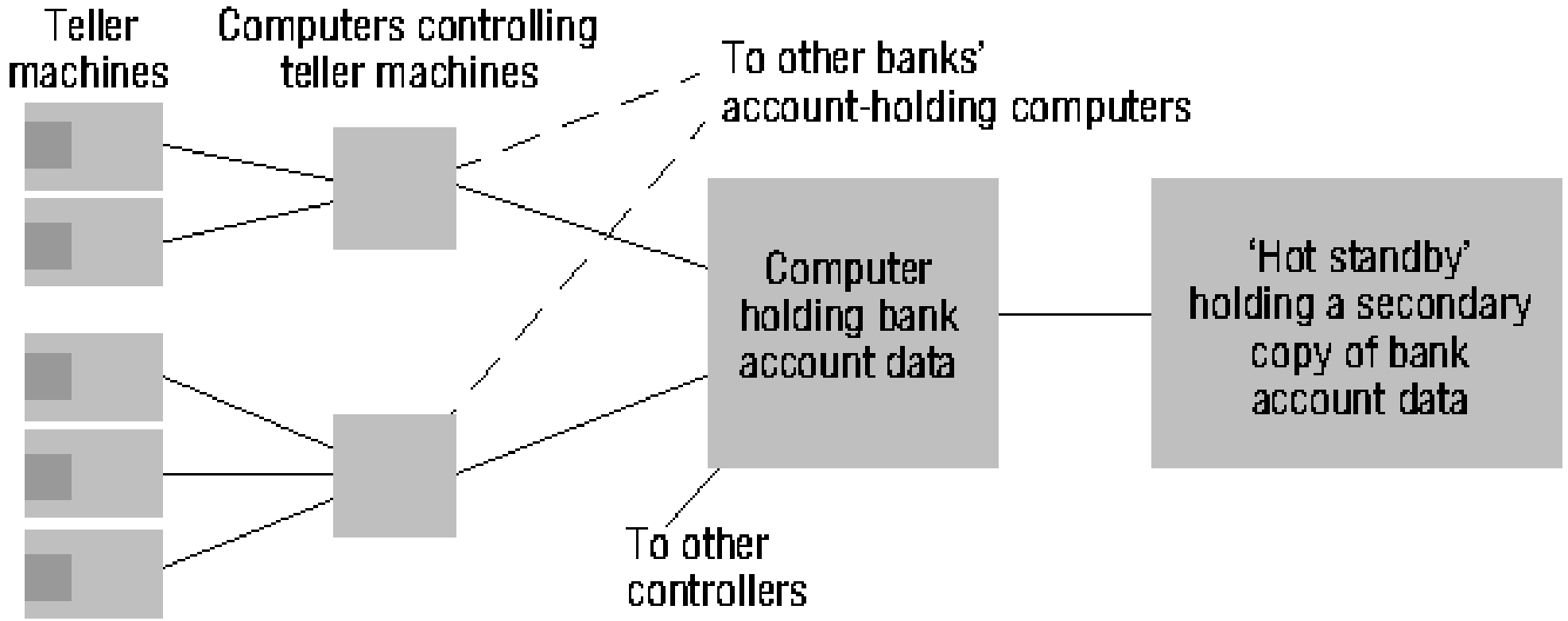
3. Financial Trading

distributed event-based systems



Examples

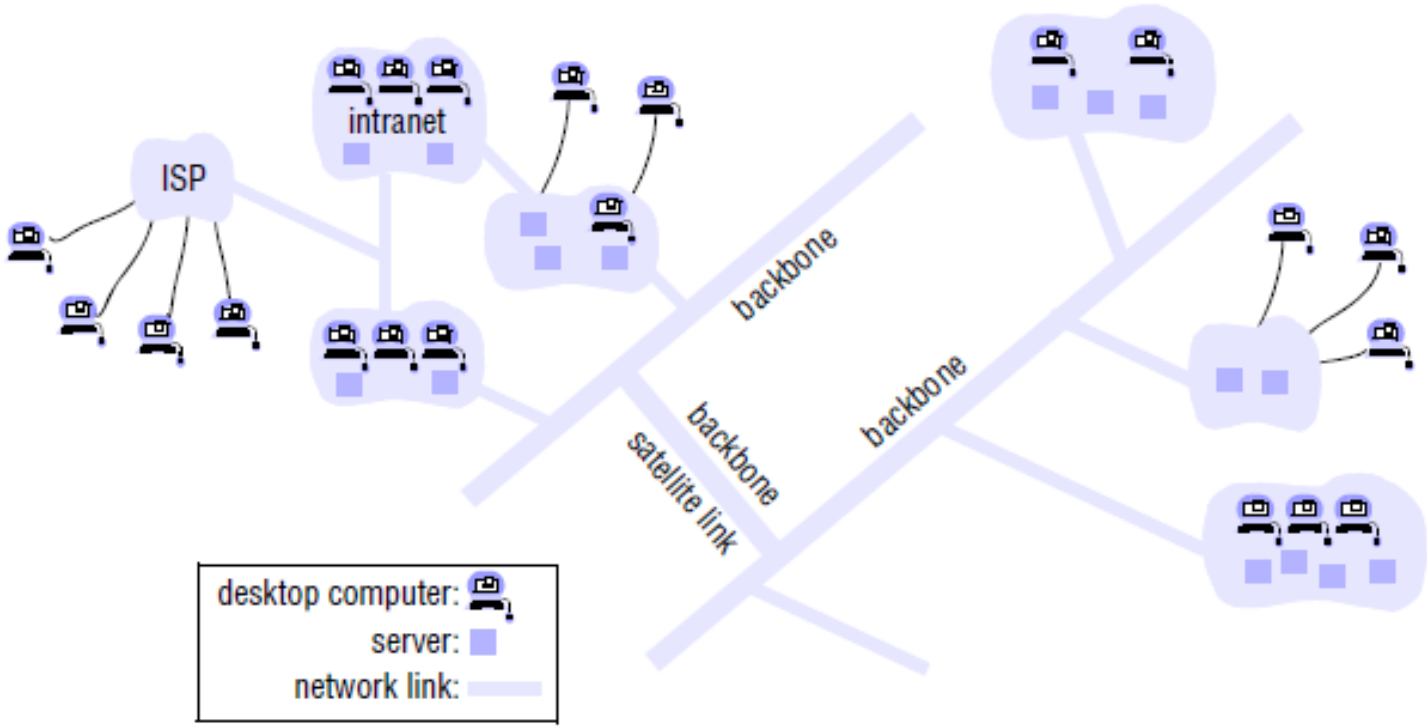
4. Automatic Teller Machine



Trends in Distributed Systems

1. Emergence of pervasive networking and modern Internet

Figure A typical portion of the Internet



Trends in Distributed Systems

2. Mobile and ubiquitous computing

Integration of small and portable computing devices into distributed systems. These devices include:

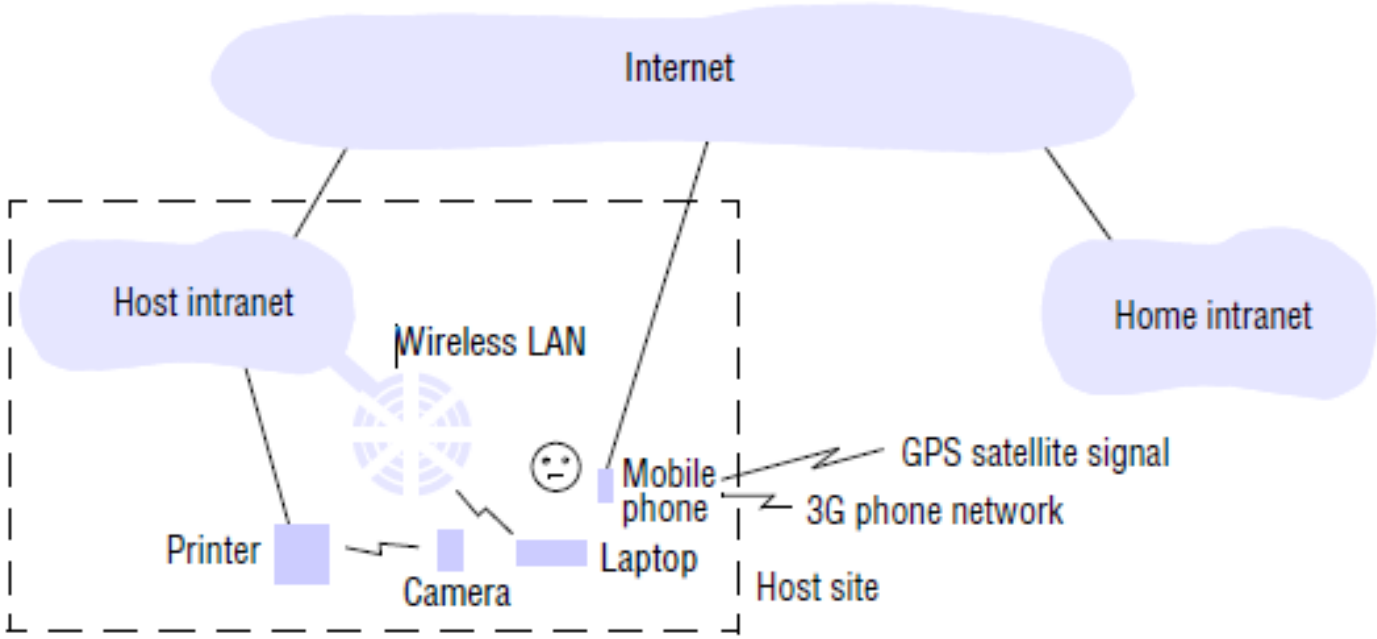
- Laptop computers.
- Handheld devices, including mobile phones, smart phones, GPS-enabled devices, pagers, personal digital assistants (PDAs), video cameras and digital cameras.
- Wearable devices, such as smart watches with functionality similar to a PDA.
- Devices embedded in appliances such as washing machines, hi-fi systems, cars and refrigerators.

Ubiquitous computing is the harnessing of many small, cheap computational devices that are present in users' physical environments, including the home, office and even natural settings.

Trends in Distributed Systems

2. Mobile and ubiquitous computing

Portable and handheld devices in a distributed system



Trends in Distributed Systems

3. Distributed multimedia systems

- ability to support a range of media types in an integrated manner
- to support the storage, transmission and presentation of discrete media types, such as pictures or text messages
- able to store and locate audio or video files, to transmit them across the network
- video-on-demand services, access to music libraries, the provision of audio and video conferencing facilities, etc.

Example: Webcasting is the ability to broadcast continuous media, typically audio or video, over the Internet.

Hotstar?

Trends in Distributed Systems

4. Distributed computing as a utility

Resources are provided by appropriate service suppliers and effectively rented rather than owned by the end user

Example: cloud computing

Clouds are generally implemented on cluster computers to provide the necessary scale and performance required by such services.

A cluster computer is a set of interconnected computers that cooperate closely to provide a single, integrated high performance computing capability.

Challenges

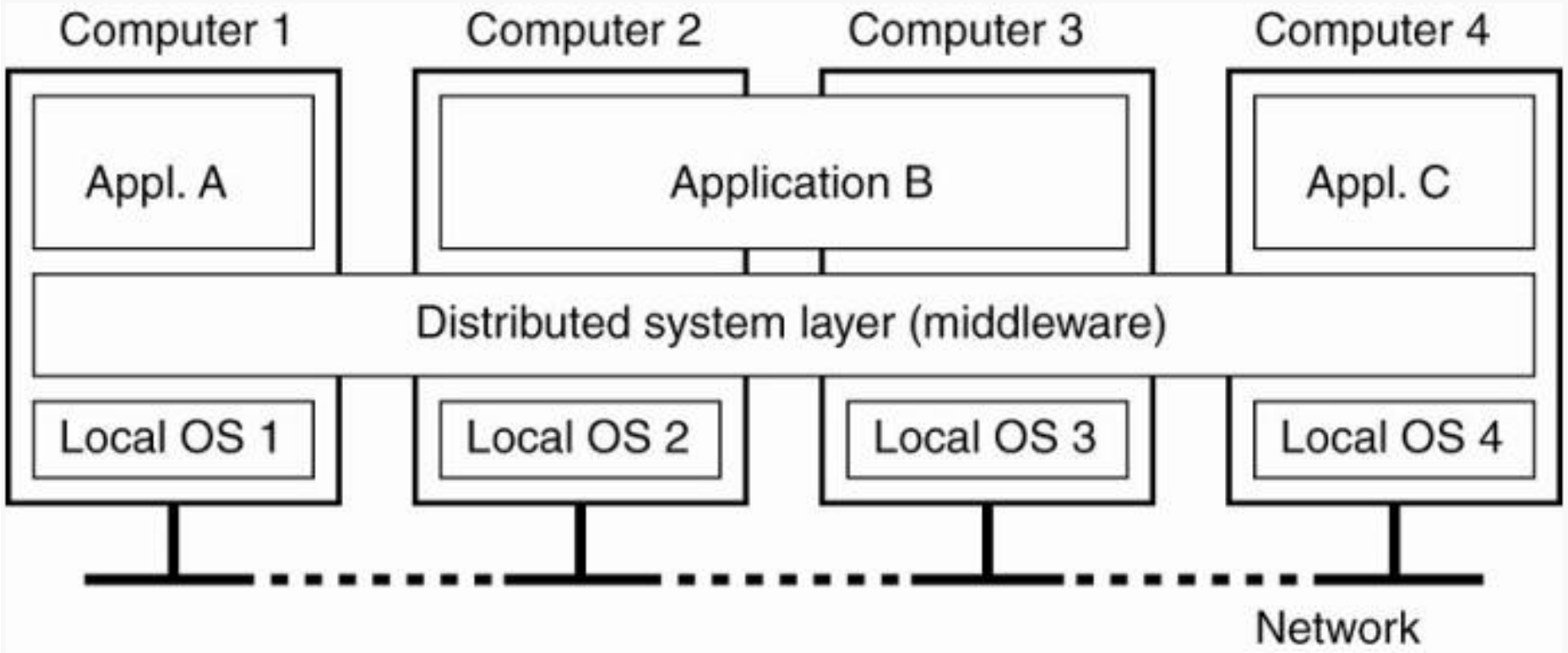
1. Heterogeneity
2. Openness
3. Security
4. Scalability
5. Failure Handling
6. Concurrency
7. Transparency – Access, Location, Concurrency, Replication, Failure, Mobility, Performance and Scaling
8. Quality of Service

1. 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., CORBA).
- *Mobile Code* to refer to code that can be sent from one computer to another and run at the destination (e.g., Java applets and Java *virtual machine*).

Heterogeneity

A distributed system organized as middleware.



2. 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.

3. Security

- Security for information resources has three components:
  - confidentiality, integrity and availability

- 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

4. 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.
- Design components to be scalable!

5. Failure Handling

- 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

The following techniques are available for dealing with failures:

Detecting failure, masking failure, Tolerating from failure, Recovery from failure, Redundancy

6. 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

7. 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.
- These represent various properties that distributed systems should have.

7. Transparency Types

- *Access transparency* enables local and remote resources to be accessed using identical operations.
- *Location transparency* enables resources to be accessed without knowledge of their physical or network 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 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.

8. Quality of Service

Factors affecting QoS

- Reliability
- Security
- Performance
- Adoptability
- Resource availability