고려대학교
KOREA UNIVERSITY

KU-The Future

Cloud computing - Distributed Systems

Heonchang Yu

Distributed and Cloud Computing Lab.

Concept of Distributed systems

- **Definition**
  - **[Coulouris]** System in which hardware or software components located at networked computers communicate and coordinate their actions only by message passing.
  - System that consists of a collection of two or more independent computers which coordinate their processing through the exchange of synchronous or asynchronous message passing
  - **[Tanenbaum]** Distributed system is a collection of independent computers that appear to the users of the system as a single computer.
  - Distributed system is a collection of autonomous computers linked by a network with software designed to produce an integrated computing facility.

Concept of Distributed systems

- **Consequences**
  - **Concurrency**
    - Can do my work on my computer while you do your work on yours, sharing resources
    - The coordination of concurrently executing programs that share resources is an important topic.
    - A and B are concurrent if either A can happen before B, or B can happen before A (interleaving semantics)
    - Synchronization and coordination by message passing
    - Deadlocks
    - Lifelocks

Concept of Distributed systems

- **Consequences**
  - **No global clock**
    - Close coordination depends on a shared idea of the time at which the programs' actions occur.
    - There are limits to the accuracy with which the computers in a network can synchronize their clocks.
    - The only communication is by sending messages through a network.
  - **Independent failures**
    - Faults in the network result in the isolation of the computers that are connected to it, but that doesn't mean that they stop running. The programs on them may not be able to detect whether the network has failed or has become unusually slow.
    - The failure of a computer, or the unexpected termination of a program somewhere in the system is not immediately made known to the other components with which it communicates.

Examples of distributed systems

- The Internet
  - A vast interconnected collection of computer networks of many different types
  - Programs running on the computers connected to it interact by passing messages.
  - The set of services is open-ended: it can be extended by the addition of server computers and new types of service.
  - Multimedia Services
    - ✓ Enabling users to access audio and video data including music, radio and TV channels
    - ✓ Enabling users to hold phone and video conferences

Examples of distributed systems

– The Intranets

- A portion of the Internet that is separately administered and has a boundary that can be configured to enforce local security policies
- Is composed of several local area networks (LANs) linked by backbone connections
- Firewall – to protect an intranet by preventing unauthorized messages leaving or entering
- Issues in the design of components for use in intranets
  - ✓ File services are needed to enable users to share data
  - ✓ When resource sharing between internal and external users is required, firewalls must be complemented by the use of fine-grained security mechanisms.
  - ✓ The cost of software installation and support

Examples of distributed systems

- Mobile and ubiquitous computing
  - Technological advances in device miniaturization and wireless networking
  - devices
    - ✓ Laptop computers
    - ✓ Handheld devices, such as PDAs, mobile phones, and digital camera
    - ✓ Wearable devices, such as smart watches
    - ✓ Devices embedded in appliances such as washing machines
  - Mobile computing (nomadic computing) is the performance of computing tasks while the user is on the move, or visiting places other than their usual environment.
  - *Location-aware* or *context-aware computing* – there is increasing provision for users to utilize resources such as printers that are conveniently nearby as they move around.

Examples of distributed systems

- Mobile and ubiquitous computing
  - 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 setting.
  - The term 'ubiquitous' is intended to suggest that small computing devices will eventually become so pervasive in everyday objects that they are scarcely noticed.

Challenges - Heterogeneity

- Enables users to access services and run applications over a heterogeneous collection of computers and networks
  - Networks
  - Computer hardware
  - Operating systems
  - Programming languages
  - Implementations by different developers
- Middleware
  - Software layer that provides a programming abstraction as well as masking the heterogeneity of the underlying networks, hardware, operating systems and programming languages
  - CORBA, Java Remote Method Invocation(RMI)
- Mobile code : Code that can be sent from one computer to another and run at the destination - Java applets

Challenges - Openness

- The characteristic that determines whether the system can be extended and re-implemented in various ways
- Determined by the degree to which new resource-sharing services can be added
- The key interfaces are published – standardization of interfaces
- The provision of a uniform communication mechanism and published interfaces for access to shared resources
- Can be constructed from heterogeneous hardware and software, possibly from different vendors

Challenges - Security

- Three components
  - Confidentiality
    - ✓ Protection against disclosure to unauthorized individuals
  - Integrity
    - ✓ Protection against alteration or corruption
  - Availability
    - ✓ Protection against interference with the means to access the resource
- Security challenges
  - Denial of service attacks - by bombarding the service with such a large number of pointless requests
  - Security of mobile code – by receiving an executable program as an electronic mail attachment

Challenges - Scalability

- Remain effective when there is a significant increase in the number of resources and the number of users
- Main challenges
  - Controlling the cost of physical resources
    - ✓ as the demand for a resource grows, it should be possible to extend the system, at reasonable cost, to meet it.
    - ✓ For a system with *n* users to be scalable, the quantity of physical resources required to support them should be at most O(*n*).
  - Controlling the performance loss
    - ✓ Size is proportional to the number of users or resources in the system.
    - ✓ Algorithms that use hierarchic structures scale better than those that use linear structures. But even with hierarchic structures an increase in size will result in some loss in performance.

Challenges - Scalability

– Main challenges

- Preventing software resources running out
  - ✓ Internet (IP) addresses – In the late 1970s, it was decided to use 32 bits for this purpose, but the supply of available Internet addresses is running out. For this reason, a new version of the protocol with 128-bit Internet addresses is being adopted and this will require modifications to many software components.
  - ✓ For a system with *n* users to be scalable, the quantity of physical resources required to support them should be at most O(*n*).
- Avoiding performance bottlenecks
  - ✓ The Domain Name System removed this bottleneck by partitioning the name table between servers located throughout the Internet and administered locally.

Challenges – Failure handling

- Detecting failures
  - Checksums can be used to detect corrupted data in a message or a file.
  - The challenge is to manage in the presence of failures that cannot be detected but may be suspected.
- Masking (hiding) failures
  - Can be retransmitted when they fail to arrive
  - Can be written to a pair of disks so that if one is corrupted, the other may still be correct
- Tolerating failures
  - Can be designed to tolerate failures
- Recovery from failures
  - Can be recovered or rolled back after a server has crashed
  - The computations performed by some programs will be incomplete when a fault occurs.

Challenges – Failure handling

- Redundancy
  - By the use of redundant components
  - Examples
    - ✓ Be at least two different routes between any two routers in the Internet
    - ✓ In the Domain Name System, every name table is replicated in at least two different servers.
    - ✓ A database may be replicated in several servers.
- Availability
  - A measure of the proportion of time that it is available for use

Challenges – Concurrency

- A possibility that several clients in a distributed system
- The process that manages a shared resource could take one client request at a time. => limitation of throughput
- Allow multiple client requests to be processed concurrently
- Operations must be synchronized in such a way that its data remains consistent.
- Can be achieved by standard techniques such as semaphores

Challenges – Transparency

- Concealment from the user and the application programmer
- 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
  - Web resource names or URLs
- 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

Challenges – Transparency

- Failure transparency
  - Enables the concealment of faults
  - The context of e-mail is eventually delivered. (by retransmitting)
- Mobility transparency
  - Allows the movement of resources and clients without affecting the operation of users or programs (ex) mobile phone
- 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
- Network transparency
  - Access and location transparency (ex) electronic mail address