1 Characterization of distributed systems

1.1 Introduction

Weeks:1-2

What is a Distributed System?

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

A distributed system consists of a **collection of** autonomous **computers linked by** a computer **network** and equipped with **distributed system software**. This software enables computers to **coordinate** their activities and to **share the resources of the system hardware, software, and data.**

How to characterize a distributed system?

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

Leslie Lamport :-)

You know you have a distributed system when the crash of a computer you've never heard of stops you from getting any work done!

Prime motivation: **to share resources**

What are the challenges?

- heterogeneity of their components
- openness
- security
- scalability – the ability to work well when the load or the number of users increases
- failure handling
- concurrency of components
- transparency
- providing quality of service

1.2 Examples of distributed systems

Distributed Systems application domains connected with networking:

Finance and commerce	eCommerce e.g. Amazon and eBay, PayPal, online banking and trading
The information society	Web information and search engines, ebooks, Wikipedia; social networking: Facebook and MySpace
Creative industries and entertainment	online gaming, music and film in the home, user-generated content, e.g. YouTube, Flickr
Healthcare	health informatics, on online patient records, monitoring patients
Education	e-learning, virtual learning environments; distance learning
Transport and logistics	GPS in route finding systems, map services: Google Maps, Google Earth
Science	The Grid as an enabling technology for collaboration between scientists
Environmental management	sensor technology to monitor earthquakes, floods or tsunamis

1.2.1 Web search

An example: Google

Highlights of this infrastructure:

- physical infrastructure
- distributed file system
- structured distributed storage system
- lock service
- programming model

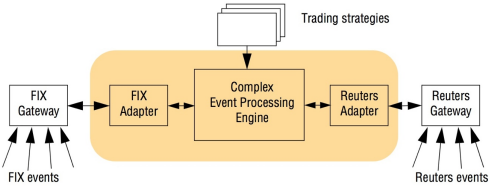
1.2.2 Massively multiplayer online games (MMOGs)

Examples

- EVE online – *client-server architecture*!
- EverQuest – more distributed architecture
- Research on completely decentralized approaches based on *peer-to-peer (P2P) technology*

1.2.3 Financlial trading

- *distributed even-based systems*



- Reuters market data events

- FIX events (events following the specific format of the Financial Information eXchange protocol)

```
WHEN
        MSFT price moves outside 2% of MSFT Moving
            Average
FOLLOWED-BY (
        MyBasket moves up by 0.5%
        AND (
                HPQ-s price moves up by 5%
                OR
                MSFT-s price moves down by 2%
        )
)
ALL WITHIN
        any 2 minute time period
THEN
        BUY MSFT
        SELL HPQ
```

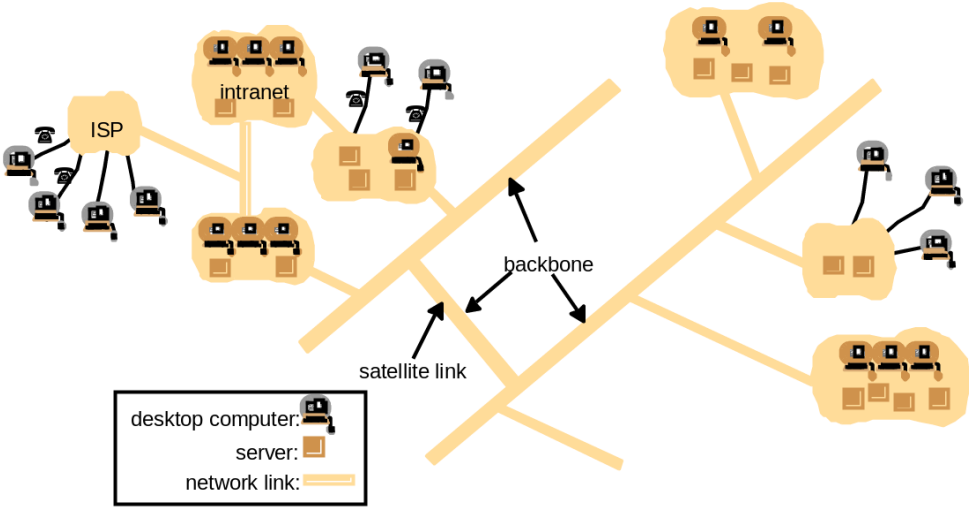
1.3 Trends in distributed systems

- emergence of pervasive networking technology
- emergence of ubiquitous computing coupled with the desire to support user mobility
- multimedia services
- distributed systems as utility

1.3.1 Pervasive networking and the modern Internet

networking has become a pervasive resource and devices can be conected at any time and any place

A typical portion of the Internet:

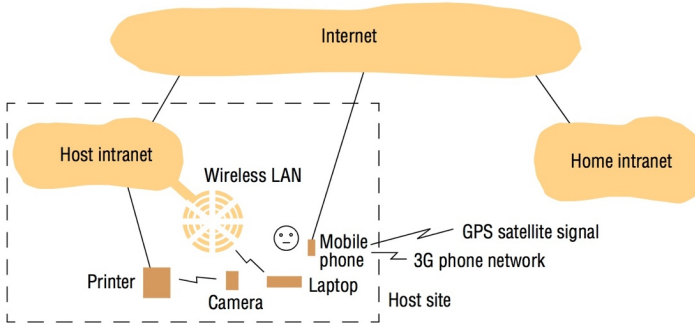


1.3.2 Mobile and ubiquitous computing

- laptop computers
- handheld devices (mobile phones, smart phones, tablets, GPS-enabled devices, PDAs, video and digital cameras)
- wearable devices (smart watches, glasses, etc.)
- devices embedded in appliances (washing machines, refrigerators, cars, etc.)

Portable and handheld devices in a distributed system

- mobile computing
- location/context-aware computing
- ubiquitous computing
- spontaneous interoperation
- service discovery

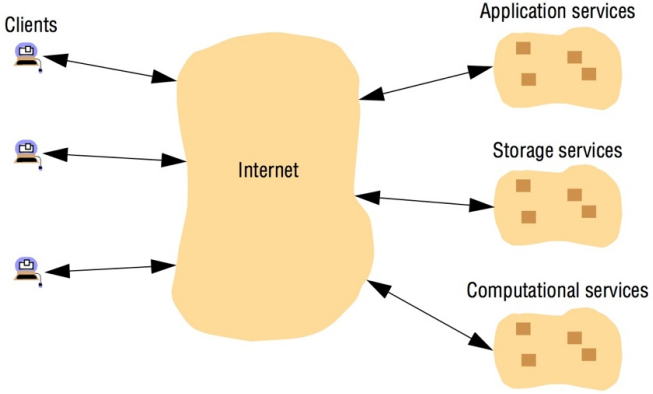


1.3.3 Distributed multimedia systems

- live or pre-ordered television broadcasts
- video-on-demand
- music libraries
- audio and video conferencing

1.3.4 Distributed computing as a utility

- Cluster computing
- Grid computing
- Cloud computing



1.4 Sharing resources

What are the resources?

- Hardware
  - Not every single resource is for sharing
- Data
  - Databases
  - Proprietary software
  - Software production
  - Collaboration

Sharing Resources

- Different resources are handled in different ways, there are however some generic requirements:
  - Namespace for identification
  - Name translation to network address
  - Synchronization of multiple access

1.5 Challenges

1.5.1 Heterogeneity

Heterogeneity – variety and difference in:

- networks
- computer hardware
- OS
- programming languages
- implementations by different developers

Middleware

- *middleware* – software layer providing:
  - programming abstraction
  - masking heteorogeneity of:
    - underlying networks
    - hardware
    - operating systems

Heterogeneity and mobile code

Mobile code – programming code that can be transferred from one computer to another and run at the destination (Example: think Java applets)

***Virtual machine* approach** – way of making code executable on a variety of host computers – the compiler for a particular language generates code for a virtual machine instead of a particular hardware order code.

1.5.2 Openness

OPENNESS of a:

computer system - can the system be extended and reimplemented in various ways?

distributed system - can new resource-sharing services be added and made available for use by variety of client programs?

An open system – key interfaces need to be published!

An open distributed system has:

- uniform communication mechanism
- published interfaces to shared resources

Open DS - heterogeneous hardware and software, possibly from different vendors, but conformance of each component to published standard must be tested and verified for the system to work correctly

1.5.3 Security

1. ***Confidentiality*** – protection against disclosure to unauthorized individuals
2. ***Integrity*** – protection against alteration or corruption
3. ***Availability*** – protection against interference with the means to access the resources

Security challenges not yet fully met:

- *denial of service attacks*
- *security of mobile code*

1.5.4 Scalability

– the ability to work well when the system load or the number of users increases

Challanges with building scalable distributed systems:

- Controlling the cost of physical resources
- Controlling the performance loss
- Preventing software resources running out (like 32-bit internet addresses, which are being replaced by 128 bits)
- Avoiding performance bottlenecks
  - Example: some web-pages accessed very frequently – remedy: *caching* and *replication*

1.5.5 Failure handling

Techniques for dealing with failures

- Detecting failures
- Masking failures
    1. messages can be retransmitted
    2. disks can be replicated in a synchronous action
- Tolerating failures
- Recovery from failures

- Redundancy
  - redundant components
    1. at least two different routes
    2. like in DNS every name table replicated in at least two different servers
    3. database can be replicated in several servers

Main goal: **High *availability*** – measure of the proportion of time that it is available for use

1.5.6 Concurrency

Example: Several clients trying to access shared resource at the same time

Any object with shared resources in a DS must be responsible that it operates correctly in a concurrent environment

Discussed in Chapters 7 and 17 in the book

1.5.7 Transparency

Transparency – concealment from the user and the application programmer of the separation of components in a Distributed System for the system to be perceived as a whole rather than a collection of independent components

- **Acess transparency** – access to local and remote resources identical
- **Location transparency** – resources accessed without knowing their physical or network location
- Concurrency transparency – concurrent operation of processes using shared resources without interference between them
- Replication transparency – multiple instances seem like one
- Failure transparency – fault concealment
- Mobility transparency – movement of resources/clients within a system without affecting the operation of users or programs

Access and Location transparancy – together called also **Network transparency**

1.5.8 Quality of service

Main nonfunctional properties of systems that affect *Quality of Service (QoS):*

- **reliability**
- **security**
- **performance**

Time-critical data transfers

Additional property to meet changing system configuration and resource availability:

- **adaptability**

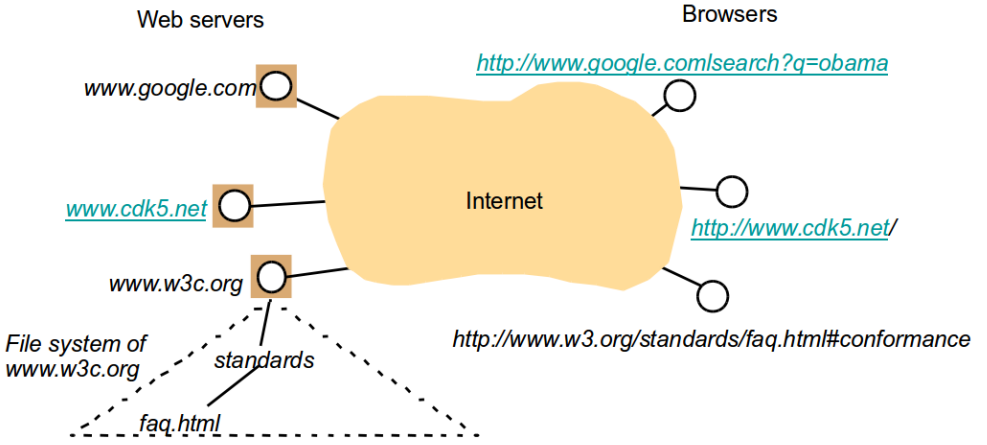
1.6 Case study: The World Wide Web

CERN 1989

hypertext structure, *hyperlinks*

- Web is an open system
- content standards freely published and widely implemented
- Web is open with respect to types

Figure 1.7 Web servers and web browsers



HTML

HyperText Markup Language www.w3.org

URL-s

Uniform Resource Locators (also known as URI-s - Uniform Resourse Identifiers)

http://servername[:port][/pathName][?query][#fragment]

HTTP

- Request-reply interactions
- Content types
- One resource per request
- Simple access control
- Dynamic pages

Web services

HTML – limited – not extensible to applications beyond information browsing

The Extensible Markup language (XML) designed to represent data in standard, structured, application-specific way

XML data can be transmitted by POST and GET operations

- Semantic web – web of linked metadata resources

Web as a system – main problem – the problem of scale

End of week 1