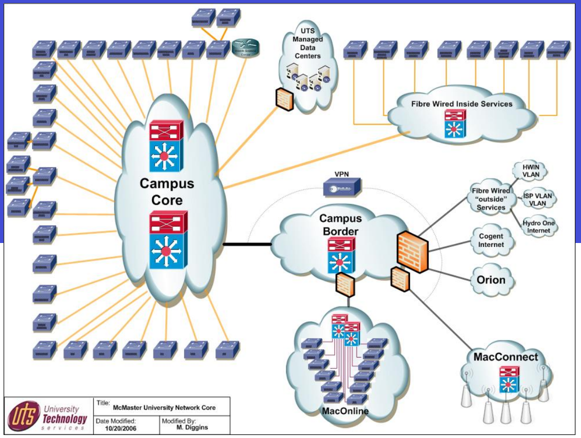
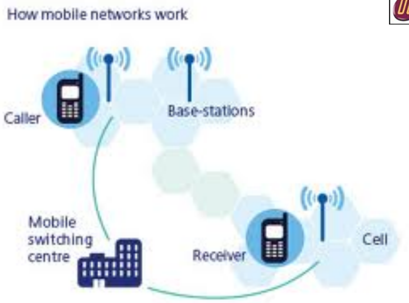
Presentation Outline

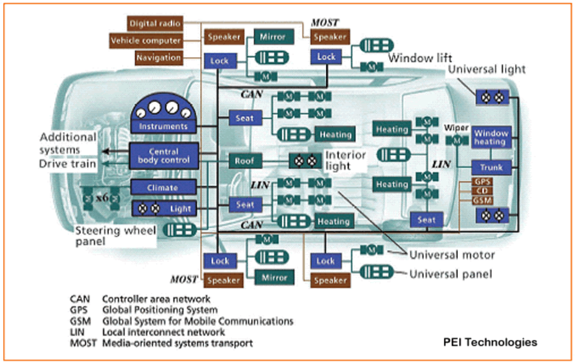
- Introduction
- Defining Distributed Systems
- Characteristics of Distributed Systems
- Example Distributed Systems
- Challenges of Distributed Systems
- Summary

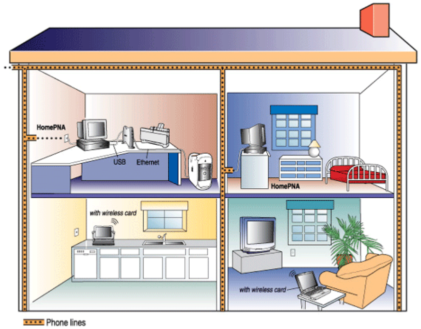
Introduction


UTS Managed Data Centers
Fibre Wired Inside Services
Campus Core
VPN
Campus Border
HWIN VLAN
Fibre Wired "outside" Services
ISP VLAN
Hydro One Internet
Cogent Internet
Orion
MacConnect
MacOnline
University Technology Services
Title: McMaster University Network Core
Date Modified: 10/20/2006
Modified By: M. Diggins

- Networks of computers are everywhere!
  - Mobile phone networks
  - Corporate networks
  - Factory networks
  - Campus networks
  - Home networks
  - In-car networks
  - On board networks in planes and trains
- This subject aims:
  - to cover characteristics of networked computers that impact system designers and implementers, and
  - to present the main concepts and techniques that have been developed to help in the tasks of designing and implementing systems and applications that are based on them (networks).





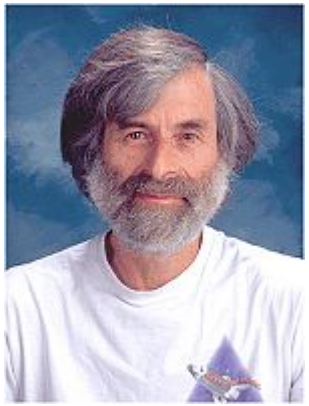


Defining Distributed Systems

- *"A system in which hardware or software components located at networked computers communicate and coordinate their actions only by message passing."* [Coulouris]
- *"A distributed system is a collection of independent computers that appear to the users of the system as a single computer."* [Tanenbaum]
- Example Distributed Systems:
  - Cluster:
    - *"A type of parallel or distributed processing system, which consists of a collection of interconnected stand-alone computers cooperatively working together as a single, integrated computing resource" [Buyya].*
  - Cloud:
    - *"a type of parallel and distributed system consisting of a collection of interconnected and **virtualised computers** that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers" [Buyya].*

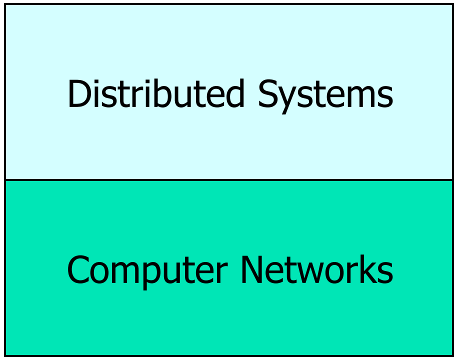
Leslie Lamport's Definition

- *"A distributed system is one on which I cannot get any work done because some machine I have never heard of has crashed."*
  - Leslie Lamport – a famous researcher on timing, message ordering, and clock synchronization in distributed systems.

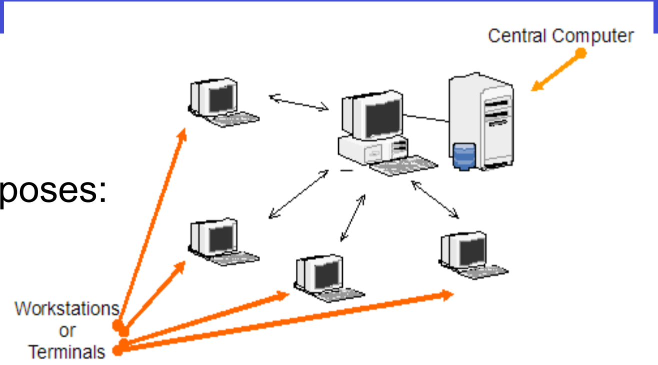


Networks vs. Distributed Systems

- Networks: A media for interconnecting local and wide area computers and exchange messages based on protocols. Network entities are visible and they are explicitly addressed (IP address).
- Distributed System: existence of multiple autonomous computers is transparent
- However,
  - many problems (e.g., openness, reliability) in common, but at different levels.
    - Networks focuses on packets, routing, etc., whereas distributed systems focus on applications.
    - Every distributed system relies on services provided by a computer network.



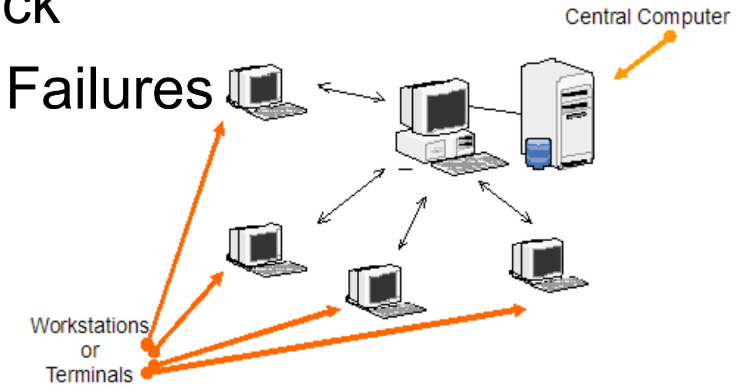
Reasons for Distributed Systems



- Functional Separation:
  - Existence of computers with different capabilities and purposes:
    - Clients and Servers
    - Data collection and data processing
- Inherent distribution:
  - Information:
    - Different information is created and maintained by different people (e.g., Web pages)
  - People
    - Computer supported collaborative work (virtual teams, engineering, virtual surgery)
  - Retail store and inventory systems for supermarket chains (e.g., Coles, Woolworths)
- Power imbalance and load variation:
  - Distribute computational load among different computers.
- Reliability:
  - Long term preservation and data backup (replication) at different locations.
- Economies:
  - Sharing a printer by many users and reduce the cost of ownership.
  - Building a supercomputer out of a network of computers.

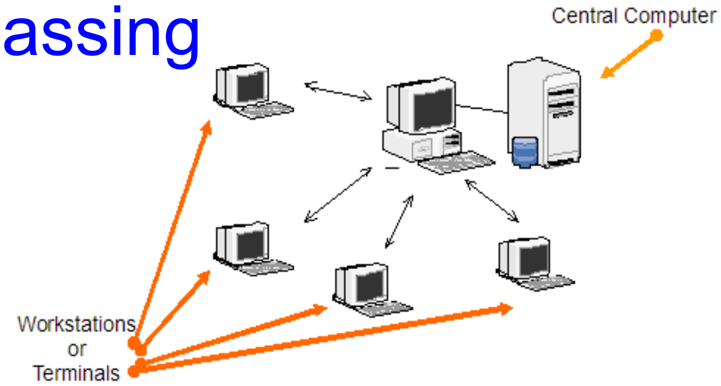
Consequences of Distributed Systems

- Computers in distributed systems may be on separate continents, in the same building, or the same room. DSs have the following consequences:
  - Concurrency – each system is autonomous.
    - Carry out tasks independently
    - Tasks coordinate their actions by exchanging messages.
  - Heterogeneity
  - No global clock
  - Independent Failures



Characteristics of Distributed Systems

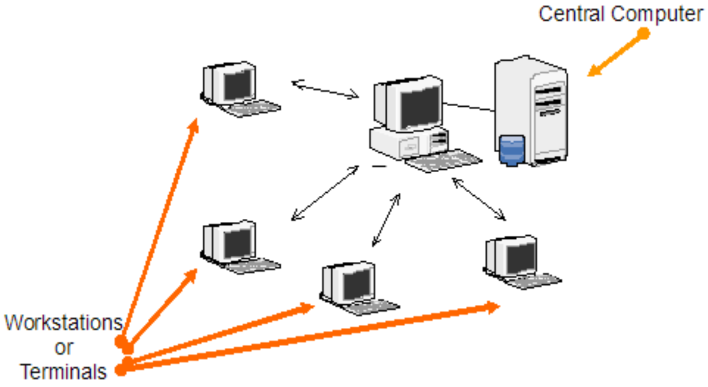
- Parallel activities
  - Autonomous components executing concurrent tasks
- Communication via message passing
  - No shared memory
- Resource sharing
  - Printer, database, other services
- No global state
  - No single process can have knowledge of the current global state of the system
- No global clock
  - Only limited precision for processes to synchronize their clocks

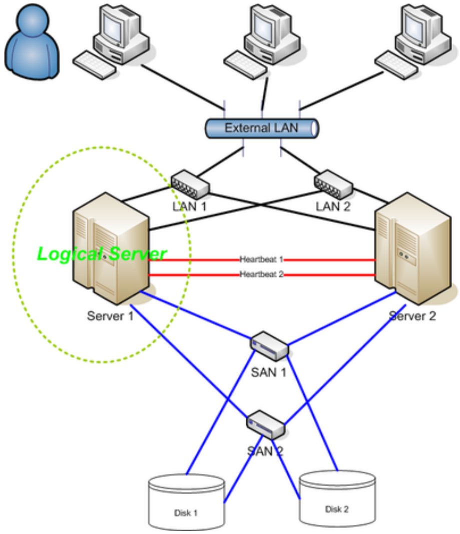


Goals of Distributed Systems

- Connecting Users and Resources
- Transparency
- Openness
- Scalability
- Enhanced Availability

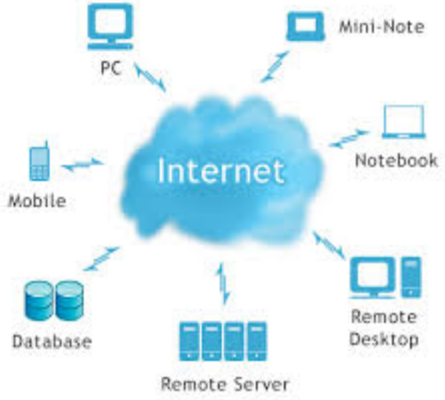
Grid Computing
Grid Computing is all about sharing, aggregating hosting, offering service across the world for the benefits of man kind.
"grid is need is a grid indeed"
adarsh



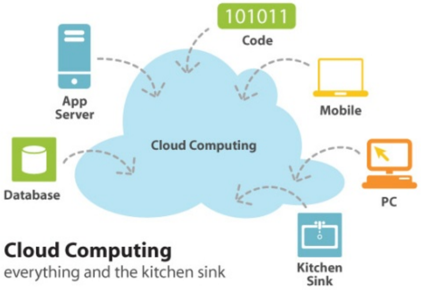


Examples of Distributed Systems

- They (DS) are based on familiar and widely used computer networks:
  - Internet
  - Intranets, and
  - Wireless networks



- Example DS:
  - Web (and many of its applications like Online bookshop)
  - Data Centers and Clouds
  - Wide area storage systems
  - Banking Systems
  - User-level communication (Facebook, Skype)



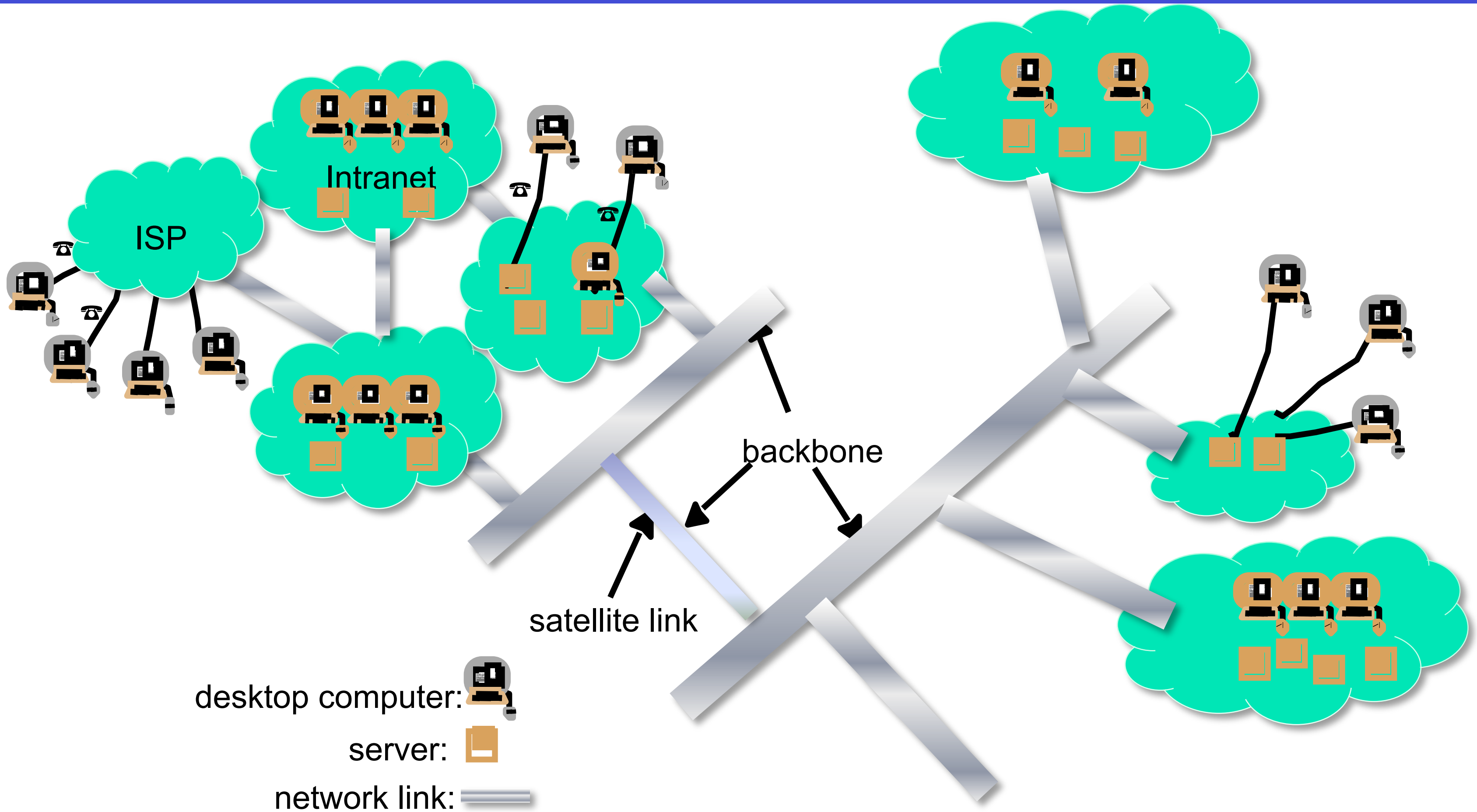
Cloud Computing
everything and the kitchen sink

Selected application domains and associated networked applications

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	Grid computing as an enabling technology for collaboration between scientists
Environmental management	sensor technology to monitor earthquakes, floods or tsunamis

A typical portion of the Internet and its services:

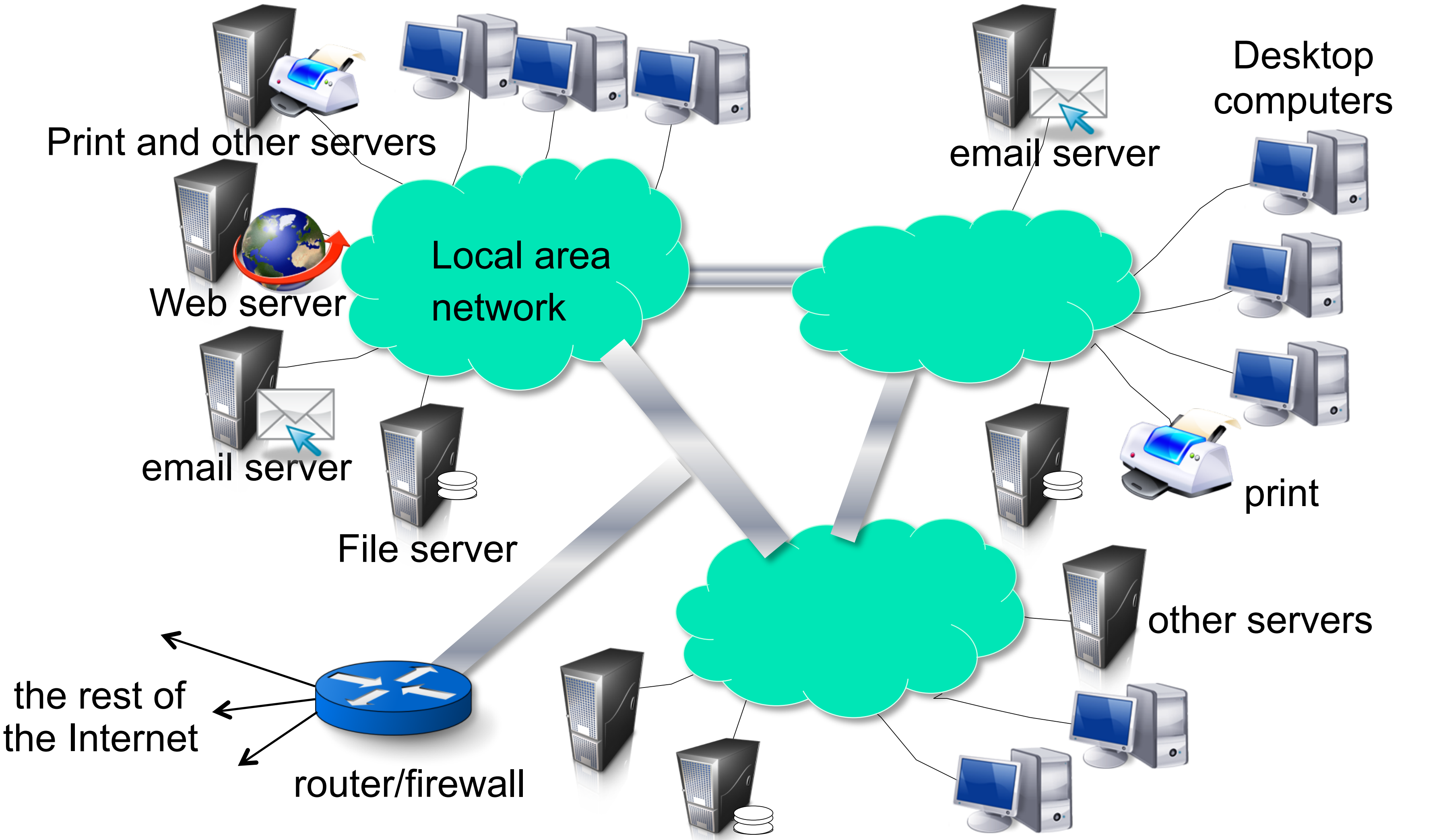
Multimedia services providing access to music, radio, TV channels, and video conferencing supporting several users.



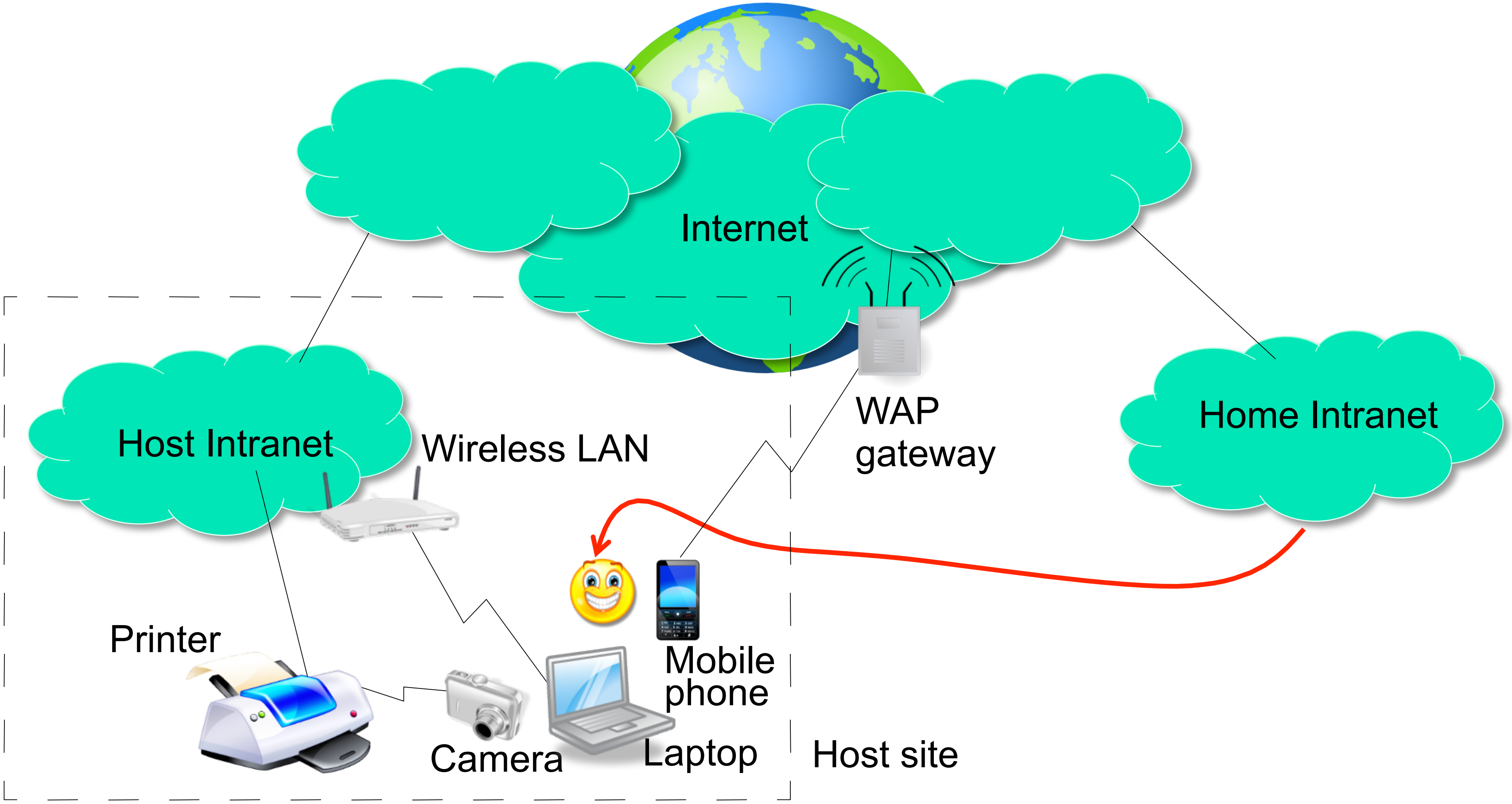
- The Internet is a vast collection of computer networks of many different types and hosts various types of services.

A typical Intranet:

A portion of Internet that is separately administered & supports internal sharing of resources (file/storage systems and printers)

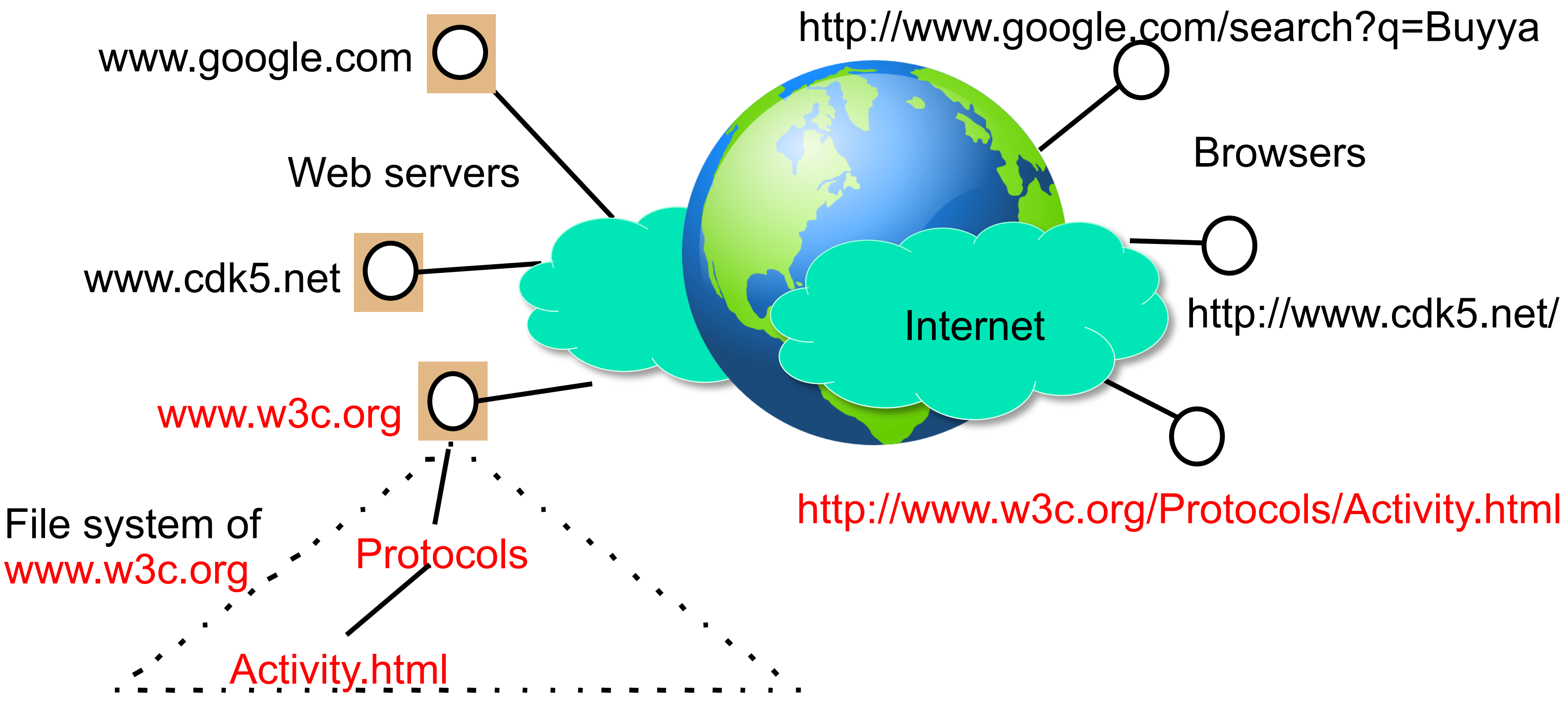


Mobile and ubiquitous computing: portable and handheld devices in a distributed system



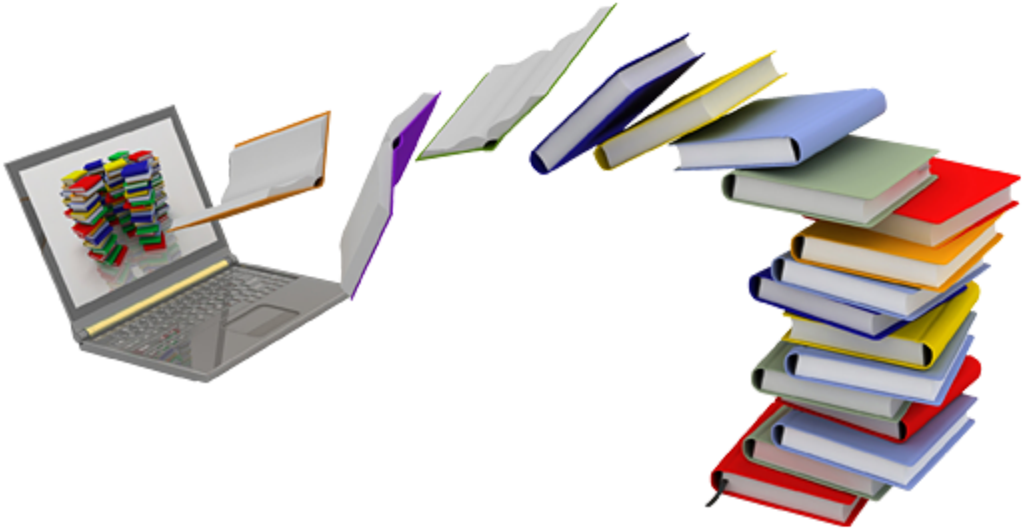
- Supports continued access to Home intranet resources via wireless and provision to utilise resources (e.g., printers) that are conveniently located (location-aware computing).

Resource sharing and the Web: open protocols, scalable servers, and pluggable browsers



Business Example and Challenges

- Online bookstore (e.g. in World Wide Web)
  - Customers can connect their computer to your computer (web server):
    - Browse your inventory
    - Place orders
    - …



This example has been adapted from **Torbin Weis**, Berlin University of Technology

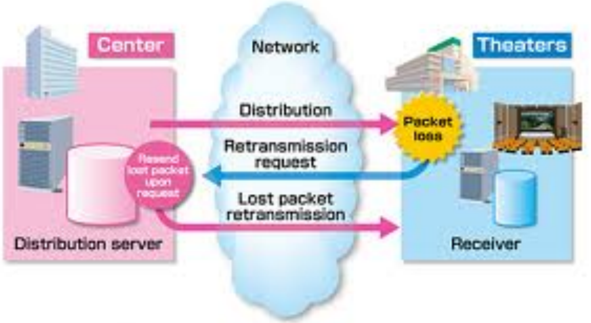
Business Example – Challenges I

- What if
  - Your customer uses a completely different hardware? (PC, MAC,…)
  - … a different operating system? (Windows, Unix,…)
  - … a different way of representing data? (ASCII, EBCDIC, …)
  - **Heterogeneity**
- Or
  - You want to move your business and computers to the Caribbean (because of the weather)?
  - Your client moves to the Caribbean (more likely)?
  - **Distribution transparency**

Business Example – Challenges II

- What if
  - Two customers want to order the same item at the same time?
  - **Concurrency**
- Or
  - The database with your inventory information crashes?
  - Your customer’s computer crashes in the middle of an order?
  - **Fault tolerance**

Business Example – Challenges III



Reliable and secure distribution

- What if
  - Someone tries to break into your system to steal data?
  - … sniffs for information?
  - … your customer orders something and doesn’t accept the delivery saying he didn’t?
  - **Security**
- Or
  - You are so successful that millions of people are visiting your online store at the same time?
  - **Scalability**

Business Example – Challenges IV

- When building the system…
  - Do you want to write the whole software on your own (network, database,…)?
  - What about updates, new technologies?
  - **Reuse** and **Openness** (Standards)

Overview Challenges I

- Heterogeneity
  - Heterogeneous components must be able to interoperate
- Distribution transparency
  - Distribution should be hidden from the user as much as possible
- Fault tolerance
  - Failure of a component (partial failure) should not result in failure of the whole system
- Scalability
  - System should work efficiently with an increasing number of users
  - System performance should increase with inclusion of additional resources

Overview Challenges II

- Concurrency
  - Shared access to resources must be possible
- Openness
  - Interfaces should be publicly available to ease inclusion of new components
- Security
  - The system should only be used in the way intended

Heterogeneity

- Heterogeneous components must be able to interoperate across different:
  - Operating systems
  - Hardware architectures
  - Communication architectures
  - Programming languages
  - Software interfaces
  - Security measures
  - Information representation

Distribution Transparency I

- To hide from the user and the application programmer the separation/distribution of components, so that the system is perceived as a whole rather than a collection of independent components.
- ISO Reference Model for Open Distributed Processing (ODP) identifies the following forms of transparencies:
- Access transparency
  - Access to local or remote resources is identical
  - E.g. Network File System / **Dropbox**
- Location transparency
  - Access without knowledge of location
  - E.g. separation of domain name from machine address.
- Failure transparency
  - Tasks can be completed despite failures
  - E.g. message retransmission, failure of a Web server node should not bring down the website.

Distribution Transparency II

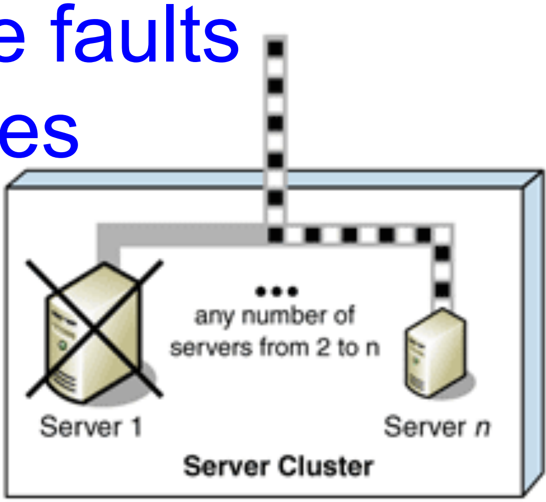
- Replication transparency
  - Access to replicated resources as if there was just one. And provide enhanced reliability and performance without knowledge of the replicas by users or application programmers.
- Migration (mobility/relocation) transparency
  - Allow the movement of resources and clients within a system without affecting the operation of users or applications.
  - E.g. switching from one name server to another at runtime; migration of an agent/process from one node to another.

Distribution Transparency III

- Concurrency transparency
  - A process should not notice that there are other sharing the same resources
- Performance transparency:
  - Allows the system to be reconfigured to improve performance as loads vary
  - E.g., dynamic addition/deletion of components, switching from linear structures to hierarchical structures when the number of users increase
- Scaling transparency:
  - Allows the system and applications to expand in scale without changes in the system structure or the application algorithms.
- Application level transparencies:
  - Persistence transparency
    - Masks the deactivation and reactivation of an object
  - Transaction transparency
    - Hides the coordination required to satisfy the transactional properties of operations

Fault Tolerance

- Failure: an offered service no longer complies with its specification (e.g., no longer available or very slow to be usable)
- Fault: cause of a failure (e.g. crash of a component)
- Fault tolerance: no failure despite faults i.e., programmed to handle failures and hides them from users.



Fault Tolerance Mechanisms

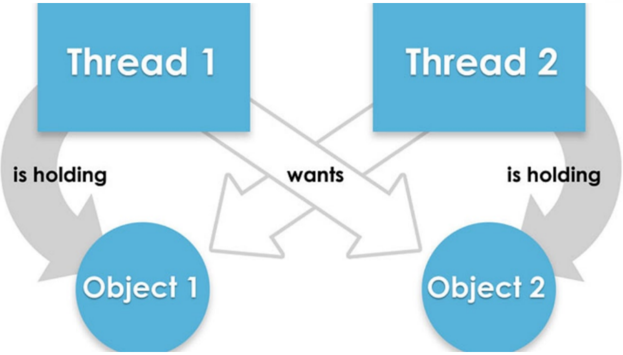
- Fault detection
  - Checksums, heartbeat, …
- Fault masking
  - Retransmission of corrupted messages, redundancy, …
- Fault toleration
  - Exception handling, timeouts,…
- Fault recovery
  - Rollback mechanisms,…

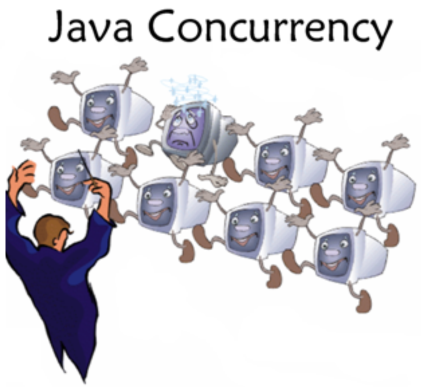
Scalability

- System should work efficiently at many different scales, ranging from a small Intranet to the Internet
- Remains effective when there is a significant increase in the number of resources and the number of users
- Challenges of designing scalable distributed systems:
  - Cost of physical resources
    - Cost should linearly increase with system size
  - Performance Loss
    - For example, in hierarchically structure data, search performance loss due to data growth should not be beyond O(*log n*), where n is the size of data
  - Preventing software resources running out:
    - Numbers used to represent Internet addresses (32 bit->64bit)
    - Y2K-like problems
  - Avoiding performance bottlenecks:
    - Use of decentralized algorithms (centralized DNS to decentralized)

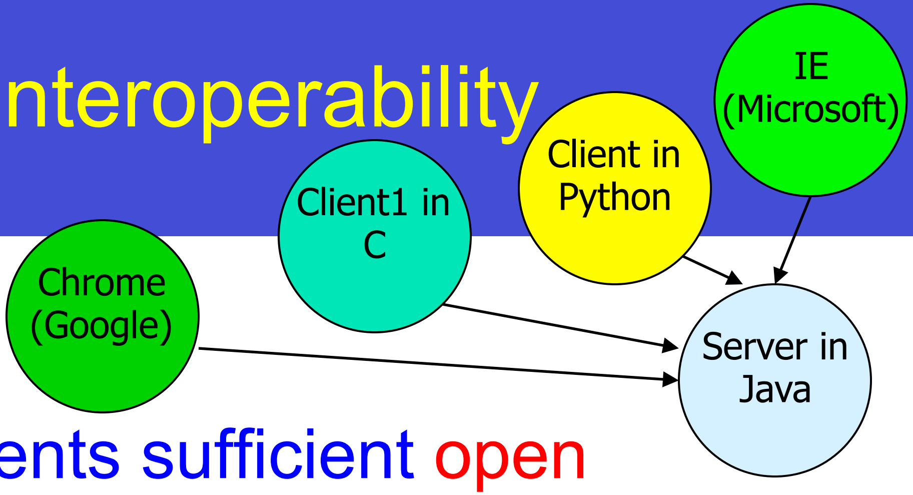
Concurrency

- Provide and manage concurrent access to shared resources:
  - Fair scheduling
  - Preserve dependencies (e.g. distributed transactions -- buy a book using Credit card, make sure user has sufficient funds prior to finalizing order)
  - Avoid deadlocks





Openness and Interoperability



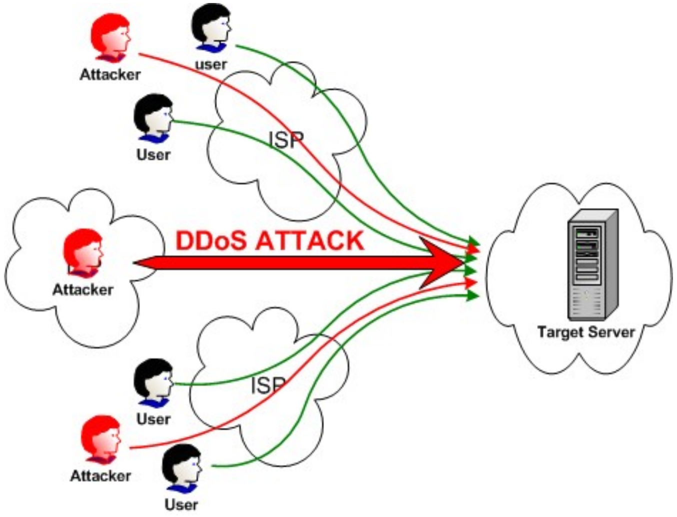
- Open system:
  "... a system that implements sufficient open specifications for interfaces, services, and supporting formats to enable properly engineered applications software to be ported across a wide range of systems with minimal changes, to interoperate with other applications on local and remote systems, and to interact with users in a style which facilitates user portability" (POSIX Open Systems Environment, IEEE POSIX 1003.0)
- Open spec/standard developers - communities:
  - ANSI, IETF, W3C, ISO, IEEE, OMG, Trade associations,...

Security I

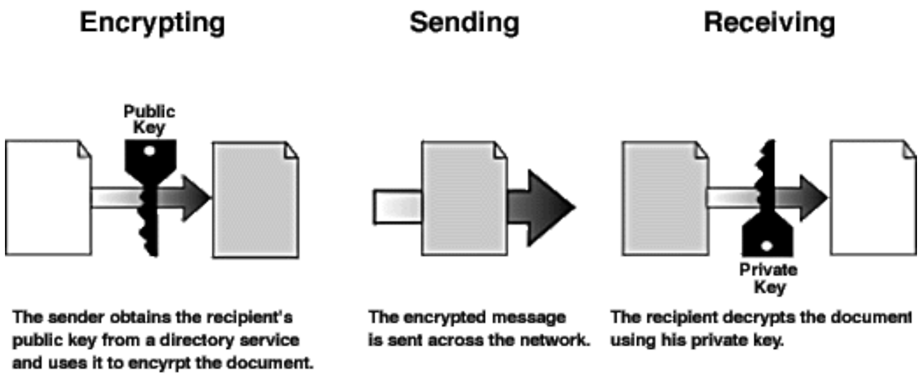
- Resources are accessible to authorized users and used in the way they are intended
- Confidentiality
  - Protection against disclosure to unauthorized individual information
  - E.g. ACLs (access control lists) to provide authorized access to information
- Integrity
  - Protection against alteration or corruption
  - E.g. changing the account number or amount value in a money order

Security II

- Availability
  - Protection against interference targeting access to the resources.
  - E.g. denial of service (DoS, DDoS) attacks
- Non-repudiation
  - Proof of sending / receiving an information
  - E.g. digital signature



Security Mechanisms


Encrypting
Sending
Receiving
Public Key
Private Key
The sender obtains the recipient's public key from a directory service and uses it to encyrpt the document.
The encrypted message is sent across the network.
The recipient decrypts the document using his private key.

- Encryption
  - E.g. Blowfish, RSA
- Authentication
  - E.g. password, public key authentication
- Authorization
  - E.g. access control lists

Summary

- Distributed Systems are everywhere
- Internet enables users throughout the world to access its (application) services from anywhere
- Resource sharing is the main motivating factor for constructing distributed systems
- Construction of DS produces many challenges:
  - Heterogeneity, Openness, Security, Scalability, Failure handling, Concurrency, and Transparency
- Distributed systems enable globalization:
  - Community (Virtual teams, organizations, social networks)
  - Science (e-Science)
  - Business (..e-Banking..)
  - Entertainment (YouTube, e-Friends)