DISTRIBUTED COMPUTING

Ani Sunny

MODULE 1:

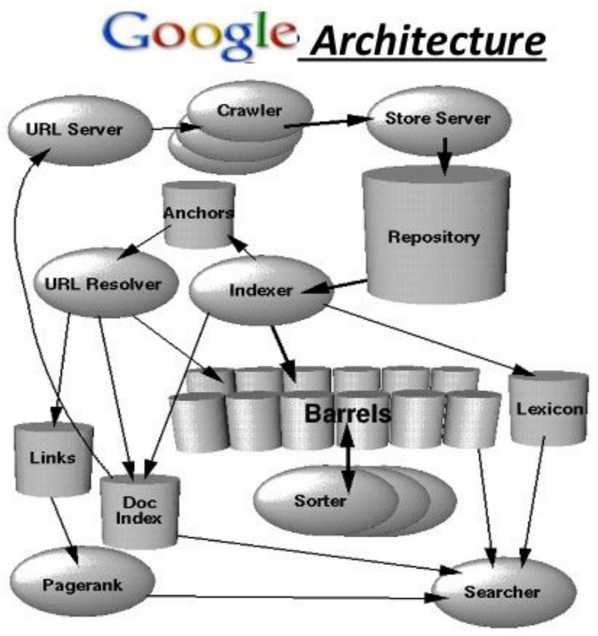
Evolution of Distributed Computing -Issues in designing a distributed system- Challenges- Minicomputer model – Workstation model - Workstation-Server model– Processor - pool model - Trends in distributed systems

What is Distributed Computing?

- studies **distributed systems**
- use of distributed systems to solve computational problems
- In distributed computing,
  - a problem is divided into many tasks
  - **each solved by one or more computers**
  - which communicate with each other via message passing
- Different types of implementations for the message passing mechanism:
  - pure HTTP
  - RPC
  - message queues
- **Distributed program:** A computer program that runs within a distributed system

WEB SEARCH

- **physical infrastructure consisting of very large numbers of** networked computers
- **a distributed file system**
- **an associated structured distributed storage system**
- **a lock service that offers distributed system functions**
- **a programming model for very large parallel and** distributed computations



Various Data Structures Used In Google

❖Repository
❖Lexicon
❖Document Index
❖Hit Lists
❖Forward Index
❖Inverted Index

MASSIVELY MULTIPLAYER ONLINE GAMES (MMOGS)

Features

- Complex playing and multifarious social and economic systems.
- Able to support over large number of simultaneous online players
- 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**.

Underlying Distributed System can be of different types:

- Client-server architecture where a single copy of the state of the world maintained centralized server

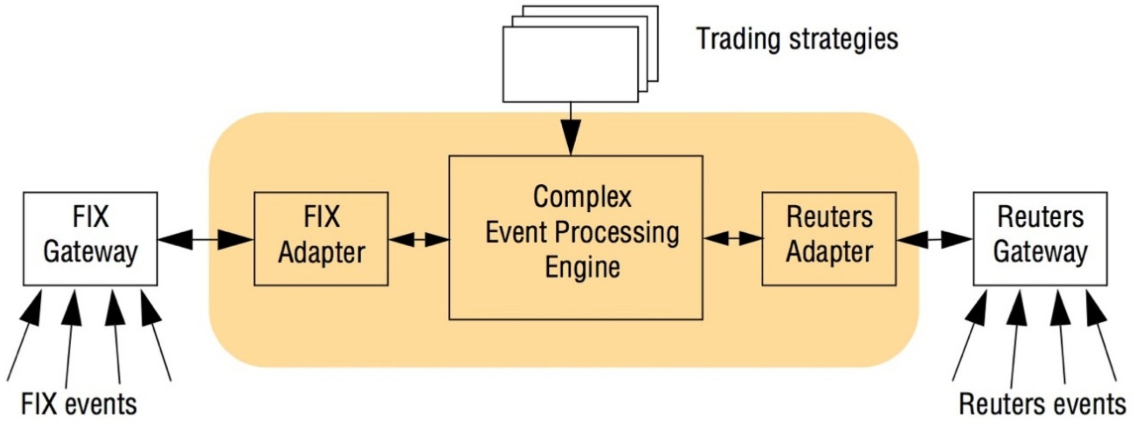
- Distributed universe is partitioned across a (potentially very large) number of servers that may also be geographically distributed.

- Completely decentralized approaches based on peer-to-peer technology

FINANCIAL TRADING

DISTRIBUTED EVENT-BASED SYSTEMS

- Events: Communication and processing of items of interest. Eg. Drop in a share price
- need to deliver events reliably and in a timely manner to potentially very large numbers of clients

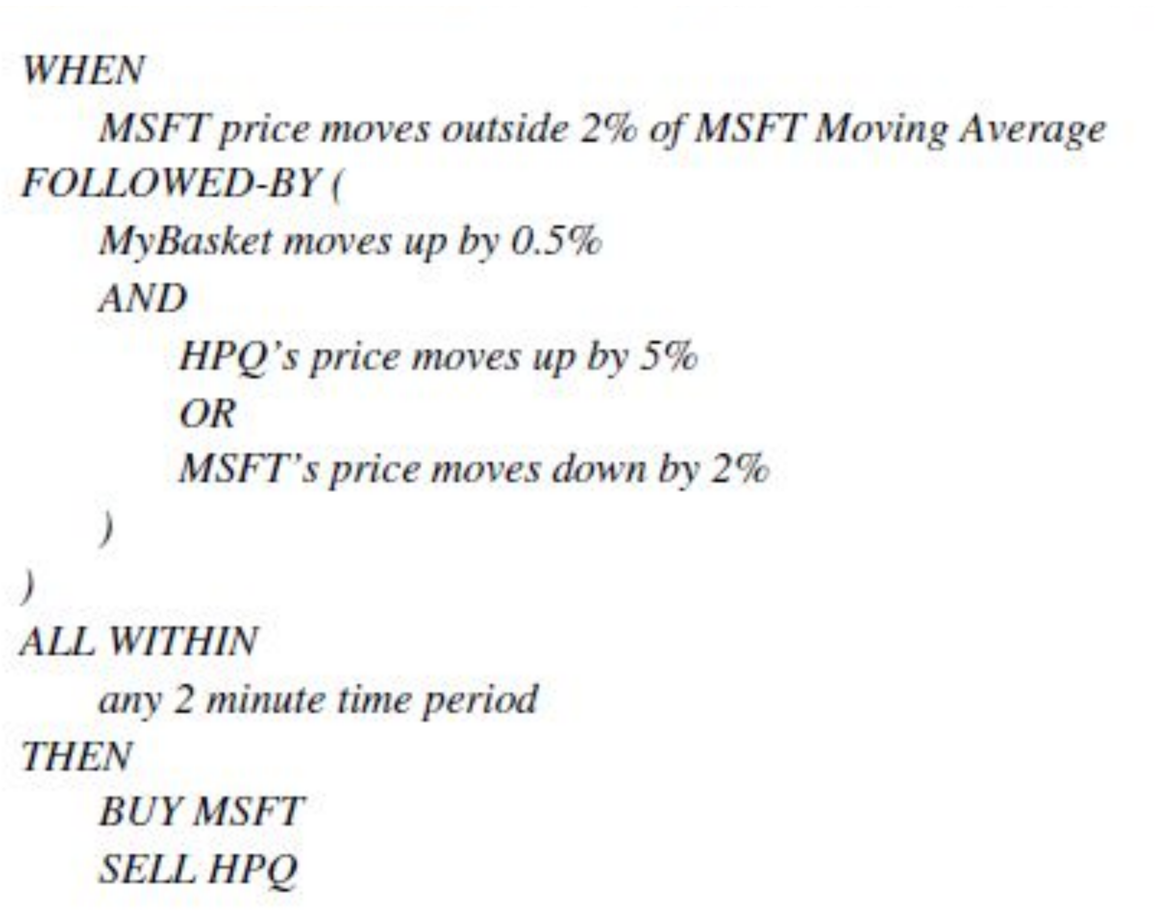


FINANCIAL TRADING SYSTEM

Financial Information eXchange protocol

Complex Event Processing (CEP)

- Buying and selling of stocks and shares, in particular looking for patterns that indicate a trading opportunity and then automatically responding by placing and managing orders

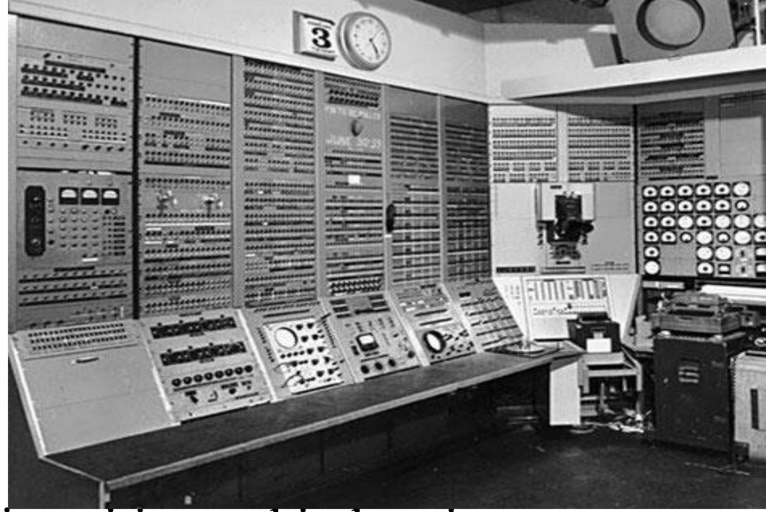


```
WHEN
    MSFT price moves outside 2% of MSFT Moving Average
FOLLOWED-BY (
    MyBasket moves up by 0.5%
    AND
        HPQ's price moves up by 5%
        OR
        MSFT's price moves down by 2%
    )
)
ALL WITHIN
    any 2 minute time period
THEN
    BUY MSFT
    SELL HPQ
```

APPLICATION DOMAINS

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

EVOLUTION



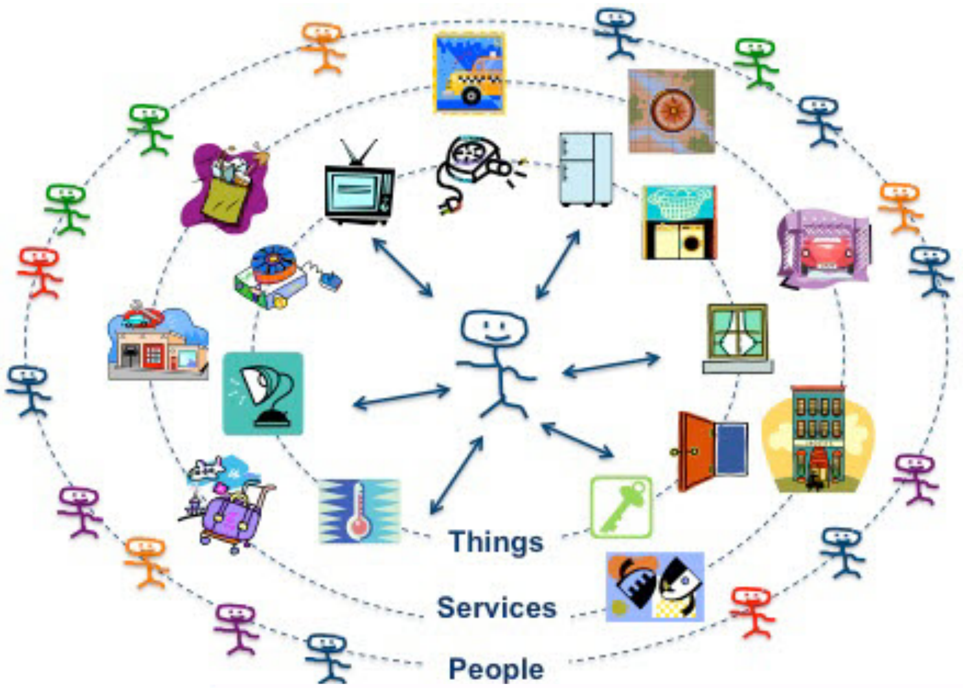
- First computers in research laboratories of universities and industries.
  - Run from console by an operator; programs on punched cards
  - **Set up** necessary environment before processing
  - Printed output
- Batch processing
  - Batching together jobs with similar needs before processing
- Automatic Job Sequencing
  - Using control cards to define beginning and end of a job
- Off-line Processing
  - Overlap of CPU an I/O operations by executing them in separate independent machines.
  - Improves CPU utilization
- Multiprogramming
  - Organising jobs so that CPU always has something to do

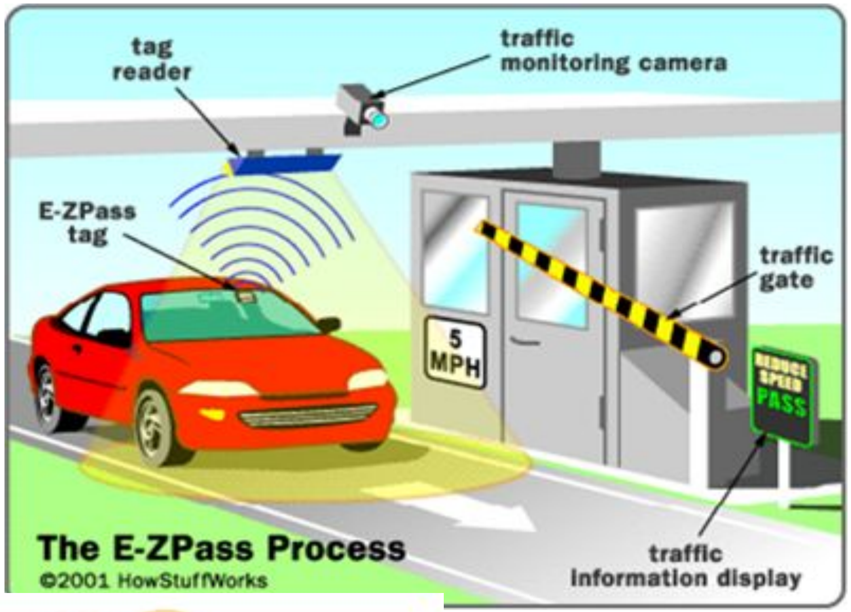
- Time sharing
  - Earlier: dumb terminals attached to main computer. (1970s)
  - Sharing of resources simultaneously
  - Access computers from a different place
  - Minicomputers: Advancements in hardware, reduction in size and increased speed.
  - Dumb terminals replaced by intelligent terminals
  - Workstations: Single user computers (1980s)
    - Xerox PARC- high resolution monochrome display,128KB mem, 2.5MB hard disk, microprogrammed CPU executes machine level instructions 2-6 microsecond speed.
    - IBM 5100

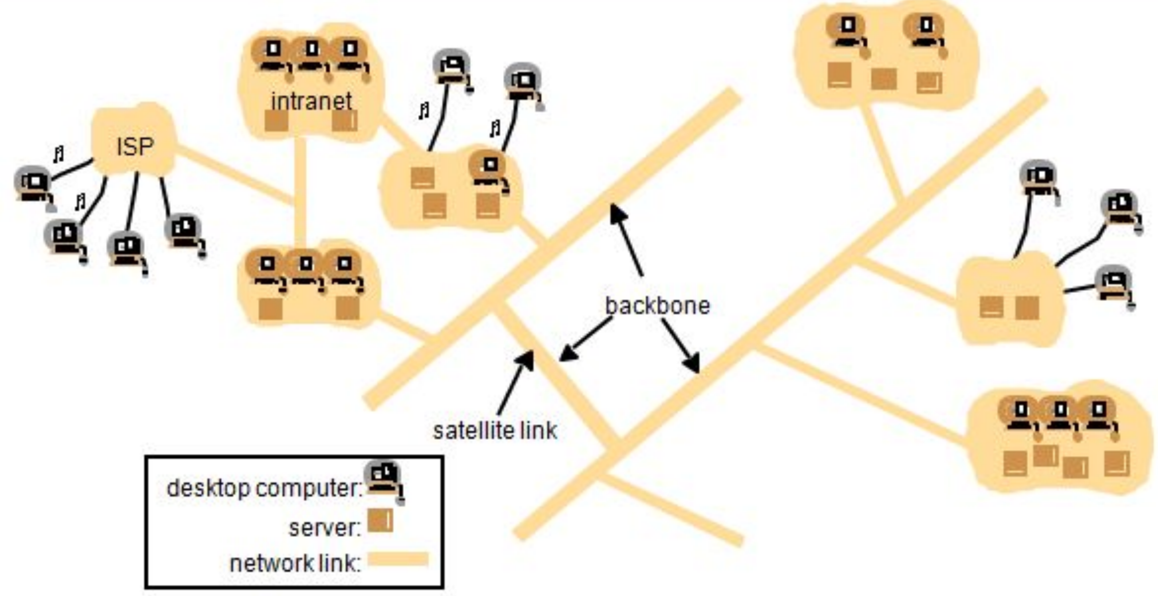
TRENDS in DS

- Distributed systems are undergoing a period of significant change
  - the emergence of **pervasive networking technology**;
  - the emergence of **ubiquitous computing** coupled with the desire to **support user mobility** in distributed systems;
  - the increasing demand for **multimedia services**;
  - the view of distributed systems as a **utility**.

PERVASIVE NETWORKING TECHNOLOGY



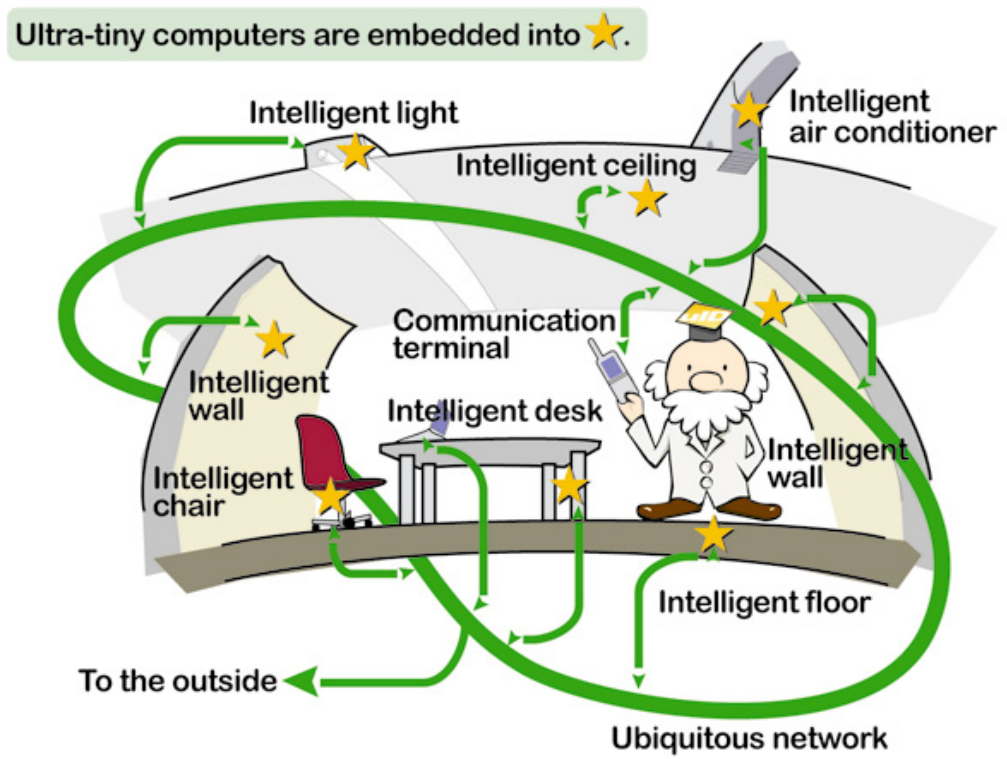


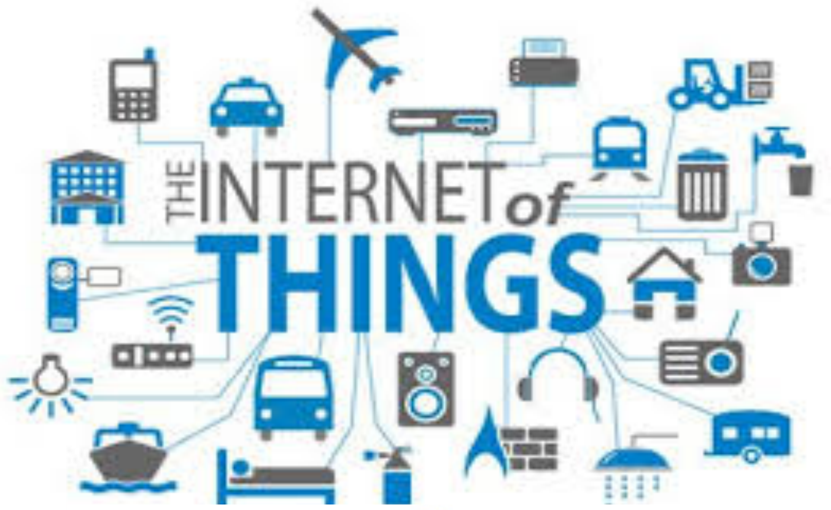


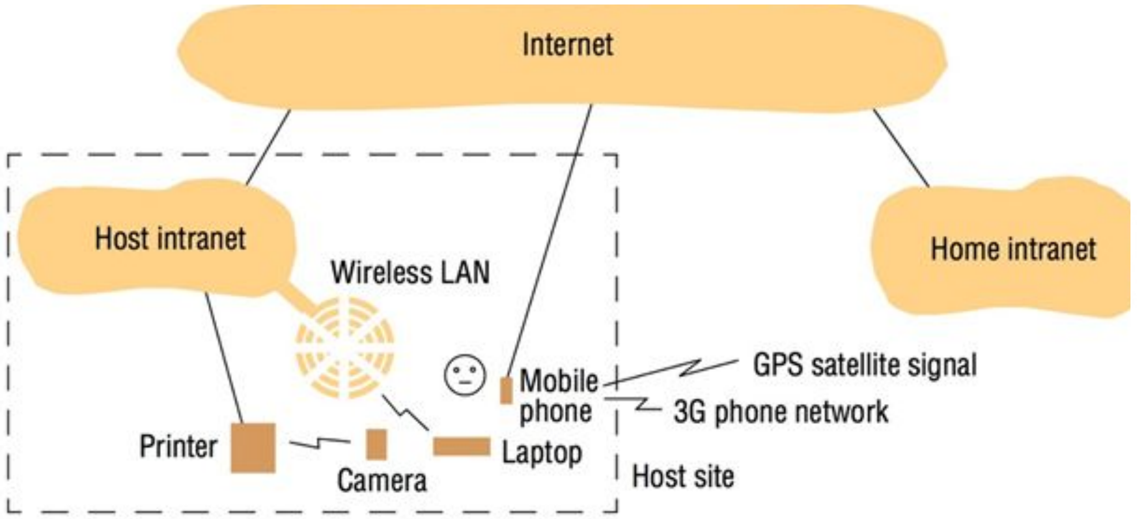
DISTRIBUTED MULTIMEDIA SERVICES

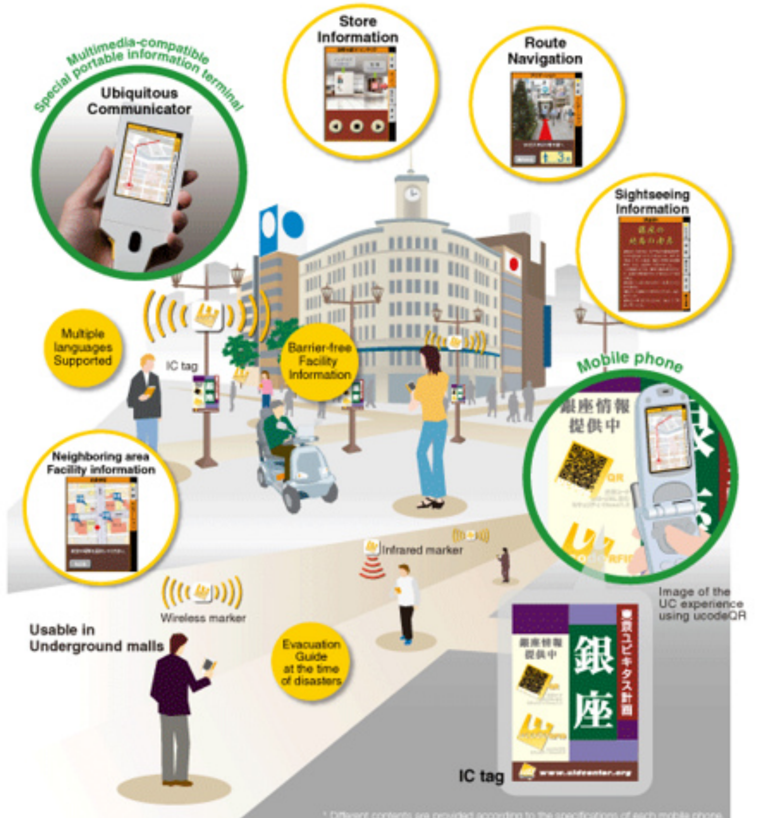
Audio
Video
3D Graphics
Multimedia
Flash Production
Presentation
Photographic

MOBILE AND UBIQUITIOUS COMPUTING

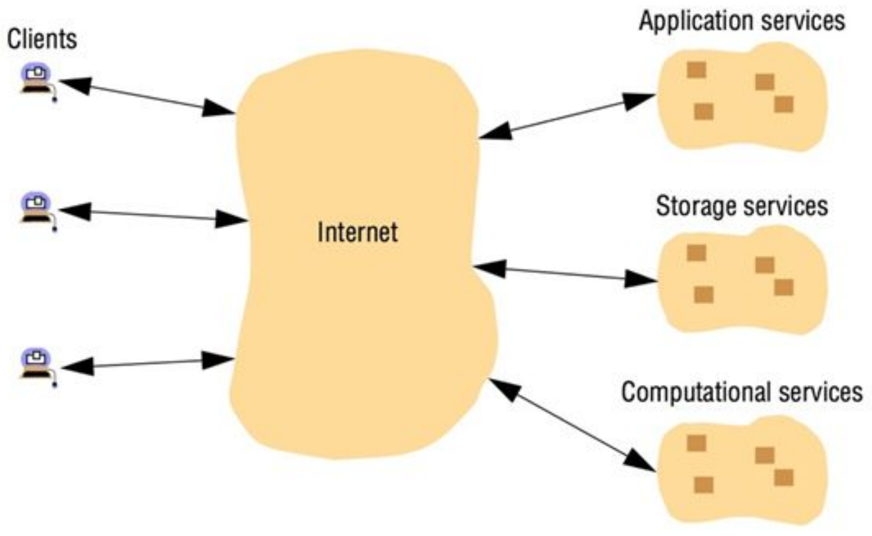


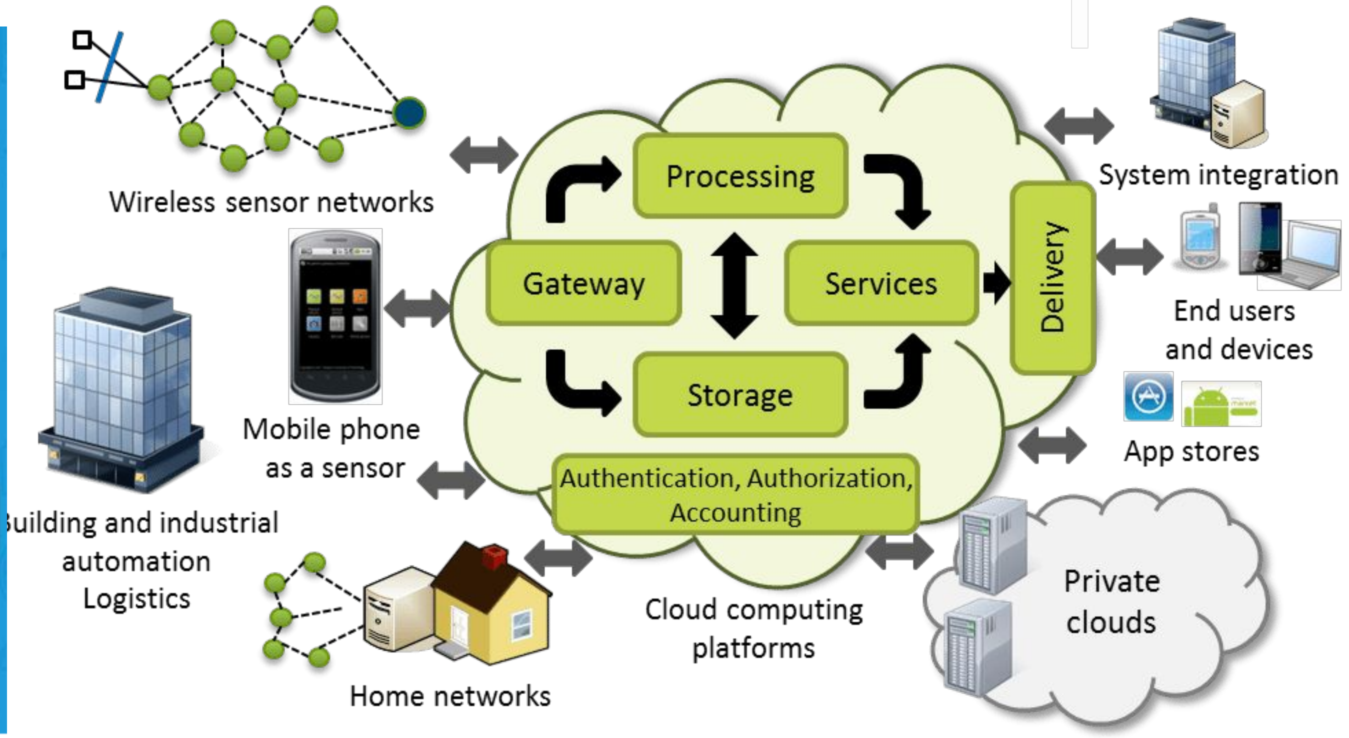




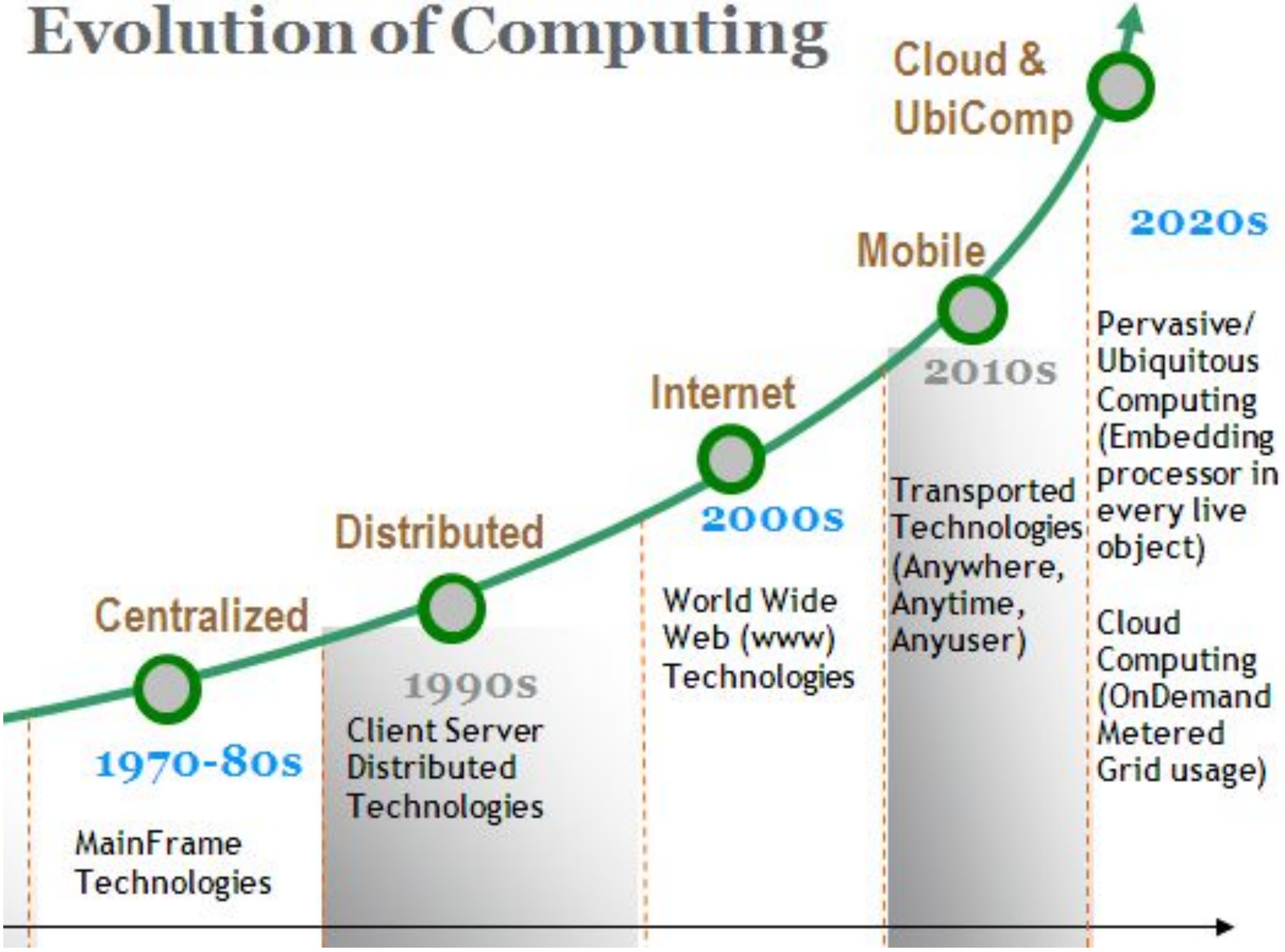

Multimedia-compatible Special portable information terminal
Ubiquitous Communicator
Store Information
Route Navigation
Sightseeing Information
Multiple languages Supported
IC tag
Barrier-free Facility Information
Mobile phone
Neighboring area Facility information
Infrared marker
Wireless marker
Usable in Underground malls
Evacuation Guide at the time of disasters
Image of the UC experience using ucodeQR
銀座情報提供中
銀座
東京ユビキタス計画
IC tag
www.uidcenter.org

DISTRIBUTED COMPUTING AS A UTILITY





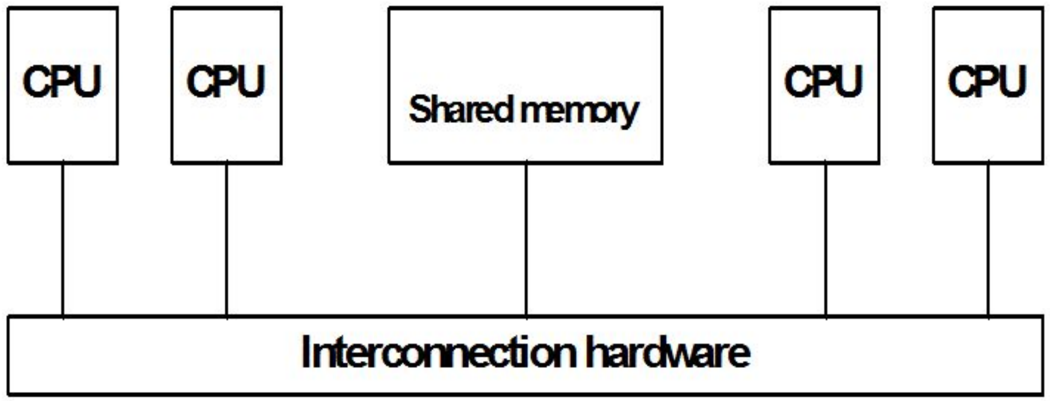
Evolution of Computing



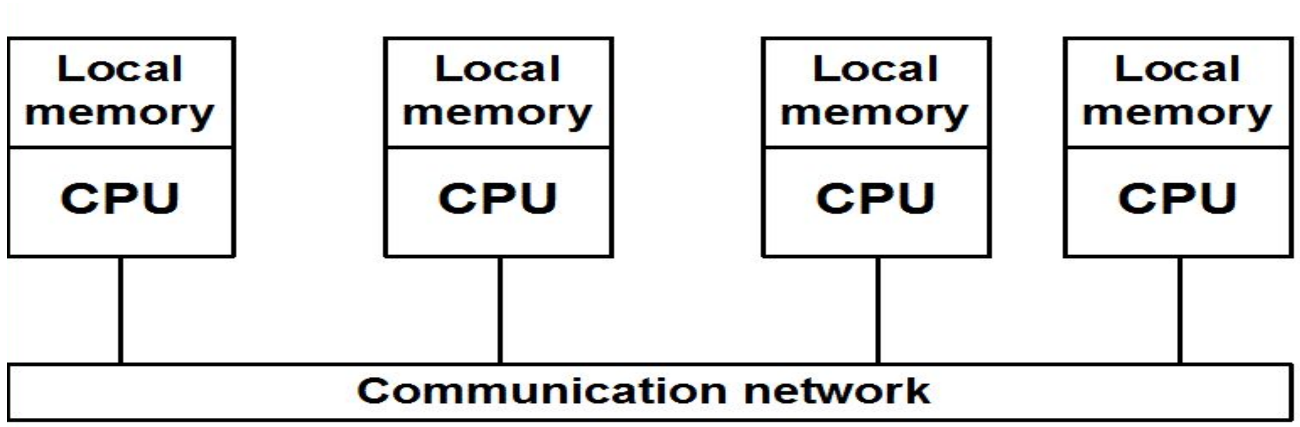
Centralized
1970-80s
MainFrame Technologies
Distributed
1990s
Client Server Distributed Technologies
Internet
2000s
World Wide Web (www) Technologies
Mobile
2010s
Transported Technologies (Anywhere, Anytime, Anyuser)
Cloud & UbiComp
2020s
Pervasive/ Ubiquitous Computing (Embedding processor in every live object)
Cloud Computing (OnDemand Metered Grid usage)

DISTRIBUTED COMPUTING SYSTEMS

- Advancements in
  - microelectronic technology-> fast inexpensive processors
  - communication technology->cost effective, highly efficient computer networks
- Interconnected multiple processors
  - Tightly coupled systems (parallel processing systems)



- Loosely coupled systems (distributed computing systems)



- Loosely coupled systems:
  - Cover wider geographical area
  - Freely expandable with almost unlimited number of processors

Parallel and Distributed Computing

- **Parallel computing:** all processors may have access to a **shared memory** to exchange information between processors

- **Distributed computing:** each processor has its own private memory (**distributed memory**). Information is exchanged by passing messages between the processors

What is a distributed System?

- **A distributed system is one in which components (hardware and software) located at networked computers communicate and coordinate their actions only by passing messages.**

- Computers may be spatially separated by any distance

- **Distributed Systems:**
  - May have a **common goal**
  - **Each computer may have its own user** with individual needs
  - The system has to **tolerate failures**
  - The **structure** of the system is **not known in advance**
  - **Each computer** has only a **limited, incomplete view** of the system

- Desire to share ‘**resources**’;
  - hardware components such as disks and printers
  - software-defined entities such as files, databases and data objects of all kinds

    It includes the stream of video frames that emerges from a digital video camera and the audio connection that a mobile phone call represents.

- Significant **characteristics of distributed systems**:

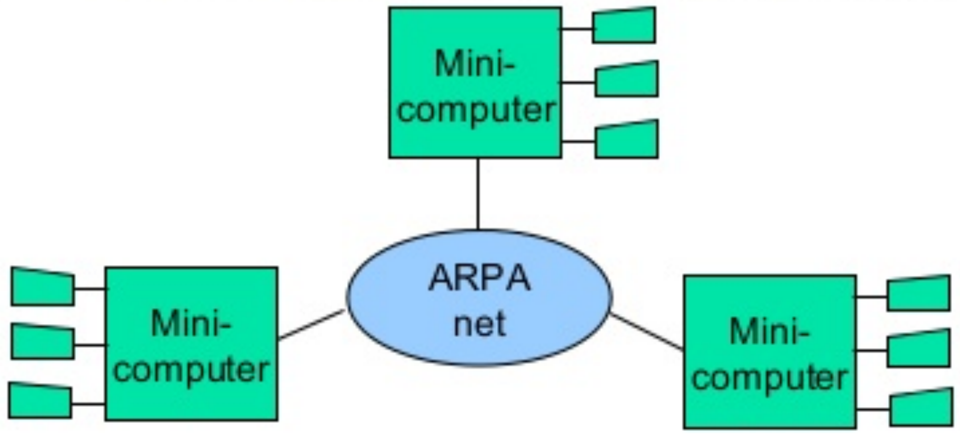
  concurrency of components

  lack of a global clock and

  independent failures of components

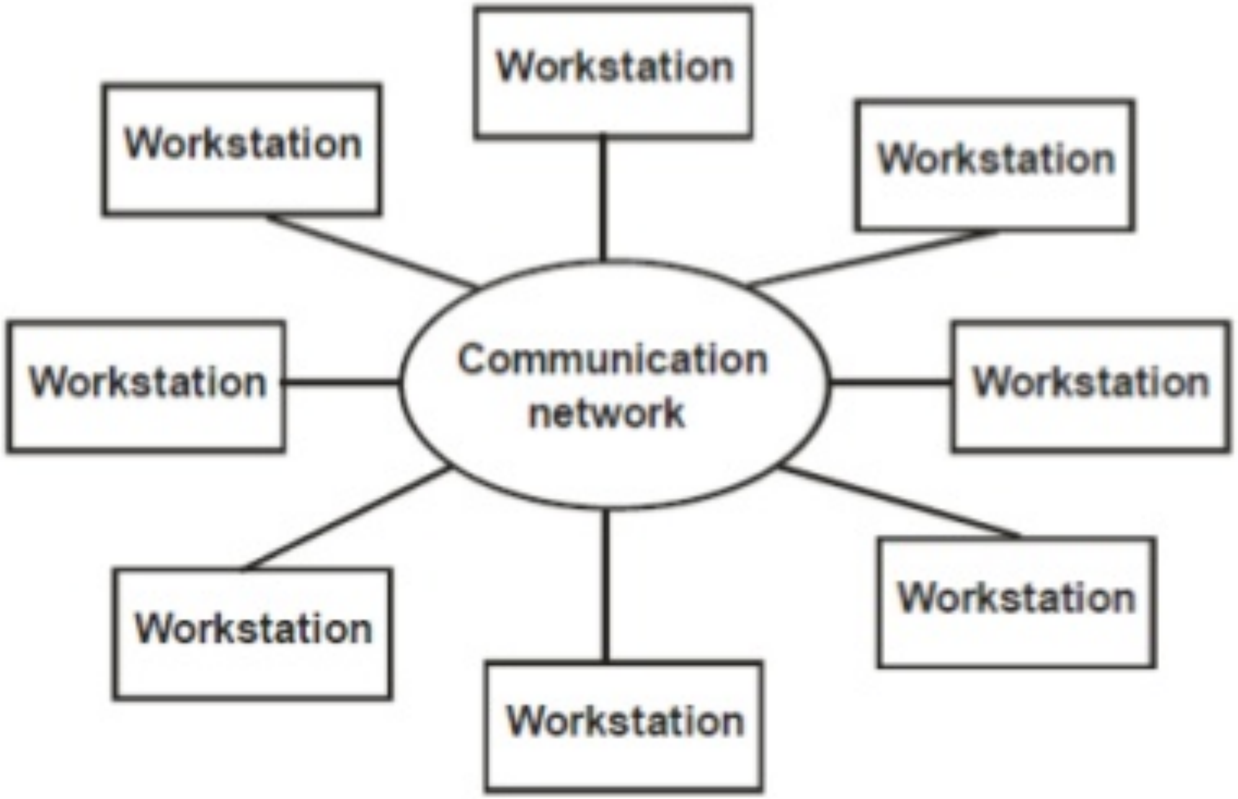
DISTRIBUTED COMPUTING SYSTEM MODELS

Minicomputer Model



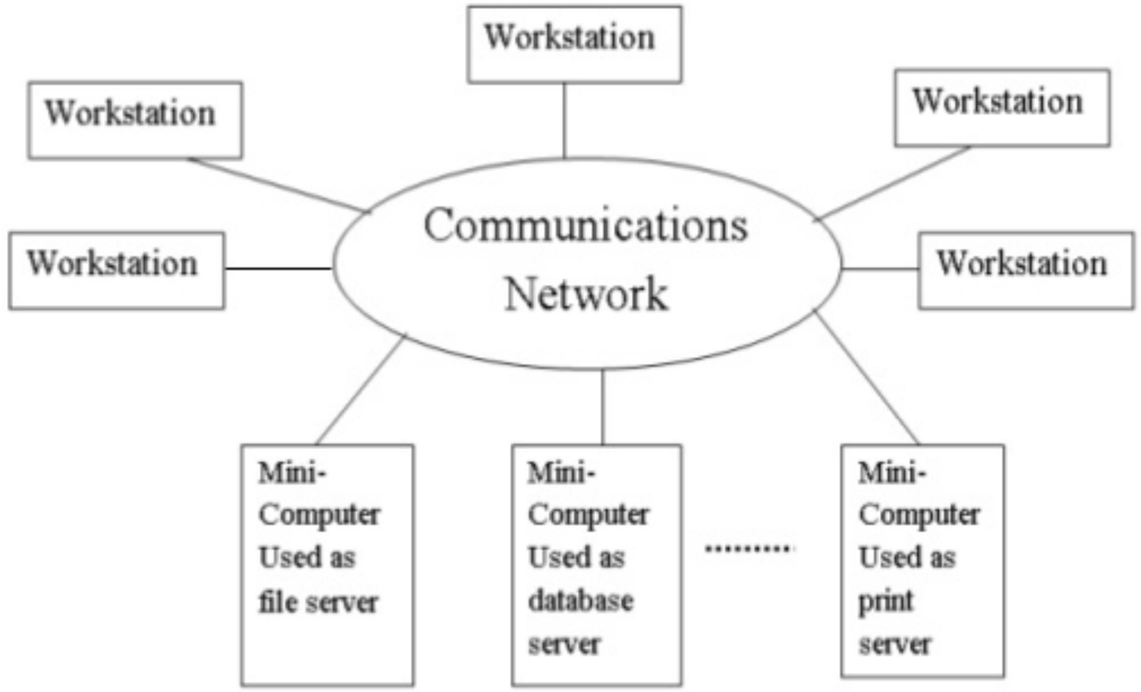
- Extension of Time sharing system
  - User must log on his/her home minicomputer.
  - Thereafter, he/she can log on a remote machine by telnet.
- Resource sharing
  - Database
  - High-performance devices

WORKSTATION MODEL



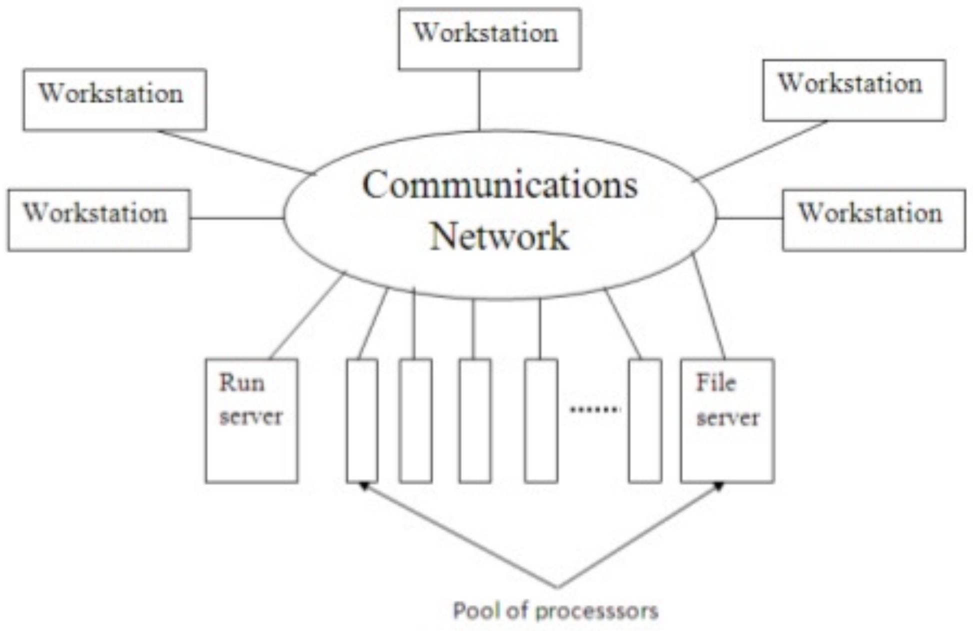
- Each workstation has its own disk serving as a single user computer.
- Home worstation and remote workstation

Workstation-Server Model (Cont. …)


Workstation
Workstation
Workstation
Workstation
Workstation
Communications Network
Mini-Computer Used as file server
Mini-Computer Used as database server
……….
Mini-Computer Used as print server

- Cheaper to use few minicomputers
- Diskless WS easy to maintain, backup and h/w maintenance of few disks, installing new releases
- Users can use any WS
- Client – server model: Request – response protocol, no process migration
- No remote process in WS, hence guaranteed response time

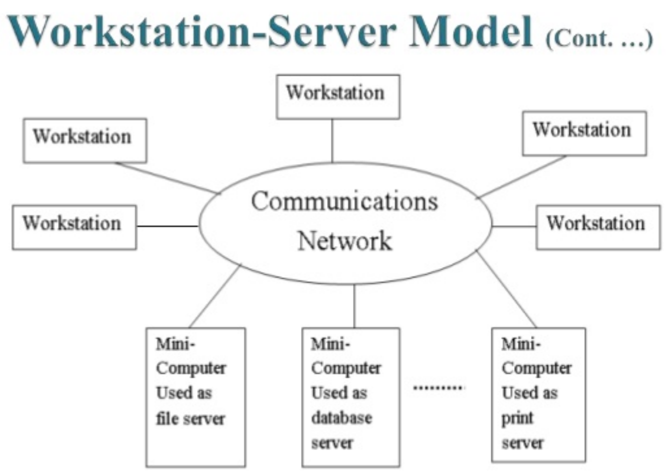
Processor Pool Model (Cont. …)

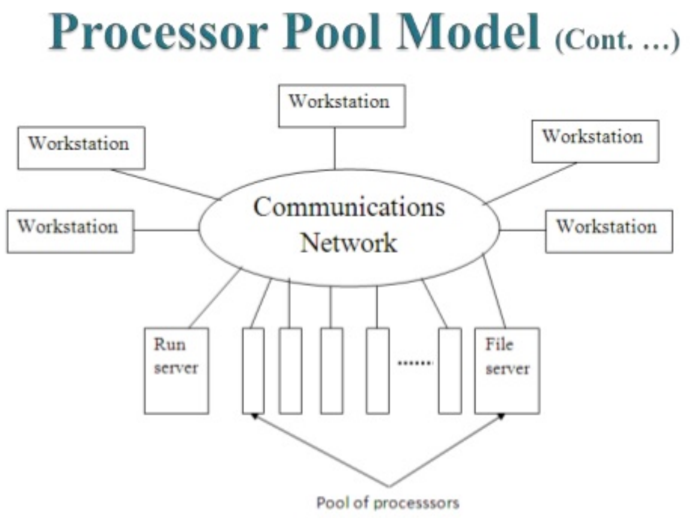


- Processors pooled together and shared, terminals not directly attached
- Terminals- diskless WS or graphics terminals
- Run server- allocates processors to users when job is submitted
- No home machine, log onto the whole system
- Better utilization of available processing power(no idle WS)
- Greater flexibility, Easily expanded
- Not suitable for high performance interactive applications.Communcation delay b/w computer and terminal

HYBRID MODEL

- WS Server + processor pool


Workstation-Server Model (Cont. ...)
Workstation
Workstation
Workstation
Workstation
Workstation
Communications Network
Mini-Computer Used as file server
Mini-Computer Used as database server
....
Mini-Computer Used as print server



- WS Server plus additional pool of processors.

CHALLENGES

1. Heterogeneity
2. Openness
3. Security
4. Scalability
5. Failure Handling
6. Concurrency
7. Transparency
8. Quality of Service

Heterogeneity

- Users access services and run applications over a heterogeneous collection of computers and networks

- Heterogeneity (that is, variety and difference) applies to all of the following:
  - networks
  - computer hardware
  - operating systems
  - programming languages
  - implementations by different developers

- **Middleware**:
  - provides a **programming abstraction**
  - **masks the heterogeneity**
  - Provides uniform computational model for use by the programmers of servers and distributed applications

    eg. Common Object Request Broker (CORBA), Java Remote Method Invocation (RMI)

- **Mobile code**
  - program code that can be transferred from one computer to another
  - runs at the destination

    Eg. Java applets
  - The **virtual machine** approach provides a way of making code executable on a variety of host computers

Openness

- **System**: whether it can be extended and re-implemented

- **Distributed System**: the degree to which
  - new resource-sharing services can be added
  - Services can be made available for use by a variety of client programs

- **Publish** (make available to software developers):
  - The **specification and documentation of the key software interfaces**

- Designers need to tackle the **complexity** of distributed systems:
  - consisting of **many components**
  - engineered by **different people**

- Open distributed systems are based on
  - the provision of a **uniform communication mechanism**
  - **published interfaces** for access to shared resources

- **Each component must conform to** the published **standard**

Security

- Security for information resources has three components:
  - **confidentiality**
  - **integrity**
  - **Availability**
- **Firewall:**
  - can be used to form a barrier around an intranet
  - this does not deal with
    - ensuring the appropriate use of resources by users within an intranet
    - the appropriate use of resources in the Internet that are not protected by firewalls

- Encryption:
  - Send sensitive data over a network in a secure manner.

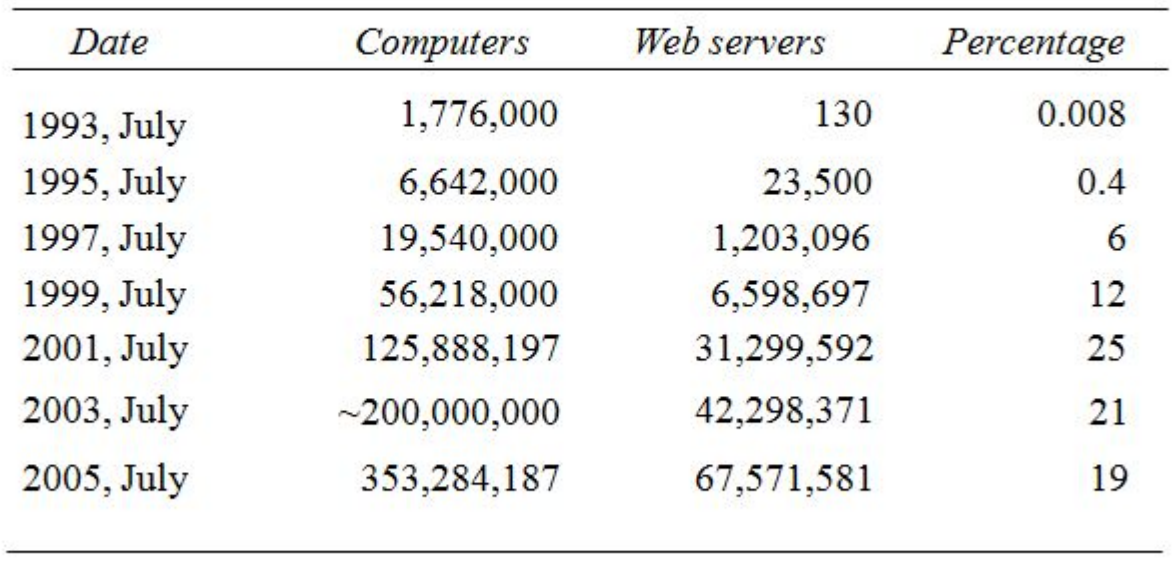
    Eg patient info, ecommerce& banking

  - Ensure the identity of the user.
- Not fully resolved:
  - Denial of service attacks
  - Security of mobile code

Scalability

- DS operate effectively and efficiently at many different scales
- Scalability: ability to remain effective when there is a significant increase in the number of resources and the number of users.

GROWTH OF THE INTERNET



Date	*Computers*	*Web servers*	*Percentage*
1993, July	1,776,000	130	0.008
1995, July	6,642,000	23,500	0.4
1997, July	19,540,000	1,203,096	6
1999, July	56,218,000	6,598,697	12
2001, July	125,888,197	31,299,592	25
2003, July	~200,000,000	42,298,371	21
2005, July	353,284,187	67,571,581	19

The design of scalable distributed systems presents the following challenges:

- **Controlling the cost of physical resources**
  - Eg. More users -> more servers -> should be proportional O(n)

- **Controlling the performance loss**
  - Eg. Management of DNS ; hierarchical algorithms scale better but performance loss exists
  - Max acceptable performance loss -> O(log n) ->Access time for hierarchical data structures

- **Preventing software resources running out**
  - Eg. IP Addresses IPv4(32 bits) -> IPv6 (128 bits)

- **Avoiding performance bottlenecks**
  - algorithms should be decentralized
  - Eg. DNS

Failure Handling

- Failures in a distributed system are partial

1. **Detecting failures**
   - Detectable
     - eg. Corrupted data using checksum

   - Difficult (or even impossible) to detect some failures
     - eg. remote crashed server

   - Challenge:
     Managing in the presence of **undetected but suspected failures**

2. **Masking failures:**
   - Failures detected can be hidden or made less severe
     - Eg. Retransmission of msgs, multiple copies of file data, reduce severity by dropping corrupted msg.
   - Does not work in worst cases
     - Eg. both copies of file corrupted, retransmission unsuccesful etc.

3. **Tolerating failures:**
   - Clients can be designed to tolerate failures
   - Involves the users tolerating failures as well
     - eg. web browser cannot contact a web server

5. **Recovery from failures:**
   - the state of permanent data should be recoverable
     - 'rolled back' after a server has crashed

5. **Redundancy:**
   1. At least two different routes between any two routers in the Internet
   2. In the Domain Name System- replicated name servers
   3. Replicated Database

- **CHALLENGE**: The design of effective techniques for **keeping replicas of rapidly changing data up-to-date** without excessive loss of performance.

- DS - provide a high degree of availability in the face of hardware faults

- **Availability** of a system- measure of the proportion of time that it is available for use

Concurrency

- **Shared resource** must **operate correctly** in a **concurrent** environment

  Eg Bids for an auction, online sale of limited items

- Operations must be synchronized such that its data remains consistent

- Semaphores

Transparency

- **Concealment** of the **separation of components** in a distributed system
- **The system is perceived as a whole** rather than as a collection of independent components

Types:

- **Access transparency**
- **Location transparency**
- **Concurrency transparency**
- **Replication transparency**
- **Failure transparency**
- **Mobility transparency**
- **Performance transparency**
- **Scaling transparency**

- **Network transparency** *: i.e.* access and location transparency;

  Eg.
  - Graphical user interface with folders

    : Access Transparency

  - URL’s

    : Location Transparency

  - Email in the internet

    : network transparent; failure transparent

Quality of Service

Non-functional properties of systems that affect the quality of the service experienced

- **Reliability**
- **Security**
- **Performance**
  - Time critical data streams like movies service
- **Adaptability**
  - To meet changing system configurations
  - To resource availability
- Depends upon the **availability** of the necessary **computing and network resources** at the appropriate times

Distributed system advantages

- Inherently distributed applications
- Information sharing
- Resource sharing
- Better price performance ratio
- Shorter response time , higher throughput
- Higher reliability
- Extensibility and incremental growth
- Better flexibility

Distributed Operating Systems

System Image:

Automatically and dynamically allocates jobs to different machines in the system for processing.