OUTLINE

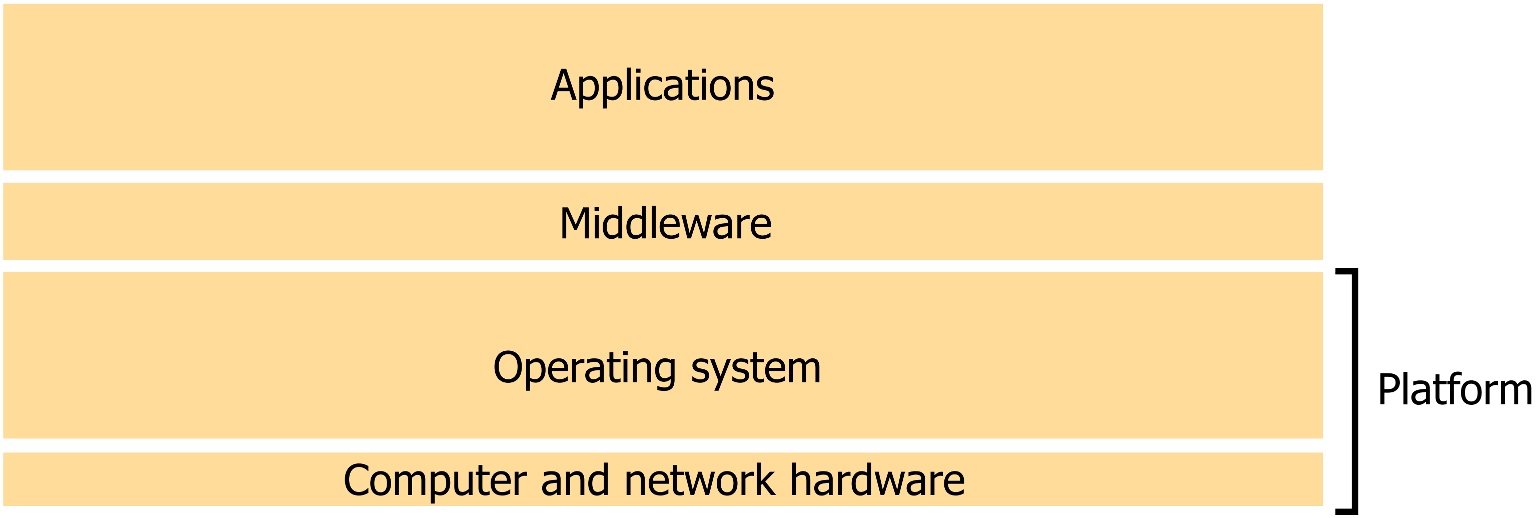
- Distributed System Models.
- Distributed System Software Layers
- Distributed System Architectures.
- Client-Server Model Variations.
- Distributed Processes Interfaces and Objects.
- Distributed Architectures Design Requirements.
- Fundamental Models:
  - Interaction Model.
  - Failure Model.
  - Security Model.

Distributed System Models

- Architectural models: (*as client-server and peer process models*)
  - Define the way in which the components of systems are:
    - Interact with one another, and
    - Mapped onto an underlying network of computers.
  - Describe the layered structure of distributed system software.
- Fundamental models:
  - Concerned with properties that are common in all of the architectural models.
  - Addressed by three models:
    - The interaction model: deals with the difficulty of setting time limits.
    - The failure model: attempts to give a specification of the exhibited faults by processes and communication channels.
    - The security model: discusses possible threats to processes and communication channels.

Software Layers

- In the layered view of a system each layer offers its services to the level above and builds its own service on the services of the layer below.
- Software architecture is the structuring of software in terms of layers (modules) or services that can be requested locally or remotely.



Applications

Middleware

Operating system

Computer and network hardware

Platform

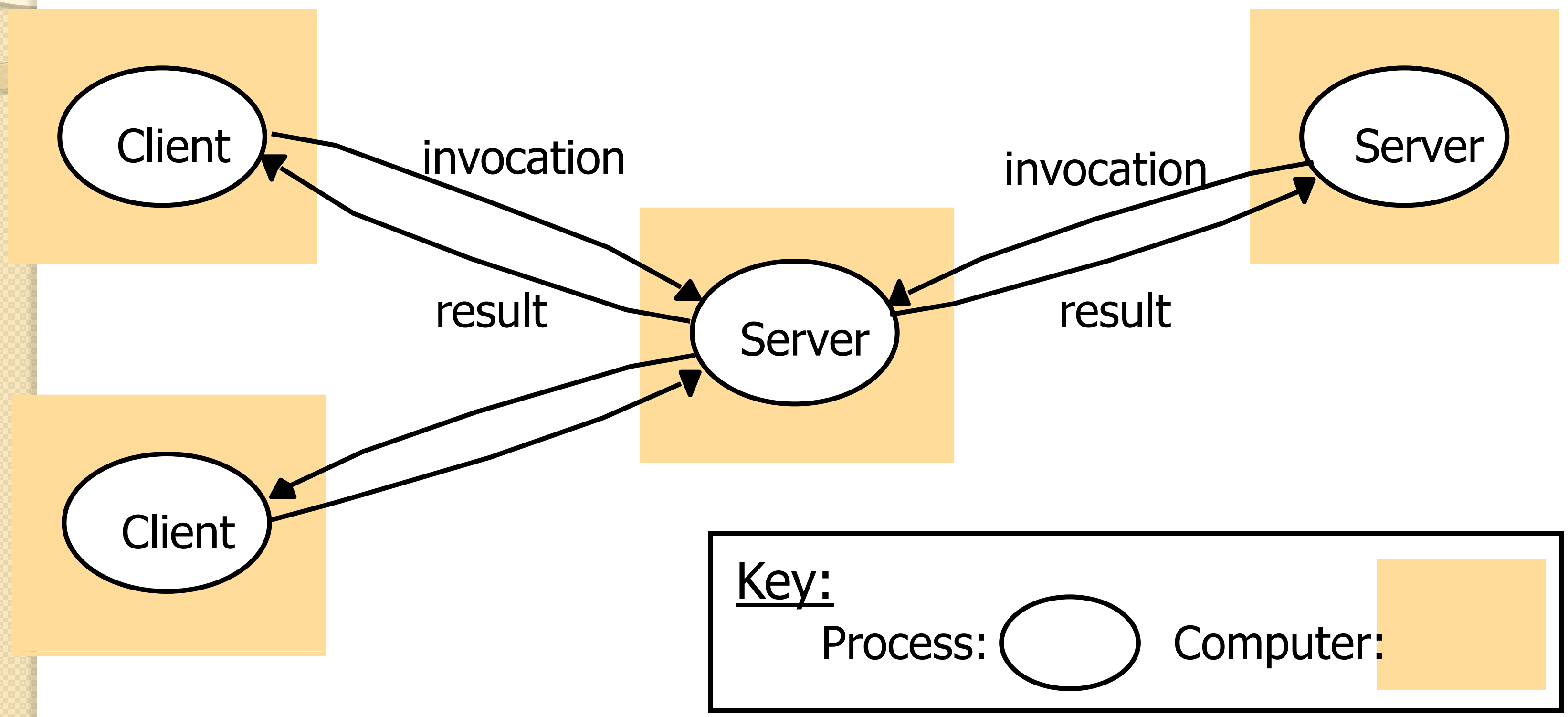
Software Layers

- Platform:
  - Lowest-level layers that provide services to other higher layers.
  - bring a system's programming interface for communication and coordination between processes .
  - Examples:
    - Pentium processor / Windows NT
    - SPARC processor / Solaris
- Middleware:
  - Layer of software to mask heterogeneity and provide a unified distributed programming interface to application programmers.
  - Provide services, *infrastructure services*, for use by application programs.
  - Examples:
    - Object Management Group's Common Object Request Broker Architecture (CORBA).
    - Java Remote Object Invocation (RMI).
    - Microsoft's Distributed Common Object Model (DCOM).
  - Limitation: require application level involvement in some tasks.

System Architectures

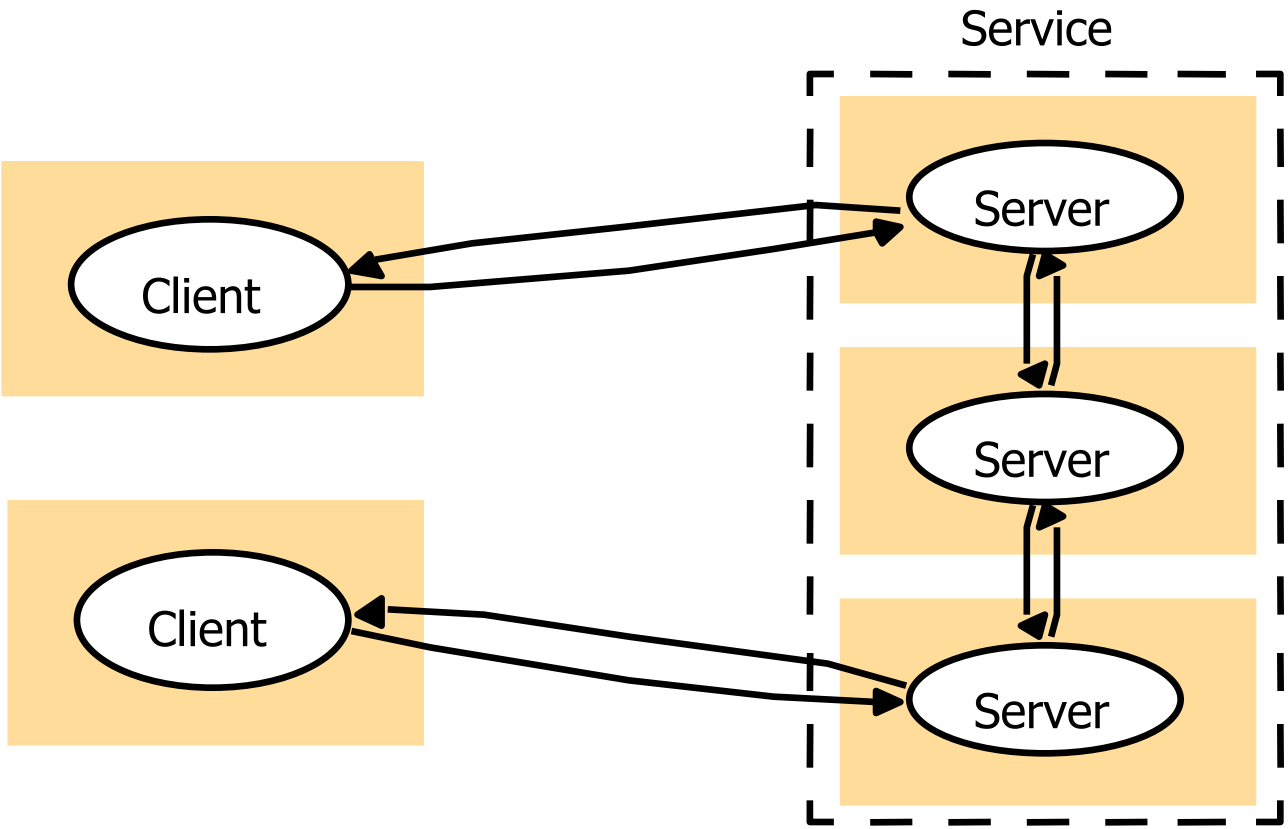
- The architecture include:
  - The division of responsibilities between system components.
  - The placement of the components on computers in the network.
- **<u>Client-server model:</u>**
  - Most important and most widely distributed system architecture.
  - Client and server roles are assigned and changeable.
    - Servers may in turn be clients of other servers.
  - Services may be implemented as several interacting processes in different host computers to provide a service to client processes:
    - Servers partition the set of objects on which the service is based and distribute them among themselves (e.g. Web data and web servers)

System Architectures



Clients invoke individual servers

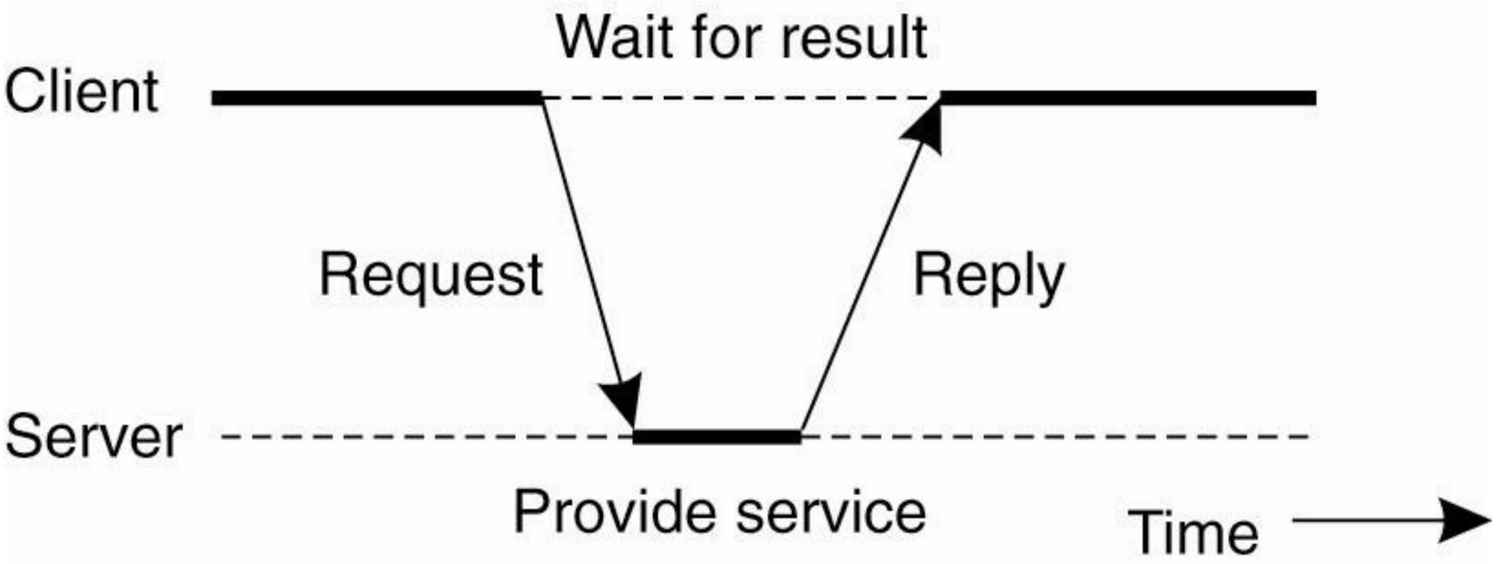
System Architectures



A service provided by multiple servers

Centralized Architectures

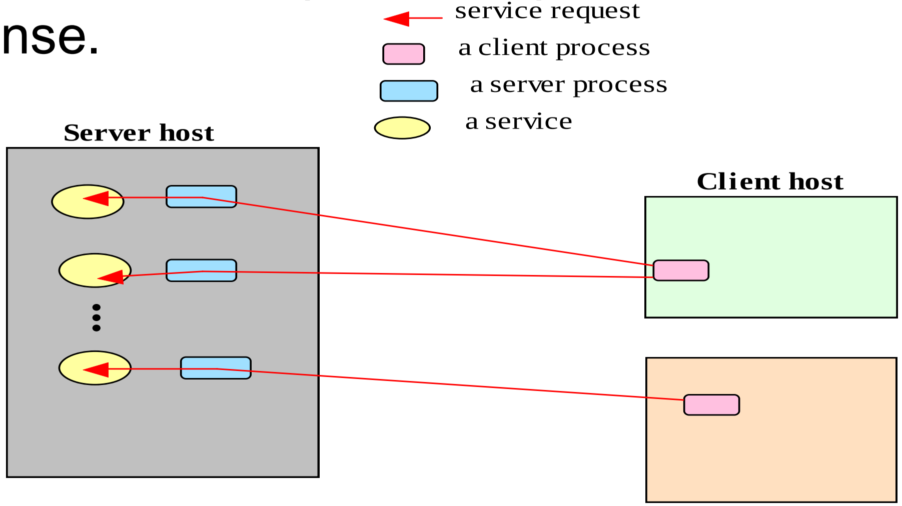
- Figure 2-3. General interaction between a client and a server.



The Client-Server Paradigm(نموذج)

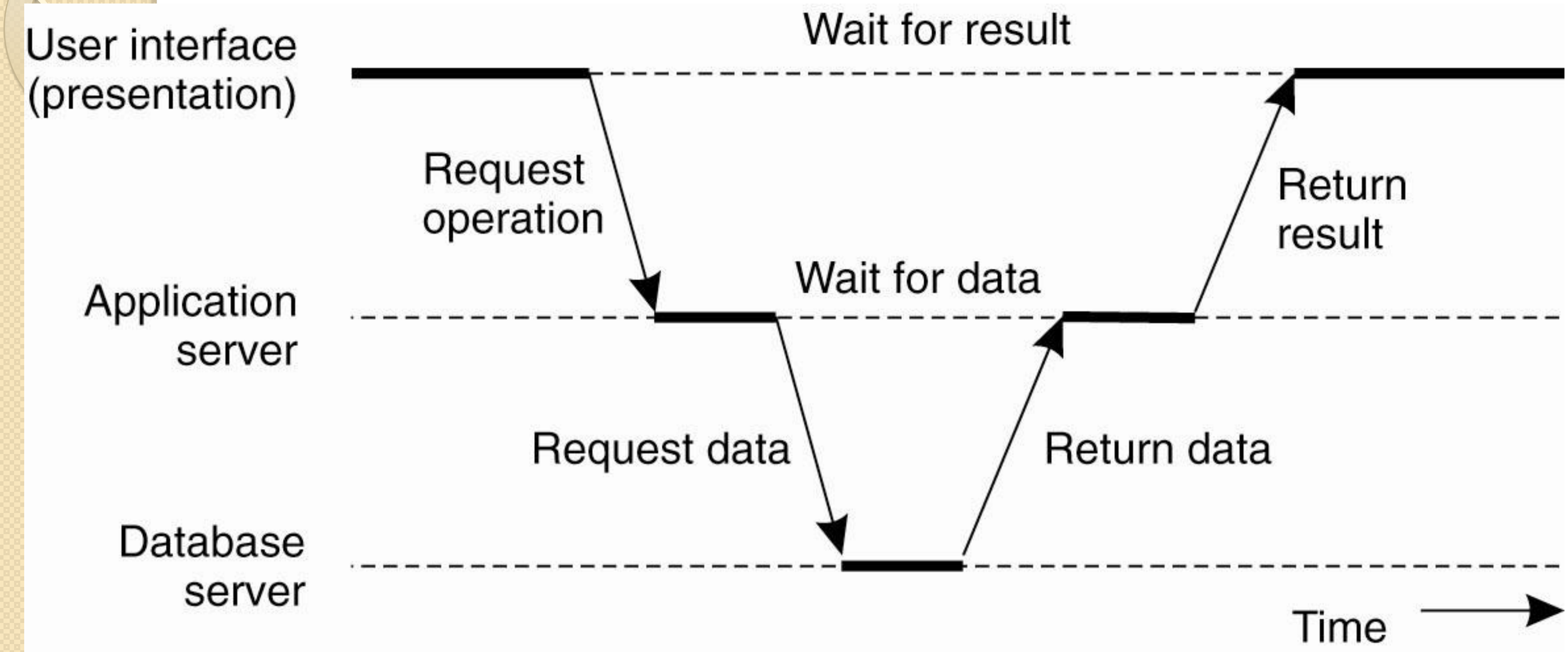
Perhaps the best known paradigm for network applications, the client-server[2] model assigns asymmetric roles to two collaborating processes.

One process, the server, plays the role of a service provider which waits passively for the arrival of requests. The other, the client, issues specific requests to the server and awaits its response.



The Client-Server Paradigm, conceptual

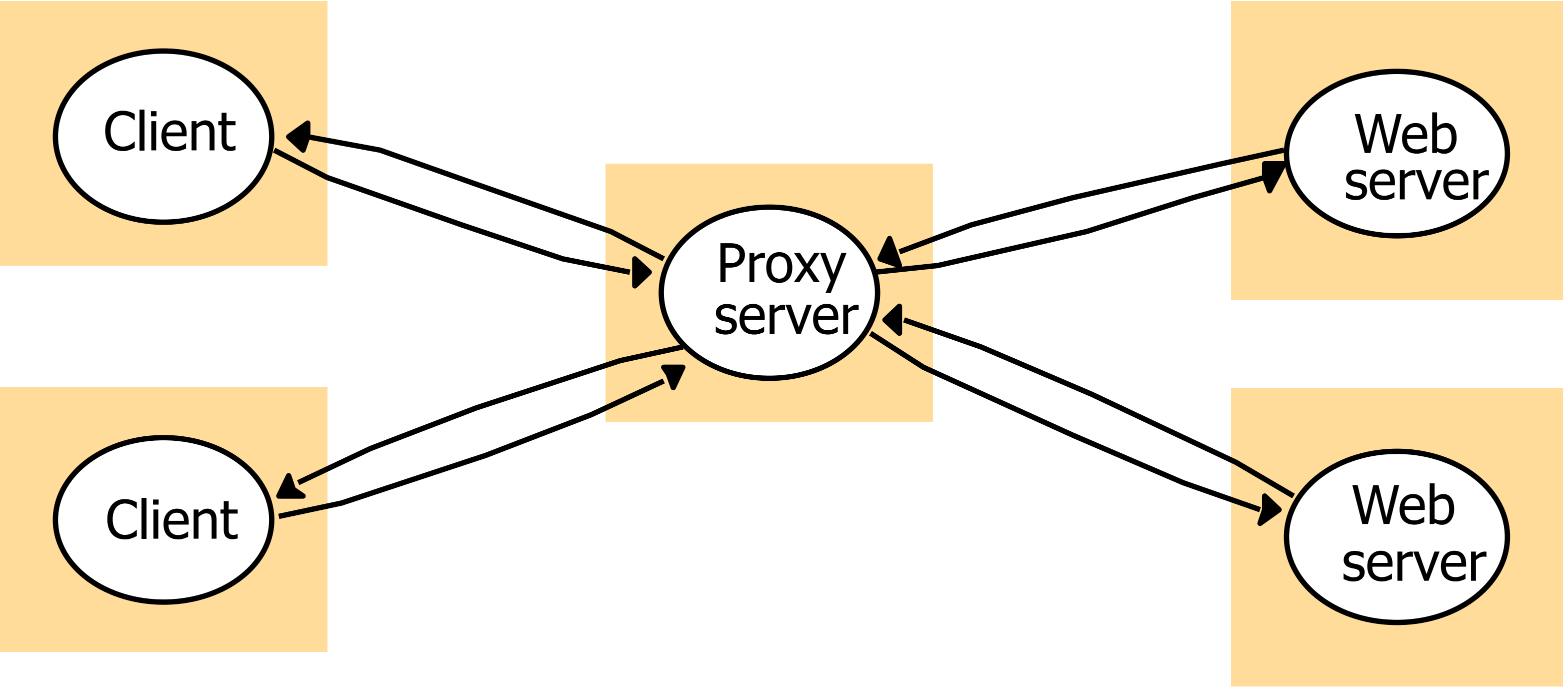
Multitiered Architectures (3)



System Architectures

- **Caches and proxy servers:**
  - Cache:
    - A store of recently used data objects that is closer to the client process than those remote objects.
    - When an object is needed by a client process the caching service checks the cache and supplies the object from there in case of an up-to-date copy is available.
  - Proxy server:
    - Provides a shared cache of web resources for client machines at a site or across several sites.
    - Increase availability and performance of a service by reducing load on the WAN and web servers.
    - May be used to access remote web servers through a firewall.

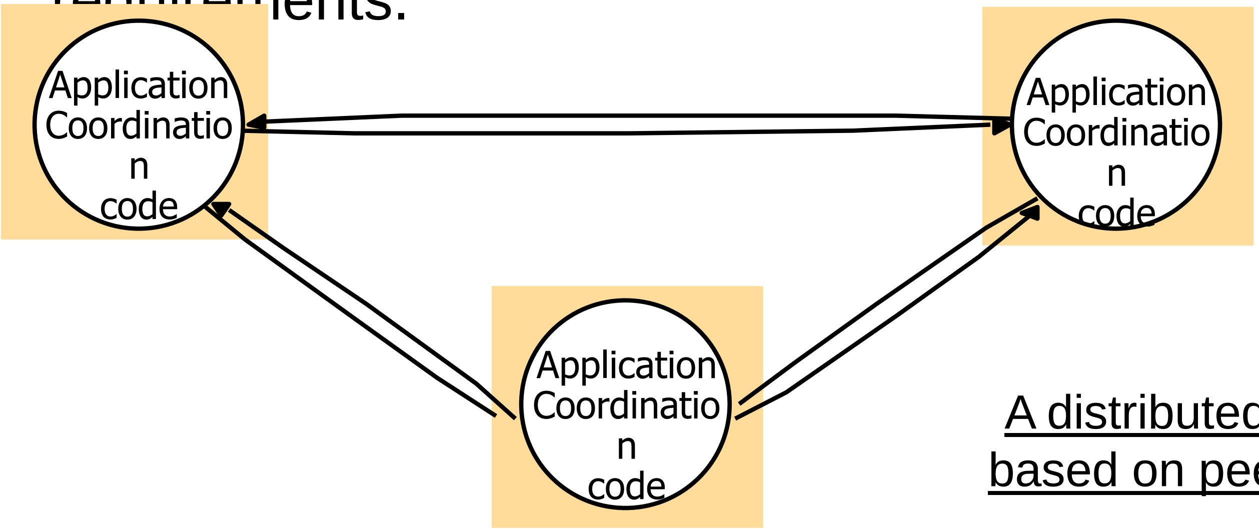
System Architectures



Web proxy server

System Architectures

- **Peer processes:**
  - All processes play similar roles without destination as a client or a server.
  - Interacting cooperatively to perform a distributed activity.
  - Communications pattern will depend on application requirements.


Application Coordinatio n code
Application Coordinatio n code
Application Coordinatio n code

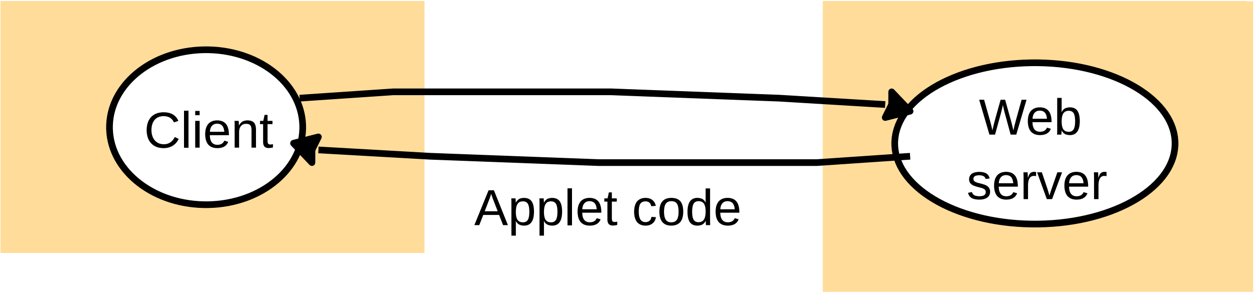
A distributed application based on peer processes

Client-server Model Variations (Mobile Code)

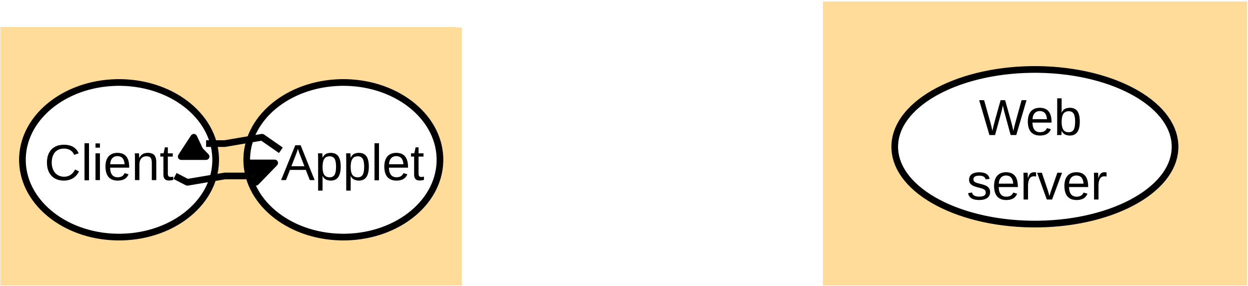
- Example: Java applets
  - The user running a browser selects a link to an applets whose code is stored on a web server.
  - The code is downloaded to the browser and runs there.
- Advantage:
  - Good interactive response since.
  - Does not suffer from the delays or variability of bandwidth associated with network communication.
- Disadvantage:
  - Security threat to the local resources in the destination computer.

Client-server Model Variations (Mobile Code)

a) client request results in the downloading of applet code



b) client interacts with the applet



Web applets

Client-server Model Variations (Mobile Agents)

- A running program (including both code and data) that travels from one computer to another in a network carrying out a task on someone's behalf.
- Can make many invocations to local resources at each visited site.
- Visited sites must decide which local resources are allowed to use based on the identity of the user owning the agent.
- Advantage: Reduce communication cost and time by replacing remote invocation with local ones.
- Disadvantages:
  - Limited applicability.
  - Security threat of the visited sites resources.

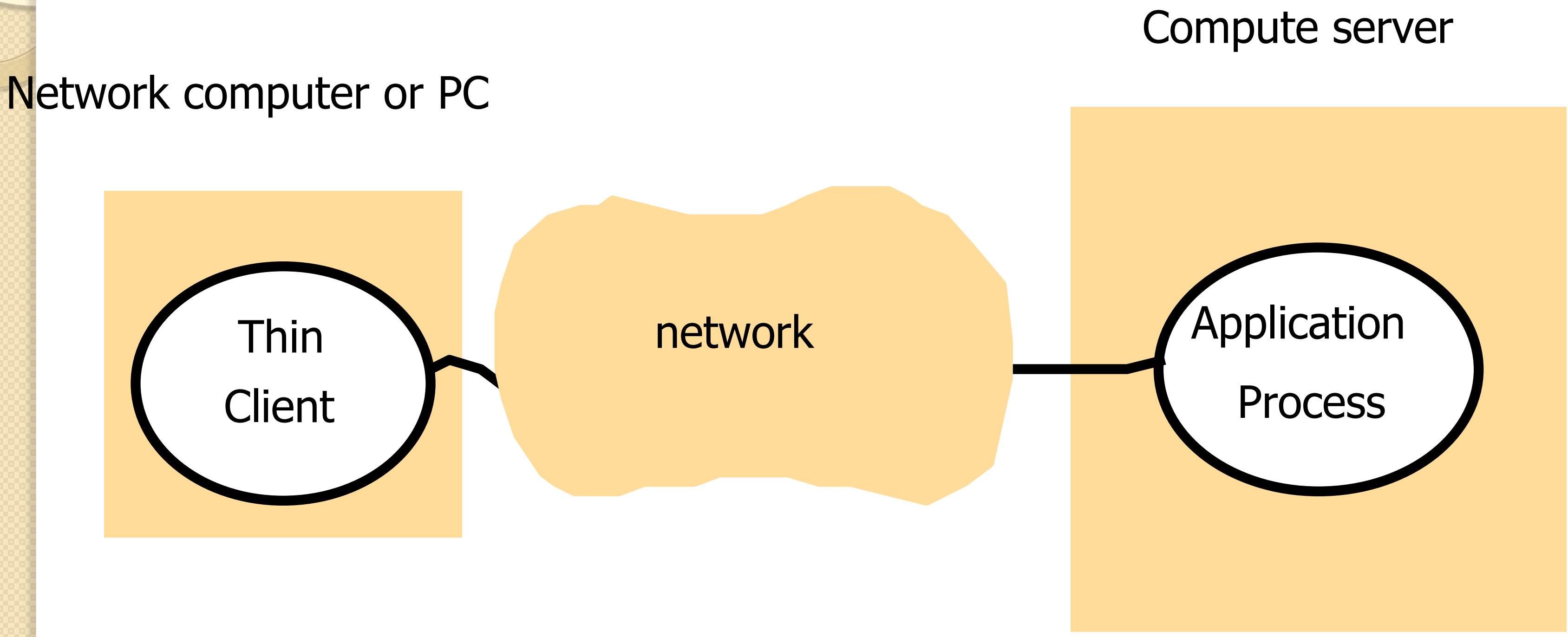
Client-server Model Variations (Network Computers)

- Downloads its operating system and any applications needed by the user from a remote file server.
- Applications run locally but files are managed by a remote file server.
- Users can migrate from one network computer to another.
- Its processor and memory capacities can be restricted to reduce its cost.
- Its disk (if included) holds only a minimum of software and use the reminder space as cache storage to hold copies of the most recently software and data files loaded from servers.

Client-server Model Variations (Thin Clients)

- Software layer that supports a window-based user interface on a local computer while executing application programs on a remote computer.
- Same as the network computer scheme but instead of downloading the applications code into the user's computer, it runs them on a server machine, *compute server*.
- Compute server is a powerful computer that has the capacity to run large numbers of applications simultaneously.
- Disadvantage: Increasing of the delays in highly interactive graphical applications

Client-server Model Variations (Thin Clients)



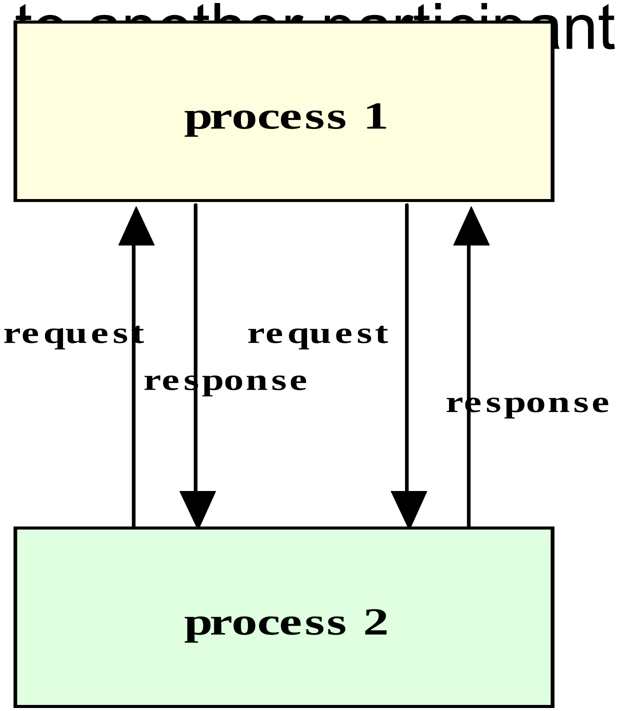
Thin clients and compute servers

The Peer-to-Peer System Architecture http://www.peer-to-peerwg.org/whatis/index.html

- In system architecture and networks, peer-to-peer is an architecture where computer resources and services are direct exchanged between computer systems.
- These resources and services include the exchange of information, processing cycles, cache storage, and disk storage for files..
- In such an architecture, computers that have traditionally been used solely as clients communicate directly among themselves and can act as both clients and servers, assuming whatever role is most efficient for the network.

The Peer-to-Peer Distributed Computing Paradigm

In the peer-to-peer paradigm, the participating processes play equal roles, with equivalent capabilities and responsibilities (hence the term “peer”). Each participant may issue a request to another participant and receive a response.



The future of peer-to-peer

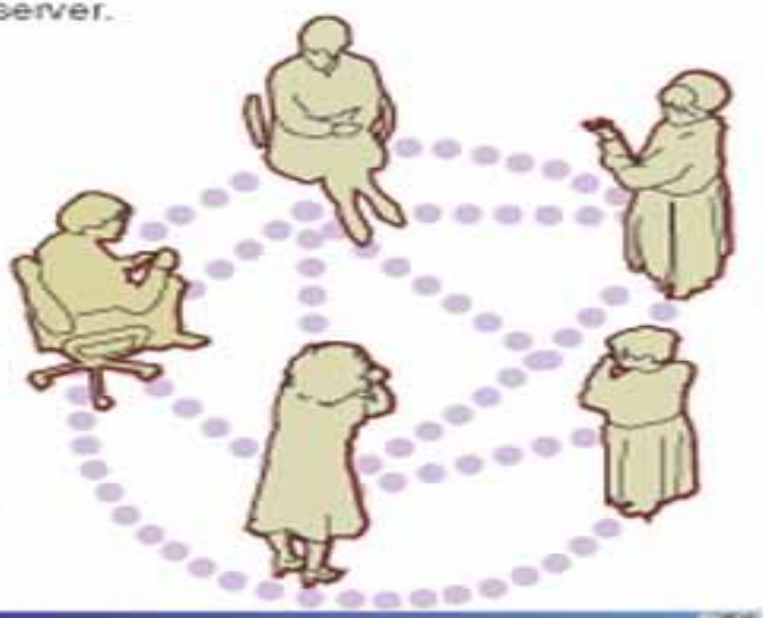
The file-swapping technology popularized by Napster, known as peer-to-peer networking, is about to change how people and corporations use the Internet. Instead of relying on central servers to process and relay information, new applications being developed will allow users to turn any computing device into a server.

A virtual meeting room

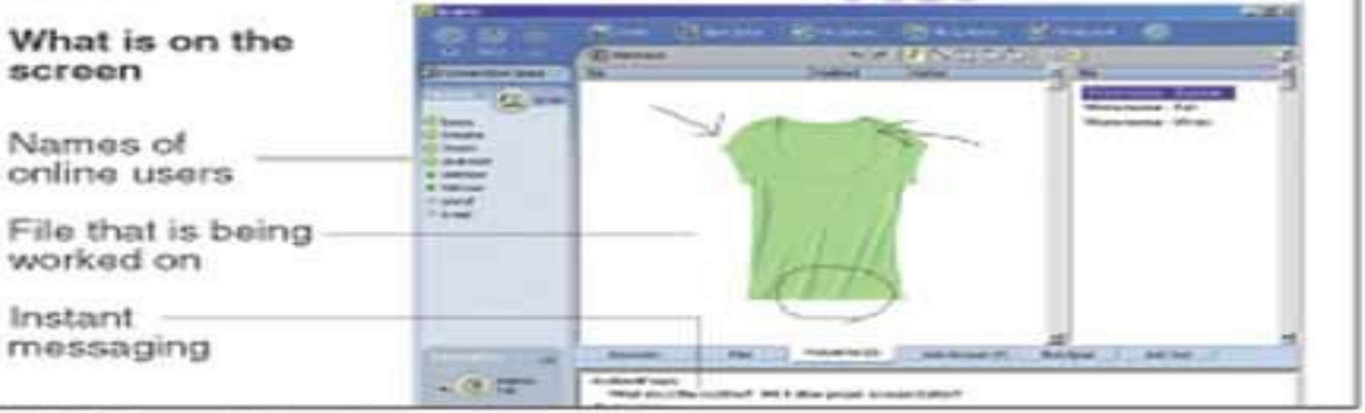
Users logon to the Internet using a program that looks like an online chat room.

A file is placed into a "shared space" within the virtual meeting room, which allows users to work on data files at the same time.

Users work in real time and can instant message each other. In the future, this might be done through devices such handhelds and cell phones.



What is on the screen



SOURCES: Groove Networks; compiled from AP wire reports

AP

Peer-to-Peer distributed computing

The peer-to-peer paradigm can be implemented with facilities using any tool that provide message-passing, or with a higher-level tool such as one that supports the point-to-point model of the Message System paradigm.

For web applications, the ***web agent*** is a protocol promoted by the XNSORG (the XNS Public Trust Organization) for peer-to-peer interprocess communication

“Project JXTA is a set of open, generalized peer-to-peer protocols that allow any connected device (cell phone, to PDA, PC to server) on the network to communicate and collaborate. JXTA is short for Juxtapose, as in side by side. It is a recognition that peer to peer is juxtapose to client server or Web based computing -- what is considered today's traditional computing model. “

ASSIGNMENT

- Q: Differentiate between thin and thick client.