CGS 3763: Operating System Concepts
Spring 2006

Client-Server Computing – Part 1

Instructor :	Mark Llewellyn
	markl@cs.ucf.edu
	CSB 242, 823-2790
	http://www.cs.ucf.edu/courses/cgs3763/spr2006

School of Electrical Engineering and Computer Science
University of Central Florida

Client-Server Computing

- The concept of client-server computing and related concepts, have become increasingly important in information technology systems.
- As with many other new waves in the computer field, client-server computing comes with its own set of jargon.

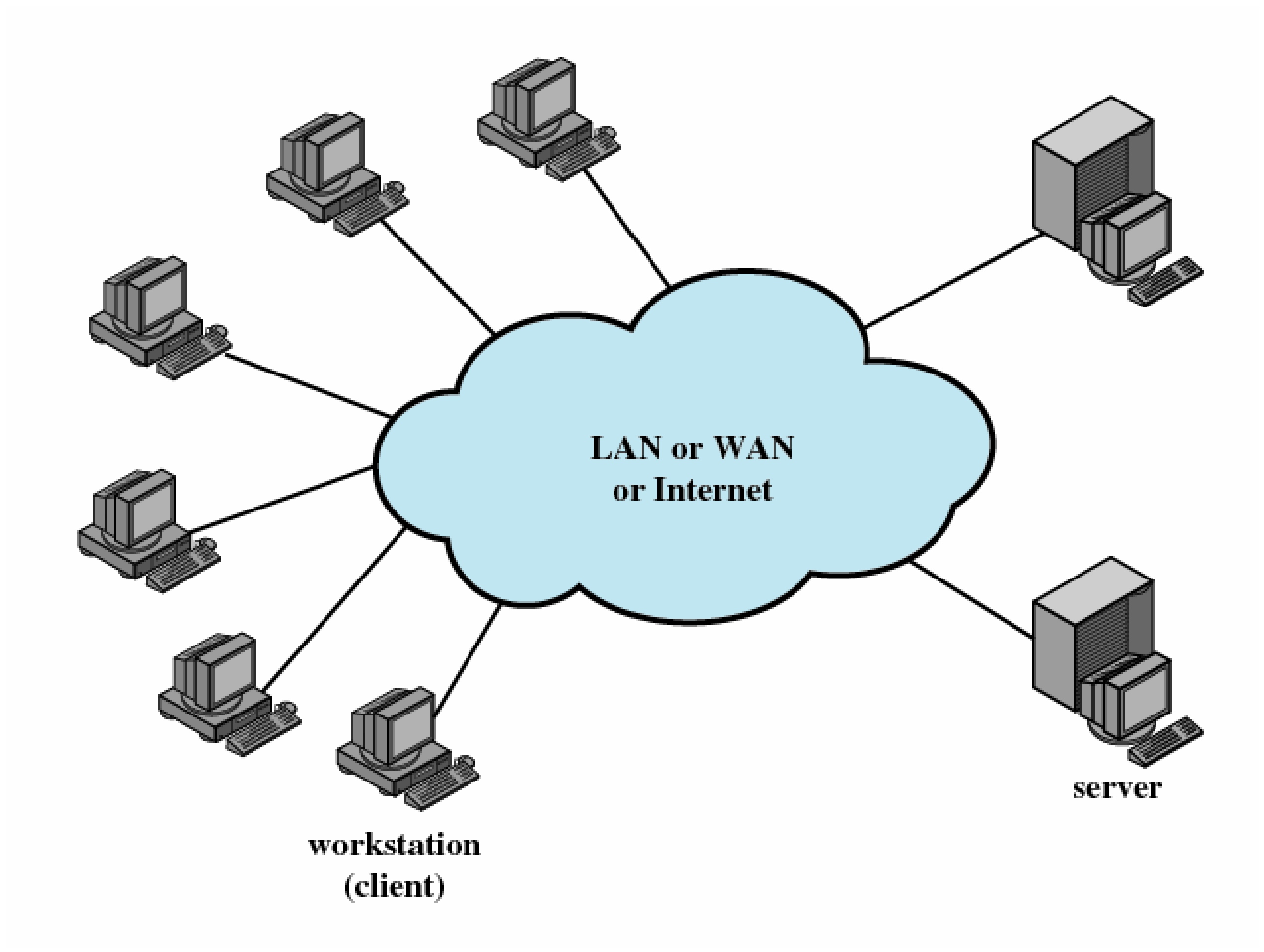
Client-Server Jargon

- **Applications Programming Interface (API)**
  - A set of function and call programs that allow clients and servers to intercommunicate.
- **Client**
  - A networked information requester, usually a PC or workstation, that can query database and/or other information from a server.
- **Middleware**
  - A set of drivers, APIs, or other software that improves connectivity between a client application and a server.
- **Relational Database**
  - A database in which information access is limited to the selection of rows that satisfy all search criteria.
- **Server**
  - A computer, usually a high-powered workstation, a minicomputer, or a mainframe, that houses information for manipulation by networked clients.
- **Structured Query Language (SQL)**
  - A language developed by IBM and standardized by ANSI for addressing, creating, updating, or querying relational databases.

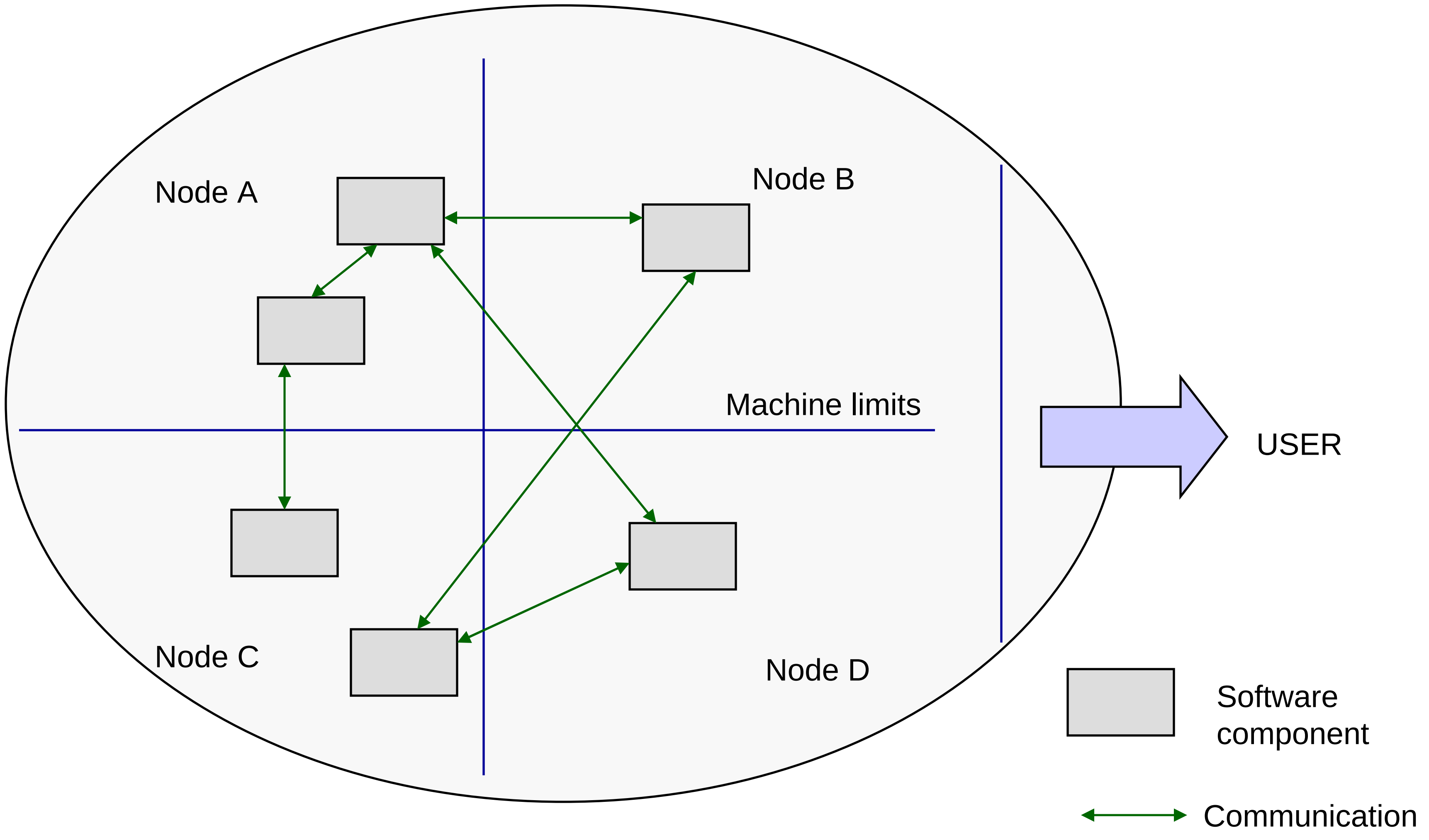
Client/Server Computing

- Client machines are generally single-user PCs or workstations that provide a highly user-friendly interface to the end user.
- Each server provides a set of shared services to the clients.
- The server enables many clients to share access to the same database and enables the use of a high-performance computer system to manage the database.

Generic Client-Server Environment



What is a Distributed System?



What is a Distributed Application?

- A distributed application is an application *A*, the functionality of which is subdivided into a set of cooperating subcomponents A_1, A_2, …, A_n with $n > 1$. The subcomponents A_i are autonomous processing units which can run on different computers and exchange information over the network controlled by coordination software.
- There are typically three levels defined for a distributed system:

Level 3	Distributed Applications
Level 2	Coordination Software
Level 1	Distributed Computer System

What is a Distributed Application? (cont.)

- The application on level 3 will ideally "know" nothing of the distribution of the system, as it uses the services of level 2, the administration software that takes over the coordination of all the components and hides the complexity from the application.
- In turn, level 2 itself uses the available distributed computing environment.

As an aside, a more humorous definition of a distributed system was given by Leslie Lamport (the guy who developed LaTex), who defined a distributed system as a system "in which my work is affected by the failure of components, of which I knew nothing previously."

Basic Communication Models

- Communication between the individual components of a distributed system can occur in two basic ways: using either shared memory or message passing.

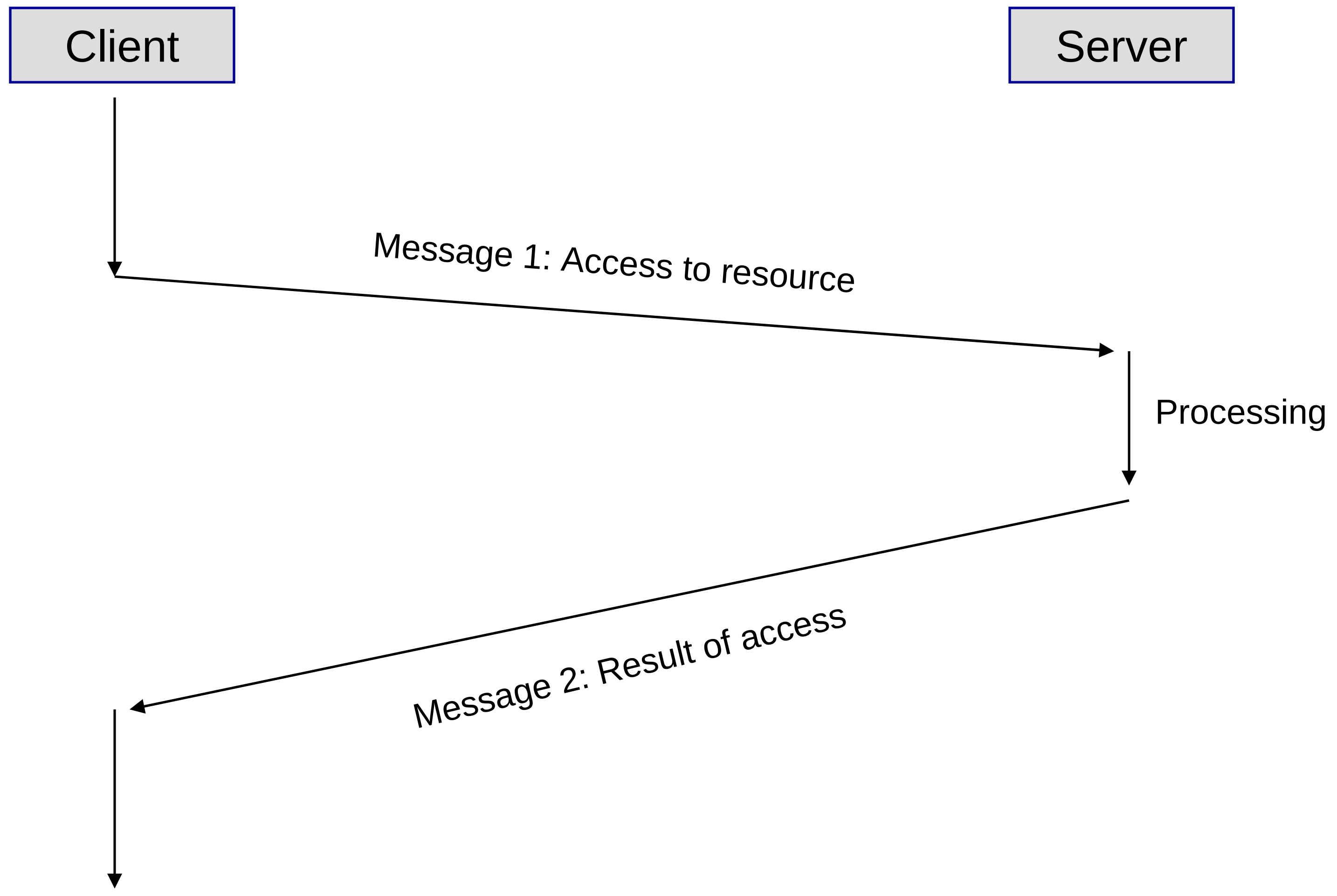
- Shared memory is an indirect form of communication, as both partners exchanging information do not communicate directly with each other, but via a third component: the shared memory.

- Message passing is a direct form of communication between the sender and receiver by means of a communication medium. Two functions are generally available for the execution of message exchange, usually called send and receive.

Basic Communication Models (cont.)

- Send is defined as: `send(r: recevier, m: message)`
  - This function sends the message *m* to the receiver *r*.
- Receive is defined as: `receive(s: sender, b: buffer)`
  - This function waits for a message from sender *s* and writes it in buffer *b* (part of the memory made available for the application process).
- The basic form of exchange of a single message can be combined with more complex models. One of the most important of these models is the client-server model.
  - In this model, the communication partners adopt the role of either a client or a server. A server is assigned to administer access to a certain resource, while a client wishes to use the resource.

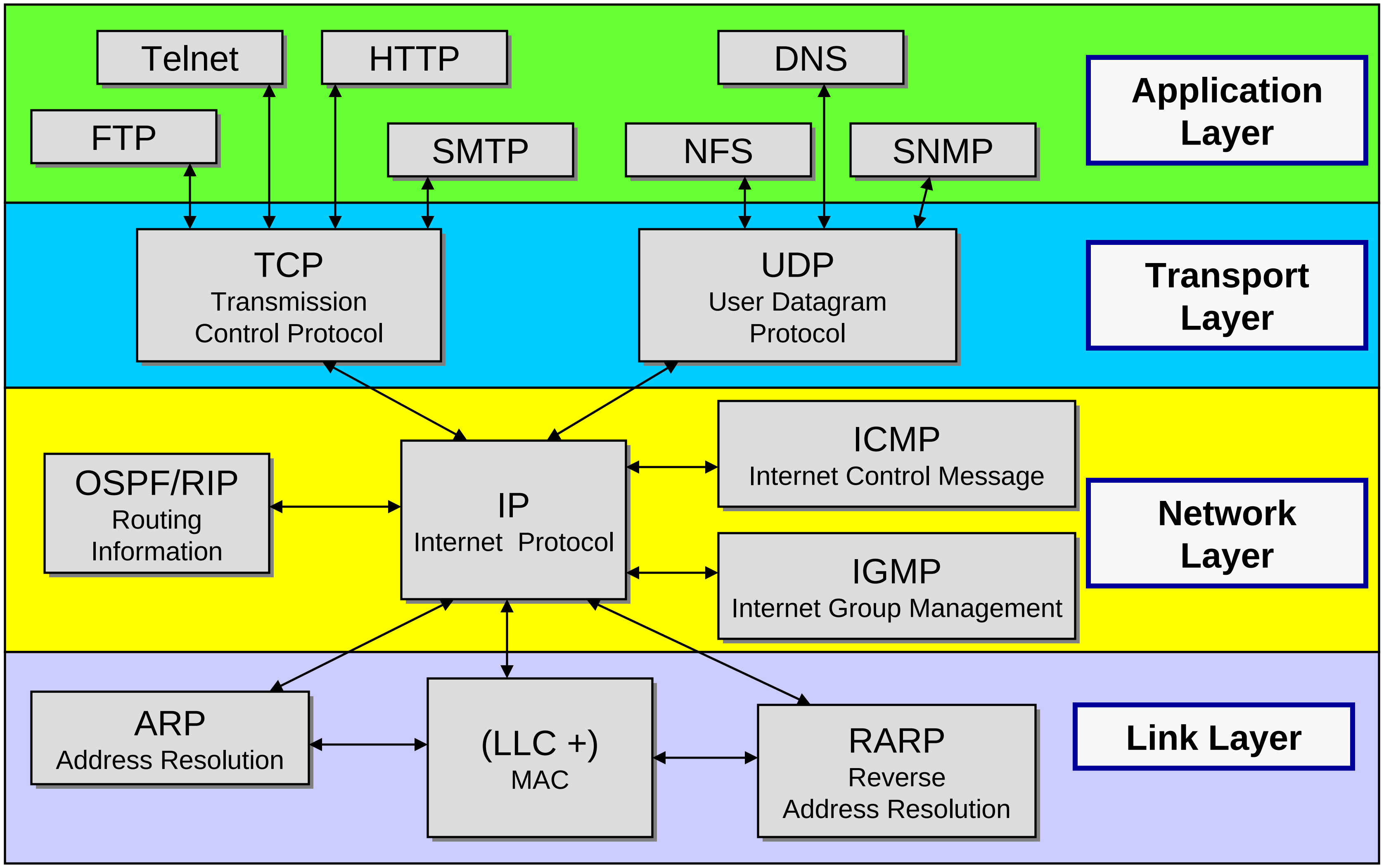
Message Exchange in the Client-Server Model



Technical Principles of the Internet

- Communications systems such as the Internet are best described using layered models because of their complexity.
- Every layer within the model has a certain task, and all layers together produce a particular communication service for the user.
- The layers are arranged in hierarchical form. Layers lower in the hierarchy produce a service used by the higher layers. The uppermost layer finally combines all lower layer services and constitutes the interface for applications.
- For the Internet, the so-called Internet reference model is used and is shown on the next slide.

Internet Reference Model


Telnet
HTTP
DNS
FTP
SMTP
NFS
SNMP
Application Layer
TCP Transmission Control Protocol
UDP User Datagram Protocol
Transport Layer
ICMP Internet Control Message
OSPF/RIP Routing Information
IP Internet Protocol
Network Layer
IGMP Internet Group Management
ARP Address Resolution
(LLC +) MAC
RARP Reverse Address Resolution
Link Layer

Internet Reference Model (cont.)

- The Link Layer describes the possible sub-networks of the Internet and their medium access protocols. These are, for example, Ethernets, token rings, FDDI, or ISDN networks. To its upper layer, the link layer offers communication between two computers in the same sub-network as a service.

- The Network Layer unites all the sub-networks to become the Internet. The service offered involves making communication possible between any two computers on the Internet. The network layer accesses the services of the link layer, in that a connection between two computers in different networks is put together for many small connections in the same network.

Internet Reference Model (cont.)

- The Transport Layer oversees the connection of two (or more) processes between computers communicating with each other via the network layer.

- The Application Layer makes application-specific services available for inter-process communication. These standardized services include e-mail, file transfer and the World Wide Web.

- Within the layers, protocols are used for the production of a service. Protocols are instances which can be implements either in hardware or software, and communicate with their partner instances in the same levels, but on other computers. It is only this cooperation that enables the service to be produced for the next level up.

Internet Reference Model (cont.)

- The TCP/IP Protocol constitutes the core of Internet communication technology in the transport and network layers.
- Every computer on the Internet always has an implementation of both protocols, TCP (Transmission Control Protocol) and IP (Internet Protocol).
- The task of IP is to transfer data from one Internet computer (the sender) to another (the receiver). On this basis, TCP then organizes the communication between the two processes on these two computers.

Some Important Application Layer Internet Protocols

- Telnet – makes a terminal emulation available on the remote computer. The protocol enables logins to other computers using the network.
- HTTP – (Hypertext Transport Protocol) is the underlying protocol of the World Wide Web. It is responsible for the transfer of hypertext documents.
- SMTP – (Simple Mail Transfer Protocol) is the protocol used for the transfer of e-mail messages.
- FTP – (File Transfer Protocol) is able to manage filestores on a server and enables clients to access files.
- SNMP – (Simple Network Management Protocol) is used for network management on the Internet.
- DNS – (Domain Name Service) is responsible for the mapping of symbolic names to IP addresses.
- NFS – (Network File System) makes the basic functionality for a distributed file system available.

Basic Constituents of Web Applications

- In order to access the Web, first a web server is required. The server administers the entire data material intended for publication on the Web.
- The web server is also responsible for replying to client requests, by delivering the desired documents according to the entitlement of the client.
- Web servers usually record all Web files access, so different analyses can be made using the log files created, from the simplest of tasks such as how many hits have been made in a certain time period, or an analysis of the geographical distribution of users, to more sophisticated tasks such as monitoring attempts at unauthorized access.
- Web servers might also start other programs executing, with which additional information can be obtained or generated. It is this capability that forms the basis of all distributed applications on the WWW.

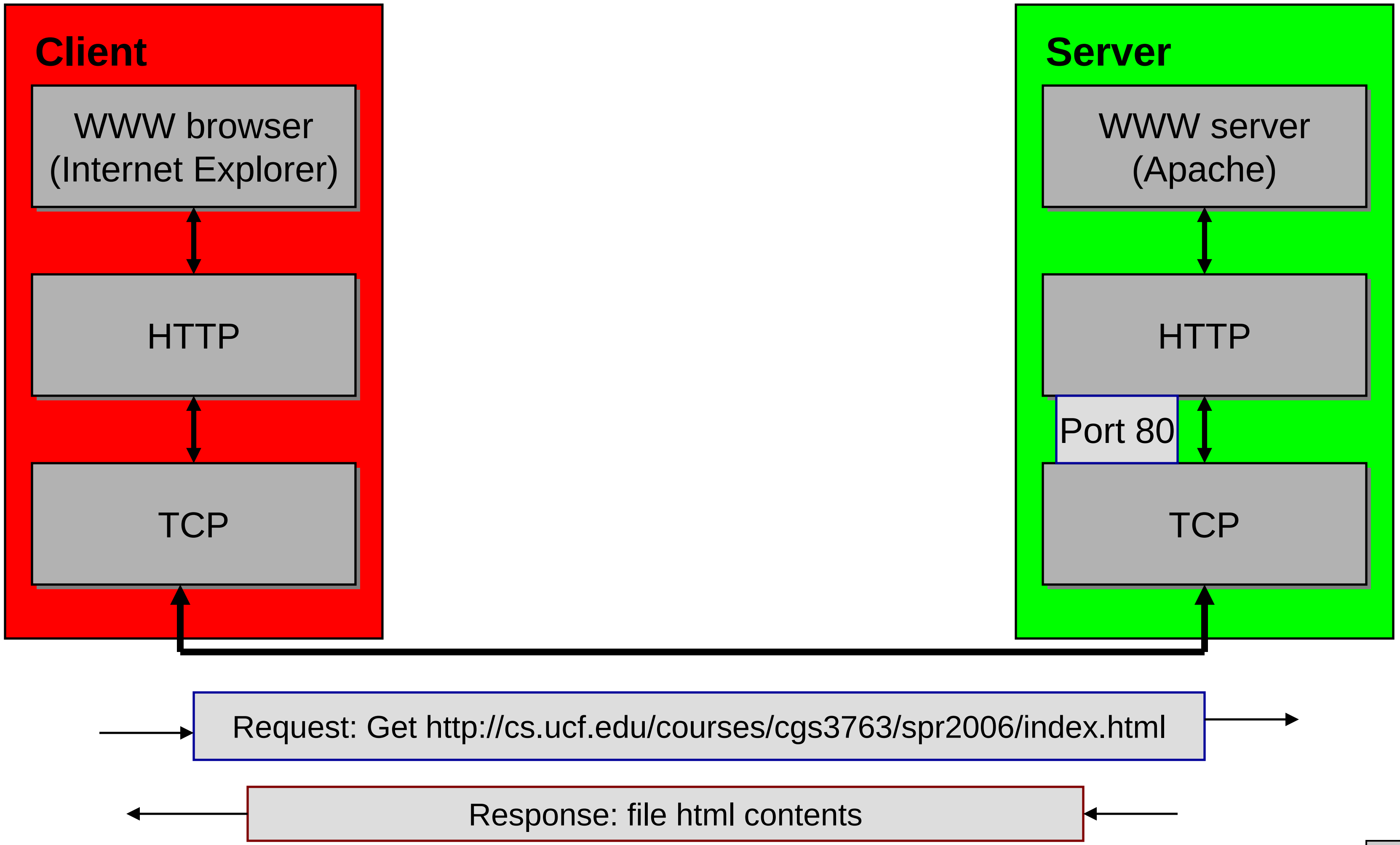
Basic Constituents of Web Applications (cont.)

- The communication between client and server on the WWW takes place using the HTTP protocol.

- HTTP is a purely text-based protocol. This means that the requests for a document are transferred by the client to the server using a “readable” command such as “get”. The server responds to the request by making the requested document available to the client, together with a header giving further information. The server may sometimes respond with an error message, such as if the requested document is not available or the user does not have the proper permission to view the document.

- This protocol is illustrated on the next page. HTTP uses the TCP service for the actual transfer of data between client and server. For every transfer of a Web document, a TCP connection is first established, via which HTTP protocol messages are transferred. (Actually, the TCP connection persists over several HTTP requests.)

The Architecture of a Web Service


Client
WWW browser (Internet Explorer)
HTTP
TCP
Server
WWW server (Apache)
HTTP
Port 80
TCP
Request: Get http://cs.ucf.edu/courses/cgs3763/spr2006/index.html
Response: file html contents

Construction of Web Applications

- The simplest form of an application on the WWW is that in which the provider places a number of static documents on a server.
- A static document is a document which can only be changed from outside, by human intervention.
- Clearly, this prototype is not flexible. As soon as information has to be modified on the server, a slow and cost-intensive process is required. For many applications, this process is simply not an option. Consider, for example, a provider that publishes current weather information. Since this information is constantly changing, an employee would need to be constantly updating the web pages.
- There are a number of approaches today which allow for the dynamic creation of web pages, some of which are already fairly old and others are relatively new. Among the newer of these are Java Servlets and Java Server Pages.

The Architecture of Distributed Web Applications

- There are basically four major components which can or could constitute a distributed application on the Internet. These are:

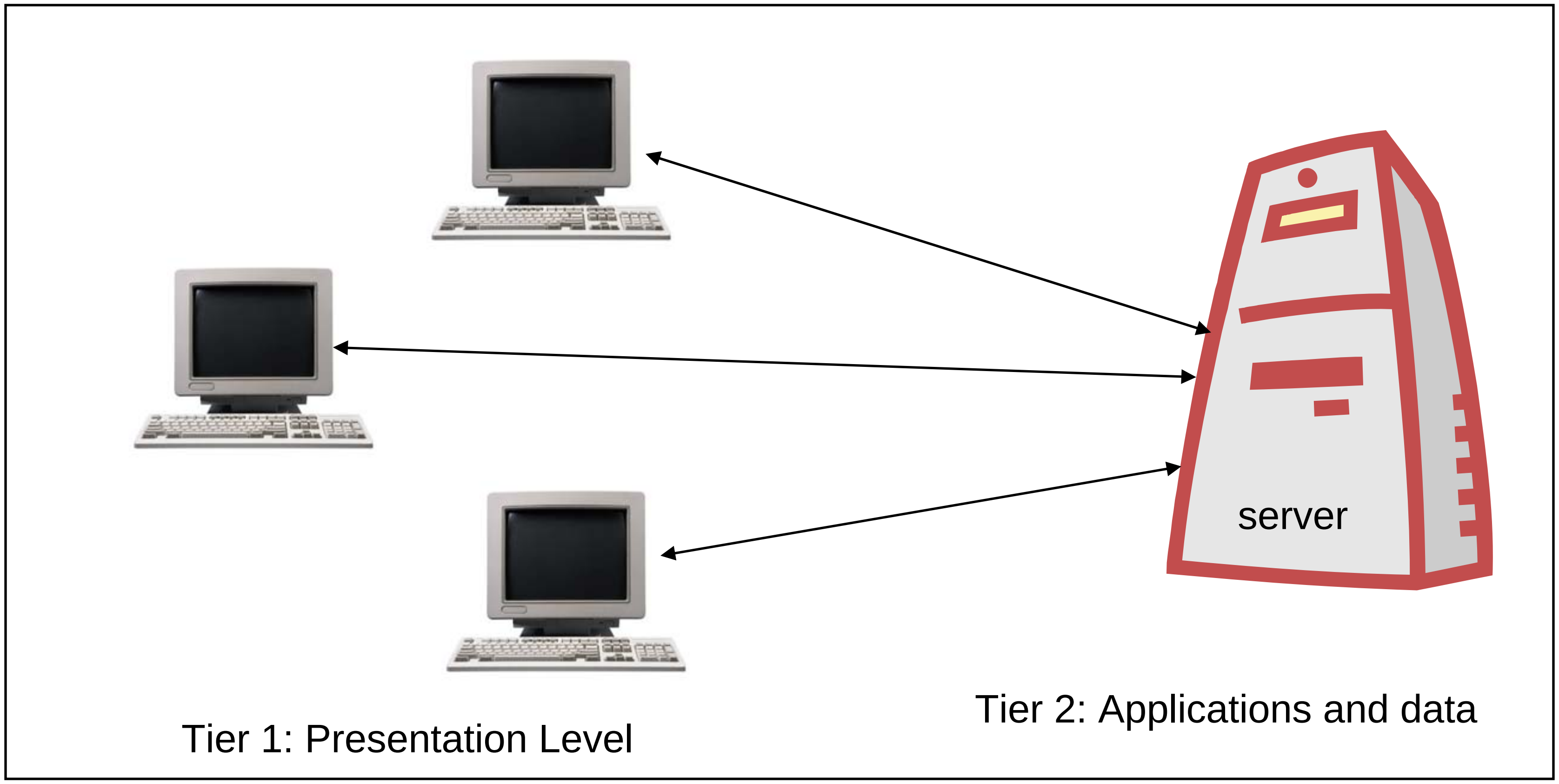
  1. The presentation interface to the user, as well as access programs to server components.
  2. An access interface to server components.
  3. The server application logic.
  4. File storage, databases, etc.

- In distributed applications, these four generic components can be distributed on the physical nodes of the system in different configurations.

- The term n-tier architecture was coined for the different variants that can be produced. The term indicates the number of levels on which the components are distributed. In practice, 2-, 3-, and 4-tier architectures are used.

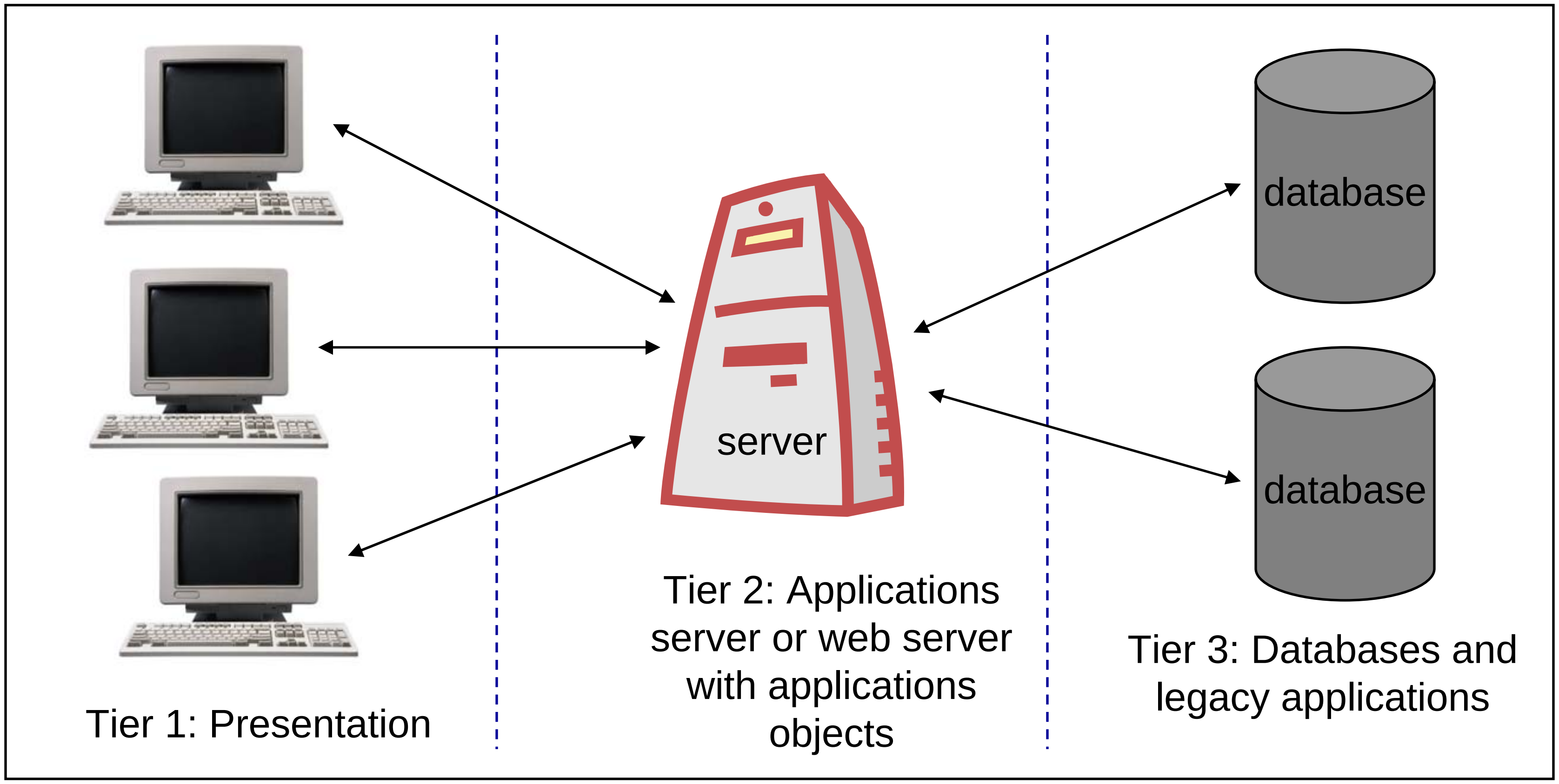
A Two-Tier Architecture

- The simplest version is the 2-tier architecture in which the presentation components are placed on the client computers, and all other components reside on one server computer. The most common example of this is TCP based client-server applications in which databases are accessed directly from the server process.



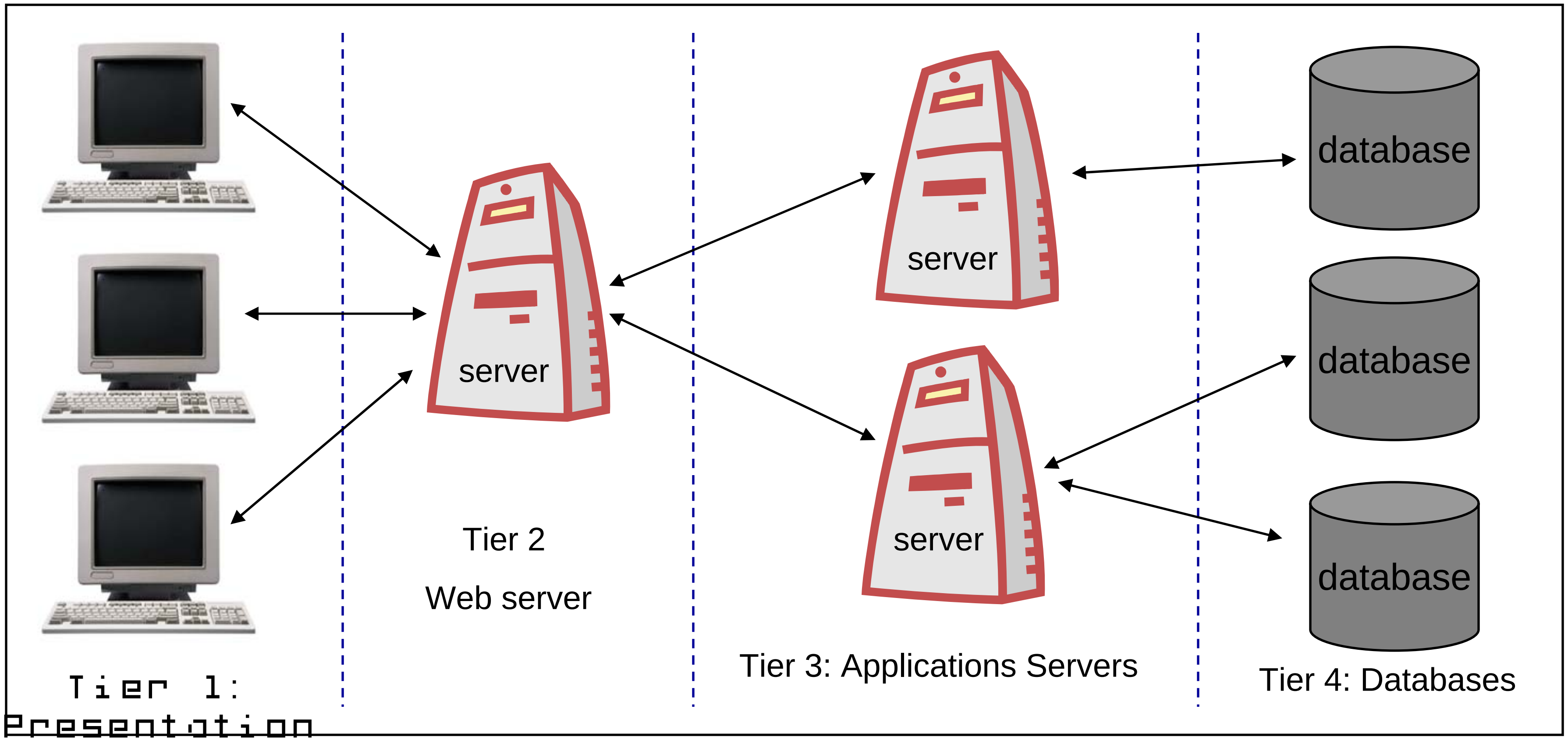
A Three-Tier Architecture

- The 3-tier architecture goes one step further, so that the actual applications are separated from data stocks. This is the common configuration for most servlet applications.


server
database
database
Tier 1: Presentation
Tier 2: Applications server or web server with applications objects
Tier 3: Databases and legacy applications

A Four-Tier Architecture

- The 4-tier architecture refines the 3-tier version by partitioning the server interface from the applications. Although not as common as the 3-tier version, this is the common configuration for many CORBA applications.



Thin Clients

- The extensive partitioning of the architecture of a distributed system into different components with respectively different areas of responsibility, basically allows for the creation of simpler and therefore more controllable individual components.
- The result of this on the client side is the development of thin clients. A thin client is a client program which contains almost no application logic, but offers only the presentation interface to the actual application program, which may run in a distributed fashion on several servers.
- While the application is executed and the graphical interface is in use, application logic is partly loaded on the client computer and executed there locally. However, it is not loaded from the local hard drive, but always by a server via the network.
- The most common version of a thin client today is a web browser. A web browser has no information whatsoever on specific applications.

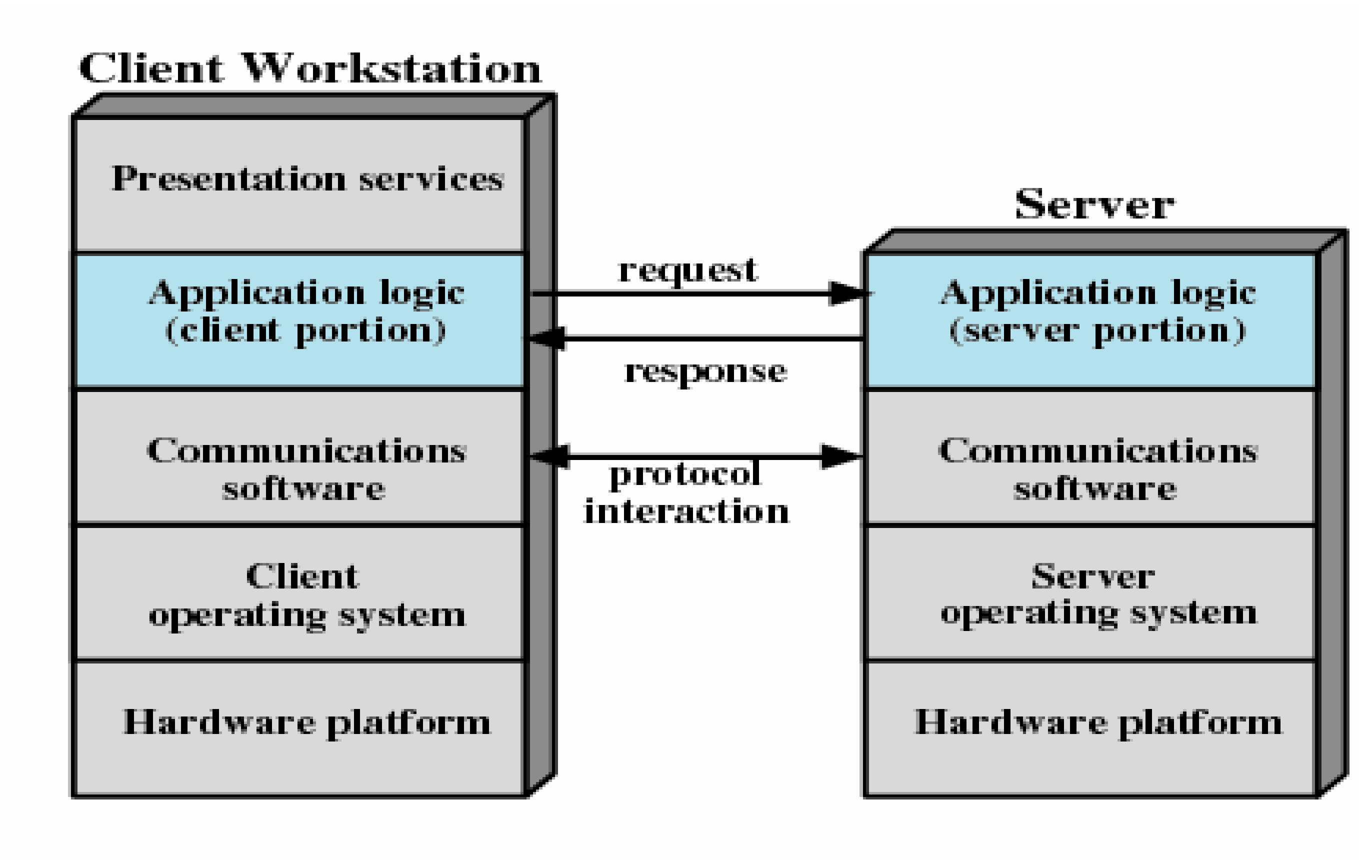
Thin Clients (cont.)

- A web browser is only able to represent web pages, and possibly execute applets.
- If a certain application is to be used, then the corresponding web pages must be loaded by a web server.
- The use of thin clients has several advantages (as opposed to heavy clients):
  - The installation of program components on the client computer is unnecessary. Neither a reconfiguration of the computer nor regular updates of client software are required.
  - Users do not have to adjust to a new user interface for every distributed application. Access is always made using a well-known web browser interface. This renders a potentially large amount of training unnecessary.
  - Client computers can, on the whole, be equipped more inexpensively, as large hard drives for storing application programs are not needed.

Client-Server Applications

- Basic software is an operating system running on the hardware platform.
- Platforms and the operating systems of client and server may differ.
- These lower-level differences are irrelevant as long as a client and server share the same communications protocols and support the same applications.

Generic Client-Server Architecture



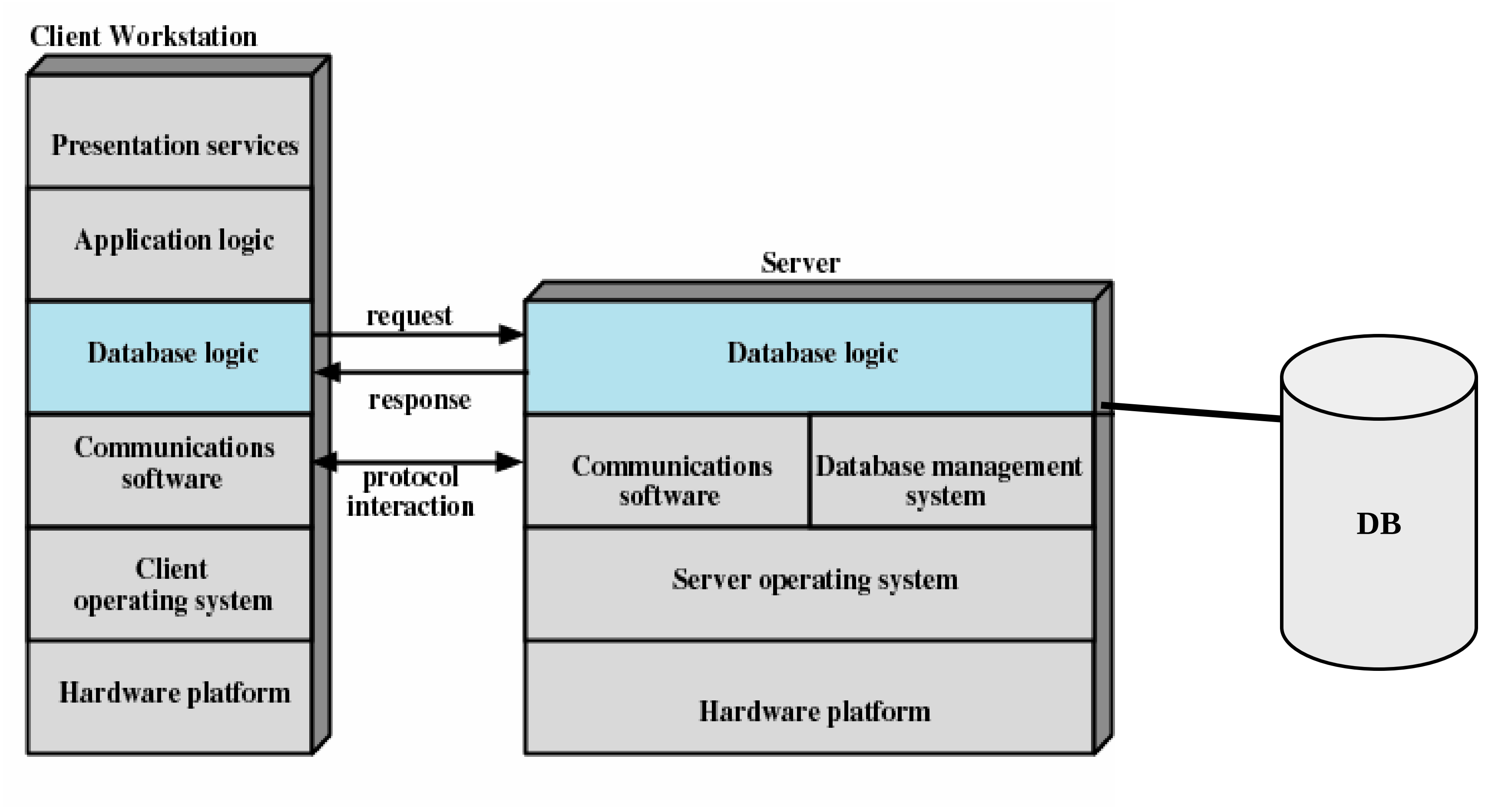
Client-Server Applications

- The bulk of application software executes on the server.
- Application logic is located at the client.
- Presentation services are in the client.

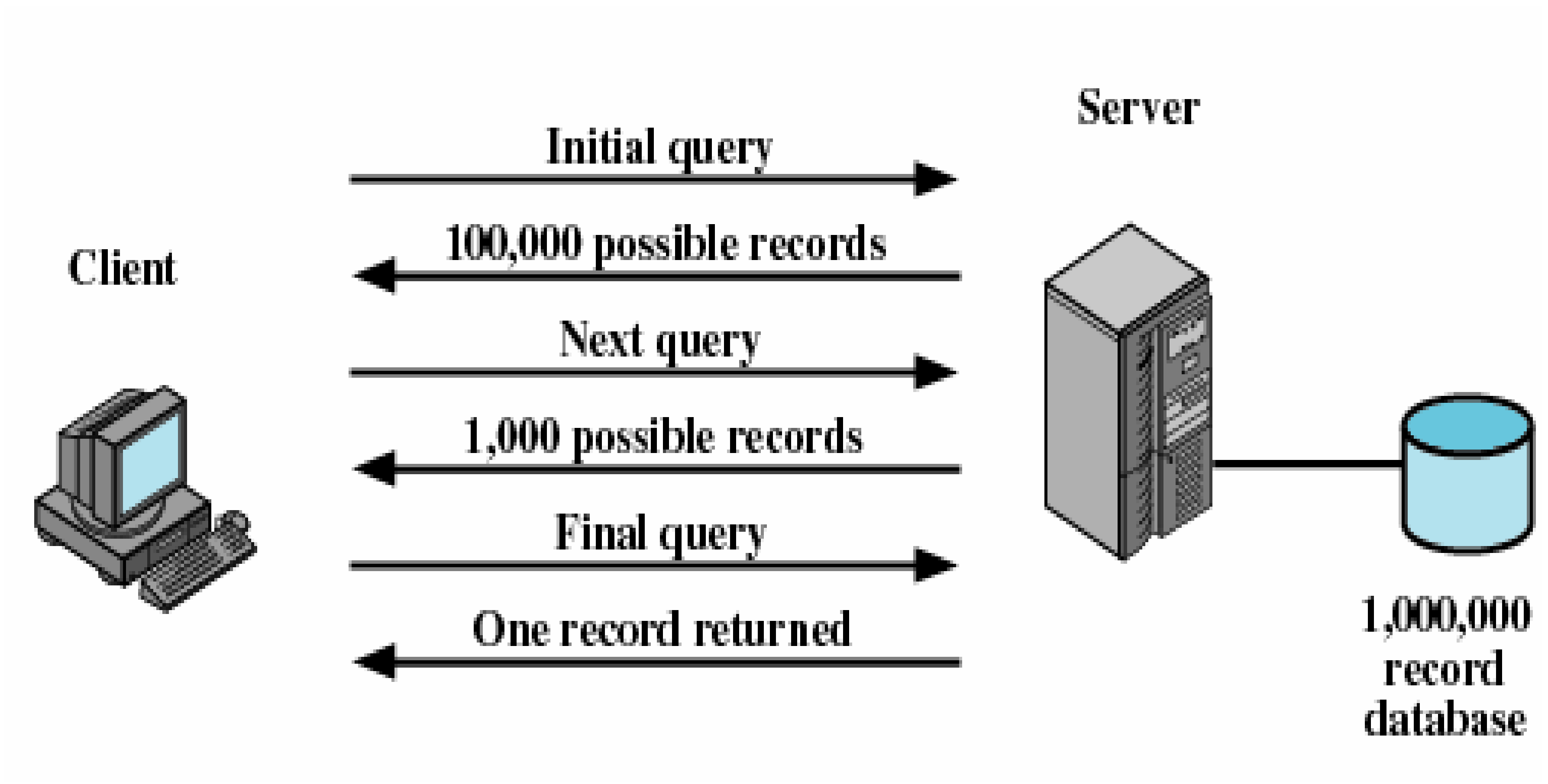
Database Applications

- The server is a database server.
- Interaction between client and server is in the form of transactions against the database.
  - the client makes a database request and receives a database response.
- Server is responsible for maintaining the database.

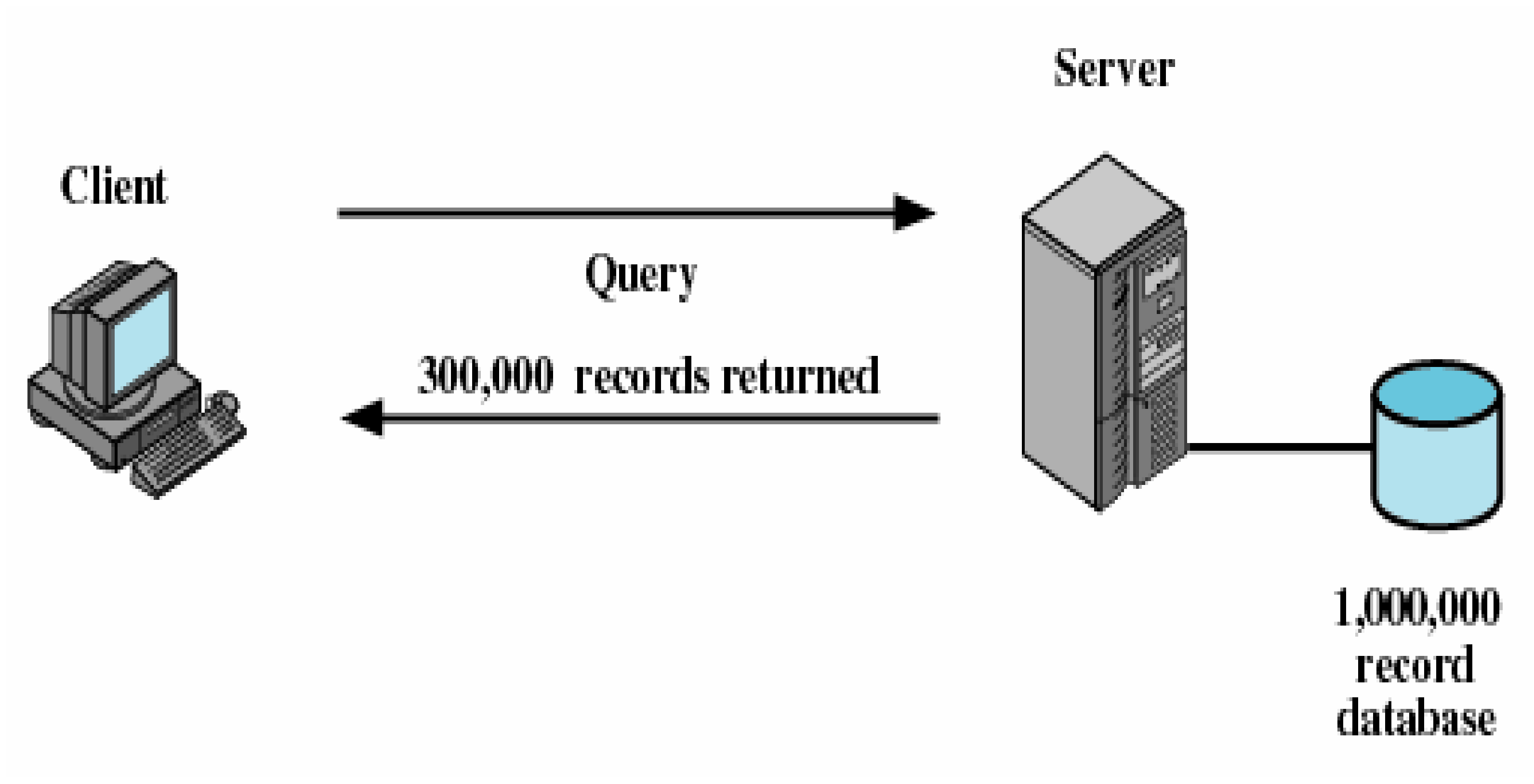
Client-Server Architecture for Database Applications



Desirable Client/Server Database Usage

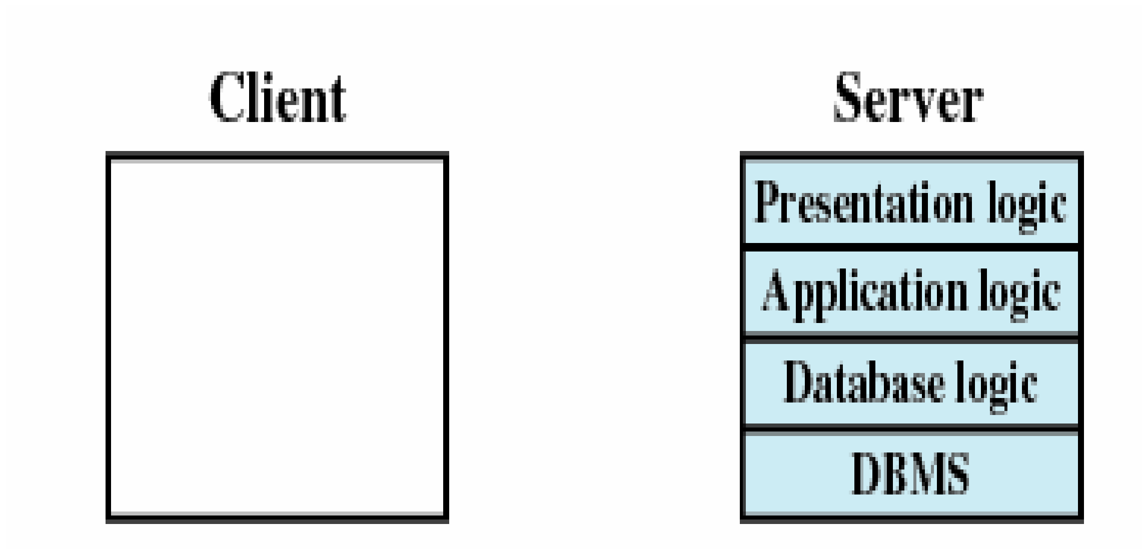


Undesirable Client/Server Database Usage



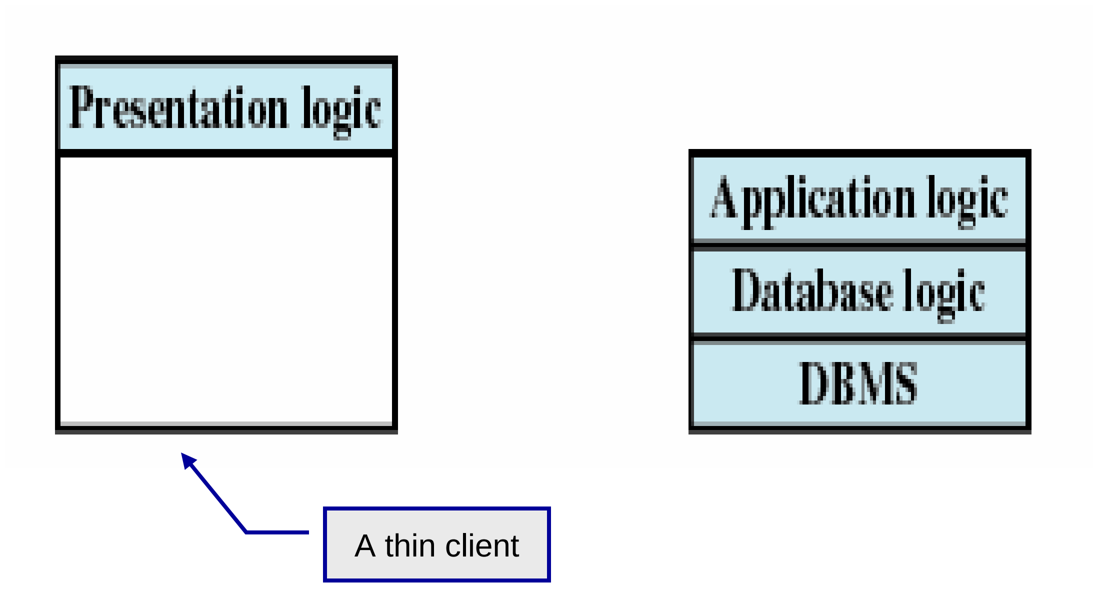
Classes of Client/Server Applications

- Host-based processing
  - Not true client/server computing
  - Traditional mainframe environment



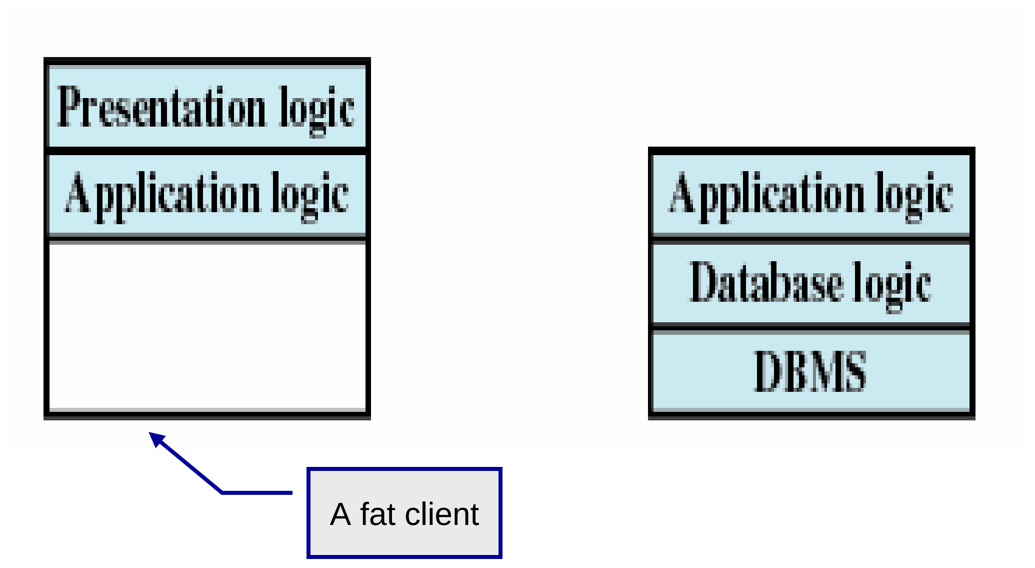
Classes of Client/Server Applications (cont.)

- Server-based processing
  - Server does all the processing
  - Client provides a graphical user interface



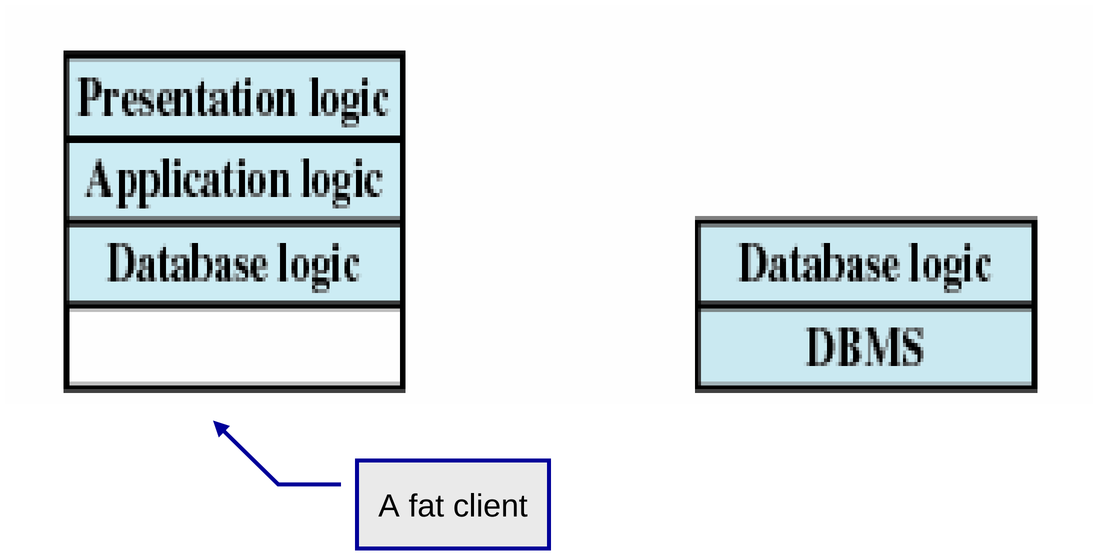
Classes of Client/Server Applications (cont.)

- Client-based processing
  - All application processing done at the client.
  - Data validation routines and other database logic functions are done at the server.



Classes of Client/Server Applications (cont.)

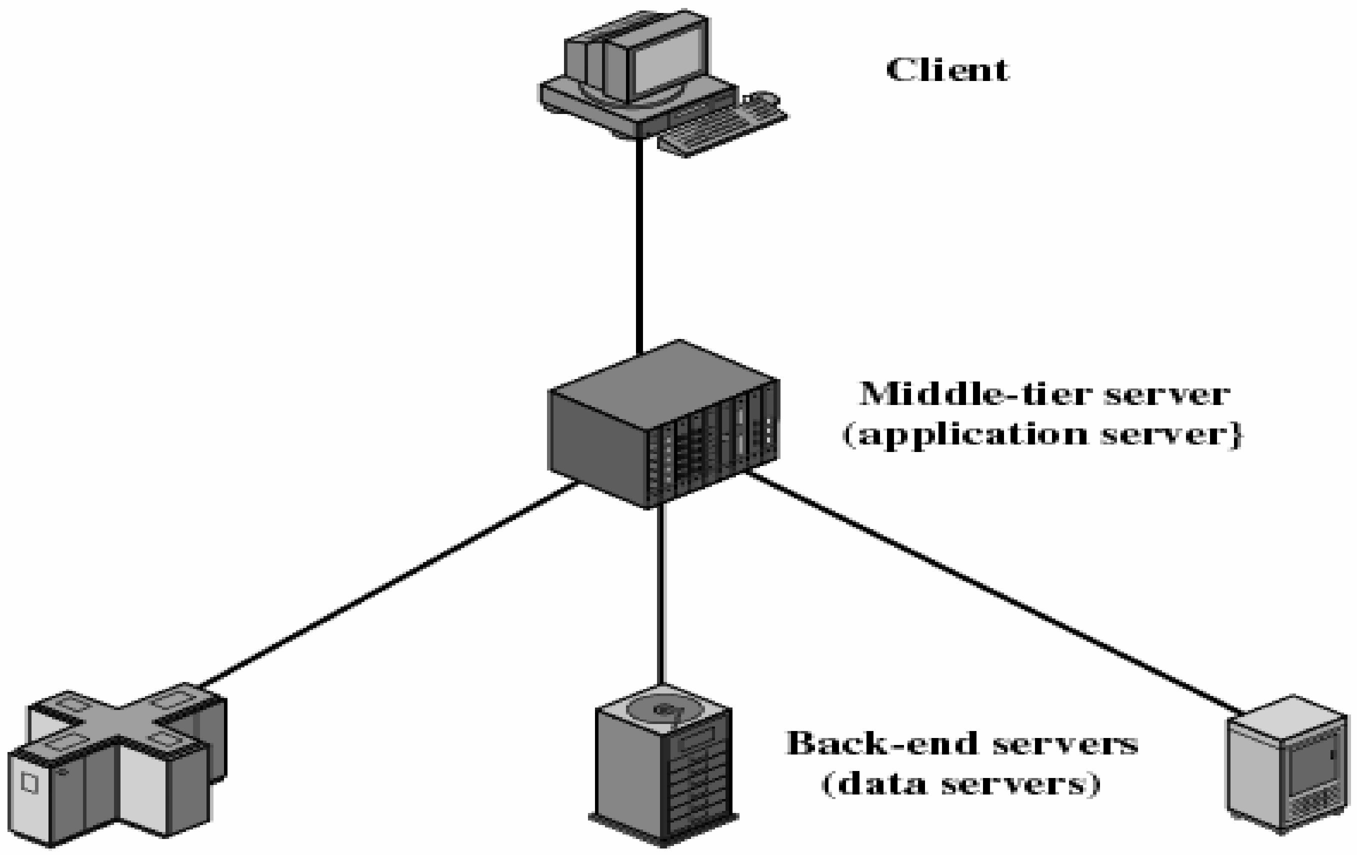
- Cooperative processing
  - Application processing is performed in an optimized fashion.
  - Complex to set up and maintain.



Three-Tier Client/Server Architecture

- Application software distributed among three types of machines
  - Client (user) machine
    - Thin client
  - Middle-tier server
    - Gateway
    - Convert protocols
    - Merge/integrate results from different data sources
  - Backend server

Three-Tier Client/Server Architecture


Client
Middle-tier server
(application server}
Back-end servers
(data servers)