Client-Server Applications

Prof. Sanjeev Setia

Distributed Software Systems

CS 707

Network Protocols

Common bandwidths available

Service	Bandwidth
ISDN	64 Kbps
T1	1.544 Mbps
T3	44.7 Mbps
STS-1	51.84 Mbps
STS-3	155.25 Mbps
STS-12	622 Mbps
STS-24	1.233 Gbps
STS-48	2.488 Gbps

Issues in Networks

- Naming
- Routing strategies – fixed, virtual circuit, dynamic
- Connection strategies – connection-oriented vs connectionless
- Contention

Local area network technologies and their transfer rates (from page 62)

Network type	*Standard*	*Data transfer rate (megabits per second)*
Ethernet	IEEE 803.2	10
FDDI	FDDI-I	100
IBM token ring	IEEE 803.5	4 or 16
Apple LocalTalk	–	0.23

Figure 3.1 A typical campus internetwork.

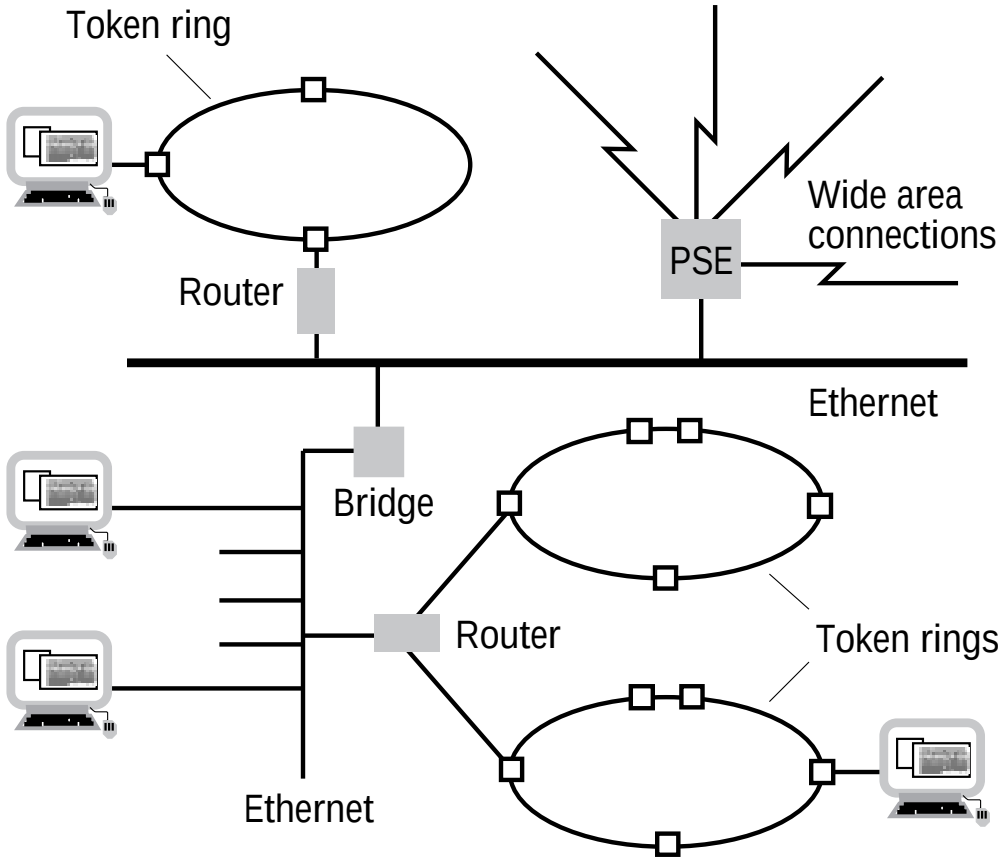


Figure 3.2 A wide area network.

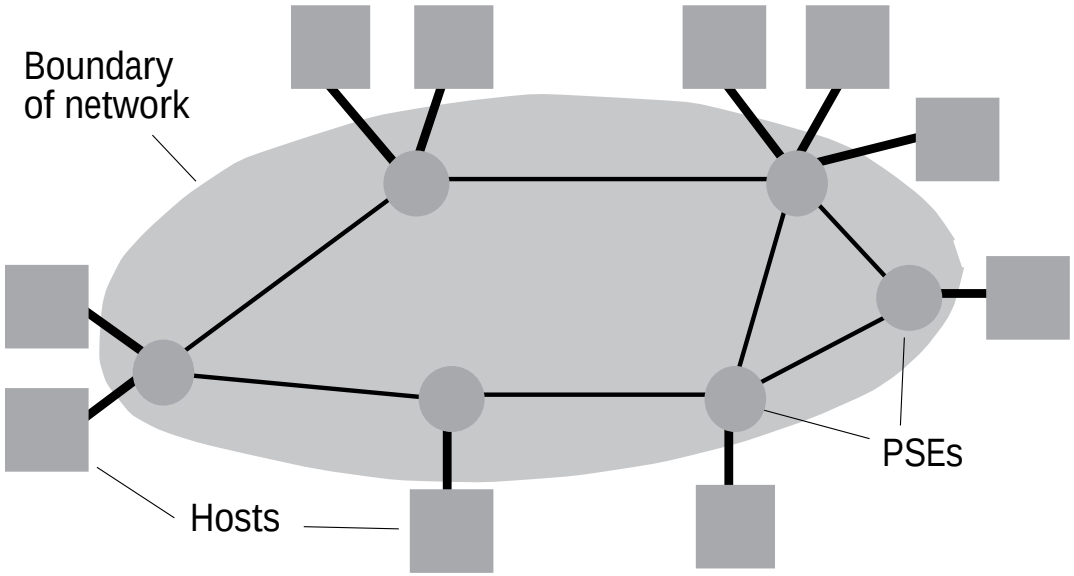


Figure 3.3 Bus topologies.

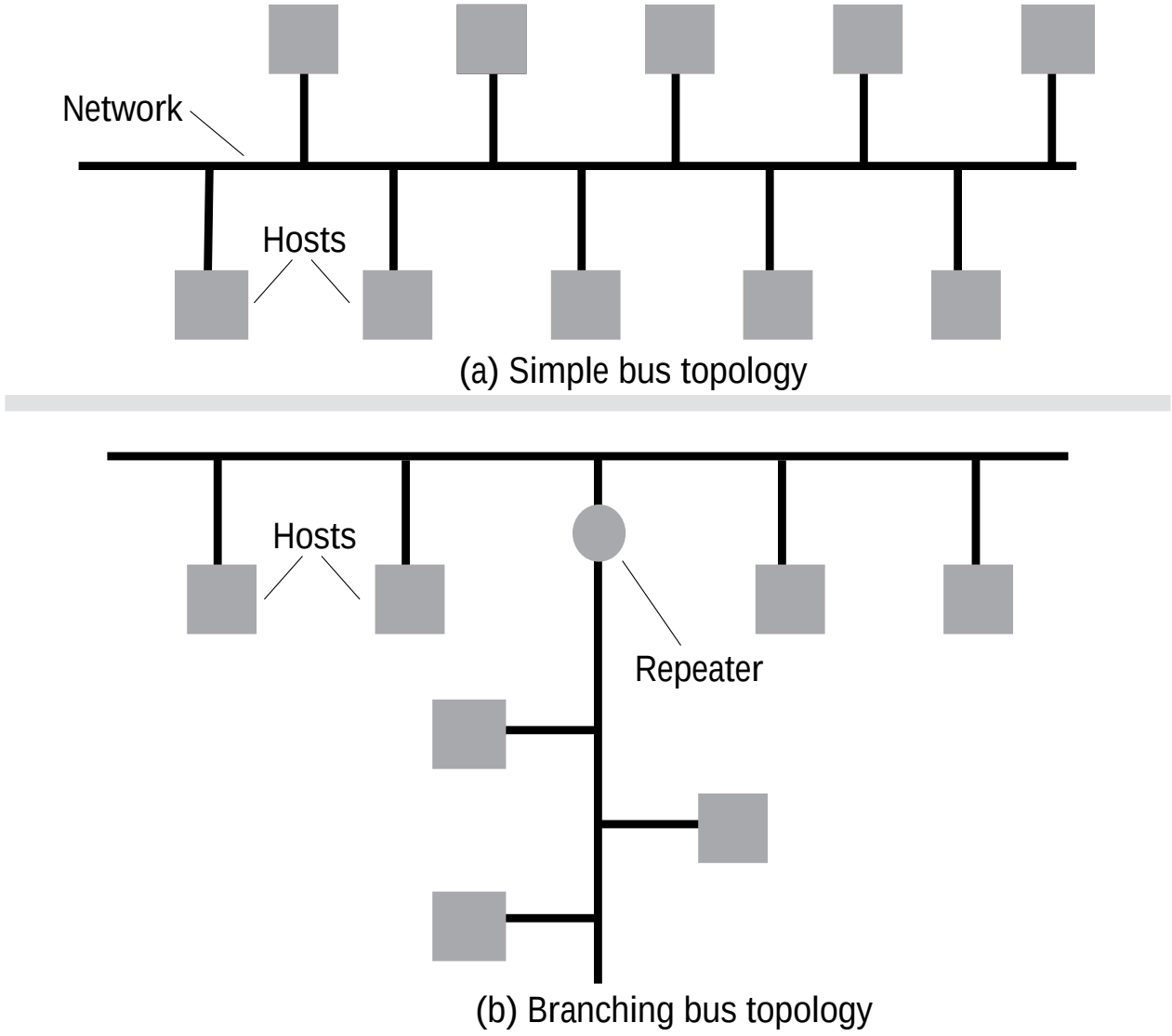


Figure 3.4 Ring topology.

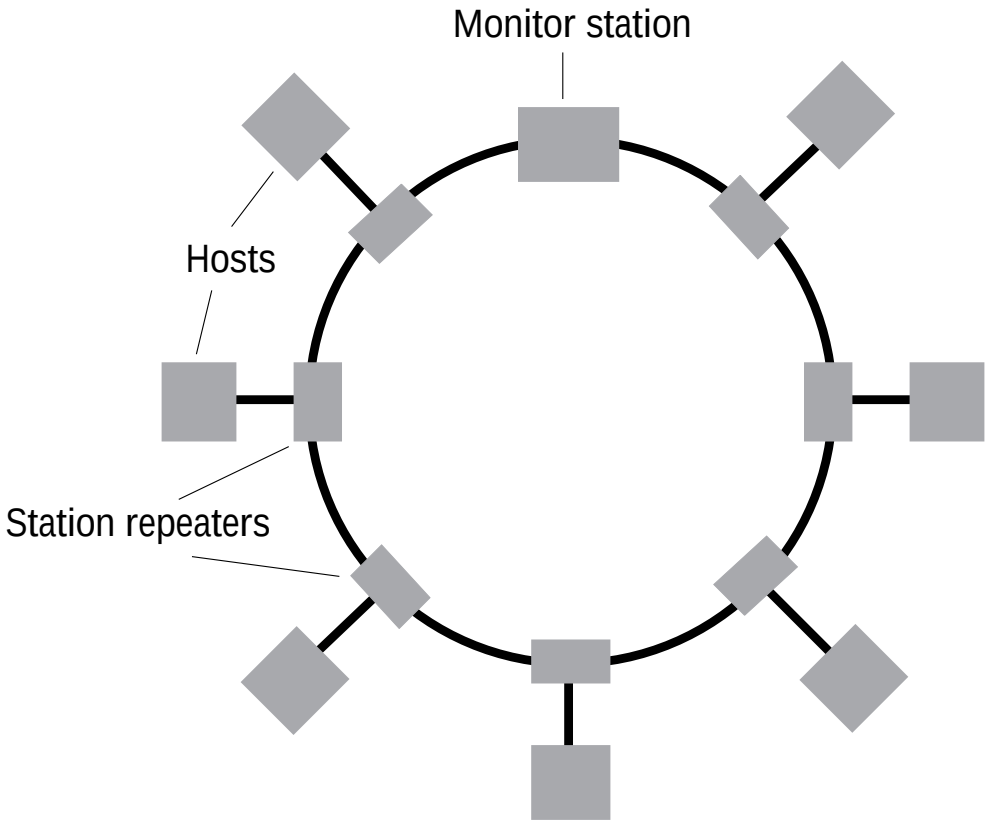


Figure 3.5 Conceptual layering of protocol software.

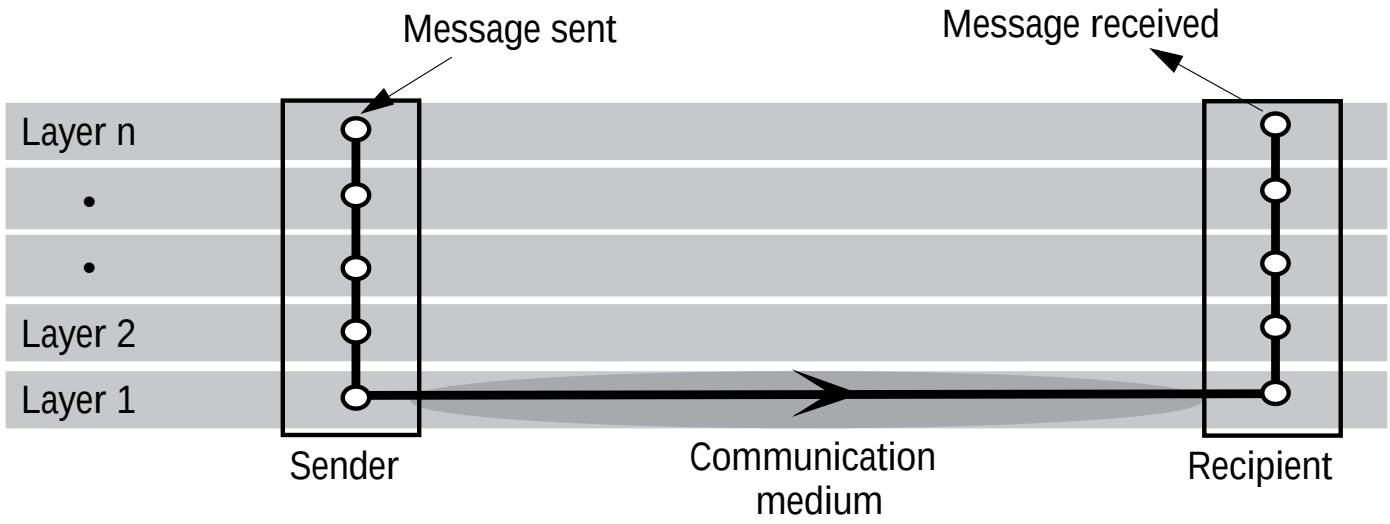


Figure 3.6 Protocol layers in the ISO *Open Systems Interconnection (OSI)* protocol model.

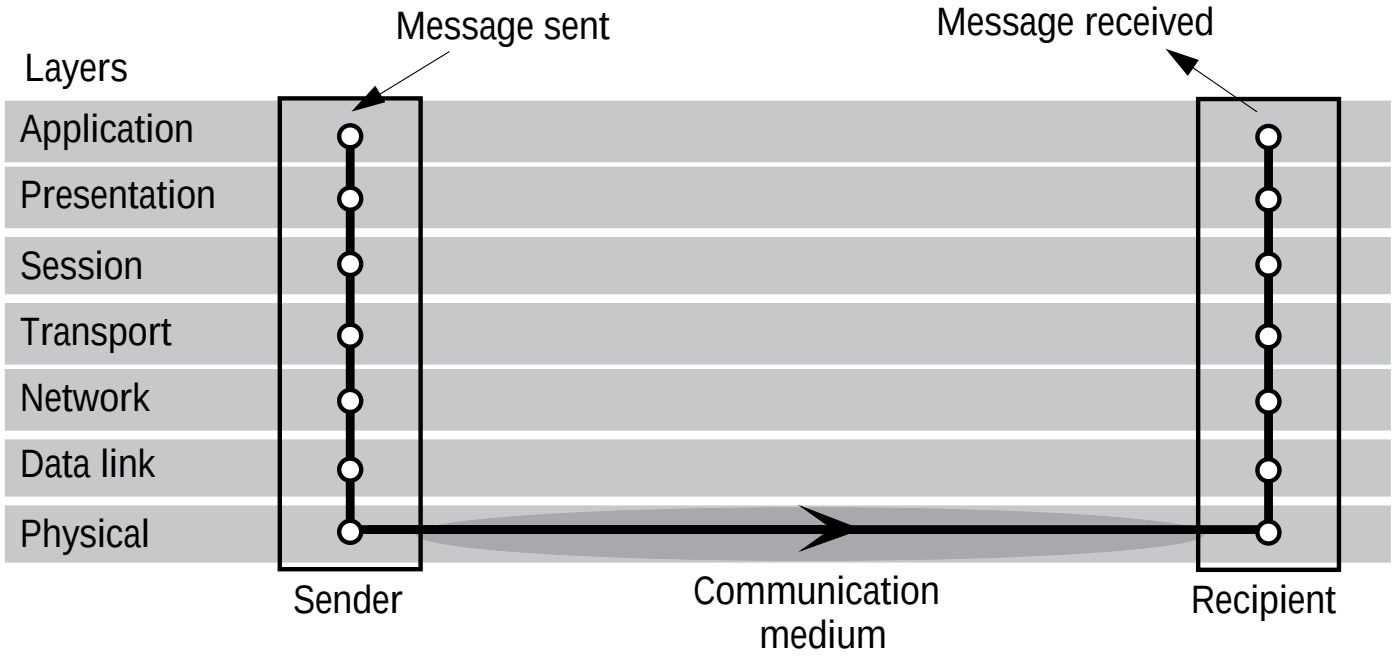
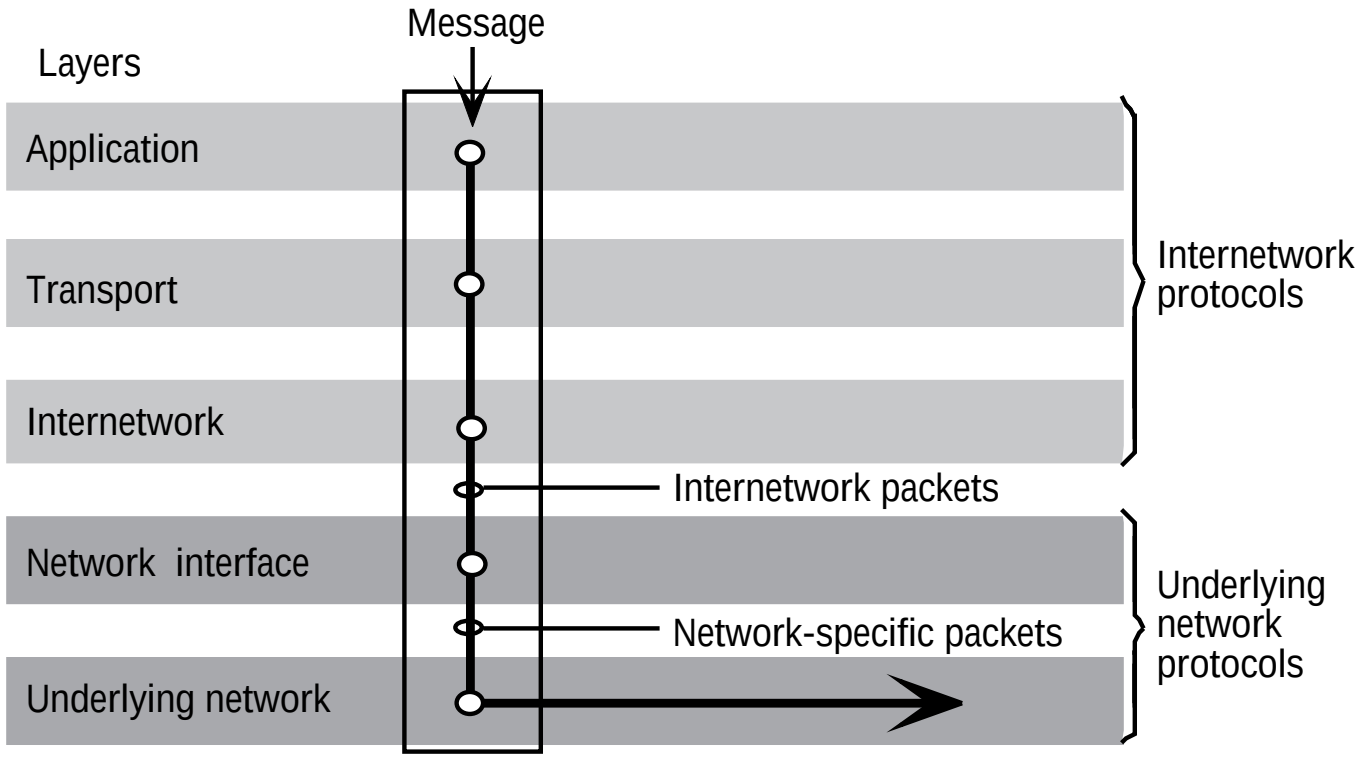


Figure 3.7 OSI protocol summary.

Layer	*Description*	*Examples*
Application	Protocols that are designed to meet the communication requirements of specific applications, often defining the interface to a service.	FTP, Telnet, SMTP, X400, X500
Presentation	Protocols at this level transmit data in a network representation that is independent of the representations used in individual computers, which may differ. Encryption is also performed in this layer, if required.	XDR, ASN.1, encryption
Session	At this level communication between processes is established and error recovery is performed. It is not required for connectionless communication.	
Transport	This is the lowest level at which messages (rather than packets) are handled. Messages are addressed to communication ports, Protocols in this layer may be connection-oriented or connectionless.	TCP, UDP
Network	Transfers data packets between computers in a specific network. In a WAN or an internetwork this involves the generation of a route passing through PSEs or routers. In a single LAN no routing is required.	X25,IP
Data link	Responsible for error-free transmission of packets between computers that are directly connected. In a WAN the connections are between pairs of PSEs and PSEs and hosts. In a LAN they are pairs of hosts.	HDLC Ethernet: CSMA/CD
Physical	The circuits and hardware that drives the network. It transmits sequences of binary data by analogue signalling, using amplitude or frequency modulation of electrical signals (on cable circuits), light signals (on fibre-optic circuits) or electromagnetic signals (on radio and microwave circuits).	X.21 Ethernet: baseband signalling

Figure 3.8 Internetwork layers.



Ethernet packet layout (from page 78):

6 bytes	*6 bytes*	*2 bytes*	*46 bytes ≤ length ≤ 1500 bytes*	*4 bytes*
Destination address	Source address	Type	Data for transmission	Frame check sequence

Token ring packet layout (from page 81):

3 bytes	*6 bytes*	*6 bytes*	*≤ 5000 bytes*	*4 bytes*	*1 byte*	*1 byte*
Token	Destination address	Source address	Data for transmission	Frame check seq.	End delimiter	Frame status

A token has the following format:

1 byte	*1 byte*	*1 byte*
Starting delimiter	Access control	Frame control

Figure 3.9 ATM protocol layers.

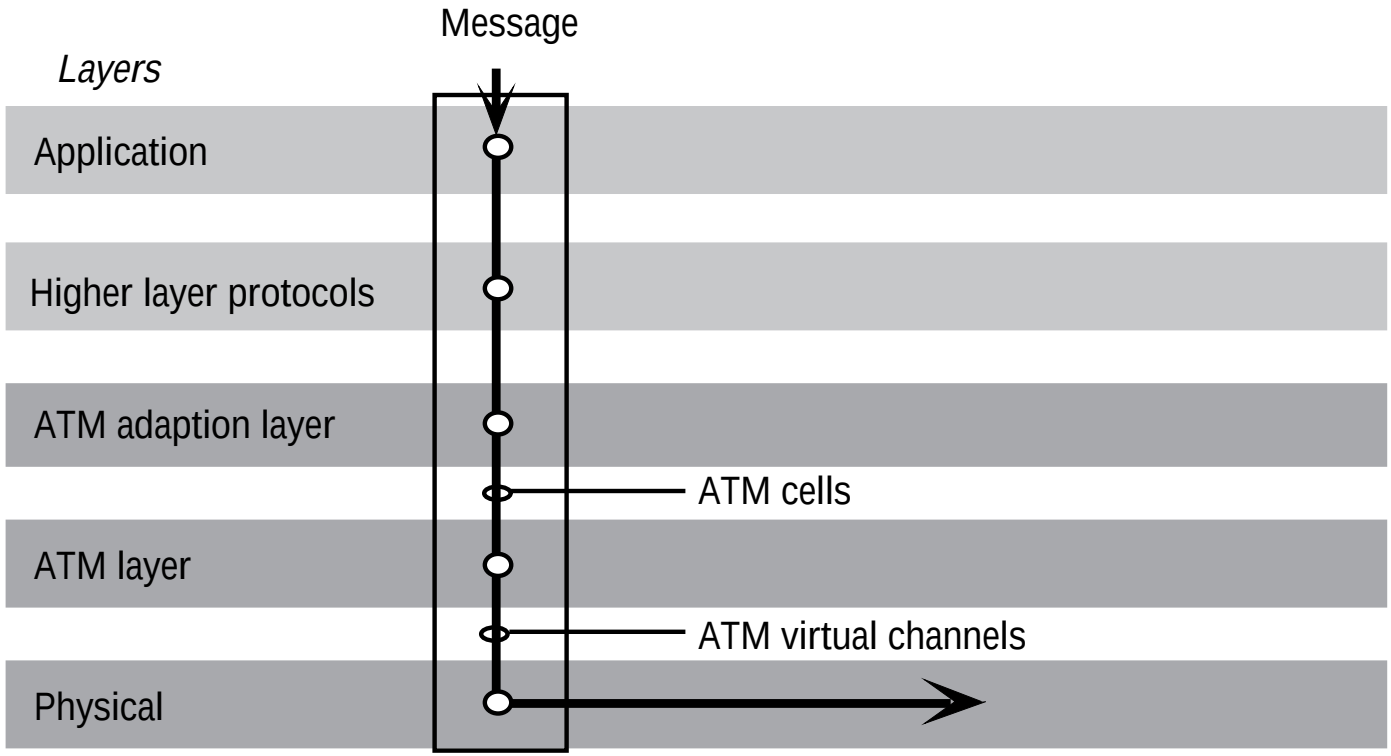


Figure 3.12 TCP/IP layers.

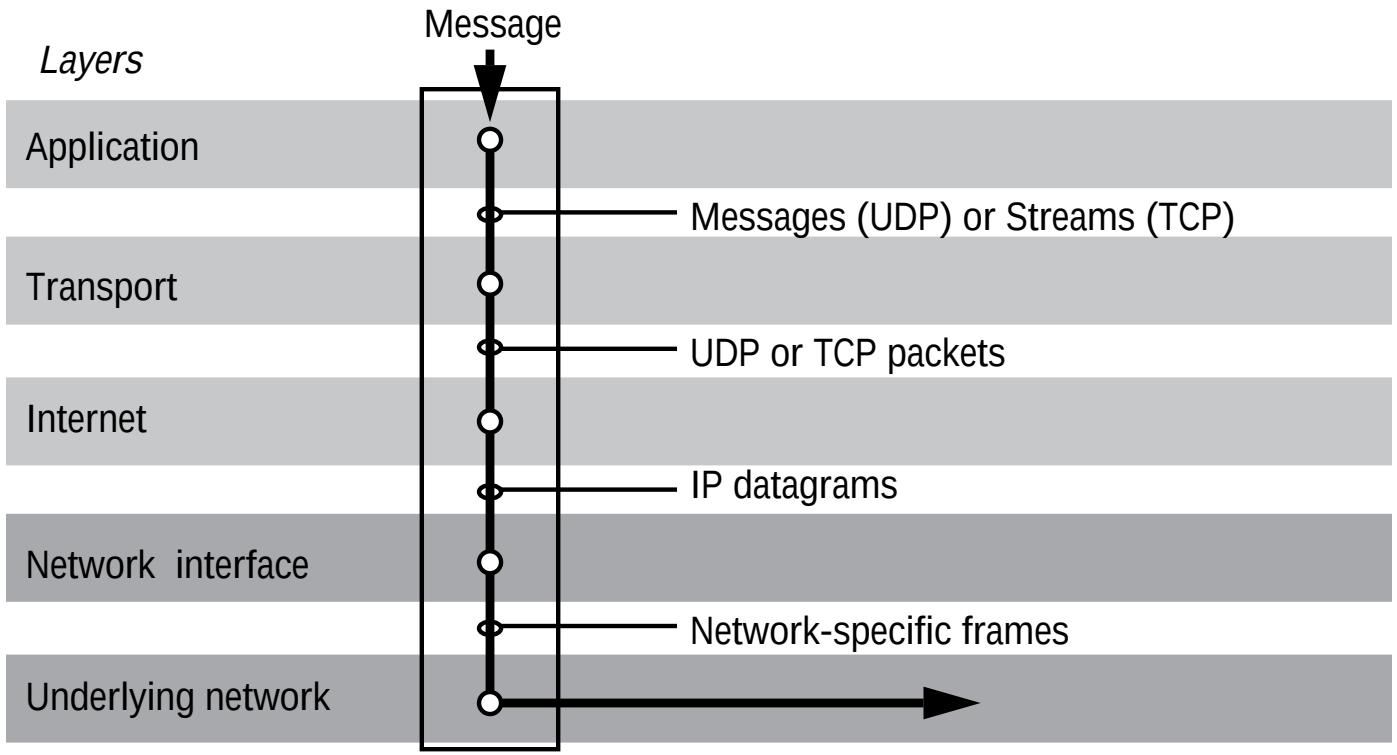


Figure 3.13 The programmer's conceptual view of a TCP/IP Internet.

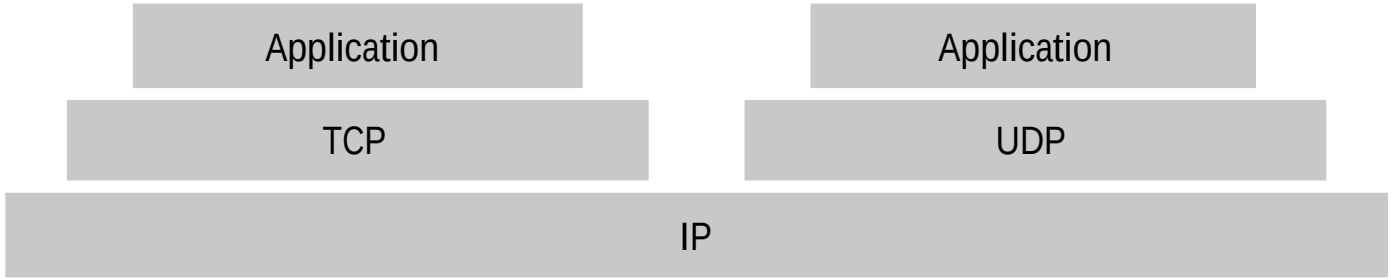


Figure 3.14 Internet address structure.

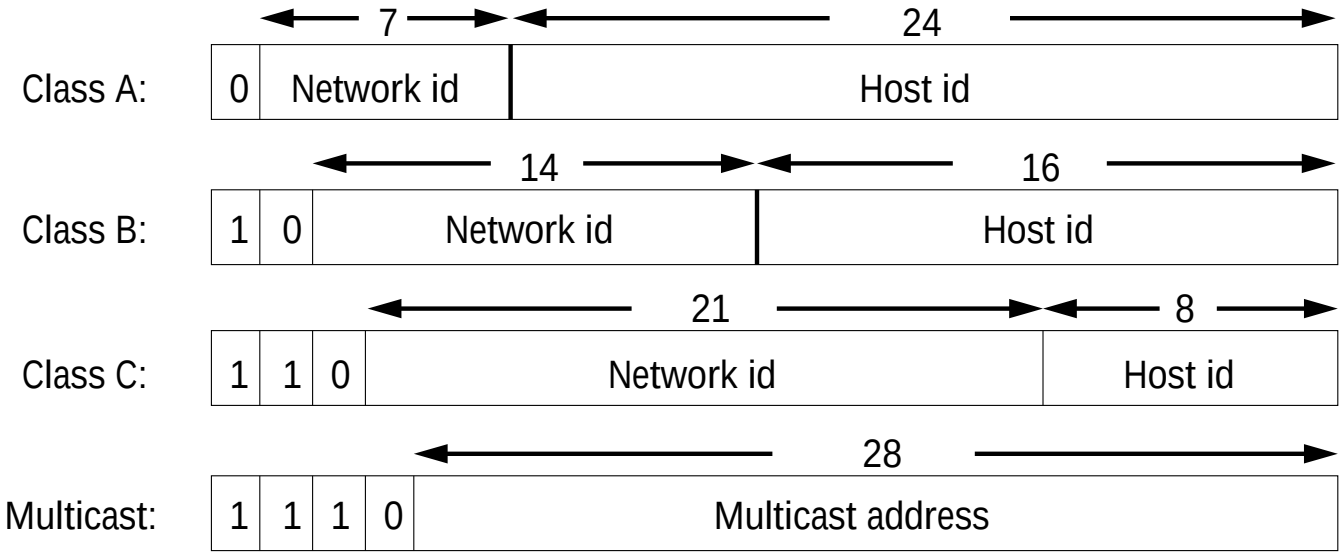


Figure 3.15 Decimal representation of Internet Addresses.

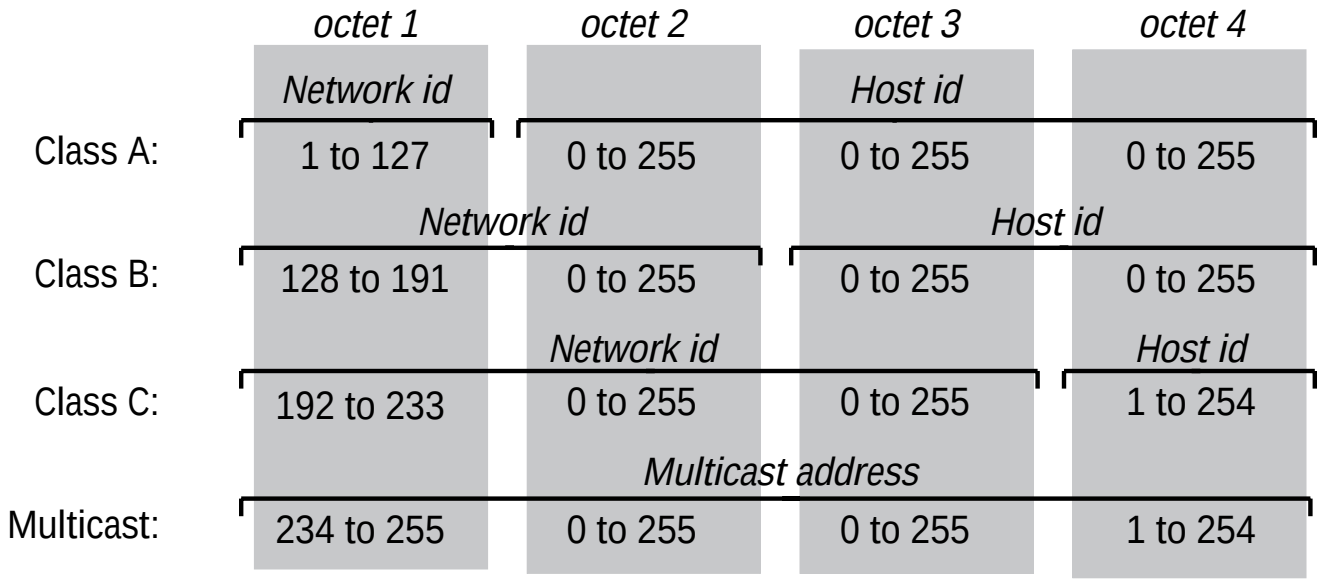


Figure 3.16 Part of the QMW network

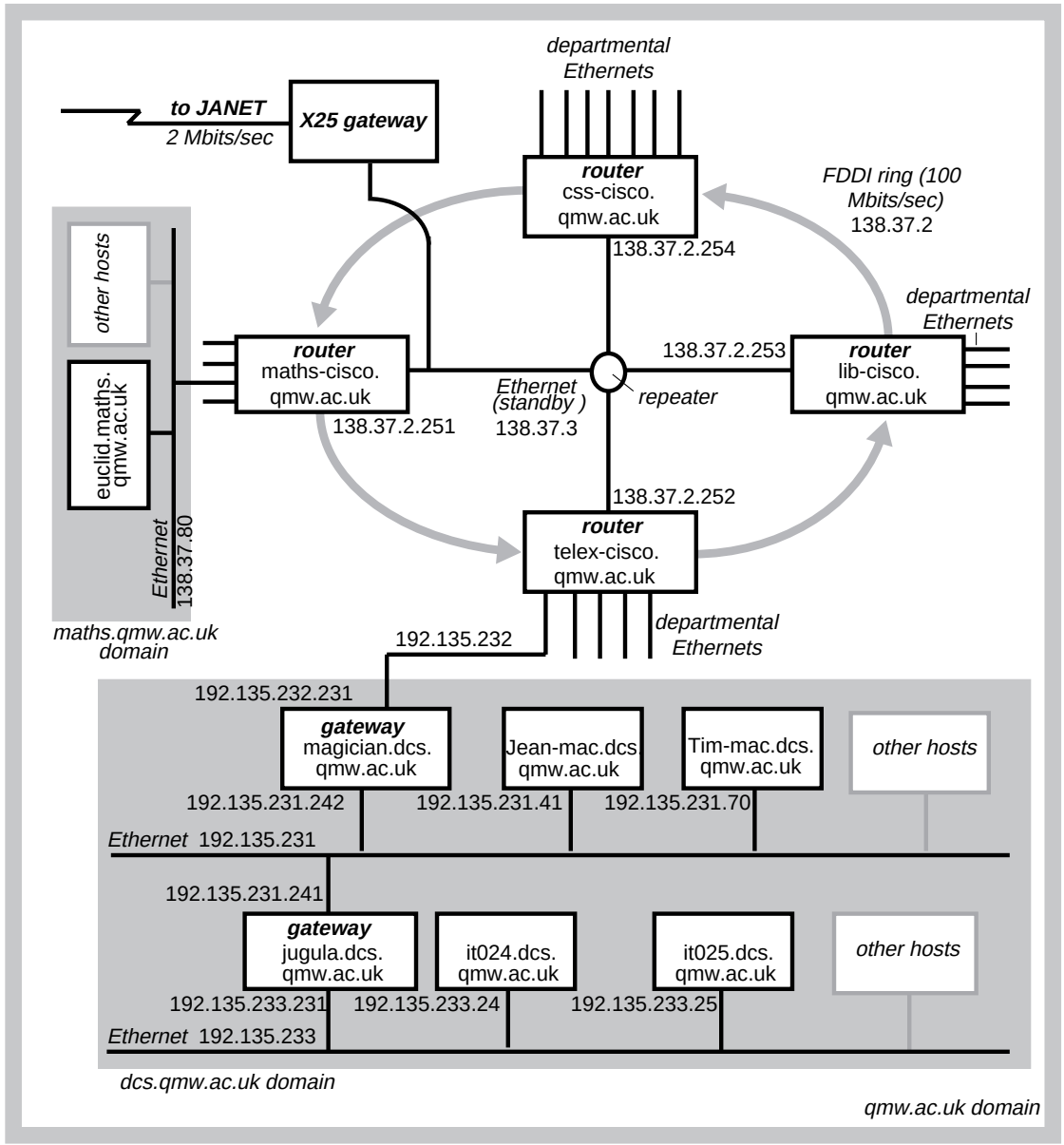


Figure 3.17 IP packet layout.

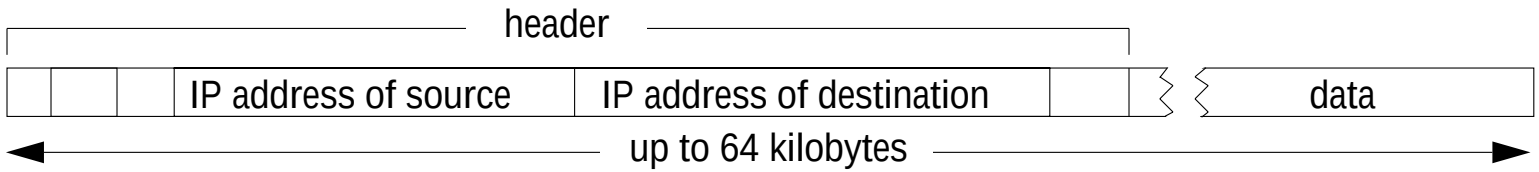
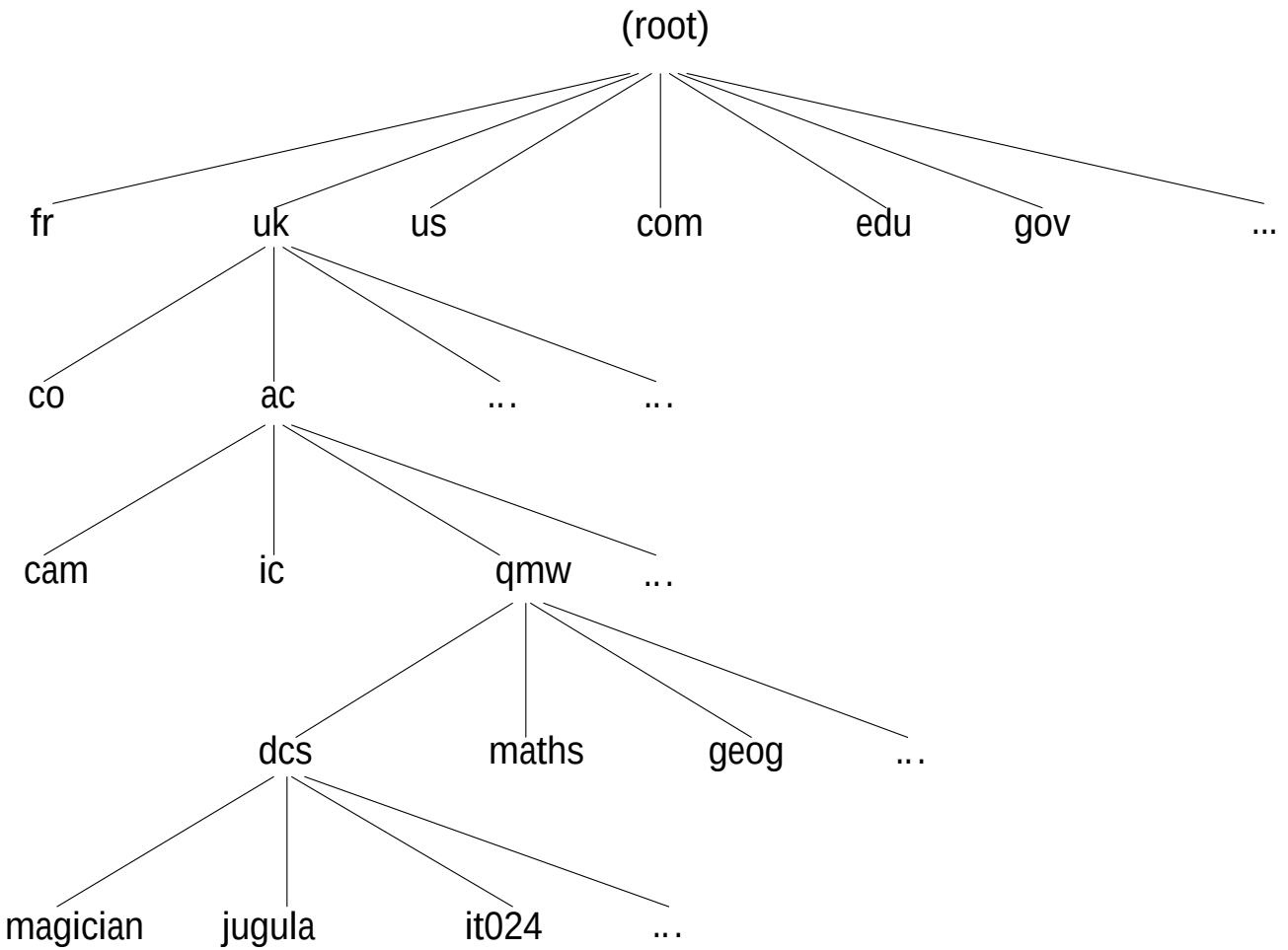
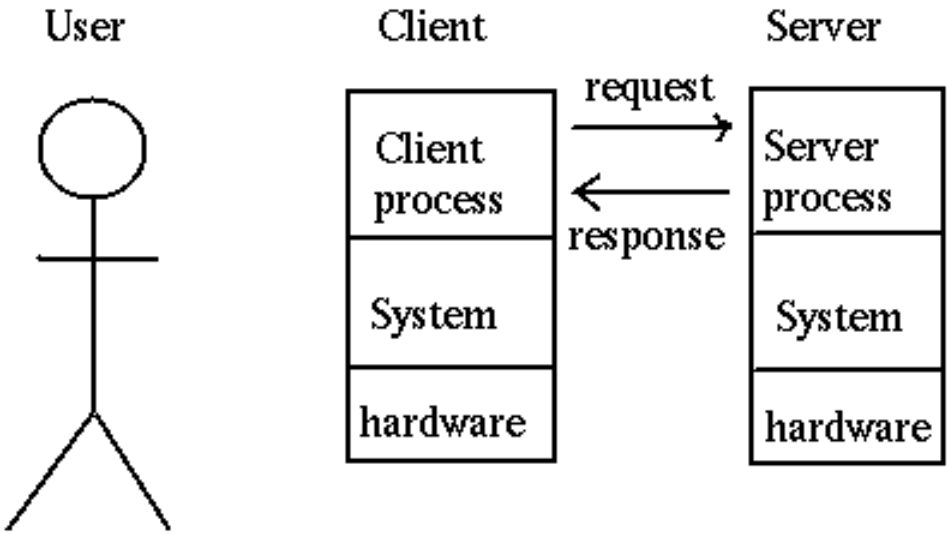


Figure 3.18 Domain name hierarchy.

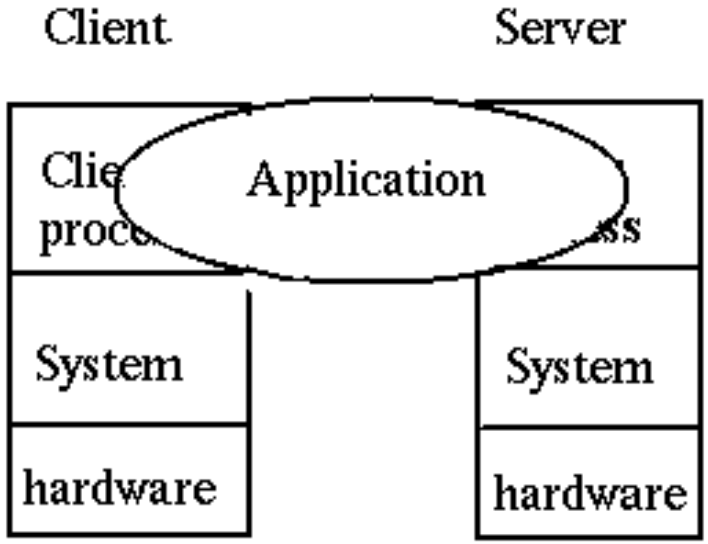


The Client-Server Model

Client Server Systems



Client/Server Application

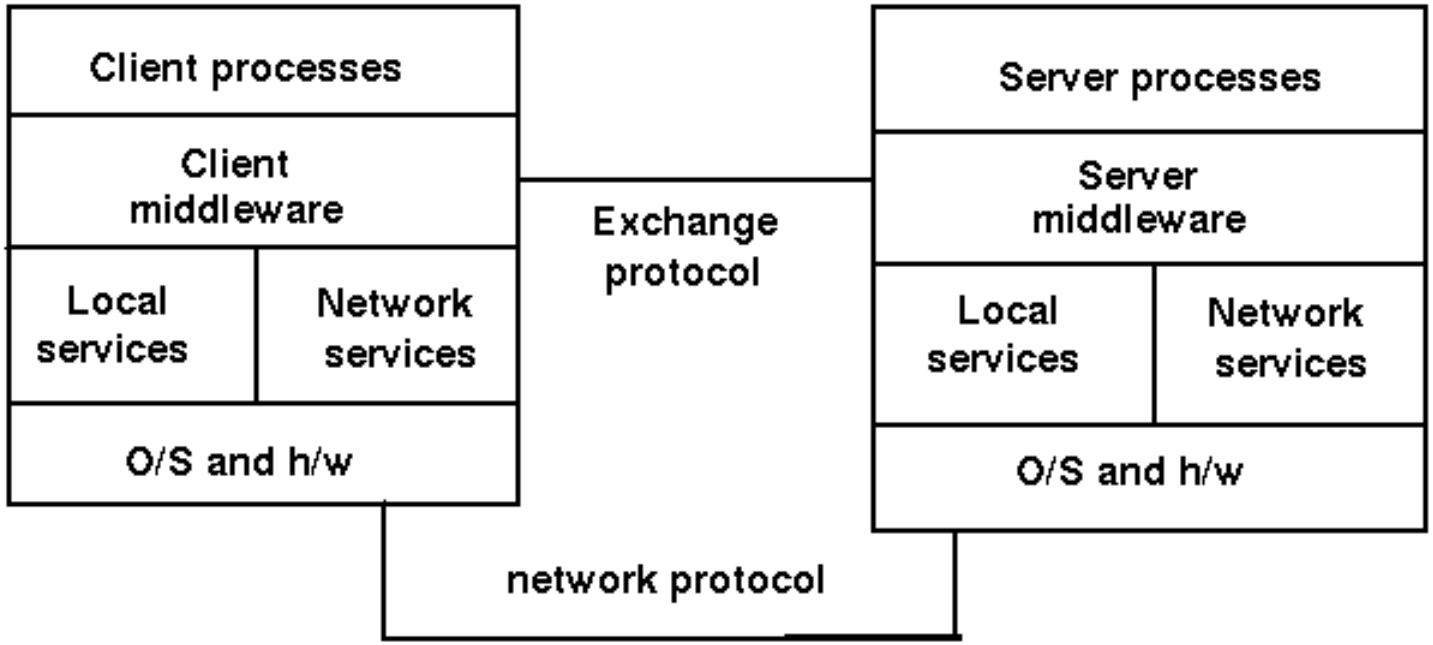


Overview

- Common communication patterns in a distributed applications
  - Client-server
  - Group (multicast) communication
- Client: process that requests service
- Server: process that provides service
- client usually *blocks* until server responds

- client usually invoked by end users when they require service
- server usually waits for incoming requests
- server can have many clients making concurrent requests
- server often a program with special system privileges

Middleware


Client processes
Client middleware
Local services
Network services
O/S and h/w
Exchange protocol
Server processes
Server middleware
Local services
Network services
O/S and h/w
network protocol

Client & Server Functions

- Clients
  - interacts with user through a user interface
  - performs application functions
  - interacts with *client middleware* using middleware API
  - recieves response and displays them if needed
- Servers
  - implement services
  - invoked by server middleware
  - provide error-recovery and failure-handling services

Client Middleware

- provide an API that can be invoked by clients
- establish connection with server middleware using network services
- send request to server middleware using an exchange protocol
- recieve response from the server middleware
- handle failure and synchronization of activity if needed
- handle access control

Server Middleware

- recieve client request from network services
- authenticate request
- schedule a server process to handle the client request
- receive response from the server process and send it back to the client middleware
- provide concurrency control
- handle failures on client side, network, servers, etc.

Middleware

Definitions:

- Middleware is a set of common business-unaware services that enable applications and end-users to interact with each other across a network
- distributed system services that have standard programming interfaces and protocols....services "sit in the middle" above OS and network software and below industry-specific applications
- the "/" in client/server applications
- software nobody wants to pay for

Examples

- ftp, email
- Web browsers
- Database drivers and gateways
- OSF's DCE (Distributed Computing Environment)
- OMG's CORBA (Common Object Request Broker Architecture)

Functional View of Middleware

- information exchange services
- application-specific services
  - specialized services, e.g., transactional services and replication services for distributed databases, groupware services for collaborative applications, specialized services for multimedia applications
  - *business-unaware*
- management and support services
  - needed for locating distributed resources and administering resources across the network

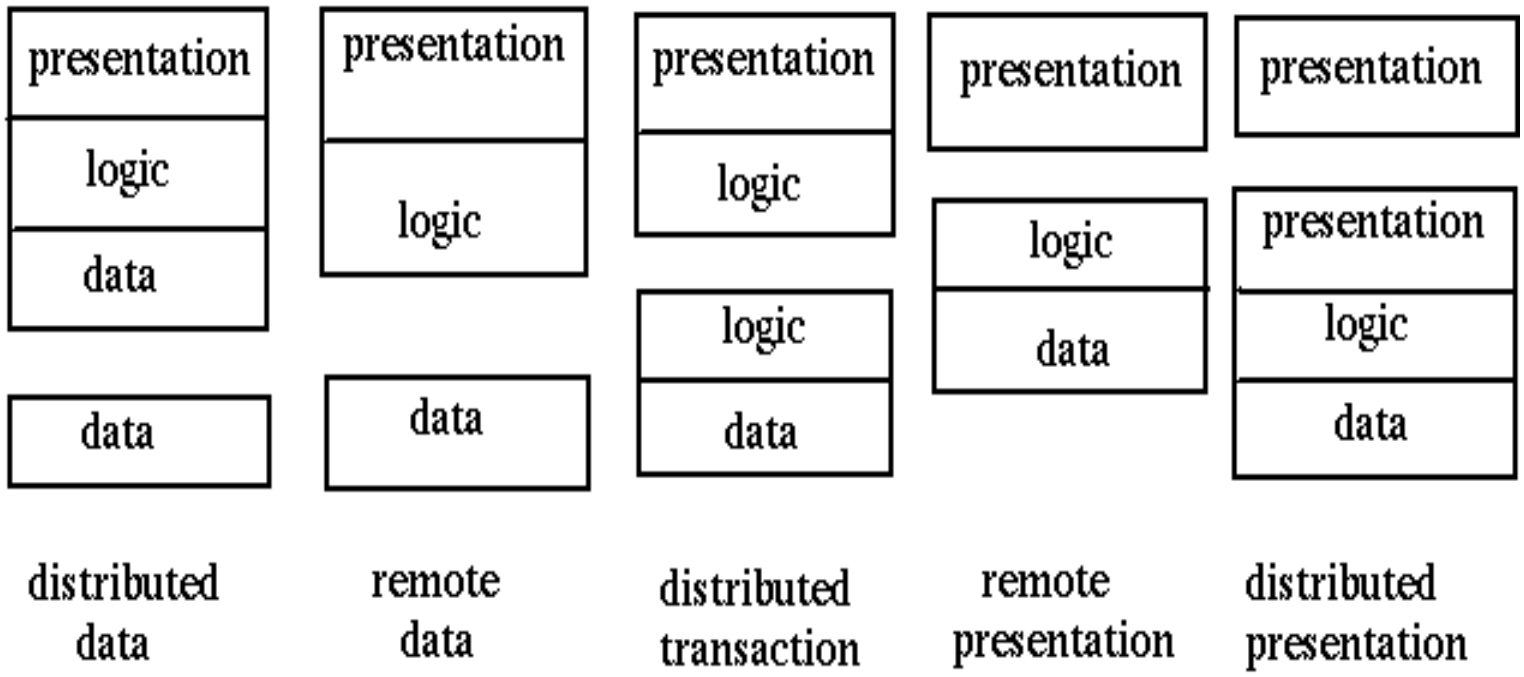
Commercial Middleware

- Middleware components that provide only one service
  - HTTP for retrieving remote documents, SUNRPC for RPC, etc.
- Middleware environments that combine many services
  - OSF DCE integrates RPC, security, directory, time, and file services
  - CORBA
- Compound middleware environments that combine many middleware environments into a single framework
  - IBM's Open Distributed Computing (ODC) Blueprint combines DCE, CORBA, and transaction management

Application Software Architectures

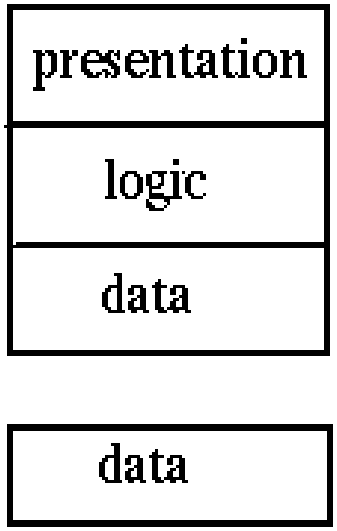
- many applications can be considered to be made up of three software components or logical tiers
  - user interface
  - processing layer
  - data layer
- Client/Server architectures
  - single-physical-tiered, two-physical-tiered, three-physical-tiered

“Gartner Group” Configurations



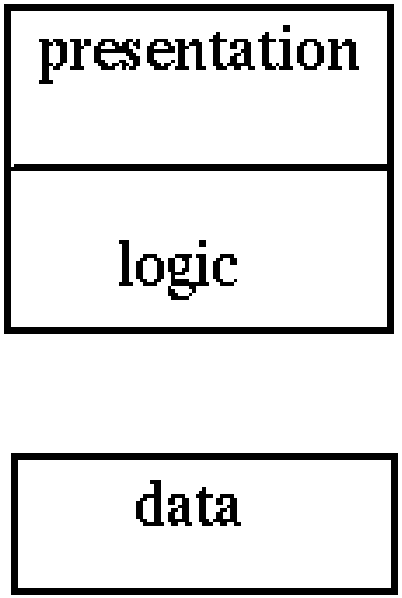
Distributed Data

Example: Distributed Database



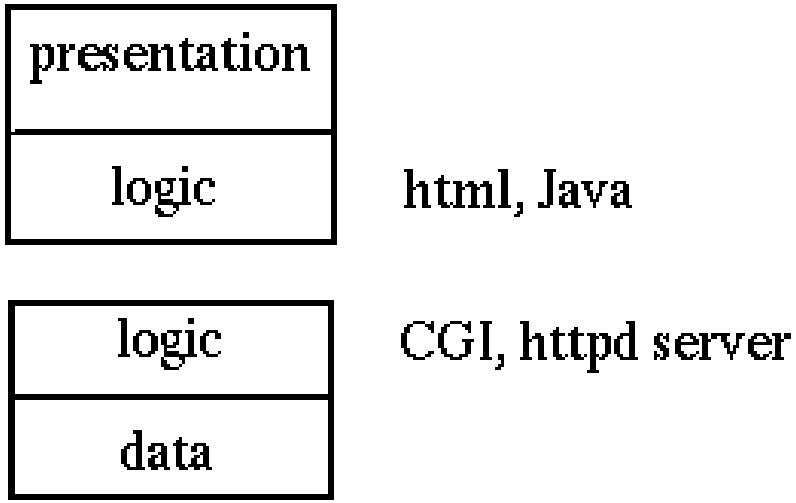
Remote Data

Example: Network File Systems



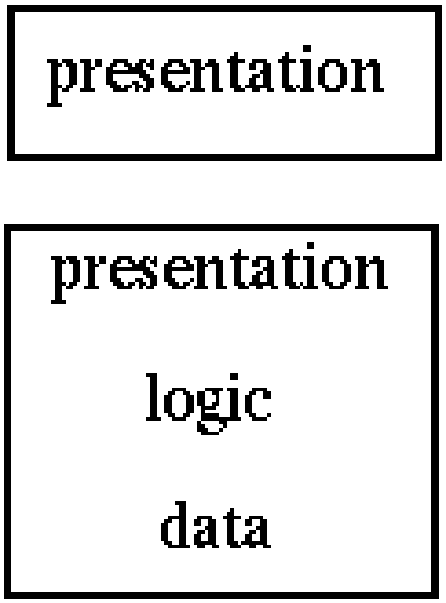
Distributed Programs

Example: World Wide Web



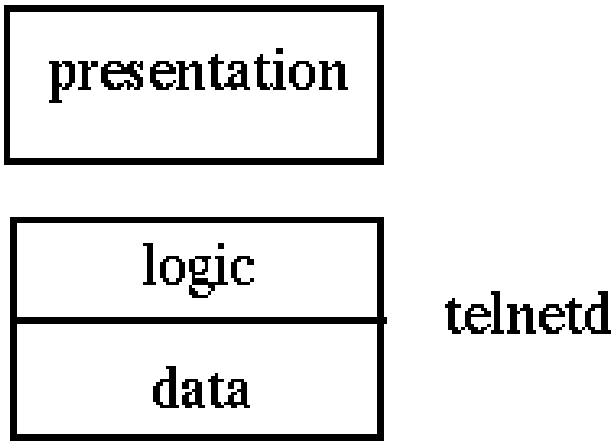
Distributed Presentation

Example: X Windows

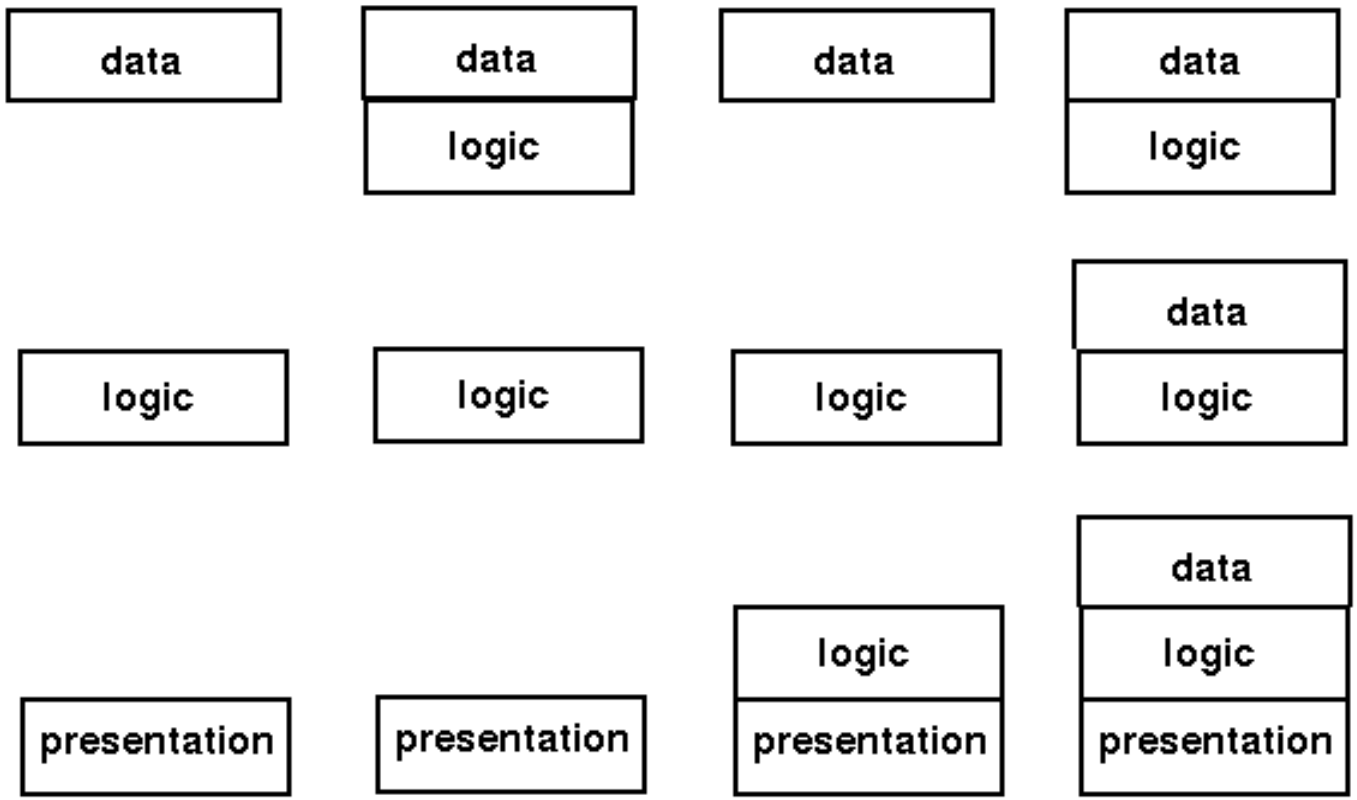


Remote Presentation

Example: telnet



Three-tier architectures



Motivation for multi-tier architectures

- Frees clients from dependencies on the exact implementation of the database
- It allows "business logic" to be concentrated in one place
  - Software updates are restricted to middle layer
- Performance improvements possible by batching requests from many clients to the database
- Database and business logic tiers could be implemented by multiple servers for scalability

Fat vs thin clients

- Thin client = network computer
  - Typically no local storage
- Fat client = typical desktop PC, workstation
- Motivation for thin clients: hidden costs of system administration and support
  - Network computers a move towards *centralized* system admin but *local* processing at client
  - Java (mobile code) an enabling technology
- Degrees of "thinness", e.g. PDAs