Networking history

1969 ARPANET - 56Kb dedicated lines, IMPs

1970's DECNET, XNS, SNA - proprietary designs

1977 - 1983 Ethernet (3 and 10 MHz), token ring

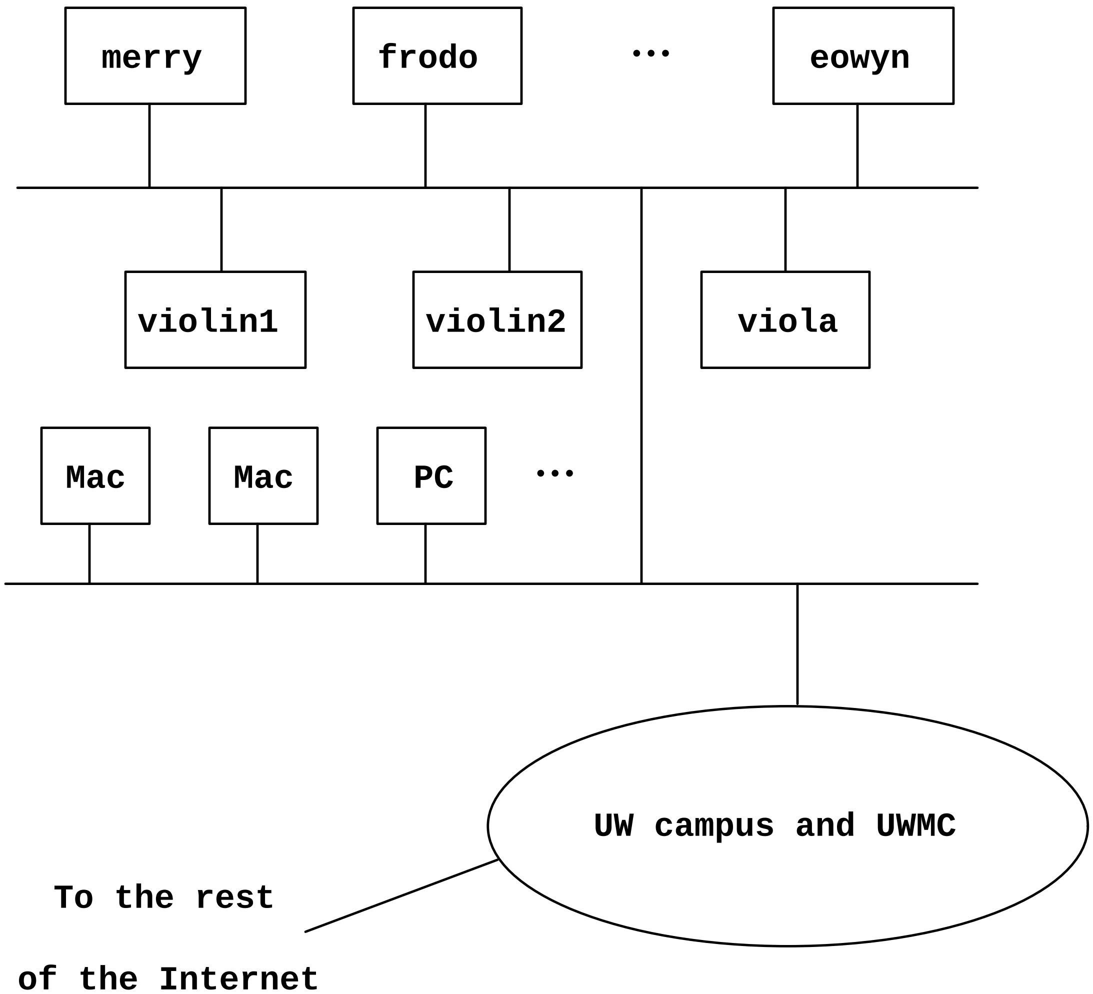
1981 BITnet - an IBM dinosaur (9600 baud)

1981 TCP/IP - supported local and wide area networks, the beginning of the Internet

1980's - present FDDI, 100 MHz Ethernet, gigabit networks

1980's - present Applications: FTP, Telnet, SMTP, X, HTTP, DICOM, and more

Department network



Clients and servers

- A *server* is a program running on a computer, waiting for connection requests.

- A *client* is a program running on a computer, that makes connection requests to a server.

A running program can act as both a client and server at the same time. For example, a file server waits for and accepts file access requests from clients, and it may make requests (as a client) to a time server.

Protocols

Clients and servers use *protocols* to communicate.
A protocol includes:

- a *format* for *messages*,
- a *specification* of *when* each message is sent.

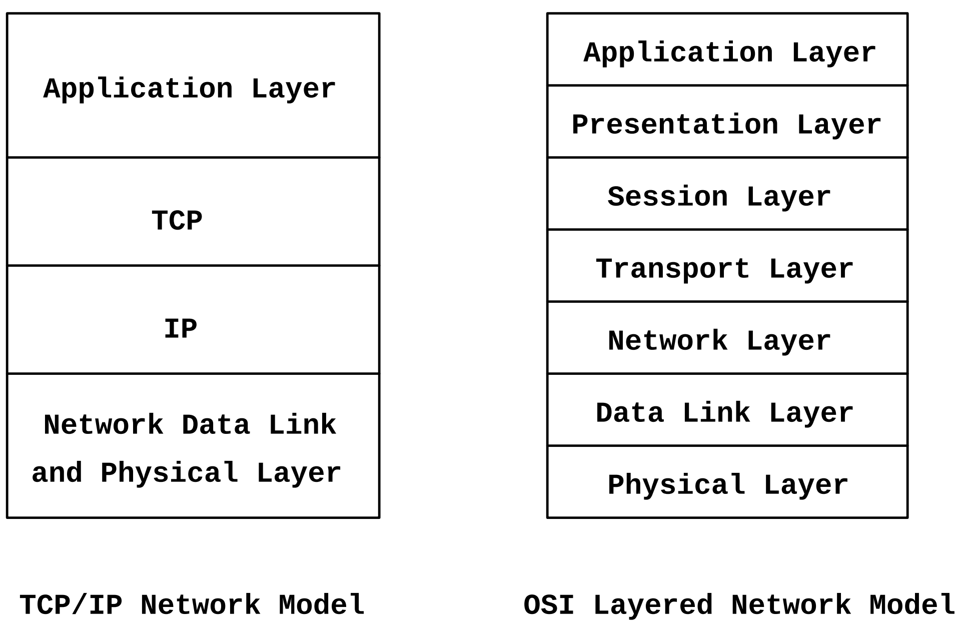
Examples of protocols:

- Ethernet - a hardware and data link protocol
- SLIP - another hardware and data link protocol
- IP, UDP, TCP - network layers
- SMTP - simple mail transfer protocol
- Telnet - a simple terminal emulator
- FTP - file transfer protocol
- X - a window display protocol
- HTTP - hypertext transfer protocol
- HL7 - a protocol for exchanging health data
- DICOM - medical image data exchange

The ISO Open Systems Interconnect (OSI) network model

OSI Layer	Content and function
Application Layer	User functions and programs
Presentation Layer	Encoded data - useable form
Session Layer	Logical link (system specific)
Transport Layer	Logical link (sys. independent)
Network Layer	Network addressing, routing
Data Link Layer	Error free data transmission
Physical Layer	Hardware connection

TCP/IP and the OSI model



The TCP/IP network layers

TCP Transmission Control Protocol, reliable stream transport, acts like a terminal I/O device - synchronous reads and writes, transparent to packet boundaries.

UDP User Datagram Protocol, a connectionless packet transport, is raw packet at a time transmission - the application needs to insure receipt and order.

IP Internet Protocol, handles packet delivery and routing, getting packets from place to place.

- A *router* accepts packets from one network and forwards them to the next network or to their destination.
- A *gateway address* or route is an IP address to use for destinations on a given network.
- A *default route* is used for any address that is not local and that has no more specific information.

Packet formats

An ethernet packet:

Preamble	Destination Address	Source Address	Frame Type	Frame Data	CRC
64 bits	48 bits	48 bits	16 bits	368-12000 bits	32 bits

An IP packet:

DATAGRAM HEADER	DATAGRAM DATA AREA

The header contains information like: total length, ID, protocol, source IP address, destination IP address.

A UDP packet:

UDP SOURCE PORT	UDP DESTINATION PORT
UDP MESSAGE LENGTH	UDP CHECKSUM
DATA	
...	

A TCP packet:

SOURCE PORT		DESTINATION PORT	
SEQUENCE NUMBER			
ACKNOWLEDGEMENT NUMBER			
HLEN	RES.	CODE BITS	WINDOW
CHECKSUM		URGENT POINTER	
OPTIONS (IF ANY)			PADDING
DATA			
...			

The socket abstraction

A socket is analogous to a UNIX file. It is a source for reading bytes and a sink for writing bytes.

Instead of a file name, a socket is associated with a *port* specified by a number.

The analog of "open" is to assign a port to a socket, and either (as a server) wait for an incoming connection, or (as a client) request a connection to a remote port.

Some well known ports and TCP services associated with them:

Port	Keyword	Description
7	ECHO	Sends a copy of data
21	FTP	File transfer
23	TELNET	Remote terminal
25	SMTP	Simple Mail Transport
79	FINGER	User information
80	HTTP	Web server
104	DICOM	Medical image transfer

Types of protocols

There are several dimensions of protocol (and server) design:

- Connectionless vs. connection-oriented:

  connectionless transport (UDP) is more efficient, suitable for short messages, but unreliable;

  connection-oriented transport (TCP) provides reliable connections but consumes more resources.

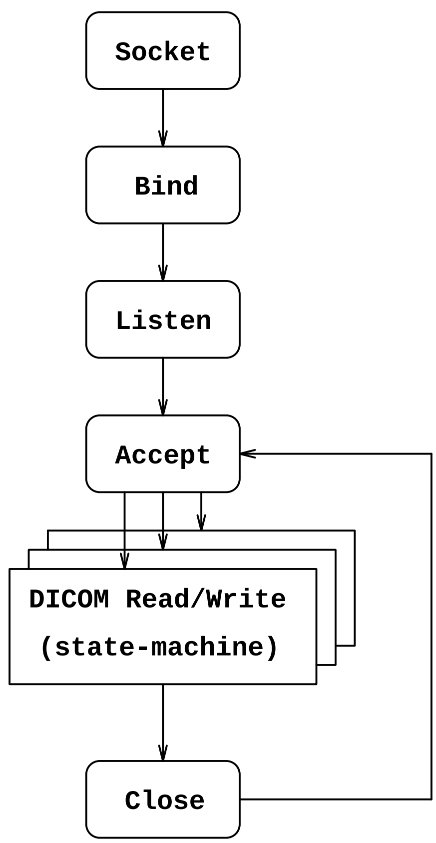
- Stateless vs. stateful:

  stateless protocols serve well for information retrieval (HTTP);

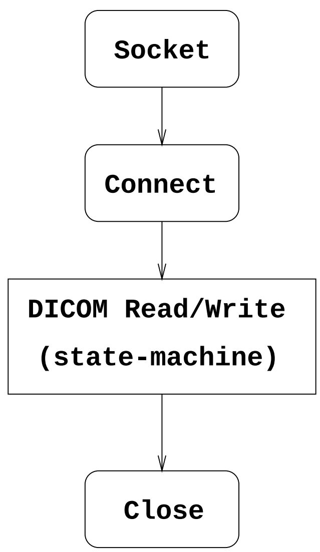
  stateful protocols are necessary for complex transactions (SMTP).

- Iterative vs. concurrent: mainly an issue of server design rather than protocol design.

DICOM: a typical TCP/IP network server design



DICOM: a typical TCP/IP network client design



The Socket Interface

socket creates a socket and returns a descriptor:

```
retcode = socket ( family, type, protocol );
```

connect specifies the remote endpoint (and establish, for TCP)

```
retcode = connect ( socket, addr, addrlen );
```

close terminates a connection gracefully and removes the socket

```
retcode = close ( socket );
```

bind specifies a local IP address and port number

```
retcode = bind ( socket, localaddr, addrlen );
```

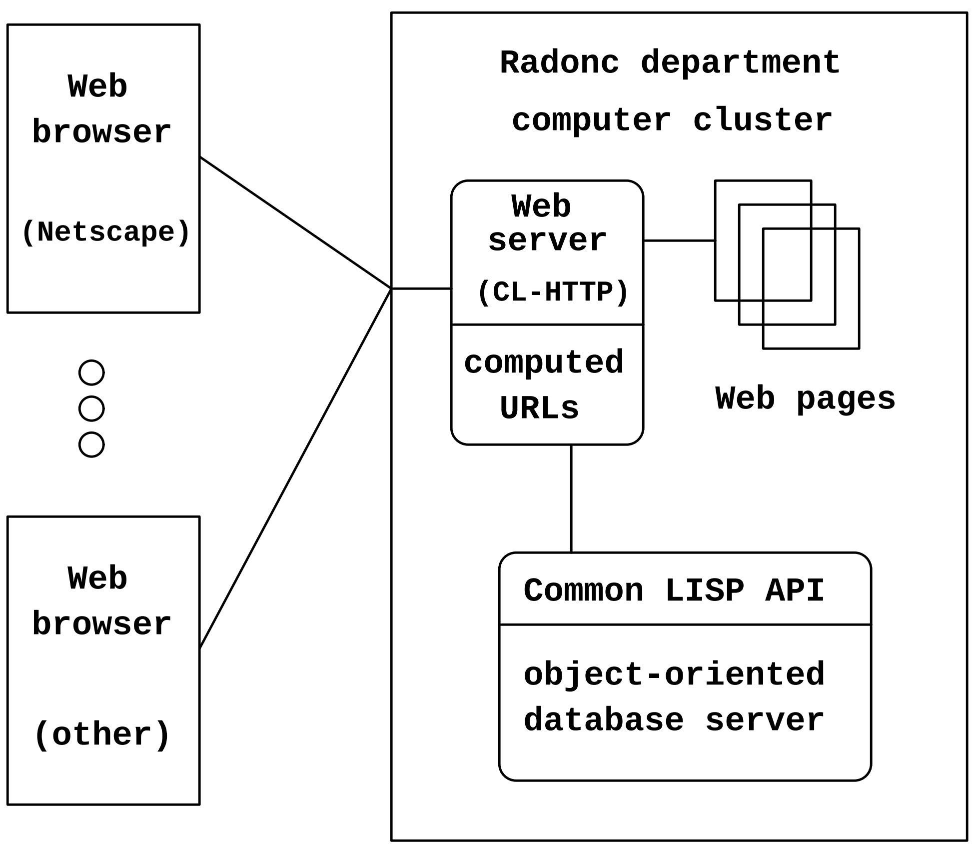
listen used by servers to make a socket passive

```
retcode = listen ( socket, queuelen );
```

accept used by servers to accept the next incoming connection

```
retcode = accept ( socket, addr, addrlen );
```

Web-based database access



Network addresses

Format of an IP address: `a.b.c.d`

Subdivision of the 32 bit address space:

Network class	range of `a`	network address	host address	max hosts per network
A	0 to 127	a	b.c.d	16,777,214
B	128 to 191	a.b	c.d	65,534
C	192 to 223	a.b.c	d	254
Multicast	224 to 239	N/A	N/A	268,435,456
Reserved	240 to 255	N/A	N/A	–

The Internet Domain Name System (DNS)

Names are more *abstract* than addresses.

Format of an Internet Domain name:

host.domain `aol.com`, `nsf.org`

host.subdomain.domain `castor.wustl.edu`

host.dept.subdomain.domain `godot.radonc.unc.edu`

... no limit on the length of these names.

Aliases can hide details that may change, e.g.,

`radonc.washington.edu` for addressing mail, and

`www.radonc.washington.edu` for access to a department web server.

Name to address translation

The `/etc/hosts` file provides static name to address translation:

```
# The form for each entry is:
# <internet address><official hostname> <aliases>
#
128.95.120.1 dns1.cac.washington.edu  dns1
128.95.181.100 default
128.95.181.182 tuba.radonc.washington.edu tuba
128.95.181.154 flute.radonc.washington.edu flute
128.95.181.51 cello.radonc.washington.edu cello
128.95.181.50 viola.radonc.washington.edu viola
128.95.181.49 violin2.radonc.washington.edu violin2
128.95.181.48 violin1.radonc.washington.edu violin1
128.95.181.52 bass.radonc.washington.edu bass
128.95.181.153 oboe.radonc.washington.edu  oboe
127.0.0.1 localhost loopback
```

Searching for abbreviated names

The name resolver tries specified domains and subdomains, using information in `resolv.conf`:

```
search radonc.washington.edu washington.edu
nameserver 128.95.120.1     # dns1.cac.washington.edu
nameserver 128.95.112.1     # dns2.cac.washington.edu
```

The `nslookup` command asks the name server for information:

```
ira@oboe% nslookup www.nlm.nih.gov
Name Server:  dns1.cac.washington.edu
Address:  128.95.120.1

Non-authoritative answer:
Name:    public.nlm.nih.gov
Address:  130.14.74.3
Aliases:  www.nlm.nih.gov

ira@oboe% nslookup sleepy.vamc
Name Server:  dns1.cac.washington.edu
Address:  128.95.120.1

Name:    sleepy.vamc.washington.edu
Address:  205.175.96.197
```

References for network programming and administration

- Comer and Stevens, "Internetworking with TCP/IP, Vol III: Client-server programming and applications", Prentice-Hall 1997.

- Cheswick and Bellovin, "Firewalls and Internet Security: Repelling the Wily Hacker", Addison-Wesley 1994.