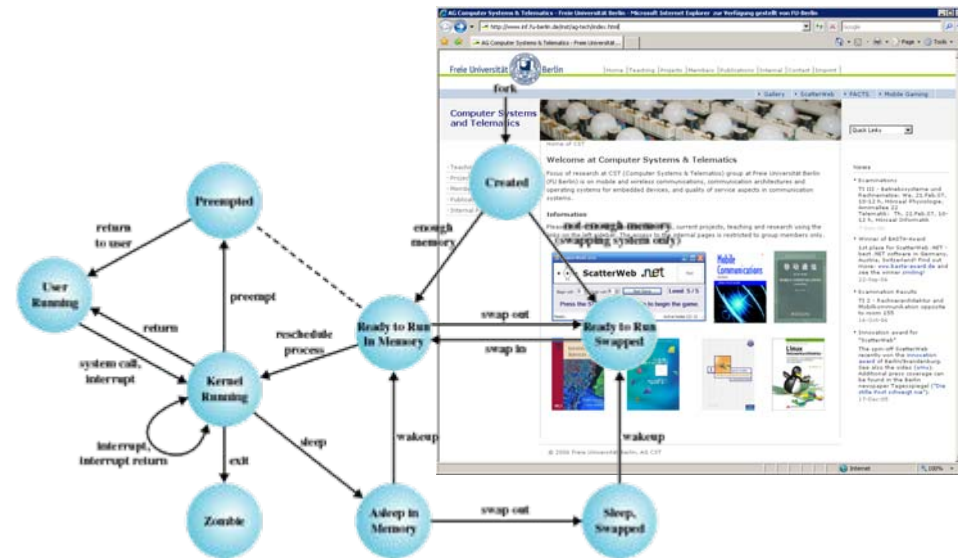


TI III: Operating & Communication Systems Example

Under the Hood of Surfing the Web



Content (3)

14. Security

- Basic concepts & terms
- Cryptology
- Examples
 - Firewalls
 - Virtual Private Networks (VPNs)
 - IP Security
 - Email security with PGP

15. Application

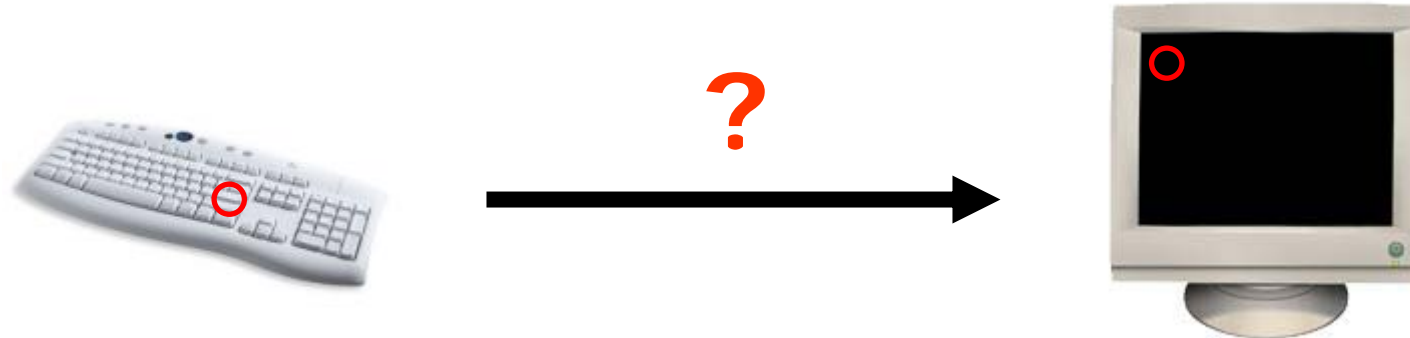
- Domain Name System
- Email
- World Wide Web

16. Example

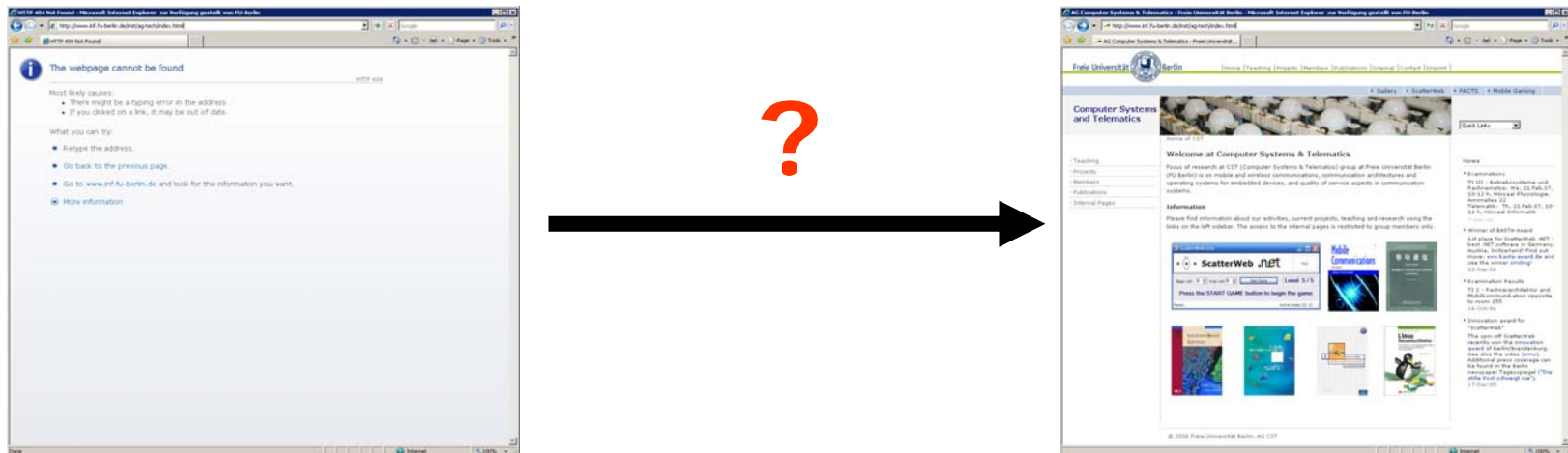
- **Under the Hood of Surfing the Web**

A Comprehensive Example

- What happens if one presses a key on the computer?



- What if that key causes an web page to be displayed?



Keyboard Interrupt

- Keyboard controller raises interrupt flag
- CPU interrupts execution of current process and starts Interrupt Service Routine (ISR)
 - unconditional jump

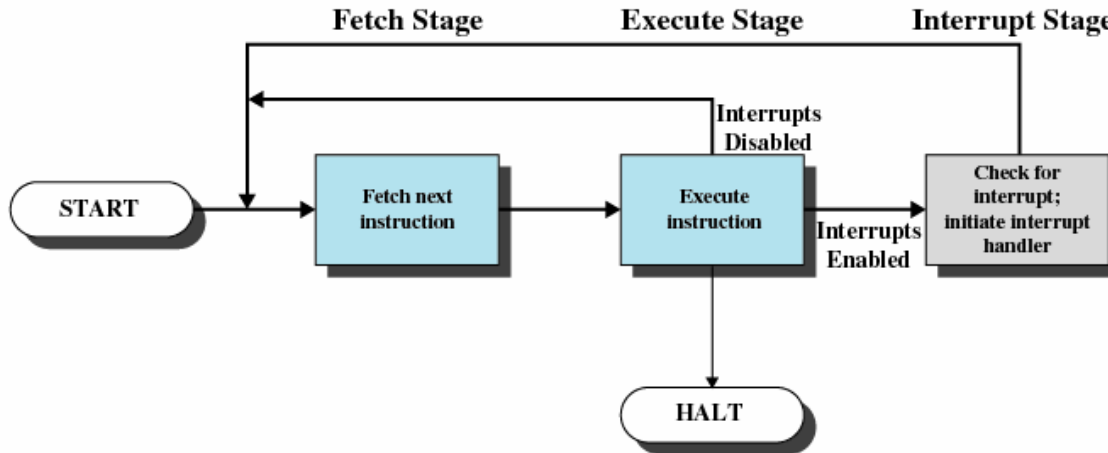
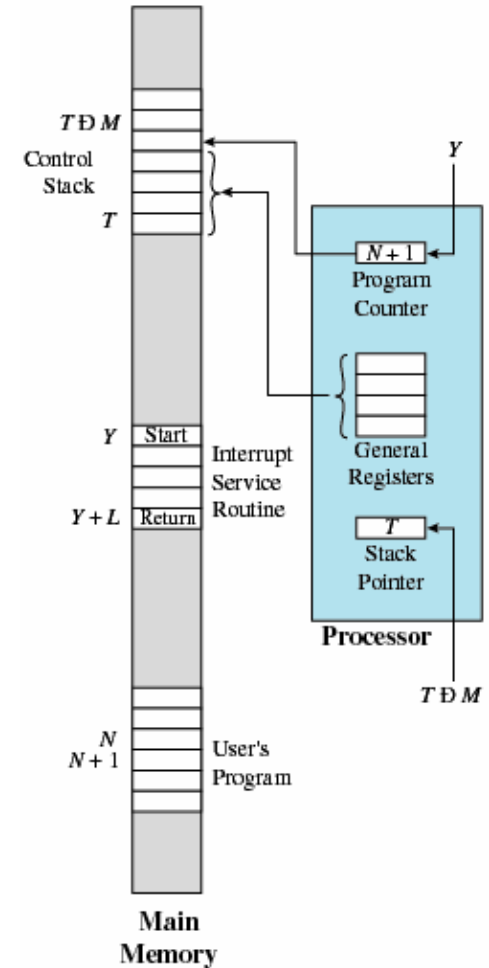
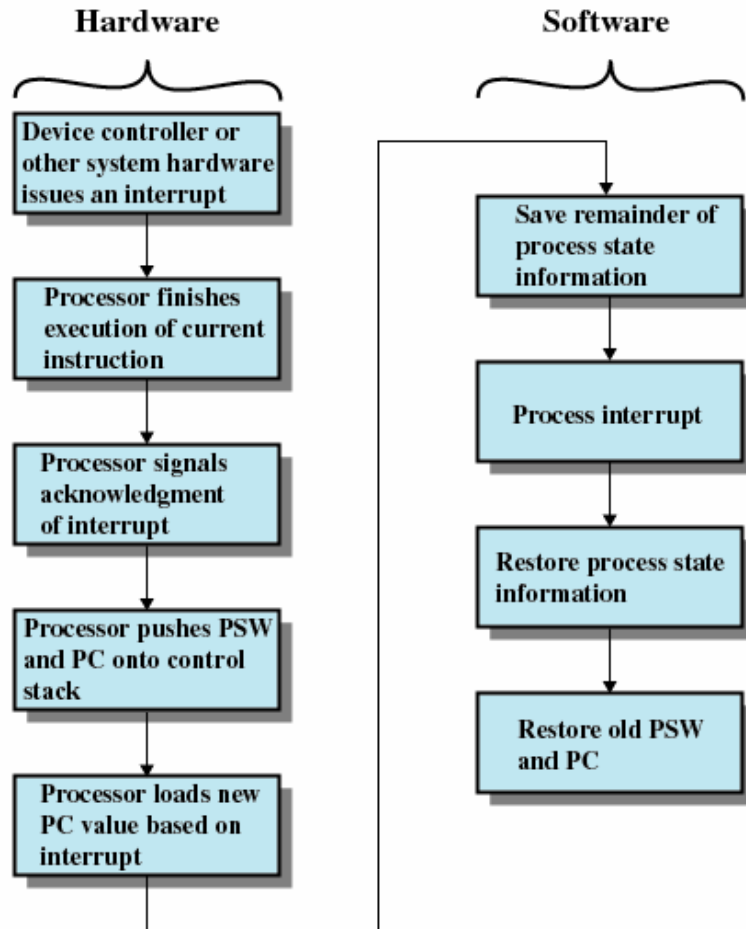


Figure 1.7 Instruction Cycle with Interrupts



(a) Interrupt occurs after instruction at location N



- ISR processes input from keyboard
 1. Clears interrupt flag
 2. Transfers data from device into buffer
 3. Establishes owner of device
 4. Triggers notification of user process

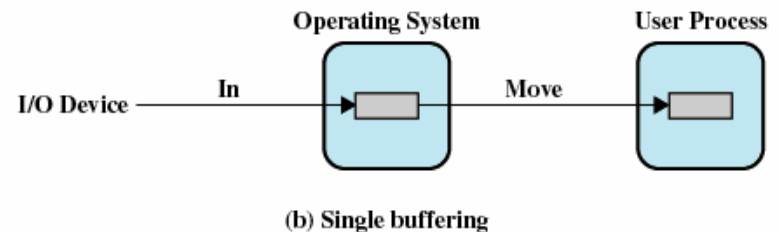


Figure 1.10 Simple Interrupt Processing

In the meantime...

- Web browser is one of many processes running locally
- Other processes include
 - Other user processes (possibly of different users)
 - System processes implementing system services
 - Kernel processes

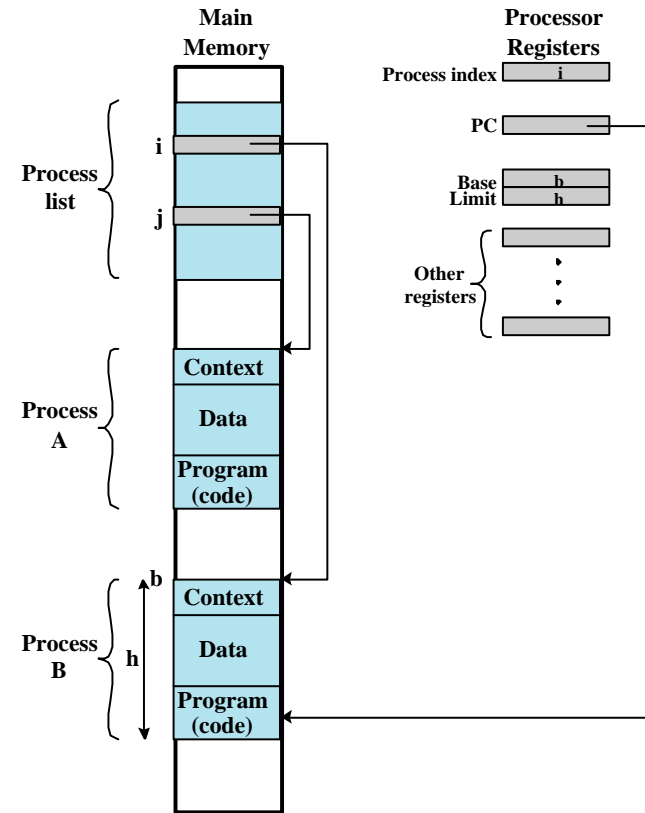
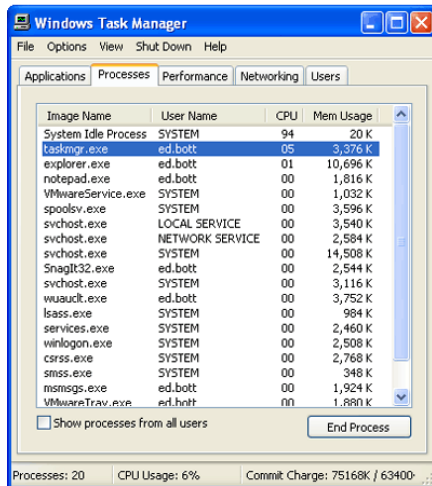
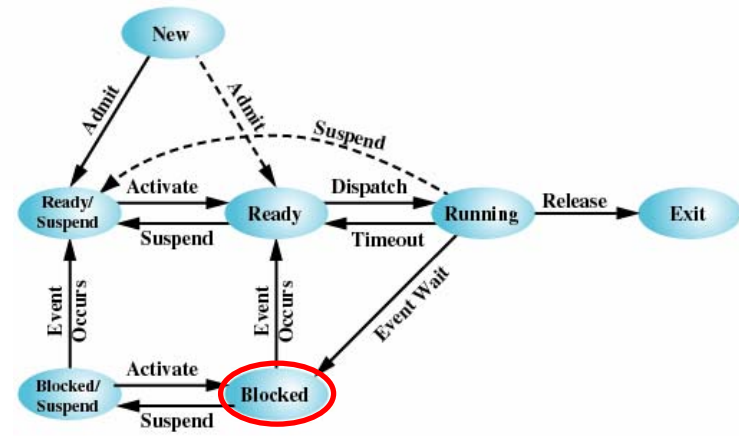
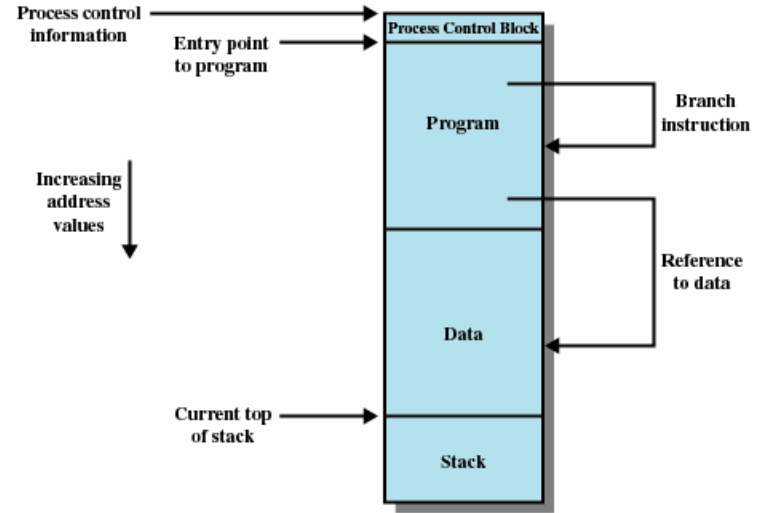
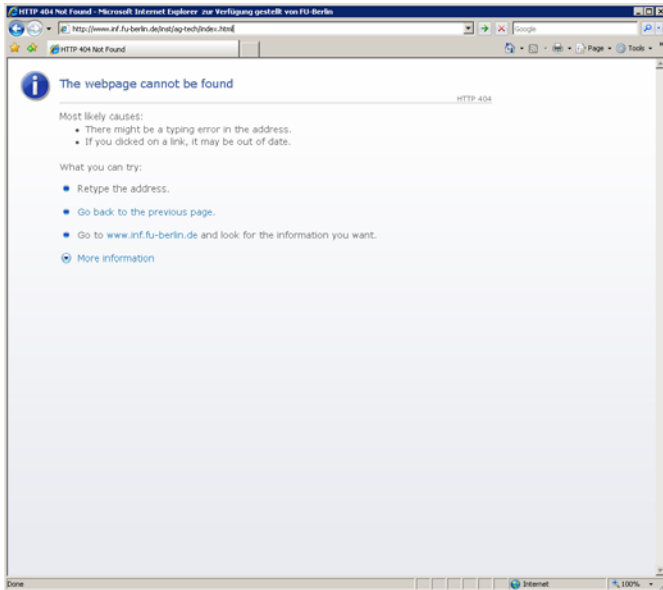


Figure 2.8 Typical Process Implementation

Web Browser Process in Detail

- Web browser processes
 - Currently waiting for input
 - e.g. using `select()`
 - Process state **blocked**

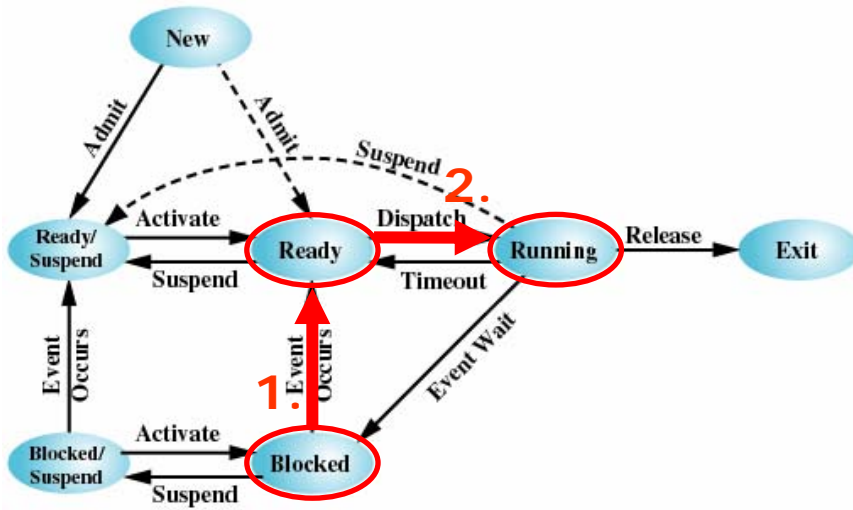


(b) With Two Suspend States

Figure 3.9 Process State Transition Diagram with Suspend States

Reaction to External Event

1. ISR changes process state to **ready**
2. Scheduling algorithm eventually changes process state to **running**



(b) With Two Suspend States

Figure 3.9 Process State Transition Diagram with Suspend States

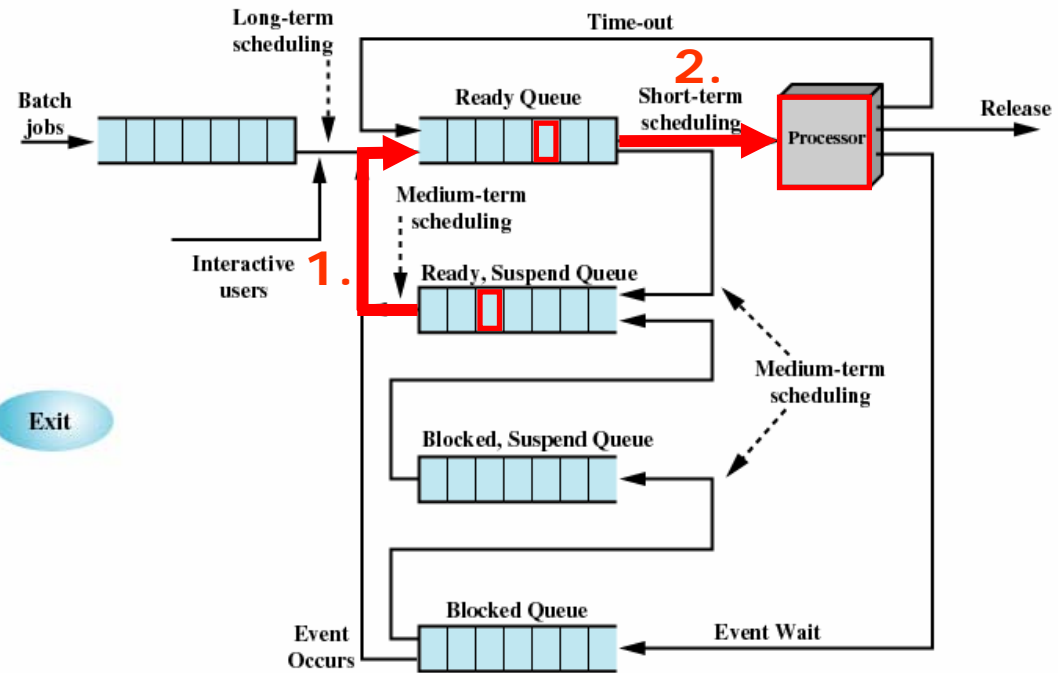
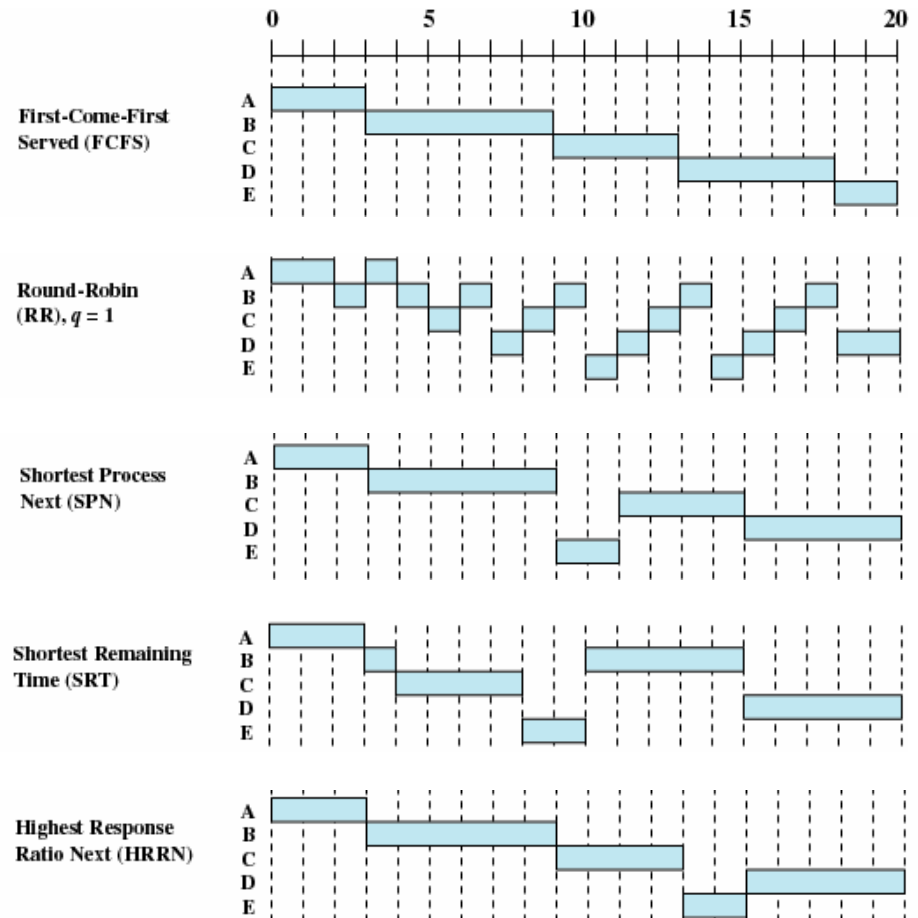


Figure 9.3 Queuing Diagram for Scheduling

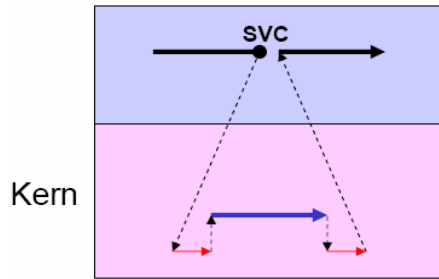
- Scheduling is handled by variety of scheduling algorithms
 - Non-preemptive / preemptive
 - Maximize throughput, responsiveness, etc...
- Processes may have priorities
 - Priority inversion due to lock on shared resources
 - Priority inheritance



Web Browser Processes Event

- Assume input requires web browser to display a web page with a given URL

- String processing (user space)
- Connect to server and retrieve necessary data (system calls)



- Render web page (user space)
- Update user interface (system calls)

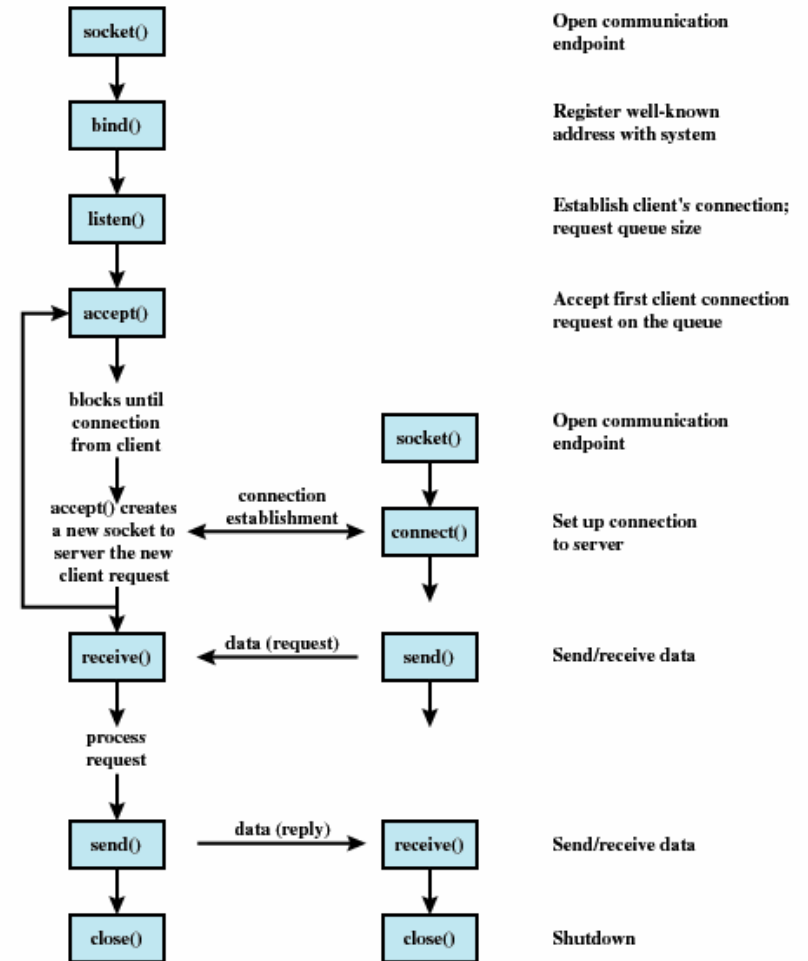
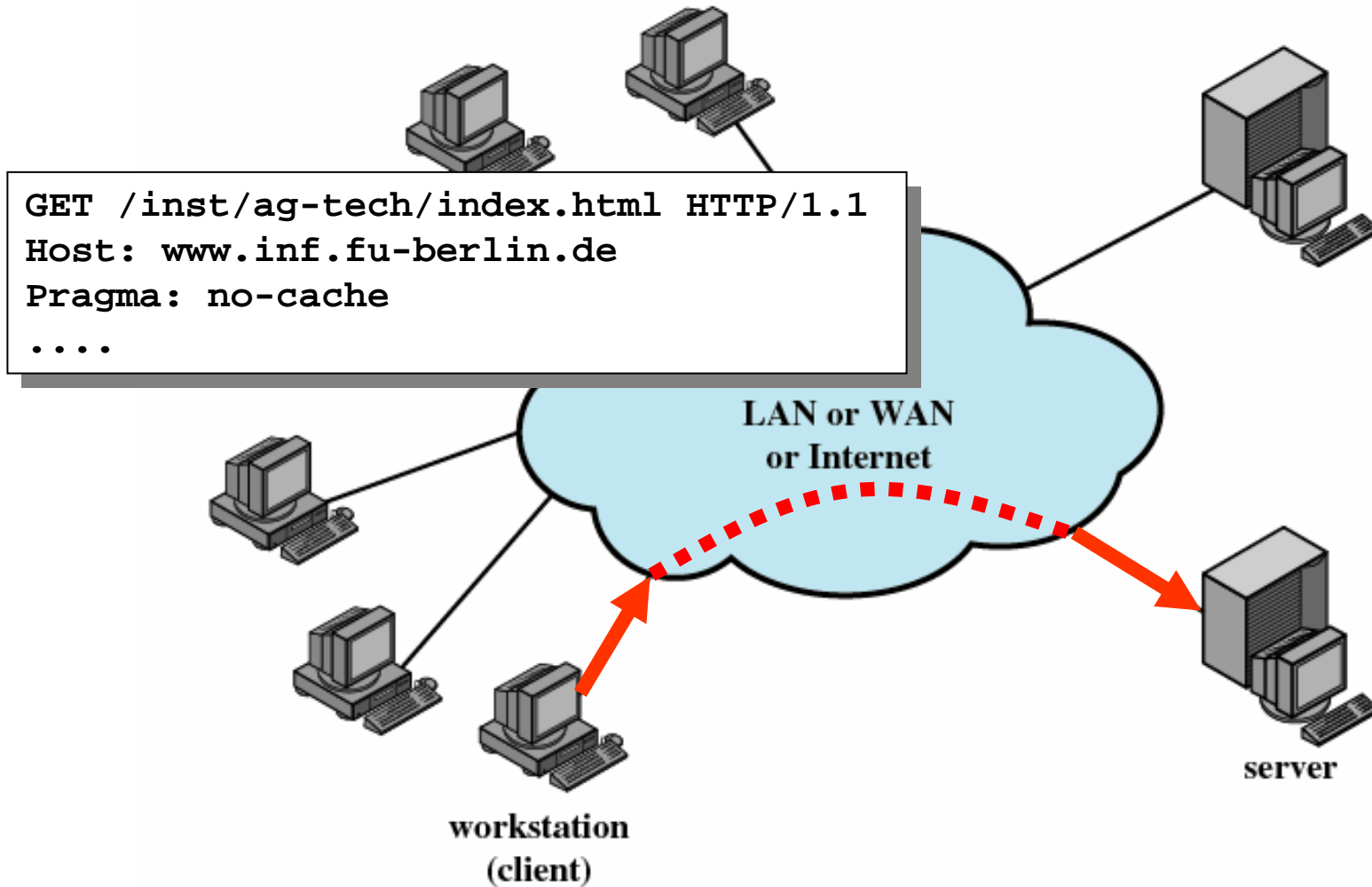


Figure 13.6 Socket System Calls for Connection-Oriented Protocol

Client/Server Communication



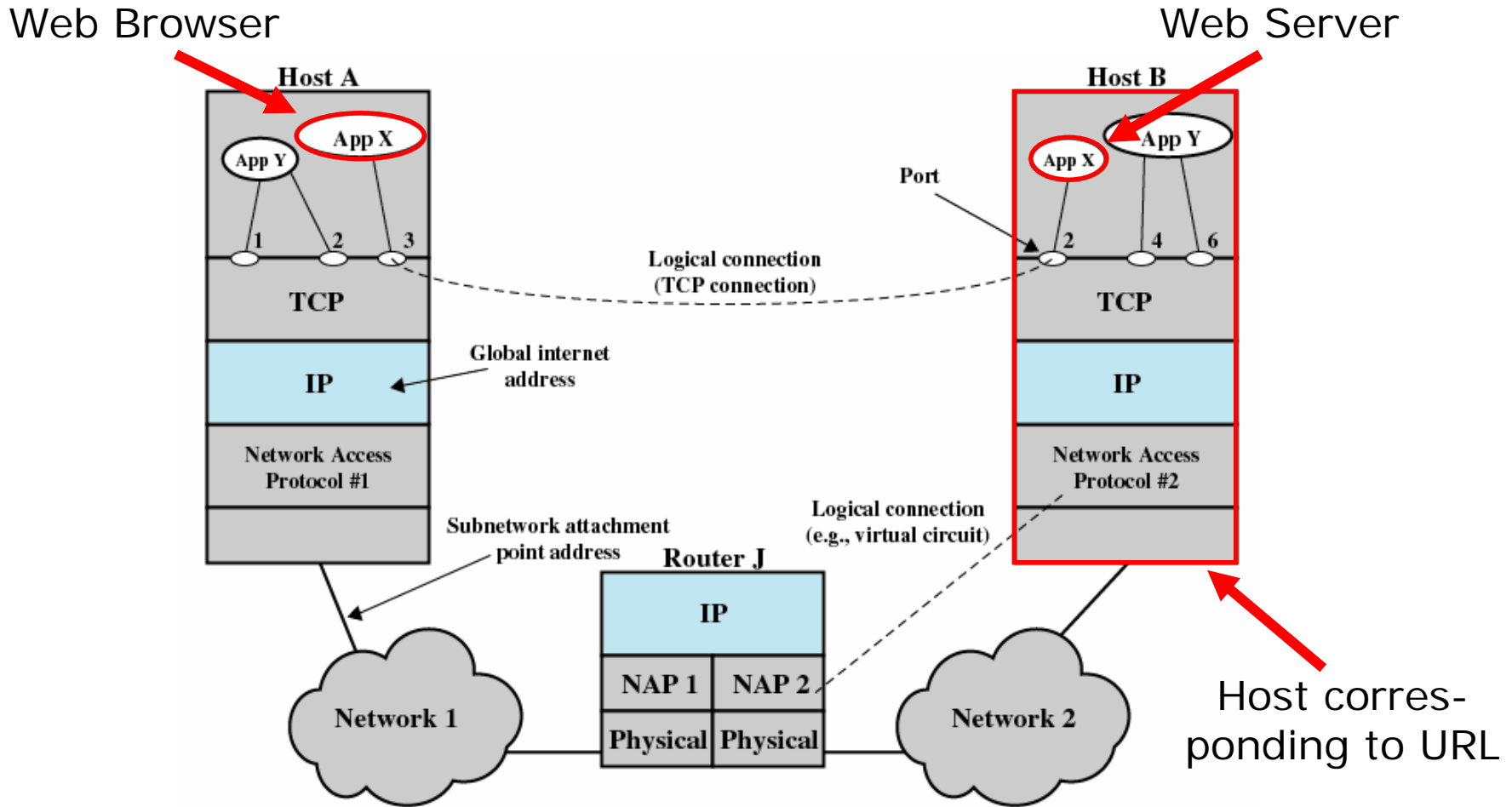
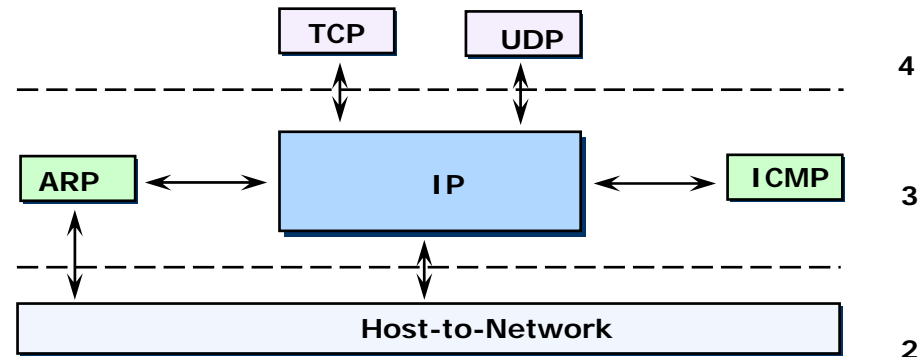


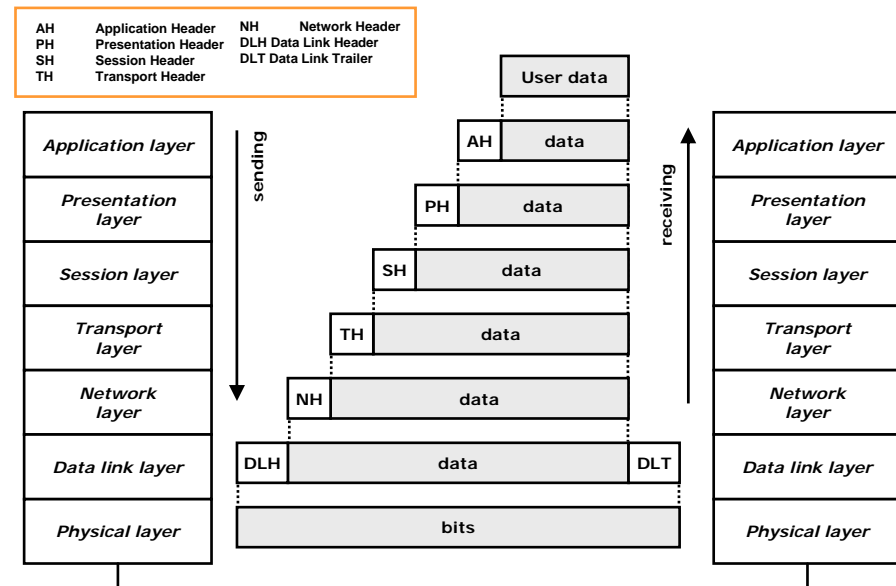
Figure 13.4 TCP/IP Concepts

Interaction Between Network Layers

- Layered protocol architecture
 - Each layer uses only services of layer directly below
 - Each layer provides services to layer directly above
 - Protocol independence
 - Modularity



- Data encapsulation
 - Lower layers treat upper layer packets as simple data
 - Headers contain control information for each layer
 - Repeated encapsulation causes overhead

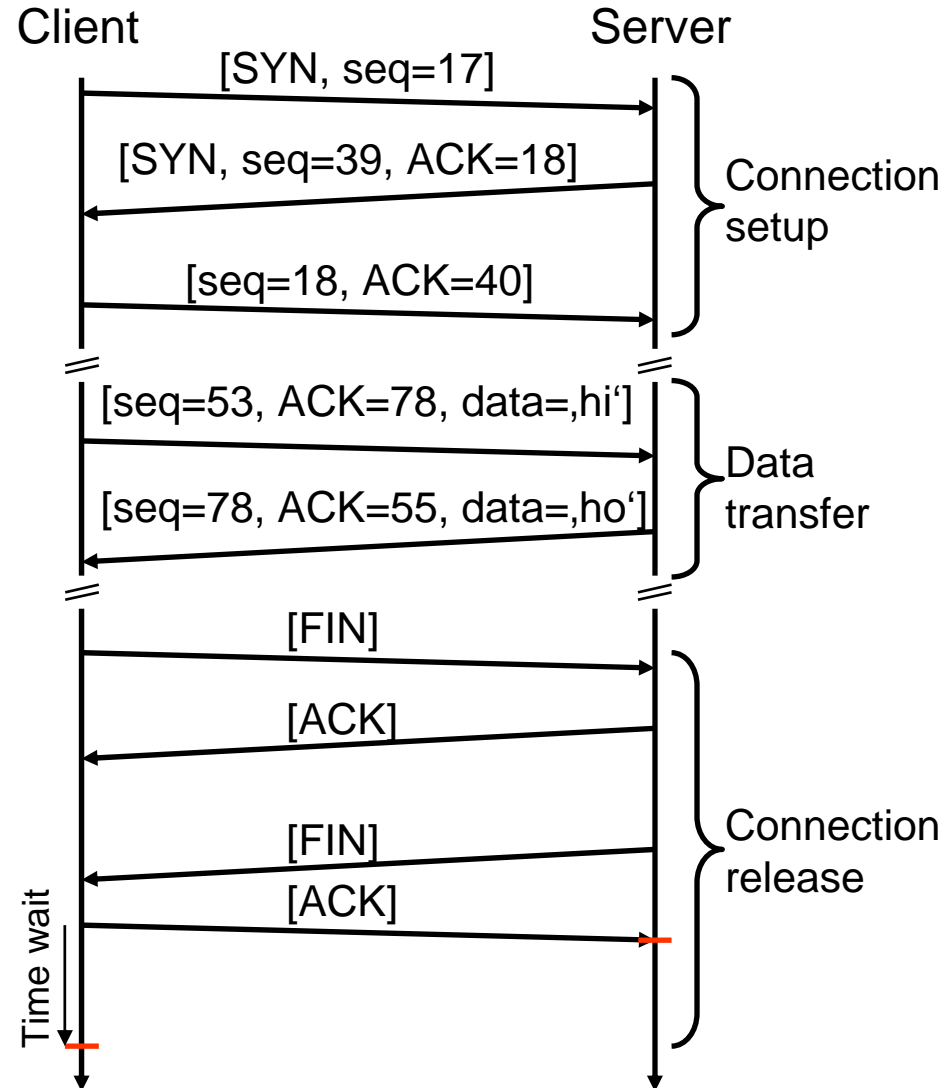
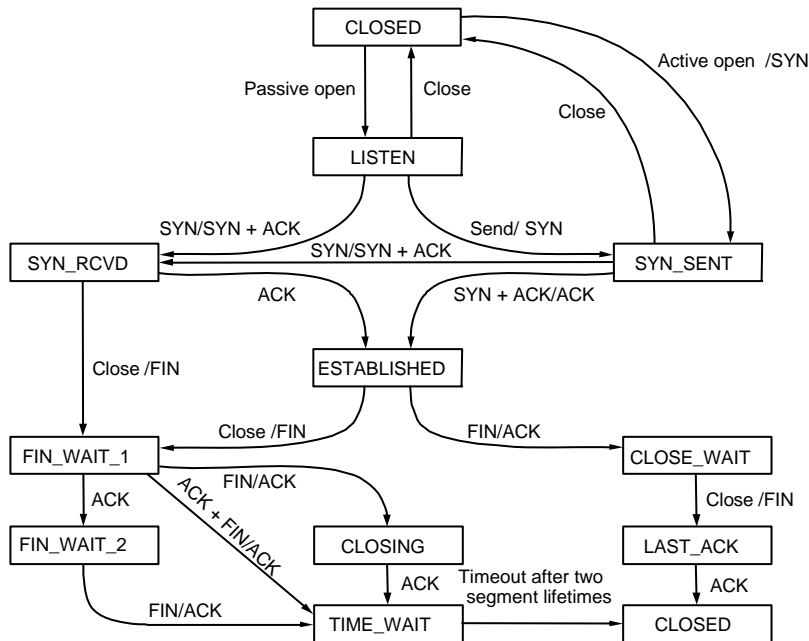


`http://www.inf.fu-berlin.de/inst/ag-tech/index.html`

- **http**: Hypertext Transfer Protocol (HTTP)
 - Protocol for accessing web pages and related content
 - Implies communication over port 80 (unless other port given in URL)
- **www.inf.fu-berlin.de**: Host name
 - Resolved to IP address via Domain Name System (DNS)
 - `www.inf.fu-berlin.de` -> `160.45.117.8`
- **inst/ag-tech/index.html**: Local resource name
 - Protocol specific parameter
 - Handled by web server

Connection Setup / Transport Layer

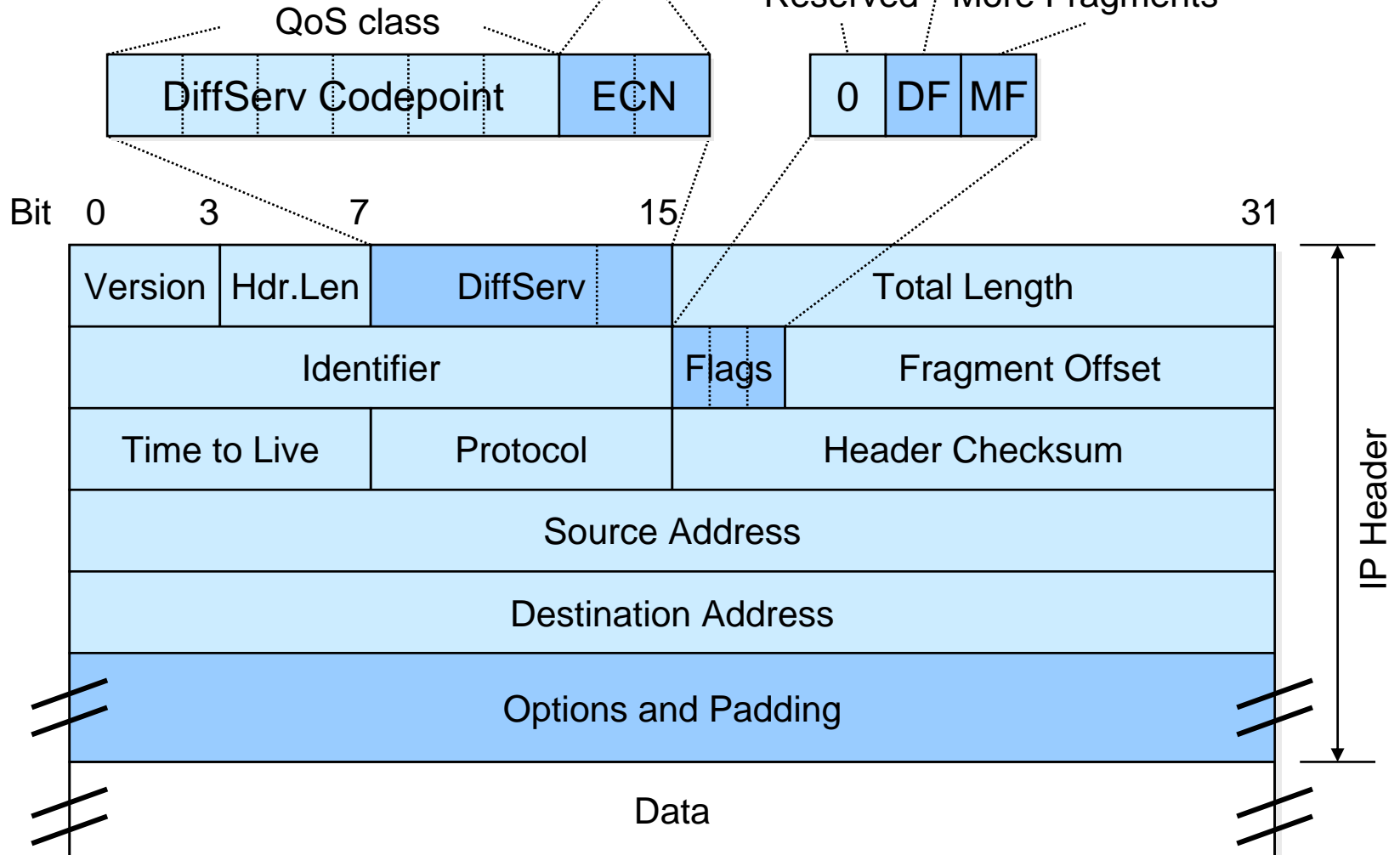
- Reliable end-to-end connection between processes
- Call to **connect()** initiates connection setup
 - TCP 3-way handshake
 - Connection parameters



Structure of Network Layer IP-Packet

Congestion control (Explicit Congestion Notification)

Don't Fragment
Reserved More Fragments



HTTP Data Encapsulation



The image shows the Wireshark network protocol analyzer interface. The main pane displays the details of a selected packet (Frame 9). The packet structure is as follows:

- Frame 9 (530 bytes on wire, 530 bytes captured)
- Ethernet II, Src: DellPcba_7d:d4:fd (00:0d:56:7d:d4:fd), Dst: Cabletro_c0:00:04 (00:e0:63:c0:00:04)
- Internet Protocol, Src: 160.45.114.44 (160.45.114.44), Dst: 160.45.117.8 (160.45.117.8)
 - Version: 4
 - Header length: 20 bytes
 - Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
 - Total Length: 516
 - Identification: 0xd0a4 (53412)
 - Flags: 0x04 (Don't Fragment)
 - Fragment offset: 0
 - Time to live: 64
 - Protocol: TCP (0x06)
 - Header checksum: 0x40c0 [correct]
 - Source: 160.45.114.44 (160.45.114.44)
 - Destination: 160.45.117.8 (160.45.117.8)
- Transmission Control Protocol, Src Port: 53523 (53523), Dst Port: www (80), Seq: 1, Ack: 1, Len: 464
- Hypertext Transfer Protocol

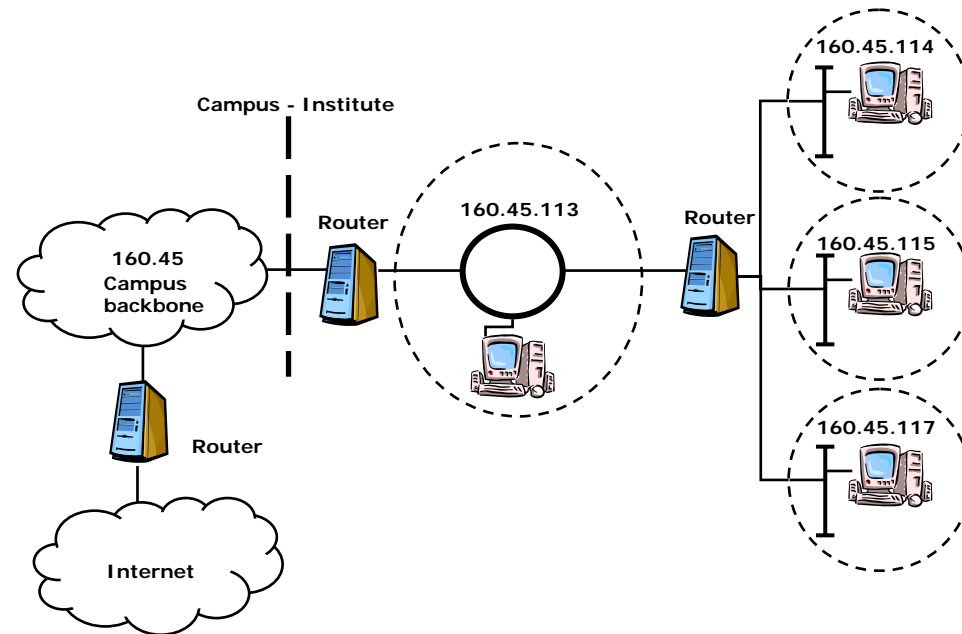
The packet bytes pane shows the raw data in hexadecimal and ASCII. The ASCII portion of the packet is an HTTP GET request:

```
....@.@. @.[-r].-
u....Pq. --w....
.m)..... 4.
|@GET /i nst/ag-t
ech/inde x.html H
TTP/1.1. User-Ag
ent: Moz illa/5.0
(compat ible; Ko
nqueror/ 3.5; Lin
ux) KHTM L/3.5.5
(like Ge cko) (De
bian)..I f-None-M
atch: "5 864234f-
25fn.457 7f3r?"
```

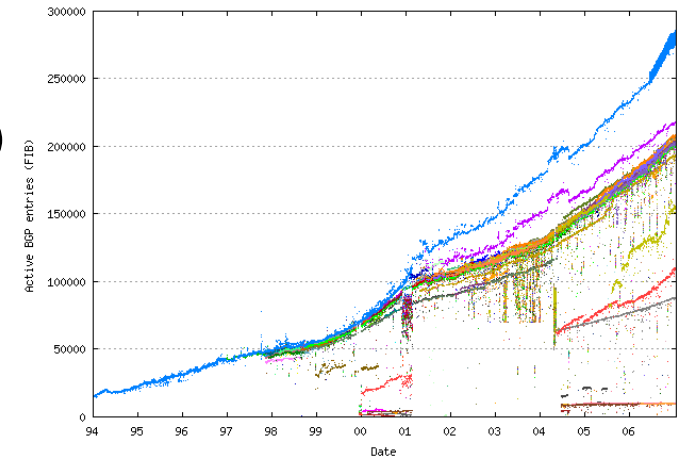
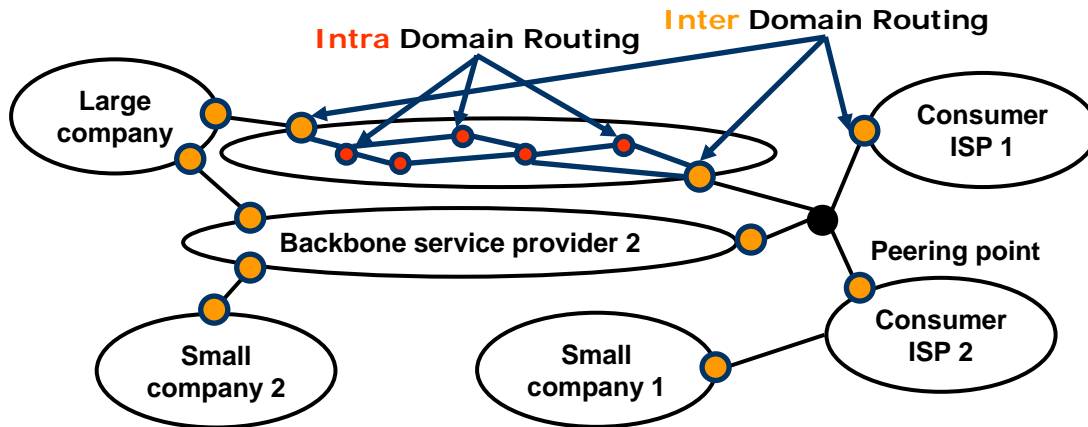
At the bottom of the interface, the source IP address is highlighted: Source (ip.src), 4 bytes. The status bar at the bottom right shows: P: 14 D: 14 M: 1 Drops: 0.

Network Layer Routing (Local Scope)

- Globally unique per host addressing
- Routers maintain tables of known networks
 - Optional route to default gateway
- Subnetting implements logical structure
 - Subnet mask builds hierarchy using host part of IP address
 - Limits broadcasts
 - More efficient routing
- Network topology may be part of security concept

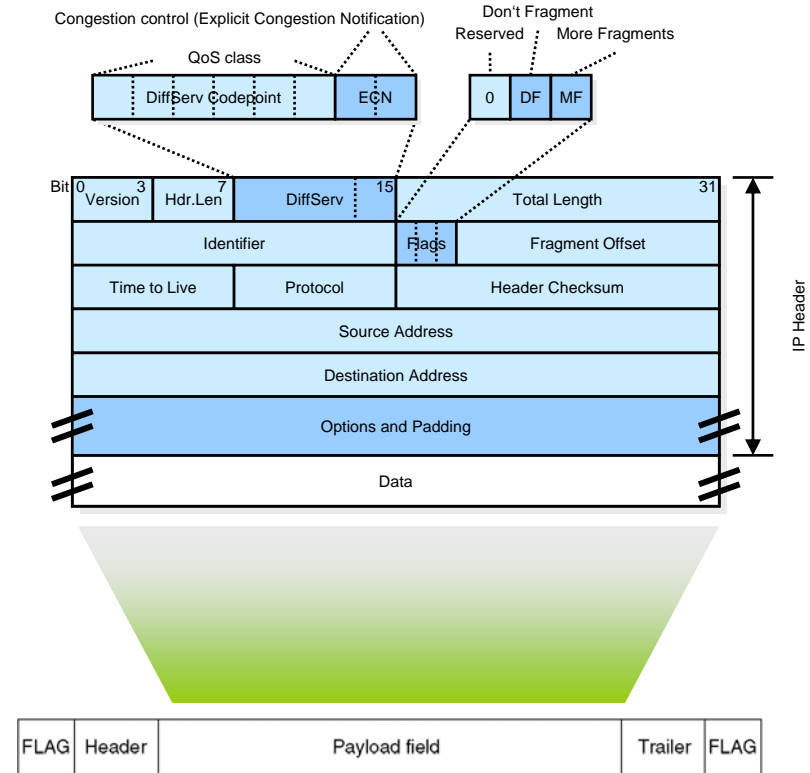
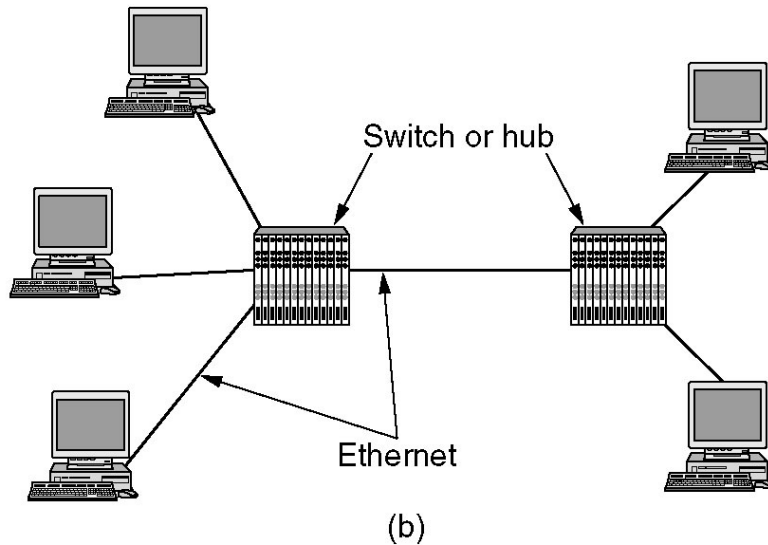


- Internet organized into autonomous systems (AS)
 - Commonly, one AS per major organization
 - Peering points to exchange data between ASs
- Intra-domain routing: OSPF, link state algorithm
- Inter-domain routing: BGPv4, distance vector protocol
 - May involve non-technical routing choices



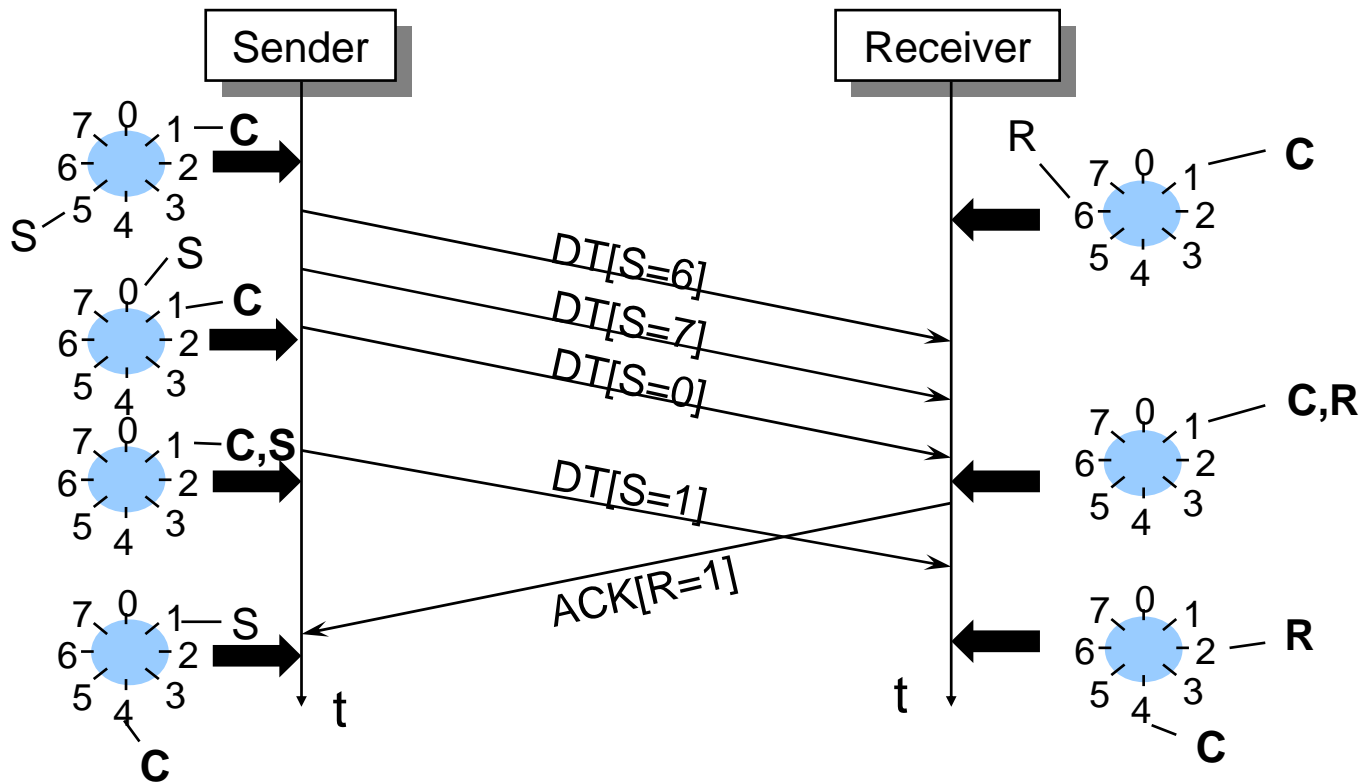
Data Link Layer Comm. (Local Scope)

- Transparent communication between two directly connected nodes
- Services include: framing, error control, connection maintenance, acknowledgements, flow control

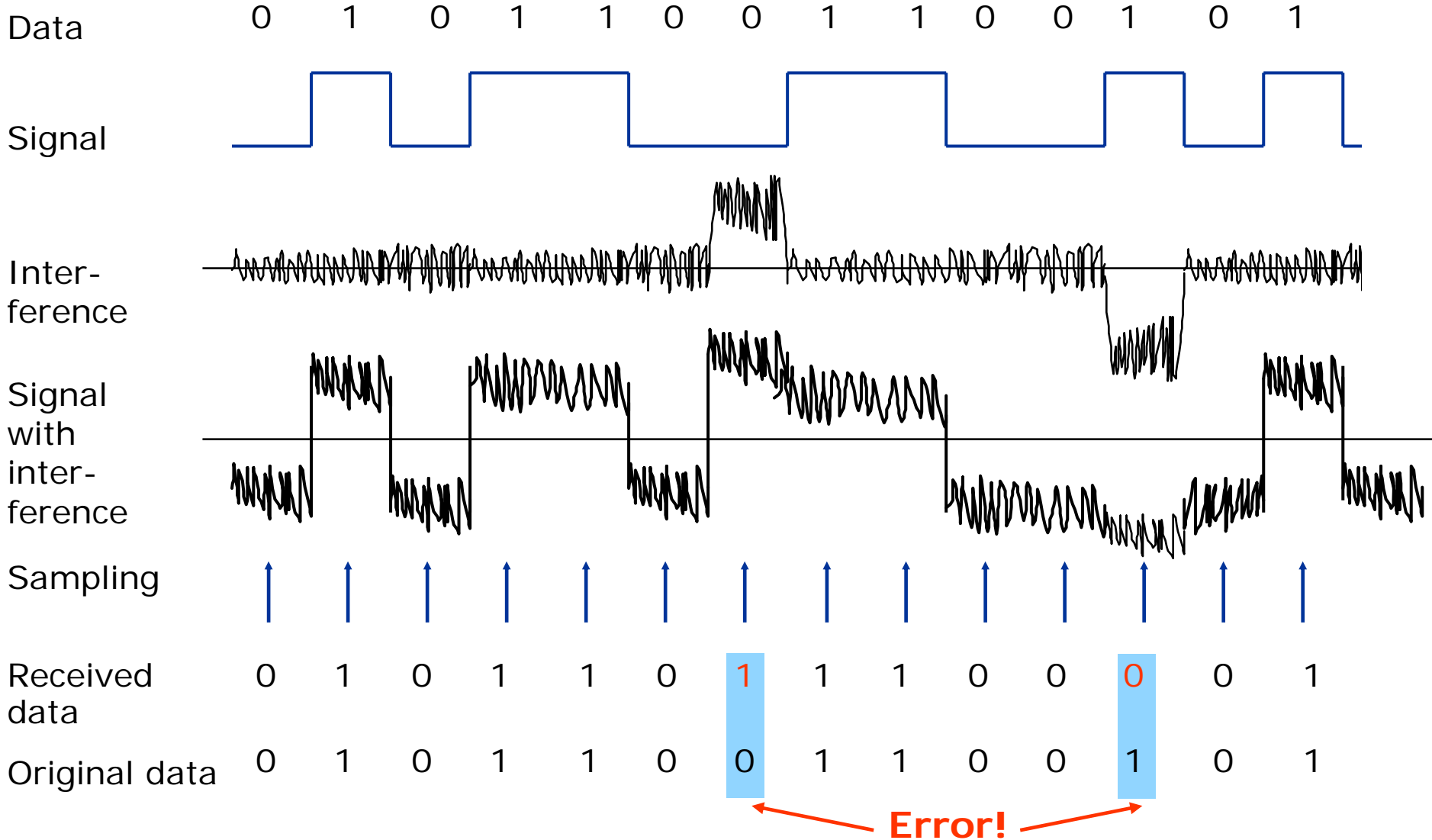


Flow Control

- Packets may be lost / Sender may flood receiver
- Need to acknowledge packets / control transmission rate
 - Credit-based schemes, e.g. sliding window



Errors During Transmission



Error Detection: Cyclic Redundancy Check (CRC)

- Reception of a correct bit sequence:

$$11\ 0011\ 1001 \div 1\ 1001 = 10\ 0001$$

$$\begin{array}{r} 11\ 001 \\ \hline \end{array}$$

$$00\ 0001\ 1001$$

$$\begin{array}{r} 1\ 1001 \\ \hline \end{array}$$

$$0\ 0000 = \text{remainder}$$

- No remainder, thus the received bits *should* be error free

- Reception of a erroneous bit sequence:

$$11\ 1111\ 1000 \div 1\ 1001 = 10\ 1001$$

$$\begin{array}{r} 11\ 001 \\ \hline \end{array}$$

$$00\ 1101\ 1$$

$$\begin{array}{r} 1100\ 1 \\ \hline \end{array}$$

$$0001\ 0000$$

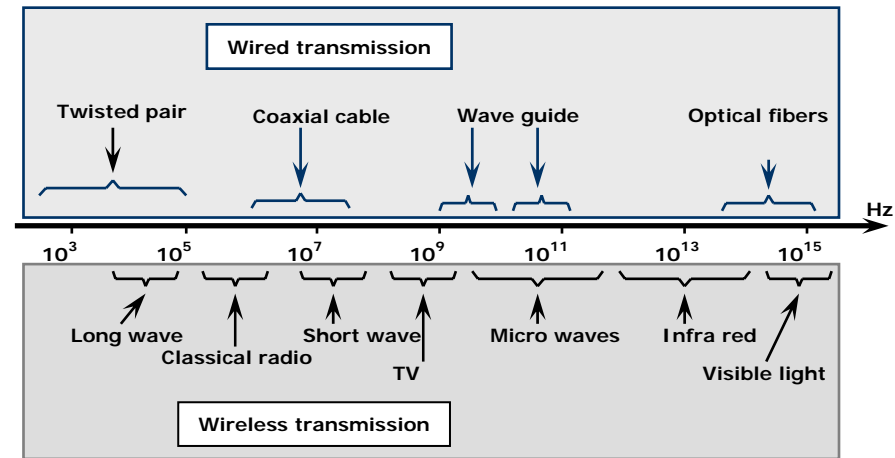
$$\begin{array}{r} 1\ 1001 \\ \hline \end{array}$$

$$0\ 1001 = \text{remainder} \neq 0$$

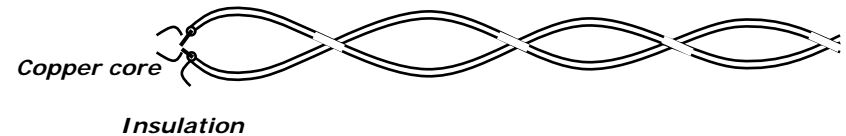
- There is a remainder unequal 0, thus there was definitely a transmission error

Physical Layer

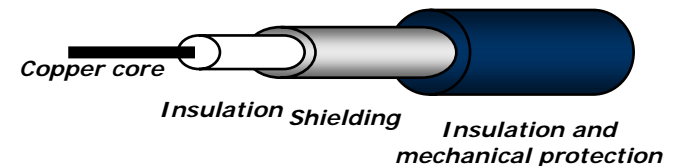
- Packet / sequence of bits turned into physical signal
- Signal propagation depends on physical medium (limited bandwidth, attenuation, dispersion) and background noise
- Mapping between bits and (multi-valued) symbols
- Baseband transmission vs. modulation (broadband transmission)



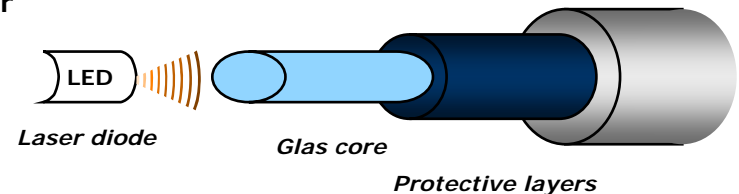
Twisted pair



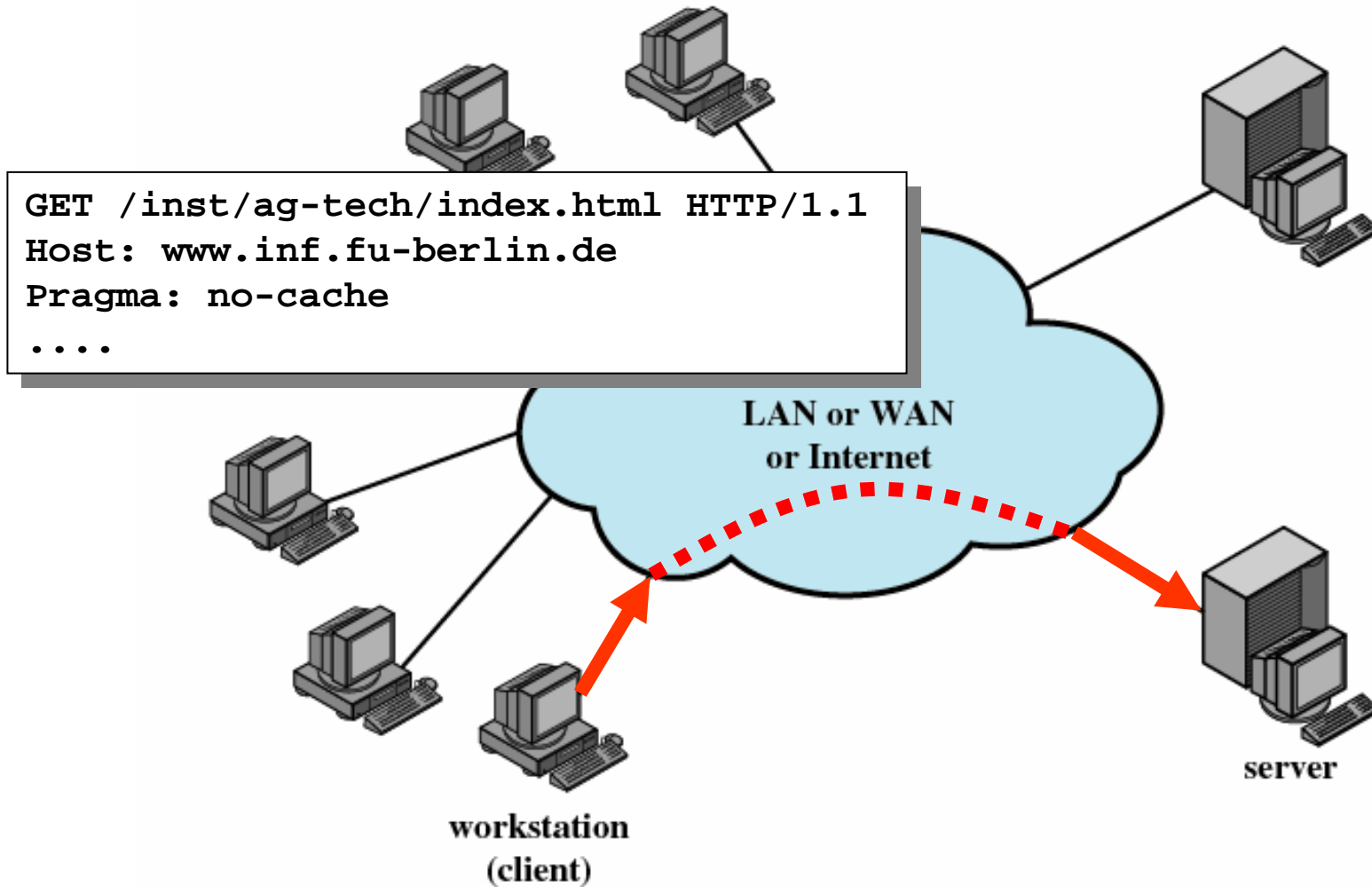
Coaxial



Optical fiber



Client/Server Communication



At the server...

- Web server is one of many processes running locally

```
wittenbu@vienna: home/datsche/wittenbu - Shell - Konsole
top - 10:33:30 up 2 days, 1:04, 1 user, load average: 0.41, 0.26, 0.17
Tasks: 93 total, 1 running, 92 sleeping, 0 stopped, 0 zombie
Cpu(s):  0.3% us,  0.0% sy,  0.0% ni, 99.7% id,  0.0% wa,  0.0% hi,  0.0% si
Mem:  1833264k total,  967528k used,  65736k free,  160112k buffers
Swap: 2015992k total,    0k used,  2015992k free,  445496k cached

  PID USER      PR  NI  VIRT  RES  SHR  S %CPU  %MEM    TIME+  COMMAND
 11843 wittenbu  16   0 1944   568  740  R  0.3   0.1   0:00.54 top
    1 root      15   0 1584   520  452  S  0.0   0.1   0:01.44 init
    2 root       0   0  0     0  0  S  0.0   0.0   0:00.00 migration/0
    3 root      34  19  0     0  0  S  0.0   0.0   0:00.00 ksoftirqd/0
    4 root      10  -5  0     0  0  S  0.0   0.0   0:02.72 events/0
    5 root      13  -5  0     0  0  S  0.0   0.0   0:00.02 khelper
    6 root      10  -5  0     0  0  S  0.0   0.0   0:00.00 kthread
    8 root      10  -5  0     0  0  S  0.0   0.0   0:00.15 kblockd/0
   11 root      10  -5  0     0  0  S  0.0   0.0   0:00.00 khubd
   13 root      10  -5  0     0  0  S  0.0   0.0   0:00.00 kseriod
  104 root      20   0  0     0  0  S  0.0   0.0   0:00.00 pdflush
  105 root      15   0  0     0  0  S  0.0   0.0   0:01.00 pdflush
  106 root      15   0  0     0  0  S  0.0   0.0   0:00.02 kswapd0
  107 root      20  -5  0     0  0  S  0.0   0.0   0:00.00 aio/0
  108 root      20  -5  0     0  0  S  0.0   0.0   0:00.00 xfslogd/0
  109 root      20  -5  0     0  0  S  0.0   0.0   0:00.00 xfsdatad/0
   764 root      11  -5  0     0  0  S  0.0   0.0   0:00.00 ata/0
   781 root      11  -5  0     0  0  S  0.0   0.0   0:00.00 kpsnoused
```

- Upon receiving packet, network interface controller (NIC) will raise interrupt
- Kernel will handle the packet and notify the web server process

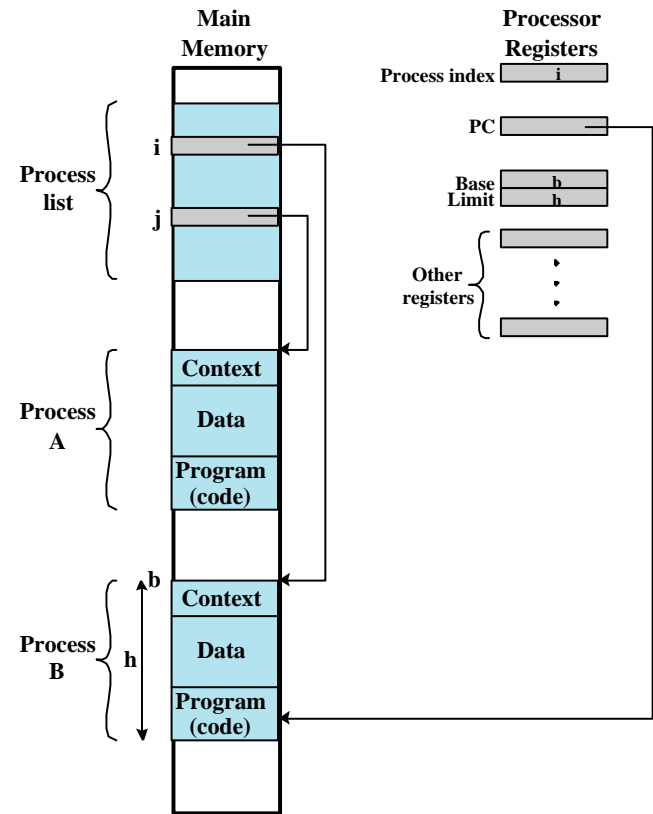


Figure 2.8 Typical Process Implementation

Processing of HTTP-GET Request

- Web server retrieves file `inst/ag-tech/index.html` from local file system
 - System calls to access secondary storage
 - Kernel maps file name to data layout on disk
- Web server sends data to client

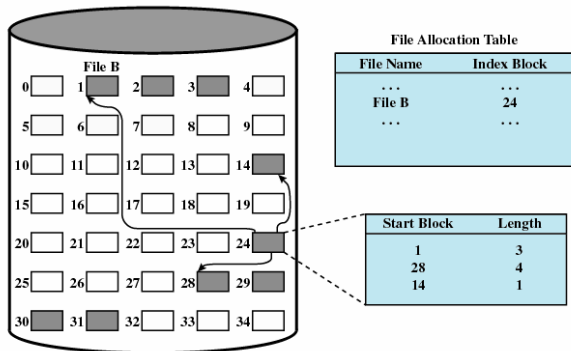


Figure 12.12 Indexed Allocation with Variable-Length Portions

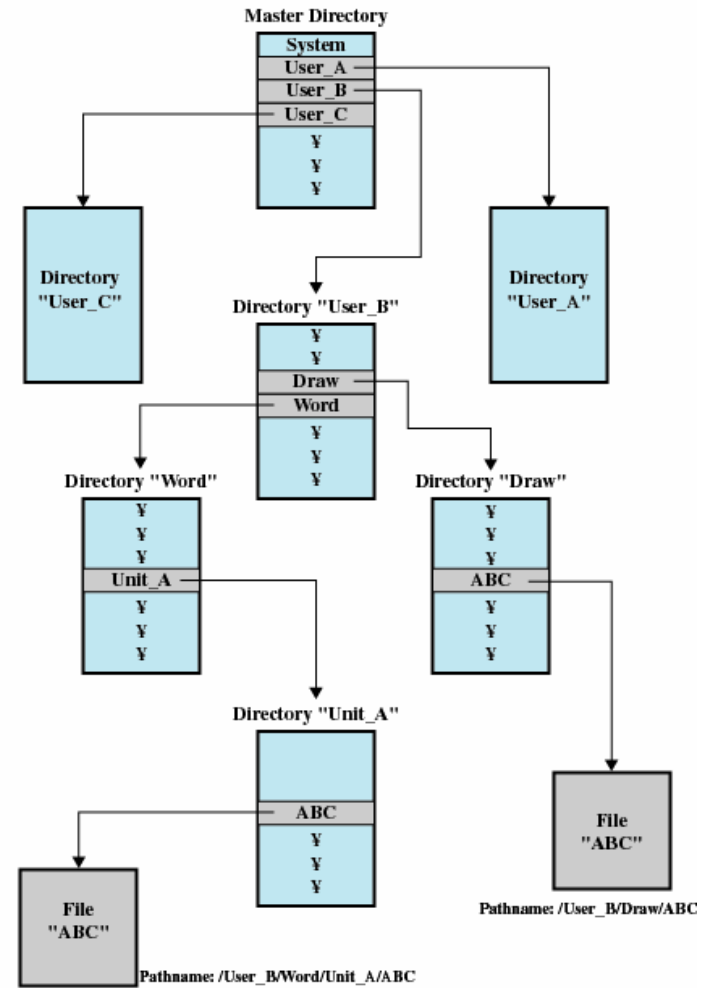
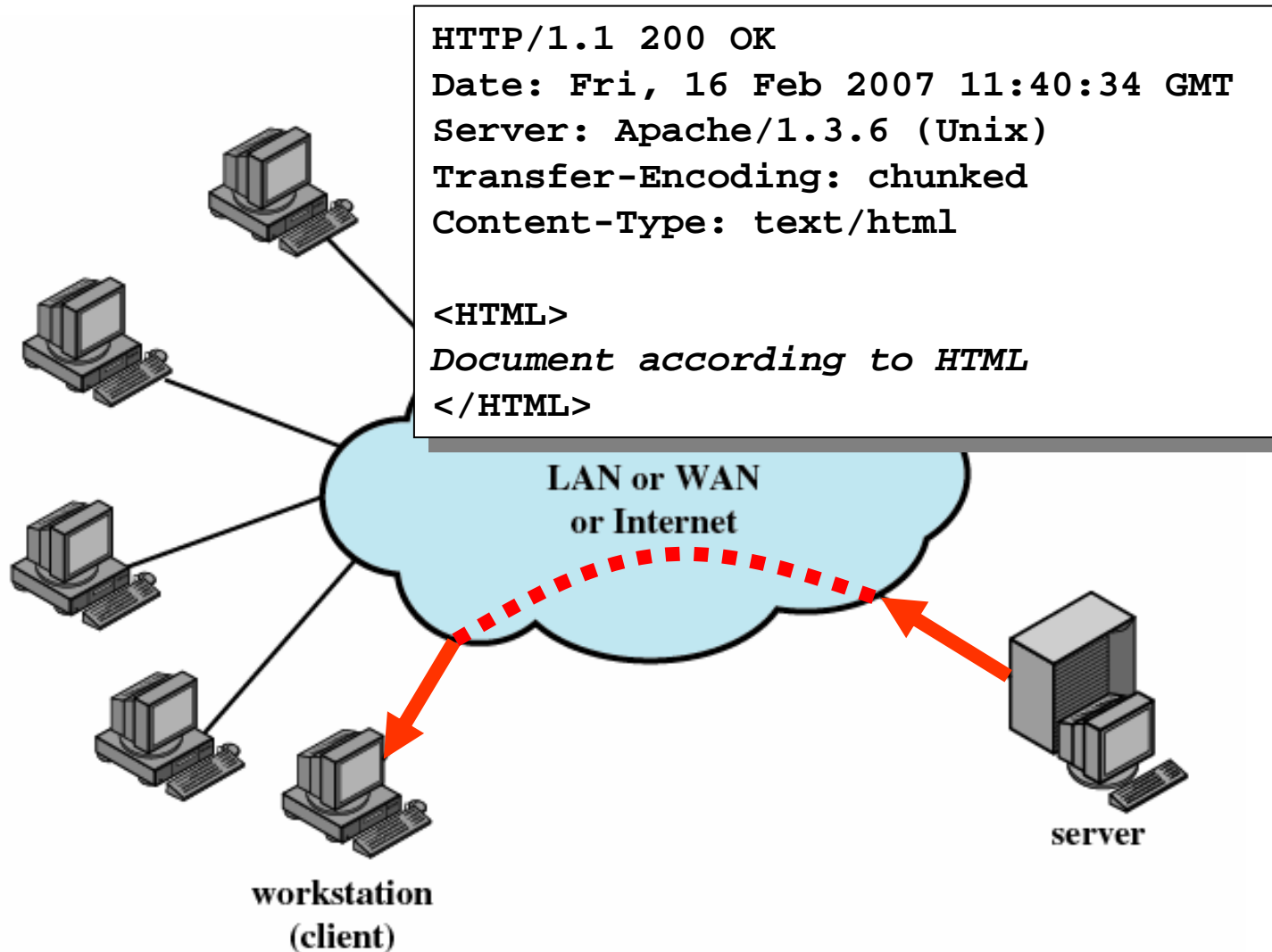


Figure 12.5 Example of Tree-Structured Directory

Server Replies to Client



Client Data Processing

- Client host receives packet
- Kernel hands data to web browser process
- Web browser renders page
 - May have to allocate memory in the process
- Finally, browser updates user interface via system call

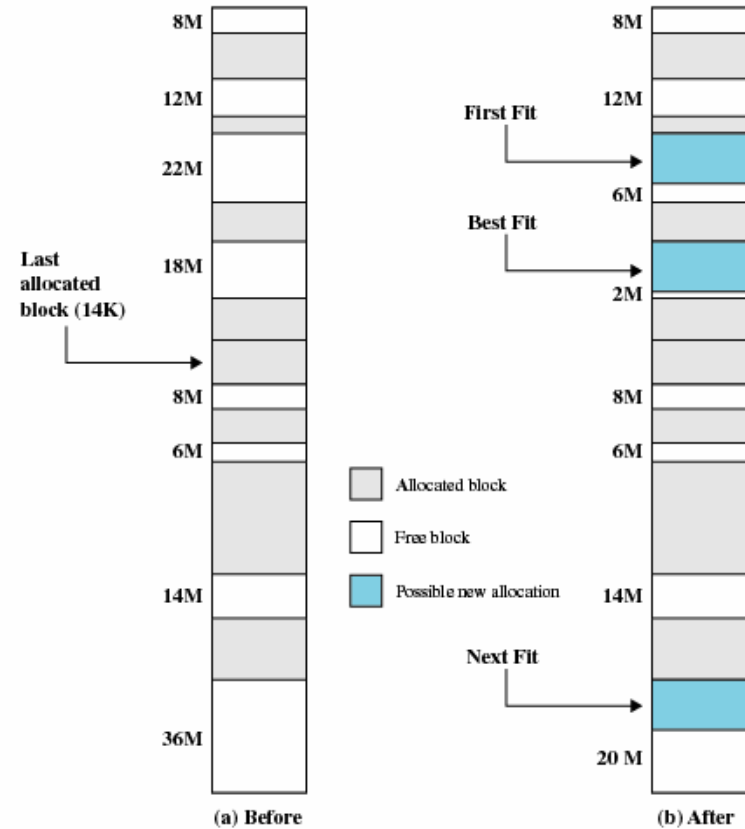
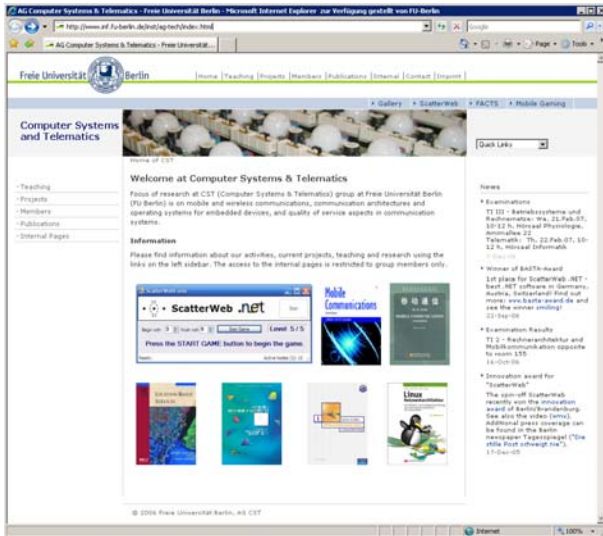
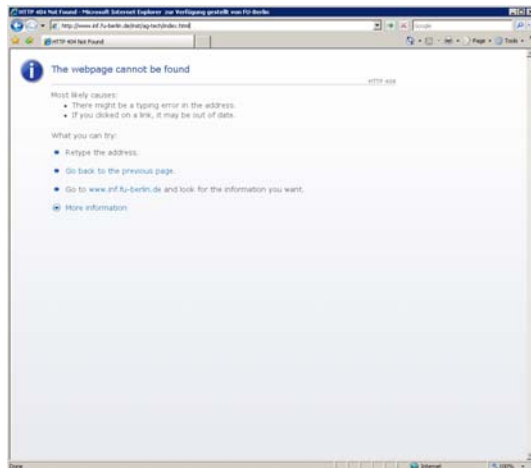


Figure 7.5 Example Memory Configuration Before and After Allocation of 16 Mbyte Block

A Comprehensive Example



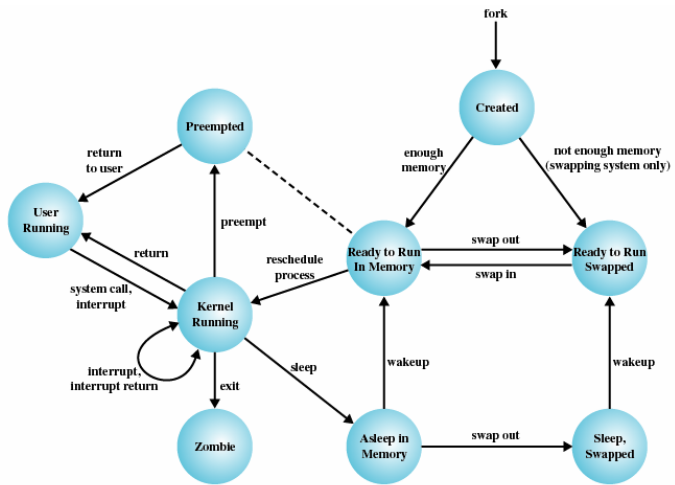


Figure 3.17 UNIX Process State Transition Diagram

Fin

