Internet, HTTP and DNS Examples

The Internet and an internet

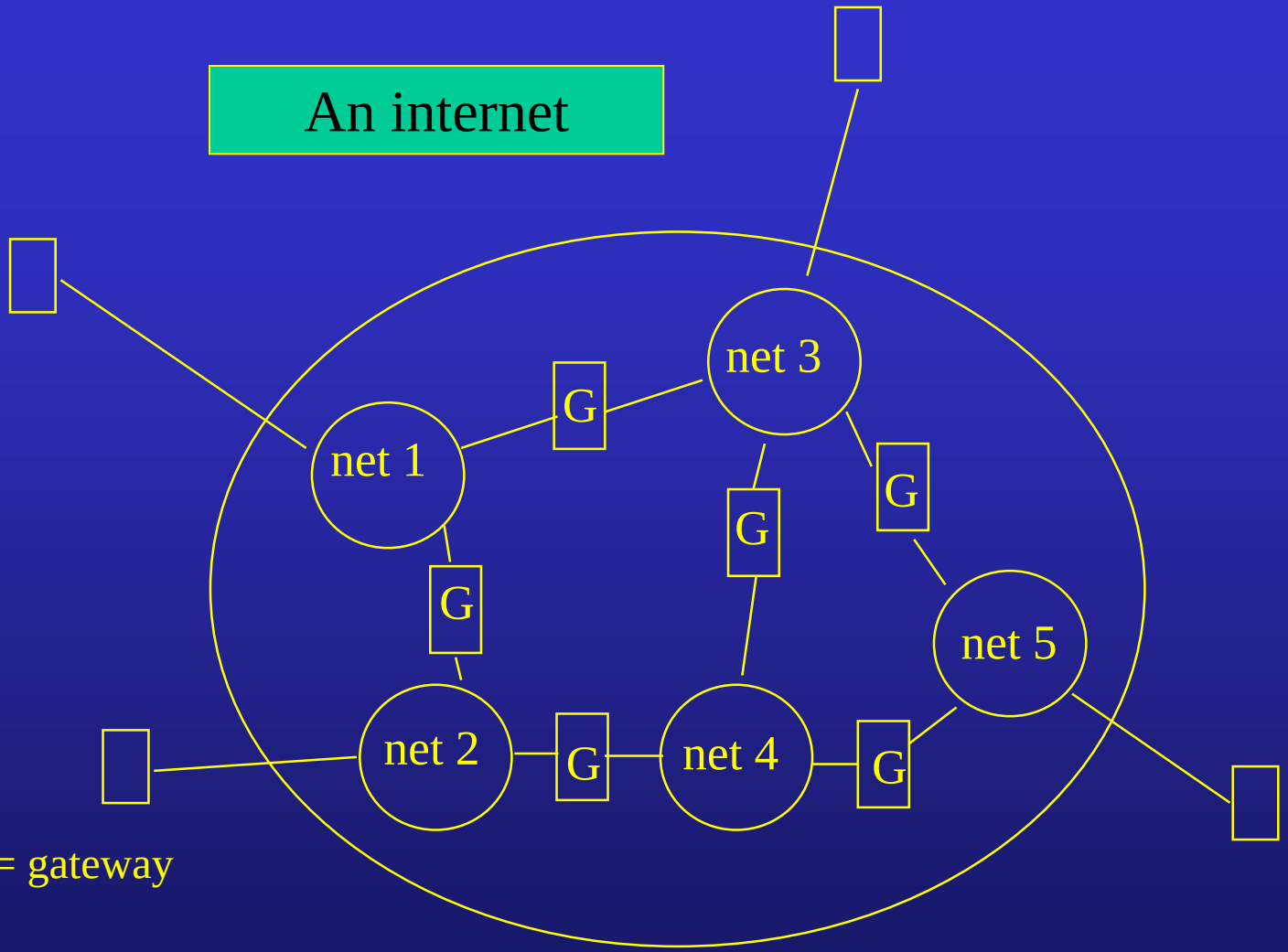
[LG&W pp.26-28]

internet :: involves the *interconnection* of multiple networks into a single large networks.

the "Internet" :: refers to the successor to ARPANET.

IP (the Internet Protocol) :: provides connectionless transfer of packets across an internet.

An internet



G = gateway

Leon-Garcia & Widjaja: *Communication Networks*

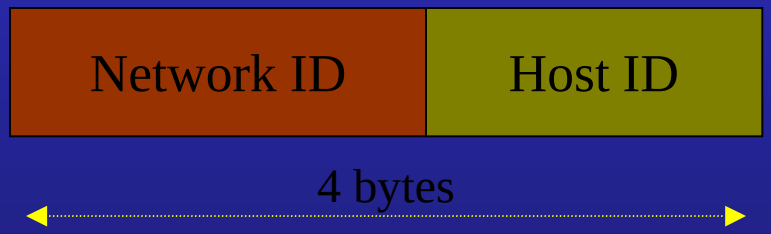
Figure 1.18

IP

- Currently provides *best-effort service*
  - packets may be lost (i.e., IP is unreliable).
- General design philosophy
  - Keep internal operations simple by relegating complex functions to the edge of the subnet.
  - IP can operate over any network
  - allow IP to scale!!!
  - *The end-to-end mechanisms are responsible for recovery of packet losses and congestion control.*

IP

- Uses *hierarchical address space* with location information embedded in the structure.



- IP address is usually expressed in *dotted-decimal notation* (e.g., 128.100.11.56).

Internet

- Provides a *name space* to refer to machines connected to the Internet (e.g. chablis.cs.wpi.edu).
- The name space is hierarchical, but is only administrative and not used in network routing operations.
- DNS (Domain Name Service) provides automatic translation of names to addresses.

Applications and Layered Architectures

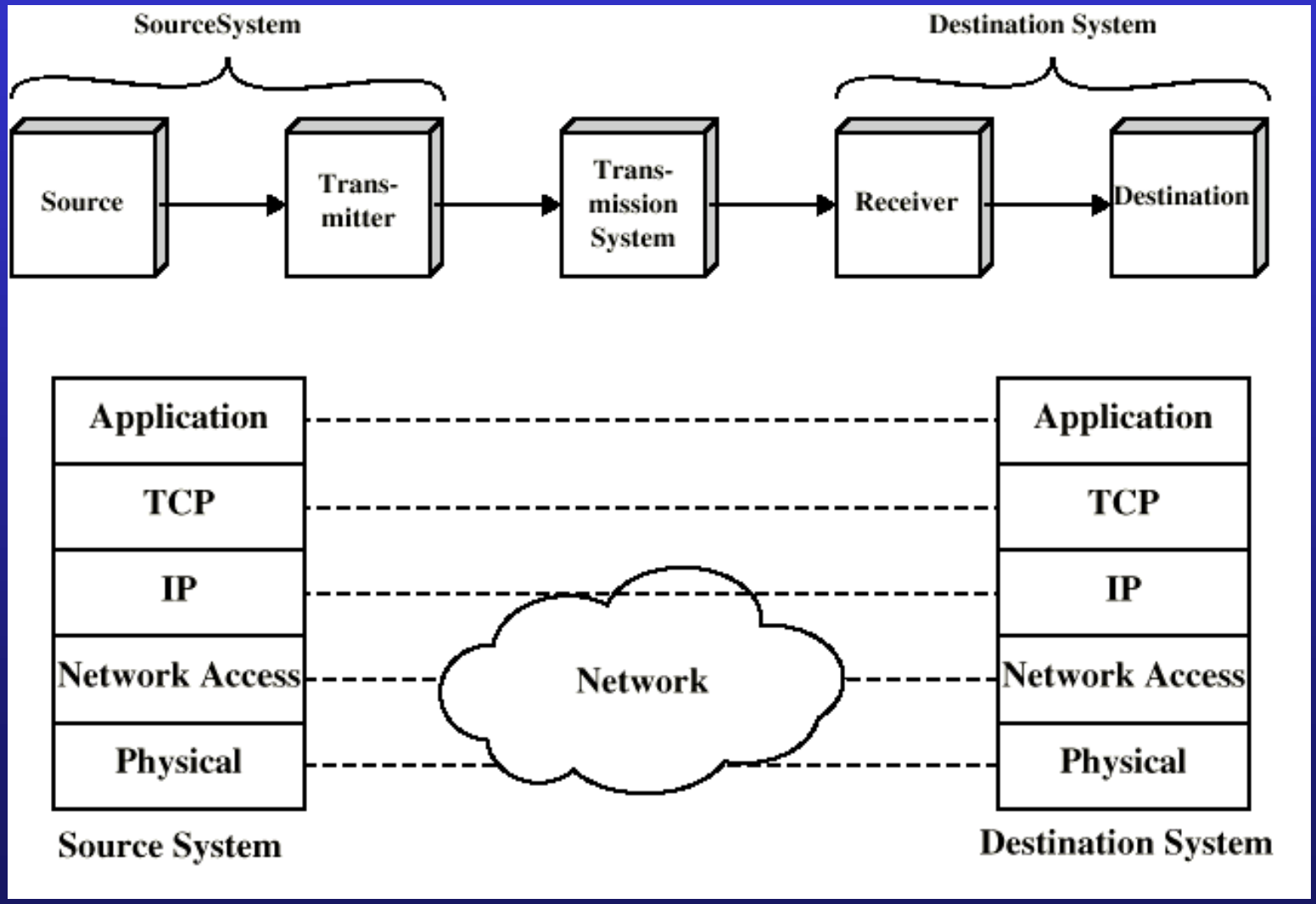
[LG&W pp.43-49]

- In the 1970's vendor companies (IBM and DEC) developed *proprietary networks* with the common feature of grouping communication functions into related and manageable sets called **layers**.

network architecture :: a set of protocols that specify how every layer is to function.

TCP/IP Protocol Architecture Model

DCC 6th Ed., W. Stallings Figure 1.9



Layering Advantages

- Simplified the design process.
- Led to flexibility in modifying and developing the network.
- Accommodates incremental changes more readily.

Layering Examples

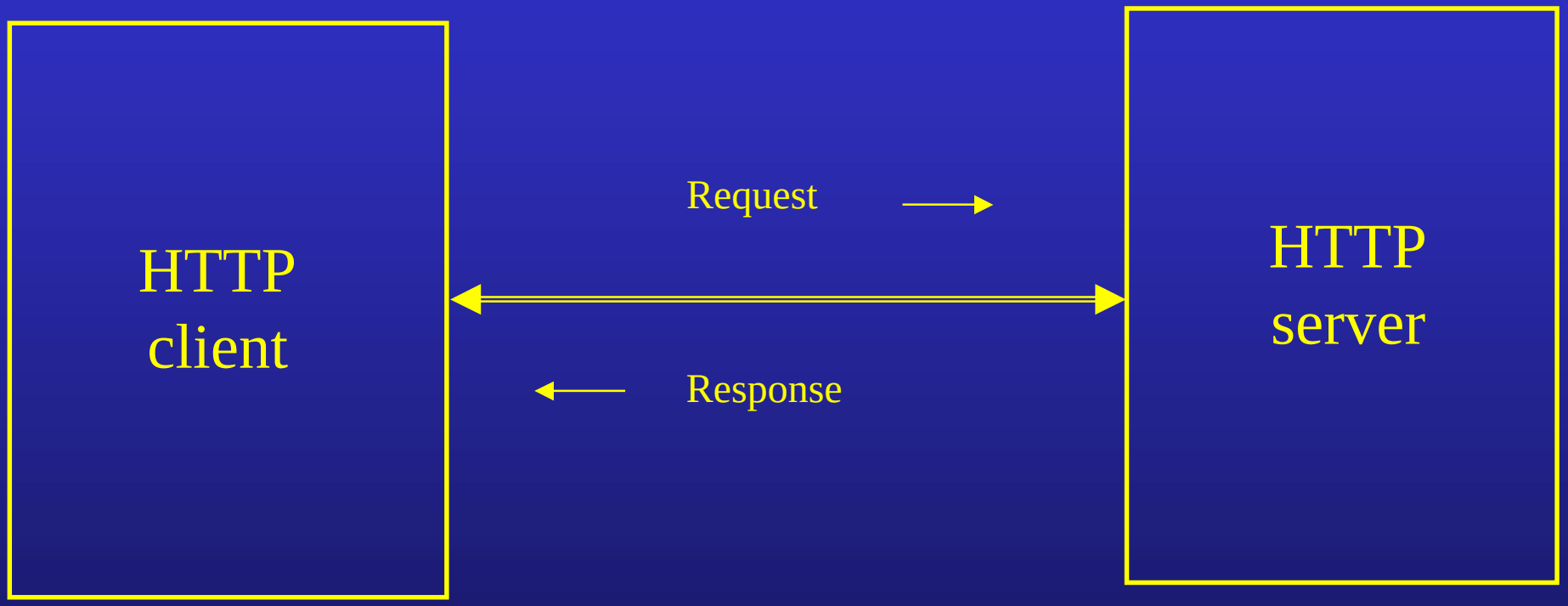
Client/server relationship ::

- Server process waits for incoming requests by listening to a **port**.
- Client process makes *requests* as required.
- Server process provides *responses* to these requests.
- The server process usually runs in the background as a **daemon** **(e.g. httpd is the server daemon for HTTP).**

HTTP Example

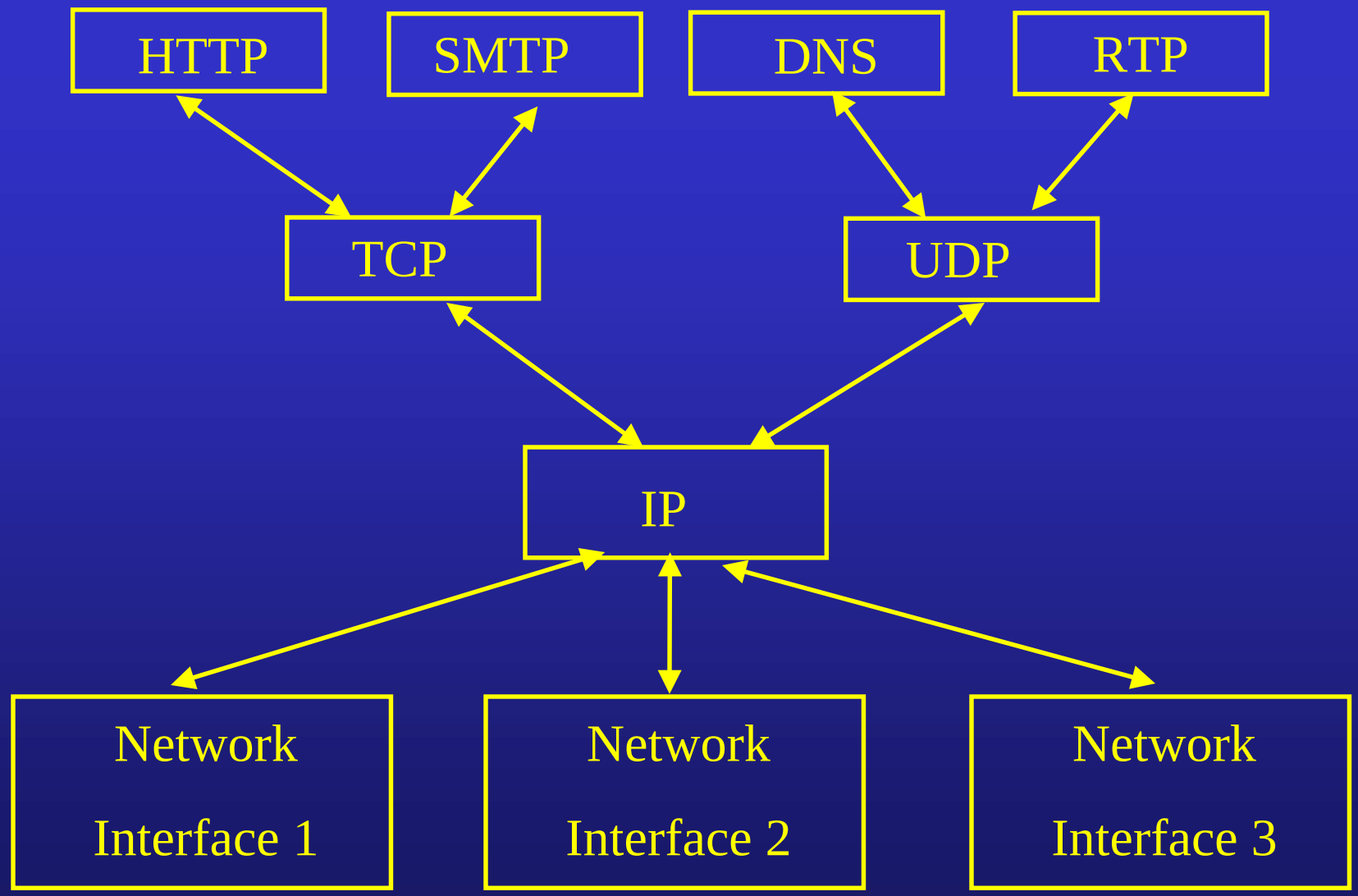
- HTTP (HyperText Transfer Protocol) specifies rules by which the client and the server interact so as to retrieve a document.
- The protocol assumes the client and the server can exchange messages directly
- The client software needs to set up a two-way connection prior to the HTTP request.

HTTP client/server interaction



Leon-Garcia & Widjaja: *Communication Networks*

Figure 2.1



Leon-Garcia & Widjaja: *Communication Networks*

Figure 2.12

1. The user clicks on a link to indicate which document is to be retrieved.

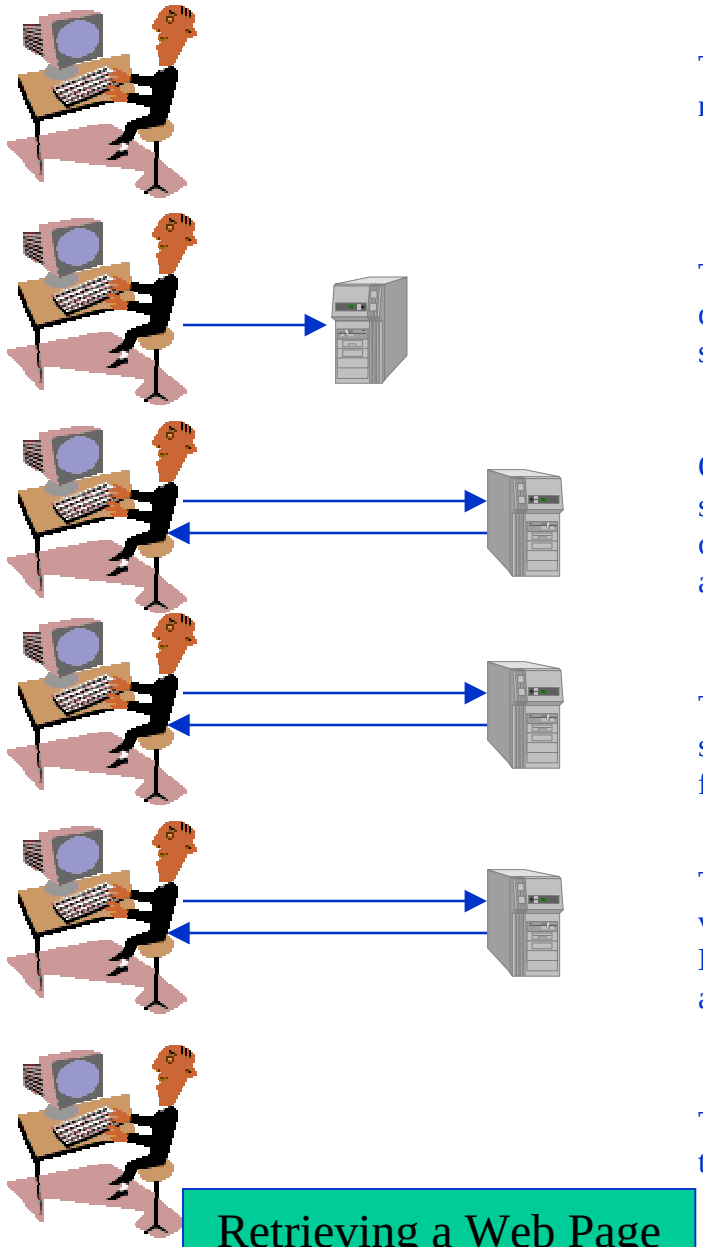
2. The browser must determine the address that contains the document. It does this by sending a query to its local name server.

3. Once the address is known the browser establishes a connection to the specified machine, usually a TCP connection. In order for the connection to be successful, the specified machine must be ready to accept TCP connections.

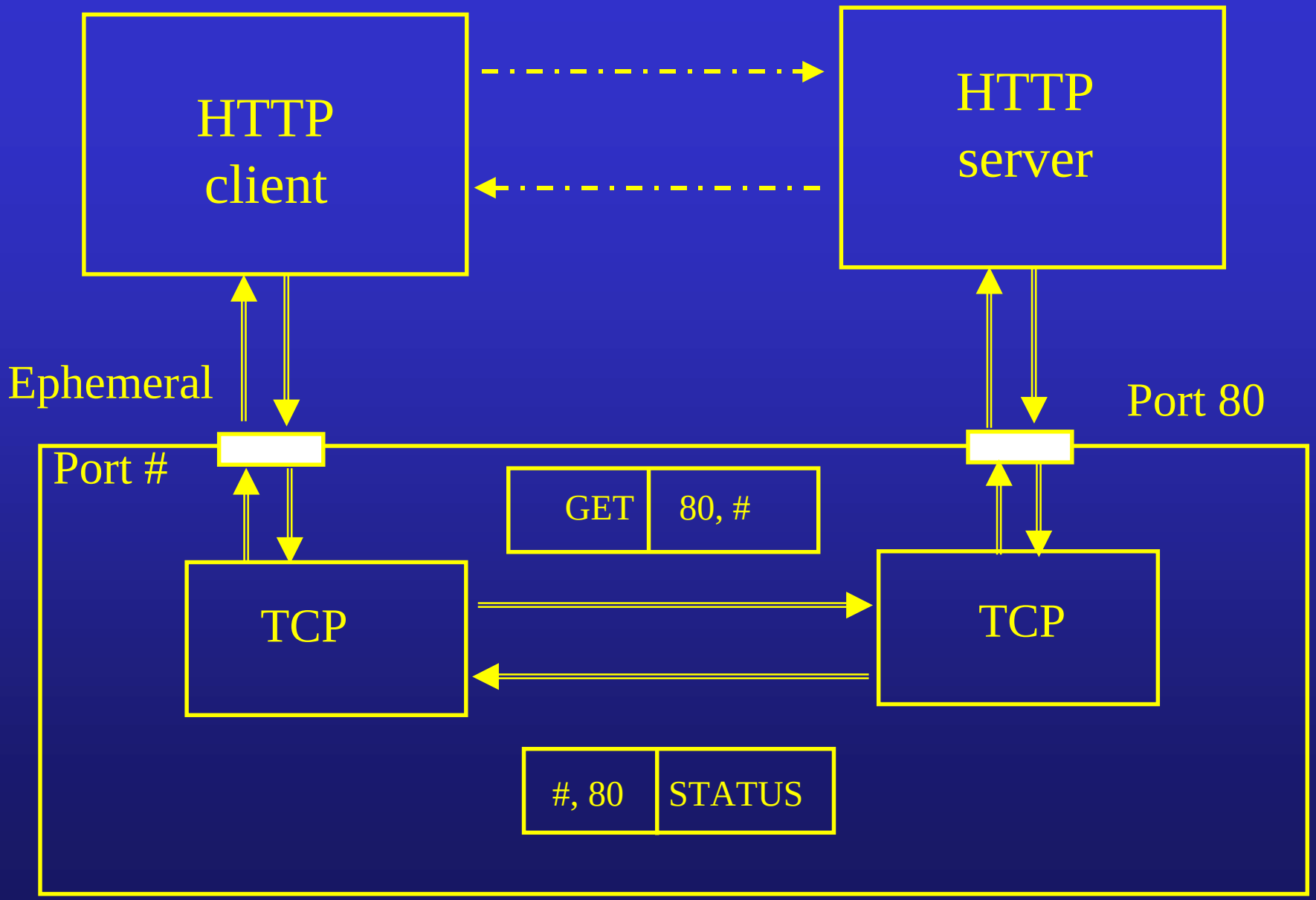
4. The browser runs a client version of HTTP, which issues a request specifying both the name of the document and the possible document formats it can handle.

5. The machine that contains the requested document runs a server version of HTTP. It reacts to the HTTP request by sending an HTTP response which contains the desired document in the appropriate format.

6. The TCP connection is then closed and the user may view the document.



Retrieving a Web Page


HTTP client
HTTP server
Ephemeral
Port 80
Port #
GET
80, #
TCP
TCP
#, 80
STATUS