DIPARTIMENTO
DI INFORMATICA
Via Salaria 113 - 00198, Roma

SAPIENZA
UNIVERSITÀ DI ROMA

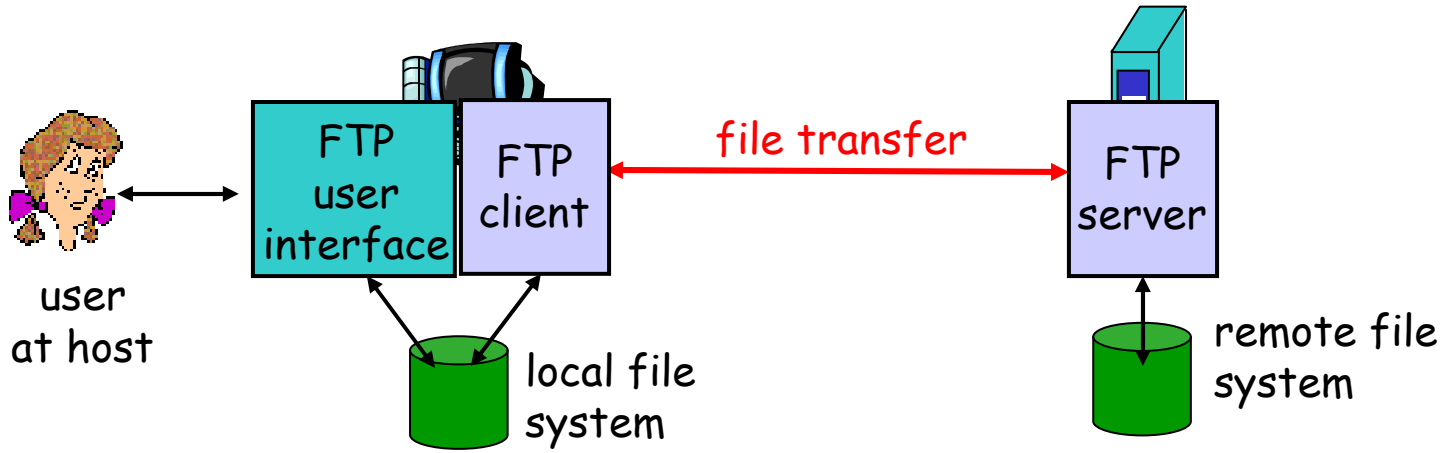
Chapter 2
Application Layer

Reti di Elaboratori
Corso di Laurea in Informatica
Università degli Studi di Roma "La Sapienza"
Canale A-L
Prof.ssa Chiara Petrioli

Chapter 2 outline

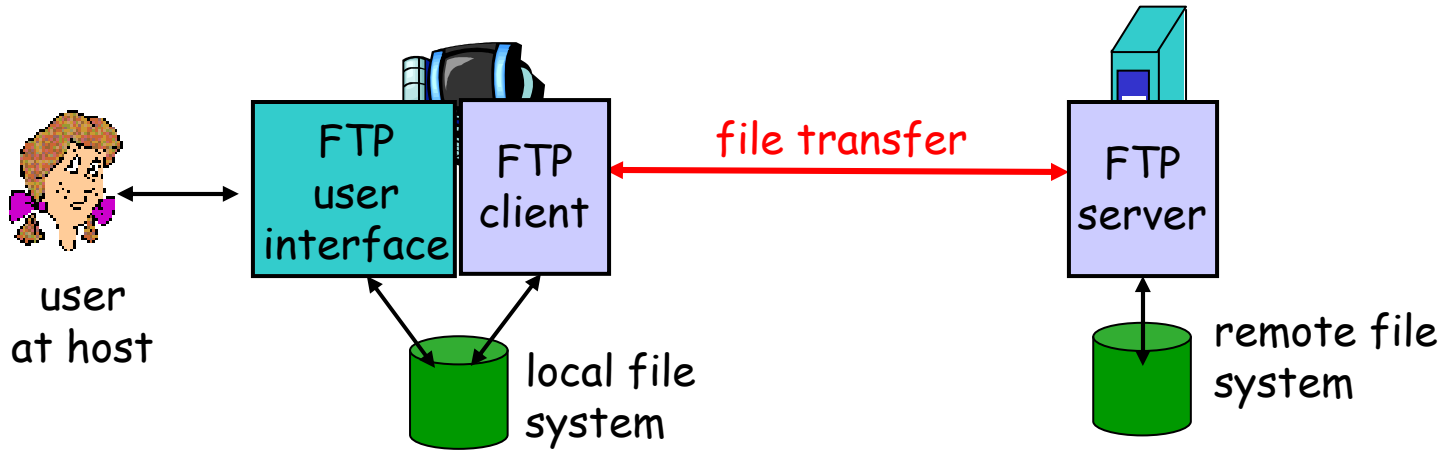
- 2.1 Principles of app layer protocols
  - clients and servers
  - app requirements
- 2.2 Web and HTTP
- 2.3 FTP
- 2.4 Electronic Mail
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.9 Content distribution
  - Network Web caching
  - Content distribution networks
  - P2P file sharing
- Telnet

FTP: the file transfer protocol



- transfer file to/from remote host
- client/server model
  - *client:* side that initiates transfer (either to/from remote)
  - *server:* remote host
- ftp: RFC 959
- ftp server: port 21

FTP: the file transfer protocol

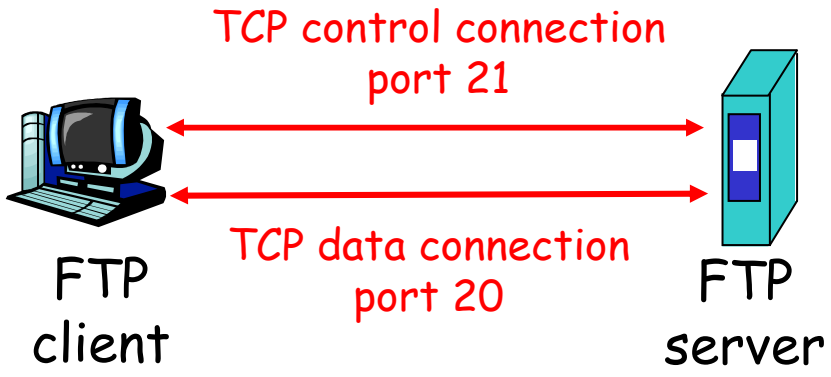


The FTP commands specify the parameters for the data connection (data port, transfer mode, representation type, and structure) and the nature of file system operation (store, retrieve, append, delete, etc.).

- data connection may be used for simultaneous sending and receiving.

FTP: separate control, data connections

- FTP client contacts FTP server at port 21, TCP is transport protocol
- client authorized over control connection
- client browses remote directory by sending commands over control connection.
- when server receives file transfer command, server opens 2^{nd} TCP connection (for file) to client
- after transferring one file, server closes data connection.

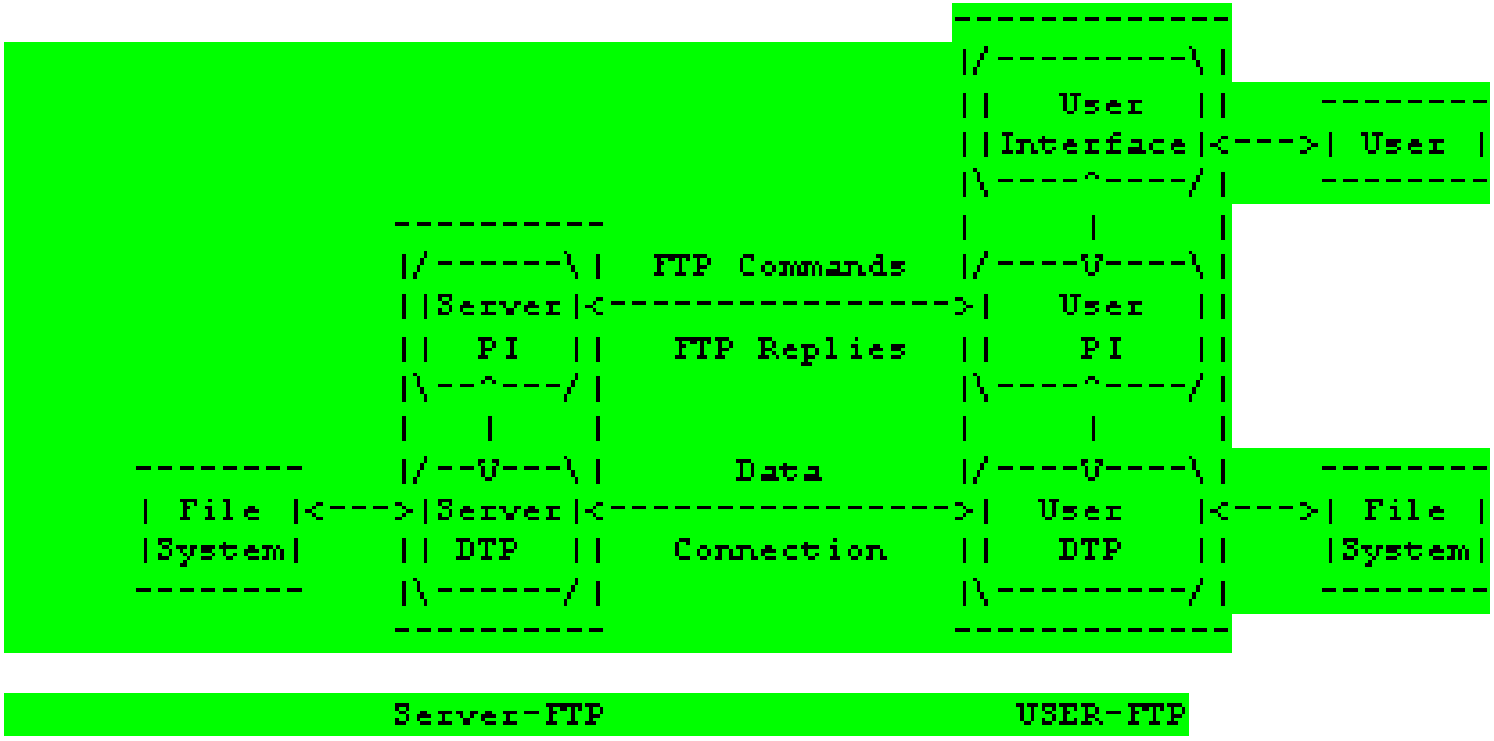


- server opens another TCP data connection to transfer another file.
- control connection: "out of band"
- FTP server maintains "state": current directory, earlier authentication

Establishing a connection

- Data connection
  - Setting up the data connection to the appropriate ports and choosing the parameters for transfer. Both the user and the server-DTPs have a default data port.
    - The user-process default data port is the same as the control connection port
    - The server-process default data port is the port adjacent to the control connection port
  - It is possible for the user to specify an alternate data port by use of the PORT command.
  - In general, it is the server's responsibility to maintain the data connection--to initiate it and to close it
  - . The server MUST close the data connection under the following conditions: 1. The server has completed sending data in a transfer mode that requires a close to indicate EOF. 2. The server receives an ABORT command from the user. 3. The port specification is changed by a command from the user. 4. The control connection is closed legally or otherwise. 5. An irrecoverable error condition occurs.

FPT model (da standard) 1/2



FPT model (da standard) 2/2

- A user might wish to transfer files between two hosts, neither of which is a local host.
- The user sets up control connections to the two servers and then arranges for a data connection between them.
- Control information is passed to the user-PI but data is transferred between the server data transfer processes.

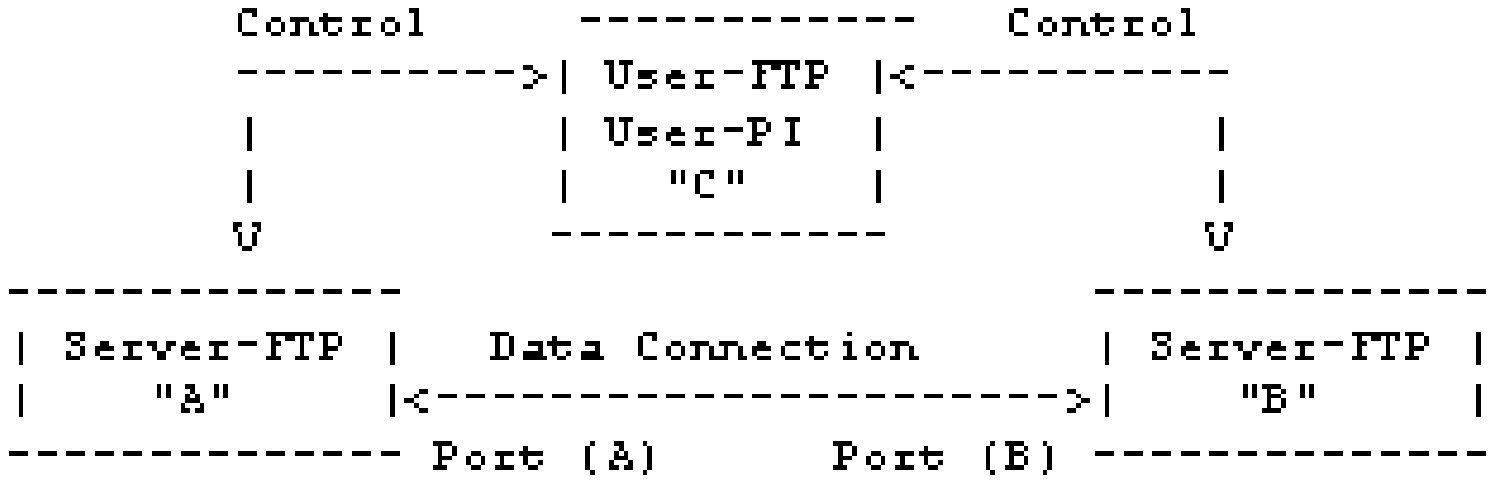


Figure 2

FTP commands, responses

Sample commands:

- sent as [ASCII text] over control channel
- **USER *username***
- **PASS *password***
- **LIST** return list of file in current directory
- **RETR filename** retrieves (gets) file
- **STOR filename** stores (puts) file onto remote host

Sample return codes

- status code and phrase (as in HTTP)
- **331 Username OK, password required**
- **125 data connection already open; transfer starting**
- **425 Can't open data connection**
- **452 Error writing file**

FTP Commands

- DATA PORT (PORT)
- PASSIVE (PASV) This command requests the server-DTP to "listen" on a data port (which is not its default data port) and to wait for a connection rather than initiate one upon receipt of a transfer command.
- REPRESENTATION TYPE (TYPE)
- FILE STRUCTURE (STRU)
- TRANSFER MODE (MODE)

FTP commands

- RETRIEVE (RETR) : This command causes the server-DTP to transfer a copy of the file, specified in the pathname, to the server- or user-DTP at the other end of the data connection.
- STORE (STOR) : This command causes the server-DTP to accept the data transferred via the data connection and to store the data as a file at the server site. If the file specified in the pathname exists at the server site, then its contents shall be replaced by the data being transferred.
- STORE UNIQUE (STOU): This command behaves like STOR except that the resultant file is to be created in the current directory under a name unique to that directory.
- APPEND (with create) (APPE): This command causes the server-DTP to accept the data transferred via the data connection and to store the data in a file at the server site. If the file specified in the pathname exists at the server site, then the data shall be appended to that file
- RENAME FROM (RNFR): This command specifies the old pathname of the file which is to be renamed.
- RENAME TO (RNTO): This command specifies the new pathname of the file specified in the immediately preceding "rename from" command.

FTP Commands

- USER NAME (USER): string identifying the user
- PASSWORD (PASS): string specifying the user's password
- ACCOUNT (ACCT): string identifying the user's account. The command is not necessarily related to the USER command, as some sites may require an account for login and others only for specific access
- CHANGE WORKING DIRECTORY (CWD) This command allows the user to work with a different directory or dataset for file storage or retrieval without altering his login or accounting information.
- CHANGE TO PARENT DIRECTORY (CDUP)
- LOGOUT (QUIT) if file transfer is not in progress, the server closes the control connection. If file transfer is in progress, the connection will remain open for result response and the server will then close it
- REINITIALIZE (REIN) This command terminates a USER, flushing all I/O and account information, except to allow any transfer in progress to be completed. All parameters are reset to the default settings and the control connection is left open. This is identical to the state in which a user finds himself immediately after the control connection is opened.

FTP commands

- ABORT (ABOR): This command tells the server to abort the previous FTP service command and any associated transfer of data.
- DELETE (DELE) : This command causes the file specified in the pathname to be deleted at the server site.
- REMOVE DIRECTORY (RMD) : This command causes the directory specified in the pathname to be removed as a directory
- MAKE DIRECTORY (MKD): This command causes the directory specified in the pathname to be created as a directory
- LIST (LIST): If the pathname specifies a directory or other group of files, the server should transfer a list of files in the specified directory
- PRINT WORKING DIRECTORY (PWD) : This command causes the name of the current working directory

FTP response

- An FTP reply consists of a three digit number (transmitted as three alphanumeric characters) followed by some text.
- 1yz Positive Preliminary reply The requested action is being initiated; expect another reply before proceeding with a new command.
- 2yz Positive Completion reply The requested action has been successfully completed. A new request may be initiated.
- 3yz Positive Intermediate reply The command has been accepted, but the requested action is being held in abeyance, pending receipt of further information. The user should send another command specifying this information.
- 4yz Transient Negative Completion reply The command was not accepted and the requested action did not take place, but the error condition is temporary and the action may be requested again.
- 5yz Permanent Negative Completion reply The command was not accepted and the requested action did not take place. The User-process is discouraged from repeating the exact request (in the same sequence).

More on FTP

- In the control channel it is possible to specify
  - Data Type
    - ASCII: default type, must be accepted by all FTP implementations. It is intended primarily for the transfer of text files.
      - take a second (optional) parameter; this is to indicate what kind of vertical format control, if any, is associated with a file (for printing).
    - EBCDIC: efficient transfer between hosts which use EBCDIC for their internal character representation.
      - take a second (optional) parameter; this is to indicate what kind of vertical format control, if any, is associated with a file (for printing).
    - IMAGE: efficient storage and retrieval of files and for the transfer of binary data
    - Local Type: The data is transferred in logical bytes of the size specified by the obligatory second parameter, Byte size.

More on FTP

- Data structures
  - file-structure, where there is no internal structure and the file is considered to be a continuous sequence of data bytes (default);*
  - record-structure, where the file is made up of sequential records;*
  - page-structure, where the file is made up of independent indexed pages.
    - each with a header (header length, page length, page size, and type-last page/simple page/descriptor page/access control to be specified)

*both must be accepted

Chapter 2: Application layer

- 2.1 Principles of network applications
- 2.2 Web and HTTP
- 2.3 FTP
- 2.4 Electronic Mail
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 P2P applications
- 2.7 Socket programming with UDP
- 2.8 Socket programming with TCP

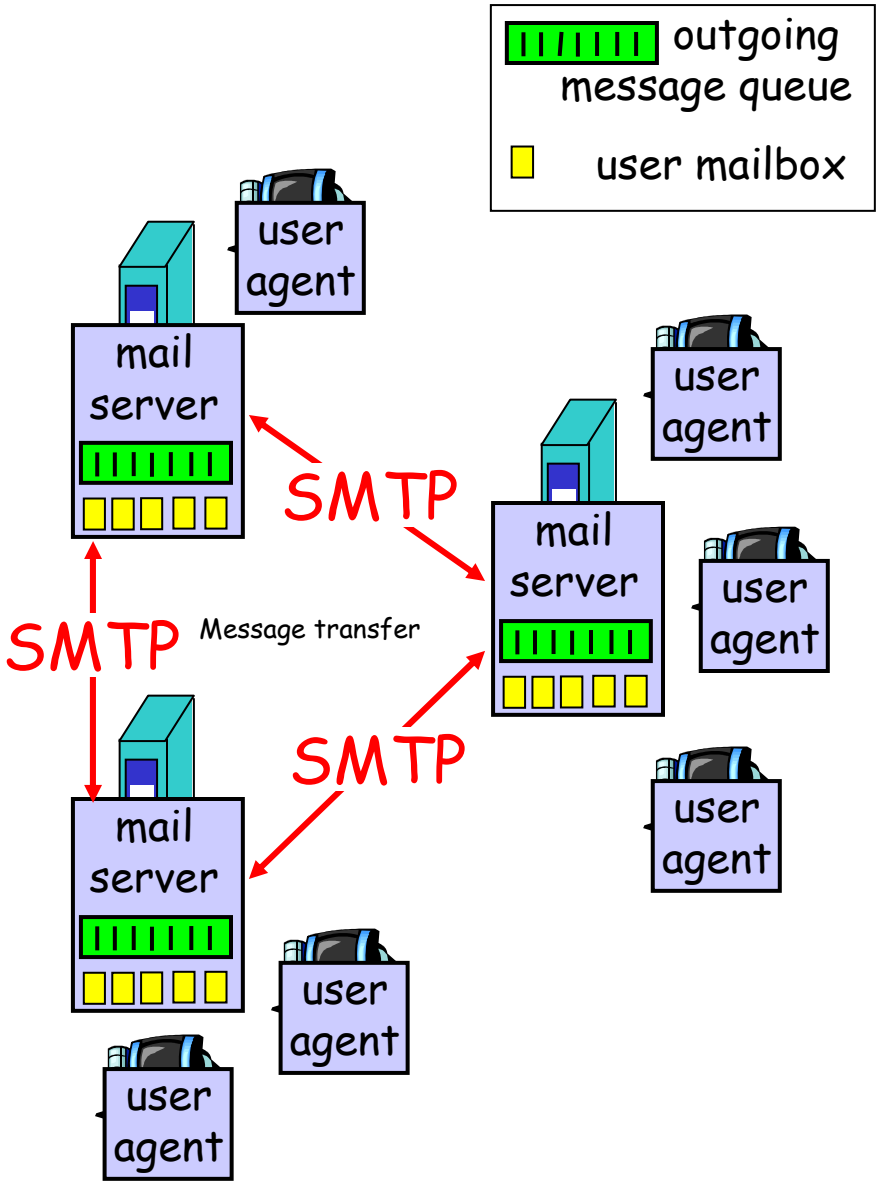
Electronic Mail

Three major components:

- user agents
- mail servers
- simple mail transfer protocol: SMTP

User Agent

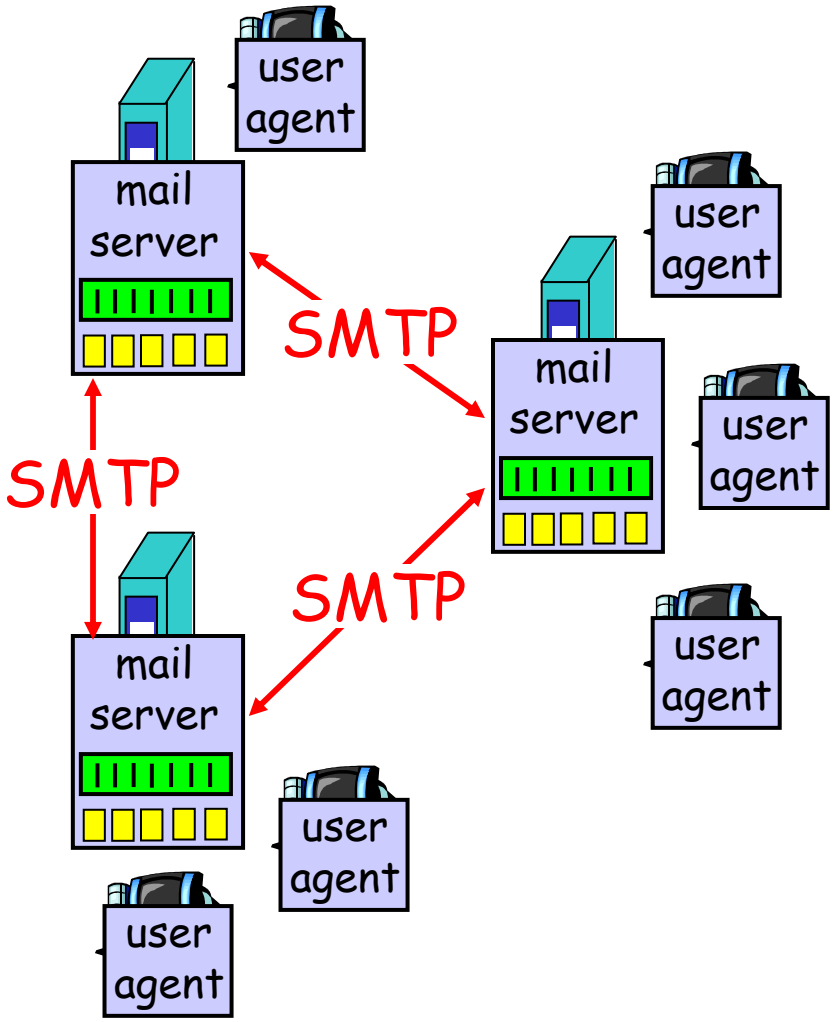
- a.k.a. "mail reader"
- composing, editing, reading mail messages
- e.g., Eudora, Outlook, elm, Mozilla Thunderbird
- outgoing, incoming messages stored on server



Electronic Mail: mail servers

Mail Servers

- mailbox contains incoming messages for user
- message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages
  - client: sending mail server
  - "server": receiving mail server



Email systems basic functions

- Composition (creating messages and answers)
- Tranfer (moving messages from the originator to the recipient)
- Reporting (what happened to the message?)
- Displaying (e.g., invoking special viewers)
- Disposition (e.g., deleting, saving the e-mail etc)

Asynchronous and reliable transfer

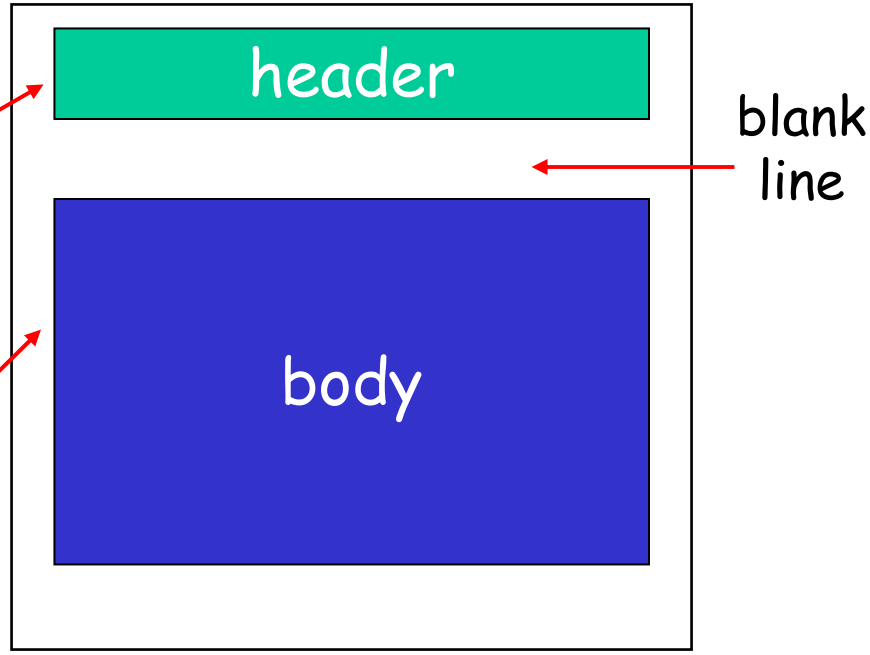
Mail message format

SMTP: protocol for exchanging email msgs

RFC 822: standard for text message format:

- header lines, e.g.,
  - To:
  - From:
  - Subject:

  different from SMTP commands!
- body
  - the "message", ASCII characters only



Formato messaggi (RFC 822)

- Header
  - To:/Cc:/Bcc:/From:/Sender:/Received:/Return-Path fields
    - Received: a line added by each message transfer agent along the way (agent identity/date and time of reception)
    - Return-Path: typically only the sender address
  - Date:/Reply-To:/Message-Id:/In-Reply-To:/References:/Keywords:/Subject:
- Body
  - Nowadays messages in different languages (with and without accents, different alphabets, or in languages with no alphabet), multimedia messages info included.

MIME (Multipurpose Internet Mail Extensions) RFC 1521

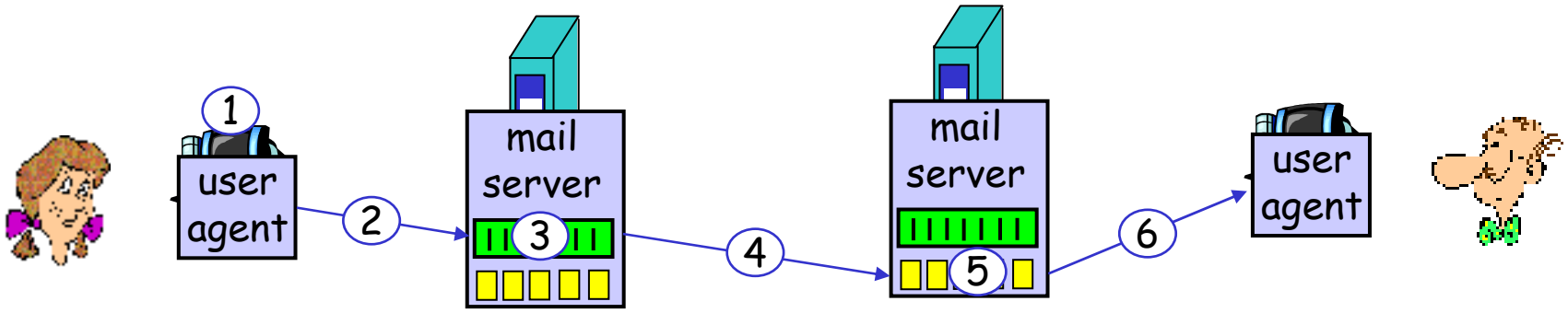
- structures the message body
- New message headers:
  - MIME-Version/Content-Description:/Content-ID:/Content Transfer Encoding:
  - Content-Type:
    - Text/Image/Audio/Video/Application (e.g. Postcript subtype), Multipart

Electronic Mail: SMTP [RFC 2821]

- uses TCP to reliably transfer email message from client to server, port 25
- direct transfer: sending server to receiving server
- three phases of transfer
  - handshaking (greeting,)
  - transfer of messages
  - closure
- command/response interaction
  - commands: ASCII text
  - response: status code and phrase
- messages must be in 7-bit ASCII

Scenario: Alice sends message to Bob

1) Alice uses UA to compose message and "to" `bob@someschool.edu`
2) Alice's UA sends message to her mail server; message placed in message queue
3) Client side of SMTP opens TCP connection with Bob's mail server
4) SMTP client sends Alice's message over the TCP connection
5) Bob's mail server places the message in Bob's mailbox
6) Bob invokes his user agent to read message



Sample SMTP interaction

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

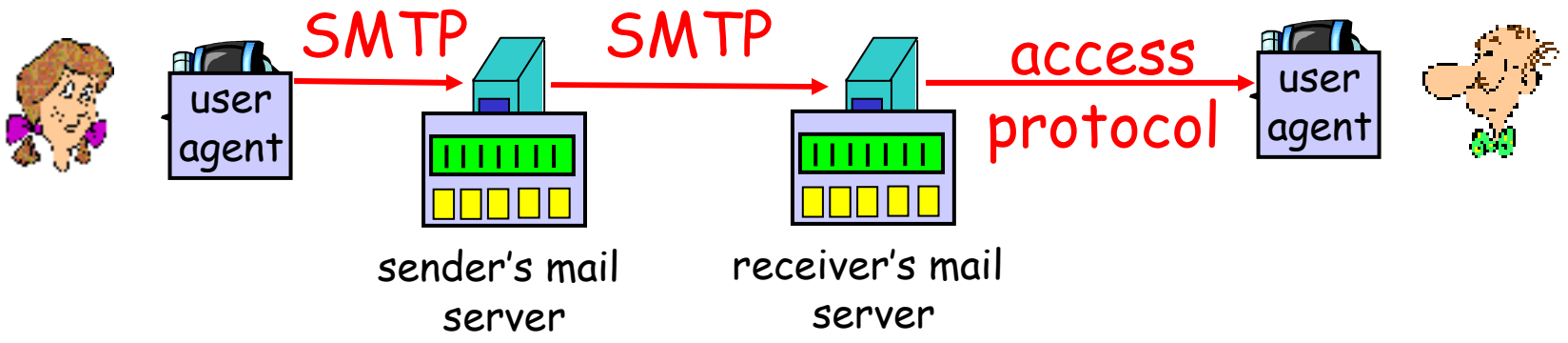
SMTP: final words

- SMTP uses persistent connections
- SMTP requires message (header & body) to be in 7-bit ASCII
- SMTP server uses `CRLF.CRLF` to determine end of message

Comparison with HTTP:

- HTTP: pull
- SMTP: push
- both have ASCII command/response interaction, status codes
- HTTP: each object encapsulated in its own response msg
- SMTP: multiple objects sent in multipart msg

Mail access protocols



- SMTP: delivery/storage to receiver's server
- Mail access protocol: retrieval from server
  - POP: Post Office Protocol [RFC 1939]
    - authorization (agent <-->server) and download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - more features (more complex)
    - manipulation of stored msgs on server
  - HTTP: gmail, Hotmail, Yahoo! Mail, etc.

POP3 protocol

authorization phase

- client commands:
  - `user:` declare username
  - `pass:` password
- server responses
  - `+OK`
  - `-ERR`

transaction phase, client:

- `list:` list message numbers
- `retr:` retrieve message by number
- `dele:` delete
- `quit`

```
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

POP3 (more) and IMAP

More about POP3

- Previous example uses "download and delete" mode.
- Bob cannot re-read e-mail if he changes client
- "Download-and-keep": copies of messages on different clients
- POP3 is stateless across sessions

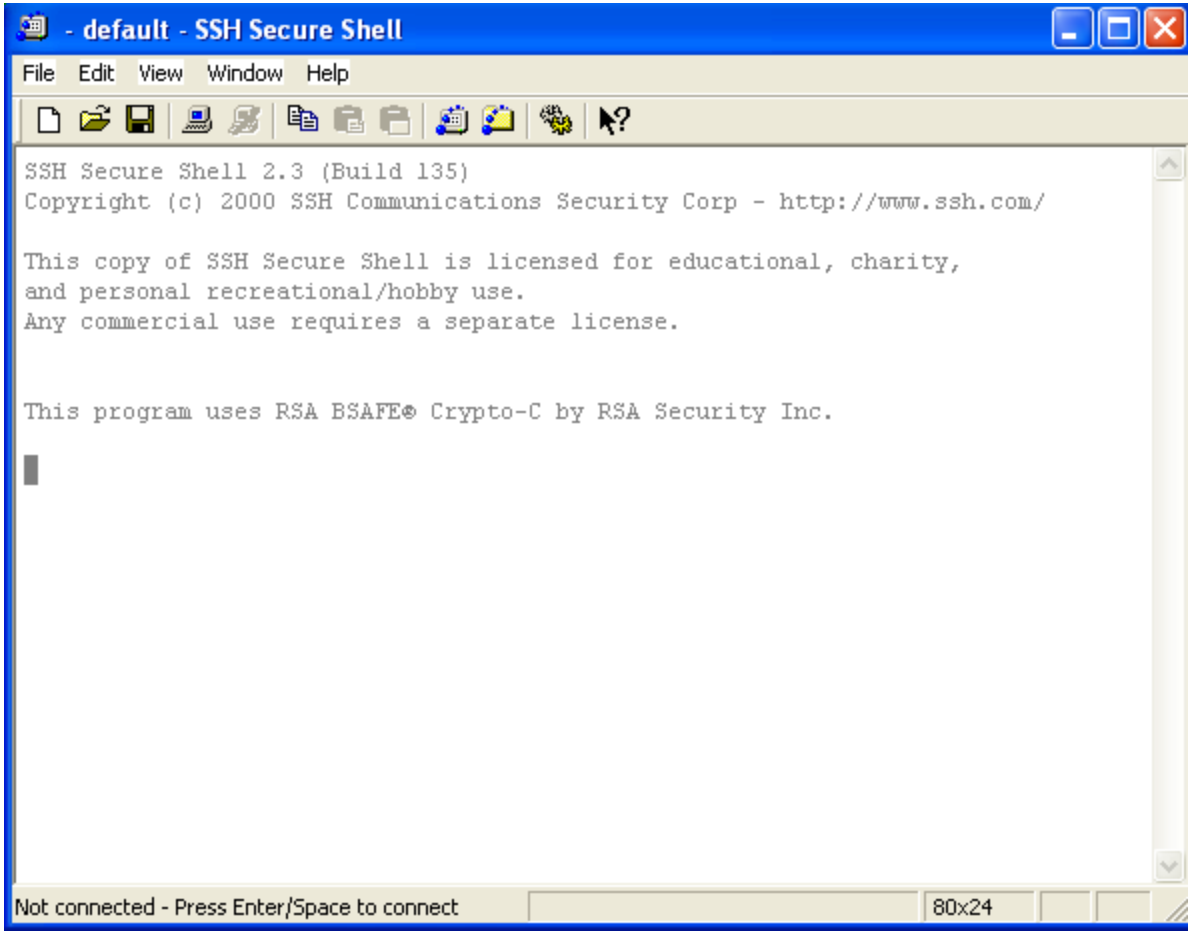
IMAP (Interactive Mail Access Protocol) RFC 1064

- Keep all messages in one place: the server
- Allows user to organize messages in folders
- IMAP keeps user state across sessions:
  - names of folders and mappings between message IDs and folder name

Chapter 2 outline

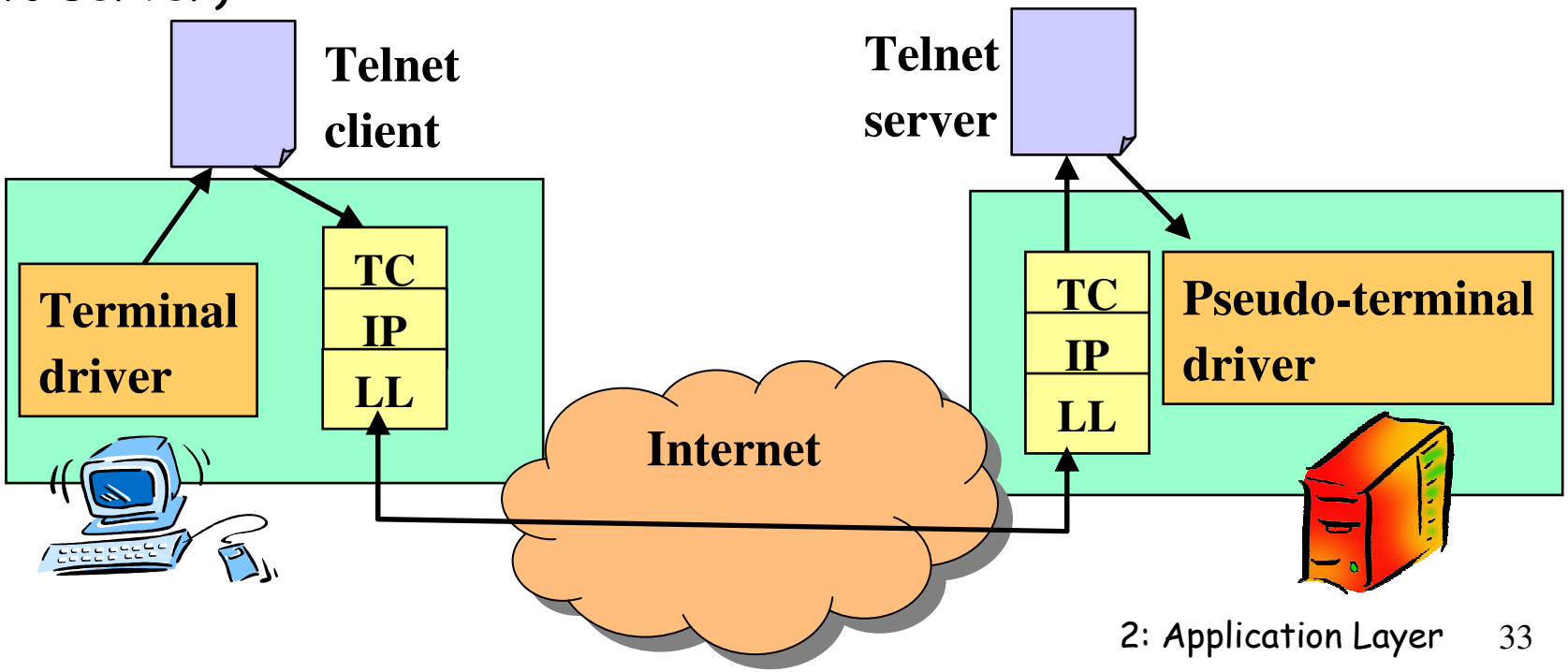
- 2.1 Principles of app layer protocols
  - clients and servers
  - app requirements
- 2.2 Web and HTTP
- 2.3 FTP
- 2.4 Electronic Mail
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.9 Content distribution
  - Network Web caching
  - Content distribution networks
  - P2P file sharing
- Telnet

TELNET (TErminaL NETwork)



TELNET (TErminaL NETwork)

- E' un semplice applicativo che consente ad un programma su un host (TELNET client) di accedere alle risorse su un host remoto (TELNET server) come se il client fosse un terminale (testuale) attaccato all'host remoto
- Un applicativo TELNET agisce da client o server a seconda che contatti o sia contattato
- I comandi sono trasferiti su una connessione TCP (well-know porta 23 lato server)



TELNET: Basic Ideas

- Network Virtual Terminal
  - Imaginary device having a basic structure common to a wide range of real terminals→guarantees the colloquium follows clear rules and formats independent of the real terminal features
    - E.g., Data represented in 7 bit ASCII transmitted in 8-bit bytes
    - NVT half-duplex device operating in line-buffered mode (the characters of a line are buffered and a complete line is transmitted together)
    - Local echo support provided
  - NTV made of: a printer (or display) and a keyboard. The keyboard produces the data transmitted over the TCP connection; the printer receives and displays the data.
- Negotiation of terminal options
  - Two hosts can negotiate different options to extend the NVT capabilities to reflect the capabilities of the real HW in use (at the beginning of the colloquium or dynamically during it)
- Symmetric view of terminals and processes
    - Being a client or server may only depend on who is contacing whom

Terminal options

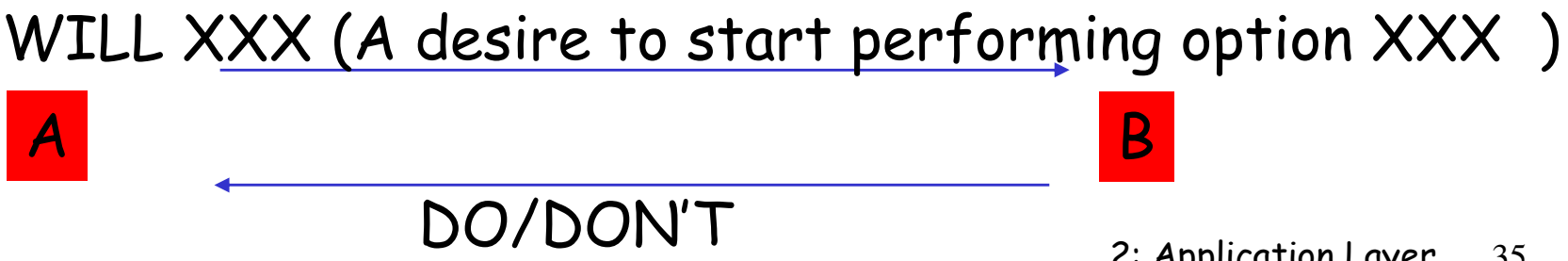
- Terminal options: kind of transmission (e.g. binary), window size, terminal type, line width, etc.
- How to negotiate them?
  - Usage of DO/DON't/WILL/WON'T followed by option code
  - Es (CLIENT). DO Transmit binary (requests that the other party use binary transmission)

    Reply: WILL Transmit binary (OK),

    WON'T Transmit binary (NO)

    Es2 (CLIENT) WILL Transmit binary (desire to use)

    Reply: DO Transmit binary(OK, I expect you to do so)



Terminal options

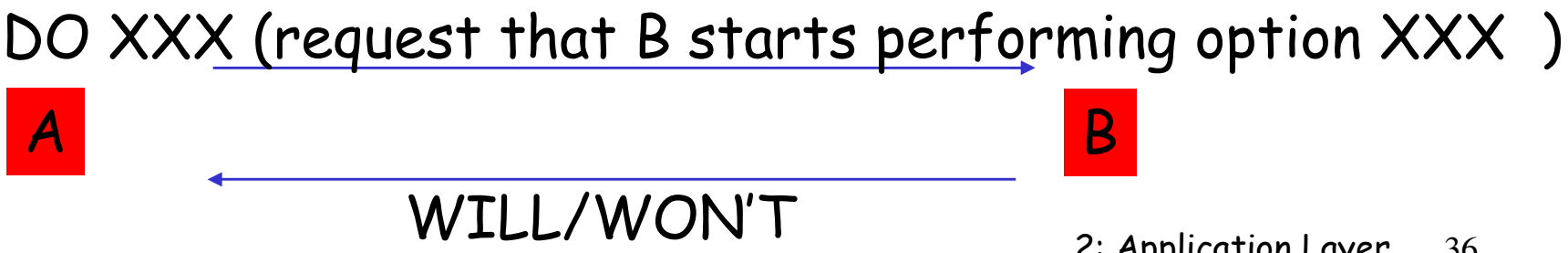
- Terminal options: kind of transmission (e.g. binary), window size, terminal type, line width, etc.
- How to negotiate them?
  - Usage of DO/DON't/WILL/WON'T followed by option code
  - Es (CLIENT). DO Transmit binary (requests that the other party use binary transmission)

    Reply: WILL Transmit binary (OK),

    WON'T Transmit binary (NO)

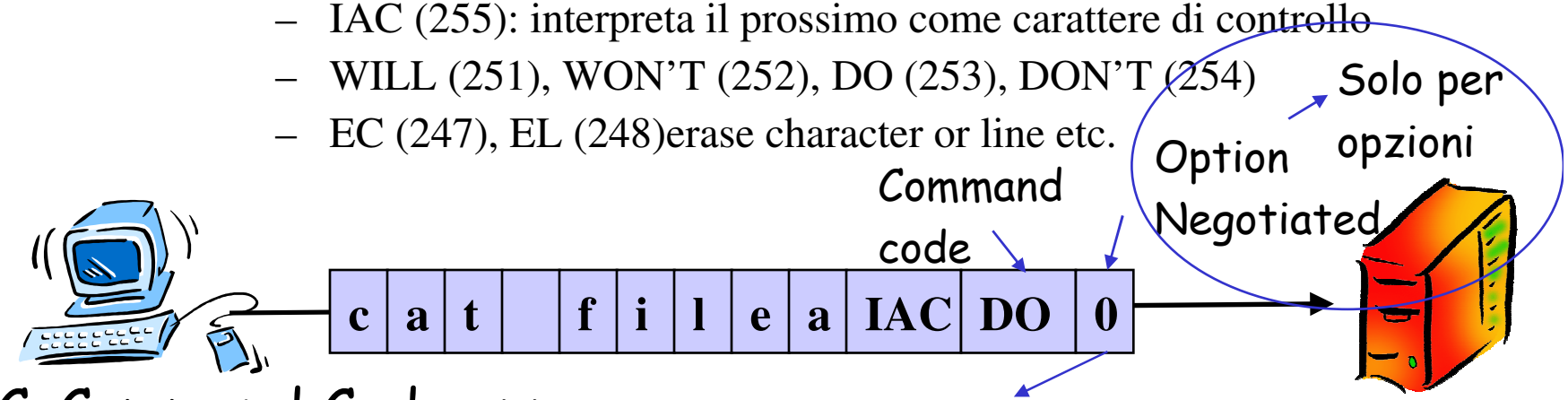
    Es2 (CLIENT) WILL Transmit binary (desire to use)

    Reply: DO Transmit binary(OK, I expect you to do so)



TELNET (TErminaL NETwork)

- TELNET trasferisce caratteri
  - Caratteri dati:
    - Sono caratteri ASCII con il primo bit pari a 0
    - Si possono trasferire anche caratteri ASCII con il primo bit pari a 1 se li si fa precedere da un byte di controllo speciale
  - Caratteri di controllo:
    - Sono comandi codificati in sequenze di 8 bit con il primo pari ad 1
    - Tra questi
      - IAC (255): interpreta il prossimo come carattere di controllo
      - WILL (251), WON'T (252), DO (253), DON'T (254)
      - EC (247), EL (248)erase character or line etc.



IAC+Command Code oppure

IAC+Command Code+Option

Telnet options (examples)

- 0 Binary Transmission
- 1 Echo
- 17 Extended ASCII
- 37 Authentication Option
- 38 Encryption Option
- 45 Telnet Suppress Local Echo

- more http://www.iana.org/assignments/telnet-options

Come leggere uno standard

- Lo standard RFC di FTP disponibile sul sito (in verde aspetti piu' importanti quindi vari livelli di dettaglio crescente: celeste e giallo)
- Vari livelli di lettura: primo pass per capire il funzionamento (negli standard anche molte info di dettaglio sin dall'inizio che non potete capire/che non vi interessano). NON come un libro: cio' che non capite potrebbe essere spiegato chiaramente pagine dopo. Piu' pass per una comprensione totale.
- Distinguere le info necessarie per una prima comprensione da info di dettaglio che fanno esempi/specificano meglio alcuni aspetti da info di estremo dettaglio (implementazioni in alcuni sistemi, formati etc.) che potrebbero non interessarvi affatto!!
- Dopo aver compreso con una lettura delle parti importanti il funzionamento tornate sugli argomenti di dettaglio e li rileggete con piu' attenzione se e solo se vi serve quel livello di dettaglio (un unico standard risponde a diverse esigenze: studente, implementatore, etc.!!)

Lo stack di Internet