Application Layer

Internet Electronic Mail

Unit 2: Application layer

- 2.1 Distributed Applications
- 2.2 Web and HTTP
- 2.3 FTP and TFTP
- 2.4 Electronic Mail
  - SMTP, POP3, IMAP
- 2.5 Domain Name Service
- 2.6 Socket Programming

Internet Electronic mail

- 1982, first standards
  - RFC 821 y RFC 2821, protocol for tranfer of mail messages
  - RFC 822, messages format
- Basic services
  - Composition of messages and responses
  - Message transfer
  - Notification to the sender of status of sent messages
  - Presentation and disposition
- Advanced services
  - Forwarding , automatic reply, mail lists, secret mail, high priority mail, hidden copies, alternative receivers

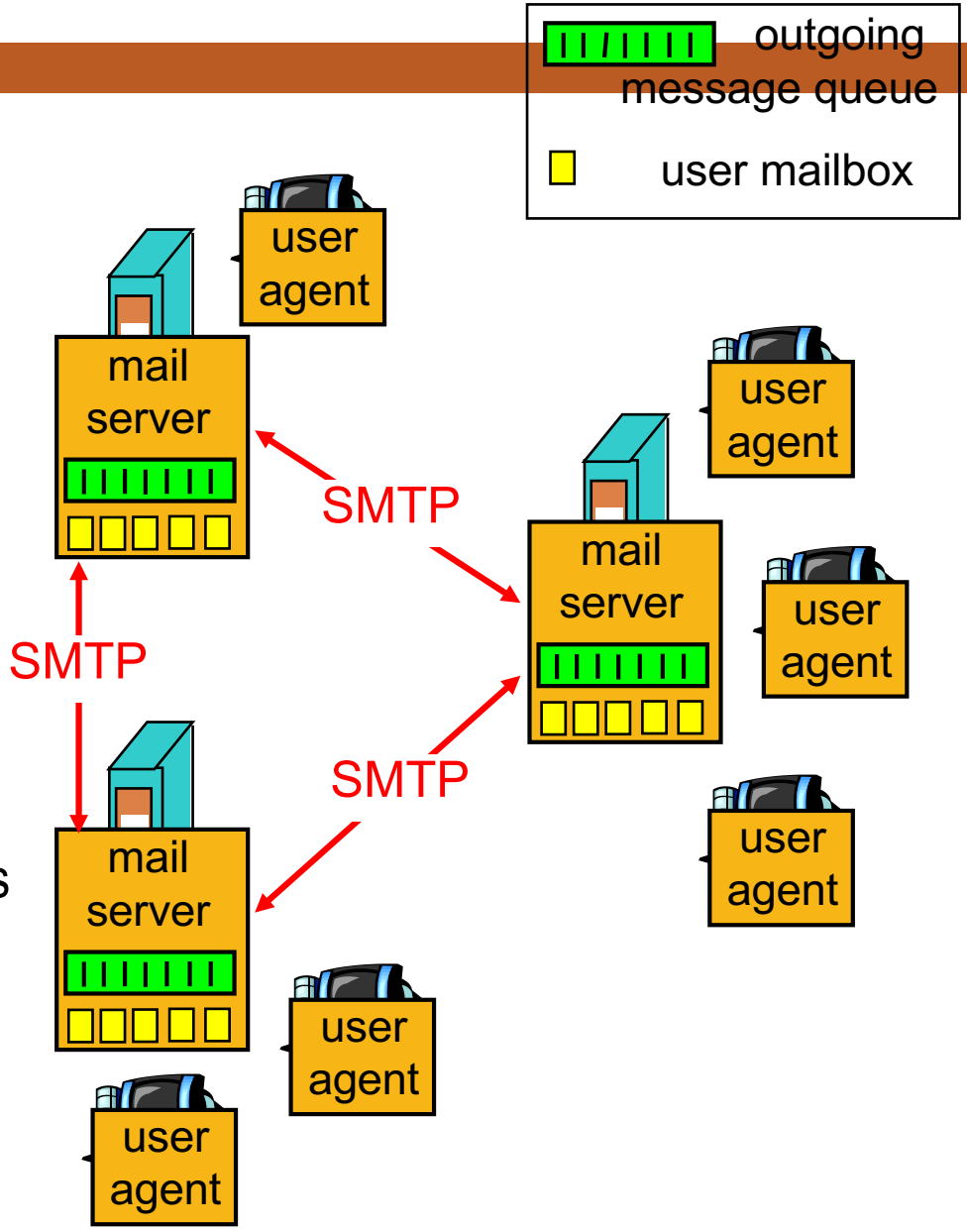
Electronic Mail

Three major components:

- user agents
- mail servers
- mail transfer protocol: SMTP
- mail access protocols: POP3, IMAP, HTTP

User Agent

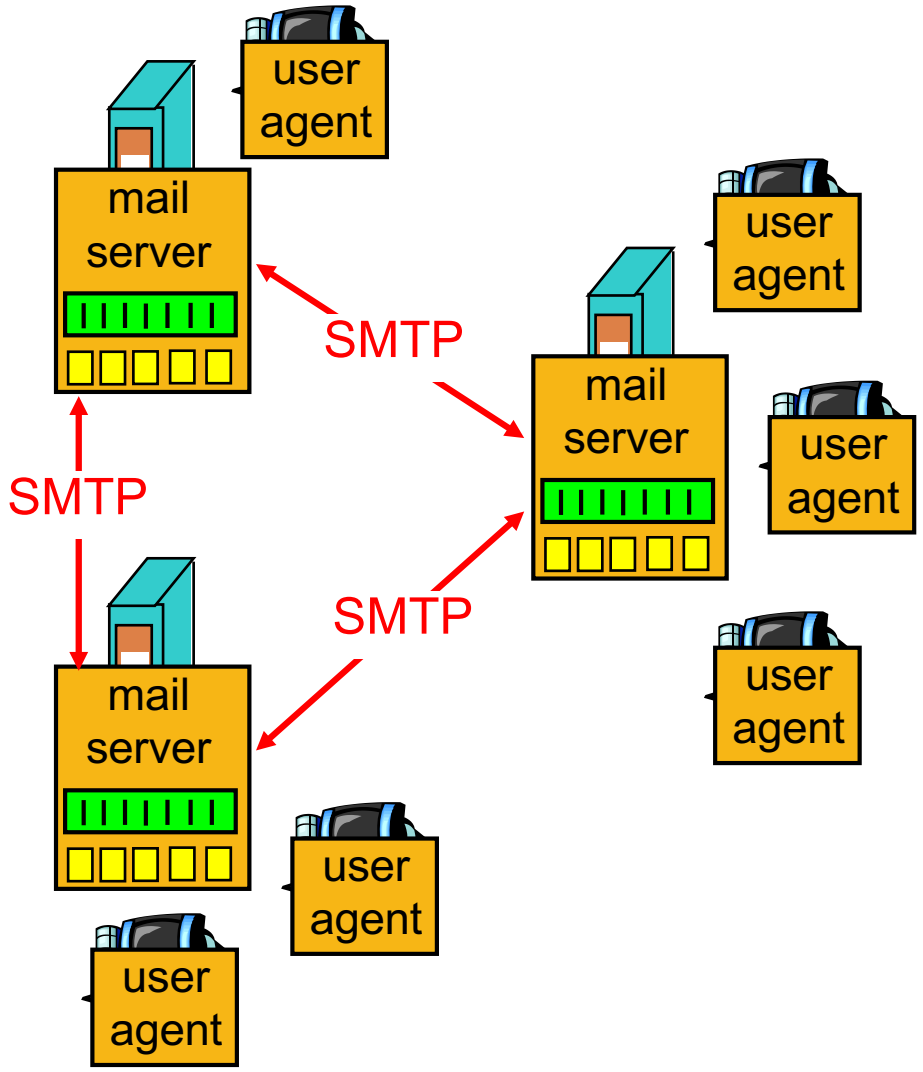
- a.k.a. “mail reader”
- composing, editing, reading mail messages
- e.g., Mail (Mac OS X), Outlook, Mozilla Thunderbird
- Outgoing, incoming messages stored on server



Electronic Mail: mail servers

Mail Servers

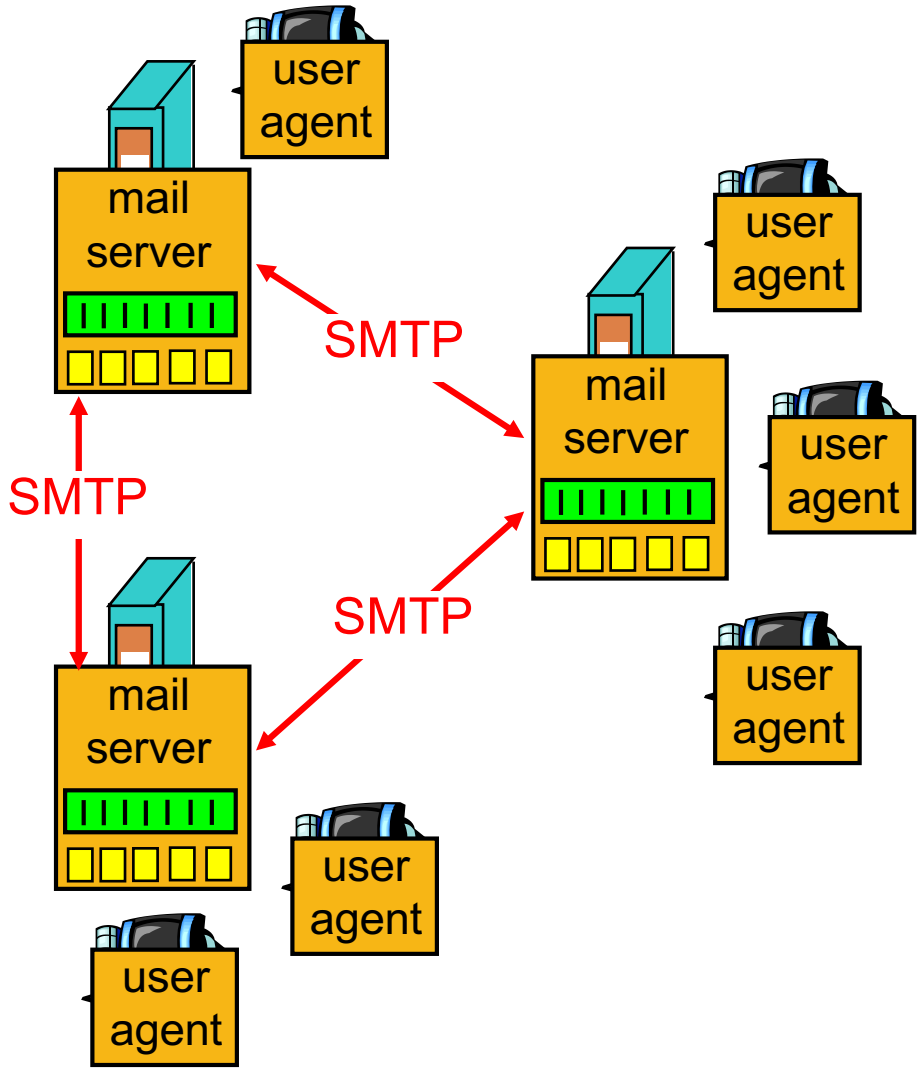
- mailbox contains incoming messages for user
- message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages (mail transfer)
  - client: sending mail server
  - "server": receiving mail server



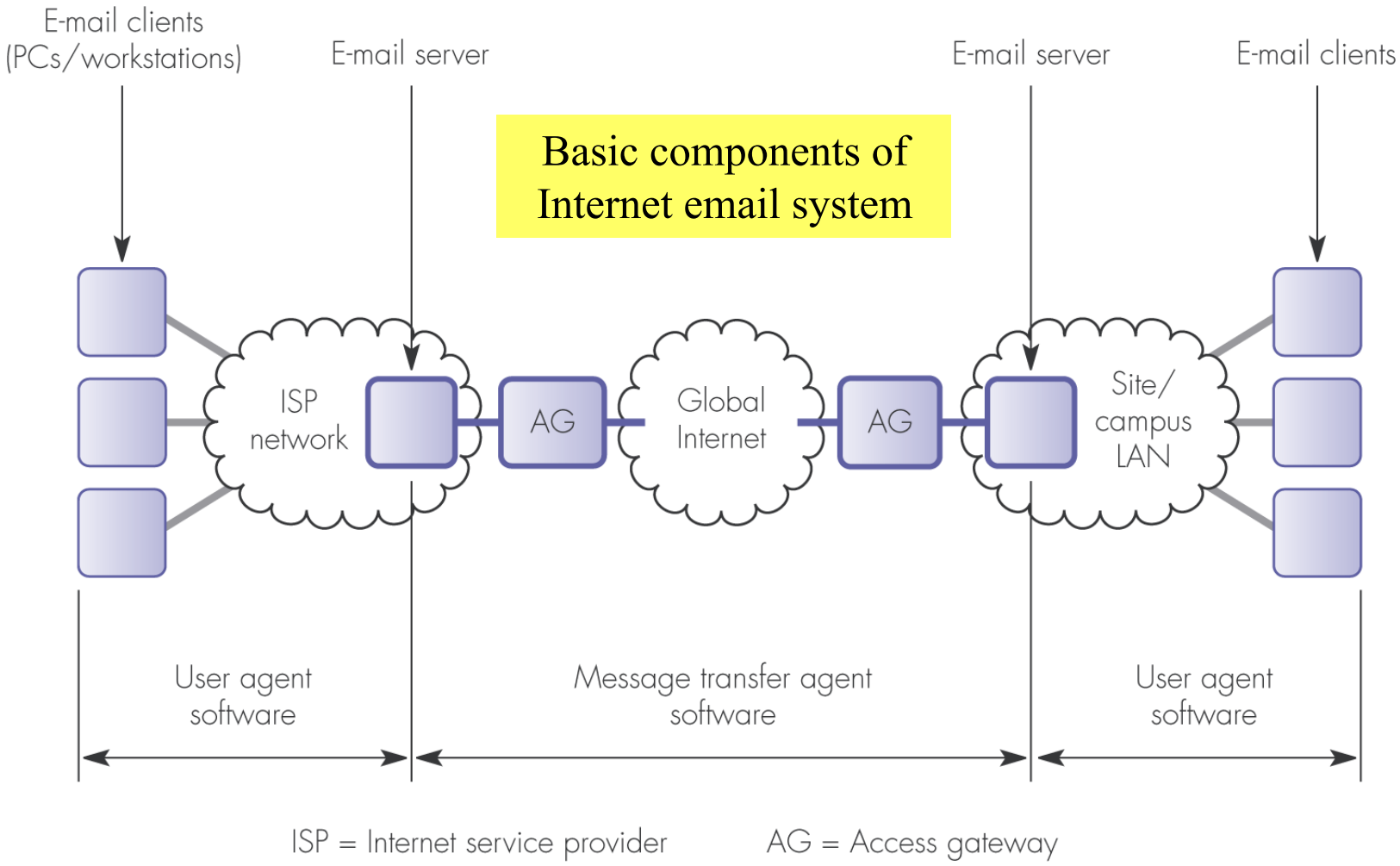
Electronic Mail: mail servers

Mail Servers

- Communication software:
  - User Agent Server
    - To interact with UA Client
  - Message Transfer Agent
    - For message transfer between mail servers



Internet electronic mail



Halsall, *Computing Networking and the Internet,* 5th Edition © Pearson Education Limited 2005

- Processes for sending email
  - Two processes, one performed by the UA and the other by the MTA
    - By the UA:
      - Creating the message
      - Entering the message headers to content
      - Transfer the message to the MTA
    - By the MTA:
      - Message is encapsulated in an "electronic envelope"
        - Obtained from headers "From" and "To" Post
      - Message transfer through Internet
        - To another MTA

Message Transfer

- UA client
  - Converts message to 7-bit ASCII format, and sends it to the UA server
- UA server
  - Deposits message at queue of outgoing mail server
- MTA Client
  - Periodically checks message queue, and if there is any message, attempts to send it
  - Envelope is created "on" delivery
  - From "From" and "To" headers
  - One, or as may be required (if sending copies)
  - Format of email addresses:
    - *user-name @ name-of-server*

Domain name of mail server

Message Transfer (cont'd)

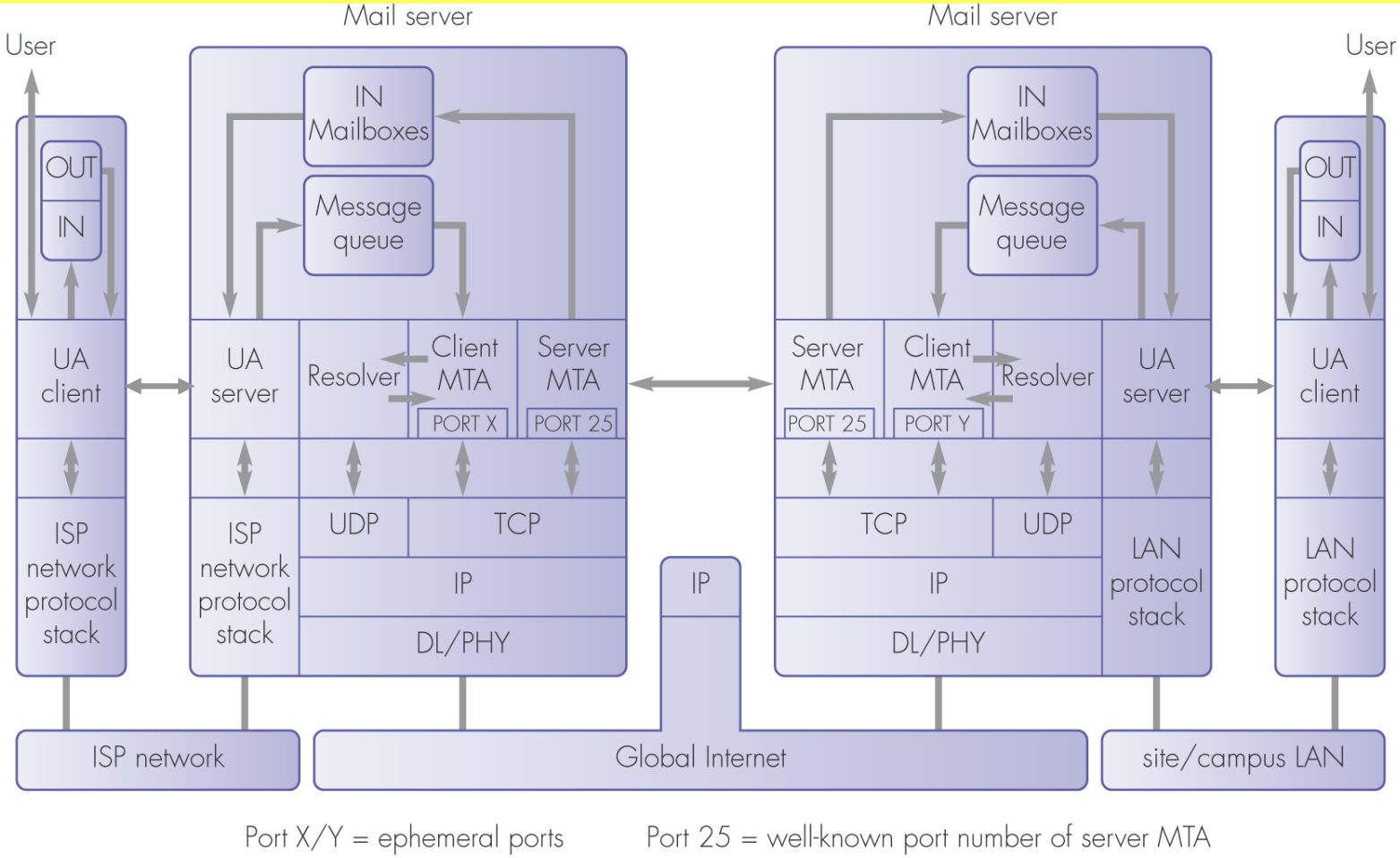
- MTA Client (continued):
  - Gets IP address of the recipient mail server
    - Through the intervention of the "Resolver"
  - Tries to establish TCP connection with recipient mail server
    - With the MTA server, the default port 25
  - TCP connection established (accepted) between mail servers
    - Originating server (Client) expects that target server (server) will send identity and indicate willingness to receive emails
      - Is not available, connection is released and subsequent retries
        - Further retries depend on configuration of originating server
    - Then it is willing ...

Message Transfer (cont'd)

- Server MTA (destination server) ready to receive emails
  - Server MTA checks recipient, if so, agrees to receive the message
  - Client MTA sends message, MTA server returns acknowledgment
  - If there are more messages to other recipients
    - The process is repeated
  - The open TCP connection is used to transfer messages towards the source server that created the connection
    - If there were any messages
- When both servers end the exchange of messages
  - TCP connection between Mail Servers is released

Internet Electronic mail

Components of email message transfer system



Halsall, *Computing Networking and the Internet,* 5th Edition © Pearson Education Limited 2005

SMTP: Simple Mail Transfer Protocol

- SMTP
  - Client - server model
  - Basic protocol for email
    - Main Application Layer protocol for Internet mail
  - Protocol of type: "offer"
    - Push protocol
    - HTTP protocol is an "on demand" (pull protocol)
  - It uses persistent TCP connections
- SMTP messages
  - Client sends "commands" in text mode, and
  - Server responds with "status codes"
    - And optionally a text
  - Everything in 7-bit ASCII readable!!!!

SMTP: Simple Mail Transfer Protocol

1
- S: 220 unsitio.es
- C: HELO otro.com
- S: 250 Hello otro.com, pleased to meet you

2
- C: MAIL FROM: <elena@otro.com>
- S: 250 elena@otro.com... Sender ok
- C: RCPT TO: <juan@unsitio.es>
- S: 250 juan@unsitio.es ... Recipient ok

3
- C: DATA
- S: 354 Enter mail, end with "." on a line by itself
- C: Que tal estas?
- C: nos vemos en el cine
- C: Elena
- C: .
- S: 250 Message accepted for delivery

4
- C: QUIT
- S: 221 unsitio.es closing connection

Example of SMTP interaction

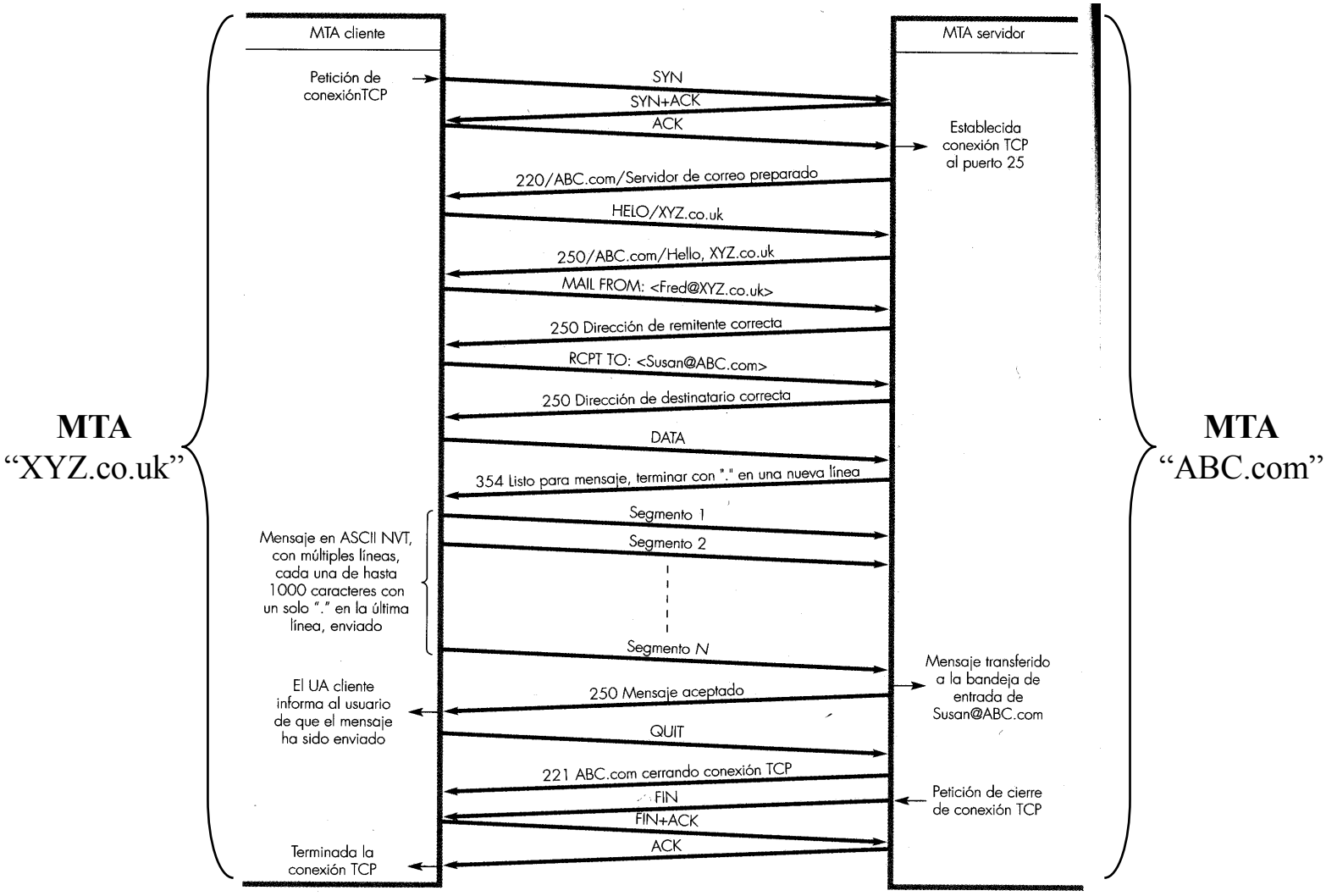
Cliente SMTP: "otro.com"

Servidor SMTP: "unsitio.es"

SMTP phases:

1. Presentation
2. Negociation
3. Transfer
4. Bye

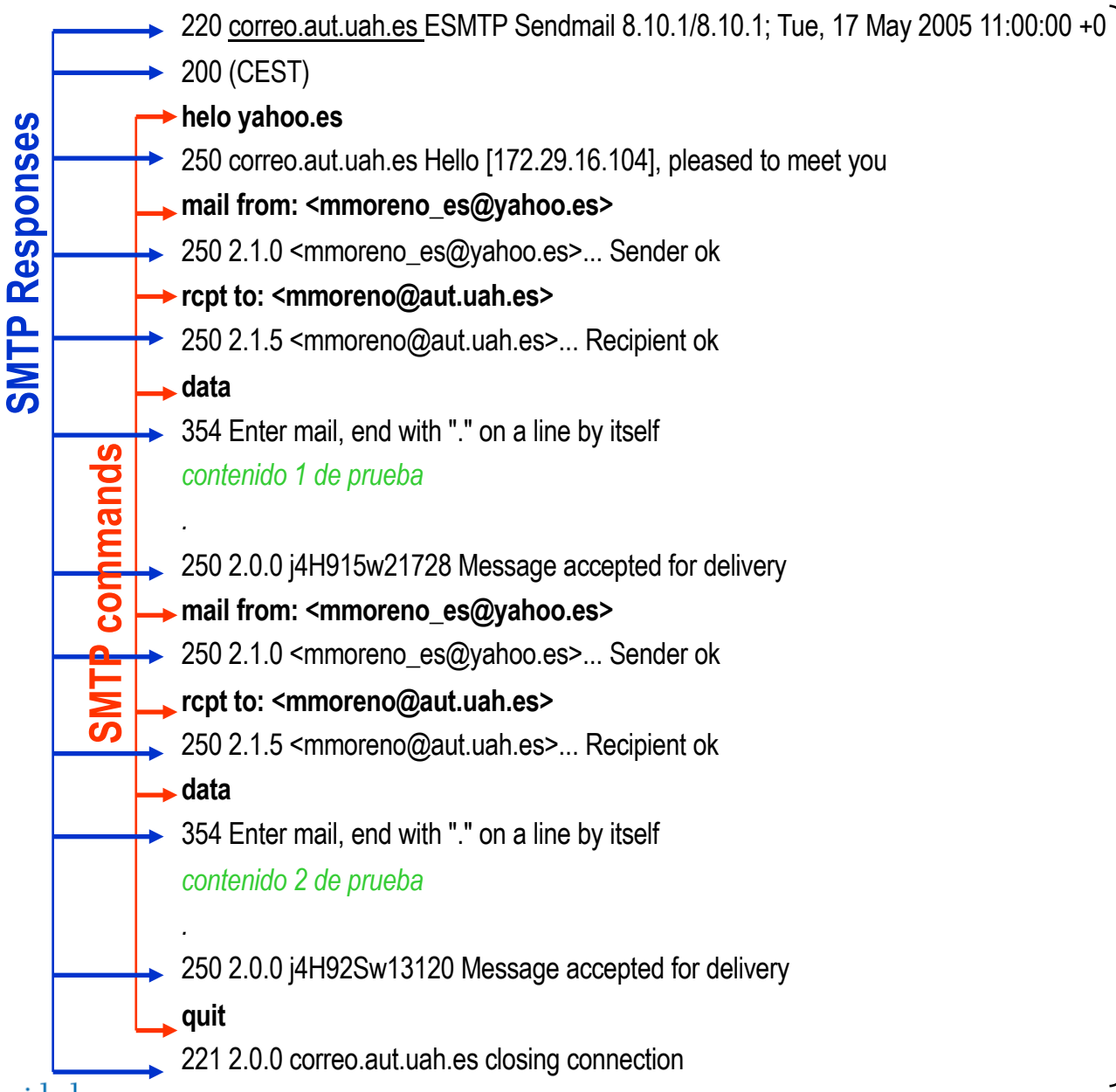
SMTP: Simple Mail Transfer Protocol



SMTP: Simple Mail Transfer Protocol

- It is possible to send messages without a mail UA
  - Using SMTP commands in direct dialog with mail server
  - In its “language”
- Try your own SMTP interaction
  - With TELNET to port 25. e.g:
    - telnet correo.uah.es 25
  - Mail server should respond :
    - “220 reply from server”
    - And some additional text
  - Introduce SMTP command
    - HELO, MAIL FROM, RCPT TO, DATA, QUIT

SMTP: Simple Mail Transfer Protocol



Commands window of a user with an SMTP session via Telnet to mail server "correo.aut.uah.es"

Set of Protocols for Internet mail

- Protocols involved in sending Internet email messages
  - UA → Mail server: SMTP
  - Between Mail servers: SMTP
  - Mail server → UA: POP3, IMAP or HTTP

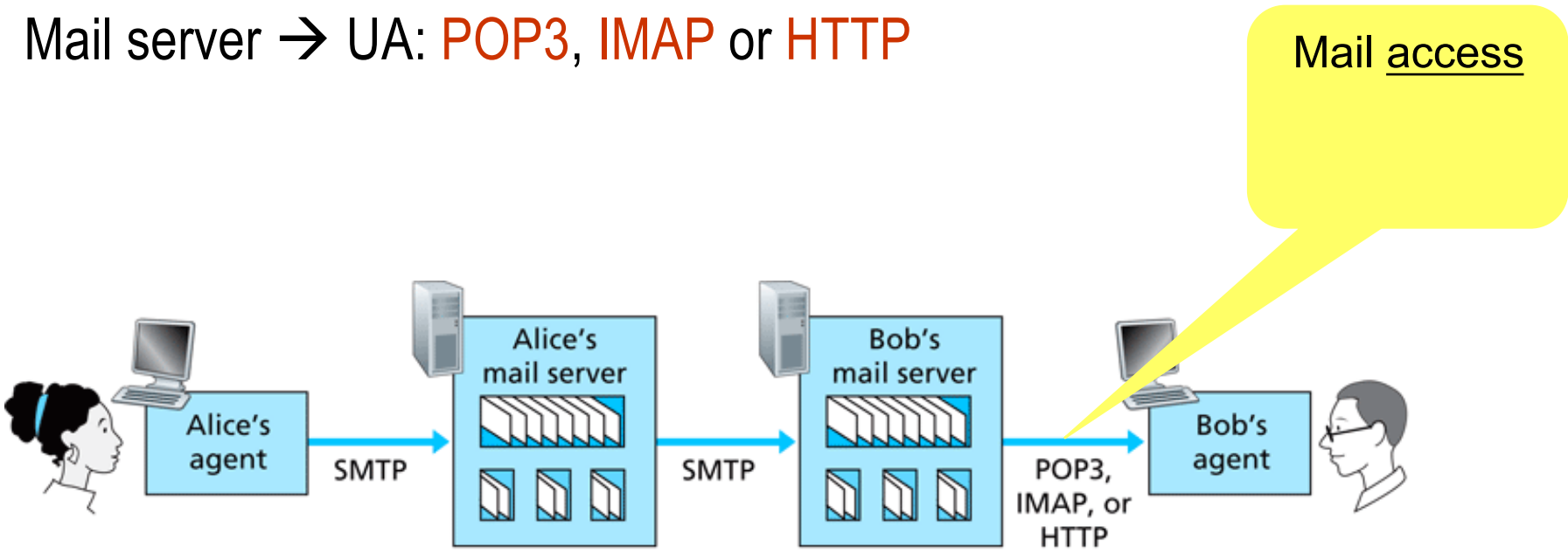


Figure 2.18 ♦ E-mail protocols and their communicating entities

It can happen, but is not frequent, that more than two mail servers mediate between sender and receiver UA´s

Mail Access Protocols

- Mail access protocols
  - Needed to get the messages from mail server
    - Transfer mails from Server to UA
  - SMTP not valid
    - It is an “offer” type protocol (PUSH)
    - User system should be always on and connected
    - Mail server configuration more complex
  - Solution ➔ “Demand” protocols (PULL)
    - POP3 (RFC 1939)
      - Post Office Protocol v.3
    - IMAP (RFC 3501)
      - Internet Mail Access Protocol
    - HTTP

Mail Access Protocols: POP3

- Client-Server Model
- Supported on TCP, port 110
- Text-mode protocol
  - Commands and responses using ASCII codes (7 bit)
- Very simple protocol, limited features, two operation modes:
  - Download and delete
  - Download and keep

Mail Access Protocols: POP3

- Implements **"offline mail processing"**
  - All mail processing is performed at user systems, where the UA resides
  - Protocol:
    - Downloads messages from server, according to UA configuration:
      - Automatic or manual mode
      - Download and delete or download and keep
    - Stores messages at user system, for later processing
    - Then, it disconnects from Server
- Suitable for user accessing mail from a unique machine
- Requires few resources at Server
  - State information at Server
    - No state between sessions ➔ Much simpler!!!
      - No status information stored from one session to next
    - Some state stored during session

Mail Access Protocols: POP3

- POP3 Meesages:
  - Commands
    - **user**: to declare user
    - **pass**: password

    } Authorization Phase

    - **list**: list messages at mail Server
    - **retr**: retrieve a message
    - **dele**: delete a message
    - **quit**: quit POP3 session

    } Transaction Phase

  - Responses
    - **+OK**
    - **-ERR**

    } During Authorization and Transaction phases

  - Update Phase, only at mail Server
    - After POP3 session ends

All in readable text format!!!

Mail access protocols: POP3

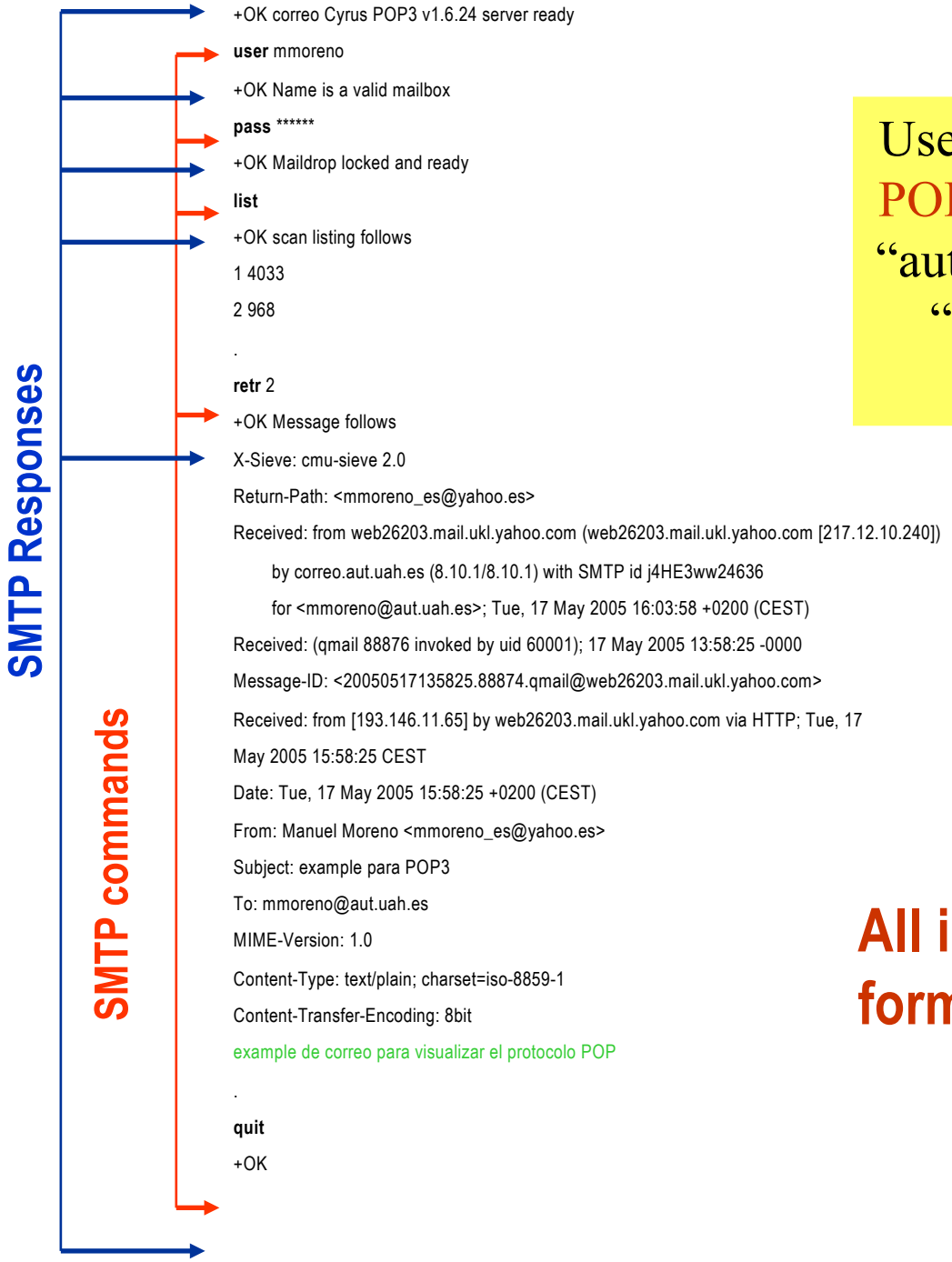
Authorization Phase

```
S: +OK POP3 server ready
C: user juan
S: +OK
C: pass juanpass
S: +OK user successfully logged on
```

Transaction Phase

```
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 2 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

POP3 session. Generic example of interaction between Client and server in "download and delete" mode



User command screen of a POP3 session via Telnet to "aut.uah.es" mail Server in "download and keep" mode

All in readable text format!!!

Mail Access Protocols: IMAP

- Client-Server Model
- Supported on TCP, port 143
- Provides UA at the mail Server
- Stores state information
  - During and between sessions
- Suitable for nomadic users
  - That access mail from multiple end systems
- Enables three access modes to mail:
  - “Offline”
  - “Online”
  - “Disconnected”

Mail Access Protocols: IMAP

- Offline Access
  - Similar to POP3
  - Minimizes:
    - Resources usage at server
    - Connection time
      - Suitable to access via telphone modem
- Online Access
  - “On line” interaction between Client UA and mail Server
    - “On line” mail processing
    - Requires open connection with Server to read mail
    - Messages are kept at server
      - Temporary copy at local host

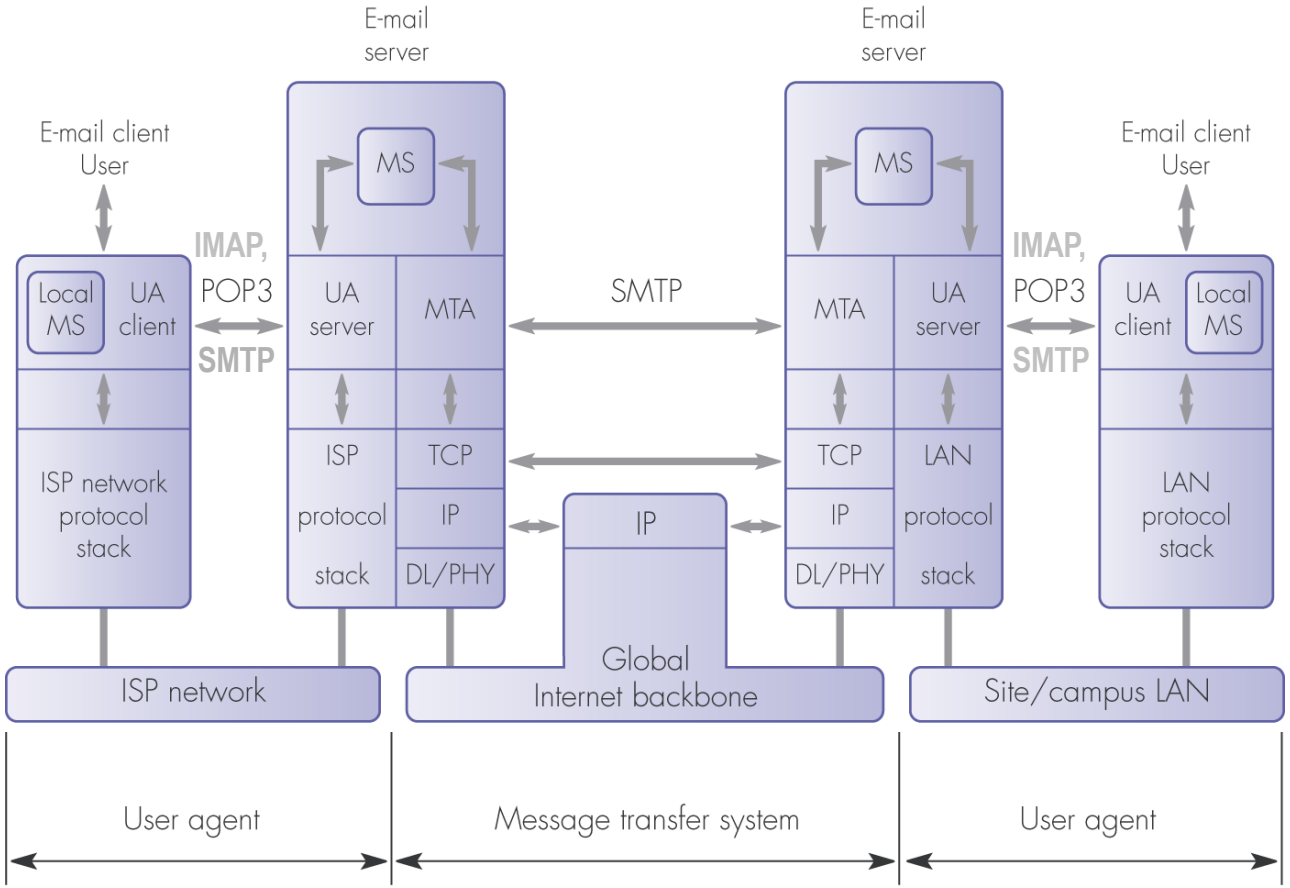
Mail Access Protocols: IMAP

- "Disconnected" access
  - "Off line" mail processing
  - Does NOT requires connection open with Server to read mail
  - Messages are stored temporarily at client cache
    - The are synchronized (updated) at next connection to mail Server
  - Messages are stored at Server
  - Suitable for POTS (telephone) connections
- IMAP:
  - Text mode in readable ASCII 7 bit!!!

Mail Access Protocols: IMAP

- Another usage posibilities:
  - Download of mails that satisfy some conditions, e.g.:
    - Messages from a specific sender
    - Messages with specific contents
    - Only some parts of messages (for multipart MIME messages)
  - Read only headers of messages
  - Others
- Mail is centralized
  - Back up of mails is easier
- Drawbacks
  - More resources needed at mail server
    - More storage and processing capacity
  - More complex software, both at Client and Server

Internet electronic mail protocols: transfer and access

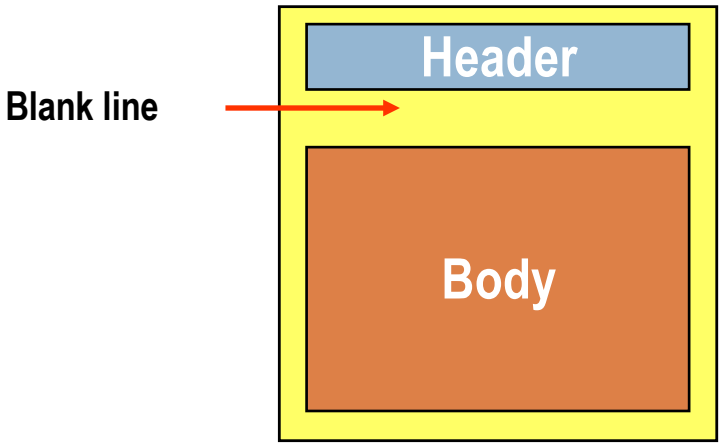


POP = post office protocol
SMTP = simple mail transfer protocol
ISP = Internet service provider
MTA = message transfer agent
UA = user agent
DL/PHY = data link/physical layer

MS = message store (contains a queue of mail to be sent and an IN mailbox for each of its local users)

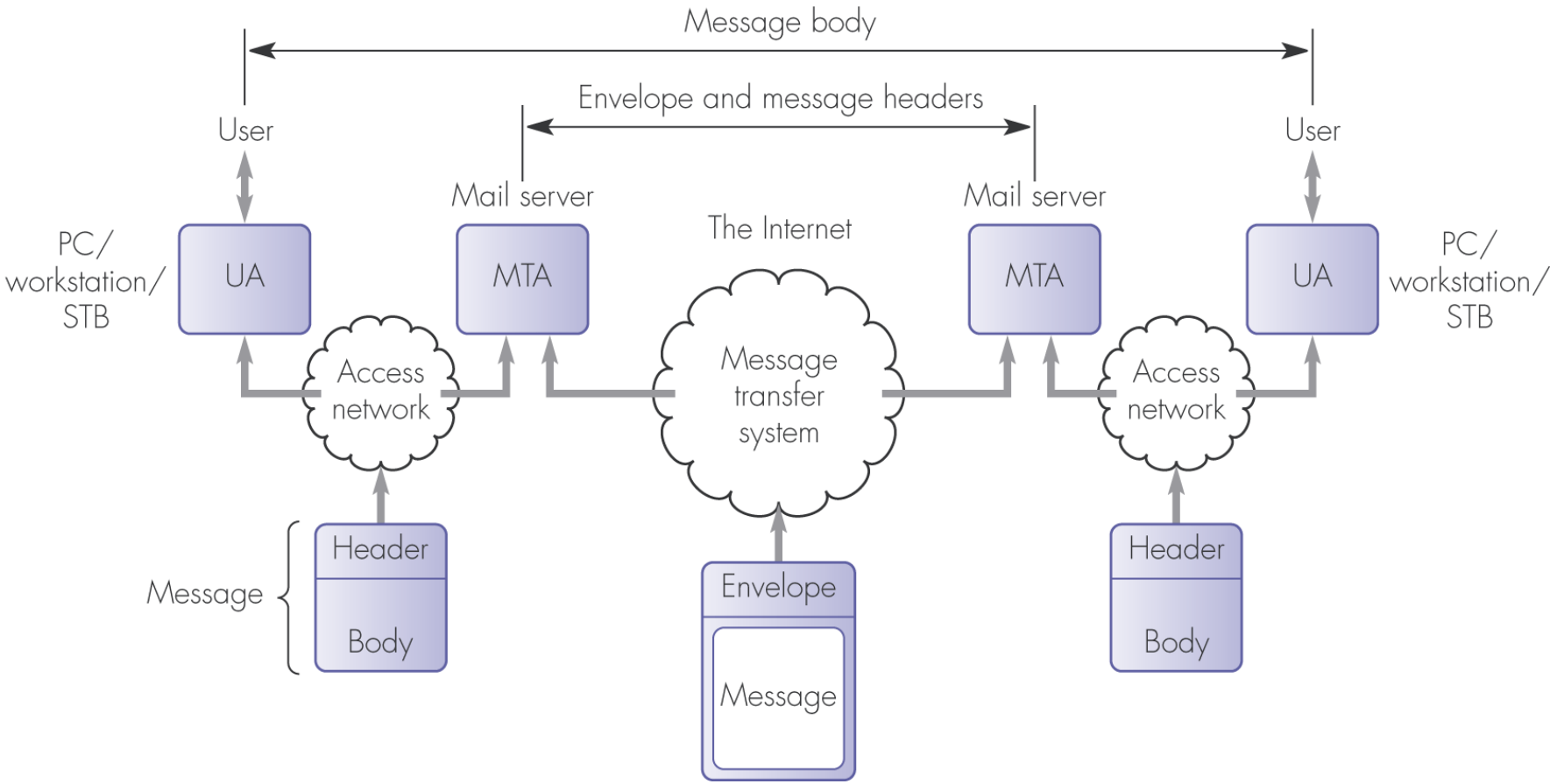
Mail Messages Format

- As defined by RFC 822, mail messages are composed of:
  - Header, with different fields
    - Mandatory and optional headers
  - Body
    - Message contents
    - Ends with a line with only one character (a “dot”).
  - A blank line to separate header from body
  - Header and body contain only ASCII 7 bit text format!!!



- UA builds the message and sends it to mail server
  - Sending mail server uses some header fields to build the “envelope” and know where to send it
    - “From” and “To” headers

Mail Messages Format



Halsall, *Computing Networking and the Internet,* 5th Edition © Pearson Education Limited 2005

Mail Messages Format

Transport related headers

To: address of main addressee

Cc: address of secondary addressee

Bcc: (**b**lind **c**arbon **c**opy) address of secondary hidden addressee

From: address of sender[1]

Sender: address of message emitter [1]

Received: Header added by every mail server traversed in the route, if any. Includes:

- Names of Sender and Receiver Servers
- Date and time message is received

Return-Path: name of last Server

- Added by last server of the route
- Rarely used, often contains the sender address

1: "From" and "Sender" usually coincide➔ "Sender" is omitted

Mail Messages Format

- Other headers

Date: date and time message was sent

Reply-To: address to send the response

Message-Id: unique identifier asigned by the UA, with the format:

random-string@domain

In-Reply-To: Id of the message that this mail responds

References: Message-Id of other messages

Subject: message subject (one line)

- And more…

Basic example of RFC 822 message

From: pepe@uah.es <CR><LF>
To: juani@upd.es <CR><LF>
Subjet: Planes futuros <CR><LF>
<CR><LF>
Hi, I want to tell you that … bla bla <CR><LF>
so that, bla bla <CR><LF>
etc, bla bla <CR><LF>
Regards <CR><LF>
. <CR><LF>

MIME Extensions

- RFC 822 **only specifies messages coded in ASCII** (7-bits/character)
  - English text messages only
- SMTP can only handle 7-bit ASCII text messages
- With the growth of Internet new needs appear:
  - Languages using accents
  - Languages with non latin alphabets (hebraic, russian)
  - No-alphabetic languages (chinese)
  - Non text messages (video, audio, images, etc.)
- Solution:
  - RFC´s 2045 y 2046 ➔ MIME extensions
  - MIME: **M**ultipurpose **I**nternet **M**ail **E**xtensions

MIME Extensions

- Basic MIME concept:
  - Do NOT modify transport related software
  - **Modify only Software at User Agents (UA´s)**
- Objective:
  - MIME messages being sent by existing programms and protocols
- **MIME ➔ Extensions to message format defined by RFC 822**
  - Adds new structure to message body
    - Adapts message structure to support non-ASCII contents
  - Defines coding rules for non-ASCII messages

Procedure fo sending mail with non ASCII contents

- UA sender **codes and sends original non ASCII in ASCII format**, indicating it
  - Required to:
    - Prevent confusion of SMTP, that accepts only 7 bit ASCII code
    - Prevent message contents being interpreted as SMTP messages

- UA **receptor receives-decodes** ASCII to its original **non-ASCII** format
  - Based on the values of MIME header

MIME Extensions

- MIME defines five new headers
  - MIME-Version
    - Only for MIME messages
  - Content-Description
    - Optional
  - Content-Id
    - Optional
  - Content-Transfer-Encoding
  - Content-Type

} These are the key headers

MIME Extensions

- "Content-Transfer-Encoding" Header
  - Indicates type of MIME coding used
    - Coding-syntax of transfer
  - Types of MIME codings
    - ASCII 7 bit with lines of less than 1000 characters (RFC 822)
    - ASCII 8 bit with lines of less than 1000 characters
      - For networks implementing some RFC 822 extensions
    - ASCII 8 bit with lines of arbitrary length
      - E.g. : executable programs
    - Base 64 coding (base64)
      - For coding binary data in ASCII, or other 8-bit per character alphabets
    - Quoted-printable coding
      - For alphabets very similar to, but with a limited number of special characters, with the 8th bit set to "1"

MIME extensions

- "Content-Type" Header
  - Specifies:
    - MIME content of message, its kind
  - Indicates the receiver UA what to do with the content
  - Several types and subtypes defined
    - Notation: type/subtype
  - Every content "type" has an associated list of "subtypes"
    - Some content "types" :

      text

      image

      application

      multipart

      message

MIME Extensions: MIME types

text: indicatex that message body contains text

examples:

- text/plain
  - Plain text, does not contain commands or format directives. Displayed as is, no special software needed
- text/html
  - Indicates to interpret HTML tags to display message content as a Web page

image: indicates the body contains image

examples:

- image/gif
  - Image in “gif” format
- image/jpeg
  - Image in “jpeg” format

MIME extensions : MIME types

application: indicates that the contents must be first processed by an application to become visible or usable by the user

Examples:

- application/msword
- application/msaccess
- application/msexcel
- application/postscript

MIME Extensions: MIME types

multipart: indicates the message has multiple parts or annexes (objects)

e.g, text, images, audio, etc.

Receiver UA, in order to proceed, needs to know:

- Where every object starts and ends
  - Via "boundary characters") between every pair of objects of message
- How every non-ASCI object was coded
  - Via header "Content-Transfer-Encoding" of every object
- Type of content of every object
  - Via "Content-Type" header of every object
- Example:

  multipart/mixed;boundary=XYZ

Indicates how the parts (objects) of a multipart message separate.
Separation always starts with two (or more) hyphen (see example)

"message": indicates that message content has a relation with another MIME message

- Examples:

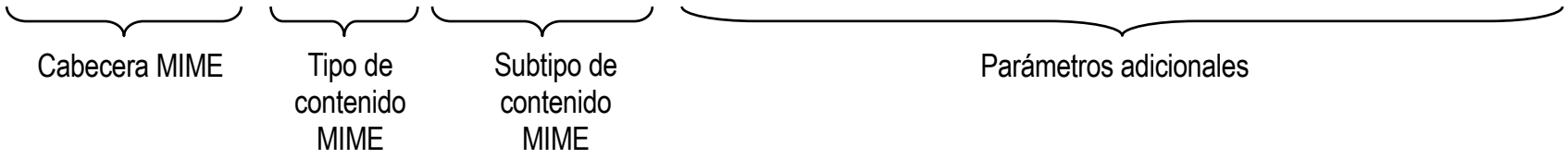
message/rfc822 ← indicates that contains another RFC 822 message

message/partial ← indicates that the content is part (fragment) of a bigger message

message/external-body ← indicates that the content is not present, instead a reference to the place where the real content is available

Content-Type: message/partial; id="file-name@host-name";number=1;total=20

Content-Type: message/external-body; access-type="mail-server";server="server-name"



MIME Extensions

Some MIME types and subtypes

TYPE	SUBTYPE	CONTENT DESCRIPTION
Text	Plain	ASCII text without format
	Richtext	Formatted text based onHTML
Image	GIF	GIF digital image
	jPEG	jPEG digital image
Audio	Basic	Digital audio
Video	MPEG	Video sequence or digital moviel
Application	Octet-stream	Byte string
	Postscript	Printable Adobe PostScript document
Message	RFC 822	Another MIME message
	Partial	Part of a bigger message
	External-body	Pointer to where the body of message can be obtained
Multipart	Mixed	Each part has different content or type
	Alternative	Every part has same content, but different type or subtype.
	Parallel	Part must be shown simultaneously
	Digest	Multiple messages

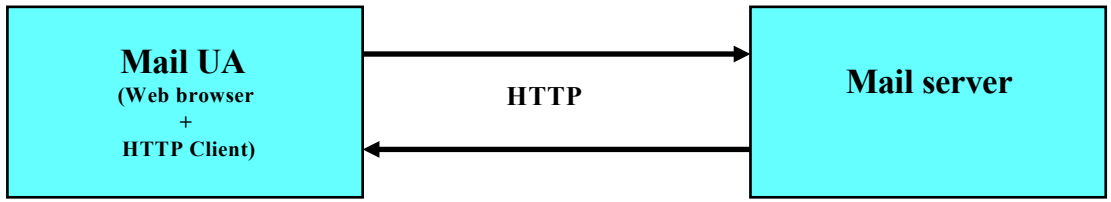
MIME extensions


```
From: xyz@abc.com
To: abc@xyz.com
Subject: Happy birthday Irene
MIME-Version: 1.0
Content-Type: Multipart/Alternative; boundary = "TryAgain";

– – TryAgain

Content-Type: Message/External-body;
        name = "Irene.audio";
        directory = "Irene";
        access-type = "anon-ftp";
        site = "myserver.abc.com";
Content-Type: Audio/Basic;                    (Message in audio accessed remotely)
Content-Transfer-Encoding: Base64

– – TryAgain

Content-Type: Text/Richtext;
<B> ***Happy birthday Irene*** </B> (Message in richtext)

– – TryAgain

Content-Type: Text/Plain;
        ***Happy birthday Irene*** (Message in plaintext)

– – TryAgain
```

Example of headers and MIME types/subtypes for multimedia mail message.
Same message in three different formats

Webmail

- Mail UA = Web Browser + HTTP Client
  - For mail servers that “talk” HTTP (Hotmail, Gmail, …)
- HTTP:
  - As a protocol to access mail
  - For message transfer between UA and mail Server
  - Enables some IMAP options
- Addressing:
  - URL = IP addres or domain name of mail Server



Webmail

- Advantage
  - Access to mail from any device connected to Internet
- Disadvantage
  - Sending and reception of messages may be slow
    - Due to the transfer procedure between browser and mail server
      - e.g: mail message in HTML format
      - Using forms and CGI scripts
  - Conclusion:
    - Whenever possible, it is preferrable to use a conventional mail UA

ASCII code

Bit positions			7	0	0	0	0	1	1	1	1
			6	0	0	1	1	0	0	1	1
			5	0	1	0	1	0	1	0	1
4	3	2	1								
0	0	0	0	NUL	DLE	SP	0	@	P	\	p
0	0	0	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	STX	DC2	"	2	B	R	b	r
0	0	1	1	ETX	DC3	#	3	C	S	c	s
0	1	0	0	EOT	DC4	$	4	D	T	d	t
0	1	0	1	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	ACK	SYN	&	6	F	V	f	v
0	1	1	1	BEL	ETB	'	7	G	W	g	w
1	0	0	0	BS	CAN	(	8	H	X	h	x
1	0	0	1	HT	EM	)	9	I	Y	i	y
1	0	1	0	LF	SUB	*	:	J	Z	j	z
1	0	1	1	VT	ESC	+	;	K	[	k	{
1	1	0	0	FF	FS	,	<	L	\	l	|
1	1	0	1	CR	GS	-	=	M	]	m	}
1	1	1	0	SO	RS	.	>	N	^	n	~
1	1	1	1	SI	US	/	?	O	_	o	DEL

Personal Work

- See student guide