

Redes de Computadores

Nivel de Aplicación:

Mail, Telnet, FTP, P2P

Mikel Izal Azcárate
(mikel.izal@unavarra.es)

En clases anteriores...

- ▶ El nivel de aplicación en Internet, protocolos de aplicación que usan los servicios de TCP/UDP.
No hay nivel de aplicación uniforme
- ▶ Analizando servicios: Web, DNS

Hoy:

- ▶ Mas servicios

Servicios de Internet

Objetivos:

- ▶ Aprender con el ejemplo: Funcionamiento de protocolos de nivel de aplicación
 - > **SMTP/POP3**
 - > Telnet
 - > FTP
 - > P2P

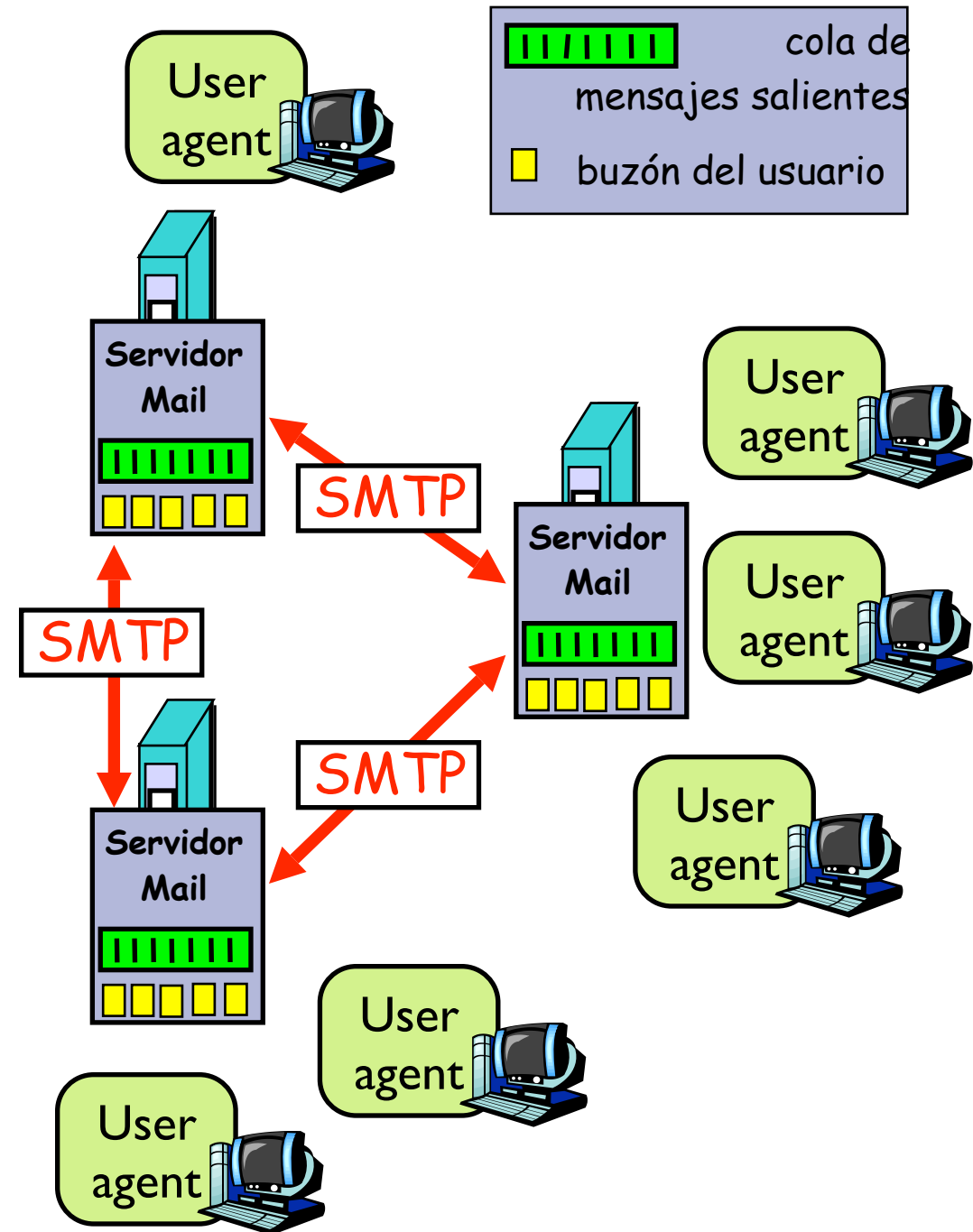
Electronic Mail

Tres elementos principales:

- ▶ Agentes de usuario (*user agents*)
- ▶ mail servers
- ▶ Simple Mail Transfer Protocol: SMTP

User Agent

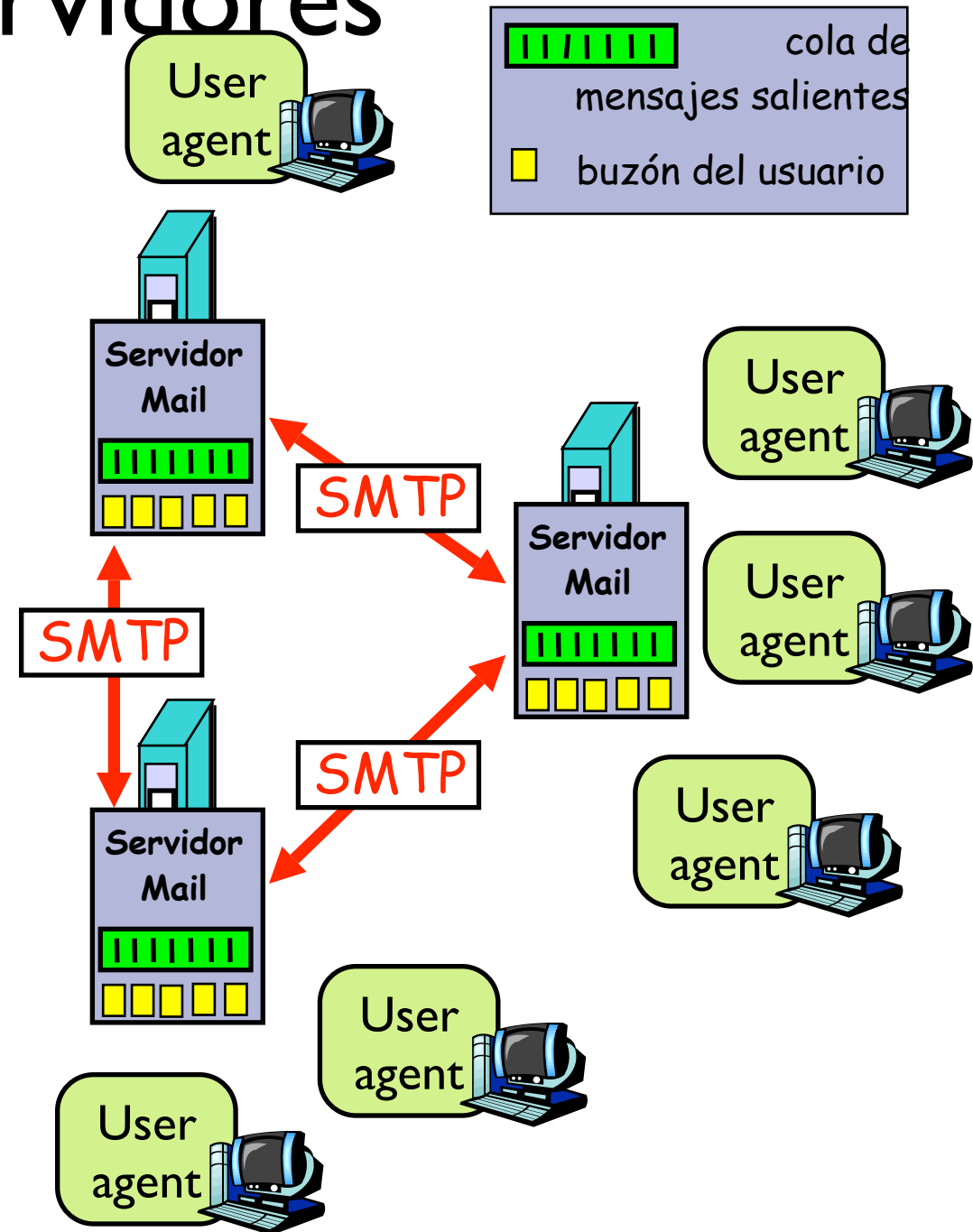
- ▶ alias “programa de correo”
- ▶ Componer, editar, leer mensajes de correo
- ▶ ej., Eudora, Outlook, elm, Netscape Messenger
- ▶ Mensajes salientes y entrantes en el servidor



Electronic Mail: Servidores

Servidores de Mail:

- ▶ **mailbox** contiene los mensajes entrantes para el usuario
- ▶ **cola de mensajes** salientes (a enviar)
- ▶ **Protocolo SMTP** entre servidores de correo para enviar mensajes
- > cliente: el servidor de correo que envía
- > “servidor”: el servidor de correo que recibe

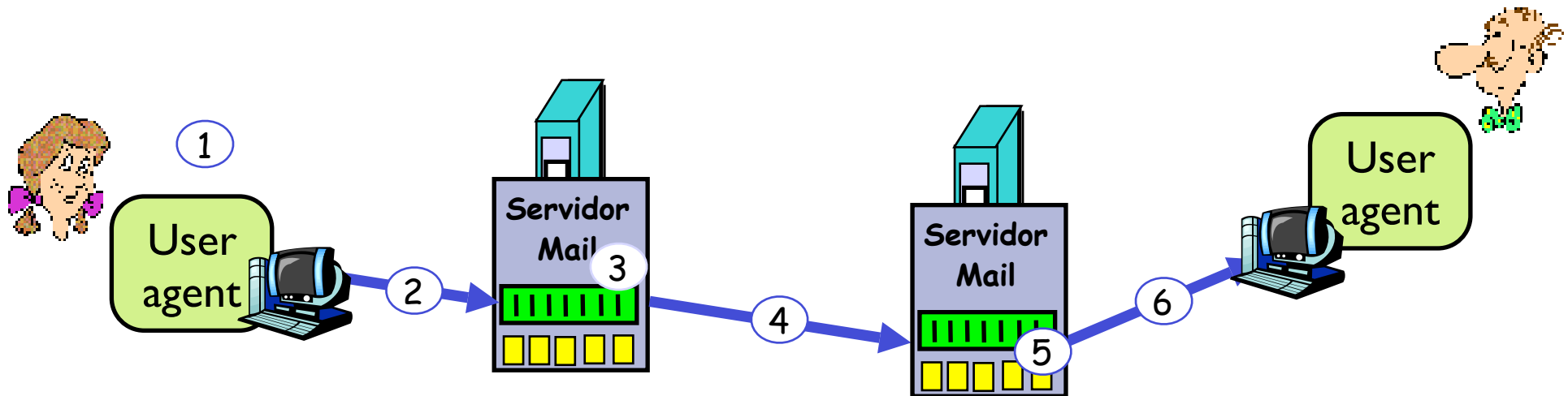


Electronic Mail: SMTP [RFC 2821]

- ▶ Emplea TCP para entregar de forma fiable los mensajes entre el cliente y el servidor
- ▶ Puerto 25
- ▶ Transferencia directa: del servidor del emisor al servidor del receptor
- ▶ Tres fases en la transferencia
 - > *handshaking* (el saludo)
 - > transferencia de mensajes
 - > cierre
- ▶ Interacción mediante comandos y respuestas
 - **comandos:** texto ASCII
 - **respuestas:** código de estado y frase de estado
- ▶ Los mensajes deben estar en ASCII de 7 bits

Ejemplo: Alicia envía mensaje a Bob

- 1) Alicia emplea un UA para crear el mensaje para `bob@somechool.edu`
- 2) El programa envía el mensaje a su servidor de correo y lo coloca en una cola de mensajes
- 3) El Servidor de Mail, como cliente, abre una conexión TCP con el Servidor de Bob
- 4) Envía el mensaje de Alicia empleando SMTP sobre esa conexión TCP
- 5) El servidor de mail de Bob coloca el mensaje en su buzón
- 6) Bob lanza su UA para leer el mensaje (volveremos a esta parte)



Ejemplo de SMTP

```
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
```

[S]ervidor [C]liente

Probando SMTP

▶ Escriba:

```
$ telnet servername 25
```

- > Pruebe los comandos HELO, MAIL FROM, RCPT TO, DATA, QUIT
- > Con esos comandos puede enviar un email sin emplear un programa de email

```
$ telnet si.unavarra.es 25
Trying 130.206.166.108...
Connected to si.unavarra.es.
Escape character is '^]'.
220 unavarra.es ESMTP Sendmail 8.9.3/8.9.1 (IRIS 3.0); Sun, 7 Aug 2005
14:06:28 +0200 (MET DST)
HELO mikel.tlm.unavarra.es
250 unavarra.es Hello s169m177.unavarra.es [130.206.169.177], pleased to
meet you
MAIL FROM: mikel.izal@unavarra.es
250 mikel.izal@unavarra.es... Sender ok
RCPT TO: mikel.izal@gmail.com
250 mikel.izal@gmail.com... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Hola esto es para probar que va
.
250 OAA13441 Message accepted for delivery
QUIT
221 unavarra.es closing connection
Connection closed by foreign host.
```

Más sobre SMTP

- ▶ SMTP emplea **conexiones persistentes**
- ▶ SMTP requiere que el mensaje (cabecera y contenido) esté en ASCII de 7 bits
- ▶ El servidor de SMTP reconoce el fin del mensaje al ver una línea que sólo contenga .
`CRLF.CRLF` `"\r\n.\r\n"`

Comparación con HTTP:

- ▶ HTTP: pull
- ▶ SMTP: push
- ▶ Ambos usan comandos y respuestas en ASCII

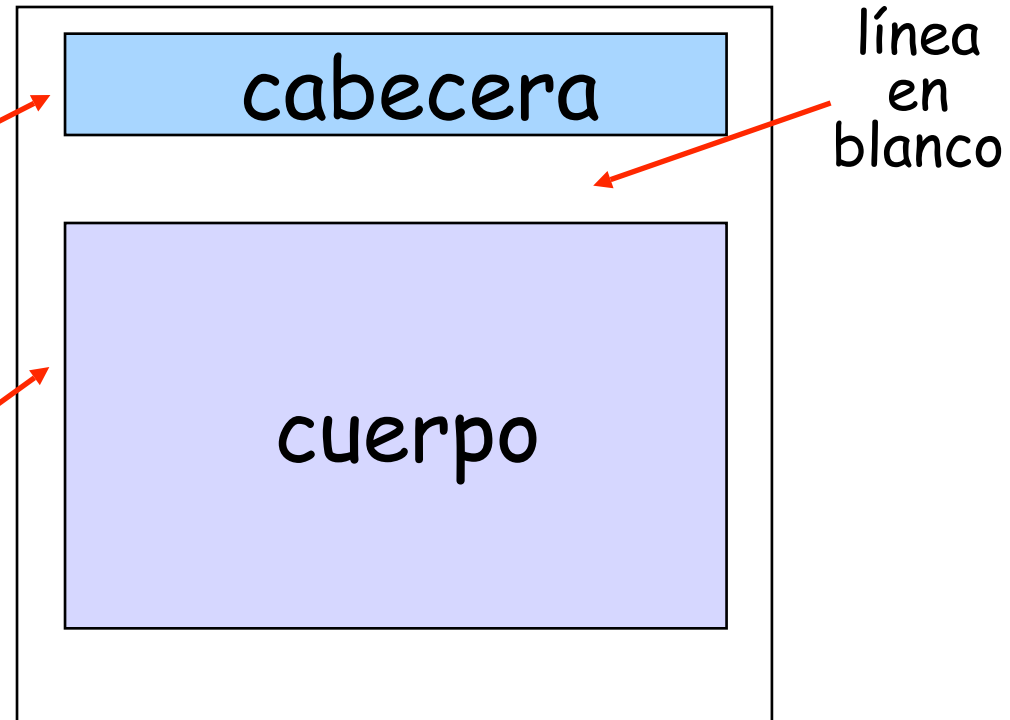
Formato del mensaje de email

SMTP: protocolo para intercambiar mensajes de email

RFC 822: estándar para el formato del mensaje:

- ▶ líneas de cabecera, ej.,
 - > **To:**
 - > **From:**
 - > **Subject:**

diferentes de los comandos de SMTP
- ▶ cuerpo
 - > el “mensaje”, solo caracteres ASCII



Formato del mensaje: multimedia

- ▶ MIME: Multimedia Mail Extension, RFC 2045, 2056
- ▶ Permite mandar contenido que no sea texto ASCII
- ▶ Líneas adicionales en la cabecera del mensaje para declarar el tipo del contenido

versión de MIME

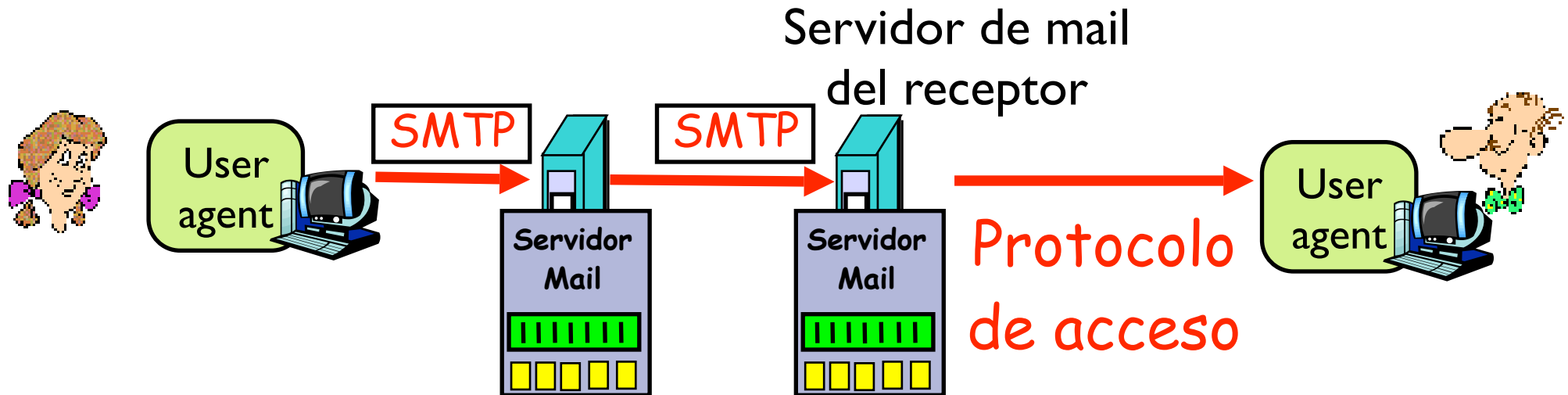
método empleado para
codificar los datos

tipo, subtipo, parametros
de los datos multimedia

datos codificados

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
base64 encoded data .....
.....base64 encoded data
```

Protocolos de acceso al Mail



- ▶ SMTP: entrega/almacena en el servidor del receptor
- ▶ Protocolo de acceso al Mail: obtención de mensajes del servidor
 - > POP: Post Office Protocol [RFC 1939]
 - + Autorización (agente <-->servidor) y descarga
 - > IMAP: Internet Mail Access Protocol [RFC 1730]
 - + más funcionalidades (más complejo)
 - + manipulación de mensajes almacenados en el servidor
 - > HTTP: Hotmail ,Yahoo! Mail, etc.

Protocolo POP3

Autorización

- ▶ Comandos del cliente:
 - > **user**: declara el nombre de usuario
 - > **pass**: clave
- ▶ Respuestas del servidor:
 - > **+OK**
 - > **-ERR**

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

Fase de transacción, cliente:

- ▶ **list**: lista números de mensajes
- ▶ **retr**: descarga mensaje por número
- ▶ **dele**: borrar
- ▶ **quit**

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

Más sobre POP3 e IMAP

Más sobre POP3

- ▶ El ejemplo anterior era “descargar y borrar”
- ▶ Bob no puede volver a leer los mensajes si cambia de cliente
- ▶ “Descargar y mantener”: copia el mensaje pero no lo borra. Permite descargarlos en otro cliente
- ▶ POP3 es sin estado entre sesiones

IMAP

- ▶ Mantiene todos los mensajes en un lugar: el servidor
- ▶ Permite al usuario organizar los mensajes en carpetas
- ▶ IMAP mantiene el estado entre sesiones:
- ▶ Nombres de carpetas y relación entre ID de mensaje y carpeta en la que está

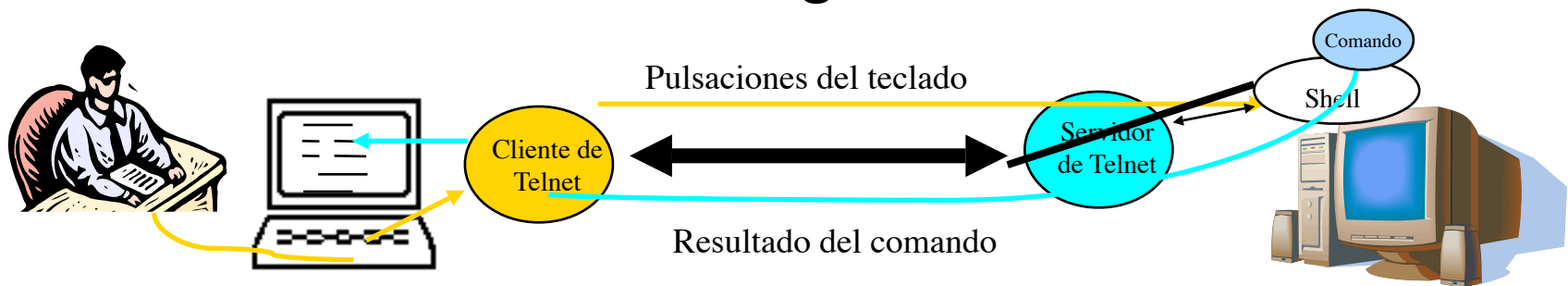
Servicios de Internet

Objetivos:

- ▶ Aprender con el ejemplo: Funcionamiento de protocolos de nivel de aplicación
 - > SMTP/POP3
 - > **Telnet**
 - > FTP
 - > P2P

Login remoto (Telnet)

- ▶ Permite el uso interactivo de otra computadora (UNIX) de forma remota como desde un terminal
- ▶ Funcionamiento:
 - > El usuario ejecuta un cliente de Telnet especificando una máquina servidor...
 - > Se crea una conexión TCP con el servidor (puerto del servidor de Telnet=23)...
 - > El servidor crea un proceso Shell que queda conectado a la conexión TCP...
 - > Las pulsaciones del teclado del usuario se transmiten por la conexión a la Shell...
 - > La shell ejecuta los comandos que escribe el usuario...
 - > El resultado que el comando mandaría a la pantalla vuelve por la conexión TCP y sale en la pantalla del cliente...
- ▶ Otros servicios similares: rlogin, rsh, ssh



Login remoto (Telnet): Ejemplo

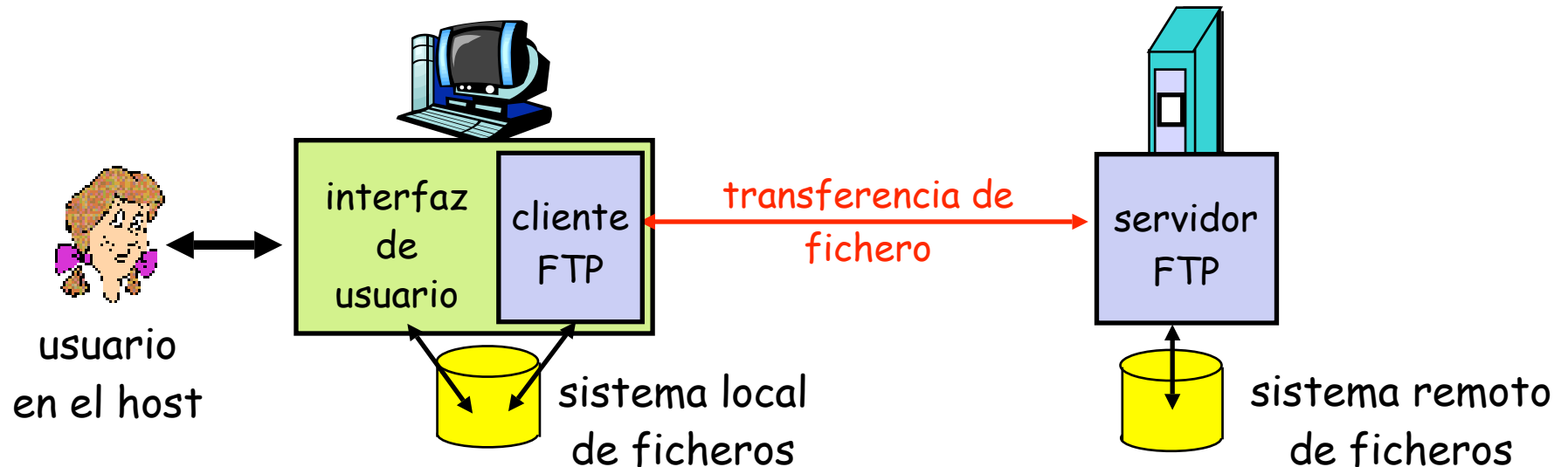
```
$ telnet tlm22.net.tlm.unavarra.es
Trying 10.1.1.22...
Connected to tlm22.net.tlm.unavarra.es.
Escape character is '^]'.
Fedora Core release 2 (Tettnang)
Kernel 2.6.5-1.358 on an i686
login: mikel
Password:
Last login: Fri May 6 20:33:54 from bender.net.tlm.unavarra.es
[mikel@tlm22 mikel]$ ls -l
total 106132
drwxr-xr-x  2 mikel staff      4096 Mar  5 00:17 a
drwxr-xr-x 15 mikel staff      4096 Oct 25  2004 apache
-rw-r--r--  1 mikel staff 41685513 Nov 12  2004 borrar
drwxr-xr-x  4 mikel staff      4096 Dec  2  2004 demo-italia
-rw-----  1 mikel staff    116627 Dec  2  2004 demo-italia.tar.gz
drwxr-xr-x  3 mikel staff      4096 Jun 27 09:43 Desktop
drwx-----  4 mikel staff      4096 May 27 15:38 evolution
drwxr-xr-x  2 mikel staff      4096 Oct  6  2004 prueba_c
-rw-r--r--  1 mikel staff    209211 May  6 10:24 prueba.ps
drwxr-xr-x  2 mikel staff      4096 May 11 21:34 py23
[mikel@tlm22 mikel]$
```

Servicios de Internet

Objetivos:

- ▶ Aprender con el ejemplo: Funcionamiento de protocolos de nivel de aplicación
 - > SMTP/POP3
 - > Telnet
 - > **FTP**
 - > P2P

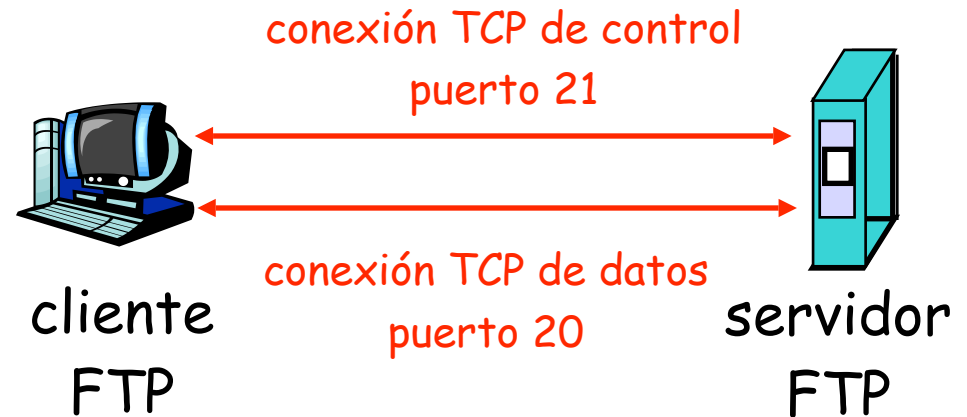
FTP: File Transfer Protocol



- ▶ Transferencia de fichero hacia/desde host remoto
- ▶ modelo cliente-servidor
 - > **cliente**: extremo que inicia la transferencia (bien sea desde o hacia el extremo remoto)
 - > **servidor**: host remoto
- ▶ FTP: RFC 959
- ▶ Servidor FTP: TCP puerto 21

FTP: conexiones de datos y control

- ▶ El cliente FTP contacta con el servidor en el puerto 21 empleando TCP
- ▶ El cliente se autentica a través de esta conexión de control
- ▶ El cliente puede explorar los directorios remotos enviando comandos por la conexión de control
- ▶ Cuando el servidor recibe un comando para una transferencia de fichero abre una conexión TCP con el cliente
- ▶ Tras transferir el fichero cierra esa conexión de datos



- ▶ El servidor abre una segunda conexión TCP para transferir el fichero
- ▶ Conexión de control “out of band”
- ▶ El servidor FTP mantiene el “estado”: directorio actual, autenticación

Comandos y respuestas FTP

Comandos de ejemplo:

Enviados como texto ASCII por el canal de control

- ▶ **USER username**
- ▶ **PASS password**
- ▶ **LIST** devuelve una lista de los ficheros en el directorio actual
- ▶ **RETR filename**
Obtiene el fichero
- ▶ **STOR filename**
Almacena el fichero en el host remoto

Códigos de respuesta:

Código de estado y frase (como en HTTP)

- ▶ **331 Username OK, password required**
- ▶ **125 data connection already open; transfer starting**
- ▶ **425 Can't open data connection**
- ▶ **452 Error writing file**

Ejemplo de FTP

```
$ ftp tlm22.net.tlm.unavarra.es
Connected to tlm22.net.tlm.unavarra.es (10.1.1.22).
220 (vsFTPd 1.2.1)
Name (tlm22.net.tlm.unavarra.es:mikel): mikel
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
227 Entering Passive Mode (10,1,1,22,215,255)
150 Here comes the directory listing.
drwxr-xr-x   3 1003   1000           4096 Jun 27 07:43 Desktop
-rw-r--r--   1 1003   1000       209211 May 06 08:24 prueba.ps
drwxr-xr-x   2 1003   1000           4096 Oct 06 2004 prueba_c
drwxr-xr-x   2 1003   1000           4096 May 11 19:34 py23
-rw-r--r--   1 1003   1000     20083157 May 11 19:34 py23.tgz
226 Directory send OK.
ftp> get py23.tgz
local: py23.tgz remote: py23.tgz
227 Entering Passive Mode (10,1,1,22,95,165)
150 Opening BINARY mode data connection for py23.tgz (20083157
bytes).
226 File send OK.
20083157 bytes received in 1.86 secs (1.1e+04 Kbytes/sec)
ftp>
```

Servicios de Internet

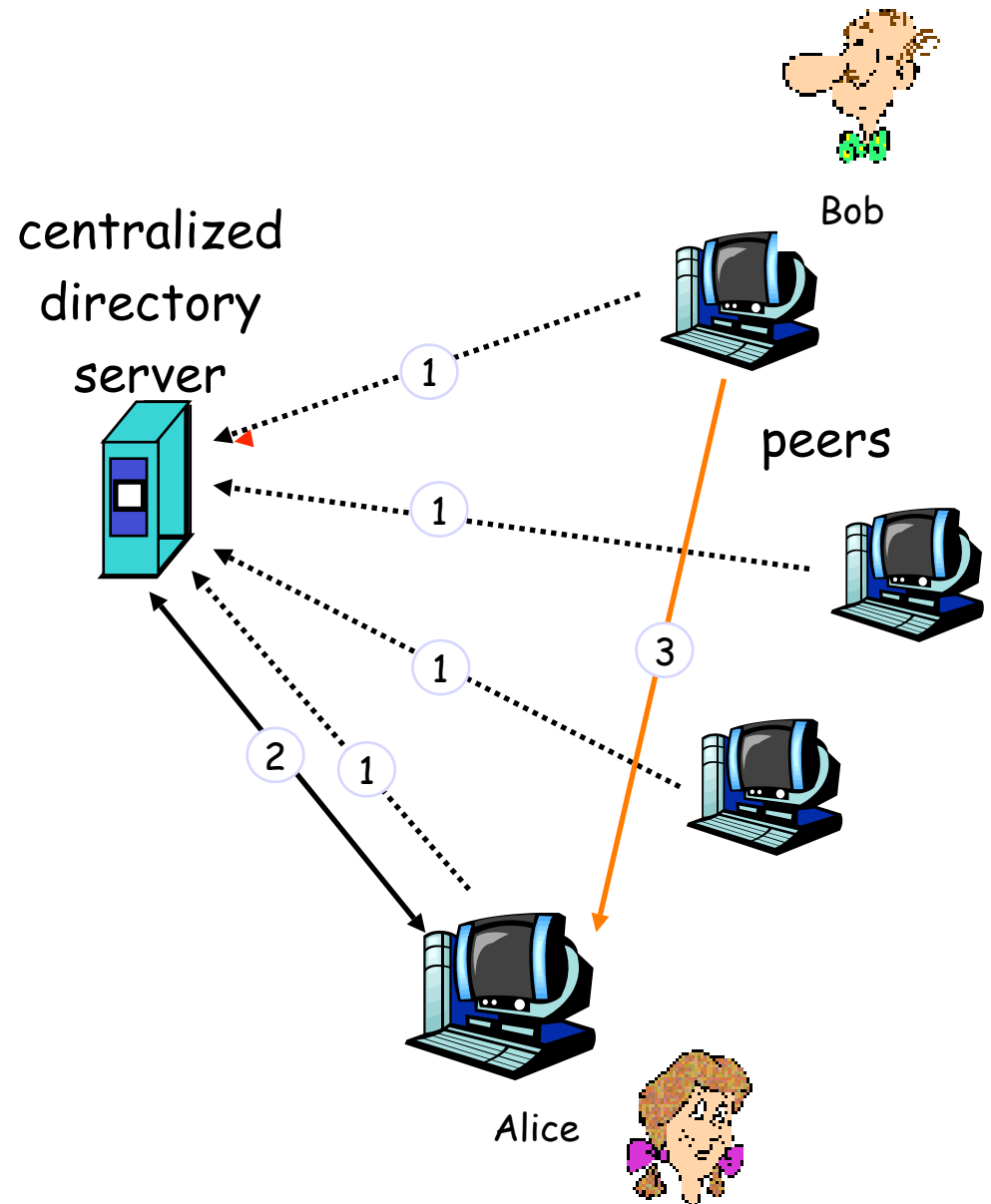
Objetivos:

- ▶ Aprender con el ejemplo: Funcionamiento de protocolos de nivel de aplicación
 - > SMTP/POP3
 - > Telnet
 - > FTP
 - > **P2P**

P2P: directorio centralizado

Diseño original de
“Napster”

- 1) Cuando un peer se conecta, informa al servidor central:
 - > Dirección IP
 - > contenido
- 2) Alice hace una búsqueda de “Hey Jude”
- 3) Alice pide el fichero a Bob



Ventajas e inconvenientes

Ventajas

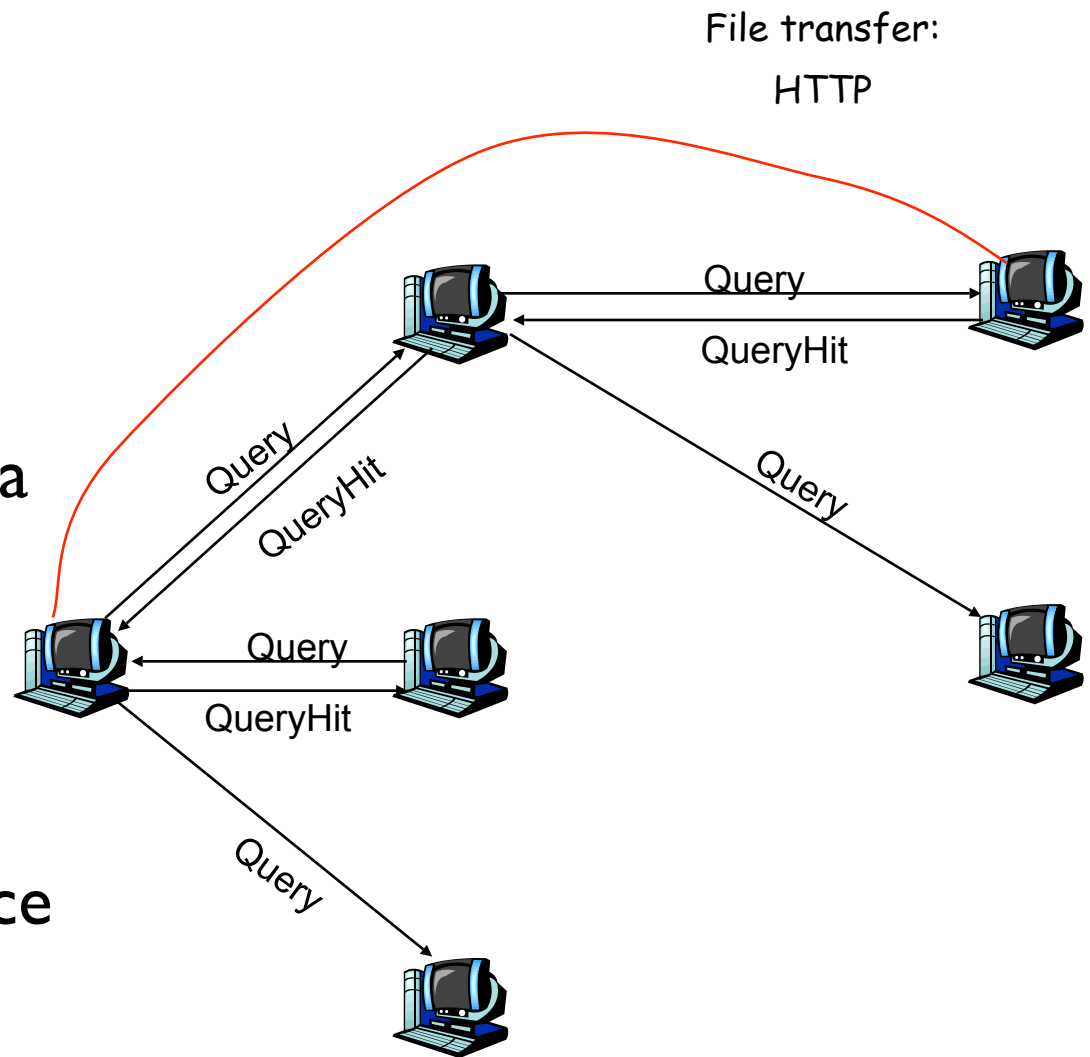
- ▶ Todos los peers son servidores
- ▶ **Altamente escalable**

Inconvenientes

- ▶ Un **punto de fallo** central
- ▶ Impone un límite de prestaciones
- ▶ Infracción de copyrights!

Gnutella

- ▶ Completamente distribuido
- ▶ Dominio público
- ▶ Overlay network
 - > Grafo
 - > Cada conexión un enlace
- ▶ Petición de búsqueda enviada sobre las conexiones TCP
- ▶ peers reenvían la petición
- ▶ Respuesta enviada por el camino inverso
- ▶ Escalabilidad: limitar el alcance de la inundación



Conclusiones

- ▶ Estudio de servicios y protocolos de aplicación
 - > HTTP
 - > FTP
 - > e-mail
 - > DNS
 - > P2P
 - > Telnet

- ▶ Próxima clase:
¿Como se construyen aplicaciones de red?
El API de **Sockets**