Bloque II: El nivel de aplicación

Tema 3: Aplicaciones orientadas a conexión

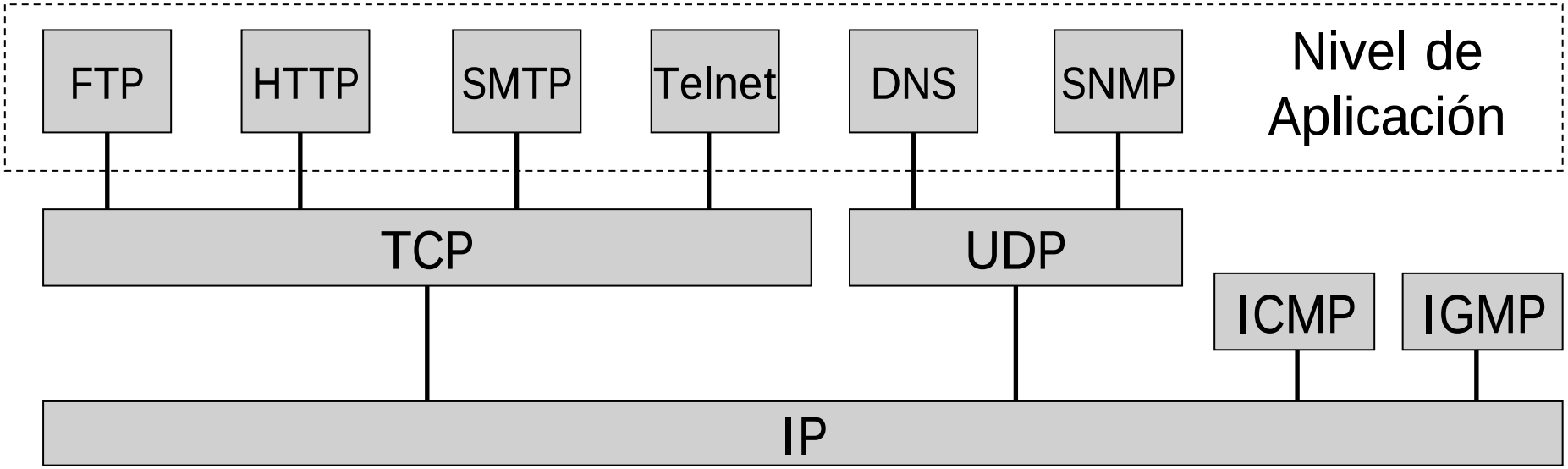
Índice

- Bloque II: El nivel de aplicación
  - Tema 3: Aplicaciones orientadas a conexión
    - Introducción
    - World Wide Web
    - Transferencia de ficheros
    - Correo electrónico

- **Referencias**
  - Capítulo 2 de “Redes de Computadores: Un enfoque descendente basdado en Internet”. James F. Kurose, Keith W. Ross. Addison Wesley, 2ª edición. 2003.
  - Capítulos 27 y 28 de “TCP/IP Illustrated, Volume 1: The Protocols”, W. Richard Stevens, Addison Wesley, 1994.

Introducción

- Dos procesos en dos sistemas finales (distintos) se comunican intercambiando mensajes a través de una red de computadores.
- Modelo cliente-servidor
  - Cliente envía mensajes al servidor
  - Servidor recibe los mensajes, procesa la respuesta y la envía
- Protocolos del nivel de aplicación:
  - Definen el formato y el orden de intercambio de los mensajes
  - Acciones en la transmisión o recepción de mensajes



Introducción

- Protocolo del nivel de aplicación es sólo una parte de la aplicación de red. Por ejemplo, en el Web:
  - Formato de los documentos (HTML)
  - Navegadores Web (Mozilla, Explorer, ...)
  - Servidores Web (Apache, MIIS, ...)
  - Protocolo de la capa de aplicación (HTTP)

- Protocolo del nivel de aplicación:
  - Tipo de mensajes intercambiados (petición/respuesta)
  - Sintaxis de los mensajes
  - Semántica de los campos
  - Reglas que determinan cuándo y cómo un proceso envía un mensaje y responde a los mensajes.

- Aplicaciones y protocolos más utilizados:
  - Web – HTTP (RFC 2616)
  - Transferencia de ficheros – FTP (RFC 959)
  - Correo electrónico – SMTP (2821)
  - Acceso a terminales remotos – Telnet (RFC 854)

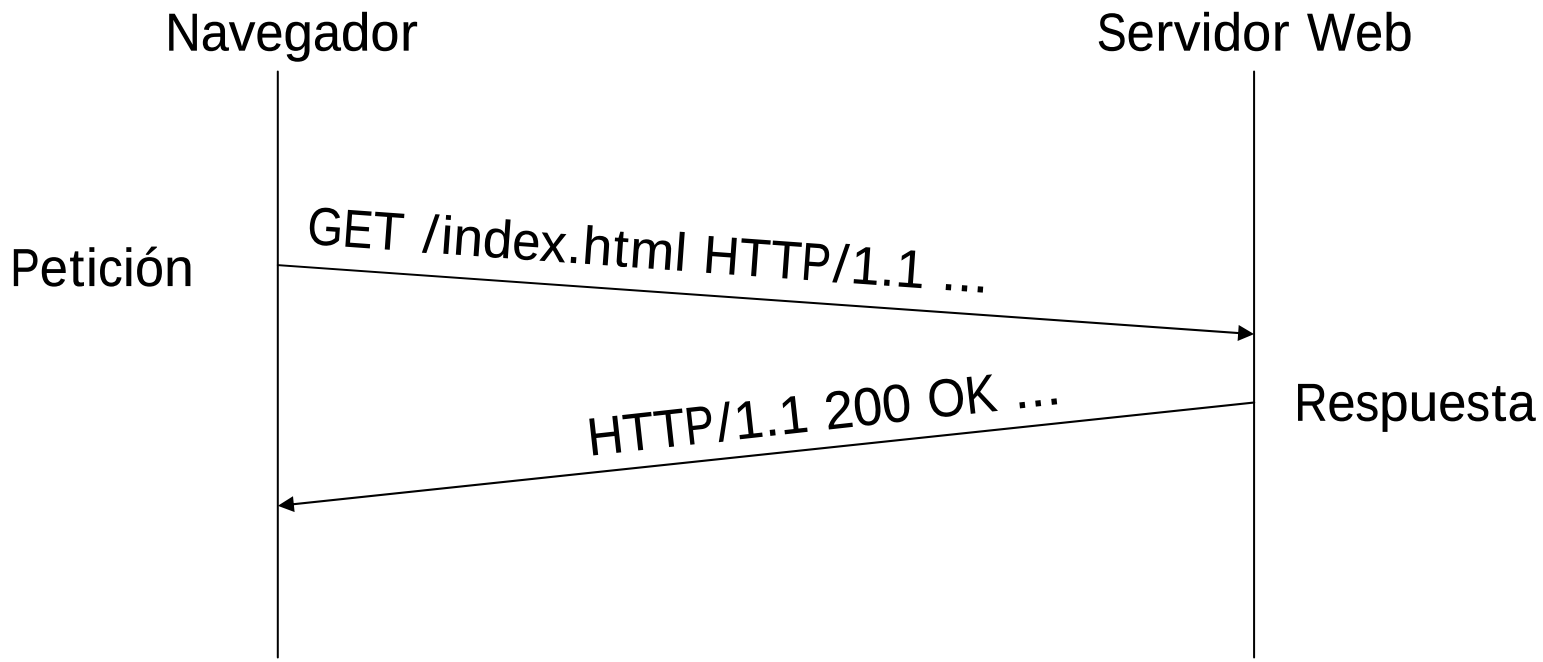
Web

- Es una aplicación en Internet.
- El World Wide Web surgió a principios de los 90, en el CERN (Tim Berners-Lee), para organizar los documentos de investigación disponibles en Internet.
- Combina cuatro ideas que no eran nuevas:
  - **Hipertexto**: formato de la información que permite moverse de una parte a otra de un documento o entre documentos mediante conexiones internas entre estos documentos (hiperenlaces o enlaces).
  - **Identificadores de recursos**: identificadores únicos que permiten localizar un recursos en la red (URL – Uniform Resource Locator o URI – Uniform Resource Identifier)
  - **Modelo cliente-servidor**
  - **Lenguaje de marcas**: caracteres o códigos embebidos en texto que indican estructura, semántica o recomendaciones para su presentación (HTML – HyperText Markup Language).
- Tim Berners-Lee desarrolló el protocolo HTTP inicial, las URLs, HTML y el primer servidor Web.

- Componentes:
  - **Página Web**: archivo HTML base + objetos (imágenes)
  - **Navegador**: agente de usuario para el Web
  - **Servidor Web**: almacena objetos Web direccionables a través de una URL
  - **Protocolo HTTP**: permite comunicarse al servidor y al navegador

HTTP

- HyperText Transfer Protocol
- Especificado en RFC 1945 (HTTP/1.0) y RFC 2616 (HTTP/1.1)
  - HTTP/1.1 compatible con HTTP/1.0
- Define cómo los clientes (navegadores) solicitan páginas Web y cómo los servidores transfieren estas páginas.
- Utiliza el protocolo TCP (servicio orientado a conexión y fiable) ➔ Cada mensajes HTTP emitido por el cliente o servidor llega al otro extremo sin modificaciones.
- HTTP es un protocolo **sin estado** ➔ El servidor HTTP no guarda información sobre los clientes.



HTTP: Conexiones no persistentes

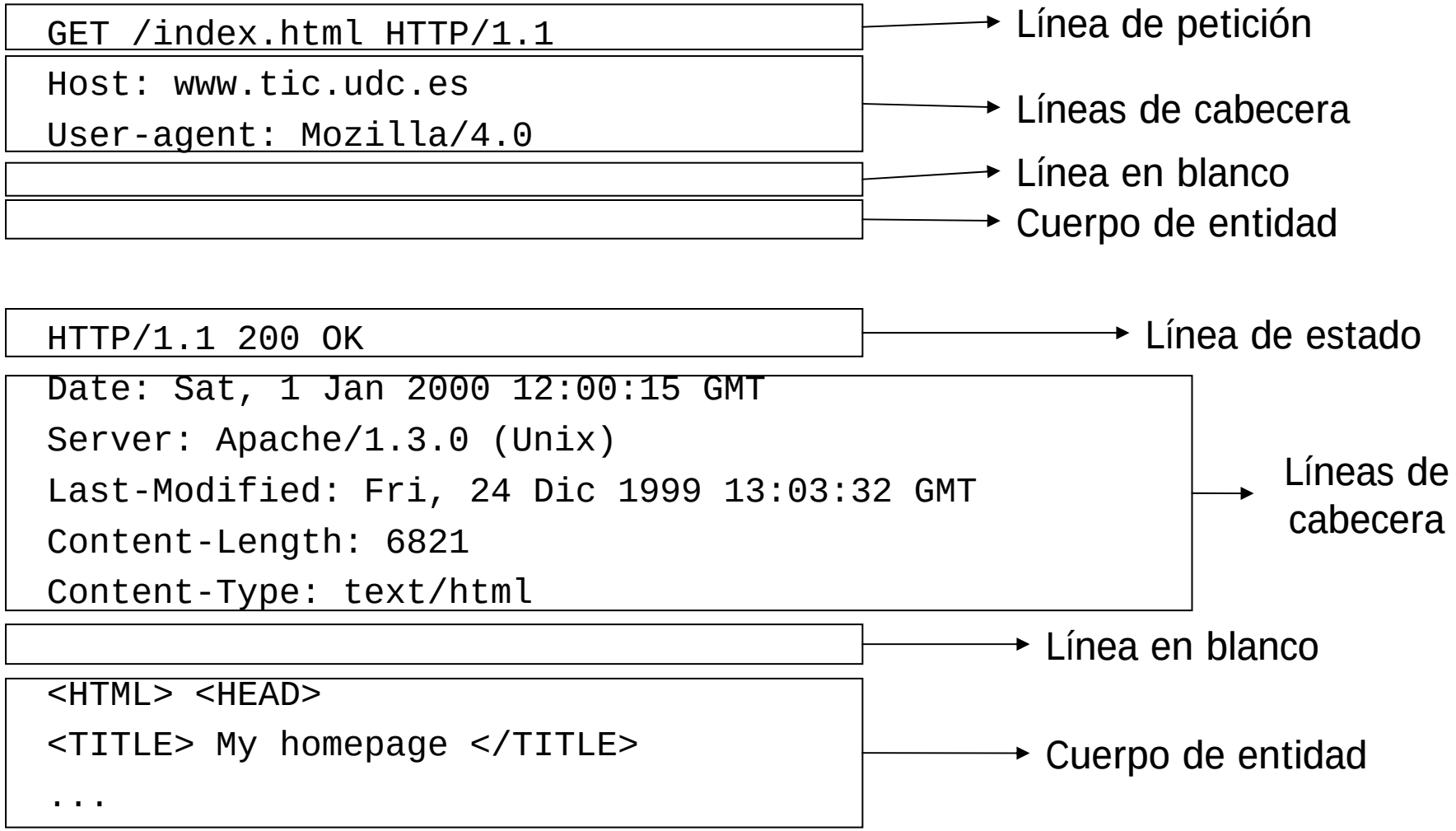
- HTTP/1.0 usa conexiones no persistentes
- Petición de una URL (http://www.tic.udc.es/index.php)
  1. El cliente HTTP inicia la conexión TCP con el servidor www.tic.udc.es en el puerto 80.
  2. El cliente HTTP envía al servidor el mensaje de petición solicitando el objeto /index.php
  3. El servidor HTTP recibe la petición, lee el objeto, lo encapsula en el mensaje HTTP de respuesta y lo envía.
  4. El servidor finaliza la conexión TCP.
  5. El cliente HTTP recibe la respuesta y finaliza la conexión TCP.
  6. El cliente extrae el archivo del mensaje de respuesta, examina el archivo HTML y encuentra referencias a otros objetos HTML (p.e. imágenes)
  7. Volver al paso 1, para cada objeto.

- Dependiendo del navegador, las nuevas conexiones podrían ser en paralelo.
- Inconvenientes:
  - Se necesita una conexión (buffers, variables, timeouts, ...) para cada objeto solicitado.
  - Retardo de dos veces el RTT: establecimiento de conexión + petición y recepción del objeto

HTTP: Conexiones persistentes

- Por defecto, en HTTP/1.1
- El servidor HTTP deja abierta la conexión TCP, esperando nuevas petición/respuestas.
  - El servidor cerrará la conexión después de un tiempo de inactividad.
- Sin pipeline: el cliente sólo envía una nueva petición cuando ha recibido la respuesta previa.
- Con pipeline: el cliente realiza una petición tan pronto encuentra una referencia a un objeto.

Mensajes HTTP



Mensaje petición HTTP

- Línea de petición + línea en blanco: obligatorio
- Línea de petición:
  - Método:
    - GET: utilizando cuando el navegador solicita un objeto.
    - HEAD: el servidor responde con un mensaje HTTP, pero sin incluir el objeto solicitado.
    - POST: incluye datos en el cuerpo de entidad (no en el caso de un GET).
    - PUT (1.1): permite a un usuario cargar un objeto en la ruta especificada.
    - DELETE (1.1): permite borrar un objeto de un servidor Web.
  - URL: objeto al que se hace referencia
  - Versión
- Host: especifica el host en el que reside el objeto.
- User-agent: especifica el tipo de navegador que está haciendo la petición.

- POST: utilizado comúnmente cuando un usuario rellena un formulario.
  - El cuerpo de entidad contiene los datos introducidos por el usuario.
- GET: también soporta el envío de datos introducidos por el usuario.
  - Se envían codificados en la URL real.
  - Por ejemplo: www.google.com/search?keywords=information+retrieval

Mensaje respuesta HTTP

- Línea de estado: Versión + Código de estado + Frase
  - Códigos de estado y frases:
    - 200 OK
    - 400 Bad Request: petición no comprendida por el servidor.
    - 404 Not Found: el objeto pedido no existe en el servidor.
- Date: fecha y hora en la que se creó y envió la respuesta HTTP.
- Server: especifica el tipo de servidor Web que ha atendido a la petición.
- Last-Modified: indica la fecha y hora en que el objeto fue creado o modificado por última vez.
- Content-Length: indica el número de bytes del objeto enviado.
- Content-Type: indica el tipo de objeto incluido en el cuerpo de entidad.
  - La extensión del archivo no especifica (formalmente) el tipo de objeto.

HTTP: GET condicional

- La utilización de una caché reduce los retardos de recuperación de objetos y reduce el tráfico que circula por la red.
- Problema: la copia de un objeto en caché puede ser obsoleta.
- Solución: GET + If-Modificed-Since
  - Sólo devuelve el objeto si ha sido modificado después de la fecha indicada.

- Solicitar un objeto por primera vez:

```
GET /images/udc.gif HTTP/1.1
User-agent: Mozilla/4.0
```

- Recibir la respuesta del servidor:

```
HTTP/1.1 200 OK
Date: Sat, 1 Jan 2000 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Fri, 24 Dic 1999 13:03:32 GMT
Content-Type: image/gif

(datos)...
```

HTTP: GET condicional

- Pasado un tiempo, se vuelve a solicitar el mismo objeto, sólo si se ha modificado:

```
GET /images/udc.gif HTTP/1.1
User-agent: Mozilla/4.0
If-modified-since: Fri, 24 Dic 1999 13:03:32
   GMT
```

- Si no se ha modificado, el servidor no envía el objeto de nuevo.

```
HTTP/1.1 304 Not Modified
Date: Wed, 5 Jan 2000 20:30:43 GMT
Server: Apache/1.3.0 (Unix)
```

HTTP: Identificación de usuarios

- **Autorización**: el usuario debe identificarse (login y password) al acceder a un servidor Web.
  - El usuario realiza una petición normal.
  - El servidor responde con un mensaje `401 AuthorizationRequired`.
    - Campo `WWW-Authenticate` que especifica cómo se realiza la autentificación.
  - El navegador solicita el login y password al usuario e incluye una cabecera `Authorization`.
  - En las peticiones subsiguientes se repite el login y password.
- **Cookies**: mecanismo de almacenamiento en la máquina del cliente
  - Línea de cabecera de cookie en el mensaje HTTP de respuesta (`Set-cookie`)
  - Línea de cabecera de cookie en la petición (`Cookie`)
  - Archivo de cookies almacenado en el ordenador cliente (gestionado por el navegador)
  - Base de datos de apoyo en el servidor Web
  - Ejemplo: carrito de la compra

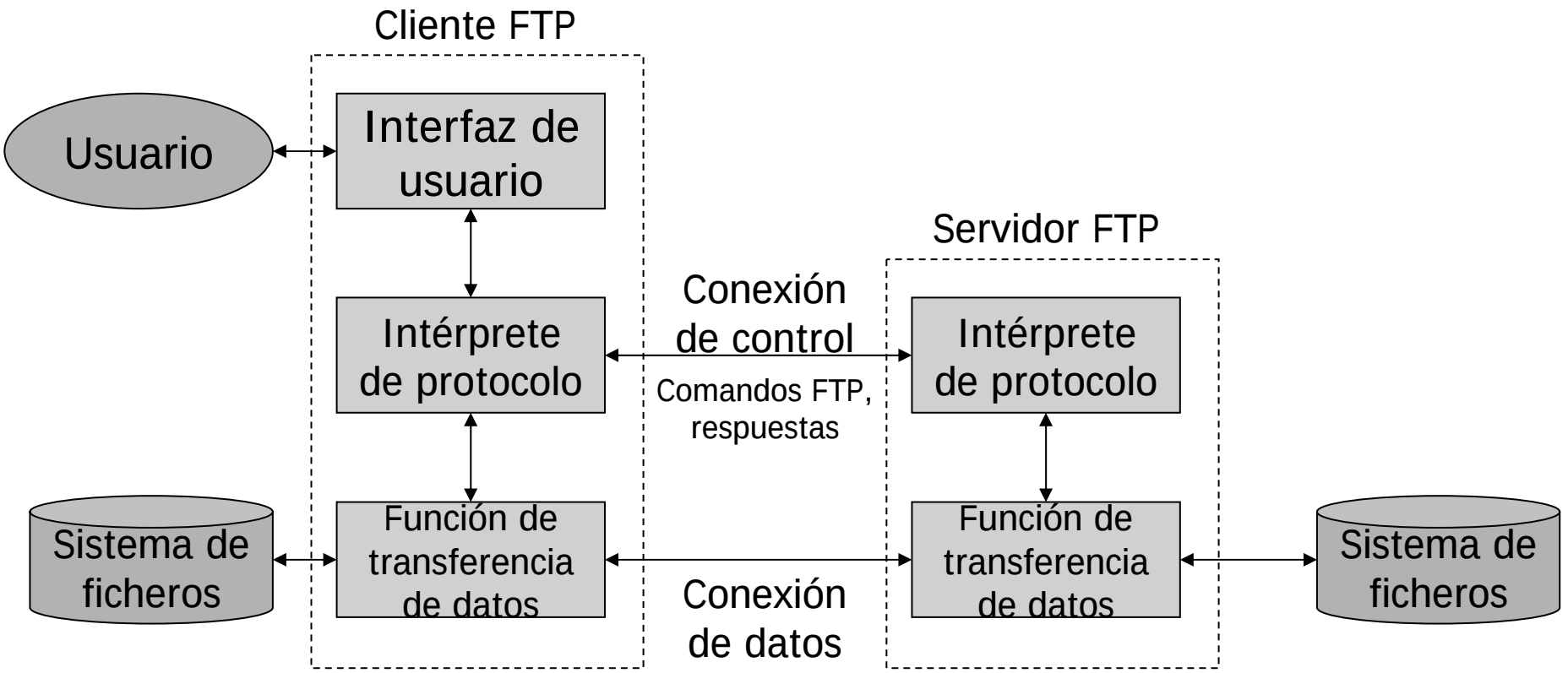
Transferencia de ficheros

- Se basa en el protocolo FTP (File Transfer Protocol).
- **Transferencia de ficheros**: permite copiar ficheros desde un sistema a otro.
- Acceso a ficheros: permite acceder a un fichero desde un sistema a otro ➔ NFS (Network File System).
- FTP fue diseñado para operar con sistemas heterogéneos ➔ Soporte para distintos tipos de ficheros (ASCII, binarios, …)
- Las passwords y el contenido de los ficheros se envían sin cifrar.
  - FTP sobre SSH: secure FTP
- FTP anónimo: un usuario no necesita cuenta para acceder al servidor FTP.
  - Se “solicita” el e-mail del usuario.
- Integrado hoy en día en los principales navegadores:
  - ftp://login:password@ftp.udc.es/

FTP

- Se especifica en el RFC 959.
- Basado en TCP.
- Operación:
  - El usuario se conecta al host remoto indicando su nombre (puerto 21).
  - El usuario se identifica (login y password) y sus datos son enviados a través de la conexión TCP.
  - Cuando el servidor ha autorizado al usuario, éste puede copiar/descargarse archivos al/del servidor de FTP.
- Utiliza dos conexiones TCP:
  - Conexión de control: envía información de control entre los dos hosts (login, password, comandos, ...) ➔ TOS: minimizar retardo
  - Conexión de datos: se utiliza para enviar un archivo ➔ TOS: maximizar throughput
- ➔ FTP envía su información de control fuera de banda.
  - HTTP envía su información de control en banda: con los datos.
- El servidor FTP mantiene información de estado del usuario
  - HTTP es un protocolo sin estado (más sencillo).

FTP


Cliente FTP
Usuario
Interfaz de usuario
Intérprete de protocolo
Función de transferencia de datos
Sistema de ficheros
Conexión de control
Comandos FTP, respuestas
Conexión de datos
Servidor FTP
Intérprete de protocolo
Función de transferencia de datos
Sistema de ficheros

FTP

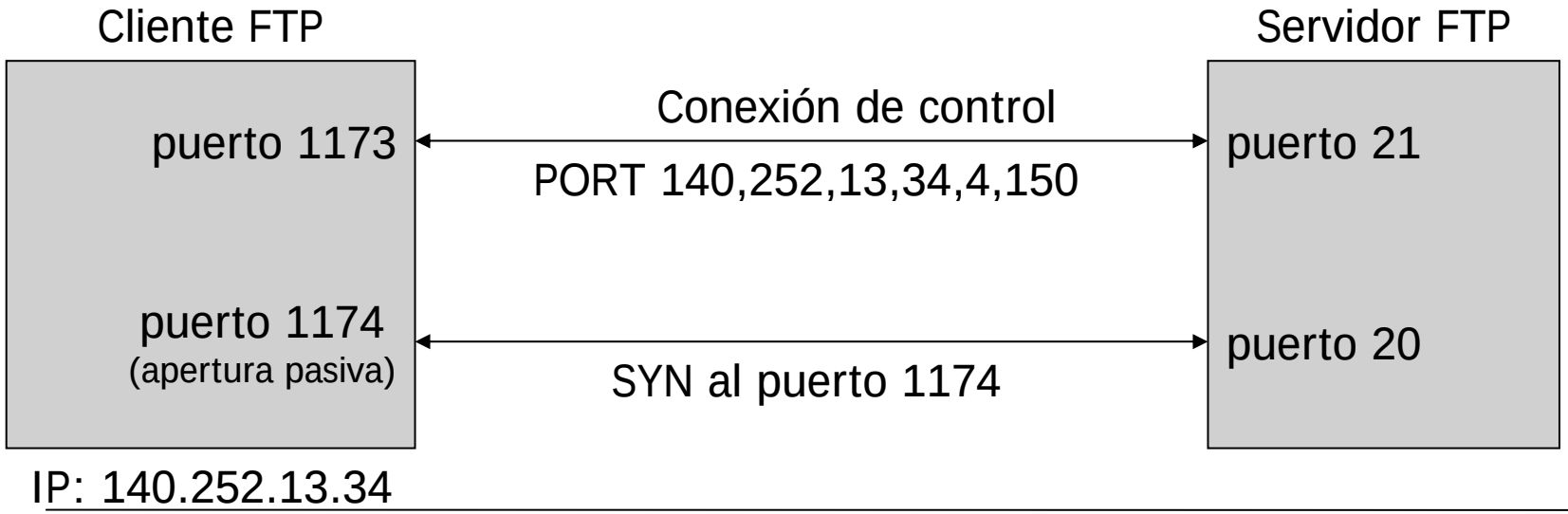
- Los comandos y respuestas se envían a través de la conexión de control como texto ASCII.
- Comandos FTP:
  - USER <nombre de usuario>
  - PASS <password>
  - LIST <directorio o lista ficheros>
  - RETR <nombre de archivo>
  - STOR <nombre de archivo>
  - PORT <n1, n2, n3, n4, n5, n6>
  - QUIT
  - SYST
  - TYPE <tipo>

FTP

- Respuestas: números de 3 dígitos en ASCII (para ordenador) + un mensaje opcional (para usuarios).
- Respuestas:
  - 1YZ: respuesta positiva preliminar ➔ la acción ha empezado pero se debe esperar otra respuesta antes de enviar otro comando.
  - 2YZ: respuesta positiva completa ➔ se puede enviar otro comando.
  - 3YZ: respuesta positiva intermedia ➔ se ha aceptado el comando pero se debe enviar otro comando.
  - 4YZ: respuesta negativa completa transitoria ➔ el comando no se pudo ejecutar debido a un error transitorio, se reintentará más tarde.
  - 5YZ: respuesta negativa completa permanente ➔ el comando no se aceptó y no será reintentado.
- Por ejemplo:
  - 200 Command OK
  - 331 Username OK, password requiered
  - 425 Can’t open data connection
  - 500 Syntax error (unrecognized command)

FTP

- Conexión de datos:
  - Envío de un fichero desde el cliente al servidor.
  - Envío de un fichero desde el servidor al cliente.
  - Envío de un listado de ficheros o directorios desde el servidor al cliente.
- Cada transferencia requiere una nueva conexión de datos:
  - La creación de la conexión de datos se realiza en el cliente.
  - El cliente selecciona un puerto efímero y realiza una apertura pasiva.
  - El cliente envía el número de puerto al servidor a través de la conexión de control, usando el comando PORT.
  - El servidor recibe el número de puerto y realiza una apertura activa. El servidor utiliza el puerto 20.

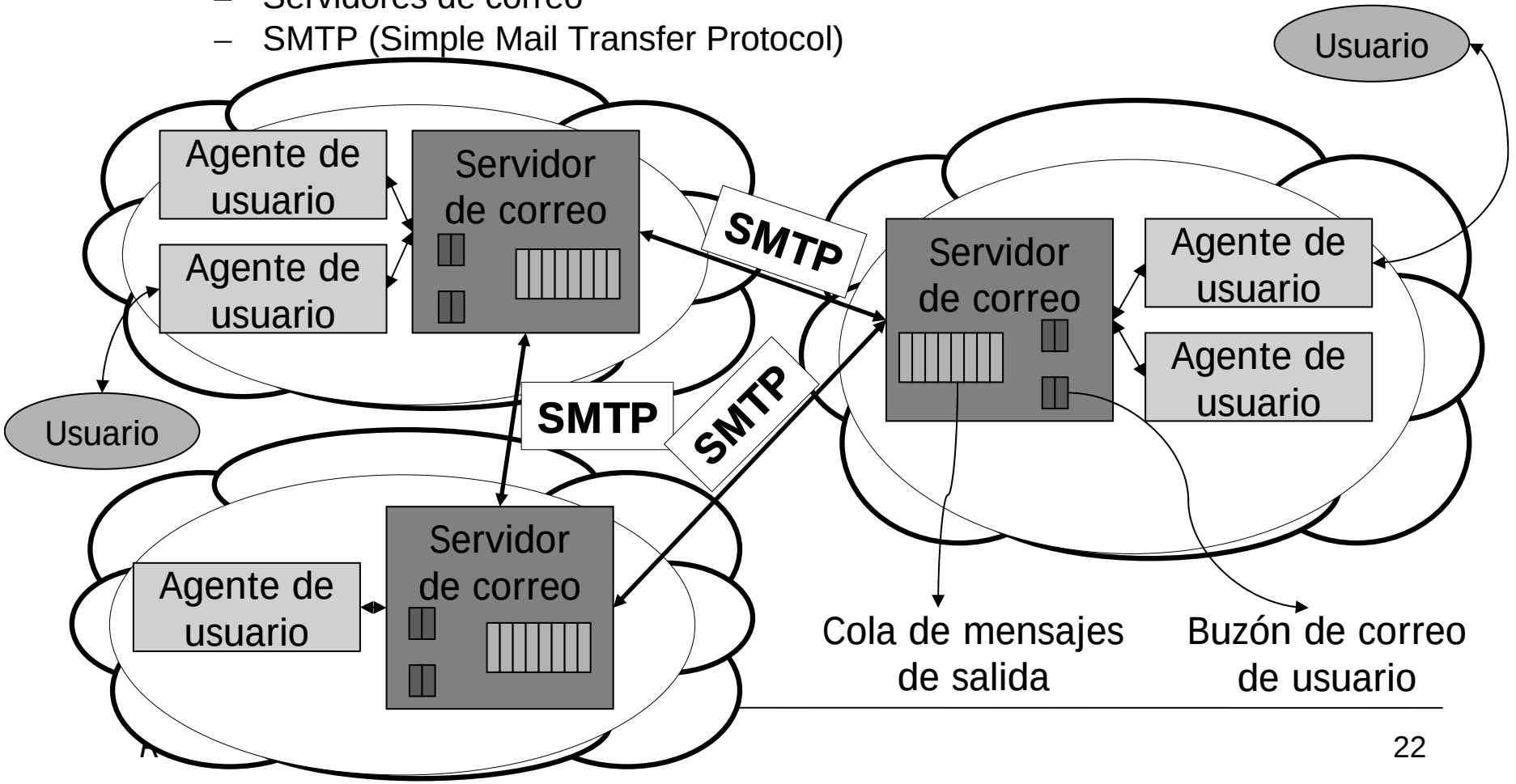


FTP activo y pasivo

- FTP modo activo:
  - El cliente FTP abre un puerto aleatorio (> 1023) – Apertura pasiva
  - El cliente envía al servidor FTP el número de puerto utilizando la conexión de control
  - El cliente espera a que el servidor abra la conexión
  - El servidor FTP inicia la conexión al puerto del cliente desde el puerto 20
- FTP modo pasivo:
  - El cliente envía el comando PASV al servidor
  - El servidor FTP abre un puerto aleatorio (> 1023) – Apertura pasiva
  - El servidor envía al cliente el número de puerto, sobre la conexión de control (utilizando el comando PORT)
  - El servidor espera a que el cliente abra la conexión
  - El cliente FTP inicia la conexión desde un puerto aleatorio (> 1023)
- El modo pasivo ha sido desarrollado para permitir las conexiones FTP a través de un firewall.
  - Las conexiones FTP desde un navegador se basan en el modo pasivo (se requiere que el servidor FTP lo soporte).

Correo electrónico

- Medio asíncrono de comunicación: los usuarios envían y reciben mensajes sin tener que coordinarse con otros usuarios.
- Componentes:
  - Lectores de correo o agentes de usuario
  - Servidores de correo
  - SMTP (Simple Mail Transfer Protocol)

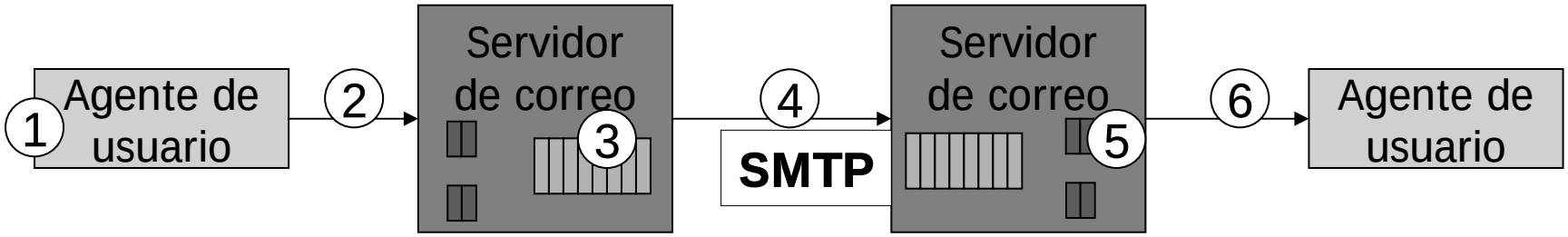


SMTP

- Definido en el RFC 2821.
- Permite el intercambio de mensajes entre servidores de correo.
  - El remitente actúa como cliente
  - El destinatario actúa como servidor
- El cliente SMTP establece una conexión TCP con el puerto 25 del servidor SMTP
  - Si el servidor está fuera de servicio se intentará más tarde.
- Se realiza la sincronización entre emisor y receptor
  - Se indica la dirección de correo electrónico del remitente
- El cliente envía el mensaje
  - Este proceso se repite si hay más mensajes (para el mismo servidor) y se cierra la conexión TCP.

- SMTP utiliza mensajes en formato ASCII (sólo texto) ➔ Si el mensaje tiene caracteres no ASCII o binarios ➔ Tiene que ser codificado (MIME)
- Es un protocolo de oferta (el cliente envía al servidor), frente a HTTP que es un protocolo de demanda.

SMTP



- Proceso de envío de un mensaje de correo electrónico:
  1. El cliente, mediante su lector de correo, crea el mensaje (p.e. para fidel@udc.es).
  2. El lector de correo envía el mensaje al servidor de correo del emisor.
     - Se almacena en la cola de mensajes.
  3. El servidor de correo (actuando como cliente SMTP) se conecta al servidor de correo del remitente (mail.udc.es)
  4. El cliente SMTP envía el mensaje.
  5. El servidor SMTP recibe el mensaje y lo almacena en el buzón del destinatario.
  6. El destinatario utiliza su lector de correo para obtener el mensaje.

SMTP

- Comandos básicos:
  - HELO: el cliente se identifica mediante este comando. Debe incluir su nombre de dominio absoluto.
  - MAIL: identifica al remitente del mensaje.
  - RCPT: identifica al destinatario del mensaje.
  - DATA: se incluye el contenido del mensaje (finaliza con un “.”).
  - QUIT: finaliza el intercambio de correo.
- Más comandos:
  - RSET: aborta el intercambio de correo y los dos extremos se resetean.
  - VRFY: el cliente consulta al servidor sobre una dirección de correo.
  - NOOP: fuerza al servidor a responder con el código de respuesta 200 (OK).
  - TURN: cliente y servidor se intercambian.

SMTP

```
bash-2.03$ telnet mail.udc.es 25
Trying 193.147.41.3...
Connected to unica.udc.es.
Escape character is '^]'.
220 mail.udc.es ESMTP "Servidor de correo del SIAIN"
HELO udc.es.
250 mail.udc.es
MAIL From:<fidel@udc.es>
250 Ok
RCPT To:<fidel@udc.es>
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
From: fidel@udc.es
To: fidel@udc.es
Subject: Prueba

Esto es un correo electrónico.
.
250 Ok: queued as 3E0E430CAA
QUIT
221 Bye
```

Formato correo electrónico

- Un mensaje de correo electrónico consta de dos partes: cabecera y cuerpo (separadas por una línea en blanco).
  - Cabecera: información sobre el correo
  - Cuerpo: el propio correo electrónico
- Algunos campos de la cabecera son:
  - **From**: sólo una por mensaje. Pueden usarse estos formatos:
    - Fidel Cacheda fidel@udc.es
    - fidel@udc.es
    - fidel@udc.es (Fidel Cacheda)
  - **To**: una o más por mensaje.
  - **Cc** y **Bcc**
  - **Subject**: tema del mensaje.
  - **Date**: fecha y hora en que el mensaje fue enviado.
  - **Message-Id**: identificador de cada mensaje, insertado por el ordenador remitente.
  - **Received**: información sobre el envío del mensaje, como las máquinas por las que pasó el mensaje.
  - **Reply-To**: dirección a la que se debe responder (no tiene porque coincidir con la del remitente).

MIME

- Multipurpose Internet Mail Extensions
- Definido en el RFC 822, y actualizado en los RFC 1341 y 1521.
- Permite enviar contenidos distintos de texto ASCII en mensajes de correo electrónico.
  - Mensajes en idiomas con acentos (español, francés y alemán).
  - Mensajes en alfabetos no latinos (hebreo y ruso).
  - Mensajes en idiomas sin alfabetos (chino y japonés).
  - Mensajes que no contienen texto (audio y vídeo).
- Sólo afecta a los agentes de usuario, ya que para SMTP es transparente.
- Campos MIME:
  - MIME-Version:
  - Content-Description: cadena de texto que describe el contenido. Es necesaria para que el destinatario decida si descodificar y leer el mensaje.
  - Content-Id:
  - Content-Transfer-Encoding: indica la manera en que está codificado el cuerpo del mensaje.
  - Content-Type: tipo del cuerpo del mensaje

MIME

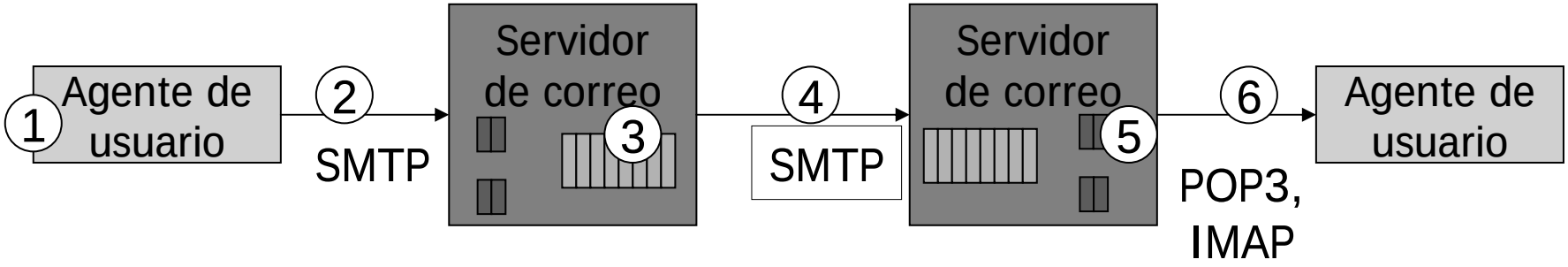
- Content-Type: especifica la forma del cuerpo del mensaje.
  - Existen 7 tipos definidos en el RFC 1521, cada uno de los cuales tiene uno o más subtipos.
    - El tipo y el subtipo se separan mediante un carácter diagonal (/).
- La lista de tipos y subtipos está cambiando continuamente, pero estos son algunos de los más utilizados:
  - text/plain, text/richtext y text/html
  - image/gif y image/jpeg
  - application/octet-stream, application/postscript y application/msword
  - audio/basic y video/mpeg
  - multipart: un mensaje contiene varios objetos o partes
    - multipart/mixed: varias partes de distinto tipo.
    - multipart/alternative: mismo mensaje en distintas codificaciones.
    - multipart/parallel: todas las partes se ven simultáneamente.
    - multipart/digest: varios mensajes unidos.

MIME

- Esquemas de codificación:
  - ASCII de 7 bits y ninguna línea excederá de 1000 caracteres.
  - ASCII de 8 bits y ninguna línea excederá de 1000 caracteres (viola el protocolo original del correo electrónico).
  - Binaria: utiliza los 8 bits y no respetan el límite de 1000 caracteres por línea.
    - Los programas ejecutables caen en esta categoría. No se da ninguna garantía de que los mensajes en binario lleguen correctamente.
  - Base64: se dividen grupos de 24 bits en unidades de 6 bits (26 mayúsculas, 26 minúsculas, 10 dígitos, “+” y “/”), enviándose cada unidad como carácter ASCII legal.
  - Quoted-printable: ASCII de 7 bits, con todos los caracteres por encima de 127 codificados como un signo de igual seguido del valor del carácter en dos dígitos hexadecimales.
    - Se utiliza en el caso de mensajes que son casi completamente ASCII, pero con algunos caracteres no ASCII. En este caso la codificación base64 es ineficiente.

Protocolos de acceso al correo

- ¿Cómo se comunican los lectores de correo con los servidores de correo (pasos 2 y 6)?



- El lector del emisor puede utilizar SMTP, pero el lector del receptor no ➔ POP3, IMAP.

Protocolos de acceso al correo

- POP3 – Post Office Protocolv3: definido en RFC 1939.
- Protocolo de acceso al correo muy simple.
- Modo de operación en tres fases:
  - Autorización: login y password
  - Transacción: recuperar los mensajes, marcar para borrado y estadísticas de correo.
  - Actualización: cuando finaliza la sesión, el servidor de correo borra los mensajes marcados.
- Dos configuraciones del cliente POP3:
  - Descargar y borrar
  - Descargar y guardar

- IMAP – Internet Mail Access Protocol: RFC 2060.
- Permite crear y gestionar buzones remotos (en el servidor de correo).
- IMAP asocia cada mensaje con un buzón.
  - Inicialmente al INBOX
- Proporciona comandos para crear buzones, mover mensajes, buscar mensajes.
- IMAP mantiene información de estado de los usuarios entre sesiones (nombres buzones, mensajes, ...)
- Dispone de comandos para recuperar componentes de los mensajes.
  - P.e. sólo las cabeceras

POP3

- Principales comandos POP3:
  - USER <nombre>: envía el identificador de usuario.
  - PASS <password>: envía la clave del usuario al servidor.
  - STAT: devuelve el número total de mensajes almacenados en el buzón y su longitud total.
  - LIST: devuelve un listado de todos los mensajes no borrados con su longitud.
  - RETR <identificador>: recupera el mensaje especificado en el campo identificador.
  - DELE <identificador>: marca para borrado el mensaje especificado. Todos los mensajes marcados serán borrados cuando se cierre la conexión.
  - RSET: recupera los mensajes borrados (mientas la conexión está establecida).
  - TOP <identificador> <líneas>: muestra la cabecera y el número de líneas indicado del mensaje especificado.
  - QUIT: salir