

Tema 5

Aplicaciones distribuidas en Internet

Profesores:

Mercedes Amor

Inmaculada Ayala

Javier Cámara

Daniel Muñoz

Lidia Fuentes

Delia Rico

Gabriel Luque

Francisco Rus

Contenido del tema

- Servicios clásicos de Internet
 - Sistema de Nombres de Dominio - DNS
 - Terminal Virtual
 - Transferencia de Ficheros
 - Correo Electrónico
 - La World Wide Web
- Servicios avanzados en Internet
 - ~~Servicios P2P~~ x UO CAE
 - Servicios multimedia

Tema 1. Introducción a las redes y sistemas distribuidos

Tema 2. Técnicas de acceso y control de enlace

Tema 3. Protocolos de Interconexión de Redes

Tema 4. Servicios básicos para el nivel de transporte en Internet

Tema 5. Aplicaciones distribuidas en Internet

- Sistema de Nombre de Dominio – DNS
- Terminal Virtual
- Transferencia de Ficheros
- Correo Electrónico
- La World Wide Web

SERVICIOS CLÁSICOS DE INTERNET

Protocolos a nivel de aplicación

- Objetivo básico
 - Los protocolos a nivel de aplicación definen cómo los procesos de aplicación, que se ejecutan en sistemas diferentes, se envían mensajes entre sí
- Protocolos propietarios y protocolos de dominio público
 - Los protocolos de aplicaciones de red de gran número de empresas son propietarios
 - Los protocolos relacionados con Internet son de dominio público. Se definen en documentos denominados **Requests for Comments (RFCs)**
 - [RFC 2821]: **SMTP** (Simple Mail Transfer Protocol)
 - [RFC 2616]: **HTTP** (HyperText Transfer Protocol)

Servicios necesarios para una aplicación de red

- Varios aspectos a considerar
 - **Fiabilidad de la transmisión** Conexión Constante (Sin caídas)
 - Fiable / no fiable
 - **Ancho de banda (bandwidth)**
 - Requisitos mínimos / flexible
 - **Requisitos de latencia (timing)**
 - Sensibles / no sensibles a la latencia
 - **Seguridad**

Servicios necesarios para una aplicación de red

- Ejemplos de aplicaciones:

Aplicación	Pérdida de datos	Ancho de banda	Timing
Transferencia de ficheros	No	Flexible	No
Correo electrónico	No	Flexible	No
Páginas Web	No	Flexible (varios Kbps)	No
Audio/video en tiempo real	Sí	Audio: kbps – 1Mbps Video: 10kbps – 5 Mbps	100s de mseggs
Mensajería instantánea	No	Flexible	Pocos segs

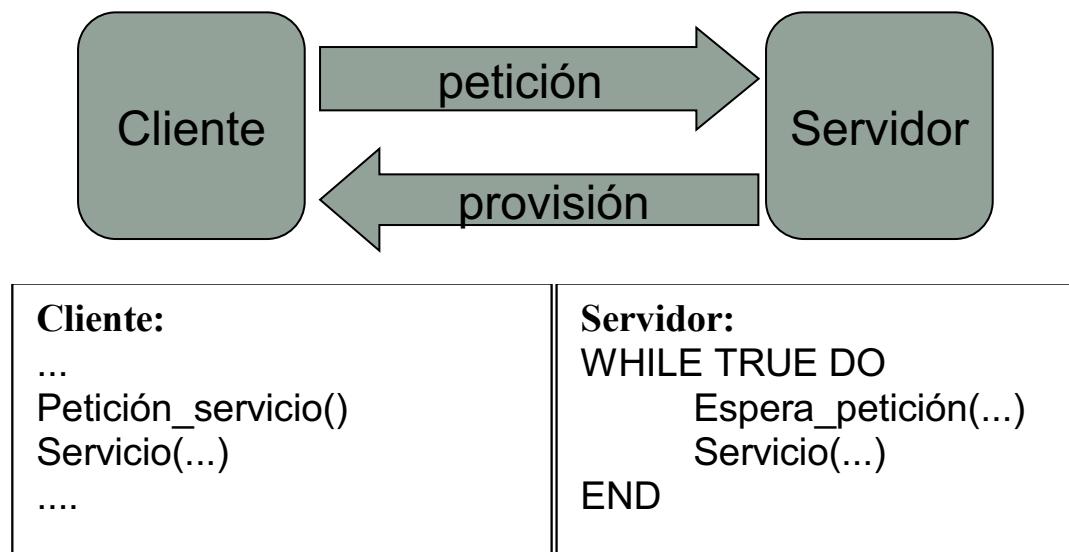
Modelos de aplicaciones

- Dos modelos básicos:
 - Cliente/servidor
 - ~~Peer to Peer (P2P)~~
- Modelo cliente/servidor:
 - Las aplicaciones se componen de dos entidades: clientes y servidores (no son entidades excluyentes!)
 - Los servidores son entidades que ofrecen servicios
 - Los clientes solicitan los servicios de los servidores
- Modelo P2P:
 - No existen distinción de roles entre las entidades de la aplicación

→ esto es un modelo de aplicación

Modelo cliente/servidor

- Esquema:



- Ejemplos:
 - Telnet: terminales virtuales remotos
 - FTP: transferencia de ficheros
- Un sistema puede implementar a su vez un cliente y un servidor

Paradigma cliente-servidor

- Los **procesos servidores**:
 - Son procesos **permanentemente activos** que ofrecen un servicio concreto siempre disponible para los usuarios
 - El **servidor tiene una dirección IP fija y un puerto conocido por el usuario**
 - Los **procesos clientes** → Finalizan una vez obtienen respuesta del servidor o no (TCP)
 - Piden un servicio en un momento dado
 - Se comunican sólo con el servidor
 - Clientes y servidores de una misma aplicación se comunicarán mediante el **intercambio de mensajes** utilizando los servicios del **nivel de Transporte**
 - El **envío y recepción de mensajes** se realiza a través de **sockets**
- En UDP se mantiene zombie esperando, puesto que envía el datagrama el cliente y se despreocupa del servidor.

Direccionamiento

- Puertos
 - Vienen identificados por números enteros
 - Muchos números están asignados a aplicaciones de red concretas
- Fichero /etc/services

Servicio	Puerto	Puerto (TLS/SSL)	Protocolo
ftp-data	20/tcp	989/tcp	File Transfer [Default Data]
ftp	21/tcp	990/tcp	File Transfer [Control]
ssh	22/tcp		Acceso remoto seguro
telnet	23/tcp	992/tcp	Telnet
smtp	25/tcp	465/tcp	Correo electrónico
dns	53/udp		Resolución de nombres
dhcp	67/udp	-	Configuración dinámica
http	80/tcp	443/tcp	World Wide Web HTTP
pop3	110/tcp	995/tcp	Recuperación de e-correo
imap	143/tcp	993/tcp	Recuperación de correo

Servicios ofrecidos por la capa de transporte de Internet

- Dos protocolos de transporte:
 - TCP (Transmission Control Protocol)
 - UDP (User Datagram Protocol)
- Servicios TCP
 - Servicio orientado a la conexión (connection-oriented)
 - Servicio de transporte fiable
- Servicios UDP
 - Servicio sin conexión (connectionless)
No espera respuesta del servidor. Simplemente envía el datagrama y se despreocupa.
 - Servicio de transmisión no fiable
- TCP y UDP no garantizan tasas mínimas de transmisión

Ejemplos de aplicaciones y protocolos que usan

Aplicación	Protocolo de aplicación	Protocolo de transporte
Correo electrónico	SMTP [RFC 2821]	TCP
Terminal remoto	Telnet [RFC 854]	TCP
Acceso a la Web	HTTP [RFC 2616]	TCP
Transferencia de ficheros	FTP [RFC 959]	TCP
Multimedia (<i>streaming</i>)	Propietario	TCP o UDP
Telefonía por Internet (VoIP)	Propietario	Típicamente UDP
Resolución de Nombres	DNS	UDP

TCP y SSL

- TCP y UDP no ofrecen encriptación de datos
- Secure Sockets Layer (SSL)
 - Es una mejora sobre TCP implementada en la capa de aplicación
- SSL proporciona:
 - **Encriptación:** Ningún elemento intermedio entiende el mensaje
 - **Integridad de datos:** Nadie modifica el mensaje sin ser detectado
 - **Autenticación:** El destino es quién dice ser

Tema 1. Introducción a las redes y sistemas distribuidos

Tema 2. Técnicas de acceso y control de enlace

Tema 3. Protocolos de Interconexión de Redes

Tema 4. Servicios básicos para el nivel de transporte en Internet

Tema 5. Aplicaciones distribuidas en Internet

SERVICIO DE NOMBRES (DNS)

Sistema de Nombres de Dominio

- Los dispositivos se identifican por la dirección IP
- Los usuarios utilizan “nombres” :
 - www.uma.es, www.nasa.gov, ...
- ¿Cómo se establece la correspondencia entre direcciones IP y estos nombres?
- Mediante el **sistema de nombres de dominio**
 - Es una base de datos distribuida implementada por una jerarquía de servidores de nombres
- Mediante un protocolo de la capa de aplicación: los dispositivos y los servidores de nombres se comunican para *resolver* nombres (traducción dirección/nombre)

Es una función básica de Internet que se implementa como una aplicación

- Funciona sobre UDP (puerto 53)

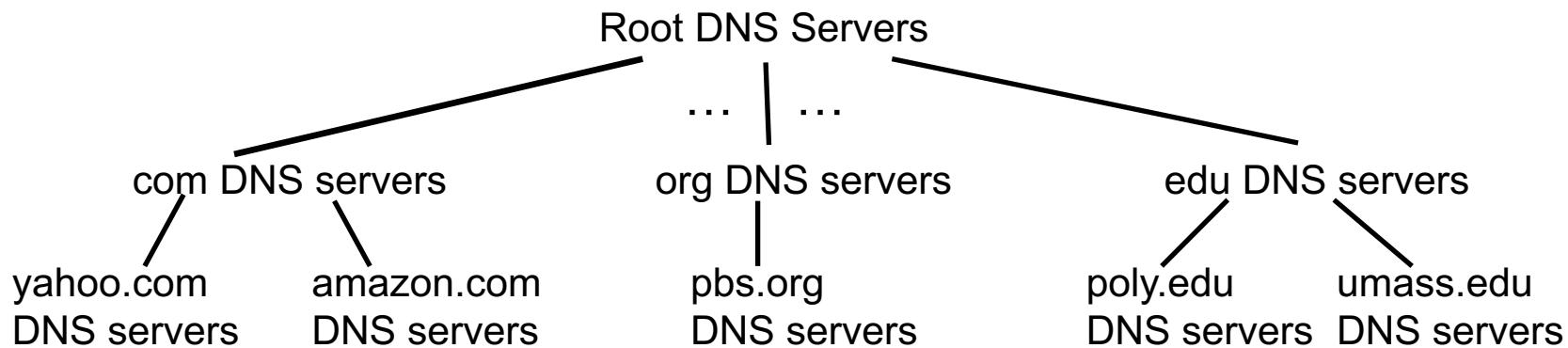
DNS: Hosts, dominios, FQDN

- **FQDN: Fully Qualified Domain Name:**
 - Nombre completo con equipo y dominios
 - Máximo: 255 caracteres (63 por dominio)



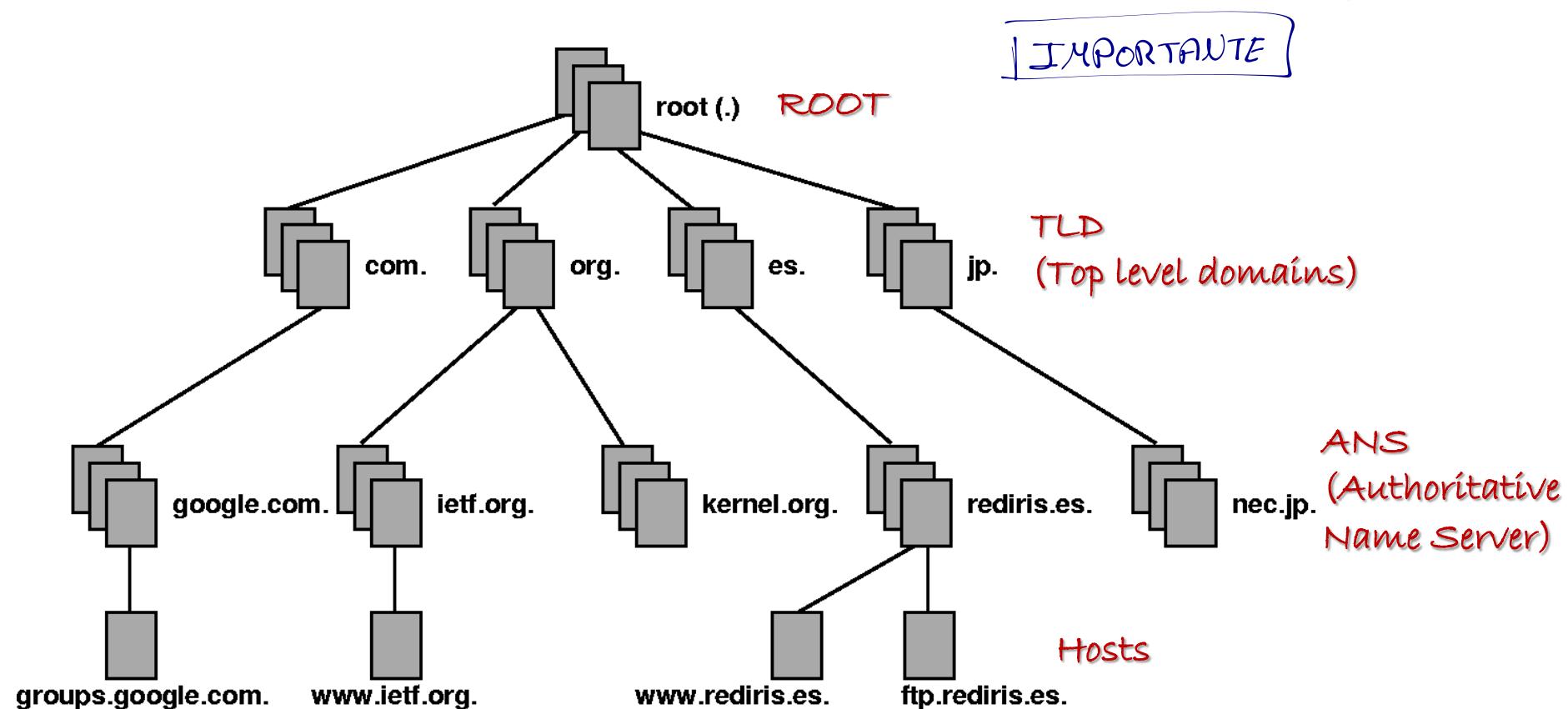
DNS: Servicios y estructura

- **Servicios:**
 - Traducción de nombre de host a dirección IP
 - Mantenimiento de alias
 - Balanceo de carga:
 - Hay servidores Web replicados que tienen asociadas varias dirección IP a un mismo nombre
- **Estructura:**
 - Jerárquica
 - ¿Por qué no una única base de datos centralizada?



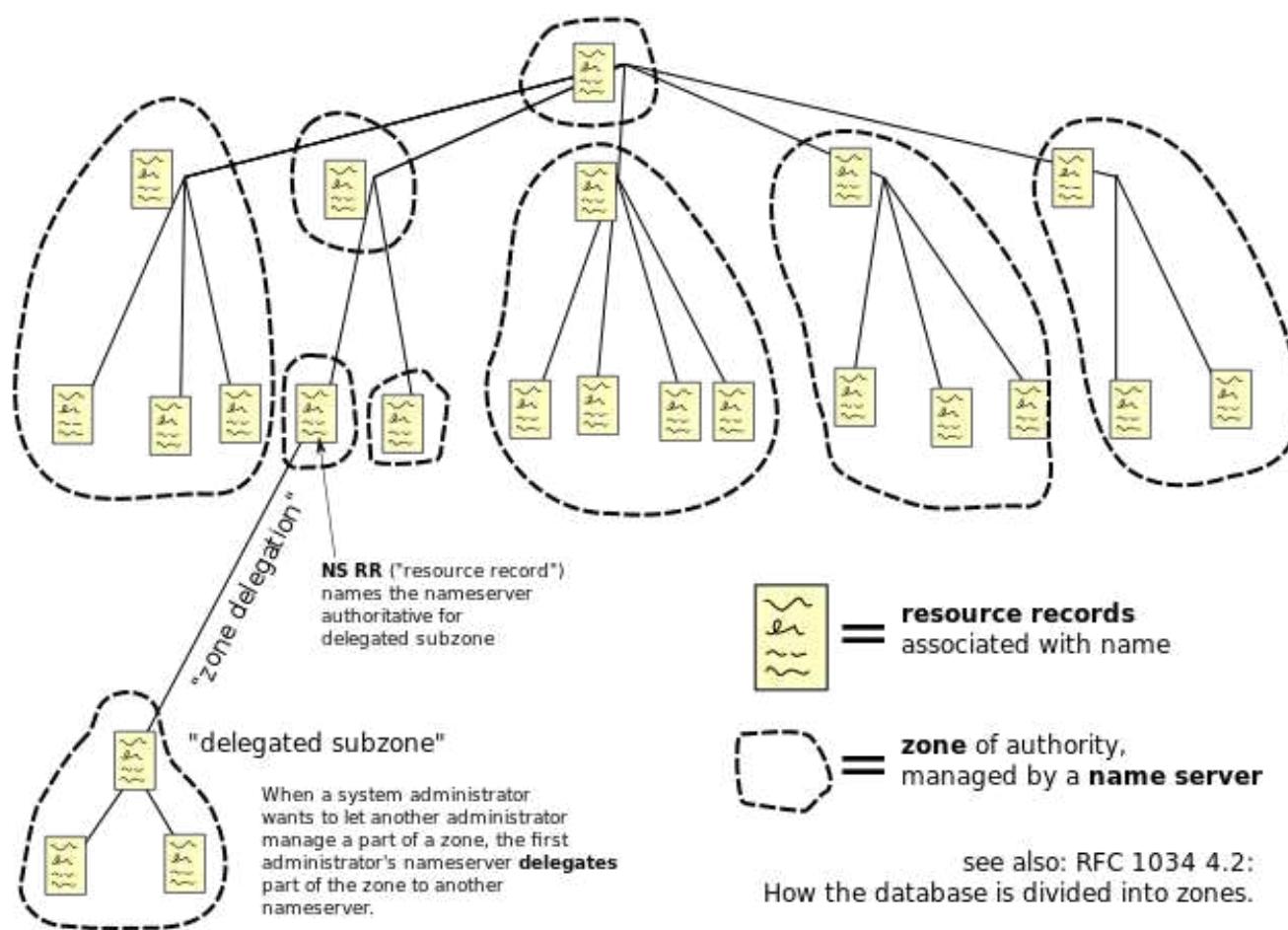
DNS: Servicios y estructura

- Cada nodo tiene ciertos registros (equivalencia nombre e IP) y sabe a quién consultar sus subdominios (hijos)



DNS: Servicios y estructura

- Un mismo organismo (servidor) puede tener almacenado varios nodos del árbol (**zona**)

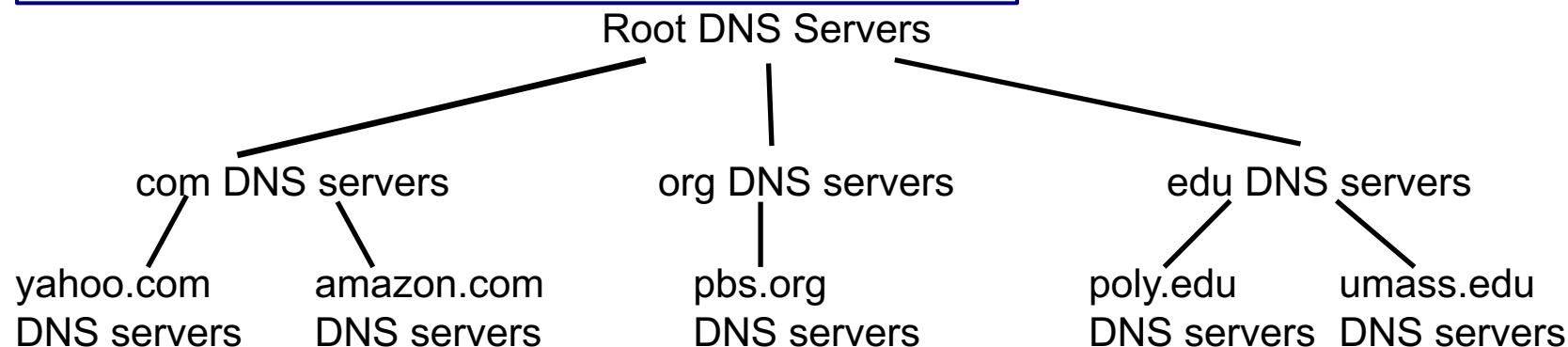


DNS: Tipos de servidores

- **Tipos:**

- Servidores raíz (root)
- Servidores top-level domain (TLD)
- Servidores DNS autorizados (ANS)
- Servidores locales (LNS)

Importante



- **Ejemplo de uso:** www.amazon.com

- Consultamos nuestro servidor local (**LNS**)
- LNS solicita **root** server encontrar el servidor DNS (**TLD**) de .com
- LNS solicita al TLD .com obtener el servidor DNS (**ANS**) amazon.com
- LNS pregunta al ANS amazon.com la dirección IP de www.amazon.com

DNS: Servidores de nombre raíz

- Servidor de nombre raíz (root name server):
 - Contacta con un servidor de nombres con autoridad si no se conoce una asignación de nombres



DNS: TLD, Servidores de Autoridad y Servidores Locales

Servidores top-level domain (TLD):

- Responsables de los dominios com, org, net, edu, aero, jobs, museums, y todos los dominio de paises: : uk, fr, ca, jp, es, ...

Servidores DNS autorizados:

- Servidores DNS propios de una organización que proporcionan traducciones autorizadas de los hosts de su organización
- Pueden ser mantenidos por la propia organización o por un proveedor de servicios

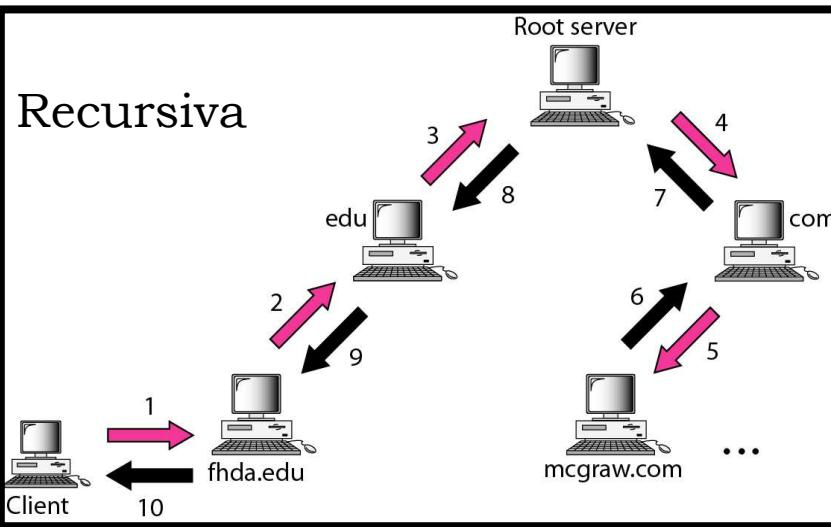
Servidores locales:

- No pertenece estrictamente a la jerarquía y cada ISP tiene uno:
 - Denominado “servidor de nombres por defecto”
- Cuando un host realiza una consulta (DNS query), ésta es enviada a su servidor DNS local
 - Tiene una cache local con las traducciones más recientes
 - Actúa de proxy, reenviado la consulta hacia la jerarquía

DNS: Resolución

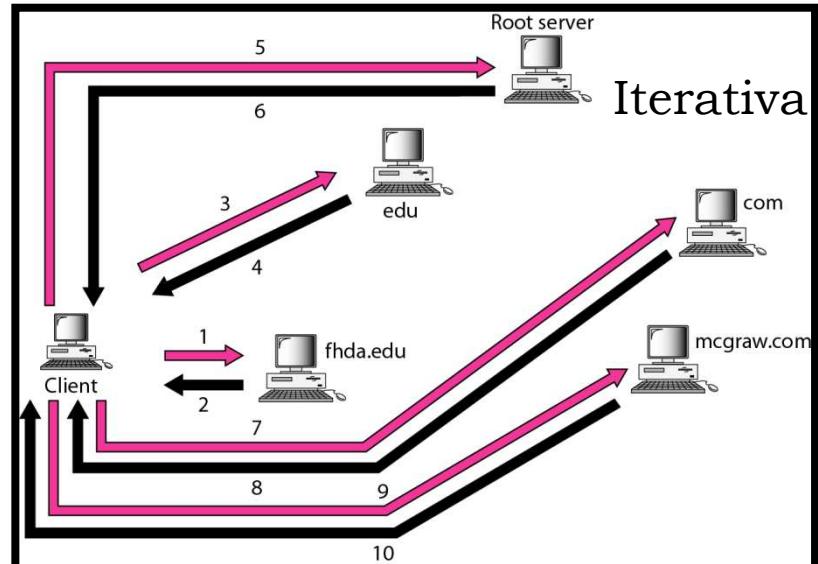
- Dos tipos de resoluciones
 - Recursiva (mucha carga en niveles superiores)
 - El servidor local debe entregar al cliente una respuesta final
 - Iterativa
 - Si el servidor es un servidor de autoridad, envía la respuesta
 - Si no lo es, devuelve al cliente la dirección IP del servidor que cree que puede resolver el nombre

Recursiva



Destino

Iterativa

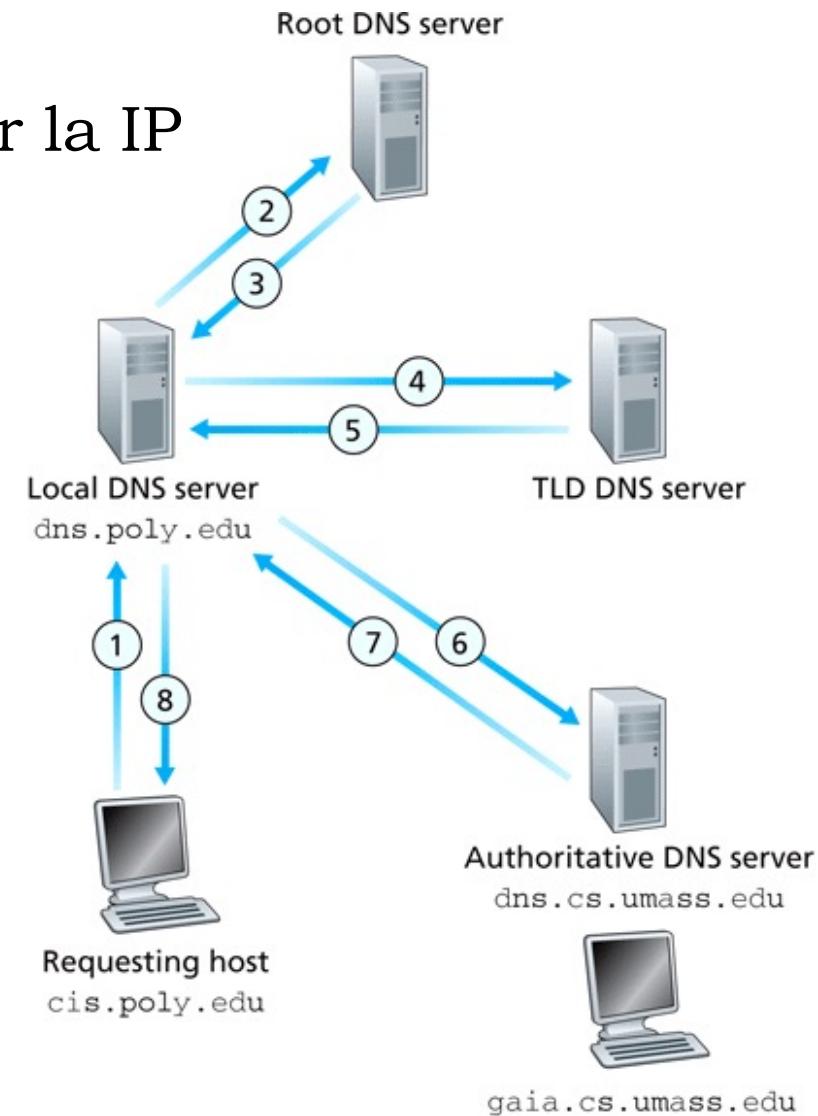


DNS: Mezcla de resolución iterativa/recursiva

- Ejemplo:

El host **cis.poly.edu** quiere saber la IP de **gaia.cs.umass.edu**

Consultas iterativas



Consulta recursiva

DNS: cacheado, actualización de registros

- Cuando un servidor de nombres aprende un nombre, lo almacena en cache
 - Las entradas de la cache son limpiadas periódicamente (top level domain (.com, .es...))
 - Los servidores TLD servers están normalmente guardados en los servidores de nombre
 - Haciendo que los servidores raíz no necesiten ser visitados
- Las entradas memorizadas pueden estar desfasadas
 - Cada entrada tiene un tiempo de vida (TTL) tras el cual expira.
- Los mecanismos para la actualización y notificación están recogidos en el RFC 2136

Tema 1. Introducción a las redes y sistemas distribuidos

Tema 2. Técnicas de acceso y control de enlace

Tema 3. Protocolos de Interconexión de Redes

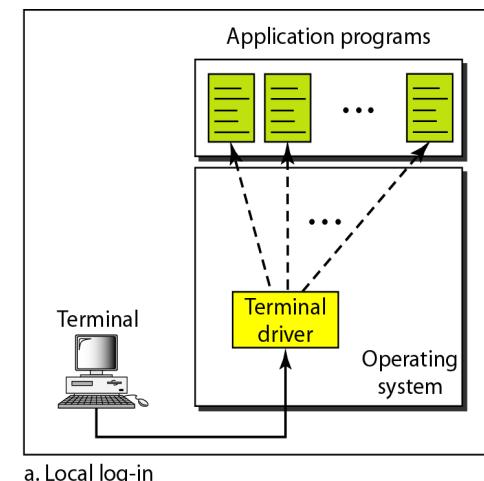
Tema 4. Servicios básicos para el nivel de transporte en Internet

Tema 5. Aplicaciones distribuidas en Internet

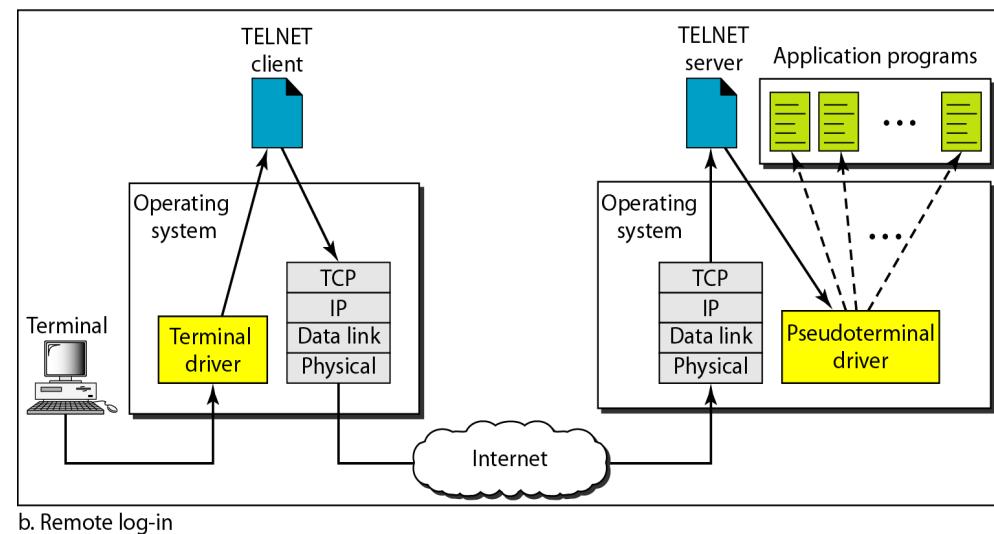
TELNET – TERMINAL VIRTUAL

Telnet

- Es una aplicación cliente-servidor de propósito general que permite el establecimiento de una conexión con un sistema remoto de forma que el terminal local aparece como un terminal del sistema remoto
- Protocolo/puerto: TCP/23



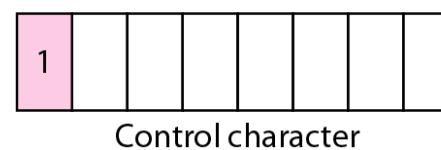
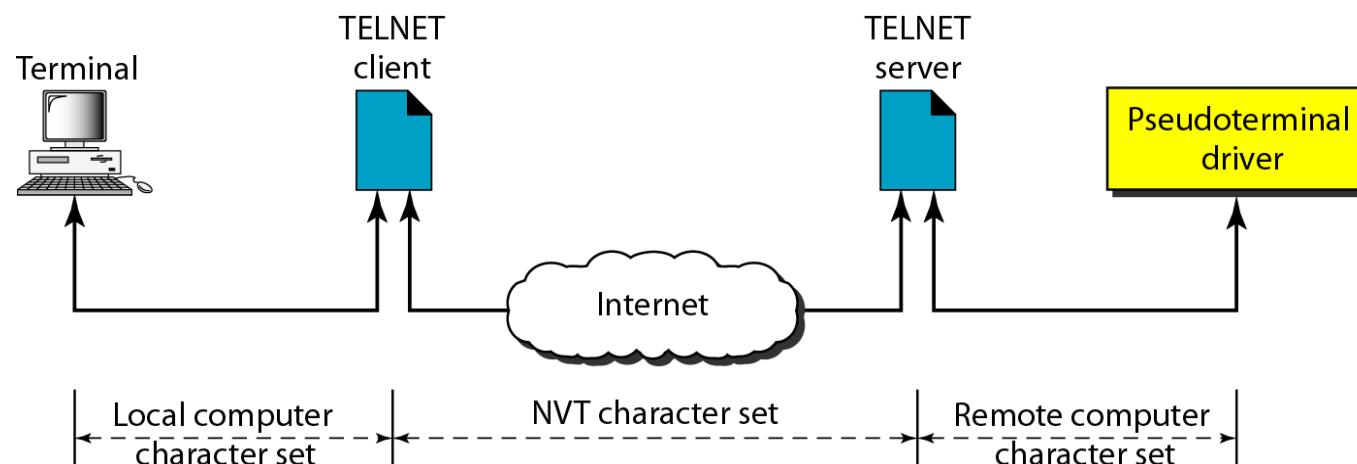
a. Local log-in



b. Remote log-in

Telnet

- Problemas debido a la heterogeneidad de los sistemas.
- Ej., fin de fichero -> Windows: Ctrl + z, Linux: Ctrl + d.
- Los caracteres viajan a través de la conexión en un conjunto de caracteres universal denominado Caracteres de terminal virtual de red (NVT).



Telnet

- **Telnet:**

- Desarrollado en 1969.
 - Sólo sirve para acceder en modo terminal.
 - Su mayor problema es de seguridad, ya que los **nombres de usuario y contraseñas viajan por la red como texto plano**
 - Esto facilita que cualquiera que espíe el tráfico de la red pueda obtener los nombres de usuario y contraseñas, además de la información.

- **SSH:** *Alternativa Moderna a "Telnet."*

- Desarrollado en 1995.
 - Protocolo de comunicación muy similar, pero en el que las **comunicaciones viajan de forma encriptada.**
 - Requiere **más recursos del ordenador.**
 - **Cifra la información antes de transmitirla.**
 - **Autentica la máquina a la cual se conecta.**
 - SSH suele trabajar en el puerto 22 (tcp)

Telnet

- **Otros usos:**

- El telnet puede utilizarse para establecer conexiones TCP a diferentes puertos:

telnet dirección puerto

- Permite realizar ciertas pruebas y comprobaciones:
 - Comprobar si un determinado puerto TCP está “abierto”.
 - Probar un servicio que se está desarrollado (si es “modo texto”)
 - Ciertos servicios lo utilizan (Ej. Interactuar con el emulador de Android).

```
mgcc112517:~ gpzpati$ telnet localhost 5554
Trying ::1...
telnet: connect to address ::1: Connection refused
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Android Console: type 'help' for a list of commands
OK
sms send 1234567890 This is sample SMS for the Android Simulator
OK
```

Tema 1. Introducción a las redes y sistemas distribuidos

Tema 2. Técnicas de acceso y control de enlace

Tema 3. Protocolos de Interconexión de Redes

Tema 4. Servicios básicos para el nivel de transporte en Internet

Tema 5. Aplicaciones distribuidas en Internet

INTERCAMBIO DE FICHEROS (FTP)

FTP: *File Transfer Protocol*

- Características
 - Es un protocolo que permite transferir ficheros entre ordenadores
 - Se basa en el modelo cliente/servidor
 - Usa el protocolo TCP
 - Los usuarios necesitan autorización
 - Dos modos de transferencia:
 - ASCII: Los datos son convertidos a ASCII de 8 bits para su envío (Tipo A).
 - Binario (también llamado modo Imagen): Los datos se envían byte a byte sin conversión (Tipo I).
 - Operaciones (usuario): open, bye, put, get, mput, mget

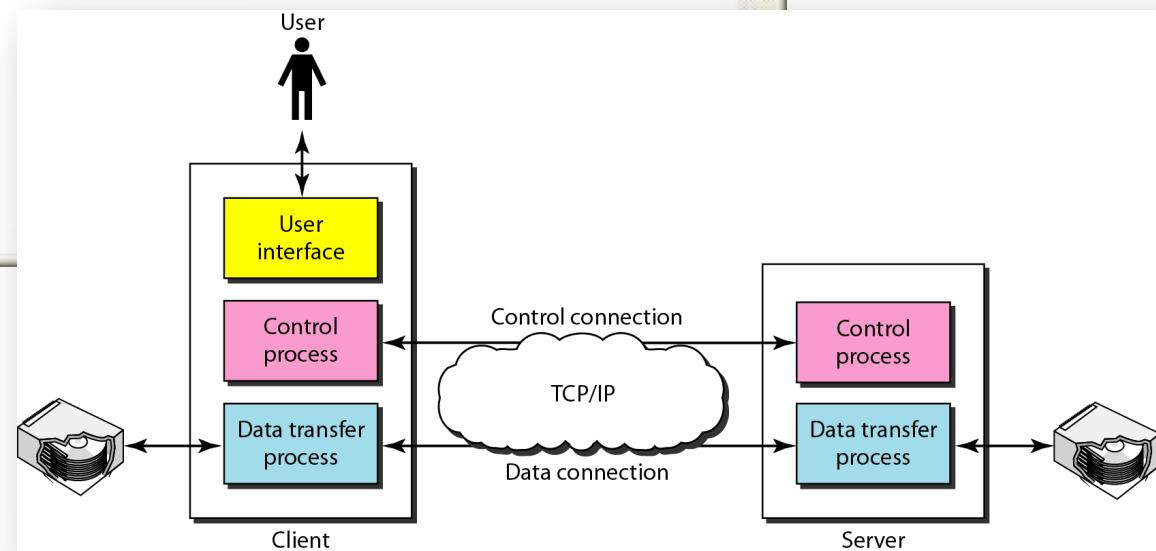
FTP

- Conexiones
 - Establece dos tipos de conexiones simultáneas entre los computadores involucrados en la transferencia de los ficheros:
 - Una conexión se utiliza para la transmisión de los ficheros en sí (TCP, puerto 20)
 - Otra conexión se dedica al envío de información de control (TCP, puerto 21)
 - Dos estrategias diferentes en cada conexión:
 - La de control permanece abierta mientras dura la sesión interactiva
 - Las de datos se crean y destruyen dinámicamente en cada transferencia de fichero

FTP

- Ejemplo de sesión

```
Antonio@Antur2 ~  
$ sftp antonio@192.168.255.128  
Connecting to 192.168.255.128...  
antonio@192.168.255.128's password:  
sftp> ls  
Desktop Examples hola.txt softw workspace  
sftp> pwd  
Remote working directory: /home/antonio  
sftp> get hola.txt  
Fetching /home/antonio/hola.txt to hola.txt  
/home/antonio/hola.txt 100% 8 0.0KB/s 00:00  
sftp> cd Examples  
sftp> pwd  
Remote working directory: /usr/share/example-content  
sftp> bye  
  
Antonio@Antur2 ~  
$ -
```





Modos de FTP

Activo

El cliente crea el socket para la transferencia de datos.

El cliente envía el puerto al servidor mediante un comando PORT

El servidor se conecta (puerto 20) y establece la conexión de datos.

Pasivo

El cliente envía el comando PASV
Conectarse y transferir archivos a un Servidor FTP Sin necesidad de una dirección IP pública única

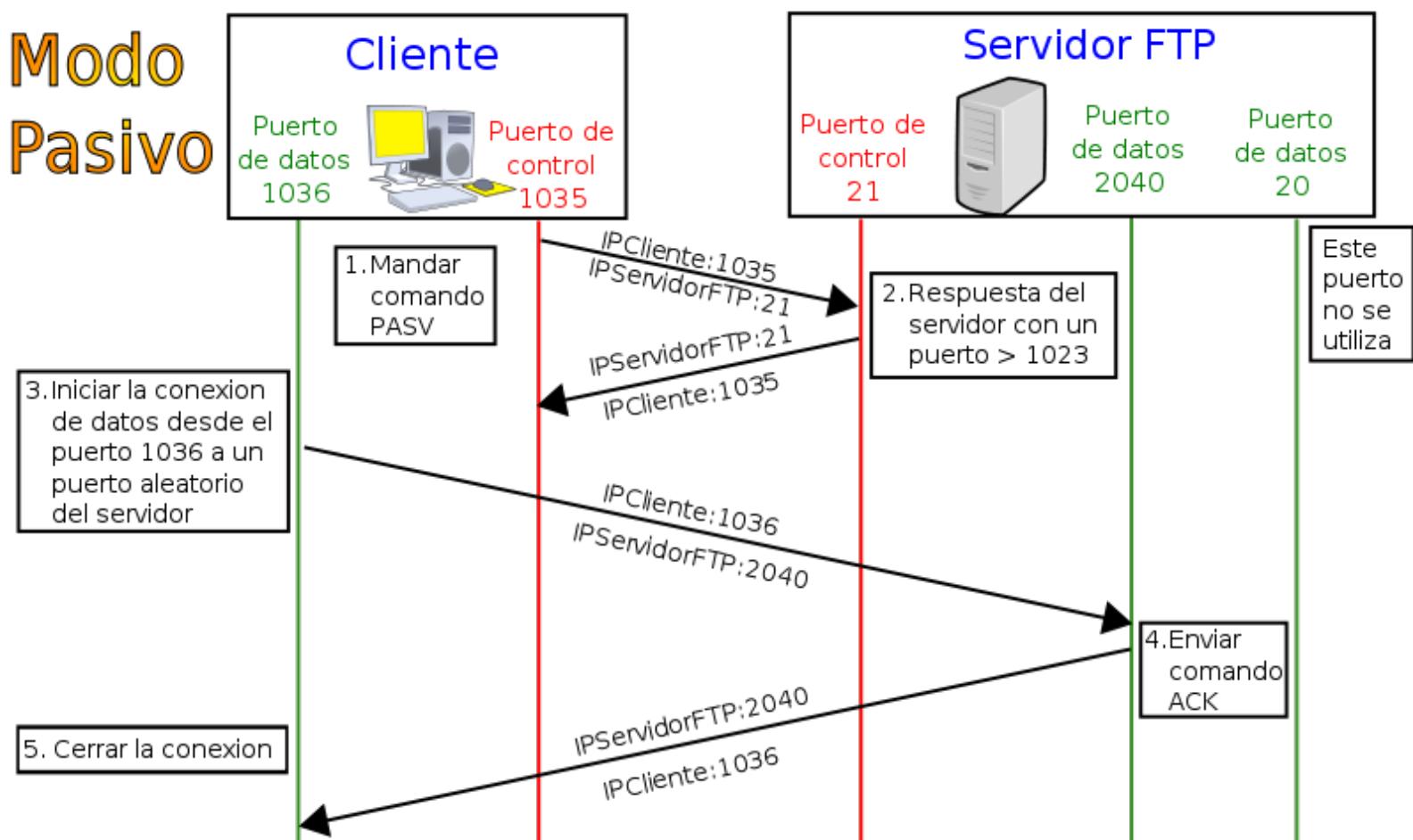
El servidor crea el socket para la transferencia de datos.

El servidor comunica el puerto al que debe conectarse.

El cliente se conecta a dicho puerto y se establece la conexión de datos.

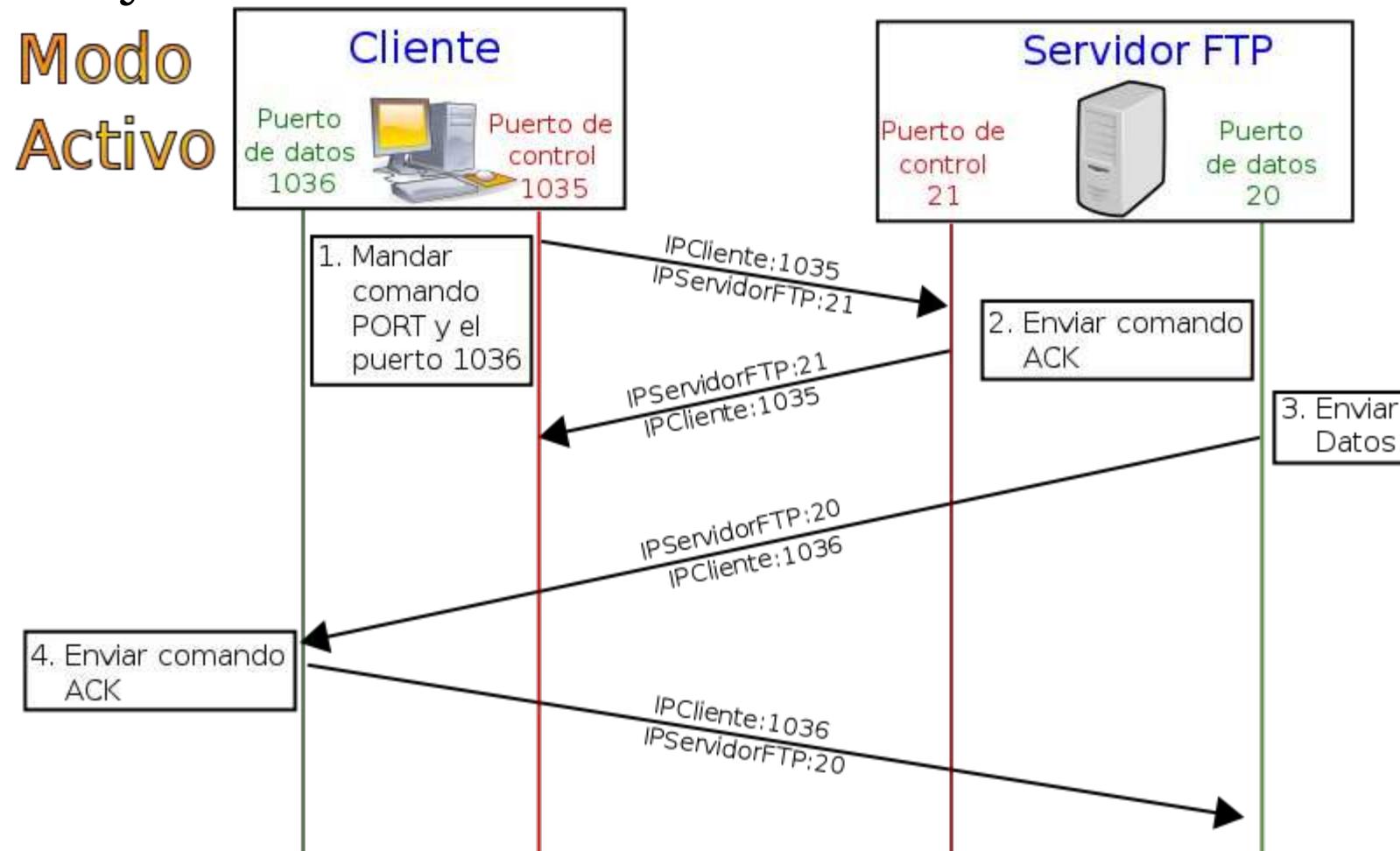
FTP: Modo Pasivo (PASV)

- El cliente envía **PASV**. Entonces el servidor crea un nuevo puerto e informa al cliente y éste último inicia la conexión.



FTP: Modo Activo (PORT)

- El cliente envía al servidor (**PORT**) el puerto de datos usado y el servidor inicia la conexión.



Mensajes FTP

- Mensajes (SP: espacio y CRLF: Salto de línea y retorno de carro):

COMANDO	[<SP> parametro]	<CRLF>
----------------	-------------------------------	---------------------

USER	<SP> <username>	<CRLF>
------	-----------------	--------

PASS	<SP> <password>	<CRLF>
------	-----------------	--------

CWD	<SP> <pathname>	<CRLF>
-----	-----------------	--------

QUIT		<CRLF>
------	--	--------

PORT	<SP> <host-port>	<CRLF>
------	------------------	--------

PASV		<CRLF>
------	--	--------

TYPE	<SP> <type-code>	<CRLF>
------	------------------	--------

MODE	<SP> <mode-code>	<CRLF>
------	------------------	--------

RETR	<SP> <pathname>	<CRLF>
------	-----------------	--------

STOR	<SP> <pathname>	<CRLF>
------	-----------------	--------

LIST	[<SP> <pathname>]	<CRLF>
------	-------------------	--------

DELE	[<SP> <pathname>]	<CRLF>
------	-------------------	--------

HELP	[<SP> <string>]	<CRLF>
------	-----------------	--------

NOOP		<CRLF>
------	--	--------

Respuestas FTP

- Si se requiere transmitir datos, se creará la conexión oportuna.
- Se enviará un mensaje de respuesta por la conexión de control:

XYZ <SP> Explicación <CRLF>

X = tipo

Y = función

Z = detalle

1yz<SP>Positive Preliminary reply<CRLF>

2yz<SP>Positive Completion reply<CRLF>

3yz<SP>Positive Intermediate reply<CRLF>

4yz<SP>Transient Negative Completion reply<CRLF>

5yz<SP>Permanent Negative Completion reply<CRLF>

Ej:

**250<SP>Command successful.<CRLF>
(La respuesta ocupa 24 bytes)**

Tema 1. Introducción a las redes y sistemas distribuidos

Tema 2. Técnicas de acceso y control de enlace

Tema 3. Protocolos de Interconexión de Redes

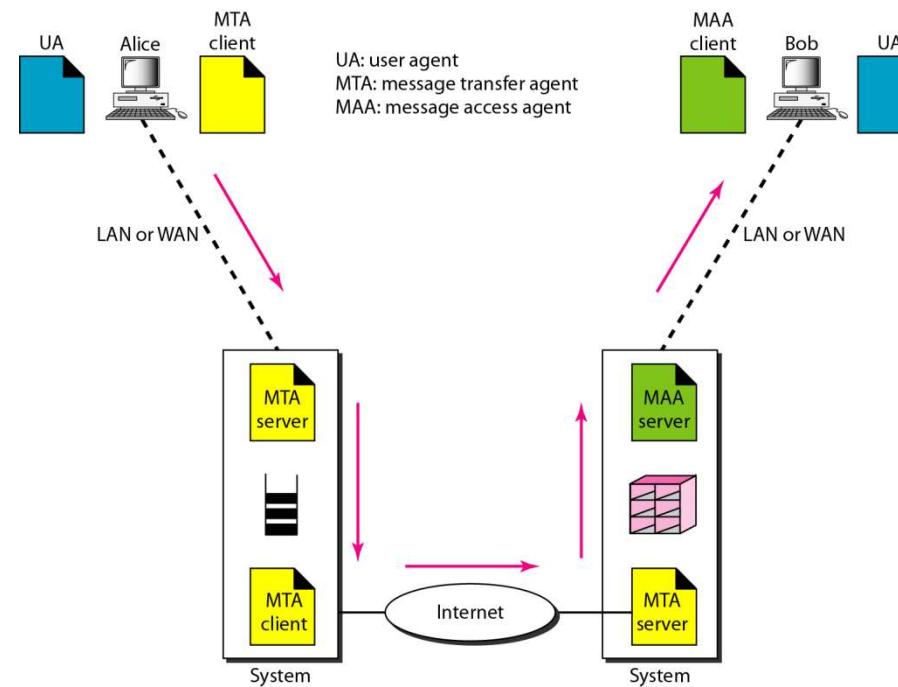
Tema 4. Servicios básicos para el nivel de transporte en Internet

Tema 5. Aplicaciones distribuidas en Internet

CORREO ELECTRÓNICO (SMTP Y POP3)

Correo electrónico

- Uno de los servicios más populares de Internet (y de los más antiguos)
- Hay tres componentes involucrados en el envío y recepción de correo electrónico
 - Agentes de usuario (UA)
 - Agentes de transferencia de mensajes (MTA)
 - Agentes de acceso a mensajes (MAA)



Correo electrónico

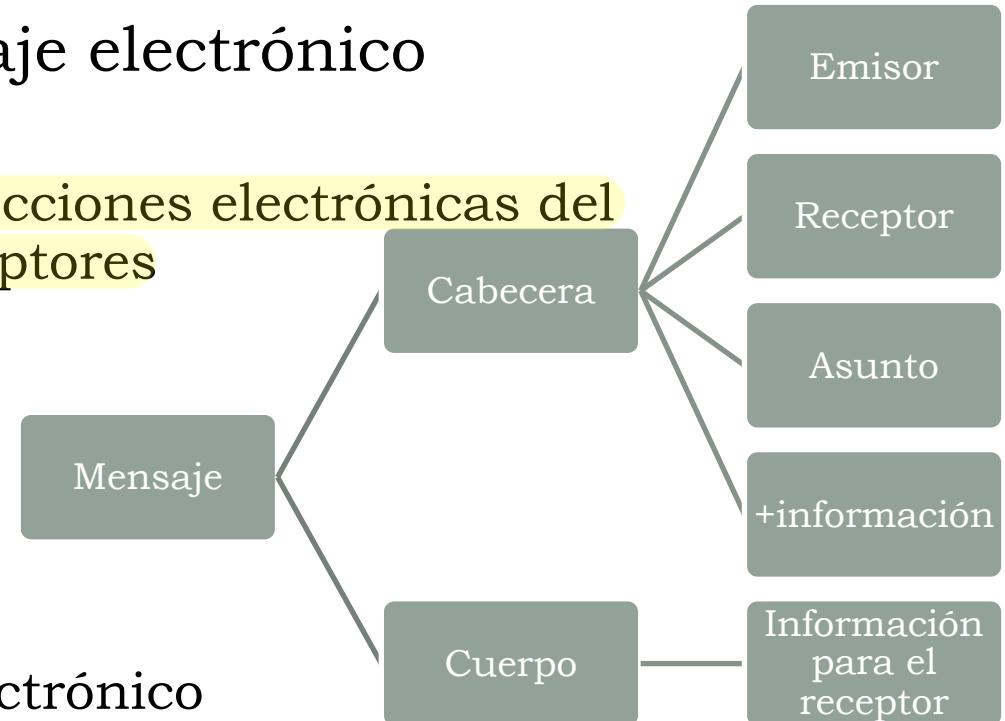
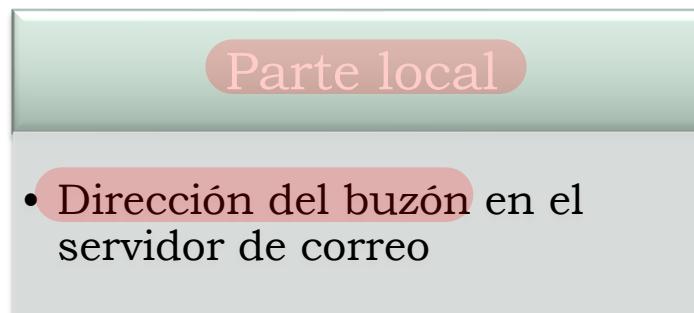
- Partes de un mensaje electrónico

- Sobre

- Contiene las direcciones electrónicas del emisor y los receptores

- Mensaje

- Dirección de correo electrónico



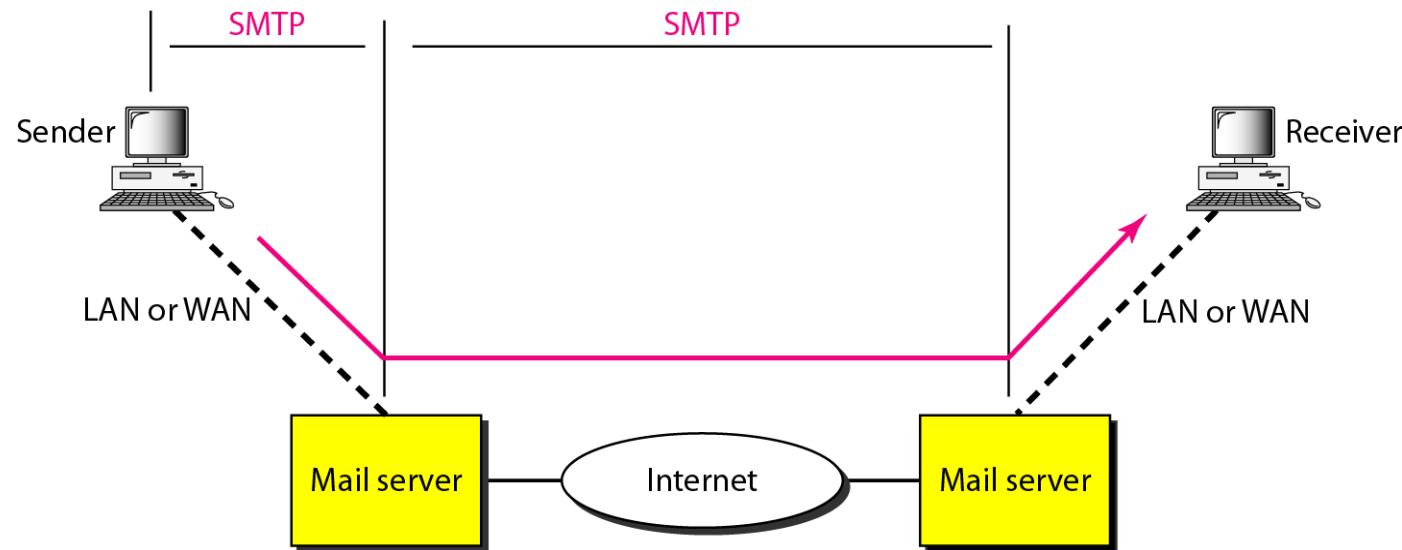
Correo electrónico

- **Agente de usuario** Programa para enviar correos.
 - Ofrece servicios para el envío o recepción de un mensaje
 - Es un programa que compone, lee, responde y envía mensajes
 - También incluye y gestiona buzones (uno de entrada y otro de salida)
 - Ejemplos → Eudora, Outlook, Netscape Messenger, Thunderbird, Mail



Correo electrónico

- Agente de transferencia de mensajes (SMTP)
 - Para enviar un correo electrónico, un sistema debe tener un cliente MTA y para recibir un correo electrónico debe tener un servidor MTA
 - El protocolo entre el cliente y el servidor MTA es SMTP (protocolo sencillo de transferencia de correo electrónico)
 - Se utilizan dos pares de cliente/servidor MTA en la mayoría de las situaciones
 - Puerto TCP/25



Mensajes y respuestas SMTP

Mensajes SMTP (implementación mínima)

HELO <SP> <domain> <CRLF>	Dominio
MAIL <SP> FROM:<sender> <CRLF>	De...
RCPT <SP> TO:<receiver> <CRLF>	Para...
DATA <CRLF>	
RSET <CRLF>	
NOOP <CRLF>	
QUIT <CRLF>	

Respuestas SMTP (Ídem códigos respuestas FTP)

Ejemplo:

250<SP>Requested mail action okay completed<CRLF>

Escenario de ejemplo SMTP (R=receptor, S=Sender)

S: 220 sol10.lcc.uma.es Simple Mail Transfer Service Ready
C: **HELO** sol10.lcc.uma.es
S: 250 sol10.lcc.uma.es
C: **MAIL FROM:** gabriel@lcc.uma.es
S: 250 OK
C: **RCPT TO:** rusman@lcc.uma.es
S: 250 OK
C: **RCPT TO:** mercedes@lcc.uma.es
S: 550 No such user here
C: **DATA**
C: Subject: Reunión de RySD
C: To: rusman@lcc.uma.es
C: Content-Type: text/plain; charset=ISO-8859-1;
C: Content-Transfer-Encoding: 8bit
C: Hola,
C: ¿Quedamos a las 19:00 en mi despacho?
C: Gabriel
C: .
S: 250 OK
C: **QUIT**
S: 221 sol10.lcc.uma.es Service closing transmission channel

The diagram illustrates the structure of an SMTP message. On the right side, three red brackets group the message into three main sections: 'Sobre' (envelope), 'Cabecera' (headers), and 'Mensaje' (body). The 'Sobre' bracket groups the initial server response and the HELO command. The 'Cabecera' bracket groups all commands related to the message headers, including MAIL FROM, RCPT TO, DATA, Subject, To, Content-Type, Content-Transfer-Encoding, and the final QUIT command. The 'Mensaje' bracket groups the actual message body content, which consists of the greeting 'Hola,' and the question '¿Quedamos a las 19:00 en mi despacho?'. The 'Cabecera' and 'Mensaje' sections are nested within the 'Sobre' section.

Extended SMTP (ESTMP)

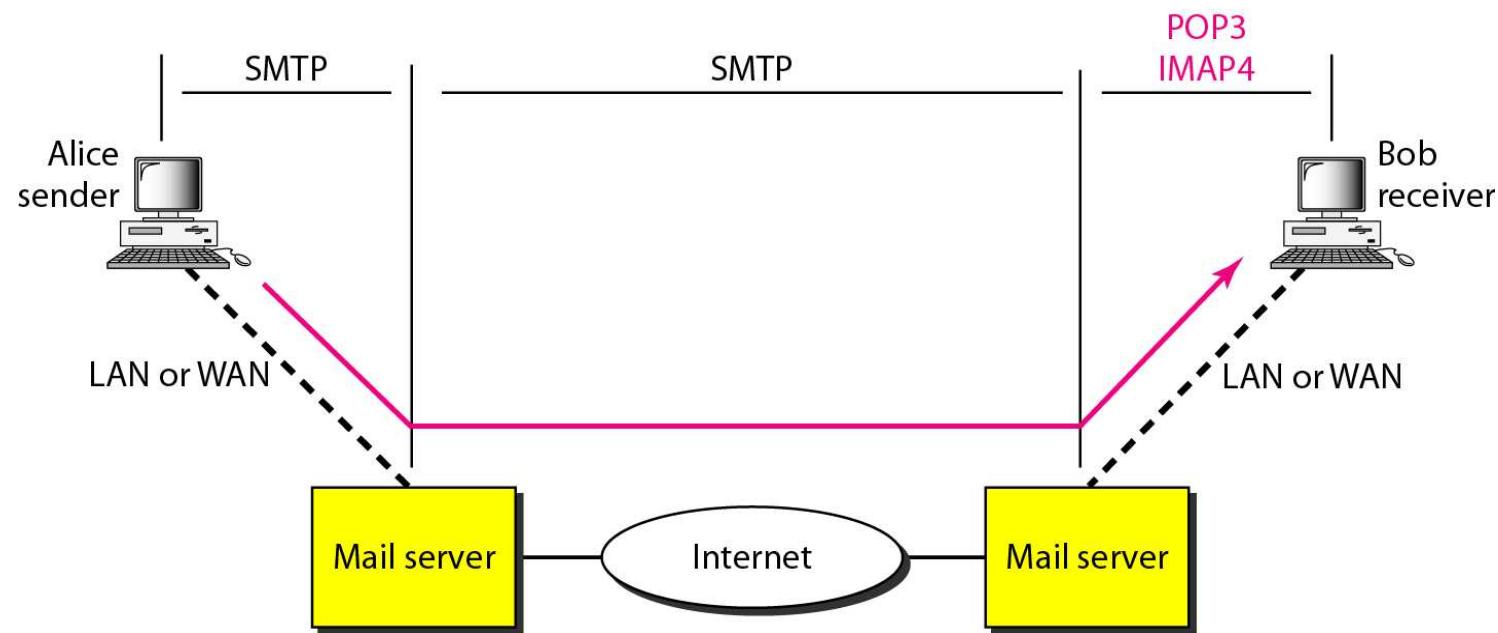
SMTP se extendió con algunos comandos:

- **EHLO** (vs HELO): El servidor responde informando de las extensiones soportadas
- **STARTTLS**: Cambia a formato encriptado usando TLS
- **SIZE**: Indica el tamaño máximo aceptado (servidor) o el tamaño estimado del correo (cliente)
- **AUTH**: Autenticación del usuario:
 - **AUTH PLAIN**: Tras enviar este comando el usuario debe enviar su nombre de usuario y clave juntos codificado en Base64
 - **AUTH LOGIN**: Tras enviar este comando el servidor pedirá el nombre de usuario y posteriormente la contraseña (todo lo enviado, cliente y servidor, en Base64)
 - **AUTH CRAM-MD5**: El servidor envía un reto para que el cliente envíe su contraseña cifrada.

Correo electrónico

- **Agente de Acceso a mensajes: POP e IMAP**

- Se encarga de extraer los mensajes del (buzón) servidor
- Los mensajes van del servidor al cliente



Correo electrónico

- **POP3** (protocolo de la oficina de correos)
 - Sencillo pero de funcionalidad limitada
 - El cliente se instala en la máquina del usuario
 - Permite la descarga de correos electrónicos almacenados en el buzón del usuario
 - Protocolo TCP/110
 - Dos modos de funcionamiento: borrado y mantenimiento
- **IMAP** (protocolo de acceso a correo de Internet)
 - Protocolo TCP/143
 - Más potente y complejo que POP
 - Para usuarios *nómadas*
 - Permite gestionar el buzón del servidor de correo
 - Consultar mensajes antes de su descarga
 - Buscar antes de descargar
 - Descarga parcial
 - Crear carpetas en el buzón, y crear, borrar o renombrar buzones en el servidor de correo
- **Correo basado en la Web (Webmail)** → **Gmail, Hotmail, Yahoo**
 - Permite el acceso desde un navegador al buzón de correo electrónico
 - Combina el uso de SMTP con HTTP

Mensajes/Respuestas POP

Mensajes POP-3 (implementación reducida)

USER<SP>name<CRLF>

PASS<SP>string<CRLF>

STAT<CRLF>

LIST [<SP>msg] <CRLF>

RETR<SP>msg<CRLF>

DELE<SP>msg<CRLF>

QUIT<CRLF>

Respuestas (ejemplos)

+OK<CRLF>

-ERR<CRLF>

Escenario POP (C=cliente, S=Servidor)

```
S: <wait for connection on TCP port 110>
C: <open connection>
S: +OK POP3 server ready <1896.697170952@dbc.mtvview.ca.us>
C: USER rdo
S: +OK rdo
C: PASS rdo123
S: +OK
C: STAT
S: +OK 2 320
C: LIST
S: +OK 2 messages (320 octets)
S: 1 120
S: 2 200
S: .
C: RETR 1
S: +OK 120 octets
S: <the POP3 server sends message 1>
S: .
C: DELE 1
S: +OK message 1 deleted
C: RETR 2
S: +OK 200 octets
S: <the POP3 server sends message 2>
S: .
C: DELE 2
S: +OK message 2 deleted
C: QUIT
S: +OK dewey POP3 server signing off (maildrop empty)
C: <close connection>
```

Tema 1. Introducción a las redes y sistemas distribuidos

Tema 2. Técnicas de acceso y control de enlace

Tema 3. Protocolos de Interconexión de Redes

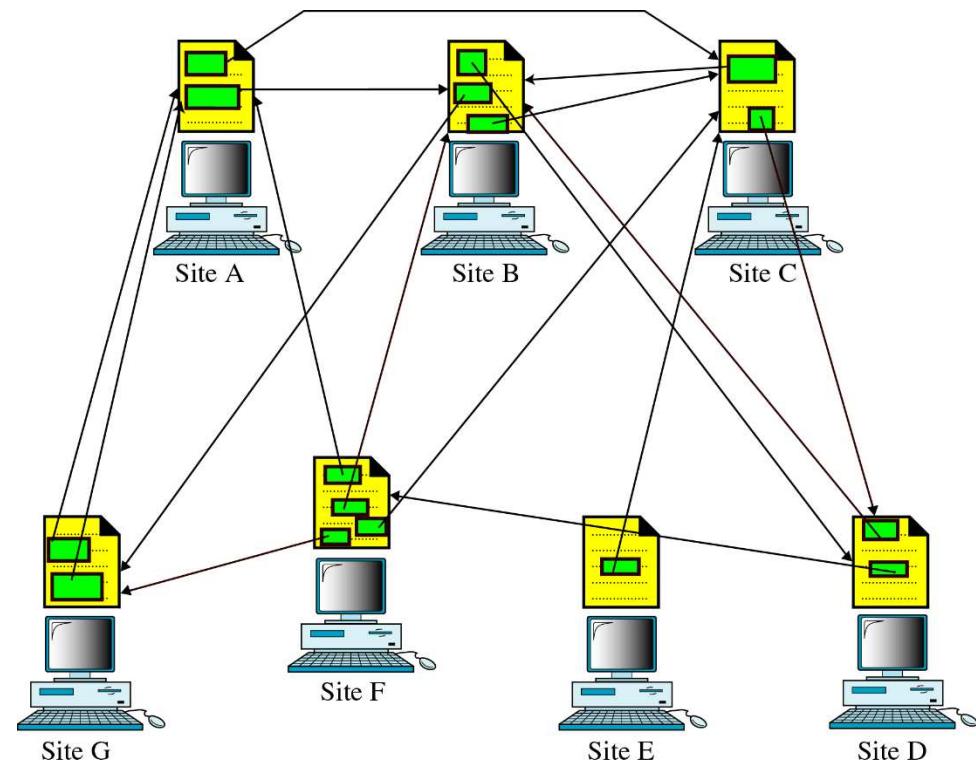
Tema 4. Servicios básicos para el nivel de transporte en Internet

Tema 5. Aplicaciones distribuidas en Internet

LA WEB (HTTP)

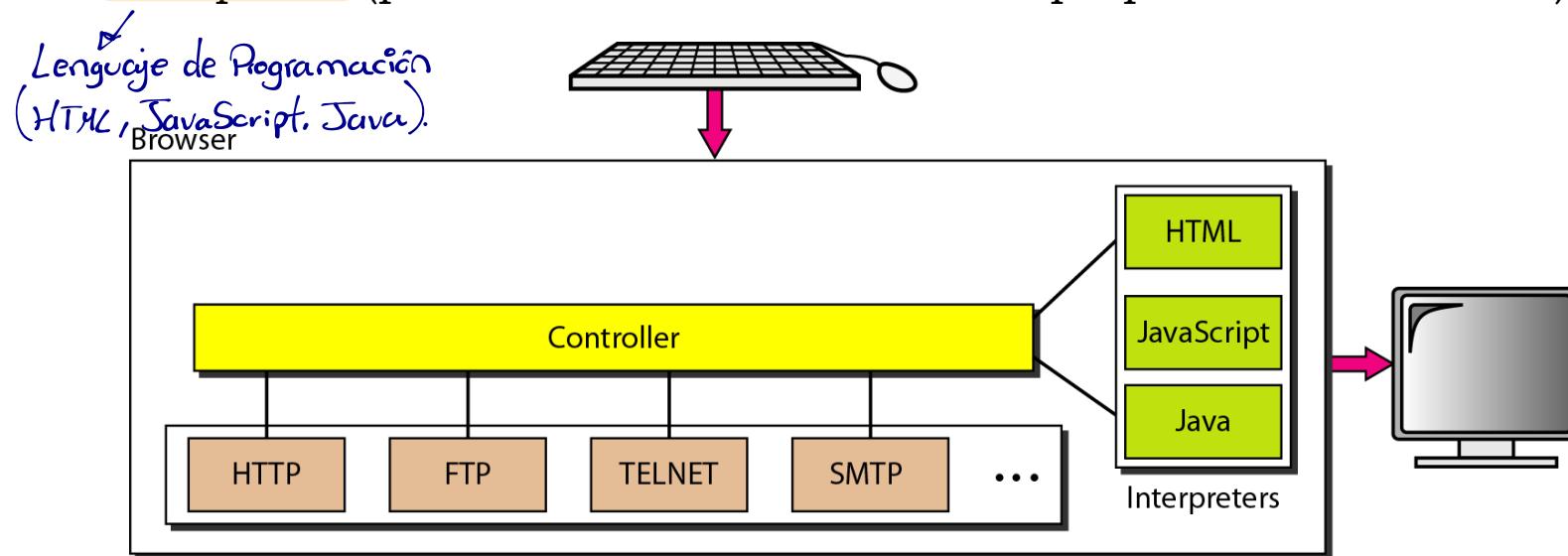
World Wide Web

- La **World Wide Web (WWW)**
 - Repositorio de información diseminada por todo el mundo y enlazada entre sí
 - Iniciado por el CERN (laboratorio europeo de física de partículas) para gestionar y acceder a los recursos distribuidos requeridos por los investigadores de este centro
- Arquitectura
 - Servicio cliente/servidor
 - Los datos servidos están distribuidos en muchas localizaciones llamadas **sitios**
 - Cada sitio almacena **páginas web** y otros recursos
 - Cada página web puede contener enlaces a páginas en el mismo sitio o de un sitio diferente



World Wide Web: el navegador

- El **navegador (browser)**
 - Obtiene, interpreta y visualiza documentos públicos (generalmente documentos webs y multimedia)
 - Explorer/Edge, Firefox, Chrome, Safari, Opera
 - Puede usar diferentes protocolos para obtener los documentos públicos
 - Su estructura incluye:
 - Un controlador (generalmente recibe los datos de teclado)
 - Un protocolo cliente (permiten obtener el documento)
 - Intérpretes (permiten visualizar de forma apropiada el documento)



World Wide Web: URL

- El localizador de Recursos Uniforme (URL) es un estándar que identifica y especifica cualquier tipo de información en Internet
 - HTTP lo utiliza para el acceso de los documentos distribuidos por la Web
- El URL define 4 elementos:
 - El protocolo Sigue ser HTTPS [S → Seguro]
 - Protocolo o aplicación cliente/servidor utilizado para recuperar el recurso
 - La estación (host)
 - Dirección IP o nombre de dominio de la máquina donde se está el recurso
 - El puerto
 - Información opcional. Normalmente se toma el puerto por defecto (80/443 en http/https) del protocolo
 - El camino (path)
 - Camino completo para localizar el recurso en la estación indicada (directorios ...)



World Wide Web: Páginas web

- Las páginas Web
 - Consiste en un fichero HTML que incluyen referencias a otros objetos:
 - Ficheros HTML
 - Elementos multimedia (imagenes, audio, video)
 - Código (ej, Javascript)
 - HTML (Hyper Text Markup Language)
 - Es un texto anotado con etiquetas (*término entre <...>*)

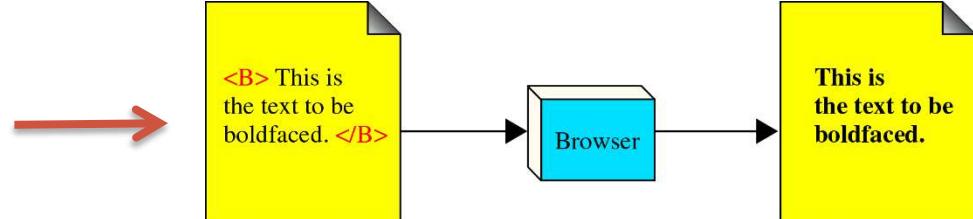
The screenshot shows a code editor window with the title 'page.html'. The code is as follows:

```
<!DOCTYPE html>
<head>
    <title>Aprendiendo a hacer p醙inas</title>
    <meta http-equiv="Content-Type" content="text/html; charset-iso-8859-1">
</head>
<body bgcolor="lightblue">
    <h1 align="center">Los inicios</h1>
    <p>Me inici醡 en la programaci髇 de p醙inas webs desde muy joven ...</p>
    <p align="center"></p>
    <p><a href="primera.html">Enlace a mi primera web</a></p>
</body>
</html>
```

Annotations in the original image highlight 'fichero HTML', 'referencias a otros objetos', 'Ficheros HTML', 'Elementos multimedia', 'Código (ej, Javascript)', 'HTML (Hyper Text Markup Language)', 'testo anotado', and 'etiquetas'.

Beginning of boldface
The text to be boldfaced
End of boldface

 This is the text that must be boldfaced.



World Wide Web: Páginas web: HTML

- Estructura de una página Web: visualización



El protocolo HTTP

- **HTTP (HyperText Transfer Protocol)**

- Protocolo de capa de aplicación para acceso de documentos Web

- Modelo Cliente/servidor:

- Puerto: 80 (TCP)

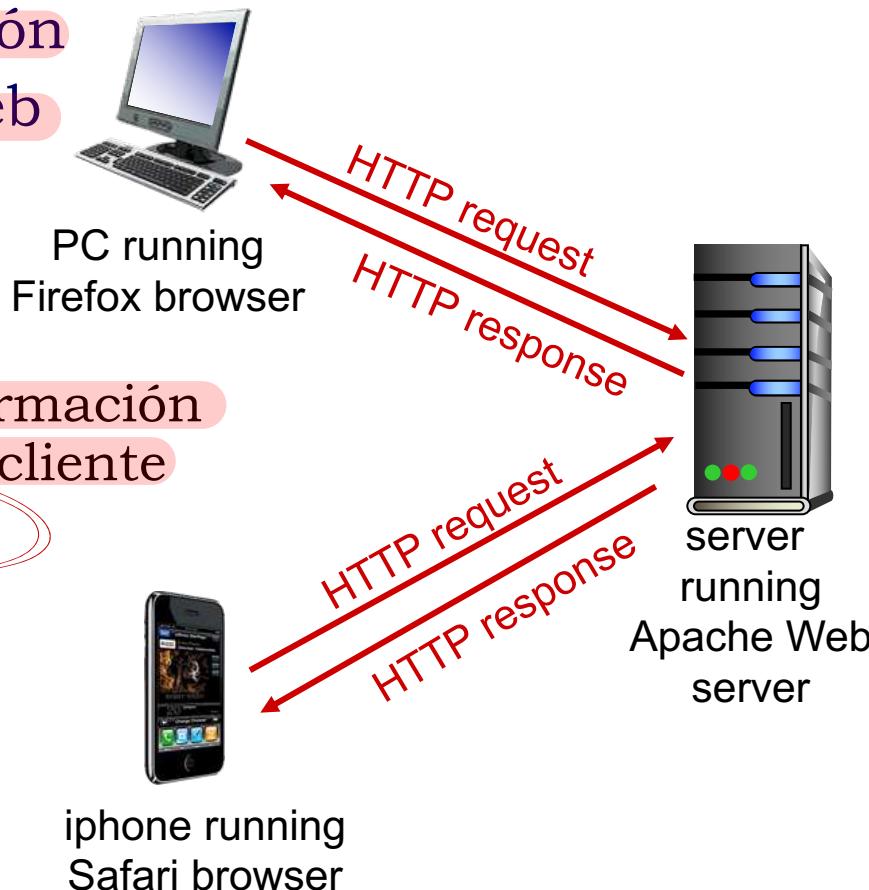
- Protocolo sin estado:

- El servidor no mantiene información de las peticiones previas del cliente

- **HTTP Secure (HTTPS)**

- HTTP sobre TLS

- Puerto: 443



HTTP/0.9
HTTP/1.0
HTTP/1.1

HTTP/2

1991

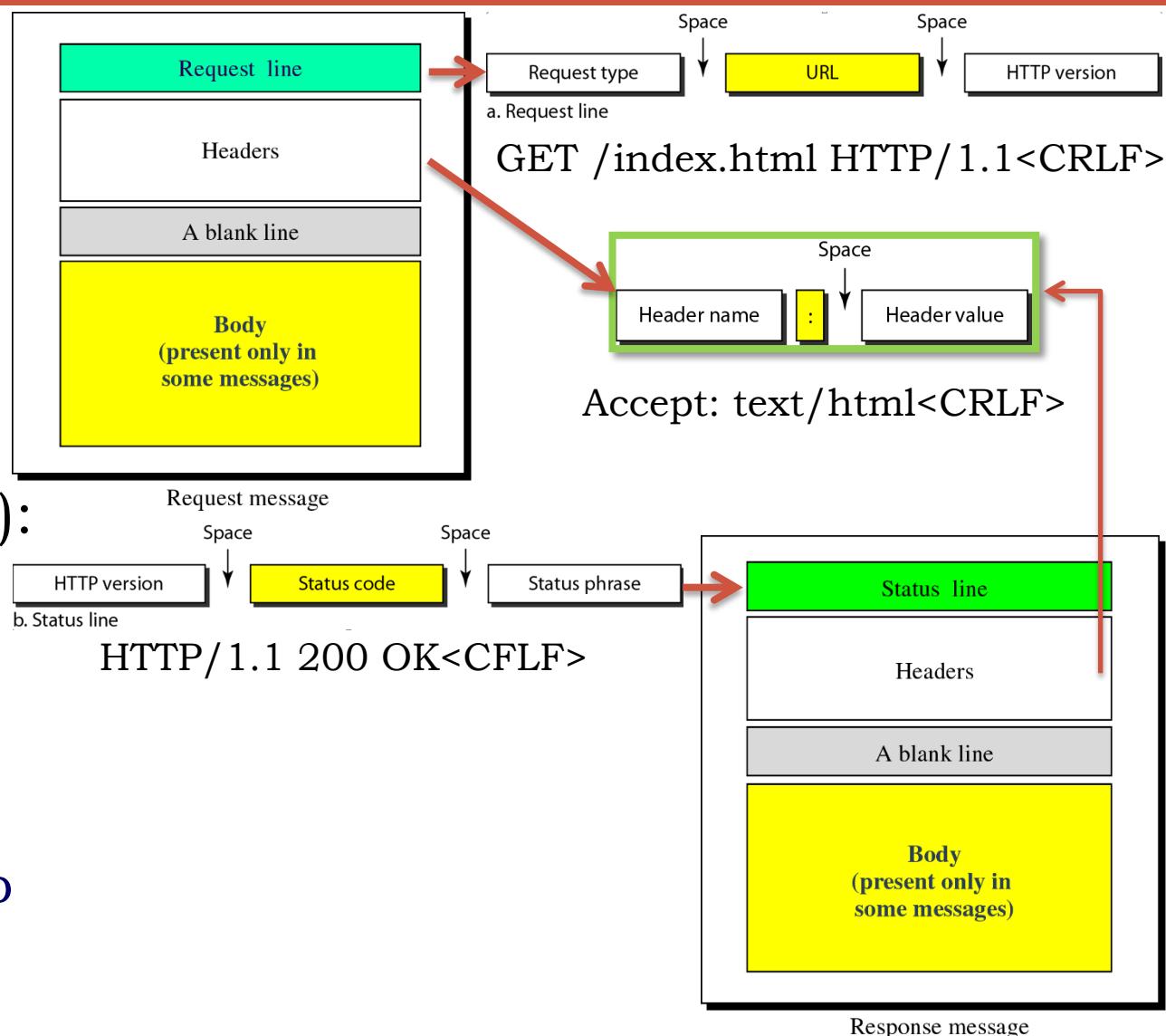
1996

1999

2015

HTTP: Mensajes

- Petición HTTP
 - Del cliente al servidor
- Respuesta HTTP
 - Del servidor al cliente
- Formato (textual):
 - Línea inicial:
 - Petición
 - Respuesta
 - Cabeceras
 - Línea en blanco
 - Datos



Petición: Métodos HTTP

Método	Acción
GET	Solicitud de un documento a un servidor
HEAD	Solicitud de información sobre un documento (no el documento en sí)
POST	Envío de información al servidor para que sean procesados por el recurso.
PUT	Envío de un documento del servidor al cliente
TRACE	Eco de petición entrante
CONNECT	Se utiliza para saber si se tiene acceso a un host
OPTION	Solicitud de algunas opciones disponibles
DELETE	Elimina un recurso del servidor

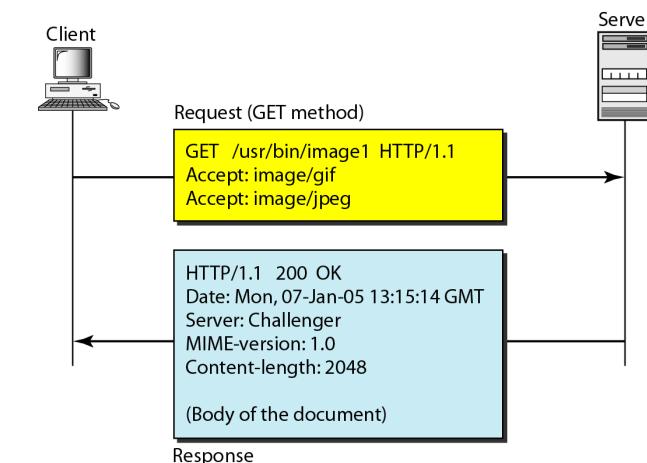
HTTP/0.9

- GET

HTTP/1.0

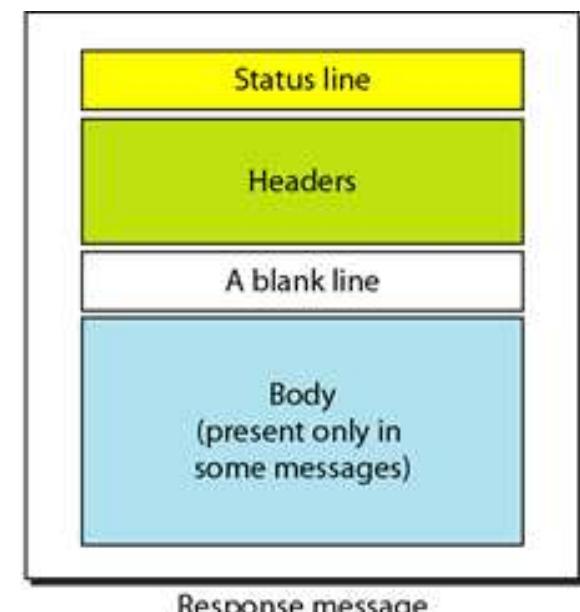
- GET
- POST
- HEAD
- GET, POST, HEAD
- PUT
- DELETE
- ...

HTTP/1.1



Respuestas: Tipos

- Mensajes de respuesta (estados y frases) + contenido solicitado según el caso:
 - 1xx Confirmación preliminar
 - 2xx Confirmación
 - 3xx Se necesitan más acciones por parte del cliente
 - 4xx Error en la petición
 - 5xx Error en el servidor
- Ejemplos
 - 101 Switching
 - 200 OK
 - 301 Moved Permanently
 - 400 Bad Request
 - 404 Not Found
 - 505 HTTP Version Not Supported



HTTP: Cabeceras

- **Formato requerido para la respuesta:**
 - Accept, Accept-Charset, Accept-Encoding, Accept-Language, ...
- **Formato de los datos enviados:**
 - Content-Encoding, Content-Language, Content-Length, Content-Type, ...
- **Eficiencia:**
 - Cache-control, Expires, If-modified-since, ...
- **Control del protocolo:**
 - Authorization, Upgrade, Strict-Transport-Security, Forwarded, Connection, ...
- **Información adicional:**
 - Date, User-Agent, Server, Host, X-...
- **“Sesiones”:**
 - Set-Cookie, Cookie

HTTP: Documentos web

- Tres tipos de documentos:**

- Estáticos**

- Recurso fijo del servidor

- Dinámicos**

- El documento se crea en el servidor cuando llega una petición
 - CGI, Servlets, lenguajes scripts (PHP, JS, ...)

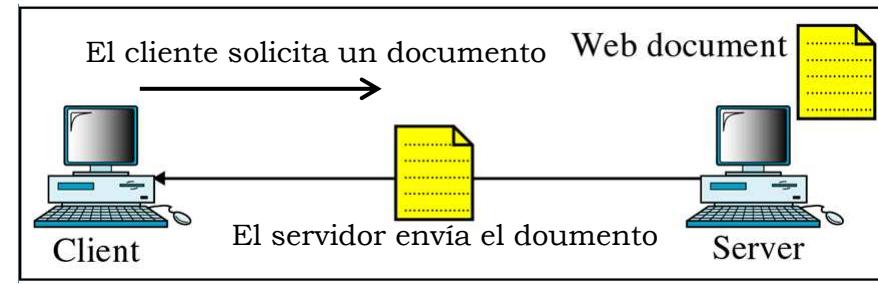
- Activos**

- Programas que se ejecutan en el lado del cliente
 - El servidor envía un documento activo que se ejecuta en el lado de cliente
 - JavaScript, Silverlight, Flash, Applets de Java

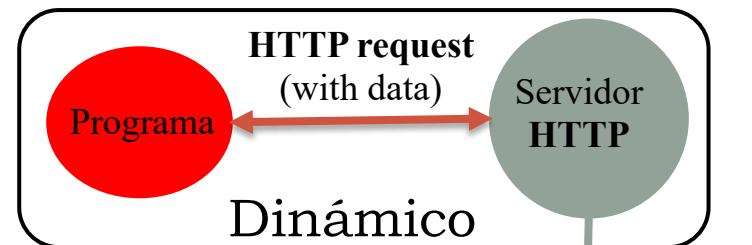
- Documentos web (HTML):**

- Contienen múltiples objetos**

- Código Javascript, CSS, imágenes, vídeos, etc.



Estático



Dinámico

1º Necesito unos datos
2º Petición HTTP de los datos al server
3º Respuesta HTTP del server con los datos que necesito

Este cuadro representa un formulario HTML con los siguientes elementos:

- Un cuadro de texto para "Name:".
- Un cuadro de texto para "Age:".
- Un botón rectangular para "Submit".

Formulario

Petición HTTP (datos)

Respuesta HTTP (p.e., página web)

Cliente HTTP

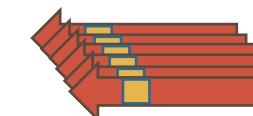
HTTP/1.1: Problemas

- HTTP/1.1

- Persistencia:

- La misma conexión para múltiples recursos
 - Problema: dependencia entre recursos (*head of the line blocking*), esperas (no *pipelining*)

Cliente

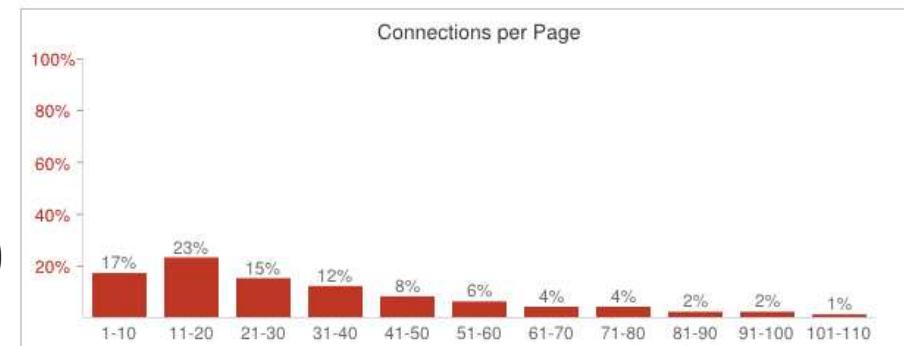


Servidor

- Optimizaciones:

- Descargas en paralelo:

- Múltiples conexiones
(Ej, 6 por dominio - Chrome)



- “Trucos” de desarrollo:

- Concatenar ficheros, incrustar ficheros (Ej, meter imágenes en un fichero CSS), etc.
 - Uso de varios dominios por servidor web (media: 18)

Cliente



Servidor

- Problema: complejidad, peticiones DNS, problemas con el uso de caché, modularidad, ...

HTTP/1.1: Problemas

- ¿Es necesario una nueva versión de HTTP?
 - Evitar trucos que incrementen la eficiencia
 - Solucionar otros problemas que tiene
- Objetivos de HTTP/2
 - Aumentar la eficiencia
 - No requerir múltiples conexiones y evitar otros problemas
 - Compatibilidad HTTP/1.1 (no romper la web ya existente)
- Aprobado el 18 de febrero de 2015:
 - HTTP/2 (RFC 7540) – 76 páginas
 - HPACK: Compresión de la cabecera (RFC 7541) – 55 págs.
- Dos implementaciones:
 - HTTP/2 sobre TLS (h2)
 - HTTP/2 sobre TCP plano (h2c)

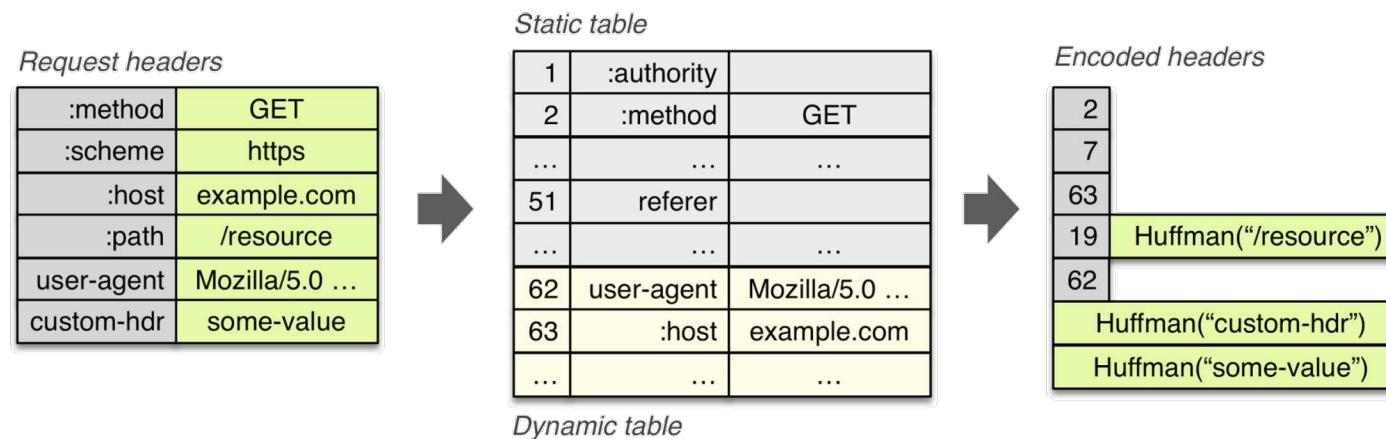
La mayoría de los
navegadores/servidores
no darán soporte

HTTP/2

- La idea es mantener la semántica de HTTP/1.1:
 - Mismos métodos (GET, PUT, ...), cabeceras, casos de uso
 - Mantiene el modelo cliente/servidor
- Novedades:
 - Binario (envío de “tramas”).
 - Compresión de las cabeceras.
 - Una sola conexión:
 - Múltiples de flujos (multiplexación de flujos):
 - Múltiples recursos por conexión.
 - Se pueden limitar (al menos se recomienda el uso de 100).
 - Cierre de flujos sin cerrar conexión (RST_STREAM).
 - Prioridades/dependencias en flujos y cambios dinámicos:
 - Recursos “fuera de orden”
 - *Server Push*: envío de información sin ser solicitada por parte del servidor

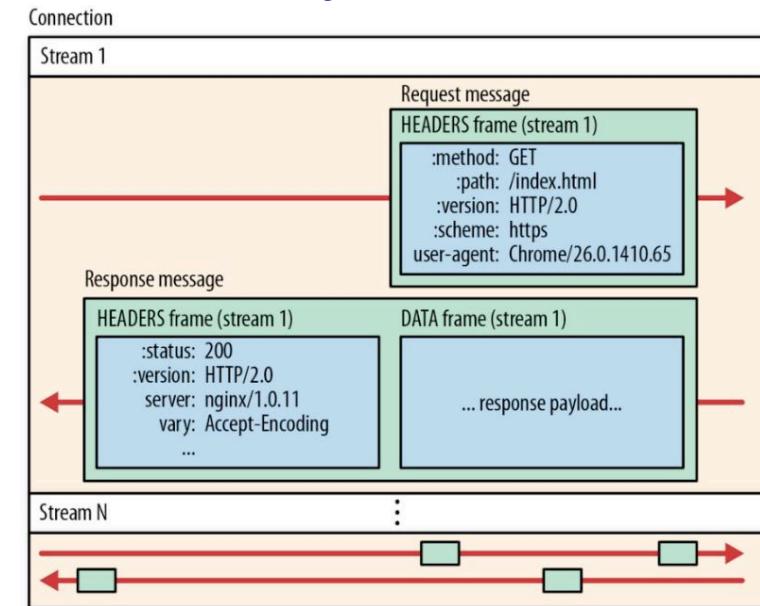
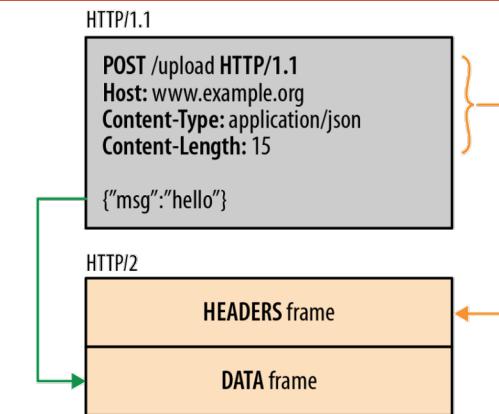
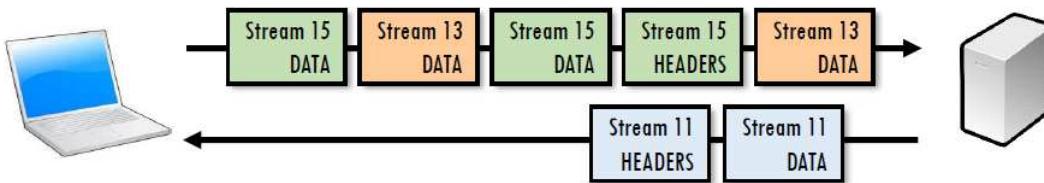
HTTP/2: Compresión de cabeceras (HPACK)

- Uso de tablas de cabeceras
 - Se crea tabla con cabeceras habituales (estática)
 - Se crea tabla con cabeceras usadas previamente (dinámica)
- Se envían los índices de esas tablas
- Los valores/cabeceras nuevas se codifican con Huffman:
 - Caracteres más usados usan menos bits:
 - ‘e’ -> 00101
 - ‘\’ -> 111111111111110000



HTTP/2: Comunicaciones

- Tramas: unidad mínima de transferencia
- Flujo: una petición/respuesta
 - Compuesto por una o varias tramas
- Una única conexión TCP:
 - Está formada por múltiples flujos
 - Prioridad (dinámica) y dependencia entre flujos
 - Se pueden mezclar tramas



HTTP/2: Trama

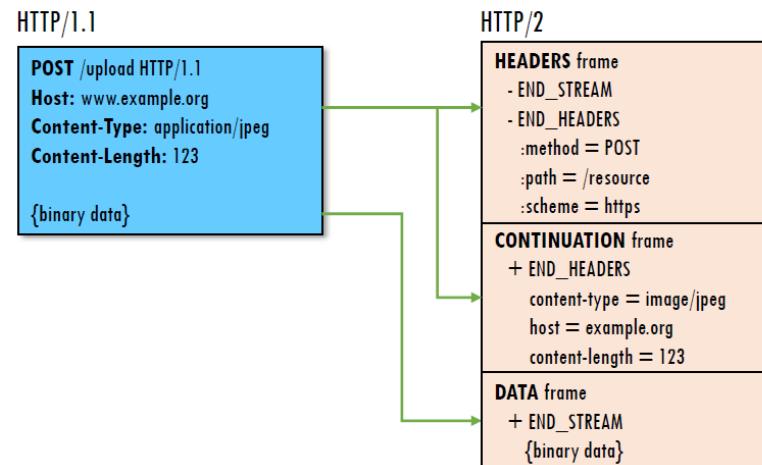
- 9 Bytes + parte variable según el tipo:

Bit	+0..7	+8..15	+16..23	+24..31
0		Length	Type	Flags
32	R		Stream Identifier	
...			Frame Payload	

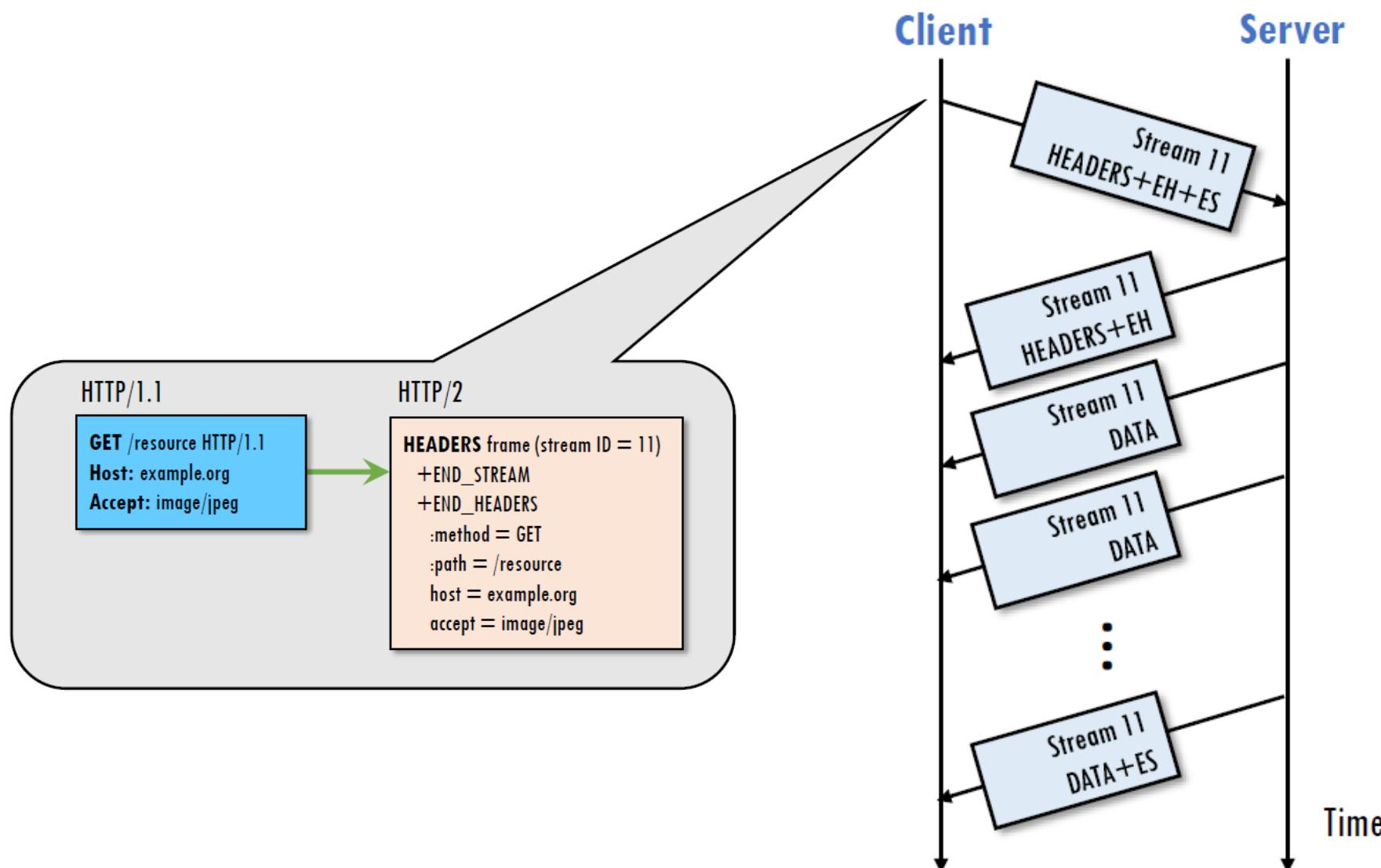
- **Length:** Longitud de la trama (sin incluir los 9 bytes iniciales)
- **Type:** Tipo de trama
- **Flags:** definidos atendiendo al tipo de trama
- **R:** Reservado (0)
- **Stream Identifier:** Identificador de flujo (petición/respuesta):
 - 0: reservado para control de la conexión en general
 - 1: para la conexión HTTP/1.1 inicial (si la hay)
 - Iniciados por el cliente: impares
 - Iniciados por el servidor: pares
- **Frame Payload:** Datos (depende del tipo de trama)

HTTP/2: Tipos de trama

Frame type	Description
DATA	Cuerpo HTTP
HEADERS	Cabeceras
PRIORITY	Prioridad de un flujo
RST_STREAM	Finalización de un flujo
SETTINGS	Parámetros de configuración de la conexión (solo flujo 0)
PUSH_PROMISE	Envío de datos (sin haber sido pedidos)
PING	Para medir el rtt y comprobar que el otro extreme está activo
GOAWAY	Informa al otro extreme que no cree más flujos en esta conexión
WINDOW_UPDATE	Control de flujo (flujo concreto o conexión global – flujo 0)
CONTINUATION	Continúa enviado cabeceras

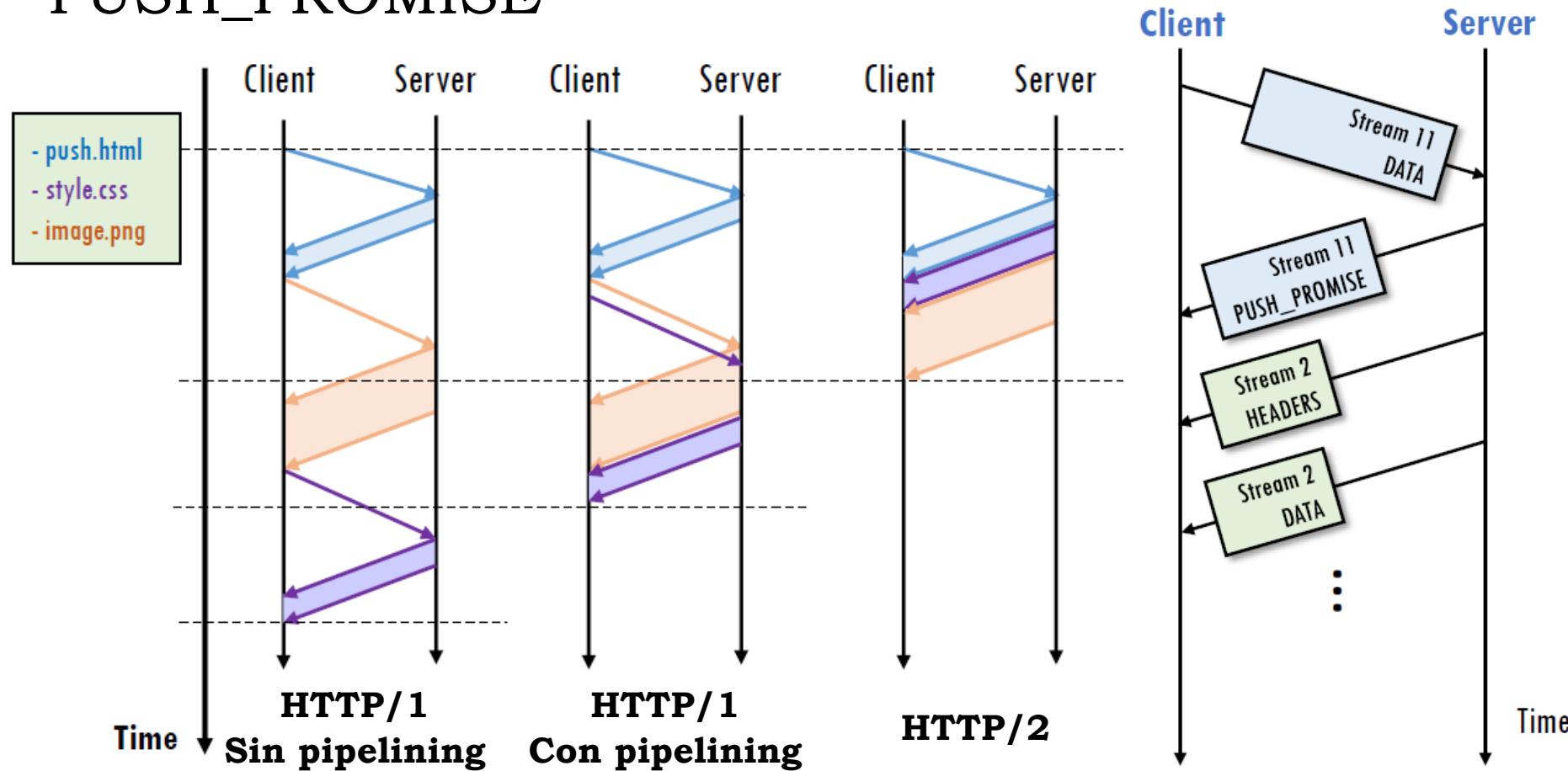


HTTP/2: Trama: Ejemplo de comunicación



HTTP/2: Server Push

- El servidor puede enviar datos al cliente que sabe que va a necesitar sin pedirlos mediante la trama PUSH_PROMISE



HTTP/2: Interacción con HTTP/1.1

- Implementación h2 (sobre TLS):
 - Durante el proceso de autenticación TLS usa el APLN (*Application Layer Protocol Negotiation*) para fijar http/2
- Implementación h2c (sobre TCP plano):
 - Usa la cabecera Upgrade

```
GET /page HTTP/1.1
Host: server.example.com
Connection: Upgrade, HTTP2-Settings
Upgrade: h2c
HTTP2-Settings: (Datos trama SETTINGS)
```



```
HTTP/1.1 200 OK
Content-length: 243
Content-type: text/html
(... Respuesta HTTP/1.1 ...)
```

--- o ---

```
HTTP/1.1 101 Switching Protocols
Connection: Upgrade
Upgrade: h2c
(... Respuesta HTTP/2 ...)
```

- Si ya se sabe que usa HTTP/2, el cliente puede empezar con:
0x505249202a20485454502f322e300d0a0d0a534d0d0a0d0a...
PRI * HTTP/2.0\r\n\r\nSM\r\n\r\n\r\n + trama SETTINGS

Tema 1. Introducción a las redes y sistemas distribuidos

Tema 2. Técnicas de acceso y control de enlace

Tema 3. Protocolos de Interconexión de Redes

Tema 4. Servicios básicos para el nivel de transporte en Internet

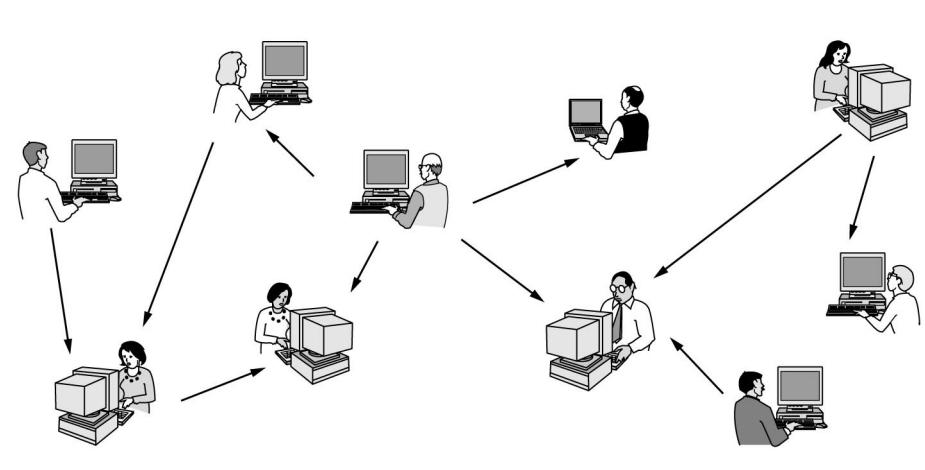
Tema 5. Aplicaciones distribuidas en Internet

- Servicios P2P
- Servicios Multimedia

SERVICIOS AVANZADOS DE INTERNET

Modelo P2P

- Esquema



- Ejemplos

- BitTorrent: intercambio de ficheros
- Redes *ad hoc*
- Skype
- Spotify
- Bitcoin

Compartición de contenidos en redes peer-to-peer

- Las redes P2P (peer-to-peer)
 - son redes que no tienen clientes ni servidores fijos, sino un conjunto de nodos que se comportan simultáneamente como clientes y servidores respecto a los demás nodos de la red
- Permiten compartir y transferir archivos (legales) de forma eficiente
 - Aprovechan la conectividad (ad hoc) entre nodos de la red, mejorando el rendimiento en la conexiones y transferencias frente a redes centralizadas
 - Tiene más utilidades, pero la más demandada en la compartición de archivos
 - También es utilizada para la telefonía VoIP y streaming en general
- Cualquier nodo puede iniciar, detener o completar una transacción
 - Los archivos se almacenan de forma distribuida entre los nodos de la red
 - Filosofía P2P:
 - El que más comparta más privilegios tiene

Compartición de Archivos P2P

Ejemplo

- Alicia ejecuta su cliente P2P en su portátil
- Tiene una conexión intermitente a Internet; en cada conexión obtiene una dirección IP diferente
- Sigue “Hey Jude”
- La aplicación muestra otros nodos “peers” que disponen de una copia de “Hey Jude”.

- Alicia elige uno de los peers, Bob.
- El fichero se transfiere y copia desde el PC de Bob al portátil de Alicia (HTTP)
- Mientras Alicia descarga, otros usuarios se descargan desde el portátil de Alicia
- El nodo de Alicia es tanto un cliente, como un servidor transitorio

Todos los nodos son servidores = muy escalable!!!

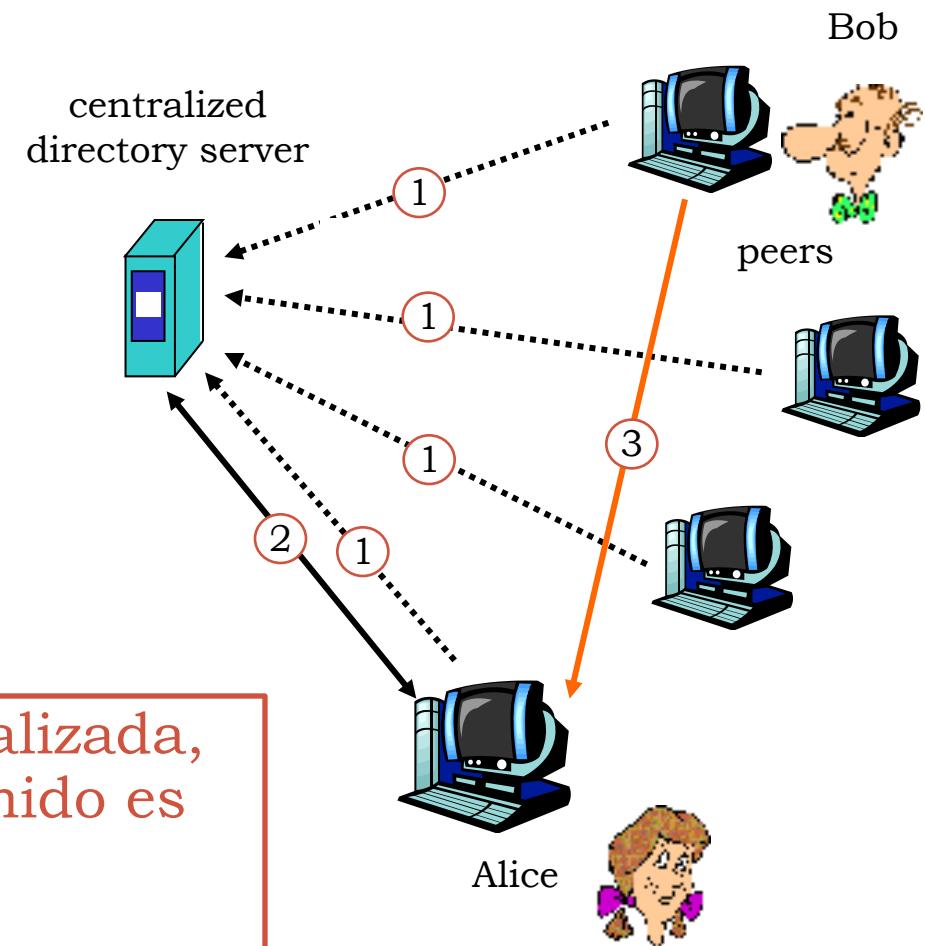
P2P

- Una vez que se ha elegido el nodo, la descarga es directa.
- Dada la conectividad intermitente de los nodos, la dificultad está en localizar y buscar contenidos (en los nodos).
- Tres acercamientos:
 - Centralizada
 - Distribuida
 - Híbrida

P2P: localización de contenidos centralizada

Hay un servidor dedicado a funciones de directorio

- 1) Cuando un nodo se conecta, informa al servidor central
 - Dirección IP
 - Contenido
- 2) Alice pregunta por “Hey Jude”
- 3) Alice le pide el fichero a Bob



La transferencia es descentralizada, pero la localización de contenido es centralizada

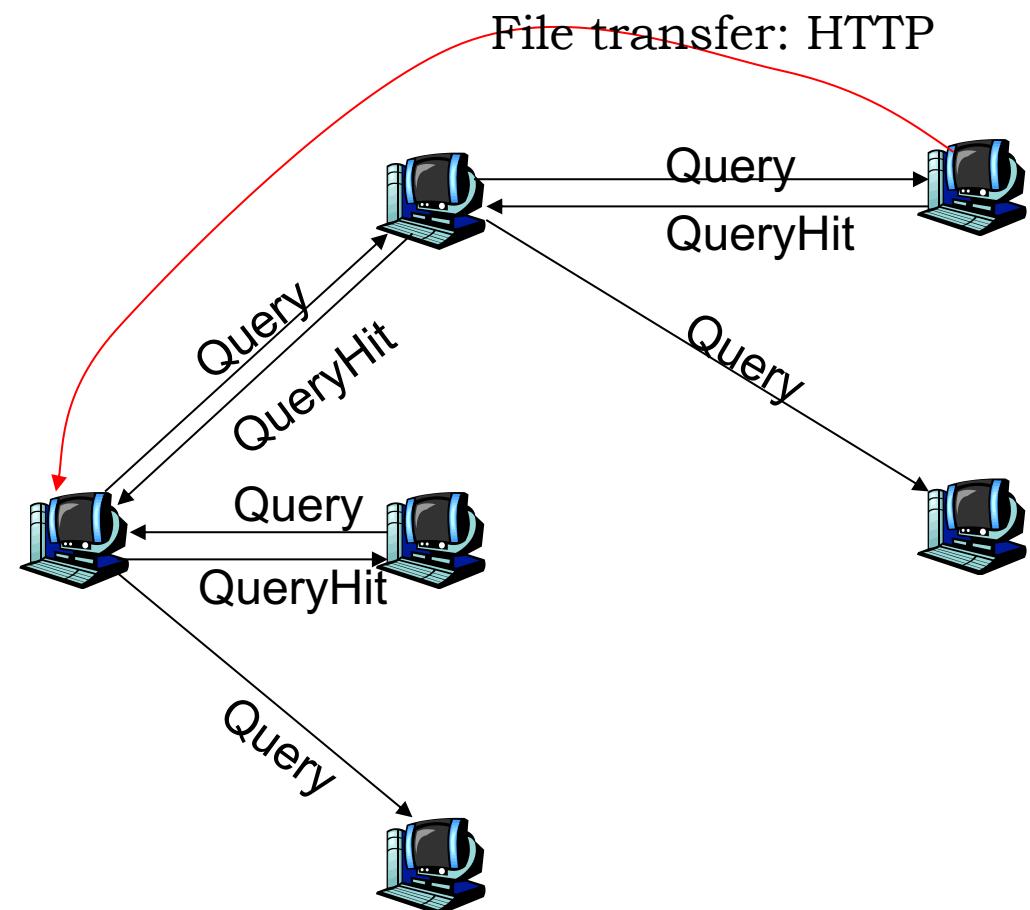
- Cuello de botella
- Único punto de fallo

P2P: localización de contenidos distribuida

- Totalmente distribuida
 - No hay servidor central
- Los nodos forman un grafo → overlay network
 - Hay un arco entre los nodos X e Y si hay entre ellos una conexión TCP
 - La red *overlay* está formada por todos los nodos activos y los arcos entre ellos
 - Un arco no es un enlace físico
 - Típicamente un nodo estará conectado con <10 vecinos *overlay*
 - La localización de contenidos se realiza mediante inundación de consultas en la red *overlay*
 - protocolo “query” - “query hit” entre los nodos.

P2P: localización de contenidos distribuida

1. El mensaje es enviado a todos los nodos vecinos a través de la conexión TCP
2. Los nodos reenvían el mensaje de localización
3. Cuando el recurso se localiza en un nodo, se envía un mensaje QueryHit a través del mismo camino en dirección contraria



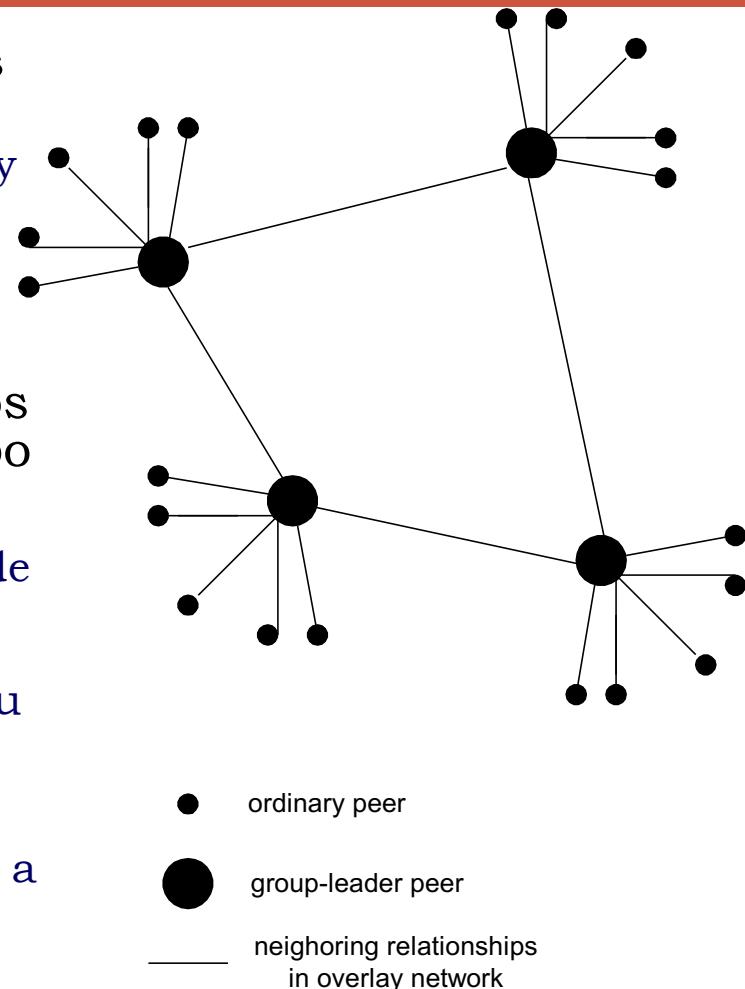
Escalabilidad:
Inundación de ámbito limitado

P2P: localización de contenidos distribuida

- El problema está en unirse a la red
 - Para unirse a la red, el nodo X debe encontrar otro nodo perteneciente a la red
 1. Usar una lista de candidatos
 2. Intentar de forma secuencial la conexión con al menos uno (Y)
 3. X envía un mensaje de Ping Y; Y reenvía el mensaje de Ping a sus vecinos.
 4. Todos los nodos que reciben el mensaje de Ping responden con un mensaje de Pong
 5. X puede establecer conexiones TCP adicionales

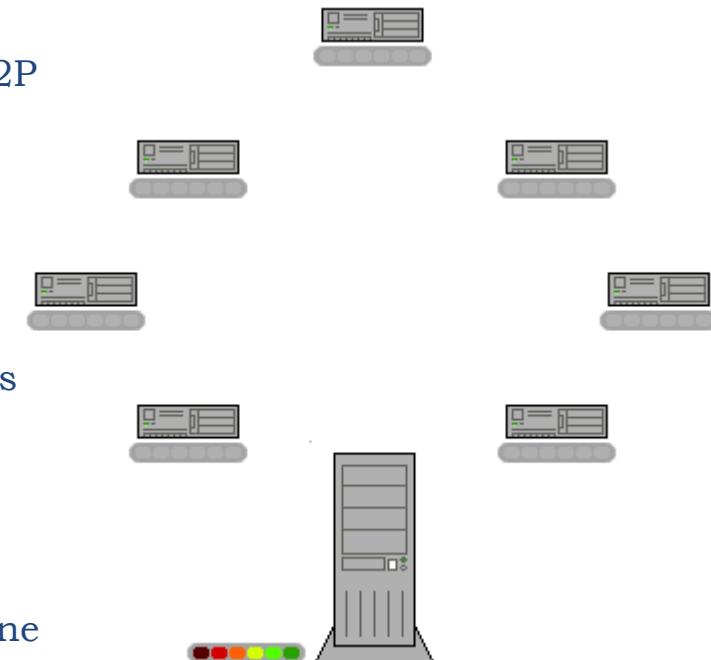
P2P: localización de contenidos híbrida

- Cada nodo es un líder de grupo o es asignado a un líder de grupo
 - Hay una conexión TCP entre un nodo y su líder de grupo
 - Hay conexiones TCP entre pares de líderes de grupo
- Un líder de grupo mantiene información acerca de los contenidos que tienen los miembros de su grupo
- Búsqueda:
 - El cliente envía la consulta a su líder de grupo
 - El líder de grupo responde con los contenidos que se corresponden con su consulta
 - Para cada correspondencia proporciona:
 - metadatos, dirección IP
 - Si el líder de grupo reenvía la consulta a otros líderes de grupo, responden con correspondencias
 - El cliente selecciona alguno de los ficheros para descarga mediante una petición HTTP a la dirección IP

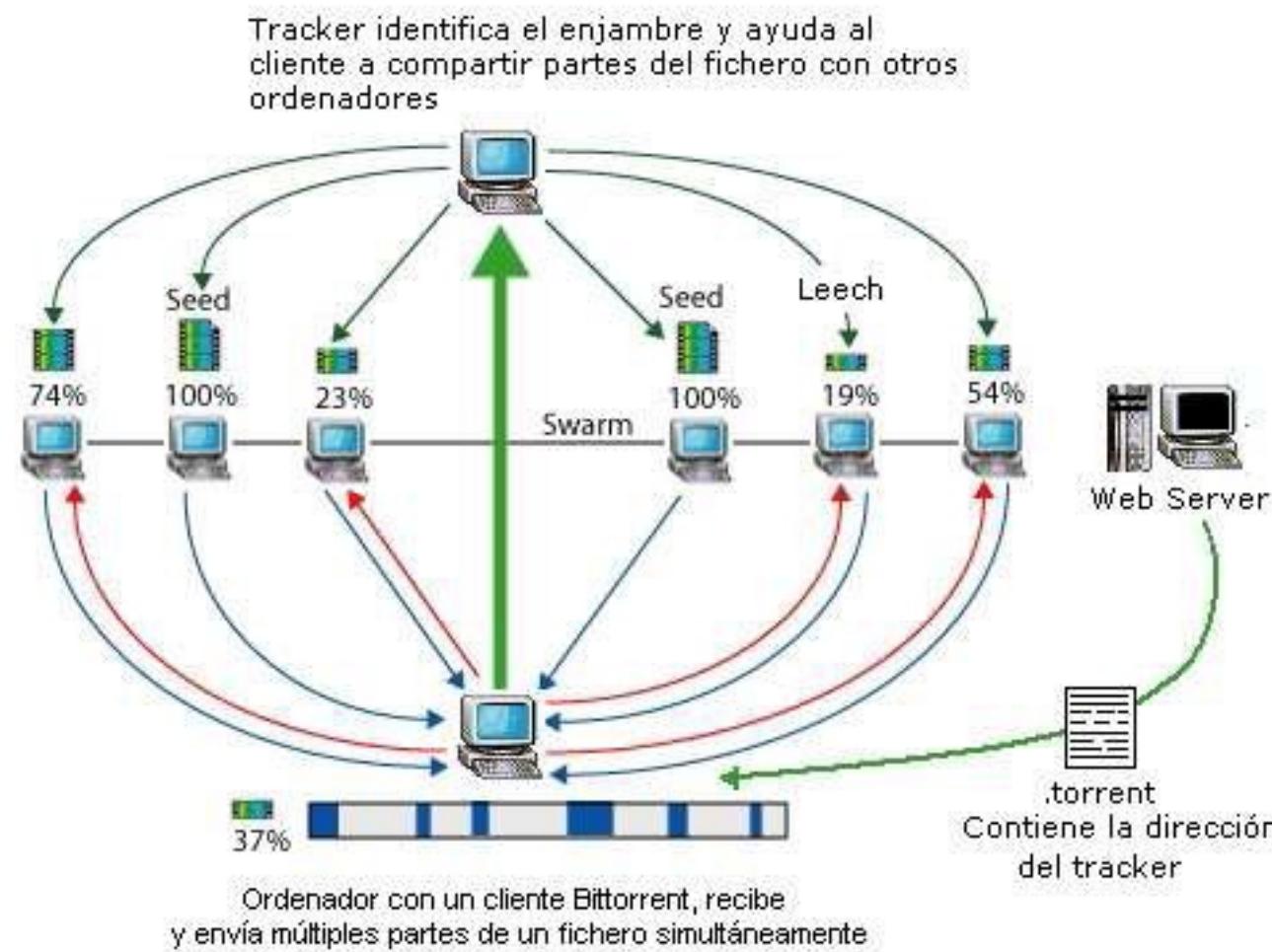


Compartición de contenidos en redes P2P

- Localización de nodos
 - A través de un servidor que mantiene una lista de nodos conectados a la red
- BITTORRENT
 - Protocolo para el intercambio de archivos en redes P2P
 - Los fichero se dividen en “trozos” - chunks
 - Un servidor localiza las posibles fuentes del fichero o partes de él
 - Un pequeño fichero .torrent contiene la dirección del servidor de búsqueda
 - El fichero o colección de ficheros es descargado de las fuentes encontradas por el servidor.
 - Cuando se inicia la descarga se comienza a subir las partes disponibles a otras fuentes.
 - La descarga se inicia por partes al azar
 - Si entre todos los usuarios conectados se dispone del fichero completo todos obtendrán una copia de él
 - Inicialmente alguien debe tener la copia completa



BITTORRENT



Tema 1. Introducción a las redes y sistemas distribuidos

Tema 2. Técnicas de acceso y control de enlace

Tema 3. Protocolos de Interconexión de Redes

Tema 4. Servicios básicos para el nivel de transporte en Internet

Tema 5. Aplicaciones distribuidas en Internet

MULTIMEDIA

Introducción a Multimedia

- Definición:
 - [WIKIPEDIA] **Multimedia** es un sistema que utiliza más de un medio de comunicación al mismo tiempo en la presentación de la información, como el texto, la imagen, la animación, el vídeo y el sonido.
 - “El término **Multimedia** se utiliza para indicar que la información y los datos de una aplicación integran o pueden estar compuestos de diferentes tipos de medios (de comunicación)”
- Los tipos de medios de comunicación son:
 - Texto
 - Imágenes (Gráficos, fotografías)
 - **Audio** (Voz , Música)
 - **Vídeo**

Introducción a Multimedia

- En la práctica, las aplicaciones multimedia son **aplicaciones distribuidas**, y los datos son transmitidos por diferentes tipos de redes y presentados en diferentes soportes.
- Los dispositivos multimedia deben soportar la presentación de los datos multimedia – **REPRESENTACIÓN** de los datos
- Las redes de comunicación deben proporcionar la infraestructura necesaria para soportar los requisitos de los servicios de comunicación multimedia – **COMPRESIÓN** y **TRANSMISIÓN**

Introducción a Multimedia

- Clasificación de las aplicaciones multimedia:
 - Streaming de audio y vídeo almacenado
 - Youtube o Netflix son claros ejemplos
 - Los canales de TV y radio ofrecen sus contenidos propios de esta manera después de haberlo emitido en TV.
 - Streaming de audio y vídeo en directo
 - Casi todos las cadenas de TV y radio ofrecen su emisión en directo a través de Internet
 - Muchos eventos (especialmente los deportivos/streaming de videojuegos) se emiten en directo por Internet.
 - Audio y vídeo interactivo en tiempo real
 - Conferencias y videoconferencias a través de Internet
 - Skype, Meet, Teams, etc.

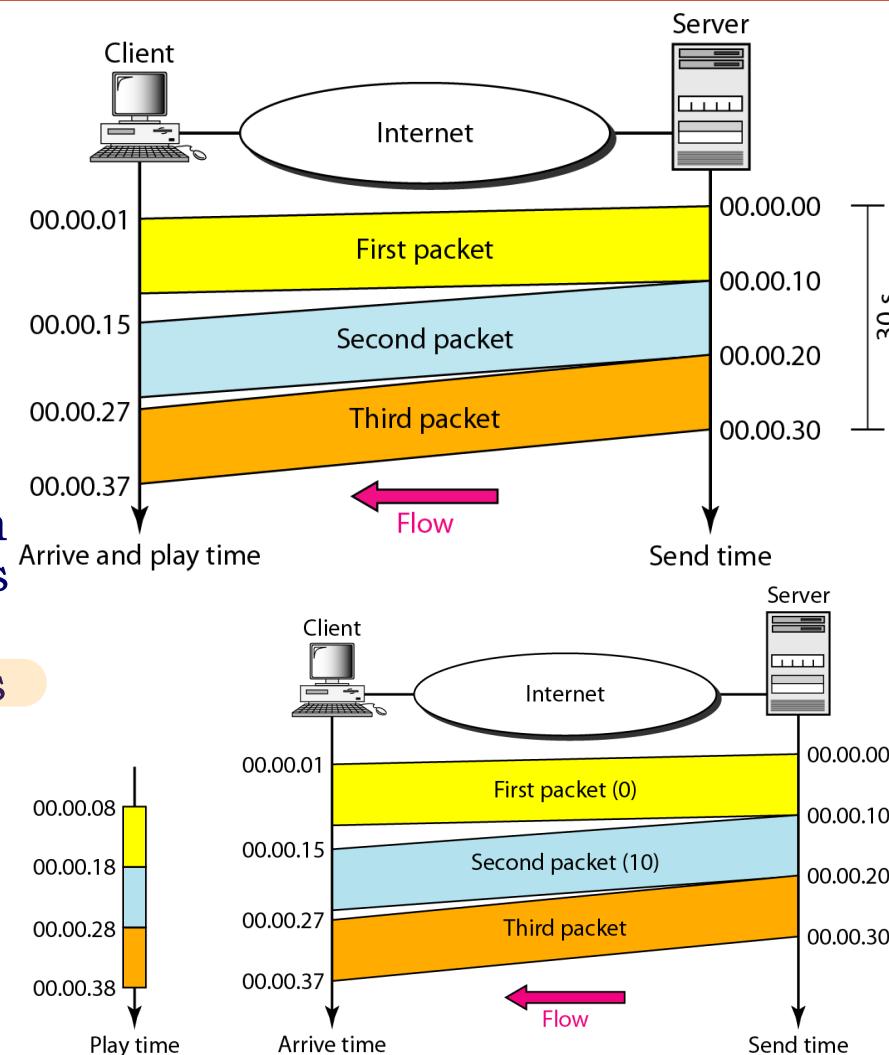
Introducción a Multimedia

- **Problemas a resolver**

- Pérdida de paquetes
- Retraso en los paquetes
- Latencia variable (jitter)

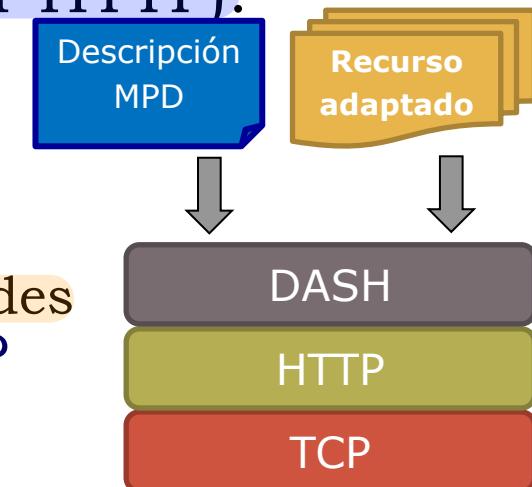
- Soluciones

- Entrelazamiento de datos (e.g., un segmento lleva las muestras pares y el siguiente las impares)
- Marcas de tiempo en los mensajes con datos
- Números de secuencia para ordenación
- Almacenamiento temporal en un buffer



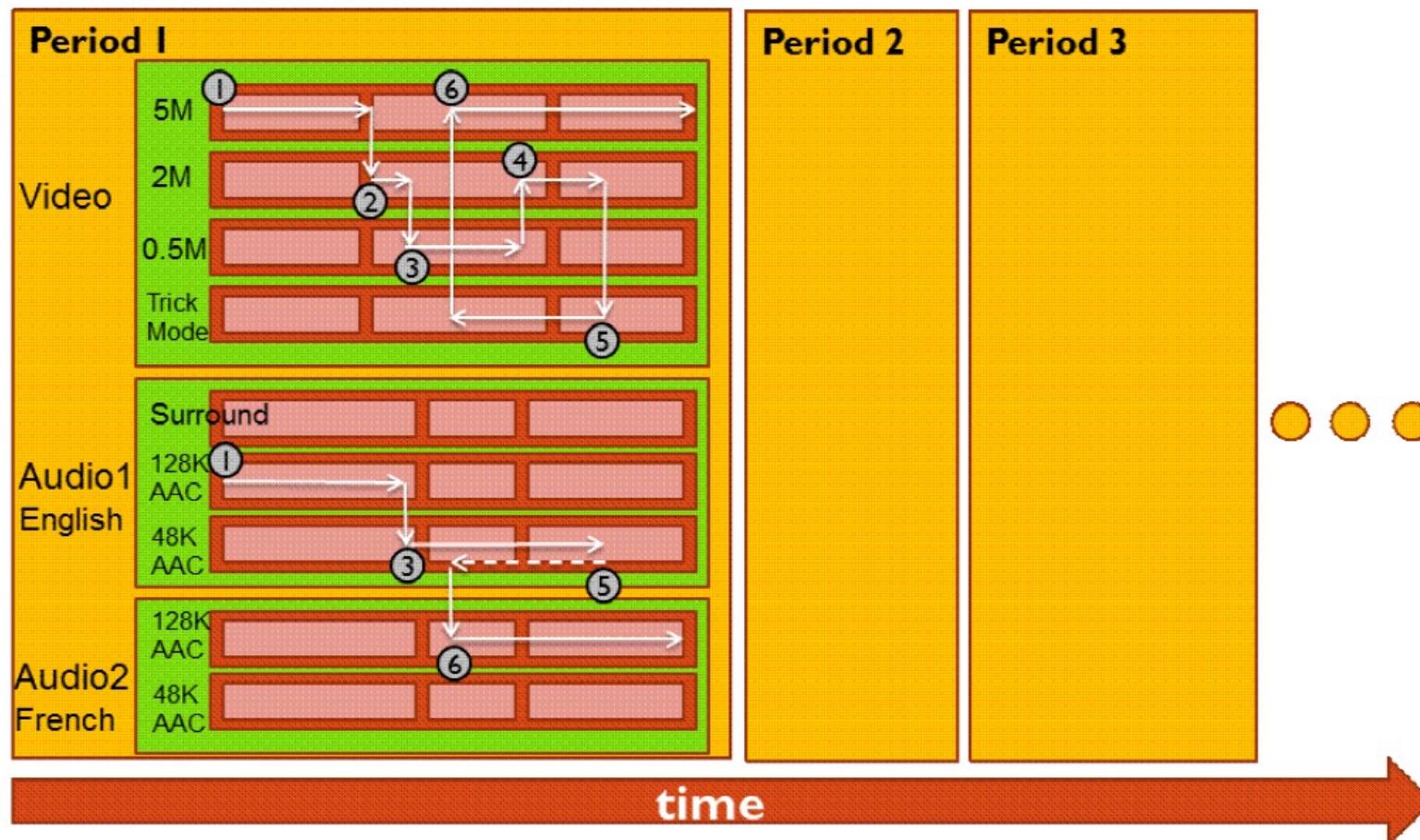
Caso de estudio 1: Netflix, Youtube...

- Los recursos multimedia ya están disponibles antes de su envío al usuario y se pueden pre-procesar
- Características deseables:
 - Localización del recurso conocida
 - Accesibilidad universal y sencilla
 - Transmisión sin errores
 - Ajustar a las características del usuario
- DASH (Dynamic Adaptive Streaming over HTTP):
 - Transmisión: HTTP
 - extendido y permitido universalmente
 - Recursos multimedia adaptables:
 - Dividido en pequeños trozos
 - Cada trozo disponible en diferentes calidades
 - ¿Cómo se conocen los trozos disponibles?
 - ¿Cómo se elige la calidad?



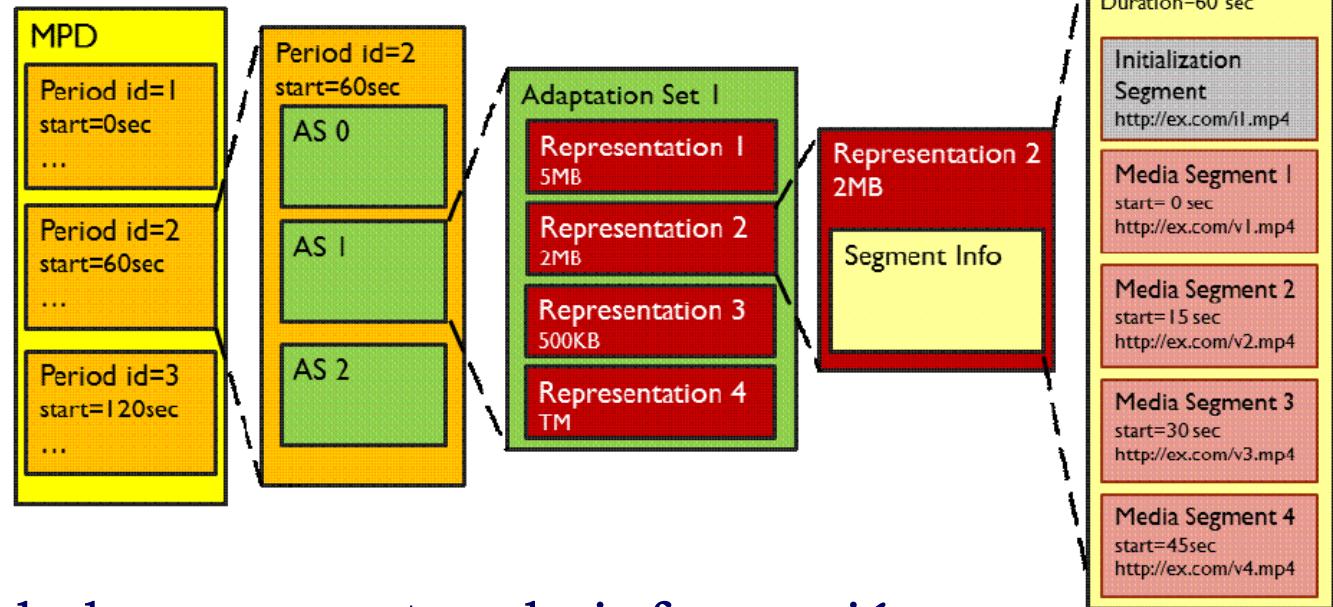
Dynamic Adaptive Streaming over HTTP (DASH)

- Esquema básico de funcionamiento a nivel de usuario:



Dynamic Adaptive Streaming over HTTP (DASH)

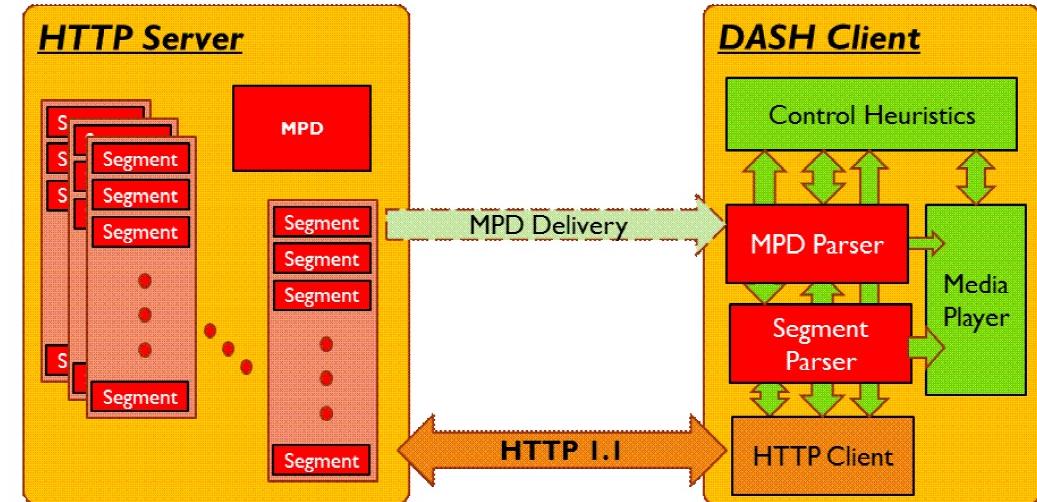
- DASH define:
 - Cómo se describe el **recurso multimedia (MPD –xml– Multimedia Presentation Description)**



- Formato de los segmentos de información
- Otras características: DRM, publicidad, selección de flujos, segmentos de longitud variable, ...

Dynamic Adaptive Streaming over HTTP (DASH)

- Funcionamiento:
 - **Servidor:** trocea y codifica en varios formatos/calidades el recurso multimedia
 - **Servidor:** Genera el **fichero MPD**
 - **Cliente:** obtiene el **fichero MPD** usando **HTTP**
 - **Cliente:** analiza **MPD**
 - **Cliente:** usa estadísticas => segmento apropiado con **HTTP**
 - **Cliente:** recibe segmento y obtiene el recurso y se lo pasa al reproductor

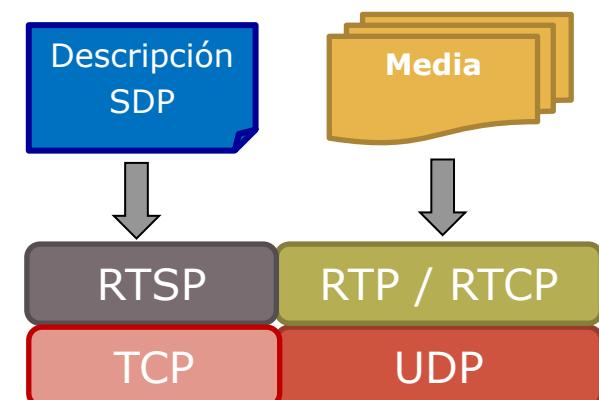


Imágenes de:

Sodagar, I. (2011). The mpeg-dash standard for multimedia streaming over the internet. *IEEE MultiMedia*, 18(4), 62-67.

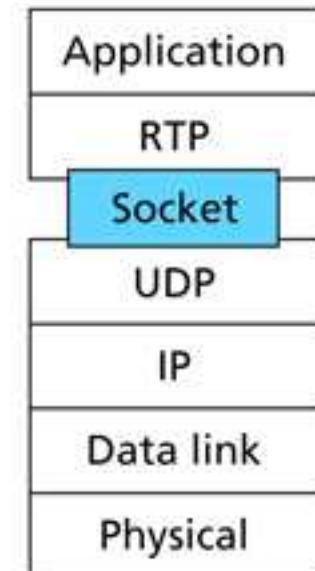
Caso de estudio 2: Videocámara IP, TV online en directo...

- El recurso multimedia se va generando a la vez que se consume. No es bidireccional.
- Características deseables:
 - Localización del recurso conocida
 - Primar la velocidad a la corrección
- Solución propuesta:
 - Transporte: UDP
 - ¿Es UDP suficiente?
 - Sincronización flujos multimedia
 - Temporización, congestión...
 - ¿Cómo controla el cliente el flujo (parar, reiniciar...) y hace control de flujo?
 - ¿Cómo se saben los flujos y características?



Protocolo de tiempo real (RTP)

- Es un protocolo del nivel de aplicación que permite la transferencia de datos con características de tiempo real e información de control.
- Es un protocolo desarrollado por la comunidad de Internet y se recoge en el RFC 3550 (que reemplaza al RFC 1889).
- Este protocolo está compuesto de dos protocolos que colaboran estrechamente entre sí: RTP y RTCP
- Este protocolo utiliza los servicios de UDP para la transmisión de sus paquetes.



Protocolo de tiempo real (RTP)

- El protocolo define un único paquete de datos, que se denomina *paquete RTP*:
 - Tipo de carga útil
 - Número de secuencia
 - Marca de tiempo
 - Identificador de fuente de sincronización (identificador de las comunicación - Coordinador)
 - Identificadores de fuente de contribución (identificador del origen de este flujo de vídeo/audio, ...)

Payload type (PT)	Value	bit 0	4	8 9	16	31
5	DVI4	V	P	X	CC	
26	JPEG			M	Payload Type	Sequence Number
31	H.261				Timestamp	
32	MPEG I/II				Synchronization Source (SSRC) Identifier	
					Contributing Source (CSRC) Identifier	
					•	
					•	
					•	
					Contributing Source (CSRC) Identifier	

Protocolo de tiempo real (RTP)

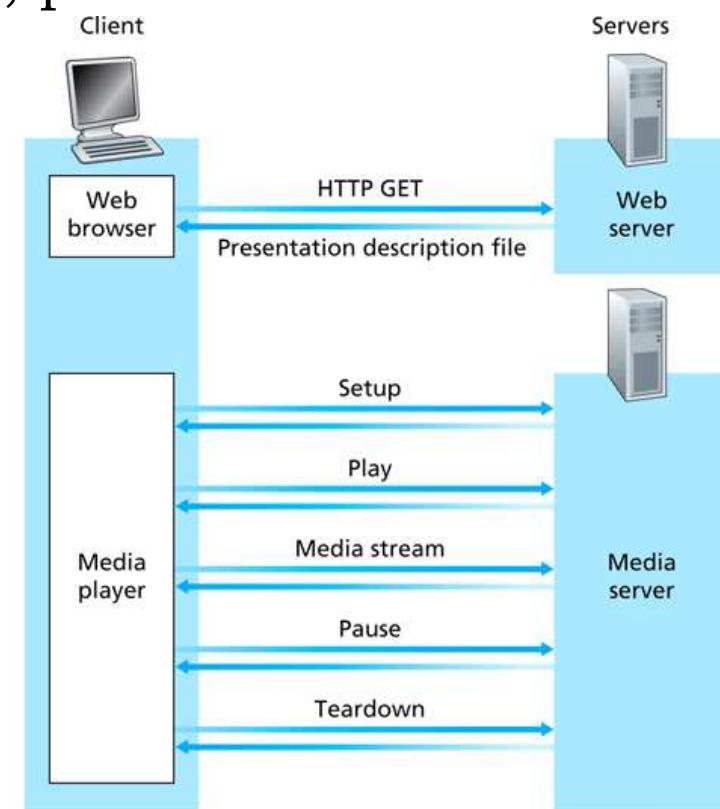
- RTCP (*Real-Time Control Protocol*) ofrece los siguientes servicios:
 - Monitorización de la calidad de servicio:
 - Permite adaptar la codificación a la calidad
 - Monitorización control de congestión.
 - Identificación de la fuente (por si hay fallos o reinicios)
 - Sincronización entre flujos
 - Indicación de tiempo real y su correspondiente *timestamp*
 - Información de control escalable
 - Estos paquetes se envían de forma periódica a todos los nodos
 - Mantienen datos para calcular estadísticas (rtt, jitter, ...)
 - Cambian de frecuencia para usar un máximo de 5% de todo el tráfico de la sesión (RTP).

SRTP y SRTCP

- **SRTP** (Secure Real-Time Transport Protocol)
 - Define un perfil de RTP para proporcionar:
 - Cifrado
 - Autenticación de mensajes
 - Integridad
- **SRTCP** (Secure RTCP)
 - Añade características de seguridad al protocolo de control.

Protocolo de streaming en tiempo real (RTSP)

- Real-Time Streaming Protocol
- Este protocolo se utiliza para intercambiar comandos de control de la transmisión de datos de tiempo real con el servidor
- No se encarga de transmitir los datos, para eso se utiliza RTP/UDP, UDP o TCP.
- Es el equivalente al “mando a distancia”
- Los mensajes intercambiados son de texto plano y similar a HTTP



Protocolo de streaming en tiempo real (RTSP)

1

```
C: SETUP movie.Mjpeg RTSP/1.0
C: CSeq: 1
C: Transport: RTP/UDP; client_port= 25000

S: RTSP/1.0 200 OK
S: CSeq: 1
S: Session: 123456

C: PLAY movie.Mjpeg RTSP/1.0
C: CSeq: 2
C: Session: 123456

S: RTSP/1.0 200 OK
S: CSeq: 2
S: Session: 123456

C: PAUSE movie.Mjpeg RTSP/1.0
C: CSeq: 3
C: Session: 123456

S: RTSP/1.0 200 OK
S: CSeq: 3
S: Session: 123456
```

2

```
C: PLAY movie.Mjpeg RTSP/1.0
C: CSeq: 4
C: Session: 123456

S: RTSP/1.0 200 OK
S: CSeq: 4
S: Session: 123456

C: TEARDOWN movie.Mjpeg RTSP/1.0
C: CSeq: 5
C: Session: 123456

S: RTSP/1.0 200 OK
S: CSeq: 5
S: Session: 123456
```

Session Description Protocol (SDP)

- Es un lenguaje que permite describir los recursos multimedia, temporización y características de conexión de una reunión:
 - Líneas del tipo <letra>=<configuración>
- Ejemplo:

v=0

o=RySD 2890844526 2890844526 IN IP4 10.120.42.3

s=Meeting

c=IN IP4 10.120.42.3

...

m=audio 49170 RTP/AVP 0 8 97

a=rtpmap:0 PCMU/8000

a=rtpmap:8 PCMA/8000

a=rtpmap:97 iLBC/8000

m=video 51372 RTP/AVP 31 32

a=rtpmap:31 H261/90000

Sesión llamada “Meeting”

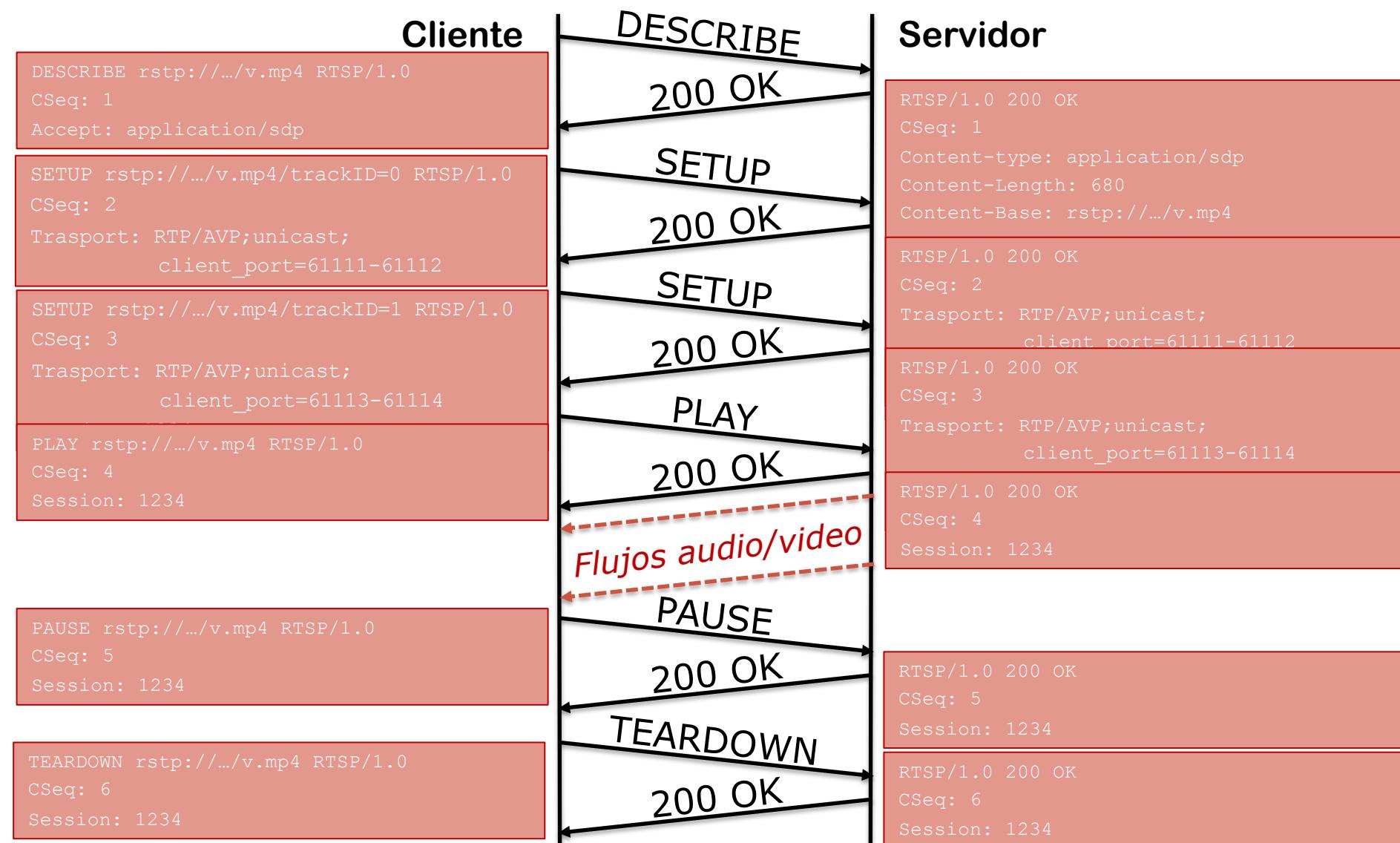
Creada por RySD.

Los recursos están en 10.120.42.3

Dos flujos multimedia (video y audio)

- Audio con 3 posibles codificaciones y se envía por el puerto 49170/RTP
- Video con 1 sola posible codificación y se envía por el puerto 51372/RTP

Ejemplo de sesión de streaming



Caso de estudio 3: Discord, Meets, MS Teams, WhereBy...

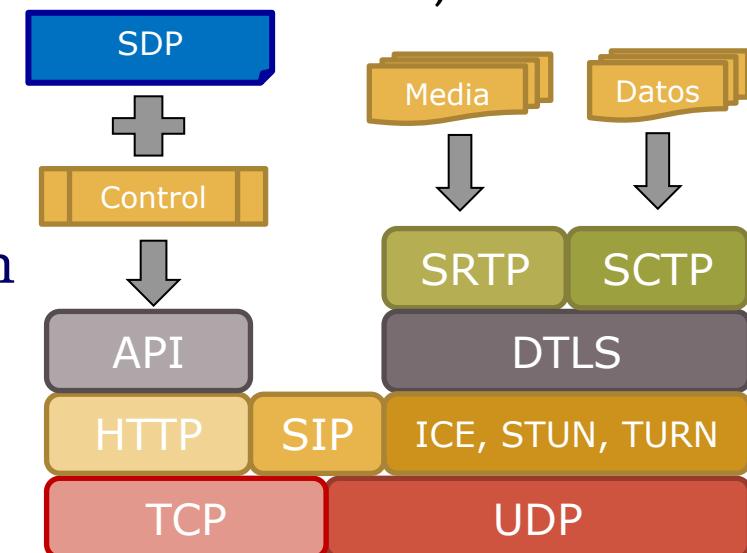
- El recurso multimedia se va generando a la vez que se consume. Es multidireccional (videoconferencia).

- Características deseables:

- Localización de los extremos
- Comunicaciones P2P y multicast
- Primar la velocidad a la corrección
- Utilizable desde el navegador
- Transmisión de datos fiable
- Seguridad

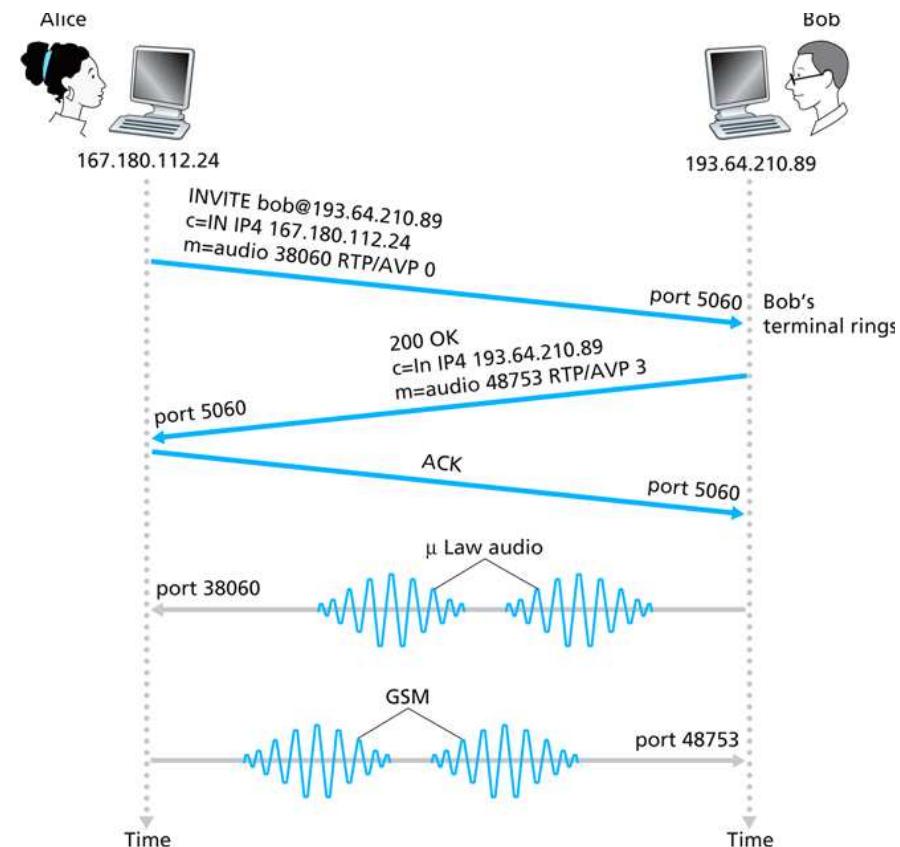
- WebRTC:

- Base conocida: UDP, SRTP/SRTCP, SDP
- Localización de usuarios: SIP
- Usuarios como “servidor”: ICE, STUN, TURN
- Más seguridad: DTLS
- Datos fiables: SCTP
- Uso web: HTTP + API Javascript



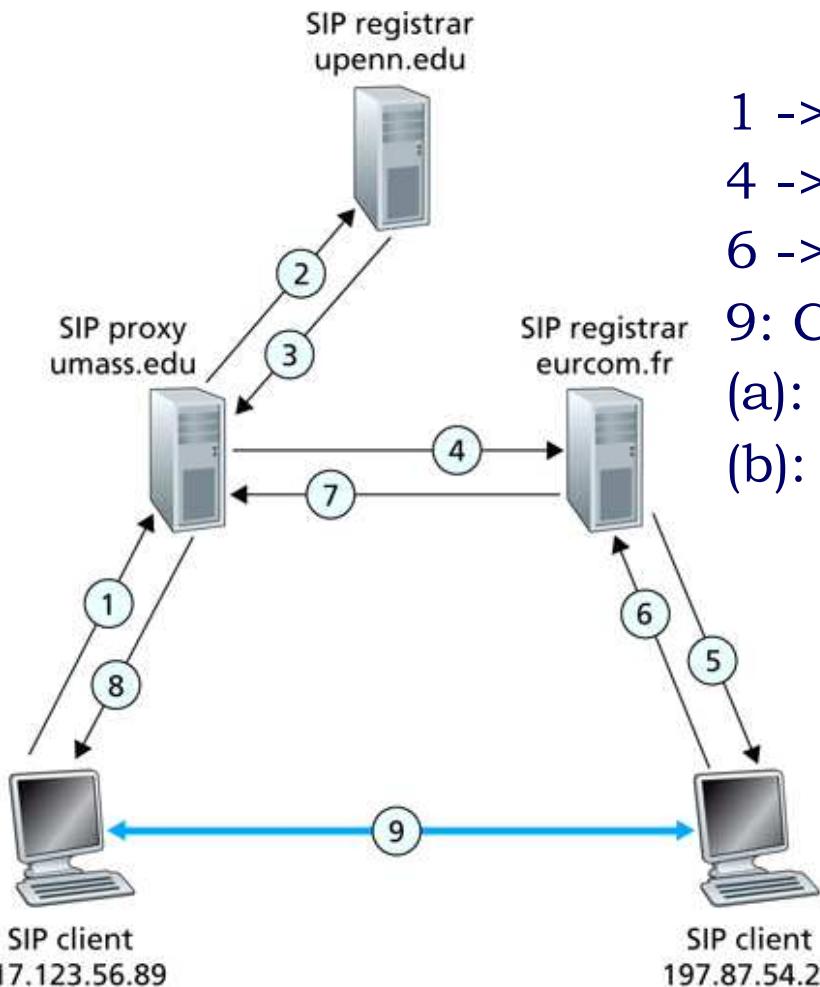
Protocolo de inicio de sesión (SIP)

- Este protocolo se utiliza para establecer conexiones multimedia entre dos usuarios (muy usado en VoIP)
- No se encarga de transmitir los datos, para eso se utiliza RTP/UDP, UDP o TCP
- Mensajes:
 - INVITE
 - ACK
 - BYE
 - OPTIONS
 - CANCEL
 - REGISTER



Protocolo de inicio de sesión (SIP)

- Permite localizar la IP de un usuario gracias a un sistema de registro



1 -> 2 -> 3: INVITE (buscando destino)

4 -> 5: INVITE (avisando al destino)

6 -> 7 -> RINGING (180) y OK (200)

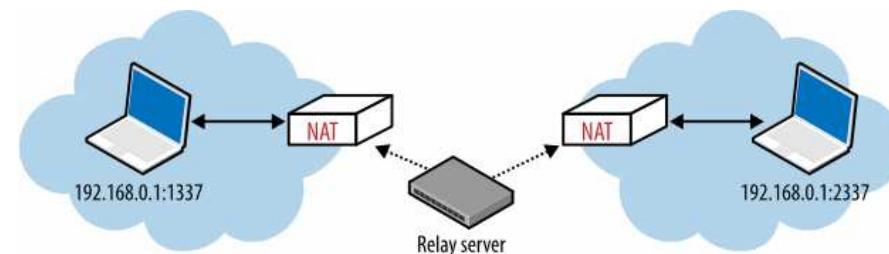
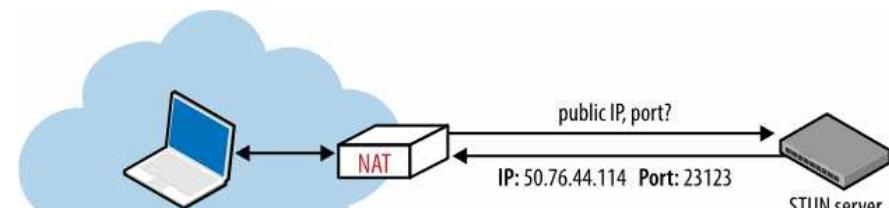
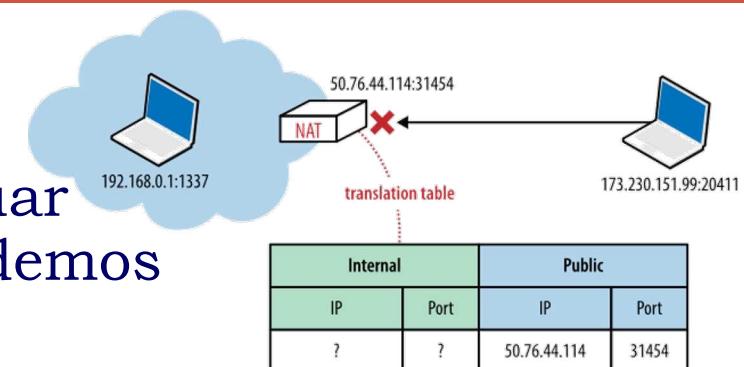
9: Conversación:

(a): origen -> destino: ACK

(b): origen <-> destino: conversación (RTP)

Funcionando como servidor temporal: ICE, TURN, STUN

- Mayoritariamente se emplea NAT y dificulta actuar de servidor
 - En estos sistemas queremos actuar de servidor pero temporal (no podemos configurar de forma permanente)
- STUN: Nos conectamos a un servidor externo (STUN)
 - Permite que conozcamos la IP/puerto público
 - Se añade la entrada a la tabla NAT
- TURN: Dependiendo de la configuración puede no ser posible -> comunicación con nodo intermedio
- ICE: Permite elegir entre STUN y TURN (lo más efectivo)



Otros protocolos: DTLS y SCTP

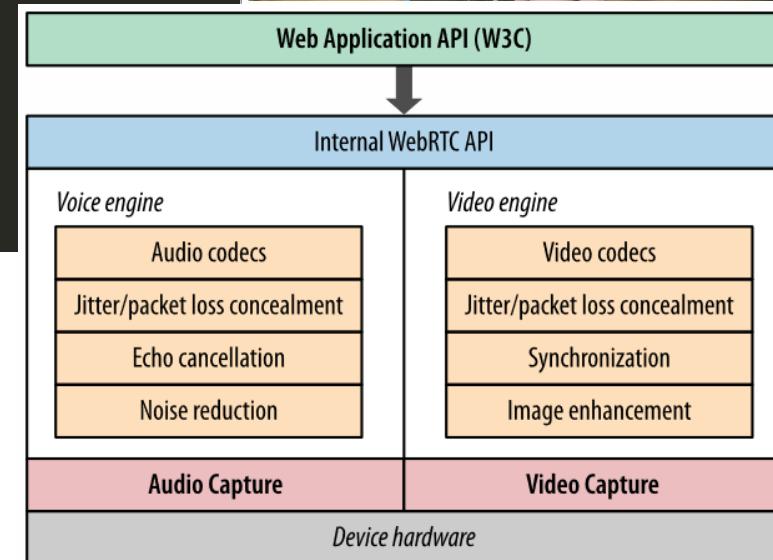
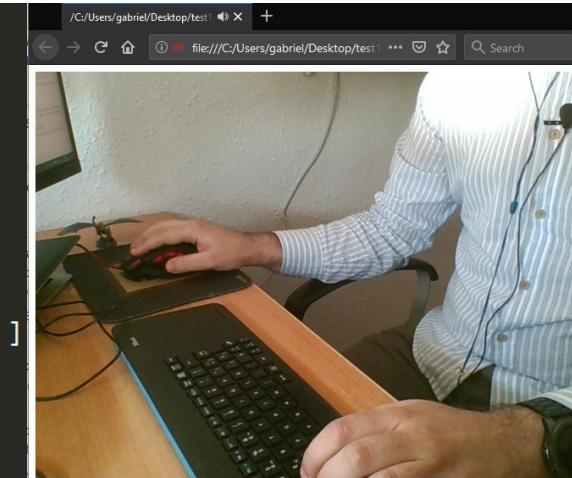
- DTLS:
 - Implementación de TLS sobre datagramas UDP
 - Ofrece encriptación, integridad y autenticación
 - En SRTP solo se usa al inicio para compartir información inicial de configuración
- SCTP:
 - Alternativa a TCP y UDP pero poco usado debido a problemas de la configuración con equipos
 - Usarlo sobre UDP soluciona esos problemas
 - Características:
 - Envía mensajes (como UDP)
 - Orientado a la conexión, confiable, con control de flujo y de congestión y secuenciación (como TCP)
 - Selección y monitorización de múltiples flujos (como HTTP/2)
 - Mensajes fuera de orden y protección contra ataques.

Uso

- Ofrece un API sencillo para su uso en la web

```
<html>
<body>
  <p><video autoplay playsinline></video></p>
  <script src="https://webrtc.github.io/adapter/adapter-latest.js"></script>
  <script>
    'use strict';
    const constraints = {
      audio: true,
      video: {
        mandatory: { width: { min: 320 }, height: { min: 180 } },
        optional: [ { width: { max: 1280 }}, { frameRate: 30 }, { facingMode: "user" } ]
      }
    };
    async function init() {
      const stream = await navigator.mediaDevices.getUserMedia(constraints);
      const video = document.querySelector('video');
      const videoTracks = stream.getVideoTracks();
      video.srcObject = stream;
    }
    init()
  </script>
</body>
</html>
```

<https://webrtc.github.io/samples/>



- Define motores de audio/video