Teoría de las Comunicaciones

Segundo Cuatrimestre del 2017

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Argentina

Nivel de Aplicación

Aplicaciones

Capa de aplicación

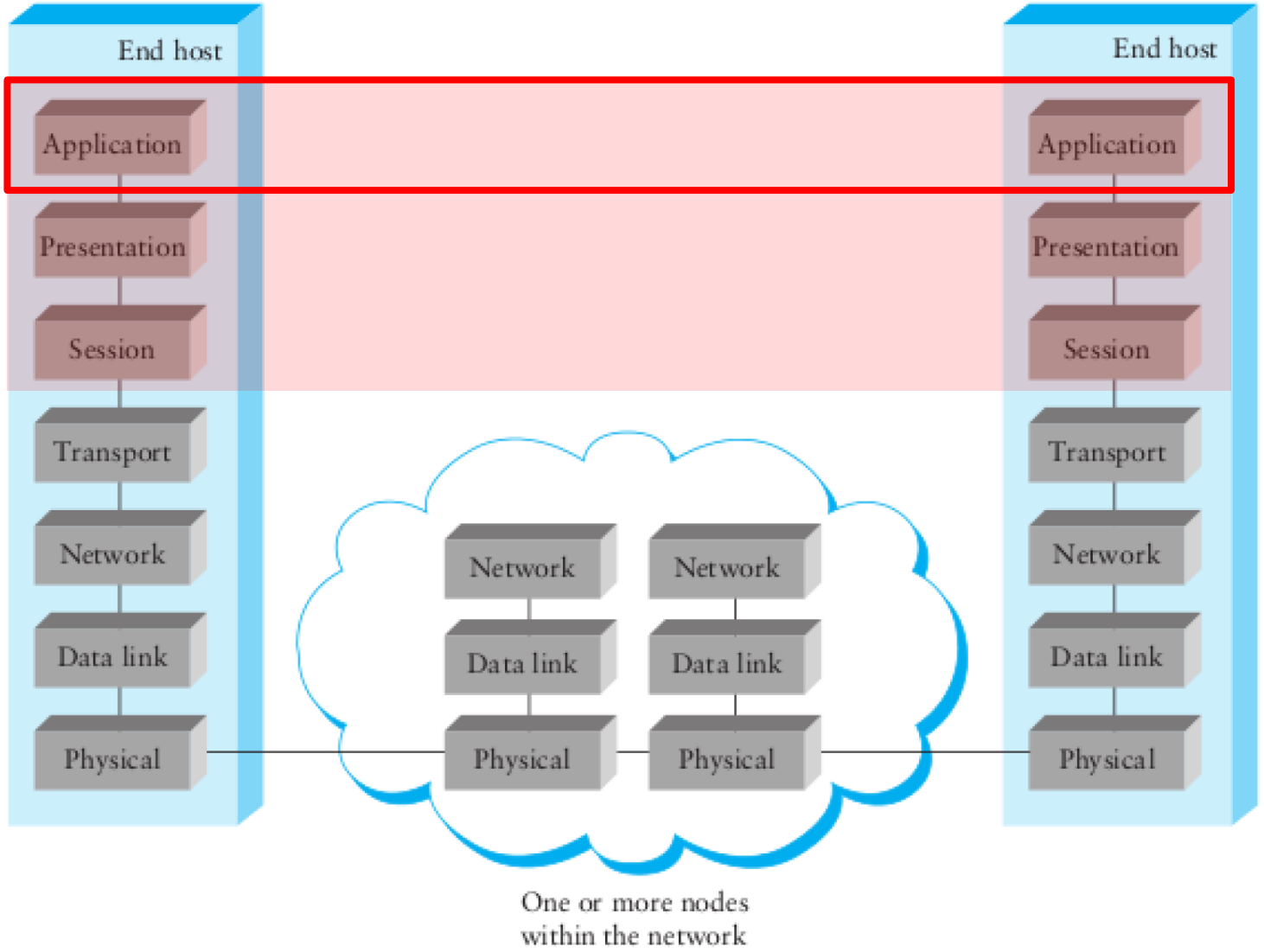
Nuestros objetivos:

- Aspectos conceptuales de los protocolos de las aplicaciones de red.
  - Modelos de servicio de la capa de transporte
  - Paradigma cliente-servidor.
- Aprendizaje de protocolos por medio del estudio de protocolos a nivel de aplicación.
  - DNS
  - SMTP / POP3 / IMAP
  - HTTP

Adaptado de:

Redes de computadores: un enfoque descendente basado en Internet,
2ª edición.
Jim Kurose, Keith Ross

Arquitectura en capas



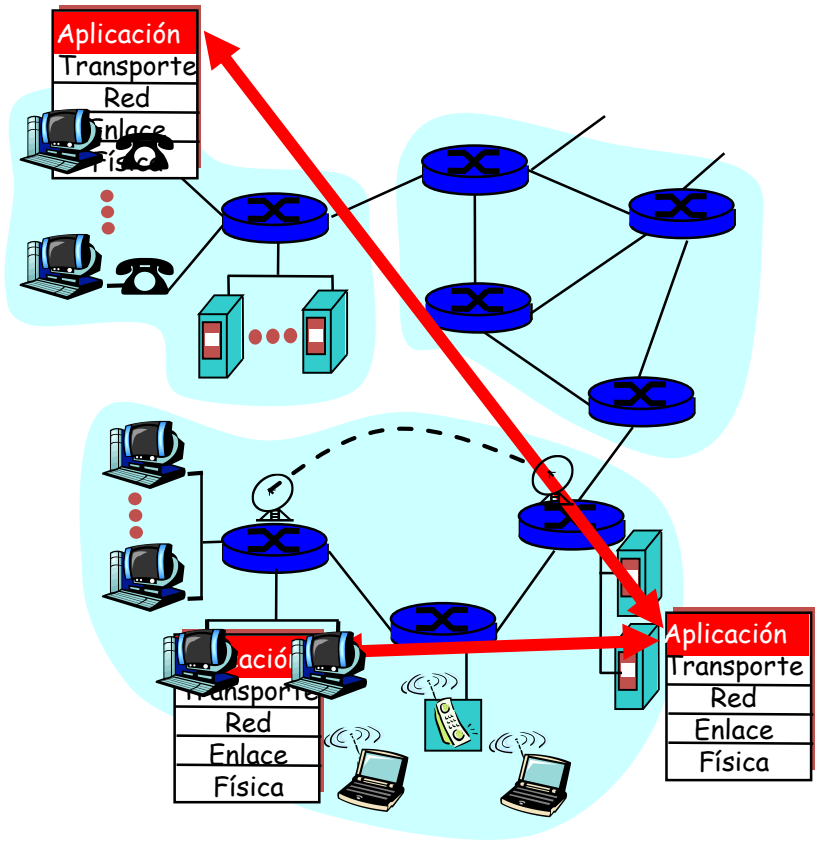
Aplicaciones y protocolos de la capa de aplicación

Aplicación: comunicación, procesos distributivos.

- **Por ejemplo:** correo electrónico, web, compartir archivos entre iguales, mensajería instantánea.
- Funcionamiento en **sistemas finales** (hosts).
- Intercambio de **mensajes** para la implementación de aplicaciones.

Protocolos de capa de aplicación

- Una "parte" de la aplicación.
- Definición de **mensajes** intercambiados entre las aplicaciones y las **acciones** tomadas en los hosts.
- Uso de **servicios** de comunicación proporcionados por los protocolos de capas inferiores (TCP, UDP).



Aplicaciones de red: terminología

Proceso: programa que se ejecuta en un host.

- En el mismo host, dos procesos se comunican utilizando comunicación interproceso (definidos por un sistema operativo).
- Los procesos que se ejecutan en diferentes hosts se comunican con un protocolo de capa de aplicación.

Agentes de usuario: **interfaces** con un usuario "por encima" y una red "por debajo".

- Implementa interfaces de usuario y protocolos a nivel de aplicación.
  - Web: navegador.
  - Correo electrónico: lector de correo.
  - Transmisión de audio/vídeo: reproductor multimedia.

Características de los protocolos de capa de aplicación

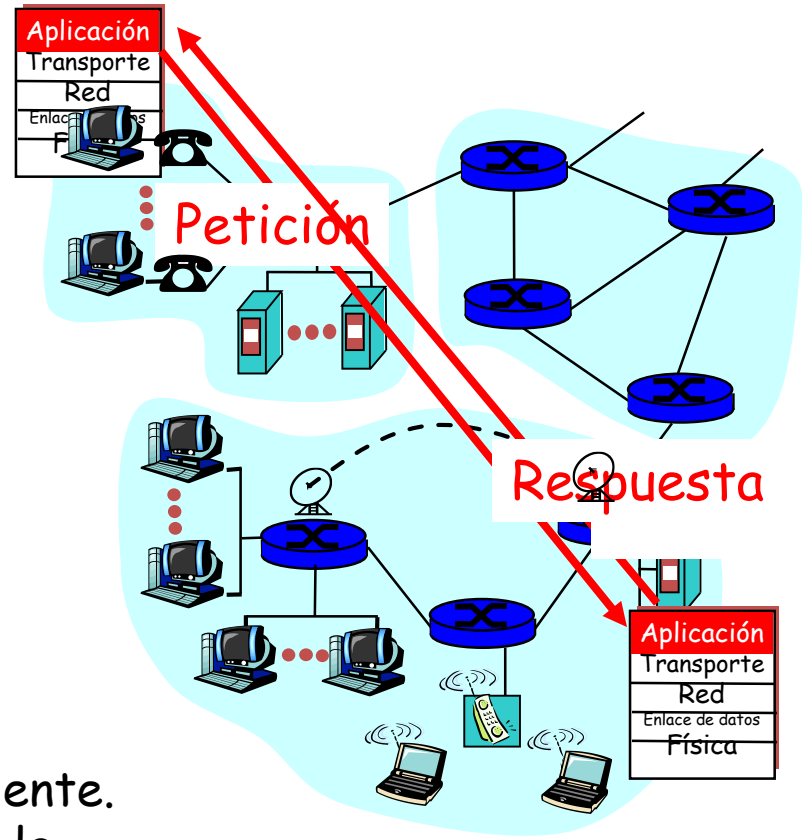
- Tipos de mensajes intercambiados: típicamente **de petición** y **de respuesta**.
- **Sintaxis** de los tipos de mensajes:
  - qué campos hay en los mensajes y su estructura
- **Semántica** de los campos:
  - significado de la información en los campos.
- **Reglas** que determinan cuándo y cómo los procesos envían y responden a los mensajes.

Protocolos de dominio público:

- Definidos en RFC.
- Permiten interoperabilidad.
- Por ejemplo: HTTP, SMTP.

Paradigma tipo **cliente-servidor**

La aplicación de red "típicamente" tiene dos partes: *el cliente* y *el servidor*



Cliente:

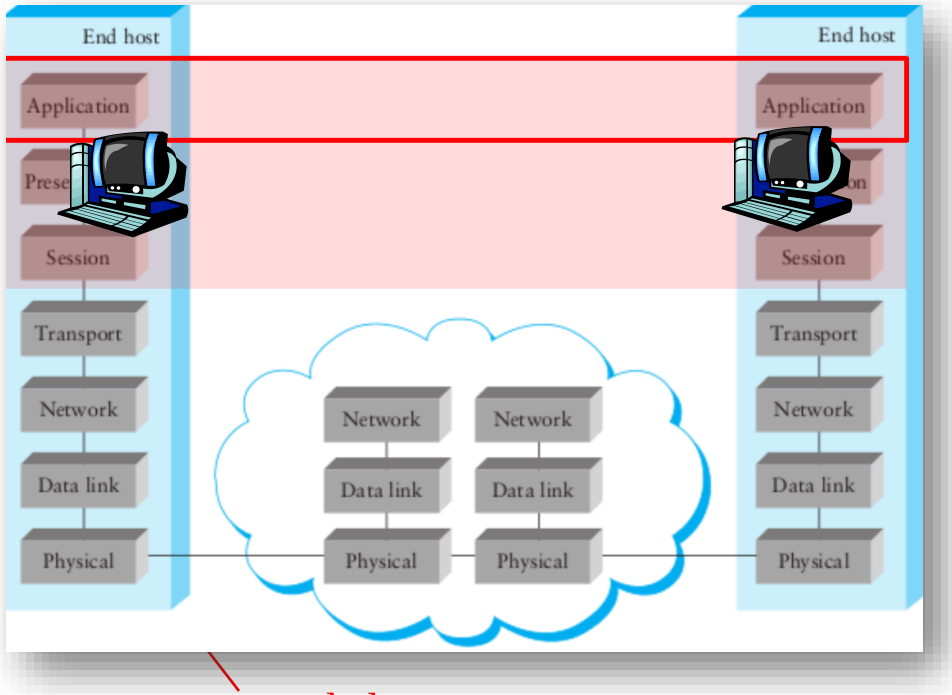
- Inicia el contacto con el servidor ("habla primero").
- Normalmente solicita un servicio del servidor.
- Web: cliente implementado en el navegador. Correo electrónico: en un lector de correo, etc.

Servidor:

- Proporciona el servicio solicitado al cliente.
- Por ejemplo, el servidor de Web envía la página Web solicitada, el servidor de correo entrega el correo electrónico, etc.

Procesos que se comunican a través de la red

- El proceso envía/recibe mensajes hacia/desde su socket.
- El socket es análogo a una puerta:
  - El **proceso emisor** envía el mensaje por su puerta.
  - Asume la existencia de una **infraestructura de transporte** al otro lado de la puerta que transportará el mensaje hacia un socket en el **proceso receptor**.



- API: (1) elección del protocolo de transporte; (2) posibilidad de fijar algunos parámetros -opciones- de operación.

Servicios de los protocolos de transporte de Internet

Servicio TCP:

- *Orientado a la conexión:* Sistema requerido entre el cliente y el servidor.
- *Transporte fiable* entre el proceso emisor y el receptor.
- *Control de flujo:* el emisor no debe sobrecargar al receptor.
- *Control de congestión:* regulación del emisor si la red se sobrecarga.
- *No proporciona:* temporización, garantías de un ancho de banda mínimo.

Servicio UDP:

- Transferencia de datos no fiable entre el proceso emisor y el receptor.
- No proporciona: sistema de conexión, fiabilidad, control de flujo, control de congestión, temporización y garantía de ancho de banda.

¿Qué servicio de transporte necesita una aplicación? Criterios de selección

Pérdida de datos

- Ciertas aplicaciones (por ejemplo, audio) pueden tolerar algunas pérdidas.
- Otras aplicaciones (por ejemplo, transferencia de archivos, Telnet) requieren un 100% de **transferencia confiable** de datos.

Temporización

- Algunas aplicaciones (por ejemplo, Telefonía sobre Internet, juegos interactivos) requieren un retardo artificial para ser "efectivas" (eliminar el jitter)

Ancho de banda

- Algunas aplicaciones (por ejemplo, multimedia) requieren un mínimo de ancho de banda para ser "efectivas".
- Otras aplicaciones ("aplicaciones flexibles") hacen uso de cualquier ancho de banda que tengan a su disposición.

Requisitos de los servicios de transporte para aplicaciones comunes

Aplicación	Pérdida de datos	Ancho de banda	Sensible al tiempo
Transf. de archivos	No pérdida	Flexible	No
Correo electrónico	No pérdida	Flexible	No
Documentos Web	No pérdida	Flexible	No
Audio/vídeo de tiempo real	Tolerante	Audio: 5Kbps-1Mbps Vídeo:10Kbps-5Mbps	Sí, 100 mseg
Audio/vídeo almacenado	Tolerante	Igual que el anterior	Sí, pocos seg
Juegos interactivos	Tolerante	Pocos Kbps-10Kbps	Sí, 100 mseg
Mensajería instantánea	No pérdida	Flexible	Sí y no

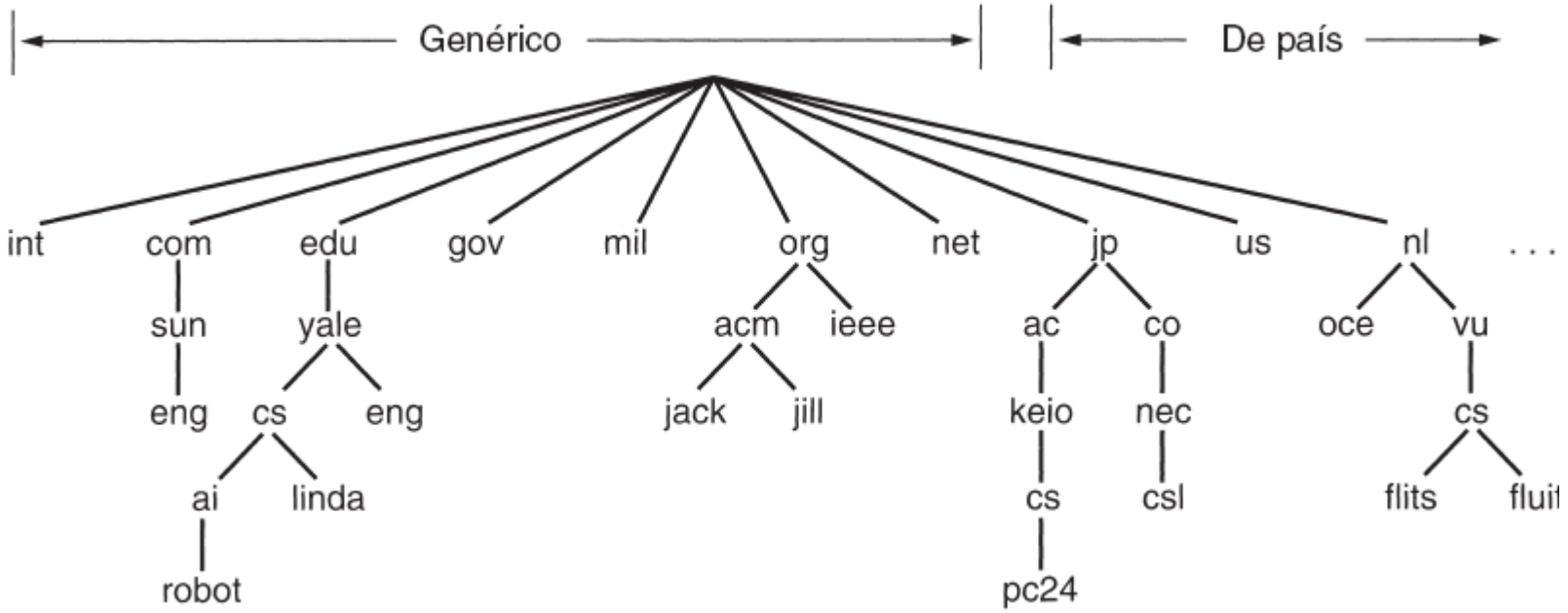
Aplicaciones de Internet: aplicación, protocolos de transporte

Aplicaciones	Protocolo de la capa de aplicación	Protocolo de transporte subyacente
Correo electrónico	SMTP [RFC 2821]	TCP
Acceso a terminales remotos	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
Transferencia de archivos	FTP [RFC 959]	TCP
Flujo de multimedia	HTTP (YouTube, Netflix) Propietario (Real Networks)	TCP o UDP
Telefonía Internet	SIP [RFC 3261], RTP [RFC 3550], Propietario (Skype)	Típicamente UDP A veces TCP

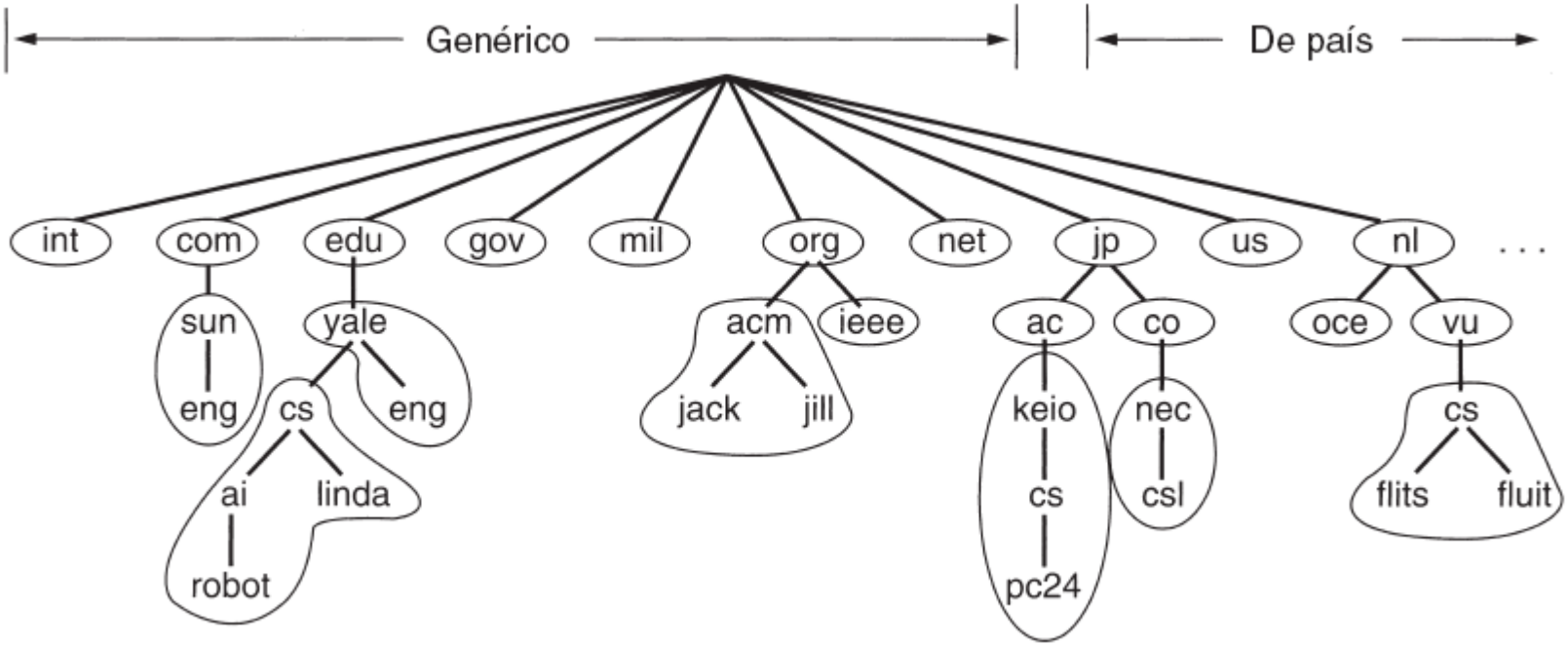
Nivel de Aplicación

DNS: Domain Name System

DNS : espacio de nombres



Servidores de nombres: DNS zones

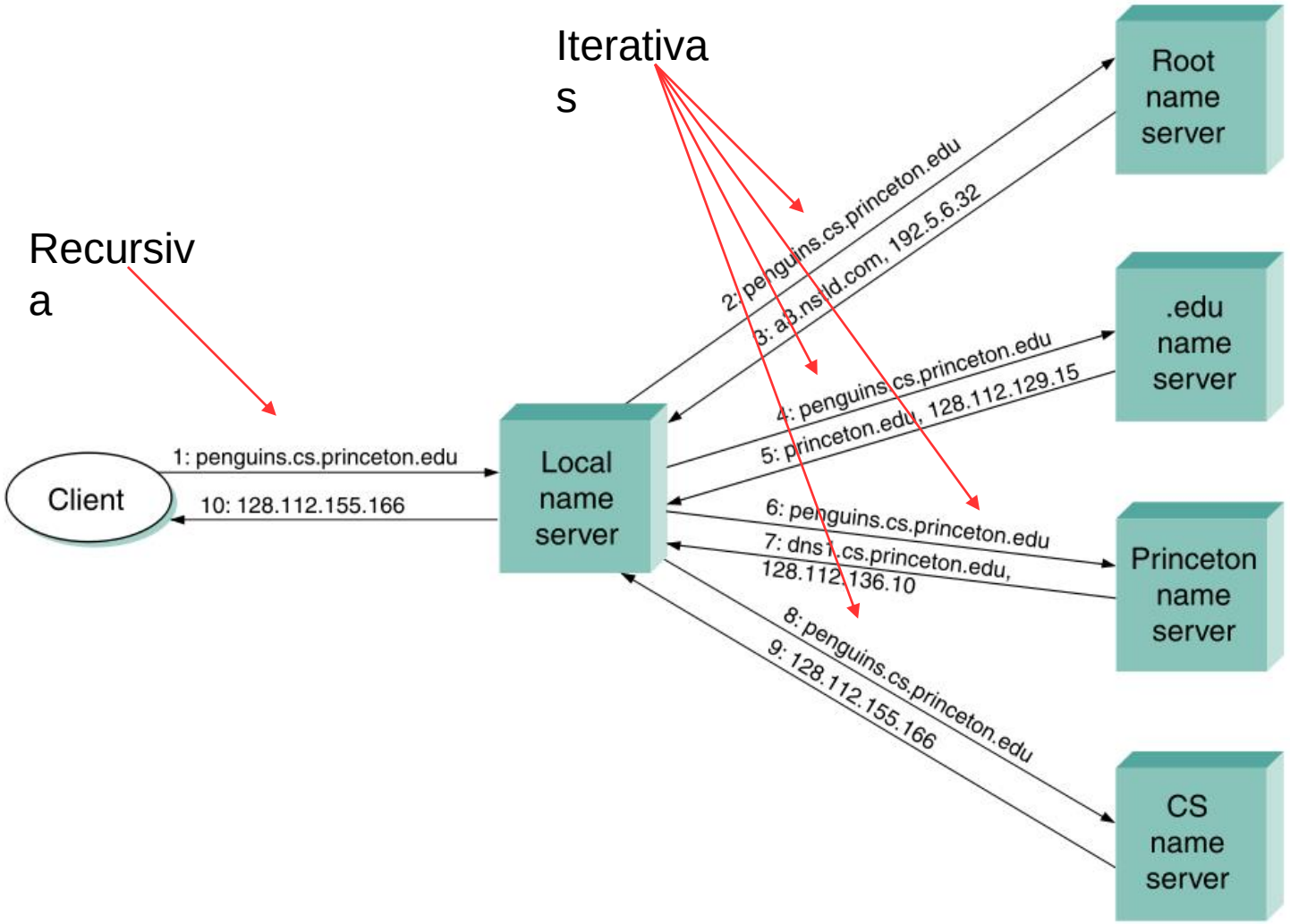


Registros DNS: tiene cinco tuplas

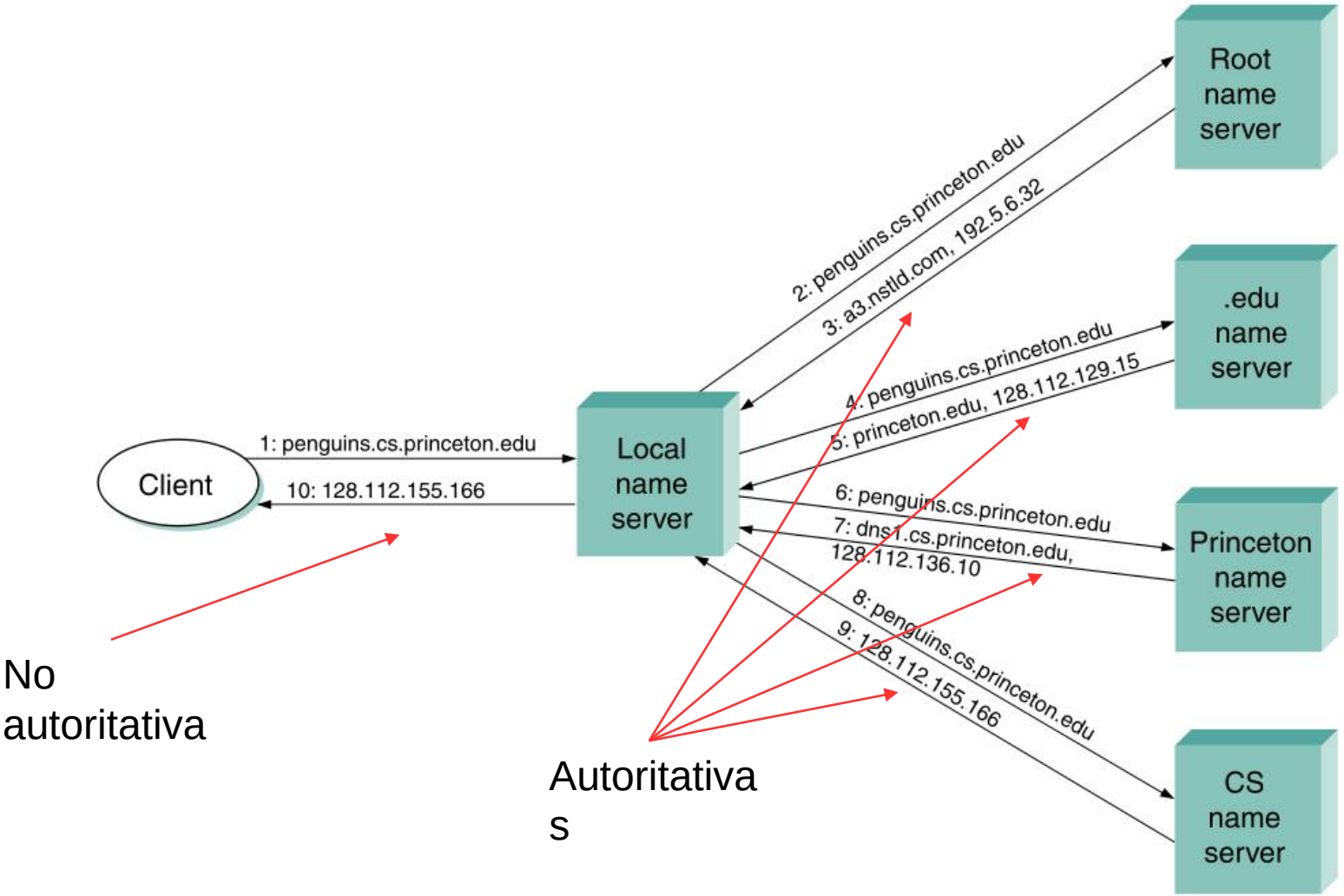
- Nombre_dominio Tiempo_de_vida Clase Tipo Valor

Tipo	Significado	Valor
SOA	Inicio de autoridad	Parámetros para esta zona
A	Dirección IP de un *host*	Entero de 32 bits
MX	Intercambio de correo	Prioridad, dominio dispuesto a aceptar correo electrónico
NS	Servidor de nombres	Nombre de un servidor para este dominio
CNAME	Nombre canónico	Nombre de dominio
PTR	Apuntador	Alias de una dirección IP
HINFO	Descripción del *host*	CPU y SO en ASCII
TXT	Texto	Texto ASCII no interpretado

Tipos de consulta: iterativa y recursiva



Tipos de respuestas: autoritativas y no autoritativas



Nivel de Aplicación

SMTP: Simple Mail Transfer Protocol

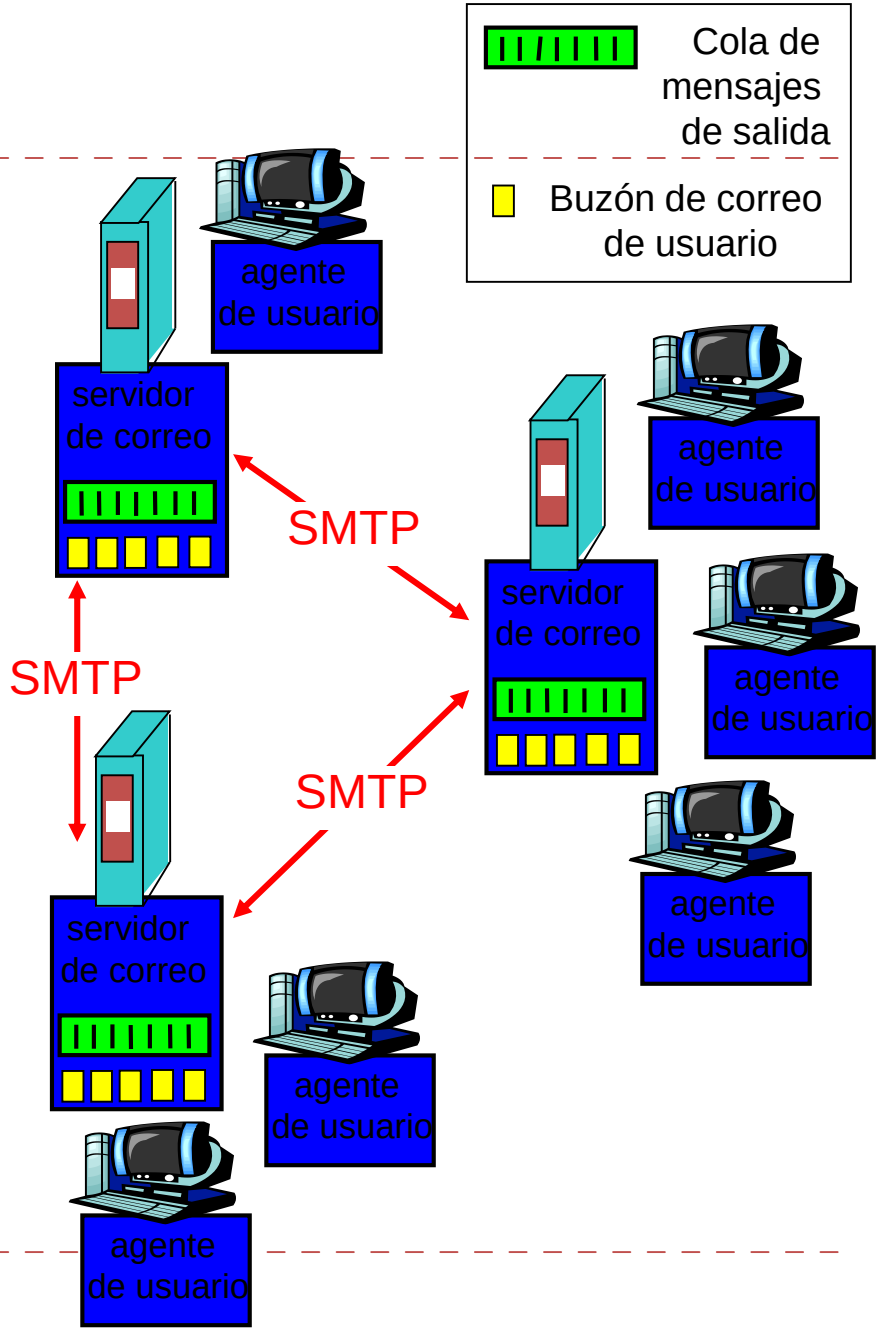
Correo electrónico

Los tres componentes principales:

- Agentes de usuario.
- Servidores de correo.
- Protocolo simple de transferencia de correo: SMTP.

Agente usuario

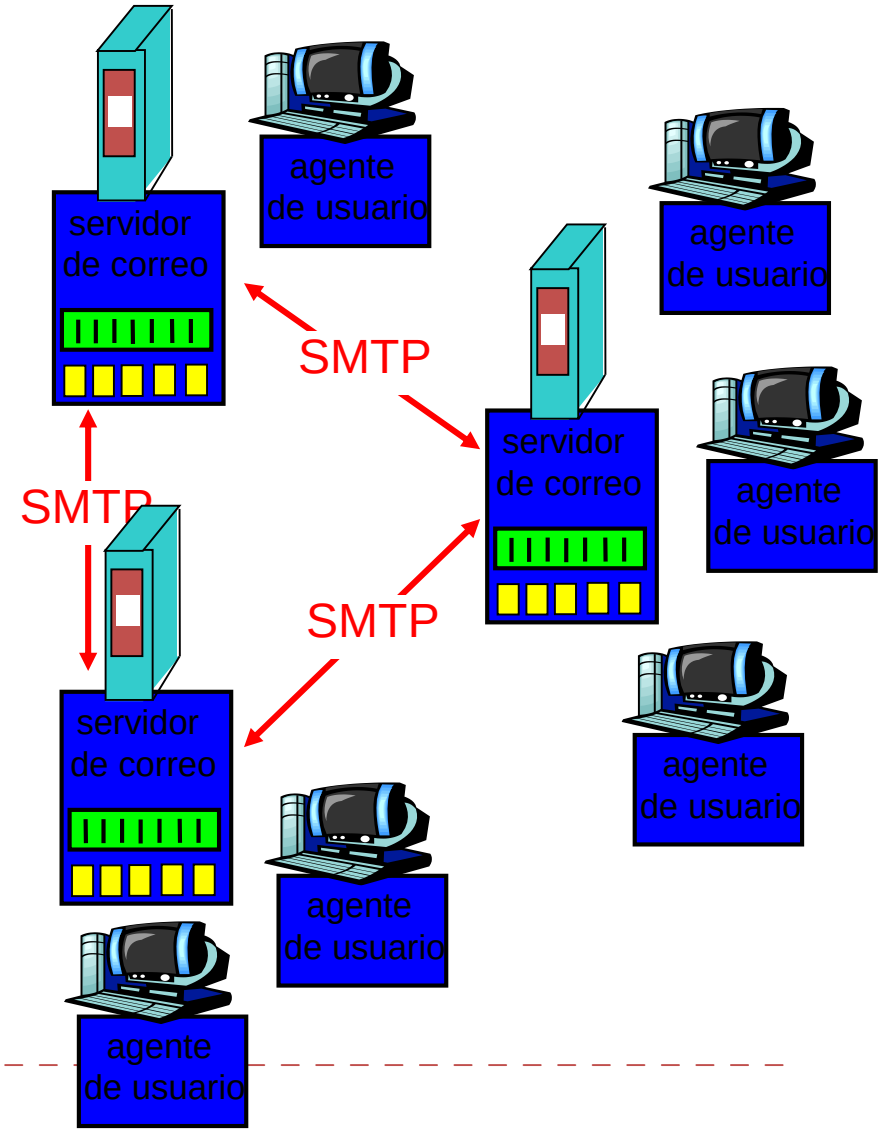
- También conocido como "lector de correo".
- Composición, edición y lectura de mensajes de correo.
- Por ejemplo, Eudora, Outlook, elm, Netscape Messenger.
- Salida y entrada del los mensajes almacenados en el servidor.



Correo electrónico: servidores de correo

Servidores de correo:

- Buzón de correo: contiene los mensajes de entrada del usuario.
- Cola de mensajes: mensajes de correo de salida (para ser enviados).
- Protocolo SMTP: entre servidores para mandar mensajes de correo electrónico.
  - Cliente: envía correo al servidor.
  - "Servidor": recibe correo de otro servidor.

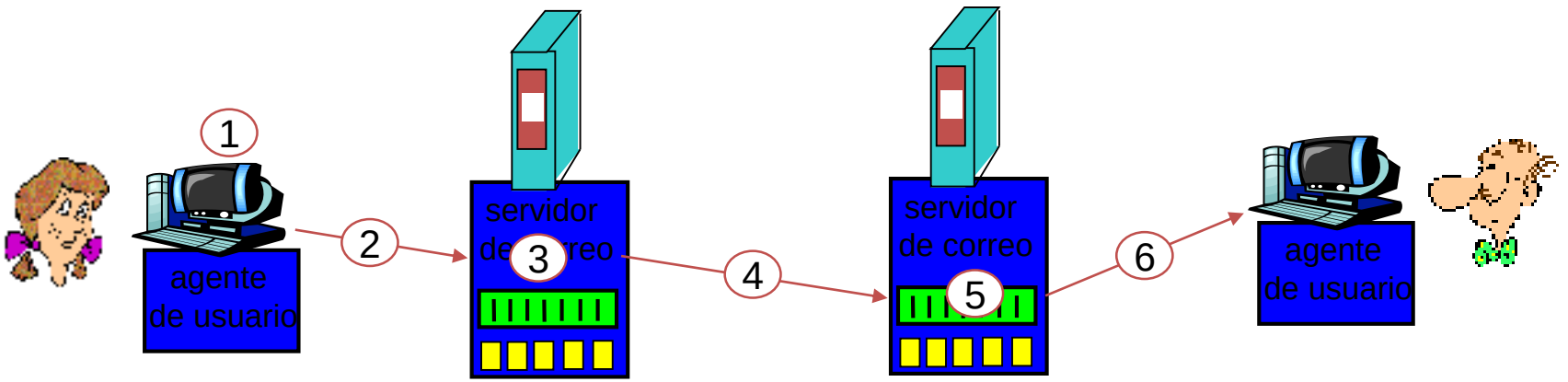


Correo electrónico: SMTP [RFC 2821]

- Utiliza TCP para transferir con seguridad el mensaje de correo electrónico del cliente al servidor, puerto 25.
- Transferencia directa: del servidor que envía al servidor que recibe.
- Las tres fases de la transferencia son:
  - "Acuerdo" (saludo).
  - Transferencia de mensajes.
  - Cierre.
- Interacción comando/respuesta:
  - Comandos: texto ASCII.
  - Respuesta: código de estatus y frase.
- Los mensajes deben tener siete bits en ASCII.

Ejemplo: Alicia le envía un mensaje a Roberto

1) Alicia utiliza su agente usuario para componer el mensaje "a" `bob@escuela.edu`
2) El agente de usuario de Alicia envía un mensaje a su servidor de correo; y el mensaje es ubicado en la cola de mensajes.
3) El lado cliente del SMTP abre una conexión TCP con el servidor de correo de Roberto.
4) El cliente SMTP envía el mensaje de Alicia sobre la conexión TCP.
5) El servidor de correo de Roberto deposita el mensaje en el buzón de correo de Roberto.
6) Roberto recurre a su agente de usuario para leer el mensaje.



Ejemplo de interacción SMTP

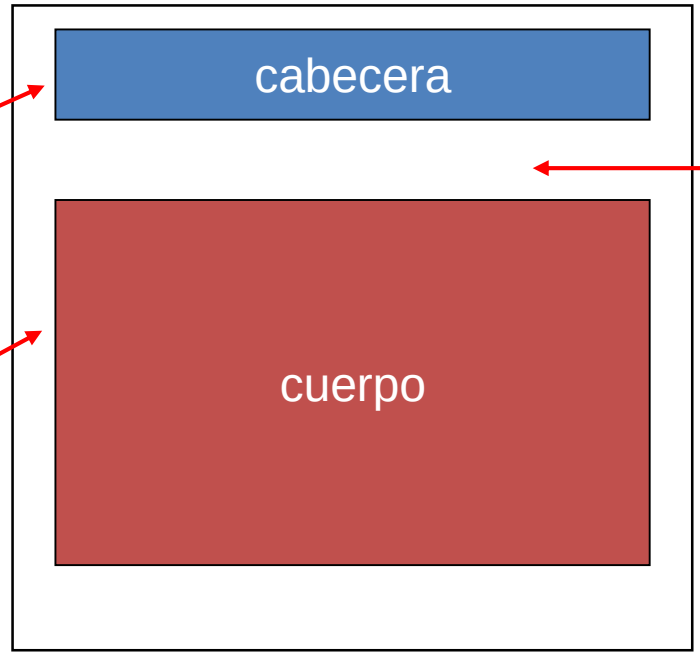
```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alicia@crepes.fr>
S: 250 alicia@crepes.fr... Sender ok
C: RCPT TO: <roberto@hamburger.edu>
S: 250 roberto@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: ¿Te gusta el ketchup?
C: ¿Y los encurtidos?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Formato de los mensajes de correo

SMTP: protocolo para intercambiar mensajes de correo electrónico.

RFC 822: estándar para el formato de texto del mensaje:

- Cabecera:
  - Para:
  - De:
  - Asunto:
  - Etc:
- Cuerpo:
  - el "mensaje", sólo caracteres ASCII.



RFC 822

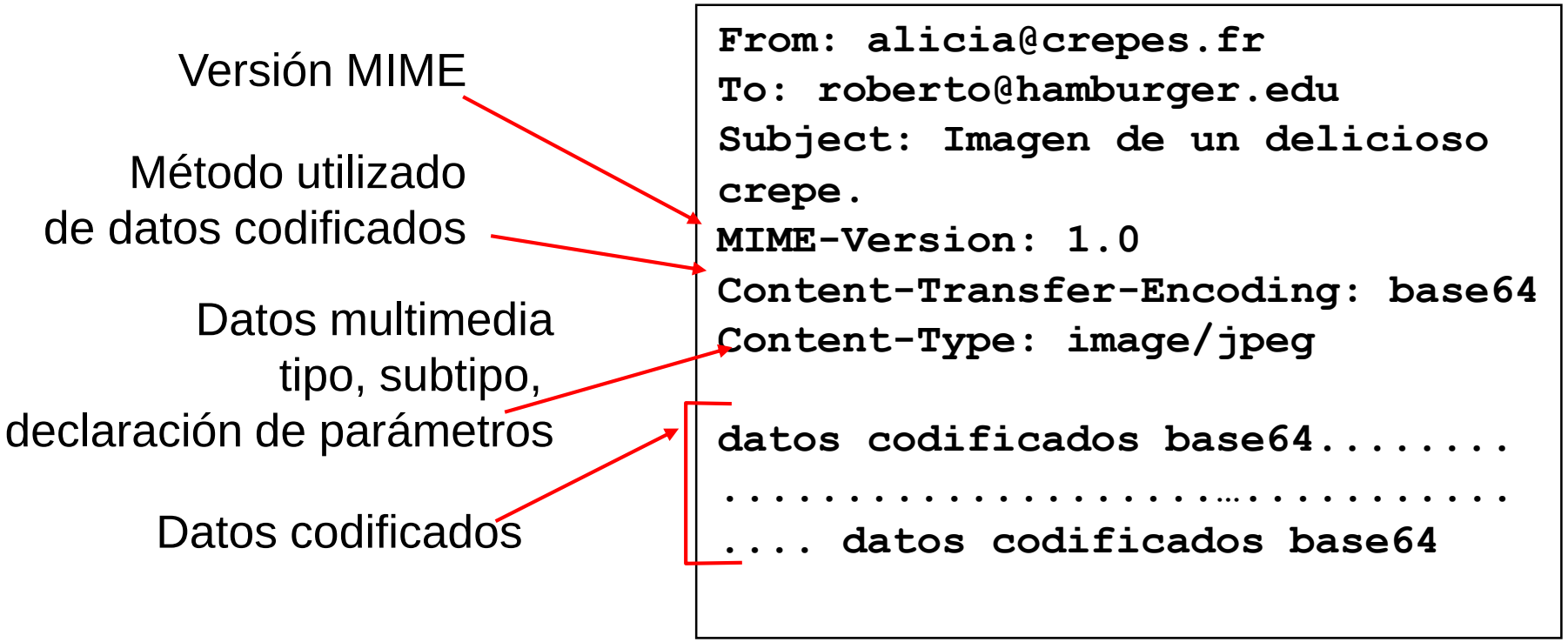
Encabezado	Significado
To:	Direcciones de correo electrónico de los destinatarios primarios
Cc:	Direcciones de correo electrónico de los destinatarios secundarios
Bcc:	Direcciones de correo electrónico para las copias ocultas
From:	Persona o personas que crearon el mensaje
Sender:	Dirección de correo electrónico del remitente
Received:	Línea agregada por cada agente de transferencia en la ruta
Return-Path:	Puede usarse para identificar una ruta de regreso al remitente

RFC 822

Encabezado	Significado
Date:	Fecha y hora de envío del mensaje
Reply-To:	Dirección de corre electrónico a la que deben enviarse las contestaciones
Message-Id:	Número único para referencia posterior a este mensaje
In-Reply-To:	Identificador del mensaje al que éste responde
References:	Otros identificadores de mensaje pertinentes
Keywords:	Claves seleccionadas por el usuario
Subject:	Resumen corto del mensaje para desplegar en una línea

MIME

- MIME: extensiones de correo multimedia, RFC 2045, 2056
- Las líneas adicionales en la cabecera del mensaje declaran el tipo de contenido MIME.



MIME : Extensiones Multipropósito de Correo Internet

Encabezado	Significado
MIME-Version:	Identifica la versión de MIME
Content-Description:	Cadena de texto que describe el contenido
Content-Id:	Identificador único
Content-Transfer-Encoding:	Cómo se envuelve el mensaje para su transmisión
Content-Type:	Naturaleza del mensaje

- R ıales
  ti an
  mediante una diagonal, como en
- Content-Type: video/mpeg

MIME : RFC 2045

Tipo	Subtipo	Descripción
Texto	Plano	Texto sin formato
	Enriquecido	Texto con comandos de formato sencillos
Imagen	Gif	Imagen fija en formato GIF
	Jpeg	Imagen fija en formato JPEG
Audio	Básico	Sonido
Vídeo	Mpeg	Película en formato MPEG
Aplicación	Octet-stream	Secuencia de bytes no interpretada
	Postscript	Documento imprimible en PostScript
Mensaje	Rfc822	Mensaje MIME RFC 822
	Parcial	Mensaje dividido para su transmisión
	Externo	El mensaje mismo debe obtenerse de la red
Multipartes	Mezclado	Partes independientes en el orden especificado
	Alternativa	Mismo mensaje en diferentes formatos
	Paralelo	Las partes deben verse en forma simultánea
	Compendio	Cada parte es un mensaje RFC 822 completo

POP3 - IMAP

Característica	POP3	IMAP
En dónde se define el protocolo	RFC 1939	RFC 2060
Puerto TCP utilizado	110	143
En dónde se almacena el correo electrónico	PC del usuario	Servidor
En dónde se lee el correo electrónico	Sin conexión	En línea
Tiempo de conexión requerido	Poco	Mucho
Uso de recursos del servidor	Mínimo	Amplio
Múltiples buzones	No	Sí
Quién respalda los buzones	Usuario	ISP
Bueno para los usuarios móviles	No	Sí
Control del usuario sobre la descarga	Poco	Mucho
Descargas parciales de mensajes	No	Sí
¿Es un problema el espacio en disco?	No	Con el tiempo podría serlo
Sencillo de implementar	Sí	No
Soporte amplio	Sí	En crecimiento

La Web y HTTP

En primer lugar, un poco de jerga

- Una página web consta de objectos.
- Un objeto puede ser un archivo HTML, una imagen JPEG,un applet Java, un archivo de audio, etc.
- Una página web está formada por un archivo HTML base, que incluye diversos objetos referenciados.
- Cada objeto es direccionable por un URL.
- Ejemplo de URL:

`www.escuela.edu/departamento/imagen.gif`

nombre de host | nombre de ruta

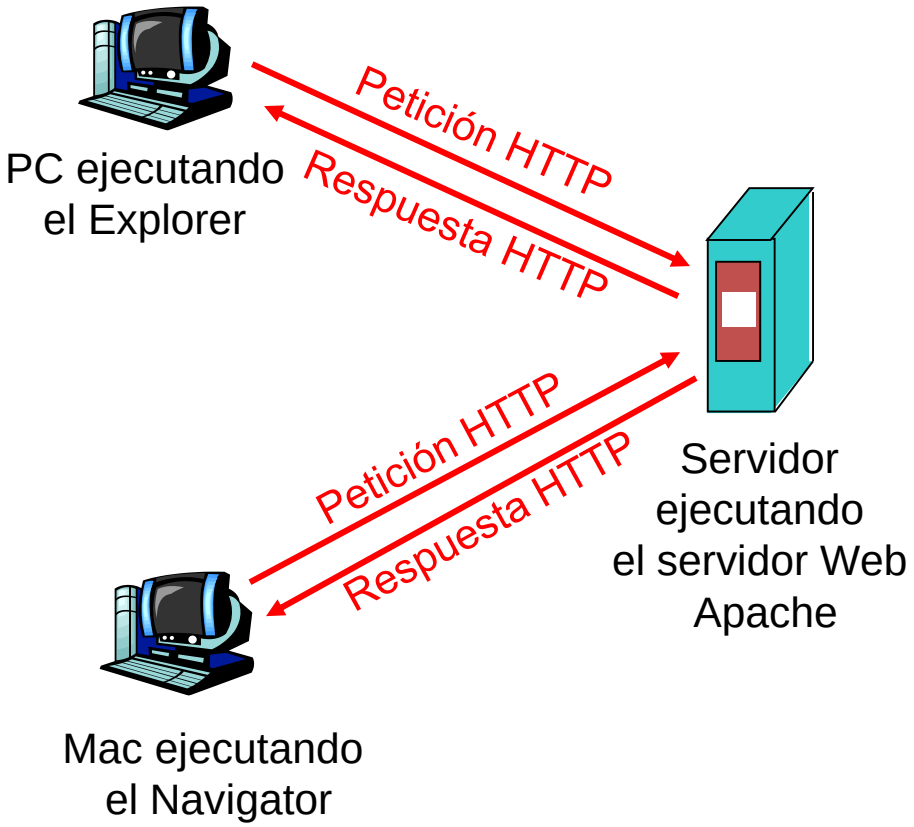
Nivel de Aplicación

HTTP: HyperText Transfer Protocol

Introducción a HTTP

HTTP: protocolo de transferencia de hipertexto:

- Protocolo de la capa de aplicación de la Web.
- Modelo cliente/servidor:
  - *cliente:* navegador que solicita, recibe y "descarga" objetos Web.
  - *servidor:* servidor Web que envía los objetos correspondientes en respuesta a las peticiones.
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068



Introducción a HTTP

Usos de TCP:

- El cliente inicia la conexión TCP (crea un socket) al servidor, puerto 80.
- El servidor accepta la conexión TCP del cliente.
- Los mensajes HTTP (mensajes de protocolo de la capa de aplicación) intercambiados entre el navegador (cliente HTTP) y el servidor Web (servidor HTTP)
- La conexión TCP se cierra.

HTTP está "sin estado"

- El servidor no conserva ninguna información sobre las peticiones previas de clientes.

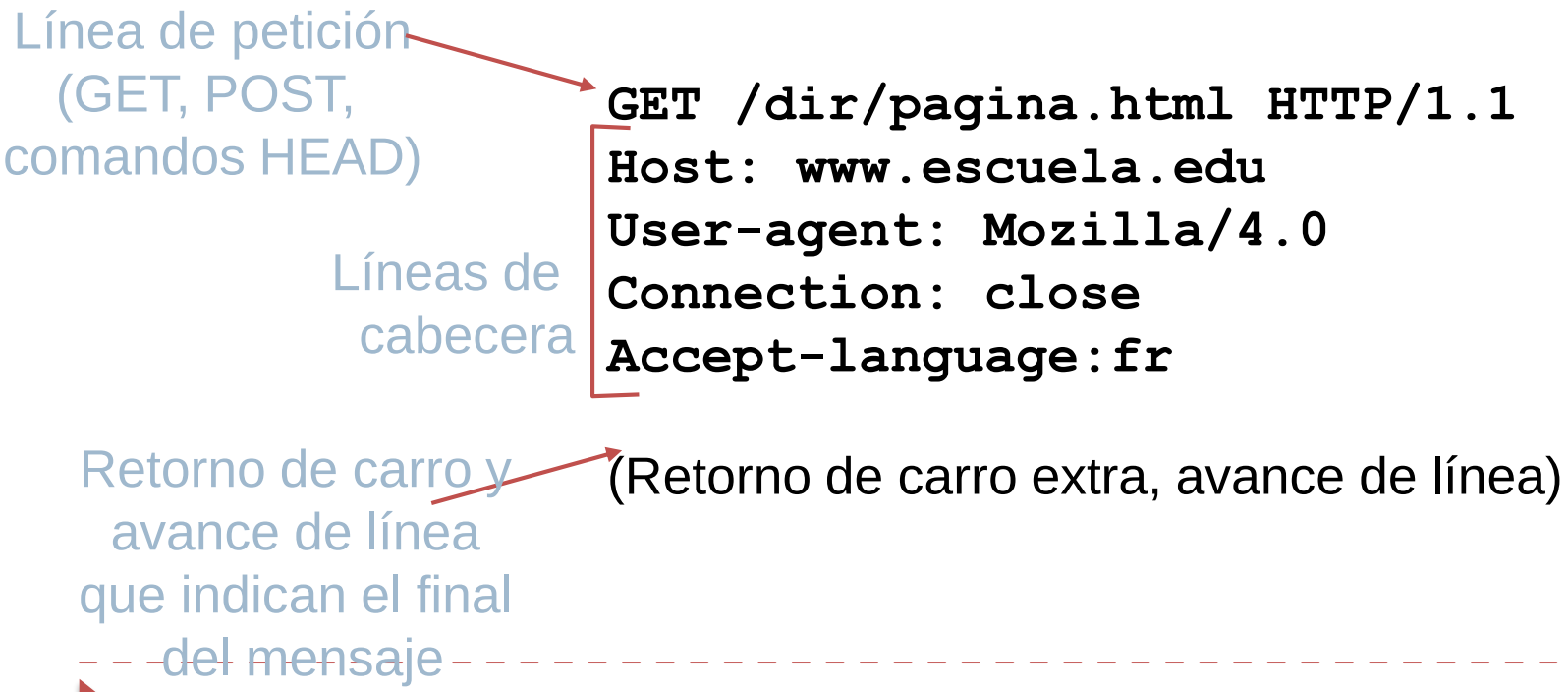
aparte

Los protocolos que conservan "estado" son complicados

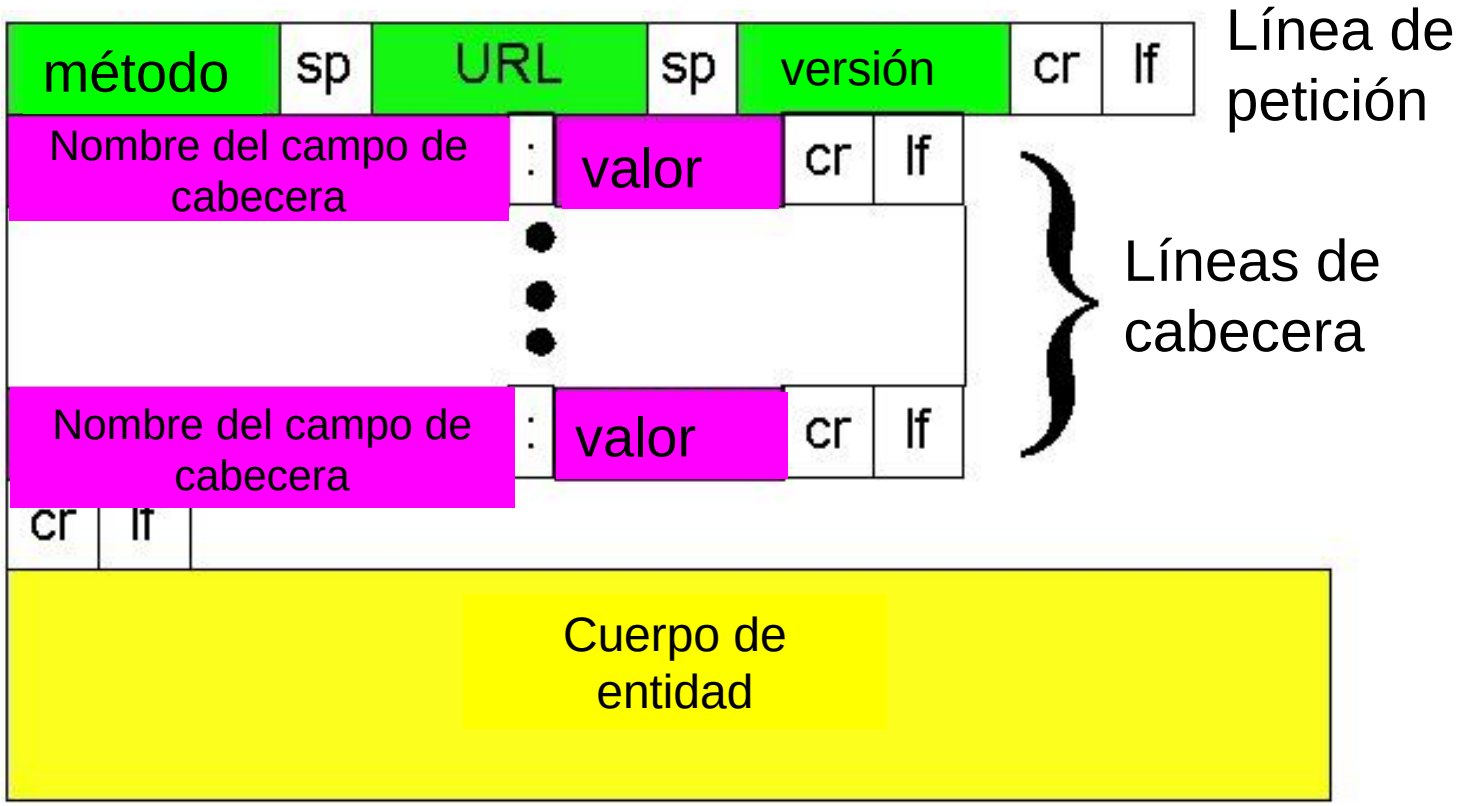
- La historia previa (estado) debe conservarse.
- Si el servidor/cliente se bloquea, sus visiones del "estado" pueden ser inconsistentes y deben ser recompuestas.

Mensaje HTTP de petición

- Hay dos tipos de mensajes HTTP: *de petición*, de respuesta.
- Mensaje HTTP de petición:
  - ASCII (formato leído por personas)



Mensaje HTTP de petición: formato general



Tipos de mensajes/métodos

HTTP/1.0

- GET.
- POST.
- HEAD.
  - Pide al servidor que excluya el objeto solicitado de la respuesta.

HTTP/1.1

- GET, POST, HEAD.
- PUT:
  - Descarga el archivo en el cuerpo de entidad a la ruta especificada en el campo URL.
- DELETE:
  - Borra el archivo especificado en el campo URL.

Mensaje HTTP de respuesta

Línea de estatus (protocolo del código de estatus y la frase de estatus)

Líneas de cabecera

Datos, por ejemplo, el archivo HTML solicitado

```
HTTP/1.1 200 OK
Connection close
Date: Thu, 06 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 1998 …...
Content-Length: 6821
Content-Type: text/html

datos datos datos datos datos ...
```

Código de estatus HTTP de respuesta

En la primera línea en el mensaje de respuesta servidor -> cliente.
Algunos ejemplos de códigos:

200 OK

- petición exitosa, el objeto requerido aparece posteriormente en este mensaje.

301 Moved Permanently

- el objeto demandado ha sido movido, su nueva localización se especifica posteriormente en este mensaje (Location:).

400 Bad Request

- el servidor no comprendió el mensaje de petición.

404 Not Found

- el documento pedido no existe en este servidor.

505 HTTP Version Not Supported

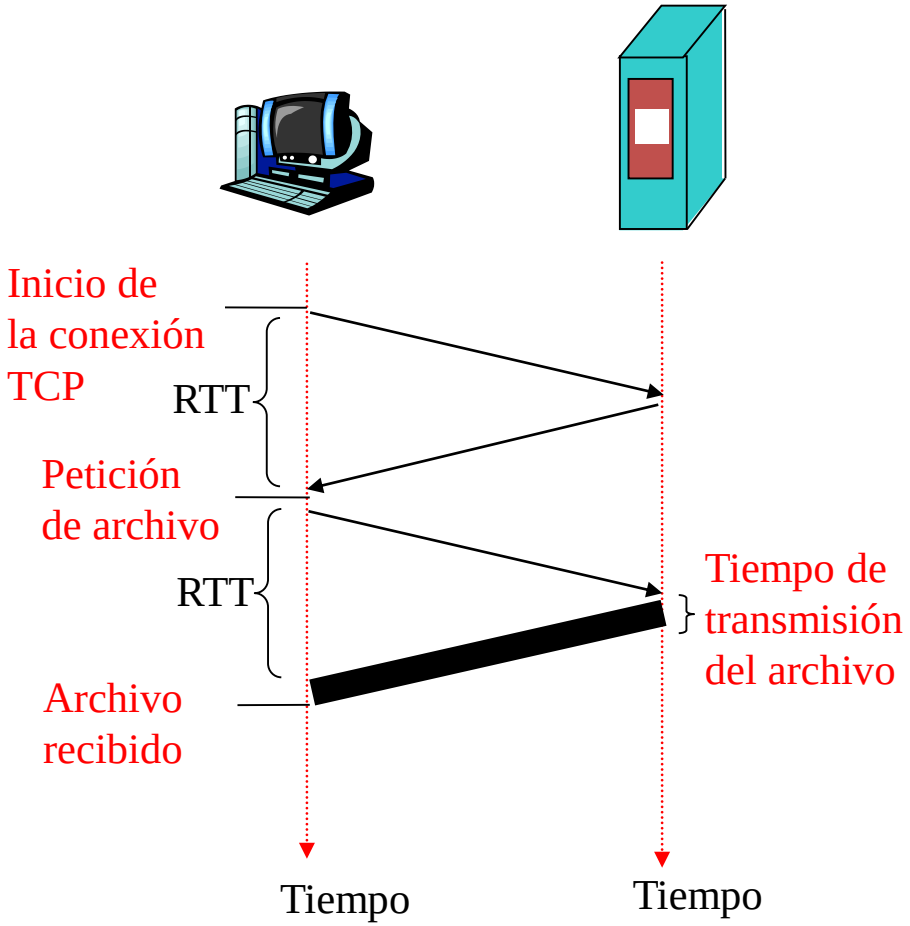
Modelo del tiempo de respuesta

Definición de RRT: tiempo necesario para enviar un paquete pequeño desde el cliente hasta el servidor y después de vuelta al cliente.

Tiempo de respuesta:

- Un RTT para iniciar la conexión TCP.
- Un RTT para la petición HTTP y los primeros bytes de respuesta HTTP de vuelta.
- Tiempo de transmisión del archivo:

total = 2RTT+tiempo de transmisión



Conexiones HTTP

Conexiones HTTP no persistentes

- Se envía un objeto como máximo con una conexión TCP.
- HTTP/1.0 utiliza conexiones HTTP no persistentes.

Conexiones HTTP persistentes

- Se pueden enviar múltiples objetos con una sola conexión TCP entre el cliente y el servidor.
- HTTP/1.1 utiliza conexiones persistentes en su modo por defecto.

Conexiones HTTP no persistentes

Supongamos que el usuario entra en el URL:
`www.escuela.edu/departamento/home.index`

(consta de texto y referencias a 10 imágenes jpeg)

1a. El cliente HTTP inicia la conexión TCP con el servidor HTTP (proceso) de www.escuela.edu en el puerto 80.

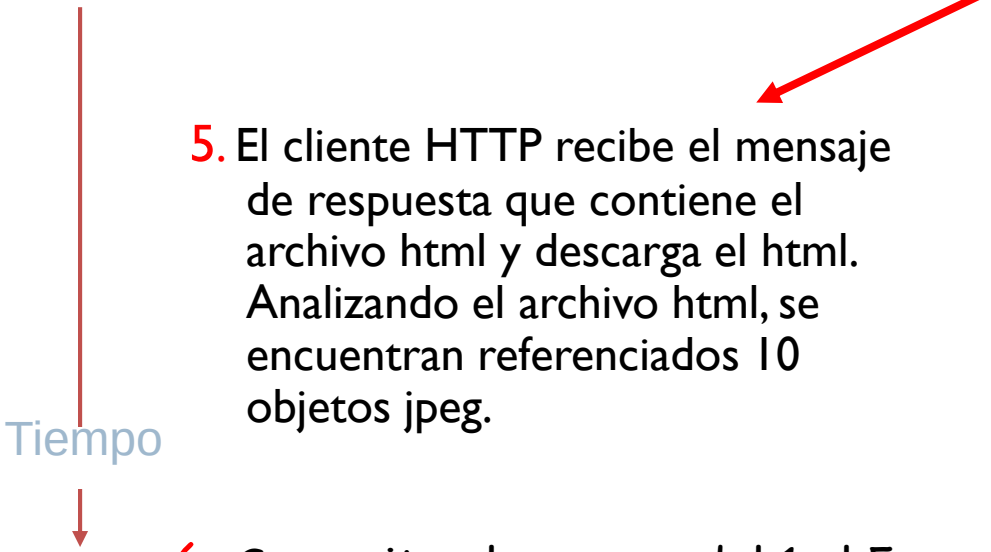
1b. El servidor HTTP en el host www.escuela.edu espera la conexión TCP del puerto 80 y "acepta" la conexión, notificando al cliente.

2. El cliente HTTP envía un *mensaje HTTP de petición* (que contiene el URL) a través del socket de la conexión TCP. El mensaje indica que el cliente quiere el objeto departamento/home.index.

3. El servidor HTTP recibe el mensaje de petición, compone un *mensaje de respuesta* que contiene el objeto solicitado, y lo envía al socket.

Tiempo

Conexiones HTTP no persistentes



4. El servidor HTTP cierra la conexión TCP.

5. El cliente HTTP recibe el mensaje de respuesta que contiene el archivo html y descarga el html. Analizando el archivo html, se encuentran referenciados 10 objetos jpeg.

6. Se repiten los pasos del 1 al 5 para cada uno de los 10 objetos jpeg.

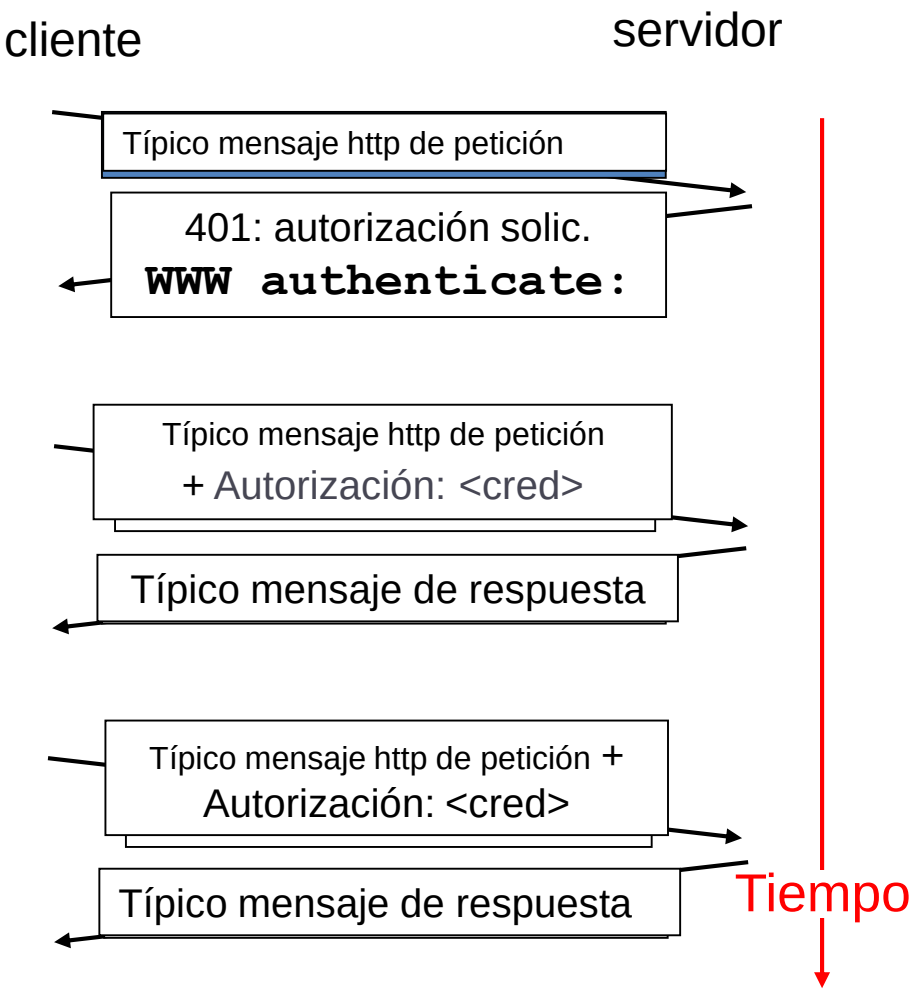
Interacción usuario-servidor: autorización

Autorización : control al acceso del contenido del servidor.

- Credenciales de autorización, normalmente son: nombre, contraseña.
- Sin estado: el cliente debe presentar la autorización para *cada* petición:
  - Autorización: línea de cabecera en cada petición.
  - Si no hay autorización: cabecera, el servidor rechaza el acceso y envía

`WWW authenticate:`

línea de cabecera en respuesta.



Cookies: mantenimiento del "estado"

Muchos sitios web importantes utilizan cookies.

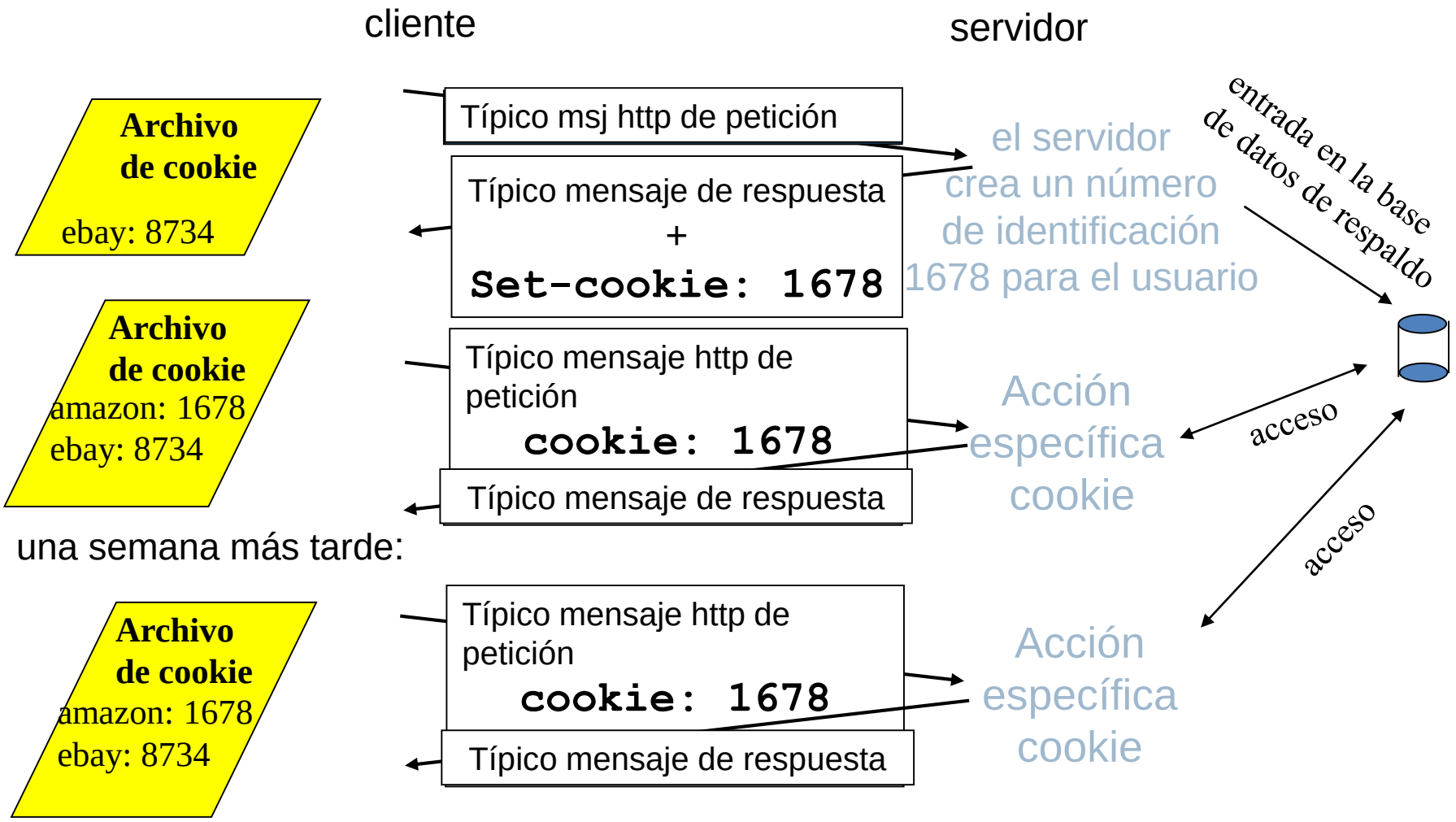
Cuatro componentes:

1) Línea de cabecera de cookie en el mensaje HTTP de respuesta.
2) Línea de cabecera de cookie en el mensaje HTTP de petición.
3) Archivo de cookie que se almacena en el host del usuario y que es gestionado por el navegador del usuario.
4) Base de datos de respaldo en el sitio Web.

Ejemplo:

- Susana siempre accede a Internet desde el mismo PC.
- Visita un sitio de comercio electrónico por primera vez.
- Cuando la petición HTTP inicial llega al sitio, éste crea un número de identificación único y también crea en su base de datos una entrada indexada por el número de identificación.

Cookies: mantenimiento del “estado”



Nivel de Aplicación

Peer to Peer: BitTorrent

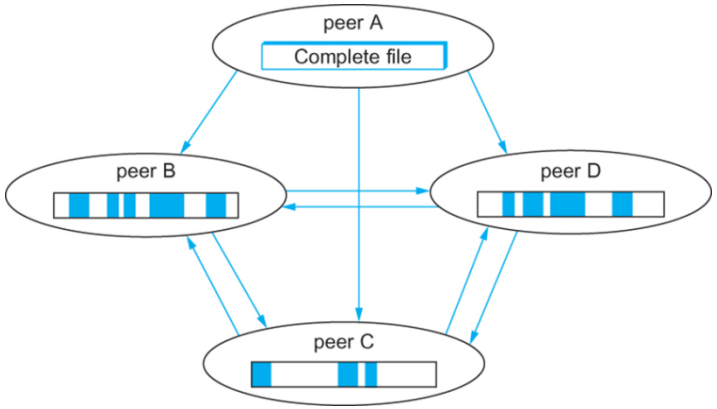
BitTorrent

- BitTorrent [*] es un protocolo para distribuir archivos en Internet en el cual los nodos que bajan el archivo contribuyen al mismo tiempo a distribuirlo, subiendo partes del archivo a otros nodos que lo están bajando
- De esta manera es posible que un nodo con poco " ancho de banda" distribuya un archivo a muchos otros, sin sufrir un incremento lineal en la demanda a medida que crece el número de nodos bajando el archivo
- El conjunto de peers bajando el archivo se denomina *swarm*
-
-
-
- [*] B. Cohen, Incentives Build Robustness in BitTorrent," 2003.

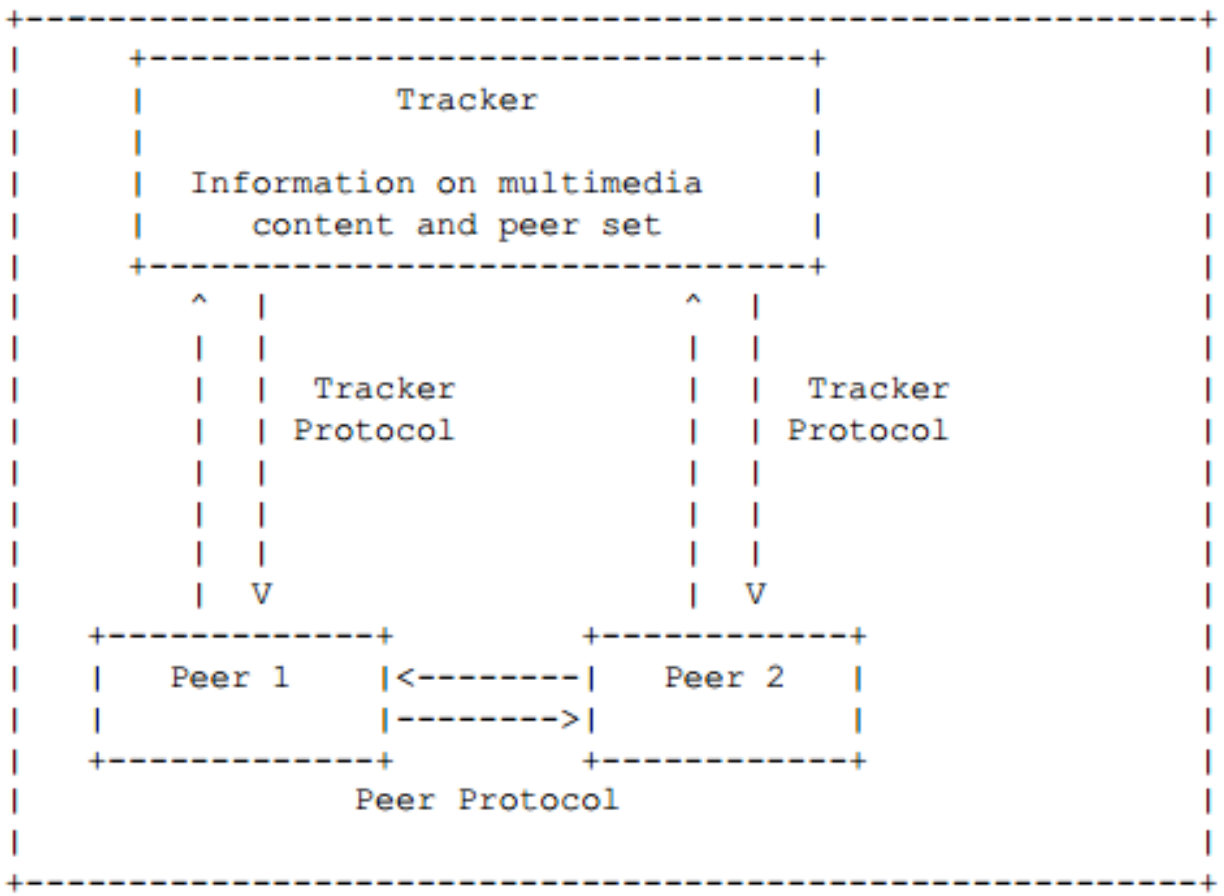
BitTorrent

- El archivo a bajar (replicar!) se divide en piezas (pieces) de tamano fijo para facilitar su intercambio entre los peers
- Una vez que un peer termino de replicar una pieza entera, ya puede empezar a ofrecerla a los demas peers

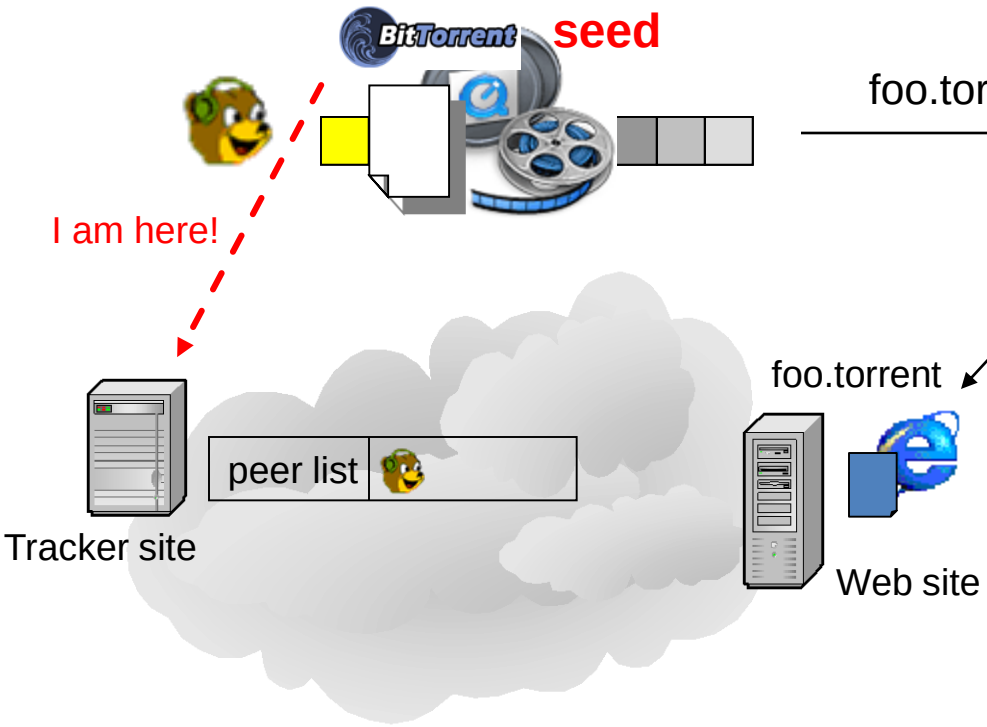
- Asi es como las piezas y el archivo se replican una vez bajadas del nodo que tenia el archivo originalmente, que es el primer seed.



Modelo de alto nivel



Publicación

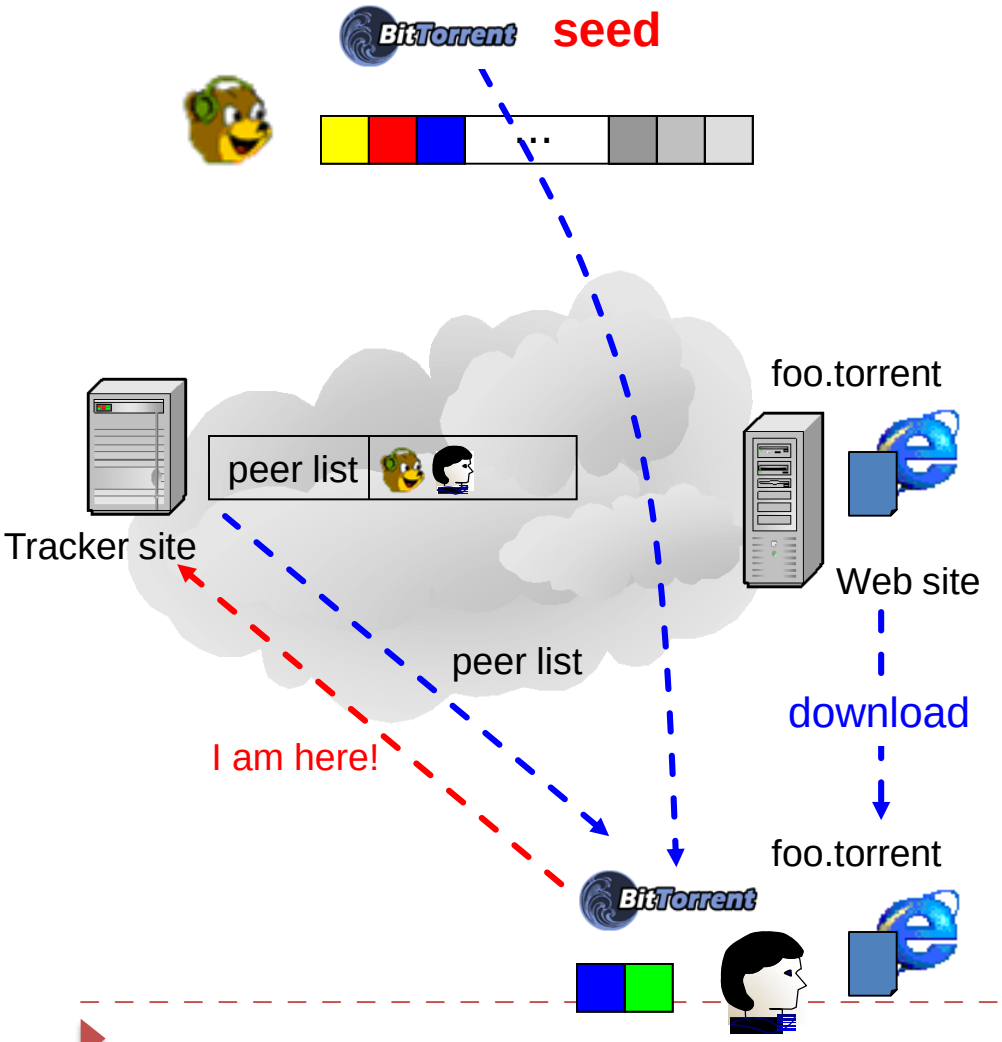


announce: tracker URL for bootstrap
creation date: epoch time of file creation
length: file size
name: file name
piece length: chunk size
pieces: SHA1 hash key of each chunk

quien publica

- Crea un meta file
- Publica en Web site
- Start tracker site
- Start BT client como seed inicial

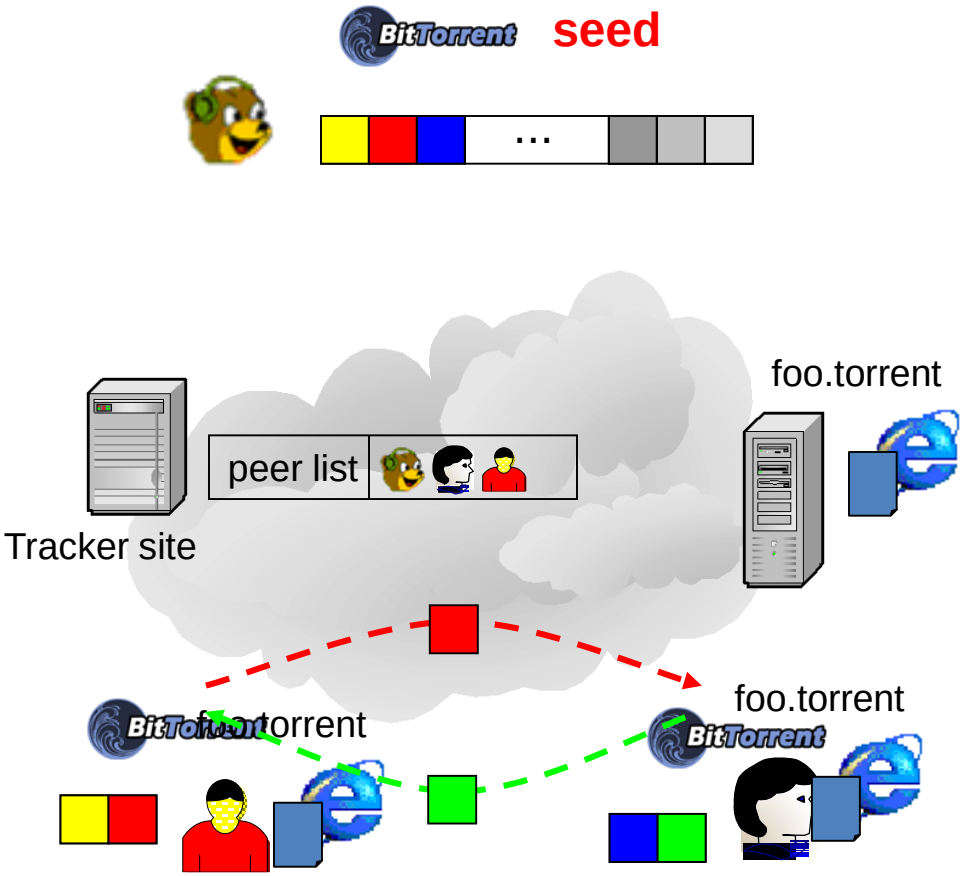
Downloading



El consumidor " downloader"

- Baja meta file
- Start BT client, se connect al tracker site
- "Get peer list" del tracker
- "Get first chunk" de otros peers (seeds)

Downloading



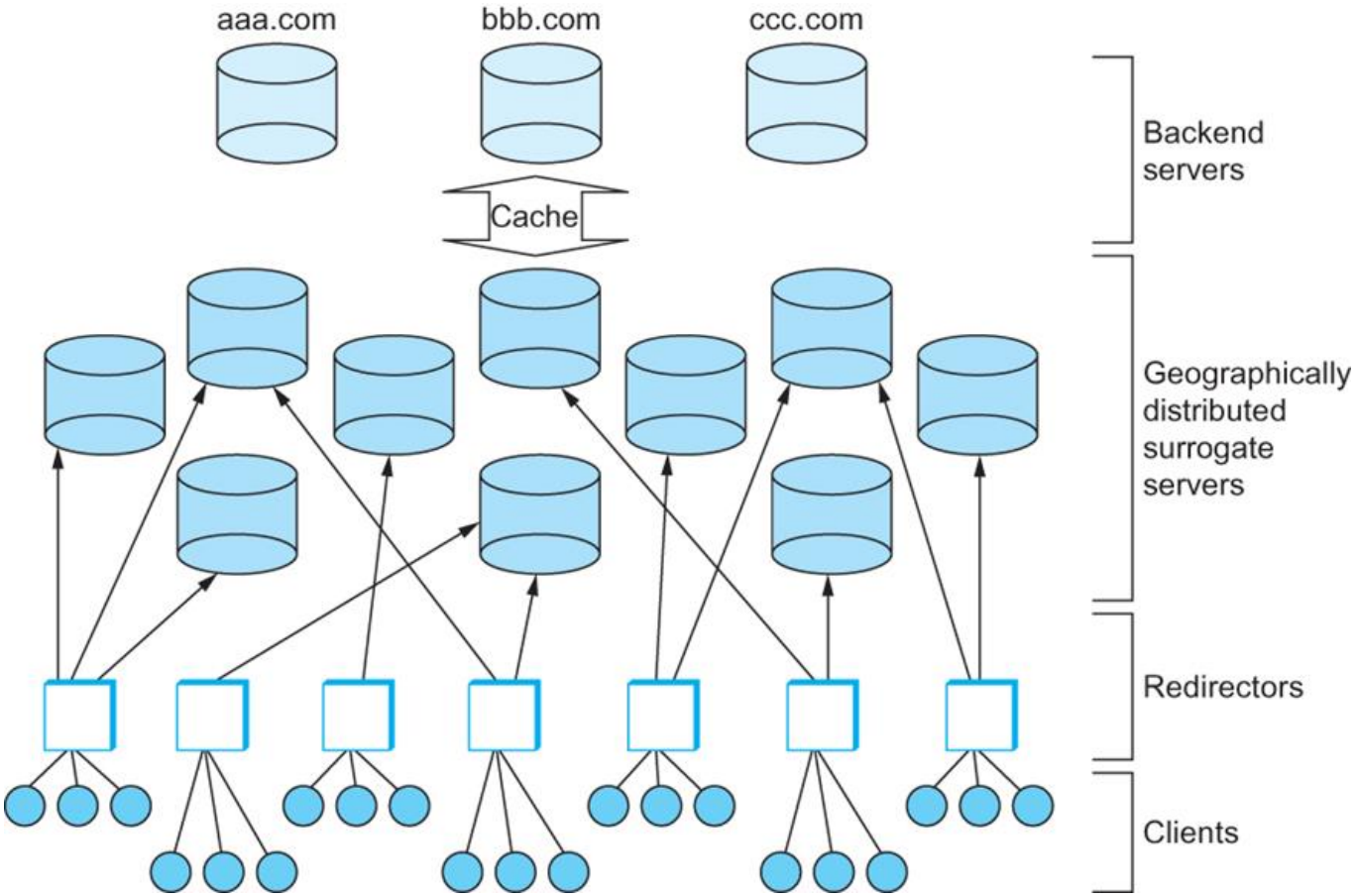
El consumidor " downloader"

- Baja meta file
- Start BT client, se connect al tracker site
- "Get peer list" del tracker
- "Get first chunk" de otros peers (seeds)
- intercambio file chunk with con otros peers
- Download completo: un nuevo seed

CDN

Content Distribution Networks

Modelo genérico de una CDN

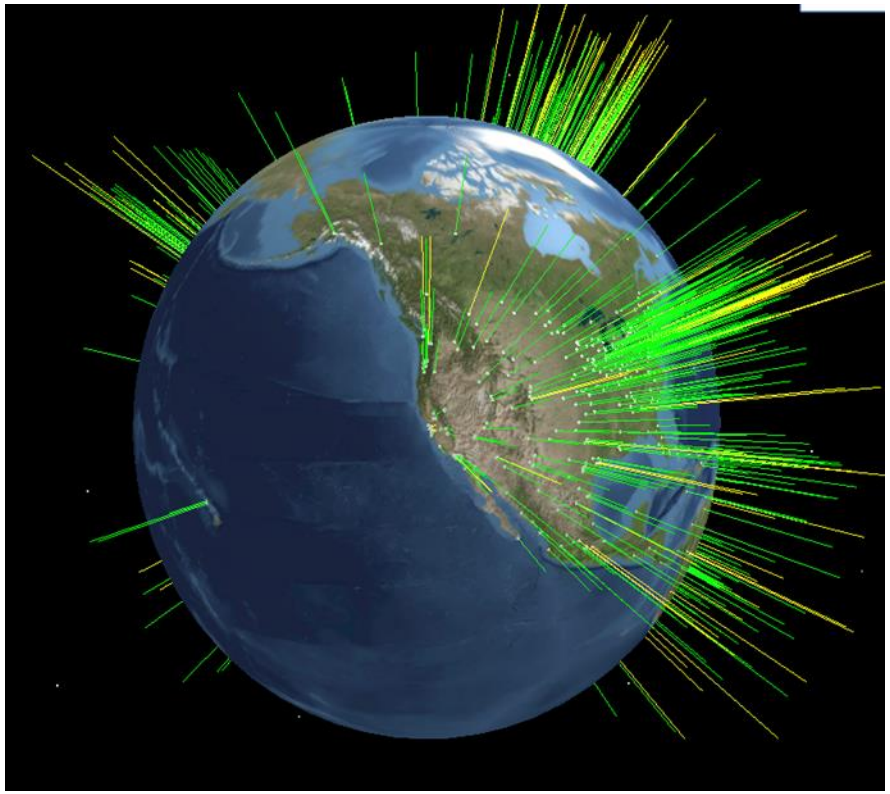


CDN

[10]Las CDNs, como su nombre lo indica, son redes de distribución de contenido. Funcionan como nexo entre los proveedores, por ejemplo los sitios Web de canales de televisión, y los consumidores en los ISPs. Colocando servidores propios en los ISPs, servidores que cachean o almacenan enteramente el contenido, las CDNs se aseguran de que los clientes lo accedan de manera local. Esto es especialmente importante para la distribución de video, en la cual una latencia razonable es importante para que el video no se vea entrecortado.

La distribución de usuarios a servidores se realiza mediante redirecciones DNS. Cuando un usuario solicita contenido almacenado en una CDN, la dirección IP que recibe para una URL dada se resuelve teniendo en cuenta su localización geográfica y en la red. De esta manera, accederá a la copia del contenido presente en el servidor más cercano (geográficamente, o en términos de latencia) a él.

Una CDN comercial : Akamai



http://wwwnui.akamai.com/gnet/globe/index.html

Amazon

Netflix

Quienes tienen su propia CDN

Google

Entre otros ….

Bibliografía Básica

- Computer Networks, Fifth Edition: A Systems Approach (The Morgan Kaufmann Series in Networking) Larry L. Peterson , Bruce S. Davie 2011
- Capitulo 9 Nivel de Aplicación : Mail - Web págs. 700-718 y DNS págs. 745-755
  - Se explican arquitecturas, fundamentos en pizzarrón + slides de la clase "práctica
- Capitulo 9 Nivel de Aplicación : Overlays Networks págs. 759-789