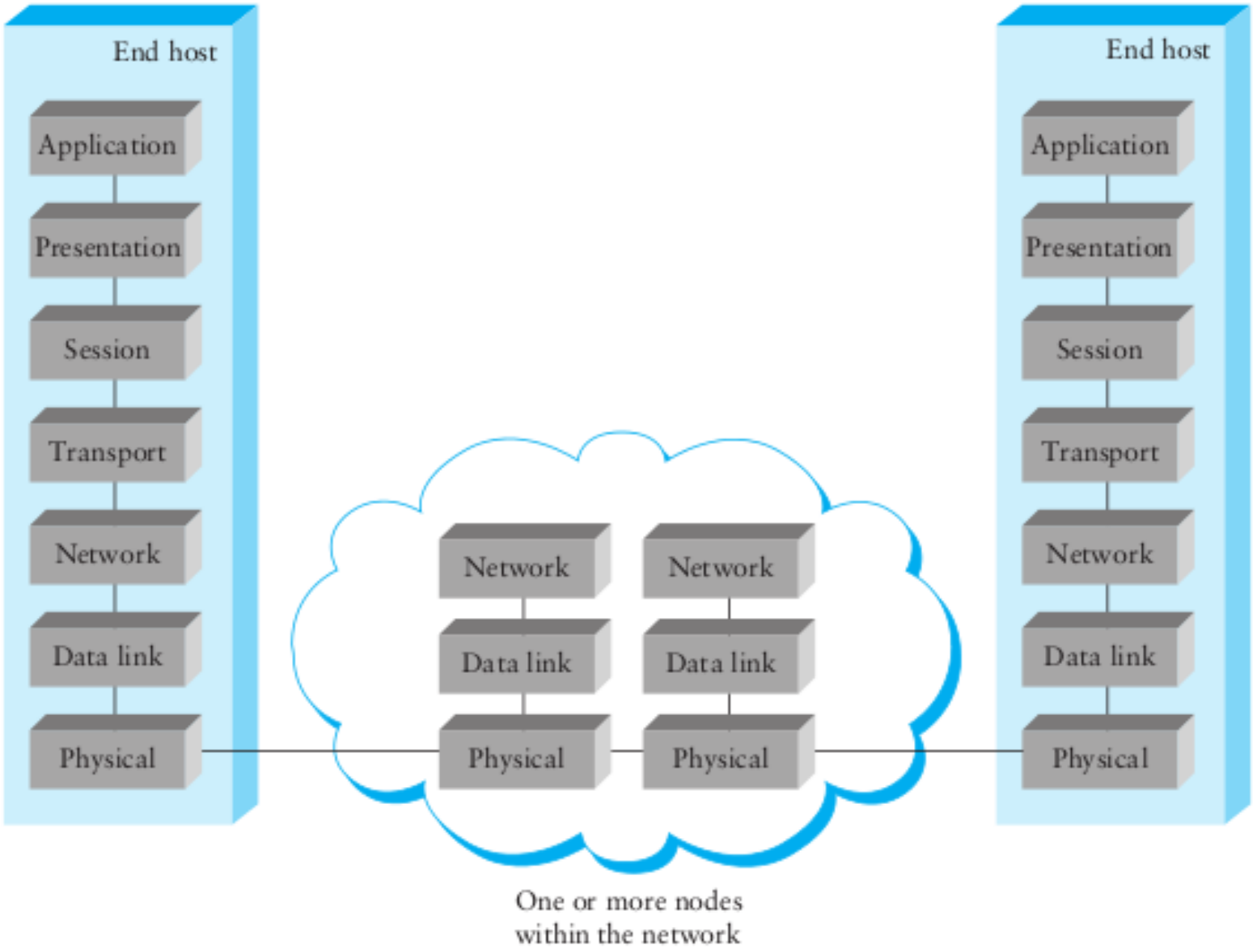
Capa de Aplicación

DC - FCEyN - UBA

Primer cuatrimestre 2018

Arquitectura en capas



Menú del día

Capa de Aplicación

- MIME, HTTP, SMTP,...
- Otras Aplicaciones

Utilización de protocolos existentes

Específicos de la capa de aplicación

- Aplicaciones son software (procesos)
- Tipo de servicio de las capas inferiores.
- Arquitecturas de comunicación: Paradigma cliente - servidor, Paradigma P2P, Híbridos, RPC
- Syntaxis y semántica de los mensajes.

Arquitectura de las aplicaciones tradicionales

De tipo Request / Reply de mensajes entre cliente y servidor.

Aplicación versus protocolo

Ejemplo: HTTP (Protocolo) $\neq$ Firefox (Aplicación).

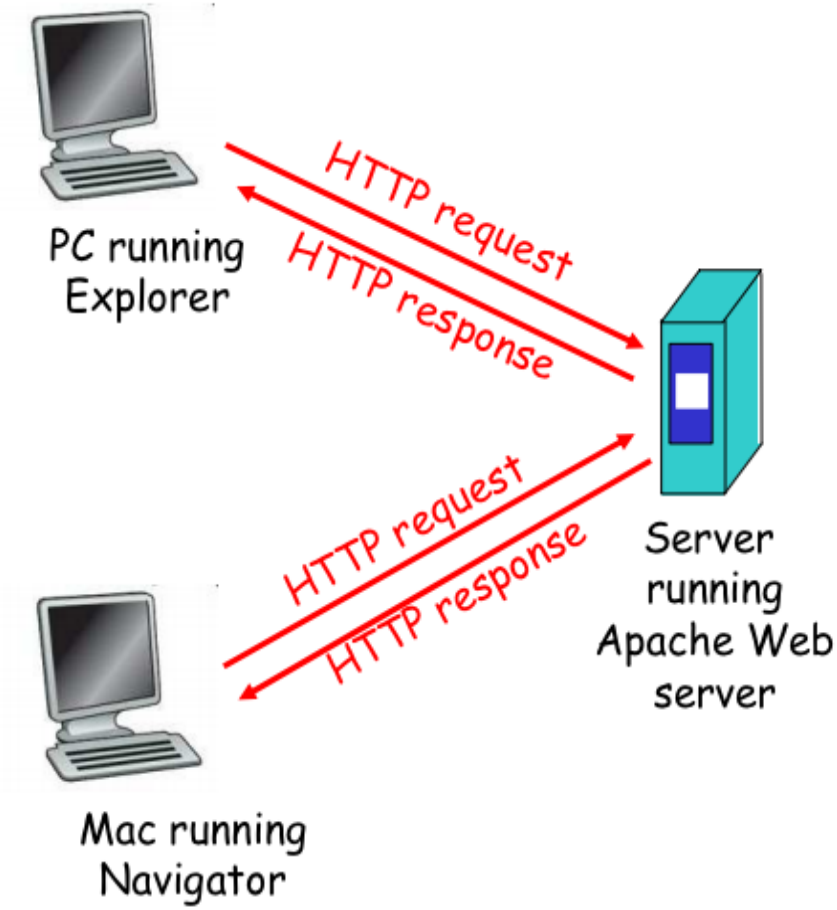
Conceptos fundamentales

Generales

- Usa TCP (puerto 80).
- Browser y servidor se intercambian mensajes
- HTTP $\Rightarrow$ HEADER
- HTML $\Rightarrow$ BODY.
- Cada objeto se referencia por un identificador único (URL).
- HTTP **no** mantiene estado.

HTTP: hypertext transfer protocol.

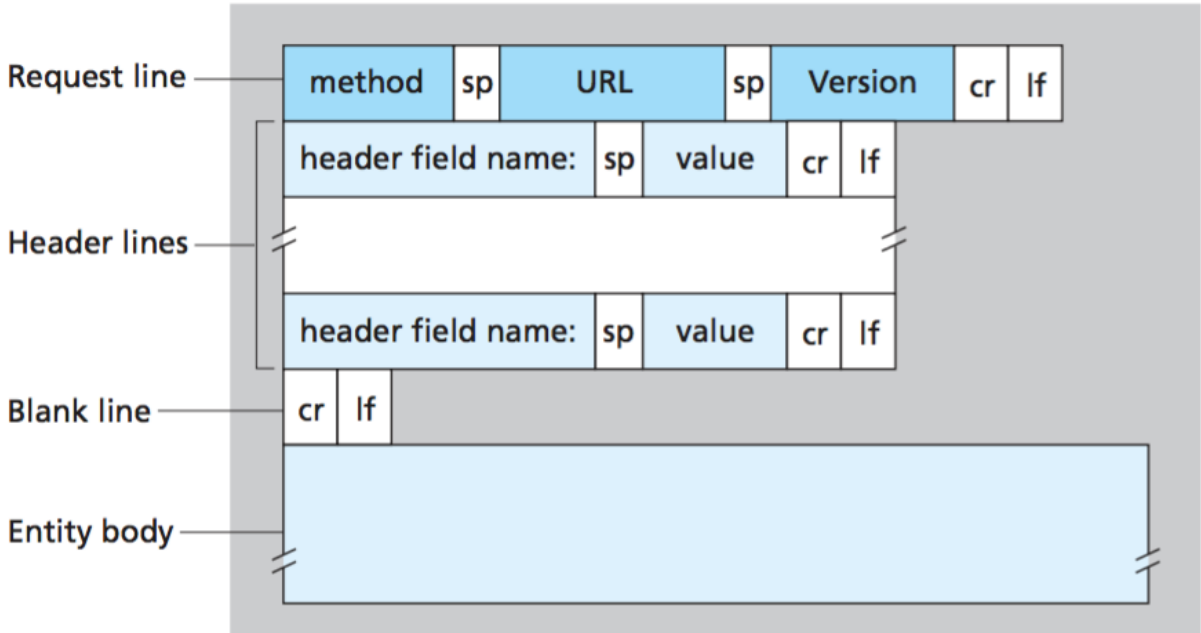
- Protocolo de la capa aplicación de la Web.
- Modelo cliente/servidor
  - *cliente:* browser que requiere, recibe, "despliega" objetos Web.
  - *servidor:* Servidor Web envía objetos en respuesta a requerimientos.
- HTTP 1.0: RFC 1945.
- HTTP 1.1: RFC 2616.



Petición HTTP

Tipos

- GET: Solicitar información
- HEAD: Solicitar encabezados
- POST: Enviar datos para crear recurso
- PUT: Enviar datos para actualizar/reemplazar recurso
- DELETE: Solicitar eliminación de recurso



Petición HTTP

Línea de requerimiento (comandos GET, POST, HEAD)

Líneas de encabezado

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
User-agent: Mozilla/4.0
Connection: close
Accept-language:es
```

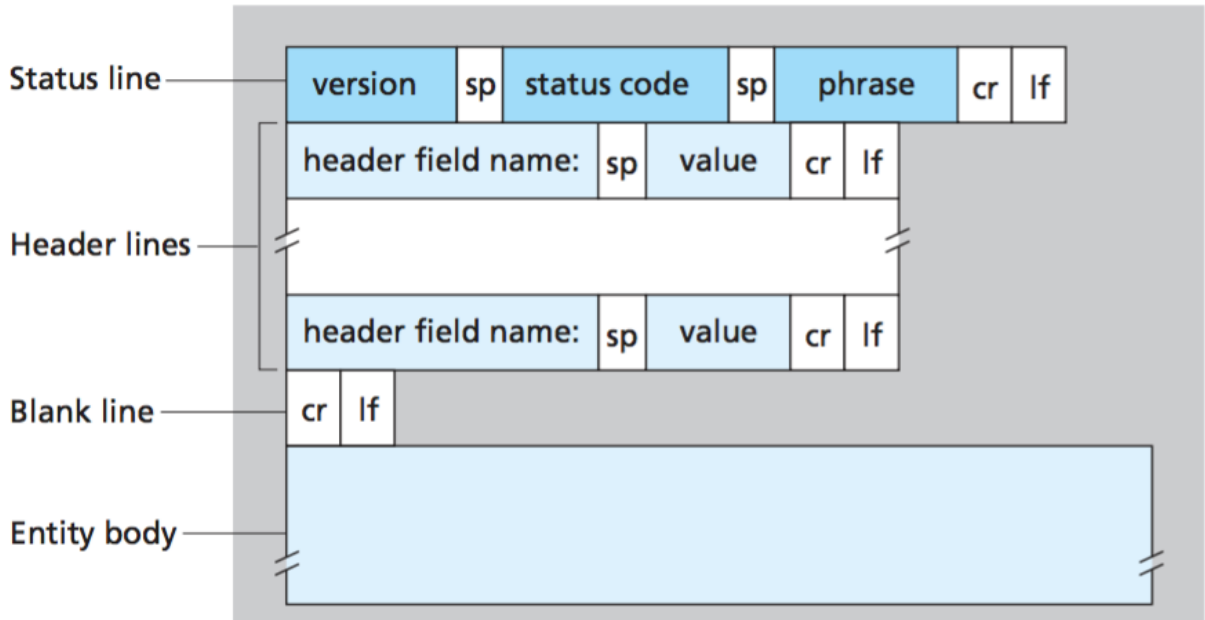
Carriage return, line feed
Indica fin de mensaje

(carriage return, line feed extra)

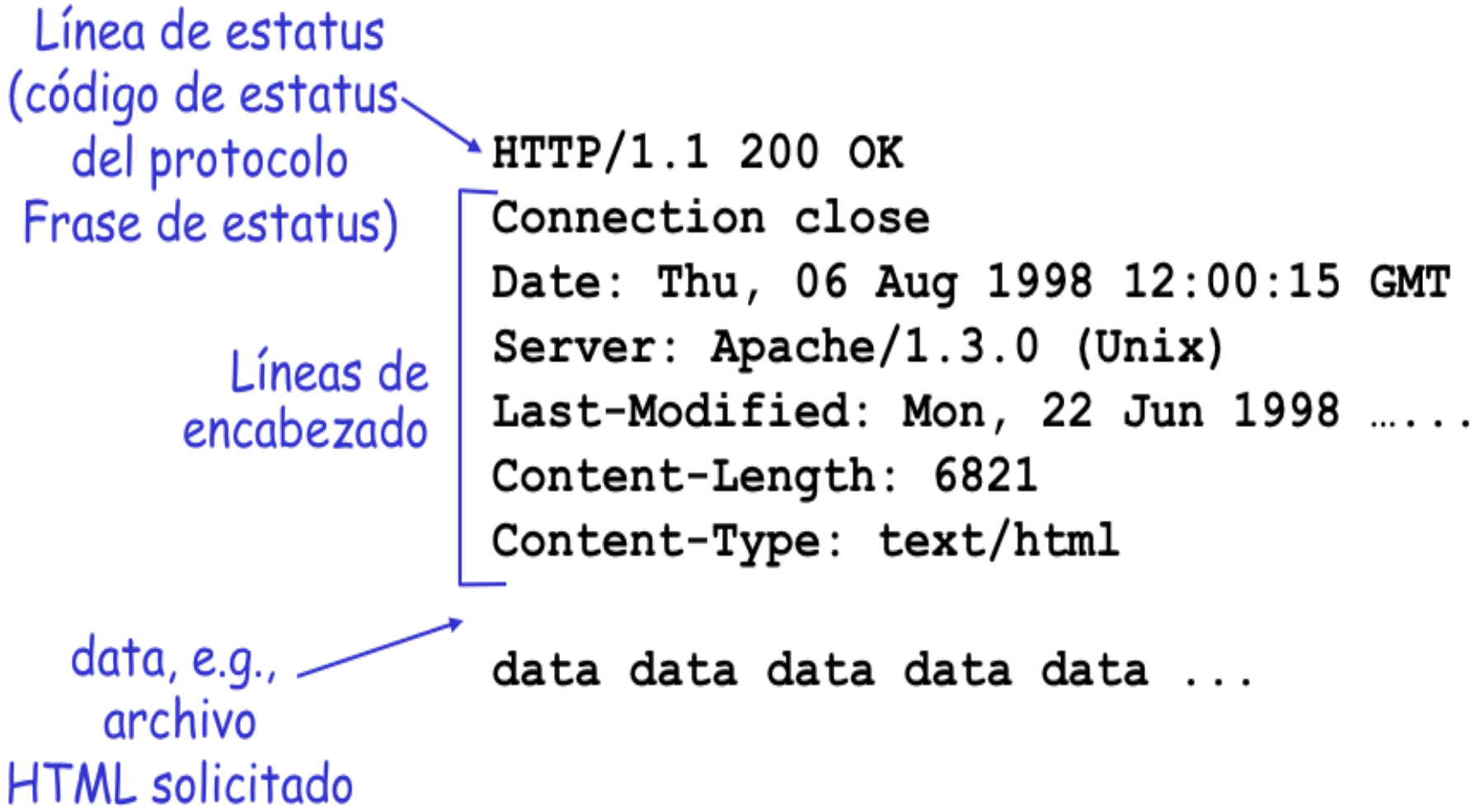
Respuesta HTTP

Tipos

- 1xx. Informativo
- 2xx. Exitoso
- 3xx. Redireccion
- 4xx. Error del Cliente
- 5xx. Error del Servidor



Respuesta HTTP



Encabezados HTTP Esenciales

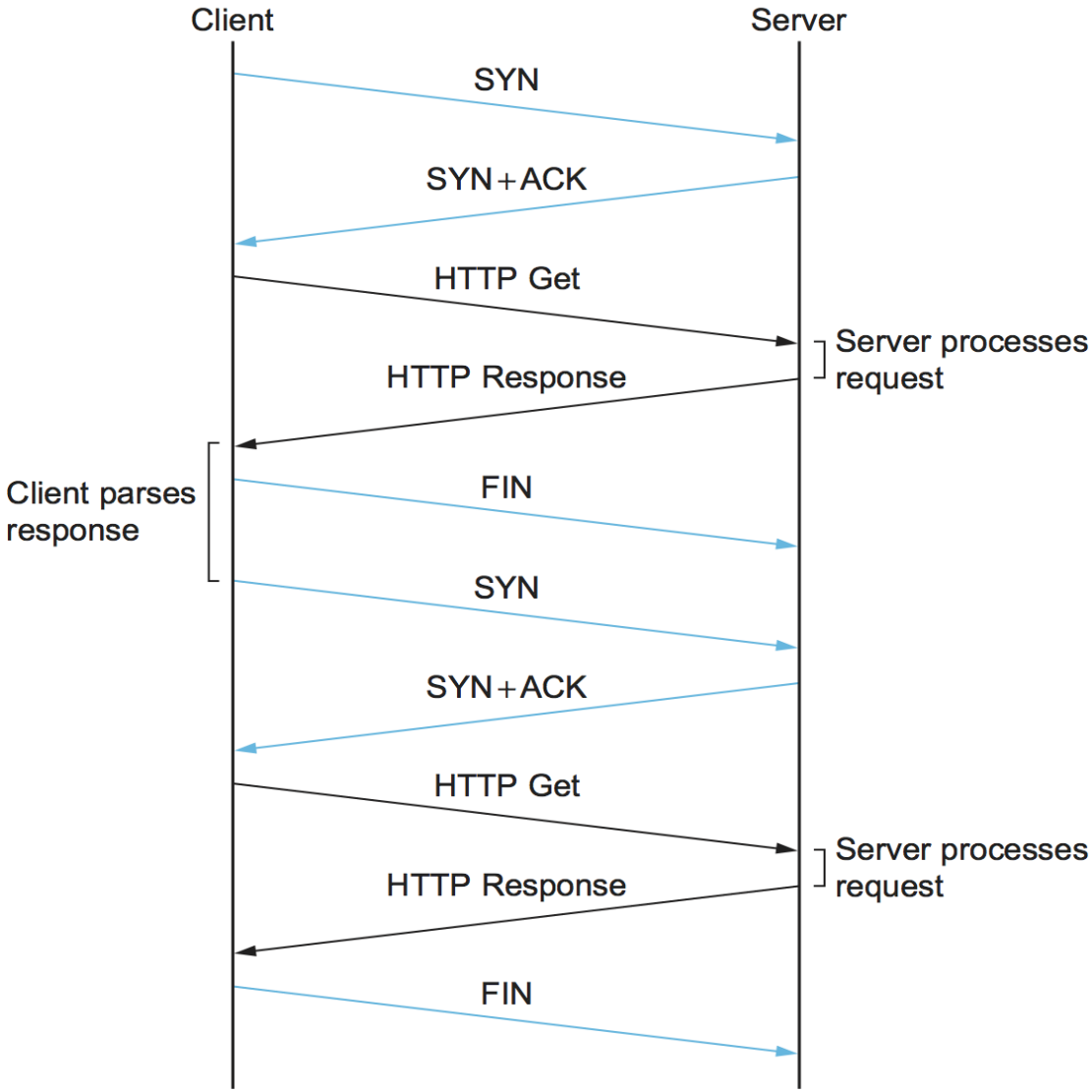
Encabezado	Petición/Respuesta	Contenido
`Host`	Petición	Nombre de dominio del servidor solicitado.
`Content-Type`	Ambos	Tipo de medio (ej. HTML, JSON, Imagen) de los datos.
`Content-Length`	Ambos	Tamaño del cuerpo del mensaje en bytes.
`Cache-Control`	Ambos	Directivas para el control de la caché.
`If-Modified-Since`	Petición	Se solicita el recurso si fue modificado despues de la fecha.
`Authorization`	Petición	Credenciales de autenticación del cliente.
`Cookie`	Petición	Envía cookies almacenadas del cliente al servidor.
`User-Agent`	Petición	Identifica el software del agente de usuario.

El protocolo HTTP permite hacer distintos tipos de pedidos para recursos de un determinado dominio. Escriba los Requests HTTP 1.1 que permitan obtener los siguientes pedidos al sitio web del departamento de computación:

1. El recurso /
2. Encabezado del recurso `/tdc`
3. El recurso `/logo.jpg` si no fue modificado desde una determinada fecha.
4. Crear un nuevo post en el foro del departamento

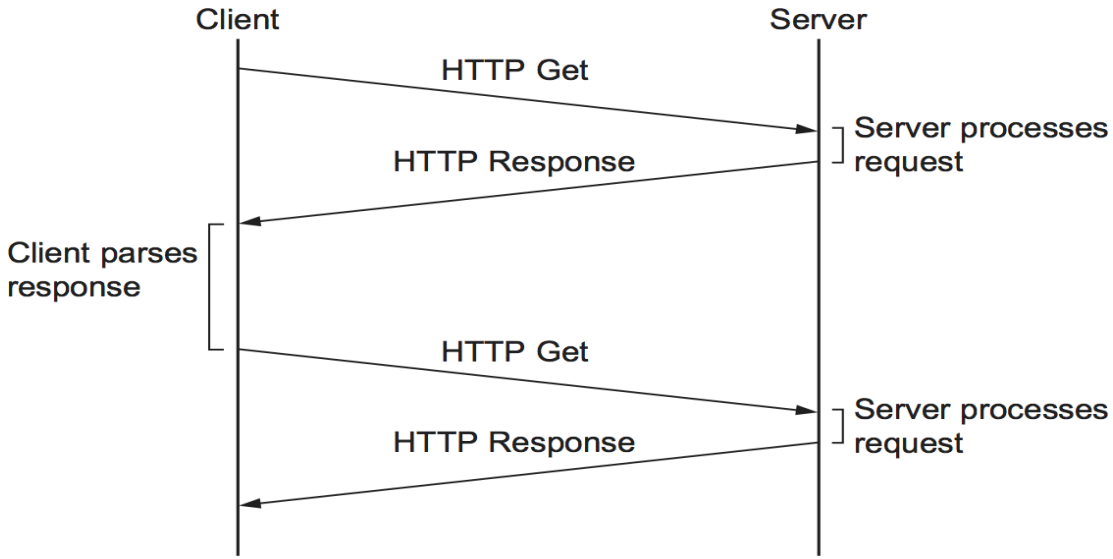
Versiones

- HTTP 1.0 es NO persistente. Requiere una conexion TCP por cada recurso



Versiones

- HTTP 1.1 lo es.
  - Muchos objetos pueden ser transmitidos por una única conexión TCP.
  - Permite pipelinening
  - Tiene el problema conocido como head-of-line blocking ya que HTTP/1.1 exige que las respuestas del servidor se envíen en el mismo orden en que se recibieron las solicitudes.



HTTP persistente

¿Por qué?

- $2 * \text{RTT} * \text{Objeto}$.
- Muchas conexiones, sobrecarga OS.
- Congestion Window.

Pipelining

- Espera completar los pedidos antes de envíar nuevos requests.
- Con PL, envía request apenas encuentra referencias (default en 1.1).

Etiquetas HTML Relevantes

Algunas etiquetas HTML provocan que el navegador realice solicitudes HTTP para obtener recursos:

- `<script>`: Carga scripts externos.
- `<link rel="stylesheet»`: Carga hojas de estilo CSS externas.
- `<form>` (al enviar): Envía datos al servidor.
- `<a>` Navega a otra URL **al hacer clic**.
- `<img>` `<audio>`, `<video>`: Carga archivos multimedia (atributo `src`).

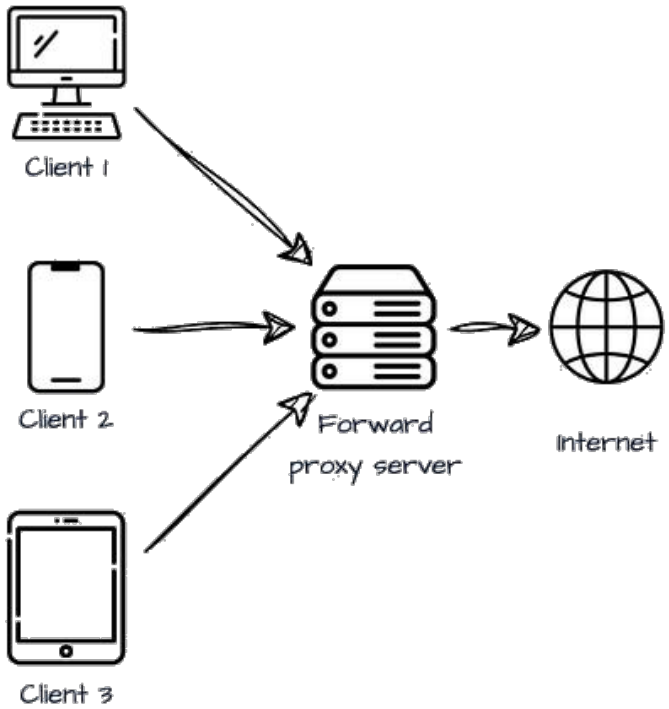
Ejemplo

```
<html>
<head> <title> AMALGAMATED WIDGET, INC. </title> </head>
<body> <h1> Welcome to AWI's Home Page </h1>
<img src="http://www.widget.com/images/logo.gif" ALT="AWI Logo"> <br>
We are so happy that you have chosen to visit <b> Amalgamated Widget's</b>
home page. We hope <i> you </i> will find all the information you need here.
<p>Below we have links to information about our many fine products.
You can order electronically (by WWW), by telephone, or by fax. </p>
<hr>
<h2> Product information </h2>
<ul>
   <li> <a href="http://widget.com/products/big"> Big widgets </a>
   <li> <a href="http://widget.com/products/little"> Little widgets </a>
</ul>
<h2> Telephone numbers </h2>
<ul>
   <li> By telephone: 1-800-WIDGETS
   <li> By fax: 1-415-765-4321
</ul>
</body>
</html>
```

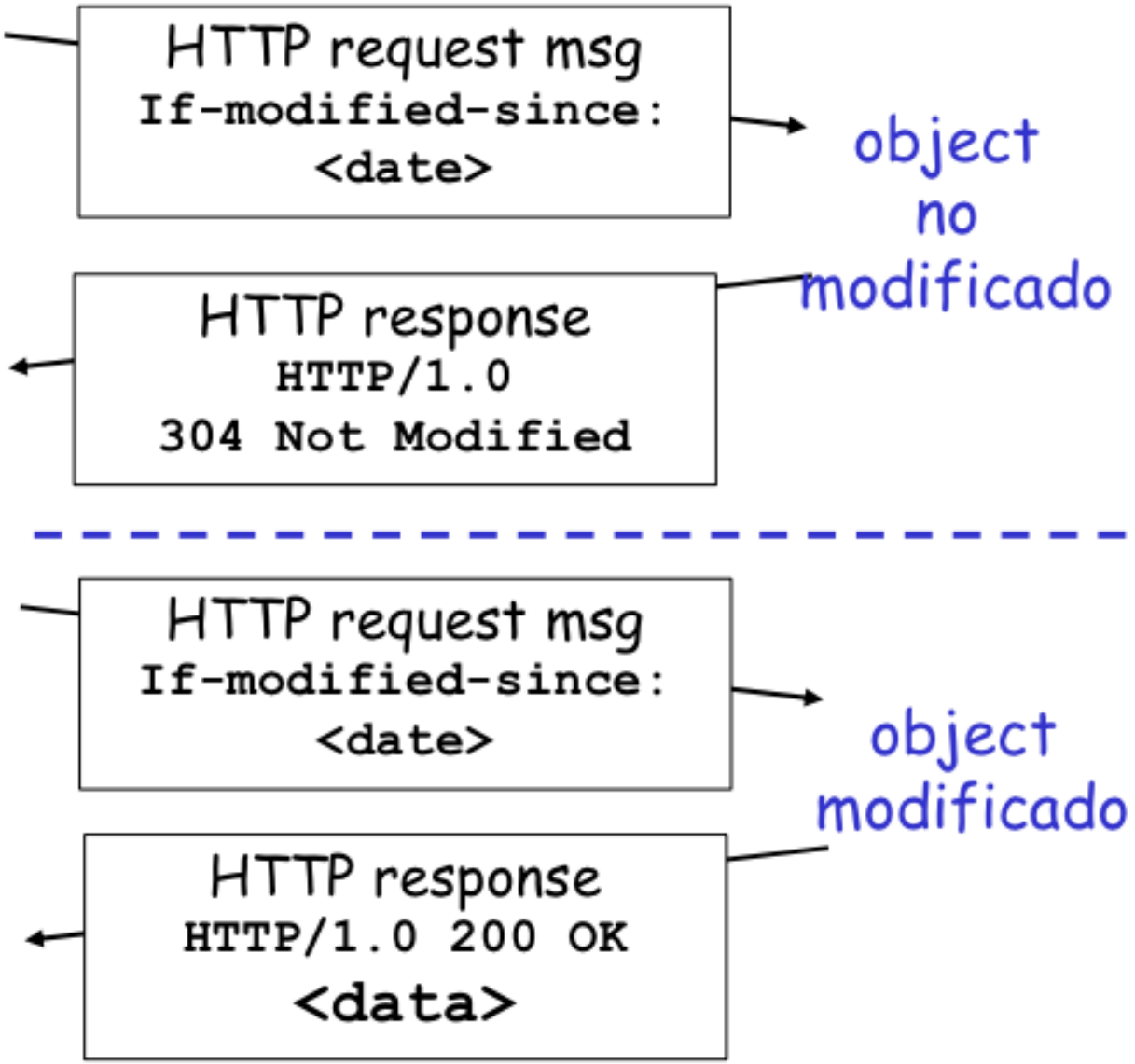
(a)

Técnicas de cacheo

- Usuario configura el browser: Acceso Web vía cache.
- Browser envía todos los requerimientos HTTP al cache:
  - Si objeto está en cache: cache retorna objeto.
  - Sino cache requiere los objetos desde el servidor Web, y retorna el objeto al cliente.



Cacheo HTTP y formato de respuesta



Ejercicio 2

Suponga la siguiente página escrita en HTML que reside en el servidor www.fcen.uba.ar:

```
1  <html>
2          <head>
3              <title>Facultad de Ciencias Exactas y Naturales</title>
4              <link rel="stylesheet" type="text/css" href="style.css" />
5          </head>
6          <body>
7              <div>
8                  <img src="searchline.png" />
9                  <a href="avsearch.php"> <img src="home.png" /> </a>
10             </div>
11             <div>
12                 <form name="searchform" action="search">
13                     <label>Buscar</label>
14                     <input name="SearchableText" type="text" title="Buscar en el Sitio" />
15                     <input type="image" src="search_icon.gif" />
16                 </form>
17             </div>
18         </body>
19     </html>
```

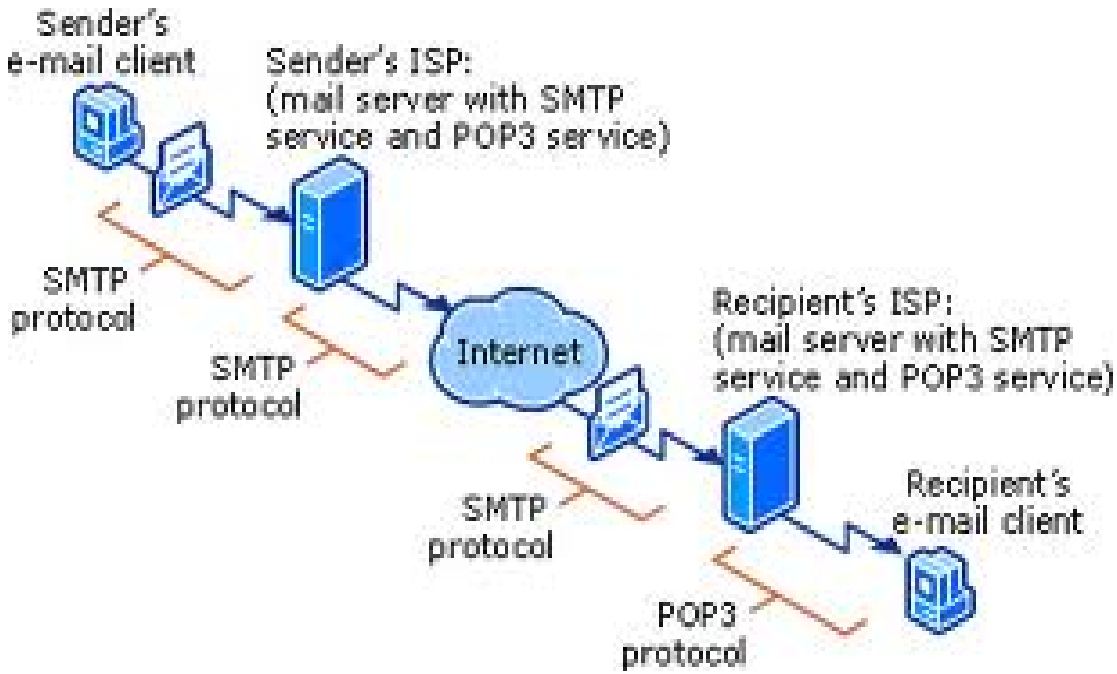
1. ¿Cuánto tiempo en términos de RTTs transcurrirá como mínimo, hasta transferir la totalidad de la información en HTTP/1.0?
2. ¿Y en HTTP/1.1?

SMTP, POP3, IMAP, Webmail, MIME

Infraestructura

Actores

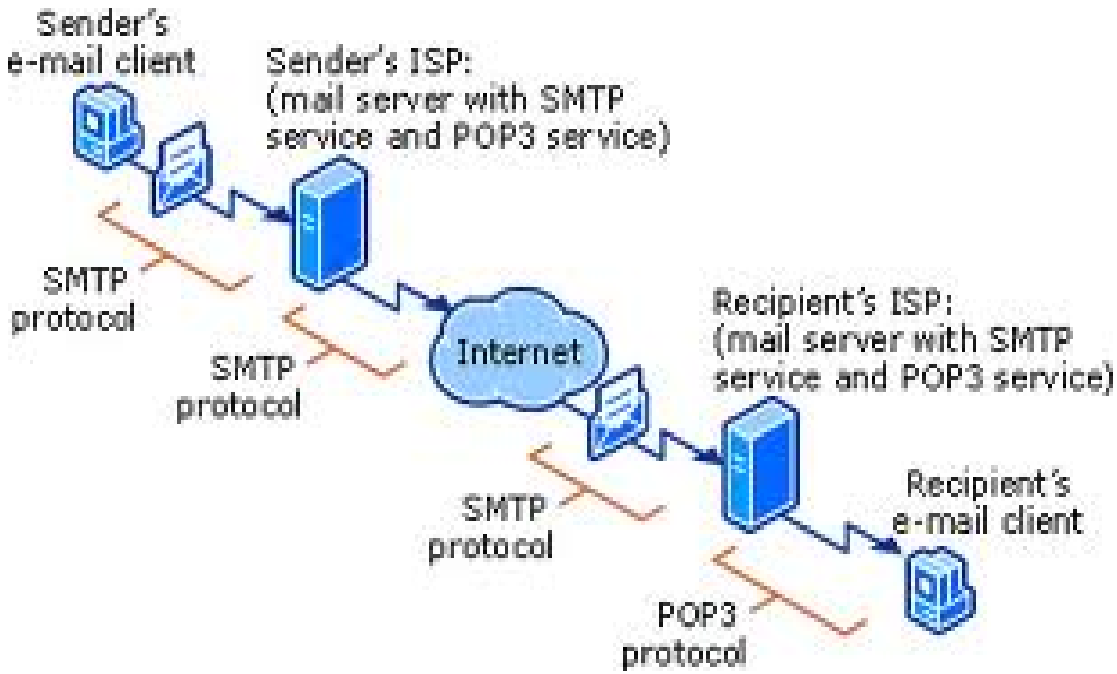
- Servidor saliente
- Servidor entratnte
- Relay
- Interfaz de usuario



Infraestructura

Protocolos

- Servidor saliente: SMTP (RFC 2821)
- Servidor entratnte: POP3 (RFC 1939)/IMAP (RFC 3501)
- Relay
- Interfaz de usuario: User Agent



SMTP: Mensajes

Cliente

- HELO
- MAIL
- RCPT
- QUIT
- DATA

Servidor

- 2xx Ok
- 5xx Error
- etc.

SMTP: Ejemplo

```
S: 220 Servidor ESMTP
C: HELO miequipo.midominio.com
S: 250 Hello, please to meet you
C: MAIL FROM: <yo@midominio.com>
S: 250 Ok
C: RCPT TO: <destinatario@sudominio.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: Subject: Campo de asunto
C: From: yo@midominio.com
C: To: destinatario@sudominio.com
C:
C: Hola,
C: Esto es una prueba.
C: Hasta luego.
C:
C: .
S: 250 Ok: queued as 12345
C: quit
S: 221 Bye
```

Mensaje de correo

Formato de mensaje estándar

- Header (Comandos Separados por CRLF).
  - From:.
  - To:, Cc:, etc.
- Body (ASCII plano).

Extensiones al formato: MIME

- En los mensajes se transfiere solo texto en formato ASCII.
- La interpretación de mensajes queda del lado del RFC822 y las extensiones MIME (RFC 2045).

Extensiones al formato de mensaje

Multipurpose Internet Mail Extensions

- ¿Por qué?
  - Quiero poder mandar más que solo texto plano.
- ¿Qué permite?
  - Transferir contenido multimedia sin modificar los protocolos básicos.
- ¿Qué se agrega para poder implementarlo?
  - Nuevos campos en el header (MIME-Version:, Content-Type:).
  - Definiciones de tipos y subtipos (Imagenes, Videos, Documentos).
- ¿Cómo hago que todo sea ASCII?
  - Encodings para codificar los distintos tipos de datos en ASCII (base64).
- ¿Cómo hago para enviar más de un tipo de archivo?
  - Uso delimitadores como tecnica de framing.

Ejemplo

```
MIME-Version: 1.0
Content-Type: multipart/mixed;
boundary="--------417CA6E2DE4ABCAFBC5"
From: Alice Smith <Alice@cisco.com>
To: Bob@cs.Princeton.edu
Subject: promised material
Date: Mon, 07 Sep 1998 19:45:19 -0400

----------417CA6E2DE4ABCAFBC5
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

Bob,

Here's the jpeg image and draft report I promised.

--Alice

----------417CA6E2DE4ABCAFBC5
Content-Type: image/jpeg
Content-Transfer-Encoding: base64
... unreadable encoding of a jpeg figure
```

IMAP: Mensajes

Cliente

- LOGIN
- SELECT
- FETCH
- DELETE

Servidor

- OK
- NO
- BAD

Ejemplo

The initial `telnet: >` symbolises your shell prompt.

```
telnet: > telnet imap.example.com imap
telnet: Trying 192.0.2.2...
telnet: Connected to imap.example.com.
telnet: Escape character is '^]'.
server: * OK Dovecot ready.
client: a1 LOGIN MyUsername MyPassword
server: a1 OK Logged in.
client: a2 LIST "" "*"
server: * LIST (\HasNoChildren) "." "INBOX"
server: a2 OK List completed.
client: a3 EXAMINE INBOX
server: * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
server: * OK [PERMANENTFLAGS ()] Read-only mailbox.
server: * 1 EXISTS
server: * 1 RECENT
server: * OK [UNSEEN 1] First unseen.
server: * OK [UIDVALIDITY 1257842737] UIDs valid
server: * OK [UIDNEXT 2] Predicted next UID
server: a3 OK [READ-ONLY] Select completed.
client: a4 FETCH 1 BODY[]
server: * 1 FETCH (BODY[] {405}
server: Return-Path: sender@example.com
server: Received: from client.example.com ([192.0.2.1])
server:          by mx1.example.com with ESMTP
server:          id <20040120203404.CCCC18555.mx1.example.com@client.example.com>
server:          for <recipient@example.com>; Tue, 20 Jan 2004 22:34:24 +0200
server: From: sender@example.com
server: Subject: Test message
server: To: recipient@example.com
server: Message-Id: <20040120203404.CCCC18555.mx1.example.com@client.example.com>
server:
server: This is a test message.
server: )
server: a4 OK Fetch completed.
client: a5 LOGOUT
server: * BYE Logging out
server: a5 OK Logout completed.
```

Desde una PC se envía un correo a tdc-doc@dc.uba.ar y luego se accede al recurso web `http://www.exactas.uba.ar/index.html`. La PC se conecta a la web a través de un servidor proxy con dirección IP `208.190.1.22`.

a. Enumere los mensajes SMTP que son necesarios para el envío de un mail a `pedro@exactas.uba.ar` desde un host en el dominio `untref.edu.ar` hasta que llega al servidor de destino.

b. Detalle los mensajes HTTP (*Requests* y *Responses*) desde que la PC envía el primer GET hasta que le llega el respectivo *Response*.

¿Repasando...?

¿Dudas?,

¿Preguntas?