Sistemas de Información

Tecnologías Web: Interactividad y envío de información Cliente → Servidor

CGI

Agradecimientos: Jesus Villamor Lugo, Simon Pickin de IT/UCIIIM.

mcfp@it.uc3m.es

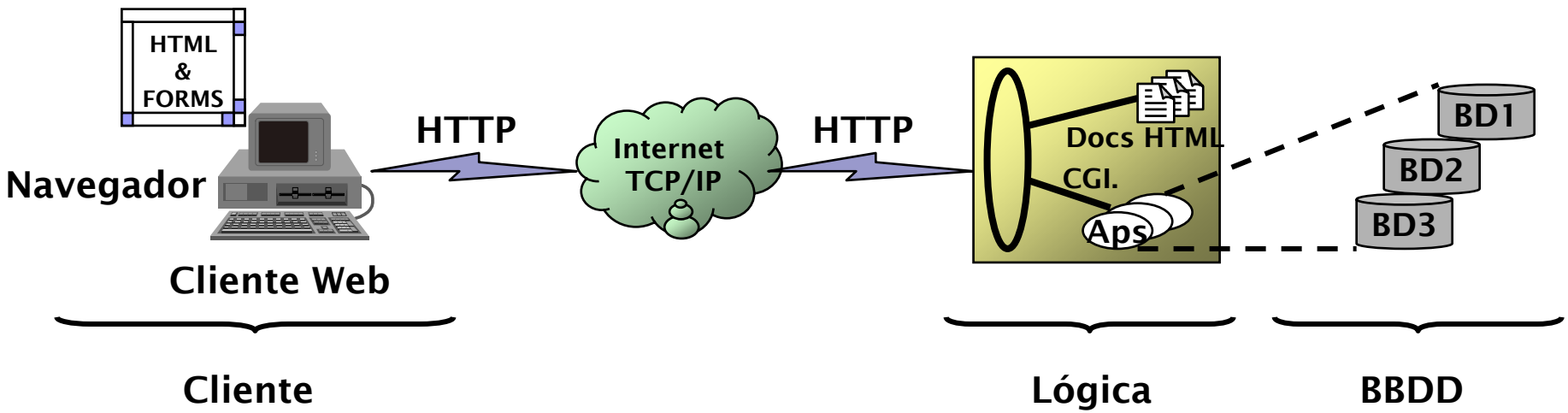
Concepto

- *CGI: **C**ommon **G**ateway **I**nterface*
- Protocolo para ejecutar programas en el servidor vía HTTP.
- Permite hacer disponible en el servidor HTTP:
  - programas escritas en cualquier lenguaje interpretado o compilado. (C, Perl y Bourne Shell)
- Normalmente, las aplicaciones exportadas residen en directorio particular (usualmente *cgi-bin*)
  - Se puede configurar el servidor para utilizar otros directorios.

¿Qué hace la parte servidora?

- Recibe unos datos, los procesa y devuelve otros
- Toma los datos por la entrada estándar (teclado) y envía los datos de salida por la salida estándar (pantalla).
- Cualquier programa capaz de leer por entrada estándar y escribir en salida estándar puede actuar como CGI
- El usuario no puede interacturar directamente con el programa CGI. (salvo llamadas a varios cgis)

Arquitectura CGI (Ejemplo)



- Ej.: CGI hace que clientes Web puedan actualizar BBDD.
- Uso típico en aplicaciones con tres niveles:
  - ***Cliente***: navegador que envía peticiones HTTP GET ó POST que contienen los datos que han sido introducidos mediante un formulario.
  - ***Logica***: programa que procesa la solicitud, recibe parámetros por medio del CGI, toma acciones y responde (por abuso de lenguaje, a veces se le llama “un programa CGI” o “un CGI”).
  - ***Base de datos***: puede ser actualizada por la acción del programa

¿Qué necesito?

- Página HTML
- Programa o script que lea datos de entrada estándar y escriba en salida estándar
- Permisos adecuados en el servidor para permitir su ejecución
  - chmod 750 *.cgi" (o "chmod 755 *.cgi", si tu servidor no tiene accesos de grupo a tus archivos)
  - En los laboratorios de telemática utilizad “chmod 755 *.cgi”

¿Cómo referenciar un CGI desde una página web?

- Usando **etiquetas HTML**
  - Usando la etiqueta **A**:

    Texto
  - Usando la etiqueta IMG:

    <**img** src="direccion_del_CGI">
  - Otras (javascript, css, etc.)
- Usando un **formulario** (forma + habitual):

  <form action="direccion_del_CGI" > <!-- Elementos del formulario --> </form>

Ejemplo (usando etiquetas)

fecha.html

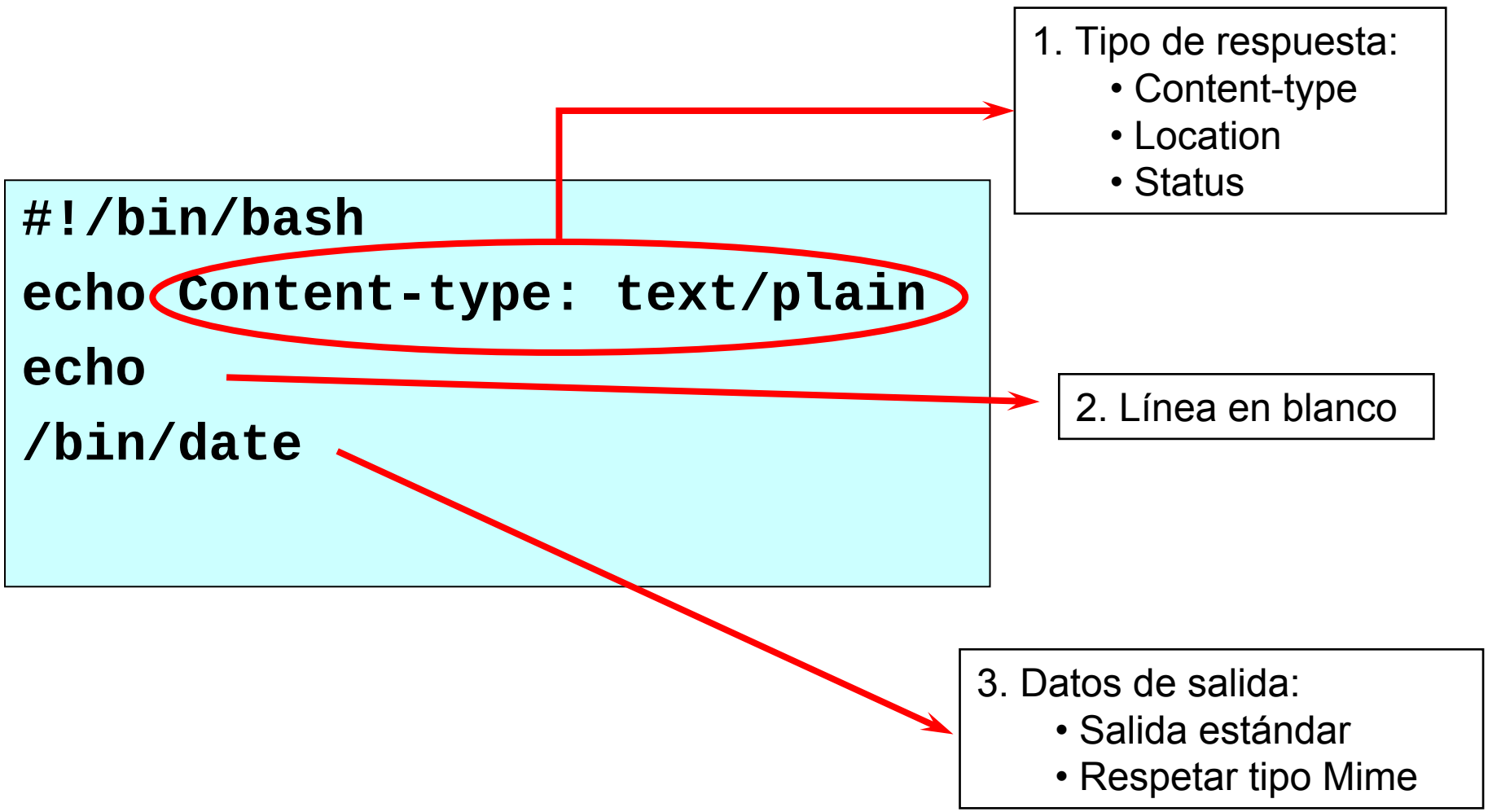
```
<a href="http://www.sitioweb.es/cgi-bin/fecha.cgi"> Dime la fecha </a>
```

fecha.cgi

```
#!/bin/bash
echo Content-type: text/plain
echo
/bin/date
```

Prueba

Anatomía de un CGI



Tipo de respuesta

- **Content-type**
  - text/html
  - image/gif
  - video/mpeg
- **Location**
  - Para enviar archivo existente como respuesta
  - Location: response.html
- **Status**
  - Si el script sirve para tratar condición de error
  - Envio información del estado
  - Códigos
    - **1xx** msg de información
    - **2xx** éxito de algún tipo (200 Ok la petición ha tenido éxito)
    - **3xx** redirección a otra url (301 Moved Permanently)
    - **4xx** error en cliente (404 not found. The requested resource doesn't exist)
    - **5xx** error en el servidor (500 Server Error)

Probando en el laboratorio

- Páginas **HTML**
  - ¿Dónde las pongo?

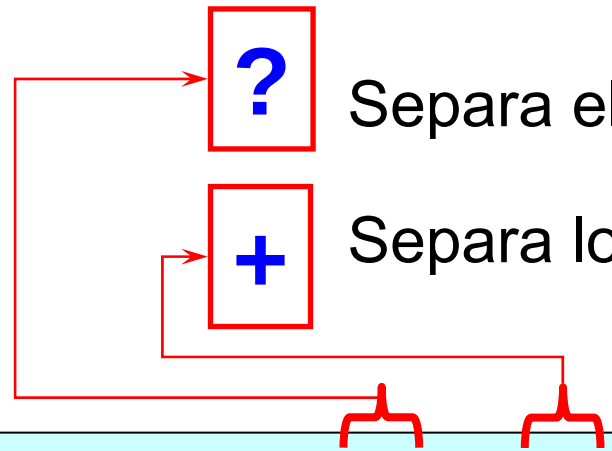
    ~tuCuenta/lib/www/
  - ¿Cómo accedo a ellas?

    http://www.lab.it.uc3m.es/~tuCuenta
- Programas y scripts **cgi**
  - ¿Dónde los pongo?

    ~tuCuenta/lib/www/cgi-bin/
  - ¿Cómo accedo a ellos?

http://www.lab.it.uc3m.es/alum-cgi/tuCuenta/cgi-bin/app.cgi

CGIs con argumentos




```
<a href="/cgi-bin/cgi?arg1+arg2+arg3">pulsa aqui</a>
```

```
<a href="/cgi-bin/cgi/infoAdicional?arg1+arg2">pulsa aqui</a>
```

Información adicional recuperable utilizando Variable de entorno **PATH_INFO**

CGIs y Formularios

- Usando GET
  - Se usa para obtener fichero o recurso
  - Los datos se leen de var de entorno QUERY_STRING
  - Tamaño limitado 256 caracteres
  - No registra cada petición
- Usando POST
  - Se usa para enviar datos al servidor
  - Los datos se leen de la entrada estándar
  - Tamaño ilimitado (preguntar CONTENT_LENGTH)
  - Registra cada petición

CGIs y Formularios

<form action="**cgi-bin/fichero.cgi**"> Contenido </form>

- Información del formulario codificada en pares campo/valor
- **&** separa entre sí las diferentes parejas **campo/valor**
- **=** Separa el nombre del campo de su valor
- **%NN** Identifica los caracteres especiales,(≠ ASCII 7 bits) NN es el valor en hexadecimal (ASCII extendido)
- **%NN** También para codificar **&**, **=** y **%** cuando son datos para no confundirlos con caracteres de control
- **+** para codificar los espacios

Ejemplo

getYpost.cgi

```
#!/bin/bash
echo "Content-Type: text/html"
echo ""
if [ $QUERY_STRING != "" ]
  then
      echo "Datos enviados por GET<br/>"
      echo "----------------------<br/>"
      echo $QUERY_STRING
  else
      echo "Datos enviados por POST<br/>"
      echo "----------------------<br/>"
      cat
fi
```

1. Tipo de respuesta

2. Línea en blanco

3. Datos de salida

PruebaGet

PruebaPost

Ejemplo

- Probando get:

PruebaGet (sin página web)

```
http: //monitor03.lab.it.uc3m.es/alum-cgi/00xxxxx/cgi-bin/getYpost.cgi?mensaje=hola
```

PruebaGet.html (desde página web)

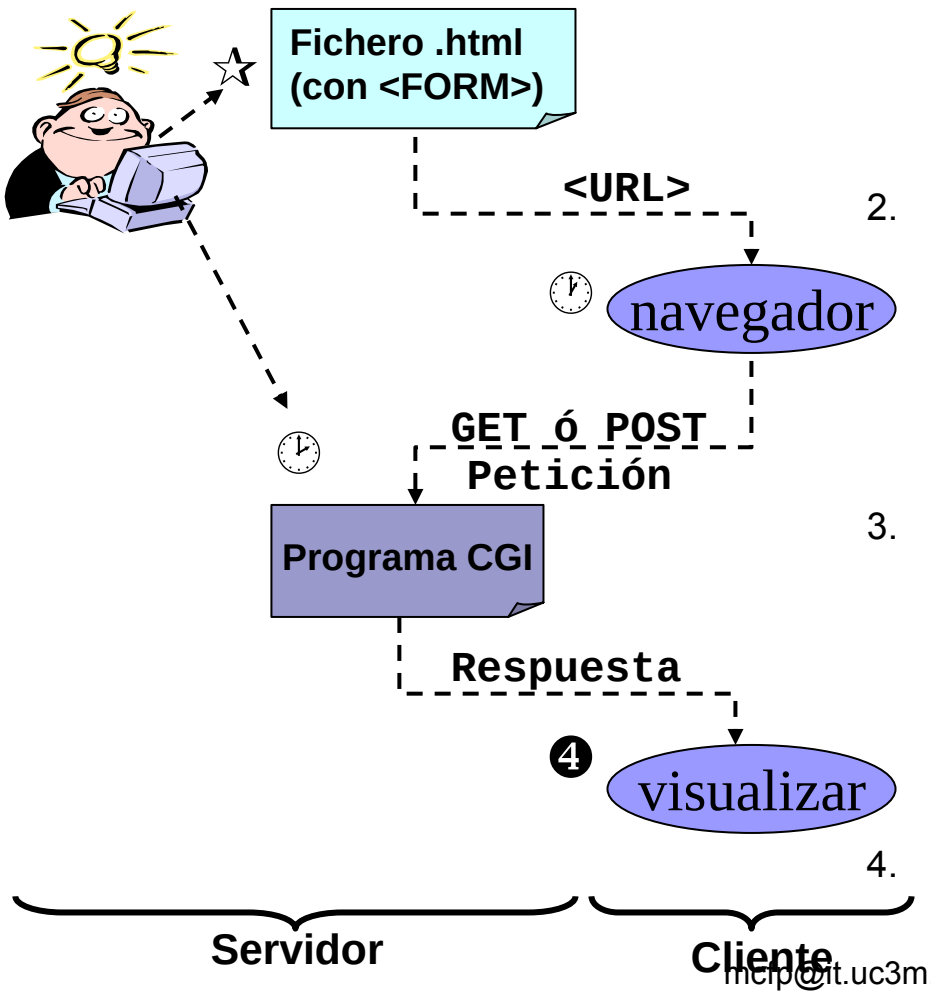
```
<html>
  <body> <a href="DireccionCgi?arg1+arg2+arg3">pulsa aqui</a> </body>
</html>
```

- Probando post:

PruebaPost.html

```
<html>
  <body>
    <form method="post" action="direccionDelCgi">
      Texto: <textarea name="mensaje"></textarea> <br/>
      <input type="submit" value="Enviar"/>
    </form>
  </body>
</html>
```

HTTP/CGI
Proceso de Desarrollo



1. Se parte de un Fichero.html **que tenga un formulario** (etiqueta <FORM>) accesible a través de Internet
   - El <FORM> deberá especificar las diversas formas de paso de parámetros
2. El cliente accederá al fichero html a través de un Navegador, rellenará el formulario y dará a la tecla de aceptar (Submit). Empaquetando la petición.
3. La petición de cliente es atendida por un Programa CGI (usualmente un script) quien da la debida respuesta tras procesarse la petición.
4. El cliente Visualizará la respuesta

Acceso al CGI

Entrada vía formularios

- Mediante el uso de formularios HTML:
  - Etiqueta <FORM>
    - Comunica al cliente que estamos comenzando un formulario.
    - Dos atributos:
      - Método:
        - cómo recoger la información del usuario.
        - **GET** (por defecto) o **POST**.
      - Acción:
        - tipo de URL que recibirá el formulario (puede no ser CGI, ej. mailto).
    - Ejemplo:

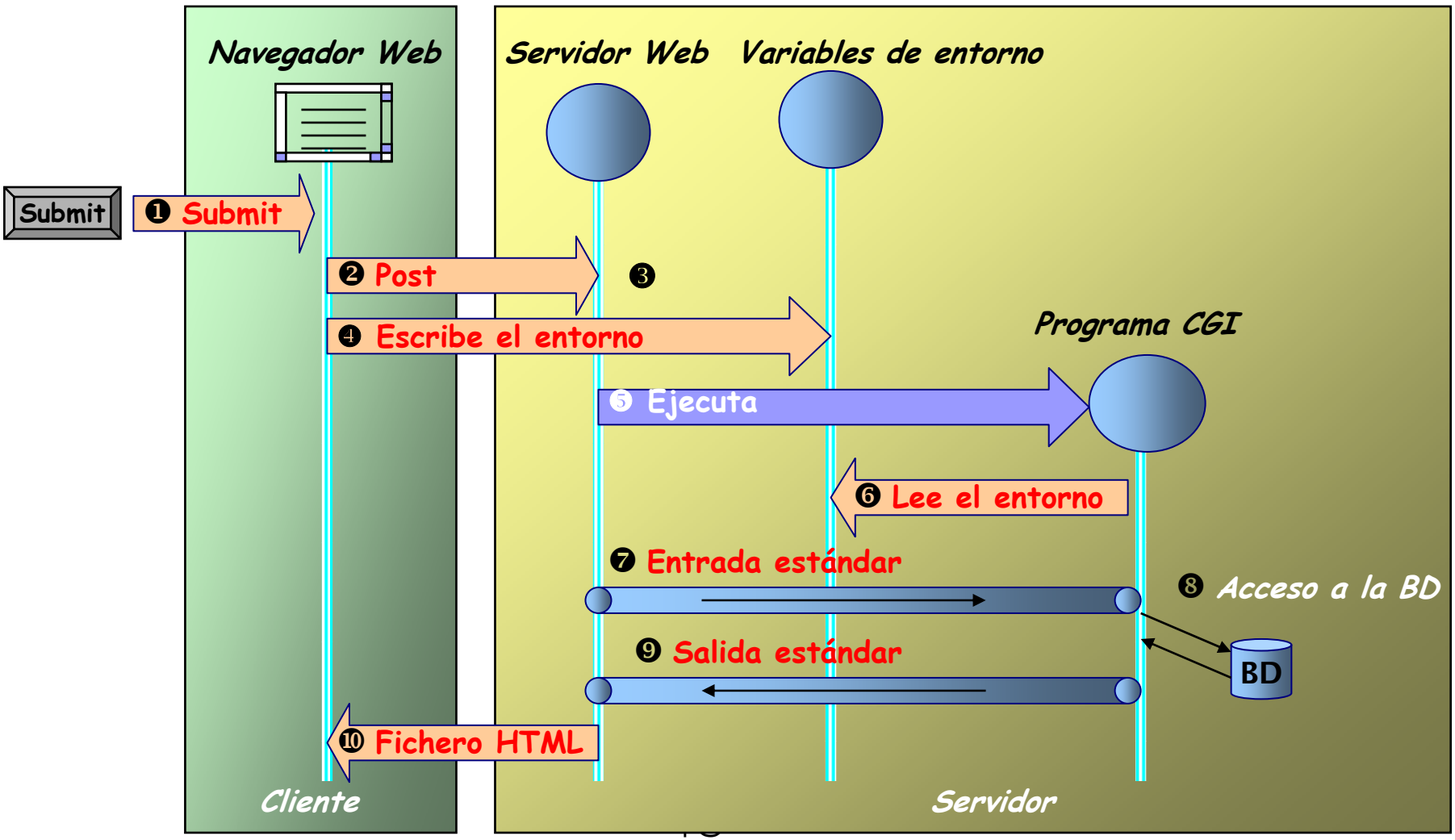
      <form method="POST" action="http://www.ncsu.edu/cgi-bin/post-query">
  - Campos de Información: <input ...>, <select ...>, <textarea ...>
    - Name, size, type (ya explicados bajo HTTP/HTML)
    - Recoge datos en forma de parejas de cadenas (*nombre*, *valor*)

Acceso al CGI

Cómo pasar parámetros al programa

- No se puede pasar datos al programa vía línea de comando.
  - La forma de hacerlo depende del método de envío.
- Primero, se construye la cadena llamada *query string*
  - es una serie de parejas *nombre=valor* separados por *&*
    - ej: nombre1=valor1&nombre2=valor2&...
  - se codifica de manera similar al "quoted printable" de MIME:
    - espacios reemplazados por "+" y caracteres especiales en hexadecimal
- Segundo, se envía al CGI según el método de envío
  - método GET:
    - se agrega a la URL, separado por un signo de interrogación:
      http://<host>:<port>/<path>?<query_string>
    - El CGI lo lee mediante la variable de entorno QUERY_STRING
  - método POST:
    - Se pone **en el cuerpo del mensaje** que se envía en el POST.
    - El CGI lo lee por entrada estándar (stdin)

Escenario básico
Con el método POST



Escenario básico
Con el método POST

1. El usuario pincha en el botón *submit*.
2. El navegador envía la petición con el método POST de HTTP.
3. El servidor recibe la petición y descubre el método y la acción.
4. El servidor establece las variables de entorno tales como *server_name*, *request_method*, *content_type*, *content_extension*, etc.
5. El servidor inicia la ejecución del programa CGI especificado en el URL
6. El programa CGI lee las variables de entorno y, en particular, descubre que está respondiendo a un POST.
7. El programa CGI recibe el cuerpo del mensaje − que contiene el *query string* − por la entrada estándar y utiliza la variable *content_extension* para saber el tamaño de los datos (no disponible con GET).
8. El programa CGI hace sus labores (interactúa con la BD etc.), construye la respuesta en forma de un tipo MIME reconocido y posiblemente escribe también las cabeceras HTTP de la respuesta.
9. El programa CGI devuelve los resultados por la salida estándar.
10. El servidor devuelve los resultados al navegador, añadiendo las cabeceras HTTP si el programa no las ha proporcionado.

CGI. Un protocolo sin estado

- El protocolo CGI es totalmente **sin estado**
  - después de responder, el servidor lo olvida todo
- Soluciones:
  - Campos ocultos en el siguiente formulario
    - se transmiten al servidor, **cuando el cliente invoca el "submit" del formulario**
      - Normalmente el número de campos ocultos va aumentando conforme se suceden los formularios
  - Cookies
    - Pequeñas porciones de datos del servidor que éste almacena en su nombre en el cliente
      - Típicamente almacenan identificadores de usuarios o información de configuración
      - Los cookies se transmiten al servidor **en posteriores peticiones del cliente**

CGI. Consideraciones de seguridad

¿Dónde está el riesgo?

- Peligros
  - Ejecutar programa en servidor
  - Controlar ejecución mediante paso de parámetros
- Vulnerabilidad
  - Programa interpretado
    - Ejecutar programas imprevistos (concierne sobre todo a la invocación de comandos del sistema / de la shell desde el programa CGI)
    - Revelar información del servidor donde se ejecuta (p.e. comandos *finger*, *ps*). Puede ayudar a un usuario que quiera montar un ataque)
  - Programa compilado
    - Hacer suposiciones sobre el tamaño de los datos de entrada
      - puede llegar a producirse el desbordamiento de sus buffers
      - el crash consiguiente puede ser explotado por el atacante

CGI. Consideraciones de seguridad
¿Dónde está el riesgo?

- Server side includes (SSI)
  - Directivas incrustadas en código HTML
    - Muestran contenido de un fichero
    - Muestran salida de un comando
  - Ventajas: flexibilidad, contenido dinámico
  - Inconvenientes
    - Portabilidad
    - El autor de la página decide qué ejecutar, con qué parámetros
  - Recomendación: Deshabilitarlos

CGI. Consideraciones de seguridad
Consejos generales

- No confiar en que el cliente haga nada
- Mejor compilado que interpretado
- Precaución
  - al manejar ficheros (leer y escribir, permisos)
  - con la interacción con otros programas (eval, system, etc.)
- Nunca poner permiso "suid" (*set user id*) a un CGI
- No hacer suposiciones acerca del tamaño de la entrada
  - en caso de duda, POST mejor que GET
- **Nunca** pasar a un comando shell entradas del usuario sin chequear
  - en el chequeo, se debe eliminar metacaracteres de shell:

    &;`'\"{}|*?~<>^()[]{}$\n\r

CGI. Consideraciones de seguridad

Consejos para el administrador

- Si usas cgis de otros:
  - Qué chequear
    - ¿Lee o escribe ficheros en el servidor?
    - ¿Interactúa con otros programas del sistema?
    - ¿Corre con privilegios suid?
    - ¿Se valida la entrada procedente de formularios?
    - ¿Se emplean nombres de camino explícitos?
  - Qué hacer
    - Evitar que tengan más **privilegios** de los necesarios.
    - **Aislarlos del resto del sistema**. (Ejecutar los CGI en un entorno set-user-id)
    - **Establecer límites de recursos**, (memoria, CPU o espacio en disco)

CGI. Consideraciones de seguridad
Consejos para el desarrollador

- Si creas tus propios CGIs.
  - Comprobar siempre la **longitud de las cadenas**
  - **Filtrar** siempre datos que vengan del exterior:
    - **No pasar nunca datos** del exterior **sin comprobarlos** antes (ej: paso de argumentos a un comando del shell)
    - **No usar nunca datos** del exterior **sin comprobarlos** (ej: seleccionar un nombre de fichero para su apertura).
  - **Comprobar el código de retorno** de todas las llamadas al sistema.
  - **No hacer suposiciones** acerca del directorio de trabajo actual, el entorno o el camino de ejecución de comandos.
  - Chequear contenido de **campos ocultos**
  - **No revelar** demasiada información sobre el sistema, con servicios como: finger

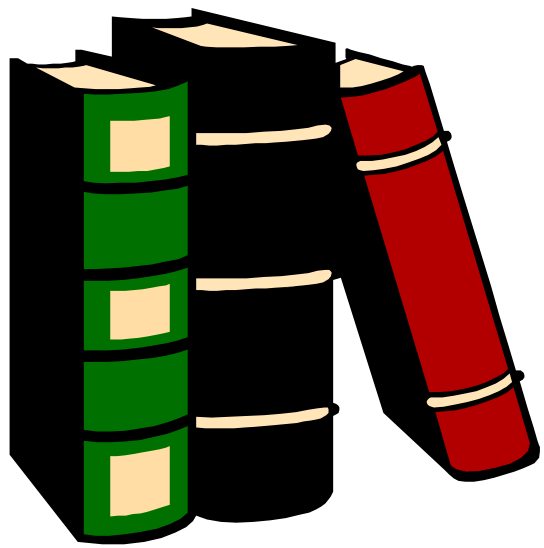
CGI
Algunas referencias

– ***Dan Harkey, Robert Orfali***, ***Client/Server Programming with Java and CORBA, 2nd Edition*** *2nd. Edition (1998) John Wiley & Sons, Inc. ISBN: 0-471-24578-X*
Mirarse el capítulo 11

–***William E. Weinman,*** ***The CGI Book*** *Bk&Cd Rom edition (1996) New Riders Publishing; ISBN: 1562055712*

– ***Stephen Asbury, et al.*** ***Cgi How-To : The Definitive Cgi Scripting Problem-Solver*** *Bk&Cd-Rom edition (1996) Waite Group Pr; ISBN: 157169028X*

–***Thomas Boutell***. ***Cgi Programming in C & Perl*** *(1996) Addison-Wesley Pub Co; ISBN: 0201422190*



Bibliografía Web

- Información general:

  http://www.programacion.com/tutorial/cgi/
- Consejos de seguridad:

  http://www.iec.csic.es/criptonomicon/cgi/
- Ejemplos prácticos:

  http://www.jmarshall.com/easy/cgi/spanish/

Repositorios de CGIs

- http://www.cgi-resources.com/Programs_and_Scripts/
- http://www.cs.cmu.edu:8001/afs/cs/usr/rgs/mosaic/perl.html
- http://sunsite.unc.edu/~boutell/cgic/cgic.html

Más ejemplos

- http://www.fpx.de/fp/Software/ProcCGIsh.html
- http://www.yolinux.com/TUTORIALS/LinuxTutorialCgiShellScript.html
- Cgis y cookies:

  http://www.iec.csic.es/criptonomicon/cookies/receshell.html

CGIs y variables de entorno

Variable	Significado
SERVER_NAME	Dirección IP del host donde está el CGI.
SERVER_SOFTWARE	Tipo de servidor Web que se está usando.
GATEWAY_INTERFACE	Versión del interfaz CGI.
SERVER_PROTOCOL	Versión del protocolo HTTP.
SERVER_PORT	El puerto TCP que se está usando. En la mayoría de servidores Web es el 80.
REQUEST_METHOD	Método de envío de información: POST o GET.
HTTP_ACCEPT	Lista de los tipos MIME (content-types) que acepta el navegador.
HTTP_USER_AGENT	El navegador usado por el usuario.
HTTP_REFERER	Dirección URL del documento HTML donde estaba el formulario.

CGIs y variables de entorno

Variable	Significado
PATH_INFO	Información extra (visto antes)
PATH_TRANSLATED	La variable PATH_INFO adaptada al sistema específico donde está el CGI.
SCRIPT_NAME	Nombre del CGI.
QUERY_STRING	Argumentos pasados al CGI.
REMOTE_HOST	Nombre del ordenador que envió los datos.
REMOTE_ADDR	Dirección IP de dicho ordenador.
REMOTE_USER	Nombre del usuario.
REMOTE_IDENT	Sirve para el método de identificación ident.
CONTENT_TYPE	Tipo de información que llega por la entrada estándar. Por defecto será: x-www-form-urlencoded que es la codificación que hemos visto antes.
CONTENT_LENGTH	Longitud de los datos que llegan por la entrada estándar cuando hemos usado POST.