Capitulo 7: Tecnologías de la Capa Media, Lógica del Negocio.

- Arquitectura MVC.
- Operaciones de la capa media.
- CGI, PHP, Java Servlet y JSP.
- Diseño de la capa media.

Temario

- Revision de la Arquitectura MVC.
- Operaciones de la capa media.
- CGI, PHP, Java Servlet y JSP.
- Diseño de la capa media.
- CGI
- Apache Module: PHP
- Java: Servlets, JSP
- Aplicación de MVC: Construcción de prototipos Boundary, Controler y Entity.

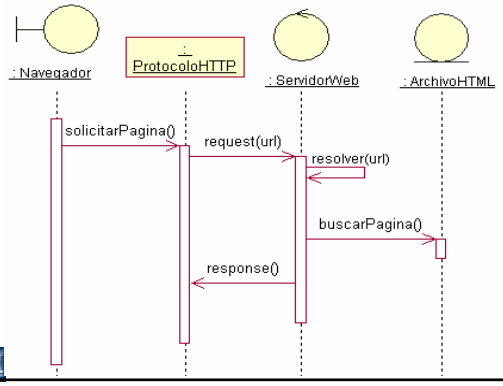
Operaciones en la capa media

- Comunicación con la capa de interfase
  - Recepción de datos de la capa cliente.
  - Atención a las peticiones de operaciones de la capa cliente.
  - Envío de datos para ser "presentados" (Rendering)
- Ejecución de Lógica del negocio
  - Procesamiento de datos.
  - Validación de condiciones del proceso.
  - Control de transacciones.
- Comunicación con la capa de datos
  - Peticiones de Datos.
  - Actualizaciones de datos.
  - Control del estado de las transacciones.

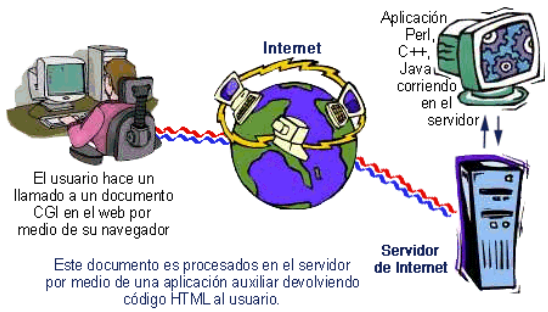
Diferentes arquitecturas para la capa media

- Pagina HTML en la capa media
- CGI-BIN
- Apache Modules (PHP)
- Servlet y JSP

Mecanismo de solicitud de paginas estáticas HTML.



CGI: Common Gateway Interface



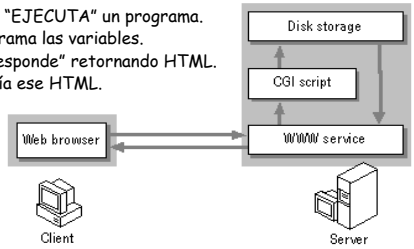
Procesamiento en la CM: CGI-BIN

- Mecanismo mas antiguo de procesamiento.
- Todos los servidores implementan el mecanismo por defecto.
- El Web "habla" HTML. ¿Qué pasa si deseamos hacerlo interactuar con datos que están en otro formato?
- Surge la idea de hacer un conversor, algo así como un formateador de datos.
- El formateador puede ser un programa escrito en algún lenguaje.
- El Web Server "controla" el flujo de datos de la pagina al programa, el lenguaje debe soportar I/O estándar para recibir datos y enviar la pagina de manera correcta.

CGI-BIN: El proceso

CGI (Common Gateway Interface)

- El CGI especifica como se pasan los datos desde el servicio WWW a una aplicación externa y como se recuperan los resultados. Como todo en el WWW, la interfaz CGI es simple pero muy potente.
- Servidor ahora "EJECUTA" un programa.
- Le pasa al programa las variables.
- El programa "responde" retornando HTML.
- El servidor envía ese HTML.



CGI-BIN: El formulario

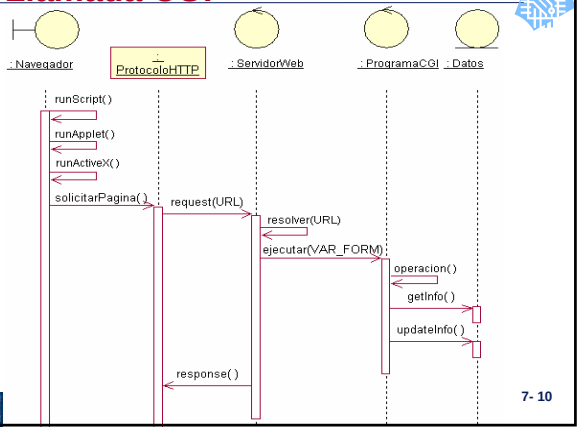
Dos métodos de envío

- **Los formularios electrónicos sirven para recibir datos desde el usuario y almacenarlos en algún dispositivo.**

Un formulario tiene la siguiente estructura:

```
<form action="url" method={Post|Get} >
<input type=... Name="var1">
.....
<input type=submit value=Enviar>
<input type=reset value=Borrar>
</form>
```


Llamada CGI



CGI-BIN: El programa receptor

- El CGI-BIN se puede escribir en cualquier lenguaje que soporte STDIN y STDOUT

```
typedef struct {
      char name[128];
   char val[128];
} entry;

int m,x;

   cl = getenv("QUERY_STRING");
   for(x=0;cl[0] != '\0';x++) {
      m=x;
      getword(entries[x].val,cl,'&');
      plustospace(entries[x].val);
      unescape_url(entries[x].val);
      getword(entries[x].name,entries[x].val,'=');
   }
```


CGI BIN : El Método GET

1. El cliente WWW solicita un servicio de una aplicación CGI.
2. El servidor HTTPD recibe la solicitud y los datos de entrada.
3. El servidor crea un ambiente y variables con los datos de entrada.
4. El servidor ejecuta la ampliación CGI en este ambiente.
5. El CGI procesa las variables de ambiente y recupera los datos de entrada.
6. La aplicación CGI se ejecuta produciendo un resultado sobre su salida estándar.
7. El servidor HTTP redirecciona la salida estándar de la aplicación CGI hacia el cliente WWW.
8. El cliente WWW recibe el resultado de su consulta.

CGI BIN: El Método POST

1. El cliente WWW solicita un servicio de una aplicación CGI.
2. El servidor HTTPD recibe la solicitud y los datos de entrada.
3. El servidor ejecuta la ampliación CGI pasándole la información a través de la entrada estándar.
4. La aplicación CGI procesa su entrada estándar y recupera los datos de entrada.
5. La aplicación CGI se ejecuta produciendo un resultado sobre su salida estándar.
6. EL servidor HTTP redirecciona la salida estándar de la aplicación CGI hacia el cliente WWW.
7. El cliente WWW recibe el resultado de su consulta.

Lenguajes de programación de CGI

- El Web Server llama a un CGI creador de una página Web. Luego toma el código de la página y lo envía al browser que lo solicitó.
- Desde un punto de vista funcional, la transacción anterior fue un intercambio de archivos entre dos procesos.
- El intercambio clásico de archivos entre procesos es a través de la I/O estándar.
- Entonces, los lenguajes de programación de CGI deben soportar I/O

Lenguajes de programación de CGI (2)

- Perl. Practical Extraction and Report Language es un lenguaje de programación para la programación en sistemas unix.
- Surgió de otras herramientas de UNIX como son : sed,grep,awk,c-shell.
- Sirve para labores de procesamiento de texto, para la programación de software de sistemas y ahora último para programar aplicaciones para Web

Lenguajes de programación de CGI (3)

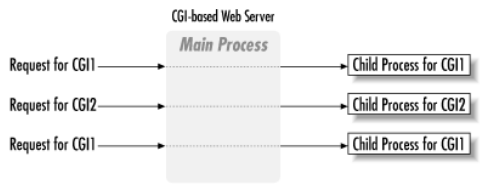
- C Desde su creación, soporta la entrada y salida estándar, por cuanto siempre ha privilegiado la comunicación entre procesos
- C++
- Para conectarlos con bases de datos, se han usado APIS y características adicionales de los lenguajes, tales como Pro*C

CGI: Ventajas y Desventajas

- Ventajas:
  - Ejecuta un PROCESO PESADO por cada conexión lo cual lo hace útil para usuarios que requieran de efectuar "grandes" procesamientos vía web.
  - Arquitectura primitiva pero simple.
  - No tiene un lenguaje definido.
  - Requiere de poca configuración.
- Desventajas:
  - **Ineficiente para sitios con muchos usuarios conectados** (Memoria y tiempo de procesador).
  - Se deben manejar muchas cosas de bajo nivel (ej: post, get)

El ciclo de vida de un CGI

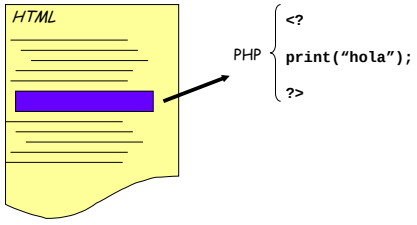
1 proceso por cliente



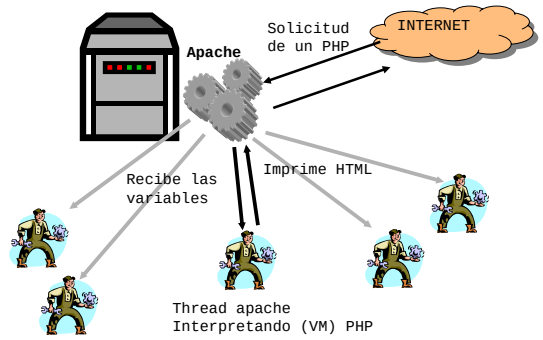
Módulos de Apache: PHP, the revolution is comming.

El próximo paso lógico a los CGI.

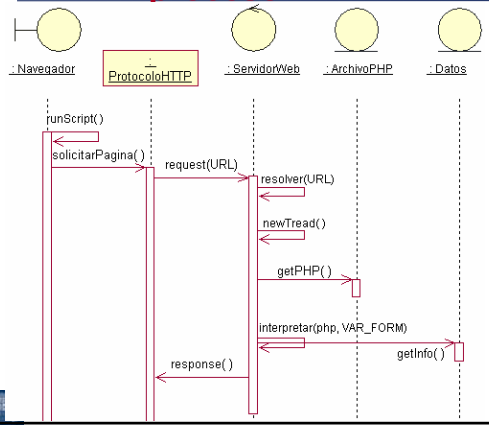
PHP: Hypertext Pre-Procesor



PHP: como modulo de apache



PHP: El proceso



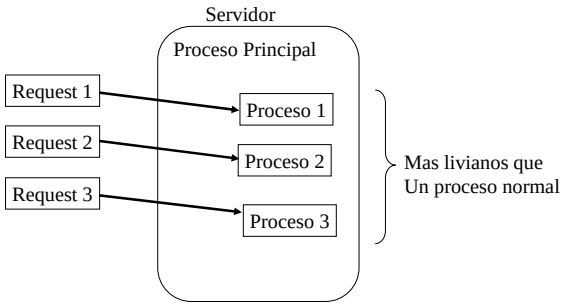
PHP: Características Básicas

- Fácil de aprender. En 2 semanas aprendizaje se pueden lograr aplicaciones Web con uso de bases de datos.
- No tiene tipos de datos. Se puede hacer "hola"+1 sin arrojar errores.
- Software Libre.
- Programas son textos.
- Al encontrarse como módulo de apache se evita el overhead que significa levantar un nuevo proceso en el caso de los CGI. (=mas liviano que un CGI)
- Apache además provee de un manejo de sessiones. (CGI no tiene un manejo directo de ellas)

PHP: Desventajas

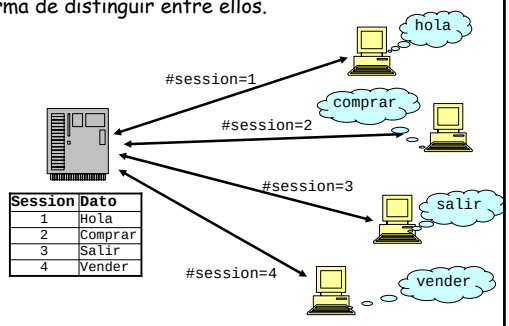
- Grandes cantidades de clientes, como sitios de e-commerce muy solicitados pueden sobrecargar al servidor. (miles de conexiones por minuto)
- Se puede buscar mejorar la situación por el lado de un balanceador de carga u optimizaciones del software (compilar php, cache del interprete, …).
- PHP como lenguaje no restringe al programador a un esquema ordenado y OO. Aunque si lo soporta.
- No tiene garantía.

Ciclo de vida de PHP



Sesiones

- Conjunto de datos asociados a cada cliente conectado. De forma de distinguir entre ellos.



Session	Dato
1	Hola
2	Comprar
3	Salir
4	Vender

Java en el Servidor: Servlet y JSP.

La orientación al objeto en la capa media.

Java en el Servidor: Servlets

- Funcionamiento
  - Se instalan los programas en un Servidor Web que Soporte este tipo de Aplicación
  - Se conectan los clientes en la URL del Servlet
  - Ejecuta distintos métodos según el tipo de requerimiento (GET, POST, PUT, etc.)
  - Devuelve una página HTML normal.

Java en el Servidor: Servlets (2)

- Aspectos Destacables
  - El Servlet permite manejar en forma separada cada tipo de Petición
  - Tiene un manejo muy eficiente de los threads
  - Permite manejo de Sesiones
  - Genera Páginas HTML legibles por cualquier Browser

Java en el Servidor: Servlets (4)

- Aspectos Destacables (cont.)
  - Permite el desarrollo de aplicaciones utilizando Java, visibles desde cualquier Browser
  - Cualquier modificación del código se actualiza automáticamente
  - Permite el desarrollo de aplicaciones realmente escalables

Java en el Servidor: Servlets (5)

- Aplicaciones Típicas
  - Interacción con Base de Datos
  - Aplicaciones Transaccionales
  - E-business
  - etc.

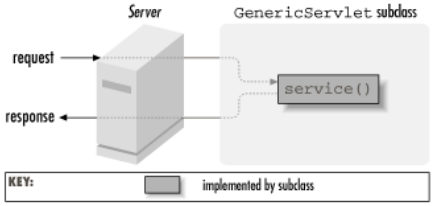
Servlets

- Son aplicaciones del lado del servidor construidas en Java
- Son "equivalentes" a los CGI.
- Al usarlos con una base de datos, generan una sola conexión y atienden a los clientes a través de thread.

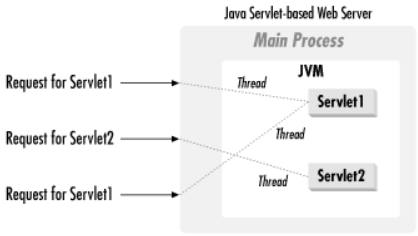
¿ Qué son los Servlets ?

- Ventajas sobre los CGI
  - Construidos en Java
    - Más ordenados
    - Amplio Soporte del Mercado
  - Excelente Manejo de Procesos
    - Cada conexión se maneja como un proceso liviano
  - Utilización de Recursos
    - Pueden compartir recursos para todos los proceso

¿ Qué son los Servlets ?



¿ Qué son los Servlets ?



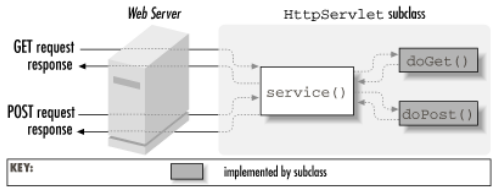
Ciclo de Vida de un Servlet (Modelo Request/Response)

- Inicialización
  - Se realiza cuando el primer cliente hace una petición del Servlet
  - Se define mediante el método **init**
    - Se pueden especificar parámetros de configuración
    - Se pueden definir variables que son visibles para cualquier proceso de manejo de peticiones

Ciclo de Vida de un Servlet (2)

- Manejo de Petición
  - Para cada cliente que hace una petición del Servlet, se crea un proceso que lo maneja
  - Se ejecutan distintos métodos según el tipo de petición
    - Petición GET (URL's): método **doGet**
    - Petición POST (Formularios): método **doPost**
    - Petición PUT (Upload): método **doPut**

Ciclo de Vida de un Servlet (2)



Ciclo de Vida de un Servlet (2)

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class HelloWorld extends HttpServlet {
  public void doGet(HttpServletRequest req, HttpServletResponse res)
              throws ServletException, IOException {
    res.setContentType("text/html");
    PrintWriter out = res.getWriter();
    out.println("<HTML>");
    out.println("<HEAD><TITLE>Hello World</TITLE></HEAD>");
    out.println("<BODY>");
    out.println("<BIG>Hello World</BIG>");
    out.println("</BODY></HTML>");
  }
}
```


Ciclo de Vida de un Servlet (3)

- Destrucción
  - Se ejecuta cuando se baja el servidor de Aplicación
  - Antes el servidor llama el método **destroy**
  - Sirve para realizar operaciones de finalización de la aplicación para evitar corrupción (desconexión de la BD, eliminación de archivos, etc.)

Objeto Request

- Objeto que contiene la información relacionada con la petición que realiza el cliente
  - Parámetros del Formularios HTML.
  - Cookies.
  - Sesiones.
  - Tipo MIME de la petición
  - Información del Browser

Objeto Response

- Objeto que contiene la información que será enviada al cliente
  - Código HTML a enviar.
  - Cookies a setear.
  - Tipo MIME de la respuesta.

Objeto Response (2)

- El contenido se maneja mediante **PrintWriter**, que se obtiene con el método **getWriter()**
  - El contenido se agrega con el método **print()** o **println()** sobre el **PrintWriter**

Objeto Response (3)

- **PrintWriter** (cont)
  - Por ejemplo

```
public void doGet(HttpServletRequest
  request, HtttpServletResponse
  response) {
    ...
    PrintWriter out =
  response.getWriter();
    out.println("Hello world!);
    ...
}
```

Objeto Response (4)

- **PrintWriter** (cont)
  - Se debe ejecutar el método **close** de **PrintWriter** el terminar el método
    - Permite que se envíe efectivamente el contenido
    - Si no se ejecuta por lo general no se ve nada en el browser

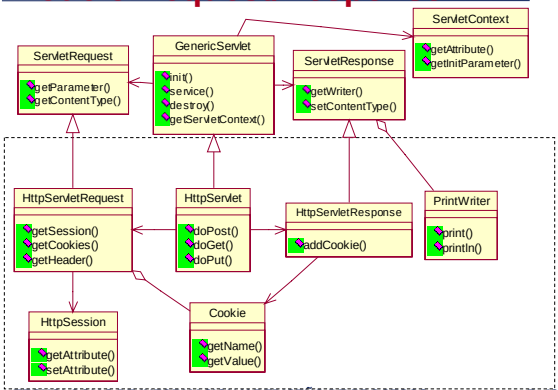
Objeto Request y Response

```
import java.io.*; import javax.servlet.*; import javax.servlet.http.*;
public class Hello extends HttpServlet {
  public void doGet(HttpServletRequest req, HttpServletResponse
    res)
      throws ServletException, IOException {
   res.setContentType("text/html");
    PrintWriter out = res.getWriter();
    String name = req.getParameter("name");
    out.println("<HTML>");
    out.println("<HEAD><TITLE>Hello, " + name +
   "</TITLE></HEAD>");
    out.println("<BODY>");
    out.println("Hello, " + name);
    out.println("</BODY></HTML>");
  }
}
```

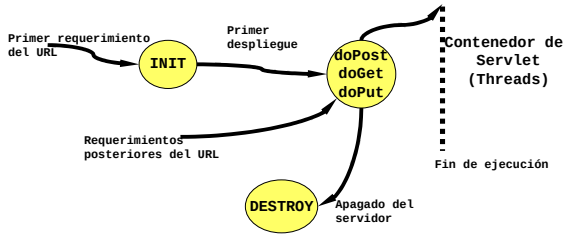

Manejo de Sesiones

- Se realiza mediante el objeto **HttpSesion**
  - Se obtiene del objeto **Request**, con el método **getSession()**
  - Se puede asignar cualquier objeto al objeto **HttpSesion**, que podrá ser visto por cualquier otro método de manejo de peticiones (mientras dure la sesión)

Modelo Request/Response



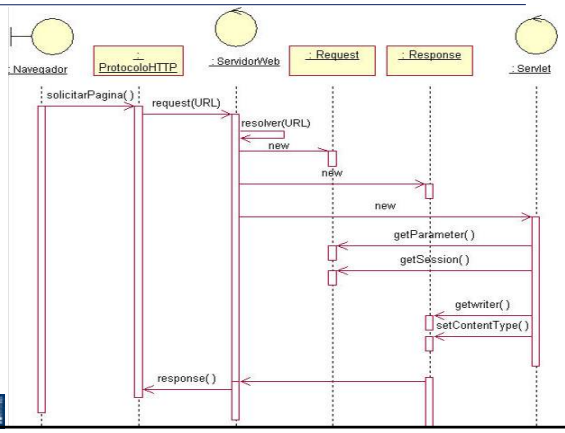
Ciclo de vida Servlet



TAREA

- Confeccione el diagrama de secuencia usando las clases de la API Servlet.

Servlet: El Proceso



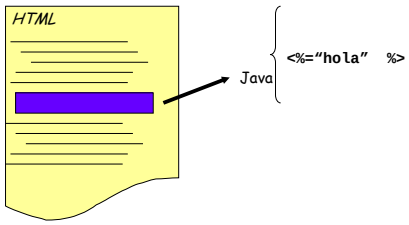
Solución Transaccional

- A través de los servlets es posible construir sistemas de alto rendimiento y en entornos transaccionales
- La posibilidad de manejar objetos comunes permite una máxima utilización de recursos, como por ejemplo conexiones a la Base de Datos

Solución Transaccional (2)

- Su esquema de manejo de procesos es muy superior a otras tecnologías desarrolladas para este tipo de aplicaciones
- Existe un gran soporte del mercado, de forma que existen interfaces para casi la totalidad de los sistemas comerciales que están en operación

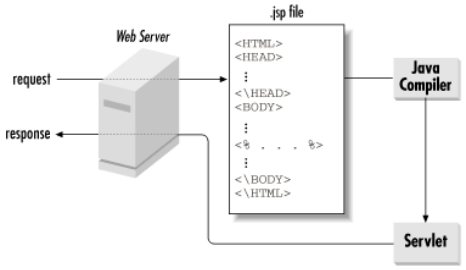
JSP: Java Server Page



JSP: Java como si fuera php

O una variante de incorporar a un servlet en el HTML

JSP: Java Server Page



JSP: Java Server Page

request: HttpServletRequest object
response: HttpServletResponse object
out: PrintWriter object
in:BufferedReader object

JSP: Java Server Page

```
<HTML>
  <HEAD>
  <TITLE>Hello</TITLE>
  </HEAD>
<BODY>
<H1>
<%
    if (request.getParameter("name") == null) {
%>
Hello World
<% } else { %>
Hello,
<%= request.getParameter("name") %>
<% } %>
</H1>
</BODY>
</HTML>
```

Preguntas típicas

- ¿cuando usar jsp vs servlet?
  - Jsp: su fuerte es el manejo de interfase dinámica en el servidor. Puede ser editado en dreamweaver por ejemplo.
  - Servlet: Lógica de negocio.
  - Jsp: Mas rápido de implementar, pero menos estructurado.
  - Servlet: Mas lento de implementar por la estructuración OO a realizar.