



UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

AUTOMATIZACION DEL INFORME DE LABORES PARA
LOS PROFESORES DE LA FMVZ VIA INTERNET.

T E S I S

QUE PARA OBTENER EL TITULO DE:
INGENIERO EN COMPUTACION
P R E S E N T A N :
MEJIA TREJO MARIA JUANA
RAMIREZ MONTERO LUIS



DIRECTOR: ING. JUAN MANUEL MARTINEZ VILLALOBOS

MEXICO, D. F.

JULIO DE 2002

TESIS CON
FALLA DE ORIGEN



Universidad Nacional
Autónoma de México



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

A mi padre que siempre nos
orientaste a crecer en la vida,
a enfrentarnos a la realidad y
lograr nuestras metas a base de trabajo.

A mi hermano porque a pesar
de los momentos difíciles nunca
dejaste de confiar y creer en mí,
además de ser un pilar en mi formación.

A mi hermana que con tu paciencia
y tu cariño, siempre me diste fuerzas
para continuar en la vida.

A ti hermana que después de nuestras
diferencias en la infancia, ahora adultas
me has dado grandes ejemplos.

A mi gran amor que con paciencia y
dulzura hemos podido superar barreras,
gracias a ti soy una mujer feliz.

Luis gracias por tu paciencia y tu buen
humor que me acorto los días.

Al Ing. Juan Manuel , que nos
dedico su tiempo, su amistad y
su sonrisa.

A la vida por darme la
oportunidad de haber
conocido a grandes
personas y continuar
disfrutando de su
presencia.

Maria Juana Mejía Trejo

A Dios, por el Sol,
la noche, la lluvia,
por el viento que
nos recuerda vivir libres
y libres morir,
por toda la fuerza,
el conocimiento y el mundo
tan bello que nos dio.

A mi madre,
gracias por luchar para que
mis hermanas y yo tuviéramos
un camino más claro,
por el amor incondicional
y su gran ayuda.

Para Haide,
gracias por la libertad de ser
yo mismo, aceptar y tolerar cada uno
de mis defectos, apreciar mis virtudes,
y regalarme un lugar muy especial en tu
corazón.

A la vida, por ser
tan bella.

Luis Ramírez Montero

Introducción	2
CAPÍTULO 1. Fundamentos Teóricos	
1. Introducción a las Bases de Datos	4
2. Modelo Entidad / Relación (E/R)	4
2.1 Entidad	4
2.2 Atributo	6
2.3 Dominio y Valor	6
2.4 Relación	6
2.5 Elementos de una relación	7
3. Modelo Relacional	8
3.1 Objetivos del Modelo Relacional	8
3.2 Objetos del Modelo Relacional.....	8
3.3 Formas Normales	9
4. Lenguaje Unificado de Modelado (UML)	11
4.1 Visión general de UML.....	11
4.2 Elementos de UML	11
4.3 Relaciones en UML	11
4.4 Diagramas UML.....	12
5. Extensión de UML para la WEB	14
5.1 Estereotipos de clase.....	14
5.2 Estereotipos de asociación	17
5.3 Estereotipos de atributo	18
5.4 Asociaciones válidas entre estereotipos.....	19
5.5 Estereotipos de componentes	19
6. Introducción a Internet / Intranet.....	22
6.1 Redes de computadoras	22
6.2 Protocolo TCP/IP	22
6.3 Arquitectura TCP/IP	23
7. Arquitectura Cliente / Servidor.....	25
7.1 Servidor	25
7.2 Cliente	25
7.3 Cliente / Servidor en el web	25
CAPÍTULO 2. Análisis	
1. Situación actual.....	28
2. El usuario y sus requerimientos.....	33
3. Propuesta de solución.....	33
4. Herramientas comerciales de desarrollo.....	34
4.1 Servidor Web.....	34
4.2 Sistemas Operativos.....	35

4.3 Lenguajes de programación.....	35
5. Análisis del sistema propuesto	35
5.1 Diagramas de casos de uso para el sistema	36
5.2 Diagramas de secuencia para el sistema	39
5.3 Diagramas de clases para el sistema.....	44
CAPÍTULO 3. Diseño	
1. Plataforma y tecnología seleccionada	48
1.1 Sistema Operativo	48
1.2 Servidor Web.....	49
1.3 Lenguajes de programación.....	50
1.4 Sistema de base de datos.....	52
2. Diseño de la base de datos	53
2.1 Modelo E/R.....	53
2.2 Modelo Relacional	55
2.3 Diccionario de datos	56
3. Diseño del sistema propuesto	59
3.1 Diagramas de secuencia.....	59
3.2 Diagramas de clases	67
CAPÍTULO 4. Desarrollo	
1. Desarrollo del sistema.....	74
1.1 Programación orientada a objetos y Java.....	74
1.2 Servlets y JSP	77
1.3 Programación del sistema propuesto	82
CAPÍTULO 5. Pruebas	
1. Pruebas del sistema.....	92
1.1 Diseño de casos de prueba	92
1.2 Categorías de pruebas	92
1.3 Consideraciones importantes para las pruebas.....	93
1.4 Pruebas realizadas.....	94
Conclusiones	96
Bibliografía.....	98
Anexos	
Anexo A: Manual de usuario	100

Introducción

Introducción

Todos los académicos pertenecientes a las dependencias de la UNAM tienen por obligación elaborar un informe de labores anual. En el caso de la Facultad de Medicina Veterinaria y Zootecnia (FMVZ) la dependencia encargada de reunir los Informes de Labores de la planta docente es la Unidad de Planeación.

El Informe de labores recolecta información detallada acerca de las actividades académicas de cada profesor como lo son: Datos Generales, Docencia Dentro de la UNAM, Docencia Fuera de la UNAM, Difusión e Investigación.

Los objetivos de este trabajo son:

- Acortar los tiempos de entrega del Informe; la recopilación de los informes de labores tarda aproximadamente de 3 a 4 meses.
- Disponibilidad; contar con una herramienta que permita al académico llevar un registro de sus actividades anuales.
- Generar reportes, tener la opción de que el académico genere reportes en el momento que lo requiera.
- Evitar traslados; la entrega y consulta del informe ha requerido acudir a la Unidad de Planeación.
- Organizar la información; mantener una organización , para facilitar el llenado del informe.

En el primer capítulo se definen los conceptos fundamentales para la elaboración del sistema, como lo son los elementos que componen las Bases de Datos, su definición, manipulación y modelado; Metodología Orientada a Objetos (UML) y su extensión en aplicaciones Web, Teoría de Redes (Protocolo TCP/IP), y el funcionamiento de Arquitecturas Cliente/Servidor.

En el segundo capítulo se hace un análisis del problema, que implica hacer un estudio de la situación actual y el entorno, conocer las necesidades del usuario final, así como del personal de la Unidad de Planeación. Se plantea una solución con respecto al análisis realizado y se determinan las herramientas necesarias para el desarrollo del sistema, al último, se realiza un análisis enfocado al sistema de Informe de labores utilizando para ello diagramas UML.

El tercer capítulo contiene el diseño del sistema con la tecnología seleccionada; el diseño de la Base de Datos y el diseño del sistema por medio de la Metodología UML.

El cuarto capítulo contiene el desarrollo del sistema que consta de un breve recorrido por el lenguaje de programación Java y sus herramientas necesarias para aplicaciones Cliente/Servidor, así como un bosquejo del código del Sistema.

El quinto capítulo trata sobre las pruebas realizadas al sistema.

Esperamos que el siguiente trabajo sirva de base para futuros proyectos enfocados a la creación de sistemas en Internet basados en una arquitectura cliente/servidor.

Fundamentos Teóricos

CAPÍTULO 1. Fundamentos Teóricos

1. Introducción a las Bases de Datos.

El almacenamiento, manipulación y recuperación de información en forma eficiente, es vital así como estratégica para cualquier organización. Las Bases de Datos juegan un rol crítico en casi todas las áreas donde las computadoras son usadas, incluyendo negocios, ingeniería, medicina, leyes, educación, etc.

Una Base de Datos (BD) es una colección de datos relacionados que representa un cierto modelo o abstracción del mundo real. Una base de datos es diseñada, construida y llenada con datos para un propósito específico. Tiene un grupo de usuarios particular, y algunas aplicaciones preestablecidas en las cuales estos usuarios están interesados.

El uso de las BD es contrario al enfoque tradicional, en que cada sistema maneja sus propios datos y archivos. Al usar BD, todos los datos se almacenan en forma integrada, y están sujetos a un control centralizado. Las diversas aplicaciones operan sobre este conjunto de datos.

Al usar BD, todos los datos se almacenan en forma integrada, y están sujetos a un control centralizado, teniéndose las siguientes ventajas:

- Evitar que los datos se repitan (redundancia)
- Evitar que distintas copias de un dato tengan valores distintos (inconsistencia)
- Evitar que usuarios no autorizados accedan a los datos (seguridad)
- Proteger los datos contra valores no permitidos (integridad o restricciones de consistencia)
- Permitir que uno o más usuarios puedan acceder simultáneamente a los datos (conurrencia)

2. Modelo Entidad/Relación (E/R)

El modelo Entidad-Relación (ER), también llamado *modelo conceptual*, es uno de los modelos de datos más populares. Se basa en una técnica de representación gráfica que incorpora información relativa a los datos y la relación existente entre ellos, para darnos una visión del mundo real. Este modelo se desarrolló para facilitar el diseño de las bases de datos, y fue presentado por Chen en 1976.

Las características del modelo entidad-relación son:

- Reflejan tan sólo la existencia de los datos, no lo que se hace con ellos.
- Se incluyen todos los datos del sistema en estudio y, por tanto no están orientado a aplicaciones particulares.
- Es independiente de las bases de datos y sistemas operativos concretos.
- No se tienen en cuenta restricciones de espacio, almacenamiento, ni tiempo de ejecución.
- Está abierto a la evolución del sistema.

2.1 Entidad

El principal concepto del modelo ER es la **entidad**, que es una "cosa" en el mundo real con existencia independiente. Una entidad puede ser un objeto físico (una persona, un auto, una casa o un empleado) o un objeto conceptual (una compañía, un puesto de trabajo o un curso universitario)

Existen dos clases de entidades: **Fuertes**, que son aquellas cuyos ejemplares tienen existencia por sí mismos, y **débiles**, en las cuales la existencia de un ejemplar depende de que existan un cierto ejemplar de otro tipo de entidad.

La representación gráfica de un tipo de entidad fuerte en éste modelo es un rectángulo etiquetado en cuyo interior está el nombre del tipo de entidad, Fig. 1.1.

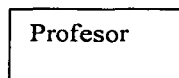


Fig. 1.1

Los tipos de entidad débil se representan con dos rectángulos concéntricos con un nombre en el interior, Fig. 1.2.



Fig. 1.2

2.2 Atributo

Cada entidad tiene propiedades específicas, llamadas **atributos**, que la describen. Por ejemplo, un salón de clases tiene un nombre, una ubicación, un cupo máximo, etc. Un atributo es representado gráficamente por una elipse (Fig. 1.3), los atributos aparecerán en el orden de la estructura a partir del vértice superior derecho de la entidad y según las manecillas del reloj.

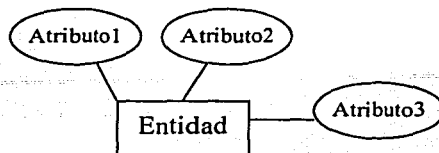


Fig. 1.3

2.3 Dominio y Valor

Los distintos atributos de un tipo de entidad toman **valores** para cada atributo. El conjunto de posibles valores que puede tomar un cierto atributo se denomina **dominio**, el que corresponde al tipo del atributo. Por ejemplo, "Folio" tiene como dominio al conjunto de los enteros positivos y "RFC" tiene como dominio al conjunto de enteros positivos en combinación con letras del abecedario.

2.4 Relación

Una **relación** se puede definir como una asociación, vinculación o correspondencia, sin existencia propia, entre varias entidades. Se representa mediante un rombo, etiquetado con el nombre de la relación, Fig. 1.4.



Fig. 1.4

2.5 Elementos de una relación

En un tipo de relación se pueden distinguir los siguientes elementos:

Nombre: Cada relación tiene un nombre que lo distingue unívocamente del resto, y ,mediante el cual ha de ser referenciado.

Grado: Es el número de entidades que participan en una relación.

Tipos de correspondencia: Representa la participación en la relación de cada una de las Entidades afectadas y existen tres tipos posibles:

1. **Relaciones Uno a Uno (1:1)** A cada ocurrencia de una Entidad corresponde no más de una ocurrencia de la otra y a la inversa.
2. **Relaciones Uno a Muchos (1:n)** A cada ocurrencia de la primera Entidad pueden corresponderle varias ocurrencias de la segunda y a cada ocurrencia de la segunda le corresponde no más de una de la primera.
3. **Relaciones Muchos a Muchos (n:m)** A cada ocurrencia de la primera Entidad pueden corresponderle más de una ocurrencia de la segunda y viceversa.

Para representar los tipos de relaciones se pone una etiqueta con 1:1, 1:n, n:m según corresponda al lado de la relación, o bien se oriente el arco de unión en el sentido 1 a n mediante una punta de flecha como aparece en la Fig. 1.5.

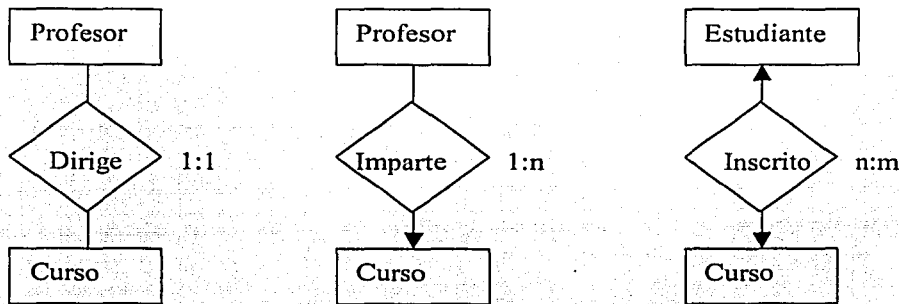


Fig. 1.5

Papel (Rol): Es la función que cada una de las entidades realiza en el tipo de relación, se representa poniendo el nombre del papel en el arco que une cada tipo de entidad con el tipo de relación, Fig. 1.6.

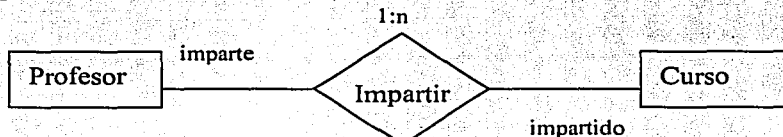


Fig. 1.6

3. Modelo Relacional

El Modelo Relacional fue definido en 1970 por E. F. Codd quien propone un modelo de datos basados en la teoría de las relaciones, en donde los datos se estructuran lógicamente en forma de relaciones - TABLAS -. El Modelo Relacional (MR) se propone como principal objetivo aislar al usuario de las estructuras físicas de los datos, finalidad perseguida desde los inicios de las bases de datos.

3.1 Objetivos del Modelo Relacional

- Independencia Física.
- Independencia lógica.
- Flexibilidad.
- Uniformidad.
- Sencillez.

Independencia Física:

Es decir, que el modo en que se almacenan los datos no influya en su manipulación lógica y, por tanto, los usuarios que acceden a esos datos no tengan que modificar sus programas por cambios en el almacenamiento físico.

Independencia Lógica:

Esto es, que el añadir, eliminar o modificar objetos de la base de datos no repercuta en los programas y / o usuarios que están accediendo a subconjuntos parciales de los mismos.

Flexibilidad:

En el sentido de poder presentar a cada usuario los datos de la forma en que éste prefiera.

Uniformidad:

Las estructuras lógicas de los datos presentan un aspecto uniforme, lo que facilita la concepción y manipulación de la base de datos por parte de los usuarios.

Sencillez:

Las características anteriores, así como unos lenguajes de usuario muy sencillos, producen como resultado que el modelo de datos relacional sea fácil de comprender y de utilizar por parte del usuario final.

Para cumplir los objetivos citados se introduce el concepto de **RELACION (TABLA)** como estructura básica del modelo.

Todos los datos de una base de datos se representan en forma de relaciones cuyo contenido varía en el tiempo. Formalmente una relación es un conjunto de filas en terminología relacional.

3.2 Objetos del Modelo Relacional

Existe una serie de objetos utilizados en el modelo relacional los cuales son:

Relación (Entidad): Conjunto de filas. Puede asociarse a lo que se conoce como tabla, con ciertas propiedades.

Tupla: Corresponde a una fila de esa tabla. Al número de tuplas se denomina cardinalidad.

Atributo: Se refiere a una columna de esa tabla. La cantidad de atributos determina el grado de la relación.

Dominio: Es el conjunto de donde los atributos toman sus valores. Puede suceder que dos atributos distintos de una misma relación tomen sus valores del mismo dominio.

Grado: Es el número de atributos o columnas que posee una relación.

Clave: Definimos como clave de una relación a aquél o aquellos atributos que nos determinan de forma unívoca y mínima a una tupla de esa relación.

3.3 Formas Normales

La Normalización se basa en que los datos son independientes de las aplicaciones que los gestionan, y su objetivo es obtener el mayor número de tablas posibles, dejando en cada una de ellas los atributos imprescindibles para representar a la entidad, o a la relación entre tablas mediante la conexión de sus claves.

Las ventajas que se obtienen tras la normalización son:

Facilidad de uso: Los datos están agrupados en tablas que identifican claramente un objeto o una relación.

Flexibilidad: La información que necesitan los usuarios se puede obtener de las tablas relacionales.

Precisión: Las relaciones entre las tablas consiguen mantener información diferente relacionada con toda exactitud.

Seguridad: Los controles de acceso para consultar o actualizar información son mucho más sencillos de implementar.

Facilidad de implementación: Las tablas se almacenan físicamente como archivos planos.

Independencia de los datos: Los programas no están ligados a las estructuras, con lo que se consigue aumentar la base de datos añadiendo nuevos atributos o nuevas tablas sin que afecten a los programas que las usa.

Claridad: La representación de la información es clara y sencilla para el usuario.

Facilidad de gestión: Los lenguajes manipulan la información de forma sencilla.

Mínima redundancia: La información no estará duplicada innecesariamente dentro de las estructuras.

Máximo rendimiento de las aplicaciones: Se trata de aquella información que va a servir de utilidad cada aplicación.

El proceso de normalización parte de las formas normales definidas por E. F. Codd (1970) creador de las bases de datos relacionales. Codd formuló tres formas normales (1FN, 2FN y 3FN).

Primera forma normal (1FN)

Una tabla se dice que esta en 1FN si y solo si los valores que componen el atributo de una tupla son atómicos. Es decir, en un atributo no deben aparecer valores repetitivos y por tanto tienen que ser elementales y únicos.

Segunda forma normal (2FN)

Una tabla se dice que esta en 2FN si y solo si cumple dos condiciones.

1. Se encuentra en 1FN.
2. Todo atributo secundario depende totalmente de la clave completa y no de una parte de ella.

Esta forma normal solo se considera si la clave principal es compuesta y, por tanto, esta formada por varios atributos. Si la clave principal de la tabla esta formada por un único atributo, entonces la tabla ya se encuentra en 2FN.

Tercera forma normal (3FN)

Una tabla de dice que esta en 3FN si y sólo si se cumplen dos condiciones:

1. Se encuentra en 2FN.
2. Un atributo secundario sólo se debe conocer a través de la clave principal o claves secundarias de la tabla y no por medio de otro atributo no primario.

4. Lenguaje Unificado de Modelado (UML)

4.1 Visión general de UML

El Lenguaje de Modelado Unificado (UML - Unified Modeling Language) es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reutilizables.

4.2 Elementos de UML

UML cuenta con cuatro elementos para describir las diferentes etapas de modelado de un sistema.

Elementos estructurales

Los elementos estructurales son los nombres de los modelos de UML. Estos incluyen clases, interfaces, colaboraciones, casos de uso, clases activas, componentes y nodos.

Elementos de comportamiento

Son las partes dinámicas de los modelos de UML. Estos incluyen diagramas de interacción y máquinas de estados.

Elementos de agrupación

Son las partes organizativas de los modelos de UML. Estos incluyen los paquetes.

Elementos de anotación

Son las partes explicativa de los modelos de UML. Estos incluyen las notas.

4.3 Relaciones en UML

Es una conexión entre elementos. En el modelado orientado a objetos, las tres relaciones más importantes son las dependencias, las generalizaciones y las asociaciones.

Dependencia

Una dependencia es una relación semántica entre dos elementos, en el cual un cambio a un elemento (el elemento independiente) puede afectar a la semántica del otro elemento (el elemento dependiente).

Asociación

Una asociación es una relación estructural que describe un conjunto de enlaces; un enlace es una conexión entre objeto.

Generalización

La generalización es una relación de especialización / generalización en la cual los objetos del elemento especializado (el hijo) pueden sustituir a los objetos del elemento general (el padre).

4.4 Diagramas UML

Un diagrama es la representación gráfica de un conjunto de elementos, visualizado la mayoría de las veces como un grafo conexo de nodos (elementos) y arcos (relaciones). Los diagramas se dibujan para visualizar un sistema desde diferentes perspectivas, de forma que un diagrama es una proyección de un sistema. Para todos los sistemas, excepto los más triviales, un diagrama representa una vista resumida de los elementos que constituyen un sistema. UML incluye nueve de estos diagramas:

1. Diagrama de clases.
2. Diagrama de objetos.
3. Diagrama de casos de uso.
4. Diagrama de secuencia.
5. Diagrama de colaboración.
6. Diagrama de estados (*statechart*)
7. Diagrama de actividades.
8. Diagrama de componentes.
9. Diagrama de despliegue.

Diagrama de Clases

Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Estos diagramas son los diagramas más comunes en el modelado de sistemas orientados a objetos. Los diagramas de clases cubren la vista de diseño estática de un sistema. Los diagramas de clases activas cubren la vista de procesos estática de un sistema.

Diagrama de Objetos

Un diagrama de objetos muestra un conjunto de objetos y sus relaciones. Los diagramas de objetos representan instancias de los elementos encontrados en los diagramas de clases. Estos diagramas cubren la vista de diseño estática o la vista de procesos estática de un sistema como lo hacen los diagramas de clases, pero desde la perspectiva de casos reales o prototípicos.

Diagrama de Estados

Muestra una máquina de estados, que consta de estados, transiciones, eventos y actividades. Los diagramas de estados cubren la vista dinámica de un sistema. Son especialmente importantes en el modelado del comportamiento de una interfaz, una clase o una colaboración y resaltan el comportamiento dirigido por eventos de un objeto, lo cual es especialmente útil en el modelado de sistemas reactivos.

Diagrama de Actividades

Es un tipo especial de diagrama de estados que muestra el flujo de actividades dentro del sistema. Los diagramas de actividades cubren la vista dinámica de un sistema. Son especialmente importantes al modelar el funcionamiento de un sistema y resaltan el flujo de control entre objetos.

Diagrama de Componentes

Muestra la organización y las dependencias entre un conjunto de componentes. Los diagramas de componentes cubren la vista de implementación estática de un sistema. Se relacionan con los diagramas de clases en que un componente corresponde, por lo común, con una o más clases, interfaces o colaboraciones.

Diagrama de Despliegue

Muestra la configuración de nodos de procesamiento en tiempo de ejecución y los componentes que residen en ellos. Los diagramas de despliegue cubren la vista de despliegue estática de una arquitectura. Se relacionan con los diagramas de componentes en que un nodo incluye, por lo común, uno o más componentes.

5. Extensión de UML para la WEB

5.1 Estereotipos de clase

Para representar los elementos particulares de las aplicaciones web, tenemos los siguientes estereotipos de clase:

- Server Page.
- Client Page.
- Form.
- Frameset.
- Target.
- JavaScript Object.

Server Page

Los Server Page son páginas web con *scripts* ejecutados por el servidor, los *scripts* interactúan con recursos del servidor como lo son: BD, lógica del negocio, sistemas externos, etc.

Las operaciones representan funciones en el *script* y los atributos representan variables visibles en el ámbito de la página, las cuales son accesibles por todas las funciones en la página.

La restricción de las Server Page, es que sólo pueden participar en relaciones con objetos del servidor.

Su representación gráfica se muestra en la figura 1.7

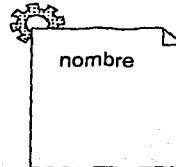


Fig.1.7 Server Page

Client Page

Una instancia de Client Page es una página Web con formato HTML, en donde se mezclan datos presentación y lógica. Esta página es mostrada por un navegador o browser y puede contener *scripts* interpretados por el browser.

Las operaciones representan funciones en el *script* en la página y los atributos representan variables del *script*, accesibles por toda función en la página. Pueden participar en asociaciones con <<Client Page>>'s o <<Server Page>>'s.

No tienen ninguna restricción y sus valores etiquetados son:

- *Title Tag*: Título de la página mostrado por el browser.
- *Base Tag*: URL base para de-referenciar URLs relativas.
- *BodyTag*: Conjunto de atributos que establece el fondo y texto por defecto.

Su representación grafica es la siguiente: Fig. 1.8

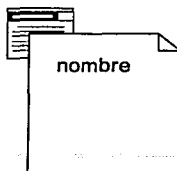


Fig. 1.8 Client Page

Form

Un Formulario es una colección de campos de entrada de datos, y forma parte de una <<Client Page>>.

Los atributos representan campos de entrada del form HTML y no tiene operaciones, ya que toda operación que interactúa con el form es una propiedad de la página que contiene form.

No tiene restricciones y los valores etiquetados del Form es:

SubmitMethod: método (GET o POST) utilizado para enviar los datos a la página servidor que los procesará.

Su representación gráfica se muestra a continuación: Fig. 1.9

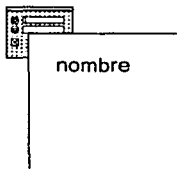


Fig. 1.9 Form

Frameset

Un Frameset es un contenedor de múltiples páginas web, se representa por un área visible dividida en rectángulos o frames, cada frame puede asociarse con un *target* y puede contener una página Web u otro Frameset.

Un Frameset es una <<Client Page>> por lo cual puede contener atributos y operaciones, pero las operaciones sólo pueden ser activadas por browsers que no soportan frames.

No tienen restricciones y sus valores etiquetados son los siguientes:

- *Rows*: cadena de alturas de fila separadas por comas
- *Cols*: anchuras de columna separadas por comas

Su representación gráfica se muestra en la Figura siguiente. Fig. 1.10

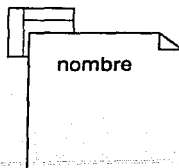


Fig. 1.10 Frameset

Target

Un Target es un compartimiento (con nombre) en una ventana de browser en donde puede mostrarse páginas web, puede ser un Frame en una ventana definido por un Frameset o una nueva instancia de browser o ventana.

Una asociación <<Targeted Link>> especifica un Target como el lugar en el que una página web ha de ser mostrada.

Las restricciones son que el nombre de un <<Target>> ha de ser único por cada cliente del sistema, sólo puede existir una instancia de un <<Target>> en el mismo cliente. No tiene valores etiquetados.

Su representación gráfica se muestra a continuación: Fig. 1.11

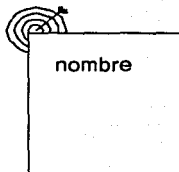


Fig. 1.11 Target

JavaScript Object

Un JavaScript Object es un objeto a medida (definido por el usuario) corresponde a la instancia de objeto genérico al que puede asociarse propiedades y funciones y puede ser utilizado por los scripts. Una instancia <<JavaScript Object>> solo existe en el contexto de <<Client Page>>'s

La JavaScript Object no tiene restricciones ni Valores etiquetados

Su representación se muestra a continuación: Fig. 1.12

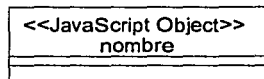


Fig. 1.12 JavaScript Object

5.2 Estereotipos de asociación

Los estereotipos de asociación se clasifican en:

- Link.
- Targeted Link.
- Frame Content.
- Submit.
- Builds.
- Redirect.

Link

Un Link es una asociación o vínculo entre una <<Client Page>> y otra <<Client Page>> o una <<Server Page>>

Sus valores etiquetados son:

- Parameters: lista de nombres de parámetros que deben ser pasados con la petición de la página ligada.

Targeted Link

La Targeted Link es un vínculo entre una página y otra que debe ser mostrada en un *target* determinado.

Sus valores etiquetados son:

- Parameters: igual que en <<Link>>
- TargetName: target en el que debe ser mostrada la página apuntada por el vínculo (destino).

Frame Content

El Frame Content es una agregación que expresa el contenido de un *frame*: página o *target*. Puede también apuntar a otro *frameset*, indicando *frames* anidadas.

Sus valores etiquetados son:

- Rows, Col: fila y columna concretas del frame en el frameset al que pertenece la página o target asociado.

Submit

Un Submit es una asociación entre <<Form>> y >>Server Page>>, los forms envían información al servidor web a través de <<Server Page>>'s para su procesamiento. Indica qué páginas pueden procesar el <<Form>> y los Forms conocidos por una >>server Page>>.

Los valores etiquetados de un Submit son:

- Parameters

Build

Es una asociación entre la <<Server Page>> y la <<Client Page>>. Identifica la <<Server Page>> responsable de la creación de una <<Client Page>>, su asociación es unidireccional.

Una <<Server Page>> puede generar muchas <<Client Page>>'s, pero una <<Client Page>> solo es creada por una <<Server Page>>. Carece de restricciones y valores etiquetados.

Redirect

Es una asociación unidireccional entre paginas WEB. Desde una <<Server Page>> a otra página, el procesamiento de la solicitud de página puede continuar con la página destino, la página destino puede o no participar en la construcción de una <<Client Page>>

5.3 Estereotipos de atributo

Existen dos tipos de atributos:

- Input Element.
- Select Element.

Input Element

Es el atributo de un objeto <<Form>> usado para la introducción de texto, carece de restricciones y sus valores etiquetados son:

- Type: Tipo de control de entrada (Text, Number, Password, Checkbox, Radio, Submit o Reset)
- Size: Tamaño de la área en pantalla, dado en caracteres.
- Maxlength: Máximo número de caracteres que el usuario puede introducir.
- Name: Nombre del elemento input.

Select Element

Control de entrada usado en un <<Form>>, permite al usuario seleccionar uno o más elementos de una lista. Se muestra en forma de un ComboBox o ListBox. Sus valores etiquetados son:

Size: Número de elementos mostrados a la vez.

Múltiple: Booleano que indica si se puede seleccionar más de un elemento a la vez.

Name: Nombre del elemento select.

5.4 Asociaciones válidas entre estereotipos.

La forma en que se asocian los diferentes estereotipos se muestra en la siguiente tabla 1.1:

	Client Page	Sever Page	Frameset	Target	Form
Client Page	Link Targeted Link Redirect	Link Targeted Link Redirect	Link Targeted Link Redirect	Dependency	Aggregation
Server Page	Build Redirect	Redirect	Redirect Build		
Frameset	Frame Content		Frame Content	Frame Content	
Target					
Form	Aggregated By	Submit			

Tabla 1.1. Asociaciones válidas entre estereotipos.

5.5 Estereotipos de componentes

Existen cuatro estereotipos de componentes:

- Web Page.
- ASP Page.
- JSP Page.
- Servlet.

Web Page

Descripción

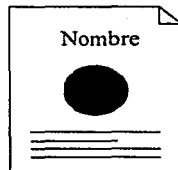
- El navegador puede solicitarla por su nombre
- Puede o no contener scripts de cliente o servidor

Puede ser: Archivo de texto accesible por el servidor Web, o Módulo compilado cargado e invocado por el servidor. Produce un documento HTML que se envía como respuesta a una petición del navegador cliente, Fig. 1.13

Restricciones: Ninguna

Valores etiquetados

Path : Path de la Web Page en el servidor Web



1.13 Web Page

ASP Page

Descripción:

- Sólo en entornos de aplicaciones basadas en Microsoft's Active Server Pages.
- Web Page que implementa código ASP del lado del servidor.

Restricciones: Ninguna.

Valores etiquetados : los de Web Page.

Fig. 1.14

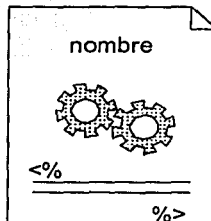


Fig. 1.14 ASP Page

JSP Page

Descripción:

- Sólo en entornos de desarrollo de aplicaciones que usen Java Server Pages.
- Web Page que implementa código JSP del lado del servidor.

Restricciones: Ninguna.

Valores etiquetados : los de Web Page.

Fig. 1.15

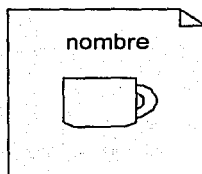


Fig. 1.15 JSP Page

Servlet**Descripción:**

- Sólo en entornos de desarrollo de aplicaciones que soporten servlets de Sun
- Componente de código Java que se ejecuta en el servidor

Restricciones: Ninguna

Valores etiquetados : los de Web Page

Fig. 1.16

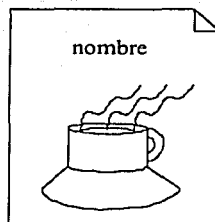


Fig. 1.16 Servlet

6. Introducción a Internet / Intranet

La red **Internet** es hoy día la red de computadoras más extensa del planeta. Para ser más precisos, **Internet** es una red que enlaza centenares de miles de redes locales heterogéneas.

En 1990, **Tim Berners-Lee**, un joven estudiante del Laboratorio Europeo de Física de Partículas (**CERN**) situado en Suiza, desarrolló un nuevo sistema de distribución de información en **Internet** basado en páginas **hipertexto**, al que denominó **World Wide Web**.

Realmente, el concepto de documento **hipertexto** no es nuevo: fue introducido por **Ted Nelson** en 1965 y básicamente se puede definir como *texto de recorrido no secuencial*, pulsando en las palabras con **enlaces** (links) se puede acceder al documento al que apuntan, que normalmente contiene una información más detallada sobre el concepto representado por las palabras del enlace.

6.1 Redes de computadoras

Una **red** es una **agrupación de computadores**. Mediante una red, se posibilita el intercambio de información entre computadoras de un modo eficiente y transparente. Una red permite ver los discos de otras computadoras como si fueran discos locales. Según sea la estructura de dicha agrupación, o según el número de computadoras integradas en ella se pueden establecer diferentes clasificaciones:

Red Local (LAN: Local Area Network)

De ordinario es una red dentro de un mismo edificio, como por ejemplo las redes de alumnos o de profesores de la FI.

Red de campus (CAN: Campus Area Network)

Es una red que une distintos edificios dentro de una zona geográfica limitada, por ejemplo el campus de una universidad. De ordinario todos los cables por los que circula la información son privados.

Red de ciudad (MAN: Metropolitan Area Network)

Se trata de una red que une distintos edificios dentro de un área urbana. En la transmisión de la información interviene ya una empresa de telecomunicaciones, que podría ser de ámbito local o regional.

Red de área extensa (WAN: Wide Area Network)

En este caso la red puede unir centros dispersos en una zona geográfica muy amplia, en ocasiones por todo el mundo. Es la red típica de las empresas multinacionales.

Hay que mencionar la **jerarquía** y **estructuración** existente en las redes: unas computadoras poseen unos derechos que otras no poseen (tienen accesos a archivos a los que otras no pueden acceder, las computadoras con más jerarquía pueden controlar a las de menor rango, etc.)

6.2 Protocolo TCP/IP

Lo que permite que computadoras remotas con procesadores y sistemas operativos diferentes se entiendan y en definitiva que **Internet** funcione como lo hace en la actualidad, es un conjunto de instrucciones o reglas conocidas con el nombre de **protocolo**. La **Internet** utiliza varios protocolos, pero los que están en la base de todos los demás son el **Transport Control Protocol (TCP)** y el llamado **Internet Protocol (IP)**, o en definitiva **TCP/IP** para abreviar. Se trata de una serie de reglas para mover de una computadora a otra los datos electrónicos descompuestos en **paquetes**, asegurándose de que todos los paquetes lleguen y sean ensamblados correctamente en su destino. La mayoría de las computadoras en **Internet** utilizan el protocolo **TCP/IP**, y gracias a ello se consigue eliminar la barrera de la heterogeneidad de las computadoras y resolver los problemas de direccionamiento.

6.3 Arquitectura TCP/IP.

Este tipo de arquitectura se ha impuesto en los años 90, contrariamente a lo que se opinaba que ocurriría en los 80, cuando se creía que el modelo OSI sería el que finalmente triunfaría. TCP/IP ha llegado a convertirse en un estándar de facto en redes, de hecho, es la familia de protocolos de comunicaciones empleada por Internet. TCP. Estos protocolos se crearon y normalizaron mucho antes de que se definiera el modelo de referencia OSI de la ISO.

No existe un modelo oficial de protocolos TCP/IP, al contrario que en OSI. Los protocolos se han ido definiendo anárquicamente, y a posteriori han sido englobados en capas. En el modelo TCP/IP no es estrictamente necesario el uso de todas las capas sino que, por ejemplo, hay protocolos de aplicación que operan directamente sobre IP y otros que lo hacen por encima de IP. En la figura 1.17 se pueden apreciar los 5 niveles de la arquitectura, comparados con los siete de OSI.

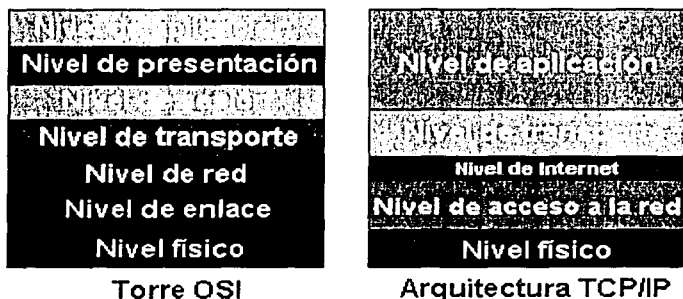


Figura 1.17. Comparación entre modelo OSI y TCP/IP.

Nivel físico (Nivel 1)

Coincide aproximadamente con el nivel físico de OSI. Define las características del medio, su naturaleza, el tipo de señales, la velocidad de transmisión, la codificación, etcétera.

Nivel de acceso a la red (Nivel 2)

Comprende el nivel de enlace y buena parte del nivel de red de OSI. Es el nivel responsable del intercambio de datos entre dos sistemas conectados a una misma red. Controla la interfaz entre un sistema final y una subred.

Nivel de Internet (Nivel 3)

Comprende el resto del nivel 3 de OSI no incluido en el nivel de acceso a la red. Se encarga de conectar equipos que están en redes diferentes. Permite que los datos atraviesen distintas redes interconectadas desde un origen hasta un destino. El principal protocolo utilizado es IP (Internet Protocol)

Nivel de transporte (Nivel 4)

Incluye el nivel 4 y parte del nivel 5 de OSI. Proporciona transferencia de datos extremo a extremo, asegurando que los datos llegan en el mismo orden en que han sido enviados, y sin errores. Esta capa puede incluir mecanismos de seguridad. Los principales protocolos utilizados son TCP y UDP.

Nivel de aplicaciones (Nivel 5)

Proporciona una comunicación entre procesos o aplicaciones en computadores distintos. Además de las aplicaciones, este nivel se ocupa de las posibles necesidades de presentación y de sesión. Los protocolos más utilizados con TCP en el nivel 4 son: TELNET, FTP, HTTP y SMTP, sobre el que a su vez se apoya MIME. Y el más utilizado con UDP en el nivel 4 es SNMP.

7. Arquitectura Cliente / Servidor

Los sistemas basados en la arquitectura cliente/servidor están formados por dos partes lógicas: un **servidor** que proporciona servicios, y un **cliente** que solicita servicios del servidor o servidores. Los dos, juntos, forman un sistema de computación completo con una clara división de responsabilidades, Fig. 1.18.



Fig. 1.18 Arquitectura cliente/servidor.

7.1 Servidor

Un **servidor** es un proceso que contesta a la solicitud del cliente realizando la tarea propuesta por éste. Los clientes gestionan recursos compartidos como archivos, impresoras, enlaces de comunicación, bases de datos, etc.

7.2 Cliente

Un **cliente** es un proceso que envía un mensaje a un proceso servidor, solicitando que realice determinada tarea. Los procesos cliente normalmente gestionan la porción de interfaz de usuario de la aplicación, validan los datos introducidos por el usuario, realizan las solicitudes a los servidores y, a veces, ejecutan cierta lógica de negocio.

7.3 Cliente / Servidor en el Web

La arquitectura de Internet está basada en el modelo cliente-servidor, utilizando un sistema de **hipertexto** global, el cual permite enlazar documentos con otros documentos afines en el mismo servidor o a lo largo de toda la red, en la figura 1.19 se muestra la operación conjunta de las piezas de cliente y servidor en el web.

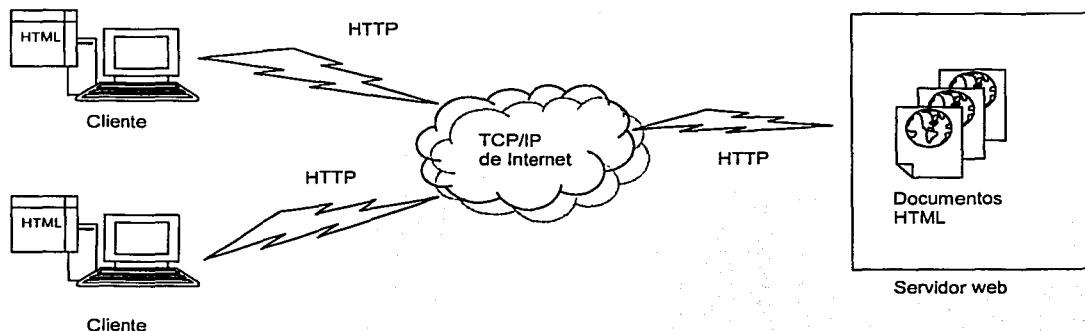


Fig. 1.19 Cliente / Servidor en el Web.

Análisis

CAPÍTULO 2. Análisis

El análisis del sistema define el papel de cada elemento de un sistema informático, asignando finalmente al software el papel que va a desempeñar. Durante este proceso se trata de identificar qué información ha de ser procesada, qué función y rendimiento se desea, qué interfaces han de establecerse, qué función y rendimiento se desea, qué interfaces han de establecerse, qué restricciones de diseño existen y qué criterios de validación se necesitan para definir un sistema correcto. Por tanto, han de identificarse los requisitos clave del sistema, pues una vez que se han definido, se procesará a identificar las entradas, salidas y archivos del mismo; así como el flujo de los datos, los diferentes procesos de transformación y los principales elementos de datos que se van a manejar en el sistema.

1. Situación actual

En la Universidad Nacional Autónoma de México se realiza anualmente el Informe de Labores por parte del personal académico de carrera, el cual deberá someter oportunamente a la consideración del consejo de la dependencia de su adscripción, el proyecto de las actividades de investigación, preparación, estudio y evaluación del curso o cursos que impartan, dirección de tesis o prácticas, aplicación de exámenes, dictado de cursillos y conferencias y demás que pretenda realizar durante el año siguiente; llevar a cabo y rendir en su oportunidad un informe sobre la realización de las mismas. Dicho proyecto constituirá su informe anual de labores una vez que sea aprobado por el consejo técnico, interno o asesor.

En la Facultad de Medicina Veterinaria y Zootecnia (FMVZ) como en otras dependencias se elabora el informe de labores anualmente, establecido en el Estatuto Universitario del Personal Académico de la UNAM capítulo VII sección B artículo 56 inciso (b). El departamento encargado de esto en la FMVZ es la Unidad de Planeación.

La Unidad de Planeación se creó con el fin de establecer una cultura de planeación, evaluación y rendición de cuentas a la sociedad, dentro de cuya estructura se ubican los siguientes departamentos:

Desarrollo institucional, cuyo objetivo es coordinar y apoyar un proceso de planeación estratégica que permita a la facultad transformarse a sí misma para adaptarse a las circunstancias cambiantes de su entorno.

Estadística y Sistemas de Información, cuyo objetivo es acopiar y analizar la información requerida para obtener indicadores de diagnóstico útiles en las actividades de evaluación, planeación y toma de decisiones.

Evaluación educativa, cuyo fin es establecer, fortalecer y coordinar el proceso de evaluación en la facultad, analizando las acciones previstas en la planeación y midiendo resultados, para que a través de estrategias adecuadas se efectúen las modificaciones y correcciones necesarias durante la elaboración, ejecución, desarrollo de los programas y proyectos educativos.

Departamento de Vinculación y Transferencia Tecnológica, que pretende impactar favorablemente, a través de la participación de académicos y alumnos, en el desarrollo de las organizaciones y empresas de los sectores primario, secundario, y de servicio, beneficiando al mismo tiempo a la facultad mediante el acceso a nuevas posibilidades de formación de personal, enseñanza práctica e investigación.

En los últimos años, el informe de labores se ha venido realizando por medio de un programa elaborado en el lenguaje de programación Visual Basic versión 3.0, su pantalla principal se muestra en la Fig. 2.1.

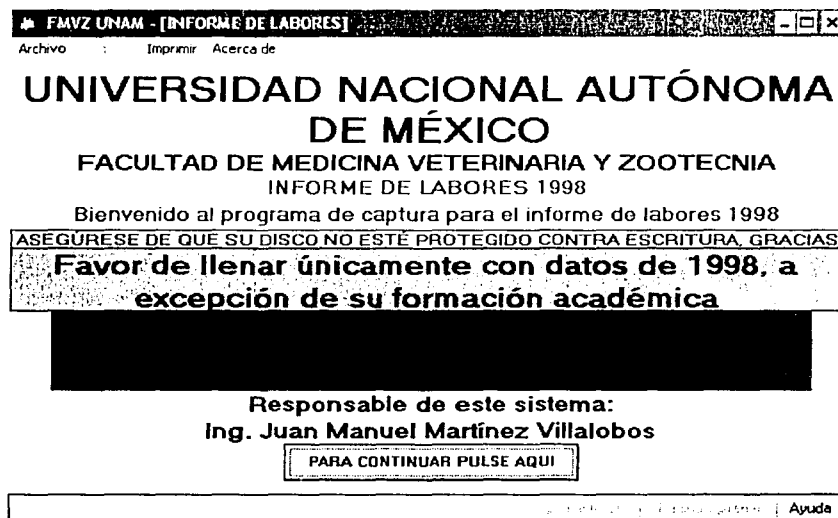


Fig. 2.1 Pantalla Principal.

El programa es entregado en discos flexibles de 3 ½ a cada profesor a través de su departamento de adscripción, cada profesor realiza un vaciado de información en dicho programa, al término, éste regresa el disco a su departamento, el cual a su vez hace entrega de los discos a la Unidad de Planeación, como la base de datos no cabe en un disco flexible, toda la información es almacenada en archivos de texto, por lo cual la Unidad de planeación gestiona la información a través de un programa especial para obtener los datos almacenados en el archivo de texto, para que los datos puedan ser vaciados a la base de datos, de forma esquemática Fig. 2.2.

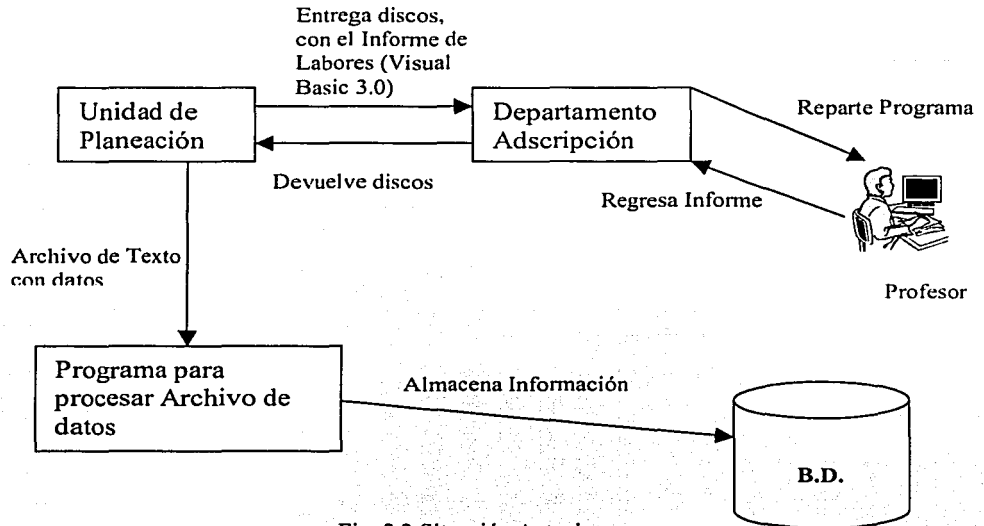


Fig. 2.2 Situación Actual.

Entre las desventajas que presenta esta forma de distribuir el Informe de Labores se encuentran las siguientes:

- Elaboración de aproximadamente 500 copias del programa para recopilar la información.
- Envío de discos a cada Departamento de adscripción.
- En ocasiones resulta que el profesor no se encuentra físicamente en ese departamento, por lo tanto debe realizarse una búsqueda de él.
- Uso inapropiado del programa. Falta de capacitación o manual para uso del programa, lo cual acarrea problemas como querer abrir el programa en un procesador de textos u otro programa inadecuado, sin realizar la ejecución directa del programa.
- Pérdidas y daños de discos. El disco puede dañarse por el uso, acarrear,do la pérdida total de la información, ocasionando que el profesor acuda a la Unidad de Planeación para que le sea entregada una nueva copia, retardando la entrega de ésta.
- Dificultades en la recopilación de discos.
- Entrega inoportuna de discos.
- Vaciado de la información de cada disco en la base de datos; como la base de datos no cabe en un disco flexible, la información es almacenada en archivos de texto, posteriormente cada archivo es procesado por otro programa para obtener la información y pasarla a la base de datos. En ocasiones los archivos de texto no podían ser abiertos, lo cual implica que debería ser llenado el informe nuevamente, acarreando pérdida de tiempo.
- Limitación en cuanto a la cantidad de información que se puede almacenar en un disco flexible.
- Si el profesor desea obtener un listado del informe correspondiente al año 1999 por ejemplo, debe acudir a la Unidad de Planeación de la FMVZ para que se le proporcione esa información.
- Si por alguna razón alguien olvida agregar cierta información y ya había entregado su disco, tiene que acudir a la Unidad de Planeación, pedir su disco y agregar los datos faltantes.
- La forma en que esta diseñado el sistema actual, no permite una interacción directa con la base de datos.

El sistema actual abarca los siguientes rubros, desglosados de la siguiente forma:

Datos Generales:

Esta parte abarca: datos personales, domicilio particular y de oficina, así como datos referentes a su nacionalidad actual.

Formación Académica:

Este rubro abarca información acerca de su trayectoria académica.

Formación Académica (Cursos):

Información acerca de actualización didáctica y educación continua.

Formación Académica (Premios y Distinciones):

Este rubro corresponde a premios, distinciones profesionales o científicas e inscripciones a asociaciones académicas o profesionales.

Experiencia Profesional Fuera de la UNAM:

Corresponde a la experiencia ya sea a la experiencia en el sector público, privado o ejercicio privado de la profesión.

Actividades Docentes en la UNAM(Profesor):

Indica en orden cronológico las asignaturas impartidas en la UNAM.

Actividades Docentes en la UNAM (Profesor de Carrera):
Son todas las categorías que ha ocupado como profesor de carrera.

Actividades Docentes en la UNAM (Ayudante de Profesor):
Abarca todas las categorías que ha ocupado como ayudante de profesor.

Actividades Docentes en la UNAM (Técnico Académico):

Actividades Docentes en la UNAM (Labores de servicios dentro de la UNAM): Corresponde a las labores de servicio realizados como profesores de medio tiempo y tiempo completo dentro de la UNAM.

Actividades Docentes en la UNAM (Dirección de Tesis): Contiene información acerca de las tesis en que ha participado ya sea como: asesor, coasesor, etc.

Actividades Docentes fuera de la UNAM (Profesor): Denota el orden cronológico de las asignaturas impartidas fuera de la UNAM.

Actividades Docentes fuera de la UNAM (Profesor de Carrera): Comprende las categorías que ha ocupado como profesor de carrera.

Actividades Docentes fuera de la UNAM (Ayudante de Profesor): Señala las categorías que ha ocupado como ayudante de profesor.

Actividades Docentes fuera de la UNAM (Técnico Académico):
Actividades Docentes fuera de la UNAM (Labores de Servicio) Labores de Servicio realizados como profesores de medio tiempo y tiempo completo fuera de la UNAM.

Actividades Docentes fuera de la UNAM (Dirección de Tesis): Contiene información acerca de las tesis en que ha participado ya sea como: asesor, coasesor, etc.

Actividades Académico - Administrativas (Puestos): Abarca los puestos que ha ocupado ya sea por elección o designación.

Actividades Académico - Administrativas (Comisiones y Participaciones): Se refiere a las comisiones de H. Consejo Técnico de la FMVZ y del Consejo Universitario de la UNAM, así como la participación en organismos Nacionales e Internacionales.

Difusión: Incluye la participación en conferencias, cursos, clases por televisión, organización o coordinación de cursos o seminarios, etc.

Asistencia a Congresos y Reuniones: Son todas las asistencias a congresos científicos o reuniones similares, incluye asistencia a reuniones científicas culturales de carácter internacional o nacional, en comisión o representación de la UNAM u otra institución.

Proyectos. Información sobre proyectos que desea realizar.

Exámenes: Comprende exámenes extraordinarios y de oposición.

Participación en Comités: Refiere a los comités en donde ha participado.
Comisión, Sabático y Licencia. Señala si durante el periodo gozó de comisión, sabático, licencia o ninguno.

2. El usuario y sus requerimientos

Como usuario principal del Informe de Labores esta el personal académico de carrera perteneciente a la FMVZ.

Dentro de los requerimientos generales proporcionados por el usuario tenemos:

- Acceso al informe a cualquier hora del día y sin importar su ubicación.
- No tener problemas en el tamaño de la información.
- Poder generar un currículo general, donde venga incluida información de varios años, o de un año en especial, en ambos casos ordenada por rubros.
- Reportes con formato agradable.
- Claridad en la Información que se pide.
- Debe ser amigable y contener una ayuda que describa el llenado del informe.
- Poder manipular la información, ya sea modificar, actualizar, borrar o agregar nuevos registros sin esperar hasta fin de año, época en que se reparte el informe de labores en discos.
- Organización de la información por rubros bien definidos.
- Que no se pierda la información.
- Fácil acceso al informe.
- Que nadie más pueda acceder a mi información.

3. Propuesta de solución

Dadas las condiciones y necesidades expuestas, la naturaleza del sistema para el Informe de Labores será de un sistema con soporte para acceso y transferencia de información a través de protocolos de red y soporte para múltiples usuarios.

El sistema residirá en forma centralizada para asegurar la integridad de los datos y evitar su duplicidad.

Con base en los argumentos anteriores consideramos para ello utilizar la red Internet para una comunicación bidireccional utilizando un protocolo TCP/IP haciendo uso de HTTP para mostrar la información, por lo cual es necesario de un servidor web, de forma esquemática lo podemos representar en la Fig. 2.3.

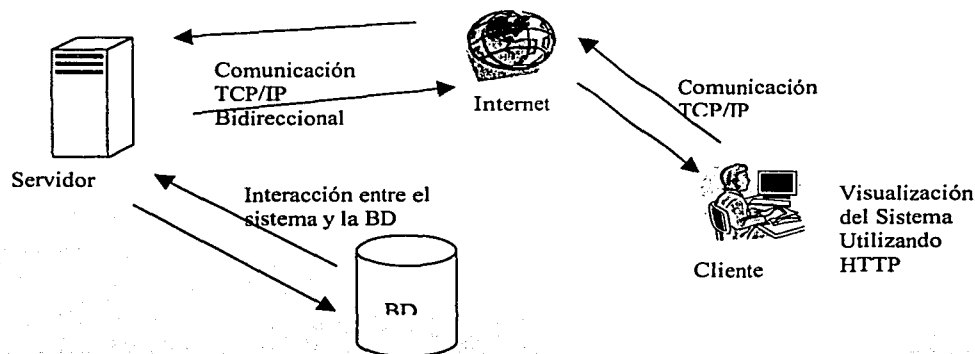


Fig. 2.3 Propuesta de solución.

El sistema mantiene una comunicación constante con la base de datos, por lo cual es necesario restringir el acceso solamente a los profesores que conforman la planta docente de la FMVZ, de forma esquemática, Fig. 2.4.

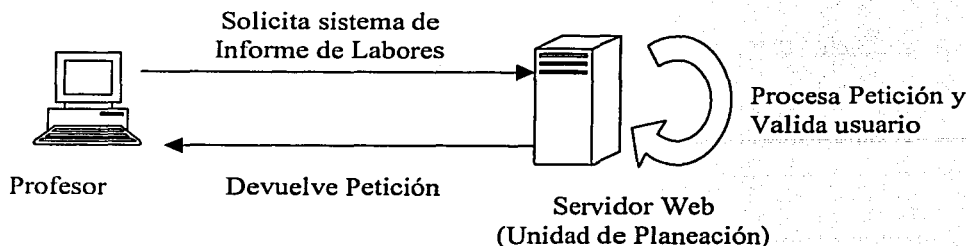


Fig. 2.4 Restricción de acceso.

4. Herramientas comerciales de desarrollo

Un sitio web es un novedoso medio de comunicación cuya estructura técnica se basa en millones de computadoras interconectadas entre sí en todo el mundo (las 24 horas del día, los 365 días del año) que ofrecen todo tipo de información. Por este medio los usuarios pueden comunicarse a los distintos aspectos del proyecto.

Algunos beneficios de tener un sitio Web son:

- Facilidad de acceso y mayor velocidad de comunicación.
- Interactividad y retroalimentación para mejorar la calidad de comunicación.
- Comunicación directa con el usuario.
- Mejor y mayor distribución de la información.

La programación para crear un ambiente web que pueda ser accedido desde Internet implica lo siguiente:

4.1 Servidor Web

Un servidor web se compone de Hardware y Software. El funcionamiento principal de un servidor web es copiar la gran cantidad (generalmente pequeños) de archivos que componen una página web del disco a la red de equipos lo más rápido posible para los numerosos usuarios simultáneos. Una misión secundaria es ejecutar los programas para los usuarios individuales y entregar el resultado tan rápido como sea posible. A continuación se muestra los servidores web más usados en Internet: Tabla 2.1

Servidor	Sistema Operativo
Apache	NetBSD, Digital UNIX, OS/2, SCO, HPUX, Novell NetWare, Macintosh, Windows NT, Linux, AS/400, FreeBSD, IRIX, Solaris
Internet Information Server	Windows NT
Java Web Server	OS/2, HPUX, Windows NT, Linux, Windows 95, IRIX, Solaris
Netscape Enterprise Server	Digital UNIX, AIX, HPUX, Windows NT, IRIX

Tabla 2.1 Servidores Web.

4.2 Sistemas Operativos

El sistema operativo es el componente de software que, en un sistema electrónico, administra la utilización de los recursos disponibles y provee la base sobre la cual operan los diversos servicios al usuario, desde utilerías para la administración del sistema hasta las aplicaciones más complejas, Tabla 2.2.

Sistema Operativo
UNIX
Windows NT/2000
Linux
Windows 95/98/ME
Macintosh

Tabla 2.2 Sistemas Operativos.

4.3 Lenguajes de Programación

Existe una gran variedad de lenguajes que se utilizan para el desarrollo de la parte funcional de un sitio web, la programación CGI (Interfaz de Compuerta común) se define como una capa intermedia entre una petición proveniente de un navegador Web u otro cliente HTTP, y las bases de datos o aplicaciones del servidor HTTP. Su finalidad es la de:

- Leer los datos enviados por el usuario.
- Buscar información respecto a la petición que esté incrustada en la petición HTTP.
- Generar resultados.
- Dar formato a los resultados dentro de un documento.
- Establecer los parámetros HTTP de respuesta adecuados.
- Devolver el documento al cliente.

Para realizar estos propósitos se utilizan varios lenguajes de programación como pueden ser, Tabla 2.3.

Lenguajes de programación
PHP
Java (Servlets)
Java (JSP)
ASP
Perl

Tabla 2.3 Lenguajes.

5. Análisis del sistema propuesto

El sistema de Informe de labores debe estar enfocado a cumplir con las expectativas del usuario para actualizar y consultar la información del modo en que lo requiera.

Para evitar equivocación en cuanto al tipo de información que el sistema pide y considerando los errores que el usuario cometía durante la interacción con el sistema anterior, es necesario incluir una ayuda que proporcione información sobre el uso del sistema, así como del tipo de información que se pide para cada rubro.

El sistema esta compuesto por los siguientes elementos:

- Usuarios.
- Sistema de Informe de Labores.
- Base de datos.

Usuarios

Son todas las entidades que interactúan de forma directa o indirecta con la base de datos o con el sistema en sí.

Sistema de Informe de Labores

Es el sistema de cómputo que interactúa con la base de datos.

Base de datos

Almacena la información que se maneja en un Informe de Labores.

5.1 Diagramas de caso de uso para el sistema

Sirven para modelar el comportamiento deseado del sistema, permite ver el sistema entero como una caja negra; se puede ver qué fuera del sistema y cómo reacciona a los elementos externos, pero no se puede ver cómo funciona por dentro.

Elementos

Casos de uso: Un caso de uso es, en esencia, una interacción típica entre un usuario y un sistema de computo, el caso de uso capta alguna función visible para el usuario, logrando un objetivo.

Actor: Llamamos así al usuario, cuando desempeña un papel con respecto al sistema.

Relación: La forma en que interactúa ya sea un caso de uso con otro o con un Actor.

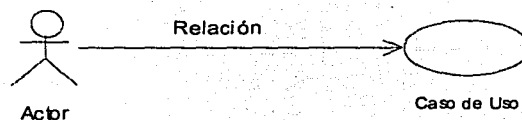


Fig. 2.5 Diagrama de caso de uso.

Dentro de los casos de uso para el sistema tenemos:

- Ver información existente.
- Actualizar información.
- Borrar información.
- Agregar información.
- Crear reporte ya sea en pantalla o en un archivo.
- Imprimir reporte.

A continuación se muestran los diagramas de caso de uso para el sistema propuesto.

Ver información existente.

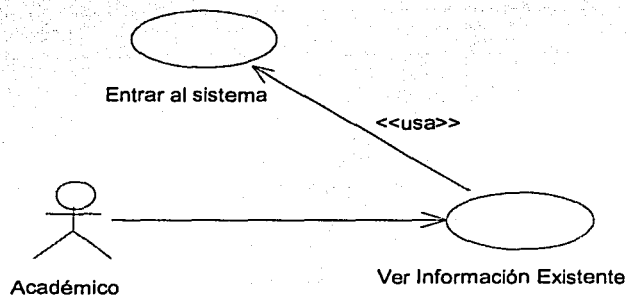


Fig. 2.6 Ver información existente

Actualizar información.

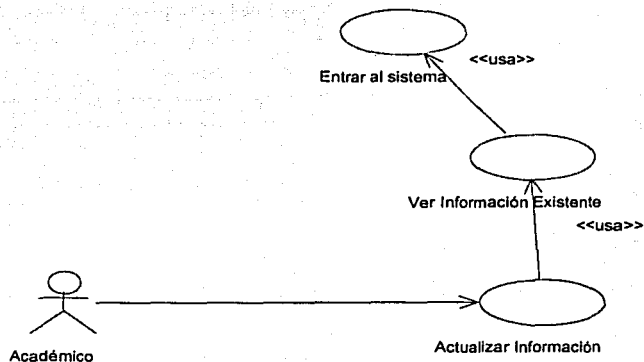


Fig. 2.7 Actualizar información

Borrar información.

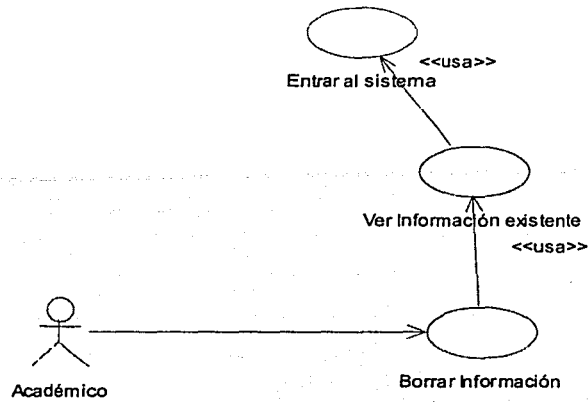


Fig. 2.8 Borrar información

Agregar información.

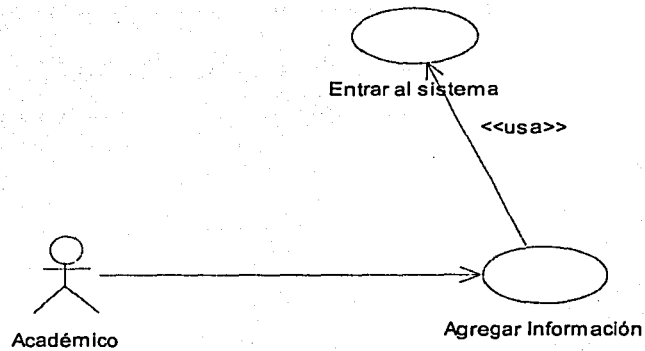


Fig. 2.9 Agregar Información

5.2 Diagramas de secuencia para el sistema

Un diagrama de secuencia destaca la ordenación temporal de los mensajes. Como se muestra en la Fig. 2.10. Normalmente, los diagramas de interacción contienen:

- Objetos.
- Enlaces.
- Mensajes.

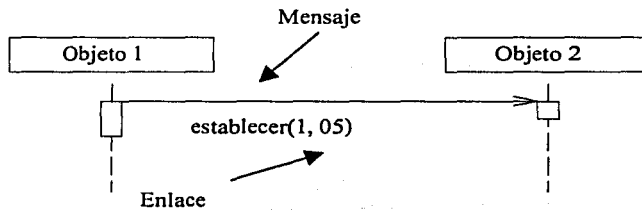


Fig. 2.10 Diagrama de secuencia.

Para que el usuario principal pueda ingresar al sistema es necesario que sea validado para controlar el acceso al sistema, a continuación se muestra los pasos que debe seguir un usuario para acceder al sistema, Fig. 2.11

Acceso Autorizado al Sistema

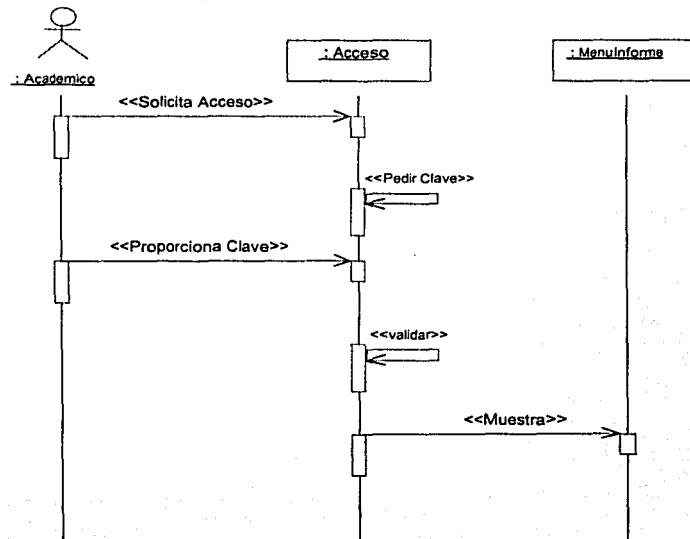


Fig. 2.11 Acceso Autorizado.

TFCIS CON FALLA DE ORIGEN

Puede darse el caso que la petición del usuario sea no aceptada por lo cual se le manda un mensaje de acceso denegado, Fig. 2.12.

Acceso No Autorizado al Sistema

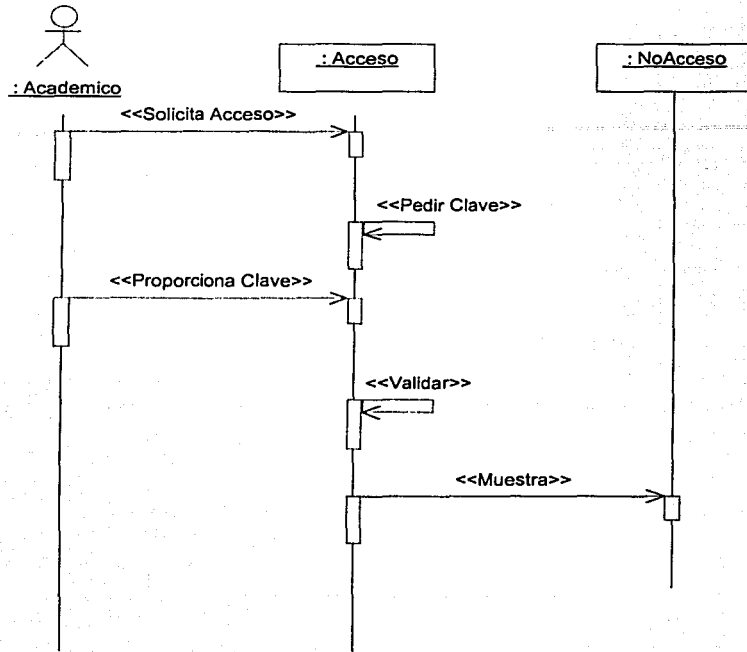


Fig. 2.12 Acceso no Autorizado.

Ya que el usuario esta dentro del sistema tiene la opción de ver la información existente y poder modificarla, esto es, actualizar o borrar contando con la opción de agregar nueva información. Todas estas acciones se realizan seleccionando un rubro en especial desde la Ventana de Menú Principal, Fig. 2.13.

Actualizar Información

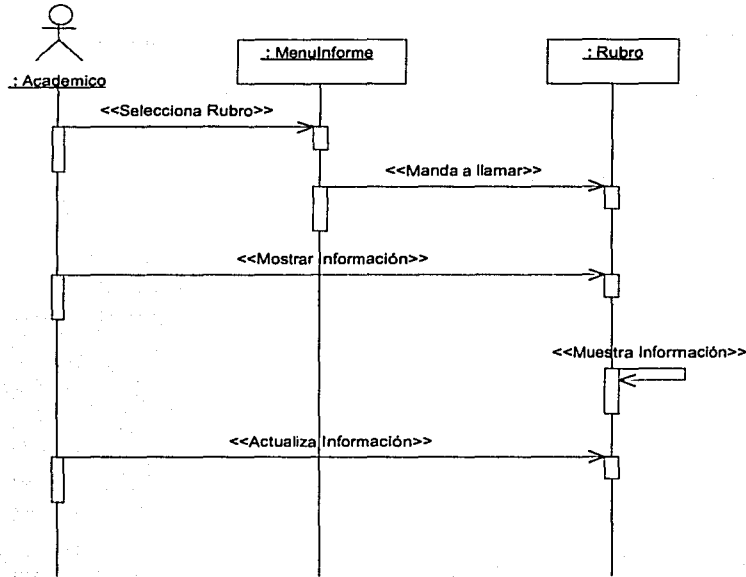


Fig. 2.13 Actualizar Información.

Borrar Información

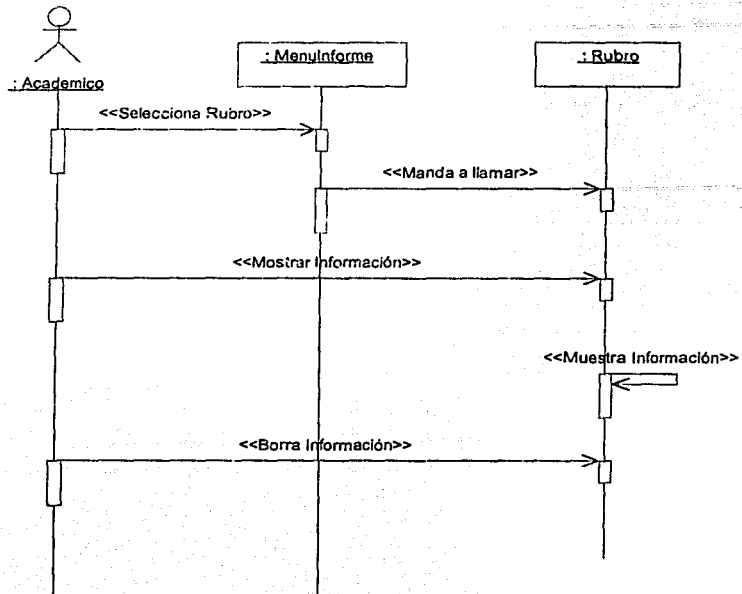


Fig. 2.14 Borrar Información.

Agregar Información

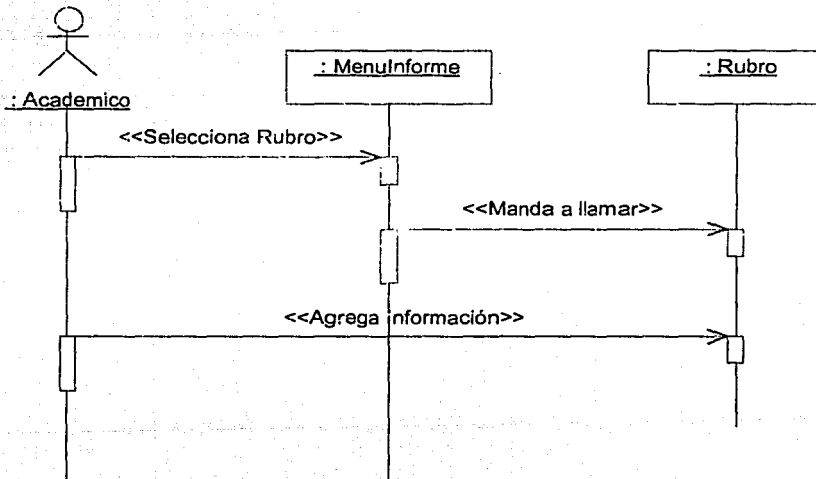


Fig. 2.15 Agregar Información.

Como parte del sistema existe la necesidad de generar reportes como lo especificamos en los casos de usos, Fig. 2.16.

Crear Reporte

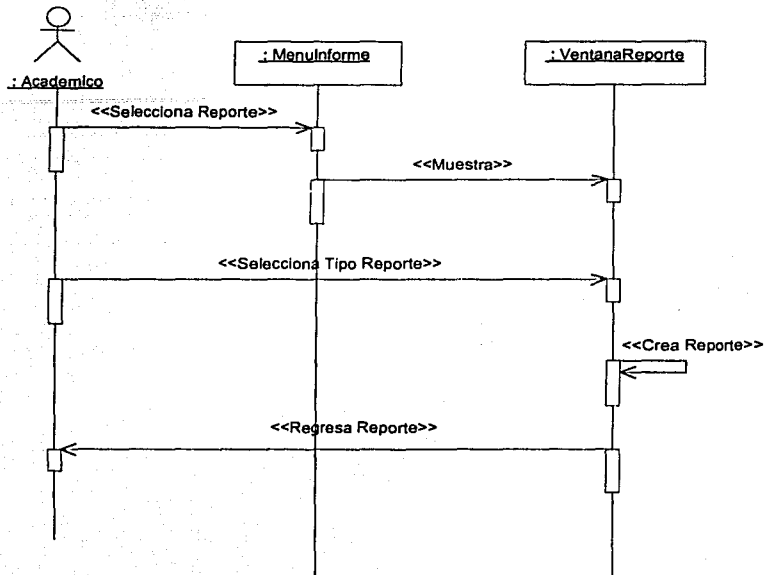


Fig. 2.16 Crear Reporte.

5.3 Diagramas de clases para el sistema

Un diagrama de clases describe los tipos de objetos que hay en el sistema y las diversas clases de relaciones que existen entre ellos. Hay dos tipos principales de relaciones:

- **Asociación:** Como se relacionan las clases.
- **Subtipo:** Cuando una clase se deriva de otra.

Las partes más importantes de una clase son: nombre, sus atributos y sus operaciones, Fig. 2.17.

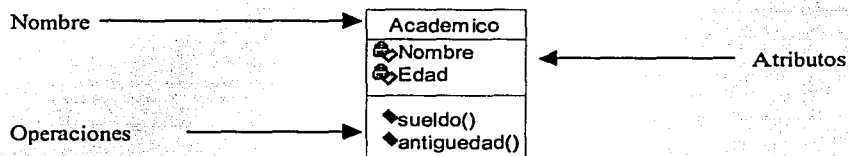


Fig. 2.17 Clase

A continuación se muestran las clases más importantes del sistema y su funcionalidad.

- **Academico:** Representa al usuario principal del sistema.
- **Acceso:** Es la clase encargada de validar el acceso al sistema.
- **VentanaMenu:** Es la ventana principal del sistema.
- **VentanaReporte:** Clase encargada de generar los reportes.
- **SInforme:** Se encargara de realizar cualquier modificación de la información.
- **SRubro:** Es la clase encargada de gestionar la información de un rubro en especial.
- **CRubro:** Representa un rubro en especial.
- **VentanaRubro:** Es la ventana que maneja la información de un rubro en especial.

Clase Academico

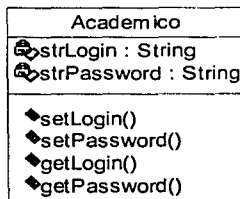


Fig. 2.18 Clase Academico.

Clase Acceso

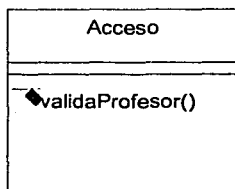


Fig. 2.19 Clase Acceso.

Clase MenuInforme

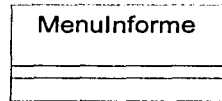


Fig. 2.20 Clase MenuInforme.

Clase MenuReporte

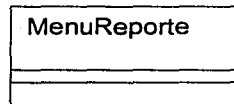


Fig. 2.21 Clase MenuReporte.

Clase SInforme

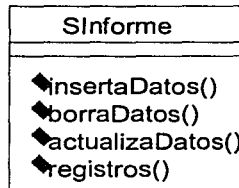


Fig. 2.22 Clase SInforme.

Clase SRubro

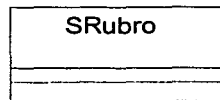


Fig. 2.23 Clase SRubro.

Clase VentanaRubro

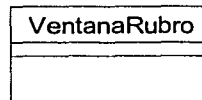


Fig. 2.24 Clase VentanaRubro.

Clase CRubro

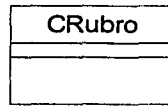


Fig. 2.25 Clase CRubro.

Diagrama de Clases del Sistema

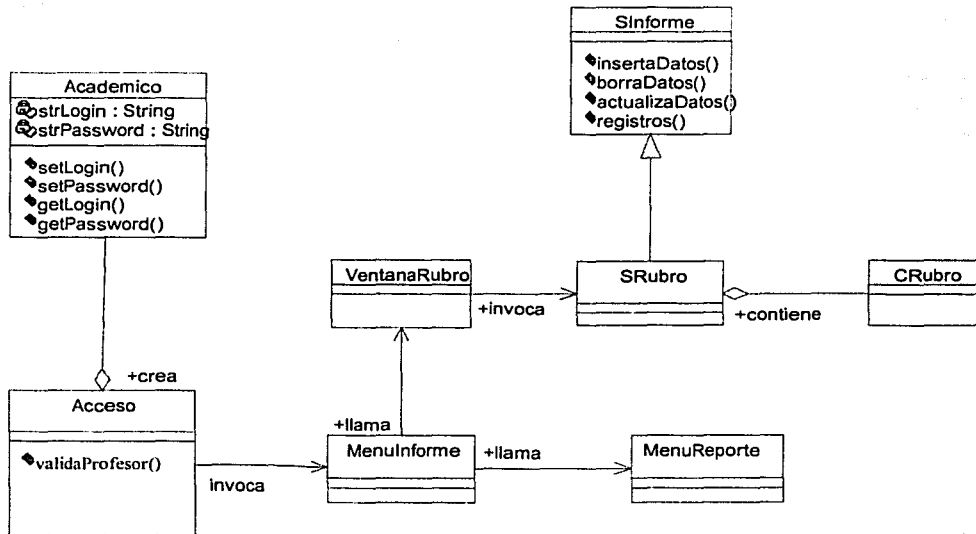


Fig. 2.26 Diagrama de Clases.

Diseño

CAPÍTULO 3. Diseño

**TESIS CON
FALLA DE ORIGEN**

1. Plataforma y tecnología seleccionada

Durante el análisis se realizó una comparativa entre las diferentes tecnologías disponibles para el desarrollo de aplicaciones web, las cuales separamos en cinco rubros, que son:

- Sistema Operativo.
- Servidor Web.
- Lenguajes de programación.
- Sistema de base de datos.
- Herramientas de desarrollo.

1.1 Sistema Operativo

Se inclino por usar el sistema operativo Windows, ya que cuenta con una interfaz de usuario sencilla, entendible y popular, además prescinde de capacitación especializada, por último, este sistema operativo tiene una gran difusión en el lugar donde será montado el sistema y la base de datos, Fig.3.1

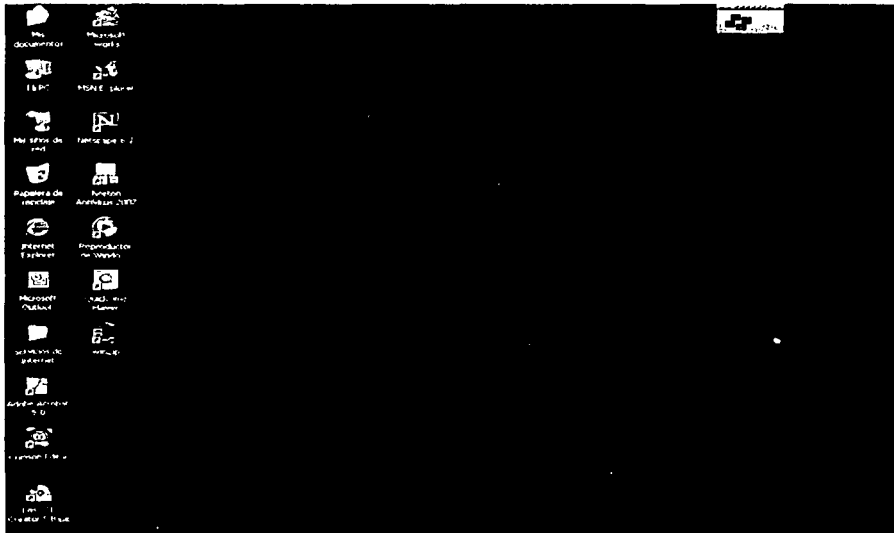


Fig. 3.1 Sistema Operativo Windows

1.2 Servidor Web

Se opta por el servidor web Apache, ya que es muy rápido, confiable y sobretodo esta libre de licencia, se cuenta con una versión de acuerdo a la plataforma donde se desea montar, Fig. 3.2

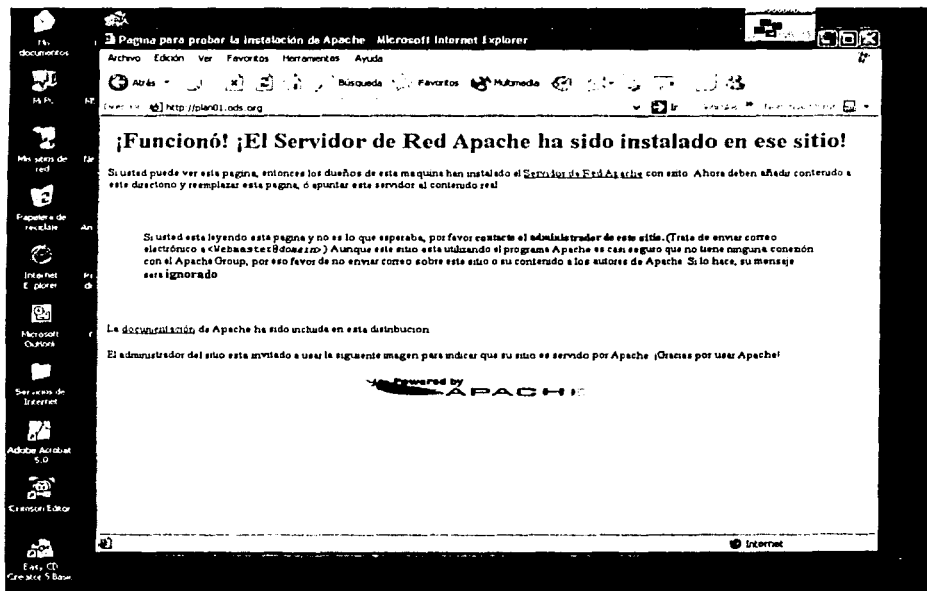


Fig. 3.2 Instalación del Servidor Web Apache en un sistema Windows

1.3 Lenguajes de programación

Existe una gran variedad de lenguajes para la programación de sistemas para la web, pero la mayoría tiene el inconveniente de la portabilidad, se eligieron los siguientes para el desarrollo del sistema:

- Java (Servlets y JSP).
- JavaScript.

Java (Servlets y JSP)

Java es un lenguaje multiplataforma, de libre distribución y con una API de seguridad muy amplia, entre las extensiones que cuenta para aplicaciones web están la API Servlet y el API JSP.

Los Servlets cuentan con una extensa infraestructura para analizar y decodificar automáticamente los datos de formularios HTML, leer y establecer encabezados http, administrar las cookies, rastrear las sesiones y muchas otras utilidades de alto nivel como estas, poseen la ventaja de compartir datos entre ellos, lo que facilita implementar un grupo de conexiones a bases de datos y optimizaciones similares para compartir recursos.

La tecnología de las JSP (Java Server Pages) permiten mezclar HTML regular y estático con un contenido dinámicamente generado a partir de los Servlets, la parte dinámica de una JSP está escrita en JAVA.

La combinación de Servlets con JSP permite la creación de sistemas complicados y robustos que podrían necesitar diversas presentaciones sustancialmente distintas, un Servlet, podría manejar la petición inicial, procesar de manera parcial los datos, establecer y reenviar los resultados a una de varias páginas JSP distintas de acuerdo con las circunstancias.

Para poder ejecutar Servlets y páginas JSP es necesario contar con un programa que se liga al servidor web, uno de los mejores motores y que además es de libre distribución es Jakarta Tomcat

Apache Tomcat

Tomcat es la implementación de referencia oficial de las especificaciones de Servlets y JSP. Puede ser usado como un pequeño servidor autónomo para probar los Servlets y las paginas JSP, o puede estar integrado dentro del servidor web Apache. Tomcat como el propio Apache son de libre distribución, rápidos y confiables. Fig. 3.3

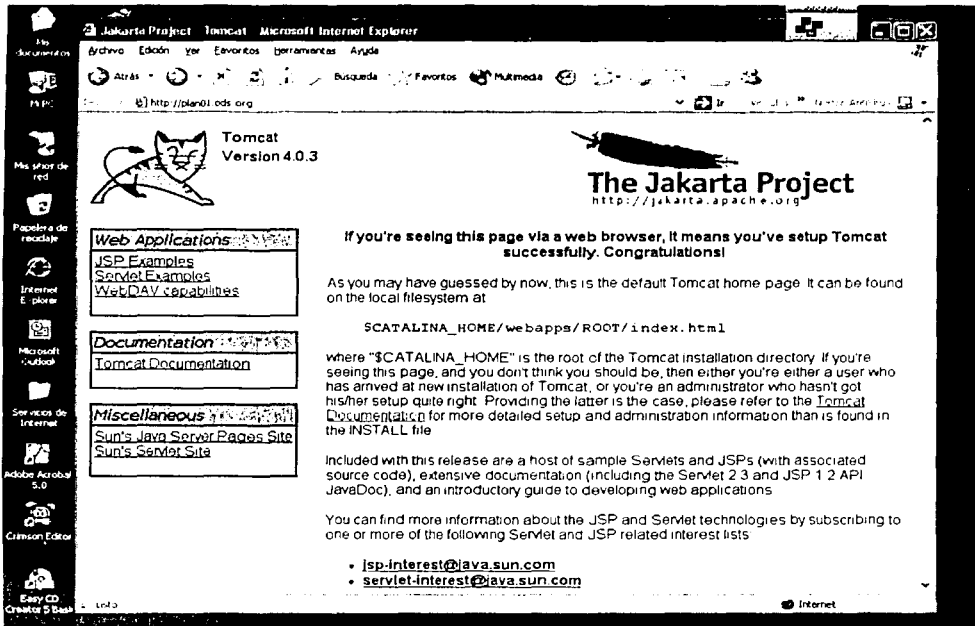


Fig. 3.3 Instalación de Tomcat en un sistema Windows.

JavaScript

Como se estará enviando mucha información al servidor ésta debe cumplir con ciertas características, como puede ser el formato de fechas, que los datos numéricos lo sean, que no se rebase el tamaño de un campo, etc. JavaScript permite validar la información que el usuario proporciona, de esta forma se garantiza que la información sea la adecuada.

1.4 Sistema de base de datos

Access 2000

Access cuenta con características que lo hacen un producto fácil de utilizar, potente y flexible, es accesible para todos los niveles de usuarios, desde principiantes hasta diseñadores de bases de datos, con la finalidad de que las bases de datos se transformen en una aplicación de uso rutinario para los usuarios de computadoras.

Access es un programa para el manejo de bases de datos relacionales que nos permiten el almacenamiento, agrupamiento y búsqueda rápida de todo tipo de datos. Además de ser un programa poderoso y fácil de usar se encontró que no existen razones suficientes razones para la adquisición de nuevo software y permitirá un menor tiempo de adaptación al sistema, ya que es conocido por las personas que lo manejarán, Fig. 3.4.

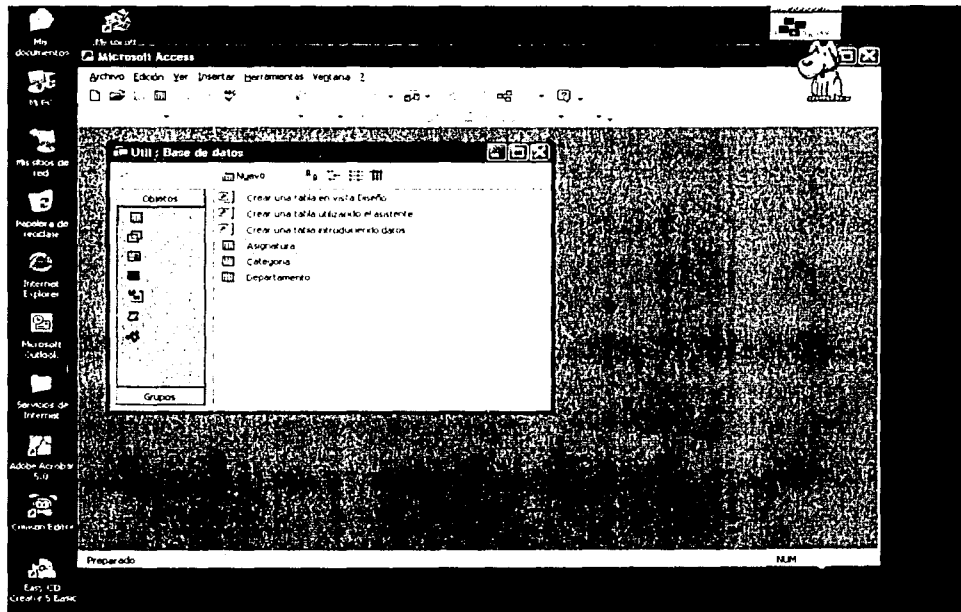


Fig. 3.4 Sistema de base de datos Access.

2. Diseño de la base de datos

2.1 Modelo E/R

A continuación se muestran las entidades definidas para la creación de la base de datos por medio de un diccionario de datos inicial, Tabla 3.1

Entidad	Descripción
Academico	Se refiere a los datos personales del Académico
DomicilioParticular	Información referente al domicilio del Académico
Curso	Descripción de los cursos que tomo en el año.
Congreso	Asistencia a congresos.
TrabajoCongreso	Trabajos presentados en congresos.
Difusion	Son conferencias magistrales, científicas, cursos, presentación de carteles, organización o coordinación de cursos o seminarios, que haya realizado en el año.
Tesis	Participación en tesis como Tutor principal, cotutor, coasesor o colaborador invitado.
Memoria	Memorias realizadas sobre cursos, congresos, diplomados.
Extra	Realización de exámenes extraordinarios.
Tutorial	Participación como asesor en tutorales..
ArticuloArbitrada	Presentación de artículos en revistas arbitradas.
FormacionAcademica	Trayectoria académica.

Tabla 3.1 Diccionario de datos inicial.

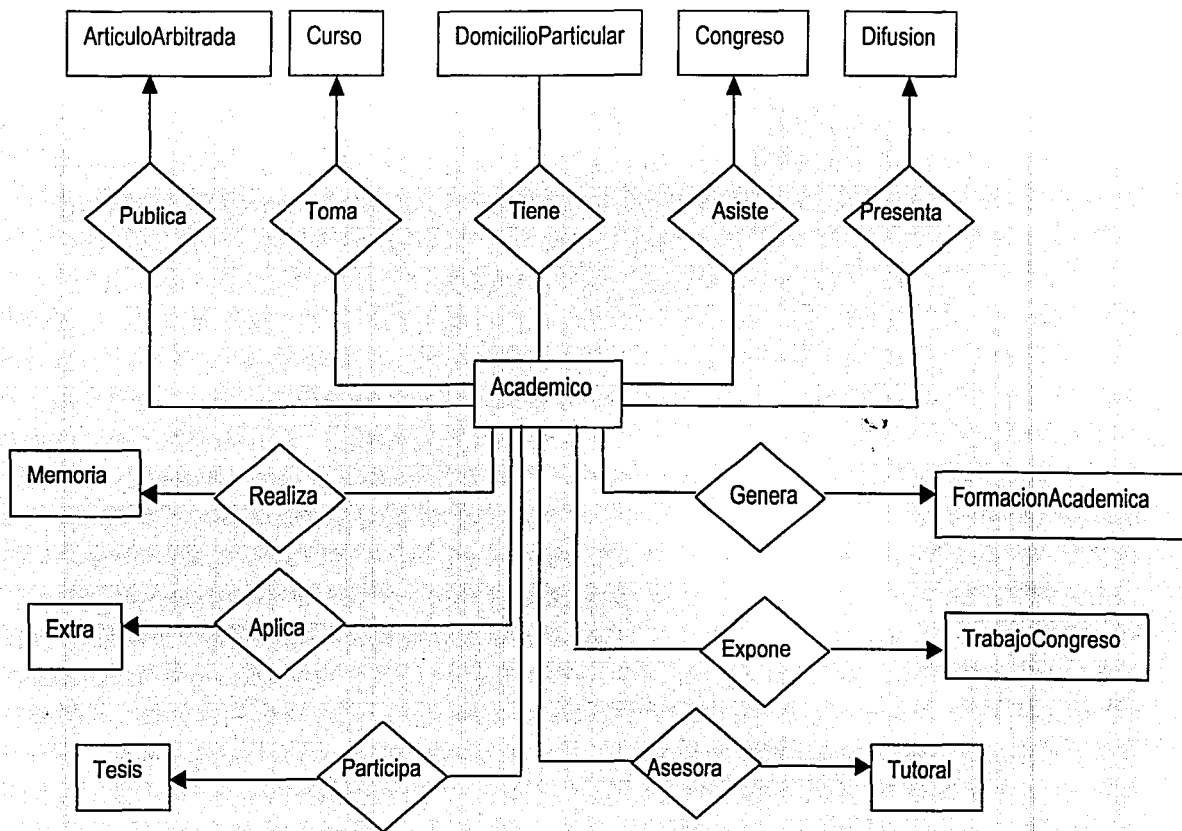


Fig. 3.5 Diagrama entidad / relación.

2.2 Modelo Relacional

A continuación se muestra el modelo relacional para la base de datos del sistema, Fig.3.6.

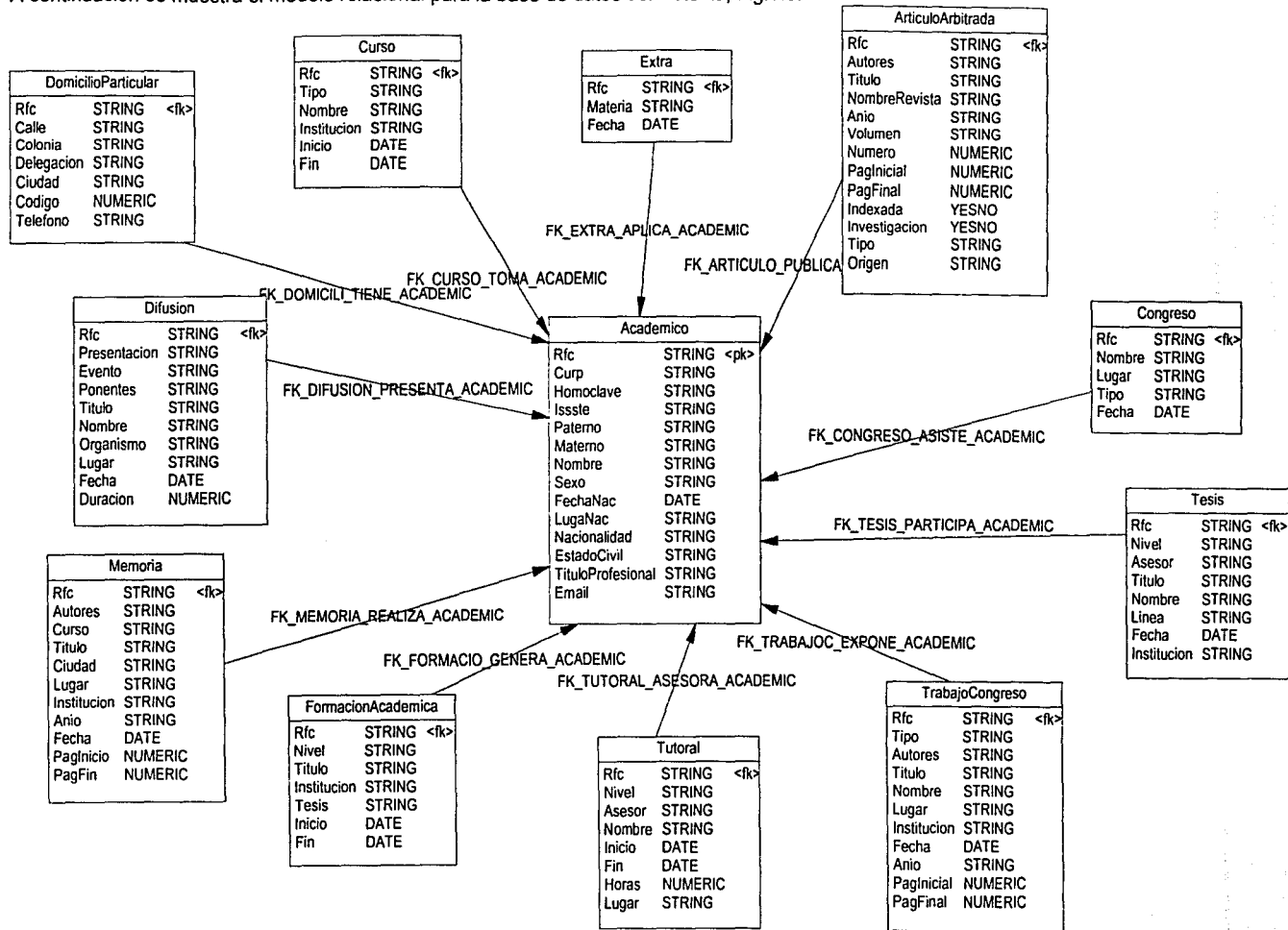


Fig. 3.6 Diagrama Relacional.

2.3 Diccionario de Datos

Diccionario de datos (Tabla 3.2) almacena información acerca de la estructura de la base de datos.

Sus principales funciones son las siguientes:

- Describe todos los elementos en el sistema.
- Los elementos se centran en los datos y la forma en que están estructurados.

Diccionario de datos

Entidad	Atributo	Tipo	Tamaño		Valor por omisión
			máximo	mínimo	
Academico	Rfc	String	10	10	
	Curp	String	18		
	Homoclave	String	3		
	Issste	String	10		
	Paterno	String	25		
	Materno	String	25		
	Nombre	String	35		
	Sexo	String	9	8	
	FechaNac	Date	dd/mm/aaaa	dd/mm/aa	
	LugarNac	String	25		
	Nacionalidad	String	20		
	EstadoCivil	String	7		
TituloProfesional	String	80			
Email	String	50			
DomicilioParticular	Calle	String	35		
	Colonia	String	35		
	Delegación	String	25		
	Ciudad	String	50		
	Codigo	Numeric			
	Telefono	String	17	8	
Curso	Tipo	String	15		
	Nombre	String	80		
	Institución	String	50		
	Inicio	Date	dd/mm/aaaa	dd/mm/aa	
	Fin	Date	dd/mm/aaaa	dd/mm/aa	Actual
Extra	Materia	String	80		
	Fecha	Date	dd/mm/aaaa	dd/mm/aa	
Congreso	Nombre	String	80		
	Lugar	String	80		
	Tipo	String	14	8	
	Fecha	Date	dd/mm/aaaa	dd/mm/aa	

Diccionario de datos

Difusion	Presentacion	String	50	22	
	Evento	String	9	5	
	Ponentes	String	200		
	Titulo	String	250		
	Nombre	String	250		
	Organismo	String	250		
	Lugar	String	100		
	Fecha	Date	dd/mm/aaaa	dd/mm/aa	
	Duracion	Numeric			
Memoria	Autores	String	200		
	Curso	String	13	6	
	Titulo	String	250		
	Ciudad	String	150		
	Lugar	String	150		
	Institucion	String	150		
	Anio	String	Actual		
	Fecha	Date	dd/mm/aaaa	dd/mm/aa	
	PagInicio	Numeric			
	PagFin	Numeric			
FormacionAcademica	Nivel	String	12	8	
	Titulo	String	250		
	Institucion	String	200		
	Tesis	String	250		
	Inicio	Date	dd/mm/aaaa	dd/mm/aa	
	Fin	Date	dd/mm/aaaa	dd/mm/aa	Actual
Tesis	Nivel	String	12	8	
	Asesor	String	20	7	
	Titulo	String	250		
	Nombre	String	250		
	Linea	String	150		
	Fecha	Date	dd/mm/aaaa	dd/mm/aa	
	Institucion	String	150		
ArticuloArbitrada	Autores	String	250		
	Titulo	String	250		
	NombreRevista	String	150		
	Anio	String	Actual		
	Volumen	String	30		
	Numero	Numeric			
	PagInicial	Numeric			
	PagFinal	Numeric			
	Indexada	Bolean			
	Investigación	Bolean			
	Tipo	String	30	20	
Origen	String	14	8		

Diccionario de datos

TrabajoCongreso	Tipo	String	17	7	
	Autores	String	250		
	Titulo	String	250		
	Nombre	String	250		
	Lugar	String	200		
	Institucion	String	200		
	Fecha	Date	dd/mm/aaaa	dd/mm/aa	
	Anio	String	Actual		
	PagInicial	Numeric			
PagFinal	Numeric				

Tutoral	Nivel	String	12	8	
	Asesor	String	20	15	
	Nombre	String	250		
	Inicio	Date	dd/mm/aaaa	dd/mm/aa	
	Fin	Date	dd/mm/aaaa	dd/mm/aa	
	Horas	Numeric			
	Lugar	String	100		

Tabla 3.2 Diccionario de datos.

3. Diseño del sistema propuesto

3.1 Diagramas de secuencia

Este diagrama representa el acceso autorizado del usuario al sistema desde un navegador, Fig.3.7

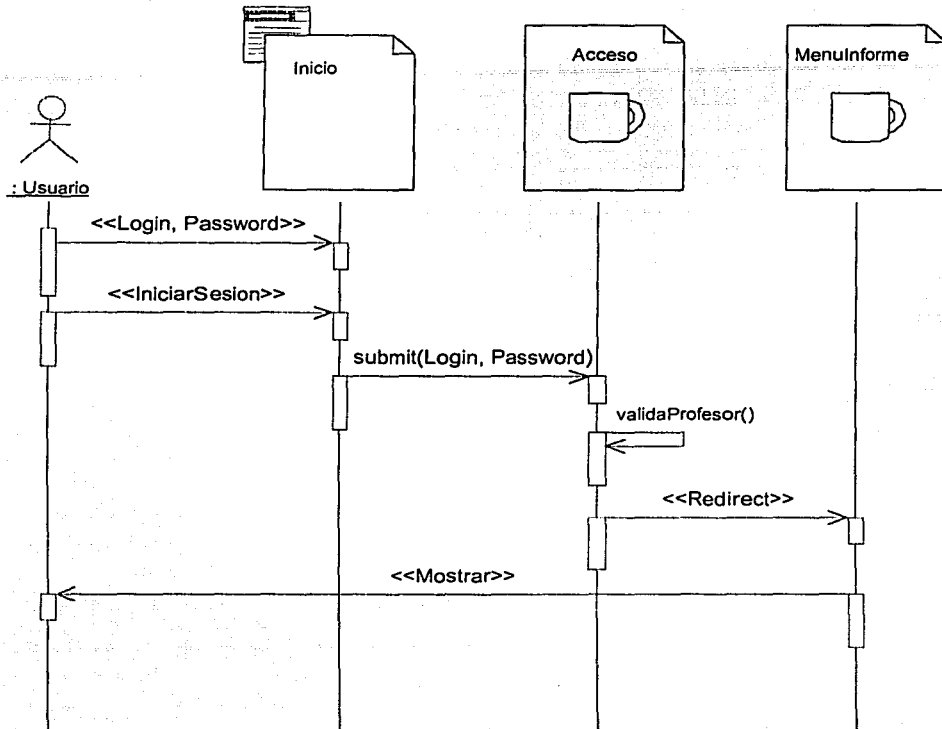


Fig. 3.7 Acceso Autorizado.

Para un acceso no autorizado el diagrama es el siguiente, Fig. 3.8.

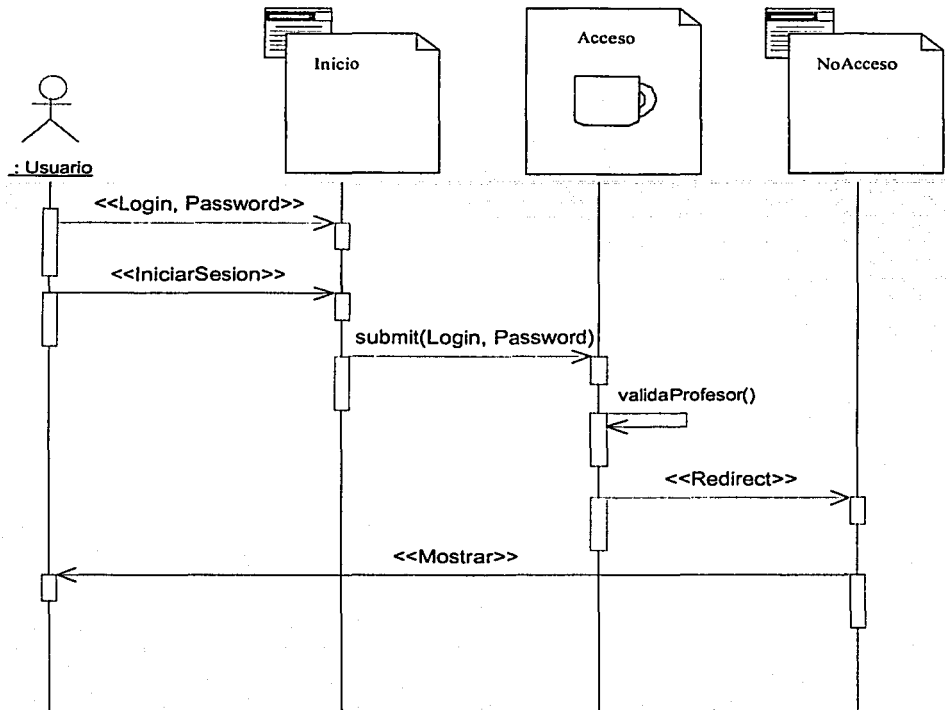


Fig. 3.8 Acceso No Autorizado.

Cuando el usuario ingresa al Menú del Informe tiene que elegir algún rubro, este diagrama se muestra a continuación, Fig. 3.9.

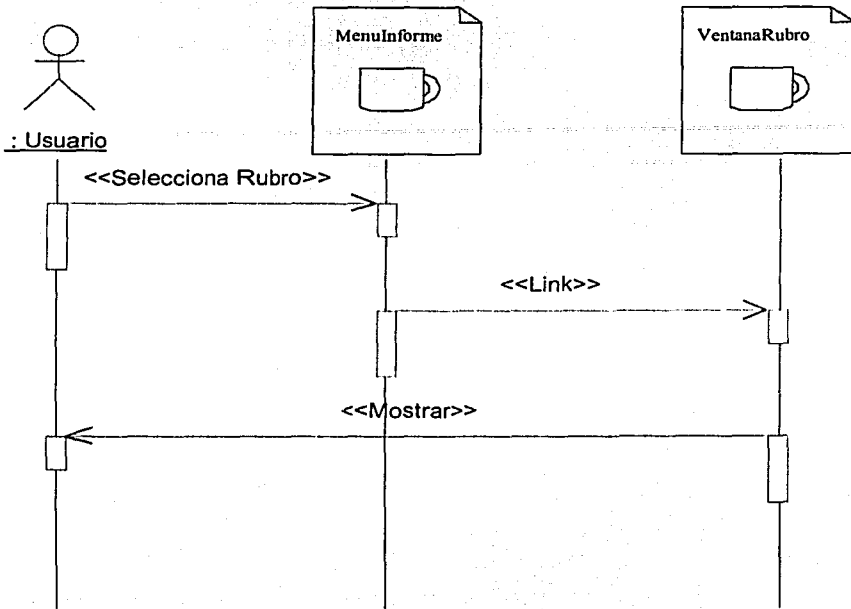


Fig. 3.9 Selección de rubro.

Después de que selecciono un rubro se muestra la ventana correspondiente, en donde pueden realizar las acciones de "Registros" Fig. 3.9 que permite ver la información existente o "Insertar" Fig. 3.10 que permite agregar nueva información a la base de datos.

Acción "Registros"

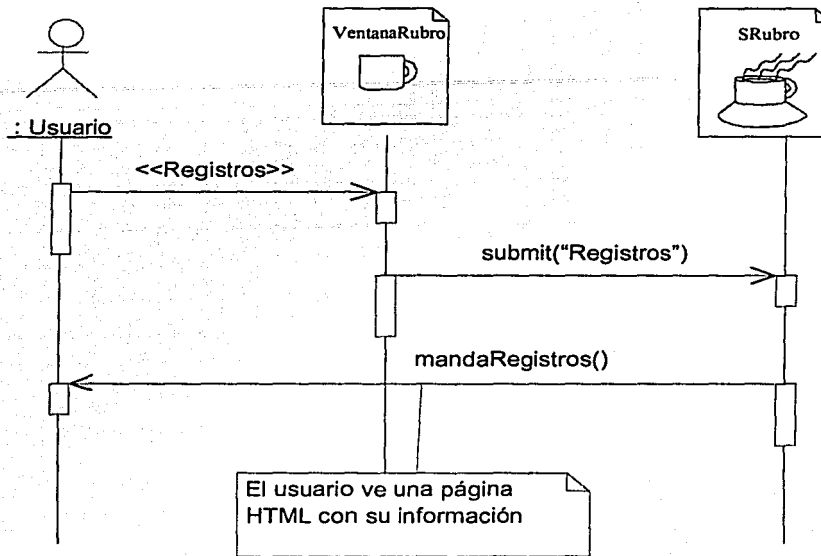


Fig. 3.10 Acción "Registros".

Cuando el usuario visualiza los registros en pantalla, tiene la opción de modificar o borrar alguno, si desea modificar un registro este se carga en la "VentanaRubro" correspondiente donde puede modificarlo y enviar la actualización, para lo cual el diagrama es el siguiente. Fig. 3.11

Acción "Modificar"

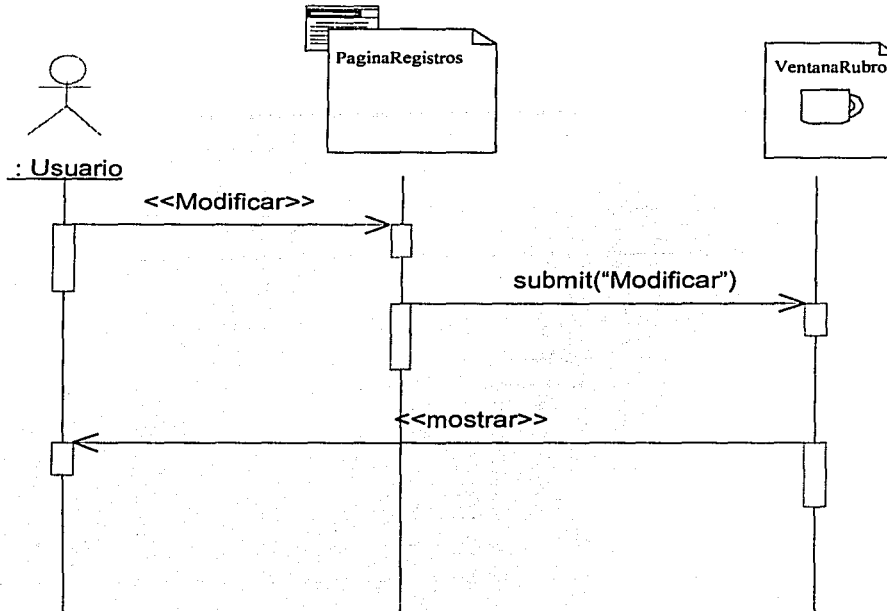


Fig. 3.11 Acción "Modificar".

Desde "PaginaRegistros" el usuario puede borrar un registro, este acción se envía al "SRubro" correspondiente el cual lleva a cabo la acción devolviendo un mensaje de si se llevo a cabo la acción requerida o no, Fig. 3.12.

Acción "Borrar"

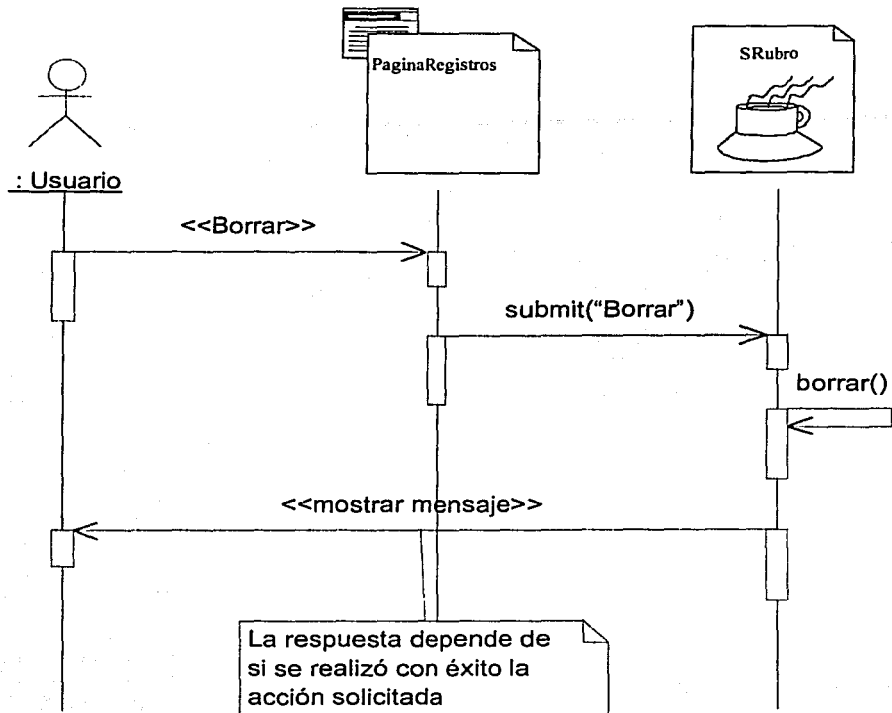


Fig. 3.12 Acción "Borrar".

Acción "Insertar"

Si el usuario selecciona "Insertar" desde la "VentanaRubro" el "SRubro" correspondiente genera un registro nuevo en la base de datos, previamente debió haber escrito datos en el formulario antes de enviarlo, Fig. 3.13

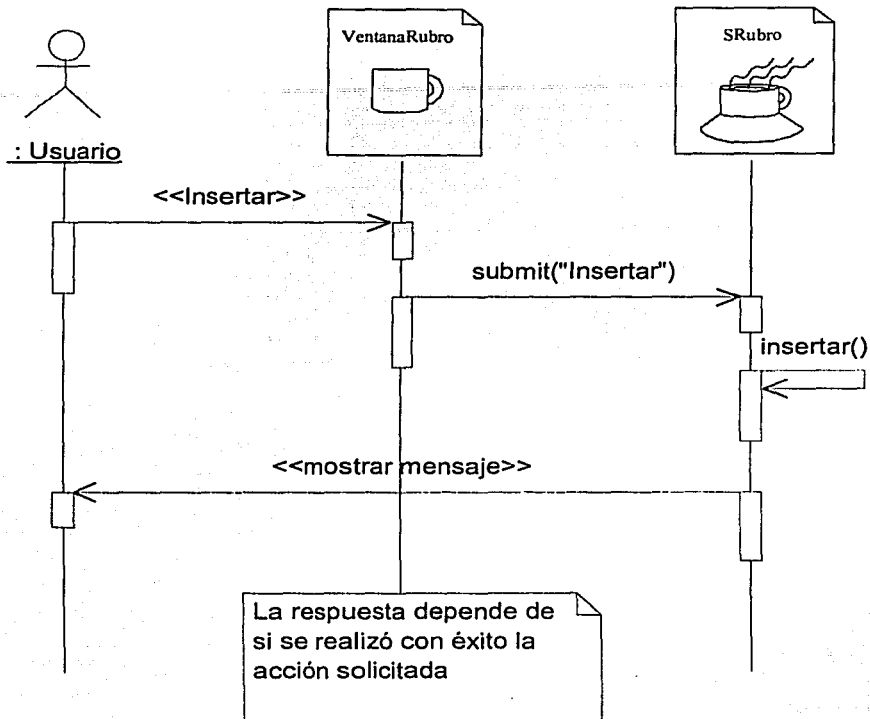


Fig. 3.13 Acción "Insertar".

Crear reporte

Estando en la "VentanaMenu" puede seleccionar la opción "Reporte" la cual lo lleva a la página "VentanaReporte", donde puede seleccionar entre dos formatos diferentes para crear sus reporte, Fig. 3.14.

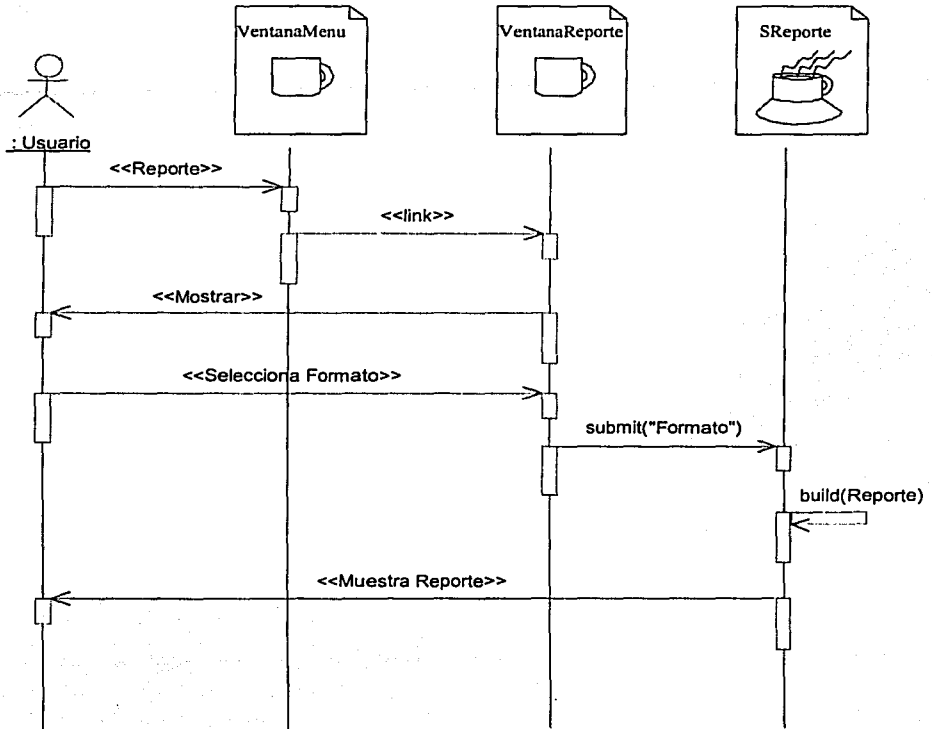


Fig. 3.14 Crear Reporte.

3.2 Diagramas de clases

La Tabla 3.3 muestra la estructura de cada una de las clases.

Nombre	Tipo	Atributos	Métodos	Descripción
Profesor	<i>Pura</i>	strLogin strPassword	setLogin() setPassword() getLogin() getPassword() Limpia()	Representa a cada usuario dentro del sistema.
Inicio	<i>ClientPage</i>		cambia()	Es la página de inicio.
frmInicio	<i>Form</i>	txtLogin txtPassword		Formulario para iniciar sesión
Acceso	<i>JSP Page</i>		validaProfesor()	Se encarga de validar el acceso del usuario.
NoAcceso	<i>ClientPage</i>			Es una página que notifica que el usuario no tiene acceso
MenuInforme	<i>JSP Page</i>			Página principal para acceder a los rubros.
HttpSession	<i>Pura</i>			Clase de Java para manejar las sesiones.
VentanaReporte	<i>JSP Page</i>			Página para generar el reporte de un usuario.
frmReporte	<i>Form</i>	lstTipo		Formulario que permite seleccionar el tipo de reporte.
SReporte	<i>Servlet</i>		crearReporte()	Servlet encargado de generar el reporte de cada usuario.
VentanaRubro	<i>JSP Page</i>			Página que muestra el formulario del rubro seleccionado.
frmRubro	<i>Form</i>			Formulario del rubro seleccionado.
SRubro	<i>Servlet</i>		mandaRegistros()	Servlet encargado de gestionar los eventos dados por el frmRubro

CRubro	<i>Pura</i>		insertar() borrar() modificar()	Clase para representar cada uno de los rubros.
PaginaRegistros	<i>ClientPage</i>			Página que se construye con los registros del usuario.
SInforme	<i>Servlet</i>	conexión con objCon seHizo	getConexion() liberar() cabecera() finSesion() insertaDatos() borraDatos() actualizaDatos() registros() hecho() noHecho()	Servlet principal encargado de administrar y de reutilizar las conexiones a la base de datos.

Tabla 3.3 Descripción de clases.

El diagrama de la Fig. 3.15 representa el inicio de sesión de un usuario.

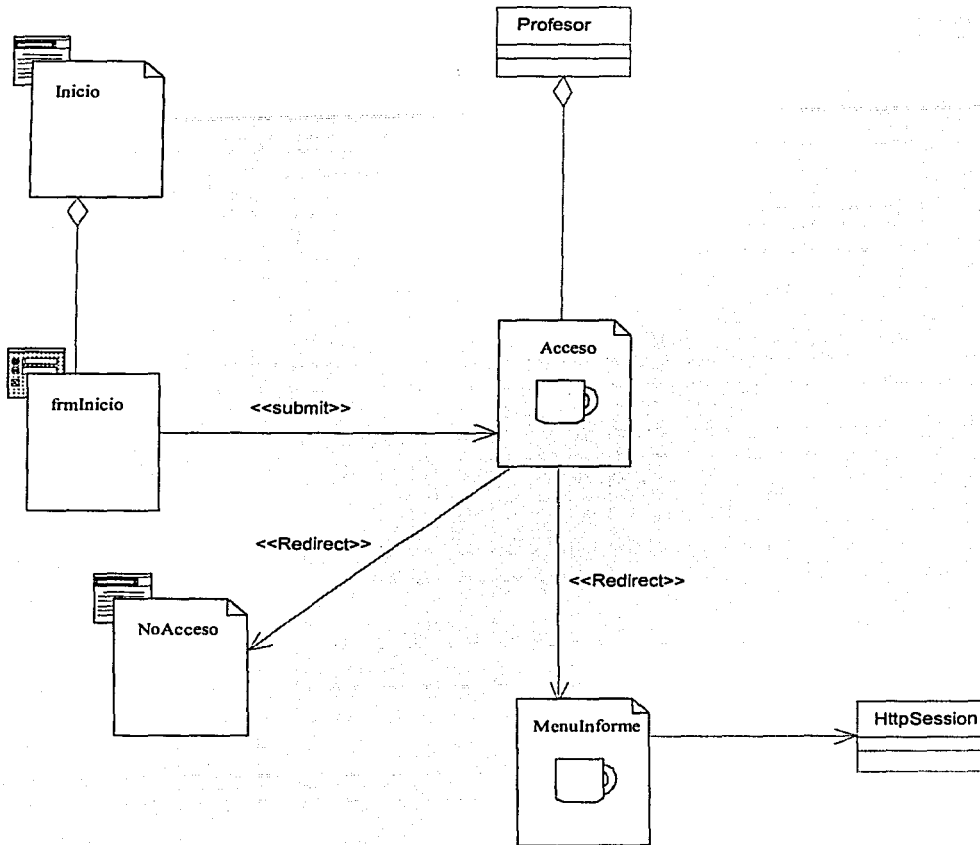


Fig. 3.15 Inicio Sesión.

Cuando el usuario selecciona un rubro desde el "Menuinforme" se abre la ventana del rubro seleccionado mostrando el formulario correspondiente, aquí puede "Insertar" registros, Fig. 3.16.

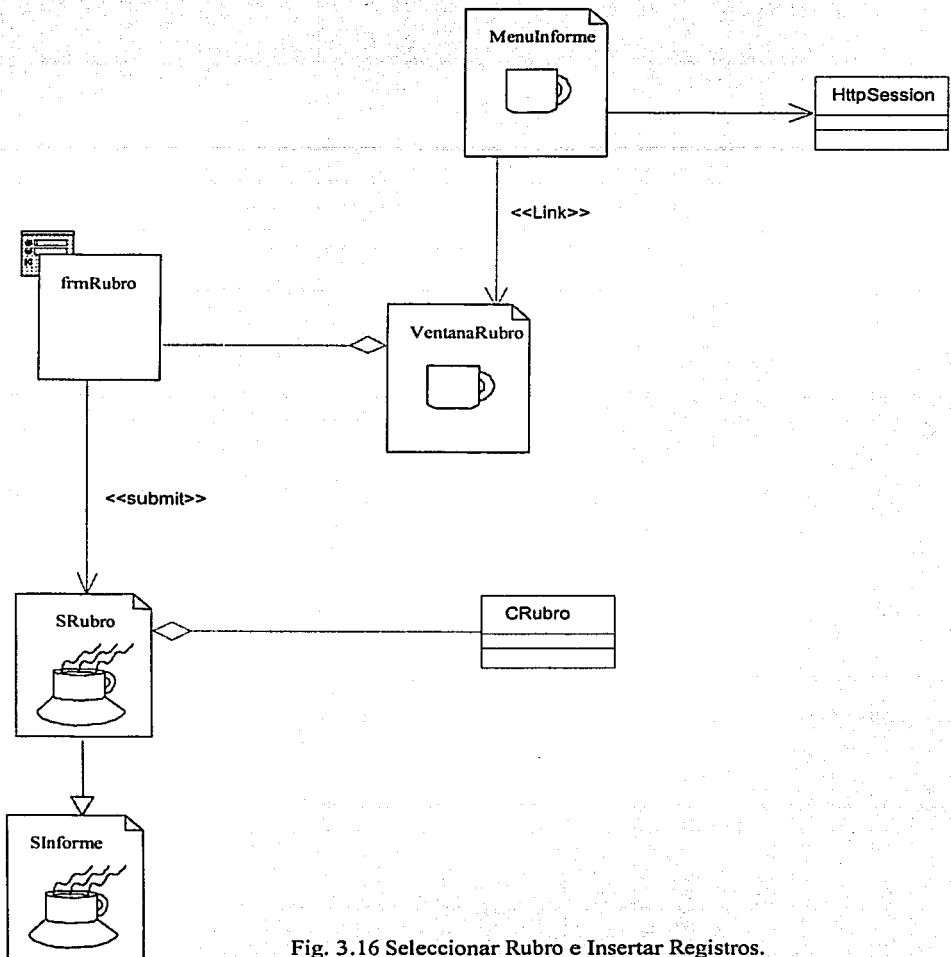


Fig. 3.16 Seleccionar Rubro e Insertar Registros.

En la "PaginaRegistros" el usuario puede borrar o mandar a modificar algún registro en especial,
 Fig. 3.17

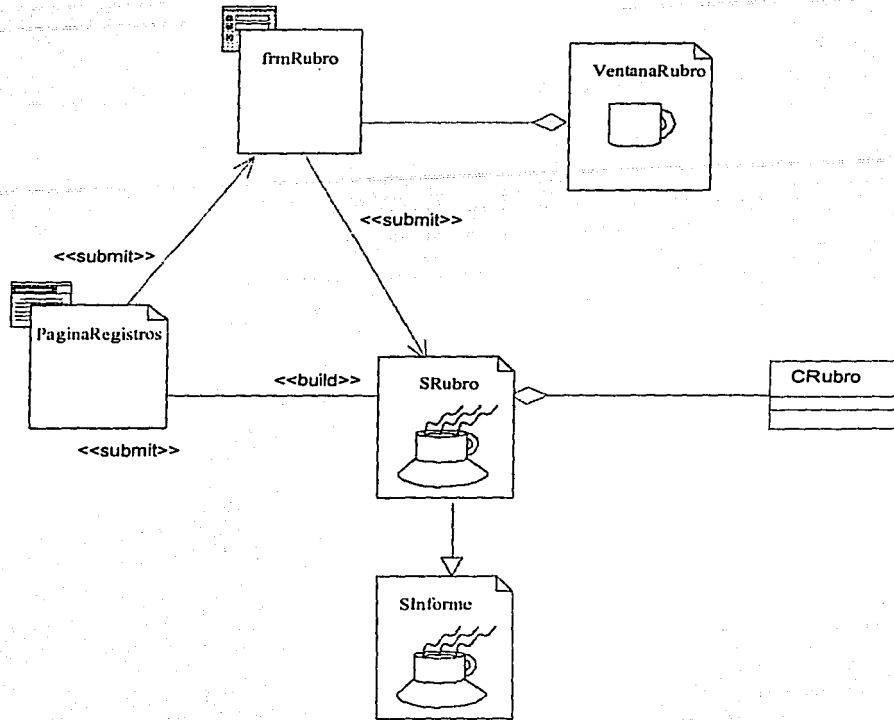


Fig. 3.17 Modificar y borrar registros.

Diagrama de clases para la creación de un reporte, Fig. 3.18

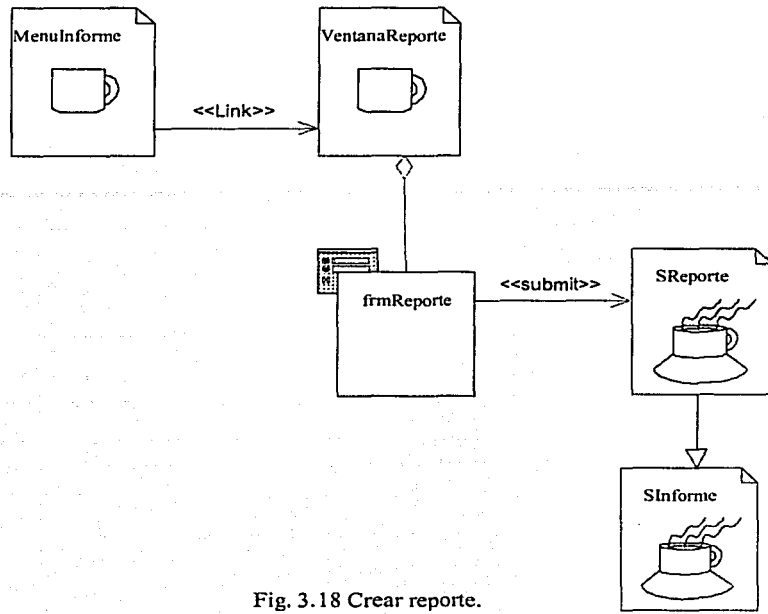


Fig. 3.18 Crear reporte.

Desarrollo

CAPÍTULO 4. Desarrollo

1. Desarrollo del sistema

1.1 Programación orientada a objetos y Java

La programación orientada a objetos es, realmente, otra técnica para implementar el famoso dicho de la programación: "divide y vencerás". La idea es encapsular datos y métodos en objetos, de forma que cada objeto sea semiautónomo, encerrando métodos y datos privados y salvándolos del desorden general que les rodea. Así, el objeto puede interactuar con el resto del programa por medio de una interfaz bien definida por sus métodos públicos (es decir, se les puede invocar desde fuera).

El resultado de encapsular partes de un programa en un objeto es que es concebido como un elemento sencillo y no hay que tratar todo lo que el objeto hace internamente.

La programación orientada a objetos gira sobre algunos conceptos clave: objetos, clase, encapsulación, herencia, polimorfismo. De forma rápida, éstos términos significan:

- Dentro de la programación orientada a objetos, un objeto es cualquier cosa, real o abstracta, acerca de la cual almacenamos datos y los métodos que controlan dichos datos. Un objeto puede estar compuesto por otros objetos. Estos últimos a su vez también pueden estar compuestos por otros objetos. Esta intrincada estructura es la que permite construir objetos muy complejos.
- Una clase es una implantación de un tipo de objeto. Especifica una estructura de datos y los métodos operativos permisibles que se aplican a cada uno de sus objetos.
- El encapsulado oculta los detalles de su implantación interna a los usuarios de un objeto. Los usuarios se dan cuenta de las operaciones que puede solicitar del objeto, pero desconocen los detalles de cómo se lleva a cabo la operación. Todos los detalles específicos de los datos del objeto y la codificación de sus operaciones están fuera del alcance del usuario.
- Una clase puede *heredar* tanto los atributos como las operaciones de una clase *madre*, también llamada *super clase*. Esto implica que una nueva clase que sea bastante similar a otra clase en varios aspectos no necesita volver a implementar nuevamente todos los métodos de esa segunda clase, sino que puede heredar de ella los métodos y sobrescribir aquellos que son diferentes.
- El polimorfismo es la propiedad que tienen algunos objetos para comportarse como otros, esto es, si un objeto hereda de una super clase entonces se puede comportar como la super clase o como es definido.

El siguiente ejemplo muestra los conceptos de clase, encapsulación y herencia, Ejemplo 4.1.

```
public class Natural
{
    private int IntNumero;

    public static int DOS = 2;
    public static int TRES = 3;

    public Natural(int entero)
    {
        setNumero(entero);
    }

    public void setNumero(int ent)
    {
        intNumero = ent;
    }

    public int getNumero()
    {
        return intNumero;
    }

    public static boolean entreDos(int numero)
    {
        if(numero%DOS==0)
            return true;
        else
            return false;
    }

    public static boolean entreTres(int numero)
    {
        if(numero%TRES==0)
            return true;
        else
            return false;
    }
}

class EnteroDos extends Natural implements Divisible
{
    public EnteroDos(int par)
    {
        super(par);
    }

    public int vecesCabe(int num)
    {
        return num / DOS;
    }
}

class EnteroTres extends Natural implements Divisible
{
    public EnteroTres(int impar)
    {
        super(impar);
    }

    public int vecesCabe(int num)
    {
        return num / TRES;
    }
}

interface Divisible
{
    public int vecesCabe(int num);
}
```

Este otro ejemplo muestra el concepto de objetos y polimorfismo, Ejemplo 4.2.

```
import java.util.Vector;

public class Probar
{
    public static void main(String arg[])
    {
        Vector dos = new Vector();
        Vector tres = new Vector();

        for(int i=0; i<arg.length; i++)
        {
            try
            {
                int num = Integer.parseInt(arg[i]);

                if(Natural.entreDos(num))
                    dos.addElement(new EnteroDos(num));

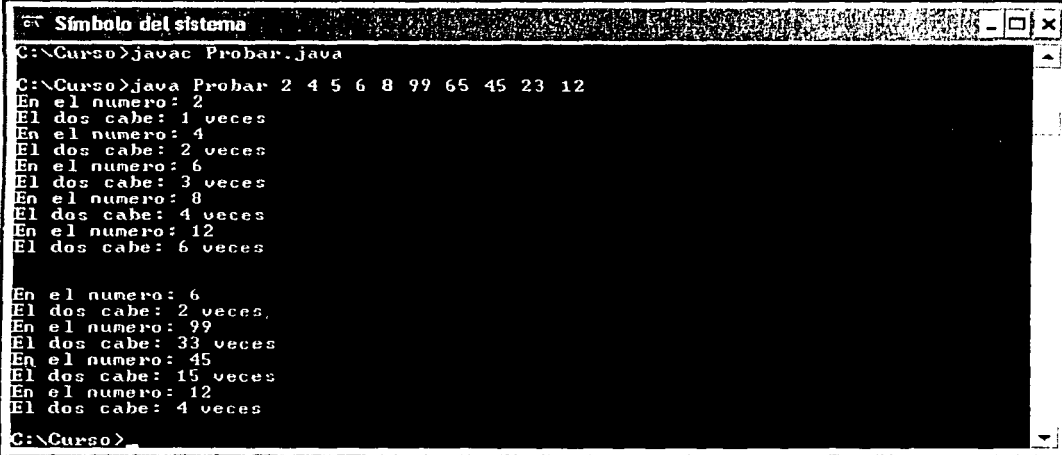
                if(Natural.entreTres(num))
                    tres.addElement(new EnteroTres(num));
            }
            catch(NumberFormatException e)
            {
                System.out.println("Número no entero: "+arg[i]);
            }
        }

        for(int i=0; i<dos.size(); i++)
        {
            Natural numDos = (Natural)dos.elementAt(i);
            System.out.println("En el numero: "+numDos.getNumero());
            Divisible div = (Divisible)dos.elementAt(i);
            System.out.println("El dos cabe: "+div.vecesCabe(numDos.getNumero())+" veces");
        }

        System.out.println("\n");

        for(int i=0; i<tres.size(); i++)
        {
            Natural numTres = (Natural)tres.elementAt(i);
            System.out.println("En el numero: "+numTres.getNumero());
            Divisible div = (Divisible)tres.elementAt(i);
            System.out.println("El tres cabe: "+div.vecesCabe(numTres.getNumero())+" veces");
        }
    }
}
```

A continuación se muestra la salida del ejemplo anterior, Fig. 4.1:



```
C:\Curso>javac Probar.java
C:\Curso>java Probar 2 4 5 6 8 99 65 45 23 12
En el número: 2
El dos cabe: 1 veces
En el número: 4
El dos cabe: 2 veces
En el número: 6
El dos cabe: 3 veces
En el número: 8
El dos cabe: 4 veces
En el número: 0
El dos cabe: 4 veces
En el número: 12
El dos cabe: 6 veces

En el número: 6
El dos cabe: 2 veces
En el número: 99
El dos cabe: 33 veces
En el número: 45
El dos cabe: 15 veces
En el número: 12
El dos cabe: 4 veces
C:\Curso>
```

Fig. 4.1 Ejecución de la clase Probar.

1.2 Servlets y JSP

Servlets

Un servlet básico administrar tanto peticiones GET como POST que son peticiones usuales del navegador web, para ser un servlet una clase deberá extender a `HttpServlet` y sustituir los métodos `doGet()` o `doPost()`, dependiendo de si los datos son enviados por GET o por POST. Ambos métodos toman un par de argumentos: `HttpServletRequest` y `HttpServletResponse`, el `HttpServletRequest` contiene métodos por los cuales puede encontrar información entrante como los datos del formulario, encabezados de petición http y el nombre del host del cliente. `HttpServletResponse` le permite especificar la información de salida como los códigos de estado de http, encabezados de respuesta y, lo más importante le permite obtener un `PrintWriter` utilizado para enviar el contenido del documento de regreso al cliente.

TESIS CON
ORIGEN

Ejemplo 4.3 de un servlet que maneja peticiones GET

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class ManejaGet extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response)
    {
        PrintWriter salida = null;
        try
        {
            response.setContentType("text/html");
            salida = response.getWriter();

            salida.println("<html>");
            salida.println("<head><title>Salida GET</title></head>");
            salida.println("<body>");
            salida.println("Salida GET");
            salida.println("</body>");
            salida.println("</html>");

            salida.flush();
            salida.close();
        }
        catch(IOException e)
        {
        }
    }
}
```

Salida del servlet que maneja peticiones GET, Fig. 4.2

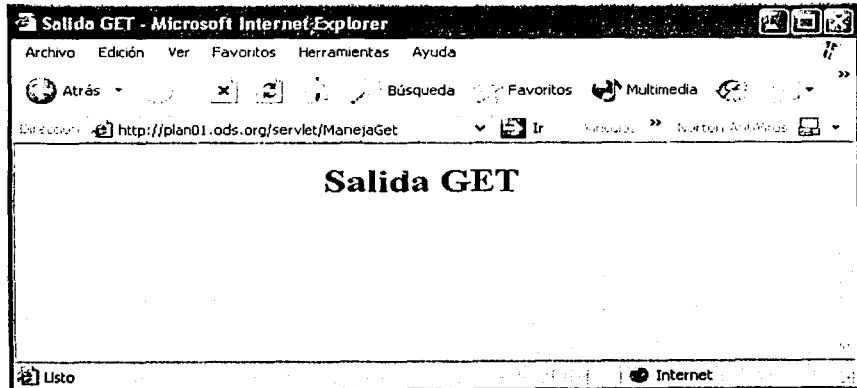


Fig. 4.2 Salida del servlet ManejaGet.

Ejemplo 4.4 de un servlet que maneja peticiones POST

```

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class ManejaPost extends HttpServlet
{
    public void doPost(HttpServletRequest request, HttpServletResponse response)
    {
        PrintWriter salida = null;
        try
        {
            response.setContentType("text/html");
            salida = response.getWriter();

            salida.println("<html>");
            salida.println("<head><title>Salida POST</title></head>");
            salida.println("<body>");
            salida.println("Salida POST");
            salida.println("</body>");
            salida.println("</html>");

            salida.flush();
            salida.close();
        }
        catch(IOException e)
        {
        }
    }
}

```

Para probar el servlet que maneja peticiones POST debemos crear una página HTML con un formulario que invoque este servlet.

Ejemplo 4.5 Página HTML para probar el método POST

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
    <title>Probando POST</title>
</head>
<body>
<form name="frmProbando" action="/servlet/ManejaPost" method="post">
Escribe un saludo:<br>
<input type="text" name="txtSaludo" size="25"><br>
<input type="Submit" value="Enviar Saludo">
</form>
</body>
</html>

```

Página HTML de prueba en el navegador, Fig. 4.3

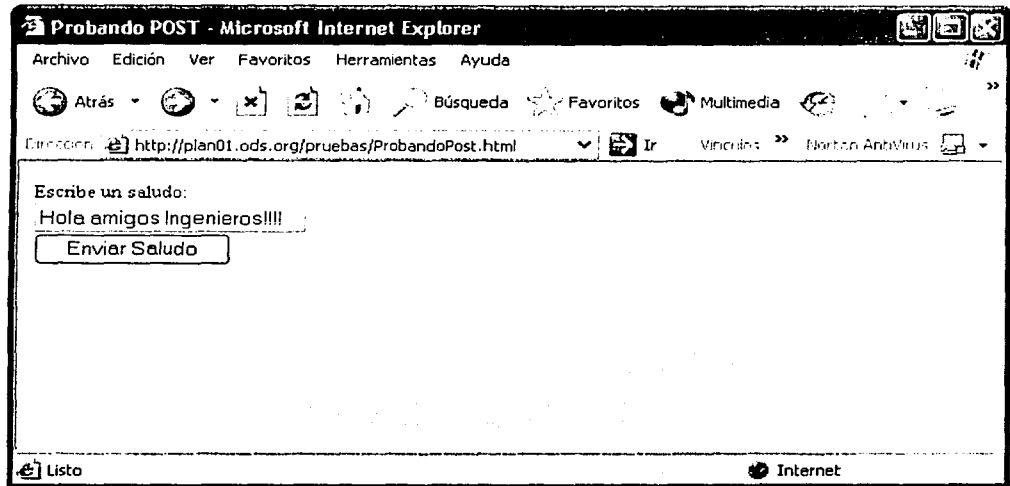


Fig. 4.3 Salida de la Página HTML.

Resultado de invocar al servlet ManejaPost desde la página HTML, Fig. 4.4.

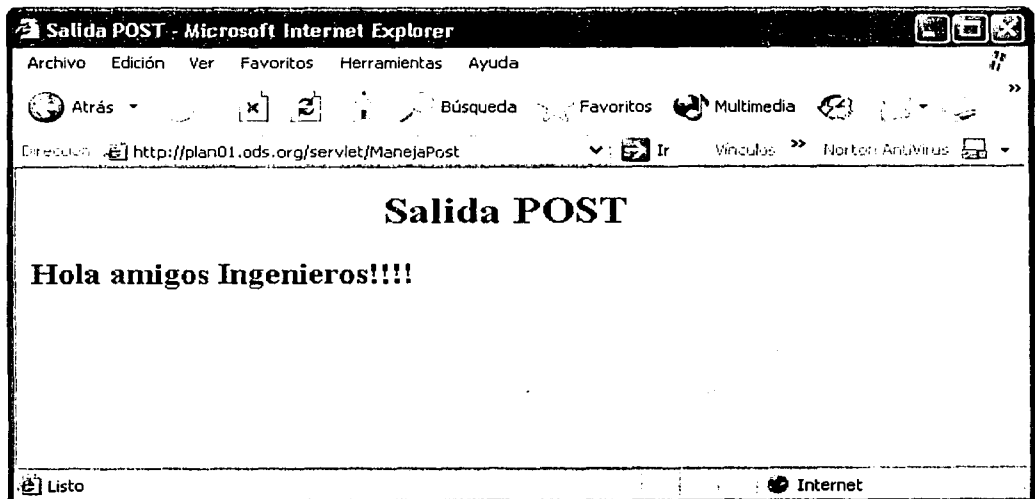


Fig. 4.4 Resultado de invocar el servlet ManejaPost

JSP

La tecnología JSP permite mezclar HTML estándar y estático con un contenido dinámico generado por los servlets. Tan solo debe escribir el código HTML de manera normal, luego encerrar el código de las partes dinámicas en etiquetas especiales, muchas de las cuales inician con `<%` y finalizan con `%>`

Ejemplo 4.5 de una página JSP sencilla.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transicional//EN">
<html>
<head>
  <title>Probando JSP</title>
</head>
<body>
<%
for(int i=1; i<=5; i++)
{
    out.println("<h3>Esta es la línea: "+i+"</h3>");
    out.println("<hr size='5px' noshade><br>");
}
%>
</body>
</html>
```

La salida en el navegador es la siguiente, Fig. 4.5:

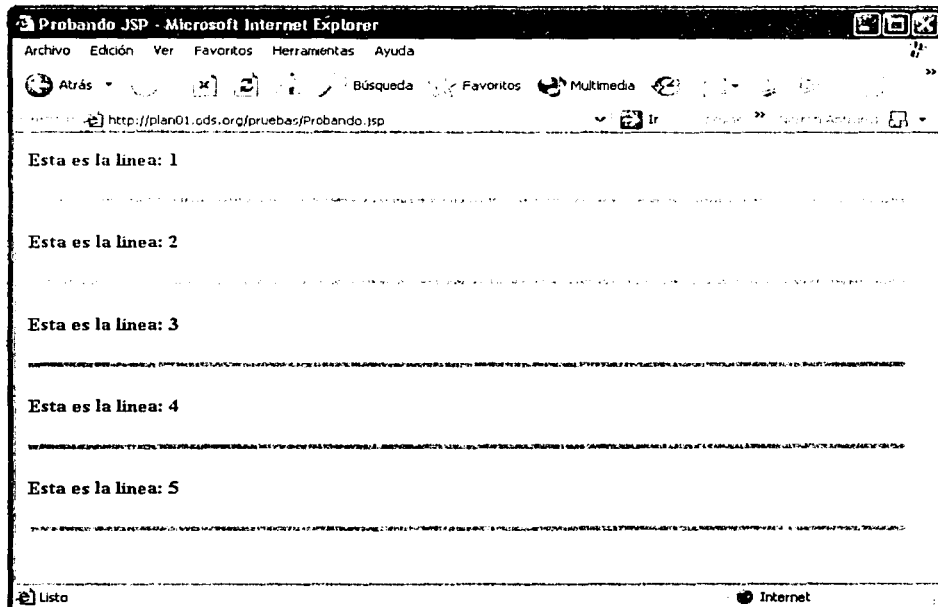


Fig. 4.5 Salida de la página JSP.

1.3 Programación del sistema propuesto

La implementación del sistema consta de varias pantallas que tienen una apariencia similar, por lo cual sólo se muestran algunas de ellas.

A continuación se muestra la programación de la página HTML de inicio del sistema y su correspondiente salida en el navegador.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
  <title>Informe de Labores</title>
  <link rel="stylesheet" href="Estilos1.css" type="text/css">
  <script language="JavaScript1.2" type="text/javascript">
  <!--
  function cambia(campo)
  {
    campo.value=campo.value.toUpperCase();
  }
  //-->
  </script>
</head>
<body style="color: Navy;">
<hr style="color: Navy;" />
<h2 align="center" style="font-size:30px;">Unidad de Planeación</h2>
<h3 align="center" style="font-size:20px;">Informe de Labores de los Académicos</h3>
<hr style="color: Navy;" />
<br />
<form name="frmInicio" method="post" action="/informe/Acceso.jsp">
<table cellpadding="10" cellspacing="10" align="center">
<tr>
<td>
  
</td>
</tr>
<tr>
<th>Inicio de Sesión</th>
</tr>
<tr>
<td>Login:</td>
<td>
  <input type="text" name="txtLogin" size="15" maxlength="10" onchange="cambia(this)">
</td>
</tr>
<tr>
<td>Password:</td>
<td>
  <input type="password" name="txtPassword" size="15" maxlength="10">
</td>
</tr>
<tr>
<th colspan="2">
  <input type="submit" name="cmdAccion" value="Entrar a Sesión">
</th>
</tr>
</table>
</form>
<hr style="color: Navy;" />
</body>
</html>
```


La salida del Navegador se muestra en la Fig. 4.6

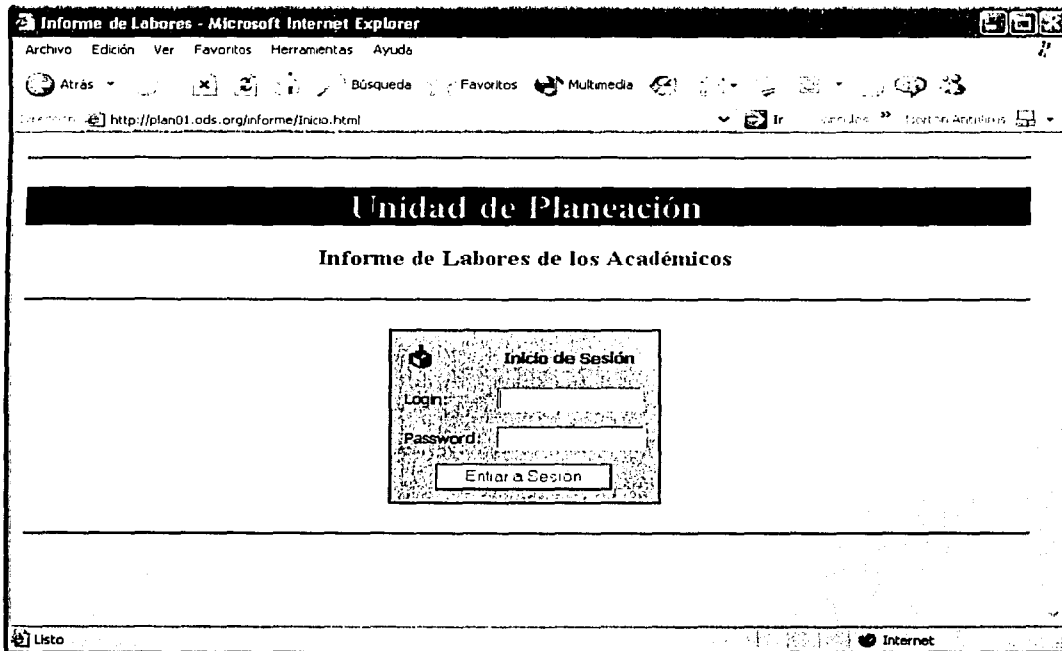


Fig. 4.6 Salida de la página de inicio HTML.

La programación de la página JSP del Menú Informe se muestra a continuación y su correspondiente salida en el navegador. Fig. 4.7

```

<%@ page session="true" %>
<%
    HttpSession sesion = request.getSession(true);
    if(sesion.getAttribute("profesor")==null)
    {
%>
        <jsp:forward page="SinSesion.jsp"/>
<%
    }
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>
<head>
<TITLE>Informe de labores</title>

<link REL="stylesheet" HREF="Estilos.css" TYPE="text/css">
<link rel="stylesheet" href="Estilos1.css" type="text/css">
</head>
<body style="font-size: 12px;">

<script language="JavaScript" src="Codigo1.js">
</script>

<script language="JavaScript" src="Menus.js">
</script>

<script language="JavaScript" src="Codigo2.js">
</script>

<script language="JavaScript" src="Crear.js">
</script>
<br><br><br><br><br><br>
<br><br><br><br><br><br>
<br><br><br><br><br><br>
<br><br><br><br>
<br><br><br><br>
<table align="center" CELSPACING="10">
<tr>
<td>
</td>
<td>
<IMG SRC="Clipart/imagenes/menu_undo_24.gif" ALT="Regresar">
</td>
</tr>
<tr>
<td>
<a href="/informe/Seleccion.jsp">Inicio</a>
</td>
<td>

</td>
</tr>
<tr>
<td>
<a href="/informe/Cerrar.jsp">Cerrar Sesión</a>
</td>
<td>
</td>
</tr>
</tr>
</table>
</body>
</html>

```

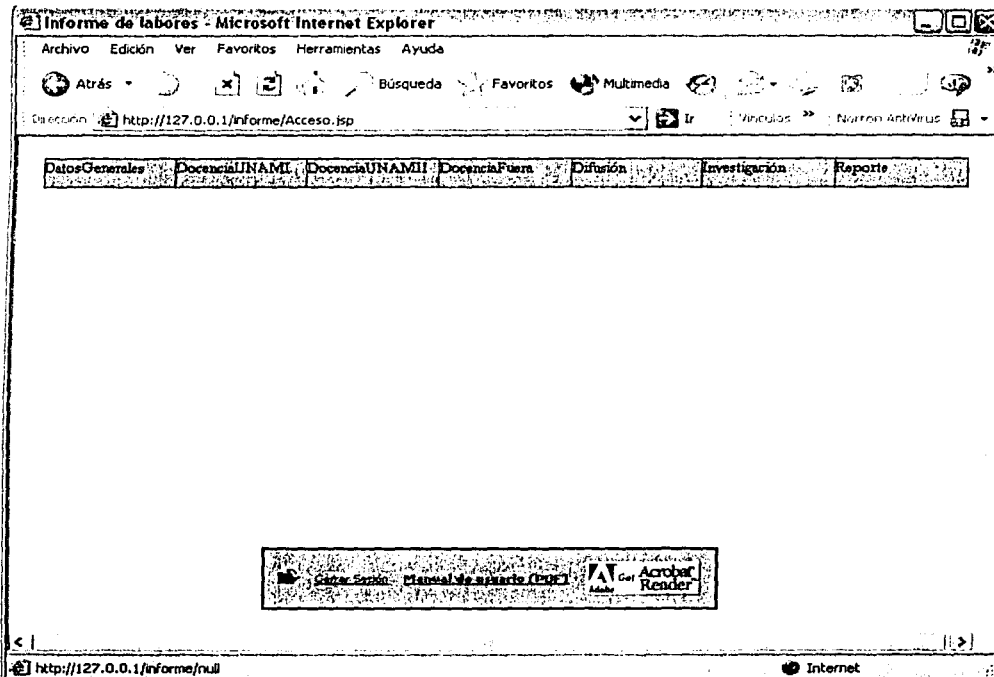


Fig. 4.7 Salida de la página JSP del Menú Informe.

Como se puede apreciar el Menú Inicio contiene varios submenús correspondientes a cada rubro manejado, seleccionando algún rubro en especial se muestra un formulario como el siguiente, Fig.4.8.

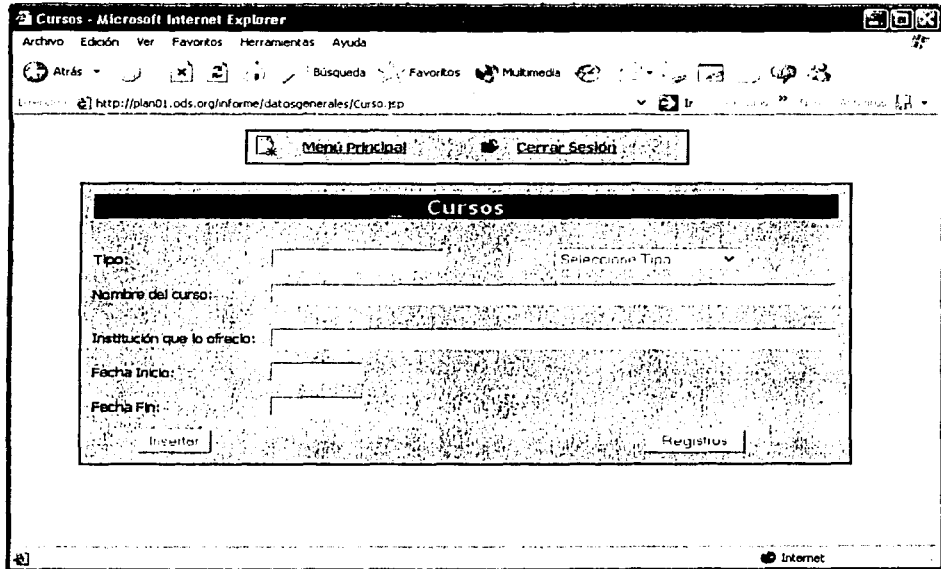


Fig. 4.8 Página JSP Curso.

Al llenarse los datos del formulario y oprimirse la opción de Insertar (se pueden insertar más de un registro), aparece el mensaje de que la acción se realizó con éxito, Fig. 4.9

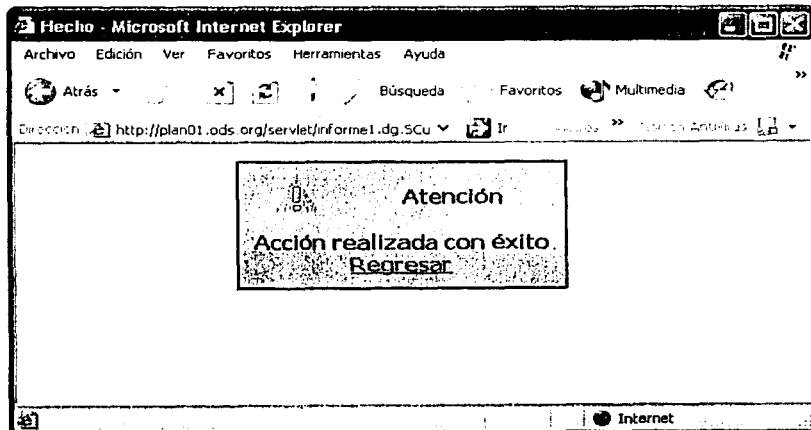


Fig. 4.9. Mensaje.

Al regresar y oprimir la opción de registros, se pueden ver los registros que se tienen hasta el momento. Fig. 4.10.

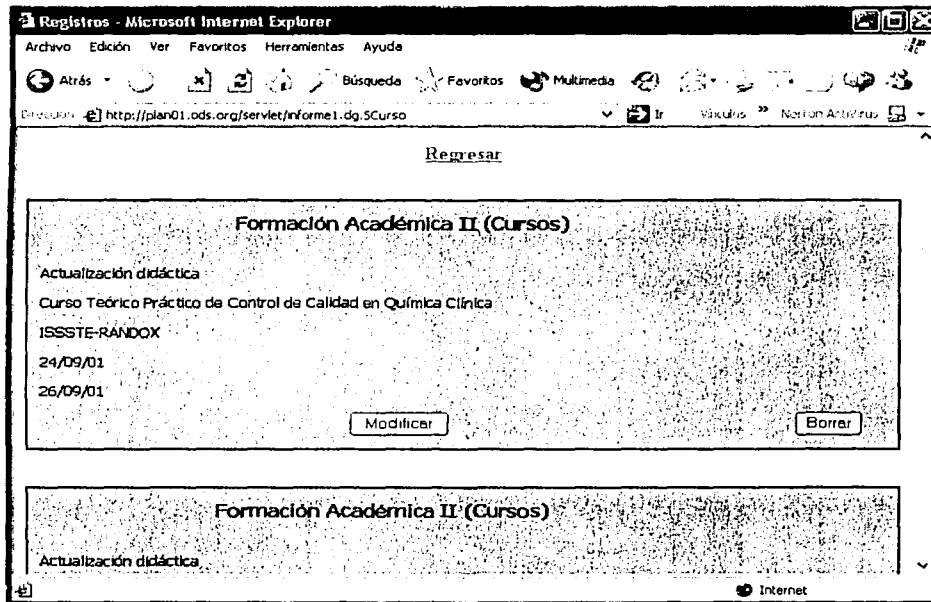


Fig. 4.10. Página JSP de los Registros insertados.

En la pantalla anterior se tiene la opción de Borrar el registro, en el caso de elegir esa opción el mensaje que se muestra es el de la figura 4.9.

Si se elige Modificar la pantalla de la figura 4.11, permite realizar correcciones a cualquier campo del registro, que permite actualizar los cambios realizados, crear un nuevo registro (es decir, insertar un nuevo registro) o ver los registros (para comprobar que la actualización se realizó).

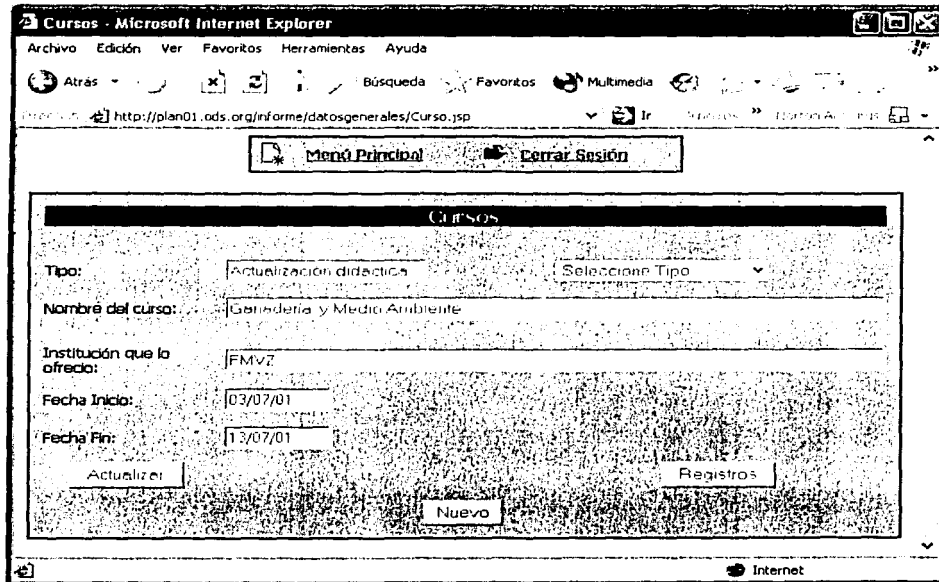


Fig. 4.11. Página JSP de Modificar Registro.

La sesión se mantiene abierta por un tiempo limitado, esto es por seguridad de la información del usuario. Cuando una sesión se cierra se muestra un mensaje que permite iniciar la sesión lo que implica introducir nuevamente la clave de acceso. Fig. 4.12

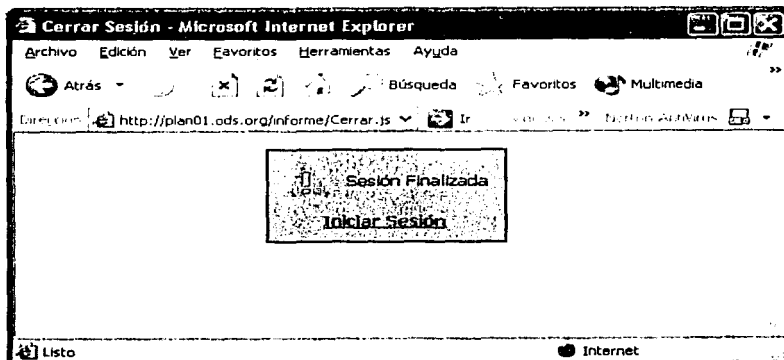


Fig. 4.12. Página JSP Sesión Finalizada.

Si el usuario desea crear un reporte, tiene que seleccionar del menú principal la opción de "Reporte" y posteriormente "Crear Reporte", esto lo llevara a la pantalla que se muestra en la figura 4.13, esta opción le permite crear un reporte en formato HTML o WORD2000.

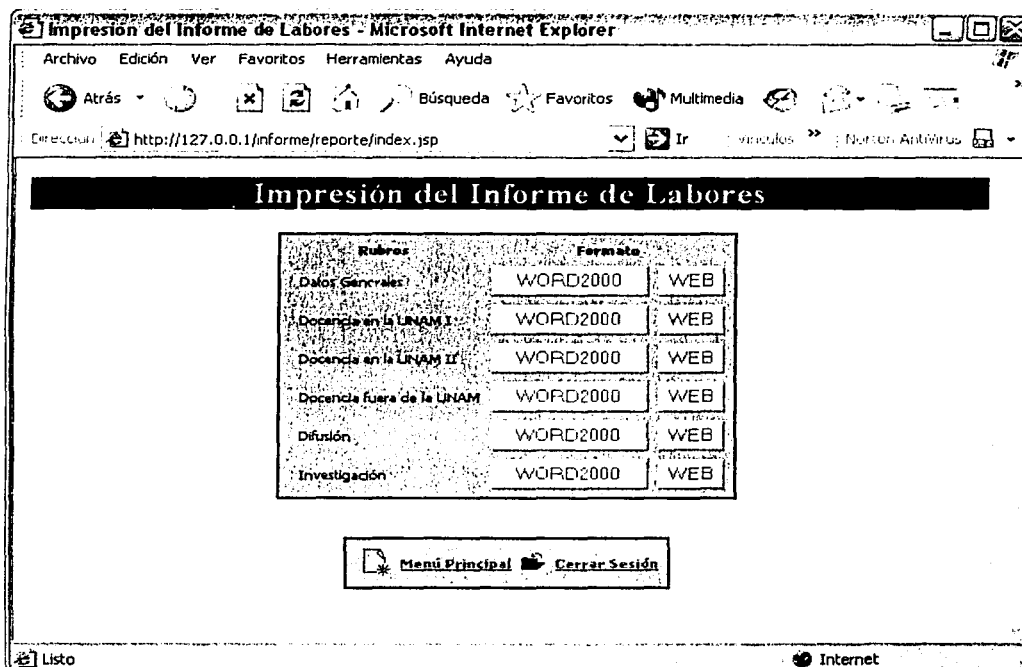


Fig. 4.13. Página JSP Crear Reporte.

Pruebas

CAPÍTULO 5. Pruebas

1. Pruebas del sistema

Las pruebas constituyen una parte integral y vital del ciclo de vida del desarrollo de sistemas. Se realizan con el propósito de descubrir defectos y se establecen para mejorar la calidad del sistema.

Los fundamentos de las pruebas definen los objetivos esenciales para las pruebas del software, que son un elemento crítico para la garantía de calidad y representa un último repaso de la especificaciones, del diseño y de la codificación.

1.1 Diseño de casos de prueba

Cualquier producto de ingeniería puede ser probado de dos formas:

- Conociendo la función específica para la que fue diseñado el producto, se pueden llevar a cabo pruebas que demuestren que cada función es completamente operativa.
- Conociendo el funcionamiento del producto, se pueden desarrollar pruebas que aseguren que "todas las piezas encajan"; es decir, que la operación interna se ajusta a las especificaciones y que todos los componentes internos se han comprobado de forma adecuada.

Los diseños de casos de pruebas se pueden dividir en:

Pruebas de Caja Blanca

Pruebas basadas en el conocimiento sobre la lógica y estructura interna. Usualmente dirigidas a la lógica.

Pruebas de Caja Negra

Pruebas funcionales basadas en los requerimientos sin conocimiento sobre como fue construido el sistema y usualmente dirigidas a los datos.

1.2 Categoría de pruebas

Las categorías de pruebas se deducen directamente de la lista de requerimientos funcionales y estructurales del plan de pruebas. Las especificaciones de las categorías de pruebas a usar son importantes ya que basados en ellas podemos determinar la infraestructura de pruebas requeridas. Las categorías de las mismas pueden ser: Unitarias, de Integración, de Regresión, de Concurrencia, de Volumen, de Aceptación, Manuales o Automáticas.

Pruebas unitarias

Se realizan sobre un programa o objeto con la intención de encontrar problemas funcionales en la lógica, así como problemas técnicos en el código. La prueba de unidad centra el proceso de verificación en la menor unidad del diseño del software. Usando la descripción del diseño detallado como guía, se prueban los caminos del control importantes, con el fin de descubrir errores dentro del ámbito del objeto y sus métodos..

Pruebas de integración

Son pruebas realizadas a un grupo de objetos para asegurar que los mensajes sean pasados adecuadamente entre objetos. La prueba de integración es una técnica sistemática para construir

la estructura de la aplicación mientras que, al mismo tiempo, se lleva a cabo para detectar errores asociados con la interacción, el objetivo es tomar los objetos probados en unidad y construir una estructura de aplicación que este de acuerdo con lo que dicta el diseño.

Pruebas de regresión

Son pruebas selectivas para detectar fallas que se hayan introducido durante las modificaciones de un sistema o componente, que permiten verificar que estas modificaciones no impacten de forma negativa y que se siga cumpliendo con los requerimientos planteados.

Pruebas de volumen

Se usan para verificar el comportamiento adecuado y eficiente de una aplicación bajo unas condiciones de volumen (número de operaciones), competencia de recursos (conurrencia) y carga máxima (velocidad de petición de ejecución de una operación) así como el comportamiento eficiente bajo las condiciones de volumen máximo (cantidad de datos) en las aplicaciones.

Pruebas de aceptación del sistema

Pruebas finales ejecutadas por el usuario, para asegurar que el sistema satisfaga las necesidades de la organización y Usuario Final.

Pruebas Estáticas

Consiste en la revisión y validación de los documentos generales en las distintas fases del ciclo de vida de un proyecto. Verificación realizada sin ejecutar el código del sistema.

Pruebas Estructurales

Se valida la arquitectura del sistema confirmando que todas sus partes funcionen sincronizadamente y que la tecnología está siendo usada apropiadamente. Se refiere a las características técnicas, como su comportamiento con grandes volúmenes de información, tiempos de respuestas, etc.

Pruebas Funcionales

Validan los requerimientos de la organización, pretenden descubrir estos errores cometidos en la implantación de dichos requerimientos.

1.3 Consideraciones importantes para las pruebas

- **Riesgos y suposiciones para las pruebas:**
Los riesgos son aquellos factores que pueden afectar negativamente la ejecución de las pruebas. Las suposiciones son las premisas que pueden afectar positiva o negativamente la ejecución de las pruebas complicando o facilitando las actividades de pruebas.
- **Condiciones y restricciones:**
Generalmente son limitaciones o problemas de naturaleza técnica y están relacionadas con el desarrollo del proyecto en sí.
- **Cobertura operacional de las pruebas:**
Se deben describir y listar de manera clara y concisa las operaciones a probar, así como aquellas operaciones a no ser probadas aún siendo parte del proyecto, ya que son necesarias especialmente cuando se requiere explicar el porqué de su exclusión, definiendo el alcance de las pruebas y delimitando responsabilidades

1.4 Pruebas realizadas

Las pruebas realizadas en este proceso al sistema de Informe de Labores son las siguientes:

Pruebas de Volumen.

Problema:

Al momento de verificar el comportamiento del sistema, se observó un acceso a la base de datos muy concurrido, lo que complicó la ejecución adecuada de ciertas operaciones.

Solución:

Para tener un mejor control de acceso a la base de datos se optó por crear lo que se llama un Pool de conexiones, esto es, al iniciar el sistema se crean automáticamente un número determinado de conexiones, los usuarios hacen uso de ellas al ejecutar una acción sobre la base de datos, al terminar dicha acción se libera la conexión para que otro usuario pueda usarla.

Pruebas de Aceptación.

Problema

El usuario final, mostró un descontento en la interfaz y la limitación de acciones a realizar, así como la falta de un manual para la operación del sistema.

Solución

En este caso se realizó una interfaz sencilla en la cual el usuario puede manipular por completo su información, teniendo mayor número de opciones. Se creó un manual que explica de forma gráfica el funcionamiento del sistema, así como el tipo de información que deben de capturar en cada rubro, ya que llegaban a confundirlos.

Pruebas Estructurales.

Problema

El sistema mostró un retraso en tiempo para la operatividad de los datos, debido a la concurrencia que sobrepasó las expectativas iniciales.

Solución

El problema de velocidad se mejoró al montar el sistema en un servidor de mayor capacidad y velocidad, así como reducir el número de conexiones a la base de datos.

Pruebas Funcionales.

Problema

El sistema generó errores en la inserción de información en la base de datos.

Solución

Se requirió ampliar el tamaño de algunos campos y modificar el formato de fechas que con anterioridad se había restringido.

Conclusiones

Conclusiones

Los sistemas en Internet representan un desarrollo tecnológico avanzado, ya que se han convertido en una herramienta útil en el intercambio de información en estos tiempos donde la tecnología crece a pasos agigantados. Las aplicaciones cliente/servidor proporcionan servicios a un número grande de clientes conectados a través de la red.

Nuestro sistema tiene las bondades del Internet, permitiendo el fácil acceso si se cuenta con una PC conectada a esta red, por lo que el problema referente a distancia, ubicación y traslados innecesarios queda solucionado. El sistema está disponible todo el año para modificar, agregar, borrar y revisar información, ocasionando que la entrega del informe sea oportuna; además, permite la creación de reportes en formatos diferentes siendo estos de gran utilidad para el académico.

La información que contiene el sistema esta organizada por rubros bien definidos, de tal manera que el académico puede identificar el tipo de datos que contienen.

Unos de los problemas más comunes en la elaboración de sistemas es que sólo pueden funcionar en una plataforma determinada, en el caso del sistema de Informe de Labores esto no es un problema pues es fácil de transferir a cualquier otra ya que está desarrollado en un lenguaje multiplataforma.

Es un sistema extensible, si se requiere agregar nuevos campos a la base de datos sólo es necesario crear las pantallas y clases necesarias para su manejo, como todo el acceso a la base de datos está controlado por una sola clase sólo es necesario extenderla para tener acceso a los recursos de la base de datos.

El sistema es un medio útil para concentrar información referente a las actividades de los académicos de una manera más rápida y sencilla, su uso no requiere de un personal avanzado ni de capacitación especial. Lo que permite a la Unidad de Planeación manipular para usos específicos la información recolectada.

Analizar la información es más rápido y sencillo, se estandarizaron varios conceptos como nombres de departamentos, materias, tipos de estímulos, RFCs y categorías, esto permite al académico ahorrar tiempo en la introducción de algunos campos.

Este trabajo fomentó la habilidad de investigación para adquirir información útil en su desarrollo, ayudó a incrementar nuestros conocimientos en el área de aplicaciones basadas en Internet y las implicaciones que esto requiere.

Una de las habilidades que desarrollamos más fue la de trabajar en equipo, aportando ideas, transfiriendo conocimientos y creciendo de manera profesional y humana.

Cuenta con un sistema de ayuda en línea que ha trabajado adecuadamente para resolver las dudas de los usuarios en el llenado de los datos requeridos.

El sistema está construido en un lenguaje que avanza a pasos agigantados, pero que permite actualizar las aplicaciones sin tener que realizar un cambio en la estructura de las mismas, por lo cual se puede garantizar la futura actualización de nuestro trabajo.

Bibliografía

- Adoración de Miguel, Mario Piattini, Esperanza Marcos
Diseño de Bases de Datos Relacionales
Alfaomega
México, 2000.
- Bobadilla Jesús, Alejandro Alcocer, Santiago Alonso y Abraham Gutiérrez
HTML Dinámico ASP y JavaScript a través de ejemplos
Alfaomega
México, 2000.
- Booch Grady, James Rumbaugh, Ivar Jacobson
El Lenguaje Unificado de Modelado
Addison Wesley Iberoamericana
Madrid, 1999.
- Cisneros González José Luis
Panorama sobre bases de datos (Un enfoque práctico)
Universidad Autónoma de Baja California
México, 1998
- Esteban Trigos García
Guía práctica para usuarios. JSP
Anaya
España, 2001.
- F. Javier Moldes
Guía práctica para usuarios. JAVA 2
Anaya
España, 2000.
- Foculer Martin , Kendall Scott.
UML gota a gota
Addison Wesley Longman de México, S.A. de C.V.
México, 1999.
- Lucas Gómez Angel, Paloma Romera García, Ma. Victoria Fraile Dates,
Fco. José Argente del Castillo, Antonio Alfaro Pesa
Diseño y Gestión de Sistemas de Bases de Datos
Paraninfo
España, 1993.
- Marty Hall
Servlets y JavaServer Pages. Guía Práctica
Prentice Hall
México, 2001
- Orfali Robert, Dan Harkey-Jerl Edwards
Cliente/Servidor. Guía de supervisión 2da. Edición
McGraw-Hill
México, 1998.
- <http://www.conallen.com/technologyCorner/webextension/>
Extensión de UML para la WEB

Anexos

Anexo A: Manual de usuario

A continuación realizaremos un recorrido por el sistema "Informe de labores", con el fin de explicar la forma de interactuar con él, también se da una explicación sobre las diferentes pantallas y el fin de cada una.

Ventana "Presentación"

Al momento de abrir el sitio del sistema aparece la presentación, Fig. M0, en esta pantalla aparece una liga que dice "**Inicio**" la cual nos lleva a la pantalla de Inicio de sesión.

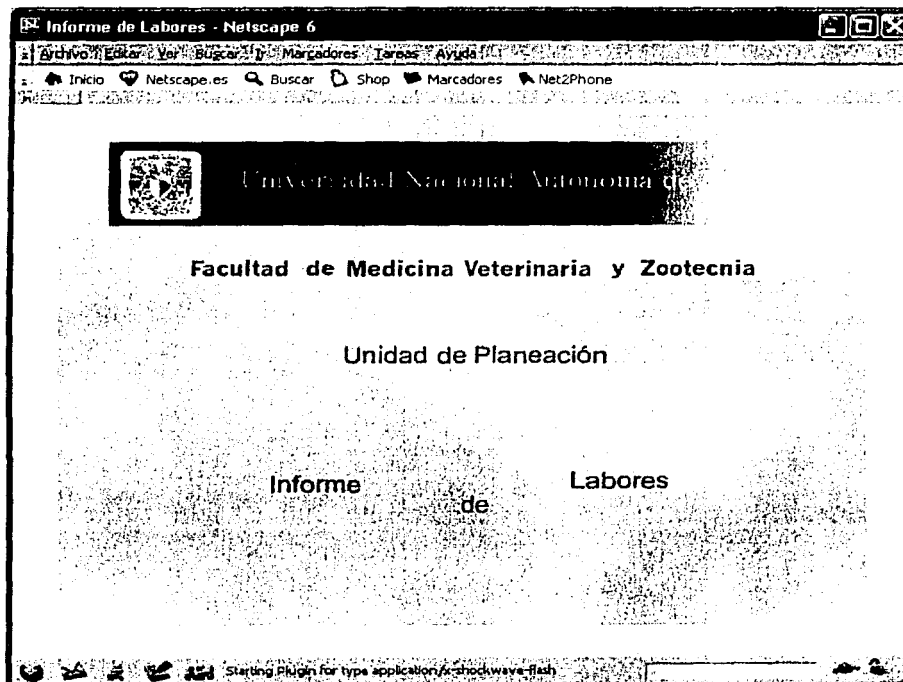


Fig. M0 Ventana "Presentación"

Ventana "Inicio"

Esta ventana es nuestra puerta de entrada al sistema, aquí nos registramos proporcionando nuestro login y password, posteriormente debemos pulsar el botón "Entrar a Sesión" para continuar, Fig. M1.



Fig. M1 Ventana "Inicio"

Ventana "No Acceso"

Si los datos que proporcionamos no son los correctos, el sistema nos mostrará una advertencia indicándonoslo, Fig. M2.

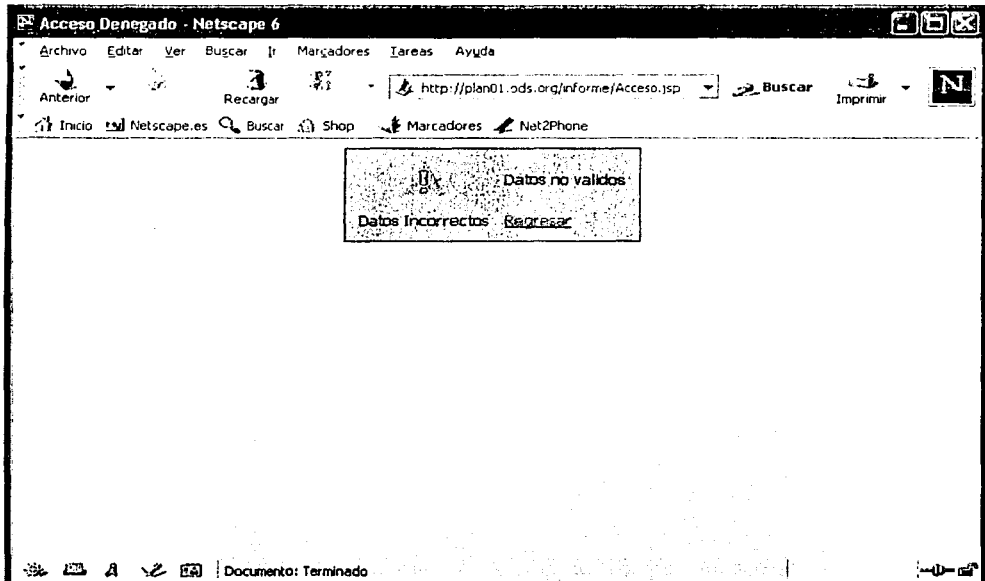


Fig. M2 Ventana "No Acceso"

Ventana "Menú Informe"

Por el contrario, si los datos que proporcionamos son correctos, el sistema nos mostrará el menú principal, desde aquí podemos acceder cualquier rubro, Fig. M3.

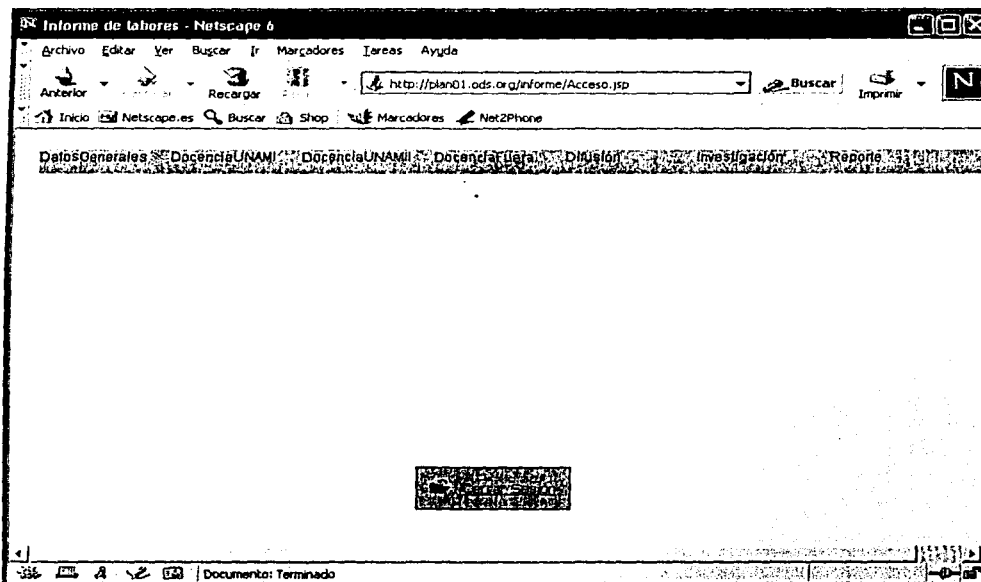


Fig. M3 Ventana "Menú Informe"

Seguimiento de la sesión

El sistema mantiene un seguimiento del usuario, si se deja de interactuar con el sistema por un lapso de más de 10 minutos mostrará un mensaje informando que la sesión terminó Fig. M4, esto se hace por la seguridad e integridad de la información, es importante que si se quiere dejar de interactuar con el sistema se cierre la sesión en curso, en el menú principal en la parte inferior aparece una liga que nos permite cerrar nuestra sesión, esta liga aparece en cada uno de los rubros que veremos más adelante.

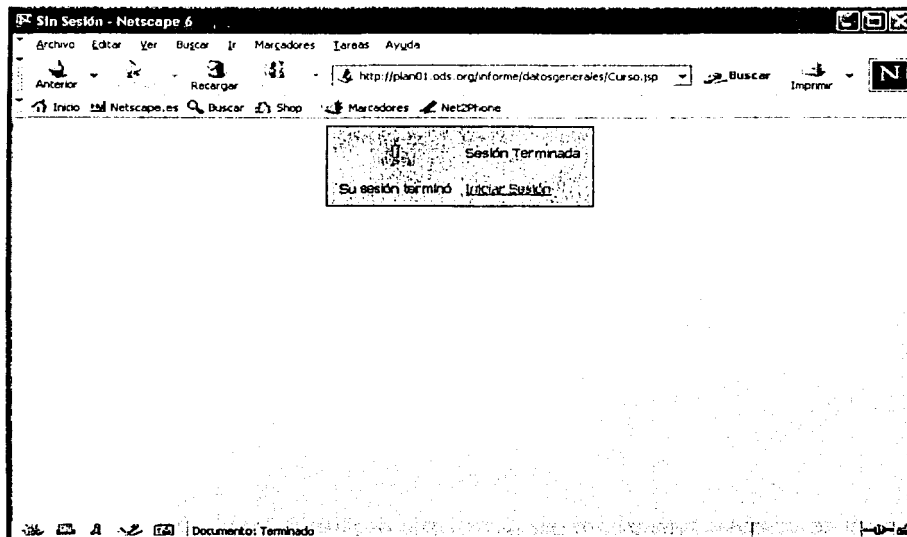


Fig. M4 Ventana "Sesión Terminada"

Ventanas rubro

Cada rubro que maneja el sistema tiene asociada una ventana, todas estas ventanas siguen una misma filosofía y cuentan con una misma estructura, tomemos el rubro "Cursos" como ejemplo, Fig. M5:

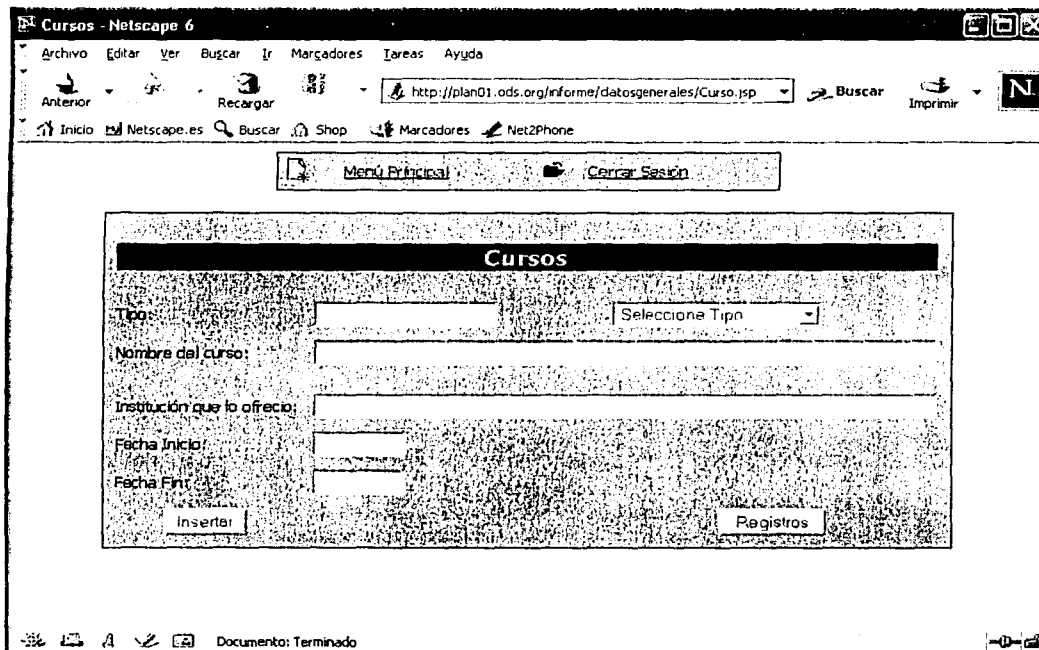


Fig. M5 Ventana "Cursos"

Cada ventana tiene una pequeña barra en la parte superior la cual cuenta con dos opciones:

Menú Principal:

Esta opción permite regresar al menú principal del sistema, ver Fig. M3.

Cerrar Sesión:

Permite cerrar la sesión en curso.

Podemos observar dos botones en el formulario que están situados en la parte inferior:

Botón "Insertar"

Permite insertar un nuevo registro después de terminar el llenado del formulario.

Botón "Registros"

Si lo que queremos es ver los registros que tenemos guardados pulsamos este botón.

Anexo A: Manual de usuario

A continuación se muestran otras ventanas para que veamos la similitud que existe entre ellas, Fig. M6 y M7.

The screenshot shows a Netscape 6 browser window titled "Exámenes Extraordinarios". The address bar contains the URL "http://plan01.ods.org/informe/docencia2/Extra.jsp". The page features a navigation bar with "Menú Principal" and "Cerrar Sesión". The main content area is titled "Exámenes Extraordinarios" and contains a form with the following fields: "Selección" (a dropdown menu), "Materia" (a text input field), and "Fecha" (a text input field). There are "Insertar" and "Registrar" buttons at the bottom of the form. The status bar at the bottom indicates "Documento: Terminado".

Fig. M6 Ventana "Exámenes Extraordinarios"

The screenshot shows a Netscape 6 browser window titled "Asistencia a Congresos". The address bar contains the URL "http://plan01.ods.org/informe/difusion/Congreso.jsp". The page features a navigation bar with "Menú Principal" and "Cerrar Sesión". The main content area is titled "Asistencia a Congresos" and contains a form with the following fields: "Nombre" (a text input field), "Lugar" (a text input field), "Tipo" (a dropdown menu), and "Fecha" (a text input field). There are "Insertar" and "Registrar" buttons at the bottom of the form. The status bar at the bottom indicates "Documento: Terminado".

Fig. M7 Ventana "Asistencia a Congresos"

Insertar un nuevo registro

Después de llenar un formulario debemos insertar el registro pulsando el botón "Insertar", si la acción se lleva a cabo correctamente el sistema nos muestra un mensaje como el que sigue, Fig. M8.

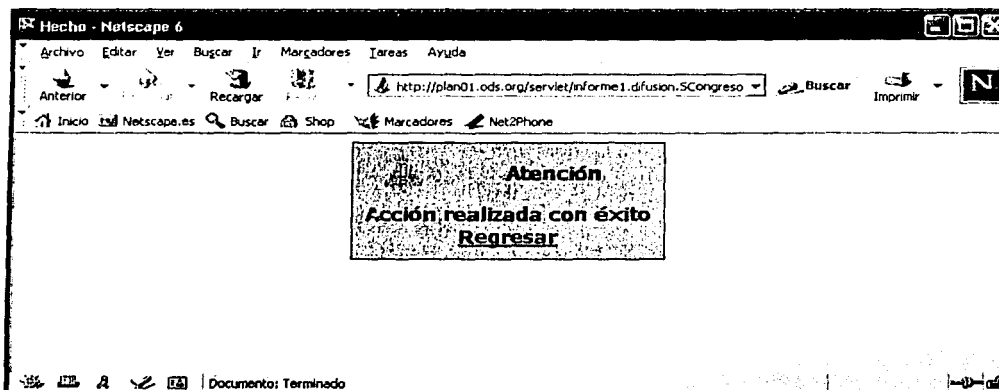


Fig. M8 Ventana "Acción realizada con éxito"

Por el contrario, si no se puede llevar a cabo la acción solicitada el sistema mostrará un mensaje como el siguiente, Fig. M9.

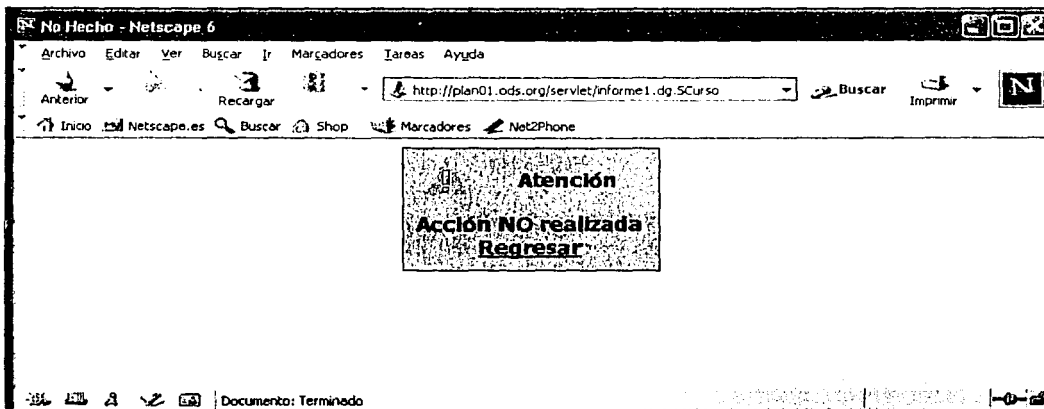


Fig. M9 Ventana "Acción NO realizada"

Ver registros existentes

En la ventana de un rubro dado se cuenta con la opción de ver los registros que se tienen almacenados, pulsamos el botón "Registros" y el sistema nos mostrará una ventana con todos los registros existentes, Fig. M10.

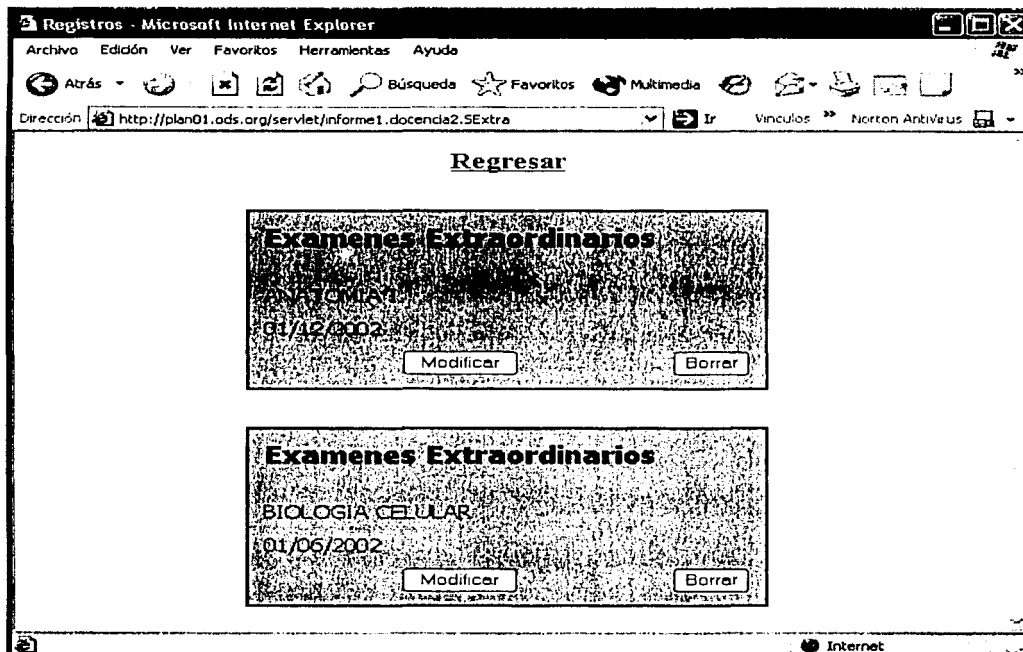


Fig. M10 Ventana "Registros"

La ventana "Registros" cuenta con la opción de regresar al rubro en el que se esta trabajando, también se muestran los registros existentes, cada uno con dos botones: Modificar y Borrar.

Borrar:

Permite borrar el registro, si la acción se realiza con éxito se muestra una ventana indicándolo, ver figura M8, pero si no se puede borrar el registro aparece una ventana como la de la figura M9.

Modificar:

Esta opción nos permite cargar el registro en la ventana correspondiente para que lo podamos editar.

Modificar un registro

Desde la ventana "Registros" podemos seleccionar el registro que deseamos modificar, al hacer esto el registro se carga en la ventana correspondiente, Fig. M11.

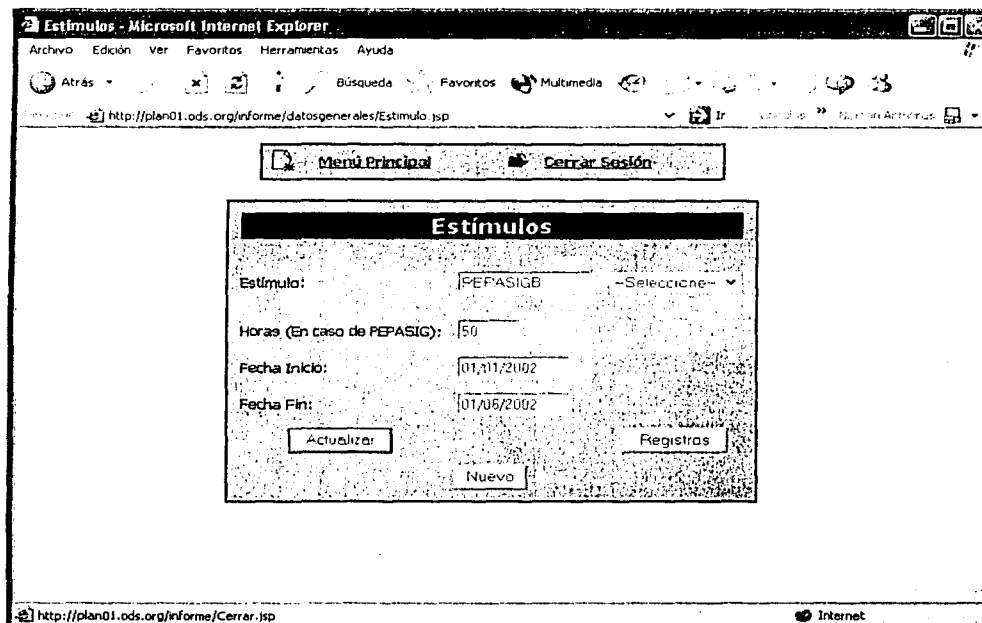


Fig. M11 Modificando un registro

Vemos que la ventana "Rubro" cambia, ahora aparecen tres botones: Actualizar, Nuevo, Registros.

Registros:

La funcionalidad de este botón ya la conocemos.

Actualizar:

Al modificar un registro debemos guardar los cambios ya que en caso contrario el registro no será modificado, este botón permite realizar esta acción.

Nuevo:

Si ya no queremos modificar el registro podemos pulsar este botón, el registro desaparece del formulario, pero no de la base de datos, al pulsar este botón la ventana toma la forma como la de la figura M5.

Todas las ventanas de los distintos rubros tienen la misma funcionalidad.

Crear Reporte

Para la creación e impresión de un reporte es necesario abrir la ventana de Reporte desde el menú principal, seleccionando la opción "Reporte" y a continuación "Crear Reporte", esto lo llevará a una ventana similar a la figura M12, aquí puede seleccionar cualquiera de los dos formatos disponibles para crear su reporte.

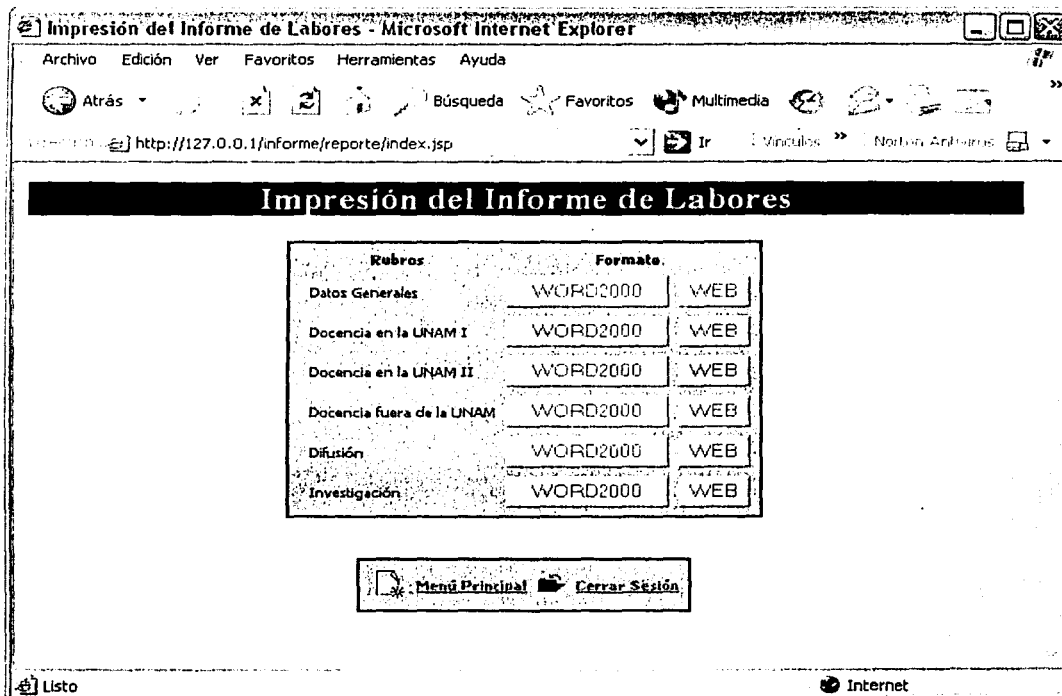


Fig. M12 Ventana Reporte