UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

SISTEMA DE INFORMACIÓN DE USUARIOS

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A

EDGAR EDUARDO GARCÍA CANO CASTILLO

DIRECTOR DE TESIS:
ING. ALEJANDRO VELÁZQUEZ MENA

CIUDAD UNIVERSITARIA 2007

Dedicatorias

A mi madre Hortencia, por ser un gran ejemplo de fuerza y motivación para seguir adelante a pesar de las adversidades y por darme su apoyo incondicional.

A mi padre Eduardo, por brindarme parte de los medios para completar mi educación.

A mi hermano Eric, que forma parte fundamental en mi vida y que espera le siga echando muchas ganas para cumplir sus metas.

A toda mi demás familia, los que viven y a la memoria de los que ya no están con nosotros; a todos ellos gracias por el apoyo, por los buenos consejos y por el cariño que me han demostrado a lo largo de mi vida.

A Alejandro Velázquez Mena, Dan Schmidt, Julio César Saynez, Eduardo Ríos y Alberto Conrado, Ramiro Vázquez, Carlos Zamitiz, Ricardo Manríquez, Iván Figueroa, Beatriz Ordóñez, por su amistad y vivencias a lo largo de toda mi carrera. En general, para todas aquellas personas que han entrado y salido de mi vida, pero que contribuyeron con su granito de arena en ella.

A Amellali Manjarrez, que últimamente me ha brindado su apoyo, cariño y comprensión.

Agradecimientos

A la Universidad Nacional Autónoma de México.

A la Facultad de Ingeniería.

A PROTECO.

Al Ing. Alejandro Velázquez Mena.

Al Ing. Carlos Alberto Román Zamitiz.

A todos estos, por el apoyo y conocimientos brindados para mi formación académica.

A los sinodales, Dra. Ana María Vázquez Vargas, M.I. Aurelio Adolfo Millán Nájera, M.I. Jorge Valeriano Assem por la contribución de sus comentarios.

Reflexiones

La gente suele decir que la vida es sólo una línea recta, pero si cambiamos la perspectiva y en lugar de ver esa línea de frente, la observamos desde arriba, la veríamos de manera diferente.

Lo que veríamos en esa ese línea, que llamamos vida, es que tenemos intersecciones con otras líneas, estas intersecciones se dan cuando hemos coincidido con la vida de otra persona y que por un momento pareciera que las dos líneas van en forma paralela y parecen juntarse y ser una sola, pero de repente suelen separarse e irse en diferente camino.

Hay otras líneas que van en forma paralela, las más cercanas son los familiares, los cuáles siempre están a nuestro lado, pase lo que pase. Otras un poco más alejadas son los amigos, que en algunas ocasiones, la distancia que no separa puede agrandarse o acortarse, pero a fin de cuentas, uno sabe que están ahí.

También tenemos quiebres, cada uno de esos éstos, muestra las decisiones que hemos tomado, tanto correctas como incorrectas, pero que finalmente para bien o mal nos hicieron cambiar el rumbo que llevábamos. De la infinidad de ángulos que podía escoger para generar un nuevo quiebre en la línea de mi vida, sólo uno fue el correcto y es el que me tiene en éste preciso instante, que termina una etapa, pero que es el inicio de otra mucho mejor.

Edgar Eduardo García Cano Castillo.

Controla tus pensamientos y se transformaran en tus palabras,
controla tus palabras y se transformaran en tus acciones,
controla tus acciones y se transformaran en hábitos,
controla tus hábitos y se transformaran en tu carácter.

Mahatma Gandhi.

Temario

Figuras

Tablas

Introducción

Este proyecto ideado por el Ing. Alejandro Velázquez Mena, surge como una necesidad de tener un sistema capaz de obtener datos estadísticos de los usuarios que ocupan los servicios del Laboratorio de Computación del edificio Luis G. Valdés Vallejo de la División de Ingeniería Eléctrica (DIE); así como las características con las que cuenta el mismo para proveer servicio.

Los puntos básicos con los que debe contar el sistema denominado "Sistema de Información de Usuarios" son los siguientes:

- ✓ Independientemente de la tecnología que se utilice, el sistema debe ser realizado para visualizarse vía interfaz Web, además de estar realizado con software libre, ya que no se cuenta con recursos para comprar licencias.
- ✓ Debe ser capaz de obtener los datos personales de todos los usuarios que pueden utilizar los servicios del Laboratorio de Computación de la DIE.
- ✓ Se debe poder hacer consultas sobre los recursos con los que cuenta el Laboratorio de Computación de la DIE.
- ✓ Se debe tener registros de los cursos curriculares y extracurriculares que se realizan en el Laboratorio de Computación de la DIE.
- ✓ La principal característica es que debe realizar gráficas que muestren la afluencia de los usuarios, tanto de los semestres anteriores de los cuales se tengan registros, así como del actual semestre.

Haciendo un refinamiento de los puntos anteriores se obtuvieron los siguientes lineamientos con los que se desarrollará el sistema.

1. Deberá obtener los datos sobre el software que se encuentra instalado en cada computadora con las que cuenta el laboratorio de computación, además de los sistemas operativos que se manejan.

2. Deberá mostrar los cursos y clases que se imparten en el laboratorio de computación, los cuáles se dividen de la siguiente manera:
    a. Clase semestral: Son las clases que son constantes y que se imparten o se han impartido en todo un semestre. Debe contar con los siguientes datos:
        i. Materia.
        ii. Profesor.
        iii. Fecha de inicio.
        iv. Fecha de terminación.
        v. Horario inicial.
        vi. Horario final.
        vii. Días.
    b. Clase ocasional: Son las clases que se dan de manera ocasionales y que se imparten o se han impartido en cierto semestre. Debe contar con los siguientes datos:

i. Materia.
ii. Profesor.
iii. Fecha de inicio.
iv. Fecha de terminación.
v. Horario inicial.
vi. Horario final.

c. Cursos: Son los cursos que se dan, ya sea durante el semestre o ínter semestralmente. Debe contar con los siguientes datos:
   i. Materia.
   ii. Profesor.
   iii. Fecha de inicio.
   iv. Fecha de terminación.
   v. Horario inicial.
   vi. Horario final.

3. Usuarios por semestre: Debe mostrar un reporte de en qué semestre se encuentran inscritos los usuarios en el laboratorio de computación. Los datos que se deben mostrar son los siguientes:
   a. Usuario.
   b. Nombre.
   c. Número de cuenta.

4. Usuarios por carrera. Debe mostrar un reporte de en qué carrera se encuentran inscritos los usuarios en el laboratorio de computación. Los datos que se deben mostrar son los siguientes:
   a. Usuario.
   b. Nombre.
   c. Número de cuenta.

5. Consulta de ingreso de usuarios al laboratorio se pueden generar diversos reportes con carrera, semestre, fecha de entrada, fecha de salida, horario de entrada y horario de salida en que los diversos usuarios hacen uso del laboratorio, cada combinación de dato debe tener un reporte donde se indique:
   a. Computadora ocupada por el usuario.
   b. Login del usuario.
   c. Fecha de entrada.
   d. Hora de entrada.

6. Se debe consultar el hardware con la que se cuenta para dar servicio a los usuarios del laboratorio de computación. Los datos que debe contener el reporte son:
   a. Marca.
   b. Procesador.
   c. Disco duro.
   d. Memoria RAM.
   e. Sistema operativo.

7. Se debe consultar el historial de cada usuario. Este reporte debe mostrar:

   a. Nombre del usuario.
   b. Número de cuenta.
   c. Fecha de registro.
   d. Usuario que dio el alta.
   e. Estado actual de su cuenta.
   f. Observaciones.

8. Debe mostrar las siguientes gráficas divididas de la siguiente manera:
   a. Semestrales: Esta gráfica indica el número de usuarios inscritos al laboratorio de computación, divididos por semestres.
   b. Carreras: Esta gráfica indican el número de usuarios inscritos al laboratorio de computación, divididos por carreras.
   c. Gráficas por semestre: Esta gráfica indica el número de accesos que se tuvo en cada mes dependiendo del semestre seleccionado.
   d. Gráficas mensuales del semestre AAAA-D: Esta gráfica muestra, dependiendo del mes, la afluencia de usuario por día del mes seleccionado, correspondiente a los meses del semestre que se encuentra transcurriendo.

9. Debe de haber una sección administrativa que se encargue de:
   a. Altas, bajas y cambios de usuarios.
   b. Altas, bajas y cambios que se requieran para administrar cada una de las opciones que se presentaron en los puntos anteriores, como lo son:
      i. Cursos.
      ii. Semestres.
      iii. Hardware.
      iv. Software.

10. Se debe contar con un repositorio de datos, el contendrá todos los datos necesarios para ser utilizados en cada uno de los puntos que se mencionaron anteriormente.

11. Tanto el repositorio de datos, el lenguaje de programación y el servidor de aplicaciones deben de ser software libre, ya que el laboratorio de computación no cuenta con recursos para comprar licencias.

12. El servidor que se utilizará será proporcionado por el laboratorio de computación y éste cuenta con el sistema operativo Linux.

13. La instalación de cada servicio que se requiere, como lo es el servidor de aplicaciones, la base de datos y el mismo sistema, serán instalados por el tesista.

14. La administración de todo lo que involucre al sistema correrá a cargo del tesista.

La realización del sistema se va a llevar a cabo por medio de ingeniería de software utilizando un método iterativo, el cual consiste en dividir el proyecto en etapas y hacer iteraciones sobre las mismas, esto con el fin de detectar problemas o realizar cambios y mejoras respecto de lo que se tiene.

En los siguientes capítulos se detalla el contenido del proceso que se siguió para llegar a la realización del producto final.

Capitulo 1: En este capítulo se hace una reseña sobre el significado de existir del laboratorio de computación dentro de la División de Ingeniería Eléctrica. También se hace un análisis sobre Ingeniería de Software, Servidores de Aplicación, Lenguajes de Programación, Almacenamiento de Datos y el Patrón Modelo – Vista – Controlador, conceptos que se utilizan a lo largo del presente trabajo.

Capítulo 2: Este capítulo está dedicado al análisis del sistema, se hace una definición de lo que es el sistema, como ayuda del análisis se ocupan diagramas de UML, cómo lo son los Diagramas de Caso de Uso y el Diagrama de Clases.

Capítulo 3: En este capítulo se realiza el diseño de la funcionalidad del sistema, esto se hace mediante Diagramas de Secuencia y de Actividades, que forman parte de UML. Se realiza el diseño de la Base de Datos, además se presenta la secuencia de pantallas del sistema.

Capítulo 4: En este capítulo quedan completamente especificados los estándares que se utilizarán en el sistema y que se definen en el Anexo A y B. Se realiza una reseña de las características de las tecnologías que se utilizan para el desarrollo del sistema, como lo son: el lenguaje de programación Java, SQL, gráficas, el formato y estilo de las vistas con la utilización de Hojas de estilo y Javascript; además se describe la manera en que se utilizaron para el desarrollo en la Tesis.

Capítulo 5: Este capítulo está enfocado ha la realización de pruebas tanto del servidor web, como a la aplicación. Se realizó una prueba de estrés al servidor web, simulando usuario y acciones que estos harían en el sistema, esto con el fin de que el servidor web tuviera buen soporte para mantener la aplicación, sin que diera de baja sus servicios o se saturara Se realizaron casos de prueba, para verificar que según datos específicos de entrada, se obtuviera como resultado, datos correctos. Finalmente se realizaron pruebas a todas las sentencias SQL, para verificar que cada consulta o actualización a la base de datos, se realizara correctamente.

Capítulo 6: El capítulo final de la tesis se dedica a la liberación del sistema. Se describen brevemente cada una de las etapas que aprobaron para su liberación.

1. Conceptos Básicos.

1.1. Antecedentes del Laboratorio de Computación de la DIE.

1.1.1. Visión

Seremos un Laboratorio que preste servicio de excelencia, para la formación de los futuros ingenieros en computación del país, incorporando las herramientas de último desarrollo para su auxilio.

1.1.2. Misión

Somos un Laboratorio responsable de servicio de cómputo a los alumnos de las carreras de la División de Ingeniería Eléctrica, apoyando a las asignaturas que se imparten el la Facultad de Ingeniería, con equipo de cómputo y software especializado.

1.1.3. Objetivo

Se encarga de promover y fomentar el aprendizaje de los alumnos de la División de Ingeniería Eléctrica como Administrar y mantener en óptimo desempeño la RED DIE para las actividades de académicos y alumnos.

1.1.4. Materias a las que atiende

Las correspondientes a la División de Ingeniería Eléctrica; por mencionar algunas: Bases de Datos, Bioingeniería, Compiladores, Computadoras y Programación, Estructuras de Datos, Estructuras de Discretas, Control Analógico y Digital.

1.1.5. Actividades

- Administración de servicios de Internet (WWW, Correo electrónico, ftp anónimo entre otros), mantenimiento de servidores de impresión, de base de datos, servidor de archivos.
- Asesoría y cursos de lenguajes de programación (HTML, Visual Basic, C, Java, etc.).
- Mantenimiento y administración de la red de la División de Ingeniería Eléctrica.

1.2. Ingeniería de Software

Ingeniería de Software es el establecimiento y uso de principios de ingeniería para obtener software que sea confiable y que funcione eficientemente en máquinas reales.
Es un conjunto de etapas parcialmente ordenadas con la intención de lograr un objetivo, que en este caso, es la obtención de un producto de software de calidad. El proceso de desarrollo de software es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para su uso operativo. Concretamente define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo.

1.2.1. Ciclo de vida de un producto

El proceso de desarrollo de software requiere por un lado un conjunto de conceptos, una metodología y un lenguaje propio. A este proceso también se le llama el ciclo de vida del software.
El ciclo de vida de un producto es una metodología a seguir para cualquier persona que busque hacer el desarrollo de cualquier producto. Está formado por seis pasos mencionados a continuación:

I. Análisis.
II. Diseño.
III. Desarrollo.
IV. Pruebas.
V. Implementación.
VI. Mantenimiento.

1.2.2. Análisis

El análisis es el proceso de investigación de un problema que se busca resolver con un producto, es el qué va ha hacer la aplicación.
Las fases con las que cuenta el análisis son:

- Definir claramente el problema que se busca resolver, esto nos lleva a encontrar el dominio del mismo, así como el alcance que va a tener.
- Identificar los componentes clave que definen el problema.

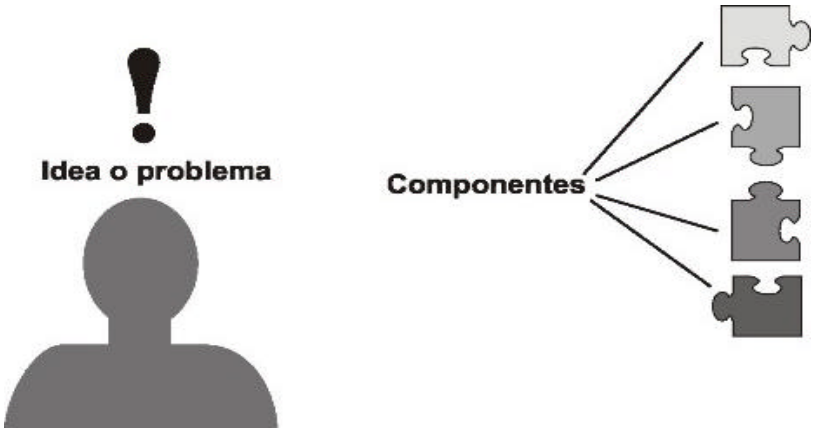


Figura 1.1 Fase de Análisis del ciclo de vida de un producto.

1.2.3. Diseño

El diseño es el proceso de aplicar los puntos clave obtenidos del análisis de previo; es la manera de cómo se va a resolver el problema. La tarea principal durante el diseño es construir plantillas de los componentes del sistema que se pretende realizar.

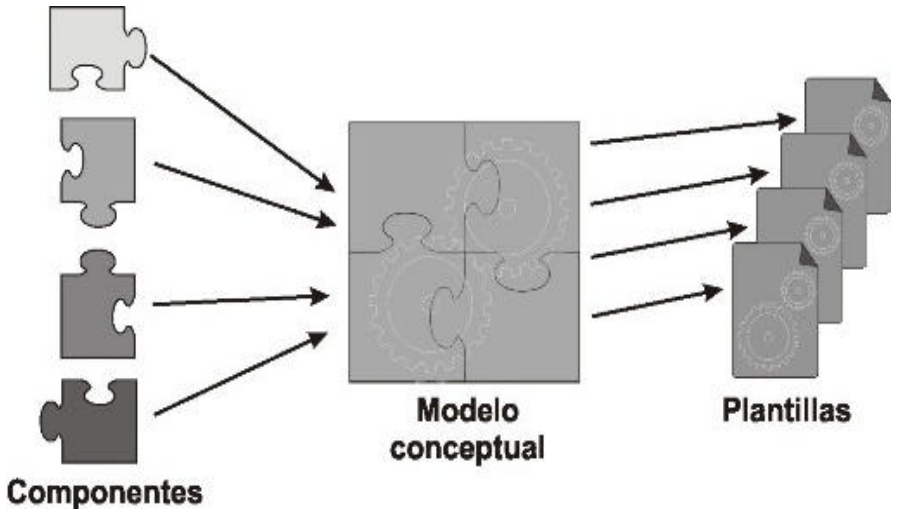


Figura 1.2 Fase de diseño del ciclo de vida de un producto.

1.2.4. Desarrollo

El desarrollo consiste en usar las platillas creadas durante el diseño para crear los componentes de la aplicación.

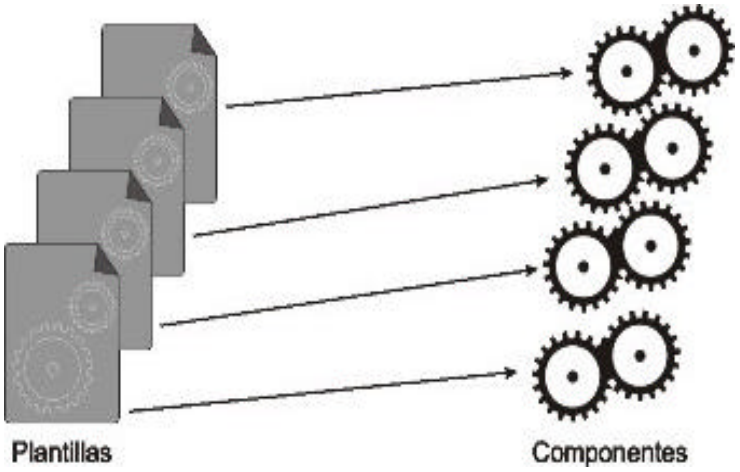


Figura 1.3 Fase de desarrollo del ciclo de vida de un proyecto.

1.2.5. Funcionalidad

La funcionalidad consiste en asegurar que los componentes, unidos como un todo para crear el producto, cumplan con los requerimientos y especificaciones creadas durante el diseño.

La funcionalidad es generalmente probada por los desarrolladores junto con el usuario final, para asegurar que se cumpla con los objetivos planteados en el análisis del producto.

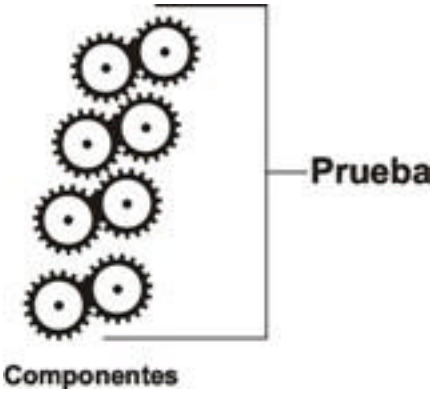


Figura 1.4 Fase de pruebas del ciclo de vida de un producto.

1.2.6. Implementación

La implementación consiste en hacer que el producto esté disponible para los usuarios. A esta fase también se le conoce como poner en producción un producto.

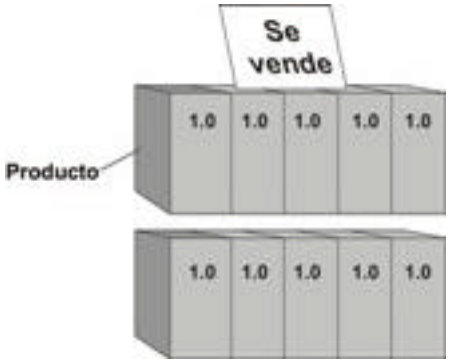


Figura 1.5 Fase de implementación del ciclo de vida de un producto.

1.2.7. Mantenimiento

El mantenimiento consiste en arreglar algún problema, hacer una actualización, revisión o versión del producto. Se descubren fallas y se hacen las mejoras.

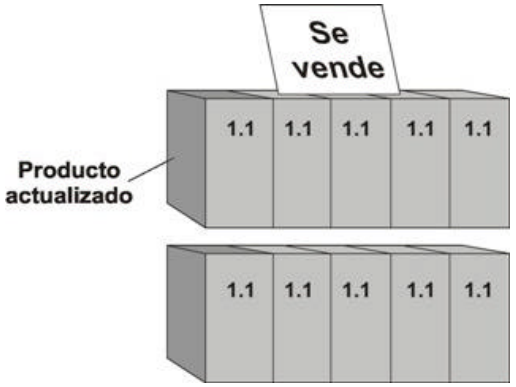


Figura 1.6 Fase de Mantenimiento del ciclo de vida de un producto.

1.3. Arquitectura de Software

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema, programa o aplicación y tiene la responsabilidad de:

- ✓ Definir los módulos principales.
- ✓ Definir las responsabilidades que tendrá cada uno de estos módulos.
- ✓ Definir la interacción que existirá entre dichos módulos.
- ✓ Control y flujo de datos.
- ✓ Secuenciación de la información.
- ✓ Protocolos de interacción y comunicación.
- ✓ Ubicación en el hardware.

La Arquitectura del Software aporta una visión abstracta de alto nivel, posponiendo el detalle de cada uno de los módulos definidos a pasos posteriores del diseño.

No es necesario inventar una nueva arquitectura software para cada sistema de información. Lo habitual es adoptar una arquitectura conocida en función de sus ventajas e inconvenientes para cada caso en concreto.

1.4. Servidores Web y de Aplicación

Un servidor web es un programa que implementa el protocolo HTTP (hypertext transfer protocol). Este protocolo está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML (hypertext markup language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonidos.

Un servidor web se encarga de mantenerse a la espera de peticiones HTTP llevada a cabo por un cliente HTTP que solemos conocer como navegador. El navegador realiza una petición al servidor y éste le responde con el contenido que él cliente solicita.

Sobre el servicio web clásico podemos disponer de aplicaciones web. Éstas son fragmentos de código que se ejecutan cuando se realizan ciertas peticiones o respuestas HTTP. Hay que distinguir entre:

- Aplicaciones en el lado del cliente: el cliente web es el encargado de ejecutarlas en la máquina del usuario. Son las aplicaciones tipo script, por ejemplo: vbscript o javascript. El servidor proporciona el código de las aplicaciones al cliente y éste, mediante el navegador, las ejecuta. Es necesario, por tanto, que el cliente disponga de un navegador con capacidad para ejecutar aplicaciones.
- Aplicaciones en el lado del servidor. El servidor web ejecuta la aplicación; ésta, una vez ejecutada, genera cierto código HTML; el servidor toma este código recién creado y lo envía al cliente por medio del protocolo HTTP.

Un servidor de aplicaciones generalmente gestiona la mayor parte (o la totalidad) de las funciones de lógica de negocio y de acceso a los datos de la aplicación. Los principales beneficios de la tecnología de servidores de aplicación son la centralización y la disminución de la complejidad en el desarrollo de aplicaciones.

El servidor de aplicaciones ejecuta los programas de negocio en lugar del cliente (navegador), del servidor web o sistemas finales. Se sitúa en el medio entre un cliente y los datos empresariales y otras aplicaciones. Físicamente separa la lógica del negocio del cliente y los datos dentro de una arquitectura conocida como multicapa. Los servidores de aplicaciones permiten a las empresas desarrollar y desplegar aplicaciones rápida y fácilmente e incrementan la cantidad de sus usuarios sin reprogramación.

Los servidores de aplicaciones son parte de una arquitectura multicapa. Esta es una arquitectura donde hay una separación física entre el cliente que solicita la información, los programas que la procesan y los datos sobre los que operan. La arquitectura multicapa evolucionó desde los mainframe donde el cliente, los datos, y el proceso estaban centralizados en un único lugar. En los 80's siguió la arquitectura cliente/servidor donde el proceso estaba dividido entre el cliente (una PC) y un servidor (un mainframe) y las solicitudes eran manejadas en consultas a un sistema de bases de datos relacionales. La presentación de la lógica de negocios se aplicaba a la PC después de recibir los datos desde el mainframe.

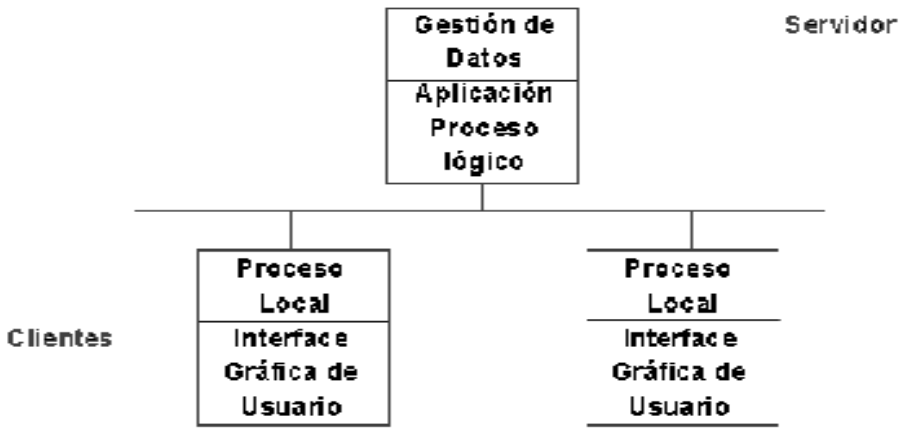


Figura 1.6 Arquitectura de dos capas (cliente / servidor).

Entonces la arquitectura de tres capas separó la lógica de presentación de la lógica de negocio. Esta separación significó que el código de negocio era independiente de cómo y donde se presentaba. La capa de la lógica de negocio, ahora en la capa central, no necesita saber que tipo de cliente muestra los datos. Las tres capas eran más portables, trabajaban en diferentes tipos de plataformas y permitían el balance de las solicitudes del cliente entre varios servidores. La seguridad era fácil de implementar ya que el software de la aplicación estaba ahora fuera del cliente y el costo se redujo considerablemente. Pero proporcionar las funciones subrayadas en la capa central como un proceso de transacciones, la seguridad y el acceso a las capas de datos era todavía complejo. La salida de las herramientas de desarrollo y un entorno de tiempo de ejecución para resolver este problema vienen juntas en el Servidor de Aplicaciones.

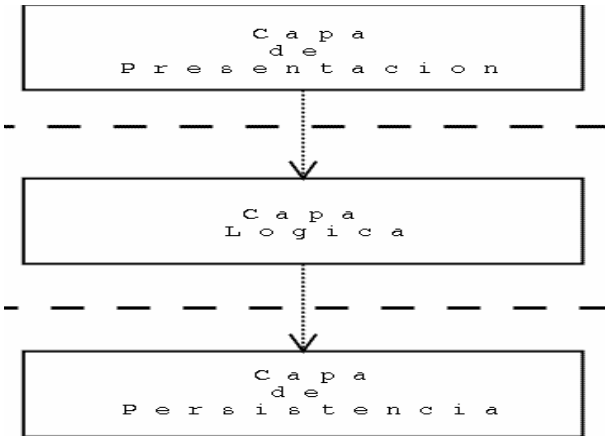


Figura 1.7 Arquitectura de tres capas.

1.5. Almacenamiento de datos

Un archivo es un elemento de información conformado por un conjunto de registros. Estos registros a su vez están compuestos por una serie de caracteres o bytes. Los archivos alojados en dispositivos de almacenamiento conocidos como memoria secundaria, pueden almacenarse de dos formas diferentes: archivos convencionales o bases de datos.

Los archivos convencionales, pueden organizarse como archivos secuenciales o archivos directos. Sin embargo, el almacenamiento de información a través de archivos convencionales presenta una serie de limitaciones que restringen de manera importante la versatilidad de los programas de aplicación que se desarrollan.

1.5.1. Archivos convencionales

El uso de sistemas de información por parte de las organizaciones requiere el almacenamiento de grandes cantidades de información, ya sea para el uso mismo del sistema, para generar resultados o para compartir dicha información con otros sistemas.

Las formas en las cuales pueden organizarse son archivos secuenciales o archivos directos. En los archivos secuenciales los registros están almacenados en una secuencia que depende de algún criterio definido.

Los archivos directos permiten acceder directamente un registro de información sin tener que buscar uno a uno por todos los registros del archivo, utilizando una llave de acceso dentro del archivo.

1.5.2. Base de Datos

Se define como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información. Las bases de datos proporcionan la infraestructura requerida para los sistemas, ya que estos sistemas, explotan la información contenida en las bases de datos.

La base de datos esta compuesta de cuatro partes:

- Datos. Los datos que contiene o va a contener.
- Hardware: Son los dispositivos de almacenamiento.
- Software: Es el conjunto de programas que se conoce como Sistema Manejador de Bases de Datos (DBMS).
- Usuarios: Hay dos tipos de usuarios
    - Usuario final: Accede a la base por medio de un lenguaje o interfaz.
    - Administrador: Se encarga del control de la base de datos.

1.5.3. Ventajas en el uso de Bases de Datos

- Globalización de la información: Permite a los diferentes usuarios considerar la información como un recurso corporativo que carece de dueños específicos.
- Eliminación de información redundante.
- Eliminación de información inconsistente.
- Permite compartir información. Varios sistemas o usuarios pueden utilizar una misma entidad.
- Permite mantener la integridad en la información. Solo se almacena la información correcta.
- Independencia de datos. La independencia de datos implica la separación entre programas y datos; es decir, se pueden hacer cambios a la información que contiene la base de datos o tener acceso a la base de datos de diferente manera, sin hace cambios en las aplicaciones o en los programas.

1.5.4. Modelo de datos

En el proceso de abstracción que conduce a la creación de una base de datos desempeña una función prioritaria el modelo de datos. El modelo de datos, como abstracción del universo de discurso, es el enfoque utilizado para la representación de las entidades y sus características dentro de la base de datos, y puede ser dividido en dos grandes tipos:

1. Modelos lógicos basados en objetos: está dividido en dos y son el modelo entidad-relación y el orientado a objetos:

- El modelo entidad-relación (E-R) se basa en una percepción del mundo compuesta por objetos, llamados entidades, y relaciones entre ellos. Las entidades se diferencian unas de otras a través de atributos.
- El orientado a objetos también se basa en objetos, los cuales contienen valores y métodos, entendidos como órdenes que actúan sobre los valores, en niveles de anidamiento. Los objetos se agrupan en clases, relacionándose mediante el envío de mensajes.

2. Modelos lógicos basados en registros: está divido en tres y son:

- El modelo relacional representa los datos y sus relaciones mediante tablas bidimensionales, que contienen datos tomados de los dominios correspondientes.
- El modelo jerárquico representa los registros se organizan como colecciones de árboles.
- El modelo de red está formado por colecciones de registros, relacionados mediante apuntadores o ligas.

1.6. Lenguajes de Programación

Un lenguaje de programación permite a un programador especificar de manera precisa los datos que una computadora debe operar, cómo deben ser estos almacenados y transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. Todo esto, a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o natural.

Un programa escrito en un lenguaje de programación necesita pasar por un proceso de compilación, es decir, ser traducido al lenguaje máquina, o ser interpretado para que pueda ser ejecutado por la computadora.

Los lenguajes de programación pueden clasificarse empleando distintos métodos y puntos de vista. Esta clasificación se basa en el paradigma que utilizan.

1.6.1. Paradigma de la programación

Un paradigma es una manera de realizar una acción o resolver un problema, en el caso de la programación provee (y determina) la visión y métodos de un programador en la construcción de un programa o subprograma. Diferentes paradigmas resultan en diferentes estilos de programación y en diferentes formas de pensar la solución de problemas.

1.6.2. Clasificación de los paradigmas de programación

Paradigma Estructurado: la programación se divide en bloques (procedimientos y funciones) que pueden o no comunicarse entre sí. Además la programación se controla con secuencia, selección e iteración.

Paradigma Orientado a Objetos: está basado en la idea de encapsular estado y operaciones en objetos. En general, la programación se resuelve comunicando dichos objetos a través de mensajes, además posee herencia y subtipos entre objetos. Su principal ventaja es la reutilización de códigos y su facilidad para pensar soluciones a determinados problemas.

Paradigma Lógico: se basa en la definición de reglas lógicas para luego, a través de un motor de inferencias lógicas, responder preguntas planteadas al sistema y así resolver los problemas.

Paradigma Funcional: es un paradigma de programación declarativa basado en la utilización de funciones matemáticas. El objetivo es conseguir lenguajes expresivos y matemáticamente elegantes, en los que no sea necesario bajar al nivel de la máquina para describir el proceso llevado a cabo por el programa.

1.7. Patrones

En el diseño orientado a objetos existen cierto tipo de problemas que se presentan con frecuencia. Para analizar y compartir el conocimiento sobre ellos, se ha desarrollado un tipo formal de documentación llamado "patrón".

Un patrón es un conjunto de información que aporta la solución a un problema que se presenta en un contexto determinado. Para elaborarlo se aíslan sus aspectos esenciales y se añaden cuantos comentarios y ejemplos sean necesarios. En particular debemos identificar:

- El contexto en el que es posible aplicar el patrón.
- Las fuerzas (objetivos y restricciones) que intervienen en ese contexto.
- El diseño a aplicar para equilibrar objetivos y restricciones.

Las características de un buen patrón:

- Resuelve un problema cuya solución no es obvia.
- Ha sido revisado por una comunidad de desarrolladores.
- Ha sido experimentado en la práctica.
- Muestra como equilibrar restricciones y objetivos.

Los patrones son importantes por varios motivos:

- Encapsulan conocimiento detallado sobre un tipo de problema y sus soluciones.
- Proporcionan un vocabulario común.
- Estimula la reutilización del software (un patrón no es tal si no es reutilizable).
- Al aplicar soluciones probadas evitamos los riesgos y ahorramos el esfuerzo de reinventar nuestras soluciones.

1.7.1. Historia de los patrones

El concepto de "patrón de diseño" que tenemos en Ingeniería del Software se ha tomado prestado de la arquitectura. En 1977 se publica el libro "A Pattern Language: Towns/Building/Construction", de Christopher Alexander.

Alexander comenta que "Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, para describir después el núcleo de la solución a ese problema, de tal manera que esa solución pueda ser usada más de un millón de veces sin hacerlo siquiera dos veces de la misma forma". El patrón es un esquema de solución que se aplica a un tipo de problema, esta aplicación del patrón no es mecánica, sino que requiere de adaptación y matices. Por ello, dice Alexander que los numerosos usos de un patrón no se repiten dos veces de la misma forma.

La idea de patrones de diseño estaba "en el aire", la prueba es que numerosos diseñadores se dirigieron a aplicar las ideas de Alexander a su contexto. El catálogo más famoso de patrones se encuentra en “Design Patterns: Elements of Reusable Object-Oriented Software”, de Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides, 1995, Addison-Wesley, también conocido como el libro GOF (Gang-Of-Four).

1.7.2. Clasificación

Existen varios tipos de patrones, dependiendo del nivel de abstracción, del contexto particular en el cual aplican o de la etapa en el proceso de desarrollo. Algunos de estos tipos, de los cuales existen publicaciones al respecto son:

Patrones de creación

Los patrones de creación abstraen la forma en la que se crean los objetos, permitiendo tratar las clases a crear de forma genérica dejando para más tarde la decisión de qué clases crear o cómo crearlas.

Según donde se tome dicha decisión podemos clasificar a los patrones de creación en patrones de creación de clase (la decisión se toma en los constructores de las clases y usan la herencia para determinar la creación de las instancias) y patrones de creación de objeto (se modifica la clase desde el objeto).

Algunos patrones de creación son los siguientes: Abstract Factory (Fábrica abstracta), Builder (Constructor), Factory Method (Método de fabricación), Prototype (Prototipo) y Singleton (Único).

Patrones estructurales

Tratan de conseguir que cambios en los requisitos de la aplicación no ocasionen cambios en las relaciones entre los objetos. Lo fundamental son las relaciones de uso entre los objetos y éstas están determinadas por las interfaces que soportan los objetos. Estudian como se relacionan los objetos en tiempo de ejecución. Sirven para diseñar las interconexiones entre los objetos.

Algunos patrones estructurales son: Adapter (Adaptador), Bridge (Puente), Composite (Compuesto), Facade (Fachada) y Proxy (Representante).

Patrones de comportamiento

Los patrones de comportamiento estudian las relaciones con el flujo de control de un sistema. Ciertas formas de organizar el control en un sistema pueden derivar en grandes beneficios para la eficiencia y el mantenimiento del sistema.

Algunos patrones de comportamiento son: Chain of responsability (Cadena de responsabilidades), Command (Comando), Interpreter (Intérprete), Observer (Observador), State (Estado), Strategy (Estrategia) y Visitor (Visitante)

Patrones de sistema

Los patrones de sistema constituyen el más diverso de los cuatro tipos de patrones. Llevan su aplicación al nivel más abstracto de su arquitectura. Los patrones de sistema pueden aplicarse a los procesos principales de una aplicación, o incluso entre aplicaciones.

Algunos patrones de sistema son: Model-View-Controler (Modelo-Vista-Controlador), Session (Sesión), Callback (Retrollamada), Seccessive Update (Actualización sucesiva) y Transaction (Transacción).

1.7.3. Modelo – Vista – Controlador (MVC)

El MVC se encuentra integrado dentro de los patrones de sistema y su principal objetivo es aislar tanto los datos de la aplicación, como el estado (modelo) de la misma y el mecanismo utilizado para representar (vista) dicho estado, así como para modularizar esta vista y modelar la transición entre estados del modelo (controlador). Las aplicaciones MVC se dividen en tres grandes áreas funcionales:

- Vista: La presentación de los datos.
- Controlador: El que atenderá las peticiones y componentes para toma de decisiones de la aplicación. Controla la transición entre el procesamiento de los datos y su visualización.
- Modelo: Es la lógica del negocio o servicio y los datos asociados con la aplicación, representa sus datos y comportamientos.

El propósito del MVC es aislar los cambios. Es una arquitectura preparada para los cambios, que desacopla datos y lógica de negocio de la lógica de presentación, permitiendo la actualización y desarrollo independiente de cada uno de los componentes citados.

A continuación se muestra un esquema de este modelo:

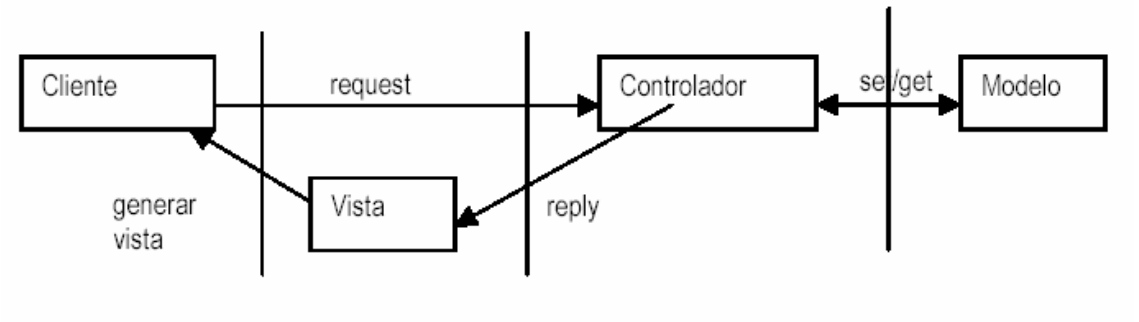


Figura 1.8 Esquema del patrón MVC.

2. Análisis del Sistema

2.1. Definición del Sistema de Información de Usuarios

Es un espacio virtual que se consulta vía Web y que realiza las siguientes labores:

Proporciona diversa información sobre características administrativas del Laboratorio de Computación, como lo son:

- Equipo con el que cuenta el Laboratorio de Computación.
- Software del que se dispone en el Laboratorio de Computación.
- Cursos impartidos en el Laboratorio de Computación.
- Clases impartidas en el Laboratorio de Computación.
- Ver información de los usuarios que utilizan el Laboratorio de Computación:
    - Datos específicos de un alumno.
    - Usuarios que se encuentran haciendo uso del Laboratorio de Computación en tiempo real.
    - Graficas de la concurrencia de los usuarios al Laboratorio de Computación.

Para ver la especificación de toda la información referente a todos los usuarios, cuenta con un repositorio de datos. Este repositorio de datos es manejado por otro sistema denominado Caifán, que se encarga de manejar los datos correspondientes, por lo tanto el SIU, sólo hará el manejo de estos datos para hacer representaciones gráficas y estadísticas de los usuarios.

También maneja su propio repositorio de datos que utilizará para mantener la información requerida de los demás módulos anexos al sistema.

2.2. Lenguaje Unificado de Modelado (UML)

UML fue desarrollado cerca de 1990, por tres líderes en modelado de objetos a nivel mundial: Grady Booch, James Rumbaugh e Ivar Jacobson.
UML es un lenguaje gráfico para modelar sistemas de software. No es un lenguaje de programación, es un paquete de diagramas que se usan para especificar, visualizar y documentar software. Los ingenieros de software usan diagramas de UML para construir y explicar el diseño de su software, es similar a cómo los arquitectos hacen planos para realizar construcciones.

2.2.1. Diagramas de Caso de Uso

Los Casos de Uso no son parte del diseño (cómo), sino parte del análisis (qué). De forma que al ser parte del análisis nos ayudan a describir qué es lo que el sistema debe hacer. Es decir, describen un uso del sistema y cómo este interactúa con el usuario.

Por lo tanto, un diagrama de Casos de Uso, son todos los casos que la aplicación debe hacer y los posibles actores que intervienen en el sistema.

2.2.1.1. Actores

Consultor. Puede hacer consultas, dependiendo el perfil que se le asigne.

Administrador. Puede hacer consultas, dar altas nuevas, administrar el repositorio de datos y administrar módulos.

2.2.1.2. Diagrama del consultor

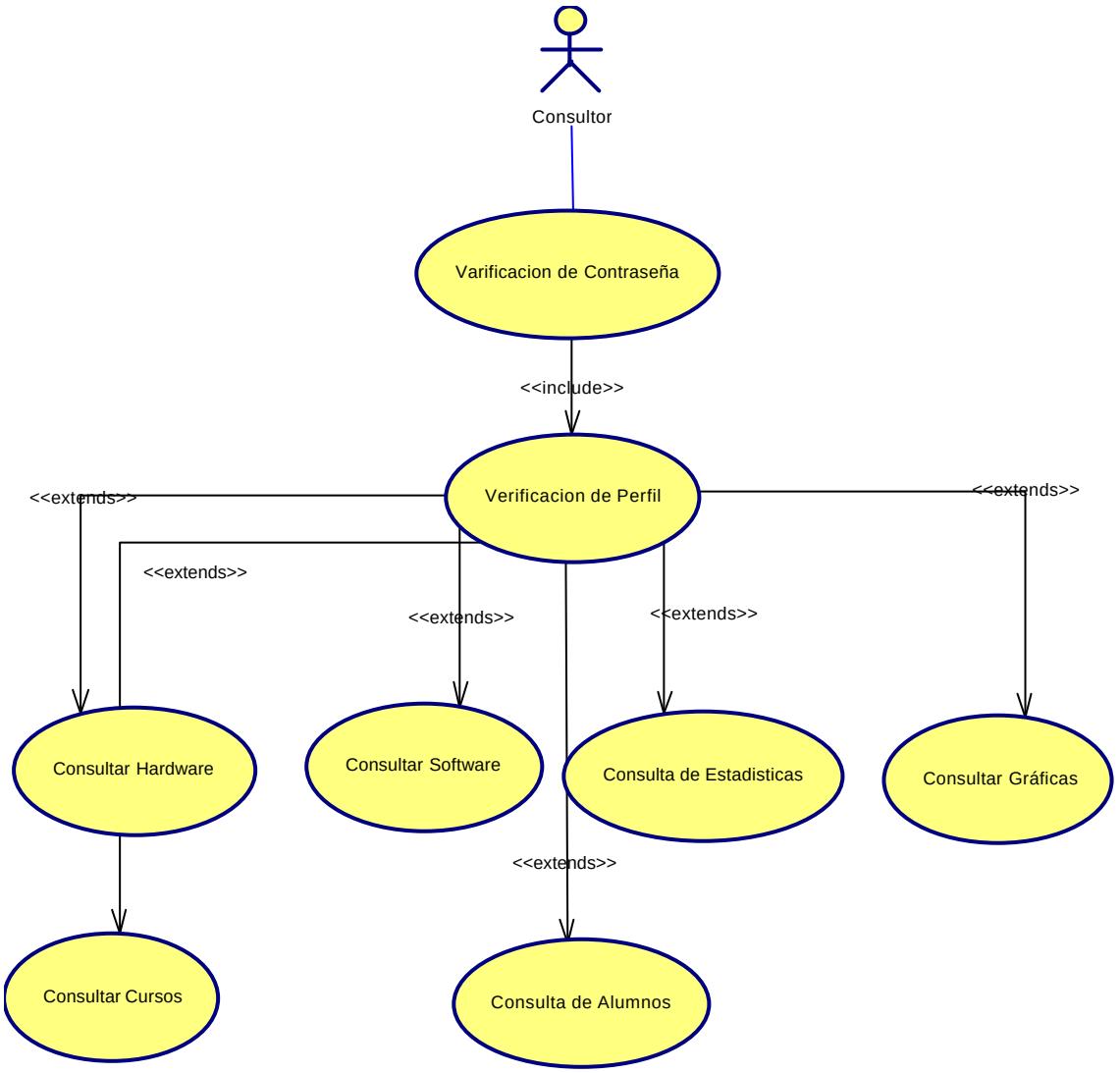


Figura 2.1 Casos de Uso del actor denominado Consultor.

2.2.1.3. Diagrama del Administrador

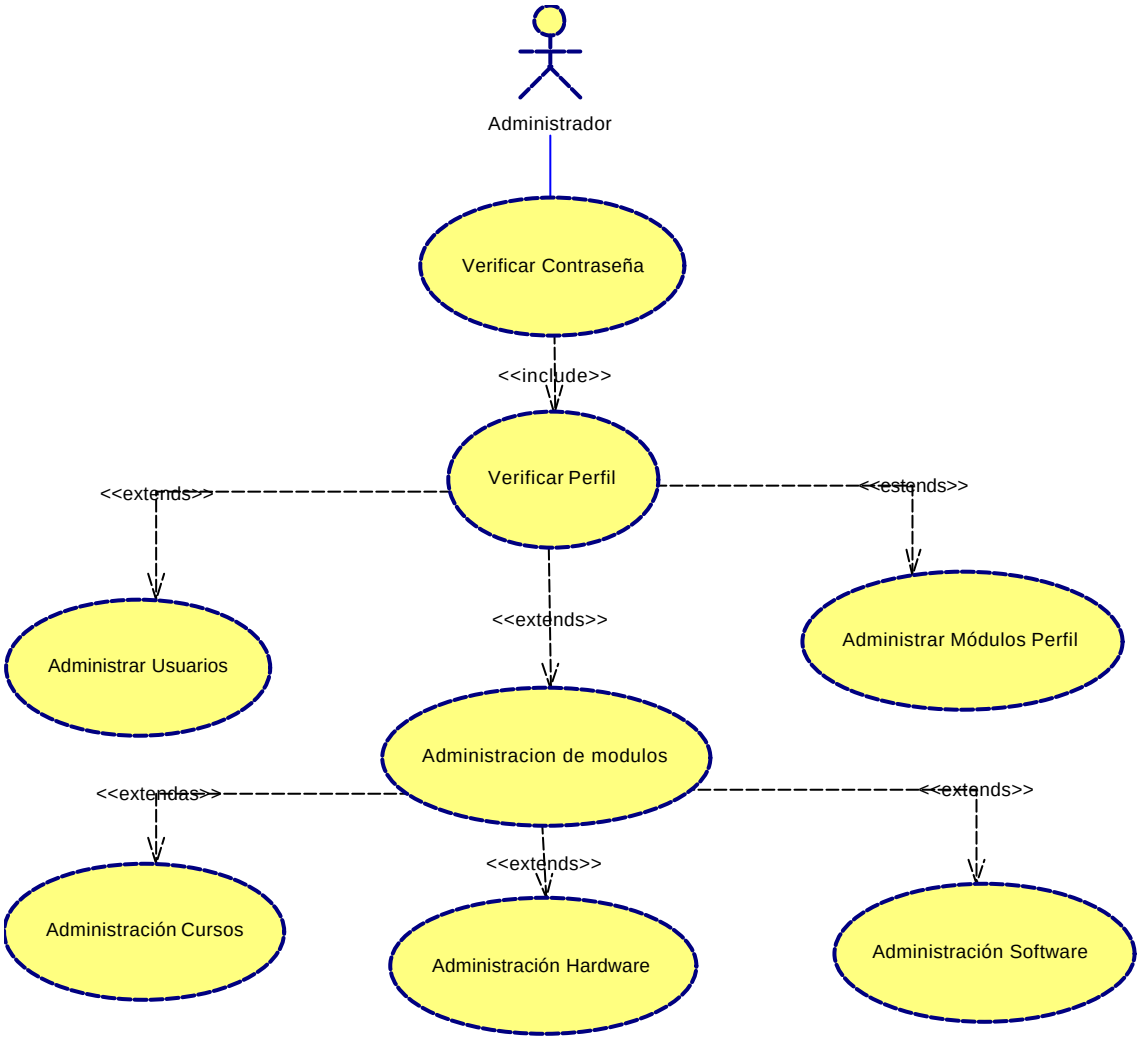


Figura 2.2 Casos de Uso del actor denominado Administrador.

2.2.1.4. Descripción de los Casos de Uso

Nombre: Verificación de Contraseña.

Descripción: Validación del consultor o administrador para entrar a la sección de consultas del SIU.

Flujo Básico: El consultor o administrador ingresa al formulario de validación, donde el sistema le pide que proporcione su login y contraseña para poder ingresar al sistema.

El sistema presenta dos opciones de idioma: Español e Inglés.

El usuario escoge el idioma de su preferencia e introduce los datos solicitados.

El caso de uso termina cuando el consultor o administrador es validado para usar el sistema.

Flujo Alterno: El login o contraseña del usuario no es correcto; el sistema manda un mensaje de error.

Nombre: Verificación de Perfil.

Descripción: Validación de los perfiles del consultor o administrador.

Flujo Básico: El consultor o administrador ha validado su login y la contraseña y el sistema le permite ingresar.

Una vez dentro del sistema, se hace una validación sobre el perfil con que cuenta al consultor o administrador, esto se refiere a que se checa a qué módulos tiene acceso.

El caso de uso termina cuando el consultor o administrador esta dentro del sistema y puede ver los módulos a los que tiene acceso.

Flujo Alterno: El consultor no tiene ningún módulo asignado; el sistema le manda una notificación del error.

El consultor no tiene perfil asignado para ingresar en el módulo de administración; el sistema manda una notificación del error.

Nombre: Consultar de Alumnos.

Descripción: El consultor ve la referencia del historial de un usuario.

Flujo Básico: El consultor ha sido validado y tiene éste perfil asignado.

El sistema le muestra un formulario con la opción de Alumnos y la escoge.

El caso de uso termina cuando el cliente ve el historial del usuario seleccionado.

Flujo Alterno: El login del usuario seleccionado no existe.

El historial del usuario seleccionado no se encuentra disponible en el repositorio de datos.

Nombre: Consultar de Estadísticas.

Descripción: El consultor tiene diversas opciones para ver estadísticas de usuarios referentes a los usuarios que se encuentran inscritos dentro del Laboratorio de Computación.

Flujo Básico: El consultor ha sido validado y tiene éste perfil asignado.

El sistema le muestra un formulario con la opción de Estadísticas y la escoge.

El consultor ve un formulario que cuenta con las opciones de:

- Semestre.
- Carreras.
- Fecha inicial.
- Fecha final.
- Hora inicial.
- Hora final.

El consultor hace una selección de una o más opciones.

El sistema le muestra los datos referentes a las opciones seleccionadas.

El usuario puede crear un reporte en Excel de las opciones que le arrojó la consulta.

El caso de uso termina cuando el consultor ve los datos de la consulta.

Flujo Alterno: El sistema no muestra ningún dato, debido a que no hay con las opciones seleccionadas.

Nombre: Consultar Hardware.

Descripción: El consultor ve el hardware con el que cuentan los usuarios dentro del Laboratorio de Computación.

Flujo Básico: El consultor ha sido validado y tiene éste perfil asignado.

El sistema le muestra un formulario con la opción de Hardware y la escoge.

El caso de uso termina cuando el consultor ve el hardware disponible dentro del Laboratorio de Computación.

Flujo Alterno: No se encuentran datos en el repositorio de datos.

Nombre: Consultar Software.

Descripción: El consultor ve el software con el que cuentan los usuarios dentro del Laboratorio de Computación.

.*Flujo Básico*: El consultor ha sido validado y tiene éste perfil asignado.

El sistema le muestra un formulario con la opción de Software y la escoge.

El caso de uso termina cuando el consultor ve el software disponible dentro del Laboratorio de Computación.

Flujo Alterno: No se encuentran datos en el repositorio de datos.

Nombre: Consultar Gráficas.

Descripción: El consultor ve las gráficas con las que cuenta el SIU.

Flujo Básico: El consultor ha sido validado y tiene éste perfil asignado.

El sistema le muestra un formulario con la opción de Gráficas y la escoge.

El sistema muestra un formulario donde puede escoger las siguientes opciones:

- Carreras: Usuarios inscritos por carrera dentro del Laboratorio de Computación.
- Semestres: Usuarios inscritos por semestre dentro del Laboratorio de Computación.

- Gráficas por semestre: Número de accesos de usuarios a lo largo de un semestre.
- Gráficas mensuales en el semestre actual: Número de acceso de usuarios a lo largo de un semestre, distribuido por los meses que contienen al semestre en curso.

El caso de uso termina cuando el consultor ve alguna(s) de las gráficas mencionadas.

Flujo Alterno: No hay datos disponibles en el repositorio de datos.

Nombre: Consultar Cursos.

Descripción: El consultor ve los cursos y clases que han impartido dentro del Laboratorio de Computación.

Flujo Básico: El consultor ha sido validado y tiene éste perfil asignado.

El sistema le muestra un formulario con la opción de Cursos y la escoge.

El sistema le muestra al consultor un formulario que contiene tres categorías ordenadas por semestre:

- Clase semestral.
- Clase ocasional.
- Curso.

El caso de uso termina cuando el consultor ve cualquiera de las opciones antes señaladas.

Flujo Alterno: No hay datos disponibles para la opción indicada.

Nombre: Administrar Usuarios (Agregar usuarios).

Descripción: El administrador puede dar de agregar un usuario que hará uso del sistema.

Flujo Básico: El administrador ha sido validado y tiene perfil asignado para ingresar al módulo de administración.

El sistema le muestra un formulario con la opción de Usuarios y la escoge.

El sistema le muestra al administrador un formulario que contiene dos opciones: Agregar y Consultar.

El administrador escoge la opción Agregar.

El sistema muestra un formulario que tiene los campos de:

- Nombre: Nombre del usuario.
- Apellido paterno: Apellido paterno del usuario.
- Apellido materno: Apellido materno del usuario.
- Estatus: Estado de activado o desactivado del usuario al momento de ser dado de alta.

El administrador llena cada uno de los campos indicados y envía los datos.

El caso de uno termina cuando el sistema muestra un formulario con el nombre completo, login, contraseña y estatus del usuario dado de alta.

Flujo Alterno: No se insertó el usuario en el repositorio de datos.

Nombre: Administrar Usuarios (Consultar usuarios).

Descripción: El administrador puede consultar un usuario para cambiar características de los usuarios dados de alta en el sistema.

Flujo Básico: El administrador ha sido validado y tiene perfil asignado para ingresar al módulo de administración.

El sistema le muestra un formulario con la opción de Usuarios y la escoge.

El sistema le muestra al administrador un formulario que contiene dos opciones: Agregar y Consultar.

El administrador escoge la opción Consultar.

El sistema le muestra un formulario con las siguientes opciones:

- Datos Personales: El sistema muestra un formulario con los datos del usuario consultado.
- Cambio de contraseña: El sistema muestra un formulario donde se puede cambiar la contraseña del usuario.
- Perfiles: El sistema muestra un formulario donde se ven los módulos que se encuentran disponibles para que el usuario tenga acceso.
- Habilitar usuario: Se habilita el usuario para que pueda entrar al sistema.

- Deshabilitar usuario: Se deshabilita el usuario seleccionado, con esto el usuario no podrá ingresar al sistema.

Para ingresar a los formularios anteriores, se selecciona cualquiera de las opciones anteriores y el sistema muestra un formulario que contiene un campo para ingresar el login del usuario que se quiere modificar.

El administrador ingresa el login de un usuario.

El caso de uso termina cuando el administrador modifica al usuario seleccionado.

Flujo Alterno: No se encontró en el sistema el usuario seleccionado.

Nombre: Administrar Módulos Perfil.

Descripción: El administrador puede administrar características de los módulos que componen al SIU.

Flujo Básico: El administrador ha sido validado y tiene perfil asignado para ingresar al módulo de administración.

El sistema le muestra un formulario con la opción de Módulos y la escoge.

El sistema le muestra al administrador un formulario que contiene dos opciones: Agregar, Borrar, Activar/Desactivar y Administrar.

- Agregar: El sistema muestra un formulario que contiene los campos de:
  - Nombre: Es el nombre que se le asignará al módulo.
  - URL SIU: Es el url que lleva a la primera página correspondiente para ese módulo en la parte pública.
  - URL Admón.: Es el url que lleva a la parte de administración para dicho módulo anexado, este se vera reflejado en el submódulo denominado "Administrar", el cuál se detalla más adelante.
  - Id etiqueta: Este indica el id con el que se encuentra guardada en la base de datos la etiqueta que se va a mostrar tanto en la parte pública como en la de administración para éste módulo.
  - Estatus: Indica si el módulo se va a estar activado o desactivado.
- Borrar: El sistema muestra un formulario de los módulos que se pueden borrar. Las opciones seleccionadas serán removidas del repositorio de datos.
- Activar/Desactivar: El sistema muestra un formulario de los módulos que se pueden activar y/o desactivar, tanto

predeterminados con los anexados al sistema. Los módulos seleccionados quedarán activos en el sistema.

- Administrar: El sistema muestra los enlaces a los formularios de administración para cada uno de los módulos que cuenten con dicha sección, como son el caso de cursos, software y hardware.

Para ingresar a los formularios anteriores, se selecciona cualquiera de las opciones anteriores y el sistema muestra un formulario correspondiente.

El caso de uso termina cuando el administrador modifica un módulo con alguna de las opciones antes mencionadas.

Flujo Alterno: No se insertó o actualizaron los datos en el repositorio de datos.

Nombre: Administrar Módulos.

Descripción: El administrador puede administrar características de los módulos que componen al SIU.

Flujo Básico: El administrador ha sido validado y tiene perfil asignado para ingresar al módulo de administración.

El sistema le muestra un formulario con la opción de Módulos y la escoge.

El sistema le muestra al administrador un formulario que contiene dos opciones: Agregar, Borrar, Activar/Desactivar y Administrar.

El caso de uso termina cuando el administrador escoge alguno de los enlaces para entrar al área de administración de la opción seleccionada.

Flujo Alterno: No se encontró el formulario para la opción indicada.

Nombre: Administración Cursos.

Descripción: El administrador puede administrar los cursos y clases que se imparten semestralmente en el Laboratorio de Computación.

Flujo Básico: El administrador ha sido validado y tiene perfil asignado para ingresar al módulo de administración.

El sistema le muestra un formulario con la opción de Módulos y la escoge.

El sistema le muestra al administrador un formulario que contiene dos opciones: Agregar, Borrar, Activar/Desactivar y Administrar.

El administrador escoge la opción Administrar y el sistema muestra un formulario con los enlaces existentes para administrar módulos que son: Cursos, Software y Hardware.

El administrador escoge la opción de Cursos y el sistema la muestra un formulario con cuatro opciones:

- Nuevo Semestre: El sistema muestra un formulario con los campos de:
  - Nombre: Nombre del semestre actual.
  - Fecha de inicio: Fecha de inicio del semestre.
  - Fecha de fin: Fecha de fin del semestre.

  Al dar de alta un nuevo semestre, éste es el que estará activado como semestre actual.

- Alta materia semestral: El sistema muestra un formulario con los siguientes campos:
  - Materia: Indica la materia que se impartirá.
  - Profesor: Indica el profesor que impartirá la materia.
  - Horario inicial: Es la hora en que inicia la clase.
  - Horario final: Es la hora en que termina la clase.
  - Días: Se habilitan los días en que se da la clase.

- Alta de materia ocasional: El sistema muestra un formulario con los siguientes campos:
  - Materia: Indica la materia que se impartirá.
  - Profesor: Indica el profesor que impartirá la materia.
  - Fecha: Es el día en que se dará la clase.
  - Horario inicial: Es la hora en que inicia la materia.
  - Horario final: Es la hora en que termina la materia.

- Alta de cursos: El sistema muestra un formulario con los siguientes campos:
  - Materia: Indica la materia que se impartirá.
  - Profesor: Indica el profesor que impartirá la materia.

- Fecha Inicial: Es el día en que se iniciará el curso.
- Fecha Final: Es el día en que se terminará el curso
- Horario inicial: Es la hora en que inicia el curso.
- Horario final: Es la hora en que termina el curso.

Para ingresar a los formularios anteriores, se selecciona cualquiera de las opciones anteriores y el sistema muestra un formulario correspondiente.

El caso de uso termina cuando el administrador hace uso de algún formulario de las opciones antes mencionadas.

Flujo Alterno: No se insertó o actualizaron los datos en el repositorio de datos.

Nombre: Administración Software.

Descripción: El administrador puede administrar el Software que se encuentra disponible dentro del Laboratorio de Computación y que está disponible para el uso de los usuarios.

Flujo Básico: El administrador ha sido validado y tiene perfil asignado para ingresar al módulo de administración.

El sistema le muestra un formulario con la opción de Módulos y la escoge.

El sistema le muestra al administrador un formulario que contiene dos opciones: Agregar, Borrar, Activar/Desactivar y Administrar.

El administrador escoge la opción Administrar y el sistema muestra un formulario con los enlaces existentes para administrar módulos que son: Cursos, Software y Hardware.

El administrador escoge la opción de Software y el sistema la muestra un formulario con cuatro opciones:

- Agregar Software: Muestra un formulario con al campo nombre del software, que es el software que se quiere utilizar. El Software insertado se ve reflejado en el formulario de Quitar Software.

- Quitar Software: Muestra un formulario con el Software que se encuentra dado de alta en el repositorio de datos. Lo nombres del Software que sean seleccionados se eliminaran del repositorio de datos.

Para ingresar a los formularios anteriores, se selecciona cualquiera de las opciones anteriores y el sistema muestra un formulario correspondiente.

El caso de uso termina cuando el administrador hace uso de algún formulario de las opciones antes mencionadas.

Flujo Alterno: No se insertó el nombre del Software en el repositorio de datos.

No se pudo eliminar el Software seleccionado.

Nombre: Administración Hardware.

Descripción: El administrador puede administrar el Hardware que se encuentra disponible dentro del Laboratorio de Computación y que está disponible para el uso de los usuarios.

Flujo Básico: El administrador ha sido validado y tiene perfil asignado para ingresar al módulo de administración.

El sistema le muestra un formulario con la opción de Módulos y la escoge.

El sistema le muestra al administrador un formulario que contiene dos opciones: Agregar, Borrar, Activar/Desactivar y Administrar.

El administrador escoge la opción Administrar y el sistema muestra un formulario con los enlaces existentes para administrar módulos que son: Cursos, Software y Hardware.

El administrador escoge la opción de Hardware y el sistema la muestra un formulario con cuatro opciones:

- Agregar Hardware: Muestra un formulario con los siguientes campos:
  - Marca.
  - Procesador.
  - Disco Duro.
  - Memoria RAM.
  - Sistema Operativo.
  - Imagen.

  El Hardware insertado se ve reflejado en el formulario de Quitar Hardware.

- Quitar Hardware: Muestra un formulario con el Hardware que se encuentra dado de alta en el repositorio de datos. Lo nombres del Hardware que sean seleccionados se eliminaran del repositorio de datos.

Para ingresar a los formularios anteriores, se selecciona cualquiera de las opciones anteriores y el sistema muestra un formulario correspondiente.

El caso de uso termina cuando el administrador hace uso de algún formulario de las opciones antes mencionadas.

Flujo Alterno: No se insertó el nombre del Hardware en el repositorio de datos.

No se pudo eliminar el Hardware seleccionado.

2.2.2. Diagrama de Clases

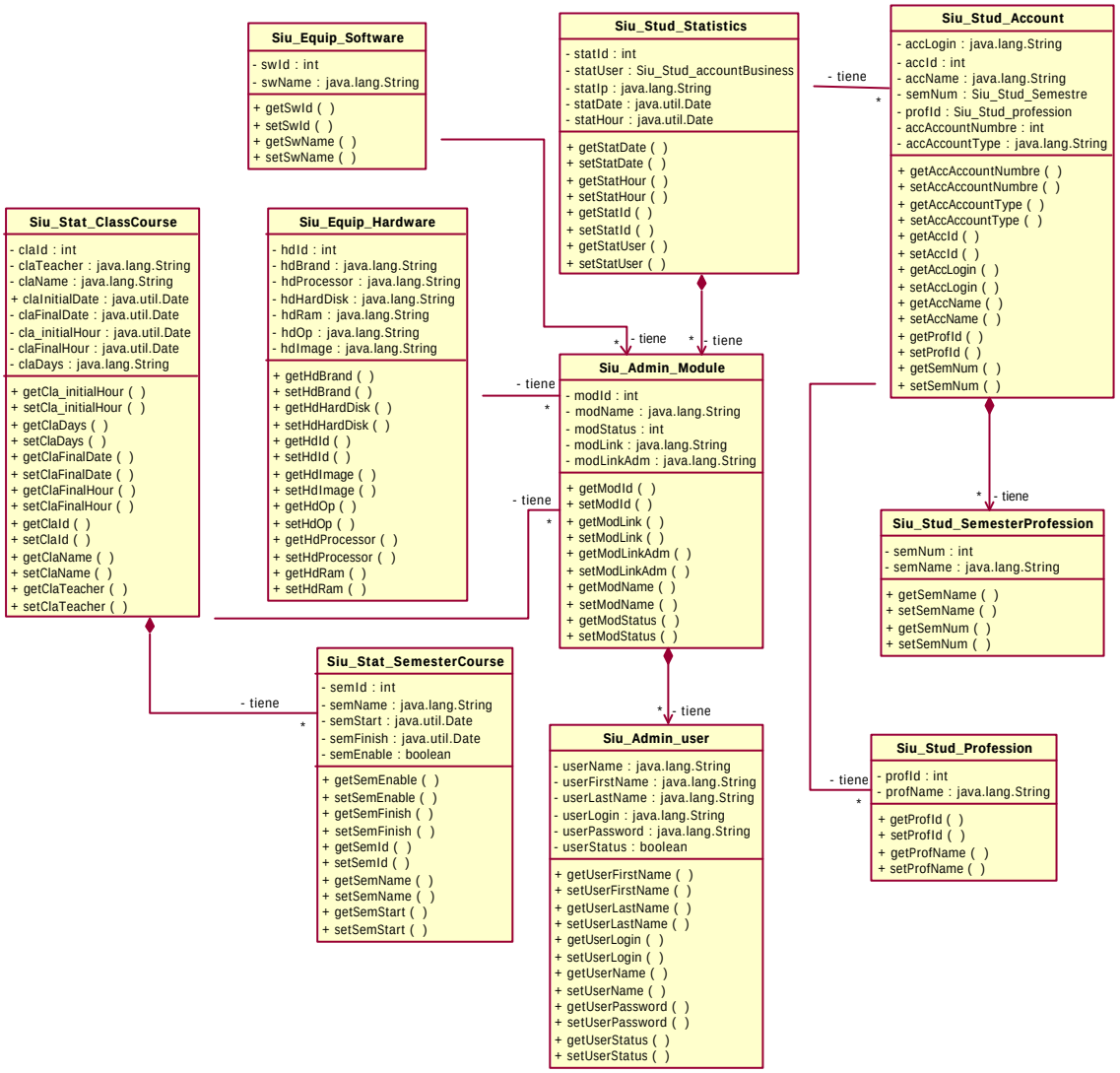


Figura 2.3 Diagrama de Clases

3. Diseño del Sistema

3.1. Especificaciones de Diseño

El Sistema de Información de Usuarios (SIU) está conformado por módulos, el núcleo principal del SIU es "Administration". Este módulo tiene una Zona de Vistas, en las que los usuarios ingresan a ver las diferentes pantallas de cada módulo. De igual modo contiene una Zona de Administración que se encargará de las siguientes labores:

- Administra a los usuarios
  - Altas.
  - Cambios.
  - Asignación de Perfiles.
  - Activación y Desactivación.

- Administración de Módulos
  - Agregar.
  - Borrar.
  - Activación y Desactivación.
  - Administrador de módulos anexos.

Los módulos que tiene de manera predefinida el SIU son:

- Equipment que se divide en:
  - Software
  - Hardware

- Students que se divide en:
  - Students
  - Statistics

- Statistics que se divide en:
  - Courses
  - Graghics

El sistema es capaz de anexar módulos, esto lo hace en la Zona de Administración. El módulo que se anexa debe de tener una vista que se pública que será visible a través de la Zona de Vistas y de ser necesario puede tener un vista de administración, que se publica en la Zona de Administración, en la opción que hace referencia a administrador de módulos.

En el caso de que los módulos tengan submódulos como en los predefinidos, cada submódulo tendrá su correspondiente vista y de ser necesario su vista administrativa y serán agregados de la misma manera que cualquier módulo.

Estructuralmente la manera en que se construye cada módulo está especificado en el "ANEXO A: Especificaciones de Programación".

En el sistema, cada módulo tendrá sus propias tablas, éstas deben estar integradas en la misma base de datos que el módulo Administration, pero siguiendo las especificaciones del "ANEXO B: Especificaciones de Diseño de la Base de Datos".

Automáticamente al agregar un módulo, se crea un nuevo perfil. Los perfiles se asignan a los usuarios para que tengan posibilidad de tener acceso a la Zona de Vistas e incluso a la Zona de Administración.

Cada módulo puede ser habilitado o deshabilitado del sistema, pero sólo los módulos que no sean predeterminados se pueden eliminar del sistema. El módulo principal que es "Administration" no puede ser ni deshabilitado ni eliminado.

Este sistema soporta varios idiomas, simplemente se agregarán las etiquetas correspondientes a dicho idioma; En la base de datos existe una tabla que contiene las claves de los idiomas, con esta clave se relacionarán las etiquetas correspondientes a cada idioma. Por el momento sólo se tiene contemplado que el sistema tenga etiquetas en idioma Español e Inglés.

Para agregar un idioma se deben dar de altas las etiquetas correspondientes a dicho idioma, estas siguen las siguientes características:

- Las etiquetas tienen un código formado por 5 dígitos y se tiene ese mismo código por cada idioma. Por ejemplo:

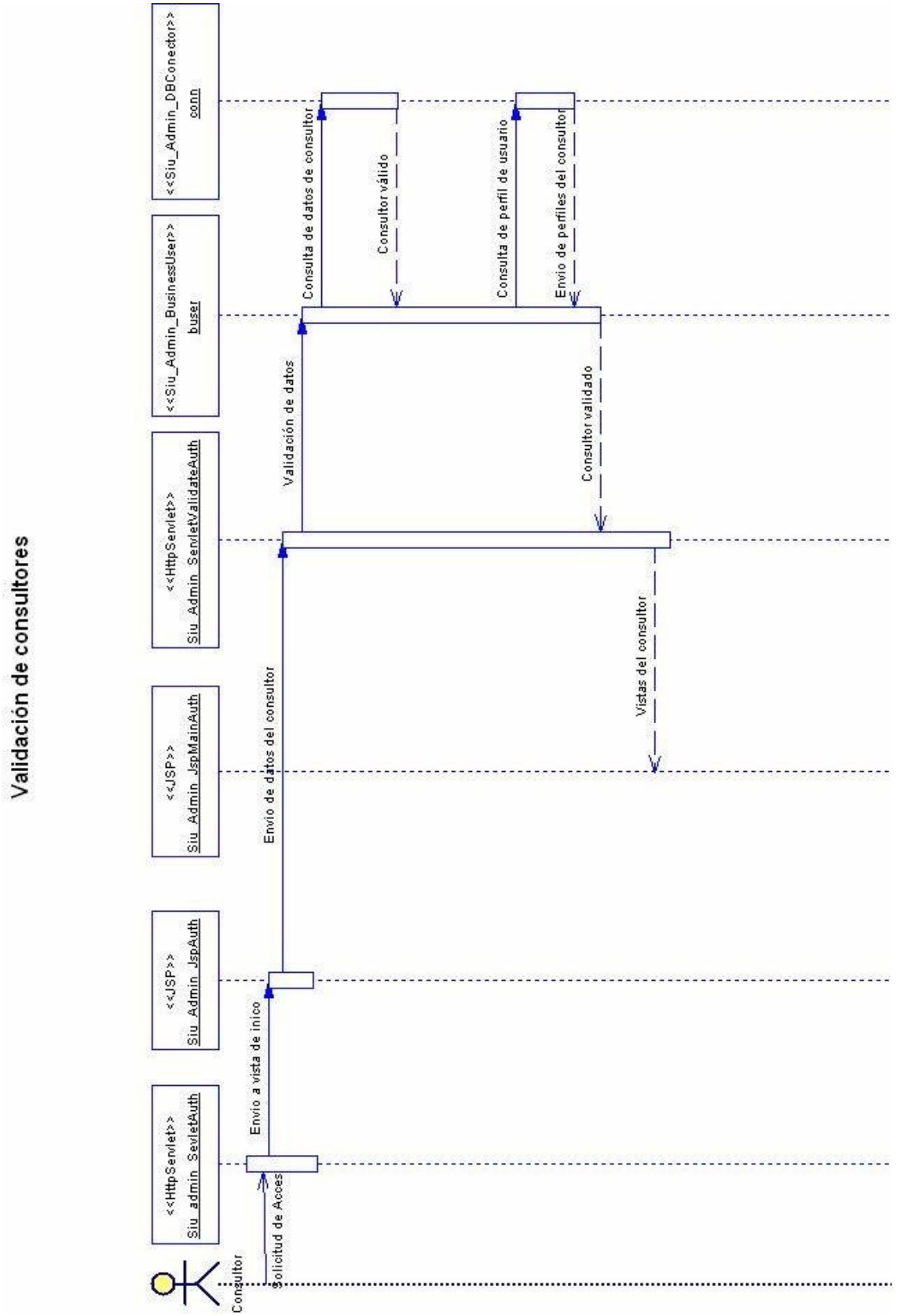
    - 00001 corresponde a "Usuario" en idioma Español
    - 00001 corresponde a "User" en idioma Inglés
    - 00001 corresponde a "Usager" en idioma Francés

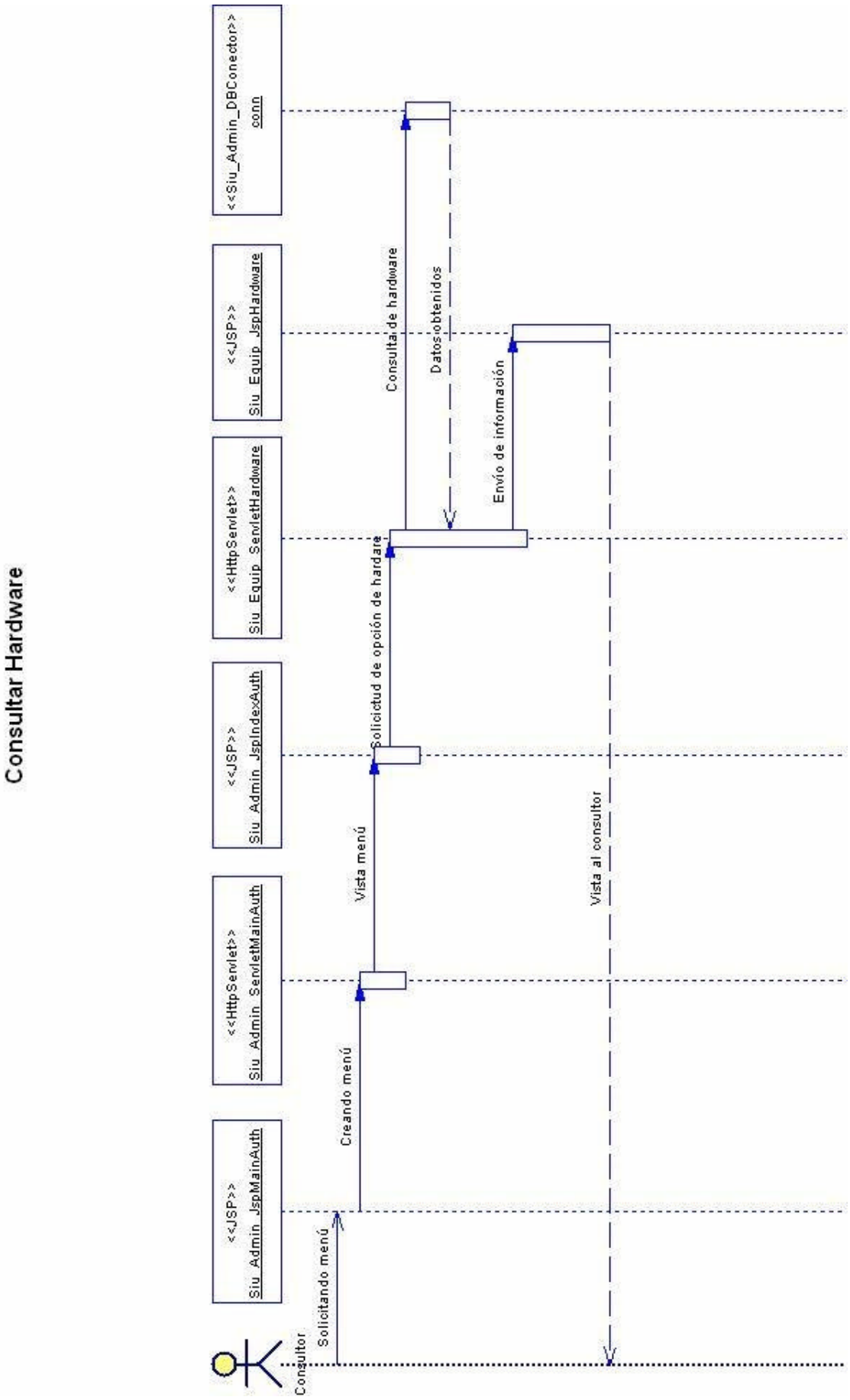
    Cada una de las etiquetas esta relacionada con el idioma al que pertenece. Tomando los ejemplos de las etiquetas, las claves de los idiomas con que se relacionan las etiquetas serían:
    - 1 – Español.
    - 2 – Inglés.
    - 3 – Francés.

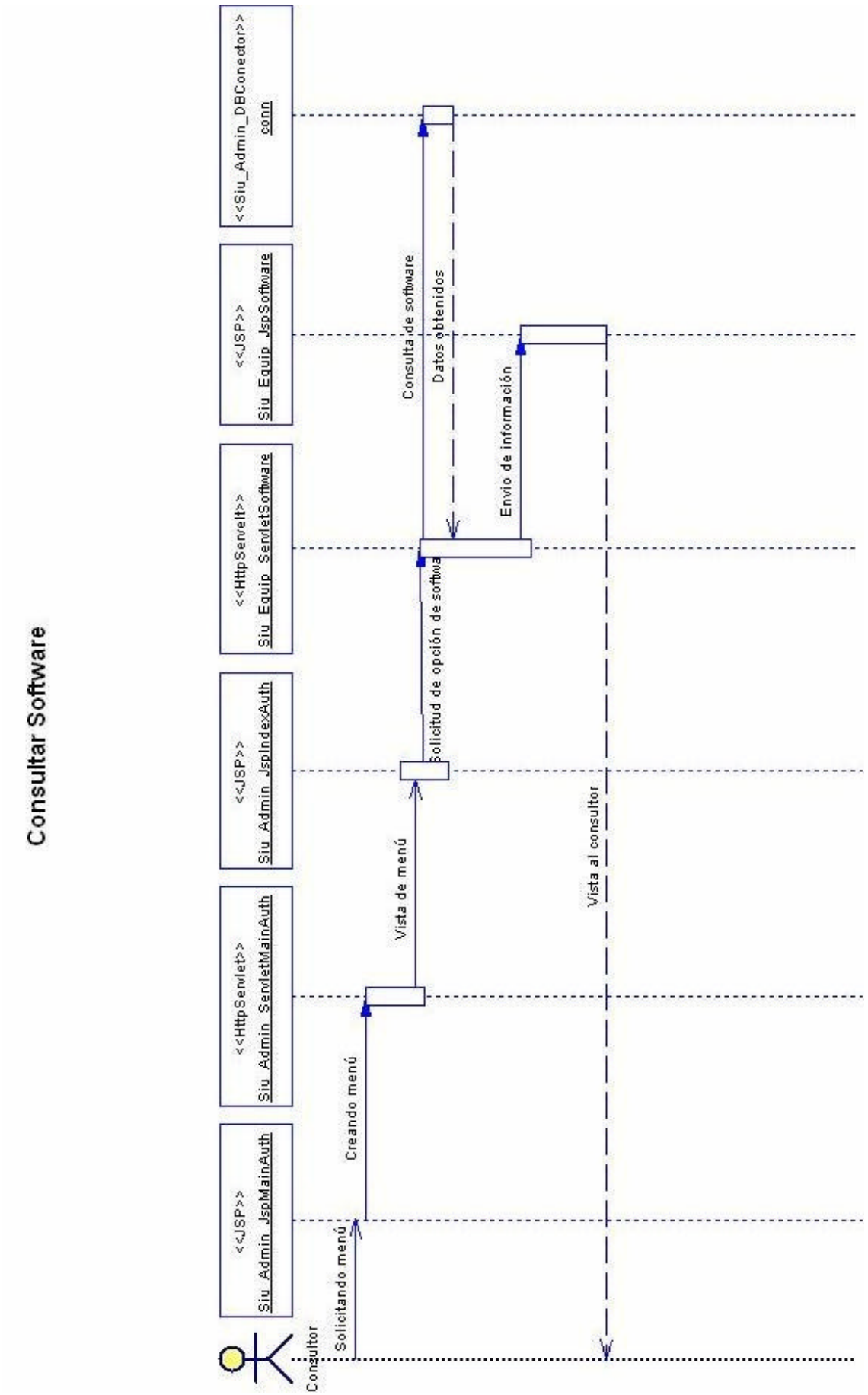
- Dado el punto anterior, ninguna etiqueta que sea mostrada en alguna de las vistas es estáticas, todas se obtienen dinámicamente.

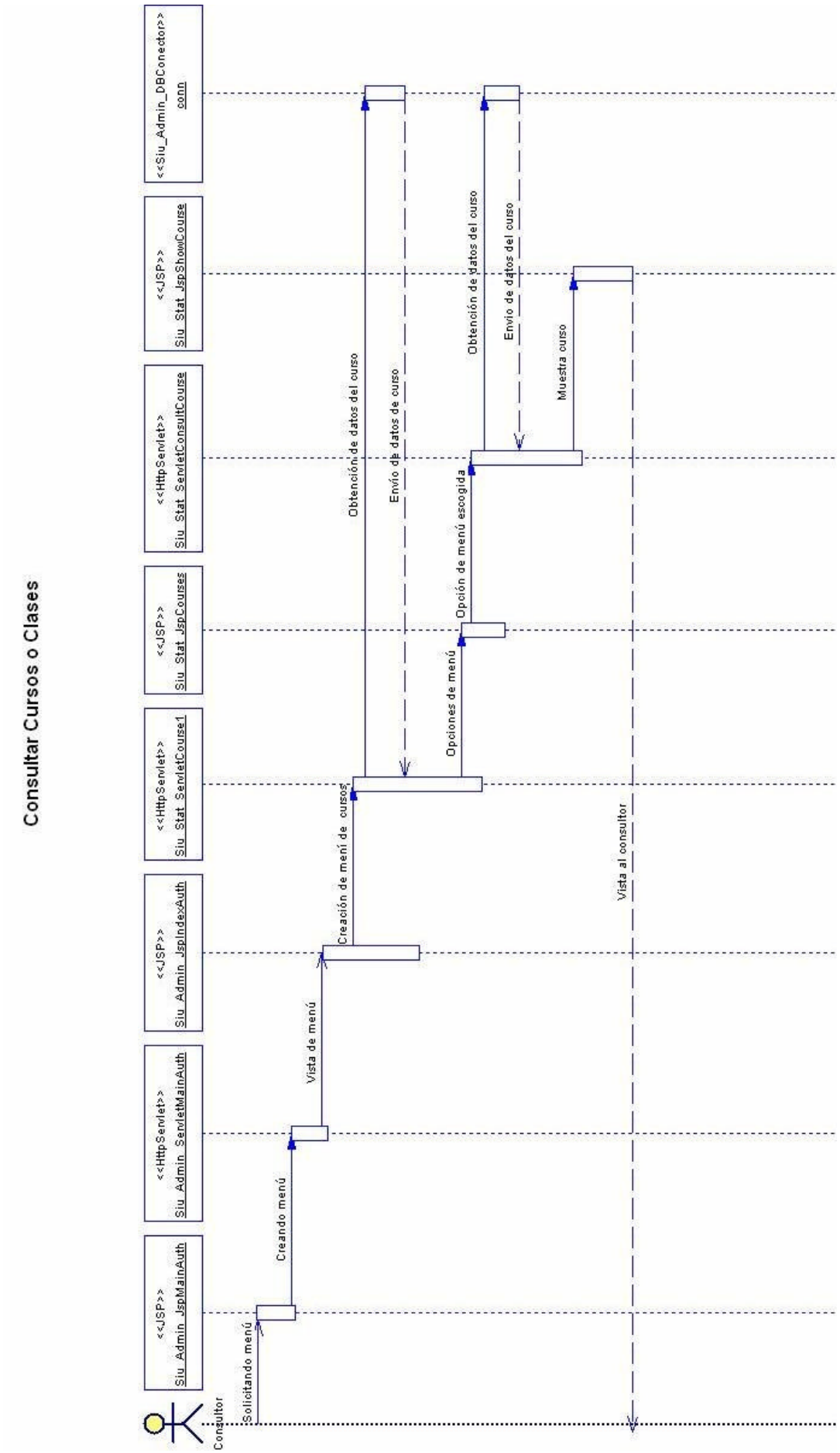
3.2. Diagramas de Secuencias

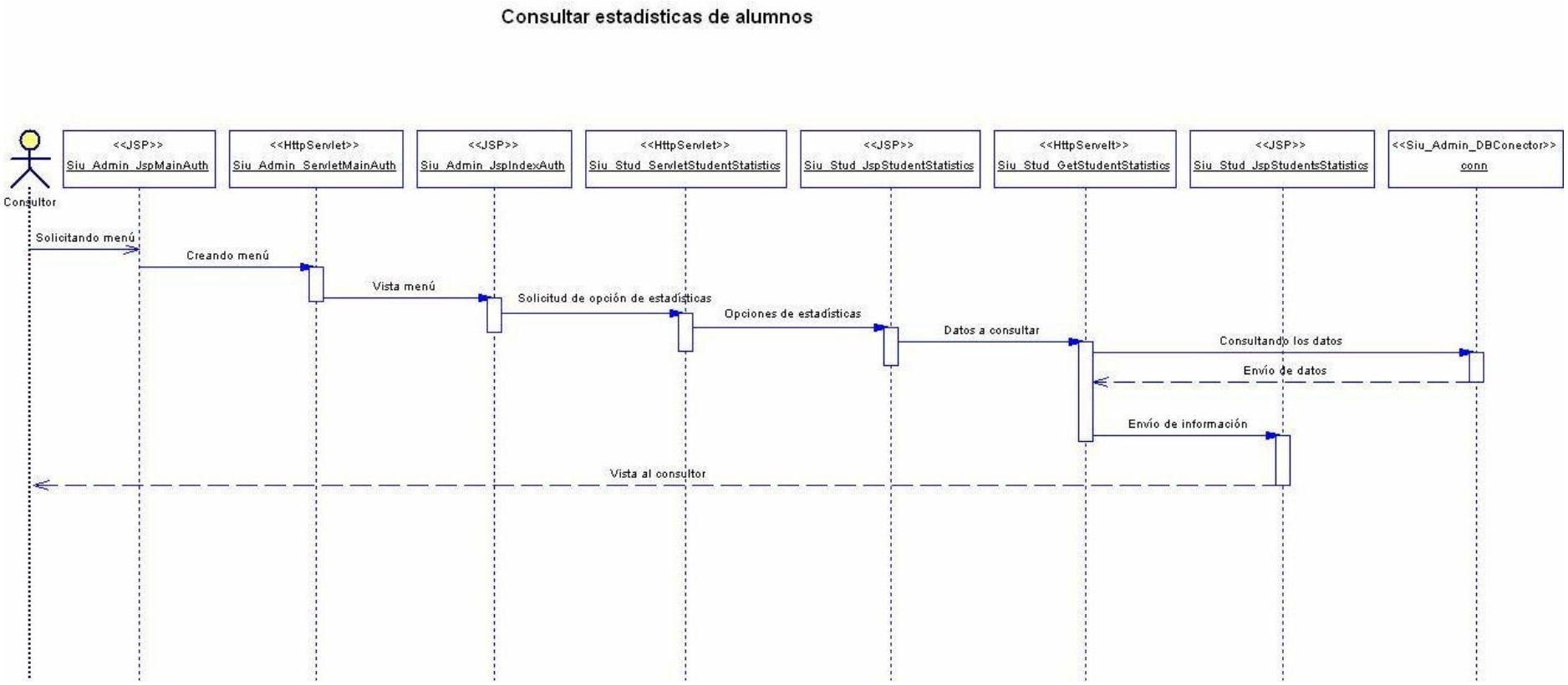
3.2.1. Zona de Vistas

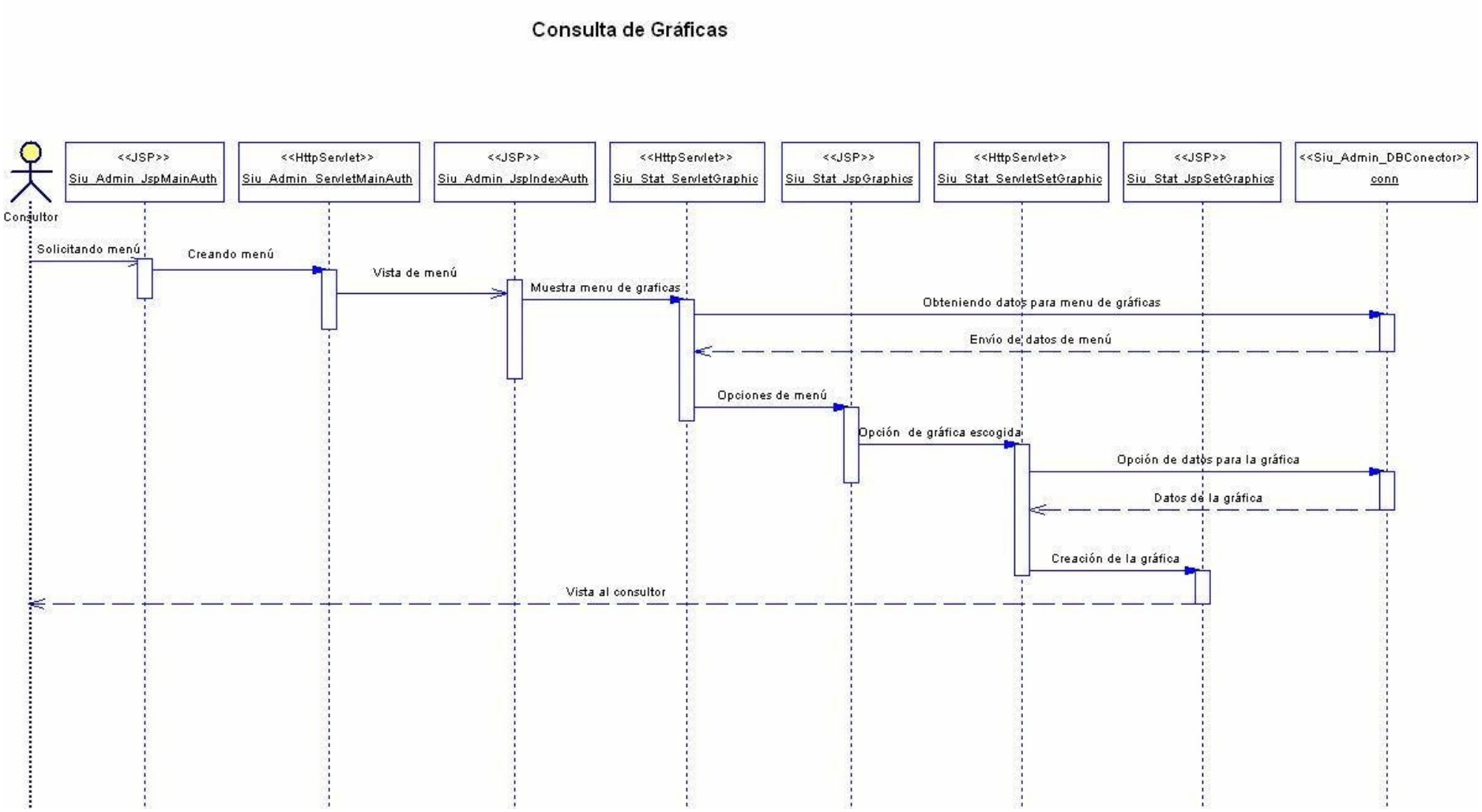


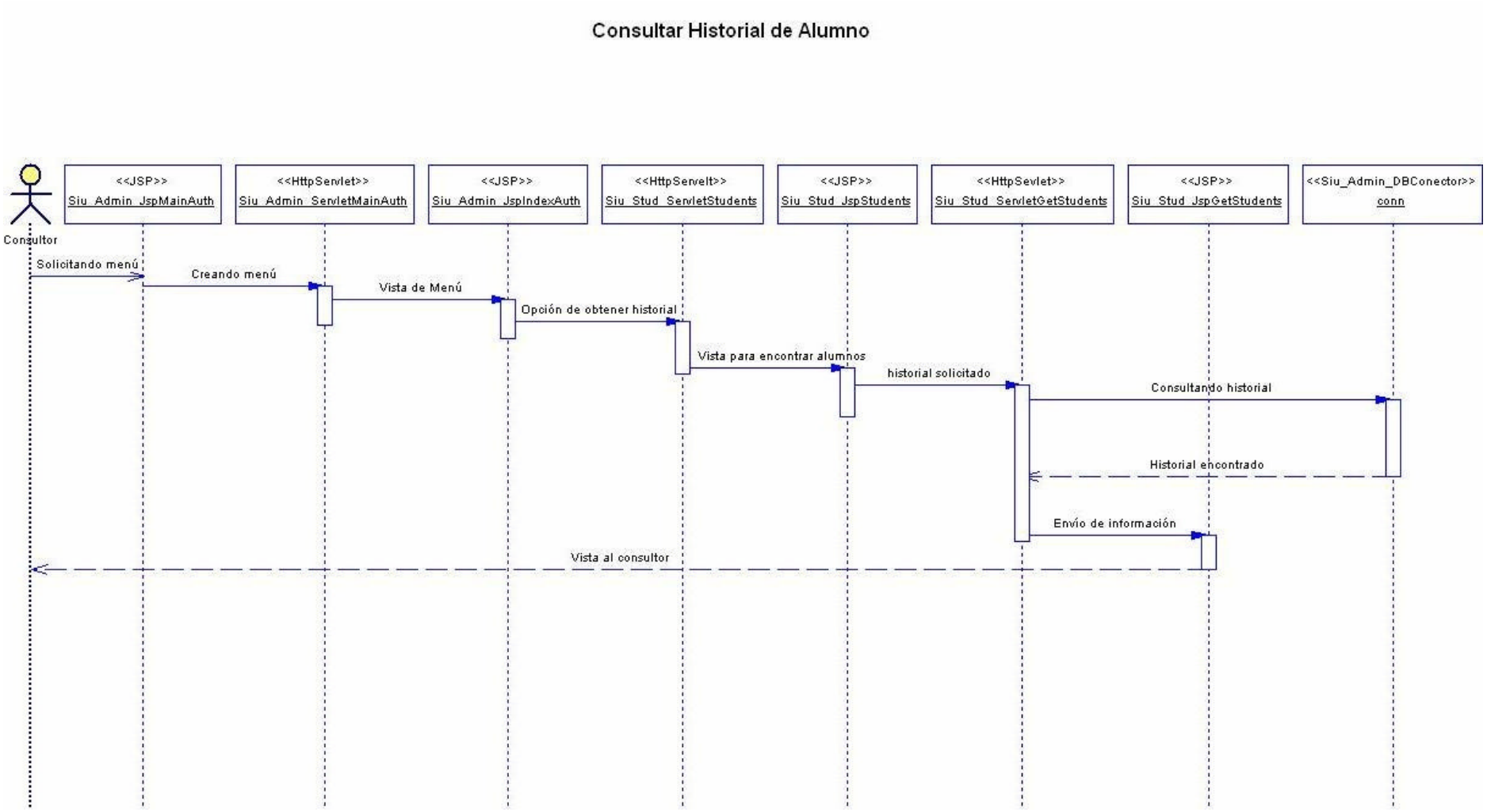

Consultar Hardware
Consultor
<<JSP>>
Siu_Admin_JspMainAuth
<<HttpServlet>>
Siu_Admin_ServletMainAuth
<<JSP>>
Siu_Admin_JspIndexAuth
<<HttpServlet>>
Siu_Equip_ServletHardware
<<JSP>>
Siu_Equip_JspHardware
<<Siu_Admin_DBConector>>
conn
Solicitando menú
Creando menú
Vista menú
Solicictud de opción de hardare
Consulta de hardware
Datos obtenidos
Envío de información
Vista al consultor


Consultar Software
<<JSP>>
Siu_Admin_JspMainAuth
<<HttpServlet>>
Siu_Admin_ServletMainAuth
<<JSP>>
Siu_Admin_JspIndexAuth
<<HttpServlet>>
Siu_Equip_ServletSoftware
<<JSP>>
Siu_Equip_JspSoftware
<<Siu_Admin_DBConector>>
conn
Consultor
Solicitando menú
Creando menú
Vista de menú
Solicitud de opción de softwa
Consulta de software
Datos obtenidos
Envio de información
Vista al consultor

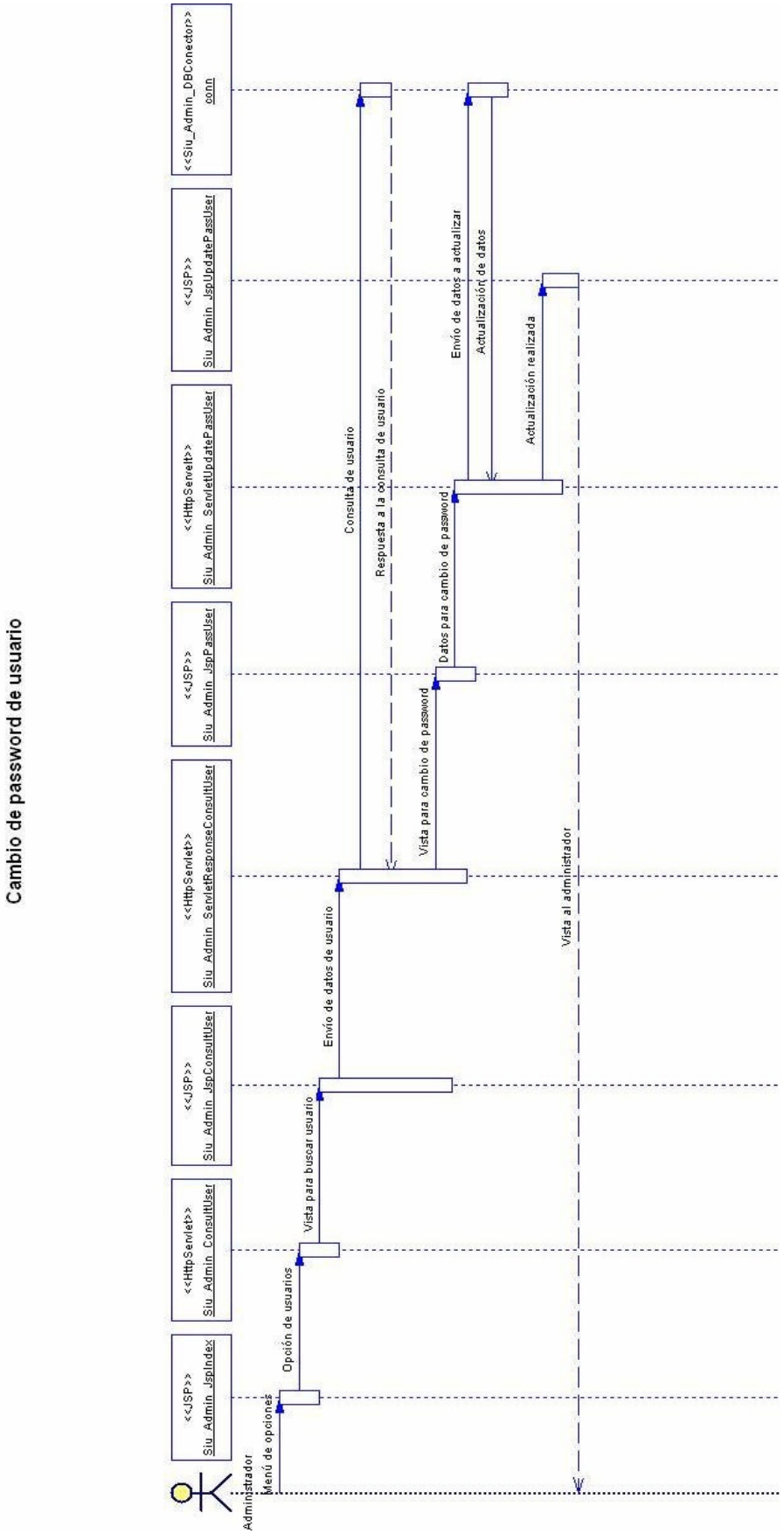

Consultar Cursos o Clases
Consultor
<<JSP>>
Siu_Admin_JspMainAuth
<<HttpServlet>>
Siu_Admin_ServletMainAuth
<<JSP>>
Siu_Admin_JspIndexAuth
<<HttpServlet>>
Siu_Stat_ServletCourse1
<<JSP>>
Siu_Stat_JspCourses
<<HttpServlet>>
Siu_Stat_ServletConsultCourse
<<JSP>>
Siu_Stat_JspShowCourse
<<Siu_Admin_DBConector>>
conn
Solicitando menú
Creando menú
Vista de menú
Creación de menú de cursos
Obtención de datos del curso
Envío de datos de curso
Opciones de menú
Opción de menú escogida
Obtención de datos del curso
Envío de datos del curso
Muestra curso
Vista al consultor

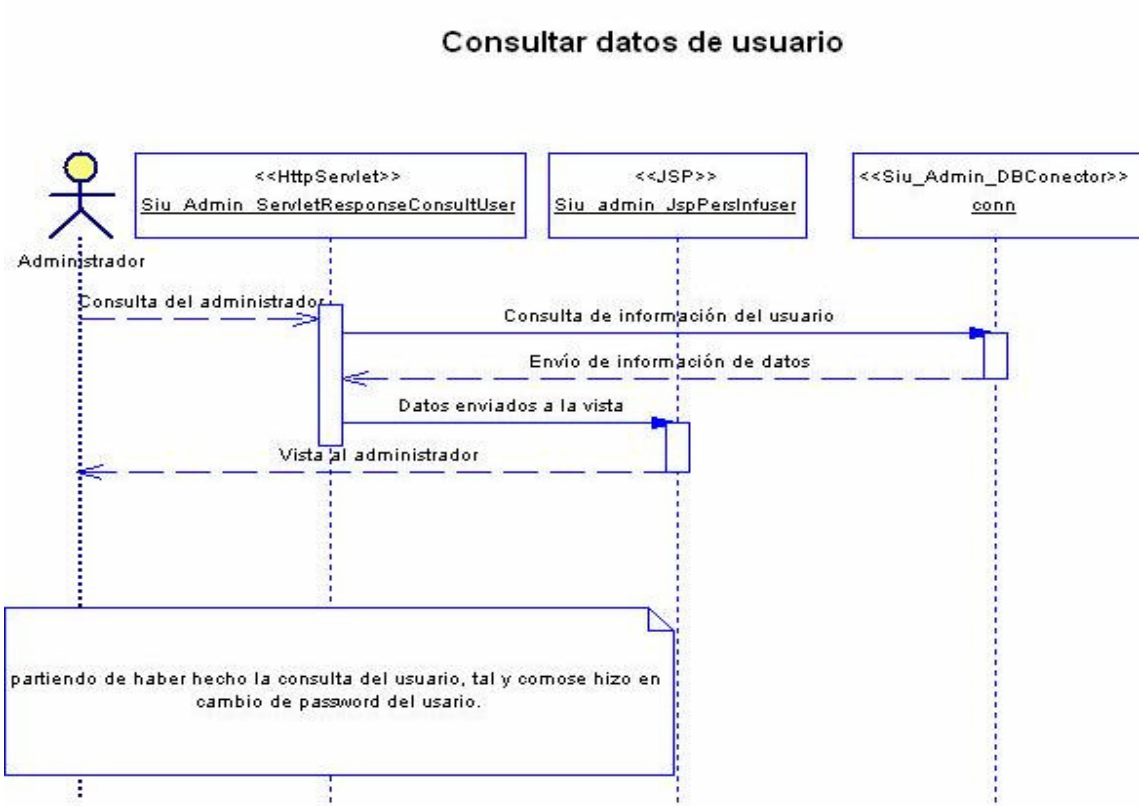

Consultar estadísticas de alumnos
Consultor
<<JSP>>
Siu_Admin_JspMainAuth
<<HttpServlet>>
Siu_Admin_ServletMainAuth
<<JSP>>
Siu_Admin_JspIndexAuth
<<HttpServlet>>
Siu_Stud_ServletStudentStatistics
<<JSP>>
Siu_Stud_JspStudentStatistics
<<HttpServelt>>
Siu_Stud_GetStudentStatistics
<<JSP>>
Siu_Stud_JspStudentsStatistics
<<Siu_Admin_DBConector>>
conn
Solicitando menú
Creando menú
Vista menú
Solicitud de opción de estadísticas
Opciones de estadísticas
Datos a consultar
Consultando los datos
Envío de datos
Envío de información
Vista al consultor

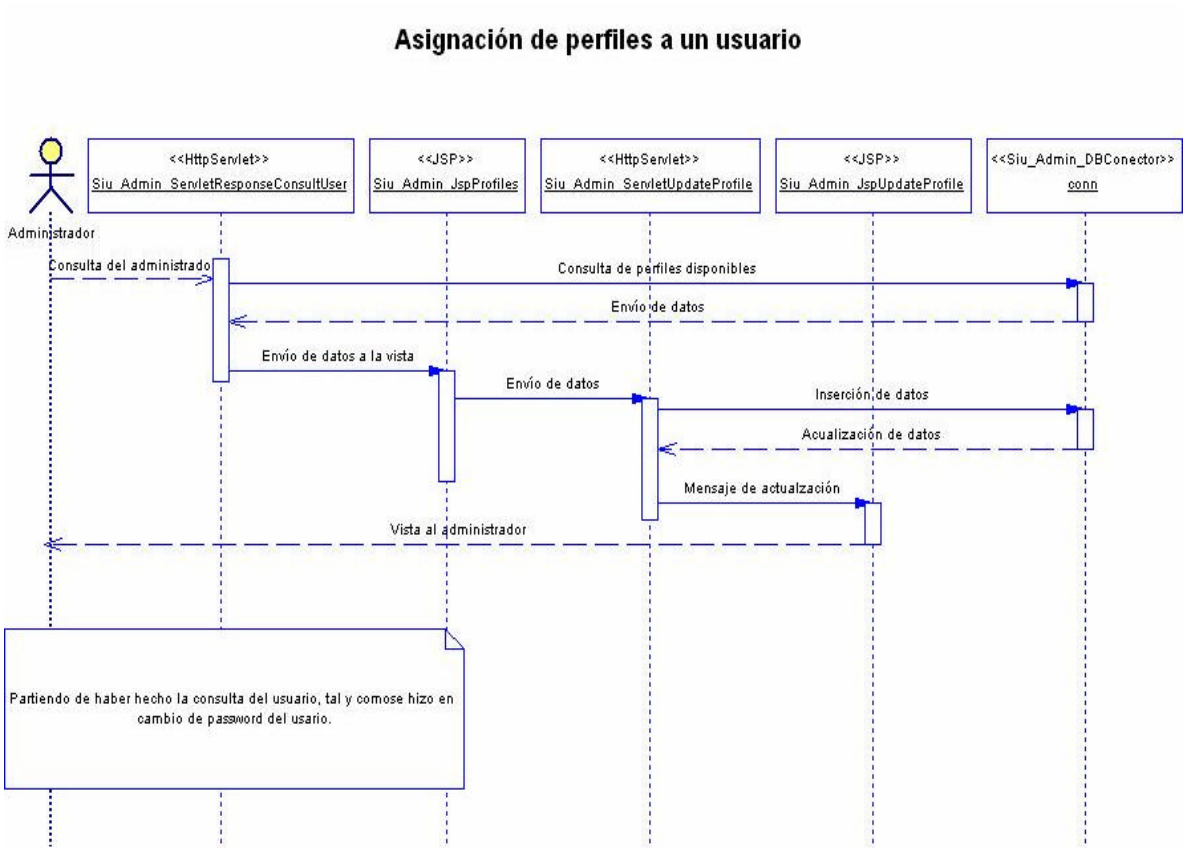

Consulta de Gráficas
Consultor
<<JSP>>
Siu_Admin_JspMainAuth
<<HttpServlet>>
Siu_Admin_ServletMainAuth
<<JSP>>
Siu_Admin_JspIndexAuth
<<HttpServlet>>
Siu_Stat_ServletGraphic
<<JSP>>
Siu_Stat_JspGraphics
<<HttpServlet>>
Siu_Stat_ServletSetGraphic
<<JSP>>
Siu_Stat_JspSetGraphics
<<Siu_Admin_DBConector>>
conn
Solicitando menú
Creando menú
Vista de menú
Muestra menu de graficas
Obteniendo datos para menu de gráficas
Envío de datos de menú
Opciones de menú
Opción de gráfica escogida
Opción de datos para la gráfica
Datos de la gráfica
Creación de la gráfica
Vista al consultor

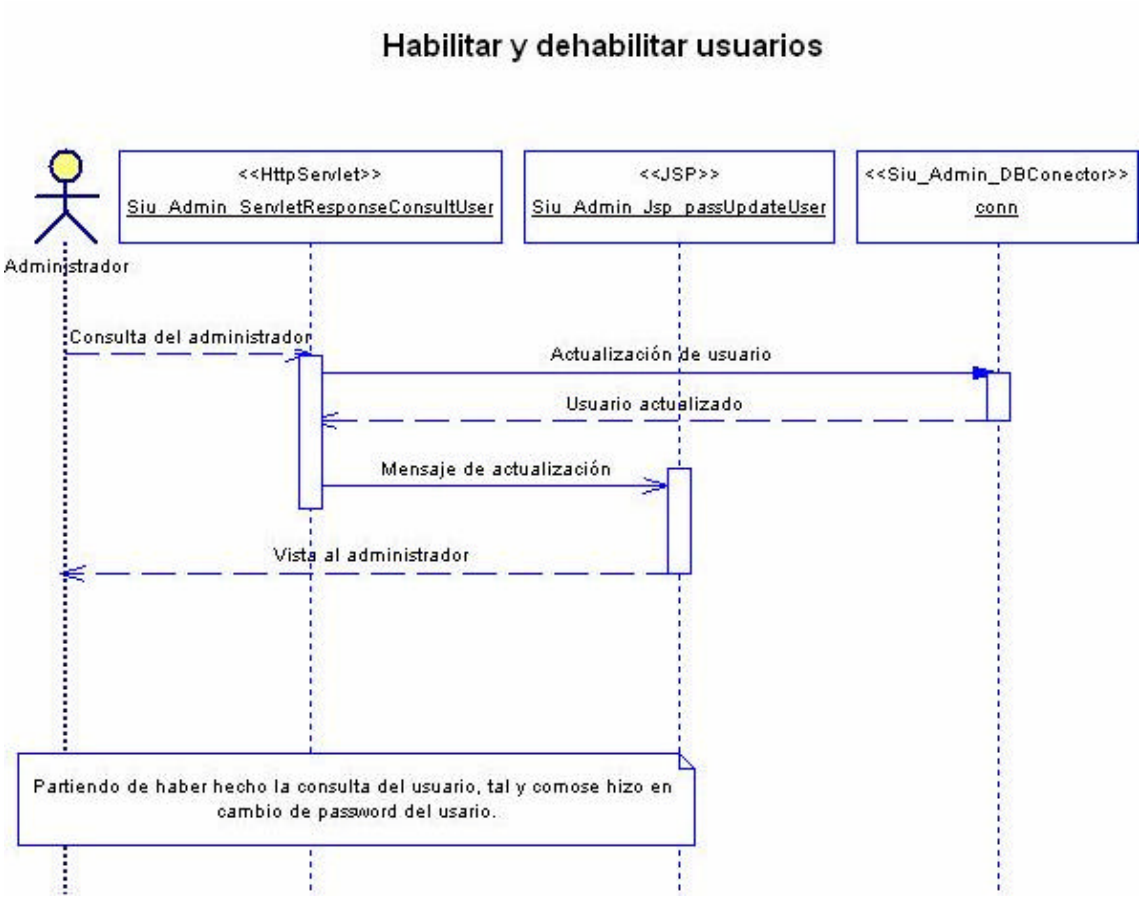

Consultar Historial de Alumno
Consultor
<<JSP>>
Siu_Admin_JspMainAuth
<<HttpServlet>>
Siu_Admin_ServletMainAuth
<<JSP>>
Siu_Admin_JspIndexAuth
<<HttpServlet>>
Siu_Stud_ServletStudents
<<JSP>>
Siu_Stud_JspStudents
<<HttpServlet>>
Siu_Stud_ServletGetStudents
<<JSP>>
Siu_Stud_JspGetStudents
<<Siu_Admin_DBConector>>
conn
Solicitando menú
Creando menú
Vista de Menú
Opción de obtener historial
Vista para encontrar alumnos
historial solicitado
Consultando historial
Historial encontrado
Envío de información
Vista al consultor

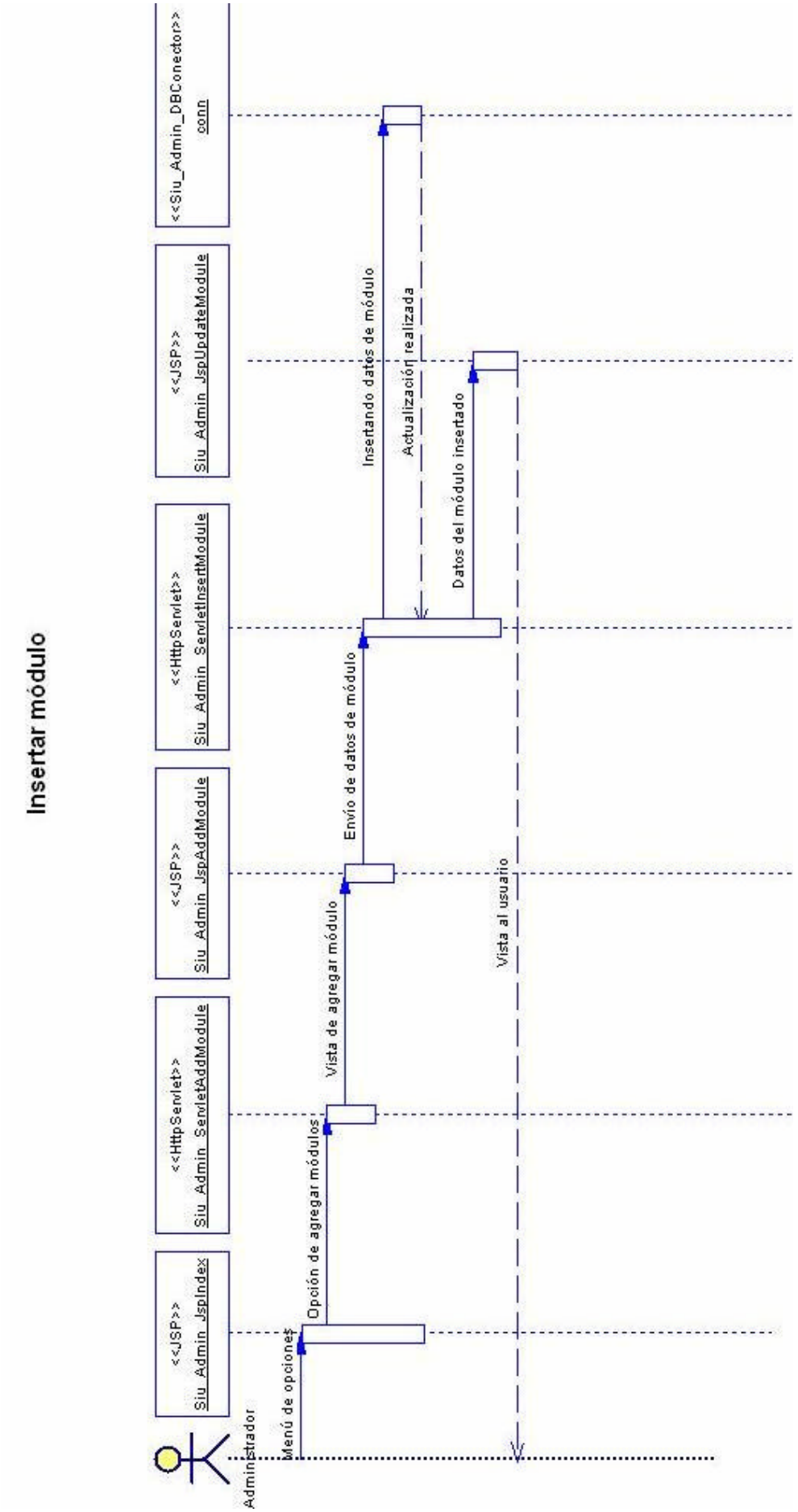
3.2.2. Zona de Administración

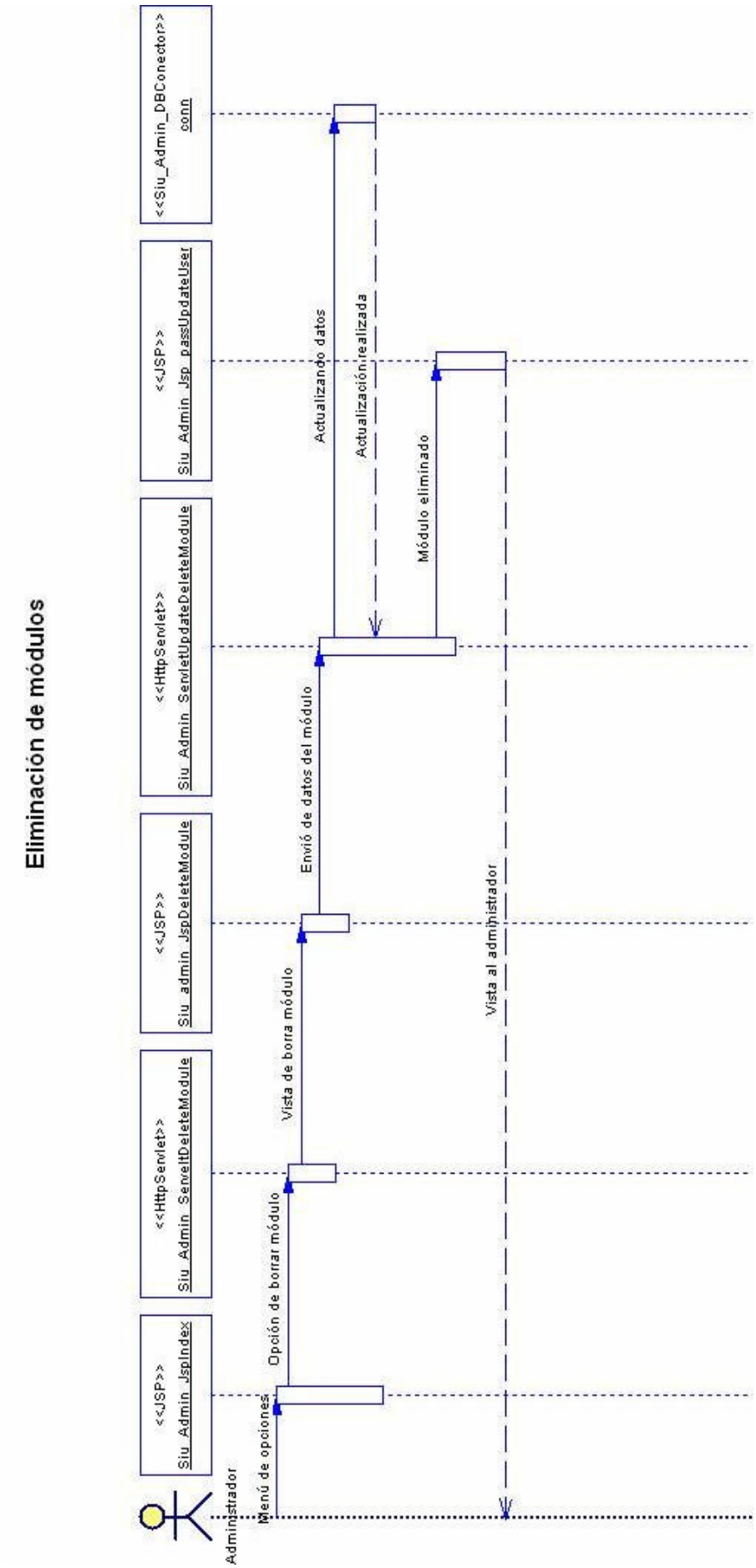


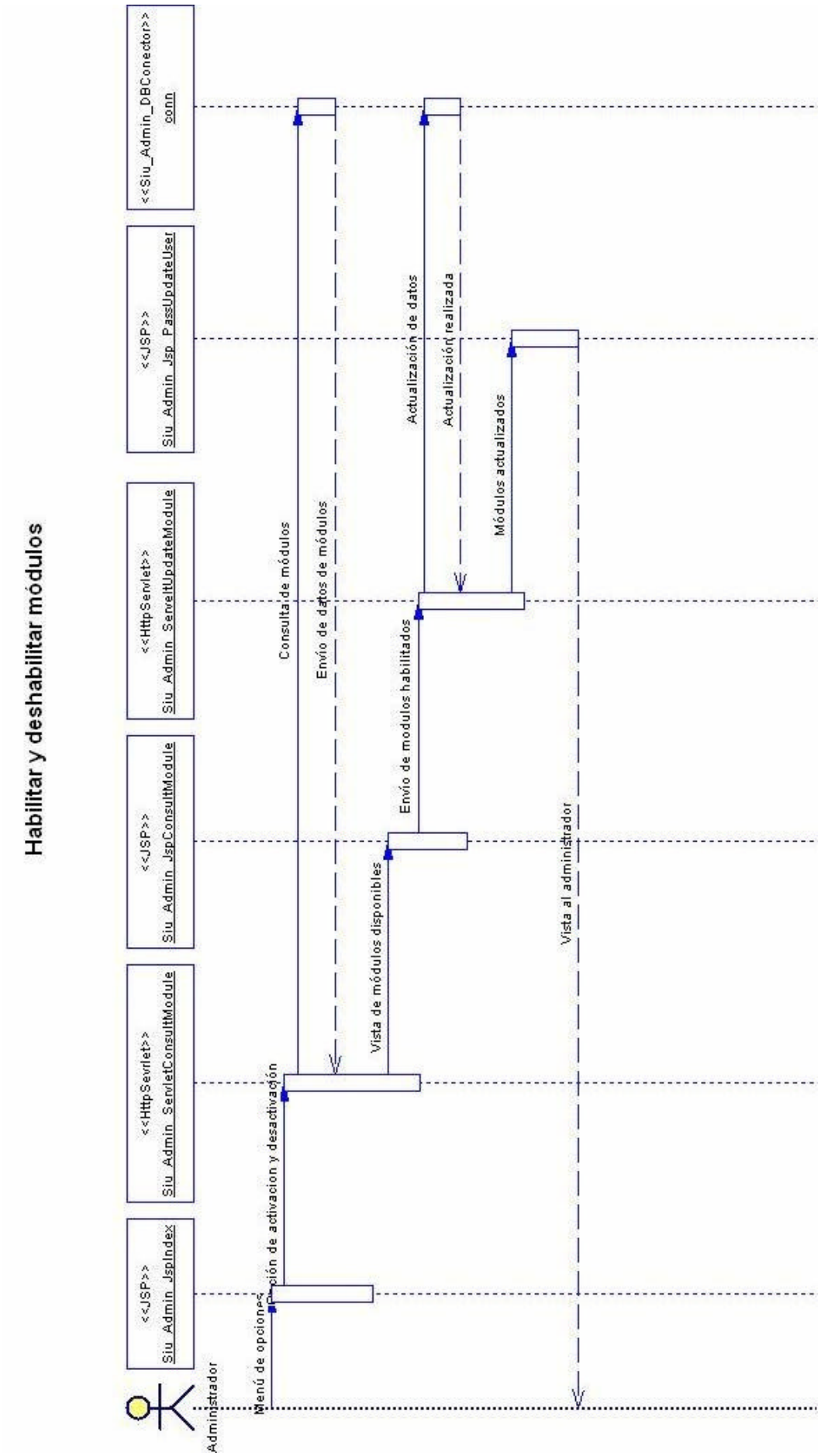

Consultar datos de usuario
<<HttpServlet>>
Siu_Admin_ServletResponseConsultUser
<<JSP>>
Siu_admin_JspPersInfuser
<<Siu_Admin_DBConector>>
conn
Administrador
Consulta del administrador
Consulta de información del usuario
Envío de información de datos
Datos enviados a la vista
Vista al administrador
partiendo de haber hecho la consulta del usuario, tal y comose hizo en cambio de password del usario.

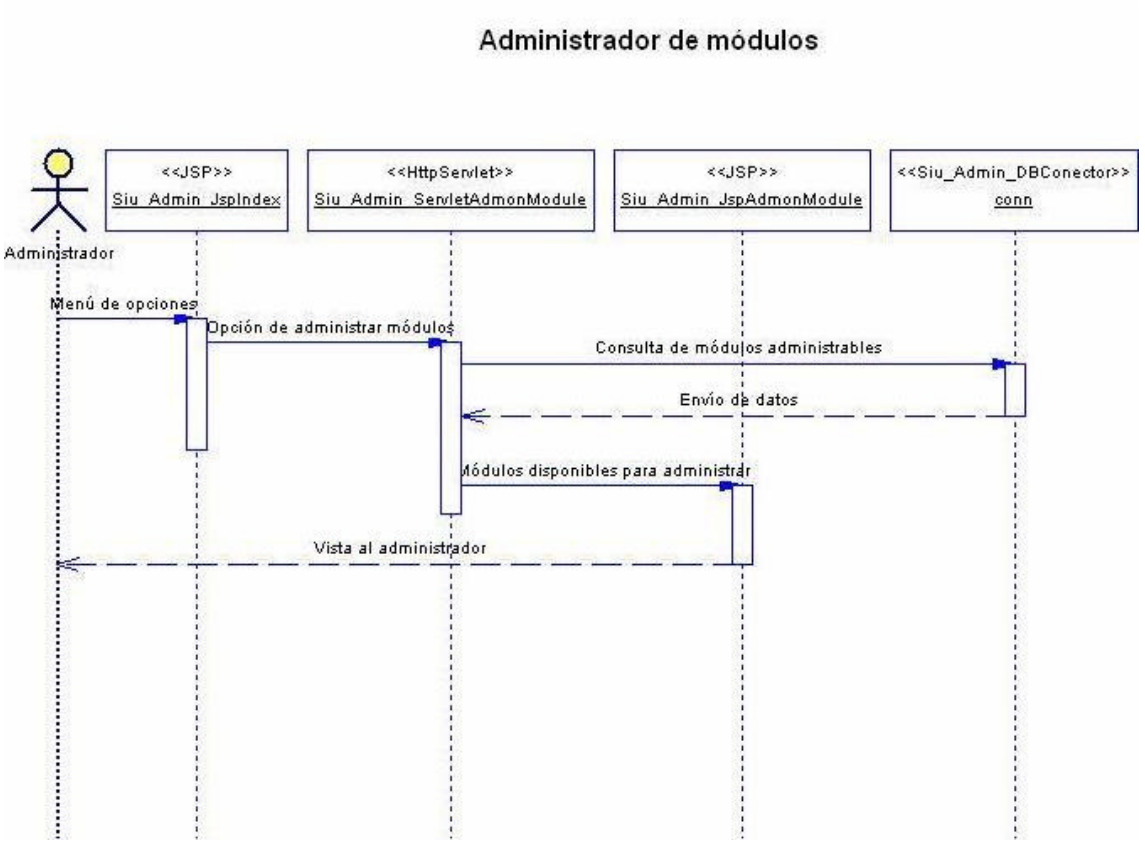

Asignación de perfiles a un usuario
<<HttpServlet>>
Siu_Admin_ServletResponseConsultUser
<<JSP>>
Siu_Admin_JspProfiles
<<HttpServlet>>
Siu_Admin_ServletUpdateProfile
<<JSP>>
Siu_Admin_JspUpdateProfile
<<Siu_Admin_DBConector>>
conn
Administrador
Consulta del administrado
Consulta de perfiles disponibles
Envío de datos
Envío de datos a la vista
Envío de datos
Inserción de datos
Acualización de datos
Mensaje de actualzación
Vista al administrador
Partiendo de haber hecho la consulta del usuario, tal y comose hizo en cambio de password del usario.


Habilitar y dehabilitar usuarios
Administrador
<<HttpServlet>>
Siu_Admin_ServletResponseConsultUser
<<JSP>>
Siu_Admin_Jsp_passUpdateUser
<<Siu_Admin_DBConector>>
conn
Consulta del administrador
Actualización de usuario
Usuario actualizado
Mensaje de actualización
Vista al administrador
Partiendo de haber hecho la consulta del usuario, tal y comose hizo en cambio de password del usario.


Insertar módulo
Administrador
<<JSP>>
Siu_Admin_JspIndex
<<HttpServlet>>
Siu_Admin_ServletAddModule
<<JSP>>
Siu_Admin_JspAddModule
<<HttpServlet>>
Siu_Admin_ServletInsertModule
<<JSP>>
Siu_Admin_JspUpdateModule
<<Siu_Admin_DBConector>>
conn
Menú de opciones
Opción de agregar módulos
Vista de agregar módulo
Envío de datos de módulo
Insertando datos de módulo
Actualización realizada
Datos del módulo insertado
Vista al usuario

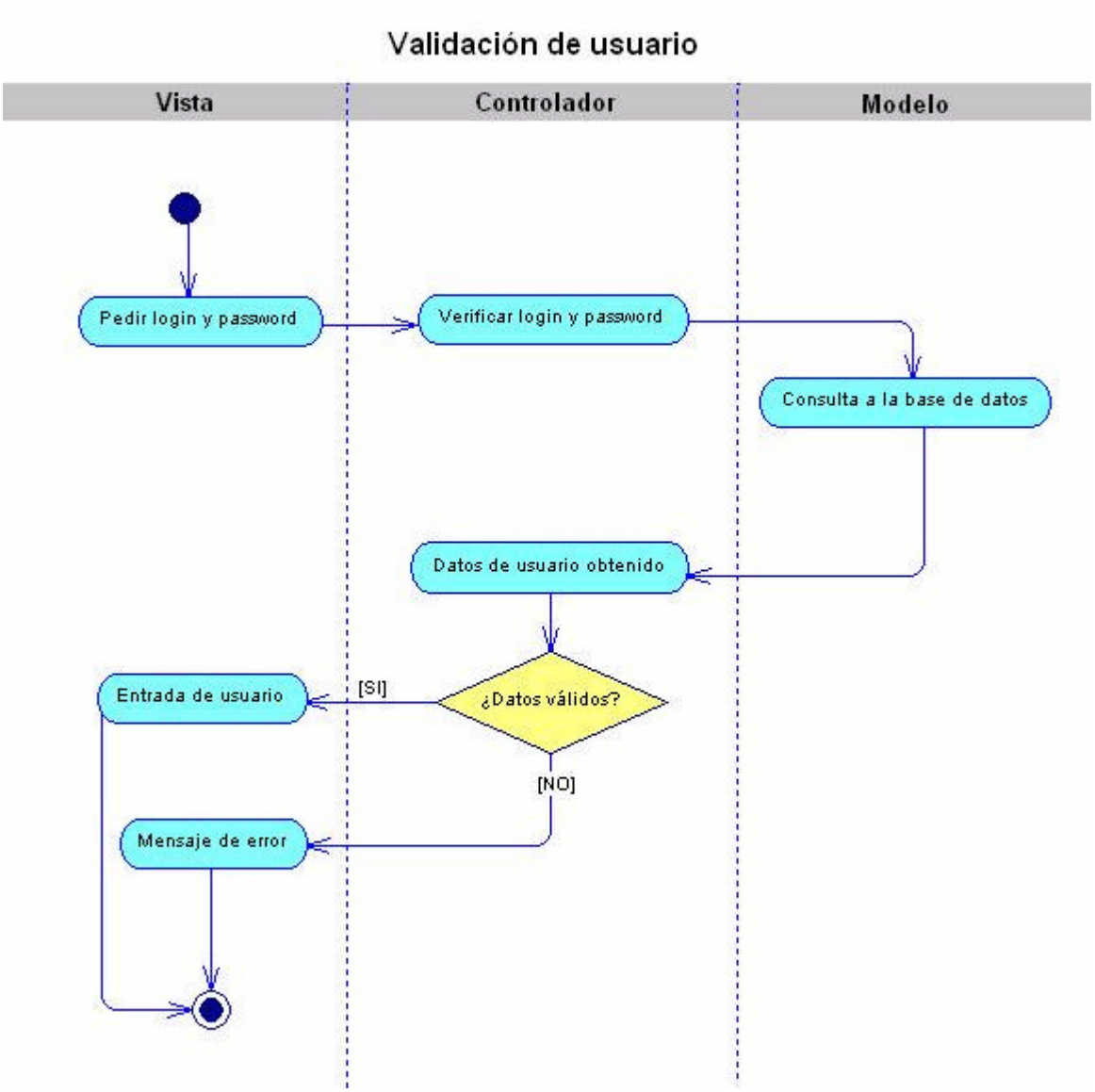

Eliminación de módulos
Administrador
<<JSP>>
Siu_Admin_JspIndex
<<HttpServlet>>
Siu_Admin_ServeltDeleteModule
<<JSP>>
Siu_admin_JspDeleteModule
<<HttpServlet>>
Siu_Admin_ServletUpdateDeleteModule
<<JSP>>
Siu_Admin_Jsp_passUpdateUser
<<Siu_Admin_DBConector>>
conn
Menú de opciones
Opción de borrar módulo
Vista de borra módulo
Envió de datos del módulo
Actualizando datos
Actualización realizada
Módulo eliminado
Vista al administrador

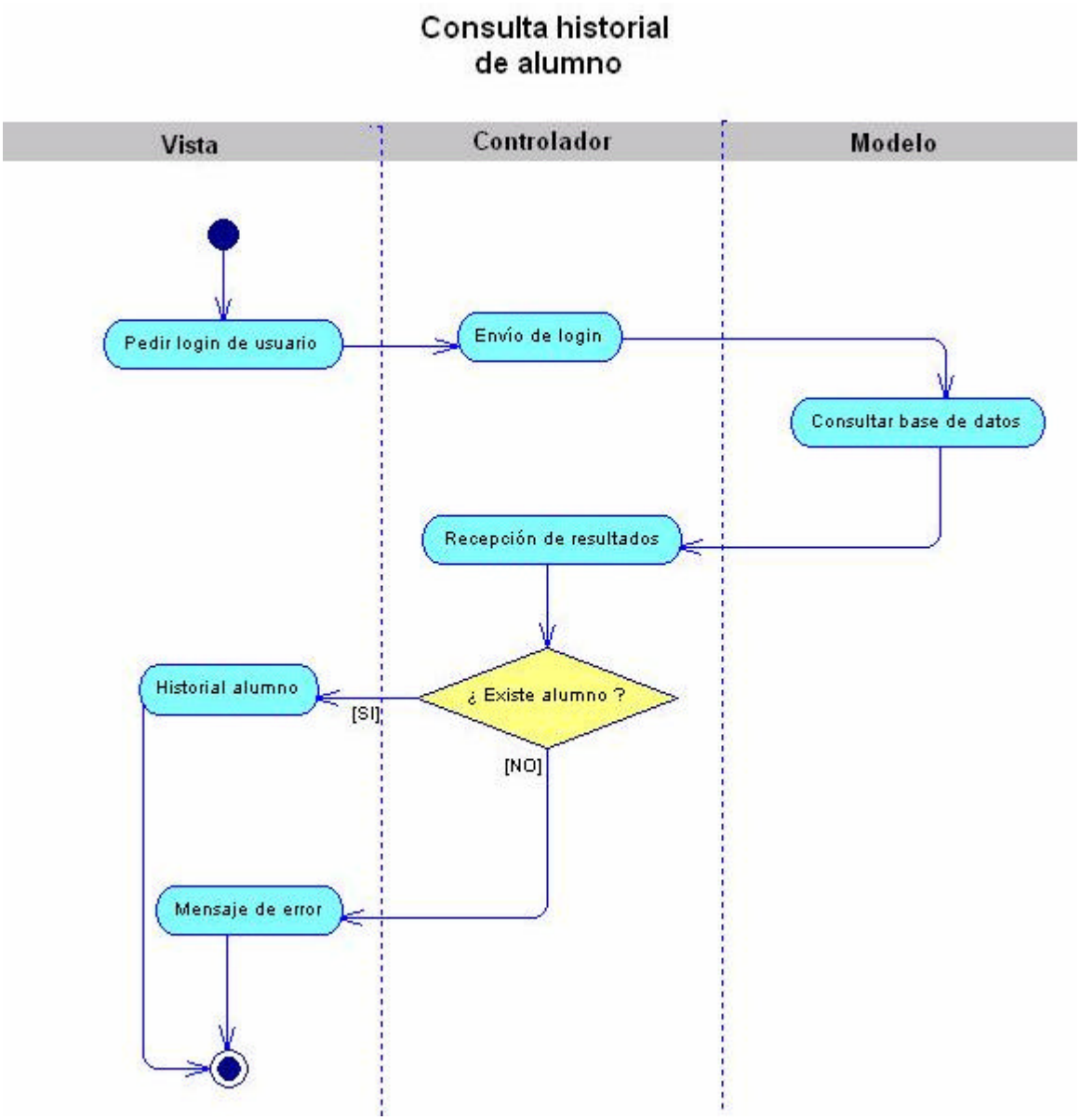

Habilitar y deshabilitar módulos
Administrador
<<JSP>>
Siu_Admin_JspIndex
<<HttpServlet>>
Siu_Admin_ServletConsultModule
<<JSP>>
Siu_Admin_JspConsultModule
<<HttpServlet>>
Siu_Admin_ServletUpdateModule
<<JSP>>
Siu_Admin_Jsp_PassUpdateUser
<<Siu_Admin_DBConector>>
conn
Menú de opciones
Opción de activacion y desactivación
Consulta de módulos
Envío de datos de módulos
Vista de módulos disponibles
Envío de modulos habilitados
Actualización de datos
Actualización realizada
Módulos actualizados
Vista al administrador

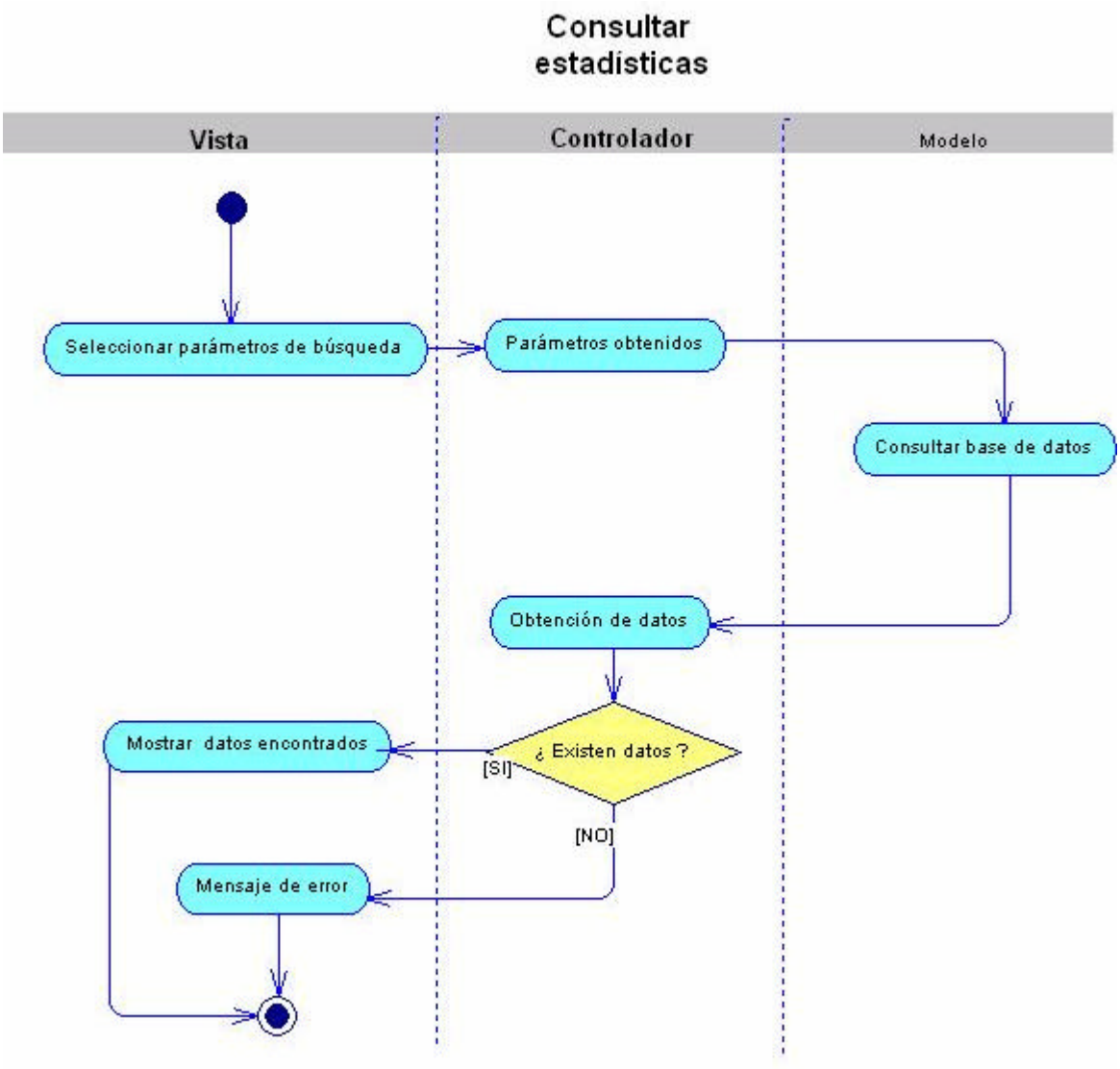

Administrador de módulos
<<JSP>>
Siu_Admin_JspIndex
<<HttpServlet>>
Siu_Admin_ServletAdmonModule
<<JSP>>
Siu_Admin_JspAdmonModule
<<Siu_Admin_DBConector>>
conn
Administrador
Menú de opciones
Opción de administrar módulos
Consulta de módulos administrables
Envío de datos
Módulos disponibles para administrar
Vista al administrador

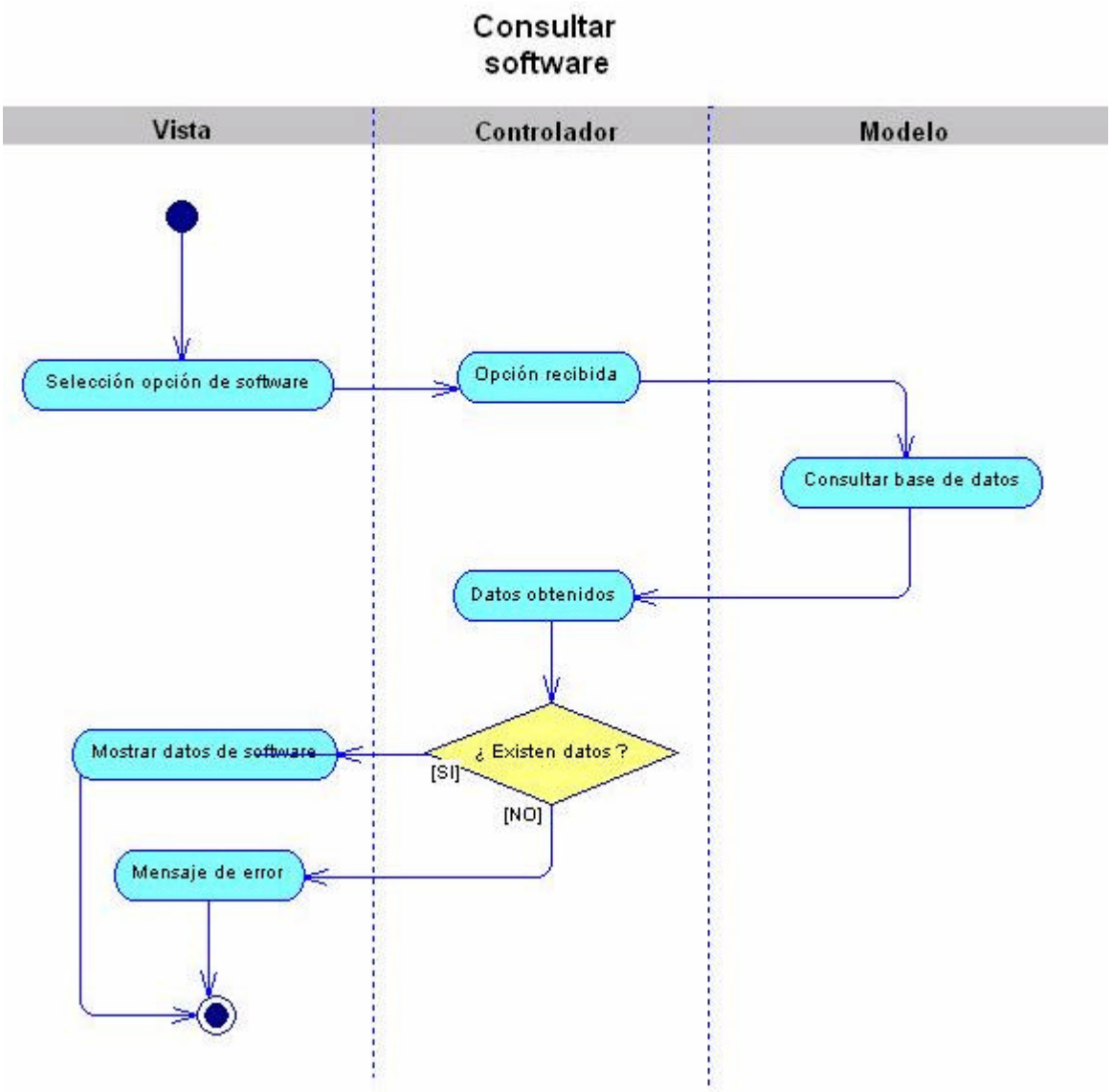
3.3. Diagramas de Actividades

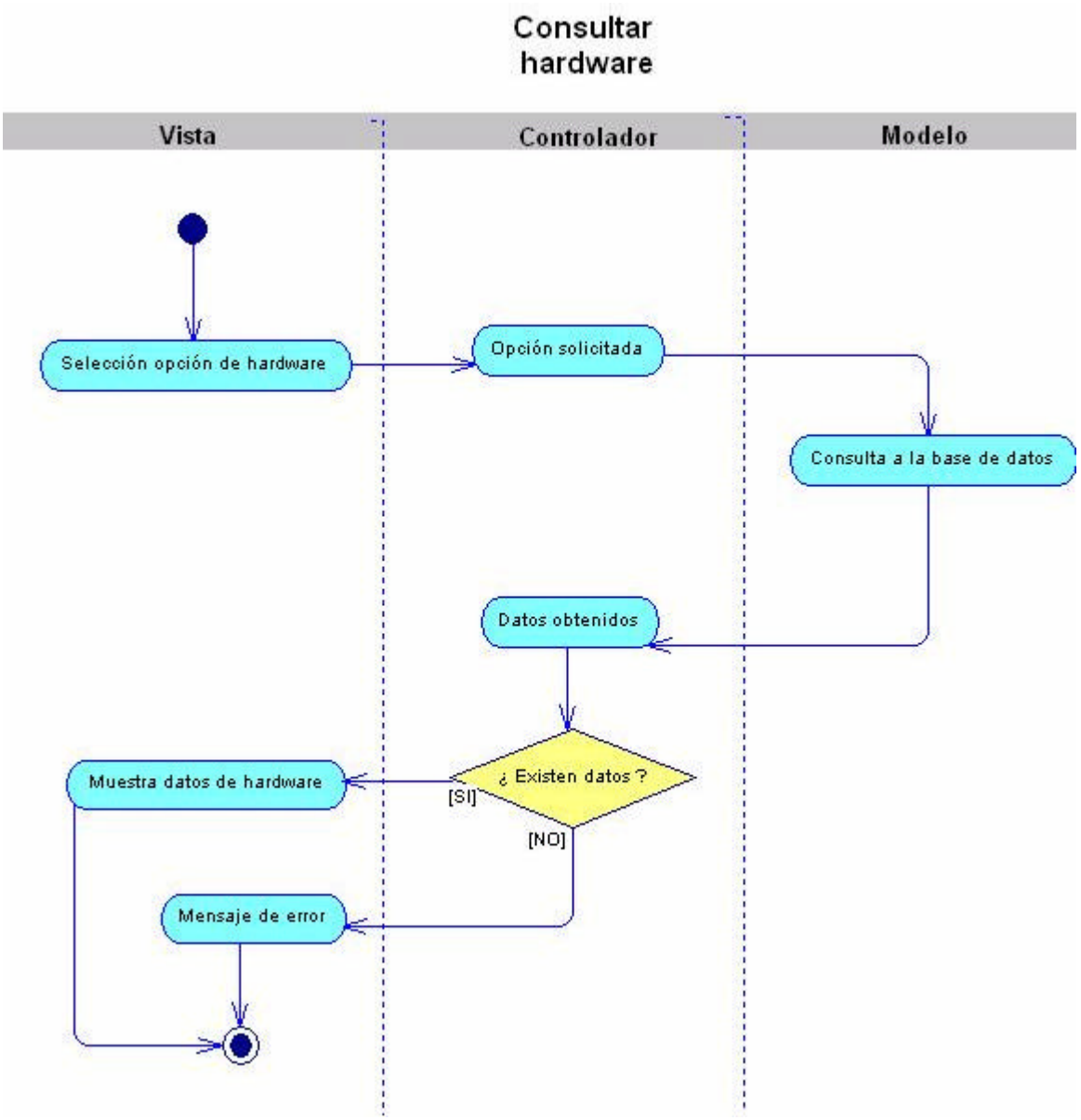
3.3.1. Zona de Vistas

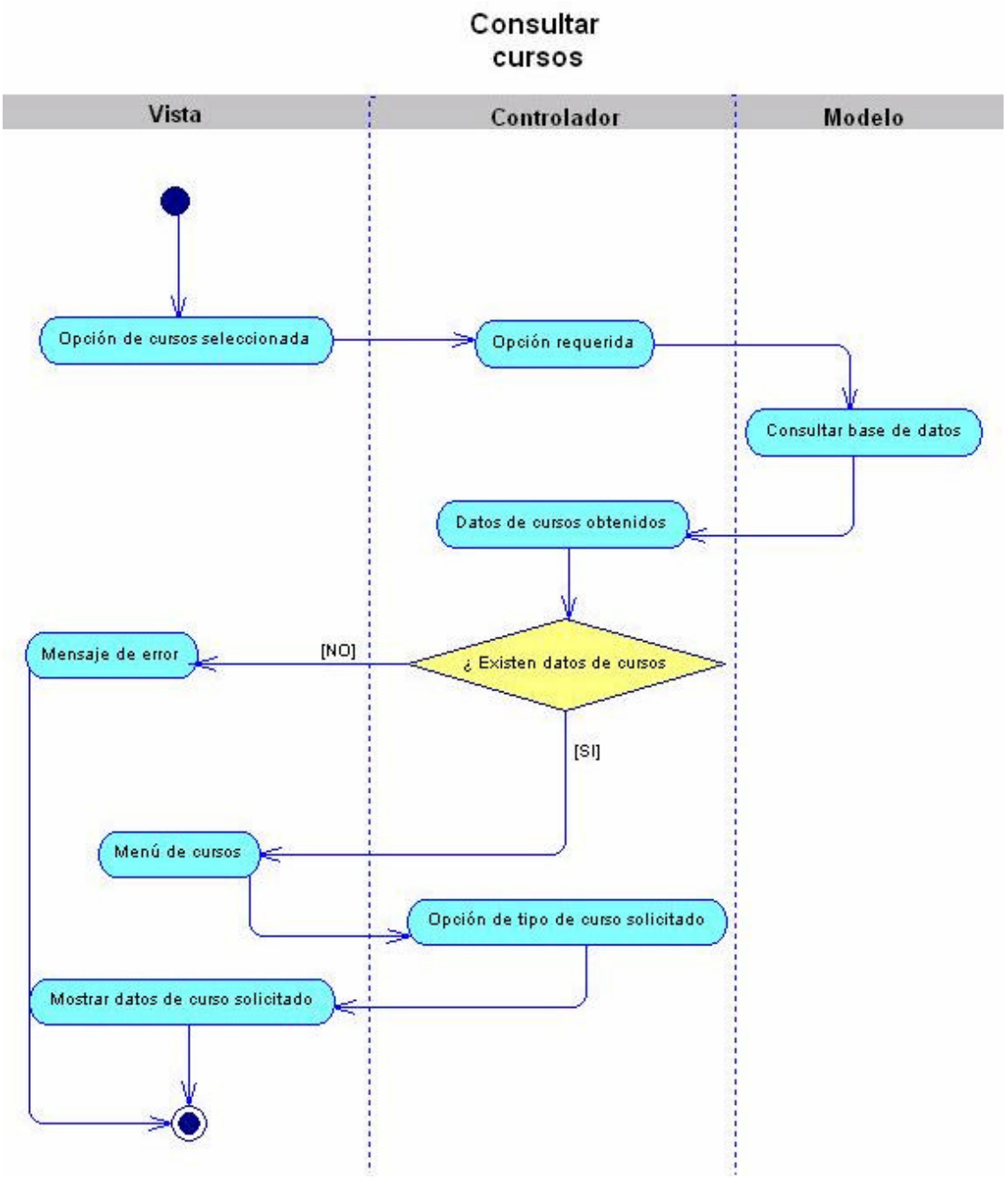


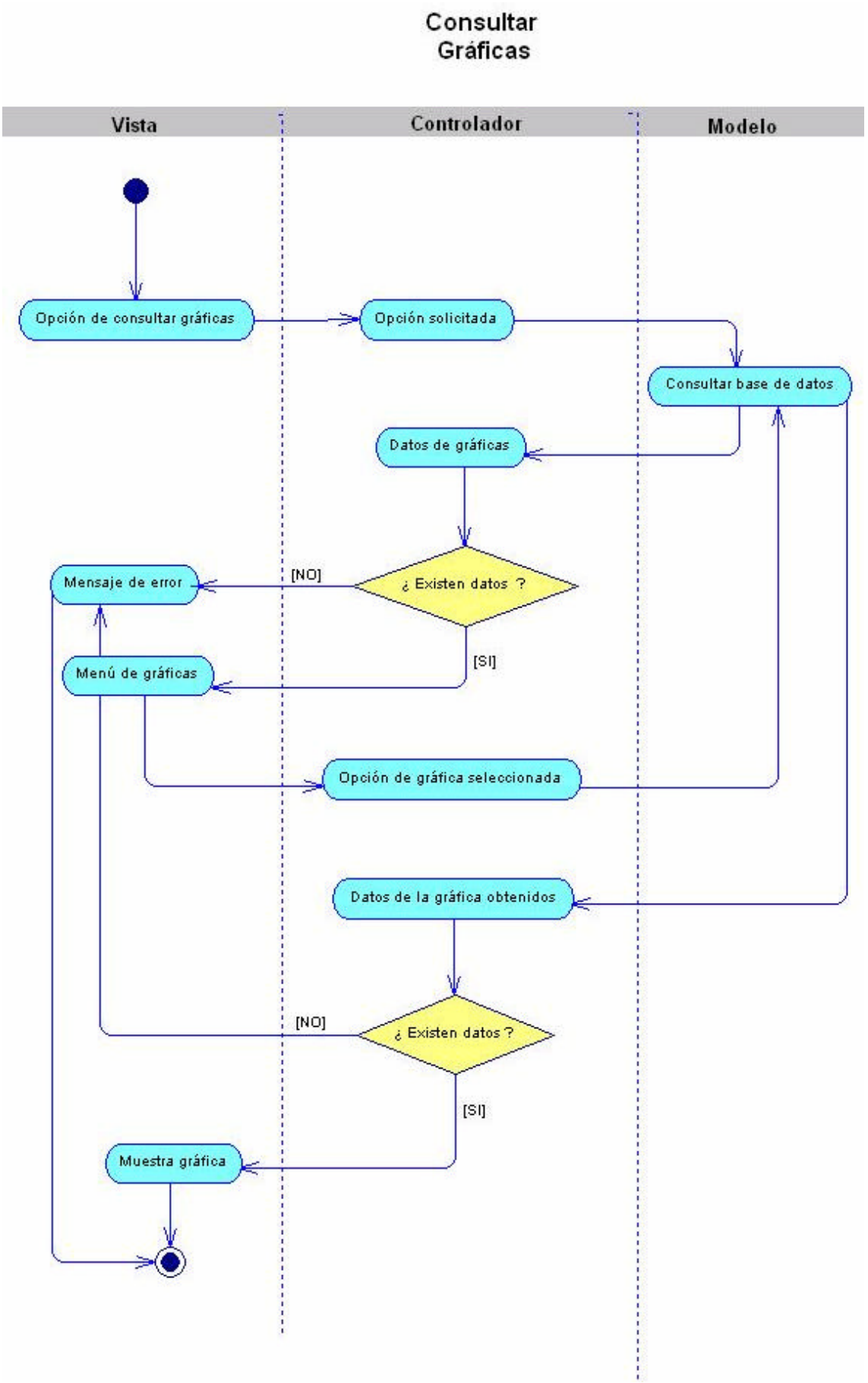

Consulta historial de alumno
Vista
Controlador
Modelo
Pedir login de usuario
Envío de login
Consultar base de datos
Recepción de resultados
¿ Existe alumno ?
[SI]
Historial alumno
[NO]
Mensaje de error


Consultar estadísticas
Vista
Controlador
Modelo
Seleccionar parámetros de búsqueda
Parámetros obtenidos
Consultar base de datos
Obtención de datos
¿ Existen datos ?
[SI]
[NO]
Mostrar datos encontrados
Mensaje de error

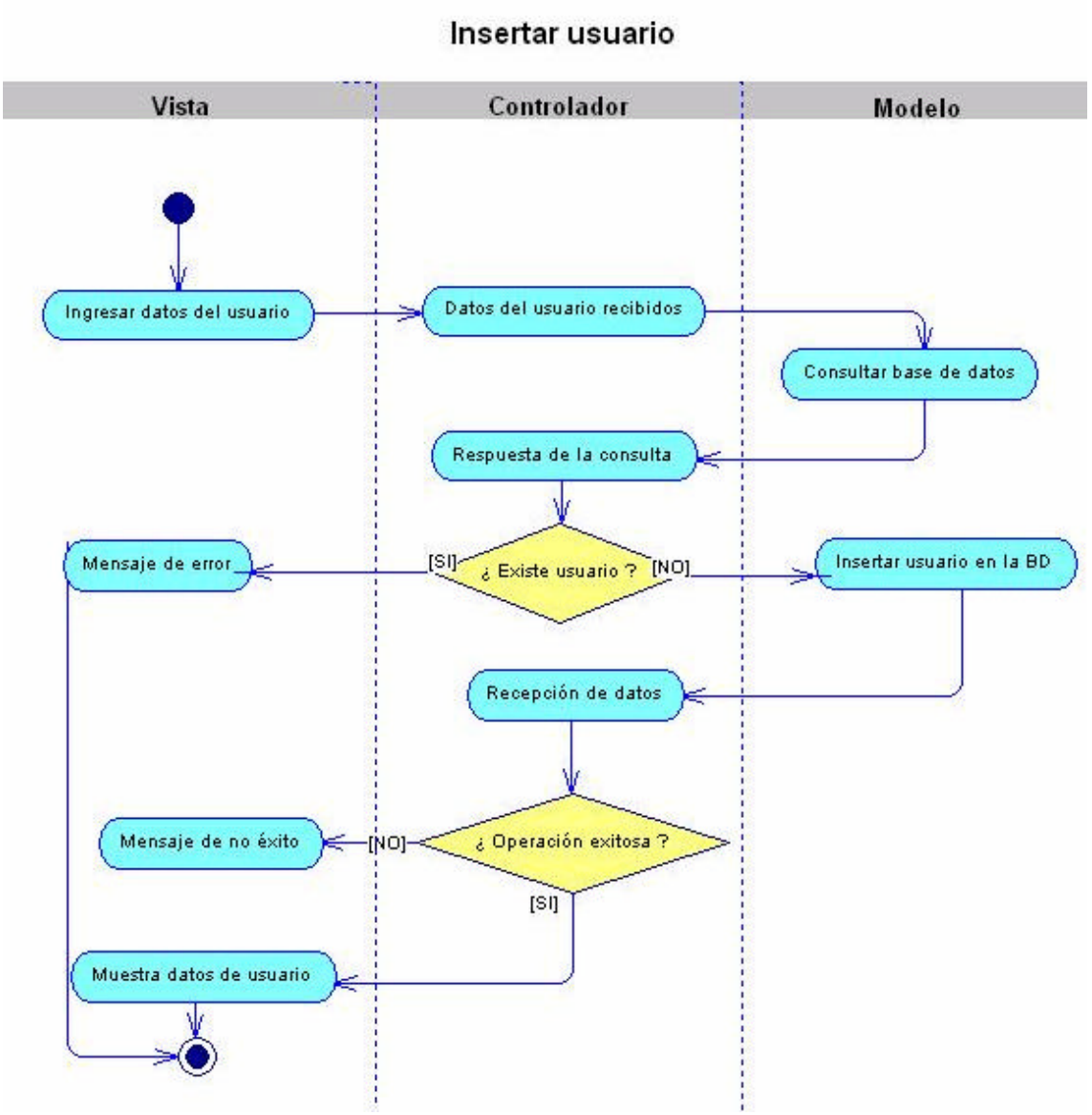

Consultar software
Vista
Controlador
Modelo
Selección opción de software
Opción recibida
Consultar base de datos
Datos obtenidos
¿ Existen datos ?
[SI]
[NO]
Mostrar datos de software
Mensaje de error

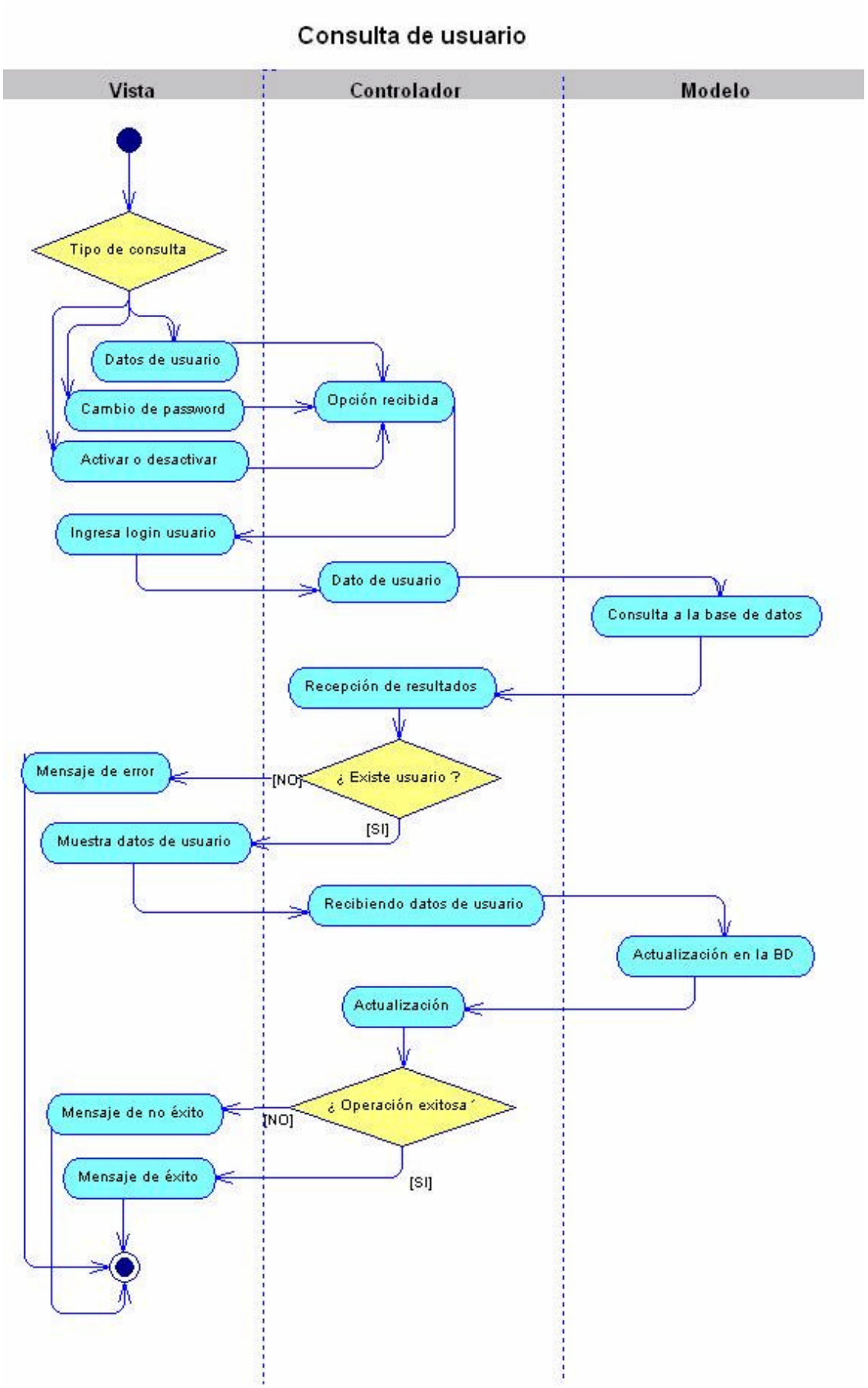

Consultar hardware
Vista
Controlador
Modelo
Selección opción de hardware
Opción solicitada
Consulta a la base de datos
Datos obtenidos
¿ Existen datos ?
[SI]
[NO]
Muestra datos de hardware
Mensaje de error

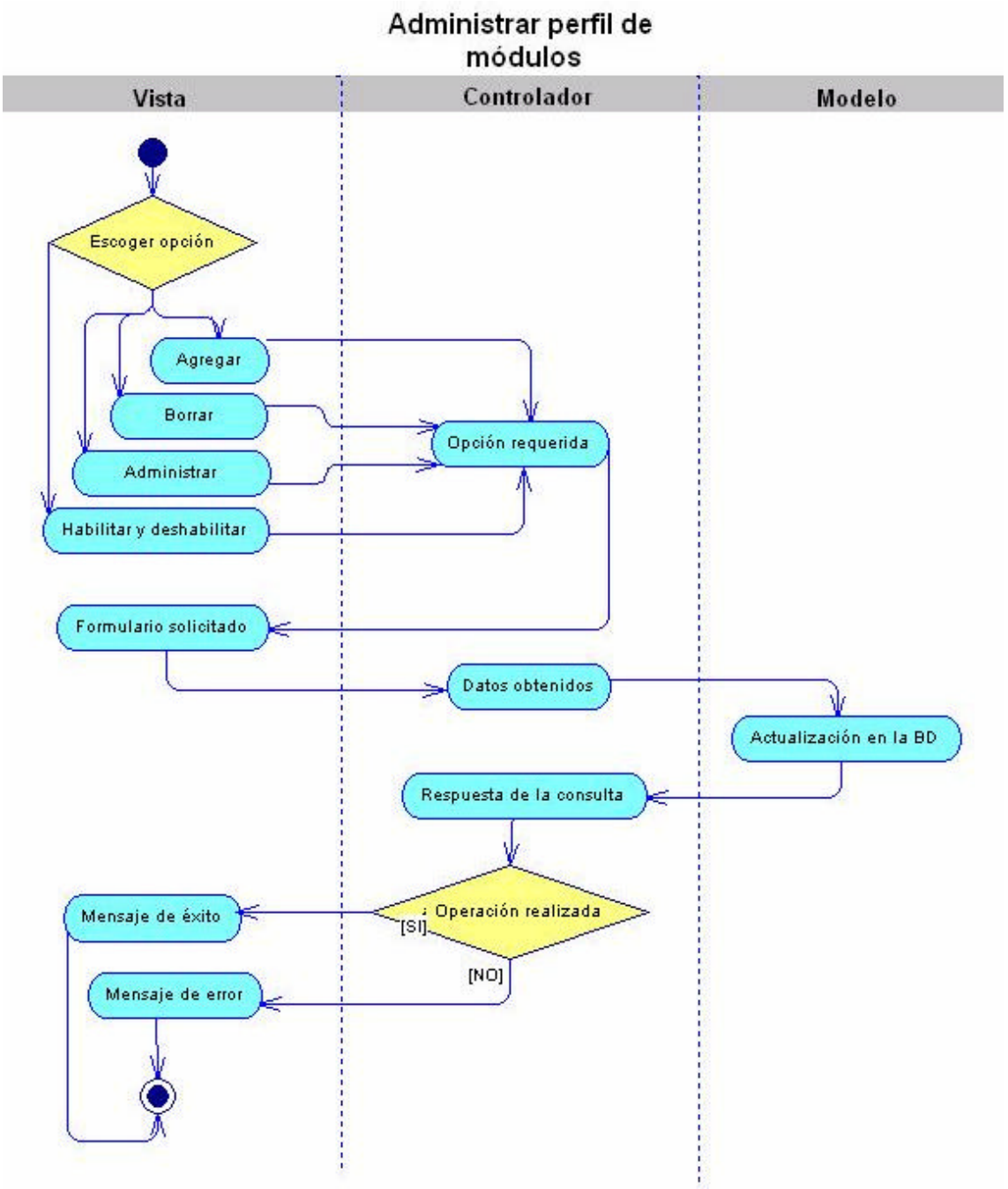

Consultar cursos
Vista
Controlador
Modelo
Opción de cursos seleccionada
Opción requerida
Consultar base de datos
Datos de cursos obtenidos
¿ Existen datos de cursos
[NO]
Mensaje de error
[SI]
Menú de cursos
Opción de tipo de curso solicitado
Mostrar datos de curso solicitado

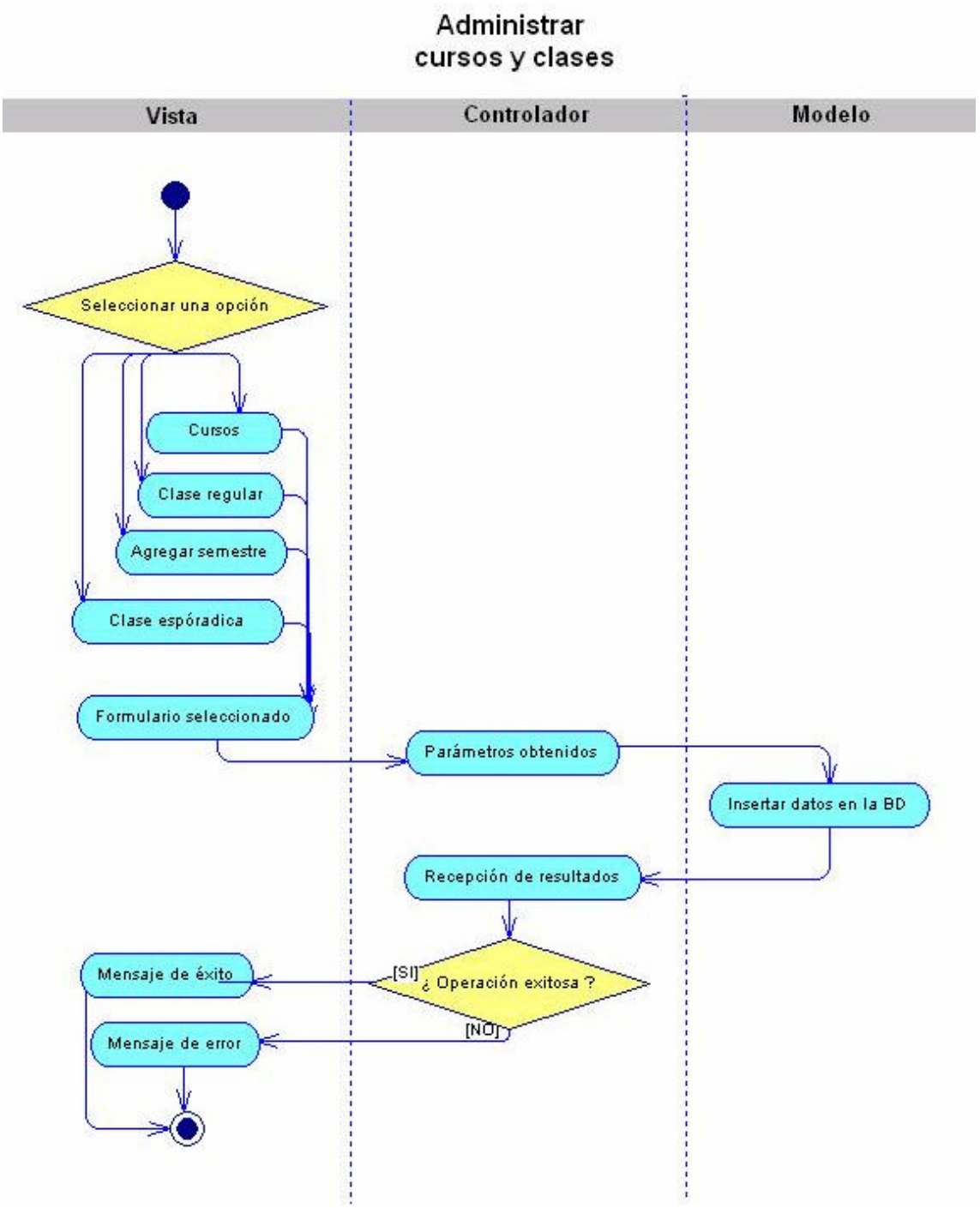

Consultar
Gráficas
Vista
Controlador
Modelo
Opción de consultar gráficas
Opción solicitada
Consultar base de datos
Datos de gráficas
¿ Existen datos ?
[NO]
Mensaje de error
[SI]
Menú de gráficas
Opción de gráfica seleccionada
Datos de la gráfica obtenidos
¿ Existen datos ?
[NO]
[SI]
Muestra gráfica

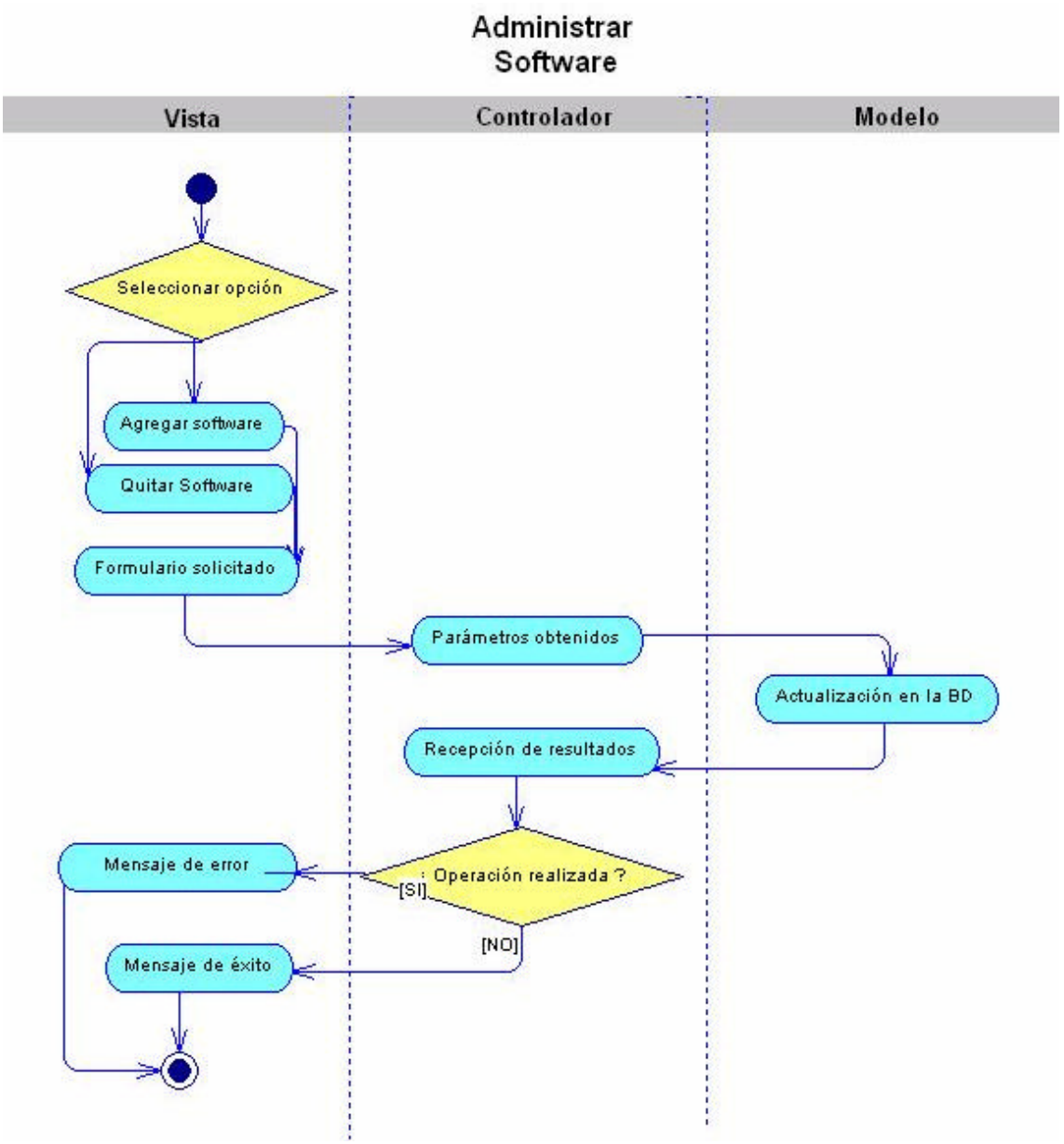
3.3.2. Zona de Administración

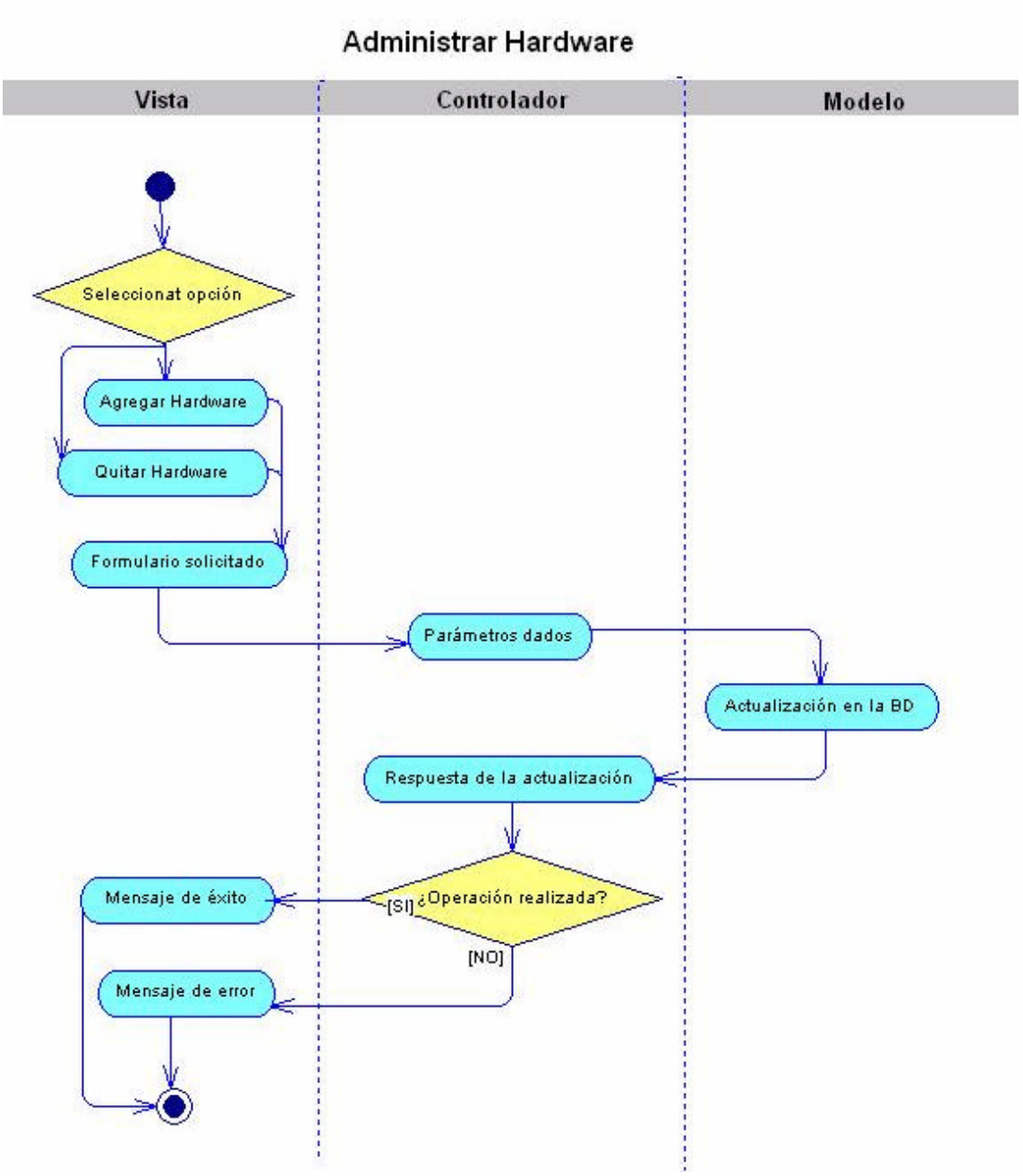



Consulta de usuario
Vista
Controlador
Modelo
Tipo de consulta
Datos de usuario
Cambio de password
Activar o desactivar
Opción recibida
Ingresa login usuario
Dato de usuario
Consulta a la base de datos
Recepción de resultados
¿ Existe usuario ?
[NO]
[SI]
Mensaje de error
Muestra datos de usuario
Recibiendo datos de usuario
Actualización en la BD
Actualización
¿ Operación exitosa ´
[NO]
[SI]
Mensaje de no éxito
Mensaje de éxito


Administrar perfil de módulos
Vista
Controlador
Modelo
Escoger opción
Agregar
Borrar
Administrar
Habilitar y deshabilitar
Opción requerida
Formulario solicitado
Datos obtenidos
Actualización en la BD
Respuesta de la consulta
Operación realizada
[SI]
[NO]
Mensaje de éxito
Mensaje de error


Administrar
cursos y clases
Vista
Controlador
Modelo
Seleccionar una opción
Cursos
Clase regular
Agregar semestre
Clase espóradica
Formulario seleccionado
Parámetros obtenidos
Insertar datos en la BD
Recepción de resultados
¿ Operación exitosa ?
[SI]
[NO]
Mensaje de éxito
Mensaje de error


Administrar Software
Vista
Controlador
Modelo
Seleccionar opción
Agregar software
Quitar Software
Formulario solicitado
Parámetros obtenidos
Actualización en la BD
Recepción de resultados
Operación realizada ?
[SI]
Mensaje de error
[NO]
Mensaje de éxito


Administrar Hardware
Vista
Controlador
Modelo
Seleccionat opción
Agregar Hardware
Quitar Hardware
Formulario solicitado
Parámetros dados
Actualización en la BD
Respuesta de la actualización
¿Operación realizada?
[SI]
[NO]
Mensaje de éxito
Mensaje de error

3.4. Datos

3.4.1. Modelos Conceptuales

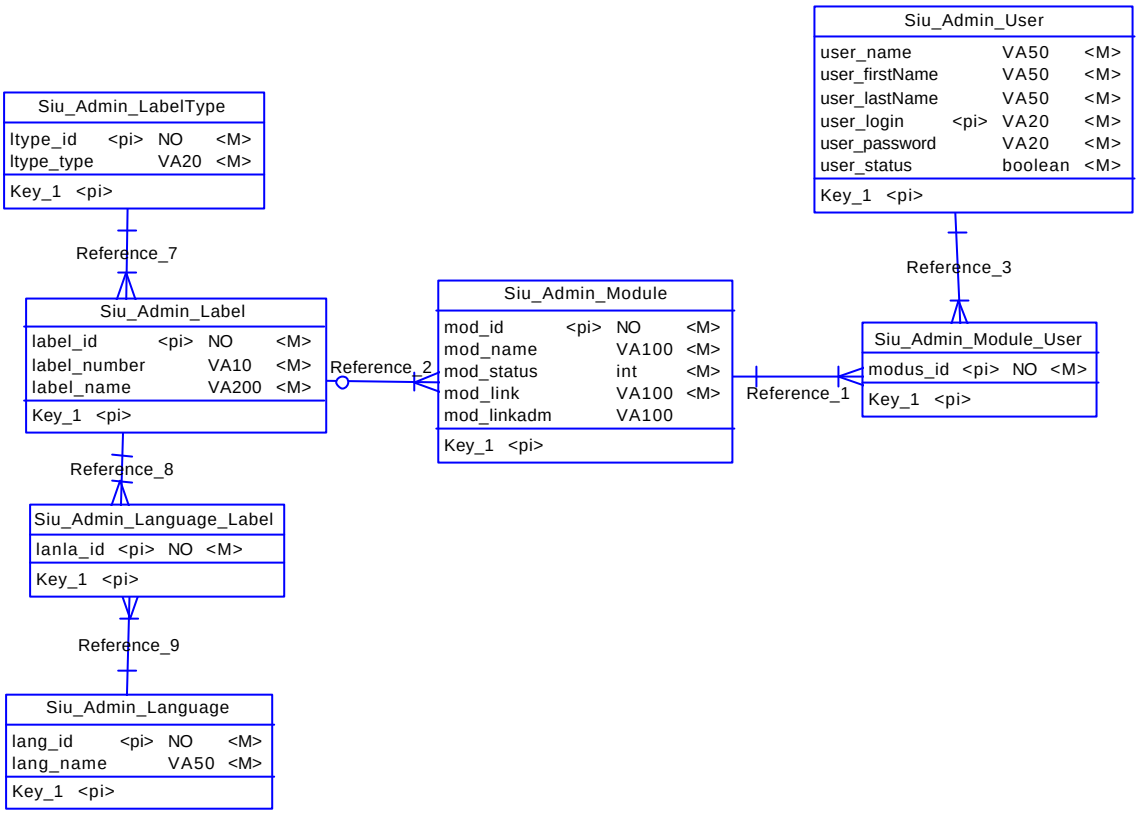


Figura 3.1 Modelo conceptual correspondiente al módulo de Administración del SIU.

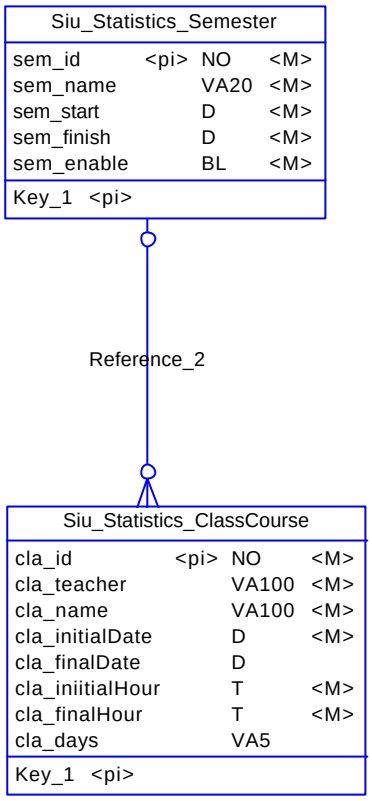


Figura 3.2 Modelo conceptual correspondiente al módulo de Cursos del SIU.

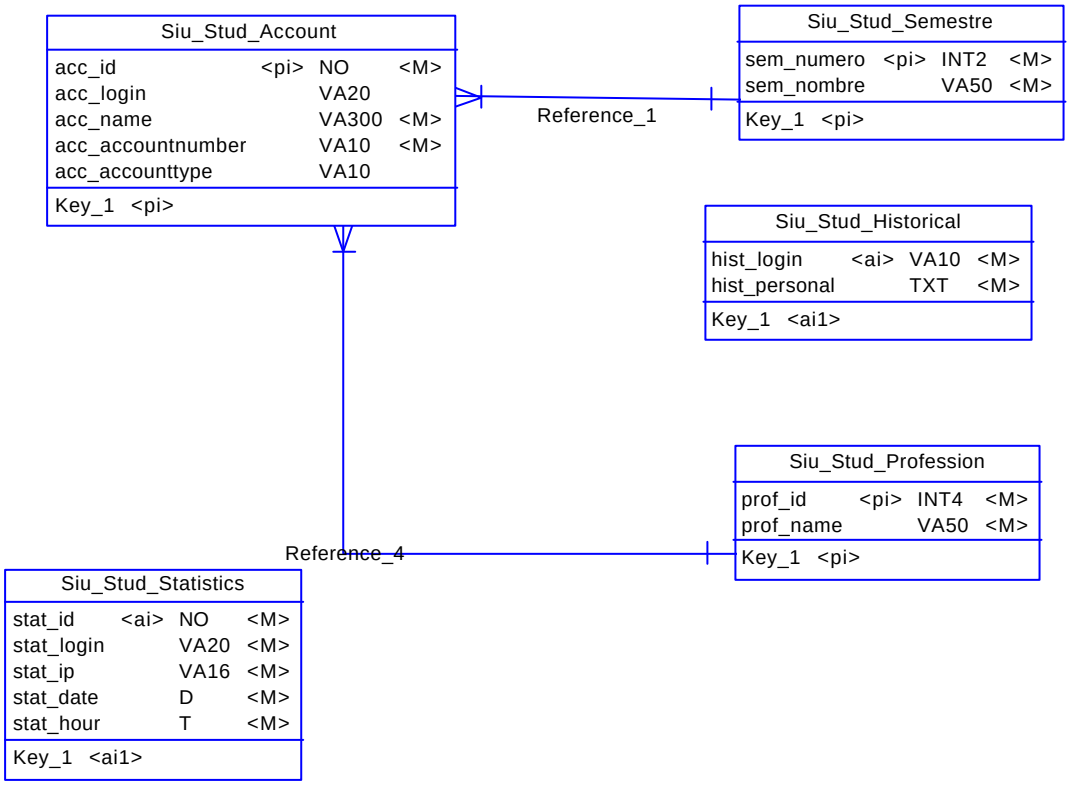


Figura 3.3 Modelo conceptual correspondiente al módulo de Alumnos, Estadísticas y Gráficas del SIU.

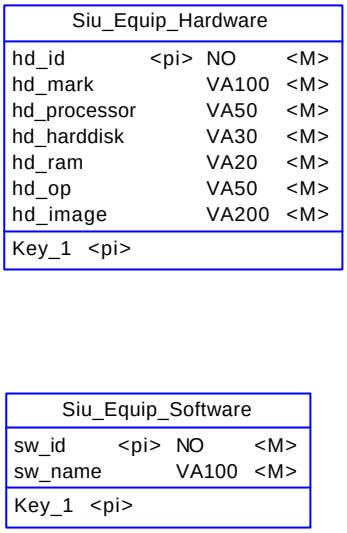


Figura 3.4 Modelo conceptual correspondiente al módulo de Software y Hardware del SIU.

3.4.2. Modelos Físicos

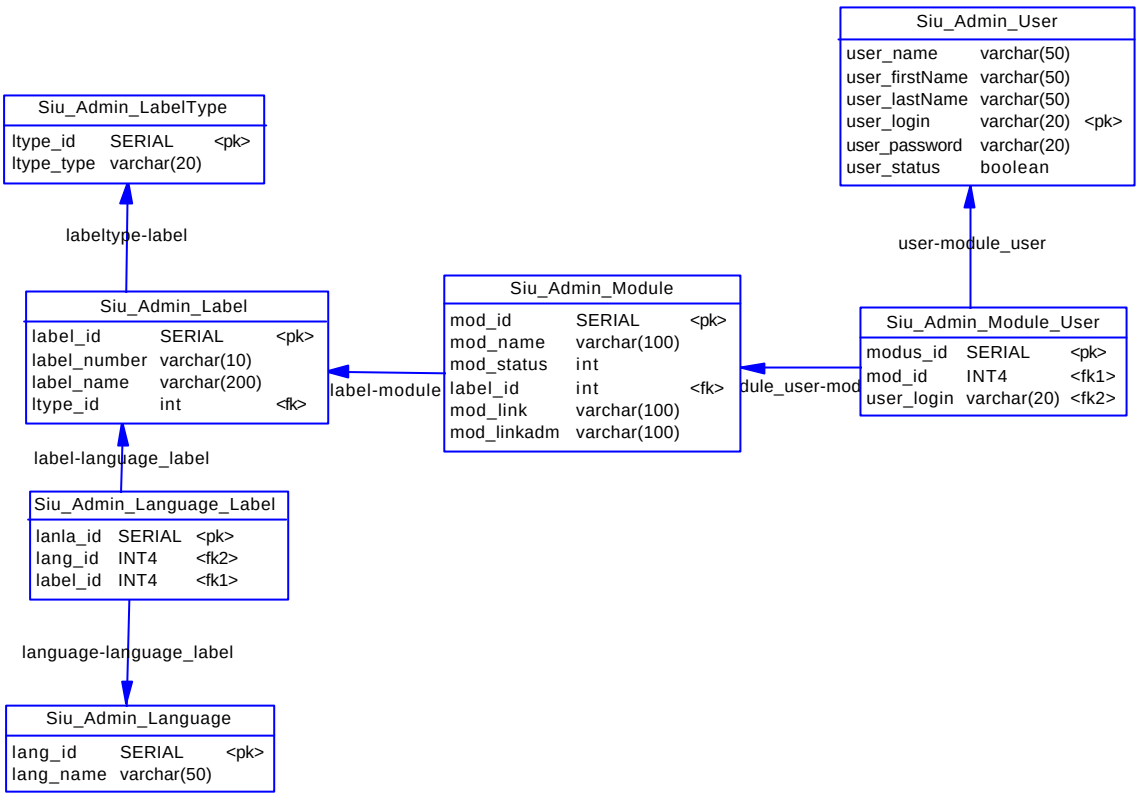


Figura 3.5Modelo físico correspondiente al módulo de Administración del SIU.

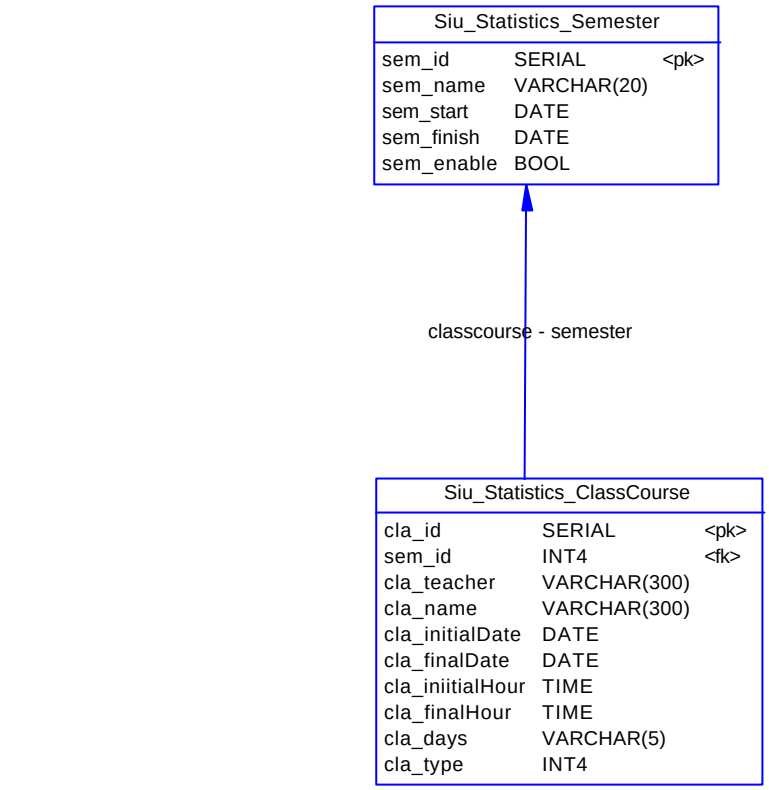


Figura 3.6 Modelo físico correspondiente al módulo de Cursos del SIU.

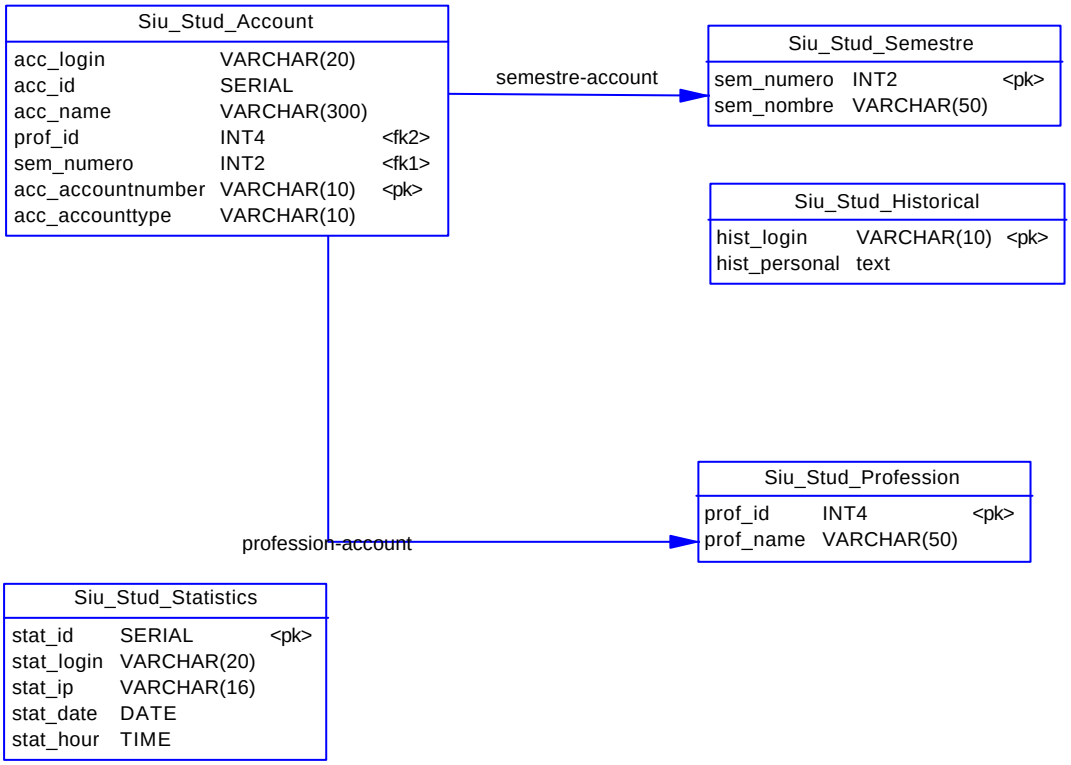


Figura 3.7 Modelo físico correspondiente al módulo de Alumnos, Estadísticas y Gráficas del SIU.

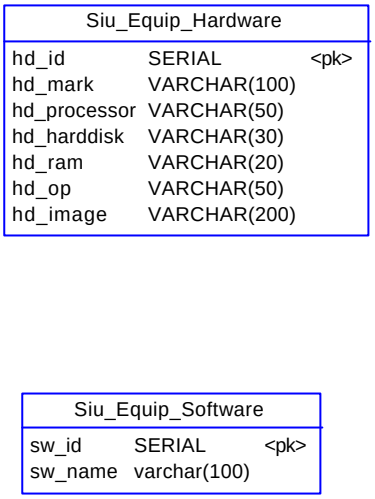


Figura 3.8 Modelo conceptual correspondiente al módulo de Software y Hardware del SIU.

3.5. Diccionario de Datos

3.5.1. Modelos Conceptuales

Nombre	Tipo de Dato	Tamaño
label_id	NO	
label_name	VA200	200
label_number	VA10	10
lang_id	NO	
lang_name	VA50	50
lanla_id	NO	
ltype_id	NO	
ltype_type	VA20	20
mod_id	NO	
mod_link	VA100	100
mod_linkadm	VA100	100
mod_name	VA100	100
mod_status	int	
modus_id	NO	
user_firstName	VA50	50
user_lastName	VA50	50
user_login	VA20	20
user_name	VA50	50
user_password	VA20	20
user_status	boolean	

Tabla 3.1 Diccionario de datos correspondiente al módulo de Administración del SIU.

Nombre	Entidad
ltype_id	Siu_Admin_LabelType
label_id	Siu_Admin_Label
lanla_id	Siu_Admin_Language_Label
lang_id	Siu_Admin_Language
mod_id	Siu_Admin_Module
modus_id	Siu_Admin_Module_User
user_login	Siu_Admin_User

Tabla 3.2 Llaves primarias correspondientes al módulo de Administración del SIU.

Nombre	Tipo de Dato	Tamaño
cla_days	VA5	5
cla_finalDate	D	
cla_finalHour	T	
cla_id	NO	
cla_iniitialHour	T	
cla_initialDate	D	
cla_name	VA100	100
cla_teacher	VA100	100
sem_enable	BL	
sem_finish	D	
sem_id	NO	
sem_name	VA20	20
sem_start	D	

Tabla 3.3 Diccionario de datos correspondiente al módulo de Cursos del SIU.

Nombre	Entidad
sem_id	Siu_Statistics_Semester
cla_id	Siu_Statistics_ClassCourse

Tabla 3.4 Llaves primarias correspondientes al módulo de Cursos del SIU.

Nombre	Tipo de Dato	Tamaño
acc_accountnumber	VA10	10
acc_accounttype	VA10	10
acc_id	NO	
acc_login	VA20	20
acc_name	VA300	300
hist_login	VA10	10
hist_personal	TXT	
prof_id	INT4	4
prof_name	VA50	50
sem_nombre	VA50	50
sem_numero	INT2	2
stat_date	D	
stat_hour	T	
stat_id	NO	
stat_ip	VA16	16
stat_login	VA20	20

Tabla 3.5 Diccionario de datos correspondiente al módulo de Alumnos, Estadísticas y Gráficas del SIU.

Nombre	Entidad
acc_accountnumber	Siu_Stud_Account
stat_id	Siu_Stud_Statistics
sem_numero	Siu_Stud_Semestre
prof_id	Siu_Stud_Profession
hist_login	Siu_Stud_Historical

Tabla 3.6 Llaves primarias correspondientes al módulo de Alumnos, Estadísticas y Gráficas del SIU.

Nombre	Tipo de Dato	Tamaño
hd_harddisk	VA30	30
hd_id	NO	
hd_image	VA200	200
hd_mark	VA100	100
hd_op	VA50	50
hd_processor	VA50	50
hd_ram	VA20	20
sw_id	NO	
sw_name	VA100	100

Tabla 3.7 Diccionario de datos correspondiente al módulo de Software y Hardware del SIU.

Nombre	Entidad
hd_id	Siu_Equip_Hardware
sw_id	Siu_Equip_Software

Tabla 3.8 Llaves primarias correspondientes al módulo de Software y Hardware del SIU.

3.6. Descripción de las tablas del módulo de Administration.

Siu_admin_label : Contiene la lista de las etiquetas disponibles en cada página del sistema. La primera columna *label_id* indica el número de etiqueta, la segunda *label_number* es el número con el que se va a identificar la etiqueta, la tercera *label_name* indica el texto de la etiqueta, tanto en idioma inglés como en español, la tercera *ltype_id* identifica el tipo de etiqueta que es.

Siu_admin_labeltype: Contiene cuál es el tipo de etiqueta. La columna *ltype_id* los tres tipos de etiquetas que pueden ser, la columna *ltype_type* indica si la etiqueta es de tipo titulo, etiqueta simple o liga.

Siu_admin_language: Contiene que idiomas se manejan en el sistema. La primera columna *land_id* especifica los idiomas que se encuentran registrados, la segunda columna *lang_name* indica el nombre del idioma.

Siu_admin_language_label: Contiene la relación que indica a qué idioma se encuentra relacionada una etiqueta. La primera columna *lang_id* indica, de acuerdo con la tabla Siu_admin_language el idioma al que pertenece; la segunda columna *label_id* indica el identificador de la etiqueta.

Siu_admin_module: Contiene la lista de los módulos que se tienen disponibles para el siu. La primera columna *mod_id* indica el número de módulo, la segunda *mod_name* indica el nombre del módulo y la tercera *mod_status* indica si el módulo esta habilitado o no.

Siu_admin_module_user: Contiene la información del usuario y a qué módulos tiene acceso. La primera columna *mod_id* indica el número del módulo, la segunda columna user_login indica el login del usuario que tiene acceso a el modulo indicado.

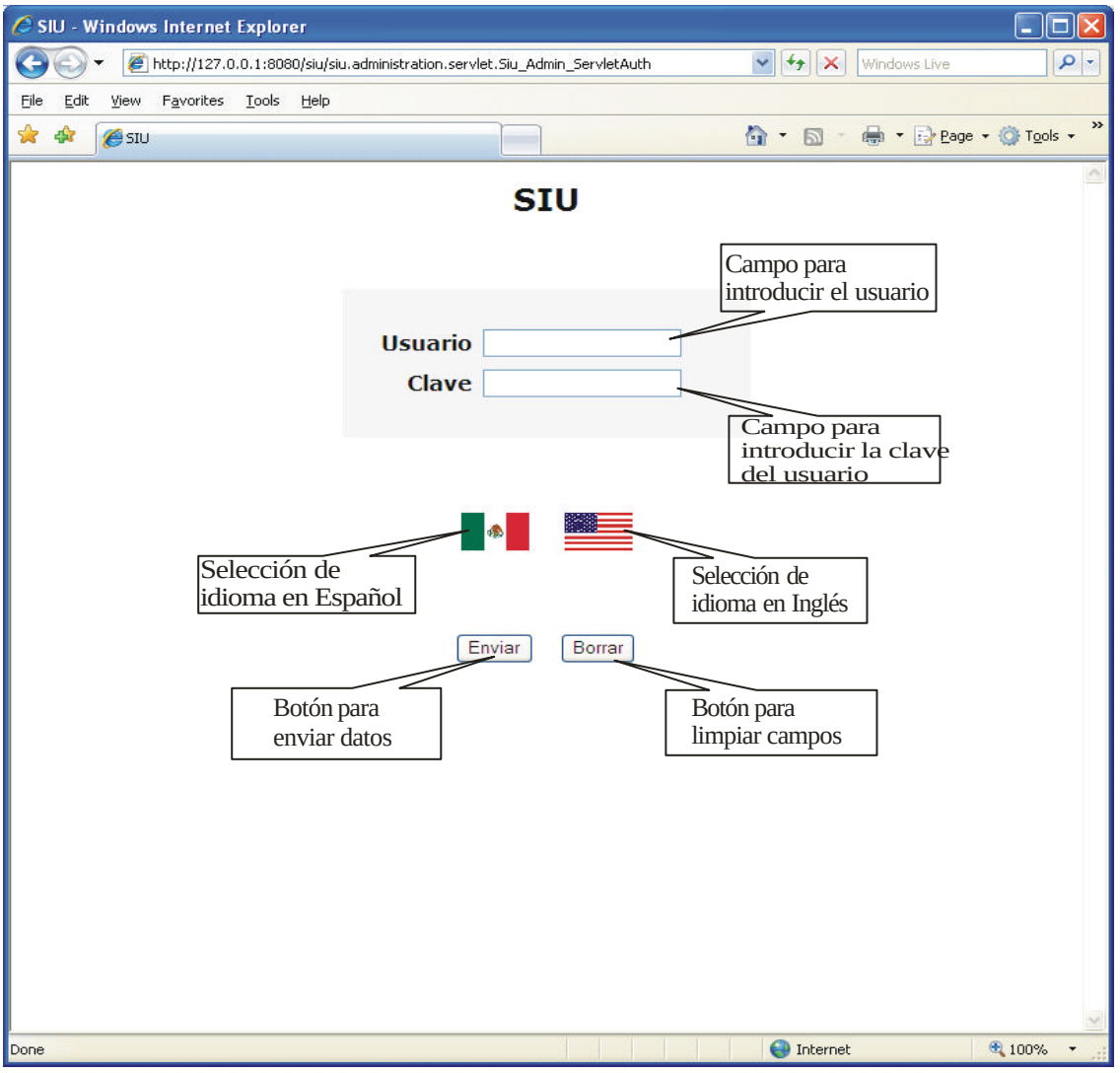
3.7. Secuencia de Pantallas

3.7.1. Zona de Vistas

La parte de Zona de Vistas del SIU sirve para ver los módulos que se encuentran dados de alta y habilitados.

Nombre: Siu_Admin_JspAuth.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Admin_JspMainAuth.jsp
Descripción: Página que permite el ingreso de los consultores autorizados a la Zona de Vistas.

Pantalla:

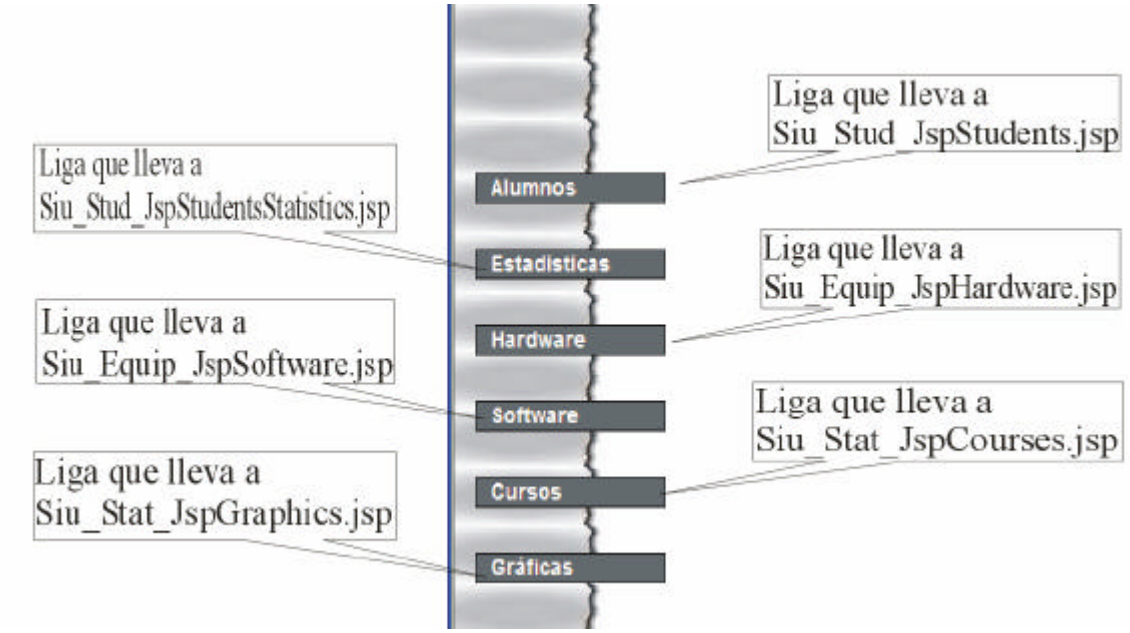


Campos:

Campo	Representación
login	Text Field
pass	Text Field
Enviar	Botón
Borrar	Botón
Imagen (mexico.jpg)	Liga
Imagen (eeuu.gif)	Liga

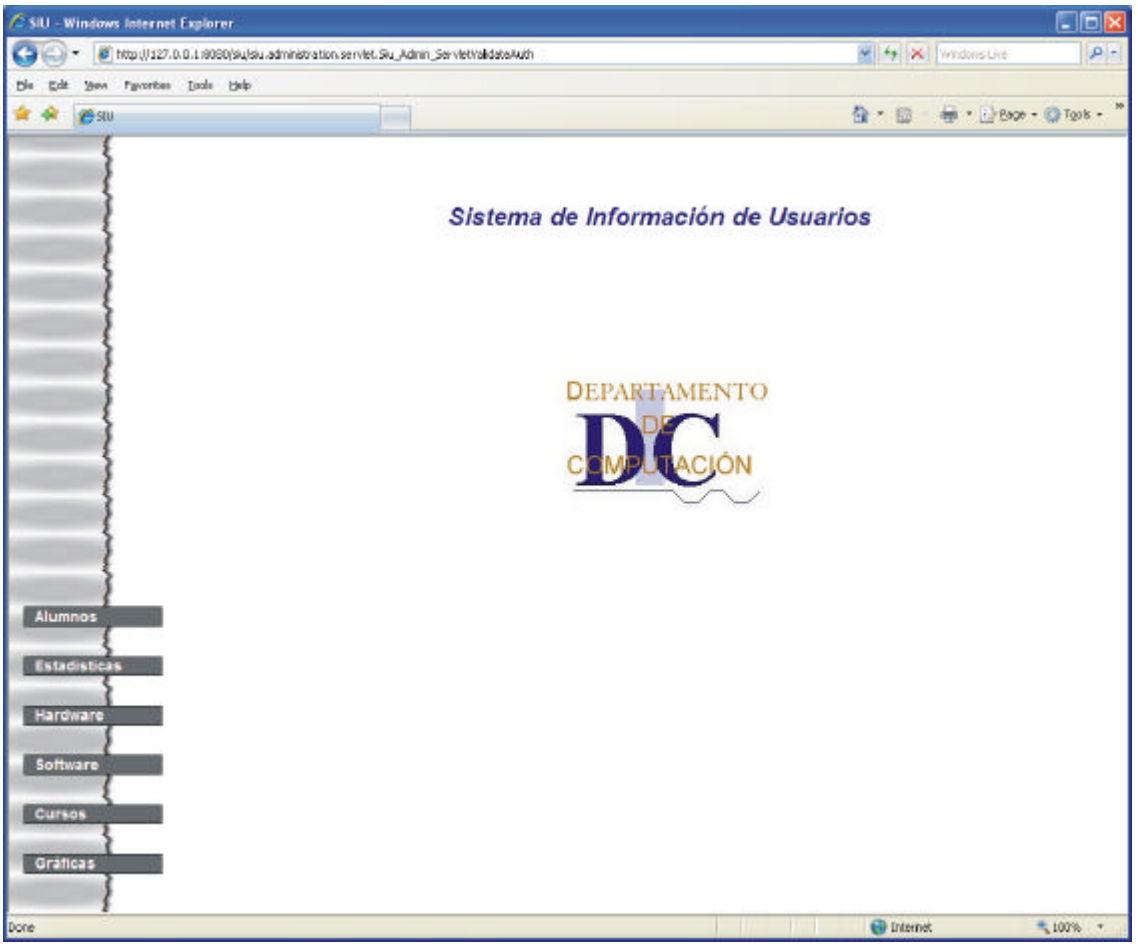
Nombre: Siu_Admin_JspIndexAuth.jsp
Página Anterior: Ninguna
Página Siguiente: Ninguna
Descripción: Página que contiene los enlaces para cada uno de los módulos del SIU.

Pantalla:



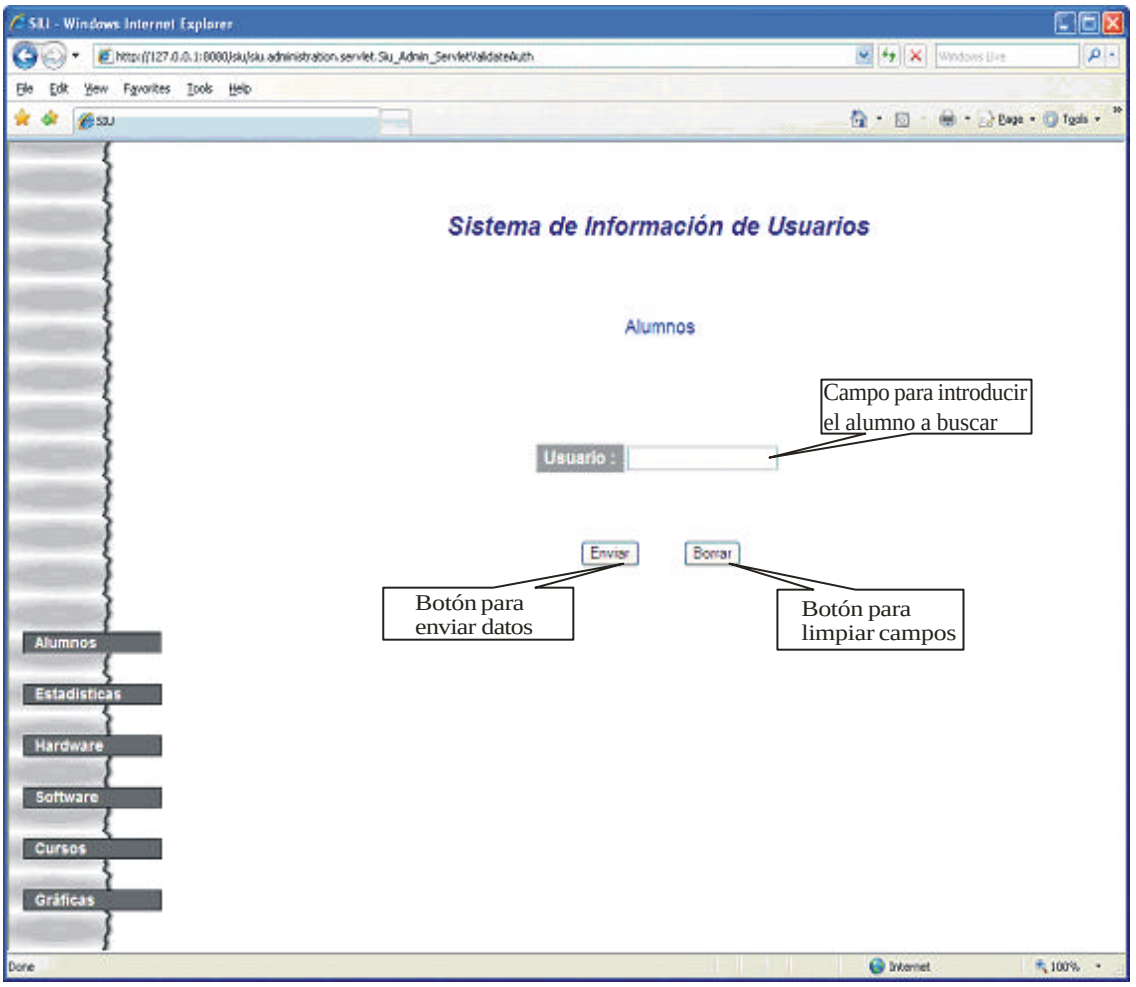
Nombre: Siu_Admin_JspPresentation.jsp
Página Anterior: Ninguna
Página Siguiente: Ninguna
Descripción: Página de inicio de la Zona de Vistas del SIU.

Pantalla:



Nombre: Siu_Admin_JspStudents.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Stud_JspGetStudents.jsp
Descripción: Página que sirve para buscar un usuario y encontrar su historial.

Pantalla:

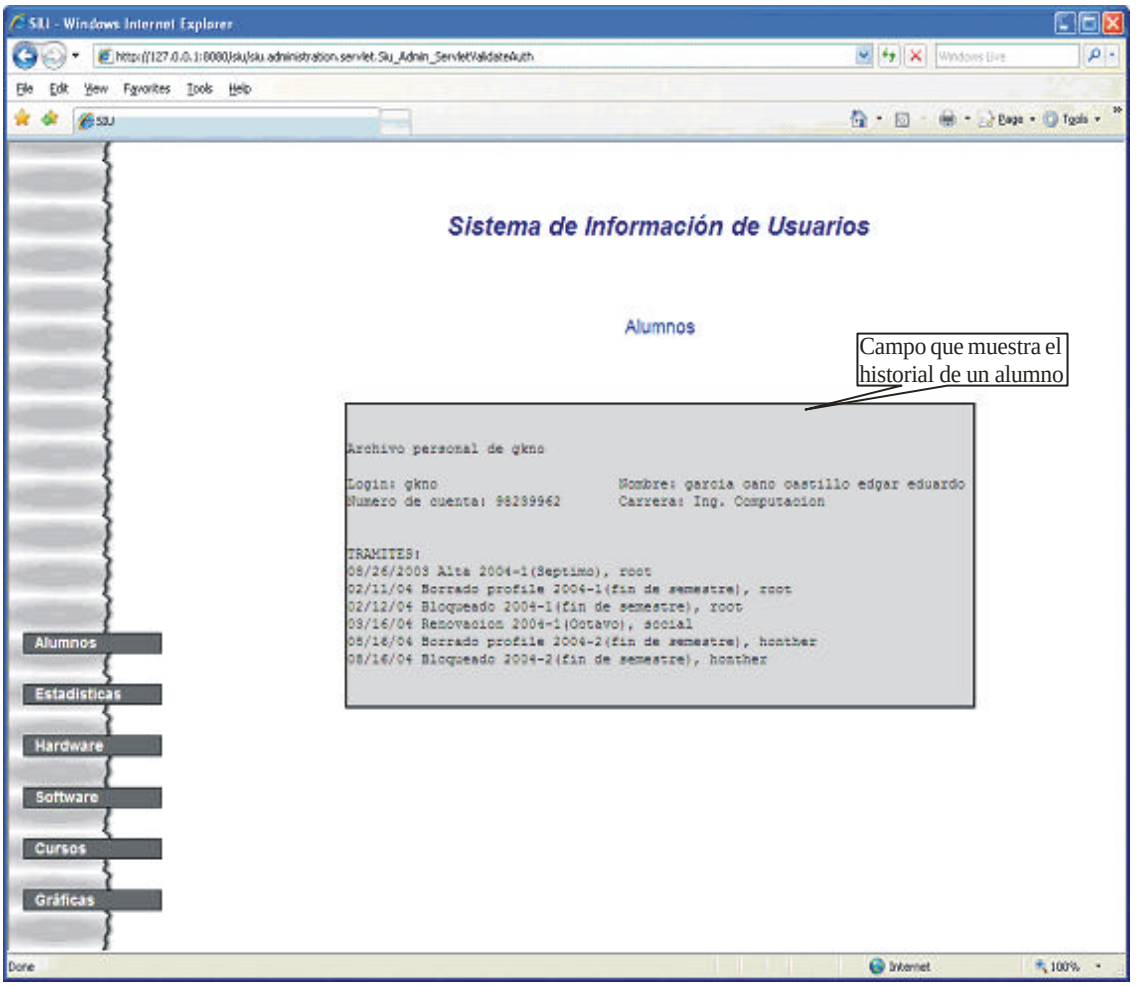


Campos:

Campo	Representación
login	Text Field
Enviar	Botón
Borrar	Botón

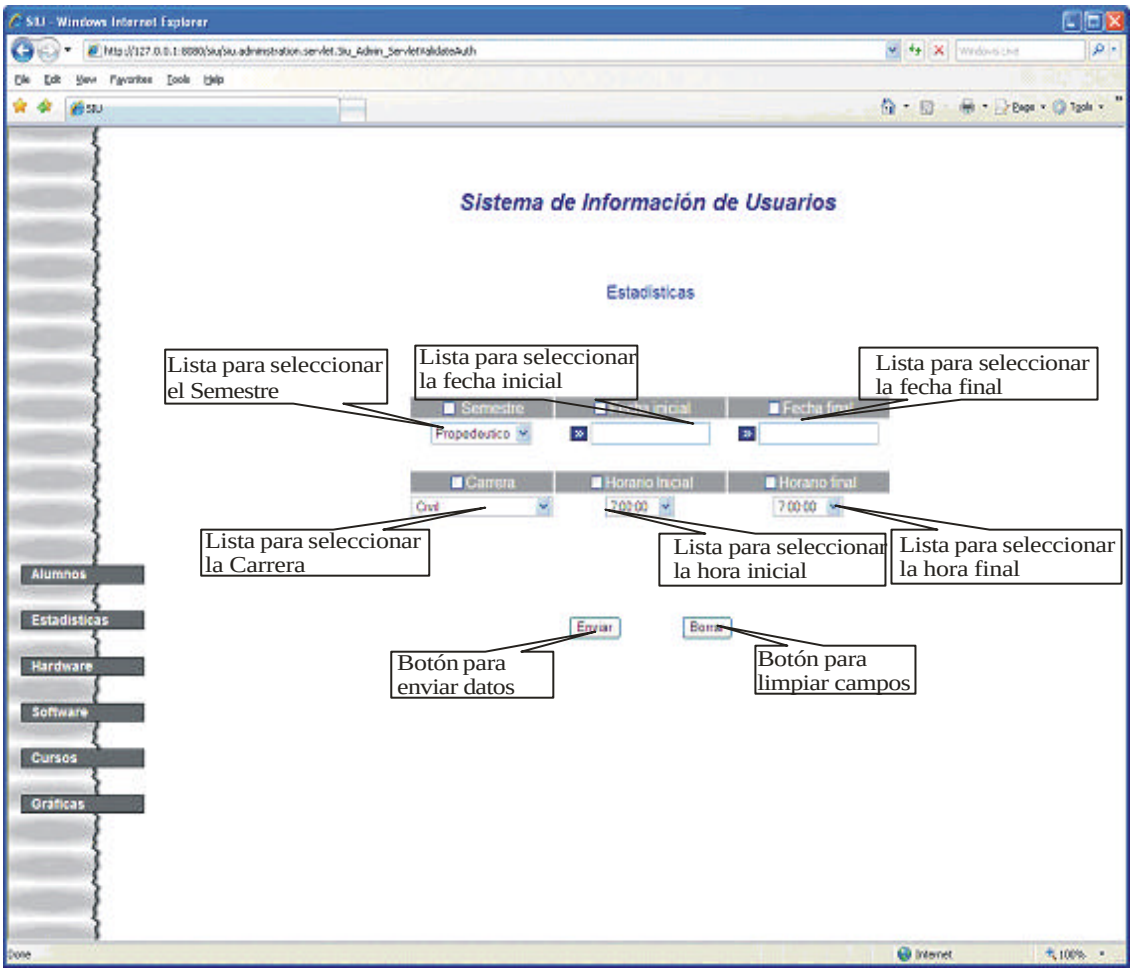
Nombre: Siu_Stud_JspGetStudents.jsp
Página Anterior: Ninguna
Página Siguiente: Ninguna
Descripción: Página que muestra el historial del usuario buscado.

Pantalla:



Nombre: Siu_Admin_JspStudentsStatistics.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Stud_JspGetStudentsStatistics.jsp
Descripción: Página que sirve para consultar estadísticas de los usuarios de acuerdo con las opciones seleccionadas.

Pantalla:



Campos:

Campo	Representación
menu_semestre	List Menu
txtFech	Text Field
txtFech2	Text Field
menu_carrera	List Menu
menu_horaini	List Menu
menu_horafin	List Menu
Enviar	Botón
Borrar	Botón

Nombre: Siu_Stud_JspGetStudentsStatistics.jsp (forma A)
Página Anterior: Siu_Admin_JspStudentsStatistics.jsp
Página Siguiente: Ninguna
Descripción: Página que sirve para mostrar los datos estadísticos seleccionando los campos de Semestre y/o Carrera solamente.

Pantalla:

Campos:

Campo	Representación
Genera Reporte	Liga

Nombre: Siu_Stud_JspGetStudentsStatistics.jsp (forma B)
Página Anterior: Siu_Admin_JspStudentsStatistics.jsp
Página Siguiente: Ninguna
Descripción: Página que sirve para mostrar los datos estadísticos haciendo selección de todos los campos de la página anterior.

Pantalla:

Campos:

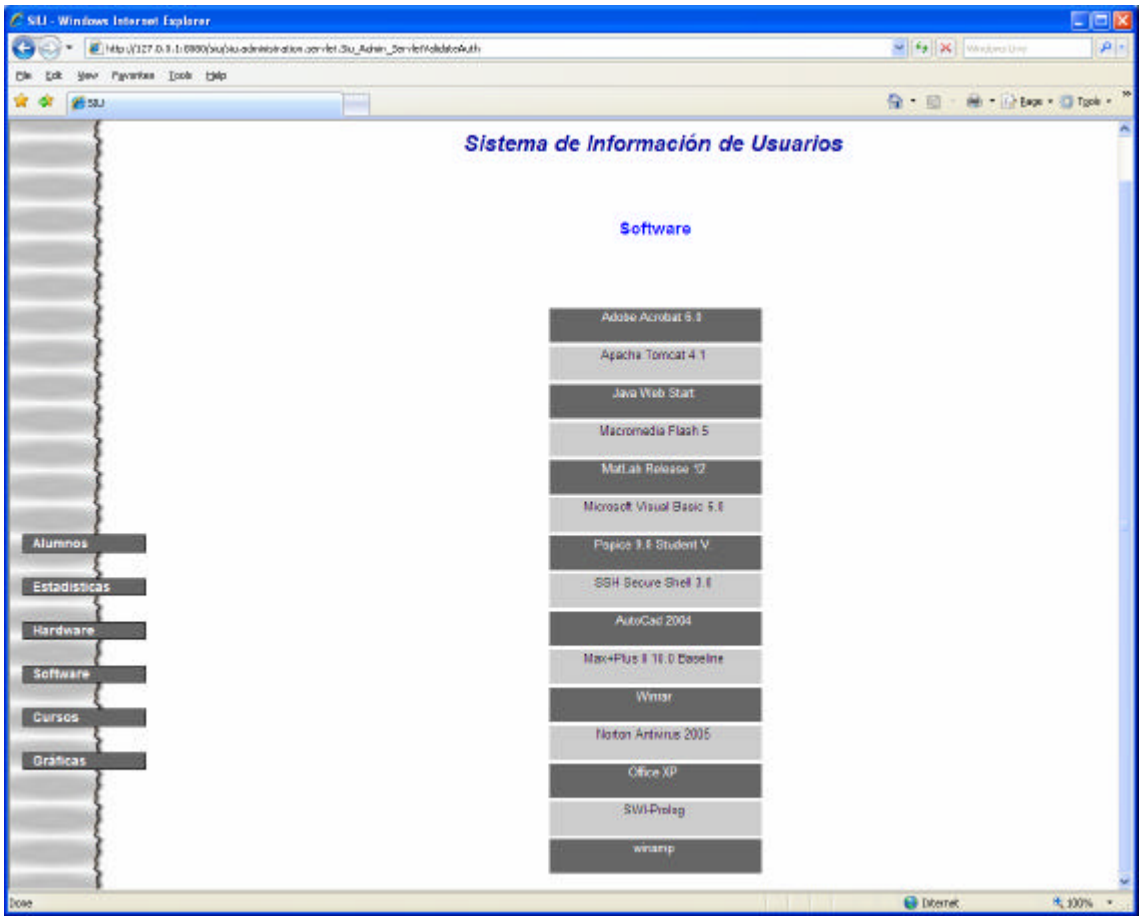
Campo	Representación
Genera Reporte	Liga

Nombre: Siu_Stud_JspHardware.jsp
Página Anterior: Ninguna
Página Siguiente: Ninguna
Descripción: Página que sirve para mostrar la información de hardware del Laboratorio de Computación.

Pantalla:

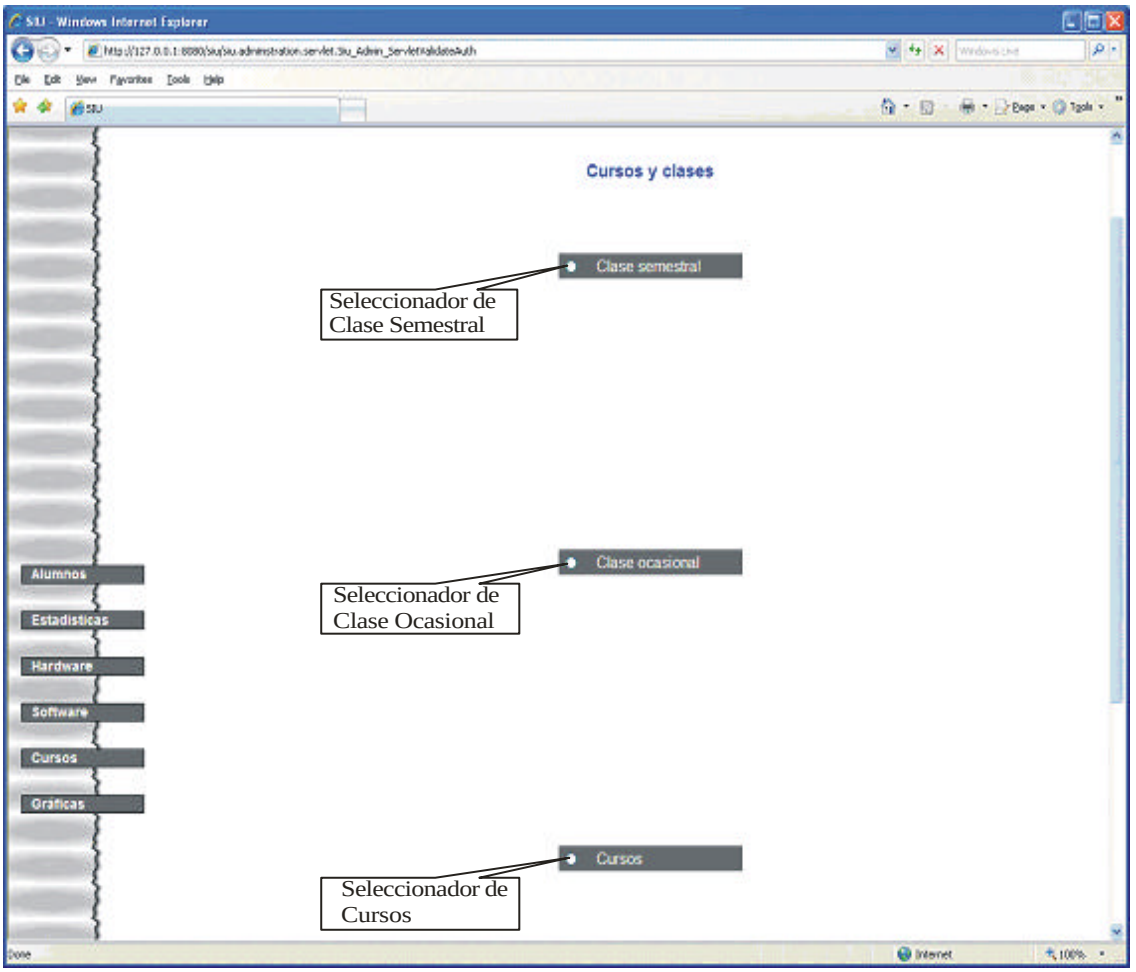
Nombre: Siu_Stud_JspSoftware.jsp
Página Anterior: Ninguna
Página Siguiente: Ninguna
Descripción: Página que sirve para mostrar la información de software del Laboratorio de Computación.

Pantalla:



Nombre: Siu_Stat_JspCourses.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Stat_JspShowCourses.jsp
Descripción: Página que sirve para seleccionar el tipo de información sobre cursos y clases impartidos en el Laboratorio de Computación.

Pantalla:



Campos:

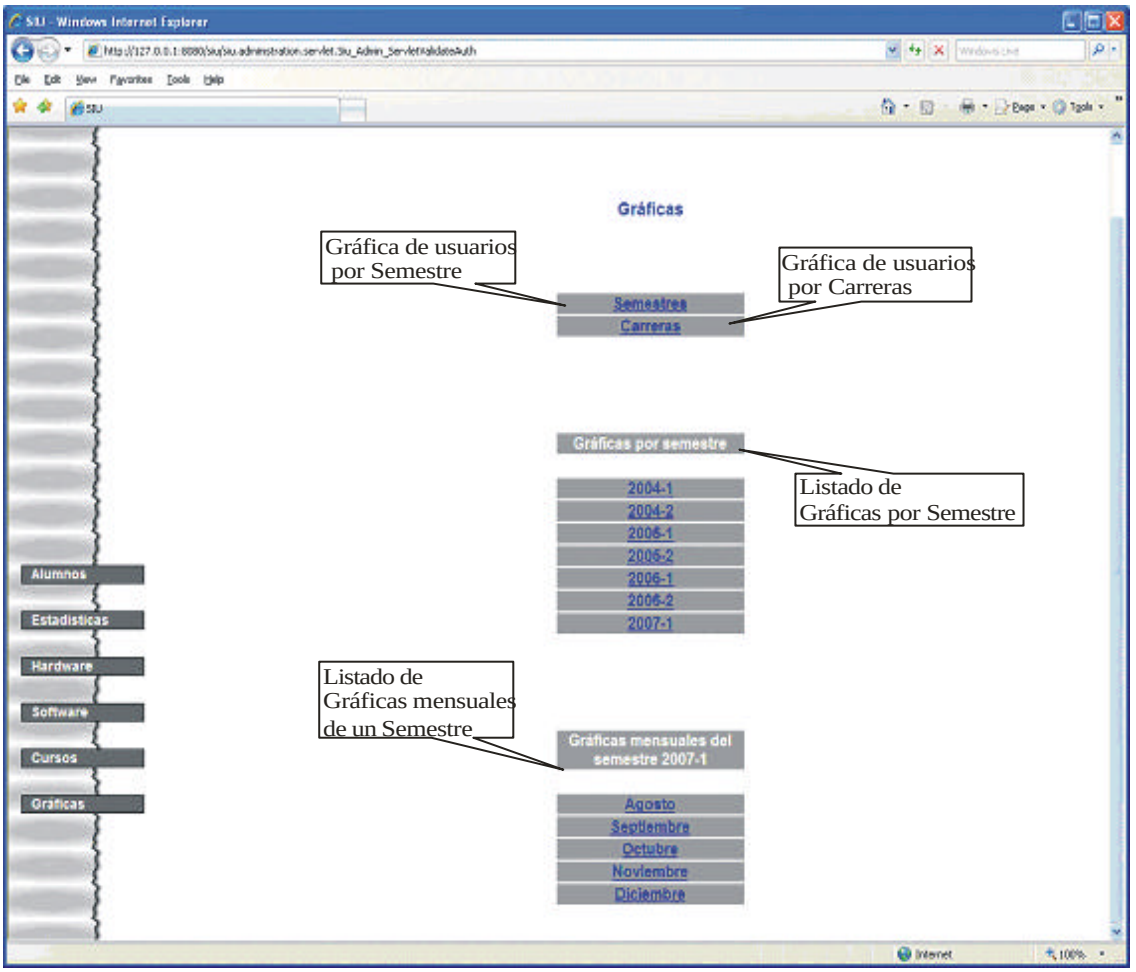
Campo	Representación
RadioGroup	radio

Nombre: Siu_Stat_JspShowCourses.jsp
Página Anterior: Siu_Stat_JspCourses.jsp
Página Siguiente: Ninguna
Descripción: Página que sirve para mostrar la información sobre cursos y clases impartidos en el Laboratorio de Computación dependiendo de la opción seleccionada en la página anterior.

Pantalla:

Nombre: Siu_Stat_JspGraphics.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Stat_JspShowGraphics.jsp
Descripción: Página que sirve para mostrar el menú de gráficas que puede generar el SIU.

Pantalla:



Nombre: Siu_Stat_JspShowGraphics.jsp
Página Anterior: Siu_Stat_JspGraphics.jsp
Página Siguiente: Ninguna
Descripción: Página que sirve para mostrar la gráfica seleccionada según el menú de gráficas de la página anterior.

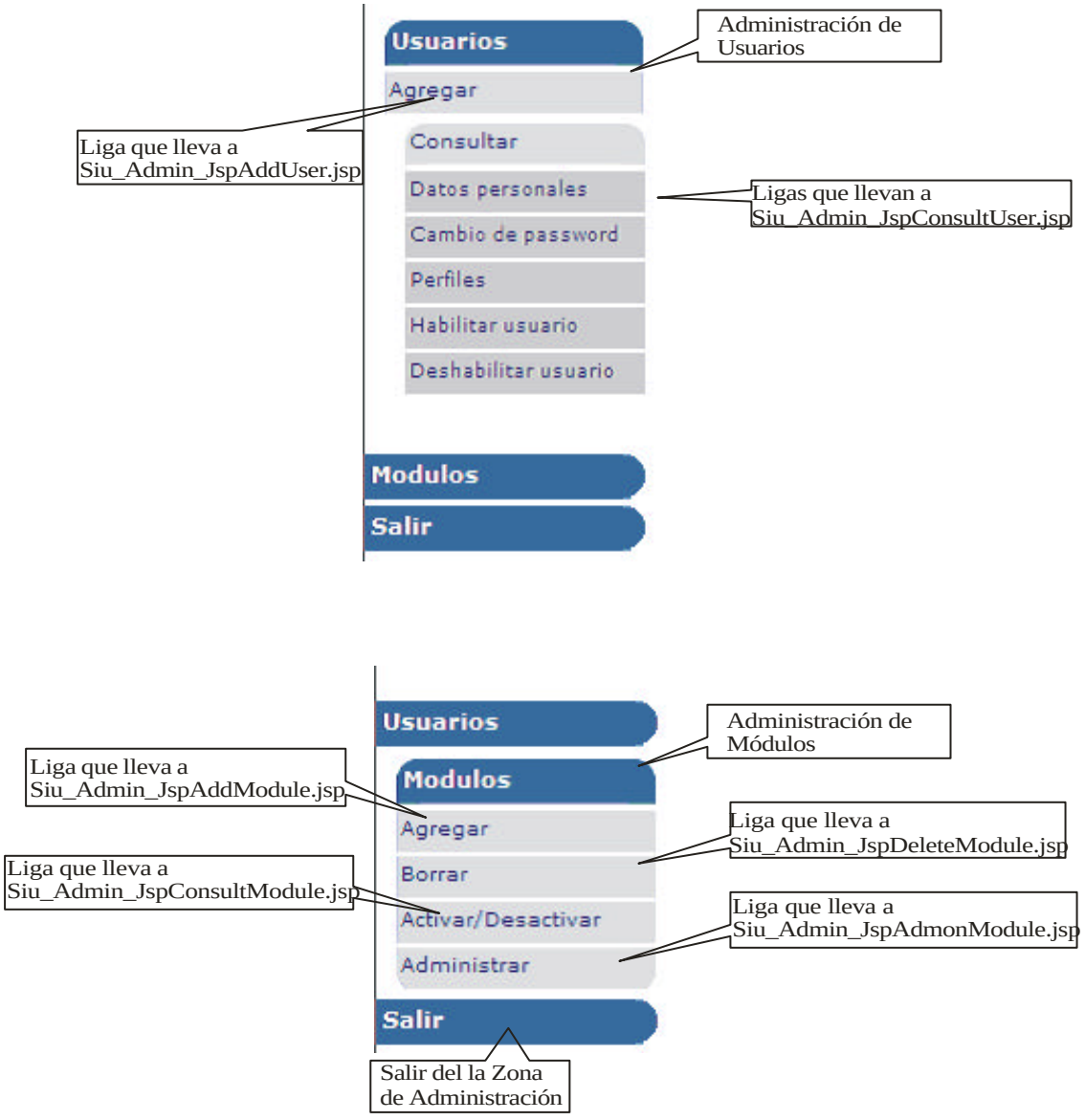
Pantalla:

3.7.2. Zona de Administración

La parte de Zona de Administración sirve para dar mantenimiento a la zona de vistas del SIU.

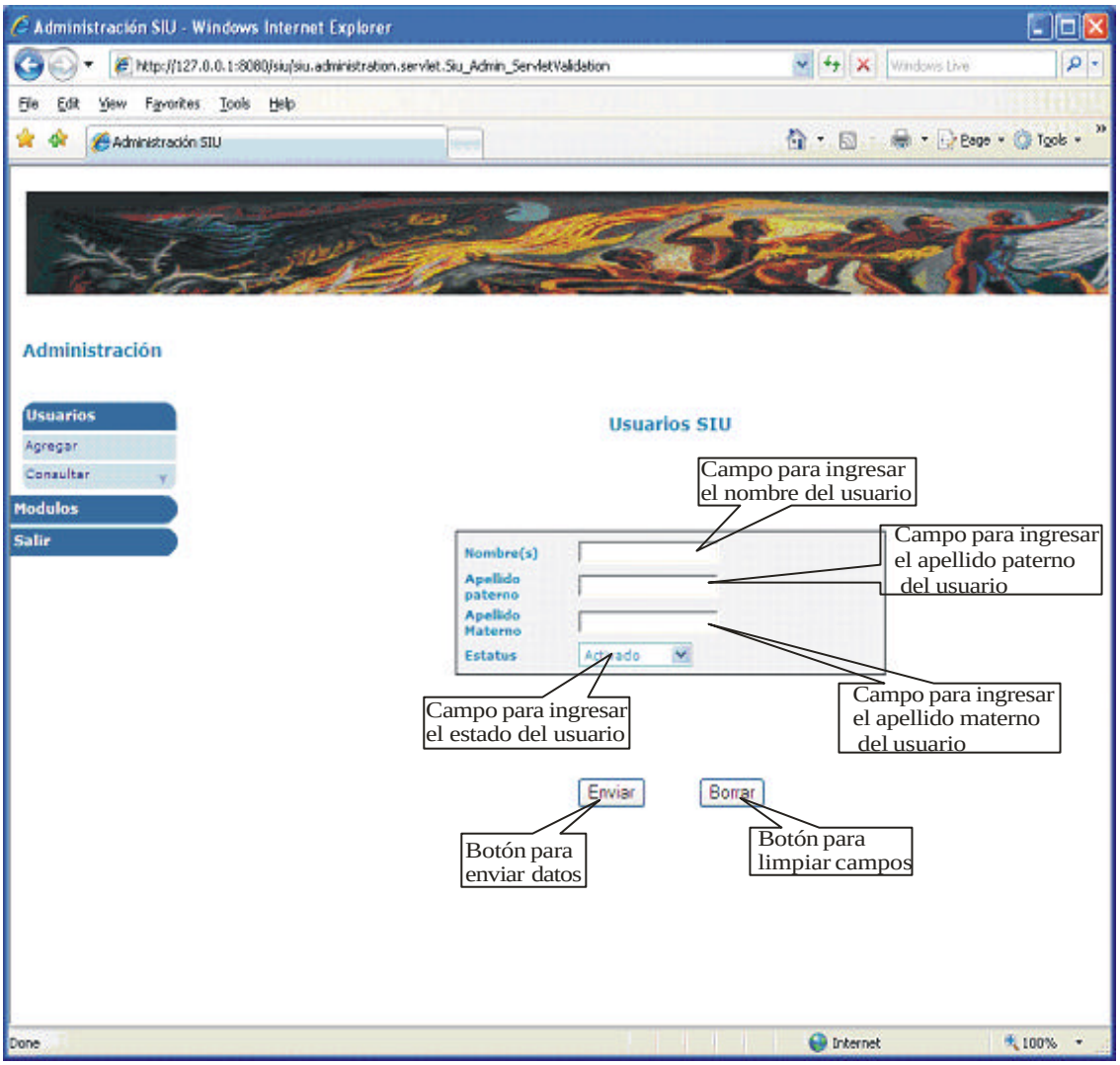
Nombre: Siu_Admin_JspIndex.jsp
Página Anterior: Ninguna
Página Siguiente: Ninguna
Descripción: Página que contiene los enlaces para cada uno de los módulos las zonas de administración del SIU.

Pantalla:



Nombre: Siu_Admin_JspAddUser.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Admin_JspUpdateUser.jsp
Descripción: Página para dar de alta a los usuarios que van a manejar el sistema.

Pantalla:

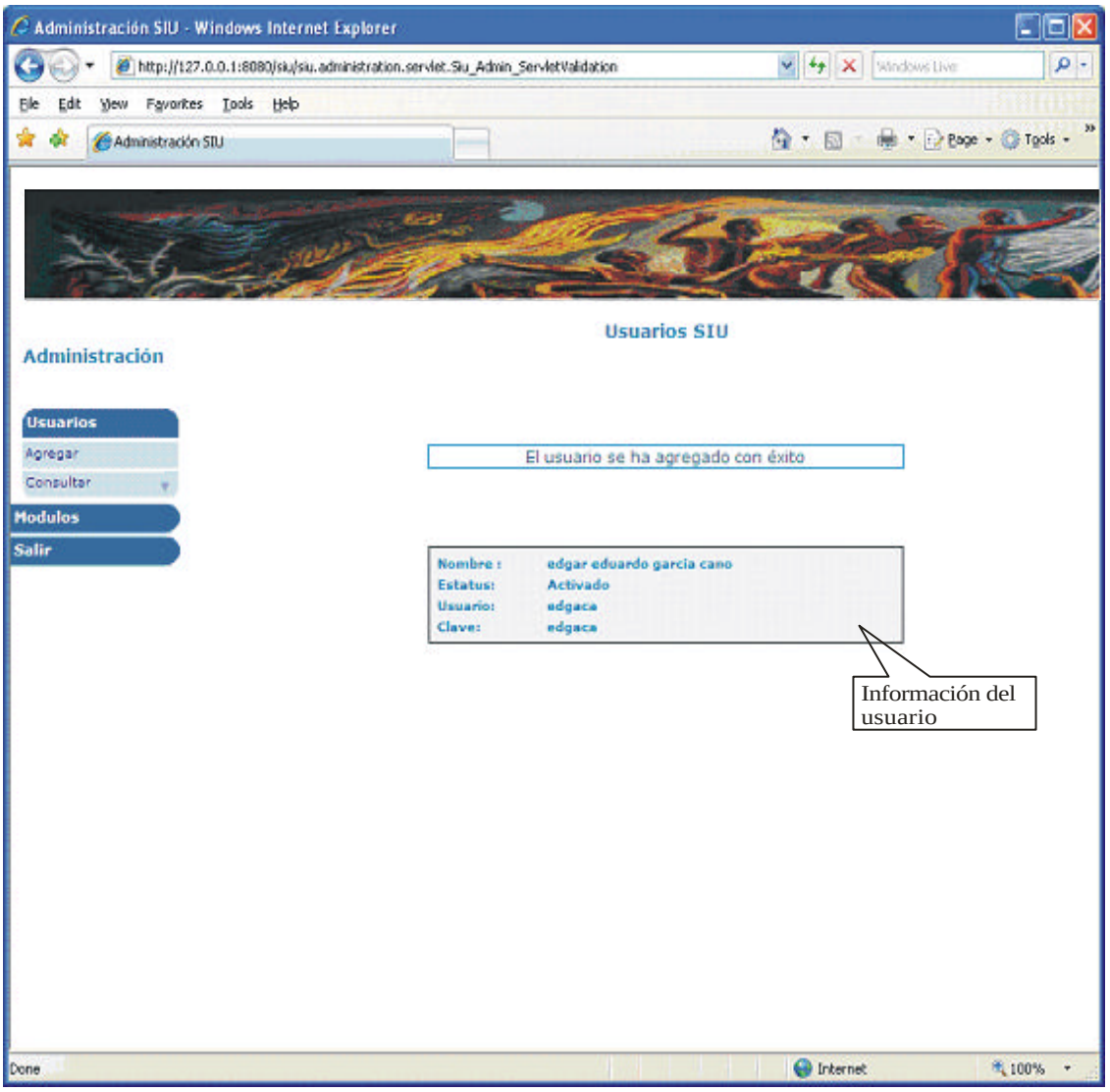


Campos:

Campo	Representación
user_name	Text Field
user_firstName	Text Field
user_lastName	Text Field
user_status	List Menu
Enviar	Botón
Borrar	Botón

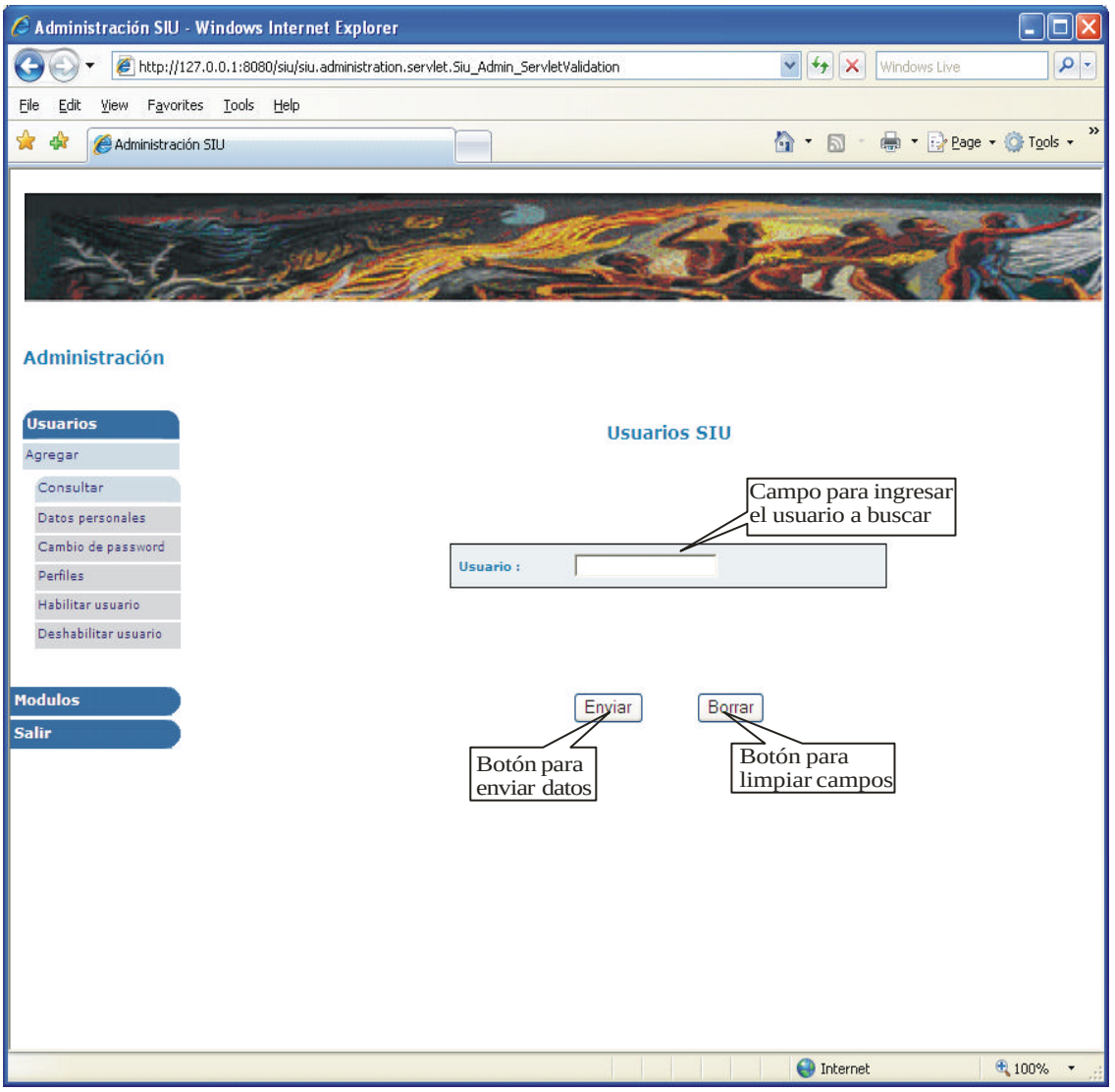
Nombre: Siu_Admin_JspUpdateUser.jsp
Página Anterior: Siu_Admin_JspAddUser.jsp
Página Siguiente: Ninguna
Descripción: Página que muestra la información del usuario dado de alta.

Pantalla:



Nombre: Siu_Admin_JspConsultUser.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Admin_JspPassUpdateUser.jsp (opción 1.Datos Personales)
Siu_Admin_JspPassUser.jsp (opción 2.Cambio de Password)
Siu_Admin_JspProfileUser.jsp (opción 3.Perfiles)
Siu_Admin_JspPassUpdateUser.jsp (opción 4. Habilitar usuario)
Siu_Admin_JspPassUpdateUser.jsp (opción 5. Deshabilitar usuario)
Descripción: Página para solicitar la información de un usuario dado de alta.

Pantalla:

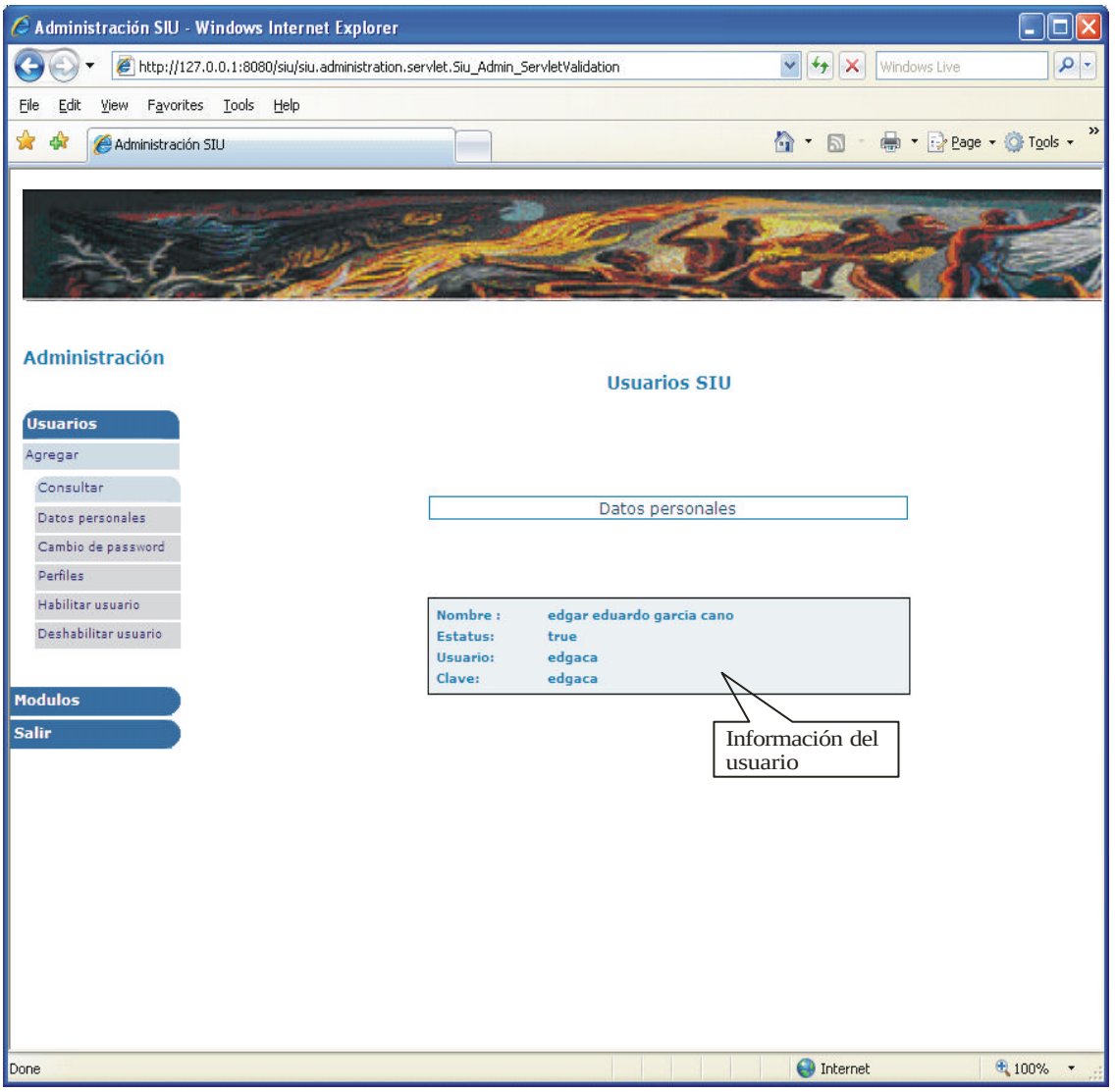


Campos:

Campo	Representación
user_login	Text Field

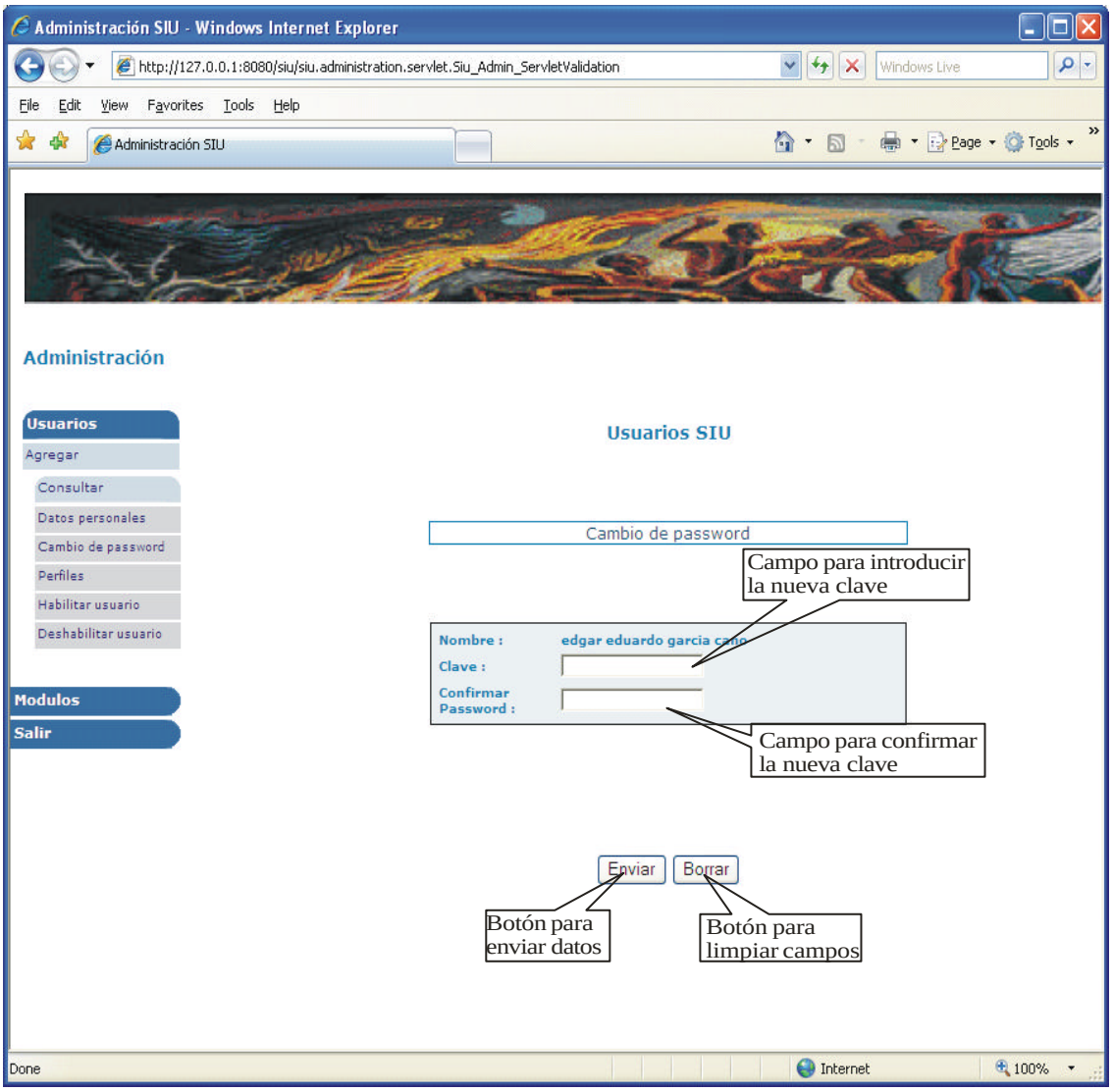
Nombre: Siu_Admin_JspPassUpdateUser.jsp
Página Anterior: Siu_Admin_JspConsultUser.jsp
Página Siguiente: Ninguna
Descripción: Página que muestra la información de un usuario solicitado.

Pantalla:



Nombre: Siu_Admin_JspPassUser.jsp
Página Anterior: Siu_Admin_JspConsultUser.jsp
Página Siguiente: Siu_Admin_JspPassUpdateUser.jsp
Descripción: Página para cambiar el password de un usuario solicitado.

Pantalla:

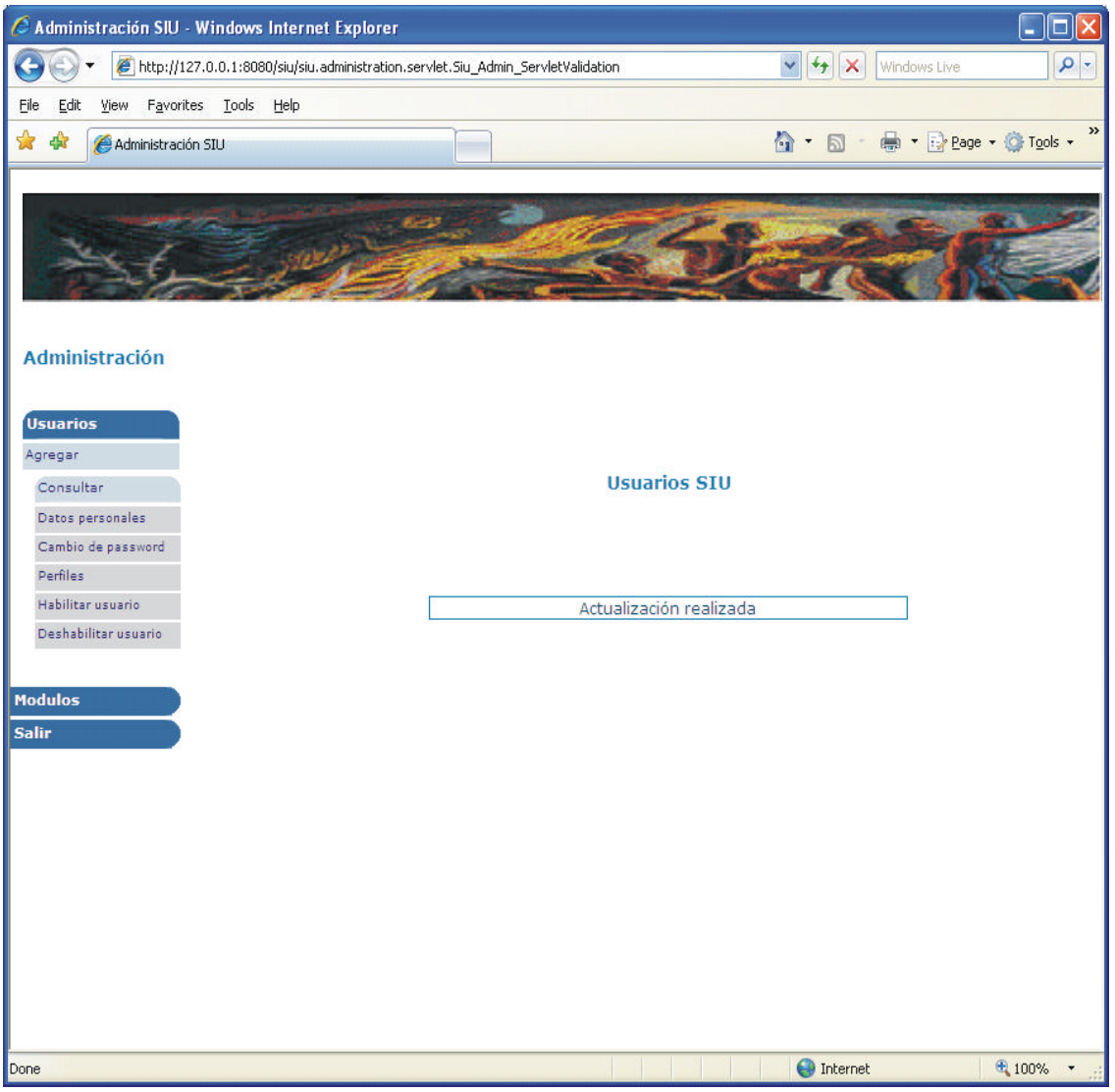


Campos:

Campo	Representación
pass	Text Field
confpass	Text Field
Enviar	Botón
Borrar	Botón

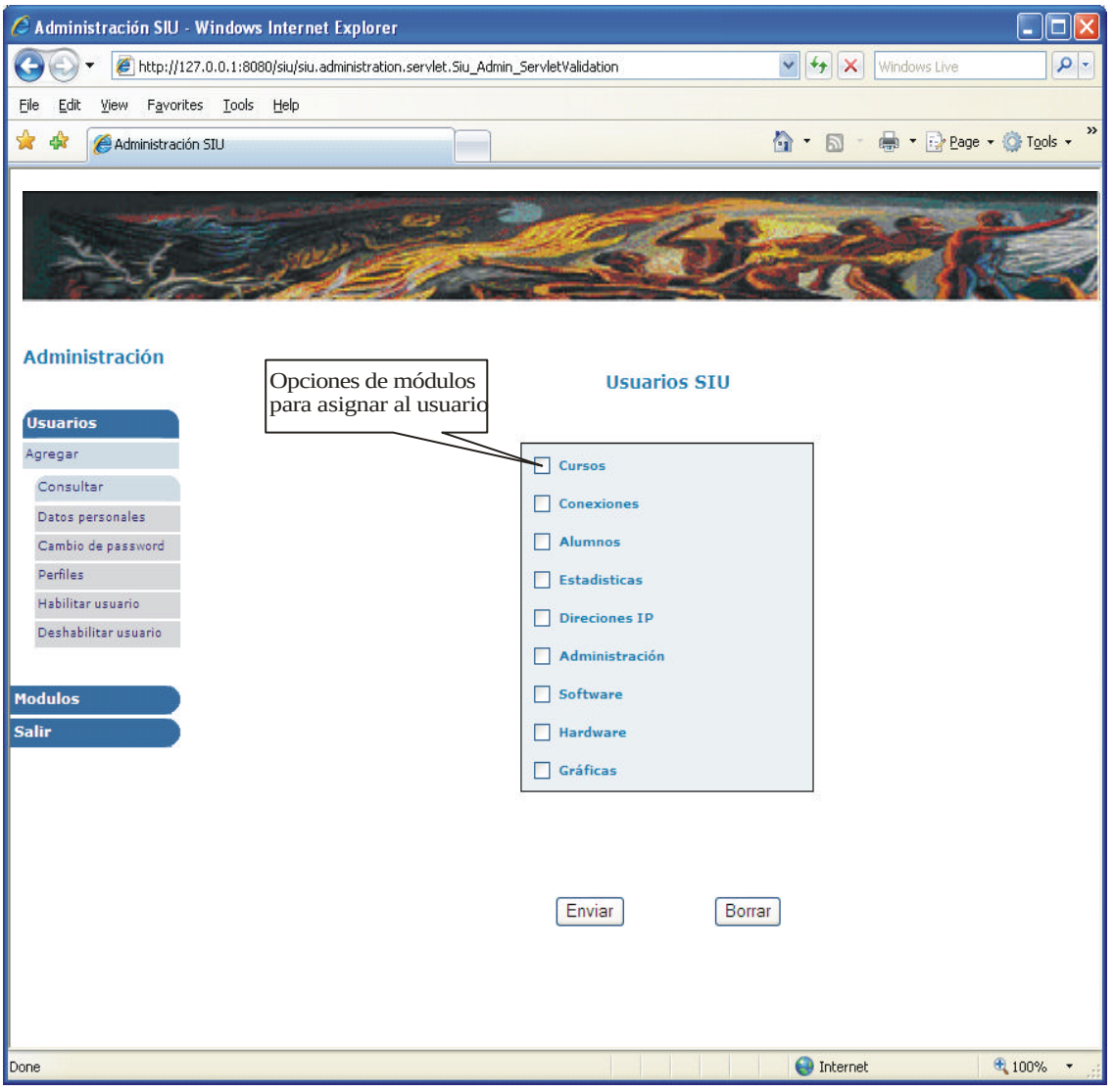
Nombre: Siu_Admin_JspPassUpdateUser.jsp
Página Anterior: Siu_Admin_JspPassUser.jsp (opción 2. Cambio de password)
Siu_Admin_JspProfileUser.jsp (opción 3. Perfiles)
Siu_Admin_JspConsultUser.jsp (opción 4. Habilitar usuario)
Siu_Admin_JspConsultUser.jsp (opción 5. Deshabilitar usuario)
Página Siguiente: Ninguna
Descripción: Página que muestra el resultado de la operación ejecutada.

Pantalla:



Nombre: Siu_Admin_JspProfileUser.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Admin_JspPassUser.jsp
Descripción: Página que muestra los módulos a los cuales puede tener acceso el usuario.

Pantalla:

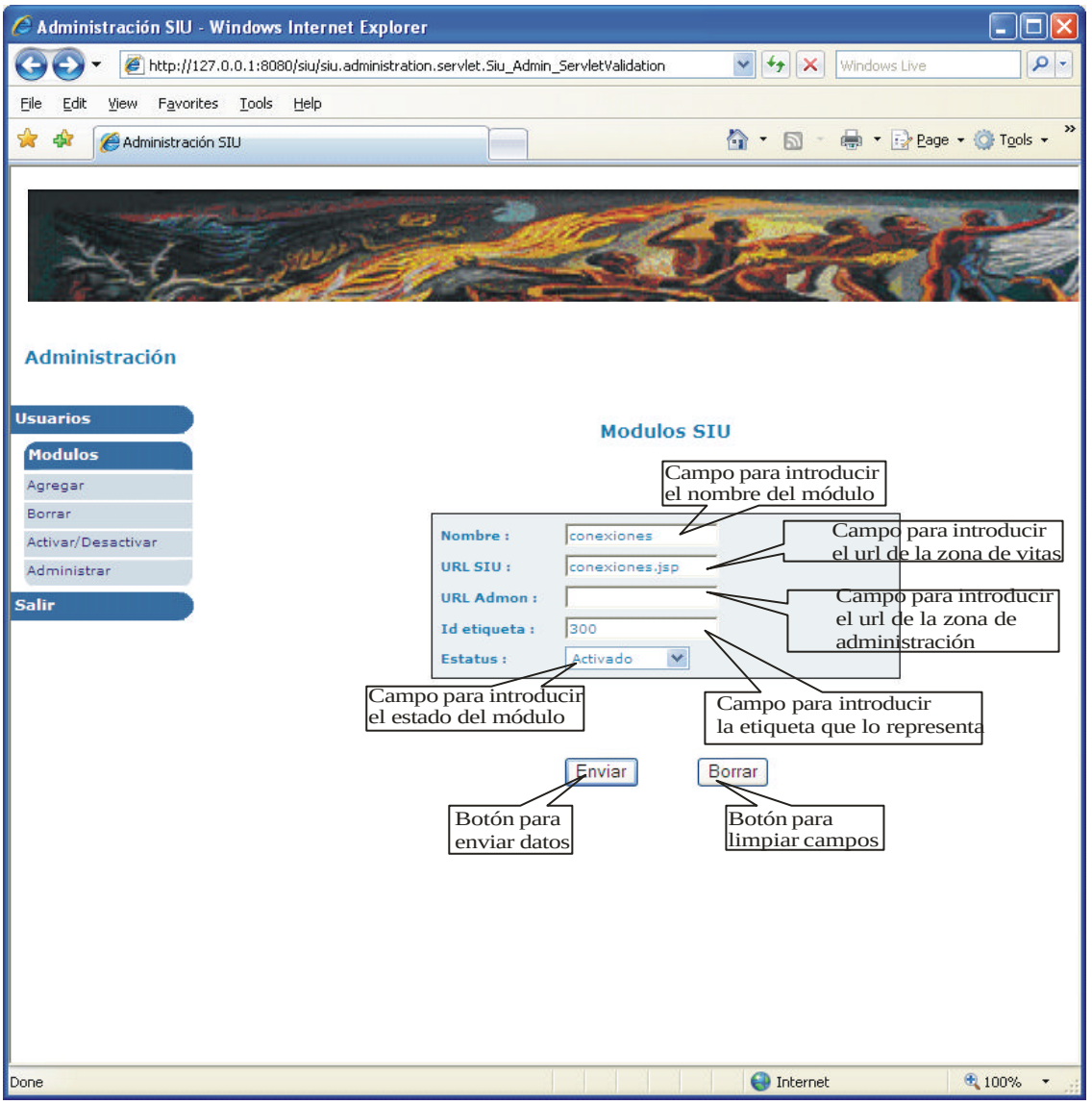


Campos:

Campo	Representación
Variable	Checkbox
Enviar	Botón
Borrar	Botón

Nombre: Siu_Admin_JspAddModule.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Admin_JspUpdateModule.jsp
Descripción: Página para agregar un módulo al SIU.

Pantalla:



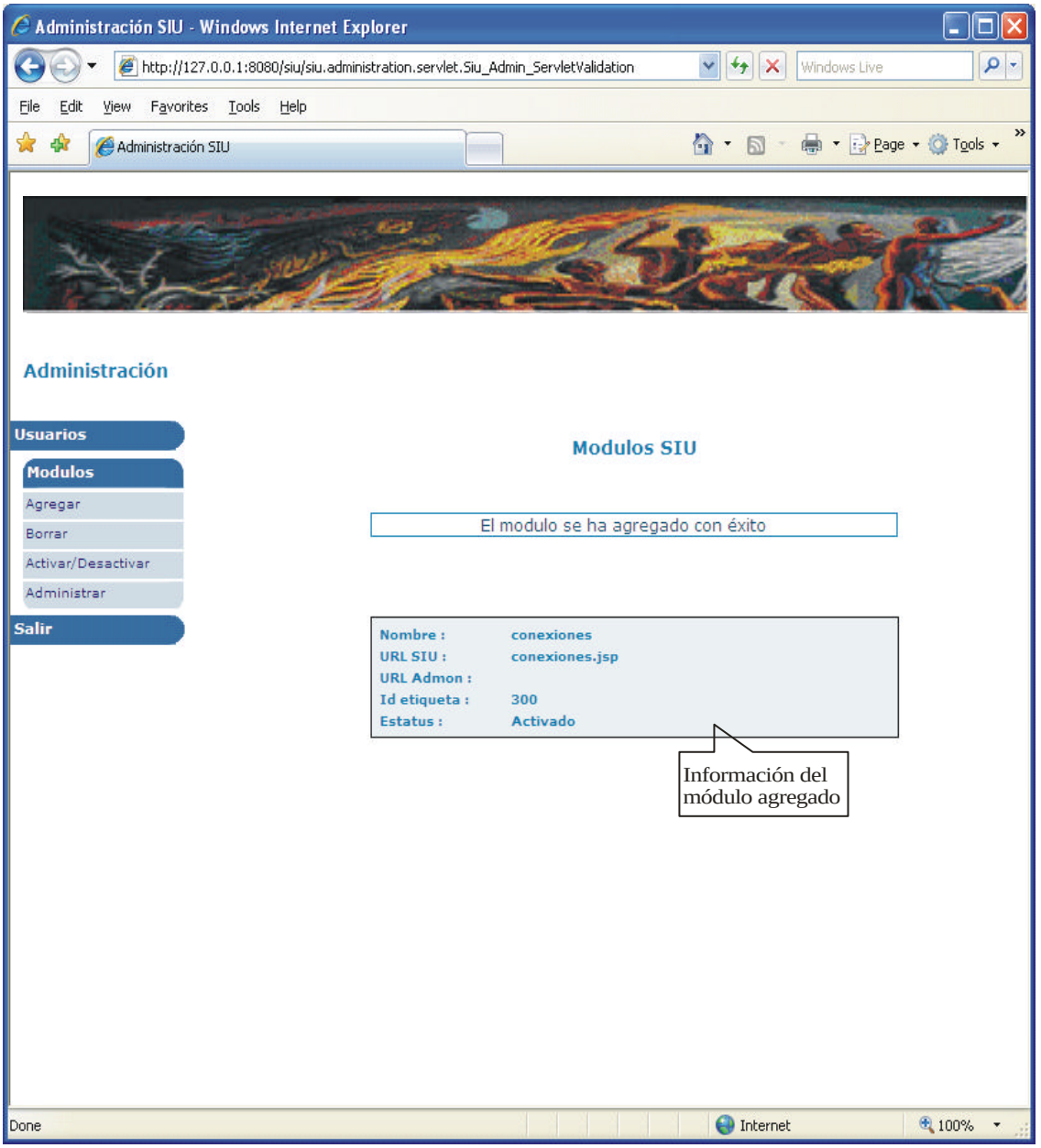
Campos:

Campo	Representación
mod_name	Text Field
mod_link	Text Field
mod_linkadm	Text Field
label_id	Text Field

mod_status	List Menu
Enviar	Botón
Borrar	Botón

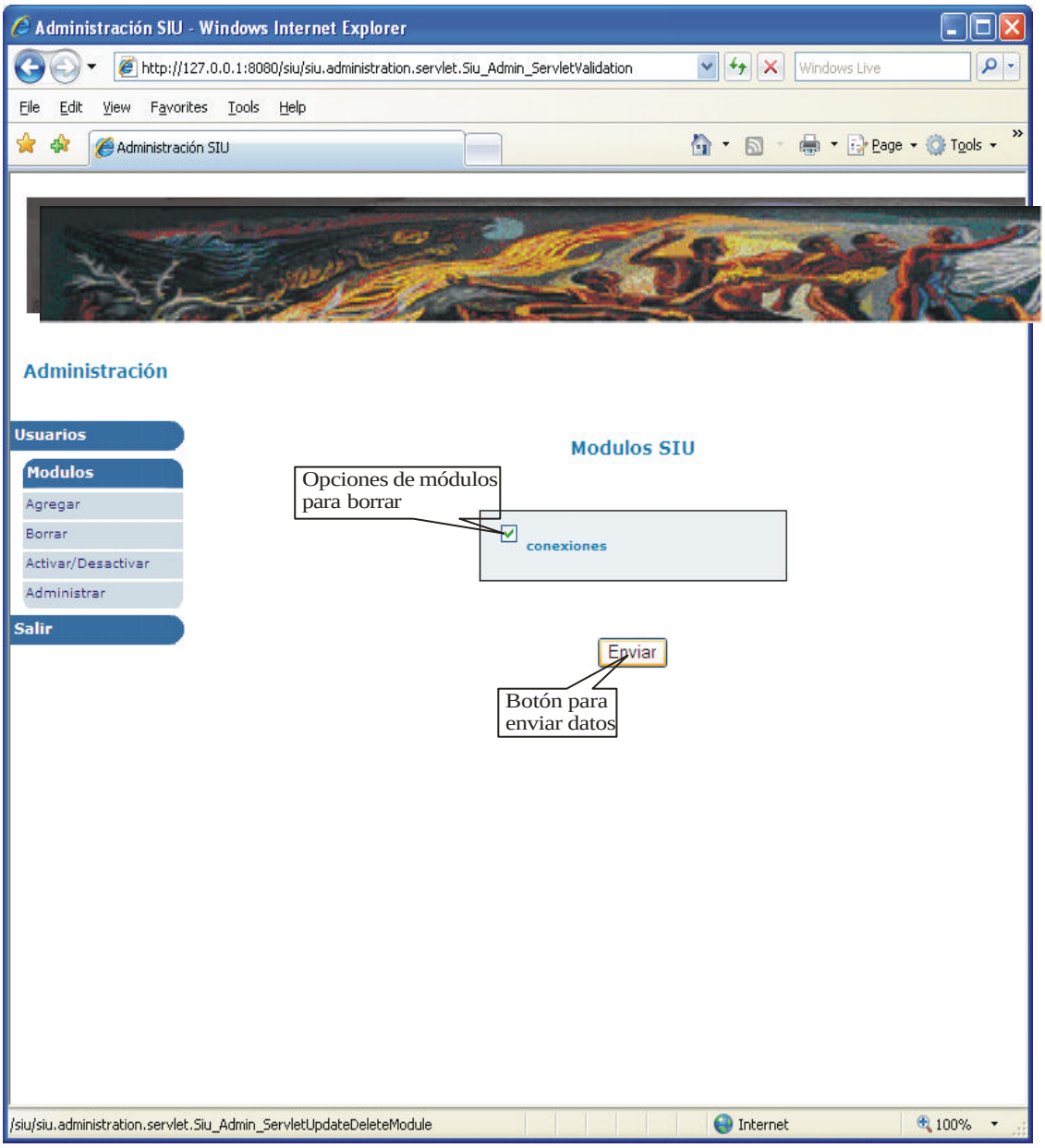
Nombre: Siu_Admin_JspUpdateModule.jsp
Página Anterior: Siu_Admin_JspAddModule.jsp
Página Siguiente: Ninguna
Descripción: Página que muestra la información del módulo agregado.

Pantalla:



Nombre: Siu_Admin_JspDeleteModule.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Admin_JspPassUser.jsp
Descripción: Página que muestra los módulos que pueden ser borrados del sistema.

Pantalla:

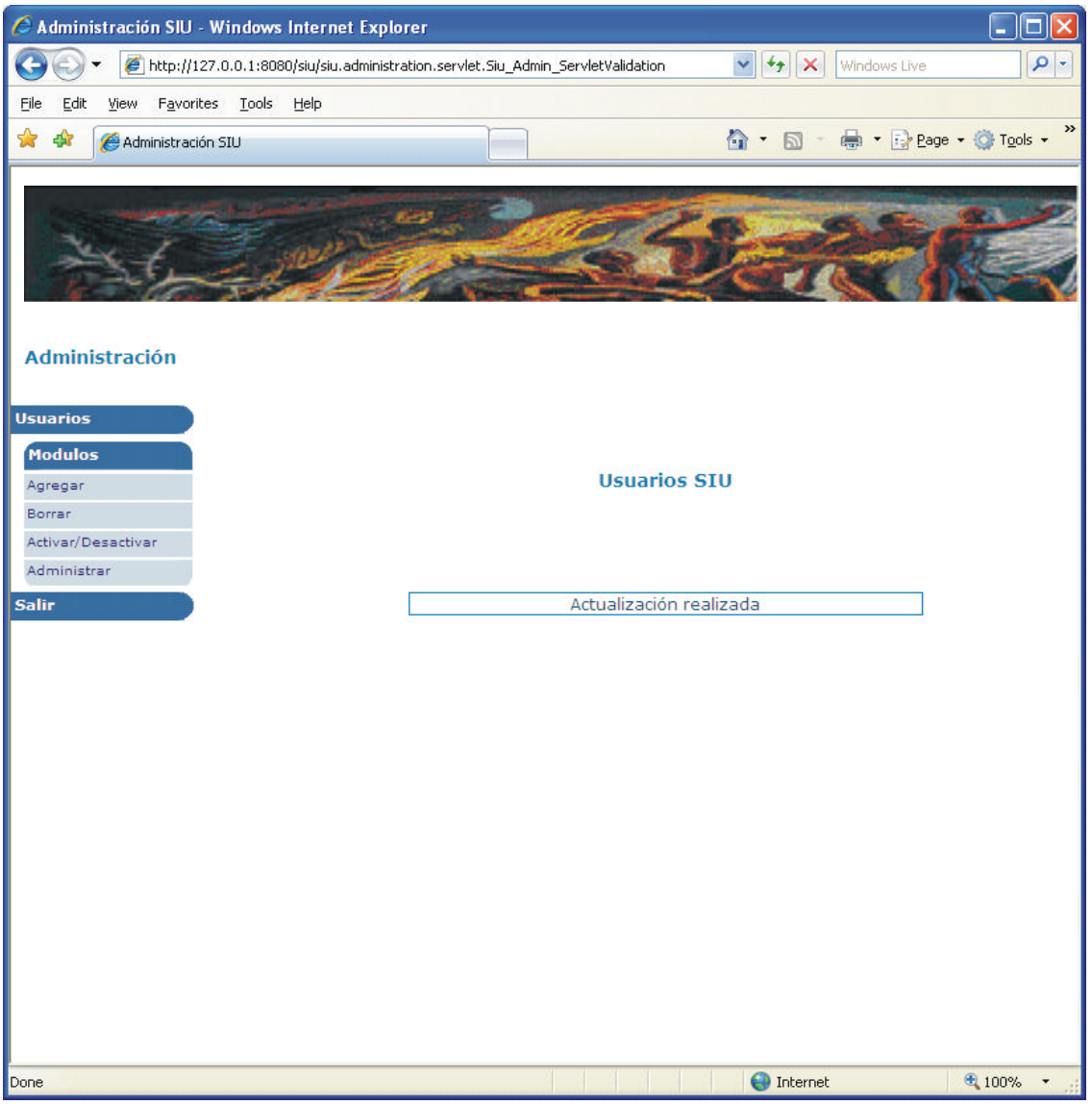


Campos:

Campo	Representación
Variable	Checkbox
Enviar	Botón

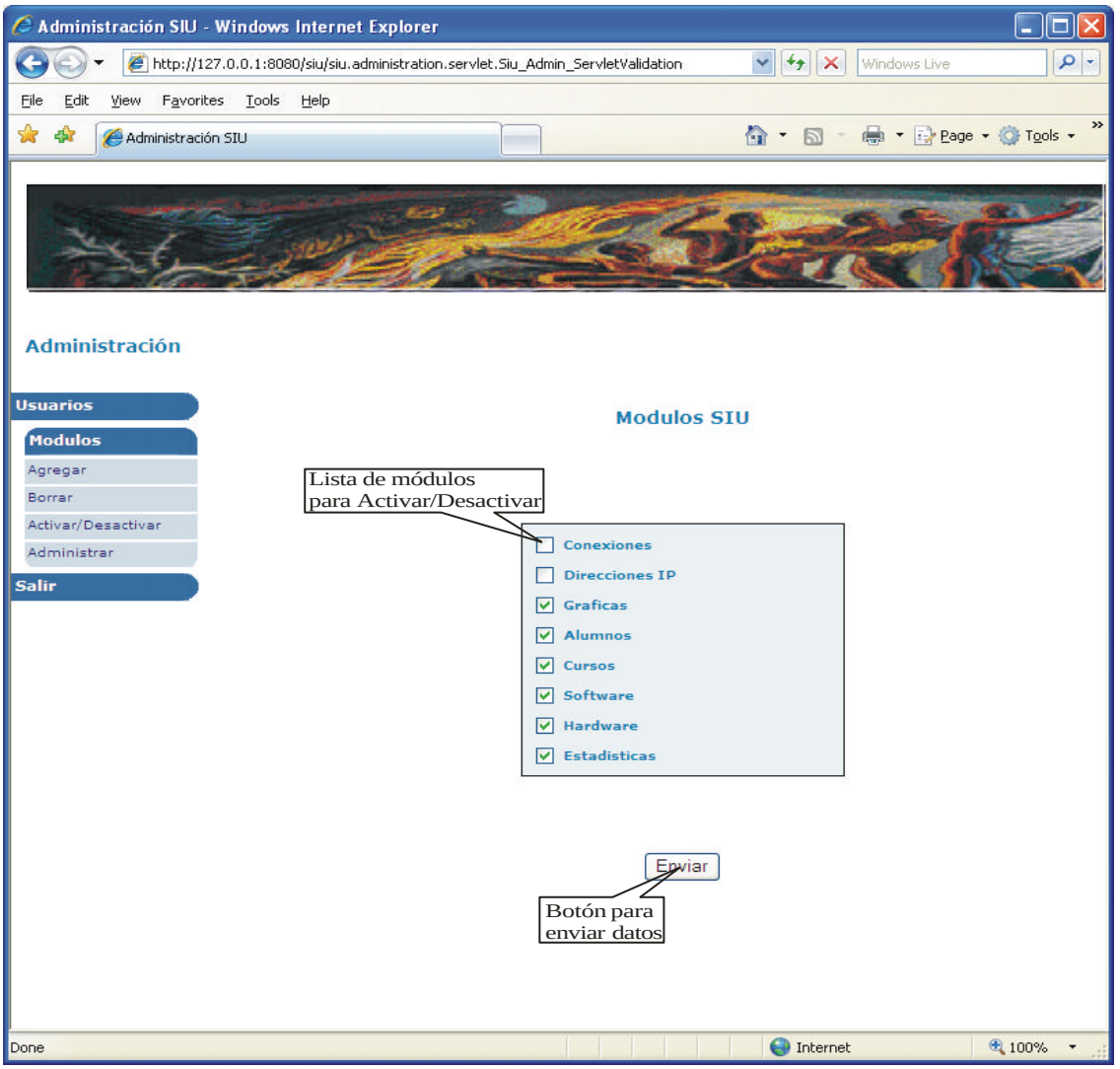
Nombre: Siu_Admin_JspPassUser.jsp
Página Anterior: Siu_Admin_JspDeleteModule.jsp (opción Borrar)
Siu_Admin_JspConsultModule.jsp (opción Activar/Desactivar)
Página Siguiente: Ninguno
Descripción: Página que muestra el resultado de la operación anterior.

Pantalla:



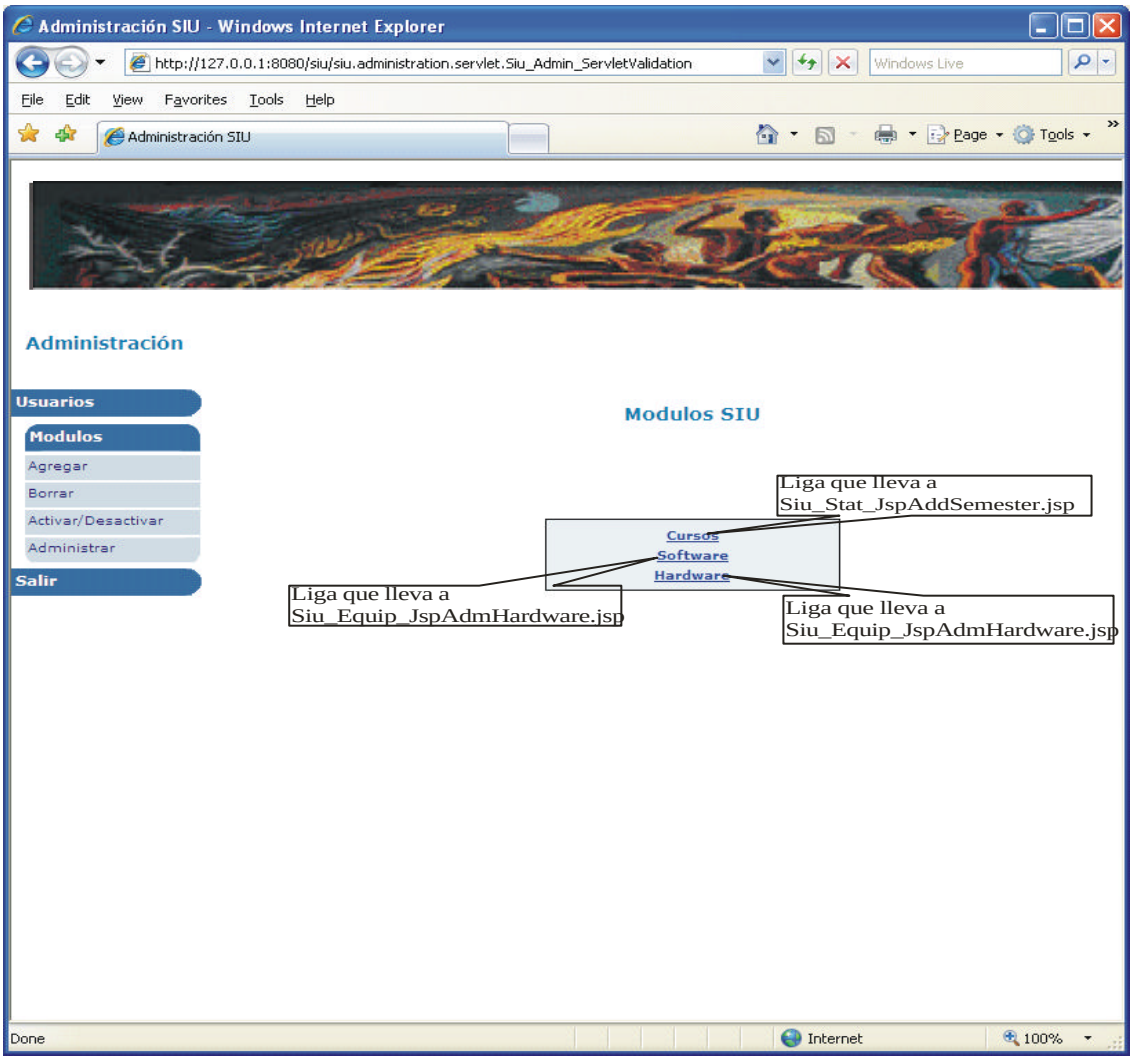
Nombre: Siu_Admin_JspConsultModule.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Admin_JspPassUser.jsp
Descripción: Página que muestra los módulos que pueden ser activados o desactivados.

Pantalla:



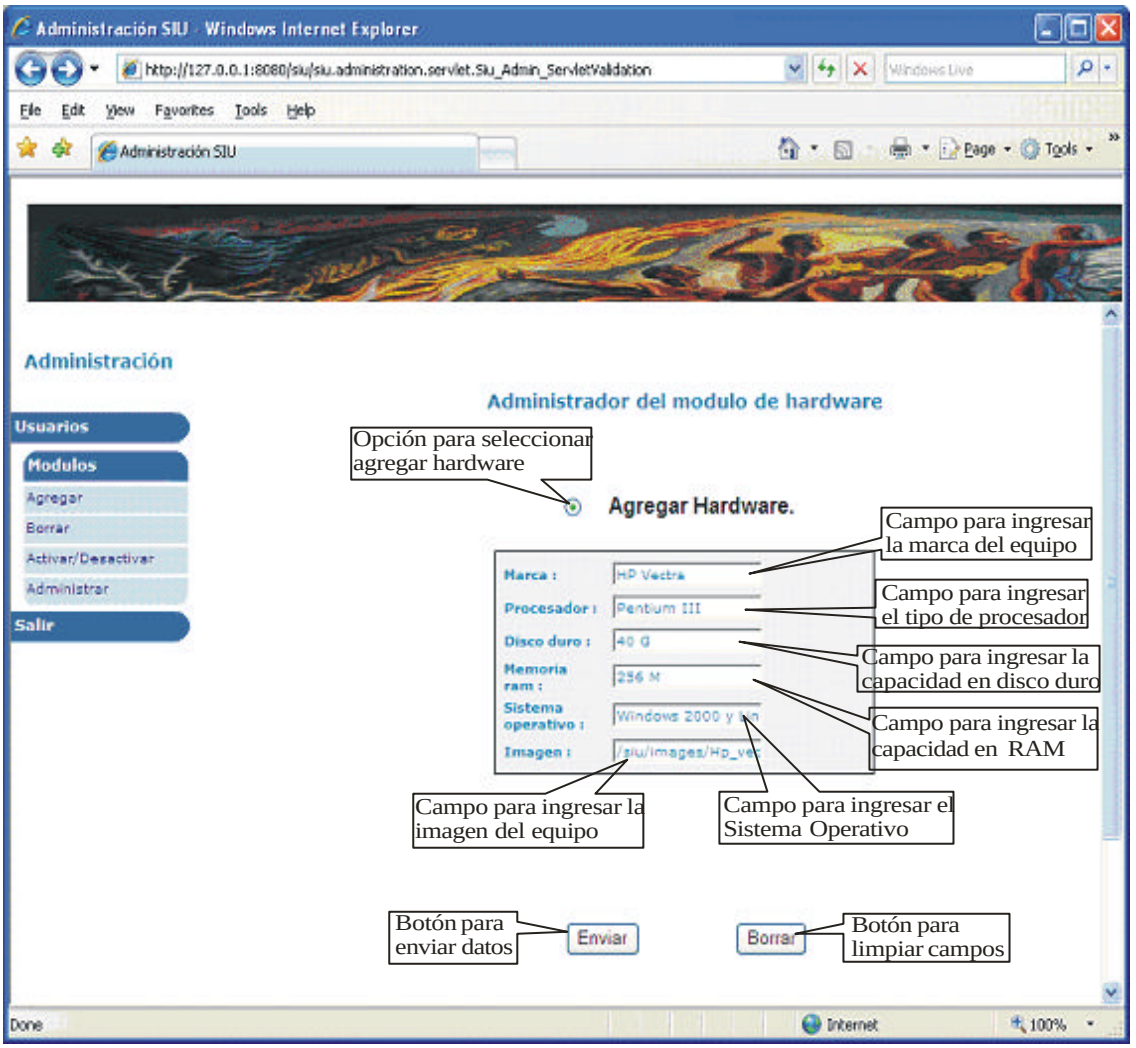
Nombre: Siu_Admin_JspAdmonModule.jsp
Página Anterior: Ninguna
Página Siguiente: Siu_Stat_JspAdmSemester.jsp (opción Cursos)
Siu_Stat_JspAdmHardware.jsp (opción Software)
Siu_Stat_JspAdmSoftware.jsp (opción Hardware)
Descripción: Página que muestra los módulos que pueden ser activados o desactivados.

Pantalla:



Nombre: Siu_Stat_JspAdmHardware.jsp
Página Anterior: Siu_Admin_JspAdmonModule.jsp
Página Siguiente: Siu_Stat_JspAdmHardware.jsp
Descripción: Página para administrar el módulo de hardware.

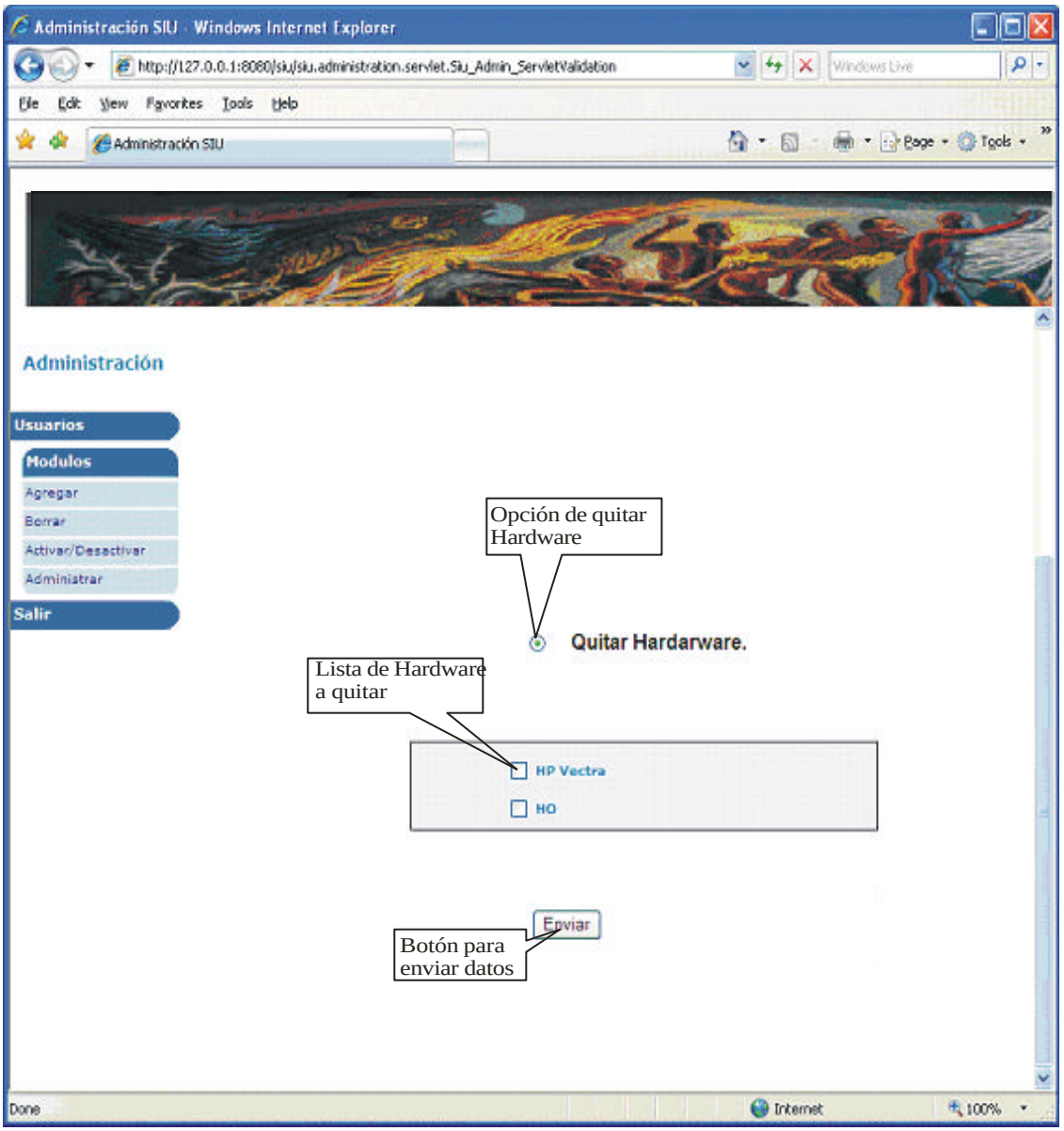
Pantalla con la opción de agregar hardware:



Campos:

Campo	Representación
hd_brand	Text Field
hd_processor	Text Field
hd_harddisk	Text Field
hd_ram	Text Field
hd_os	Text Field
hd_image	Text Field

Pantalla con la opción de quitar hardware:

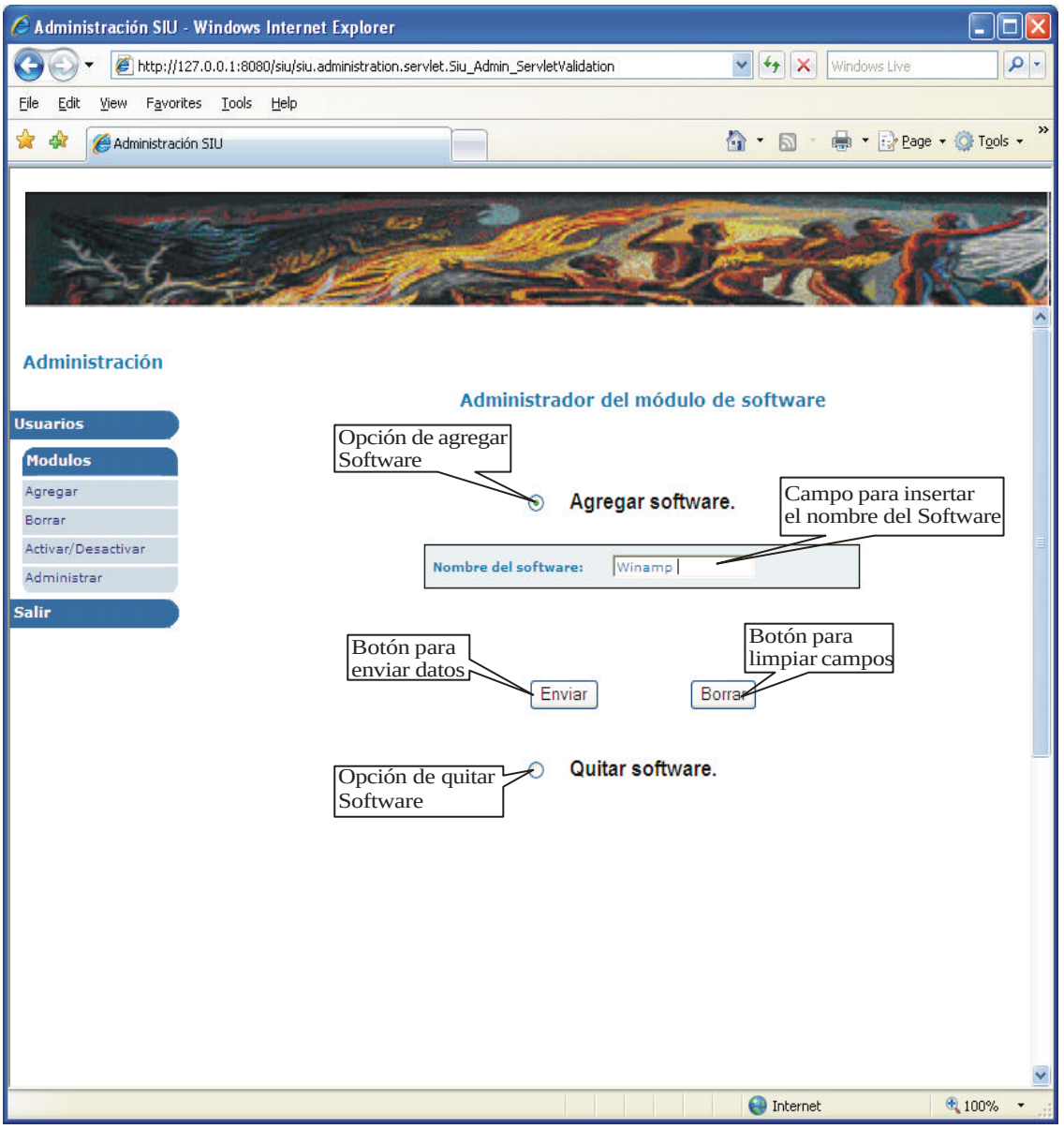


Campos:

Campo	Representación
Variable	Checkbox
Enviar	Botón

Nombre: Siu_Stat_JspAdmSoftware.jsp
Página Anterior: Siu_Admin_JspAdmonModule.jsp
Página Siguiente: Siu_Stat_JspAdmSoftware.jsp
Descripción: Página para administrar el módulo de software.

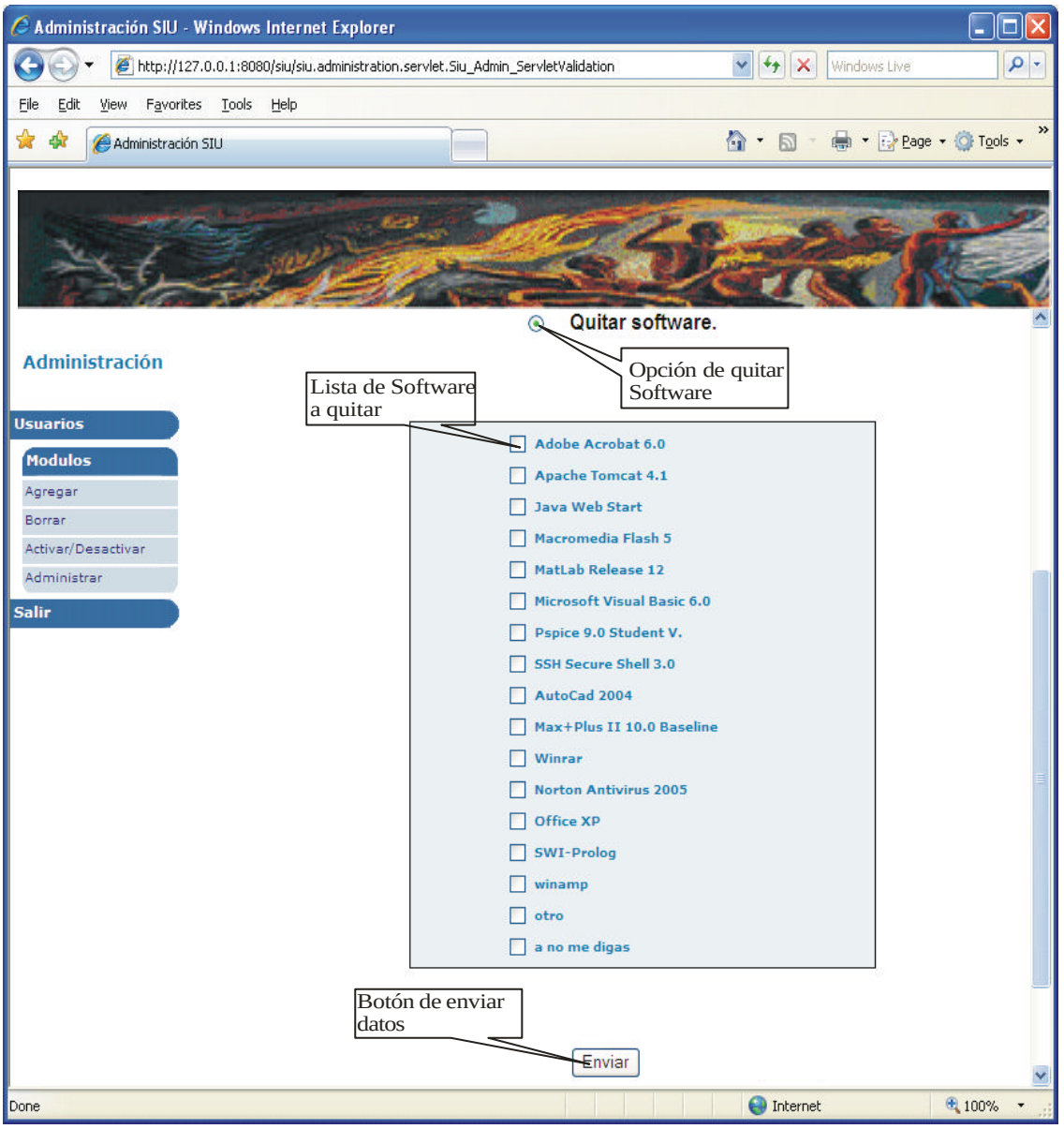
Pantalla con la opción de agregar software:



Campos:

Campo	Representación
agregar	Text Field
Enviar	Botón
Borrar	Botón

Pantalla con la opción de quitar hardware:

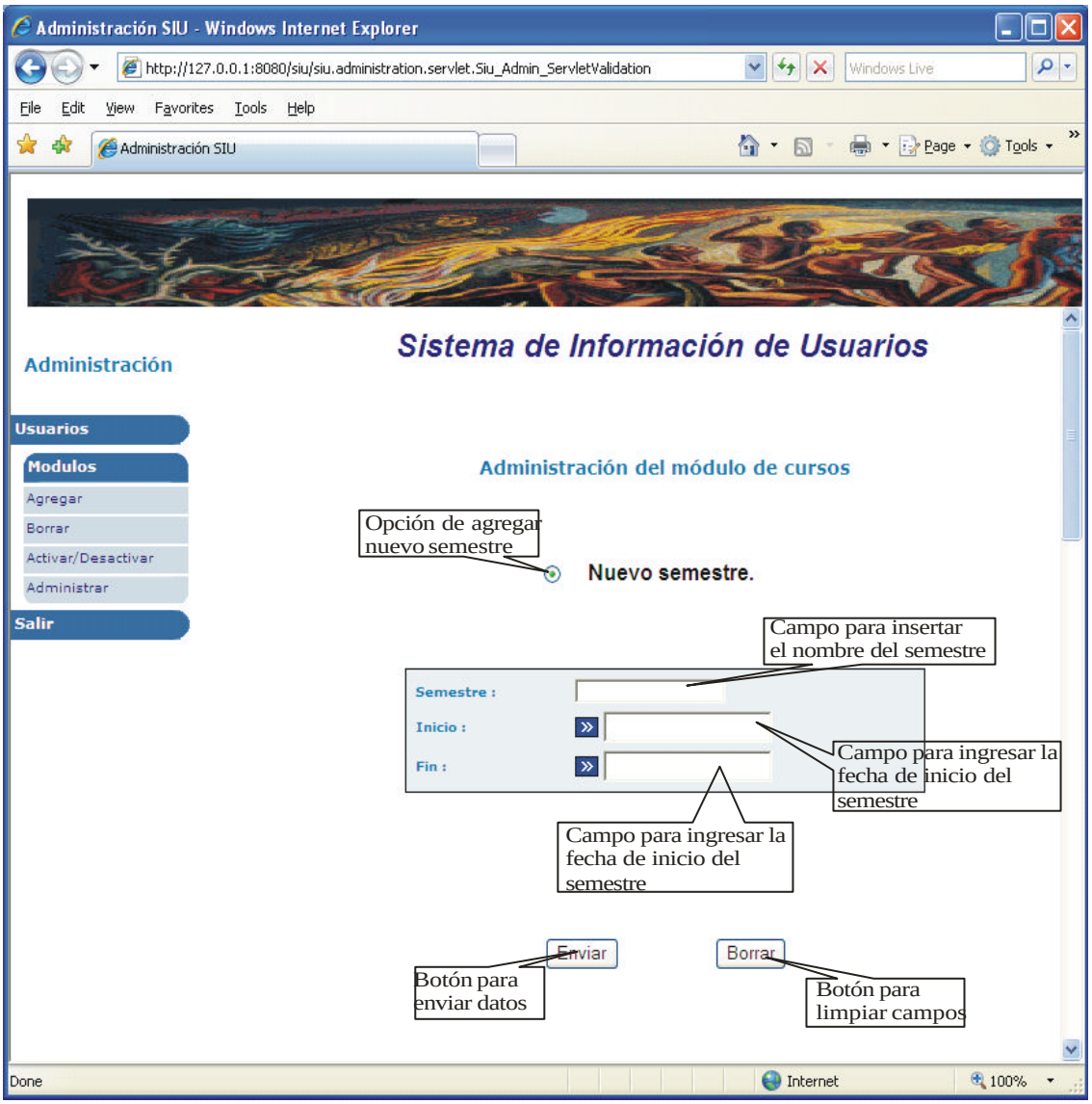


Campos:

Campo	Representación
Variable	Checkbox
Enviar	Botón

Nombre: Siu_Stat_JspAddSemester.jsp
Página Anterior: Siu_Admin_JspAdmonModule.jsp
Página Siguiente: Siu_Stat_JspAddSemester.jsp
Descripción: Página para administrar los cursos y clases impartidos en el Laboratorio de Computación.

Pantalla con la opción de agregar nuevo semestre:

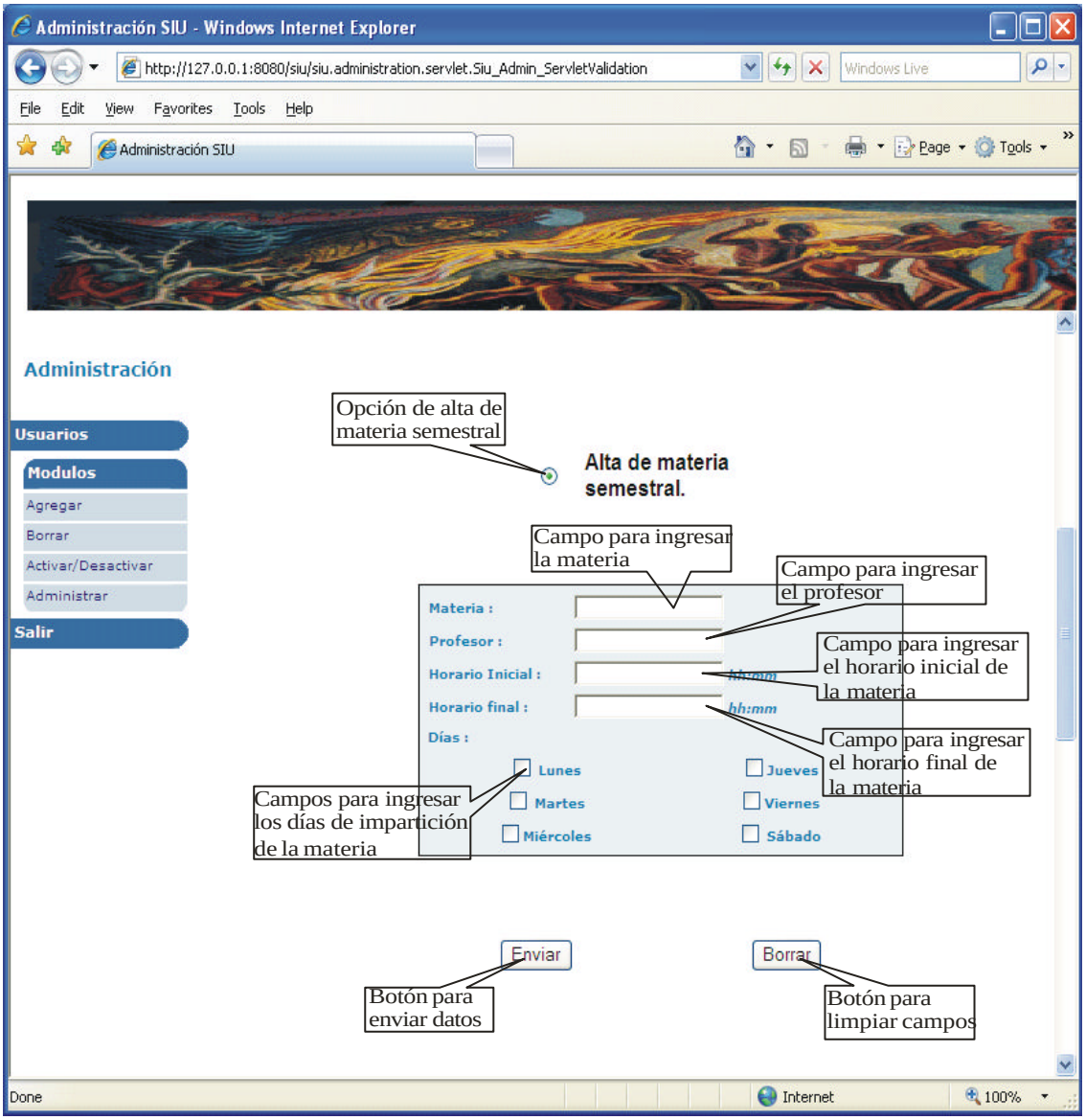


Campos:

Campo	Representación
agregar	Text Field
txtFech	Text Field
txtFech2	Text Field
Enviar	Botón

Borrar	Botón

Pantalla con la opción dar alta de materia semestral:

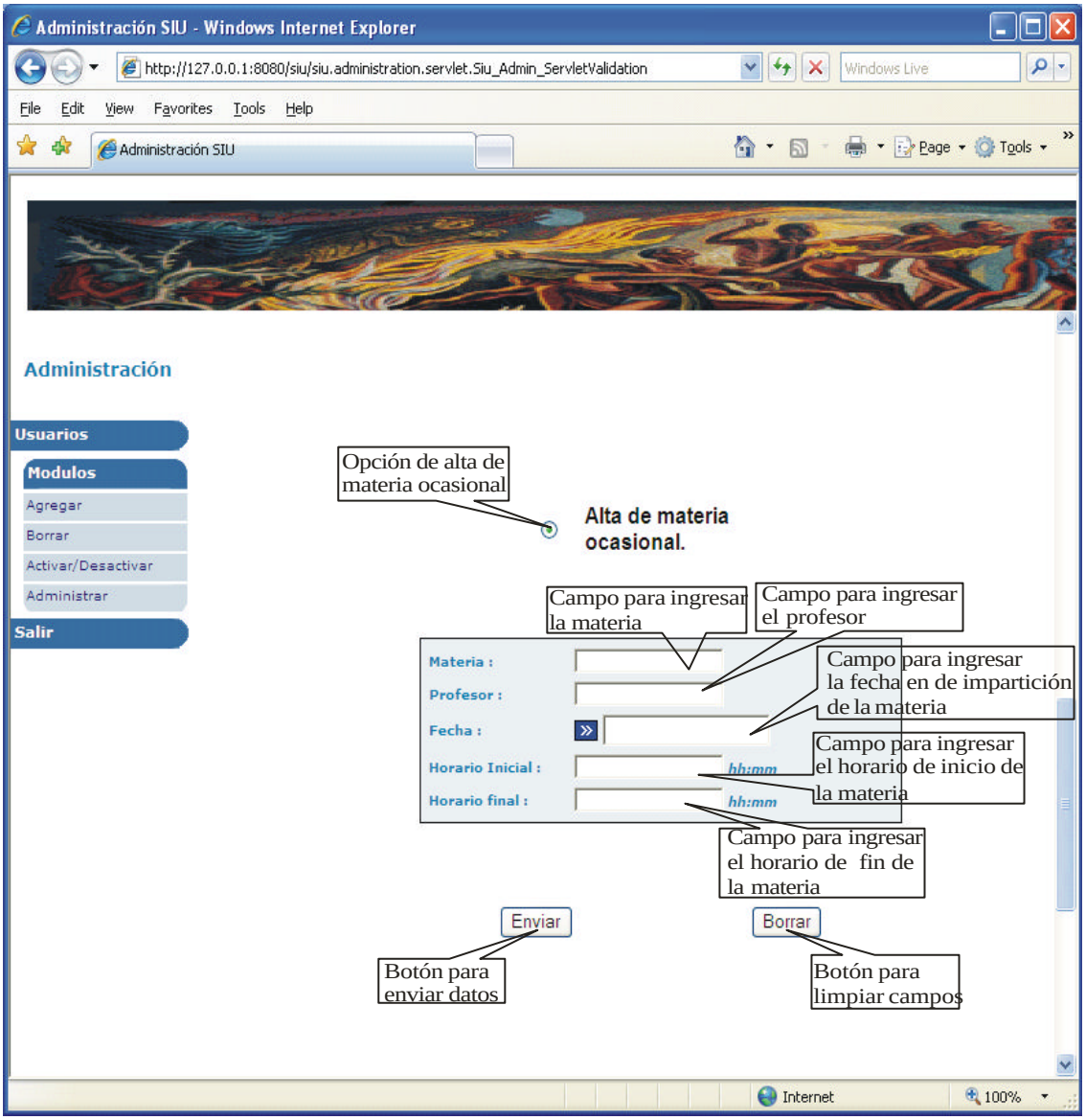


Campos:

Campo	Representación
materia	Text Field
profesor	Text Field
horaIni	Text Field
horaFin	Text Field
Checkbox	checkbox
Enviar	Botón

Borrar	Botón

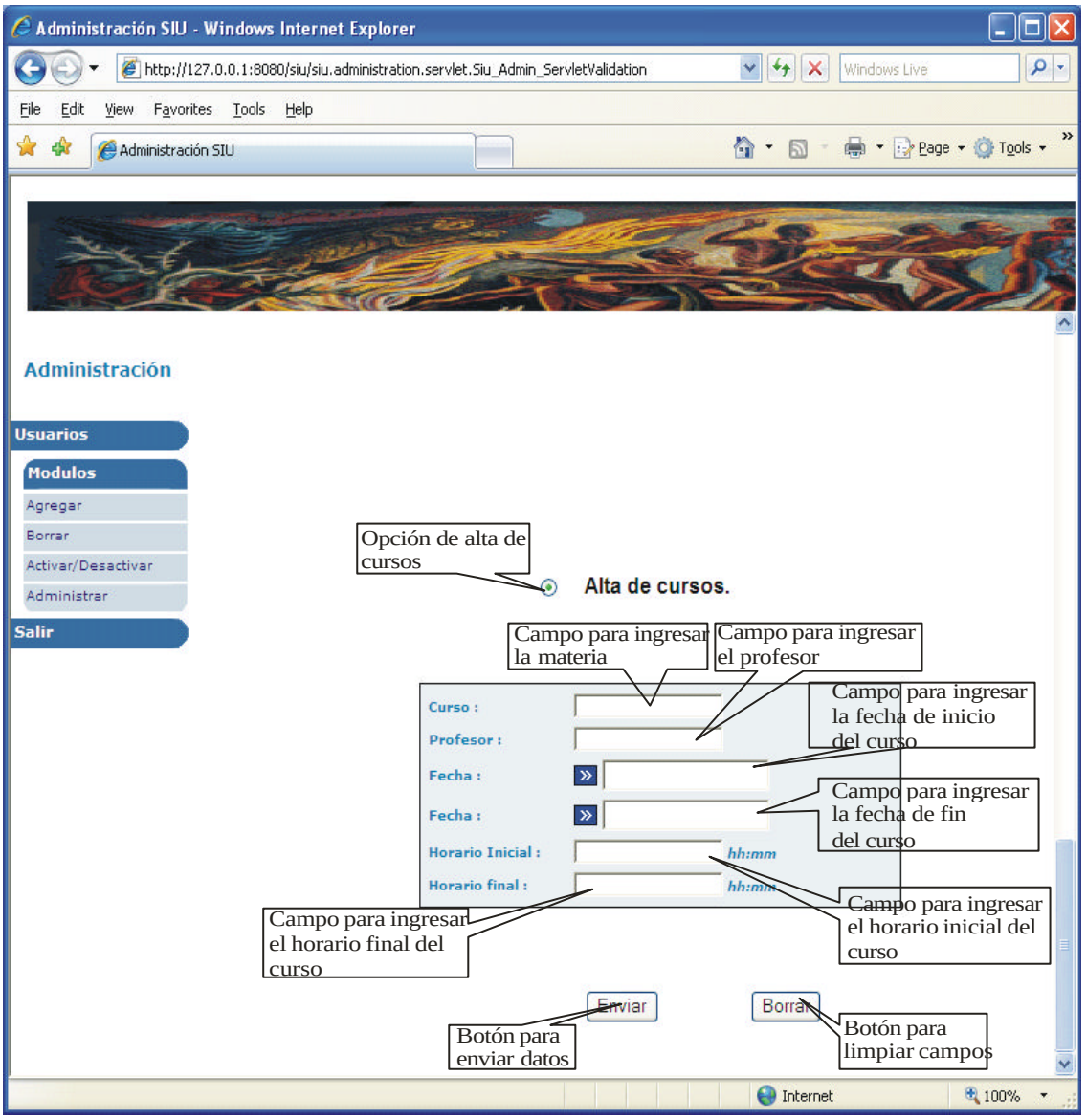
Pantalla con la opción dar alta de materia ocasional:



Campos:

Campo	Representación
materia	Text Field
profesor	Text Field
horaIni	Text Field
horaFin	Text Field
txtFech3	Text Field
Enviar	Botón
Borrar	Botón

Pantalla con la opción dar alta de cursos:



Campos:

Campo	Representación
materia	Text Field
profesor	Text Field
horaIni	Text Field
horaFin	Text Field
txtFech4	Text Field
txtFech5	Text Field
Enviar	Botón
Borrar	Botón

3.8. Elección de Tecnologías

3.8.1. Lenguaje de programación

Para la codificación del SIU se utilizará el lenguaje de programación Java, creado por Sun Microsystems. El motivo es que en la actualidad es un lenguaje muy extendido y tiene gran peso en el ámbito de Internet, ya que no sólo es un lenguaje de programación, si no que provee toda una plataforma empresarial. Algunas de sus principales ventajas es que el lenguaje es independiente de la plataforma, es potente, seguro y gratuito.

Otra razón de peso es que la herramienta que genera gráficas y que se utiliza para generar las mismas en el SIU, está realizada en Java. El software se obtiene del sitio de Internet http://java.sun.com

Como un lenguaje de programación auxiliar se utilizará Javascript. Este lenguaje posee gran cantidad de funciones que se ejecutan directamente en el diente y que usan generalmente para la validación de datos. Como algunas características de funcionalidad de Javascript cambian dependiendo del Navegador, sólo se utilizaran funciones que sean independientes del explorador que se utilice.

3.8.2. Servidor Web y Aplicaciones

El servidor web que se utilizará para el SIU es Tomcat (también llamado Jakarta Tomcat o Apache Tomcat) debido a que es gratuito y funciona como un contenedor de Servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat. Implementa las especificaciones de los Servlets y JavaServer Page (JSP) de Sun Microsystems. El software se obtiene del sitio de Internet http://www.apache.org/

3.8.3. Base de datos y manejador de base de datos

Para el manejador de la base de datos se utilizará es PostgresSQL por las siguientes razones:

1. Es software libre, además de ser potente, flexible, gratuito, fácil de administrar y se encuentra disponible para instalarse tanto en sistemas operativos Windows, como en Linux.
2. Sigue todos los estándares ANSI y SQL y toda la referencia puede ser localizada fácilmente en Internet.
3. Con la versión 8.1 trae un BDMS que hace mucho más flexible el uso.
4. Se encuentra instalado en el servidor donde se lleva a cabo la inserción de los datos de los usuarios que utilizan el Laboratorio de Computación.

El software se obtiene del sitio de Internet http://www.postgresql.org

4. Programación

4.1. Java

4.1.1. Historia de Java

El lenguaje de programación Java se originó en 1991 como parte de la búsqueda para crear un nuevo lenguaje de programación, llamado "Oak", que sería el puente de comunicación de interacción de las personas con sus electrodomésticos, como los televisores; pretendían unir a estos dispositivos con una unidad central de proceso.

El concepto inicial falló y los creadores tuvieron que verse en la necesidad de encontrar un nuevo mercado para su nuevo lenguaje de programación. Afortunadamente el mundo del Web comenzó ha hacerse popular y los creadores de Oak creyeron que el lenguaje era perfecto para la construcción de componentes multimedia que complementaran las páginas Web. Estos nuevos productos fueron llamados "Applets", estos componentes se convirtieron en el uso inicial de Oak, los programadores adoptaron este nuevo lenguaje que se convertiría en Java.

4.1.2. Características de Java

Este lenguaje de programación ofrece las siguientes características:

Sencillo: Ofrece toda la funcionalidad de un lenguaje potente, pero sin las particularidades sofisticadas que provocan confusiones. Cabe mencionar que fue diseñado con la finalidad de que su aprendizaje y utilización resultaran naturales para el programador profesional.

Orientado a objetos: Java ha tomado prestadas muchas ideas de entornos orientados a objetos de las últimas décadas, consiguiendo un equilibrio espectacular entre el modelo purista "todo es un objeto" y el modelo pragmático. El modelo de objetos es sencillo y de fácil ampliación. Trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma orientado a objetos: encapsulación, herencia y polimorfismo.

Distribuido: Es distribuido porque puede estar almacenado en un lugar y ser ejecutado en otro, además de que provee soporte tecnologías que implementan sistemas distribuidos.

Robusto: Para ganar confiabilidad, realiza verificaciones tanto en tiempo de compilación como en tiempo de ejecución y, así, consigue encontrar rápidamente los errores más comunes en el desarrollo del programa.

Arquitectura neutral: Para hacer independientes las aplicaciones escritas en Java, se compila su código en un archivo objeto (bytecodes) que no depende de la arquitectura de la máquina en que será ejecutado. Cualquier máquina que tenga el sistema "runtime" puede ejecutar ese código objeto, sin importar la plataforma en que ha sido generado.

Seguridad: El código en Java pasa por una serie de pruebas antes de ejecutarse en una computadora. Dicho código se pasa a través de un verificador de "bytecodes" que comprueba el formato de los fragmentos de código y aplica un probador de teoremas para detectar fragmentos de código ilegal, el cual viola derechos de acceso sobre objetos o intenta cambiar el tipo o clase de un objeto. Este mismo mecanismo permite la portabilidad entre plataformas.

Portabilidad: Implementa estándares para facilitar el desarrollo, tal es el caso de los tipos de datos o interfaces gráficas de usuario que mantienen la independencia de la plataforma.

Interpretado: El intérprete Java, que es un sistema "runtime", puede ejecutar directamente el código objeto. La independencia de la plataforma tiene un costo, Java es más lento que otros lenguajes de programación, ya que su código debe ser interpretado y no ejecutado, como sucede en lenguajes como C++.

Multihilos: Fue diseñado para satisfacer los requerimientos del mundo real, en cuanto a crear programas interactivos en un ambiente de red. Para ello, proporciona la programación multihilo, la cual permite la escritura de programas que realicen varias cosas simultáneamente.

Desarrollo en la Tesis

La versión del lenguaje de programación Java que se utilizó en la tesis fue la 1.5, que en el momento en que se empezó el desarrollo del sistema estaba disponible como la última versión estable.

Servlet: Es un programa del lado del servidor que puede tener embebido código HTML y ser mostrado como una página Web. En el caso del sistema los Servlets funcionan como controladores en el sistema. Las clases e interfaces que se ocupan para la creación de Servlets se encuentran en el paquete "javax.servlet.*".

JSP: Lo JSPs son documentos tipo texto que escriben como procesar un request de un ciente y crear una respuesta. Al contrario de los Servlets, los JSPs tienen código Java embebido en código HTML. El tener código Java ayuda a que las páginas Web sean dinámicas. En el sistema, los JSPs sirven para crear las vistas.

JDBC: (Java Database Connectivity) es la parte de Java que se encarga de la conexión a bases de datos ya que provee las clases e interfaces necesarias para conectar los programas java a bases de datos relacionales de distintos proveedores (como Oracle,

Sybase, Informix, MS SQL Server, etc). Las clases e interfaces para la conexión a bases de datos vía JDBC, se encuentran en el paquete “java.sql.*”.

JavaBeans: Son clases Java que permiten modificar sus atributos de manera dinámica (sin necesidad de recompilar). En el sistema funcionan como el modelo de datos.

El uso de estándares de programación se especifican en el Anexo A: Estándares de programación.

4.2. SQL

4.2.1. Antecedentes

La historia de SQL empieza en 1974 con la definición, por parte de Donald Chamberlin y de otras personas que trabajaban en los laboratorios de investigación de IBM, de un lenguaje para la especificación de las características de las bases de datos que adoptaban el modelo relacional. Este lenguaje se llamaba SEQUEL (Structured English Query Language) y se implementó en un prototipo llamado SEQUEL-XRM entre 1974 y 1975. Las experimentaciones con ese prototipo condujeron, entre 1976 y 1977, a una revisión del lenguaje (SEQUEL/2), que a partir de ese momento cambió de nombre por motivos legales, convirtiéndose en SQL. El prototipo (System R), basado en este lenguaje, se adoptó y utilizó internamente en IBM y lo adoptaron algunos de sus clientes elegidos. Gracias al éxito de este sistema, que no estaba todavía comercializado, también otras compañías empezaron a desarrollar sus productos relacionales basados en SQL. A partir de 1981, IBM comenzó a entregar sus productos relacionales y en 1983 empezó a vender DB2. En el curso de los años ochenta, numerosas compañías (por ejemplo Oracle y Sybase, sólo por citar algunos) comercializaron productos basados en SQL, que se convierte en el estándar industrial de hecho por lo que respecta a las bases de datos relacionales.

En 1986, el ANSI adoptó SQL (sustancialmente adoptó el dialecto SQL de IBM) como estándar para los lenguajes relacionales y en 1987 se transformó en estándar ISO. Esta versión del estándar va con el nombre de SQL/89. En los años siguientes, éste ha sufrido diversas revisiones que han conducido primero a la versión SQL/89 y, posteriormente, a la actual SQL/92.

4.2.2. SQL/89

En su primera versión del SQL-89 se tienen tres partes:

- *El lenguaje de definición de datos (LDD).* Contiene todas las instrucciones para definir el esquema de una base de datos, como son: create, alter y drop.
- *El lenguaje de manipulación de datos (LMD).* Contiene las instrucciones de manejo de las tablas como son: select, insert, delete y update, y para control de concurrencia como: commit y rollback.
- *El lenguaje de control de datos (LCD).* Contiene aquellas instrucciones para dar y revocar permisos de acceso a los datos de la base de datos, como son: grant y revoke.

4.2.3. SQL92

Este estándar contiene muchas más especificaciones que la presentadas en la versión anterior, como lo son:

- Creación de Esquemas: Sirve para crear privilegio a usuarios sobre la actualización de tablas a ciertos usuarios.
- Tipos de datos:
  - Números exactos (Integer, Small Integer, Numeric y Decimal).
  - Números aproximados (Real, Double y Float).
  - Cadenas de caracteres (Character y Character varyng).
  - Cadenas de bits (Bit).
  - Fechas y horas (Date, Time y Timestamp).
  - Intervalos (Yera-moth, Day-time).
- Definición de tablas:
  - Dar el nombre de la tabla.
  - Definir cada columna, incluyendo restricciones de columna.
- Manipulación de datos:
  - Consultas simples: Una consulta que involucra una sola tabla de la base de datos.
  - Consultas multicapa: Hacer consultas en más de una tabla.
  - Subconsultas: Hacer una consulta dentro de otra.
- GROUP BY y HAVING:
  - GROUP BY: Para indicar cuáles filas deben de agruparse sobre un valor común de las columna(s) especificada(s).
  - HAVING: Una cláusula que impone condiciones a los grupos.
- Definición de vistas: Una vista es una porción restringida de la base de datos, y se obtiene de una o más tablas.

4.2.4. SQL3

El lenguaje estándar llamado SQL3, prometió ser un aumento de la segunda generación de SQL (comúnmente conocido como SQL/92, debido al año de su publicación), SQL3 fue originalmente planeado para su uso en el año 1996, pero tardó 7 años en desarrollarse en vez de los tres o cuatro que se pensaba iba a tardar.

SQL3 está caracterizado como “SQL orientado a objetos” y es la base de algunos sistemas de manejo de bases de datos orientadas a objetos (incluyendo ORACLE, Informix Universal Server, IBM’s DB Universal Database y Cloudscape, además de otros).

Envuelve características adicionales que se consideran herencia de los SQL relacionales, así como también una reestructuración de los documentos del estándar con vista a una mayor progresión hacia normas más efectivas.

Esta dividido divididos en “aspectos relacionales” y “aspectos relacionados con objetos”.

Aspectos relacionales

Es más adecuado llamar a esta categoría como "aspectos que relacionan el papel de SQL en el modelado de datos". Estos aspectos no están estrictamente limitados al modelo relacional, pero no están relacionados con la orientación a objetos.

Estos aspectos se dividen en cinco grupos:

- Nuevos tipos de datos:
  - Character Large Object (CLOB): Funciona como una cadena de caracteres, pero contiene restricciones que impiden su uso como PRIMARY KEY o predicados UNIQUE, FOREIGN KEY, y en comparaciones a excepción de las de igualdad o desigualdad.
  - Binary Large Object (BLOB): Cadena de caracteres que no puede ser usado con las cláusulas GROUP BY u ORDER BY.
  - Bolean: Permite registrar valores de verdad o de falso.
- Nuevos predicados:
  - Se introdujo el predicado SIMILAR, que ofrece la posibilidad de comparar modelos. Por ejemplo:
    - … WHERE NAME SIMILAR TO '(SQL-(89|92)) | (SQL(1|2|3))'
  - El otro predicado introducido fue DISTINCT, considera que dos valores nulos no pueden ser distintos el uno del otro.
- Semántica mejorada: Ofrece la oportunidad de construir aplicaciones recursivas y determinar un "savepoint" en el que una aplicación puede deshacer las acciones realizadas después de comenzar el savepoint, sin deshacer todas las acciones de una transferencia entera.
- Seguridad adicional: Los privilegios pueden ser otorgados según un rol y este a su vez puede otorgar privilegios individuales para otros roles.
- La base de datos activa: Se hace por medio de "triggers", esto permite realizar operaciones seguras siempre que una aplicación realice operaciones en tablas particulares.

Orientación a objetos

SQL3 se caracteriza porque fue desarrollado principalmente para manejar objetos. Algunas de las características que están dentro de esta categoría fueron definidas en el estándar SQL/PSM publicado en 1996 específicamente para llamadas a funciones y procedimientos desde SQL.

- Notaciones funcionales y de punto: El acceso a los atributos de tipos definidos por el usuario puede hacerse mediante dos notaciones:
  - WHERE emp.salary > 1000
  - WHERE salary(emp) > 1000
- Tipos de estructuras definidas por el usuario.

El hecho de tener un estándar definido por un lenguaje para bases de datos relacionales abre potencialmente el camino a la ínter comunicabilidad entre todos los productos que se basan en él, esto debido a la diversidad de lenguajes y de bases de datos existentes, de no existir el estándar la manera de comunicarse entre unos y otros sería difícil. Es de eso de lo que trata el Structured Query Language, que no es más que un lenguaje estándar de comunicación con bases de datos. Hablamos por tanto de un lenguaje normalizado que nos permite trabajar con cualquier tipo de lenguaje (ASP o PHP) en combinación con cualquier tipo de base de datos (MS Access, SQL Server, MySQL, PostgresSQL...).

En general cada productor adopta e implementa en la propia base de datos sólo el corazón del lenguaje SQL (el así llamado Entry level o al máximo el Intermediate level), extendiéndolo de manera individual según la propia visión que cada cual tenga del mundo de las bases de datos.

Desarrollo en la Tesis

La base de datos que se utilizó para el almacenamiento de datos del sistema es PostgresSQL versión 8.1, que es la versión más actualizada de dicha base de datos. Esta versión cuenta con su propio manejador llamado pgAdmin III. Este DBMS hace más manejable la base de datos, ya que se hace de forma gráfica y no se tienen que ejecutar las sentencias desde una ventana de comandos.

Existe un archivo que contiene todas las tablas del sistema, para que en cualquier momento se pueda restaurar la base de datos. También existen archivos que contienen los respaldos de los datos que se encuentran en la base, principalmente de los datos referentes a los usuarios. Los datos de los usuarios son generados por el servidor mediante scripts que contienen código AWK y Shell, y que junto con comandos propios de PostgresSQL se hace la inserción de datos, de esta manera se pueden procesar datos en tiempo real.

Se hizo uso de la función “DISCTINCT“, para eliminar datos repetidos al momento de hacer una consulta; además de la función “to_char” que transforma cualquier tipo de dato a cadena.

Cada una de las tablas tiene una llave primaria que es un número secuencial y que se genera escribiendo la palabra “DEFAULT”, en la columna correspondiente al momento de insertar datos en alguna tabla. Lo anterior ayuda a no tener que saber el número antes de que se inserte un nuevo dato en una tabla.

Los estándares que se tomaron para los nombres de tablas y columnas de la base de datos del sistema, se encuentran descritos en el Anexo B: Estándares de Diseño de la Base de Datos.

4.3. Gráficas

Visual Engineering, Inc. fue fundado en 1983 para crear las herramientas de software y aplicaciones los sistemas operativos Unix. Hoy, la compañía desarrolla, vende y da soporte en el desarrollo de herramientas gráficas y aplicaciones para una gran variedad de plataformas.

La herramienta para el desarrollo de gráficas denominada "KavaChart" permite el desarrollo de software, de manera que con el simple hecho de tener una serie de números, estos se transforman en gráficas o charts con el mínimo esfuerzo. Un chart es una representación mediante gráficos de los movimientos de las acciones u otros tipos de títulos a lo largo del tiempo.

En la última versión disponible para descargar es KavaChart 5.0, tiene una aplicación gráfica llamada "KavaChartWizard"; esta ayuda al usuario a generar modelos de gráficas. La herramienta muestra todos los tipos de gráficas que se pueden realizar con la herramienta y después se seleccionan sus propiedades o características, de esta manera visual, es mucho más fácil construir una gráfica. Al finalizar la construcción de la gráfica, se genera un archivo ".properties", que contiene los parámetros con los que creará la gráfica.

Las gráficas, independientemente del tipo que sea, se compone de ejes (X Axis y Y Axis), un área donde se dibuja la gráfica (Plotarea), un fondo (Background), representación de los tipos de gráfica (DataRepresentation), una leyenda que indica la descripción de los datos que se graficaron (Legend) y datos (Datum y Dataset).

Los datos se componen de dos tipos Dataset y Datum. Los Datum guardan guardan información de una gráfica en particular, inclusive los valores, etiquetas e información de la gráfica. Los Dataset son utilizados para organizar Datum en series o grupos. También contienen una serie de nombe y las propiedades comunes de la gráfica, como lo son: la línea, el color o el relleno.

Desarrollo en la Tesis

Para cubrir la parte de generación de gráficas se hizo un JSP genérico, denominado Siu_Stat_JspGetGraphics.jsp, el cuál se encarga de crear los dos tipos de gráficas existentes en el sistema, que son gráficas de Pie (PieApp) y de Columnas (ColumnApp).

Los datos correspondientes a cada gráfica son obtenidos desde la base de datos, el JSP se encarga de capturarlos y procesarlos para generar la gráfica (según sea el caso de las dos posibles gráficas) y se encarga de cargar las propiedades de la gráfica a partir de dos archivos .properties, uno denominado pie.properties (para las gráficas de pie) y otro column.properties (para las gráficas de columna).

Todas las gráficas que se generan son almacenadas en un repositorio para imágenes, éste repositorio se encuentra en el servidor. Las gráficas son almacenadas por un tiempo determinado dentro del servidor y después son desechadas.

4.4. Formato y estilo

4.4.1. Javascript

Javascript es un lenguaje de programación utilizado para crear pequeños programitas encargados de realizar acciones dentro del ámbito de una página web.

Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado.

Con Javascript podemos crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

Javascript es el siguiente paso, después del HTML, que puede dar un programador de páginas Web, para mejorar sus páginas y la potencia de sus proyectos. Es un lenguaje de programación bastante sencillo y pensado para hacer las cosas con rapidez, a veces con ligereza.

Entre las acciones típicas que se pueden realizar en Javascript tenemos dos vertientes. Por un lado los efectos especiales sobre páginas Web, para crear contenidos dinámicos y elementos de la página que tengan movimiento, cambien de color o cualquier otro dinamismo. Por el otro, javascript nos permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que podemos crear páginas interactivas con programas como calculadoras, agendas, o tablas de cálculo.

Javascript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Además, Javascript pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente.

El lenguaje ha ido avanzando durante sus años de vida e incrementando sus capacidades. En un principio podía realizar muchas cosas en la página web, pero tenía pocas instrucciones para crear efectos especiales.

Las distintas versiones de Javascript:

- Javascript 1: nació con el Netscape 2.0 y soportaba gran cantidad de instrucciones y funciones, casi todas las que existen ahora ya se introdujeron en el primer estándar.
- Javascript 1.1: Es la versión de Javascript que se diseñó con la llegada de los navegadores 3.0. Implementaba poco más que su anterior versión, como por ejemplo el tratamiento de imágenes dinámicamente y la creación de arreglos.

- Javascript 1.2: La versión de los navegadores 4.0. Esta tiene como desventaja que es un poco distinta en plataformas Microsoft y Netscape, ya que ambos navegadores crecieron de distinto modo y estaban en plena lucha por el mercado.
- Javascript 1.3: Versión que implementan los navegadores 5.0. En esta versión se han limado algunas diferencias y asperezas entre los dos navegadores.
- Javascript 1.5: Versión actual.
- Por su parte, Microsoft también ha evolucionado hasta presentar su versión 5.5 de JScript (así llaman al javascript utilizado por los navegadores de Microsoft).

Desarrollo en la Tesis

El Javascript utilizado en el sistema, principalmente fue para verificar la validez de datos en las vistas. Lo anterior quiere decir que se hizo verificación de que se ingresaran caracteres válidos, en dado caso de que se trate de ingresar un caracter inválido, simplemente el javascript evita que se pueda escribir el mismo. En el caso de dar de alta un usuario, donde al momento de dar clic al botón enviar, se verifica que todos los campos tengan datos, de lo contrario manda un mensaje de error; esto con el fin de que sean ingresados de manera correcta, todos lo datos requeridos para mantener la integridad en la base de datos. Esto le quita trabajo de al servidor y se la deja al cliente.

Se tiene un script que en el caso de las vistas que contienen Checkbox, se valide que al menos sea marcado uno, esto evita que se haga una actualización en la base de datos, sin que existan datos que actualizar.

Otro código de Javascript que se utilizó, fue para generar un menú dinámico en la Zona de Administración del SIU, esto con el fin de que fuera más vistoso y se tuviera un menú desplegable con varias opciones. También existe un script en vistas donde se pueden realizar varias operaciones, éste script se encarga de desplegar el formulario correspondiente a la opción seleccionada de un radiogroup.

Para la verificación de contraseñas al momento de que se quiere cambiar alguna, existe un javascript que se encarga de verificar que las contraseñas sean iguales antes de que se produzca el envío de información a la base de datos.

4.4.2. Hojas de estilo

CSS (Cascading Style Sheets, u Hojas de Estilo en Cascada) es la tecnología desarrollada por el World Wide Web Consortium (W3C) con el fin de separar la estructura de la presentación. A pesar de que la recomendación oficial del grupo de trabajo de la W3C ya había alcanzado la estabilidad requerida para que fuera soportada por los principales navegadores comerciales, como Netscape e Internet Explorer, tan tempranamente como en el año 1998, la situación de entonces, comúnmente conocida como la "guerra de los navegadores", hacía que los intereses comerciales de las dos compañías en lucha por el mercado de usuarios de Internet se interpusieran en el camino de las CSS.

Netscape 4 e Internet Explorer 4 incorporaron parcialmente un soporte a esta recomendación, pero ésta dejaba mucho que desear, especialmente en Netscape 4, por lo que no era plausible la incorporación de las CSS en el diseño de sitios salvo en un muy mínimo número de características y esto, aún con reservas, puesto que la manera de tratar los estándares era muy diferente y por eso ni aún así se podía asegurar una visualización correcta de la misma hoja de estilos.

La situación, hoy, es muy diferente; Netscape 4 acaparaba, según estadísticas, el 80% de usuarios, al menos en Estados Unidos. Desde el lanzamiento de Internet Explorer 5 que tuvo problemas al principio, luego solucionados mayormente con la versión 5.5, esta situación cambió radicalmente y hoy es éste el navegador más usado.

Ahora ya es posible utilizar ciertas posibilidades más amplias de las CSS, como el control de otras características gráficas tales como imágenes y colores de fondo, márgenes exactos y bordes, que incluye frecuentemente tablas anidadas y complicados algoritmos de combinación de celdas, características que hacen al archivo muy pesado para descargar, porque inundan el código con la extensa serie de etiquetas requeridas.

Los beneficios de usar CSS son dobles. Por un lado, evitamos hacer a los archivos demasiado pesados (excluyendo el largo código requerido para las tablas anidadas y el añadido de características gráficas), y definimos el "estilo visual" de un sitio entero sin necesidad de hacerlo etiqueta por etiqueta, para cada una de las páginas.
Por otro, se trabaja con estándares, y separamos hasta cierto punto la estructura (el código) de la presentación, logrando una manera más nítida de trabajar, y lo que es más: en un sencillo documento CSS. Cualquier cambio hecho a un estilo CSS, se reflejará en todos los elementos que sean referidos a éste, automáticamente, con sólo editar un sencillo documento CSS.

Desarrollo en la Tesis

Las hojas de estilo se encargaron de darle presentación a todas las vistas, aunque el estilo aplicado a cada Zona se hizo diferente. Para la Zona de Vistas se definieron colores grises, esto con el fin de dar un aspecto serio.

En la Zona de Vistas, la CSS que define el menú principal se hizo de tal manera que cada liga pareciera un botón que cambia de color al momento de que se pasa el mouse sobre él. Se usó letra tipo Arial en color blanco; cada uno de los campos y etiquetas se encuentra dentro de una tabla, que para las etiquetas tiene como fondo un color gris y para los campos tiene un fondo blanco.

Para la Zona de Administración la CSS define un tipo de letra Arial; tablas van en un color azul claro, y el color de las etiquetas es un color azul índigo. El color de las letras dentro de los campos es azul índigo también.

5. Pruebas

5.1. Servidor Web (Concurrencia)

Para realizar las pruebas de concurrencia del Servidor Web que contiene el sistema, se utilizó el software Webserver Stress Tool versión 7 de la empresa Paessler, el cuál es posible descargar de la página http://www.paessler.com/webstress/download.

Se realizó la prueba a la raíz del sistema, ya que esta parte es la que se encuentra expuesta a tener errores al momento de solicitar consultas. El sistema se va a probar simulando varios usuarios haciendo consultas en el sistema; para esto se configuró el Webserver Stress Tool de la siguiente manera:

- Se va a ejecutar la prueba con carga constante hasta que cada uno de los usuarios haya generado un número específico de pulsaciones.
- Se va a tener un número de 10 usuarios simultáneos.
- El tiempo entra cada clic será aleatorio.

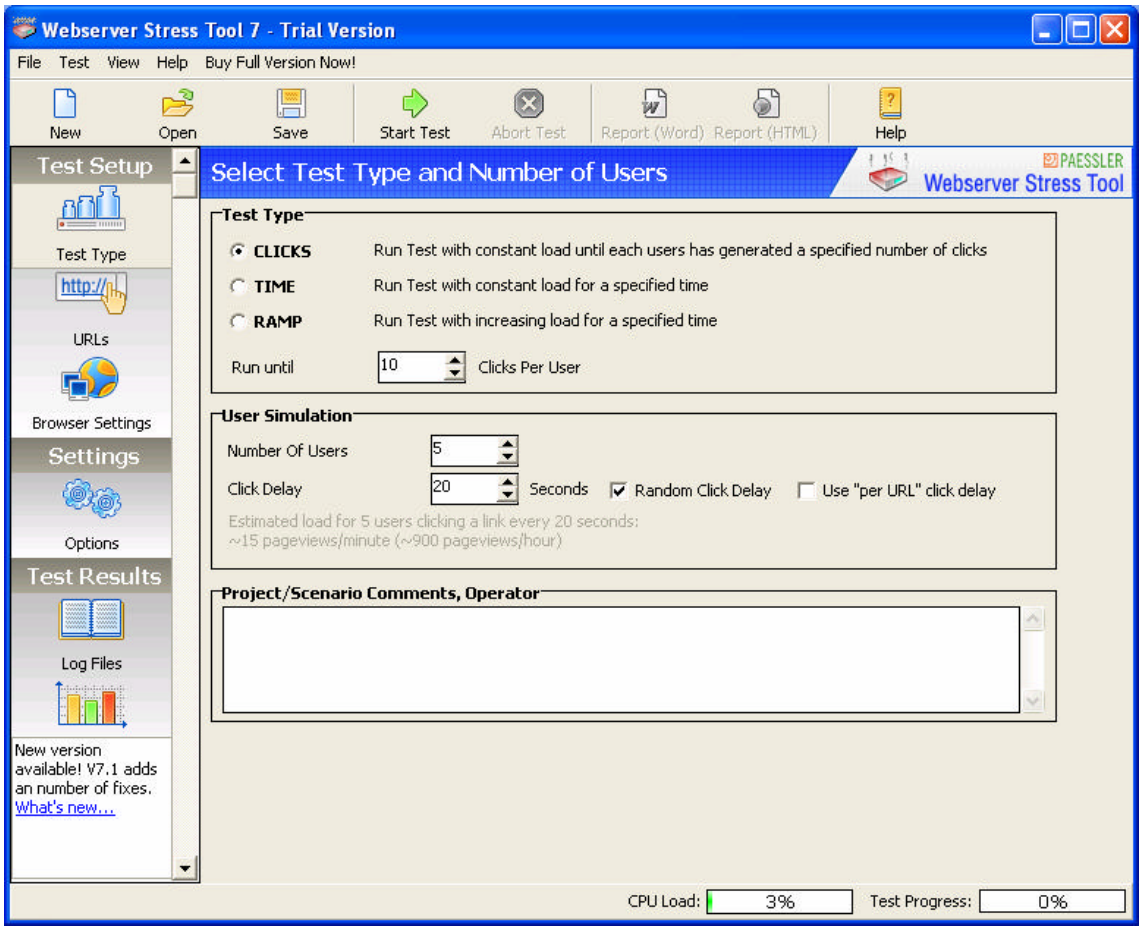


Figura 5.1 Configuración de parámetros del Webserver Stress Tool.

Los resultados de la prueba se muestran a continuación.

Gráfica de tiempos de protocolo: Consiste en diferentes estados. Primero el URL es convertido a una dirección IP usando un DNS (time for DNS), después un puerto IP es abierto en el servidor y el cliente manda una petición (time to connect). El servidor responde la petición (time to first byte) y manda todos los datos. Cuando todos los datos han sido enviados, la petición finaliza (clic time).

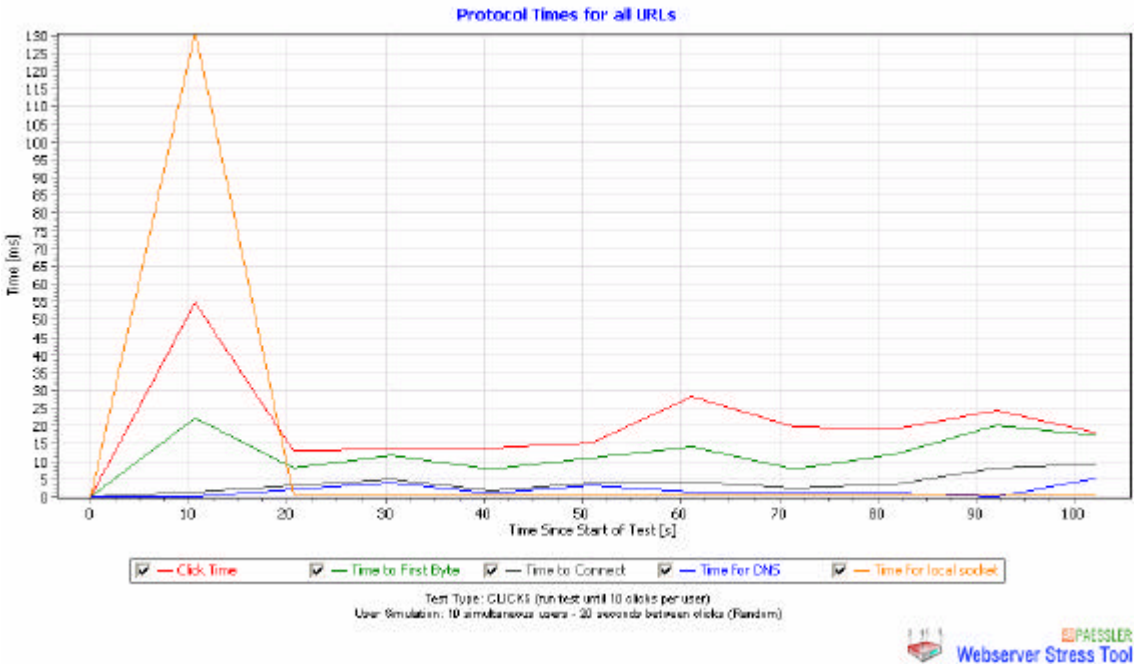


Figura 5.2 Tiempos de Protocolo para cada URL.

Gráfica de ancho de banda del servidor y usuario: Despliega el ancho de banda que en el servidor, así como el ancho de banda que están experimentando los usuarios simulados.

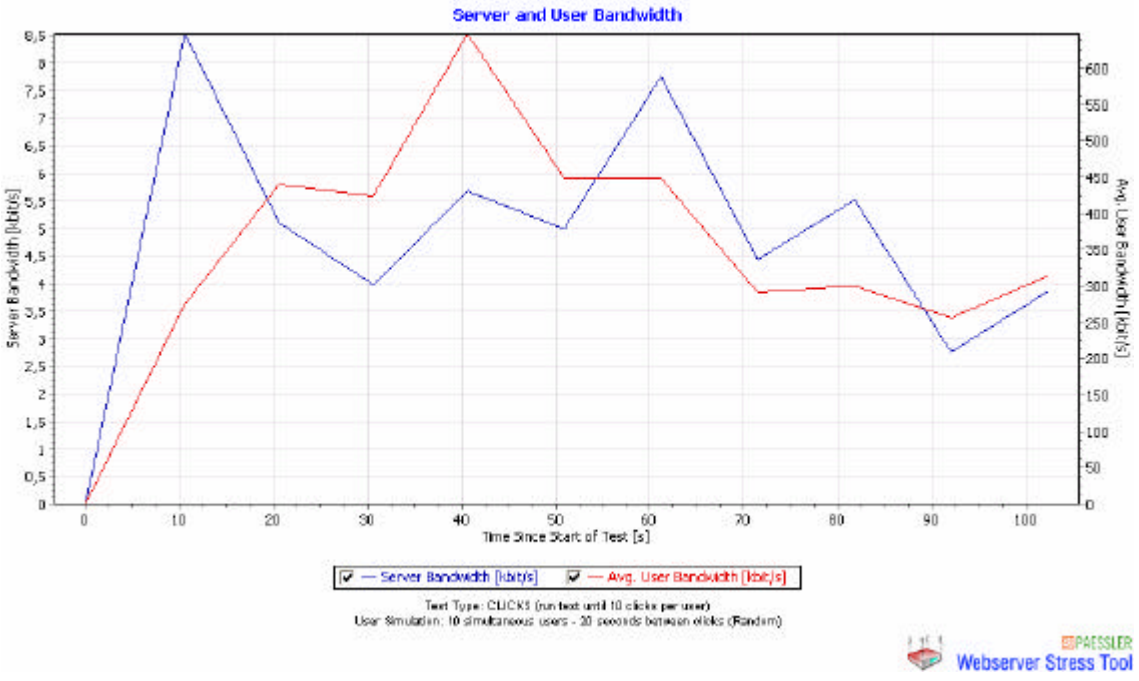


Figura 5.3 Ancho de Banda del Servidor y Usuario.

Gráfica de peticiones abiertas y transferencia de datos: Muestra el número de peticiones abiertas, así como el envío y recepción de peticiones comparadas con el tráfico en la red.

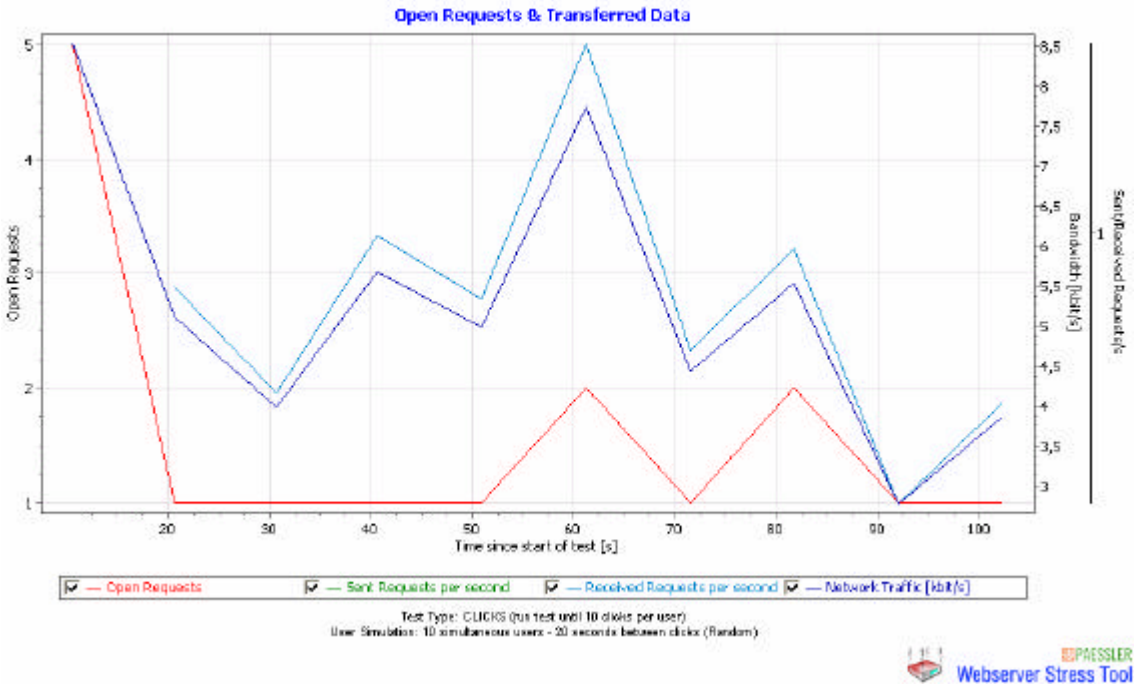


Figura 5.4 Peticiones abiertas y transferencia de datos.

Gráfica de instantes de peticiones realizadas: Por cada petición simulada por el Webserver stress tool enviada al servidor, se crea una flecha.

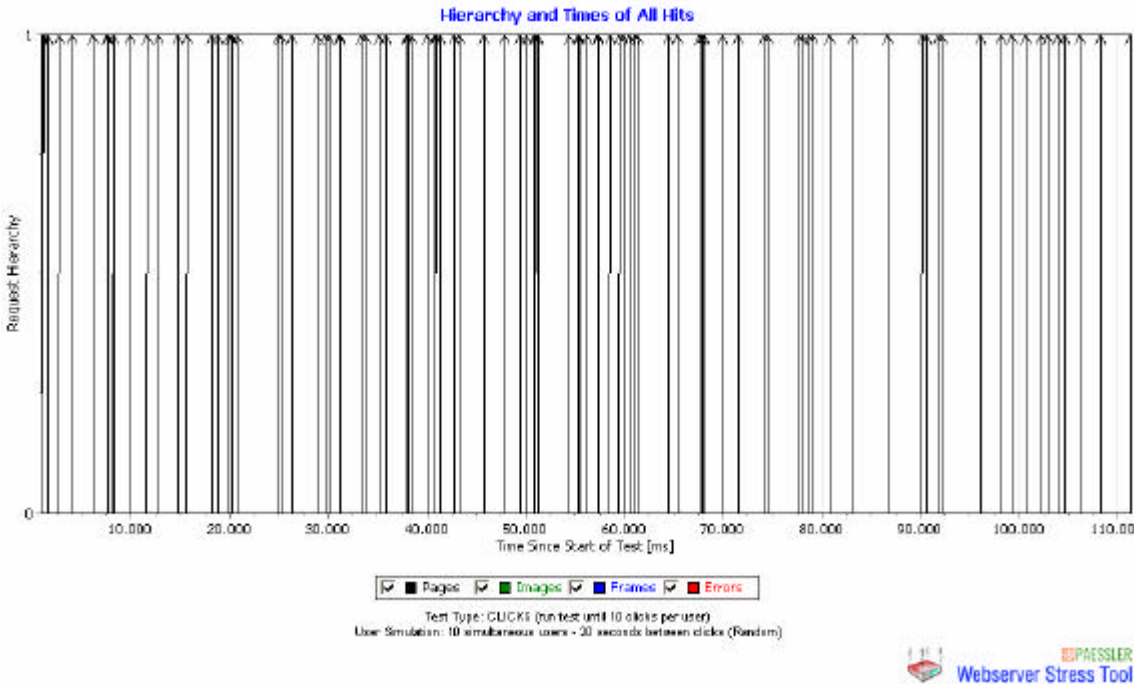


Figura 5.5 Instantes de peticiones realizadas.

Gráfica de tiempo de espera después de la petición: En esta gráfica se considera un margen de 10 segundos como máximo para cada respuesta, después de haber realizado una petición.

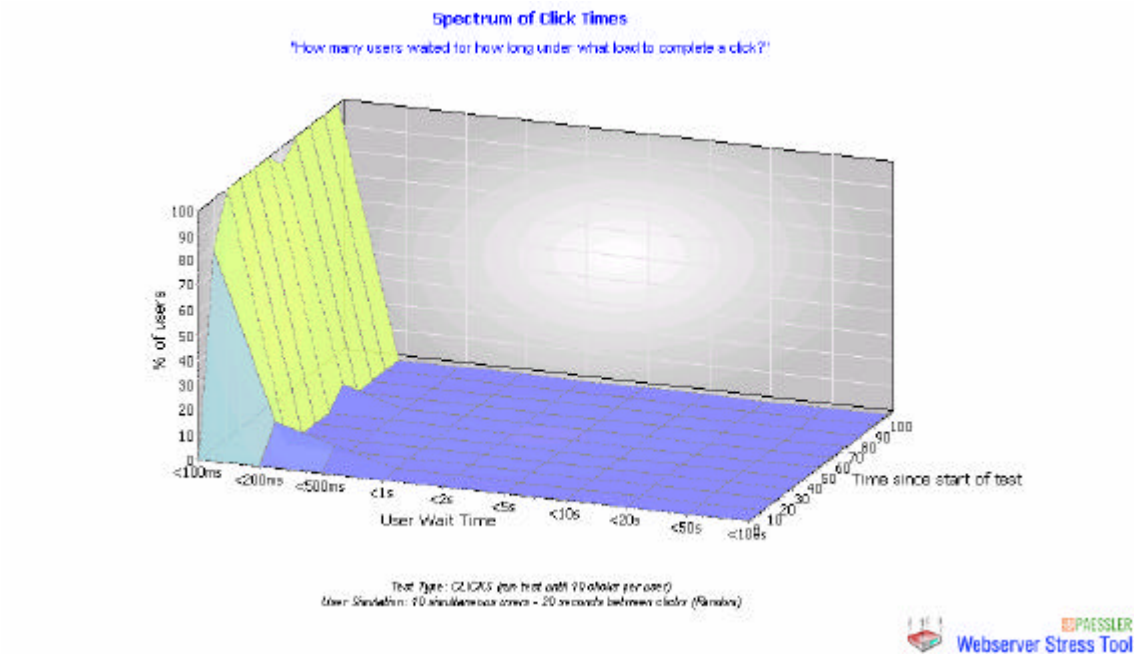


Figura 5.6 Tiempo de espera después de la petición.

Gráfica de instantes de petición y errores: Esta gráfica muestra el tiempo promedio, así como los errores que los usuario que se simularon están experimentando cuando consultan una página.

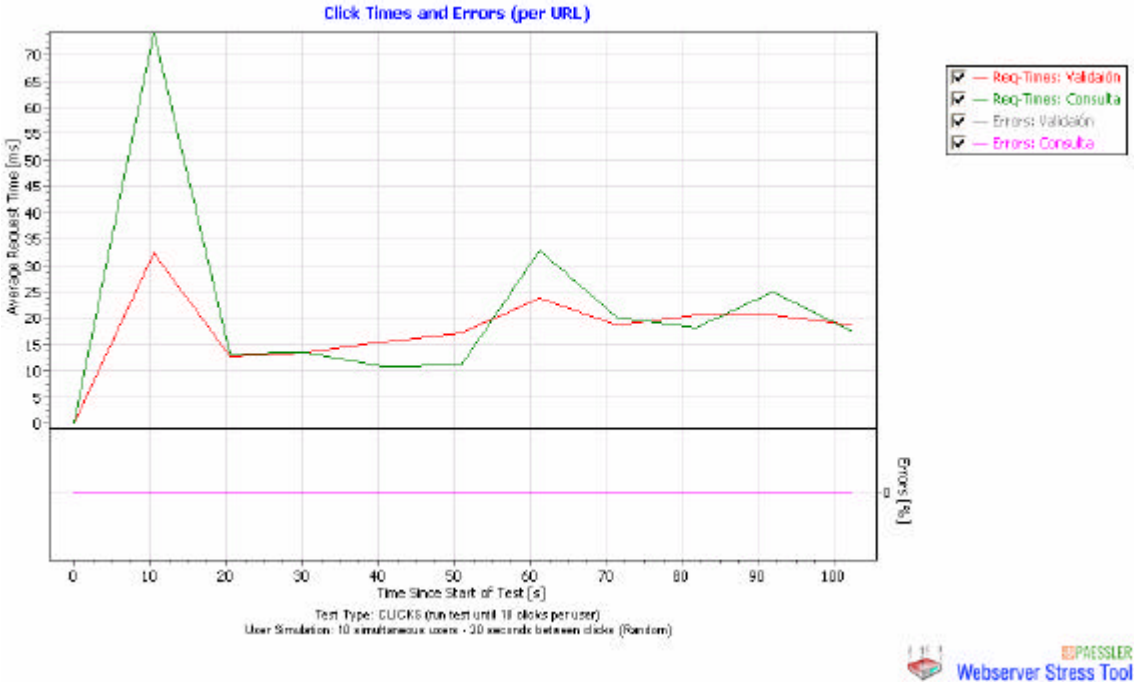


Figura 5.7 Instantes de petición y errores.

Summary Log

** Test Logfile by Webserver Stress Tool 7.0.3.189 Trial Version **

Test run on 21/12/2006 19:37:07

** Project and Scenario Comments, Operator **

Results of period #1 (from 1 sec to 11 sec):
****
Completed Clicks: 15 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 55 ms
Successful clicks per Second: 1,50 (equals 5.400,29 Clicks per Hour)

Results of period #2 (from 11 sec to 21 sec):
****
Completed Clicks: 9 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 13 ms
Successful clicks per Second: 0,90 (equals 3.235,22 Clicks per Hour)

Results of period #3 (from 21 sec to 31 sec):
****
Completed Clicks: 7 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 14 ms
Successful clicks per Second: 0,70 (equals 2.516,01 Clicks per Hour)

Results of period #4 (from 31 sec to 41 sec):
****
Completed Clicks: 10 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 14 ms
Successful clicks per Second: 1,00 (equals 3.588,74 Clicks per Hour)

Results of period #5 (from 41 sec to 51 sec):
****
Completed Clicks: 9 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 15 ms
Successful clicks per Second: 0,88 (equals 3.153,37 Clicks per Hour)

Results of period #6 (from 51 sec to 62 sec):
****
Completed Clicks: 14 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 28 ms
Successful clicks per Second: 1,36 (equals 4.899,22 Clicks per Hour)

Results of period #7 (from 62 sec to 72 sec):
****
Completed Clicks: 8 with 0 Errors (=0,00%)

Average Click Time for 10 Users: 20 ms
Successful clicks per Second: 0,78 (equals 2.802,86 Clicks per Hour)

Results of period #8 (from 72 sec to 82 sec):
****
Completed Clicks: 10 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 19 ms
Successful clicks per Second: 0,97 (equals 3.499,48 Clicks per Hour)

Results of period #9 (from 82 sec to 92 sec):
****
Completed Clicks: 5 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 24 ms
Successful clicks per Second: 0,49 (equals 1.751,82 Clicks per Hour)

Results of period #10 (from 92 sec to 103 sec):
****
Completed Clicks: 7 with 0 Errors (=0,00%)
Average Click Time for 10 Users: 18 ms
Successful clicks per Second: 0,68 (equals 2.449,09 Clicks per Hour)

Results of complete test
****

** Results per URL for complete test **

URL#1 (Validaión): Average Click Time 20 ms, 47 Clicks, 0 Errors
URL#2 (Consulta): Average Click Time 29 ms, 47 Clicks, 0 Errors

Total Number of Clicks: 94 (0 Errors)
Average Click Time of all URLs: 24 ms

Glosario:

Click: Es una simulación del click de un mouse cuando manda una petición.
Request: Es una petición HTTP enviada al servidor para recibir una respuesta.
Hit: Una respuesta HTTP completa (se envía la petición al servidor y éste contesta).
Time for DNS: Es el tiempo que tomo resolver un URL.
Time to connect: Tiempo que tarda en hacerse una conexión al servidor.
Time to first byte (TFB): Tiempo que tarda entre en envío de la petición y el primer byte enviado por el servidor.
Click Time: El tiempo que le toma a un usuario esperar desde que da el click, hasta que finaliza.
User Bandwidth: El ancho de banda que un usuario puede conseguir.
Sent Requests: Número de peticiones enviadas al servidor en un periodo de tiempo.
Received Requests: Número de respuestas recibidas del servidor durante un periodo de tiempo.

URL#	Name	URL
1	Validaión	http://132,248,59,12:8080/siu/siu.administration.servlet.Siu_Admin_ServletValidateAuth
2	Consulta	http://132,248,59,12:8080/siu/siu.students.servlet.Siu_Stud_ServletGetStatistics

Tabla 5.1 URLs probadas.

User No.	Clicks	Hits	Errores	Avg. Click Time [ms]	Bytes	kbit/s
1	10	10	0	17	7.110	328,17
2	10	10	0	29	7.110	198,02
3	10	10	0	35	7.110	163,59
4	10	10	0	40	7.110	142,76
5	10	10	0	15	7.110	369,95
6	10	10	0	29	7.110	198,76
7	10	10	0	16	7.110	358,35
8	10	10	0	27	7.110	214,36
9	10	10	0	15	7.110	371,66
10	10	10	0	18	7.110	316,96

Tabla 5.2 Resultado por usuario.

URL No.	Name	Clicks	Errors	Errors [%]	Time Spent [ms]	Avg. Click Time [ms]
1	Validación	47	0	0	935	20
2	Consulta	47	0	0	1.364	29

Tabla 5.3 Resultados por URL.

** Test Logfile by Webserver Stress Tool 7.0.3.189 Trial Version **
© 1998-2005 Paessler GmbH, http://www.paessler.com

Test run on 21/12/2006 19:37:07

** Project and Scenario Comments, Operator **

** Test Setup **

Test Type: CLICKS (run test until 10 clicks per user)
User Simulation: 10 simultaneous users - 20 seconds between clicks (Random)

Logging Period: Log every 10 seconds

** URLs **
URL Sequencing: Users always click the same URL (to spreads load evenly on all URLs, set number of users to a multiple of the number of URLs!)
2 URLs
URL#1: POST
http://127.0.0.1:8080/siu/siu.administration.servlet.Siu_Admin_ServletValidateAuth
POSTDATA=login=admin&pass=1dm3n Click Delay=

URL#2: POST
http://127.0.0.1:8080/siu/siu.students.servlet.Siu_Stud_ServletGetStatistics
POSTDATA=check_semestre=T&menu_semestre=Decimo&txtFech=&txtFech2=&check_carrera=T&menu_carrera=Computacion&menu_horaini=7%3A00%3A00&menu_horafin=7%3A00%3A00 Click Delay=

** Browser Settings **
Browser Simulation:
User Agent: * IE 6.0 *
HTTP Request Timeout: 120 s

Recursive Browsing / HTML Parsing:

** Options **
Advanced Settings:
Logging:
Write detailed log(s)
Timer: not enabled

Using Local IPs:
132.248.59.12
Advanced Data Merging Features:

** Client System **
Windows XP V5.1 (Build 2600) Service Pack 2, CPU Proc. Type 586 (Rev. 771) at 3.199 MHz,
295 MB available RAM of 527 MB total physical RAM, 1863 MB available pagefile, 3328 MB free disk space on C:

Test is starting

Creating Users...
Pre-Requesting URLs once... (for DNS/DLL/SYSTEM initialization)
Pre-Requesting: HEAD Request to URL #1
Pre-Requesting: HEAD Request to URL #2
Pre-Requests done
Test preparations done

Master Controller is started
Switching to 10 users.

Results of period #1 (from 1 sec to 11 sec):
****
Analyzed Time span of this period: 9.999 msec
Sent/Received Requests: 15 / 15 (=0,00% not answered in this period)
Completed Hits: 15 (including images, frames etc.)
Completed Clicks: 15 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
80,00% of All Users waited <100ms
13,33% of All Users waited <200ms
6,67% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 131 ms
Average DNS Time for 10 Users: 0 ms
Average Time to Connect for 10 Users: 1 ms
Average Time to First Byte for 10 Users: 22 ms
Average Click Time for 10 Users: 55 ms

Hits per Second: 1,50 (equals 5.400,29 Hits per Hour)
Successful clicks per Second: 1,50 (equals 5.400,29 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 32 ms, 7 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 74 ms, 8 Clicks, 0 Errors,
Average Click Time of all URLs: 55 ms

Resulting page statuscodes for this period:
15x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 275,37 kbit/s
Total Bytes: 10.665 Bytes (11 kByte) (Throughput ~9 kbit/sec)

Results of period #2 (from 11 sec to 21 sec):
****
Analyzed Time span of this period: 10.015 msec
Sent/Received Requests: 9 / 9 (=0,00% not answered in this period)
Completed Hits: 9 (including images, frames etc.)
Completed Clicks: 9 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
100,00% of All Users waited <100ms
0,00% of All Users waited <200ms
0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms
Average DNS Time for 10 Users: 2 ms
Average Time to Connect for 10 Users: 3 ms
Average Time to First Byte for 10 Users: 8 ms
Average Click Time for 10 Users: 13 ms

Hits per Second: 0,90 (equals 3.235,22 Hits per Hour)
Successful clicks per Second: 0,90 (equals 3.235,22 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 13 ms, 4 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 13 ms, 5 Clicks, 0 Errors,
Average Click Time of all URLs: 13 ms

Resulting page statuscodes for this period:
9x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 439,25 kbit/s

Total Bytes: 6.399 Bytes (6 kByte) (Throughput ~5 kbit/sec)

Results of period #3 (from 21 sec to 31 sec):
****
Analyzed Time span of this period: 10.016 msec
Sent/Received Requests: 7 / 7 (=0,00% not answered in this period)
Completed Hits: 7 (including images, frames etc.)
Completed Clicks: 7 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
100,00% of All Users waited <100ms
0,00% of All Users waited <200ms
0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms
Average DNS Time for 10 Users: 4 ms
Average Time to Connect for 10 Users: 5 ms
Average Time to First Byte for 10 Users: 12 ms
Average Click Time for 10 Users: 14 ms

Hits per Second: 0,70 (equals 2.516,01 Hits per Hour)
Successful clicks per Second: 0,70 (equals 2.516,01 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 13 ms, 5 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 13 ms, 2 Clicks, 0 Errors,
Average Click Time of all URLs: 13 ms

Resulting page statuscodes for this period:
7x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 423,36 kbit/s
Total Bytes: 4.977 Bytes (5 kByte) (Throughput ~4 kbit/sec)

Results of period #4 (from 31 sec to 41 sec):
****
Analyzed Time span of this period: 10.031 msec
Sent/Received Requests: 10 / 10 (=0,00% not answered in this period)
Completed Hits: 10 (including images, frames etc.)
Completed Clicks: 10 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
100,00% of All Users waited <100ms
0,00% of All Users waited <200ms
0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms
Average DNS Time for 10 Users: 1 ms
Average Time to Connect for 10 Users: 2 ms
Average Time to First Byte for 10 Users: 8 ms
Average Click Time for 10 Users: 14 ms

Hits per Second: 1,00 (equals 3.588,74 Hits per Hour)
Successful clicks per Second: 1,00 (equals 3.588,74 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 15 ms, 7 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 11 ms, 3 Clicks, 0 Errors,
Average Click Time of all URLs: 14 ms

Resulting page statuscodes for this period:
10x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 646,83 kbit/s
Total Bytes: 7.110 Bytes (7 kByte) (Throughput ~6 kbit/sec)

Results of period #5 (from 41 sec to 51 sec):
****
Analyzed Time span of this period: 10.275 msec
Sent/Received Requests: 9 / 9 (=0,00% not answered in this period)
Completed Hits: 9 (including images, frames etc.)

Completed Clicks: 9 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
100,00% of All Users waited <100ms
0,00% of All Users waited <200ms
0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms
Average DNS Time for 10 Users: 3 ms
Average Time to Connect for 10 Users: 4 ms
Average Time to First Byte for 10 Users: 11 ms
Average Click Time for 10 Users: 15 ms

Hits per Second: 0,88 (equals 3.153,37 Hits per Hour)
Successful clicks per Second: 0,88 (equals 3.153,37 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 17 ms, 6 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 11 ms, 3 Clicks, 0 Errors,
Average Click Time of all URLs: 15 ms

Resulting page statuscodes for this period:
9x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 447,59 kbit/s
Total Bytes: 6.399 Bytes (6 kByte) (Throughput ~5 kbit/sec)

Results of period #6 (from 51 sec to 62 sec):
****

Analyzed Time span of this period: 10.287 msec
Sent/Received Requests: 14 / 14 (=0,00% not answered in this period)
Completed Hits: 14 (including images, frames etc.)
Completed Clicks: 14 with 0 Errors (=0,00%)

Results for Images for this period:

Image Hits: 0 with 0 Errors

Spectrum of Click Times
92,86% of All Users waited <100ms
7,14% of All Users waited <200ms
0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms
Average DNS Time for 10 Users: 2 ms
Average Time to Connect for 10 Users: 4 ms
Average Time to First Byte for 10 Users: 14 ms
Average Click Time for 10 Users: 28 ms

Hits per Second: 1,36 (equals 4.899,22 Hits per Hour)
Successful clicks per Second: 1,36 (equals 4.899,22 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 24 ms, 7 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 33 ms, 7 Clicks, 0 Errors,
Average Click Time of all URLs: 28 ms

Resulting page statuscodes for this period:
14x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 447,98 kbit/s
Total Bytes: 9.954 Bytes (10 kByte) (Throughput ~8 kbit/sec)

Results of period #7 (from 62 sec to 72 sec):
****
Analyzed Time span of this period: 10.275 msec
Sent/Received Requests: 8 / 8 (=0,00% not answered in this period)
Completed Hits: 8 (including images, frames etc.)
Completed Clicks: 8 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
100,00% of All Users waited <100ms
0,00% of All Users waited <200ms

0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms
Average DNS Time for 10 Users: 2 ms
Average Time to Connect for 10 Users: 2 ms
Average Time to First Byte for 10 Users: 8 ms
Average Click Time for 10 Users: 20 ms

Hits per Second: 0,78 (equals 2.802,86 Hits per Hour)
Successful clicks per Second: 0,78 (equals 2.802,86 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 19 ms, 2 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 20 ms, 6 Clicks, 0 Errors,
Average Click Time of all URLs: 20 ms

Resulting page statuscodes for this period:
8x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 291,24 kbit/s
Total Bytes: 5.688 Bytes (6 kByte) (Throughput ~4 kbit/sec)

Results of period #8 (from 72 sec to 82 sec):
****
Analyzed Time span of this period: 10.287 msec
Sent/Received Requests: 10 / 10 (=0,00% not answered in this period)
Completed Hits: 10 (including images, frames etc.)
Completed Clicks: 10 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
100,00% of All Users waited <100ms
0,00% of All Users waited <200ms
0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s

0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms
Average DNS Time for 10 Users: 1 ms
Average Time to Connect for 10 Users: 4 ms
Average Time to First Byte for 10 Users: 12 ms
Average Click Time for 10 Users: 19 ms

Hits per Second: 0,97 (equals 3.499,48 Hits per Hour)
Successful clicks per Second: 0,97 (equals 3.499,48 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 21 ms, 4 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 18 ms, 6 Clicks, 0 Errors,
Average Click Time of all URLs: 19 ms

Resulting page statuscodes for this period:
10x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 299,93 kbit/s
Total Bytes: 7.110 Bytes (7 kByte) (Throughput ~6 kbit/sec)

Results of period #9 (from 82 sec to 92 sec):
****
Analyzed Time span of this period: 10.275 msec
Sent/Received Requests: 5 / 5 (=0,00% not answered in this period)
Completed Hits: 5 (including images, frames etc.)
Completed Clicks: 5 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
100,00% of All Users waited <100ms
0,00% of All Users waited <200ms
0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms
Average DNS Time for 10 Users: 0 ms
Average Time to Connect for 10 Users: 8 ms
Average Time to First Byte for 10 Users: 20 ms
Average Click Time for 10 Users: 24 ms

Hits per Second: 0,49 (equals 1.751,82 Hits per Hour)
Successful clicks per Second: 0,49 (equals 1.751,82 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 21 ms, 1 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 25 ms, 4 Clicks, 0 Errors,
Average Click Time of all URLs: 24 ms

Resulting page statuscodes for this period:
5x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 256,99 kbit/s
Total Bytes: 3.555 Bytes (4 kByte) (Throughput ~3 kbit/sec)

Results of period #10 (from 92 sec to 103 sec):
****
Analyzed Time span of this period: 10.290 msec
Sent/Received Requests: 7 / 7 (=0,00% not answered in this period)
Completed Hits: 7 (including images, frames etc.)
Completed Clicks: 7 with 0 Errors (=0,00%)

Results for Images for this period:
Image Hits: 0 with 0 Errors

Spectrum of Click Times
100,00% of All Users waited <100ms
0,00% of All Users waited <200ms
0,00% of All Users waited <500ms
0,00% of All Users waited <1s
0,00% of All Users waited <2s
0,00% of All Users waited <5s
0,00% of All Users waited <10s
0,00% of All Users waited <20s
0,00% of All Users waited <50s
0,00% of All Users waited <100s

Measured Times:
Average Time to Create Local Socket for 10 Users: 1 ms

Average DNS Time for 10 Users: 5 ms
Average Time to Connect for 10 Users: 9 ms
Average Time to First Byte for 10 Users: 17 ms
Average Click Time for 10 Users: 18 ms

Hits per Second: 0,68 (equals 2.449,09 Hits per Hour)
Successful clicks per Second: 0,68 (equals 2.449,09 Clicks per Hour)

Results per URL for this Period:
URL#1 (Validaión): Average Click Time 19 ms, 4 Clicks, 0 Errors,
URL#2 (Consulta): Average Click Time 17 ms, 3 Clicks, 0 Errors,
Average Click Time of all URLs: 18 ms

Resulting page statuscodes for this period:
7x "200" (=OK)

Datavolume and Bandwidth for this period:
Average User Bandwidth: 314,58 kbit/s
Total Bytes: 4.977 Bytes (5 kByte) (Throughput ~4 kbit/sec)

****
Test Done, now waiting for simulated users to stop surfing
****
Waiting for 5 seconds for users to cool down and stop surfing...
All surfers passed away...
Master Controller is halted

Results of complete test
****

** Results per User for complete test **

User #1: Avg. Click Time: 17,33 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 328,17 kbit/s
User #2: Avg. Click Time: 28,72 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 198,02 kbit/s
User #3: Avg. Click Time: 34,77 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 163,59 kbit/s
User #4: Avg. Click Time: 39,84 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 142,76 kbit/s
User #5: Avg. Click Time: 15,37 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 369,95 kbit/s
User #6: Avg. Click Time: 28,62 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 198,76 kbit/s
User #7: Avg. Click Time: 15,87 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 358,35 kbit/s

User #8: Avg. Click Time: 26,53 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 214,36 kbit/s
User #9: Avg. Click Time: 15,30 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 371,66 kbit/s
User #10: Avg. Click Time: 17,95 ms, 10 Clicks, 10 Hits, 0 Errors, 7.110 Bytes, 316,96 kbit/s

** Results per URL for complete test **

URL#1 (Validaión): Average Click Time 20 ms, 47 Clicks, 0 Errors
URL#2 (Consulta): Average Click Time 29 ms, 47 Clicks, 0 Errors

Total Number of Clicks: 94 (0 Errors)
Average Click Time of all URLs: 24 ms

Cleaning up
Done

5.2. Casos de Prueba

Ingreso a Zona de Vistas.

Funcionalidad o Característica:

Ingresando los datos correctos, es posible tener acceso a la Zona de Vistas para realizar consultas.

Secuencia:

- El usuario solicita la página de validación.
- El usuario ingresa su login y contraseña.
- Clic en botón “Enviar”.
- El usuario entra a la Zona de Vistas.

Prerrequisitos:

- login = admin.
- contraseña = 1dm3n

Resultados de la prueba:

El usuario es validado y ve, de acuerdo con el perfil que tenga, los módulos en los cuáles puede hacer consultas.

Evaluación de resultados:

El procedimiento es exitoso si se proporcionan los datos correctos. Si al momento de dar clic al botón de “Enviar”, algún campo está vacío manda un mensaje de: ”Ingresa datos en el campo …”. En dado caso de que algún dato fuera incorrecto, el sistema manda un mensaje de error “ERROR: No existe un usuario con los datos proporcionados”.

Pantallas**:**

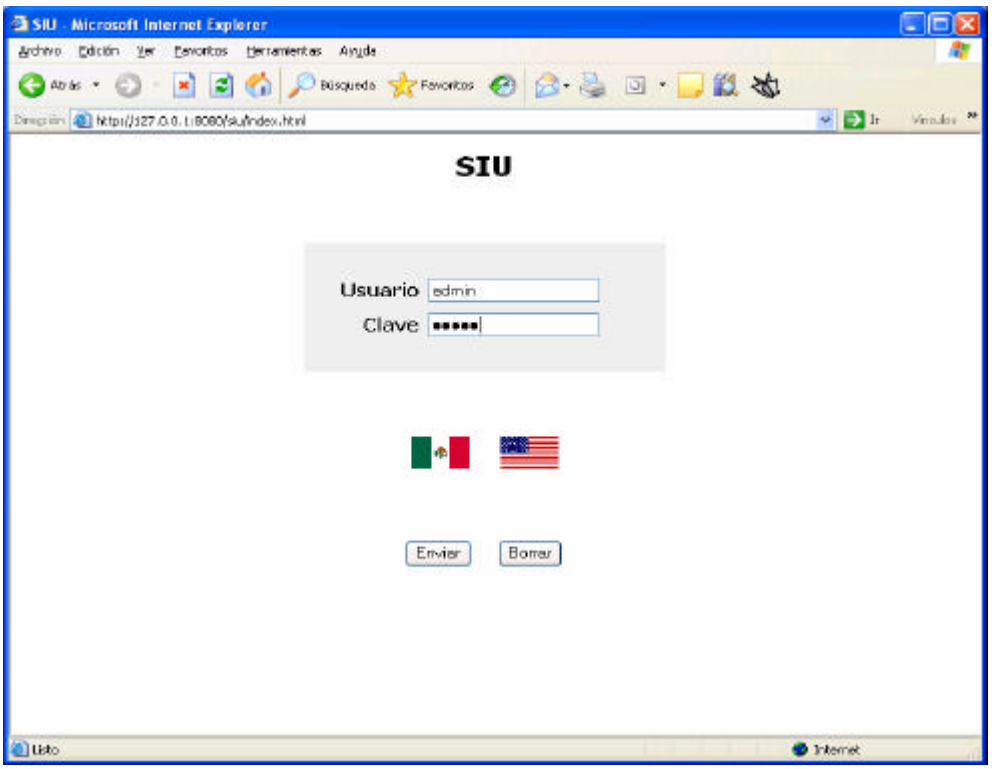


Figura 5.8 Validación de usuario y contraseña.

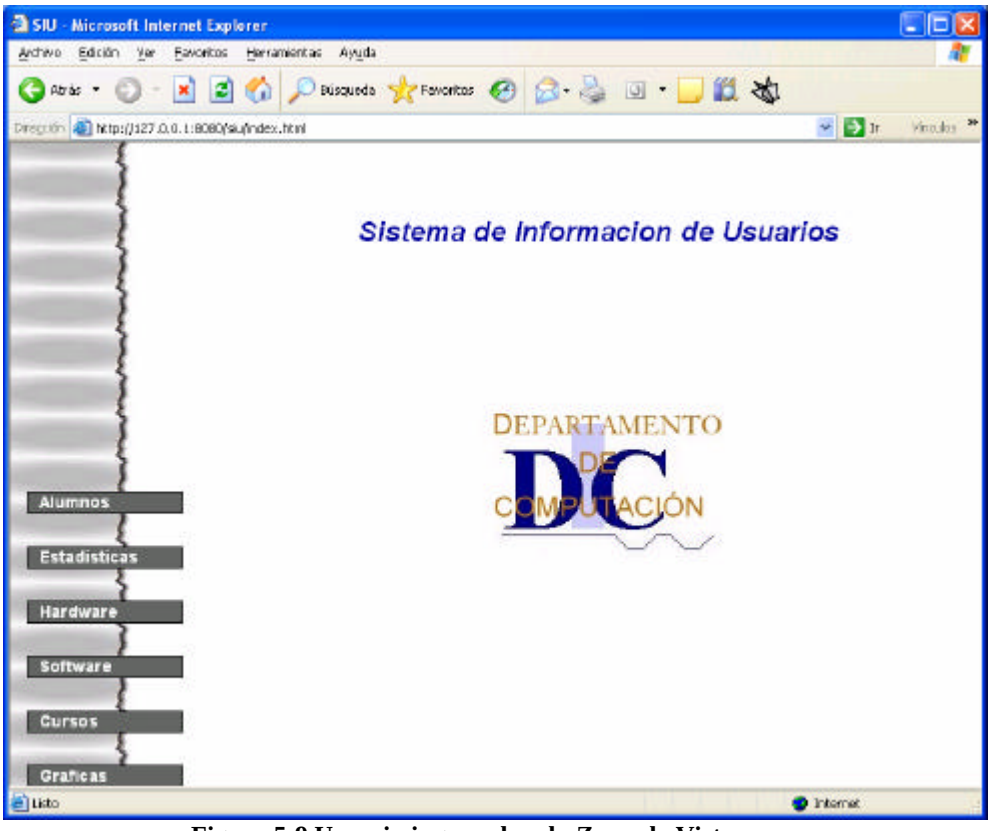


Figura 5.9 Usuario ingresado a la Zona de Vistas.

Consulta de Historial de Alumno.

Funcionalidad o Característica:

Se realiza la consulta del historial del alumno solicitado.

Secuencia:

- El usuario solicita la página de validación.
- El usuario ingresa su login y contraseña.
- Clic en botón “Enviar”.
- El usuario entra a la Zona de Vistas.
- El usuario selecciona la opción “Alumnos”.
- El usuario introduce el login del alumno a buscar.
- Respuesta del sistema.

Prerrequisitos:

- Haber entrado al sistema al ingresar login y contraseña correctos.
- alumno = gkno

Resultados de la prueba:

La consulta muestra el historial del alumno que se seleccionó.

Evaluación de resultados:

El procedimiento es exitoso al proporcionar un login de alumno correctamente. En dado caso de que se hiciera una consulta de un usuario que no existe, el sistema manda el mensaje “ERROR: No existe un usuario con ese login”.

Pantallas**:**

Habiendo pasado por la parte de validación, las pantallas obtenidas son:

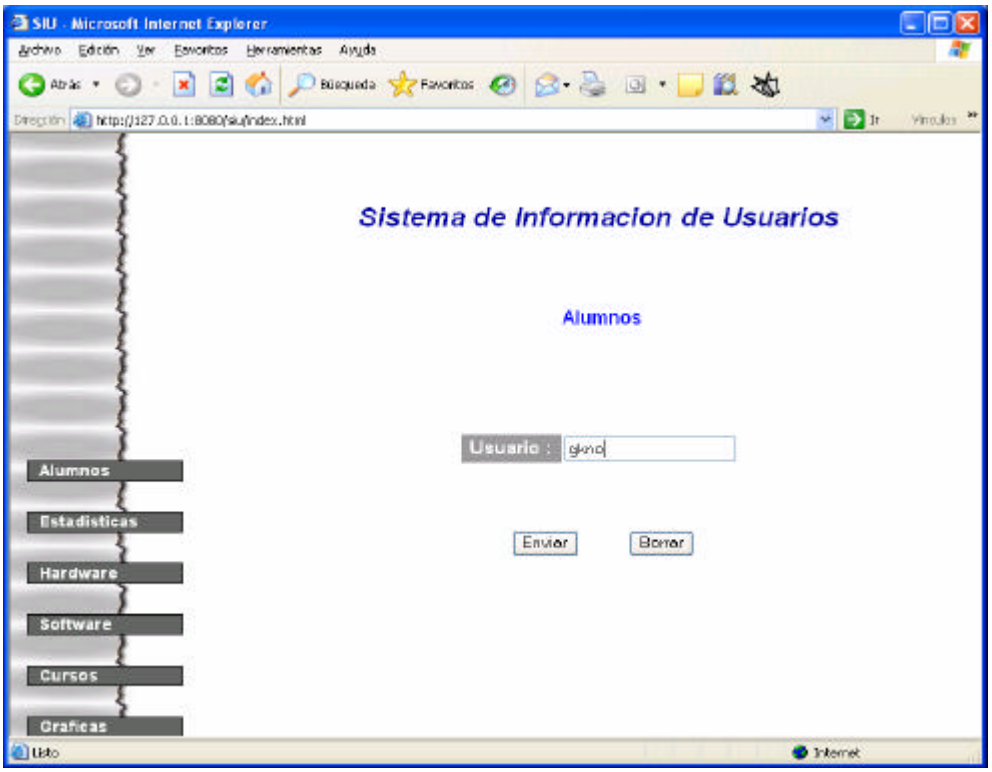


Figura 5.10 Ingreso de alumno para buscar historial.

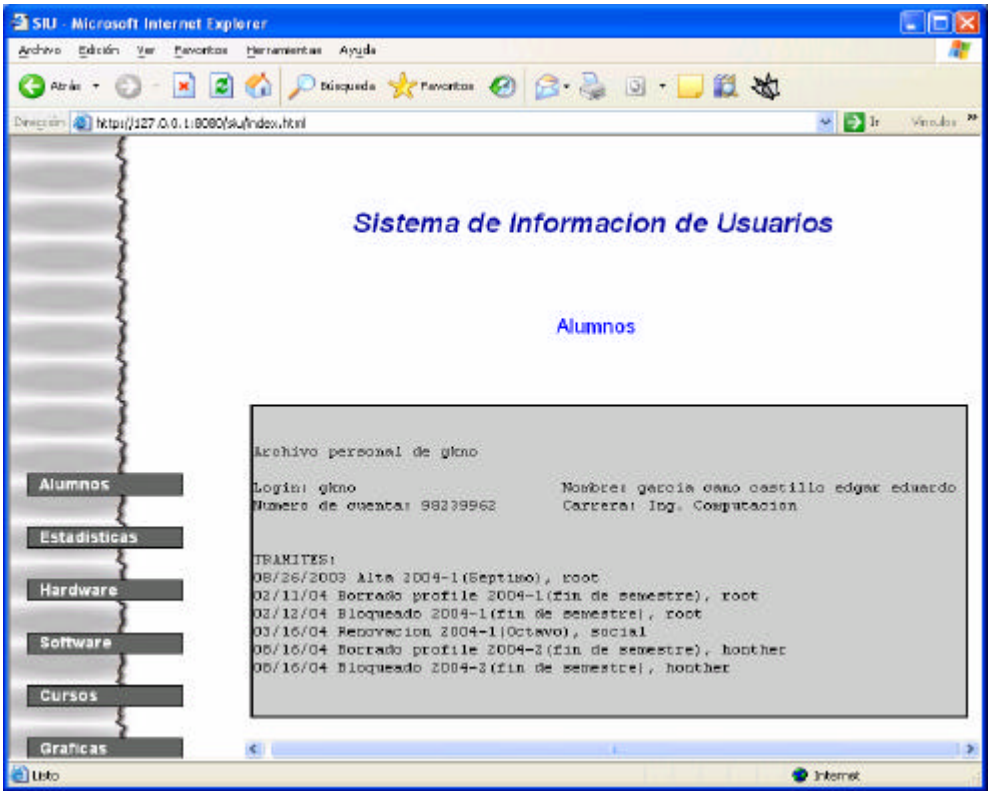


Figura 5.11 Historial mostrado del usuario seleccionado.

Consulta Estadísticas de Alumnos.

Funcionalidad o Característica:

De acuerdo con los parámetros que se proporcionen, se obtienen estadísticas de alumnos que ingresan al Laboratorio de Computación.

Secuencia:

- El usuario solicita la página de validación.
- El usuario ingresa su login y contraseña.
- Clic en botón “Enviar”.
- El usuario entra a la Zona de Vistas.
- El usuario selecciona la opción “Estadísticas”.
- El usuario introduce los respectivos parámetros a buscar.
- Respuesta del sistema.

Prerrequisitos:

- Haber entrado al sistema al ingresar login y contraseña correctos.
- Semestre = Décimo
- Carrera = Computación.

Resultados de la prueba:

Se obtiene una tabla donde se muestran todos los alumnos que se encuentran inscritos en el semestre que se está cursando y que son de la carrera de “Computación” y que se encuentran inscritos en “Décimo” semestre.

Evaluación de resultados:

El procedimiento es exitoso al proporcionar por lo menos un parámetro. Si no se proporciona al menos un parámetro el sistema arrojará un mensaje de “Escoja una opción”.

Pantallas:

Habiendo pasado por la parte de validación, las pantallas obtenidas son:

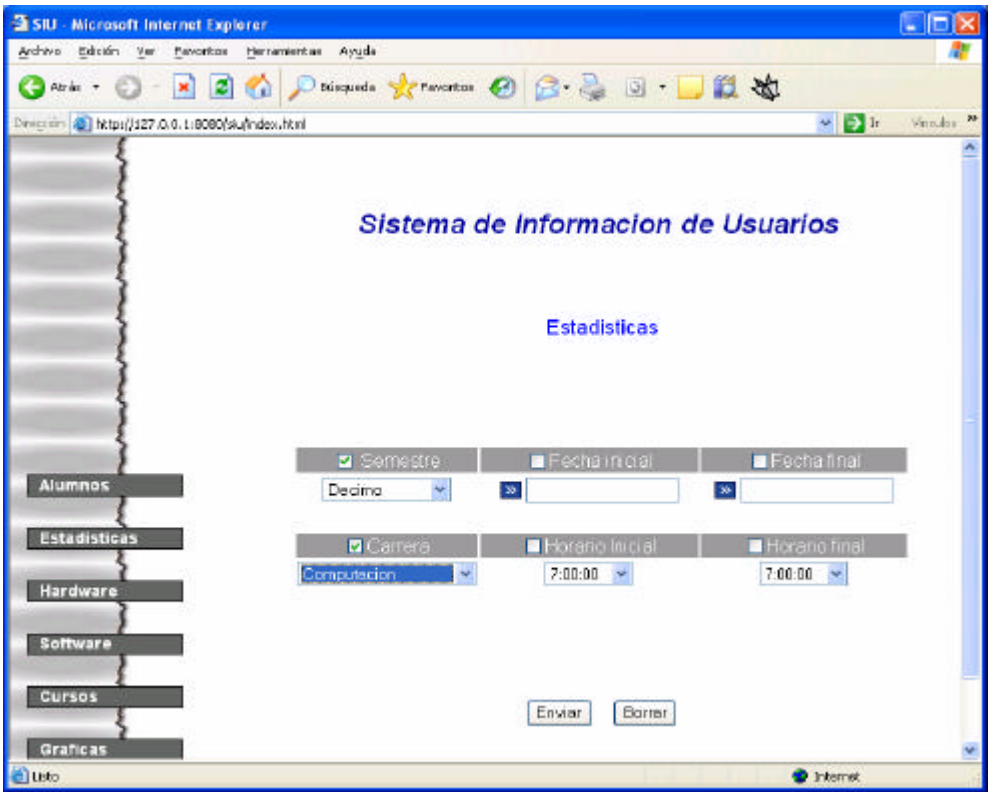


Figura 5.12 Selección de parámetros para realizar consultas.

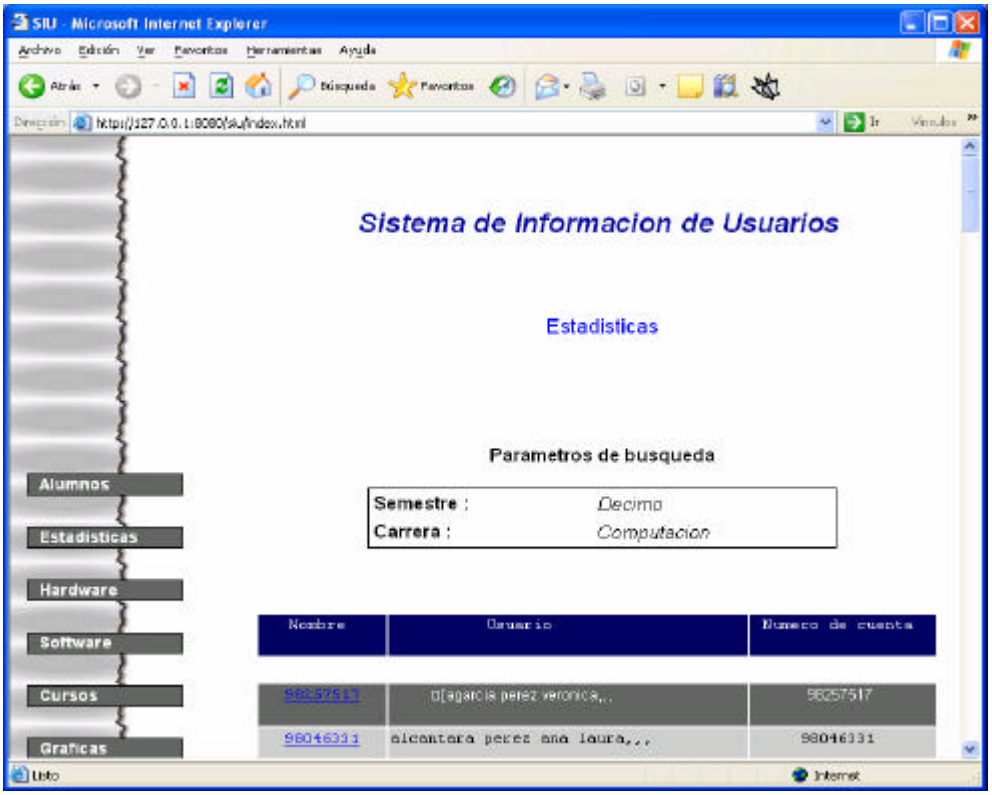


Figura 5.13 Resultado de la consulta seleccionada.

Consulta Hardware .

Funcionalidad o Característica:

Consultar el hardware disponible en el Laboratorio de Computación.

Secuencia:

- El usuario solicita la página de validación.
- El usuario ingresa su login y contraseña.
- Clic en botón “Enviar”.
- El usuario entra a la Zona de Vistas.
- El usuario selecciona la opción “Hardware”.
- Respuesta del sistema.

Prerrequisitos:

- Haber entrado al sistema al ingresar login y contraseña correctos.

Resultados de la prueba:

Se obtienen las características del Hardware que utilizan los usuarios que hacen uso del Laboratorio de Computación.

Evaluación de resultados:

El procedimiento es exitoso. En caso de no existir datos, el sistema muestra un mensaje de “No hay datos disponibles”.

Pantallas:

Habiendo pasado por la parte de validación, la pantalla obtenida es:

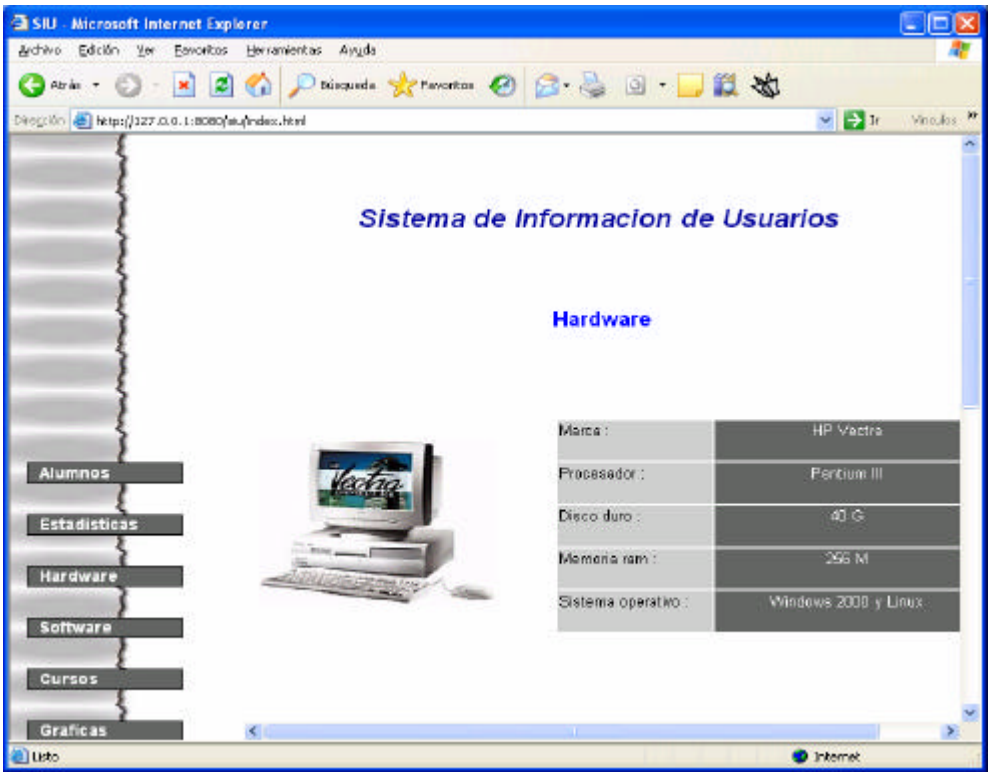


Figura 5.14 Detalle del Hardware

Consulta Software.

Funcionalidad o Característica:

Consultar el Software disponible en el Laboratorio de Computación.

Secuencia:

- El usuario solicita la página de validación.
- El usuario ingresa su login y contraseña.
- Clic en botón “Enviar”.
- El usuario entra a la Zona de Vistas.
- El usuario selecciona la opción “Software”.
- Respuesta del sistema.

Prerrequisitos:

- Haber entrado al sistema al ingresar login y contraseña correctos.

Resultados de la prueba:

Se obtiene el software con el que cuentan los equipos disponibles para los usuarios que hacen uso del Laboratorio de Computación.

Evaluación de resultados:

El procedimiento es exitoso. En caso de no existir datos, el sistema muestra un mensaje de “No hay datos disponibles”.

Pantallas:

Habiendo pasado por la parte de validación, la pantalla obtenida es:

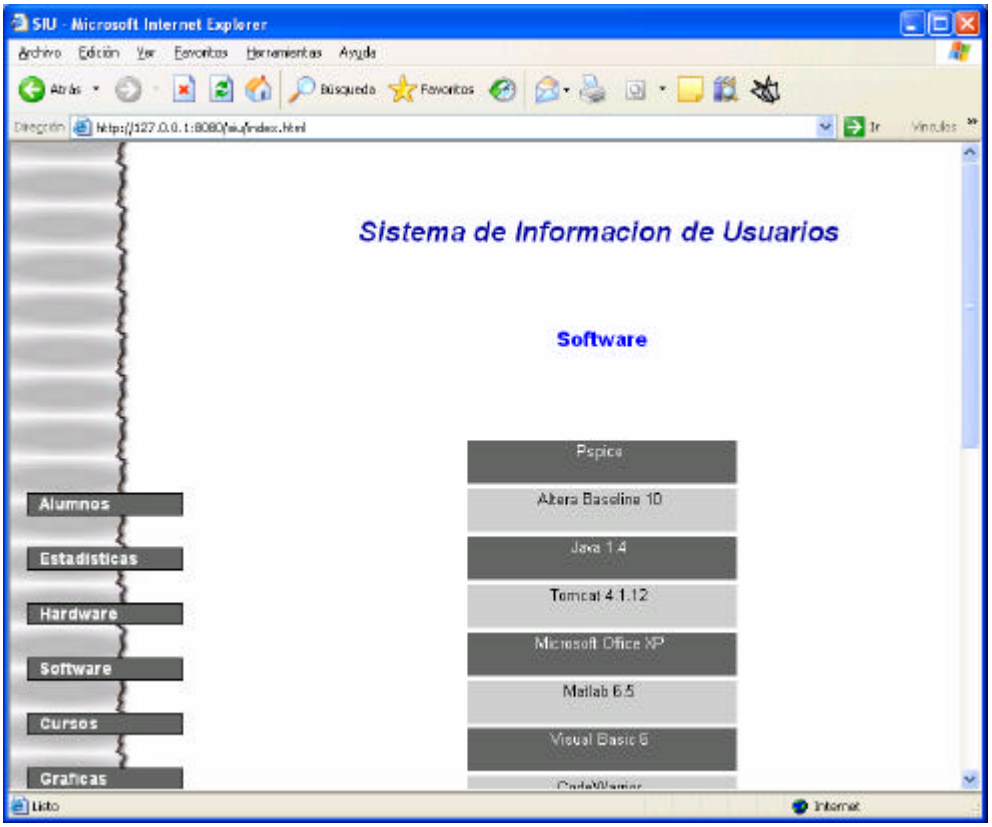


Figura 5.15 Detalle del Software.

Consulta Cursos.

Funcionalidad o Característica:

Consultar los cursos o clases que se han o se imparten.

Secuencia:

- El usuario solicita la página de validación.
- El usuario ingresa su login y contraseña.
- Clic en botón “Enviar”.
- El usuario entra a la Zona de Vistas.
- El usuario selecciona la opción “Cursos”.
- El usuario selecciona “Clases semestrales”.
- El usuario selecciona un semestre.
- Respuesta del sistema.

Prerrequisitos:

- Haber entrado al sistema al ingresar login y contraseña correctos.
- Seleccionar opción “Clases Semestrales”.
- Seleccionar Semestre 2007-1.

Resultados de la prueba:

Se obtiene las clases que se han impartido en el semestre 2007-1.

Evaluación de resultados:

El procedimiento es exitoso. En caso de no existir datos, el sistema muestra un mensaje de “No hay datos disponibles”.

Pantallas:

Después de pasar por la parte de validación, la pantalla obtenida es:

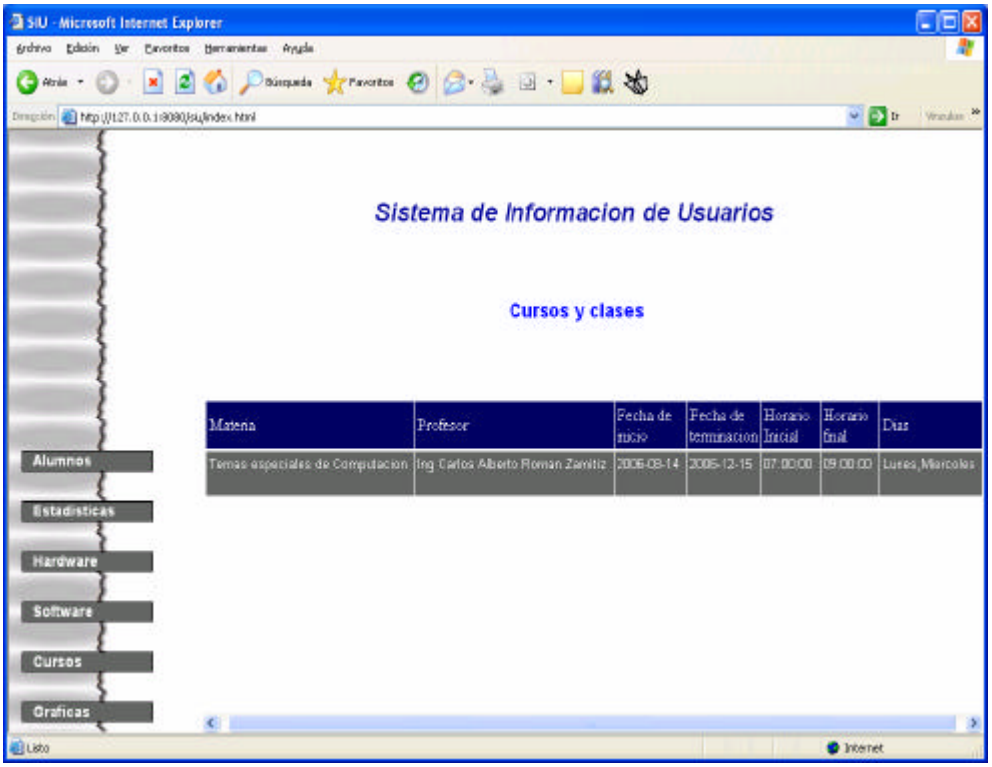


Figura 5.16 Detalle de Clases Semestrales.

Consulta Gráficas.

Funcionalidad o Característica:

Consultar los cursos o clases que se han o se imparten.

Secuencia:

- El usuario solicita la página de validación.
- El usuario ingresa su login y contraseña.
- Clic en botón “Enviar”.
- El usuario entra a la Zona de Vistas.
- El usuario selecciona la opción “Gráficas”.
- El usuario selecciona “Carreras”.
- Respuesta del sistema.

Prerrequisitos:

- Haber entrado al sistema al ingresar login y contraseña correctos.
- Seleccionar opción “Carreras”.

Resultados de la prueba:

Se obtiene la gráfica del porcentaje de alumnos inscritos en el Laboratorio de Computación que se encuentran inscritos en el semestre actual.

Evaluación de resultados:

El procedimiento es exitoso. En caso de no existir datos, el sistema muestra un mensaje de “No hay datos disponibles”.

Pantallas:

Habiendo pasado por la parte de validación, la pantalla obtenida es:

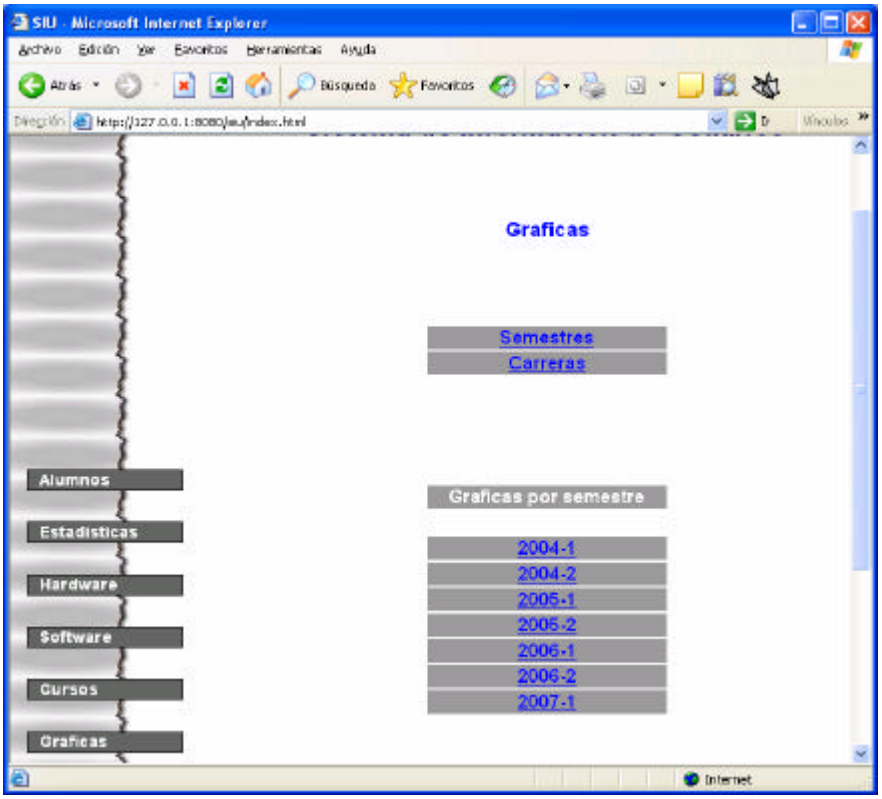


Figura 5.17 Opciones para generar gráficas.

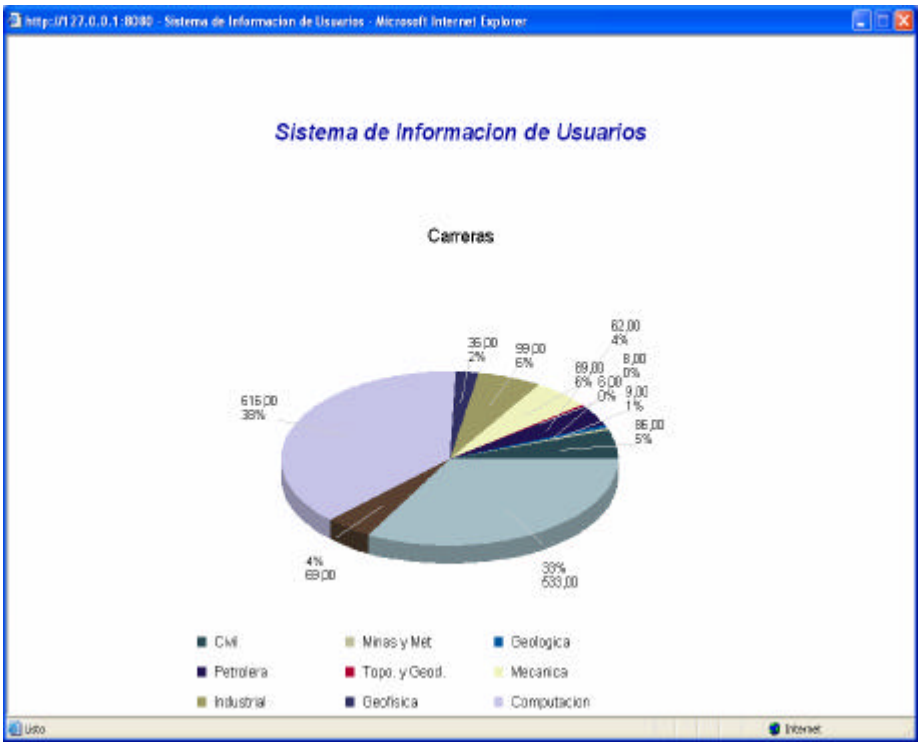


Figura 5.18 Figura obtenida por Carreras.

5.3. Conexión a la base de Datos.

- Requerimientos a cubrir.

  Para realizar la conexión por medio de JDBC necesita los siguientes parámetros:

  - Dirección del servidor.
  - Puerto de la base de datos.
  - Nombre de la base de datos.
  - Nombre de usuario.
  - Contraseña de usuario.

- Condiciones prerrequisito.

  Para poder realizar una conexión como cliente a la base de datos, se debe de tener una cuenta para que esta nos habilite y podamos establecer una conexión.

- Estradas de prueba.

  Las pruebas consisten en establecer una conexión a la base de datos y una vez teniéndola empezar a realizar algunas consultas como INSERT, UPDATE, DELETE o JOIN.

- Las pruebas que se hicieron fueron.

  - Obtener todos los alumnos que ingresaron al Laboratorio de Computación en el semestre 2006-2 y a su vez ordenarlos por mes.
  - Obtener todos los usuarios registrados en el semestre 2007-1.
  - Obtener todos los accesos de usuarios del 15/04/2006 al 15/09/2006.

  Habiendo terminado de hacer las consultas, se debe cerrar la conexión.

- Resultados de prueba.

  Los datos registrados conservan la integridad deseada y como se esperaba.

- Evaluación de resultados.

  La conexión es satisfactoria. Las consultas se realizan adecuadamente, de esta manera se pueden analizar los resultados obtenidos. Las instrucción de SQL que se utiliza para realizar consultas específicas de los meses del año es extract(month from year) y para agruparlas se utiliza “GROUP BY”, así se asocian los registros por mes. En caso de ser necesario se utiliza la instrucción “DISTINCT” para eliminar datos repetidos o “COUNT” para contar el número de registros seleccionados. Para realización de la consulta específica de una

fecha, esta se puso en las sentencias SQL en el siguiente formato “AAAA-MM-DD” utilizando >,< o = para hacer las comparaciones.

Es de vital importancia cerrar todas las conexiones que se abran, para evitar congestionar el servidor y éste genere problemas.

5.4. Pruebas de Usuario

Estas pruebas implican, una revisión continua sobre los objetivos planteados para la realización del sistema, de tal manera que el usuario quede completamente satisfecho con el producto final entregado. Esto se realiza mediante sesiones con el usuario final; éste hace las pruebas necesarias sobre el sistema hasta quedar plenamente satisfecho con los resultados que arroja el mismo.

Entre las observaciones realizadas a lo largo del desarrollo del sistema se obtuvieron lo siguiente:

- Respecto a las validaciones de las vistas.
  - En el caso de los campos que van a contener nombre referente a un usuario o módulo, sólo se deben aceptar letras, número y espacios en blanco.
  - Para las rutas de alguna vistas, se deben aceptar letras, números, “=”, “/” y “&”.
  - Se debe validar que no se dejen campos en blanco antes de enviar datos.
  - En la vista de estadísticas, al menos se debe de escoger una opción para poder realizar una consulta.

- Respecto a las consultas.
  - En la vista de estadísticas, si se selecciona la opción de Carreras, Semestres o ambas, los resultados deben de obtener los alumnos que se encuentren dados de alta en el semestre actual.
  - Para las gráficas Carreras y Semestres, los resultados que se obtengan tienen que ser referentes a los alumnos inscritos en el semestre actual.
  - Los resultados que muestre la gráfica deben de tener etiquetado el valor que representa.

6. Liberación

6.1. Análisis

Esta etapa consta de la lista de requerimientos, descripción de los conceptos básicos y del sistema, diagrama de casos de uso, descripción de los mismos y finalmente el diagrama de clases; todos puntos necesarios para la creación del sistema. Esta información se encuentra en los capítulos primero y segundo. Esta etapa fue liberada por el Ing. Alejandro Velázquez Mena, el mes de Enero de 2006.

6.2. Diseño

Para realizar el diseño del sistema se hizo uso de diagramas de secuencia, diagramas de actividades, diagramas de la base de datos (modelos conceptual y físico) y finalmente secuencias de pantallas. Esta información se encuentra en el capítulo tercero. Esta etapa fue liberada por el Ing. Alejandro Velázquez Mena, el mes de Marzo de 2006.

6.3. Desarrollo

El trabajo de desarrollo engloba la programación en Java, el código SQL para la generación de la base de datos, la creación de las gráficas, validación de javascript, implementación de las hojas de estilo. Toda la información referente a la información antes mencionada se encuentra referenciada en el capítulo cuarto, así como en los anexos A y B. Esta etapa fue liberada por el Ing. Alejandro Velázquez Mena, el mes de Noviembre de 2006.

Los entregables consistieron en:

- Diagramas de Casos de Uso.
- Diagramas de Secuencia.
- Diagramas de Actividades.
- Diagramas de la base de datos (conceptual y físico).
- Scripts de creación de la base de datos.
- Archivo siu.war que contiene toda la aplicación.
- Manual de usuario.

6.4. Pruebas

En esta etapa las pruebas que se realizaron, están referenciadas en el capítulo quinto y constan de pruebas de estrés al Servidor Web, esto mediante un software especializado. También se realizaron casos de prueba al sistema, con lo cual se confirmó que lo que el sistema da como resultados, cumple con las especificaciones dadas en la etapa de análisis. Las pruebas fueron realizadas por el tesista, así como el Ing. Alejandro Velázquez Mena, el cual hizo la liberación de ésta etapa el mes de diciembre de 2006.

7. Conclusiones

En la actualidad es necesario contar con sistemas computacionales que proporcionen información sobre los recursos que se proveen, quiénes los utilizan y con qué frecuencia se hace. Esto con el fin, de que en dado caso de ser necesario, se implementen medidas para satisfacer la demanda del personal que utiliza los servicios. Lo anterior llevó a la creación del Sistema de Información de Usuarios (SIU).

Para implementar un sistema, fue necesario contar con una metodología que explicara paso a paso cómo funciona el sistema, en este caso se utilizó la Ingeniería de Software, en la cuál a través de sus diferentes fases se realizó la especificación del sistema.

Para obtener los requerimientos del sistema, lo que se hizo fue tener pláticas con el usuario final, para que éste especificara los objetivos del mismo, ya que el sistema, como producto final debe de cumplir con todos los objetivos planteados.

Después de obtener todos los requerimientos que tiene el usuario, para el sistema, se hizo un extenso análisis para poder plantear la forma en que se resolverán dichos objetivos, de esta manera se realizó una solución que se expuso al usuario final, de tal forma que ésta satisfaciera todos los objetivos del sistema. Para esto se hizo se hizo una definición de lo que es el sistema, además de generarse un diagrama de casos de uso y de clases, explicando detalladamente el funcionamiento de cada componente.

Al terminar esta fase se hizo un diseño apegado al análisis obtenido, para esto se realizaron diagramas y se hizo una especificación del sistema, para así tener la idea del esqueleto que lo formaría. Después se realizaron diagramas de secuencia y actividades, esto muestra las capas en que está conformado el sistema (Modelo-Vista-Controlador), y la manera en que interactúa una con la otra; así como los diagramas físico y conceptual de la base de datos.

Se realizaron las pantallas que conformarían el sistema, esto sólo como una especificación de cómo se navegaría en el sistema. También se hizo la especificación es estándares de programación y de la base de datos, con el fin de que el sistema cuente con una misma estructura para su mayor comprensión.

Los diagramas de casos de uso, clases, secuencia y actividades antes mencionados, forman parte de UML, una poderosa herramienta en la creación de software que ayuda a que mientras se va creando el sistema, éste se vaya documentando automáticamente. De esta manera se utilizó una herramienta que puede ser entendida por cualquier desarrollador, que si en un futuro alguien necesita saber la funcionalidad del sistema, ésta sea comprendida con facilidad.

Para el desarrollo del sistema se hizo uso del lenguaje de programación en Java, ya que es una de las principales plataformas a nivel mundial, además de que es portable y por lo tanto es posible utilizarlo en cualquier sistema operativo que cuente con la JVM. Además de que existe una gran cantidad de documentación disponible en el sitio de Sun Microsystems y en general en la red existen una amplia cantidad de ejemplos.

La base de datos PostgresSQL ha funcionado adecuadamente, además, gracias a su manejador es más fácil utilizarla de manera visual. Un aspecto importante es que PostgresSQL existe para diversas versiones de sistemas operativos, lo que hace que se pueda implantar en cualquiera de ellos.

En cuanto al servidor Web Tomcat, éste cumplió con todas las pruebas de estrés realizadas, esto es importante porque debe de poder despachar todas las peticiones de manera adecuada, ya que en dado caso de que se sobrecargara, pararía los servicios.

El utilizar el lenguaje de programación Java, la base de datos PostgresSQL y como servidor Web, Tomcat satisface el objetivo de utilizar software libre.

Se hicieron pruebas sobre el funcionamiento del sistema, con lo cual se verificó que dados ciertos datos, se obtuvieran resultados ya esperados. El sistema es capaz de absorber cualquier cantidad de módulos que vayan de acuerdo con los estándares planteados en el análisis y diseño del sistema, lo cual hace que éste sea expandible.

El realizar un archivo .war hace que el sistema pueda ser portado a cualquier servidor de aplicaciones. El generar scripts para la creación de la base de datos, hace que el sistema no sea dependiente de una base de datos y se pueda implantar en cualquier base de datos. Todo lo anterior hace que el sistema sea totalmente portable e independiente del sistema operativo, la base de datos o el servidor de aplicaciones. Un punto importante es que puede ser implantado para otros lugares que ofrezcan servicios similares que el Laboratorio de Computación de la DIE.

Por el API de Visual Engineering fue fácil implementar las gráficas, las cuales son útiles para visualizar los datos del sistema, cómo los alumnos por carrera o por semestre, ver la afluencia por día en el caso del semestre que se cursa, o ver la afluencia de usuarios en algún semestre ya transcurrido.

Con en el módulo de estadísticas podemos obtener datos en tiempo real de los usuarios que se encuentran utilizando el Laboratorio de Computación de la DIE u obtener datos de días anteriores, ya sea por algún rango de fechas o de horario, o con las opciones de semestres o carreras.

En general el sistema cumple con todos los objetivos planteados desde el momento del análisis y en particular, a mí me sirvió para aplicar conceptos aprendidos en algunas de las materias que estudié a lo largo de la carrera, como lo son: Computadoras y Programación, Ingeniería de Software, Bases de Datos, Redes de Computadoras y Temas Especiales de Computación. De esta manera no sólo apliqué lo aprendido en dichas materias, si no que dejo algo de utilidad a la Facultad de Ingeniería y a la UNAM.

Apéndices

Anexo A: Estándares de Programación

Definición de estándares para nombres de archivos y su almacenamiento

El sistema SIU maneja, principalmente, los siguientes tipos de archivos binarios:

- Archivos de PowerDesigner (*.sws, *.oom, *.pdm, *.cdm, *.pdb).
- Archivos imágenes (*.gif, *.jpg).

El sistema SIU maneja, principalmente, los siguientes tipos de archivos texto con código fuente:

- Archivos HTML (*.html) y CSS (*.css).
- Archivos clases Java (*.java).
- Archivos JSP's (*.jsp).
- Archivos de JavaScript (*.js).
- Archivos de scripts de SQL (*.sql).

Los siguientes directorios deben de estar debajo de la raíz (o contexto de la aplicación) que llevará el nombre de "siu".

Todos estos archivos, binarios y de texto, deberán estar guardados debajo de una directorio denominada "files".

Todas las imágenes deberán estar guardadas debajo de una directorio denominada "images".

Todos los archivos HTML deberán estar guardados bajo un directorio denominado "html".

Todos los archivos .css deberán estar guardados bajo el directorio denominado "css".

Todos los archivos .js deberán estar guardados bajo el directorio denominado "js".

Todos los archivos .properties deberán estar guardados bajo el directorio denominado "styles".

Todos los archivo .jsp deberán estar guardados bajo el directorio denominado "jsp", y estarán guardados en paquetes, que siguen el estándar de definición de paquetes.

Todos los archivo .java deberán estar guardados bajo el directorio denominado "WEB-INF/classes/siu", y estarán guardados en paquetes, que siguen el estándar de definición de paquetes.

Definición de estándares para la nomenclatura de JSP's.

Todos los archivos .jsp deben de cumplir con las siguientes características:

- El nombre del JSP debe de estar en idioma inglés.
  Siu_Modulo_JspNombre.jsp
- La primera letra de cada palabra debe de estar escrito en mayúsculas.
- La primera palabra Siu identificara que pertenece al sistema "Sistema de Información de Usuarios (SIU)".
- Después del primer guión bajo va escrito el nombre del módulo al que pertenece dicho JSP y cuenta con las siguientes características:
  - Debe contener máximo cinco caracteres.
  - Debe estar escrito en idioma inglés.
- Después del segundo guión bajo va escrito el nombre del JSP con las siguientes características:
  - Debe tener al principio la palabra "Jsp".
  - La siguiente palabra o palabras del JSP que describen su nombre deben de tener la primera letra en mayúscula.
  - Todos los JSP's deben estar contenidos en un paquete de acuerdo con la definición de estándares de paquetes.
- Todo el texto mostrado debe de ser dinámico y extraído de la Base de Datos a través de un de la clase Siu_Admin_DBContextFile y será enviado a través de la aplicación en un objeto de la clase Siu_ContainerLabel; este objeto contendrá las etiquetas correspondientes a la vista de cada módulo e idioma seleccionado al inicio de sesión.
- Sólo existirá una página de error denominada Siu_Admin_JspError.jsp, la cuál sólo mostrará el mensaje que se obtiene de igual manera desde la Base de Datos, tal y cómo se indicó en el punto anterior. El JSP sólo recibe un parámetro llamado "error" que contiene el identificador de la etiqueta a mostrar.

Definición de estándares para la nomenclatura de Servlets.

Todos los archivos .java que definan un Servlet deben de cumplir con las siguientes características:

- El nombre del Servlet debe de estar en idioma inglés.
  Siu_Modulo_ServletNombre.java
- La primera letra de cada palabra debe de estar escrito en mayúsculas.
- La primera palabra Siu identificara que pertenece al sistema "Sistema de Información de Usuarios (SIU)".
- Después del primer guión bajo va escrito el nombre del módulo al que pertenece dicho Servlet y cuenta con las siguientes características:
  - Debe contener máximo cinco caracteres.
  - Debe estar escrito en idioma inglés.
- Después del segundo guión bajo va escrito el nombre del Servlet con las siguientes características:
  - Debe tener al principio la palabra "Servlet".
  - La siguiente palabra o palabras del Servlet que describen su nombre deben de tener la primera letra en mayúscula.
  - Todos los Servlet's deben estar contenidos en un paquete de acuerdo con la definición de estándares de paquetes.
- La función de cada Servlet es fungir como el controlador de las vistas, que serán enviadas a través de un objeto de la clase RequestDispatcher.

Definición de estándares para la nomenclatura de archivos fuente Java

- Cada archivo con código Java debe contener una sola clase o interfaz.
- Dicha clase o interfaz debe ser public.
- Todas las clases e interfaces deben estar contenidas en un paquete (package) y la sintaxis y nombre de ese paquete está definido en la sección de definición de estándares de paquetes.
- El nombre de la clase o interfaz debe estar en inglés.
- Deben estar contenidos en un paquete de acuerdo con el estándar de definición de paquetes.

Para el caso de interfaces, las variables (si es que tiene) deben ser public static final.

Para el caso de clases, éstas caen en alguna de las siguientes tres categorías:

- Clases de Entidad o Persistencia de Datos.
- Clases Contenedoras.

Hasta este momento sólo se tiene contemplado crear dos clases contenedoras que van a contener objetos de tipo Object, por lo que es genérica y será utilizada por las clases de negocio. Estas clases se llamarán Siu_Container y Siu_ContainerLabel estará en su archivo Siu_Container.java y Siu_ContainerLabel.java .

Las clases de entidad o persistencia deben cumplir la siguiente sintaxis en su nombre de archivo:

Siu_*Nombre*.java

Donde *Nombre* sea un nombre asociado al ambiente del negocio del sistema, por ejemplo User, Courses, etc.

Por ejemplo, para el manejo de los usuarios La clase de entidad Siu_Admin_User.java. La clase de entidad va a fungir como JavaBean y va a tener los atributos de un usuario con visibilidad private y sus métodos públicos (public) getXXX() y setXXX(), donde XXX representa cada uno de los atributos de la clase Siu_Admin_User.

Convenciones de nomenclatura para variables (atributos) y métodos

Las convenciones de nomenclatura Proporcionan información sobre la funcionalidad del identificador; por ejemplo, si es una constante, un paquete o una clase, lo que también redunda en una ayuda adicional a la hora de entender el código.

Elemento	**Reglas de Nomenclatura**	**Ejemplo**

Métodos	Los métodos deben ser en mayúsculas y minúsculas, con la primera letra en minúscula, y la primera letra de cada una de las palabras internas en mayúscula.	run();
		runFast();
		getBackground();

Variables	Todos los nombres de variables de instancia o de clase deben estar constituidos por palabras con la primera letra de la primera palabra en minúscula y la primera letra de las palabras internas en mayúscula. Se deben evitar las variables de una sola letra, excepto en variables temporales de corto uso. Nombres comunes para este tipo de variables son: i, j, k, m y n para enteros; c, d, y e para caracteres.	int i; float myWidth;
Constantes	Los nombres de variables declaradas como constantes de clase (que sean public static final) se deben escribir siempre en mayúsculas, con las palabras internas separadas por un guión bajo ("_").	int MIN_WIDTH = 4; int MAX_WIDTH = 9;

Definición de estándares para paquetes para las clases.

Las clases están debajo del directorio WEB-INF/classes/, a partir de aquí estarán organizados los paquetes dependiendo del módulo en el que se encuentren. Por ejemplo para el caso del módulo de administración el paquete sería:

siu.administration

Las características de cada paquete son las siguientes:

- "siu" es el paquete que contendrá a todos los paquetes.
- El nombre del paquete debe estar en inglés.
- Debajo del nombre del paquete se debe seguir la siguiente estructura
  - beans : Aquí se encuentran las clases de entidad del módulo.
  - business: Aquí se encuentran las clases de negocio del módulo.
  - Servlet: dónde se van a encontrar almacenados todos los Servlets corrspondientes al módulo
- El módulo de administracón cuenta con otro paquete llamado "common" al nivel de los paquetes del punto anterior, que a su vez esta dividido en dos paquetes más y que se describen a continuación:
  - database: contiene las clases referentes a la base de datos y el Servlet Listener:
    - Siu_Admin_DBConector
    - Siu_Admin_DBContextFile
  - Util: contiene las clases de utilierías como:
    - Siu_Container
    - Siu_ContainerLabel

Definición de estándares para paquetes de JSP's

Los JSP's deben de ir debajo del directorio "jsp", dentro de éste directorio los JSP's se almacenaran en un directorio con el nombre del módulo al que pertenezcan, debe contar con las siguientes características:

- Debe estar escrito en idioma inglés
- Debe estar formado por dos subdirectorios:
  - common: Aquí se guardan todos los JSP's correspondientes al módulo. Si se tuviera el siguiente JSP "Siu_Admin_JspConsultModule.jsp", estaría guardado en la siguiente ruta:

    /siu/jsp/administration/common/Siu_Admin_JspConsultModule.jsp
  - error: Donde se almacena el JSP que será genérico para envío de mensajes de error (Siu_Admin_JspError.jsp), que tendría asignada la siguiente ruta
    /siu/jsp/administration/error/ Siu_Admin_JspError.jsp

Definición de estándares de la conexión a la base de datos.

En la conexión a la base de datos debe hacer referencia siempre al componente Siu_Admin_DBConnection. Los parámetros de configuración de la conexión a la base de datos se van a tomar de un archivo llamado DBParameters.txt.

Los parámetros de la base de datos serán cargados por el componente Siu_Admin_DBContextFile y serán enviados a través del sistema por un objeto de la clase Siu_Container.

La clase Siu_Admin_DBConnection contiene todos los métodos que se ocupen de la interacción con la conectividad a la base de datos, como lo son los métodos hacerSelect o hacerInsert.

Anexo B: Estándares de Diseño de la Base de Datos

Tablas de base de datos:

El nombre de la tabla debe tener las siguientes características y sintaxis:

Siu_Modulo_Nombre

- Debe de estar escrita la primera letra de cada palabra en mayúscula y en singular.
- La primera palabra Siu identificará que la tabla pertenece al sistema "Sistema de Información de Usuarios (SIU)".
- Después del primer guión bajo va escrito el nombre del módulo al que pertenece dicha tabla y cuenta con las siguientes características:
  - Debe contener máximo cinco caracteres.
  - Debe estar escrito en idioma inglés.
- Después del segundo guión bajo va escrito el nombre de la tabla con las siguientes características:

  - Debe estar escrito en idioma inglés.
  - Si es una tabla que haga la relación entre dos tablas, debe de llevar el nombre de las dos tablas separadas por un guión bajo. Por ejemplo:
    Siu_Admin_Module_User

  - Si el nombre de la tabla está formado de dos palabras, la letra inicial de cada palabra de escribe en mayúscula y las palabras se escriben juntas. Por ejemplo:
    Siu_Admin_LabelType

El nombre de cada columna debe de tener las siguientes características y sintaxis.

prefijo_columna

- Debe de estar escrito en minúsculas y en singular.
- Para el "prefijo", si la tabla esta compuesta por dos palabras, se pone la primera letra de la primera palabra y los cuatro caracteres restantes se asignan a la segunda palabra. Por ejemplo si la tabla es Siu_Admin_LabelType, sus columnas tendrían de prefijo:
  ltype_columna

- El “prefijo” debe de ser de tres letras a cinco letras y va a servir como un identificador del nombre de la tabla.
- Si la columna tiene dos palabras para referirse a ella; a partir del prefijo y separando por un subguión o guión bajo la primera letra de la segunda palabra se escribe en mayúsculas; su sintaxis sería:

user_firstName

Anexo C: Servidores Web y de Aplicación

Apache

Apache es un código abierto (el código fuente es fácil de conseguir y puede ser compartido) Es un software de servidor de Web HTTP. Es actualmente el servidor de web más popular en la Red. Es ejecutado generalmente en versiones Unix de sistema operativo como Linux o BSD, aunque también puede ser corrido Windows. Es un servidor representado repleto de muchos componentes adicionales poderosos y fáciles de conseguir.

Internet Information Services (IIS)

Es una serie de servicios para los ordenadores que funcionan con Windows. Fue integrado en otros sistemas operativos de Microsoft destinados a ofrecer servicios, como Windows 2000 o Windows Server 2003
Este servicio convierte a una computadora en un servidor de Internet o Intranet es decir que en las computadoras que tienen este servicio instalado se pueden publicar páginas web tanto local como remotamente.

Jakarta Tomcat

Tomcat (también llamado Jakarta Tomcat o Apache Tomcat) funciona como un contenedor de Servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat es un servidor web con soporte de Servlets y JSP's. Incluye el compilador Jasper, que compila JSP's convirtiéndolas en Servlets.

Es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad y dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual.

Oracle Aplication Server

Es un software creado por Oracle, incorpora en un solo producto una solución de servidor web y base de datos empleada ampliamente en el desarrollo e implementación de portales de comercio electrónico; se utiliza para crear y desplegar portales. Generalmente es usado para contener aplicaciones basadas en los estándares de J2EE; ofreciendo una arquitectura de software que facilita el uso de aplicaciones empresariales. Proporciona un entorno seguro y manejable para acceder e interactuar con servicios de software de empresa y recursos informativos.

BEA WebLogic Server

Permite desarrollar y desplegar rápidamente, aplicaciones fiables, seguras, escalables y manejables. Maneja los detalles a nivel del sistema para que podamos concentrarnos en la lógica de negocio y la presentación.

WebLogic Server utiliza tecnologías de la plataforma Java 2, Enterprise Edition (J2EE). J2EE es la plataforma estándar para desarrollar aplicaciones multicapa basadas en el lenguaje de programación Java. WebLogic Server proporciona un conjunto completo de servicios para esos componentes y maneja automáticamente muchos detalles del comportamiento de la aplicación, sin requerir programación.

Anexo D: Bases de datos

PostgresSQL

Es un poderoso sistema de base de datos relacionada de código abierto: tiene más de 15 años de desarrollo activo y una arquitectura probada que ha ganado una reputación fuerte para la fiabilidad, la integridad de datos, y la corrección. Es una alternativa a otros sistemas de bases de datos de código abierto (como MySQL, Firebird y MaxDB), así como sistemas propietarios como Oracle o DB2.

PostgreSQL corre en los principales sistemas operativos, comprendiendo con esto a Linux, UNIX (AIX, BSD, HP-UX, Solaris), y Windows; cuenta además con numerosos procedimientos almacenados (en múltiples lenguajes).

PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92. Tiene soporte para lenguajes procedurales internos como el PL/pgSQL que es su lenguaje nativo.

Usa una arquitectura cliente/servidor donde hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

MySQL

MySQL es una de las bases de datos más populares desarrolladas bajo la filosofía de código abierto. Fue escrito en C y C++ y destaca por su gran adaptación a diferentes entornos de desarrollo, permitiendo su interacción con los lenguajes de programación más utilizados como PHP, Perl y Java; además de, su integración en distintos sistemas operativos.

Es muy criticado porque carece de muchos elementos vitales en bases de datos relacionales y no es posible lograr una integridad referencial verdadera. Es más utilizado en plataformas Linux aunque puede usarse en otras plataformas. Su uso en un servidor web es gratuito salvo en los casos que se necesite el uso de aplicaciones especiales.

Oracle

Oracle es básicamente un herramienta cliente/servidor para la gestión de base de datos, es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hace que solo se vea en empresas muy grandes y multinacionales, por norma general.

Soporta acceso por SQL y por lenguajes de programación. También posee un lenguaje de procedimientos llamado PL/SQL.

La base de datos tiene estructuras lógicas y estructuras físicas. Y gracias a que estas estructuras son separadas, el almacenamiento físico de datos puede ser manejado sin afectar el acceso a las estructuras lógicas del almacenamiento.

Microsoft SQL Server

Es un sistema de gestión de bases de datos relacionales basada en el lenguaje SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea.

Permite trabajar en modo cliente- servidor, donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información.

Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para la mayoría de las plataformas de desarrollo, incluyendo .NET.

Microsoft SQL Server no es multiplataforma, ya que sólo está disponible en Sistemas Operativos de Microsoft.

Anexo E: Lenguajes de programación

Java

Java es un lenguaje de programación de alto nivel orientado a objetos; fue desarrollado por Sun Microsystems. Es semejante a C + +, pero ha sido simplificado para evitar que el programador cometa errores al introducir el código fuente.

Se pueden realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general.

Una de las principales características por las que Java se ha hecho muy famoso es que es un lenguaje independiente de la plataforma. Eso quiere decir que si hacemos un programa en Java podrá funcionar en cualquier ordenador del mercado. Es una ventaja significativa para los desarrolladores de software, pues antes tenían que hacer un programa para cada sistema operativo, por ejemplo Windows, Linux, Apple, etc. Esto lo consigue porque se ha creado una Máquina de Java para cada sistema que hace de puente entre el sistema operativo y el programa de Java y posibilita que este último se entienda perfectamente.

La independencia de plataforma es una de las razones por las que Java es interesante para Internet, ya que muchas personas deben tener acceso con ordenadores distintos. Pero no se queda ahí, Java está desarrollándose incluso para distintos tipos de dispositivos además del ordenador como móviles, agendas y en general para cualquier cosa que se le ocurra a la industria.

PHP

PHP es un lenguaje de escritura embebido para las web HTML. Esto significa que el código de PHP puede ser insertado dentro de una página de HTML. Cuando se accede a una página con código PHP, éste es leído o analizado por el servidor de la página. La producción de las funciones de PHP en la página es vuelta típicamente como código de protocolo de transferencia de hipertexto que puede ser leído por el examinador. El código de PHP es transformado en el protocolo de transferencia de hipertexto antes de que la página sea cargada para de ésta manera evitar que el usuario lea el código.

Las páginas hechas con éste código aseguran conseguir acceso a bases de datos y otra información segura. Mucha sintaxis de PHP es pedido prestada de otros idiomas tales como C, Java y Perl.

La meta del lenguaje es permitir crear desarrolladores de web escribir dinámicamente y generar páginas de una manera rápida y fácil.

.NET

.NET es una plataforma de software que conecta información, sistemas, personas y dispositivos, se podría considerar como una respuesta de Microsoft al creciente mercado de los negocios en entornos Web, como competencia a la plataforma Java de Sun Microsystems.

La plataforma .NET conecta una grande variedad de tecnologías de uso personal y de negocios, de teléfonos celulares a servidores corporativos, permitiendo el acceso a información importante, donde y cuando se necesiten.

Desarrollado con base en los estándares de Servicios Web XML, .NET permite que los sistemas y aplicaciones, ya sea nuevos o existentes, conecten sus datos y transacciones independientemente del sistema operativo, tipo de computadora o dispositivo móvil que se utilice, o del lenguaje de programación empleados para crearlo.

Anexo F: Instalación de Java

Primero bajaremos el software necesario, que no es más que el J2SE 5.0 JDK (También conocido como SDK: Software Development Kit). Desde la página de Java SUN podemos descargarlo: http://java.sun.com/j2se/1.5.0/download.jsp

Hay que asegurarse de que se descargue el JDK y no el JRE. Es fácil darse cuenta porque el JDK ocupa aproximadamente 42 MB, mientras que el JRE son 15 MB. La diferencia entre el JDK y JRE es que el segundo sirve para ejecutar aplicaciones en Java, más no para crearlas. En cambio, el JDK nos trae todas las herramientas para crear aplicaciones. En si, el JRE está incluido dentro del JDK.

Se descargó el JDK en la modalidad "Linux self-extracting file". Existen 2 modalidades de instalación:

- Self-extracting Binary File. Este archivo nos permite instalar el JDK en el directorio que queramos.
- RPM Packages. El archivo rpm requiere que el usuario root lo instale y por defecto lo instalará en un directorio específico y reemplazará la versión del Java que ya estuviera instalado en el Linux.

Instalando el Self-extracting file

1. Establecer permisos de ejecución al archivo:
 # chmod +x jdk-1_5_0-linux-i586.bin

2. Luego nos situamos en el directorio donde queremos instalar el JDK. Por ejemplo:
 # /usr/local/jdk-1.5.0

3. Ejecutamos el archivo de instalación. Supongamos que este archivo está en el directorio actual, entonces ejecutamos:
 # ./jdk-1_5_0-linux-i586.bin

Se mostrará la licencia y tendremos que aceptar algunos términos de uso. Y eso es todo, se tiene Java 5 instalado.

Configurando las variables de entorno

Tenemos que editar el archivo */etc/profile* y agregar las siguientes variables.

JAVA_HOME=/usr/local/jdk-1.5.0
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin
CLASSPATH=$HOME:/jre/lib

export PATH JAVA_HOME CLASSPATH

Para comprobar la instalación escribimos en la línea de comandos: java –version, esto ha de mostrarnos la versión del JDK que hemos instalado.

Anexo G: Instalación de PostgresSQL

Lo primero que se hace es descargar del sitio http://www.postgresql.org/download/ la versión de PostgresSQL. Una vez descargado el archivo postgresql-8.1.4-1.tar.gz lo colocamos en /usr/local/, lo desempacamos y descomprimimos:

 # cd /usr/local
 # tar -zxvf postgresql-8.1.4-1.tar.gz

Creamos el Makefile, compilamos y copiamos los ejecutables a los lugares apropiados:

 # cd postgresql-8.0.4
 # ./configure
 # make
 # make install

Si no ocurrió algún error, en este momento ya tenemos instalado PostgresSQL. Para correr el servidor de bases de datos, es requisito indispensable que no corra como root, así que primero crearemos un usuario de sistema llamado postgres:

 # adduser postgres

Configuramos en el archivo /etc/passwd la línea del usuario que acabamos de crear:

postgres:x:1001:1001::/usr/local/pgsql/data:/bin/bash

Le asignamos un password al usuario

 # passwd postgres

El siguiente paso es crear un clúster de bases de datos de PostgreSQL (PostgreSQL database cluster), que contendrá las bases de datos que vayamos creando. Al momento de inicializar un clúster de bases de datos de PostgreSQL se tiene que indicar el directorio donde se desea que se cree. El propietario de ese directorio tiene que ser un usuario que no sea root. Como nosotros indicamos en el /etc/passwd que el HOME del usuario postgres es /usr/local/pgsql/data, siguiendo con esa lógica, vamos a indicar que ése sea el directorio donde se localice el clúster de bases de datos. Entonces creamos el directorio y le cambiamos de propietario:

 # chown postgres /usr/local/pgsql/data

Ahora nos cambiamos al usuario postgres e inicializamos el clúster de bases de datos con el comando initdb:

 # su - postgres
 # /usr/local/pgsql/bin/initdb -D /var/pgsql/data

Después de recibir el mensaje:

Success. You can now start the database server using:

/usr/local/pgsql/bin/postmaster -D /var/pgsql/data
or
/usr/local/pgsql/bin/pg_ctl -D /var/pgsql/data -l logfile Stara
Hay que agregar las siguientes variables en el archivo /etc/profile:

PGDATA=/usr/local/pgsql/data
PATH=/usr/local/pgsql/bin:$PATH

export PATH PGDATA

Como usuario postgres inicializamos el servidor de bases de datos con el comando pg_ctl:

 # pg_ctl start

Con esto ya se tiene PostgreSQL funcionando.

Existe un archivo de configuración llamado pg_hba.conf en donde se especifica quién, cómo y a qué puede accesar de nuestro servidor de bases de datos, y por default viene especificado que todas las conexiones locales se acepten sin necesidad de password.

Para propósito del SIU, éste archivo se configuró de la siguiente manera:

```
# TYPE  DATABASE    USER        CIDR-ADDRESS          METHOD

# IPv4 local connections:
host    all         all         132.248.59.12/32          trust
host    all         all         132.248.59.20/32          trust
```

Después de haber hecho estos cambios, tenemos que reiniciar el servidor de bases de datos.

 # pg_ctl stop
 # pg_ctl start

Otro archivo importante que se tiene que revisar es el archivo de configuración postgresql.conf. En este archivo se puede configurar el puerto en el que escuchará el servidor, el número máximo de conexiones que permitirá, el uso de los recursos de sistema y las bitácoras, entre otras cosas más. Las opciones que están comentadas son los valores por default. Para cambiar alguna, se tiene que quitar el signo "#" de comentario de la línea y especificar el valor nuevo. Si se hace algún cambio, se tiene que reiniciar el servidor de bases de datos.

Para utilizarlo con el sistema, los únicos parámetros del archivo postgresql.conf que se cambiaron para configurar lo respectivo a conectividad y autentificación fueron:

```
listen_addresses = '*'
port = 5432
max_connections = 100
```

Para la creación de la base de datos del SIU, se ejecutó el comando:

 # createdb siu --owner=postgres

Donde “siu” es el nombre de la base de datos y cuyo propietario es postgres.

Para conectarse a un servidor de base datos PostgreSQL se utiliza el comando psql. Se le pueden pasar muchos parámetros a este comando de la forma psql [OPTIONS]... [DBNAME [USERNAME]].

Para entrar desde la línea de comandos a la base de datos “siu” se ejecuta:

 # psql –U postgres –d siu

Anexo H: Instalación de Tomcat

Se descarga la aplicación desde el sitio web http://www.apache.org/. Del sitio se descarga la aplicación jakarta-tomcat-4.1.12.tar.gz. En este caso no tenemos más que descomprimir el archivo en el directorio que queramos, ya que se trata de una aplicación Java.

Para descomprimir la aplicación se hace lo siguiente:
 # tar -xvzf jakarta-tomcat-4.1.12.tar.gz

Lo anterior crea un directorio con el nombre
jakarta-tomcat-4.1.12

Se debe de editar el archivo /etc/profile para añadir la siguiente variable de entorno:

CATALINA_HOME=/usr/local/ jakarta-tomcat-4.1.12
CLASSPATH=$CLASSPATH:$ CATALINA_HOME/common/lib

export CATALINA_HOME

Y actualizamos:
 # source /etc/profile

Por último ejecutamos el script de iniciación de tomcat:
 # /usr/local/ jakarta-tomcat-4.1.12/bin/startup.sh

Se debería de ver la página de bienvenida de Tomcat introduciendo la URL http://localhost:8080 en un navegador.

Para parar Tomcat se ejecuta:
 # /usr/local/ jakarta-tomcat-4.1.12/bin/shutdown.sh

Después de que se descomprime Tomcat, se obtiene la siguiente jerarquía:

La jerarquía de directorios de instalación de Tomcat incluye:

Nombre de Directorio	Descripción
bin	Contiene los scripts de arrancar/parar.
conf	Contiene varios archivos de configuración incluyendo server.xml (el archivo de configuración principal de Tomcat) y web.xml que configura los valores por defecto para las distintas aplicaciones desplegadas en Tomcat.
doc	Contiene varia documentación sobre Tomcat.
lib	Contiene varios archivos jar que son utilizados por Tomcat. Sobre UNIX, cualquier fichero de este directorio se añade al classpath de Tomcat.
logs	Aquí es donde Tomcat sitúa los archivos bitácora.

src	Los archivos fuentes del API Servlet. Estos son sólo los interfaces vacíos y las clases abstractas que debería implementar cualquier contenedor de servlets.
webapps	Contiene aplicaciones Web de ejemplo.
work	Generado automáticamente por Tomcat, este es el sitio donde Tomcat sitúa los archivos intermedios (como las páginas JSP compiladas) durante su trabajo. Si borramos este directorio mientras se está ejecutando Tomcat no podremos ejecutar páginas JSP.

:

La configuración de Tomcat se basa en dos ficheros:

server.xml es el archivo de configuración principal de Tomcat. Sirve para dos objetivos:

1. Proporcionar configuración inicial para los componentes de Tomcat.
2. Especifica la estructura de Tomcat, lo que significa, permitir que Tomcat arranque y se construya a sí mismo los componentes especificados en el mismo archivo.

Para configurar el servidor en el puerto 8080.

```
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
port="8080" minProcessors="5"
maxProcessors="75" enableLookups="true"
redirectPort="8443" acceptCount="10"
debug="0" connectionTimeout="20000"
useURIValidationHack="false" />
```

web.xml es el archivo para configurar los contextos de Tomcat y cuenta con las siguientes secciones:

Para especificar una variable de Contexto.

```
<context-param>
<param-name>DBParameters</param-name>
<param-value>files/DBParameters.txt</param-value>
</context-param>
```

Para declarar Servlets listeners.

```
<listener>
<listener-class>
    siu.administration.common.database.Siu_Admin_DBContextFile
</listener-class>
</listener>
```

Para declaración de Servlets.

```
<servlet>
<servlet-name>Siu_Admin_ServletIndex</servlet-name>
<display-name>siu.administration.servlet.Siu_Admin_ServletIndex</display-name>
<servlet-class>siu.administration.servlet.Siu_Admin_ServletIndex</servlet-class>
</servlet>
```

Para declaración de JSPs.

```
<servlet>
<servlet-name>Siu_Admin_JspAccess</servlet-name>
<display-name>Siu_Admin_JspAccess</display-name>
<jsp-file>/jsp/administration/common/Siu_Admin_JspAccess.jsp</jsp-file>
</servlet>
```

Para mapeo de Servlets.

```
<servlet-mapping>
<servlet-name>Siu_Admin_ServletIndex</servlet-name>
<url-pattern>/siu.administration.servlet.Siu_Admin_ServletIndex</url-pattern>
</servlet-mapping>
```

Para mapeo de JSPs.

```
<servlet-mapping>
<servlet-name>Siu_Admin_JspAccess</servlet-name>
<url-pattern>/jsp/administration/common/Siu_Admin_JspAccess</url-pattern>
</servlet-mapping>
```

Para configurar archivos como páginas de bienvenida.

```
<welcome-file-list>
<welcome-file>index.html</welcome-file>
</welcome-file-list>
```

Para configurar el tiempo de sesión de 15 minutos.

```
<session-config>
<session-timeout>15</session-timeout>
</session-config>
```

Para terminar la configuración del servidor, se deben agregar los archivos pg73jdbc3.jar y kcServlet.jar al directorio /usr/local/ jakarta-tomcat-4.1.12/common/lib de Tomcat y posteriormente al CLASSPATH en el archivo /etc/profile.

Glosario

API: Interfaz para la programación de aplicaciones. Conjunto de herramientas, rutinas y protocolos utilizados para la construcción de aplicaciones.

Applet: Programa escrito en Java, que se ejecuta en un navegador que sea compatible con dicha tecnología.

Clase: Plantilla que define como son los objetos, indicando sus atributos y sus acciones

Cliente: Es el que inicia un requerimiento de servicio. El requerimiento inicial puede convertirse en múltiples requerimientos de trabajo a través de redes LAN o WAN. La ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente.

Clúster: Es un conjunto contiguo de sectores que componen la unidad más pequeña de almacenamiento de un disco. Los archivos se almacenan en uno o varios clústeres, dependiendo de su tamaño.

Back End: Se denomina de esta manera a los sistemas encargados de almacenar los datos que se utilizan en aplicaciones.

Bucle: Es una sentencia que se realiza repetidas veces.

Front End: Colección de servidores que proveen el núcleo de los servicios Web.

FTP: Es uno de los diversos protocolos de la red Internet, concretamente significa File Transfer Protocol (Protocolo de Transferencia de Archivos) y es el ideal para transferir grandes bloques de datos por la red.

GUI: Interfaz gráfica de usuario. Es un método para facilitar la interacción del usuario con la computadora a través de la utilización de un conjunto de imágenes y objetos pictóricos (iconos, ventanas..) además de texto.

HTML: El protocolo de transferencia de hipertexto (HTTP, HyperText Transfer Protocol) es el protocolo usado en cada transacción de la Web. El hipertexto es el contenido de las páginas Web, y el protocolo de transferencia es el sistema mediante el cual se envían las peticiones de acceder a una página Web, y la respuesta de esa Web, remitiendo la información que se verá en pantalla. También sirve el protocolo para enviar información adicional en ambos sentidos, como formularios con mensajes y otros similares.

Host: Una computadora que realiza algunas tareas en beneficio de otras aplicaciones llamadas clientes.

Interación: Cada una de las repeticiones de las acciones contenidas en un bucle de programa.

Internet: Red mundial de computadoras que transporta datos y hace posible el intercambio de información.

Intranet: Es una red privada dentro de una compañía u organización que utiliza el mismo tipo de software usado en Internet, pero que es sólo para uso interno.

Ipv4: Es la versión 4 del Protocolo IP (Internet Protocol). Esta fue la primer versión del protocolo que se implemento extensamente, y forma la base de Internet.

JAR: Es un formato desarrollado por "Sun" que permite agrupar las clases diseñadas en el lenguaje Java, este formato es ampliamente utilizado en ambientes Java de todo tipo, esto se debe a que otorga un nivel de compresión y reduce la carga administrativa al distribuir clases en el lenguaje.

Java Bean: Es un componente utilizado en Java que permite agrupar funcionalidades para formar parte de una aplicación.

JDBC: Java Database Connectivity. API de Java que brinda a los programadores la posibilidad de interactuar con una base de datos y ejecutar instrucciones SQL.

Kernel: Es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora.

LAN: Es un acrónimo inglés de Local Area Network (Red de Área Local), y que se refiere a las redes locales de computadoras. Sirve para dar servicios a un área geográfica máxima de unos pocos kilómetros cuadrados.

Mainframe: Así se les llama a las grandes computadoras, capaces de atender a miles de usuarios y miles de programas "al mismo tiempo" asignándole un periodo muy pequeño a la atención de cada programa. Su capacidad de trabajo es muy alta, por lo que normalmente se encuentran en empresas de gran tamaño. Sus programas están compuestos por cientos de miles o millones de líneas de código.

MAN: Acrónimo de Metropolitan Area Network que en español significa Red de Área Metropolitana. Es una red de distribución de datos para un área geográfica en el entorno de una ciudad. La distancia entre las estaciones más alejadas es de 10 Km.

Middleware: Software de comunicaciones que se encuentra en el punto intermedio entre el cliente y el servidor de aplicaciones.

Objeto: Es una instancia de una clase, que encapsula estado y tiene un comportamiento, además de una identidad. Pueden representar cosas reales o conceptuales.

ODBC: Conectividad abierta de bases de datos (Open Database Connectivity). Una interfaz de programación de aplicaciones (API), que permite a las aplicaciones tener acceso a bases de datos multiplataforma desde diversas especificaciones estándar de orígenes de datos.

Protocolo IP: Es un protocolo NO orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados.

Servidor: Es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LANs o WANs, para proveer de múltiples servicios a los clientes y ciudadanos tales como impresión, acceso a bases de datos, fax, procesamiento de imágenes, etc.

Servlet: Es un programa escrito en Java y que se encuentra del lado del servidor.

Servlet Engine: Ofrece un Ambiente donde habitan los JSP y Servlets, es ahí donde se contemplan una gran cantidad de funcionalidades como: threading, manutención de sesiones, conectividad con el Servidor de Páginas, es por esto al Servlet Engine también se le denomina Web Container.

Shell: Parte fundamental de un sistema operativo encargada de ejecutar las órdenes básicas para el manejo del sistema. Es el intérprete de comandos que se establece entre el usuario y el kernel.

SSH:Es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquinas remotas a través de una red.

Software libre: Es el software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente.

SQL: El Lenguaje de Consulta Estructurado (Structured Query Language) es un lenguaje declarativo de acceso a base de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

TCP/IP: Familia de protocolos sobre los cuales funciona Internet, que se ha convertido en el estándar actual de comunicación entre computadoras. Conocida por estas siglas debido a que los dos protocolos más importantes son: el protocolo IP, que se ocupa de transferir los paquetes de datos hasta su destino adecuado y el protocolo TCP, que se

ocupa de garantizar que la transferencia se lleve a cabo de forma correcta y confiable.

URL: Uniform Resource Locator, es una secuencia de caracteres con un formato estándar, se usa para nombrar recursos como documentos o imágenes por Internet.

WAN: Wide Area Network o Red de Área Amplia. Es una red de computadoras que puede estar localizada en un área geográfica muy extensa y puede contener varios miles de computadoras interconectadas por medio de canales de comunicación de alta velocidad.

WAR: Web Archive es una especificación desarrollada por Sun que permite agrupar un conjunto de clases y documentos que conforman una aplicación Web en Java.

XML: De las siglas en inglés de eXtensible Markup Language (lenguaje de marcas extensible), es un metalenguaje extensible de etiquetas desarrollado por el W3C. Se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas.

Bibliografía

Fundamentals of the Java Programming language
Sun Microsystems, U.S.A., 2005

Java Programming Language
Sun Microsystems, U.S.A., 2005

Migrating to OO Programming with Java Technology
Sun Microsystems, U.S.A., 2000

Stephen Stelting, Olav Maasen
Patrones de diseño aplicados a Java
Editorial Person Education, S.A., Madrid, 2003

Referencias

http://agamenon.uniandes.edu.co/~pfiguero/soo/Magister_Patrones/intropatrones.html
http://darkwing.uoregon.edu/~ielp/webworkshopglossary.htm
http://datacenter.5points.net/datacenter/support/glossary/
http://es.wikipedia.org/wiki/Paradigma_de_programaci%C3%B3n
http://www.1x4x9.info/files/patrones/html/online-chunked/
http://www.alegsa.com.ar/Diccionario/diccionario.php?letra=o
http://www.alegsaonline.com/art/13.php
http://www.angelfire.com/scifi/jzavalar/apuntes/IngSoftware.html#IngSoft
http://www.desarrolloweb.com/articulos/25.php
http://www.desarrolloweb.com/articulos/26. php
http://www.desarrolloweb.com/articulos/494.php
http://www.desarrolloweb.com/articulos/497.php
http://www.desarrolloweb.com/articulos/1622.php
http://www.elrincondelprogramador.com/default.asp?pag=articulos/leer.asp&id=29
http://www.enterate.unam.mx/Articulos/2006/febrero/arquitec.htm
http://www.htmlpoint.com/sql/sql_04.htm
http://www.maestrosdelweb.com/editorial/introcss/
http://www.microsoft.com/latam/net/introduccion/quees.asp
http://www.monografias.com/trabajos12/basdat/basdat.shtml
http://www.monografias.com/trabajos24/arquitectura-cliente-servidor/arquitectura-cliente-servidor.shtml#algunos
http://www.netchico.com/support/glossary/p.html
http://www.oracle10g.de/oracle_10g_documentation/server.101/b10743/intro.htm#g78401
http://www.postgresql.org/about/
http://portal.uam.es/portalHelp/es_ES/__inl.inline.true/__inl.topic.pbmwwdb_htm/__inl.html
http://www.programacion.com/java/tutorial/beaintro/1/
http://www.programacion.com/java/tutorial/ipintro/2/
http://www.public.asu.edu/~iddwb/writeups/glossary.html
http://www.sistemas.unam.mx/software.html
http://www.sobl.org/traducciones/practical-postgres/node19.html
http://www.sunsite.unam.mx/servidores/docs/bd/ANSI%20SQL.pdf
http://tramullas.com/documatica/2-3.html
http://www.w3c.es/Divulgacion/Guiasbreves/HojasEstilo
http://www.webhostingdiary.com/web-hosting-glossary.htm