Universidad Autónoma Metropolitana Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Reporte final

Implementación de un sistema web para dar seguimiento a las visitas realizadas a entidades financieras por parte de la Comisión Nacional Bancaria y de Valores.

Proyecto de Estancia profesional

Trimestre 14 Primavera

Elaborado por
Nancy Castillo Franco
Matrícula: 208202454

Asesores

Dra. Maricela Claudia Bravo Contreras
Profesor Asociado D
Departamento de Sistemas

Ing. Myriam Valeria Anaya
Directora de Área

14 de julio de 2014

DECLARATORIA

Yo, Dra. Maricela Claudia Bravo Contreras, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Dra. Maricela Claudia Bravo Contreras
Profesor Asociado D
Departamento de Sistemas

Yo, Ing. Myriam Valeria Anaya, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Ing. Myriam Valeria Anaya
Directora de Área

Yo, Nancy Castillo Franco, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Nancy Castillo Franco

Contenido

Resumen 1

Introducción 1

Antecedentes 2

Justificación 2

Objetivos 2

- Objetivo general 2
- Objetivos Particulares 2

Marco Teórico 3

- Arquitectura de 3 niveles 3
- Aplicaciones Web 4
- Servicios Web 4
- Axis2 4
- SQL Server 2005 Express 4

Desarrollo del proyecto 5

- Análisis de requerimientos 5
- Diseño e implementación de las interfaces gráficas 9
- Migración de la base de datos a SQL Server 9
- Diseño de módulos del Back-End 9
- Integración del Back-End y Front-End 11
- Pruebas al sistema 12

Resultados 13

- Análisis y discusión de resultados 17
- Conclusiones 18
- Referencias bibliográficas 18

Índice de figuras

Figura 1. Arquitectura de 3 capas, 3

Figura 2. Gestión de Visitas, 6

Figura 3. Registro de información sobre una entidad financiera, 7

Figura 4. Gestión de Solicitudes, 7

Figura 5. Gestión de Aplicaciones, 8

Figura 6. Gestión de Catálogos, 8

Figura 7. Arquitectura general del sistema web, 10

Figura 8. Login, 13

Figura 9. Mensaje de error, 14

Figura 10. Contraseña incorrecta, 14

Figura 11. Pantalla de bienvenida, 14

Figura 12. Inventario de aplicaciones, 15

Figura 13. Registro de Observaciones, 15

Figura 14. Alcance y riesgo de la visita, 16

Figura 15. Registro exitoso, 16

Figura 16. Registro de Observaciones por visita actualizado, 16

Figura 17. Llamada al servicio Web, 17

Resumen

En una Dirección de la Comisión Nacional Bancaria y de Valores se maneja un sistema de información construido sobre una plataforma de Microsoft Access. Para facilitar que el registro de información se haga exteriormente se propuso este proyecto.

Se reimplementó el sistema existente a través de una aplicación Web. Para conseguir que se satisficieran los requerimientos que tiene el sistema en la aplicación Web se utilizó una combinación de las tecnologías JSP, Servlets, Axis2 (Servicios Web) y el manejador base de datos Microsoft SQL Server Express 2005. Este reporte describe el proceso de diseño e implementación de la aplicación Web.

Introducción

En la Comisión Nacional Bancaria y de Valores (CNBV), revisan que las instituciones financieras supervisadas tengan controles para que, se mantenga la integridad, confidencialidad y disponibilidad, de la información de los usuarios. En la Dirección General Supervisada de Riesgo Operacional y Tecnológico (DGSROT) cuentan con un sistema de información que tiene como objetivo almacenar y actualizar los datos que se obtienen de estas instituciones.

Actualmente el sistema tiene una arquitectura cliente-servidor. En cada PC está instalada una interfaz gráfica de usuario que envía solicitudes a un servidor donde reside la base de datos sobre la plataforma de Access. La mayor parte de la funcionalidad planteada en el diseño original de dicho sistema ya está implementada y en uso, sin embargo, están robusteciendo algunos módulos por lo que estos aún se encuentran en desarrollo.

El problema de este sistema solo puede ser accedido dentro de la Institución; sin embargo es de interés tenga un mayor alcance, para ello, se requiere implementarlo en una plataforma que pueda ser accedida desde diferentes sitios. Para resolver esta problemática el sistema se migrará a una aplicación web que pueda ser utilizada fuera de la institución utilizando tecnologías de licenciamiento libre.

Se desarrollarán servicios Web para que los empleados cuenten con una interfaz gráfica que les permita interactuar con el sistema desde cualquier sitio (a través de un navegador) con los controles de seguridad necesarios para preservar la integridad y confidencialidad de los datos.

Antecedentes

El personal de la dirección realiza visitas de supervisión a las entidades financieras donde se recaban datos relacionados a estas visitas. Los datos son alimentados al sistema Access conforme se reciben en la dirección los documentos de las visitas. El registro de la información se hace internamente debido a que el sistema Access no tiene acceso desde el exterior.

Para ofrecer una solución que permitiera mejorar la conectividad, se propuso reimplementar el sistema en una aplicación Web que brindara la posibilidad de registrar la información fuera de la institución.

Justificación

La DGSROT tiene la necesidad de acceder al sistema fuera de la CNBV, para que los empleados puedan registrar las observaciones en campo, cuando realizan visitas a las instituciones financieras.
Este proyecto utilizará una arquitectura de 3 capas y servicios Web que permita el acceso desde el exterior y a la vez fomente la escalabilidad del sistema y la seguridad.

Se utilizarán servicios Web en este proyecto porque brindan ventajas como la reutilización de código y la capacidad de ser invocados en diferentes tipos de aplicaciones cliente, sin importar el lenguaje de programación ni la plataforma sobre la que se ejecutan.

Objetivos

Objetivo general

Diseñar e implementar un sistema web para registrar, consultar y actualizar información relevante de las visitas que se realizan a las instituciones financieras por parte de la Dirección General.

Objetivos Particulares

- Diseñar las interfaces gráficas de la aplicación web.
- Diseñar y programar los módulos que integran el sistema final.

- Planear el proceso de migración de la base de datos existente a SQL Server.
- Planear el proceso de migración de la base de datos existente a SQL Server.
- Diseñar y realizar pruebas para el sistema.

Marco Teórico

Arquitectura de 3 niveles

Actualmente en el diseño de los sistemas informáticos suelen usar arquitecturas multinivel o programación por capas. Cada nivel tiene una función en especifico, lo que permite que el diseño de las arquitecturas sean escalables, esto quiere decir que se pueden ampliar con facilidad.

Normalmente se utilizan arquitecturas de 3 capas (Figura 1), las cuales son la capa de presentación, capa de negocio y la capa de datos; se describen a continuación:

1. Capa de presentación: son las interfaces gráficas de la aplicación. Esta capa se comunica solamente con la capa de negocio.
2. Capa de negocio: es donde residen los programas que se ejecutan, recibiendo las peticiones del usuario y enviando las respuestas tras el proceso. Esta capa se comunica con la capa de presentación para recibir las solicitudes y presentar los resultados, y con la capa de datos para almacenar o recuperar datos.
3. Capa de datos: es donde residen los datos, que pueden ser administrados en cualquier manejador de base de datos.

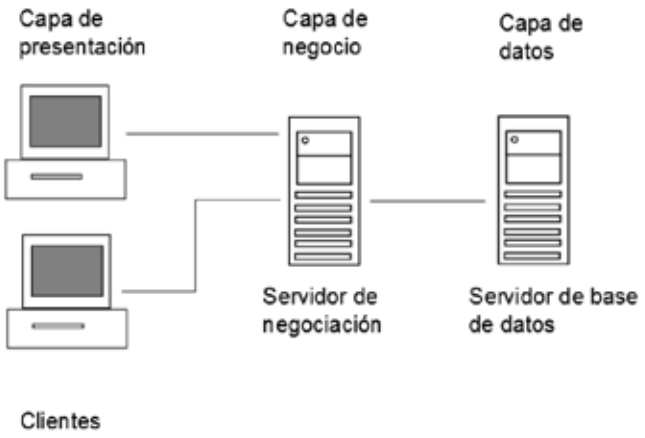

Figura1. Arquitectura de 3 capas

Aplicaciones Web

Las aplicaciones Web son un tipo especial de aplicación concebidas para funcionar en el entorno de la internet. Están desarrolladas en un modelo cliente/servidor, y utilizan el protocolo de comunicación HTTP.

El cliente de una aplicación web se encarga de solicitar recursos e información a los servidores y desplegarla en la terminal de usuario. Del otro lado el servidor pone disposición de los clientes de manera pública una serie de medios de información de tipo texto, imagen y multimedia.

Servicios Web

Los servicios Web se componen principalmente de dos estándares fundamentales: Protocolo de acceso a objetos (SOAP) y de un Lenguaje descriptivo de servicios Web (WSDL).

SOAP es el protocolo utilizado para intercambiar mensajes y datos entre aplicaciones.

WSDL tiene una sintaxis que permite describir las funciones de un servicio Web, se podría utilizar como herramienta para generar el código necesario para acceder al servicio Web o también se puede usar para ver que parámetros recibe el servicio y poder ser utilizado [2].

Axis2

Axis2 es un motor de servicios Web SOAP/WSDL. Es usado para el despliegue de los servicios Web dentro de Apache Tomcat.

SQL Server 2005 Express

El manejador de base de datos SQL Server es una solución creada para desarrollar bases de datos aplicables a cualquier aplicación profesional y su principal ventaja es que es compatible con los entornos de desarrollo y sistemas operativos de Microsoft.

La versión Express es una versión gratuita completamente funcional, pero limitada a ciertos aspectos:

- Falta de compatibilidad con características empresariales
- Límite de una CPU
- Límite de 1 GB de memoria para el grupo de búferes
- Bases de datos con un tamaño máximo de 4 GB [3]

Desarrollo del proyecto

El proyecto fue dividido en etapas conformadas por tareas específicas que permitieran ir observando los avances conseguidos tras la conclusión de cada etapa. A continuación se describen las tareas efectuadas en cada etapa del desarrollo del sistema, de manera general. Al final de esta sección se muestran algunos casos específicos de la implementación.

Análisis de requerimientos

El sistema actual cubre satisfactoriamente las necesidades de información que presenta la dirección general por el momento. Es por esto que los requerimientos exigidos para este proyecto buscan alcanzar la misma funcionalidad que en el sistema implementado en Access.

Las operaciones principales que debe administrar el sistema son: Gestión de visitas, registro de información de una entidad financiera, gestión de solicitudes, gestión de aplicaciones y gestión de catálogos. Estas operaciones fueron modeladas a través de escenarios de uso, cada uno con sus casos de usos particulares, que tienen asociado un perfil de usuario. Dependiendo del perfil de usuario el sistema muestra las acciones que puede realizar y restringe la información que puede ser accedida.

Los modelos de escenarios de uso implementados en el sistema son los siguientes:

- Gestión de Visitas
  Los usuarios se encargarán de introducir nueva información al sistema relativa a las visitas realizadas a las entidades financieras y actualizar la información ya existente. Los tipos de usuario involucrados son: Administrador y Calidad. Los casos de uso que se componen son los siguientes:

    - Registrar Visitas. Los usuarios están interesados en registrar una nueva visita realizada a una institución financiera.
    - Actualizar Visitas. El usuario actualiza la información de una visita en específico (Figura 2).

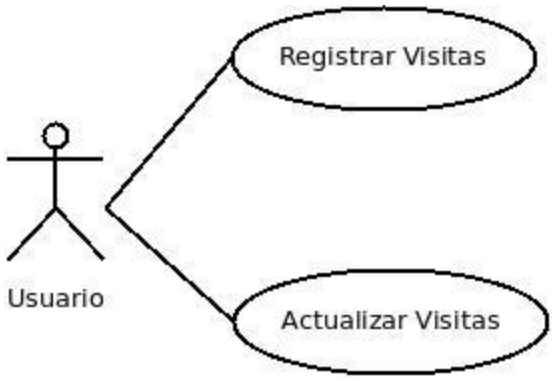

Figura 2. Gestión de Visitas

- Registro de información sobre una entidad financiera
  Los usuarios ingresan información al sistema acerca de las entidades financieras, derivada, tanto de las visitas realizadas a las entidades, como de las peticiones recibidas de parte de alguna institución. Los tipos de usuario involucrados en este escenario son: Coordinador In Situ (Registro de Observaciones, Registrar Temas Relevantes), Administrador, Coordinador Extra Situ Seguimiento (Registrar Temas Relevantes), Calidad (Registrar Temas Relevantes) y Coordinador Extra Situ Opiniones (Registro de Opiniones, Registrar Temas Relevantes). Los casos de uso que componen a este escenario son:

    - Registro de Observaciones. Los usuarios ingresan al sistema las observaciones resultantes de las anomalías detectadas durante una visita realizada a una entidad financiera.
    - Registro de Opiniones. El usuario registra los datos sobre las solicitudes de autorización, opiniones y consultas de información provenientes de las entidades financieras.
    - Registrar Temas Relevantes. Los usuarios registran información importante de la visita que realizaron a alguna entidad financiera.
    - Registro de Sanciones. Los usuarios registran sanciones de una visita, para saber posteriormente si hubo alguna multa a esa entidad financiera (Figura 3).

- Gestión de Solicitudes
  Los usuarios ingresan información al sistema de las solicitudes recibidas de parte de alguna Dirección de la misma institución, actualizan el estatus y la información de las solicitudes existentes, mientras no hayan sido entregadas, y asignan la solicitud a un Analista. Los tipos de usuario involucrados en este escenario son: Administrador, Coordinador Extra Situ Seguimiento y Coordinador Extra Situ Opiniones. Los casos de uso que componen a este escenario son:

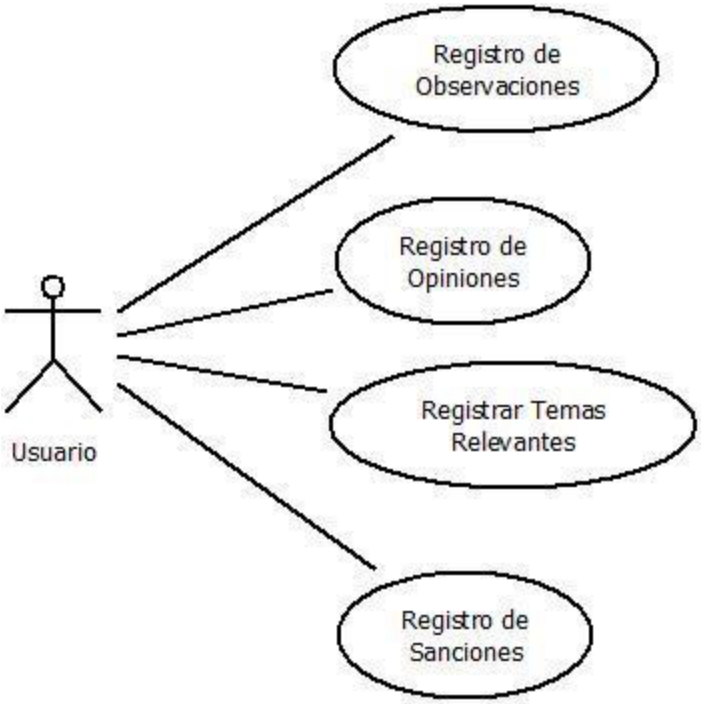

Figura 3. Registro de información sobre una entidad financiera

- Registro de Solicitudes. Los usuarios registran información sobre una solicitud hecha por una Dirección o la Oficialía de Partes. Posteriormente debe ser asignada la solicitud a un Analista (caso de uso Asignación de Solicitudes).
- Actualización de Solicitudes. Los usuarios actualizan la información y el estatus de una solicitud.
- Asignación de Solicitudes. Los usuarios asignan la solicitud a un Analista para que sea atendida (Figura 4).

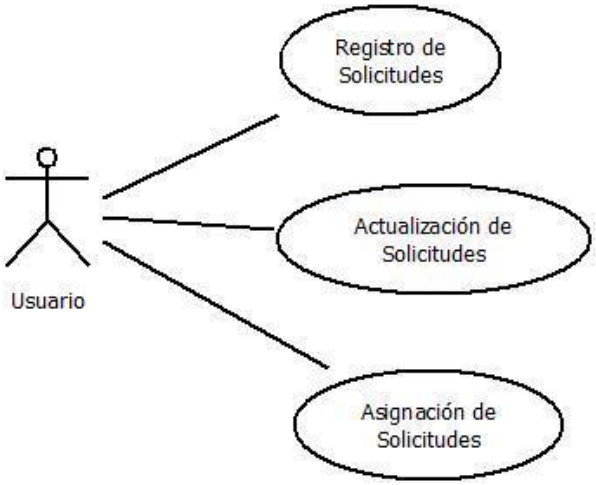

Figura 4. Gestión de Solicitudes

- Gestión de Aplicaciones
  El usuario ingresa información en el sistema sobre las aplicaciones de software que fueron analizadas en una institución financiera y actualiza la información ya registrada. Los tipos de usuario involucrados en este escenario son: Administrador, Calidad y Mantenimiento de catálogos. Los casos de uso que componen a este escenario son:

    - Registro de Aplicaciones. Los usuarios registran información sobre las aplicaciones de software que fueron revisadas en una institución financiera.
    - Actualización de Aplicaciones. Los usuarios pueden actualizar la información de las aplicaciones por si hubo algún error en el registro de la información.

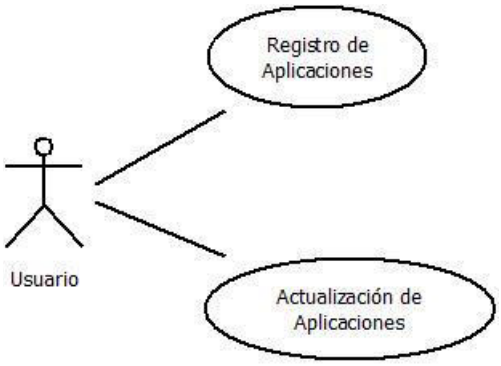

Figura 5. Gestión de Aplicaciones

- Gestión de Catálogos
  El usuario registra o actualiza información en el sistema en los catálogos de la base de datos que almacenan datos cruciales para el funcionamiento interno del sistema. Los tipos de usuario involucrados en este escenario son: Administrador, Coordinador Extra Situ Opiniones (Actualización en catálogos) y Mantenimiento de catálogos. Los casos de uso que componen a este escenario son:

    - Registro en catálogos. Los usuarios pueden ingresar nuevos registros en los catálogos de la base de datos.
    - Actualización en catálogos. Los usuarios actualizan información de un catálogo en específico.

Figura 6. Gestión de Catálogos

Diseño e implementación de las interfaces gráficas

Desde el comienzo, se tuvo un especial cuidado en el diseño de las interfaces gráficas, tanto desde el punto de vista de la usabilidad como de la presentación. Los controles de cada interfaz gráfica web debían corresponder con los controles de las interfaces gráficas existentes desarrolladas en Access. A la vez, la disposición de los controles en las páginas web del sistema debía ser lo más semejante posible a las interfaces gráficas de Access, para facilitar a los usuarios la transición de un sistema a otro.

En cuanto a la presentación gráfica del nuevo sistema web, se eligió una combinación de colores que coincidiera con los estándares de la institución, y que además resultara estética. También se seleccionaron las tipografías y disposición en pantalla de los elementos que conforman al sistema. Una vez que se eligieron estos aspectos, se procedió a programar un conjunto de hojas de estilo (CSS) que se aplican a todas las interfaces gráficas del nuevo sistema web, brindándoles un aspecto uniforme.

El diseño general de las interfaces gráficas web consistió, entonces, en replicar en todo lo posible la disposición de los controles de las interfaces gráficas Access del sistema existente, utilizando los controles disponibles en HTML, y las hojas de estilo. Siguiendo esta estrategia general, se programaron todas las interfaces gráficas del sistema, enfocándose en conseguir un buen aspecto gráfico.

Migración de la base de datos a SQL Server

El proceso de migración de la base de datos desde Access a SQL Server 2005 Express se llevó a cabo en un servidor independiente dedicado para desarrollo. Una vez que la institución determine el momento más conveniente para poner en producción el sistema web, se deberá migrar la base de datos al servidor de producción.
Para comenzar el proceso de migración se requirió instalar y configurar el servidor SQL Server 2005 Express. Se emplearon las herramientas Microsoft SQL Server Migration Assistant for Acces y Microsoft SQL Server Management Studio Express para facilitar el proceso. Ambas herramientas se encuentran disponibles en el sitio web de descargas de Microsoft de manera gratuita. La primera se utilizó para realizar la propia migración de la base de datos, y la segunda para configurar el servicio de SQL Server 2005 Express, y verificar los resultados del proceso.

Diseño de módulos del Back-End

La arquitectura del sistema web consta de cuatro capas: el nivel de vista (interfaces gráficas), el nivel de lógica de negocios (servlets), el nivel de acceso a datos (servicios

web), y el nivel de la persistencia (base de datos). La figura 7 muestra esta arquitectura general.

La arquitectura general presenta una separación entre la aplicación web y el acceso a los datos, por medio de la capa de los servicios web. Los servicios web ejecutan las consultas directamente en el servidor de base de datos, y devuelven los resultados a la capa de la lógica de negocios y de vista. Por este motivo, se decidió comenzar a diseñar e implementar los módulos correspondientes a los servicios web.

Los servicios web fueron diseñados en tres categorías generales: servicios de inserción de datos, de actualización y de consulta. La conectividad con la base de datos se realizó empleando el driver de SQL Server en una clase de nombre "DBConnection" que encapsula la conexión y proporciona acceso a la base de datos.

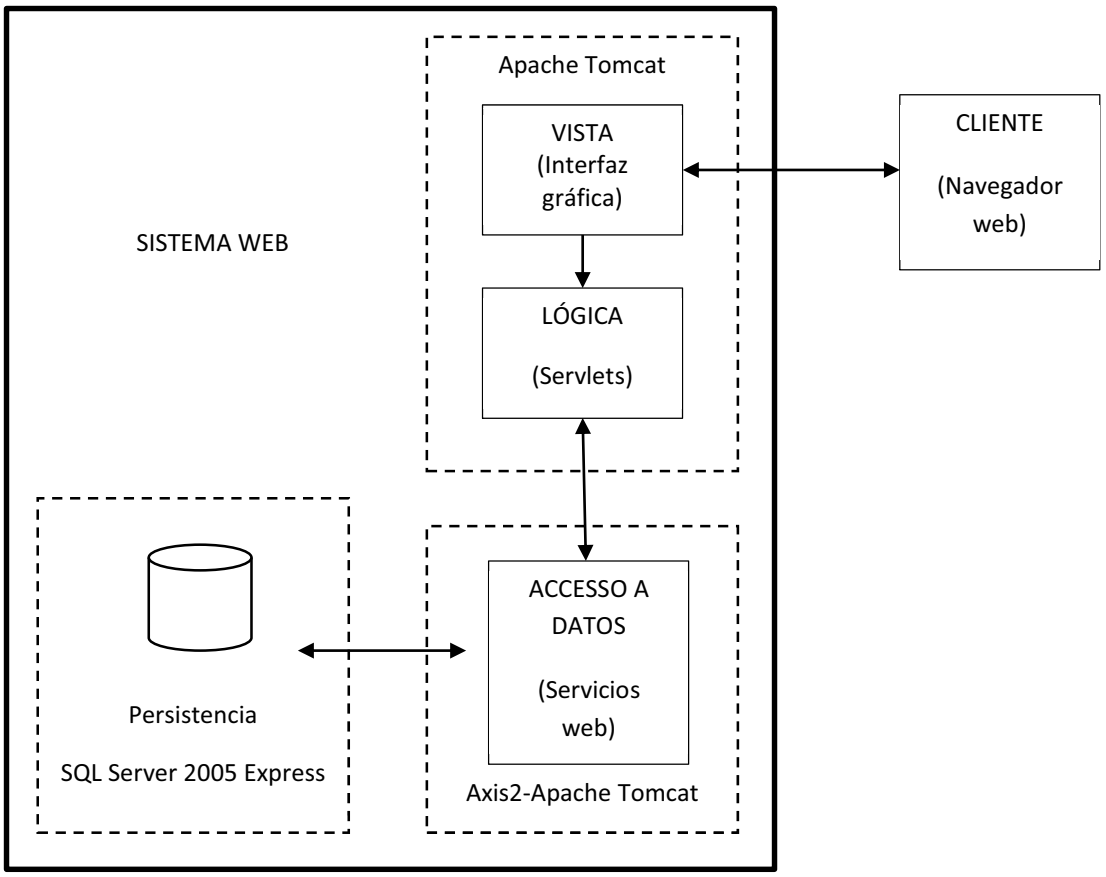

Figura 7. Arquitectura general del sistema web.

La mayoría de los métodos de los servicios web reciben parámetros de tipo String. También, los métodos que devuelven los resultados de alguna consulta a la base de datos agrupan los valores de los campos en listas de tipo String. Dependiendo de la operación requerida en determinada parte del sistema, se diseñó un método de servicio web que interactuara con la base de datos. Después de haber sido programados los servicios web, se

levantaron en el servidor Apache Tomcat utilizando la herramienta Axis2, que los despliega en el servidor y proporciona acceso tanto a sus operaciones como a su WSDL.

Por otro lado, en la parte de la lógica de negocios se implementó una clase denominada "Servicio.servicios" que se encarga de albergar los puertos de conexión con los servicios web. Los métodos de esa clase efectúan la llamada a los servicios web. Tanto los servlets de la capa de lógica de negocios como los JSPs de la capa de la vista recuperan la información de los servicios web utilizando los métodos de la clase "Servicio.Servicios". Cuando se trata del caso de servicios que recuperan información de la base de datos, algunos de los métodos de "Servicio.servicios" introducen los datos provenientes del servicio web en estructuras del tipo Arraylist, para facilitar su uso en los servlets y los JSPs que los utilizan.

Integración del Back-End y Front-End

Una vez que se concluyó con la creación del servicio web, se procedió a implementar la capa de la lógica de negocios. Como se mencionó anteriormente, esta capa está programada utilizando la tecnología de servlets, que se encargan de consumir el servicio web.

Los servlets de la capa de lógica de negocio se encargan de dos funciones principales: validar la entrada de datos proveniente de los formularios JSP de la capa de vista y establecer una comunicación con la capa de persistencia a través de la capa de los servicios web. Cuando los valores de una entrada proveniente de los formularios son inválidos, los servlets muestran un mensaje de error indicando al usuario aquellos valores cuyo valor no corresponde al esperado para determinado campo. Los valores inválidos pueden ser campos vacíos, campos con un tipo de dato distinto al esperado o campos de fecha con un formato incorrecto, entre otros.
En especial, un error que puede presentarse es el de enviar cadenas de texto con palabras reservadas del lenguaje SQL. Se programó una clase denominada "Utils.validador" que contiene, entre otros, al método "ValidarStringSQL(String str)", el cual se encarga de determinar si en el String de entrada existe alguna palabra reservada de SQL o un conjunto de caracteres que tenga un significado especial en SQL. Los servlets se encargan de evaluar si los Strings recibidos de los formularios contienen esas cadenas, y de ser así, generan un mensaje de error.

Un par de servlets "login.java" y "logout.java" son encargados de administrar los datos de la sesión del usuario. Cuando un usuario se valida ante el sistema, ingresando su identificador de cuenta de usuario y su contraseña, el formulario envía los valores al servlet "login.java", el cual a su vez, tomará esos valores para hacer una llamada al método "ValidarUsuario()" del servicio web. El método devolverá los datos de inicio de sesión cuando la validación del usuario ante el sistema haya sido exitosa. Los datos de inicio de sesión se almacenan en un objeto de tipo "DatosSesion.DatosSesion", que se encuentra en el ámbito de la sesión. Cuando se cierra la sesión, el servlet "logout.java" destruye esta información.

El siguiente paso consistió en completar las funciones dinámicas que deben ofrecer las interfaces gráficas producidas en la etapa de diseño. Estas funciones dinámicas fueron, por lo general, recuperar la información proveniente de la base de datos que sirve para llenar los valores de listas y otros controles de forma dinámica.

La capa de la vista está conformada por archivos JSP. Para implementarla, se tomó como base los archivos HTML generados como producto de la etapa del diseño e implementación de interfaces gráficas. A las interfaces gráficas en código HTML se les agregó el comportamiento dinámico haciendo uso de llamadas a los servicios web e implementando el manejo de sesiones.

La estructura general de las interfaces gráficas dinámicas JSP utilizadas como formularios es la siguiente:

- La presentación de la interfaz gráfica y la disposición de los controles se hace en HTML
- Cuando los formularios requieren cargar datos provenientes de la base de datos, se hace uso de los servicios web y de javascript. A través de los eventos de javascript se recarga el formulario, pero haciendo la solicitud de datos a los servicios web para llenar cualquier campo que sea necesario.
- Los formularios incluyen un botón principal que llama a algún servlet encargado de realizar la llamada final a los servicios web. El botón principal puede tener etiquetas como “Guardar” o “Actualizar”, que le indican al usuario que se ha concluido con la captura de los datos necesarios en una determinada parte del sistema.

Adicional a los formularios, se diseñó un tipo especial de JSPs que se encargan sólo de mostrar información: los reportes. Los reportes muestran los datos por medio de tablas, respetando un diseño gráfico especial que los distingue de los formularios. Los reportes reciben los parámetros que se enviarán a los servicios web a través de los datos en el request. Hacen la llamada al servicio web y recuperan la información que desplegarán.

Pruebas al sistema

La fase de pruebas se dividió en dos pruebas generales, aplicadas a todo el sistema:

- *Prueba de servicio web*. Se hicieron llamadas a los métodos del servicio web y se verificó que se efectuaran los cambios esperados en la base de datos. Se realizó esta prueba a través del mismo IDE de Netbeans, que incluye la posibilidad de probar la operación de los métodos del servicio web, mostrando las respuestas del servicio en el navegador.
- *Prueba de integración interfaz gráfica - lógica de negocios - persistencia.* Se probó el funcionamiento correcto de todas las interfaces gráficas una vez que se tenía integrado el sistema. En estas pruebas se realizaron diversas entradas en los formularios, con valores en el rango normal esperado, y con valores incorrectos, para verificar que el sistema se comportara de acuerdo a lo planeado. Se comprobó que la secuencia del flujo de la navegación fuera correcta y, una vez más, que la

base de datos reflejara los cambios que se deben realizar por medio de la interfaz gráfica del sistema.

En ambos casos, las pruebas fueron en general exitosas. Existieron casos específicos de servicios o interfaces gráficas que no presentaban el comportamiento adecuado, debido a algún error en la implementación. Los resultados de estas pruebas permitieron corregir los errores detectados.

Resultados

A continuación se muestran algunos ejemplos de uso y funcionamiento del sistema.
La figura 8 muestra un detalle de la pantalla inicial de bienvenida. En esta pantalla el usuario debe introducir su usuario y contraseña, mismos que fueron asignados por el Administrador de acuerdo a su perfil de usuario. Como ya se mencionó, el usuario y la contraseña serán validados desde el servicio Web.

Figura 8. Login

En caso de que el usuario sea incorrecto nos mostrará el siguiente mensaje (Figura 9):

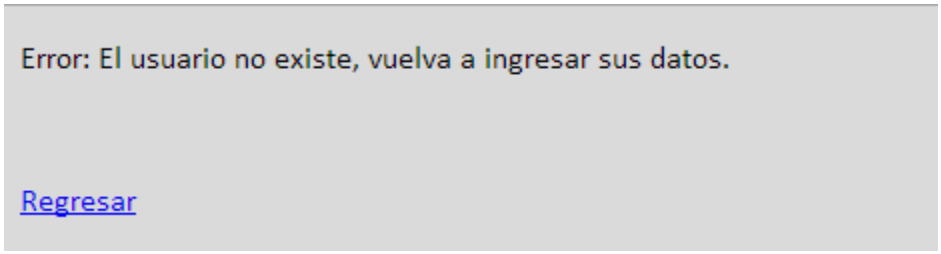

Figura 9. Mensaje de error

En caso de que el usuario se haya equivocado solo en su contraseña nos mostrará el sistema el siguiente mensaje (Figura 10):

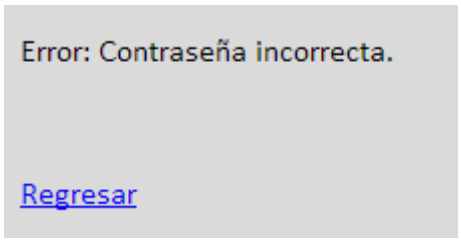

Figura 10. Contraseña incorrecta

Si son correctos los datos, se mostrará un menú dependiendo del perfil que tenga el usuario, en este ejemplo el usuario tiene permisos de Administrador y nos muestra la pantalla (Figura 11).

Figura 11. Pantalla de bienvenida

A continuación se muestra el flujo de la opción Otras Opciones->Inventario de Aplicaciones, que nos muestra todas las aplicaciones que tiene una entidad, en este caso esta seleccionada “Entidad otra” y solo tiene una aplicación como se puede ver en la Figura 12.

Figura 12. Inventario de Aplicaciones

El siguiente flujo (figura 13) muestra el registro de las Observaciones de una visita, cuando el usuario elije un núm. de visita y si no tiene rubros registrados con su riesgo, se debe registrar el alcance y el riesgo de la visita.

Figura 13. Registro de Observaciones

Al darle clic en el botón de “Registrar alcance y riesgo de la visita” se muestra la pantalla de la Figura 13 y se debe seleccionar de la lista los rubros que fueron revisados en la visita que se realizó, con su riesgo respectivamente y se le da clic en agregar, todos los elementos agregados se pasan a una segunda lista de rubros revisados, Se le da clic en guardar y nos despliega el sistema un mensaje que el registro fue insertado (Figura 14).

Figura 14. Alcance y riesgo de la visita

Una vez registrado el alcance y riesgo de la visita nos muestra la siguiente pantalla (Figura 15) para capturar cada observación con su riesgo. Observe que la figura 16 tiene la lista de los datos de cada rubro que se inserto anteriormente en la pantalla de la Figura 14.

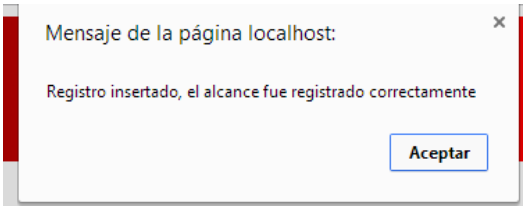

Figura 15. Registro exitoso

Figura 16. Registro de Observaciones por visita actualizado

Análisis y discusión de resultados

Se comprobó que el manejo de las sesiones por parte del sistema funciona adecuadamente. Cuando se accede con una cuenta valida el sistema siempre recupera el perfil del usuario y las opciones que le corresponden. A nivel de los JSPs la validación de la sesión impide que un usuario no autorizado acceda a información restringida para su perfil.

Para comprobar la seguridad del sistema se realizaron varios ataques de inyección de SQL ante los cuales el sistema respondió de la manera esperada sin permitir que los ataques tuvieran éxito.

Cada operación del servicio Web fue probada con ayuda del IDE de NetBeans. Se utilizó la función "Test Operation in Browser", que hace una llamada a la operación del servicio Web con parámetros fijos y despliega el resultado en el navegador. La figura 17 muestra el resultado de una llamada a la operación ObtenerAnalista del servicio Web.

Figura 17. Llamada al servicio Web

Al final de la integración del sistema, pudo comprobarse que los servlets de la capa de lógica de negocios comunican correctamente a los JSPs con los servicios Web. La funcionalidad de los JSPs pudo completarse gracias a que los servlets obtienen la información proveniente de los servicios Web de forma cabal y exacta.

Conclusiones

En este proyecto se cumplieron satisfactoriamente los objetivos planteados. Se desarrollaron las interfaces gráficas en HTML y posteriormente se trasladaron a JSP, facilitándose así la implementación de las funcionalidades que debía prestar cada interfaz gráfica. La creación de hojas de estilo CSS permitió utilizar el mismo formato gráfico en todas las páginas.

El proceso de migración de la base de datos también se completó exitosamente. El empleo de las herramientas Microsoft SQL Server Migration Assistant for Access y SQL Server Management Studio Express facilitaron significativamente el proceso de migración, lo que se tradujo en un importante ahorro de tiempo.

Se diseñó y programó un servicio Web que comunicara la aplicación con el SGBD. En el servicio Web se diseñaron métodos específicos con consultas SQL especializadas en atender cada tarea de gestión de datos que se efectúa en el sistema.

En el lado del servidor web se desarrolló una capa de comunicación con el servicio web basada en servlets. Los datos de sesión, validación de entradas de formularios, y el control del flujo de la navegación también se implementaron en este nivel. De esta forma, los servlets actúan como un puente de enlace entre la vista que se presenta al usuario y la persistencia de datos. Con este enfoque se consiguió un aislamiento completo entre el servidor web y el servidor de base de datos, favoreciendo la seguridad del sistema.

Al concluir el proceso de desarrollo y prueba de los módulos, el sistema Web se integró exitosamente y ahora esta en condiciones de ofrecer todas las funcionalidades que se plantearon al inicio del proyecto.

Referencias bibliográficas

[1] Falkner, Jayson y Jones, Kevin, “Servlets and Java Server Pages. The J2EE Web Tier”, Adison-Wesley, Unitede States of America, 2004.
[2] Guía Breve de Servicios Web. Guía Breve de Servicios Web. (en línea) http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb
[3] ”Información general de SQL Server 2005 Express Edition”(en línea), http://msdn.microsoft.com/es-es/library/ms345154(v=sql.90).aspx
[4] Alonso Cebrián, José María, “Ataques a bases de datos, SQL Injection”, Universitat Oberta de Catalunya, 2005

Entregables

ServletActualizarDG.java

```
package servlets;

import Servicio.servicios;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class ServletActualizarDG extends HttpServlet {

     @Override
     protected    void    doGet(HttpServletRequest    request,
HttpServletResponse response)
               throws ServletException, IOException {
          response.setContentType("text/html;charset=UTF-8");

     response.getWriter().println(ErrorHandler.HTMLEnvolveMsg
(ErrorHandler.ErrorMsg(ErrorHandler.ErrorGet)));
     }

     @Override
     protected   void   doPost(HttpServletRequest   request,
HttpServletResponse response)
               throws ServletException, IOException {

          response.setContentType("text/html;charset=UTF-8");

          PrintWriter out = response.getWriter();

          //Invalida caché del explorador
          response.setHeader("Pragma", "no-cache");
          response.addHeader("Cache-Control",          "must-
revalidate");
          response.addHeader("Cache-Control", "no-cache");
          response.addHeader("Cache-Control", "no-store");
          response.setDateHeader("Expires", 0);

          if
(!(request.getParameter("txt_des_larga").equals("")        ||
request.getParameter("txt_des_corta").equals("")           ||
request.getParameter("sel_Vis").equals(""))) {
```

```
     servicios.ActualizarDG(request.getParameter("sel_descL")
,                       request.getParameter("txt_des_corta"),
request.getParameter("txt_des_larga"),
request.getParameter("sel_Vis"));
               out.println("<html>\n"
                         + "  <head>\n"
                         + "       <link
href=\"../HojasEstilo/estilos.css\"        rel=\"stylesheet\"
type=\"text/css\">\n"
                         + "       <title></title>\n"
                         + "  </head><body
onload=\"javascript:alert('Registro  actualizado,  la  DG  fue
actualizada    correctamente');    location.replace('"    +
request.getContextPath()                                    +
"/Catalogos_Actu/actu_dg.jsp');\">"
                         + "</body></html>");

          } else {
               out.println("<html>\n"
                         + "  <head>\n"
                         + "       <link
href=\"../HojasEstilo/estilos.css\"        rel=\"stylesheet\"
type=\"text/css\">\n"
                         + "       <title></title>\n"
                         + "  </head><body
onload=\"javascript:alert('Parámetros
inválidos');history.back();\"></body></html>");
          }
     }
}
```

ServletInsertarFederacion.java

```
package servlets;

import Servicio.servicios;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class ServletInsertarFederacion extends HttpServlet {
```

```
    @Override
    protected    void    doGet(HttpServletRequest    request,
HttpServletResponse response)
                throws ServletException, IOException {
          response.setContentType("text/html;charset=UTF-8");

    response.getWriter().println(ErrorHandler.HTMLEnvolveMsg
(ErrorHandler.ErrorMsg(ErrorHandler.ErrorGet)));
    }

    @Override
    protected   void   doPost(HttpServletRequest   request,
HttpServletResponse response)
                throws ServletException, IOException {
          response.setContentType("text/html;charset=UTF-8");

          PrintWriter out = response.getWriter();

          //Invalida caché del explorador
          response.setHeader("Pragma", "no-cache");
          response.addHeader("Cache-Control",          "must-
revalidate");
          response.addHeader("Cache-Control", "no-cache");
          response.addHeader("Cache-Control", "no-store");
          response.setDateHeader("Expires", 0);

          if
(!(request.getParameter("txt_nameLFed").equals("")         ||
request.getParameter("txt_nameSFed").equals(""))) {


    servicios.InsertarFederacion(request.getParameter("txt_n
ameLFed"), request.getParameter("txt_nameSFed"));
                //manda una página

                out.println("<html>\n"
                          + "  <head>\n"
                          + "       <link
href=\"../HojasEstilo/estilos.css\"        rel=\"stylesheet\"
type=\"text/css\">\n"
                          + "       <title></title>\n"
                          + "  </head><body
onload=\"javascript:alert('Registro insertado, la Federación
fue                                        registrada
correctamente');history.back();\"></body></html>");

          } else {
```

```
                out.println("<html>\n"
                          + "  <head>\n"
                          + "       <link
href=\"../HojasEstilo/estilos.css\"        rel=\"stylesheet\"
type=\"text/css\">\n"
                          + "       <title></title>\n"
                          + "  </head><body
onload=\"javascript:alert('Parámetros
inválidos');history.back();\"></body></html>");
          }
     }
}
```

ServletActualizarPerfil.java

```
package servlets;

import Servicio.servicios;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class ServletActualizarPerfil extends HttpServlet {

     @Override
     protected    void    doGet(HttpServletRequest    request,
HttpServletResponse response)
               throws ServletException, IOException {
          response.setContentType("text/html;charset=UTF-8");

     response.getWriter().println(ErrorHandler.HTMLEnvolveMsg
(ErrorHandler.ErrorMsg(ErrorHandler.ErrorGet)));
     }

     @Override
     protected    void    doPost(HttpServletRequest    request,
HttpServletResponse response)
               throws ServletException, IOException {

          response.setContentType("text/html;charset=UTF-8");

          PrintWriter out = response.getWriter();

          //Invalida caché del explorador
```

```
          response.setHeader("Pragma", "no-cache");
          response.addHeader("Cache-Control",          "must-
revalidate");
          response.addHeader("Cache-Control", "no-cache");
          response.addHeader("Cache-Control", "no-store");
          response.setDateHeader("Expires", 0);

          if
(!(request.getParameter("txt_desc_nu").equals(""))) {


     servicios.ActualizarPerfil(request.getParameter("sel_des
c"), request.getParameter("txt_desc_nu"));
               out.println("<html>\n"
                         + "  <head>\n"
                         + "       <link
href=\"../HojasEstilo/estilos.css\"        rel=\"stylesheet\"
type=\"text/css\">\n"
                         + "       <title></title>\n"
                         + "  </head><body
onload=\"javascript:alert('Registro  actualizado,  el  Perfil
fue   actualizado   correctamente');   location.replace('"   +
request.getContextPath()                                    +
"/Catalogos_Actu/actu_perfil.jsp');\">"
                         + "</body></html>");

          } else {
               out.println("<html>\n"
                         + "  <head>\n"
                         + "       <link
href=\"../HojasEstilo/estilos.css\"        rel=\"stylesheet\"
type=\"text/css\">\n"
                         + "       <title></title>\n"
                         + "  </head><body
onload=\"javascript:alert('Parámetros
inválidos');history.back();\"></body></html>");
          }
     }

}
```

registro_opiniones.jsp

```
<%   //Invalida caché del explorador
    response.setHeader("Pragma", "no-cache");
    response.addHeader("Cache-Control", "must-revalidate");
    response.addHeader("Cache-Control", "no-cache");
```

```
    response.addHeader("Cache-Control", "no-store");
    response.setDateHeader("Expires", 0);
%>
<html>
    <head>
        <jsp:include page="/verificaSesion" />
        <link               href="../HojasEstilo/estilos.css"
rel="stylesheet" type="text/css">
        <title></title>
        <script lang="javascript" type="text/javascript">

            function validar()
            {

              if (document.getElementById("txt_coment").value
=== "")
                {
                    alert("Favor de capturar la observación
de la opinión");
                    return;
                }

document.getElementById("Frm_RegOpiniones").submit();
            }

            function limpiar()
            {
                document.getElementById("txt_coment").value =
"";

            }

            function cargar()
            {
                document.getElementById("escondido").value  =
document.getElementById("sel_Sector").value;
                document.getElementById("escondido1").value =
document.getElementById("sel_Entidad").value;
                document.getElementById("frmOc").submit();
            }
        </script>
    </head>


<body>

    <%
```

```
        DatosSesion.DatosSesion           dsesion           =
(DatosSesion.DatosSesion)
request.getSession(false).getAttribute("dsesion");
        int perfil = dsesion.getPerfil();
        if  (perfil  ==  3  ||  perfil  ==  6  ||  perfil  ==  7  ||
perfil == 8 || perfil == 9 || perfil == 10) {

    %>

    <br><br>
    <h1>Control de opiniones</h1>
        <br><br>

        <table Style="width:100%" border="0">

            <tr>
                <td Style="width:20%">   </td>

                <td>
                    <form               id="Frm_RegOpiniones"
action="../ServletRegistrarOpinion" method="post">
                        <table            class="tabla_ancho"
border="0">
                            <tr>

                                <td     Style="width:15%"><p>
Sector </p> </td>
                                <%
java.util.ArrayList<String>              lista              =
Servicio.servicios.ObtenerSector();%>
                                <td>
                                    <%
                                        float idSel; //id del
elemento seleccionado en la lista
                                        if
(request.getParameter("escondido") != null) {
                                            idSel           =
Float.parseFloat(request.getParameter("escondido"));
                                        } else {
                                            idSel           =
Float.parseFloat(lista.get(0));
                                        }

                                    %>
                                    <select   id="sel_Sector"
name="sel_Sector" onchange="javascript:cargar()">
```

```
                                        <% for (int i = 0; i
< lista.size(); i += 2) {%>
                                            <option
value="<%=          lista.get(i)%>"          <%          if
(Float.parseFloat(lista.get(i)) == idSel) {

out.print("selected=\"selected\"");
                                            }%>>       <%
out.print(lista.get(i + 1));%>
                                        </option>
                                        <% }%>
                                    </select>

                                </td>
                            </tr>

                            <tr>

                                <td><p> Entidad </p></td>
                                <%
java.util.ArrayList<String>             lista1             =
Servicio.servicios.ObtenerEntidad("" + idSel);%>
                                <td>
                                     <%
                                     float  idSel1;  //id  del
elemento seleccionado en la lista
                                        if
(request.getParameter("escondido1") != null) {
                                            idSel1          =
Float.parseFloat(request.getParameter("escondido1"));
                                        } else {
                                            idSel1          =
Float.parseFloat(lista1.get(0));
                                        }

                                    %>
                                    <select  id="sel_Entidad"
name="sel_Entidad">

                                        <% for (int i = 0; i
< lista1.size(); i += 2) {%>
                                            <option
value="<%=          lista1.get(i)%>"          <%          if
(Float.parseFloat(lista1.get(i)) == idSel1) {

out.print("selected=\"selected\"");
```

```
                                                }%>>        <%
out.print(lista1.get(i + 1));%>
                                            </option>
                                            <% }%>
                                        </select>

                                    </td>
                                </tr>

                                <tr>

                                    <td><p>  Tipo  de  Opinión</p>
</td>
                                     <%
java.util.ArrayList<String>              lista2              =
Servicio.servicios.ObtenerTipoOpinion();%>
                                    <td>

                                        <select    id="sel_TipOp"
name="sel_TipOp">

                                            <% for (int i = 0; i
< lista2.size(); i += 2) {%>
                                            <option    value="<%=
lista2.get(i)%>" >
                                                <%
out.print(lista2.get(i + 1));%>
                                            </option>
                                            <% }%>
                                        </select>
                                     </td>

                                </tr>

                                <tr>
                                    <td><p>    Descripción    </p>
</td>
                                    <td><br>
                                        <textarea  id="txt_coment"
type="text" name="txt_coment" rows="6" cols="60"></textarea>
                                    </td>
                                </tr>

                                <tr>
                                    <td>
                                         
                                    </td>
```

```
                            </tr>

                            <tr>
                                <td                colspan="2"
class="tabla_centrada">
                                    <input       type="button"
name="btn_guardar"                              value="Guardar"
onclick="javascript:validar()">
                                     
                                    <input       type="button"
name="btn_limpiar"                              value="Limpiar"
onclick="javascript:limpiar()">

                                </td>
                            </tr>

                        </table>
                    </form>

                </td>

                <td Style="width:20%">   </td>
            </tr>

        </table>
        <form      id="frmOc"      method="post"      action
="registro_opiniones.jsp">
            <input type="hidden" value ="dfdf" id="escondido"
name="escondido" />
            <input       type="hidden"       id="escondido1"
name="escondido1" />
        </form>

        <% } else {%>
        <p>Error en la petición al sistema</p>
        <% }%>


</body>
</html>
```

actu_federacion.jsp

```
<%   //Invalida caché del explorador
    response.setHeader("Pragma", "no-cache");
    response.addHeader("Cache-Control", "must-revalidate");
    response.addHeader("Cache-Control", "no-cache");
```

```
    response.addHeader("Cache-Control", "no-store");
    response.setDateHeader("Expires", 0);
%>
<html>
    <head>
        <jsp:include page="/verificaSesion" />
        <link               href="../HojasEstilo/estilos.css"
rel="stylesheet" type="text/css">
        <title></title>
        <script lang="javascript" type="text/javascript">
               function validar()
               {
                    if
(document.getElementById("txt_Nom_larg").value === "")
                    {
                         alert("Favor  de  capturar  el  nombre
largo");
                         return;
                    }


                    if
(document.getElementById("txt_Nom_cor").value === "")
                    {
                         alert("Favor  de  capturar  el  nombre
corto");
                         return;
                    }


     document.getElementById("Frm_ActuFed").submit();
               }

               function limpiar()
               {

     document.getElementById("txt_Nom_larg").value = "";

     document.getElementById("txt_Nom_cor").value = "";

               }

               function cargar()
               {

     document.getElementById("escondido").value             =
document.getElementById("sel_desc").value;
```

```
     document.getElementById("frmOc").submit();
               }

        </script>
    </head>

    <body>
        <%
            DatosSesion.DatosSesion         dsesion         =
(DatosSesion.DatosSesion)
request.getSession(false).getAttribute("dsesion");
            int perfil = dsesion.getPerfil();
            if (perfil == 3 || perfil == 5 || perfil == 9 ||
perfil == 10) {
        %>
        <br><br>
        <h1>Actualización Federaciones</h1>
        <br><br>

        <table style="width:100%">
            <tr>

                <td style="width:20%">   </td>

                <td>
                    <form                     id="Frm_ActuFed"
action="../ServletActualizarFederacion" method="post">
                        <table class="tabla_ancho">
                            <tr>

                                <td
style="width:10%"><p>Descripción(Corta)</p></td>
                                <%
java.util.ArrayList<String>              lista              =
Servicio.servicios.ObtenerFederacion();%>
                                <td>
                                    <%
                                        int  idSel;  //id  del
elemento seleccionado en la lista
                                        if
(request.getParameter("escondido") != null) {
                                            idSel           =
Integer.parseInt(request.getParameter("escondido"));
                                        } else {
                                            idSel           =
Integer.parseInt(lista.get(0));
```

```
                                    }

                                %>

                                <select     id="sel_desc"
name="sel_desc" onchange="javascript:cargar()">
                                    <% for (int
i = 0; i < lista.size(); i += 2) {%>
                                    <option
value="<%=          lista.get(i)%>"          <%          if
(Integer.parseInt(lista.get(i)) == idSel) {

     out.print("selected=\"selected\"");
                                                }%>> <%
out.print(lista.get(i + 1));%>
                                    </option>
                                    <% }%>
                                </select>
                            </td>
                        </tr>
                        </tr>
                        <tr>
                            <td colspan="2"><hr/></td>
                        </tr>
                        <tr>
                            <td        colspan="2"><br><p
class="negrita">Actualización de datos</p></td>
                        </tr>
                        <tr>
                            <td colspan="2"> </td>
                        </tr>

                        <% String nombreC = "", nombreL =
"";
                            java.util.ArrayList<String>
datos;
                            if
(request.getParameter("escondido") != null) {
                                datos                   =
Servicio.servicios.ObtenerDatosFederacion(request.getParamete
r("escondido"));

                            } else {
                                datos                   =
Servicio.servicios.ObtenerDatosFederacion("" + idSel);

                            }
```

```
                                nombreC = datos.get(0);
                                nombreL = datos.get(1);


                            %>
                            <tr>
                                <td
style="width:10%"><p>Nombre(largo)</p></td>        <td><input
type="text"  size="90"  id="txt_Nom_larg"  name="txt_Nom_larg"
value='<%= nombreL%>'/></td>
                            </tr>
                            <tr>
                                <td
style="width:10%"><p>Nombre(corto)</p></td>        <td><input
type="text"   size="80"   id="txt_Nom_cor"   name="txt_Nom_cor"
value='<%= nombreC%>'/></td>
                            </tr>

                            <tr>
                                <td colspan="2"> </td>
                            </tr>
                            <tr>
                                <td               colspan="2"
class="tabla_centrada">
                                    <input      type="button"
value="Guardar"                          name="btn_Guardar"
onclick="javascript:validar()"/> 
                                    <input      type="button"
value="Limpiar"                             name="btn_limp"
onclick="javascript:limpiar()"/>
                                </td>
                            </tr>
                        </table>

                    </form>
                </td>

                <td Style="width:20%">   </td>
            </tr>
        </table>

        <form      id="frmOc"      method="post"      action
="actu_federacion.jsp">
            <input type="hidden" value ="dfdf" id="escondido"
name="escondido" />
        </form>
```

```
        <% } else {%>
        <p>Error en la petición al sistema</p>
        <% }%>
    </body>
</html>
```

registro_dg.jsp

```
<%   //Invalida caché del explorador
    response.setHeader("Pragma", "no-cache");
    response.addHeader("Cache-Control", "must-revalidate");
    response.addHeader("Cache-Control", "no-cache");
    response.addHeader("Cache-Control", "no-store");
    response.setDateHeader("Expires", 0);
%>
<html>
    <head>
        <jsp:include page="/verificaSesion" />
        <link               href="../HojasEstilo/estilos.css"
rel="stylesheet" type="text/css">
        <title></title>
        <script lang="javascript" type="text/javascript">
            function validar()
            {
                if
(document.getElementById("txt_descL").value === "")
                {
                    alert("Favor de capturar el nombre largo
de la DG");
                    return;
                }


                if
(document.getElementById("txt_descS").value === "")
                {
                    alert("Favor de capturar el nombre corto
de la DG");
                    return;
                }
                 if (document.getElementById("sel_vis").value
=== "-1")
                {
                    alert("Favor    de    seleccionar    la
Vicepresidencia");
                    return;
                }
```

```
document.getElementById("Frm_RegDG").submit();

            }

            function limpiar()
            {
                document.getElementById("txt_descL").value   =
"";
                document.getElementById("txt_descS").value   =
"";
            }

        </script>
    </head>

    <body>
        <%
            DatosSesion.DatosSesion         dsesion         =
(DatosSesion.DatosSesion)
request.getSession(false).getAttribute("dsesion");
            int perfil = dsesion.getPerfil();
            if (perfil == 3 || perfil == 5 || perfil == 6 ||
perfil == 7 || perfil == 9 || perfil == 10) {
        %>
        <br><br>
        <h1>Registro de Dirección General</h1>
        <br><br>

        <table style="width:100%">
            <tr>

                <td style="width:20%">   </td>

                <td>
                    <form                       id="Frm_RegDG"
action="../ServletInsertarDG" method="post">
                        <table class="tabla_ancho">
                            <tr>
                                <td><p>Nombre    (Descripción
larga)</p></td>     <td><input     type="text"      size="60"
id="txt_descL" name="txt_descL"/></td>
                            </tr>
                            <tr>
                                <td><p>Nombre    (Descripción
corta)</p></td>     <td><input     type="text"      size="60"
id="txt_descS" name="txt_descS"/></td>
```

```
                            </tr>
                            <tr>

<td><p>Vicepresidencia</p></td>
                                <%
java.util.ArrayList<String>              lista              =
Servicio.servicios.ObtenerVicepresidencia();

                                lista.add(0,"--Selecciona una
opción--");
                                lista.add(0,"-1");

                                %>
                                <td>
                                    <select      id="sel_vis"
name="sel_vis">
                                        <% for (int i = 0; i
< lista.size(); i += 2) {%>
                                        <option    value="<%=
lista.get(i)%>">   <%   out.print(lista.get(i   +   1));   %>
</option>
                                        <% } %>
                                    </select>
                                </td>
                            </tr>
                            <tr>
                                <td colspan="2"> </td>
                            </tr>
                            <tr>
                                <td                colspan="2"
class="tabla_centrada">
                                    <input       type="button"
value="Guardar"        name="btn_Guardar"        id="guardar"
onclick="javascript:validar()"/> 
                                    <input       type="button"
value="Limpiar"           name="btn_limp"           id="clean"
onclick="javascript:limpiar()"/>
                                </td>
                            </tr>
                        </table>
                    </form>
                </td>

                <td Style="width:20%">   </td>
            </tr>
        </table>
        <% } else { %>
```

```
          <p>Error en la petición al sistema</p>
     <% }%>
    </body>
</html>
```

Fragmento de servicios.java

```
public class servicios {

     public static void InsertarAnalista(String name, String
     id) {
          insertarAnalista(name, id);
         }

         public static ArrayList<String> ObtenerPerfiles() {
             return new ArrayList<String>(obtenerPerfiles());
         }

         public static ArrayList<String> ObtenerDG() {
             return new ArrayList<String>(obtenerDG());
         }

         public static ArrayList<String> ObtenerAnalista() {
             return       new       ArrayList       <String>
     (obtenerAnalista());
         }
     }
```