Universidad Autónoma Metropolitana

Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Sistema de gestión para la oficina parroquial de San Juan Bautista Coyoacán

Roldan Ramírez Patricia 205206920

Oliva García Emmanuel 204204307

Trimestre 2013 Primavera

Asesor: Maricela Claudia Bravo Contreras, Asociado, Departamento de Sistemas

TABLA DE CONTENIDO

1. Introducción

El registro de ceremonias es un procedimiento que se hace cada vez que un cliente solicita una ceremonia esta tarea la realiza el personal administrativo de las oficinas parroquiales, en algunas oficinas esta tarea se lleva a cabo de forma manual haciendo que este procedimiento requiera de consultar en la agenda la disponibilidad de horarios para asignar una ceremonia así como generar de forma manual el comprobante de registro.

En la oficina de la Parroquia de San Juan Bautista Coyoacán actualmente existe el sistema "PROVINCIA FRANCISCANA (PARROQUIA DE SAN BAUTISTA)" el cual es una aplicación de escritorio que realiza la asignación y el registro de ceremonias, para administrar la información de este sistema se utiliza una base de datos. Este sistema cuenta con algunas deficiencias, entre estas se encuentran:

1. Cuando un cliente solicita ***n*** ceremonias por el mismo concepto (intención) se capturan ***n*** veces los datos requeridos para la intención, al momento de emitir el recibo de pago si este concepto fue de ***n*** veces se necesitan imprimir los ***n*** recibos de pago lo cual genera un gasto en hojas para la impresión.
2. Para generar la información de algunos tipos de reportes, debido a que no hay forma de extraer toda la información que se requiere, se tiene que generar en hojas de cálculo por separado (Excel, Open Office, etc.) y aplicar manualmente técnicas de filtrado para generar diferentes tipos de reporte.
3. Al ser una aplicación de escritorio solamente un usuario puede manipular la información, no tiene portabilidad ya que solo se puede instalar en ambiente Windows XP.

La aplicación web desarrollada permitirá realizar de manera eficiente el registro de ceremonias solicitadas y corregir las deficiencias mencionadas, al ser un sistema web, el usuario tendrá acceso a él desde cualquier navegador y en cualquier momento. La aplicación permitirá asignar una o varias ceremonias solicitadas por un cliente. Además de permitir generar reportes para el control.

1.1. Objetivo

Diseñar e implementar un sistema que administra las solicitudes y el registro de las celebraciones eucarísticas (misas ó ceremonias) de una oficina parroquial. Para llevar a cabo esto es necesario el diseño y construcción de los siguientes puntos:

- Base de datos relacional que almacene la información de las ceremonias.
- Base de datos relacional que almacene los datos de usuarios con diferentes roles para el acceso al sistema.
- Base de datos relacional que almacene los datos del archivo histórico digitalizado.

Así como la implementación de módulos que permitan:

- El acceso al sistema.
- Administrar el catálogo de usuarios para el acceso al sistema.
- Administrar los catálogos requeridos para el sistema, que permita realizar altas, bajas, cambios y consultas.
- Generar los reportes requeridos por el usuario del sistema, como son reportes económicos, reportes de ceremonias por día.
- El acceso y administración de los datos del archivo histórico.
- Diseñar e implementar un módulo que permita la emisión de recibos de pago por ceremonias.
- Diseñar e implementar un módulo que permita generar gráficos estadísticos.

1.2. Resumen

La realización de aplicaciones de carácter administrativo que permiten ahorrar tiempo y esfuerzo a medida de que lo que actualmente se hace de manera manual pueda automatizarse brindando el apoyo en las actividades del personal administrativo de una oficina.

Al ser un sistema web el usuario podrá tener acceso a él desde cualquier navegador y en cualquier momento permitiendo la portabilidad del sistema el cual contara con distintos módulos para realizar la asignación de una ceremonia.

El sistema tendrá una primera interfaz que será para acceder al sistema mediante un usuario y su respectiva contraseña una vez que el usuario fue validado se mostrará un menú donde el usuario podrá elegir cualquier módulo que tendrá el sistema de acuerdo a los permisos asignados por él administrador. Los módulos con los que se contara serán los siguientes:

Módulo de Control de Acceso.

En este módulo se llevará a cabo el control del acceso al sistema, definido para varios usuarios y la validación mediante el uso de una contraseña con encriptación.

Módulo de Catálogo de Seguridad.

- Usuarios. Este módulo permitirá realizar altas, bajas, cambios y consultas de los usuarios, que tendrán acceso al sistema y a las funciones que ellos tendrán por las características de su trabajo.

- Perfiles. Este módulo permitirá realizar altas, bajas, cambios y consultas de los perfiles y definir los permisos que tendrá el usuario.

Módulo de Catálogos.

En este módulo se permitirá realizar altas, bajas, cambios y consultas a los catálogos que se mencionan a continuación: parroquias, ceremonias, clientes, sacerdotes.

Módulo de Celebraciones.

En este módulo se permitirá realizar altas, bajas, cambios y consultas a los catálogos que se mencionaran a continuación: celebraciones y cancelaciones.

Módulo de Emisión de recibo de pago.

Este módulo permitirá generar un recibo de pago para cada tipo de celebración.

Módulo de Reportes.

Este módulo permitirá generar los reportes requeridos por el usuario final, así como la emisión de gráficos estadísticos.

El manejo de la información se realizara mediante un conjunto de conexiones abiertas a distintas bases de datos de manera que puedan ser reutilizadas al realizar múltiples consultas o actualizaciones para esto se realizara un agrupamiento de conexiones (connection pool) con la finalidad de controlar que varios usuarios traten de acceder al mismo registro de la base de datos al mismo tiempo, para la comunicación entre la base de datos y la aplicación web se utilizará programación orientada a objetos.

El *pool de conexiones* involucra las siguientes bases de datos:

- Para el bloque de seguridad existirá la conexión con Oracle.
- Para los bloques de aplicación y reportes se tendrá una base de datos en Oracle.

Para llevar a cabo el control del pool de conexiones es necesario configurar el Driver Manager JDBC el se comunican con la capa de Negocio y la capa de Recursos. Utilizando el patrón DAO y DTO.

1.3. Trabajos relacionados

Titulo	Analogía	Diferencias
"Sistema de gestión de información de pacientes para una clínica homeopática" (Trimestre 2008 Primavera [1]).	Aplicación web que administra la información en una base de datos implementada en MySQL utiliza el estándar J2EE de Java, realiza la exportación en formatos PDF y DOC, maneja un módulo de seguridad y un módulo de control de usuarios.	La generación de las vistas están en formato JSP implementadas con AJAX, la aplicación a desarrollar utilizara JSF implementados con ICEfaces; el módulo de seguridad a desarrollar manejará contraseñas encriptadas.
"Sistema de gestión de nómina para PyMES" (Trimestre 2010 Primavera [2]).	Aplicación web maneja un módulo de seguridad y un módulo de control de usuarios, genera exportación de reportes en formato PDF.	Utiliza los frameworks Struts y Hibernate para el mapeo a la base de datos, Jasper Report para el módulo de reportes.

"Sistema de gestión de facturación y cotización" (Trimestre 2011 Invierno [3]).	Aplicación web maneja un módulo de seguridad, un módulo de control de usuarios, módulo de reportes y un módulo de facturación.	Utiliza los frameworks Struts y Hibernate para el mapeo a la base de datos.

2. Desarrollo

2.1. Diseño

2.1.1. Comunicación entre bloques

En el diagrama de bloques que se muestra en la
Figura **1** se observa la distribución de los bloques que conforman el sistema a realizar.

Figura 1. Diagrama de bloques.

La navegación de las funciones se contempla de acuerdo a la estructura del siguiente diagrama mostrado en la Figura 2.

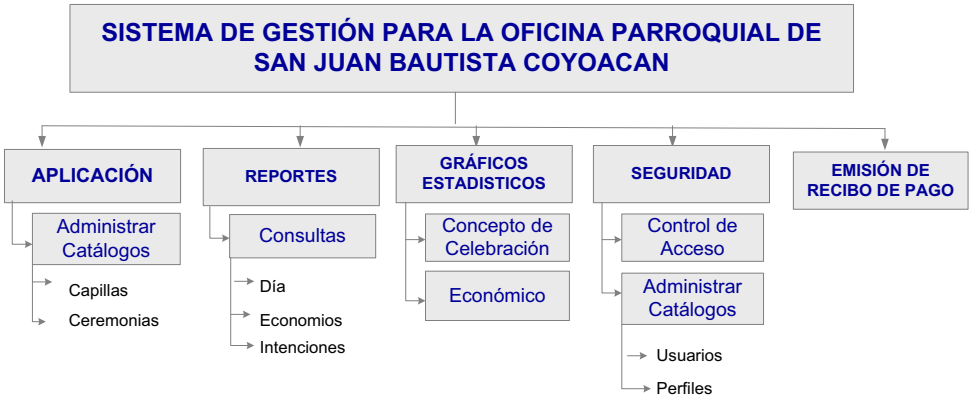


Figura **2**. Mapa de navegación de la administración del sistema a desarrollar.

2.1.2. Arquitectura

El número de las entradas y salidas se considerará abierto puesto que estos parámetros serán almacenados en una base de datos la cual no tiene un límite de entradas o salidas. La Figura **3** muestra la arquitectura tecnológica a utilizar.

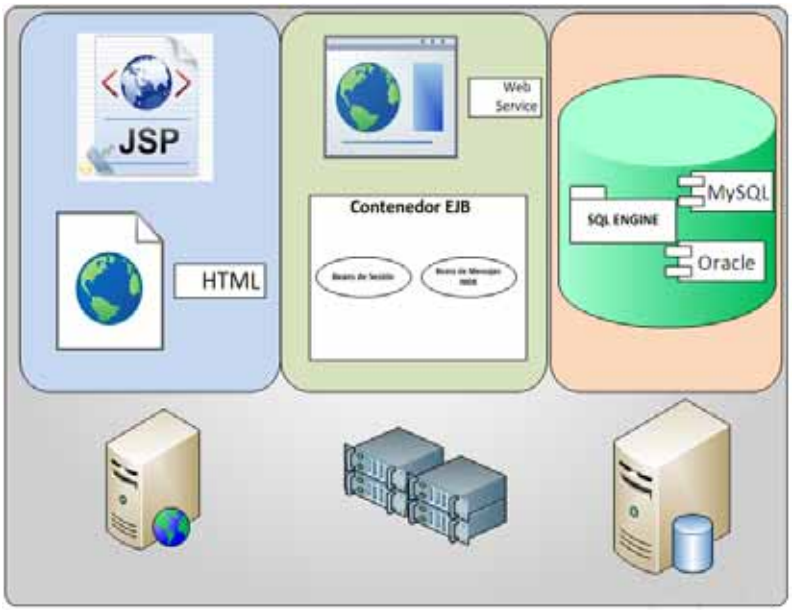


Figura 3. Arquitectura cliente servidor.

El sistema a desarrollar es un sistema web multicapa (cliente, presentación, negocio, integración y recursos). La Figura **4** muestra la interoperabilidad de las capas del sistema web a desarrollar.

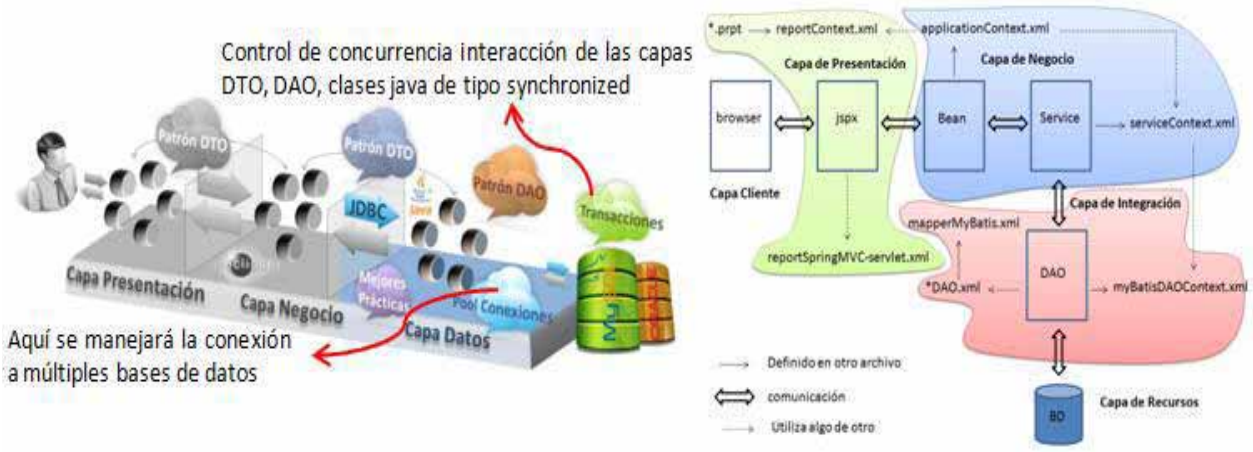


Figura 4. Interoperabilidad entre capas.

2.1.3. Descripción de las Capas

- **Capa de Cliente:** Es la capa donde se ejecuta una aplicación de consola, PC desktop o con navegador web.
- **Capa de Presentación:** Aquí residen los componentes dinámicos de vista y control que generan una salida HTML, XML y los Reportes que se envían al cliente, por ejemplo JSP, Servlets [4].
- **Capa de Negocio:** Aquí residen los procesos de negocio.
- **Capa de Integración:** Aquí residen los componentes que permiten la comunicación al exterior tal como componentes de acceso a base de datos, web services, clientes de web service [5].
- **Capa de Recursos:** Aquí se encuentran todas las fuentes de información externas, tales como Bases de Datos, sistemas de Archivos.

2.2. Seguridad

El sistema debe proteger los datos sensibles y debe protegerse de los ataques de seguridad web más comunes, propuestas para cubrir el requerimiento:

Como medida de seguridad si un usuario introdujo mal la contraseña solo se permitirán tres intentos, al cuarto intento se bloqueará el acceso al sistema por un determinado tiempo.

No se codificará reglas de negocio en JavaScript o HTML para evitar cross-scripting[1] [6].

La información de todos los catálogos y los reportes se podrá exportar a un formato en PDF y XLS, para su posterior visualización o impresión. El recibo de pago se exportará en formato PDF.

[1] Ataque de inyección de código malicioso para su posterior ejecución que puede realizarse a sitios web.

2.3. Entorno de desarrollo

El sistema se desarrollará bajo el Sistema Operativo Windows. Al tratarse de una aplicación web se utilizarán las tecnologías: JSF 2.1 utilizando el framework ICEfaces [7] para generar las interfaces, el estándar J2EE 5 utilizando el framework Spring y Apache Tomcat; con un protocolo de comunicación entre cliente y servidor enfocado en HTTP con el fin de lograr una portabilidad en el contexto de la plataforma tanto el cliente como el servidor podrán ejecutarse en diferentes sistemas operativos, se usará MyBatis [8] como framework para la comunicación con la base de datos, las bases de datos se desarrollarán en Oracle 11g. El entorno de desarrollo será Eclipse. Para generar los reportes se utilizará Pentaho Reporting de Pentaho Open Source Business Intelligence [9]. El análisis del flujo del sistema se detalla en el caso de uso (Apéndice A).

2.3.1. Herramientas

Spring-Tool-Suite-3.0.0.

El proyecto se implemento utilizando java como lenguaje de programación, mediante el uso de Spring-Tool-Suite-3.0.0. [10] como entorno de desarrollo integrado (IDE). El uso de este IDE permite al programador un desarrollo más ágil y una mejor estructuración del proyecto. Esta herramienta permitió una mejor documentación del código fuente de la aplicación así como la realización de las pruebas unitarias.

Mediante el uso de la plataforma de herramientas Web pre instaladas. “Editor de páginas Web” que permite ver los elementos que se están editado en una ventana de vista previa.

Oracle SQL Developer.

Oracle SQL Developer [11] es un entorno de desarrollo integrado gratuito que simplifica el desarrollo y gestión de base de datos Oracle. SQL Developer ofrece un desarrollo completo de extremo a extremo de las aplicaciones PL / SQL, una hoja de cálculo para ejecutar consultas y scripts, una consola de DBA para la gestión de la base de datos, una interfaz de informes, una solución completa de modelado de datos y una plataforma de migración para mover el bases de datos

Con Oracle SQL Developer se exportaron los modelos de entidad-relación por esquemas y se genero el script de inicialización con los datos relacionados a las tablas de seguridad, funciones y perfiles para lograr el acceso al sistema desde la interfaz.

Oracle SQL Developer Data Model.

Oracle SQL Developer Data Modeler [12] es una herramienta de diseño y modelado de datos, proporcionando los medios para desarrollar modelos de datos relacionales y físicos, tipos de datos definidos por el usuario, multidimensionales, lógicos. Oracle SQL Developer Data Modeler proporciona un enfoque impulsado por modelo para el diseño y generación de base de datos, implementados por un conjunto integrado de

modelos - lógico, tipos de datos, dimensional, relaciona, diagramas de flujo de datos y modelos físicos compatibles con bases de datos Oracle, Microsoft SQL Server 2000 y 2005, IBM DB2/390 y DB2 LUW V7 y V8 que representan diferentes aspectos o nivel de detalles y características de proveedor del sistema de información modelado.

Oracle Data Modeler es una aplicación que puede ejecutarse de manera independiente o incorporarse como un módulo en otras herramientas por ejemplo Oracle SQL Developer. Los modelos diseñados se almacenaron en el sistema de ficheros, bajo una estructura de directorios (Apéndice B). Cada modelo puede tener diferentes implementaciones físicas (en diferentes tecnologías).

Microsoft Office Visio.

Las herramientas que componen a Microsoft Office Visio [13] permiten realizar diagramas de oficinas, diagramas de bases de datos, diagramas de flujo de programas, UML, y más, que permiten iniciar al usuario en los lenguajes de programación. Se utilizo Microsoft Office Visio para el diseño de las vistas del proyecto, es muy intuitivo y cuenta con numerosos elementos de ayuda que facilito dicha tarea.

2.4. Pruebas del sistema

La prueba que se muestra a continuación corresponde al modulo de seguridad. Como accesar al sistema desde la interfaz, dar de alta a un usuario y la asignación de su perfil:

1. Para acceder a la aplicación, se deberá abrir un navegador de Internet y capturar en la barra de direcciones la siguiente URL:
2. Ingresar el usuario y contraseña asignados por el administrador, en este caso se ingresan los datos del mismo administrador. Oprimir el botón “Ingresar”. Figura 5.

Figura 5. Inicio de sesión para acceder al sistema SPOFM.

3. Desplegar el menú “Administración del Sistema” y seleccionar el submenú “Usuarios”. Figura 6.

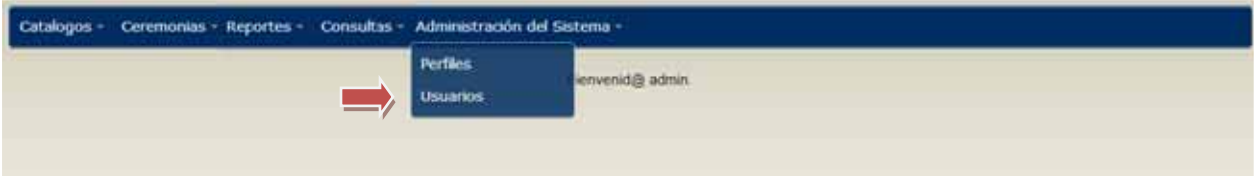


Figura 6. Lista de catálogos.

4. Este catálogo permite la administración de los usuarios que tendrán acceso al sistema y a las funciones que ellos tendrán por las características de su trabajo. Figura 7. Para dar de alta un nuevo usuario solo se debe de oprimir el botón “agregar” 

Figura 7. Datos de los usuarios almacenados en la base de datos, visualizados por medio de un grid.

5. En la parte superior se encuentra una sección donde se ingresan los datos requeridos para realizar el alta de un nuevo usuario en la base de datos .En la parte inferior se mostraran los datos correspondientes a todos los usuarios mediante grids. Figura 8.

Figura 8. Campos solicitados para el registro de un nuevo usuario.

Al ingresar el área de trabajo se desplegara un combo box para elegir una opción del sistema. Figura 9.

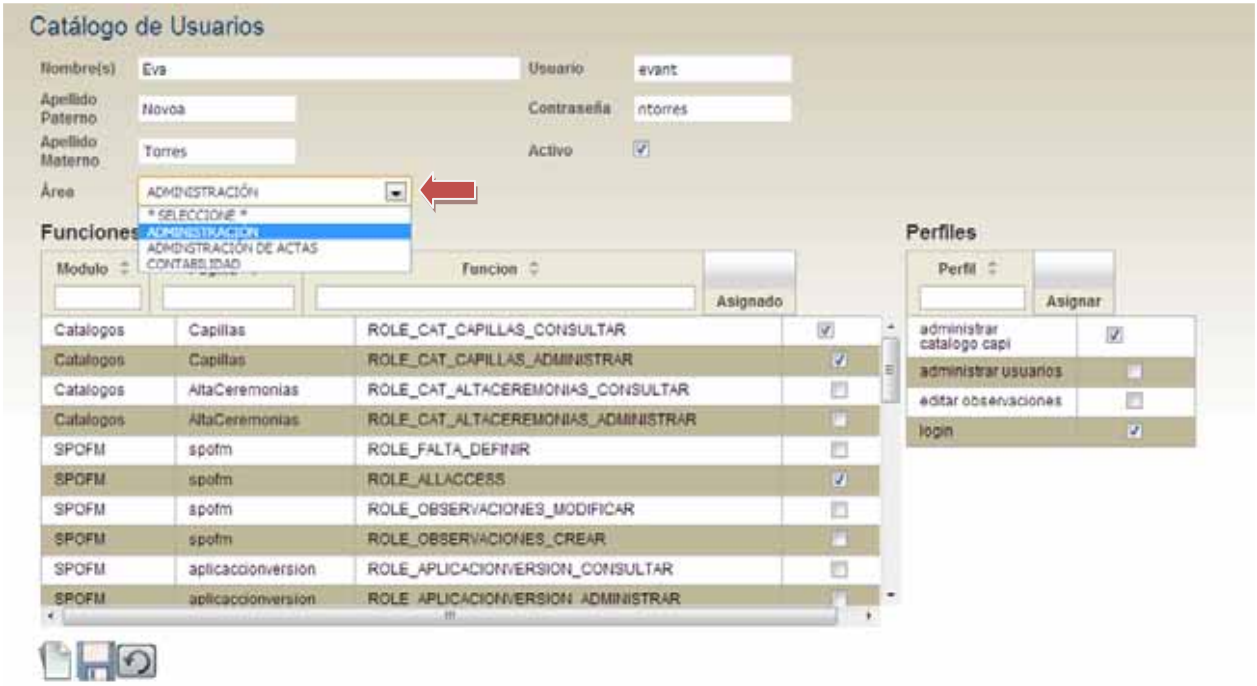


Figura 9. Combo desplegado con las opciones de área de trabajo.

6. Al término de ingresar los datos se debe pulsar el botón “guardar”  y se mostrara un mensaje de confirmación. Figura 10.

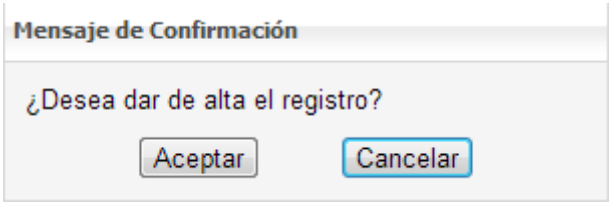


Figura 10. Mensaje de confirmación.

El mensaje de error se muestra cuando los datos capturados por el usuario son incorrectos (duplicidad de datos en alguno de los campos, caracteres no permitidos, campos vacios). Figura 11.

Figura 11. Mensaje de error por dejar campos vacios.

El mensaje de éxito se muestra cuando los datos capturados por el usuario son correctos y validados por el sistema. Figura 12.

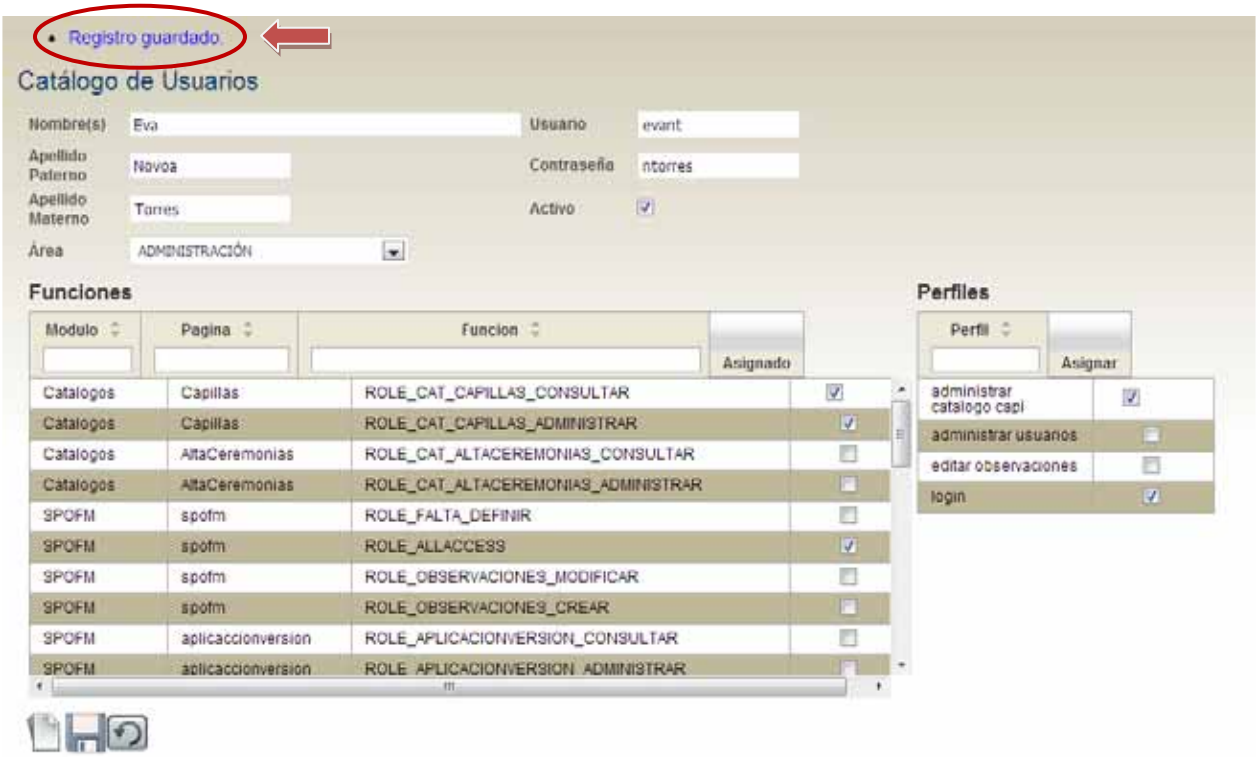


Figura 12. Mensaje de éxito.

7. Al final se observa que los datos del nuevo usuario aparecen en el gid. Figura 13.

Nombre(s)	Apellido Paterno	Apellido Materno	Usuario	Área	Activo
Patricia	Roldan	Ramirez	Paty	ADMINISTRACIÓN	Activo
Eva	Novoa	Torres	evant	ADMINISTRACIÓN	Activo
agustin	anguiano	navarro	agus	CONTABILIDAD	Activo
Diana	Sanchez	hgf	diana	CONTABILIDAD	Inactivo
Salvador	Oliva	Garcia	chava	CONTABILIDAD	Activo
Gustavo	Fernandez	Guzman	gus	CONTABILIDAD	Activo
Jorge Antonio	Mendoza	Matinez	jameofm	CONTABILIDAD	Activo
Maricela	Brsvo	Contrearas	Bravo	CONTABILIDAD	Activo
Emmanuel	Oliva	García	admin	ADMINSTRACIÓN DE ACTAS	Activo
Juan Carlos	Garcia	Ricaldi	gari	ADMINSTRACIÓN DE ACTAS	Inactivo

Figura 13. Datos actualizados de los usuarios.

8. Probando que el nuevo usuario tenga acceso al sistema ingresando el usuario y contraseña asignados por el administrador. Figura 14.

Figura 14. Ingresando al sistema con los datos del nuevo usuario.

Se presenta la pantalla principal donde se muestran las opciones a las que tiene permiso de acceder el usuario. En el área de trabajo se muestra un mensaje de bienvenida con el alias del usuario y en la parte inferior su(s) nombre(s) y apellidos. Figura 15.

Figura 15. Pantalla de bienvenida y área de trabajo.

3. Conclusiones

Si bien pueden existir múltiples definiciones del término “servicios web”, una definición de sencillo entendimiento puede ser que los Servicios Web son un conjunto de aplicaciones o de tecnologías que interoperan, (intercambian información y utilizan la información intercambiada), con el objetivo de ofrecer servicios mediante interoperabilidad basada en la Web.

La intención del presente trabajo es mejorar y eficientar la administración de datos en una parroquia, así como la autentificación de los mismos, la optimización de recursos y la implementación como aplicación web.

Uno de los aspectos tratados en el desarrollo del trabajo fue la documentación, es un aspecto sumamente importante, tanto en el desarrollo de la aplicación como en el mantenimiento de la misma. La documentación de un programa empieza a la par que la construcción del mismo y finaliza justo antes de la entrega del programa o aplicación.

Algunas de las dificultades con las que nos encontramos fue: el módulo de seguridad ya que se pensaba realizar con permisos a nivel comando. Esto es, cada usuario almacenado en la base de datos debe contar con un permiso por cada comando (acción de un botón) utilizado en las páginas a las cuales tendrá acceso en la aplicación, esto implica la realización de una bitácora por cada tabla estratégica en la bese de datos con ayuda de campos de auditoría como: FECHA_CREACION este campo indica la fecha de creación de un registro, FECHA_ULTIMA_MODIFICACION campo donde indicara la fecha de la última modificación en un registro, USUARIO_MODIFICO nombre del usuario que realizo la última modificación, FECHA_BAJA campo donde se guardara la fecha en la que el registro se dé de baja utilizando un borrado lógico así físicamente permanecerá en la base de datos pero no ya no se mostrara en la aplicación.

Por la complejidad y la falta de tiempo el modulo de seguridad se diseño de la siguiente manera: se crearon dos roles: Administrador y Consulta. El rol de administrador engloba crear, modificar, borrar y consultar los registros de la base de datos. El rol de consultar solo muestra la información sin permiso de realizar alguna modificación. Estos roles son asignados por pagina y a su vez asignados a cada usuario del sistema.

De acuerdo a los calendarios planteados en la propuesta del presente trabajo (Apéndice C), los módulos desarrollados con las características señaladas en la misma se llevaron a cabo en tiempo y forma conforme al plan.

Todo esto fue desarrollado gracias a los conocimientos adquiridos en la formación académica de la carrera de Ingeniería en computación en nuestra estancia en la Universidad Autónoma Metropolitana.

4. Referencias

[1] S. Manzanares Soriano, "Sistema de gestión de información de pacientes para una clínica homeopática", Proyecto terminal concluido, División de CBI, Universidad Autónoma Metropolitana Azcapotzalco, México, 2008.

[2] H. Iturbide García, "Sistema de gestión de nómina para PyMES", Proyecto terminal concluido, División de CBI, Universidad Autónoma Metropolitana Azcapotzalco, México, 2010.

[3] E. E. González Herrera, "Sistema de gestión de facturación y cotización", Proyecto terminal concluido, División de CBI, Universidad Autónoma Metropolitana Azcapotzalco, México, 2011.

[4] P of EAA: Data Transfer Object [En línea]
Disponible: http://martinfowler.com/eaaCatalog/dataTransferObject.html
Revisado: martes, 23 de octubre del 2012.

[5] Que es Data Access Object | Java México [En línea]
Disponible: http://www.javamexico.org/blogs/richardmx/que_es_data_access_object
Revisado: martes, 23 de octubre del 2012.

[6] Introducción al Cross-Site-Scripting [En línea]
Disponible: http://www.seguridad.unam.mx/documento/?id=35
Revisado: martes, 23 de octubre del 2012.

[7] ICEfaces Showcase [En línea]
Disponible: http://icefaces-showcase.icesoft.org/showcase.jsf?grp=aceMenu&exp=dataTableBean
Revisado: martes, 23 de octubre del 2012.

[8] MyBatis 3 Introducción [En línea]
Disponible: http://www.mybatis.org/core/es/index.html
Revisado: martes, 23 de octubre del 2012.

[9] Introducción a Pentaho Business Intelligence [En línea]
Disponible: http://pentaho.almacen-datos.com/
Revisado: martes, 23 de octubre del 2012.

[10] Spring-Tool-Suite-3.0.0. [En línea]
Disponible: http://blog.springsource.org/2012/08/13/springsource-tool-suites-3-0-0-released-reorganized-open-sourced-and-at-github/
Revisado: martes, 18 de junio del 2013.

[11] Oracle SQL Developer [En línea]
Disponible: http://www.oracle.com/technetwork/developer-tools/sql-developer/overview/index.html
Revisado: martes, 18 de junio del 2013.

[12] Oracle SQL Developer Data Model [En línea]
Disponible: http://www.oracle.com/technetwork/developer-tools/datamodeler/overview/index.html
Revisado: martes, 18 de junio del 2013.

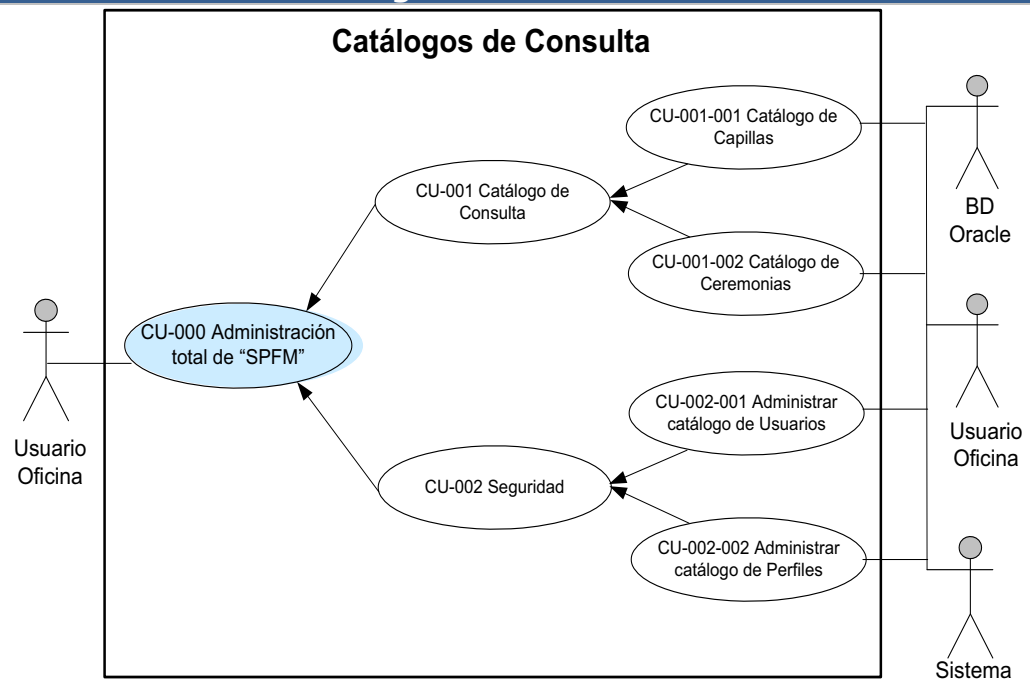
[13] Microsoft Visio [En línea]
Disponible: http://microsoft-office-visio-pro.softonic.com/
Revisado: martes, 18 de junio del 2013.

5. Apéndices

A

Caso de Uso	CU-000 Administración total del "SPFM"	**Proceso**	Administrador total del SPFM
Nombre del documento	CU-000 Administración total del "SPFM" V1.0(PRR)(130611)	**Tarea**	Administra los catálogos de consulta y la seguridad del sistema.
Fecha de elaboración	2013-06-11	**Elaboró**	Patricia Roldán Ramírez

Diagrama Caso de Uso



Descripción de la funcionalidad

La administración del SPFM implica la administración de las solicitudes y el registro de las celebraciones eucarísticas (misas ó ceremonias) y la administración de la seguridad.

Precondiciones

- Disponibilidad de la base de datos de Oracle.
- Disponibilidad del sistema SPFM.

Post Condiciones

- Alta de capillas, usuarios, solicitantes, cerebraciones.

- Baja de capillas, usuarios, solicitantes, cerebraciones.
- Actualización de los datos de capillas, usuarios, solicitantes, cerebraciones.
- Modificación de tablas de capillas, usuarios, solicitantes, cerebraciones.
- Consultas capillas, usuarios, solicitantes, cerebraciones.
- Impresión de la tabla capillas.

Flujo Normal		
Paso	**Requerimiento**	**Descripción**
○	Administrar Seguridad	El sistema administrara los permisos de usuarios, con el fin de controlar el acceso de los mismos en las consultas, modificaciones y conexiones de la base de datos. Usuarios: Se podrán crear, editar, eliminar y consultar usuarios; así como, asignar permisos a las funciones, consultas, modificaciones y conexiones de los usuarios del sistema. Perfiles: Se podrán dar de alta, editar, eliminar y consultar los perfiles; además de asociar los permisos dentro del módulo de administración a los perfiles, que, posteriormente, serán asignados a los usuarios del sistema.
○	Administrar Catálogos	En el rubro de catálogos permitirá definir los módulos que podrán ser visualizados de acuerdo a las características del usuario. En la administración se podrán crear, editar, eliminar y consultar la información correspondiente a cada modulo. Capillas: En la administración de capillas se podrán agregar, modificar, consultar, eliminar e imprimir los registros de la tabla capillas. Ceremonias: Se podrán dar de alta, editar, eliminar y consultar las ceremonias así como los solicitantes asociados a ellas. Solicitantes: En la administración de solicitantes se podrán agregar, modificar, consultar, eliminar, imprimir y exportar los registros de la tabla solicitantes así como las ceremonias asociados al solicitante.

Notas para Implementación	
Id	**Notas**
•	Botones e iconos Un botón es un dispositivo utilizado para activar alguna función. Los botones pueden ser de diversas formas y tamaños. Un ícono es un pequeño gráfico en pantalla que identifica y representa a algún objeto, usualmente con algún simbolismo gráfico para establecer una asociación y servirá para realizar una función específica en el sistema. A continuación se presentan los botones e Íconos utilizados en el sistema:

Botón/Ícono	Descripción de acción
	Botón que permite el acceso al sistema después de capturar nombre de usuario y contraseña.
	**Icono Agregar**: Permite Agregar un nuevo registro. Al dar clic en este ícono, el sistema muestra los campos de captura.
	**Icono Guardar**: Permite guardar los cambios realizados al registro. Al dar clic en este ícono, se presenta una leyenda “Guardando cambios”
	**Icono Guardar Deshabilitado:** Ícono que por default se muestra deshabilitado, se habilita cuando el usuario va a agregar o modificar registros.
	**Icono Cancelar**: Permite cancelar la acción elegida por el usuario, si este decidió cancelar cuando estaba agregando un nuevo registro, el sistema limpia los campos capturados y si la cancelación es cuando el usuario modifica algún registro, no se guarda el registro y se deja en la versión anterior.
	**Icono Cancelar Deshabilitado:** Ícono que por default se muestra deshabilitado, se habilita cuando el usuario va agregar o modificar registros.
	**Icono para exportación en formato Excel**: Al dar clic en este ícono, el sistema genera un archivo con información en formato Excel.
	**Icono para exportación en formato PDF**: Al dar clic en este ícono, el sistema genera un archivo con información en formato PDF.

Bitácora de Cambios				
Versión	**Fecha**	**Modificación**	**Solicitado por:**	**Modificado por:**

Caso de Uso	CU-001-001 Administrar Catálogo de Capillas	Proceso	Catálogos de Capillas
Nombre del documento	CU-001-001 Administrar Catálogo de Capillas V1.1(PRR)(130507)	Tarea	Altas, bajas, modificaciones e impresión de los catálogos de Capillas
Fecha de elaboración	2013-05-07	Elaboró	Patricia Roldán Ramírez

Diagrama Caso de Uso

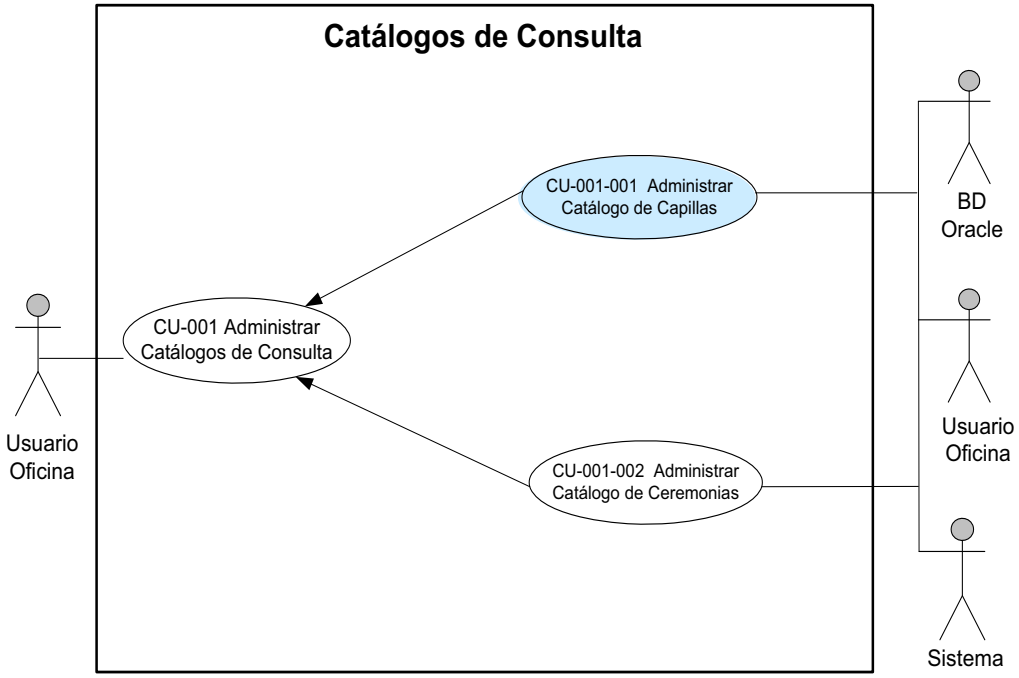
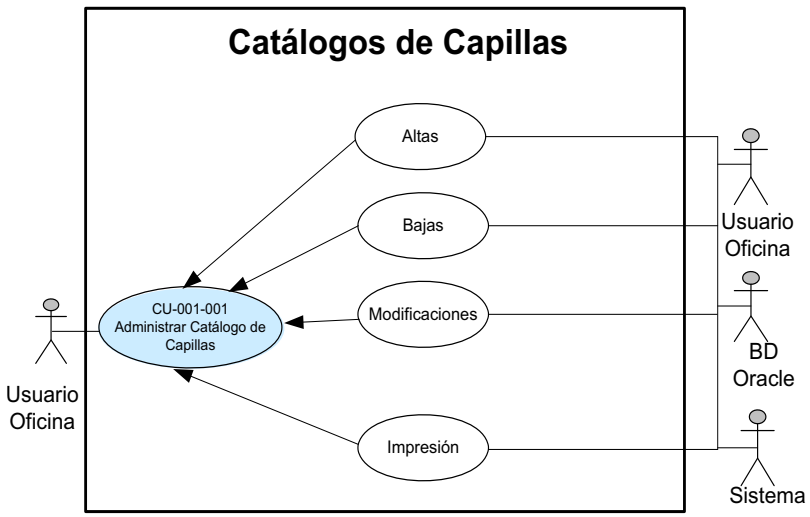


Diagrama de función



Descripción de la funcionalidad

Dentro del rubro de "catálogos de consulta", pertenece el catálogo de capillas en el cual se puede: agregar, eliminar, modificar e imprimir los registros de la tabla capillas.

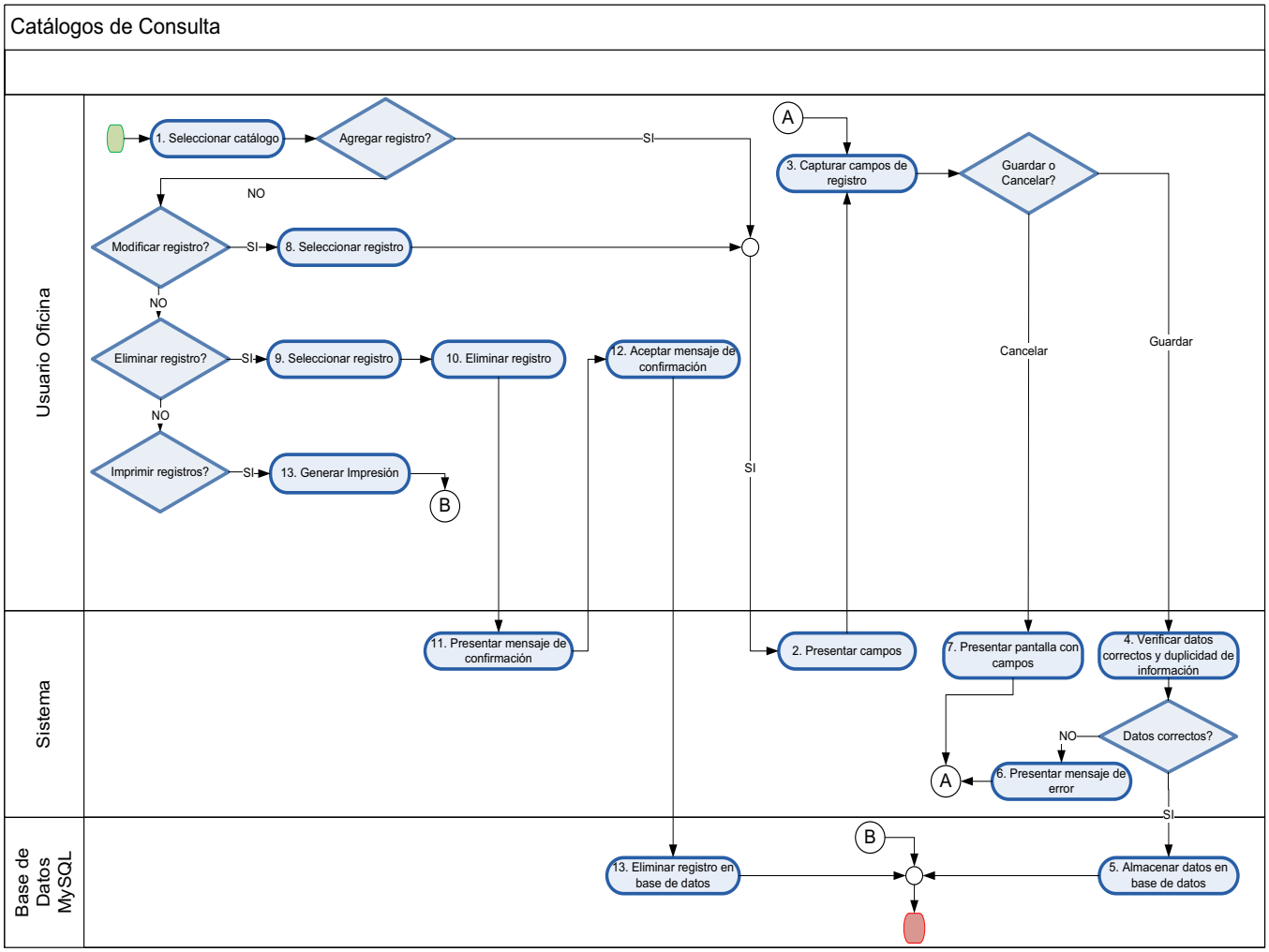
Precondiciones

- Disponibilidad de la base de datos de Oracle.
- Deben existir datos en la tabla entidad de la base de datos de Oracle.
- Deben existir datos en la tabla municipio de la base de datos de Oracle.
- Cuenta de usuario activa y vigente con los accesos necesarios para agregar, modificar y eliminar registros, en la tabla capillas de la base de datos de Oracle.

Post Condiciones

- Alta de una o varias capillas.
- Baja de una o varias capillas.
- Actualización de los datos de una o varias capillas.
- Modificación de la tabla capillas.
- Consulta de las capillas almacenadas y registradas en el sistema.
- Impresión de la tabla capillas.

Diagrama de procesos



Escenario Principal

Paso	Acción
o	**"Agregar Capilla" [Administrador]** Catálogos > Capillas > Agregar Capilla > Capilla Nuevo Registro En la pantalla Nuevo Registro > Agregar Capilla, se presentará un formulario donde el usuario deberá de capturar la información referente a la capilla que se requiere dar de alta en el sistema (Ver Figura 1). El usuario podrá agregar los datos de una capilla pulsando el icono  ** (Ver Notas para Implementación) Sistema: Presentar Área de Trabajo

Nota: Se deberá mostrar en pantalla el área de trabajo en un grid con la información de todas las capillas almacenadas en la tabla CAPILLAS de la base de datos SPOFM (Ver Figura 2).

Usuario:

Pulsar Icono:  [Agregar] **[4]

Sistema:

Validar: Tipo Usuario

Obtener Información: Usuario Firmado en Sesión

[USF].Id_Usuario

[USF].Id_Funcion

[FN].Id_Pagina

Criterios: Consulta

[FN].Id_Funcion = Role_AllAccess

[PF].Id_Funcion = [PRL].Id_Funcion

[FN].Id_Funcion = [USF].Id_Funcion

[USF].Id_Usuario = [US].Id_Usuario (Usuario Firmado en Sesión) ** [6]

[PG].Id_Pagina = [FN].Id_Pagina

[USF].Fecha_Baja = Null

[FN].Fecha_Baja = Null

[PF].Fecha_Baja = Null

[PRL].Fecha_Baja = Null

[PG].Fecha_Baja = Null

Fin Obtener: Usuario Firmado en Sesión

Si es: Administrador Entonces:

Presentar Página: Agregar Capilla.jsp

En otro caso:

Presentar Mensaje: No cuenta con los permisos para agregar

Fin Validar: Tipo Usuario

Nota: Sólo el usuario administrador podrá agregar nuevos registros de capillas.

Ver caso de uso Catálogo de usuarios y Perfiles paso 1 “”

Presentar Información: Nueva Capilla.

Campo: * Nombre Parroquia:
Campo: * Nombre Capilla:
Campo: * Entidad:
Campo: * Municipio:
Campo: * Calle:
Campo: * Número:

Campo: * Colonia:
Campo: * Código Postal:
Campo: * Lada:
Campo: * Teléfono uno:
Campo: Teléfono dos:
Campo: Teléfono tres:
Campo: Fax:
Campo: Correo Electrónico:
Campo: * Activo:
Fin Presentar: Nueva Capilla

Usuario:

Capturar Información: Nueva Capilla

Campo: * Nombre Parroquia:
Campo: * Nombre Capilla:
Campo: * Entidad: ** [1]
Campo: * Municipio: ** [2]
Campo: * Calle:
Campo: * Número:
Campo: * Colonia:
Campo: * Código Postal:
Campo: * Teléfono uno: ** [3]
Campo: Teléfono dos: ** [3]
Campo: Teléfono tres: ** [3]
Campo: Fax:
Campo: Correo Electrónico:
Campo: * Activo: ** [5]
Fin Capturar: Nueva Capilla

Pulsar Icono:  [Guardar]

Sistema:

Validar: Información Requerida

Si es: Omitida Entonces:

Presentar Mensaje: Complete los datos requeridos.

(Ver Tabla de validaciones)

Fin Validar: Información Requerida

Agregar Información: Capilla Nueva
[CAP].Id_Parroquia = Campo: Nombre Parroquia:
[CAP].Id_Capilla = Autoincrementar
[CAP].Nombre_Capilla = Campo: Nombre:
[CAP].Id_Entidad = Campo: Entidad:
[CAP].Id_Municipio = Campo: Municipio:
[CAP].Calle = Campo: Calle:
[CAP].Numero = Campo: Número:
[CAP].Colonia = Campo: Colonia:
[CAP].Codigo_Postal = Campo: Código Postal:
[CAP].Lada = Campo: Lada:
[CAP].Telefono_uno = Campo: Teléfono uno:
[CAP].Telefono_dos = Campo: Teléfono dos:
[CAP].Telefono_tres = Campo: Teléfono tres:
[CAP].Fax = Campo: Fax:
[CAP].Correo_Electronico = Campo: Correo Electrónico:
[CAP].Activo = Campo: Activo:

o	**"Buscar Capilla"** Catálogos > Capillas > Pantalla Capillas Para buscar alguna capilla, el usuario debe ingresar a Catálogos > Capillas donde se mostrará un grid con la información de todas las capillas almacenadas en el sistema, el usuario deberá capturar la palabra completa o el primer carácter de la palabra que se desea buscar. ** (Ver Notas para Implementación) Usuario: Capturar Criterios Búsqueda: Parroquias Campo: Nombre: Campo: Entidad: Campo: Municipio: Campo: Calle: Campo: Número: Campo: Colonia: Campo: Código Postal: Campo: Teléfono uno: Campo: Teléfono dos: Campo: Teléfono tres: Campo: Fax: Campo: Correo Electrónico:**¡Error! No se encuentra el origen de la referencia.** Campo: Activo: Fin Capturar Criterios: Parroquias Nota: Las búsquedas se hacen mediante la combinación de caracteres que se desea filtrar correspondiente al campo o a los campos elegidos por el usuario. Nota: En el encabezado de cada columna se deberán mostrar las cajas de texto para capturar la palabra a buscar. Nota: Sólo se presentará la información de las parroquias que se encuentren almacenadas en el sistema. Sistema: Asignar variables ID CAPILLA = [CAP].Id_Capilla NOMBRE CAPILLA = [CAP].Nombre_Capilla ID ENTIDAD = [CAP].Id_Entidad ID MUNICIPIO = [CAP].Id_Municipio CALLE = [CAP].Calle NÚMERO =[CAP].Numero COLONIA = [CAP].Colonia CÓDIGO POSATAL = [CAP].Codigo_Postal LADA = [CAP].Lada TELEFONO UNO = [CAP].Telefono_uno

```
TELEFONO DOS = [CAP].Telefono_dos
TELEFONO TRES = [CAP].Telefono_tres
FAX = [CAP].Fax
CORREO ELECTRONICO = [CAP].Correo_Electronico
ACTIVO = [CAP].Activo

Obtener: Datos Capilla
  [CAP].Id_Capilla
  [CAP].Nombre_Capilla
  [CAP].Id_Entidad
  [CAP].Id_Municipio
  [CAP].Calle
  [CAP].Numero
  [CAP].Colonia
  [CAP].Codigo_Postal
  [CAP].Lada
  [CAP].Telefono_uno
  [CAP].Telefono_dos
  [CAP].Telefono_tres
  [CAP].Fax
  [CAP].Correo_Electronico
  [CAP].Activo

Criterios:
  ID CAPILLA = Campo: Capilla
  NOMBRE CAPILLA = Campo: Nombre:
  ID ENTIDAD = Campo: Entidad:
  ID MUNICIPIO = Campo: Municipio:
  CALLE = Campo: Calle:
  NÚMERO = Campo: Número:
  COLONIA = Campo: Colonia:
  CÓDIGO POSTAL = Campo: Código Postal:
  LADA = Campo: Lada:
  TELEFONO UNO = Campo: Teléfono uno:
  TELEFONO DOS = Campo: Telefono dos:
  TELEFONO TRES = Campo: Teléfono tres:
  FAX = Campo: Fax:
  CORREO ELECTRONICO = Campo: Correo Electrónico:

  [EN].Id_Entidad = [MU].Id_Entidad
```

	[EN].Fecha_Baja = Null [MU].Fecha_Baja = Null Fin Obtener: Datos Capilla Presentar Resultado Búsqueda: Capilla Campo: ID CAPILLA = [CAP].Id_Capilla Campo: NOMBRE: = [CAP].Nombre_Capilla Campo: ENTIDAD = [CAP].Entidad Campo: MUNICIPIO = [CAP].Municipio Campo: CALLE = [CAP].Calle Campo: NÚMERO =[CAP].Numero Campo: COLONIA = [CAP].Colonia Campo: CÓDIGO POSATAL = [CAP].Codigo_Postal Campo: LADA = [CAP].Lada Campo: TELEFONO UNO = [CAP].Telefono_uno Campo: TELEFONO DOS = [CAP].Telefono_dos Campo: TELEFONO TRES = [CAP].Telefono_tres Campo: FAX = [CAP].Fax Campo: CORREO ELECTRONICO = [CAP].Correo_Electronico Campo: ACTIVO = [CAP].Activo Fin Presentar: Capilla
3	**"Modificar Datos Capilla"** Catálogos > Capillas > Pantalla Capillas Para modificar la información de alguna capilla, el usuario debe ingresar a Catálogos > Capillas y buscar la capilla que se desea editar, el sistema presentará un grid con la información de todas las capillas almacenadas en el sistema, el usuario deberá capturar la palabra completa o el primer carácter de la palabra que se desea buscar, pulsar sobre la fila seleccionada de acuerdo al resultado de la búsqueda. ****** (Ver Notas para Implementación) Usuario: Capturar Criterios Búsqueda: Capillas Campo: Nombre: Campo: Entidad: Campo: Municipio: Campo: Calle: Campo: Número: Campo: Colonia: Campo: Código Postal:

Campo: Teléfono uno:

Campo: Teléfono dos:

Campo: Teléfono tres:

Campo: Fax:

Campo: Correo Electrónico:

Campo: Activo:

Fin Capturar Criterios: Capillas

Nota: Se deberá mostrar en la parte superior de la pantalla la información de la capilla elegida por el usuario.

Sistema:

Asignar variables

ID CAPILLA = [CAP].Id_Capilla

NOMBRE CAPILLA = [CAP].Nombre_Capilla

ID ENTIDAD = [CAP].Id_Entidad

ID MUNICIPIO = [CAP].Id_Municipio

CALLE = [CAP].Calle

NÚMERO =[CAP].Numero

COLONIA = [CAP].Colonia

CÓDIGO POSATAL = [CAP].Codigo_Postal

LADA = [CAP].Lada

TELEFONO UNO = [CAP].Telefono_uno

TELEFONO DOS = [CAP].Telefono_dos

TELEFONO TRES = [CAP].Telefono_tres

FAX = [CAP].Fax

CORREO ELECTRONICO = [CAP].Correo_Electronico

ACTIVO = [CAP].Activo

Obtener: Datos Capilla

[CAP].Id_Capilla

[CAP].Nombre_Capilla

[CAP].Id_Entidad

[CAP].Id_Municipio

[CAP].Calle

[CAP].Numero

[CAP].Colonia

[CAP].Codigo_Postal

[CAP].Lada

[CAP].Telefono_uno

```
    [CAP].Telefono_dos
    [CAP].Telefono_tres
    [CAP].Fax
    [CAP].Correo_Electronico
    [CAP].Activo

    Criterios:
    ID CAPILLA = Campo: Capilla
    NOMBRE CAPILLA = Campo: Nombre:
    ID ENTIDAD = Campo: Entidad:
    ID MUNICIPIO = Campo: Municipio:
    CALLE = Campo: Calle:
    NÚMERO = Campo: Número:
    COLONIA = Campo: Colonia:
    CÓDIGO POSTAL = Campo: Código Postal:
    LADA = Campo: Lada:
    TELEFONO UNO = Campo: Teléfono uno:
    TELEFONO DOS = Campo: Teléfono dos:
    TELEFONO TRES = Campo: Teléfono tres:
    FAX = Campo: Fax:
    CORREO ELECTRONICO = Campo: Correo Electrónico:

    [EN].Id_Entidad = [MU].Id_Entidad
    [EN].Fecha_Baja = Null
    [MU].Fecha_Baja = Null
  Fin Obtener: Datos Capilla


Usuario:
  Capturar la información correspondiente a los siguientes campos: Teléfono uno, Teléfono
  dos, Teléfono tres, Fax, Correo Electrónico, Activo.


  Capturar Criterios Búsqueda: Capillas
    Campo: * Teléfono uno: ** [3]
    Campo: * Teléfono dos: ** [3]
    Campo: * Teléfono tres: ** [3]
    Campo: Fax:
    Campo: Correo Electrónico:
    Campo: * Activo: ** [5]
  Fin Capturar Criterios: Capillas
```

	Pulsar Icono:  [Guardar] Sistema: Validar: Información Requerida Si es: Omitida Entonces: Presentar Mensaje: Complete los datos requeridos. ** (Ver Tabla de validaciones) Fin Validar: Información Requerida Actualizar Información: Capilla [CAP].Id_Capilla = ID CAPILLA [Oculto] [CAP].Lada = Campo: Lada: [CAP].Telefono_uno = Campo: Teléfono uno: [CAP].Telefono_dos = Campo: Teléfono dos: [CAP].Telefono_tres = Campo: Teléfono tres: [CAP].Fax = Campo: Fax: [CAP].Correo_Electronico = Campo: Correo Electrónico:

Notas para Implementación	
Id	**Notas**
•	El combo Entidad se debe cargar con la información de la tabla de [EN] ENTIDAD ocupando el siguiente criterio de consulta: Campo: [EN].Nombre_Entidad [Todos los registros existentes en la tabla] La opción por default será: -- Seleccione Opción -- Nota: Las opciones de Entidad se presentarán ordenadas alfabéticamente.
•	Al seleccionar una opción en el combo entidad, el combo Municipio se debe cargar con la información de la tabla [MU] MUNICIPIO ocupando el siguiente criterio de consulta: Obtener Información: Municipio [MU].Id_Municipio [MU].Nombre_Municipio Criterios: [EN].Id_Entidad = [MU].Id_Entidad [EN].Fecha_Baja = Null [MU].Fecha_Baja = Null Fin Obtener: Municipio

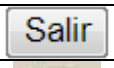
	Cargar Lista: Municipio Campo: Municipio = [MU].Nombre_Municipio Fin Cargar: Municipio Nota: La opción por default será: -- Seleccione Opción -- Nota: Las opciones de Municipio se presentarán ordenadas alfabéticamente.
•	Al seleccionar una opción en el combo Municipio se debe cargar la información de la lada correspondiente al municipio seleccionado: Obtener Información: Lada [MU].Lada Criterios: [Id].Id_Municipio = Campo: **¡Error! No se encuentra el origen de la referencia.** [EN].Id_Entidad = [MU].Id_Entidad [EN].Fecha_Baja = Null [MU].Fecha_Baja = Null Fin Obtener: Lada Cargar: Lada Campo: Lada = [MU].Lada Fin Cargar: Lada Nota: En el campo teléfono se debe mostrar la lada obtenida de acuerdo al municipio seleccionado, permitiendo poder modificar el dato si este fuera incorrecto.

Botones e iconos

Un botón es un dispositivo utilizado para activar alguna función. Los botones pueden ser de diversas formas y tamaños.

Un ícono es un pequeño gráfico en pantalla que identifica y representa a algún objeto, usualmente con algún simbolismo gráfico para establecer una asociación y servirá para realizar una función específica en el sistema.

A continuación se presentan los botones e Íconos utilizados en el sistema:

Botón/Ícono	Descripción de acción
	Botón que permite el acceso al sistema después de capturar nombre de usuario y contraseña.
	Botón que permite salir del sistema.
	**Icono Agregar**: Permite Agregar un nuevo registro. Al dar clic en este ícono, el sistema muestra los campos de captura.
	**Icono Guardar**: Permite guardar los cambios realizados al registro. Al dar clic en este ícono, se presenta una leyenda “Guardando cambios”
	**Icono Guardar Deshabilitado:** Ícono que por default se muestra deshabilitado, se habilita cuando el usuario va a agregar o modificar registros.
	**Icono Cancelar**: Permite cancelar la acción elegida por el usuario, si este decidió cancelar cuando estaba agregando un nuevo registro, el sistema limpia los campos capturados y si la cancelación es cuando el usuario modifica algún registro, no se guarda el registro y se deja en la versión anterior.
	**Icono Cancelar Deshabilitado:** Ícono que por default se muestra deshabilitado, se habilita cuando el usuario va agregar o modificar registros.
	**Icono para exportación en formato Excel**: Al dar clic en este ícono, el sistema genera un archivo con información en formato Excel.
	**Icono para exportación en formato PDF**: Al dar clic en este ícono, el sistema genera un archivo con información en formato PDF.

- Por default se mostrara activo.

Figura 1 – "Alta de una Capilla"

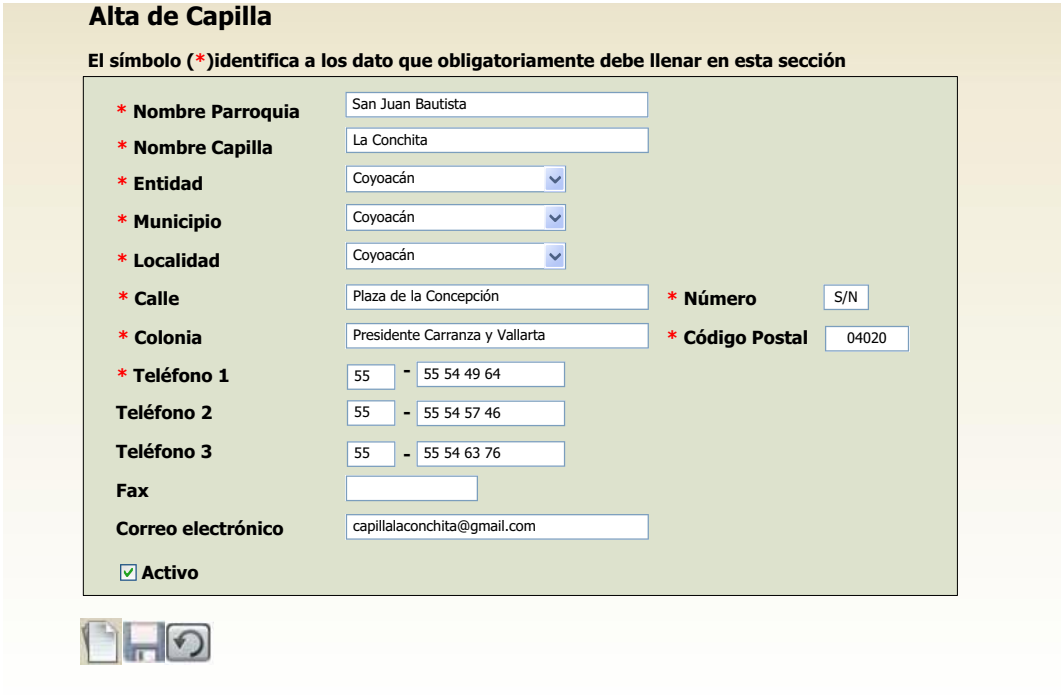


Figura 2 – "Consulta de datos de Capillas"

ID	PARROQUIA	CAPILLA	ENTIDAD	MUNICIPIO	LOCALIDAD	COLONIA	ACTIVO	CALLE	Número	C.P.	TELÉFONO 1	TELÉFONO 2	TELÉFONO 3	FAX	CORREO
1	Santiago Tlatelolco	Casa de la Fraternidad	Tlatelolco	Tlatelolco	Tlatelolco	Tlatelolco	✓	Cerrada Allende	39-B	06900	55 55 49 19 22	55 55 49 19 22		55 55 49 19 22	casafraternidad@gmail.com
2	San Juan Bautista	San José	Hultzilac	Hultzilac	Hultzilac	Hultzilac	✓	Tres Marías	S/N	62510	55 55 54 63 10	55 55 54 63 10	55 55 54 63 10	55 55 54 63 10	capillasanjoseq@gmail.com
3	San Juan Bautista	La Conchita	Coyoacan	Coyoacan	Coyoacan	Coyoacan	✓	Plaza de la Concepción	S/N	04020	55 55 54 49 64			55 55 54 49 64	capillaconchita@hotmail.com
4	San Juan Bautista	Santa Catarina	Coyoacan	Coyoacan	Coyoacan	Coyoacan	✓	Plaza de la Catarina	S/N	04010	55 55 54 29 58			55 55 54 29 58	capillacatarina@yahoo.com.mx
5	San Juan Bautista	San Antonio Panzacola	Coyoacan	Coyoacan	Coyoacan	Coyoacan	✓	Av. Universidad y Francisco Sosa	S/N	04350					capillasantonio@hotmail.com

Bitácora de Cambios

Versión	Fecha	Modificación	Solicitado por:	Modificado por:

Firma de aceptación

Nombre	Firma
DR. Maricela Bravo Contreras	

Caso de Uso	CU-001-002 Administrar Catálogo de Ceremonias	Proceso	Catálogos de Ceremonias
Nombre del documento	CU-001-002 Administrar Catálogo de Ceremonias V1.1(EOG)(130507)	Tarea	Altas, bajas, modificaciones, impresión y exportación de los catálogos de Ceremonias
Fecha de elaboración	2013-05-07	Elaboró	Emmanuel Oliva García

Diagrama Caso de Uso

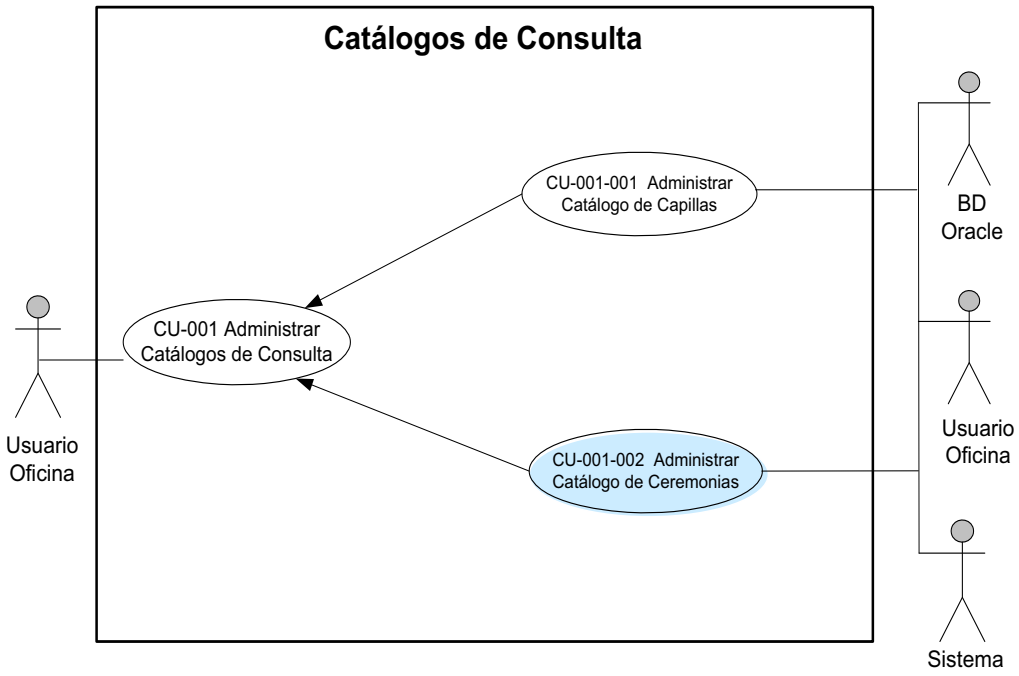
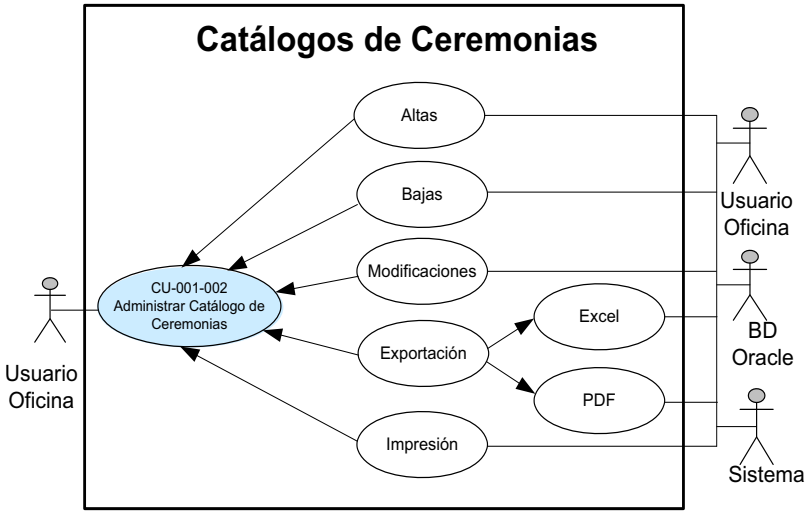


Diagrama de función



Descripción de la funcionalidad

Dentro del rubro de "catálogos de consulta", pertenece el catálogo de ceremonias en el cual se puede: agregar, eliminar, modificar, imprimir y exportar los registros de la tabla ceremonias.

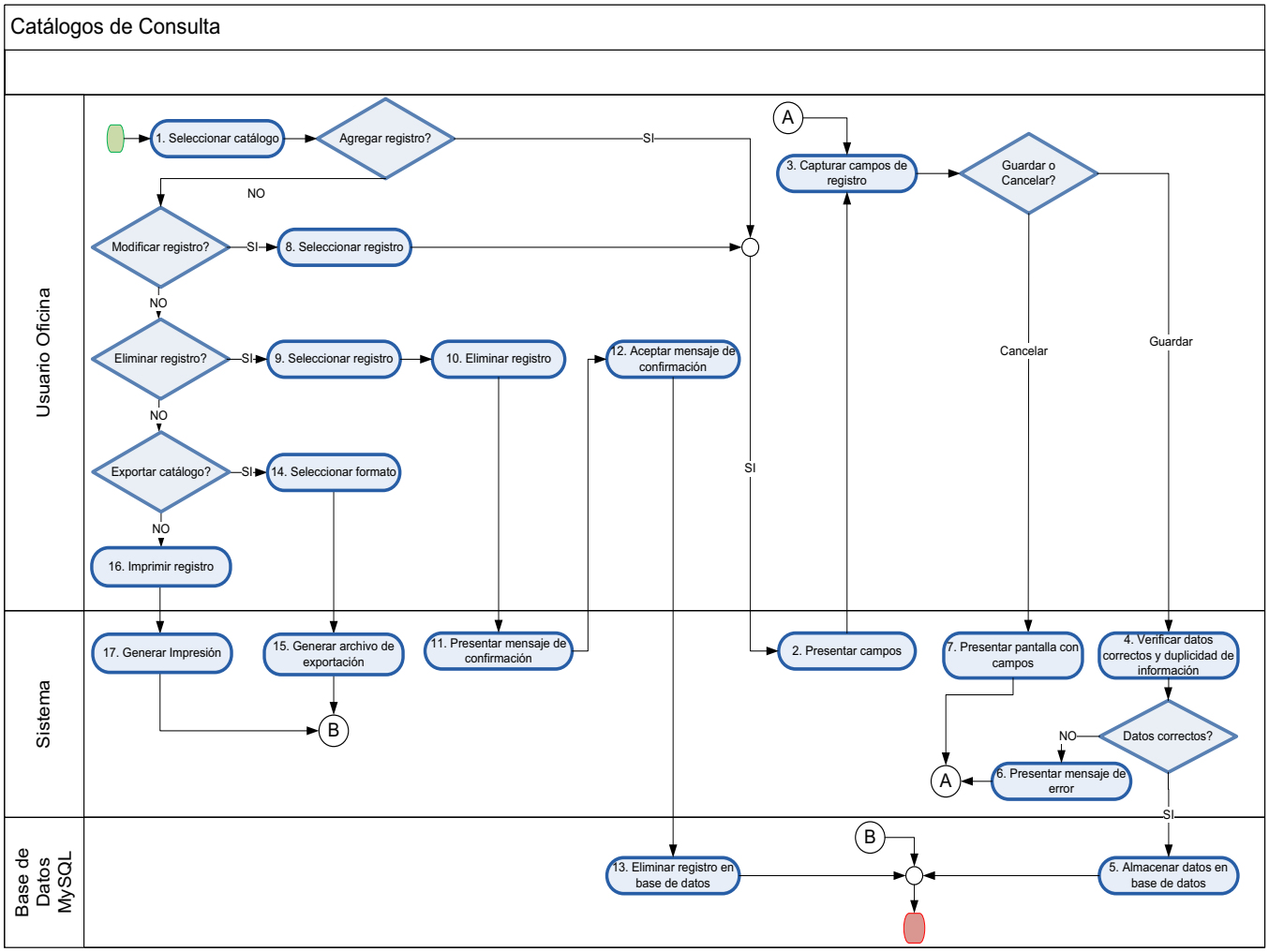
Precondiciones

- Disponibilidad de la base de datos de Oracle.
- Deben existir datos en la tabla entidad de la base de datos de Oracle.
- Deben existir datos en la tabla municipio de la base de datos de Oracle.
- Cuenta de usuario activa y vigente con los accesos necesarios para agregar, modificar y eliminar registros, en la tabla ceremonias de la base de datos de Oracle.
- Software de Excel y PDF para el caso de exportar información con estos formatos.

Post Condiciones

- Alta de una o varias ceremonias.
- Baja de una o varias ceremonias.
- Actualización de los datos de una o varias ceremonias.
- Modificación de la tabla ceremonias.
- Consulta de las ceremonias almacenadas y registradas en el sistema.
- Impresión de la tabla ceremonias.
- Emisión de archivos con formato PDF o Excel, conteniendo los registros del catálogo ceremonias.

Diagrama de procesos



Escenario Principal

Paso	Acción
o	**"Agregar Ceremonia"** Catálogos > Ceremonias > Agregar Ceremonia > Ceremonia Nuevo Registro En la pantalla Nuevo Registro > Agregar Ceremonia, se presentará un formulario donde el usuario deberá de capturar la información referente a la ceremonia que se requiere dar de alta en el sistema (Ver Figura 1). El usuario podrá agregar los datos de una ceremonia pulsando el icono  ** (Ver Notas para Implementación) Sistema: Presentar Área de Trabajo

Nota: Se deberá mostrar en pantalla el área de trabajo en un grid con la información de todos las ceremonias almacenadas en la tabla CEREMONIAS de la base de datos SPOFM (Ver Figura 2).

Usuario:

Pulsar Icono:  [Agregar] **[4]

Sistema:

Validar: Tipo Usuario

Obtener Información: Usuario Firmado en Sesión

[USF].Id_Usuario

[USF].Id_Funcion

[FN].Id_Pagina

Criterios: Consulta

[FN].Id_Funcion = Role_AllAccess

[PF].Id_Funcion = [PRL].Id_Funcion

[FN].Id_Funcion = [USF].Id_Funcion

[USF].Id_Usuario = [US].Id_Usuario (Usuario Firmado en Sesión) ** [6]

[PG].Id_Pagina = [FN].Id_Pagina

[USF].Fecha_Baja = Null

[FN].Fecha_Baja = Null

[PF].Fecha_Baja = Null

[PRL].Fecha_Baja = Null

[PG].Fecha_Baja = Null

Fin Obtener: Usuario Firmado en Sesión

Si es: Cualquier usuario conpermiso Entonces:

Presentar Página: Agregar Ceremonia.jsp

En otro caso:

Presentar Mensaje: No cuenta con los permisos para agregar

Fin Validar: Tipo Usuario

Presentar Información: Nueva Ceremonia.

Datos del Solicitante:
Campo: * Nombre:
Campo: * Apellido Paterno:
Campo: * Apellido Materno:
Campo: * Teléfono uno:
Campo: Teléfono dos:

Datos Ceremonia:
Campo: * Ceremonia:
Campo: * Intención:
Campo: Lugar:

Campo: Otro:
Campo: * Fecha de celebración:
Campo: * Hora:
Campo: Servicios:
Campo: * Donativo:

Fin Presentar: Nueva Celebración

Datos del Solicitante:
Campo: * Nombre:
Campo: * Apellido Paterno:
Campo: * Apellido Materno:
Campo: * Teléfono uno: ** [1]
Campo: Teléfono dos: ** [1]

Datos Ceremonia:
Campo: * Ceremonia:
Campo: * Intención:
Campo: Lugar: ** [2]
Campo: Otro:
Campo: * Fecha de celebración: ** [3]
Campo: * Hora:
Campo: Servicios:
Campo: * Donativo:

Usuario:

Capturar Información: Nueva Celebración

Fin Capturar: Nueva Cewlebración

Pulsar Icono:  [Guardar]

Sistema:

Validar: Información Requerida

Si es: Omitida Entonces:

Presentar Mensaje: Complete los datos requeridos.

(Ver Tabla de validaciones)

Fin Validar: Información Requerida

Agregar Información: Celebración Nueva
[CER].Nombre_Solicitante = Campo: Nombre:
[CER].Apellido_Paterno_Solicitante = Campo: Apellido Paterno:

[CER].Id_Parroquia = Campo: Nombre Parroquia:
[CER].Id_Capilla = Autoincrementar
[CER].Nombre_Capilla = Campo: Nombre:
[CER].Id_Entidad = Campo: Entidad:
[CER].Id_Municipio = Campo: Municipio:
[CER].Calle = Campo: Calle:
[CER].Numero = Campo: Número:
[CER].Colonia = Campo: Colonia:
[CER].Codigo_Postal = Campo: Código Postal:
[CER].Lada = Campo: Lada:
[CER].Telefono_uno = Campo: Teléfono uno:
[CER].Telefono_dos = Campo: Teléfono dos:
[CER].Telefono_tres = Campo: Teléfono tres:
[CER].Fax = Campo: Fax:

	[CER].Correo_Electronico = Campo: Correo Electrónico: [CER].Activo = Campo: Activo:
o	**"Buscar Capilla"** Catálogos > Capillas > Pantalla Capillas Para buscar alguna capilla, el usuario debe ingresar a Catálogos > Capillas donde se mostrará un grid con la información de todas las capillas almacenadas en el sistema, el usuario deberá capturar la palabra completa o el primer carácter de la palabra que se desea buscar. ** (Ver Notas para Implementación) Usuario: Capturar Criterios Búsqueda: Parroquias Campo: Nombre: Campo: Entidad: Campo: Municipio: Campo: Calle: Campo: Número: Campo: Colonia: Campo: Código Postal: Campo: Teléfono uno: Campo: Teléfono dos: Campo: Teléfono tres: Campo: Fax: Campo: Correo Electrónico:**¡Error! No se encuentra el origen de la referencia.** Campo: Activo: Fin Capturar Criterios: Parroquias Nota: Las búsquedas se hacen mediante la combinación de caracteres que se desea filtrar correspondiente al campo o a los campos elegidos por el usuario. Nota: En el encabezado de cada columna se deberán mostrar las cajas de texto para capturar la palabra a buscar. Nota: Sólo se presentará la información de las parroquias que se encuentren almacenadas en el sistema. Sistema: Asignar variables ID CERILLA = [CER].Id_Capilla NOMBRE CERILLA = [CER].Nombre_Capilla ID ENTIDAD = [CER].Id_Entidad ID MUNICIPIO = [CER].Id_Municipio CALLE = [CER].Calle NÚMERO =[CER].Numero COLONIA = [CER].Colonia CÓDIGO POSATAL = [CER].Codigo_Postal

```
LADA = [CER].Lada
TELEFONO UNO = [CER].Telefono_uno
TELEFONO DOS = [CER].Telefono_dos
TELEFONO TRES = [CER].Telefono_tres
FAX = [CER].Fax
CORREO ELECTRONICO = [CER].Correo_Electronico
ACTIVO = [CER].Activo

Obtener: Datos Capilla
  [CER].Id_Capilla
  [CER].Nombre_Capilla
  [CER].Id_Entidad
  [CER].Id_Municipio
  [CER].Calle
  [CER].Numero
  [CER].Colonia
  [CER].Codigo_Postal
  [CER].Lada
  [CER].Telefono_uno
  [CER].Telefono_dos
  [CER].Telefono_tres
  [CER].Fax
  [CER].Correo_Electronico
  [CER].Activo

Criterios:
  ID CERILLA = Campo: Capilla
  NOMBRE CERILLA = Campo: Nombre:
  ID ENTIDAD = Campo: Entidad:
  ID MUNICIPIO = Campo: Municipio:
  CALLE = Campo: Calle:
  NÚMERO = Campo: Número:
  COLONIA = Campo: Colonia:
  CÓDIGO POSTAL = Campo: Código Postal:
  LADA = Campo: Lada:
  TELEFONO UNO = Campo: Teléfono uno:
  TELEFONO DOS = Campo: Telefono dos:
  TELEFONO TRES = Campo: Teléfono tres:
  FAX = Campo: Fax:
  CORREO ELECTRONICO = Campo: Correo Electrónico:
```

	[EN].Id_Entidad = [MU].Id_Entidad [EN].Fecha_Baja = Null [MU].Fecha_Baja = Null Fin Obtener: Datos Capilla Presentar Resultado Búsqueda: Capilla Campo: ID CERILLA = [CER].Id_Capilla Campo: NOMBRE: = [CER].Nombre_Capilla Campo: ENTIDAD = [CER].Entidad Campo: MUNICIPIO = [CER].Municipio Campo: CALLE = [CER].Calle Campo: NÚMERO =[CER].Numero Campo: COLONIA = [CER].Colonia Campo: CÓDIGO POSATAL = [CER].Codigo_Postal Campo: LADA = [CER].Lada Campo: TELEFONO UNO = [CER].Telefono_uno Campo: TELEFONO DOS = [CER].Telefono_dos Campo: TELEFONO TRES = [CER].Telefono_tres Campo: FAX = [CER].Fax Campo: CORREO ELECTRONICO = [CER].Correo_Electronico Campo: ACTIVO = [CER].Activo Fin Presentar: Capilla
3	**"Modificar Datos Capilla"** Catálogos > Capillas > Pantalla Capillas Para modificar la información de alguna capilla, el usuario debe ingresar a Catálogos > Capillas y buscar la capilla que se desea editar, el sistema presentará un grid con la información de todas las capillas almacenadas en el sistema, el usuario deberá capturar la palabra completa o el primer carácter de la palabra que se desea buscar, pulsar sobre la fila seleccionada de acuerdo al resultado de la búsqueda. ** (Ver Notas para Implementación) Usuario: Capturar Criterios Búsqueda: Capillas Campo: Nombre: Campo: Entidad: Campo: Municipio: Campo: Calle: Campo: Número:

Campo: Colonia:

Campo: Código Postal:

Campo: Teléfono uno:

Campo: Teléfono dos:

Campo: Teléfono tres:

Campo: Fax:

Campo: Correo Electrónico:

Campo: Activo:

Fin Capturar Criterios: Capillas

Nota: Se deberá mostrar en la parte superior de la pantalla la información de la capilla elegida por el usuario.

Sistema:

Asignar variables

ID CERILLA = [CER].Id_Capilla

NOMBRE CERILLA = [CER].Nombre_Capilla

ID ENTIDAD = [CER].Id_Entidad

ID MUNICIPIO = [CER].Id_Municipio

CALLE = [CER].Calle

NÚMERO =[CER].Numero

COLONIA = [CER].Colonia

CÓDIGO POSATAL = [CER].Codigo_Postal

LADA = [CER].Lada

TELEFONO UNO = [CER].Telefono_uno

TELEFONO DOS = [CER].Telefono_dos

TELEFONO TRES = [CER].Telefono_tres

FAX = [CER].Fax

CORREO ELECTRONICO = [CER].Correo_Electronico

ACTIVO = [CER].Activo

Obtener: Datos Capilla

[CER].Id_Capilla

[CER].Nombre_Capilla

[CER].Id_Entidad

[CER].Id_Municipio

[CER].Calle

[CER].Numero

[CER].Colonia

[CER].Codigo_Postal

```
    [CER].Lada
    [CER].Telefono_uno
    [CER].Telefono_dos
    [CER].Telefono_tres
    [CER].Fax
    [CER].Correo_Electronico
    [CER].Activo

    Criterios:
    ID CERILLA = Campo: Capilla
    NOMBRE CERILLA = Campo: Nombre:
    ID ENTIDAD = Campo: Entidad:
    ID MUNICIPIO = Campo: Municipio:
    CALLE = Campo: Calle:
    NÚMERO = Campo: Número:
    COLONIA = Campo: Colonia:
    CÓDIGO POSTAL = Campo: Código Postal:
    LADA = Campo: Lada:
    TELEFONO UNO = Campo: Teléfono uno:
    TELEFONO DOS = Campo: Teléfono dos:
    TELEFONO TRES = Campo: Teléfono tres:
    FAX = Campo: Fax:
    CORREO ELECTRONICO = Campo: Correo Electrónico:

    [EN].Id_Entidad = [MU].Id_Entidad
    [EN].Fecha_Baja = Null
    [MU].Fecha_Baja = Null
  Fin Obtener: Datos Capilla


Usuario:
  Capturar la información correspondiente a los siguientes campos: Teléfono uno, Teléfono
  dos, Teléfono tres, Fax, Correo Electrónico, Activo.


  Capturar Criterios Búsqueda: Capillas
    Campo: * Teléfono uno: ** [3]
    Campo: * Teléfono dos: ** [3]
    Campo: * Teléfono tres: ** [3]
    Campo: Fax:
    Campo: Correo Electrónico:
    Campo: * Activo: ** [5]
```

	Fin Capturar Criterios: Capillas Pulsar Icono:  [Guardar] Sistema: Validar: Información Requerida Si es: Omitida Entonces: Presentar Mensaje: Complete los datos requeridos. ** (Ver Tabla de validaciones) Fin Validar: Información Requerida Actualizar Información: Capilla [CER].Id_Capilla = ID CERILLA [Oculto] [CER].Lada = Campo: Lada: [CER].Telefono_uno = Campo: Teléfono uno: [CER].Telefono_dos = Campo: Teléfono dos: [CER].Telefono_tres = Campo: Teléfono tres: [CER].Fax = Campo: Fax: [CER].Correo_Electronico = Campo: Correo Electrónico:

Notas para Implementación	
Id	**Notas**
•	El combo Entidad se debe cargar con la información de la tabla de [EN] ENTIDAD ocupando el siguiente criterio de consulta: Campo: [EN].Nombre_Entidad [Todos los registros existentes en la tabla] La opción por default será: -- Seleccione Opción -- Nota: Las opciones de Entidad se presentarán ordenadas alfabéticamente.
•	Al seleccionar una opción en el combo entidad, el combo Municipio se debe cargar con la información de la tabla [MU] MUNICIPIO ocupando el siguiente criterio de consulta: Obtener Información: Municipio [MU].Id_Municipio [MU].Nombre_Municipio Criterios: [EN].Id_Entidad = [MU].Id_Entidad [EN].Fecha_Baja = Null

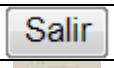
	[MU].Fecha_Baja = Null Fin Obtener: Municipio Cargar Lista: Municipio Campo: Municipio = [MU].Nombre_Municipio Fin Cargar: Municipio Nota: La opción por default será: -- Seleccione Opción -- Nota: Las opciones de Municipio se presentarán ordenadas alfabéticamente.
•	Al seleccionar una opción en el combo Municipio se debe cargar la información de la lada correspondiente al municipio seleccionado: Obtener Información: Lada [MU].Lada Criterios: [Id].Id_Municipio = Campo: **¡Error! No se encuentra el origen de la referencia.** [EN].Id_Entidad = [MU].Id_Entidad [EN].Fecha_Baja = Null [MU].Fecha_Baja = Null Fin Obtener: Lada Cargar: Lada Campo: Lada = [MU].Lada Fin Cargar: Lada Nota: En el campo teléfono se debe mostrar la lada obtenida de acuerdo al municipio seleccionado, permitiendo poder modificar el dato si este fuera incorrecto.

Botones e iconos

Un botón es un dispositivo utilizado para activar alguna función. Los botones pueden ser de diversas formas y tamaños.

Un ícono es un pequeño gráfico en pantalla que identifica y representa a algún objeto, usualmente con algún simbolismo gráfico para establecer una asociación y servirá para realizar una función específica en el sistema.

A continuación se presentan los botones e Íconos utilizados en el sistema:

Botón/Ícono	Descripción de acción
	Botón que permite el acceso al sistema después de capturar nombre de usuario y contraseña.
	Botón que permite salir del sistema.
	**Icono Agregar**: Permite Agregar un nuevo registro. Al dar clic en este ícono, el sistema muestra los campos de captura.
	**Icono Guardar**: Permite guardar los cambios realizados al registro. Al dar clic en este ícono, se presenta una leyenda “Guardando cambios”
	**Icono Guardar Deshabilitado:** Ícono que por default se muestra deshabilitado, se habilita cuando el usuario va a agregar o modificar registros.
	**Icono Cancelar**: Permite cancelar la acción elegida por el usuario, si este decidió cancelar cuando estaba agregando un nuevo registro, el sistema limpia los campos capturados y si la cancelación es cuando el usuario modifica algún registro, no se guarda el registro y se deja en la versión anterior.
	**Icono Cancelar Deshabilitado:** Ícono que por default se muestra deshabilitado, se habilita cuando el usuario va agregar o modificar registros.
	**Icono para exportación en formato Excel**: Al dar clic en este ícono, el sistema genera un archivo con información en formato Excel.
	**Icono para exportación en formato PDF**: Al dar clic en este ícono, el sistema genera un archivo con información en formato PDF.

- Por default se mostrara activo.

Figura 1 – "Alta de una Capilla"

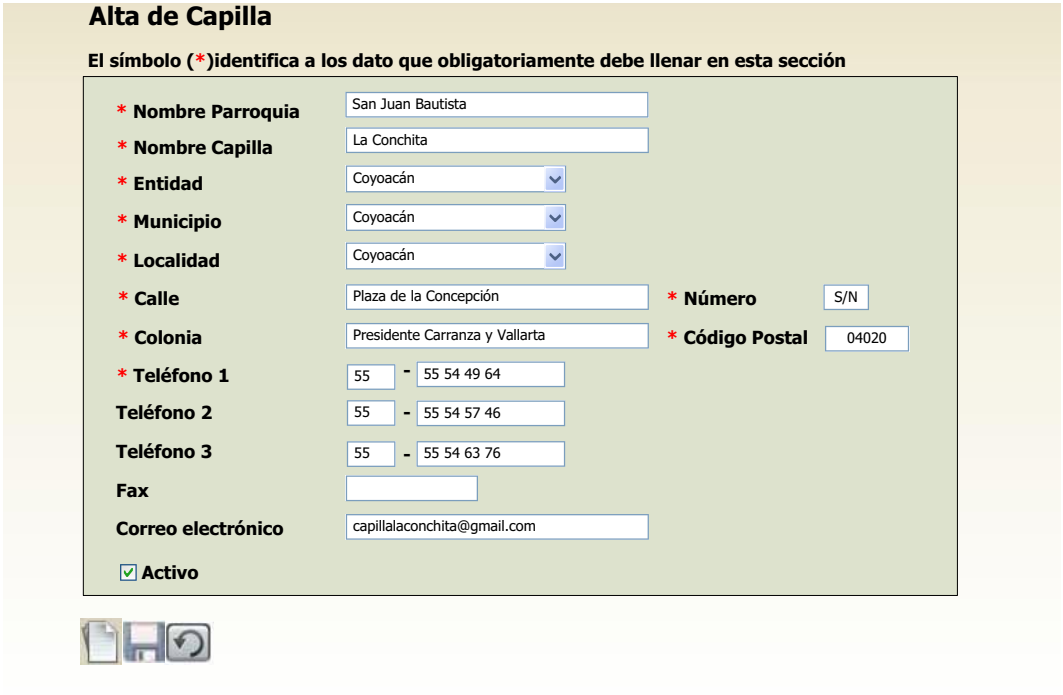


Figura 2 – "Consulta de datos de Capillas"

ID	PARROQUIA	CAPILLA	ENTIDAD	MUNICIPIO	LOCALIDAD	COLONIA	ACTIVO	CALLE	Número	C.P.	TELÉFONO 1	TELÉFONO 2	TELÉFONO 3	FAX	CORREO
1	Santiago Tlatelolco	Casa de la Fraternidad	Tlatelolco	Tlatelolco	Tlatelolco	Tlatelolco	✓	Cerrada Allende	39-B	06900	55 55 49 19 22	55 55 49 19 22		55 55 49 19 22	casafraternidad@gmail.com
2	San Juan Bautista	San José	Hultzilac	Hultzilac	Hultzilac	Hultzilac	✓	Tres Marías	S/N	62510	55 55 54 63 10	55 55 54 63 10	55 55 54 63 10	55 55 54 63 10	capillasanjoseq@gmail.com
3	San Juan Bautista	La Conchita	Coyoacan	Coyoacan	Coyoacan	Coyoacan	✓	Plaza de la Concepción	S/N	04020	55 55 54 49 64			55 55 54 49 64	capillaconchita@hotmail.com
4	San Juan Bautista	Santa Catarina	Coyoacan	Coyoacan	Coyoacan	Coyoacan	✓	Plaza de la Catarina	S/N	04010	55 55 54 29 58			55 55 54 29 58	capillacatarina@yahoo.com.mx
5	San Juan Bautista	San Antonio Panzacola	Coyoacan	Coyoacan	Coyoacan	Coyoacan	✓	Av. Universidad y Francisco Sosa	S/N	04350					capillasantonio@hotmail.com

Bitácora de Cambios

Versión	Fecha	Modificación	Solicitado por:	Modificado por:

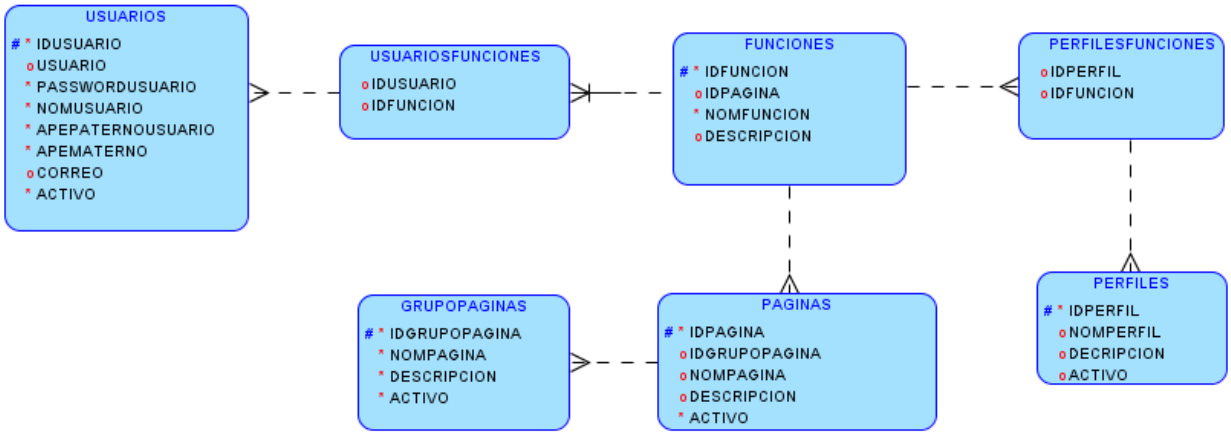
Firma de aceptación

Nombre	Firma
DR. Maricela Bravo Contreras	

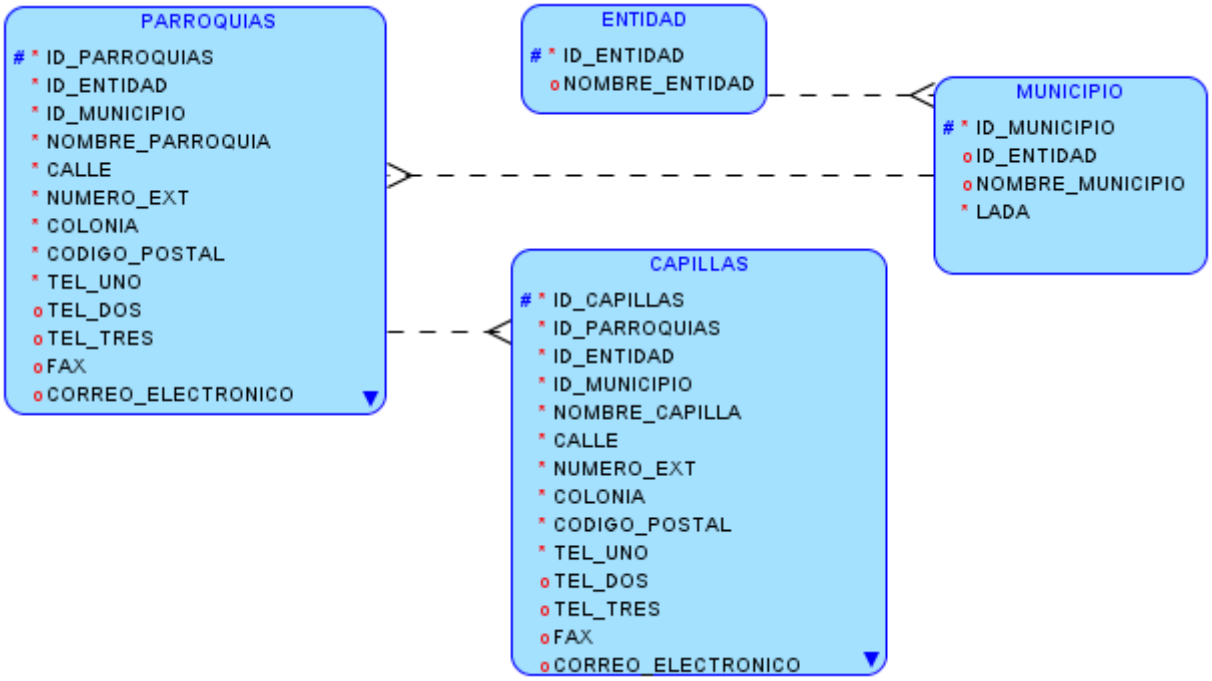
B

Diagramas de Entidad-Relación

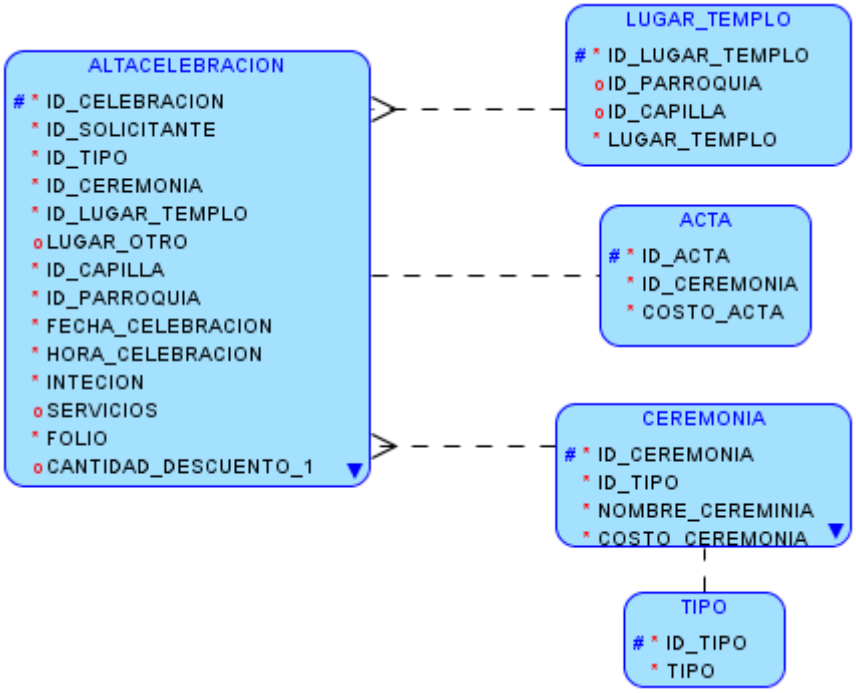
- Seguridad.



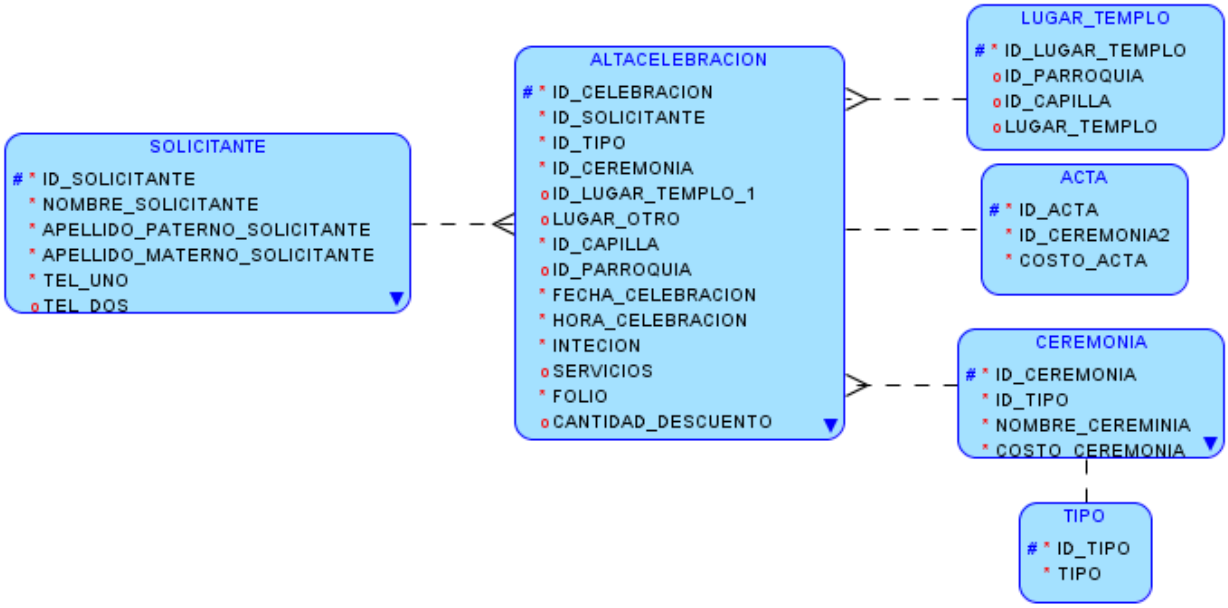
- Parroquias.



- Celebraciones.



- Solicitantes



C

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
USUARIOS	USUARIO		VARCHAR2	250 CHAR	NOMBRE DEL USUARIO
	PASSWORDUSUARIO		VARCHAR2	255 CHAR	CONTRASEÑA ASIGANADA A UN USUARIO PARA EL INGRESO AL SISTEMA
	NOMUSUARIO		VARCHAR2	255 CHAR	NOMBRE(S) DEL USUARIO
	APEPATERNOUSUARIO		VARCHAR2	255 CHAR	APELLIDO PATERNO DEL USUARIO
	APEMATERNOUSUARIO		VARCHAR2	255 CHAR	APELLIDO MATERNO DEL USUARIO
	ACTIVO		CHAR	1 CHAR	INDICA QUE EL REGISTRO ESTA DADO DE BAJA
	IDUSUARIO	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UN USUARIO
	IDAREA		NUMBER	38 BYTE	IDENTIFICADOR DEL AREA AL QUE FUE ASIGNADO EL USUARIO
	CORREO		VARCHAR2	250 CHAR	CORREO ELECTRÓNICO DEL USUARIO
	FECHACREACION		TIMESTAMP (6)		FECHA DE CREACIÓN DEL NUEVO USUARIO

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
USUARIOSFUNCIONES	IDUSUARIO	LLAVE FORANEA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UN USUARIO
	IDFUNCION		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UNA FUNCIÓN

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
FUNCIONES	NOMFUNCION		VARCHAR2	255 CHAR	NOMBRE DE LA FUNCIÓN
	DESCRIPCION		VARCHAR2	255 CHAR	DESCRIPCION DE LA FUNCIÓN
	IDPAGINA		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UNA PÁGINA
	IDFUNCION	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UNA FUNCIÓN

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
PAGINAS	IDGRUPOPAGINA		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UN GRUPO DE PAGINAS
	IDPAGINA	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UNA PÁGINA
	DESCRIPCION		VARCHAR2	255 CHAR	DESCRIPCION DE LA PÁGINA
	ACTIVO		CHAR	I CHAR	INDICA QUE EL REGISTRO ESTA DADO DE BAJA
	NOMPAGINA		VARCHAR2	255 CHAR	NOMBRE DE LA PÁGINA

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
GRUPOSPAGINAS	IDGRUPOPAGINA	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UN GRUPO DE PAGINAS
	NOMGRUPOPAGINA		VARCHAR2	255 CHAR	NOMBRE DEL GRUPO DE PAGINAS
	DESCRIPCION		VARCHAR2	255 CHAR	DESCRIPCIÓN DEL GRUPO DE PAGINAS
	ACTIVO		CHAR	1 CHAR	INDICA QUE EL REGISTRO ESTA DADO DE BAJA

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
PERFILESFUNCIONES	IDPERFIL		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL PERFIL
	IDFUNCION		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UNA FUNCIÓN

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
PERFILES	IDPERFIL	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL PERFIL
	NOMBREPERFIL		VARCHAR2	255 CHAR	NOMBRE DEL PERFIL
	DESCRIPCION		VARCHAR2	4000 CHAR	DESCRIPCIÓN DEL PERFIL
	ACTIVO		CHAR	1 CHAR	INDICA QUE EL REGISTRO ESTA DADO DE BAJA

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
PARROQUIAS	ID_PARROQUIA	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA PARROQUIA
	ID_ENTIDAD	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA ENIDAD
	ID_MUNICIPIO	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL MUNICIPIO
	NOMBRE_PARROQUIA		VARCHAR2	255 CHAR	NOMBRE DE LA PARROQUIA
	CALLE		VARCHAR2	255 CHAR	NOMBRE DE LA CALLE DONDE SE UBICA LA PARROQUIA
	NUMERO_EXT		VARCHAR2	255 CHAR	NUMERO DE LOTE DONDE SE UBICA LA PARROQUIA
	COLONIA		VARCHAR2	255 CHAR	NOMBRE DE LA COLONIA DONDE SE LOCALIZA LA PARROQUIA
	CODIGO_POSTAL		NUMBER	6 BYTE	CÓDIGO POSTAL CORRESPONDIENTE A LA ZONA DONDE SE LOCALIZA LA PARROQUIA
	TEL_UNO		VARCHAR2	255 CHAR	NUMERO TELEFÓNICO PRINCIPAL DE LA PARROQUIA
	TEL_DOS		VARCHAR2	255 CHAR	NUMERO TELEFONICO PROBICIONAL DE LA PARROQUIA
	TEL_TRES		VARCHAR2	255 CHAR	NUMERO TELEFONICO PROBICIONAL DE LA PARROQUIA
	FAX		VARCHAR2	255 CHAR	NUMERO DE FAX DE LA PARROQUIA
	CORREO_ELECTRONICO		VARCHAR2	255 CHAR	CORREO ELECTRÓNICO DE LA PARROQUIA

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
CAPILLAS	ID_CAPILLA	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO DE LA CAPILLA
	ID_PARROQUIA	LLAVE PRIMARIA LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA PARROQUIA
	ID_ENTIDAD	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA ENIDAD
	ID_MUNICIPIO	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL MUNICIPIO
	NOMBRE_CAPILLA		VARCHAR2	255 CHAR	NOMBRE DE LA CAPILLA
	CALLE		VARCHAR2	255 CHAR	NOMBRE DE LA CALLE DONDE SE UBICA LA CAPILLA
	NUMERO_EXT		VARCHAR2	255 CHAR	NUMERO DE LOTE DONDE SE UBICA LA CAPILLA
	COLONIA		VARCHAR2	255 CHAR	NOMBRE DE LA COLONIA DONDE SE LOCALIZA LA CAPILLA
	CODIGO_POSTAL		NUMBER	6 BYTE	CÓDIGO POSTAL CORRESPONDIENTE A LA ZONA DONDE SE LOCALIZA LA CAPILLA
	TEL_UNO		VARCHAR2	255 CHAR	NUMERO TELEFÓNICO PRINCIPAL DE LA CAPILLA
	TEL_DOS		VARCHAR2	255 CHAR	NUMERO TELEFONICO PROBICIONAL DE LA CAPILLA
	TEL_TRES		VARCHAR2	255 CHAR	NUMERO TELEFONICO PROBICIONAL DE LA CAPILLA
	FAX		VARCHAR2	255 CHAR	NUMERO DE FAX DE LA CAPILLA
	CORREO_ELECTRONICO		VARCHAR2	255 CHAR	CORREO ELECTRÓNICO DE LA CAPILLA

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
V_MUNICIPIOS	ID_MUNICIPIO	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL MUNICIPIO
	ID_ENTIDAD	LLAVE PRIMARIA LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA ENTIDAD
	S_NOMBRE_MUNICIPIO		VARCHAR2	255 CHAR	NOMBRE DEL MUNICIPIO
	LADA		VARCHAR2	255 CHAR	LADA RELACIONADA AL MUNICIPIO Y ENDIDAD

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
V_ENTIDADES	ID_ENTIDAD	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA ENTIDAD
	S_NOMBRE_ENTIDAD		VARCHAR2	255 CHAR	NOMBRE DE LA ENTIDAD

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
ALTACELEBRACION	ID_CELEBRACION	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA CELEBRACIÓN
	ID_SOLICIANTE		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UN SOLICITANTE
	ID_TIPO	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL TIPO DE CELEBRACIÓN SOLICITADA
	ID_CEREMONIA	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL TIPO DE CEREMONIA
	ID_LUGAR_TEMPLO	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA ALGUN LUGAR PERTENECIENTE A UN TEMPLO
	LUGAR_OTRO		VARCHAR2	255 BYTE	DIRECCIÓN DEL LUGAR QUE NO PERTENECIENTE A ALGUN TEMPLO, ELEJIDO POR EL SOLICITANTE
	ID_CAPILLA		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA CAPILLA
	ID_PARROQUIA		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA PARROQUIA
	FECHA_CELEBRACION		DATE		FECHA EN AL QUE SE LLEVARA ACABO LA CELEBRACIÓN
	HORA_CELEBRACION		VARCHAR2	20 CHAR	HORA EN LA QUE SE LLEVARA ACABO LA CEREMONIA
	INTENCION		VARCHAR2	255 CHAR	RAZÓN DE LA CELEBRACIÓN
	SERVICIOS		VARCHAR2	255 CHAR	SERVICIOS EXTRAS SOLICITADOS
	FOLIO		NUMBER	9 BYTE	NÚMERO DE FOLIO PARA UN CONTROL
	CANTIDAD_DESCUENTO		NUMBER	38 BYTE	CANTIDAD PARA AJUSTAR UN DONATIVO

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
LUGARTEMPLO	ID_LUGAR_TEMPLO	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL TEMPLO
	ID_PARROQUIA	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA PARROQUIA
	ID_CAPILLA	LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA CAPILLA
	LUGAR_TEMPLO		VARCHAR2	255 CHAR	LUGAR INTERNO DEL TEMPLO QUE NO SEA EN SI EL TEMPLO PRINCITAL

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
CEREMONIAS	ID_CEREMONIA	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UNA CEREMONIA
	ID_TIPO	LLAVE PRIMARIA LLAVE SECUNDARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA UN TIPO DE CEREMONIA
	NOMBRE_CEREMONIA		NUMBER	38 BYTE	NOMBRE DE LA CEREMONIA
	COSTO_CEREMONIA		VARCHAR2	255 CHAR	CANTIDAD DEL DONATIVO RELACIONADO A UNA CEREMONIA

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
TIPO	ID_TIPO	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL TIPO DE CEREMONIA
	TIPO		VARCHAR2	255 CHAR	TIPO DE CEREMONIA

Tabla	Dato en Tabla	Llave	Tipo Dato	Tamaño	Descripción
SOLICITANTES	ID_SOLICITANTE	LLAVE PRIMARIA	NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA EL SOLICITANTE
	ID_CELEBRACION		NUMBER	38 BYTE	IDENTIFICADOR ÚNICO PARA LA CELEBRACIÓN
	NOMBRE_SOLICITANTE		VARCHAR2	255 BYTE	NOMBRE (S) DEL SOLICITANTE
	APELLIDO_PATERNO_SOLICITANTE		VARCHAR2	255 BYTE	APELLIDO PATERNO DEL SOLICITANTE
	APELLIDO_MATERNO_SOLICITANTE		VARCHAR2	255 BYTE	APELLIDO MATERNO DEL SOLICITANTE
	TEL_UNO		VARCHAR2	255 BYTE	NÚMERO TELEFONICO PRINCITAL DEL SOLICITANTE PARA SU LOCALIZACIÓN
	TEL_DOS		VARCHAR2	255 BYTE	NÚMERO TELEFONICO PROBICIONAL DEL SOLICITANTE PARA SU LOCALIZACIÓN

D

En función para el trimestre invierno 2013, cada semana equivalió a 9 horas de trabajo mientras que para el trimestre primavera 2013, cada semana equivalió a 18 horas de trabajo.

Plan de trabajo Emmanuel Oliva García.

La distribución de tiempos en semanas para el trimestre invierno 2013 se hará de la siguiente manera:

Iteración	Descripción	1	2	3	4	5	6	7	8	9	10	11
1	Diseño de las bases de datos, elaboración de las fichas técnicas de casos de uso del módulo control de acceso y de los catálogos de seguridad, parroquias, sacerdotes, ceremonias, cancelaciones y gráficos.											
2	Configuración de los servidores (Apache HTTP Server, Apache Tomcat Server y SunGlasfish), configuración del pool de conexión a las bases de datos. Poblar la base de datos.											
3	Implementación del módulo catálogo de seguridad. Implementación del módulo control de acceso.											
8	Elaboración de reporte y documentación.											

La distribución de tiempos en semanas para el trimestre primavera 2013 se hará de la siguiente manera:

Iteración	Descripción	1	2	3	4	5	6	7	8	9	10	11
5	Implementación del módulo de catálogos: ○ Parroquias. ○ Sacerdotes.											
6	Implementación del módulo de catálogos: ○ Ceremonias. ○ Cancelaciones.											
7	Implementación del módulo de reportes: ○ Gráficos.											
8	Elaboración de reporte y documentación.											

Plan de trabajo Patricia Roldán Ramírez.

La distribución de tiempos en semanas para el trimestre invierno 2013 se hará de la siguiente manera:

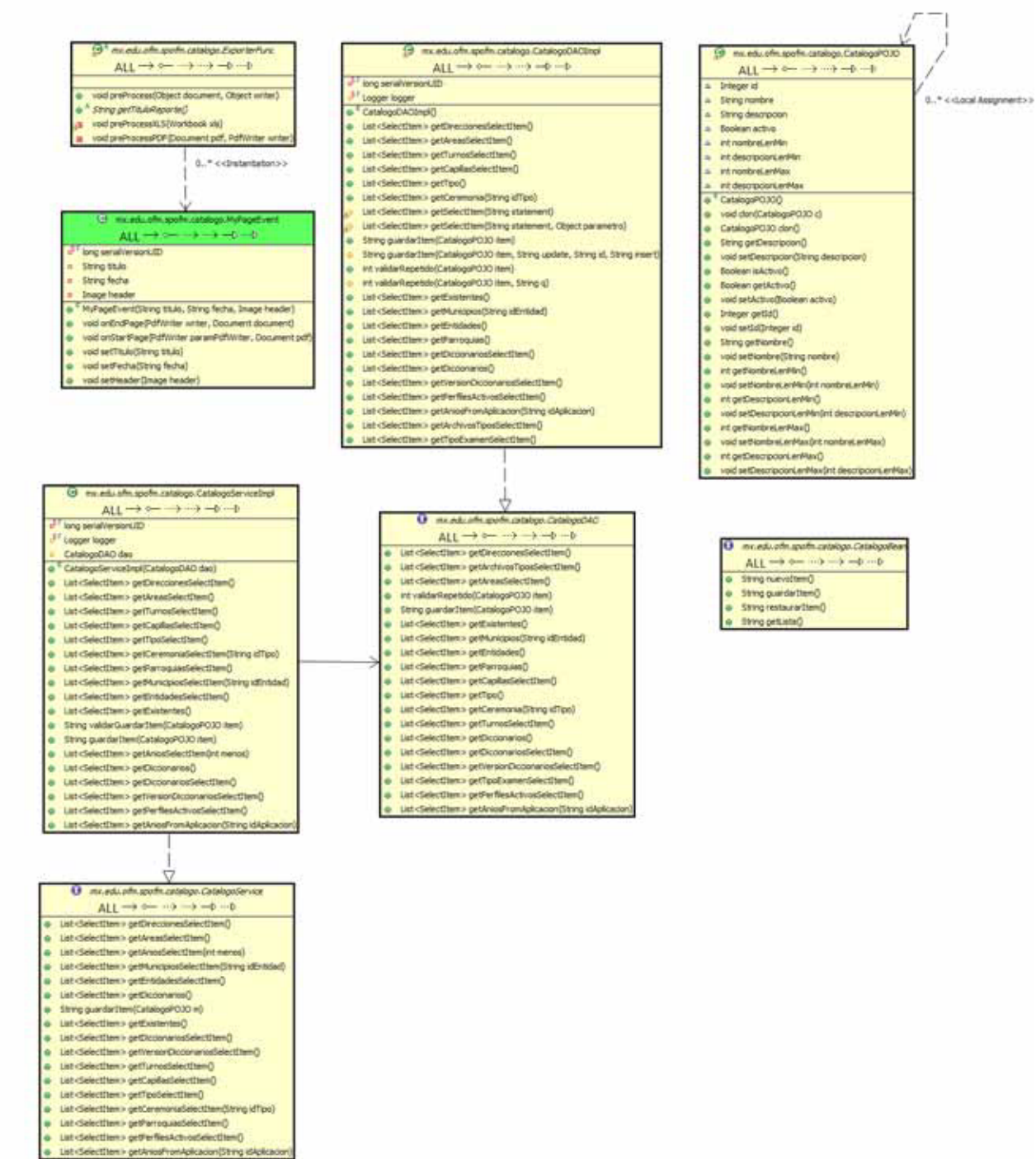
Iteración	Descripción	1	2	3	4	5	6	7	8	9	10	11
1	Diseño de las bases de datos, elaboración de las fichas técnicas de casos de uso del módulo recibos de pagos y de los catálogos de actas, ceremonias, clientes y reportes por día, económico e intención.											
2	Construcción hojas de estilo y menús, Poblar la base de datos.											
4	Implementación del módulo catálogo de actas.											
8	Elaboración de reporte y documentación.											

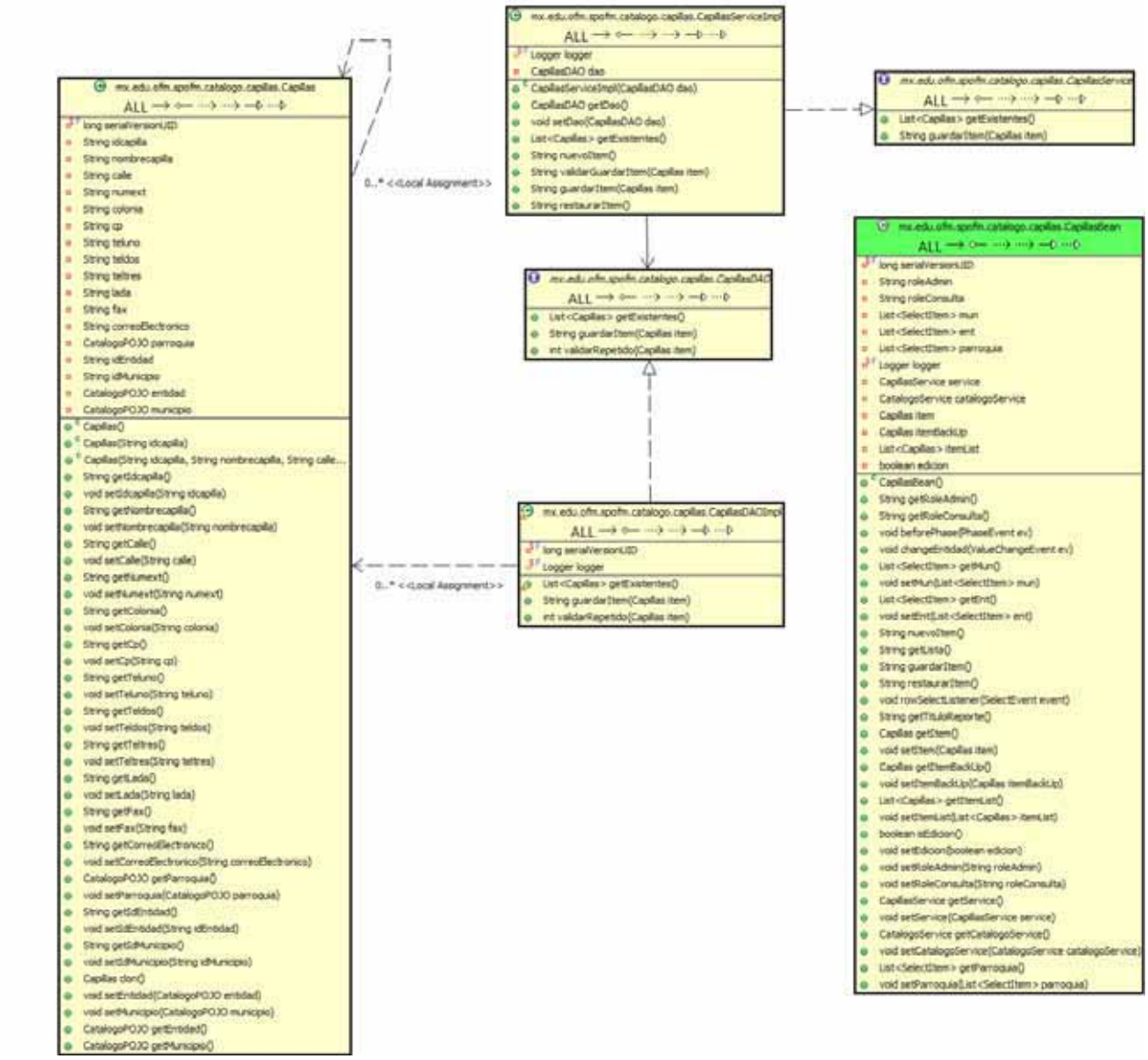
La distribución de tiempos en semanas para el trimestre primavera 2013 se hará de la siguiente manera:

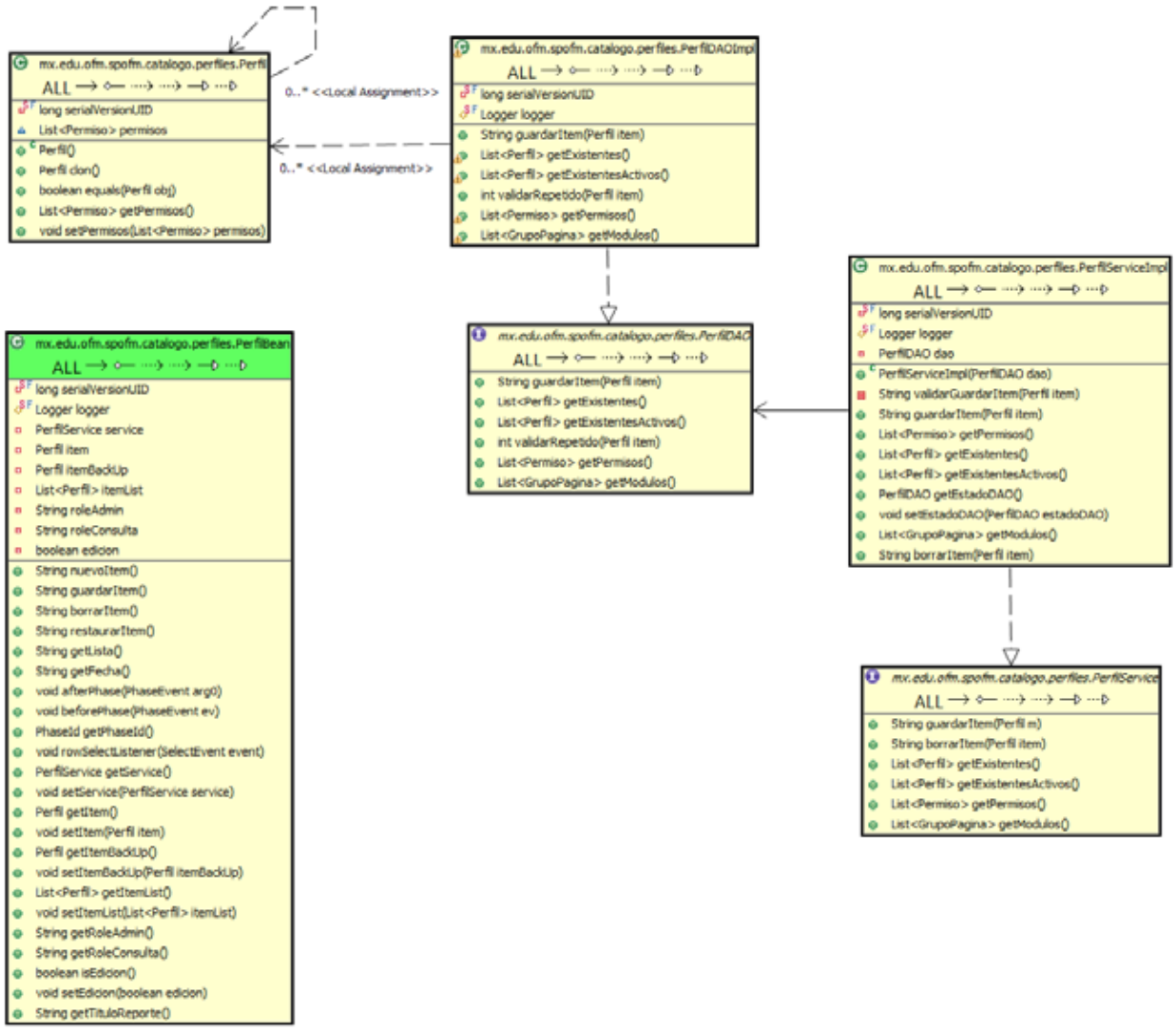
Iteración	Descripción	1	2	3	4	5	6	7	8	9	10	11
5	Implementación del módulo de catálogos: o Ceremonias. o Clientes.											
6	Implementación del módulo de pago.											
7	Implementación del módulo de reportes: o Por día. o Económico. o Intención.											
8	Elaboración de reporte y documentación.											

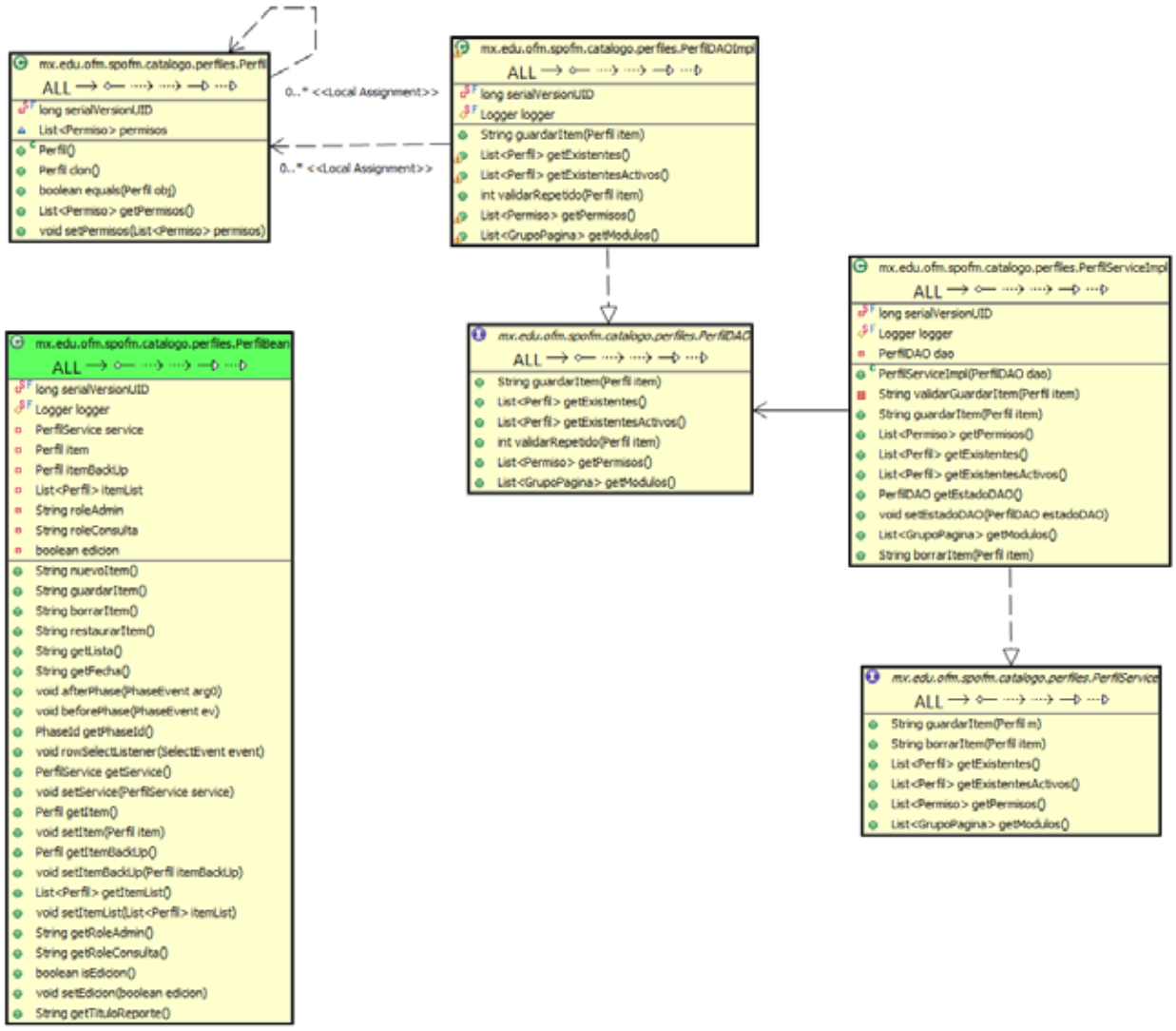
E

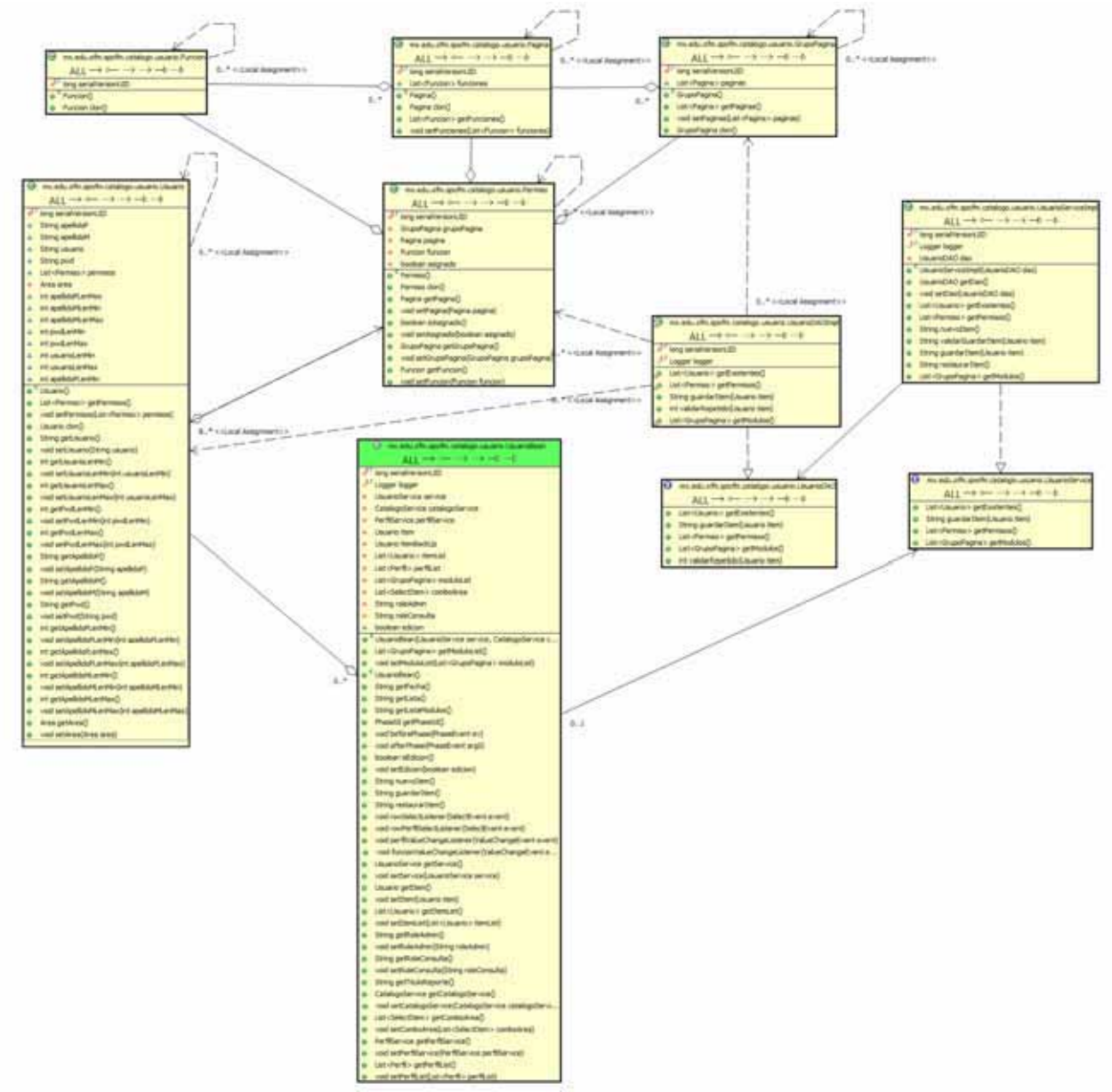
Diagrama de clases del código fuente.

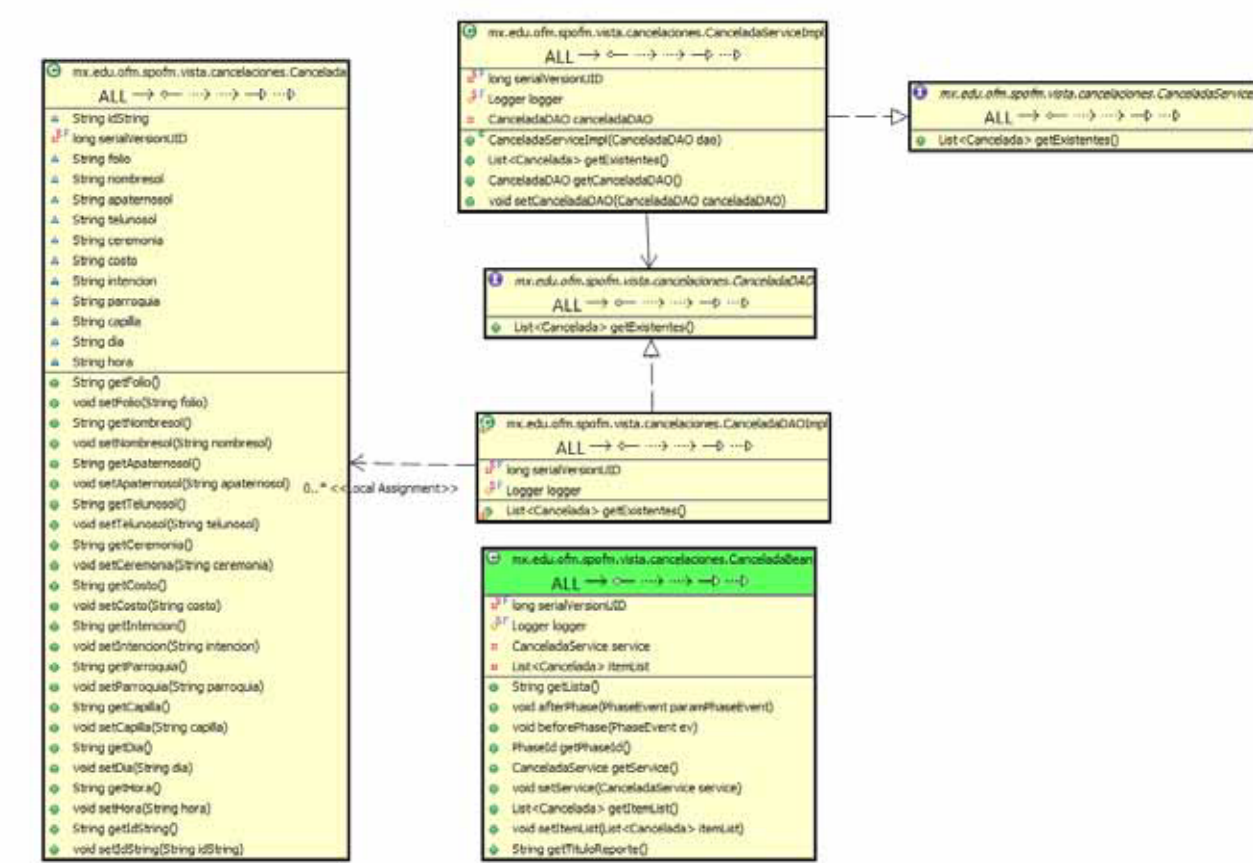


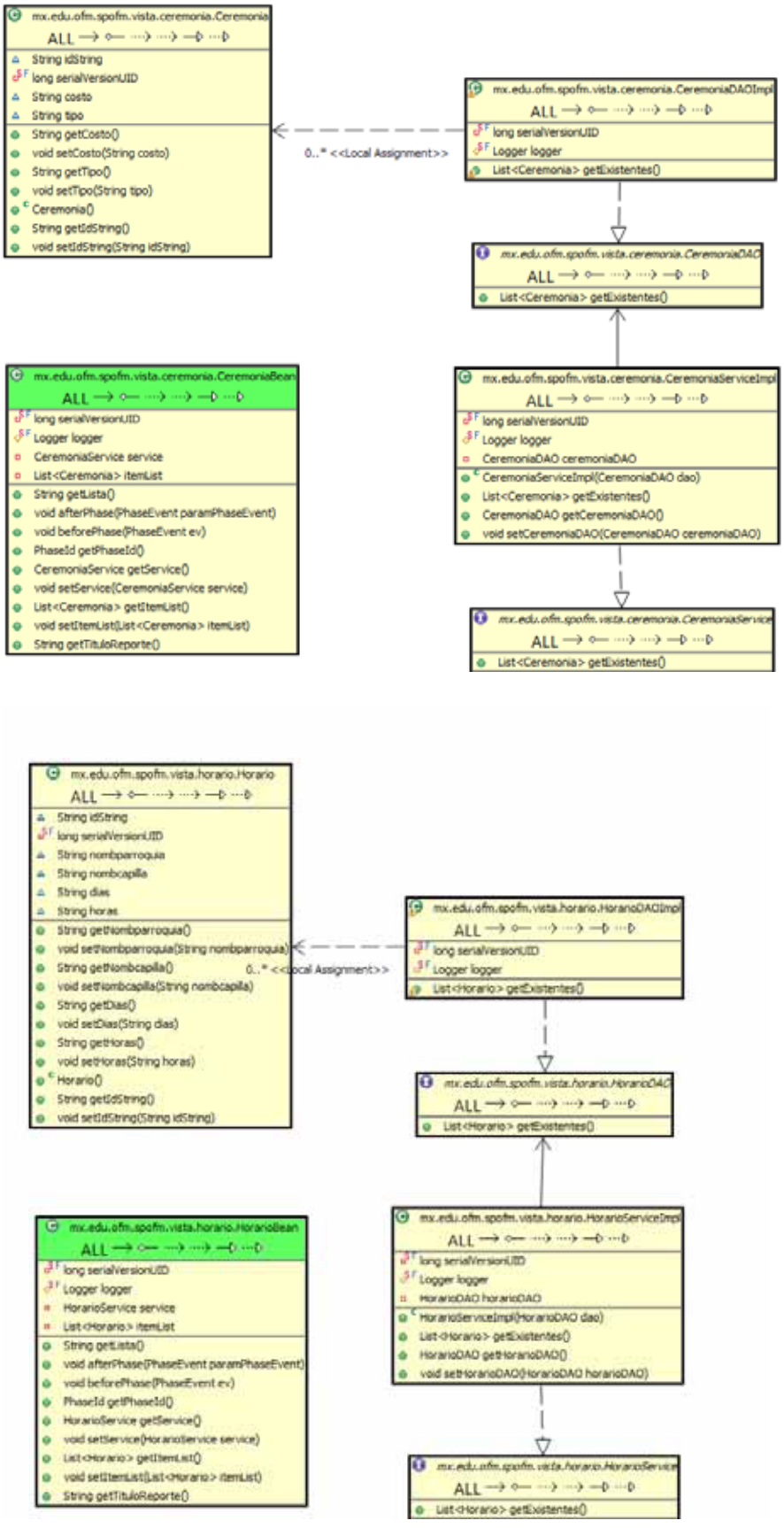

mx.edu.ofm.spofm.catalogo.capillas.CapillasServiceImpl
ALL
Logger logger
CapillasDAO dao
CapillasServiceImpl(CapillasDAO dao)
CapillasDAO getDao()
void setDao(CapillasDAO dao)
List<Capillas> getExistentes()
String nuevoItem()
String validarGuardarItem(Capillas item)
String guardarItem(Capillas item)
String restaurarItem()
mx.edu.ofm.spofm.catalogo.capillas.CapillasService
ALL
List<Capillas> getExistentes()
String guardarItem(Capillas item)
mx.edu.ofm.spofm.catalogo.capillas.Capillas
ALL
long serialVersionUID
String idcapilla
String nombrecapilla
String calle
String numext
String colonia
String cp
String teluno
String teldos
String teltres
String lada
String fax
String correoElectronico
CatalogoPOJO parroquia
String idEntidad
String idMunicipio
CatalogoPOJO entidad
CatalogoPOJO municipio
Capillas()
Capillas(String idcapilla)
Capillas(String idcapilla, String nombrecapilla, String calle...
String getIdcapilla()
void setIdcapilla(String idcapilla)
String getNombrecapilla()
void setNombrecapilla(String nombrecapilla)
String getCalle()
void setCalle(String calle)
String getNumext()
void setNumext(String numext)
String getColonia()
void setColonia(String colonia)
String getCp()
void setCp(String cp)
String getTeluno()
void setTeluno(String teluno)
String getTeldos()
void setTeldos(String teldos)
String getTeltres()
void setTeltres(String teltres)
String getLada()
void setLada(String lada)
String getFax()
void setFax(String fax)
String getCorreoElectronico()
void setCorreoElectronico(String correoElectronico)
CatalogoPOJO getParroquia()
void setParroquia(CatalogoPOJO parroquia)
String getIdEntidad()
void setIdEntidad(String idEntidad)
String getIdMunicipio()
void setIdMunicipio(String idMunicipio)
Capillas clon()
void setEntidad(CatalogoPOJO entidad)
void setMunicipio(CatalogoPOJO municipio)
CatalogoPOJO getEntidad()
CatalogoPOJO getMunicipio()
0..* <<Local Assignment>>
mx.edu.ofm.spofm.catalogo.capillas.CapillasDAO
ALL
List<Capillas> getExistentes()
String guardarItem(Capillas item)
int validarRepetido(Capillas item)
mx.edu.ofm.spofm.catalogo.capillas.CapillasDAOImpl
ALL
long serialVersionUID
Logger logger
List<Capillas> getExistentes()
String guardarItem(Capillas item)
int validarRepetido(Capillas item)
0..* <<Local Assignment>>
mx.edu.ofm.spofm.catalogo.capillas.CapillasBean
ALL
long serialVersionUID
String roleAdmin
String roleConsulta
List<SelectItem> mun
List<SelectItem> ent
List<SelectItem> parroquia
Logger logger
CapillasService service
CatalogosService catalogosService
Capillas item
Capillas itemBackUp
List<Capillas> itemList
boolean edicion
CapillasBean()
String getRoleAdmin()
String getRoleConsulta()
void beforePhase(PhaseEvent ev)
void changeEntidad(ValueChangeEvent ev)
List<SelectItem> getMun()
void setMun(List<SelectItem> mun)
List<SelectItem> getEnt()
void setEnt(List<SelectItem> ent)
String nuevoItem()
String getLista()
String guardarItem()
String restaurarItem()
void rowSelectListener(SelectEvent event)
String getTituloReporte()
Capillas getItem()
void setItem(Capillas item)
Capillas getItemBackUp()
void setItemBackUp(Capillas itemBackUp)
List<Capillas> getItemList()
void setItemList(List<Capillas> itemList)
boolean isEdicion()
void setEdicion(boolean edicion)
void setRoleAdmin(String roleAdmin)
void setRoleConsulta(String roleConsulta)
CapillasService getService()
void setService(CapillasService service)
CatalogosService getCatalogosService()
void setCatalogosService(CatalogosService catalogosService)
List<SelectItem> getParroquia()
void setParroquia(List<SelectItem> parroquia)

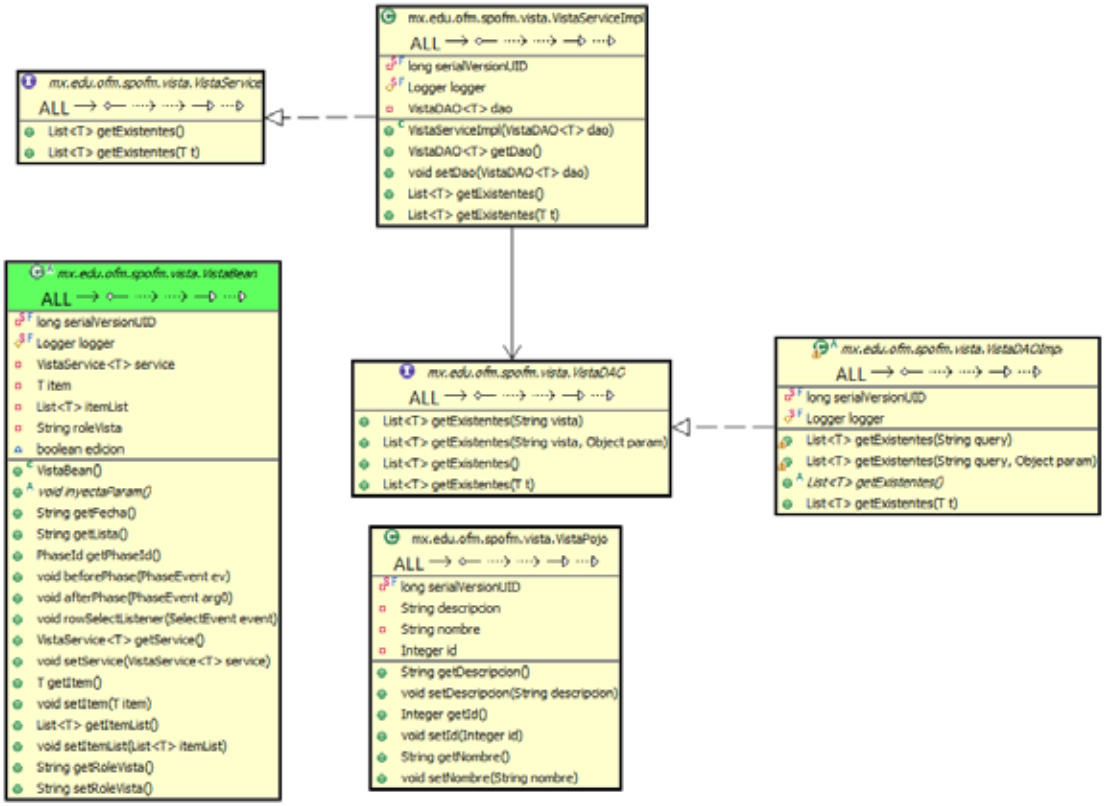

mx.edu.ofm.spofm.catalogo.perfiles.Perfil
ALL
long serialVersionUID
List<Permiso> permisos
Perfil()
Perfil clon()
boolean equals(Perfil obj)
List<Permiso> getPermisos()
void setPermisos(List<Permiso> permisos)
0..* <<Local Assignment>>
0..* <<Local Assignment>>
mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAOImpl
ALL
long serialVersionUID
Logger logger
String guardarItem(Perfil item)
List<Perfil> getExistentes()
List<Perfil> getExistentesActivos()
int validarRepetido(Perfil item)
List<Permiso> getPermisos()
List<GrupoPagina> getModulos()
mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAO
ALL
String guardarItem(Perfil item)
List<Perfil> getExistentes()
List<Perfil> getExistentesActivos()
int validarRepetido(Perfil item)
List<Permiso> getPermisos()
List<GrupoPagina> getModulos()
mx.edu.ofm.spofm.catalogo.perfiles.PerfilServiceImpl
ALL
long serialVersionUID
Logger logger
PerfilDAO dao
PerfilServiceImpl(PerfilDAO dao)
String validarGuardarItem(Perfil item)
String guardarItem(Perfil item)
List<Permiso> getPermisos()
List<Perfil> getExistentes()
List<Perfil> getExistentesActivos()
PerfilDAO getEstadoDAO()
void setEstadoDAO(PerfilDAO estadoDAO)
List<GrupoPagina> getModulos()
String borrarItem(Perfil item)
mx.edu.ofm.spofm.catalogo.perfiles.PerfilService
ALL
String guardarItem(Perfil m)
String borrarItem(Perfil item)
List<Perfil> getExistentes()
List<Perfil> getExistentesActivos()
List<Permiso> getPermisos()
List<GrupoPagina> getModulos()
mx.edu.ofm.spofm.catalogo.perfiles.PerfilBean
ALL
long serialVersionUID
Logger logger
PerfilService service
Perfil item
Perfil itemBackUp
List<Perfil> itemList
String roleAdmin
String roleConsulta
boolean edicion
String nuevoItem()
String guardarItem()
String borrarItem()
String restaurarItem()
String getLista()
String getFecha()
void afterPhase(PhaseEvent arg0)
void beforePhase(PhaseEvent ev)
PhaseId getPhaseId()
void rowSelectListener(SelectEvent event)
PerfilService getService()
void setService(PerfilService service)
Perfil getItem()
void setItem(Perfil item)
Perfil getItemBackUp()
void setItemBackUp(Perfil itemBackUp)
List<Perfil> getItemList()
void setItemList(List<Perfil> itemList)
String getRoleAdmin()
String getRoleConsulta()
boolean isEdicion()
void setEdicion(boolean edicion)
String getTituloReporte()


mx.edu.ofm.spofm.catalogo.perfiles.Perfil
long serialVersionUID
List<Permiso> permisos
Perfil()
Perfil clon()
boolean equals(Perfil obj)
List<Permiso> getPermisos()
void setPermisos(List<Permiso> permisos)
0..* <<Local Assignment>>
0..* <<Local Assignment>>
mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAOImpl
long serialVersionUID
Logger logger
String guardarItem(Perfil item)
List<Perfil> getExistentes()
List<Perfil> getExistentesActivos()
int validarRepetido(Perfil item)
List<Permiso> getPermisos()
List<GrupoPagina> getModulos()
mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAO
String guardarItem(Perfil item)
List<Perfil> getExistentes()
List<Perfil> getExistentesActivos()
int validarRepetido(Perfil item)
List<Permiso> getPermisos()
List<GrupoPagina> getModulos()
mx.edu.ofm.spofm.catalogo.perfiles.PerfilServiceImpl
long serialVersionUID
Logger logger
PerfilDAO dao
PerfilServiceImpl(PerfilDAO dao)
String validarGuardarItem(Perfil item)
String guardarItem(Perfil item)
List<Permiso> getPermisos()
List<Perfil> getExistentes()
List<Perfil> getExistentesActivos()
PerfilDAO getEstadoDAO()
void setEstadoDAO(PerfilDAO estadoDAO)
List<GrupoPagina> getModulos()
String borrarItem(Perfil item)
mx.edu.ofm.spofm.catalogo.perfiles.PerfilService
String guardarItem(Perfil m)
String borrarItem(Perfil item)
List<Perfil> getExistentes()
List<Perfil> getExistentesActivos()
List<Permiso> getPermisos()
List<GrupoPagina> getModulos()
mx.edu.ofm.spofm.catalogo.perfiles.PerfilBean
long serialVersionUID
Logger logger
PerfilService service
Perfil item
Perfil itemBackUp
List<Perfil> itemList
String roleAdmin
String roleConsulta
boolean edicion
String nuevoItem()
String guardarItem()
String borrarItem()
String restaurarItem()
String getLista()
String getFecha()
void afterPhase(PhaseEvent arg0)
void beforePhase(PhaseEvent ev)
PhaseId getPhaseId()
void rowSelectListener(SelectEvent event)
PerfilService getService()
void setService(PerfilService service)
Perfil getItem()
void setItem(Perfil item)
Perfil getItemBackUp()
void setItemBackUp(Perfil itemBackUp)
List<Perfil> getItemList()
void setItemList(List<Perfil> itemList)
String getRoleAdmin()
String getRoleConsulta()
boolean isEdicion()
void setEdicion(boolean edicion)
String getTituloReporte()

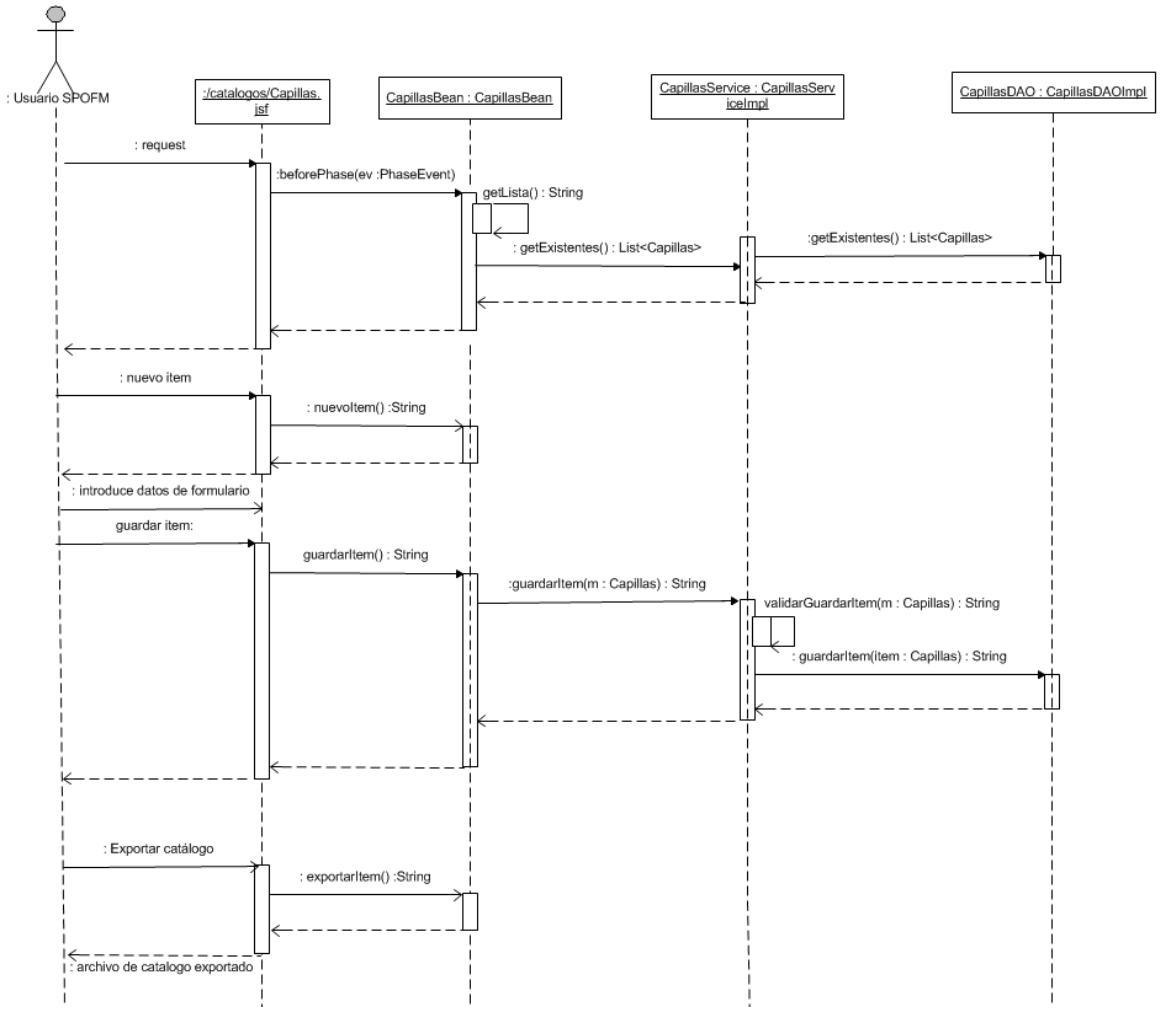


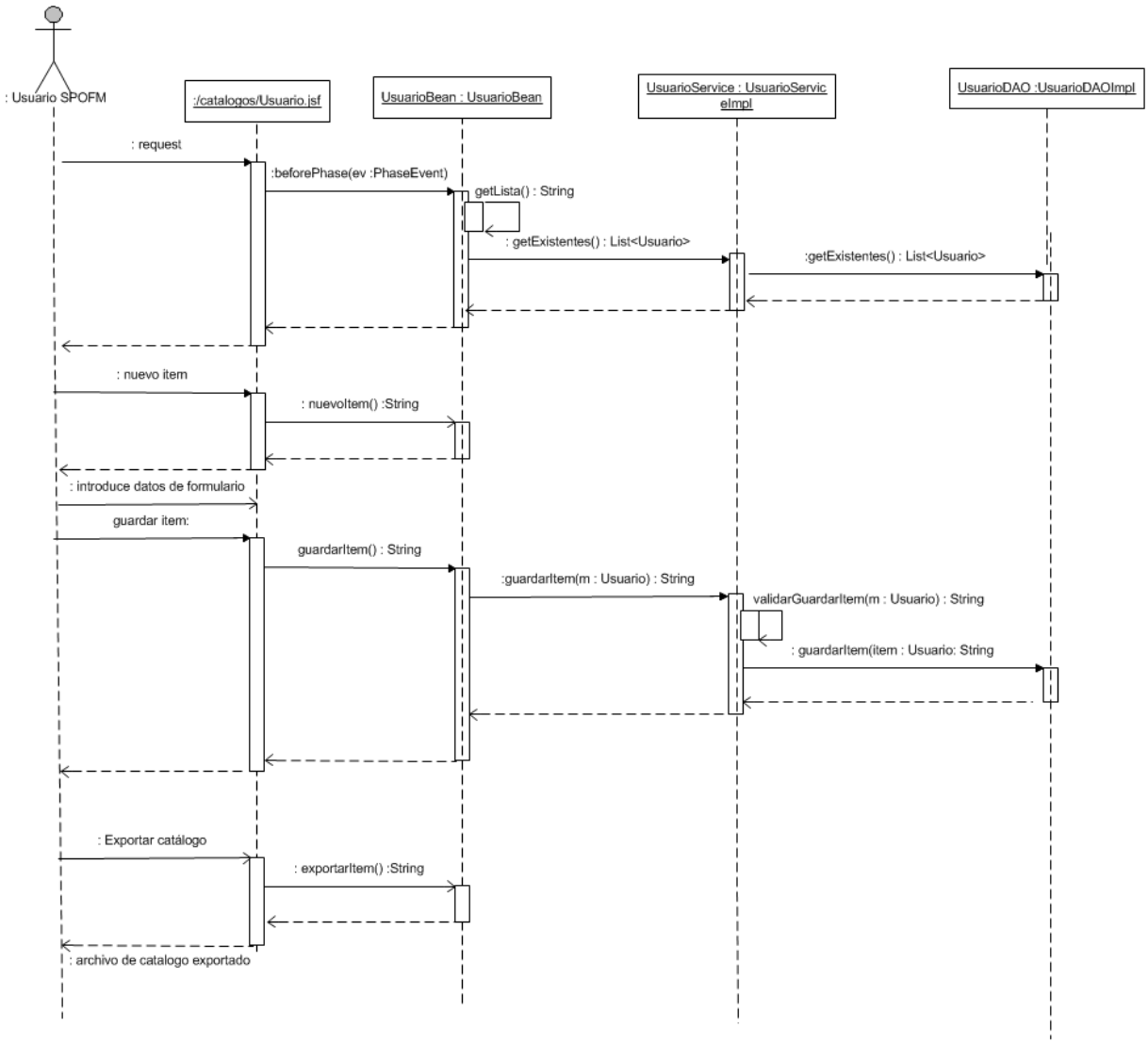

mx.edu.ofm.spofm.vista.cancelaciones.CanceladaServiceImpl
long serialVersionUID
Logger logger
CanceladaDAO canceladaDAO
CanceladaServiceImpl(CanceladaDAO dao)
List<Cancelada> getExistentes()
CanceladaDAO getCanceladaDAO()
void setCanceladaDAO(CanceladaDAO canceladaDAO)
mx.edu.ofm.spofm.vista.cancelaciones.CanceladaService
List<Cancelada> getExistentes()
mx.edu.ofm.spofm.vista.cancelaciones.CanceladaDAO
List<Cancelada> getExistentes()
mx.edu.ofm.spofm.vista.cancelaciones.Cancelada
String idString
long serialVersionUID
String folio
String nombresol
String apaternosol
String telunosol
String ceremonia
String costo
String intencion
String parroquia
String capilla
String dia
String hora
String getFolio()
void setFolio(String folio)
String getNombresol()
void setNombresol(String nombresol)
String getApaternosol()
void setApaternosol(String apaternosol)
String getTelunosol()
void setTelunosol(String telunosol)
String getCeremonia()
void setCeremonia(String ceremonia)
String getCosto()
void setCosto(String costo)
String getIntencion()
void setIntencion(String intencion)
String getParroquia()
void setParroquia(String parroquia)
String getCapilla()
void setCapilla(String capilla)
String getDia()
void setDia(String dia)
String getHora()
void setHora(String hora)
String getIdString()
void setIdString(String idString)
0..* <<Local Assignment>>
mx.edu.ofm.spofm.vista.cancelaciones.CanceladaDAOImpl
long serialVersionUID
Logger logger
List<Cancelada> getExistentes()
mx.edu.ofm.spofm.vista.cancelaciones.CanceladaBean
long serialVersionUID
Logger logger
CanceladaService service
List<Cancelada> itemList
String getLista()
void afterPhase(PhaseEvent paramPhaseEvent)
void beforePhase(PhaseEvent ev)
PhaseId getPhaseId()
CanceladaService getService()
void setService(CanceladaService service)
List<Cancelada> getItemList()
void setItemList(List<Cancelada> itemList)
String getTituloReporte()
mx.edu.ofm.spofm.vista.tipo.Tipo
String idString
long serialVersionUID
String folio
String nombresol
String apaternosol
String telunosol
String ceremonia
String costo
String intencion
String parroquia
String capilla
String dia
String hora
String getFolio()
void setFolio(String folio)
String getNombresol()
void setNombresol(String nombresol)
String getApaternosol()
void setApaternosol(String apaternosol)
String getTelunosol()
void setTelunosol(String telunosol)
String getCeremonia()
void setCeremonia(String ceremonia)
String getCosto()
void setCosto(String costo)
String getIntencion()
void setIntencion(String intencion)
String getParroquia()
void setParroquia(String parroquia)
String getCapilla()
void setCapilla(String capilla)
String getDia()
void setDia(String dia)
String getHora()
void setHora(String hora)
String getIdString()
void setIdString(String idString)
mx.edu.ofm.spofm.vista.tipo.TipoDAOImpl
long serialVersionUID
Logger logger
List<Tipo> getExistentes()
0..* <<Local Assignment>>
mx.edu.ofm.spofm.vista.tipo.TipoDAO
List<Tipo> getExistentes()
mx.edu.ofm.spofm.vista.tipo.TipoServiceImpl
long serialVersionUID
Logger logger
TipoDAO tipoDAO
TipoServiceImpl(TipoDAO dao)
List<Tipo> getExistentes()
TipoDAO getTipoDAO()
void setTipoDAO(TipoDAO tipoDAO)
mx.edu.ofm.spofm.vista.tipo.TipoBean
long serialVersionUID
Logger logger
TipoService service
List<Tipo> itemList
String getLista()
void afterPhase(PhaseEvent paramPhaseEvent)
void beforePhase(PhaseEvent ev)
PhaseId getPhaseId()
TipoService getService()
void setService(TipoService service)
List<Tipo> getItemList()
void setItemList(List<Tipo> itemList)
String getTituloReporte()
mx.edu.ofm.spofm.vista.tipo.TipoService
List<Tipo> getExistentes()

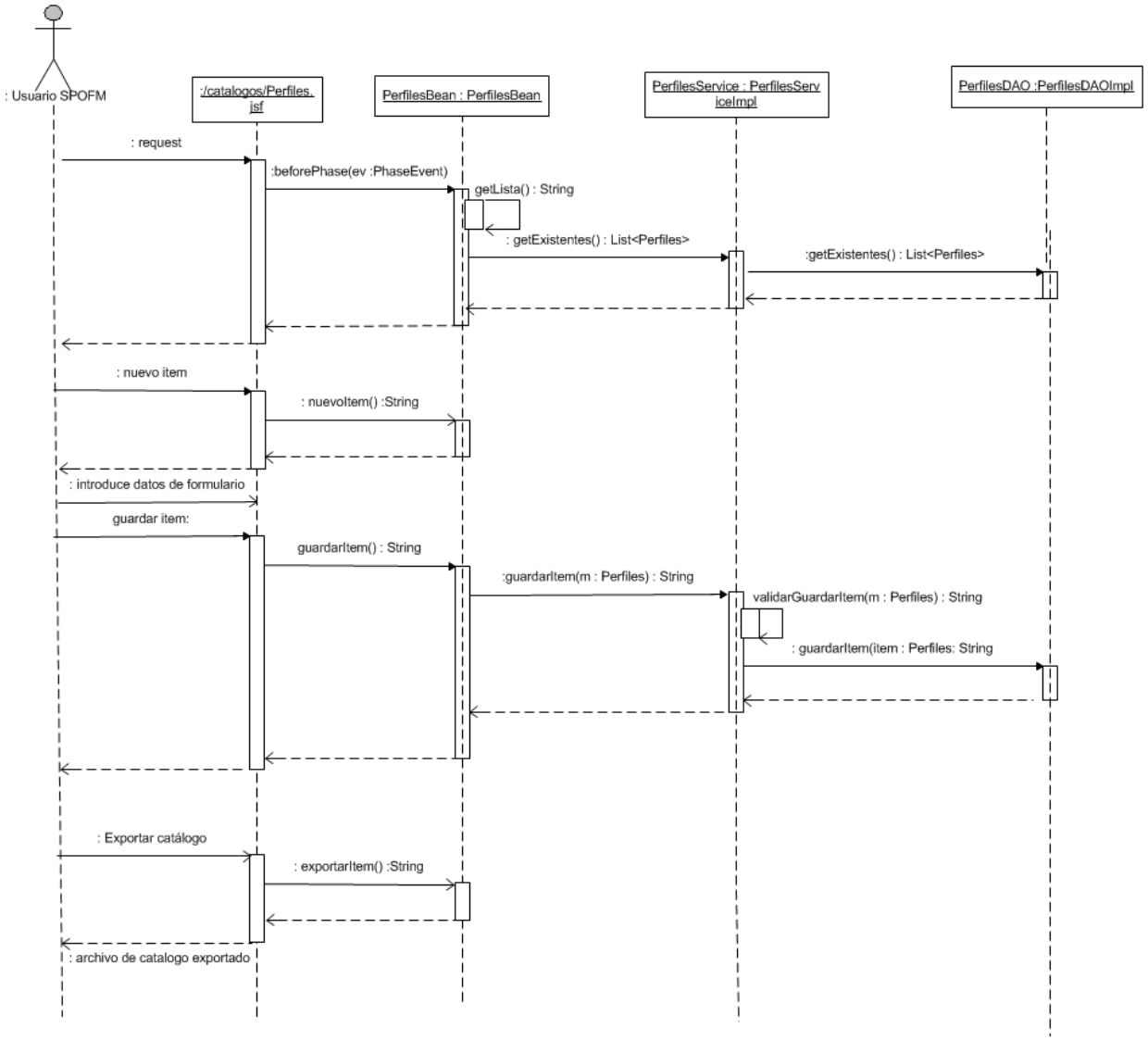

mx.edu.ofm.spofm.vista.ceremonia.Ceremonia
ALL
String idString
long serialVersionUID
String costo
String tipo
String getCosto()
void setCosto(String costo)
String getTipo()
void setTipo(String tipo)
Ceremonia()
String getIdString()
void setIdString(String idString)
0..* <<Local Assignment>>
mx.edu.ofm.spofm.vista.ceremonia.CeremoniaDAOImpl
ALL
long serialVersionUID
Logger logger
List<Ceremonia> getExistentes()
mx.edu.ofm.spofm.vista.ceremonia.CeremoniaDAO
ALL
List<Ceremonia> getExistentes()
mx.edu.ofm.spofm.vista.ceremonia.CeremoniaBean
ALL
long serialVersionUID
Logger logger
CeremoniaService service
List<Ceremonia> itemList
String getLista()
void afterPhase(PhaseEvent paramPhaseEvent)
void beforePhase(PhaseEvent ev)
PhaseId getPhaseId()
CeremoniaService getService()
void setService(CeremoniaService service)
List<Ceremonia> getItemList()
void setItemList(List<Ceremonia> itemList)
String getTituloReporte()
mx.edu.ofm.spofm.vista.ceremonia.CeremoniaServiceImpl
ALL
long serialVersionUID
Logger logger
CeremoniaDAO ceremoniaDAO
CeremoniaServiceImpl(CeremoniaDAO dao)
List<Ceremonia> getExistentes()
CeremoniaDAO getCeremoniaDAO()
void setCeremoniaDAO(CeremoniaDAO ceremoniaDAO)
mx.edu.ofm.spofm.vista.ceremonia.CeremoniaService
ALL
List<Ceremonia> getExistentes()
mx.edu.ofm.spofm.vista.horario.Horario
ALL
String idString
long serialVersionUID
String nombparroquia
String nombcapilla
String dias
String horas
String getNombparroquia()
void setNombparroquia(String nombparroquia)
String getNombcapilla()
void setNombcapilla(String nombcapilla)
String getDias()
void setDias(String dias)
String getHoras()
void setHoras(String horas)
Horario()
String getIdString()
void setIdString(String idString)
0..* <<Local Assignment>>
mx.edu.ofm.spofm.vista.horario.HorarioDAOImpl
ALL
long serialVersionUID
Logger logger
List<Horario> getExistentes()
mx.edu.ofm.spofm.vista.horario.HorarioDAO
ALL
List<Horario> getExistentes()
mx.edu.ofm.spofm.vista.horario.HorarioBean
ALL
long serialVersionUID
Logger logger
HorarioService service
List<Horario> itemList
String getLista()
void afterPhase(PhaseEvent paramPhaseEvent)
void beforePhase(PhaseEvent ev)
PhaseId getPhaseId()
HorarioService getService()
void setService(HorarioService service)
List<Horario> getItemList()
void setItemList(List<Horario> itemList)
String getTituloReporte()
mx.edu.ofm.spofm.vista.horario.HorarioServiceImpl
ALL
long serialVersionUID
Logger logger
HorarioDAO horarioDAO
HorarioServiceImpl(HorarioDAO dao)
List<Horario> getExistentes()
HorarioDAO getHorarioDAO()
void setHorarioDAO(HorarioDAO horarioDAO)
mx.edu.ofm.spofm.vista.horario.HorarioService
ALL
List<Horario> getExistentes()

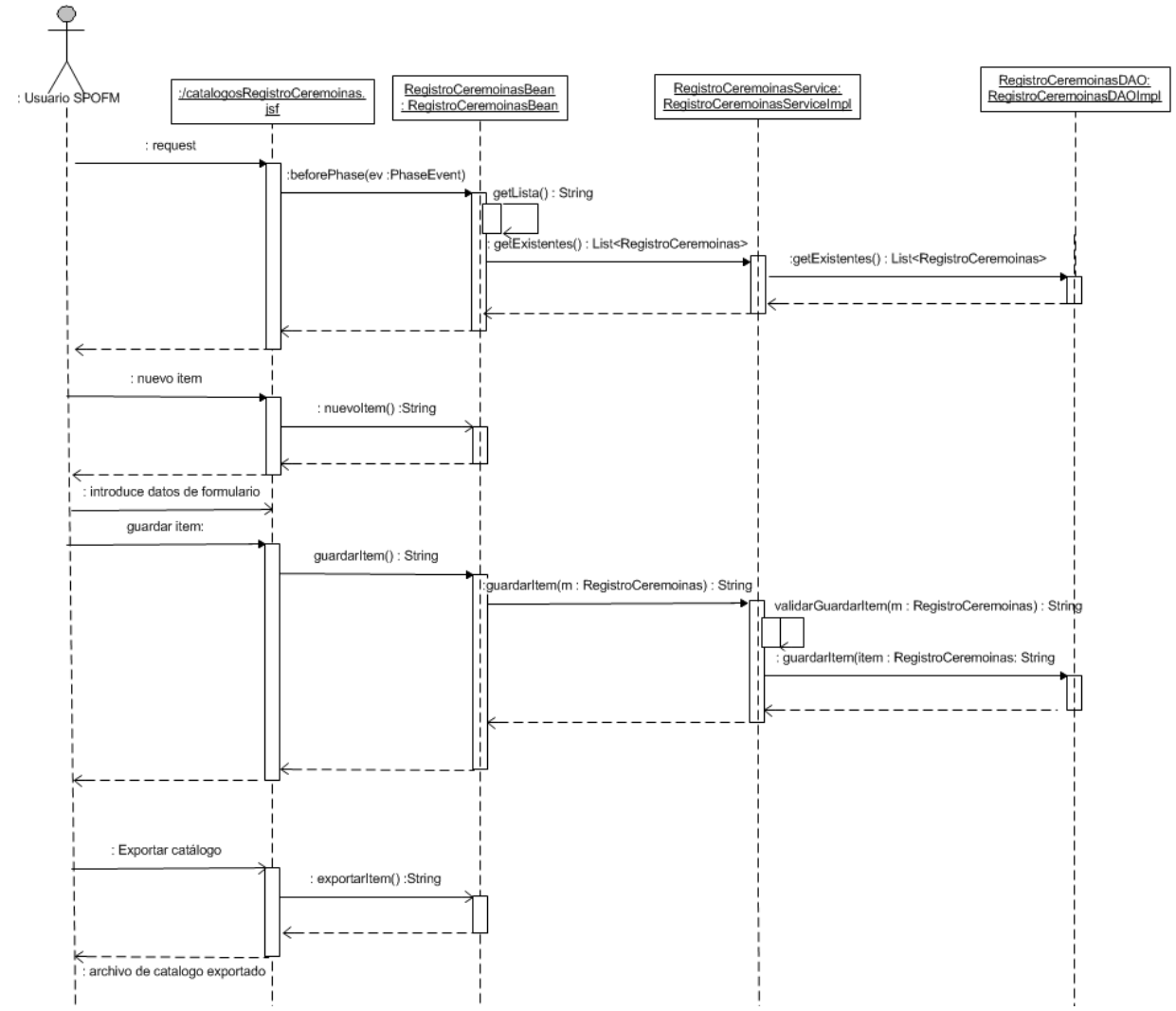

mx.edu.ofm.spofm.vista.VistaService
ALL
List<T> getExistentes()
List<T> getExistentes(T t)
mx.edu.ofm.spofm.vista.VistaServiceImpl
ALL
long serialVersionUID
Logger logger
VistaDAO<T> dao
VistaServiceImpl(VistaDAO<T> dao)
VistaDAO<T> getDao()
void setDao(VistaDAO<T> dao)
List<T> getExistentes()
List<T> getExistentes(T t)
mx.edu.ofm.spofm.vista.VistaBean
ALL
long serialVersionUID
Logger logger
VistaService<T> service
T item
List<T> itemList
String roleVista
boolean edicion
VistaBean()
void inyectaParam()
String getFecha()
String getLista()
PhaseId getPhaseId()
void beforePhase(PhaseEvent ev)
void afterPhase(PhaseEvent arg0)
void rowSelectListener(SelectEvent event)
VistaService<T> getService()
void setService(VistaService<T> service)
T getItem()
void setItem(T item)
List<T> getItemList()
void setItemList(List<T> itemList)
String getRoleVista()
String setRoleVista()
mx.edu.ofm.spofm.vista.VistaDAO
ALL
List<T> getExistentes(String vista)
List<T> getExistentes(String vista, Object param)
List<T> getExistentes()
List<T> getExistentes(T t)
mx.edu.ofm.spofm.vista.VistaDAOImp
ALL
long serialVersionUID
Logger logger
List<T> getExistentes(String query)
List<T> getExistentes(String query, Object param)
List<T> getExistentes()
List<T> getExistentes(T t)
mx.edu.ofm.spofm.vista.VistaPojo
ALL
long serialVersionUID
String descripcion
String nombre
Integer id
String getDescripcion()
void setDescripcion(String descripcion)
Integer getId()
void setId(Integer id)
String getNombre()
void setNombre(String nombre)

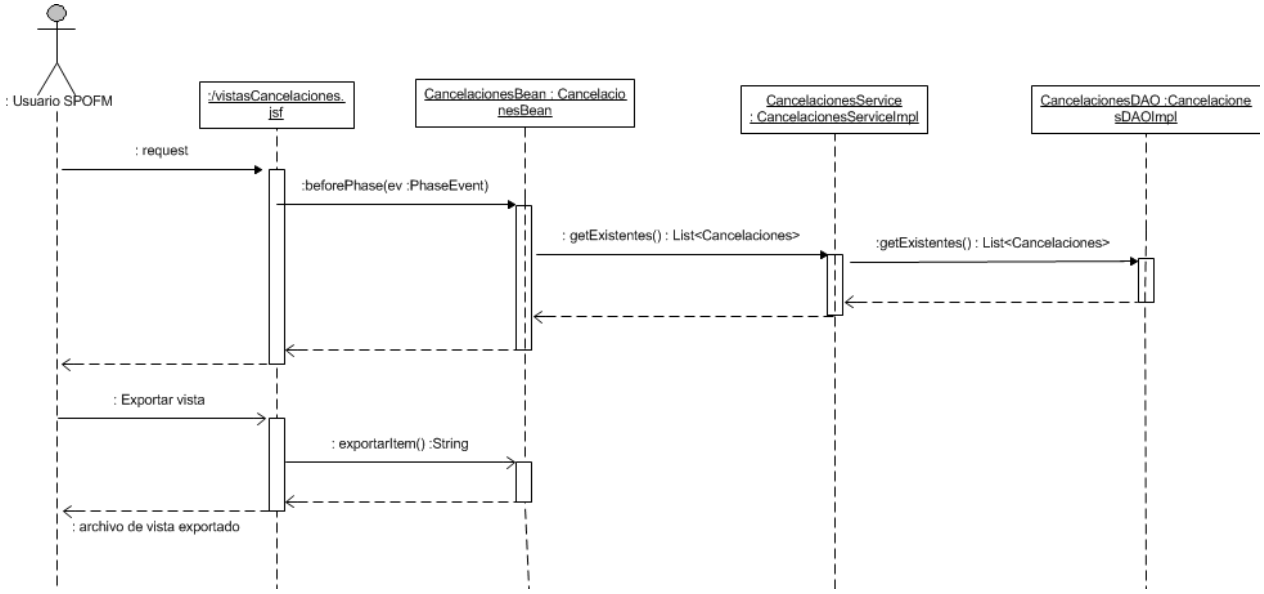
Diagramas de secuencia.

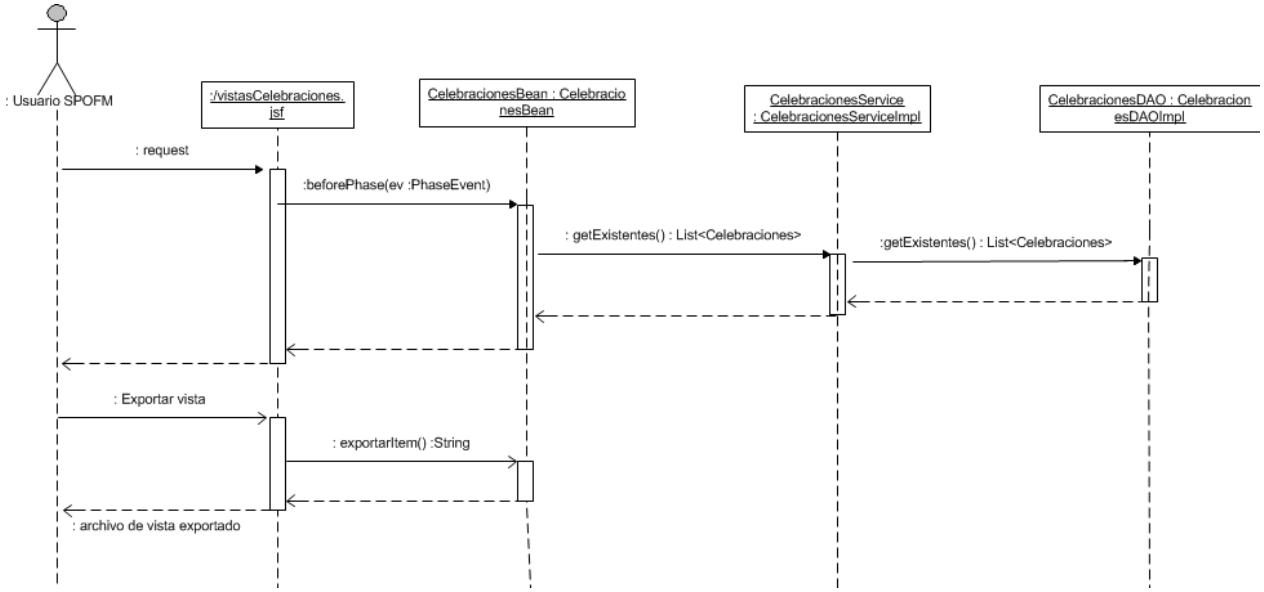


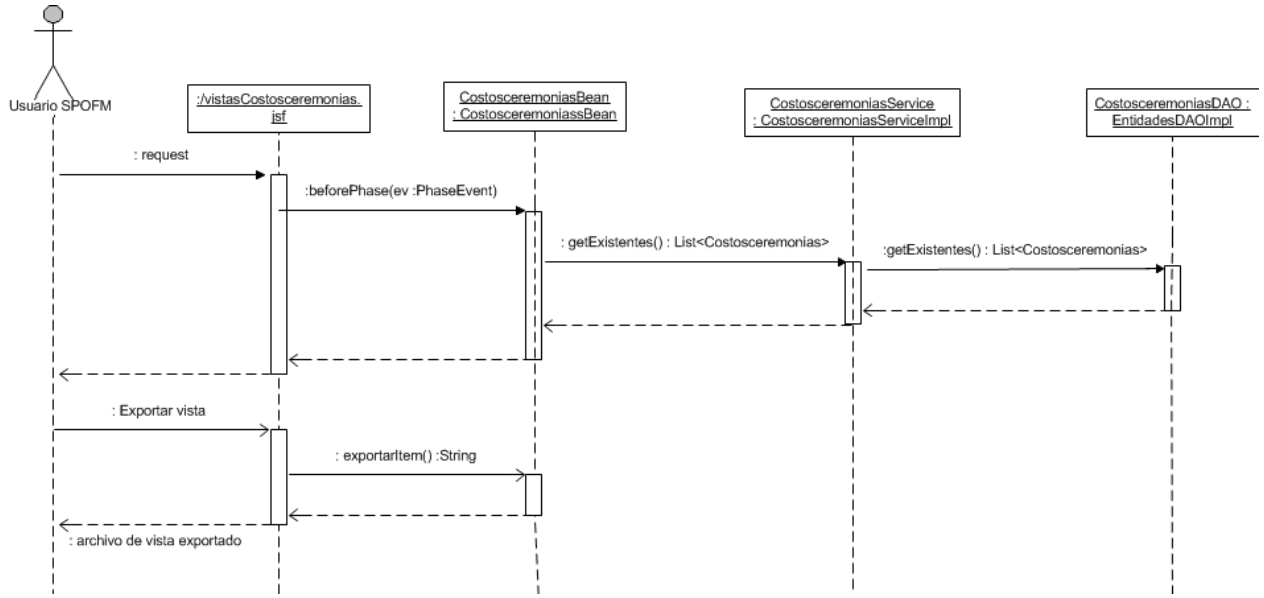

: Usuario SPOFM
:/catalogos/Usuario.jsf
UsuarioBean : UsuarioBean
UsuarioService : UsuarioServiceImpl
UsuarioDAO :UsuarioDAOImpl
: request
:beforePhase(ev :PhaseEvent)
getLista() : String
: getExistentes() : List<Usuario>
:getExistentes() : List<Usuario>
: nuevo item
: nuevoItem() :String
: introduce datos de formulario
guardar item:
guardarItem() : String
:guardarItem(m : Usuario) : String
validarGuardarItem(m : Usuario) : String
: guardarItem(item : Usuario: String
: Exportar catálogo
: exportarItem() :String
: archivo de catalogo exportado

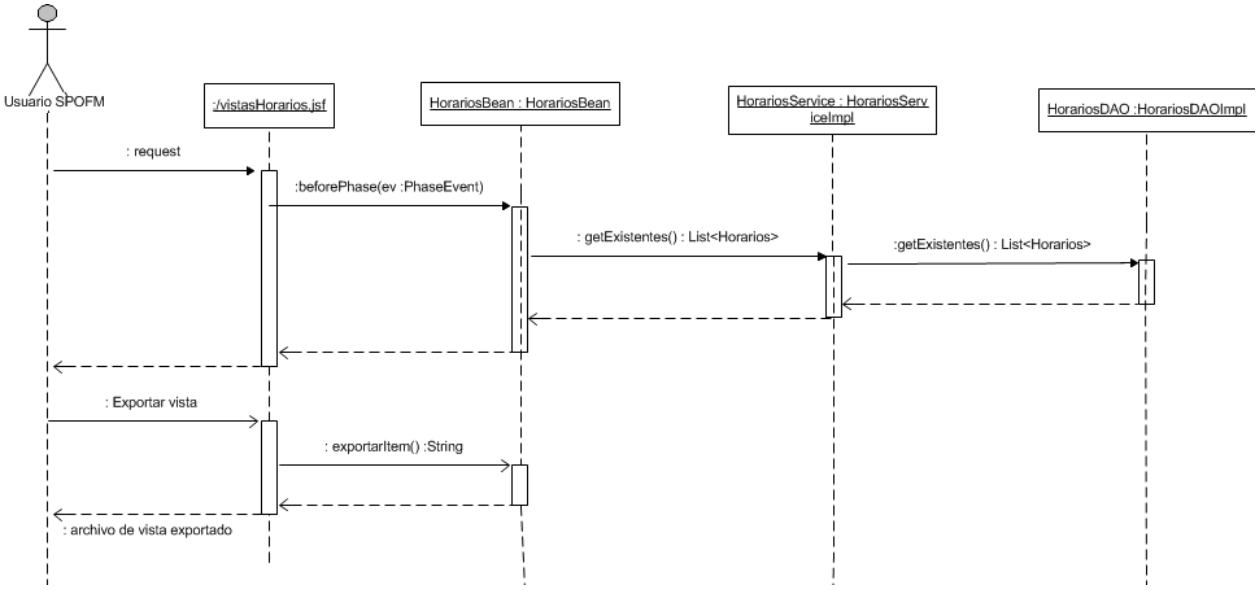

: Usuario SPOFM
:/catalogos/Perfiles.jsf
PerfilesBean : PerfilesBean
PerfilesService : PerfilesServiceImpl
PerfilesDAO :PerfilesDAOImpl
: request
:beforePhase(ev :PhaseEvent)
getLista() : String
: getExistentes() : List<Perfiles>
:getExistentes() : List<Perfiles>
: nuevo item
: nuevoItem() :String
: introduce datos de formulario
guardar item:
guardarItem() : String
:guardarItem(m : Perfiles) : String
validarGuardarItem(m : Perfiles) : String
: guardarItem(item : Perfiles: String
: Exportar catálogo
: exportarItem() :String
: archivo de catalogo exportado


: Usuario SPOFM
:/catalogosRegistroCeremoinas.jsf
RegistroCeremoinasBean : RegistroCeremoinasBean
RegistroCeremoinasService: RegistroCeremoinasServiceImpl
RegistroCeremoinasDAO: RegistroCeremoinasDAOImpl
: request
:beforePhase(ev :PhaseEvent)
getLista() : String
getExistentes() : List<RegistroCeremoinas>
:getExistentes() : List<RegistroCeremoinas>
: nuevo item
: nuevoItem() :String
: introduce datos de formulario
guardar item:
guardarItem() : String
:guardarItem(m : RegistroCeremoinas) : String
validarGuardarItem(m : RegistroCeremoinas) : String
: guardarItem(item : RegistroCeremoinas: String
: Exportar catálogo
: exportarItem() :String
: archivo de catalogo exportado


: Usuario SPOFM
:/vistasCancelaciones.jsf
CancelacionesBean : CancelacionesBean
CancelacionesService : CancelacionesServiceImpl
CancelacionesDAO :CancelacionesDAOImpl
: request
:beforePhase(ev :PhaseEvent)
: getExistentes() : List<Cancelaciones>
:getExistentes() : List<Cancelaciones>
: Exportar vista
: exportarItem() :String
: archivo de vista exportado


: Usuario SPOFM
:/vistasCelebraciones.jsf
CelebracionesBean : CelebracionesBean
CelebracionesService : CelebracionesServiceImpl
CelebracionesDAO : CelebracionesDAOImpl
: request
:beforePhase(ev :PhaseEvent)
: getExistentes() : List<Celebraciones>
:getExistentes() : List<Celebraciones>
: Exportar vista
: exportarItem() :String
: archivo de vista exportado


Usuario SPOFM
:/vistasCostosceremonias.jsf
CostosceremoniasBean : CostosceremoniassBean
CostosceremoniasService : CostosceremoniasServiceImpl
CostosceremoniasDAO : EntidadesDAOImpl
: request
:beforePhase(ev :PhaseEvent)
: getExistentes() : List<Costosceremonias>
:getExistentes() : List<Costosceremonias>
: Exportar vista
: exportarItem() :String
: archivo de vista exportado



F

Perfiles
Perfil.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Perfiles Seguridad
 * fecha creación: 17/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.perfiles;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;

import mx.edu.ofm.spofm.catalogo.CatalogoPOJO;
import mx.edu.ofm.spofm.catalogo.usuario.Permiso;

/**
 * Class Perfil.
 */
public class Perfil extends CatalogoPOJO implements Serializable {

	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	  * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = 7629140185073972807L;


	List<Permiso> permisos;

	/**
```

```
	 * Instancia un nuevo perfil.
	 */
	public Perfil() {
		super();
		permisos = new ArrayList<Permiso>();
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoPOJO#clon()
	 */
	public Perfil clon() {
		Perfil c = new Perfil();
		super.clon(c);

		List<Permiso> lp = new ArrayList<Permiso>();

		for (Permiso p: permisos){
			lp.add(p.clon());
		}
		c.setPermisos(lp);

		return c;
	}

	//este metodo puedes ser colocado en clase CatalogoPOJO
	/**
	 * Equals.
	 * @param obj
	 *            obj
	 * @return true, si es verdadero
	 */
	public boolean equals(Perfil obj) {
		boolean ret=true;
		if (!super.equals(obj)) {
			if(this.getId() != null && obj.getId() != null)
				ret = this.getId().equals(obj.getId());
			if(ret && (this.getNombre() != null && obj.getNombre() != null))
				ret = this.getNombre().equals(obj.getNombre());
			if(ret && (this.getDescripcion() != null	&&
obj.getDescripcion() != null))
				ret = this.getDescripcion().equals(obj.getDescripcion());
			if(ret && this.getActivo() != null && this.getActivo() != null)
				ret = this.getActivo().equals(obj.getActivo());
		}
		return ret;
	}

	/**
	 * Obtiene permisos.
	 * @return permisos
	 */
	public List<Permiso> getPermisos() {
		return permisos;
	}

	/**
	 * Asigna permisos.
	 * @param permisos
	 *            nuevo permisos
	 */
	public void setPermisos(List<Permiso> permisos) {
		this.permisos = permisos;
	}
```

```
}
```

PerfilBean.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Perfiles Seguridad
 * fecha creación: 17/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.perfiles;

import java.util.Date;
import java.util.List;

import javax.faces.event.PhaseEvent;
import javax.faces.event.PhaseId;
import javax.faces.event.PhaseListener;

import mx.edu.ofm.spofm.catalogo.CatalogoBean;
import mx.edu.ofm.spofm.catalogo.ExporterFunc;
import mx.edu.ofm.spofm.util.Constantes;

import org.apache.log4j.Logger;
import org.icefaces.ace.event.SelectEvent;

/**
 * Class PerfilBean.
 */
public class PerfilBean extends ExporterFunc implements PhaseListener, CatalogoBean {

	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	  * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = 8950137094755740662L;
	protected static final Logger logger = Logger
			.getLogger(PerfilBean.class);

	private PerfilService service;
	private Perfil item;
	private Perfil itemBackUp;
	private List<Perfil> itemList;

	private String roleAdmin = "ROLE_CAT_PERFILES_ADMINISTRAR";
	private String roleConsulta = "ROLE_CAT_PERFILES_CONSULTAR";

	private boolean edicion;
	//private boolean nuevo;

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoBean#nuevoItem()
	 */
	@Override
	public String nuevoItem() {
		item = new Perfil();
		itemBackUp = new Perfil();
		item.setPermisos(service.getPermisos());
		itemBackUp.setPermisos(service.getPermisos());
		edicion = true;
		Constantes.refresh();
```

```
		return "";
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoBean#guardarItem()
	 */
	@Override
	public String guardarItem() {
		// guardar el objeto
		String msg = service.guardarItem(item);
		if (msg.equals("")) {
			Constantes.mostrarMensajeInfo("general.guardado");
		} else {
			Constantes.mostrarMensajeError(msg);
		}
		// actualizar la lista
		getLista();
		if ("".equals(msg)) {
			itemBackUp = item.clon();
		}
		return msg;
	}

	/**
	 * Borrar item.
	 * @return string
	 */
	public String borrarItem(){
		String msg = service.borrarItem(item);
		getLista();
		if ("".equals(msg)) {
			itemBackUp = item.clon();
		}
		return msg;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoBean#restaurarItem()
	 */
	@Override
	public String restaurarItem() {
		item = itemBackUp.clon();
		Constantes.refresh();
		return null;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoBean#getLista()
	 */
	@Override
	public String getLista() {
		String msg = "";
		try {
			itemList = service.getExistentes(); // obtener lista
		} catch (Exception e) {
			msg = e.getMessage();
			logger.error(e.getMessage()); // hacer log de la excepcion
		}
		return msg;
	}

	/**
	 * Obtiene fecha.
```

```
	 * @return fecha
	 */
	public String getFecha() {
		return new Date().toString();
	}

	/* (non-Javadoc)
	 * @see javax.faces.event.PhaseListener#afterPhase(javax.faces.event.PhaseEvent)
	 */
	@Override
	public void afterPhase(PhaseEvent arg0) {
		// TODO Auto-generated method stub

	}

	/* (non-Javadoc)
	 * @see
javax.faces.event.PhaseListener#beforePhase(javax.faces.event.PhaseEvent)
	 */
	@Override
	public void beforePhase(PhaseEvent ev) {
		if (PhaseId.RENDER_RESPONSE == ev.getPhaseId()) {
			getLista();
		}
	}

	/* (non-Javadoc)
	 * @see javax.faces.event.PhaseListener#getPhaseId()
	 */
	@Override
	public PhaseId getPhaseId() {
		// TODO Auto-generated method stub
		return null;
	}

	/**
	 * Row select listener.
	 * @param event
	 *            event
	 */
	public void rowSelectListener(SelectEvent event) {
		item = ((Perfil) event.getObject()).clon();
		itemBackUp = item.clon();
		edicion = true;
		Constantes.refresh();
	}

	/**
	 * Obtiene service.
	 * @return service
	 */
	public PerfilService getService() {
		return service;
	}

	/**
	 * Asigna service.
	 * @param service
	 *            nuevo service
	 */
	public void setService(PerfilService service) {
		this.service = service;
	}
```

```
/**
 * Obtiene item.
 * @return item
 */
public Perfil getItem() {
	return item;
}

/**
 * Asigna item.
 * @param item
 *            nuevo item
 */
public void setItem(Perfil item) {
	this.item = item;
}

/**
 * Obtiene item back up.
 * @return item back up
 */
public Perfil getItemBackUp() {
	return itemBackUp;
}

/**
 * Asigna item back up.
 * @param itemBackUp
 *            nuevo item back up
 */
public void setItemBackUp(Perfil itemBackUp) {
	this.itemBackUp = itemBackUp;
}

/**
 * Obtiene item list.
 * @return item list
 */
public List<Perfil> getItemList() {
	return itemList;
}

/**
 * Asigna item list.
 * @param itemList
 *            nuevo item list
 */
public void setItemList(List<Perfil> itemList) {
	this.itemList = itemList;
}

/**
 * Obtiene role admin.
 * @return role admin
 */
public String getRoleAdmin() {
	return roleAdmin;
}

/**
 * Obtiene role consulta.
 * @return role consulta
```

```
	 */
	public String getRoleConsulta() {
		return roleConsulta;
	}

	/**
	 * Verifica si edicion.
	 * @return true, si es edicion
	 */
	public boolean isEdicion() {
		return edicion;
	}

	/**
	 * Asigna edicion.
	 * @param edicion
	 *            nuevo edicion
	 */
	public void setEdicion(boolean edicion) {
		this.edicion = edicion;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.ExporterFunc#getTituloReporte()
	 */
	@Override
	public String getTituloReporte() {
		// TODO Auto-generated method stub
		return "catalogo.perfil.titulo";
	}

}
```

PerfilDAO.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Perfiles Seguridad
 * fecha creación: 17/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.perfiles;

import java.util.List;
import mx.edu.ofm.spofm.catalogo.usuario.GrupoPagina;
import mx.edu.ofm.spofm.catalogo.usuario.Permiso;

/**
 * Interface PerfilDAO.
 */
public interface PerfilDAO {

	/**
	 * Guardar item.
	 * @param item
	 *            item
	 * @return string
	 */
	public String guardarItem(Perfil item);

	/**
```

```
         * Obtiene existentes.
         * @return existentes
         */
        public List<Perfil> getExistentes();

        /**
         * Obtiene existentes activos.
         * @return existentes activos
         */
        public List<Perfil> getExistentesActivos();

        /**
         * Validar repetido.
         * @param item
         *            item
         * @return int
         */
        public int validarRepetido(Perfil item);

        /**
         * Obtiene permisos.
         * @return permisos
         */
        public List<Permiso> getPermisos();

        /**
         * Obtiene modulos.
         * @return modulos
         */
        public List<GrupoPagina> getModulos();
}
```

PerfilDAOImpl.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Perfiles Seguridad
 * fecha creación: 17/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.perfiles;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import mx.edu.ofm.spofm.catalogo.usuario.GrupoPagina;
import mx.edu.ofm.spofm.catalogo.usuario.Pagina;
import mx.edu.ofm.spofm.catalogo.usuario.Permiso;
import mx.edu.ofm.spofm.catalogo.usuario.Usuario;

import org.apache.log4j.Logger;
import org.springframework.dao.DataAccessException;
import org.springframework.orm.ibatis.SqlMapClientTemplate;
import org.springframework.orm.ibatis.support.SqlMapClientDaoSupport;

/**
 * Class PerfilDAOImpl.
 */
```

```
public class PerfilDAOImpl extends SqlMapClientDaoSupport implements
		PerfilDAO, Serializable {

	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	  * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = -8199657358678030774L;
	protected static final Logger logger = Logger.getLogger(PerfilDAOImpl.class);

	/* (non-Javadoc)
	 * @see
mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAO#guardarItem(mx.edu.ofm.spofm.catalogo.perf
iles.Perfil)
	 */
	@Override
	public String guardarItem(Perfil item) {
		String msg="";
		try{
			SqlMapClientTemplate template = getSqlMapClientTemplate();
			if (item.getId()>0){
				template.insert("updateCatPerfil", item);
			}else{

	item.setId((Integer)template.queryForObject("getCatPerfilId"));
				template.insert("insertCatPerfil", item);
			}
			//guardar/eliminar permisos
			//eliminar todos los permisos
			template.delete("deletePerfilesFunciones", item.getId());

			//agregar los seleccionados
			Map<String, Integer>map = new HashMap<String, Integer>();
			for (Permiso p: item.getPermisos()){
				if (p.isAsignado()){
					map.clear();
					map.put("idPerfil", item.getId());
					map.put("idFuncion", p.getFuncion().getId());
					template.insert("insertPerfilFuncion", map);
				}
			}

		}catch(Exception e){
			msg = e.getMessage();
			logger.error(e.getMessage());
		}

		return msg;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAO#getExistentes()
	 */
	@Override
	public List<Perfil> getExistentes() {
		List<Perfil> lista = null;
		try{
			SqlMapClientTemplate template = getSqlMapClientTemplate();
			lista = template.queryForList("selectCatPerfiles");
			if (lista!=null){
				for (Perfil u:lista){

	u.setPermisos(template.queryForList("selectPermisosByIdPerfil", u.getId()));
```

```
                        }
                }
        }catch(Exception e){
                lista = new ArrayList<Perfil>();
                logger.error(e.getMessage());
        }
        return lista;
}

/* (non-Javadoc)
 * @see mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAO#getExistentesActivos()
 */
@Override
public List<Perfil> getExistentesActivos() {
        List<Perfil> lista = null;
        try{
                SqlMapClientTemplate template = getSqlMapClientTemplate();
                lista = template.queryForList("selectCatPerfilesActivos");
                if (lista!=null){
                        for (Perfil u:lista){

u.setPermisos(template.queryForList("selectPermisosActivosByIdPerfil",
u.getId()));
                        }
                }
        }catch(Exception e){
                lista = new ArrayList<Perfil>();
                logger.error(e.getMessage());
        }
        return lista;
}

/* (non-Javadoc)
 * @see
mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAO#validarRepetido(mx.edu.ofm.spofm.catalogo.
perfiles.Perfil)
 */
public int validarRepetido(Perfil item) {
        int rep = 1;
        try{

                SqlMapClientTemplate template = getSqlMapClientTemplate();
                rep =
(Integer)template.queryForObject("validarRepetidoCatPerfiles", item);

        }catch (DataAccessException dae){
                System.out.println(dae.getCause());
                System.out.println(dae.getMostSpecificCause());
        }
        catch(Exception e){
                rep = 1;
                logger.error(e.getMessage());
        }
        return rep;
}

/* (non-Javadoc)
 * @see mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAO#getPermisos()
 */
public List<Permiso> getPermisos() {
        List<Permiso> lista = null;
        try{
                SqlMapClientTemplate template = getSqlMapClientTemplate();
```

```
                lista = template.queryForList("selectPermisosByIdPerfil", -1);
            }catch(Exception e){
                lista = new ArrayList<Permiso>();
                logger.error(e.getMessage());
            }
            return lista;
        }

        /* (non-Javadoc)
         * @see mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAO#getModulos()
         */
        public List<GrupoPagina> getModulos() {
            List<GrupoPagina> lista = null;
            try{
                SqlMapClientTemplate template = getSqlMapClientTemplate();
                lista = template.queryForList("selectCatModulos");
                if (lista!=null){
                    for (GrupoPagina m:lista){

        m.setPaginas(template.queryForList("selectPaginasByIdModulo", m.getId()));
                            for (Pagina p : m.getPaginas()){

        p.setFunciones(template.queryForList("selectPerfilesByIdPagina", p.getId()));
                            }
                        }
                }
            }catch(Exception e){
                lista = new ArrayList<GrupoPagina>();
                logger.error(e.getMessage());
            }

            return lista;
        }

}
```

PerfilService.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Perfiles Seguridad
 * fecha creación: 17/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.perfiles;

import java.util.List;
import mx.edu.ofm.spofm.catalogo.usuario.GrupoPagina;
import mx.edu.ofm.spofm.catalogo.usuario.Permiso;

/**
 * Interface PerfilService.
 */
public interface PerfilService {

        /**
         * Guardar item.
         * @param m
         *            m
         * @return string
         */
```

```
	public String guardarItem(Perfil m);

	/**
	 * Borrar item.
	 * @param item
	 *            item
	 * @return string
	 */
	public String borrarItem(Perfil item);

	/**
	 * Obtiene existentes.
	 * @return existentes
	 */
	public List<Perfil> getExistentes();

	/**
	 * Obtiene existentes activos.
	 * @return existentes activos
	 */
	public List<Perfil> getExistentesActivos();

	/**
	 * Obtiene permisos.
	 * @return permisos
	 */
	public List<Permiso> getPermisos();

	/**
	 * Obtiene modulos.
	 * @return modulos
	 */
	public List<GrupoPagina> getModulos();
}
```

PerfilServiceImpl.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Perfiles Seguridad
 * fecha creación: 17/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.perfiles;

import java.io.Serializable;
import java.util.List;
import mx.edu.ofm.spofm.catalogo.area.Area;
import mx.edu.ofm.spofm.catalogo.usuario.GrupoPagina;
import mx.edu.ofm.spofm.catalogo.usuario.Permiso;
import mx.edu.ofm.spofm.util.Constantes;

import org.apache.log4j.Logger;
import org.springframework.dao.DataAccessException;
import org.springframework.orm.ibatis.SqlMapClientTemplate;
import org.springframework.orm.ibatis.support.SqlMapClientDaoSupport;

/**
 * Class PerfilServiceImpl.
 */
public class PerfilServiceImpl  implements PerfilService, Serializable {
```

```
	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	  * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = -2202021879733507317L;
	protected static final Logger logger = Logger
			.getLogger(PerfilServiceImpl.class);

	private PerfilDAO dao;

	/**
	 * Instancia un nuevo perfil service impl.
	 * @param dao
	 *            dao
	 */
	public PerfilServiceImpl(PerfilDAO dao) {
		this.dao = dao;
	}

	/**
	 * Validar guardar item.
	 * @param item
	 *            item
	 * @return string
	 */
	private String validarGuardarItem(Perfil item) {

		String msg="";
		if (dao.validarRepetido(item)>0){
			msg = Constantes.getMensaje("general.guardado.error.repetido");
		}
		return msg;

	}

	/* (non-Javadoc)
	 * @see
mx.edu.ofm.spofm.catalogo.perfiles.PerfilService#guardarItem(mx.edu.ofm.spofm.catalogo.
perfiles.Perfil)
	 */
	@Override
	public String guardarItem(Perfil item) {
		String msg = validarGuardarItem(item);
		if (msg.equals("")) {
			msg = dao.guardarItem(item);
		}
		return msg;
	}

	 /* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.perfiles.PerfilService#getPermisos()
	 */
	public List< Permiso> getPermisos() {
			return dao.getPermisos();
		}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.perfiles.PerfilService#getExistentes()
	 */
	@Override
	public List<Perfil> getExistentes() {
		return dao.getExistentes();
```

```
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.perfiles.PerfilService#getExistentesActivos()
	 */
	public List<Perfil> getExistentesActivos() {
		return dao.getExistentesActivos();
	}

	/**
	 * Obtiene estado dao.
	 * @return estado dao
	 */
	public PerfilDAO getEstadoDAO() {
		return dao;
	}

	/**
	 * Asigna estado dao.
	 * @param estadoDAO
	 *            nuevo estado dao
	 */
	public void setEstadoDAO(PerfilDAO estadoDAO) {
		this.dao = estadoDAO;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.perfiles.PerfilService#getModulos()
	 */
	public List<GrupoPagina> getModulos() {
		return dao.getModulos();
	}

	/* (non-Javadoc)
	 * @see
mx.edu.ofm.spofm.catalogo.perfiles.PerfilService#borrarItem(mx.edu.ofm.spofm.catalogo.p
erfiles.Perfil)
	 */
	@Override
	public String borrarItem(Perfil item) {
		item.setActivo(false);
		return this.guardarItem(item);
	}

}
```

perfilDAO.xml

```
<!DOCTYPE sqlMap
PUBLIC "-//iBATIS.com//DTD SQL Map 2.0//EN"
"http://ibatis.apache.org/dtd/sql-map-2.dtd">

<sqlMap>
    <typeAlias alias="catPerfil" type="mx.edu.ofm.spofm.catalogo.perfiles.Perfil"/>

      <resultMap class="catPerfil" id="catPerfilMap">
		<result property = "id" column = "IDPERFIL" jdbcType="INTEGER"
javaType="java.lang.Integer"/>
          <result property = "nombre" column = "NOMPERFIL"/>
          <result property = "descripcion" column = "DESCRIPCION"/>
          <result property = "activo" column = "ACTIVO"/>
    </resultMap>
```

```
	<statement id="selectCatPerfiles" resultMap="catPerfilMap"
parameterClass="catPerfil">
		select * from perfiles at
		<dynamic prepend="WHERE">
			<isNotNull prepend="AND" property="nombre">
				translate(upper(NOMPERFIL), 'Ã Ã‰Ã Ã“Ã¡', 'AEIOU')  =
translate(upper(#nombre#), 'Ã Ã‰Ã Ã“Ã¡', 'AEIOU')
			</isNotNull>
		</dynamic>
		order by at.NOMPERFIL
	</statement>

	<statement id="selectCatPerfilesActivos" resultMap="catPerfilMap">
		select * from perfiles at
		where activo = 1
		order by at.NOMPERFIL
	</statement>

	<!--
		algunas tablas no necesitan secuencia para los id porque son pequeÃ±as
		en otras mas grandes si se usara
	 -->
	<select id="getCatPerfilId" resultClass="Integer">
		select nvl(max(idperfil), 0) + 1 from perfiles
	</select>

    <insert id="insertCatPerfil" parameterClass="catPerfil">
    INSERT INTO perfiles(
		IDPERFIL,
		NOMPERFIL,
		DESCRIPCION,
		ACTIVO)
	VALUES(
		#id#,
		#nombre#,
		#descripcion#,
		#activo#
	)
    </insert>

    <update id="updateCatPerfil" parameterClass="catPerfil">
		UPDATE
			perfiles
		SET
			NOMperfil=#nombre#,
			DESCRIPCION=#descripcion#,
			ACTIVO=#activo#
		WHERE
			IDperfil = #id#

	</update>

	<select id="validarRepetidoCatPerfiles" resultClass="Integer"
parameterClass="catPerfil">
		select count(1) from perfiles
		where
			rownum = 1 and
			<dynamic>
				<isNotNull property="id">
					<isNotEmpty property="id">
						<isNotEqual property="id" compareValue="0">
							<![CDATA[IDPERFIL <> #id# and]]>
						</isNotEqual>
```

```
                                </isNotEmpty>
                        </isNotNull>
                </dynamic>
                (
                        upper(NOMPERFIL) = upper(#nombre#)
                )
        </select>

        <select id="selectPermisosByIdPerfil" resultMap="catPermisoMap"
parameterClass="Integer">
                select GP.IDGRUPOPAGINA, GP.NOMGRUPOPAGINA, PA.IDPAGINA,
pa.nompagina,FU.IDFUNCION, FU.NOMFUNCION,
                nvl(
                    (select 1 from PERFILESFUNCIONES up where UP.IDFUNCION=FU.IDFUNCION
and UP.IDPERFIL = #value#)
                , 0) asignado
                from GRUPOSPAGINAS GP,
                PAGINAS pa,
                FUNCIONES fu
                where
                GP.IDGRUPOPAGINA = PA.IDGRUPOPAGINA
                and PA.IDPAGINA = FU.IDPAGINA
                <!-- order by NOMGRUPOPAGINA, nompagina, NOMFUNCION  -->
        </select>

        <select id="selectPermisosActivosByIdPerfil" resultMap="catPermisoMap"
parameterClass="Integer">
                select GP.IDGRUPOPAGINA, GP.NOMGRUPOPAGINA, PA.IDPAGINA,
pa.nompagina,FU.IDFUNCION, FU.NOMFUNCION,
                1 asignado
                from GRUPOSPAGINAS GP,
                PAGINAS pa,
                FUNCIONES fu,
                PERFILESFUNCIONES PF
                where
                GP.IDGRUPOPAGINA = PA.IDGRUPOPAGINA
                and PA.IDPAGINA = FU.IDPAGINA
                and PF.IDFUNCION = FU.IDFUNCION
                and PF.IDPERFIL = #value#

                <!-- order by NOMGRUPOPAGINA, nompagina, NOMFUNCION  -->
        </select>

        <delete id="deletePerfilesFunciones" parameterClass="Integer">
                delete PERFILESFUNCIONES where IDPERFIL = #value#
        </delete>

        <parameterMap class="java.util.HashMap" id="perfilFuncionMap">
                <parameter property="idPerfil" javaType="Integer" jdbcType="INTEGER"/>
                <parameter property="idFuncion" javaType="Integer" jdbcType="INTEGER"/>
    </parameterMap>

        <insert id="insertPerfilFuncion" parameterMap="perfilFuncionMap">
                insert into PERFILESFUNCIONES (
                        IDPERFIL,
                        IDFUNCION)
                values (
                        ?,
                        ?
                )
        </insert>

</sqlMap>
```

Usuario
Funcion.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.io.Serializable;
import mx.edu.ofm.spofm.catalogo.CatalogoPOJO;

/**
 * Class Funcion.
 */
public class Funcion extends CatalogoPOJO implements Serializable{
	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	  * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = 28062568773454761L;

	//poner las longitudes minimas (si se necesitan) y maximas que correspondan con
la db

	/**
	 * Instancia un nuevo funcion.
	 */
	public Funcion() {
		super();
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoPOJO#clon()
	 */
	public Funcion clon(){
		Funcion c= new Funcion();

		//clonar los valores de la clase padre
		super.clon(c);

		return c;
	}

}
```

GrupoPagina.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;
```

```
import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;
import mx.edu.ofm.spofm.catalogo.CatalogoPOJO;

/**
 * Class GrupoPagina.
 */
public class GrupoPagina extends CatalogoPOJO implements Serializable{
	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	  * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = 28062568773454761L;

	List<Pagina> paginas;

	//poner las longitudes minimas (si se necesitan) y maximas que correspondan con
la db

	/**
	 * Instancia un nuevo grupo pagina.
	 */
	public GrupoPagina() {
		super();
		paginas = new ArrayList<Pagina>();
	}

	/**
	 * Obtiene paginas.
	 * @return paginas
	 */
	public List<Pagina> getPaginas() {
		return paginas;
	}

	/**
	 * Asigna paginas.
	 * @param paginas
	 *            nuevo paginas
	 */
	public void setPaginas(List<Pagina> paginas) {
		this.paginas = paginas;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoPOJO#clon()
	 */
	public GrupoPagina clon(){
		GrupoPagina c= new GrupoPagina();

		//clonar los valores de la clase padre
		super.clon(c);

		return c;
	}

}
```

Pagina.java

```
/*
```

```
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;
import mx.edu.ofm.spofm.catalogo.CatalogoPOJO;

/**
 * Class Pagina.
 */
public class Pagina extends CatalogoPOJO implements Serializable{
	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	 * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = 28062568773454761L;

	List<Funcion> funciones;

	//poner las longitudes minimas (si se necesitan) y maximas que correspondan con
la db

	/**
	 * Instancia un nuevo pagina.
	 */
	public Pagina() {
		super();
		funciones = new ArrayList<Funcion>();
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoPOJO#clon()
	 */
	public Pagina clon(){
		Pagina c= new Pagina();

		//clonar los valores de la clase padre
		super.clon(c);

		return c;
	}

	/**
	 * Obtiene funciones.
	 * @return funciones
	 */
	public List<Funcion> getFunciones() {
		return funciones;
	}


	/**
	 * Asigna funciones.
	 * @param funciones
	 *            nuevo funciones
	 */
```

```
	public void setFunciones(List<Funcion> funciones) {
		this.funciones = funciones;
	}

}
```

Permiso.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;
import mx.edu.ofm.spofm.catalogo.CatalogoPOJO;

/**
 * Class Permiso.
 */
public class Permiso implements Serializable{
	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	 * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = 28062568773454761L;

	private GrupoPagina grupoPagina;
	private Pagina pagina;
	private Funcion funcion;
	private boolean asignado;

	//poner las longitudes minimas (si se necesitan) y maximas que correspondan con
la db

	/**
	 * Instancia un nuevo permiso.
	 */
	public Permiso() {
		super();
		grupoPagina = new GrupoPagina();
		pagina = new Pagina();
		funcion = new Funcion();
		asignado = false;
	}

	/**
	 * Clon.
	 * @return permiso
	 */
	public Permiso clon(){
		Permiso c= new Permiso();

		c.setAsignado(asignado);
		c.setGrupoPagina(grupoPagina.clon());
		c.setPagina(pagina.clon());
		c.setFuncion(funcion.clon());
```

```
		return c;
}

	/**
	 * Obtiene pagina.
	 * @return pagina
	 */
	public Pagina getPagina() {
		return pagina;
	}

	/**
	 * Asigna pagina.
	 * @param pagina
	 *            nuevo pagina
	 */
	public void setPagina(Pagina pagina) {
		this.pagina = pagina;
	}

	/**
	 * Verifica si asignado.
	 * @return true, si es asignado
	 */
	public boolean isAsignado() {
		return asignado;
	}

	/**
	 * Asigna asignado.
	 * @param asignado
	 *            nuevo asignado
	 */
	public void setAsignado(boolean asignado) {
		this.asignado = asignado;
	}

	/**
	 * Obtiene grupo pagina.
	 * @return grupo pagina
	 */
	public GrupoPagina getGrupoPagina() {
		return grupoPagina;
	}
```

Usuario.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ----------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;
import mx.edu.ofm.spofm.catalogo.CatalogoPOJO;
import mx.edu.ofm.spofm.catalogo.area.Area;
```

```
/**
 * Class Usuario.
 */
public class Usuario extends CatalogoPOJO implements Serializable{
	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	 * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = 28062568773454761L;

	String apellidoP;
	String apellidoM;
	String usuario;
	String pwd;

	List<Permiso> permisos;
	private Area area;
	//poner las longitudes minimas (si se necesitan) y maximas que correspondan con
la db

	int apellidoPLenMax;
	int apellidoMLenMin;
	int apellidoMLenMax;
	int pwdLenMin;
	int pwdLenMax;
	int usuarioLenMin;
	int usuarioLenMax;

	/**
	 * Instancia un nuevo usuario.
	 */
	public Usuario() {
		super();
		apellidoP="";
		apellidoM="";
		pwd="";

		apellidoPLenMin = 0;
		apellidoPLenMax = 255 ;
		apellidoMLenMin = 0;
		apellidoMLenMax = 255 ;
		pwdLenMin = 0;
		pwdLenMax = 255 ;
		usuarioLenMin = 0;
		usuarioLenMax = 255 ;

		permisos = new ArrayList<Permiso>();
		area = new Area();
	}

	/**
	 * Obtiene permisos.
	 * @return permisos
	 */
	public List<Permiso> getPermisos() {
		return permisos;
	}

	/**
	 * Asigna permisos.
	 * @param permisos
	 *            nuevo permisos
```

```
 */
public void setPermisos(List<Permiso> permisos) {
	this.permisos = permisos;
}

/* (non-Javadoc)
 * @see mx.edu.ofm.spofm.catalogo.CatalogoPOJO#clon()
 */
public Usuario clon(){
	Usuario c= new Usuario();

	//clonar los valores de la clase padre
	super.clon(c);

	c.setApellidoP(this.apellidoP);
	c.setApellidoM(this.apellidoM);
	c.setPwd(this.pwd);
	c.setUsuario(this.usuario);
	c.setArea(area.clon());

	List<Permiso> lp = new ArrayList<Permiso>();

	for (Permiso p: permisos){
		lp.add(p.clon());
	}
	c.setPermisos(lp);
	return c;
}

/**
 * Obtiene usuario.
 * @return usuario
 */
public String getUsuario() {
	return usuario;
}

/**
 * Asigna usuario.
 * @param usuario
 *            nuevo usuario
 */
public void setUsuario(String usuario) {
	this.usuario = usuario;
}

/**
 * Obtiene usuario len min.
 * @return usuario len min
 */
public int getUsuarioLenMin() {
	return usuarioLenMin;
}

/**
 * Asigna usuario len min.
 * @param usuarioLenMin
 *            nuevo usuario len min
 */
public void setUsuarioLenMin(int usuarioLenMin) {
	this.usuarioLenMin = usuarioLenMin;
}
```

```
/**
 * Obtiene usuario len max.
 * @return usuario len max
 */
public int getUsuarioLenMax() {
	return usuarioLenMax;
}

/**
 * Asigna usuario len max.
 * @param usuarioLenMax
 *            nuevo usuario len max
 */
public void setUsuarioLenMax(int usuarioLenMax) {
	this.usuarioLenMax = usuarioLenMax;
}

int apellidoPLenMin;

/**
 * Obtiene pwd len min.
 * @return pwd len min
 */
public int getPwdLenMin() {
	return pwdLenMin;
}

/**
 * Asigna pwd len min.
 * @param pwdLenMin
 *            nuevo pwd len min
 */
public void setPwdLenMin(int pwdLenMin) {
	this.pwdLenMin = pwdLenMin;
}

/**
 * Obtiene pwd len max.
 * @return pwd len max
 */
public int getPwdLenMax() {
	return pwdLenMax;
}

/**
 * Asigna pwd len max.
 * @param pwdLenMax
 *            nuevo pwd len max
 */
public void setPwdLenMax(int pwdLenMax) {
	this.pwdLenMax = pwdLenMax;
}

/**
 * Obtiene apellido p.
 * @return apellido p
 */
public String getApellidoP() {
	return apellidoP;
}

/**
 * Asigna apellido p.
```

```
 * @param apellidoP
 *            nuevo apellido p
 */
public void setApellidoP(String apellidoP) {
	this.apellidoP = apellidoP;
}

/**
 * Obtiene apellido m.
 * @return apellido m
 */
public String getApellidoM() {
	return apellidoM;
}

/**
 * Asigna apellido m.
 * @param apellidoM
 *            nuevo apellido m
 */
public void setApellidoM(String apellidoM) {
	this.apellidoM = apellidoM;
}

/**
 * Obtiene pwd.
 * @return pwd
 */
public String getPwd() {
	return pwd;
}

/**
 * Asigna pwd.
 * @param pwd
 *            nuevo pwd
 */
public void setPwd(String pwd) {
	this.pwd = pwd;
}

/**
 * Obtiene apellido p len min.
 * @return apellido p len min
 */
public int getApellidoPLenMin() {
	return apellidoPLenMin;
}

/**
 * Asigna apellido p len min.
 * @param apellidoPLenMin
 *            nuevo apellido p len min
 */
public void setApellidoPLenMin(int apellidoPLenMin) {
	this.apellidoPLenMin = apellidoPLenMin;
}

/**
 * Obtiene apellido p len max.
 * @return apellido p len max
 */
public int getApellidoPLenMax() {
```

```
		return apellidoPLenMax;
	}

	/**
	 * Asigna apellido p len max.
	 * @param apellidoPLenMax
	 *            nuevo apellido p len max
	 */
	public void setApellidoPLenMax(int apellidoPLenMax) {
		this.apellidoPLenMax = apellidoPLenMax;
	}

	/**
	 * Obtiene apellido m len min.
	 * @return apellido m len min
	 */
	public int getApellidoMLenMin() {
		return apellidoMLenMin;
	}

	/**
	 * Asigna apellido m len min.
	 * @param apellidoMLenMin
	 *            nuevo apellido m len min
	 */
	public void setApellidoMLenMin(int apellidoMLenMin) {
		this.apellidoMLenMin = apellidoMLenMin;
	}

	/**
	 * Obtiene apellido m len max.
	 * @return apellido m len max
	 */
	public int getApellidoMLenMax() {
		return apellidoMLenMax;
	}

	/**
	 * Asigna apellido m len max.
	 * @param apellidoMLenMax
	 *            nuevo apellido m len max
	 */
	public void setApellidoMLenMax(int apellidoMLenMax) {
		this.apellidoMLenMax = apellidoMLenMax;
	}

	/**
	 * Obtiene area.
	 * @return area
	 */
	public Area getArea() {
		return area;
	}

	/**
	 * Asigna area.
	 * @param area
	 *            nuevo area
	 */
	public void setArea(Area area) {
		this.area = area;
	}
}
```

UsuarioBean.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.util.Date;
import java.util.ArrayList;
import java.util.List;

import javax.faces.context.FacesContext;
import javax.faces.event.PhaseEvent;
import javax.faces.event.PhaseId;
import javax.faces.event.PhaseListener;
import javax.faces.event.ValueChangeEvent;
import javax.faces.model.SelectItem;

import mx.edu.ofm.spofm.catalogo.CatalogoBean;
import mx.edu.ofm.spofm.catalogo.CatalogoService;
import mx.edu.ofm.spofm.catalogo.ExporterFunc;
import mx.edu.ofm.spofm.catalogo.direccion.Direccion;
import mx.edu.ofm.spofm.catalogo.perfiles.Perfil;
import mx.edu.ofm.spofm.catalogo.perfiles.PerfilService;
import mx.edu.ofm.spofm.util.Constantes;

import org.apache.log4j.Logger;
import org.icefaces.ace.event.SelectEvent;
import com.icesoft.faces.component.ext.RowSelectorEvent;

/**
 * Class UsuarioBean.
 */
public class UsuarioBean extends ExporterFunc implements PhaseListener, CatalogoBean {
	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	 * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = 3754240932218185724L;
	protected static final Logger logger = Logger.getLogger(UsuarioBean.class);
	private UsuarioService service;
	private CatalogoService catalogoService;
	private PerfilService perfilService;
	private Usuario item;
	private Usuario itemBackUp;
	private List<Usuario> itemList;
	private List<Perfil> perfilList;

	private List<GrupoPagina> moduloList;
	private List<SelectItem> comboArea;

	private String roleAdmin = "ROLE_CAT_USUARIOS_ADMINISTRAR";
	private String roleConsulta = "ROLE_CAT_USUARIOS_CONSULTAR";

	/**
	 * Instancia un nuevo usuario bean.
	 * @param service
```

```
 *            service
 * @param catalogoService
 *            catalogo service
 * @param perfilService
 *            perfil service
 */
public UsuarioBean(
            UsuarioService service,
            CatalogoService catalogoService,
            PerfilService perfilService) {
      super();
      this.service = service;
      this.catalogoService = catalogoService;
      this.perfilService = perfilService;

}

/**
 * Obtiene modulo list.
  * @return modulo list
 */
public List<GrupoPagina> getModuloList() {
      return moduloList;
}

/**
 * Asigna modulo list.
 * @param moduloList
 *            nuevo modulo list
 */
public void setModuloList(List<GrupoPagina> moduloList) {
      this.moduloList = moduloList;
}

boolean edicion;

/**
 * Instancia un nuevo usuario bean.
 */
public UsuarioBean() {
      item = new Usuario();
      itemList = new ArrayList<Usuario>();
      edicion = false;
}

/**
 * Obtiene fecha.
 * @return fecha
 */
public String getFecha(){
      return new Date().toString();
}

/* (non-Javadoc)
 * @see mx.edu.ofm.spofm.catalogo.CatalogoBean#getLista()
 */
public String getLista(){
      String msg = "";
      try{
            itemList=service.getExistentes(); //obtener lista
      }catch (Exception e){
            msg=e.getMessage();
            logger.error(e.getMessage()); //hacer log de la excepcion
```

```
		}
		return msg;
	}

	/**
	 * Obtiene lista modulos.
	 * @return lista modulos
	 */
	public String getListaModulos(){
		String msg = "";
		try{
			moduloList=service.getModulos(); //obtener lista
		}catch (Exception e){
			msg=e.getMessage();
			logger.error(e.getMessage()); //hacer log de la excepcion
		}
		return msg;
	}

	/* (non-Javadoc)
	 * @see javax.faces.event.PhaseListener#getPhaseId()
	 */
	public PhaseId getPhaseId() {
		return PhaseId.ANY_PHASE;
		}

	/*
	 * (non-Javadoc)
	 * @see
javax.faces.event.PhaseListener#beforePhase(javax.faces.event.PhaseEvent)
	 * se utiliza para inicializar la vista, existen varias fases
	 * en la fase render se hacen las consultas
	 */
	public void beforePhase(PhaseEvent ev){
		//obtener la lista inicial
		if (PhaseId.RENDER_RESPONSE==ev.getPhaseId()){
			getLista();
			comboArea = catalogoService.getAreasSelectItem();
		}
	}

	/*
	 * (non-Javadoc)
	 * @see javax.faces.event.PhaseListener#afterPhase(javax.faces.event.PhaseEvent)
	 * metodo que se ejecuta despues de cada fase
	 */
	public void afterPhase(PhaseEvent arg0) {
		// TODO Auto-generated method stub
	}

	/**
	 * Verifica si edicion.
	 * @return true, si es edicion
	 */
	public boolean isEdicion() {
		return edicion;
	}

	/**
	 * Asigna edicion.
	 * @param edicion
	 *            nuevo edicion
	 */
```

```
	public void setEdicion(boolean edicion) {
		this.edicion = edicion;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoBean#nuevoItem()
	 */
	@Override
	public String nuevoItem() {
		item = new Usuario();
		itemBackUp = new Usuario();
		item.setPermisos(service.getPermisos());
		itemBackUp.setPermisos(service.getPermisos());
		edicion = true;
		perfilList = perfilService.getExistentesActivos();

		for (Perfil p: perfilList){
			p.setActivo(false);
		}
		Constantes.refresh();
		return "";
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoBean#guardarItem()
	 */
	@Override
	public String guardarItem() {
		//guardar el objeto
		String msg = service.guardarItem(item);
		//actualizar la lista
		getLista();

		if ("".equals(msg)){
			itemBackUp=item.clon();
			Constantes.mostrarMensajeInfo("general.guardado");
		}else{
			Constantes.mostrarMensajeError(msg);
		}
		return msg;

	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.CatalogoBean#restaurarItem()
	 */
	@Override
	public String restaurarItem() {
		item=itemBackUp.clon();
		Constantes.refresh();
		return null;
	}

	/**
 * Row select listener.
 * @param event
 *            event
 */
public void rowSelectListener(SelectEvent event) {
		item=((Usuario) event.getObject()).clon();
	    itemBackUp=item.clon();
	    edicion=true;
```

```
            perfilList = perfilService.getExistentesActivos();

            //marcar los perfiles que el usuario tenga asignados
            for (Perfil perfil: perfilList){
                int asignados=0;
                    for (Permiso permisoP: perfil.getPermisos()){
                            System.out.println("cambiando "+
permisoP.getFuncion().getNombre());
                            for (Permiso permisoU : item.getPermisos()){
                                if
(permisoU.getFuncion().getId().intValue()==permisoP.getFuncion().getId().intValue()){
                                    if (permisoU.isAsignado()){
                                        asignados++;
                                    }
                                }
                            }
                        }
                    perfil.setActivo(asignados==perfil.getPermisos().size());
                }
        }

        /**
         * Row perfil select listener.
         * @param event
         *            event
         */
        public void rowPerfilSelectListener(SelectEvent event) {
                Perfil perfil=((Perfil) event.getObject()).clon();

            List<Permiso> permisos = item.getPermisos();

            perfil.setActivo(!perfil.isActivo());

            for (Permiso permisoP: perfil.getPermisos()){
                     if (permisoP.isAsignado()){
                            for (Permiso permisoU : permisos){
                                if
(permisoU.getFuncion().getId()==permisoP.getFuncion().getId()){
                                    System.out.println("cambiando "+
permisoU.getFuncion().getNombre()+ " "+perfil.isActivo());
                                    permisoU.setAsignado(perfil.isActivo());
                                }
                            }
                        }
                }
        }

        /**
         * Perfil value change listener.
         * @param event
         *            event
         */
        public void perfilValueChangeListener(ValueChangeEvent event) {
                Perfil perfil=(Perfil)
event.getComponent().getAttributes().get("perfil");

            perfil.setActivo((Boolean)event.getNewValue());
            System.out.println("perfilValueChangeListener");
            for (Permiso permisoP: perfil.getPermisos()){
                    System.out.println("cambiando "+
permisoP.getFuncion().getNombre());
                    for (Permiso permisoU : item.getPermisos()){
```

```
                        if
(permisoU.getFuncion().getId().intValue()==permisoP.getFuncion().getId().intValue()){
                            System.out.println("se cambio "+
permisoU.getFuncion().getNombre()+ " "+perfil.isActivo());
                            permisoU.setAsignado(perfil.isActivo());
                        }
                    }
                }
            Constantes.refresh();
        }

        /**
         * Funcion value change listener.
         * @param event
         *             event
         */
        public void funcionValueChangeListener(ValueChangeEvent event) {
              //TODO si se marca/desmarca una funcion indiidual activar los perfiles
que contengan todas las funciones asignadas
             Permiso permiso=(Permiso)
event.getComponent().getAttributes().get("permiso");

            permiso.setAsignado((Boolean)event.getNewValue());
            System.out.println("funcionValueChangeListener");
          //marcar los perfiles que el usuario tenga asignados
            for (Perfil perfil: perfilList){
               int asignados=0;
                   for (Permiso permisoP: perfil.getPermisos()){
                            System.out.println("cambiando "+
permisoP.getFuncion().getNombre());
                            for (Permiso permisoU : item.getPermisos()){
                                  if
(permisoU.getFuncion().getId().intValue()==permisoP.getFuncion().getId().intValue()){
                                        if (permisoU.isAsignado()){
                                              asignados++;
                                        }
                                  }
                            }
                      }
                   perfil.setActivo(asignados==perfil.getPermisos().size());
            }
            Constantes.refresh();
        }

        /**
         * Obtiene service.
         * @return service
         */
        public UsuarioService getService() {
              return service;
        }

        /**
         * Asigna service.
         * @param service
         *             nuevo service
         */
        public void setService(UsuarioService service) {
              this.service = service;
        }

        /**
         * Obtiene item.
```

```
 * @return item
 */
public Usuario getItem() {
	return item;
}

/**
 * Asigna item.
 * @param item
 *            nuevo item
 */
public void setItem(Usuario item) {
	this.item = item;
}

/**
 * Obtiene item list.
 * @return item list
 */
public List<Usuario> getItemList() {
	return itemList;
}

/**
 * Asigna item list.
 * @param itemList
 *            nuevo item list
 */
public void setItemList(List<Usuario> itemList) {
	this.itemList = itemList;
}

/**
 * Obtiene role admin.
 * @return role admin
 */
public String getRoleAdmin() {
	return roleAdmin;
}

/**
 * Asigna role admin.
 * @param roleAdmin
 *            nuevo role admin
 */
public void setRoleAdmin(String roleAdmin) {
	this.roleAdmin = roleAdmin;
}

/**
 * Obtiene role consulta.
 * @return role consulta
 */
public String getRoleConsulta() {
	return roleConsulta;
}

/**
 * Asigna role consulta.
 * @param roleConsulta
 *            nuevo role consulta
 */
public void setRoleConsulta(String roleConsulta) {
```

```
		this.roleConsulta = roleConsulta;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.ExporterFunc#getTituloReporte()
	 */
	@Override
	public String getTituloReporte() {
		// TODO Auto-generated method stub
		return "catalogo.usuario.titulo";
	}

	/**
	 * Obtiene catalogo service.
	 * @return catalogo service
	 */
	public CatalogoService getCatalogoService() {
		return catalogoService;
	}

	/**
	 * Asigna catalogo service.
	 * @param catalogoService
	 *            nuevo catalogo service
	 */
	public void setCatalogoService(CatalogoService catalogoService) {
		this.catalogoService = catalogoService;
	}

	/**
	 * Obtiene combo area.
	 * @return combo area
	 */
	public List<SelectItem> getComboArea() {
		return comboArea;
	}

	/**
	 * Asigna combo area.
	 * @param comboArea
	 *            nuevo combo area
	 */
	public void setComboArea(List<SelectItem> comboArea) {
		this.comboArea = comboArea;
	}

	/**
	 * Obtiene perfil service.
	 * @return perfil service
	 */
	public PerfilService getPerfilService() {
		return perfilService;
	}

	/**
	 * Asigna perfil service.
	 * @param perfilService
	 *            nuevo perfil service
	 */
	public void setPerfilService(PerfilService perfilService) {
		this.perfilService = perfilService;
	}
```

```
	/**
	 * Obtiene perfil list.
	 * @return perfil list
	 */
	public List<Perfil> getPerfilList() {
		return perfilList;
	}

	/**
	 * Asigna perfil list.
	 * @param perfilList
	 *            nuevo perfil list
	 */
	public void setPerfilList(List<Perfil> perfilList) {
		this.perfilList = perfilList;
	}
}
```

UsuarioDAO.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ----------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.util.List;

/**
 * Interface UsuarioDAO.
 */
public interface UsuarioDAO {

	/**
	 * Obtiene existentes.
	 * @return existentes
	 */
	public List< Usuario> getExistentes();

	/**
	 * Guardar item.
	 * @param item
	 *            item
	 * @return string
	 */
	public String guardarItem(Usuario item);

	/**
	 * Obtiene permisos.
	 * @return permisos
	 */
	public List<Permiso> getPermisos();

	/**
	 * Obtiene modulos.
	 * @return modulos
	 */
	public List<GrupoPagina> getModulos();
```

```
	/**
	 * Validar repetido.
	 * @param item
	 *            item
	 * @return int
	 */
	public int validarRepetido(Usuario item);

}
```

UsuarioDAOImpl.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ----------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.apache.log4j.Logger;
import org.springframework.dao.DataAccessException;
import org.springframework.orm.ibatis.SqlMapClientTemplate;
import org.springframework.orm.ibatis.support.SqlMapClientDaoSupport;

/**
 * Class UsuarioDAOImpl.
 */
public class UsuarioDAOImpl extends SqlMapClientDaoSupport implements UsuarioDAO,
Serializable {
	/**
	 * Verificar que el emisor y el receptor de un objeto serializado
	 * mantengan una compatibilidad en lo que a serialización se refiere.
	 */
	private static final long serialVersionUID = -3378558296519478076L;
	protected static final Logger logger = Logger.getLogger(UsuarioDAOImpl.class);

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.usuario.UsuarioDAO#getExistentes()
	 */
	public List<Usuario> getExistentes() {
		List<Usuario> lista = null;
		try{
			SqlMapClientTemplate template = getSqlMapClientTemplate();
			lista = template.queryForList("selectCatUsuarios");
			if (lista!=null){
				for (Usuario u:lista){

	u.setPermisos(template.queryForList("selectPermisosByIdUsuario", u.getId()));
				}
			}
		}catch(Exception e){
			lista = new ArrayList<Usuario>();
			logger.error(e.getMessage());
		}
```

```
		return lista;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.usuario.UsuarioDAO#getPermisos()
	 */
	public List<Permiso> getPermisos() {
		List<Permiso> lista = null;
		try{
			SqlMapClientTemplate template = getSqlMapClientTemplate();
			lista = template.queryForList("selectPermisosByIdUsuario", -1);
		}catch(Exception e){
			lista = new ArrayList<Permiso>();
			logger.error(e.getMessage());
		}
		return lista;
	}

	/* (non-Javadoc)
	 * @see
mx.edu.ofm.spofm.catalogo.usuario.UsuarioDAO#guardarItem(mx.edu.ofm.spofm.catalogo.usua
rio.Usuario)
	 */
	@Override
	public String guardarItem(Usuario item) {
		String msg="";
		try{
			SqlMapClientTemplate template = getSqlMapClientTemplate();
			if (item.getId()>0){
				template.insert("updateCatUsuarios", item);
			}else{

	item.setId((Integer)template.queryForObject("getCatUsuariosId"));
				template.insert("insertCatUsuarios", item);
			}
			//guardar/eliminar permisos

			//eliminar todos los permisos
			template.delete("deleteUsuariosFunciones", item.getId());

			//agregar los seleccionados
			Map<String, Integer>map = new HashMap<String, Integer>();
			for (Permiso p: item.getPermisos()){
				if (p.isAsignado()){
					map.clear();
					map.put("idUsuario", item.getId());
					map.put("idFuncion", p.getFuncion().getId());
					template.insert("insertUsuarioFuncion", map);
				}
			}

		}catch(Exception e){
			msg = e.getMessage();
			logger.error(e.getMessage());
		}

		return msg;
	}

	/* (non-Javadoc)
	 * @see
mx.edu.ofm.spofm.catalogo.usuario.UsuarioDAO#validarRepetido(mx.edu.ofm.spofm.catalogo.
usuario.Usuario)
```

```
	 */
	public int validarRepetido(Usuario item) {
		int rep = 1;
		try{
			SqlMapClientTemplate template = getSqlMapClientTemplate();
			rep =
(Integer)template.queryForObject("validarRepetidoCatUsuarios", item);
		}catch (DataAccessException dae){
			System.out.println(dae.getCause());
			System.out.println(dae.getMostSpecificCause());
		}
		catch(Exception e){
			rep = 1;
			logger.error(e.getMessage());
		}
		return rep;
	}

	/* (non-Javadoc)
	 * @see mx.edu.ofm.spofm.catalogo.usuario.UsuarioDAO#getModulos()
	 */
	public List<GrupoPagina> getModulos() {
		List<GrupoPagina> lista = null;
		try{
			SqlMapClientTemplate template = getSqlMapClientTemplate();
			lista = template.queryForList("selectCatModulos");
			if (lista!=null){
				for (GrupoPagina m:lista){

	m.setPaginas(template.queryForList("selectPaginasByIdModulo", m.getId()));
					for (Pagina p : m.getPaginas()){

	p.setFunciones(template.queryForList("selectFuncionesByIdPagina", p.getId()));
					}
				}
			}
		}catch(Exception e){
			lista = new ArrayList<GrupoPagina>();
			logger.error(e.getMessage());
		}

		return lista;
	}
}
```

UsuarioService.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ---------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.util.List;

import mx.edu.ofm.spofm.catalogo.CatalogoService;

/**
 * Interface UsuarioService.
```

```
 */
public interface UsuarioService {

	/**
	 * Obtiene existentes.
	 * @return existentes
	 */
	public List< Usuario> getExistentes();

	/**
	 * Guardar item.
	 * @param item
	 *            item
	 * @return string
	 */
	public String guardarItem(Usuario item);

	/**
	 * Obtiene permisos.
	 * @return permisos
	 */
	public List<Permiso> getPermisos();

	/**
	 * Obtiene modulos.
	 * @return modulos
	 */
	public List<GrupoPagina> getModulos();

}
```

UsuarioServiceImpl.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: Catálogo Usuario Seguridad
 * fecha creación: 28/03/2013
 * ----------------------------------------
 * cambios:
 */
package mx.edu.ofm.spofm.catalogo.usuario;

import java.io.Serializable;
import java.util.List;

import mx.edu.ofm.spofm.catalogo.CatalogoService;
import mx.edu.ofm.spofm.util.Constantes;

import org.apache.log4j.Logger;

/**
 * Class UsuarioServiceImpl.
 */
public class UsuarioServiceImpl implements UsuarioService, Serializable {

    /**
     * Verificar que el emisor y el receptor de un objeto serializado
     * mantengan una compatibilidad en lo que a serialización se refiere.
     */
    private static final long serialVersionUID = -3914423880093849401L;
    protected static final Logger logger = Logger.getLogger(UsuarioServiceImpl.class);
    private UsuarioDAO dao;
```

```java
/**
 * Instancia un nuevo usuario service impl.
 * @param dao dao
 */
public UsuarioServiceImpl(UsuarioDAO dao) {
    this.dao = dao;
}

/**
 * Obtiene dao.
 * @return dao
 */
public UsuarioDAO getDao() {
    return dao;
}

/**
 * Asigna dao.
 * @param dao nuevo dao
 */
public void setDao(UsuarioDAO dao) {
    this.dao = dao;
}

/*
 * (non-Javadoc) @see
 * mx.edu.ofm.spofm.catalogo.usuario.UsuarioService#getExistentes()
 */
public List< Usuario> getExistentes() {
    return dao.getExistentes();
}

/*
 * (non-Javadoc) @see
 * mx.edu.ofm.spofm.catalogo.usuario.UsuarioService#getPermisos()
 */
public List< Permiso> getPermisos() {
    return dao.getPermisos();
}

/**
 * Nuevo item.
 * @return string
 */
public String nuevoItem() {
    // TODO Auto-generated method stub
    return null;
}

/**
 * Validar guardar item.
 * @param item item
 * @return string
 */
public String validarGuardarItem(Usuario item) {
    String msg = "";
    if (dao.validarRepetido(item) > 0) {
        msg = Constantes.getMensaje("general.guardado.error.repetido");
    }
    return msg;
}
```

```
    /*
     * (non-Javadoc) @see
     *
mx.edu.ofm.spofm.catalogo.usuario.UsuarioService#guardarItem(mx.edu.ofm.spofm.catalogo.
usuario.Usuario)
     */
    public String guardarItem(Usuario item) {
        String msg = "";
        msg = validarGuardarItem(item);
        if ("".equals(msg)) {
            msg = dao.guardarItem(item);
            if (!"".equals(msg)) {
                msg = Constantes.getMensaje("general.guardado.error");
            }
        }
        return msg;
    }

    /**
     * Restaurar item.
     * @return string
     */
    public String restaurarItem() {
        // TODO Auto-generated method stub
        return null;
    }

    /*
     * (non-Javadoc) @see
     * mx.edu.ofm.spofm.catalogo.usuario.UsuarioService#getModulos()
     */
    public List<GrupoPagina> getModulos() {
        return dao.getModulos();
    }
}
```

usuarioDAO.xml

```
<!DOCTYPE sqlMap
PUBLIC "-//iBATIS.com//DTD SQL Map 2.0//EN"
"http://ibatis.apache.org/dtd/sql-map-2.dtd">

<sqlMap>
    <typeAlias alias="catUsuario" type="mx.edu.ofm.spofm.catalogo.usuario.Usuario"/>
    <typeAlias alias="catGrupoPagina"
type="mx.edu.ofm.spofm.catalogo.usuario.GrupoPagina"/>
    <typeAlias alias="catPagina" type="mx.edu.ofm.spofm.catalogo.usuario.Pagina"/>
    <typeAlias alias="catFuncion" type="mx.edu.ofm.spofm.catalogo.usuario.Funcion"/>
    <typeAlias alias="catPermiso" type="mx.edu.ofm.spofm.catalogo.usuario.Permiso"/>

       <resultMap class="catUsuario" id="catUsuarioMap">
             <result property = "id" column = "IDUSUARIO"/>
           <result property = "pwd" column = "PASSWORDUSUARIO"/>
           <result property = "usuario" column = "USUARIo"/>
           <result property = "nombre" column = "NOMUSUARIO"/>
           <result property = "apellidoP" column = "APEPATERNOUSUARIO"/>
           <result property = "apellidoM" column = "APEMATERNOUSUARIO"/>
           <result property = "area.id" column = "IDAREA"/>
           <result property = "area.nombre" column = "NOMAREA"/>
           <result property = "activo" column = "ACTIVO"/>
    </resultMap>

    <resultMap class="catGrupoPagina" id="catGrupoPaginaMap">
```

```
            <result property = "id" column = "IDGRUPOPAGINA" />
          <result property = "nombre" column = "NOMGRUPOPAGINA"/>
          <result property = "descripcion" column = "DESCRIPCION"/>
          <result property = "activo" column = "ACTIVO"/>
    </resultMap>

    <resultMap class="catPagina" id="catPaginaMap">
            <result property = "id" column = "IDPAGINA" />
          <result property = "descripcion" column = "DESCRIPCION"/>
          <result property = "activo" column = "ACTIVO"/>
    </resultMap>

    <resultMap class="catFuncion" id="catFuncionMap">
            <result property = "id" column = "IDFUNCION"/>
          <result property = "nombre" column = "NOMFUNCION"/>
          <result property = "descripcion" column = "DESCRIPCION"/>
    </resultMap>

    <resultMap class="catPermiso" id="catPermisoMap">
            <result property = "grupoPagina.id" column = "IDGRUPOPAGINA" />
          <result property = "grupoPagina.nombre" column = "NOMGRUPOPAGINA"/>
          <result property = "pagina.id" column = "IDPAGINA" />
          <result property = "pagina.nombre" column = "NOMPAGINA"/>
          <result property = "funcion.id" column = "IDFUNCION"/>
          <result property = "funcion.nombre" column = "NOMFUNCION"/>
          <result property = "asignado" column = "ASIGNADO"/>

    </resultMap>

        <select id="selectCatUsuarios" resultMap="catUsuarioMap">
            select u.*, a.NomArea from usuarios u
            left join Areas a on u.idArea = a.idArea
      </select>

      <select id="selectPermisosByIdUsuario" resultMap="catPermisoMap"
parameterClass="Integer">
            select GP.IDGRUPOPAGINA, GP.NOMGRUPOPAGINA, PA.IDPAGINA,
pa.nompagina,FU.IDFUNCION, FU.NOMFUNCION,
            nvl(
                (select 1 from USUARIOSFUNCIONES up where UP.IDFUNCION=FU.IDFUNCION
and UP.IDUSUARIO = #value#)
            , 0) asignado
            from GRUPOSPAGINAS GP,
            PAGINAS pa,
            FUNCIONES fu
            where
            GP.IDGRUPOPAGINA = PA.IDGRUPOPAGINA
            and PA.IDPAGINA = FU.IDPAGINA
            <!-- order by NOMGRUPOPAGINA, nompagina, NOMFUNCION  -->
      </select>

      <delete id="deleteUsuariosFunciones" parameterClass="Integer">
            delete USUARIOSFUNCIONES where IDUSUARIO = #value#
      </delete>

      <parameterMap class="java.util.HashMap" id="usuarioFuncionMap">
            <parameter property="idUsuario" javaType="Integer" jdbcType="INTEGER"/>
            <parameter property="idFuncion" javaType="Integer" jdbcType="INTEGER"/>
    </parameterMap>

      <insert id="insertUsuarioFuncion" parameterMap="usuarioFuncionMap">
            insert into USUARIOSFUNCIONES (
                  IDUSUARIO,
```

```
                IDFUNCION)
            values (
                ?,
                ?
            )
        </insert>

        <select id="selectFuncionesByIdUsuario" parameterClass="Integer"
resultMap="catFuncionMap">
            select P.* from
            USUARIOSFUNCIONES UF, FUNCIONES f where   P.IDFUNCION=UF.IDFUNCION
            and up.idusuario = #value#
        </select>

        <select id="selectCatModulos" resultMap="catGrupoPaginaMap">
            select * from GRUPOSPAGINAS
        </select>

        <select id="selectPaginasByIdModulo" resultMap="catPaginaMap"
parameterClass="Integer">
            select * from paginas where idgrupopagina = #value#
        </select>

        <select id="selectFuncionesByIdPagina" resultMap="catFuncionMap"
parameterClass="Integer">
            select * from funciones where idpagina = #value#
        </select>

        <!--
            algunas tablas no necesitan secuencia para los id porque son pequeÃ±as
            en otras mas grandes si se usara-->

        <select id="getCatUsuariosId" resultClass="Integer">
            select nvl(max(idusuario), 0) + 1 from usuarios
        </select>

    <insert id="insertCatUsuarios" parameterClass="catUsuario">
    INSERT INTO USUARIOS(
            IDUSUARIO,
            IDAREA,
            USUARIO,
            PASSWORDUSUARIO,
            NOMUSUARIO,
            APEPATERNOUSUARIO,
            APEMATERNOUSUARIO,
            ACTIVO)
       VALUES(
            #id#,
            #area.id#,
            #usuario#,
            #pwd#,
            #nombre#,
            #apellidoP#,
            #apellidoM#,
            #activo#
       )
    </insert>

    <update id="updateCatUsuarios" parameterClass="catUsuario">
            UPDATE
                USUARIOS
            SET
                IDAREA=#area.id#,
```

```
                        USUARIO = #usuario#,
                        PASSWORDUSUARIO = #pwd#,
                        NOMUSUARIO = #nombre#,
                        APEPATERNOUSUARIO = #apellidoP#,
                        APEMATERNOUSUARIO = #apellidoM#,
                        ACTIVO = #activo#
                WHERE
                        IDUSUARIO = #id#
        </update>

        <select id="validarRepetidoCatUsuarios" resultClass="Integer"
parameterClass="catUsuario">
                select count(1) from USUARIOS
                where
                        rownum = 1 and
                        <dynamic>
                                <isNotNull property="id">
                                        <isNotEmpty property="id">
                                                <isNotEqual property="id" compareValue="0">
                                                        <![CDATA[IDUSUARIO <> #id# and]]>
                                                </isNotEqual>
                                        </isNotEmpty>
                                </isNotNull>
                        </dynamic>
                        (
                                translate(upper(USUARIO), 'Ã Ã‰Ã Ã“Ã¡', 'AEIOU')  =
translate(upper(#usuario#), 'Ã Ã‰Ã Ã“Ã¡', 'AEIOU')
                        )
        </select>

</sqlMap>
```

Security

AppUser.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: com.icesoft.icefaces.security
 * fecha creación: 11/02/2013
 * ---------------------------------------
 * cambios:
 */
package com.icesoft.icefaces.security;

import java.io.Serializable;
import java.util.Set;

/**
 * Class AppUser.
 */
public class AppUser implements Serializable {

    /**
     * Verificar que el emisor y el receptor de un objeto serializado
     * mantengan una compatibilidad en lo que a serialización se refiere.
     */
    private static final long serialVersionUID = -1523107730870725259L;
    private String nombre;
    private String apPat;
    private String apMat;
```

```
private String idUsuario;
private String usuario;

/**
 * Obtiene usuario.
 * @return usuario
 */
public String getUsuario() {
    return usuario;
}

/**
 * Asigna usuario.
 * @param usuario nuevo usuario
 */
public void setUsuario(String usuario) {
    this.usuario = usuario;
}
private String password;
private Boolean activo;
private Set<String> roles;

/**
 * Instancia un nuevo app user.
 */
public AppUser() {
}

 /**
 * Obtiene password.
 * @return password
 */
public String getPassword() {
    return password;
}

/**
 * Instancia un nuevo app user.
 *
 * @param nombre nombre
 * @param apPat ap pat
 * @param apMat ap mat
 * @param idUsuario id usuario
 * @param password password
 * @param activo activo
 * @param roles roles
 */
public AppUser(String nombre, String apPat, String apMat, String idUsuario,
        String password, boolean activo, Set<String> roles) {
    super();
    this.nombre = nombre;
    this.apPat = apPat;
    this.apMat = apMat;
    this.idUsuario = idUsuario;
    this.password = password;
    this.activo = activo;
    this.roles = roles;
}

/**
 * Obtiene roles.
 * @return roles
 */
```

```
public Set<String> getRoles() {
    return roles;
}

/**
 * Obtiene nombre.
 * @return nombre
 */
public String getNombre() {
    return nombre;
}

/**
 * Asigna nombre.
 * @param nombre nuevo nombre
 */
public void setNombre(String nombre) {
    this.nombre = nombre;
}

/**
 * Obtiene ap pat.
 * @return ap pat
 */
public String getApPat() {
    return apPat;
}

/**
 * Asigna ap pat.
 * @param apPat nuevo ap pat
 */
public void setApPat(String apPat) {
    this.apPat = apPat;
}

/**
 * Obtiene ap mat.
 * @return ap mat
 */
public String getApMat() {
    return apMat;
}

/**
 * Asigna ap mat.
 * @param apMat nuevo ap mat
 */
public void setApMat(String apMat) {
    this.apMat = apMat;
}

/**
 * Obtiene id usuario.
 * @return id usuario
 */
public String getIdUsuario() {
    return idUsuario;
}

/**
 * Asigna id usuario.
 * @param idUsuario nuevo id usuario
```

```
     */
    public void setIdUsuario(String idUsuario) {
        this.idUsuario = idUsuario;
    }

    /**
     * Asigna password.
     * @param password nuevo password
     */
    public void setPassword(String password) {
        this.password = password;
    }

    /**
     * Asigna roles.
     * @param roles nuevo roles
     */
    public void setRoles(Set<String> roles) {
        this.roles = roles;
    }

    /**
     * Obtiene activo.
     * @return activo
     */
    public Boolean getActivo() {
        return activo;
    }

    /**
     * Asigna activo.
     * @param activo nuevo activo
     */
    public void setActivo(Boolean activo) {
        this.activo = activo;
    }
}
```

UserDAO.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: com.icesoft.icefaces.security
 * fecha creación: 11/02/2013
 * ---------------------------------------
 * cambios:
*/
package com.icesoft.icefaces.security;

/**
 * Interface UserDAO.
 */
public interface UserDAO  {

    /**
        * Find user.
        * @param username
        *            username
        * @return app user
        */
    AppUser findUser(String username) ;
}
```

UserDAOImpl.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: com.icesoft.icefaces.security
 * fecha creación: 11/02/2013
 * ---------------------------------------
 * cambios:
 */
package com.icesoft.icefaces.security;

import java.util.HashSet;
import java.util.List;
import java.util.Set;


import org.apache.log4j.Logger;
import org.springframework.orm.ibatis.SqlMapClientTemplate;
import org.springframework.orm.ibatis.support.SqlMapClientDaoSupport;

/**
 * Class UserDAOImpl.
 */
public class UserDAOImpl extends SqlMapClientDaoSupport  implements UserDAO {
      protected static final Logger logger = Logger.getLogger(UserDAOImpl.class);

    /* (non-Javadoc)
     * @see com.icesoft.icefaces.security.UserDAO#findUser(java.lang.String)
     */
    public  AppUser findUser(String userName) {
             SqlMapClientTemplate template = getSqlMapClientTemplate();
             AppUser lista = null;
             lista = (AppUser)template.queryForObject("findUser", userName);
             if (lista != null ){
                    lista.setRoles(findRolbyUser(lista.getIdUsuario()));
             }
             return lista;
      }

    /**
        * Find rolby user.
        * @param userName
        *            user name
        * @return asigna
        */
    public  Set<String> findRolbyUser(String userName) {


             SqlMapClientTemplate template = getSqlMapClientTemplate();
             List<String> lista = null;
             Set<String> set = new HashSet<String>();
             lista = template.queryForList("findRolbyUser", userName);
             if (lista!=null){
                    set.addAll(lista);
             }
             return set;
      }
}
```

UserDetalisServiceImpl.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: com.icesoft.icefaces.security
 * fecha creación: 11/02/2013
 * ---------------------------------------
 * cambios:
 */
package com.icesoft.icefaces.security;

import org.springframework.dao.DataAccessException;
import org.springframework.security.GrantedAuthority;
import org.springframework.security.GrantedAuthorityImpl;
import org.springframework.security.userdetails.User;
import org.springframework.security.userdetails.UserDetails;
import org.springframework.security.userdetails.UserDetailsService;
import org.springframework.security.userdetails.UsernameNotFoundException;

/**
 * Class UserDetailsServiceImpl.
 */
public class UserDetailsServiceImpl implements UserDetailsService {

    private UserDAO userDAO;
    private AppUser usuario;
    private User user;

    /**
        * Instancia un nuevo user details service impl.
        * @param userDao
        *            user dao
        */
    public UserDetailsServiceImpl(UserDAO userDao) {
        this.userDAO = userDao;
    }

    /* (non-Javadoc)
     * @see
org.springframework.security.userdetails.UserDetailsService#loadUserByUsername(java.lan
g.String)
     */
    public UserDetails loadUserByUsername(String username)
            throws UsernameNotFoundException, DataAccessException {
       usuario = userDAO.findUser(username);
        if (usuario == null)
            throw new UsernameNotFoundException("Usuario NO Econtrado: " + username);
        else {
            return makeUser(usuario);
        }
    }

    /**
        * Make user.
        * @param appUser
        *            app user
        * @return org.springframework.security.userdetails. user
        */
    private org.springframework.security.userdetails.User makeUser(AppUser appUser) {
       user = new org.springframework.security.userdetails.User(appUser.getIdUsuario(),
appUser
                .getPassword(), true, true, true, true,
                makeGrantedAuthorities(appUser));
        return user;
```

```
}

/**
    * Obtiene user dao.
    * @return user dao
    */
public UserDAO getUserDAO() {
          return userDAO;
   }

   /**
    * Asigna user dao.
    * @param userDAO
    *             nuevo user dao
    */
   public void setUserDAO(UserDAO userDAO) {
          this.userDAO = userDAO;
   }

   /**
    * Obtiene usuario.
    * @return usuario
    */
   public AppUser getUsuario() {
          return usuario;
   }

   /**
    * Asigna usuario.
    * @param usuario
    *             nuevo usuario
    */
   public void setUsuario(AppUser usuario) {
          this.usuario = usuario;
   }

   /**
    * Obtiene user.
    * @return user
    */
   public User getUser() {
          return user;
   }

   /**
    * Asigna user.
    *
    * @param user
    *             nuevo user
    */
   public void setUser(User user) {
          this.user = user;
   }

   /**
    * Make granted authorities.
    * @param user
    *             user
    * @return granted authority[]
    */
   private GrantedAuthority[] makeGrantedAuthorities(AppUser user) {
     GrantedAuthority[] result = new GrantedAuthority[user.getRoles().size()];
     int i = 0;
```

```
        for (String role : user.getRoles()) {
            result[i++] = new GrantedAuthorityImpl(role);
        }
        return result;
    }
}
```

userDAO.xml

```
<!DOCTYPE sqlMap
PUBLIC "-//iBATIS.com//DTD SQL Map 2.0//EN"
"http://ibatis.apache.org/dtd/sql-map-2.dtd">

<sqlMap>
    <typeAlias alias="user" type="com.icesoft.icefaces.security.AppUser"/>

      <resultMap class="user" id="userMap">
            <result property = "nombre" column = "NOMUSUARIO"/>
          <result property = "apPat" column = "APEPATERNOUSUARIO"/>
          <result property = "apMat" column = "APEMATERNOUSUARIO"/>
          <result property = "idUsuario" column = "IDUSUARIO"/>
          <result property = "usuario" column = "USUARIO"/>
          <result property = "password" column = "PASSWORDUSUARIO"/>
          <result property = "activo" column = "ACTIVO"/>
    </resultMap>

      <select id="findUser" parameterClass="String" resultMap="userMap">
            select * from usuarios where usuario=#value#
            and rownum = 1
      </select>

      <select id="findRolbyUser" parameterClass="String" resultClass="String">
            select P.NOMFUNCION from
            USUARIOSFUNCIONES up, FUNCIONES p where   P.IDFUNCION=UP.IDFUNCION
            and idusuario=#values#
      </select>

</sqlMap>
```

Security.Beans
LoginBean.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: com.icesoft.icefaces.security
 * fecha creación: 11/02/2013
 * ---------------------------------------
 * cambios:
 */
package com.icesoft.icefaces.security.beans;

import org.springframework.security.ui.AbstractProcessingFilter;

import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.servlet.http.HttpServletRequest;

/**
 * Class LoginBean.
 */
```

```
public class LoginBean {

    // properties
    private String userId;

    private String password;

    /**
        * Instancia un nuevo login bean.
        */
    public LoginBean() {

        Exception ex = (Exception) FacesContext
                .getCurrentInstance()
                .getExternalContext()
                .getSessionMap()
                .get(AbstractProcessingFilter.SPRING_SECURITY_LAST_EXCEPTION_KEY);

        if (ex != null)
            FacesContext.getCurrentInstance().addMessage(
                    null,
                    new FacesMessage(FacesMessage.SEVERITY_ERROR, ex
                            .getMessage(), ex.getMessage()));
    }

    /**
        * Obtiene password.
        * @return password
        */
    public String getPassword() {
        return password;
    }

    /**
        * Asigna password.
        * @param password
        *            nuevo password
        */
    public void setPassword(String password) {
        this.password = password;
    }

    /**
        * Obtiene user id.
        * @return user id
        */
    public String getUserId() {
        return userId;
    }

    /**
        * Asigna user id.
        * @param userId
        *            nuevo user id
        */
    public void setUserId(String userId) {
        this.userId = userId;
    }

    /**
        * Login.
        * @param e
        *            e
```

```
        * @throws IOException
        *             Signals that an I/O exception has occurred.
        */
    public void login(ActionEvent e) throws java.io.IOException {
       HttpServletRequest req = (HttpServletRequest)
FacesContext.getCurrentInstance().getExternalContext().getRequest();

FacesContext.getCurrentInstance().getExternalContext().redirect(req.getContextPath().to
String()+"/j_spring_security_check?j_username=" + userId + "&j_password=" + password);
      }

    /**
        * Logout.
        * @param e
        *            e
        * @throws IOException
        *             Signals that an I/O exception has occurred.
        */
    public void logout(ActionEvent e) throws java.io.IOException {
       HttpServletRequest req = (HttpServletRequest)
FacesContext.getCurrentInstance().getExternalContext().getRequest();

FacesContext.getCurrentInstance().getExternalContext().redirect(req.getContextPath().to
String()+"/j_spring_security_logout");
    }
}
```

ShoppingBean.java

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 * caso de uso: com.icesoft.icefaces.security
 * fecha creación: 11/02/2013
 * ---------------------------------------
 * cambios:
 */
package com.icesoft.icefaces.security.beans;

import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.servlet.http.HttpServletRequest;

/**
 * Class ShoppingBean.
 */
public class ShoppingBean {

    // properties
    private String productId;

    /**
        * Instancia un nuevo shopping bean.
        */
    public ShoppingBean() {
    }

    /**
        * Send.
        * @return string
        */
    public String send() {
        return ("success");
```

```
    }

    /**
        * Obtiene product id.
        * @return product id
        */
    public String getProductId() {
        return productId;
    }

    /**
        * Asigna product id.
        * @param productId
        *            nuevo product id
        */
    public void setProductId(String productId) {
        this.productId = productId;
    }

    /**
        * Logout.
        * @param e
        *            e
        * @throws IOException
        *             Signals that an I/O exception has occurred.
        */
    public void logout(ActionEvent e) throws java.io.IOException {
       HttpServletRequest req = (HttpServletRequest)
FacesContext.getCurrentInstance().getExternalContext().getRequest();

FacesContext.getCurrentInstance().getExternalContext().redirect(req.getContextPath().to
String()+"/j_spring_security_logout");
    }
}
```

serviceContext.xml

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 */

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:security="http://www.springframework.org/schema/security"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
            http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-2.0.2.xsd">

       <!-- catalogos -->

    <bean id="catalogoService"
          class="mx.edu.ofm.spofm.catalogo.CatalogoServiceImpl">
        <constructor-arg ref="catalogoDAO"/>
    </bean>

    <bean id="usuarioService"
          class="mx.edu.ofm.spofm.catalogo.usuario.UsuarioServiceImpl">
        <constructor-arg ref="usuarioDAO"/>
    </bean>
```

```
    <bean id="direccionService"
          class="mx.edu.ofm.spofm.catalogo.direccion.DireccionServiceImpl">
        <constructor-arg ref="direccionDAO"/>
    </bean>

    <bean id="areaService"
          class="mx.edu.ofm.spofm.catalogo.area.AreaServiceImpl">
        <constructor-arg ref="areaDAO"/>
    </bean>

    <bean id="entidadesService"
          class="mx.edu.ofm.spofm.vista.entidades.EntidadesServiceImpl">
        <constructor-arg ref="entidadesDAO"/>
    </bean>

     <bean id="tipoService"
          class="mx.edu.ofm.spofm.vista.tipo.TpoServiceImpl">
        <constructor-arg ref="tipoDAO"/>
    </bean>

    <bean id="canceladaService"
          class="mx.edu.ofm.spofm.vista.cancelaciones.CanceladaServiceImpl">
        <constructor-arg ref="canceladaDAO"/>
    </bean>

    <bean id="municipioService"
          class="mx.edu.ofm.spofm.vista.municipio.MunicipioServiceImpl">
        <constructor-arg ref="municipioDAO"/>
    </bean>

    <bean id="ceremoniaService"
          class="mx.edu.ofm.spofm.vista.ceremonia.CeremoniaServiceImpl">
        <constructor-arg ref="ceremoniaDAO"/>
    </bean>

     <bean id="horarioService"
          class="mx.edu.ofm.spofm.vista.horario.HorarioServiceImpl">
        <constructor-arg ref="horarioDAO"/>
    </bean>

     <bean id="parroquiaService"
          class="mx.edu.ofm.spofm.vista.parroquia.ParroquiaServiceImpl">
        <constructor-arg ref="parroquiaDAO"/>
    </bean>

    <bean id="parametroWebService"
          class="mx.edu.ofm.spofm.catalogo.parametrosweb.ParametroWebServiceImpl">
        <constructor-arg ref="parametroWebDAO"/>
    </bean>

    <bean id="capillasService"
          class="mx.edu.ofm.spofm.capillas.CapillasServiceImpl">
        <constructor-arg ref="capillasServiceDAO"/>
    </bean>

    <bean id="altaCeremoniasService"
      class="mx.edu.ofm.spofm.altaceremonias.AltaCeremoniasServiceImpl">
        <constructor-arg ref="altaCeremoniasServiceDAO"/>
    </bean>

    <!-- catalogos fin -->

</beans>
```

Config
applicationContext.xml

```
/*
 * sistema: Parroquial OFM Web
 * autor: Emmanuel Oliva
 */

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:security="http://www.springframework.org/schema/security"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
            http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-2.0.2.xsd">

       <!-- catalogos -->

    <bean id="catalogoService"
              class="mx.edu.ofm.spofm.catalogo.CatalogoServiceImpl">
        <constructor-arg ref="catalogoDAO"/>
    </bean>

    <bean id="usuarioService"
              class="mx.edu.ofm.spofm.catalogo.usuario.UsuarioServiceImpl">
        <constructor-arg ref="usuarioDAO"/>
    </bean>

    <bean id="direccionService"
              class="mx.edu.ofm.spofm.catalogo.direccion.DireccionServiceImpl">
        <constructor-arg ref="direccionDAO"/>
    </bean>

    <bean id="areaService"
         class="mx.edu.ofm.spofm.catalogo.area.AreaServiceImpl">
        <constructor-arg ref="areaDAO"/>
    </bean>

    <bean id="entidadesService"
          class="mx.edu.ofm.spofm.vista.entidades.EntidadesServiceImpl">
        <constructor-arg ref="entidadesDAO"/>
    </bean>

       <bean id="tipoService"
          class="mx.edu.ofm.spofm.vista.tipo.TipoServiceImpl">
        <constructor-arg ref="tipoDAO"/>
    </bean>

     <bean id="canceladaService"
          class="mx.edu.ofm.spofm.vista.cancelaciones.CanceladaServiceImpl">
        <constructor-arg ref="canceladaDAO"/>
    </bean>

    <bean id="municipioService"
           class="mx.edu.ofm.spofm.vista.municipio.MunicipioServiceImpl">
        <constructor-arg ref="municipioDAO"/>
    </bean>

    <bean id="ceremoniaService"
           class="mx.edu.ofm.spofm.vista.ceremonia.CeremoniaServiceImpl">
        <constructor-arg ref="ceremoniaDAO"/>
```

```
    </bean>

    <bean id="horarioService"
           class="mx.edu.ofm.spofm.vista.horario.HorarioServiceImpl">
        <constructor-arg ref="horarioDAO"/>
    </bean>

     <bean id="parroquiaService"
           class="mx.edu.ofm.spofm.vista.parroquia.ParroquiaServiceImpl">
        <constructor-arg ref="parroquiaDAO"/>
    </bean>

        <bean id="parametroWebService"
          class="mx.edu.ofm.spofm.catalogo.parametrosweb.ParametroWebServiceImpl">
        <constructor-arg ref="parametroWebDAO"/>
    </bean>

     <bean id="capillasService"
         class="mx.edu.ofm.spofm.catalogo.capillas.CapillasServiceImpl">
        <constructor-arg ref="capillasDAO"/>
    </bean>

    <bean id="altaCeremoniasService"
         class="mx.edu.ofm.spofm.catalogo.altaceremonias.AltaCeremoniasServiceImpl">
        <constructor-arg ref="altaCeremoniasDAO"/>
    </bean>

    <bean id="perfilService"
         class="mx.edu.ofm.spofm.catalogo.perfiles.PerfilServiceImpl">
        <constructor-arg ref="perfilDAO"/>
    </bean>

    <!-- catalogos fin -->

</beans>
```

mapperMyBatis.xml

```
<!--sistema: Parroquial OFM Web-->
<!--autor: Emmanuel Oliva-->

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE sqlMapConfig
       PUBLIC "-//ibatis.apache.org//DTD SQL Map Config 2.0//EN"
       "http://ibatis.apache.org/dtd/sql-map-config-2.dtd">
<sqlMapConfig>

    <sqlMap resource="com/icesoft/icefaces/security/userDAO.xml" />
    <sqlMap resource="mx/edu/ofm/spofm/catalogo/catalogoDAO.xml" />
    <sqlMap resource="mx/edu/ofm/spofm/catalogo/usuario/usuarioDAO.xml" />
    <sqlMap resource="mx/edu/ofm/spofm/catalogo/direccion/direccionDAO.xml" />
    <sqlMap resource="mx/edu/ofm/spofm/catalogo/area/areaDAO.xml" />

   <sqlMap resource="mx/edu/ofm/spofm/vista/entidades/entidadesDAO.xml" />
    <sqlMap resource="mx/edu/ofm/spofm/vista/tipo/tipoDAO.xml" />
    <sqlMap resource="mx/edu/ofm/spofm/vista/parroquia/parroquiaDAO.xml" />
    <sqlMap resource="mx/edu/ofm/spofm/vista/cancelaciones/canceladaDAO.xml" />

    <sqlMap resource="mx/edu/ofm/spofm/vista/municipio/municipioDAO.xml" />
    <sqlMap resource="mx/edu/ofm/spofm/vista/ceremonia/ceremoniaDAO.xml" />
     <sqlMap resource="mx/edu/ofm/spofm/vista/horario/horarioDAO.xml" />

    <sqlMap resource="mx/edu/ofm/spofm/catalogo/parametrosweb/parametroWebDAO.xml" />
```

```
    <sqlMap resource="mx/edu/ofm/spofm/catalogo/capillas/capillasDAO.xml"/>
    <sqlMap resource="mx/edu/ofm/spofm/catalogo/altaceremonias/altaceremoniasDAO.xml"/>

    <sqlMap resource="mx/edu/ofm/spofm/catalogo/perfiles/perfilDAO.xml" />

</sqlMapConfig>
```

mensajes.properties

```
# sistema: Parroquial OFM Web
# autor: Patricia Roldan

# Messages JSF
javax.faces.component.UIInput.REQUIRED = Dato requerido.
javax.faces.validator.LongRangeValidator.NOT_IN_RANGE = El valor del campo debe de
estar entre {0} y {1}.
javax.faces.validator.LengthValidator.MAXIMUM = La longitud del campo es mayor que
''{0}''
javax.faces.validator.LengthValidator.MINIMUM = La longitud del campo es menor que
''{0}''
javax.faces.validator.RegexValidator.NOT_MATCHED = El valor del campo tiene caracteres
invalidos.
javax.faces.validator.LongRangeValidator.TYPE =
javax.faces.component.UIInput.CONVERSION =
javax.faces.component.UISelectOne.INVALID = {0}: M\u00f3dulo Requerido.
javax.faces.converter.NumberConverter.NUMBER=''{0}'' no es un numero valido.
javax.faces.converter.NumberConverter.NUMBER_detail=''{0}'' no es un numero. Ejemplo:
{1}

#Validacion de Validaciones en vista
noDouble = S\u00f3lo se permiten valores n\u00famericos.
noAlfanumerico = S\u00f3lo se permiten valores alfanumericos.
noClave = Dato incorrecto, favor de verificar. Los caracteres validos son: a-z, A-Z, 0-
9, - y _.
noAZaz09 = Dato incorrecto, favor de verificar. Los caracteres validos son: a-z, A-Z,
0-9, y vocales acentuadas.
noPositivo = S\u00f3lo se permiten n\u00fameros positivos.
noInteger = S\u00f3lo se permiten n\u00fameros enteros.
noNombrePropio = Dato incorrecto, capture valores de la A-Z, a-z, espacio y vocales
acentuadas.
noNombre = Dato incorrecto, capture valores de la A-Z, a-z, 0-9, espacio y vocales
acentuadas.
noRango = Debe ser un n\u00famero entre -2147483648 y 2147483647. Ejemplo: 9346.

mensajeProceso = Procesando...

#general
general.guardado = Registro guardado.
general.guardado.error.repetido = Revise los datos ingresados, no pueden existir campos
repetidos.
general.guardado.error = Error al intengar guardar el registro, contacte con su
administrador de sistemas.
general.confirmAgregarRegistro = \u00bfDesea guardar los cambios realizados?
general.toggle.activo.true = Usted est\u00e1 habilitando el registro
general.toggle.activo.false = Usted est\u00e1 deshabilitando el registro

#mensaje catalogo Nombre y apellido
catalogo.nombre.error.notequal = Solo pueden variar en acentos y Mayusculas y
minusculas.
```

myBatisDAOContext.xml

```
<!--sistema: Parroquial OFM Web-->
<!--autor: Emmanuel Oliva-->

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:batch="http://www.springframework.org/schema/batch"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
       http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
       http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-2.0.xsd
       http://www.springframework.org/schema/batch
http://www.springframework.org/schema/batch/spring-batch-2.0.xsd">

    <!-- Bean Dao Spring -->
    <bean id="userDAO" class="com.icesoft.icefaces.security.UserDAOImpl">
        <property name="dataSource" ref="basicDataSource"/>
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean"/>
    </bean>

    <!-- Fabrica de mapeos -->
    <bean id="sqlMapClientFactoryBean"
class="org.springframework.orm.ibatis.SqlMapClientFactoryBean">
        <property name="configLocation">
            <value>classpath:config/mapperMyBatis.xml</value>
        </property>
    </bean>

          <!-- Data Source directo -->
    <bean id="basicDataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-
method="close">
        <property name="driverClassName">
            <value>oracle.jdbc.driver.OracleDriver</value>
        </property>
        <property name="url">
            <value>jdbc:oracle:thin:@//localhost:1521/xe</value>

        </property>
        <property name="username">
            <value>Emmanuel</value>
        </property>
        <property name="password">
            <value>scorpions</value>
        </property>
    </bean>

          <!-- catalogos -->
    <bean id="catalogoDAO" class="mx.edu.ofm.spofm.catalogo.CatalogoDAOImpl">
        <property name="dataSource" ref="basicDataSource"/>
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean"/>
    </bean>

    <bean id="usuarioDAO" class="mx.edu.ofm.spofm.catalogo.usuario.UsuarioDAOImpl">
        <property name="dataSource" ref="basicDataSource"/>
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean"/>
    </bean>

    <bean id="areaDAO" class="mx.edu.ofm.spofm.catalogo.area.AreaDAOImpl">
        <property name="dataSource" ref="basicDataSource"/>
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean"/>
    </bean>
```

```
    <bean id="parametroWebDAO"
          class="mx.edu.ofm.spofm.catalogo.parametrosweb.ParametroWebDAOImpl">
        <property name="dataSource" ref="basicDataSource" />
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean" />
    </bean>

    <bean id="altaCeremoniasDAO"
               class="mx.edu.ofm.spofm.catalogo.altaceremonias.AltaCeremoniasDAOImpl">
        <property name="dataSource" ref="basicDataSource" />
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean" />
    </bean>

    <bean id="capillasDAO"
              class="mx.edu.ofm.spofm.catalogo.capillas.CapillasDAOImpl">
        <property name="dataSource" ref="basicDataSource" />
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean" />
    </bean>

    <bean id="perfilDAO"
          class="mx.edu.ofm.spofm.catalogo.perfiles.PerfilDAOImpl">
        <property name="dataSource" ref="basicDataSource" />
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean" />
    </bean>

    <!-- catalogos fin -->

           <!-- vistas -->

    <bean id="entidadesDAO"
          class="mx.edu.ofm.spofm.vista.entidades.EntidadesDAOImpl">
        <property name="dataSource" ref="basicDataSource" />
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean" />
    </bean>

    <bean id="tipoDAO"
          class="mx.edu.ofm.spofm.vista.tipo.TipoDAOImpl">
        <property name="dataSource" ref="basicDataSource" />
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean" />
    </bean>

    <bean id="canceladaDAO"
          class="mx.edu.ofm.spofm.vista.cancelaciones.CanceladaDAOImpl">
        <property name="dataSource" ref="basicDataSource" />
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean" />
    </bean>

      <bean id="parroquiaDAO"
          class="mx.edu.ofm.spofm.vista.parroquia.ParroquiaDAOImpl">
        <property name="dataSource" ref="basicDataSource" />
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean" />
    </bean>

      <bean id="municipioDAO"
class="mx.edu.ofm.spofm.vista.municipio.MunicipioDAOImpl">
        <property name="dataSource" ref="basicDataSource"/>
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean"/>
    </bean>

      <bean id="ceremoniaDAO"
class="mx.edu.ofm.spofm.vista.ceremonia.CeremoniaDAOImpl">
        <property name="dataSource" ref="basicDataSource"/>
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean"/>
    </bean>
```

```
      <bean id="horarioDAO" class="mx.edu.ofm.spofm.vista.horario.HorarioDAOImpl">
        <property name="dataSource" ref="basicDataSource"/>
        <property name="sqlMapClient" ref="sqlMapClientFactoryBean"/>
    </bean>

          <!-- vistas fin -->

</beans>
```

serviceContext.xml

```
<!--sistema: Parroquial OFM Web-->
<!--autor: Emmanuel Oliva-->

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:security="http://www.springframework.org/schema/security"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
            http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-2.0.2.xsd">

          <!-- catalogos -->

    <bean id="catalogoService"
              class="mx.edu.ofm.spofm.catalogo.CatalogoServiceImpl">
        <constructor-arg ref="catalogoDAO"/>
    </bean>

    <bean id="usuarioService"
              class="mx.edu.ofm.spofm.catalogo.usuario.UsuarioServiceImpl">
        <constructor-arg ref="usuarioDAO"/>
    </bean>

    <bean id="direccionService"
              class="mx.edu.ofm.spofm.catalogo.direccion.DireccionServiceImpl">
        <constructor-arg ref="direccionDAO"/>
    </bean>

    <bean id="areaService"
         class="mx.edu.ofm.spofm.catalogo.area.AreaServiceImpl">
        <constructor-arg ref="areaDAO"/>
    </bean>

   <bean id="entidadesService"
          class="mx.edu.ofm.spofm.vista.entidades.EntidadesServiceImpl">
        <constructor-arg ref="entidadesDAO"/>
    </bean>

       <bean id="tipoService"
          class="mx.edu.ofm.spofm.vista.tipo.TipoServiceImpl">
        <constructor-arg ref="tipoDAO"/>
    </bean>

     <bean id="canceladaService"
          class="mx.edu.ofm.spofm.vista.cancelaciones.CanceladaServiceImpl">
        <constructor-arg ref="canceladaDAO"/>
    </bean>

    <bean id="municipioService"
           class="mx.edu.ofm.spofm.vista.municipio.MunicipioServiceImpl">
```

```
        <constructor-arg ref="municipioDAO"/>
    </bean>

    <bean id="ceremoniaService"
           class="mx.edu.ofm.spofm.vista.ceremonia.CeremoniaServiceImpl">
        <constructor-arg ref="ceremoniaDAO"/>
    </bean>

    <bean id="horarioService"
           class="mx.edu.ofm.spofm.vista.horario.HorarioServiceImpl">
        <constructor-arg ref="horarioDAO"/>
    </bean>

     <bean id="parroquiaService"
           class="mx.edu.ofm.spofm.vista.parroquia.ParroquiaServiceImpl">
        <constructor-arg ref="parroquiaDAO"/>
    </bean>

        <bean id="parametroWebService"
          class="mx.edu.ofm.spofm.catalogo.parametrosweb.ParametroWebServiceImpl">
        <constructor-arg ref="parametroWebDAO"/>
    </bean>

     <bean id="capillasService"
         class="mx.edu.ofm.spofm.catalogo.capillas.CapillasServiceImpl">
        <constructor-arg ref="capillasDAO"/>
    </bean>

    <bean id="altaCeremoniasService"
         class="mx.edu.ofm.spofm.catalogo.altaceremonias.AltaCeremoniasServiceImpl">
        <constructor-arg ref="altaCeremoniasDAO"/>
    </bean>


    <bean id="perfilService"
         class="mx.edu.ofm.spofm.catalogo.perfiles.PerfilServiceImpl">
        <constructor-arg ref="perfilDAO"/>
    </bean>

    <!-- catalogos fin -->

</beans>
```

springSecurityContext.xml

```
<!--sistema: Parroquial OFM Web-->
<!--autor: Emmanuel Oliva-->

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:security="http://www.springframework.org/schema/security"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
    http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-2.5.xsd
    http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-2.0.xsd">

    <security:http auto-config="true" access-denied-page="/accessDenied.jspx">
        <security:intercept-url pattern="/secured/**"
```

```
                                access="ROLE_ALLACCESS, ROLE_URLACCESS"/>
        <security:form-login login-page="/springSecurityLogin.jspx"
                             default-target-url="/secured/spofm.jspx"/>
        <security:logout logout-success-url="/springSecurityLogin.jspx"/>
    </security:http>

<security:authentication-provider user-service-ref="userDetailsService"/>

    <bean id="userDetailsService"
          class="com.icesoft.icefaces.security.UserDetailsServiceImpl">
        <constructor-arg ref="userDAO"/>
    </bean>

</beans>
```

textos.properties

```
#sistema: Parroquial OFM Web
#autor: Patricia Roldan

# textos generales
general.nuevo = NUEVO
general.guardar = GUARDAR
general.modificar = MODIFICAR
general.eliminar = ELIMINAR
general.cancelar = CANCELAR
general.activo = Activo
general.inactivo = Inactivo
general.id = Identificador
general.buscar = Buscar
general.terminar = TERMINAR
general.siguiente = SIGUIENTE
general.imprimir = IMPRIMIR

#menu
menu.catalogo = Cat\u00e1logos
menu.catalogo.ceremonias = Ceremonias
menu.catalogo.solicitantes = Solicitantes
menu.catalogo.parroquias = Parroquias
menu.catalogo.capillas = Capillas
menu.catalogo.altaceremonias = Registro de Ceremonias
menu.catalogo.sacerdotes = Sacerdotes
menu.administracion = Administraci\u00f3n del Sistema
menu.vista = Consultas
menu.reporte = Reportes
menu.recepcion = Ceremonias
menu.otros = Otros

#textos de catalogos
catalogo.usuario.titulo = Usuarios
catalogo.usuario.titulo.file = Usuarios
catalogo.usuario.edicion.atributo.1.label = Nombre(s)
catalogo.usuario.edicion.atributo.2.label = Apellido Paterno
catalogo.usuario.edicion.atributo.3.label = Apellido Materno
catalogo.usuario.edicion.atributo.4.label = Usuario
catalogo.usuario.edicion.atributo.5.label = Contrase\u00f1a
catalogo.usuario.edicion.atributo.6.label = Activo
catalogo.usuario.edicion.atributo.7.label = \u00c1rea
catalogo.usuario.edicion.funcion.atributo.1.label = M\u00f3dulo
catalogo.usuario.edicion.funcion.atributo.2.label = P\u00e1gina
catalogo.usuario.edicion.funcion.atributo.3.label = Funci\u00f3n
catalogo.usuario.edicion.funcion.atributo.4.label = Asignado
catalogo.usuario.edicion.perfil.atributo.1.label = Perfil
```

```
catalogo.usuario.edicion.perfil.atributo.2.label = Asignar

catalogo.area.titulo = \u00c1reas Institucionales
catalogo.area.titulo.file = \u00c1reas Institucionales
catalogo.area.edicion.atributo.1.label = Nombre
catalogo.area.edicion.atributo.2.label = Descripci\u00f3n
catalogo.area.edicion.atributo.3.label = Direcci\u00f3n

catalogo.perfil.titulo = Perfiles
catalogo.perfil.titulo.file = perfiles
catalogo.perfil.edicion.atributo.1.label = Clave
catalogo.perfil.edicion.atributo.2.label = Nombre
catalogo.perfil.edicion.atributo.3.label = Descripci\u00f3n
catalogo.perfil.edicion.funcion.atributo.1.label = M\u00f3dulo
catalogo.perfil.edicion.funcion.atributo.2.label = P\u00e1gina
catalogo.perfil.edicion.funcion.atributo.3.label = Funci\u00f3n
catalogo.perfil.edicion.funcion.atributo.4.label = Asignado

catalogo.capillas.titulo = Capillas
catalogo.capillas.titulo.file = Capillas
catalogo.capillas.edicion.atributo.1.label = Entidad
catalogo.capillas.edicion.atributo.2.label = Municipio
catalogo.capillas.edicion.atributo.3.label = Parroquia
catalogo.capillas.edicion.atributo.4.label = Nombre Capilla
catalogo.capillas.edicion.atributo.5.label = Colonia
catalogo.capillas.edicion.atributo.6.label = Calle
catalogo.capillas.edicion.atributo.7.label = N\u00famero Exterior
catalogo.capillas.edicion.atributo.8.label = C\u00f3digo Postal
catalogo.capillas.edicion.atributo.9.label = Tel\u00e9fono 1
catalogo.capillas.edicion.atributo.10.label = Tel\u00e9fono 2
catalogo.capillas.edicion.atributo.11.label = Tel\u00e9fono 3
catalogo.capillas.edicion.atributo.12.label = Fax
catalogo.capillas.edicion.atributo.13.label = Correo Electr\u00f3nico
catalogo.capillas.edicion.atributo.14.label = Id Capilla

catalogo.altaceremonias.titulo = Registro Ceremonias
catalogo.altaceremonias.titulo.file = altaceremonias
catalogo.altaceremonias.edicion.atributo.1.label = Datos Solicitante
catalogo.altaceremonias.edicion.atributo.2.label = Nombre(s):
catalogo.altaceremonias.edicion.atributo.3.label = Apellido Paterno:
catalogo.altaceremonias.edicion.atributo.4.label = Apellido Materno:
catalogo.altaceremonias.edicion.atributo.5.label = Tel\u00e9fono Uno
catalogo.altaceremonias.edicion.atributo.6.label = Tel\u00e9fono Dos
catalogo.altaceremonias.edicion.atributo.7.label = Datos Ceremonia
catalogo.altaceremonias.edicion.atributo.8.label = Tipo
catalogo.altaceremonias.edicion.atributo.9.label = Ceremonia
catalogo.altaceremonias.edicion.atributo.10.label = Intenci\u00f3n
catalogo.altaceremonias.edicion.atributo.11.label = Lugar
catalogo.altaceremonias.edicion.atributo.12.label =
catalogo.altaceremonias.edicion.atributo.13.label =
catalogo.altaceremonias.edicion.atributo.14.label = Servicios
catalogo.altaceremonias.edicion.atributo.15.label = Donativo
catalogo.altaceremonias.edicion.atributo.16.label =
catalogo.altaceremonias.edicion.atributo.17.label =
catalogo.altaceremonias.edicion.atributo.18.label =
catalogo.altaceremonias.edicion.atributo.19.label =
catalogo.altaceremonias.edicion.atributo.20.label =

catalogo.version.titulo = Prueba Registro de Ceremonias
catalogo.version.titulo.file = registroceremonias
catalogo.version.edicion.atributo.1.label = Id Solicitante
catalogo.version.edicion.atributo.2.label = Fecha de Registro
catalogo.version.edicion.atributo.3.label = Fecha de Cancelaci\u00f3n
```

```
catalogo.version.edicion.atributo.4.label = Nombre del Solicitante
catalogo.version.edicion.atributo.5.label = Ceremonia
catalogo.version.edicion.atributo.6.label = Intenci\u00f3n

#textos de vistas
vista.tipo.titulo = Celebraciones Registradas
vista.tipo.titulo.file = celebracionesRegistradas
vista.tipo.atributo.1.label = ID Ceremonia
vista.tipo.atributo.2.label = Folio
vista.tipo.atributo.3.label = Nombre Solicitante
vista.tipo.atributo.4.label = Apellido Paterno
vista.tipo.atributo.5.label = Tel\u00e9fono
vista.tipo.atributo.6.label = Ceremonia
vista.tipo.atributo.7.label = Donativo
vista.tipo.atributo.8.label = Intenci\u00f3n
vista.tipo.atributo.9.label = Parroquia
vista.tipo.atributo.10.label = Capilla
vista.tipo.atributo.11.label = Fecha Celebraci\u00f3n
vista.tipo.atributo.12.label = Hora Celebraci\u00f3n

vista.cancelaciones.titulo = Celebraciones Canceladas
vista.cancelaciones.titulo.file = celebracionesCanceladas
vista.cancelaciones.atributo.1.label = ID Ceremonia
vista.cancelaciones.atributo.2.label = Folio
vista.cancelaciones.atributo.3.label = Nombre Solicitante
vista.cancelaciones.atributo.4.label = Apellido Paterno
vista.cancelaciones.atributo.5.label = Tel\u00e9fono
vista.cancelaciones.atributo.6.label = Ceremonia
vista.cancelaciones.atributo.7.label = Donativo
vista.cancelaciones.atributo.8.label = Intenci\u00f3n
vista.cancelaciones.atributo.9.label = Parroquia
vista.cancelaciones.atributo.10.label = Capilla
vista.cancelaciones.atributo.11.label = Fecha Celebraci\u00f3n
vista.cancelaciones.atributo.12.label = Hora Celebraci\u00f3n

vista.parroquia.titulo = Datos Parroquia
vista.parroquia.titulo.file = datos parroquia
vista.parroquia.atributo.1.label = San Juan Bautista
vista.parroquia.atributo.2.label = Entidad:
vista.parroquia.atributo.3.label = Distrito Federal
vista.parroquia.atributo.4.label = Municipio:
vista.parroquia.atributo.5.label = Coyoac\u00e1n
vista.parroquia.atributo.6.label = Calle:
vista.parroquia.atributo.7.label = Parque Centenario No. 8
vista.parroquia.atributo.8.label = Colonia:
vista.parroquia.atributo.9.label = Coyoac\u00e1n
vista.parroquia.atributo.10.label = C.P.:
vista.parroquia.atributo.11.label = 4000
vista.parroquia.atributo.12.label = Tel\u00e9fono 1:
vista.parroquia.atributo.13.label = [01 55] 55 54 63 10
vista.parroquia.atributo.14.label = Tel\u00e9fono 2:
vista.parroquia.atributo.15.label = [01 55] 55 54 05 60
vista.parroquia.atributo.16.label = Tel\u00e9fono 3:
vista.parroquia.atributo.17.label = [01 55] 55 54 57 46
vista.parroquia.atributo.18.label = Fax:
vista.parroquia.atributo.19.label =
vista.parroquia.atributo.20.label = Correo electronico:
vista.parroquia.atributo.21.label =

vista.ceremonia.titulo = Precios Ceremonias
vista.ceremonia.titulo.file = Precios Ceremonias
vista.ceremonia.atributo.1.label = Id Ceremonia
vista.ceremonia.atributo.2.label = Ceremonia
```

```
vista.ceremonia.atributo.3.label = Tipo
vista.ceremonia.atributo.4.label = Costo $

vista.horario.titulo = Horarios Misas
vista.horario.titulo.file = HorariosMisas
vista.horario.atributo.1.label = Id Horario
vista.horario.atributo.2.label = Parroquia
vista.horario.atributo.3.label = Capilla
vista.horario.atributo.4.label = D\u00edas
vista.horario.atributo.5.label = Horas
vista.horario.atributo.6.label = Horarios Celebraci\u00f3n Eucar\u00edstica de la
Parroquia Santa Mar\u00eda Magdalena

#reportes
reportes.titulo=Reportes
reportes.atributo.1.label = A\u00f1o
reportes.atributo.2.label = Parroquia
reportes.atributo.3.label = Capilla
reportes.atributo.4.label = Fecha Inicio
reportes.atributo.5.label = Fecha Fin
reportes.atributo.6.label = Tipo Reporte

#ventanas
window.alert.title = Mensaje de Alerta
window.guardar.confirmacion.title = Mensaje de Confirmaci\u00f3n
window.guardar.confirmacion.msg.nuevo = \u00bfDesea dar de alta el registro?
window.guardar.confirmacion.msg.modificacion = \u00bfDesea guardar los cambios?
```

Perfil.cnt.jspx

```
<!-- sistema: Parroquial OFM Web -->
<!-- autor: Patricia Roldan -->

<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
            xmlns:ui="http://java.sun.com/jsf/facelets"
            xmlns:f="http://java.sun.com/jsf/core"
            xmlns:h="http://java.sun.com/jsf/html"
            xmlns:icecore="http://www.icefaces.org/icefaces/core"
            xmlns:ace="http://www.icefaces.org/icefaces/components"
            xmlns:ice="http://www.icesoft.com/icefaces/component">

<ui:composition>
            <f:view beforePhase="#{perfilBean.beforePhase}">
              <h:head>
              </h:head>
              <h:body>
                    <!--  -->
              <f:loadBundle basename="config.textos" var="txt" /><f:loadBundle
basename="config.mensajes" var="msg" />
              <ice:panelGrid columns="2" width="100%"
              columnClasses="emptyStyle alignBottom">
                <ice:form id="form" renderedOnUserRole="#{perfilBean.roleAdmin}">
                     <ice:messages globalOnly="true" errorStyle="color: red;"
infoStyle="color: blue;" warnStyle="color: yellow;" />
                          <ice:outputText styleClass="titulo"
                            value="Catálogo de #{txt['catalogo.perfil.titulo']}" />
                               <!-- zona de edicion del elemento -->
                               <ice:panelGrid id="editGrid" columns="1">
                                     <!-- zona de atributos -->
                                     <ice:panelGrid styleClass="" columns="2"
```

```
                                        rendered="#{perfilBean.edicion }">
                                        <ice:outputLabel for="input1"

            value="#{txt['catalogo.perfil.edicion.atributo.2.label']}" />
                    <ice:panelGroup>
                    <ice:inputText id="input1" value="#{perfilBean.item.nombre}"
                          required="true" maxlength="#{perfilBean.item.nombreLenMax}"
                          styleClass="TextBoxSizeM">
                          <f:validator validatorId="validatorNombre" />
                            </ice:inputText>
                    <ice:message errorStyle="color: red;" infoStyle="color: blue;"
                                  warnStyle="color: yellow;" for="input1" />
                                        </ice:panelGroup>
                                        <ice:outputLabel for="input2"

            value="#{txt['catalogo.perfil.edicion.atributo.3.label']}" />
                                  <ice:panelGroup>
                                  <ice:inputText id="input2"

            value="#{perfilBean.item.descripcion}" required="true"

            maxlength="#{perfilBean.item.descripcionLenMax}"
            styleClass="TextBoxSizeL">
                          </ice:inputText>
                    <ice:message errorStyle="color: red;" infoStyle="color: blue;"
                    warnStyle="color: yellow;" for="input2" />
                          </ice:panelGroup>
                          <ice:outputLabel for="inputActivo"
                    value="#{txt['general.activo']}" />
                    <ice:panelGroup>
                    <ice:selectBooleanCheckbox id="inputActivo"
onclick="toggleActivo(this, '#{msg['general.toggle.activo.true']}',
'#{msg['general.toggle.activo.false']}');"

            value="#{perfilBean.item.activo}" />
              <ice:message for="inputActivo" errorStyle="color: red;"
                    infoStyle="color: blue;" warnStyle="color: yellow;" />
                                          </ice:panelGroup>
                                  </ice:panelGrid> 
                                  <!-- zona de perfiles -->

            <ice:outputText styleClass="subtitulo"    value="Funciones"
rendered="#{perfilBean.edicion }"/>

            <ice:panelGrid columns="2" visible="#{perfilBean.edicion }">
            <ace:dataTable id="listPerfilesDataTable"
            value="#{perfilBean.item.permisos}" var="per" height="250"
            paginator="false" styleClass="listDataTableSize"
            scrollable="true">

            <ace:column sortBy="#{per.grupoPagina.nombre}"
            filterBy="#{per.grupoPagina.nombre}"
            filterMatchMode="contains" styleClass="TextLeft"

            headerText="#{txt['catalogo.perfil.edicion.funcion.atributo.1.label']}">
            <ice:outputText value="#{per.grupoPagina.nombre}" />
            </ace:column>
            <ace:column sortBy="#{per.pagina.nombre}"

            filterBy="#{per.pagina.nombre}" filterMatchMode="contains"
            styleClass="TextLeft"

            headerText="#{txt['catalogo.perfil.edicion.funcion.atributo.2.label']}">
```

```
            <ice:outputText value="#{per.pagina.nombre}" />
            </ace:column>
            <ace:column sortBy="#{per.funcion.nombre}"

            filterBy="#{per.funcion.nombre}" filterMatchMode="contains"
            styleClass="TextLeft"

            headerText="#{txt['catalogo.perfil.edicion.funcion.atributo.3.label']}">
            <ice:outputText value="#{per.funcion.nombre}" />
            </ace:column>
            <ace:column

            headerText="#{txt['catalogo.perfil.edicion.funcion.atributo.4.label']}">
            <ice:selectBooleanCheckbox value="#{per.asignado}" />
            </ace:column>
            </ace:dataTable>
            </ice:panelGrid>

            <!-- zona de acciones horizontal-->
            <ice:panelGrid columns="5">
            <ice:commandLink title="#{txt['general.nuevo']}"

            actionListener="#{perfilBean.nuevoItem }" immediate="true">
            <h:graphicImage url="/img/add.png" styleClass="commandImage" />
            </ice:commandLink>
            <ice:commandLink title="#{txt['general.guardar']}"

            actionListener="#{perfilBean.guardarItem }"

            panelConfirmation="#{perfilBean.item.id!=0?'pnlConfirGuardarGral':'pnlConfir
GuardarNvoGral'}"
            disabled="#{!perfilBean.edicion}">
            <ice:graphicImage url="/img/save-dis.png"

            visible="#{!perfilBean.edicion }" styleClass="commandImage" />
            <ice:graphicImage url="/img/save.png"

            visible="#{perfilBean.edicion }" styleClass="commandImage" />
            </ice:commandLink>
            <ice:commandLink title="#{txt['general.cancelar']}"

            actionListener="#{perfilBean.restaurarItem }" immediate="true"
            disabled="#{!perfilBean.edicion}">
            <ice:graphicImage url="/img/cancel-dis.png"

            visible="#{!perfilBean.edicion}" styleClass="commandImage" />
            <ice:graphicImage url="/img/cancel.png"

            visible="#{perfilBean.edicion}" styleClass="commandImage" />
            </ice:commandLink>
            <ice:outputLink title="PDF" styleClass="commandImage"
            rendered="false">
            <h:graphicImage url="/img/pdf.png" styleClass="commandImage" />
            </ice:outputLink>
            <ui:include src="/catalogo/confirmGenerales.jspx" />
            </ice:panelGrid>
            </ice:panelGrid>
            </ice:form>
            <ice:form

            renderedOnUserRole="#{perfilBean.roleAdmin},#{perfilBean.roleConsulta}">
            <!-- tipos excel y cvs -->
              <ice:panelGroup>
```

```
                <div align="right">
                <ace:dataExporter target="listDataTable"

            fileName="#{txt['catalogo.perfil.titulo.file']}_#{constantes.fechaArchivo}"
            type="xls">
            <h:graphicImage url="/img/xls.png" />
            </ace:dataExporter>
            <ace:dataExporter target="listDataTable"

            preProcessor="#{perfilBean.preProcess}" encoding=""

            fileName="#{txt['catalogo.perfil.titulo.file']}_#{constantes.fechaArchivo}"
            type="pdf">
            <h:graphicImage url="/img/pdf.png" />
            </ace:dataExporter>
            </div>
            </ice:panelGroup>
            </ice:form>
            </ice:panelGrid>
            <ice:form

            renderedOnUserRole="#{perfilBean.roleAdmin},#{perfilBean.roleConsulta}">
            <!-- zona de elementos existentes -->
            <ice:panelGrid id="listGrid" columns="1" styleClass="ListGrid">
            <ace:dataTable id="listDataTable" value="#{perfilBean.itemList}"
            var="item" paginator="true"
            paginatorPosition="bottom" rows="#{constantes.rows}"
            selectionMode="single"

            rowSelectListener="#{perfilBean.rowSelectListener}"
              >
            <ace:column sortBy="#{item.nombre}" filterBy="#{item.nombre}"
            filterMatchMode="contains" styleClass="TextLeft"

            headerText="#{txt['catalogo.perfil.edicion.atributo.2.label']}">
            <ice:outputText value="#{item.nombre}" />
            </ace:column>
            <ace:column sortBy="#{item.descripcion}"
            filterBy="#{item.descripcion}" filterMatchMode="contains"
            styleClass="TextLeft"

            headerText="#{txt['catalogo.perfil.edicion.atributo.3.label']}">
            <ice:outputText value="#{item.descripcion}" />
            </ace:column>
            <ace:column sortBy="#{item.activo}"
            filterBy="#{item.activo ? txt['general.activo'] : txt['general.inactivo']}"
            filterMatchMode="startsWith" styleClass="TextLeft"
            headerText="#{txt['general.activo']}">
            <ice:outputText
            value="#{ item.activo ? txt['general.activo'] : txt['general.inactivo']}" />
                                        </ace:column>
                                    </ace:dataTable>
                                </ice:panelGrid>
                            </ice:form>
              </h:body>
            </f:view>
</ui:composition>
</html>
```

Usuariol.cnt.jspx

```
<!-- sistema: Parroquial OFM Web -->
<!-- autor: Patricia Roldan -->
```

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
           xmlns:ui="http://java.sun.com/jsf/facelets"
           xmlns:f="http://java.sun.com/jsf/core"
           xmlns:h="http://java.sun.com/jsf/html"
           xmlns:icecore="http://www.icefaces.org/icefaces/core"
           xmlns:ace="http://www.icefaces.org/icefaces/components"
           xmlns:ice="http://www.icesoft.com/icefaces/component">
<ui:composition>
           <f:view beforePhase="#{usuarioBean.beforePhase}">
             <h:head>
             </h:head>
             <h:body>
             <!--  -->
             <f:loadBundle basename="config.textos" var="txt" /><f:loadBundle
basename="config.mensajes" var="msg" />
             <ice:panelGrid columns="2" width="100%"
               columnClasses="emptyStyle alignBottom">
                 <ice:form id="form" renderedOnUserRole="#{usuarioBean.roleAdmin}">
                   <ice:messages globalOnly="true" errorStyle="color: red;"
infoStyle="color: blue;" warnStyle="color: yellow;" />
                   <ice:outputText styleClass="titulo"
                     value="Catálogo de #{txt['catalogo.usuario.titulo']}" />

                   <!-- zona de edicion del elemento -->
                   <ice:panelGrid id="editGrid" columns="1" columnClasses="col100"
width="99%">
                   <!-- zona de atributos -->
                   <ice:panelGrid columns="1" rendered="#{usuarioBean.edicion }"
columnClasses="col100">
                   <ice:panelGrid columns="4" columnClasses="col20 col30 col20 col30">
                   <ice:outputLabel for="input1"

           value="#{txt['catalogo.usuario.edicion.atributo.1.label']}"
           visible="true" />
           <ice:panelGroup>
           <ice:inputText id="input1" value="#{usuarioBean.item.nombre}"
           styleClass="TextBoxSizeM"
           required="true" maxlength="#{usuarioBean.item.nombreLenMax}">
           <f:validator validatorId="validatorNombrePropio" />
           </ice:inputText>
           <ice:message errorStyle="color: red;" infoStyle="color: blue;"
           warnStyle="color: yellow;" for="input1" />
           </ice:panelGroup>
           <ice:outputLabel for="input4"

           value="#{txt['catalogo.usuario.edicion.atributo.4.label']}" />
             <ice:panelGroup>
             <ice:inputText id="input4" value="#{usuarioBean.item.usuario}"
           required="true" >
             <f:validator validatorId="validatorClave" />
                </ice:inputText>
                   <ice:message errorStyle="color: red;" infoStyle="color: blue;"
           warnStyle="color: yellow;" for="input4" />
             </ice:panelGroup>
               <ice:outputLabel for="input2"

           value="#{txt['catalogo.usuario.edicion.atributo.2.label']}" />
             <ice:panelGroup>
               <ice:inputText id="input2"
```

```
            value="#{usuarioBean.item.apellidoP}" required="true"
maxlength="#{usuarioBean.item.apellidoPLenMax}" >
              <f:validator validatorId="validatorNombrePropio" />
                </ice:inputText>
              <ice:message errorStyle="color: red;" infoStyle="color: blue;"
            warnStyle="color: yellow;" for="input2" />
              </ice:panelGroup>
              <ice:outputLabel for="input5"

            value="#{txt['catalogo.usuario.edicion.atributo.5.label']}" />
              <ice:panelGroup>
                <ice:inputText id="input5" value="#{usuarioBean.item.pwd}"
                  required="true" maxlength="#{usuarioBean.item.pwdLenMax}" />
                    <ice:message errorStyle="color: red;" infoStyle="color: blue;"
            warnStyle="color: yellow;" for="input5" />
              </ice:panelGroup>
                <ice:outputLabel for="input3"

            value="#{txt['catalogo.usuario.edicion.atributo.3.label']}" />
              <ice:panelGroup>
                <ice:inputText id="input3"

            value="#{usuarioBean.item.apellidoM}" required="true"
            maxlength="#{usuarioBean.item.apellidoMLenMax}" >
                  <f:validator validatorId="validatorNombrePropio" />
                    </ice:inputText>
                      <ice:message errorStyle="color: red;" infoStyle="color: blue;"
            warnStyle="color: yellow;" for="input3" />
              </ice:panelGroup>
                <ice:outputLabel for="inputActivo"

            value="#{txt['catalogo.usuario.edicion.atributo.6.label']}" />
              <ice:panelGroup>
                <ice:selectBooleanCheckbox id="inputActivo" onclick="toggleActivo(this,
'#{msg['general.toggle.activo.true']}', '#{msg['general.toggle.activo.false']}');"

            value="#{usuarioBean.item.activo}" />
              <ice:message errorStyle="color: red;" infoStyle="color: blue;"
            warnStyle="color: yellow;" for="inputActivo" />
                </ice:panelGroup>
                  <ice:outputLabel for="input7"

            value="#{txt['catalogo.usuario.edicion.atributo.7.label']}" />
              <ice:panelGroup>
                <ice:selectOneMenu id="input7"

            value="#{usuarioBean.item.area.id}" partialSubmit="true"
              required="true" styleClass="SelectBoxSizeM">
              <f:validator validatorId="validatorComboRequerido" />
              <f:selectItems value="#{usuarioBean.comboArea}" />
              </ice:selectOneMenu>
              <ice:message errorStyle="color: red;" infoStyle="color: blue;"
            warnStyle="color: yellow;" for="input7" />
              </ice:panelGroup>
              </ice:panelGrid>

              <!-- zona de perfiles -->

              <ice:panelGrid columns="2" columnClasses="alignTop alignTop">
              <ice:outputText styleClass="subtitulo"  value="Funciones" />

              <ice:outputText styleClass="subtitulo"  value="Perfiles" />
```

```
            <ace:dataTable id="listFuncionesDataTable"

          value="#{usuarioBean.item.permisos}" var="fun" height="250"
            paginator="false" styleClass="listDataTableSize"
            scrollable="true">

            <ace:column sortBy="#{fun.grupoPagina.nombre}"

          filterBy="#{fun.grupoPagina.nombre}"

          filterMatchMode="contains" styleClass="TextLeft"

          headerText="#{txt['catalogo.usuario.edicion.funcion.atributo.1.label']}">
                                                                <ice:outputText
value="#{fun.grupoPagina.nombre}" />
          </ace:column>

          <ace:column sortBy="#{fun.pagina.nombre}"

          filterBy="#{fun.pagina.nombre}" filterMatchMode="contains"
          styleClass="TextLeft"

          headerText="#{txt['catalogo.usuario.edicion.funcion.atributo.2.label']}">
          <ice:outputText value="#{fun.pagina.nombre}" />
          </ace:column>
          <ace:column sortBy="#{fun.funcion.nombre}"

          filterBy="#{fun.funcion.nombre}" filterMatchMode="contains"
          styleClass="TextLeft"

          headerText="#{txt['catalogo.usuario.edicion.funcion.atributo.3.label']}">
          <ice:outputText value="#{fun.funcion.nombre}" />
          </ace:column>
          <ace:column

          headerText="#{txt['catalogo.usuario.edicion.funcion.atributo.4.label']}">

          <ice:selectBooleanCheckbox value="#{fun.asignado}" partialSubmit="true"

          valueChangeListener="#{usuarioBean.funcionValueChangeListener}">
          <f:attribute name="permiso" value="#{fun}" />

          </ice:selectBooleanCheckbox>
          </ace:column>
            </ace:dataTable>

            <ace:dataTable id="listPerfilesDataTable"

          value="#{usuarioBean.perfilList}" var="per" height="250"
          paginator="false" styleClass="listDataTableSize"
          scrollable="true">

          <ace:column sortBy="#{per.grupoPagina.nombre}"

          filterBy="#{per.grupoPagina.nombre}"

          filterMatchMode="contains" styleClass="TextLeft"

          headerText="#{txt['catalogo.usuario.edicion.perfil.atributo.1.label']}">
          <ice:outputText value="#{per.nombre}" />
          </ace:column>
          <ace:column
```

```
            headerText="#{txt['catalogo.usuario.edicion.perfil.atributo.2.label']}">

            <ice:selectBooleanCheckbox value="#{per.activo}"

            partialSubmit="true"

            valueChangeListener="#{usuarioBean.perfilValueChangeListener}">
            <f:attribute name="perfil" value="#{per}" />

            </ice:selectBooleanCheckbox>
            </ace:column>
            </ace:dataTable>

            </ice:panelGrid>
            </ice:panelGrid>

            <!-- zona de acciones vertical-->
            <ice:panelGrid columns="1" rendered="false">
            <ice:commandButton value="#{txt['general.nuevo']}"

            actionListener="#{usuarioBean.nuevoItem }" immediate="true" />
            <ice:commandButton value="#{txt['general.guardar']}"

            actionListener="#{usuarioBean.guardarItem }"
            disabled="#{!usuarioBean.edicion }" />
            <ice:commandButton value="#{txt['general.cancelar']}"

            actionListener="#{usuarioBean.restaurarItem }" immediate="true"
            disabled="#{!usuarioBean.edicion}" />
              </ice:panelGrid>

            <!-- zona de acciones horizontal-->
              <ice:panelGroup >
              <ice:panelGrid columns="2" width="99%">
              <ice:panelGroup >
            <ice:commandLink title="#{txt['general.nuevo']}"

            actionListener="#{usuarioBean.nuevoItem }" immediate="true">
            <h:graphicImage url="/img/add.png" styleClass="commandImage" />
            </ice:commandLink>
            <ice:commandLink title="#{txt['general.guardar']}"

            actionListener="#{usuarioBean.guardarItem }"

            disabled="#{!usuarioBean.edicion }"

            panelConfirmation="#{usuarioBean.item.id!=0?'pnlConfirGuardarGral':'pnlConfi
rGuardarNvoGral'}">
            <ice:graphicImage url="/img/save-dis.png"

            visible="#{!usuarioBean.edicion }" styleClass="commandImage" />
            <ice:graphicImage url="/img/save.png"

            visible="#{usuarioBean.edicion }" styleClass="commandImage" />
            </ice:commandLink>
            <ice:commandLink title="#{txt['general.cancelar']}"

            actionListener="#{usuarioBean.restaurarItem }" immediate="true"

            disabled="#{!usuarioBean.edicion}">
              <ice:graphicImage url="/img/cancel-dis.png"
```

```
            visible="#{!usuarioBean.edicion}" styleClass="commandImage" />
              <ice:graphicImage url="/img/cancel.png"

            visible="#{usuarioBean.edicion}" styleClass="commandImage" />
              </ice:commandLink>
              </ice:panelGroup>
              <ice:panelGroup>
              <div align="right">

            </div>
            </ice:panelGroup>
            </ice:panelGrid>
            </ice:panelGroup>
            <ui:include src="/catalogo/confirmGenerales.jspx" />
            </ice:panelGrid>

            </ice:form>
            <!-- tipos xls pdf cvs xml -->
            <ice:panelGroup rendered="false">
            <ice:form
renderedOnUserRole="#{usuarioBean.roleAdmin},#{usuarioBean.roleConsulta}">
              <div align="right">
              <ace:dataExporter target="listDataTable"

            fileName="#{txt['catalogo.usuario.titulo.file']}_#{constantes.fechaArchivo}"
            type="xls">
            <h:graphicImage url="/img/xls.png" />
            </ace:dataExporter>
            <ace:dataExporter target="listDataTable"

            preProcessor="#{usuarioBean.preProcess}" encoding=""

            fileName="#{txt['catalogo.usuario.titulo.file']}_#{constantes.fechaArchivo}"
            type="pdf">
            <h:graphicImage url="/img/pdf.png" />
            </ace:dataExporter>
              </div>
                </ice:form>
                  </ice:panelGroup>
                     </ice:panelGrid>

                     <!-- Aqui empieza el grid -->

                     <ice:form

            renderedOnUserRole="#{usuarioBean.roleAdmin},#{usuarioBean.roleConsulta}">
                     <!-- zona de elementos existentes -->
                     <ice:panelGrid id="listGrid" styleClass="ListGrid" columns="1">
                       <ace:dataTable id="listDataTable"
                       value="#{usuarioBean.itemList}"
                       var="item"
                       rows="#{constantes.rows}"
                       paginator="true"
                       paginatorPosition="bottom"
                       selectionMode="single"

            rowSelectListener="#{usuarioBean.rowSelectListener}"
                     styleClass="listDataTableSize">
                     <ace:column sortBy="#{item.nombre}" filterBy="#{item.nombre}"
                     filterMatchMode="contains" styleClass="TextLeft"

            headerText="#{txt['catalogo.usuario.edicion.atributo.1.label']}">
```

```
                    <ice:outputText value="#{item.nombre}" />
                    </ace:column>
                    <ace:column sortBy="#{item.apellidoP}"
                    filterBy="#{item.apellidoP}" styleClass="TextLeft"
                    filterMatchMode="contains"

            headerText="#{txt['catalogo.usuario.edicion.atributo.2.label']}">
                    <ice:outputText value="#{item.apellidoP}" />
                    </ace:column>
                    <ace:column sortBy="#{item.apellidoM}"
                    filterBy="#{item.apellidoM}" styleClass="TextLeft"
                    filterMatchMode="contains"

            headerText="#{txt['catalogo.usuario.edicion.atributo.3.label']}">
                    <ice:outputText value="#{item.apellidoM}" />
                    </ace:column>
                    <ace:column sortBy="#{item.usuario}" filterBy="#{item.usuario}"
                    filterMatchMode="contains" styleClass="TextLeft"

            headerText="#{txt['catalogo.usuario.edicion.atributo.4.label']}">
                    <ice:outputText value="#{item.usuario}" />
                    </ace:column>
                    <ace:column sortBy="#{item.area.nombre}"
filterBy="#{item.area.nombre}"
                    filterMatchMode="contains" styleClass="TextLeft"

            headerText="#{txt['catalogo.usuario.edicion.atributo.7.label']}">
                    <ice:outputText value="#{item.area.nombre}" />
                    </ace:column>
                    <ace:column sortBy="#{item.activo}"
                    filterBy="#{ item.activo ? txt['general.activo'] :
txt['general.inactivo']}"
                    filterMatchMode="startsWith" headerText="#{txt['general.activo']}">
                    <ice:outputText
                    value="#{ item.activo ? txt['general.activo'] :
txt['general.inactivo']}" />
                    </ace:column>
                    </ace:dataTable>
                    </ice:panelGrid>
                    </ice:form>
              </h:body>
            </f:view>
</ui:composition>
</html>
```

Faces-config.xml

```
<!-- sistema: Parroquial OFM Web -->
<!-- autor: Patricia Roldan -->

<!-- de web-if _ lib -->
<?xml version="1.0" encoding="UTF-8"?>

<faces-config
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-facesconfig_2_0.xsd"
    version="2.0">


    <application>
             <!-- Delegar Beans a Spring -->
```

```
            <variable-resolver>
                  org.springframework.web.jsf.DelegatingVariableResolver
            </variable-resolver>
            <!-- mensajes -->
            <locale-config>
                  <default-locale>es_MX</default-locale>
            </locale-config>
            <message-bundle>config.mensajes</message-bundle>
          </application>

          <!-- seguridad, hay que quitar los ejemplos-->

          <managed-bean>
        <managed-bean-name>shoppingBean</managed-bean-name>
        <managed-bean-class>com.icesoft.icefaces.security.beans.ShoppingBean
        </managed-bean-class>
        <managed-bean-scope>request</managed-bean-scope>
    </managed-bean>

    <managed-bean>
        <managed-bean-name>loginBean</managed-bean-name>
        <managed-bean-class>com.icesoft.icefaces.security.beans.LoginBean
        </managed-bean-class>
        <managed-bean-scope>request</managed-bean-scope>
    </managed-bean>

          <navigation-rule>
        <from-view-id>/secured/spofm.jspx</from-view-id>
        <navigation-case>
            <from-outcome>success</from-outcome>
            <to-view-id>/secured/successfulPurchase.jspx</to-view-id>
        </navigation-case>
    </navigation-rule>

<navigation-rule>
        <from-view-id>*</from-view-id>
        <navigation-case>
            <from-outcome>login</from-outcome>
            <to-view-id>/springSecurityLogin.jspx</to-view-id>
        </navigation-case>
    </navigation-rule>

          <!-- seguridad fin -->

          <!-- navegacion catalogos -->

          <navigation-rule>
            <from-view-id>*</from-view-id>
             <navigation-case>
            <from-outcome>direccion</from-outcome>
            <to-view-id>/catalogo/direccion.jspx</to-view-id>
        </navigation-case>
          <navigation-case>
            <from-outcome>area</from-outcome>
            <to-view-id>/catalogo/area.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>usuario</from-outcome>
            <to-view-id>/catalogo/usuario.jspx</to-view-id>
        </navigation-case>

        <navigation-case>
            <from-outcome>instrumentosJerarquias</from-outcome>
```

```
        <to-view-id>/catalogo/instrumentosJerarquias.jspx</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-outcome>instrumentosTipos</from-outcome>
        <to-view-id>/catalogo/instrumentosTipos.jspx</to-view-id>
    </navigation-case>

    <navigation-case>
        <from-outcome>diccionarios</from-outcome>
        <to-view-id>/catalogo/diccionarios.jspx</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-outcome>versionDiccionarios</from-outcome>
        <to-view-id>/catalogo/versionDiccionarios.jspx</to-view-id>
    </navigation-case>

   <navigation-case>
        <from-outcome>usuario</from-outcome>
        <to-view-id>/catalogo/usuario.jspx</to-view-id>
    </navigation-case>

        <navigation-case>
        <from-outcome>parametrosWeb</from-outcome>
        <to-view-id>/catalogo/parametrosWeb.jspx</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-outcome>escprocedencia</from-outcome>
        <to-view-id>/catalogo/escProcedencia.jspx</to-view-id>
    </navigation-case>

     <navigation-case>
        <from-outcome>capilla</from-outcome>
        <to-view-id>/catalogo/capillas.jspx</to-view-id>
    </navigation-case>

     <navigation-case>
        <from-outcome>altaceremonias</from-outcome>
        <to-view-id>/catalogo/altaCeremonias.jspx</to-view-id>
    </navigation-case>

    <navigation-case>
        <from-outcome>idsaplicacion</from-outcome>
        <to-view-id>/catalogo/idsAplicacion.jspx</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-outcome>instrumentosCategorias</from-outcome>
        <to-view-id>/catalogo/instrumentosCategoria.jspx</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-outcome>instrumentos</from-outcome>
        <to-view-id>/catalogo/instrumentos.jspx</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-outcome>instrumentosSubCategorias</from-outcome>
        <to-view-id>/catalogo/instrumentosSubCategoria.jspx</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-outcome>opcionesaplicacion</from-outcome>
        <to-view-id>/catalogo/opcionesaplicacion.jspx</to-view-id>
    </navigation-case>

    <navigation-case>
        <from-outcome>instrumentosPropietarios</from-outcome>
```

```
            <to-view-id>/catalogo/instrumentosPropietarios.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>examen</from-outcome>
            <to-view-id>/catalogo/examen.jspx</to-view-id>
        </navigation-case>

        <navigation-case>
            <from-outcome>perfil</from-outcome>
            <to-view-id>/catalogo/perfil.jspx</to-view-id>
        </navigation-case>
    </navigation-rule>

          <!-- navegacion catalogos fin -->

          <!-- navegacion vistas -->

          <navigation-rule>
            <from-view-id>*</from-view-id>
             <navigation-case>
            <from-outcome>institucion</from-outcome>
            <to-view-id>/vista/institucion.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>clasificacionAplicacion</from-outcome>
            <to-view-id>/vista/clasificacionaplicacion.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>entidades</from-outcome>
            <to-view-id>/vista/entidades.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>formaEntregaResultados</from-outcome>
            <to-view-id>/vista/formaentregaresultados.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>formaEnvioResultados</from-outcome>
            <to-view-id>/vista/formaenvioresultados.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>formaRegistro</from-outcome>
            <to-view-id>/vista/formaregistro.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>formaRegistro</from-outcome>
            <to-view-id>/vista/formaRegistro.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>modalidadExamen</from-outcome>
            <to-view-id>/vista/modalidadExamen.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
         <from-outcome>tiempoEntregaResultados</from-outcome>
           <to-view-id>/vista/tiempoEntregaResultados.jspx</to-view-id>
           </navigation-case>
           <navigation-case>
         <from-outcome>tipoAplicacion</from-outcome>
           <to-view-id>/vista/tipoAplicacion.jspx</to-view-id>
           </navigation-case>
           <navigation-case>
         <from-outcome>tipoCalificacion</from-outcome>
           <to-view-id>/vista/tipoCalificacion.jspx</to-view-id>
           </navigation-case>
```

```
          <navigation-case>
         <from-outcome>municipio</from-outcome>
          <to-view-id>/vista/municipio.jspx</to-view-id>
          </navigation-case>
          <navigation-case>
         <from-outcome>campus</from-outcome>
          <to-view-id>/vista/campus.jspx</to-view-id>
          </navigation-case>
          </navigation-rule>

          <!-- navegacion vistas fin -->

          <!-- navegacion reportes -->
          <navigation-rule>
                   <from-view-id>*</from-view-id>
             <navigation-case>
           <from-outcome>reportes</from-outcome>
           <to-view-id>/reportes/reportes.jspx</to-view-id>
        </navigation-case>
        <navigation-case>
            <from-outcome>reportesEjemplo</from-outcome>
            <to-view-id>/reportes/ejemplo1.jspx</to-view-id>
        </navigation-case>
<navigation-case>
                <from-outcome>reportesEjemplo2</from-outcome>
            <to-view-id>/reportes/ejemplo2.jspx</to-view-id>
        </navigation-case>
    </navigation-rule>
          <!-- navegacion reportes fin -->

          <!-- navegacion recepcion -->

          <navigation-rule>
            <from-view-id>*</from-view-id>

        <navigation-case>
            <from-outcome>recepciondia</from-outcome>
            <to-view-id>/recepcion/recepcionDia.jspx</to-view-id>
        </navigation-case>
          </navigation-rule>

        <!-- pruebas -->
    <navigation-rule>
          <from-view-id>*</from-view-id>
        <navigation-case>
            <from-outcome>prueba</from-outcome>
            <to-view-id>/catalogo/prueba.jspx</to-view-id>
        </navigation-case>
     </navigation-rule>
          <!-- navegacion recepcion fin -->

          <!-- conversores y validadores -->

          <!-- todos los strings se quitan espacios iniciales y finales -->
          <converter>
          <converter-id>trimConverter</converter-id>
          <converter-class>mx.edu.ofm.spofm.converter.TrimConverter</converter-class>
          </converter>

          <validator>
              <validator-id>validatorInteger</validator-id>
            <validator-class>mx.edu.ofm.spofm.validator.ValidatorInteger</validator-
class>
```

```
            </validator>
            <validator>
                <validator-id>validatorPositivo</validator-id>
              <validator-class>mx.edu.ofm.spofm.validator.ValidatorPositivo</validator-
class>
            </validator>
            <validator>
                <validator-id>validatorDouble</validator-id>
              <validator-class>mx.edu.ofm.spofm.validator.ValidatorDouble</validator-
class>
            </validator>
            <validator>
                <validator-id>validatorClave</validator-id>
              <validator-class>mx.edu.ofm.spofm.validator.ValidatorClave</validator-
class>
            </validator>
            <validator>
                <validator-id>validatorNombrePropio</validator-id>
              <validator-
class>mx.edu.ofm.spofm.validator.ValidatorNombrePropio</validator-class>
            </validator>
            <validator>
                <validator-id>validatorNombre</validator-id>
              <validator-class>mx.edu.ofm.spofm.validator.ValidatorNombre</validator-
class>
            </validator>
            <validator>
                <validator-id>validatorComboRequerido</validator-id>
              <validator-
class>mx.edu.ofm.spofm.validator.ValidatorComboRequerido</validator-class>
            </validator>
            <validator>
                <validator-id>validatorRango</validator-id>
              <validator-class>mx.edu.ofm.spofm.validator.ValidatorRango</validator-
class>
            </validator>

            <!-- conversores y validadores fin -->

</faces-config>
```

springSecurityLogin.jspx

```
<!-- sistema: Parroquial OFM Web -->
<!-- autor: Patricia Roldan -->

<!-- de vista -->
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
            xmlns:ui="http://java.sun.com/jsf/facelets"
            xmlns:f="http://java.sun.com/jsf/core"
            xmlns:h="http://java.sun.com/jsf/html"
            xmlns:icecore="http://www.icefaces.org/icefaces/core"
          xmlns:ace="http://www.icefaces.org/icefaces/components"
            xmlns:ice="http://www.icesoft.com/icefaces/component">


<ui:composition template="./plantilla/plantilla.jspx">
              <ui:define name="content">

<f:view>
```

```
    <h:head>
        <title>SPOFM Login</title>
        <ice:outputStyle href="/css/capas/layout_principal.css"/>
        <!-- <ice:outputStyle href="/css/rime/rime.css"/>
               <ice:outputStyle href="/css/capas/layout.css"/>
               <ice:outputStyle href="/css/estilos.css"/>  -->



               <script type="text/javascript"><!--


           function oamSetHiddenInput(formname, name, value)
           {
             var form = document.forms[formname];
             if (typeof form == 'undefined')
             {
                    form = document.getElementById(formname);
             }

             if(typeof form.elements[name]!='undefined' &&
(form.elements[name].nodeName=='INPUT' || form.elements[name].nodeName=='input'))
             {
                    form.elements[name].value=value;
             }
             else
             {
                    var newInput = document.createElement('input');
                    newInput.setAttribute('type','hidden');
                    newInput.setAttribute('id',name);
                    newInput.setAttribute('name',name);
                    newInput.setAttribute('value',value);
                    form.appendChild(newInput);
             }

           }


           function oamClearHiddenInput(formname, name, value)
           {
             var form = document.forms[formname];
             if (typeof form == 'undefined')
             {
                    form = document.getElementById(formname);
             }

             var hInput = form.elements[name];
             if(typeof hInput !='undefined')
             {
                    form.removeChild(hInput);
             }

           }

           function oamSubmitForm(formName, linkId, target, params)
           {

             var clearFn = 'clearFormHiddenParams_'+formName.replace(/-/g,
'\$:').replace(/:/g,'_');
             if(typeof window[clearFn] =='function')
             {
                    window[clearFn](formName);
```

```
}

if(typeof window.getScrolling!='undefined')
{
	oamSetHiddenInput(formName,'autoScroll',getScrolling());
}

var form = document.forms[formName];
if (typeof form == 'undefined')
{
	form = document.getElementById(formName);
}

var oldTarget = form.target;
if(target != null)
{

	form.target=target;
}
if((typeof params!='undefined') && params != null)
{

	for(var i=0, param; (param = params[i]); i++)
	{
		oamSetHiddenInput(formName,param[0], param[1]);
	}

}

oamSetHiddenInput(formName,formName +':'+'_idcl',linkId);

if(form.onsubmit)
{
	var result=form.onsubmit();
	if((typeof result=='undefined')||result)
	{
		try
		{
			form.submit();
		}
		catch(e){}
	}

}
else
{
	try
	{
		form.submit();
	}
	catch(e){}
}

form.target=oldTarget;
if((typeof params!='undefined') && params != null)
{

	for(var i=0, param; (param = params[i]); i++)
	{
		oamClearHiddenInput(formName,param[0], param[1]);
	}

}
```

```
            oamClearHiddenInput(formName,formName +':'+'_idcl',linkId);return false;
          }


//--></script>

          <script type="text/javascript"><!--

          function getScrolling()
          {
            var x = 0; var y = 0;if (self.pageXOffset || self.pageYOffset)
            {
                  x = self.pageXOffset;
                  y = self.pageYOffset;
            }
             else if ((document.documentElement &&
document.documentElement.scrollLeft)||(document.documentElement &&
document.documentElement.scrollTop))
            {
                  x = document.documentElement.scrollLeft;
                  y = document.documentElement.scrollTop;
            }
             else if (document.body)
            {
                  x = document.body.scrollLeft;
                  y = document.body.scrollTop;
            }
            return x + "," + y;
          }

//--></script>




    </h:head>
    <h:body>
    <ice:panelGrid columns="1" width="100%">
               <ice:panelGroup>
                 <div align="center" style="height: 100px; vertical-align: bottom;">
                 </div>
                 <div align="center" style="height: 150px; vertical-align: bottom;">
                   <ice:form partialSubmit="false" id="loginForm" prependId="false">
                           <ice:panelGrid columns="2">
                               <ice:outputLabel value="Usuario" for="j_username"/>
                               <ice:inputText id="j_username"
                                             value="#{loginBean.userId}" size="40"
                                             maxlength="80"/>
                               <ice:outputLabel value="Contraseña" for="j_password"/>
                               <ice:inputSecret id="j_password"
                                               value="#{loginBean.password}"
size="40"
                                               maxlength="80"/>
                           </ice:panelGrid>
                                <ice:commandButton actionListener="#{loginBean.login}"
value="Ingresar" type="submit" onclick="if(typeof
window.getScrolling!='undefined'){oamSetHiddenInput('loginForm','autoScroll',getScrolli
ng());}"/>
                               <ice:messages globalOnly="true" errorStyle="color:
red;" infoStyle="color: blue;" warnStyle="color: yellow;"/>
```

```
                    </ice:form>
                </div>
            </ice:panelGroup>
            <!-- <ice:graphicImage url="/img/cancel.png" visible ="false"/>
            <ice:graphicImage url="/img/save.png" visible="false"/> -->
            <ice:panelGroup>
              <div align="center" style="height: 100px;" >
                    <ice:graphicImage url="./img/dpoc.png"></ice:graphicImage>
              </div>
            </ice:panelGroup>
    </ice:panelGrid>

    </h:body>
</f:view>
</ui:define>
</ui:composition>

</html>
```