UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA: INGENIERÍA DE SISTEMAS

Tesis previa a la obtención del título de: INGENIERO EN SISTEMAS

TEMA:

ANÁLISIS, DISEÑO, CONSTRUCCIÓN E IMPLEMENTACIÓN DEL MÓDULO DE ANEXOS DEL S.R.I Y GESTIÓN INTEGRADORA DE LOS MÓDULOS DEL SISTEMA CONTABLE PARA LA CASA DE INSPECTORÍA SALESIANA

AUTORES:

BYRON ALEJANDRO ANDRANGO CORREA

DIEGO FERNANDO JÁCOME GUAYASAMIN

DIRECTOR:

DANIEL GIOVANNY DÍAZ ORTIZ

Quito, junio de 2013

DECLARATORIA DE RESPONSABILIDAD Y AUTORIZACIÓN DE USO DELTRABAJO DE GRADO

Nosotros Byron Alejandro Andrango Correa y Diego Fernando Jácome Guayasamín autorizamos a la Universidad Politécnica Salesiana la publicación total o parcial de este trabajo de grado y su reproducción sin fines de lucro.

Además declaramos que los conceptos y análisis desarrollados y las conclusiones del presente trabajo son de exclusiva responsabilidad de los autores.

BYRON ALEJANDRO ANDRANGO CORREA
CC: 1712329661

DIEGO FERNANDO JÁCOME GUAYASAMIN
CC: 1719020883

DEDICATORIA

Dedico este trabajo a Dios, a mis padres y a mi familia
por ser mi apoyo incondicional durante toda mi vida.

Byron Andrango

Dedico este trabajo a toda mi familia y especialmente a mi madre por estar siempre a mi lado y brindarme siempre su sabiduría.

Diego Jácome

AGRADECIMIENTO

Agradecemos a todas las personas que directa o indirectamente han colaborado con este trabajo, a todos los docentes de la carrera de sistemas a nuestro tutor de tesis y a la Universidad Politécnica Salesiana.

Byron Andrango, Diego Jácome

ÍNDICE

ÍNDICES DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El trabajo muestra la investigación de la arquitectura, los componentes, la configuración e instalación del framework y del Sistema Contable Financiero. Además del desarrollo del Módulo del Sistema, el cual es el encargado de la integración y administración de la aplicación también se implementó el Módulo de Anexos y Formularios del S. R. I con sus respectivos manuales de usuario. Se realizaron pruebas de funcionamiento en el sistema para poder medir el rendimiento de la aplicación en condiciones extremas. Con la implementación del framework en el desarrollo de la aplicación web se tiene la facilidad de que el desarrollador no necesite tener experiencia en el desarrollo con el framework JSF, JavaScript, AJAX y conocimientos en programación HTML ahorrando tiempo y esfuerzo.

ABSTRACT

The research work shows the architecture, components, configuration and installation of the framework and the Financial Accounting System. Besides the development of the System Module, which is responsible for the integration and application management also implemented the Annexes and Forms module S. R. I with their respective user manuals. Operation tests were conducted on the system to measure the performance of the application in extreme conditions. With the implementation of the framework in the development of the web application has the facility to which the developer does not need to have experience in developing the JSF framework, JavaScript, AJAX and HTML programming knowledge saving time and effort.

INTRODUCCIÓN

La Casa de Inspectoría Salesiana ubicada en la Calle Madrid E12-68 y Andalucía - Quito, se encuentra la Administración Financiera de la Sociedad Salesiana en el Ecuador. En la actualidad cuenta con un solo sistema contable SITAC, el cual no satisface las necesidades del personal contable; por lo tanto nace la solución de desarrollar un sistema contable financiero para las Casas y Obras de la Inspectoría Salesiana "Sagrado Corazón de Jesús", cumpliendo con todos los requerimientos legales aplicados en el Ecuador, basados en normas y principios contables generalmente aceptados, aplicando un diseño modular e integrado con una operatividad altamente eficiente y fácil de utilizar.

La Inspectoría Salesiana "Sagrado Corazón de Jesús" nace el 31 de julio de 1973 gracias al Decreto del Rector Mayo. Es producto de la unificación de la Inspectoría de Quito con la de Cuenca; es así que se crea la Inspectoría del Ecuador con sede en Quito. Actualmente existen 26 casas salesianas y 173 salesianos: 129 sacerdotes; 16 coadjutores; 20 salesianos clérigos y 8 novicios. Los Salesianos de Don Bosco (SDB) forman una comunidad dedicada a la realización de una vida religiosa, ser el proyecto apostólico del Fundador, ser en la Iglesia signos y portadores del amor de Dios a los jóvenes, especialmente a los más pobres.

El crecimiento de la Obra Salesiana y la diversidad de actividades que se desarrollan en la misma, ha obligado a mejorar la gestión económica, para lo cual la institución requiere un nuevo sistema contable en un entorno Web. En cuanto al módulo de Anexos del SRI se ha resuelto enunciar los siguientes problemas que se han presentado al no contar con un sistema contable integral, que permita llevar un control adecuado de las actividades diarias y a largo plazo. El tiempo excesivo y el doble trabajo que conlleva el proceso actual para generar los formularios 103, 104 y los anexos ATS (Anexo Transaccional Simplificado) REOC (Anexo de Retenciones en la fuente de Impuesto a la Renta por otros conceptos) y RDEP (Anexo de Retenciones en la fuente por relación de dependencia) , debido a que se utiliza una aplicación independiente, porque el sistema actual no cuenta con la funcionalidad de generar los archivos XML compatibles para poder realizar las declaraciones a través

del sitio web que ofrece el SRI. La necesidad de generar automáticamente los formularios y los anexos del SRI utilizando un solo sistema, por lo que es prioridad contar con un único sistema que integre y gestione toda la contabilidad.

La Casa de Inspectoría Salesiana se encuentra desarrollando una serie de proyectos e implementando nuevas tecnologías de información, con el principal objetivo de automatizar las tareas de Contabilidad y hacer más fácil el acceso a la información. El desarrollo del sistema informático contable se podrá realizar pensando en un enfoque integral en cuanto se refiere al manejo de los Inventarios, Facturación, Préstamos e Inversiones y Anexos del SRI, ya que estos módulos son indispensables para realizar un proceso contable organizado y eficiente. Es fundamental que la información sea confiable, eficaz y clara, para el desarrollo de actividades que tiene cada usuario del sistema, realizando los procesos de manera automatizada en un corto tiempo, lo que permitirá potencializar la administración. La Casa de Inspectoría Salesiana requiere implementar un sistema contable en un entorno Web, orientado a sus diversas actividades y necesidades.

Para el proceso de la investigación se recopiló requisitos de usuario, documentos, y cualquier tipo de información que sea útil para el desarrollo del sistema, los cuales serán analizados, para que el resultado sea plasmado en diagramas y diseños que facilitarán la construcción e implementación del sistema. En cuanto a la creación del framework se pretende disminuir la generación de código fuente, pensando siempre en los conceptos de reutilización y herencia, lo cual permitirá optimizar la programación de los módulos que contendrá el sistema contable.

Los objetivos que nacieron de la necesidad de tener un sistema contable se presentaron al analizar, diseñar, construir e implementar el módulo de Anexos del SRI y gestionar la integración de los módulos del Sistema Contable para la Casa de Inspectoría Salesiana, de tal manera que permita el registro y control sistemático de todas las operaciones financieras. Analizar las necesidades y parámetros a tener en cuenta dentro de un sistema contable.

Investigar los procesos para generar los formularios y anexos del Servicio de Rentas Internas en base a la regularización vigente. Aplicar la metodología UP para el desarrollo del proyecto, la cual permite generar los diagramas y diseños del sistema definiendo estándares para el diseño de la base de datos y para la programación de los módulos. Se desarrolló un framework en lenguaje Java a partir de la tecnología Primefaces para gestionar e integrar los módulos del sistema contable; capaz de generar los formularios 103, 104 y anexos REOC, ATS y RDEP según la regularización que mantenga el Servicio de Rentas Internas.

CAPÍTULO 1
MARCO TEÓRICO

1.1. Framework

Existen en el mercado distintos servidores de aplicaciones licenciados y OpenSource que resuelven el problema de una arquitectura de tipo enterprise. Por ejemplo hay varias casas de desarrollo que implementan la especificación JEE basadas en lenguaje Java, la cuales presentan la posibilidad de ser utilizadas de distinta forma, por ejemplo:

- EJB en la modalidad BMP (Bean Managed Persistence).
- CMP (Component Managed Persistence).
- Session Bean Stateless
- Session Bean Statefull.

Estas plataformas permiten además la integración con sistemas de terceros para alguna de sus capas, por ejemplo la de persistencia, entre ellas se puede mencionar:

- BEA WebLogic.
- IBM Websphere.
- Oracle 9i AS.
- JBoss.

1.2. Regularización del SRI vigente

Regularización es el proceso que permite a una persona cumplir los requisitos que señalan las leyes que el Estado imponga y así poder ejercer actividades dentro de la ley implantada. En el Ecuador la entidad regularizadora es El Servicio de Rentas Internas (SRI) la cual es una entidad técnica y autónoma que tiene la responsabilidad de recaudar los tributos internos establecidos por Ley mediante la aplicación de la normativa vigente. Su finalidad es la de consolidar la cultura tributaria en el país a

efectos de incrementar sostenidamente el cumplimiento voluntario de las obligaciones tributarias por parte de los contribuyentes.

1.3. Formularios del SRI

Para el desarrollo de la aplicación contable para La Casa de Inspectoría Salesiana se puede deducir que requiere generar automáticamente los siguientes formularios: Formulario 101 (Instructivo declaración del impuesto a la renta y presentación de balances). Permite seleccionar el mes al que corresponde la declaración, de ser necesaria la presentación de un formulario semestral a través del cual, se debe de seleccionar el mes de junio para la declaración del primer semestre y el mes de diciembre para declaración del segundo semestre.

El impuesto a la renta se grava sobre los ingresos, la ley tributaria permite que existan deducciones sobre estos ingresos, por tanto, el gravamen recae sobre la utilidad de las actividades económicas de los sujetos pasivos. Las deducciones que se pueden utilizar son costos y gastos que se han realizado con el objetivo de obtener, mantener o mejorar los ingresos sujetos al pago del impuesto a la renta.

Los gastos personales son los efectuados por el propio contribuyente, su esposa o cónyuge e hijos menores de edad o discapacitados, siempre y cuando que no posean ingresos gravados con el impuesto a la renta. Serán deducibles hasta un máximo del 50% del ingreso gravado, sin que dicho 50% sea mayor a 1,3 veces la fracción gravada con tarifa 0% del impuesto a la renta.

1.3.1. Formulario 103 (Retenciones en la Fuente del Impuesto a la Renta)

Permite a la persona jurídica o persona natural obligada a llevar contabilidad que pague o acredite en cuenta cualquier tipo de ingreso que constituya renta gravada para quien los reciba, actuará como agente de retención del Impuesto a la Renta. Los agentes de retención están obligados a entregar el respectivo comprobante de retención, dentro del término no mayor de cinco días de

recibido el comprobante de venta, a las personas a quienes deben efectuar la retención. Igualmente están obligados a proporcionar al SRI cualquier tipo de información vinculada con las transacciones por ellos efectuadas. Los pagos que realicen los empleadores a los contribuyentes que trabajan con relación de dependencia, deberán realizar la retención en la fuente correspondiente. En este caso, el comprobante de retención será entregado dentro del mes de enero de cada año en relación con las rentas del año precedente. Así mismo, están obligados a declarar y depositar mensualmente los valores retenidos en las entidades legalmente autorizadas para recaudar tributos. (SRI, 2012)

1.3.2. Formulario 104 (Impuesto al Valor Agregado)

Permite sustituir las declaraciones sustitutivas que pueden ser presentadas cuando tales correcciones originen un mayor valor a pagar, cuando no se modifique el impuesto a pagar o implique diferencias a favor del contribuyente.

El Impuesto al Valor Agregado (IVA) es uno de los impuestos de mayor recaudación para el Estado; es un impuesto indirecto sobre el consumo, que se genera en todas las etapas de comercialización y por su naturaleza debe ser pagado por los consumidores finales. Grava a la transferencia de dominio de bienes muebles, de los derechos de autor, de propiedad industrial y derechos conexos, así como a la prestación de servicios. La Ley de Régimen Tributario Interno y su Reglamento de aplicación son las normas que regulan este impuesto.

El producto de las recaudaciones por el impuesto al valor agregado se depositará en la cuenta del Servicio de Rentas Internas que, para el efecto, se abrirá en el Banco Central del Ecuador. Luego de efectuados los respectivos registros contables, los valores se transferirán en el plazo máximo de 24 horas a la Cuenta Corriente Única del Tesoro Nacional para su distribución a los partícipes. (SRI, 2012)

1.3.3. Objeto del Impuesto al Valor Agregado

El Impuesto al Valor Agregado (IVA), grava al valor de la transferencia de dominio o a la importación de bienes muebles de naturaleza corporal, en todas sus etapas de comercialización, así como a los derechos de autor, de propiedad industrial, derechos conexos; y al valor de los servicios prestados, en la forma y en las condiciones que prevé la Ley de Régimen Tributario Interno y su reglamento, también pudiendo realizar esta acción registrando en el campo correspondiente del formulario de la declaración que se va a sustituir.

1.4. Anexos del SRI

La Administración Tributaria requiere para efectos de control que los contribuyentes presenten información adicional a las declaraciones de impuestos. Esta información recibe el nombre de anexo, cuya finalidad es proporcionar a nivel de detalle la información que sustenta las declaraciones de impuestos u otra información relevante.

Estos anexos deben ser presentados por los contribuyentes de acuerdo al tipo de actividad que realizan. Actualmente el SRI solicita los siguientes anexos:

- Anexo Transaccional Simplificado (ATS)
- Anexo de Retenciones en la fuente de Impuesto a la Renta por otros conceptos (REOC)
- Anexo de Retenciones en la fuente por relación de dependencia (RDEP)

- **Anexo Transaccional Simplificado (ATS)**

  De acuerdo a la Resolución NAC-DGER-2007-1319 los contribuyentes deben presentar un reporte detallado de las transacciones correspondientes a compras, ventas, exportaciones y retenciones de IVA, y de Impuesto a la Renta. Los contribuyentes que deben presentar esta información son los siguientes:

- Contribuyentes Especiales,
- Instituciones del Sector Público,
- Autoimpresores,
- Quienes soliciten devoluciones de IVA, (excepto tercera edad y discapacitados)
- Instituciones Financieras
- Emisoras de tarjetas de crédito
- Administradoras de Fondos y Fideicomisos.

Si no se genera ningún tipo de movimiento para un determinado mes, no se tendrá la obligación de presentar el anexo.

La información se entrega en un medio magnético, a través de un archivo comprimido en formato xml. (SRI, 2012)

- **Anexo de Retenciones en la fuente de Impuesto a la Renta por otros conceptos (REOC)**

La Resolución NAC-DGER-2007-1319 señala que deben presentar la información mensual relativa a las compras o adquisiciones detalladas por comprobante de venta y retención, y los valores retenidos en la Fuente de Impuesto a la Renta por Otros Conceptos lo siguientes contribuyentes:

- Las sociedades
- Personas naturales obligadas a llevar contabilidad (que no tengan la obligación de presentar el ATS).

Si no se genera ningún tipo de movimiento para un determinado mes, no tendrá la obligación de presentar el anexo en mención. En caso de que exista error en la información presentada mediante el anexo, el contribuyente deberá presentar una sustitutiva de esta información. La información se entrega en un medio magnético, a través de un archivo comprimido en formato XML (SRI, 2012).

- **Anexo de Retenciones en la fuente por relación de dependencia (RDEP)**

  De acuerdo a la Resolución NAC-DGER2006-0791 todas las sociedades y empleadores en su calidad de agentes de retención deben presentar un reporte detallado de los pagos y retenciones en la fuente de impuesto a la renta realizadas. La información se entrega en un medio magnético, a través de un archivo comprimido en formato XML (SRI, 2012).

1.5. Integración en Sistemas Contables

Un sistema de información contable es el encargado de recoger, almacenar, tratar los estados financieros que son utilizados por el personal encargado en la toma de decisiones. Un sistema de información contable es generalmente un método computarizado que ayuda con el seguimiento de la actividad contable en relación con los recursos de tecnología de la información. Inicialmente los sistemas de información de contabilidad se han desarrollado en su mayor parte in-house. Estas soluciones son difíciles de desarrollar y caros de mantener, hoy en día, los sistemas de información contable son más vendidos como paquetes de software pre construidos por proveedores como Microsoft, Grupo Sage, SAP y Oracle en el que está configurado y personalizado para que realice los procesos de negocio en la organización.

Como la necesidad de conectividad en tiempo real, la consolidación entre otros sistemas del negocio, el poder acceder desde cualquier lugar de la empresa teniendo un navegador web, nacen los sistemas contables web. Con el nacimiento de las nuevas tecnologías en sistemas contables web se crea otra necesidad la cual es poder integrar sus diversos módulos entre si dando como resultado un sistema web robusto capaz de solucionar y facilitar las tareas contables ahorrando tiempo, esfuerzo y dinero a las personas o empresas.

Para la integración de todo nuestro sistema contable se lo hace mediante el Módulo Sistema el cual está a cargo de unir cada una de las funcionales del Sistema Financiero Contable y así poder ver reflejada la información consolidada con la generación de cada uno de los formularios y anexos en el Módulo de Anexos del SRI. Para poder realizar el proceso de integración se utiliza El Modulo Sistema el cual utiliza la Base de Datos con las tablas de los asientos contables, en las cuales se guardan los datos que servirán para la generación de declaraciones de cada uno de los formularios, anexos que se necesite enviar al S. R. I y reportes en cada una de las funcionalidades del sistema.

Para el proceso de desarrollo de todo el sistema es necesario contar con un software de versionamiento (Visual SVN para Netbeans), ya que esta permitirá que todos los grupos de tesis cuenten con el mismo código fuente permitiendo realizar modificaciones y controlar los avances del proyecto, teniendo una eficiente programación en paralelo. En capítulos posteriores se especificara más con más detalle sobre la integración del sistema contable.

1.6. Metodología UP

Para el desarrollo del software se utilizará UP cómo metodología para el desarrollo del sistema y UML (Lenguaje de Modelamiento Unificado), ya que están estrechamente relacionados entre sí, pues mientras el primero establece las actividades y los criterios para conducir un sistema desde su máximo nivel de abstracción (la idea en la cabeza del cliente), hasta su nivel más concreto (un programa ejecutándose en las instalaciones del cliente), el segundo ofrece la notación gráfica necesaria para representar los sucesivos modelos que se obtienen en el proceso de refinamiento. La metodología de UP es un método iterativo de diseño de software que describe cómo desarrollar software de forma eficaz, utilizando técnicas probadas en la industria.

El Proceso Unificado de Desarrollo de Software o simplemente Proceso Unificado es un marco de desarrollo de software que se caracteriza por estar dirigido por casos de

uso, centrado en la arquitectura, enfocado en el riesgo, y por ser iterativo e incremental. El Proceso Unificado no es simplemente un proceso, sino un marco de trabajo extensible que puede ser adaptado a organizaciones o proyectos específicos.
El nombre Proceso Unificado se usa para describir el proceso genérico que incluye aquellos elementos que son comunes a la mayoría de los refinamientos existentes. Es una metodología orientada a conducir el proceso de desarrollo de software en sus aspectos técnicos; los flujos y productos de trabajo de UP no incluyen la administración del proyecto. UP es una versión libre y abierta del modelo propuesto por Jacobson, Booch y Rumbaugh. Divide el trabajo de desarrollo de software en cuatro fases: inicio, elaboración, construcción y transición, las cuales se describen a continuación.

Fase de Inicio en UP

En esta fase corresponde definir el negocio. Es la etapa donde se define la factibilidad del proyecto a realizar, se representa el modelo de negocio, visión y metas del proyecto, se identifican actores, conceptos de dominio y deseos de usuario. Adicionalmente se complementa con la definición de la arquitectura preliminar, y estimaciones (imprecisas, preliminares) de plazos y costos. También se define la viabilidad del proyecto.

Fase de Elaboración en UP

En la fase de elaboración se obtiene la visión refinada del proyecto a realizar, la implementación iterativa del núcleo central de la aplicación, la resolución de los riesgos más altos, la identificación de nuevos requisitos y nuevos alcances, y estimaciones más ajustadas. A esta altura existe la posibilidad de detener el proyecto por complejidad técnica.

Fase de Construcción en UP

La fase de construcción es la implementación iterativa del resto de los requisitos de menor riesgo y elementos más sencillos. Es la evolución hasta convertirse en un producto listo, incluyendo todos los requisitos (100%), para entregarse al Cliente. Al final de esta fase el sistema contiene todos los casos de uso que el cliente y la dirección del proyecto han acordado. La mayoría de los casos de uso que no se desarrollaron en la fase anterior se desarrollan en iteraciones, en grupos de requisitos o casos de uso durante esta fase.

Organización de disciplinas según UP

Las disciplinas identificadas son modelado de: negocios, requisitos, análisis, diseño, implementación y pruebas, como también se identifican las disciplinas de apoyo, tales como: configuración y manejo de proyectos. Todas estas disciplinas son representadas con su correspondiente esfuerzo estimado para cada una de las fases definidas por UP.

El desarrollo de software iterativo e incremental corresponde a mantener permanentemente un enfoque de cambio en los proyectos de desarrollo. Los llamados ciclos por fases intentan poner en manos del usuario un sistema con prestaciones parciales, que se va completando con nuevas prestaciones en fases sucesivas. Así, el usuario tiene en producción algunas funcionalidades mientras se van desarrollando las otras. Por lo tanto, existen al menos dos sistemas funcionando en paralelo.

El sistema en desarrollo (la siguiente versión) que está siendo preparada para reemplazar la versión en producción, que puede aún conservar partes de implementaciones anteriores o faltarle funcionalidades.

1.7. Aplicaciones web

Para el desarrollo de la aplicación se decidió que era la necesidad de hacer una aplicación web con lo cual se puede tener una aplicación distribuida y que pueda ser utilizada dentro de cualquier sistema operativo brindando todas las características que posee una aplicación web. Una aplicación basada en navegadores los cuales son programas que puedan funcionar a través de un navegador de internet, es decir, son aplicaciones que se ejecutan de forma online. Por qué se escogió una aplicación web y no una aplicación normal de escritorio, pues como se dijo anteriormente la aplicación web se ejecutan de forma online, a diferencia de las aplicaciones offline de escritorio es que es una aplicación que se ejecuta en el cliente (ordenador), el cual para poder iniciarla y hacerla funcionar requiere estar instalada dicha aplicación en el mismo.

Una aplicación online por el contrario reside en un servidor, y su ejecución requiere disponer de un pc con conexión a internet, un navegador como Internet Explorer, Mozilla Firefox, Opera, entre otros y por supuesto que la aplicación esté funcionando en un servidor que la aloja. Las Ventajas que brindan las aplicaciones web son que proporcionan movilidad, dado que puedes ejecutarlas desde cualquier ordenador con conexión a internet. La información que manejan se accede a través de internet, motivo por el cual son especialmente interesantes para desarrollar aplicaciones multiusuario basadas en la compartición de información. El cliente o usuario que utiliza la aplicación no necesita tener un ordenador de grandes prestaciones para trabajar con ella.

1.8. Arquitectura del Framework

1.8.1. Arquitectura Java EE

La plataforma JEE utiliza un modelo de aplicación distribuida multicapa. Las partes de una aplicación JEE son presentadas en los siguientes componentes.

- Componentes de la Capa Cliente (Client-Tier) que corren en la máquina cliente.
- Componentes de la Capa Web (Web-Tier) que corren en el servidor JEE.
- Componentes de la Capa de Negocio (Business-tier) que corren en el servidor JEE.

Software de la Capa de Enterprise Information System (EIS-tier) que corren en el servidor EIS. Aunque pueda contar de 3 o 4 capas lógicas, las aplicaciones JEE en capas son denominadas aplicaciones de 3 capas, ya que hace referencia a capas físicas, pues Session Bean son distribuidas en 3 lugares físicos diferentes, máquina Cliente, máquina Servidor JEE y la máquina Servidor Base de Datos.

Las aplicaciones JEE están basadas en componentes, cada componente tiene su propia funcionalidad y permite ser ensamblado en una aplicación JEE, Estos componentes corren y son manejados por un servidor JEE. La especificación JEE define los siguientes componentes:

- Application Clients y applets son componentes que corren en el cliente.
- Java Servlet y JavaServer Pages (JSP) con componentes web que corren en el servidor.
- Enterprise Beans (EJB) son componentes de negocio que corren en el servidor.
- Existen 3 tipos de Enterprise Beans:
- Session Beans: Representa un objeto no persistente que vive mientras viva la sesión.
- Entity Beans: Representa datos persistentes en una base de datos, es decir se encarga de persistir sus datos aunque la sesión del cliente termine o el servidor se apague.
- Message-Driven Beans: Combina características de un Session Bean y un Java Message Service (JMS), permitiendo a un componente de negocio, recibir mensajes JMS asincrónicos. (ORACLE, 2012).

1.8.2. Características de Sistemas de Tipo Enterprise

Entre las características salientes de un sistema de tipo enterprise se pueden mencionar las siguientes:

- Datos masivos (gran volumen) y persistentes.
- Acceso concurrente, lo que implica gran cantidad de usuarios.
- Variedad de interfaces de usuario, lo que implica diversidad en la funcionalidad brindada.
- Integración con otros sistemas, lo que implica que comparten funcionalidad y / o datos.
- Disonancia conceptual (modelo de datos con distintas visiones), debido a que poseen un modelo de negocio subyacente que abarca distintos aspectos de un área de negocio. Por lo tanto prestan distintas funcionalidades a distintos tipos de usuarios.
- Lógica de negocio, lo que implica procesamiento de datos.

Ejemplos típicos de estos sistemas son B2C (comercio electrónico), sistemas financieros en línea, sistemas ERP (Enterprise Resource Planning). Estos sistemas por su volumen están generalmente instalados físicamente en varios nodos (servidores). Por sus características de crecimiento es importante en su diseño el concepto de escalabilidad y por la necesidad de prestar servicios en forma continua es importante el concepto de robustez. Ambos conceptos condicionan el diseño de la arquitectura de este tipo de sistemas.

1.8.3. Arquitectura de Sistemas de Tipo Enterprise

Muchas formas de plantear una solución para este tipo de sistemas, y básicamente todo sistema enterprise tiene una estructura cliente/servidor, distribuido en capas verticales. Estas capas consisten generalmente en algunas de las siguientes:

- Capa cliente de aplicación o web server.

- Capa de acceso de negocio.
- Capa de modelo de negocio.
- Capa de persistencia.
- Capa de base de datos.

1.8.4. JSF 2.0

Java Server Faces es el framework adoptado para el desarrollo de la arquitectura web. Este framework es el estándar que brinda la especificación JEE 6 para construir interfaces de usuario el cual reside en el lado del servidor. La tecnología de JSF consiste en los siguientes puntos:

- API (Application Programming Interface) para representar los componentes y la gestión de su estado, manejo de eventos, validación del lado del servidor y también del lado del cliente, comunicación entre componentes, conversión de datos, definición de la navegación entre las páginas que en la versión 2 de JSF es casi implícita, internacionalización y acceso directo a los datos de una base de datos.
- Librerías de etiquetas para agregar componentes en las páginas web y para conectar los componentes del lado del servidor.

1.8.5. Primefaces

Primefaces es un conjunto de librerías que implementan el framework JSF el cual sirven como complemento al mismo, para potenciarlo y optimizarlo de acuerdo a las necesidades de desarrollo que se quieran implementar. Primefaces reside en un servidor de aplicaciones, también provee componentes que enriquecen la interface de usuario y optimizan la utilización del framework.

1.8.6. API Java Persistence

La capa de mayor criticidad y por consiguiente en la cual más trabajo se ha desarrollado en los últimos años es la de persistencia. Debido al choque de impedancia que se produce entre los objetos del modelo de negocio y los datos persistidos en una base de datos relacional, esta capa requiere un tratamiento particular (del Rocío, 2010).

Gran parte de los productos que se han generado atacan el problema del mapeo y acceso a los datos persistentes. Algunos de los más conocidos son:

- EJB Entity beans
- JDBC
- SQLJ
- TopLink
- CocoBase
- Hibernate / nHibernate
- JPOX (JDO)
- Versant (JDO)
- OBJ
- Object Spaces

Debido a algunas limitaciones de EJB Entity Beans han surgido las otras alternativas. Básicamente los Entity Beans presentan la característica de ser usables cuando los modelos de dominio son simples, deben ser distribuidos y conectarse a otros sistemas. Su utilización es buena cuando existe un mapeo uno a uno con las tablas de la base de datos, cuando la granularidad es baja. No admiten herencia entre clases componentes; por esta razón y debido a esta limitación se desaconseja su utilización y se pueden sugerir otros productos.

Toplink es un producto muy utilizado en el mercado, ahora integrado al servidor de aplicaciones Oracle 9i AS. Permite entre otras cosas:

- Integración de EJB CMP
- Mapeo de objetos a múltiples tablas
- Uso de herencia
- Soporta bloqueo optimista de tablas
- Transacciones anidadas
- Permite adaptar el mapeo de datos entre objetos y tablas
- Permite realizar el diseño en ambas direcciones, objetos / tablas y tablas / objetos.
- Permite adaptar código SQL generado
- Permite el uso de Store Procedures
- Administra pool de objetos

Tiene una desventaja, es dependiente de APIs propietarias, Hibernate también es muy utilizado tanto en el ambiente java como .net. JDO es una especificación java que se espera se convierta en el estándar de administración de bases de datos orientadas a objetos, no exige la implementación de interfaces, presenta menor sobre-caga en la creación de objetos y su configuración se realiza en un archivo XML de forma más simple que la de los EJB CMP.

Presenta ventajas que heredó de EJB tales como el encapsulamiento de datos de la aplicación, el mapeo a tablas, trabaja en transacciones delimitadas por los Session Beans, administra pool de objetos. También posee ventajas propias como trabajar con clases java genéricas y las hace persistentes, requiere menor infraestructura.

1.8.7. Servidor de Aplicaciones

Un servidor de aplicaciones es una plataforma de software que proporciona un enfoque generalizado para la creación de una aplicación, sin tener en cuenta que las funciones de aplicación son la parte de servidor de una instancia específica de ejecución. La función del servidor se dedica a la ejecución eficaz de los procedimientos (programas, rutinas, guiones) para soportar las aplicaciones en deployment.

La mayoría de los Servidores de Aplicaciones contiene un modelo integral por capas; actuando como un conjunto de componentes accesibles para el desarrollador de software a través de un API definido por la propia plataforma.

Los componentes web se realizan generalmente en el mismo entorno ejecutándose como un servidor web, y su función principal es apoyar la construcción de páginas dinámicas ayudando a trabajar en la lógica de negocio. Implementando servicios como:

- Clustering.
- Conmutación por error (Fail-over).
- Balanceo de carga (Load-balancing).

En el caso de servidores de aplicaciones Java, el servidor se comporta como una máquina virtual ampliada para aplicaciones que se ejecutan de forma transparente para el manejo de las conexiones de la base de datos en un lado, y las conexiones con el cliente Web en el otro.

1.8.8. Glassfish

GlassFish es un proyecto de Servidor de Aplicación de código abierto iniciado por Sun Microsystems para la plataforma JavaEE y ahora patrocinada por Oracle Corporation. La versión soportada es Sun GlassFish Enterprise Server con licencia dual bajo dos licencias de software libre: el desarrollo común y la Licencia de Distribución (CDDL) y la GNU General Public License (GPL), con la excepción classpath. GlassFish es la implementación de referencia de JavaEE y como tal soporta EJB, JPA, JavaServerFaces, JMS, RMI, JavaServer Pages, servlets, entre otros.

Esto permite a los desarrolladores crear aplicaciones empresariales que son portátiles y escalables, y que se integran con las tecnologías; los componentes opcionales también se pueden instalar para servicios adicionales.

Se utiliza un derivado de Apache Tomcat como contenedor de servlets, con un componente adicional llamado Grizzly que usa JavaNew I/O (NIO) para la escalabilidad y velocidad.

1.8.9. Seguridad en aplicaciones web

Seguridad en aplicaciones Web es una rama de la seguridad de la información que tiene que ver específicamente con la seguridad de los sitios web, aplicaciones web y servicios web. A un alto nivel, la seguridad de aplicaciones Web se basa en los principios de seguridad de la aplicación, pero se aplica específicamente a los sistemas de Internet y la Web. Normalmente las aplicaciones web se desarrollan utilizando lenguajes de programación como PHP, JavaEE, Java, Python, Ruby, ASP.NET, C#, VB.NET o ASP clásico.

- **Las amenazas de seguridad**

Con la aparición de la Web 2.0 aumentó el intercambio de información a través de redes sociales conjuntamente con la creciente adopción empresarial de la Web como un medio de hacer negocios y ofrecer un servicio, los sitios web son a menudo atacados directamente. La mayoría de los ataques se producen a través de aplicaciones web cross-site scripting (XSS) y los ataques de inyección SQL, que suelen resultar de codificación erróneas y falta de sanear la entrada y salida de la aplicación web. Los errores de programación más peligrosos según el proveedor de seguridad Cenzic:

Tabla 1: Amenazas de Seguridad

Errores de Programación	Porcentaje (%)
Cross Site Scripting	37
SQL Injection	16
PHP inyección	5
Javascript inyección	5
Path Disclosure	5
Denegación de servicio	4
Memory Corruption	4
Cross Site Request Forgery	4
Divulgación de información	3
Arbitrary File	3
Local File Include	2
Remote File Include	1
Overflow	1
Otros	15

Elaborado por: Byron Andrango y Diego Jácome

1.8.10. Spring Security Framework

Spring Security fundado en 2003 y mantenido activamente por Spring Source es un framework potente y altamente personalizable en el marco de control de acceso. Spring Security bajo licencia Apache 2.0 es uno de los proyectos más maduros y ampliamente utilizados.

Hoy en día se utiliza para asegurar numerosos entornos exigentes, incluyendo organismos gubernamentales, aplicaciones militares y de los bancos centrales. Spring Security es también fácil de aprender, implementar y administrar. Permite la seguridad de aplicaciones completa en tan sólo unas pocas líneas de XML. Ofrecen una integración completa con herramientas:

- Spring Source Tool Suite
- Spring Roo framework
- The Spring Community Forum Spring Source
- Spring Spring Web Flow
- Spring Servicios Web

- Enterprise Spring Source
- Spring Source Application Management Suite
- Server Spring Source, entre otros (SpringSource, 2003).

1.8.11. Entorno de desarrollo Integrado

> Un entorno de desarrollo integrado (IDE) es una aplicación de software que proporciona servicios integrales para los programadores de computadoras para el desarrollo de software. Un IDE normalmente consiste de un editor de código fuente, constructor automático y un depurador. (Blanco, 2012).

Algunos IDE contienen compilador, intérprete, o ambos, como Microsoft Visual Studio y Eclipse, mientras que otros no lo hacen, como Sharp Develop y Lázaro. El límite entre un entorno de desarrollo integrado y otras partes del entorno de desarrollo de software más amplio no está bien definido. Un sistema de control de versión y diversas herramientas están integrados para simplificar la construcción de una interfaz gráfica de usuario.

Los IDEs modernos también tienen un navegador de clases, un inspector de objetos, y un diagrama de clases de jerarquía, para su uso con el desarrollo de software orientado a objetos.

- **Netbeans**

> NetBeans es un entorno de desarrollo integrado (IDE) para el desarrollo principalmente con Java, también con otros idiomas, en particular, PHP, C / C+ + y HTML5, tiene un marco de plataforma de aplicación para las aplicaciones Java de escritorio y otros. El IDE NetBeans está escrito en Java y puede ejecutarse en Windows, OSX, Linux, Solaris y otras plataformas que soportan una JVM compatible (Java, 2012).

La plataforma NetBeans es un marco reutilizable para simplificar el desarrollo de aplicaciones de escritorio Java Swing. El paquete NetBeans IDE para Java SE contiene herramientas para el desarrollo de plugins y aplicaciones basadas en NetBeans Platform, sin SDK adicional.
Las aplicaciones se pueden instalar módulos de forma dinámica. Cualquier aplicación puede incluir el módulo de centro de actualizaciones para que los usuarios de la aplicación descarguen actualizaciones de firma digital y las nuevas funciones directamente en la aplicación ejecutándose.

- **Jasper Report**

Es una herramienta bajo licencia libre utilizada para informes Java que puede escribir en una variedad de objetivos, tales como: pantalla, una impresora, en PDF, HTML, Microsoft Excel, RTF, ODT, valores separados por comas o archivos XML.

Se puede utilizar en aplicaciones compatibles con Java, incluyendo Java EE o aplicaciones web, para generar contenido dinámico. Algunos IDEs para Java proporcionan instrucciones para los usuarios que deseen integrar JasperReports en un proyecto.

Características

Es una biblioteca de código abierto de informes que puede ser embebida en cualquier aplicación Java. Las características incluyen:

- Scriptlets pueden acompañar a la definición del informe, que la definición de informe puede invocar en cualquier momento para realizar un procesamiento adicional. El scriptlet se construye utilizando Java, y tiene muchos ganchos que pueden ser invocados antes o después delas etapas de la generación de informes, como los informes, página, columna o grupo.
- Sub-informes para los usuarios con requisitos de gestión más sofisticados en informes, reportes diseñados para JasperReports pueden ser fácilmente importados a la JasperServer (JasperSoft, 2011).

1.9. Herramientas

- **SVN**

Es un sistema de control de versiones diseñado específicamente para reemplazar populares CVS bajo una licencia libre de Apache/BSD. SVN es conocida así por ser el Nombre de la Herramienta utilizada en la línea de Comandos. Subversion maneja un número de identificación único de versión para todo el repositorio sin un estado en común dentro de todos los archivos en un instante determinado del trabajo realizado en el repositorio. Subversion accede al repositorio configurado a través de la red, lo que le permite ser usado por personas que se encuentran en distintas computadoras.

La posibilidad de que varias personas puedan modificar y administrar el mismo conjunto de datos desde sus respectivas ubicaciones fomenta la colaboración progresando rápidamente sin tener un único conducto por el cual deban pasar todas las modificaciones. El trabajo se encuentra bajo el mando de versiones, no hay razón de temer por la calidad del proyecto se encuentre afectada si se ha hecho un cambio incorrecto en los datos se puede deshacer el último cambio realizado a una versión anterior del proyecto.

Ventajas

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.
- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente. Tiene coste de complejidad constante y no lineal como en CVS.
- Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos).

- Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.

Desventajas

- El manejo de cambio de nombres de archivos no es completo. Lo maneja como la suma de una operación de copia y una de borrado.
- No resuelve el problema de aplicar repetidamente parches entre ramas, no facilita llevar la cuenta de qué cambios se han realizado. Esto se resuelve siendo cuidadoso con los mensajes de commit.

- **Navicat**

Navicat es una herramienta de gestión de base de datos gráfica y desarrollo de software producido por PremiumSoft CyberTech Ltd. para MySQL, Oracle, SQLite, PostgreSQL y Microsoft SQL Server. Cuenta con una interfaz similar al Explorador de usuario gráfica y soporta múltiples conexiones de base de datos para bases de datos locales y remotos. Su diseño está hecho para satisfacer las necesidades de una variedad de audiencias, de los administradores de bases de datos y programadores para diversas empresas o compañías que sirven a clientes e intercambiar información con los socios.

- **Navicat Premium**

En 2009, lanzó PremiumSoft Navicat Premium, una serie de software que combina Navicat Navicat todas las versiones anteriores en una sola versión y se puede conectar a la base de datos de tipos diferentes, incluyendo MySQL, Oracle, PostgreSQL y al mismo tiempo, lo que permite a los usuarios hacer la migración de datos entre bases de datos cruzadas.

Navicat versión Premium también es compatible con varias plataformas de administración, que sirve Windows, Mac OS X y Linux. En abril de 2010, la versión 9 de Navicat Premium fue puesto en libertad, que se sumó la conectividad de base de datos SQLite para Navicat Premium, permitiendo Navicat Premium para conectarse a MySQL, Oracle, PostgreSQL y SQLite en una sola aplicación. En noviembre de 2010, el soporte para Microsoft SQL Server se añadió. En enero de 2011, SQL Azure se incluyó.

Características

- Visual Query Builder
- SSH y HTTP tunneling.
- Migración de datos y la estructura y sincronización.
- Importación y exportación y copia de seguridad de los datos.
- Generador de informes.
- Tareas de programación y herramientas asistentes (Navicat, 2012).

1.10. Construcción del Framework

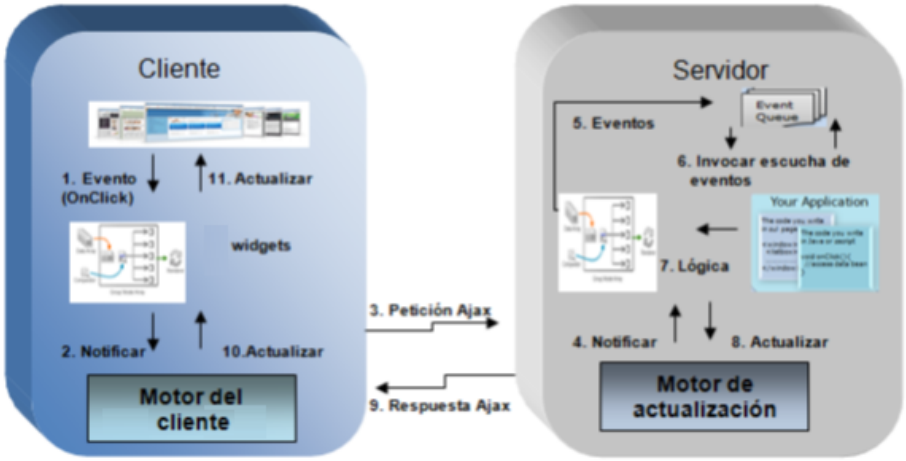
1.10.1. Fundamentos esenciales del Framework

Es un framework dirigido por eventos y basado en componentes, para desarrollar aplicaciones web sin JavaScript y basadas en Ajax, que permite al programador disminuir la codificación, el tiempo de desarrollo y lograr interfaces ricas para el usuario. Para la utilización del framework no es necesario tener ningún conocimiento de JavaScript para desarrollar aplicaciones basadas en Ajax, ya que el motor q es Primefaces genera automáticamente el código JavaScript. Para desarrollar una aplicación web se necesita saber sólo un poco sobre HTML.

1.10.2. Arquitectura centrada en el servidor

La Figura de abajo muestra cómo trabajan juntos los componentes internos y externos del framework en una petición Ajax.

Figura 1: Arquitectura centrada en el servidor



Elaborado por: Byron Andrango y Diego Jácome

Las aplicaciones se ejecutan en el servidor y pueden tener acceso a los recursos y servicios del mismo, construir la interfaz de usuario con componentes, interactuar con la actividad del usuario, y luego manipular los componentes para esta interfaz, tal como se explica en la figura anterior. La sincronización de los estados de los componentes entre el navegador y el servidor se realiza automáticamente por el motor de actualización es transparente a la aplicación. Cuando está ejecutándose del lado del servidor, accede fácilmente a cualquier tecnología Java. Las actividades del usuario, incluyendo las peticiones Ajax y la modificación de datos en el servidor, se abstraen de los objetos de evento.

Como la mayoría de los frameworks Ajax, la función del servidores pasiva, ya que sólo es responsable del suministro y aceptación de los datos después de recibirlas peticiones del cliente. La comunicación entre los componentes es bastante compleja y requiere una gran cantidad de programación JavaScript, aparte existe un gran problema de incompatibilidad entre JavaScript y navegadores. Por el contrario, en la

solución del framework, todos los componentes son creados en el servidor, lo que hace más fácil la comunicación entre ellos ya que se puede acceder a estos componentes directamente en el servidor. La forma en la que los componentes se comunican entre sí, es por eventos, lo que significa que la interacción puede ser provocada por las actividades de un usuario en el cliente o eventos enviados desde otros componentes.

1.10.3. Capa de presentación

El Framework está diseñado para ser lo más ligero posible, por este motivo el framework se focaliza más en la capa de presentación. La misma que no requiere de ninguna otra tecnología especifica back-end para funcionar correctamente. Aparte el framework deja al desarrollador la opción de elegir cualquier middleware como Java DataBase Connectivity (JDBC), Hibernate, Enterprise Java Beans (EJBs), Spring Security, entre otros. Todas estas tecnologías funcionan perfectamente con el framework. De esta manera el framework construye aplicaciones dinámicas e interactivas en el cliente con la posibilidad de ser integrada con tecnologías familiares en el servidor.

1.10.4. Librerías Relacionadas

El archivo inspectoria.war incluye todos los archivos JAR necesarios en el directorio /lib. Dentro del archivo .WAR. La siguiente lista es una introducción a cada archivo JAR y su función:

Tabla 2: Librerías Ireport

Ireport	
appframework-1.0.3.jar	MVC-estilo para escribir una sola página aplicaciones JavaScript.
commons-beanutils-1.8.0.jar	Se utilizan para la creación de clases de Java que se ajustan a los patrones de JavaBeans nomenclatura para los getters y setters de propiedad.
commons-collections-3.2.1.jar	
commons-digester-1.7.jar	
commons-logging-1.1.jar	
hsqldb.jar	El principal motor de base de datos relacional SQL escrito en Java.
jasperreports-4.1.3.jar	Librería para generación de reportes.
substance.jar	Es un look-and-feel multi-plataforma y la sensación de las aplicaciones Swing.
swing-worker-1.1.jar	Permite un uso correcto del hilo de despacho de eventos.

Elaborado por: Byron Andrango y Diego Jácome

Tabla 3: Librerías Spring

Spring	
aopalliance-1.0.jar	Solución a muchos problemas que hacen las tecnologías existentes, tales como EJB.
jstl-api-1.2.jar	Permite además desarrollar nuestras propias bibliotecas de etiquetas.
jstl-impl-1.2.jar	
myfaces-impl-2.0.0.jar	El módulo Impl implementa todos los "invisibles" de las clases que se necesitan para una aplicación JSF funcionamiento pero que no forman parte de la API pública.
spring-security-config-3.0.3.RELEASE.jar	Librerías para el módulo de seguridad con spring-security.
spring-security-core-3.0.3.RELEASE.jar	
spring-security-web-3.0.3.RELEASE.jar	

Elaborado por: Byron Andrango y Diego Jácome

Tabla 4: Librerías Generales

Librerías Generales	
barbecue-1.5-alpha3.jar	Generador de código de barras.
commons-fileupload-1.2.2.jar	Librería para subir archivos a la BD.
commons-io-2.0.1.jar	Para trabajar con los lectores, los escritores y los archivos.
itext-2.1.7.jar	Permite crear y manipular documentos PDF.
jxl.jar	Para leer, escribir y modificar hojas de cálculo de Excel.
poi-3.7.jar	Para la manipulación de documentos de Microsoft
primefaces-3.4.2.jar	Librería Jsf con Ajax.
sistema.jar	Librería del framework.

Elaborado por: Byron Andrango y Diego Jácome

1.10.5. Componentes del framework

Una vez entendida la arquitectura del framework y realizada la configuración del mismo, es necesario explicar el funcionamiento y la composición básica de sus componentes y cómo se puede escribir páginas con dichos componentes. Un componente es un objeto de la interfaz de usuario (UI), tal como un botón, una etiqueta o un árbol. El componente define la representación gráfica y comportamiento lógico de un elemento en la interfaz de usuario.

Al integrar Primefaces a nuestro framework se puede utilizar toda la gran variedad de componentes que posee el mismo.

1.10.6. Creación de componentes de forma dinámica

PrimeFaces trajo un nuevo componente Dynaform con cual se puede construir de forma muy sencilla mediante h:panelGrid o p:panelGrid si el número de filas, columnas y la posición de los elementos son conocidos. Se puede construir un formulario de forma dinámica en tiempo de ejecución si la definición del form se lo realiza desde una base de datos o un archivo XML. Dynaform hace posible la construcción de un form dinámico con etiquetas, entradas, selección y los elementos de un modelo en específico.

Para la generación de componentes dinámicos es importante entender cómo los componentes difieren entre sí, si requieren un controlador de eventos, y si tienen problemas de validación o de navegación tienen que ser considerados. La validación dinámica y la navegación también se pueden lograr, pero sin la asignación correcta y lógica para controlar el flujo del proceso, las cosas pueden salirse de control.

El principal objetivo del framework es manejar JSF dinámico, es decir crear todas las pantallas con sus componentes desde una clase, una de las ventajas de utilizar jsf dinámico es que se utiliza la propiedad binding el cual se encarga de crear el componente en el form.

<h:form binding="#{bean.formulario}"/>

- **ValueExpression**

Un valueExpresion puede obtener o establecer un valor. Expresiones que pueden tener un valor establecido en ellas se conocen como expresiones de valor-l. Los que no pueden se conocen como expresiones de valor r-. No todas las expresiones de valor r- se puede utilizar como expresiones de valor de L- por ejemplo, "${1+1}" o "${firstName} ${lastName}". El método puede ser utilizado para analizar una cadena de expresión y devolver una instancia concreta de ValueExpression que encapsula la expresión analizada.

El FunctionMapper se utiliza en tiempo de análisis y no el tiempo de evaluación, por lo que no es necesario para evaluar una expresión utilizando esta clase. Sin embargo, El Context es necesaria en el momento de la evaluación. ValueExpression tiene los siguientes métodos:

- getValue(javax.el.ELContext).
- setValue(javax.el.ELContext,java.lang.Object).
- isReadOnly(javax.el.ELContext).

- getType(javax.el.ELContext).

- **ActionListener**

Action Listener es una interfaz oyente para recibir ActionEvents de los elementos de un form. La implementación de esta interfaz debe ser thread-safe. La clase que está interesada en recibir este tipo de eventos implementa la interfaz y, a continuación registra el código fuente UIComponent de interés, llamando al addActionListener().

1.10.7. Características principales del framework

- Manejo de permisos: Maneja permisos por perfiles de usuario, lo que contempla permisos a pantallas, reportes, botones o cualquier componente de una pantalla, y también el manejo de permisos a los campos de cualquier tabla de la base de datos.
- Aplicaciones multiempresa y multi sucursal: Permite gestionar todas las sucursales de una empresa centralizando los datos principales, pero manteniendo independiente los datos de cada sucursal.
- Módulo de Seguridad: Controla el proceso de la autentificación y la autorización al sistema, para lo cual se utiliza Spring Security.
- Módulo de Auditoria: Controlará el acceso de los usuarios a las diferentes pantallas, los eventos que se realicen como insertar, eliminar, consultar, generar reportes.
- Persistencia y Concurrencia: Permitirá trabajar utilizando persistencia a la base de datos utilizando EclipseLink (JPA 2.0) y como servidor Web de aplicaciones completamente compatible con Glassfish Server, Tomcat y JBoos.
- Visualización de reportes desarrollados en iReport: Permitirá desplegar los reportes en diferentes formatos como por ejemplo PDF, XLSX, DOCX ya que cuenta con todas las clases y librerías necesarias para visualizar reportes en un ambiente Web.

- Parametrizable: Cuenta con una tabla de parámetros los cuales podrán ser configurados por el administrador del sistema.
- Optimizar la programación: Permitirá desarrollar pantallas a partir de configuraciones ya que contará con modelos de pantallas genéricas que recibirán como parámetros el nombre de la tabla, la clave primaria, la clave foránea dependiendo del tipo de pantalla, por lo que los programadores no tendrán la necesidad de programar este tipo de pantallas que no son específicas.
- Estilos y Temas por usuario: Cuenta con una variedad de estilos y temas visuales, lo que permite que el usuario seleccione el que más le agrade.

1.11. Técnicas de prueba

Una estrategia de prueba del software integra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados que dan como resultado una correcta construcción del software. Una estrategia de prueba del software proporciona un mapa a seguir para el responsable del desarrollo del software, a la organización de control de calidad y al cliente: un mapa que describe los pasos que hay que llevar a cabo como parte de la prueba, cuándo se deben planificar y realizar esos pasos, y cuánto esfuerzo, tiempo y recursos se van a requerir. Por tanto, cualquier estrategia de prueba debe incorporar la planificación de la prueba, el diseño de casos de prueba, la ejecución de las pruebas y la agrupación y evaluación de los datos resultantes.

Una estrategia de prueba del software debe ser suficientemente flexible para promover la creatividad y la adaptabilidad necesarias para adecuar la prueba a todos los grandes sistemas basados en software.

- **Organización para la prueba del software**

En cualquier proyecto de software existe un conflicto de intereses inherente que aparece cuando comienza la prueba. Se pide a la gente que ha construido el software

que lo pruebe. Esto parece totalmente inofensivo: después de todo, ¿quién puede conocer mejor un programa que los que lo han desarrollado? Desafortunadamente, los mismos programadores tienen un gran interés en demostrar que el programa está libre de errores, que funciona de acuerdo con las especificaciones del cliente y que estará listo de acuerdo con los plazos y el presupuesto. Cada uno de estos intereses se convierte en inconveniente a la hora de encontrar errores a lo largo del proceso de prueba.

Desde un punto de vista psicológico, el análisis y el diseño del software (junto con la codificación) son tareas constructivas. El ingeniero de software crea un programa de computadora, su documentación y sus estructuras de datos asociadas. Al igual que cualquier constructor, el ingeniero de software está orgulloso del edificio que acaba de construir y se enfrenta a cualquiera que intente sacarle defectos. Cuando comienza la prueba, aparece una sutil, aunque firme intención de "romper" lo que el ingeniero del software ha construido. Desde el punto de vista del constructor, la prueba se puede considerar (psicológicamente) destructiva. Por lo tanto, el constructor anda con cuidado, diseñando y ejecutando pruebas que demuestren que el programa funciona, en lugar de detectar errores. Desgraciadamente, los errores seguirán estando.

El responsable del desarrollo del software siempre es responsable de probar las unidades individuales (módulos) del programa, asegurándose de que la una lleva a cabo la función para la que fue diseñada. En muchos casos, también se encargará de la prueba de integración, el paso de prueba que lleva a la construcción (y prueba) de la estructura total del sistema. Las técnicas de prueba para software son:

- Prueba Unitaria
- Prueba de Integración
- Prueba de Regresión
- Pruebas de humo
- Pruebas del Sistema
- Pruebas de desempeño

- Pruebas de carga
- Prueba de estress
- Pruebas de volumen
- Pruebas de recuperación
- Prueba de recuperación y tolerancia a fallas
- Prueba de Múltiples sitios
- Prueba de compatibilidad y conversión
- Pruebas de integridad de datos y BD
- Pruebas de seguridad y control de acceso
- Pruebas del ciclo del negocio
- Pruebas de GUI
- Prueba de Configuración
- Pruebas de estilo
- Pruebas de aceptación
- Prueba de la instalación
- Prueba funcionales
- Prueba de campo
- Pruebas Beta
- Pruebas de documentación y procedimiento
- Prueba de usabilidad
- Pruebas alfa.

Figura 2: Estructura de Técnicas de Prueba

Elaborado por: Byron Andrango y Diego Jácome

Prueba Unitaria

- Ejecutar cada módulo
- Particionar, definir los casos de prueba.
- Comparar el resultado.

Prueba de Regresión

- Identificar errores introducidos por la combinación de programas probados unitariamente.
- Determinar cómo la base de datos de prueba será cargada
- Utilizar la técnica down-top.

Pruebas de Humo

- Detectar los errores en realeases tempranos y de manera fácil.
- Su objetivo es probar el sistema constantemente buscando que saque "humo".
- Realizar una integración de todo el sistema cada cierto periodo (se recomienda un día, máximo una semana).

Pruebas del Sistema

- Asegurar la apropiada navegación dentro del sistema, ingreso de datos, procesamiento y recuperación.
- Debe enfocarse en requisitos que puedan ser tomados directamente de casos de uso y reglas y funciones de negocios.
- Ejecutar cada caso de uso, flujo básico o función utilizando datos válidos e inválidos.

Pruebas de Stress

- Verificar que el sistema funciona apropiadamente y sin errores.
- Las pruebas de stress se proponen encontrar errores debidos a recursos bajos o completitud de recursos.
- Use los scripts utilizados en las pruebas de desempeño.

Pruebas de desempeño

- Validar el tiempo de respuesta para las transacciones.
- Medir tiempos de respuesta, índices de procesamiento de transacciones y otros requisitos sensibles al tiempo.
- Modificar archivos de datos (para incrementar el número de transacciones) o los scripts para incrementar el número de veces que ocurre cada transacción.

Pruebas de carga

- Validar el tiempo de respuesta para las transacciones.
- Medir tiempos de respuesta, índices de procesamiento de transacciones y otros requisitos sensibles al tiempo.
- Modificar archivos de datos (para incrementar el número de transacciones) o los scripts para incrementar el número de veces que ocurre cada transacción.

Pruebas de volumen

- Verificar el tamaño de la BD, el equipo si es suficiente etc.
- Las pruebas de volumen hacen referencia a grandes cantidades de datos para determinar los límites en que se causa que el Sistema falle.
- Usar múltiples clientes, ya sea corriendo las mismas pruebas o pruebas complementarias para producir el peor caso de volumen.

Pruebas de Recuperación y Tolerancia a fallas

- Verificar que los procesos de recuperación (manual o automática) restauran apropiadamente la Base de datos.
- Estas pruebas aseguran que una aplicación o sistema se recupere de una variedad de anomalías de hardware, software o red con pérdidas de datos o fallas de integridad.
- Se debe utilizar las pruebas creadas para la funcionalidad del sistema y procesos de negocios para crear una serie de transacciones.

Prueba de Múltiples Sitios

- Detectar fallas en configuraciones y comunicaciones de datos entre múltiples sitios.
- El propósito de esta prueba es evaluar el correcto funcionamiento del sistema o subsistema en múltiples instalaciones.
- Consistencia, empaquetamiento, sincronización.

Prueba de Compatibilidad y Conversión

- Buscar problemas de compatibilidad y conversión en los sistemas.
- El propósito es demostrar que los objetivos de compatibilidad no han sido logrados y que los procedimientos de conversión no funcionan.
- Compatibilidad entre programas y conversión de datos.

Pruebas de Integridad de Datos y Base de Datos

- Asegurar que los métodos de acceso y procesos funcionan adecuadamente y sin ocasionar corrupción de datos.
- La Base de datos y los procesos de Base de datos deben ser probados como sistemas separados del proyecto.
- Invocar cada método de acceso y proceso de la Base de datos, utilizando en cada uno datos válidos e inválidos. Analizar la BD.

Pruebas de Seguridad y Control de Acceso

- Nivel de seguridad de la aplicación: Verificar que un actor solo pueda acceder a las funciones y datos que su usuario tiene permitido.
- Seguridad del sistema, incluyendo acceso a datos o Funciones de negocios e incluyendo accesos remotos.
- Funciones/Seguridad de Datos: Identificar cada tipo de usuario y las funciones y datos a los que se debe autorizar.

Pruebas del Ciclo del Negocio

- Asegurar que el sistema funciona de acuerdo con el modelo de negocios emulando todos los eventos en el tiempo y en función del tiempo.

- Debería emular las actividades ejecutadas en el a través del tiempo.
- Debería identificarse un periodo, como por ejemplo un año, y las transacciones y actividades que podrían ocurrir durante un periodo.
- Ejecutar cada caso de uso, flujo básico o función utilizando datos válidos e inválidos.

Pruebas de GUI

- La navegación, los objetos de la ventana y características, tales como menús, medidas, posiciones, estados y focos.
- La prueba de interfaz de usuario verifica la interacción del usuario con el software.
- Pruebas de crear/modificar cada ventana para verificar la adecuada navegación y estado de los objetos.

Pruebas de Configuración

- Validar y verificar que el cliente del sistema funciona apropiadamente en las estaciones de trabajo recomendadas.
- Estas pruebas verifican la operación del sistema en diferentes configuraciones de hardware y software.
- Incluya la apertura o cierre de varias aplicaciones Microsoft, como Excel y Word (o algún tipo de software similar a la que se está probando).

Prueba de Estilo

- Comprobar que la aplicación sigue los estándares de estilo propios del cliente.
- Se entienden como tales el formato de las ventanas, colores corporativos, tipos de letra etc.
- Se realiza una navegación por la aplicación verificando si se cumplen con los estándares de GUI del cliente.

Prueba de Aceptación

- Determinación por parte del cliente de la aceptación o rechazo del sistema desarrollado.
- La prueba de aceptación es ejecutada antes de que la aplicación sea instalada dentro de un ambiente de producción.
- Realización de los documentos de planes de prueba de aceptación y especificación de los mismos, basados en los criterios de aceptación del cliente.

Prueba de Instalación

- Verificar y validar que el sistema se instala apropiadamente en cada cliente, bajo las siguientes condiciones: Instalaciones nuevas y actualizaciones.
- El primero es asegurar que el sistema puede ser instalado en todas las configuraciones posibles. El segundo propósito verificar que, una vez instalado, el sistema opera correctamente.
- Diseñar scripts para validar las condiciones de la máquina a instalar.

Prueba de Documentación y Procedimiento

- Evaluar la documentación del usuario.
- Evaluar la exactitud y claridad de la documentación del usuario y para determinar si el manual de procedimientos trabajará correctamente como una parte integral del sistema.
- Revisar la documentación del proyecto contra las funcionalidades del sistema y su configuración física.

Pruebas Funcionales

- Se asegura el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados.
- Las pruebas de funcionalidad deben enfocarse en los requerimientos establecidos para diseñar scripts que validen las condiciones de la máquina a instalar.
- Los resultados esperados ocurran cuando se usen datos válidos.

Prueba de Usabilidad

- Determinar la usabilidad del sistema.
- Determina cuán bien el usuario podrá usar y entender la aplicación.
- Identifica las áreas de diseño que hacen al sistema de difícil uso para el usuario.
- Verificar que la aplicación no presenta los siguientes problemas de usabilidad típicos: sistema es demasiado complejo, recuperación de errores es pobre, procedimientos no son simples ni obvios.

Prueba de Campo

- Correr el sistema en el ambiente real para encontrar errores y validar el producto contra sus especificaciones originales.
- Realizar un subconjunto válido de pruebas de sistema.
- Determinar que pruebas de sistema serán corridas para validar el sistema en producción.

Pruebas Alfa

- Prueba de aceptación para detectar errores en el sistema bajo un ambiente controlado.
- La verificación que involucra la ejecución de partes o todo del sistema en ambientes simulados, con el fin de encontrar errores.

Pruebas Beta

- Realizar la validación del sistema por parte del usuario.
- Prueba de aceptación donde la validación (o pruebas beta) involucra el uso del software en un ambiente real.
- Se selecciona un grupo de usuarios que ponen a trabajar el sistema en un ambiente real.
- Usan el sistema en sus actividades cotidianas, procesan transacciones y producen salidas normales del sistema.

CAPÍTULO 2
ANÁLISIS Y DISEÑO

2.1. Especificación de requerimientos de software

Los requisitos funcionales son tareas observables o procesos que deben llevarse a cabo por el sistema en desarrollo. Por ejemplo, un requisito funcional de un sistema contable es los inventarios, los cuales deben actualizar cantidades de stock, recordar precios de los artículos, entre otros.

Para un motor de búsqueda en la web se debe analizar con exactitud las expresiones booleanas, porque se tiene que procesar grandes volúmenes de datos para poder presentar la información solicitada.

Los requerimientos no funcionales son las cualidades o las normas que el sistema en desarrollo se debe tener o cumplir, pero que no son tareas que se pueden automatizar por el sistema. Ejemplo de requisitos no funcionales de un sistema son:

- Precio de instalación.
- Sistema Operativo en el cual debe de funcionar.
- Sistema de seguridad implementado en el sistema.

Es importante señalar que este tipo de requisitos existen siempre, independientemente del enfoque o método utilizado para gestionar el desarrollo de software. Una metodología de desarrollo de software ayuda a identificar, documentar, y fijar los requisitos.

2.2. Identificación de Actores

Tabla 5: Identificación de Actores

ACTOR	DESCRIPCIÓN
Administrador	Se encarga de la definición y administración de accesos, perfiles de usuarios al sistema de usuarios.
Empleado	Se encarga de la generación de los formularios y anexos del S. R. I

Elaborado por: Byron Andrango y Diego Jácome

2.3. Identificacion de casos de uso

Tabla 6: Identificación Casos de Uso Generales

NOMBRE	FUNCIÓN	
	Nro.	DESCRIPCIÓN
Administrador/ Empleado	CU1	Iniciar sesión
	CU2	Cerrar sesión
	CU3	Cambiar contraseña personal

Elaborado por: Byron Andrango y Diego Jácome

Tabla 7: Identificación Casos de Uso Administrador

NOMBRE	FUNCIÓN	
	Nro.	DESCRIPCIÓN
Administrador	CU1	Resetear contraseña
	CU2	Guardar registros
	CU3	Insertar detalle de registros
	CU4	Modificar registros
	CU5	Eliminar registros
	CU6	Consultar registros
	CU7	Agregar o quitar permisos
	CU8	Activar o desactivar usuarios
	CU9	Asignar sucursal(es) al usuario

Elaborado por: Byron Andrango y Diego Jácome

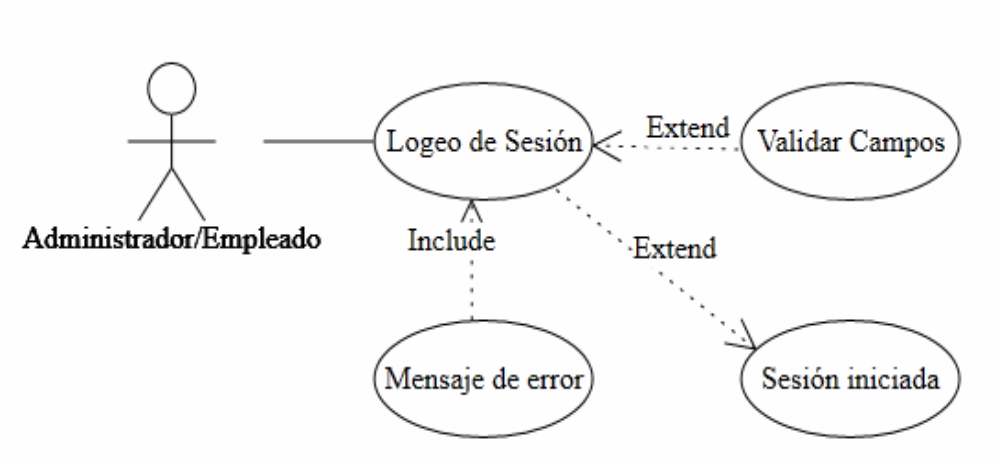
Tabla 8: Casos de Uso Empleado

NOMBRE	FUNCIÓN	
	Nro.	DESCRIPCIÓN
Empleado	CU1	Generar Formulario
	CU2	Generar Anexo

Elaborado por: Byron Andrango y Diego Jácome

2.4. Diagramas de Casos de Uso

Figura 3: Caso de Uso Iniciar Sesión



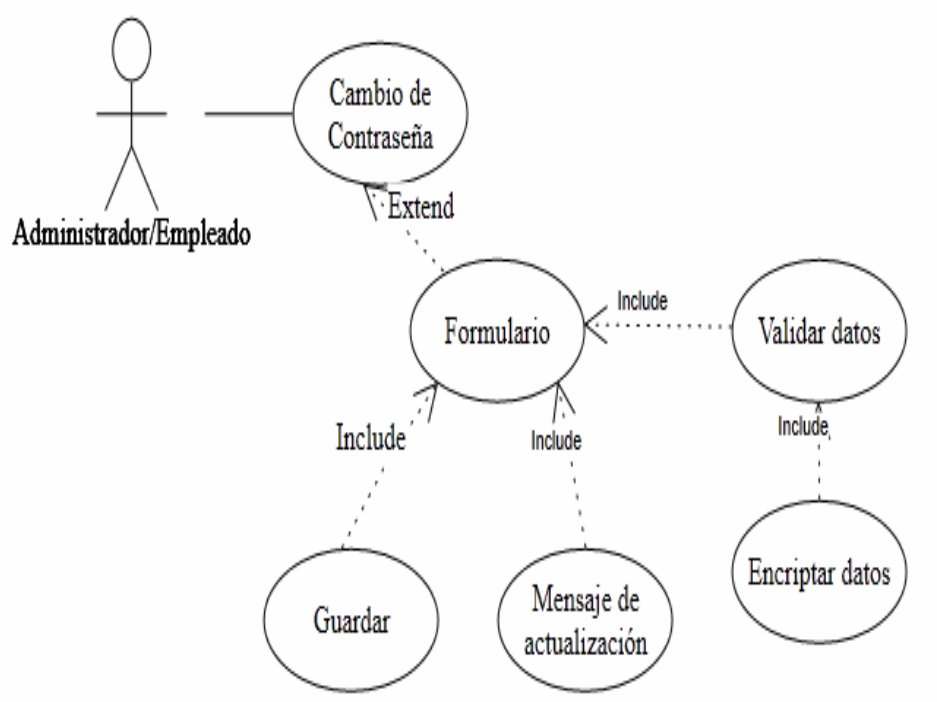
Elaborado por: Byron Andrango y Diego Jácome

Tabla 9: Caso de Uso Iniciar Sesión

Nro. Caso Uso		CU1
Nombre		Iniciar sesión
Descripción		Ingresar usuario y contraseña para iniciar sesión en el sistema.
Actores		Administrador / Empleado
Precondiciones		No haber iniciado una sesión.
		Usuario registrado en el sistema.
		Usuario habilitado.
		Ingresar nombre de usuario correcto.
		Ingresar contraseña correcta.
Postcondiciones		Administrador / Empleado logueado en el sistema
FLUJO PRINCIPAL		
#	**Administrador / Empleado**	**Sistema**
1	Inicia el sistema	
2	Ingresa nombre de usuario y contraseña	
		Valida los datos ingresados
3		Encripta contraseña
4		Recuperar datos de usuario en la base de datos
5		Autentica al usuario
6		Verifica estado de usuario
7		Registra nueva sesión
8		Direccionar a la página principal de la aplicación
FLUJO SECUNDARIO		
1. Datos Inválidos. 1.1. El sistema muestra el mensaje "El nombre del usuario o la calve son incorrectos."		

Elaborado por: Byron Andrango y Diego Jácome

Figura 4: Caso de Uso Cambio Contraseña Personal



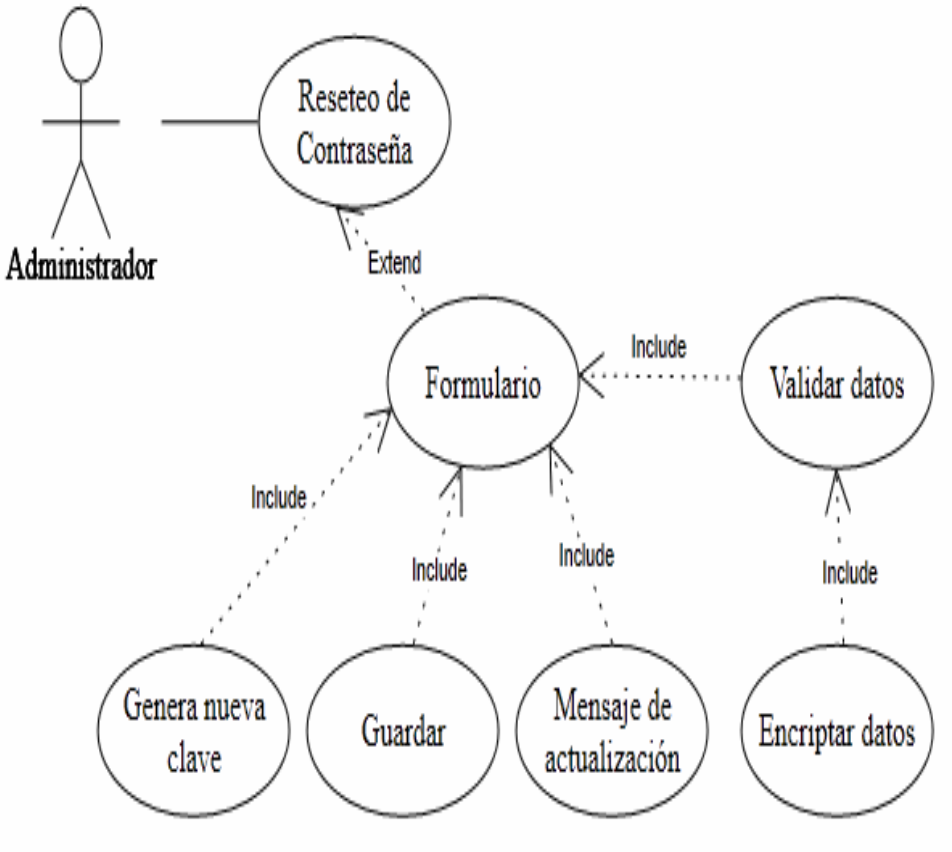
Elaborado por: Byron Andrango y Diego Jácome

Tabla 10: Caso de Uso Cambiar Contraseña Personal

Nro. Caso Uso		CU2
Nombre		Cambiar contraseña personal
Descripción		Cambio de contraseña para acceso al sistema
Actores		Administrador / Empleado
Precondiciones		Estar registrado en el sistema
		Iniciar sesión
Postcondiciones		Redirección a la página de login.
		Ingresar usuario y nueva contraseña para ingresar al sistema.
FLUJO PRINCIPAL		
#	**Administrador / Empleado**	**Sistema**
1	Elegir opción Usuario	
2		Despliega el formulario de cambio de contraseña
3	Ingresa contraseña actual	
4	Ingresa nueva contraseña	
5	Da click en guardar	
6		Valida los datos
7		Verifica contraseña
8		Encripta contraseña
9		Almacena nueva contraseña
10		Despliega mensaje de cambio satisfactorio.
FLUJO SECUNDARIO		
1. Validación de datos. 1.1. El usuario al iniciar sesión deberá cambiar la contraseña. 1.2. Si la nueva contraseña no cumple con las seguridades mínimas del sistema se despliega un mensaje de error.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 5: Caso de Uso Reseteo de contraseña



Elaborado por: Byron Andrango y Diego Jácome

Tabla 11: Caso de Uso Resetear contraseña

Nro. Caso Uso		CU1
Nombre		Resetear contraseña usuario
Descripción		Reseteo de la contraseña de acceso al sistema del usuario
Actores		Administrador
Precondiciones		Ser usuario administrador del sistema
		Iniciar sesión
Postcondiciones		Validación por parte del usuario del reseteo de la contraseña
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Selecciona opción usuario	
2	Selecciona el usuario a resetear la contraseña	
3	Da click en Generar nueva clave	
4		Genera nueva contraseña.
5		Encripta contraseña
6		Almacena nueva contraseña en la base de datos
7		Despliega mensaje de reseteo de contraseña exitoso.
FLUJO SECUNDARIO		
1. Validación de datos. 1.1. Si se produce algún error en la opción de reseteo de contraseña el sistema despliega el mensaje de error.		

Elaborado por: Byron Andrango y Diego Jácome

Casos de Uso Crear Registros

Los casos de uso para crear registros son genéricos ya que para crear un nuevo registro se sigue el mismo flujo para las siguientes opciones:

Módulo Sistema:

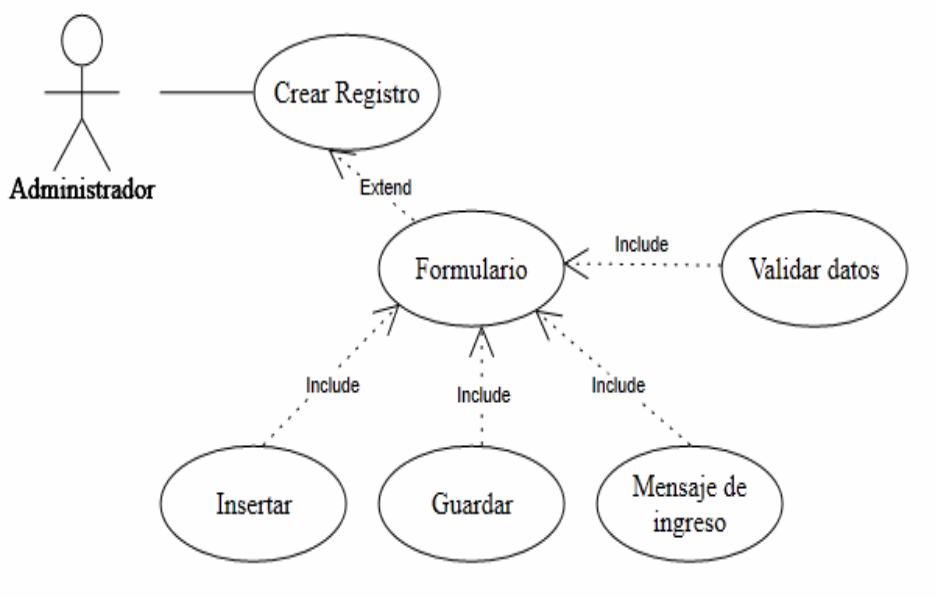
- Empresas
- Opciones
- Parámetros

- Permisos
- Sucursal
- Tablas
- Usuarios

Módulo SRI

- Fecha Pago Formulario
- Impuesto Renta
- Interés Mora
- Tipo Sustento Tributario

Figura 6: Caso de Uso Crear Registro



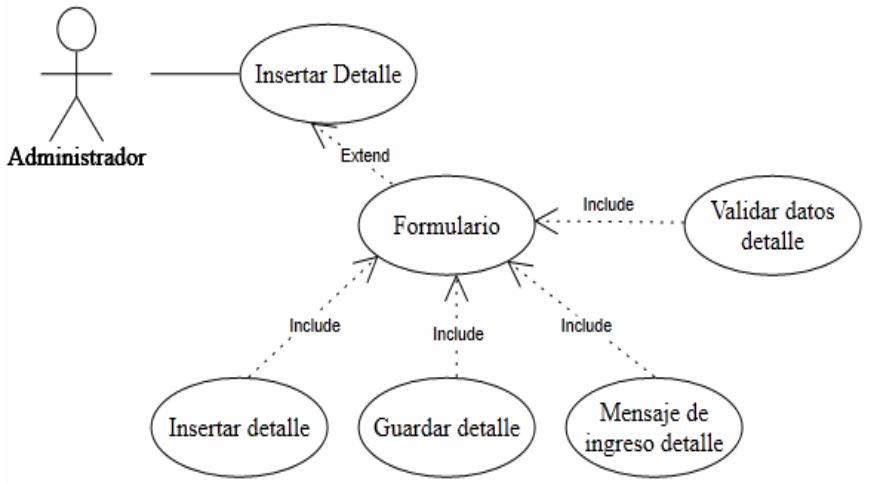
Elaborado por: Byron Andrango y Diego Jácome

Tabla 12: Caso de Uso Crear Registro

Nro. Caso Uso		CU2
Nombre		Crear Registro
Descripción		Creación de nuevos registros en el sistema
Actores		Administrador
Precondiciones		Ser usuario administrador
		Iniciar sesión
Postcondiciones		Verificar si los datos de los registros fueron ingresados correctamente.
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Da click en el Módulo Sistema o S.R.I	
2	Selecciona una opción del Módulo	
3	Da click en insertar	
4	Ingresa los datos en el formulario	
5	Da click en guardar	
6		Valida datos ingresados en el formulario
7		Almacena registro en base de datos
8		Retorna mensaje de ingreso exitoso del registro
FLUJO SECUNDARIO		
1. Validación de datos. 1.1. Si el usuario no cumple con las seguridades requeridas, despliega mensaje de error.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 7: Caso de Uso Insertar Detalle



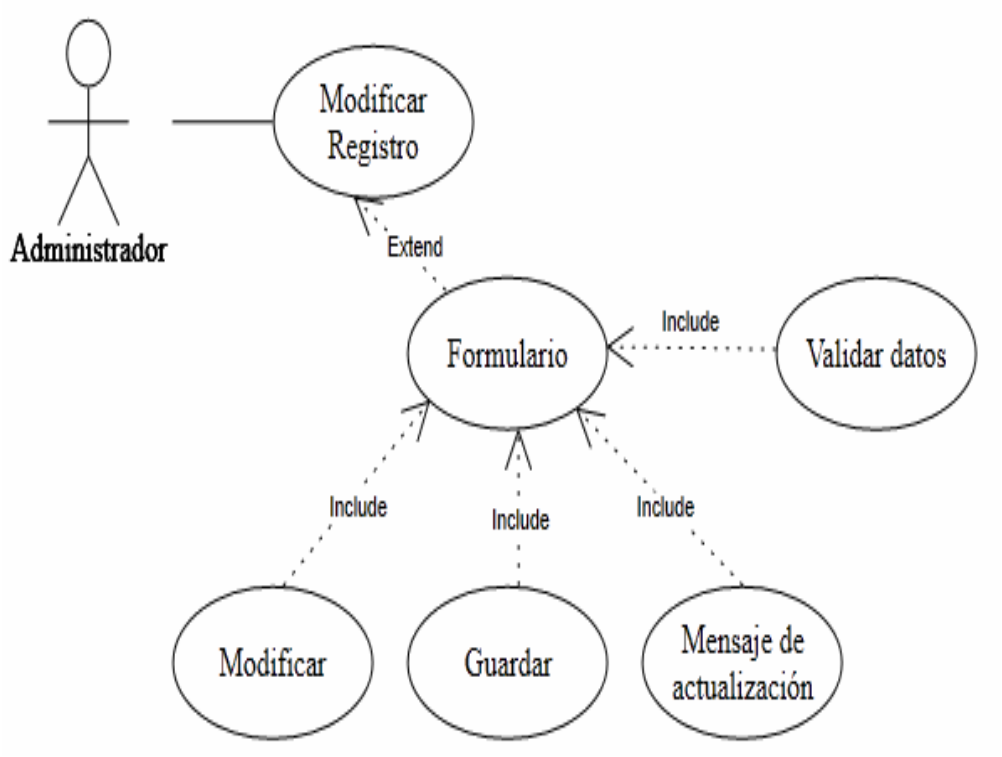
Elaborado por: Byron Andrango y Diego Jácome

Tabla 13: Caso de Uso Insertar Detalle

Nro. Caso Uso		CU3
Nombre		Insertar detalle
Descripción		Inserta el detalle de la cabecera seleccionada
Actores		Administrador
Precondiciones		Ser usuario administrador
		Iniciar sesión
Postcondiciones		Verificar que el detalle de la cabecera se ingresó correctamente.
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Da click en el Módulo Sistema o S.R.I	
2	Selecciona una Opción del Módulo	
3	Da click derecho en el formulario	
4	Selecciona insertar	
5	Ingresa datos en el detalle	
6		Valida datos ingresados en el detalle
7		Modifica la cabecera y guarda el detalle en la base de datos
8		Retorna mensaje de ingreso exitoso del resgistro
FLUJO SECUNDARIO		
1. Validación de datos. 1.1. Si no cumple con las especificaciones requeridas, despliega mensaje de error.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 8: Caso de Uso Modificar Registro



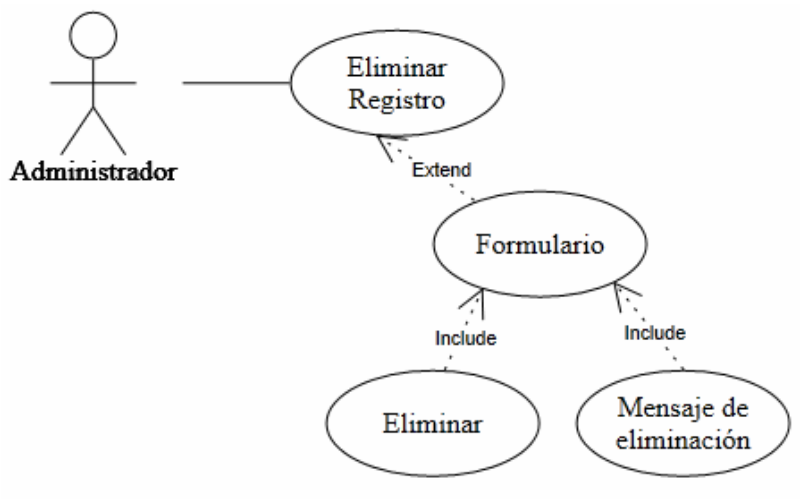
Elaborado por: Byron Andrango y Diego Jácome

Tabla 14: Caso de Uso Modificar Registro

Nro. Caso Uso		CU4
Nombre		Modificar registro
Descripción		Modificación de registros en el sistema.
Actores		Administrador
Precondiciones		Ser usuario administrador
		Iniciar sesión
		El registro debe de existir en base de datos.
Postcondiciones		Verificar la modificación del registro en el sistema.
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Da click en el Módulo Sistema o S.R.I	
2	Selecciona una Opción del Módulo	
3	Selecciona el registro a modificar	
4	Modifica los datos del registro en el formulario	
5	Da click en guardar	
6		Valida datos ingresados en el formulario
7		Modifica el registro en la base de datos
8		Retorna mensaje de actualización exitosa del registro.
FLUJO SECUNDARIO		
1. Validación de datos. 1.1. Si no cumple con las especificaciones requeridas, despliega mensaje de error.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 9: Caso de Uso Eliminar Registro



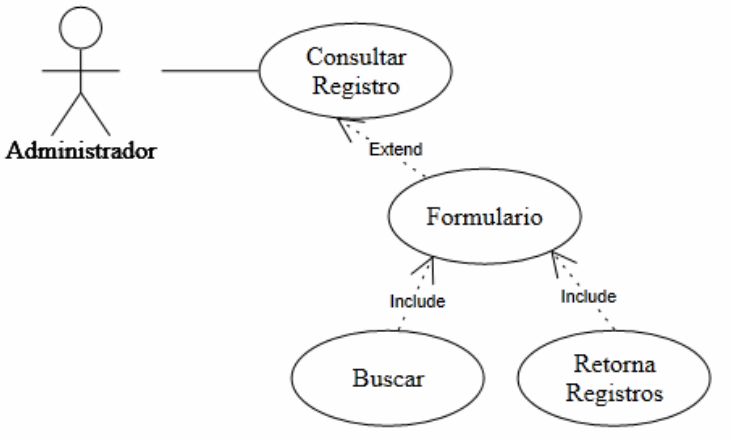
Elaborado por: Byron Andrango y Diego Jácome

Tabla 15: Caso de Uso Eliminar Registro

Nro. Caso Uso		CU5
Nombre		Eliminar registro
Descripción		Eliminación de registros en el sistema.
Actores		Administrador
Precondiciones		Ser usuario administrador
		Iniciar sesión
		El registro debe de existir en base de datos.
Postcondiciones		Verificar la eliminación del registro en el sistema.
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Da click en el Módulo Sistema o S.R.I	
2	Selecciona una Opción del Módulo	
3	Selecciona el registro a eliminar	
4	Da click en eliminar	
6		Elimina el registro de la base de datos
7		Retorna mensaje de eliminación exitosa del registro
FLUJO SECUNDARIO		
1. Validación de datos. 1.1. Si no cumple con las especificaciones requeridas, despliega mensaje de error.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 10: Caso de Uso Consultar Registro



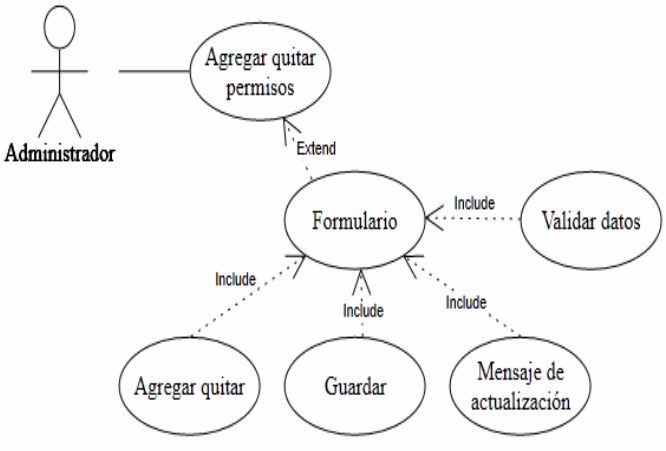
Elaborado por: Byron Andrango y Diego Jácome

Tabla 16: Caso de Uso Consultar Registro

Nro. Caso Uso		CU6
Nombre		Consultar registro
Descripción		Consultar registros en el sistema.
Actores		Administrador
Precondiciones		Ser usuario administrador
		Iniciar sesión
Postcondiciones		Verificar los datos retornados de la base de datos.
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Da click en el Módulo Sistema o S.R.I	
2	Selecciona una Opción del Módulo	
3	Da click derecho en el formulario	
4	Selecciona Buscar	
5	Selecciona el campo e ingresa el valor a buscar	
6	Da click en Aceptar	
7		Consulta el registro en la base de datos con los parámetros seleccionados
8		Retorna la información del registro
FLUJO SECUNDARIO		
1. Listado de datos. 1.1. Si no existen registros se muestra mensaje informativo.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 11: Caso de Uso Agregar o quitar Permiso



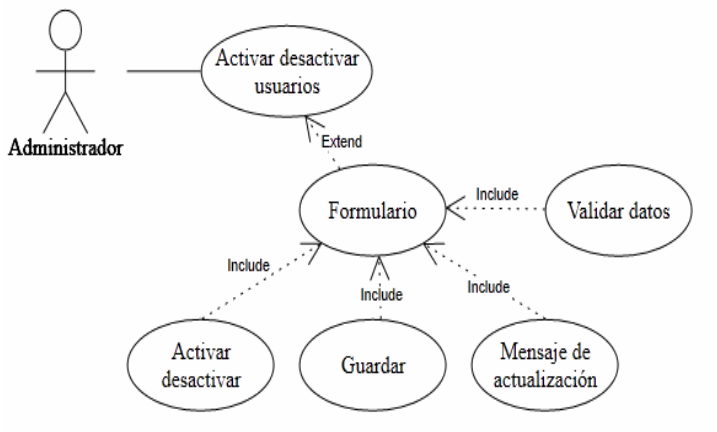
Elaborado por: Byron Andrango y Diego Jácome

Tabla 17: Caso de Uso Agregar o Quitar Permisos

Nro. Caso Uso		CU7
Nombre		Agregar o quitar permisos
Descripción		Se agregan o quitan permisos de acceso al sistema.
Actores		Administrador
Precondiciones		Ser usuario administrador
		Iniciar sesión
		El permiso debe existir en base de datos.
Postcondiciones		Verificar los permisos agregados o quitados a cada usuario en el sistema.
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Da click en el Módulo Sistema	
2	Selecciona la Opción Permisos	
3	Agrega o quita permisos de acceso al sistema	
4	Da click en guardar	
5		Actualiza los permisos de acceso al sistema en la base de datos
6		Retorna mensaje de actualización exitosa del permiso
FLUJO SECUNDARIO		
1. Listado de datos. 1.1. Si no existen registros se muestra mensaje informativo.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 12: Caso de Uso Activar o desactivar Usuario



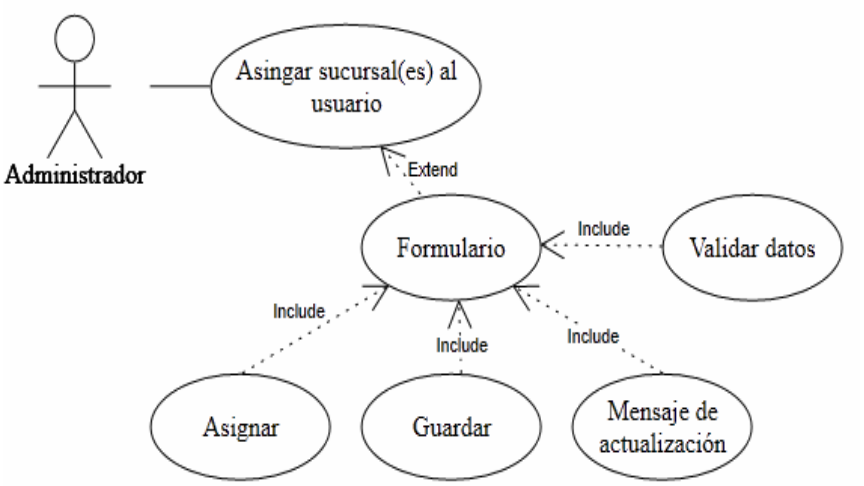
Elaborado por: Byron Andrango y Diego Jácome

Tabla 18: Caso de Uso Activar o desactivar Usuario

Nro. Caso Uso		CU8
Nombre		Activar o desactivar usuarios
Descripción		Modificar los datos de un usuario del sistema.
Actores		Administrador
Precondiciones		Ser usuario administrador
		Iniciar sesión
		El usuario debe de estar registrado en el sistema.
Postcondiciones		Verificar si el usuario se encuentra activo o desactivo en el sistema.
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Da Click en el Módulo Sistema	
2	Selecciona la Opción Usuarios	
3	Selecciona el usuario	
4	Activa o desactiva el usuario	
5	Da click en guardar	
6		Valida los datos ingresados del usuario
7		Retorna mensaje de actualización exitosa del usuario
FLUJO SECUNDARIO		
1. Validación de datos ingresados. 1.1. Si no cumple con las especificaciones requeridas, despliega mensaje de error.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 13: Caso de Uso Asignar Sucursal (es) al usuario



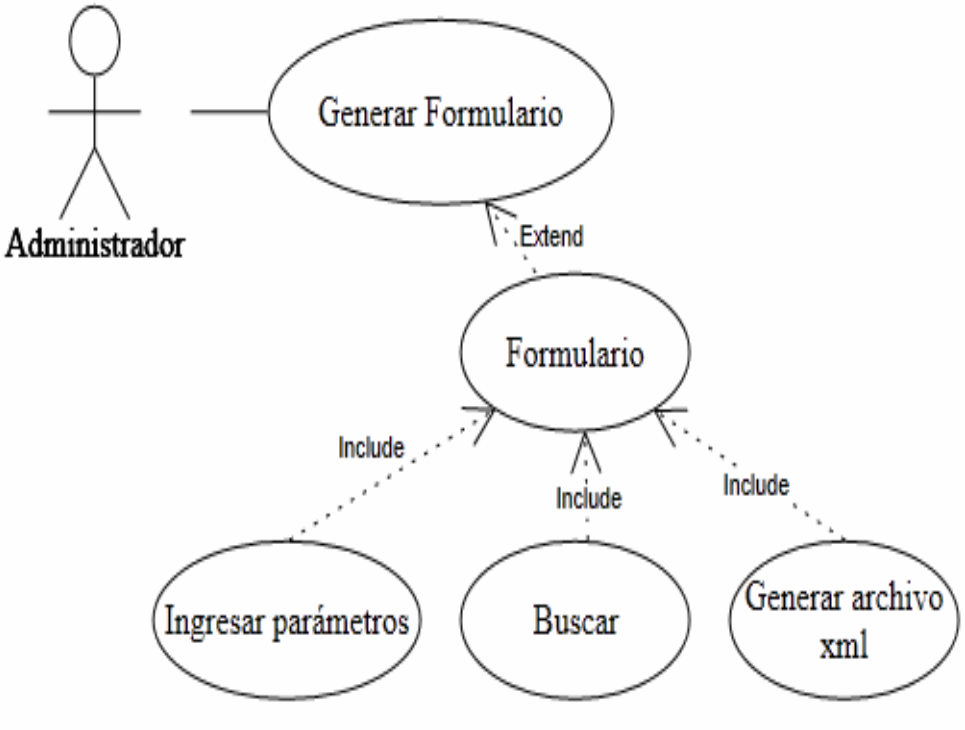
Elaborado por: Byron Andrango y Diego Jácome

Tabla 19: Caso de Uso Asignar Sucursal(es) al Usuario

Nro. Caso Uso		CU9
Nombre		Asignar sucursal(es) al usuario
Descripción		Asignar sucursales a los usuarios.
Actores		Administrador
Precondiciones		Ser usuario administrador
		Iniciar sesión
		La sucursal debe de estar registrada en el sistema.
Postcondiciones		Verificar si los usuarios tienen asignados sucursales.
FLUJO PRINCIPAL		
#	**Administrador**	**Sistema**
1	Da Click en el Módulo Sistema	
2	Selecciona la Opción Usuarios	
3	Selecciona el usuario	
4	Inserta Sucursal al usuario	
5	Da click en guardar	
6		Valida los datos ingresados del usuario
7		Retorna mensaje de actualización exitosa del usuario
FLUJO SECUNDARIO		
1. Validación de datos ingresados. 1.1. Si no cumple con las especificaciones requeridas, despliega mensaje de error.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 14: Caso de Uso Generar Formulario



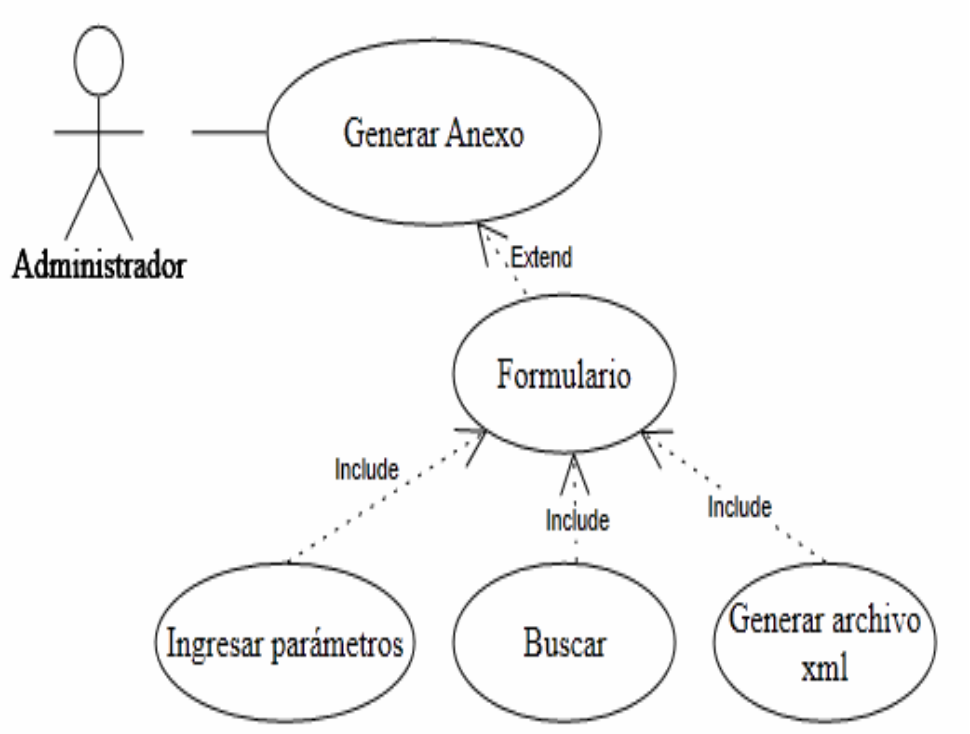
Elaborado por: Byron Andrango y Diego Jácome

Tabla 20: Caso de Uso Generar Formulario

Nro. Caso Uso		CU1
Nombre		Generar formulario S. R. I
Descripción		Generación del formulario S. R. I en el sistema.
Actores		Usuario
Precondiciones		Estar registrado en el sistema
		Iniciar sesión
Postcondiciones		Validar xml generado por el sistema en el DIMM.
FLUJO PRINCIPAL		
#	**Usuario**	**Sistema**
1	Da Click en el Módulo S.R.I	
2	Selecciona Formulario 103 o 104	
3	Selecciona el mes, año, tipo de declaración y genera la información.	
4		Consulta en la base de datos los registros con los parámetros enviados por el usuario.
5		Genera archivo xml con datos consultados de la base de datos.
6	Exportar el archivo xml generado por el sistema.	
7	Validar el archivo xml generado en el programa DIMM	
FLUJO SECUNDARIO		
1. Validación de datos. 1.1. Si no existen registros con el año, mes, tipo de declaración, se muestra mensaje de información sin registros. 1.2. Puede presentar problemas al generar o exportar el archivo xml mantener esta consideración para el resto de escenarios.		

Elaborado por: Byron Andrango y Diego Jácome

Figura 15: Caso de Uso Generar Anexo



Elaborado por: Byron Andrango y Diego Jácome

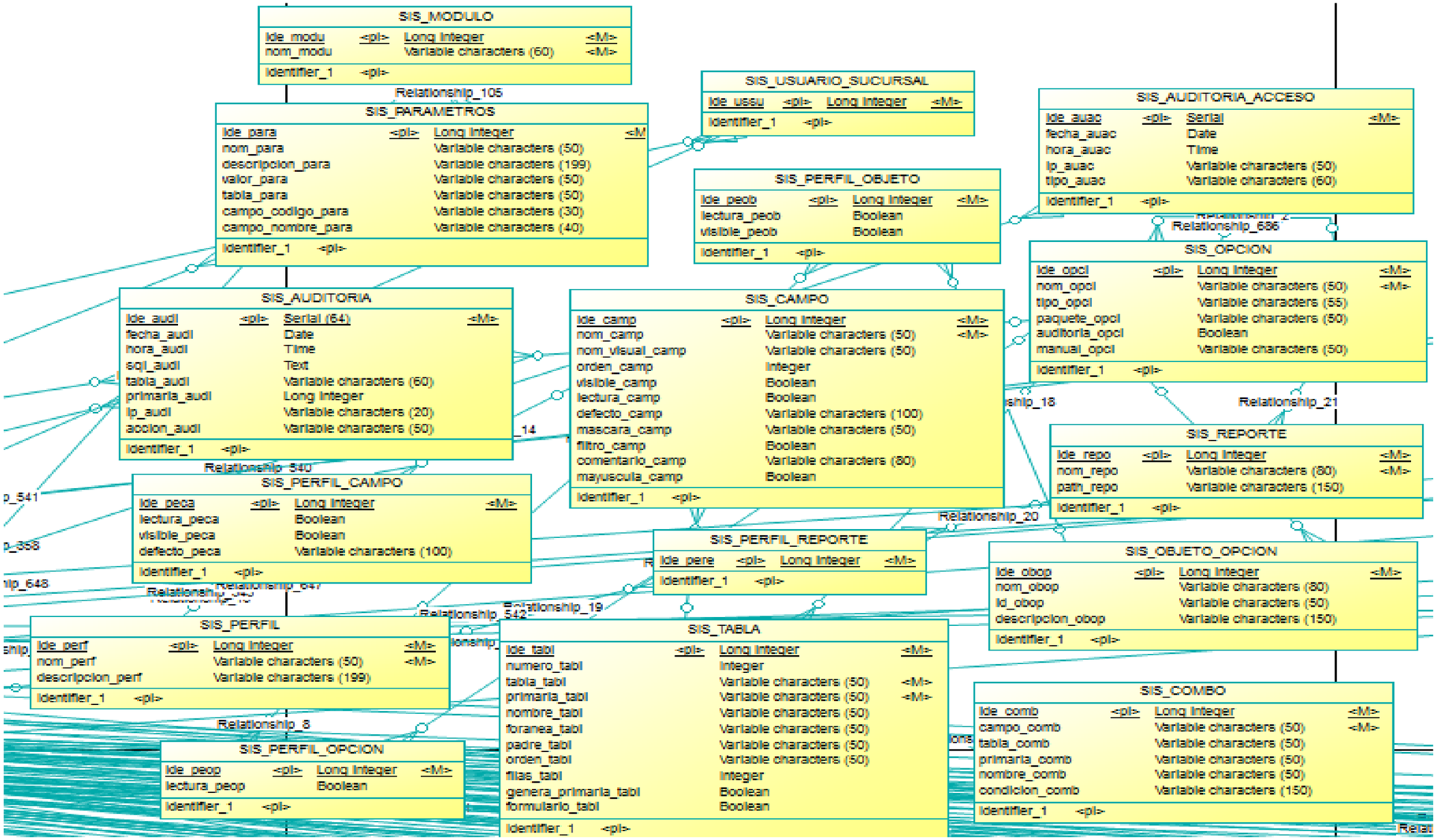
Tabla 21: Caso de Uso Generar Anexo

Nro. Caso Uso		CU2
Nombre		Generar Anexo S. R. I
Descripción		Generación del Anexo S. R. I en el sistema.
Actores		Usuario
Precondiciones		Estar registrado en el sistema
		Iniciar sesión
Postcondiciones		Validar xml generado por el sistema en el DIMM.
FLUJO PRINCIPAL		
#	**Usuario**	**Sistema**
1	Da Click en el Módulo S.R.I	
2	Selecciona Anexo RDEP, REOC, ATS	
3	Selecciona el mes, año, tipo de declaración y genera la información.	
4		Consulta en la base de datos los registros con los parámetros enviados por el usuario.
5		Genera archivo xml con datos consultados de la base de datos.
6	Exportar el archivo xml generado por el sistema.	
7	Validar el archivo xml generado en el programa DIMM	
FLUJO SECUNDARIO		
1. Validación de datos. 1.1. Si no existen registros con el año, mes, tipo de declaración, se muestra mensaje de información sin registros. 1.2. Puede presentar problemas al generar o exportar el archivo xml mantener esta consideración para el resto de escenarios.		

Elaborado por: Byron Andrango y Diego Jácome

2.5. Diseño de la Base de Datos

Figura 16: Diseño base de datos framework



Elaborado por: Byron Andrango y Diego Jácome

2.5.1. Diseño de la Base de Datos del módulo SRI

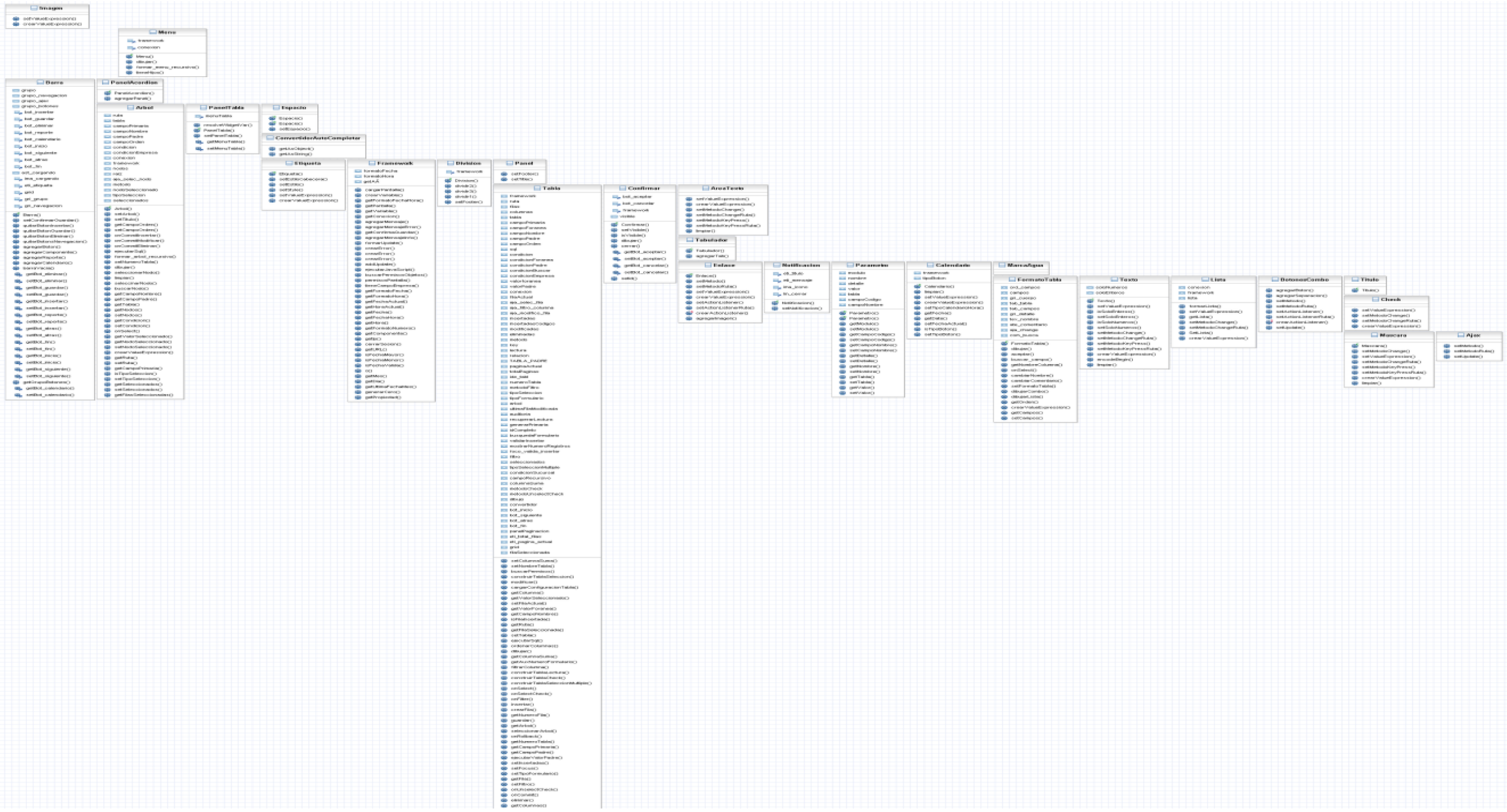
Figura 17: Diseño base de datos módulo S. R. I

Elaborado por: Byron Andrango y Diego Jácome

2.6. Diagrama de Clases

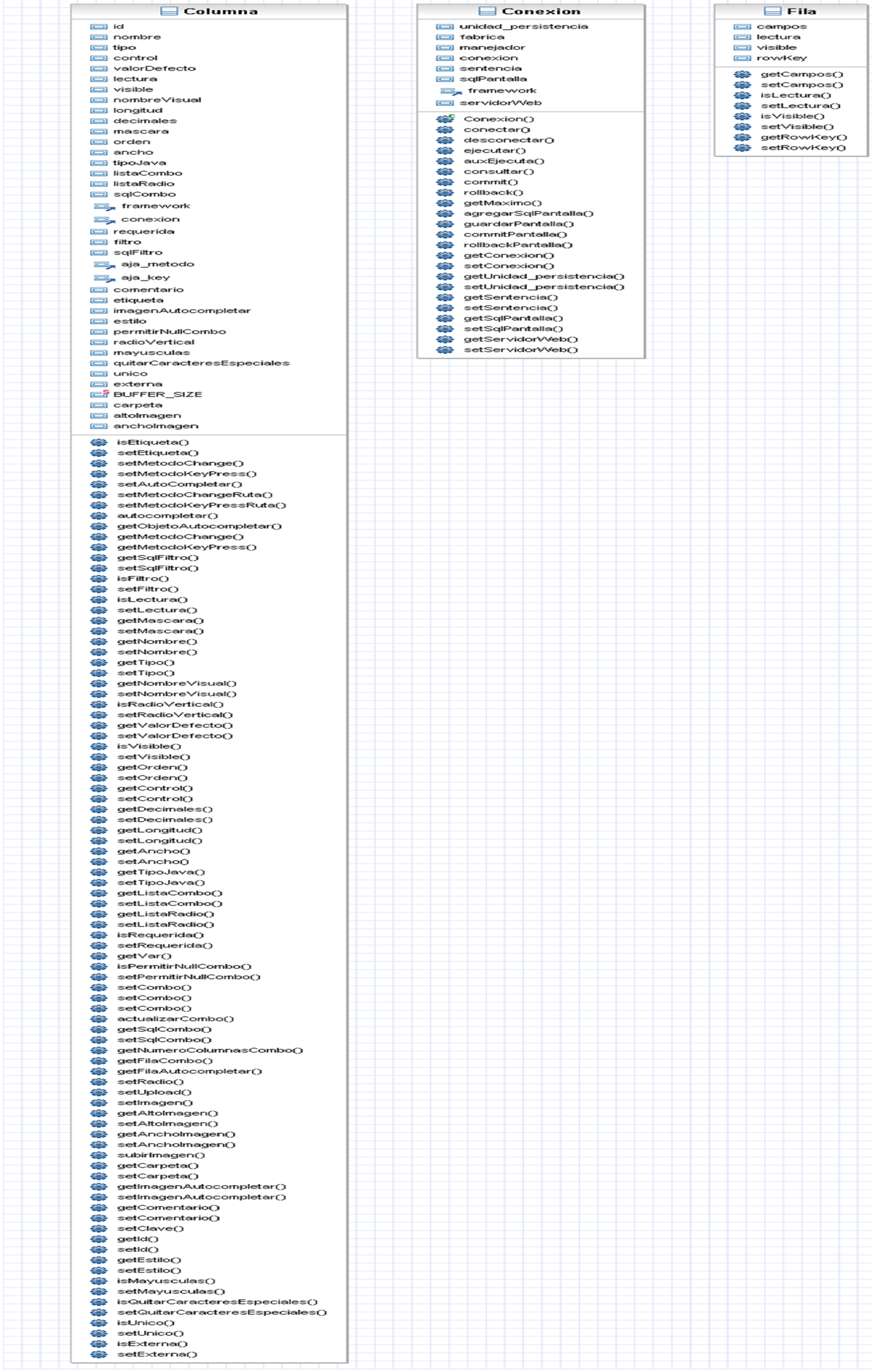
2.6.1. Diagrama de Clases del framework

Figura 18: Diagrama de clases paquete framewor



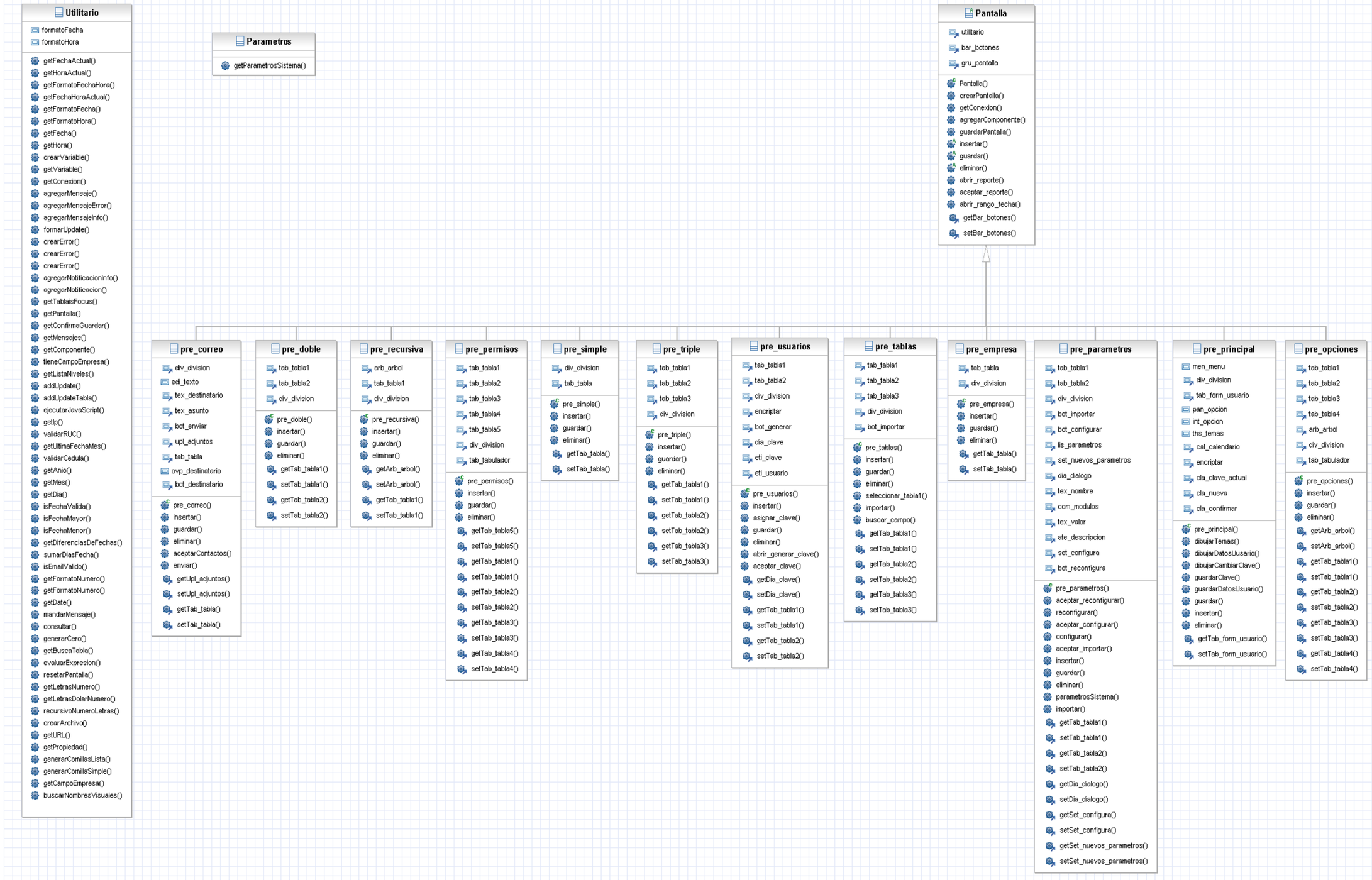
Elaborado por: Byron Andrango y Diego Jácome

Figura 19: Diagrama de clases paquete persistencia



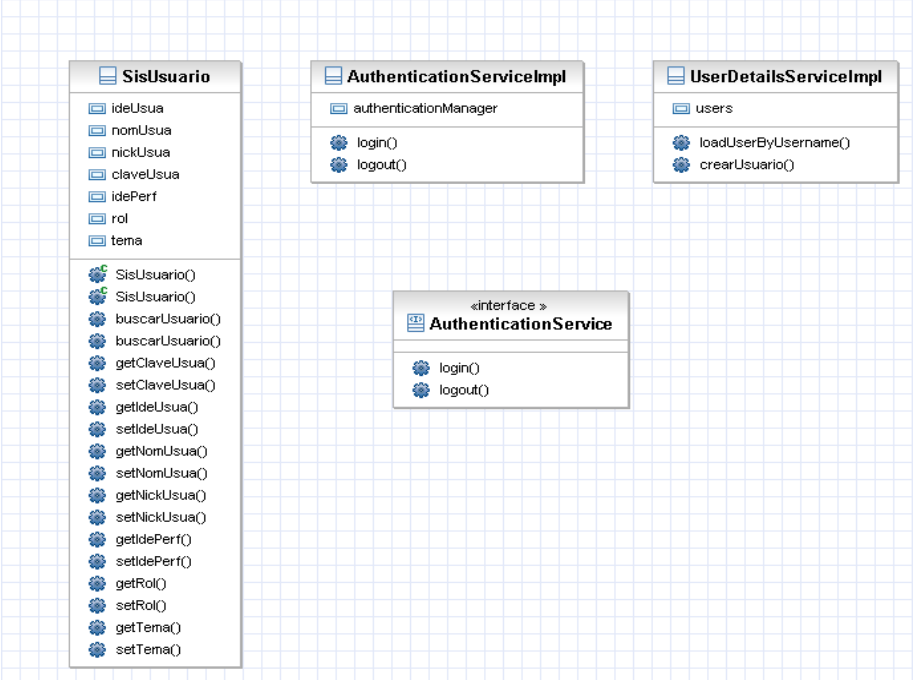
Elaborado por: Byron Andrango y Diego Jácome

Figura 20: Diagrama de clases paquete sistema



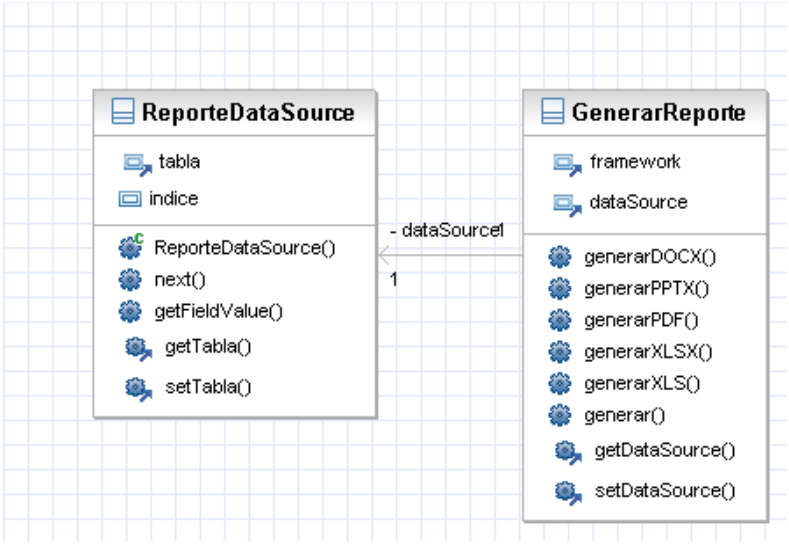
Elaborado por: Byron Andrango y Diego Jácome

Figura 21: Diagrama de clases paquete seguridad



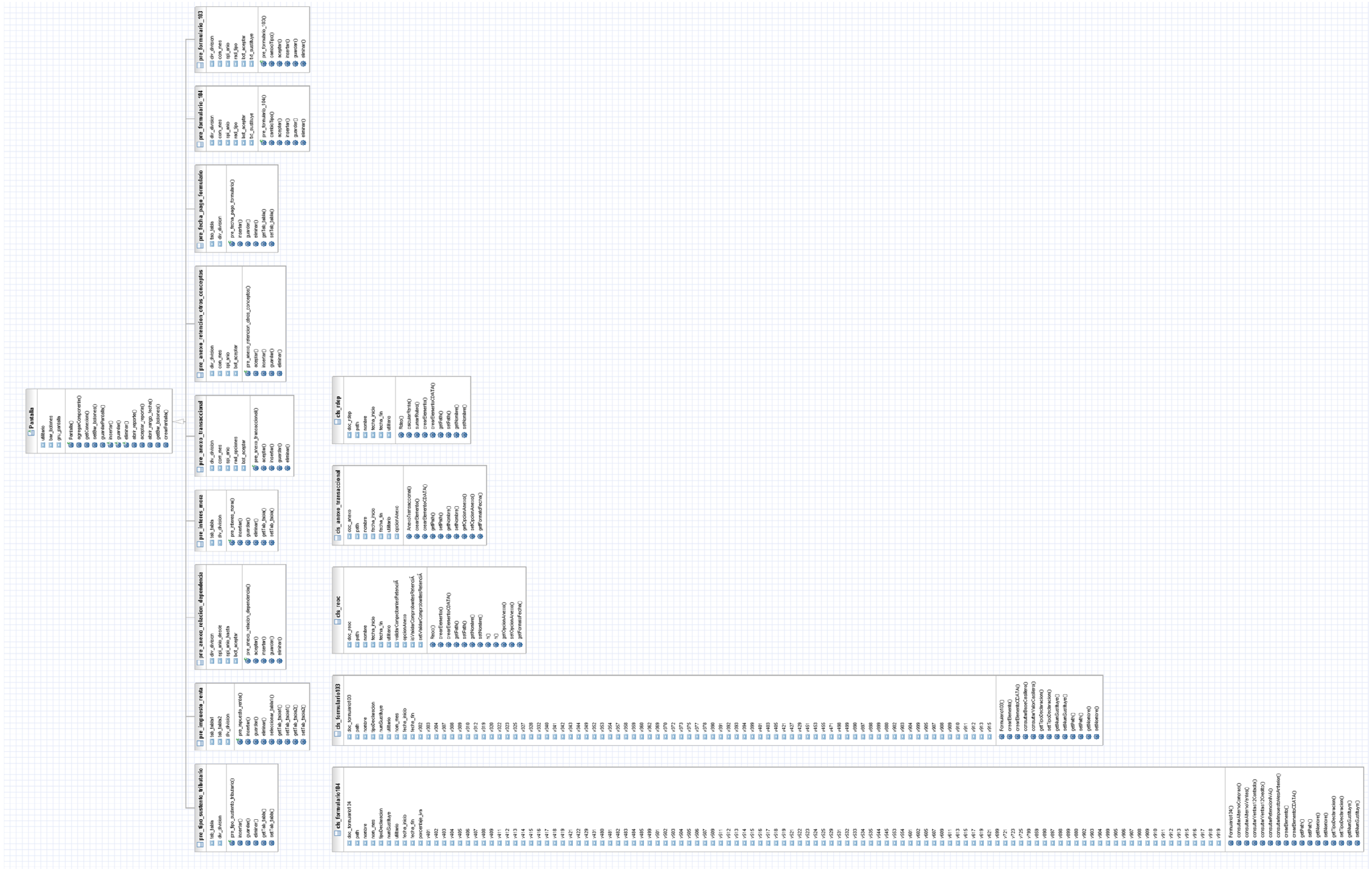
Elaborado por: Byron Andrango y Diego Jácome

Figura 22: Diagrama de clases paquete reportes



Elaborado por: Byron Andrango y Diego Jácome

Figura 23: Diagrama de clases módulo S. R. I

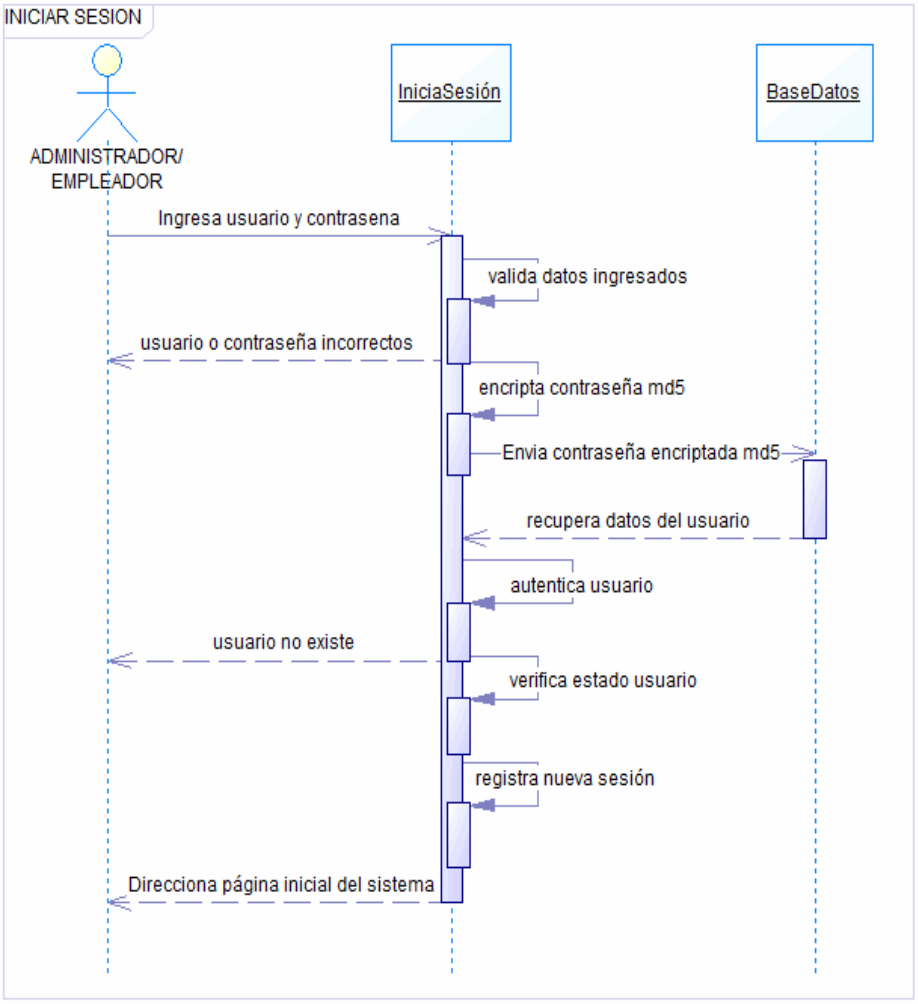


Elaborado por: Byron Andrango y Diego Jácome

2.7. Diagramas de Secuencia del Sistema

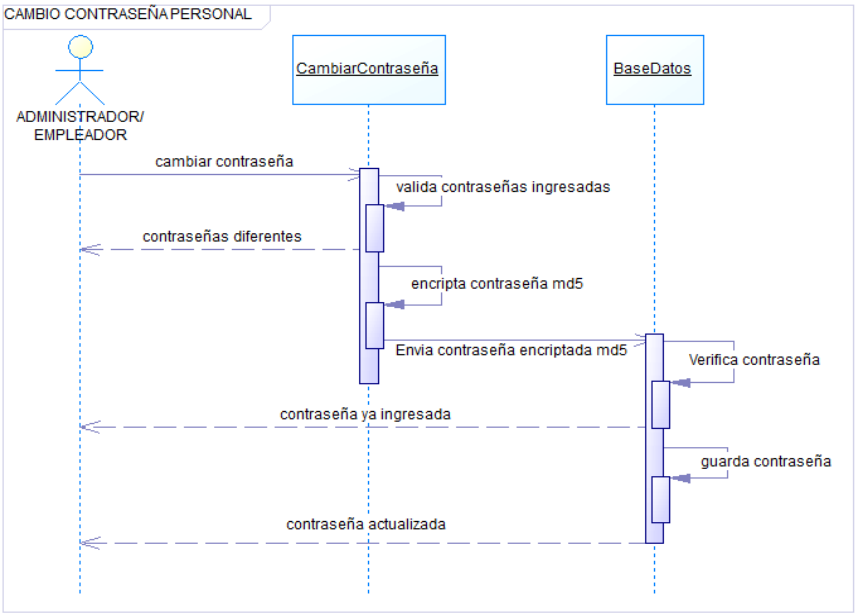
- **Diagramas de Secuencia Generales**

Figura 24: Diagrama Secuencia Iniciar Sesión



Elaborado por: Byron Andrango y Diego Jácome

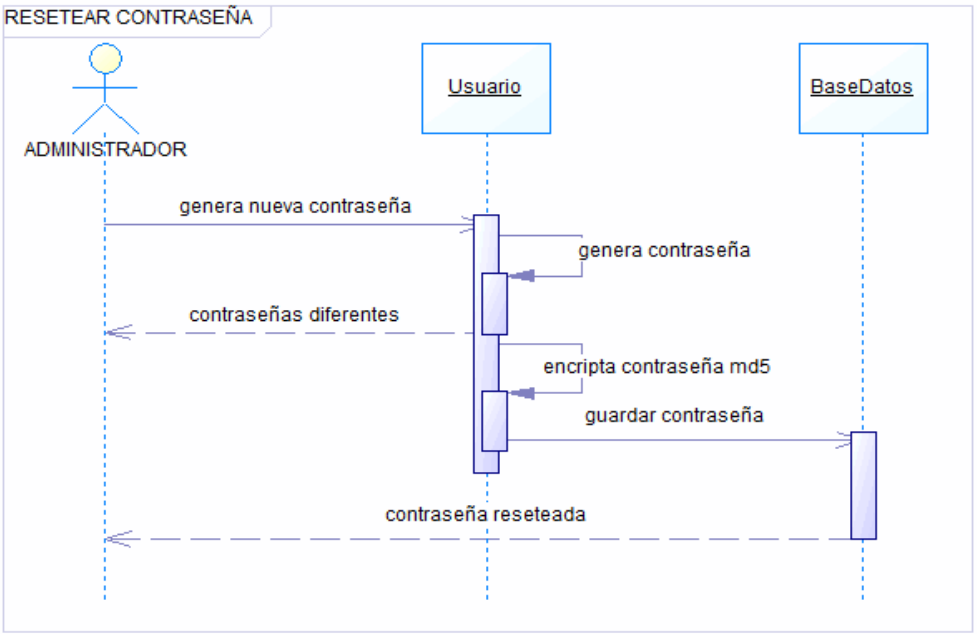
Figura 25: Diagrama Secuencia Cambio Contraseña Personal



Elaborado por: Byron Andrango y Diego Jácome

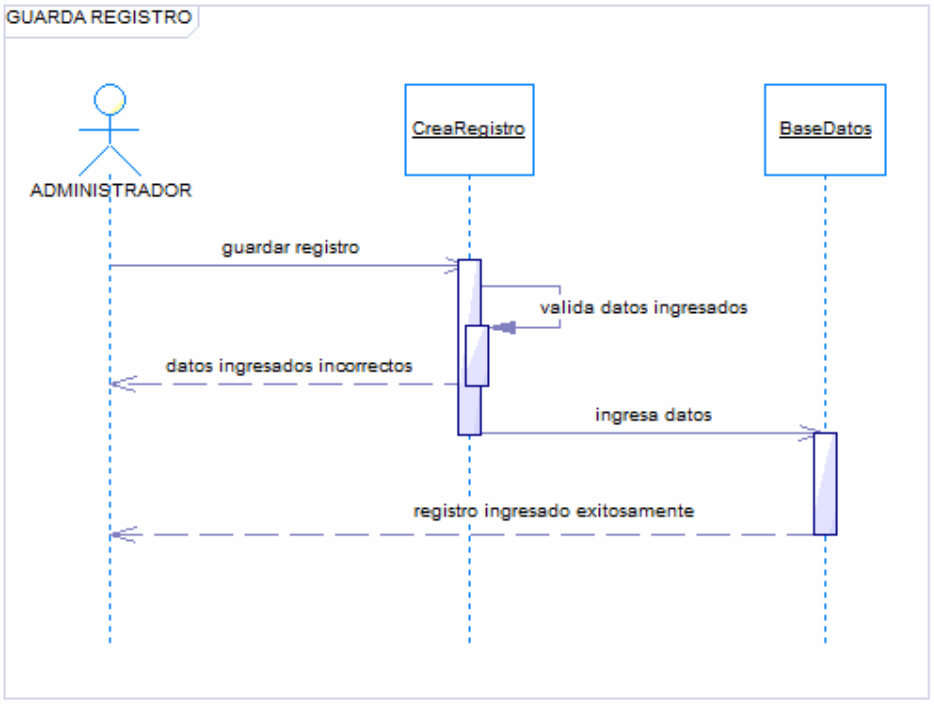
- **Diagramas de Secuencia Usuario Administrador**

Figura 26: Diagrama Secuencia Resetear Contraseña



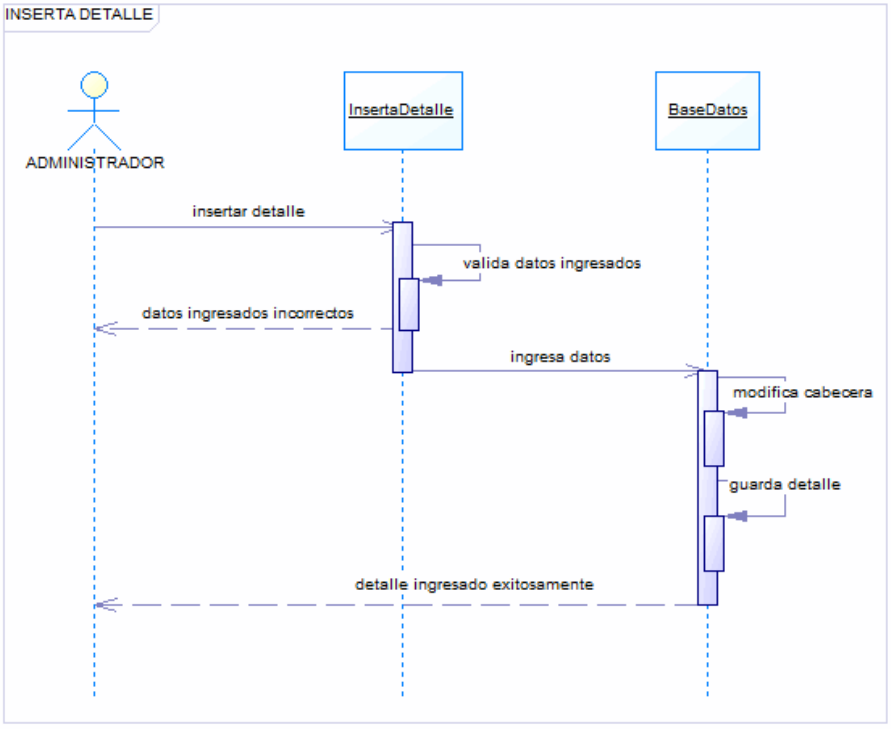
Elaborado por: Byron Andrango y Diego Jácome

Figura 27: Diagrama Secuencia Guardar Registro



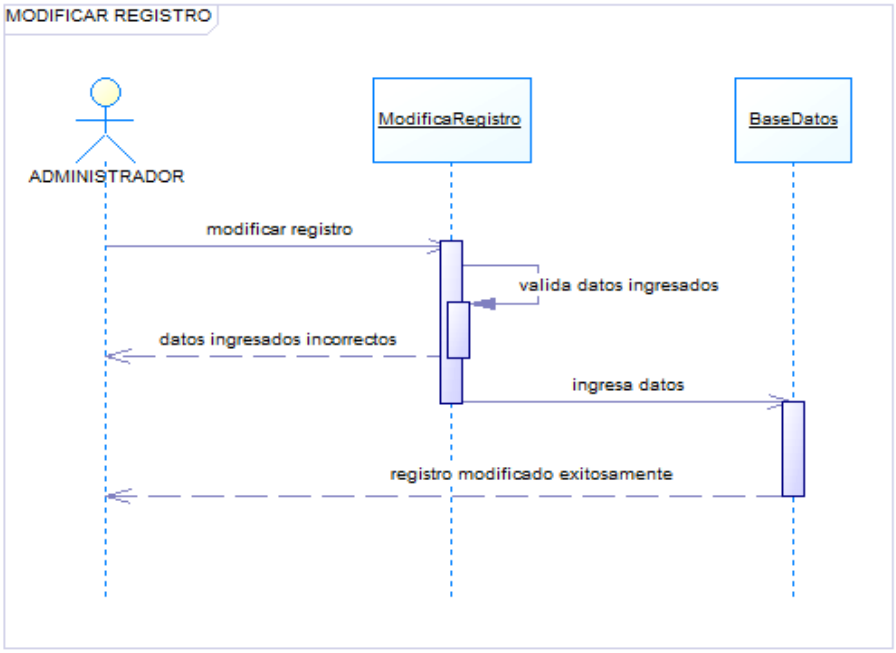
Elaborado por: Byron Andrango y Diego Jácome

Figura 28: Diagrama Secuencia Inserta Detalle



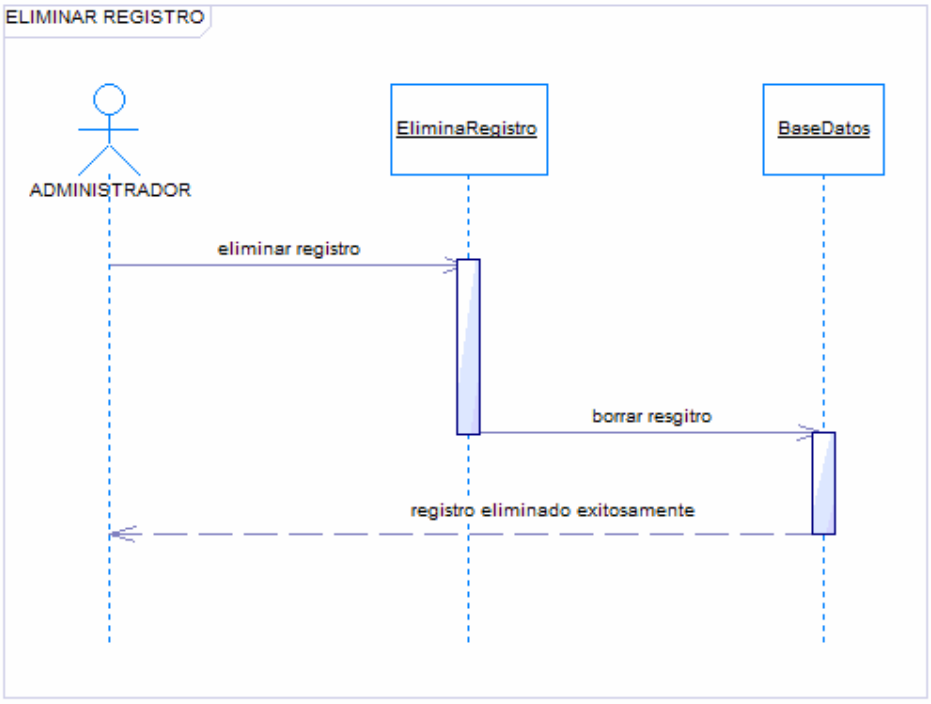
Elaborado por: Byron Andrango y Diego Jácome

Figura 29: Diagrama Secuencia Modificar Registro



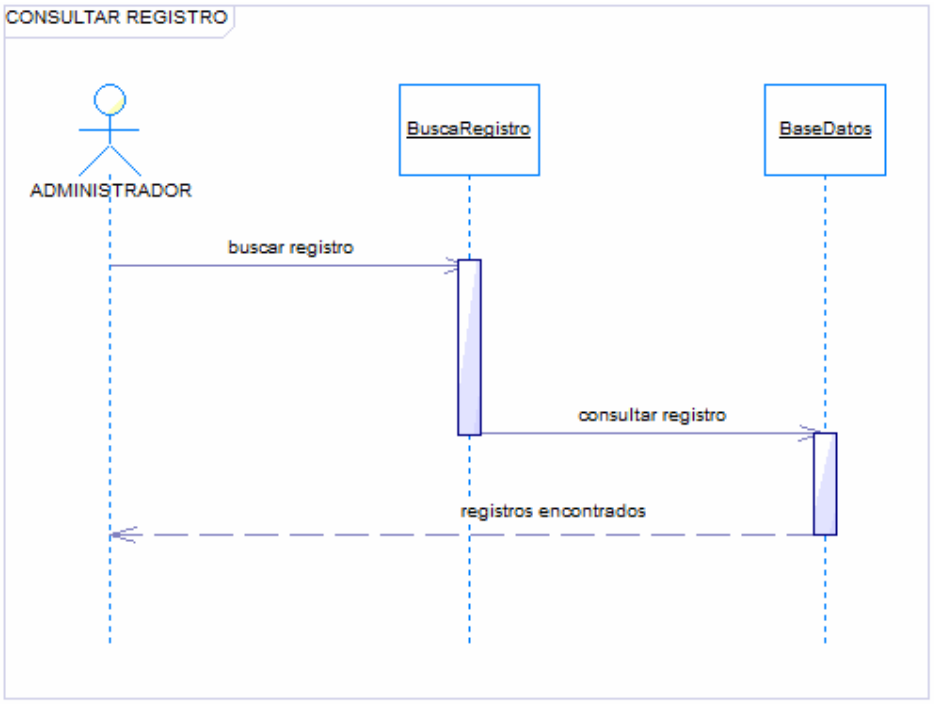
Elaborado por: Byron Andrango y Diego Jácome

Figura 30: Diagrama Secuencia Eliminar Registro



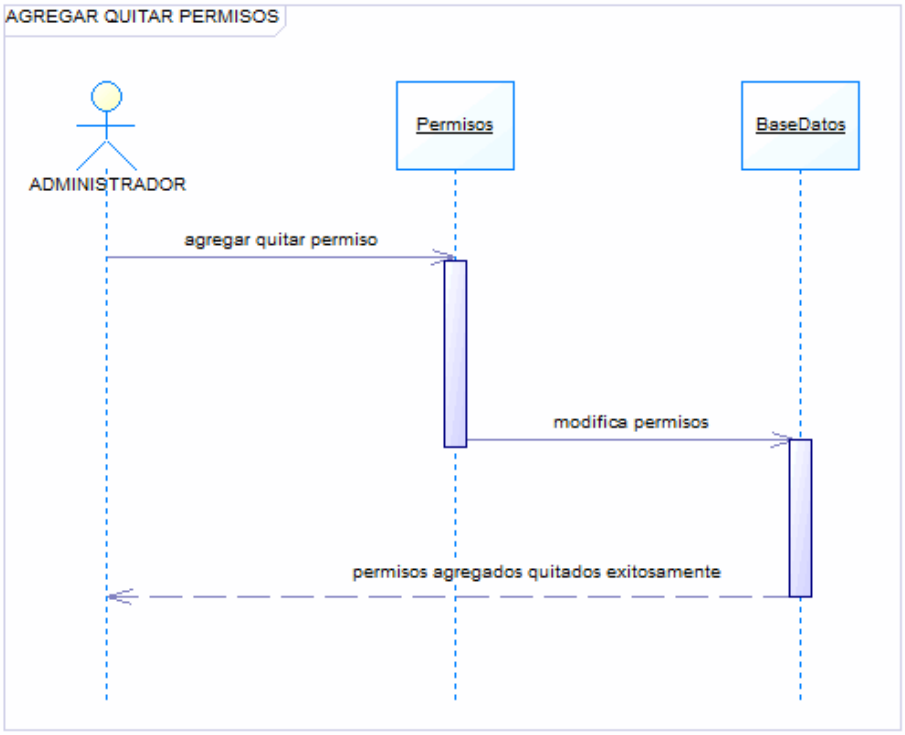
Elaborado por: Byron Andrango y Diego Jácome

Figura 31: Diagrama Secuencia Consultar Registro



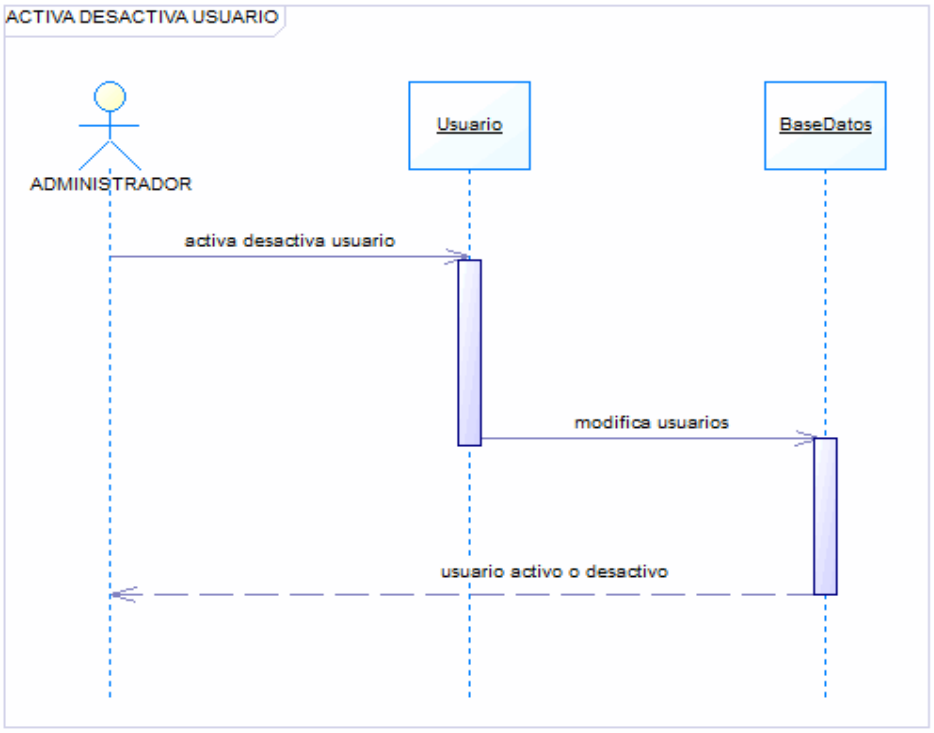
Elaborado por: Byron Andrango y Diego Jácome

Figura 32: Diagrama Secuencia Agregar Quitar Permisos



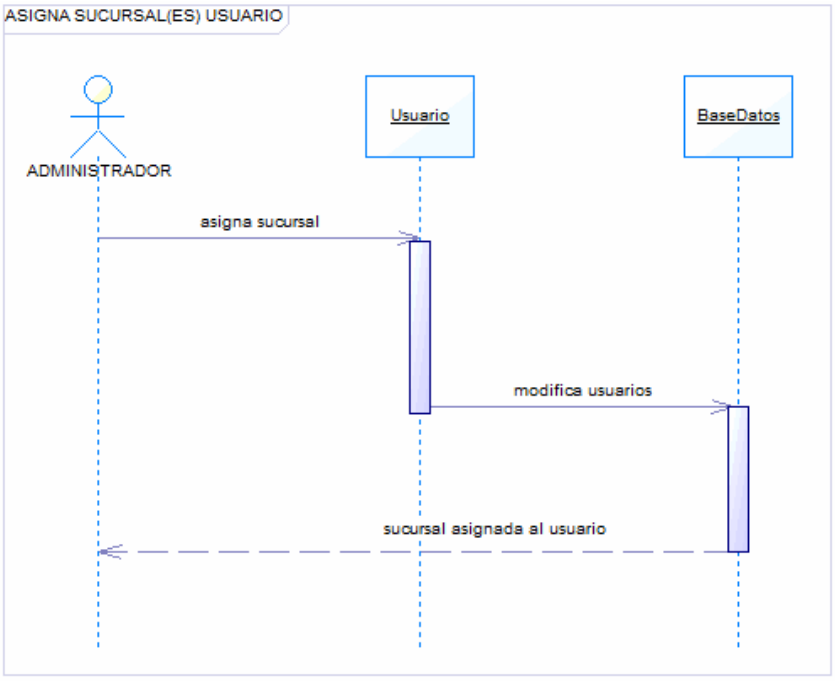
Elaborado por: Byron Andrango y Diego Jácome

Figura 33: Diagrama Secuencia Activa Desactiva Usuario



Elaborado por: Byron Andrango y Diego Jácome

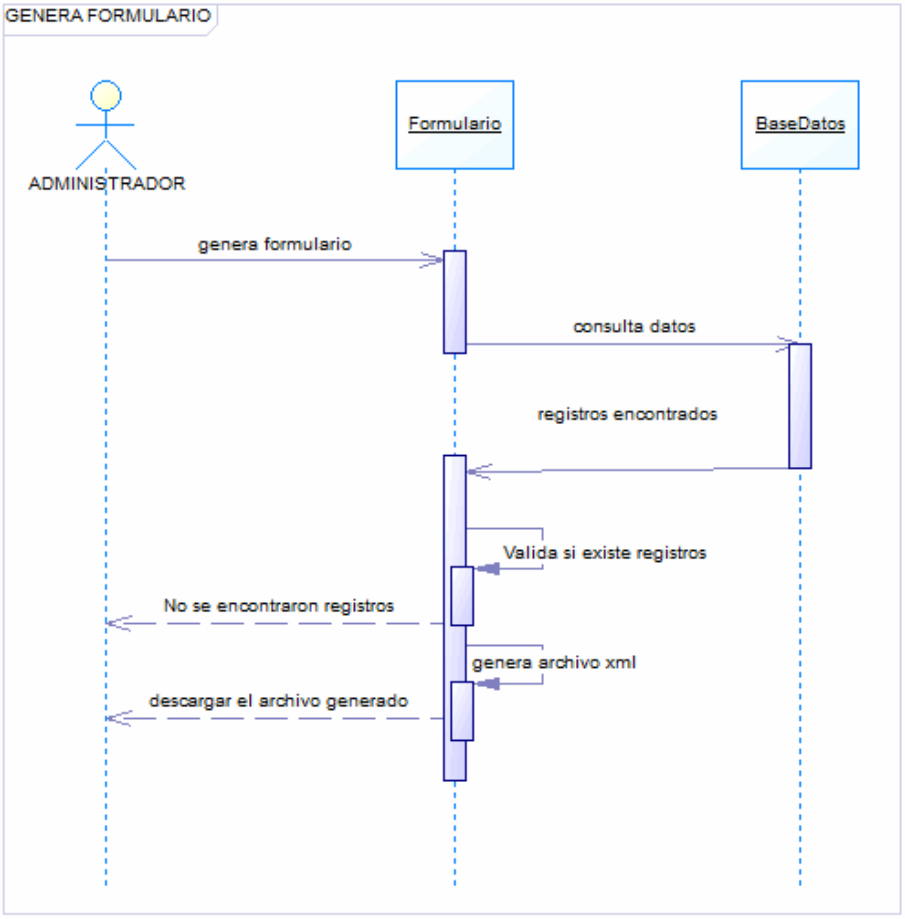
Figura 34: Diagrama Secuencia Asignar Sucursal Usuario



Elaborado por: Byron Andrango y Diego Jácome

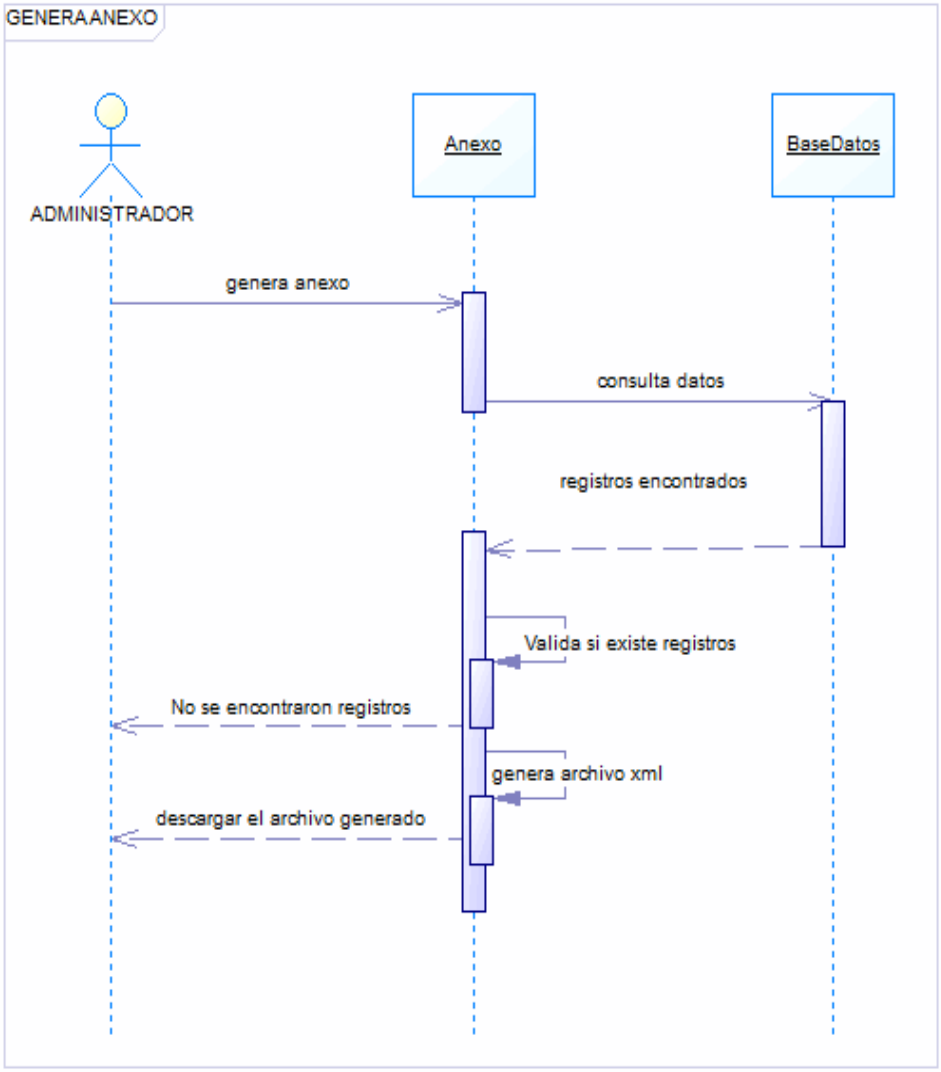
- **Diagramas de Secuencia Usuario Empleado**

Figura 35: Diagrama Secuencia Generar Formulario



Elaborado por: Byron Andrango y Diego Jácome

Figura 36: Diagrama Secuencia Genera Anexo

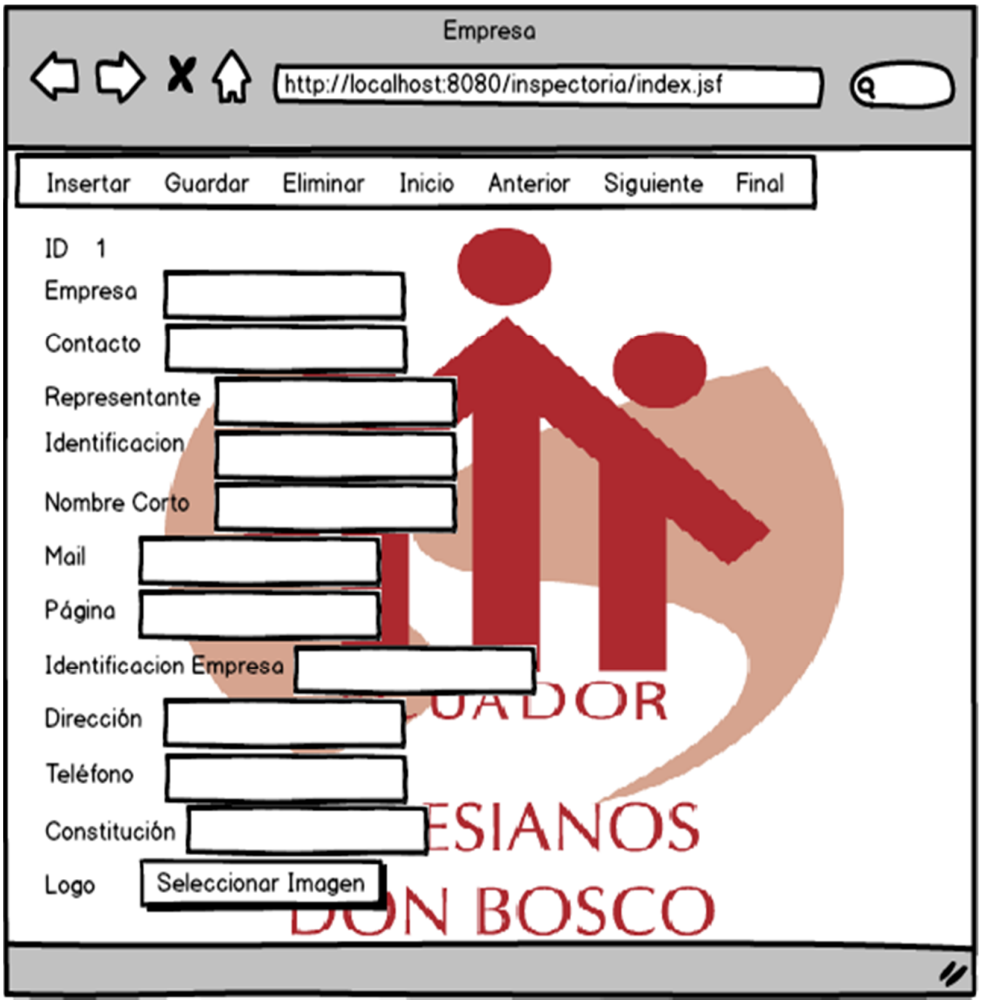


Elaborado por: Byron Andrango y Diego Jácome

2.8. Diseño de interfaces de usuario

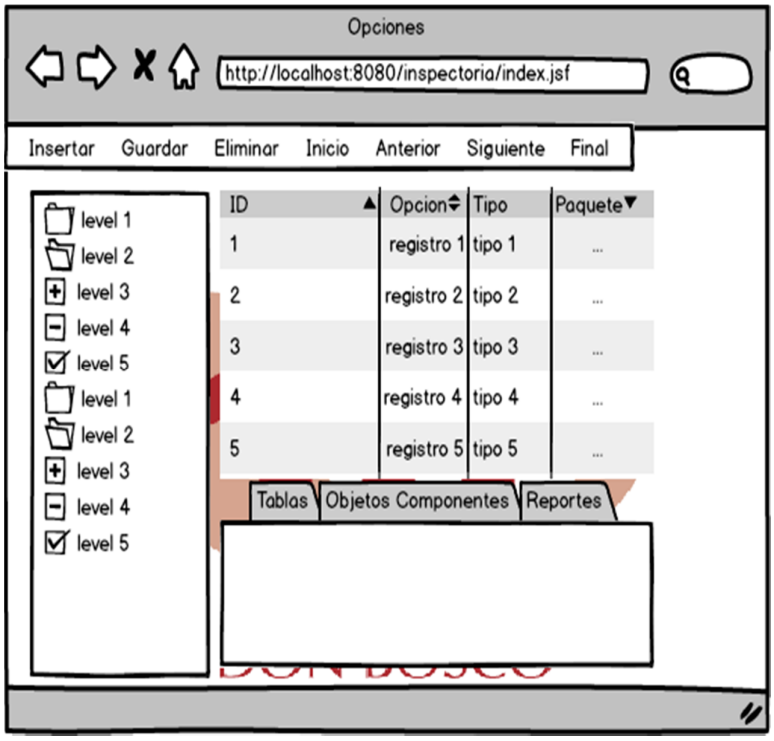
2.8.1. Diseño de interfaces de usuario Módulo Sistema

Figura 37: Interface Empresa



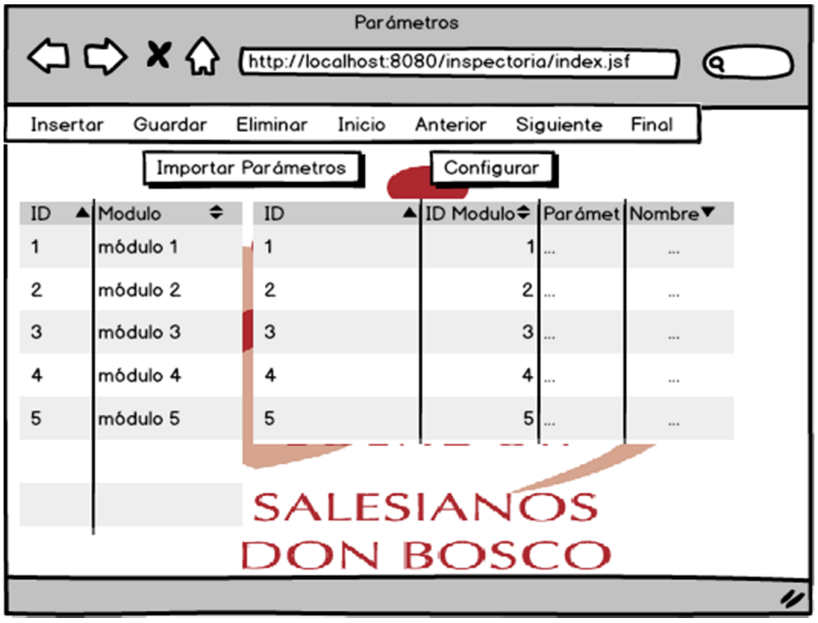
Elaborado por: Byron Andrango y Diego Jácome

Figura 38: Interface Opciones



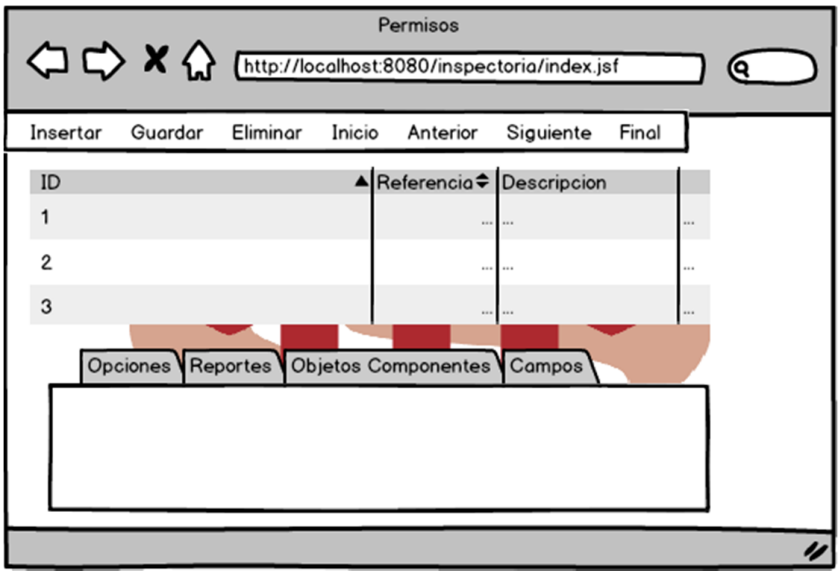
Elaborado por: Byron Andrango y Diego Jácome

Figura 39: Interface Parámetros



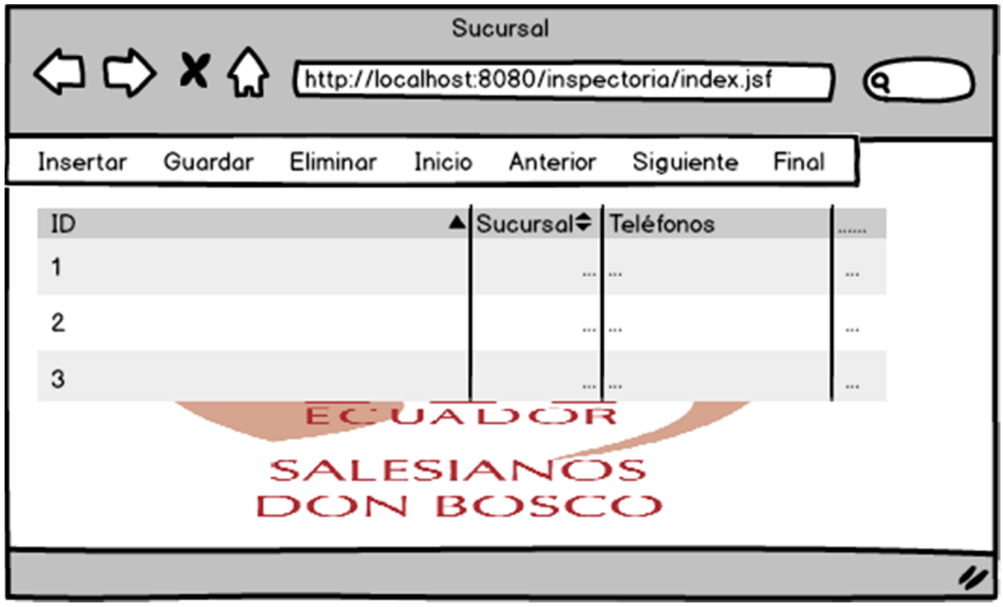
Elaborado por: Byron Andrango y Diego Jácome

Figura 40: Interface Permisos



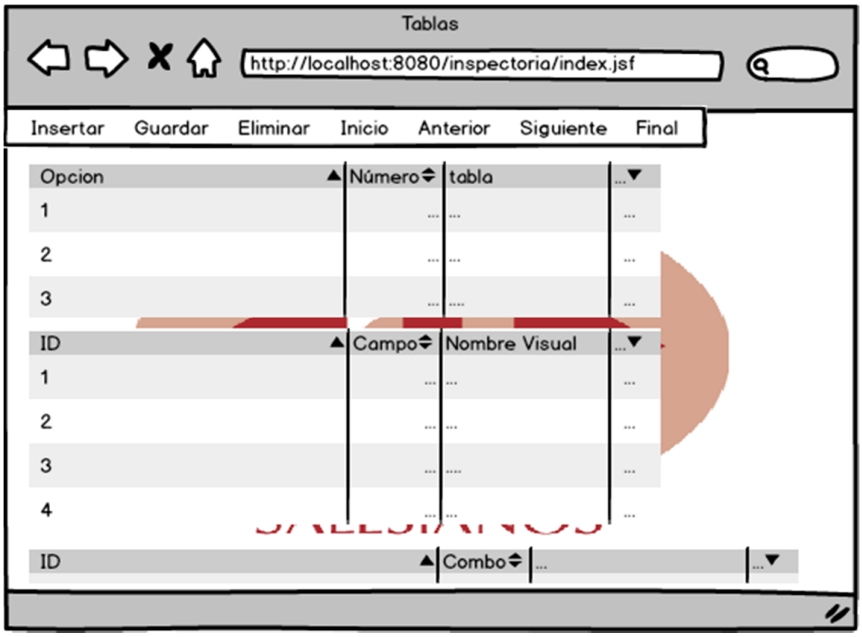
Elaborado por: Byron Andrango y Diego Jácome

Figura 41: Interface Sucursal



Elaborado por: Byron Andrango y Diego Jácome

Figura 42: Interface Tablas



Elaborado por: Byron Andrango y Diego Jácome

Figura 43: Interface Usuario

Elaborado por: Byron Andrango y Diego Jácome

2.8.2. Diseño de interfaces de usuario Módulo S. R. I

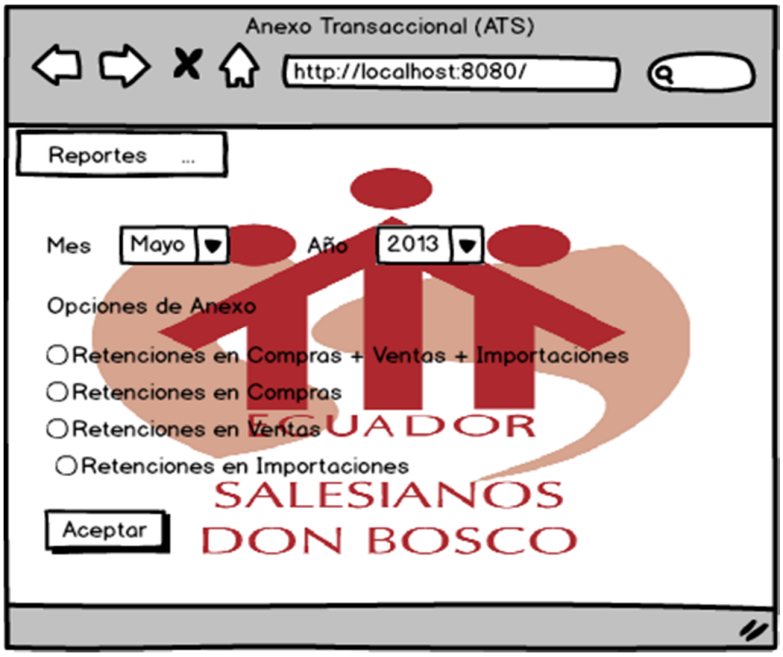
Figura 44: Interface Anexo Relación de Dependencia

Elaborado por: Byron Andrango y Diego Jácome

Figura 45: Interface Retenciones Otros Conceptos

Elaborado por: Byron Andrango y Diego Jácome

Figura 46: Interface Anexo Transaccional ATS



Elaborado por: Byron Andrango y Diego Jácome

Figura 47: Interface Fecha Pago Formulario

Elaborado por: Byron Andrango y Diego Jácome

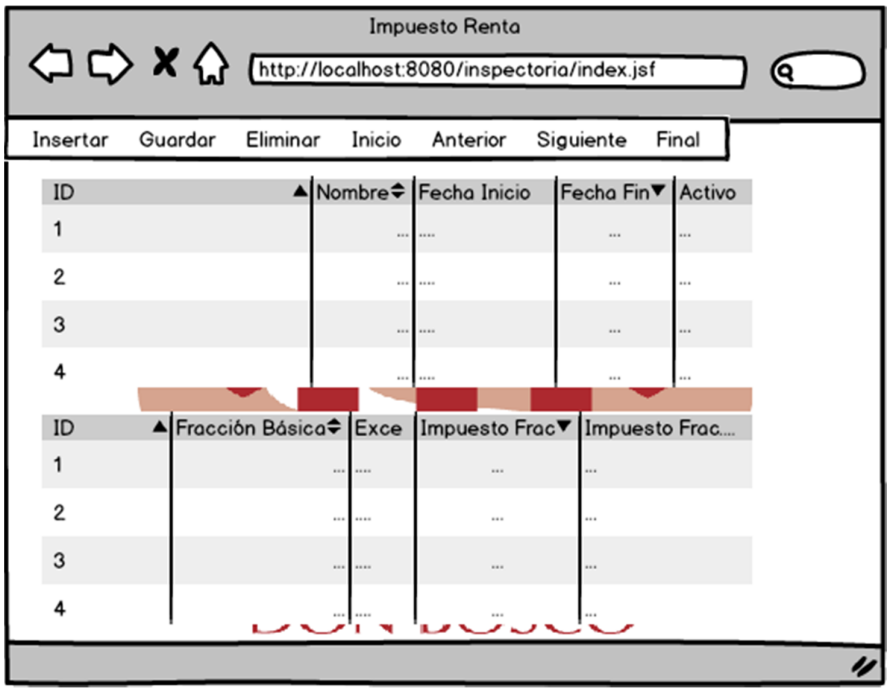
Figura 48: Interface Formulario 103

Elaborado por: Byron Andrango y Diego Jácome

Figura 49: Interface Formulario 104

Elaborado por: Byron Andrango y Diego Jácome

Figura 50: Interface Impuesto Renta



Elaborado por: Byron Andrango y Diego Jácome

Figura 51: Interface Interés Mora

Elaborado por: Byron Andrango y Diego Jácome

Figura 52: Interface Tipo Sustento Tributario

Elaborado por: Byron Andrango y Diego Jácome

CAPÍTULO 3
CONSTRUCCIÓN Y PRUEBAS

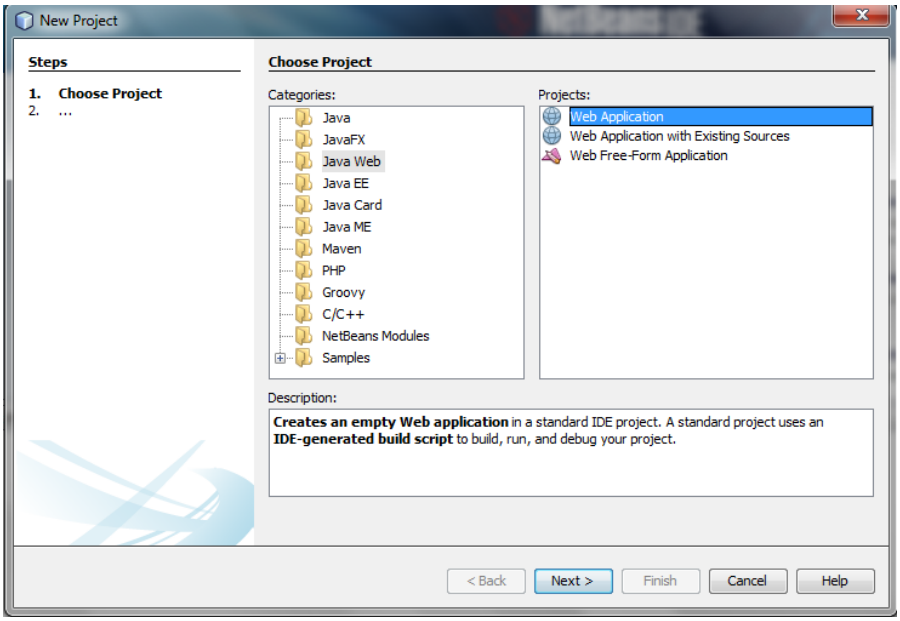
3.1. Configuraciones

3.1.1. Introducción a la configuración del Framework

Una vez instaladas las herramientas necesarias, se debe crear un nuevo proyecto web bajo el framework JavaServer Faces y comenzar a desarrollar una aplicación web. En las siguientes secciones, se explicará cómo implementar una aplicación web con el framework, y cómo desplegar la aplicación en un servidor de aplicaciones para verla en un navegador web.

Un nuevo proyecto Web se puede crear utilizando Netbeans. Esto se logra siguiendo los pasos:
En el menú de Netbeans, seleccionar File. New Project. Ahí se elige la categoría Java Web y seleccionar Web Application, tal como muestra la figura.

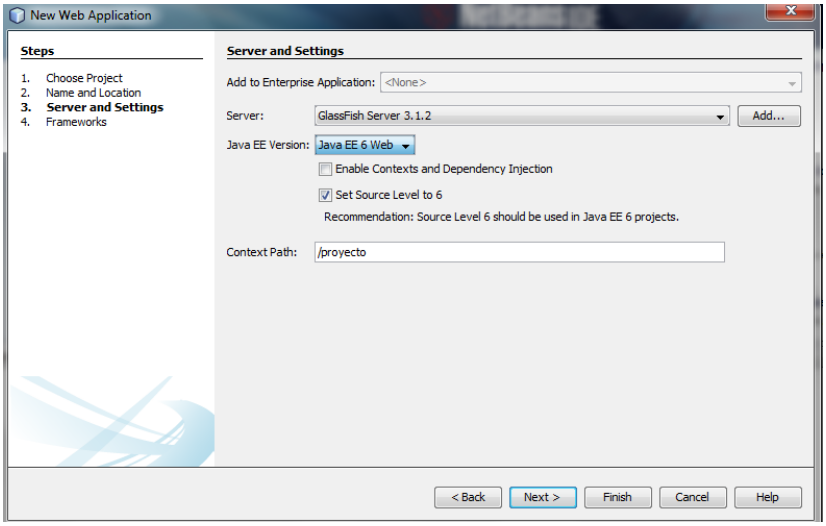
Figura 53: Nuevo Web Project



Elaborado por: Byron Andrango y Diego Jácome

Presionar siguiente y darle un nombre al proyecto y su localización, después se debe seleccionar el servidor web de aplicaciones que por defecto es GlassFish Server, y la versión de Java EE, tal como muestra en la siguiente figura.

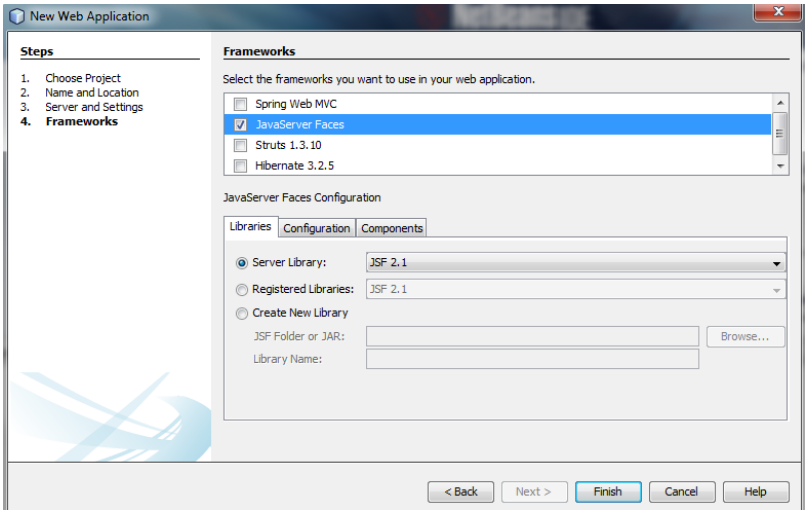
Figura 54: Selección Servidor GlassFish



Elaborado por: Byron Andrango y Diego Jácome

Después seleccionar el framework JavaServer Faces y luego dar click en el botón finalizar, tal como muestra la siguiente figura.

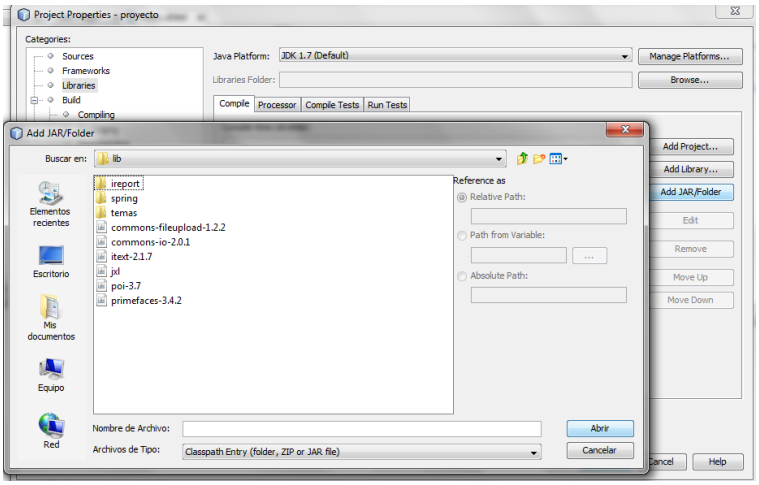
Figura 55: Selección Framework JavaServer Faces



Elaborado por: Byron Andrango y Diego Jácome

Importar las librerías y configurar los archivos web del proyecto, para esto es necesario dar click derecho sobre el proyecto creado, elegir la opción Propiedades, a continuación se elige la opción Librerías y presionar el botón Add JAR/Folder y añadir todos los .jar que se encuentran en la carpeta lib, tal como se muestra en la siguiente figura.

Figura 56: Importación de librerías al nuevo proyecto web



Elaborado por: Byron Andrango y Diego Jácome

3.1.2. Configuración del archivo web.xml

El archivo web.xml se encuentra en la carpeta WEB-INF/ y describe cómo una aplicación web debe ser desplegada. Si se desea construir una aplicación con el framework, se tiene que configurar dicho archivo correctamente, con el fin de que se relacionen bien los componentes del framework como los servlets, los listeners, y los filtros.

A continuación se muestran ejemplos de configuraciones en el archivo web.xml:

- Servlet para poder utilizar en el proyecto la librería de jsf.

```
<servlet>
    <servlet-name>Faces Servlet</servlet-name>
    <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>*.jsf</url-pattern>
</servlet-mapping>
```

- Configuración del sesión-timeout en 30 minutos para el cierre de sesión al detectar inactividad en el sistema.

```
<session-config>
    <session-timeout>
        30
    </session-timeout>
</session-config>
```

- Configuración de la página por default al abrir el sistema.

```
<welcome-file-list>
    <welcome-file>login.jsf</welcome-file>
</welcome-file-list>
```

- Servlet para utilizar la librería primefaces en jsf 2.0.

```
<servlet>
    <servlet-name>Resource Servlet</servlet-name>
    <servlet-class>org.primefaces.resource.ResourceServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>Resource Servlet</servlet-name>
    <url-pattern>/primefaces_resource/*</url-pattern>
</servlet-mapping>
```

- Filtro para utilizar el fileupload de primefaces y permitir subir archivos con un tamaño máximo de 51200 bytes.

```
<filter>
    <filter-name>PrimeFaces FileUpload Filter</filter-name>
    <filter-class>
        org.primefaces.webapp.filter.FileUploadFilter
    </filter-class>
    <init-param>
        <param-name>thresholdSize</param-name>
        <param-value>51200</param-value>
    </init-param>
</filter>
<filter-mapping>
    <filter-name>PrimeFaces FileUpload Filter</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
</filter-mapping>
```

- Listeners para poder utilizar la librería de spring-security con jsf 2.0.

```
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/applicationContext.xml</param-value>
</context-param>

<context-param>
    <param-name>javax.faces.FACELETS_SKIP_COMMENTS</param-name>
    <param-value>true</param-value>
</context-param>

<filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>
<filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

- Página de error configurada al producirse un inicio de sesión inválido.

```
<error-page>
    <exception-type>org.springframework.security.access.AccessDeniedException</exception-type>
    <location>/login.jsf</location>
</error-page>
```

3.1.3. Configuración del archivo config.properties

> Los ficheros .properties son archivos que permiten almacenar variables de configuración de nuestra aplicación. En la práctica, no deja de ser un fichero de texto donde se almacena por cada línea, un par clave valor, indicando el nombre de la variable y su valor (V3RGU1, 2012).

En la aplicación se encuentra el fichero config.properties donde se encuentran nuestras variables de configuración para conectar a la base de datos, el servidor web de aplicaciones y el tema inicial de la aplicación, de esta forma en un entorno de producción este permitirá que sea fácilmente reconfigurable.

```
#Código config.properties
#Nombre del recursojdbc
recursojdbc=inspectorialocal
#Nombre del servidor de aplicaciones web tomcat o glassfish
servidorWeb=glassfish
#Nombre del tema para la página login.xhtml
temaInicial=blitzer
```

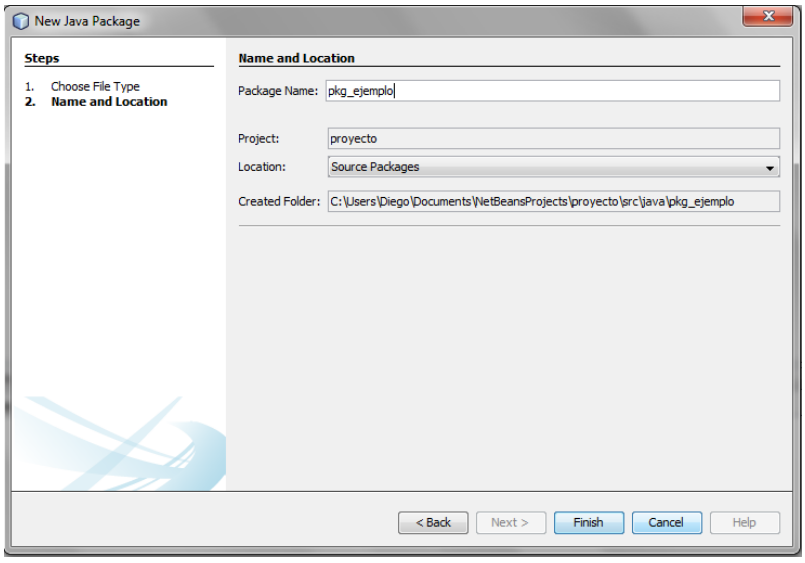
3.2. Integración de los Módulos

La siguiente parte de la guía, es una referencia de cómo se pueden programar e integrar las pantallas y los archivos Java utilizando el framework, para lo cual se pondrán como ejemplo pequeños programas que ilustran una manera óptima y ágil en el desarrollo de aplicaciones.

3.2.1. Creación de pantallas

Cabe recalcar que para crear pantallas utilizando el framework todos los componentes se los debe crear de manera dinámica es decir desde una clase Java.
Para crear una pantalla primero se debe crear un nuevo paquete esto se logra dando click derecho sobre la carpeta Source Packages de nuestro proyecto y después se seleccionar New /Java Package y a continuación ingresar un nombre para nuestro paquete, se recomienda utilizar estándares para nombrar a todos los paquetes, tal como muestra en la siguiente figura.

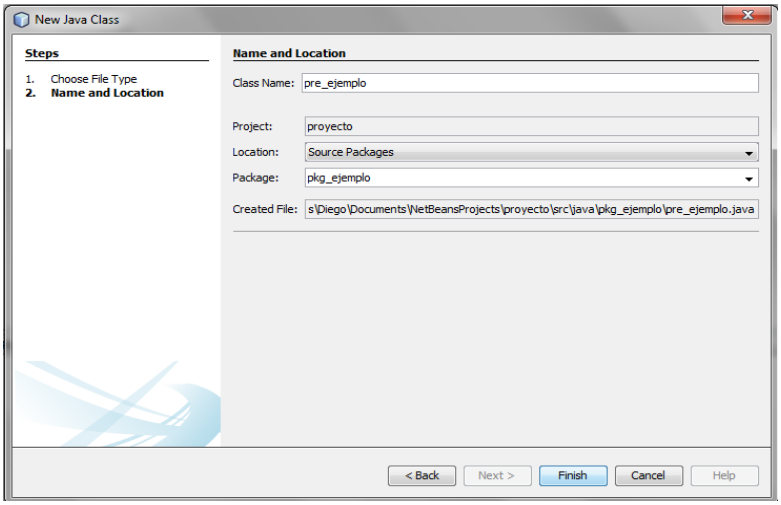
Figura 57: Creación nuevo paquete



Elaborado por: Byron Andrango y Diego Jácome

A continuación crear la pantalla en la cual se visualizara en el navegador web mediante la integración con el framework, para lo cual se necesita implementar una clase Java en nuestro paquete, esto se logra dando click derecho en el paquete y se selecciona New/Java Class y a continuación se ingresa el nombre de la nueva clase, se recomienda utilizar estándares para nombrar a todas las clases, tal como muestra en la siguiente figura.

Figura 58: Creación nueva clase java



Elaborado por: Byron Andrango y Diego Jácome

Para que la clase creada pueda ser integrada con el framework, esta debe heredar de la clase abstracta Pantalla.java la cual se encuentra en el paquete sistema del Proyecto, y solicitará que se implemente los métodos abstractos que posee la clase Pantalla, quedando como se muestra en el siguiente código.

```
package pkg_ejemplo;

import sistema.Pantalla;
public class pre_ejemplo extends Pantalla {

    @Override
    public void insertar() {
    }

    @Override
    public void guardar() {
    }

    @Override
    public void eliminar() {
    }
}
```

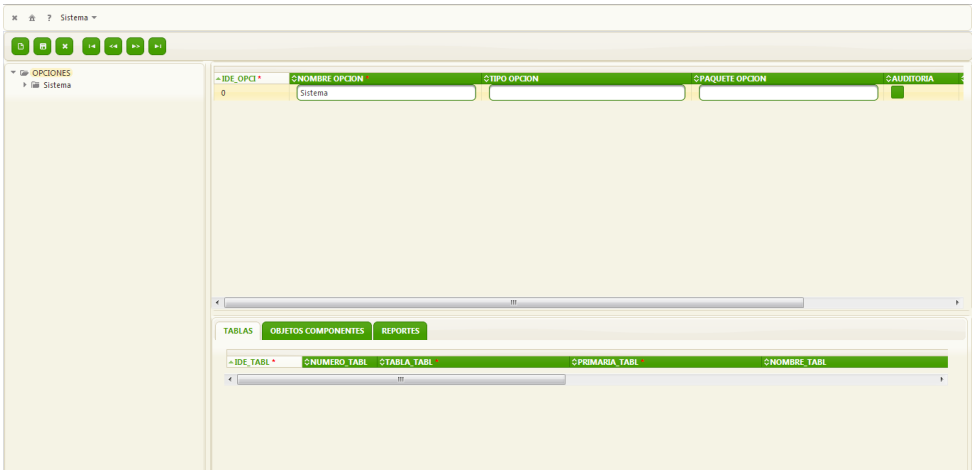
Estos tres métodos que son necesarios implementarlos son importantes, ya que en la barra de botones que automáticamente genera la clase Pantalla.java se crean los botones de Insertar, Guardar y Eliminar los cuales ejecutan cada uno de estos métodos.

3.2.2. Integración de pantallas al framework

Para integrar una pantalla al framework se debe realizar los siguientes pasos.

1) Una vez que el usuario inicia la sesión en el sistema se debe dirigir a la barra de menú y seleccionar Sistema/Opciones, y se abrirá la pantalla tal como muestra en la siguiente figura.

Figura 59: Creación de Opción en el Sistema



Elaborado por: Byron Andrango y Diego Jácome

2) Esta pantalla contiene todas las configuraciones de las pantallas que posee nuestra aplicación web, básicamente forma la barra de menú y las configuraciones para la integración de las pantallas que se podrán visualizar en el navegador web.

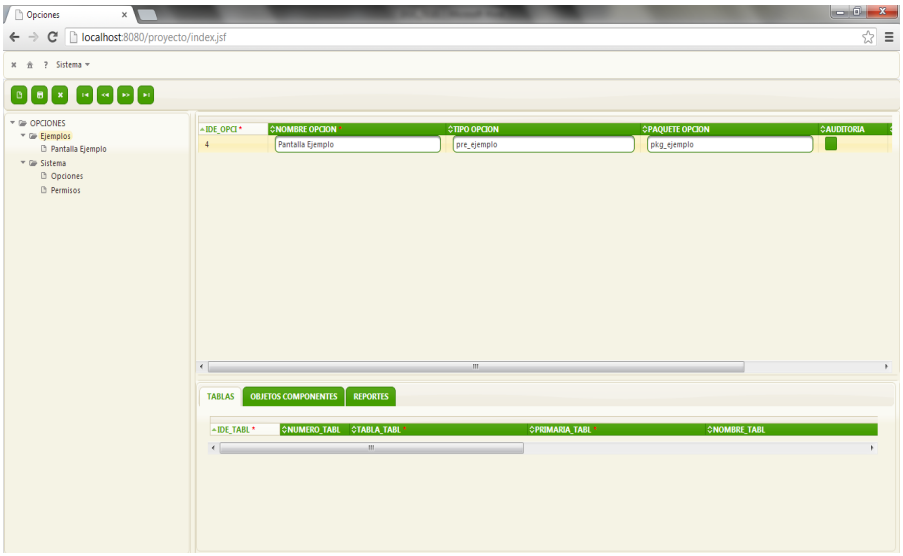
   En el panel de la izquierda se tiene un árbol recursivo el cual sirve para formar la barra de menú del sistema, aquí se puede crear nuevas opciones.

En el panel de la derecha se tiene la parte de configuración de las pantallas, a continuación se configurará la integración de nuestra clase pre_ejemplo.java que se encuentra en el paquete pkg_prueba.

- **Nombre opción:** En este campo se coloca el nombre de nuestra pantalla la cual se visualizara en la barra de menú.
- **Tipo opción:** En este campo se coloca el nombre de nuestra clase java a la cual se va a enlazar la opción.
- **Paquete opción:** En este campo se coloca el nombre de nuestro paquete de la clase java a la cual se va a enlazar la opción.

Si se quiere agregar sub niveles a nuestro menú simplemente se llena el campo **Nombre opción.** El resultado se lo muestra en la siguiente figura.

Figura 60: Ejemplo de creación de Opción en el Sistema



Elaborado por: Byron Andrango y Diego Jácome

3.2.3. Gestión de permisos a pantallas

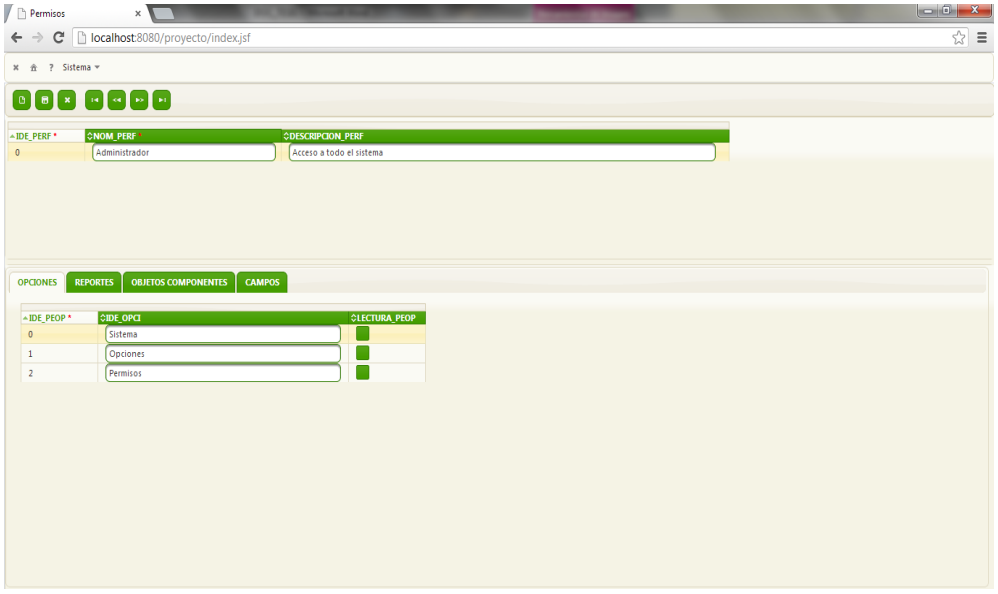
Esta sección proporciona el manejo de permisos, el cual se lo realiza por perfiles, ya que con esto se puede crear varios perfiles de usuarios de acuerdo a las necesidades que se presenten. El framework controla los siguientes permisos:

- Opciones: Permite controlar el acceso a las pantallas que existen en nuestro proyecto, además también se puede controlar que una pantalla sea de solo lectura con lo que se controla la integridad de los datos ya que no cualquier usuario puede modificar la información.
- Reportes: Permite controlar la visualización de los diferentes reportes que conforman el sistema, logrando que los reportes sean visualizados solo por los usuarios que requieren la información y resultados que proporcionan los reportes.
- Objetos o Componentes de Pantallas: Permite controlar el acceso a cualquier componente de una pantalla, un componente es un botón, una lista, un cuadro de texto, en si a cualquier elemento que se visualice en una pantalla.
- Campos: Permite controlar el acceso a cualquier campo de una tabla o formulario que se encuentre en una pantalla, logrando proteger información que un usuario no necesita visualizar o modificar.

Para dar permisos a una pantalla al framework se debe realizar los siguientes pasos:

1) Una vez que el usuario se encuentre autenticado en el sistema ir a la barra de menú y seleccionar Sistema/Permisos, y se abrirá la pantalla tal como muestra en la siguiente figura.

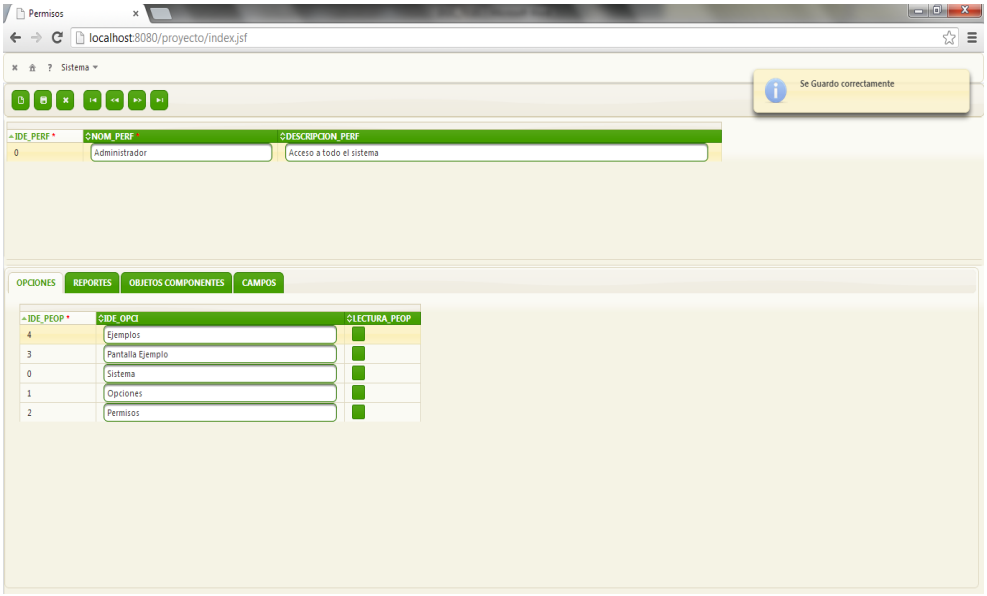
Figura 61: Creación de Permisos en el Sistema



Elaborado por: Byron Andrango y Diego Jácome

2) Esta pantalla contiene todos los perfiles del sistema los cuales podrán ser asignados a usuarios, en el panel superior se tiene el nombre del perfil y una observación, en el panel inferior se controla los permisos que posee el framework, para dar permiso a nuestra pantalla se selecciona la pestaña OPCIONES, click en el botón insertar para crear una nueva fila, posteriormente en el cuadro de texto que se comporta como un Autocompletar digitando el nombre de nuestra opción, si nuestra opción se encuentra en un sub nivel se debe de crear una fila por cada nivel superior de la opción. El resultado se lo muestra en la siguiente figura.

Figura 62: Ejemplo de Creación de Opción en el Sistema

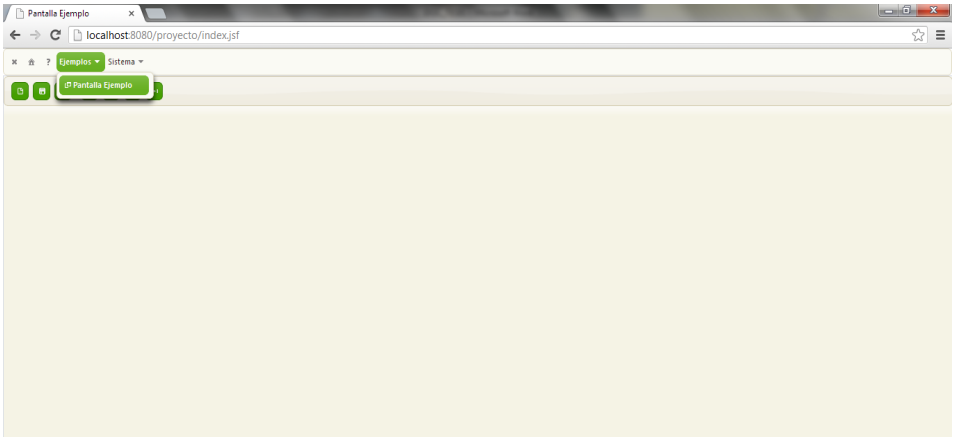


Elaborado por: Byron Andrango y Diego Jácome

3.2.4. Visualización de pantallas

Una vez creada nuestra pantalla y configurado el permiso a un perfil, cuando el usuario se encuentre autenticado en el sistema se debe ir a la barra de menú y aparecerá nuestra opción, tal como muestra la figura.

Figura 63: Visualización de la pantalla creada en el Sistema



Elaborado por: Byron Andrango y Diego Jácome

Como la clase de ejemplo no contiene ningún componente y nada de código el resultado es una pantalla con la barra de menú y el panel inferior vacío para agregar componentes.

Este proceso es el necesario para integrar pantallas al framework, y depende de la experiencia del programador y de los requerimientos para que una pantalla contenga varios componentes y procesos.

3.3. Pruebas

3.3.1. Resultado de las stress

- **Pruebas de Stress del Módulo de Autenticación**

Se realizaron las pruebas de stress sobre el Módulo de Autenticación, para las cuales se realizaron las pruebas con los siguientes parámetros:

Tabla 22: Parámetros para las pruebas de carga

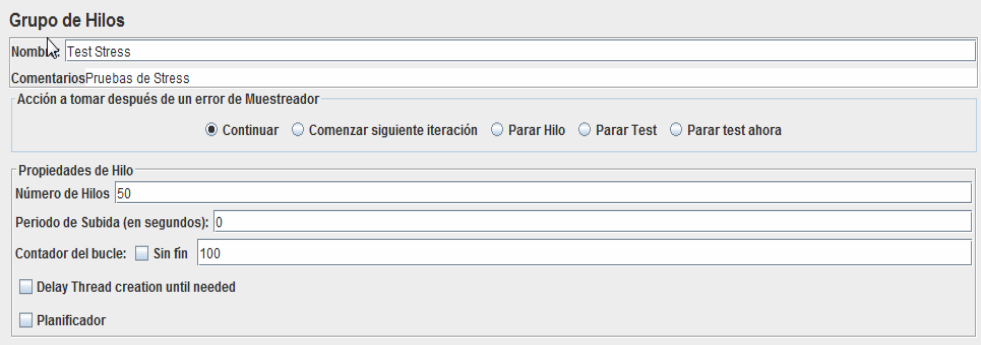
Prueba	Usuarios	Repeticiones	Transacciones
1	50	100	5000
2	140	10	1400
3	150	30	4500

Elaborado por: Byron Andrango y Diego Jácome

Prueba de stress 1

Para poder realizar la prueba 1 de stress se realizó con 50 usuarios y 100 repeticiones de login al sistema como se muestra en la figura.

Figura 64: Configuración de hilos e intentos prueba stress 1



Elaborado por: Byron Andrango y Diego Jácome

Teniendo como resultados los siguientes datos:

Tabla 23: Resultados de la prueba de stress 1

Hilos	50
Repeticiones	100
Hora Inicio	20:16:23
Hora Fin	20:32:03
Tiempo Proceso	0:15:39
Transacciones	5000
Peticiones	35000
Rate (tx/s)	5,414771

Elaborado por: Byron Andrango y Diego Jácome

Los resultados de las transacciones en el sistema son:

Tabla 24: Porcentajes de peticiones en prueba de stress 1

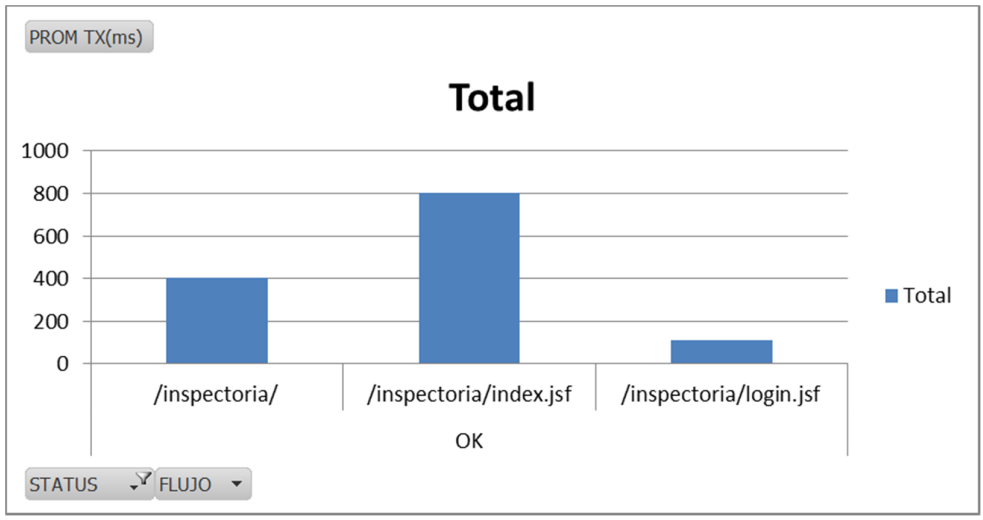
Transacciones	# Peticiones	%
OK	6834	19,53
Non HTTP response message: The target server failed to respond	92	0,26
Non HTTP response message: Connection to http://172.17.10.40:8080 refused	28041	80,12
Non HTTP response message: Connection reset	21	0,06
Internal Server Error	12	0,03
TOTAL	**35000**	**100,00**

Elaborado por: Byron Andrango y Diego Jácome

Los resultados se encuentran distribuidos de la siguiente manera:

Transacciones Exitosas

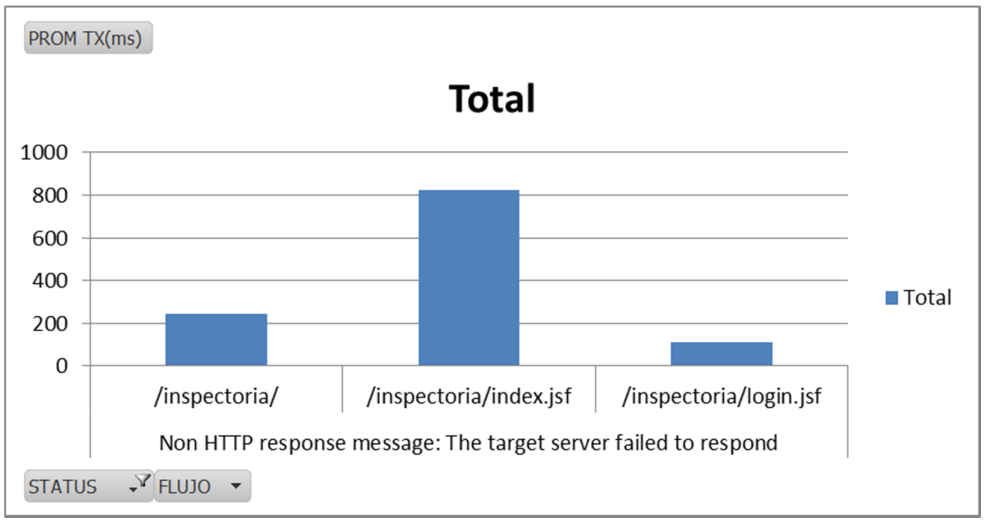
Figura 65: Transacciones Exitosas en prueba de stress 1



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: The target server failed to respond

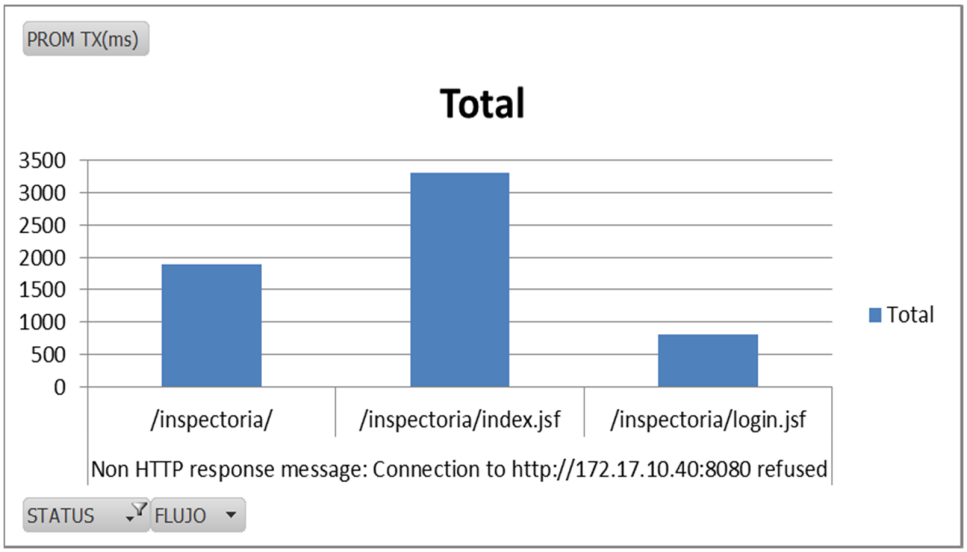
Figura 66: Transacciones Exitosas en prueba de stress 1



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: Connection to http://172.17.10.40:8080 refused

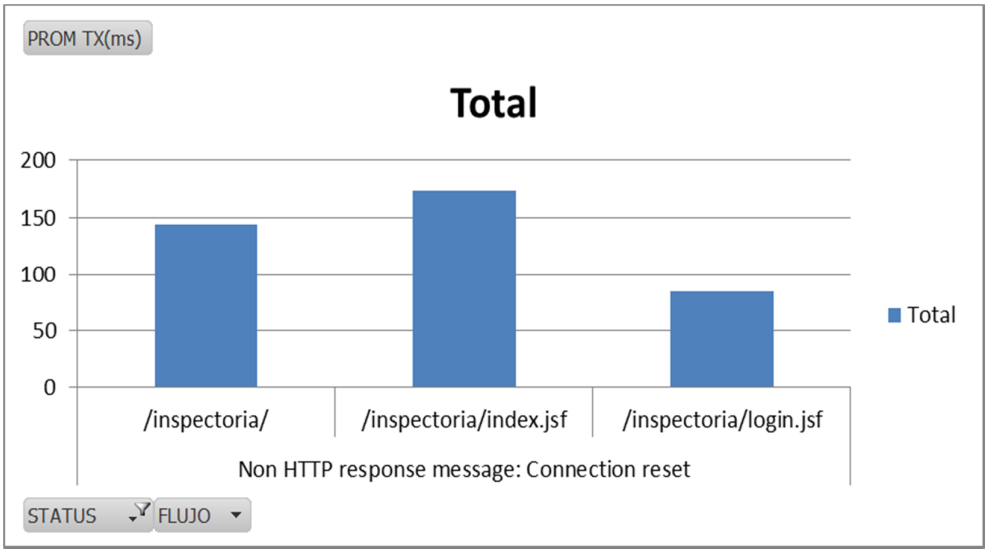
Figura 67: Transacciones Connection refuse en prueba de stress 1



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: Connection reset

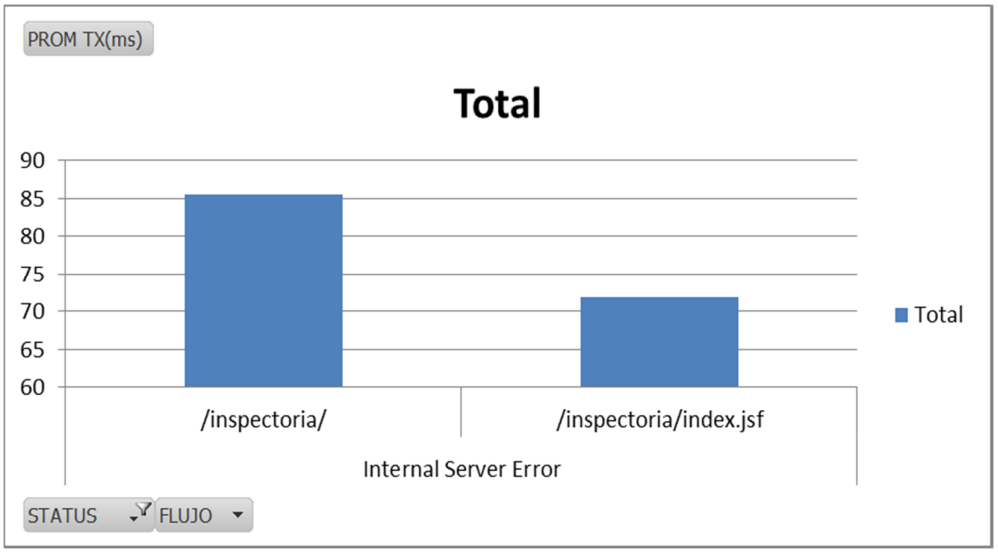
Figura 68: Transacciones Connection reset en prueba de stress 1



Elaborado por: Byron Andrango y Diego Jácome

Internal Server Error

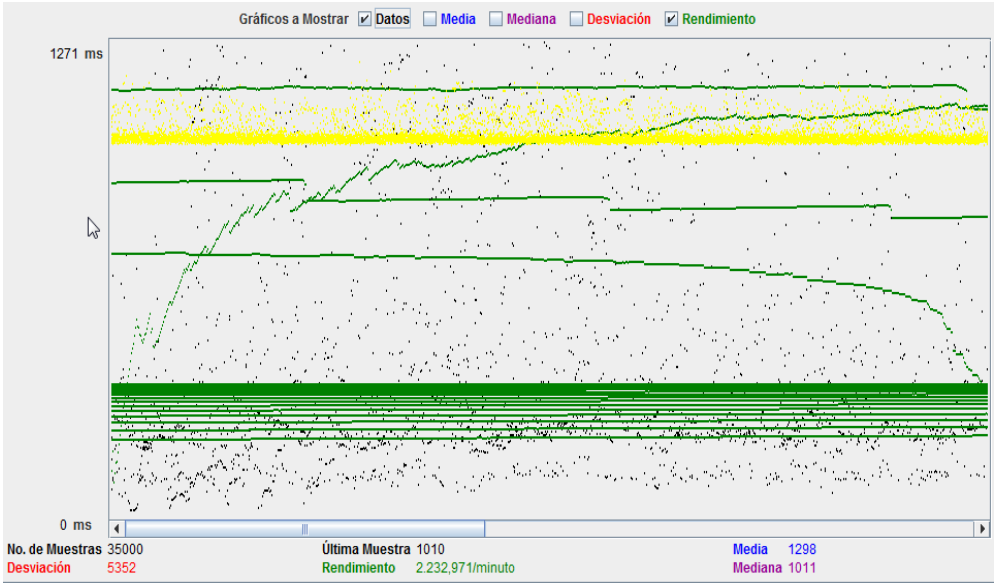
Figura 69: Transacciones Internal server en prueba de stress 1



Elaborado por: Byron Andrango y Diego Jácome

Distribución de datos

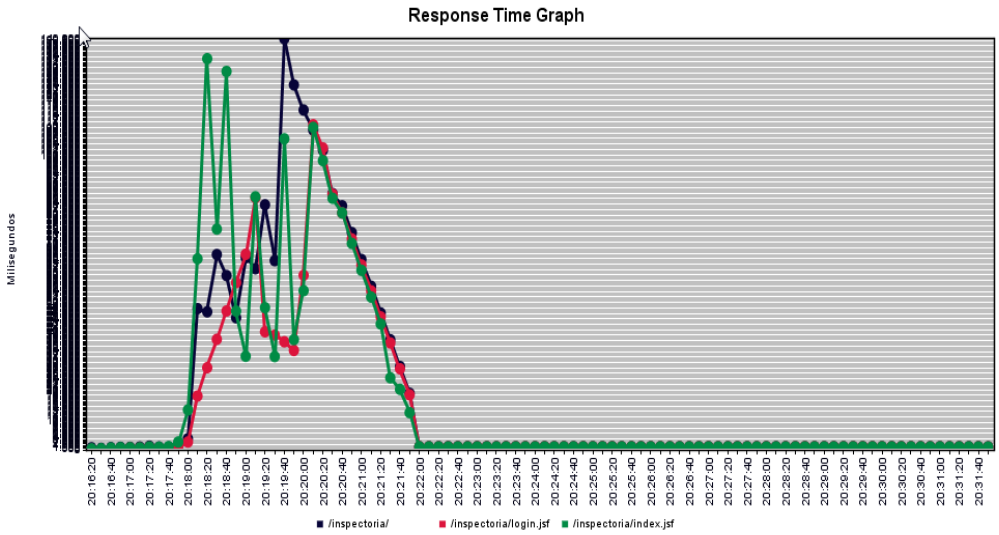
Figura 70: Gráfico de distribución de datos



Elaborado por: Byron Andrango y Diego Jácome

Tiempo de respuesta

Figura 71: Gráfico tiempo de respuesta



Elaborado por: Byron Andrango y Diego Jácome

Resultado de la prueba de stress 1:

Se realizó con los siguientes parámetros:

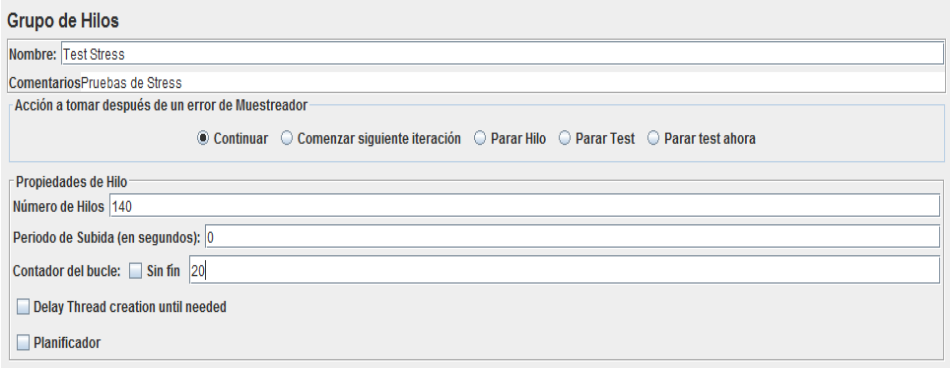
- 50 usuarios.
- 100 repeticiones.
- 5000 solicitudes de login.

El procesamiento de la solicitud duró 15:39 minutos, el objetivo de la prueba fue ver la capacidad y rendimiento de la aplicación en horas picos teniendo como resultado la caída del servidor de aplicaciones.

Pruebas de stress 2

Para poder realizar la prueba 2 de stress se realizó con 140 usuarios y 20 repeticiones de login al sistema como se muestra en la figura.

Figura 72: Configuración de hilos y repeticiones pruebas de stress 2



Elaborado por: Byron Andrango y Diego Jácome

Teniendo como resultados los siguientes datos:

Tabla 25: Resultados de prueba de stress 2

Hilos	140
Repeticiones	20
Hora Inicio	20:11:04
Hora Fin	20:14:09
Tiempo Proceso	0:03:05
Transacciones	2800
Peticiones	19600
Rate (tx/s)	15,30

Elaborado por: Byron Andrango y Diego Jácome

Los resultados de las transacciones en el sistema son:

Tabla 26: Porcentaje de peticiones exitosas prueba de stress 2

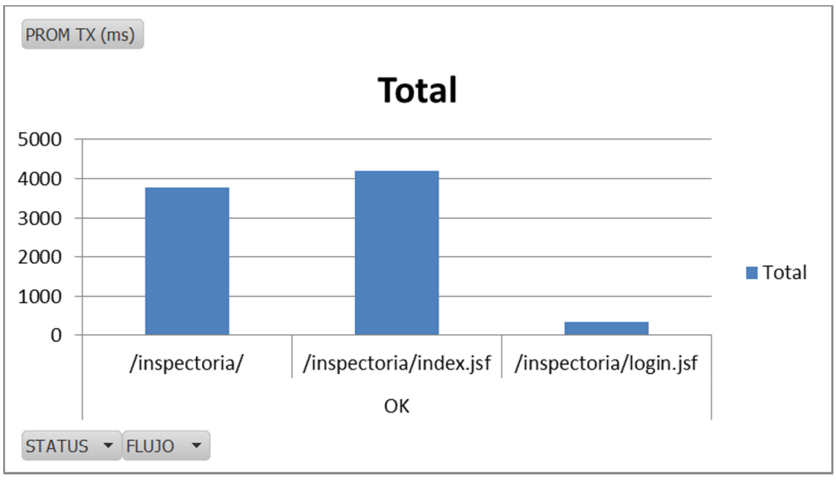
Transacciones	# Peticiones	%
OK	19600	100,00
TOTAL	**19600**	**100,00**

Elaborado por: Byron Andrango y Diego Jácome

Los resultados se encuentran distribuidos de la siguiente manera:

Transacciones Exitosas

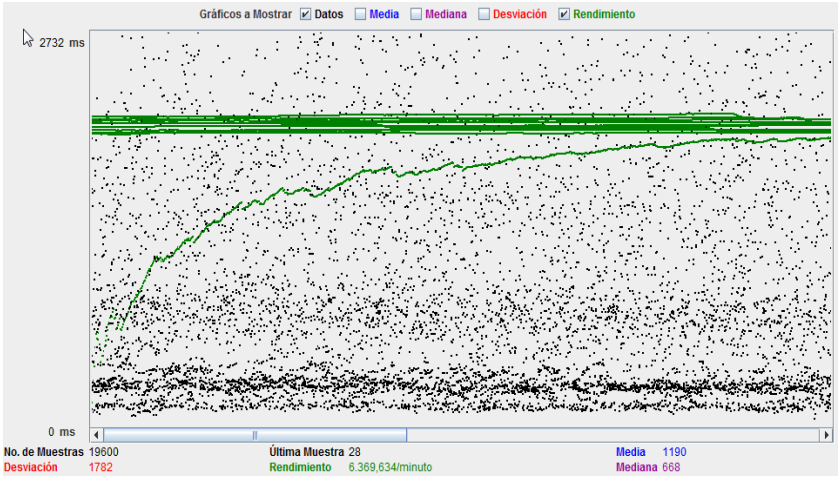
Figura 73: Transacciones Exitosas prueba de stress 2



Elaborado por: Byron Andrango y Diego Jácome

Distribución de datos

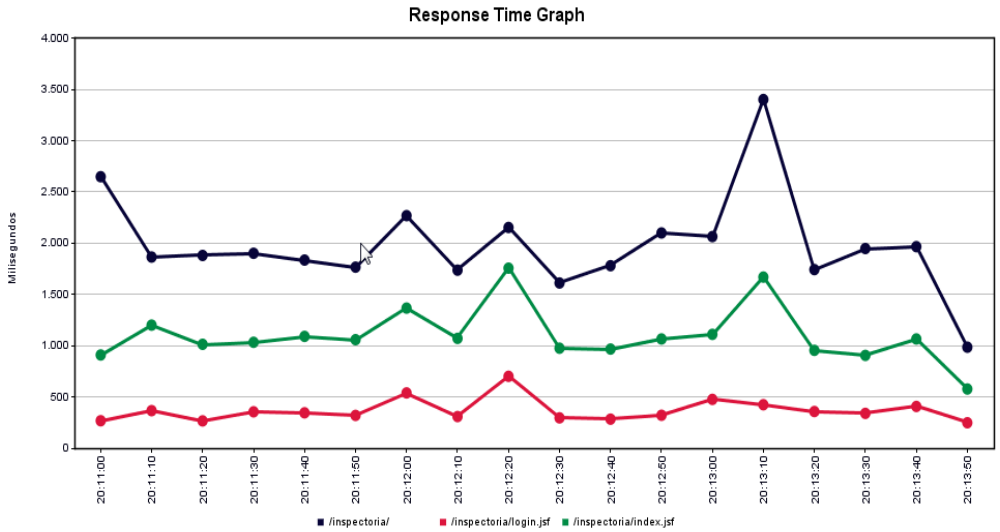
Figura 74: Gráfico distribución de datos prueba stress 2



Elaborado por: Byron Andrango y Diego Jácome

Tiempo de respuesta

Figura 75: Gráfico tiempo de respuesta prueba stress 2



Elaborado por: Byron Andrango y Diego Jácome

Resultados de la prueba de stress 2

Se realizó con los siguientes parámetros:

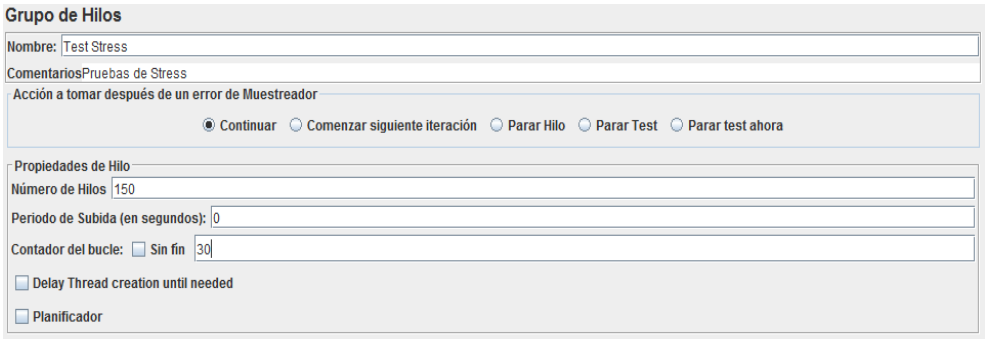
- 140 usuarios.
- 20 repeticiones.
- 2800 solicitudes de login.

El procesamiento de la solicitud duró 03:05 minutos, el objetivo de la prueba fue ver la capacidad y rendimiento de la aplicación en horas picos teniendo como resultado el correcto funcionamiento; ya que aumentaron los usuarios pero se disminuyeron los ciclos de prueba por cada usuario.

Pruebas de stress 3

Para poder realizar la prueba 3 de stress se realizó con 140 usuarios y 20 repeticiones de login al sistema como se muestra en la figura.

Figura 76: Configuración usuarios y repeticiones prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Teniendo como resultados los siguientes datos:

Tabla 27: Resultados de la prueba stress 3

Hilos	150
Repeticiones	30
Hora Inicio	19:49:01
Hora Fin	19:54:03
Tiempo Proceso	0:05:02
Transacciones	4500
Peticiones	31384
Rate (tx/s)	14,94

Elaborado por: Byron Andrango y Diego Jácome

Los resultados de las transacciones en el sistema son:

Tabla 28: Transacciones Exitosas o erróneas prueba stress 3

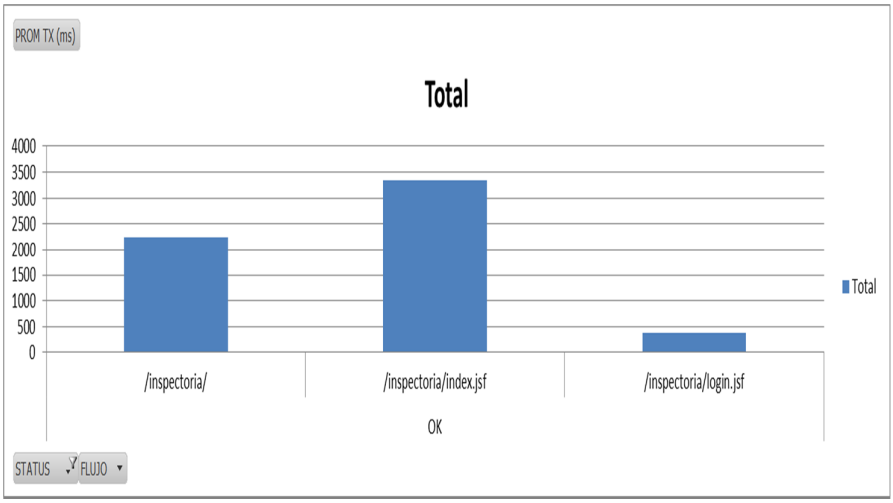
Transacciones	# Peticiones	%
OK	16028	51,07
Non HTTP response message: The target server failed to respond	346	1,10
Non HTTP response message: Premature end of Content-Length delimited message body (expected: 236; received: 0)	1	0,00
Non HTTP response message: Premature end of Content-Length delimited message body (expected: 217; received: 0)	1	0,00
Non HTTP response message: Premature end of Content-Length delimited message body (expected: 1503; received: 0)	1	0,00
Non HTTP response message: Connection to http://172.17.10.40:8080 refused	14942	47,61
Non HTTP response message: Connection reset	52	0,17
Internal Server Error	13	0,04
TOTAL	**31384**	**100,00**

Elaborado por: Byron Andrango y Diego Jácome

Los resultados se encuentran distribuidos de la siguiente manera:

Transacciones Exitosas

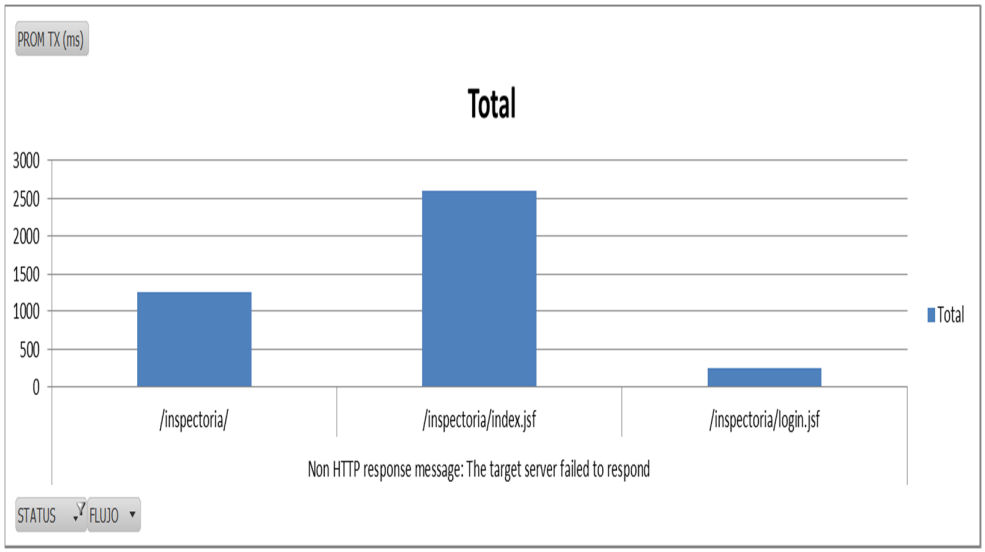
Figura 77: Transacciones Exitosas prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: The target server failed to respond

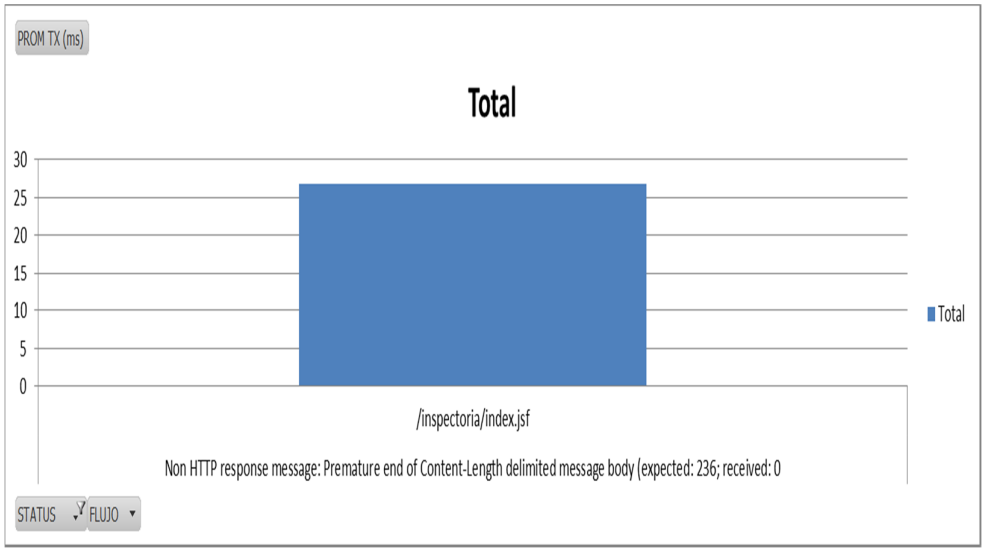
Figura 78: Server failed to respond prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: Premature end of Content-Length delimited message body (expected: 236; received: 0)

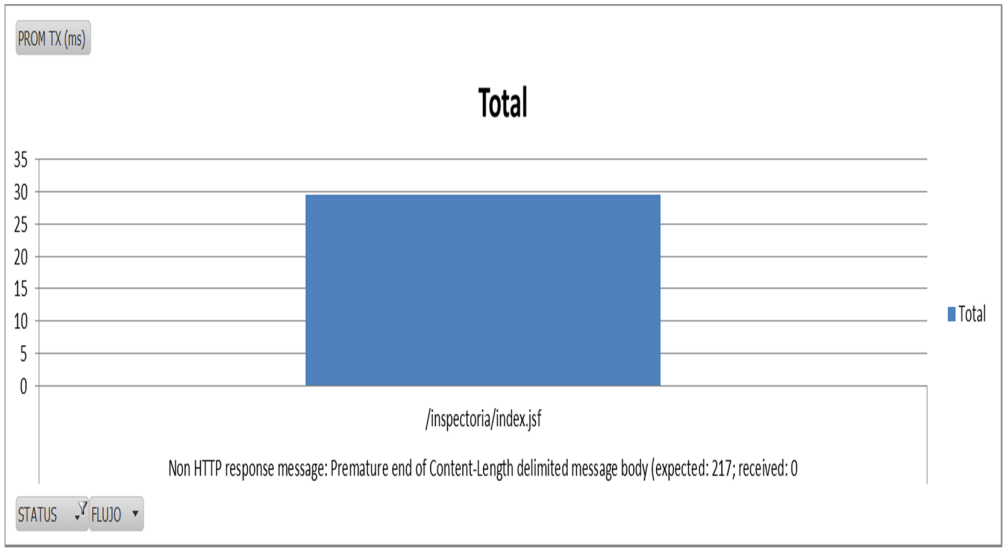
Figura 79: Demilited message body prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: Premature end of Content-Length delimited message body (expected: 217; received: 0)

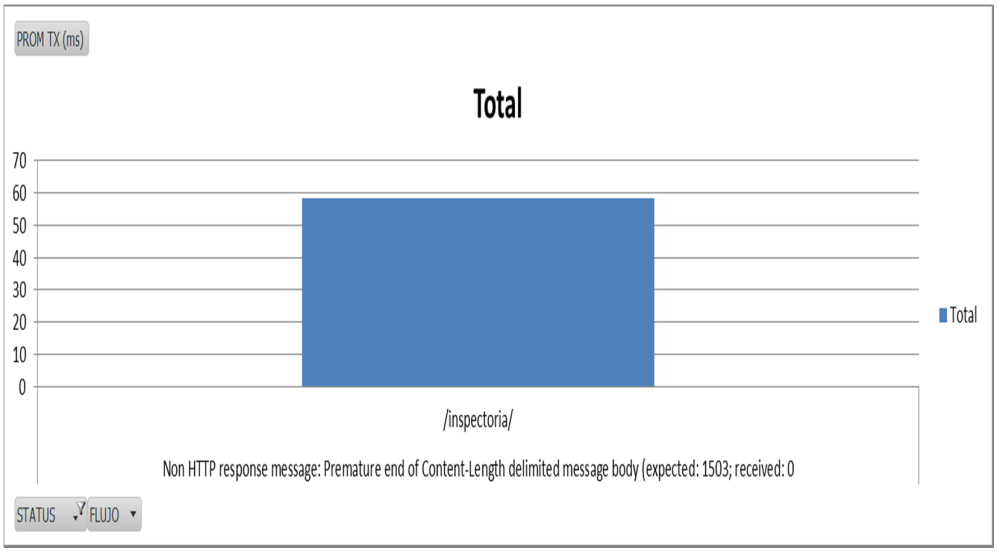
Figura 80: Demilited message body prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: Premature end of Content-Length delimited message body (expected: 1503; received: 0)

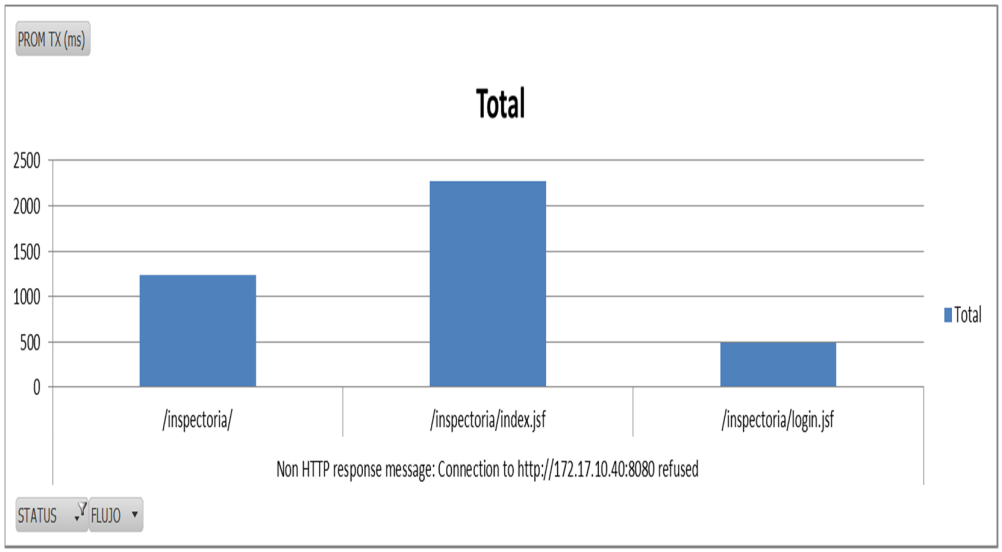
Figura 81: Demilited message body prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: Connection to http://172.17.10.40:8080 refused

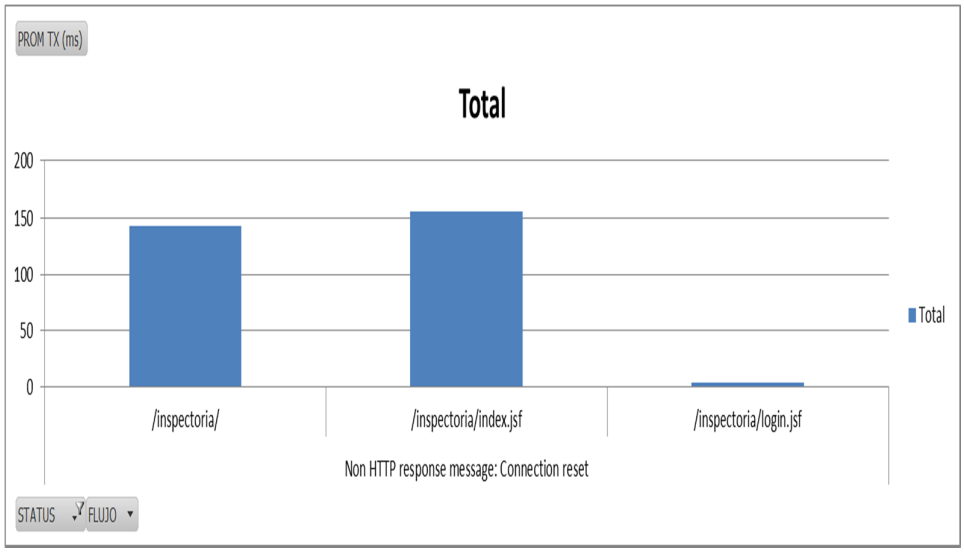
Figura 82: Connection refuse prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Non HTTP response message: Connection reset

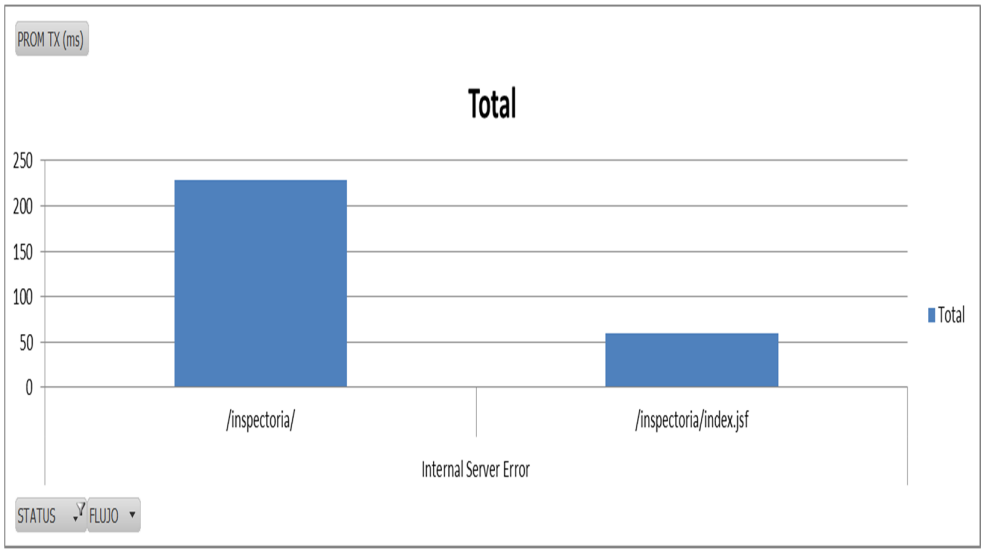
Figura 83: Connection reset prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Internal Server Error

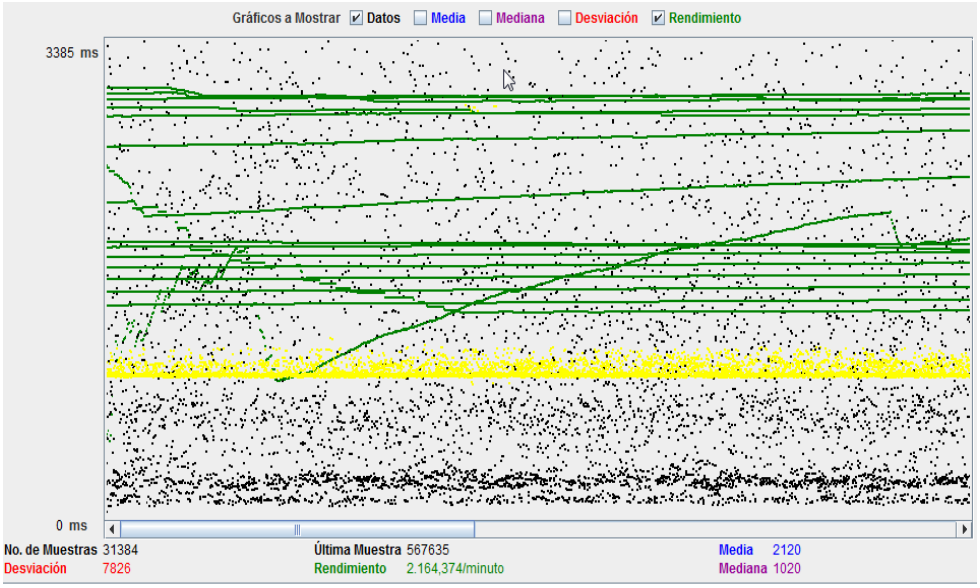
Figura 84: Internal Server Error prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Distribución de datos

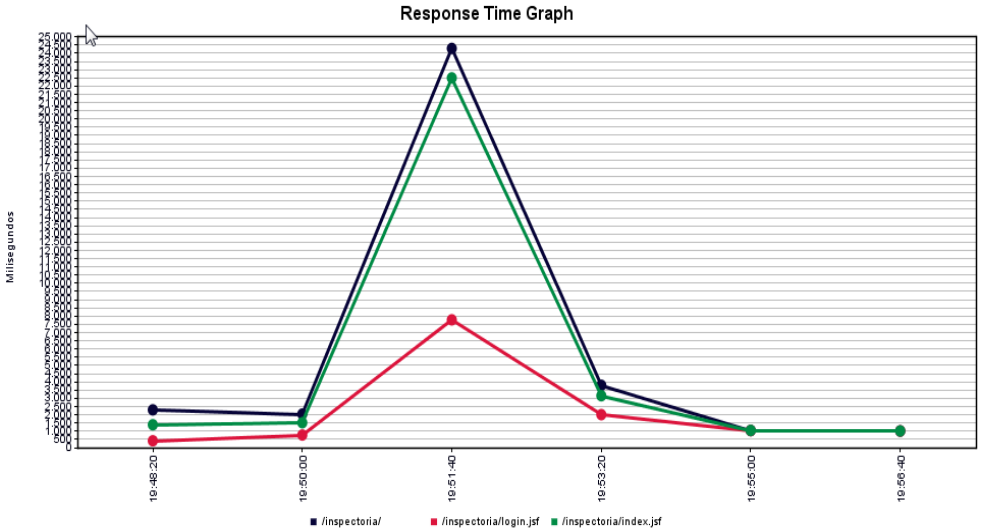
Figura 85: Distribución gráfica prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Tiempo de respuesta

Figura 86: Tiempo respuesta prueba stress 3



Elaborado por: Byron Andrango y Diego Jácome

Resultado de las pruebas de stress 3

Se realizó con los siguientes parámetros:

- 150 usuarios.
- 30 repeticiones.
- 4500 solicitudes de login.

El procesamiento de la solicitud duró 05:02 minutos, el objetivo de la prueba fue ver la capacidad y rendimiento de la aplicación en horas picos teniendo como resultado la caída del servidor de aplicaciones.

3.3.2. Pruebas de Caja Negra

Tabla 29: Caso de Prueba 1

CASO DE PRUEBA 1:	Comprobar que el sistema no permita el ingreso de un usuario no registrado.
DATOS DE ENTRADA:	Usuario: usuario Inventado
	Clave: clave Inventada
	Datos del registro
CONDICIONES DE EJECUCIÓN:	En la tabla "sis_usuarios" el registro del usuario ingresado no debe de estar en base de datos.
RESULTADO ESPERADO:	El sistema no permite el acceso al usuario no registrado.
SALIDA:	Sesión inválida, los datos ingresados son incorrectos.

Elaborado por: Byron Andrango y Diego Jácome

Tabla 30: Caso de Prueba 2

CASO DE PRUEBA 2:	Comprobar el ingreso de un nuevo registro en el sistema
DATOS DE ENTRADA:	Usuario: admin
	Clave: ups123
	Datos del registro
CONDICIONES DE EJECUCIÓN:	El registro a ingresar no debe existir en la base de datos.
RESULTADO ESPERADO:	El sistema envía a guardar el registro a la base de datos.
SALIDA:	Registro ingresado exitosamente.

Elaborado por: Byron Andrango y Diego Jácome

Tabla 31: Caso de Prueba 3

CASO DE PRUEBA 3:	Comprobar la actualización del registro en el sistema
DATOS DE ENTRADA:	Usuario: admin
	Clave: ups123
CONDICIONES DE EJECUCIÓN:	El registro a actualizar debe existir en la base de datos.
RESULTADO ESPERADO:	El sistema envía a actualizar el registro a la base de datos.
SALIDA:	Registro actualizado exitosamente.

Elaborado por: Byron Andrango y Diego Jácome

Tabla 32: Caso de Prueba 4

CASO DE PRUEBA 4:	Comprobar la eliminación del registro en el sistema
DATOS DE ENTRADA:	Usuario: admin
	Clave: ups123
	Registro a eliminar
CONDICIONES DE EJECUCIÓN:	El registro a actualizar debe existir en la base de datos.
RESULTADO ESPERADO:	El sistema envía a actualizar el registro a la base de datos.
SALIDA:	Registro actualizado exitosamente.

Elaborado por: Byron Andrango y Diego Jácome

Tabla 33: Caso de Prueba 5

CASO DE PRUEBA 5:	Consultar un registro en el sistema
DATOS DE ENTRADA:	Usuario: admin
	Clave: ups123
	Parámetros de búsqueda
CONDICIONES DE EJECUCIÓN:	Deben existir registros en la base de datos.
RESULTADO ESPERADO:	El sistema presenta los registros recuperados de la base de datos con los parámetros de búsqueda ingresados.
SALIDA:	Listado de registros encontrados.

Elaborado por: Byron Andrango y Diego Jácome

Tabla 34: Caso de Prueba 6

CASO DE PRUEBA 6:	Insertar detalles de un registro en el sistema
DATOS DE ENTRADA:	Usuario: admin
	Clave: ups123
	Datos del registro
CONDICIONES DE EJECUCIÓN:	Debe existir el registro en la base de datos.
RESULTADO ESPERADO:	El registro debe tener un detalle asignado.
SALIDA:	Detalle ingresado exitosamente.

Elaborado por: Byron Andrango y Diego Jácome

Tabla 35: Caso de Prueba 7

CASO DE PRUEBA 7:	Generar formulario
DATOS DE ENTRADA:	Usuario: admin
	Clave: ups123
	Parámetros de generación de formulario
CONDICIONES DE EJECUCIÓN:	Debe existir el registro en la base de datos.
RESULTADO ESPERADO:	Archivo xml generado por el sistema.
SALIDA:	Archivo xml generado exitosamente.

Elaborado por: Byron Andrango y Diego Jácome

Tabla 36: Caso de Prueba 8

CASO DE PRUEBA 8:	Generar Anexo
DATOS DE ENTRADA:	Usuario: admin
	Clave: ups123
	Parámetros de generación de anexo
CONDICIONES DE EJECUCIÓN:	Debe existir el registro en la base de datos.
RESULTADO ESPERADO:	Archivo xml generado por el sistema.
SALIDA:	Archivo xml generado exitosamente.

Elaborado por: Byron Andrango y Diego Jácome

3.3.3. Pruebas de Validación

Tabla 37: Validación Empresas

EMPRESAS				
Tipo componente	**Campo**	**Patrón**	**Ejemplo**	**Mensaje del sistema**
Texto	Nombre empresa	Requerido	Campo vacío	Datos requeridos

Elaborado por: Byron Andrango y Diego Jácome

Tabla 38: Validación Opción

OPCIONES				
Tipo componente	**Campo**	**Patrón**	**Ejemplo**	**Mensaje del sistema**
Texto	Nombre opción	Requerido	Campo vacío	Datos requeridos
Texto	Número tabla	Numeric/requerido	Asadd/Campo vacío	No digita letras/datos requeridos
Texto	Tabla	Requerido	Campo vacío	Datos requeridos

Elaborado por: Byron Andrango y Diego Jácome

Tabla 39: Validación Permiso

PERMISOS				
Tipo componente	**Campo**	**Patrón**	**Ejemplo**	**Mensaje del sistema**
Texto	Nombre perfil	Requerido	Campo vacío	Datos requeridos

Elaborado por: Byron Andrango y Diego Jácome

Tabla 40: Validación Sucursal

SUCURSAL				
Tipo componente	**Campo**	**Patrón**	**Ejemplo**	**Mensaje del sistema**
Texto	Nombre sucursal	Requerido	Campo vacío	Datos requeridos

Elaborado por: Byron Andrango y Diego Jácome

Tabla 41: Validación Usuario

USUARIO				
Tipo componente	**Campo**	**Patrón**	**Ejemplo**	**Mensaje del sistema**
Texto	Nombre usuario	Requerido	Campo vacío	Datos requeridos
Texto	Nickname	Requerido	Campo vacío	Datos requeridos
Texto	Clave usuario	Requerido	Campo vacío	Datos requeridos

Elaborado por: Byron Andrango y Diego Jácome

Tabla 42: Validación Impuesto Renta

IMPUESTO RENTA				
Tipo componente	**Campo**	**Patrón**	**Ejemplo**	**Mensaje del sistema**
Texto	Nombre impuesto renta	Requerido	Campo vacío	Datos requeridos

Elaborado por: Byron Andrango y Diego Jácome

Tabla 43: Validación Tipo Sustento Tributario

TIPO SUSTENTO TRIBUTARIO				
Tipo componente	**Campo**	**Patrón**	**Ejemplo**	**Mensaje del sistema**
Texto	Nombre tipo sustento tirbutario	Requerido	Campo vacío	Datos requeridos

Elaborado por: Byron Andrango y Diego Jácome

CAPÍTULO 4
IMPLEMENTACIÓN

4.1. Instalación del Sistema

Para la instalación del sistema se necesita el siguiente software:

- Instalar JDK 1.7
- Instalar Servidor de aplicaciones GlassFish 3.1.
- Instalar Postgres 9.1

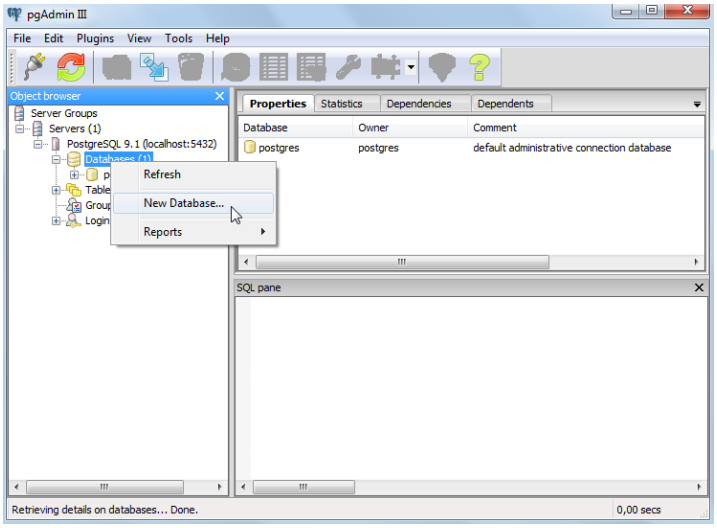
Después de haber instalado el software se necesita:

- Iniciar el servidor de aplicaciones Glassfish 3.1
    - **path-install/bin/asadmin start-domain**
- Inicia el gestor de base de datos postgres.

4.1.1. Restaurar backup de la base de datos

Se inicia sesión postgres y se crea la base de datos del sistema

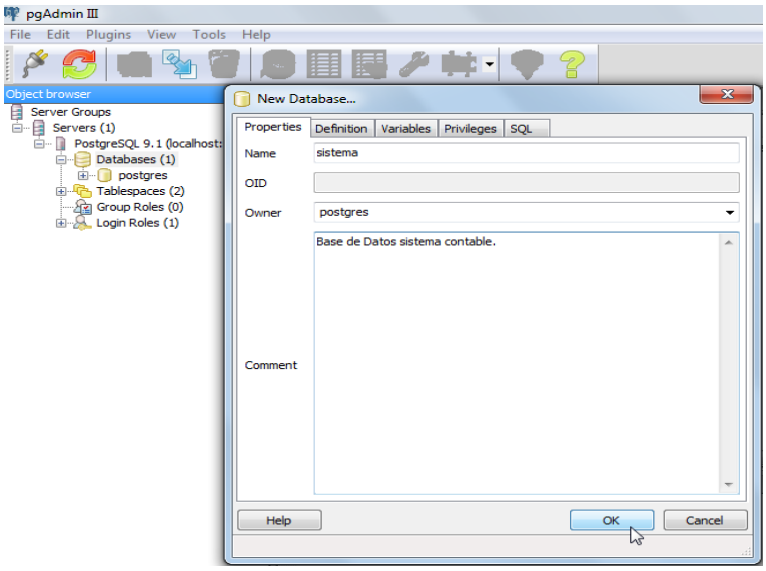
Figura 87: Crear Base de Datos



Elaborado por: Byron Andrango y Diego Jácome

Se nombre la nueva base de datos como “sistema”.

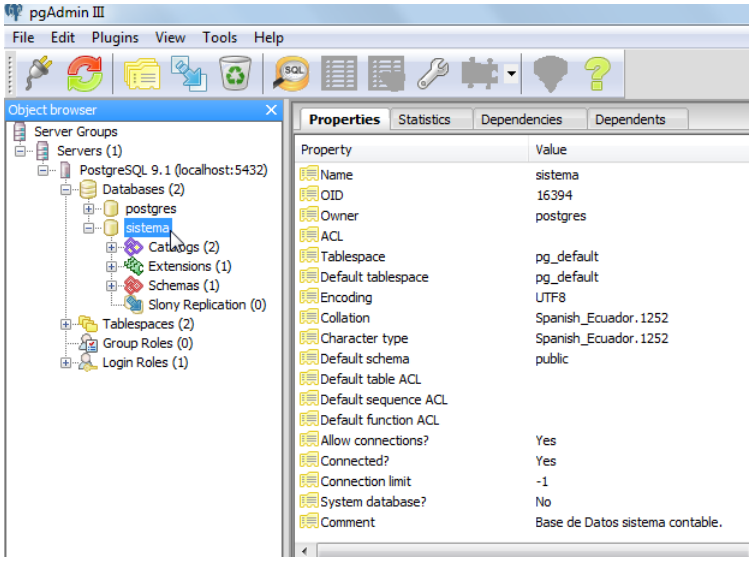
Figura 88: Ingresar Nombre Base de Datos



Elaborado por: Byron Andrango y Diego Jácome

Se abre la base de datos creada "sistema"

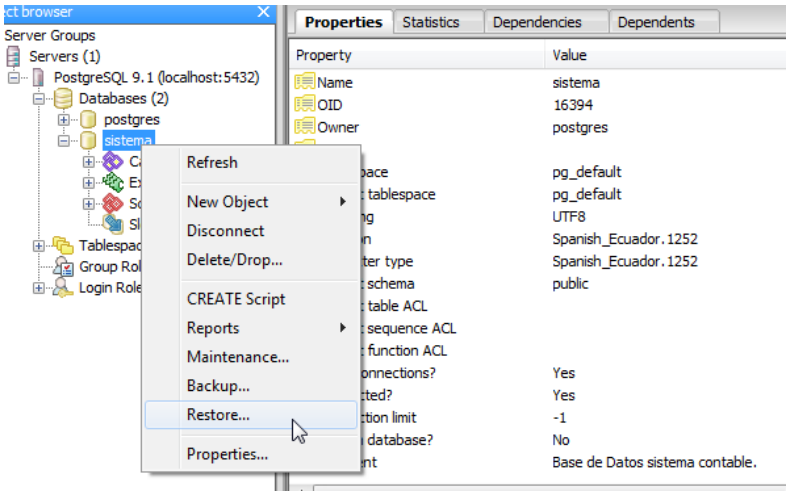
Figura 89: Abrir Base de Datos



Elaborado por: Byron Andrango y Diego Jácome

Se restaura el backup en la base de datos creada "sistema" del modelo de la base de datos

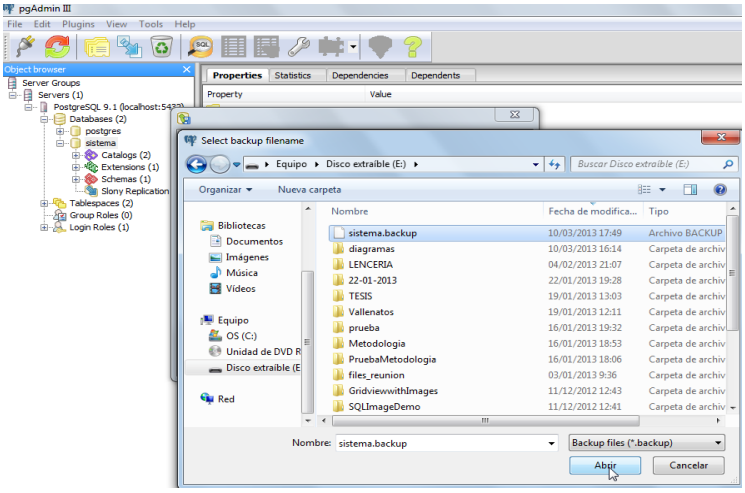
Figura 90: Restaurar Base de Datos



Elaborado por: Byron Andrango y Diego Jácome

Se selecciona el archivo de backup de la base de datos en el directorio ubicado

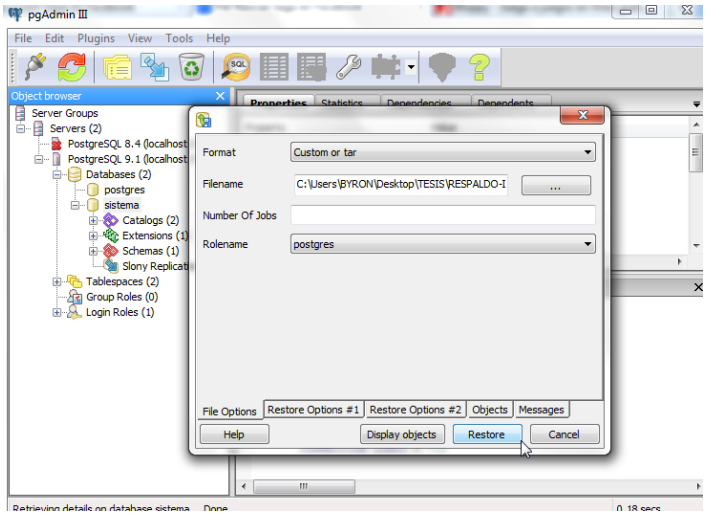
Figura 91: Selecciona el archivo a restaurar



Elaborado por: Byron Andrango y Diego Jácome

Se presiona Restore

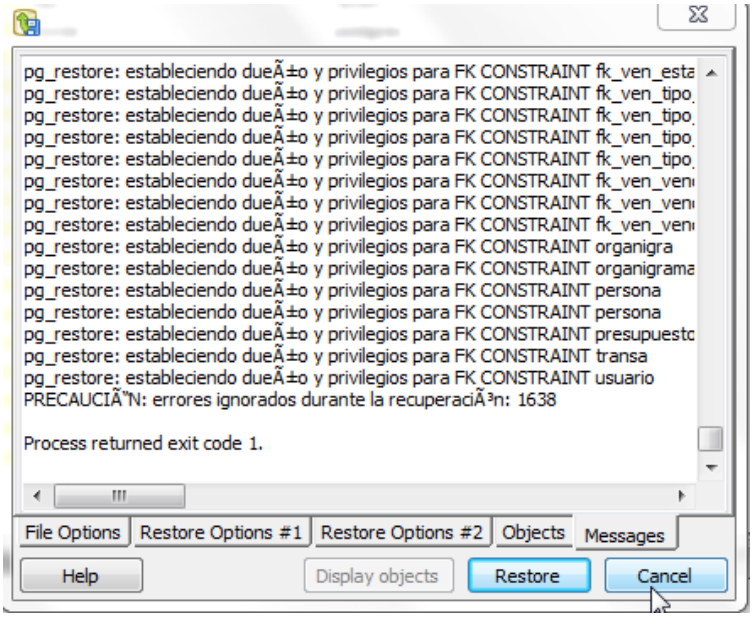
Figura 92: Restaurar Base de Datos



Elaborado por: Byron Andrango y Diego Jácome

El pgAdmin comienza el proceso de restauración del backup

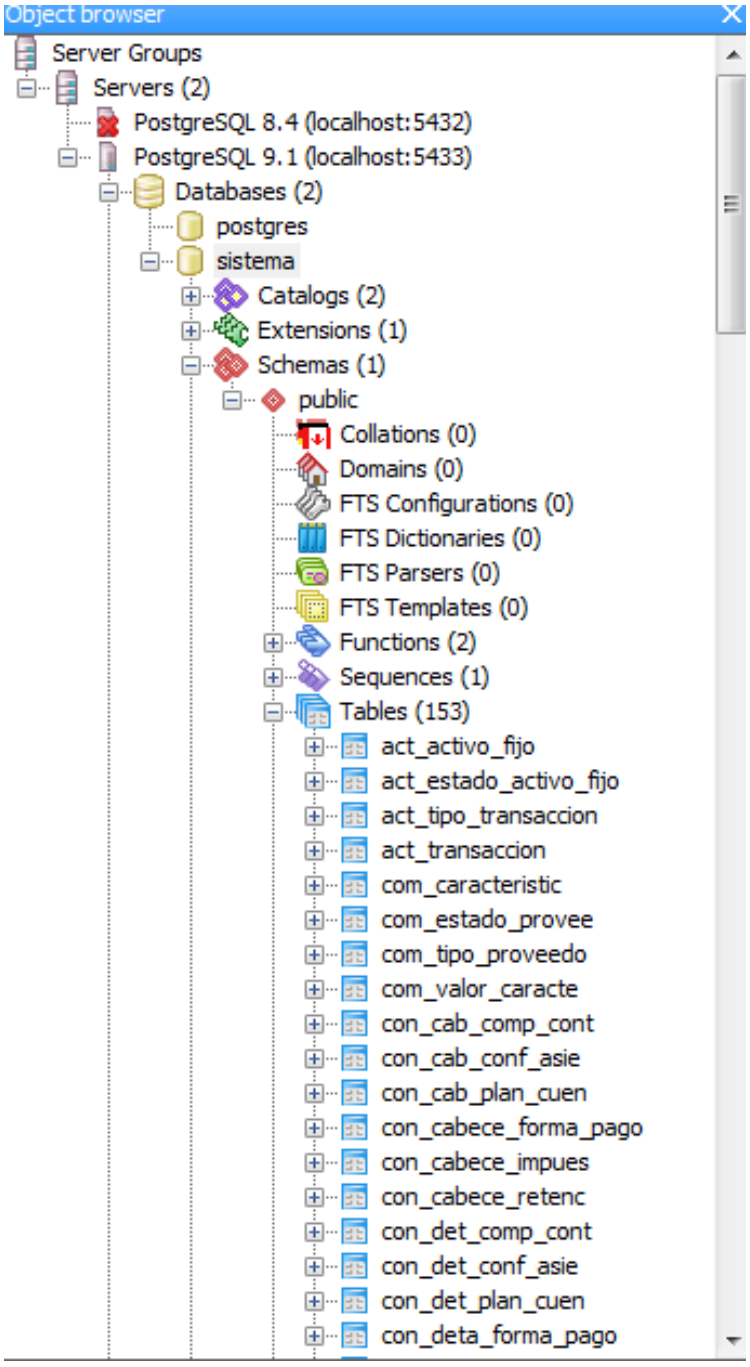
Figura 93: Proceso de Restauración



Elaborado por: Byron Andrango y Diego Jácome

Terminado el proceso de restauración se puede observar las tablas en la base de datos "sistema"

Figura 94: Base de Datos Restaurada

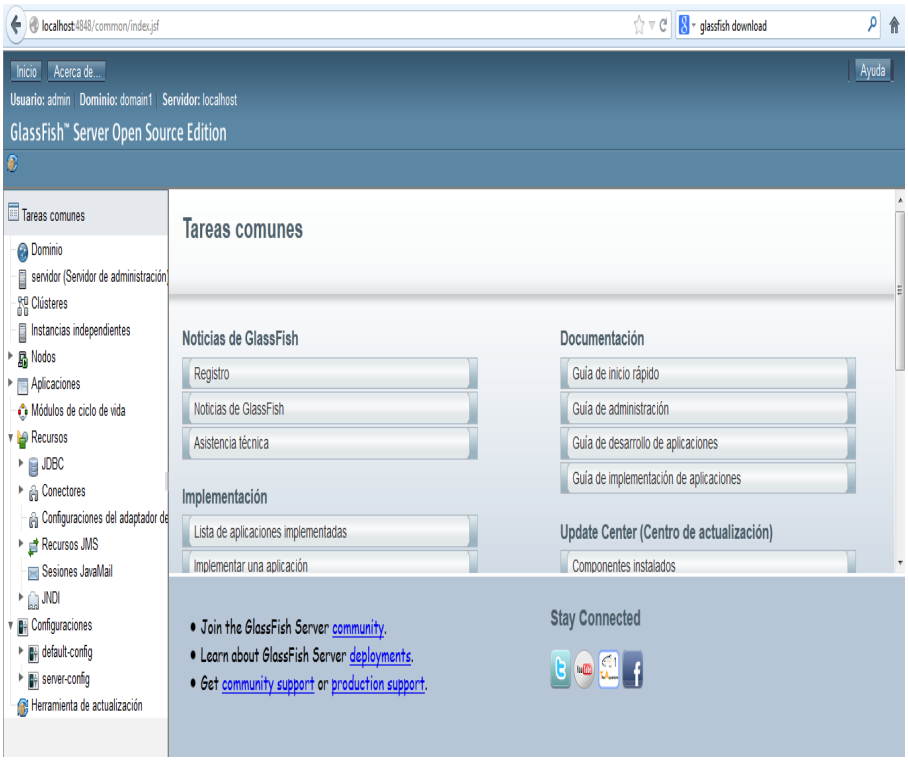


Elaborado por: Byron Andrango y Diego Jácome

4.1.2. Configuración pool de conexiones en Glassfish

Se inicia la consola de administración de Glassfish

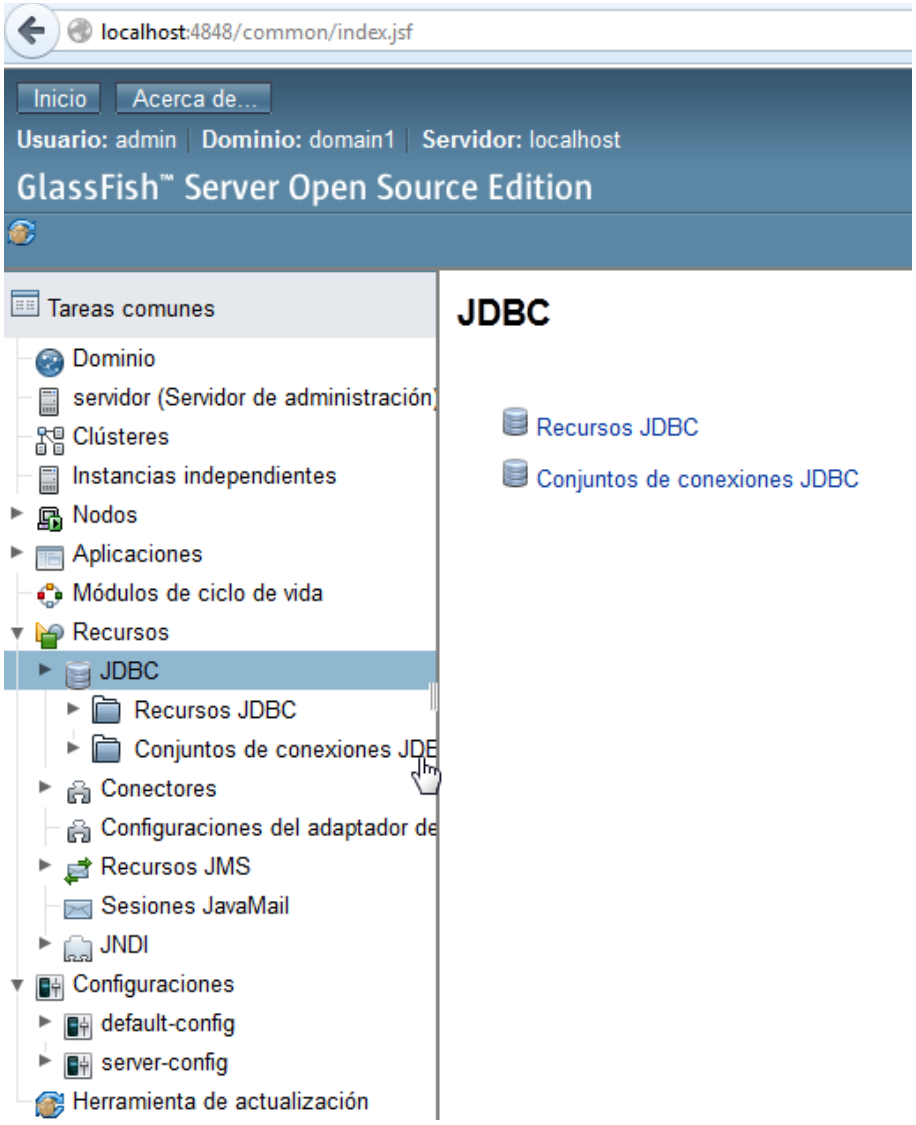
Figura 95: Consola de administración GlassFish



Elaborado por: Byron Andrango y Diego Jácome

Se selecciona Recursos/JDBC/Conjunto de conexiones JBDC/Nuevo

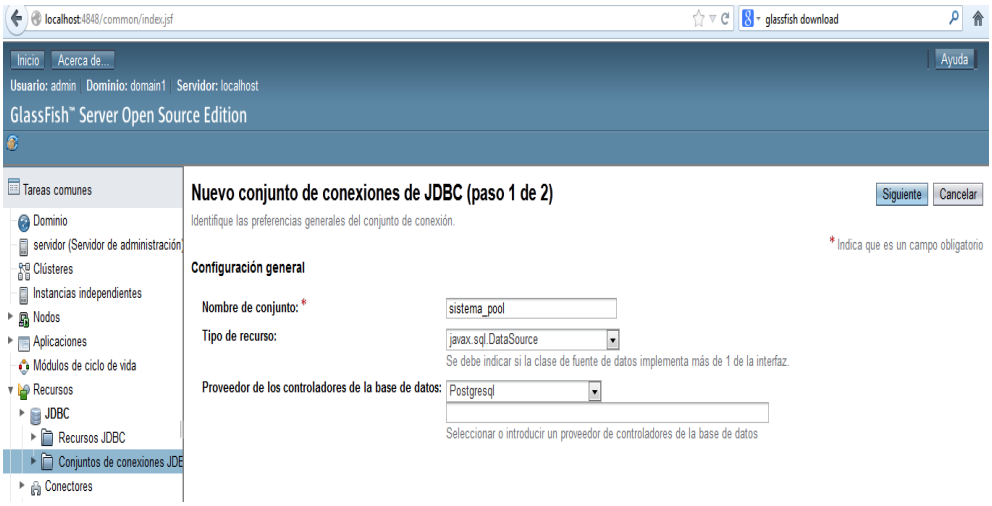
Figura 96: Configuración recursos JDBC



Elaborado por: Byron Andrango y Diego Jácome

Se ingresa el nombre, tipo de recurso y el proveedor de base de datos del nuevo pool de conexiones.

Figura 97: Creación del pool de conexiones



Elaborado por: Byron Andrango y Diego Jácome

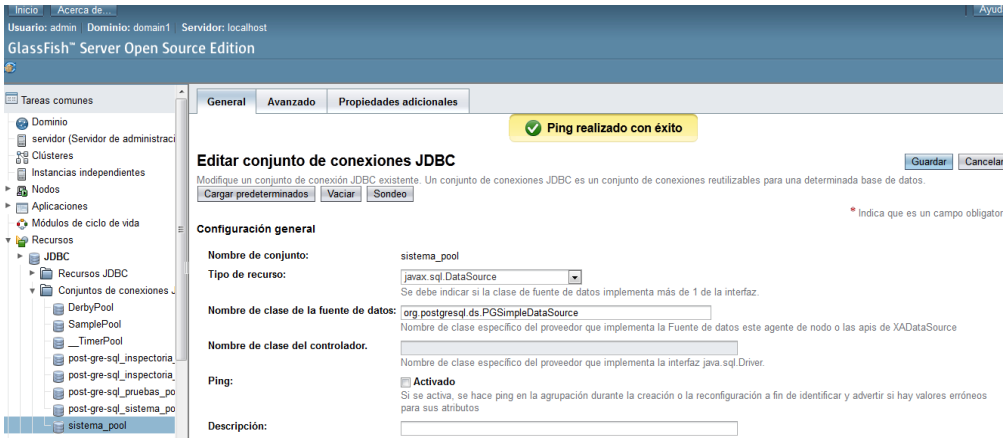
Se presiona siguiente y se ingresa los valores para establecer la conexión con la base de datos "sistema" con el servidor Postgres.

Figura 98: Parámetros para la conexión de la base de datos

Elaborado por: Byron Andrango y Diego Jácome

Se presiona finalizar y se verifica el estado de la conexión con la base de datos "sistema", presionando el botón Sondeo.

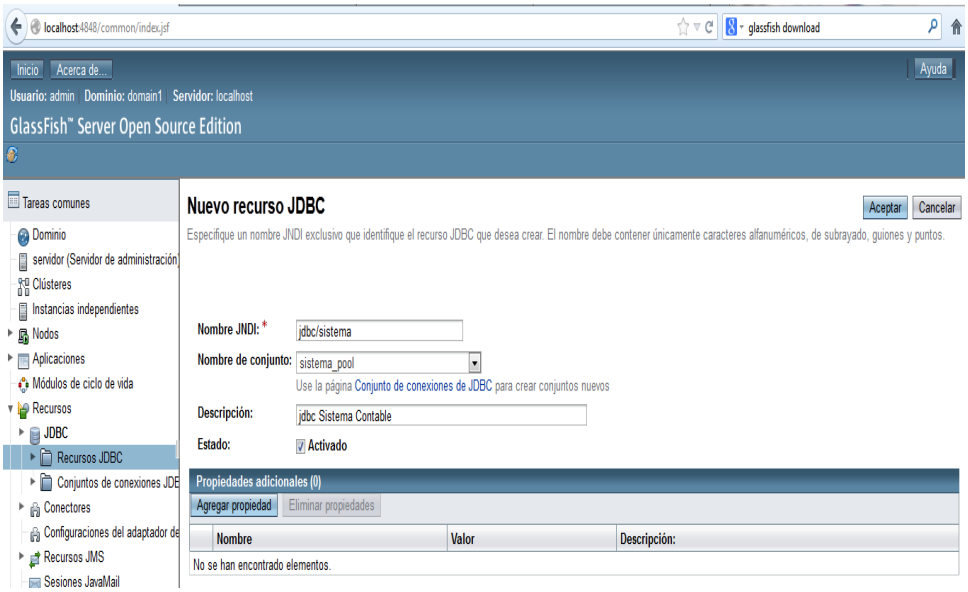
Figura 99: Probar de conectividad con el pool de conexiones definido



Elaborado por: Byron Andrango y Diego Jácome

Se selecciona Recursos/JDBC/Recursos JDBC/Nuevo y se le asigna pool de conexiones creado anteriormente.

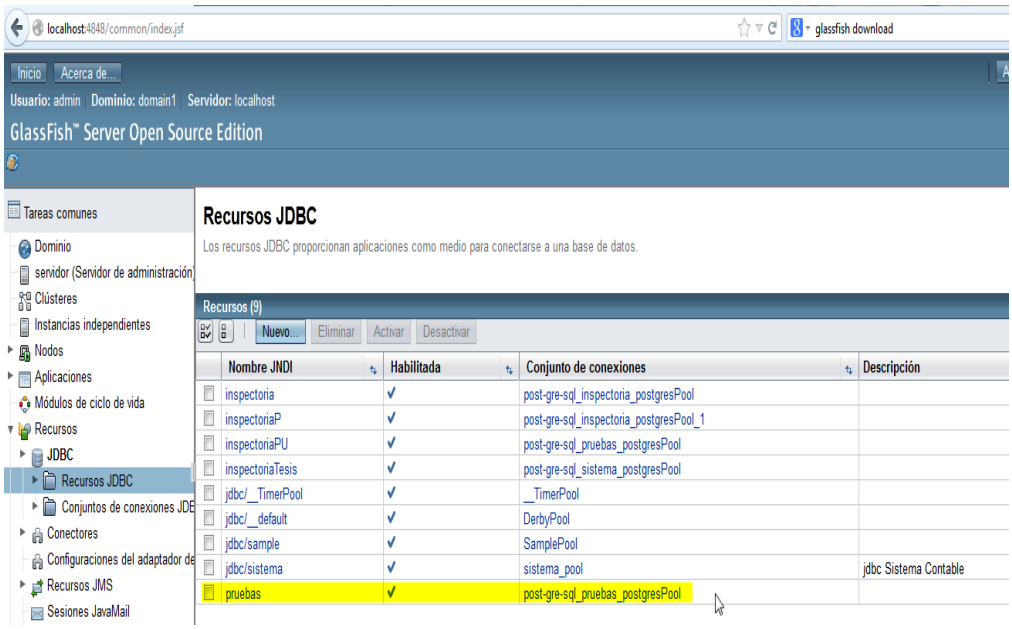
Figura 100: Creación nuevo recurso JDBC



Elaborado por: Byron Andrango y Diego Jácome

Se presiona Aceptar y se tiene configurado el Recurso JDBC de la nueva conexión para la base de datos “sistema”.

Figura 101: Visualización del recurso JDBC creado



Elaborado por: Byron Andrango y Diego Jácome

CONCLUSIONES

- El framework es ideal para realizar prototipos de aplicaciones web escritas en Java, las mismas que pueden ser adaptadas e integradas con el framework JSF o utilizar tecnologías de soporte en diferentes capas de la plataforma JEE como Hibernate o JPA.
- Para poder usar el framework en el desarrollo de aplicaciones web se necesita tener conocimientos básicos en programación java y sentencias sql.
- El framework reduce el tiempo de desarrollo de las aplicaciones web, ya que abstrae la implementación de tecnologías importantes para la construcción de aplicaciones webs dinámicas de forma transparente para el desarrollador.
- Con el uso del framework en el desarrollo de aplicaciones web se tiene la opción que un programador inexperto desarrolle sus aplicaciones sin tener conocimientos previos de tecnologías embebidas como JavaScript, Ajax y sin tener una noción básica de programación HTML.
- El framework JSF maneja contenedores para los controladores, por lo cual no tiene la necesidad de realizar solicitudes HTTP Request al servidor para poder continuar la secuencia de un flujo; de tal manera las solicitudes son transparentes para un HTTP Proxy brindando seguridad extra a ataques de fuerza bruta.
- El framework realizado sirvió para que el desarrollo del Sistema Financiero Contable sea confiable, escalable y sea de fácil mantenimiento; además permite al desarrollador la reutilización de código ahorrando tiempo y esfuerzo.
- La generación de los archivos xml para los formularios y anexos S. R. I pasaron exitosamente la validación del programa DIMM; dando la facilidad al usuario contable de generar la información lista para su envío, ahorrando tiempo y esfuerzo.
- Se analizó la posibilidad de realizar pruebas de carga sobre el nuevo Sistema Contable Financiero, pero al carecer la Casa Inspectorial de un sistema web similar no se tenía como identificar los cuellos de botella en el proceso contable.

- Para la integración del Sistema Contable Financiero se utiliza el Módulo Sistema en el cual el usuario administrador es el encargado de crear las opciones del menú, armar perfiles asignando permisos de acuerdo a la funcionalidad y rol del usuario contable en la empresa.
- Con las pruebas de stress realizadas en el Sistema Contable Financiero se pudo medir el rendimiento y la rapidez en procesar el ingreso a la aplicación en las horas picos y con números de usuarios superiores a los de la Casa Inspectorial Salesiana.

RECOMENDACIONES

- Se recomienda que el personal de la Casa Inspectorial Salesiana tenga capacitaciones en el Sistema Contable Financiero para que puedan utilizar la aplicación de forma correcta y rápida en el día a día en sus actividades laborales.
- Construir las aplicaciones empresariales utilizando un proyecto web, que consuma recursos del servidor y que se dedique a la navegación y lógica de las páginas y no a la persistencia de datos o lógica de negocio.
- En el caso que el sistema falle (en casos excepcionales), las actividades de la Casa Inspectorial se paralizarían. Debido a este problema se debe implementar una política de seguridad en la que se contemple la obtención de una copia diaria de la base de datos y pueda pasar a un servidor temporal.
- Después de haber generado el formulario o anexo del S. R. I se debe validar el archivo.xml en el programa DIMM para comprobar si el archivo es correcto.
- Si se desea desarrollar una aplicación web ágil, con resultados en pantalla en muy poco tiempo y sin complicaciones se debe usar el framework.

LISTA DE REFERENCIAS

- Blanco, C. (2012). Recuperado el 20 de diciembre del 2012 de http://carlosblanco.pro/2012/04/entornos-desarrollo-integrado-introduccion/
- del Rocío, P. (2010). Fundamentos de Investigacion: TESIS DE GRADO EN INGENIERÍA EN INFORMÁTICA. Recuperado el 20 de diciembre del 2012 de blogspot Web site: http://perla-del-rocio.blogspot.com/2011/11/tesis-de-grado-en-ingenieria-en.html
- JasperSoft. (2011). Recuperado el 01-2013 de http://www.jaspersoft.com/
- Java, N. (2012). Recuperado el 10 de enero del 2012 de http://netjava.bligoo.com/netbeans
- Navicat. (2012). Recuperado el 20 de diciembre del 2012 de http://www.navicat.com/products/navicat-premium
- ORACLE. (2012). Recuperado el 20 de diciembre del 2012 de http://www.oracle.com/us/sun/index.html
- SpringSource. (2003). Recuperado el 10 de enero del 2012 de http://static.springsource.org/spring/docs/3.2.x/spring-framework-reference/pdf/spring-framework-reference.pdf
- SRI. (Diciembre de 2012). Recuperado el 20 de diciembre del 2012 de http://www.sri.gob.ec/web/10138/169
- SRI. (2012). Recuperado el 10 de enero del 2012 de http://www.sri.gob.ec//DocumentosAlfrescoPortlet/descargar/3afe3708-f0af-4427-8946-52780536106c/FORMULARIO+104.pdf
- SRI. (2012). Recuperado el 20 de diciembre del 2012 de http://www.sri.gob.ec/web/guest/174
- SRI. (2012). Recuperado el 20 de diciembre del 2012 de http://www.sri.gob.ec/web/guest/anexo-transaccional-simplificado-ats
- SRI. (2012). Recuperado el 20 de diciembre del 2012 de http://www.sri.gob.ec/web/guest/173
- V3RGU1. (2012). Recuperado el 10 de enero del 2012 de http://www.v3rgu1.com/blog/476/2011/programacion/java-y-los-ficheros-properties/

GLOSARIO

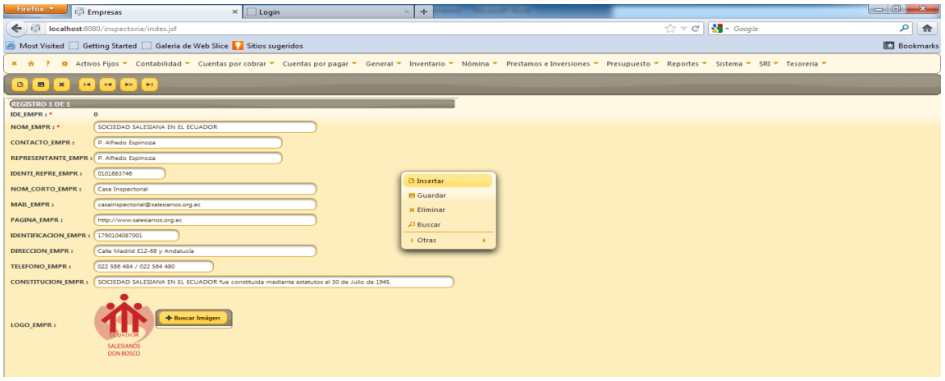
- **Clustering:** Se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de hardwares comunes y que se comportan como si fuesen una única computadora.
- **Dynaform:** Hace posible la construcción de una forma dinámica, con etiquetas, entradas, selecciona y cualesquiera otros elementos de modelo.
- **EJB CMP:** Es un bean de entidad cuyo estado se encuentra sincronizada con la base de datos de forma automática.
- **Embebida:** Es un sistema de computación diseñado para realizar una o algunas pocas funciones dedicadas frecuentemente en un sistema de computación en tiempo real.
- **Herencia:** La herencia es permitir la creación de nuevas clases basadas en clases existentes.
- **in-house:** En programación es todo desarrollo de software dentro de la empresa
- **JDO:** Es una especificación de Java objeto persistencia.
- **JEE:** Java Platform, Enterprise Edition. (Anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE).
- **Online:** Es proveer información en tiempo real a través de internet o una intranet.
- **Offline:** Se encentra fuera de línea o sin acceso a internet o una intranet.
- **Releases:** Es la acción de lanzar a ejecutar un proceso o transacción.
- **SDK:** Es un conjunto de herramientas de desarrollo de software que le permite al programador crear aplicaciones para un sistema concreto
- **Sincronización:** Es hacer que coincidan en el tiempo dos o más procesos.
- **Técnica Top-Down:** Estrategia de procesamiento de información característica de las ciencias de la información, especialmente en lo relativo al software.

ANEXO

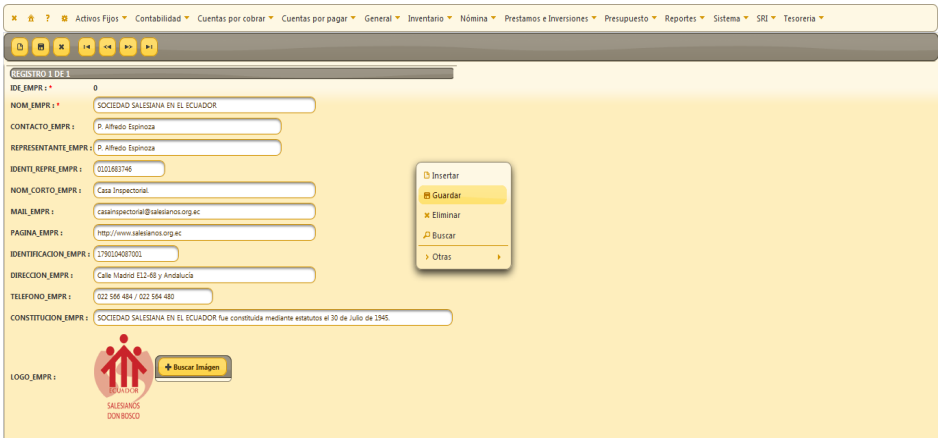
Manual de usuario módulos sistema y S.R.I

Opción empresa

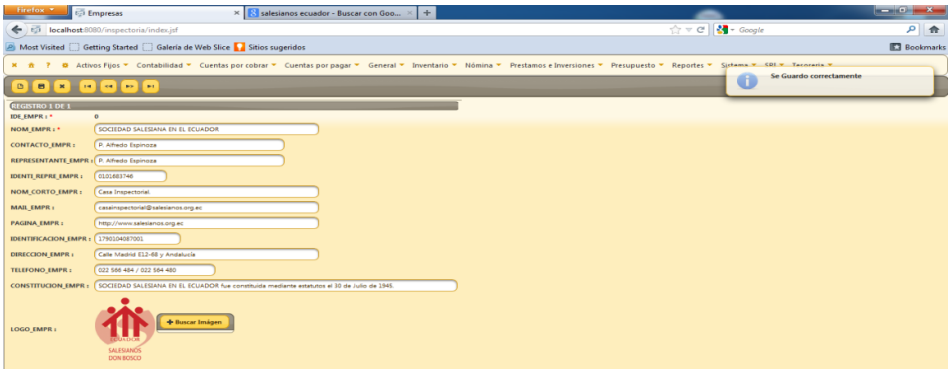
Para ingresar una nueva Empresa dar click derecho sobre la pantalla y seleccionar insertar



Ingresar los datos en el formulario y dar click en guardar



Se presentará mensaje de guardado

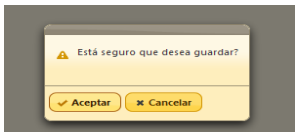


Opción Sucursal

Para ingresar nueva sucursal dar click en Insertar

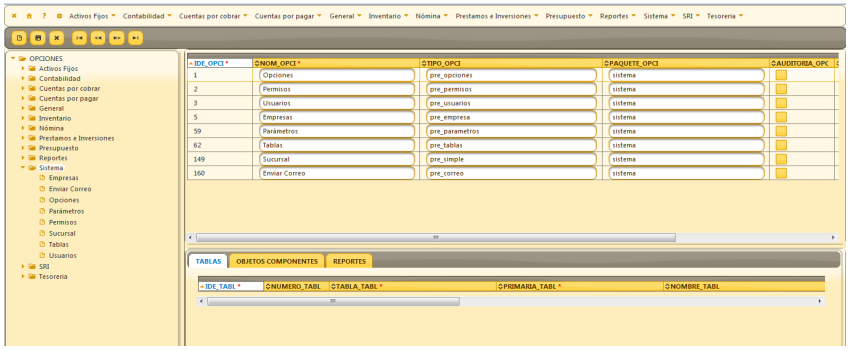
Ingresar los datos en el formulario

Dar click en Guardar y Aceptar

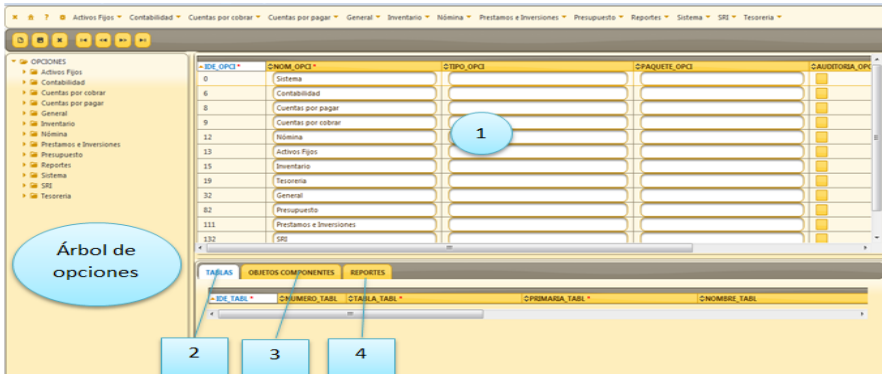


Opciones

Para ingresar una nueva opción se da click en insertar



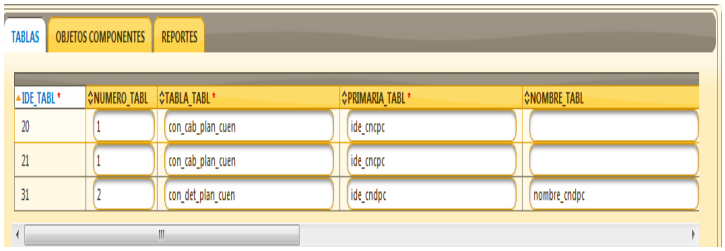
Para ingresar datos se debe de tomar en cuenta la siguiente explicación del funcionamiento de la pantalla:



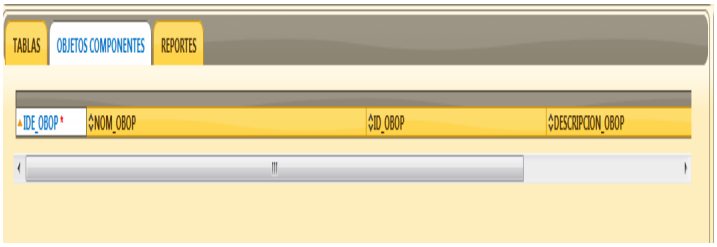
En esta opción se pueden crear las opciones para el menú de la barra principal, y sub menús que se requieran al igual que configurar las pantallas, los reportes y los componentes según su necesidad.

Árbol de Opciones: Se muestra todas las opciones que se encuentran configuradas de una forma dinámica y visual para su visualización.

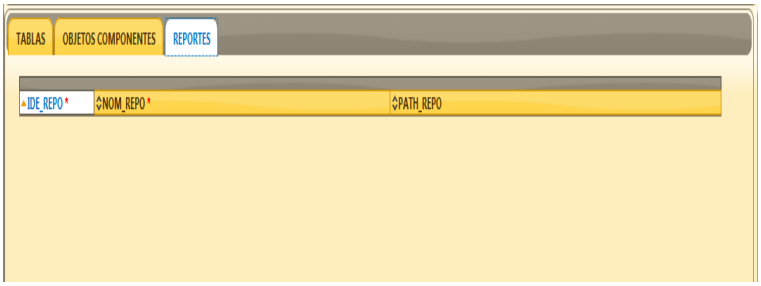
1. **Opciones:** Se muestra una tabla para ingresar y configurar la pantalla llenado los campos de nom_opci, tipo_opci y paquete_opci que se requiera mostrar según su configuración deseada.
2. **Tablas:** Se muestra la configuración de la pantalla configurada anteriormente, aquí se detalla las tablas que posee como es el ide, el nombre de la tabla, y el número de tabla.



3. **Objetos Componentes:** Se muestra la opción para configurar los componentes que se necesiten que sean administrados

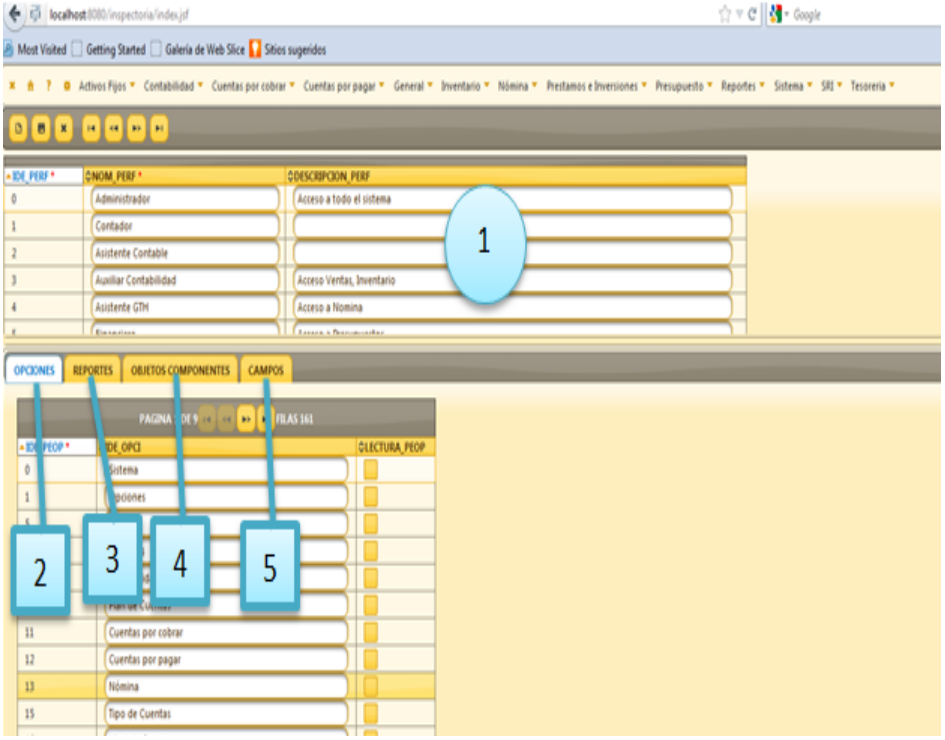


4. **Reportes:** Se muestra la opción para configurar los reportes que se necesitan que sean requeridos en las pantallas que hayan sido configuradas anteriormente realizadas



Opción Permisos

Dar click en la opción Permisos

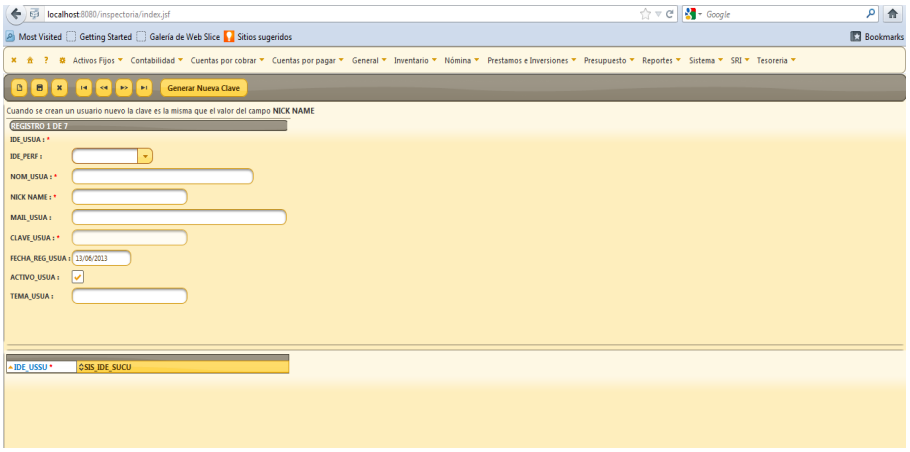


A continuación se detalla cada una de las propiedades del formulario

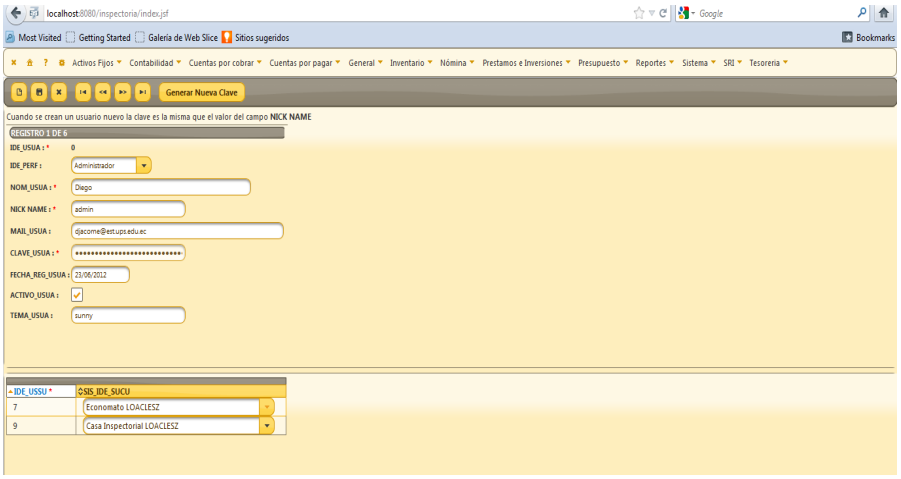
1. **Perfil:** Permite crear un perfil y una descripción
2. **Opciones:** De acuerdo al perfil creado se procede a crear el permiso a la opción que es la pantalla creada anteriormente la cual podrá visualizar al momento que ingrese con su usuario.
3. **Reportes:** Permite dar el permiso para que pueda visualizar los reportes que se le han asignado al usuario
4. **Objetos Componentes:** Se crea de acuerdo a la necesidad de uso del componente se da el permiso necesario poniendo el nombre del componente.
5. **Campos:** Se crea permiso para que pueda visualizar el o los campos necesarios que requiera el usuario.

Opción Usuario

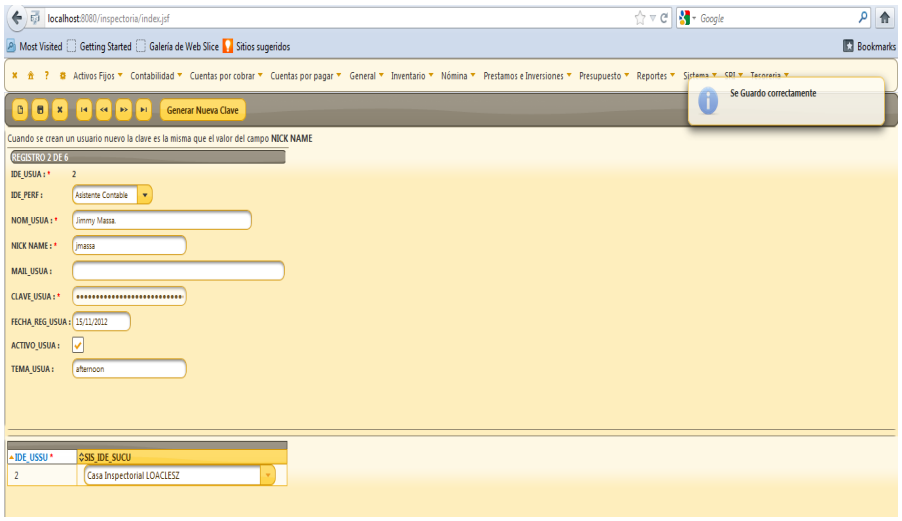
Para ingresar un nuevo usuario se da click en Insertar.



Se llenan los datos en el formulario y se le asigna la sucursal al usuario.

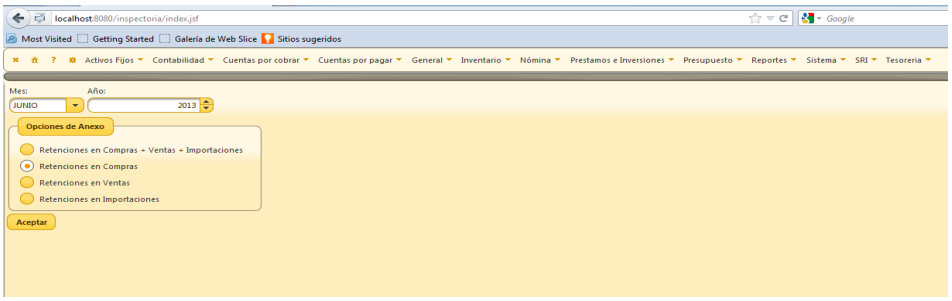


Dar click en guardar y aparecerá un mensaje de guardado exitosamente.

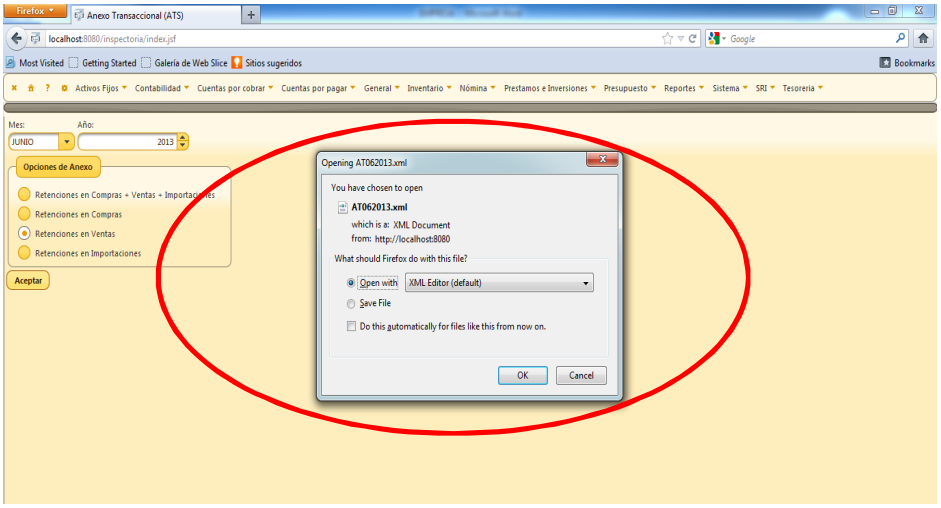


Generación de Anexos del S.R.I

Para generar los anexos se escoge el mes y el año en el que se requiera generar

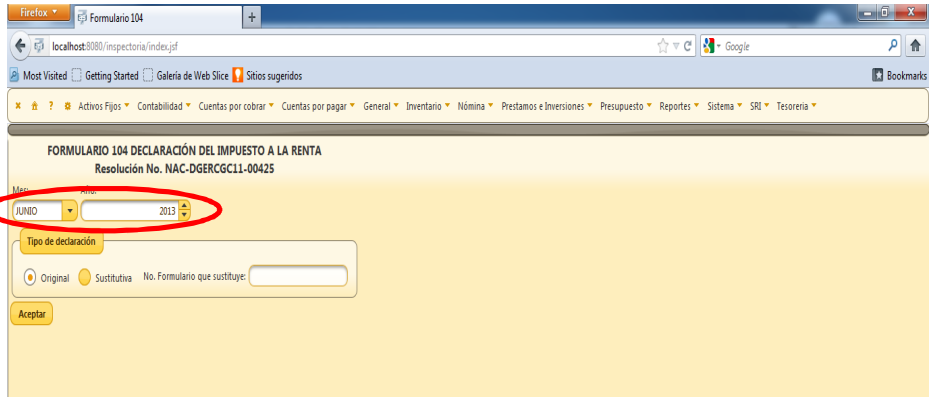


Dar click en Aceptar para generar el archivo .xml,



Generación de formularios del S.R.I

Para generar los formularios se debe escoger el mes, año en el que se requiera generar y el tipo de declaración



Dar click en Aceptar para generar el archivo .xml,