**ESCUELA POLITÉCNICA DEL EJÉRCITO**

**DPTO. DE CIENCIAS DE LA COMPUTACIÓN**

# CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA

## "EVALUACIÓN DE TRES IMPLEMENTACIONES JAVA SERVER FACES 1.2 PARA SU APLICACIÓN EN EL DESARROLLO DEL MÓDULO DE SEGURIDADES DEL SISTEMA GESTOR FIDUCIA FONDOS EN GESTORINC S.A."

**Previa a la obtención del Título de:**

## INGENIERA EN SISTEMAS E INFORMÁTICA

**POR: GORDILLO CHASIPANTA LUCÍA CATALINA**

**SANGOLQUÍ, 27 DE ABRIL DEL 2010**

# CERTIFICACIÓN

Certifico que el presente trabajo fue realizado en su totalidad por la Srta. GORDILLO CHASIPANTA LUCÍA CATALINA como requerimiento parcial a la obtención del título de INGENIERA EN SISTEMAS E INFORMÁTICA.

Sangolquí, Abril del 2010

_____

ING. HENRY CORAL C.

# DEDICATORIA

A mis padres por creer en mí y apoyarme en cada decisión a lo largo de mi vida y a Dios por permitirme culminar con éxito una etapa más.

**Lucía Catalina Gordillo Chasipanta**

# AGRADECIMIENTO

A todos mis profesores que han dejado su legado en mí, a mis compañeros de aula con quiénes compartí diariamente y me permitieron aprender de ellos, y a todas las personas que de una u otra manera han colaborado para la elaboración de éste trabajo.

**Lucía Catalina Gordillo Chasipanta**

# ÍNDICE DE CONTENIDOS

# LISTADO DE TABLAS

# LISTADO DE FIGURAS

# LISTADO DE ANEXOS

# RESUMEN

El presente proyecto se encuentra encaminado a la evaluación de tres implementaciones de la Tecnología JAVA SERVER FACES 1.2 con soporte para AJAX, en su correspondiente aplicación a la capa de presentación del Sistema Gestor Fiducia Fondos de la Empresa GESTORINC S.A.

La evaluación de las diferentes Tecnologías se realizó mediante dos tipos de consideraciones; la primera se refiere a la investigación y selección de varias evaluaciones de diferentes expertos de la comunidad de arquitectos y desarrolladores de aplicaciones JEE; y la segunda, a la investigación y experimentación directa con cada una de las implementaciones seleccionadas para el estudio, Apache My Faces, IceFaces y RichFaces.

Luego de realizar las evaluaciones mencionadas anteriormente, se seleccionó la implementación RichFaces de Jboss (RedHat), por la madurez, robustez y la suma de componentes y funcionalidades adicionales que este framework pone a disposición de los desarrolladores. Una de las características principales de RichFaces es la gran cantidad y calidad de recursos de información que posee y la continuidad en la publicación de nuevas versiones, que además de corregir errores, añaden funcionalidades.

Seguido se realizó la implementación de un piloto que muestra las características y funcionalidades que provee RichFaces. Este piloto consistió en la implementación de uno de los módulos del Sistema Gestor Fiducia Fondos, para

lo cual, se siguió un proceso metodológico que forma parte de la definición del proceso de Desarrollo de Software de la Empresa GESTORINC S.A.

Finalmente, tras crear el Piloto del Módulo de Seguridades del Sistema Gestor Fiducia Fondos, se pudo apreciar todas las bondades que brinda una aplicación RIA (Aplicaciones de Internet Enriquecidas) frente a una aplicación Web de tipo tradicional y en especial, las ventajas que ofrece la utilización del framework RichFaces.

# ABSTRACT

This Project is aimed at the evaluation of three implementations for the Java Server Faces 1.2 Technology with Ajax support to its corresponding application in the presentation layer of the “Gestor Fiducia Fondos” Software for Enterprise GESTORINC S.A.

The evaluation of different technologies JAVA SERVER FACES 1.2 with Ajax support, was conducted with two considerations: the first relates to the investigation and selection of various assessments of different experts in the community of architects and application developers JEE and the second, research and direct experimentation with each of the implementations selected for study, Apache My Faces, ICEfaces and RichFaces.

After carrying out the evaluations above, we selected the implementation JBoss’s RichFaces (RedHat), by the maturity, robustness and the large number of components and additional functionality that this framework offers to developers. One of the main features of RichFaces is the sheer quantity and quality of information resources and continuity in the new versions publications, in addition to correct errors, add functionality.

The final section of this project was carried out a pilot implementation that shows the features and functionality that provides RichFaces. The pilot was the implementation of one of the modules of System “Gestor Fiducia Fondos”, for

which, it followed a methodical process that is part of the process definition of Software Development Enterprise GESTORINC S.A.

Finally, after creating the Security Module Pilot of System “Gestor Fiducia Fondos”, was showed all the goodness that provides a RIA (Rich Internet Applications) against a traditional Web application and in particular the advantages of use the RichFaces framework.

# CAPÍTULO I

# GENERALIDADES

## 1.1. Introducción

La empresa GESTORINC S.A. es la empresa líder en el mercado ecuatoriano y latinoamericano en la provisión de software para la administración de Fideicomisos y Negocios relacionados con la Banca de Inversión.

El producto Gestor Fiducia Fondos, fue desarrollado hace 5 años utilizando tecnología Cliente Servidor propietaria de Oracle y está conformado por varios módulos que interactúan entre sí.

En la actualidad la empresa GESTORINC S.A., ha visto la necesidad de desarrollar un producto que reúna los 10 años de experiencia en el desarrollo de este tipo de software pero utilizando última tecnología.

GESTORINC S.A. ha seleccionado la plataforma JAVA[1] como plataforma de desarrollo ya que en la actualidad la mayoría de sus clientes que corresponden a grandes bancos a nivel nacional, regional y mundial definen en sus estándares corporativos el uso de la plataforma JAVA.

Al ser JAVA una plataforma extensa y con una variedad de opciones para el desarrollo de aplicaciones se ha decidido también el uso de la tecnología JSF[2]

[1] JAVA : Plataforma virtual de software desarrollada por Sun Microsystems.
[2] JSF : JAVA SERVER FACES

con soporte AJAX[3] para dar la característica de Aplicación de Internet Enriquecida RIA[4] al nuevo producto a ser desarrollado y de esta forma seguir liderando el mercado latinoamericano como en los últimos años.

## 1.2. Antecedentes

GESTORINC S.A. es una empresa de tecnología que provee soluciones aplicacionales para el sector financiero internacional desde 1997, dentro del ámbito de Banca de Inversión. La empresa ha suministrado al mercado soluciones basadas en las mejores prácticas de negocios y en las tecnologías más avanzadas.

Cuenta actualmente con importantes clientes en 8 países de Latinoamérica: México, Guatemala, Panamá, Venezuela, Colombia, Ecuador, Perú y Argentina.

Sus productos se basan en el Enfoque de Versión Única, con la concentración en un sistema global y replicable, complementados con el servicio de Soporte y Mantenimiento.

## 1.3. Situación actual

GESTORINC. S.A. basa sus soluciones en una sola aplicación madre y completa, que en base a su modularidad, funcionalidad y uso, plantea soluciones de clase mundial dentro del Ámbito de Banca de Inversión.

[3] AJAX: Asynchronous JavaScript + XML
[4] RIA : RICH INTERNET APPLICATIONS

La empresa cuenta en la actualidad con un sin número de productos de software, algunos de los cuales requieren de ciertas actualizaciones que permitan mantener el liderazgo en el mercado, como es el caso del producto Gestor Fiducia Fondos.

El sistema Gestor Fiducia Fondos, es una solución informática diseñada para el manejo y administración de Fideicomisos y Fondos (Fondos de Inversión, Fondos Mutuos, Fondos de Retiros y Pensiones) bajo el esquema de administración tanto del Activo como del Pasivo (Cuentas) y con la propia Administración de la Administradora (Banco o Fiduciaria) con más de 30 implementaciones instaladas y funcionando en 8 países de Latinoamérica.

### 1.4. Alcance

El proyecto de tesis incluirá el estudio de las principales implementaciones de la tecnología JSF con soporte AJAX.

Del estudio teórico realizado se seleccionará la implementación que cuente con las mejores características para desarrollar un Piloto del Módulo de Seguridades del producto Gestor Fiducia Fondos. El módulo de Seguridades del Sistema Gestor Fiducia Fondos tiene como principales características las siguientes: Definición de reglas globales del Sistema, Administración de Usuarios permitiendo la creación, modificación, cambio de estado y reseteo de clave de los mismos y Autogestión de claves.

## 1.5. Justificación

Los profesionales de la empresa GESTORINC S.A. cuentan con sólidos conocimientos en el desarrollo de aplicaciones Cliente Servidor, plataforma Oracle[5] y del Negocio relacionado a Fideicomisos, Fondos y Banca de Inversión.

GESTORINC S.A. ha seleccionado la plataforma JEE[6] para el desarrollo de la nueva versión de su sistema para Administración de Fideicomisos y Fondos, por la madurez, escalabilidad y confiabilidad de la misma.

Como parte del desarrollo de su nuevo producto se encuentra la definición de la tecnología a ser utilizada en la capa de presentación para lo cual se ha decidido utilizar la tecnología JSF 1.2[7] que es el actual estándar de la plataforma JAVA.

Sin embargo, en la actualidad existen diversas implementaciones de la tecnología JSF con soporte AJAX cada una de ellas con diferentes características y limitaciones, tantas que es necesaria la realización de un estudio a profundidad para la selección de la implementación que será utilizada en este nuevo reto de la compañía.

Es por esto que se ha definido la necesidad de auspiciar el desarrollo de este tema de tesis, el cual deberá incluir el estudio de las principales implementaciones de la tecnología JSF con soporte AJAX y consecutiva selección

[5] Oracle: Herramienta cliente/servidor para la gestión de Bases de Datos.
[6] JEE: Java Entreprise Edition
[7] JSF 1.2: Java Server Faces versión 1.2

de la mejor tecnología a través de él y su correspondiente demostración en el aplicativo en uno de sus principales módulos.

## 1.6. Objetivos

### 1.6.1. Objetivo General

Desarrollar la evaluación de tres implementaciones Java Server Faces 1.2 para su aplicación en el desarrollo del Módulo de Seguridades del Sistema Gestor Fiducia Fondos en GESTORINC S.A. mejorando así su productividad y consistencia.

### 1.6.2. Objetivos Específicos

- Realizar una evaluación de tres implementaciones Java Server Faces 1.2 con soporte para AJAX.
- Seleccionar la implementación que será adoptada por la compañía bajo la plataforma JEE como estándar del Sistema Gestor Fiducia Fondos.
- Desarrollar una implementación piloto del Módulo de Seguridades del Sistema Gestor Fiducia Fondos con la implementación de JSF seleccionada.

# CAPÍTULO II

# SISTEMA GESTOR FIDUCIA FONDOS

## 2.1. Introducción al Sistema Gestor Fiducia Fondos

Gestor Fiducia Fondos es una solución globalizada, especialmente diseñada para Fideicomisos, Fondos de Inversión, Fondos Mutuos, Fondos de Retiros y Pensiones y Administración de Activos.

Al ser una solución fiduciaria y de fondos, Gestor Fiducia Fondos cubre la totalidad del negocio Fiduciario ofreciendo sólidos controles de seguridad sobre su operación, ya sea para un Banco de Inversión, Administradora de Fondos de Inversión y Administradora de Pensiones, como también para Casas de Valores y las Tesorerías de Bancos Comerciales.

## 2.2. Características Principales

- Enfoque: Soluciones construidas de manera modular, que cubren la totalidad del negocio con módulos a utilizarse por demanda y necesidad.
- Flexibilidad y Control: Creación flexible, dinámica y altamente paramétrica de Productos del Negocio, sin restricción de tipo de productos pero con control sobre el negocio y su operación.
- 100% Operación: Cobertura de Front y Back Office, de punta a punta del negocio, sin áreas desatendidas y no contempladas en la operación. Incluye proceso contable automático y flexible.

- Integridad: Integridad natural entre módulos, procesos y funcionalidades, contemplando un solo acceso.
- Canales/Autoservicio: Posibilidad de incorporar Canales de Interacción con el cliente, vía internet, cajeros y callcenter.
- Históricos: Manejo ordenado y bajo integridad de Históricos de Información, en los distintos módulos y funcionalidades del sistema.
- Estados de Cuenta: Manejo flexible de Estados de Cuenta, con detalles de información requerido y posibilidad de generación a fechas anteriores.
- Inteligencia de Negocios: Modelo de datos documentado e integral, utilizando herramientas de explotación de información.
- Política de Sustentabilidad: GESTORINC S.A. mantiene una relación a largo plazo con sus clientes mediante un plan de sustentabilidad que incorpora los avances tecnológicos, operacionales y funcionales en sus productos.

## 2.3. Características Tecnológicas

- Sistema Integrado Modular
- Arquitectura Cliente – Servidor, con acceso vía Browser
- Basado en plataforma ORACLE
- Multiempresa, Multimoneda, Multiportafolio y Multiagencia
- Sistema Operativo: Unix, Windows, Linux
- Sólidos controles de seguridad:
    - Varios niveles de acceso de acuerdo al perfil del usuario
- Auditoría completa de cambios

## 2.4. Mercado Objetivo del Producto Gestor Fiducia Fondos

Dirigido a bancos, instituciones financieras, casas de bolsa, aseguradoras, administradoras de fondos de inversión, fondos mutuos, pensiones o retiro. Permite la administración de fideicomisos de inversión, inmobiliarios, de garantía, administración y pagos, titularización, encargos fiduciarios. Cubre tanto la administración del pasivo como del activo del fondo, y los procesos operativos de la administradora.

## 2.5. Módulos del Sistema Gestor Fiducia Fondos

### 2.5.1. Seguridades

Controles de acceso por usuario, pantalla y/o producto fiduciario en base a perfiles y horarios de trabajo.

### 2.5.2. Personas

Módulo central donde se registran todas las personas o entidades que interactúan con el sistema en el manejo fiduciario y de fondos de inversión. Controla la información global de los clientes y las personas relacionadas de manera integrada.

### 2.5.3. Contabilidad

Interactúa con los otros módulos generando de manera automática los movimientos contables correspondientes. Contabilidad independiente opcional por cada producto, fideicomiso, encargo fiduciario o fondo de inversión (patrimonio autónomo), contemplando cierre masivo o individual por negocio.

Plan de cuentas parametrizable por producto, manejo multimoneda, tomando en cuenta ajuste por diferencial cambiario y ajuste por inflación.

### 2.5.4. Producto

Actúa como punto inicial en la creación y operación de los fideicomisos/fondos. Apoya en la parametrización de controles, normas del negocio y transacciones de ingreso y egreso, interactúa con otros módulos para particularizar las características de cada fideicomiso o fondo, permite acceso a información de clientes en lo referente a compromisos o asuntos pendientes, relacionando tareas, fechas y responsables.

### 2.5.5. Auditoria

Genera una bitácora de auditoría de todos los campos del sistema registrando la información de usuario, equipo, fecha, hora, valor anterior y valor nuevo para las operaciones de ingreso, cambio o eliminación de datos.

### 2.5.6. Partícipes

Permite el registro de Fideicomitentes/Partícipes y Beneficiarios así como el ingreso de Clientes que aportan a un fideicomiso o fondo (Cuentas individuales de aportación). Configura las transacciones de aporte o retiro, estableciendo controles de horario, control de firmas, co-partícipes, montos máximos, retenciones, disponibilidad.

Posibilita manejar subcuentas para aportes de partícipe, patronales, voluntarios, extraordinarios y otros, permite la generación de transacciones en volumen, vía archivo parametrizable, para clientes corporativos.

### 2.5.7. Caja / Bancos

Cumple con la función operativa de registro y control de ingresos y egresos, permitiendo diferenciarlos por tipos, conciliación bancaria automática, generación de comprobantes de retención y comprobantes de ingreso (facturas), con sus procesos de control y consulta.

### 2.5.8. Titularización

Administración de pasivo-patrimonio a través de sus sub-módulos: Libro de Accionistas, Dividendos y Emisión de Deuda Primaria (Captaciones).

Manejo del activo a titularizar a través de los módulos de Crédito, Inversiones, Bienes, Presupuesto, Caja y Bancos, según el tipo de activo a titularizar. Permite registrar las transacciones de “Emisión Primaria” realizadas respecto a la propiedad de los diferentes títulos de participación que se emiten, registra las transacciones de “Mercado Secundario” que se realicen con los títulos valores llevando un histórico de la propiedad de cada uno de ellos y permitiendo el agrupamiento o división según las negociaciones que se efectúen.

Posibilita la generación de dividendos registrando dividendo acción y/o dividendo efectivo.

### 2.5.9. Garantía

Posibilita generar, administrar, consultar y controlar los Certificados de Garantía con sus respectivas condiciones y características totales, verifica los certificados emitidos frente a los valores de los activos registrados y sus diferentes porcentajes de garantía, así como la amortización de capital de los créditos. Registra el crédito relacionado a la garantía en el Módulo de Crédito, sin contabilizarlo dentro del fideicomiso, para controlar el avance de los pagos.

### 2.5.10. Inversiones

Incluye: Políticas de Portafolio, Control de Riesgos, Manejo de Liquidez, Mesa de Dinero, Mesa de Capitales, Mesa de Cambios/Divisas y Derivados. Permite la generación de captaciones como Tesorería Pasiva (Emisión de Obligaciones), manejo del flujo de fondos individual y consolidado. Inversiones individuales y consolidas, proyección del flujo de caja entre períodos de tiempo y planificación de las necesidades de liquidez de los fondos, utilización en línea de herramientas financieras para apoyo en cálculo y negociación, realiza el control de los títulos entregados en custodia.

### 2.5.11. Crédito

Permite el registro y administración de cartera como un activo del patrimonio, así como créditos de compra de bienes en Fideicomisos Inmobiliarios o de Administración y adelantos de dinero vía crédito en Fondos o Fideicomisos de Inversión. Además registra los créditos de Fideicomisos en Garantía.

Almacena las tablas de amortización y/o plan de pagos incluyendo intereses, moras, dividendos y moneda de las obligaciones, así como valores adicionales al crédito, controla el cumplimento de pagos y calcula mora/multas, en caso de existir.

### 2.5.12. Bienes

Registra el aporte o inclusión de diferentes tipos de bienes al patrimonio autónomo del producto, registra para cada bien sus avalúos, seguros, impuestos, descripciones técnicas, documentación legal y de propiedad y porcentajes de participación. Soporte a imágenes que permitan identificar y registrar el bien a través de medios gráficos y fotográficos.

### 2.5.13. Mensajería

Permite manejar y controlar diversidad de compromisos de la administración fiduciaria a través de un utilitario que recuerda actividades, controla su lectura y confirma su ejecución.

### 2.5.14. Honorarios / Comisiones

Registro de una tabla universal de comisiones, tomando en cuenta condiciones particulares y específicas de cada fideicomiso o fondo. Se administra a través de fórmulas que permiten conformar las diferentes comisiones diseñadas directamente por el usuario final.

### 2.5.15. Proyectos / Presupuestos

Permite registrar y controlar un presupuesto sobre cualquier proyecto y efectuar comparaciones con el movimiento real. Alimentación de datos reales a través de la contabilidad así como control de movimientos.

## 2.6. Nuevas Características del Sistema Gestor Fiducia Fondos

Este proyecto, es una evolución tecnológica del actual Producto Gestor, que la empresa ha visto la necesidad de desarrollar para estar acorde a los nuevos estándares y tecnologías mundiales y sus principales características son las siguientes:

- Será desarrollado en su totalidad bajo la Plataforma JEE y su metodología de Desarrollo será propia de la empresa GESTORINC S.A.
- Utilizará diferentes tecnologías y frameworks para cada una de las capas lógicas del mismo.
- En la capa de Reglas de Negocio se utilizará Spring Framework y finalmente para la capa de Presentación la tecnología JSF con soporte para AJAX.
- En la capa de Persistencia se utilizará la especificación Java Persistence API[8] con la implementación de Hibernate.
- Deberá ser multilenguaje, multiempresa, multiagencia y multimoneda.

[8] API: Application Programming Interface

# CAPÍTULO III

# MARCO TEÓRICO

## 3.1. Rich Internet Applications (RIA) / Aplicaciones de Internet Enriquecidas

### 3.1.1. Introducción

Típicamente las aplicaciones han seguido un modelo simple: enviar y recibir, en donde el cliente visualiza una página, llena una forma o siguen un enlace, envía la petición al servidor y el servidor devuelve una página nueva al cliente. Este proceso se repite hasta que la sesión finaliza. El navegador se limita únicamente a mostrar la información y toda la lógica de la aplicación reside en el servidor.

Las aplicaciones de internet enriquecidas llevan las cosas más allá, creando una experiencia enriquecida al usuario, una aplicación de este tipo es ejecutada en el interior de un navegador y se diferencia de las aplicaciones web corrientes por presentar un comportamiento similar al de las aplicaciones de escritorio. El aspecto fundamental es que no se precisa una actualización de la página completa, actúa como interfaz de usuario cada vez que se interactúa con la funcionalidad que ofrece el programa.

Las aplicaciones de internet enriquecidas son independientes del sistema operativo y se ejecutan sin necesidad de una instalación previa. Esto se consigue colocando algo de la lógica de la aplicación en el cliente y permitiendo crear una interfaz similar al de una aplicación de escritorio.

Como consecuencia de dotar de lógica al lado del cliente, el navegador debe ser capaz de ejecutar las RIAs y para ello es necesario instalar en los mismos algunos plug-ins.

RIA se apoya más sobre un desarrollo cliente-servidor y no en un desarrollo web tradicional, en donde el estado se mantiene en el servidor en sesiones. El cliente sabe acerca de sí mismo y el tipo de datos que está solicitando y únicamente solicita los datos que necesita sin ninguna otra información.

### 3.1.2. Definición

Son un nuevo tipo de aplicaciones web que disponen de casi la totalidad de las características de las aplicaciones tradicionales y que a través de un navegador web y plug-ins agregan las características adicionales.

Su objetivo es mejorar la interactividad con el usuario, puesto que constituyen una mezcla de las mejores características que brindan las aplicaciones web y las aplicaciones tradicionales.

Además, en los entornos RIA se evita el tráfico entre el cliente y el servidor, ya que, no existen las recargas de página como anteriormente con cada ejecución sobre un enlace y desde el principio se carga toda la aplicación, de esta manera, sólo se produce comunicación con el servidor cuando se necesitan datos externos como datos de una Base de Datos o de otros archivos externos.

### 3.1.3. Evolución

Las RIA no son nuevas, han estado presentes casi desde la aparición de la web, en ese entonces los plug-ins hechos en Flash proporcionaban la experiencia enriquecida que los navegadores permitían. En el año 2000 Jakob Nielsen un conocido investigador en tecnologías web llamó a las RIAs como objetos Flash y 99% malas, según Nielsen los objetos multimedia enriquecidos serían mejor si se les quitase las partes Flash. Las RIA se han vuelto populares y esto se debe a que se volvieron amigables para el programador.

En el año 2004, Adobe introdujo una nueva tecnología para la programación en la capa de presentación llamada Flex, la cual permitía la programación de aplicaciones Flash, capaces de manejar datos. Con nuevas APIs y un ambiente de desarrollo orientado a programadores y no ha diseñadores, Flash se convirtió en un plataforma para aplicaciones enriquecidas y no solo en una plataforma de animación enriquecida.

Luego en el año 2006 las RIAs se fortalecieron con la presentación de Silverlight de Microsoft. Al igual que Flex, Silverlight está basado en una tecnología basada en plug-ins que se ejecuta sobre un navegador y permite desarrollar aplicaciones que manejen datos.

Las RIAs son herramientas invaluables para las empresas, pues en un entorno empresarial globalizado como el actual, en el que los clientes son más exigentes que nunca y la fidelidad a la marca es cada vez más difícil de lograr, un compromiso sólido de los clientes es esencial para el éxito de una empresa.

Las aplicaciones RIA pueden hacer que las interacciones de los clientes sean llamativas, dinámicas y útiles; en una palabra: atractivas.

En marzo del 2007, Forrester Research publicó "The Business Case for Rich Internet Applications" (El caso empresarial para aplicaciones de Internet sofisticadas), un informe basado en entrevistas a diseñadores y proveedores de tecnología RIA, así como a clientes de Forrester Research. El informe reveló que las RIA bien diseñadas pueden producir resultados llamativos que ayuden a demostrar el valor de las inversiones actuales y exponer los argumentos a favor de que se desarrollen más proyectos RIA en el futuro.

Según las conclusiones, las empresas que miden el impacto empresarial de sus RIA afirman que las aplicaciones sofisticadas cumplen o superan sus objetivos. Conclusiones específicas demuestran que una mayor facilidad de uso para los clientes que utilicen aplicaciones RIA sirve de impulso para mayores índices de conversión y pedidos de mayor tamaño. Cada vez más compradores potenciales se convierten en compradores reales cuando son capaces de encontrar con facilidad el equilibrio entre las opciones de productos y los costes en tiempo real. Además, debido a que es cada vez más fácil completar complejos pedidos en línea, son cada vez menos los clientes que abandonan.

Adicionalmente, la capacidad de las RIA para incorporar medios sofisticados da dividendos. Los medios sofisticados ayudan a incrementar los márgenes. Las RIA no sólo permiten mejores configuraciones, también permiten a las empresas incorporar vídeo y otros contenidos de ayuda contextual a las

aplicaciones. El porcentaje de usuarios que acceden a este tipo de funciones de ayuda que se convierten en compradores reales es mucho mayor que en el caso de los que no lo hacen.

### 3.1.4. Arquitectura

Comúnmente las RIA se separan en tres capas que son:

- Del lado del servidor – Capa de Negocio
- Del lado del cliente – Capa de Presentación
- Una capa intermedia encargada de comunicar el lado del cliente con el lado del servidor.


**Figura 3.1:** Arquitectura de una Aplicación de Internet Enriquecida (RIA)[9]

[9] Fuente: http://www.adobe.com/resources/business/rich_internet_apps/#open

#### 3.1.4.1. Capa de negocios

La capa de negocios es el núcleo de la aplicación del lado del cliente y está completamente encapsulada de tal forma que no es visible para el usuario. Su trabajo consiste en proporcionar siempre los datos necesarios a la interfaz de usuario, cuando esta lo necesite.

Normalmente esta capa incluye: el modelo de datos, las clases de la aplicación, un modelo de eventos, su propia capa de servicios para comunicarse con la base de datos desde el lado del cliente (si la aplicación lo requiere) y el servidor, así como una clase controladora primaria que la enlace con la interfaz de usuario.

#### 3.1.4.2. Capa de comportamiento y personalización de componentes

Es el intermediario con la capa de usuario. Su función es la de escuchar los eventos disparados por la capa de negocios e instanciarlos junto con sus respectivos manejadores de evento, lo que puede: mostrar un componente tipo pop-up o cambiar el estado de la aplicación y enviar comandos a uno de los componentes de la interfaz gráfica.

En un componente y en su comportamiento se puede trabajar sobre cosas como: la animación y el movimiento, la apariencia, manejo del API de graficación o la creación de clases que controlan el comportamiento de los objetos mostrados.

#### 3.1.4.3. Capa de presentación

Esta es la capa donde se encuentra la interfaz de usuario, que es la encargada de interactuar con el usuario mostrando y recabando información. También es donde se encuentran los archivos que definen la forma en que se ve la interfaz.

### 3.1.5. Funcionamiento

Esencialmente las RIAs toman una parte de la lógica de la aplicación del servidor y la incorporan en el cliente. Así no es necesario que cada vez que un usuario realiza una petición se deba ir al servidor para realizar la tarea, realizándose ciertas partes de la lógica de negocio en el cliente.

#### 3.1.5.1. Prefetching (Precarga)

La técnica de Precarga se anticipa al comportamiento del usuario y carga anticipadamente contenido de la aplicación para que se encuentre disponible cuando la necesite el cliente. Un ejemplo de esto lo podemos ver en Google Maps, cuando una ubicación es mapeada el usuario puede buscar locaciones vecinas sin que existan retardos. Esto es posible ya que las imágenes alrededor de la imagen visible se han precargado (prefetching) en caso de que el usuario quiera navegar por esas zonas.

#### 3.1.5.2. State Synchronization (Sincronización de Estado)

Como se ha definido parte de la lógica de la aplicación reside en el lado del cliente, y se requiere de un esquema que permita sincronizar ambos lados según

el estado de la sesión. Esta comunicación es esencial para mantener un estado consistente entre todos los componentes de la aplicación.

#### 3.1.5.3. Real-time Update (Actualización en tiempo real)

Las RIAs ofrecen actualizaciones en tiempo real a la aplicación. Por ejemplo un programa de mensajería utiliza el lado del cliente para preguntar periódicamente al servidor por nuevos mensajes. Las RIAs cuando requieren de una interfaz en tiempo real, se encuentran comunicándose constantemente con el servidor, esto se conoce como Active Polling. Esta comunicación constante tiene un cierto impacto en la red.

### 3.1.6. Características clave

- Comunicaciones avanzadas: Con servidores que soporten nuevas tecnologías se puede mejorar la experiencia del usuario al utilizar protocolos de red optimizados con entradas y salidas asíncronas. Se requiere de una conexión de banda ancha confiable.
- Consistencia: La interfaz de usuario y las experiencias pueden ser controladas por el sistema operativo que se utilice, el monitoreo del rendimiento y diagnóstico de errores puede ser difícil.
- Offline: Puede ser soportada reteniendo el estado en la máquina cliente.
- Rendimiento: Puede ser mejorado dependiendo de la aplicación y de las características de la red. Aplicaciones que pueden procesar localmente en el cliente evitan viajes hacia el servidor lo que incrementa su rendimiento. Dándole más trabajo al cliente también se puede incrementar el rendimiento del servidor.

- Seguridad: Mejora la seguridad por medio de actualizaciones automáticas.
- Riqueza: Añadiendo características que no son nativas en los "navegadores web" como la captura de video.

### 3.1.7. Impacto de las RIAs en la red

Con el auge de las RIAs, los administradores de redes notaron un incremento en el tráfico y su impacto en la infraestructura de servidores y redes. A continuación se expondrán las formas en que las RIA han afectado a las redes debido al tráfico generado por la constante comunicación entre cliente y servidor.

#### 3.1.7.1. Mayor cantidad de conexiones TCP[10]

Nuevas conexiones son requeridas por las RIAs para facilitar la comunicación entre cliente y servidor, estas son requeridas para la precarga, sincronización en tiempo real, sincronización de estado, entre otras.

#### 3.1.7.2. Mayor cantidad de transacciones HTTP[11]

Por su naturaleza las RIAs cargan porciones de una página web, los mecanismos de sincronización en tiempo real requieren intercambiar información HTTP en cada iteración, incluso si el tamaño de las páginas disminuye el número de intercambios aumenta. Normalmente se puede apreciar muchas transacciones pequeñas en lugar de pocas y grandes transacciones. Esto generalmente causa un esfuerzo entre los servidores y cualquier elemento proxy entre el navegador y la aplicación.

[10] TCP:Transmission Control Protocol
[11] HTTP: Hypertext Transfer Protocol

#### 3.1.7.3. Páginas más grandes

Las interfaces de las RIA proporcionan código extra al navegador sea por medio de flash o de JavaScript. Este código adicional está embebido en la página (interfaz de usuario) o es precargado como un objeto, de cualquier forma una RIA es mucho más grande que una aplicación web tradicional.

#### 3.1.7.4. Mayor ancho de banda requerido

Las RIAs mantienen una comunicación continua entre servidor y cliente, que es vital para la operación de la aplicación. Las precargas por ejemplo, requieren que muchos objetos sean solicitados al servidor, objetos que pueden no ser utilizados nunca. En estas situaciones es requerido un mayor ancho de banda y con el consiguiente impacto en la infraestructura de la red. Este impacto varía de aplicación a aplicación, es importante entender que el añadir inteligencia al cliente tiene un costo.

### 3.1.8. Desafíos para el desarrollo de RIA

El esfuerzo de desarrollar RIAs requiere de armar y mantener distintos equipos de profesionales trabajando en diferentes aspectos de la aplicación. Como resultado, el diseño, planificación y administración del proyecto se vuelven más riesgosos y caros dada su complejidad, no es menos cierto que su costo de propiedad en cambio es bajo y proporciona mayor seguridad y una mejor experiencia para el usuario, ya que este no debe preocuparse de actualizaciones y configuraciones.

#### 3.1.8.1. El acoplamiento

Las aplicaciones cliente-servidor tradicionales cuentan con una conexión permanente con un diseño fuertemente acoplado, apenas se requiere manejar explícitamente y menos preservar los distintos estados de lógica de la aplicación. Por el contrario en las aplicaciones web que se encuentran centralizadas en el servidor dejan desacoplado o débilmente acoplado al cliente.

Mientras las aplicaciones web realicen pequeños y simples procesos lógicos o su interactividad sea limitada, estas pueden implementarse con arquitecturas estándar y un manejo simple de sesión.

La arquitectura RIA procura proveer el confort de la experiencia del usuario altamente interactiva y fuertemente acoplada, pero con una configuración débilmente acoplada. Esto requiere un sofisticado diseño de sistemas y capacidades de programación que raramente se encuentran en equipos de desarrollo focalizados en soluciones de negocios.

#### 3.1.8.2. Superando las limitaciones del ancho de banda

Con el incremento sostenido del ancho de banda, cada vez más recursos se encuentran disponibles para las aplicaciones web, junto con el uso de tecnologías con AJAX la interacción de las RIAs se acerca a la velocidad de aplicaciones cliente servidor dentro de una red de área local. Un ejemplo de esto lo vemos en la función Google Suggest que nos muestra un rango de términos de las búsquedas más populares a medida que uno ingresa las letras en el campo de búsqueda.

Para utilizar esta funcionalidad en aplicaciones fuertemente acopladas, con decenas de campos activos por pantalla la aplicación RIA debe soportar más lógica en el lado del cliente y particionar el procesamiento entre el Servidor y el Cliente. De forma que se trabaja con dos procesos físicamente separados y lógicamente dependientes y que se ejecutan en tándem.

### 3.1.9. Frameworks para Desarrollo de Aplicaciones RIA

Las aplicaciones RIA se encuentra en auge y por lo tanto se pueden encontrar una gran variedad de frameworks que facilitan su desarrollo, entre los que se encuentran:

- Flex: Es un framework de Adobe Open Source, basado en XML[12] y soporta los patrones más populares de diseño, soporta ActionScript 3 que es un lenguaje orientado a objetos. Cuenta con una extensa colección de objetos.
- Silverlight: Es el framework de Microsoft basado en el framework .NET provee herramientas para el desarrollo de RIA, es cross-browser, cross-plataform y cross-device. Así como una gran colección de widgets listos para utilizar.
- Active Widget: Es básicamente una librería de componentes JavaScript diseñados al estilo AJAX, su interfaz provee elementos visuales como datagrids, trees, combos y tabs, esto permite cargar el código desde archivos XML y JavaScript.

[12] XML: Extensible Markup Language

- ZK Direct RIA: Este framework está enfocado en aumentar la productividad del programador al integrar transparentemente el backend y el frontend de las aplicaciones. Permite el acceso directo a bases de datos, servicios web y las interfaces de acceso, está basado en Java y esta orientado a eventos.
- Yahoo User Interface Library (YUI Library): Es un conjunto de utilidades y controles escritos en JavaScript y CSS[13], para la creación de RIAs se utilizan técnicas como el DOM[14] scripting, DHTML[15] y AJAX. YUI es escalable, rápido y robusto.
- Dojo Toolkit: Soporta AJAX, JSON[16], utilidades para lenguajes, widgets entre otras utilidades.
- jQuery UI: Provee abstracción de bajo nivel para la interacción y animación, utiliza sus propias librerías JavaScript y widgets personalizables.

### 3.1.10. Beneficios

A pesar de que el desarrollo de aplicaciones multimedia para navegadores web está mucho más limitado y es más difícil que otro tipo de aplicaciones de escritorio, los esfuerzos se justifican por varios motivos:

- Las actualizaciones hacia nuevas versiones son automáticas.
- Ofrecen aplicaciones interactivas que no se pueden obtener utilizando sólo HTML, incluyendo arrastrar y pegar, cálculos en el lado del cliente sin la necesidad de enviar la información al servidor.

[13] CSS: Cascading Style Sheets
[14] DOM: Document Object Model
[15] DHTML: Dynamic HyperText Markup Language
[16] JSON: JavaScript Object Notation

- Se pueden utilizar desde cualquier ordenador con una conexión a Internet sin depender del sistema operativo que este utilice.
- Generalmente es menos probable la infección por virus, que utilizando por ejemplo programas ejecutables.
- No necesitan instalación, sólo es necesario mantener actualizado el navegador web.
- Más capacidad de respuesta, ya que el usuario interactúa directamente con el servidor, sin necesidad de recargar la página.
- Evita la problemática del uso de diferentes navegadores al abstraerse de ellos a través de un framework.

## 3.2. Asynchronous JavaScript And XML (AJAX)

### 3.2.1. Introducción

El término AJAX es un acrónimo de Asynchronous JavaScript + XML, se utilizó por primera vez en el artículo "AJAX: A New Approach to Web Applications", publicado por Jesse James Garrett el 18 de Febrero de 2005.

Hasta ese momento, no existía un término normalizado que hiciera referencia a un nuevo tipo de aplicación web que estaba apareciendo y lo define de la siguiente forma: "AJAX no es una tecnología en sí mismo en realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes."[17]

[17] Fuente: http://www.uberbin.net/archivos/internet/ajax-un-nuevo-acercamiento-a-aplicaciones-web.php

### 3.2.2. Definición

AJAX es un conjunto de tecnologías de desarrollo web para la creación de aplicaciones interactivas o RIA (Rich Internet Applications), las mismas que se ejecutan en el cliente (navegador de los usuarios), mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. Permitiendo así, realizar cambios sobre las páginas sin necesidad de recargarlas, lo que incrementa la interactividad, velocidad y usabilidad en las aplicaciones.

AJAX es una combinación de las siguientes tecnologías:

- XHTML[18] y CSS, para crear una presentación basada en estándares, diseño que acompaña a la información.
- DOCUMENT OBJECT MODEL (DOM), para la interacción y manipulación dinámica de la presentación. Accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
- XML, XSLT[19] y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información. En algunos frameworks y en algunas situaciones concretas, se usa un objeto iframe en lugar del XMLHttpRequest para realizar dichos intercambios
- JavaScript, para unir todas las demás tecnologías.

[18] XHTML: Extensible Hypertext Markup Language
[19] XSLT: Extensible Stylesheet Transformations

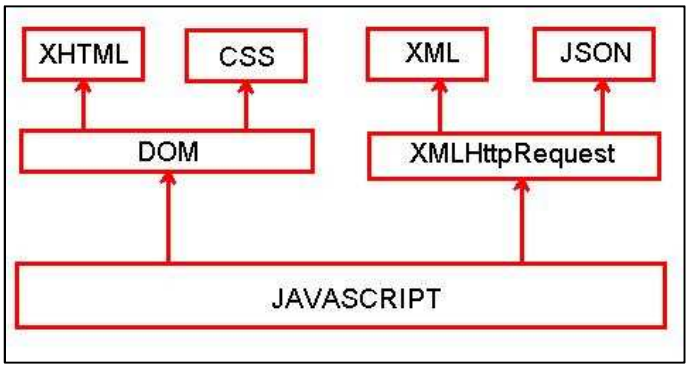


**Figura 3.2:** Combinación de Tecnologías para AJAX[20]

### 3.2.3. Evolución

A pesar de que el término "AJAX" fuese creado en el año 2005, la historia de las tecnologías que permiten AJAX se remonta a una década antes con la iniciativa de Microsoft en el desarrollo de Scripting Remoto. Sin embargo, las técnicas para la carga asíncrona de contenidos en una página existente sin requerir recarga completa remontan al tiempo del elemento *iframe* (introducido en Internet Explorer 3 en el año 1996) y el tipo de elemento *layer* (introducido en Netscape 4 en el año 1997, abandonado durante las primeras etapas de desarrollo de Mozilla). Ambos tipos de elementos tenían el atributo "*src*" que podía tomar cualquier dirección URL[21] externa, y con la carga de una página que contenga JavaScript que manipule la página paterna, pueden lograrse efectos parecidos al AJAX.

El Microsoft's Remote Scripting (o MSRS, introducido en el año 1998) resultó un sustituto más elegante para estas técnicas, con envío de datos a través

[20] Fuente: http://coronet.iicm.tugraz.at/lectures/mmis/material/images/ajax.png
[21] URL: Uniform Resource Locator

de un applet Java el cual se puede comunicar con el cliente usando JavaScript. Esta técnica funcionó en ambos navegadores, Internet Explorer versión 4 y Netscape Navigator versión 4.

Los primeros ejemplos incluyen la librería JSRS[22] en el 2000, la introducción a la técnica imagen/cookie en el mismo año y la técnica JavaScript on Demand en 2002. En ese año, se realizó una modificación por parte de la comunidad de usuarios al Microsoft's Remote Scripting para reemplazar el applet Java por XMLHttpRequest.

Desde que XMLHttpRequest está implementado en la mayoría de los navegadores, raramente se usan técnicas alternativas. Sin embargo, todavía se utilizan donde se requiere una mayor compatibilidad, una reducida implementación, o acceso cruzado entre sitios web.

### 3.2.4. Funcionamiento

El modelo clásico de aplicaciones Web funciona de esta forma: La mayoría de las acciones del usuario en la interfaz disparan un requerimiento HTTP al servidor web. El servidor efectúa un proceso y devuelve una página HTML al cliente. Este modelo es bueno para el hipertexto, pero no necesariamente bueno para las aplicaciones de software.

Este acercamiento tiene mucho sentido a nivel técnico, pero no lo tiene para una gran experiencia de usuario.  Mientras el servidor está haciendo su

[22] JSRS: Javascript Remote Scripting

parte, el usuario está esperando. Y, en cada paso de la tarea, el usuario siempre tiene que esperar.

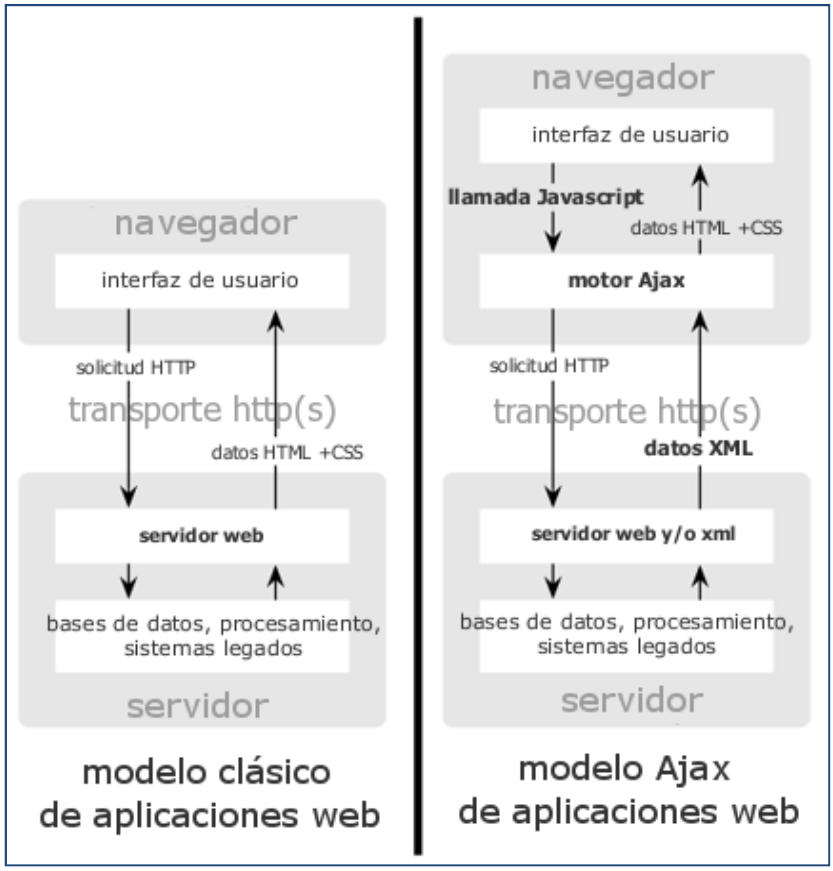


**Figura 3.3:** Funcionamiento Aplicaciones Web Clásicas vs Modelo AJAX[23]

AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano, pues el navegador carga al motor AJAX (escrito en JavaScript y usualmente encapsulado en un frame oculto).

[23] Fuente: http://www.sicuma.uma.es/sicuma/independientes/argentina08/LaRed-Eugenia/Inform4.jpg

Este motor es el responsable de presentar la interfaz que el usuario ve y por comunicarse con el servidor en nombre del usuario.

El motor AJAX permite que la interacción del usuario con la aplicación suceda asincrónicamente (independientemente de la comunicación con el servidor).

Las aplicaciones construidas con AJAX eliminan la recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor.

La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la respuesta del servidor.

Las peticiones HTTP al servidor se sustituyen por peticiones JavaScript que se realizan al elemento encargado de AJAX. Las peticiones más simples no requieren intervención del servidor, por lo que la respuesta es inmediata. Si la interacción requiere una respuesta del servidor, la petición se realiza de forma asíncrona mediante AJAX.

En este caso, la interacción del usuario tampoco se ve interrumpida por recargas de página o largas esperas por la respuesta del servidor.

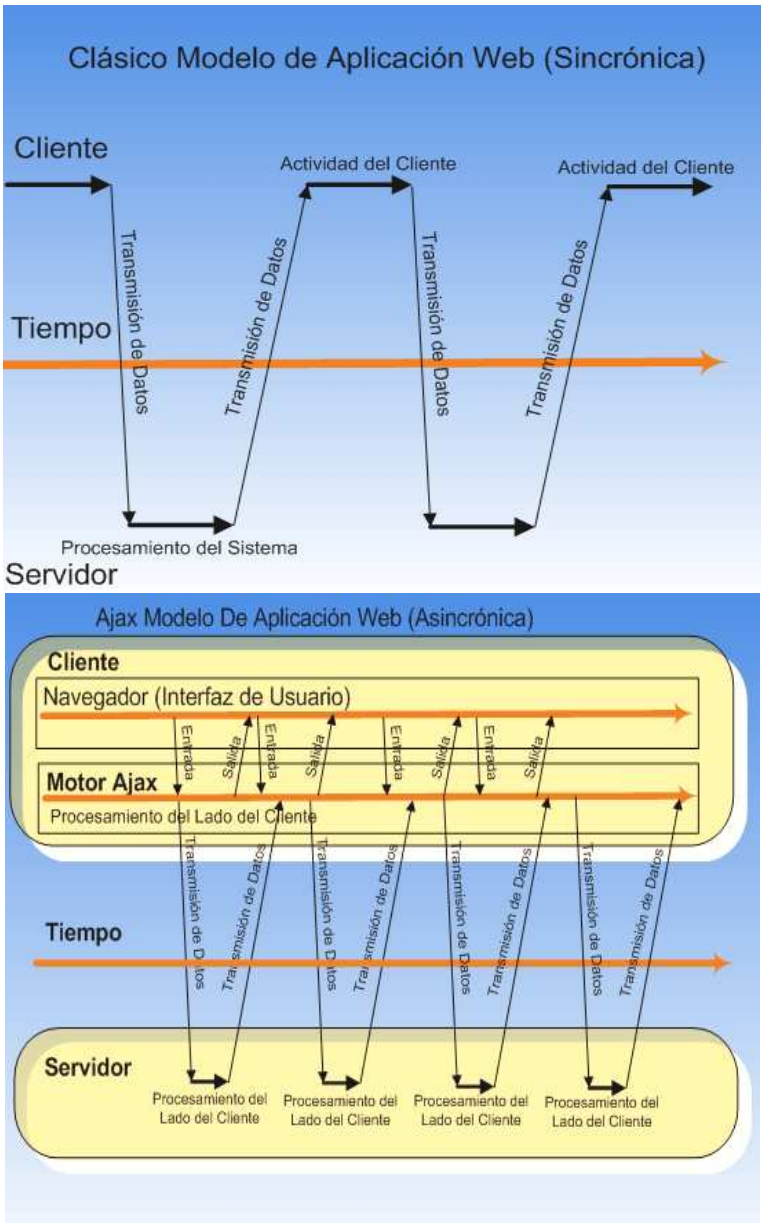


**Figura 3.4:** Comunicación Síncrona vs Comunicación Asíncrona de una Aplicación Web[24]

### 3.2.5. Arquitectura

La arquitectura de una aplicación AJAX difiere de una aplicación web estándar en tres puntos:

[24] Fuente: http://www.webtaller.com/images/contenidos/articulos/usabilidad_ajax1.png

- Existe un motor AJAX en el lado del cliente que sirve como intermediario entre la interfaz de usuario y el servidor.
- Las acciones realizadas por el usuario llaman al motor AJAX en lugar de acudir al servidor.
- XML se encarga de transmitir datos entre el servidor y el motor AJAX en el lado del cliente.

La clave del modelo AJAX reside en su motor. Sin este, todos los eventos disparados por el usuario deberían enviarse para su ejecución en el servidor. Las ventajas de este tipo de arquitectura son las siguientes:

- La percepción del usuario se ve mejorada, dado que la carga inicial de la página es muy rápida, trayendo luego los datos mediante llamadas AJAX.
- Se reduce el ancho de banda ya que sólo el contenido necesario se pasa entre cliente y servidor.
- Se reduce el trabajo en el servidor. El server se encarga sólo de servir datos, ahora la lógica de visualización ocurre en el cliente.
- Se tiene mayor flexibilidad para balancear la carga. Se puede separar físicamente los servidores de las páginas web “estáticas” (poca carga), de los servidores que sirven los servicios de datos (probablemente más carga).

### 3.2.6. El objeto XMLHttpRequest

También se conoce como XMLHTTP (*Extensible Markup Language / Hypertext Transfer Protocol*), es una interfaz empleada para realizar peticiones

HTTP y HTTPS[25] a servidores Web. Para los datos transferidos se usa cualquier codificación basada en texto, incluyendo: texto plano, XML, HTML y codificaciones particulares específicas. La interfaz se presenta como una clase de la cual una aplicación cliente puede generar tantas instancias como necesite para manejar el diálogo con el servidor. El uso más popular de esta interfaz es proporcionar contenido dinámico y actualizaciones asíncronas en páginas Web mediante tecnologías construidas sobre ella como AJAX.

### 3.2.7. DOM (Document Object model/ Modelo de Objetos del Documento)

Cuando se definió el lenguaje XML, surgió la necesidad de procesar y manipular el contenido de los archivos XML mediante los lenguajes de programación tradicionales. XML es un lenguaje sencillo de escribir pero complejo para procesar y manipular de forma eficiente. Por este motivo, surgieron algunas técnicas entre las que se encuentra DOM.

DOM es un conjunto de utilidades específicamente diseñadas para manipular documentos XML. Por extensión, DOM también se puede utilizar para manipular documentos XHTML y HTML. Técnicamente, DOM es una API de funciones que se pueden utilizar para manipular las páginas XHTML de forma rápida y eficiente.

Antes de poder utilizar sus funciones, DOM transforma internamente el archivo XML original en una estructura más fácil de manejar formada por una

[25] HTTPS: Hypertext Transfer Protocol Secure

jerarquía de nodos. De esta forma, DOM transforma el código XML en una serie de nodos interconectados en forma de árbol.

El árbol generado no sólo representa los contenidos del archivo original (mediante los nodos del árbol) sino que también representa sus relaciones (mediante las ramas del árbol que conectan los nodos).

Con el Modelo de Objetos del Documento los programadores pueden construir documentos, navegar por su estructura, y añadir, modificar o eliminar elementos y contenido.

Se puede acceder a cualquier elemento que se encuentre en un documento HTML o XML, y se puede modificar, eliminar o añadir usando el Modelo de Objetos del Documento, salvo algunas excepciones. Para el desarrollador la jerarquía que proporciona el DOM es obtener para cada objeto una referencia, una dirección, a través de la cual acceder a las propiedades del objeto. El primer objeto en el nivel de jerarquía es el objeto 'window'. Todo está enmarcado dentro de la ventana del navegador.

Cuando apareció una primera aproximación al DOM, los desarrolladores inmediatamente vieron las posibilidades que se abrían para añadir interactividad a las estáticas páginas web de entonces. El problema no tardó en surgir, puesto que los dos navegadores principales de aquellos años divergieron, desarrollando cada uno de ellos un modelo de DOM con múltiples puntos de incompatibilidad

con el otro. En los últimos tiempos los esfuerzos se dirigen en normalizar un Modelo de Objetos del Documento.

En el nuevo Modelo de objetos (DOM) cada elemento en un documento HTML es parte de un árbol, y se puede acceder a todos los elementos navegando a través de las ramas hasta alcanzar al "nodo" u objeto que se busca.

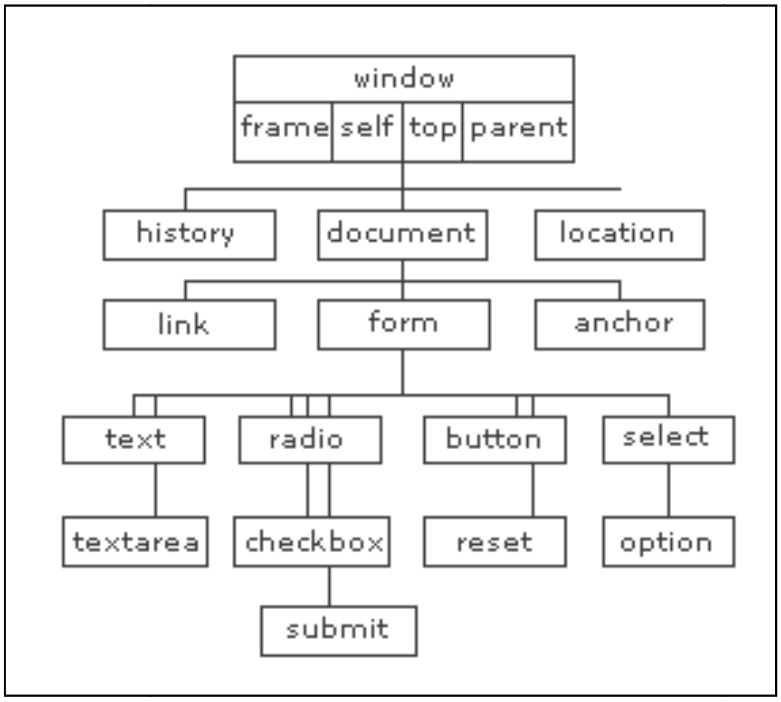


**Figura 3.5:** Jerarquía de Objetos[26]

### 3.2.8. Tipos de nodos

Los documentos XML y HTML tratados por DOM se convierten en una jerarquía de nodos. Los nodos que representan los documentos pueden ser de diferentes tipos. A continuación se detallan los tipos más importantes:

[26] Fuente: http://personal5.iddeo.es/romeroa/cursohtml/images/dom.gif

- Document: Es el nodo raíz de todos los documentos HTML y XML. Todos los demás nodos derivan de él.
- DocumentType: Es el nodo que contiene la representación del DTD empleado en la página (indicado mediante el DOCTYPE).
- Element: Representa el contenido definido por un par de etiquetas de apertura y cierre (<etiqueta>...</etiqueta>) o de una etiqueta abreviada que se abre y se cierra a la vez (<etiqueta/>). Es el único nodo que puede tener tanto nodos hijos como atributos.
- Attr: Representa el par nombre-de-atributo/valor.
- Text: Almacena el contenido del texto que se encuentra entre una etiqueta de apertura y una de cierre. También almacena el contenido de una sección de tipo CDATA.
- CDataSection: Es el nodo que representa una sección de tipo <CDATA>.
- Comment: Representa un comentario de XML.

### 3.2.9. Atributos

Además del tipo de etiqueta HTML y su contenido de texto, DOM permite el acceso directo a todos los atributos de cada etiqueta. Para ello, los nodos de tipo "*Element*" contienen la propiedad "*attributes*", que permite acceder a todos los atributos de cada elemento.

### 3.2.10. Eventos

En la programación tradicional, las aplicaciones se ejecutan secuencialmente de principio a fin para producir sus resultados. Sin embargo, en

la actualidad el modelo predominante es el de la programación basada en eventos.

Los scripts y programas esperan sin realizar ninguna tarea hasta que se produzca un evento. Una vez producido, ejecutan alguna tarea asociada a la aparición de ese evento y cuando concluye, el script o programa vuelve al estado de espera.

JavaScript permite realizar scripts con ambos métodos de programación: secuencial y basada en eventos. Los eventos de JavaScript permiten la interacción entre las aplicaciones JavaScript y los usuarios. Cada vez que se pulsa un botón, se produce un evento. Cada vez que se pulsa una tecla, también se produce un evento.

No obstante, para que se produzca un evento no es obligatorio que intervenga el usuario, ya que por ejemplo, cada vez que se carga una página, también se produce un evento. Por este motivo, muchas de las propiedades y métodos actuales relacionados con los eventos son incompatibles con los de DOM. De hecho, navegadores como Internet Explorer tratan los eventos siguiendo su propio modelo incompatible con el estándar.

### 3.2.11. Tipos de Eventos

Cada elemento XHTML tiene definida su propia lista de posibles eventos que se le pueden asignar. Un mismo tipo de evento (por ejemplo, dar click sobre

el botón izquierdo del ratón) puede estar definido para varios elementos XHTML y un mismo elemento XHTML puede tener asociados diferentes eventos.

El nombre de los eventos se construye mediante el prefijo “*on*”, seguido del nombre en inglés de la acción asociada al evento. Así, el evento de dar click en un elemento con el ratón se denomina “*onclick*” y el evento asociado a la acción de mover el ratón se denomina “*onmousemove*”.

### 3.2.12. Manejadores de eventos

En la programación orientada a eventos, las aplicaciones esperan a que se produzcan los eventos. Una vez que se produce un evento, la aplicación responde ejecutando cierto código especialmente preparado. Este tipo de código se denomina "manejadores de eventos" ("event handlers") y las funciones externas que se definen para responder a los eventos se suelen denominar "funciones manejadoras".

### 3.2.13. Frameworks y librerías

Para el desarrollo de RIAs con AJAX han surgido librerías y frameworks. Utilizando estas librerías, se reduce el tiempo de desarrollo y se tiene la seguridad de que las aplicaciones funcionan igual de bien en cualquiera de los navegadores más populares:

- Rich Web Client SDK: Este framework está disponible para ser usado con la plataforma .Net, genera automáticamente objetos de las clases .Net

existentes, soporta Arraylist y HashTables en el lado del cliente, los objetos de ambos lados son intercambiables

- AJAXAnywhere: Es una forma de ampliar las aplicaciones existentes basadas en JSP27/Struts/Spring/JSF con AJAX. Se usa principalmente para refrescar zonas de una página web, no se requiere hacer cambios de código ya que simplemente enlaza la aplicación con AJAX. A diferencia de otros frameworks, no esta basada en componentes.
- Prototype: Es un framework que facilita el desarrollo de aplicaciones web con JavaScript y AJAX. A pesar de que incluye decenas de utilidades, la librería es compacta y está programada de forma muy eficiente. Prototype se ha convertido en poco tiempo en una referencia básica de AJAX y es la base de muchos otros frameworks y librerías relacionadas. Las primeras versiones de Prototype no incluían ningún tipo de documentación, lo que dificultaba su uso y provocaba que la mayoría de usuarios desconocieran su verdadero potencial. Las versiones más recientes del framework disponen de una completa documentación de todas las funciones y métodos que componen su API. La documentación incluye la definición completa de cada método, sus atributos y varios ejemplos de uso.
- SAJAX Simple AJAX Toolkit: Es una herramienta de código abierto diseñada para ayudar a los sitios web que usen AJAX. Permite al programador llamar a funciones PHP, Perl o Python desde su página web por medio de JavaScript sin necesidad de forzar una actualización de la página en el navegador.

[27] JSF: Java Sever Pages

### 3.2.14. Desventajas de AJAX

- El desarrollo de las páginas con AJAX es más complejo que el de las páginas estáticas.
- Usa más recursos en el servidor.
- Los motores de búsquedas no entienden JavaScript.
- La información en la página dinámica no se almacena en los registros del buscador.
- Hay problemas usando AJAX entre nombres de dominios. Eso es una función de seguridad.
- Es posible que páginas con AJAX no puedan funcionar en teléfonos móviles, PDA[28] u otros aparatos.
- AJAX no es compatible con todo el software para ciegos u otras discapacidades.
- Las páginas creadas dinámicamente mediante peticiones sucesivas AJAX, no son registradas de forma automática en el historial del navegador, así que haciendo clic en el botón de "volver" del navegador, el usuario no será devuelto a un estado anterior de la página, en cambio puede volver a la última página que visitó.

## 3.3. Plataforma Java Edición Empresarial (JEE)

### 3.3.1. Introducción

JEE son las siglas de Java Enterprise Edition que es la edición empresarial de la plataforma Java creada y distribuida por Sun Microsystems.

[28] PDA: Personal Digital Assistant

JEE comprende un conjunto de especificaciones y funcionalidades orientadas al desarrollo de aplicaciones empresariales distribuidas.

La plataforma Java Enterprise Edition (JEE) es fruto de la colaboración de SUN con los líderes del sector del software empresarial (IBM, Apple, Bea Systems, Oracle, Hewlett-Packard, Novell) para definir una plataforma robusta y flexible orientada a cubrir las necesidades empresariales en e-business y business-to-business. JEE cumple con los siguientes objetivos de servicio:

- **Alta disponibilidad**, para percibir las necesidades de hoy, en un ambiente global de negocios.
- **Seguridad**, para proteger la privacidad de los usuarios y la integridad de la información de la empresa.
- **Fiabilidad y escalabilidad**, para asegurar que las transacciones del negocio sean procesadas prontamente y con precisión.

La plataforma JEE es entregada a los desarrolladores con los siguientes elementos:

- JEE BluePrint: Un modelo de aplicación estándar para desarrollo multi-capa, y servicios de cliente liviano.
- JEE Compatiility Test Suite: Una colección de pruebas de compatibilidad para verificar que un producto JEE cumple con el estándar de la plataforma JEE.
- JEE Platform: Una plataforma estándar para albergar las aplicaciones JEE.

- JEE Reference Implementation: Una aplicación de referencia para demostrar las capacidades de JEE y proporcionar una definición operacional de la plataforma JEE. La implementación de referencia constituye un servidor de aplicaciones JEE, no recomendado para ambientes de producción.

### 3.3.2. Arquitectura de JEE

JEE está basado en la arquitectura del lado del servidor (Served-based). Este tipo de arquitectura concentra la mayoría de los procesos de la aplicación en el servidor o en un pedazo de este. Este tipo de arquitectura tiene dos ventajas críticas en comparación con los otros tipos, estos son:

- Múltiples Clientes: Una arquitectura basada en el servidor requiere una clara separación entre la capa cliente (interfaz) y la capa servidor, en la cual se realizan los procesos de la aplicación. Esto permite que una simple aplicación soporte simultáneamente clientes con distintos tipos de interfaces, incluyendo poderosas interfaces (gráficas) para equipos corporativos, interfaces multimedia interactivas para usuarios con conexiones de alta velocidad, interfaces eficientes basadas en texto para usuarios con conexiones de baja velocidad.
- Operaciones robustas: Una arquitectura basada en el servidor soporta escalabilidad, confiabilidad, disponibilidad y recuperabilidad. Aplicaciones basadas en el servidor pueden ser divididas y distribuidas en múltiples procesadores. Componentes de la aplicación pueden ser replicados para dar soporte a caídas instantáneamente.

La plataforma de JEE provee un conjunto de APIs de Java y servicios necesarios para el soporte de aplicaciones para empresas.

La plataforma completa puede ser implementada en un solo sistema, o la plataforma de servicios puede ser distribuida a través de varios sistemas, pero todas las APIs especificadas deben ser incluidas en alguna parte del sistema completo.

El ambiente de ejecución (runtime) de JEE consta de las siguientes partes:

#### 3.3.2.1. Componentes de la aplicación

El modelo de programación de JEE define cuatro tipos de componentes de la aplicación que un producto JEE debe soportar:

1. Clientes de la Aplicación, son programas creados en Java que son generalmente programas GUI[29], que se ejecutan sobre una computadora de escritorio. La aplicación cliente ofrece a un usuario la experimenta similar al de las aplicaciones nativas, y tiene acceso a todo de los medios de la capa media de las aplicaciones JEE.

2. Applets, son componentes GUI que generalmente procesan un programa en un navegador web, pero pueden procesar una variedad de otras aplicaciones o dispositivos que soportan el modelo de programación del applet. Los Applets pueden ser usados para proporcionar una poderosa interfaz de usuario para las aplicaciones JEE. (Simples páginas HTML se pueden usar

[29] GUI: Graphical User Interface

también para proporcionar una interfaz de usuario más limitada para aplicaciones JEE).

3. Servlets, páginas Java Server Pages y páginas Java Server Faces generalmente procesan un programa en un servidor Web y responden a las peticiones HTTP de los clientes Web. Los Servlets, páginas JSP y páginas JSF pueden ser utilizados para que generen páginas HTML que son aplicación de interfaz de usuario. Pueden también ser usadas para generar XML u otro formato de datos que son consumidos por otros componentes de la aplicación. Los Servlets, y las páginas creadas con la tecnología JavaServer Pages o Java Server Faces, se refieren conjuntamente a menudo en ésta especificación como "Web components", componentes web. Las aplicaciones Web están compuestas de componentes Web y otros datos tal como las páginas HTML.

4. Enterprise JavaBeans (EJB), procesan en un ambiente controlado las transacciones soportadas. Los Enterprise Java Beans generalmente contienen la lógica del negocio y persistencia de una aplicación JEE.

Estos componentes de la aplicación se pueden dividir en tres categorías:

1. Componentes que se despliegan, manejan, y se ejecutan sobre un servidor JEE. Estos componentes incluyen JavaServer Pages, Servlets, Java Server Faces y Enterprise JavaBeans.

2. Componentes que se despliegan y manejan en un servidor JEE, pero está cargado y se ejecuta en una máquina cliente. Estos componentes incluyen páginas HTML y Applets incluidas en las páginas HTML.

3. Componentes cuyo despliegue y manejo no se definió completamente en esta especificación. Aplicaciones de los clientes caen en esta categoría. Futuras versiones de esta especificación pueden definir completamente el manejo de las aplicaciones del cliente.

#### 3.3.2.2. Contenedores

Los contenedores proporcionan el soporte para los componentes de la aplicación. Un contenedor proporciona una vista del subyacente JEE-APIs de los componentes de la aplicación.

Interponer un contenedor entre el componente de la aplicación y el servicio JEE permite a los contenedores inyectar transparentemente servicios definidos por los componentes, tal como el manejo de transaccionabilidad, chequeos de seguridad.

La especificación define un conjunto de servicios estándar para cada producto JEE, a continuación se listan estos servicios:

- JDBC[30] es el API para acceso a Bases de Datos Relacionales desde Java.
- Java Transaction API (JTA) es el API para manejo de transacciones a través de sistemas heterogéneos.
- Java Naming and Directory Interface (JNDI) es el API para acceso a servicios de nombres y directorios.

[30] JDBC: Java Database Connectivity

- Java Message Service (JMS) es el API para el envío y recepción de mensajes por medio de sistemas de mensajería empresarial como IBM MQ Series.
- JavaMail es el API para envío y recepción de email.
- Java IDL es el API para llamar a servicios CORBA[31].

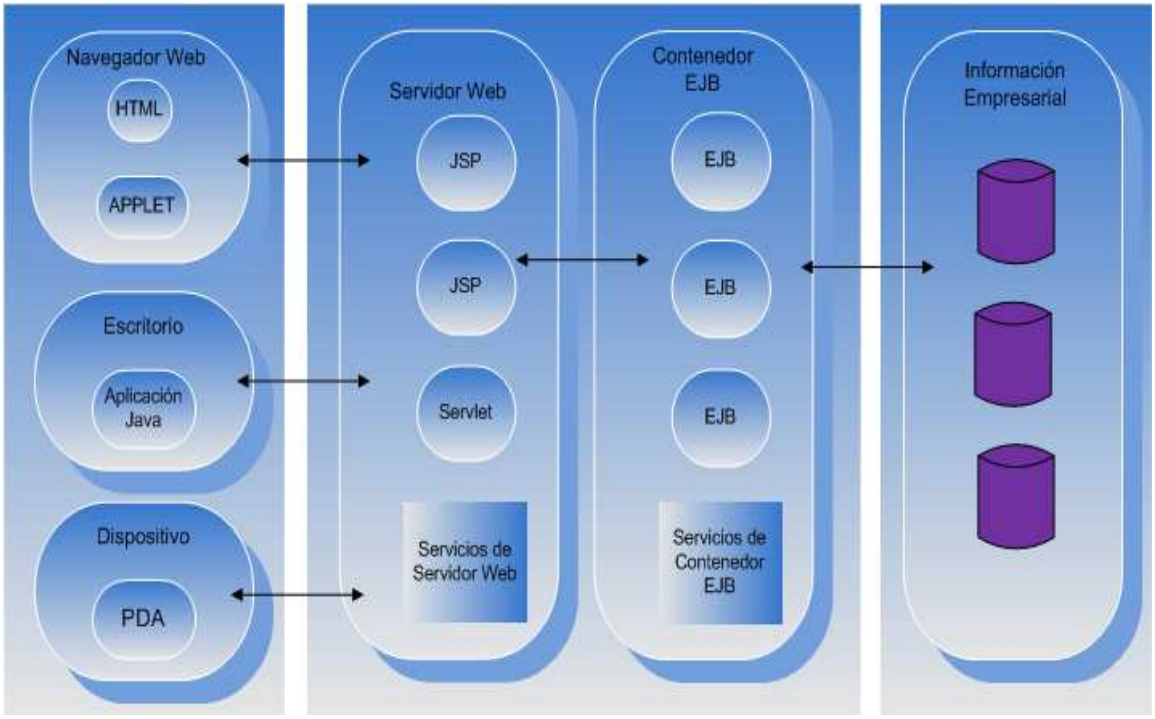


**Figura 3.6:** Arquitectura de JEE[32]

[31] CORBA: Common Object Request Broker Architecture
[32] Fuente: http://www.proactiva-calidad.com/java/arquitectura/imagenes/arquitectura_J2EE.JPG

## 3.4. JavaServer Faces (JSF)

### 3.4.1. Introducción

Java Server Faces es una especificación de desarrollo basado en el patrón MVC (Modelo Vista Controlador).

Uno de los patrones más conocidos en el desarrollo web es el patrón MVC (Modelo Vista Controlador). Este patrón nos permite separar la lógica de control de la lógica de negocio y la lógica de presentación.

Al igual que Struts, JSF pretende normalizar y estandarizar el desarrollo de aplicaciones web. Hay que tener en cuenta que JSF es posterior a Struts, y por lo tanto se ha nutrido de la experiencia de este, mejorando algunas sus deficiencias. De hecho el creador de Struts (Craig R. McClanahan) también es líder de la especificación de JSF.

**Tabla 3.1:** Generalidades Tecnología JavaServer Faces

| **JavaServer Faces** | |
|---|---|
| **Desarrollador** | Sun Microsystems |
| **Última versión estable** | 2.0<br>18 de agosto de 2009 |
| **Escrito en** | Java |
| **Sistema Operativo** | Máquina virtual Java |
| **Sitio web** | http://java.sun.com/ |

### 3.4.2. Objetivos de desarrollo en JSF

Estos objetivos de diseño representan la clave de desarrollo en JSF:

- Definir un conjunto simple de clases base de Java para componentes de la interfaz de usuario, estado de los componentes y eventos de entrada. Estas clases tratarán los aspectos del ciclo de vida de la interfaz de usuario, controlando el estado de un componente durante el ciclo de vida de su página.
- Proporcionar un conjunto de componentes para la interfaz de usuario, incluyendo los elementos estándares de HTML para representar un formulario.
- Proporcionar un modelo de JavaBeans para enviar eventos desde los controles de la interfaz de usuario del lado del cliente a la aplicación del servidor.
- Definir APIs para la validación de entrada, incluyendo soporte para la validación en el lado del cliente.
- Automatizar la generación de salidas apropiadas para el objetivo del cliente, teniendo en cuenta todos los datos de configuración disponibles del cliente, como versión del navegador.

### 3.4.3. Definición

JSF es una especificación para el desarrollo de aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones JEE.

JSF nos ofrece un marco de trabajo que facilita el desarrollo de aplicaciones, separando las diferentes capas de una arquitectura: presentación, reglas y entidades de negocio. En su mayor parte, una aplicación JSF es como cualquier otra aplicación web de Java. Una típica aplicación JSF incluye lo siguiente:

- Un API para representar componentes de interfaz de usuario y la gestión de su estado, manejo de eventos, validación en el servidor, la conversión de datos, la accesibilidad, y proporcionar extensibilidad para todas estas características.
- Un conjunto de páginas JSP (aunque no se limita al uso de páginas JSP como tecnología de presentación).
- Un conjunto de beans de respaldo, los cuales son JavaBeans que definen propiedades y funciones para los componentes de interfaz de usuario en una página.
- Un archivo de configuración de recursos de la aplicación, que define las reglas de navegación de la página y configura los beans y otros objetos como los componentes típicos.
- Posiblemente un conjunto de objetos personalizados creados por el desarrollador de la aplicación. Estos objetos pueden incluir componentes típicos, validadores, convertidores o detectores.
- Un conjunto de etiquetas para representar los típicos objetos en la página.
- Un modelo de eventos en el lado del servidor.
- Administración de estados.

### 3.4.4. Características

Este modelo de programación bien definido y la librería de etiquetas para componentes UI[33], facilitan la tarea de la construcción y mantenimiento de aplicaciones web con UIs del lado del servidor. Con un mínimo esfuerzo podemos:

- Arrastrar componentes sobre una página mediante la adición de etiquetas de componentes.
- Vincular a los componentes de la interfaz de usuario en una página para los datos del lado del servidor.
- Construir una interfaz de usuario con componentes reutilizables y extensibles.
- Guardar y restaurar el estado de la interfaz de usuario.

### 3.4.5. Interfaz de Usuario

La interfaz de usuario de la tecnología Java Server Faces (representado por MyUI en el gráfico) se ejecuta en el servidor y se presenta o se traduce para el cliente.

La página JSP (myform.jsp) es una página JSF, la cual es una página JSP que incluye etiquetas JSF. Estas expresan los componentes de la interfaz de usuario mediante el uso de etiquetas personalizadas por la tecnología JSF.

[33] UI: User Interface

La interfaz de usuario para la aplicación web (representada por myUI en el gráfico) administra los objetos referenciados por la página JSP.

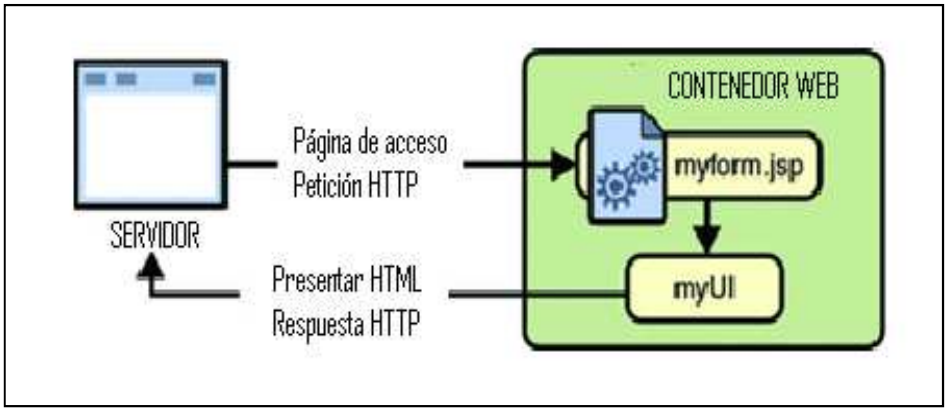


**Figura 3.7:** Interfaz de usuario Tecnología Java Server Faces[34]

### 3.4.6. Implementaciones de JSF

Actualmente existen muchas librerías de etiquetas JSF que pueden complementar a la implementación de la especificación oficial. A continuación se describen las principales implementaciones JSF.

#### 3.4.6.1. Icefaces

Icefaces proporciona un entorno de presentación web para aplicaciones JSF que mejora el framework JSF estándar y el ciclo de vida con características interactivas basadas en AJAX.

Para trabajar con Icefaces puede elegirse cualquiera de las dos implementaciones estándar.

[34] Fuente: http://docs.sun.com/source/819-3669/images/jsfIntro-server.gif

#### 3.4.6.2. Ajax4jsf

Fue desarrollado por java.net y patrocinado por Exadel. A partir del 5 de Marzo del 2007 Exadel y Red Hat decidieron colaborar en el desarrollo de esta y otras tecnologías. Actualmente el proyecto se encuentra dentro del proyecto RichFaces. Ajax4JSF es una extensión de código abierto para el estándar JSF que añade capacidades AJAX a las aplicaciones JSF sin la necesidad de escribir código JavaScript.

#### 3.4.6.3. RichFaces

RichFaces es un framework de código abierto que añade capacidad Ajax dentro de aplicaciones JSF existentes sin recurrir a JavaScript. RichFaces incluye ciclo de vida, validaciones, conversores y la gestión de recursos estáticos y dinámicos. Los componentes de RichFaces están construidos con soporte Ajax y un alto grado de personalización del “look-and-feel” que puede ser fácilmente incorporado dentro de las aplicaciones JSF.

#### 3.4.6.4. MyFaces Tomahawk

Este conjunto de componentes también es compatible con la implementación de Sun, así como con cualquier implementación compatible con JSF 1.1.

#### 3.4.6.5. AJAX Blueprints Components

Consiste en una serie de componentes AJAX basados en la tecnología JSF. Tiene dos librerías de componentes JSF. Una basada en la versión 1.2 de

JSF y que puede ser usada en un servidor de aplicaciones JEE5. El otro conjunto tiene componentes basados en JSF 1.1 y que pueden ejecutarse en servidores de aplicaciones J2EE 1.4.

### 3.4.7. Funcionamiento

Normalmente las aplicaciones web se construyen como un conjunto de pantallas con las que va interactuando el usuario. Estas pantallas contienen textos, botones, imágenes, tablas y elementos de selección que el usuario modifica. Todos estos elementos estarán agrupados en formularios HTML, que es la manera en que las páginas web envían la información introducida por el usuario al servidor.

La principal función del controlador JSF es asociar a las pantallas, clases Java que recogen la información introducida y que disponen de métodos que responden a las acciones del usuario. JSF nos resuelve de manera muy sencilla y automática muchas tareas como:

- Mostrar datos al usuario en cajas de texto y tablas.
- Recoger los datos introducidos por el usuario en los campos del formulario.
- Verificar el estado de los controles del formulario según el estado de la aplicación.
- Realizar validaciones y conversiones de datos introducidos por el usuario.
- Rellenar campos, listas, combos y otros elementos a medida que el usuario va interactuando con la pantalla.

- Verificar los eventos que ocurren en los controles (pulsaciones de teclas, botones y movimientos del ratón).

### 3.4.8. Ciclo de Vida JSF

Para entender el procesamiento de una página JSF hay que entender el ciclo de vida de la petición dentro del controlador JSF. Este ciclo de vida está compuesto de 6 fases en la versión 1.2 y son las siguientes:

- **Restaurar los componentes de la vista** (Restore View). En esta etapa el controlador construye en memoria la estructura de componentes de la página.
- **Aplicar los valores de la petición** (Apply Request Values). En esta etapa se recuperan los valores de la petición y se asignan a los objetos (beans) de la página.
- **Procesamiento de las validaciones** (Process Validations). Se verifican los parámetros de entrada según un conjunto de reglas definidas en un archivo de configuración.
- **Actualizar los valores del modelo** (Update Model Values). Los valores leídos y validados son cargados en los beans.
- **Invocación a la aplicación** (Invoke Application). Se ejecutan las acciones y eventos solicitados para la página. Si es necesario se realiza la navegación.
- **Generación de la página** (Render Response). En esta fase se genera la página que será enviada al usuario con todos sus elementos y valores actualizados.

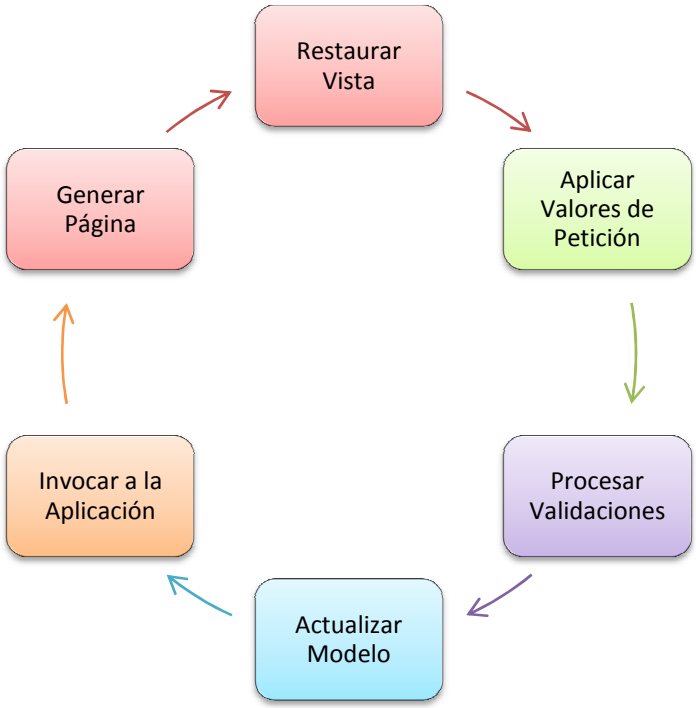


**Figura 3.8:** Ciclo de Vida de Java Server Faces

### 3.4.9. BACKING BEANS (Beans de Respaldo)

A las clases Java que se asocian a los formularios JSF se les denomina "backing beans" (beans de respaldo). Estos beans (objetos Java) se referencian en el archivo de configuración de JSF en la sección de "managed beans", ya que son objetos gestionados por el controlador JSF. Este se encarga de su construcción y destrucción automáticas cuando es necesario.

### 3.4.10. Estructura de las páginas

En su versión más sencilla, cada página JSF está formada por una página JSP que contiene un formulario (HTML FORM) y un "backing bean".

El controlador JSF registra en el servidor de aplicaciones un tipo especial de petición, típicamente *.jsf, que estará asociado a estas páginas.

El primer caso comienza cuando el usuario realiza en su navegador una petición de navegación a una Url de tipo *.jsf. Cuando al servidor web llega una petición del tipo página JSF, el controlador JSF entra en funcionamiento.

Primero comprueba si es la primera vez que se accede a dicha página. Si es así, carga la página JSP asociada pagina.jsp y la procesa construyendo en memoria la representación de los controles de la página. Tras esta etapa JSF sabe cómo construir el código HTML de salida y la lista de controles de usuario que la cumplen, es decir, sabe lo que contiene y cómo desplegarla.

El siguiente paso es asociarle los “backing beans”. Para ello, del procesamiento de la página JSP, el controlador ha obtenido la lista de backing beans asociados, por lo que procede a buscarlos en sus correspondientes ámbitos de la aplicación como petición y sesión. Los beans que no existan se crean llamando a los constructores de sus clases, definidos en la sección de managed beans del archivo de configuración de JSF.

El tercer paso es dar valores a las propiedades de los elementos JSF de la página. Aquí juega un papel fundamental el lenguaje de expresiones de JSF, que es parecido al lenguaje de expresiones que se permite en las páginas JSP normales.

Finalmente el servidor devuelve al usuario una página creada a partir de una página JSP que incluye normalmente etiquetas JSF, cuyos valores se extraerán del “backing bean” asociado, ahora ya actualizados.

### 3.4.11. La navegación entre páginas

Cuando se ejecuta una petición que incluye una acción se ejecuta el mecanismo de navegación de JSF. Tras la ejecución de la acción, el controlador determina cómo se debe mostrar al usuario el resultado de la petición. Existen varias posibilidades como:

- Finalizar la petición mostrando la página JSP que originó la petición, que es la opción por defecto.
- Mostrar otra página JSP diferente.
- Enviar al usuario una petición de redirección, por lo que el navegador del usuario se dirigirá automáticamente a otra página cuando reciba la respuesta a su petición.

Este mecanismo de navegación se implementa de manera sencilla en la página JSF. Cuando el controlador JSF llama al método asociado a la acción, este devuelve un valor de tipo String. Este valor es utilizado junto con las reglas de navegación creadas en el archivo de configuración de JSF para determinar la página que se debe enviar como respuesta al usuario.

Las reglas de navegación definen:

- La página de origen. Indica el JSP que originó la petición.
- La etiqueta de destino. Es la cadena que identifica el destino. Esta cadena es devuelta por el método del backbean que procesa la acción.
- La página de destino para cada etiqueta. Normalmente es el JSP el que procesará la petición de salida, utilizando los datos que hay en la petición y en la sesión.
- Si es un envío directo interno o una redirección externa. En el primer caso la respuesta se generará en la misma petición mediante una redirección interna a otro JSP o servlet. En el segundo caso se enviará como respuesta al navegador una instrucción de redirección para que el navegador realice una nueva petición de otra página.

Además las direcciones de origen admiten el * para que una misma regla sirva para múltiples páginas. También se pueden poner reglas por defecto que se aplican a todas las peticiones.

### 3.4.12. Ventajas de JSF

- Ofrece una clara separación entre el comportamiento y la presentación. Las aplicaciones Web construidas con tecnología JSP conseguían parcialmente esta separación. Sin embargo, una aplicación JSP no puede mapear peticiones HTTP al manejo de eventos específicos del componente o manejar elementos de Interfaz de Usuario como objetos con estado en el servidor.

- La separación de la lógica de la presentación también le permite a cada miembro del equipo de desarrollo de una aplicación Web enfocarse en su parte del proceso de desarrollo, y proporciona un sencillo modelo de programación para enlazar todas las piezas.
- Mejora los conceptos de componente-UI y capa-Web sin limitar a una tecnología de script particular o un lenguaje de marcas. Los APIs de la tecnología Java Server Faces se han creado directamente sobre el API Java Servlet. Esto permite: usar otra tecnología de presentación junto a JSP, crear componentes propios personalizados directamente desde las clases de componentes, y generar salida para diferentes dispositivos de cliente.
- Proporciona una rica arquitectura para la gestión del estado del componente, componente de procesamiento de datos, la validación de datos del usuario, y manejo de eventos.
- Ofrece una gran cantidad de componentes de licencia libre para las funcionalidades que se necesiten. Además, también existe una gran cantidad de herramientas para el desarrollo IDE[35] en JSF al ser el estándar de JAVA.

### 3.4.13. Desventajas de JSF

- Su naturaleza como estándar hace que la evolución de JSF no sea tan rápida como pueda ser la de otros entornos como WebWork, Wicket, Spring.

[35] IDE: Integrated Development Environment

- Debe ocurrir un mapeo de datos entre los dos modelos de objetos a fin de separar los objetos vista del modelo de objetos de dominio.

### 3.4.14. Versiones de la Especificación

Como versiones de la especificación Java Server Faces se tiene las siguientes:

**Tabla 3.2:** Versiones JavaServer Faces

| Versión | Fecha | Características |
|---|---|---|
| **JSF 1.0** | 11-03-2004 | Lanzamiento inicial de las especificaciones de JSF. |
| **JSF 1.1** | 27-05-2004 | Lanzamiento que solucionaba errores. Sin cambios en las especificaciones ni en el renderkit de HTML. |
| **JSF 1.2** | 11-05-2006 | Lenguaje de Expresión Unificado<br>Integración con JSTL[36]<br>Soporte Ajax |
| **JSF 2.0** | 12-08-2009 | Último lanzamiento.<br>Expansión del ciclo de vida para brindar soporte a peticiones Ajax.<br>Pretende mejorar la interoperabilidad entre librerías de diferentes proveedores. |

### 3.4.15. El Futuro de JSF

La especificación JSF forma parte importante del estándar JEE. De hecho se está preparando una nueva versión que traerá numerosas novedades, sobre

[36] JSTL: JavaServer Pages Standard Tag Library

todo en lo que se refiere a su integración con AJAX. También se está comenzando a utilizar en numerosas aplicaciones empresariales, ya que permite crear pantallas de usuario bastante complejas con una cierta facilidad. En la nueva versión se espera una mejora sobre el control de las fases del ciclo de vida de la petición que faciliten la creación de componentes JSF complejos que se usan de manera simple.

## 3.5. Icefaces

### 3.5.1. Introducción

Icefaces nace como una implementación de JSF, que surgió en el año 2004 como la oferta de Sun ante el despropósito de J2EE al programar aplicaciones web. Las dificultades de integración y las limitaciones para desarrollar en un “frontend” fueron las causas que motivaron a IceSoft desarrollar una librería de componentes ricos para JSF con un framework avanzado para la integración sencilla de las funcionalidades AJAX dentro del desarrollo de aplicaciones de negocio, con características similares a RichFaces.

### 3.5.2. Definición

Icefaces es un marco integrado AJAX de aplicaciones Java que permite a los desarrolladores de aplicaciones AJAX JEE, crear y desplegar aplicaciones RIA, en Java puro o ampliar las existentes sin costo alguno.

Icefaces aprovecha los estándares JEE para el desarrollo de aplicaciones, así como las herramientas y entornos de ejecución.

Es un framework de desarrollo web creado sobre la especificación JSF, con capacidad de procesamiento de solicitudes AJAX, que permite a los desarrolladores web construir aplicaciones con contenido enriquecido, programando únicamente en Java y sin tener que agregar un Applet u objetos que dependan de complementos propios del navegador que se esté usando.

Provee un ambiente de presentación amplio de Java Server Faces (JSF) que mejora la estructura de un JSF normal y el ciclo de desarrollo, basado en las características interactivas de AJAX. Reemplaza los servicios de JSF normal con servicios DOM, e incluye un puente de AJAX para entregar los cambios que presenta el navegador del cliente y comunicar los eventos de interacción de usuario al servidor de las aplicaciones JSF.

Proporciona una colección de componentes AJAX extensa que facilita el desarrollo rápido de aplicaciones basadas en características interactivas. Permite al programador incluir una serie de etiquetas AJAX en sus páginas JSP o XHTML de tal manera que el código AJAX es generado por el propio framework automáticamente.

### 3.5.3. Arquitectura

Los principales elementos de la arquitectura Icefaces incluyen:

- **Persistent Faces Servlet:** Las URLs con extensión ".iface" son mapeadas por el servlet “Persistent Faces Servlet”. Cuando se realiza una petición de

la página inicial en la aplicación, éste servlet se hace responsable de la ejecución del ciclo de vida JSF para la petición asociada.

- **Blocking Servlet:** Responsable de gestionar el bloqueo y desbloqueo de las peticiones después de iniciado un evento en la página.
- **D2D ViewHandler:** Responsable de establecer el Direct-to-DOM (D2D), incluyendo la inicialización de la 'DOM Respuesta-Escritura'. También invoca al Parser para analizar el árbol de componentes JSF en la página inicial.
- **El D2D Parser:** Responsable de agrupar el árbol de componentes de un documento JSP.
- **El D2D RenderKit:** Responsable de establecer un árbol del componente en el DOM vía el DOM Respuesta-Escritura durante un paso normal de un JSF.
- **DOM Response Writer:** Responsable de escribir en el DOM. También inicia la serialización del DOM para la primera prestación, y desbloquea el DOM Updater para actualizaciones de DOM incrementales.
- **DOM Updater:** Responsable de agrupar las mutaciones de DOM en una sola actualización de DOM incremental. DOM Updater bloquea el DOM incremental para actualizar las demandas hasta que estén completas.
- **Client-side AJAX Bridge**: Responsable de la actualización DOM en curso generada por la solicitud y la respuesta del proceso. También es el encargado de centrar la gestión y de presentar el proceso.
- **DOM Serializer:** Responsable de la fabricación en serie del DOM para la respuesta de la página.

- **Component Suite:** Proporciona componentes 'rich JSF' con influencia AJAX, dando los elementos básicos para aplicaciones Icefaces del lado del Cliente.

### 3.5.4. Características

- Es de código abierto y se basa en estándares.
- Extiende las funcionalidades de Java Server Faces.
- Desarrollo en Java puro**:** Permite desarrollar aplicaciones web en Java puro sin JavaScript.
- Integración de estructuras**:** Integra tres estructuras de frameworks y middleware.
- Actualización incremental de la página**:** Facilita la actualización de la página de manera incremental.
- Transparencia de la perspectiva de desarrollo**:** Estos rasgos de la presentación de Icefaces son completamente transparentes de la perspectiva de desarrollo de la aplicación.
- Actualización de la presentación de forma asíncrona: Las aplicaciones JSF normales pueden entregar sólo cambios de la presentación en la respuesta a un evento iniciado por el usuario. Icefaces introduce un mecanismo accionador que permite a la lógica de aplicación residente en el servidor realizar los cambios de presentación en el cliente, en respuesta a los cambios en el estado de la aplicación. Esto posibilita a los desarrolladores diseñar sistemas que entreguen datos al usuario en un tiempo real.

- Integración AJAX con JEE: Integra funcionalidad AJAX y permite a los desarrolladores JEE crear aplicaciones RIA (Rich Internet Applications) de una manera sencilla.
- Prescindir de Applets y Plugins: Las aplicaciones desarrolladas en Icefaces no necesitan plugins de navegador o applets para ser vistas.
- Aparta completamente al desarrollador de AJAX: No hacen falta etiquetas especiales, Icefaces se encarga de enviar sólo la información necesaria entre el cliente y servidor.
- Permite el uso de desarrollos basados en JavaScript: Icefaces, al estar basado en Java Server Faces (JSF), permite el desarrollo de aplicaciones JEE con la posibilidad de utilizar de forma fácil desarrollos basados en JavaScript.

### 3.5.5. Integración con Servidores de Aplicaciones JEE

- Apache Tomcat.
- BEA Weblogic Server.
- JBoss Application Server.
- IBM Websphere Application Server.
- Oracle Application Server Container for J2EE (OC4J).
- Sun GlassFish.

### 3.5.6. Integración con IDE´s

- Eclipse (Web Tools Platform + JSF tools).
- Sun NetBeans.

- MyEclipse Enterprise Workbench.
- Oracle JDeveloper 10g Studio Edition.

### 3.5.7. Compatibilidad con Portales y Frameworks JEE

- Liferay Portal 4.3.
- JBoss Seam 1.3.
- JavaServer Faces (JSF) 1.1, 1.2.
- Facelets.
- Spring Web Flow.

### 3.5.8. Navegadores Soportados

- Internet Explorer 6, 7.
- Firefox 1.x, 2.0.
- Mozilla 1.7.x y Netscape 7.x
- Safari 1.3
- Opera 9.x

### 3.5.9. Ventajas de Icefaces

- AJAX transparente**:** Iceface soporta múltiples desarrollos JSF.
- Experiencia de Usuario**:** Crea una experiencia superior en el usuario, que puede aprovechar las ventajas de las aplicaciones JEE, gracias a los componentes de Iceface.

- Código Abierto**:** Es un framework basado en AJAX con licencia libre. Iceface tiene actualmente una comunidad de alrededor de 20.000 desarrolladores en 36 países.
- Estandarización**:** Implementación basada en estándares Java hay una extensa variedad de plugins desarrollados para integrar Iceface con otros frameworks.
- Compatibilidad**:** Soporta todos los servidores de aplicaciones, aporta plugins para distintos IDE´s y efectos JavaScript de librerías de cualquier empresa que haya desarrollado AJAX.
- Seguridad**:** Es una de las soluciones AJAX más seguras del mercado. Previene los scripts de cruce de sitios, inyección de código malicioso. No utiliza datos de usuarios, previene fallos en los submits de los formularios y el ataque SQL[37] por inyección.
- Escalabilidad y clustering**:** El servidor asíncrono HTTP aporta una alta escalabilidad para aplicaciones Iceface y puede ser usado por un alto número de usuarios concurrentes.
- Carga incremental**:** Puede cargar páginas incrementalmente con edición de secciones y sin recargas de página completas.
- Aplicaciones en tiempo real**:** Recarga de páginas de modo asíncrono.
- Opciones de Soporte de niveles empresariales**:** Las suscripciones a IceSoft incluyen un soporte técnico profesional para el mantenimiento y producción de sistemas críticos con niveles de servicio garantizados, consultorías 24/7 y la administración de respuestas vía Iceface online.

[37] SQL: Structured Query Language

## 3.6. Richfaces

### 3.6.1. Historia

RichFaces se originó del sistema de Ajax4jsf que fue creado y diseñado por Alexander Smirnov. En el otoño del año 2005, Smirnov se unió a Exadel y continuó desarrollando el sistema. La primera versión de lo que se convertiría Ajax4jsf fue lanzado en marzo del 2006. Más tarde, en el mismo año, Exadel se separó y el framework Ajax4jsf y RichFaces nació.

Mientras que RichFaces provee componentes de caja blanca o también llamados componentes de enfoque centrado en Ajax (componentes que hacen todo lo que se necesite), Ajax4jsf provee soporte para páginas completas en Ajax. Como programador se debe especificar qué partes de la página deben ser procesadas en el servidor después de algunas acciones de los usuarios del lado del cliente y qué partes deben actualizarse tras el procesamiento.

Ajax4jsf se convirtió en un proyecto de código abierto alojado en Java.net, mientras que RichFaces se convirtió en una biblioteca de componentes JSF comerciales. En marzo del 2007 JBoss (en la actualidad una división de Red Hat) y Exadel firmaron un acuerdo de asociación en la que Ajax4jsf y RichFaces estaría ahora bajo el mando de JBoss y seria llamado JBoss RichFaces.

RichFaces pasó a ser de código abierto y libre. La unión de estas dos librerías de código abierto permitió la solución de algunos problemas de compatibilidad y versiones.

### 3.6.2. Arquitectura Richfaces

El framework está implementado como una biblioteca de componentes que agrega capacidad de Ajax en páginas existentes, por lo que no es necesario escribir ningún código JavaScript o sustituir componentes existentes a los nuevos "Ajax widgets".

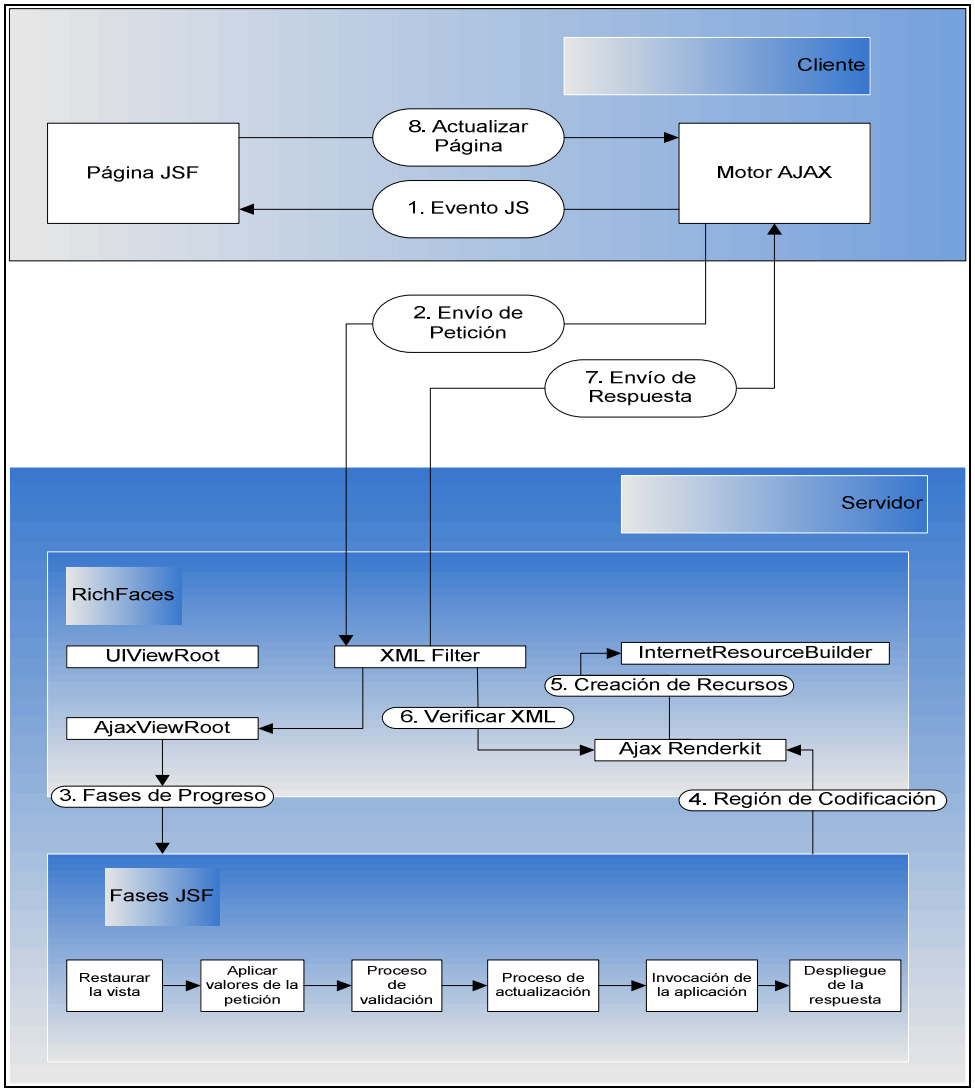


**Figura 3.9:** Flujo de Solicitud de Procesamiento [38]

[38] Fuente:http://www.jboss.org/fileaccess/default/members/jbossrichfaces/development/images/newpic1.png

Los elementos que componen RichFaces son:

- Componentes de filtro de AJAX
- Componentes de acción de AJAX
- Contenedores AJAX
- Un motor de JavaScript

Elementos importantes del framework RichFaces.

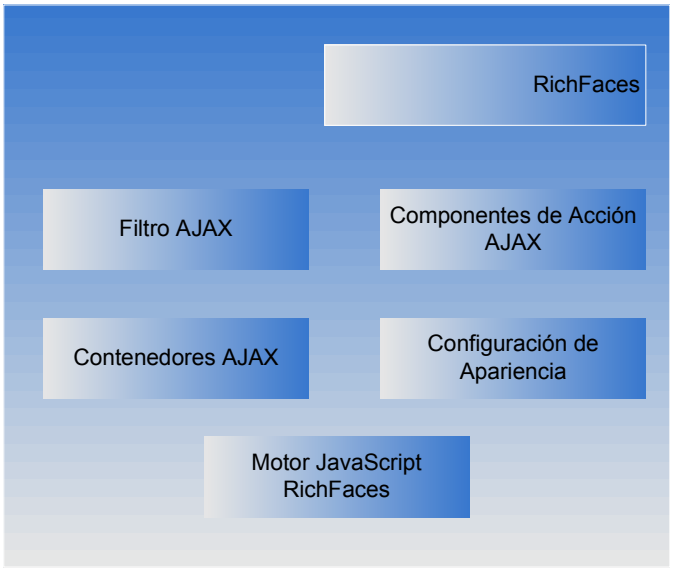


**Figura 3.10:** Estructura de componentes Core Ajax[39]

- Filtro Ajax: Para obtener todos los beneficios de RichFaces, se debe registrar un filtro en el archivo de configuración de la aplicación web. Este filtro reconoce múltiples tipos de peticiones. El diagrama de secuencia de la siguiente figura muestra la diferencia en el tratamiento de una página JSF regular y una solicitud Ajax. En el primer caso todo el árbol completo de

[39] Fuente: http://www.jboss.org/file-access/default/members/jbossrichfaces/html/images/newpic2.png

JSF será codificado, mientras que en la segunda opción depende del tamaño de la región Ajax. Como se puede ver, en el segundo caso el filtro analiza el contenido de la respuesta Ajax antes de enviarlo al cliente.

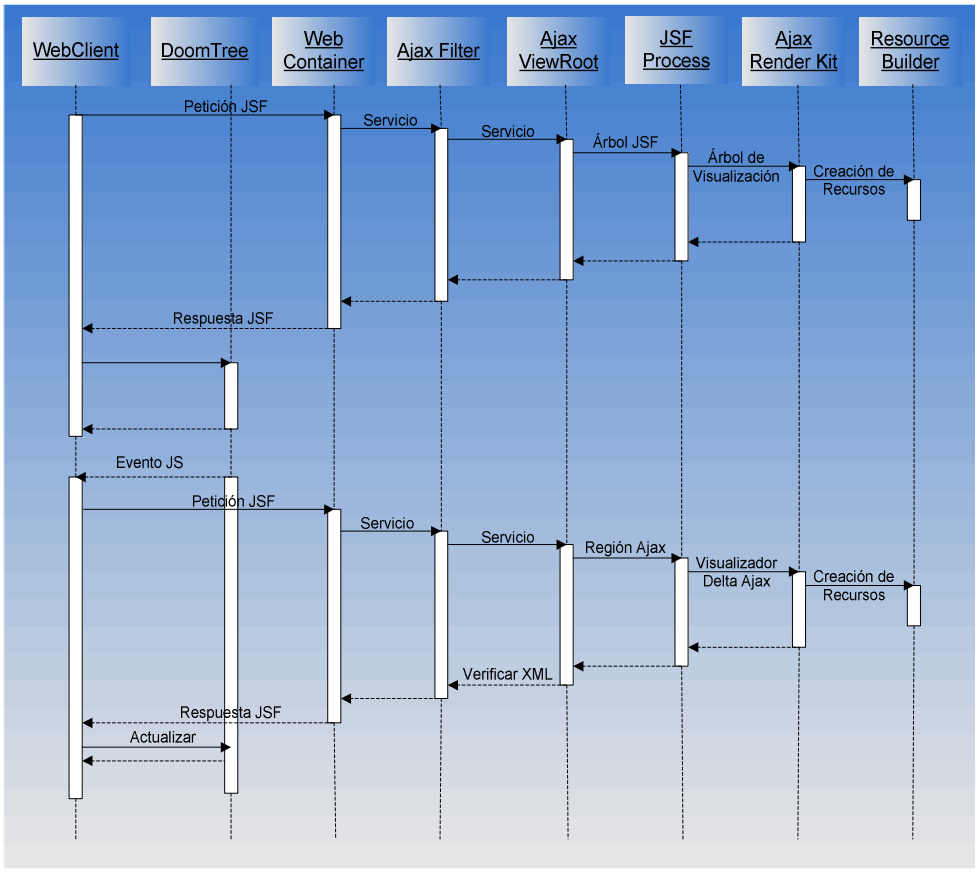


**Figura 3.11:** Diagrama de secuencia de una página JSF regular y una solicitud Ajax[40]

En ambos casos, la información necesaria sobre recursos estáticos o dinámicos que pide la aplicación será registrada por la clase "*ResourceBuilder*". Cuando llega una petición de un recurso, el filtro de RichFaces comprueba la caché de recursos para ese recurso y si está, el recurso se envía al cliente.

[40] Fuente: http://www.juntadeandalucia.es/xwiki/bin/download/MADEJA/RichFaces/richfaces3.jpg

De lo contrario, el filtro busca el recurso dentro de los que estén registrados por el ResourceBuilder. Si el recurso está registrado, el filtro de RichFaces enviará una petición al ResourceBuilder para crear (entregar) el recurso. La figura siguiente muestra la forma de procesar la petición de un recurso.

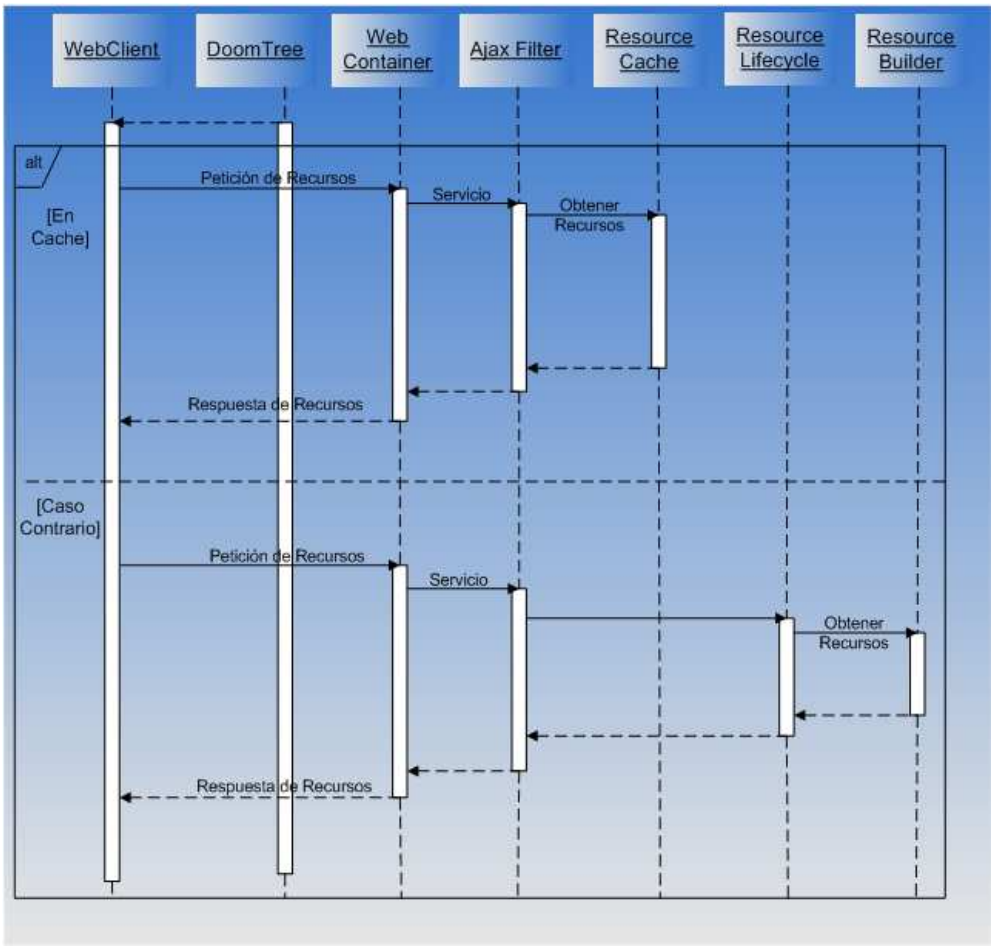


**Figura 3.12:** Petición de Recursos [41]

- Componentes de acción AJAX: Los componentes de acción más comunes que se utilizan en AJAX son: AjaxCommandButton, AjaxCommandLink,

[41] Fuente: http://www.juntadeandalucia.es/xwiki/bin/download/MADEJA/RichFaces/richfaces4.jpg

AjaxPoll, AjaxSupport. Pueden utilizar para enviar peticiones Ajax desde el cliente.

- Contenedor AJAX: El contenedor Ajax es una interfaz que describe un área de la página JSF que debería ser decodificada durante la solicitud Ajax. AjaxViewRoot y AjaxRegion son implementaciones de esta interfaz.
- Motor de JavaScript: RichFaces se ejecuta en el cliente. Sabe cómo actualizar las diferentes áreas en la página JSF basada en la información de la respuesta Ajax.

### 3.6.3. Características de Richfaces

- Se integra perfectamente en el ciclo de vida de JSF.
- Incluye funcionalidades Ajax, de modo que nunca se ve el JavaScript y tiene un contenedor Ajax propio.
- Contiene un conjunto de componentes visuales, los más comunes para el desarrollo de una aplicación RIA, con un número bastante amplio que cubren casi todas las necesidades.
- Soporta Facelets.
- Soporta CSS, themes y skins.
- Es un proyecto de código abierto, activo y con una comunidad también activa.
- Posee una integración sencilla de capacidades de AJAX en aplicaciones de negocio.
- Dinámica de manipulación de recursos.

### 3.6.4. Temas (Themes)

Otra característica importante son los temas (themes). Cualquier número de temas (definidos por medio de un archivo de propiedades) pueden ser creados con diversos esquemas de color.

Cuando se fija un tema particular, el componente se referirá a ese tema y generará los colores y los estilos basados en ese tema. Esto significa que se puede cambiar fácilmente la apariencia y la sensación del uso simplemente cambiando a otro tema. RichFaces proporciona un sistema de temas predefinidos:

- Claro
- Clásico
- Laguna (new - RichFaces 3.2.1)
- GlassX (new - RichFaces 3.2.2)
- DarkX (new - RichFaces 3.2.2)

Cada componente tiene un archivo de XCSS[42] (un formato de archivo especial que combina la flexibilidad de XML y del CSS) que realiza el trazado de los selectores del CSS a las características de un desarrollo particular.

[42] XCSS: Extends Cascading Style Sheets

### 3.6.5. Versiones Soportadas Java

- JDK 1.5 o superior

### 3.6.6. Soporte JavaServer Faces Implementados y Frameworks

- Sun JSF-RI - 1.2_12
- MyFaces 1.2.5
- Facelets 1.1.1 - 1.2
- Seam 1.2. - 2.1.0

### 3.6.7. Servidores de Aplicaciones JEE Soportados

- Apache Tomcat 5.5 - 6.0
- BEA WebLogic 9.1 - 10.0
- Sun Application Server 9 (J2EE 1.5)
- Glassfish (J2EE 5)
- JBoss 4.2.x - 5
- Websphere 7.0. and higher

### 3.6.8. Navegadores Soportados

- Internet Explorer 6.0 - 8.0
- Firefox 2.0 - 3.0
- Opera 8.5 - 9.5
- Safari 3.0
- Google Chrome

## 3.7. Apache Myfaces

### 3.7.1. Introducción

Apache MyFaces es un proyecto de la Fundación Apache Software, y alberga varios sub proyectos relacionados con la tecnología Java Server Faces.

El proyecto Apache MyFaces provee:

- Una implementación Java Server Faces (MyFaces API, módulos de implementación MyFaces).
- Varias librerías de componentes que contienen widgets de interfaz de usuario para la construcción de aplicaciones web con JSF (como por ejemplo: MyFaces Tomahawk, MyFaces Trinidad, MyFaces Tobago).
- Paquetes de extensión para Java Server Faces (ejemplo: MyFaces Orchestra, MyFaces Extensions Validator).
- Módulos de integración para otras tecnologías y estándares (ejemplo: MyFaces Portlet Bridge para integración con el estándar portlet).

Este proyecto se divide en 5 componentes principales:

- **Core:** Es una implementación de JSF 1.1 y 1.2 y los componentes que se han especificado en el Java Specification Requests (JRS) 127 y 252 respectivamente.
- **Tomahawk:** Es un conjunto de componentes creados por el equipo de desarrollo de MyFaces y donados a Apache.

- **Trinidad:** Un conjunto de componentes JSF aportados por Oracle.
- **Tobago:** Un conjunto de componentes contribuidos a MyFaces por Atanion GmbH.
- **Orchestra:** Un framework usado para el control de la persistencia entre sesiones.

### 3.7.2. MyFaces Tomahawk

MyFaces proporciona una serie de componentes JSF que van más allá de la especificación JSF. Estos componentes son 100% compatibles con la especificación de Sun JSF 1.1 Implementación de Referencia (IR) o cualquier otra aplicación compatible con JSF 1.1. Los componentes personalizados también se pueden utilizar con la aplicación de Apache MyFaces JSF.

#### 3.7.2.1. Componentes Extendidos

Además de los componentes personalizados que no se encuentran en la especificación JSF, el paquete de componentes MyFaces también incluyen una versión "extendida" de algunos de los componentes predeterminados. Estos son básicamente los componentes que existen en la versión básica o librerías de etiquetas HTML, pero las funciones adicionales que han sido añadidas van más allá de la especificación.

**Ejemplo:**

- <t:inputText>: Este es similar al componente <h:inputText>, pero este proporciona atributos adicionales, como forceId. Cuando este atributo es verdadero el código HTML generado utilizará el id especificado por el

atributo *código* id */código* en lugar del que normalmente se generaba por la especificación JSF.

- Convertidores (Converters): MyFaces contiene varios objetos personalizados que no implementan la interfaz UIComponent. Algunos de estos incluyen los objetos que implementan la interfaz Converter.
- Soporte de Tiles: MyFaces proporciona una solución personalizada para permitir el uso de los tiles y JSF juntos.

### 3.7.2.2. Evolución

Entre las versiones de Myfaces Tomahawk se tiene las siguientes:

**Tabla 3.3:** Versiones de MyfacesTomahawk

| Fecha | Versión | Descripción |
|---|---|---|
| **9 de mayo 2006** | Tomahawk 1.1.2 | Primera versión de Tomahawk para ser compatible con el nuevo proyecto MyFaces Core.<br>Funciona con cualquier versión de MyFaces Core. |
| **15 de junio 2006** | Tomahawk 1.1.3 | Compatible con la Implementación de Referencia (IR),<br>Todos los vínculos de mando y los componentes que une ahora el uso de comandos debe ser incluida dentro de un elemento <h:form>. Esto significa que los componentes que hacen uso de los enlaces de comandos (como tree2) debe estar dentro de un <h:form>para funcionar correctamente. |
| **12 de junio 2007** | Tomahawk 1.1.6 | Corrige un grave error XSS (cross-site scripting) que existía en la anterior versión. |
| **16 de noviembre 2008** | Tomahawk 1.1.8 | Pone a HttpServletRequest en Portlets<br><br>Añade atributo conmutado a toggleGroup |

### 3.7.3. MyFaces Orchestra

Orchestra es una pequeña biblioteca que puede ser utilizada en aplicaciones web para incorporar las siguientes características:

- Provee de alcance a los componentes de respaldo (backing beans).
- Anotaciones para definición de transacciones declarativas.
- Un componente JSF tipo "Dynaform" que ayuda a crear formas para la edición de datos persistentes.

Juntas estas características facilitan el desarrollo de aplicaciones que realizan una gran cantidad de persistencia. En particular, el componente de Dynaform (que requiere otras partes de Orchestra) hace que sea fácil programar el ingreso de datos que están ligados a una Base de Datos relacional.

Orchestra actualmente soporta JSF1.1, JSF1.2 y JSF2.0, pero el soporte debería ser posible para otros frameworks de presentación web en el futuro. MyFaces Orchestra es un miembro relativamente nuevo de la familia del proyecto Apache MyFaces, sin embargo ha sido utilizado en varios proyectos del mundo real. Desde la versión 1.4, Orchestra es compatible con JDK 1.5.

#### 3.7.3.1. Estructura

El proyecto Apache MyFaces Orchestra contiene varios módulos:

- Sandbox : Un espacio para los componentes que aún no son APIs estables, o que dependen de los componentes inéditos de otros proyectos.

- Core: Es compatible con Java 1.5 y JSF 1.1
- Core12: Es compatible con Java 1.5 y JSF 1.2
- Core20: Es compatible con Java 1.5 y JSF 2.0

#### 3.7.3.2. Características Importantes

- Funciona con una sintaxis compatible con Java 1.5, y opcionalmente se puede utilizar anotaciones.
- Utiliza el poderoso mecanismo de configuración de Spring Framework, para administrar el ciclo de vida de los componentes de respaldo (backing beans). Esta característica le permite integrarse transparentemente a Spring cuando se utilizan conversaciones.
- MyFaces Orchestra es compatible con frameworks de persistencia como Hibernate y Toplink (y en general cualquier implementación de JPA). Sin embargo, cualquier framework de persistencia puede ser integrado a Orchestra.
- El API de Orchestra puede ser adaptado para utilizar otros frameworks web además de JSF.

#### 3.7.3.3. Limitaciones

- Las características de persistencia de Orchestra asumen que la capa de presentación tiene acceso a la Base de Datos, lo cual implica un fuerte acoplamiento entre capas, esto puede llegar a ser una mala práctica y limitante en el desarrollo de grandes aplicaciones empresariales, pero puede ser de gran ayuda en el desarrollo de pequeñas aplicaciones.

- Orchestra no es compatible con portlets.
- Orchestra no soporta "sesiones distribuidas", es decir, las configuraciones donde las sesiones HTTP serán almacenadas y enviadas a otras máquinas en un clúster.

#### 3.7.3.4. Versiones

Entre las versiones de Myfaces Tomahawk se tiene las siguientes:

**Tabla 3.4:** Versiones de Myfaces Orchesta

| Fecha | Versión |
|---|---|
| 11 de marzo 2008 | MyFaces Orquesta Core 1.1 |
| 26 de agosto 2008 | MyFaces Orquesta Core 1.2 |

### 3.7.4. Myfaces Tobago

El objetivo de Tobago es proporcionar un conjunto bien diseñado de componentes de interfaz de usuario basados en JSF y que puedan ejecutarse bajo la arquitectura de MyFaces. Tobago es más que una librería de etiquetas (tag library). Las siguientes características de Tobago lo hacen diferente de otros frameworks:

- El enfoque de Tobago es la creación de aplicaciones de negocios sin la necesidad de un diseño HTML.
- Los componentes de interfaz de usuario se abstraen de HTML y cualquier capa de información que no pertenece a la estructura general de la página. El formato de salida final es determinado por el cliente / agente de usuario.

- Un mecanismo de tematización hace que sea fácil cambiar la apariencia y proporcionar implementaciones especiales para ciertos navegadores. Una solución alternativa asegura que el código sea reutilizado para nuevos temas tanto como sea posible.
- Una capa de gestión se utiliza para organizar los componentes de forma automática. Esto significa, que no es necesaria la creación manual de tablas HTML o de otras estructuras.

#### 3.7.4.1. Evolución

Entre las versiones de Myfaces Tobago se tiene las siguientes:

**Tabla 3.5:** Versiones de Myfaces Tobago

| Fecha | Versión | Descripción |
|---|---|---|
| **16 de agosto 2009** | MyFaces Tobago 1.0.22 | Documentación y la aclaración del uso del atributo tabindex.<br><br>Las etiquetas de comandos tienen la posibilidad de vincularse a los recursos (HTML, JSP, páginas JSF) con respecto a la gestión de los recursos. |
| 12 de junio 2009 | MyFaces Tobago 1.0.21 | Restricción de longitud de entrada para textarea.<br><br>Mejoras de rendimiento para TobagoReponseWriter, XmlUtils, HtmlWriterUtil y ResponseWriterBuffer |
| **26 de agosto 2008** | MyFaces Tobago 1.0.18 | Adopción de medidas en el lado del servidor. |
| **23 de marzo 2006** | Tobago 1.0.7 | Proporciona un bien diseñado conjunto de componentes de interfaz de usuario basado en JSF y se ejecuta en MyFaces. |

### 3.7.5. MyFaces Trinidad

Trinidad se remonta al desarrollo de código Java desde principios del año 2001. Oracle se encaminó al desarrollo de un framework web basado en componentes llamado UIX el cual evolucionó en una librería de componentes básicos para aplicaciones JSF y posteriormente estos mismos en frameworks JSF. Hoy en día JSF es una de las tecnologías centrales cuando se trata de implementar aplicaciones web a mediana y gran escala.

En el año 2006 Oracle tomó la decisión de donar el framework originalmente conocido como Oracle ADF Faces a la empresa Apache Software Foundation.

El equipo Oracle y Apache se concentraron en intercambiar la fuente desde las dependencias para asegurar que el framework conste solamente de código fuente abierto. Como acotación el desarrollo de Oracle ADF Faces todavía continua, pero ahora es basado en Trinidad.

En Mayo 5 del 2007, Trinidad deja la incubadora de Apache y se une a MyFaces como un subproyecto. Apache MyFaces es un proyecto de Apache Software Foundation que es responsable de la implementación JSF con el mismo nombre que incluye un número de subproyectos tales como Trinidad.

Para finales de Junio, Trinidad ve su primer lanzamiento 1.0.1 soportando JSF 1.1. En Julio 5 Trinidad 1.2.1, la versión para JSF 1.2 fue lanzado.

Para finales del 2007 Trinidad empieza a ser una librería de componentes con un ciclo frecuente de lanzamientos, alrededor de cada dos meses como promedio.

#### 3.7.5.1. Descripción General

Trinidad no es solo una librería de componentes JSF. Esta provee al desarrollador de un moderno framework web que se enfoca en amplitud y una filosofía de diseño de software de mundo cerrado. Esto tiene dos ventajas generales:

- El usuario no necesita combinar varias librerías de componentes. Esto evita problemas de integración.
- Un alto grado de consistencia es recomendado. Como consecuencia, cosas similares son hechas de modos similares, atributos similares tienen nomenclatura consistente, efectos colaterales de otras fuentes de código son mínimos, el crecimiento del error es mínimo, entre las más importantes.

#### 3.7.5.2. Características de Trinidad

- Representación parcial de Página (PPR): La tecnología JSF-Ajax es parte de prácticamente todas las etiquetas de Trinidad.
- Nomenclatura de atributos consistente, incluyendo conjuntos de atributos periódicos por todo el universo de las etiquetas de Trinidad.

- Un amplio número de etiquetas JSF están disponibles como una versión de Trinidad con refinamientos enfocados en tecnologías específicas de Trinidad.
- Framework de Cuadros de Diálogo: El framework de Cuadros de Dialogo de Trinidad permite a la aplicación web trabajar con ventanas emergentes encajables sin restricción de contenido.
- Trinidad además viene con sus propios ámbitos, como *"pageFlowScope"* para soportar Cuadros de Diálogo y flujo de datos de las páginas.

#### 3.7.5.3. Evolución

Entre las versiones de Myfaces Trinidad se tiene las siguientes:

**Tabla 3.6:** Versiones de Myfaces Trinidad

| Fecha | Versión | Nuevas Características |
|---|---|---|
| 05/05/2007 | Lanzamiento como subproyecto de MyFaces 1.0 | Librería de componentes de código abierto JSF 1.1 |
| 07/05/2007 | Lanzamiento 1.2.1 (en paralelo) | Librería de componentes de código abierto JSF 1.2 |
| 09/06/2007 | Lanzamiento 1.2.2/1.0.2 | Adición de un convertidor de números de lado del cliente y mejora del rendimiento |
| 10/28/2007 | Lanzamiento 1.2.2/1.0.3 | Mejoras del Panel, refinamiento del método DOM, mejora de configuración, íconos, mejoras AJAX |
| 12/10/2007 | Lanzamiento 1.2.2/1.0.4 | Mejoras CSS y mejoras de internacionalización |
| 01/14/2008 | Lanzamiento 1.2.2/1.0.5 | Mejoras de clasificación, adición de soporte de conversión. |
| 02/11/2008 | Lanzamiento 1.2.2/1.0.6 | Mejoras para apariencia y prestaciones, soporte de ámbito de aplicación. |
| 05/21/2008 | Lanzamiento 1.2.2/1.0.8 | Mejoras de apariencia, mejoras de subida de archivos, diálogos automáticos, soporte de navegador, mejoras de tree table |
| 08/07/2008 | Lanzamiento 1.2.2/1.0.9 | Mejoras de gestión de cambios, mejoras de panel, actualización de demo y documentación, mejoras de convertidor y soporte de navegador. |

# CAPÍTULO IV

# Evaluación y Selección del Framework JSF con soporte AJAX

## 4.1. Introducción

En la actualidad son muchos los frameworks JSF que se pueden encontrar en el mercado, la mayoría de ellos son gratuitos, sin embargo otros son pagados y/o tienen dependencia con ciertas herramientas. Por estos motivos la selección de un framework JSF que se adapte a las necesidades de un arquitecto de software puede ser una tarea realmente compleja.

Para la definición de este tema se realizó una evaluación previa entre todos los frameworks JSF, seleccionando tres que serían el objeto del análisis, estudio y evaluación.

La evaluación previa precisó la revisión de pruebas y evaluaciones realizadas por varios expertos de la comunidad de desarrolladores JEE, este estudio preliminar nos permitió la selección de tres implementaciones JSF con soporte para AJAX las cuales han sido estudiadas a profundidad en el capítulo anterior.

En los siguientes apartados se explorará y profundizará estas evaluaciones del estudio preliminar, las cuales también han influido en la selección final del framework JSF con soporte para AJAX a ser implementado en el desarrollo del piloto de la aplicación Gestor Fiducia Fondos bajo la plataforma JEE.

## 4.2. Enfoques de Arquitecturales de los Frameworks JSF con soporte para AJAX

Al investigar sobre componentes JSF con soporte AJAX se pueden diferenciar claramente dos estructuras:

- Arquitectura Multi-etiqueta

  Algunos frameworks semi-comerciales o ex-comerciales como ICEfaces o Trinidad ofrecen un gran conjunto de componentes AJAX habilitados, cada uno con su propio conjunto de componentes de comportamiento específico lo cual ocasiona que estos presenten algunos errores de compatibilidad. Muchos de estos componentes soportan funcionalidades AJAX otros no la soportan, algunos componentes ofrecen características que van mucho más allá del estándar, por lo que estos componentes se vuelven difíciles de mantener ya que no cumplen a cabalidad con la especificación JSF y la evolución del conjunto de etiquetas estándar.

- Arquitectura de una etiqueta

  Este tipo de arquitectura se refiere a los frameworks que soportan la adición de una sola etiqueta para proporcionar soporte AJAX, la cual puede ser anidada dentro de otros componentes JSF agregando a los mismos funcionalidad AJAX. Normalmente ciertas funciones o eventos JavaScript del componente padre (como onClick o onChange) desencadenará una petición AJAX y la respuesta será usada para actualizar algunos elementos de la página HTML. Este tipo de arquitectura tiene una gran oportunidad de sobrevivir en el futuro desarrollo de la especificación JSF, debido a las

mejoras de las etiquetas estándar y los nuevos componentes especiales que pueden ser activados fácilmente por AJAX.

Otro enfoque correspondiente a la arquitectura de los frameworks JSF con soporte AJAX se refiere a la estrategia de presentación de las páginas JSF. La estrategia de presentación de las páginas JSF permite ejecutar una petición a través del ciclo de vida JSF y desplegar la respuesta completa. El despliegue puede referirse a una porción o a toda la página.

Con un componente de tipo “Listener” JSF puede detener el ciclo de vida de las páginas JSF antes del inicio del despliegue de las mismas permitiendo mostrar únicamente los componentes que necesitan ser actualizados o incrustados por AJAX. A este tipo se estrategia se la llama “Estrategia Parcial de Despliegue”, esta estrategia reduce la cantidad de código HTML que viaja entre el cliente y el servidor y reduce el tiempo consumido para el despliegue de los componentes.

### 4.3. Matriz Comparativa

Ya que la especificación JSF es utilizada en la actualidad para el desarrollo de la mayoría de aplicaciones empresariales bajo la plataforma JEE y debido al gran número de frameworks JSF que se pueden encontrar disponibles, muchos miembros de la comunidad de desarrolladores JEE se han preocupado por la selección del framework adecuado para el desarrollo de sus aplicaciones.

A continuación se incluye una matriz elaborada por el Dr. Thomas Latka y Juergen Kniephoff, quienes desde finales del año 2006 se han encargado de probar las características de los diferentes frameworks JSF, esta matriz es actualizada constantemente por sus autores, tomando en cuenta también la opinión de muchas personas quienes colaboran con sus comentarios y opiniones.

Esta matriz pretende mostrar en forma comparativa las diferentes características que cada uno de los frameworks JSF posee. Al haber seleccionado para este proyecto la evaluación de los frameworks RichFaces, MyFaces y IceFaces, la matriz que se expone a continuación únicamente hace referencia a los mismos, si se desea acceder a la matriz completa con las opiniones de los diferentes colaboradores la misma puede ser encontrada en la siguiente dirección web: http://www.jsfmatrix.net.

**Tabla 4.1:** Matriz Comparativa Frameworks JSF[43]

| | RICHFACES | ICEFACES | MYFACES | | |
|---|---|---|---|---|---|
| | | | Tomahawk | Trinidad | Tobago |
| **Tecnología** | JSF | JSF | JSF | JSF | JSF |
| **Página Principal** | www.jboss.org/richfaces | www.icefaces.org | myfaces.apache.org/tomahawk | myfaces.apache.org/trinidad | myfaces.apache.org/tobago |
| **Ejemplos de Uso** | livedemo.exadel.com/richfaces-demo/index.jsp | | www.irian.at/myfacesexamples/home.jsf | www.irian.at/trinidad-demo/faces/index.jspx | tobago.atanion.net/tobago-example-demo/ |
| **Cantidad y Calidad de Información** | ALTA | ALTA | MEDIA | ALTA | BAJA |
| **Aplicación de Ejemplo descargable** | X | X | X | X | X |
| **URL de Documentación** | www.jboss.org/richfaces | | myfaces.apache.org/tomahawk/ | | myfaces.apache.org/tobago/documentation.html |
| **URL de Foros de Discusión** | www.jboss.org/richfaces | | | www.nabble.com/Road-America-td181.html#a181 | |

[43] Fuente : http://www.jsfmatrix.net.

| | | | | | |
|---|---|---|---|---|---|
| **Accesos para Búsqueda de Empleos** | 1 | 4 | 2 | 2 | - |
| **Búsquedas en Google** | 250.000 | 53.100 | 81.300 | 46.700 | 17.800 |
| **Componentes** | | | | | |
| **DATATABLE** | X | X | X | X | X |
| **TREE** | X | X | X | X | X |
| **TREETABLE** | | | X | X | |
| **LIVESCROLLING** | X | X | | | |
| **TAB** | X | X | X | X | X |
| **MENU** | X | X | X | X | X |
| **HTMLEDITOR** | X | X | X | | |
| **INPLACEEDITOR** | X | | | | |
| **CALENDAR** | X | X | X | X | X |
| **CHART** | | X | | X | |
| **DUALLIST** | X | | | X | |
| **UPLOAD** | X | X | X | X | X |
| **PROGRESSBAR** | X | X | | X | X |
| **DRAGDROP** | X | X | | | |
| **AUTOCOMPLETE** | X | X | | | X |
| **POPUPDIALOG** | X | X | X | X | X |
| **MODAL DIALOG** | X | X | X | | |
| **GOOGLE MAPS** | X | X | | | |
| **Características Adicionales en Tablas de Datos** | | | | | |
| **EDICIÓN EN CELDAS** | X | X | | X | X |
| **ARRASTAR Y PEGAR COLUMNAS** | X | | | | |
| **OCULTAMIENTO DE COLUMNAS** | X | X | | | |
| **FILTROS** | X | | | | |
| **RE-DIMENSIÓN DE COLUMNAS** | X | X | | | X |
| **AGRUPAMIENTO** | X | X | | | |
| **SELECCIÓN DE FILAS** | X | X | | | |
| **PAGINACIÓN POR BASE DE DATOS** | X | | | | |
| **Árbol de components** | | | | | |
| **DRAG & DROP REORDERING** | | X | | | |
| **Componentes de Tipo Pestaña** | | | | | |
| **TAB CLOXNG** | X | X | | | |
| **TAB ADDING** | X | X | | | |
| **Estrategia de Actualización** | | | | | |
| **Página Completa** | X | | | | |
| **Basada en Servidor AJAX** | X | X | | X | X |
| **Sólo lado del Cliente** | | X | | | |
| **Compatibilidad** | | | | | |
| **Coexistencia con otros Frameworks JSF** | Seam, Spring, Tomahawk | Seam, Spring, Webflow, (Tomahawk) | Ajax4jsf, Jboss Richfaces | Orchesta | |
| **JSF 1.2** | X | X | | X | |
| **FACELETS** | X | X | X | X | X |
| **JAVA 5** | X | X | | X | X |
| **PORTLET** | X | X | | | |
| **JQUERY** | X | | | | X |
| **Soporte IDE** | | | | | |
| **ECLIPSE TAG SUPPORT** | X | X | +- | | |

| | | | | | |
|---|---|---|---|---|---|
| **ECLIPSE VISUAL DESIGN** | X | X | | +- | |
| **NETBEANS TAG SUPPORT** | X | X | | | |
| **Servidor** | | | | | |
| **IE6** | X | X | X | X | X |
| **IE7** | X | X | | X | |
| **FIREFOX 2** | X | X | X | X | X |
| **FIREFOX 3** | X | X | | X | |
| **SAFARI** | X | X | | | |
| **OPERA** | | X | | | |
| **KONQUEROR** | X | | | | |
| **Misceláneos** | | | | | |
| **SERVERPUSHING** | X | X | | | |
| **TOOLTIP** | X | X | | X | |
| **Soporte de botón secundario de Mouse** | X | X | | | |
| **SKINNING** | X | X | | X | X |
| **DESING** | X | X | X | +- | |
| **Características Especiales** | Multifile upload, Ajax Queue, Colorchooser, Google Maps, Virtual Earth, Layout | | Sandbox ofrece algunos componentes AJAX | inputNumberSpinbox, selectManyShuttle y un proceso de entrenamiento y validación del lado del cliente | |
| **Licenciamiento** | | | | | |
| **LICENCIA** | LIBRE | LIBRE | LIBRE | LIBRE | LIBRE |

## 4.4. Metodología de Evaluación

Para la evaluación de los frameworks JSF con soporte AJAX seleccionados se utilizó la metodología “Empírica – Analítica”, esta metodología es un modelo de investigación científica que se basa en la lógica empírica, esta lógica se fundamenta principalmente en la experiencia para la construcción del conocimiento.

El término empírico deriva del griego antiguo de experiencia (es decir, llevando a cabo el experimento). Los datos empíricos son sacados de las pruebas acertadas y los errores, es decir, de experiencia. Su aporte al proceso de investigación es resultado fundamentalmente de la experiencia. Estos métodos posibilitan revelar las relaciones esenciales y las características fundamentales

del objeto de estudio, accesibles a la detección sensoperceptual, a través de procedimientos prácticos con el objeto y diversos medios de estudio.

La aplicación de los métodos, técnicas y procedimientos empírico analíticos, representa un nivel en el proceso de investigación cuyo contenido procede fundamentalmente de la experiencia, pero donde está presente la dialéctica entre lo subjetivo y lo objetivo, lo cuantitativo y lo cuantitativo y lo empírico y lo teórico.

### 4.5. Criterios de Evaluación

La selección de los criterios para la evaluación de los frameworks JSF con soporte AJAX es un proceso complicado, ya que de esta selección inicial dependerán los resultados finales, y por ende la selección del framework JSF que sería utilizado en el desarrollo del producto Gestor Fiducia Fondos bajo la plataforma JEE.

Para la definición de los criterios de evaluación se procedió a realizar una ronda de reuniones con los principales ejecutivos de la empresa Gestor, gerentes de las unidades de Desarrollo de Software (Ing. Antonio Deidán), Sistemas y Proyectos Internos (Ing. Dennys Guzmán); y el Director de Proyectos bajo Plataforma JEE (Ing. Henry Coral). En estas reuniones se realizo una explicación sobre las principales características de los principales frameworks JSF, se obtuvo información de los encargados del proyecto sobre las características visuales mínimas que se esperaban del framework seleccionado, para finalmente y luego

de algunas discusiones técnicas en un proceso de “lluvia de ideas” llegar a definir 17 diferentes criterios de evaluación.

Para seguir un orden en el proceso de investigación y experimentación primero se procedió a agrupar a los diferentes criterios de evaluación, de esta forma se llego a definir 4 diferentes categorías de evaluación, las cuales se expone a continuación:

- **Presente y Futuro del Framework:** Esta categoría pretende evaluar el estado de madurez actual del framework y el futuro del mismo, para esto se han seleccionado los siguientes criterios de evaluación:
    - Popularidad del Framework.
    - Confianza en las empresas propietarias del Framework.
    - Regularidad en la liberación de parches y actualizaciones.
    - Tamaño de la comunidad involucrada.
- **Prestaciones Funcionales:** Esta categoría evalúa la cantidad y calidad de las funcionalidades que provee el framework, así como la estabilidad del mismo, para esto se han seleccionado los siguientes criterios:
    - Cumplimiento del estándar JSF.
    - Componentes y/o Características Adicionales.
    - Soporte de Internacionalización.
    - Estabilidad General de los Componentes del framework.
    - Soporte para la inclusión de Otras Tecnologías.
- **Facilidad de Uso:** Esta categoría evalúa la facilidad que tendría un equipo de desarrolladores JEE sin conocimientos previos de JSF para la utilización

e inclusión del framework en un proyecto real, para esto se han seleccionado los siguientes criterios de evaluación:

  - Curva de Aprendizaje.
  - Calidad de Documentación Oficial.
  - Calidad de Documentación Profesional.
  - Calidad de Documentación No Oficial Ni Profesional.
  - Integración con IDEs conocidas.

- **Costos de Licenciamiento y Soporte:** Esta categoría evalúa los costos de licenciamiento tanto para el desarrollo de aplicaciones como para la fase de producción de las mismas; así como el costo del soporte técnico especializado.

## 4.6. Proceso de Evaluación

El proceso de evaluación se realizó siguiendo la metodología descrita anteriormente, para esto se procederá a utilizar tres técnicas: investigación, estudio y experimentación con cada uno de los criterios de evaluación definidos en el punto anterior.

El proceso de investigación está directamente relacionado con la búsqueda de información de cada uno de los frameworks seleccionados en el Internet, la técnica de investigación fue utilizada para llegar a determinar conclusiones sobre dos de las categorías de evaluación (Presente y Futuro del Framework y Costos de Licenciamiento y Soporte).

Las técnicas de estudio y experimentación se utilizaron para determinar resultados sobre los criterios de evaluación que son agrupados por las categorías de Prestaciones Funcionales y Facilidad de Uso. El estudio y experimentación se realizó sobre cada una de las aplicaciones de ejemplo provistas por cada framework.

Las aplicaciones de ejemplo demuestran los usos y características de cada uno de los componentes que conforman el framework, es por esto que su estudio y experimentación fue fundamental en la obtención de resultados.

Una vez que se ha investigado, estudiado y experimentado con cada framework, se procedió a calificar objetivamente a cada uno de ellos en todos los criterios de evaluación definidos en el punto anterior; para esto se ha definido una escala de evaluación conformada de cuatro valores posibles, siendo el valor 4 el máximo a ser otorgado en el cumplimiento del criterio y 1 el valor mínimo. La escala se basa en 4 valores, para evitar la utilización reiterada de un valor intermedio, lo cual restaría objetividad a la evaluación.

## 4.7. Cuadros de Evaluación

A continuación se muestran cuatro diferentes cuadros de evaluación uno por cada categoría definida. Las calificaciones asignadas a cada framework en los diferentes criterios de evaluación, fueron revisadas, verificadas y aprobadas por el responsable de la dirección de este proyecto de tesis, Ing. Henry Coral.

**Tabla 4.2:** Presente y Futuro del Framework

| Criterio de Evaluación | RichFaces | IceFaces | MyFaces |
|---|---|---|---|
| Popularidad del Framework | 3 | 4 | 3 |
| Confianza en las empresas propietarias del Framework | 4 | 2 | 4 |
| Regularidad en la liberación de parches y actualizaciones. | 3 | 3 | 2 |
| Tamaño de la comunidad involucrada | 4 | 3 | 2 |
| **Total:** | 14 | 12 | 11 |
| **Calificación Promedio:** | 3.5 | 3 | 2.75 |

**Tabla 4.3:** Prestaciones Funcionales

| Criterio de Evaluación | RichFaces | IceFaces | MyFaces |
|---|---|---|---|
| Cumplimiento del estándar JSF | 4 | 4 | 4 |
| Componentes y/o Características Adicionales | 4 | 4 | 2 |
| Soporte de Internacionalización | 4 | 4 | 2 |
| Estabilidad General de los Componentes del framework | 3 | 3 | 2 |
| Soporte para la Inclusión de Otras Tecnologías | 4 | 2 | 2 |
| **Total:** | 19 | 17 | 12 |
| **Calificación Promedio:** | 3.8 | 3.4 | 2.4 |

**Tabla 4.4:** Facilidad de Uso

| Criterio de Evaluación | RichFaces | IceFaces | MyFaces |
|---|---|---|---|
| Curva de Aprendizaje. | 3 | 4 | 2 |
| Calidad de Documentación Oficial | 4 | 4 | 1 |
| Calidad de Documentación Profesional | 4 | 3 | 1 |
| Calidad de Documentación No Oficial Ni Profesional | 4 | 4 | 1 |
| Integración con IDEs conocidas | 3 | 2 | 1 |
| **Total:** | 18 | 17 | 6 |
| **Calificación Promedio:** | 3.6 | 3.4 | 1.2 |

**Tabla 4.5:** Costos de Licenciamiento y Soporte

| Criterio de Evaluación | RichFaces | IceFaces | MyFaces |
|---|---|---|---|
| Costos de licenciamiento para la fase de Desarrollo | 4 | 4 | 4 |
| Costos de licenciamiento para la fase de Producción | 4 | 4 | 4 |
| Costo de Soporte Técnico especializado | 2 | 3 | 1 |
| **Total:** | 10 | 11 | 9 |
| **Calificación Promedio:** | 3.3 | 3.7 | 3 |

## 4.8. Evaluación Final

La evaluación final comprende la selección del framework JSF con soporte AJAX que fue utilizado por la empresa GESTORINC S.A. para el desarrollo de su aplicación Gestor Fiducia Fondos JEE, además el framework seleccionado fue

utilizado en la implementación de un pequeño piloto como parte práctica de este proyecto en el siguiente capítulo.

Para la evaluación final, se realizó una encuesta a tres ejecutivos de la empresa GESTORINC S.A., pidiéndoles que de acuerdo a sus expectativas y experiencia asignen pesos porcentuales a cada una de las categorías definidas anteriormente. De estas encuestas se obtuvo los siguientes pesos para cada categoría:

- Presente y Futuro del Framework: 40%,
- Prestaciones Funcionales: 30%,
- Facilidad de Uso: 20%
- Costos de Licenciamiento y Soporte: 10%

Las encuestas realizadas con sus respuestas se adjuntan en el Anexo 1 de este proyecto de tesis. A continuación se muestra el cuadro de la evaluación final, en el que se puede observar los valores asignados a cada framework, según sus diferentes características.

**Tabla 4.6:** Evaluación final

| Categoría de Evaluación | Peso | RichFaces | IceFaces | MyFaces |
|---|---|---|---|---|
| **Presente y Futuro del Framework** | 40.00% | 35.00% | 30.00% | 27.50% |
| **Prestaciones Funcionales** | 30.00% | 28.50% | 25.50% | 18.00% |
| **Facilidad de Uso** | 20.00% | 18.00% | 17.00% | 6.00% |
| **Costos de Licenciamiento y Soporte** | 10.00% | 8.25% | 9.25% | 7.50% |
| **Puntuación Final:** | 100.00% | 89.75% | 81.75% | 59.00% |

De acuerdo a lo expuesto en la Tabla 5.6 se puede observar que el framework que cumple con la mayoría de expectativas de los ejecutivos de la empresa GESTORINC S.A. fue RichFaces de Jboss, el cual obtuvo la mayor calificación porcentual correspondiente al 89.75%.

# CAPÍTULO V

# IMPLEMENTACIÓN DEL PILOTO DEL MÓDULO DE SEGURIDADES DEL SISTEMA GESTOR FIDUCIA-FONDOS

## 5.1. Descripción de la Metodología de Gestor

Para el desarrollo de los proyectos de Software bajo la plataforma JEE, GESTORINC S.A. definió inicialmente una metodología de desarrollo de software, la cual estaba orientada a ir de la mano con la plataforma y tecnología del proyecto basada especialmente en la metodología “eXtreme Programming”.

Sin embargo con el pasar del tiempo y luego de algunas modificaciones esta metodología ha sufrido algunos cambios, los cuales han afectado al Proceso de Desarrollo de Software bajo la plataforma JEE.

Actualmente la metodología de Desarrollo de Software de GESTORINC S.A. se divide en las siguientes fases:

**Análisis Inicial:** Este proceso generalmente incluye la discusión de las funcionalidades esperadas a ser desarrolladas para un módulo o sub-módulo de acuerdo a la complejidad y extensión del mismo.

Este análisis es elaborado por un conjunto de expertos de la empresa, quienes poseen el conocimiento necesario para la definición de las funcionalidades y características que debe poseer cada módulo. Como resultado

de ese análisis inicial, se elabora un documento llamado “Modelo Conceptual” el cual contiene la descripción de lo que debe ser desarrollado.

**Modelo Físico:** En este paso del proceso se define la estructura física de Base de Datos que será utilizada para soportar todas las características descritas en el “Modelo Conceptual” esta fase es desarrollado generalmente por un experto en modelado de GESTORINC S.A., el experto del módulo que va a ser desarrollado y el arquitecto de Software del Proyecto.

Como resultado de esta etapa se obtiene el Diagrama Físico de Base de Datos.

**Especificación de Requerimientos:** El arquitecto de software del proyecto es el encargado de definir los requerimientos a ser desarrollados y los asigna a los desarrolladores quienes con la información del “Modelo Conceptual” y el “Modelo Físico” son los encargados de definir los Casos de Uso que serán implementados.

Una vez que la Definición de Casos de Uso, ha sido aprobada por el líder funcional del proyecto, los desarrolladores proceden a realizar la descripción detallada de los mismos, utilizando para ello un formato propio de la empresa. Como resultado de esta etapa se obtiene el “**Diagrama de Casos de Uso**”, y la “**Descripción Detallada de Casos de Uso**”

**Desarrollo:** Una vez que los requerimientos han sido aprobados e incluidos en la planificación, el desarrollador podrá iniciar con el desarrollo de los asignados a su persona. Como resultado de esta fase se obtiene una funcionalidad del sistema desarrollada y lista para pasar al Aseguramiento de Calidad.

**Pruebas:** Al finalizar el desarrollo de cada requerimiento el desarrollador deberá enviar al departamento de Calidad la o las órdenes que involucran el desarrollo del requerimiento para la realización de las respectivas pruebas y lograr de esta forma el aseguramiento de calidad del producto.

Como resultado de esta fase se obtiene la liberación de la funcionalidad desarrollado con la certificación de haber cumplido con todos los estándares, validaciones y requerimientos.

**Certificación:** Cuando todos los requerimientos de un módulo hayan sido liberados por Calidad, se procederá a una revisión completa del módulo para certificar la funcionalidad del mismo. Como resultado de esta fase se obtiene la certificación de funcionamiento del módulo desarrollado.

Para el desarrollo de este proyecto de tesis, y teniendo en cuenta uno de los objetivos del mismo que consiste en la implementación de un piloto del módulo de Seguridades del Sistema Gestor Fiducia Fondos no se tomará en cuenta las fases de Pruebas y Certificación.

## 5.2. Modelo Conceptual

**Registro de Modelo Conceptual.**

**Nro Planificación:** 3

**Fecha:** 30 de Noviembre del 2009 **Revisión Nro:** 1.0

**Producto: Gestor Fiducia Fondos JEE**

**Desarrollo: Módulo de Seguridades del Sistema Gestor JEE**

**Cliente: GESTORINC** **Líder de Producto:** Antonio Deidán

**Resumen:**

El presente documento describe la funcionalidad general con la cual deberá cumplir el módulo de seguridades del Producto Gestor JEE.

**Responsable:** Henry Coral

**Fecha Inicio:** **Fecha Final:** **Tiempo Real(Días):**

**Detalle Conceptual.**

**Especificaciones Funcionales de Requerimientos**

**Módulo de Seguridades de GESTORINC S.A.**

***I. Administración de Usuarios y Contraseñas.***

a. Gestión de los Usuarios del Sistema.- El sistema permitirá el Ingreso, Modificación y Suspensión de los Usuarios.

b. Generación de Contraseñas.- Al inicio, el sistema generará contraseñas aleatorias y diferentes, las cuales deben cumplir con los requisitos especificados en la sección de contraseñas; además, se podrá generar contraseñas bajo demanda. No deberá existir una contraseña por defecto.

c. Distribución de las Contraseñas.- Una vez creado el usuario, se realizará de manera separada la comunicación del identificador asociado y su contraseña, garantizando en todo el proceso la confidencialidad e integridad de esta información; es decir, se deberá generar una comunicación vía correo electrónico para el envío del Identificador del usuario y otra para la contraseña.

d. Estado del Usuario.- El usuario podrá tener uno de los siguientes 4 estados: activo, bloqueado, suspendido y eliminado, el administrador, podrá hacer de manera amigable cambios de estado.

1. **Activo**.- El usuario está vigente en el sistema.
2. **Bloqueado**.- La cuenta ha sido bloqueada por inicio de sesiones fallidas y/o acceso a funcionalidades no permitidas, se mantienen sus funcionalidades asignadas.
   El usuario intenta acceder a opciones de menús o modificar campos no autorizados la operación invalida es registrada y pudiese existir sanción de bloqueo de cuenta.

3. **Suspendido**.- La cuenta es válida pero no puede ingresar al sistema por razones operativas, se mantienen sus funcionalidades asignadas. Ejemplo el usuario salió de vacaciones.
4. **Eliminado**.- La cuenta no es válida dentro del sistema, todas las funcionalidades asociadas en los perfiles son eliminadas.
   Eliminado significa que no se lo puede ver en los reportes, no puede ser accesado, ni puede volver a activarse; pero, no se borra de la base de datos. Adicionalmente, cuando se va a crear un identificador igual a un eliminado, no debe permitir el sistema.

e. Vigencia.- El sistema debe permitir parametrizar un tiempo de vigencia de las cuentas de usuario. Todos los Identificadores de usuarios que no se hayan utilizado durante un plazo determinado (plazo de inactividad) serán deshabilitados. El plazo máximo para suspensión debe ser parametrizable en días. Se establece que el plazo máximo de suspensión o deshabilitación es parametrizable en días calendario.

***II. Autenticación***

a. Pantallas de presentación del inicio de sesión.- Esta debe mostrar mínima información para ingresar el usuario.
   - **Ingrese su Usuario y Contraseña.**
   - **Usuario:**
   - **Contraseña:**

   1. No deberá ofrecer ningún detalle u otro tipo de información acerca del sistema.
   2. No deberá permitir visualizar el ingreso de la contraseña, se debe usar otro tipo de caracteres no reconocibles.
   3. Si la autenticación es errónea, el sistema sólo facilitará información anónima de la cual no pueda interpretarse cuál ha sido la secuencia del proceso que ha fallado. Ejemplo Nombre de usuario o contraseña errónea

b. Bloqueo de Usuario.- Los identificadores de usuario se bloquearán si hay más de n intentos fallidos de autenticación, esto debe ser parametrizable.

c. Reactivación Automática.- En caso de bloqueo de usuario, el tiempo mínimo de reactivación automático será de n minutos, esto debe ser parametrizable.

d. Suspensión de Usuario por Intentos de Autenticación Fallidos.- Después de

un n número de intentos fallidos de autenticación, el sistema suspenderá al usuario. Esto debe ser parametrizable.

e. Pantalla Inicial.- Una vez completado el proceso de autenticación, se presentará al usuario la fecha y hora del anterior acceso satisfactorio, así como el número de autenticaciones fallidas realizadas desde ésta fecha.

f. Mensaje Inicial.- Es necesario que el momento en que el usuario ingrese al sistema, se presente un mensaje con las responsabilidades legales de la utilización del mismo, esto debe ser parametrizable en la base.

### *III. Cambio de Contraseñas*

a. Sintaxis de Contraseñas.- La sintaxis de las contraseñas contemplará al menos los siguientes puntos:
    1. Las contraseñas tendrán un mínimo de N caracteres, este valor será parametrizable.
    2. Contendrán al menos un caracter no alfanumérico, una letra y un número (Opcional, configurable por parte del administrador).
    3. No comenzarán por el identificador del usuario
    4. No serán igual que el identificador al revés
    5. Se podrá definir una lista de palabras no admitidas como claves.

b. Cambio Inicial.- El sistema forzará al usuario a cambiar la contraseña en el primer acceso.

c. Cambio luego de Reseteo.- El sistema forzará al usuario a cambiar la contraseña en el primer acceso después del reseteo realizado por el administrador.

d. Cambio por Caducidad.- El sistema forzará el cambio de aquellas contraseñas que no se hayan cambiado en un período mayor a n días parametrizables.

e. Cambio Voluntario.- El sistema ofrecerá mecanismos que permitan al usuario el cambio de contraseña en cualquier momento.

f. Mecanismos de Funcionalidad.- El sistema deberá cumplir con las siguientes funcionalidades:

    1. Las contraseñas no se visualizarán en pantalla durante la introducción de las mismas.

2. Se pedirá confirmación de la nueva contraseña antes de proceder al cambio (para evitar posibles errores de escritura).
4. Se debe validar que la clave no sea la misma que el usuario.
5. No debe permitir ingresar una cantidad n de contraseñas anteriores, esto debe ser parametrizable.
6. Se verificará la correcta sintaxis de la nueva contraseña antes de proceder al cambio.

### *IV. Almacenamiento de Contraseñas*

a. Las contraseñas no se almacenarán en formato texto sino que se almacenarán codificadas o cifradas con el mayor nivel de restricción de acceso posible, de forma que se garantice su confidencialidad e integridad.

b. Se deberá utilizar un algoritmo de cifrado seguro, esto es, que utilice patrones de encriptación de 16 o más bits como MD5.

### *V. Auditoria*

a. Se debe generar auditoria para:

1. Gestión o administración de usuarios y contraseñas
2. Monitoreo de las Sesiones de Usuario
3. Autenticación (inicios de sesión exitosos y fallidos)
4. Cambio de las contraseñas
5. Reseteos
6. Seguimiento de funcionalidades utilizadas por sesión.

Para cada uno de los casos se deberá almacenar:

1. Identificador del usuario
2. Fecha y hora del evento
3. Código del evento
4. Descripción del evento o id de la funcionalidad
5. Conexión o máquina desde la que se produjo el evento

### *VI. Ambiente de Operación.*

a. Durante el ciclo de autenticación (pantalla donde el sistema solicita la contraseña) se deberá tener un tiempo de expiración en minutos que será parametrizable, pasado este tiempo habrá un ciclo de refresco de pantalla, borrando la identificación de usuario.

b. Una vez que el usuario ha sido identificado y autenticado por el sistema de seguridad, se debe tener las siguientes consideraciones:

i. Todo el ambiente de operación del sistema debe ser garantizado como área segura hasta que el usuario salga del sistema o su página caduque.
ii. Todas las subsiguientes pantallas no deben presentar herramientas de navegación. Solo debe presentar una opción de salir la misma que cerrara la pantalla segura con la que se estableció la conexión.

c. El ambiente debe mantener un tiempo máximo de espera (TIMEOUT) de operación para considerar que la página ha caducado. El tiempo de espera debe ser parametrizable.

d. Monitoreo de las sesiones de usuario en el sistema. El administrador del Módulo de Seguridades debe tenar la capacidad de ver qué sesiones de usuario están iniciadas, cuantas son activas y además tener la capacidad de caducar o cerrar sesiones de usuario. Adicionalmente debería permitir saber en que parte del sistema se encuentra el usuario en una sesión determinada.

e. Proveer tipos de acceso a los datos de acuerdo al perfil del usuario (lectura, escritura, consulta, etc.)

f. No utilizar Identificadores de sesión predecibles o información similar en URLs o mensajes de la aplicación.

g. No utilizar paths, nombres de archivos o rutas de archivos en los mensajes de error.

h. Restringir el uso de cookies.

**VII. Lineamientos Generales**

- Garantizar que no exista bajo ninguna circunstancia una conexión directa entre los usuarios y las bases de datos y que dicha conexión se dé siempre a través de la aplicación.

**Alcance.**

El desarrollo de este módulo cubrirá únicamente la autenticación y autorización de usuarios, con sus funcionalidades relacionadas.

| **Observaciones.** | **Anexos.** |
|---|---|

**Elaborado por: Lucía Gordillo**

| **Revisado por: Henry Coral** | **Aprobado por: Antonio Deidán** |
|---|---|

## 5.3. Diagrama Entidad Relación

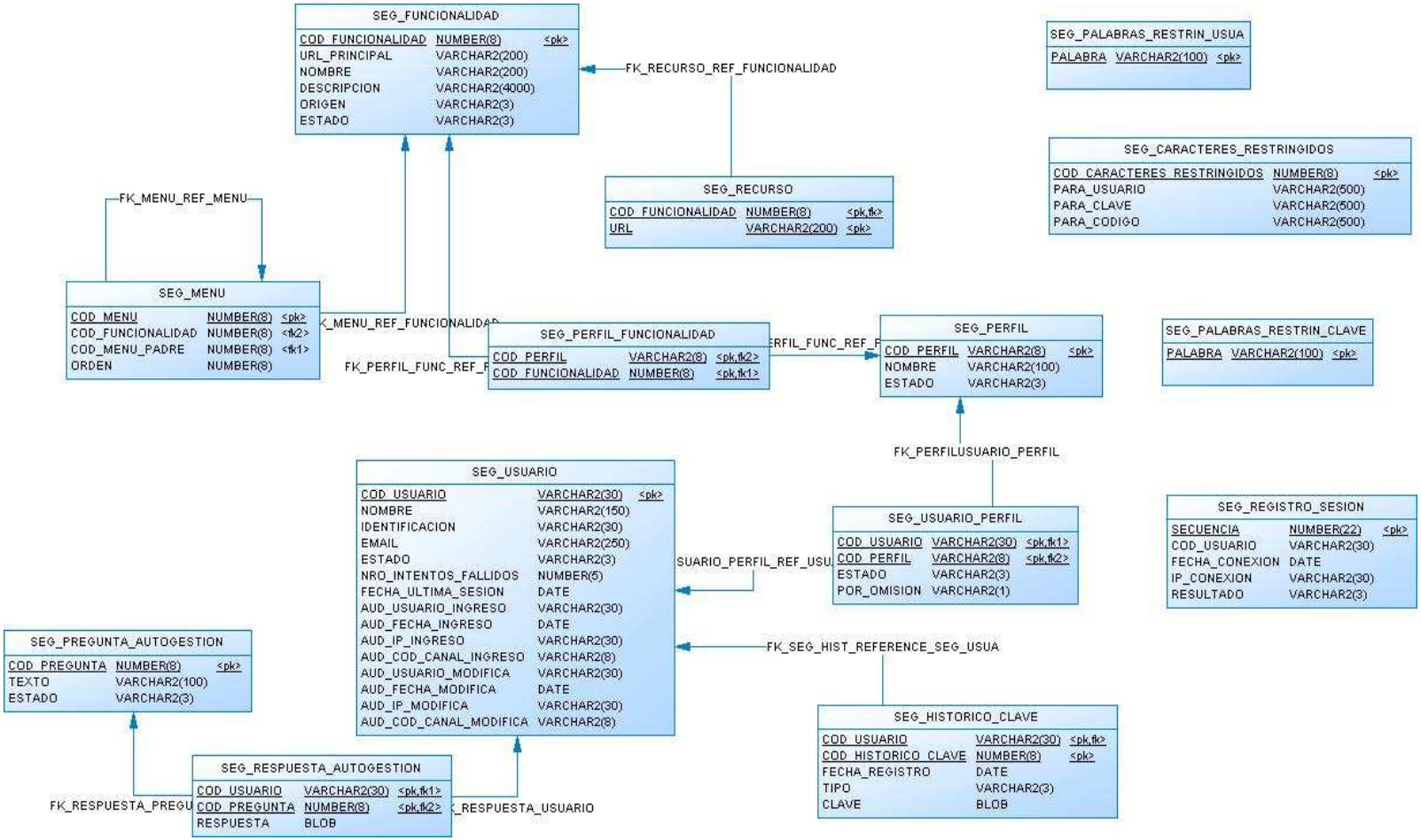


**Figura 5.1:** Entidad Relación

## 5.4. Diagrama de Casos de Uso

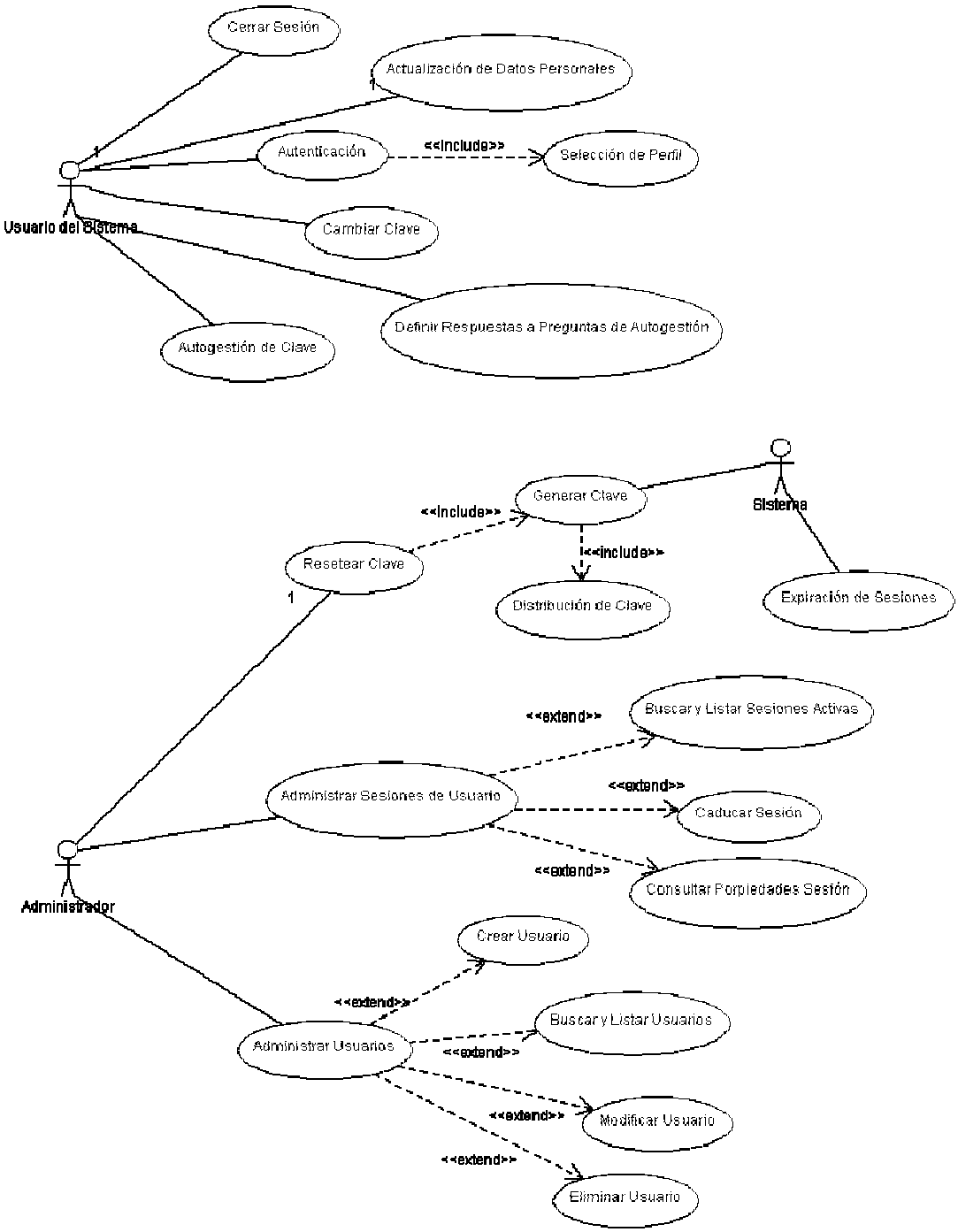


**Figura 5.2:** Casos de Uso

## 5.5. Descripción Detallada de Casos de Uso

### 5.5.1. Autorización y Autenticación de Usuarios

| | |
|---|---|
| **COD: SEG.001.001** | **Autenticación** |
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 15-12-2009 |
| **Versión** | 2 |
| **Objetivo** | Iniciar una sesión en el Sistema Gestor |
| **Resumen** | El usuario ingresa su código y clave, el sistema verifica que estos estén correctos y que el estado del mismo sea válido para iniciar la sesión. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Siempre |
| **Actores** | Administrador seguridades, Usuario Configuración, Usuario Gestor |
| **Precondiciones** | |
| **Escenario principal** | **1.** El sistema despliega la interfaz para iniciar el proceso de autenticación<br>**2.** El usuario ingresa su código y clave<br>**3.** El sistema verifica que los datos ingresados correspondan a un usuario del sistema<br>**4.** El sistema verifica que el estado del usuario sea válido<br>**5.** El sistema verifica si el último inicio de sesión fue después de los N días especificados para suspensión por inactividad.<br>**6.** El sistema verifica que la clave esté activa<br>**7.** El sistema registra el inicio de sesión satisfactorio<br>**8.** El sistema inicia una sesión de usuario.<br>**9.** El sistema ejecuta el Caso de Uso “Asignación de Perfil a Sesión” SEG.001.002 |

| **Escenarios alternativos** | **SI** | **ENTONCES** |
|---|---|---|
| | **2.1 El usuario no recuerda su clave** | • El usuario da clic en el vínculo de Autogestión de Clave<br>• Inicio de Caso de uso “Autogestión de Clave” SEG.002.003 |
| | **3.1 Los datos ingresados no corresponden a un usuario registrado** | • El sistema presenta la interfaz para inicio del proceso de autenticación, agregando un mensaje de error, sin dar indicios sobre la existencia o no del usuario.<br>• Se debe borrar automáticamente el texto ingresado en el campo clave.<br>• Si el código de usuario corresponde a un usuario válido debe registrar un intento de acceso fallido, si el usuario no existe se registrará un intento fallido de conexión en el log general. |
| | **3.2 Numero mayor a los intentos fallidos permitidos en la autenticación** | • El sistema deberá cambiar automáticamente el estado del usuario a Bloqueado.<br>• El sistema presentará una interfaz con el mensaje que especifique el bloqueo del usuario a causa de exceder el número permitido de intentos fallidos. |
| | **4.1 El estado del usuario no es válido** | • Si el estado de la cuenta es “BLO” y el tiempo transcurrido desde el ultimo intento de conexión supera el valor definido en el parámetro “Tiempo de Bloqueo por número de intentos fallidos” el sistema actualiza el estado a “ACT” y continúa paso 5. |

| | | |
|---|---|---|
| | | • El sistema registra el intento de inicio de sesión<br>• El sistema presenta la interfaz para inicio del proceso de autenticación, agregando un mensaje de error que describa la no posibilidad de inicio de sesión debido a que el estado del usuario no es válido. |
| | **4.2 La clave ingresada fue generada por el sistema** | • Si la clave es correcta y la misma fue generada por el sistema por cualquiera de los siguientes casos de uso: Reseteo de clave SEG.002.002, Autogestión de Clave SEG.002.003, Creación de Usuario SEG.003.002.<br>El sistema deberá iniciar el caso de uso: "Cambio de clave" SEG.002.001. |
| | **5.1 El último inicio de sesión excede los N días permitidos antes de Suspensión** | • El sistema cambia el estado de la cuenta del usuario a "SUS" |
| | **6.1 La fecha de registro de la clave ha caducado** | • El sistema presenta una advertencia al usuario indicándole que su clave ha expirado.<br>• El sistema inicia el caso de uso: "Cambio de Clave" SEG.002.001. |
| **Validaciones – Reglas de Negocio** | • Código de Usuario: Requerido, Alfanumérico, Máx 30 caracteres, no sensible a mayúsculas/minúsculas.<br>• Clave: Requerido, Alfanumérico, Max 30 Caracteres, sensible a mayúsculas/minúsculas<br>• Estados Validos del Usuario: ACT<br>• Estados No Validos para inicio de sesión: BLO, SUS, ELI<br>• Estado de Clave para cambio: RES, INI | |
| **Notas** | • Ninguna | |

| **COD: SEG.001.002** | **Selección de Perfil** | |
|---|---|---|
| **Autor** | Lucía Gordillo<br>Henry Coral | |
| **Fecha** | 15-12-2009 | |
| **Versión** | 2.0 | |
| **Objetivo** | Asignar un Perfil a una sesión autenticada. | |
| **Resumen** | Una vez que se ha iniciado una sesión en el sistema Gestor, el sistema da la posibilidad de elegir el Perfil con el cual va a trabajar el usuario, cuando el usuario tenga más de un Perfil asignado, caso contrario se asigna el único perfil a la sesión iniciada. | |
| **Prioridad** | Alta | |
| **Frecuencia de uso** | Siempre | |
| **Actores** | Administrador seguridades, Usuario Configuración, Usuario Gestor | |
| **Precondiciones** | • Usuario Autenticado | |
| **Escenario principal** | **1.** El sistema obtiene todos los perfiles que dispone la Sesión de Usuario<br>**2.** El sistema despliega una interfaz en la que se muestran todos los perfiles a los que el usuario tiene acceso.<br>**3.** El usuario selecciona el perfil con el cual desea trabajar y da clic en el botón "Aceptar".<br>**4.** El sistema asigna el perfil seleccionado a la sesión de usuario. | |
| **Escenarios alternativos** | **SI** | **ENTONCES** |
| | **1.1 La sesión de usuario solo tiene un perfil Asignado** | • Saltar a paso 4 del escenario principal, sin desplegar interfaz para selección de Perfil. |

| | |
|---|---|
| **Validaciones – Reglas de Negocio** | • Todo usuario debe tener al menos un perfil asignado para poder trabajar. |
| **Notas** | • Ninguna |

## 5.5.2. Gestión de claves

| | |
|---|---|
| **COD: SEG.002.001** | **Cambiar Clave** |
| **Autor** | Lucía Gordillo<br><br>Henry Coral |
| **Fecha** | 18-12-2009 |
| **Versión** | 2.5 |
| **Objetivo** | Permitir el cambio de la clave al usuario |
| **Resumen** | Este proceso hará posible que un usuario pueda cambiar su clave en cualquier momento. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Ocasionalmente |
| **Actores** | Administrador seguridades, Usuario Configuración, Usuario Gestor |
| **Precondiciones** | • Usuario autenticado. |
| **Escenario principal** | **1.** El sistema despliega la interfaz de cambio de clave<br>**2.** El Usuario ingresa los datos en los campos para cambio de clave: clave anterior, nueva clave, confirmación de la nueva clave y da clic en Guardar<br>**3.** El sistema verifica la clave anterior<br>**4.** El sistema verifica que la nueva clave sea válida.<br>**5.** El sistema verifica que la nueva clave no haya sido registrada anteriormente en las ultimas N claves.<br>**6.** El sistema despliega un mensaje de cambio satisfactorio, e inactiva la sesión actual del usuario.<br>**7.** El sistema redirecciona a la pantalla de autenticación. |
| **Escenarios alternativos** | **SI** / **ENTONCES** |
| | **3.1 La clave anterior no es valida** / • El sistema despliega mensaje de error, por no concordancia con la clave actual. |
| | **4.1 La nueva clave no cumple con las validaciones** / • El sistema despliega mensaje de error, por no concordancia con las reglas que debe tener la clave. |
| | **4.2 Nueva Clave y Confirmación de Nueva Clave no son iguales** / • El sistema despliega mensaje de error, por no ser iguales los campos de nueva clave y confirmación de nueva clave. |

| | |
|---|---|
| **Validaciones – Reglas de Negocio** | • Clave Anterior: Requerido<br>• Nueva clave: Requerido. Longitud Parametrizable(8,20)<br>• Confirmación de Nueva Clave: Requerido<br>• Se deben aplicar las validaciones por defecto mencionadas en el caso de uso "Generar Clave" SEG.002.004 |
| **Notas** | • Tener en cuenta que en este caso de uso se aplica validaciones personalizadas para la clave definidas por los clientes, en la sección de Validaciones del caso de Uso "Generar Clave" SEG.002.004 se ha descrito las validaciones por defecto de seguridad con las que cuenta el sistema Gestor. |

| **COD: SEG.002.002** | **Resetear Clave** |
|---|---|
| **Autor** | Lucía Gordillo<br><br>Henry Coral |
| **Fecha** | 18-12-2009 |
| **Versión** | 2.5 |
| **Objetivo** | Generar nueva clave para un usuario por solicitud del mismo o a criterio del Administrador de Seguridades. |
| **Resumen** | Se genera una nueva clave por medio del sistema para un usuario |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Regularmente |
| **Actores** | Administrador seguridades |
| **Precondiciones** | • Usuario autenticado |
| **Escenario principal** | **1.** El sistema despliega interfaz con listado de usuarios.<br>**2.** El Administrador de Seguridades selecciona el usuario al que se le va a resetear la clave y da clic en el botón Resetear.<br>**3.** El sistema despliega interfaz para la confirmación del reseteo de la clave.<br>**4.** El sistema ejecuta el caso de uso "Generar clave" SEG.002.004.<br>**5.** El sistema despliega interfaz con listado de usuarios |
| **Escenarios alternativos** | **SI** / **ENTONCES**<br>**3.1 No se confirma el reseteo de la clave cuando se selecciona un usuario** / • Desplegar nuevamente interfaz con listado de usuarios. |
| **Validaciones – Reglas de Negocio** | • Ninguna. |

| | |
|---|---|
| **Notas** | • Ninguna |
| **COD: SEG.002.003** | **Autogestión de clave** |
| **Autor** | Lucía Gordillo<br><br>Henry Coral |
| **Fecha** | 18-12-2009 |
| **Versión** | 2.5 |
| **Objetivo** | Permitir la generación automática de una nueva clave para un usuario luego de responder una serie de preguntas. |
| **Resumen** | El sistema genera una nueva clave para un usuario, luego de que este ha respondido una serie de preguntas anteriormente definidas por el. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Ocasionalmente |
| **Actores** | Administrador seguridades, Usuario Configuración, Usuario Gestor |
| **Precondiciones** | • Usuario haya definido previamente las respuestas a las preguntas de autogestión. |
| **Escenario principal** | **1.** El sistema despliega una interfaz en la que se permitirá ingresar el código de usuario, para el cual se va a realizar el proceso de autogestión de clave.<br>**2.** El usuario ingresa el código de usuario y da clic en el botón Aceptar.<br>**3.** El sistema obtiene las preguntas definidas de acuerdo al código de usuario ingresado.<br>**4.** El sistema despliega las preguntas de una en una, dando la opción al usuario para que ingrese las respuestas.<br>**5.** El sistema valida las respuestas ingresadas por el usuario a las diferentes preguntas.<br>**6.** El sistema ejecuta el caso de uso “Generar Clave” SEG.002.004.<br>**7.** El sistema presenta la interfaz de autenticación. |

| **Escenarios alternativos** | **SI** | **ENTONCES** |
|---|---|---|
| | **3.1 El código de usuario ingresado no corresponde a una cuenta de usuario válida** | • El sistema presenta un mensaje de error, por no concordancia del código de usuario con una cuenta válida. |
| | **3.2 El código de usuario ingresado pertenece a una cuenta de usuario para la cual no se han definido respuestas para Autogestión de Claves.** | • El sistema presenta un mensaje de error por no existir respuestas definidas para el proceso de Autogestión de Claves |
| | **4.1 Respuesta errónea a pregunta.** | • El sistema suspende el proceso y despliega un mensaje advirtiendo las penalidades que pueden darse por tratar de suplantar la identidad de un usuario. |

| | |
|---|---|
| **Validaciones – Reglas de Negocio** | • Código de Usuario: Requerido<br>• Respuesta: Requerido. |
| **Notas** | • Ninguna |

| **COD: SEG.002.004** | **Generar Clave** |
|---|---|
| **Autor** | Lucía Gordillo<br><br>Henry Coral |
| **Fecha** | 18-12-2009 |
| **Versión** | 2.5 |
| **Objetivo** | Generar una clave para un usuario |
| **Resumen** | Se genera una clave aleatoria para un usuario de acuerdo a las reglas predefinidas. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Regularmente |
| **Actores** | Sistema |
| **Precondiciones** | • Este caso de uso es ejecutado por los casos de uso “Crear usuario” SEG.003.002, “Resetear clave” SEG.002.002 y “Autogestión de Clave” SEG.002.003. |
| **Escenario principal** | **1.** El sistema recibe el código de usuario para el cual va a generar la clave<br>**2.** El sistema carga las reglas definidas para la generación de la clave.<br>**3.** El sistema genera una clave para el usuario<br>**4.** El sistema encripta la nueva clave y la registra.<br>**5.** El sistema ejecuta el caso de uso “Distribución Clave” SEG.002.005 |
| **Escenarios alternativos** | **SI** / **ENTONCES**<br>**2.1 No existen reglas definidas por el cliente para la generación de claves** / • El sistema carga las reglas por defecto para la generación de la clave<br>**Error al generar la clave** / • El sistema lanza un Excepción en la que describe el motivo por el cual no pudo generar la clave. |
| **Validaciones – Reglas de Negocio** | • La clave debe ser generada aleatoriamente y siguiendo las reglas definidas.<br>Validaciones por Defecto<br><br>• Clave no sea la misma que el código de usuario. |

| | |
|---|---|
| | • Caracteres restringidos.<br>• Palabras restringidas.<br>• Opción de caracteres no alfanumérico<br>• Opción de al menos una letra<br>• Opción de al menos un número<br>• No comenzarán por el identificador del usuario.<br>• No serán igual que el identificador del usuario al revés.<br>• Número de claves almacenadas históricamente para que no se repitan. |
| **Notas** | • Las reglas para la generación de clave son fijas. |

| **COD: SEG.002.005** | **Distribución Clave** |
|---|---|
| **Autor** | Lucía Gordillo<br><br>Henry Coral |
| **Fecha** | 18-12-2009 |
| **Versión** | 2.5 |
| **Objetivo** | Enviar la clave generada al usuario correspondiente |
| **Resumen** | El sistema vía correo electrónico envía la clave generada al usuario |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Regularmente |
| **Actores** | Sistema |
| **Precondiciones** | • Este caso de uso es ejecutado por el Caso de uso "Generar clave" SEG.002.004. |
| **Escenario principal** | **1.** El Caso de uso "Generar Clave" envía la clave generada y el código de usuario al que pertenece la misma.<br>**2.** El sistema obtiene la dirección electrónica del usuario a partir del código de usuario<br>**3.** El sistema carga el texto del mensaje de "Aviso de Seguridad y Manejo de Claves" a ser enviado al usuario.<br>**4.** El sistema reemplaza las variables definidas en el mensaje por los valores correspondientes.<br>**5.** El sistema envía vía correo electrónico el mensaje que contiene la clave generada al usuario correspondiente |
| **Escenarios alternativos** | **SI** / **ENTONCES**<br>**5.1 Error al enviar el mensaje con la nueva clave** / • Registrar en el log de la aplicación el error producido. |
| **Validaciones – Reglas de Negocio** | • Variables posibles a ser utilizadas en el mensaje de Seguridad<br>o Código de Usuario: Obligatorio<br>o Clave: Obligatorio |

| | |
|---|---|
| | o Nombre: Opcional<br>o Fecha Actual: Opcional |
| **Notas** | • Definir Caso de Uso en sección de Parametrización del Sistema, para la inclusión del mensaje de "Aviso de Seguridad y Manejo de Claves" |

### 5.5.3. Administración de Usuarios

| **COD: SEG.003.001** | **Administrar Usuarios** |
|---|---|
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 19-12-2009 |
| **Versión** | 2.2 |
| **Objetivo** | Realizar la gestión de los usuarios del sistema |
| **Resumen** | Permite buscar, listar, crear, modificar, cambiar de estado y resetear clave de un usuario del sistema |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Regularmente |
| **Actores** | Administrador seguridades |
| **Precondiciones** | • Usuario Autenticado |
| **Escenario principal** | **1.** El sistema despliega interfaz para búsqueda rápida de usuarios, además presenta controles para acceder a las diferentes acciones como: Búsqueda Avanzada, Crear, Modificar, Eliminar y Resetear Clave.<br>**2.** El usuario selecciona la acción que desea realizar |
| **Escenarios alternativos** | • Dependiendo de la acción seleccionada por el usuario se ejecuta el caso de uso correspondiente. |
| **Validaciones – Reglas de Negocio** | • Ninguna |
| **Notas** | • Ninguna |

| **COD: SEG.003.002** | **Crear Usuario** |
|---|---|
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 19-12-2009 |
| **Versión** | 2.2 |
| **Objetivo** | Crear una cuenta de usuario para el sistema |
| **Resumen** | Se realiza la creación de una cuenta de usuario para el sistema además de asignar perfiles a los que pertenecerá la cuenta de usuario. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Ocasionalmente |
| **Actores** | Administrador seguridades |
| **Precondiciones** | • Usuario Autenticado |
| **Escenario principal** | **1.** En la interfaz de Administración de Usuario se habilita la sección de "Formulario" en la cual se habilitan los campos necesarios para la creación del usuario. Los datos de ingreso del usuario deben dividirse en dos secciones: Datos de Usuario y Perfiles.<br>**2.** El Administrador de Seguridades ingresa la información solicitada para las dos secciones.<br>**3.** Los campos requeridos para el ingreso deberán estar identificados claramente de acuerdo al estándar.<br>**4.** Una vez que el Administrador de seguridades ha ingresado la información da clic en el botón "Aceptar"<br>**5.** El sistema realiza las validaciones necesarias y registra la información<br>**6.** El sistema ejecuta el caso de uso "Generar Clave" SEG.002.004 |

| | |
|---|---|
| | **7.** El Sistema deshabilita la sección Formulario y muestra al inicio de la lista de usuarios, la información del usuario que acaba de ser registrado. |
| **Escenarios alternativos** | **SI** / **ENTONCES**<br>**3.1 No se ingresa información en un campo obligatorio** → • Sistema despliega una advertencia obligando al usuario a ingresar la información con formato correcto<br>**3.2 La información ingresada no corresponde al tipo esperado** → • Sistema despliega una advertencia obligando al usuario a ingresar la información con formato correcto. |
| **Validaciones – Reglas de Negocio** | Datos Usuario<br>• Código Usuario: Requerido, Verificar caracteres restringidos, Longitud de acuerdo a Reglas, por defecto: (5;30).<br>• Nombres: Requerido, Alfanumérico, Longitud de acuerdo a regla, por defecto (2, 50)<br>• Apellidos: Requerido, Alfanumérico, Longitud de acuerdo a regla, por defecto (2, 50)<br>• Identificación: Alfanumérico, Longitud Máxima 30<br>• Email, Requerido, Formato Mail, Máximo 250<br>• Estado: Requerido, Por defecto ACT Activo<br>• Lenguaje: Requerido, Lista de valores, por defecto el lenguaje de la aplicación.<br><br>Perfiles<br>• Cada usuario debe tener al menos un perfil asignado<br>• Solo un perfil puede tener la marca Por Omisión<br><br>Globales<br>• El código de usuario no puede repetirse |
| **Notas** | • Ninguna |

| | |
|---|---|
| **COD: SEG.003.003** | **Buscar y Listar** |
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 19-12-2009 |
| **Versión** | 2.1 |
| **Objetivo** | Listar las cuentas de usuarios y permitir hacer búsquedas. |
| **Resumen** | El usuario ingresa un valor para búsqueda rápida o varios valores para búsquedas avanzadas. El sistema ejecuta la búsqueda y despliega la información. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Siempre |
| **Actores** | Administrador de seguridades |
| **Precondiciones** | • Ninguna |
| **Escenario principal** | **1.** El sistema despliega la interfaz de administración de usuarios, habilitando la sección de búsqueda rápida<br>**2.** El usuario selecciona el filtro de búsqueda rápida con el cual desea ejecutar la búsqueda e ingresa el valor del filtro.<br>**3.** El sistema ejecuta la búsqueda de acuerdo a la información ingresada<br>**4.** El sistema muestra la información de la búsqueda ejecutada. |
| **Escenarios alternativos** | **SI** / **ENTONCES**<br>**2.1 Se da clic en "Avanzada"** → • El sistema habilita la sección de formulario con los campos del usuario para realizar la búsqueda avanzada.<br>• El Usuario procede a ingresar los datos y hace |

| | | |
|---|---|---|
| | | clic en “Consultar”.<br>. |
| | **3.1 No encuentra datos** | • El sistema despliega una lista vacía |
| **Validaciones – Reglas de Negocio** | • Ninguna | |
| **Notas** | • Ninguna | |

| | |
|---|---|
| **COD: SEG.003.004** | **Modificar Usuario** |
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 19-12-2009 |
| **Versión** | 2.1 |
| **Objetivo** | Permitir modificar la información de una cuenta de usuario |
| **Resumen** | El sistema despliega la información de la cuenta de usuario a ser modificada, el Administrador de seguridades, realiza los cambios y guarda la información. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Ocasionalmente |
| **Actores** | Administrador Seguridades |
| **Precondiciones** | • Usuario autenticado |
| **Escenario principal** | **1.** El sistema despliega una lista de usuarios<br>**2.** El Administrador de Seguridades selecciona la cuenta de usuario que desea modificar y da clic en el botón modificar.<br>**3.** El sistema habilita la sección Formulario y carga la información de la cuenta de usuario permitiendo la edición de ciertos campos.<br>**4.** El administrador de seguridades realiza los cambios y da clic en el botón “Aceptar”<br>**5.** El sistema realiza las validaciones necesarias y registra la información<br>**6.** El Sistema deshabilita la sección Formulario y muestra en la lista de usuarios registrados la información del usuario que acaba de ser modificado |
| **Escenarios alternativos** | |

| **SI** | **ENTONCES** |
|---|---|
| **3.1 No se ingresa información en un campo obligatorio** | • Sistema despliega una advertencia obligando al usuario a ingresar la información solicitada |
| **3.2 La información ingresada no corresponde al tipo esperado** | • Sistema despliega una advertencia obligando al usuario a ingresar la información con formato correcto. |
| | |

| | |
|---|---|
| **Validaciones – Reglas de Negocio** | Datos Usuario<br>• Nombres: Requerido, Alfanumérico, Longitud de acuerdo a regla, por defecto (2, 50)<br>• Apellidos: Requerido, Alfanumérico, Longitud de acuerdo a regla, por defecto (2, 50)<br>• Identificación: Alfanumérico, Longitud Máxima 30<br>• Email, Requerido, Formato Mail, Máximo 250<br>• Estado: Requerido,<br>• Lenguaje: Requerido, Lista de valores.<br><br>Perfiles<br>• Cada usuario debe tener al menos un perfil asignado<br>• Solo un perfil puede tener la marca Por Omisión |
| **Notas** | • El campo estado no deberá tener la opción de Eliminado (ELI) |

| COD: SEG.003.005 | Eliminar Usuario |
|---|---|
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 19-12-2009 |
| **Versión** | 2.0 |
| **Objetivo** | Eliminar un usuario. |
| **Resumen** | Elimina un usuario |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Casi Nunca |
| **Actores** | Administrador seguridades |
| **Precondiciones** | • Usuario Autenticado |
| **Escenario principal** | **1.** El sistema despliega la lista de usuarios.<br>**2.** El administrador selecciona un usuario y presiona el botón “Eliminar”.<br>**3.** El sistema presenta una ventana pidiendo confirmación de la acción.<br>**4.** El sistema valida la posibilidad de eliminación del usuario<br>**5.** El sistema elimina el usuario y guarda los cambios.<br>**6.** El sistema despliega la lista de usuarios en la que ya no aparece el usuario eliminado. |
| **Escenarios alternativos** | **SI** / **ENTONCES**<br>**3.1 El usuario responde NO en la ventana de confirmación.** / • El sistema presenta el listado de usuarios. |
| **Validaciones – Reglas de Negocio** | • El usuario podrá ser eliminado solo si no está asignado a ningún perfil y si no ha iniciado sesiones. |
| **Notas** | • Ninguna |

## 5.5.4. Administración de Sesiones de Usuario

| COD: SEG.004.001 | Administración de Sesiones de Usuario |
|---|---|
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 19-12-2009 |
| **Versión** | 2.2 |
| **Objetivo** | Realizar la Gestión de las sesiones activas de los Usuarios |
| **Resumen** | Permite caducar y consultar las propiedades de las sesiones de usuarios. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Ocasionalmente |
| **Actores** | Administrador seguridades |
| **Precondiciones** | • Usuario Autenticado |
| **Escenario principal** | **1.** El sistema despliega la interfaz de Administración de Sesiones, presentando las opciones de: “Caducar Sesión” (SEG.004.002), “Monitorear” (SEG.004.003) y “Buscar y Listar” (SEG.004.004) sesiones activas<br>**2.** El usuario selecciona la acción que desea realizar |
| **Escenarios alternativos** | • Dependiendo de la acción seleccionada por el usuario se ejecuta el caso de uso correspondiente. |
| **Validaciones – Reglas de Negocio** | • Ninguna |
| **Notas** | • Ninguna |

<table>
<tr><td>COD: SEG.004.002</td><td colspan="2">Buscar y Listar Sesiones de Usuario Activas</td></tr>
<tr><td>Autor</td><td colspan="2">Lucía Gordillo<br>Henry Coral</td></tr>
<tr><td>Fecha</td><td colspan="2">19-12-2009</td></tr>
<tr><td>Versión</td><td colspan="2">2.1</td></tr>
<tr><td>Objetivo</td><td colspan="2">Buscar y Listar las Sesiones de Usuario Activas.</td></tr>
<tr><td>Resumen</td><td colspan="2">El usuario ejecuta la búsqueda de Sesiones de usuario activas.</td></tr>
<tr><td>Prioridad</td><td colspan="2">Alta</td></tr>
<tr><td>Frecuencia de uso</td><td colspan="2">Ocasionalmente</td></tr>
<tr><td>Actores</td><td colspan="2">Administrador seguridades</td></tr>
<tr><td>Precondiciones</td><td colspan="2">• Usuario autenticado</td></tr>
<tr><td>Escenario principal</td><td colspan="2">1. El sistema despliega la interfaz para la administración de Sesiones de Usuario, habilitando la sección de “Búsqueda Rápida”;<br>2. El Administrador de seguridades ingresa el valor para el filtro de búsqueda y da clic en el botón “Buscar”;<br>3. El sistema despliega la lista de Sesiones de Usuario respectiva.</td></tr>
<tr><td>Escenarios alternativos</td><td colspan="2"><table><tr><th>SI</th><th>ENTONCES</th></tr><tr><td>3.1 No encuentra datos</td><td>• El sistema despliega una lista vacía</td></tr><tr><td>3.2 La búsqueda contiene más registros del máximo definido</td><td>• El sistema despliega una advertencia en la que indica el número total de registros de la búsqueda ejecutada preguntando al usuario si está seguro de desplegar todos los registros.<br>• Si el usuario da clic en el botón No, se despliega el resultado de la búsqueda presentando los N primeros registros de la misma.<br>• Si el usuario da clic en el botón Sí el sistema despliega todos los registros de la búsqueda.</td></tr></table></td></tr>
<tr><td>Validaciones – Reglas de Negocio</td><td colspan="2">• Ninguna</td></tr>
<tr><td>Notas</td><td colspan="2">• No se habilitará la funcionalidad de Búsqueda Avanzada para este requerimiento, debido a que no aplica.</td></tr>
<tr><td>COD: SEG.004.003</td><td colspan="2">Caducar session</td></tr>
<tr><td>Autor</td><td colspan="2">Lucía Gordillo<br>Henry Coral</td></tr>
<tr><td>Fecha</td><td colspan="2">09-12-2009</td></tr>
<tr><td>Versión</td><td colspan="2">2.2</td></tr>
<tr><td>Objetivo</td><td colspan="2">Caducar las sesiones que se encuentren activas en el sistema.</td></tr>
<tr><td>Resumen</td><td colspan="2">Obliga a que se caduquen una sesión de usuario abierta en el sistema a criterio del administrador de seguridades</td></tr>
<tr><td>Prioridad</td><td colspan="2">Alta</td></tr>
<tr><td>Frecuencia de uso</td><td colspan="2">Ocasionalmente</td></tr>
<tr><td>Actores</td><td colspan="2">Administrador seguridades</td></tr>
<tr><td>Precondiciones</td><td colspan="2">• Usuario autenticado</td></tr>
<tr><td>Escenario principal</td><td colspan="2">1. El sistema despliega el listado de las sesiones activas en el sistema<br>2. El Administrador de seguridades selecciona las sesiones y da clic en “Caducar Sesión”.<br>3. El Sistema despliega el mensaje confirmación para caducar las sesiones escogidas.<br>4. El Administrador de seguridades da clic en aceptar<br>5. El sistema caduca las sesiones elegidas y despliega la lista de sesiones activas actualizada.</td></tr>
<tr><td>Escenarios alternativos</td><td colspan="2"><table><tr><th>SI</th><th>ENTONCES</th></tr><tr><td>4.1 No se confirma la</td><td>• El sistema despliega nuevamente la interfaz con</td></tr></table></td></tr>
</table>

| | | |
|---|---|---|
| | **aceptación para Caducar la sesión** | el listado de las sesiones activas en el sistema. |
| **Validaciones – Reglas de Negocio** | • Ninguna | |
| **Notas** | • Ninguna | |

| | |
|---|---|
| **COD: SEG.004.004** | **Consultar Propiedades Sesión** |
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 19-12-2009 |
| **Versión** | 2.1 |
| **Objetivo** | Mostrar las propiedades de una sesión activa en el sistema. |
| **Resumen** | Presentar la información completa de una sesión en particular. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Ocasionalmente |
| **Actores** | Administrador seguridades |
| **Precondiciones** | • Usuario autenticado |
| **Escenario principal** | **1.** El sistema despliega la interfaz con el listado de las sesiones activas en el sistema;<br>**2.** El Administrador de seguridades selecciona la sesión que se desea consultar sus propiedades y da clic en el botón consultar;<br>**3.** El sistema habilita la sección "Formulario" en modo lectura únicamente y despliega la información de la sesión seleccionada. |

| **Escenarios alternativos** | **SI** | **ENTONCES** |
|---|---|---|
| | **3.1 Error al consultar las propiedades de la sesión** | • El sistema despliega la interfaz con información de que se produjo un error en el sistema. |

| | |
|---|---|
| **Validaciones – Reglas de Negocio** | • Ninguna |
| **Notas** | • Los campos a presentarse serán definidos en la Descripción Funcional, que referencia a este Caso de Uso. |

## 5.5.5. Actualización de Datos de usuario

| | |
|---|---|
| **COD: SEG.005.001** | **Definir Respuestas a Preguntas de Autogestión** |
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 26-12-2009 |
| **Versión** | 2.5 |
| **Objetivo** | Definir un grupo de preguntas que son necesarias para autogestionar la clave de acceso al sistema. |
| **Resumen** | Seleccionar un grupo de preguntas incluyendo las respuestas personalizadas del usuario que serán almacenadas para poder realizar la autogestión de la clave de acceso al sistema. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Ocasionalmente |
| **Actores** | Administrador seguridades, Usuario Configuración, Usuario Gestor |
| **Precondiciones** | • Usuario autenticado |
| **Escenario principal** | **1.** El sistema despliega la interfaz con el listado de las preguntas para realizar la autogestión de la clave.<br>**2.** El Usuario selecciona las n preguntas que se requieren para la autogestión y da clic en siguiente. |

| | |
|---|---|
| | **3.** El sistema despliega una interfaz con las preguntas escogidas y el respectivo campo para responder a cada pregunta.<br>**4.** El Usuario ingresa cada una de las respuestas y da clic en el botón guardar<br>**5.** El sistema verifica que las preguntas estén respondidas y registra las preguntas seleccionadas y las respectivas respuestas<br>**6.** Desplegar la interfaz de confirmación con los datos registrados. |
| **Escenarios alternativos** | • Ninguno |
| **Validaciones – Reglas de Negocio** | • Validar que se seleccione el número de preguntas requerido o parametrizado en el sistema.<br>• Los campos de las respuestas para las preguntas seleccionadas deben ser requeridos y de longitud (8, 250). |
| **Notas** | • Ninguna |

| **COD: SEG.005.002** | **Actualización de Datos Personales** |
|---|---|
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 26-12-2009 |
| **Versión** | 2.5 |
| **Objetivo** | Permitir a cualquier usuario del sistema modificar sus datos personales. |
| **Resumen** | Se permitirá a un usuario cambiar datos personales. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Ocasionalmente |
| **Actores** | Administrador seguridades, Usuario Configuración, Usuario Gestor |
| **Precondiciones** | • Usuario autenticado |
| **Escenario principal** | **1.** El sistema despliega la interfaz con los campos identificación, email y lenguaje en forma de edición, los otros campos se presentaran como no editables.<br>**2.** El Usuario modifica los datos que desee y da clic en el botón “Aceptar”.<br>**3.** El sistema verifica los cambios hechos y actualiza el registro.<br>**4.** El sistema muestra un mensaje de aviso anunciando la actualización de los datos. |
| **Escenarios alternativos** | • Ninguno |
| **Validaciones – Reglas de Negocio** | • Identificación, Alfanumérico, Longitud Máxima 30<br>• Email, Requerido, Formato Mail, Máximo 250<br>• Lenguaje: Requerido, Lista de valores |
| **Notas** | • Los campos no editables deberán describirse en el documento “Descripción Funcional” |

## 5.5.6. Otros Procesos

| **COD: SEG.006.001** | **Cerrar session** |
|---|---|
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 26-12-2009 |
| **Versión** | 2.1 |
| **Objetivo** | Cerrar la sesión del usuario autenticado |
| **Resumen** | Cerrar la sesión del usuario autenticado. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Siempre |
| **Actores** | Administrador seguridades, Usuario Configuración, Usuario Gestor |
| **Precondiciones** | • Usuario autenticado |
| **Escenario principal** | **1.** El Usuario da clic en el botón Cerrar Sesión. |

| | |
|---|---|
| | **2.** El Sistema presenta una alerta para que el usuario confirme el Cierre de la Sesión.<br>**3.** El sistema invalida la sesión del usuario en el servidor de aplicaciones<br>**4.** El sistema redirecciona a la página de Autenticación. |
| **Escenarios alternativos** | **SI** / **ENTONCES**<br>**2.1 El usuario no confirma el Cierre de Sesión** / • Sistema no cierra la sesión. |
| **Validaciones – Reglas de Negocio** | • Este CU podrá ser llamado desde cualquier Funcionalidad de la aplicación. |
| **Notas** | • Ninguna |

| **COD: SEG.006.002** | **Expiración de sesiones** |
|---|---|
| **Autor** | Lucía Gordillo<br>Henry Coral |
| **Fecha** | 26-12-2009 |
| **Versión** | 2.5 |
| **Objetivo** | Se cierra la sesión de usuario, por tiempo de inactividad superado. |
| **Resumen** | El sistema Cierra la sesión de usuario una vez que ha superado el tiempo parametrizado para expiración de sesiones. |
| **Prioridad** | Alta |
| **Frecuencia de uso** | Siempre |
| **Actores** | Sistema |
| **Precondiciones** | • Usuario autenticado |
| **Escenario principal** | **1.** Todas las interfaces de usuario del sistema, deberán cargar el parámetro de "Tiempo máximo de inactividad"<br>**2.** La interfaz deberá controlar el tiempo de inactividad de un usuario en una página.<br>**3.** Cuando el tiempo de inactividad supere el tiempo máximo la página deberá cerrarse automáticamente, redireccionando a una página de error. |
| **Escenarios alternativos** | • Ninguno |
| **Validaciones – Reglas de Negocio** | • El parámetro "Tiempo máximo de inactividad" deberá estar configurado en el archivo web.xml<br>• Las sesiones deben cumplir con la regla de expiración de sesión |
| **Notas** | • Se considera como inactividad cuando una página no ha sido actualizada o cuando no se ha pasado a otra página. |

# CAPÍTULO VI

# CONCLUSIONES

- De la evaluación realizada entre las implementaciones de Java Server Faces 1.2 con soporte AJAX se determinó, a través del estudio comparativo, pruebas y revisiones por parte de expertos desarrolladores JEE que los frameworks que ofrecen mejores características y funcionalidades, y que además se adaptan eficazmente a las exigencias de las aplicaciones web actuales son: RichFaces, IceFaces y Myfaces.
- Todas las funcionalidades y características adicionales que cada framework JSF con soporte AJAX ofrece pueden ser estudiadas y experimentadas en las aplicaciones de ejemplo provistas por los desarrolladores de los frameworks.
- El framework que cubrió la mayoría de expectativas de los ejecutivos de la Empresa GESTORINC S.A. fue Richfaces de Jboss, que obtuvo la mayor calificación porcentual correspondiente al 89.75%, demostrando así, calidad, madurez y prestación del mismo.

# RECOMENDACIONES

- Tras el desarrollo del Piloto del Módulo de Seguridades del Sistema Gestor Fiducia Fondos con Richfaces, se pudo apreciar las excelentes funcionalidades que esta implementación ofrece, por lo que se recomienda, utilizar el framework en todos los productos de software que la Empresa

GESTORINC S.A. así lo requiera, al igual que todas las Empresas que desarrollan aplicaciones web no tradicionales. Mejorando de esta manera su captación de mercado, productividad y vanguardismo tecnológico.

- La evolución de la especificación JSF, ha significado un gran esfuerzo de parte de cada una de las casas desarrolladoras de software que auspician los frameworks que implementan la especificación, por lo cual se recomienda visitar continuamente los sitios web de cada producto, para revisar las actualizaciones que se van liberando continuamente.
- Debido a los problemas de estabilidad que se presentan en algunos componentes, se recomienda realizar las actualizaciones de las versiones de cada framework de forma continua. De esta forma se asegurará la solución de errores reportados y la estabilidad de la aplicación que se está desarrollando.

# GLOSARIO DE TÉRMINOS

## A

***AJAX.-*** Acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications).

***API.-*** Interfaz de programación de aplicaciones o API (del inglés application programming interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

## B

***BACK OFFICE.-*** Un back office (trastienda de la oficina) es la parte de las empresas donde tienen lugar las tareas destinadas a gestionar la propia empresa y con las cuales el cliente no necesita contacto directo. El término se construye a partir del concepto de que la oficina visible es el departamento de ventas y clientes y en la trastienda es donde se fabrica, diseña y gestiona la actividad.

***BACKEND.-*** El back-end es el estado final de un proceso. La conexión entre front-end y el back-end es un tipo de interfaz.

***BROWSER.-*** (Browser, explorador, navegador web). Aplicación que sirve para acceder a la WWW (todas las páginas web) y "navegar" por ella a través de los enlaces.

# C

***CLUSTERING.-*** El término cluster se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de componentes de hardware comunes y que se comportan como si fuesen una única computadora.

***CSS.-*** Las hojas de estilo en cascada (en inglés Cascading Style Sheets), CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C (World Wide Web Consortium) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.

# D

***DHTML.-*** El HTML Dinámico o DHTML (del inglés Dynamic HTML) designa el conjunto de técnicas que permiten crear sitios web interactivos utilizando una combinación de lenguaje HTML estático, un lenguaje interpretado en el lado del cliente (como JavaScript), el lenguaje de hojas de estilo en cascada (CSS) y la jerarquía de objetos de un DOM.

***DOM.-*** El Document Object Model (Modelo en Objetos para la representación de Documentos o también Modelo de Objetos del Documento ), abreviado DOM, es esencialmente una interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.

# E

***ECLIPSE.-*** Es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores.

# F

***FIDEICOMISO.-*** Es un contrato o convenio en virtud del cual una persona, llamada fideicomitente o también fiduciante, transmite bienes, cantidades de dinero o derechos, presentes o futuros, de su propiedad a otra persona (una persona natural, llamada fiduciaria), para que ésta administre o invierta los bienes en beneficio propio o en beneficio de un tercero, llamado fideicomisario.

***FLASH.-*** Programa de edición multimedia desarrollado originalmente por Macromedia (ahora parte de Adobe) que utiliza principalmente gráficos vectoriales, pero también imágenes ráster, sonido, código de programa, flujo de vídeo y audio bidireccional para crear proyectos multimedia.

***FLEX.-*** Adobe Flex (hasta 2005 Macromedia Flex) es un término que agrupa una serie de tecnologías publicadas desde Marzo de 2004 por Macromedia para dar soporte al despliegue y desarrollo de Aplicaciones Enriquecidas de Internet, basadas en su plataforma propietaria Flash.

***FRAME.-*** Se denomina frame en inglés, a un fotograma o cuadro, una imagen particular dentro de una sucesión de imágenes que componen una animación. La continua sucesión de estos fotogramas producen a la vista la sensación de movimiento, fenómeno dado por las pequeñas diferencias que hay entre cada uno de ellos.

***FRAMEWORK.-*** Estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, en base a la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

***FRONT OFFICE.-*** Es un término que traducido literalmente significa: oficina de delante. Viene indicado como el conjunto de las estructuras de una organización que gestionan la interacción con el cliente.

***FRONTEND.-*** En general el front-end es responsable de recoger entradas de los usuarios, procesarlas de tal manera que cumplan las especificaciones para que el back-end pueda usarlas.

# H

***HIBERNATE.-*** Hibernate es una herramienta de Mapeo objeto-relacional para la plataforma Java (y disponible también para .Net con el nombre de NHibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones.

***HTML.-*** (Hyper Text Mark-up Language o Lenguaje de Marcas de Hipertexto). Lenguaje desarrollado por el CERN que sirve para modelar texto y agregarle funciones especiales (por ej. hipervínculos). Es la base para la creación de páginas web tradicionales.

***HTTP.-*** (Hyper Text Mark-up Language o Lenguaje de Marcas de Hipertexto). Lenguaje desarrollado por el CERN que sirve para modelar texto y agregarle funciones especiales (por ej. hipervínculos). Es la base para la creación de páginas web tradicionales.

***HTTPS.-*** Hypertext Transfer Protocol Secure es una combinación del protocolo HTTP y protocolos criptográficos. Se emplea para lograr conexiones más seguras en la

WWW, generalmente para transacciones de pagos o cada vez que se intercambie información sensible (por ejemplo, claves) en internet.

## I

***IDE.-*** (Integrated Development Environment - Entorno integrado de desarrollo). Aplicación compuesta por un conjunto de herramientas útiles para un programador. Un entorno IDE puede ser exclusivo para un lenguaje de programación o bien, poder utilizarse para varios. Suele consistir de un editor de código, un compilador, un debugger y un constructor de interfaz gráfica GUI.

***IFRAME.-*** Inline frame o marco incorporado en inglés es un elemento HTML que permite insertar o incrustar un documento HTML dentro de un documento HTML principal. Fue introducido en el navegador Microsoft Internet Explorer en 1997 y durante mucho tiempo solo fue soportado en este navegador, la etiqueta Iframe actualmente es ya ampliamente soportado por gran variedad de navegadores.

***INPUT.-*** Una entrada se refiere a la información recibida en un mensaje, o bien al proceso de recibirla.

***INTERFAZ.-*** Parte de un programa que permite el flujo de información entre un usuario y la aplicación, o entre la aplicación y otros programas o periféricos. Esa parte de un programa está constituida por un conjunto de comandos y métodos que permiten estas intercomunicaciones.

# J

***JAVA.-*** Lenguaje de programación orientado a objetos. Fue desarrollado por James Gosling y sus compañeros de Sun Microsystems al principio de la década de los 90.

***JAVA PERSISTENCE API.-*** Java Persistence API, más conocida por su sigla JPA, es la API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, como sí pasaba con EJB2, y permitir usar objetos regulares (conocidos como POJOs).

***JAVASCRIPT.-*** JavaScript es un lenguaje de scripting basado en objetos, utilizado para acceder a objetos en aplicaciones. Principalmente, se utiliza integrado en un navegador web permitiendo el desarrollo de interfaces de usuario mejoradas y páginas web dinámicas. JavaScript es un dialecto de ECMAScript y se caracteriza por ser un lenguaje basado en prototipos, con entrada dinámica y con funciones de primera clase. JavaScript ha tenido influencia de múltiples lenguajes y se diseñó con una sintaxis similar al lenguaje de programación Java, aunque más fácil de utilizar para personas que no programan.

***JEE.-*** Java Platform, Enterprise Edition o Java EE (anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE hasta la versión 1.4), es una plataforma

de programación para desarrollar y ejecutar software de aplicaciones en Lenguaje de programación Java con arquitectura de N niveles distribuida, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. La plataforma Java EE está definida por una especificación. Similar a otras especificaciones del Java Community Process, Java EE es también considerada informalmente como un estándar debido a que los suministradores deben cumplir ciertos requisitos de conformidad para declarar que sus productos son conformes a Java EE; estandarizado por The Java Community Process / JCP.

***JSF.-*** Java Server Faces, una especificación de desarrollo basado en el patrón MVC (Modelo Vista Controlador).

***JSON.-*** Acrónimo de JavaScript Object Notation, es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.

***JSRS.-*** El Scripting remoto en JavaScript o (JSRS) es una técnica de desarrollo web para crear aplicaciones web interactivas.

***JSTL.-*** JavaServer Pages Standard Tag Library. Obtención de los parámetros de funcionamiento de un controlador por medio de algún método.

# P

***Plug-in.-*** Es un módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

# W

***Widget*.-** Componente con el cual un usuario interactúa en una interfaz gráfica. Son ejemplos de widget las ventanas, cajas de texto, checkboxs, listbox, entre otros. Son utilizados por los programadores para hacer aplicaciones con interfaces gráficas (GUI).

# BIBLIOGRAFÍA


- http://www.cienciaytecnologia.gob.bo/convocatorias/publicaciones/Metodologia.pdf
- http://www.jboss.org/richfaces
- http://www.icefaces.org
- http://myfaces.apache.org
- http://www.jsfmatrix.net/
- http:// livedemo.exadel.com/richfaces-demo/index.jsp
- http:// www.irian.at/myfacesexamples/home.jsf
- http:// www.irian.at/trinidad-demo/faces/index.jspx
- http://tobago.atanion.net/tobago-example-demo
- http://www.theserverside.com/news/1363997/AJAX-JSF-Frameworks-Review
- http://www.adobe.com/resources/business/rich_internet_apps/#open
- http://www.alegsa.com.ar/Dic/java.php

# ANEXO A

# FORMATO DE ENCUESTA

## ENCUESTA DE VALORACIÓN DE CARACTERISTICAS TECNOLOGICAS

**Nombre:** _____

**Cargo:** _____

Por favor, califique cada una de las siguientes características tecnológicas de acuerdo a la importancia que usted cree tienen al momento de la selección de una tecnología para el desarrollo de software.

Utilice una escala del 1 al 10 para su calificación. 1 Menos Importante, 10 Lo más importante.

| CARACTERISTICA TECNOLOGICA | IMPORTANCIA |
|---|---|
| **Presente y Futuro**<br>Corresponde al estado de madurez actual de la tecnología/herramienta y el futuro del mismo | |
| **Prestaciones Funcionales**<br>Funcionalidades que provee la tecnología/herramienta, así como la estabilidad de la misma | |
| Facilidad de Uso<br>Facilidad que tendría un equipo de desarrolladores sin conocimientos previos para la utilización e inclusión de la tecnología/herramienta en un proyecto real | |
| Costos de Licenciamiento y Soporte<br>Costos de licenciamiento tanto para el desarrollo de aplicaciones como para la fase de producción de las mismas; así como el costo del soporte técnico especializado | |

## HOJA DE LEGALIZACIÓN DE FIRMAS

## ELABORADA POR

_____

Lucía Gordillo

## COORDINADOR DE LA CARRERA

_____

Ing. Danilo Martínez

Lugar y Fecha:_____