ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA EN SISTEMAS

"ANÁLISIS COMPARATIVO ENTRE LOS FRAMEWORKS MYFACES, ICEFACES Y RICHFACES APLICADO AL SISTEMA NUTRICIONAL DE LA ESPOCH"

TESIS DE GRADO

Previo a la obtención del Título de

INGENIERA EN SISTEMAS INFORMÁTICOS

Presentado por:

MIRIAM EUGENIA JARAMILLO ESTRADA

Riobamba – Ecuador
2013

A G R A D E C I M I E N T O

Agradezco a Dios que con su inmenso amor me ha permitido llegar hasta este punto de mi vida; a mis padres por creer siempre en mi; al padre José Bravo Román, que con su cariño inculcó en mí valores de perseverancia; a mi esposo Ricardo, por su afecto y apoyo incondicional; a mi familia y amigos por sus consejos y palabras de aliento; a mis profesores, en especial al Ing. Jorge Menéndez, por su tiempo compartido, su paciencia, sus valiosas sugerencias y criterios en el presente trabajo y a la Escuela de Nutrición y Dietética, por colaborar en la parte aplicativa de la investigación.

D E D I C A T O R I A

A Dios por darme el regalo de la vida y guiarme siempre en todo momento; a mis padres por su amor y que por el orgullo que sienten por mí me han llevado a cumplir una más de mis metas; a mi esposo Ricardo por ser mi complemento y estar siempre a mi lado apoyándome en los momentos más difíciles de mi carrera; a mi hija Alina por ser la fuente de inspiración para continuar con la realización de mis objetivos y que con su amor, ternura y cariño ha llenado mi vida de felicidad y finalmente a mis amigos por la motivación constante para culminar una etapa más de mi vida.

F I R M A S R E S P O N S A B L E S Y N O T A S

NOMBRE	FIRMA	FECHA
Ing. Iván Menes **DECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**	……	……
Ing. Raúl Rosero **DIRECTOR DE LA ESCUELA DE INGENIERÍA EN SISTEMAS**	……	……
Ing. Jorge Menéndez **DIRECTOR DE TESIS**	……	……
Ing. Iván Menes **MIEMBRO DEL TRIBUNAL**	……	……
Tnlgo. Carlos Rodríguez **DIRECTOR DEL CENTRO DE DOCUMENTACIÓN**	……	……

NOTA DE LA TESIS ……

R E S P O N S A B I L I D A D D E L A U T O R

"Yo, Miriam Eugenia Jaramillo Estrada soy responsable de las ideas, doctrinas y resultados expuestos en esta tesis, y el patrimonio intelectual de la Tesis de Grado pertenece a la Escuela Superior Politécnica de Chimborazo"

_____

Miriam Eugenia Jaramillo Estrada

Í N D I C E D E A B R E V I A T U R A S

AJAX (Asynchronous JavaScript And XML), JavaScript asíncrono y XML.

AMMB Área Muscular Media del Brazo.

API (Application Programming Interface), Interfaz de programación de aplicaciones.

CAIS Centro de Atención Integral de Salud.

CDK (Careflow Development Kit), Kit de desarrollo de componentes.

CMMB Circunferencia Circular Media del Brazo.

CORBA (Common Object Broker Architecture), Arquitectura estándar de objetos gestionados en red.

CSS (Cascade Style Sheets), Hojas de Estilo en Cascada.

DAO (Data Access Object), Objeto de Acceso a Datos.

DESITEL Departamento de Sistemas y Telemática.

DOM (Document Object Model), Modelo en Objetos para la Presentación de Documentos.

ESPOCH Escuela Superior Politécnica de Chimborazo.

FTP (File Transfer Protocol), Protocolo de transferencia de archivos.

HTML (HyperText Markup Language), Lenguaje de marcado de hipertexto.

HTTP (Hypertext Transfer Protocol), Protocolo de transferencia de hipertexto.

IDE (Integrated Development Environment), Entorno integrado de desarrollo.

IEEE Instituto de Ingenieros Eléctricos y Electrónicos.

IMC Índice de Masa Corporal.

IU Interfaz de Usuario.

JCP (Java Community Process), Proceso de la Comunidad Java.

JDK (Java Development Kit), Herramientas de desarrollo para Java.

JEE (Java Plataform, Enterprise Edition), Plataforma Java, Edición Empresa.

JDBC (Java Database Connectivity), Conectividad de Base de Datos Java.

JNDI (Java Naming and Directory Interface), Interfaz de Nombrado y Directorio Java.

JPA Java Persistence API.

JSR (Java Specification Request), Peticiones de Especificación para Java.

JTA (Java Transaction API), API para transacciones en Java.

JVM (Java Virtual Machine), Máquina Virtual Java.

MGRS Multicenter Growth Referente Study.

MNA Mini Encuesta Nutricional del Anciano.

MSF Microsoft Solutions Framework.

MVC (Model View Controler), Modelo Vista Controlador.

OMS Organización Mundial de la Salud.

ORM (Object-Relational Mapping), Mapeo Relacional de Objetos.

POJO (Plain Old Java Object), Objeto Java Plano a la Antigua.

PPR Renderizado Parcial de Página.

RCUI Retraso del Crecimiento Intrauterino.

RIA (Rich Internet Aplicattions), Aplicaciones de Internet enriquecidas.

RMI (Java Remote Method Invocation), Invocación remota de métodos en Java.

RRHH Recursos Humanos.

SO Sistema Operativo.

SRS Especificación de Requerimientos de Software.

SSL (Secure Sockets Layer), Capa de conexión segura.

SQL (Structured Query Language), Lenguaje de consulta estructurado.

TCP/IP (Transmission Control Protocol/Internet Protocol), Protocolo de Control de Transmisiones/Protocolo de Internet.

UIX User Interface and Executive.

WEB (World Wide Web), Red informática mundial.

WHO World Health Organization.

XML (eXtensible Markup Language), Lenguaje de marcas extensible.

XUL (XML-based User-interface Language), Lenguaje basado en XML para la interfaz de usuario.

Í N D I C E G E N E R A L

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

I N T R O D U C C I Ó N

La Escuela de Nutrición y Dietética preocupada en mejorar la calidad de vida, propone mantener un control nutricional de los docentes, empleados, trabajadores, estudiantes politécnicos y público en general y dada la complejidad en el desarrollo de interfaces de usuario web con JSP, se han desarrollado frameworks multiplataforma orientado a las interfaces de usuario que maximizan la productividad en los desarrolladores, como Java Server Faces (JSF), permitiendo concentrarse en resolver problemas específicos de la lógica de negocios, dejando de lado los detalles de la programación de interfaces.

Entre los frameworks orientados a la interfaz de usuario se tiene a Java Server Faces (JSF). JSF es un framework de código abierto que simplifica el desarrollo de las interfaces en aplicaciones web, éste tiene varias implementaciones, entre las que se puede mencionar: MyFaces, IceFaces y RichFaces.

El presente trabajo investigativo pretende realizar un análisis comparativo entre los frameworks MyFaces, IceFaces y RichFaces, con el objetivo de determinar el más eficiente para desarrollar la aplicación del Sistema Nutricional de la ESPOCH.

C A P Í T U L O I

MARCO REFERENCIAL

1.1. Antecedentes del proyecto de Tesis

Actualmente la creación de un sitio WEB permite ampliar las perspectivas y horizontes de cualquier empresa o institución que haga uso de ellas, así como mostrar una imagen vanguardista.

Uno de los objetivos de la Escuela de Nutrición y Dietética es mejorar la calidad de vida de los docentes, trabajadores y estudiantes politécnicos, a través del reforzamiento de buenos hábitos alimentarios, recurriendo a la técnica de evaluación del estado nutricional; para lograr esto se pretende desarrollar una aplicación Web que se integre al Sistema del Dispensario Médico, que permita manejar la información nutricional y de laboratorio del paciente de una forma adecuada, emitiendo resultados, de tal manera que el profesional de nutrición pueda crear planes dietéticos y de actividades físicas.

Existen varios lenguajes de programación que permiten crear aplicaciones Web dinámicas, rentables y de calidad, entre los que se destacan: PHP, Java, Perl, Ruby, Python, HTLM, XML y ASP. Para el desarrollo de esta investigación se propone utilizar la tecnología Java Server Face (JSF), que trabaja bajo "componentes de interface de usuario del lado del servidor para aplicaciones Web basadas en Java" [1]. "JSF encaja bien de la capa de presentación basada en Model View Controller (MVC)" [1], esto proporciona mejoras a gran medida la experiencia del usuario más fiable y más rápidamente, por tal motivo muchos de los desarrolladores han visto factible implementar sus aplicaciones Web bajo frameworks creados sobre JSF, porque estos permiten crear productos escalables y de alto rendimiento, que satisfacen tanto las necesidades del cliente como las expectativas del desarrollador.

Existen frameworks que buscan mejorar la tecnología JSF, porque tratan de "acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones" [2]. Entre las implementaciones más importantes basadas en JSF se tienen a MyFaces, IceFaces y RichFaces por la gran popularidad que estos han adquirido en la Web.

La presente investigación pretende realizar un análisis comparativo entre los frameworks MyFaces, IceFaces y RichFaces, con el objetivo de determinar el más eficiente para crear la aplicación Web para el Sistema Nutricional de la ESPOCH.

[1] http://blog.pucp.edu.pe/item/4825/model-view-controler-jsf
[2] http://www.slideshare.net/jlbugarin/frameworks-java-14568352010-

1.2. Justificación del proyecto de tesis

En este apartado se comentará la justificación teórica y práctica, relacionada con los diferentes marcos de trabajo a utilizar durante el transcurso de la investigación, señalando las razones, importancia y ventajas que conllevará el desarrollo de esta investigación.

1.2.1. Justificación Teórica

Para la creación del Sistema Nutricional de la ESPOCH se desarrollará una aplicación Web. Las aplicaciones Web permiten mantener una comunicación activa entre el usuario y la información, mediante el uso de un navegador web como cliente ligero, además son multiplataforma, no ocupan espacio en el disco duro, el consumo de recursos es bajo, son portables y suelen tener una alta disponibilidad.

En la Escuela Superior Politécnica de Chimborazo (ESPOCH) se han desarrollado dos sistemas bajo el framework JSF, uno de ellos se encuentra en producción en la Escuela de Postgrado y de Educación Continua (EPEC), mientras que la otra aplicación que corresponde al Sistema Financiero, aún está en la fase de pruebas, motivo por el cual la Escuela Superior Politécnica de Chimborazo cuenta con la infraestructura necesaria para el desarrollo de este tipo de aplicaciones Web.

Teniendo en cuenta lo planteado anteriormente, para esta investigación se han elegido frameworks basados en JSF para realizar el análisis comparativo, esta tecnología permite "construir desarrollos del lado del servidor más rápidos" y "reutilizar objetos, optimizando los tiempos de desarrollo" [3].

Entre las implementaciones basadas en JSF más importantes se tiene a MyFaces, IceFaces y RichFaces por la popularidad que estos han cobrado en la Web. Elegir un framework adecuado constituye un pilar fundamental para la implementación de una aplicación, al hacerlo adecuadamente se conseguirá acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas para el desarrollo.

Existen investigaciones en curso sobre el framework JSF específicamente en el "ANÁLISIS DE LA TECNOLOGÍA JAVA SERVER FACE, COMO FRAMEWORK DE DESARROLLO DE APLICACIONES WEB, PARA PROCESOS DE COMERCIALIZACIÓN DE COMBUSTIBLE DE EP-PETROECUADOR" y en el "ANALISIS COMPARATIVO DE TECNOLOGIAS DE APLICACIONES WEB EN EL ENTORNO JSF Y ADF. CASO PRACTICO: IESS DE RIOBAMBA-CHIMBORAZO", investigaciones que no enfocan su estudio a la comparación entre los frameworks MyFaces, IceFaces y RichFaces.

El Departamento de Sistemas y Telemática (DESITEL) de la ESPOCH para disminuir costos y optimizar procesos, propone trabajar con el servidor de aplicaciones de software libre GlassFish y con la base de datos relacional orientada a objetos de código libre PostgreSQL, para proporcionar a los usuarios el acceso a la información ordenada, permitiéndoles visualizar, ingresar o actualizar los datos.

[3] http://www.clubdesarrolladores.com/articulos/mostrar/38-java-su-historia-ediciones-versiones-y-caracteristicas-como-plataforma-y-lenguaje-de-programacion/1

Desarrollar un buen software depende de una serie de etapas y actividades, donde el impacto de elegir una buena metodología es trascendental para el éxito de la aplicación, por tal motivo para el análisis, diseño y desarrollo del Sistema Nutricional se ha seleccionado la metodología Microsoft Solution Framework (MSF). MSF constituye una serie de modelos que se pueden adaptar a cualquier tecnología, ayudando al usuario a identificar prioridades, mediante la toma de decisiones estratégicas para lograr minimizar los riesgos y contrarrestar los problemas que puedan surgir.

1.2.2. Justificación Aplicativa

La Escuela de Nutrición y Dietética preocupada en mejorar la calidad de vida de los docentes, trabajadores, empleados, estudiantes politécnicos y público en general, ha visto importante integrar al CAIS (Centro de Atención Integral de Salud) de la ESPOCH una aplicación que cubra las expectativas deseadas; para lograr esto se seleccionará el framework más eficaz que se adapte al desarrollo de la aplicación Web para el Sistema Nutricional de la ESPOCH.

Esta aplicación Web busca ser un asistente eficiente para el profesional en nutrición, pues a través de la información nutricional y de laboratorio del paciente se pretende emitir el estado nutricional, valorar las necesidades o requerimientos nutricionales, así como también diagnosticar los posibles riesgos de salud que presente el ser humano con relación a su estado nutricional.

El Sistema Nutricional de la ESPOCH interactuará con la Base de Datos Centralizada para consultar los datos personales de los docentes, empleados y estudiante, en el caso de que se necesite obtener datos institucionales esta se integrará con los Sistemas de Recursos Humanos y Académico, también complementará al CAIS, con la finalidad de emitir reportes sobre el estado nutricional del paciente, de tal manera que los profesionales puedan hacer el uso adecuado de esta información.

1.3. Objetivos

En este apartado se describe los objetivos que se pretende conseguir con esta investigación.

1.3.1. Objetivo General

Realizar un análisis comparativo entre los frameworks MyFaces, IceFaces y RichFaces, para seleccionar el más eficiente y aplicarlo al desarrollo del Sistema Nutricional de la ESPOCH.

1.3.2. Objetivos Específicos

- Estudiar la arquitectura, estructura y componentes de los frameworks a investigar para definir los parámetros de evaluación.
- Comparar los frameworks MyFaces, IceFaces y RichFaces, con la finalidad de elegir el que mejor se adapte a la aplicación a realizar.
- Desarrollar e implantar la aplicación Web con el framework seleccionado para el Dispensario Médico de la ESPOCH en el área nutricional.

1.4. Hipótesis

El estudio comparativo entre los frameworks MyFaces, IceFaces y RichFaces permitirá seleccionar el más eficiente para desarrollar la aplicación Web para el Sistema Nutricional de la ESPOCH.

C A P Í T U L O I I

MARCO TEÓRICO

El marco teórico que se desarrolla a continuación, es el fundamento básico, que permite conocer los conceptos necesarios para el desarrollo de esta investigación.

Primero se partirá por definir el lenguaje de programación java y sus ediciones, con el fin de comprender el ámbito de la plataforma sobre la cual se trabajará.

Posteriormente se describirá la plataforma Java EE, su arquitectura, el Patrón Modelo Vista Controlador (MVC) y los Frameworks aplicables al modelo, con el fin comprender la aplicación del framework Java Server Faces (JSF).

A continuación se definirá el framework JSF, los objetivos que presenta el foco de desarrollo, las características, la interfaz de usuario, el ciclo de vida, funcionamiento, estructura, versiones e implementaciones, con el fin de conocer los Frameworks que se han desarrollado sobre JSF.

Por último se definirá cada una de las implementaciones desarrolladas sobre JSF: IceFaces, RichFaces, MyFaces Trinidad, MyFaces Tobago, MyFaces Tomahawk y MyFaces Orchestra, mismas que serán el fundamento de esta investigación.

2.1 JAVA

Java es toda una tecnología orientada al desarrollo de software con el cual se puede realizar cualquier tipo de programa. Hoy en día, la tecnología Java ha cobrado mucha importancia en el ámbito de Internet gracias a su plataforma J2EE. Pero Java no se queda ahí, ya que en la industria para dispositivos móviles también hay una gran acogida para este lenguaje [3].

Java también es un lenguaje de programación de propósito general, basado en clases y objetos, sencillo, robusto, distribuido, multihilo, arquitecturalmente neutro, portable, interpretado, de alto rendimiento y seguro, fue desarrollado por Sun Mycrosystems a principios de los años 90. Este lenguaje es muy extendido y cada vez cobra más importancia tanto en Internet como en el ámbito informático.

2.1.1 Ediciones de Java

Según la Descripción de J2EE [4], Java dispone las siguientes ediciones, cada una creada para cierto ambiente en particular.

- **Java Standard Edition (Java SE):** Entorno básico de Java. Ofrece un conjunto de clases y APIs que permiten desarrollar y ejecutar aplicaciones clientes y servidor, así como programas que se ejecuten en navegadores (applets).

[4] http://leo.ugr.es/J2ME/INTRO/intro_2.htm

- **Java Micro Edition (Java ME):** Específicamente diseñado para desarrollar aplicaciones para dispositivos embebidos y electrónicos, que tienen características peculiares ya que dos ediciones anteriores no son adecuadas para su utilización con ellos. Estos dispositivos normalmente tienen una potencia limitada, posibilidad de conectividad a una red (normalmente sin cables) y poseen interfaces gráficos.

- **Java Entreprise Edition (Java EE):** Agrupa APIs Java y tecnologías que no están basadas en este lenguaje. Se aconseja para el desarrollo de aplicaciones distribuidas.

Java es un lenguaje de programación orientado a objetos, que se distingue de otros por la independiente de la arquitectura, es decir se ejecuta en una variedad de equipos con diferentes sistemas operativos y microprocesadores. Es un programa fácil de aprender y muy bien estructurado, que permite crear aplicaciones fiables, el cual ha tomado importancia en el Internet gracias a la Plataforma Java EE, pues esta permite el desarrollo de aplicaciones empresariales del lado del servidor.

2.2 PLATAFORMA JAVA EDICIÓN EMPRESARIAL (JEE)

La API de la plataforma Java SE define las funcionalidades fundamentales del lenguaje de programación Java (tipos básicos, objetos y clases de alto nivel necesarias para la interconexión en red, seguridad, acceso a base de datos, desarrollo de interfaces gráficas y parseo de XML) y está formada por una máquina virtual, herramientas de desarrollo, despliegue de tecnologías y otras librerías de clases y toolkits usados para el desarrollo de aplicaciones Java. La plataforma Java EE ha sido construida encima de la plataforma Java SE y está formada igualmente por una API y un entorno de ejecución, pero a diferencia de Java SE, está enfocada al desarrollo y ejecución de aplicaciones de gran envergadura, multicapa, escalable, fiable y en una red segura. Este conjunto de

características definen lo que conocemos como Aplicaciones empresariales, por lo tanto, la plataforma Java EE esta especialmente diseñada para desarrollar aplicaciones empresariales [5].
JEE conocida anteriormente como J2EE, "es una herramienta claramente enfocada al mundo empresarial, orientada a un tipo de desarrollo específico" [6].

J2EE simplifica las aplicaciones empresariales basándolas en componentes modulares y estandarizados, proveyendo un complejo conjunto de servicios a estos componentes, y manejando muchas de las funciones de forma automática, sin necesidad de una programación compleja [7], además permite "ahorrar a la compañía, porque habilita una plataforma que reduce de manera significativa los costos y la complejidad de desarrollo de soluciones multicapa, resultando en servicios que pueden ser desarrollados rápidamente y ampliados fácilmente [8].
En este capítulo se definirá la Plataforma Java EE, así como también se dará a conocer su arquitectura, componentes, contenedores y servicios.

2.2.1 Definición

Sobre la plataforma J2EE, expone S. Allamaraju et al. (Anaya Multimedia) "La plataforma J2EE es esencialmente un entorno distribuido aplicación-servidor, un entorno Java que ofrece" [9]:

- Un conjunto de varios API de extensión Java para construir aplicaciones. Estos API definen un modelo de programación para aplicaciones J2EE.

[5] CARVAJAL, J. (2008). METODOLOGÍAS ÁGILES: Herramientas y modelos de desarrollo para aplicaciones Java EE como metodología empresarial. Barcelona. Tesis Final de Máster. 15p.
[6] GARCÍA, R. (2008). Diseño e Implementación de un Framework de Persistencia. Memoria PFC. Barcelona-España. Universitat Oberta de Catalunya. 17p.
[7] CACIÁ, D. (2007). Arquitectura de Aplicaciones J2EE basadas en el Patrón MVC utilizando Oracle ADF. Guatemala. Trabajo de Graduación. Univ. San Carlos de Guatemala. Fac. Ingeniería 6p.
[8] PALACÍN, M. (2007). Portal Web 2.0 utilizando Frameworks Struts. Trabajo de Fin de Carrera. Pp. 60.
[9] ALLAMARJU S., y otros (2002)., Programación Java Server Faces con J2EE., Pp. 55.

- Una infraestructura de periodo de ejecución para albergar y gestionar aplicaciones. Éste es el periodo de ejecución en el que residen sus aplicaciones.

Las aplicaciones que puede desarrollar con estos dos elementos pueden ser programas para controlar páginas Web o componentes para implementar transacciones complejas de bases de datos, o incluso applets de Java, todos ellos distribuidos por la red.

La plataforma Java EE está construida sobre la plataforma Java SE. La plataforma Java EE proporciona un entorno API y un tiempo de ejecución para desarrollar y ejecutar aplicaciones de red a gran escala, en múltiples niveles, escalable, confiable y segura [10]**.**

2.2.2 Arquitectura JEE

M. Abarca y G. Donoso [11], señalan que la Plataforma JEE utiliza una lógica de programación desarrollada en niveles o capas, las que permiten "encapsular o separar elementos de nuestras aplicaciones en partes claramente definidas", es decir, se puede "dejar procesos en un lugar, datos en otros, mostrar interfaces en otro".

En la programación por capas básicamente la idea es buscar la forma de separar lo que ve el usuario con los procesos creados por el desarrollador". Así se tiene "diferentes lenguajes que nos permiten desarrollar aplicaciones por capa por ejemplo JSP, ASP, PHP [11].

[10] http://docs.oracle.com/javaee/6/firstcup/doc/gkhoy.html#gcrkk
[11] ABARCA, M. Manual de Desarrollo Básico de Aplicaciones en la Plataforma J2EE en Ubuntu 7.x. <http://xxito.files.wordpress.com/2008/05/manualj2ee.pdf>

Las aplicaciones Java EE suelen ser consideradas aplicaciones de tres capas porque se distribuyen en tres localizaciones, ordenadores clientes, el sistema donde se ejecuta el servidor de aplicaciones, y el sistema donde reside la base de datos [12].

J. Riola (2008: 16:17) [5], expone en su tesis final**,** las siguientes capas de una aplicación JEE:

- **Capa Cliente:** La capa cliente está integrada por aplicaciones clientes que acceden al servidor Java EE y normalmente se encuentran en una máquina diferente a la del servidor. Los clientes hacen peticiones al servidor y este procesa las peticiones y las responde. Hay muchos tipos de aplicaciones diferentes que pueden ser clientes Java EE, y no tienen por qué ser aplicaciones Java EE, es más, a menudo no lo suelen ser. Los clientes pueden ser un navegador web, una aplicación autónoma, o incluso otros servidores que estén corriendo en otra máquina diferente de donde se encuentra el servidor Java EE.

- **Capa Web:** La capa o nivel Web, consiste en el conjunto de componentes que capturan la interacción entre los clientes y la capa de negocio. Sus principales tareas son las siguientes:
  - Generación dinámica del contenido, en varios formatos, para el cliente.
  - Recoger las entradas de los usuarios de la interfaz de las aplicaciones cliente y devolver los resultados apropiados desde los componentes de la capa de negocio.
  - Controlar el flujo de pantallas o páginas en el cliente.
  - Mantener el estado de los datos para una sesión de un usuario.
  - Realizar algunas operaciones básicas pertenecientes a la lógica de la aplicación y guardar temporalmente alguna información (usando JavaBeans).

[12] http://parasitovirtual.wordpress.com/category/cursos-y-articulos/desarrollo-aplicaciones-software/java/j2ee/

Las tecnologías más comunes de la capa de presentación o Web son:

- Servlets, procesan dinámicamente las peticiones y construyen las respuestas de los clientes, comúnmente utilizado con páginas HTML.
- JavaServerPages, definen como el contenido dinámico puede ser añadido a las páginas estáticas.
- JavaServer Faces, Componente framework de la interfaz de usuario para aplicaciones web, que permite la inclusión de componentes de interfaz de usuario (como pueden ser botones) en una página, convierte y valida datos de los componentes IU.
- JavaServerPages Estándar Tag Library, Una librería de tags que encapsula las funcionalidades principales más comunes de las páginas JSP.
- JavaBeans Components, objetos que temporalmente almacenan lo datos de las páginas de una aplicación.

- **Capa de Negocio:** La capa de Negocio está integrada por componentes que proveen la lógica de negocio para una aplicación. La lógica de negocio es el código que provee las funcionalidades para un particular dominio del negocio, como puede ser finanzas o un sitio de e-commerce. En un diseño correcto de una aplicación empresarial, las principales funcionalidades se encuentran en los componentes de la capa de negocio.

  Las tecnologías más comunes de la capa de negocio son:

  - Enterprise JavaBeans (EJB).
  - JAX-WS, web service endpoints.
  - Java Persistence API entities.

- **Capa de Datos o del Sistema de Información Empresarial:** La capa de datos está integrada por los servidores de bases de datos, sistemas ERP y otras fuentes de datos ya existentes, como mainframes. Habitualmente, estos recursos están situados en una máquina independiente de en la que se encuentra el servidor Java EE, y se acceden a ellos a través de la capa de negocio.

  Las tecnologías más comunes de la capa de datos son:

  - Java Database Connectivity API (JDBC).
  - Java Persistence API.
  - J2EE Connector Architecture.
  - Java Transaction API (JTA).

2.2.3 Patrón MVC y Frameworks para JEE

"La gran mayoría de Frameworks web J2EE están basados en el patrón MVC" [13], de tal manera que se empezará por definir el patrón MVC.

- **Patrón MVC (Modelo – Vista – Controlador):** El patrón MVC está indicado especialmente para el diseño de arquitecturas de aplicaciones que requieran de una gran interactividad con los usuarios, como es el caso de aplicaciones Web.

  Este patrón organiza la aplicación en tres partes bien diferenciadas. Por un lado tenemos el Modelo, el cual representa los datos de la aplicación y sus reglas de negocio, por otro la Vista, compuesta de vistas que representan los formularios de entrada y salida de datos, y finalmente, el Controlador, encargado de procesar las peticiones entrantes del usuario y controlar el flujo de ejecución del sistema [13].

[13] GÓMEZ, A. (2008). Implementación de una Aplicación Web utilizando Frameworks J2EE. Trabajo final de Carrera. Barcelona. Univ. Barcelona. Fac. Matemáticas. 11p.

El patrón MVC en la programación web J2EE se le conoce como arquitectura de modelo 2. Esta arquitectura consiste en la utilización de Servlets para procesar las peticiones, que estarían contenidos en el Controlador del patrón, y páginas JSP para mostrar la interfaz del usuario que representaría la Vista, y finalmente los famosos JavaBeans ubicados en el modelo [13].

Modelo. El modelo es el objeto que representa y trabaja directamente con los datos del programa: gestiona los datos y controla todas sus transformaciones. El modelo no tiene conocimiento específico de los diferentes controladores y/o vistas, ni siquiera contiene referencia a ellos. Es el propio sistema que tiene encomendada la responsabilidad de mantener enlaces entre el modelo y sus vistas, y notificar las vistas cuándo deben reflejar un cambio en el modelo [14].”Encapsula la lógica de negocio de la aplicación, acceso a los datos y su manipulación” [13].

Vista. La vista es el objeto que maneja la presentación visual de los datos gestionados por el modelo. Genera una representación visual del modelo y muestra los datos al usuario [14]**.** La vista es la encargada de generar las respuestas que deben ser enviadas al cliente. Esta respuesta normalmente incluirá datos generados por el controlador, entonces el contenido de la página no será estático sino que será generado de forma dinámica” [13].

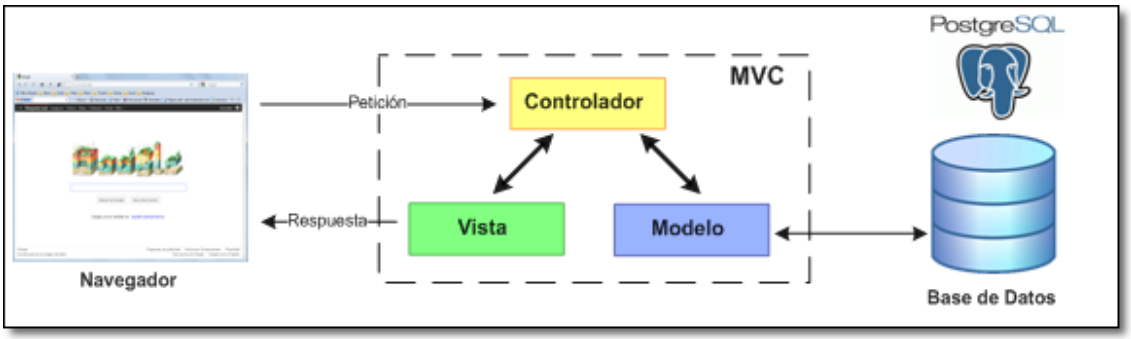


Figura II. 1. Patrón MVC con la interacción de la capa de presentación y de datos.
Fuente: http://www.maia.ub.es/~jaume/TFC/AngelGomezGarcia.pdf

[14] http://www.sicuma.uma.es/sicuma/Formacion/documentacion/JSF.pdf

En la **Figura II. 1,** se puede observar claramente que “este patrón nos proporciona una clara separación entre las distintas responsabilidades de la aplicación web” [14], mismas que se explican a continuación:

Controlador. El controlador es el objeto que proporciona significado a las órdenes del usuario actuando sobre los datos representados por el modelo. Entra en acción cuando se realiza alguna operación, ya sea un cambio en la información del modelo o una interacción sobre la vista. Se comunica con el modelo y la vista a través de una referencia del propio modelo [14]. Todas las peticiones a la capa intermedia que se realicen desde el cliente pasarán por el Controlador, éste determinará las acciones a realizar e invocar al resto de los componentes de la aplicación como pueden ser el modelo o la vista [13].

- **Frameworks para JEE:** Un Framework es una estructura de soporte definida en la cual un proyecto de software puede ser organizado y desarrollado. Típicamente, un Framework puede incluir soporte de programas, bibliotecas y un lenguaje de scripting para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Provee de una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio [8].

  Básicamente, los Frameworks son construidos en base a lenguajes orientados a objetos. Esto permite la modularización de componentes y una óptima reutilización de código. Además, en la mayoría de los casos, cada framework específico implementará uno o más patrones de diseño de software que aseguren la escalabilidad del producto [6].

Los objetivos principales que persigue un framework son [6]:

- Acelerar el proceso de desarrollo.
- Reutilizar código ya existente.
- Promover buenas prácticas de desarrollo como el uso de patrones.

En particular, un framework web J2EE, es un conjunto de componentes de software, por ejemplo clases JAVA, descriptores y archivos de configuración en XML, basados en la plataforma J2EE y, que constituyen un diseño reutilizable que facilita y agiliza el desarrollo de sistemas web que se ejecutarán en servidores J2EE [15].

"Para facilitar el desarrollo de las aplicaciones J2EE se han ideado varios Frameworks" [16].

E. Fernández (2008:16:17) [16], destaca como principales frameworks J2EE los siguientes:

- **Java Server Faces:** orientado a la creación de interfaces de usuario.
- **Hibernate:** que tiene como objetivo facilitar la persistencia de objetos Java en bases de datos relacionales y al mismo tiempo la consulta de estas bases de datos para obtener objetos.
- **Spring:** que tiene como objetivo facilitar la configuración de los java beans dentro de una aplicación. Su meta es conseguir separar los accesos a datos y los aspectos relacionados con las transacciones, para permitir objetos de la capa de negocio reutilizables que no dependan de ninguna estrategia de acceso a datos o transacciones.

[15] http://www.ing.unp.edu.ar/wicc2007/trabajos/ISBD/109.pdf
[16] FERNÁNDEZ, E. (2008). Sistema de gestión de productos con emulación de RFID mediante sensores IEEE 802.15.4. Universidad Politécnica de Catalunya. 16-17p.

- **Struts:** orientado a la parte de control de eventos.

Estos Frameworks pueden ser utilizados entre sí, pues en la mayoría de los casos están destinados a dar solución a determinados problemas.

En la **Figura II. 2**, se puede observar una estructura del patrón MVC junto con los Frameworks JEE aplicables al modelo.

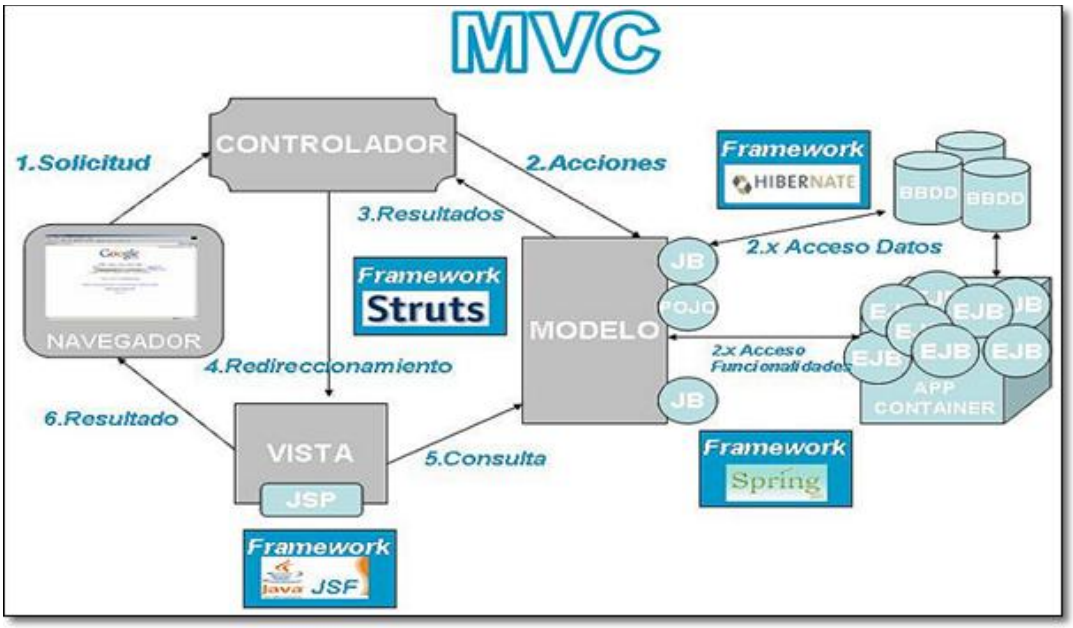


Figura II. 2. MVC junto a los Frameworks J2EE aplicables.
Fuente: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/625/1/00868tfc.pdf

- **Uso del Patrón MVC y framework**

R. García (2007-2008: 16) [6]**,** señala que el uso del patrón MVC y framework "de manera sintética se puede resumir en":

- El framework "llama" al código de nuestra aplicación.

- El desarrollador tiene que implementar interfaces y/o extender clases abstractas que proporcionará el framework.

Así, cada desarrollo de un framework realiza una implementación determinada y específica del Patrón MVC adaptándole a la casuística determinada por la arquitectura a la que se orienta al framework, el dominio que se desea implementar y las características de despliegue del mismo.

JEE ha evolucionado significativamente en el mercado, convirtiéndose en una buena solución para desarrollar aplicaciones empresariales, por las características que éste posee. JEE utiliza una arquitectura basada en cuatro capas, misma que puede ayudar en mayor disponibilidad y escalabilidad, aunque estos beneficios pueden incrementar la complejidad en el diseño y desarrollo del sistema.

Existe una gran cantidad de patrones de diseño y Frameworks para el desarrollo en JEE, sin embargo para la investigación se utilizará MVC y el framework Java Server Faces. JSF es un framework para construir interfaces de usuario para las aplicaciones que corren en un servidor.

2.3 JAVA SERVER FACES

Java Server Faces (JSF) es un framework Web J2EE de la familia de código fuente abierto, basado en componentes de interfaz de usuario (del lado del servidor) y eventos. “Un modelo orientado a eventos permite a las aplicaciones estar menos atado a detalles HTTP y simplifica el esfuerzo necesario para el desarrollo” [17].

[17] http://openaccess.uoc.edu/webapps/o2/bitstream/10609/669/1/00848tfc.pdf

Al igual que Struts, JSF pretende normalizar y estandarizar el desarrollo de aplicaciones web. Hay que tener en cuenta JSF es posterior a Struts, y por lo tanto se ha nutrido de la experiencia de éste, mejorando algunas de sus deficiencias. De hecho el creador de Struts (Craig R. McClanahan) también es líder de la especificación de JSF [18].

A continuación se presentará un resumen del framework Java Server Faces, explicando sus objetivos, características, estructura, interfaz de usuario, ciclo de vida, funcionamiento, implementaciones, versiones, ventajas y desventajas.

2.3.1 Definición

"JavaServer Faces (JSF) es una tecnología y framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE" [19], facilitando el desarrollo de éstas, y que sin embargo, realiza una separación entre comportamiento y presentación, además de proporcionar su propio servlet como controlador, implementando así los principios del patrón de diseño Model-View-Controller (MVC), lo que da como resultado un desarrollo más simple y una aplicación mejor estructurada [20].

"JSF usa JavaServer Pages (JSP) como la tecnología que permite hacer el despliegue las páginas, pero también se puede acomodar a otras tecnologías como XUL" [19].

[18] http://www.rincondeloajeno.com/introduccion-a-jsf-java-server-faces-parte-i/
[19] http://www.scribd.com/doc/28079982/Curso-de-JSF-2-0-con-Hibernate-3
[20] VIVEROS, C. (2006). Sistema generador de aplicaciones Web configurables para el registro en línea de congresos utilizando JavaSerer Faces. Tesis. Chochula-Puebla-México. Universidad de las Américas Puebla. 14p.

C. Simón (2009-2011: 6) [19], menciona que “JSF incluye”:

- Un conjunto de APIs para representar componentes de una interfaz de usuario y administrar su estado, manejar eventos, validar entrada, definir un esquema de navegación de las páginas y dar soporte para internacionalización y accesibilidad.
- Un conjunto por defecto de componentes para la interfaz de usuario.
- Dos bibliotecas de etiquetas personalizadas para JavaServer Pages que permiten expresar una interfaz JavaServer Faces dentro de una página JSP.
- Un modelo de eventos en el lado del servidor.
- Administración de estados.
- Beans administrados.

2.3.2 Objetivos

La Enciclopedia libre Wikipedia (2011) [21], señala, los siguientes objetivos de diseño como foco de desarrollo de 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. Estos componentes se obtendrán de un conjunto básico de clases base que se pueden utilizar para definir componentes nuevos.

[21] http://es.wikipedia.org/wiki/JavaServer_Faces

- 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.
- Especificar un modelo para la internacionalización y localización de la interfaz de usuario.
- 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.

2.3.3 Características

Las principales características que ofrece el framework JavaServer Faces son [22]:

- Utiliza páginas JSP para generar las vistas, añadiendo una biblioteca de etiquetas propia para crear los elementos de los formularios HTML.
- Asocia a cada vista con formularios un conjunto de objetos java manejados por el controlador (managed beans) que facilitan la recogida, manipulación y visualización de los valores mostrados en los diferentes elementos de los formularios.
- Introduce una serie de etapas en el procesamiento de la petición, como por ejemplo la de validación, reconstrucción de la vista, recuperación de los valores de los elementos, etc.
- Utiliza un sencillo fichero de configuración para el controlador en formato xml.
- Es extensible, pudiendo crearse nuevos elementos de la interfaz o modificar los ya existentes.
- Forma parte del estándar J2EE. En efecto, hay muchas alternativas para crear la capa de presentación y control de una aplicación web java, como Struts y otros frameworks, pero solo JSP forma parte del estándar.

[22] http://www.ecured.cu/index.php?title=Especial:Pdfprint&page=JSF

2.3.4 Interfaz de usuario

Oracle (2010) [23], señala que: La interfaz de usuario que es creada por la tecnología JavaServer Faces (representado por myUI en la **Figura II. 3**) se ejecuta en el servidor y se presenta o traduce para el cliente.

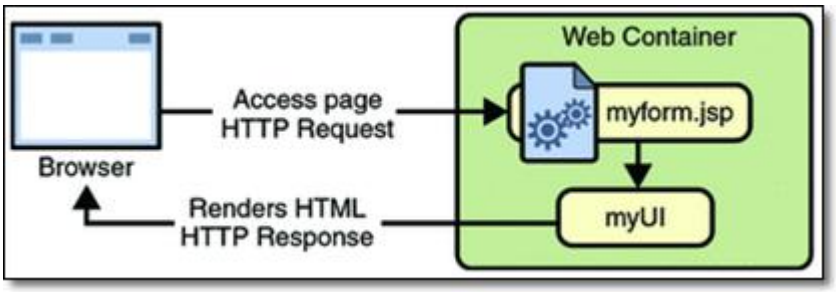


Figura II. 3. La ejecución de UI sobre el servidor
Fuente: http://download.oracle.com/javaee/5/tutorial/doc/gentextid-10788.html

La página JSP **myform.jsp**, es una **página JSF,** la cual es una página JSP que incluye etiquetas JSF. Estas expresan los componentes de interfaz de usuario a través de las etiquetas personalizadas definidas por la tecnología JSF. La UI de la aplicación web (representada por myUI en la **Figura II. 3**) administra los objetos referenciados por la página JSP.

Estos objetos incluyen:

- Los objetos componentes UI que mapean las etiquetas sobre la página JSP.
- Los detectores, validadores y convertidores que se registran en los componentes.
- Los componentes JavaBeans que encapsulan los datos y las funcionalidades específicas de los componentes específicos de la aplicación

[23] http://download.oracle.com/javaee/5/tutorial/doc/gentextid-10788.html

2.3.5 Ciclo de vida de JSF

El ciclo de vida de una aplicación JSF describe todas las acciones llevadas a cabo desde que el cliente inicializa una sesión. La mayoría de las fases en una aplicación JSF son llevadas a cabo por el Framework JSF dejando una pequeña parte de la responsabilidad en el desarrollador.

El ciclo de vida de una página de una página JSF está compuesto de seis faces, mismas que se pueden apreciar en la **Figura II. 4.**

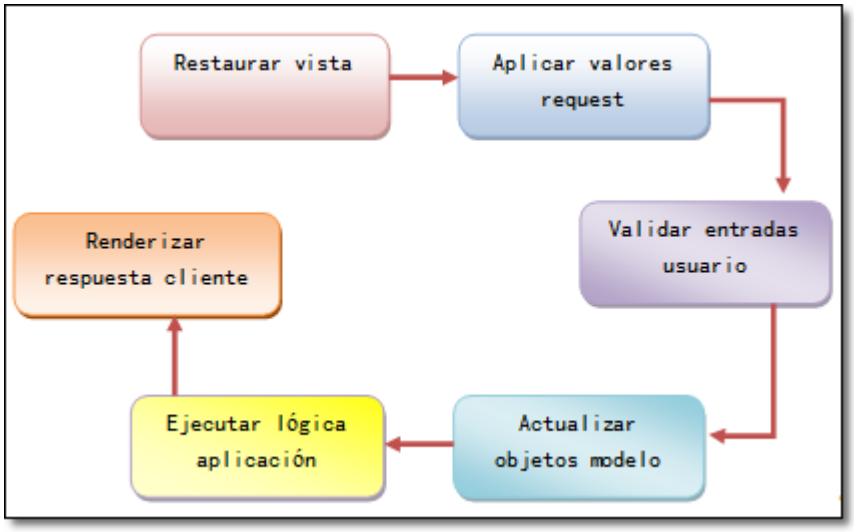


Figura II. 4. Ciclo de vida de JSF

Según ATI WIKI [24], las fases del ciclo de vida de JSF son:

- **Fase de restauración de la vista:** La fase de restauración de vista comienza cuando se realiza una solicitud de página JSF, esto sucede cuando se presiona un botón o un enlace. Una vista en JSF representa un árbol de componentes IU.

[24] http://150.185.75.30/atiwiki/index.php/JSF#Ciclo_de_vida

- **Fase de aplicación de valores del Request**: Luego de que se restaura la vista (o se crea una nueva si no existe) y el cliente ingresa la data, estos valores se mapean a su correspondientes componentes IU.

- **Fase de validación de las entradas del usuario**: JSF iterará sobre todos los nodos del árbol de componentes IU llamando tanto al validador por defecto como los personalizados, según se haya configurado cada componente.

- **Fase de actualización de los objetos del modelo**: En esta fase, los valores de los componentes mapeados y validados durante las fases anteriores se sitúan en los correspondientes JavaBeans.

- **Fase de ejecución de la lógica de la aplicación**: Mediante el mecanismo de mapeo de eventos se ejecutan las funciones definidas en los JavaBeans para llevar a cabo las operaciones de negocio de la aplicación.

- **Fase de renderizado de respuesta:** De acuerdo al mecanismo de mapeo de navegación, en función de los resultados obtenidos durante la fase anterior, se escoge la vista apropiada para mostrar al usuario.

2.3.6 Funcionamiento

Según O. Quesada, en su ponencia de Java Server Faces y el Uso de Patrones de Diseño [25], menciona que el funcionamiento de JSF se realiza de la siguiente manera:

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:

- Mostrar datos al usuario en cajas de texto y tablas.
- Recoger los datos introducidos por el usuario en los campos del formulario.
- Controlar el estado de los controles del formulario según el estado de la aplicación, activando, ocultando o añadiendo y eliminando controles y demás elementos.
- Realizando validaciones y conversiones de los datos introducidos por el usuario.
- Rellenando campos, listas, combos y otros elementos a medida que el usuario va interactuando con la pantalla.

[25] QUEZADA, J. Java Server Faces y el uso de Patones de Diseño. UCR Puntaneras - Costa Rica. Universidad de Costa Rica. 3-4p.

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

2.3.7 Estructura de JSF

El Tutorial de JavaServer Faces, menciona que [14]:

Las aplicaciones web correctamente planificadas tienen dos partes: la parte de presentación y la lógica de negocio

La parte de presentación afecta a la apariencia de la aplicación, y en el contexto de una aplicación basada en navegadores, la apariencia está determinada por las etiquetas HTML, esto comprende marcos, tipos de caracteres, imágenes, etc. La lógica de negocio se implementa en Java y determina el comportamiento de la aplicación.

En el contexto de JSF, la lógica de negocio está contenida en los beans, y el diseño está contenido en las páginas web.

- **Componentes de JSF**

J. Loor [26], señala que "JSF introduce dos nuevos términos al mundo del desarrollo de aplicaciones para JAVA":

Managed Bean: Un Managed Bean es un objeto identificado para el ambiente de la aplicación, para el cual se describe:

- Una identificación.

[26] http://www.scribd.com/doc/73476293/nDeveloper-JavaServerFaces

- Un alcance (scope) que puede ser: request, session, application, etc.
- Propiedades.

Backing Bean: Un Backing Bean es usualmente un Bean común de java que sirve de soporte para un objeto manejado dentro de la aplicación. Pongamos por ejemplo una página JSP en una aplicación orientada a web:

- La página JSP está especificada como un ManagedBean para la aplicación, con un identificador que la describe para toda la aplicación en general.
- En este archivo JSP se dibujan los controles necesarios para proveer a la página de funcionalidad.
- Esta página tiene asociado un Backing Bean que es un Bean de Java. En este Bean se codifican los comportamientos específicos asociados a cada control del Managed Bean representado por la página JSP.

La ventaja de los Beaking Beans es que pueden ser compartidos por un mismo Managed Bean, de manera que para diferentes páginas se pueden agrupar comportamientos comunes en un mismo Bean que se comparte con ambos.

- **Estructura de las páginas**

O. Quesada [25], en su ponencia de Java Server Faces y el uso de Patrones de Diseño, menciona sobre la estructura de páginas lo siguiente:

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

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 pintarla.

El siguiente paso es asociarle los backbeans. Para ello, del procesamiento de la página JSP, el controlador ha obtenido la lista de backbeans asociados, por lo que procede a buscarlos en sus correspondientes ámbitos de la aplicación como la request y la session. Los beans que no existan se crean llamando a los constructores de sus clases, definidos en la session de managed beans del fichero 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 backbean asociado.

- **Navegación**

O. Quesada [25], en su ponencia de Java Server Faces y el uso de Patrones de Diseño, da conocer lo siguiente sobre la navegación:

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. Hay varias posibilidades":

- Finalizar la petición mostrando la página JSP que originó la petición, que es la opción por defecto.
- Mostrando 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 fichero de configuración de JSF para determinar la página que se debe enviar como respuesta al usuario

- 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 request 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.

2.3.8 Implementaciones

EcuRed [27], en su artículo de JSF, menciona las siguientes implementaciones:

- **JSF Reference Implementation** de Sun Microsoft: Es la implementación de referencia de JavaServer Faces. Es la implementación de la especificación JSR 252 que ofrece Sun MicroSystems y que ha pasado a denominarse Mojarra, anteriormente llamada SUN JSF-RI.

[27] http://www.ecured.cu/index.php/JSF

Tras los resultados de la comparativa efectuada, ha sido elegida como referencia por delante de myFaces debido a su mejor respuesta a las diversas pruebas efectuadas [28]**.**

- **MyFaces** proyecto de Apache Software Foundation: MyFaces es un proyecto de la fundación Apache que ofrece una implementación en código abierto de JavaServer Faces, así como un amplio conjunto de componentes adicionales. Entre ellos se dispone de un menú, árboles, pestañas, componentes para gestionar el estado de los diálogos, etc. (…) [29].

- **RichFaces**: "Rich Faces es un framework de código abierto que añade capacidad Ajax dentro de aplicaciones JSF existentes sin recurrir a JavaScript. Rich Faces incluye ciclo de vida, validaciones, conversores y la gestión de recursos estáticos y dinámicos" [30]**.**

- **ICEFaces**: "Contiene diversos componentes para interfaces de usuarios más enriquecidas, tales como editores de texto enriquecidos, reproductor de multimedia entre otros" [27].

- **jQuery4jsf**: "Contiene diversos componentes sobre la base de uno de los más populares framework javascript jQuery" [27].

- **PrimeFaces:** "PrimeFaces es una librería código abierto para JavaServer Faces. Esta tecnología ofrece un conjunto de componentes ricos para facilitar la creación de aplicaciones web" [31].

[28] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/92
[29] MEDÍN, J. (2006). Hacia una Arquitectura con JavaServer Faces, Spring, Hibernate y otros Frameworks. Ayuntamiento de Madrid. 7p.
[30] http://code.google.com/p/fap-devel/wiki/JavaServerFaces
[31] RUFO, D. (2010). Sede Electrónica. Universidad Rey de Juan Carlos. 10p.

- **OpenFaces:** “Librería open source que contiene diferentes componentes JSF, un Framework Ajax y un Framework de validación por parte del cliente.” [27].

2.3.9 Ventajas

Según MADEJA [32], en su artículo de JavaServer Faces, menciona las siguientes ventajas:

- Una de las grandes ventajas de la tecnología JavaServer Faces es que ofrece una clara separación entre el comportamiento y la presentación. Las aplicaciones web construidas con la utilización de 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 UI como objetos con estado en el servidor. La tecnología JavaServer Faces permite construir aplicaciones web que implementan una separación entre el comportamiento y la presentación tradicionalmente ofrecida por arquitectura UI del lado del cliente.
- 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.
- La tecnología JavaServer Faces trata de mejorar los conceptos familiares de componentes-UI y capa-Web sin limitarnos a una tecnología de scripts particular o un lenguaje de marcas. Aunque la tecnología JavaServer Faces incluye una librería de etiquetas JSP personalizadas para representar componentes en una página JSP, los APIs de la tecnología JavaServer Faces se han creado directamente sobre el API JavaServlet. Esto permite: usar otra tecnología de

[32] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/101

presentación junto a JSP, crear nuestros propios componentes personalizados directamente desde las clases de componentes, y generar salida para diferentes dispositivos cliente.

- JavaServer Faces 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 en JSF al ser el estándar de JAVA.
- Lo más importante es que la tecnología JavaServer Faces proporciona una rica arquitectura para manejar el estado de los componentes, procesar los datos, validar la entrada del usuario, y manejar eventos.

2.3.10 Desventajas

Según MADEJA [32], en su artículo de JavaServer Faces, menciona la siguiente desventaja:

- 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, etc.

Java Server Faces (JSF) es un framework basado en el patrón Modelo Vista Controlador (MVC) para el desarrollo de aplicaciones Web, framework que está constituido por una serie de componentes de interfaz de usuario en el servidor, permitiendo al desarrollado la creación de interfaces de alta calidad con facilidad.

Además JSF es una especificación (JSR 127) aprobada por el Java Community Process (JCP), esto quiere decir que es un estándar y que pueden existir muchas implementaciones amigables y escalables, compatibles con los navegadores Web actuales, entre otras características que exige la especificación antes mencionada.

2.4 ICEFaces

ICEFaces es un framework AJAX de código abierto basado en JSF. “ICEFaces aísla completamente al desarrollador de AJAX. No hacen falta etiquetas especiales: se ponen los controles en la pantalla e ICEFaces se encarga de enviar sólo la información necesaria entre cliente y servidor. Es decir, ya no se envían los formularios a la antigua usanza, en un POST de HTTP, sino que sólo se envían los cambios que ha hecho el usuario del cliente al servidor, y los cambios en la pantalla del servidor al cliente” [33].

En esta sección se definirá el framework ICEFaces, se mencionará sus características, beneficios, compatibilidades e incompatibilidades.

2.4.1 Definición

C. Nadal [34], en su proyecto final de carrera, menciona sobre ICEFaces lo siguiente:

ICEFaces “es un framework integrado de código abierto para aplicaciones Java, que permite a los desarrolladores crear y desplegar fácilmente Aplicaciones de Cliente Enriquecido (Rich Internet Aplication, o más usualmente RIA) mediante el uso de Ajax para plataformas Java EE”.
“Los beneficios de usar Frameworks RIA son básicamente los de enriquecer las aplicaciones web de forma a que funcionalmente sean similares a las aplicaciones de escritorio”.

[33] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/133
[34] http://upcommons.upc.edu/pfc/bitstream/2099.1/9616/1/62016.pdf

Una de las grandes ventajas que proporcionan Ajax es que permite recargar un conjunto de campos definidos por el desarrollador, agilizando la transmisión de datos entre el cliente y el servidor y evitando la recarga de toda la página a cada petición HTTP, que a menudo y según el ancho de banda, puede resultar lento y tedioso. Es útil para validar campos en tiempo real o para disminuir la carga de datos en el servidor. Además ofrece cierta comodidad y velocidad de trabajo (…).

2.4.2 Características

Según MADEJA [35], las características de ICEFaces son:

ICEfaces es considerado un framework que integra funcionalidad AJAX y permite a los desarrolladores Java EE crear aplicaciones RIA (Rich Internet Applications) de una manera sencilla.

Las aplicaciones desarrolladas en ICEfaces no necesitan plugins de navegador o applets para ser vistas.
Estas aplicaciones están basadas en JavaServer Faces (JSF), así que permite el desarrollo de aplicaciones Java EE con la posibilidad de utilizar de forma fácil desarrollos basados en JavaScript.

Entorno a AJAX han surgido varios frameworks (Prototype, DWR, GWT, etc.) que, si bien aportaban facilidad de uso, no acababan de convencer a la comunidad de programadores. Algunos porque sólo eran clientes Javascript, otros porque, si bien integraban la parte de servidor con la de cliente, no eran realmente frameworks, sino librerías de comunicación. Además, no estaba claro cómo juntarlos con la arquitectura JEE.

[35] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/133

Con la llegada de JSF, se empezó a vislumbrar posibilidades de integración. Si JSF permitía al desarrollador aislarse de la arquitectura web y ver sus aplicaciones como algo parecido a una aplicación de escritorio, debería entonces ser sencillo utilizar AJAX para hacer estos controles más funcionales. Y así fue, empezaron a aparecer AJAX4JSF, ICEFaces, Tobago, etc.

Sin embargo, de estas propuestas, ICEFaces fue una de las más acogidas ya que aisla completamente al desarrollador de AJAX. No hacen falta etiquetas especiales: se ponen los controles en la pantalla e ICEFaces se encarga de enviar entre cliente y servidor sólo la información necesaria.

2.4.3 Integración con IDEs

MADEJA [36], menciona que, "ICEfaces aporta integración básica con un alto número de interfaces de desarrollo adaptados por la comunidad de desarrolladores". De los cuales destacan los siguientes: Eclipse (Web Tools Platform + JSF tools), Sun NetBeans, MyEclipse Enterprise Workbench, BEA Workshop Studio 3, Workshop Studio for JSF y Oracle JDeveloper 10g Studio Edition.

2.4.4 Integración con servidores de aplicaciones

MADEJA [36], menciona que, ICEfaces proporciona integración con los siguientes servidores de aplicaciones J2EE: Apache Tomcat, BEA Weblogic Server, JBoss Application Server, IBM Websphere Application Server, Oracle Application Server Container for J2EE (OC4J), SAP NetWeaver, Sun GlassFish, Sun Java System Application Server y Webtide Jetty.

[36] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/103

2.4.5 Compatibilidad con portales y frameworks JEE

MADEJA [36], menciona que, ICEfaces soporta los siguientes portales y frameworks de desarrollo JEE: Liferay Portal 4.3, JBoss Seam 1.3, JavaServer Faces (JSF) 1.1, 1.2, Facelets y Spring Web Flow.

2.4.6 Navegadores soportados

MADEJA [36], señala que, han comprobado la compatibilidad de ICEfaces en los siguientes navegadores: Internet Explorer 6, 7, Firefox 1.x, 2.0, Mozilla 1.7.x and Netscape 7.x, Safari 1.3 y Opera 9.x.

2.4.7 Versiones

MADEJA [36], menciona que: La versión más estable lanzada por la empresa propietaria de ICEfaces (IceSoft) es la versión 1.6 del framework. No obstante, ya se encuentra una nueva versión ICEfaces v1.7.0 con un conjunto de mejoras y arreglos (nuevo popup contextual, google maps, soporte para funciones adicionales...) aunque actualmente se encuentra en fase de pruebas y debe ser considerada no apta para su uso en producción.

2.4.8 Ventajas e inconvenientes

MADEJA [36], señala, que, algunos de los beneficios que presenta ICEFaces con respecto a otros Frameworks son:

- Experiencia de usuario enriquecedora: crea una experiencia de usuario superior además de utilizar las ventajas de aplicaciones Java EE. Esto se consigue gracias a los componentes que vienen incluidos dentro de la distribución de ICEfaces.
- Está basado en código abierto: ICEfaces es un framework basado en Ajax bajo licencia de código abierto. La comunidad de desarrolladores de ICEfaces incluye cerca de 20.000 desarrolladores en 36 países.
- Basado en estándares: ICEfaces es una solución basada en Java, así que los desarrolladores pueden continuar trabajando de la misma forma que lo hacen. Hay multitud de plugins desarrollados para que ICEfaces sea integrado con multitud de IDEs Java.
- El Ajax es transparente: ICEfaces aporta a los programadores un desarrollo con mínimo esfuerzo en la sección JSF.
- Compatibilidad: ICEfaces soporta todos los servidores de aplicaciones, aporta plugins para los distintos IDEs y efectos javascript de librerías de cualquier empresa que haya desarrollado Ajax del mercado.
- Seguridad: ICEfaces es una de las soluciones Ajax más seguras del mercado. Es compatible con SSL, previene los scripts de cross-site, inyección de código malicioso. Es una solución Ajax basada en servidor, la cual no utiliza datos de usuarios, además es especialmente efectivo en la prevención de fallos en los submits de los formularios y el ataque SQL por inyección.

- Escalabilidad y clustering: El servidor asíncrono HTTP (AHS) aporta una alta escalabilidad para aplicaciones ICEfaces y pueden ser utilizadas por un gran número de usuarios concurrentes, además aporta despliegue en clúster (un requisito crítico que algunas soluciones no aportan).
- Carga de páginas incremental con edición de secciones y sin recargas de página completas.
- Se preserva el contexto del usuario durante la actualización de la página, incluyendo posición del foco y scroll.
- En aplicaciones de tiempo real, las recargas de páginas son asíncronas.

En una comparativa con un producto similar, se opta por compararlo con RichFaces. Es la competencia de RichFaces, tiene tantos o más componentes que ésta y visualmente tiene la misma calidad. La documentación posiblemente es peor que la de RichFaces. El modo en que RichFaces hace funcionar Ajax permite interactuar con otras librerías e incluso añadirle Ajax a componentes que no lo tuvieran, mientras que ICEfaces limita a trabajar con sólo aquellos componentes para los que da soporte.

2.5 RichFaces

En el artículo ¿Qué es RichFaces? [37], se menciona una breve historia sobre RichFaces, misma que trata sobre:

RichFaces proviene del marco Ajax4JSF. Fue creado por Alexander Smirnov, quien se unió a Exadel en 2005 con el fin de continuar con el desarrollo. La idea era juntar todas las características de Ajax en el nuevo marco de JavaServer Faces.

[37] http://www.scribd.com/doc/26352343/JBoss-RichFaces-Capitulo-1-%C2%BFQue-es-RichFaces

La primera versión comercial fue lanzada en marzo de 2006 con el nombre de Exadel VCP. En el mismo año, se dividió en dos proyectos Ajax4Jsf (open source) y RichFaces (versión comercial).

Ajax4jsf proporcionan el marco básico y los componentes Ajax para "ajaxizar" los componentes JSF en una página (soporte Ajax en las páginas JSF). RichFaces fue un componente JSF comercial para las bibliotecas Ajax.

En marzo de 2007, Exadel y JBoss (una división de RedHat) anunciaron una alianza para abrir el código fuente de RichFaces y los dos proyectos se fusionaron en un único proyecto de fuente abierta llamado simplemente "RichFaces". Fue una buena decisión para resolver los problemas de compatibilidad con la versión que tenían los dos proyectos por separado.

Hoy en día, el proyecto se está moviendo muy rápido con la ayuda de la comunidad de usuarios que participan con el equipo RichFaces que también deciden la evolución futura que el marco tendrá (como que característica se desarrollará más, que componente nuevo se debe de hacer en primer lugar, etc.)**.**

En esta sección se definirá el framework RichFaces, se mencionará su arquitectura, características, ventajas e inconvenientes, compatibilidades e incompatibilidades.

2.5.1 Definición

En RichFaces Developer Guide [38], se señala sobre RichFaces lo siguiente:

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.

MADEJA [39], menciona las siguientes características que RichFaces permite:

- Intensificar el conjunto de beneficios JSF al trabajar con Ajax. RichFaces está completamente integrado en el ciclo de vida de JSF. Mientras que otros marcos sólo dan acceso a los managed bean, Rich Faces permite acceder al action y al valor del listener, así como invocar a validadores y convertidores durante el ciclo de petición-respuesta de Ajax.
- Añadir capacidad Ajax a aplicaciones JSF. El framework proporciona dos librerías de componentes (Core Ajax y la interfaz de usuario). La librería Core nos permite agregar la funcionalidad Ajax en las páginas que queramos sin necesidad de escribir nada de código JavaScript. RichFaces permite definir eventos en la propia página. Un evento invoca a una petición Ajax, sincronizándose así zonas de la página y componentes JSF después de recibir la respuesta del servidor por Ajax.

[38] http://www.scribd.com/doc/36023226/Richfaces-Reference
[39] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/134

- Crear rápidamente vistas complejas basándose en la caja de componentes. La librería UI (Interfaz de usuario) que contiene componentes para agregar características de interfaz de usuario a aplicaciones JSF. Se amplía el framework de RichFaces incluyendo un gran conjunto de componentes “habilitación de Ajax” que extiende el soporte de la página. Además, los componentes de RichFaces están diseñados para ser usados sin problemas con otras librerías de componentes en la misma página, de modo que existen más opciones para el desarrollo de aplicaciones.
- Escribir componentes propios con función soportada por Ajax. El CDK o Kit de Desarrollo de Componentes basado en maven, incluye un generador de código para plantillas JSP utilizando una sintaxis similar. Estas capacidades ayudan a evitar un proceso de rutina de un componente de creación.
- Proporciona un paquete de recursos con clases de aplicación Java. Además de su núcleo, la funcionalidad de RichFaces para Ajax proporciona un avanzado soporte a la gestión de diferentes recursos: imágenes, código JavaScript y hojas de estilo CSS. El framework de recursos hace posible empaquetar fácilmente estos recursos en archivos jar junto con el código de los componentes personalizados.
- Generar fácilmente recursos binarios sobre la marcha. Los recursos del framework pueden generar imágenes, sonidos, hojas de cálculo de Excel, etc.
- Crear una moderna interfaz de usuario 'look-and-feel' basadas en tecnología de skins. RichFaces proporciona una función que permite definir y administrar fácilmente diferentes esquemas de color y otros parámetros de la interfaz de usuario, con la ayuda de los parámetros del skin. Por lo tanto, es posible acceder a los parámetros del skin desde el código JSP y el código de Java (por ejemplo, para ajustar las imágenes generadas sobre la marcha basadas en la interfaz de usuario). RichFaces viene con una serie de skins predefinidas para empezar, pero también se pueden crear fácilmente skins propios.

2.5.2 Características

M. Sánchez (2010) [40], menciona las siguientes características de RichFaces:

- Se integra perfectamente en el ciclo de vida de JSF.
- Incluye funcionalidades Ajax, de modo que nunca vemos el JavaScript y tiene un contenedor Ajax propio.
- Contiene un set de componentes visuales, los más comunes para el desarrollo de una aplicación RIA, con un número bastante amplio que cubren casi todas nuestras necesidades.
- Soporta Facelets.
- Soporta CSS, themes y skins.
- Es un proyecto open source, activo y con una comunidad también activa.

2.5.3 Versiones de Java soportadas

Según MADEJA [41], la versión de Java soportada es JDK 1.5 y superiores.

2.5.4 Implementaciones de JavaServer Faces soportadas

Según MADEJA [41], las implementaciones de JavaServer Faces que soporta RichFaces son: Sun JSF 1.1 RI, MyFaces 1.1.1 - 1.2, Facelets JSF 1.1.1 - 1.2 y Seam 1.2. - 2.0.

[40] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro
[41] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/85

2.5.5 Servidores soportados

Según MADEJA [41], RichFaces proporciona integración con los siguientes servidores: Apache Tomcat 4.1 - 6.0, IBM WebSphere 5.1 - 6.0, BEA WebLogic 8.1 - 9.0, Oracle AS/OC4J 10.1.3, Resin 3.0, Jetty 5.1.X, Sun Application Server 8 (J2EE 1.4), Glassfish (J2EE 5), JBoss 3.2 - 4.2.x y Sybase EAServer 6.0.1.

2.5.6 Navegadores soportados

Según MADEJA [41], RichFaces es compatible con los siguientes navegadores: Internet Explorer 6.0 - 7.0, Firefox 1.5 - 2.0, Opera 8.5 - 9.0, Netscape 7.0 y Safari 2.0.

2.5.7 Ventajas e inconvenientes

MADEJA [41], señala algunas de las ventajas que aporta la utilización de RichFaces:

- Al pertenecer RichFaces a un subproyecto de JBoss, su integración con Seam es perfecta.
- Al ser RichFaces ex propiedad de Exadel, se ajusta perfectamente al IDE Red Hat Developer Studio (permite desarrollar aplicaciones visuales con RichFaces de forma fácil).

Inconvenientes detectados:

- No se puede realizar una aplicación combinándolo con IceFaces y Seam.

Si no se va a utilizar RichFaces con Seam, y no es demasiado importante el tema de AJAX, IceFaces es interesante, porque trae un set de componentes mucho más rico que el de RichFaces. IceFaces tiene actualmente una buena integración con Seam, pero no es tan flexible como Ajax4Jsf+RichFaces a la hora de trabajar con AJAX.

Los componentes de RichFaces no tienen soporte natural de AJAX, para esto se debe añadir un componente de Ajax4JSF, lo que permite dar más control a las partes de la página que se desea ajaxizar. En cambio con ICEFaces es distinto, el desarrollador se despreocupa del tema de AJAX, pues éste ya viene incluido en sus componentes.

2.6 Apache MyFaces

Apache MyFaces "es un proyecto de Apache Software Foundation, y alberga varios sub proyectos relacionados con JavaServer Faces" [42].

El proyecto de Apache MyFaces provee [42]:

- Una implementación JavaServer Faces (JSF 1.x y JSF 2.x) conocida como MyFaces Core.
- Varias librerías de componentes que contienen widgest de UI para la construcción de aplicaciones Web con JSF (MyFaces Trinidad, MyFaces Tobago y MyFaces Tomahawk).
- Paquetes de extensión para JavaServer Faces (MyFaces Test, MyFaces Orchestra, MyFaces Extension Validator, MyFaces Extensions Scripting y MyFaces Extension CDI).
- Integración de módulos para otras tecnologías y estándares (MyFaces Portlet Bridge para la integración con el estándar porlet).

[42] https://cwiki.apache.org/confluence/display/MYFACES/Index

El proyecto se divide en cinco componentes principales [43]:

- **Core**: Es una implementación de JSF 1.1 y 1.2 y los componentes que se han especificado en JSR (Java Specification Request) 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, también conocidos como ADF Faces.
- **Tobago**: Un conjunto de componentes contribuidos a MyFaces por Atanion GmbH.
- **Orchestra**: Un framework usado para el control de la persistencia entre sesiones.

2.6.1 MyFaces Trinidad

D. Thomas (2009:8:10) [44], menciona lo siguiente sobre MyFaces Trinidad:

Trinidad se remonta al desarrollo de código desde principio del año 2001. Oracle se encaminó al desarrollo de un framework Web basado en componentes llamado UIX que se convirtió en una librería de componentes para aplicaciones JSF y posteriormente estos mismos en Frameworks JSF. En un inicio la decisión de Oracle en invertir en JSF cuando su futuro era incierto fue una apuesta que valió la pena. Hoy en día, JSF es tecnología esencial cuando se trata implementar aplicaciones Web a mediana y gran escala.

[43] http://www.datuopinion.com/apache-myfaces
[44] THOMAS, D. (2009), Apache MyFaces Trinidad 1.2, p.8:10.

La próxima decisión de Oracle, basado en su compromiso con la comunidad de código libre Java, fue donar este framework, originalmente conocido como Oracle ADF Faces a la Apache Software Fundation en el año 2006. Una parte del equipo de desarrollo se mantiene en Oracle, lo que garantiza la consistencia del diseño. Oracle y Apache se concentraron en intercambiar la fuente desde la dependencia para asegurar que el código conste solamente de código abierto. De la mano, el desarrollo de Oracle ADF Faces todavía continúa, pero ahora se basa en Trinidad.

El 5 de mayo del 2007, Trinidad dejó la incubadora de Apache MyFaces y se unió como sub proyecto. Apache MyFaces es un sub proyecto de Apache Software Foundation que es responsable de la implementación de código abierto JSF con el mismo nombre se incluye una serie de sub proyectos como Trinidad.

A finales de junio, Trinidad ve su primer lanzamiento 1.0.1 soportando JSF. El 5 de julio, Trinidad 1.2.1, la versión para JSF 1.2, fue puesta en libertad. A finales del 2007, Trinidad empezó a convertirse en una librería de componentes con un ciclo frecuente de lanzamientos, aproximadamente cada dos meses como promedio.

2.6.1.1 Descripción

Trinidad “no es solamente una librería de componentes de JSF. Esta provee al desarrollador un moderno framework Web que se centra en amplitud y filosofía de diseño en el mundo cerrado de software” [44].

D. Thomas (2009:9) [44], señala que, Trinidad tiene dos ventajas generales:

- El usuario no necesita combinar varias librerías de componentes. Esto evita problemas de integración, debido al número elevado de componentes que incluye Trinidad.
- Un alto grado de consistencia es recomendado. Como consecuencia, las cosas similares se hacen de forma similar, los atributos similares tienen una nomenclatura consistente, los efectos colaterales de otras fuentes de código son mínimas, el crecimiento de error es mínimo, etc.

2.6.1.2 Características

D. Thomas (2009:9:10) [44], menciona lo siguiente sobre las características de Trinidad:

Concretamente hablando, las siguientes características de Trinidad ilustran el enfoque en amplitud y en el diseño de software de un mundo cerrado:

- Renderizado Parcial de Página (PPR): La tecnología JSF-Ajax integrado es parte prácticamente de todas las etiquetas de Trinidad.
- Nomenclatura de atributos consistentes, incluyendo conjuntos de atributos periódicos en todas las etiquetas del universo de Trinidad (con diferentes niveles frecuentes de concurrencia).
- Un gran número de etiquetas de JSF está disponible como una versión de Trinidad, con mejoras enfocadas en tecnologías específicas de Trinidad, por ejemplo, PPR, error o soporte de mensajería de información, tooltips, labels, diseño, etc.
- Framework para diálogos de pantalla: el framework de diálogos permite trabajar a la aplicación Web con ventanas emergentes.

- Trinidad también posee sus propios ámbitos, por ejemplo, “pageFloweScope” para comunicación entre páginas.

2.6.2 MyFaces Tobago

En la página oficial de Tobago [45], se menciona las siguientes afirmaciones que caracterizan a Tobago y lo hacen diferente de otros Frameworks:

- La finalidad de Tobago es crear aplicaciones de negocio sin la necesidad de un diseño HTML. El desarrollo de páginas con Tobago se parece más al diseño de interfaces de usuarios convencionales que al diseño de páginas webs.
- Los componentes de la Interfaz de Usuario son abstracciones del HTML y cualquier diseño de información que haga, no pertenece a la estructura general de la página. El formato final de salida es determinado por el cliente.
- Un mecanismo basado en temas, hace fácil cambiar la apariencia y permite proveer implementaciones especiales para ciertos navegadores. Una solución basada en la readaptación ante fallos nos asegura la reusabilidad de gran parte del código para nuevos temas.
- Se usa un gestor de diseño para organizar los componentes de manera automática. Esto quiere decir, que no hace falta un diseño manual con HTML para tablas u otros componentes.

[45] http://myfaces.apache.org/tobago/index.html

2.6.2.1 Ventajas e Inconvenientes

MADEJA [46], señala lo siguiente sobre las ventajas e inconvenientes de Tobago:

Tobago forma parte del proyecto MyFaces de Apache, que ofrece una implementación en código abierto de JSF, así como un amplio conjunto de componentes adicionales. Entre ellos se dispone de un menú, árboles, pestañas, componentes para gestionar el estado de los diálogos, etc. Algunos de estos componentes están implementados haciendo uso de Ajax. A lo largo del tiempo se van incorporando componentes cada vez más potentes y ya se encuentran en pruebas componentes basados en Ajax.

Un gran inconveniente que plantea el uso de Tobago es la deficiente documentación de la que se dispone sobre el framework

2.6.2.2 Entornos

MADEJA [46], presenta una relación de entornos con los que trabaja Tobago:

- Java: JDK 1.4.x, JDK 1.5.x
- Servlet Container: Tomcat 4.x, Tomcat 5.x, Jrun 4 (SP1a), JBoss 3.2.x, Jboss 4.0.x, BEA Weblogic 8.1, Jonas 3.3.6 w/Tomcat, Resin 2.1.x, Jetty 4.2.x, Jetty 5.1.x y Websphere 5.1.2, OC4J
- Custom JSF componentes: ADF Faces de Oracle.

[46] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/88

2.6.3 MyFaces Tomahawk

Apache Tomahawk, es un conjunto de componentes JSF que van más allá de la especificación JSF. Son totalmente compatibles con la Implementación de Referencia de SUN (SUN IR) versión 1.1 así como con cualquier otra implementación compatible con la versión 1.1 de la especificación JSF. Por supuesto, es totalmente compatible con la implementación de Apache MyFaces ya que es un sub proyecto de MyFaces [47].

2.6.3.1 Características

MADEJA [47], señala lo siguiente sobre las características de Tomahawk:

Todos los componentes estándar tienen una versión equivalente en Tomahawk. Estas versiones extendidas se caracterizan por mejorar el rendimiento de los componentes estándar y por tener nuevos atributos que le proporcionan características adicionales entre las que se pueden destacar:

- Conciencia de Rol de Usuario: El renderizador permite ver al usuario ciertos componentes en función de su rol en la aplicación.
- Mostrar Sólo el Valor: Capacidad de intercambiar el estado de los componentes entre modo de entrada y de salida.
- Forzar Id: No permite a JSF generar un identificador para el atributo id del componente y de sus padres sino que en su lugar utiliza uno proporcionado por el desarrollador.

[47] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/137

2.6.4 MyFaces Orchestra

Orchestra es una pequeña librería que se puede utilizar en aplicaciones Web para proporcionar las siguientes características [48]:

- Provee alcance a los componentes de respaldo (backing beans).
- Diálogo en el ámbito de contextos de persistencia.
- Anotaciones para definición de transacciones declarativas (solamente java 1.5).
- Un componente JSF "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 (es decir están fuertemente acoplados con una base de datos). En particular, el componente "dynaForm" (que requiere otras partes de Orchestra) hace que sea fácil programar el ingreso de datos que se especializan de Oracle Forms o Microsoft Access.
Orchestra actualmente soporta JSF 1.1, JSF 1.2 – JSF 2.0, pero el soporte debería ser posible para otros Frameworks Web que se presenten en el futuro.

Apache MyFaces Orchestra es un miembro relativamente nuevo en la familia del proyecto Apache MyFaces, sin embargo se ha utilizado en varios proyectos del mundo real.

Desde la versión 1.4, Orchestra es compatible con JDK 1.5.

[48] http://myfaces.apache.org/orchestra/index.html

2.6.4.1 Características

Según la página oficial de MyFaces Orchestra [48], las características destacadas de Orchestra son:

- Trabaja con una sintaxis compatible con Java 1.5, y opcionalmente se puede utilizar anotaciones.
- Utiliza el poderos mecanismo de configuración Spring en lugar del maged-bean que se instala con JSF. El lanzamiento de Spring 2.0 hizo posible definir un bean personalizado en Spring. Si el managed bean de JSF es declarado en Spring usando el ámbito de conversación Orchestra, entonces cuando es referenciado desde JSF este es creado automáticamente dentro del ámbito de conversación. Es decir, 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 generalmente en cualquier implementación JPA). Sin embargo, cualquier framework de persistencia puede ser conectado con Orchestra.
- El API de Orchestra puede ser adaptado para usarse con otros Frameworks Web de JSF.

2.6.4.2 Limitaciones

La página oficial de Orchestra [48], menciona sobre las limitaciones lo siguiente:

La característica de persistencia de Orchestra asume que la capa de presentación tiene acceso a la Base de Datos, es decir, que las capas de presentación y base de datos se combinan. Esto sucede a menudo en aplicaciones de mediana y gran escala. Esto puede llegar a ser una mala práctica en el

desarrollo de grandes aplicaciones empresariales, pero puede ser de gran ayuda en aplicaciones pequeñas.

Orchestra no es compatible con los porlets.

Orchestra no soporta “sesiones distribuidas”, es decir, las configuraciones donde las sesiones HTTP serán almacenadas y enviadas a otras máquinas en un clúster.

MyFaces es un desarrollo de código abierto creado por la fundación de Apache, que ofrece una gama de implementaciones bajo Java Server Faces, así como un conjunto de componentes adicionales, que permite programar aplicaciones Web con mayor funcionalidad y con más orientación al usuario.

MyFaces es un marco de desarrollo de aplicaciones web que siguen el patrón Modelo Vista Controlador (MVC), con la diferencia a otros Frameworks que éste tiene una fuerte orientación a considerar los componentes de la aplicación, botones, cajas de texto, tendiendo a asimilarse a los programas de escritorio.

C A P Í T U L O I I I

ANÁLISIS COMPARATIVO DE LOS FRAMEWORKS MYFACES, ICEFACES Y RICHFACES

Hoy en día se puede encontrar una variedad de frameworks JSF en el mercado, de los cuales algunos son gratuitos y otros son pagados, todos con buenas características, es por tal motivo que la elección de un framework que cumpla con las necesidades y requerimientos que un desarrollador de software desea es de vital importancia y por tanto su selección una tarea difícil.

Entre los frameworks JSF comúnmente usados por la comunidad J2EE se tiene MyFaces, IceFaces y RichFaces, cabe mencionar que entre los sub proyectos que alberga Apache MyFaces se ha seleccionado a MyFaces Trinidad.

La elección de utilizar Trinidad entre Tomahawk y Tobago se lo realizó con la ayuda de la Matriz de JSF [49]**,** por la cantidad de características que presenta a su favor y por el parámetro cantidad de documentación. Esta matriz se encuentra constantemente actualizada por los miembros de la comunidad de desarrolladores J2EE, mismos que se han preocupado por seleccionar el framework

[49] http://www.jsfmatrix.net/

más adecuado para la implementación de sus aplicaciones, en ésta se puede observar que las características de Trinidad sobre Tomahawk y Tobago sobresalen.

En esta sección se efectúa un análisis comparativo entre los Frameworks MyFaces Trinidad, IceFaces y RichFaces, con la finalidad de seleccionar el más eficiente para el desarrollador en la realización del Sistema Nutricional de la ESPOCH.

3.1 DEFINICIÓN DE LOS PARÁMETROS DE COMPARACIÓN

Los factores que intervienen para la selección del framework más eficiente en el desarrollo de la aplicación entre uno y otro, han sido considerados por la determinación de disponibilidad de información, por la calidad que estos ofrecen a través del seguimiento de problemas, por la cantidad de componentes que proveen y por la facilidad de desarrollo.

Los parámetros estimados para el análisis de los frameworks y el porcentaje que cada uno de ellos aportará para el total acumulado del framework se resumen en la **Tabla III. I**, el detalle de cada parámetro se define de manera inmediata a la tabla de resumen.

Tabla III. I. Resumen de Parámetros de Análisis y su Porcentaje

Parámetros	Criterios	Porcentaje (%)	Umbral
Aprendizaje	Disponibilidad de información Soporte	10%	1
Calidad	Calidad	10%	1
Componentes de interfaz de usuario	Disponibilidad de componentes Componentes utilizables	30%	3
Facilidad para el desarrollo	Facilidad de instalación Facilidad para iniciar Facilidad de uso Soporte Ajax Líneas de código	50%	5
TOTAL		100%	10 pts.

- **Aprendizaje:** Determina la simplicidad o complejidad de obtener un conocimiento aceptable del uso de componentes de cada uno de los frameworks. Con este parámetro se considera qué framework sería más simple y fácil de abordar en una primera etapa, misma que se denomina como aprendizaje.

  El aprendizaje se evaluará en base a los siguientes criterios:

  Disponibilidad de información: Permite valorar la existencia de información y conocimiento que se puede conseguir de cada framework por medios accesibles, como Internet, libros y manuales.

  Soporte: Criterio en la cual se toma en cuenta las responsabilidades que cada organización a cargo del desarrollo del framework tiene para dar soporte, como por ejemplo bugs, ayuda en línea o foros, de tal manera que permitan al desarrollador una mejor y fácil integración con el framework.

- **Calidad:** La calidad de un producto puede evaluarse desde diferentes puntos de vista, en el presente análisis la calidad se valorará a través de los seguimientos a problemas abiertos [50] que cada framework proporciona. Los seguimientos de errores permiten asegurar la calidad del producto y por tanto asisten a los desarrolladores y demás personas involucradas en el desarrollo, en el seguimiento de defectos y mejoras realizadas al framework.

[50] http://www.clearminds-it.com/index.php?/JSF/primefaces-vs-richfaces-vs-icefaces.html

- **Componentes de interfaz de usuario:** Los controles de interfaz de usuario son importantes en la programación, pues éstos permiten agilizar el proceso de diseño y desarrollo durante la creación de una aplicación Web.

  Los componentes de interfaz de usuario se evaluarán en base a los siguientes criterios:

  Disponibilidad de componentes [50]***:*** Permite determinar la robustez que los componentes de los frameworks proporcionan para el desarrollo de la aplicación Web.

  Componentes utilizables: Este criterio establece la disponibilidad de componentes a utilizar de acuerdo a los requerimientos previamente analizados para la aplicación del Sistema Nutricional de la ESPOCH.

- **Facilidad para el desarrollo:** Este parámetro se refiere a la facilidad e inversión de tiempo necesarios que utilizará el desarrollador para obtener un conocimiento aceptable durante la instalación e iniciación del framework, así como las funcionalidades que el entorno de desarrollo proporciona durante la utilización de los componentes.

  La facilidad para el desarrollo se evaluará en base a los siguientes criterios:

  Facilidad de instalación: Se refiere a la capacidad del framework para ser instalado en el ambiente de desarrollo.

  Facilidad para iniciar [50]***:*** Criterio que busca determinar la facilidad de configuración que cada framework presta al desarrollador para empezar a trabajar con el mismo.

Facilidad de uso: Este criterio se refiere al grado en que la interfaz, la distribución de los paneles y la forma en que los componentes facilitan o dificultan su manejo.

Soporte Ajax [51]***:*** Permite valorar la facilidad y eficiencia que prestan los componentes de cada framework al momento de trabajar de manera asíncrona entre el cliente y servidor.

Líneas de código: Este criterio permite obtener de forma práctica el framework que ocupa el menor número de líneas de código en la vista o presentación al usuario; mientras menos número de líneas de código mejor, porque el mantenimiento resulta fácil y además ahorra tiempo al desarrollador.

3.2 CRITERIOS DE EVALUACIÓN

En esta sección se establece una escala para determina los porcentajes que permitirá seleccionar el framework adecuado para el desarrollo de la aplicación.

En la **Tabla III. II**, se ilustra las posibles interpretaciones que se les puede otorgar a los frameworks.

Tabla III. II. Pesos de Valoración Cualitativa - Cuantitativa

Cuantitativa	**0**	**1**	**2**	**3**
Porcentaje (%)	< 25	>= 25 y < 50	>= 50 y < 75	>= 75
Cualitativa	Malo	Regular	Bueno	Muy bueno
	Muy difícil	Difícil	Fácil	Muy fácil
	Muy poco	Algo	Bastante	Mucho
	Si			No

[51] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro

3.3 ANÁLISIS COMPARATIVO DE LOS FRAMEWORKS MYFACES, ICEFACES Y RICHFACES

El análisis comparativo de los frameworks MyFaces Trinidad, IceFaces y RichFaces, que se desarrollará a continuación, será a través de la observación, experimentación, encuesta realizada a los desarrolladores del DESITEL (ver Anexo B – Sección 1) y de la recopilación de información que se obtenga de revisiones bibliográficas de textos, documentos y publicaciones.

3.3.1 Aprendizaje

Para determinar la simplicidad o complejidad de obtener conocimientos acerca del uso de los componentes de los frameworks, se analizará la documentación y soporte proporcionado por los distintos medios de información al desarrollador, por ende en la **Tabla III. III**, se definen los criterios de comparación con sus respectivas valoraciones, tomando en cuenta que a este parámetro se le ha otorgado un peso de 1 punto sobre 10 (1/10).

Tabla III. III. Criterios y Valoración del parámetro Aprendizaje

Criterios	Valoración	Porcentaje (%)
Disponibilidad de información	0,5	50%
Soporte	0,5	50%
TOTAL	1	100%

a) Disponibilidad de información

Para valorar la existencia de información y conocimiento se utilizará medios como: Internet, manuales, documentos y documentación oficial.

Internet: La búsqueda de información disponible sobre conceptos de MyFaces Trinidad, IceFaces y RichFaces se obtuvo mediante el buscador Web más utilizado, Google. Primeramente se buscó el concepto limitando las páginas solo en español y posteriormente se realizó la búsqueda en la web, tomando como referencia la suma de todos los promedios para la obtención de los resultados.

El máximo valor es 1.216.500 (ver Anexo A – Sección 1) el cual corresponde al 50% de información disponible en el Internet de RichFaces, por lo tanto para obtener el valor de IceFaces y MyFaces Trinidad se realizó una regla de 3 (ver Anexo A – Sección 1).

Manuales y documentos: Se efectuó una búsqueda de libros y temas afines en Amazon [52]**,** "página número 1 de ventas a través del Internet" [53]; se limitó el idioma con la finalidad de obtener resultados en inglés (ver Anexo A – Sección 1).

El máximo valor es 10,67 el cual corresponde al 50% de manuales y documentos de RichFaces, por lo tanto para obtener el valor de MyFaces Trinidad e IceFaces se realizó una regla de 3 (ver anexo A – Sección 1).

Documentación oficial**:** La documentación oficial se ha analizado obteniendo un total de los resultados encontrados en los enlaces oficiales de cada framework.

El máximo valor es 12 que pertenece al framework IceFaces, este puntaje es asignado como un cumplimiento total que corresponde al 50%, por lo tanto para determinar los valores de MyFaces Trinidad y RichFaces se realizó un regla de 3 (ver Anexo A – Sección 1).

[52] https://www.amazon.com
[53] http://diarioadn.co/actualidad/tecnología/páginas-para-realizar-compras-en-internet-1.39708

Tomando en cuenta todos los valores obtenidos (ver Anexo A – Sección 1), el resultado final de documentación disponible se presenta en la **Tabla III. IV**.

Tabla III. IV. Análisis del Criterio Disponibilidad de Información

	MyFaces Trinidad	IceFaces	RichFaces
Disponibilidad de información	21,65%	26,93%	40,28%

b) Soporte

Se tomó como dato los enlaces existentes en los sitios oficiales sobre el soporte dado a cada uno de los frameworks.

El máximo valor obtenido es 9 (ver Anexo A – Sección 2) el cual corresponde al 50% de enlaces existentes para el soporte del framework IceFaces, por lo tanto para conseguir el valor de MyFaces Trinidad y RichFaces se realizó una regla de 3 (ver Anexo A – Sección 1).

Considerando la valoración obtenida, el resultado de soporte para cada framework se muestra en la **Tabla III. V.**

Tabla III. V. Análisis del Criterio Soporte

	MyFaces Trinidad	IceFaces	RichFaces
Soporte	16,67%	50%	33,33%

Como resultado final del parámetro aprendizaje, en la **Tabla III. VI** se ilustra los resultados de cada framework analizado en el parámetro aprendizaje.

Tabla III. VI. Análisis del Parámetro Aprendizaje

	MyFaces Trinidad	IceFaces	RichFaces
Disponibilidad de información	21,65%	26,93%	40,28%
Soporte	16,67%	50,00%	33,33%
TOTAL	38,32%	76,93%	73,61%

RichFaces ofrece documentación propia que se puede encontrar en línea, misma que es actualizada constantemente liberación tras liberación. Uno de los grandes problemas que existe con RichFaces es que no existe documentación adicional que haga referencia a la creación de aplicaciones.

IceFaces ofrece una documentación muy amplia, misma que se puede encontrar en su página oficial o en el internet, sobre este framework se consigue guías, tutoriales, video tutoriales y ejemplos, facilitando en gran medida al desarrollador en el aprendizaje.

MyFaces Trinidad lamentablemente ofrece muy poca documentación disponible, esto complica significativamente en la facilidad para el aprendizaje.

La mayoría de información disponible de los frameworks se encuentra en inglés, siendo éste el caso de manuales, documentos y guías de ayuda, proporcionada por cada uno de los frameworks analizados o desarrollada por terceros, pues este es una gran limitante, ya que al no encontrarse en el idioma del desarrollador, el proceso de aprendizaje se complica, al tener que dedicar tiempo a traducir y aprender.

Como se observa en la **Figura III. 5**, IceFaces supera a MyFaces Trinidad y RichFaces. IceFaces y RichFaces presentan casi un mismo nivel de aprendizaje, con la salvedad que la documentación disponible de RichFaces supera a IceFaces, y el soporte que ofrece IceFaces es mayor a RichFaces.

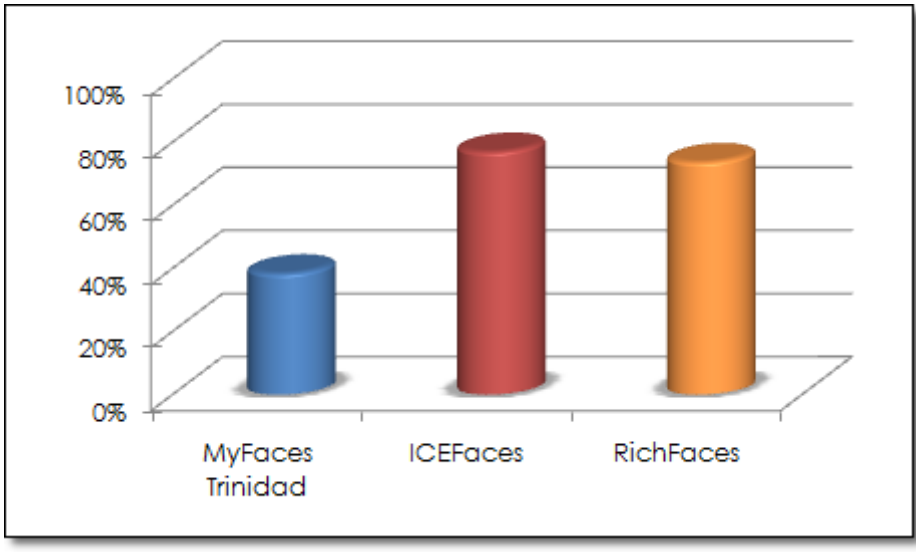


Figura III. 5. Análisis del Parámetro Aprendizaje

3.3.2 Calidad

Los seguimientos de errores permiten asegurar la calidad del producto y asistir a los desarrolladores, por tal motivo, para analizar este parámetro se tomará como dato los resultados que la aplicación JIRA proporciona. JIRA es una aplicación basada en Web que se creó inicialmente para el desarrollo de software, gestión de requisitos, seguimiento de status y posteriormente seguimiento de errores.

Para el análisis de este parámetro se debe considerar que se le ha asignado un peso de 1 sobre 10 puntos (1/10).

Del seguimiento de problemas de cada framework (ver Anexo A – Sección 3), se obtuvo como resultado la sumatoria total de 4.288 problemas sin resolver, considerando que en este parámetro cumple en su totalidad MyFaces Trinidad, con el mínimo número de problemas sin resolver, cantidad que corresponde a 607.

Para obtener los resultados deseados se resta del total de problemas sin resolver (4.288) cada uno de los totales resultantes de cada framework (ver Anexo A – Sección 3).

El mayor valor tomado es 3.681, mismo que corresponde al 100% del parámetro calidad para el framework MyFaces Trinidad, y para determinar el valor de IceFaces y RichFaces se aplica una regla de tres (ver Anexo A – Sección 3).

Los resultados finales se reflejan en la **Tabla III.VII**.

Tabla III. VII. Análisis del Parámetro Calidad

	MyFaces Trinidad	IceFaces	RichFaces
Calidad	100%	70,8%	62,18%

En la **Figura III. 6**, se observa que MyFaces supera a IceFaces y RichFaces en lo que se refiere al parámetro calidad.

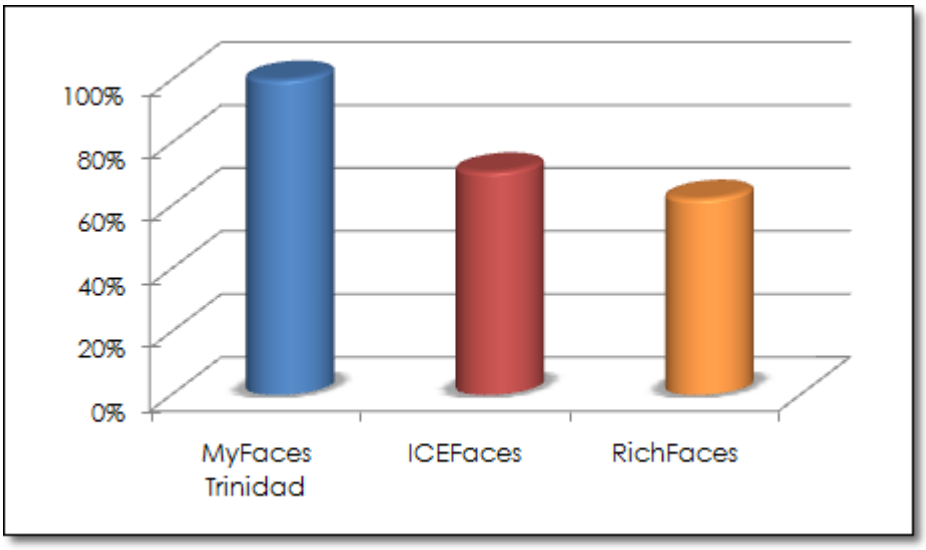


Figura III. 6. Análisis del Parámetro Calidad

3.3.3 Componentes de interfaz de usuario

Los componentes de usuario son un factor importante para agilizar el desarrollo de la aplicación, por esta razón, en este parámetro se analizará tanto los componentes disponibles como los componentes utilizables. A los componentes de interfaz de usuario se le ha asignado un peso 3 puntos sobre 10 (3/10).

Para los componentes de usuario, **Tabla III. VIII**, se deben tomar en cuenta los siguientes criterios con sus respectivas valoraciones:

Tabla III. VIII. Criterios y Valoración del Parámetro Componentes de Interfaz de Usuario

Criterios	Valoración	Porcentaje (%)
Disponibilidad de componentes	1	33,33%
Componentes utilizables	2	66,67%
TOTAL	3	100%

a) Disponibilidad de componentes

La disponibilidad de componentes permite determinar la robustez que los frameworks proporcionan, por ende, para analizar este criterio se tomó como referencia los demos y guías de los sitios oficiales de cada uno de los frameworks estudiados.

El máximo valor obtenido es 116 el cual corresponde al 33,33% de disponibilidad de componentes para el framework IceFaces, por lo tanto para obtener el valor de MyFaces Trinidad y RichFaces se realizó una regla de 3 (ver Anexo A – Sección 4).

Los valores cuantitativos asignados al criterio disponibilidad de componentes se resume en la **Tabla III. IX**.

Tabla III. IX. Análisis del Criterio Disponibilidad de Componentes

	MyFaces Trinidad	IceFaces	RichFaces
Disponibilidad de componentes	21,26%	33,33%	21,55%

b) Componentes utilizables

Los componentes utilizables se analizaron en base a los requerimientos previamente establecidos por la Escuela de Nutrición y Dietética, de los cuales se pudo distinguir que la aplicación necesita para su correcto funcionamiento componentes tales como: ventanas emergentes, menús de opciones, combos seleccionables, calendarios, componentes para subir archivos, paneles de pestañas, paneles de agrupación, paneles divisibles, gráficas lineales y listas paginadas.

Para determinar el framework que facilita los componentes antes mencionados, se utilizaron los recursos proporcionados por los sitios web oficiales de los frameworks analizados y la programación en el IDE NetBeans 7.1.1 para el desarrollo de los posibles componentes a utilizar en la implementación del Sistema Nutricional de la ESPOCH (ver Anexo A – Sección 5). Las aplicaciones de ejemplo demuestran las características de los componentes que conforman los frameworks MyFaces Trinidad, IceFaces y RichFaces.

En base a la tabla resumida (Ver Anexo A – Sección 5) se observa que los frameworks MyFaces Trinidad e IceFaces tienen disponibles la misma cantidad de componentes utilizables, faltándole a RichFaces el componente Gráfica 2D, es por tal razón, que en la encuesta realizada se incluyó una pregunta relacionada con el componente antes mencionado (ver Anexo B – Sección 1).

En base a la tabulación realizada (ver Anexo B – Sección 2), se tomaron en cuenta los valores finales de la pregunta número cinco, que corresponde al conocimiento de la existencia del componente Gráfica 2D. El máximo valor obtenido es 15 que pertenece a MyFaces Trinidad e IceFaces, asignándoles la valoración total del criterio que pertenece al 66,67%, RichFaces obtuvo 0

puntos, por tal motivo el total de los valores asignados para el criterio componentes utilizables se muestra en la **Tabla III. X.**

Tabla III. X. Análisis del Criterio Componentes Utilizables

	MyFaces Trinidad	IceFaces	RichFaces
Componentes utilizables	66,67%	66,67%	0%

Tomando en cuenta el análisis de la disponibilidad de componentes y los componentes utilizables, como resultado final se ilustra en la **Tabla III. XI** los valores cuantitativos obtenidos por cada uno de los frameworks MyFaces Trinidad, IceFaces y RichFaces en el parámetro componentes de interfaz de usuario

Tabla III. XI. Análisis del Parámetro Componentes de Interfaz de Usuario

	MyFaces Trinidad	IceFaces	RichFaces
Disponibilidad de componentes	21,26%	33,33%	21,55%
Componentes utilizables	66,67%	66,67%	0,00%
TOTAL	87,93%	100,00%	21,55%

MyFaces Trinidad posee muchos componentes que permiten crear aplicaciones ricas con mucha facilidad, pero lamentablemente la poca documentación deja bastante que desear.

Los componentes ACE (Advanced Component Environment) de IceFaces, son controles de usuario mejorados y perfeccionados, que permiten realizar tareas comunes y además prestan al desarrollador la funcionalidad de utilizar directamente JavaScript, sin la necesidad de tener conocimientos previos, e incluso no se requiere aprender JavaScript.

RichFaces provee dos conjuntos de librerías: Core Ajax y UI. Los componentes Core Ajax permiten incluir Ajax en los controles de RichFaces y los componentes UI son un conjunto avanzado de

etiquetas que soportan Ajax. Otra característica particular de RichFaces es su CDK, este es un conjunto de herramientas que permiten crear componentes Rich con soporte Ajax.

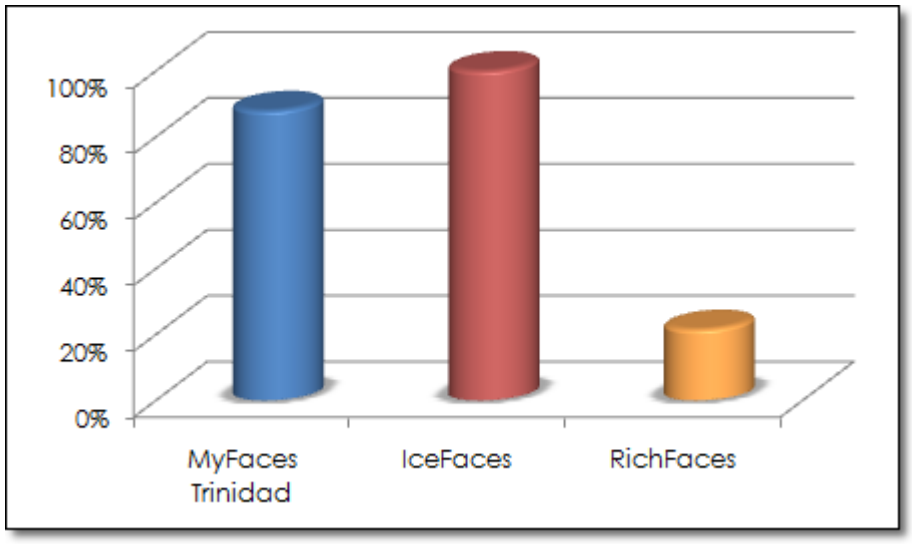


Figura III. 7. Análisis del Parámetro Componentes de Interfaz de Usuario

En la **Figura III. 7**, se observa que el framework IceFaces supera a MyFaces Trinidad y a RichFaces en el parámetro componentes de interfaz de usuario.

3.3.4 Facilidad para el desarrollo

La facilidad para el desarrollo permite optimizar tiempos en la utilización de los componentes, para analizar este parámetro se determinó como IDE NetBeans 7.1.1, el motivo de esta elección es porque el desarrollador se encuentra adaptado al ambiente de trabajo de NetBeans, por tal razón la realización de la aplicación se tornará ágil, al no tener que dedicar tiempo en aprender un nuevo entorno de desarrollo, sino sólo en adquirir conocimientos sobre los componentes del framework ganador y por tanto ponerlos en práctica.

Para el parámetro facilidad para el desarrollo, **Tabla III. XII**, se deben tomar en cuenta las siguientes criterios con sus respectivas valoraciones, tomando en cuenta que a este parámetro se le ha asignado un peso de 5 puntos sobre 10 (5/10).

Tabla III. XII. Criterios y Valoración del Parámetro Facilidad Para el Desarrollo

Criterios	Valoración	Porcentaje (%)
Facilidad de instalación	1	20%
Facilidad para iniciar	1	20%
Facilidad de uso	1	20%
Soporte Ajax	1	20%
Líneas de código	1	20%
TOTAL	5	100%

La facilidad de instalación, facilidad para iniciar, facilidad de uso y líneas de código se evaluará en base a los resultados obtenidos de la encuesta realizada (ver Anexo B – Sección 1).

a) Facilidad de instalación

Este criterio se refiere a la capacidad de cada framework al ser instalado en el ambiente de desarrollo, a continuación se resume la instalación realizada de cada uno de los frameworks.

MyFaces Trinidad cuenta con una guía de instalación no muy completa y poca actualizada, además encontrar información adicional que sirva como guía de inicio al desarrollador se vuelve un proceso dificultoso, debido a que la documentación existente para este framework es escasa. Para incorporar MyFaces Trinidad a NetBeans, a más de sus propias librerías se necesita de bibliotecas adicionales (ver Anexo A – Sección 6) para su correcto funcionamiento.

IceFaces cuenta con una guía de inicio rápido, además ofrece la posibilidad de descargar las bibliotecas para los IDEs Eclipse y NetBeans, facilitando una guía de instalación para cada entorno

de desarrollo, esto facilita en gran medida al desarrollador, al no tener que buscar información adicional para comenzar a trabajar. La instalación de IceFaces sobre NetBeans se lo puede realizar a través de los módulos con extensión .nbm (ver Anexo A – Sección 6).

RichFaces no contiene un tutorial de inicio rápido, la documentación oficial se centra en Maven y JBoss, estas herramientas son dos estándares muy acogidos por la comunidad J2EE. Los dos entornos de desarrollo antes mencionados no son los únicos adoptados por los desarrolladores, por tal motivo para lograr instalar RichFaces en NetBeans, se debe investigar otras fuentes de información. Para la instalación de RichFaces en NetBeans se necesita descargar librerías propias y otras dependencias (ver Anexo A – Sección 6).

Para valorar los resultados finales de esta variable, en la encuesta realizada a los desarrolladores del DESITEL se expuso la pregunta número uno, en la que se expone la facilidad de instalación de los framework en el IDE NetBeans (ver Anexo B – Sección 1), misma que fue contestada posteriormente.

Para el desarrollo de este criterio se debe considerar que el puntaje asignado es del 20%. En base a la tabulación realizada (ver Anexo A – Sección 2), el máximo valor que es 15 corresponde a IceFaces ocupando el porcentaje total del 20%, por tal motivo para obtener los valores de MyFaces Trinidad y RichFaces se realizó una regla de tres (ver Anexo A – Sección 6).

En la **Tabla III. XIII**, se ilustra la calificación obtenida para cada uno de los frameworks.

Tabla III. XIII. Análisis del Criterio Facilidad de Instalación

	MyFaces Trinidad	IceFaces	RichFaces
Facilidad de instalación	12%	20%	13,33%

b) Facilidad para iniciar

Mediante la facilidad de instalación se podrá determinar el grado de configuración que cada framework proporciona al desarrollador para comenzar a trabajar en la aplicación, a continuación se presente las distintas formas de configuración que presta cada uno de los frameworks.

Las librerías y dependencias de RichFaces y MyFaces Trinidad en NetBeans facilitan la configuración básica del archivo **web.xml** para la creación de una aplicación JSF (Ver Anexo A – Sección 7), teniendo que configurar correctamente el archivo antes mencionado.

El proceso de instalación de IceFaces en NetBeans a través de los módulos, proporciona una instalación intuitiva, resolviendo las dependencias necesarias al proyecto, no hace falta configurar el archivo **web.xml** para la creación de la aplicación JSF (ver Anexo A – Sección 7).

La obtención de los resultados finales de este criterio se lo realizó con la ayuda de la pregunta número dos, de la encuesta realizada (ver Anexo B – Sección 1), en la que se cuestiona sobre la configuración de los frameworks.

El porcentaje a tomarse en cuenta para la valoración de esta variable es del 20%, correspondiendo este valor a IceFaces en su máximo valor de 15 puntos (ver Anexo B – Sección 2), por tal razón

para obtener los valores porcentuales de MyFaces Trinidad y RichFaces se realizó una regla de tres (ver Anexo A – Sección 7).

En la **Tabla III. XIV**, se muestra los valores correspondientes al criterio facilidad para iniciar.

Tabla III. XIV. Análisis del Criterio Facilidad Para Iniciar

	MyFaces Trinidad	IceFaces	RichFaces
Facilidad para iniciar	13,33%	20%	13,33%

c) Facilidad de uso

La facilidad de uso se refiere al grado en que la interfaz, la distribución de los paneles y la forma en que los componentes facilitan o dificultan su manejo, posteriormente se menciona las características que prestan cada uno de los frameworks en el IDE NetBeans.

EL IDE NetBeans cuenta con un editor visual para la navegación entre páginas a través del archivo **faces-config.xml**, este editor se encuentra disponible para MyFaces Trinidad, IceFaces y RichFaces; además, permite el autocompletado de las etiquetas JSF de los frameworks (ver Anexo A – Sección 8).

Adicionalmente a lo antes mencionado, el IDE NetBeans para facilitar el desarrollo con el framework IceFaces, proporciona una paleta de componentes de los controles, mismos que pueden ser arrastrados fácilmente aumentando la eficiencia de desarrollo (ver Anexo A – Sección 8).

Para la medición de este criterio se lo realizó en base a la tabulación de la pregunta tres de la encuesta realizada (ver Anexo B – Sección 2), que trata sobre la facilidad de uso de los frameworks

en el IDE Netbeans; el máximo valor obtenido es 15 que pertenece a IceFaces, por tal motivo el porcentaje total asignado e éste es del 20%; para obtener el valor de MyFaces Trinidad y RichFaces se realizó una regla de tres (ver Anexo A – Sección 8).

En la **Tabla III. XV**, se refleja la calificación porcentual obtenida de cada uno de los frameworks en el criterio facilidad de uso.

Tabla III. XV. Análisis del Criterio Facilidad de Uso

	MyFaces Trinidad	IceFaces	RichFaces
Facilidad de uso	13,33%	20%	13,33%

d) Soporte Ajax

Mediante este criterio se valorará la facilidad de cada framework al momento de trabajar de forma asíncrona entre el cliente y servidor.

Para la metodología de pruebas de este criterio se desarrolló una pequeña aplicación con cada uno de los frameworks. La aplicación consistió en una pantalla que permite listar los datos del paciente en una tabla con paginación y para monitorizar las peticiones AJAX se utilizó el componente Firebug de Firefox (ver Anexo A – Sección 9), a continuación se menciona los resultados obtenidos.

MyFaces Trinidad permite construir aplicaciones sencillas basadas en componentes Ajax y facilita muchos caminos para atender peticiones tanto de componentes Ajax como de componentes no Ajax.

IceFaces proporciona el Ajax automático, siendo esta tarea transparente para el desarrollador, pues le ahorra tiempo a la hora de programar; además no requiere de etiquetas especiales, basta con colocar los componentes en el editor del código fuente e IceFaces se encarga de enviar la información necesaria entre el cliente y servidor.

RichFaces permite integrar funcionalidades Ajax en sus componentes visuales, mediante el soporte de la librería Ajax4JSF, por tal motivo, RichFaces no es tan transparente para el desarrollador, puesto que se tiene que añadir componentes no visuales de la librería antes mencionada.

En la **Tabla III. XVI**, se ilustra la puntuación porcentual asignada a cada uno de los frameworks analizados en el soporte Ajax.

Tabla III. XVI. Análisis del Criterio Soporte Ajax

	MyFaces Trinidad	IceFaces	RichFaces
Soporte Ajax	20%	20%	20%

e) Líneas de código

Mediante las líneas de código se medirá la cantidad utilizada para desarrollar la interfaz de usuario, para el análisis de este parámetro se pidió a los desarrolladores del DESITEL que realizaran el módulo edición y listado de pacientes en los frameworks MyFaces Trinidad, IceFaces y RichFaces (ver Anexo A – Sección 10), posteriormente a esto consideraron el número de líneas escritas en la vista o interfaz de usuario para dar su criterio en la pregunta número cuatro de la encuesta (ver Anexo B – Sección 1).

De las encuestas realizadas, el máximo valor de 15 puntos corresponde a IceFaces, por tanto se le asigna el valor porcentual del 20%, para conseguir los valores de MyFaces Trinidad y RichFaces se realiza una regla de tres (ver Anexo A – Sección 10).

En la **Tabla III. XVII**, se muestra la puntuación porcentual obtenida de cada uno de los frameworks analizados en el criterio líneas de código.

Tabla III. XVII. Análisis del Criterio Líneas de Código

	MyFaces Trinidad	IceFaces	RichFaces
Líneas de código	13,33	20%	13,33%

Tomando en cuenta la facilidad de instalación, la facilidad para iniciar, la facilidad de uso, el soporte AJAX y el número de líneas, en la **Tabla III. XVII**, se ilustra los valores cuantitativos obtenidos del análisis realizado en el parámetro facilidad para el desarrollo.

Tabla III. XVIII. Análisis del Parámetro Facilidad Para el Desarrollo

	MyFaces Trinidad	IceFaces	RichFaces
Facilidad de instalación	12,00%	20,00%	13,33%
Facilidad para iniciar	13,33%	20,00%	13,33%
Facilidad de uso	13,33%	20,00%	13,33%
Soporte AJAX	20,00%	20,00%	20,00%
Líneas de código	13,33%	20,00%	13,33%
TOTAL	71,99%	100,00%	73,32%

La documentación que permita iniciar una aplicación con MyFaces Trinidad o RichFaces en NetBeans, en un tiempo de desarrollo óptimo, es escasa, puesto que para RichFaces existe información orientada a Maven y JBoss y para MyFaces Trinidad es difícil encontrar un material útil que ayude con la inicialización de dicha aplicación, no siendo este el caso de IceFaces, pues posee documentación que permite ahorrar tiempo a la hora de empezar a crear una aplicación.

Existen varias IDEs que proporcionan un buen soporte para el desarrollo de aplicaciones JSF, entre los que se puede mencionar NetBeans y Eclipse.

Los componentes de la paleta de IceFaces poseen la acción drag&drop, está propiedad permite arrastrar y soltar los controles en el editor del código fuente, generándose automáticamente la declaración del componente seleccionado, aunque no se consigue una presentación visual del resultado.

La flexibilidad de Ajax que tiene RichFaces es una ventaja con respecto a otros frameworks, puesto que al interactuar con otras librerías que no tengan soporte Ajax, éste podrá ser incorporado en los componentes, cabe mencionar que los controles de usuario de IceFaces vienen integrados con Ajax Automático en su totalidad.

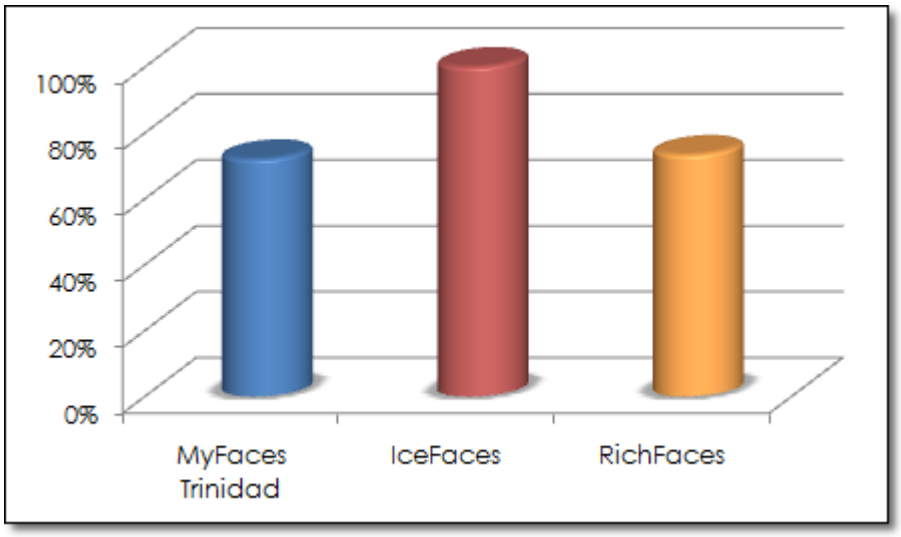


Figura III. 8. Análisis del Parámetro Facilidad Para el Desarrollo

En la **Figura III. 8**, se observa que IceFaces supera a RichFaces y MyFaces Trinidad en lo que se refiere al parámetro facilidad para el desarrollo.

3.4 RESUMEN COMPARATIVO

Los datos presentados en la **Tabla III. XIX**, son la sumatoria de todos los parámetros analizados; tal como se ha observado, todos los parámetros han sido evaluados tomando en consideración para la valoración el cien por ciento (100%).

Tabla III. XIX. Tabulación de Datos de los Parámetros Analizados

Frameworks	MyFaces Trinidad	IceFaces	RichFaces	Porcentaje asignado (%)
Aprendizaje	38,32	76,93	73,61	10
Calidad	100,00	70,80	62,18	10
Componentes de interfaz de usuario	87,93	100,00	21,55	30
Facilidad para el desarrollo	71,99	100,00	73,32	50

Luego de aplicar las fórmulas (ver Anexo A – Sección 11) para obtener los porcentajes reales de cada framework, tomando en cuenta el peso asignado a cada uno de ellos, en la **Tabla III. XX**, se muestra los resultados obtenidos.

Tabla III. XX. Análisis de los Parámetros Evaluados

Frameworks	MyFaces Trinidad	IceFaces	RichFaces	Porcentaje asignado (%)
Aprendizaje	3,83	7,7	7,36	10
Calidad	10	7,08	6,22	10
Componentes de interfaz de usuario	26,38	30	6,47	30
Facilidad para el desarrollo	36	50	36,66	50
TOTAL	76,21	94,78	56,71	100

En la **Figura III. 9**, se ilustra una representación gráfica del análisis realizado, en donde el porcentaje máximo es el valor que puede alcanzar el parámetro a evaluar.

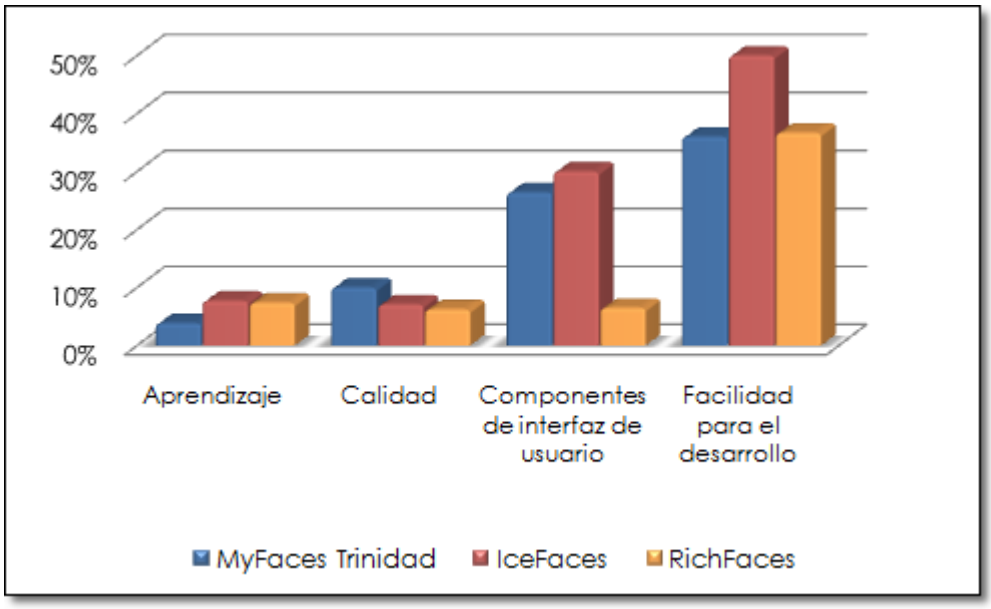


Figura III. 9. Representación de los Parámetros Analizados

En la **Figura III. 10**, se puede observar el resultado final obtenido.

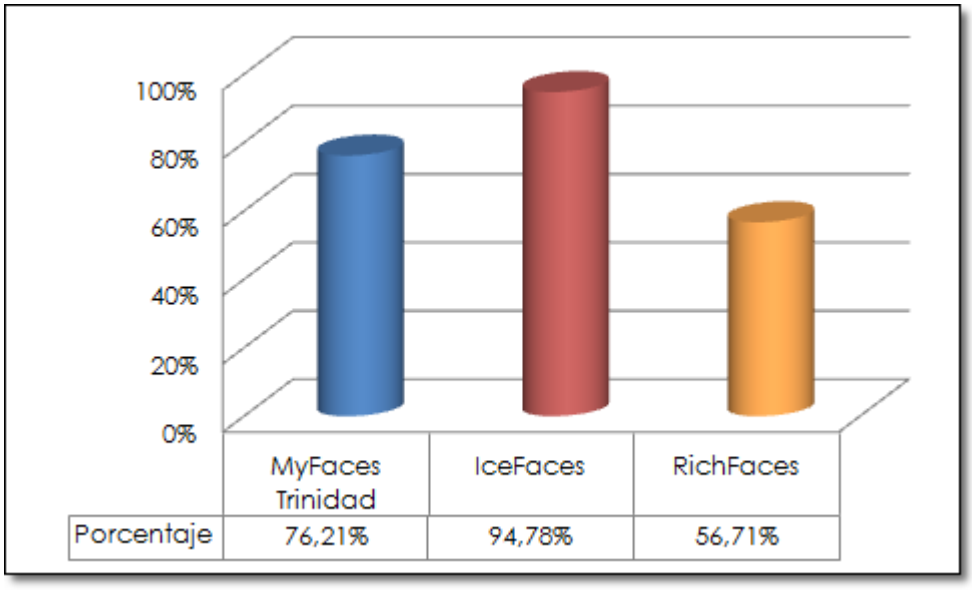


Figura III. 10. Resultado Final del Análisis

En la **Figura III. 10**, se observa claramente que IceFaces supera con un valor de 94,78% sobre MyFaces Trinidad y RichFaces que obtuvieron 76,21% y 56,71% respectivamente; estos valores nos indica que el desarrollo de la aplicación para el Sistema Nutricional de la ESPOCH se realizará con IceFaces.

3.5 COMPROBACIÓN DE LA HIPÓTESIS

La hipótesis de la tesis de investigación está formada en base a variables correlaciones y además el trabajo investigativo corresponde a distribuciones no paramétricas, por lo que se seleccionó chi cuadrado para llegar a su demostración, considerando como pasos a seguir:

- Planteamiento de la hipótesis.
- Establecimiento del nivel de significancia
- Cálculo del estadístico
- Grados de libertad
- Criterio de decisión

3.5.1 Planteamiento de la hipótesis

Para la demostración de la hipótesis se establece: la hipótesis nula (H_0) y alternativa (H_1), encontrándose H_0 dentro de la zona de aceptación y H_1 en la zona de rechazo.

H_0: "El estudio comparativo entre los frameworks MyFaces, IceFaces y RichFaces no permitirá seleccionar el más eficiente para desarrollar la aplicación Web para el Sistema Nutricional de la ESPOCH."

H_1: "El estudio comparativo entre los frameworks MyFaces, IceFaces y RichFaces permitirá seleccionar el más eficiente para desarrollar la aplicación Web para el Sistema Nutricional de la ESPOCH."

3.5.2 Establecimiento del nivel de significancia

Una vez establecida la hipótesis nula y alternativa, se bebe determinar el nivel de significancia, para el caso del presente análisis se utilizará un nivel de significación estadística de $\boldsymbol{\alpha = 0,05}$, que rechaza la hipótesis de que el estudio comparativo no permitirá seleccionar el framework más eficiente para el desarrollo del Sistema Nutricional de la ESPOCH, tal como se muestra en la **Figura III. 11**.

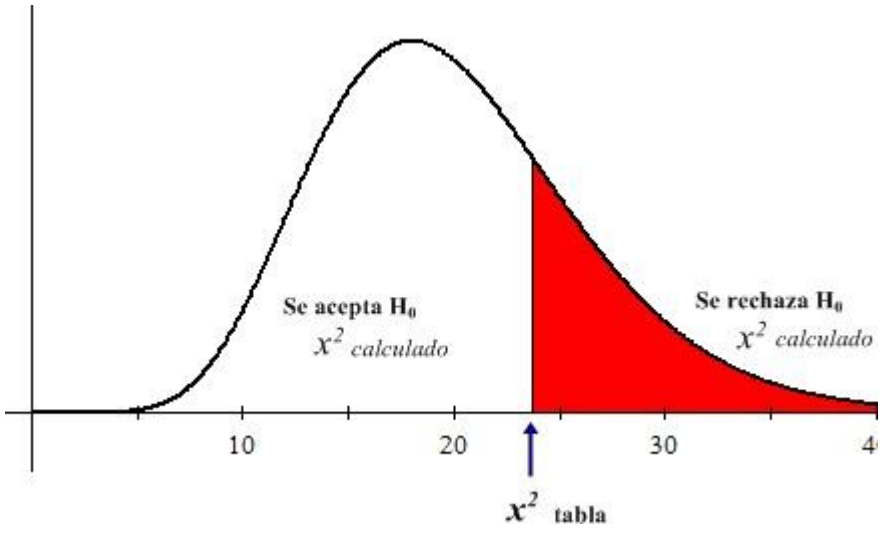


Figura III. 11. Curva de Distribución del Chi Cuadrado

3.5.3 Cálculo del estadístico

Como se mencionó anteriormente, para el presente trabajo investigativo se seleccionó como estadístico de prueba de hipótesis la técnica del **"chi cuadrado"**. La fórmula que da el estadístico es como sigue:

$$\boldsymbol{x^2 = \sum_i \frac{(observada_i - esperada_i)^2}{esperada_i}}$$

En función a la encuesta realizada a los programadores de DESITEL (ver Anexo B – Sección 1), se realizó la tabla de frecuencias observadas (ver Anexo A – Sección 12), considerando los pesos de valoración cualitativa – cuantitativa (ver Tabla III. II.), donde la variable *no eficiente* se encuentra entre los porcentajes de 0 y 50, y la variable *eficiente* se ubica entre los porcentajes de 50 y 100.

Luego de determinar las frecuencias observadas, es necesario determinar las frecuencias esperadas (ver Anexo A – Sección 12), importantes para aplicar el método de chi cuadrado.

Para conocer las frecuencias esperadas se aplicó la siguiente expresión.

$$\boldsymbol{fe = \frac{total\ del\ renglón * total\ de\ la\ columna}{total\ de\ resultados}}$$

Finalmente se determina el valor de chi cudrado, resultando el siguiente valor.

$\boldsymbol{x^2} = 5{,}019 + 0{,}642 + 1{,}406 + 0{,}585 + 4{,}500 + 0{,}000 + 0{,}000 + 0{,}000 + 0{,}320 + 0{,}716 + 0{,}547 + 0{,}026 + 1{,}409 + 0{,}000 + 0{,}000 + 0{,}000 + 3{,}451 + 0{,}028 + 5{,}413 + 1{,}196 + 0{,}842 + 0{,}000 + 0{,}000 + 0{,}000$

$\boldsymbol{x^2} = 26{,}101$

3.5.4 Grados de libertad

La determinación de los grados de libertad **(gl)**, está en función del número de filas **(r)** y el número de columnas **(k)**, con la siguiente expresión:

$$\boldsymbol{gl} = (r - 1) * (k - 1)$$

Entonces:

$$\boldsymbol{gl} = (\mathbf{8} - \mathbf{1}) * (\mathbf{3} - \mathbf{1})$$

$$\boldsymbol{gl} = \mathbf{7} * \mathbf{2}$$

$$\boldsymbol{gl} = \mathbf{14}$$

De acuerdo a la tabla estadística de distribución de chi cuadrado (ver Anexo A – Sección 13), con un nivel de significancia 0,05 a 14 grados de libertad, se obtiene un valor de 23,68.

3.5.5 Criterio de decisión

Para determinar el **criterio de decisión**, se acepta $\mathbf{H_0}$ cuando $\boldsymbol{x^2\ calculado} < \boldsymbol{x^2\ tabla}$, caso contrario se rechaza $\mathbf{H_0}$.

|

Donde el valor de $\boldsymbol{x^2\ tabla}$ representa el valor obtenido de la tabla de Distribución de chi cuadrado (ver Anexo A – Sección 13), según el nivel de significancia y el grado de libertad y $\boldsymbol{x^2\ calculado}$ representa el valor del cálculo del estadístico.

Entonces la regla de decisión es: No rechazar H_0 si el valor que se obtenga para $\boldsymbol{x^2\ calculado}$ es menor que $\boldsymbol{x^2\ tabla}$. Si el valor calculado es igual o mayor al valor crítico, se rechaza H_0 y se acepta H_1.

Como:

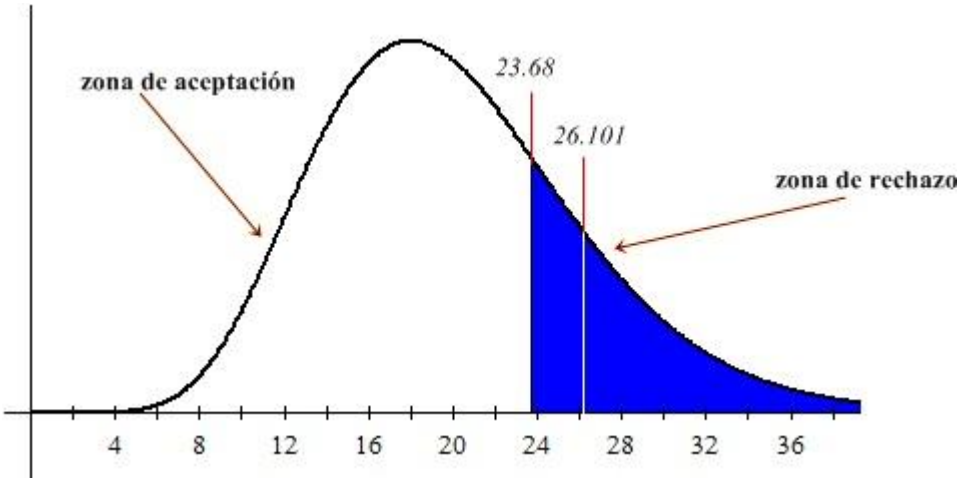


Figura III. 12. Demostración de la Hipótesis

$\boldsymbol{x^2\ calculado} = 26{,}101$ y $\boldsymbol{x^2\ tabla} = 23{,}68$

Entonces se tiene $\boldsymbol{x^2\ calculado} > \boldsymbol{x^2\ tabla}$, lo que significa que $\boldsymbol{x^2\ calculado}$ está en la zona de rechazo de la $\mathbf{H_0}$, por tanto se rechaza la hipótesis nula y se acepta la de la investigación, siendo la que sigue:

"El estudio comparativo entre los frameworks MyFaces, IceFaces y RichFaces permitirá seleccionar el más eficiente para desarrollar la aplicación Web para el Sistema Nutricional de la ESPOCH."

C A P Í T U L O I V

DESARROLLO DE LA APLICACIÓN WEB

El desarrollo de una buena aplicación obedece a una serie de etapas y procesos a seguir, en donde la selección de una buena metodología de desarrollo es fundamental para el éxito de la solución, por tal motivo, para el análisis, diseño y desarrollo del Sistema Nutricional de la ESPOCH se ha seleccionado la metodología Microsoft Solution Project (MSF). Los procesos definidos por esta metodología son: visión, planificación, desarrollo, estabilización y despliegue.

En esta sección se desarrolla cada una de las fases que constituyen la metodología MSF.

4.1 FASE I: VISIÓN Y ALCANCES

En esta fase se definirán los requerimientos del negocio, en la que la intervención del cliente con el equipo de trabajo es trascendental para la realización de una buena recolección de requisitos, además en esta etapa se definen los objetivos generales del proyecto.

4.1.1 VISIÓN

A continuación se define el problema, la visión, los beneficios y las metas que se alcanzarán con el desarrollo del proyecto.

4.1.1.1 Problema

La Escuela de Nutrición y Dietética preocupada en mejorar la calidad de vida de los docentes, trabajadores, empleados, estudiantes politécnicos y público en general, ha visto importante integrar al Dispensario Médico de la ESPOCH una aplicación que cubra las expectativas deseadas.

4.1.1.2 Visión del Proyecto

El sistema de evaluación nutricional pretende ser una aplicación web robusta y eficiente que ofrezca servicios de alto nivel y calidad para el profesional en nutrición, pues a través de la configuración de diferentes parámetros se evaluará el estado nutricional del paciente emitiendo un diagnóstico, con el cual el profesional en nutrición podrá brindar un tratamiento acertado basado en estos datos.

4.1.1.3 Beneficios

El desarrollo e implementación del Sistema Académico nos brindará beneficios [54] tangibles e intangibles.

4.1.1.4 Metas

El sistema nutricional que se desarrollará para la Escuela de Nutrición y Dietética de la ESPOCH, contará con una base de datos sólida y robusta, estará enfocado en controlar el acceso de los usuarios al sistema basándose en un mecanismo de validación y posteriormente habilitará todos los

[54] Manual Técnico: Anexo A – Información CD NutriSys

permisos que dicho usuario autenticado disponga, brindará rapidez en el momento de emitir la evaluación nutricional del paciente y además generará reportes que ayuden a la buena toma de decisiones.

El sistema a desarrollar será completamente parametrizable y configurable de acuerdo a las necesidades que se presenten.

Toda la información generada se encontrará almacenada en una base de datos y se podrá visualizar en forma de reportes de acuerdo a los requerimientos de los usuarios.

4.1.2 PERFILES DE USUARIO

Para poder identificar a los potenciales usuarios se debe identificar con perfección hacia quien o quienes van dirigida la aplicación, y alcanzar las metas propuestas de tal manera que la funcionalidad del sistema se vea reflejada en la satisfacción de los mismos.

Los usuarios del Sistema Académico identificados se definen en la **Tabla IV. XXI**, y su función a continuación.

Tabla IV. XXI. Usuarios Potenciales del Sistema

Usuario	Responsable/es
Administrador	Delegado del Departamento de Nutrición
Nutricionista (Docente, Estudiante)	Doctores y Estudiantes delegados de la Escuela de Nutrición y Dietética.

Administrador.- Administra el Sistema Nutricional.

Doctores/Estudiantes.- Ingresa datos de los pacientes y realizan las respectivas evaluaciones nutricionales.

4.1.3 ÁMBITO DEL PROYECTO

El Sistema Nutricional de la ESPOCH estará enfocado en almacenar la información personal, antropométrica, bioquímica y de consumo (en el caso del recién nacido) del paciente, además, evaluará el estado nutricional del mismo, de tal manera que el sistema emita un diagnóstico acertado y el profesional en nutrición brinde un tratamiento adecuado en función de la valoración realizada.

4.1.4 CONCEPTO DE LA SOLUCIÓN

La solución que se presenta en la **Figura IV.13**, indica que el Sistema Nutricional de la ESPOCH es una aplicación Web desarrollada con el framework IceFaces de JSF y la herramienta de mapeo objeto-relacional (ORM) Hibernate.

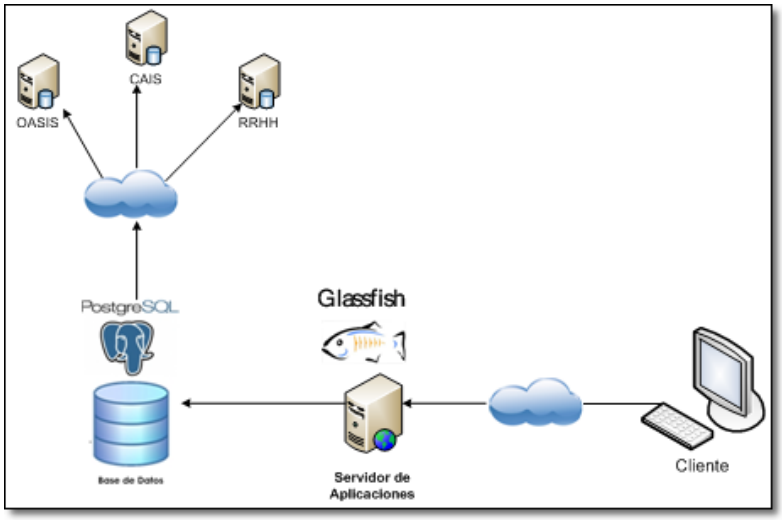


Figura IV. 13. Concepto de la Solución

La solución estará alojada en un servidor de aplicaciones GlassFish y además trabajará con el motor de bases de datos PostgreSQL. Adicionalmente, el sistema interactuará con las bases de datos de Recursos Humanos (RRHH), Académico (OASIS), y con el Centro de Atención Integral de Salud (CAIS), a través del consumo de los servicios WEB, para obtener la información de los pacientes.

4.1.4.1 Software a utilizar

El software que se utilizará para desarrollar el Sistema Nutricional de la ESPOCH, se menciona a continuación:

- **ArgoUML.-** Herramienta CASE de modelado de datos.
- **Open Proj**.-Herramienta de Planificación.
- **Netbeans 7.1.1**.- IDE de desarrollo.
- **JDK 1.6**.-Kit de desarrollo.
- **IceFaces 3.1.1-** Framework de Desarrollo JSF.
- **Glassfish 3.1 -**Servidor de Aplicaciones.
- **PostgreSQL 9.1**.- Motor de Base de Datos.
- **Ireport / JasperReport 4.7.1.-** Reporteador.
- **JFreeChart 1.0.14.-** Creador de Gráficos.
- **Toad Data Modeler.-** Modelador del diseño lógico y físico de la base de datos.

4.1.4.2 Arquitectura

El Sistema Nutricional de la ESPOCH se desarrollará bajo cuatro capas, mismas que se ilustran en la **Figura IV. 14**, y seguidamente se define cada una de ellas.

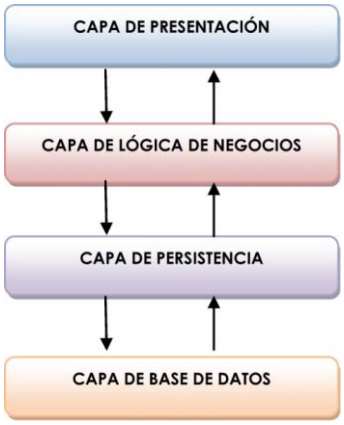


Figura IV. 14. Arquitectura

- **Capa de Presentación.-** Permite que el usuario interactúe directamente con la aplicación. Esta capa se comunica solamente con la capa lógica de negocios y hace uso de la tecnología IceFaces de JSF.

- **Capa Lógica de Negocio.-** La capa lógica de negocio es la encargada del intercambio de información que será entregada a la capa de presentación o vista, controlando todo el flujo. El correcto funcionamiento de la aplicación dependerá en gran medida de la lógica que se desarrolle.

- **Capa de Persistencia.-** Para la gestión de la persistencia, se utilizará un ORM (Hibernate en este caso), con la finalidad de realizar una aplicación elegante y totalmente desacoplada del gestor de base de datos. La función de esta capa es establecer la comunicación entre la lógica de negocios y la base de datos.

- **Capa de Base de Datos.-** El almacenamiento de todos los datos residen en esta capa.

  La capa de base de datos se comunica con la aplicación a través de la capa de persistencia, permitiendo realizar todas las operaciones que sean generadas en la lógica de negocios. La base de datos que se empleará, en el desarrollo del Sistema Nutricional de la ESPOCH es PostgreSQL.

4.1.5 OBJETIVOS DEL PROYECTO

Los objetivos de negocio que pretende alcanzar el desarrollo del Sistema Nutricional de la ESPOCH se definen a continuación:

4.1.5.1 Objetivos de Negocio

Ayudar a tomar decisiones sobre dietas nutricionales de manera certera y eficaz que beneficien al paciente mejorando sus hábitos alimentarios y con ello su salud.

4.1.5.2 Objetivos de Diseño

- Administrar y gestionar la creación de usuarios y pacientes del sistema.
- Reducir tiempos en la elaboración de informes sobre el estado nutricional del paciente.
- Brindar a cada usuario la posibilidad de tener sus propios permisos dentro de la aplicación.
- Crear un diseño escalable que prometa agrandarse y adaptarse en un futuro.
- Brindar un excelente ambiente de trabajo entre el personal que manipulará la aplicación.

4.1.6 FACTORES CRÍTICOS

El principal factor crítico en el desarrollo de un software es el análisis de riesgos [55], el enfrentamiento proactivo de los riesgos que pueden afectar al desarrollo o a la calidad de los requisitos y las acciones para evitarlos, permitirán minimizar problemas que persisten en el desarrollo de software.

Identificando los riesgos conocidos y predecibles, se da un paso adelante para evitarlos cuando sea posible y controlarlos cuando sea necesario, de esta manera se puede evitar que los riesgos se conviertan en problemas.

4.1.7 PLANIFICACIÓN INICIAL

En esta sección se describirá el equipo de trabajo, los perfiles de usuarios y el cronograma inicial de trabajo, estimados para la ejecución del proyecto.

4.1.7.1 Equipo de Trabajo

En la **Tabla IV. XXII**, se presenta la asignación de los roles MSF, mismos que conforman el equipo de trabajo a implementar el Sistema Nutricional de la ESPOCH.

Tabla IV. XXII. Equipo de Trabajo

ROL MSF	RESPONSABLE	CARGO
Product Manager	Miriam Jaramillo	Tesista
Program Manager	Miriam Jaramillo	Tesista
Development Manager	Miriam Jaramillo	Tesista
Testing Manager	Ing. Jorge Menéndez	Director de tesis
User Experience Manager	Miriam Jaramillo	Tesista
Release Manager	Ing. Jorge Menéndez	Director de tesis

[55] Manual Técnico: Anexo B – Información CD NutriSys

La función de cada uno de los roles MSF se definen a continuación:

Product Manager.- Es la persona que trabajará directamente con el cliente, para levantar los requerimientos en la etapa del diseño.

Program Manager.- Se encargará del proceso del desarrollo, su meta es entregar una solución que cumpla con las condiciones establecidas en el proyecto.

Development Manager.- Es el responsable de desarrollar las soluciones tecnológicas acorde a las especificaciones entregadas por el Program Manger.

Testing Manager.- Es el encargado de aprobar y certificar la calidad del producto, mediante el análisis de la gestión de errores e inconsistencias entre lo desarrollado y lo solicitado.

User Manager.- Es el que analizará las necesidades y asuntos de soporte de usuario.

Release Manager.- Tiene la responsabilidad de la implementación y la reducción del impacto que ésta pueda producir.

4.1.7.2 Perfiles de Usuario

Al realizar el análisis respectivo para la creación del Sistema Nutricional de la ESPOCH, se ha contemplado que para el uso del mismo se necesitará los perfiles de usuario tales como: Administrador, Docentes y Estudiantes.

En la **Tabla IV. XXIII**, se describe las funciones de cada uno de ellos:

Tabla IV.XXIII. Perfiles de Usuario

NOMBRE	PERFIL	TIPO DE ACCESO	DESCRIPCIÓN
Administrador	Administrador	General	- Es el encargado de crear usuarios, contraseñas y asignar permisos conforme lo crea conveniente. - Asesoramiento técnico de la aplicación.
Nutricionista (Docentes Estudiantes**)**	Usuario	Limitado	- Podrán acceder a todas las tareas que el Administrador les haya asignado.

4.1.7.3 Cronograma Inicial de Trabajo.

El cronograma inicial [56] de trabajo permite valorar el tiempo para la ejecución del proyecto. Una vez finalizada la construcción del cronograma de trabajo, se estimó que la duración de la implementación del Sistema Nutricional de la ESPOCH "NutriSys", tendrá un periodo de 277 días, considerando 8 horas de trabajo diarias, cronograma que inicia el 19 de diciembre del 2011 y culmina el 8 de enero del 2013.

4.2 FASE II. PLANIFICACIÓN

Iniciar la construcción de una aplicación requiere la definición de un plan que ayude a cumplir con todas las necesidades y expectativas de un sistema dentro de un tiempo limitado, es por esta razón que las actividades deben ser controladas para evitar retrasos y pérdida de recursos.

El proceso de planificación juega un papel muy importante al momento de elaborar un proyecto, por lo cual la construcción de un plan ayuda a refinar dichas actividades hasta llegar a la eficiencia requerida.

4.2.1 ESPECIFICACIÓN FUNCIONAL

En esta sección se define y comprende los requisitos necesarios para el desarrollo de la aplicación, mediante un acuerdo razonable entre el nutricionista y el equipo de trabajo del proyecto.

[56] Manual Técnico: Anexo C – Información CD NutriSys

4.2.1.1 DISEÑO CONCEPTUAL

El diseño conceptual comprende la abstracción de los requerimientos funcionales, los actores, diagramas de casos de uso, casos de uso, escenarios y glosario de términos, mismos que se detallan a continuación.

Requerimientos

La Especificación de Requerimientos de Software (SRS) [57] define de forma precisa el producto que se va a construir, por tanto se ha desarrollado el SRS aplicando la norma IEEE-830 versión 1, logrando definir 24 requerimientos funcionales, requerimientos de interfaces externas, restricciones de diseño y atributos.

Actores

Los actores que harán uso del Sistema Nutricional de la ESPOCH se describen en la **Tabla IV. XXIV**.

Tabla IV.XXIV. Actores

Actor	Descripción
Administrador	Administra el Sistema NutrSys
Nutricionista (Docente/Estudiante)	Según el perfil asignado podrán trabajar en el sistema

Diagramas de casos de uso

A través de los diagramas de casos de uso [58] se presenta el comportamiento de los requisitos funcionales mediante una notación gráfica. Estos diagramas se elaboraron basándose en la definición de los requerimientos previamente establecidos, por tal motivo, para la construcción de

[57] Manual Técnico: Anexo A – Información CD NutriSys
[58] Manual Técnico: Anexo D – Información CD NutriSys

los mismos se consideró los módulos: Gestión de Usuarios, Configuración del Sistema y Gestión de Datos del Paciente.

El modelo de casos de uso que describe el módulo de Gestión de Usuarios se ilustra en la **Figura IV. 15**.

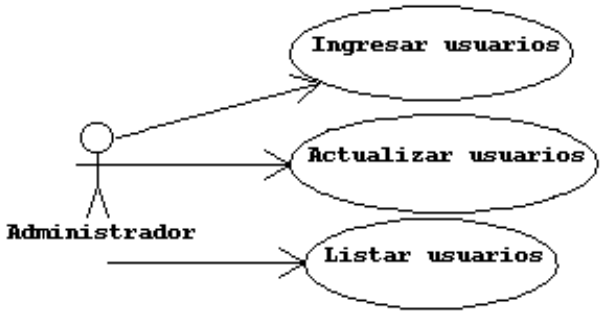


Figura IV. 15. Diagrama de Casos de Uso – Gestión de Usuarios

Casos de uso en formato extendido

Una vez definidos los diagramas de casos de uso, en esta sección se detalla cada uno de ellos, siendo éstos los casos de uso en formato extendido [59]. Para el Módulo de Gestión de Usuarios, Gestión de Datos del Paciente y Configuración del Sistema se crearon 3, 17 y 4 casos de uso respectivamente.

En la **Tabla IV. XXV** se muestra el caso de uso **"Crear usuario"**, donde se define: el identificador del caso de uso, con el nombre del mismo anteponiéndole "cu"; el nombre del caso de uso, que corresponde a una descripción abreviada del requerimiento funcional respectivo; los actores, el o los actores que desempeñará los roles; el propósito, que corresponde a la intención del caso de uso; la visión general, que es un resumen o repetición del caso de uso de alto nivel; el tipo, que puede ser primario, secundario u opcional, también esencial o real; referencia, que se relaciona con el requisito del sistema; el curso típico de eventos, donde se describe la interacción entre los actores y

[59] Manual Técnico: Anexo E – Información CD NutriSys

el sistema mediante las acciones numeradas de cada uno; los cursos alternativos, que describe los puntos que surgen de una alternativa, junto al número de la excepción**.**

Tabla IV.XXV. Caso de Uso Crear Usuario

IDENTIFICADOR DEL CASO DE USO	cuCrearUsuario
NOMBRE DEL CASO DE USO	Crear usuario
ACTORES	Administrador
PROPÓSITO	Realiza la creación de un usuario, controlando la existencia del mismo en el sistema.
VISIÓN GENERAL	El administrador crea un nuevo usuario en el sistema rellenando los datos necesarios. El sistema comprueba si el usuario existe y que los datos ingresados sean correctos. Si el usuario no existe y los datos son correctos crea el usuario.
TIPO	Primario y esencial
REFERENCIA	Req1
CURSO TÍPICO DE EVENTOS	
ACCIÓN DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Este caso de uso inicia cuando el administrador solicita crear un nuevo usuario. 3. Ingresa la información del usuario, le asigna un perfil de una lista desplegable y posteriormente guarda los datos.	2. Muestra la pantalla para ingreso de usuario. 4. Verifica que los datos sean correctos. 5. Verifica que el usuario a ingresar no exista. 6. Si el usuario no existe y los datos son correctos, crea el nuevo usuario.
CURSO ALTERNATIVO	
4ª. Si los datos son incorrectos muestra un mensaje de error "Datos incorrectos" **5ª.** Si el usuario existe muestra un mensaje de error "Usuario ya existe."	

Escenarios

Para la implantación del Sistema Nutricional de la ESPOCH "NutriSys", el Departamento de Sistemas y Telemática cuenta con un servidor HP StorageWorks 4400 Enterprise Virtual Array (EVA), sobre la cual se encuentran varios Enclosure en la que está instalada la Infraestructura VMWARE SX 3.5, las características de cada máquina virtual se detalla en la **Tabla IV. XXVI.**

Tabla IV.XXVI. Escenarios

Definición	Característica
Procesador	Quad-Core Xeon, 3166 MHz.
Memoria RAM	4.00 GB
Disco Duro	160 GB

En cuanto se refiere al cliente, la Escuela de Nutrición y Dietética cuenta con una máquina y una impresora.

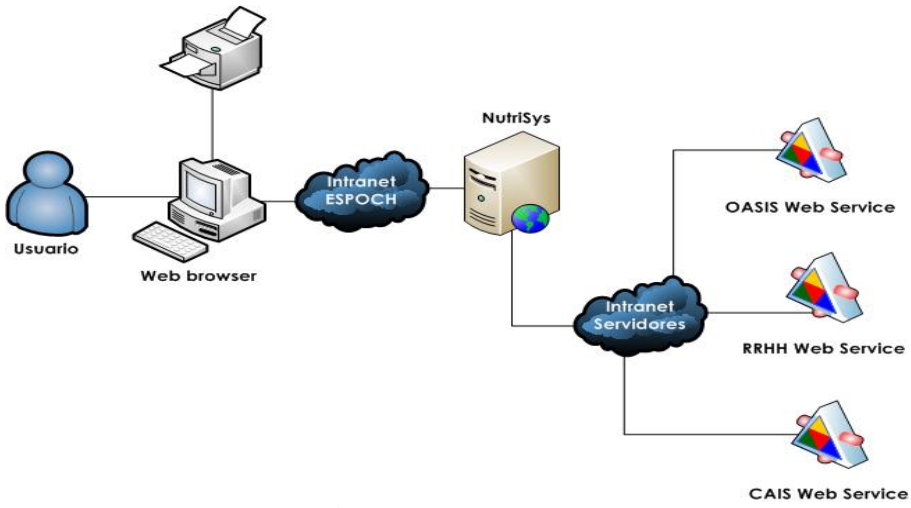


Figura IV. 16. Escenario

Tomando en cuenta todo lo expuesto anteriormente en la **Figura IV. 16**, se ilustra el escenario de implantación del Sistema Nutricional de la ESPOCH, en la que se puede observar que se utilizará una red privada con la tecnología Internet para acceder a Nutrisys, y éste a su vez a los web Service.

Glosario de Términos

Ver sección “Glosario” al final del documento.

4.2.1.2 DISEÑO LÓGICO

El diseño lógico es el proceso de describir la solución en términos de su organización, su estructura, y la interacción de sus partes desde la perspectiva del equipo del proyecto.

Tecnología a Utilizar en el Proyecto

El Departamento de Sistemas y Telemática (DESITEL) de la ESPOCH para disminuir costos y optimizar procesos, propone trabajar con el servidor de aplicaciones de software libre GlassFish y con la base de datos relacional orientada a objetos de código libre PostgreSQL.

Tomando en cuenta la recomendación anterior, a continuación se describe las tecnologías a utilizar:

- Sistema operativo Centos en donde se instalará la aplicación.
- GlassFish Server Open Source Edition 3.1.2 como servidor de aplicaciones.
- PostgreSQL 9.1 para guardar los datos.
- Netbeans 7.1 como IDE de desarrollo.
- Kit de desarrollo JDK 6u37.
- IceFaces 3.0.1 como framework de desarrollo para las interfaces.
- Hibernate 3.2.5 como framework de persistencia.
- Ireport 4.7.0 como herramienta para la generación de reportes.
- JFreeChart 1.0.14 para la creación de las gráficas nutricionales.
- C3P0 0-0.9.1.2 librería de pool de conexiones, que permite mejor el desempeño de la aplicación y la utilización de los recurso.
- ArgoUML para la creación de los distintos diagramas utilizados en este documento.
- Open Proj para la planificación del proyecto.
- Toad Data Modeler aplicación para diseñar el modelo Entidad-Relación.
- MSF (Microsoft Solution Framework) como metodología de desarrollo del proyecto de software.

Diagramas de Secuencias

Los diagramas de secuencia [60] permiten descubrir las clases que se necesitarán durante el desarrollo del Sistema Nutricional de la ESPOCH "NutriSys", además muestran una visión dinámica de los procesos de la aplicación, ayudando a identificar los métodos que se debe implementar en cada clase.

Para el desarrollo del sistema se identificaron 24 diagramas de secuencia, en la **Figura IV. 17** se presenta el diagrama de secuencia del caso de uso "Crear usuario".

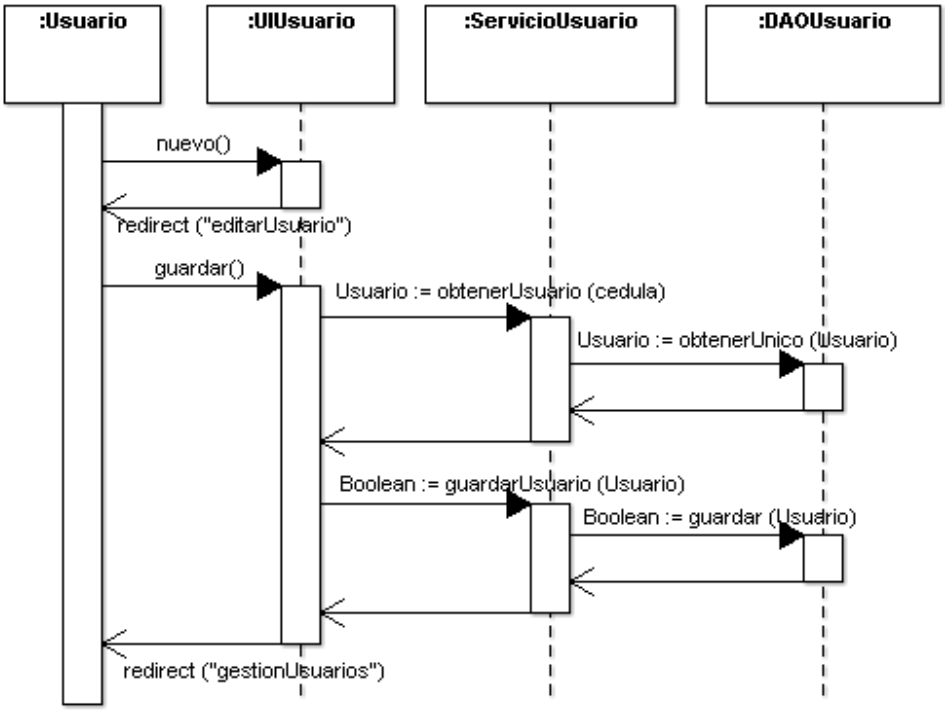


Figura IV. 17. Diagrama de Secuencia cuCrearUsuario

[60] Manual Técnico: Anexo F – Información CD NutriSys

Diagramas de Clases

A partir de los diagramas de secuencia definidos anteriores se especifica el diagrama de clases [61] de diseño. Se identifica varios paquetes de diseño de clases:

Modelo: Paquete compuesto por la factoría de sesiones de Hibernate, los DAO para la gestión de persistencia y los POJOS necesarios.

El diseño de clases que conforman la factoría de Hibernate y los DAOs para la gestión de persistencia contiene 9 clases, misma que se presentan en la **Figura IV. 18**, en la que se puede observar que todos los DAOs heredan los métodos de la clase DAO, y esta a su vez hace uso de los métodos implementados en la clase HibernateSessionFactory

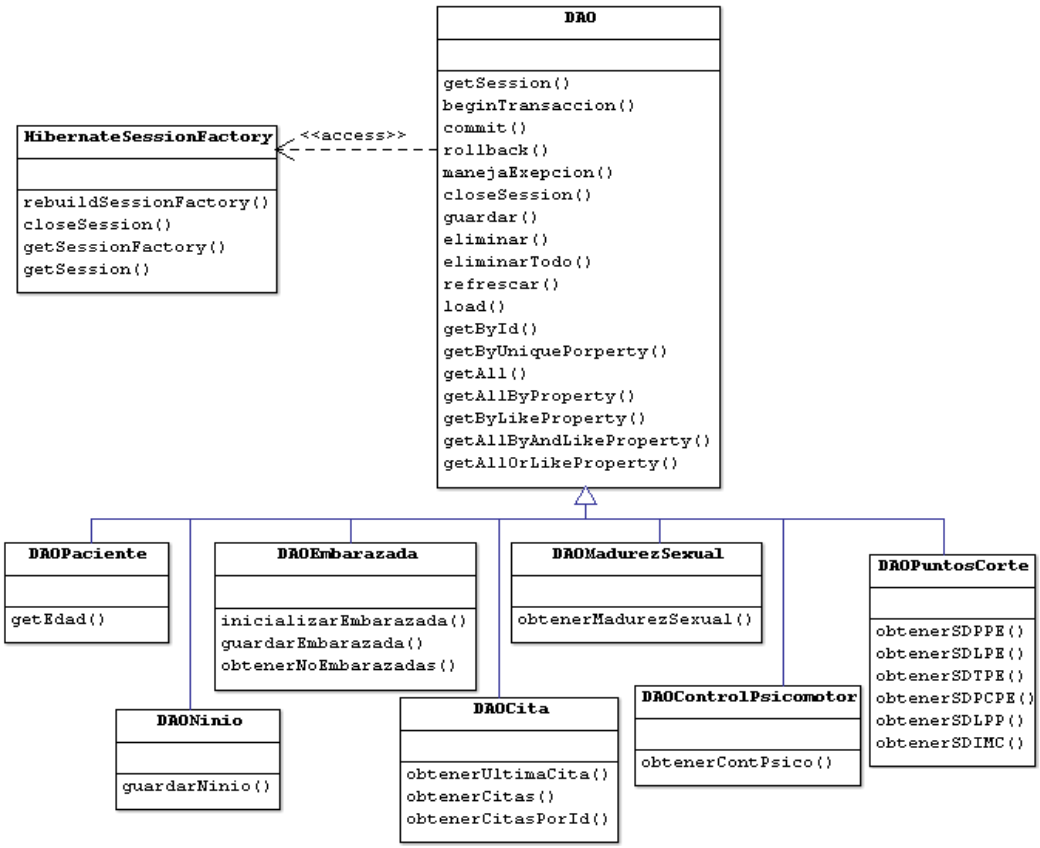


Figura IV. 18. Diseño de Clases – Paquete Modelo

[61] Manual Técnico: Anexo G – Información CD NutriSys

Para construir los POJOS [61] se consideró la organización de los módulos de Gestión de Usuarios, Configuración del Sistema y Gestión de Datos del Paciente; cabe mencionar que cada una de las clases implementadas tiene las propiedades accesibles mediante los setters y los getters, aunque éstas no se encuentren descritas en los modelos.

Una vez desarrollados los diseño de clases, se obtuvo que: el módulo Gestión de Usuarios está conformado por 1 clase, el módulo de Configuración del Sistema por 22 clases y el módulo de Gestión de Datos del Paciente por 18 clases, teniendo un total de 41 POJOS.

Control: Paquete compuesto por los Servicios [61] que configuran las órdenes de control, mismo que contiene 15 clases, de las cuales la mayoría hereda del Servicio Genérico y estas a su vez del Servicio Base. En el Servicio Base se encuentran desarrollados los métodos comunes para las clases que hacen uso de ésta.

Beans [61]**:** Paquete compuesto por clases de java que se asocian a los formularios de IceFaces, éstos son objetos de respaldo que contienen código que se ejecuta detrás de la página.

Modelo Lógico de la Base de Datos

A partir del diagrama de clases del paquete Modelo en lo que corresponde a los POJOS, se crea el modelo lógico de la base de datos [62]. En la **Figura IV. 19** se muestra parte del diseño**.**

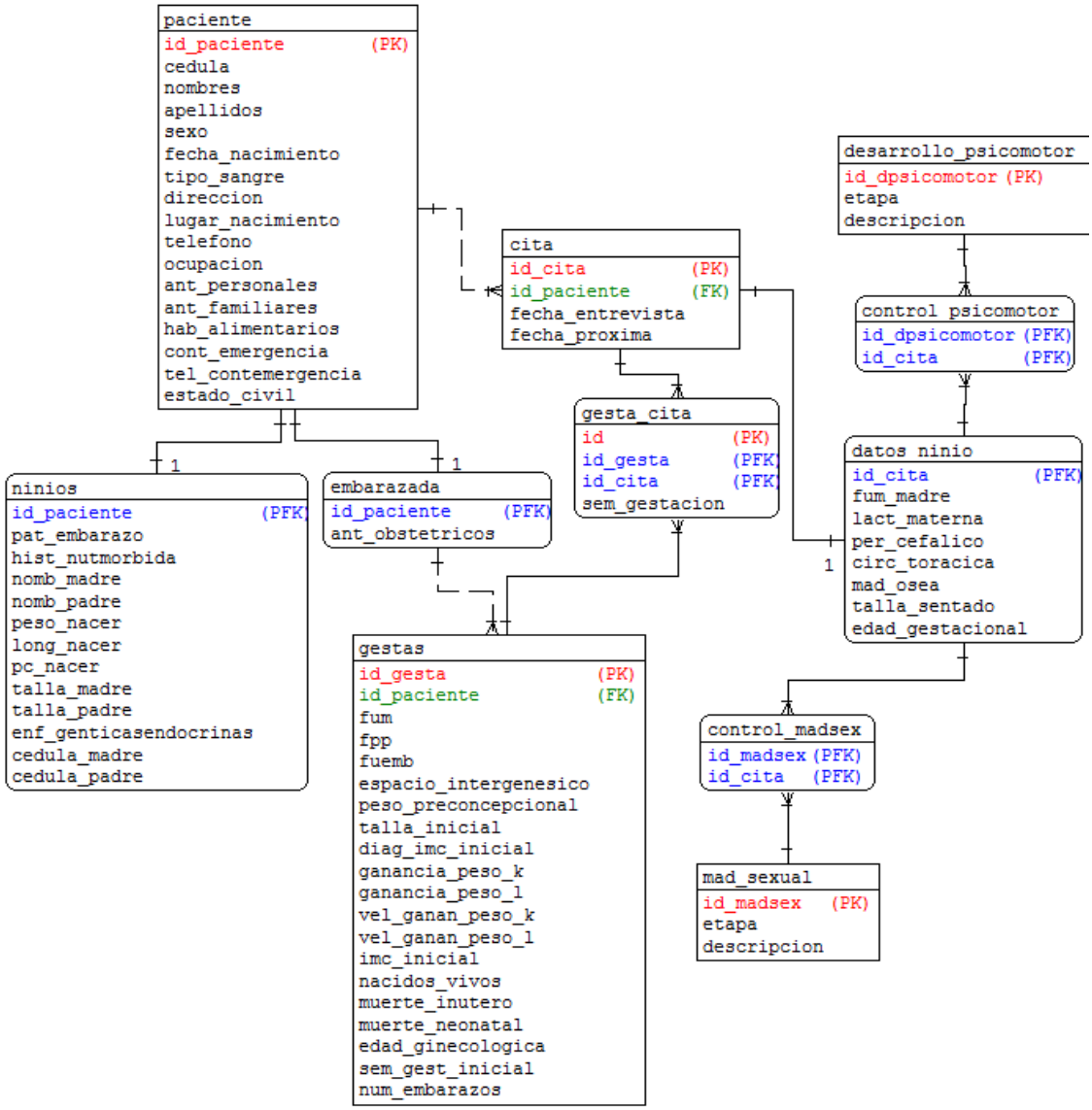


Figura IV. 19. Modelo Lógico de la Base de Datos

[62] Manual Técnico: Anexo H – Información CD NutriSys

Diseño de Interfaces de Usuario

Las interfaces de usuario se diseñaron en base a los requerimientos funcionales previamente establecidos, por tal motivo para la implementación de la aplicación se considera las siguientes interfaces:

Interfaz de autenticación: Esta funcionalidad permitirá realizar la autenticación del usuario con la cédula y la contraseña.

Interfaz principal.- Para facilidad de los usuarios se mostrará una lista de las funcionalidades a realizar, éstas se mostrarán en forma de vínculos, que estarán identificadas por un nombre que describa el uso del sistema, que al seleccionar llevará al usuario a realizar las funcionalidades pertinentes dependiendo del vínculo elegido.

Todas las interfaces se desarrollarán basándose en la estructura de la **Figura IV. 20**, en donde la sección del contenido será la que cambiará en base a las acciones realizadas.

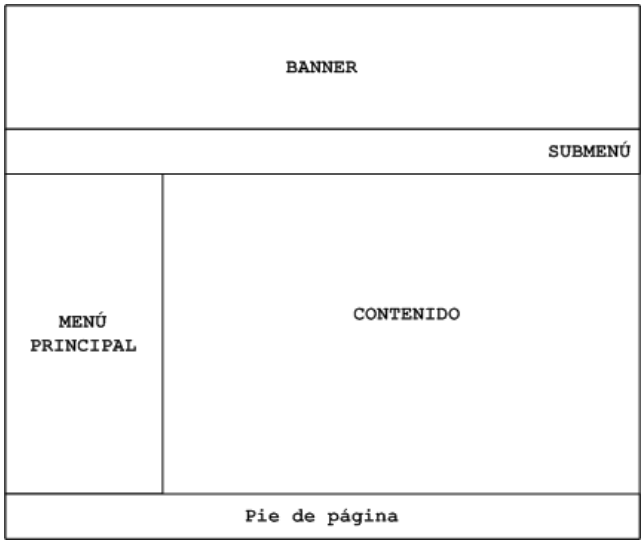


Figura IV. 20. Interfaz Principal

Interfaz de administración de usuarios.- En esta funcionalidad el sistema permitirá la creación de un nuevo usuario.

Se podrá realizar la búsqueda de los usuarios considerando como criterio parte de la cédula, nombres o apellidos.

Luego de elegir el usuario se podrá modificar todos los campos que el administrador que desee.

Interfaz de administración de pacientes.- En esta funcionalidad la aplicación permitirá crear un nuevo paciente.

Se podrá realizar la búsqueda del paciente considerando parte de la cédula, de los nombres o apellidos.

Una vez seleccionado el paciente se podrá modificar sus datos, así como también realizar las evaluaciones nutricionales y visualizar su historia antropométrica.

Interfaz de administración de evaluaciones nutricionales.- En esta funcionalidad el sistema permitirá evaluar el estado nutricional y bioquímico del paciente.

Si la evaluación nutricional se ha realizado en un rango de 0 a 24 horas, el sistema cargará la última cita nutricional para permitir su modificación.

Interfaz de administración de patrones de crecimiento.- En esta funcionalidad el sistema permitirá la gestión de las tablas de los patrones de crecimiento.

Para realizar la actualización de cada una de las tablas de los patrones de crecimiento, se deberá primero elegir de una lista desplegable el tipo de tabla y posteriormente a esto el sistema permitirá subir un archivo de texto plano, para la modificación de los datos y de las desviaciones estándar.

Interfaz de consumo de un paciente de la ESPOCH.- Esta funcionalidad permitirá buscar los datos de un paciente registrado en el OASIS y RRHH, atribuyendo los datos existentes en el CAIS.

Interfaz de reportes.- Esta funcionalidad permitirá seleccionar el tipo de reporte a visualizar.

De acuerdo a lo expuesto anteriormente, se realizó un prototipo de las interfaces que se utilizará en el desarrollo del sistema, mismas que se pueden observar en el SRS [63].

4.2.1.3 DISEÑO FÍSICO

Esta es la última etapa de la fase de planificación, en ésta se describirá las actividades, componentes, servicios y tecnologías de la solución.

Diagramas de actividades

Mediante los diagramas de actividades [64] se describirá el flujo de trabajo desde el inicio hasta el final, detallando todas las posibles rutas de decisiones existentes durante el proceso de la aplicación. En esta sección se detallarán solamente los diagramas de actividad para la creación de un paciente, modificación de la última cita y el registro de una tabla, en vista de que para estos procesos se requiere de una comprensión que ayude y facilite el desarrollo de la aplicación.

Diagrama de componentes

En la **Figura IV. 21**, se presenta el diagrama de componentes que detalla la estructura del Sistema Nutricional de la ESPOCH "NutriSys", en el que se puede observar que el paquete de Gestión de

[63] Manual Técnico: Anexo A – Información CD NutriSys
[64] Manual Técnico: Anexo I – Información CD NutriSys

Usuarios está compuesto por el componente Usuario, el paquete Gestión de Datos del Paciente por los componentes Paciente y Evaluación, y el Paquete Gestión de Datos del Sistema por el componente Tabla.

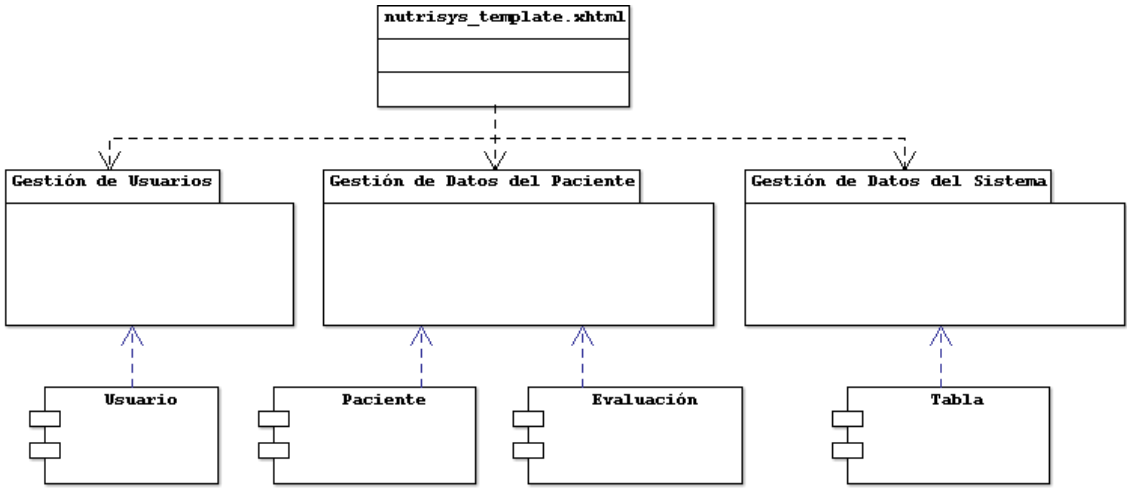


Figura IV. 21. Diagrama de Componentes

Diagrama de despliegue

La aplicación se diseñará para que la capa de presentación, negocio y acceso a datos se ejecute en el Servidor GlassFish, tal como se muestra en la **Figura IV. 22**.

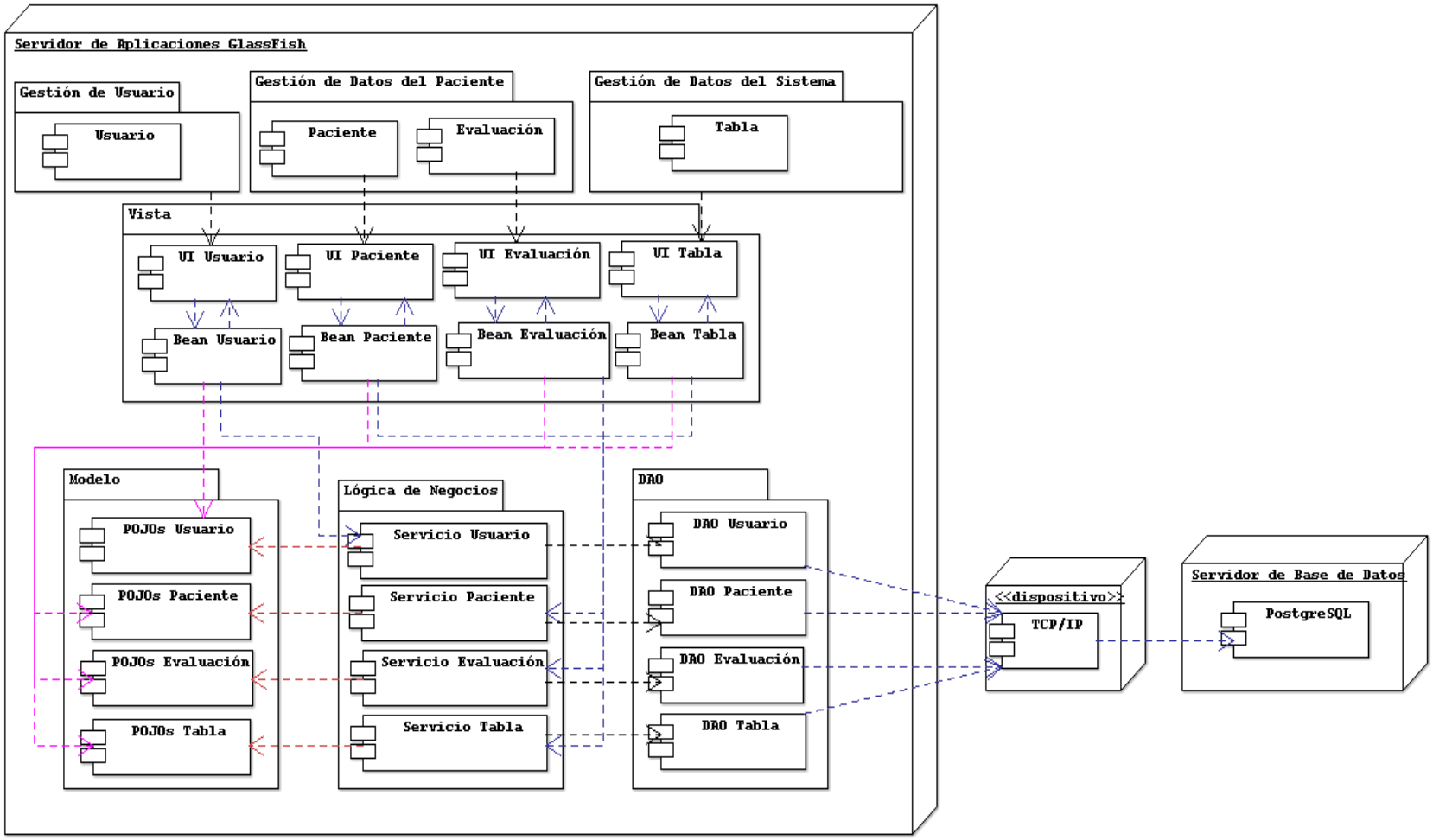


Figura IV. 22. Diagrama de Despliegue

Modelo Físico de la Base de Datos

Al crear el modelo lógico de la base de datos del sistema nutricional con el software Toad Data Modeler, las relaciones N:M convierten al modelo entidad-relación en un modelo relacional normal, dando un total de 42 tablas que se implementarán en la base de datos de "NutriSys".

Al generar el script [65] con la herramienta antes mencionada se obtuvo un archivo SQL con las tablas y los tipos de datos a ejecutar en PostgreSQL**.**

4.3 FASE III. DESARROLLO

Durante la fase de desarrollo se creará la aplicación en sí. Esta fase incluye la construcción del código, así como su documentación y la implementación de la solución.

4.3.1 NOMENCLATURA Y ESTÁNDARES PARA EL DESARROLLO

En esta sección se definirá los estándares o convenciones de programación a tomarse en cuenta para el desarrollo del Sistema Nutricional de la ESPOCH "NutriSys", estos patrones están basados en la recomendación de Sun Microsystem.

La nomenclatura utilizada se aplicará a la organización de los ficheros, al fichero fuente java, a las declaraciones de clases, al estándar Java Bean, a las declaraciones generales, a la especificación XHTML y a la creación de las tablas de la base de datos.

[65] Manual Técnico: Anexo K – Información CD NutriSys

Organización de los ficheros.- Las clases se agruparán en paquetes y éstos serán organizados de manera jerárquica, de forma que todo el código desarrollado para el Sistema Nutricional de la ESPOCH estará incluido dentro del paquete **"espoch.nutrisys"**, y dentro de éste las clases se organizarán en sub paquetes en función a la sección a la que pertenecen. Por ejemplo para el desarrollo de los DAOs, las clases se incluirán en el paquete **"espoch.nutrisys.dao"**.

Fichero fuente Java (.java).- Cada fichero fuente Java debe contener una única clase o interfaz pública. El nombre de la clase debe coincidir con el nombre el fichero. En todo fichero fuente java se debe distinguir la sentencia de paquete, las sentencias de importación y la declaración de clases e interfaces.

Declaraciones de clases.- Esta nomenclatura incluye: clases, métodos variables y constantes. A continuación se describe los elementos que componen una clase o interfaz:

- **Clases:** Para todo nombre de clase, la primera letra debe ser mayúscula, si se trata de una combinación de palabras se debe intercalar entre mayúsculas y minúsculas, este mecanismo de nombre se le conoce como **camelCase**. Por ejemplo DesarrolloPsicomotor.
- **Métodos:** Para los métodos de las clases la primera letra debe ser minúscula, si son varias palabras se debe intercalar entre minúsculas y mayúscula. Por ejemplo setNombres.
- **Variables:** Para las variables, se aplica el caso de los métodos, en donde la primera letra es minúscula y las demás se deben guiar por el mecanismo de camelCase, lo que es importante destacar, es que estos nombres deben ser cortos y descriptivos. Por ejemplo campoBusqueda.
- **Constantes:** Para las constantes, el nombre debe ser escrito completamente en mayúsculas, y para la separación entre palabras se debe utilizar el underscode/guión bajo (_). Por ejemplo VALOR_MAX.

Estándar Java Bean.- Todos los Java Beans tendrán las propiedades y métodos **setters** y **getters**, pues la característica principal es que todas las propiedades son privadas, y los métodos antes mencionados permiten el acceso a las mismas. Por ejemplo getNombre().

Declaraciones generales.- Entre las recomendaciones a tomarse en cuenta para las declaraciones se tiene:

- La declaración se debe realizar una por línea.
- Toda variable local será inicializada en el momento de su declaración, salvo que esta dependa de algún valor calculado previamente.
- Las declaraciones deben situarse en el principio de cada bloque principal en el que se utilicen, y nunca en el momento de su uso.

Especificación XHTML.- Las reglas de sintaxis que se aplicarán al desarrollo de los documentos XHTML de la solución son:

- Todo se escribirá en minúsculas.
- Todos los elementos tienen que llevar cierre, incluso los elementos vacios que habitualmente no lo tienen, como
 o <HR> que pasan a escribirse
 <hr />
- Todos los parámetros deben ir entre comillas. Por ejemplo: <table border="1"> Las comillas pueden ser dobles o simples (pero siempre formando parejas con las mismas).
- Los elementos que tienen cierre, pero que si no se utilizan no dan problemas en HTML, como por ejemplo , en XHTML deben utilizarse siempre los cierres. Es válido   pero no es válido

- Los elementos anidados deben cerrarse ordenadamente. Es válido en HTML: Texto en negrita y cursiva En XHTML debe ser: Texto en negrita y cursiva

Estándares del diseño de la base de datos.- Las reglas generales que se utilizarán en el diseño de la base de datos de la aplicación son:

- Los nombres de los campos y tablas se escribirán con letras minúsculas, si el nombre está compuesto por más de dos palabras se pondrá el guión bajo (_) entre ellas.
- Únicamente se utilizarán caracteres alfabéticos, salvo el caso que el nombre necesite dígitos numéricos.
- Las letras acentuadas se reemplazarán por su equivalente no acentuadas, y en lugar de la letra eñe (ñ) se utilizará (ni).
- El nombre debe ser lo más descriptivo posible.

4.3.2 CAPA DE PRESENTACIÓN

En la capa de presentación se detalla el desarrollo de los componentes relacionados con las interfaces de usuario. Para la implementación de la capa de presentación del Sistema Nutricional de la ESPOCH “NutriSys” se desarrolló con las páginas XHMTL y estas a su vez se integran con los Java Beans.

4.3.2.1 Implementación de Interfaces de Usuario con XHMTL y Framework IceFaces

Para el desarrollo de las interfaces de usuario se utilizó el framework Facelet para la creación de la plantilla principal, y en el contenido de ésta se presentarán las diferentes funcionalidades del sistema según el vínculo que el usuario seleccione.

Los matices utilizados para el diseño de la aplicación se los seleccionó en base al logo del sistema que en su mayoría está representado por el color RGB (107, 191, 23).

Página de inicio, diseñada para la validación de la cédula y contraseña del usuario en el sistema, misma que se puede apreciar en la **Figura IV. 23**.

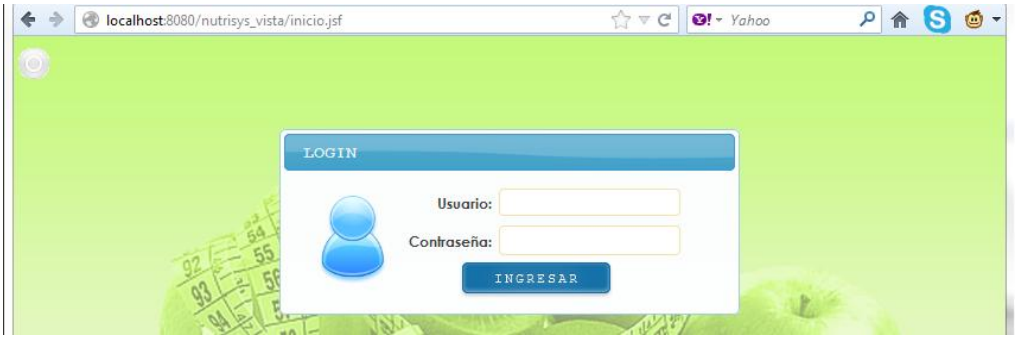


Figura IV. 23. Interfaz de Autenticación de Usuario

Plantilla principal, que permite estructurar a la página en tres partes diferentes: banner, menú y contenido. En la **Figura IV. 24** se ilustra el diseño de la plantilla principal, en donde se visualiza el banner en la parte superior, el menú en el fragmento izquierdo de la plantilla y el contenido a la derecha de la página, en base a este diseño se realizó todas las demás interfaces, considerando que el área a cambiar será solamente el contenido.

Figura IV. 24. Interfaz Principal

Página de entrada de datos, que permite el ingreso o modificación de la información. En la **Figura IV. 25**, se observa que en la parte superior se describe con un título la representación de los datos y el número de historia clínica del paciente, siguientemente a esto se tiene los datos similares agrupados en paneles, también se diferencia claramente que las etiquetas se encuentra alineadas a la parte izquierda y los datos de entrada ordenados a la derecha, además se distingue que el tamaño es el mismo para todos los componentes similares, y por último se observa la distribución horizontal en forma simétrica de los botones.

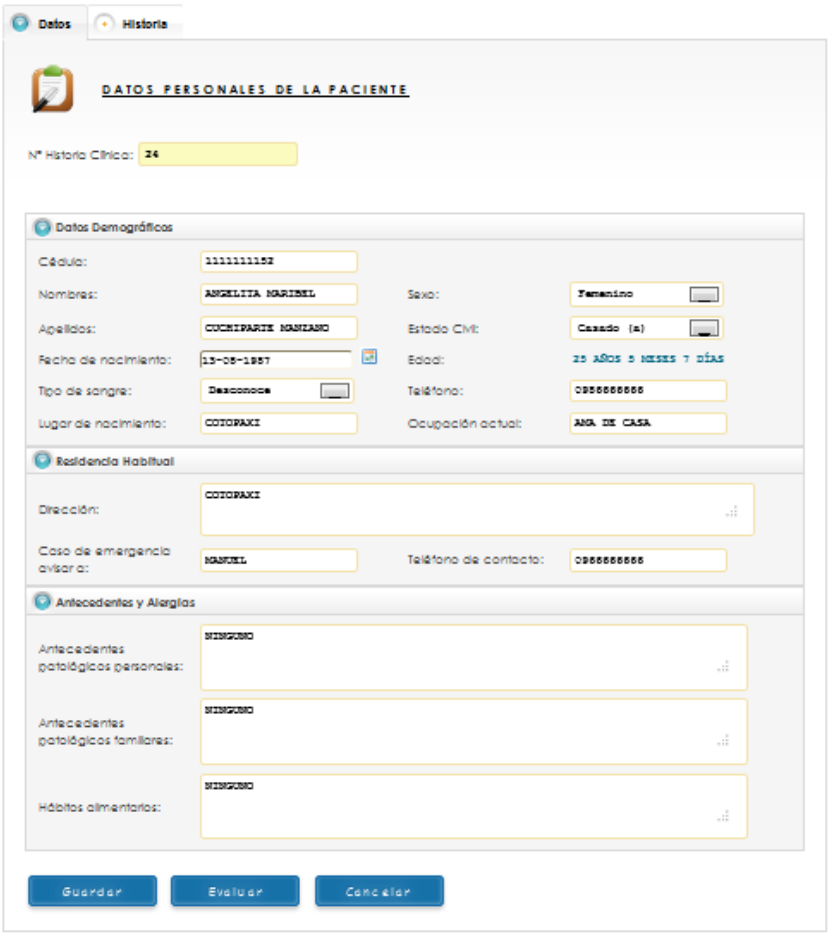


Figura IV. 25. Interfaz de Entrada de Datos

Página de evaluación nutricional (**Figura IV. 26**) al igual que la anterior asume las mismas características, con la diferencia que ésta se visualizará en un diálogo modal.

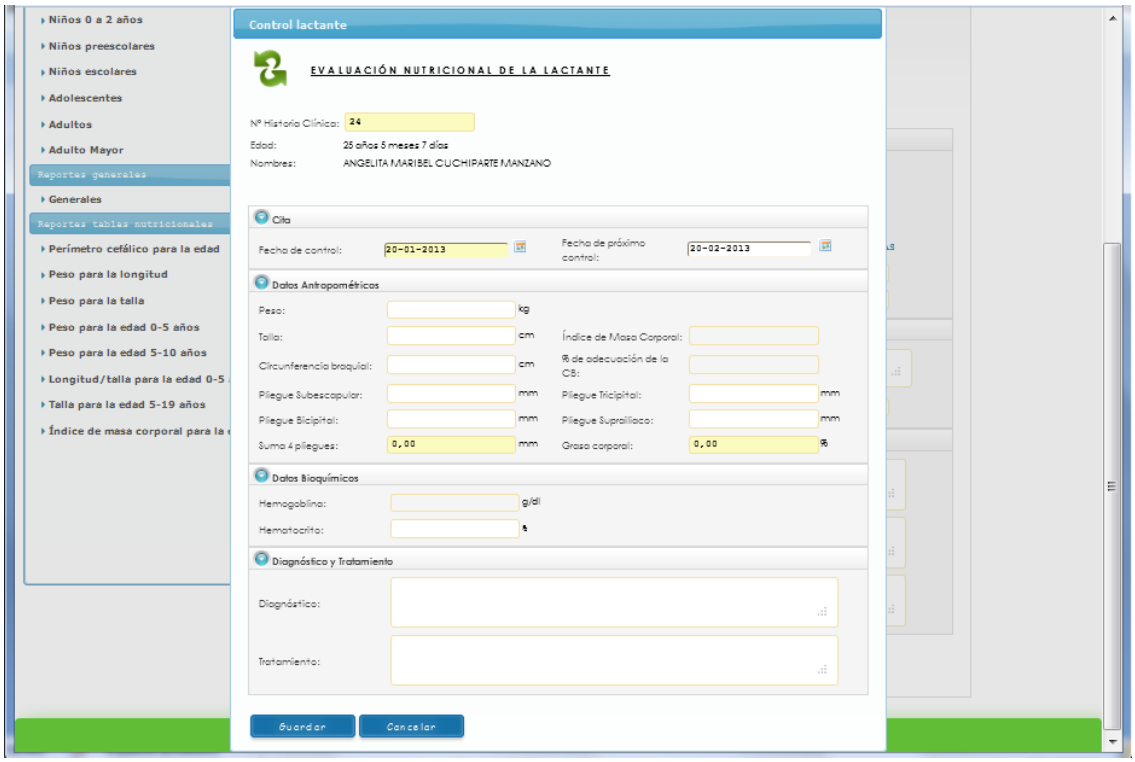


Figura IV. 26. Interfaz de Evaluación Nutricional

Gráfica nutricional, diálogo modal que permite mostrar a los usuarios los datos nutricionales del paciente. En la **Figura IV. 27,** se distingue el título de la gráfica en la parte superior, siguientemente las desviaciones estándar y la curva nutricional, y por último al pie de la página la descripción y el botón cerrar. Los colores de las desviaciones estándar son establecidos por los lineamientos nutricionales.

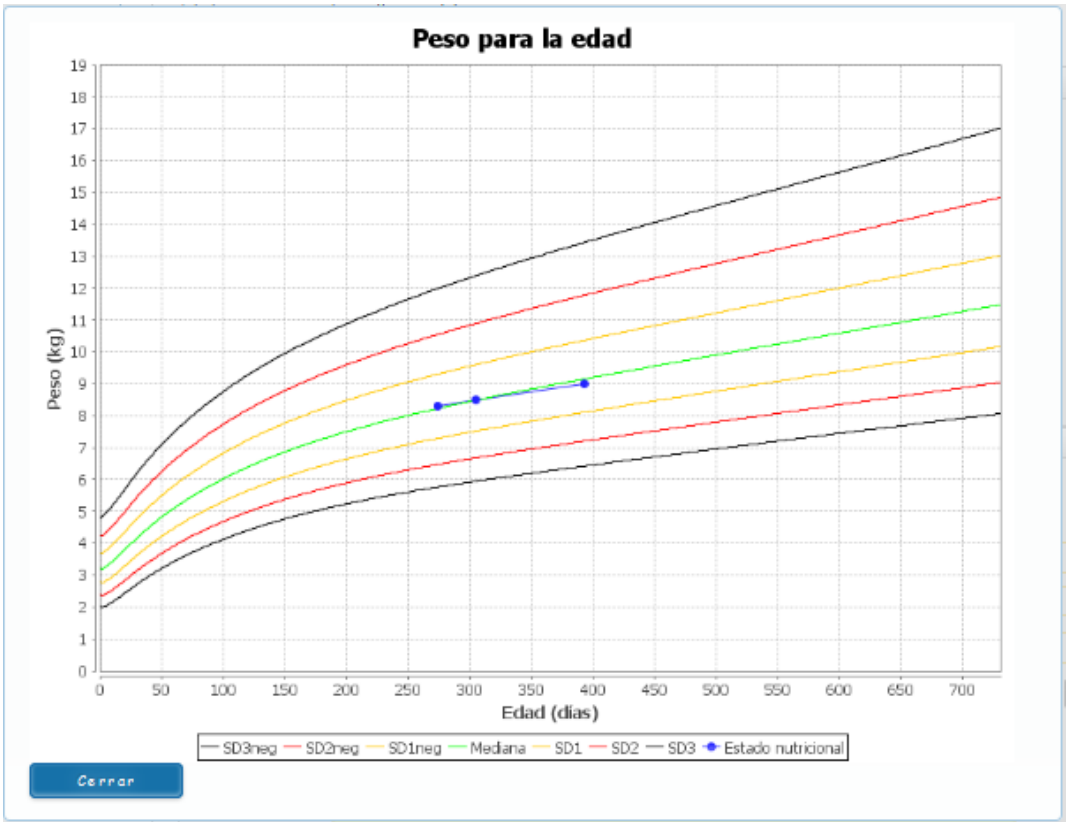
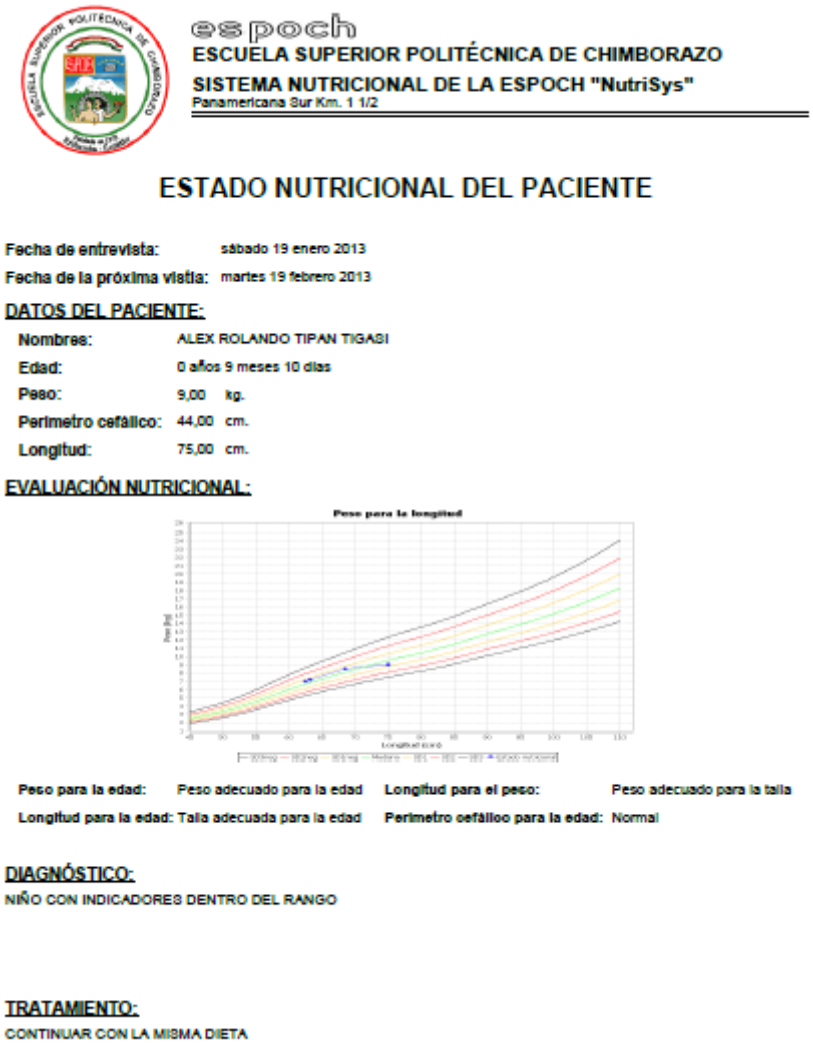


Figura IV. 27. Interfaz de la Gráfica Nutricional

Al finalizar las páginas XHTML se diseñó los reportes, mismos que tendrán en común el encabezado con la distinción de la institución para la cual se realizó el sistema nutricional y seguidamente el título del reporte.

Reporte del estado nutricional, página que permite visualizar los datos nutricionales del paciente. En la **Figura IV. 28,** además de lo antes mencionado, se observa los datos personales del paciente, después se tiene los datos nutricionales seguidos de la gráfica, y por último el diagnóstico y tratamiento respectivo.


espoch
ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
SISTEMA NUTRICIONAL DE LA ESPOCH "NutriSys"
Panamericana Sur Km. 1 1/2

ESTADO NUTRICIONAL DEL PACIENTE

Fecha de entrevista: sábado 19 enero 2013
Fecha de la próxima visita: martes 19 febrero 2013

DATOS DEL PACIENTE:

Nombres:	ALEX ROLANDO TIPAN TIGASI
Edad:	0 años 9 meses 10 días
Peso:	9,00 kg.
Perímetro cefálico:	44,00 cm.
Longitud:	75,00 cm.

EVALUACIÓN NUTRICIONAL:

Peso para la edad: Peso adecuado para la edad
Longitud para el peso: Peso adecuado para la talla
Longitud para la edad: Talla adecuada para la edad
Perímetro cefálico para la edad: Normal

DIAGNÓSTICO:
NIÑO CON INDICADORES DENTRO DEL RANGO

TRATAMIENTO:
CONTINUAR CON LA MISMA DIETA

Figura IV. 28. Interfaz del Reporte del Estado Nutricional

Reporte de listados, página que permite visualizar la información linealmente. En la **Figura IV. 29**, se observa que además de la estructura principal, el reporte presenta un título de los datos a mostrar, mismo que se encuentran dentro de un fondo plomo con las letras de color blanco, y por último la información a visualizar dividida por líneas horizontales.

Figura IV. 29. Interfaz de Reporte de Listados

Reporte estadístico, página que permite visualizar la información mediante un gráfico de pastel. En la **Figura IV. 30**, se observa un reporte con las características antes mencionadas en el encabezado y título, posteriormente se presenta las fechas de inicio y corte de la información, luego el detalle estadístico seguido del gráfico de pastel, y por último se tiene la descripción representativa de los colores del pastel.

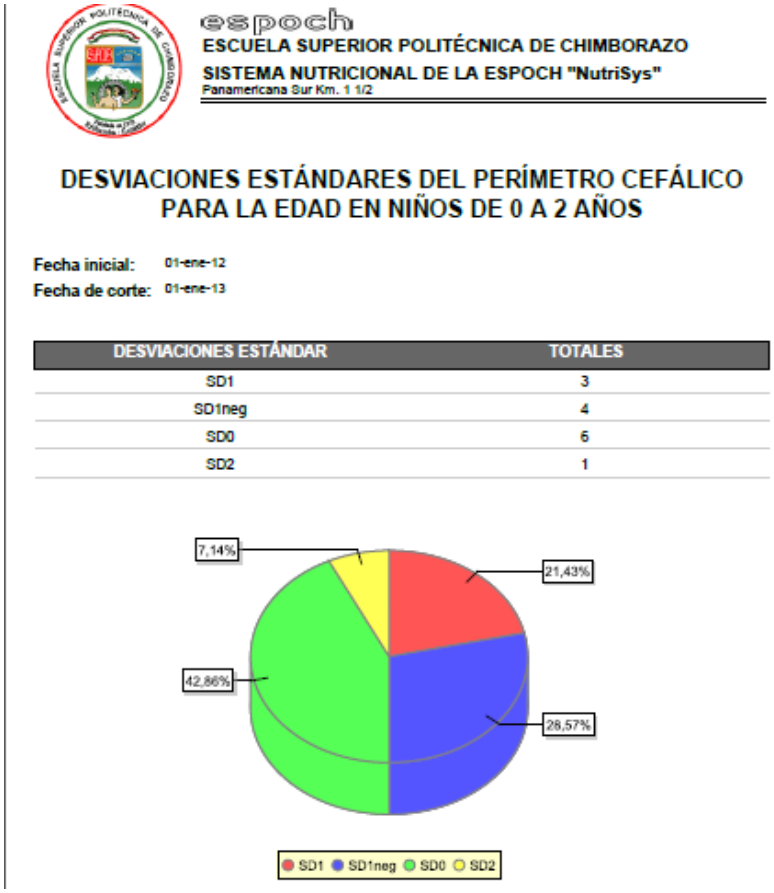


Figura IV. 30. Interfaz de Reporte Estadístico

Al finalizar el desarrollo de las interfaces de usuario se obtuvo 76 páginas y 46 reportes aproximadamente, en donde se codificó componentes tales como: formularios, controles de comando (botones, enlaces), controles de selección múltiple (listas desplegables de selección múltiple), controles de texto (campos de texto, campos de texto con caracteres ocultos, cajas de

texto multilínea), diálogos modales, menús de opciones, calendarios, paneles, paneles de pestaña, paneles divisibles, listas paginadas, gráficas y control de archivos.

4.3.2.2 Implementación de Beans

Los Java Beans son elementos característicos de una aplicación JSF, que a más de encapsular los datos procedentes de la capa cliente, también incluye los métodos de respuesta que ésta solicita.

Al culminar con el desarrollo de los Java Beans se obtuvo un total de 37 clases, en los que se implementó funciones tales como se detallan a continuación:

Eventos de acción, la acción de un botón o de un enlace que se genera con un control de comando, provoca un submit y un evento de acción al que se le puede relacionar un método escuchador.

ValueChangeEvent, este se método se produce sobre las listas de selección, fechas o cajas de texto y tiene lugar cuando se origina un cambio en el dato de entrada del componente.

Validadores personalizados, son métodos de validación, cuyo objeto es adaptar los datos escritos en la interfaz de usuario a los datos y tipos implementados en la aplicación.

4.3.3 CAPA LÓGICA DE NEGOCIOS

En esta capa se implementó los Servicios para centralizar la obtención de la información, actuando de manera intermedia entre la capa de presentación y la capa de persistencia. Se desarrollaron 29 Servicios con sus respectivas clases.

4.3.4 CAPA DE PERSISTENCIA

Para acceder a los datos se desarrolló el patrón de diseño DAO. En esta capa se definieron los correspondientes DAOs para trabajar con el modelo.
La persistencia de los datos se implementó mediante Hibernate, por lo que se configuró los archivos necesarios y se crearon los POJOs correspondientes.

Al culminar con el desarrollo de la capa de persistencia se obtuvo 41 POJOs, 38 ficheros hbm.xml y 11 DAOs.

Los POJOs se crearon con el objetivo de asociarlos a cada tabla de la base de datos, los archivos de mapeo indican a Hibernate que tabla en la base de datos debe ser accedida y que columnas deben usarse y los DAOs gestionan la persistencia de la base de datos PostgreSQL.

4.3.5 CAPA DE BASE DE DATOS

La creación de las tablas se realizó mediante Hibernate. Los parámetros de conexión a la base de datos se configuró en el archivo hibernate.cfg.xml, la ventaja de esto es hacer totalmente transparente a la base de datos, permitiendo cambiarla, sin la necesidad de modificar una sola línea de código.

Además el modelo de base de datos cuenta con vistas y funciones para su funcionamiento. Las vistas se implementaron con la finalidad de listar los pacientes tomando en cuenta la edad de los mismos, y las funciones se desarrollaron para la ejecución de los reportes.

4.4 FASE IV. ESTABILIZACIÓN

En esta fase se realizará pruebas, la carga e integración de la solución. A través de los escenarios de prueba se identificará, priorizará y resolverá los problemas, de tal manera que la solución esté preparada para su publicación.

4.4.1 REVISIÓN GENERAL DEL SISTEMA

Antes de proceder con el test del sistema se preparará el ambiente de pruebas, para esto se verificará que el código fuente de la aplicación cumpla con los estándares antes definidos, se comprobará que el script de la base de datos se ejecute sin problema alguno y que la documentación de la instalación sea un asistente eficiente para el administrador.

Código Fuente

Se comprobó que el código fuente del Sistema Nutricional de la ESPOCH "NutriSys" [66] cumpla con la nomenclatura antes especificada, para ello se observó la organización de los ficheros, las clases, los Beans, los métodos existentes en cada clase, las variables y los documentos XHTML.

Scripts de Base de Datos

Se verificó que los nombres de las tablas y de los campos de la base de datos de "NutriSys", cumpla con los estándares de diseño antes mencionado. Además se ejecutó el script de la base de datos [66] inicial en PostgreSQL para corroborar su correcto funcionamiento.

[66] Manual Técnico: Anexo K – Información CD NutriSys

Documentación de la Instalación

Como forma de comprobación de la documentación de la instalación se capacitó al administrador de "NutriSys" utilizando el manual de instalación [67], por lo que se verificó paso a paso el proceso de instalación de PostgreSQL, GlassFish y el despliegue de la aplicación.

4.4.2 PRUEBAS

Las pruebas son una etapa importante en la fase de estabilización del MSF, pues de ello depende entregar un software que cumpla con todo lo estipulado en los requerimientos funcionales de la aplicación.

Las pruebas se realizaron los días 10, 11 y 12 de diciembre del 2012 con el Ing. Jorge Menéndez (Testing Manager), éstas permitieron verificar el correcto funcionamiento de los flujos básicos y alternativos de todos los casos de uso a implementar en la iteración.

El Testing del proyecto, para el desarrollo de dicha actividad, llenó los casos de prueba [68], siguiendo el modelo que se muestra en la **Tabla IV. XXVII**, donde se define: el código, como un consecutivo; el nombre, que se corresponde con el nombre del caso de uso; las condiciones, las condiciones de ejecución; los pasos de ejecución, lista ordenada de los pasos que permitirán verificar el caso de prueba; resultados esperados, la respuesta que devuelve el sistema; evaluación de prueba, el estado obtenido al finalizar la comprobación del caso de prueba.

[67] Manual de Instalación – Información CD NutriSys
[68] Manual Técnico: Anexo J – Información CD NutriSys

Al culminar con la construcción de todos los casos de prueba se obtuvo un total de 50. En la **Tabla IV. XXVII.** se define el caso de prueba para crear un usuario en su flujo básico.

Tabla IV.XXVII. Caso de Prueba 1 Crear Usuario

PRUEBA DE ACEPTACIÓN			
CÓDIGO:	1	**NOMBRE:**	Crear Usuario
DESCRIPCIÓN:	Probar el flujo básico para crear un usuario.		
CONDICIONES:	Para crear un usuario se escribieron los datos del mismo.		
PASOS DE EJECUCIÓN:	1. Menú principal, seleccionar Administración. 2. Seleccionar Nuevo Usuario 3. Ingresar los valores correctamente 4. Seleccionar Guardar		
RESULTADO ESPERADO:	Se muestra un mensaje de confirmación que se ha creado el usuario.		
EVALUACIÓN DE PRUEBA:	Exitosa.		

De los casos de prueba realizados se obtuvo que el 86% de ellos fueron exitosos, por tanto el 14 % resultaron no exitosos, los cuales se verificarán nuevamente para obtener los resultados esperados.

4.5 FASE V. DESPLIEGUE

En esta fase se implementa la solución y se comprueba que sea estable y utilizable. Para la implementación de la solución se debe contar con un plan de operación y soporte.

Estrategia de implantación

Se realizó el despliegue de la aplicación con la persona asignada por el Departamento de Sistemas y Telemáticas de la ESPOCH, con el respectivo manual de instalación [69].

Soporte

Se informó a los usuarios del sistema acerca del despliegue de la solución y la dirección de acceso a la misma.

[69] Manual de Instalación – Información CD NutriSys

Durante la fase de despliegue, se brindó soporte a los usuarios que experimentaron problemas en el acceso a la aplicación.

Coordinación del entrenamiento

Se entrenó al personal de soporte, de tal manera que éste tenga los conocimientos necesarios para brindar soporte y ayuda.

Se capacitó a los usuarios finales mediante el manual respectivo [70].

Transferencia a operaciones

Una vez que finalizó el despliegue de la solución, se firmó un documento de entrega del sistema a la persona calificada. En este documento es una formalidad de transferencia de la responsabilidad de la aplicación al administrador.

[70] Manual de Usuario – Información CD NutriSys

C O N C L U S I O N E S

Al finalizar el análisis comparativo entre los Frameworks MyFaces Trinidad, IceFaces y RichFaces aplicado al Sistema Nutricional de la ESPOCH se han obtenido las siguientes conclusiones:

- El seleccionar el mejor framework de interfaz de usuario entre MyFaces, IceFaces y RichFaces, no pretende desplazarlos entre sí, la elección depende de las características que persigue el desarrollo del Sistema Nutricional de la ESPOCH "NutriSys"

- Al culminar el análisis comparativo se obtuvo que IceFaces alcanzó el 94,78%, MyFaces Trinidad el 76,21% y RichFaces el 56,71%, con un margen de confianza del 95%.

- La implementación del Sistema Nutricional de la ESPOCH "NutriSys" con el framework IceFaces permitió al desarrollador realizar eficientemente la aplicación, por las ventajas que éste presta, en cuanto se refiere al aprendizaje, facilidad para el desarrollo y por la suma de componentes de interfaz de usuario que el framework proporciona

- Los frameworks de JSF tales como: MyFaces Trinidad, IceFaces y RichFaces, están basados en la arquitectura MVC, misma que permite estructurar a las aplicaciones en tres partes bien diferenciadas: la lógica de control, la lógica de negocio y la lógica de presentación.

- La selección del framework más eficiente permitió desarrollar el Sistema Nutricional de la ESPOCH, cumpliendo de esta manera con todos los requisitos solicitados por la Escuela de Nutrición y Dietética de la ESPOCH.

R E C O M E N D A C I O N E S

- El estudio nutricional del ser humano abarca distintos ámbitos, entre los que se destaca la evaluación nutricional y la dieta respectiva del paciente. El Sistema Nutricional de la ESPOCH realiza solamente la evaluación nutricional del paciente, por lo que se recomienda desarrollar el módulo de dietas, con el propósito de suministrar un servicio completo al paciente al momento de evaluar su estado nutricional.

- Las actualizaciones del framework IceFaces son liberadas continuamente, por tanto se recomienda analizar la posibilidad de migrar a las últimas versiones, con la finalidad de obtener una aplicación estable.

- Es necesario realizar un análisis previo de los requerimientos de la solución, pues éstos permitirán optimizar los tiempos de desarrollo, y a la vez entregar una aplicación estable y utilizable.

- En todo proyecto es recomendable realizar una correcta planificación, por ende, seleccionar una metodología que permita minimizar los riesgos, aumentar la calidad y acortar el tiempo de implementación ayuda en gran medida al equipo de trabajo.

- Se recomienda utilizar Hibernate, éste es un framework de persistencia que se integra correctamente con IceFaces, permitiendo reducir el tiempo de desarrollo, debido a que posee capacidades para obtener y almacenar datos en la base de datos.

R E S U M E N

El análisis comparativo de los frameworks MyFaces Trinidad, IceFaces y RichFaces, se realizó con el objetivo de seleccionar el más eficiente que permita desarrollar un Sistema Nutricional para ser aplicado en la Escuela Superior Politécnica de Chimborazo.

Se evaluaron los diferentes frameworks a través de parámetros analizados a base a dos consideraciones: primera, experimentación e investigación directa; y la segunda, encuestas a los programadores del Departamento de Sistemas y Telemática de la ESPOCH. Una vez tabulados los resultados, se aplicó chi cuadrado para la demostración de la hipótesis, rechazando la hipótesis nula y aceptando la de la investigación.

Luego de realizar el análisis antes mencionado se obtuvo que: IceFaces alcanzó el 94,78%; MyFaces Trinidad, el 76,21%; y, RichFaces el 56,71%; por lo que se seleccionó el framework IceFaces por las ventajas que éste presta, en cuanto se refiere, al aprendizaje, calidad, facilidad para el desarrollo y la suma de componentes de interfaz de usuario que pone a disposición de los desarrolladores.

Se desarrolló e implantó el Sistema Nutricional para la ESPOCH, el mismo que muestra las características y funcionalidades que presta IceFaces, siguiendo la Metodología Microsoft Solution Framework.

Se recomienda la utilización de este framework a los programadores para aplicaciones similares en otras instituciones.

S U M M A R Y

The comparative analysis of the MyFaces Trinidad, IceFaces, and RichFaces frameworks was performed to select the most efficient one which permits to develop a Nutritional System to be applied in the Chimborazo Polytechnic School.

The different frameworks were evaluated through parameters analyzed on the base two considerations: 1st, direct experimentation and research; 2nd, surveys to the programmers of the ESPOCH System and Telematic Department. After the results were tabulated, square chi was applied for the hypothesis demonstration, rejecting the null hypothesis and accepting that of the research.

After the analysis performance these were the results: IceFaces reached the 94,78%; MyFaces Trinidad, the 76,21%; and, RichFaces, the 56,71%. Due to this, the IceFaces framework was selected for the advantages related to learning, quality, development easiness, and the sum of components of user interfaz that are available to the developers.

The Nutritional System for the ESPOCH was developed and implanted and shows the characteristics and functionalities of the IceFaces by following the Microsoft Solution Framework Methodology.

The use of this framework is recommended to the programmers for similar applications in other institutions.

Anexo

A

ANÁLISIS COMPARATIVO DE LOS FRAMEWORKS MYFACES, ICEFACES Y RICHFACES.

SECCIÓN 1: Aprendizaje.- Disponibilidad de información

Disponibilidad de información.- Internet

FRAMEWORK	PÁGINAS EN ESPAÑOL	LA WEB	PROMEDIO
MyFaces Trinidad	2.880 resultados en 0,19 segundos.	548.000 resultados en 0,10 segundos.	275.440
ICEFaces	52.500 resultados en 0,15 segundos.	153.000 resultados en 0,08 segundos.	102.750
RichFaces	333.000 resultados en 0,16 segundos.	2.100.000 resultados en 0,09 segundos.	1.216.500

A continuación se muestra las reglas de 3 necesarias para conseguir el resultado de los frameworks MyFaces Trinidad e ICEFaces, tomando en cuenta que 1.216.500 corresponde al máximo valor.

Para MyFaces Trinidad:

$$\frac{275.440 \times 50}{1.216.200} = 11{,}32\%$$

Para ICEFaces:

$$\frac{102.750 \times 50}{1.216.200} = 4{,}22\%$$

Disponibilidad de información.- Manuales y documentos

FRAMEWORK	BOOKS	DESARROLLO WEB	PROGRAMACIÓN JAVA	PROMEDIO
MyFaces Trinidad	8 resultados	8 resultados	5 resultados	7
ICEFaces	9 resultados	5 resultados	3 resultados	5,67
RichFaces	17 resultados	10 resultados	5 resultados	10,67

Para MyFaces Trinidad:

$$\frac{7 \times 50}{10{,}67} = 32{,}80\%$$

Para ICEFaces:

$$\frac{5{,}67 \times 50}{10{,}67} = 26{,}57\%$$

Disponibilidad de información.- Documentación oficial

MyFaces Trinidad: En la siguiente tabla, se ilustra la documentación oficial disponible para el framework MyFaces Trinidad, cuyo enlace oficial corresponde a: http://myfaces.apache.org/trinidad/index.html.

DOCUMENTACIÓN	IDIOMA
Documentación de etiquetas	Inglés
Selectores de estilos	Inglés
Guía del desarrollador	Inglés
Javadoc	Inglés
Demos	Inglés
TOTAL	5

ICEFaces: En la siguiente tabla**,** se expone la información oficial disponible para el framework ICEFaces v3.0, cuyo enlace oficial corresponde a: http://www.icesoft.org/

DOCUMENTACIÓN	IDIOMA
Notas de la versión	Inglés
Documentación de IceFaces	Inglés
Guía inicial de procedimientos	Inglés
Conceptos claves	Inglés
Core Framework	Inglés
Componentes ACE	Inglés
Componentes ICE	Inglés
Desarrollo Portlet	Inglés
Compatibilidad con ICEFaces 1.8	Inglés
Documentación API	Inglés
Tutoriales y ejemplos	Inglés
Demos	Inglés – Español
TOTAL	12

RichFaces: En la tabla siguiente**,** se expone la información oficial disponible para el framework RichFaces v4.2.0, cuyo enlace oficial corresponde a: http://www.jboss.org/richfaces

DOCUMENTACIÓN	IDIOMA
Guía de referencia RichFaces	Inglés
Documentación Generada	Inglés
Referencias CDK RichFaces	Inglés
Guías de Migración	Inglés
Código Fuente	Inglés
TOTAL	5

Para MyFaces Trinidad:

$$\frac{5 \times 50}{12} = 20{,}83\%$$

Para RichFaces:

$$\frac{5 \times 50}{12} = 20{,}83\%$$

A continuación se presenta un resumen de la variable disponibilidad de información.

FRAMEWORK	MYFACES TRINIDAD	ICEFACES	RICHFACES
Internet	11,32%	4,22%	50,00%
Manuales y documentos	32,80%	26,57%	50,00%
Documentación oficial	20,83%	50,00%	20,83%
TOTAL PROMEDIO	21,65%	26,93%	40,28%

SECCIÓN 2: Aprendizaje.- Soporte

MyFaces Trinidad: En la siguiente tabla, se muestra los enlaces de soporte disponibles que MyFaces Apache Trinidad posee.

DESCRIPCIÓN	ENLACE	IDIOMA
Foro de usuarios	http://old.nabble.com/MyFaces-f180.html	Inglés
FAQ y Soporte	http://myfaces.apache.org/trinidad/FAQ.html	Inglés
JIRA	https://issues.apache.org/jira/browse/TRINIDAD#selectedTab=com.atlassian.jira.plugin.system.project%3Aissues-panel	Inglés
TOTAL		3

ICEFaces: En la siguiente tabla, se ilustra los enlaces de soporte disponibles de ICEFaces.

DESCRIPCIÓN	ENLACE	IDIOMA
Chat	http://www.icesoft.org/	Inglés
Foros	http://www.icefaces.org/JForum/forums/list.page	Inglés
Seguimiento de problemas (Jira)	http://jira.icefaces.org/secure/Dashboard.jspa	Inglés
Wiki	http://wiki.icefaces.org/dashboard.action	Inglés
Webinars	http://www.icesoft.org/community/archived-webinars.jsf	Inglés
Foro de usuarios	http://www.icefaces.org/JForum/forums/show/12.page	Inglés
Blogs	http://blog.icefaces.org/blojsom/blog/	Inglés
Licencia FAQ	http://www.icesoft.org/community/license-faq.jsf	Inglés
Recursos externos	https://twitter.com/#!/icefaces http://www.facebook.com/pages/ICEsoft-Technologies/96892345114 http://www.youtube.com/user/ICEsoftTech?feature=watch	Inglés
TOTAL		9

RichFaces: A continuación, se expone los enlaces de soporte disponibles de RichFaces.

DESCRIPCIÓN	ENLACE	IDIOMA
Foros de usuario y desarrolladores	https://community.jboss.org/en/richfaces?view=discussions https://community.jboss.org/en/richfaces/dev?view=discussions	Inglés
Wiki	https://community.jboss.org/wiki/richfaceswikihomepage	Inglés
Bugs	http://www.jboss.org/richfaces/issuetracker.html	
Recursos y artículos de anuncios externos	http://twitter.com/richfaces http://community.jboss.org/wiki/RichFacesJSFPresentationspage http://www.jboss.org/community/wiki/RichFacesInProduction http://alexsmirnov.wordpress.com/	Inglés
Chat	irc://irc.freenode.net/richfaces	Inglés
Blogs	http://in.relation.to/Bloggers/Jay	Inglés
JIRA	https://issues.jboss.org/browse/RF	Inglés
TOTAL		6

Para MyFaces Trinidad:

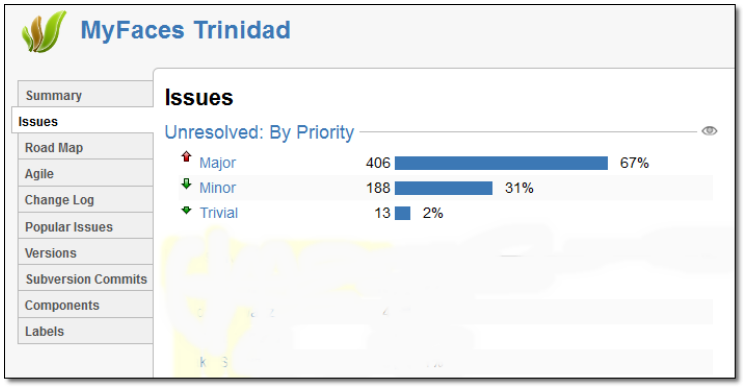
$$\frac{3 \times 50}{9} = 16{,}67\%$$

Para RichFaces:

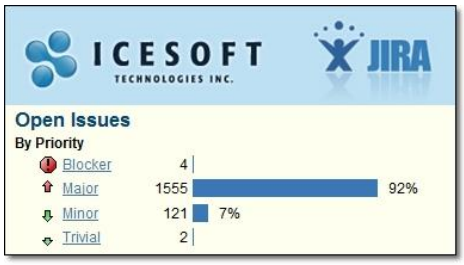
$$\frac{6 \times 50}{9} = 33{,}33\%$$

SECCIÓN 3: Calidad.

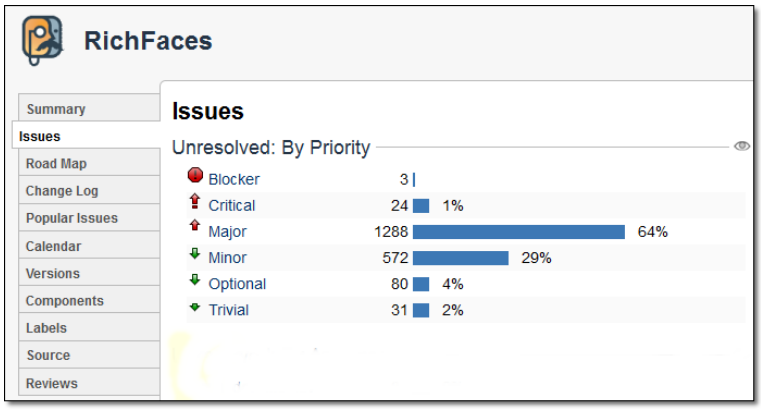
De la página oficial de MyFaces Trinidad en lo que respecta a Issues, se capturó la siguiente figura [71].



De la página oficial de ICESOFT en lo que corresponde a JIRA se capturó la siguiente figura [72].



De la página oficial de RichFaces se obtuvo la siguiente captura en lo que corresponde a Issues [73].



[71] https://issues.apache.org/jira/browse/TRINIDAD#selectedTab=com.atlassian.jira.plugin.system.project%3Aissues-panel
[72] http://jira.icesoft.org/browse/ICE?report=com.atlassian.jira.plugin.system.project:openissues-panel
[73] https://issues.jboss.org/browse/RF#selectedTab=com.atlassian.jira.plugin.system.project%3Aissues-panel

En la siguiente tabla, se presenta un resumen de los problemas sin resolver de los frameworks analizados, datos que se obtuvieron de las imágenes antes mencionadas.

DESCRIPCIÓN	MYFACES TRINIDAD	ICEFACES	RICHFACES
Bloqueados	0	4	3
Crítico	0	0	24
Mayor	406	1555	1289
Menor	188	121	572
Opcional	0	0	80
Insignificante	13	2	31
TOTAL	607	1682	1999

Al sumar todos los problemas sin resolver se obtiene un total de 4.288. Se debe considerar que en este caso el máximo valor a asignar el 100% es al framework que tenga el mínimo valor de problemas sin resolver, que corresponde a MyFaces Trinidad.

Para obtener los valores esperados, se le resta al total de problemas sin resolver los resultados obtenidos de cada framework, de la siguiente manera:

MyFaces Trinidad:	4.288,00 – 607	= 3.681,00
ICEFaces:	4.288,00 – 1.682,00	= 2.606,00
RichFaces:	4.288,00 – 1.999,00	= 2.289,00

El máximo valor que es 3.681 que corresponde a MyFaces Trinidad, cumple en su totalidad con el parámetro calidad, por lo tanto para obtener los valores de ICEFaces y RichFaces se efectúa una regla de tres de la siguiente manera:

Para ICEFaces:

$$\frac{2.606 \times 100}{3.681} = 70{,}8\%$$

Para RichFaces:

$$\frac{2.289 \times 100}{3.681} = 62{,}18\%$$

SECCIÓN 4: Componentes de usuario.- Disponibilidad de Componentes

En la siguiente, se ilustra los 74 componentes que MyFaces Trinidad ofrece según el demo de componentes publicado en su sitio oficial, mismo que se encuentra ubicado en el enlace http://example.irian.at/trinidad-demo/faces/componentDemos.jspx.

breadCrumbs	Chart	chooseColor
chooseDate	Column	commandButton
commandLink	commandNavigationItem	Form
goButton	goLink	Group
Icon	Image	inputColor
inputDate	inputFile	inputHidden
inputListOfValues	inputNumberSpinbox	inputText
Iterator	Legend	media
Message	Messages	navigationPane
navigationTree	outputDocument	outputFormatted
ouputLabel	outputText	page
panelAccordion	panelBorderLayout	panelBox
panelButtonBar	panelCaptionGroup	panelChoice
panelFormLayout	panelGroupLayout	panelHeader
panelHorizontalLayout	panelLabelAndMessage	panelList
panelPage	panelPopup	panelRadio
panelSideBar	panelTabbed	panelTip
Poll	processChiceBar	progressIndicator
resetButton	selectBooleanCheckbox	selectBooleanRadio
selectManyCheckbox	selectManyListbox	selectManyShuttle
selectOneChoice	selectOneListbox	selectOneRadio
selectOrderShuttle	selectRangeChoiceBar	Separator
showDetail	singleStepButtonBar	Spacer
statusIndicator	styleSheet	Table
Train	treeTable	

En la tabla siguiente, se muestra la gama de componentes ACE (Componentes Avanzados de IceFaces).

Accordion	Animation	checkboxButton	confirmationDialog
contextMenu	dataTable	dataExporter	dataTimeEntry
Dialog	Draggable	Droppable	fileEntry
linkButton	maskedEntry	Menú	menuBar
menuButton	menuItem	notificationPanel	panel
Printer	progressBar	pushButton	resizable
sliderEntry	Submenú	Tabset	tooltip

En la tabla siguiente**,** se observa los 76 componentes de IceFaces, información localizada en los componentes de referencia ICE, publicado en IceFaces Wiki, cuyo enlace es http://wiki.IceFaces.org/display/ICE/ICE+Components+Reference.

checkbox	Column	columnGroup	Columns
commandLink	commandSortHeader	dataExporter	dataPaginator
effect	Form	gMap	gMapControl
gMapGeoXml	gMapLatLng	gMapLatLngs	gMapMarker
headerRow	inputHidden	inputRichText	inputSecret
inputTextarea	jsEventLisner	loadBundle	menuBar
menuItems	menuItemSeparator	menuPopup	Message
outputChart	outputConnectionStatus	outputFormat	outputLabel
outputMedia	outputProgress	outputResource	ouputStyle
panelBorder	paneCollapsible	panelConfirmation	panelDriver
panelGroup	panelLayout	panelPopup	panelPositioned
panelStack	panelTab	panelTabSet	panelTooptip
radio	Repeat	rowSelector	selectBooleanCheckbox
selectInputText	selectManyCheckbox	selectManyListbox	selectManyMenu
selectOneMenu	selectOneRadio	setEventPhase	tabChangeListener
treeNode	commandButton	dataTable	gMapDirection
graphicImage	InputText	menuItem	Messages
outputLink	outputText	panelGrid	panelSeries
Portlet	selectInputDate	selectOneListbox	Tree

Según la Guía de Componentes de RichiFaces http://docs.jboss.org/richfaces/latest_4_2_X/Component_Reference/en-US/html/, éste contiene cerca de 75 componentes, tal como se ilustra en la siguiente tabla.

rich:autocomplete	rich:calendar	rich:editor
rich:fileUpload	rich:inplaceInput	rich:inplaceSelect
rich:inputNumberSlider	rich:inputNumberSpinner	rich:select
rich:orderingList	rich:pickList	rich:panel
rich:accordion	rich:collapsiblePanel	rich:popupPanel
rich:tabPanel	rich:tooglePanel	a4j:repeat
rich:dataTable	rich:column	rich:columnGroup
rich:collapsibleSubTable	rich:extendedDataTable	rich:dataGrid
rich:list	rich:dataScroller	rich:tree
rich:treeSelectionChangeListener	rich:treeNode	rich:treeModelAdaptor
rich:treeModelRecursiveAdaptor	rich:dropDrownMenu	rich:contextMenu
rich:menuItem	rich:menuGroup	rich:menuSeparator
rich:panelMenu	rich:toolbar	rich:message
rich:messages	rich:notify	rich:notifyMessage
rich:notifyMessages	rich:notifyStack	rich:progressBar
rich:tooltip	rich:dragSource	rich:dropTraget
rich:dragIndicator	rich:jQuery	rich:clientId
rich:componenet	rich:element	rich:findComponent
rich:isUserInRole	rich:componenetContro	rich:hotKey
rich:hashParam	<a4j:ajax>	<a4j:param>
<a4j:actionListener>	<a4j:commandButton>	<a4j:commandLink>
<a4j:jsfFunction>	<a4j:poll>	<a4j:push>
<a4j:mediaOutput>	<a4j:outputPanel>	<a4j:region>
<rich:validator>	<rich:graphValidator>	<rich:queue>
<rich:attachQueue>	<rich:log>	<rich:status>

En la tabla siguiente se muestra la continuación de los componentes de RichFaces.

En la siguiente tabla, se resume la cantidad de componentes disponibles, datos obtenidos de los resultados anteriores.

FRAMEWORK	NÚMERO DE COMPONENTES
MyFaces Trinidad	74
IceFaces	116
RichFaces	75

Realizando la suma de los componentes ICE y ACE de IceFaces se obtiene aproximadamente 116 componentes, que corresponde al 33,33% en el parámetro de disponibilidad de componentes, por lo tanto para determinar el valor de MyFaces Trinidad y RichFaces se realiza una regla de 3:

Para MyFaces Trinidad:

$$\frac{74 \times 33{,}33}{116} = 21{,}26\%$$

Para RichFaces:

$$\frac{75 \times 33{,}33}{116} = 21{,}55\%$$

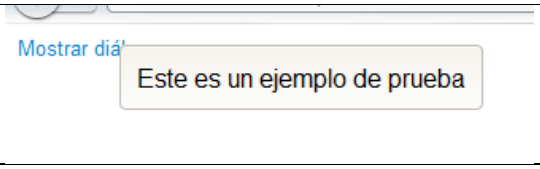
SECCIÓN 5: Componentes de usuario.- Componentes utilizables

Los componentes utilizables se analizaron en base al conjunto de requerimientos previamente establecidos por la Escuela de Nutrición y Dietética. A continuación se ilustran los componentes principales de interfaz de usuario utilizables en cada uno de los frameworks analizados.

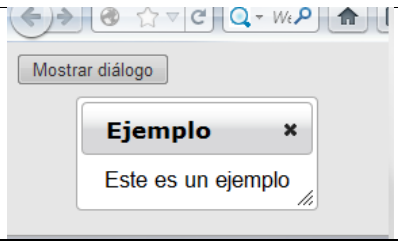
La implementación debe contar con diálogos modales, es decir, ventanas emergentes que floten encima de la ventana principal, de tal manera que no se pueda interactuar con la ventana padre.

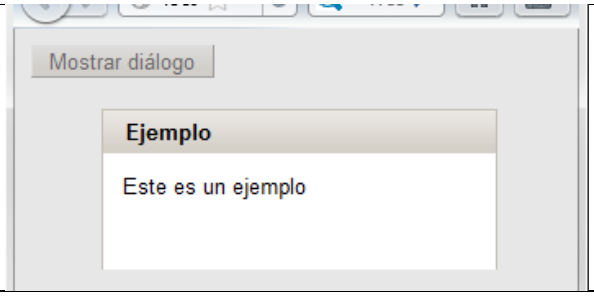
En las tablas siguientes, se ilustran las etiquetas XHTML y las imágenes del diálogo modal que facilitan los frameworks MyFaces Trinidad, IceFaces y RichFaces respectivamente.

Etiquetas XHTML	`<tr:document>`
	`  <tr:form>`
	`  <tr:panelPopup text="Mostrar diálogo">`
	`  <tr:panelGroupLayout layout="vertical">`
	`  <tr:outputText value="Este es un ejemplo de prueba"/>`
	`  </tr:panelGroupLayout>`
	`  </tr:panelPopup>`
	`  </tr:form>`
	`</tr:document>`

Imagen	

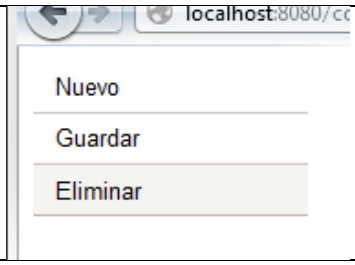
Etiquetas XHTML	`<ice:form id="frmDialog">`
	`  <ice:commandButton id="btnShowDialog"`
	`  onclick="ejemplo.show()"`
	`  value="Mostrar diálogo"/>`
	
	`  <ace:dialog header="Ejemplo"`
	`  widgetVar="ejemplo"`
	`  draggable="true"`
	`  modal="true">`
	`  <ice:outputText value="Este es un ejemplo"/>`
	`  </ace:dialog>`
	`</ice:form>`

Imagen	

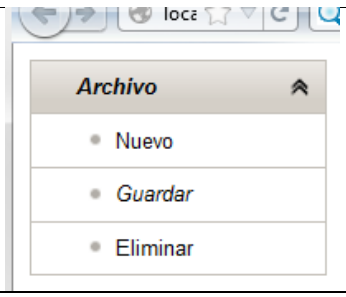
Etiquetas XHTML	<h:form> <h:commandButton value="Mostrar diálogo"> <rich:componentControl target="popup" operation="show" /> </h:commandButton> <rich:popupPanel id="popup" modal="true" width="200" height="80"> <f:facet name="header"> <h:outputText value="Ejemplo" /> </f:facet> <h:outputText value="Este es un ejemplo"/> </rich:popupPanel> </h:form>
Imagen	

La aplicación debe contar con controles de menús funcionales, que permitan realizar distintas acciones en la aplicación.

En las siguientes tablas, se muestra las etiquetas XHTML y las imágenes del componente menú que facilitan MyFaces Trinidad, IceFaces y RichFaces respectivamente.

Etiquetas XHTML	<tr:document> <tr:form> <tr:navigationPane hint="list" inlineStyle="width: 150px;"> <tr:commandNavigationItem text="Nuevo" action="none"/> <tr:commandNavigationItem text="Guardar" action="none"/> <tr:commandNavigationItem text="Eliminar" action="none"/> </tr:navigationPane> </tr:form> </tr:document>
Imagen	

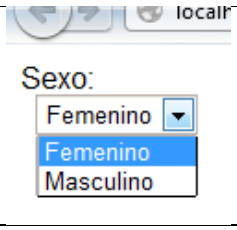
Etiquetas XHTML	<ice:form id="frmDialog"> <ace:menu type="plain" id="menu"> <ace:submenu label="Archivo" id="archivo"> <ace:menuItem id="nuevo" value="Nuevo" icon="ui-icon"> </ace:menuItem> <ace:menuItem id="guardar" value="Guardar" icon="ui-icon ui-icon-disk"> </ace:menuItem> <ace:menuItem id="eliminar" value="Eliminar" icon="ui-icon ui-icon-close"> </ace:menuItem> </ace:submenu> </ace:menu> </ice:form>
Imagen	

Etiquetas XHTML	<h:form> <rich:panelMenu style="width:150px" groupExpandedLeftIcon="triangleUp" groupCollapsedLeftIcon="triangleDown" topGroupExpandedRightIcon="chevronUp" topGroupCollapsedRightIcon="chevronDown" itemLeftIcon="disc"> <rich:panelMenuGroup label="Archivo"> <rich:panelMenuItem label="Nuevo"/> <rich:panelMenuItem label="Guardar"/> <rich:panelMenuItem label="Eliminar"/> </rich:panelMenuGroup> </rich:panelMenu> </h:form>
Imagen	

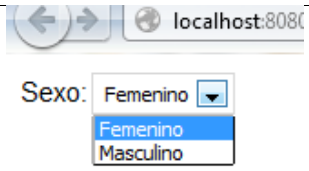
La aplicación debe contar con componentes de combos seleccionables.

En las siguientes tablas, se muestra las etiquetas XHTML y las imágenes del componente selección que ofrece MyFaces Trinidad, IceFaces y RichFaces respectivamente.

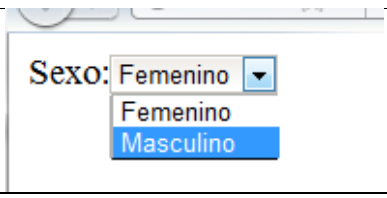
Etiquetas XHTML	`<tr:document>`
	`  <tr:form>`
	`  <tr:outputText value="Sexo:"/>`
	`  <tr:selectOneChoice value="F">`
	`  <f:selectItems value="#{pacienteBean.opcSexo}"/>`
	`  </tr:selectOneChoice>`
	`  </tr:form>`
	`</tr:document>`

Imagen	

Etiquetas XHTML	`<ice:form id="frmDialog">`
	`  <ice:outputText value="Sexo:"/>`
	`  <ice:selectOneMenu value="F">`
	`  <f:selectItems value="#{pruebaBean.opcSexo}"/>`
	`  </ice:selectOneMenu>`
	`</ice:form>`

Imagen	

Etiquetas XHTML	`<h:form id="frmPanel">`
	`  <h:outputText value="Sexo:"/>`
	`  <h:selectOneMenu value="F">`
	`  <f:selectItems value="#{pruebaBean.opcSexo}"/>`
	`  </h:selectOneMenu>`
	`</h:form>`

Imagen	

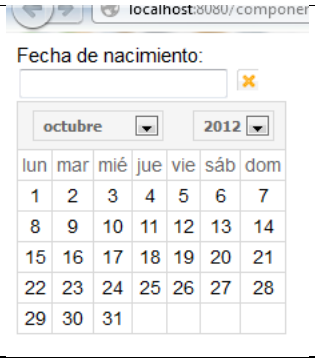
La aplicación debe contar con componentes de fechas seleccionables.

En las tablas siguientes, se ilustra las etiquetas XHTML e imágenes del componente de entrada de fechas que facilita IceFaces, RichFaces y MyFaces Trinidad respectivamente.

Etiquetas XHTML

```
<ice:form id="frmDialog">
    <ice:outputText value="Fecha de nacimiento:"/>
    <ice:selectInputDate id="fechaNacimiento"
                         renderAsPopup="true"
                         renderMonthAsDropdown="true"
                         renderYearAsDropdown="true"
                         highlightClass="weekend :newyear"
                         highlightUnit="DAY_OF_WEEK : DAY_OF_YEAR">
        <f:convertDateTime type="date" pattern="dd-MM-yyyy"/>
    </ice:selectInputDate>
</ice:form>
```

Imagen



Etiquetas XHTML

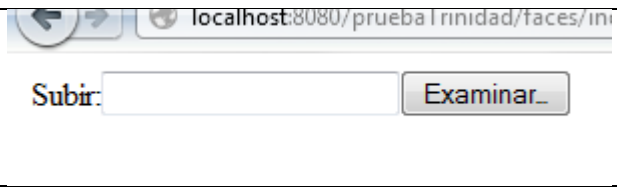
```
<h:form>
    <h:panelGrid>
        <h:outputText value="Fecha de nacimiento:"/>
        <rich:calendar>
        </rich:calendar>
    </h:panelGrid>
</h:form>
```

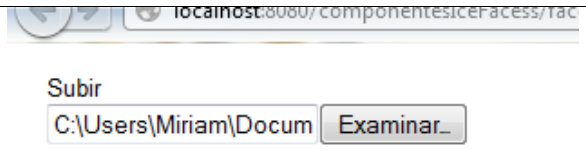
Imagen

Etiquetas XHTML	<tr:document> <tr:form> <tr:panelFormLayout rows="4" labelAlignment="start"> <tr:inputDate id="mdf2" label="Fecha de nacimiento:" simple="false"> <f:convertDateTime dateStyle="short"/> </tr:inputDate> </tr:panelFormLayout> </tr:form> </tr:document>
Imagen	

La implementación debe contar con componentes que permitan subir archivos a la aplicación filtrados por extensión, el usuario final deberá observar el típico cuadro de exploración de archivos para que seleccione el que desea subir.

En las tablas siguientes, se ilustra las etiquetas XHTML e imágenes del componente entrada de archivos que proporcionan los framewroks MyFaces Trinidad, IceFaces y RichFaces respectivamente.

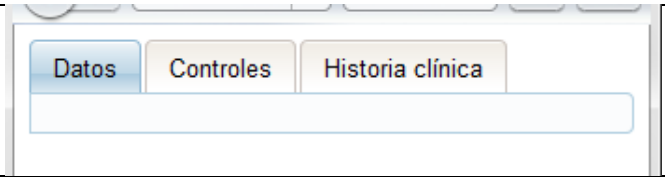
Etiquetas XHTML	<t:form usesUpload="true"> <t:inputFile label="Subir:"/> </t:form>
Imagen	 Subir: Examinar...

Etiquetas XHTML	<ice:form id="frmDialog"> <ace:panel style="width: 250px; height: 40px; border: 0px;"> <ice:outputText value="Subir"/> <ace:fileEntry id="file-entry" relativePath="/files/" maxFileSize="6291456" maxTotalSize="18874368" required="true" requiredMessage="Archivo requerido."/> </ace:panel> </ice:form>
Imagen	

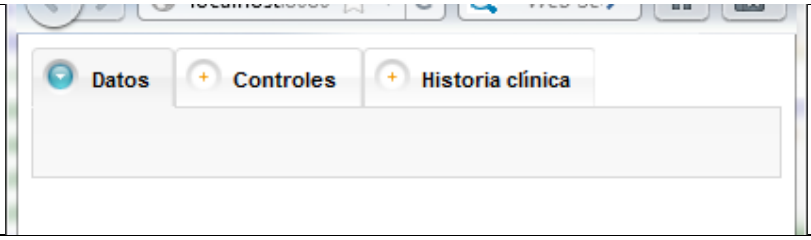
Etiquetas XHTML	<h:form id="frmPanel"> <h:outputText value="Archivo"/> <rich:fileUpload id="upload" acceptedTypes="jpg, png" addLabel="Examinar"/> </h:form>
Imagen	

La aplicación debe contar con controles que proporciones paneles con pestañas, que permitan categorizar en capas agrupadas una variedad de datos.

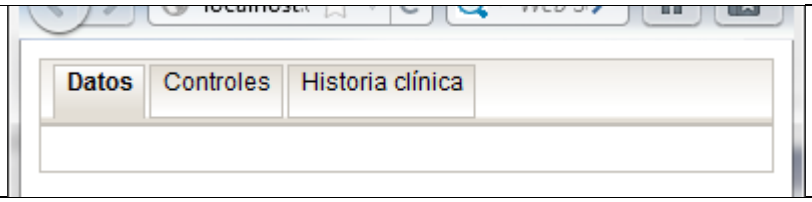
En las tablas siguientes, se muestra las etiquetas XHMTL y las imágenes del componente panel de pestañas que facilitan MyFaces Trinidad, IceFaces y RichFaces respectivamente.

Etiquetas XHTML	<tr:document> <h:form> <tr:panelTabbed position="above"> <tr:showDetailItem text="Datos"/> <tr:showDetailItem text="Controles"/> <tr:showDetailItem text="Historia clínica"/> </tr:panelTabbed> </h:form> </tr:document>
Imagen	

Etiquetas XHTML	`<ice:form id="frmDialog">`
	`  <ice:panelTabSet id="tabs">`
	`  <ice:panelTab label="Datos"></ice:panelTab>`
	`  <ice:panelTab label="Controles"></ice:panelTab>`
	`  <ice:panelTab label="Historia clínica"></ice:panelTab>`
	`  </ice:panelTabSet>`
	`</ice:form>`

Imagen	

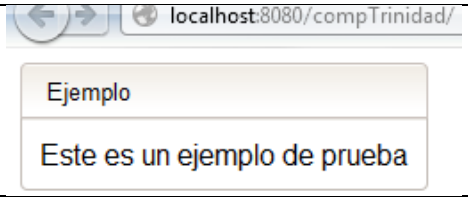
Etiquetas XHTML	`<h:form>`
	`  <rich:tabPanel>`
	`  <rich:tab header="Datos"></rich:tab>`
	`  <rich:tab header="Controles"></rich:tab>`
	`  <rich:tab header="Historia clínica"></rich:tab>`
	`  </rich:tabPanel>`
	`</h:form>`

Imagen	

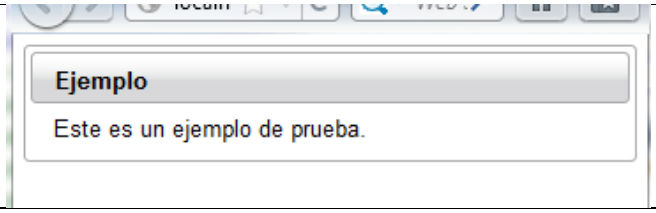
La implementación debe contar con paneles para la agrupación de controles de interfaz de usuario.

En las tablas siguientes, se visualiza las etiquetas XHTML e imágenes que tienen los frameworks MyFaces Trinidad, IceFaces y RichFaces respectivamente.

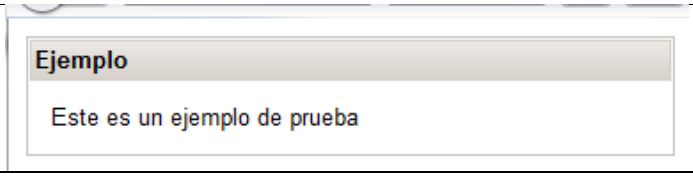
Etiquetas XHTML	`<tr:document>`
	`  <tr:form>`
	`  <tr:panelBox text="Ejemplo">`
	`  <tr:outputText value="Este es un ejemplo de prueba"/>`
	`  </tr:panelBox>`
	`  </tr:form>`
	`</tr:document>`

Imagen	

Etiquetas XHTML	`<ice:form id="frmPanel">`
	`  <ace:panel>`
	`  <f:facet name="header">`
	`  <ice:outputText value="Ejemplo"/>`
	`  </f:facet>`
	`  <ice:outputText value="Este es un ejemplo de prueba."/>`
	`  </ace:panel>`
	`</ice:form>`

Imagen	

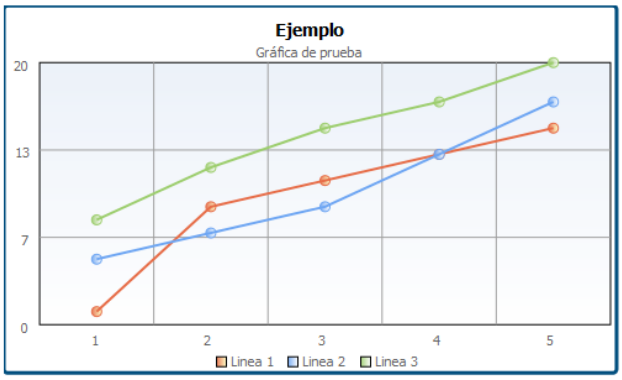
Etiquetas XHTML	`<h:form id="frmPanel">`
	`  <rich:panel>`
	`  <f:facet name="header">`
	`  <h:outputText value="Ejemplo"/>`
	`  </f:facet>`
	`  <h:outputText value="Este es un ejemplo de prueba"/>`
	`  </rich:panel>`
	`</h:form>`

Imagen	

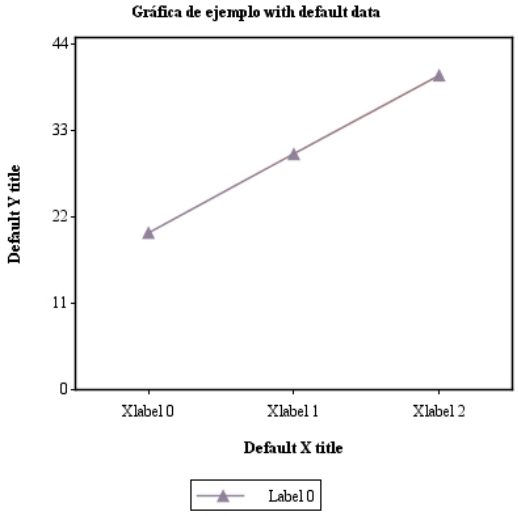
La implementación debe contar con gráficas lineales, para graficar las curvas nutricionales de la OMS.

En las tablas siguientes, se muestra las etiquetas XHTML e imágenes del componente chart que proporcionan MyFaces Trinidad e IceFaces respectivamente.

Etiquetas XHTML	`<tr:document>`
	`  <tr:form>`
	`  <tr:chart YMajorGridLineCount="2"`
	`  value="#{chartBean.value}"`
	`  type="line"`
	`  inlineStyle="width:500px; height:300px;"`
	`  perspective="false"/>`
	`  </tr:form>`
	`</tr:document>`

Imagen	

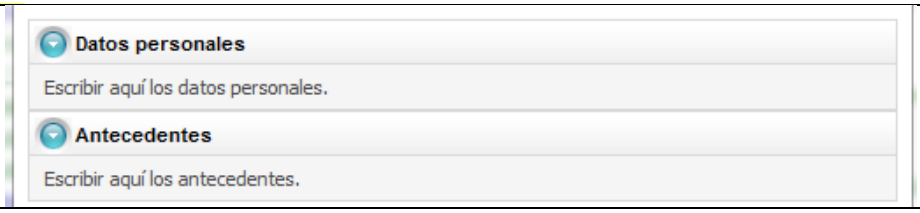
Etiquetas XHTML	`<ice:form id="frmPanel">`
	`  <ice:outputChart type="line"`
	`  chartTitle="Gráfica de ejemplo">`
	`  </ice:outputChart>`
	`</ice:form>`

Imagen	

La implementación debe contar con paneles divisibles, para organizar de mejor manera los datos de entrada.

En las siguientes tablas, se muestra las etiquetas XHTML y las imágenes del componente panel divisible que proporcionan los frameworks MyFaces Trinidad, IceFaces y RichFaces respectivamente.

Etiquetas XHTML	<tr:form> <tr:panelCaptionGroup captionText="Datos personales"> <tr:panelFormLayout> <tr:outputText value="Escriba aquí los datos personales"/> </tr:panelFormLayout> </tr:panelCaptionGroup> <tr:panelCaptionGroup captionText="Antecedentes"> <tr:panelFormLayout> <tr:outputText value="Escriba aquí los antecedentes personales"/> </tr:panelFormLayout> </tr:panelCaptionGroup> </tr:form>
Imagen	

Etiquetas XHTML	<ice:form id="frmPanel"> <ice:panelCollapsible expanded="true"> <f:facet name="header"> <ice:panelGroup> <ice:outputText value="Datos personales"/> </ice:panelGroup> </f:facet> <h:outputText value="Escribir aquí los datos personales."/> </ice:panelCollapsible> <ice:panelCollapsible expanded="true"> <f:facet name="header"> <ice:panelGroup> <ice:outputText value="Antecedentes"/> </ice:panelGroup> </f:facet> <h:outputText value="Escribir aquí los antecedentes."/> </ice:panelCollapsible> </ice:form>
Imagen	

Etiquetas XHTML	<h:form> <rich:collapsiblePanel header="Datos personales" expanded="true"> <h:outputText value="Escriba aquí los datos personales."/> </rich:collapsiblePanel> <rich:collapsiblePanel header="Antecedentes" expanded="true"> <h:outputText value="Escriba aquí los antecedentes."/> </rich:collapsiblePanel> </h:form>

Imagen			

La implementación debe contar con listas paginadas, de tal manera que permita mostrar contenidos en filas y columnas de forma paginada.

En las siguientes tablas, se muestra las etiquetas XHTML e imágenes del componente lista con paginación que facilitan los framewors MyFaces Trinidad, IceFaces y RichFaces respectivamente.

Etiquetas XHTML

```
<tr:document>
    <tr:form>
        <tr:table var="paciente" value="#{pacienteBean.listPaciente}" rows="5">
            <tr:column headerText="Cédula">
                <tr:outputText value="#{paciente.cedula}"/>
            </tr:column>
            <tr:column headerText="Nombres">
                <tr:outputText value="#{paciente.nombres}"/>
            </tr:column>
            <tr:column headerText="Apellidos">
                <tr:outputText value="#{paciente.apellidos}"/>
            </tr:column>
            <tr:column headerText="Fecha de nacimiento">
                <tr:outputText value="#{paciente.fechaNacimiento}"/>
            </tr:column>
            <tr:column headerText="Sexo">
                <tr:outputText value="#{paciente.sexo}"/>
            </tr:column>
            <tr:column headerText="Antecedentes">
                <tr:outputText value="#{paciente.antecedentes}"/>
            </tr:column>
        </tr:table>
    </tr:form>
</tr:document>
```

Imagen

Anterior | 1-5 de 10 | Siguientes 5

Cédula	Nombres	Apellidos	Fecha de nacimiento	Sexo	Antecedentes
1111111110	MANUEL ARMANDO	ARMIJOS	01/01/2007	M	NINGUNO
1111111111	ERIKA FERNANDA	ABAD	03/09/1897	F	NINGUNO
1111111112	NALDO FRANCISCO	CAJAS	05/02/1876	M	NINGUNO
1111111113	JUAN DANIEL	BELTRÁN	07/09/1987	M	NINGUNO
1111111114	GIOCONDA DEL CARMÉN	PILCO	03/03/1985	F	NINGUNO

Etiquetas XHTML

```
<ice:form>
    <ace:dataTable value="#{pacienteBean.listPaciente}"
                   var="paciente" paginator="true" rows="5">
        <f:facet name="header">
            <ice:outputText value="Pacientes"/>
        </f:facet>
        <ace:column headerText="Cédula">
            <ice:outputText value="#{paciente.cedula}"/>
        </ace:column>
        <ace:column headerText="Nombres">
            <ice:outputText value="#{paciente.nombres}"/>
        </ace:column>
        <ace:column headerText="Apellidos">
            <ice:outputText value="#{paciente.apellidos}"/>
        </ace:column>
        <ace:column headerText="Fecha nacimiento">
            <ice:outputText value="#{paciente.fechaNacimiento}"/>
        </ace:column>
        <ace:column headerText="Sexo">
            <ice:outputText value="#{paciente.sexo}"/>
        </ace:column>
        <ace:column headerText="Antecedentes">
            <ice:outputText value="#{paciente.antecedentes}"/>
        </ace:column>
    </ace:dataTable>
</ice:form>
```

Imagen

Pacientes

1 2

Cédula	Nombres	Apellidos	Fecha nacimiento	Sexo	Antecedentes
1111111110	MANUEL ARMANDO	ARMIJOS	2007-01-01	M	NINGUNO
1111111111	ERIKA FERNANDA	ABAD	1897-09-03	F	NINGUNO
1111111112	NALDO FRANCISCO	CAJAS	1876-02-05	M	NINGUNO
1111111113	JUAN DANIEL	BELTRÁN	1987-09-07	M	NINGUNO
1111111114	GIOCONDA DEL CARMEN	PILCO	1985-03-03	F	NINGUNO

1 2

Etiquetas XHTML

```
<h:form>
    <rich:dataScroller for="table" maxPages="5"></rich:dataScroller>
    <rich:dataTable id="table" value="#{pacienteBean.listPaciente}"
                    var="paciente" rows="5">
            <rich:column>
                <f:facet name="header"><h:outputText value="Cédula"/></f:facet>
                <h:outputText value="#{paciente.cedula}"/>
            </rich:column>
            <rich:column>
                <f:facet name="header"><h:outputText value="Nombres"/></f:facet>
                <h:outputText value="#{paciente.nombres}"/>
            </rich:column>
            <rich:column>
                <f:facet name="header"><h:outputText value="Apellidos"/></f:facet>
                <h:outputText value="#{paciente.apellidos}"/>
            </rich:column>
            <rich:column>
                <f:facet name="header"><h:outputText value="Fecha de Nacimiento"/></f:facet>
                <h:outputText value="#{paciente.fechaNacimiento}"/>
            </rich:column>
            <rich:column>
                <f:facet name="header"><h:outputText value="Sexo"/></f:facet>
                <h:outputText value="#{paciente.sexo}"/>
            </rich:column>
            <rich:column>
                <f:facet name="header"><h:outputText value="Antecedentes"/></f:facet>
                <h:outputText value="#{paciente.antecedentes}"/>
            </rich:column>
        </rich:dataTable>
</h:form>
```

Imagen

« « « 1 2 » » » » » »

Cédula	Nombres	Apellidos	Fecha de Nacimiento	Sexo	Antecedentes
1111111110	MANUEL ARMANDO	ARMIJOS	2007-01-01	M	NINGUNO
1111111111	ERIKA FERNANDA	ABAD	1897-09-03	F	NINGUNO
1111111112	NALDO FRANCISCO	CAJAS	1876-02-05	M	NINGUNO
1111111113	JUAN DANIEL	BELTRÁN	1987-09-07	M	NINGUNO
1111111114	GIOCONDA DEL CARMÉN	PILCO	1985-03-03	F	NINGUNO

En la siguiente tabla, se muestra un resumen comparativo en base al análisis realizado anteriormente.

COMPONENTES	MYFACES TRINIDAD	ICEFACES	RICHFACES
Diálogos modales	✓	✓	✓
Menú de opciones	✓	✓	✓
Combos	✓	✓	✓
Calendario	✓	✓	✓
Subir archivos	✓	✓	✓
Paneles	✓	✓	✓
Paneles de pestaña	✓	✓	✓
Gráficas 2D	✓	✓	×
Paneles divisibles	✓	✓	✓
Listas paginadas	✓	✓	✓

En la tabla anterior se observa que MyFaces Trinidad e IceFaces cumplen en su totalidad con los componentes requeridos para el desarrollo del Sistema Nutricional de la ESPOCH.

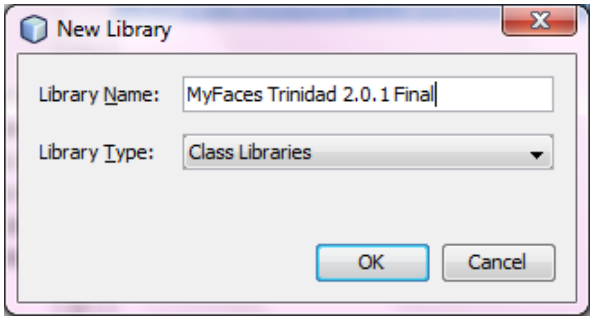
SECCIÓN 6: Facilidad para el desarrollo.- Facilidad de instalación.

INSTALACIÓN DE MYFACES TRINIDAD SOBRE NETBEANS 7.1.1

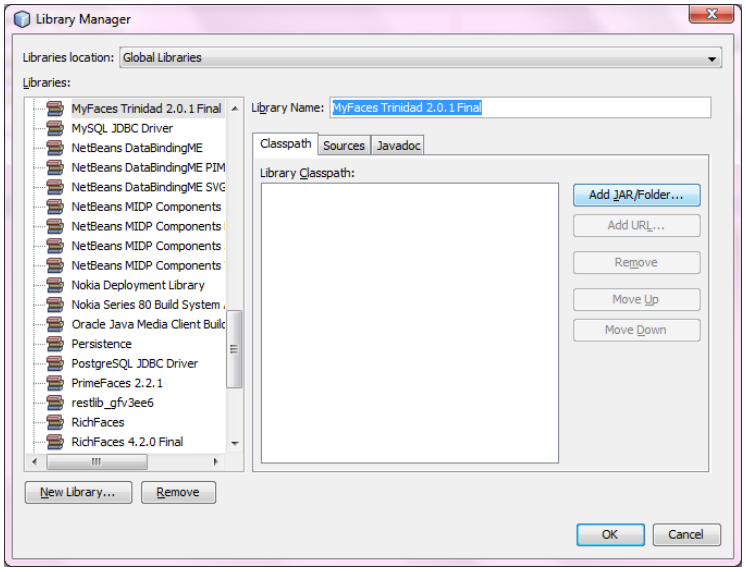
- Descargar la última versión de MyFaces Trinidad (**Apache MyFaces Trinidad 2.0.1 (zip)**) en el enlace http://myfaces.apache.org/trinidad/download.html.
- Para utilizar los componentes de MyFaces Trinidad, es necesario descargar varias librerías de Java, MyFaces utiliza librerías del proyecto Apache Commons. Las librerías a descargar se mencionan a continuación, mismas que se descargaron del enlace http://archive.apache.org/dist/commons/:

  commons-beanutils-1.8.3.jar
  commons-codec-1.6.jar
  commons-collections-3.2.jar
  commons-digester-2.0.jar
  commons-el.jar
  commons-fileupload-1.2.2.jar
  commons-io-2.1.jar
  commons-lang3-3.1.jar
  commons-validator-1.4.0.jar

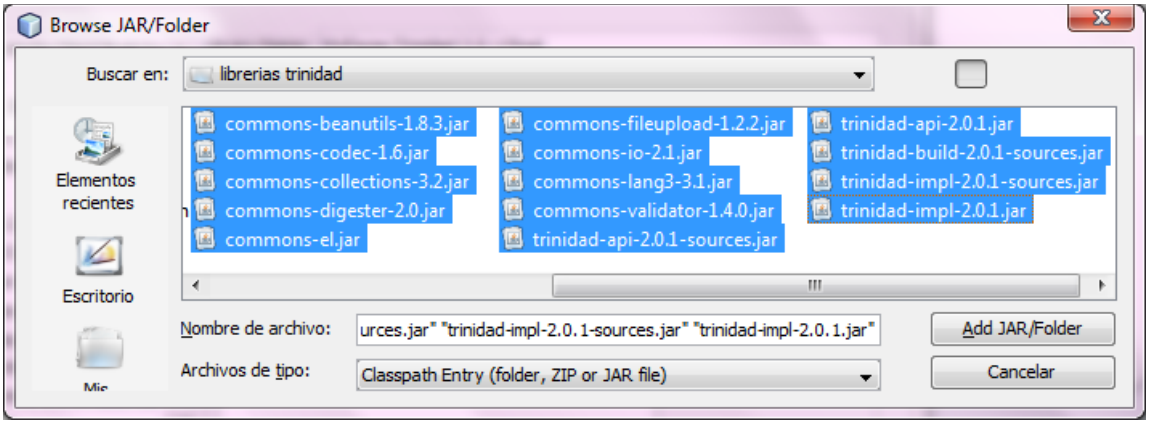
- Posteriormente a esto descomprimir los archivos descargado en un directorio, con la finalidad de extraer todas las librerías con extensión .jar.
- En la barra de menú principal de NetBeans seleccionar **Tools → Libraries**.
- Dar un nombre a la nueva librería y posteriormente en **OK**.



- Dar clic en **Add JAR/Folder…** para añadir las nuevas librerías.



- Navegar por el directorio en donde se descomprimió las librerías, seleccionarlas y añadirlas.

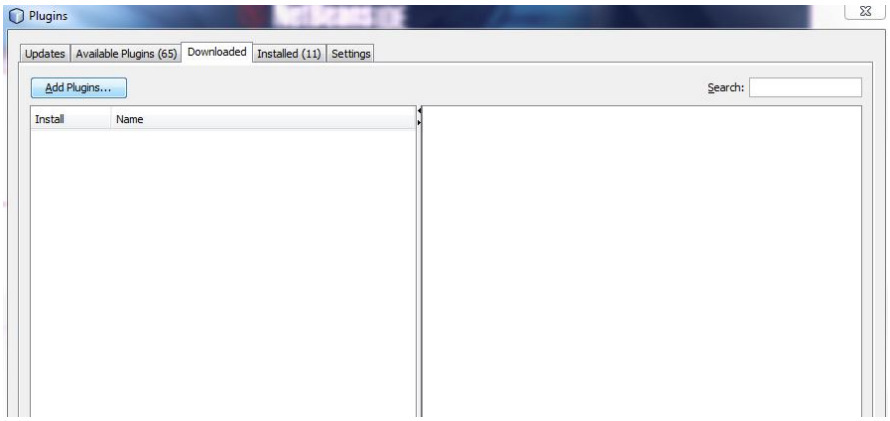


- Una vez seleccionadas todas las librerías presionar **OK** en el administrador de librerías. Finalmente se tendrá todas las librerías necesarias para poder trabajar con los componentes de MyFaces Trinidad.

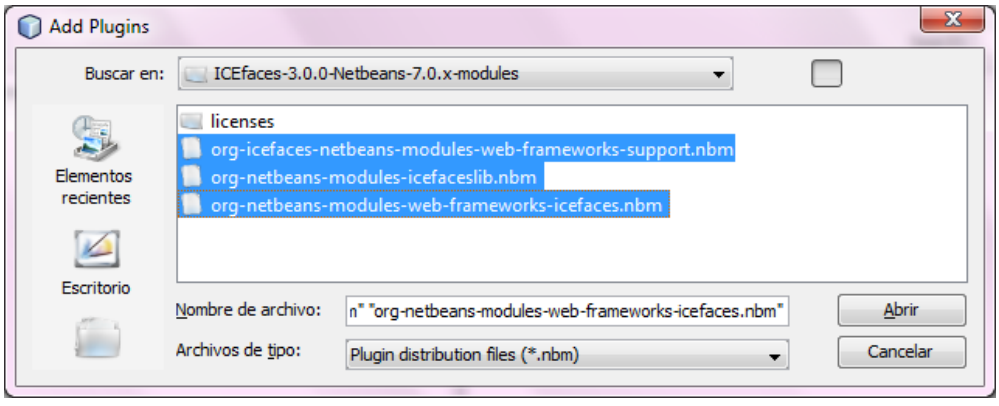
INSTALACIÓN DE ICEFACES SOBRE NETBEANS 7.1.1

- Descargar la última versión de IceFaces (**IceFaces-3.0.0-Netbeans-7.1-modules.zip**) para NetBeans 7.1.x en el enlace http://www.icesoft.org/downloads/IceFaces-downloads.jsf. Posteriormente a esto descomprimir el archivo descargado con la finalidad de extraer las 3 bibliotecas con extensión .nmbs, mismas que se mencionan a continuación:
  org-IceFaces-netbeans-modules-IceFaceslib.nbm
  org-IceFaces-netbeans-modules-web-frameworks.nbm
  org-IceFaces-netbeans-modules-web-frameworks-support.nbm

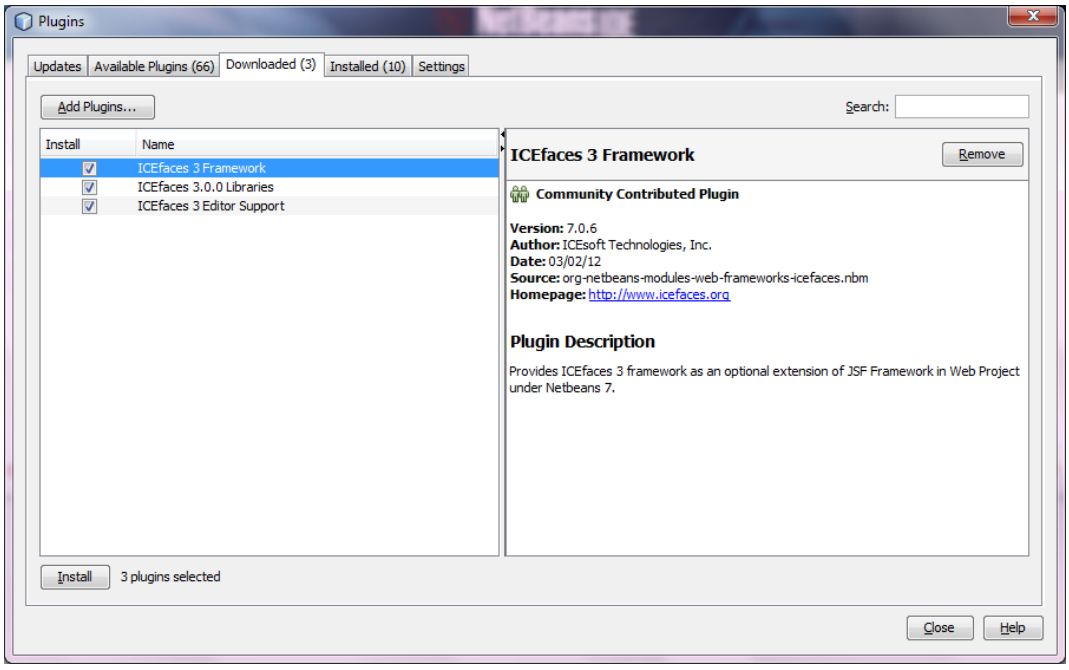
- Desde la barra del menú principal de NetBeans, seleccionar **Tools → Plugins.**
- Seleccionar la pestaña **Download** en **Plugins**, y presionar el botón **Add Plugins…**



- Navegar al directorio en donde se encuentran los tres módulos de extensión .nbms, seleccionarlos y por último hacer clic en **Abrir**.



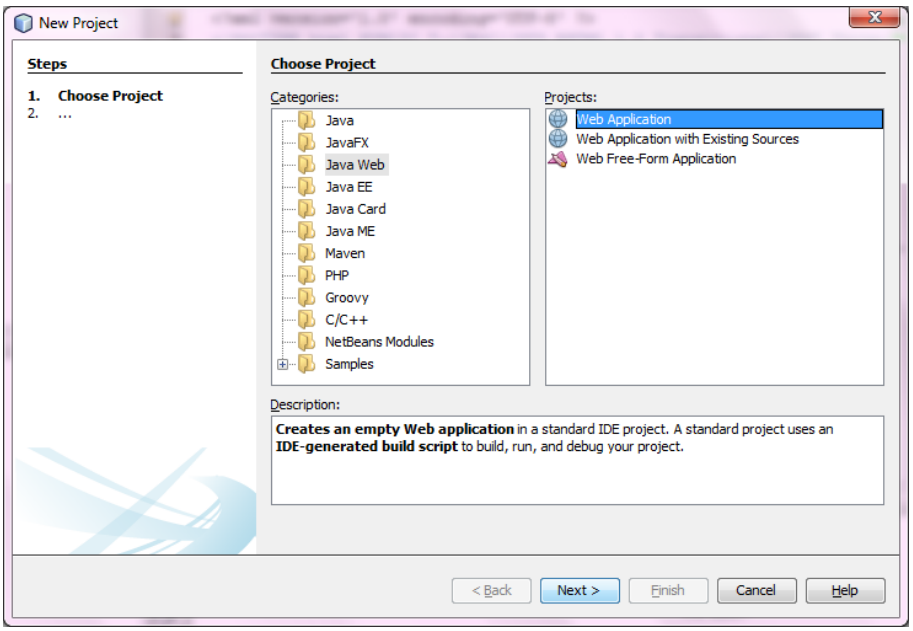
- En el asistente de instalación de Plugins, seleccionar las librerías de IceFaces 3.0.0, el editor de soporte de IceFaces e IceFaces 3. Posteriormente a esto hacer clic en **Install**.



- En la siguiente ventana de diálogo aceptar la licencia, seguir las instrucciones y finalmente presionar el botón **Close**.
- Reiniciar NetBeans.

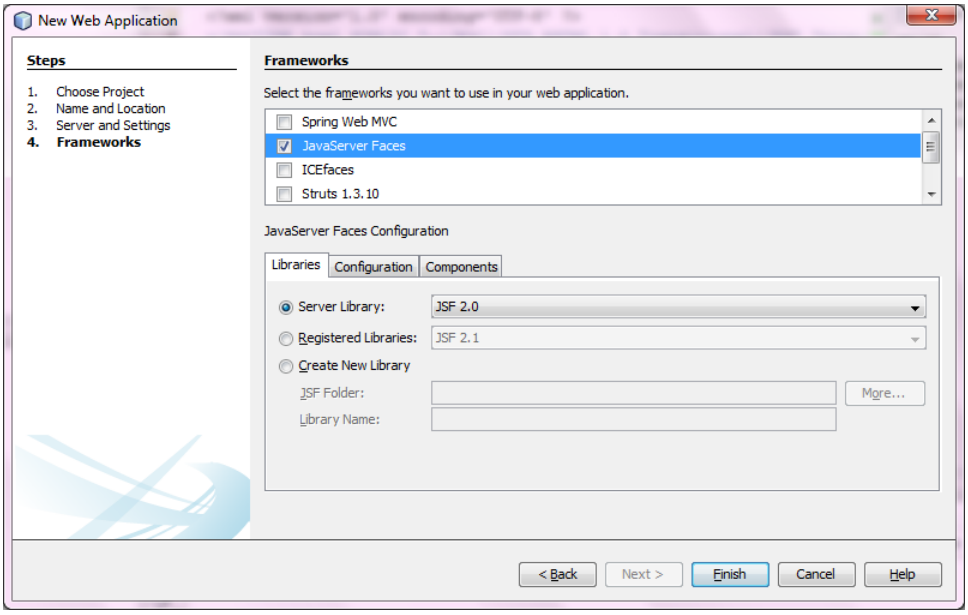
INSTALACIÓN DE RICHFACES SOBRE NETBEANS

- Descargar la última versión de RichFaces (**RichFaces 4.2.0.Final distribution (zip)**) en el enlace http://www.jboss.org/richfaces/download/stable.
- Descargar Google Guava del enlace http://code.google.com/p/guava-libraries/.
- Descargar CSS Parser del enlace http://ufpr.dl.sourceforge.net/project/cssparser/cssparser/0.9.6/cssparser-0.9.6.jar.
- Descargar el API Simple para CSS del enlace http://jaist.dl.sourceforge.net/project/jmtask/libraries/sac-1.3.jar.
- Desde la barra del menú principal de NetBeans, seleccionar **File → New Project.. → Java Web → Web Application** y por último hacer clic en **Next**.

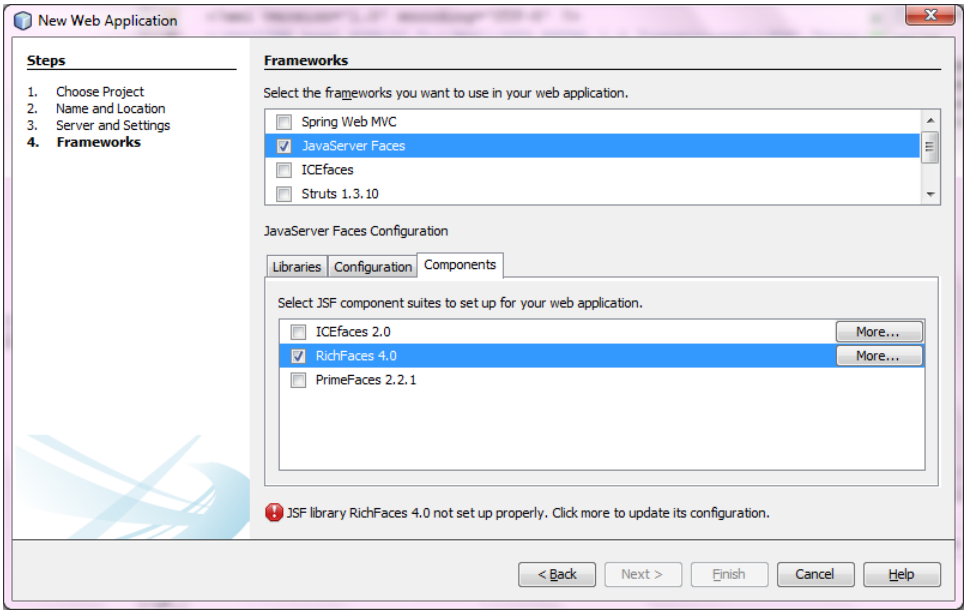


- En la próxima ventana de diálogo se da un nombre al proyecto y posteriormente se deja las opciones por defecto.

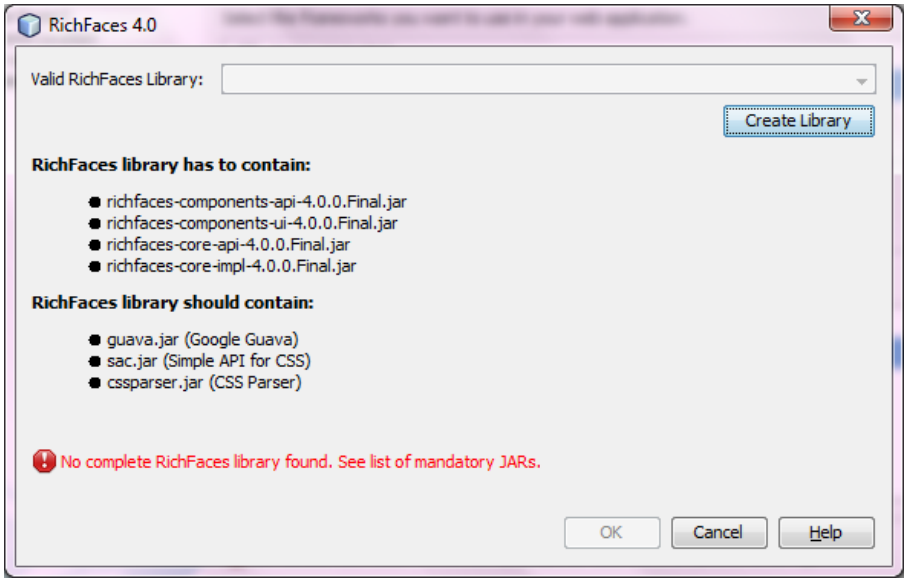
- En la vista de Frameworks seleccionar **JavaServer Faces**.



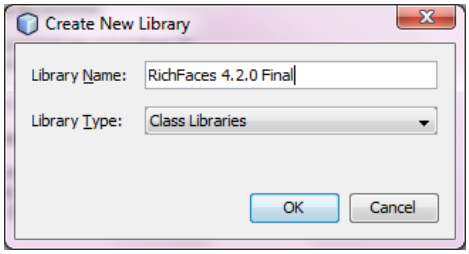
- Seleccionar la pestaña **Components**, escoger RichFaces 4.0 y dar clic en **More…**



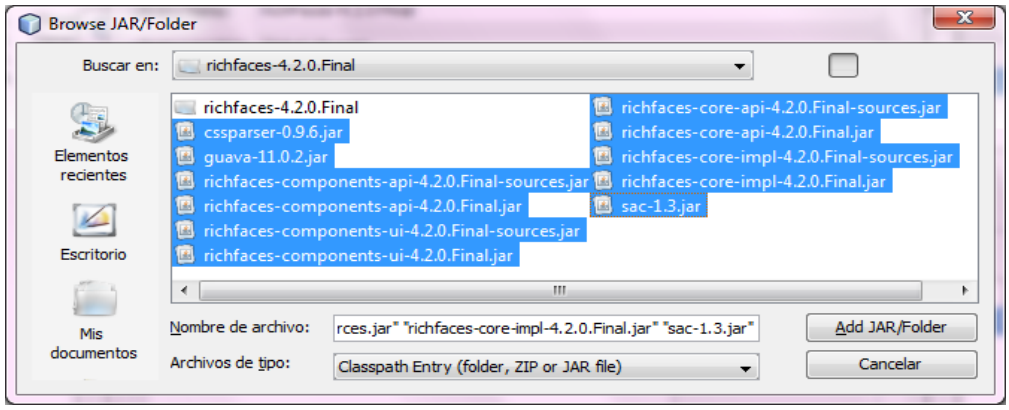
- Crear una nueva librería dando clic en **Create Library**.



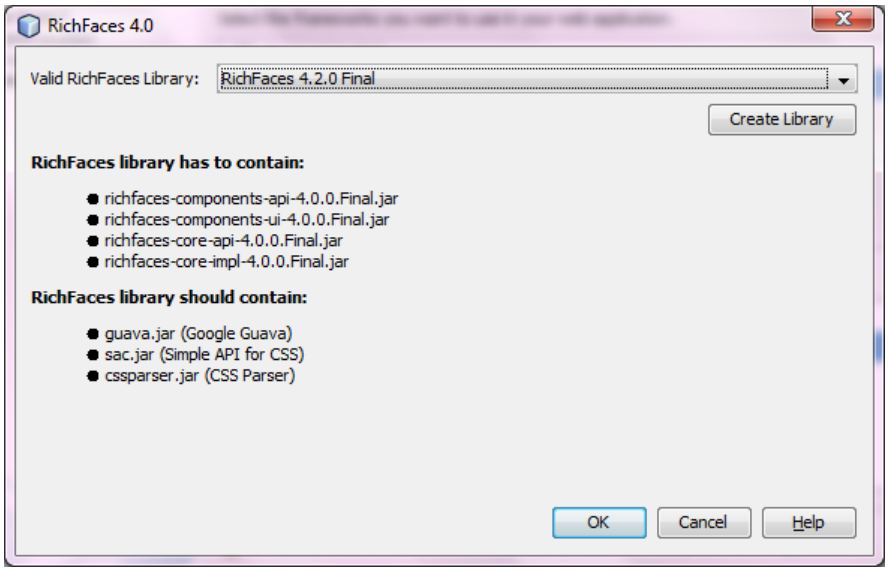
- Dar un nombre a la nueva librería.



- Seleccionar las bibliotecas necesarias para la creación de la nueva librería.



- Luego de dar clic en **OK**, se obtendrá el siguiente cuadro de diálogo, mismo que nos indica que la creación de la librería RichFaces se ha realizado con éxito.



- Después de presionar **OK**, se finaliza con la creación del proyecto y adición de las librerías necesarias para trabajar con RichFaces.

Resultado cuantitativo de la facilidad de instalación

De la encuesta realizada (ver Anexo C), los valores obtenidos para la pregunta uno se resume en la siguiente tabla.

MYFACES TRINIDAD	ICEFACES	RICHFACES
9	15	10

El valor máximo obtenido es 15 que corresponde a IceFaces, a este valor se le asigna el porcentaje del 20%, por tanto para obtener el valor de MyFaces Trinidad y RichFaces a continuación se realiza reglas de tres.

Para MyFaces Trinidad:

$$\frac{9 \times 20}{15} = 12\%$$

Para RichFaces:

$$\frac{10 \times 20}{15} = 13{,}33\%$$

SECCIÓN 7: Facilidad para el desarrollo.- Facilidad para iniciar

Configuración del archivo web.xml de MyFaces Trinidad

Al crear un proyecto JSF, el archivo **web.xml** inicial es el siguiente.

```
<context-param>
    <param-name>javax.faces.PROJECT_STAGE</param-name>
    <param-value>Development</param-value>
</context-param>
<servlet>
    <servlet-name>Faces Servlet</servlet-name>
    <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
<session-config>
    <session-timeout>
        30
    </session-timeout>
</session-config>
<welcome-file-list>
    <welcome-file>faces/index.xhtml</welcome-file>
</welcome-file-list>
```

La configuración necesaria del archivo **web.xml** para empezar a trabajar con MyFaces Trinidad, es de la siguiente manera.

```
  <context-param>
      <param-name>org.apache.myfaces.trinidad.CHECK_FILE_MODIFICATION</param-name>
      <param-value>true</param-value>
 </context-param>

 <context-param>
  <param-name>org.apache.myfaces.trinidad.CHANGE_PERSISTENCE</param-name>
  <param-value>session</param-value>
</context-param>

<context-param>
  <param-name>org.apache.myfaces.trinidad.ALTERNATE_VIEW_HANDLER</param-name>
  <param-value>com.sun.facelets.FaceletViewHandler</param-value>
</context-param>

<context-param>
  <param-name>java.faces.DEFAULT_SUFFIX</param-name>
  <param-value>.xhtml</param-value>
</context-param>

<filter>
  <filter-name>trinidad</filter-name>
  <filter-class>org.apache.myfaces.trinidad.webapp.TrinidadFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name>trinidad</filter-name>
  <servlet-name>faces</servlet-name>
</filter-mapping>
   <context-param>
       <param-name>javax.faces.PROJECT_STAGE</param-name>
       <param-value>Production</param-value>
   </context-param>

 <session-config>
       <session-timeout>
           30
       </session-timeout>
   </session-config>
   <welcome-file-list>
       <welcome-file>faces/index.xhtml</welcome-file>
   </welcome-file-list>
```

Configuración del archivo web.xml de IceFaces

Los módulos de instalación de IceFaces para NetBeans resuelven todas las dependencias necesarias. El archivo **web.xml** de IceFaces creado por las dependencias propias de los módulos se muestra a continuación.

```
<context-param>
    <param-name>javax.faces.PROJECT_STAGE</param-name>
    <param-value>Development</param-value>
</context-param>
<context-param>
    <param-name>javax.faces.STATE_SAVING_METHOD</param-name>
    <param-value>server</param-value>
</context-param>
<context-param>
    <param-name>javax.faces.FACELETS_SKIP_COMMENTS</param-name>
    <param-value>true</param-value>
</context-param>
<context-param>
    <param-name>org.icefaces.mandatoryResourceConfiguration</param-name>
    <param-value/>
</context-param>
<context-param>
    <param-name>org.icefaces.ace.theme</param-name>
    <param-value>sam</param-value>
</context-param>
<context-param>
    <param-name>javax.faces.VALIDATE_EMPTY_FIELDS</param-name>
    <param-value>false</param-value>
</context-param>
<context-param>
    <param-name>com.icesoft.faces.gmapKey</param-name>
    <param-value>ABQIAAAADlu0ZiSTam64EKaCQr9eTRTOTuQNzJNXRlYRLknj4cQ89tFfpxTEqxQnVWL4k55OPICgF5_SOZE06A</param-value>
</context-param>
<servlet>
    <servlet-name>Faces Servlet</servlet-name>
    <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet>
    <servlet-name>Resource Servlet</servlet-name>
    <servlet-class>com.icesoft.faces.webapp.CompatResourceServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>*.jsf</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>/icefaces/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>Resource Servlet</servlet-name>
    <url-pattern>/xmlhttp/*</url-pattern>
</servlet-mapping>
<session-config>
    <session-timeout>
        30
    </session-timeout>
</session-config>
<welcome-file-list>
    <welcome-file>faces/index.xhtml</welcome-file>
</welcome-file-list>
```

Configuración del archivo web.xml de RichFaces

La configuración inicial del archivo **web.xml** al crear una aplicación JSF se ilustra a continuación.

```
<context-param>
    <param-name>javax.faces.PROJECT_STAGE</param-name>
    <param-value>Development</param-value>
</context-param>
<servlet>
    <servlet-name>Faces Servlet</servlet-name>
    <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
<session-config>
    <session-timeout>
        30
    </session-timeout>
</session-config>
<welcome-file-list>
    <welcome-file>faces/index.xhtml</welcome-file>
</welcome-file-list>
```

El archivo **web.xml** configurado correctamente para trabajar con RichFaces es de la siguiente manera.

```
<context-param>
    <param-name>javax.faces.PROJECT_STAGE</param-name>
    <param-value>Development</param-value>
</context-param>
<servlet>
    <servlet-name>Faces Servlet</servlet-name>
    <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
<session-config>
    <session-timeout>
        30
    </session-timeout>
</session-config>
<welcome-file-list>
    <welcome-file>faces/index.xhtml</welcome-file>
</welcome-file-list>

<context-param>
    <param-name>org.richfaces.SKIN</param-name>
    <param-value>blueSky</param-value>
</context-param>
<!-- Making the RichFaces skin spread to standard HTML controls -->
<context-param>
    <param-name>org.richfaces.CONTROL_SKINNING</param-name>
    <param-value>enable</param-value>
</context-param>
<!-- Defining and mapping the RichFaces filter -->
<filter>
    <display-name>RichFaces Filter</display-name>
    <filter-name>richfaces</filter-name>
    <filter-class>org.ajax4jsf.Filter</filter-class>
</filter>

<filter-mapping>
    <filter-name>richfaces</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
    <dispatcher>REQUEST</dispatcher>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>INCLUDE</dispatcher>
</filter-mapping>
```

Resultado cuantitativo de la facilidad para iniciar

De la encuesta realizada (ver Anexo C), el resumen de la tabulación obtenida de la pregunta dos se presente en la siguiente tabla**.**

MYFACES TRINIDAD	ICEFACES	RICHFACES
10	15	10

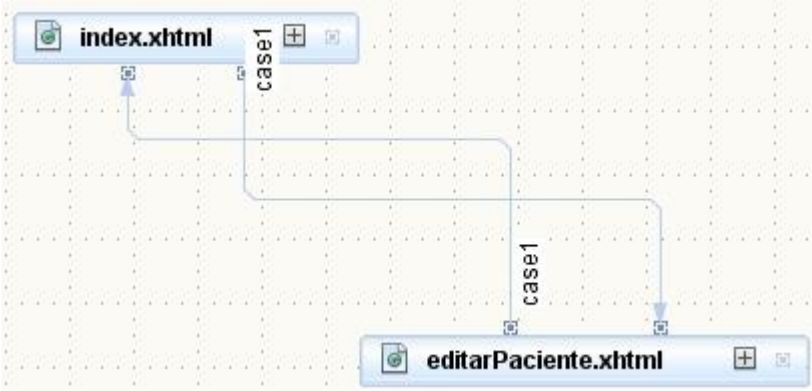
El máximo valor que es 15 corresponde a IceFaces asignándole el porcentaje total del 20%, por tal motivo para obtener los valores de MyFaces Trinidad y RichFaces se realiza una regla de tres.

Para MyFaces Trinidad y RichFaces:

$$\frac{10 \times 20}{15} = 13{,}33\%$$

SECCIÓN 8: Facilidad para el desarrollo.- Facilidad de uso

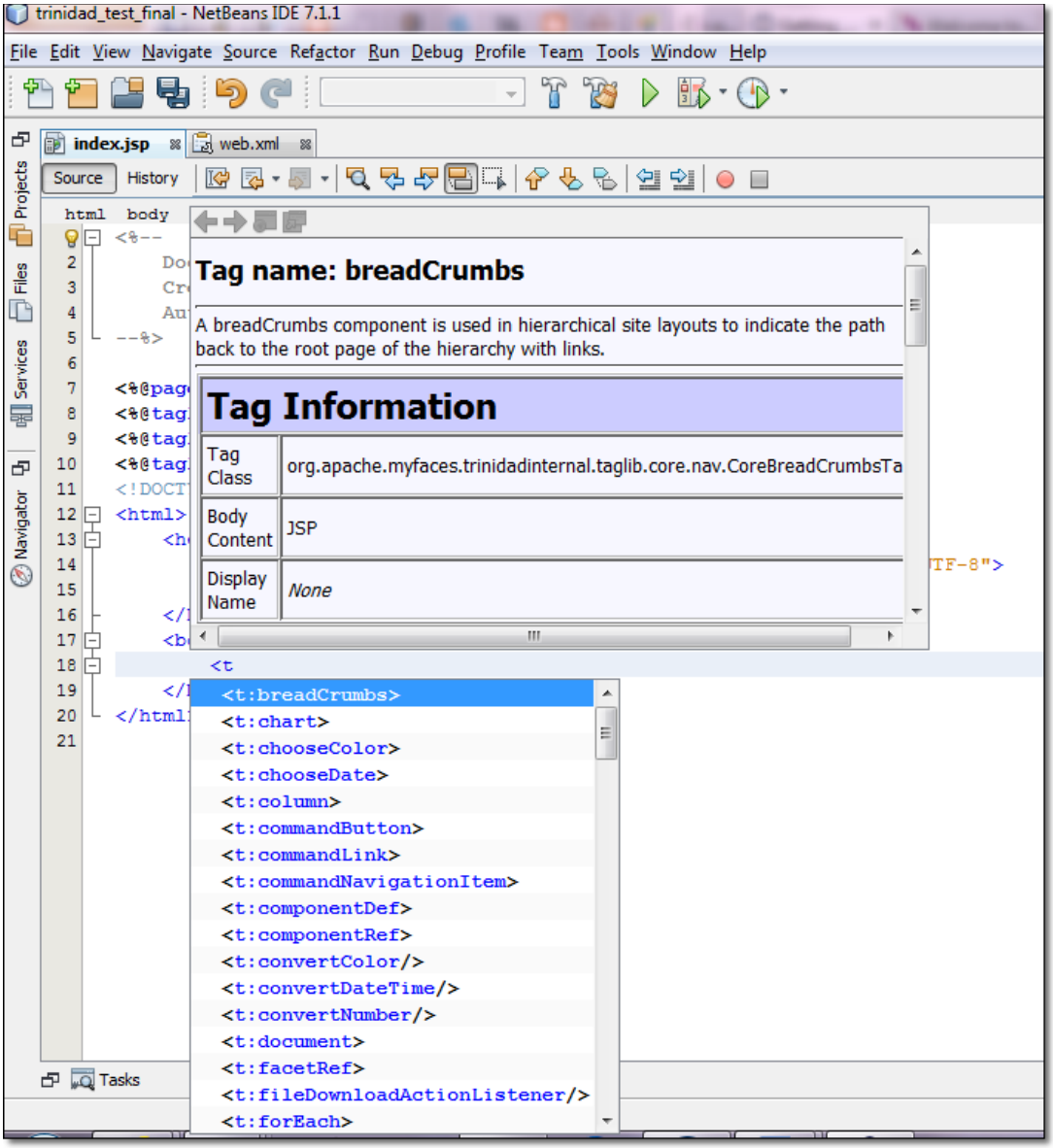
Específicamente el archivo **faces-config.xml** almacena las diferentes configuraciones y elementos a utilizar en la aplicación. El archivo faces-config.xml permite la edición visual de la navegación entre páginas, tanto para MyFaces Trinidad, IceFaces y RichFaces, tal como se muestra en la siguiente figura.

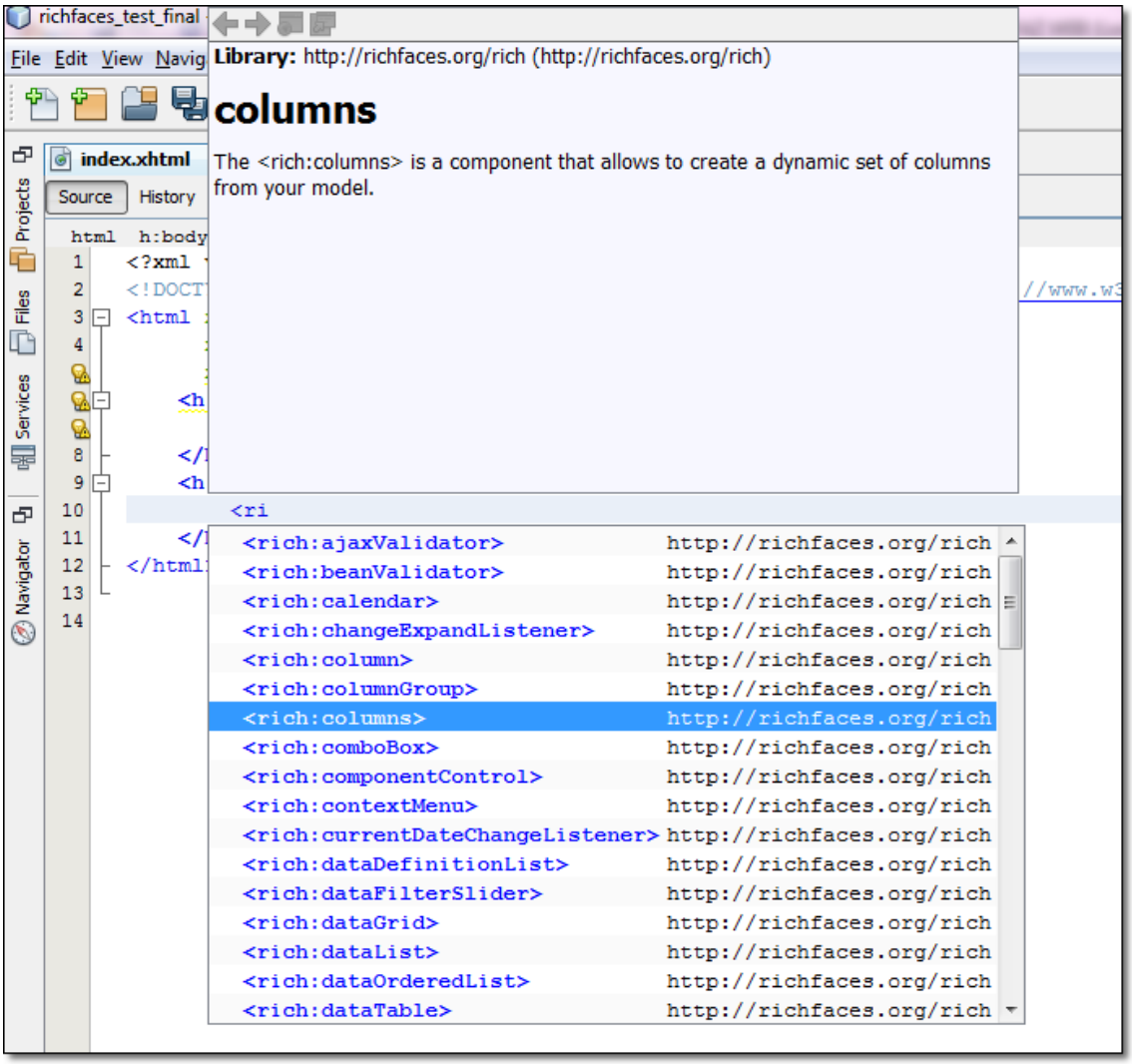


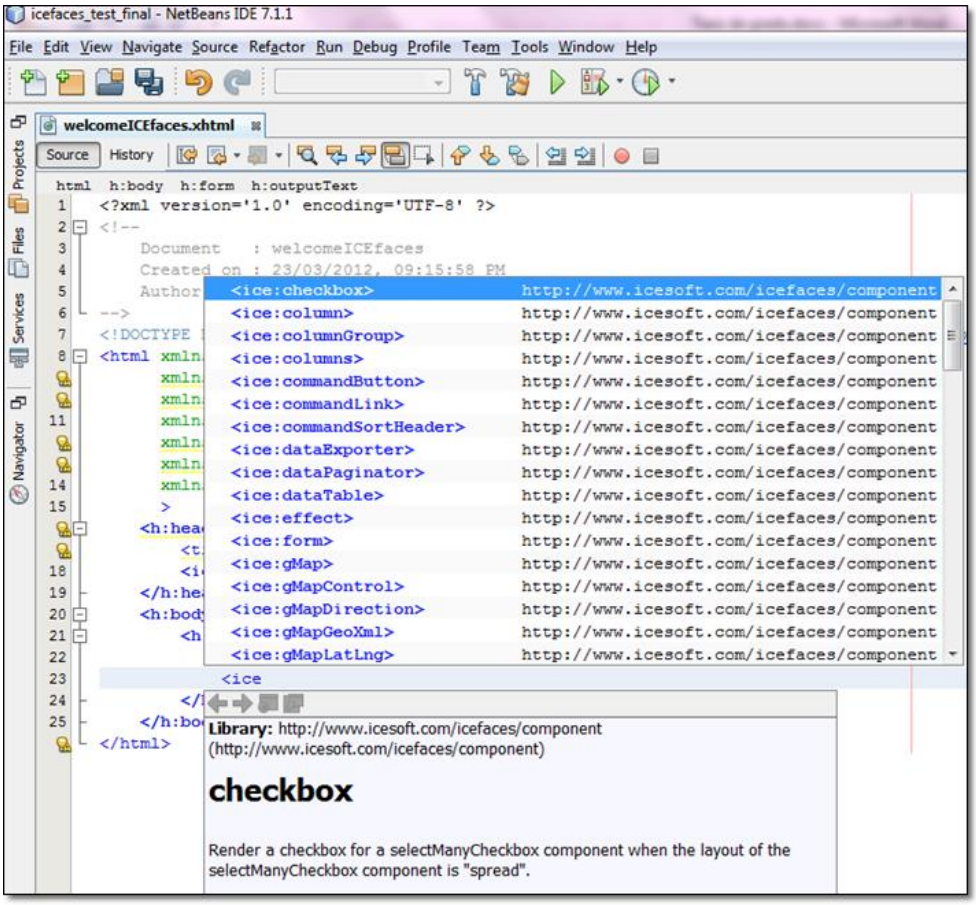
Así como se ilustra en la siguiente figura, el archivo **faces-config.xml** también se puede editar a través del código.

```
<?xml version='1.0' encoding='UTF-8'?>

<!-- ========== FULL CONFIGURATION FILE ================================== -->

<faces-config version="2.0"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-facesconfig_2_0.xsd">
    <navigation-rule>
        <from-view-id>/index.xhtml</from-view-id>
        <navigation-case>
            <from-outcome>case1</from-outcome>
            <to-view-id>/editarPaciente.xhtml</to-view-id>
        </navigation-case>
    </navigation-rule>
    <navigation-rule>
        <from-view-id>/editarPaciente.xhtml</from-view-id>
        <navigation-case>
            <from-outcome>case1</from-outcome>
            <to-view-id>/index.xhtml</to-view-id>
        </navigation-case>
    </navigation-rule>
</faces-config>
```

En las figuras siguientes se puede observar el autocompletado de etiquetas que proporciona NetBeans para los tres frameworks estudiados: MyFaces Trinidad, RichFaces e IceFaces respectivamente.

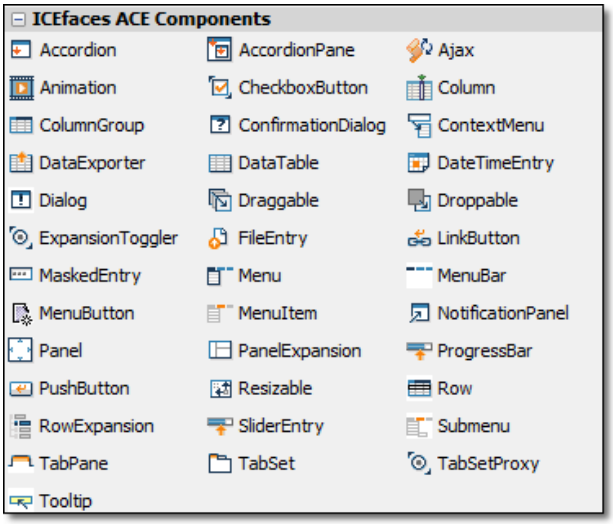



richfaces_test_final
File Edit View Navig
Library: http://richfaces.org/rich (http://richfaces.org/rich)
columns
The <rich:columns> is a component that allows to create a dynamic set of columns from your model.
index.xhtml
Source History
Projects
Files
Services
Navigator
<rich:ajaxValidator> http://richfaces.org/rich
<rich:beanValidator> http://richfaces.org/rich
<rich:calendar> http://richfaces.org/rich
<rich:changeExpandListener> http://richfaces.org/rich
<rich:column> http://richfaces.org/rich
<rich:columnGroup> http://richfaces.org/rich
<rich:columns> http://richfaces.org/rich
<rich:comboBox> http://richfaces.org/rich
<rich:componentControl> http://richfaces.org/rich
<rich:contextMenu> http://richfaces.org/rich
<rich:currentDateChangeListener> http://richfaces.org/rich
<rich:dataDefinitionList> http://richfaces.org/rich
<rich:dataFilterSlider> http://richfaces.org/rich
<rich:dataGrid> http://richfaces.org/rich
<rich:dataList> http://richfaces.org/rich
<rich:dataOrderedList> http://richfaces.org/rich
<rich:dataTable> http://richfaces.org/rich



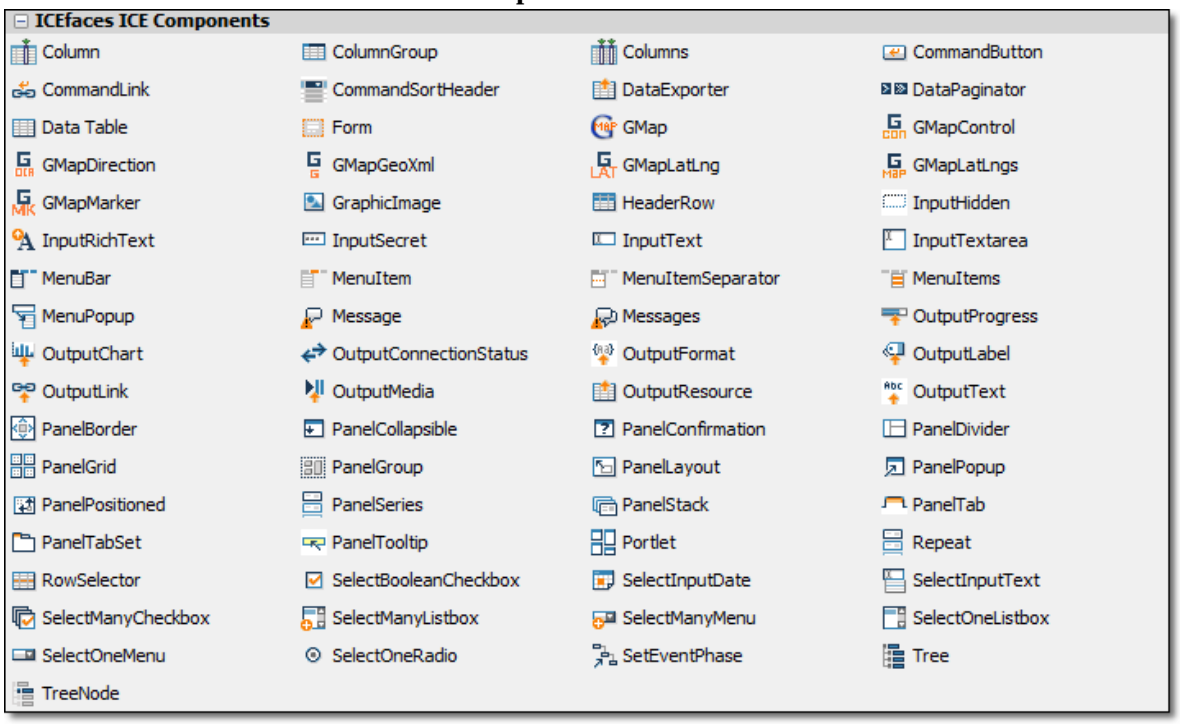
Adicionalmente al autocompletado de etiquetas NetBeans tiene incorporado al framework IceFaces una paleta de componentes, tal como se muestra a continuación.

Paleta de componentes ACE de IceFaces



Paleta de componentes Core de IceFaces

Paleta de componentes ICE de IceFaces



Resultado cuantitativo para la facilidad de uso

De la encuesta realizada (ver Anexo C), el resumen de la tabulación obtenida de la pregunta tres se presente en la siguiente tabla**.**

MYFACES TRINIDAD	ICEFACES	RICHFACES
10	15	10

El máximo valor que es 15 corresponde a IceFaces asignándole el porcentaje total del 20%, por tal motivo para obtener los valores de MyFaces Trinidad y RichFaces se realiza una regla de tres.

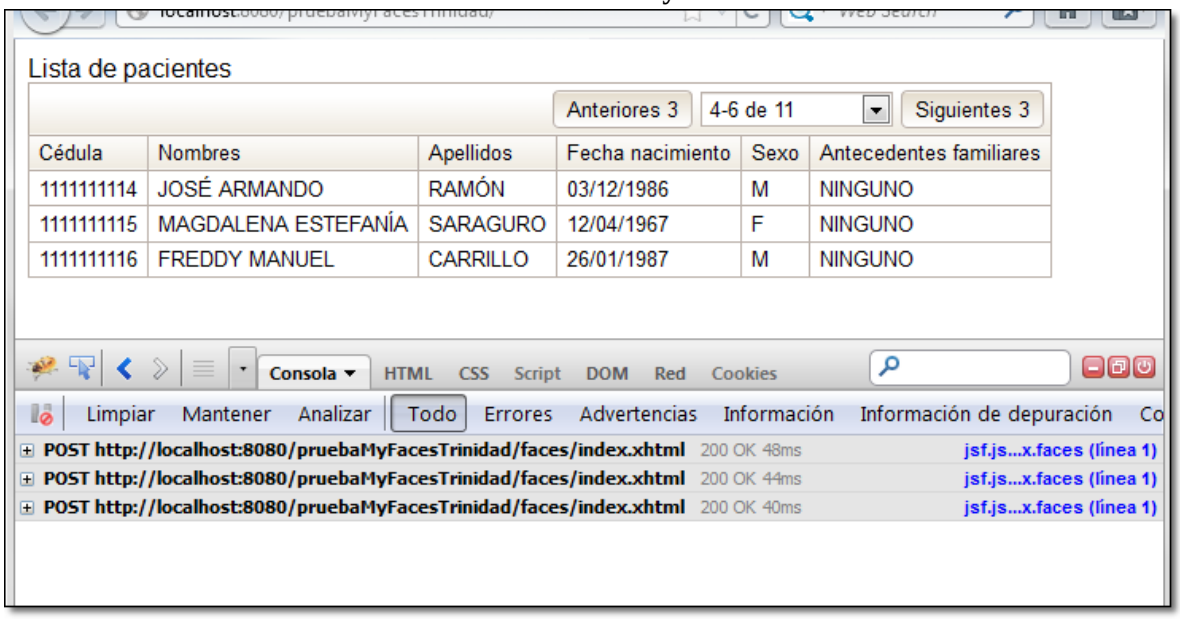
Para MyFaces Trinidad y RichFaces:

$$\frac{10 \times 20}{15} = 13{,}33\%$$

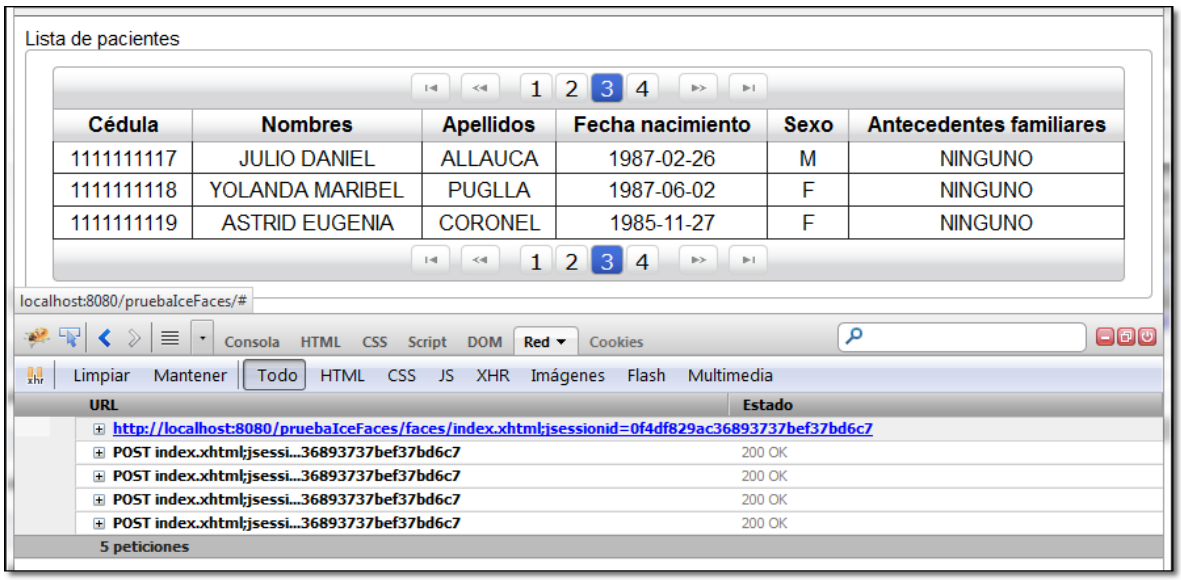
SECCIÓN 9: Facilidad para el desarrollo.- Soporte Ajax.

En las figuras siguientes, se observa una tabla con los datos de los pacientes y en la parte inferior un resumen de las peticiones AJAX, éstas se generaron al dar clic sobre la paginación de la tabla.

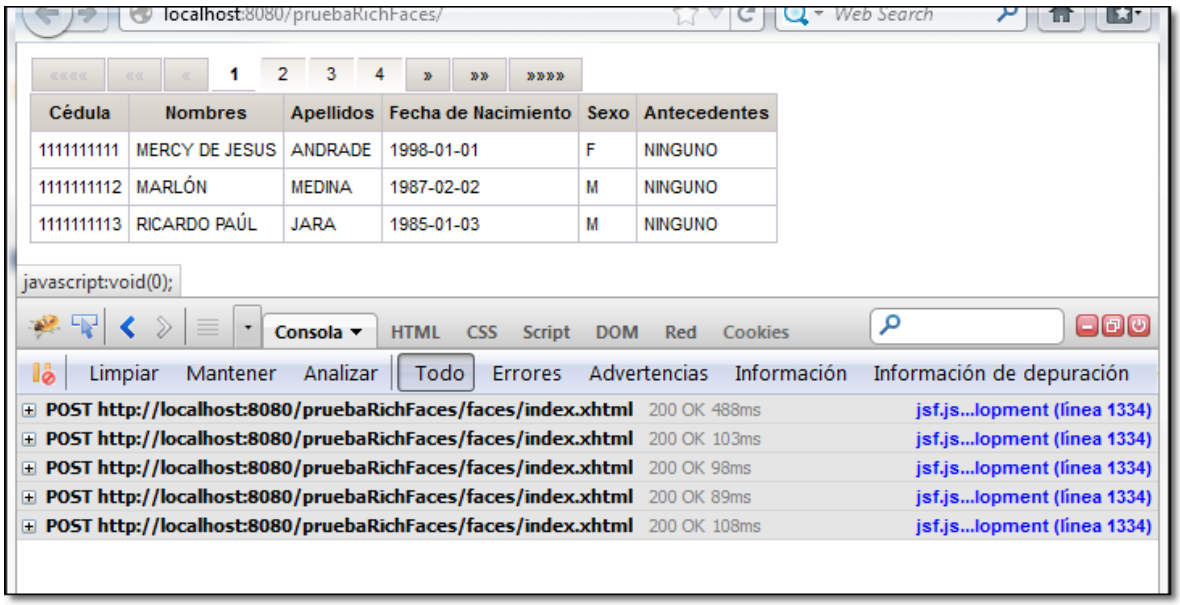
Monitorización AJAX en MyFaces Trinidad



Monitorización AJAX en IceFaces



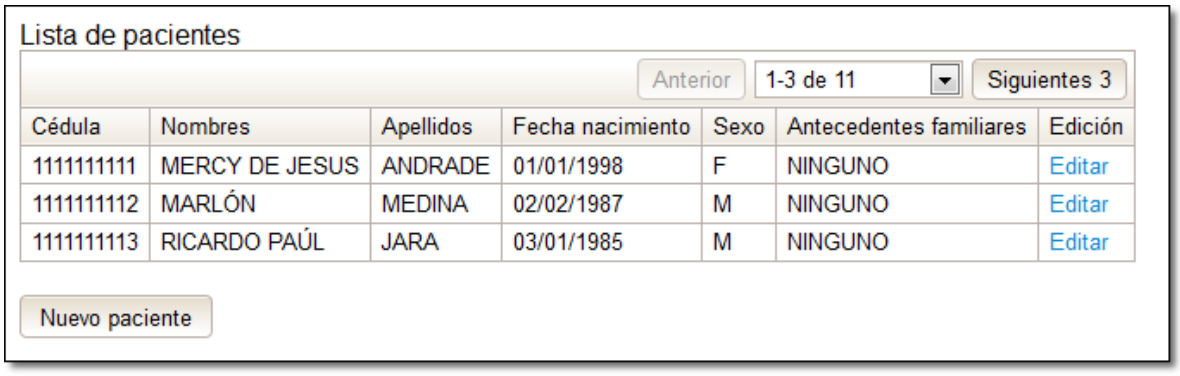
Monitorización AJAX en RichFaces



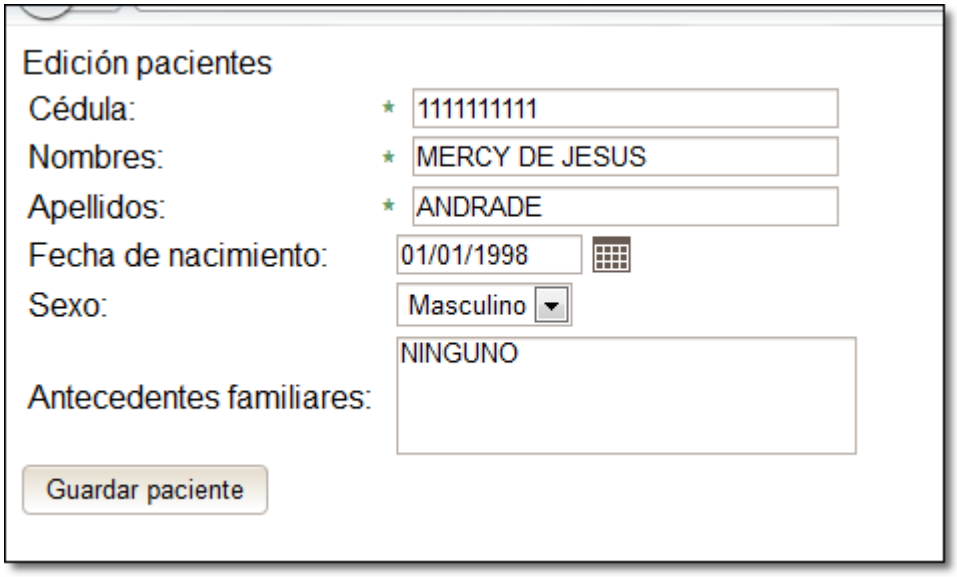
SECCIÓN 10: Facilidad para el desarrollo.- Líneas de código

El desarrollo de edición de pacientes consiste en la creación de dos vistas. La primera vista será para listar los datos de los pacientes (index.xhtml) y la segunda vista permitirá la edición de un paciente seleccionado o la creación de un nuevo paciente (edicion.xhtml); este prototipo de pruebas representa el ejemplo que los desarrolladores de DESITEL implementaron.

Lista de pacientes en MyFaces Trinidad



Edición de un paciente en MyFaces Trinidad



Líneas de código programadas en index.xhtml de MyFaces Trinidad

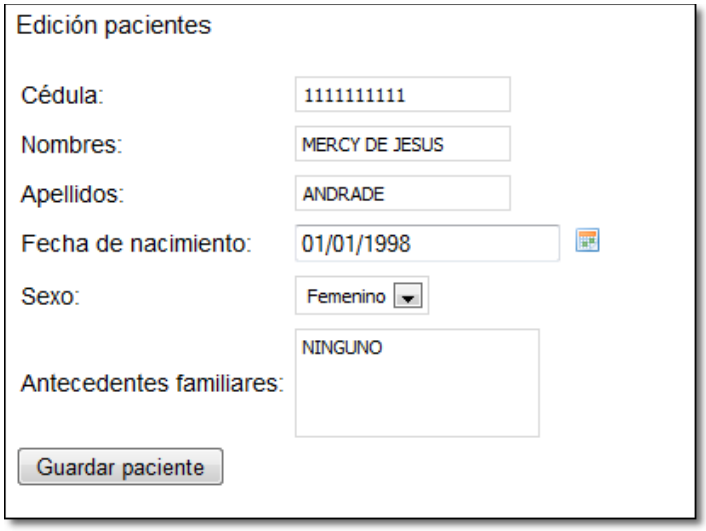
```
<tr:document>
    <tr:form>
        <tr:panelGroupLayout>
        <tr:outputText value="Lista de pacientes"/>
        <tr:table var="paciente" value="#{pacienteBean.listPaciente}" rows="3">
            <tr:column headerText="Cédula">
                <tr:outputText value="#{paciente.cedula}"/>
            </tr:column>
            <tr:column headerText="Nombres">
                <tr:outputText value="#{paciente.nombres}"/>
            </tr:column>
            <tr:column headerText="Apellidos">
                <tr:outputText value="#{paciente.apellidos}"/>
            </tr:column>
            <tr:column headerText="Fecha nacimiento">
                <tr:outputText value="#{paciente.fechaNacimiento}"/>
            </tr:column>
            <tr:column headerText="Sexo">
                <tr:outputText value="#{paciente.sexo}"/>
            </tr:column>
            <tr:column headerText="Antecedentes familiares">
                <tr:outputText value="#{paciente.antecedentes}"/>
            </tr:column>
            <tr:column headerText="Edición">
                <tr:commandLink text="Editar" action="case1" actionListener="#{pacienteBean.eventoEditar(evento)}"/>
            </tr:column>
        </tr:table>
        </tr:panelGroupLayout><br/>
        <tr:commandButton action="case1" actionListener="#{pacienteBean.eventoNuevo(evento)}" text="Nuevo paciente"/>
    </tr:form>
</tr:document>
```

Líneas de código programadas en edicion.xhtml de MyFaces Trinidad

```
<tr:document>
    <tr:form>
        <tr:panelGroupLayout>
            <tr:outputText value="Edición pacientes"/><br/>
            <h:panelGrid columns="2">
                <tr:outputText value="Cédula:"/>
                <tr:inputText value="#{pacienteBean.paciente.cedula}" required="true" autoSubmit="true"/>

                <tr:outputText value="Nombres:"/>
                <tr:inputText value="#{pacienteBean.paciente.nombres}" required="true" autoSubmit="true"/>

                <tr:outputText value="Apellidos:"/>
                <tr:inputText value="#{pacienteBean.paciente.apellidos}" required="true" autoSubmit="true"/>

                <tr:outputText value="Fecha de nacimiento:"/>
                <tr:inputDate value="#{pacienteBean.paciente.fechaNacimiento}">
                </tr:inputDate>
                <tr:outputText value="Sexo:"/>
                <tr:selectOneChoice value="#{pacienteBean.paciente.sexo}">
                    <f:selectItems value="#{pacienteBean.opcSexo}"/>
                </tr:selectOneChoice>
                <tr:outputText value="Antecedentes familiares:"/>
                <tr:inputText value="#{pacienteBean.paciente.antecedentes}" rows="3"/>
            </h:panelGrid>
            <tr:messages/>
            <tr:commandButton text="Guardar paciente" action="case1" actionListener="#{pacienteBean.eventoGuardar(evento)}"/>
        </tr:panelGroupLayout>
    </tr:form>
</tr:document>
```

Lista de pacientes en IceFaces

Edición de un paciente en IceFaces



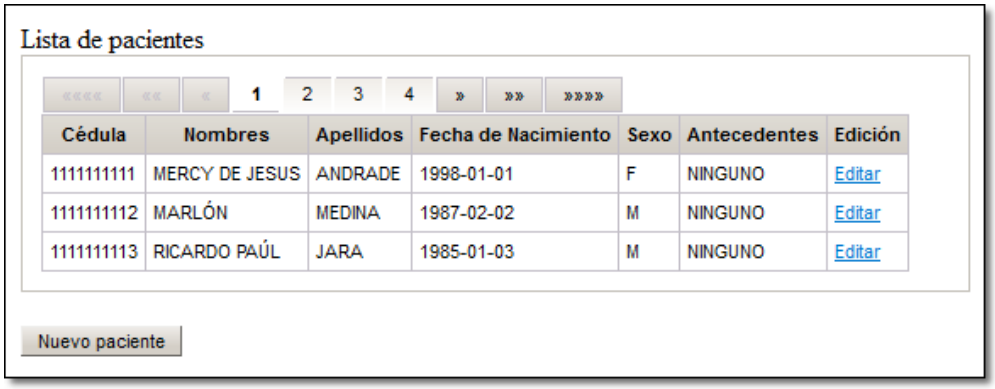
Líneas de código programadas en index.xhtml de IceFaces

```
<ice:form>
    <ice:outputText value="Lista de pacientes"></ice:outputText><br/>
    <ace:panel>
        <ace:dataTable var="paciente" value="#{pacienteBean.listPaciente}" rows="3" paginator="true">
            <ace:column headerText="Cédula">
                <ice:outputText value="#{paciente.cedula}"/>
            </ace:column>
            <ace:column headerText="Nombres">
                <ice:outputText value="#{paciente.nombres}"/>
            </ace:column>
            <ace:column headerText="Apellidos">
                <ice:outputText value="#{paciente.apellidos}"/>
            </ace:column>
            <ace:column headerText="Fecha nacimiento">
                <ice:outputText value="#{paciente.fechaNacimiento}"/>
            </ace:column>
            <ace:column headerText="Sexo">
                <ice:outputText value="#{paciente.sexo}"/>
            </ace:column>
            <ace:column headerText="Antecedentes familiares">
                <ice:outputText value="#{paciente.antecedentes}"/>
            </ace:column>
            <ace:column headerText="Edición">
                <ice:commandLink value="Editar" actionListener="#{pacienteBean.eventoEditar(evento)}" action="case1"/>
            </ace:column>
        </ace:dataTable>
    </ace:panel><br/>
    <ice:commandButton action="case1" actionListener="#{pacienteBean.eventoNuevo(evento)}" value="Nuevo paciente"/>
</ice:form>
```

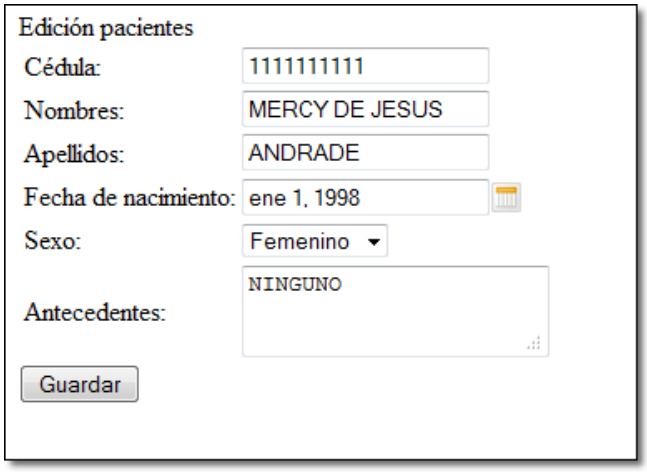
Líneas de código programadas en edición.xhtml de IceFaces

```
<ice:form>
    <ice:outputText value="Edición pacientes"/><br/>
    <ice:panelGroup>
        <ice:panelGrid columns="2">
            <ice:outputText value="Cédula:"/>
            <ice:inputText value="#{pacienteBean.paciente.cedula}" required="true" partialSubmit="true" label="cédula"/>

            <ice:outputText value="Nombres:"/>
            <ice:inputText value="#{pacienteBean.paciente.nombres}" required="true" partialSubmit="true" label="nombres"/>

            <ice:outputText value="Apellidos:"/>
            <ice:inputText value="#{pacienteBean.paciente.apellidos}" required="true" partialSubmit="true" label="apellidos"/>

            <ice:outputText value="Fecha de nacimiento:"/>
            <ice:selectInputDate value="#{pacienteBean.paciente.fechaNacimiento}" renderAsPopup="true"/>

            <ice:outputText value="Sexo:"/>
            <ice:selectOneMenu value="#{pacienteBean.paciente.sexo}">
                <f:selectItems value="#{pacienteBean.opcSexo}"/>
            </ice:selectOneMenu>

            <ice:outputText value="Antecedentes familiares:"/>
            <ice:inputTextarea value="#{pacienteBean.paciente.antecedentes}" rows="3"/>
        </ice:panelGrid>
        <ice:messages/>
        <ice:commandButton action="case1" actionListener="#{pacienteBean.eventoGuardar(evento)}" value="Guardar paciente"/>
    </ice:panelGroup>
</ice:form>
```

Lista de pacientes en RichFaces



Edición de un paciente en RichFaces



Líneas de código programadas en index.xhtml de RichFaces

```
<h:form>
    <h:outputText value="Lista de pacientes"/>
    <rich:panel>
        <rich:dataScroller for="table" maxPages="5"></rich:dataScroller>
        <rich:dataTable id="table" value="#{pacienteBean.listPaciente}"
                        var="paciente" rows="3">
                <rich:column>
                    <f:facet name="header">
                        <h:outputText value="Cédula"/>
                    </f:facet>
                    <h:outputText value="#{paciente.cedula}"/>
                </rich:column>
                <rich:column>
                    <f:facet name="header">
                        <h:outputText value="Nombres"/>
                    </f:facet>
                    <h:outputText value="#{paciente.nombres}"/>
                </rich:column>
                <rich:column>
                    <f:facet name="header">
                        <h:outputText value="Apellidos"/>
                    </f:facet>
                    <h:outputText value="#{paciente.apellidos}"/>
                </rich:column>
                <rich:column>
                    <f:facet name="header">
                        <h:outputText value="Fecha de Nacimiento"/>
                    </f:facet>
                    <h:outputText value="#{paciente.fechaNacimiento}"/>
                 <rich:column>
                     <f:facet name="header">
                         <h:outputText value="Sexo"/>
                     </f:facet>
                     <h:outputText value="#{paciente.sexo}"/>
                 </rich:column>
                 <rich:column>
                     <f:facet name="header">
                         <h:outputText value="Antecedentes"/>
                     </f:facet>
                     <h:outputText value="#{paciente.antecedentes}"/>
                 </rich:column>
            <rich:column>
                <f:facet name="header">
                    <h:outputText value="Edición"/>
                </f:facet>
                <h:commandLink value="Editar" action="case1" actionListener="#{pacienteBean.eventoEditar(evento)}"/>
            </rich:column>
        </rich:dataTable>
    </rich:panel><br/>
    <h:commandButton action="case1" actionListener="#{pacienteBean.eventoNuevo(evento)}" value="Nuevo paciente"/>
</h:form>
```

Líneas de código programadas en edición.xhtml de RichFaces

```
<h:form>
    <h:outputText value="Edición pacientes"/>
    <h:panelGroup>
        <h:panelGrid columns="2">
           <h:outputText value="Cédula:"/>
           <h:inputText value="#{pacienteBean.paciente.cedula}" required="true" label="cédula"/>

           <h:outputText value="Nombres:"/>
           <h:inputText value="#{pacienteBean.paciente.nombres}" required="true" label="nombres"/>

           <h:outputText value="Apellidos:"/>
           <h:inputText value="#{pacienteBean.paciente.apellidos}" required="true" label="apellidos"/>

           <h:outputText value="Fecha de nacimiento:"/>
           <rich:calendar value="#{pacienteBean.paciente.fechaNacimiento}"/>

           <h:outputText value="Sexo:"/>
           <h:selectOneMenu value="#{pacienteBean.paciente.sexo}">
                <f:selectItems value="#{pacienteBean.opcSexo}"/>
           </h:selectOneMenu>

           <h:outputText value="Antecedentes:"/>
           <h:inputTextarea value="#{pacienteBean.paciente.antecedentes}"/>
        </h:panelGrid>
        <rich:messages/>
        <a4j:commandButton action="case1" actionListener="#{pacienteBean.eventoGuardar(evento)}" value="Guardar"/>
    </h:panelGroup>
</h:form>
```

Resultado cuantitativo para las líneas de código

De la encuesta realizada (ver Anexo C), el resumen de la tabulación obtenida de la pregunta tres se presente en la siguiente tabla**.**

MYFACES TRINIDAD	ICEFACES	RICHFACES
15	15	10

El máximo valor que es 15 corresponde a MyFaces Trinidad y a IceFaces asignándole el porcentaje total del 20%, por tal motivo para obtener el valor de RichFaces se realiza una regla de tres.

Para RichFaces:

$$\frac{10 \times 20}{15} = 13{,}33\%$$

SECCIÓN 11: Análisis de resultados

Para MyFaces Trinidad:

$$Total\ MyFaces\ Trinidad = \frac{38{,}32 \times 10}{100} + \frac{100 \times 10}{100} + \frac{87{,}93 \times 30}{100} + \frac{71{,}99 \times 50}{100} = 76{,}21\%$$

Para IceFaces:

$$Total\ IceFaces = \frac{76{,}93 \times 10}{100} + \frac{70{,}80 \times 10}{100} + \frac{100 \times 30}{100} + \frac{100 \times 50}{100} = 94{,}77\%$$

Para RichFaces:

$$Total\ RichFaces = \frac{73{,}61 \times 10}{100} + \frac{62{,}18 \times 10}{100} + \frac{21{,}55 \times 30}{100} + \frac{73{,}32 \times 50}{100} = 56{,}70\%$$

SECCIÓN 12: Comprobación de la hipótesis.- Cálculo del estadístico

Tabla: Frecuencias Observadas

		Variable Independiente			
Variable dependiente	**Parámetros**	**MyFaces Trinidad**	**IceFaces**	**RichFaces**	**TOTAL**
H_1 Eficiencia	Aprendizaje	0.001	7.700	7.360	15.061
	Calidad	10.000	7.080	6.220	23.300
	Componentes de interfaz de usuario	26.380	30.000	6.470	62.850
	Facilidad para el desarrollo	36.000	50.000	36.660	122.660
H_0 No eficiencia	Aprendizaje	3.380	0.001	0.001	3.382
	Calidad	0.001	0.001	0.001	0.003
	Componentes de interfaz de usuario	0.001	0.001	0.001	0.003
	Facilidad para el desarrollo	0.001	0.001	0.001	0.003
TOTAL		75.764	94.784	56.714	227.262

Tabla: Frecuencias Esperadas

		Variable Independiente			
Variable dependiente	**Parámetros**	**MyFaces Trinidad**	**IceFaces**	**RichFaces**	**TOTAL**
H_1 Eficiencia	Aprendizaje	5.021	6.281	3.759	15.061
	Calidad	7.768	9.718	5.815	23.300
	Componentes de interfaz de usuario	20.953	26.213	15.684	62.850
	Facilidad para el desarrollo	40.892	51.158	30.610	122.660
H_0 No eficiencia	Aprendizaje	1.127	1.411	0.844	3.382
	Calidad	0.001	0.001	0.001	0.003
	Componentes de interfaz de usuario	0.001	0.001	0.001	0.003
	Facilidad para el desarrollo	0.001	0.001	0.001	0.003
TOTAL		75.764	94.784	56.714	227.262

SECCIÓN 13: Comprobación de la hipótesis.- Grados de libertad

Tabla: Distribución de chi-cuadrado

	Probabilidad de un valor superior				
Grados de libertad	**0.1**	**0.05**	**0.025**	**0.01**	**0.005**
1	2.71	3.84	5.02	6.63	7.88
2	4.61	5.99	7.38	9.21	10.60
3	6.25	7.81	9.35	11.34	12.84
4	7.78	9.49	11.14	13.28	14.86
5	9.24	11.07	12.83	15.09	16.75
6	10.64	12.59	14.45	16.81	18.55
7	12.02	14.07	16.01	18.48	20.28
8	13.36	15.51	17.53	20.09	21.95
9	14.68	16.92	19.02	21.67	23.59
10	15.99	18.31	20.48	23.21	25.19
11	17.28	19.68	21.92	24.73	26.76
12	18.55	21.03	23.34	26.22	28.3
13	19.81	22.36	24.74	27.69	29.82
14	21.06	**23.68**	26.12	29.14	31.32
15	22.31	25	27.49	30.58	32.8
16	23.54	26.3	28.85	32	34.27
17	24.77	27.59	30.19	33.41	35.72
18	25.99	28.87	31.53	34.81	37.16

Anexo

B

ENCUESTA Y TABULACIÓN

SECCIÓN 1: Encuesta de valoración de características de desarrollo

ENCUESTA DE VALORACIÓN DE CARACTERÍSTICAS DE DESARROLLO

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA EN SISTEMAS

Objetivo: Determinar el framework más eficiente para desarrollar el Sistema Nutricional de la ESPOCH.

CUESTIONARIO

Seleccione la opción que en su opinión corresponde.

1. ¿El instalar los frameworks en el IDE NetBeans lo considera?

	MyFaces Trinidad	**IceFaces**	**RichFaces**
Muy fácil	◯	◯	◯
Fácil	◯	◯	◯
Difícil	◯	◯	◯
Muy difícil	◯	◯	◯

2. ¿Configurar los frameworks en el IDE NetBeans lo considera?

	MyFaces Trinidad	**IceFaces**	**RichFaces**
Muy fácil	◯	◯	◯
Fácil	◯	◯	◯
Difícil	◯	◯	◯
Muy difícil	◯	◯	◯

3. ¿La manipulación de los frameworks en el IDE NetBeans lo considera?

	MyFaces Trinidad	**IceFaces**	**RichFaces**
Muy fácil	◯	◯	◯
Fácil	◯	◯	◯
Difícil	◯	◯	◯
Muy difícil	◯	◯	◯

4. Al programar la interfaz de usuario en los frameworks. ¿Cómo considera el número de líneas de código (LDC) que escribieron?

	MyFaces Trinidad	**IceFaces**	**RichFaces**
Muy poco	◯	◯	◯
Algo	◯	◯	◯
Bastante	◯	◯	◯
Mucho	◯	◯	◯

5. Señale los frameworks que disponen del componente que permita realizar gráficas en 2D.

MyFaces Trinidad	**IceFaces**	**RichFaces**
◯	◯	◯

SECCIÓN 2: Tabulación de la encuesta

Para la tabulación de los datos se consideró los valores cualitativos – cuantitativos, antes mencionados en los criterios de evaluación.

(Programadores encuestados) N = 5

1. ¿El instalar los frameworks en el IDE NetBeans lo considera?

MyFaces Trinidad

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	0	0
Fácil	2	4	8
Difícil	1	1	1
Muy difícil	0	0	0
TOTAL			**9**

IceFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	4	12
Fácil	2	1	3
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**15**

RichFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	0	0
Fácil	2	5	10
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**10**

2. ¿Configurar los frameworks lo considera?

MyFaces Trinidad

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	0	0
Fácil	2	5	10
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**10**

IceFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	5	15
Fácil	2	0	0
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**15**

RichFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	0	0
Fácil	2	5	10
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**10**

3. ¿La facilidad de uso de los frameworks en el IDE NetBeans lo considera?

MyFaces Trinidad

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	0	0
Fácil	2	5	10
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**10**

IceFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	5	15
Fácil	2	0	0
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**15**

RichFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	0	0
Fácil	2	5	10
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**10**

4. Al programar la interfaz de usuario en los frameworks. ¿Cómo considera el número de líneas de código (LDC) que escribieron?

MyFaces Trinidad

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy poco	3	5	15
Algo	2	0	0
Bastante	1	0	0
Mucho	0	0	0
TOTAL			**15**

IceFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	5	15
Fácil	2	0	0
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**15**

RichFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Muy fácil	3	0	0
Fácil	2	5	10
Difícil	1	0	0
Muy difícil	0	0	0
TOTAL			**10**

5. Señale los frameworks que disponen del componente que permita realizar gráficas en 2D.

MyFaces Trinidad

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Si	3	5	15
No	0	0	0
TOTAL			**15**

IceFaces

Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Si	3	5	15
No	0	0	0
TOTAL			**15**

RichFaces

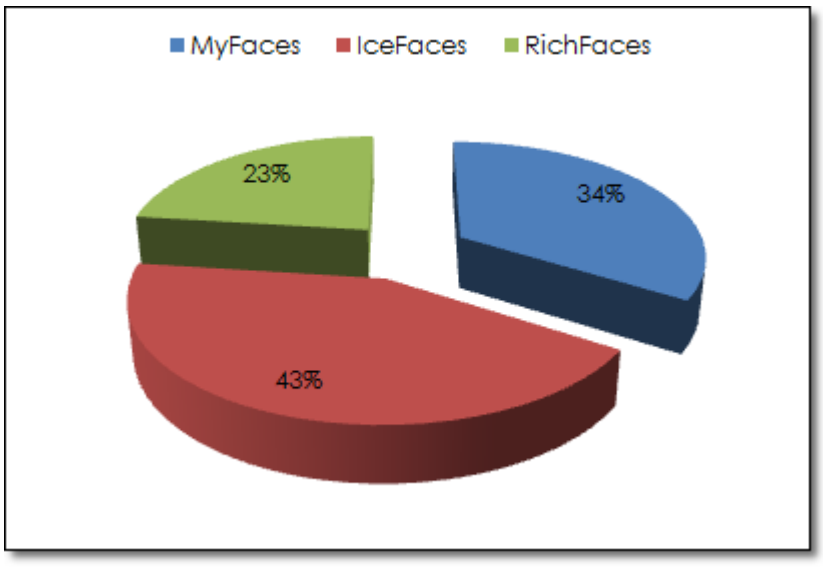
Valor cualitativito	Valor cuantitativo (vc)	Frecuencia (f)	vc * f
Si	3	0	0
No	0	5	0
TOTAL			**0**

RESUMEN DE LA ENCUESTA REALIZADA

En la **Tabla,** se resume el total obtenido por cada pregunta realizada.

Framework / Pregunta	MyFaces	IceFaces	RichFaces
Pregunta N° 1	9	15	10
Pregunta N° 2	10	15	10
Pregunta N° 3	10	15	10
Pregunta N° 4	15	15	10
Pregunta N° 5	15	15	0
TOTAL	59	75	40

Los valores anteriores en porcentaje, se presentan en el siguiente pastel**.**



G L O S A R I O D E T É R M I N O S

AJAX: JavaScript asíncrono y XML, es una técnica de desarrollo Web para crear aplicaciones interactivas o RIA.

API: (Application Programming Interface, interfaz de programación de aplicaciones), es el conjunto de funciones o procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Archivo de texto plano: Archivo informático compuesto únicamente por texto sin formato.

Cliente: Persona u organización que recibe, usa un producto o servicio. Un cliente puede ser interno o externo a la organización del suministrador.

Desarrollador: Persona encargada de analizar los requerimientos para crear la aplicación eficiente.

Diseño: Proceso de definición de la arquitectura, y otras características de un sistema o componente.

DOM: Document Object Model es esencialmente un modelo computacional a través del cual los programas y scripts pueden acceder y modificar dinámicamente el contenido, estructura y estilo de los documentos HTML y XML. Su objetivo es ofrecer un modelo orientado a objetos para el tratamiento y manipulación en tiempo real (o de forma dinámica) a la vez de manera estática de páginas de internet.

Especificación de requisitos: Proceso de redacción o registro de los requisitos.

Evaluación: El uso de revisiones, inspecciones, y /o pruebas para determinar que un producto o servicio software, hardware, etc., satisface los criterios o especificaciones previamente establecidos.

Evaluación Nutricional: Evaluaciones antropométricas, de requerimientos de energía, síndrome metabólico y riesgo cardiovascular realizadas a los pacientes.

e-commerce: Comercio electrónico, consiste en la compra y venta de productos o de servicios a través de medios electrónicos.

FTP: Protocolo de transferencia de archivos entre sistemas conectados a una red TCP, basado en la arquitectura cliente – servidor.

Hardware: Conjunto de los componentes que integran la parte material de una computadora.

HTTP: Protocolo de transferencia de hipertexto, orientado a transacciones y sigue el esquema entre petición-respuesta entre un cliente y servidor.

IDE (Integrated Development Environment, entorno de desarrollo integrado), es un programa informático compuesto por un conjunto de herramientas de programación

J2EE: Java Plataform, Enterprise Edition (JEE) anteriormente conocido como Java 2 Plataform, Enterprise Edition (J2EE).

look-and-feel: Es un término que se utiliza en relación a la interfaz gráfica de usuario y comprende los aspectos de su diseño, incluyendo elementos como colores, formas y tipo de letra ("look"), así como el comportamiento de los elementos dinámicos tales como los botones, cuadros y menús ("feel").

NutriSys: Nombre asignado al Sistema Nutricional de la ESPOCH.

Paciente: Individuo que debe ser atendido de manera rápida y eficiente.

Proceso: Conjunto de actividades interrelacionadas que usan recursos para transformar entradas en salidas.

Producto: Resultado de un proceso. Un producto software está relacionado con cuatro categorías de productos como son el hardware, software, comunicaciones, servicios.

Profesional en nutrición: Persona encargada de manipular la información personal, nutricional y de laboratorio del paciente; además maneja la aplicación para obtener resultados, y en base a estos tomar decisiones adecuadas con respecto al paciente.

Red: Las redes están formadas por conexiones entre grupos de ordenadores y dispositivos asociados que permiten a los usuarios la transferencia electrónica de información.

Requisito: Necesidad o expectativa que se establece de forma explícita o implícita.

Servicio: Producto intangible que es el resultado de realizar al menos una actividad en la interfaz entre el suministrador y el cliente.

Sistema: Conjunto de elementos interrelacionados en uno o más de los procesos que proporcionan la capacidad de satisfacer una necesidad u objetivo definido. Un sistema puede ser considerado como un producto o como el servicio que proporciona.

Software: Aplicaciones que ayudan a crear otras aplicaciones y/o son aplicaciones finales para ser usadas.

Usuario: Una persona u organización que usa el sistema para realizar una función específica.

Validación: Confirmación mediante examen y provisión de evidencia objetiva de que se cumplen los requisitos particulares para ser usado con un propósito específico y que satisface las necesidades del cliente.

WEB: World Wide Web (**WWW**) o **Red informática mundial** es un sistema de distribución de información basado en hipertexto o hipermedios enlazados y accesibles a través de Internet.

WHO: World Health Organization. Organización Mundial de la Salud, de esta se toman todos los datos que servirán como estándares para la realización de las distintas evaluaciones.

XML: Es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML pero su función principal es describir datos y no mostrarlos como es el caso de HTML.

B I B L I O G R A F Í A

1. **ALLAMARJU, S., y otros**., Programación Java Server con J2EE., Madrid – España. Anaya Multimedia-Anaya Interactiva., 2002., Pp. 55.

2. **THOMAS, D.,** Apache MyFaces Trinidad 1.2., Birmingham-United Kingdom-Reino Unido., Pack Publishing., 2009., Pp. 8-10.

3. **MEDÍN, J.,** Hacia una Arquitectura con JavaServer Faces, Spring, Hibernate y otros Frameworks., Madrid-España., 2006., Pp. 7.

4. **QUESADA, J.,** Java Server Faces y el uso de Patrones de Diseño.,..Universidad de Costa Rica., UCR Puntaneras-Costa Rica **II CONGRESO DE COMPUTACIÓN PARA EL DESARROLLO**., Pp. 3-4.

5. **ABARCA, M.,** Manual de Desarrollo Básico de Aplicaciones en la Plataforma J2EE en Ubuntu 7.x., Pp. 9.

6. **CACIÁ, D.,** Arquitectura de Aplicaciones J2EE basadas en el Patrón MVC utilizando Oracle ADF., Universidad de San Carlos de Guatemala., Facultad de Ingeniería., Escuela de Ingeniería en Ciencias y Sistema., Guatemala., **TRABAJO DE GRADUACIÓN**., 2007., Pp. 6.

7. **FERNÁNDEZ, E.,** Sistema de gestión de productos con emulación RFID mediante sensores IEEE 802.15.4., Universidad Politécnica de Catalunya., **PROYECTO FINAL DE CARRERA.**, 2008., Pp. 16-17.

8. **GARCÍA, R.,** Diseño e Implementación de un Framework de Persistencia., Universitat Oberta de Catalunya., Barcelona-España., **MEMORIA PFC.**, 2008., Pp. 17.

9. **GÓMEZ, A.,** Implementación de una Aplicación Web utilizando Frameworks J2EE., Universidad de Barcelona., Facultad de Matemáticas., Departamento de Matemáticas., Barcelona-España., **TRABAJO FINAL DE CARRERA**., 2008., Pp. 11.

10. **PALACÍN, M.,** Portal Web 2.0 utilizando Framework Struts., Universidad Politécnica de Catalunya., España., **TRABAJO DE FIN DE CARRER.**, 2007., Pp. 60.

11. **RUFO, D.,** Sede Electrónica., Universidad Rey Juan Carlos., Escuela Superior de Ingeniería Informática., Ingeniería Informática., **PROYECTO DE FIN DE CARRERA.**, 2010., Pp. 10.

12. **VIVEROS, S.,** Sistema generador de Aplicaciones Web configurables para el registro en línea congresos utilizando JavaServer Faces., Universidad de las Américas Puebla., Escuela de Ingeniería y Ciencias., Chochula-Puebla-México., **TESIS**., 2006., Pp. 14.

B I B L I O G R A F Í A D E I N T E R N E T

13. **ACERCA DE LA TECNOLOGÍA JAVA**

http://www.codexion.com/tutorialesjava/getStarted/intro/definition.html#FOOT

2012-01-03

14. **AMAZON**

https://www.amazon.com

2012-03-1

15. **AMAZON, LA MEJOR COMPRA PARA REALIZAR COMPRAS EN INTERNET**

http://diarioadn.co/actualidad/tecnología/páginas-para-realizar-compras-en-internet-1.39708

2012-03-1

16. **APLICACIÓN WEB PARA LA GESTIÓN DE EXPLOTACIONES AGRÍCOLAS**

http://es.scribd.com/doc/34022342/5/Componentes-y-Contenedores-de-J2EE

2012-01-15

17. **CARÁCTERÍSTICAS DE JAVA**

http://www.webtaller.com/manual-java/caracteristicas-java.php

2012-01-03

18. **CONOCIMIENTO CON TODOS Y PARA TODOS**

http://www.ecured.cu/index.php/JSF

2012-02-11

19. **CURSO DE JAVA SERVER FACES 2 CON HIBERNATE 3**

http://www.scribd.com/doc/28079982/Curso-de-JSF-2-0-con-Hibernate-3

2012-01-17

20. **DESCRIPCIÓN DE J2ME**

http://leo.ugr.es/J2ME/INTRO/intro_2.htm

2012-01-03

21. **FRAMEWORKS EN JAVA**

http://www.slideshare.net/jlbugarin/frameworks-java-14568352010-

2010-11-06

22. GUÍA DE REFERENCIA DE JSF

http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=GuiaJSF

2012-02-23

23. ICEFACES

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/103

2012-01-31

24. ICEFACES

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/133

2012-01-31

25. ICESOFT JIRA. ICEFACES. OPEN ISSUES.

http://jira.icesoft.org/browse/ICE?report=com.atlassian.jira.plugin.system.project:openissues-panel

2012-03-27

26. INTRODUCCIÓN A LOS SERVLETS

http://parasitovirtual.wordpress.com/category/cursos-y-articulos/desarrollo-aplicaciones-software/java/j2ee/

2012-01-13

27. INTRODUCCIÓN A RICHFACES

http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro

2012-02-04

28. JAVA (LENGUAJE DE PROGRAMACIÓN)

http://es.wikipedia.org/wiki/Java_%28lenguaje_de_programaci%C3%B3n%29

2012-01-03

29. JAVASERVERFACES

http://code.google.com/p/fap-devel/wiki/JavaServerFaces

2012-02-21

30. JAVASERVER FACES

http://es.wikipedia.org/wiki/JavaServer_Faces

2012-01-17

31. JAVASERVER FACES (JSF)

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/101

2012-02-21

32. JAVA SERVER FACES PARTE I

http://www.rincondeloajeno.com/introduccion-a-jsf-java-server-faces-parte-i/

2012-01-17

33. JAVA, SU HISTORIA, EDICIONES, VERSIONE Y CARACTERÍSTICAS COMO PLATAFORMA Y LENGUAJE DE PROGRAMACIÓN

http://www.clubdesarrolladores.com/articulos/mostrar/38-java-su-historia-ediciones-versiones-y-caracteristicas-como-plataforma-y-lenguaje-de-programacion/1

2012-01-03

34. JSF

http://150.185.75.30/atiwiki/index.php/JSF#Ciclo_de_vida

2012-02-23

35. JSF

http://www.ecured.cu/index.php?title=Especial:Pdfprint&page=JSF

2012-01-25

36. JSF JAVA SERVER FACES

http://www.scribd.com/doc/73476293/nDeveloper-JavaServerFaces

2012-01-17

37. MANUAL DE DESARROLLO BÁSICO DE APLICACIONES EN LA PALATAFORMA J2EE EN UBUNTU 7.X

http://xxito.files.wordpress.com/2008/05/manualj2ee.pdf

2012-01-11

38. MODEL VIEW CONTROLLER + JSF

http://blog.pucp.edu.pe/item/4825/model-view-controler-jsf.

2010-11-04

39. MOJARRA

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/92

2012-02-17

40. OJO VIRTUAL PFC

http://upcommons.upc.edu/pfc/bitstream/2099.1/9616/1/62016.pdf

2012-01-31

41. OPINIONES DE APACHE MYFACES

http://www.datuopinion.com/apache-myfaces

2012-02-04

42. ORCHESTRA

http://myfaces.apache.org/orchestra/index.html

2012-02-5

43. PRIMAFACES VS RICHFACES VS ICEFACES

http://www.clearminds-it.com/index.php?/JSF/primefaces-vs-richfaces-vs-icefaces.html

2012-01-11

44. ¿QUÉ ES RICHFACES?

http://www.scribd.com/doc/26352343/JBoss-RichFaces-Capitulo-1-%C2%BFQue-es-RichFaces

2012-02-04

45. RICHFACES

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/85

2012-02-04

46. RICHFACES

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/134

2012-02-04

47. RICHFACES DEVELOPER GUIDE

http://www.scribd.com/doc/36023226/Richfaces-Reference

2012-02-04

48. RICHFACES. ISSUES

https://issues.jboss.org/browse/RF#selectedTab=com.atlassian.jira.plugin.system.project%3Aissues-panel

2012-03-27

49. STRUCTS Y JAVASERVER FACES, CARA A CARA

http://www.ing.unp.edu.ar/wicc2007/trabajos/ISBD/109.pdf

2012-01-15

50. THE APACHE SOFTWARE FOUNDATION – MYFACES TRINIDAD. ISSUES

https://issues.apache.org/jira/browse/TRINIDAD#selectedTab=com.atlassian.jira.plugin.system.project%3Aissues-panel

2012-03-27

51. THE JAVA EE 5 TUTORIAL

http://download.oracle.com/javaee/5/tutorial/doc/gentextid-10788.html

2012-03-21

52. THE JSF MATRIX

http://www.jsfmatrix.net/

2012-03-01

53. TOBAGO

http://myfaces.apache.org/tobago/index.html

2012-02-05

54. TOBAGO

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/88

2012-02-12

55. TOMAHAWK

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/137

2012-02-5

56. TRABAJO DE FIN DE CARRERA – J2EE. MVC

http://openaccess.uoc.edu/webapps/o2/bitstream/10609/669/1/00848tfc.pdf

2012-01-17

57. TUTORIAL DE JAVA SERVER FACES

http://www.sicuma.uma.es/sicuma/Formacion/documentacion/JSF.pdf

2012-02-22

58. YOUR FIRTS CUP IAN EVANS

http://docs.oracle.com/javaee/6/firstcup/doc/gkhoy.html#gcrkk

2012-01-13

59. WELCOME TO THE APACHE MYFACES

https://cwiki.apache.org/confluence/display/MYFACES/Index

2012-02-04