UNIVERSIDAD NACIONAL DE CHIMBORAZO

FACULTAD DE INGENIERA

ESCUELA DE INGENIERÍA EN SISTEMAS Y COMPUTACIÓN

"Trabajo de grado previo a la obtención del

Título de Ingeniero en Sistemas y Computación"

ESTUDIO COMPARATIVO DE IMPLEMENTACIONES JSF: PRIMEFACES Y RICHFACES CON RESPECTO AL RENDIMIENTO APLICADO AL SISTEMA DE GESTIÓN DE TUTORÍAS DE LA FACULTAD DE INGENIERÍA DE LA UNACH.

AUTOR(AS):

Contento Dias Sandra Maricela

Guamán Siguenza Janneth Patricia

DIRECTOR: Ing. Diego Palacios

Riobamba – Ecuador

2015

Los miembros del Tribunal de Graduación del proyecto de investigación de título: **"ESTUDIO COMPARATIVO DE IMPLEMENTACIONES JSF: PRIMEFACES Y RICHFACES CON RESPECTO AL RENDIMIENTO APLICADO AL SISTEMA DE GESTIÓN DE TUTORÍAS DE LA FACULTAD DE INGENIERIA DE LA UNACH"** presentado por: Sandra Maricela Contento Dias y Janneth Patricia Guamán Siguenza, dirigida por el Ing. Diego Palacios.

Una vez escuchada la defensa oral y revisado el informe final del proyecto de investigación con fines de graduación escrito en la cual se ha constatado el cumplimiento de las observaciones realizadas, remite la presente para uso y custodia en la biblioteca de la Facultad de Ingeniería de la UNACH.

Para constancia de lo expuesto firman:

Ing. Danny Velasco
Presidente del Tribunal

Firma

Ing. Diego Palacios
Miembro del Tribunal

Firma

Ing. Lady Espinoza
Miembro del Tribunal

Firma

AUTORÍA DE LA INVESTIGACIÓN

"La responsabilidad del contenido de este Proyecto de Graduación, corresponde exclusivamente a: Sandra Maricela Contento Dias y Janneth Patricia Guamán Siguenza, autoras del proyecto de investigación, al Ing. Diego Palacios, Director de Tesis; y el patrimonio intelectual de la misma a la Universidad Nacional de Chimborazo.

Sandra Contento Dias
C.I. 060401789-7

Janneth Guamán Siguenza
C.I. 060405659-8

AGRADECIMIENTO

En el presente trabajo de investigación queremos agradecer a Dios por brindarnos la salud, la vida y por permitirnos hacer realidad nuestro anhelado sueño.

Expresamos nuestro agradecimiento a la **Universidad Nacional de Chimborazo**, la cual nos abrió sus puertas para culminar con éxito una etapa más de nuestras vidas, preparándonos para un futuro competitivo y poder servir a la sociedad con nuestros sólidos conocimientos para el progreso del país.

Un reconocimiento especial a nuestro Director de Tesis**, Ing. Diego Palacios Campana** por su calidad humana y todo el apoyo brindado al instruirnos y guiarnos a realizar el presente trabajo investigativo. Al **Ing. Carlos Padilla** por su paciencia y ayuda incondicional.

A los docentes de la Escuela de Ingeniería en Sistema y Computación porque todos han aportado con sus conocimientos y consejos para nuestra formación profesional.

Son muchas las personas que han formado parte de nuestra vida a las que nos encantaría agradecerles su amistad, consejos, apoyo, ánimo y compañía en los momentos más difíciles. Algunas están aquí con nosotros y otras en nuestros recuerdos y en nuestro corazón, sin importar en donde estén queremos darles las gracias por formar parte de nosotros, por todo lo que nos han brindado y por todas sus bendiciones.

Para ellos muchas gracias y que Dios les bendiga

Sandra y Janneth

DEDICATORIA

Doy gracias a Dios, por estar conmigo en cada paso que doy, por guiarme e iluminar mi mente y permitirme cumplir una meta más en mi vida.

A mi padre Wilmer por bridarme incondicionalmente su amor, consejos, ánimos para alcanzar mi más grandioso sueño el ser una profesional.

A mi mami Libia que desde el cielo me cuida y me protege y es mi inspiración para seguir adelante y cumplir mis metas.

A mis hermanas Maribel, Isabel, Daniela, Samantha a mi hermano Rolando, sobrinos, amigas y a Kleber Bustán por sus consejos constantes y su apoyo incondicional en los momentos más conflictivos y difíciles de mí vida.

A mi amiga Janneth Guamán por el apoyo incondicional en los momentos felices y tristes de mi vida

Y a todas las personas que han sido mi soporte y compañía durante toda mi formación académica.

Sandra Contento

DEDICATORIA

A Dios por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos, además de su infinita bondad y amor.

A mi papi Pedro que a pesar de nuestra distancia siento que estás conmigo en cada instante, sé que este momento es tan especial para ti como lo es para mí, a mi mami Rosita por su amor, cariño y apoyo incondicional me ha permitido ser una persona de bien.

A mis hermanas Rosita, Alexandra, Carolina, Fernanda y Gissela quienes son mi inspiración para ser mejor cada día.

A mi mami Carmen que desde el cielo iluminas cada paso de mi vida, tu quien me motivo a seguir a delante y a quien prometí terminar mis estudios.

Promesa cumplida abue!!!

A mi amiga Sandra Contento por su apoyo incondicional, porque sin el equipo que formamos no hubiéramos logrado esta meta.

A Oscar por ser alguien muy especial en mi vida y por demostrarme que en todo momento cuento contigo.

A todos los que me apoyaron para escribir y concluir esta tesis.

Janneth Guamán

ÍNDICE GENERAL

INDICE DE ILUSTRACIONES

INDICE DE TABLAS

RESUMEN

El presente trabajo tiene como objetivo investigar las implementaciones JSF: PrimeFaces y RichFaces con respecto al rendimiento para el desarrollo del Sistema de Gestión de Tutorías SIGET.

Estas implementaciones se evaluaron de acuerdo a los parámetros de facilidad de uso, facilidad para iniciar, diversidad de componentes, compatibilidad con los navegadores, documentación y rendimiento, obteniendo como resultado que la librería de componentes PrimeFaces obtuvo un valor de 67%, siendo la mejor opción para el desarrollo de la aplicación SIGET, mientras que la librería de componentes RichFaces obtuvo un valor de 33%.

En el desarrollo de la aplicación SIGET se utilizó Postgresql 9.3 como motor de Base de Datos, Netbeans 8.0 como IDE, GlassFish 4.0 como servidor Web, Java Server Faces 2.0, PrimeFaces 4.0, Ireports 3.6 para la generación de reportes.

Se concluye que al desarrollar el Sistema de Gestión de Tutorías para la Facultad de Ingeniería de la Universidad Nacional de Chimborazo la información sobre las actividades de tutorías que realiza el docente tendrá una organización adecuada y transparente. Además el desarrollo de esta aplicación es un indicador fundamental en el proceso de evaluación de desempeño de carreras según el Consejo de Evaluación, Acreditación y Aseguramiento de la Calidad de la Educación Superior CEAACES.

UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CENTRO DE IDIOMAS

Lic. Luis Guadalupe **26 de Mayo del 2015**

SUMMARY

The present research has as an aim to investigate the implementations JSF: PrimeFaces and RichFaces regarding the performance for the development of the Management System Tutorial SIGET.

These implementations were evaluated according to the parameters of: ease of use, ease of starting, diversity of components, browser compatibility, documentation and performance, resulting in the library PrimeFaces components obtained a value of 67%, the best option for the development of SIGET application while the library RichFaces components which obtained a value of 33%.

In developing the application SIGET, Postgres 9.3 a Database Engine was used as a date base motor, NetBeans 8.0 as IDE, GlassFish 4.0 as Web server, Java Server Faces 2.0, 4.0 PrimeFaces, iReports 3.6 for the report generation.

It is concluded that in developing the tutorials Management System for the Faculty of Engineering of the National University of Chimborazo, about the information on activities performed by the teacher mentoring will have a proper and transparent organization. Besides that the development of this application is a key indicator in the performance appraisal process according to the Council racing Assessment, Accreditation and Quality Assurance of Higher Education CEAACES.

CENTRO DE IDIOMAS
FACULTAD DE INGENIERIA
UNIVERSIDAD NACIONAL DE CHIMBORAZO
RIOBAMBA-ECUADOR
COORDINACION

INTRODUCCIÓN

Hoy en día la presencia de aplicaciones web tanto para empresas, entidades e instituciones que suministran productos y servicios, es una necesidad de primer nivel evitar ser desplazados por la competencia y ofrecer mejores y eficaces servicios a sus clientes.

Tradicionalmente, las aplicaciones web se han codificado mediante páginas JSP las cuales reciben peticiones a través de formularios y se construían como respuesta páginas HTML a través de bibliotecas de etiquetas de código Java. Java Server Faces facilita la construcción de estas aplicaciones proporcionando para esto un entorno de trabajo Framework a través de la web la cual gestiona las acciones realizadas por el usuario en su página HTML y las traduce a eventos que son enviados al servidor regenerando la página original y reflejando los cambios provocados por dichas acciones.

La presente investigación se encamina a estudiar y determinar cuál es el librería de mejor rendimiento para el desarrollo de aplicaciones web ricas en interfaz de usuario, estableciendo parámetros de comparación los mismos que serán analizados y probados en diferentes prototipos lo que permitirá el cumplimiento de los objetivos planteados.

CAPITULO I

PROBLEMATIZACIÓN

1.1. Identificación y descripción del problema

Con el paso del tiempo, la tecnología avanza continuamente, y se ven aplicaciones web donde sus interfaces son agradables, pues para quienes desarrollan son largos tiempos de programación de millones de líneas de código. Además, cuando se tiene varias capas o tablas anidadas existe la posibilidad de que no se vea igual en todos los exploradores.

Java Server Faces es un framework para el desarrollo web basado en tecnología Java y el patrón Modelo Vista Controlador, que usa librerías de componentes para que este desarrollo sea más ágil y rápido.

PrimeFaces es un conjunto de librerías de componentes visuales OpenSource sin dependencias ni configuraciones, mientras que RichFaces permite crear componentes Rich que soportan Ajax para el desarrollo de aplicaciones web que al utilizar optimizan el rendimiento y los recursos dinámicos utilizados.

Actualmente la Facultad de Ingeniería de la Universidad Nacional de Chimborazo posee información que es registrada en libros de Excel y administrada inadecuadamente de forma manual acerca de las horas dictadas de tutorías, con lo que se genera información no apropiada para quien requiera aumentando el desconcierto y la falta de transparencia en cierta información, además conlleva un trabajo laborioso y tedioso el cual requiere mucho tiempo.

Esta información se convierte en una herramienta estratégica y en un activo de gran valor, por lo que nace la necesidad de realizar el estudio comparativo entre las implementaciones de JSF, PrimeFaces y RichFaces con respecto al rendimiento para desarrollar el Sistema de Gestión de Tutorías de la Facultad de Ingeniería de la UNACH.

1.2. Análisis crítico

La creación de una aplicación web conlleva mucho tiempo de programación en sus diferentes componentes por lo que al desarrollador le resulta un trabajo tedioso, y no es la mejor opción para el desarrollo de la AppWeb.

Además, por la variedad de herramientas que actualmente existe se puede disminuir tiempos de programación y facilitar la reutilización de código para estas aplicaciones.

1.3. Prognosis

La implementación PrimeFace o RichFace con respecto al rendimiento facilitará el desarrollo del Sistema de Gestión de Tutorías de la Facultad de Ingeniería de la UNACH ya que disminuirán el tiempo de programación y permitirá la reutilización de código.

Además se almacenará, manejará y visualizará las interfaces de la aplicación de una manera agradable, sumado a una adecuada generación de estadísticas y reportes docentes-estudiantes.

1.4. Justificación

Según el Decreto Ejecutivo No. 1014 emitido el 10 de abril de 2008, se dispone el uso de Software Libre en los sistemas y equipamientos informáticos de la Administración Pública de Ecuador, por lo que la gama de herramientas OpenSource útiles para el desarrollo de aplicaciones web, facilitan de forma inmediata la creación de estas de una manera más rápida, minimizando el tiempo de desarrollo de un sistema web.

Java Server Faces es un lenguaje orientado a la creación de componentes visuales para el desarrollo de sistemas web basado en componentes, que describen, construyen y utilizan técnicas para la creación de sistemas abiertos y distribuidos mediante la reutilización de líneas de código, permitiendo así reducir costes, tiempo y esfuerzos de desarrollo de software.

Mediante el estudio comparativo entre PrimeFaces y RichFaces se logrará determinar cuál de estas implementaciones JSF permitirán obtener un máximo rendimiento en el desarrollo de una aplicación web.

Además, la Universidad Nacional de Chimborazo se encuentra en un proceso de evaluación de desempeño de carreras que deben cumplir indicadores de acreditación según el Consejo de Evaluación, Acreditación y Aseguramiento de la Calidad de la Educación Superior CEAACES, razón por la cual es necesario la implementación del Sistema de Gestión de Tutorías de la Facultad de Ingeniería de la UNACH que permita automatizar y gestionar la información del docente-estudiante de una manera adecuada y dé como resultado estadísticas claras, reportes y datos confiables.

1.5. Delimitación

El estudio de la investigación se basará en la comparación de las implementaciones JSF: PrimeFaces y RichFaces, con respecto al rendimiento de las librerías de componentes para seleccionar la mejor opción para el desarrollo del sistema de tutorías a desarrollarse directamente en la Facultad de Ingeniería de la Universidad Nacional de Chimborazo, que se encuentra ubicada en la Avenida Antonio José de Sucre, Km 1 ½ vía a Guano en la ciudad de Riobamba provincia de Chimborazo.

Este sistema se basará en automatizar principalmente los procesos de tutorías de docentes para obtener una administración eficiente de tareas, trabajos de estudiantes permitiendo así a las autoridades de la facultad dar un seguimiento óptimo a las actividades desarrolladas en el transcurso del semestre académico.

1.6. Formulación del problema

¿Cómo incide las implementaciones JSF: Primefaces y Richfaces con respecto al rendimiento para el desarrollo del Sistema de Gestión de Tutorías de la Facultad de Ingeniería de la UNACH?

1.7. Objetivos

1.7.1. General

Realizar un estudio comparativo de implementaciones JSF: Primefaces y Richfaces con respecto al rendimiento aplicado al Sistema de Gestión de Tutorías de la Facultad de Ingeniería de la UNACH.

1.7.2. Específicos

- Realizar el análisis de las librerías de componentes PrimeFace y RichFace.
- Seleccionar y aplicar los indicadores de rendimiento para ser evaluadas.
- Diseñar e implementar una aplicación web para el Sistema de Gestión de Tutorías de la Facultad de Ingeniería de la UNACH utilizando la implementación JSF seleccionada.

1.8. Hipótesis

El estudio comparativo de las implementaciones JSF Primefaces y JSF Richfaces, con respecto al rendimiento, determinará la mejor opción para el desarrollo del Sistema de Gestión de Tutorías de la UNACH.

CAPITULO II

FUNTAMENTACIÓN TEÓRICA

En la actualidad existen diferentes lenguajes de Programación para el desarrollo de aplicaciones web estos han ido surgiendo debido a las tendencias y necesidades de las plataformas, entre ellos se encuentra el lenguaje de programación JAVA.

2.1. LENGUAJE DE PROGRAMACIÓN JAVA[1]

Ilustración 1 Logo de Java
Fuente: https://www.java.com/es/about/

Java es un lenguaje de programación y una plataforma informática comercializada por primera vez en 1995 por Sun Microsystems. Se popularizó a partir del lanzamiento de su primera versión comercial de amplia difusión, la JDK 1.0 en 1996. Java es rápido, seguro y fiable.

2.1.1. Características

En http://java.sun.com se puede leer que Java es: "Un lenguaje simple, orientado al objeto, distribuido, interpretado, sólido, seguro, de arquitectura neutral, portable, de alto desempeño, de multihilos y dinámico".

1 https://www.java.com/es/download/whatis_java.jsp

Tabla 1: Características Java

Simple	Basado en el lenguaje C++
Orientado al objeto	Java da buen soporte a las técnicas de desarrollo Programación Orientada a Objetos (POO) y a la reutilización de componentes de software.
Distribuido	Java se ha diseñado para trabajar en ambiente de redes y contienen una gran biblioteca de clases para la utilización del protocolo TCP/IP, incluyendo HTTP y FTP. El código Java se puede manipular a través de recursos URL.
Interpretado	El compilador Java traduce cada fichero fuente de clases a código de bytes (Bytecode), que puede ser interpretado por todas las máquinas que den soporte a un visualizador que funcione con Java.
Sólido	El código Java no se quiebra fácilmente ante errores de programación.
Seguro	Java evita la manipulación de código. Actualmente se está trabajando en la encriptación del código.
De arquitectura neutral	El compilador crea códigos de byte (Bytecode) que se envía al visualizador solicitado y se interpreta en la máquina que posee un intérprete de Java o dispone de un visualizador que funciona con Java.
Portable	Al ser de arquitectura neutral es altamente portable.
Alto Desempeño	El compilador Java suele ofrecer la posibilidad de compilar Bytecode en código máquina de determinadas plataformas.
Multihilos	Java puede aplicarse a la realización de aplicaciones en las que ocurra más de una cosa a la vez.
Dinámico	Java utiliza un sistema de interfaces que permite aligerar esta dependencia. Como resultado, los programas Java pueden permitir nuevos métodos y variables en un objeto de biblioteca sin afectar a los objetos dependientes.

Fuente: http://java.sun.com

2.1.2. Beneficios de Java[2]

Java se ha convertido en un valor impagable para los desarrolladores, ya que les permite:

- Escribir software en una plataforma y ejecutarla virtualmente en otra
- Crear programas que se puedan ejecutar en un explorador y acceder a servicios Web disponibles
- Desarrollar aplicaciones de servidor para foros en línea, almacenes, encuestas, procesamiento de formularios HTML y mucho más
- Combinar aplicaciones o servicios que utilizan el lenguaje Java para crear aplicaciones o servicios con un gran nivel de personalización
- Escribir aplicaciones potentes y eficaces para teléfonos móviles, procesadores remotos, microcontroladores, módulos inalámbricos, sensores, gateways, productos de consumo y prácticamente cualquier otro dispositivo electrónico

2.1.3. Tecnologías JAVA

Las tres tecnologías de la plataforma Java facilita el proceso para que desarrolladores de software, prestadores de servicios y fabricantes de dispositivos enfoquen mercados específicos:

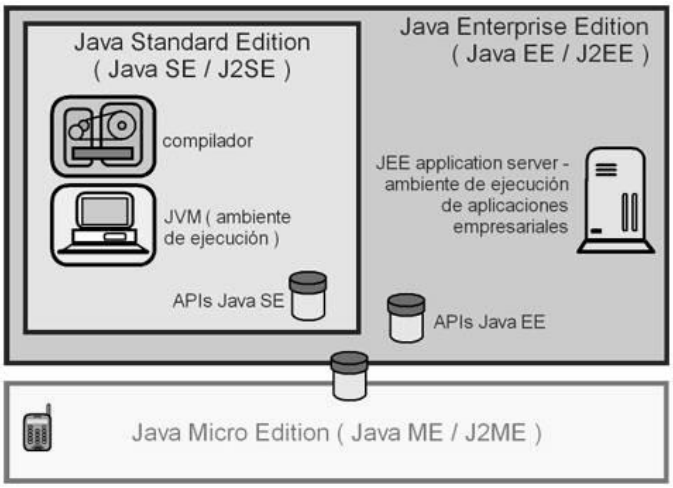


Ilustración 2 Tecnologías JAVA
Fuente: http://www3.uji.es/~belfern/pdf/libroJavaConTapa.pdf

2 https://www.java.com/es/about/

2.1.3.1. Java J2SE (Plataforma Java, Standard Edition)[3]

Ilustración 3 JAVA J2EE
Fuente: http://auladirectiva.com/java-j2se-j2ee/

J2SE permite desarrollar y desplegar aplicaciones Java en desktops y servidores, como también en entornos incorporados y en tiempo real. J2SE incluye clases que soportan el desarrollo de servicios web Java y proporciona la base para la Plataforma Java, Enterprise Edition (Java EE). Java SE 6 ("Mustang") es la versión actual de la plataforma Java SE. Muchos desarrolladores Java usan Java SE 5, también conocido como Java 5.0 o "Tiger.

2.1.3.2. Java J2ME (Plataforma Java, Micro Edition)[4]

Ilustración 4 JAVA J2ME
Fuente:http://bit.ly/1FaEJuN

J2ME proporciona un entorno para aplicaciones que operan en una gama amplia de dispositivos móviles e incorporados, como teléfonos móviles, PDAs e impresoras. La plataforma Java ME incluye interfaces de usuario flexibles, un modelo robusto de seguridad, una gama amplia de protocolos de red incorporados y amplio soporte para aplicaciones conectadas en red y offline que pueden ser descargadas dinámicamente. Las aplicaciones basadas en las especificaciones de Java ME se escriben una única vez para una gama amplia de dispositivos, pero aprovechan las posibilidades nativas de cada dispositivo.

[3] http://www.oracle.com/technetwork/java/javase/overview/index.html
[4] http://www.oracle.com/technetwork/java/embedded/javame/index.html

2.1.3.3. Java J2EE - JEE (Plataforma Java, Enterprise Edition)[5]

Ilustración 5 JAVA J2EE
Fuente: http://www.biosoft.cl/img/java.png

J2EE es una plataforma de programación para desarrollar y ejecutar software de aplicaciones con arquitecturas de múltiples capas distribuidas. Se basa ampliamente en componentes modulares de software ejecutándose sobre un servidor de aplicaciones. Basado en Java SE, J2EE proporciona APIs (Application Programming Interface) de comunicaciones, servicios web, modelos de componentes y gestión para implementar aplicaciones Web 2.0 de nivel empresarial.

Características

- Plataforma abierta y estándar
- Modelo de aplicaciones distribuidas
- Componente modulares y estandarizados

Elementos

- Componentes(cliente, web y negocio)
- Contenedores(aplicaciones, applets, web)
- Servicios(configurables y no configurables)
- Servidores de aplicaciones

Para el desarrollo de las aplicaciones J2EE se han ideado varias especificaciones que facilitan el desarrollo de estas aplicaciones web como hibernate, Struts, Java Server Pages, Java Server Faces.

2.2. TECNOLOGIA JQUERY

jQuery es una librería JavaScript open-source, que funciona en múltiples navegadores, y que es compatible con las hojas de estilo en cascada 3(CSS3). Su objetivo principal es hacer la programación "scripting" mucho más fácil y rápida

[5] http://www.oracle.com/technetwork/java/embedded/javame/index.html

del lado del cliente. Con jQuery se pueden producir páginas dinámicas así como animaciones parecidas a Flash en relativamente corto tiempo.

2.2.1. Antecedentes JQuery [6]

JQuery fue publicado por primera vez en Enero del 2006 en "BarCamp NYC" por John Resign. Soporte para AJAX fue agregado un mes después, y el modelo de licenciamientos open source del MIT (Massachusetts Institute of Technology) fue adoptado en Mayo de ese mismo año.

JQuery es una biblioteca gratuita JavaScript rápido, pequeño y rico en funciones con una combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma en que millones de personas escriben JavaScript. Su objetivo principal es simplificar las tareas de creación de páginas web responsivas, acordes a lo estipulado en la Web 2.0, la cual funciona en todos los navegadores modernos. Por otro lado, se dice que jQuery ayuda a que los desarrolladores se centren en el diseño del sitio, al abstraer por completo todas las características específicas de cada uno de los navegadores. Otra de las grandes ventajas de jQuery es que se enfoca en simplificar los scripts y en acceder y modificar el contenido de una página web. Finalmente, jQuery agrega una cantidad impresionante de efectos nuevos a Javascript, los cuales podrán ser utilizados en sitios Web.

2.2.2. Características

- Selección de elementos DOM (Modelo de Objetos del Documento).
- Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3 y un plugin básico de XPath.
- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- Animaciones personalizadas.
- AJAX.
- Soporta extensiones
- Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes.

[6] http://w3techs.com/technologies/overview/javascript_library/all

- Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+. [7]

2.2.3. Módulos

La biblioteca jQuery se compone de un único archivo JavaScript y se divide en cuatro módulos:

a) Núcleo.- Contiene las funciones básicas para el resto de módulos.

b) Interacciones.- Añade comportamientos complejos a los elementos:

- Draggable: Hace al elemento arrastrable.
- Droppable: Permite que el elemento responda a elementos arrastrables.
- Resizable: Permite redimensionar el elemento.
- Selectable: Permite seleccionar entre una lista de elementos.
- Sortable: Ordena una lista de elementos.

c) Widgets.- Es un conjunto completo de controles UI. Cada control tiene un conjunto de opciones configurables y se le puede aplicar estilos CSS.

- Accordion: Menú con efecto acordeón.
- Dialog: Ventanas con contenido.
- Slider: Elemento para elegir en un rango de valores.
- Datepicker: Calendario gráfico.
- Progressbar: Barra de progreso.

d) Efectos.- Una API para añadir transiciones animadas y facilidades para interacciones.

2.2.4. Ventajas y desventajas[8]

Tabla 2: **Tecnología jQuery-Ventajas y desventajas**

VENTAJAS	DESVENTAJAS
• jQuery es flexible y rápido para el desarrollo web • Licencia MIT y es Open Source • Excelente comunidad de soporte • Excelente integración con AJAX	• Gran cantidad de versiones publicadas en el corto tiempo • Actualizaciones constantes que pueden traer incompatibilidad con el código

Fuente: http://grupoajax.blogspot.com/2013/05/ventajas-y-desventajas-de-ajax.html

[7] http://jquery.com/browser-support/

[8] http://computacionytecnologia.com/esencia-y-ventajas-del-uso-de-jquery/

2.3. TECNOLOGIA AJAX

2.3.1. Antecedentes Tecnología AJAX[9]

A pesar de que el término "AJAX" fuese creado en 2005, la historia de las tecnologías que permiten AJAX se remonta a una década antes con la iniciativa de Microsoft en el desarrollo de Scripting Remoto. Sin embargo, las técnicas para la carga asíncrona de contenidos en una página existente desde la introducción del elemento iframe en Internet Explorer 3 en 1996 y el tipo de elemento layer en Netscape 4 en 1997, abandonado durante las primeras etapas de desarrollo de Mozilla. Ambos tipos de elemento tenían el atributo src que podía tomar cualquier dirección URL externa, y cargando una página que contenga javascript que manipule la página paterna, pudiendo lograrse efectos parecidos al AJAX.

2.3.2. Características[10]

AJAX es un acrónimo para Asynchronous JavaScript And XML (JavaScript Asíncrono y XML). Permite la creación de aplicaciones interactivas en el desarrollo web.

Es decir, con AJAX se crean sitios web que se ejecuten directamente en el navegador del usuario manteniendo una comunicación con el servidor siempre que sea necesario pero sin recargar la página que se visualiza, simplemente se realizarán cambios sobre ella.

Esto significa que la velocidad de interacción con la aplicación aumenta de forma significativa al reducir el número de peticiones que se hacen al servidor. Y aún así, la comunicación que se realiza entre el navegador y el servidor se realiza de forma asíncrona y en segundo plano, por lo que es completamente transparente para el usuario.

AJAX no constituye una tecnología en sí, sino que combina tres tecnologías ya existentes:

- XHTML y hojas de estilos (CSS) para el diseño que formatea la información.

[9] http://librosweb.es/libro/ajax/capitulo_1.html
[10] http://www.aulaclic.es/articulos/web22.html

- Document Object Model (DOM) que es el encargado de interactuar con la información presentada y es el que se ejecuta en el cliente (navegador), y
- XMLHttpRequest, que es un objeto encargado de intercambiar datos con el servidor web. Estos datos son devueltos en formato XML.

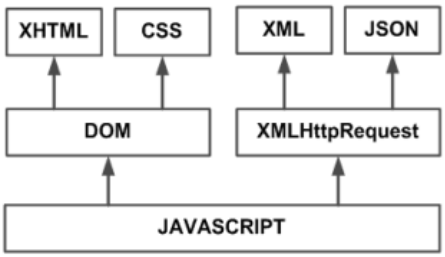


Ilustración 6 Tecnologías Agrupadas bajo el concepto de AJAX
Fuente: http://librosweb.es/libro/ajax/capitulo_1.html

2.3.3. Motor Ajax[11]

El motor AJAX (AJAX engine) colocado entre el usuario y el servidor web evita el ciclo start-stop-start-stop característico de las aplicaciones web tradicionales y no es más que un fichero JavaScript que acompaña al HTML. El cual es cargado al inicio de la sesión y tiene una doble responsabilidad, primero generar la interfaz visualizada por el usuario y segundo comunicarse con el servidor en representación del usuario, lo cual ocurre de manera asíncrona evitando que el usuario vea una página blanca o el reloj de arena (de espera) cada vez que realice una acción.

En aplicaciones AJAX se envían peticiones vía http(s) mediante eventos, scripts o rutinas al servidor Web, para obtener únicamente la información necesaria, empleando SOAP o algún otro lenguaje para servicios Web basado en XML, y usando JavaScript en el cliente para procesar la respuesta del servidor Web. Esto redunda en una mayor interacción gracias a la reducción de información intercambiada entre servidor y cliente, y a que parte del proceso de la información se hace en el propio cliente, liberando al servidor de ese trabajo. Además esta petición se realiza como proceso de fondo (background), por lo que el usuario no tiene que esperar que el proceso concluya en su totalidad para continuar

[11] http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications/

interactuando con la aplicación. La contrapartida es que la descarga inicial de la página es más lenta al tenerse que bajar todo el código JavaScript.

En el siguiente gráfico se puede ver la diferencia entre utilizar un modelo clásico de aplicación Web y utilizar el modelo de aplicación Web que propone AJAX:

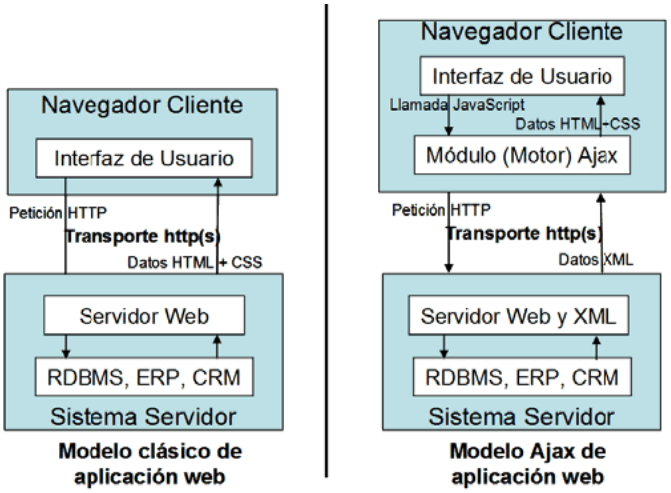


Ilustración 7 Modelo clásico de aplicación web y modelo AJAX de aplicación web
Fuente: http://bit.ly/1H7uWWF

2.3.4. Ventajas y desventajas

Tabla 3: Tecnología AJAX-Ventajas y Desventajas

VENTAJAS	DESVENTAJAS
• Utiliza tecnologías existentes. • Soportada por la mayoría de los navegadores modernos. • Interactividad.- El usuario no tiene que esperar hasta que lleguen los datos del servidor. • Portabilidad.- no requiere plug-in como Flash y Apple de Java • Mayor velocidad, esto debido que no hay que retornar toda la página nuevamente. • La página se asemeja a una aplicación de escritorio.	• Se pierde el concepto de volver a la página anterior. • La existencia de páginas con AJAX y otras sin esta tecnología hace que el usuario se desoriente. • Problemas con navegadores antiguos que no implementan esta tecnología. • No funciona si el usuario tiene desactivado el JavaScript en su navegador. • Requiere programadores que conozcan todas las tecnologías que intervienen en AJAX. • Dependiendo de la carga del servidor podemos experimentar tiempos tardíos de respuesta que desconciertan al visitante.

Fuente: http://sherekan.com.ar/blog/2008/04/19/introduccion-a-ajax/

2.4. ESPECIFICACIÓN JAVA SERVER FACES (JSF) (Loor, 2014)

La tecnología JSF es un framework de interfaz de componentes de usuarios del lado del servidor para las aplicaciones web basadas en la tecnología Java y en el patrón MCV (Modelo Vista Controlador).

Java Server Faces fue creado a través del Java Community Process (JCP) por un grupo de líderes de la tecnología, incluyendo Sun Microsystems, Oracle, Borland, BEA, IBM.

JSF pretende normalizar y estandarizar el desarrollo de aplicaciones web. La interfaz de usuario se crea con la tecnología JSF, se ejecuta en el servidor y se renderiza en el cliente.

JSF combina un enfoque de diseño MVC con un potente interfaz de usuario, basada en un marco de componentes de desarrollo que simplifica en gran medida el desarrollo Java EE Web durante el uso tecnologías de marcado y servlets existentes.

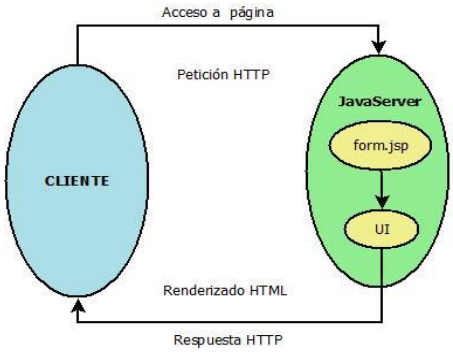


Ilustración 8 Diagrama de una aplicación JSF
Fuente: Sandra Contento/Janneth Guamán

2.4.1. Patrón Modelo Vista Controlador (MVC)

El patrón MVC está dirigido 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 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. 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 JSF para mostrar la interfaz del usuario que representaría la Vista, y finalmente los famosos JavaBeans ubicados en el modelo. (Fowler, 2009)

Este patrón proporciona una clara separación entre las distintas responsabilidades de la aplicación web.

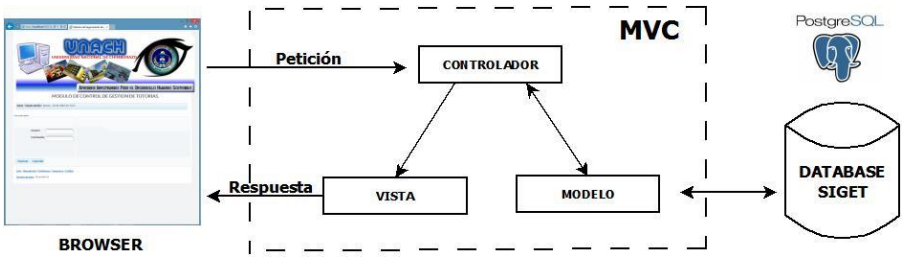


Ilustración 9 Modelo Vista Controlador
Fuente: Sandra Contento/Janneth Guamán

- **Controlador.-** 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.
- **Vista.-** 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, y ahí es donde entrarán los JSF.
- **Modelo.-** Encapsula la lógica de negocio de la aplicación, acceso a los datos y su manipulación. (YiiFramework, 2011)

2.4.2. Objetivos de diseño de Java Server Faces (Clavijo, 2010)

Según JSR # 127 especifica ocho requisitos de diseño para JSF. Estos objetivos describen un enfoque de diseño para JSF, hasta e incluyendo JSF 2.0.

1. Crear un marco de componentes de interfaz de usuario estándar que puede ser aprovechado por el desarrollo de herramientas para facilitar a los desarrolladores crear interfaces de usuario de alta calidad y gestionar las conexiones de interfaz de usuario para el comportamiento de la aplicación.

2. Definir un conjunto de clases bases Java simples y ligeras para los componentes de interfaz de usuario, componentes de estado y de entrada de eventos.
3. Proporcionar un conjunto de componentes comunes de interfaz de usuario, incluyendo los elementos estándar de entrada de un formulario HTML. Estos componentes se derivan del simple conjunto de clases base que se puede utilizar para definir nuevos componentes.
4. Proporcionar un modelo de JavaBeans para el envío de eventos de controles de interfaz de usuario del lado del cliente para el servidor del lado del comportamiento de la aplicación.
5. Definir APIs para la validación de entrada, incluido el apoyo para la validación del lado del cliente.
6. Especifica un modelo para la internacionalización y localización de la interfaz de usuario.
7. Proveer para la generación automática de salida apropiado para el cliente de destino, teniendo en cuenta todos los datos de configuración del cliente disponibles, como la versión del navegador.
8. Proporcionar para la generación automática de salida que contiene ganchos necesarios para apoyar la accesibilidad, según la definición de la Web Accessibility Initiative (WAI).

2.4.3. Componentes de JSF (University of Vigo, 2008)

JSF introduce 2 términos al mundo del desarrollo de aplicaciones para JAVA:

- Managed Bean
- Backin Bean

2.4.3.1. Managed Bean

Un Managed Bead también llamado controlador es un objeto identificado para el ambiente de la aplicación, para la cual se describe:

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

```
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ManagedProperty;
import javax.faces.bean.ViewScoped;

@ManagedBean
@ViewScoped

public class FacultadControlador {
```

Ilustración 10 Controlador - Managed Bean
Fuente: Sandra Contento/Janneth Guamán

2.4.3.2. Backing Bean

Un Backing Bean es usualmente un Bean común de java que sirve de soporte para un objeto manejado dentro de la aplicación.

La ventaja de los Backing 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.

2.4.3.3. Ámbitos de los beans *(King, 2009)*

Para comodidad del programador aplicaciones web, un contenedor de servlets suministra diferentes ámbitos, de petición, de sesión y de aplicación. Estos ámbitos normalmente mantienen beans y otros objetos que necesitan estar disponibles en diferentes componentes de una aplicación web.

Ámbito de tipo petición.- Es el de vida más corta. Empieza cuando una petición HTTP comienza a tramitarse y acaba cuando la respuesta se envía al cliente.

Ámbito de tipo sesión.- El navegador envía una petición al servidor, el servidor devuelve una respuesta, y entonces ni el navegador ni el servidor tiene cualquier obligación para conservar cualquier memoria de la transacción. Este acomodamiento simple marcha bien para recuperar información básica, pero es poco satisfactorio para aplicaciones del lado del servidor.

Ámbito de tipo aplicación.- Persiste durante toda la aplicación web. Este ámbito es compartido entre todas las peticiones y sesiones.

Existe la posibilidad de anidar beans, para conseguir objetivos más complicados. Cuando junte ámbitos de beans, hay que asegurarse de que sean compatibles, tal y como se muestra en el siguiente cuadro:

Tabla 4: Compatibilidad de ámbitos de beans

Cuando se defina un bean de ámbito	Puede usar otro ámbito de tipo
none	none
application	none, apllication
sesión	none, apllication, session
request	none, apllication, sesión, request

Fuente: http://www.sicuma.uma.es/export/sites/sicuma/es/formacion/descargas/JSF.pdf

2.4.4. Ciclo de vida de una petición JSF[12]

El ciclo de vida de una página Java Server Faces page es similar al de una página JSP. El cliente hace una petición HTTP de la página y el servidor responde con la página traducida a HTML.

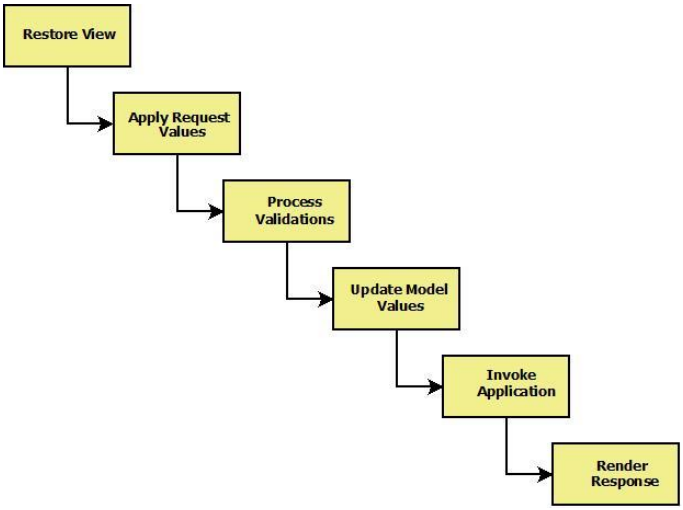


Ilustración 11 Ciclo de vida de Java Server Faces
Fuente: Sandra Contento/Janneth Guamán

El ciclo de vida completo es:

RESTORE VIEW (Fase Restaurar vista) –. En esta fase se crea el árbol de componentes. Se puede crear a partir de información existente o de cero si no hay información. Si no se encuentran datos POST o query string, se pasa directamente a producir respuesta.

APPLY REQUEST VALUES (Fase Aplicar valores de la petición). Se itera sobre los componentes del árbol, comprobando qué valor de la petición pertenece a qué componente, y los van guardando. Estos valores almacenados se llaman valores locales.

[12] http://www.oracle.com/technetwork/topics/index-090910.html

PROCESS VALIDATION (Fase Procesar validaciones). Se realizan las validaciones y conversiones necesarias de los valores locales. Si ocurre algún error en esta fase, se pasa a la fase Render Response, mostrándole al usuario otra vez la página actual y dándole así una nueva oportunidad para que pueda introducir los datos correctos.

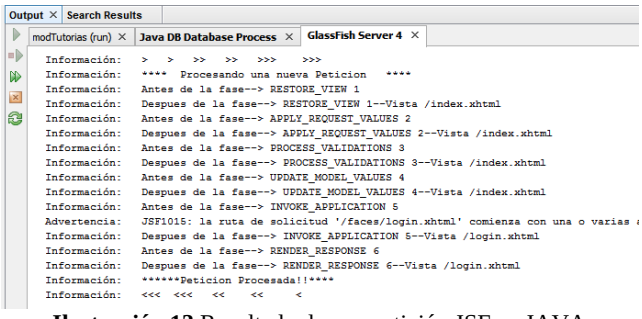
UPDATE MODEL VALUES (Fase Actualizar modelo). Se modifican los valores de los beans asociados a los componentes de la vista con los valores locales.

INVOKE APPLICATION (Fase Invocar aplicación). Se invoca el método asociado al action del botón o link y se llevan a cabo las operaciones correspondientes a las acciones del usuario.

RENDER RESPONSE (Fase Producir respuesta). El servidor devuelve la página de respuesta al navegador del usuario y guarda el estado actual de la vista para poder restaurarla en una petición posterior.

```
package recursos;

import javax.faces.event.*;
public class MyPhaseListener implements PhaseListener
{

   private static final long serialVersionUID = 1L;

   public MyPhaseListener()
   {
   }

   public void afterPhase(PhaseEvent event)
   {
      System.out.println((new StringBuilder("   Despues de la fase--> ")).append(event.getPhaseId().toString()).append("--Vista ").
      if(event.getPhaseId() == PhaseId.RENDER_RESPONSE)
      {
         System.out.println("******Peticion Procesada!!*****");
         System.out.println("<<<  <<<   <<    <<     <");
         System.out.println(("));
      }
   }

   public void beforePhase(PhaseEvent event)
   {
      if(event.getPhaseId() == PhaseId.RESTORE_VIEW)
      {
            System.out.println(("));
            System.out.println((" >   >   >>   >>   >>>    >>>"));
         System.out.println(("****  Procesando una nueva Peticion   ****"));
      }
      System.out.println((new StringBuilder("   Antes de la fase--> ")).append(event.getPhaseId().toString()).toString());
   }

   public PhaseId getPhaseId()
   {
      return PhaseId.ANY_PHASE;
   }

}
```

Ilustración 12 Listener del Ciclo de vida de una petición JSF en JAVA
Fuente: Sandra Contento/Janneth Guamán



```
Output ×  Search Results
modTutorias (run) ×  Java DB Database Process ×  GlassFish Server 4 ×
Información:   >   >   >>   >>   >>>    >>>
Información:   ****  Procesando una nueva Peticion   ****
Información:   Antes de la fase--> RESTORE_VIEW 1
Información:   Despues de la fase--> RESTORE_VIEW 1--Vista /index.xhtml
Información:   Antes de la fase--> APPLY_REQUEST_VALUES 2
Información:   Despues de la fase--> APPLY_REQUEST_VALUES 2--Vista /index.xhtml
Información:   Antes de la fase--> PROCESS_VALIDATIONS 3
Información:   Despues de la fase--> PROCESS_VALIDATIONS 3--Vista /index.xhtml
Información:   Antes de la fase--> UPDATE_MODEL_VALUES 4
Información:   Despues de la fase--> UPDATE_MODEL_VALUES 4--Vista /index.xhtml
Información:   Antes de la fase--> INVOKE_APPLICATION 5
Advertencia:   JSF1015: la ruta de solicitud '/faces/login.xhtml' comienza con una o varias
Información:   Despues de la fase--> INVOKE_APPLICATION 5--Vista /login.xhtml
Información:   Antes de la fase--> RENDER_RESPONSE 6
Información:   Despues de la fase--> RENDER_RESPONSE 6--Vista /login.xhtml
Información:   ******Peticion Procesada!!****
Información:   <<<  <<<   <<    <<     <
```

Ilustración 13 Resultado de una petición JSF en JAVA
Fuente: Sandra Contento/Janneth Guamán

2.4.5. Aplicación Java Server Faces (SicUma, 2013)

Las aplicaciones Java Server Faces son como cualquier otra aplicación web Java. Se ejecutan en un contenedor Servlets de Java y contienen:

- Componentes JavaBeans (llamados objetos del modelo en tecnología Java Server Faces) conteniendo datos y funcionalidades específicas de la aplicación.
- Oyentes de Eventos.
- Páginas, (principalmente páginas JSP).
- Clases de utilidad del lado del servidor, como controladores para acceder a las bases de datos.

Además de estos ítems, una aplicación Java Server Faces proporciona una arquitectura de componentes rica y flexible que incluye:

- Una librería de etiquetas personalizadas para dibujar componentes UI (interfaz de usuario) en una página.
- Una librería de etiquetas personalizadas para representar manejadores de eventos, validadores y otras acciones.
- Componentes UI representados como objetos con estado en el servidor.

2.4.5.1. Facelets en JSF *(Hookom, 2005)*

Facelets es un lenguaje de declaración de páginas, poderoso pero ligero, que es usado para construir vistas de Java Server Faces usando plantillas de estilo de HTML y construyendo arboles de componentes. Las características que Facelets incluyen son algunas de las siguientes:

- Uso de XHTML para crear páginas web
- Soporte para librerías de etiquetas Facelets que se suman a las librerías de Java Server Faces y JSTL
- Soporte para el Lenguaje de Expresiones (EL por su siglas en ingles)
- Plantillas para componentes y páginas

Las ventajas que Facelets incluye para el desarrollo de proyectos de gran escala son las siguientes:[13]

- Soporte para reutilización de código por medio de plantillas y componentes compuestos.

[13] http://docs.oracle.com/javaee/6/tutorial/doc/gijtu.html

- Extensibilidad funcional de componentes y otros objetos del lado del server por configuración.
- Tiempo de compilación rápido.
- Validación de EL en tiempo de compilación.
- Rendereo de alto performance.

Las vistas de Facelets son usualmente creadas como páginas XHTML. La tecnología Java Server Faces soporta varias librerías de etiquetas para agregar componentes a una página web. Para soportar el mecanismo de librerías de Java Server Faces, Facelets usa una declaración de namespaces de XML. La siguiente tabla de librerías de etiquetas es soportada por Facelets:

Tabla 5: Libreria de Etiquetas soportadas por Facelets

Biblioteca de etiqueta	URL	PREFIJO	CONTENIDO
Biblioteca de Facelets	http://java.sun.com/jsf/facelets http://xmlns.jcp.org/jsf/facelets	ui:	Etiquetas para plantillas
Biblioteca de HTML	http://java.sun.com/jsf/html http://xmlns.jcp.org/jsf/html	h:	Etiquetas para todos los componentes de tipo UIComponent
Biblioteca para elementos compatibles	http://primefaces.org/ui http://xmlns.jcp.org/jsf	p:	Etiquetas JSF compatibles para HTML5
Biblioteca para atributos compatibles	http://xmlns.jcp.org/jsf/passthrough	jsf:	Atributos JSF compatibles para HTML5
Biblioteca JSTL Core	http://xmlns.jcp.org/jsp/jstl/core	c:	Etiquetas de Core JSTL 1.2
Biblioteca de funciones JSTL	http://xmlns.jcp.org/jsp/jstl/functions	fn:	Etiquetas de Funciones JSTL 1.2

Fuente: http://docs.oracle.com/javaee/6/tutorial/doc/gijtu.html

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:p="http://primefaces.org/ui">
    <h:head>
        <title>
            <ui:define name="Gestión de Facultades">GESTION DE FACULTADES</ui:define>
        </title>
    </h:head>
```

Ilustración 14: Facelets - Vista Facultad.xhtml
Fuente: Sandra Contento/Janneth Guamán

2.4.5.2. Clases de los componentes del interface de usuario

La tecnología Java Server Faces proporciona un conjunto de clases de componentes UI, que especifican toda la funcionalidad del componente, cómo mantener su estado, mantener una referencia a objetos del modelo, y dirigir el manejo de eventos y el renderizado para un conjunto de componentes estándar.

Estas clases son completamente extensibles, lo que significa que se pueden extender para crear propios componentes personalizados.

Todas las clases de componentes UI de Java Server Faces descienden de la clase UIComponentBase, que define el estado y el comportamiento por defecto de un UIComponent. El conjunto de clases de componentes UI incluido en la última versión de Java Server Faces es:

- **UICommand:** Representa un control que dispara acciones cuando se activa.
- **UIForm:** Encapsula un grupo de controles que envían datos de la aplicación. Este componente es análogo a la etiqueta form de HTML.
- **UIGraphic:** Muestra una imagen.
- **UIInput:** Toma datos de entrada del usuario. Esta clase es una subclase de UIOutput.
- **UIOutput:** Muestra la salida de datos en un página.
- **UIPanel:** Muestra una tabla.
- **UIParameter:** Representa la sustitución de parámetros.
- **UISelectItem:** Representa un sólo ítem de un conjunto de ítems.
- **UISelectItems:** Representa un conjunto completo de ítems.
- **UISelectBoolean:** Permite a un usuario seleccionar un valor booleano en un control, selececcionándolo o deseleccionándolo. Esta clase es una subclase de UIInput.
- **UISelectMany:** Permite al usuario seleccionar varios ítems de un grupo de ítems. Esta clase es una subclase de UIInput.
- **UISelectOne:** Permite al usuario seleccionar un ítem de un grupo de ítems. Esta clase es una subclase de UIInput.

Toda aplicación Java Server Faces debe incluir una librería de etiquetas personalizadas que define las etiquetas que representan componentes UI, así como una librería de etiquetas para controlar otras acciones importantes, como

validadores y manejadores de eventos. La implementación de Java Server Faces permite estas dos librerías.

La librería de etiquetas de componentes elimina la necesidad de codificar componentes UI en HTML u otro lenguaje de marcas, lo que se traduce en el empleo de componentes completamente reutilizables, y la librería principal hace fácil registrar eventos, validadores y otras acciones de los componentes.

2.4.5.3. Modelo de renderizado de componentes

La arquitectura de componentes Java Server Faces está diseñada para que la funcionalidad de los componentes se defina mediante las clases de componentes, mientras que el renderizado de los componentes se puede definir mediante un renderizador separado. Este diseño tiene varios beneficios:

- Se puede definir sólo una vez el comportamiento de un componente, pero se pueden crear varios renderizadores, cada uno de los cuales define una forma diferente de dibujar el componente para el mismo cliente o para diferentes clientes.
- Los autores de páginas y los desarrolladores de aplicaciones pueden modificar la apariencia de un componente de la página seleccionando la etiqueta que representa la combinación componente/renderizador apropiada.

La implementación Java Server Faces incluye un RenderKit estándar para renderizar a un cliente HTML.

Cada etiqueta JSF personalizada en el RenderKit de HTML, está compuesta por la funcionalidad del componente, definida en la clase UIComponent, y los atributos de renderizado, definidos por el Renderer.

La implementación de referencia de Java Server Faces proporciona una librería de etiquetas personalizadas para renderizar componentes en HTML. A continuación se listan los componentes que son soportados:

Tabla 6: Etiquetas personalizadas para renderizar componentes en HTML

ETIQUETA	FUNCIONES	SE RENDERIZA COMO:
command_button	Enviar un formulario a la aplicación	Elemento "input type=type" HTML, donde el valor del tipo puede ser submit, reset, o image.
command_hyperlink	Enlaza a otra página o localización en otra página	Elemento"a href" HTML

Form	Representa un formulario de entrada.	Elemento "form" HTML
graphic_image	Muestra una imagen	Elemento "img" HTML
input_date	Permite al usuario introducir una fecha	Elemento "input type=text" HTML
input_datetime	Permite al usuario introducir una fecha y una hora	Elemento "input type=text" HTML
input_hidden	Permite introducir una variable oculta en una página	Elemento "input type=hidden" HTML
input_number	Permite al usuario introducir un número	Elemento "input type=text" HTML
input_secret	Permite al usuario introducir un string sin que aparezca el string real en el campo	Elemento "input type=password" HTML
input_text	Permite al usuario introducir un string	Elemento "input type=text" HTML
input_textarea	Permite al usuario introducir un texto multi-líneas	Elemento "textarea" HTML
input_time	Permite al usuario introducir una hora	Elemento "input type=text" HTML
output_date	Muestra una fecha formateada	Texto normal
output_datetime	Muestra una fecha y hora formateados	Texto normal
output_errors	Muestra mensajes de error	Texto normal
output_label	Muestra un componente anidado como una etiqueta para un campo de texto especificado	Elemento "label" HTML
output_message	Muestra un mensaje localizado (internacionalizado)	Texto normal
output_number	Muestra un número formateado	Texto normal
output_text	Muestra una línea de texto	Texto normal
output_time	Muestra una hora formateada	Texto normal
panel_data	Itera sobre una colección de datos	Conjunto de filas en una tabla
panel_grid	Muestra una tabla	Elemento "label" HTML.con elementos "tr" y "lt,td"
panel_group	Agrupa un conjunto de paneles bajo un padre	Una fila en una tabla
panel_list	Muestra una tabla de datos que vienen de una collection, un array, un iterator o un map	Elemento "table" HTML. con elementos "tr" y "lt,td"
selectboolean_checkbox	Permite al usuario cambiar el valor de una elección booleana	Elemento "input type=checkbox" HTML
Selectitem	Representa un ítem de una lista de ítems en un componente UISelectOne	Elemento "option" HTML
selectitems	Representa una lista de ítems en un componente UISelectOne	Elemento "option" HTML
selectmany_checkboxlist	Muestra un conjunto de checkbox, en los que el usuario puede seleccionar varios	Conjunto de elementos "input" HTML

selectmany_listbox	Permite a un usuario seleccionar varios ítems de un conjunto de ítems, todos mostrados a la vez	Conjunto de elementos "select" HTML
selectmany_menu	Permite al usuario seleccionar varios ítems de un grupo de ítems	Conjunto de elementos "select" HTML
selectone_listbox	Permite al usuario seleccionar un ítem de un grupo de ítems	Conjunto de elementos "select" HTML
selectone_menu	Permite al usuario seleccionar un ítem de un grupo de ítems	Conjunto de elementos "select" HTML
selectone_radio	Permite al usuario seleccionar un ítem de un grupo de ítems	Conjunto de elementos "input type=radio" HTML

Fuente: http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/130

2.4.5.4. Modelo de Conversión

Una aplicación Java Server Faces opcionalmente puede asociar un componente con datos del objeto del modelo del lado del servidor. Este objeto del modelo es un componente JavaBeans que encapsula los datos de un conjunto de componentes. Una aplicación obtiene y configura los datos del objeto modelo para un componente llamando a las propiedades apropiadas del objeto modelo para ese componente.

Cuando un componente se une a un objeto modelo, la aplicación tiene dos vistas de los datos del componente: la vista modelo y la vista presentación, que representa los datos de un forma que el usuario pueda verlos y modificarlos.

Una aplicación Java Server Faces debe asegurarse que los datos del componente puedan ser convertidos entre la vista del modelo y la vista de presentación. Esta conversión normalmente la realiza automáticamente el renderizador del componente.

La tecnología Java Server Faces incluye un conjunto de implementaciones estándar de Converter que permite crear conversores personalizados. La implementación de Converter convierte los datos del componente entre las dos vistas.

2.4.5.5. Modelo de Eventos y Oyentes

El modelo de eventos y oyentes de Java Server Faces mejora el diseño del modelo de eventos de JavaBeans, que es familiar para los desarrolladores de GUI y de aplicaciones Web.

Al igual que la arquitectura de componentes JavaBeans, la tecnología Java Server Faces define las clases Listener y Event que una aplicación puede utilizar para manejar eventos generados por componentes UI.

Un objeto Event identifica al componente que lo generó y almacena información sobre el propio evento. Para ser notificado de un evento, una aplicación debe proporcionar una implementación de la clase Listener y registrarla con el componente que genera el evento. Cuando el usuario activa un componente se dispara un evento. Esto hace que la implementación de Java Server Faces invoque al método oyente que procesa el evento. Java Server Faces soporta dos tipos de eventos: eventos value-changed y eventos action.

Un evento value-changed ocurre cuando el usuario cambia el valor de un componente. Los tipos de componentes que generan estos eventos son los componentes UIInput, UISelectOne, UISelectMany, y UISelectBoolean. Este tipo de eventos sólo se dispara si no se detecta un error de validación, mientras que un evento action ocurre cuando el usuario pulsa un botón o un hiperenlace. El componente UICommand genera este evento.

2.4.5.6. Modelo de Validación

Al igual que el modelo de conversión, el modelo de validación define un conjunto de clases estándar para realizar chequeos de validación comunes. La librería de etiquetas jsf-core también define un conjunto de etiquetas que corresponden con las implementaciones estándar de Validator.

2.4.5.7. Modelo de navegación

La implementación de Java Server Faces proporciona un nuevo oyente de eventos action por defecto para manejar este evento. Este oyente determina la salida del evento action, como success o failure. Esta salida se puede definir como una propiedad String del componente que generó el evento o como el resultado de un procesamientro extra realizado en un objeto Action asociado con el componente. Después de determinar la salida, el oyente la pasa al ejemplar de NavigationHandler asociado con la aplicación. Basándose en la salida devuelta, el NavigationHandler selecciona la página apropiada consultando el fichero de configuración de la aplicación.

2.4.6. Versiones JSF

- JSF 1.0 (11-03-2004)
    - Lanzamiento inicial de las especificaciones de JSF.
- JSF 1.1 (27-05-2004)
    - Lanzamiento que solucionaba errores.
    - Especificación JSR-127: http://jcp.org/en/jsr/detail?id=127.
    - Sin cambios en las especificaciones ni en el renderkit de HTML.
- JSF 1.2 (11-05-2006)
    - Lanzamiento con mejoras y corrección de errores.
    - Especificación JSR-252: http://jcp.org/en/jsr/detail?id=252
- JSF 2.0 (12-08-2009)
    - Lanzamiento con mejoras de funcionalidad, rendimiento y facilidad de uso.
    - Especificación JSR-314 (JSF 2.0).
- JSF 2.1 (22-10-2010)
    - Lanzamiento de mantenimiento, con mínimos cambios.
- JSF 2.2 (16-04-2013)
    - Lanzamiento que introduce soporte a HTML 5, Faces Flow, Stateless views y Resource library contracts.

2.4.7. Entornos de desarrollo integrado (IDEs) compatibles con JSF

Se pueden encontrar diferentes entornos, como por ejemplo:

- Java Studio Creator
- Eclipse SDK
- NetBeans
- Oracle JDeveloper
- Borland JBuilder

2.4.7.1. Java Studio Creator[14]

Java Studio Creator es una solución completa de desarrollo, depuración y despliegue de aplicaciones, que incorpora la tecnología de producción Java Server

[14] http://www.oracle.com/technetwork/articles/java/jscoverview-135211.html

Faces (JSF), capaz de proporcionar un mayor rendimiento y eficiencia en la codificación.

El producto también incorpora:

- El runtime de Java Enterprise System, que permite a los desarrolladores evaluar las aplicaciones en preproducción de forma rápida.
- Incluye soporte para Sun Java System Application Server, Platform Edition.
- PointBase, un servidor de bases de datos SQL (incluido en Java System Application Server).
- El kit de desarrollo (SDK) para Java 2, Standard Edition (J2SE).
- Elementos de ayuda para desarrolladores, tales como ejemplos, tutoriales, componentes visuales para la tecnología Java Server Faces, etc.

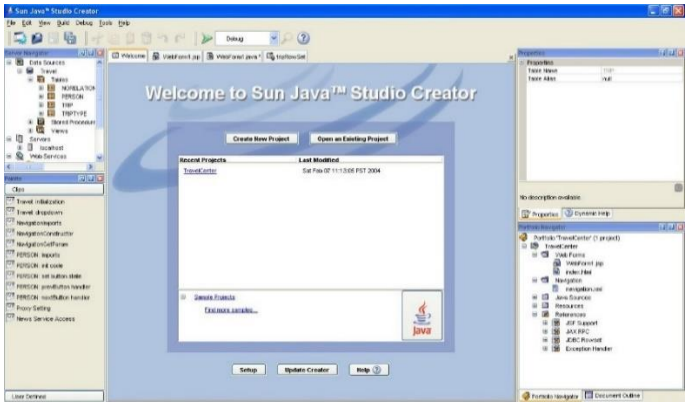


Ilustración 15 Aspecto del IDE Sun Java Studio Creator 2
Fuente: http://bit.ly/1chgTjP

2.4.7.2. Eclipse SDK[15]

Eclipse lo puede descargar en:

- http://www.eclipse.org/downloads/

Es un IDE multiplataforma libre para crear aplicaciones clientes de cualquier tipo. La primera y más importante aplicación que ha sido realizada con este entorno es la afamado IDE Java llamado Java Development Toolkit (JDT) y el compilador incluido en Eclipse, que se usaron para desarrollar el propio Eclipse.

La versión actual de Eclipse dispone de las siguientes características:

- Editor de texto
- Resaltado de sintaxis

[15] https://wiki.eclipse.org/Eclipse

- Compilación en tiempo real
- Pruebas unitarias con JUnit
- Control de versiones con CVS
- Asistentes (wizards): para creación de proyectos, clases, tests, etc.
- Refactorización

Asimismo, a través de "plugins" libremente disponibles es posible añadir:

- Control de versiones con Subversion, via Subclipse.
- Integración con Hibernate, via Hibernate Tools.

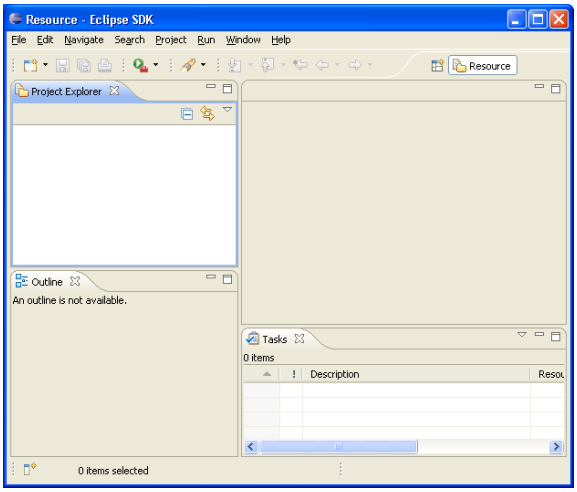


Ilustración 16 Aspecto del IDE Eclipse SDK
Fuente: http://bit.ly/1Rm2OlD

2.4.7.3. Netbeans[16]

El IDE NetBeans, lo puede descargar junto con la Máquina virtual de java, desde:

- http://java.sun.com/j2se/1.5.0/download.jsp

Es un entorno de desarrollo, de código abierto, una herramienta para programadores para escribir, compilar, corregir errores y para ejecutar programas. Está escrito en Java pero puede servir de soporte a cualquier otro lenguaje de programación. Existe también un número enorme de módulos para extender el NetBeans IDE. El NetBeans IDE es un producto libre y gratuito sin restricciones de utilización.

[16] https://netbeans.org/features/index.html

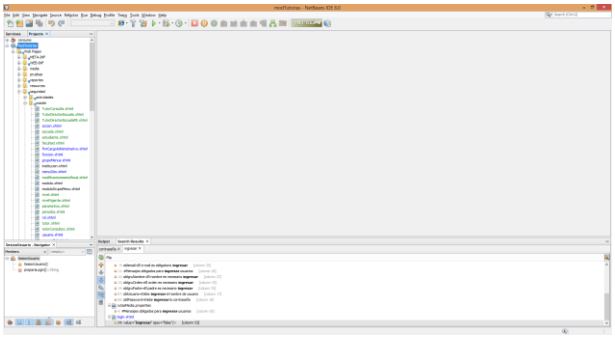


Ilustración 17 Aspecto del IDE NetBeans 8.0
Fuente: Sandra Contento/Janneth Guamán

2.4.7.4. Oracle JDeveloper[17]

KDevelop es un entorno de desarrollo integrado para sistemas GNU/Linux y otros sistemas Unix, publicado bajo licencia GPL, orientado al uso bajo el entorno gráfico KDE, aunque también funciona con otros entornos, como Gnome.

KDevelop 4.0 ha sido reconstruido completamente desde los cimientos, se dio a conocer para KDE la versión 4.0 en mayo de 2010. A diferencia de muchas otras interfaces de desarrollo, KDevelop no cuenta con un compilador propio, por lo que depende de GCC (colección de compiladores GNU) para producir código binario.

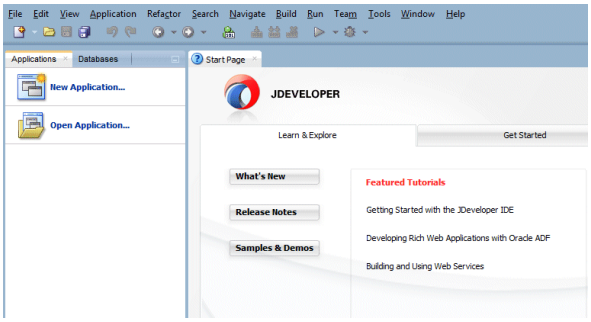


Ilustración 18 Aspecto del IDE JDeveloper
Fuente: http://bit.ly/1cx8mdo

2.4.7.5. Borland JBuilder[18]

JBuilder es un IDE Java de Borland. Es un software creado en 1995.

- Mejorar la calidad del código y el rendimiento
- Aumentar la productividad individual y de equipo
- Mejorar la comprensión del código nuevo o existente.

La versión 2006 (Borland JBuilder 2006) tiene 3 ediciones:

[17] http://www.oracle.com/technetwork/articles/java/jscoverview-135211.html

[18] http://www.embarcadero.com/products/jbuilder

- **Enterprise**.- para aplicaciones J2EE, Web Services y Struts.
- **Developer**.- para el completo desarrollo de aplicaciones Java.
- **Foundation**.- con capacidades básicas para iniciarse en el desarrollo de aplicaciones javas y de momento es de libre uso.

Ilustración 19 Aspecto del IDE Borland JBuilder
Fuente: http://bit.ly/1GUt7XK

2.4.8. Servidores de Aplicaciones JSF

2.4.8.1. GlassFish[19]

Ilustración 20 Servidor Glassfish
Fuente: glassfish.java.net

GlassFish es un servidor de aplicaciones de software libre desarrollado por Sun Microsystems, compañía adquirida por Oracle Corporation, que implementa las tecnologías definidas en la plataforma Java EE y permite ejecutar aplicaciones que siguen esta especificación. Es gratuito, de código libre y se distribuye bajo un licenciamiento dual a través de la licencia de desarrollo y distribución común (CDDL) v1.0 y la Licencia pública general GNU (GPL) v2. La versión comercial es denominada Oracle GlassFish Enterprise Server (antes Sun GlassFish Enterprise Server).

GlassFish está basado en el código fuente donado por Sun y Oracle Corporation; este último proporcionó el módulo de persistencia TopLink.1 GlassFish tiene como

[19] https://glassfish.java.net/es/

base al servidor Sun Java System Application Server de Oracle Corporation, un derivado de Apache Tomcat, y que usa un componente adicional llamado Grizzly que usa Java NIO para escalabilidad y velocidad.

2.4.8.2. Apache Tomcat[20]

Ilustración 21 Servidor Apache Tomcat
Fuente: http://tomcat.apache.org/

Apache Tomcat (también llamado Jakarta Tomcat o simplemente Tomcat) funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de Java Server Pages (JSP) de Oracle Corporation.

Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache Software License.

Tomcat es un contenedor web con soporte de servlets y JSPs. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor web Apache.

Tomcat puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones.

Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java.

[20] http://tomcat.apache.org/

2.4.8.3. JBoss[21]

Ilustración 22 Servidor JBoss
Fuente: http://bit.ly/1IXfGLw

JBoss es un servidor de aplicaciones Java EE de código abierto implementado en Java puro.

Las características destacadas de JBoss incluyen:

- Producto de licencia de código abierto sin coste adicional.
- Cumple los estándares.
- Confiable a nivel de empresa
- Incrustable, orientado a arquitectura de servicios.
- Flexibilidad consistente
- Servicios del middleware para cualquier objeto de Java.
- Soporte completo para JMX.

2.4.8.4. Jonas[22]

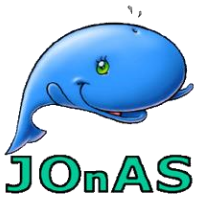


Ilustración 23 Servidor Jonas
Fuente: http://jonas.ow2.org/xwiki/bin/view/Main/

JOnAS es un servidor de aplicaciones J2EE de código abierto implementado en Java. Proporciona un contenedor EJB totalmente compatible mediante EasyBeans y está disponible con un contenedor web incrustado Tomcat o Jetty. Se admite cualquier JVM 1.5 o 1.6, y los intentos de ejecutar en una pila libre con GNU Classpath son prometedores.

[21] http://www.jboss.org/products/eap/overview/
[22] http://jonas.ow2.org/xwiki/bin/view/Main/

JOnAS se puede ejecutar en numerosos sistemas operativos como Linux, Windows, AIX, muchas plataformas POSIX y otros, siempre que esté disponible una máquina virtual Java (JVM).

2.4.9. Ventajas e inconvenientes de JSF[23]

Existen numerosas ventajas que hacen que JSF sea una tecnología apropiada para el desarrollo de aplicaciones web:

- Una de las grandes ventajas de la tecnología Java Server Faces es que ofrece una clara separación entre el comportamiento y la presentación. Las aplicaciones Web construidas con tecnología JSP conseguían parcialmente esta separación. Sin embargo, una aplicación JSP no puede mapear peticiones HTTP al manejo de eventos específicos de componentes o manejar elementos UI como objetos con estado en el servidor.
- La tecnología Java Server Faces permite construir aplicaciones Web que implementan una separación entre el comportamiento y la presentación tradicionalmente ofrecida por arquitectura de UI del lado del cliente. JSF se hace fácil de usar al aislar al desarrollador del API de Servlet.
- 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.
- Otro objetivo importante de la tecnología Java Server Faces es mejorar los conceptos de componente-UI y capa-Web sin limitar a una tecnología de script particular o un lenguaje de marcas. Aunque la tecnología Java Server Faces incluye una librería de etiquetas JSP personalizadas para representar componentes en una página JSP, los APIs de la tecnología Java Server Faces se han creado directamente sobre el API JavaServlet. Esto permite hacer algunas cosas: usar otra tecnología de presentación junto a JSP, crear componentes propios personalizados directamente desde las clases de componentes, y generar salida para diferentes dispositivos cliente. Así, se podrán encapsular otras

[23] https://code.google.com/p/fap-devel/wiki/JavaServerFaces

tecnologías como Ajax en componentes JSF, haciendo su uso más fácil y productivo, al aislar al programador de ellas.

- Java Server Faces ofrece una gran cantidad de componentes opensource para las funcionalidades que se necesiten. Los componentes Tomahawk de MyFaces y ADFFaces de Oracle son un ejemplo. Además, también existe una gran cantidad de herramientas para el desarrollo IDE en JSF al ser el estándar de JAVA.
- La tecnología Java Server Faces proporciona una rica arquitectura para manejar el estado de los componentes, procesar los datos, validar la entrada del usuario, y manejar eventos.
- Además, ofrece una rápida adaptación para nuevos desarrolladores.

No obstante, el uso de Java Server Faces también tiene un conjunto de desventajas:

- 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.

2.5. Librería de Componentes

2.5.1. PrimeFaces (PrimeFaces, 2009-2014)

PrimeFaces es una librería Open Source construida y mantenida por Prime Technology, una compañía Turca especializada en consultoría en desarrollo de aplicaciones ágiles, JSF, Java EE y Outsourcing.

PrimeFaces es una librería de componentes visuales para JSF que posee un conjunto de componentes ricos facilitando la creación de las aplicaciones web. PrimeFaces está bajo la custodia de la licencia de Apache License V2, convirtiéndose en un Framework flexible y relativamente fácil de usar, con una guía de un poco más de 450 páginas una comunidad bastante activa.

Ilustración 24 Librería Primefaces
Fuente: http://www.primefaces.org/index

2.5.1.1. Principales Características

- Muy liviana
- Cuenta con tecnología Ajax basada en el API Ajax JSF 2.0

- Cuenta con un kit mobile UI permitiendo la construcción de aplicaciones web para dispositivos móviles.
- Compatible con otras librerías de componentes, JBoss RichFaces, IceFaces.
- No requiere ninguna configuración, solo debe copiar el archivo JAR de la versión de PrimeFaces en la carpeta lib del contexto de la aplicación.
- No requiere dependencias.

PrimeFaces es transparente para el desarrollador, aunque para activarlo deben utilizarse atributos específicos para lanzar un método del servidor y para indicar los componentes a actualizar.

2.5.1.2. Propiedades

- Cuenta con un conjunto de componentes ricos por ejemplo: editor de HTML, autocompletar, cartas, gráficas o paneles, etc.
- Posee un soporte parcial Ajax, permitiendo controlar que componentes de la página se están actualizando y cuáles no.
- Cuenta con más de 35 temas pre desarrollados.
- Puede soportar el editor visual de temas.
- Posee componentes para la construcción de aplicaciones web para celulares, especiales para Iphones, Palm, Android y teléfonos móviles.

2.5.1.3. Versiones Primefaces

- Primefaces 1.- Trabaja con JSF 1.2
- Primefaces 2.- Trabaja con JSF 2.

2.5.1.4. Navegadores Soportados

Primefaces es compatible con los siguientes navegadores:

- Internet Explorer
- Firefox
- Chrome

2.5.1.5. Implementaciones de Java Server Faces Soportadas

Las implementaciones de Java Server Faces que soporta PrimeFaces son:

- Sun JSF 1.1 RI,
- Facelets JSF 1.1.1 - 1.2

2.5.1.6. Servidores Soportados

Primefaces proporciona integración con los siguientes servidores:

- Apache Tomcat 4.1 - 6.0
- Glassfish (J2EE 5)
- JBoss 3.2 - 4.2.x

2.5.1.7. Ventajas

- Soporta HTML5.
- Utiliza jQuery extensivamente.
- Está integrado con ThemeRoller Framework CSS, en donde se puede elegir entre los 36 temas pre-diseñados o incluso crear los propios con la herramienta online de generador de temas, permitiendo al desarrollador contar con una amplia gama de opciones para todos los gustos.
- Documentación mantenida por la comunidad muy actualizada.
- En cuanto a la experiencia con los clientes finales, muestran que los componentes de PrimeFaces son amigables y atractivos con el usuario, contando con diseños innovadores

2.5.1.8. Desventajas

- Para hacer uso del soporte Ajax se debe indicar explícitamente, a través de atributos específicos en cada uno de los componentes.
- No se puede utilizar el soporte Ajax de JSF con los componentes de PrimeFaces.

2.5.1.9. Componentes PrimeFaces

Tabla 7: Listado de Componentes PrimeFaces

TIPO	COMPONENTE	DESCRIPCION
AjaxCore	Basic	Muestra las características de actualización del Ajax básico.
	Partial Processing	Permite elegir los componentes para procesar mediante el atributo process. Útil para hacer validaciones parciales, actualización de modelos, invocar la aplicación y mucho más. Palabras clave como: @this, @form, @all, @none, @parent hacen que sea aún más fácil elegir lo que debe procesar, solo para Ajax.

	Partial Submit	Reduce el tráfico de red mediante la adición de los componentes solo parcialmente procesados para el cargo petición Ajax. Para las páginas grandes con muchos componentes y entrada, partialSubmit es muy útil ya que da lugar a las peticiones más ligeras.
	Validations	Se ejecutan en el servidor y la página se actualiza con Ajax.
	Selectors	La potencia total del Selector API se puede utilizar con o sin el componente referenciado regular
	Poll	Hace llamadas Ajax en un intervalo especificado.
	Remote Command	Permite ejecutar métodos de respaldo de beans y hacer actualizaciones parciales provocadas por secuencias de comandos personalizadas del lado del cliente.
	AjaxStatus	Es un indicador global para informar a los usuarios acerca de las interacciones Ajax.
	Events	Habilita el comportamiento Ajax en cualquiera de los componentes JSF.
	Search	Amplía las expresiones de búsqueda predeterminado por palabras clave y permite combinar las expresiones.
	Listeners	Componente p: Ajax ejecuta un listener en un beans JSF.
	Counter	Una variable entera que cuenta.
	Dropdown	Desplegables dependientes implementados con p: ajax behavior.
Input	AutoComplete	Autocompletar en un campo de entrada
	BooleanButton	Se utiliza para proporcionar selección binaria con una interfaz de usuario de botón en lugar de una casilla de verificación
	BoolCheckbox	Extiende SelectBooleanCheckbox estándar con capacidades de desglosar.
	Calendar	Es un componente de selector de fecha de gran alcance que ofrece diferentes modos de visualización.
	CheckboxMenu	Es un componente de entrada de selección múltiple basado en casillas de verificación en un menú.
	Color Picker	Selección de colores en lugar de escribir el código hexadecimal.
	Editor	Es un componente de entrada con ricas características de edición de texto.
	Inplace	Para la edición in-situ, “guardar” y “cancelar”.
	InputMask	Componente de entrada para ser formateado de manera específica.
	InputText	Extiende el InputText estanadar con capacidades de desglosar.
	InputTextarea	Extiende el estándar InputTextarea con autoResize, barra y características de tematización.
	Keyboard	Muestra el teclado en pantalla, según lo requerido puede mostrar el teclado completo, solo alfabético, solo numérico, tipo password, etc
	ManyButton	Es un componente de entrada para seleccionar las opciones con botones en vez de casillas de verificación.
	ManyMenu	Extiende el SelectManyMenu con personalización.
	ManyCheckbox	Extiende el SelectManyCheckbox con capacidades de desglose.

	MultiSelectListbox	Se utiliza para seleccionar un elemento de una colección de cuadros de lista que se encuentran en la relación padre-hijo.
	OneButton	Componente de entrada para seleccionar las opciones con los botones regulares en un lugar de botones de radio.
	One Menu	Extiende el SelectOneMenu estándar con capacidades de desglose, edición, efectos y visualización de contenido personalizado.
	OneListbox	Extiende el SelectOneListBox estándar con personalización.
	OneRadio	Extiende el SelectOneRadio estándar con capacidades de aplicación de aspectos y características de diseño personalizado
	Password	Versión extendida del componenete inputSecret integrando temas e indicador de intensidad o fuerza de la clave.
	Rating	Componente que provee una entrada de estrellas basada en una clasificación.
	Spinner	Usado para proporcionar entradas de botones con incrementos o decrementos en una entrada de texto o inputText
	Switch	nputSwitch se utiliza para seleccionar un valor booleano
	Slider	Usado para mostrar entradas de varias formas con una barra de desplazamiento vertical u horizontal.
Button	Button	Es una extensión de la etiqueta estandar h: button, recibe peticiones dirigidas a las direcciones URL.
	CommandButton	Extensión de h:commandLink con Ajax, procesamiento parcial y características de confirmación.
	CommandLink	Extensión de h:commandButton con Ajax, procesamiento parcial y capacidades de desglose.
	Link	Es un componente de navegación integrado con el modelo de navegación JSF
	SplitButton	Muestra un comando por defecto.
Data	Carousel	Componente multiuso para mostrar un conjunto de datos o contenido general. Tiene Simple Carousel, Item Selection, Effects, Tab Slider, SlideShow.
	DataExporter	Sirve para exportar tabla de datos a varios formatos como Excel, pdf, csv y xml.
	DataList	Presenta una colección de datos en una lista con diseño y varios tipos de visualización, características como: Unordered DataList, Ordered DataList, Definition DataList, Ajax Pagination.
	DataGrid	Muestra datos en una cuadricula diseñada. Ajax Pagination es una característica integrada y la interfaz de usuario es totalmente personalizable.
	DataScroller	Muestra los datos con la demanda de carga usando desplazamiento.
	DataTable	Es un componente iteración datos con paginación Ajax, ordenar, filtrar varias soluciones de selección de fila, encabezados anidados, filas expansibles, edicio en la celda, desplazamiento.
	Diagram	Es componente genérico para crear elementos visuales y conectarlos en una página web

	GMap	Este componente requiere atributos de zoom y el tipo para crear el mapa.
	HorizontalTree	Presenta dos modos: el modo cliente en donde todos los nodos están disponibles y el modo Ajax en donde solamente están dispoibles los nodos expandidos.
	Mindmap	Es una herramienta interactiva que ofrece mapas mentales, devoluciones de llamada, animaciones y más.
	PickList	Es un componente de lista de entrada dual con arrastrar y soltar basada con reordenamiento, efectos de transición, soporte para temas y más.
	OrderList	Se utiliza para clasificar una colección a través de arrastrar y soltar basada con reordenamiento basado, efectos de transición.
	Shedule	Es un componente parecido a una tabla Excel, con manipulación de datos, ordenamiento horizontal o vertical, teclas de navegación, etc.
	Ring	Es un componente de visualización de datos con una animación circular.
	TagCloud	Exhibe una colección de etiquetas con diferentes fortalezas.
	Tree	Se utiliza para mostrar los datos de cualquier jerarquía o propósito de navegación. Con opciones de estilo, Ajax expandir, contraer y seleccionar los eventos, una función de selección basad en casilla de verificación.
	TreeTable	Usado para mostrar jerarquías de datos en un formato tabular.
Panel	Accordion	Es un componente contenedor con los paneles verticalmente apilados.
	Dashboard	Basado en la reordenación.
	Fieldset	Componente de agrupación.
	Grid CSS	Es un peso ligero (1,4 kb) utilidad diseño sensible optimizado para dispositivos móviles, tablets y computadoras de escritorio
	Layout	Diseño de borde que puede ser aplicado a una página completa o solo partes de ella, puede expandir, contraer, redimensionar, cerrar.
	NotificationBar	Muestra un panel fijo de usos múltiples ubicado para notificación ya sea en la parte superior o inferior de la página. Cualquier grupo de componentes JSF puede colocarse dentro de la barra de notificación.
	OutputPanel	Es un elemento contenedor con varios casos de uso
	Panel	Es un componente de agrupación genérica que también soporta comunicación alternativa, el cierre y el menú de opciones.
	PanelGrid	Es una extensión del panelGrid estándar con la integración de temas y soporte colspan-rowspan.
	Ribbon	Este componente tiene un contenedor para agrupar diferentes conjuntos de controles en un diseño con pestañas.
	ScrollPanel	Se utiliza para mostrar cantidades grandes de contenido con barras de desplazamiento.
	TabView	Es un componente poderoso para panel de pestañas con pestañas, carga de contenido dinámico con Ajax,

		orientaciones diferentes, la creación / eliminación de fichas mediante programación y mucho más.
	Toolbar	Componente para agrupación de botones y otros contenidos.
	Wizard	Asistente de flujo es secuencial por defecto y esto puede ser logrado con el programa opcional flowListeners Ajax, simplemente resultado de una flowListener define el siguiente paso para mostrar. Paso actual se procesa parcialmente y el paso siguiente se muestra si pasa la validación.
Overlay	ConfirmDialog	Cuadros de diálogos de confirmación.
	Dialog	Es un componente contenedor que puede colocar otros elementos en la página. Dialog tiene varias opciones de personalización, tales como modal, cambiar el tamaño, anchura, altura, posición. Cuadro de dialogo.
	LightBox	Es un componente de superposición modal para mostrar imágenes, contenido en línea e iframes.
	OverlayPanel	Es un componente contenedor genérico que puede sobreponer otros componentes en la página. Puede presentarse en un panel básico, dinámico o a modo de imagen.
	Tooltip	Muestra mensajes de ayuda
Menu	BreadCrumb	Proporciona información contextual sobre la jerarquía de página.
	ContextMenu	Es un menú de superposiciones que se muestra al dar click en el botón derecho del raton que se puede conectar a cualquier otro componente JSF.
	Dock	Acoplar menús en enlaces
	MegaMenu	Muestra los submenús de elementos raíz juntos
	Menu	Es personalizable, componente de comando y navegación que soporta posicionamiento dinámico y estático.
	Menubar	Aporta las barras de menú de aplicaciones de escritorio para JSF.
	MenuButton	Agrupa varios comandos en un menú emergente.
	PanelMenu	Componente hibrido de árbol-acordeon usado para navegación y acción.
	SlideMenu	Muestra submenús con animación slide.
	Stack	Muestra menuitems en formato apilado.
	TabMenu	Componente menú que muestra ítems en pestañas
	TieredMenu	Muestra submenús anidados.
Charts	Area	Activa opciones de apilado y relleno de lineChart.
	Bar, Bubble, Donut, Line, Pie, MeterGauge, OHCLC, Animate Export, Interactive, Live, Update, Static, Zoom	Componentes que permite mostrar gráficos estadísticos en forma de barras, líneas, etc.

Message	Growl	Muestra mensajes en una superposición
	Messages	Muestra aviso de mensajes
Mulitmedia	Compare	Provee interfaz gráfica para comparar imágenes similares.
	Cropper	Usado para hacer recortes de imágenes.
	dynaImage	Presenta imágenes creadas en tiempo de ejecución o de la base de datos.
	Galleria	Componentes para galería de imágenes.
	Media	Incrusta audio y video.
	PhotoCam	Captura fotos desde la cámara del computador.
	Switch	Soporta 25 efectos.
File	Auto	Carga de archivos automáticamente llama ventana de archivos.
	Basic	Examina rutas para cargar archivos.
	DragDrop	Seleccionar un archivo, cargarlo o cancelar desde un panel con pestañas.
	Download	Descarga de archivos.
	Multiple	Carga de varios archivos.
	Single	Carga de archivo único.
DragDrop	Draggable	Permite el movimiento de componentes horizontal, vertical, área definida, con efectos y otras caracterisiticas.
	DataGrid	Integración con componente grid.
	DataTable	Integración con componente tabla.
Misc	Captcha	Componente para crear captcha o test controlado.
	Collector	Componente para manejar colecciones sin código java.
	DefaultCommand	Controla que componente inicia un formulario.
	Effects	Basado enjQuery, varios efectos de ventanas.
	ProgressBar	Barra de estado de preoceso.
	Resizable	Permite cambiar el tamalo de cualquier componente.
	Separator	Muestra un alinea horizontal cuando se usa como un componente individual, también para separator de menús y barra de herramientas.

Fuente: http://www.primefaces.org/showcase/

2.5.2. Richfaces (RichFaces, 2010)

RichFaces es un marco muy útil de código abierto que permite añadir capacidades de Ajax a sus aplicaciones JSF, usando los componentes estándar JSF, sin la necesidad de escribir código JavaScript y administrar la compatibilidad de JavaScript entre navegadores. 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.

Ilustración 25 Librería RichFaces
Fuente: http://richfaces.jboss.org/

2.5.2.1. Características de Richfaces

- 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.
- 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.

2.5.2.2. Versiones de Java Soportadas

La versión de Java soportada es JDK 1.5 y superiores.

2.5.2.3. Implementaciones de Java Server Faces Soportadas

Las implementaciones de Java Server Faces que soporta RichFaces son:

- Sun JSF 1.1 RI,
- MyFaces 1.1.1 - 1.2,
- Facelets JSF 1.1.1 - 1.2
- Seam 1.2. - 2.0.

2.5.2.4. Servidores Soportados

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
- Sybase EAServer 6.0.1.

2.5.2.5. Navegadores Soportados

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
- Safari 2.0.

2.5.2.6. Ventajas

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).

2.5.2.7. Inconvenientes Detectados

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

2.5.2.8. Componentes Richfaces

Tabla 8: Componentes RichFaces

TIPO	ETIQUETA	DESCRIPCION
Ajax Action	<aj4: commandButton>	Botón de envío de formulario en el que se puede indicar que únicamente actualice ciertos componentes evitando la recarga de todo el formulario.
	<aj4: commandLink>	Comportamiento similar a < aj4:commandButton > pero en un link
	<aj4: ajaxListener>	Similar a la propiedad actionListener o valueChangeListener pero con la diferencia de que la petición se hace al contenedor Ajax.

	<aj4: jsfFunction>	Se utiliza para pasarle un valor automáticamente a una función JavaScript tras recibirlo del servidor.
	<aj4: poll>	Componente encuesta define una manera de sondear periódicamente un servidor con el fin de provocar cambios de estado, o partes de actualización de su página.
	<aj4: push>	Realizar periódicamente petición AJAX al servidor, para simular los datos de inserción.
	<aj4: param>	La extensión principal es la posibilidad de asignar un valor a una propiedad de un bean gestionado directamente utilizando el atributo assignTo.
	<aj4: region>	Determina un área a decodificar en el servidor después de la petición Ajax. < aj4:status >
	<aj4:status >	Muestra el estado de la petición Ajax: procesando petición y petición terminada.
	<aj4: form>	Similar al < h:form > con la diferencia de que se puede enviar previamente el contenido al contenedor Ajax
	<aj4: actionparam>	Combina la funcionalidad de la etiqueta < f:param > y < f:actionListener >
	<aj4: outputPanel>	Se utiliza para agrupar componentes para aplicarles similares propiedades.
	<aj4: loadScript>	Inserta en la página las funciones JavaScript contenidas en un archivo *.js
	<aj4: loadStyle>	Igual que la etiqueta anterior pero para una hoja de estilos *.css
	<aj4: log>	Carga en la página una consola que muestra las trazas de los logs que devuelve el contenedor Ajax
	<aj4: include>	Se utiliza para incluir en la página el contenido de otra de acuerdo a la definición que se haga en las reglas de navegación del faces-config.
	<aj4: repeat>	Etiqueta para iterar sobre una colección y mostrar todos sus campos
	<aj4: keepAlive>	Permite mantener un bean en un estado determinado durante peticiones.
	<aj4: mediaOutput>	Componente que permite mostrar contenido multimedia.
	<rich:calendar>	Componente para crear elementos de calendario
	<rich:comboBox>	Proporciona combobox editable
	<rich:chart>	Permite al usuario trazar datos y crear gráficos. Las muestras siguientes demuestran línea, barras y gráficos circulares.

RICHFACES	<rich:contextMenu	Este componente proporciona sistemas de menú contextual jerárquicos similares a los encontrados en muchas aplicaciones de escritorio
	<rich:contextMenu>	Se utiliza para la creación de “multileveled context menus”.
	<rich:dataFilterSlider>	Se utiliza para crear un filtrar de los datos de una tabla.
	<rich:datascroller>	Diseñado para proporcionar la funcionalidad de los cuadros de desplazamiento utilizando solicitudes de Ajax.
	<rich:colums>	Permite crear columnas dinámicas.
	rich:clientId(id)	Devuelve cliente Id para el componente que es Identificación corto.
	<rich:jQuery>	Devuelve un objeto jQuery con un elemento de identificación corta dada.
	<rich:columnGroup>	Permite combinar las columnas en una fila para organizar.
	<rich:dataGrid>	Permite ver los datos de una rejilla que nos deja elegir los datos.
	<rich:dataList>	Permite prestar los datos de un modo lista.
	<rich:dataOrderedList>	Se usa para ordenar las listas de prestación que permite elegir los datos.
	<rich:dataTable>	Permite crear tablas de datos.
	<rich:editor>	Es un cmponente de entrada como editor WYSIWYG.
	<rich:fileUpload>	Este componente le permite subir archivos de la máquina de los usuarios al servidor con varias opciones.
	<rich:subTable>	Se utiliza para la inserción de sub tablas.
	<rich:dropDownMenu>	Se utiliza para crear múltiples menús desplegables.
	<rich:list>	Permite crear listas de forma dinámica a partir de datos de back-end. Las listas se pueden ordenar listas, listas desordenadas , o listas de definiciones
	<rich:menuGroup>	Se utiliza para definir un ampliable grupo de temas dentro de una lista emergente u otro grupo.
	<rich:menuItem>	Se utiliza para la definición de un único punto dentro de una lista emergente.
	<rich:menuSeparator>	Se utiliza para la definición de un separador horizontal que puede ser colocado entre los grupos de o los temas del programa.
	<rich:inplaceSelect>	Muy parecido al anterior. Se utiliza para seleccionar algo asi como un DrpDown

	<rich:listShuttle>	Se utiliza para mover los temas elegidos de una lista u otra con s facultativo reordenamiento.
	<rich:message>	Se utiliza para hacer un solo mensaje a un componente especifico
	<rich:select>	Es un componente que está diseñado para reemplazar al estándar h: selectOneMenu. Añade diversas características y opciones que incluyen la funcionalidad Ajax de núcleo, y desollado.
	<rich:panelMenu>	Este component se utiliza para crear un panel de menú lateral plegable. El componente tiene un aspecto predefinido y sentir que pueden ser de piel
	<rich:pickList>	Este componente permite seleccionar varios valores de una lista, con los controles de movimiento y reordenación de los elementos de la lista resultante.
	<rich:progressBar>	Muestra un widget de progreso estándar , y permite facetas adicionales como inicial , y personalizaciones estatales acabado
	<rich:orderingList>	Permite volver a ordenar los elementos de una lista. El orderingList también es compatible con una lista representación de varias columnas.
	<rich:toolbar>	Es un panel horizontal que se puede utilizar en un número de maneras , tales como un menú de nivel superior , un panel de información , y así sucesivamente

Fuente: http://showcase.richfaces.org/

CAPÍTULO III

ANÁLISIS COMPARATIVO DE LIBRERÍAS DE COMPONENTES

Con el estudio realizado de las librerías de componentes Primefaces y Richfaces, se procede a realizar un análisis comparativo entre estas librerías, para determinar cuál de ellas es la mejor opción con respecto al rendimiento para el desarrollo de una aplicación web. El proceso aplicado para el análisis comparativo es el siguiente:

Estudio General de las Librerías de Componentes

- Identificación de características de las librerías de componentes
- Definición de criterios y parámetros
- Valoración de librerías de componentes

Construcción de prototipos

- Escenario de Prueba
- Proceso de Prueba

Análisis de Resultados

- Analizar los resultados
- Comprobación de la Hipótesis

3.1.Estudio general de las librerías de componentes

3.1.1. Identificación de características de las librerías de componentes

Se identifica características generales que poseen cada librería de componentes:

Tabla 9: Características generales librerías de componentes

	Primefaces	Richfaces
Tecnologia	JSF	JSF
Diversidad de componentes	Alrededor de 117 componentes	Alrededor de 60 componentes
Internet Explorer (IE)	IE8 o superior	Parcialmente
Firefox	V1.4 o superior	V1.5 o superior
Chrome	V22 o superior	V1 o superior
JavaScript	SI	SI

Soporte Ajax	Es transparente para el desarrollador, aunque para activarlo deben utilizarse atributos específicos para lanzar un método del servidor y para indicar los componentes a actualizar	Se debe hacer uso de Ajax4JSF, que no es tan transparente para el desarrollador, puesto que, además de introducir los componentes de RichFaces, se tiene que añadir componentes no visuales

Fuente: Sandra Contento/Janneth Guamán

3.1.2. Definición de criterios y parámetros de valoración

Acorde con el estudio realizado en el capítulo II de las librerías de componente Primefaces y Richfaces se establece los siguientes parámetros e indicadores de comparación:

Tabla 10: Parámetros e indicadores de comparación

Parámetro	Indicador
Librería usada actualmente	Facilidad de uso
	Diversidad de Componentes
	Compatibilidad de navegadores
Librerías que han alcanzado mayor madurez a lo largo del desarrollo	Facilidad para iniciar
	Documentación
	Rendimiento

Fuente: Sandra Contento/Janneth Guamán

Librería usada actualmente

Se define este parámetro debido a las tendencias tecnológicas que cada día avanzan dejando atrás aquellas librerías que se vuelven obsoletas al momento de desarrollar una aplicación web. Para valorar este parámetro se dio un valor en base a los siguientes indicadores:

- **Facilidad de Uso:** Es necesario que las librerías de componentes permitan un manejo fácil y rápido para agilitar la programación de los desarrolladores.
- **Diversidad de Componentes:** Las librerías de componentes tienen una variedad de componentes que ayudan al desarrollador a elaborar páginas

web de una manera más rápida y obtener variedad de componentes para realizar una misma tarea.

- **Compatibilidad con Navegadores:** Es indispensable que las librerías de componentes sean capaces de poder ejecutarse sin distinción visual en cualquier navegador.

Librerías que han alcanzado mayor madurez a lo largo de su desarrollo

Se añadirá los siguientes indicadores que ayudarán a determinar las librerías que han alcanzado mayor madurez a lo largo de su desarrollo brindando al programador mejores herramientas al momento de realizar un proyecto:

- **Facilidad para iniciar.** Las librerías de componentes deben permitir al desarrollador iniciar cada uno de los componentes de una manera rápida, sin tener que incrementar librerías de inicio cada vez que se requiera usar dichas librerías.
- **Documentación:** Es necesario que cada librería de componentes posea la documentación necesaria, como tutoriales, manuales o ayudas on-line, para facilitar el uso de las mismas.
- **Rendimiento:** Al momento de ejecutar un proyecto con ayuda de las librerías de componentes, éste debe tener respuestas inmediatas, procesamiento de páginas, una tasa de transferencia de datos, etc., rápidas y eficaces.

A continuación se especifica los criterios de valoración que se utilizó para cuantificar los parámetros e indicadores que se estableceron.

Tabla 11: Parámetros de valoración

Valor	Descripción
1	Baja
2	Media
3	Alta

Fuente: Sandra Contento/Janneth Guamán

En la tabla 12 se muestra la respectiva comparación y valoración de librerías primefaces y richfaces, para determinar cuál es la librería más utilizada actualmente:

Tabla 12: Librerías de componentes

Librería de componentes	Facilidad de uso	Diversidad de componentes	Compatibilidad con navegadores	Total	%
Primefaces	3	3	3	9	60
Richfaces	2	2	2	6	40

Fuente: Sandra Contento/Janneth Guamán

A continuación se observa los valores y porcentajes en cuanto a los indicadores antes mencionados, estableciendo asi la librería que brinda un manejo fácil y rápido, con una variedad de componentes para la elaboración de páginas web, además la compatibilidad con navegadores y la capacidad de ejecutarse sin ninguna distinción visual.

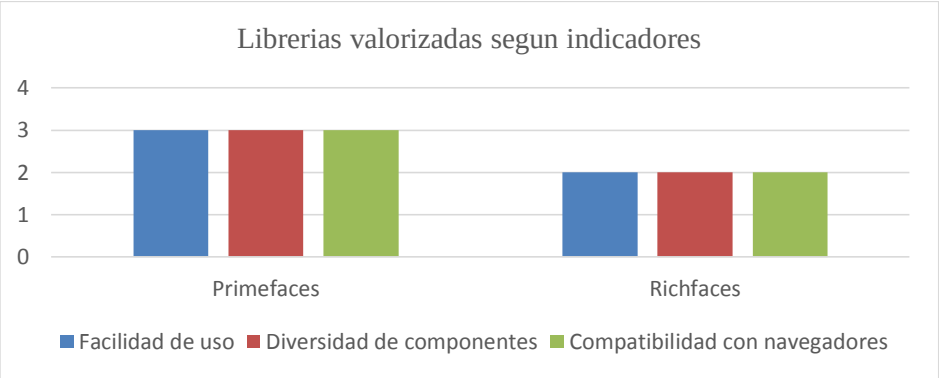


Ilustración 26: Librerías valorizadas según indicadores
Fuente: Sandra Contento/Janneth Guamán

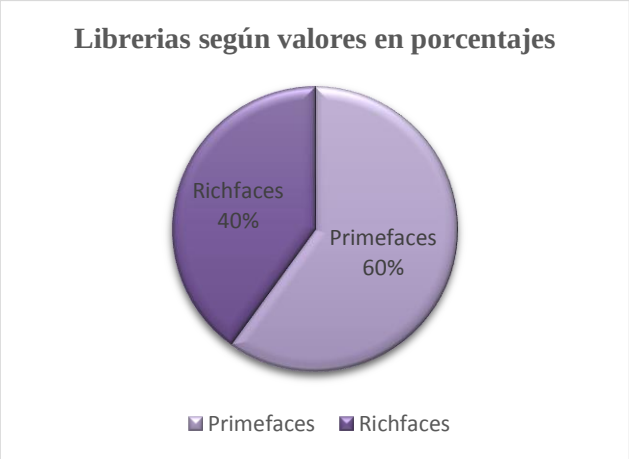


Ilustración 27: Librerías según valores en porcentajes
Fuente: Sandra Contento/Janneth Guamán

A partir de los valores se obtuvo como resultado que la librería Primefaces con el 60% es la más usada actualmente.

En la tabla 13 se muestra la respectiva comparación y valoración entre las librerías de componentes primefaces y richfaces para establecer que librería ha alcanzado la mayor madurez en el desarrollo de aplicaciones web.

Tabla 13: Madurez en el desarrollo de aplicaciones

Librería de componentes	Facilidad para iniciar	Documentación	Rendimiento	Total	%
Primefaces	3	3	3	9	64.29
Richfaces	1	2	2	5	35.71

Fuente: Sandra Contento/Janneth Guamán

De acuerdo a los resultados de la tabla, se establece la librería que ha alcanzado mayor madurez para el desarrollo de aplicaciones web, partiendo de los indicadores que permiten mayor facilidad para iniciar los componentes con un repositorio centralizado de librerías, la disponibilidad de documentación, además de mejorar el rendimiento en cuanto a tiempos de respuesta al procesar paginas, etc.

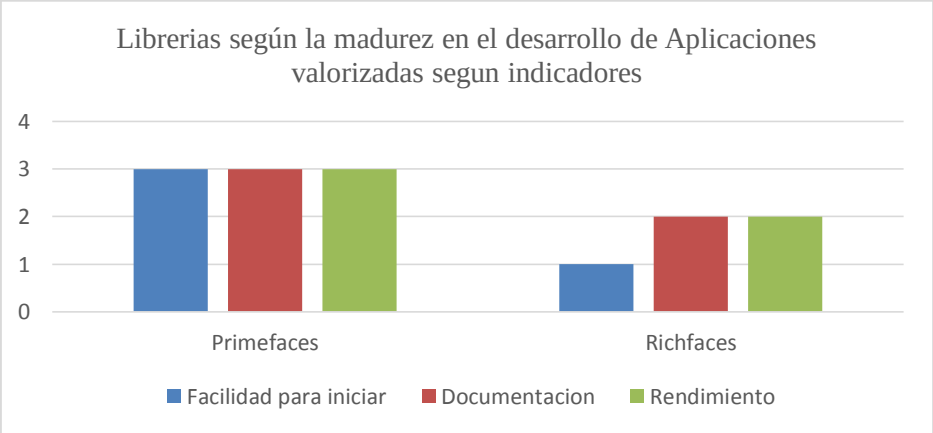


Ilustración 28: Librerías según la madurez en el desarrollo de aplicaciones según indicadores
Fuente: Sandra Contento/Janneth Guamán

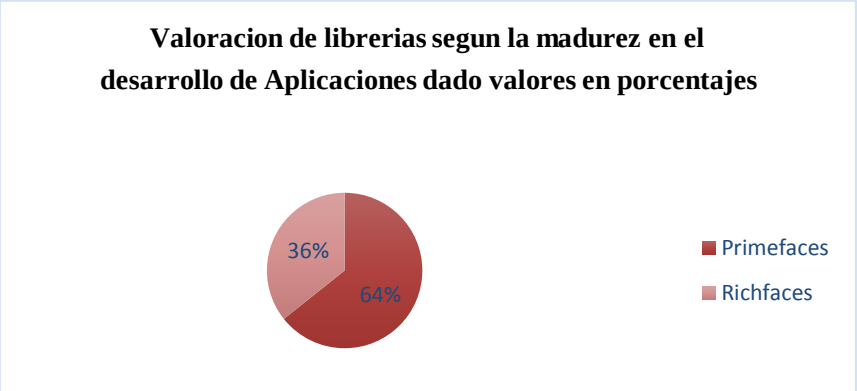


Ilustración 29: Librerías valorizadas según la madurez en el desarrollo de aplicaciones
Fuente: Sandra Contento/Janneth Guamán

Los gráficos dan a conocer que la librería Primefaces con el 64% ha alcanzado mayor madurez para el desarrollo de aplicaciones web enriquecidas.

Se concluye que la librería Primefaces es la más usada actualmente y además posee mayor madurez para ser ampliamente adoptada en el desarrollo de aplicaciones web enriquecidas.

3.2.Construcción de prototipos

3.2.1. Escenario de prueba

Se realizó dos aplicaciones, la primera utilizando la librería de componente primefaces y la segunda utilizando la librería de componentes richfaces para medir el rendimiento de cada una de las librerías. Se utilizó como herramienta de medición el programa Neoload **(Ver Anexo 8).**

3.2.2. Proceso de Prueba

Teniendo en cuenta las dos aplicaciones, se elaboró prototipos donde se utilizó el componente DataTable presente en las dos librerías. Se tomó como referencia la tabla facultad dirigiéndose a la siguiente información: código, nombre, descripción, código sicoa. El componente DataTable utiliza un Ajax para la paginación que muestra 10 facultades por página.

Al momento de realizar las pruebas con la herramienta NeoLoad, se identifica los siguientes parámetros:

- **Total throughput.-**es la suma de los tamaños de las respuestas a todas las solicitudes.

- **Average request response time.-** es el tiempo promedio de los tiempos de respuesta de las peticiones
- **Average page response.-** es el tiempo promedio de los tiempos de respuesta de las paginas.
- **Average throughput.-** es el rendimiento promedio las respuestas del servidor.

Con esta herramienta se evaluó cada prototipo. Cabe mencionar que para las pruebas tanto de Primefaces como de Richface se utilizaron 10 usuarios virtuales para medir la carga.

A continuación, se muestra los resultados en tiempo real de cada prototipo.

PRUEBA PRIMEFACES | **PRUEBA RICHFACES**

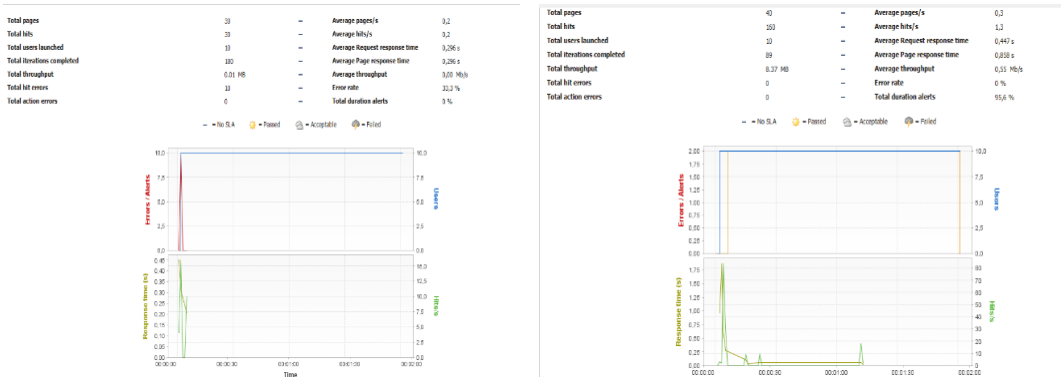


Ilustración 30: Resultado en tiempo Primefaces y RichFaces-Neoload
Fuente: Sandra Contento/Janneth Guamán

Análisis de Resultados

A partir de realizar las pruebas con las dos librerías se obtuvo los siguientes resultados:

Tiempo promedio de respuesta de Página

Neoload – Primefaces 0.296s

Neoload – Richfaces 0.858s

PRIMEFACES **RICHFACES**

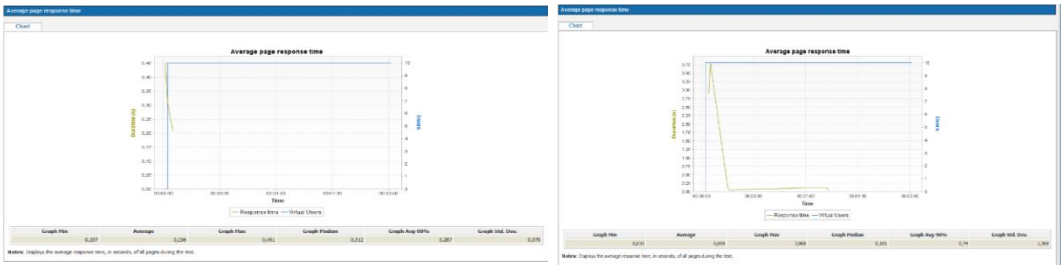


Ilustración 31 Tiempo promedio de Respuesta de Página – Primefaces y Richfaces
Fuente: Sandra Contento/Janneth Guamán

Tiempo promedio de respuesta AJAX

Neoload – Primefaces 0.296s

Neoload – Richfaces 0.447s

PRIMEFACES **RICHFACES**

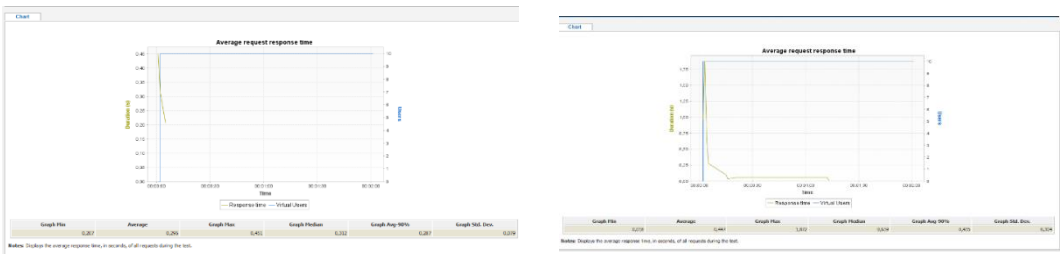


Ilustración 32 Tiempo promedio de Respuesta Ajax – Primefaces y Richfaces
Fuente: Sandra Contento/Janneth Guamán

Los datos indican en cuanto a tiempo promedio de respuesta de página Primefaces es más rápido que Richfaces porque esta librería no utiliza la comprensión a JavaScript ya que no se eliminan los caracteres innecesarios como espacios en blanco, comentarios, etc, en el tiempo de respuesta AJAX Primefaces posee un tiempo de respuesta menor que Richfaces.

Se realizó un cuadro comparativo entre las librerías Primefaces y Richfaces tanto para el tiempo promedio de respuesta de página como para el tiempo promedio de respuesta AJAX.

Tabla 14 Valores de Rendimiento

Librería de Componentes	Reducción de pagina	Reducción de tamaño AJAX	Total	Promedio %
Primefaces	3	3	6	66,67%
Richfaces	1	2	3	33,33%

Fuente: Sandra Contento/Janneth Guamán

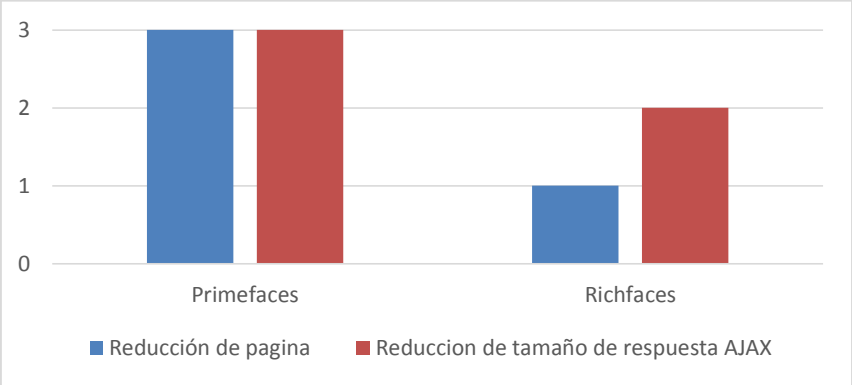


Ilustración 33 Valores de Rendimiento
Fuente: Sandra Contento/Janneth Guamán

Así después de las comparaciones valorizadas se tiene como resultado el gráfico donde porcentualmente Primefaces obtiene el mayor rendimiento con un 67%.

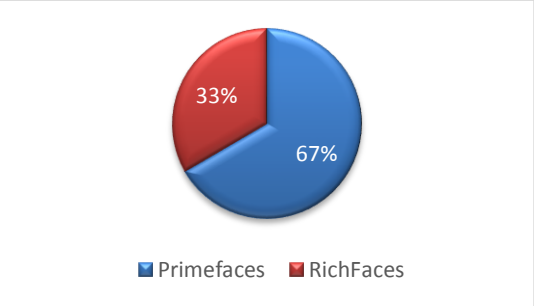


Ilustración 34 Porcentajes de valores de Rendimiento
Fuente: Sandra Contento/Janneth Guamán

Comprobación de la Hipótesis

Mediante los resultados obtenidos con la utilización de la herramienta Neoload, se puede determinar que la librería de componentes primefaces incide en un 34% más con respecto al rendimiento que la librería de componentes Richfaces. Por lo que se establece que la librería primefaces es la mejor a utilizar en el desarrollo de la aplicación SIGET ya que dispone de una variedad de componentes, documentación, compatibilidad con navegadores, facilidad de uso, facilidad para iniciar y un mayor rendimiento.

CAPITULO IV

DESARROLLO DEL SISTEMA DE GESTIÓN DE TUTORÍAS DE LA FACULTAD DE INGENIERÍA DE LA UNIVERSIDAD NACIONAL DE CHIMBORAZO

La librería de componentes más apropiada para el desarrollo de aplicaciones web con respecto al rendimiento es Primefaces. Por tal razón esta librería de componentes se utilizará para el desarrollo del Sistema de Gestión de Tutorías con una metodología de desarrollo de software eXtreme Programming (XP).

El sistema se desarrolló en la Universidad Nacional de Chimborazo, y lo utilizara la Facultad de Ingeniería, el mismo que ayudara a llevar un Control de las Tutorías planificadas y realizadas por los docentes y todos los procesos que se lleven a cabo en esta Gestión de Tutorías.

Todos estos temas, conceptos, metodologías y características serán tratados en el transcurso del presente capítulo y el desarrollo del sistema en sí.

4.1 Metodología XP

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

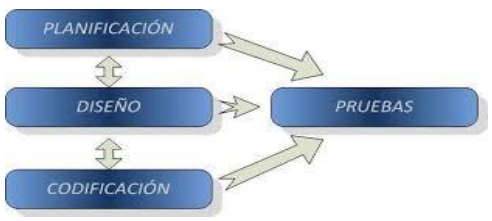


Ilustración 35: Fases de la Metodologia XP
Fuente: http://bit.ly/1dUm0Yp

Características XP

- Metodología basada en prueba y error
- Fundamentada en Valores y Prácticas
- Expresada en forma de 12 Prácticas–Conjunto completo–Se soportan unas a otras–Son conocidas desde hace tiempo. La novedad es juntarlas

Valores XP

- **Simplicidad XP.-** propone el principio de hacer la cosa más simple que pueda funcionar, en relación al proceso y la codificación. Es mejor hacer hoy algo simple, que hacerlo complicado y probablemente nunca usarlo mañana.
- **Comunicación.-** Algunos problemas en los proyectos tienen origen en que alguien no dijo algo importante en algún momento. XP hace casi imposible la falta de comunicación.
- **Realimentación.-** Retroalimentación concreta y frecuente del cliente, del equipo y de los usuarios finales da una mayor oportunidad de dirigir el esfuerzo eficientemente.
- **Coraje.-** El coraje (valor) existe en el contexto de los otros 3 valores.(si funciona mejóralo)

4.2 Desarrollo del Sistema

4.1.1. Herramientas de Desarrollo

Para la implementación del Sistema de Gestión de Tutorías se utilizará las siguientes tecnologías y herramientas.

Tabla 15: Herramientas de Desarrollo para SIGET

HERRAMIENTA	CONCEPTO	VERSION UTILIZADA
➢ **POSTGRES** 	Sus características técnicas la hacen una de las bases de datos más potentes y robustos del mercado.	Postgres9.3

➢ **NETBEANS** 	NetBeans IDE le permite desarrollar rápida y fácilmente aplicaciones de escritorio, móviles y aplicaciones web, así como aplicaciones HTML5 con HTML, JavaScript y CSS.	NetbeanIDE 8.0
➢ **SERVIDOR GLASSFISH** 	**Glassfish** es un servidor de aplicaciones que implementa la plataforma JavaEE5, por lo que soporta las últimas versiones de tecnologías como: JSP, JSF, Servlets, Servicios Web (JAX-WS), Metadatos de Servicios Web.	Glassfish 4.0
➢ **JAVE SERVER FACE** 	JSF 2.0 añade una serie de nuevas características, incluyendo soporte Ajax, la facilidad de las características de desarrollo de autoría de componente personalizado, la mejora de la gestión de configuración, un lenguaje de descripción de página, y soporte para JSR 303 Bean Validation.	JSF 2.0
➢ **LIBRERÍA PRIMEFACES** 	PrimeFaces es una librería de componentes visuales para JSF que posee un conjunto de componentes ricos facilitando la creación de las aplicaciones web.	PrimeFaces 4.0

➢ **IREPORT** 	IReport es el código abierto diseñador de informes libre para JasperReports y JasperReports Server. Crea diseños muy sofisticados. Accede a sus datos a través de JDBC, TableModels, JavaBeans, fuentes XML, Hibernate, CSV y personalizados.	IREPORT 3.6.0

Fuente: Sandra Contento/Janneth Guamán

4.1.2. Gestión del Proyecto

4.1.2.1. Planificación del Proyecto

Esta planificación del proyecto se realizó tras el estudio del problema y los requerimientos, mediante la representación de las historias se efectuó la planificación inicial la cual fue variando en el transcurso de la misma cambiando y mejorando las historias en base a concepción del problema.

4.1.2.2. Integrantes y roles

Con la participación del Director del proyecto, los miembros, los usuarios y desarrolladores, se formara el equipo encargado de la implementación del sistema.

Esto implicara que los diseños deberán ser sencillos y claros, los usuarios dispondrán de versiones de prueba del software para que puedan participar en el proceso de desarrollo mediante sugerencias y aportaciones, dicho equipo de trabajo se ve ilustrado en la Tabla 16 definiendo Integrantes y Roles.

Tabla 16: Integrantes y Roles

Miembro	Grupo	Roles XP	Metodología
Janneth Guamán	Tesista	Rastreador,Testeador,Programador	Xp
Sandra Contento	Tesista	Rastreador,Testeador,Programador	
Ing. Diego Palacios		Consultor	

Fuente: Sandra Contento/Janneth Guamán

4.1.2.3. Prototipos

Las interfaces de usuario son las más importantes ya que de esto dependerá el entendimiento fácil y rápido por parte del usuario al comenzar a manejar el sistema.

Se pretende que la interfaz del usuario sea amigable, sencilla y funcional con un alto grado de comprensión, por tal razón se crearon los prototipos generales del sistema.

A continuación se realizara una breve descripción del proceso principal.

Inicio de Sesión de usuarios:

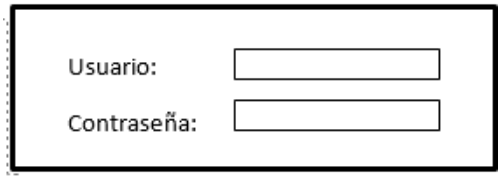


Ilustración 36 Inicio de Sesión
Fuente: Sandra Contento/Janneth Guamán

En la figura se muestra la creación de los módulos del sistema

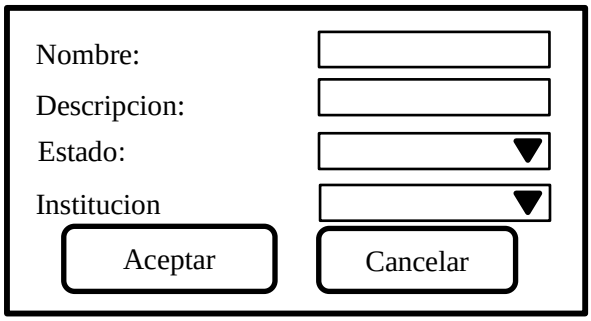


Ilustración 37 Creacion de Módulos del Sistema
Fuente: Sandra Contento/Janneth Guamán

Una pantalla para asignar roles al usuario.

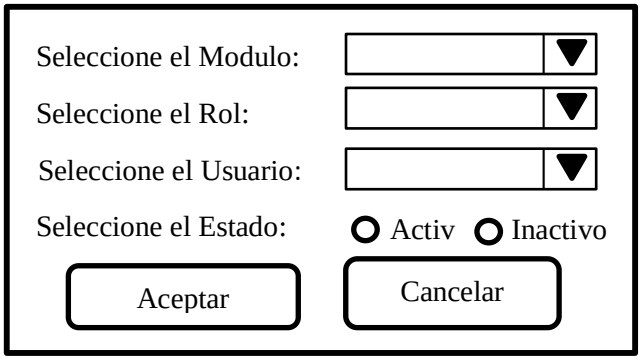


Ilustración 38 Asignación de roles al usuario
Fuente: Sandra Contento/Janneth Guamán

Una pantalla para asignar al tutor un nivel y un periodo académico.

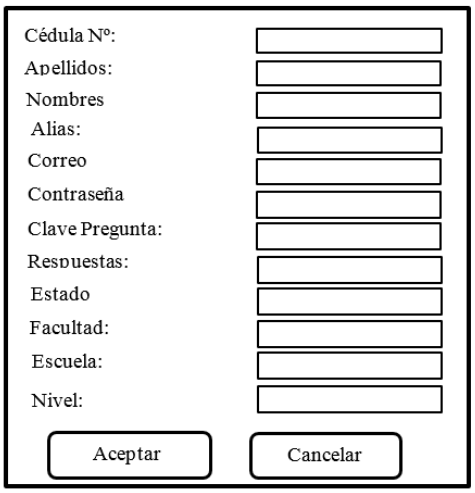


Ilustración 39 Crear Tutor
Fuente: Sandra Contento/Janneth Guamán

En la figura se muestra la creación de la planificación de una actividad donde el rol del docente sea tutor.

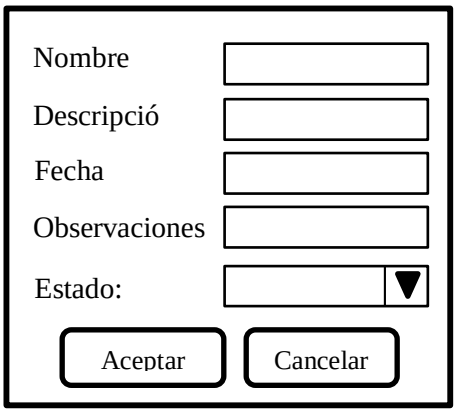


Ilustración 40 Creación de la planificación de una actividad rol tutor
Fuente: Sandra Contento/Janneth Guamán

Una pantalla donde se muestre el encabezado y el detalle de la actividad realizada

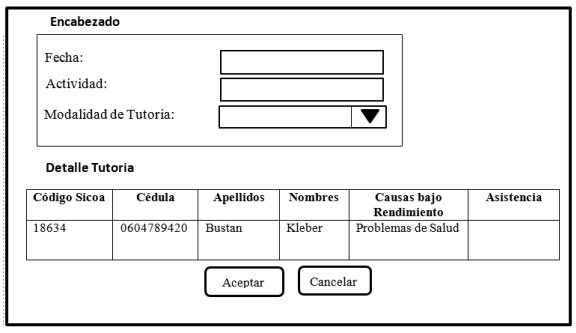


Ilustración 41 Detalle de la Actividad Realizada
Fuente: Sandra Contento/Janneth Guamán

4.1.2.4. Historias de Usuarios

Las historias de los usuarios tienen como finalidad ver las necesidades del sistema por lo tanto se realizaran descripciones cortas y escritas en el lenguaje del usuario sin terminología, detallando el tiempo que conllevara la implementación así como la estimación del riesgo de dicha historia de usuario.

Cada historia de usuario se divide en actividades planificables y medibles para su realización, estas forman una interacción, este plan nos indicara diferentes interacciones del sistema. La realización de este plan debe tener muy en cuenta la prioridad de los usuarios para satisfacerles en mayor medida como se muestra en la tabla.

Tabla 17: Historias de Usuarios

Nº	NOMBRE	PRIORIDAD	RIESGO	ESFUERZO	ITERACION
1	Gestión de Acceso de Usuarios	Alta	Alto	Medio	1
2	Gestión de la Seguridad	Alta	Medio	Medio	1
3	Gestión de Facultades	Medio	Medio	Bajo	1
4	Gestion de Carreras	Medio	Medio	Bajo	1
5	Gestión de Niveles	Medio	Medio	Bajo	1
6	Gestión de Periodos	Medio	Medio	Bajo	1
7	Gestión de Tutores	Alta	Alto	Alto	1
8	Gestión de Actividades Planificadas	Medio	Medio	Bajo	2
9	Gestión de Actividades Realizadas	Alta	Alto	Alto	2
10	Emisión de Reportes	Alta	Medio	Alto	3

Fuente: Sandra Contento/Janneth Guamán

4.1.2.5. Plan de Entregas

Con cada historia de usuario previamente evaluada en tiempo de desarrollo ideal, el usuario agrupara en orden de importancia.

Para realizar el plan de entregas se hará en función de dos parámetros: tiempo de desarrollo y grado de importancia para el usuario. Las iteraciones individuales son

planificadas en detalle antes de que comience cada iteración como se puede apreciar en la tabla y figuras.

Iteración 1

Tabla 18: Plan de Entrega Iteración 1

Historia de Usuario	Duración en semanas
Gestión de Acceso de Usuarios	2
Gestión de la Seguridad	3
Gestión de Facultades	1
Gestión de Carreras	1
Gestión de Niveles	1
Gestión de Periodos	1
Gestión de Tutores	3

Fuente: Sandra Contento/Janneth Guamán

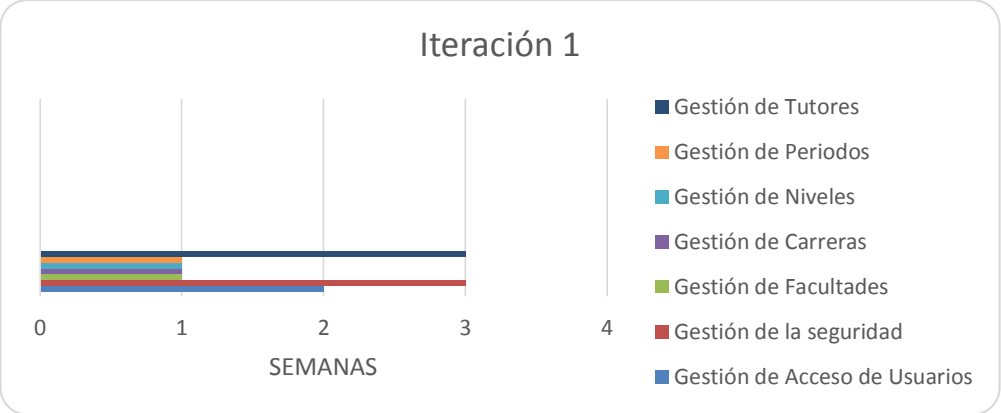


Ilustración 42 Plan de Entrega Iteracion 1
Fuente: Sandra Contento/Janneth Guamán

Iteración 2

Tabla 19: Plan de Entrega Iteracion 2

Historia de Usuario	Duración en semanas
Gestión de Actividades Planificadas	2
Gestión de Actividades Realizadas	3

Fuente: Sandra Contento/Janneth Guamán

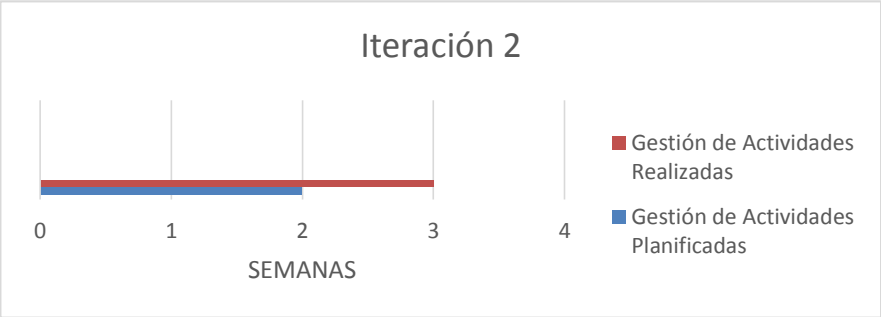


Ilustración 43: Plan de Entrega Iteración 2
Fuente: Sandra Contento/Janneth Guamán

Iteración 3

Tabla 20: Plan de Entrega Iteración 3

Historia de Usuario	Duración en semanas
Emisión de Reportes	3

Fuente: Sandra Contento/Janneth Guamán

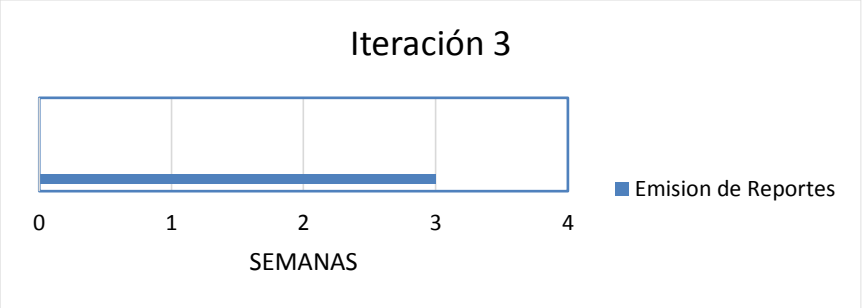


Ilustración 44: Plan de Entregas Iteración 3
Fuente: Sandra Contento/Janneth Guamán

4.1.2.6. Incidencia

Iteración primera: se tratara de tener preparadas las funcionalidades básicas con el usuario, las herramientas y la unificación de horarios para el desarrollo del sistema. Se realizó un prototipo de la base datos acorde al reglamento de tutorías de la UNACH **(Ver Anexo 9)** certificado por las autoridades para luego implementarla en el motor de base Postgresql, a continuación se realizó la creación de las funciones básicas como son: insertar, actualizar, eliminar y seleccionar de acuerdo a las necesidades que se requieren para el desarrollo del sistema. Una vez creadas ya las funciones en la base de datos se procedió a crear un package denominado

nombre_esquema_BD.logica.clases en Java para todas las tablas que se realizaron en postgresql con sus respectivos atributos, constructores y métodos get() y set(). A continuación se procedió a crear un package denominado nombre_esquema_BD.logica.funciones en Java que contiene las funciones insertar, actualizar, eliminar y seleccionar de cada tabla. El siguiente paso fue crear un package denominado nombre_esquema_BD.logica.beans en Java que contiene todas las funciones que se requieren implementar para mostrar en la vista. Y por último se creó carpetas con los esquemas de la base de datos para la creación de las respectivas vistas de cada tabla.

Iteración segunda: Como el sistema requiere consumir servicios web se procedió a realizar el trámite correspondiente con el respectivo formato de solicitud **(VER ANEXO 5)** indicando las especificaciones técnicas de los servicios web, este proceso tuvo una duración de aproximadamente 4 semanas. Para la entrega de la dirección de los servicios web el departamento de UTECA realizó la respectiva acta entrega-recepción **(VER ANEXO 6)** para poder empezar a trabajar en el consumo de los servicios web. Se modificaron las respectivas vistas insertando la opción migrar datos para las tablas usuario, facultad, carrera, nivel y periodo obteniendo datos reales desde la base de datos del Sistema de Control Académico SICOA a la base de datos SIGET.

Una vez consumidos los servicios web, la visión de los miembros del equipo puede ser interpretada de distinta forma que la del usuario es por ello que es importante las sugerencias del usuario final, por ende se sugirió realizar una vista para que el estudiante pueda conocer las actividades que se desarrollan durante el periodo vigente, además podrá visualizar un historial de todas las actividades realizadas en los distintos periodos académicos.

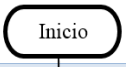
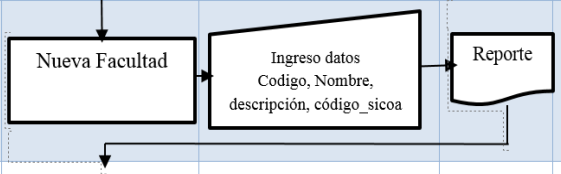
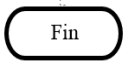
Iteración tercera: Con esta iteración se pretende generar los respectivos reportes para las diferentes actividades que se requiere de acuerdo al tipo de rol.

Por requerimiento del director de escuela ante un reporte que contenga la nómina de estudiantes con su respectivo promedio general dado un nivel, carrera y periodo se procedió a realizar la respectiva solicitud del nuevo método **(VER ANEXO 7)**

4.1.2.7. Actividades

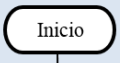
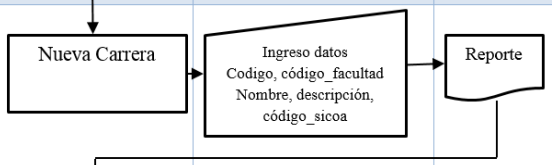
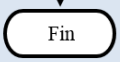
Las actividades del sistema fueron divididas en varios procesos que serán reflejados mediante flujos de procesos.

Tabla 21: Proceso Nueva Facultad

PROCESO NUEVA FACULTAD						
	Actividad	**Flujograma**	**IN**	**OUT**	**Responsable**	**Observación**
	Inicio					
1	Actividad				Administrador Master	
	Fin					

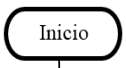
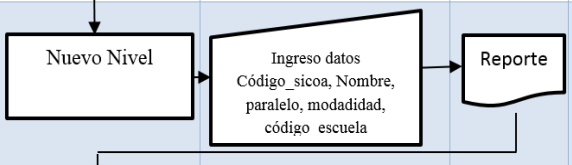
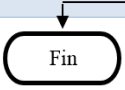
Fuente: Sandra Contento/Janneth Guamán

Tabla 22: Proceso Nueva Carrera

PROCESO NUEVA CARRERA						
	Actividad	**Flujograma**	**IN**	**OUT**	**Responsable**	**Observación**
	Inicio					
1	Actividad				Administrador Master	
	Fin					

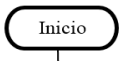
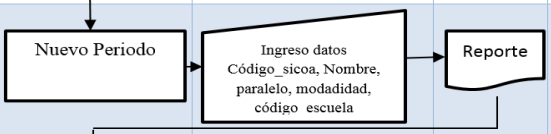
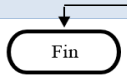
Fuente: Sandra Contento/Janneth Guamán

Tabla 23: Proceso nuevo nivel

PROCESO NUEVO NIVEL						
	Actividad	**Flujograma**	**IN**	**OUT**	**Responsable**	**Observación**
	Inicio					
1	Actividad				Administrador Master	
	Fin					

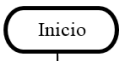
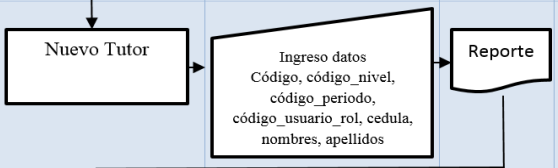
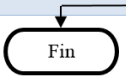
Fuente: Sandra Contento/Janneth Guamán

Tabla 24: Proceso Nuevo Periodo

PROCESO NUEVO PERIODO						
	Actividad	**Flujograma**	**IN**	**OUT**	**Responsable**	**Observación**
	Inicio					
1	Actividad				Administrador Master	
	Fin					

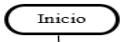
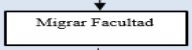
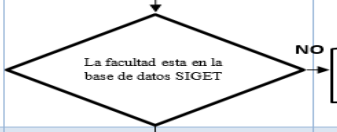
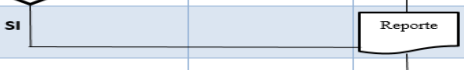
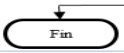
Fuente: Sandra Contento/Janneth Guamán

Tabla 25: Proceso Nuevo Tutor

PROCESO NUEVO TUTOR						
	Actividad	**Flujograma**	**IN**	**OUT**	**Responsable**	**Observación**
	Inicio					
1	Actividad				Administrador Master, Directivo	
	Fin					

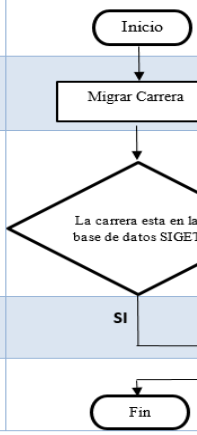
Fuente: Sandra Contento/Janneth Guamán

Tabla 26: Procesos Migrar Facultades

PROCESO MIGRAR FACULTADES						
	Actividad	**Flujograma**	**IN**	**OUT**	**Responsable**	**Observación**
	Inicio					
1					Administrador Master	
2					Administrador Master	
3					Administrador Master	
	Fin				Administrador Master	

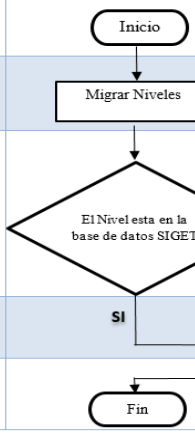
Fuente: Sandra Contento/Janneth Guamán

Tabla 27 Proceso Migrar Carreras

PROCESO MIGRAR CARRERAS						
	Actividad	Flujograma	IN	OUT	Responsable	Observación
	Inicio					
1					Administrador Master	
2					Administrador Master	
3					Administrador Master	
	Fin					

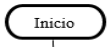
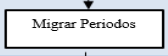
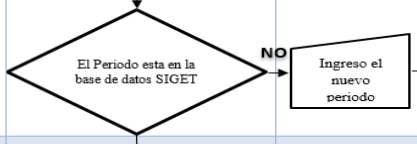
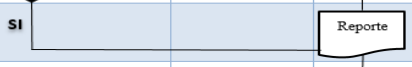
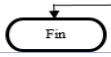
Fuente: Sandra Contento/Janneth Guamán

Tabla 28 Proceso Migrar Niveles

PROCESO MIGRAR NIVELES						
	Actividad	Flujograma	IN	OUT	Responsable	Observación
	Inicio					
1					Administrador Master	
2					Administrador Master	
3					Administrador Master	
	Fin					

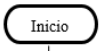
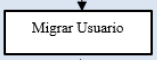
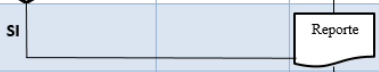
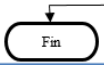
Fuente: Sandra Contento/Janneth Guamán

Tabla 29 Proceso Migrar Periodos

PROCESO MIGRAR PERIODOS						
	Actividad	Flujograma	IN	OUT	Responsable	Observación
	Inicio					
1					Administrador Master	
2					Administrador Master	
3					Administrador Master	
	Fin					

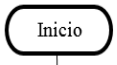
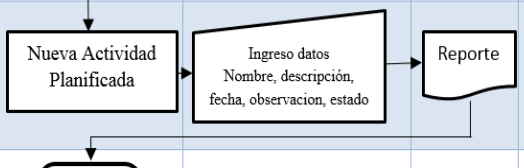
Fuente: Sandra Contento/Janneth Guamán

Tabla 30 Proceso Migrar Usuario Docente

PROCESO MIGRAR USUARIO-DOCENTE						
	Actividad	Flujograma	IN	OUT	Responsable	Observación
	Inicio					
1					Administrador Master	
2					Administrador Master	
3					Administrador Master	
	Fin				Administrador Master	

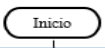
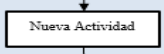
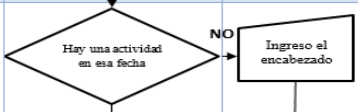
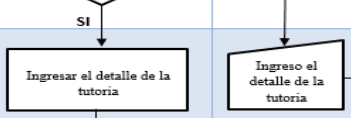
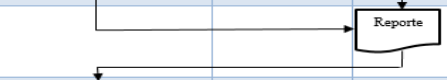
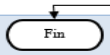
Fuente: Sandra Contento/Janneth Guamán

Tabla 31 Proceso Actividades Planificadas

PROCESO ACTIVIDADES PLANIFICADAS						
	Actividad	**Flujograma**	**IN**	**OUT**	**Responsable**	**Observación**
	Inicio					
1	Actividad				Tutor	
	Fin					

Fuente: Sandra Contento/Janneth Guamán

Tabla 32 Proceso Actividades Realizadas

PROCESO ACTIVIDADES REALIZADAS						
	Actividad	**Flujograma**	**IN**	**OUT**	**Responsable**	**Observación**
	Inicio					
1					Tutor	
2					Tutor	
3					Tutor	
4					Tutor	
	Fin					

Fuente: Sandra Contento/Janneth Guamán

4.1.3. Implementación

4.1.3.1. Base de Datos

Nuestra base de datos consta de 3 esquemas:

Master: En este esquema se maneja todo lo concerniente con la seguridad, creación de menús, y todo lo que tiene que ver con los roles, tutores, facultades, carreras, niveles, periodos.

Planificación: en este esquema se encuentran las actividades planificadas por el tutor.

Actividades: en este esquema se encuentra todo lo referente a las actividades realizadas y su respectivo detalle de la tutoría y las causas del bajo rendimiento de estudiante.

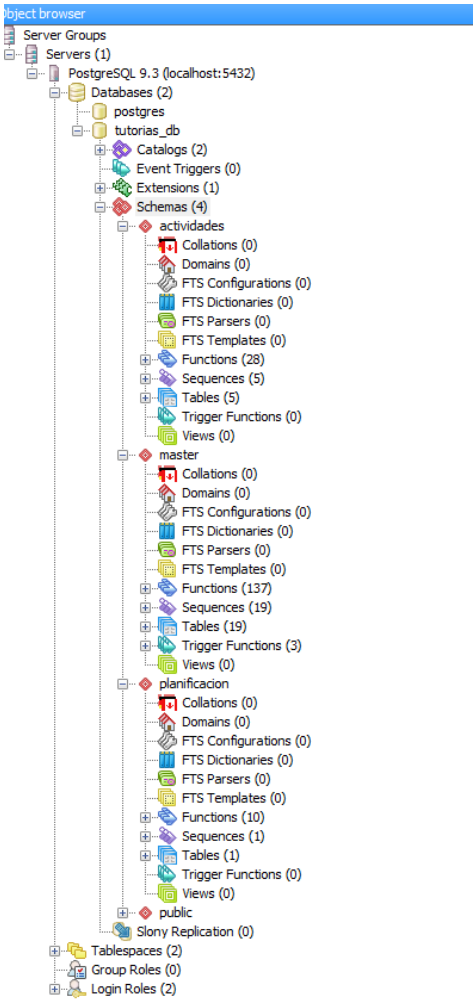


Ilustración 45 Esquema Base de Datos-Postgresql
Fuente: Sandra Contento/Janneth Guamán

DIAGRAMA ENTIDAD RELACION DE LA BASE DE DATOS SIGET

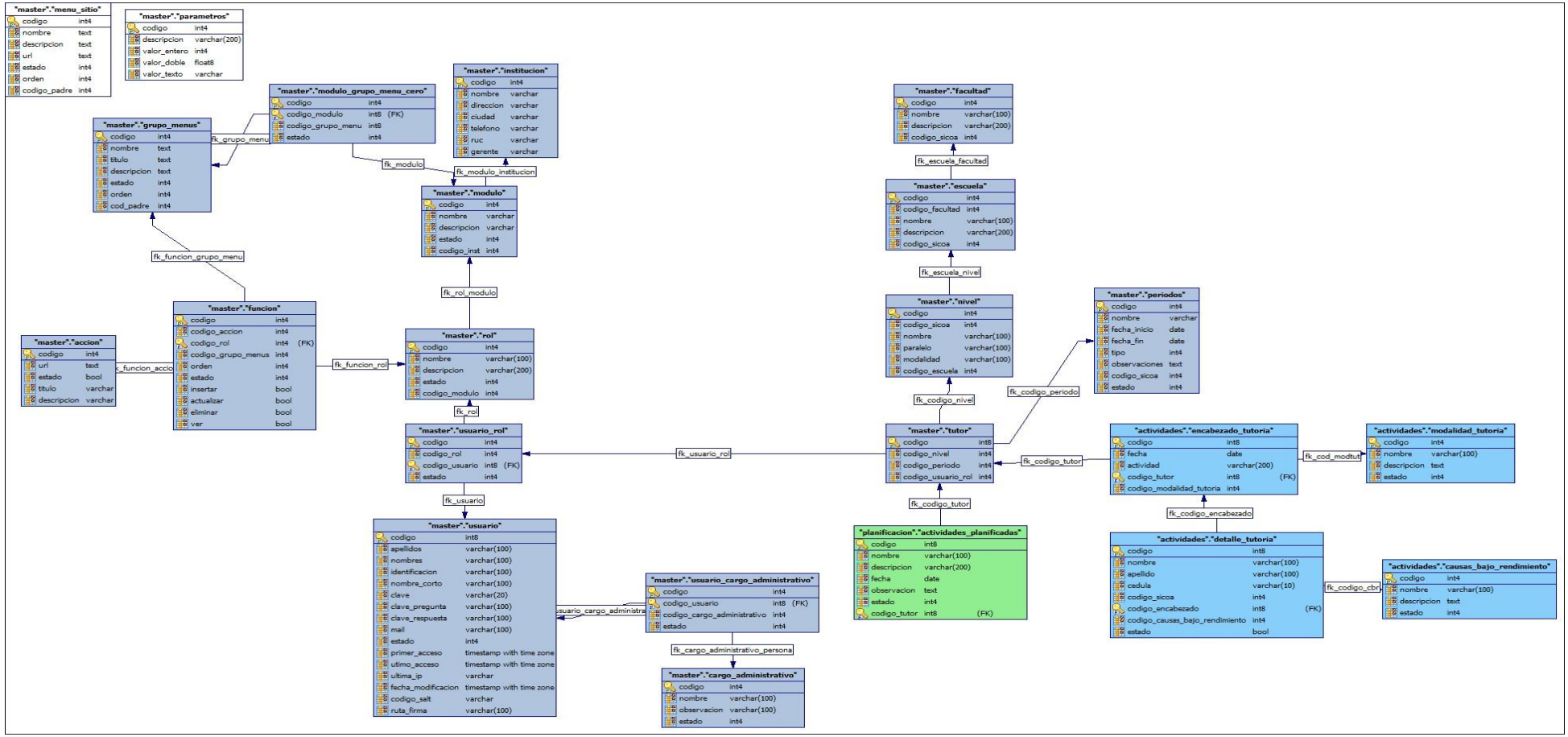


Ilustración 46 Diagrama Entidad Relacional BD SIGET
Fuente: Sandra Contento/Janneth Guamán

4.1.3.2. Diccionario de Datos

El Diccionario de datos permite guardar la estructura de la base de datos, es decir se define como se almacena y accede a la información.

- **NOMBRE DE LA TABLA: rol,** es la tabla que permite crear nuevos roles.

Tabla 33: Rol

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int4	SI	NO	SI
nombre	varchar	NO	SI	NO
descripcion	varchar	NO	SI	NO
estado	int4	NO	SI	NO
código_modulo	int4	NO	NO	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: usuario,** es la tabla que permite registrar datos de un nuevo usuario.

Tabla 34: Usuario

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int8	SI	NO	SI
apellidos	varchar	NO	SI	NO
nombres	varchar	NO	SI	NO
identificación	varchar	NO	SI	NO
nombre_corto	varchar	NO	SI	NO
clave	varchar	NO	SI	NO
clave_pregunta	varchar	NO	SI	NO
clave_respuesta	varchar	NO	SI	NO
mail	varchar	NO	SI	NO
estado	int4	NO	SI	NO
primer_acceso	timestamp with time zone	NO	SI	NO

ultimo_acceso	timestamp with time zone	NO	SI	NO
ultima_ip	varchar	NO	SI	NO
fecha_modificacion	timestamp with time zone	NO	SI	NO
código_salt	varchar	NO	SI	NO
ruta_firma	varchar	NO	SI	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: usuario_rol,** es la tabla que permite registrar el rol y el usuario.

Tabla 35: Usuario_Rol

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
Código	int4	SI	NO	SI
código_rol	int4	NO	NO	NO
código_usuario	int8	NO	NO	NO
Estado	int4	NO	SI	NO

Fuente: Sandra Contento/Janneth Guamán

NOMBRE DE LA TABLA: facultad, es la tabla que permite registrar todas las facultades que se encuentran en la institución.

Tabla 36: Facultad

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int4	SI	NO	NO
nombre	varchar	NO	SI	NO
descripción	varchar	NO	SI	NO
código_sicoa	int4	NO	SI	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: carreras,** es la tabla que permite registrar todas las carreras que se encuentran distribuidas en cada facultad de la institución.

Tabla 37: Carreras

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int4	SI	NO	NO
código_facultad	int4	NO	NO	NO
nombre	varchar	NO	SI	NO
descripción	varchar	NO	SI	NO
código_sicoa	int4	NO	SI	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: nivel,** es la tabla que permite registrar todas los niveles ya sea anules o semestrales de las distintas carreras de la institución.

Tabla 38: Nivel

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int4	SI	NO	NO
código_sicoa	int4	NO	SI	NO
nombre	varchar	NO	SI	NO
paralelo	varchar	NO	SI	NO
modalidad	varchar	NO	SI	NO
código_escuela	int4	NO	NO	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: periodo,** es la tabla que permite el registro de las fechas de inicio y fin de los periodos académicos de la institución.

Tabla 39: Periodo

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int4	SI	NO	NO
nombre	varchar	NO	SI	NO
fecha_inicio	int8	NO	SI	NO
fecha_fin	int8	NO	SI	NO
Tipo	int4	NO	SI	NO
observaciones	text	NO	SI	NO
código_sicoa	int4	NO	Si	NO
estado	int4	NO	SI	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: tutor,** es la tabla que permite el registro de las relaciones entre los datos del docente asignado como tutor a una determinado nivel en un periodo establecido.

Tabla 40: Tutor

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int8	SI	NO	SI
código_nivel	int4	NO	NO	NO
código_periodo	int4	NO	NO	NO
código_usuario_rol	int4	NO	NO	NO

Fuente: Sandra Contento/Janneth Guamán

NOMBRE DE LA TABLA: encabezado_tutoría esta tabla guarda la relación entre el tutor y la modalidad de la tutoría, así como la actividad y la fecha del encabezado de la tutoría.

Tabla 41: Encabezado de la Tutoría

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int8	SI	NO	SI
fecha	date	NO	SI	NO
actividad	varchar	NO	SI	NO

codigo_tutor	int8	NO	NO	NO
codigo_modalidad_tutoria	int4	NO	NO	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA:** modalidad_tutoria esta tabla permite almacenar la información de la modalidad de estudios, así como el estado del periodo académico.

Tabla 42: Modalidad de la Tutoría

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int4	SI	NO	SI
nombre	varchar	NO	SI	NO
descripción	text	NO	SI	NO
estado	int4	NO	NO	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: causas_bajo_rendimiento** esta tabla permite almacenar la información de los problemas que afectan el desempeño de los estudiantes.

Tabla 43: Causas del Bajo Rendimiento

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int4	SI	NO	SI
nombre	varchar	NO	SI	NO
descripción	text	NO	SI	NO
estado	int4	NO	NO	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: detalle_tutoria** esta tabla permite almacenar información detallada de la actividad realizada, también permite almacenar los datos del estudiante y las causas del bajo rendimiento académico.

Tabla 44: Detalle de la Tutoría

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int8	SI	NO	SI
nombre	varchar	NO	SI	NO
apellido	varchar	NO	SI	NO
cedula	varchar	NO	SI	NO
codigo_sicoa	int4	NO	NO	NO
codigo_encabezado	int4	NO	NO	NO
codigo_causas_bajo_rendimiento	int4	NO	NO	NO

Fuente: Sandra Contento/Janneth Guamán

- **NOMBRE DE LA TABLA: actividades_planificadas** esta tabla permite elaborar la planificación de las actividades de la tutoría, así como la fecha de la tutoría y el tutor que dicta la tutoría.

Tabla 45: Actividades Planificadas

Nombre de la Columna	Tipo de Dato	Clave Primaria	Valores Nulos	Auto Incremental
código	int8	SI	NO	SI
nombre	varchar	NO	SI	NO
descripción	varchar	NO	SI	NO
fecha	date	NO	SI	NO
observación	text	NO	SI	NO
estado	int4	NO	SI	NO
codigo_tutor	int8	NO	NO	NO

Fuente: Sandra Contento/Janneth Guamán

INDICES

A continuación se visualiza los índices de las tablas y la columna de cada uno de ellos.

Tabla 46: Claves principales de Tablas de la base de datos SIGET

NOMBRE DE LA TABLA	NOMBRE DEL INDICE	COLUMNA
master.institucion	pk_cod_institucion	codigo
master.grupo_menus	pk_tipo_cont	codigo
master.accion	pk_accion_codigo	codigo
master.modulo	pk_cod_modulo	codigo
master.rol	pk_cod_rol	codigo
master.funcion	pk_codigo_funcion	codigo
master.modulo_grupo_menu_cero	pk_codigo_modulo_grupo_menu_cero	codigo
master.usuario	pk_cod_usuario	codigo
master.usuario_rol	pk_cod_asig_rol	codigo
master.cargo_administrativo	pk_cargo	codigo
master.usuario_cargo_administrativo	pk_codigo	codigo
master.menu_sitio	pk_menu_sitio_codigo	codigo
master.parametros	pk_codigo_parametro	codigo
master.facultad	pk_facultades	codigo
master.escuela	pk_codigo_escuela	codigo
master.nivel	pk_codigo_nivel	codigo
master.periodo	pk_codigo_periodo	codigo
master.estudiante	pk_codigo_estudiante	codigo
master.tutor	pk_codigo_tutor	codigo
planificacion.actividades_planificadas	pk_cod_act_pla	codigo
actividades.encabezado_tutoria	pk_enc_tut	codigo
actividades.detalle_tutoria	pk_cod_dettut	codigo
actividades.modalidad_tutoria	pk_cod_modtut	codigo
actividades.causas_bajo_rendimiento	pk_cod_causas_rendimiento	codigo

Fuente: Sandra Contento/Janneth Guamán

CLAVES FORANEAS

A continuación se visualiza las claves foráneas de las tablas utilizadas e información relevante de esas claves.

Tabla 47: Claves foráneas de Tablas de la base de datos SIGET

NOMBRE	TABLA FUENTE	TABLA DESTINO	ENLACE
fk_grupo_menu	master.modulo_grupo_menu_cero	master.grupo_menus	código=codigo_grupo_menu
fk_modulo	master.modulo_grupo_menu_cero	master.modulo	código=código_modulo
fk_modulo_institucion	master.modulo	master.institucion	código=codigo_inst
fk_funcion_accion	master.funcion	master.accion	código=código_accion
fk_funcion_grupo_menu	master.funcion	master.grupo_menus	código=código_grupo_menus
fk_funcion_rol	master.funcion	master.rol	código=código_rol
fk_rol_modulo	master.rol	master.modulo	código=código_modulo
fk_rol	master.usuario_rol	master.rol	código=código_rol
fk_usuario	master.usuario_rol	master.usuario	código=código_usuario
fk_cargo_administrativo_persona	master.usuario_cargo_administrativo	master.cargo_administrativo	código=código_cargo_administrativo
fk_usuario_cargo_administrativo	master.usuario_cargo_administrativo	master.usuario	código=código_usuario

fk_escuela_nivel	master.nivel	master.escuela	código=código_escuela
fk_codigo_nivel	master.estudiante	master.nivel	código=código_nivel
fk_codigo_periodo	master.estudiante	master.periodo	código=código_periodo
fk_escuela_facultad	master.escuela	master.facultad	codigo=código_facultad
fk_codigo_nivel	master.tutor	master.nivel	codigo=código_nivel
fk_codigo_periodo	master.tutor	master.periodo	código=código_periodo
fk_usuario_rol	master.tutor	master.usuario_rol	código=código_usuario_rol
fk_codigo_tutor	planificación.actividades_planificadas	master.tutor	código=código_tutor
fk_cod_modttut	actividades.encabezado_tutoria	actividades.modalidad_tutoria	código=código_modalidad_tutoria
fk_codigo_tutor	actividades.encabezado_tutoria	master.tutor	código=código_tutor
fk_codigo_cbr	actividades.detalle_tutoria	actividades.causas_bajo_rendimiento	codigo=código_causas_bajo_rendimiento
fk_codigo_encabezado	actividades.detalle_tutoria	actividades.encabezado_tutoria	código=código_encabezado

Fuente: Sandra Contento/Janneth Guamán

FUNCIONES

f_select_escuela: esta función se ha creado para obtener todos los datos de las diferentes carreras.

Tabla 48: Función Seleccionar Carreras

Nombre de Parámetro	Tipo de Parámetro	Tipo de Dato
pcódigo	OUT	int4
pcódigo_facultad	OUT	int4
pnombre	OUT	varchar
pdescripción	OUT	varchar
pcódigo_sicoa	OUT	int4

Fuente: Sandra Contento/Janneth Guamán

f_select_facultad: esta función se ha creado para obtener todos los datos de la facultad.

Tabla 49: Función Seleccionar facultades

Nombre de Parámetro	Tipo de Parámetro	Tipo de Dato
pcódigo	OUT	int4
pnombre	OUT	varchar
pdescripción	OUT	varchar
pcódigo_sicoa	OUT	int4

Fuente: Sandra Contento/Janneth Guamán

f_select_periodo: esta función se ha creado para obtener todos los datos de los diferentes periodos.

Tabla 50: Función Seleccionar periodos

Nombre de Parámetro	Tipo de Parámetro	Tipo de Dato
pcódigo	OUT	int4
pnombre	OUT	varchar
pfecha_inicio	OUT	int8
pfecha_fin	OUT	int8
ptipo	OUT	int4
pobservaciones	OUT	text
pcódigo_sicoa	OUT	int4
pestado	OUT	int4

Fuente: Sandra Contento/Janneth Guamán

f_select_roles (): esta función se ha creado para obtener todos los datos de los distintos roles.

Tabla 51: Función Seleccionar roles

Nombre de Parámetro	Tipo de Parámetro	Tipo de Dato
pcódigo	OUT	int4
pnombre	OUT	varchar
pdescripcion	OUT	varchar
pestado	OUT	int4
pcódigo_modulo	OUT	int4

Fuente: Sandra Contento/Janneth Guamán

f_select_tutor (): esta función se ha creado para obtener todos los datos de los diferentes tutores.

Tabla 52: Función Seleccionar Tutor

Nombre de Parámetro	Tipo de Parámetro	Tipo de Dato
pcódigo	OUT	int8
pcódigo_nivel	OUT	int4
pcódigo_periodo	OUT	int4
pcódigo_usuario_rol	OUT	int4

Fuente: Sandra Contento/Janneth Guamán

f_select_usuarios (): esta función se ha creado para obtener todos los datos de los distintos usuarios del sistema.

Tabla 53: Función Seleccionar Usuarios

Nombre de Parámetro	Tipo de Parámetro	Tipo de Dato
pcódigo	OUT	int8
papellidos	OUT	Varchar
pnombres	OUT	Varchar
pidentificación	OUT	Varchar
pnombre_corto	OUT	Varchar
pclave	OUT	Varchar
pclave_pregunta	OUT	Varchar
pclave_respuesta	OUT	Varchar
pmail	OUT	Varchar

pestado	OUT	int4
pprimer_acceso	OUT	timestamp with time zone
pultimo_acceso	OUT	timestamp with time zone
pultima_ip	OUT	varchar
pfecha_modificacion	OUT	timestamp with time zone
pcódigo_salt	OUT	varchar
pruta_firma	OUT	varchar

Fuente: Sandra Contento/Janneth Guamán

f_select_usuarios_rol (): esta función se ha creado para obtener todos los datos de los diferentes roles del usuario.

Tabla 53: Función Seleccionar Roles del usuario

Nombre de Parámetro	Tipo de Parámetro	Tipo de Dato
pcódigo	OUT	int4
pcódigo_rol	OUT	int4
pcódigo_usuario	OUT	int8
pestado	OUT	int4

Fuente: Sandra Contento/Janneth Guamán

4.1.3.3. Prototipos interfaces de usuario finales

Con la descripción detallada de las historias de tutores, actividades planificadas y con los diagramas de procesos podemos definir interfaces de usuario finales, las cuales serán implantadas en el sistema.

Ilustración 47 Control de Acceso de Usuarios
Fuente: Sandra Contento/Janneth Guamán

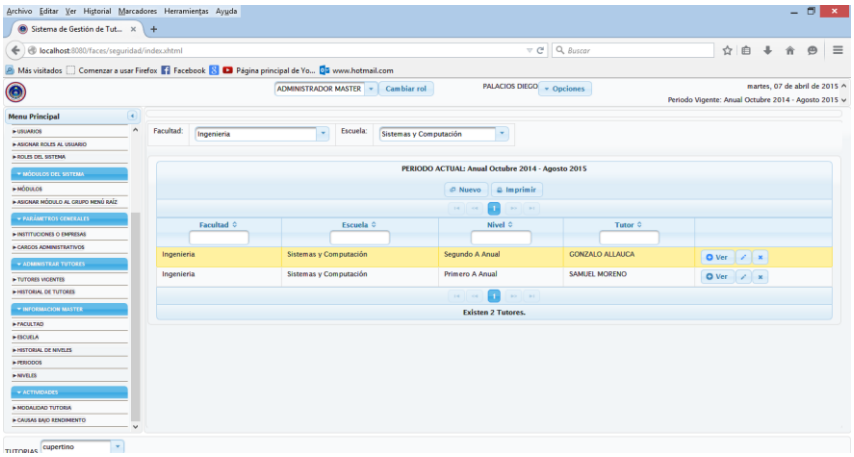


Ilustración 48 Administrador Master – Gestión Tutores
Fuente: Sandra Contento/Janneth Guamán

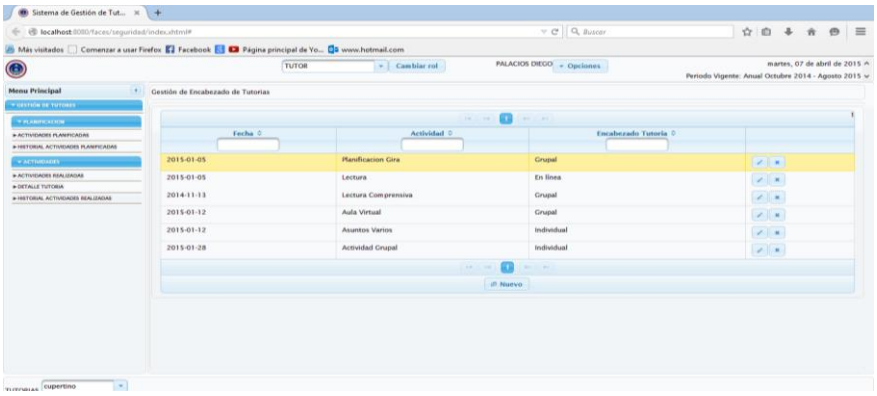


Ilustración 49 Actividades Realizadas por el Tutor
Fuente: Sandra Contento/Janneth Guamán

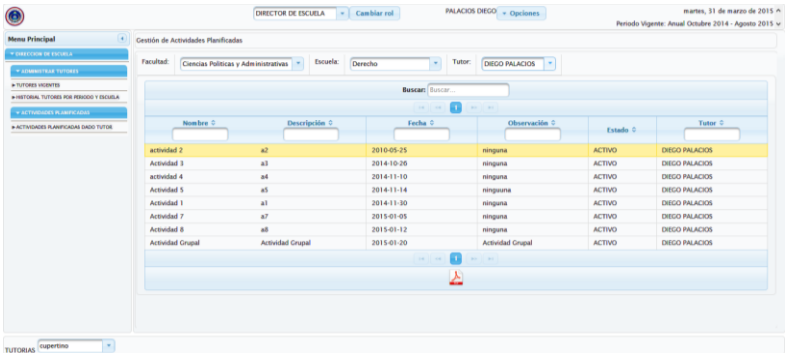
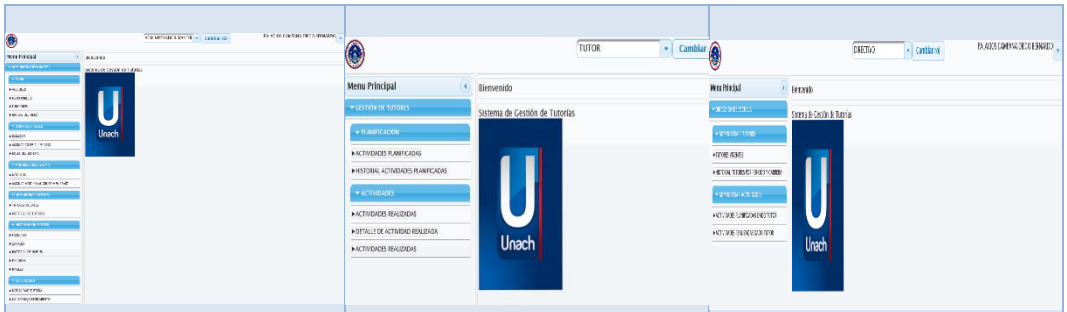


Ilustración 50 Actividades Planificadas por el tutor – Directivo
Fuente: Sandra Contento/Janneth Guamán

Iteración 1

Tabla 54: Iteración 1 Historia 1

Historia de Usuario	
Numero:1	Usuario: Administrador Master, Tutor, Directivo
Nombre historia: Gestión de Acceso de Usuarios.	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alto
Esfuerzo: Medio	Iteración asignada 1
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: Antes de iniciar el sistema se requiere la cuenta de usuario y la contraseña para poder acceder a los datos de acuerdo al tipo de usuario.	
Observaciones: Hay tres tipos de usuarios: Administrador Master, Tutor, Directivo con distintos menús y permisos de acceso dependiendo de las funciones de los usuarios.	
 **Gestión de Acceso de Usuarios**	

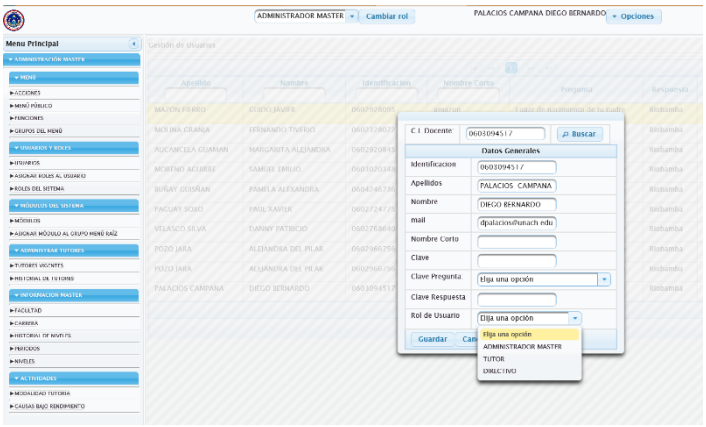


Administrador Master	Tutor	Directivo

Fuente: Sandra Contento/Janneth Guamán

Tabla 55: Iteración 1 Historia 2

Historia de Usuario	
Numero:2	**Usuario:** Administrador Master
Nombre historia: Gestión de la Seguridad.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Esfuerzo: Medio	Iteración asignada 1
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El Administrador Master una vez que ingrese al sistema podrá crear un nuevo usuario (docente) y asignar un rol ya sea tutor o directivo.	
Observaciones: El administrador master puede obtener información mediante los servicios web buscando al usuario enviando como parámetro el número de cédula en la base de datos SICOA y podrá insertar, actualizar o eliminar información en la base de datos SIGET.	

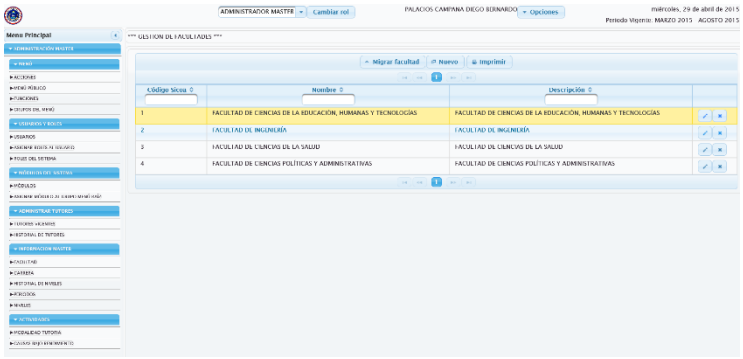


Gestión de la Seguridad

Fuente: Sandra Contento/Janneth Guamán

Tabla 56: Iteración 1 Historia 3

Historia de Usuario	
Numero:3	**Usuario:** Administrador Master
Nombre historia: Gestión de Facultades	
Prioridad en negocio: Medio	**Riesgo en desarrollo:** Medio
Esfuerzo: Bajo	Iteración asignada 1
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El Administrador Master una vez que ha ingresado al sistema, podrá escoger la opción Migrar Facultad para añadir nuevas Facultades desde la Base de Datos SICOA a la base de datos SIGET.	
Observaciones: El administrador master podrá insertar o actualizar una facultad de la base de datos SIGET.	

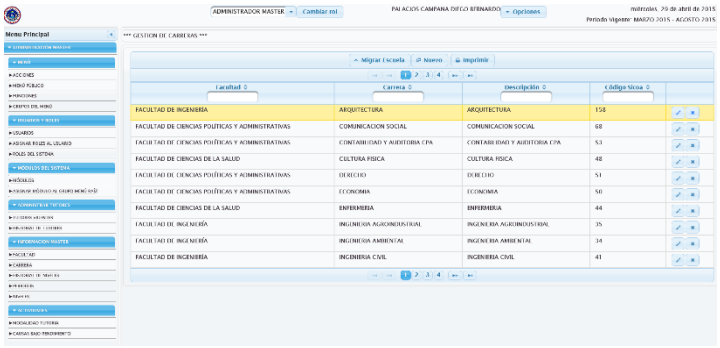


Gestión de Facultades

Fuente: Sandra Contento/Janneth Guamán

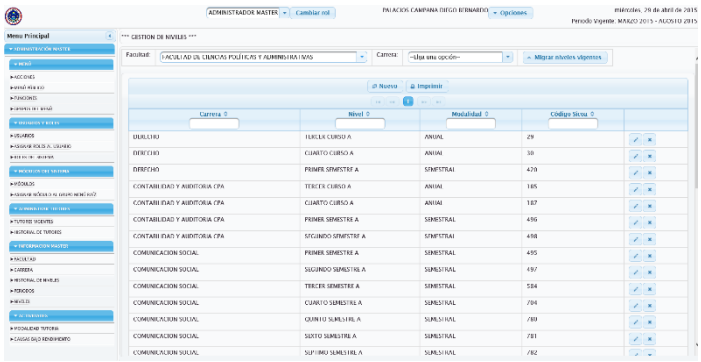
Tabla 57 Iteración 1 Historia 4

Historia de Usuario	
Numero:4	**Usuario:** Administrador Master
Nombre historia: Gestión de Carreras	
Prioridad en negocio: Medio	**Riesgo en desarrollo:** Medio
Esfuerzo: Bajo	Iteración asignada 1
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El Administrador Master una vez que ha ingresado al sistema, podrá escoger la opción Migrar Carrera para añadir nuevas Carreras desde la Base de Datos SICOA a la base de datos SIGET	

Observaciones: El administrador master podrá insertar o actualizar una carrera de la base de datos SIGET.
 **Gestión de Carreras**

Fuente: Sandra Contento/Janneth Guamán

Tabla 58 Iteración 1 Historia 5

Historia de Usuario	
Numero:5	**Usuario:** Administrador Master
Nombre historia: Gestión de Niveles	
Prioridad en negocio: Medio	**Riesgo en desarrollo:** Medio
Esfuerzo: Bajo	Iteración asignada 1
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El Administrador Master una vez que ha ingresado al sistema, podrá escoger la opción Migrar Niveles para añadir nuevos Niveles desde la Base de Datos SICOA a la base de datos SIGET.	
Observaciones: El administrador master podrá insertar o actualizar un nivel de la base de datos SIGET.	
 **Gestión de Niveles**	

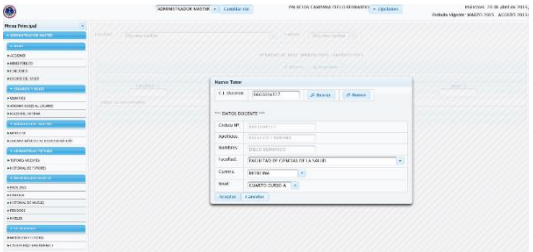
Fuente: Sandra Contento/Janneth Guamán

Tabla 59: Iteración 1 Historia 6

Historia de Usuario	
Numero:6	**Usuario:** Administrador Master
Nombre historia: Gestión de Periodos.	
Prioridad en negocio: Medio	**Riesgo en desarrollo:** Medio
Esfuerzo: Bajo	Iteración asignada 1
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El Administrador Master una vez que ha ingresado al sistema, podrá escoger la opción Migrar Periodos para añadir periodos académicos desde la Base de Datos SICOA a la base de datos SIGET	
Observaciones: El administrador master podrá insertar o actualizar una periodo académico de la base de datos SIGET.	

Fuente: Sandra Contento/Janneth Guamán

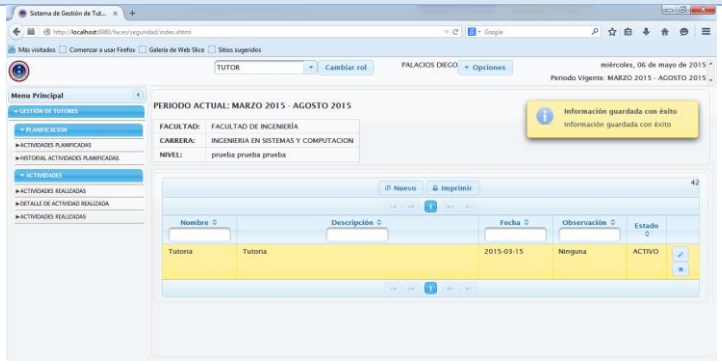
Tabla 60: Iteración 1 Historia 7

Historia de Usuario	
Numero:7	**Usuario:** Administrador Master, Directivo
Nombre historia: Gestión de Tutores.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Esfuerzo: Alto	Iteración asignada 2
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El usuario Administrador Master o Directivo una vez que ha ingresado al sistema, podrá asignar un tutor a un determinado nivel.	
Observaciones: Se puede buscar los datos del docente mediante el número de cédula, en la base de datos SIGET, si no existen datos se procede a buscar en la base de Datos SICOA para añadir al usuario y asignar el rol tutor.	
 **Gestión de Tutores**	

Fuente: Sandra Contento/Janneth Guamán

Iteración 2

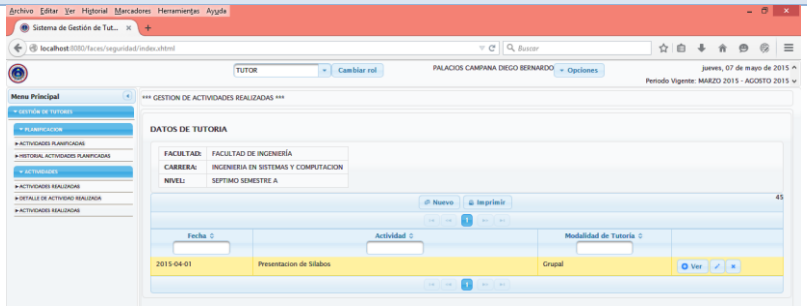
Tabla 61: Iteración 2 Historia 8

Historia de Usuario	
Numero:8	**Usuario:** Tutor
Nombre historia: Gestión de Actividades Planificadas.	
Prioridad en negocio: Medio	**Riesgo en desarrollo:** Medio
Esfuerzo: Bajo	Iteración asignada 2
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El Usuario Tutor debe autenticarse en el sistema y escoger la opción Actividades Planificadas, donde podrá insertar, actualizar o eliminar estas actividades.	
Observaciones: El usuario Tuto podrá generar un reporte de las actividades planificadas durante el periodo vigente.	
	

Fuente: Sandra Contento/Janneth Guamán

Tabla 62: Iteración 2 Historia 9

Historia de Usuario	
Numero:9	**Usuario:** Tutor
Nombre historia: Gestión de Actividades Realizadas.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Esfuerzo: Alto	Iteración asignada 2
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El usuario Tutor debe autenticarse en el sistema y escoger la opción Actividades Realizadas donde podrá insertar la actividad y el detalle de los estudiantes.	

Observaciones: el usuario Tutor podrá actualizar la información de la actividad asi como el detalle de los estudiantes.


Fuente: Sandra Contento/Janneth Guamán

Iteración 3

Tabla 63: Iteración 3 Historia 10

Historia de Usuario	
Numero:10	**Usuario:** Administrador Master, Tutor, Directivo.
Nombre historia: Emisión de Reportes.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Esfuerzo: Alto	Iteración asignada 3
Programador responsable: Sandra Contento / Janneth Guamán	
Descripción: El Administrador Master, Tutor y Directivo podrá imprimir un reporte y su historial de acuerdo a la actividad que seleccione.	
Observaciones:	

Fuente: Sandra Contento/Janneth Guamán

4.1.3.4. Código fuente

El código fuente se lo ha tomado de las actividades realizadas, puesto que todas y cada una de las clases, funciones, controladores y paginas xhtml son similares en estructura. **(Ver anexo 1, 2, 3, 4)**

4.1.4. Pruebas

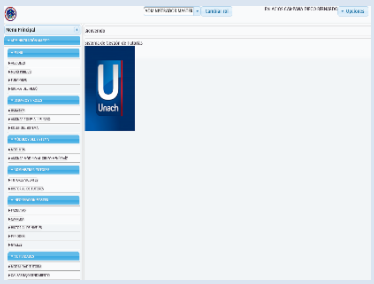
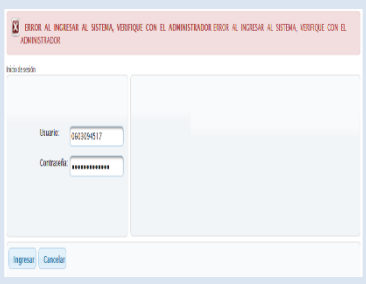
Las pruebas son muy importantes y son creadas a partir de las historias de los usuarios, el usuario debe especificar los aspectos que se van a probar cuando una historia de usuario ha sido correctamente realizada, esta prueba de usuario puede tener una o más pruebas de aceptación, las que sean necesarias para garantizar el correcto funcionamiento.

Cada prueba que se realice representa una salida esperada del sistema. Una historia de usuario no se considera completa hasta que se realicen todas las pruebas de aceptación necesarias.

A continuación se muestra las pruebas realizadas a cada una de las historias de los usuarios del sistema con su respectiva tabla de pruebas.

Historia 1

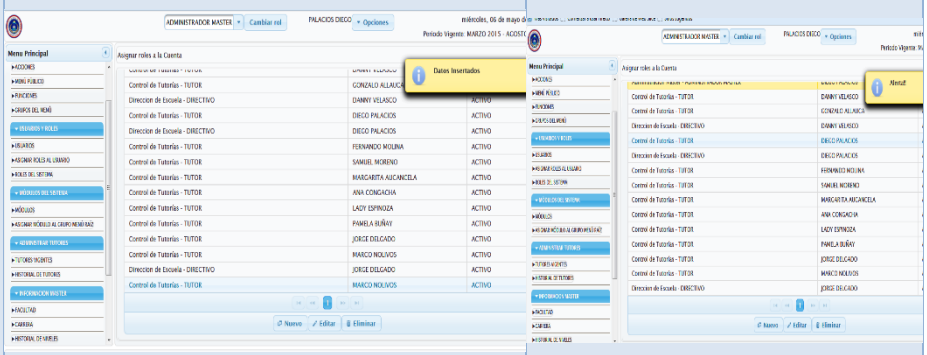
Tabla 64: Pruebas Historia 1

echa	Descripción	Autor
05/01/2015	Pruebas	Sandra Contento
Gestión de Acceso a Usuarios		
Descripción	Hay tres tipos de usuarios: Administrador Master, Tutor, Directivo.	
Condiciones de Ejecución.	Cada uno de los usuarios mencionados debe constar en la base de datos y tener asignado un rol y permisos de acceso a los módulos y menús dependiendo de las funciones que le corresponden.	
Entrada	• El usuario ingresa su cuenta y contraseña • El proceso de control de acceso a Usuarios finaliza.	
Resultado Esperado	Después de ingresar su cuenta y su contraseña, debe mostrarse automáticamente los menús asignados para cada tipo de usuario.	
Evaluación de la Prueba	**Exitosa**	**Fallida**
		

Fuente: Sandra Contento/Janneth Guamán

Historia 2

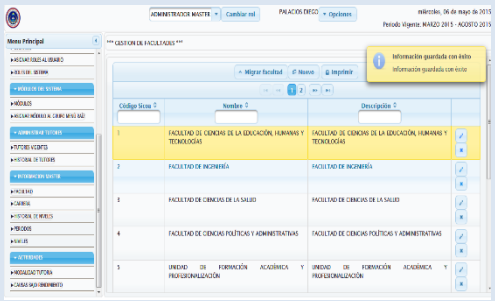
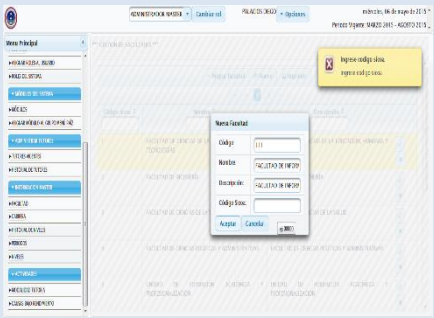
Tabla 65: Pruebas Historia 2

Fecha	Descripción	Autor
12/01/2015	Pruebas	Janneth Guamán
Gestión de la Seguridad		
Descripción	El Administrador Master una vez que ingrese al sistema tiene el control sobre el mismo, en cuanto tiene que ver a la seguridad podrá realizar diferentes actividades dependiendo de la necesidad que se presente.	
Condiciones de Ejecución.	El Administrador Master debe constar en la base de datos del sistema para poder realizar las actividades.	
Entrada	• El Administrador Master una vez que ingresa al sistema puede seleccionar la opción Usuarios o Asignar Roles al Usuario. • Si selecciona Usuarios puede insertar, actualizar o eliminar un usuario de la base de datos SIGET. • Si selecciona Asignar Roles al usuario, buscara al docente para asignarle un rol.	
Resultado Esperado	Tras ingresar el administrador los datos de usuario o de la asignación de roles se insertaron los datos exitosamente y dependiendo del tipo de rol podrá realizar las diferentes actividades.	
Evaluación de la Prueba	**Exitosa**	**Fallida**
		

Fuente: Sandra Contento/Janneth Guamán

Historia 3

Tabla 66: Pruebas Historia 3

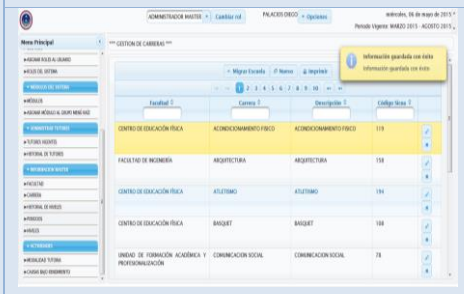
Fecha	Descripción	Autor
19/01/2015	Pruebas	Sandra Contento
Gestión de la Facultad		
Descripción	El Administrador Master podrá insertar o actualizar una facultad.	
Condiciones de Ejecución.	El Administrador Master debe constar en la base de datos del sistema para poder realizar las actividades.	
Entrada	• El Administrador Master una vez que ingresa al sistema escoge el submenú FACULTAD. • Selecciona nueva facultad o actualizar facultad	
Resultado Esperado	Tras ingresar o actualizar el administrador los datos de la facultad se insertan o actualizaran exitosamente.	
Evaluación de la Prueba	**Exitosa**	**Fallida**
		

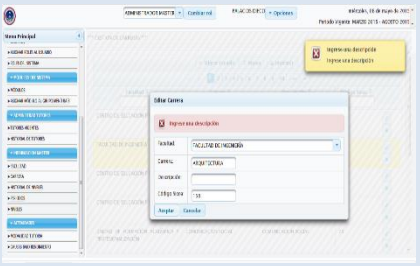
Fuente: Sandra Contento/Janneth Guamán

Historia 4

Tabla 67: Pruebas Historia 4

Fecha	Descripción	Autor
26/01/2015	Pruebas	Janneth Guamán
Gestión de la Carrera		
Descripción	El Administrador Master podrá insertar o actualizar una carrera.	
Condiciones de Ejecución.	El Administrador Master debe constar en la base de datos del sistema para poder realizar las actividades.	
Entrada	• El Administrador Master una vez que ingresa al sistema escoge el submenú FACULTAD. • Selecciona nueva facultad o actualizar facultad	
Resultado Esperado	Tras ingresar o actualizar el administrador los datos de la facultad se insertan o actualizan exitosamente.	
Evaluación de la Prueba	**Exitosa**	**Fallida**



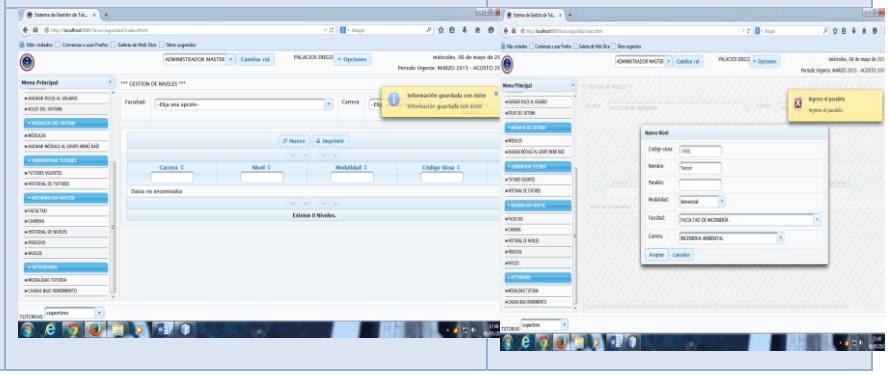


Fuente: Sandra Contento/Janneth Guamán

Historia 5

Tabla 68: Pruebas Historia 5

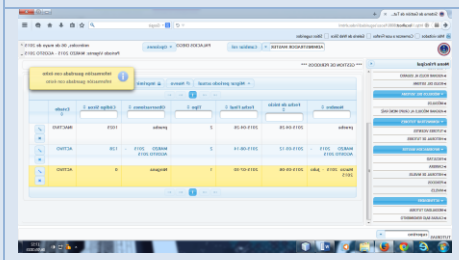
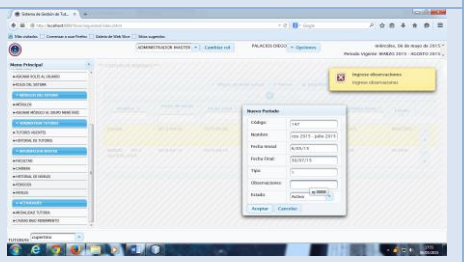
Fecha	Descripción	Autor
2/02/2015	Pruebas	Sandra Contento
Gestión de Niveles		
Descripción	El Administrador Master podrá insertar o actualizar una facultad.	
Condiciones de Ejecución.	El Administrador Master debe constar en la base de datos del sistema para poder realizar las actividades.	
Entrada	• El Administrador Master una vez que ingresa al sistema escoge el submenú FACULTAD. • Selecciona nueva facultad o actualizar facultad	
Resultado Esperado	Tras ingresar o actualizar el administrador los datos de la facultad se insertan o actualizan exitosamente.	
Evaluación de la Prueba	**Exitosa**	**Fallida**



Fuente: Sandra Contento/Janneth Guamán

Historia 6

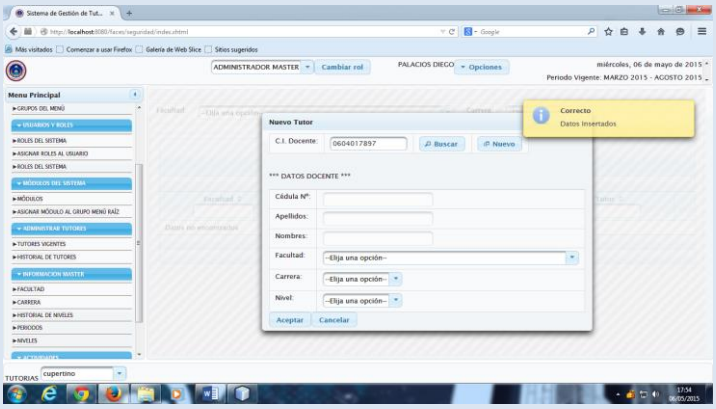
Tabla 69: Pruebas Historia 6

Fecha	Descripción	Autor
9/02/2015	Pruebas	Janneth Guamán
Gestión de Periodos		
Descripción	El Administrador Master podrá insertar o actualizar un periodo.	
Condiciones de Ejecución.	El Administrador Master debe constar en la base de datos del sistema para poder realizar las actividades.	
Entrada	• El Administrador Master una vez que ingresa al sistema escoge el submenú PERIODOS. • Selecciona nuevo periodo o actualizar periodo.	
Resultado Esperado	Tras ingresar o actualizar el administrador los datos de un periodo se insertan o actualizan exitosamente.	
Evaluación de la Prueba	**Exitosa**	**Fallida**
		

Fuente: Sandra Contento/Janneth Guamán

Historia 7

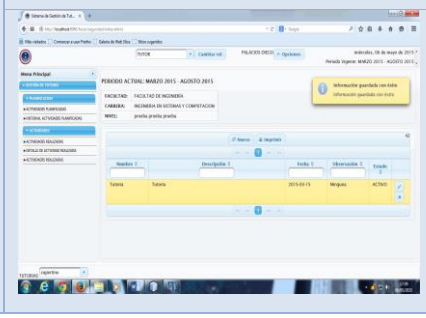
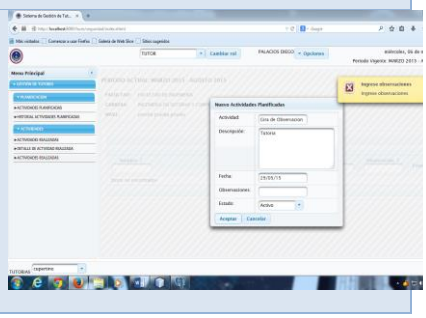
Tabla 70: Pruebas Historia 7

Fecha	Descripción	Autor
16/02/2015	Pruebas	Sandra Contento
Gestión de Tutores		
Descripción	El Administrador Master podrá insertar, actualizar o eliminar un tutor.	
Condiciones de Ejecución.	El Administrador Master debe constar en la base de datos del sistema para poder realizar las actividades.	
Entrada	• El Administrador Master una vez que ingresa al sistema escoge el submenú Tutor. • Selecciona nuevo, actualizar o eliminar un Tutor	
Resultado Esperado	Tras ingresar, actualizar o eliminar el administrador los datos del tutor se insertan, actualizan o eliminan exitosamente.	
Evaluación de la Prueba	**Exitosa** 	

Fuente: Sandra Contento/Janneth Guamán

Historia 8

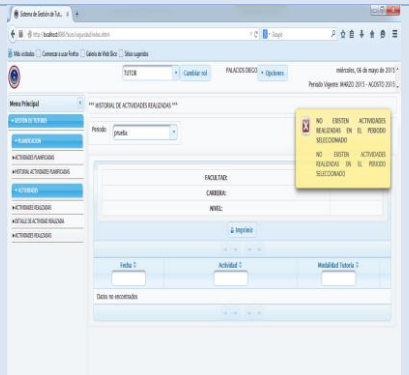
Tabla 71: Pruebas Historia 8

Fecha	Descripción	Autor
23/02/2015	Pruebas	Janneth Guamán
Gestión de Actividades Planificadas		
Descripción	El Tutor podrá insertar, actualizar o eliminar una actividad planificada.	
Condiciones de Ejecución.	El Tutor debe constar en la base de datos del sistema para poder realizar las actividades.	
Entrada	• El Tutor una vez que ingresa al sistema escoge el submenú Actividades Planificadas. • Selecciona nuevo, actualizar o eliminar una actividad planificada.	
Resultado Esperado	Tras ingresar, actualizar o eliminar el Tutor los datos de la actividad planificada se insertan, actualizan o eliminan exitosamente.	
Evaluación de la Prueba	**Exitosa**	**Fallida**
		

Fuente: Sandra Contento/Janneth Guamán

Historia 9

Tabla 72: Pruebas Historia 9

Fecha	Descripción	Autor
2/03/2015	Pruebas	Sandra Contento
Gestión de Actividades Realizadas		
Descripción	El Tutor podrá insertar, actualizar o eliminar una actividad realizada.	
Condiciones de Ejecución.	El Tutor debe constar en la base de datos del sistema para poder realizar las actividades.	
Entrada	• El Tutor una vez que ingresa al sistema escoge el submenú Actividades Realizadas. • Selecciona nuevo, actualizar o eliminar una activdad realizada.	
Resultado Esperado	Tras ingresar, actualizar o eliminar el Tutor los datos de la actividad realizada se insertan, actualizan o eliminan exitosamente.	
Evaluación de la Prueba	**Exitosa**	**Fallida**
		

Fuente: Sandra Contento/Janneth Guamán

Historia 10

Tabla 73: Pruebas Historia 10

Fecha	Descripción	Autor
9/03/2015	Pruebas	Janneth Guamán
Gestión de Reportes		
Descripción	El Administrador Master, el Tutor o el Directivo dependiendo de la actividad que requiera podrán imprimir un reporte.	
Condiciones de Ejecución.	El Administrador Master, el Tutor y Directivo debe constar en la base de datos del sistema para poder generar un reporte	
Entrada	• El Administrador Master, el Tutor o el Directivo una vez que ingresa al sistema escoge la opción imprimir.	
Resultado Esperado	Tras seleccionar la acción que requieren se imprime, exitosamente el reporte.	
Evaluación de la Prueba	**Exitosa**	**Fallida**

Fuente: Sandra Contento/Janneth Guamán

CONCLUSIONES

- Al determinar la librería de componentes adecuada para el desarrollo del sistema de Gestión de Tutorías se tomó como referencia los siguientes indicadores: documentación, diversidad de componentes, compatibilidad con navegadores, facilidad para iniciar, facilidad de uso y rendimiento.
- La librería de componentes mejor valorada para el desarrollo de aplicaciones web es Primefaces porque cuenta con 117 componentes, no requiere dependencias para utilizar la librería, es compatible con otras librerías de componentes, posee soporte Ajax, su documentación es actualizada, además los componentes son amigables, atractivos e innovadores tanto para el desarrollador como para el usuario.
- Con el desarrollo de la aplicación SIGET se ha logrado automatizar el proceso de gestión de tutorías, además permite al docente una adecuada y transparente organización de las actividades realizadas durante el periodo académico. Además el desarrollo de esta aplicación es un indicador fundamental en el proceso de evaluación de desempeño de carreras según el Consejo de Evaluación, Acreditación y Aseguramiento de la Calidad de la Educación Superior CEAACES
- Mediante el consumo de servicios web se logró la integración del sistema SIGET con el sistema de Control Académico SICOA de la UNACH, permitiendo así reducir la duplicidad de información.
- La herramienta Neoload permite medir el rendimiento de aplicaciones web en tiempo real, ya que posee soporte Push, WebSocket y HTTP streaming que simula la carga que el servidor tendrá que soportar.

RECOMENDACIONES

- Para el desarrollo de una aplicación web enriquecida se recomienda utilizar la librería de componentes Primefaces.
- Se sugiere utilizar el patrón Modelo Vista Controlador en el desarrollo de aplicaciones web porque es necesario separar el diseño, la lógica de negocios y la interfaz de usuario permitiendo realizar pruebas unitarias de los componentes y dar un mantenimiento más sencillo de las aplicaciones.
- Se sugiere un análisis exhaustivo de las diversas versiones de Primefaces con respecto a la compatibilidad de las mismas ya que algunos componentes no funciones correctamente en versiones superiores o viceversa.
- En las futuras investigaciones relacionadas al desarrollo de software en beneficio institucional, las aplicaciones se deben integrar a través de servicios web con el Sistema de Control Académico SICOA para que de esta manera no exista duplicidad de información.
- Para medir el rendimiento y la carga de una aplicación web se recomienda utilizar la herramienta Neoload por su versatilidad y fiabilidad en los resultados.

BIBLIOGRAFÍA

Bautista, J. (s.f.). Universidad Union Bolivariana. Obtenido de Ingeniería de software: http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html

Cárdenas, J. (Enero de 2013). Diseño e Implementación de un sistema web para la getión y administración empresarial basadoen tecnología JEE y Primefaces. Obtenido de http://repositorio.espe.edu.ec/bitstream/21000/6457/1/T-ESPE-038234.pdf

Clavijo, P. (Marzo de 2010). Introducción a JSF. Recuperado el 06 de Enero de 2015, de JavaServer Faces: http://www.lintips.com/files/Taller_JSF_1aSesion_Paulo_Clavijo-2010.pdf

Fowler, M. (18 de Julio de 2009). Martin Fowler. Obtenido de http://martinfowler.com/eaaDev/uiArchs.html

Hookom, J. (17 de Agosto de 2005). Inside Facelets . Obtenido de Part 1: An Introduction: http://web.archive.org/web/20130113100928/http://www.jsfcentral.com/articles/facelets_1.html

JuntadeAndalucia. (01 de Marzo de 2013). Capa de Presentacion-JavaServer Faces. Recuperado el 06 de Enero de 2015, de Marco de Desarrollo de la Junta de Andalucía: http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/101

King, G. (23 de Marzo de 2009). SeamFramework.org. Obtenido de Community Documentation: http://docs.jboss.org/webbeans/reference/current/es-ES/html/index.html

Loor, J. (2014). JSF - JavaServer Faces. Recuperado el 06 de Enero de 2015, de NDeveloper-Lider en Desarrollo de Aplicaciones Java: http://www.ndeveloper.com/ndeveloperDocuments/documents/nDeveloper_JavaServerFaces.pdf

Lou, R. (s.f.). Programación en Castellano. Recuperado el 25 de Enero de 2015, de Introducción a la Tecnología JavaServer Faces: http://programacion.net/articulo/introduccion_a_la_tecnologia_javaserver_faces_233/6

Oscar Belmonte, Carlos Granell,María Erdozan. (Octubre de 2010). Desarrollo de Proyectos Informáticos con Tecnología Java. Obtenido de Universidad Jaumet: http://www3.uji.es/~belfern/pdf/libroJavaConTapa.pdf

PrimeFaces. (2009-2014). Obtenido de Ultimate JSF Framework: http://www.primefaces.org/index

RichFaces. (2010). Obtenido de The next-generation JSF component framwork by JBoss: http://richfaces.jboss.org/

Secretaría Nacional de la Administración Pública. (10 de Junio de 2008). Obtenido de Importancia del Software Libre para un país : http://www.administracionpublica.gob.ec/software-libre/

SicUma. (2013). Tutorial de javaServer Faces. Recuperado el 06 de Enero de 2015, de Grupo de Investigación SICUMA: http://www.sicuma.uma.es/export/sites/sicuma/es/formacion/descargas/JSF.pdf

Universidad de Alicante. (26 de Junio de 2014). Desarrollo de Aplicaciones y Servicios con JAva EE. Obtenido de Introduccion a JavaServer FACes: http://expertojava.ua.es/j2ee/publico/jsf-2012-13/sesion01-apuntes.html#Librer%C3%ADas+de+implementaci%C3%B3n+de+JSF

Universidad Nacional de Chimborazo. (24 de Febrero de 2015). Reglamentos UNACH 2015. Obtenido de Reglamento de Tutorías Académicas: http://www.unach.edu.ec/reglamentos/images/pdf/reglamentodetutoriasacademicas.pdf

University of Vigo. (Noviembre de 2008). Área de Ciencias de la Computación e Inteligencia Artificiañ. Obtenido de Java Server Faces: http://ccia.ei.uvigo.es/docencia/SCS/1011/transparencias/Tema5-3.JSF.pdf

YiiFramework. (10 de Octubre de 2011). Best MVC Practices. Obtenido de YiiFramework: http://www.yiiframework.com/doc/guide/1.1/en/basics.best-practices

ANEXOS

ANEXO 1: CLASE

```
package master.logica.clases;

public class Facultad {
  private int codigo;
  private String nombre;
  private String descripcion;
  private int codigo_sicoa;

    public Facultad(int codigo, String nombre, String descripcion, int codigo_sicoa) {
        this.codigo = codigo;
        this.nombre = nombre;
        this.descripcion = descripcion;
        this.codigo_sicoa = codigo_sicoa;
    }

    public Facultad() {
    }

    public int getCodigo() {
        return codigo;
    }

    public void setCodigo(int codigo) {
        this.codigo = codigo;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getDescripcion() {
        return descripcion;
    }

    public void setDescripcion(String descripcion) {
        this.descripcion = descripcion;
    }

    public int getCodigo_sicoa() {
        return codigo_sicoa;
    }

    public void setCodigo_sicoa(int codigo_sicoa) {
        this.codigo_sicoa = codigo_sicoa;
    }
}
```

ANEXO 2: FUNCIONES

```
package master.logica.funciones;
import accesodatos.AccesoDatos;
import accesodatos.ConjuntoResultado;
import accesodatos.Parametro;
import java.sql.SQLException;
import java.util.ArrayList;
import master.logica.clases.Tutor;
import master.logica.clases.Facultad;
public class FEscuela {
   public static boolean Insertar(Escuela escuela) throws Exception {
      boolean eje = false;
      try {
         ArrayList<Parametro> lstP = new ArrayList<Parametro>();
         String sql = "select * from master.f_insert_escuela(?,?,?,?,?)";
         lstP.add(new Parametro(1, escuela.getCodigo()));
         lstP.add(new Parametro(2, escuela.getCodigo_facultad().getCodigo()));
         lstP.add(new Parametro(3, escuela.getNombre()));
         lstP.add(new Parametro(4, escuela.getDescripcion()));
         lstP.add(new Parametro(5, escuela.getCodigo_sicoa()));
         ConjuntoResultado rs = AccesoDatos.ejecutaQuery(sql, lstP);
         while (rs.next()) {
            if (rs.getString(0).equals("true"));
            eje = true;
          }
      } catch (SQLException exConec) {
         throw new Exception(exConec.getMessage());
       }
      return eje;
   }
   public static ArrayList<Escuela> llenarEscuela(ConjuntoResultado rs) throws
Exception {
      ArrayList<Escuela> lst = new ArrayList<Escuela>();
      Escuela escuela = null;
      try {
         while (rs.next()) {
            escuela = new Escuela(rs.getInt("pcodigo"),
FFacultad.ObtenerFacultadDadoCodigo(rs.getInt("pcodigo_facultad")),
                  rs.getString("pnombre"), rs.getString("pdescripcion"),
rs.getInt("pcodigo_sicoa"));
            lst.add(escuela);
         }
      } catch (Exception e) {
         lst.clear();
         throw e;
      }
```

```
        return lst;
    }
public static boolean actualizar(Escuela escuela) throws Exception {
        boolean eje = false;
        try {
            ArrayList<Parametro> lstP = new ArrayList<Parametro>();
            String sql = "select * from master.f_update_escuela(?,?,?,?,?)";
            lstP.add(new Parametro(1, escuela.getCodigo_facultad().getCodigo()));
            lstP.add(new Parametro(2, escuela.getNombre()));
            lstP.add(new Parametro(3, escuela.getDescripcion()));
            lstP.add(new Parametro(4, escuela.getCodigo_sicoa()));
            lstP.add(new Parametro(5, escuela.getCodigo()));
            ConjuntoResultado rs = AccesoDatos.ejecutaQuery(sql, lstP);
            while (rs.next()) {
                if (rs.getString(0).equals("true"));
                eje = true;
              }
        } catch (SQLException exConec) {
            throw new Exception(exConec.getMessage());
        }
        return eje;
    public static boolean eliminar(int codigo) throws Exception {
        boolean eje = false;
        try {
            ArrayList<Parametro> lstP = new ArrayList<Parametro>();
            String sql = "select * from master.f_delete_escuela(?)";
            lstP.add(new Parametro(1, codigo));
            ConjuntoResultado rs = AccesoDatos.ejecutaQuery(sql, lstP);
            while(rs.next() )
                if (rs.getString(0).equals("true"));
                eje = true;
            }          }
        } catch (SQLException exConec) {
            throw new Exception(exConec.getMessage());
        }
        return eje;
    }
}
```

ANEXO 3: CONTROLADOR

```
package master.presentacion.beans;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ManagedProperty;
import javax.faces.bean.ViewScoped;
import javax.xml.ws.WebServiceRef;
import master.logica.clases.Facultad;
import master.logica.funciones.FFacultad;
import master.logica.funciones.FParametros;
import org.primefaces.context.DefaultRequestContext;
import org.tempuri.Wstutorias;
import recursos.Tools;
import recursos.Util;
import reportes.generador.Report_;
import reportes.templates.FacultadR;
public class FacultadControlador {
   @WebServiceRef(wsdlLocation = "WEB-
INF/wsdl/sicoaweb.unach.edu.ec/wstutorias/infoAcademico.asmx.wsdl")
   private Wstutorias service;
   private ArrayList<Facultad> lstVirtual;
   private Facultad objFacultad;
   private Facultad facultadSel;
   private ArrayList<Facultad> lstFacultad;
    @ManagedProperty(value = "#{sesionUsuarioDataManager}")
   private SesionUsuarioDataManager dm;

   public Wstutorias getService() {
      return service;
   }
   public void setService(Wstutorias service) {
      this.service = service;
   }
   public ArrayList<Facultad> getLstVirtual() {
      return lstVirtual;
   }
   public void setLstVirtual(ArrayList<Facultad> lstVirtual) {
      this.lstVirtual = lstVirtual;
   }
   public SesionUsuarioDataManager getDm() {
      return dm;
   }
   public void setDm(SesionUsuarioDataManager dm) {
      this.dm = dm;
   }
   public Facultad getObjFacultad() {
```

```
        return objFacultad;
    }
    public void setObjFacultad(Facultad objFacultad) {
        this.objFacultad = objFacultad;
    }
    public ArrayList<Facultad> getLstFacultad() {
        return lstFacultad;
    }
    public void setLstFacultad(ArrayList<Facultad> lstFacultad) {
        this.lstFacultad = lstFacultad;
    }
    public Facultad getFacultadSel() {
        return facultadSel;
    }
    public void setFacultadSel(Facultad facultadSel) {
        this.facultadSel = facultadSel;
    }
    public FacultadControlador() {
        this.reinit();
    }
    private void reinit(){
        this.objFacultad = new Facultad();
        this.facultadSel = new Facultad();
        this.cargarFacultad();
        this.facultadSel = this.lstFacultad.get(0);
    }
    public void cargarFacultad() {
        try {
            this.lstFacultad = FFacultad.ObtenerFacultades();
        } catch (Exception e) { }
     }
        public void insertarFacultad() {
        try {
            if (FFacultad.Insertar(objFacultad)) {
                this.reinit();

DefaultRequestContext.getCurrentInstance().execute("wdlgNuevoFacultad.hide()
");
                Util.addSuccessMessage("Información guardada con éxito");
                //System.out.println("public void insertarFacultad dice: Error al guardar
la información");
            } else {
                Util.addSuccessMessage("Error al guardar la información");
                //System.out.println("public void insertarFacultad dice: Error al guardar
la información");
            }
        } catch (Exception e) {
```

```
            Util.addErrorMessage("VERIFIQUE LA INFORMACION Y VUELVA A
INTENTAR");
            //System.out.println("ERROR AL GUARDAR LA INFORMACION");
                }
        }
    public void eliminarFacultad() {
        try {
            if (FFacultad.eliminar(facultadSel)) {
                this.reinit();
DefaultRequestContext.getCurrentInstance().execute("wdlgEliminarFacultad.hide
()");
                Util.addSuccessMessage("Información eliminada.");
                //System.out.println("public void eliminarFacultad dice: Información
eliminada.");
            } else {
                Util.addErrorMessage("Error al eliminar la información.");
            }
        } catch (Exception e) {
            Util.addErrorMessage("VUELVA A INTENTAR" + e.getMessage());
          }
    }
  public String generaReporte() {
        String redireccion = "";
        try {
            cargarFacultad();
            reportes.generador.Report_ facultad= new Report_();
            FacultadR datasource = new FacultadR();
            for (Facultad detalle : lstFacultad ) {
                datasource.addDetalle(detalle);
            }
            HashMap encabezado = new HashMap();
            encabezado.put("usuario", this.dm.getSesionUsuario().getNombres() + " "
+this.dm.getSesionUsuario().getApellidos());
             redireccion = facultad.generaReporte("facultades", datasource,
encabezado);
            } catch (Exception e) {
            Util.addErrorMessage("VUELVA A INTENTAR" + e.getMessage());
           }
        return redireccion;
    }

    public void migrarFacultad(){
        try {

            String encriptado =
Tools.sha256(FParametros.ObtenerParametrosDadoCodigo(3).getValor_texto());
            ArrayOfFacultad lista = getFacultades(encriptado);
            this.cargarFacultad();
```

```
            if (lstFacultad.size() > 0 )
            {
            int tam = lstFacultad.size();
            for (org.tempuri.Facultad facultad : lista.getFacultad()) {
               int i = 0;
               for (master.logica.clases.Facultad facultad1 : lstFacultad) {
                  if (facultad.getCodigo() ==  facultad1.getCodigo()) {
                     break;
                  } else {
                     i = i + 1;
                  }
               if (i == tam)
                  {
                     objFacultad.setCodigo(facultad.getCodigo());
                     objFacultad.setNombre(facultad.getNombre());
                     objFacultad.setDescripcion(facultad.getNombre());
                     objFacultad.setCodigo_sicoa(facultad.getCodigo());
                     FFacultad.Insertar(objFacultad);
                      break;
                  }
               }
         }
            } else{
               for (org.tempuri.Facultad facultad : lista.getFacultad()){
                     objFacultad.setCodigo(facultad.getCodigo());
                     objFacultad.setNombre(facultad.getNombre());
                     objFacultad.setDescripcion(facultad.getNombre());
                     objFacultad.setCodigo_sicoa(facultad.getCodigo());
                     FFacultad.Insertar(objFacultad);
               }
            }
         this.cargarFacultad();
         } catch (Exception e) {
         }
   }
   private ArrayOfFacultad getFacultades(java.lang.String passwordws) {
      org.tempuri.WstutoriasSoap port = service.getWstutoriasSoap();
      return port.getFacultades(passwordws);
   }
}
```

ANEXO 4: VISTA .xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:p="http://primefaces.org/ui">
    <h:head>
        <title>
            <ui:define name="Gestión de Facultades">GESTION DE
FACULTADES</ui:define>
        </title>
    </h:head>
    <h:body>
        <ui:composition template="../../templates/plantillaInternaCenter.xhtml">
            <ui:define name="top" >*** GESTION DE FACULTADES ***</ui:define>
            <ui:define name="contenido">
                <p:growl id="mensajes" autoUpdate="true" showDetail="true"/>
                <h:form id="frmfacultad">
                    <p:ajaxStatus onstart="dlgStatus.show();" oncomplete="dlgStatus.hide();"/>
                    <p:panel id="pnlFacultad">
                        <p:dataTable id="tblFacultad"
value="#{facultadControlador.lstFacultad}"
                                     var="facultad"
                                     paginator="true" rows="10" rowKey="#{facultad.codigo}"
selectionMode="single"
                                     selection="#{facultadControlador.facultadSel}">
                            <f:facet name="header">
                                <p:commandButton id="btnMigrarDatos" value="Migrar facultad"
action="#{facultadControlador.migrarFacultad()}" icon="ui-icon-update"
                                                 title="Migrar datos" style="right: initial"
                                                  update=":frmfacultad:pnlFacultad"/>
                                <p:commandButton id="btnNuevoFacultad1" value="Nuevo"
icon="ui-icon-newwin"
                                                 oncomplete="wdlgNuevoFacultad.show()" title="Nueva
Facultad" style="right: initial"/>
                                <h:commandLink action="#{facultadControlador.generaReporte()}">
                                    <p:commandButton value="Imprimir" icon="ui-icon-print"
                                                     title="Imprimir Listado">
                                    </p:commandButton>
                                </h:commandLink>
                                <br></br>
                            </f:facet>


                            <p:column filterBy="#{facultad.codigo_sicoa}"
filterMatchMode="startsWith"
                                      sortBy="#{facultad.codigo_sicoa}" headerText="Código Sicoa">
                                <h:outputText value="#{facultad.codigo_sicoa}"/>
                            </p:column>
```

```
                    <p:column filterBy="#{facultad.nombre}"
filterMatchMode="startsWith"
                              sortBy="#{facultad.nombre}" headerText="Nombre">
                        <h:outputText value="#{facultad.nombre}"/>
                    </p:column>



                    <p:column filterBy="#{facultad.descripcion}"
filterMatchMode="startsWith"
                              sortBy="#{facultad.descripcion}" headerText="Descripción">
                        <h:outputText value="#{facultad.descripcion}"/>
                    </p:column>

                    <p:column exportable="false">
                        <p:commandButton id="btnEditarFacultad" icon="ui-icon-pencil"
title="Editar Facultad"
                                         oncomplete="wdlgEditarFacultad.show()"
                                         process="@this" update=":dlgEditarFacultad, @form">
                            <f:setPropertyActionListener value="#{facultad}"
                                                         target="#{facultadControlador.facultadSel}"/>
                        </p:commandButton>


                        <p:commandButton id="btnEliminarFacultad" icon="ui-icon-close"
title="Eliminar Facultad"
                                         oncomplete="wdlgEliminarFacultad.show()"
                                         update=":dlgEliminarFacultad">
                            <f:setPropertyActionListener value="#{facultad}"
                                                         target="#{facultadControlador.facultadSel}"/>
                        </p:commandButton>

                    </p:column>

                </p:dataTable>
            </p:panel>

        </h:form>
    </ui:define>
    <ui:define name="dialogos">
        <p:dialog id="dlgNuevoFacultad" header="Nueva Facultad"
widgetVar="wdlgNuevoFacultad"
                  modal="true" closable="false" resizable="false"
                  showEffect="clip" hideEffect="fold">



            <h:form id="frmNuevoFacultad">
                <p:ajaxStatus onstart="dlgStatus.show();"
oncomplete="dlgStatus.hide();"/>
                <p:panelGrid id="pngNuevoFacultad" columns="2">
                    <h:outputText value="Código: "/>
```

```
                    <p:inputText value="#{facultadControlador.objFacultad.codigo}"
title="ej: Ingenieria"
                                 required="true" requiredMessage="Ingrese un código"
converterMessage="VERIFIQUE CODIGO"/>

                    <h:outputText value="Nombre:"/>
                    <p:inputText value="#{facultadControlador.objFacultad.nombre}"
title="ej: Ingenieria"
                                 required="true" requiredMessage="Ingrese el nombre"
converterMessage="VERIFIQUE NOMBRE (200 CARACTERES)"/>

                    <h:outputText value="Descripción:"/>
                    <p:inputText value="#{facultadControlador.objFacultad.descripcion}"
title="ej: Ninguna"
                                 required="true" requiredMessage="Ingrese una descripción"
converterMessage="VERIFIQUE DESCRIPCION (200 CARACTERES)"/>

                    <h:outputText value="Código Sicoa:"/>
                    <p:inputText value="#{facultadControlador.objFacultad.codigo_sicoa}"
title="ej: 00000"
                                 required="true" requiredMessage="Ingrese codigo sicoa"
converterMessage="VERIFIQUE CODIGO SICOA"/>


                    </p:panelGrid>

                <p:commandButton value="Aceptar"
action="#{facultadControlador.insertarFacultad()}"
                                 update=":frmfacultad:pnlFacultad"/>
                <p:commandButton value="Cancelar"
onclick="wdlgNuevoFacultad.hide()" type="reset"/>
            </h:form>
        </p:dialog>
        <p:dialog id="dlgEditarFacultad" header="Editar Facultad"
widgetVar="wdlgEditarFacultad"
                  modal="true"
                  closable="false" resizable="false" showEffect="clip" hideEffect="fold">
            <h:form id="frmEditarFacultad">
                <p:ajaxStatus onstart="dlgStatus.show();"
oncomplete="dlgStatus.hide();"/>
                <p:messages id="messages" autoUpdate="true"/>
                <p:panelGrid id="pngEditarFacultad" columns="2">
                    <h:outputText value="Nombre:"/>
                    <p:inputText value="#{facultadControlador.facultadSel.nombre}"
title="ej: Ingenieria"
                                 required="true" requiredMessage="Ingrese el Nombre"
converterMessage="VERIFIQUE NOMBRE (200 CARACTERES)"/>
                    <h:outputText value="Descripción:"/>
                    <p:inputText value="#{facultadControlador.facultadSel.descripcion}"
title="ej: Ninguna"
                                 required="true" requiredMessage="Ingrese la Descripción"
converterMessage="VERIFIQUE DESCRIPCION (200 CARACTERES)"/>
                    <h:outputText value="Código Sicoa:"/>
```

```
                </p:panelGrid>
                <p:commandButton value="Aceptar"
action="#{facultadControlador.actualizarFacultad()}"
                                 update=":frmfacultad:pnlFacultad"/>
                <p:commandButton value="Cancelar"
onclick="wdlgEditarFacultad.hide()" type="reset"/>
             </h:form>
          </p:dialog>

            <p:dialog id="dlgEliminarFacultad" header="Eliminar Facultad"
widgetVar="wdlgEliminarFacultad"
                 modal="true" closable="false" resizable="false" showEffect="clip"
hideEffect="fold">
             <h:form id="frmEliminarFacultad">
                <p:ajaxStatus onstart="dlgStatus.show();"
oncomplete="dlgStatus.hide();"/>
                <h:outputText value="¿Desea Eliminar la  Facultad?"/>
                <p:panelGrid id="pngEliminarFacultad" columns="2">
                   <h:outputText value="Nombre:"/>
                   <p:inputText value="#{facultadControlador.facultadSel.nombre}"
title="ej: Gerencia"
                                required="true" requiredMessage="Ingrese el Nombre"
disabled="true"/>
                   <h:outputText value="Descripción:"/>
                   <p:inputText value="#{facultadControlador.facultadSel.descripcion}"
title="ej: ninguna"
                                required="true" requiredMessage="Ingrese Descripción"
disabled="true"/>

                   <h:outputText value="Código Sicoa:"/>
                   <p:inputText value="#{facultadControlador.facultadSel.codigo_sicoa}"
title="ej: 00000"
                                required="true" requiredMessage="Ingrese la Codigo_sicoa"
disabled="true"/>

                </p:panelGrid>
                <p:commandButton value="Aceptar"
action="#{facultadControlador.eliminarFacultad()}"
                                 update=":frmfacultad:pnlFacultad"/>
                <p:commandButton value="Cancelar"
onclick="wdlgEliminarFacultad.hide()" type="reset"/>
             </h:form>
          </p:dialog>
          <p:dialog modal="true" widgetVar="dlgStatus" header="Procesando"
draggable="false" closable="false"
                   resizable="false">
             <p:graphicImage value="/resources/images/ajaxloadingbar.gif" />
          </p:dialog>
       </ui:define>
    </ui:composition>
 </h:body>
</html>
```

ANEXO 5: ESPECIFICACIONES TECNICAS PARA LOS SERVICIOS WEB DE TUTORIAS-PARTE I

Especificaciones Técnicas

Proyecto

Sistema de Seguimiento de Tutorías "UNACH"

2015/02/10

Versión 1.0

Preparado por:

Ing. Diego Palacios C.

Docente Carrera de Ingeniería en Sistemas y Computación

dpalacios@unach.edu.ec

Hoja de revisión y aprobación

Cambios

Fecha	Autor	Versión	Observaciones
2015-02-10	Ing. Diego Palacios C	1.0	Primer lanzamiento

Revisores

Fecha	Nombre	Versión	Observaciones

Descripción del Documento

En este documento se describen los métodos necesarios para la implementación del Sistema de Seguimiento de Tutorías, mismo que se está desarrollando en la Carrera de Ingeniería en Sistemas y Computación como proyecto de Tesis bajo mi dirección y coordinación técnica. Es importante destacar que la información académica que reposa en el SICOA es fundamental para el desarrollo de este proyecto, ya que de esta manera se logrará trabajar con información actualizada y se evitará la duplicación de la misma.

Documentos Anexos

A este documento se anexa el diccionario de datos, elemento que da la reseña completa del tipo de dato esperado en los insumos.

Descripción de Métodos del Webservice TUTORIAS

El Contexto Técnico para la generación de los insumos llamados WebServices (WS), deberán contener todos los métodos que serán indicados y este debe ser desarrollado en WSDL, por ejemplo http://appjb06.unach.com:8380/segTutorias/infoAcademico?wsdl , el cual es el estándar XML que se usa para serializar la información y enviarla.

Resumen de Datos

OUT: define los datos que son devueltos por el web service como atributos de una clase (Response).
IN: define los datos que son enviados como parámetros para consultar al web service.
Fechas:
OUT String con formato "YYYY-MM-DD HH:MM:SS"
IN String con formato "DD-MM-YYYY".
Decimales:
OUT BigDecimal de dos decimales.
Cadenas de texto:
OUT: Cadena de texto normal.

LISTADO Y DESCRIPCIÓN DE MÉTODOS.

MÉTODO "getFacultades()"

Descripción

Devuelve la lista de Facultades del SICOA. La consulta es realizada en base a SQL directo.

Nombre método:	getFacultades()
Descripción:	Obtiene la lista de Facultades del SICOA.
Parámetro recibido	Ninguno

Valor retornado		
Descripción		**Retorna la lista de Objetos Facultades**
código	String	Código de la Facultad
nombre	String	Nombres de la Facultad

MÉTODO "getCarrerasFacultad()"

Descripción

Devuelve la lista de Carreras del SICOA dada una Facultad. La consulta es realizada en base a SQL directo.

Nombre método:	getCarrerasFacultad ()
Descripción:	Obtiene la lista de Carreras dada una Facultad.
Parámetro recibido	Código del Facultad (String)

Valor retornado		
Descripción		**Retorna la lista de Objetos Carreras.**
código	String	Código de la Carrera
nombre	String	Nombre de la Carrera

MÉTODO "getNivelesCarrera()"

Descripción

Devuelve la lista de Niveles del SICOA dada una Carrera. La consulta es realizada en base a SQL directo.

Nombre método:	getNivelesCarrera()
Descripción:	Obtiene la lista de Niveles dada una Carrera.
Parámetro recibido	Código de la Carrera (String)

Valor retornado		
Descripción		**Retorna la lista de Objetos Niveles.**
código	String	Código del Nivel
nombre	String	Nombre del Nivel
paralelo	String	Nombres del Docente
modalidad	String	Nombre de Modalidad(semestral/anual)

MÉTODO "getPeriodos()"

Descripción

Devuelve la lista de Períodos del SICOA. La consulta es realizada en base a SQL directo.

Nombre método:	getPeriodos()
Descripción:	Obtiene la lista de Períodos.
Parámetro recibido	Ninguno

Valor retornado		
Descripción		**Retorna la lista de Objetos Períodos.**
código	String	Código del Período
nombre	String	Nombre del Período
Fecha Inicio	String	Fecha de inicio del Periodo
Fecha Fin	String	Fecha fin del Periodo
estado	int	Estado del periodo (activo=1 o inactivo=0)
tipo	Int	Tipo del Período (anual=0 o semestral=1)

MÉTODO "getPeriodoVigenteCarrera()"

Descripción

Devuelve el período vigente de una Carrera dado el código de la carrera y la modalidad. La consulta es realizada en base a SQL directo.

Nombre método:	getPeriodoVigenteCarrera()
Descripción:	Obtiene el período actual de una carrera.
Parámetro recibido	Código de la carrera (String), Modalidad de la Carerra (int anual=0 o semestral=1)

Valor retornado		
Descripción		**Retorna la Clase Periodo.**
código	String	Código del Período
nombre	String	Nombre del Período
Fecha Inicio	String	Fecha de inicio del Periodo
Fecha Fin	String	Fecha fin del Periodo
estado	int	Estado del periodo (activo=1 o inactivo=0)
tipo	Int	Tipo del Período (anual=0 o semestral=1)

MÉTODO "getDocente()"

Descripción

Devuelve los datos del docente dado su identificación (cédula). La consulta es realizada en base a SQL directo.

Nombre método:	getDocente ()
Descripción:	Obtiene los datos del docente dado su identificación.
Parámetro recibido	Cédula o identificación (String)

Valor retornado		
Descripción		**Retorna la Clase Docente.**
apellidos	String	Apellidos del Docente
nombres	String	Nombres del Docente
identificación	String	Identificación del Docente
mail	String	Mail del Docente

MÉTODO "getDocentesCarrera()"

Descripción

Devuelve una lista de docentes dado la Carrera. La consulta es realizada en base a SQL directo.

Nombre método:	getDocentesCarrera()
Descripción:	Obtiene una lista de docentes dado la Carrera.
Parámetro recibido	Código de la Carrera (String)

Valor retornado		
Descripción		**Retorna la lista de Objetos Docentes.**
apellidos	String	Apellidos del Docente
nombres	String	Nombres del Docente
identificación	String	Identificación del Docente
mail	String	Mail del Docente

MÉTODO "getEstudiantesMatriculadosCarrera()"

Descripción

Devuelve una lista de estudiantes matriculados en una carrera, nivel, modalidad del período vigente. La consulta es realizada en base a SQL directo.

Nombre método:	getEstudiantesMatriculadosCarrera ()
Descripción:	Obtiene una lista de estudiantes matriculados en una carrera, nivel, modalidad del período vigente.
Parámetro recibido	Código de la Carrera (String), Código del Nivel (String), Modalidad (String semestral/anual)

Valor retornado		
Descripción		**Retorna la lista de Objetos Estudiantes.**
apellidos	String	Apellidos del Estudiante
nombres	String	Nombres del Estudiante
identificación	String	Identificación del Estudiante
mail	String	Mail del Estudiante
codigo	String	Código del estudiante

MÉTODO "getEstudiante()"

Descripción

Devuelve los datos del Estudiante dado su código y contraseña del SICOA. La consulta es realizada en base a SQL directo.

Nombre método:	getEstudiante()
Descripción:	Devuelve los datos del Estudiante dado su código y contraseña del SICOA
Parámetro recibido	Código SICOA(String), Contraseña (String)

Valor retornado		
Descripción		**Retorna la Clase Estudiante.**
apellidos	String	Apellidos del Estudiante
nombres	String	Nombres del Estudiante
identificación	String	Identificación del Estudiante
mail	String	Mail del Estudiante
codigo	String	Código del estudiante

ANEXO 6: ACTA ENTREGA-RECEPCION DE LOS SERVICIOS WEB

UTECA	PROCESO: ENTREGA DE PRODUCTO	Fecha de elaboración: 10 de abril de 2015	*Página 1 de 3*
	ACTA DE ENTREGA Y RECEPCION		

ACTA DE ENTREGA Y RECEPCION SERVICIO WEB PARA EL SISTEMA DE SEGUIMIENTO DE TUTORIAS

Proyecto	Web Service para integración con el Sistema de Tutorías
Cliente	Universidad Nacional del Chimborazo
Representante del Cliente	Ing. Danny Velasco – Carrera de Ingeniería en Sistemas y Computación
Representante de UTECA	Ing. Pedro Orozco Q
Fecha	10/04/2015

Por medio de la presente le informo que el día de hoy se culmina con las siguientes fases del proyecto:

- Planificación
- Análisis
- Diseño
- Desarrollo
- Pruebas
- Puesta en Produccion
- Cierre

Se realiza también la entrega formal del Web Service para integración con el Sistema de Tutorías el cual contiene los siguientes métodos:

- getFacultades()
- getCarrerasFacultad()
- getNivelesCarrera()
- getPeriodos()
- getPeriodoVigenteCarrera()
- getDocente()
- getDocentesCarrera()
- getEstudiantesMatriculadosCarrera()
- getEstudiante()

Esta entrega se hace conforme las especificaciones establecidas en el documento de Especificaciones Técnicas elaborado por el Ing. Diego Palacios.

La fase de transición/liberación se ha llevado a cabo. Las fuentes ya están instalados y funcionando en los ambientes solicitados. Las revisiones se efectuaron en sitio y vía remota con los siguientes resultados:

MÓDULO / METODO	COD REQ	REQUERIMIENTO	ESTADO(No iniciado, En proceso, Ejecutado)
getFacultades()	REQ-001	Consulta de facultades	EJECUTADO
getCarrerasFacultad()	REQ-002	Consulta de las carreras de las facultades	EJECUTADO
getNivelesCarrera()	REQ-003	Consulta de los niveles de la carrera	EJECUTADO
getPeriodos()	REQ-004	Consulta de periodos	EJECUTADO

getPeriodoVigenteCarrera()	REQ-005	Consulta del periodo vigente	EJECUTADO
getDocente()	REQ-006	Consulta de los datos del docente	EJECUTADO
getDocentesCarrera()	REQ-007	Consulta de los docentes por carrera	EJECUTADO
getEstudiantesMatriculadosCarrera()	REQ-008	Consulta de estudiantes matriculados por carrera	EJECUTADO
getEstudiante()	REQ-009	Consulta de los datos por estudiante	EJECUTADO

RESTRICCIONES DE USO DEL SERVICIO WEB:

Este servicio solamente podrá ser utilizado para consulta de datos dentro del Sistema de Seguimiento de Tutorías. Se prohíbe el uso en otros sistemas sin previa autorización de la Unidad Técnica de Control Académico.

Ing. Pedro Orozco Q.
Coordinador de UTECA

Ing. Danny Velasco
Dir. Ingeniería en Sistemas

Ing. Carlos Padilla
Programador UTECA

Srta. Janneth Guamán
Estudiante de ISYC

Srta. Sandra Contento
Estudiante de ISYC

ANEXO 7: ESPECIFICACIONES TECNICAS PARA EL NUEVO METODO DE TUTORIAS-PARTE II

Especificaciones Técnicas

Proyecto

Sistema de Seguimiento de Tutorías "UNACH"

2015/04/29

Versión 1.0

Preparado por:

Ing. Diego Palacios C.

Docente Carrera de Ingeniería en Sistemas y Computación

dpalacios@unach.edu.ec

Hoja de revisión y aprobación

Cambios

Fecha	Autor	Versión	Observaciones
2015-04-29	Ing. Diego Palacios C	1.0	Primer lanzamiento

Revisores

Fecha	Nombre	Versión	Observaciones

Descripción del Documento

En este documento se describen el método necesario para la implementación del Sistema de Seguimiento de Tutorías, mismo que se está desarrollando en la Carrera de Ingeniería en Sistemas y Computación como proyecto de Tesis bajo mi dirección y coordinación técnica. Es importante destacar que la información académica que reposa en el SICOA es fundamental para el desarrollo de este proyecto, ya que de esta manera se logrará trabajar con información actualizada y se evitará la duplicación de la misma.

Documentos Anexos

A este documento se anexa el diccionario de datos, elemento que da la reseña completa del tipo de dato esperado en los insumos.

3 Descripción del Métodos del Webservice TUTORIAS

El Contexto Técnico para la generación de los insumos llamados WebServices (WS), deberán contener todos los métodos que serán indicados y este debe ser desarrollado en WSDL, por ejemplo http://appjb06.unach.com:8380/segTutorias/infoAcademico?wsdl , el cual es el estándar XML que se usa para serializar la información y enviarla.

Resumen de Datos

OUT: define los datos que son devueltos por el web service como atributos de una clase (Response).
IN: define los datos que son enviados como parámetros para consultar al web service.
Fechas:
OUT String con formato "YYYY-MM-DD HH:MM:SS"
IN String con formato "DD-MM-YYYY".
Decimales:
OUT BigDecimal de dos decimales.
Cadenas de texto:
OUT: Cadena de texto normal.

LISTADO Y DESCRIPCIÓN DEL MÉTODO.

MÉTODO "getEstudiantesPromedioGeneral()"

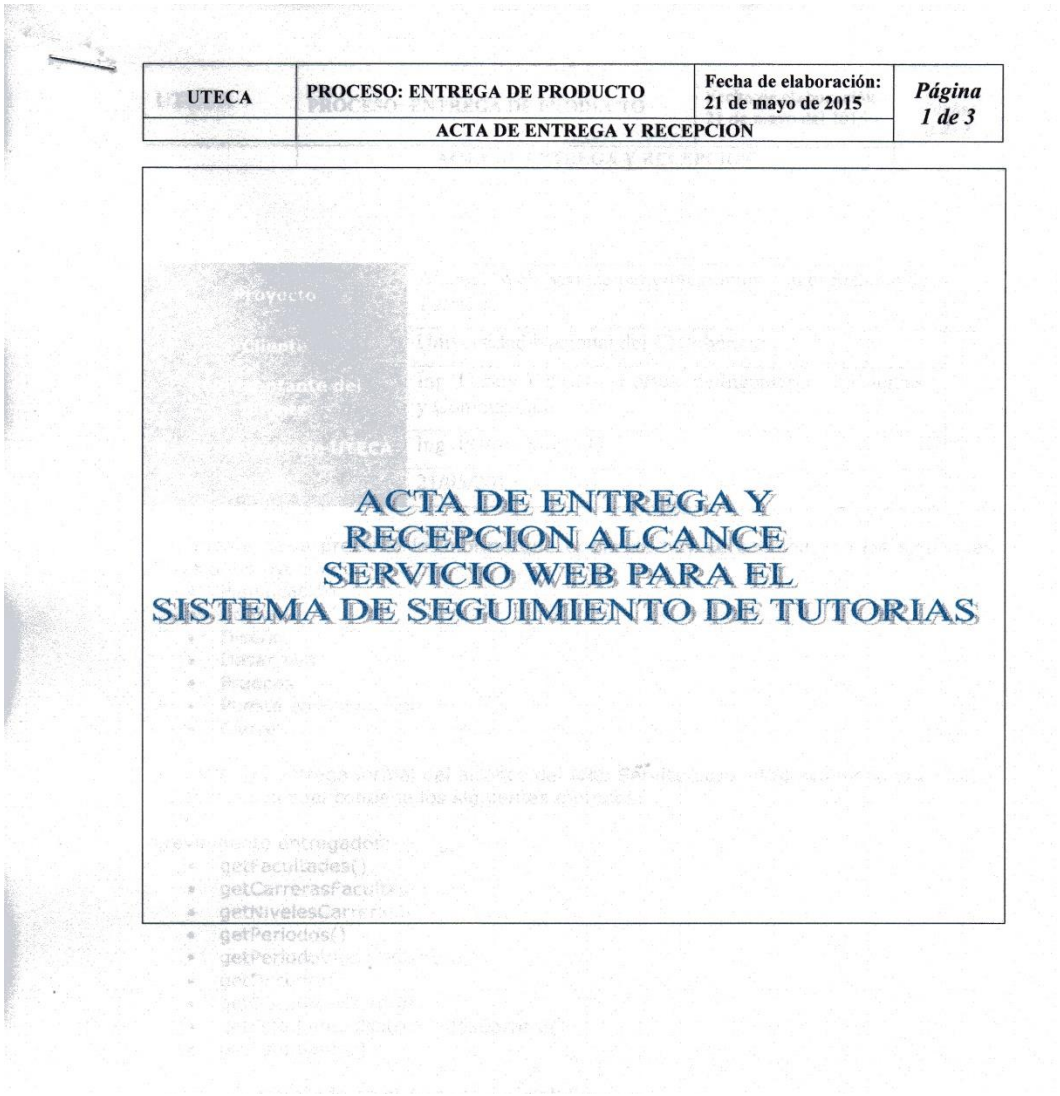
Descripción

Devuelve una lista de estudiantes con su promedio final de las asignaturas en las que se matriculó, en una carrera de un nivel de un período académico específico. La consulta es realizada en base a SQL directo.

Nombre método:	getEstudiantesPromedioGeneral()
Descripción:	Devuelve una lista de estudiantes con su promedio final de las asignaturas en las que se matriculó, en una carrera de un nivel de un período académico específico.
Parámetro recibido	Código del Nivel (String), Código de carrera (String), Código del Periodo(String), Modalidad (String semestral/anual)

Valor retornado		
Descripción		**Retorna la lista de Objetos Estudiantes.**
codigo	String	Código del estudiante
identificación	String	Identificación del Estudiante
apellidos	String	Apellidos del Estudiante
nombres	String	Nombres del Estudiante
mail	String	Mail del Estudiante
promedio_general	String	Promedio general del estudiante en ese nivel matriculado

ANEXO 8: ACTA ENTREGA RECEPCION – ALCANCE SERVICIO WEB PARA EL SISTEMA DE GESTION DE TUTORIAS



UTECA	PROCESO: ENTREGA DE PRODUCTO	Fecha de elaboración: 21 de mayo de 2015	*Página 1 de 3*
	ACTA DE ENTREGA Y RECEPCION		

ACTA DE ENTREGA Y RECEPCION ALCANCE SERVICIO WEB PARA EL SISTEMA DE SEGUIMIENTO DE TUTORIAS

Proyecto	Alcance Web Service para integración con el Sistema de Tutorías.
Cliente	Universidad Nacional del Chimborazo
Representante del Cliente	Ing. Danny Velasco – Carrera de Ingeniería en Sistemas y Computación
Representante de UTECA	Ing. Pedro Orozco Q
Fecha	21/05/2015

Por medio de la presente le informo que el día de hoy se culmina con las siguientes fases del proyecto:

- Planificación
- Análisis
- Diseño
- Desarrollo
- Pruebas
- Puesta en Produccion
- Cierre

Se realiza la entrega formal del alcance del Web Service para integración con el Sistema de Tutorías el cual contiene los siguientes métodos:

Previamente entregados:

- getFacultades()
- getCarrerasFacultad()
- getNivelesCarrera()
- getPeriodos()
- getPeriodoVigenteCarrera()
- getDocente()
- getDocentesCarrera()
- getEstudiantesMatriculadosCarrera()
- getEstudiante()

Método Desarrollado en el Alcance del Web Service:

- getEstudiantesPromedioGeneral()

Esta entrega se hace conforme las especificaciones establecidas en el documento de Especificaciones Técnicas elaborado por el Ing. Diego Palacios y solicitados por el Ing. Danny Velasco con Oficio No. 168-CISYC-2015.

La fase de transición/liberación se ha llevado a cabo. Las fuentes ya están instalados y funcionando en los ambientes solicitados. Las revisiones se efectuaron en sitio y vía remota con los siguientes resultados:

MÓDULO / METODO	COD REQ	REQUERIMIENTO	ESTADO(No iniciado, En proceso, Ejecutado)	OBSERVACIÓN
getFacultades()	REQ-001	Consulta de facultades	EJECUTADO	Previamente Entregado
getCarrerasFacultad()	REQ-002	Consulta de las carreras de las facultades	EJECUTADO	Previamente Entregado
getNivelesCarrera()	REQ-003	Consulta de los niveles de la carrera	EJECUTADO	Previamente Entregado
getPeriodos()	REQ-004	Consulta de periodos	EJECUTADO	Previamente Entregado
getPeriodoVigenteCarrera()	REQ-005	Consulta del periodo vigente	EJECUTADO	Previamente Entregado
getDocente()	REQ-006	Consulta de los datos del docente	EJECUTADO	Previamente Entregado
getDocentesCarrera()	REQ-007	Consulta de los docentes por carrera	EJECUTADO	Previamente Entregado
getEstudiantesMatriculadosCarrera()	REQ-008	Consulta de estudiantes matriculados por carrera	EJECUTADO	Previamente Entregado
getEstudiante()	REQ-009	Consulta de los datos por estudiante	EJECUTADO	Previamente Entregado
getEstudiantesPromedioGeneral	**REQ-0010**	**Consulta de estudiantes con su promedio general por nivel, periodo, carrera y modalidad**	**EJECUTADO**	**Alcance del Servicio Web**

RESTRICCIONES DE USO DEL SERVICIO WEB:

Este servicio solamente podrá ser utilizado para consulta de datos dentro del Sistema de Seguimiento de Tutorías. Se prohíbe el uso en otros sistemas sin previa autorización de la Unidad Técnica de Control Académico.

Ing. Pedro Orozco Q.
Coordinador de UTECA

Ing. Danny Velasco
Dir. Ingeniería en Sistemas

Ing. Carlos Padilla
Programador UTECA

Srta. Janneth Guamán
Estudiante de ISYC

Srta. Sandra Contento
Estudiante de ISYC

ANEXO 9: HERRAMIENTA NEOLOAD

NEOLOAD

Pruebas de aplicaciones Web

NeoLoad está diseñado para aplicaciones web. NeoLoad tiene un historial de apoyo a las últimas tecnologías web y continúa en la vanguardia de la nueva tecnología, incluyendo soporte Push, WebSocket y HTTP streaming, posicionamiento NeoLoad como solución líder para la carga de aplicaciones web y pruebas de rendimiento.

Nuestros clientes: los principales proveedores de aplicaciones web y algunos de los sitios web más grandes del mundo han seleccionado NeoLoad para probar el rendimiento de sus aplicaciones bajo carga.

Los problemas de rendimiento son comunes y tienen una amplia variedad de causas, incluyendo:

- Problemas de configuración de software (para el servidor web, base de datos, balanceadores de carga, etc.)
- Configuración de red Pobre
- Código de software (mal optimizado o no permitir el acceso simultáneo)
- La insuficiencia de los recursos de hardware

La única manera de determinar con precisión estos problemas de rendimiento antes de una aplicación web entra en producción es mediante la simulación de un gran número de usuarios simultáneos (pruebas de carga, pruebas de tensión).

Usando sólo un navegador Web estándar, NeoLoad es capaz de registrar las acciones comerciales realizadas dentro de una aplicación web o una página web, como la presentación de un formulario o realizar una búsqueda. Estas acciones pueden ser reproducidas por el mayor número de usuarios virtuales que se requieran para simular la carga que el servidor tendrá que soportar. El informe de la prueba entonces le dirá si se han cumplido sus objetivos de confiabilidad y rendimiento

predeterminados. Si no lo tienen, NeoLoad ayuda a identificar las áreas que necesitan mejorar.

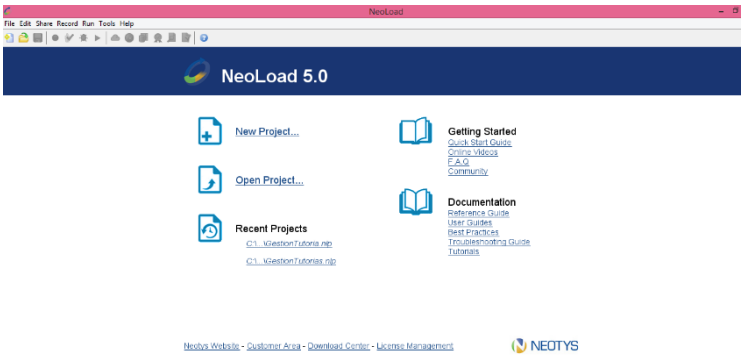
Con NeoLoad, ahora se puede implementar la aplicación web en el tiempo y con una fiabilidad del 100%.

UTILIZACION DE LA HERRAMIENTA NEOLOAD

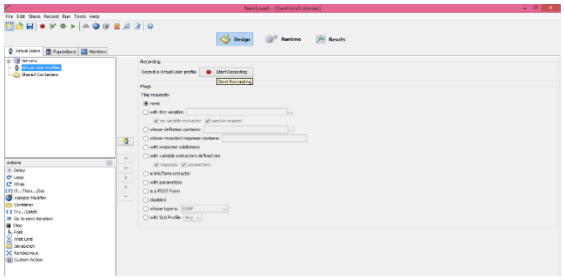
Una vez que se crearon los prototipos se procederá a crear las pruebas utilizando la herramienta Neoload en donde se describirá el funcionamiento y los resultados obtenidos de las pruebas.

Para medir la longitud de la página y la respuesta AJAX se debe capturar el prototipo a medir, es decir grabar todo el proceso, la herramienta grabara los pasos realizados desde que se carga la página hasta que finalice el prototipo.

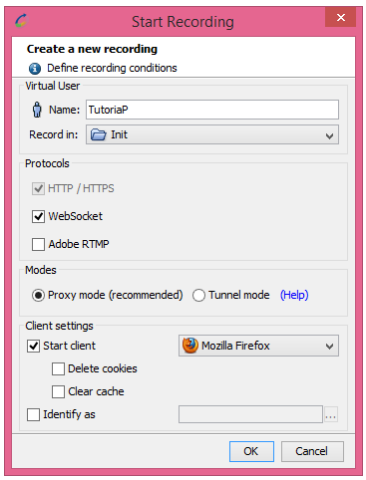
1. Abrir la herramienta Neoload y seleccionar la opción **New Project** como se muestra en la imagen.



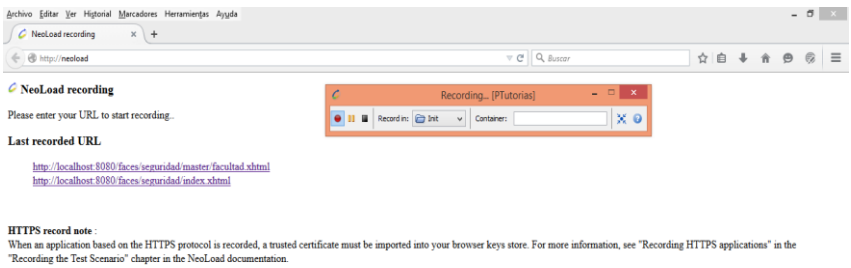
2. A continuación aparecerá la siguiente pantalla donde se seleccionará la opción **Start Recording**



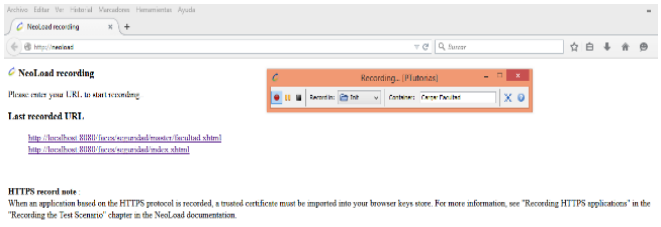
3. En el siguiente cuadro de dialogo se especificará el nombre del usuario virtual, en **Record in:** seleccionar Init que hace referencia a la parte inicial del sistema en donde empezara la captura, también se puede seleccionar el tipo de navegador en donde se abrirá la página a grabar, continuamente pulsar **OK**.



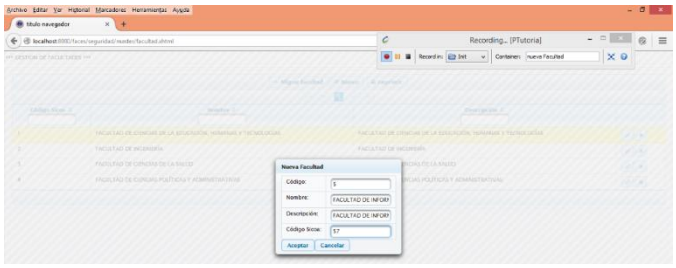
4. A continuación se abrirá el navegador predeterminado en donde indicará todas las direcciones que se han abierto recientemente seguidamente pulsamos sobre la dirección a grabar.



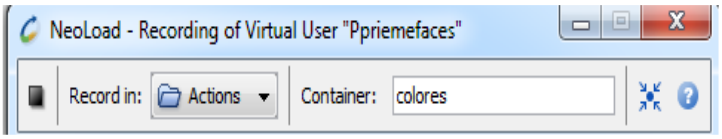
5. Como se observa en la parte superior se encuentra una ventana grabando desde el inicio en la casilla **Container** se indicará un nombre que servirá para identificar que parte del sistema está en funcionamiento en ese instante.



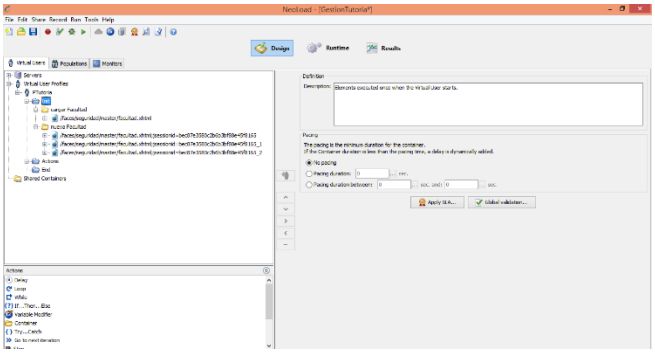
6. Al momento de realizar la gestión nueva, editar o eliminar se debe especificar que parte se está grabando, introducir un nombre que identifique que proceso se está realizando.



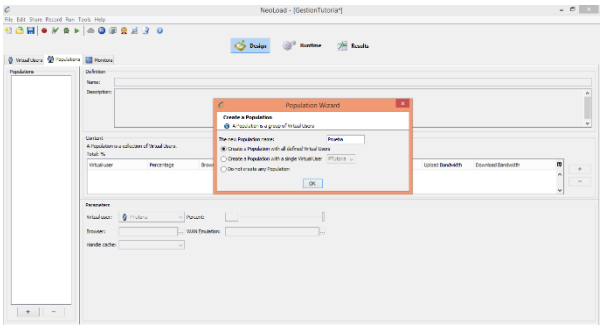
7. Una vez que haya terminado de grabar los procesos se debe parar la grabación haciendo clic en el botón **STOP** como se muestra en la imagen.



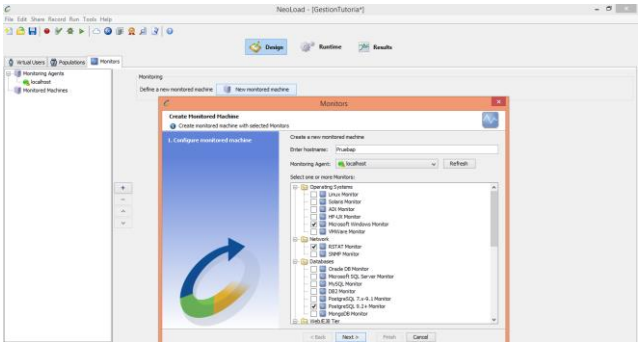
8. Una vez que hayas terminado se debe esperar a que termine la creación del entorno virtual, al finalizar la grabación se mostrará una ventana con todos los pasos que se capturan al momento de grabar, estos se capturan por procesos individuales.



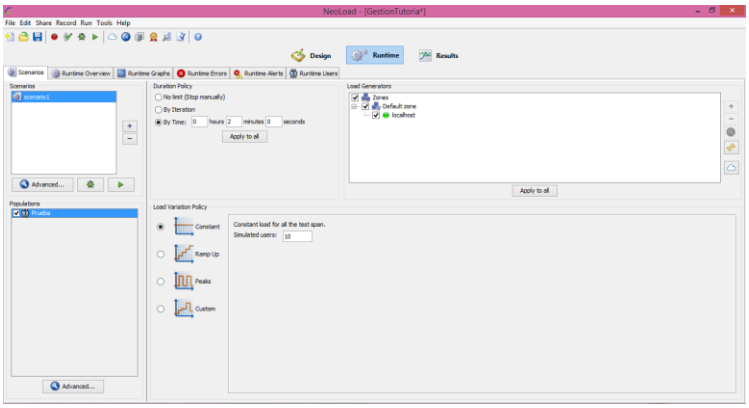
9. El siguiente paso es crear la población es decir el número de usuarios virtuales para utilizar el sistema durante la simulación de uso. Para ello seleccionar **POPULATIONS** y aparecerá una nueva ventana en donde pedirá un nombre para los usuarios virtuales, por defecto la población se creará para 10 usuarios, aunque se puede incrementar esta cantidad si desea y pulsar **OK.**



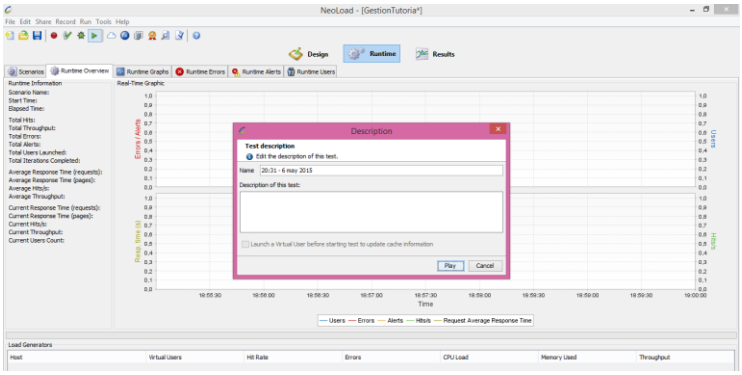
10. Ahora se procederá a crear los monitores que se va a utilizar, seleccionar los monitores para el sistema operativo, red y base de datos y pulsar **NEXT.**



11. Para la configuración de ejecución de pruebas, una vez terminado el diseño dar clic en **RUNTIME** en donde se observa el escenario de prueba así como la población que son 10 usuarios virtuales.

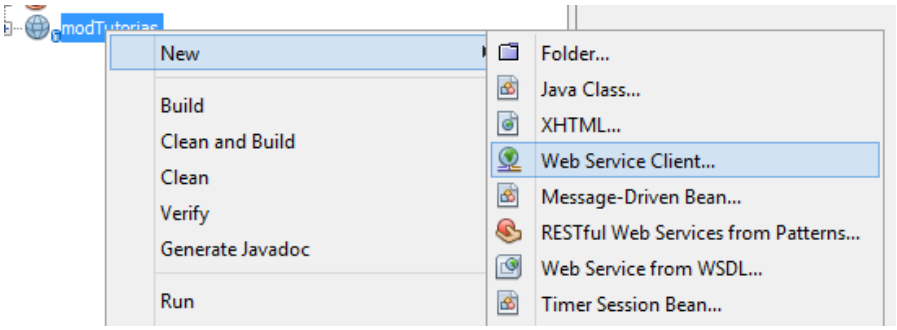


12. A continuación pulsamos **Run** y aparecerá una ventana emergente que pide ingresar un nombre de reporte que por defecto será la fecha y una descripción y pulsamos **OK.**

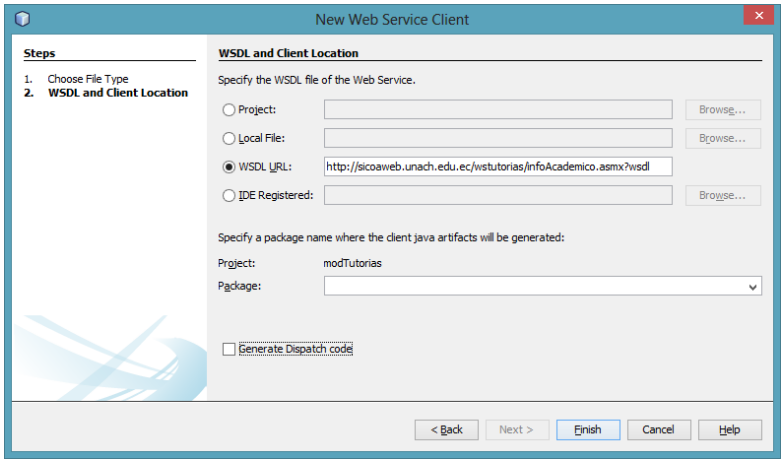


ANEXO 9: CONSUMIR SERVICIOS WEB

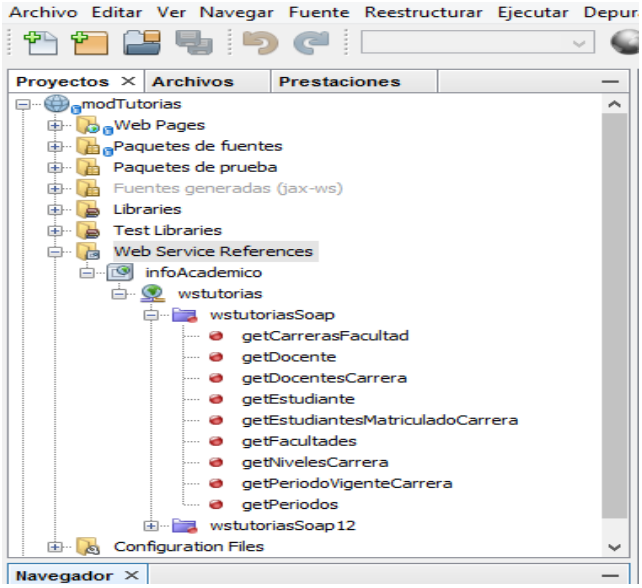
Para consumir los servicos web generados desde la base de datos SICOA se debe crear un nuevo Web Service Client como se muestra en la imagen.



A continuación se debe ingresar la WSDL URL que es la dirección de los servicios web proporcionada y damos clic en finalizar como se muestra en la imagen.



Como se puede observar se generan todos los métodos que se solicitaron para el desarrollo del sistema de Gestion de Tutorías.



ANEXO 10: REGLAMENTO DE TUTORIAS ACADEMICAS DE LA UNACH

UNIVERSIDAD NACIONAL DE CHIMBORAZO

VICERRECTORADO ACADÉMICO

UNIDAD DE PLANIFICACIÓN ACADÉMICA

CONSIDERANDO:

Que, la Constitución de la República en el Art. 350, establece que: *"El Sistema de Educación Superior tiene como finalidad la formación académica y profesional con visión científica y humanista".*

Que, la Ley Orgánica de Educación Superior (LOES) establece:

Artículo 5 literal b): Acceder a una educación superior de calidad y pertinente, que permita iniciar una carrera académica y/o profesional en igualdad de oportunidades.

Artículo 5 literal h): El derecho a recibir una educación superior laica, intercultural, democrática incluyente y diversa, que impulse la equidad de género, la justicia y la paz.

Artículo 71: El principio de igualdad de oportunidades consiste en garantizar a todos los actores del Sistema de Educación Superior las mismas posibilidades en el acceso, pertinencia, movilidad y egreso del sistema, sin discriminación de género, credo, orientación sexual, etnia, cultura, preferencia política, condición socioeconómica o discapacidad.

Artículo 86: Las instituciones de educación superior mantendrán una unidad administrativa de Bienestar Estudiantil destinada a promover la orientación vocacional y profesional, la obtención de créditos, estímulos, ayudas económicas y becas, y ofrecer los servicios asistenciales que se determinen en las normativas de cada institución. Esta unidad, además, se encargará de promover un ambiente de respeto a los derechos y a la integridad física, psicológica y sexual de las y los estudiantes, en un ambiente libre de violencia y brindará asistencia a quienes demanden por violaciones de estos derechos.

Que, el Reglamento de Régimen Académico del Consejo de Educación Superior (CES) Art. 15 componente 1 lit. b) 2do. Párrafo, dice: *Son actividades de aprendizaje colaborativo, entre otras: la sistematización de prácticas de investigación-intervención, proyectos de integración de saberes, construcción de modelos y prototipos, proyectos de problematización y resolución de problemas o casos. Estas actividades deberán incluir procesos colectivos de organización del aprendizaje con el uso de diversas tecnologías de la información y la comunicación, así como metodologías en red, tutorías in situ o entornos virtuales.*

Que, el Reglamento de Carrera y Escalafón del Profesor e Investigador de la Universidad Nacional de Chimborazo, en el Art. 4 al referirse a actividades de docencia en el Numeral 4, *establece domo actividad la "Orientación y acompañamiento a través de tutorías presenciales o virtuales, individuales o grupales".*

Que, en el Modelo General de Evaluación de Carreras con fines de Acreditación, Criterio E indicador E3, dice: *"Cada estudiante de la carrera debe contar con un profesor-tutor asignado por la carrera, evaluar periódicamente su rendimiento y monitorear su progreso con el fin de facilitar su éxito en la consecución de los resultados o logros del aprendizaje".*

El H. Consejo Universitario, en ejercicio de las atribuciones que le confiere la ley resuelve expedir el siguiente:

REGLAMENTO DE TUTORÍAS ACADÉMICAS DE LA UNIVERSIDAD NACIONAL DE CHIMBORAZO

CAPÍTULO I

DE LA TUTORÍA

Art. 1.- La tutoría, siendo una función de los docentes, se entiende como un proceso educativo dirigido a estudiantes, con el objetivo de orientar, atender y acompañar de manera sistemática, de forma grupal y preferentemente individual, el desarrollo académico universitario, mediante programas, técnicas de enseñanzas apropiadas y conformación de equipos de trabajo, conforme a criterios y mecanismos de monitoreo y control, a fin de fortalecer su formación integral en la toma de decisiones académicas, administrativas, profesionales y personales.

Art. 2.- Fundamentadas en el desarrollo de las competencias curricularmente declaradas y expresadas en logros de aprendizaje, los ámbitos de aplicación de las tutorías son los siguientes:

a. **Ámbito Académico:** Es una actividad inherente al docente, que se relaciona con la calidad del proceso educativo. Orienta sobre la planificación micro curricular, dificultades de aprendizaje, estilos de aprendizaje, hábitos de estudio, entre otros;

b. **Ámbito Administrativo:** Asesoramiento sobre trámites administrativos, tales como matrículas, becas, derechos, prácticas pre-profesionales, exenciones, beneficios, obligaciones, entre otros;

c. **Ámbito Profesional:** Orienta al desarrollo de un proyecto de vida, innovación y emprendimiento que responda a las necesidades y generación espacios laborales; y,

d. **Ámbito Personal:** Se refiere a la identificación de características de la personalidad, motivacionales, actitudinales y afectivas, que inciden en su desempeño académico, inclusión e interacción social, así como las características, sociales, económicas, culturales y familiares de los estudiantes, lo que podrían incidir en la deserción, ausentismo y repitencia escolar.

Art. 3.- Son objetivos de la tutoría:

a. Orientar la formación integral del estudiante para potenciar sus capacidades que incidan en su beneficio personal y social, con un alto sentido de responsabilidad y solidaridad; reduciendo los factores de riesgo que puedan afectar a su permanencia en la Universidad;

b. Mejorar la calidad del proceso formativo en el ámbito de la construcción de valores, actitudes y hábitos;

c. Orientar al estudiante para que desarrolle estrategias y técnicas de estudio, así como actividades extracurriculares con el propósito de mejorar su desempeño académico propendiendo a disminuir los niveles de deserción, repitencia y ausentismo;

d. Promover acciones que permitan la integración de los estudiantes a la institución, brindándoles un ambiente propicio;

e. Asistir a los estudiantes con dificultades académicas, administrativas, profesionales y personales desde la admisión y durante la permanencia en la universidad;

f. Aprovechar las oportunidades derivadas del uso de nuevas tecnologías, en el diseño y aplicación de estrategias de aproximación entre estudiantes y profesores, que propicien un mejor clima en el proceso de enseñanza aprendizaje y un mayor conocimiento de los problemas y expectativas de los estudiantes; y,

g. Retroalimentar la práctica docente mediante un diálogo efectivo entre profesores y estudiantes, que generen alternativas de atención e incidan en la formación profesional integral.

Art. 4.- Son modalidades de acción tutelares las siguientes:

a. **Tutoría individual:** Asistencia individualizada al estudiante;

b. **Tutoría grupal:** Dirigida al colectivo del curso y pequeños grupos de estudiantes; y,

c. **Tutoría en línea:** Asistencia individual o grupal mediante el uso de las TICS.

CAPÍTULO II

DE LAS COMISIONES DE TUTORÍAS

Art. 5.- Las Facultades de la Universidad Nacional de Chimborazo, contarán con una comisión de tutorías, conformadas de la siguiente manera:

a. Los Subdecanos, quienes lo presidirán; y,
b. Los Directores de Carrera.

Art. 6.- Son responsabilidades de las Comisiones de Tutorías las siguientes:

a. Coordinar y supervisar las actividades de tutoría en sus Facultades; y,
b. Presentar informes semestrales respecto a las actividades desarrolladas por los tutores al Vicerrectorado Académico, a través de los decanatos.

Art. 7.- El Director de Carrera coordinará con el DEBEYU, Departamento Médico, Responsable de la Bolsa de Empleo y otras instancias académicas y administrativas institucionales, cuando se presenten situaciones que

ameriten un tratamiento académico, administrativo, profesional y personal.

Art. 8.- El Director de Carrera será el responsable de supervisar y coordinar las actividades de los tutores y gestionar los recursos necesarios para el cumplimiento de la tutoría.

Art. 9.- Todos los docentes son responsables de la tutoría de sus estudiantes, de acuerdo al Art. 6 del Reglamento de Carrera y Escalafón del Profesor e Investigador del Sistema de Educación Superior.

CAPÍTULO III

DE LOS TUTORES

Art. 10.- El tutor es el profesor que asume el compromiso de ser guía del proceso formativo en el ámbito académico, administrativo, profesional y personal de los estudiantes bajo su tutela.

Art. 11.- Son funciones de los tutores:

a. Contribuir a la disminución de la repitencia estudiantil y al incremento de la eficiencia terminal de acuerdo al Art.1 lit. c, del Reglamento de Evaluación del Desempeño Docente de la UNACH;
b. Estimular, orientar y acompañar a los estudiantes en el proceso de aprendizaje, para su desarrollo y formación integral;
c. Aplicar los instrumentos de tutoría: ficha de identificación, seguimiento de tutoría, informe mensual del estudiante para el seguimiento respectivo;
d. Identificar problemas individuales que afectan el desempeño de los estudiantes y coordinar con el Director de Carrera para la intervención correspondiente;
e. Potenciar la tutoría entre pares, debates, intercambio de experiencias, desarrollo de proyectos;
f. Propiciar ambientes democráticos que potencialicen el fortalecimiento de su identidad y la diversidad cultural, estimulando la confianza y la cooperación;
g. Asistir a las reuniones convocadas por la Comisión de Tutoría de cada Facultad, cuando ésta así lo requiera;
h. Realizar el seguimiento del sílabo y logros de aprendizaje del estudiante;
i. Presentar su plan de trabajo al Director de carrera enmarcado en los lineamientos de este reglamento;
j. Informar al final de cada período académico al Director de carrera sobre la actividad tutorial desarrollada;
k. Llevar un registro y seguimiento de la actividad tutorial;
l. Mantener confidencialidad de la información manejada con los estudiantes; y,
m. Cumplir con las actividades concernientes a la tutoría en las instalaciones de la Universidad y/o entornos virtuales.

Art. 12.- Los tutores tienen los siguientes derechos:

a. Recibir la capacitación pertinente;
b. Se les asigne en los distributivos de trabajo, actividades de tutoría de acuerdo a los tiempos de dedicación que corresponda a cada uno de los docentes, para tutoría individual, grupal o virtual;
c. Contar con espacios físicos adecuados para la atención a los estudiantes; y,
d. Contar con material de apoyo suficiente para el buen desarrollo de su función.

CAPÍTULO IV

DE LOS ESTUDIANTES

Art. 13.- Son obligaciones:

a. Cumplir con las disposiciones emanadas por las autoridades y demás organismos institucionales, de acuerdo al contenido del presente reglamento;
b. Asistir puntualmente al desarrollo de las actividades programadas por el Tutor;
c. Mantener comunicación armónica con el Tutor;
d. Cumplir responsablemente con las actividades dispuestas en su tutoría; y,
e. Colaborar activamente en el desarrollo de la programación tutorial.

Art. 14.- Son deberes:

a. Recibir tutoría académica, administrativa, personal y profesional durante su formación universitaria, de manera oportuna y eficiente;
b. Tener acceso a los espacios, los recursos y el tiempo para el desarrollo de las tutorías; y,

c. Contar con una asesoría tutorial de calidad y calidez.

Art. 15.- Serán susceptibles de evaluación las actividades de tutoría planteadas por el tutor y coordinadas de con el Director de Carrera y la institución (actividades académicas, administrativas, profesionales y personales).

Art. 16.- La tutoría para los estudiantes del posgrado será regulada por su propio Reglamento.

DISPOSICION TRANSITORIA

PRIMERA.- Las situaciones no previstas en este Reglamento serán resueltas por el Consejo Directivo de la Facultad, Comisión General Académica y Consejo Universitario, en su orden.

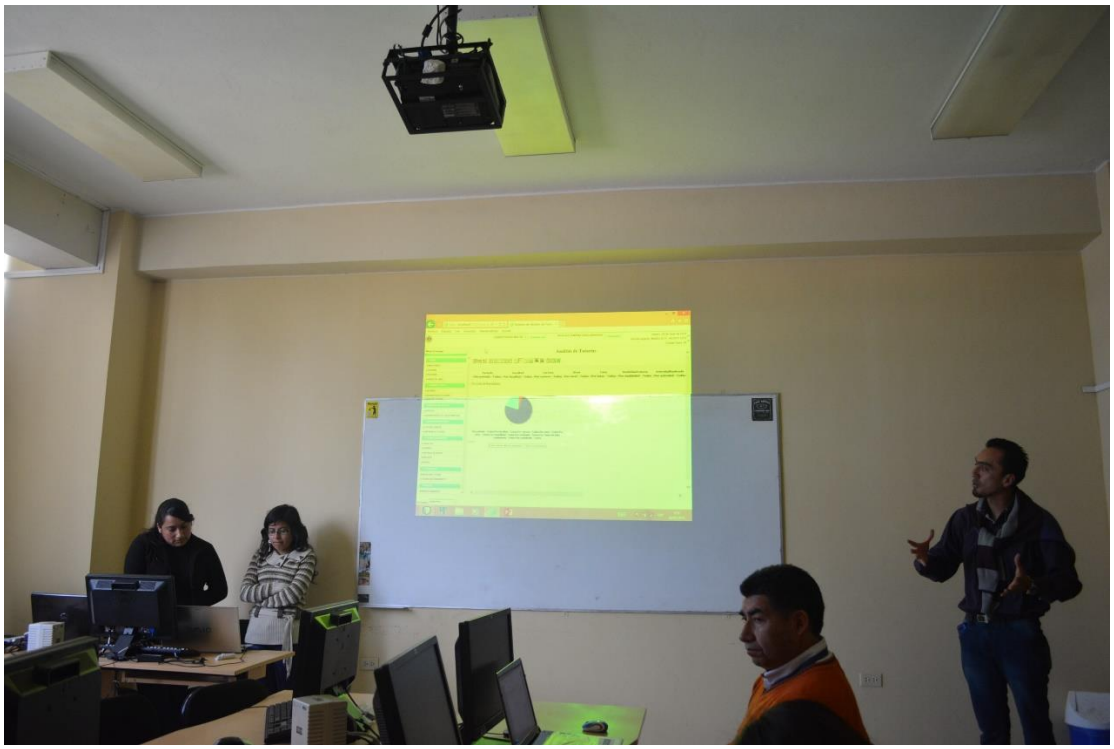
RAZÓN: Registro como tal, que el presente Reglamento fue reformado y aprobado de manera definitiva por el H. Consejo Universitario, en sesión de fecha 24 de febrero del 2015.

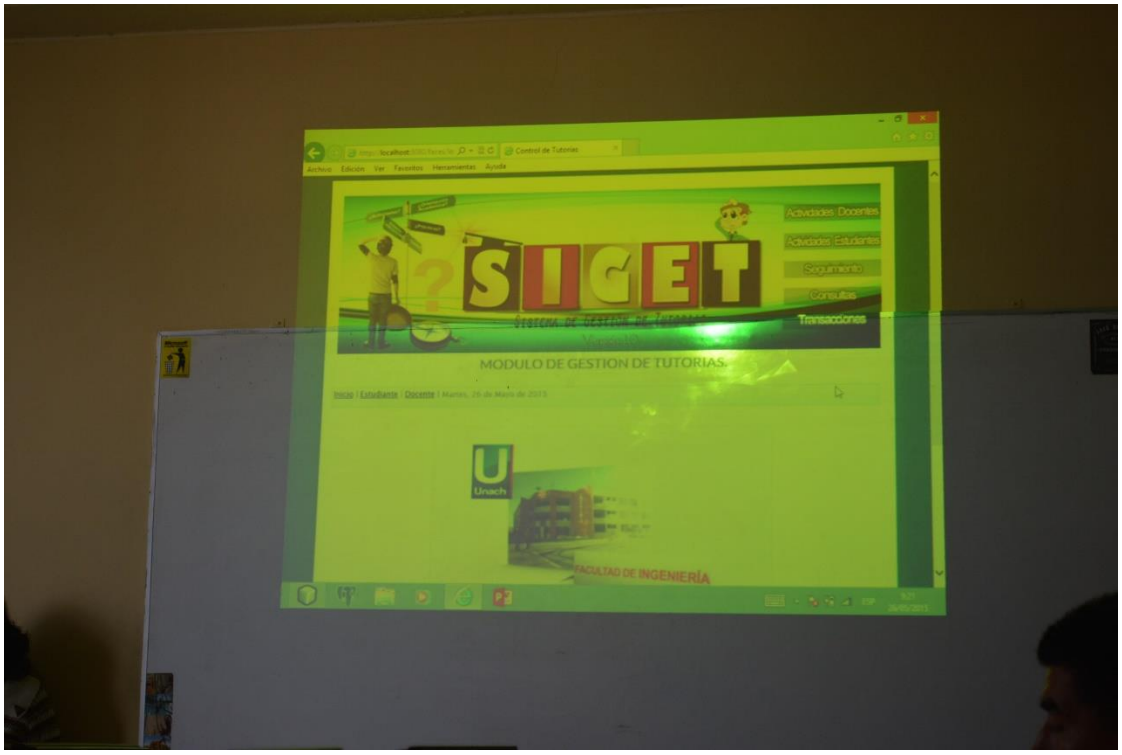
Lo Certifico:

Dr. Arturo Guerrero H.,
SECRETARIO GENERAL.
Agh.

UNIVERSIDAD NACIONAL DE CHIMBORAZO
El presente documento, es copia auténtica de su original.
Doy fe y certifico.
Fecha: 20 FEB 2015 **Hora:**
Dr. Arturo Guerrero H.
SECRETARIO GENERAL

ANEXO 11: PRESENTACION FORMAL DE SISTEMA DE GESTIÓN DE TUTORIAS SIGET – 26/05/2015




SIGET
MODULO DE GESTION DE TUTORIAS
FACULTAD DE INGENIERÍA