ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"IMPLEMENTACIÓN DE UN PORTAL WEB PARA LA AUTOMATIZACIÓN DEL PROCESO DE CONSULTORÍAS DE MENTORES GOLD DE LA REGIÓN LATINOAMERICANA DEL IEEE (R9), UTILIZANDO ARQUITECTURA JAVA 2 ENTERPRISE EDITION - J2EE Y TECNOLOGÍA AJAX"

TESIS DE GRADO

Previo a la obtención del Título de:

INGENIERO EN COMPUTACIÓN
ESPECIALIZACIÓN SISTEMAS DE INFORMACIÓN

Presentada por:

Diana Cristina Vera Santana
Rubén Alejandro Lara Vásquez

GUAYAQUIL – ECUADOR

2009

AGRADECIMIENTO

A Dios por darme Fe en cada paso.

A mis padres por su apoyo y amor durante toda mi carrera.

A mi amigo y compañero de Tesis, Rubén, por animarme siempre.

A Salo por ser el mentor de este proyecto.

A mis profesores y amigos que me acompañaron durante mis estudios.

A todas las personas que me brindaron su apoyo en la realización de esta tesis.

Diana

AGRADECIMIENTO

A Dios, por acompañarme siempre y darme fuerza y voluntad en los momentos duros de mi vida.

A mis padres por su paciencia y dedicación durante mi formación intelectual y espiritual.

A mi hijo y esposa, mis compañeros de vida.

Rubén

DEDICATORIA

A mis padres que siempre estuvieron y están a mi lado de todas las formas.

A mis hermanas Tatiana, Ana María y Andrea y a mi sobrina Lía a quienes amo mucho.

A mis queridos y siempre recordados amigos de la Rama Estudiantil IEEE-ESPOL.

A todas aquellas personas que dejaron muchas enseñanzas en mi carrera universitaria.

Diana

DEDICATORIA

A mi querida madre, hijo y esposa.

A mis buenos amigos: Salomón, Jorge, Luis y Diana.

A todas aquellas personas que de una u otra forma influyeron en mi formación.

Rubén

TRIBUNAL DE GRADUACIÓN

Ing. Jorge Aragundi SUBDECANO DE LA FACULTAD	Ing. Fabricio Echeverría DIRECTOR DE TESIS
Ing. Gustavo Bermúdez MIEMBRO DEL TRIBUNAL	Ing. Carmen Vaca MIEMBRO DEL TRIBUNAL

DECLARACIÓN EXPRESA

“La responsabilidad del contenido de esta Tesis de Grado, nos corresponde exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL”.

(Reglamento de Graduación de la ESPOL).

Diana Cristina Vera Santana

Rubén Alejandro Lara Vásquez

RESUMEN

El presente proyecto muestra la implementación de un portal Web para el programa de consultorías de mentores GOLD de la Región Latinoamericana del IEEE (Instituto de Ingenieros Eléctricos y Electrónicos) basada en el uso de la arquitectura J2EE, (Java 2 Enterprise Edition), AJAX, (Asynchronous JavaScript and XML) y la herramienta para la persistencia de datos Hibernate, en la implementación del mismo. El sistema implementado automatiza los procesos que se llevan a cabo dentro del programa de consultorías *Peer to Peer*, cuyo objetivo principal es el de lograr una interacción entre estudiantes universitarios y profesionales, ambos afiliados al IEEE. El programa *Peer to Peer* servirá como canal de comunicación e interacción entre varios usuarios y ha sido diseñado para cubrir las interrogantes que existen en los miembros estudiantiles de la Sección Ecuador y las demás Secciones de Latinoamérica, Ramas Estudiantiles y Capítulos Estudiantiles, a través de la Mentoría de Profesionales IEEE.

En el Capítulo 1, se describen los antecedentes que han originado la implementación de la administración y evaluación de consultorías, también

se plantea los objetivos que se espera alcanzar y la justificación del desarrollo de los módulos.

En el Capítulo 2, se presenta el fundamento teórico en que se basará la solución, haciendo una descripción general de la Arquitectura y patrones J2EE durante el diseño, además se describe y se justifica el uso de la tecnología AJAX.

En el Capítulo 3, se realiza un análisis de los módulos, especificando los requerimientos funcionales y no funcionales más relevantes. Además, se listan los casos de uso, escenarios y los roles de los usuarios implicados en el sistema.

En el Capítulo 4, se explican los modelos diseñados para los módulos partiendo de la Arquitectura general del sistema y se hace una descripción de los patrones de diseño utilizados, para dar paso al modelo lógico de la base de datos. Además, se da una breve explicación del modelo de persistencia de datos usado mediante la herramienta Hibernate.

En el Capítulo 5, se hace un análisis de la implementación realizada y la comunicación entre capas, basado en el diseño J2EE, luego se plantea un plan de pruebas del sistema, un diseño para la recepción de métricas que permitan medir el avance del proceso de desarrollo de los módulos y finalmente se muestran los resultados de las pruebas y las métricas obtenidas.

Finalmente, se presentan las respectivas conclusiones y recomendaciones, adjuntando los anexos y haciendo referencia a la bibliografía utilizada.

INDICE GENERAL

Pág.

ABREVIATURAS

AJAX	Asynchronous JavaScript And XML
MVC	Model View-Controller
J2EE	Java 2 Enterprise Edition
GOLD	Graduates of the Last Decade
ER	Entity Relation
P2P	Peer to Peer
GB	Giga Bytes
GHz	Giga Hertz
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
JDBC	Java DataBase Connectivity
JSP	Java Server Pages
MB	Mega Bytes
Mbits	Mega bits
MHz	Mega Hertz
PIII	Pentium III
RAM	Random Access Memory
API	Application Programming Interface
JDBC	Java Database Connectivity
RMI	Remote Method Invocation
EJB	Enterprise JavaBeans
http	Hypertext Transfer Protocol
DOM	Document Object Model
HTML	HyperText Markup Language
XML	Extensible Markup Language

INDICE DE TABLAS

INDICE DE FIGURAS

CAPITULO 1

1. ANTECEDENTES Y JUSTIFICACIÓN

1.1. Definición del problema

El IEEE, (por sus siglas en inglés Institute of Electrical and Electronics Engineers) es el Instituto de Ingenieros Eléctricos y Electrónicos más grande del mundo, con sede en los Estados Unidos. Está formado por más de 365,000 miembros, incluyendo 68,000 estudiantes, en 150 países. Posee 311 secciones en 10 Regiones Geográficas alrededor del mundo. Tiene creados 1,450 Capítulos Técnicos, más de 1,300 Ramas Estudiantiles en 80 países. 39 Sociedades Técnicas y 128 Transactions, Journals and Magazines. Además 300 Conferencias en todo el mundo cada año y 900 estándares IEEE activos y más de 400 en desarrollo. [1]

El IEEE brinda la oportunidad a Universidades e Institutos Académicos de nivel superior a ser parte de toda la organización a través de las Ramas Estudiantiles. Una Rama Estudiantil es un grupo de estudiantes miembros y voluntarios del Instituto, que representan a determinada universidad. Dicho grupo tiene la oportunidad, a través de los beneficios que brinda el IEEE, de fomentar la investigación y el liderazgo en los estudiantes de la Universidad.

Una Sección IEEE es una división del IEEE que representa a un país. El rol de la Sección es la coordinación de la membresía estudiantil y profesional IEEE en el país sede. Dentro de la Sección existe un Grupo de Afinidad de Graduados de la última década GOLD, el cual brinda soporte a los miembros jóvenes profesionales del IEEE. El GOLD es coordinado por un profesional miembro del IEEE y mantiene comunicación con el Coordinador GOLD de toda las Región Latinoamericana.

Como objetivo principal de este Grupo de Afinidad GOLD está el de lograr una interacción entre estudiantes universitarios y profesionales, para lo cual se ha planteado el programa *Peer to Peer*, que servirá como canal de comunicación e interacción entre varios usuarios. El programa ha sido diseñado para cubrir las interrogantes que existen en los miembros estudiantiles de la Sección Ecuador y las demás Secciones de Latinoamérica, Ramas Estudiantiles y Capítulos Estudiantiles, a través de la Mentoría de Profesionales IEEE.

Para mejorar la efectividad del programa, se propone la implementación de un portal web que automatice todos los procesos que se llevan a cabo dentro del programa. Esto sin lugar a dudas, constituye una herramienta de ayuda y soporte para la gestión del programa *Peer to Peer*, el cual ya contaría con un apoyo tecnológico y almacenamiento de comunicaciones y actividades ya realizadas, para que sirvan de partida para futuras actividades.

El programa *Peer to Peer* es un proyecto que se llevará a cabo a partir del desarrollo del sistema planteado en esta tesis, por lo cual no se tiene un antecedente de cómo se llevaba el manejo del programa, pero lo más parecido que se tenía es lo siguiente:

Actualmente, durante la organización de eventos en las Ramas estudiantiles del IEEE, los miembros estudiantiles buscan como principal apoyo a ingenieros de su universidad, quienes por lo general son profesionales que tengan más experiencia técnica u organizacional. Algunas veces, esos profesionales no son miembros GOLD del IEEE.

La manera en que los miembros estudiantiles piden apoyo es verbalmente o mediante uno o varios correos electrónicos, donde se intercambian preguntas y respuestas referentes al evento a realizar. Si el profesional está dispuesto a colaborar, pero no tiene conocimiento de ciertos recursos o programas del IEEE que el miembro estudiantil requiere, éste debe acudir a otras personas y así se va haciendo una cadena más grande, sin que el voluntario estudiantil tenga siempre un profesional que lo esté guiando durante todo el proceso de la actividad que desea realizar.

Finalmente, la información que alguna vez intercambiaron, se queda en el correo del miembro estudiantil y del profesional que lo está apoyando, o en sus memorias en caso de que haya sido sólo una conversación.

Este proceso se lleva de la misma manera casi en todas las Ramas Estudiantiles de la Región de Latinoamérica, pues el tipo de eventos que realizan los miembros estudiantiles, casi siempre van por la misma línea, pero de una manera muy informal y sin que se tenga una base integrada donde se almacene toda esta información para que pueda ser usada a futuro, no sólo por la Rama Estudiantil que realiza la actividad, sino por otras Ramas de la Sección y de la Región.

Adicionalmente, no se puede llevar un control del desarrollo del evento, para hacer un seguimiento de las responsabilidades del miembro estudiantil y del profesional que lo está apoyando.

Actualmente tampoco se evalúa el rendimiento de la asesoría voluntaria que le brinda un profesional a los voluntarios de una Rama Estudiantil, y tampoco se puede controlar ni garantizar que ese profesional al que se acude, sea una persona con conocimientos de programas o recursos del IEEE, de los cuales una Rama Estudiantil puede sacar mucho provecho.

Por lo expuesto ha surgido la necesidad de automatizar este proceso para mejorar el manejo y la dinámica de administrar las consultorías que realizan miembros estudiantiles a profesionales, que en el mejor de los casos deben ser miembros GOLD del IEEE.

Por otro lado, uno de los objetivos principales del Comité GOLD y el Comité de Actividades Estudiantiles de la Región Latinoamericana del IEEE (RSAC), es el aumento de la membresía, tanto de estudiantes como de profesionales y una mayor interacción entre estos dos grupos, incrementando el voluntariado en el IEEE, es por eso que el desarrollo de esta tesis, a mas de automatizar el proceso de consultoría a mentores, logrará apoyar al cumplimiento de esos objetivos, permitiendo llevar un mayor control sobre esta interacción entre miembros estudiantiles y miembros profesionales del IEEE, logrando una mayor integración entre Ramas Estudiantiles, pues todo quedará almacenado en una Base de Conocimiento, la cual puede ser consultada por otros miembros estudiantiles en el momento que deseen realizar un evento similar al realizado por otra Rama de la Sección o de la Región.

1.2. Objetivos

Luego de haber analizado las necesidades actuales y definiendo la solución adecuada, el desarrollo de esta tesis tiene por objetivos:

- Automatizar el proceso de consultoría de mentores GOLD IEEE a estudiantes de las Ramas Estudiantiles de Latinoamérica.
- Diseñar un sistema de tres capas utilizando los patrones de diseño de la Arquitectura J2EE.
- Implementar el proceso en las Ramas Estudiantiles IEEE de América Latina que se suscriban al proyecto.

1.3. Justificación

El comité GOLD de la Región Latinoamericana necesita automatizar procesos que se llevan a cabo dentro del programa *Peer to Peer,* como consultas a mentores, respuestas a los estudiantes, encuestas acerca de las mentorías, reportes de asesorías, entre otros.

Existen muchas razones por las cuales se ha vuelto imprescindible que el GOLD y el RSAC cuente con una herramienta para la administración de esas mentorías pues en la actualidad, se pueden observar los siguientes inconvenientes:

- Los procesos manuales generan equivocaciones en la información suministrada por los profesionales que apoyan las actividades de las Ramas Estudiantiles.
- Poco conocimiento de los programas que pueden desarrollar miembros estudiantiles del IEEE, mediante sus Ramas Estudiantiles.

- Existe una falta de control sobre el desarrollo de eventos.
- Administración poco eficiente de los recursos que brinda el IEEE.

Al implementar este sistema se logrará tener una herramienta que ayude a mejorar el manejo de la comunicación Mentor-Estudiante, que servirá de soporte para futuras actividades estudiantiles.

Para lograr estos objetivos e implementar el proyecto, se usará un análisis orientado a objetos, distribuyendo los objetos en 3 capas: Capa de presentación, capa del negocio y capa lógica.

Este esquema permite tener modularidad, mismo que ayudará a disminuir la propagación de cambios, para luego hacer una óptima y más real estimación de costos por modificaciones.

Además la metodología al trabajar en capas permitirá dividir el sistema en subsistemas y al definir el modelo del dominio del negocio separa claramente los detalles tecnológicos del análisis de los módulos.

Conociendo que la metodología orientada a objetos trabaja utilizando tres capas y tomando de referencia los requerimientos recibidos, se ha procedido a utilizar la arquitectura J2EE para el diseño de los módulos, a través del uso de varios de sus patrones.

1.4. Alcance

- El sistema abarca, para empezar, a los voluntarios IEEE de la Región Latinoamericana, enfocándose en Categorías y Subcategorías de actividades que comúnmente se realizan en la Región.
- El sistema cuenta con la flexibilidad para aplicarlo a todas las Regiones del IEEE a nivel mundial, por lo que en un futuro, ése podría ser un nuevo alcance.

CAPITULO 2

2. FUNDAMENTOS TEORICOS

2.1. Conceptos de arquitectura y patrones relacionados a J2EE

2.1.1. Descripción de la Arquitectura J2EE

Java 2 Enterprise Edition o Java EE, es una plataforma de programación, creada por la Sun Microsystems en 1997 que permite generar contenido Web de manera dinámica y basada en herramientas de código abierto. J2EE es parte de la Plataforma Java, para desarrollar y ejecutar software de aplicaciones en Lenguaje de programación Java con arquitectura de múltiples capas distribuidas, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. [2]

J2EE es un grupo de especificaciones que permiten la creación de aplicaciones empresariales, esto sería: acceso a base de datos

(JDBC), utilización de directorios distribuidos (JNDI), acceso a métodos remotos (RMI/CORBA), funciones de correo electrónico (JavaMail), aplicaciones Web(JSP y Servlets), entre otras y define cómo coordinarlos; también configura algunas especificaciones únicas para componentes de Java Enterprise Edition, estas incluyen Enterprise JavaBeans, servlets, portlets, Java Server Pages y varias tecnologías de servicios web, soportados a través de intercambio de datos integrados basados en estándares abiertos y protocolos XML (Extensible Markup Language). Aquí es importante notar que J2EE es sólo una especificación, esto permite que diversos productos sean diseñados alrededor de estas especificaciones.

La programación en múltiples capas es la técnica más efectiva en aplicaciones empresariales, debido a la fácil administración que implica el dividir los componentes de la aplicación en capas y la rapidez que esto implica en programas Cliente-Servidor.

En un modelo multi-capa, la primera capa es el cliente que normalmente es un navegador Web o una aplicación Java solitaria. Este invoca a la lógica del negocio de una o más capas medias que se están ejecutando sobre hardware dedicado, que a su vez acceden a los datos desde el Enterprise Information Service en la tercera capa.

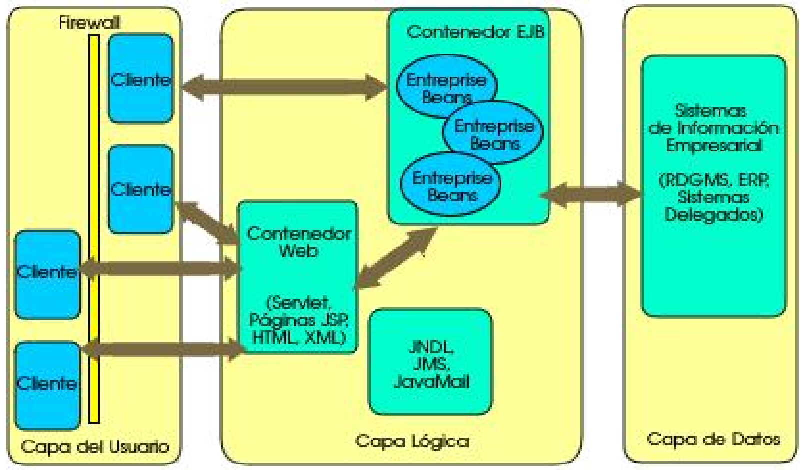

Figura 2.1 Representación de Aplicaciones Multicapas

La arquitectura J2EE implica un modelo de aplicaciones distribuidas en diversas capas o niveles:

- La capa cliente admite diversas tipos de clientes (HTML, Applet, aplicaciones Java, etc.). Ésta capa se ejecuta sobre la máquina del cliente.
- La capa intermedia contiene subcapas (el contenedor web y el contenedor EJB) y se ejecutan sobre un servidor J2EE.
- La tercera capa dentro de esta visión sintética es la de de aplicaciones 'back-end' como ERP, EIS, bases de datos, etc.

Como se puede ver un concepto clave de la arquitectura es el de contenedor, que dicho de forma genérica no es más que un entorno de ejecución estandarizado que ofrece unos servicios por medio de componentes. Los componentes externos al contenedor tienen una forma estándar de acceder a los servicios de dicho contenedor, con independencia del fabricante.

La plataforma J2EE usa un modelo de aplicación multicapas distribuido para aplicaciones empresariales. La aplicación empresarial o lógica de la empresa está divida entre componentes dependiendo su función y al integrar una aplicación J2EE, dichos componentes son instalados en diferentes máquinas dependiendo del nivel del ambiente multicapa J2EE al cual pertenecen.

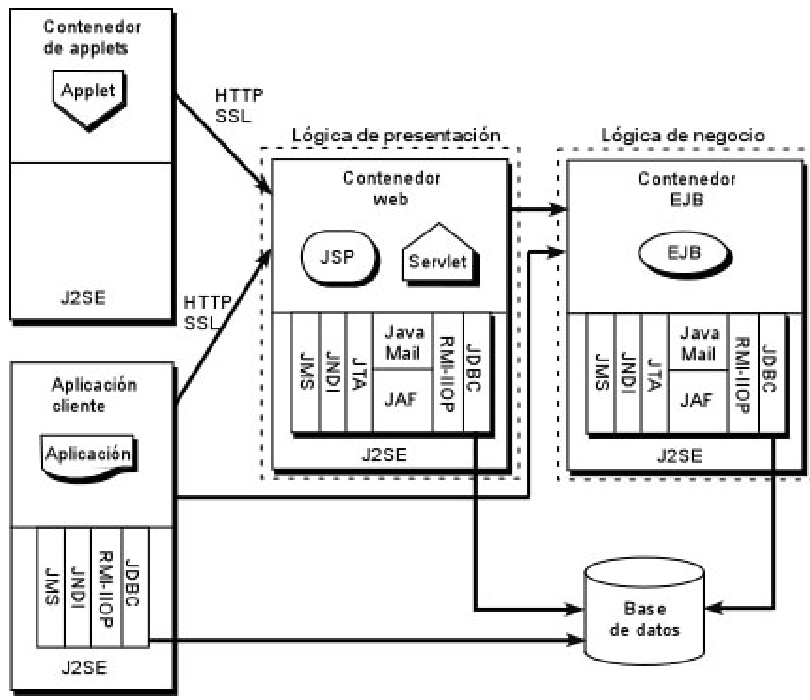

Figura 2.2 Componentes de arquitectura J2EE

Las partes de la aplicación J2EE mostradas en la Figura 2.2 son presentadas en Componentes que se mencionan a continuación:

- ***Cliente web*** (contenedor de applets): Es usualmente un navegador e interactúa con el contenedor web haciendo uso de HTTP. Recibe páginas HTML o XML y puede ejecutar applets y código JavaScript.

- ***Aplicación cliente:*** Son clientes que no se ejecutan dentro de un navegador y pueden utilizar cualquier tecnología para comunicarse con el contenedor web o directamente con la base de datos.
- ***Contenedor web:*** Es lo que comúnmente denominamos servidor web. Es la parte visible del servidor de aplicaciones. Utiliza los protocolos HTTP y SSL (seguro) para comunicarse.
- ***Servidor de aplicaciones:*** Proporciona servicios que soportan la ejecución y disponibilidad de las aplicaciones desplegadas. Es el corazón de un gran sistema distribuido.

Las aplicaciones J2EE son generalmente consideradas aplicaciones de tres capas ya que se encuentran distribuidas principalmente en tres lugares: en las máquinas clientes, la máquina del servidor J2EE, y la base de datos.

En las aplicaciones de tres capas, por lo general, el servidor de aplicaciones es considerado de dos capas, una que enlace el servidor con el cliente y otro con el almacenamiento final.

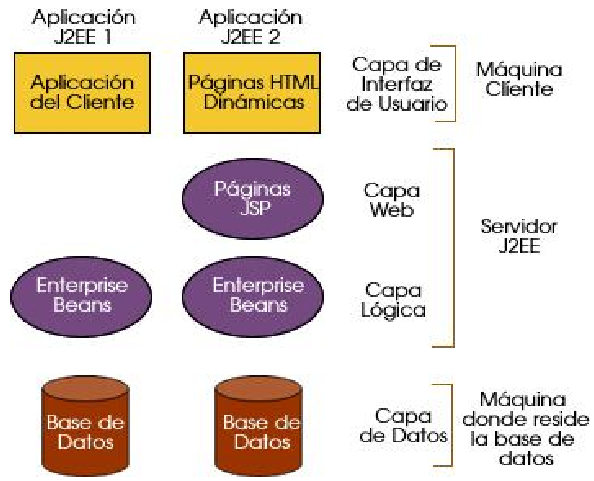

Figura 2.3 Vista de dos aplicaciones J2EE en capas

Componentes J2EE

Las aplicaciones J2EE son integradas por algunos componentes, escritos en código JAVA y son compilados de la misma forma como cualquier programa en este lenguaje. Un componente es una unidad funcional del software contenido por sí mismo que es ensamblado dentro de una aplicación J2EE con sus clases y fuentes que se comunican con otros componentes.

La especificación J2EE define los siguientes componentes:

- Aplicaciones Clientes y applets son componentes que corren en el cliente.
- Componentes Tecnológicos como Java Servlet y JavaServer Pages (JSP) son componentes web que corren sobre el servidor.

- Componentes Enterprise JavaBeans (EJB) son componentes comerciales que corren sobre el servidor.

El modelo de aplicación J2EE encapsula las capas de funcionalidad en componentes de tipos específicos. La lógica de negocio está encapsulada en componentes Enterprise JavaBeans (EJB). La interacción con los clientes pueden representarse a través de:

- Páginas web de HTML plano.
- Páginas web con Applets Java.
- El API Java Servlets.
- Tecnología JavaServer Pages.
- Aplicaciones Java solitarias.

Los componentes se comunican de forma transparente usando varios estándares, incluyendo HTML, XML, HTTP, SSL, RMI, IIOP, y otros:

J2EE está compuesto de diferentes componentes:

- ***Servlet***: Un reemplazo eficiente, independiente de la plataforma para los scripts CGI que responden a solicitudes de clientes.
- ***JavaServer Page*** (JSP): un tipo de script del lado del servidor, que puede generar páginas web dinámicamente.
- ***Enterprise JavaBeans*** (EJB): control de sesión del lado del servidor, que encapsula la lógica de

negocios y abstracción para acceder a datos persistentes.

- ***Java Database Connectivity*** (JDBC): un API que describe una librería estándar Java para acceder a fuentes de datos.
- ***Transaction Support:*** transacciones declarativas para componentes donde las transacciones pueden expandir componentes y procesos.
- ***Java Naming and Directory Interface*** (JNDI): un interface abstracto para servicios de búsqueda de uniones de nombres y directorios.
- ***Remote Method Invocation*** (RM/IIOP): Una tecnología que permite la comunicación entre objetos distribuidos. [3]

2.1.2. Justificación del uso de la Arquitectura J2EE

La arquitectura del software es algo que debe considerarse muy bien al diseñar una aplicación, el estándar J2EE permite el desarrollo de aplicaciones de empresa de una manera sencilla y eficiente. Una aplicación desarrollada con estas tecnologías, permite ser desplegada en cualquier servidor de aplicaciones o servidor web que cumpla con el estándar.

El IEEE ha planteado como requerimiento montar la aplicación sobre su web-hosting que puede migrar a nuevas versiones de plataformas, por lo que la aplicación debe de utilizar una arquitectura Open Source, que permita sobre todo la portabilidad del sitio web y su implementación en cualquier sistema operativo y mantener activo el funcionamiento de la aplicación sin mayores inconvenientes.

2.2. Conceptos relacionados a AJAX

2.2.1. Descripción de la tecnología AJAX

AJAX es un acrónimo de Asynchronous JavaScript + XML, que se puede traducir como "JavaScript asíncrono + XML". Es una técnica de desarrollo web para crear aplicaciones interactivas. Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios, y mantiene comunicación asíncrona con el servidor en segundo plano.

Las aplicaciones construidas con AJAX eliminan la recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor. La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la respuesta del servidor.

AJAX no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen. Las tecnologías que forman AJAX son:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.

- XML, XSLT y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.[4]

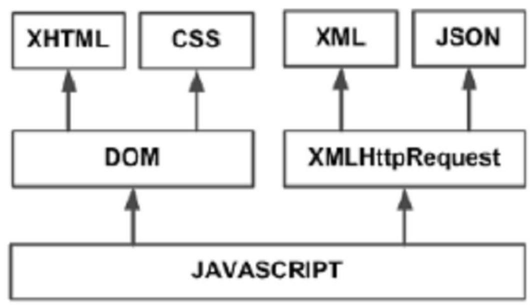

Figura 2.4 Tecnologías que agrupa AJAX

Sin embargo, se puede lograr AJAX, sin sin XML, reemplazándolo por JSON por ejemplo.

AJAX es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores, dado que está basado en estándares abiertos.

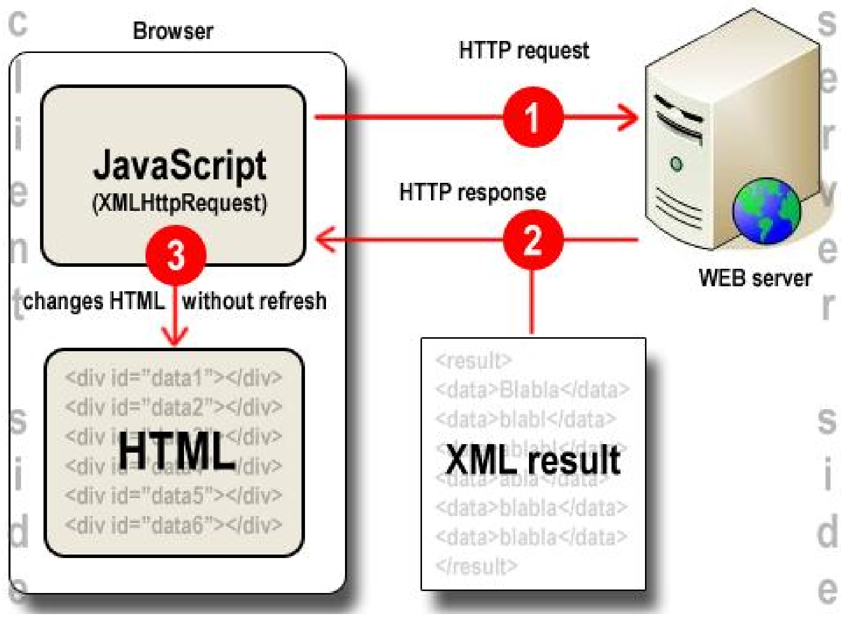

Figura 2.5 Aplicación web con AJAX

1. El cliente por medio del browser produce algún evento. Este evento (como hacer click en un link por ejemplo) este es procesado por JavaScript (o alguna otra tecnología client-side) y le envía al servidor web una HTTP request.
2. El servidor web, procesa la petición como siempre y le devuelve una respuesta con el resultado en XML.
3. Este resultado es procesado por JavaScript, que recarga las secciones de la página necesarias para mostrar el resultado al usuario
4. Por medio de esta misma página, el ciclo comienza de nuevo sin haber tenido que recargar la página.

AJAX tiene ventajas y desventajas como toda tecnología, sólo debemos aprender a sacarle el mejor provecho para minimizar las desventajas y aprovechar las ventajas que nos ofrece:

Ventajas

- Las aplicaciones son más interactivas, responden a las interacciones del usuario más rápidamente, al estilo desktop.
- Las páginas no se recargan constantemente.
- El tiempo de espera puede ser menor.
- Se pueden lograr aplicaciones web que sin AJAX definitivamente no se podrían hacer, como el conocido Google Maps por ejemplo.

Desventajas

- Falta de integración con el botón "retroceder" de los browsers. Esto puede llegar a confundir al usuario.
- Es necesario que el navegador soporte y tenga habilitado JavaScript. Pero no es una "desventaja", ya que casi todos los navegadores modernos soportan JavaScript.
- Al tener que ejecutar más código del lado del cliente, puede degradar el rendimiento de la máquina del cliente. Por eso debe usarse AJAX con moderación.
- Los usuarios no pueden obtener una URL a la cual poder referirse, en caso de querer recomendar la página, o volver a esa página en algún momento. Por eso debe saberse cuando usar AJAX y cuando no. [5]

- Más trabajo para los desarrolladores. Programar aplicaciones con AJAX puede llegar a ser tedioso a veces. Pero hay muchas herramientas y frameworks para facilitarnos el trabajo. [2]

2.2.2. Justificación del uso de la tecnología AJAX

Debido a la concurrencia de usuarios desde varios países de Latinoamérica que tendremos, y dado que la aplicación estará en un servidor en Estados Unidos, necesitamos que el sistema sea liviano al acceder a sus opciones y consultas y que tenga una buena interacción con el usuario.

Características como: flexibilidad, velocidad, y usabilidad, hacen que AJAX sea atractiva para aplicaciones que necesitan minimizar el tiempo de respuesta a solicitudes de usuarios al interactuar con las pantallas

Otra ventaja que deseamos aprovechar de esta tecnología es que sea multiplataforma, así nos despreocupamos del entorno en el que se esté ejecutando y centramos el desarrollo en realizarlo bajo los estándares abiertos.

2.3. Concepto de estándares web utilizados

2.3.1. Descripción de estándar CSS

CSS es un mecanismo para añadir estilo (p.e. fuentes, colores, espaciados) a documentos Web. Está basado en la determinación de reglas que se aplican al contenido de los documentos Web para darles cierto formato. [6]

Una característica fundamental de CSS es que más de una hoja de estilo puede influir en la presentación de un documento. Esta característica se conoce como cascada, dado que las hojas de estilos diferentes son consideradas como procedentes de una serie. La cascada es una característica fundamental de los CSS, cualquier documento único, podría muy probablemente terminara con hojas de estilo de múltiples fuentes: el navegador, el diseñador, y posiblemente el usuario.

Algunos beneficios concretos de usar CSS son:

- Diseño de control de algunos documentos desde una solo hoja de estilo;
- Un control más preciso del diseño;
- Aplicar diferente diseño para diferente tipos de medios;
- Numerosas técnicas avanzadas y sofisticadas.

La W3C es la organización que publica este y otros estándares web y en cuyo sitio web se pueden hacer validaciones en tiempo real sobre la aplicación de determinado estándar web sobre nuestro sitio [7].

2.3.2. Justificación del uso de CSS

Como hemos indicado, son innumerables los beneficios que conlleva la utilización de hojas de estilo. En nuestro caso, la solución a la problemática de mentoría GOLD a estudiantes voluntarios del IEEE, está basada en el desarrollo de una aplicación web, la cual será utilizada por miembros estudiantiles y profesionales de diversos países de Latinoamérica. Esta aplicación a más de automatizar el proceso de consultoría deberá brindar muchas facilidades a los usuarios.

Una de sus características principales será el estilo que contengan las páginas para brindar una interacción lo suficientemente agradable con el usuario. Al ser un aplicativo basado en la lógica MVC de la Arquitectura J2EE, serán muchas la páginas con las que el usuario debe interactuar, por eso la necesidad de tener un control sobre el diseño que permita añadir nuevas pantallas sin necesidad de crear nuevamente estilos repetitivos, es por esto que se usarán cascadas para utilizar un diseño base sobre varios documentos web.

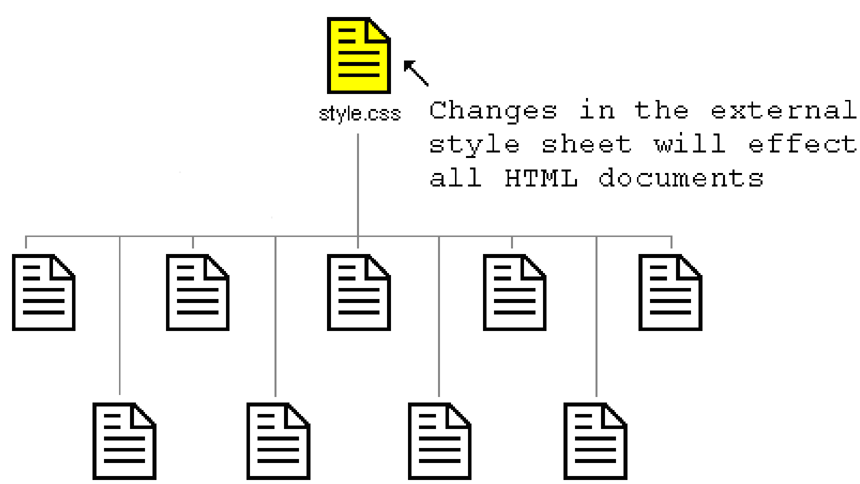

Figura 2.10 Hoja de estilo que funciona en cascada con varios documento web

Siguiendo un mismo patrón, será mucho más fácil representar el diseño de tablas, formularios, encabezados, enlaces, colores y tipos de letra.

2.3.3. Descripción de estándar XHTML

XHTML (eXtensible Hypertext Markup Language) es una versión más estricta y limpia de HTML pensado para sustituir a HTML como lenguaje estándar para las páginas Web

XHTML es en sí mismo XML por lo que sigue las reglas del mismo que son: bien formado y válido. Que esté bien formado significa que debe contener un solo elemento raíz y que todas las etiquetas que se abren deben tener una etiqueta de cierre. Que sea válido en cambio significa que debe seguir las reglas especificadas en un DTD (Document Type Definition) o un schema que son los que definen las reglas de cómo debe estar estructurado el documento XML a la hora de ser validado.

La W3C valida las páginas web para comprobar que cumplen con los estándares establecidos, nuestro sistema pasó la prueba, por lo que podemos indicar que está convalidada por el W3C.

CAPITULO 3

3. ANÁLISIS DEL PROBLEMA

Se presentan a continuación definiciones básicas de conceptos que se utilizaran en el desarrollo de la aplicación:

- **Miembro Estudiantil:** Estudiante enrolado en una Rama Estudiantil del IEEE con grado Student Member o Graduate Student Member que realizan actividades dentro de su rama Estudiantil y que requieren la asesoría de un miembro IEEE con experiencia para llevar a cabo exitosamente dicha actividad.
- **Mentor:** Miembro Profesional IEEE que cuenta con la experiencia necesaria en su voluntariado como para poder dar asesoría referente a temas concretos y relativos al IEEE al miembro Estudiantil.

- **Sección:** Unidad Organizacional del IEEE que agrupa a los miembros IEEE de un país, por ejemplo: Sección Ecuador, Sección Mexico, etc.
- **Región:** Unidad Organizacional del IEEE que agrupa a los miembros y agrupaciones IEEE de una región geográfica. El IEEE se divide en 10 regiones alrededor del mundo.
- **GOLD**: Graduate of the Last Decade, son los miembros del IEEE, que han recibido su primer título profesional en los últimos diez años. No es un grado de membresía sino un grupo de afinidad, estos estudiantes recién graduados son automáticamente parte de IEEE GOLD durante 10 años.
- **Rama Estudiantil**: Grupo oficial en el mundo IEEE, formado por alumnos de diferentes programas académicos y de posgrado de la Universidad, cuyos objetivos principales son el desarrollo profesional, investigación, desarrollo y aplicación de nuevas tecnologías.
- **Categoría:** Clasificación que se da a las actividades que puede realizar un miembro estudiantil.
- **Subcategoría:** Clasificación más específica que se da a las actividades que puede realizar un miembro estudiantil.
- **Actividad**: Programa o suceso organizado por un miembro estudiantil dentro de su rama Estudiantil para el desarrollo de la misma.
- **Consultoría**: Acción de preguntar-responder entre un estudiante y un mentor.

3.1. Requerimientos funcionales

De manera general los servicios que deben proporcionar los módulos son:

- Administración de Regiones: los administradores de región podrán ingresar y actualizar la información básica de las regiones en las cuales se divide el IEEE.
- Administración de secciones: el portal debe contener la información relativa a las secciones que pertenecen a las diferentes regiones del IEEE. Los administradores en este módulo, podrán ingresar y actualizar los datos de la directiva y de los usuarios que administrarán cada sección, además de la información general de la sección, como sitios webs.
- Administración de usuarios: Nuevos usuarios de tipo Revisores de Región o de Sección pueden registrarse en el sistema, el administrador de Región podrá administrar estos usuarios y asignar el tipo de rol que deben tener. El portal permitirá al usuario Administrador de Sección administrar los Mentores de cada Sección que darán las consultorías. Un miembro estudiantil del IEEE podrá registrarse en el portal para poder acceder a la ayuda de los Mentores de cada Sección. Administración de categorías y subcategorías: permitirá al Administrador de Región o Administrador de Sección crear y modificar las Categorías y las Subcategorías de los procesos e información necesaria que las Ramas Estudiantiles y Capítulos necesitan para realizar diferentes actividades dentro del IEEE.

- Administración de actividades: el portal permitirá administrar las actividades que los miembros estudiantiles van a ejecutar y de las cuales van a recibir la consultoría, para que el mentor GOLD y el administrador de Sección puedan medir los avances que lleva el estudiante en su actividad.
- Administración de consultoría: El portal deberá permitir intercambiar experiencias entre los mentores GOLD y los miembros estudiantiles por sección. Para cada consultoría previamente el miembro deberá seleccionar la Categoría y Subcategoría y el sistema automáticamente recomendará al mentor más apto para el tema. El sistema enviará correos en forma automática al momento de realizar una consultoría, para conocimiento de los implicados.
- Evaluación de la consultoría: Los administradores de cada Sección podrán realizar una evaluación tanto al estudiante como al mentor respecto a la ejecución de la actividad y a la consultoría. Además permitirá a los administradores de Región realizar una evaluación general de todas sus secciones, realizando encuestas para cualquier tipo de usuario del portal.

A continuación se detallan los requerimientos para todos los módulos:

ADMINISTRACIÓN DE ENTIDADES

- Definimos como Entidad a las Regiones Geográficas del IEEE, Secciones del IEEE, Universidades y Cargos del IEEE.

- El módulo debe permitir crear una región. Para la creación de la Región se deben almacenar los siguientes datos: nombre de la Región, Director de la Región, Coordinador del Comité SAC de la Región, Coordinador del Comité GOLD de la Región, una o varias páginas web de la Región, incluyendo una descripción y el estado de la Región: *activa o inactiva.*
- El administrador de cada Región debe poder actualizar los datos de su Región correspondiente.
- El módulo debe permitir crear una Sección. Para la creación de la Sección se deben almacenar los siguientes datos: nombre de la Sección, referencia a la Región que pertenece, directiva de la Sección, lo cual incluye el Presidente, Coordinador del comité GOLD, Coordinador del comité SSAC, una o varias páginas web de la Sección, incluyendo una descripción y el estado de la Sección: *activa o inactiva.*
- El administrador de cada Sección debe poder actualizar los datos de su Sección correspondiente.
- El administrador de Región debe poder dar de baja a una sección, cambiando su estado de *activo a inactivo.*
- El módulo debe permitir la creación de Universidades que cuentan con Ramas Estudiantiles del IEEE. Para la creación de la Universidad se deben almacenar los siguientes datos: nombre de la Universidad, referencia a la sección que pertenece, una o más páginas web de la Universidad, URL de la(s) página(s) web,

descripción de la(s) página(s), tipo de la web (universidad), estado de la Universidad: *Activa o inactiva.*

- El Administrador de Región debe poder modificar los datos de una universidad de su sección.
- El módulo debe permitir la creación de nuevos cargos de IEEE.
- El Administrador de Región debe poder crear nuevos cargos.
- Para la creación de un cargo nuevo, se deben almacenar los siguientes datos: Nombre del cargo, descripción, estado: Activo o inactivo y tipo de cargo, el cual puede ser: de *Rama Estudiantil, de Sección o de Región y otros*.

ADMINISTRACION DE USUARIOS

- El módulo deberá permitir crear un nuevo usuario de tipo mentor. Para la creación del mentor se deberán almacenar los siguientes datos: la *sección* y la *región* a la que pertenece, *nombre* y *apellidos* del mentor, *numero de membresía*, *correo electrónico*, *teléfono*, *dirección* del domicilio, *cargo* que tiene en el IEEE, una breve *descripción de su voluntariado* en el IEEE, *categorías* y *subcategorías* en las que se siente capacitado para dar consultorías, la *foto* y el *estado* del mentor: activo, inactivo, pendiente o en transición.
    i. Pendiente: Cuando el mentor ingresa sus datos y está en espera de que el Administrador de Sección lo apruebe.
    ii. Activo: Cuando el mentor ya ha sido aprobado por el Administrador de Sección.

iii. Inactivo: Cuando se ha dado de baja al mentor.

iv. Transición: Cuando un miembro estudiantil solicita un cambio de rol de estudiante a mentor.

- El correo electrónico será considerado como el usuario para ingresar al sistema para todos los tipos de usuario y debe tener el alias IEEE, ejemplo *alias@ieee.org*.
- El cargo que se mostrará al Mentor, deben ser todos los de tipo Regional y de Sección.
- Al ingresar sus datos personales, el mentor debe escoger las categorías y las subcategorías en las cuales está preparado para dar mentorías. Estas categorías y subcategorías deben estar previamente creadas.
- Las categorías y subcategorías que se mostrarán al mentor son todas las creadas en el sistema, aunque no estén activas para su sección.
- Cuando el usuario que aspira ser mentor se registre, el sistema le debe enviar un correo indicándole que sus datos han sido ingresados correctamente y que la contraseña le llegará una vez que sea aprobada su petición de registro, por el Administrador de Sección.
- Cuando el usuario que aspira ser mentor se registre, el sistema debe enviar un correo electrónico al Administrador de Sección avisándole que hay una nueva solicitud de aprobación de mentor pendiente.

- Los datos ingresados por el usuario que aspira ser mentor deben pasar por la aprobación del Administrador de Sección, quien decidirá si se lo aprueba como mentor o no, en base a la experiencia en el voluntariado IEEE que éste haya ingresado.
- En el momento que el Administrador de Sección decide si aprueba o no a un mentor, en caso de que éste haya seleccionado subcategorías que no estén activas para la sección, el Administrador de Sección debe poder activar esas subcategorías para su sección o dejarlas pasar por alto.
- En caso de que el Administrador de sección no apruebe las subcategorías seleccionadas por el mentor y que están inactivas para la sección, el mentor no podrá dar consultorías relacionadas a esas categorías, hasta que el Administrador de sección la apruebe, donde automáticamente el mentor estará activo para dar las correspondientes consultorías.
- Cuando un mentor no ha sido aprobado por el Administrador de Sección, el sistema debe enviarle un correo electrónico indicándole esto.
- Cuando un mentor ha sido aprobado por el Administrador de Sección, el sistema debe enviarle un correo electrónico indicándole su contraseña creada aleatoriamente y el link donde debe ingresar.
- El Administrador de Sección puede dar de baja a un mentor, pero si éste tiene una consultoría activa, primero le debe asignar a otro mentor estas consultorías. Si no hay otros mentores con las

mismas características (que tengan las mismas Subcategorías) entonces el sistema no debe permitir que se le dé de baja a ese mentor hasta que termine sus consultorías.

- Los mentores deben poder modificar los siguientes datos personales: *nombres, apellidos, teléfono, foto, correo electrónico, contraseña, dirección del domicilio y su voluntariado en el IEEE.*
- En caso de que se creen nuevas categorías o subcategorías, los mentores deben poder añadirlas a su perfil, en caso de que coincidan con su experiencia.
- El módulo debe permitir crear un nuevo miembro estudiantil. Para la creación del miembro estudiantil se deberán almacenar los siguientes datos: la *sección* y la *región* a la que pertenece, *número de membresía*, *universidad*, *cargo* que tiene en la Rama Estudiantil, *nombres, apellidos*, *correo electrónico*, teléfono, dirección del domicilio, una breve descripción de su *voluntariado en el IEEE*, *foto*, el estado del miembro estudiantil: activo, inactivo, pendiente o en transición.
    i. Activo: cuando el miembro Estudiantil se registra al sistema.
    ii. Inactivo: cuando se ha dado de baja al Miembro Estudiantil por el Administrador de Sección.
    iii. Pendiente: cuando el miembro estudiantil ingresó sus datos pero aún no ha ingresado al sistema por primera vez.
    iv. Transición: cuando el Miembro Estudiantil ha solicitado su cambio de rol a mentor.

- El miembro estudiantil debe poder escoger la Región a la que pertenece y se le debe mostrar todas las Secciones de esa Región (previamente creada en el sistema) para que el miembro estudiantil pueda escoger su ubicación geográfica.
- El cargo que se le mostrará al estudiante, deben ser todos los de tipo *Rama Estudiantil.*
- Cuando el Miembro Estudiantil ingresa sus datos, el sistema le debe enviar un correo electrónico para informarle que sus datos han sido ingresados exitosamente, indicándole su contraseña creada aleatoriamente y el link donde debe ingresar. Esto sirve para verificar que el correo existe.
- Una vez que el Miembro Estudiantil recibe este correo, debe ingresar al link indicado.
- Una vez completado el proceso de registro, los miembros estudiantiles deben poder actualizar o modificar todos sus datos personales.
- El administrador de Sección debe poder visualizar todos los miembros estudiantiles de su Sección, con las actividades correspondientes en las que éste se encuentre involucrado, pudiendo consultar el detalle de éstas.
- Un miembro estudiantil puede ser dado de baja por el Administrador de Sección, en caso de que lo considere necesario.
- Al querer dar de baja a un estudiante, y si el estudiante a dar de baja tiene una actividad en ejecución, se debe escoger a otro estudiante de la misma universidad que esté disponible. Si no

hubiese estudiantes, no se debe poder dar de baja al estudiante a cargo de la actividad y el Administrador de Sección debe ponerse en contacto con el Presidente de la Rama para asignar un nuevo miembro estudiantil encargado.

- Un miembro estudiantil puede llegar a ser mentor, para realizar esta transición, el miembro estudiantil debe actualizar los datos que considere necesarios, además de seleccionar las categorías y las subcategorías de acuerdo a su experiencia.
- Cuando un miembro estudiantil solicite su cambio de rol a “Mentor”, el sistema debe enviar un correo al administrador de Sección, para que realice la aprobación. Así mismo debe enviar un correo electrónico al miembro estudiantil indicando que su solicitud fue enviada al Administrador de Sección, quien se encargará de aprobarla o rechazarla.
- Cuando un Miembro Estudiantil solicite su cambio de rol a Mentor, su rol no debe cambiar aún, pero su estado debe ser *Transición*.
- Mientras un miembro estudiantil esté en estado *Transición* no podrá crear actividades ni consultorías.
- Si la solicitud de cambio de rol es aprobada por el Administrador de Sección, el rol definitivo del solicitante debe ser *Mentor* y el estado: *Activo*.
- Si la solicitud de cambio de rol es desaprobada por el Administrador de Sección, el rol definitivo será Miembro Estudiantil y el estado: Activo.

- El módulo debe permitir crear un nuevo usuario para el sistema de tipo *Administrador de Región, Administrador de Sección, Revisor de Región y Revisor de Sección*. Para la creación de un nuevo usuario de estos tipos, se deben almacenar los siguientes datos: la *Sección* y la *Región* a la que pertenece, *nombres* y *apellidos* del usuario, *correo electrónico*, *teléfono*, *dirección del domicilio*, *cargo en el IEEE*, *rol del usuario*, el cual puede ser: Administrador de Región, Administrador de Sección, Revisor de Región y Revisor de Sección y el estado del usuario: activo, inactivo, pendiente o transición.
  i. Activo: Cuando el usuario se registra al sistema.
  ii. Inactivo: Cuando se ha dado de baja al usuario.
  iii. Pendiente: Cuando el usuario ingresó sus datos pero aún no ha sido aprobado.
  iv. Transición: Cuando el usuario ha solicitado un cambio de rol.
- Si el usuario a crear es de tipo Administrador de Sección, Administrador de Región o Revisor de Región, deben pasar por la aprobación del Administrador de Región antes de poder ingresar al Sistema.
- Cuando un usuario ingrese sus datos, el sistema debe enviar un correo al correspondiente administrador (de Región o de Sección) para indicarle que un nuevo usuario está pendiente de aprobación.

- Si el usuario a crear es de tipo Revisor de Sección debe pasar por la aprobación del administrador de Sección antes de poder ingresar al sistema.
- Luego de que el Administrador de Región ha aprobado a un usuario de tipo Administrador de Sección, Administrador de Región o Revisor de Región, el sistema debe enviar un correo a este usuario para indicarle que ha sido aprobado para ingresar al sistema. En este correo también se le debe indicar la contraseña para ingresar al sistema, que será creada aleatoriamente.
- Un mentor puede ascender a Administrador de Sección. En este caso el mentor debe actualizar el rol con el que desea ingresar al sistema, cambiándolo de *mentor* a *Administrador de Sección*. Antes de realizar esta operación y de ser necesario, se debe dar de baja al Administrador de Sección anterior.
- Cuando un mentor solicita su ascenso a Administrador de Sección, esta solicitud debe pasar por la aprobación del Administrador de Región, a quien el sistema le debe enviar un correo electrónico indicándole la solicitud.
- Cuando un Mentor solicite su cambio de rol a Administrador de Sección, su rol no debe cambiar pero su estado debe ser *Transición* hasta que sea aprobado por el Administrador de Región.
- Si la solicitud de cambio de rol del mentor es aprobada por el Administrador de Región, el rol definitivo debe ser *Administrador de Sección* y el estado: *Activo*.

- Si la solicitud de cambio de rol es desaprobada por el Administrador de Región, el rol definitivo será *mentor* y el estado: *Activo*.
- Un Revisor de Sección puede ascender a Administrador de Sección. En este caso el Revisor de Sección debe actualizar el rol con el que desea ingresar al sistema, cambiándolo de *Revisor de Sección* a *Administrador de Sección*. Antes de realizar esta operación y de ser necesario, se debe dar de baja al administrador de Sección anterior.
- Cuando un Revisor de Sección solicita su ascenso a Administrador de Sección, esta solicitud debe pasar por la aprobación del Administrador de Región, a quien el sistema le debe enviar un correo indicándole la solicitud.
- Cuando un Revisor de Sección solicite su cambio de rol a Administrador de Sección, su rol no debe cambiar pero su estado debe ser *Transición*.
- Si la solicitud de cambio de rol es aprobada por el Administrador de Región, el rol definitivo debe ser *Administrador de Sección* y el estado: *Activo*.
- Si la solicitud de cambio de rol es desaprobada por el Administrador de Región, el rol del solicitante se mantendrá en *Revisor de Sección* y el estado: *Activo*.
- Un Revisor de Sección puede ascender a Revisor de Región. En este caso el Revisor de Sección debe actualizar el rol con el que

desea ingresar al sistema, cambiándolo de *Revisor de Sección* a *Revisor de Región*.

- Cuando un Revisor de Sección solicita su ascenso a Revisor de Región, esta solicitud debe pasar por la aprobación del Administrador de Región, a quien el sistema le debe enviar un correo indicándole la solicitud.
- Cuando un Revisor de Sección solicite su cambio de rol a Revisor de Región, su rol no debe cambiar pero su estado debe ser *Transición*.
- Si la solicitud de cambio de rol es aprobada por el Administrador de Región, el rol definitivo debe ser *Revisor de Región* y el estado: *Activo*.
- Si la solicitud de cambio de rol es desaprobada por el Administrador de Región, el rol definitivo será *Revisor de Región* y el estado: *Activo.*
- Un Administrador de Sección puede ascender a Administrador de Región. En este caso el Administrador de Sección debe actualizar el rol con el que desea ingresar al sistema, cambiándolo de Administrador de Sección a Administrador de Región. Antes de realizar esta operación y de ser necesario, se debe dar de baja al administrador de Región anterior.
- Cuando un Administrador de Sección solicite su cambio de rol a Administrador de Región, su rol debe cambiar a Administrador de Región y su estado debe ser “Transición”.

- Si la solicitud de cambio de rol es aprobada por el Administrador general, el rol definitivo debe ser Administrador de Región y el estado: Activo.
- Si la solicitud de cambio de rol es desaprobada por el Administrador general, el rol definitivo será Administrador de Sección y el estado: Activo.
- Cuando un Administrador de Sección solicita su ascenso a Administrador de Región, esta solicitud debe pasar por la aprobación del Administrador General, a quien el sistema le debe enviar un correo indicándole la solicitud.
- En este módulo, todos los usuarios deben poder modificar sus datos personales.

ADMINISTRACIÓN DE CATEGORIAS Y SUBCATEGORIAS

- El módulo permitirá al Administrador de Región crear nuevas Categorías. Los datos que se deberán almacenar para cada categoría son los siguientes: *nombre de la categoría*, *descripción* y el *estado* de la categoría: activa, inactiva o sugerida.
    i. Activa: Cuando la categoría ha sido creada por el Administrador de Región o a su vez fue aprobada de una sugerencia.
    ii. Inactiva: Cuando la categoría ha sido dada de baja por el Administrador de Región.
    iii. Sugerida: Cuando una categoría ha sido sugerida y aún no ha sido aprobada.

- El módulo debe permitir al Administrador de Región crear nuevas subcategorías. Los datos que se deben ingresar para cada subcategoría son los siguientes: *nombre de la subcategoría*, *descripción*, *categoría* a la que pertenece y el *estado* de la Subcategoría: activa, inactiva o sugerida.
    i. Activa: Cuando la subcategoría ha sido creada por el Administrador de Región o a su vez fue aprobada de una sugerencia.
    ii. Inactiva: Cuando la subcategoría ha sido dada de baja por el Administrador de Región.
    iii. Pendiente: Cuando una subcategoría ha sido sugerida y aún no ha sido aprobada.
- En este módulo el Administrador de Sección debe poder habilitar o deshabilitar las categorías y Subcategorías que requiera de acuerdo a su Sección.
- Si el Administrador de Sección intenta deshabilitar categorías o Subcategorías que están siendo usadas en su Sección por algún miembro estudiantil en una Actividad, el sistema debe mostrarle un aviso indicando que hay una consultoría activa con esa categoría o subcategoría y que debe esperar a que ésta termine.
- En este módulo el Administrador de Región debe poder modificar datos de las categorías o subcategorías.
- Si se considera que una categoría o subcategoría ya no es relevante, el Administrador de Región debe poder cambiar su estado de activa a inactiva.

- Al inactivar una categoría, se deben inactivar todas sus subcategorías.
- Al momento de crear una nueva categoría o subcategoría, el sistema debe enviar un correo electrónico a los mentores, para que modifiquen su perfil si lo desean, agregando las nuevas.
- Si existe alguna categoría o subcategoría que el mentor, el revisor de Sección, Administrador de Sección o el miembro estudiantil consideren que es importante y no está creada, el sistema les debe permitir hacer una sugerencia de nueva categoría o subcategoría al Administrador de Región, indicando sus razones y éste a su vez las debe poder aprobar o desaprobar. Estas sugerencias deben quedar almacenadas en la base de datos, en la tabla **Categorías** y **Subcategorías**, con el estado *Pendiente*.
- Las sugerencias que el Administrador de Región acepte, deben quedar almacenadas en la base de datos, en la tabla **Categorías** y **Subcategorías**, con el estado *Activa*.

ADMINISTRACIÓN DE ACTIVIDADES

- El sistema debe permitir a los miembros estudiantiles ingresar nuevas actividades a realizar en sus Ramas Estudiantiles.
- Los datos que se deben almacenar por cada actividad son los siguientes: *nombre de la actividad*, *descripción*, *objetivo de la actividad*, *fecha de inicio*, *fecha de fin*, referencia a la persona responsable de la actividad, el estado de la actividad.

- La fecha de inicio de la actividad se debe ingresar automáticamente cuando se crea la actividad.
- La fecha de fin de la actividad debe corresponder a la fecha en la que el estudiante da por terminada la actividad.
- Los estados de la Actividad pueden ser los siguientes:
    - i. Suspendida: Cuando el estudiante no quiere continuar la actividad.
    - ii. Terminada: Cuando el estudiante termina la actividad cumplida.
    - iii. Activo: Cuando la actividad está en marcha.
    - iv. Inactiva: Cuando el estudiante o el mentor no responden en un lapso de 4 semanas.
    - v. Pendiente: Cuando una actividad aún no ha sido aprobada por el administrador de sección.
- El sistema debe enviar al administrador de Sección un correo electrónico notificando que tiene que aprobar una nueva actividad creada.
- El sistema debe permitir al Administrador de Sección aprobar actividades ingresadas por los miembros estudiantiles.
- Una vez aprobada el estado de la actividad debe cambiar de *Pendiente* a *Activo*.
- El sistema debe enviar un correo electrónico automático al miembro estudiantil una vez que el Administrador de Sección haya aprobado su actividad.

- El sistema debe permitir a los miembros estudiantiles modificar los datos ingresados de una actividad.
- El sistema debe permitir cambiar el miembro estudiantil responsable de una actividad ya creada.
- El miembro estudiantil debe poder **finalizar** una actividad, en el caso de que esto ocurra, el sistema lo debe direccionar a hacer la respectiva encuesta y todas las consultorías asociadas a la actividad deben pasar al estado *Terminada.*
- El miembro estudiantil puede **suspender** una actividad, en el caso de que esto ocurra, el sistema le debe pedir ingresar los motivos de la suspensión y todas las consultorías asociadas a la actividad pasarán al estado *Suspendida.*
- Cuando una actividad, y por ende sus consultorías tienen el estado *terminada, suspendida, inactiva o pendiente*, el mentor no podrá responder preguntas ni el miembro estudiantil podrá realizar consultas.
- Una actividad debe poder tener varias consultorías asociadas a ella, no necesariamente con el mismo mentor.
- El mentor debe poder ver todas las consultorías de las actividades en las cuales se encuentra su consultorías, pero no debe poder responder preguntas de esas consultorías, solo debe poder consultarlas.

ADMINISTRACIÓN DE CONSULTORÍA

- El sistema debe permitir al miembro estudiantil realizar consultorías respecto a una actividad, luego de haberla creado y que ésta haya sido aprobada por el Administrador de Sección.
- Para crear la consultoría el miembro estudiantil debe escoger la categoría y la subcategoría que desea, referente a su actividad. Aquí se deben mostrar todas las categorías y Subcategorías relacionadas a su Sección.
- Luego de seleccionar la categoría y subcategoría, el sistema le debe asignar un mentor de su misma sección que coincida con la subcategoría seleccionada por el miembro estudiantil.
- En caso de que el sistema no encuentre un mentor acorde a la categoría y subcategoría, se debe enviar un correo al administrador de sección para que gestione la búsqueda de un mentor.
- Cuando el sistema le asigna un mentor a la categoría y subcategoría seleccionada por el miembro estudiantil, el estado de la consultoría, cambia a *Activo*.
- En caso de existir más de un mentor disponible en la categoría y subcategoría seleccionada, el sistema debe escoger al que tenga menos de 3 consultorías activas y tenga más tiempo registrado en el sistema como mentor, es decir, el que tenga mayor antigüedad.
- Un mentor no debe poder tener más de 3 consultorías activas. En ese caso, se lo debe considerar no disponible y no se le debe

poder asignar una nueva consultoría, hasta que haya concluido al menos una de las que tiene asignadas.

- Cuando el Administrador de Sección apruebe un mentor para una consultoría, el sistema debe enviar un correo electrónico al mentor indicándole la actividad que va a asesorar, el responsable de la actividad y la Rama estudiantil a la que pertenece. Además le debe enviar un correo electrónico al miembro estudiantil indicando quién es el mentor asignado para dicha consultoría.
- Una vez asignado el mentor, el estudiante debe poder realizar las consultas que desee al mentor.
- Un estudiante no debe poder ingresar otra consulta hasta que el mentor no responda la última consulta.
- Cada vez que el miembro estudiantil realice una consulta al mentor, el sistema debe enviar un correo electrónico al mentor, indicándole que tiene una consulta pendiente en el sistema.
- Cada vez que el mentor responda una consulta, el sistema debe enviar un correo electrónico al miembro estudiantil responsable, para indicarle que su consulta ha sido contestada. En este correo se debe especificar el mentor y la consultoría a la cual pertenece la respuesta, pues el miembro estudiantil puede tener varias consultorías activas.
- En caso de que no exista un mentor disponible para la categoría seleccionada, se debe poner la petición del estudiante en lista de espera (cambia el estado de la consultoría a *En espera*) y el sistema le debe notificar esto al Administrador de Sección

mediante un correo electrónico, para que reclute un mentor que cumpla con los requisitos que se solicitan.

- La consultoría debe poder estar en estado *En Espera* máximo por 2 semanas, cumplido el tiempo máximo, debe cambiar el estado a *Inactivo* y se debe notificar al miembro estudiantil que no existe un mentor para la subcategoría seleccionada.
- Las consultas ingresadas por el estudiante deben poder ser modificadas, siempre y cuando la consulta no haya sido respondida por el mentor.
- Si el estudiante modifica una pregunta (siempre y cuando no ha sido respondida por el mentor) se le debe enviar un correo notificando al mentor del cambio realizado.
- Las respuestas ingresadas por el mentor, no deben poder ser modificadas. En caso de requerir un cambio en la información que ingresaron, el mentor deberá crear una nueva respuesta.
- El sistema debe permitir al administrador de Sección cambiar el mentor de una consultoría ya creada, sin perder las consultas realizadas al mentor anterior, ingresando un comentario por el cambio.
- El sistema debe permitir a los miembros estudiantiles y mentores enviar comentarios al administrador de Sección durante la duración de la consultoría.
- El sistema deber validar que una vez finalizado una consultoría, el estudiante no pueda realizar consultas al mentor o enviar comentarios al administrador de Sección.

ADMINISTRACIÓN DE EVALUACIÓN DE CONSULTORÍA

- El sistema debe permitir al Administrador de Sección y al Administrador de Región crear evaluaciones que deben constar de al menos 1 sección y al menos 1 pregunta.
- Las preguntas pueden ser de tipo de selección simple, selección múltiple y abierta.
- Las evaluaciones pueden ser referentes a una consultoría o *abiertas*, es decir, sin tener relación con una actividad.
- Las evaluaciones referentes a una consultoría solo pueden ser creadas por el Administrador de Región y deben ser dirigidas al último miembro estudiantil que estuvo a cargo de la actividad y a todos los mentores que se involucraron dando consultorías en dicha actividad.
- En las evaluaciones referentes a una consultoría se debe escoger la subcategoría a la cual irá relacionada la evaluación.
- Las evaluaciones *abiertas* deben poder ser creadas por el Administrador de Región o por el Administrador de Sección.
- El sistema debe permitir al Administrador de Sección y al Administrador de Región modificar únicamente las evaluaciones que ellos hayan creado.
- El sistema debe permitir al miembro estudiantil y al mentor realizar evaluación de la consultoría, luego de haberla finalizado.

3.2. Requerimientos no funcionales

En esta sección se definen los requerimientos que debe cumplir el sistema en cuanto a confiabilidad y seguridad.

- Los datos presentados a los usuarios deben ser correctos y no deben prestarse a errores que puedan confundir a los usuarios.
- El proceso de consultoría debe ser claro y tener una lógica bien definida, para que los usuarios puedan realizarlo sin confusiones.
- El envío de correos debe realizarse de manera automática cada vez que se requiera.
- Debe existir coherencia en cuanto a la información presentada.

Los requerimientos en cuanto a hardware son los siguientes:

- Capacidad de Memoria en el ordenador: Para un buen desempeño del sistema, una memoria de 512 MB de RAM.
- Capacidad de Memoria en el servidor: Se recomienda un mínimo de 1GB de RAM.
- Capacidad de disco duro en el ordenador: Para un buen desempeño del sistema se recomienda un disco duro mínimo de 30 GB.
- Capacidad de disco duro en el servidor: Se recomienda un de disco duro de 120 GB como mínimo.
- Tipo de procesador en un ordenador: Recomendado Pentium III, equivalente ó superior.
- Tipo de procesador en servidor: Se recomienda un procesador Pentium IV de 3.2Ghz o equivalente.
- Ancho de Banda para el Servidor: Ancho de banda recomendado 128Mbits.

- Navegadores: Para un buen uso del sistema, se requiere Internet Explorer 5.0 o superior y Firefox 1.5 o superior.
- Sistema Operativo: Microsoft Windows 98 o superior o cualquier versión de Linux.

3.3. Análisis de las herramientas de desarrollo.

Para el manejo de la base de datos escogimos el motor MySQL 5.0.4 ya que es un motor muy popular, confiable, multiplataforma, Las principales características son las siguientes:

- Implementación multihilo, lo que hace que aproveche la potencia de sistemas multiprocesador.
- Soporta gran cantidad de tipos de datos para las columnas.
- Los clientes pueden ser creados en distintos lenguajes y para nuestro caso si soportará Java.
- Soporta hasta 32 índices por tabla.
- Gestión de usuarios, contraseñas, y privilegios muy flexible y seguro.
- Las funciones SQL están implementadas usando una librería altamente optimizada y deben ser tan rápidas como sea posible.
- No hay reserva de memoria tras toda la inicialización para consultas.

Para la implementación de la solución, elegimos el lenguaje de programación Java, porque posee características muy eficaces, a continuación mencionamos algunas:

- **Sencillez:** Java es un lenguaje de gran facilidad de aprendizaje, pues en su concepción se eliminaron todos aquellos elementos que no se consideraron absolutamente necesarios. Por ejemplo, en comparación con otros lenguajes como C ó C++, es notable la ausencia de punteros, o lo que es lo mismo: es imposible hacer referencia de forma explícita a una posición de memoria; ello ahorra gran cantidad de tiempo al momento de programar, dado que el comportamiento imprevisto de los punteros es una de las principales fuentes de errores en la ejecución de un programa. Por otra parte, el código escrito en Java es por lo general mucho más legible que el escrito en C ó C++. Por otro lado, Java dispone de un mecanismo conocido como de "recogida de basura", el cual, usando la capacidad multitarea de Java, hace que, durante la ejecución de un programa, los objetos que ya no se utilizan se eliminen automáticamente de la memoria. Dicho mecanismo facilita enormemente el diseño de un programa y optimiza los recursos de la máquina que se esté usando para la ejecución del mismo (con los lenguajes tradicionales, la eliminación de objetos y la consiguiente optimización de recursos debe planificarse cuidadosamente al idear el programa).

- **Orientación a objetos:** Java es un lenguaje orientado a objetos desde su concepción y esto hace que las aplicaciones escritas en Java tengan interesantes ventajas. En el caso de los lenguajes orientados a objetos, el concepto clave no es el de función, sino el de objeto. Un objeto es un elemento de programación, autocontenido y reutilizable, y que podríamos definir como la representación en un programa de un concepto, representación que está formada por un conjunto de variables (los datos) y un conjunto de métodos (o instrucciones para manejar los datos). Los objetos, además, poseen la capacidad de enviarse mensajes entre sí durante la ejecución de un programa. La "encapsulación" de variables y métodos en un objeto tiene claras ventajas:

- Cada objeto puede ser modificado y mantenido por separado.
- Se pueden mantener en un objeto métodos y variables que no son accesibles desde fuera de él, lo que evita multitud de posibilidades de error en el momento de confeccionar un programa. Esta característica se llama "ocultamiento de la información".
- Es posible reutilizar porciones de programa ya escritas.

Otra consecuencia de la orientación a objetos de Java es que hace que el software escrito en Java sea modular: como el objeto es la entidad clave en la programación, cada uno puede ser modificado y mantenido por separado. Además, en Java un programa no es más que un conjunto de ficheros compilados para la VM (llamados "módulos"), que no es necesario "enlazar" en un único ejecutable como ocurría en el caso de los lenguajes compilados. Esto significa que pueden realizarse modificaciones sobre cada uno de los "módulos" sin necesidad de tener que recompilar y enlazar todos ellos: basta compilar sólo los "módulos" afectados.

- **Seguridad:** En general, se considera que un lenguaje es tanto más seguro cuanto menor es la posibilidad de que errores en la programación, o diseños malintencionados de programas (virus), causen daños en el sistema. La extrema seguridad de Java se establece a tres niveles:
    - *Nivel de seguridad dado por las características del lenguaje*, tales como la ausencia de punteros (que evita cualquier error de asignación de memoria) o el "ocultamiento de la información" propio de la programación orientada a objetos, por recordar dos ejemplos ya mencionados.
    - *Nivel de seguridad dado por el diseño de la VM*:
        - La VM de Java posee un verificador de los byte codes, que antes de ejecutarlos analiza su formato comprobando que no existen punteros en ellos, que se accede a los recursos del sistema a través de objetos de Java, etc.
        - Otro elemento constitutivo de la VM es el cargador de clases. Una clase es una categoría de objetos utilizados en un programa; cuando se ejecuta un programa en Java, éste llama a determinadas clases a través del cargador de clases. Estas clases pueden provenir de tres lugares distintos, en donde residen en forma de archivos: del computador local, de la red de área local a la que pueda estar

conectado el ordenador cliente, o de Internet. En función de la procedencia de las clases, se efectúan una serie de comprobaciones diferentes y el gestor de seguridad de la VM prohíbe los accesos peligrosos.

- *Nivel de seguridad dado por la API de Java*. El conjunto de métodos y clases que estamos obligados a utilizar cuando programamos en Java para acceder a los recursos del sistema, está definido por la API, y constituye la última barrera defensiva. El diseño de dichos métodos y clases hace que éstos realicen múltiples verificaciones cuando son invocados, de modo que se dificultan los errores (voluntarios o involuntarios).

- **Arquitectura neutral:** Java permite que sus aplicaciones se interpreten de manera independiente a la plataforma donde se esté trabajando. Para hacer independientes las aplicaciones desarrolladas en JAVA, se compila su código en un archivo objeto (byte codes) que no depende de la arquitectura de la máquina donde se esté ejecutando, y las aplicaciones se comportarán de la misma manera desde Windows, Linux o Macintosh. Cualquier máquina que tenga el "run-time" puede ejecutar código escrito en Java. Para la realización de este proyecto usaremos JSDK 6.0.

- **Multihilos:** Java proporciona la programación multihilo, la cual permite la escritura de programas que realicen varias tareas simultáneamente. Java utiliza un método de memoria conocido como "Administración automática del almacenamiento" (automatic storage management), en el que el sistema en tiempo de ejecución de Java mantiene un seguimiento de los objetos. En el momento que no están siendo referenciados por alguien, automáticamente se libera la memoria asociada con ellos.

- **Manejo de errores:** Una característica importante de Java y muy aplicable en nuestro proyecto es la administración de errores y excepciones. Las excepciones son la manera como Java indica que ha ocurrido algo extraño durante la ejecución de un programa escrito en Java. Comúnmente las excepciones son generadas y lanzadas por el sistema cuando estos eventos ocurren.

Definición de Roles en la aplicación

Para el uso de la aplicación se han dividido los roles dependiendo en cierta parte de su grado de membresía en el IEEE. A continuación se muestran los principales roles considerados para la interacción en el sistema:

Rol	Función
Mentor	Encargado de responder las consultorías que realice el estudiante.
Miembro estudiantil	Encargado de crear nuevas actividades relativas a categorías y hacer consultorías al mentor.
Administrador de Sección	Administra los usuarios, universidades y demás dentro de su Sección
Administrador de Región	Administra las entidades necesarias para la administración base del sistema, usuarios regionales, administración de encuestas y

	demás dentro de su Región
Revisor de Región	Encargado de hacer revisiones generales del movimiento de las consultorías en toda la Región.
Revisor de Sección	Encargado de hacer revisiones generales del movimiento de las consultorías en toda la Sección a la que pertenece.

Tabla 3.1 Roles del sistema

3.4. Especificación UML del problema

En esta sección, mediante el uso de técnicas de modelamiento se definirá el concepto de la solución aplicado a una especificación formal.

El Lenguaje Modelad Unificado, UML es un lenguaje de modelado visual estándar, que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

En este proyecto se plasma el uso de UML y su aplicación en el análisis y diseño de sistemas. A continuación se define el diagrama de casos de uso, que muestra una visión de todos los componentes interactuando entre sí. Luego se especifican los casos de uso en el que se definirán todos los casos posibles contemplados en los requerimientos funcionales. Posteriormente se especifican los escenarios posibles que pueden contemplarse a partir de los casos de usos especificados anteriormente. Finalmente se muestra un diseño de las clases en el diagrama de clases, para mostrar la división de objetos en el sistema.

3.4.1. Casos de Uso

CASOS DE USO	ID
Iniciar Sesión	CU-001

Cerrar sesión	CU-002
Administrador de Región crea una evaluación	CU-003
Administrador de Región crea una sección a una evaluación	CU-004
Administrador de Región crea una pregunta para una sección de una evaluación	CU-005
Administrador de Región crea una alternativa para una pregunta para una sección de una evaluación	CU-007
Administrador de Región modifica evaluación	CU-008
Administrador de Región crea categoría	CU-009
Administrador de Región modifica categoría	CU-010
Administrador de Región crea subcategoría	CU-011
Administrador de Región modifica subcategoría	CU-012
Administrador de Sección asigna categorías/Subcategorías a su sección	CU-013
Recomendar nueva categoría	CU-014
Recomendar nueva subcategoría	CU-015
Administrador de Sección crea universidades	CU-016
Administrador de Región crea cargos IEEE	CU-017
Administrador de Región crea Secciones IEEE	CU-020
Administrador de Región consulta respuestas a una evaluación	CU-021
Responder Evaluación	CU-022
Crear consultoría	CU-023
Realizar consulta dentro de consultoría	CU-024
Responder consulta dentro de consultoría	CU-025
Modificar consulta dentro de consultoría	CU-026
Modificar estado de la consultoría	CU-027
Ingresar sugerencia referente a una consultoría	CU-028
Consultar Consultorías anteriores de toda la Región	CU-029

Estudiante crea nueva actividad	CU-030
Administrador de Sección aprueba actividad	CU-031
Cambiar estudiante responsable de actividad	CU-032
Registro de usuario en el sistema	CU-033
Administrador de Sección aprueba/rechaza usuario mentor	CU-034
Administrador de Región aprueba/rechaza usuario Administrador de Sección	CU-035
Modificar datos de usuario	CU-036
Solicitar cambio de Rol	CU-037
Consultar directorio GOLD-Región	CU-038
Consultar directorio GOLD-Sección	CU-039
Cambiar contraseña	CU-040

Tabla 3.2 Casos de uso de la aplicación

3.4.2. Diagrama de Casos de usos

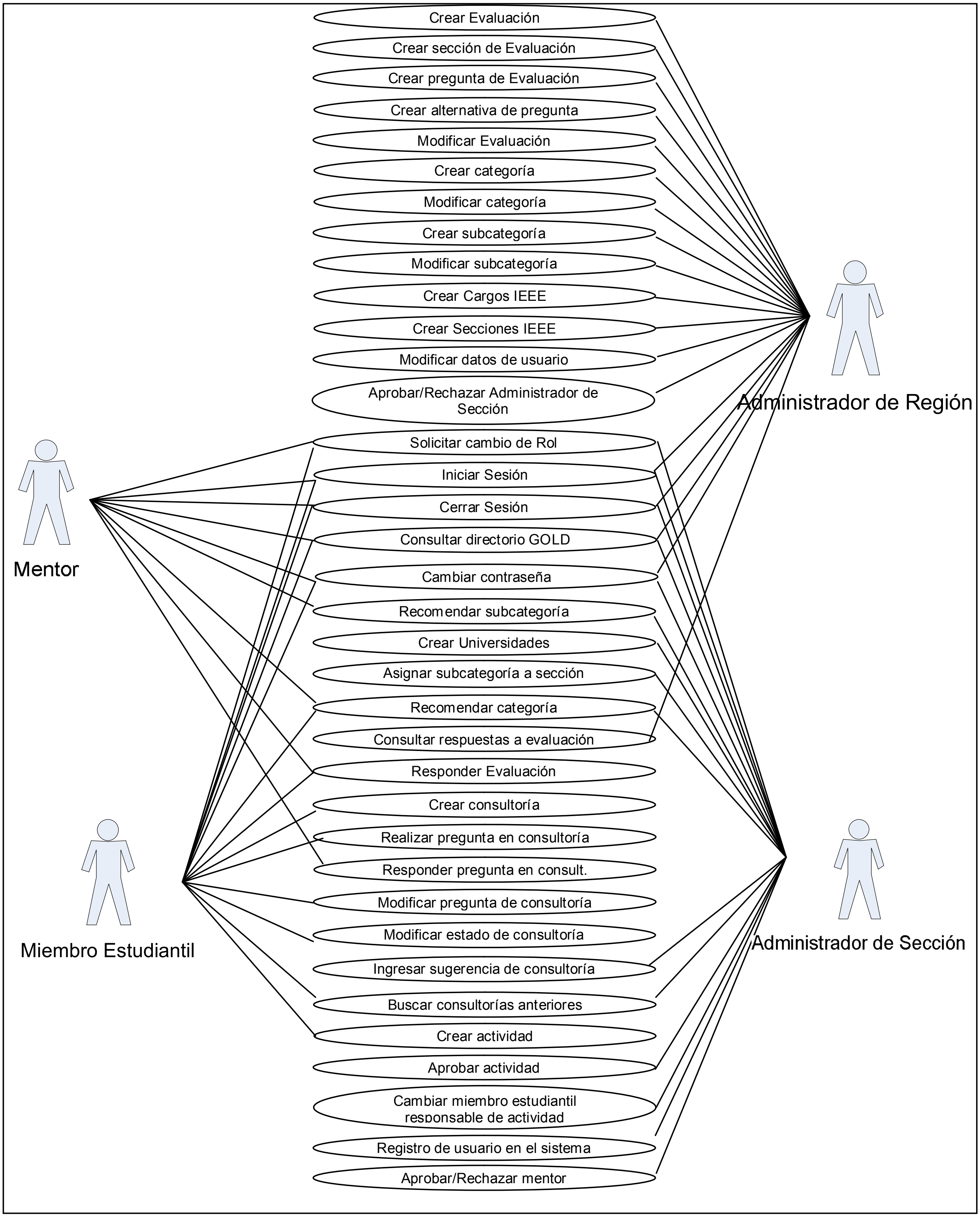

Figura 3.1 Diagrama de casos de uso

3.4.3. Escenarios

Los escenarios planteados para los casos de uso se detallan en el ANEXO A, junto a la descripción de los casos de uso.

3.4.4. Diagrama de Clases

En el desarrollo del proyecto se utilizó la herramienta de mapeo MyEclipse en el que podemos realizar un mapeo de la base de datos con las clases del sistema, de ésta manera se aplica el modelo de persistencia Hibernate explicado en el capítulo 5 de este documento.

Al realizar éste mapeo, da como resultado que cada tabla de la base de datos se convierte en una clase, con su respectivo controlador, de ésta manera el mismo diagrama de base de datos se convierte en el diagrama de clases, cada clase es un objeto de la base de datos. Para referirse a éste diagrama se puede referir en la *figura 4.10 Modelo Relacional de la Base de datos.*

3.5. Modelo de Seguridades

Un software o aplicación pueden ser inseguros dependiendo de la vulnerabilidad que se le pase por alto a un desarrollador.

El único esquema de seguridad implementado en el sistema es el de autenticación, verificación y validación de quienes utilizan las funcionalidades.

Autenticación.- Asignando un usuario y clave a cada usuario del sistema para utilizar las funcionalidades, así nos aseguramos que quienes lo utilicen sean personas relacionadas al voluntariado en el IEEE.

Verificación.- Identificar que cuando usuario trate de ingresar al sistema, exista en la base de datos de usuarios del aplicativo.

Validación.- Asegurarse que quien dice estar ingresando tiene un usuario con acceso al sistema, y cumple determinado rol.

Un proceso adicional de verificación lo cumple el coordinador de las actividades GOLD, quien debe verificar si realmente una persona es voluntaria de una Rama Estudiantil y si un profesional es Miembro de determinada Sección del IEEE en la Región 9.

3.6. Análisis financiero de la solución

Para el desarrollo de este proyecto, se necesitó de dos desarrolladores y dos personas que conozcan del negocio. Adicionalmente se necesitó de los siguientes conceptos que se detallan según costos aproximados:

Concepto	Costo
Horas de programación	$ 0
2 Laptops	$2400
Movilización	$200
Materiales de oficina	$40
Energía eléctrica	$240
Alimentación	$200
TOTAL	***$ 3080***

Tabla 3.3: Tabla de costos del proyecto

Este proyecto al ser para el IEEE, una Institución sin fines de lucro, no tiene un retorno económico de su inversión, más bien su enfoque está en beneficiar el voluntariado y la relación entre estudiantes de Ingeniería y Profesionales en los países Latinoamericanos, trayendo como consecuencia los siguientes beneficios:

- Una mejor comunicación entre los voluntarios estudiantiles y profesionales del IEEE en las 32 Secciones que actualmente existen en la Región 9.
- Una base de conocimiento sobre las experiencias que van teniendo los voluntarios que usan la aplicación para que futuros líderes tengan una referencia.
- Un canal de comunicación más efectivo y directo para las actividades estudiantiles.

Como vemos el retorno de la inversión que se ha tenido para este proyecto tienen un valor muy importante para el desarrollo de las actividades que son organizadas por voluntarios del IEEE cada mes en nuestra Región.

CAPITULO 4

4. DISEÑO DEL SISTEMA

4.1. Visión general del diseño

Siguiendo la línea de Java, se utilizó para el desarrollo de esta aplicación un *Diseño orientado a objetos*, el cual es una fase de la metodología orientada a objetos para el desarrollo de Software. Su uso induce a los programadores a pensar en términos de objetos, en vez de procedimientos, cuando planifican su código. Un objeto agrupa datos encapsulados y procedimientos para representar una entidad. La 'interfaz del objeto', esto es, las formas de interactuar con el objeto, también es definida en esta etapa. Un programa orientado a objetos es descrito por la interacción de esos objetos. El diseño orientado a objetos es la disciplina que define los objetos y sus interacciones para resolver un problema de negocio que fue identificado y documentado durante el análisis orientado a objetos. [9]

A continuación se mostrarán los patrones de diseño utilizados según las recomendaciones J2EE.

4.2. Patrones de diseño

Los patrones de diseño J2EE son soluciones bien documentadas de problemas comunes que surgen en el diseño de aplicaciones J2EE.

En todo patrón J2EE se define lo siguiente:

- Contexto: Circunstancias bajo las cuales el patrón existe.
- Problema: Describe el problema que trata se trata de resolver por medio del patrón.
- Motivación: Lista de las razones y motivaciones que afectan al problema y a la solución.
- Solución: Describe la solución brevemente y los elementos de la solución en detalle.
- Estrategias: Describen diferentes formas de cómo un patrón puede ser implementado.

Existe una amplia gama de patrones de diseño propuestos para ser utilizados en el desarrollo de aplicaciones Web. Luego de realizar el análisis del diseño de la aplicación, hemos determinado un conjunto de patrones a ser utilizados para el desarrollo del portal. Para el análisis se consideró la facilidad de implementación de los patrones, la claridad al momento de implementar el código y cuáles eran los patrones que mejor se ajustaban a los requerimientos de la aplicación a desarrollar.
Considerando lo expuesto en el párrafo anterior, el conjunto de patrones de diseño seleccionados para el portal fueron los siguientes:

- Data Transfer Object (DTO)
- Data Access Object (DAO)
- Composite View
- Model View Controller (MVC)

A continuación se presentan los detalles de implementación de los patrones utilizados, junto con una breve descripción de cada uno de ellos.

Transferencia Dato-Objeto (DTO)

Este patrón de diseño propone definir clases de Java de tal manera que cada uno de los atributos corresponda a los campos que componen las tablas de la base de datos.

Los DTO (Data Transfer Object) o también denominados VO (Value Object). Son utilizados por DAO para transportar los datos desde la base de datos hacia la capa de lógica de negocio y viceversa.

Podría decirse que un DTO es un objeto común y corriente, que tiene como atributos los datos del modelo, con sus correspondientes accessors (getters y setters). En una clase de Java, los métodos getters se utilizan para recuperar información sobre los atributos y el método setter para modificar el valor de los mismos. Todos los cambios realizados a la base implican un cambio en los objetos implementados usando este patrón

En la figura 4.1 se muestra el diagrama UML de la tabla *actividad* en la base de datos, luego se muestra el diagrama UML de la clase *actividad* en la aplicación. En dicha figura se puede apreciar la aplicación del DTO.

Figura 4.1: Diagrama UML de la tabla y clase *actividad*

Objeto de acceso a Datos (DAO)

Este patrón de diseño propone separar y encapsular toda la lógica de acceso a los datos que maneja la aplicación de la implementación, logrando de esta maneja desligar la lógica de negocio y presentación del mecanismo de manejo del repositorio de datos que se utilice para almacenar la información.

A continuación se muestra la figura 4.2 que muestra un diagrama de la implementación de un patrón DAO. Como se observa en el diagrama, la lógica del negocio usa una clase DAO quien a su vez crea o usa un objeto de transferencia de datos, en el DAO está encapsulada la base de datos, que nunca se relaciona directamente con la lógica del negocio o con el DTO sino que acceden a ella a través de esta clase.

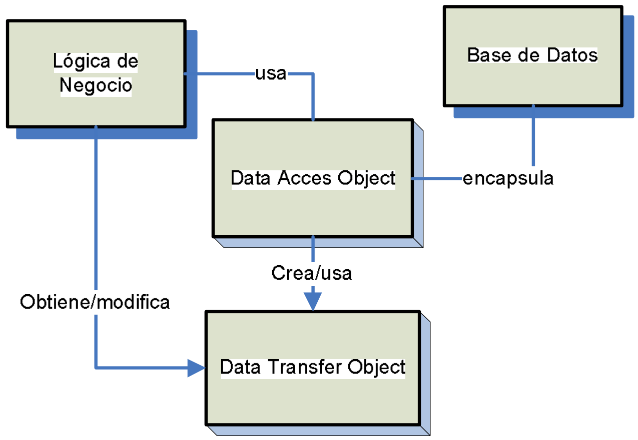

Figura 4.2: Data Access Object

Cuando la capa de lógica de negocio necesite interactuar con la base de datos, va a hacerlo a través de la API que le ofrece DAO. Generalmente esta API consiste en métodos CRUD (Create, Read, Update y Delete).

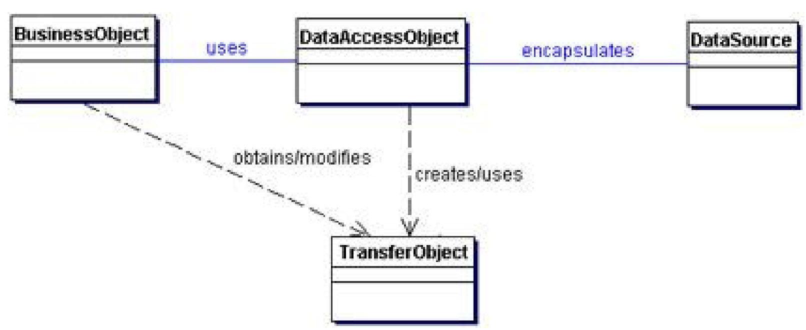

Figura 4.3 Diagrama de clases de DAO

La Figura 4.3 muestra el diagrama de clases que representa las relaciones para el patrón DAO. [9]

<u>Ventajas:</u>

- Los objetos del negocio pueden utilizar la fuente de datos sin conocer los detalles específicos de la aplicación de la fuente de datos. El acceso es transparente.

- Facilita la migración de una aplicación a una base de datos diferente.
- Simplifica el código de los objetos del negocio al realizar la complejidad del acceso a datos.
- Centraliza todos los datos en una capa separada.
- DAO utiliza objetos DVO, lo cual nos permite definir métodos cuyos prototipos no requieren una lista indefinida de parámetros (uno por cada campo de la tabla de datos).

En la aplicación web implementada para este trabajo de tesis se implementa un objeto DAO que encapsula todos los métodos de acceso a datos a una base creada en MySql Server 5.0. Cada objeto DAO tiene todos los métodos para establecer una comunicación con la base de datos, ejecutan cierta acción y un solo objeto realiza la apertura y el cierre de la conexión con la base.

4.2.1. Diseño de la aplicación y la interacción con el usuario

Vista Compuesta

Muchas páginas en una aplicación web pueden tener contenido muy diferente y algo de contenido en común. Cuando un usuario navega por las páginas, los datos y contenidos, entre las diferentes páginas varía, pero muchos elementos como un encabezado común o barra lateral en cada punto de vista siguen siendo los mismos. La estructura y el diseño de cada página puede ser el mismo en todas las páginas.

Cuando una aplicación cuenta con elementos codificados en la página directamente, son difíciles de modificar y pueden presentar incoherencias.

Normalmente, es necesario utilizar técnicas avanzadas de composición de estas páginas para simplificar su construcción y mantenimiento. Las JSP, Java Server Pages, y los servlets proporcionan mecanismos sencillos para poder incluir porciones de una página en otra (de modo estático o dinámico).

En la aplicación *Peer to Peer* usamos este patrón para el encabezado (header.jsp cuando no tiene sesión y header_sesion.jsp cuando está logoneado), para el pie de página (footer.jsp) y para la barra lateral (lateral.jsp). Esto quiere decir que las páginas JSP están compuestas por al menos 3 páginas adicionales.

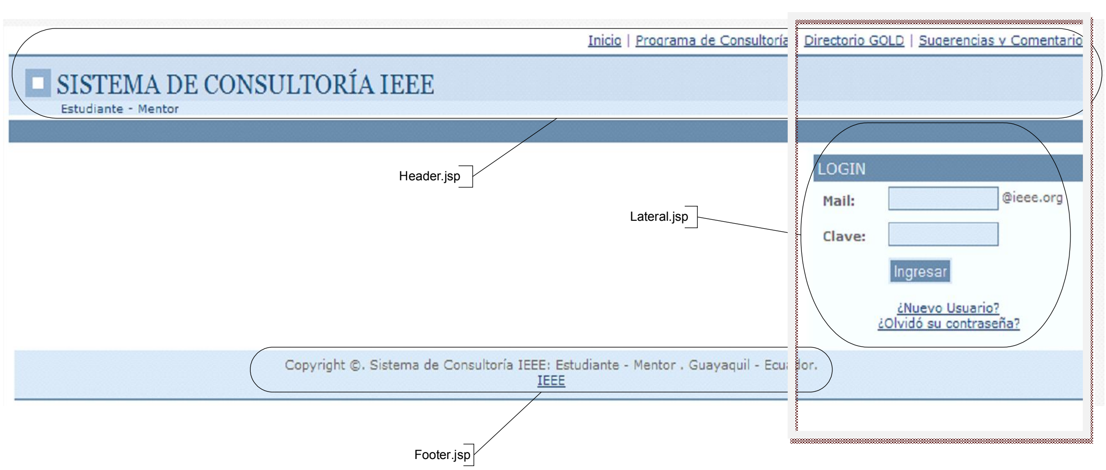

Figura 4.4 División de componentes en la vista de la página index.jsp

A continuación un fragmento de código en el que se muestra la inserción de código compuesto:

```
            <div id="cuerpo_lateral">
                <jsp:include page="../lateral.jsp" />
            </div>
        <p class="clear"> </p>
            <div id="cuerpo_pie">
                <jsp:include page="../pie.jsp" />
            </div>
        </div>
    </div>
        <div id="pie">
            <jsp:include page="../standares.jsp" />
        </div>
```

Figura 4.5 Código para componer páginas dentro de otras

Como podemos notar a través de la sentencia *include*, se invoca a cada una de las partes que conforman la página principal.

Este esquema facilita el mantenimiento de las páginas en caso de que se modifique una parte de éstas, es así como si se modifica una parte, automáticamente todas las otras páginas que la incluyan se verán modificadas.

Modelo Vista Controlador(MVC)

Model View Controller (MVC) es uno de los modelos más recomendados para el diseño de aplicaciones interactivas. Organiza los objetos en una de las tres categorías: modelos para mantenimiento de los datos, vistas para mostrar todo o una porción de los datos y controlador para manejar los eventos que afectan las vistas o el modelo [9].

El modelo representa a los datos y las reglas para acceder y actualizar estos datos. En el software empresarial, un modelo sirve como una aproximación del software al proceso real del negocio. La vista hacen ver el contenido del un modelo. Esta específica exactamente cómo le modelo debería ser presentado. Si el modelo de datos cambia, la vista debe actualizar su presentación ocmo sea necesario. El controlador traduce la interacción de los usuarios con las vistas en acciones que el modelo debe manejar [9].

MVC separa detalles del diseño (persistencia, presentación, y control de los datos), disminuye la duplicación de código, centraliza el control de la aplicación y hace que los cambios o actualizaciones sean más fácilmente manejables.

Un diseño MVC puede centralizar el control de funcionalidades como el uso de seguridad, de sesión, y el flujo de la pantalla. MVC puede adaptarse a nuevas fuentes de datos, creando código que adapta la nueva fuente con las pantallas. Finalmente, MVC define claramente las responsabilidades de las clases que participan.

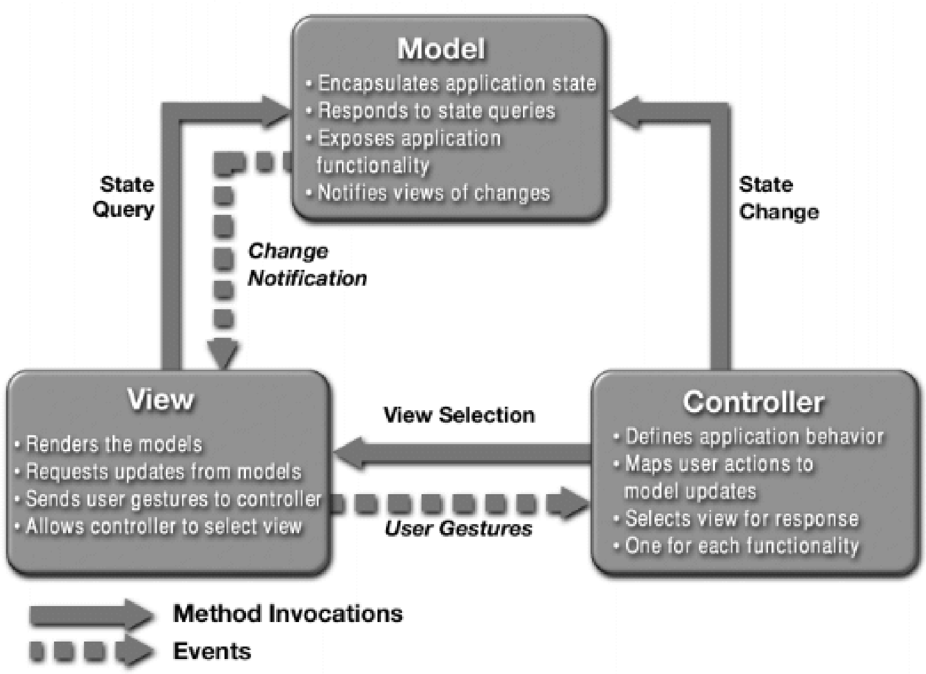

Figura 4.6 Implementación común de un modelo MVC

En la siguiente figura 4.9 se muestra con mayor detalle la interacción entre los componente del esquema MVC.

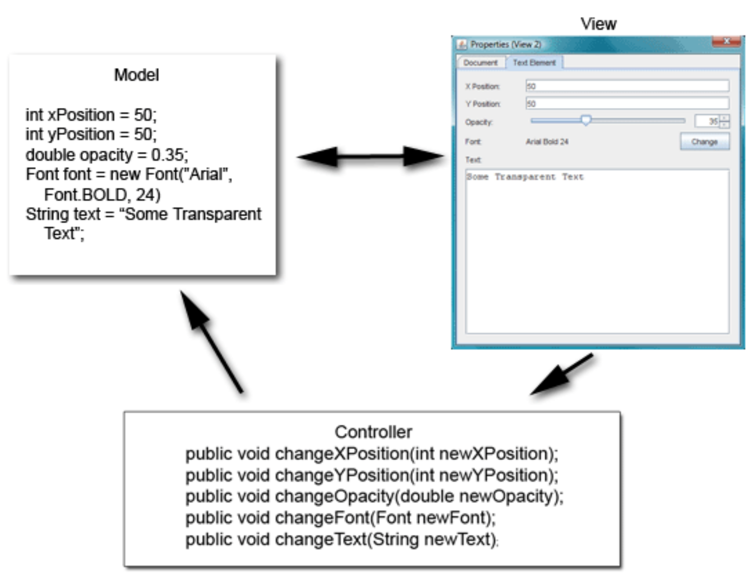

Figura 4.7 Aplicación en Java utilizando MVC

4.2.2. Modelo de persistencia de datos

El modelo utilizado para la persistencia de datos es Hibernate.

Hibernate es un poderoso objeto de alto rendimiento / persistencia relacional y servicio de consulta. Hibernate permite desarrollar clases persistentes siguiendo lenguaje orientado a objetos - incluyendo la asociación, herencia, polimorfismo, la composición y las colecciones. Hibernate permite expresar consultas en su propia extensión de SQL portátil (HQL), así como en SQL nativo, o con un API de ejemplo o criterio orientados a objetos. En las siguientes imágenes podemos comparar el modo de conexión usado por estas 3 opciones y la diferencia entre los mismos:

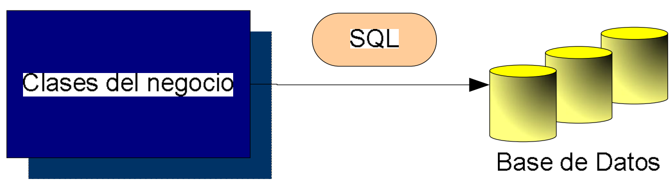

Figura 4.8 Sentencias SQL directas

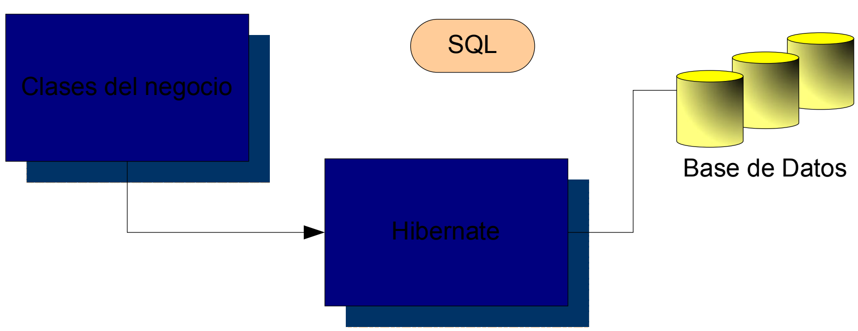

Figura 4.9 Capa de persistencia Hibernate

En la figura 4.8 podemos notar que puede ser útil para proyectos o arquitecturas sin casi clases de negocio, ya que el mantenimiento del código está altamente ligado a los cambios en el modelo de datos relacional de la base de datos, un mínimo cambio implica la revisión de casi todo el código así como su compilación y nueva instalación en el cliente.

En la figura 4.9 se agregó la capa de persistencia Hibernate, con esta capa se consigue que los desarrolladores no necesiten conocer nada acerca del esquema utilizado en la base de datos. Tan solo conocerán la interfaz proporcionada por el motor de persistencia. De esta manera conseguimos separar de manera clara y definida, la lógica de negocios de la aplicación con el diseño de la base de datos.

Esta arquitectura conllevará un proceso de desarrollo más costoso pero una vez se encuentre implementada las ventajas que conlleva merecerán la pena. Es en este punto donde entra en juego Hibernate. Como capa de persistencia desarrollada tan solo tenemos que adaptarla a nuestra arquitectura.

4.3. Diseño de la base de datos

Se diseño la base de datos tratando de cumplir con normalización para distribución de los objetos de tal manera que no se sobrecarguen las tablas, y a la vez que las consultas no tengan caminos tan largos.

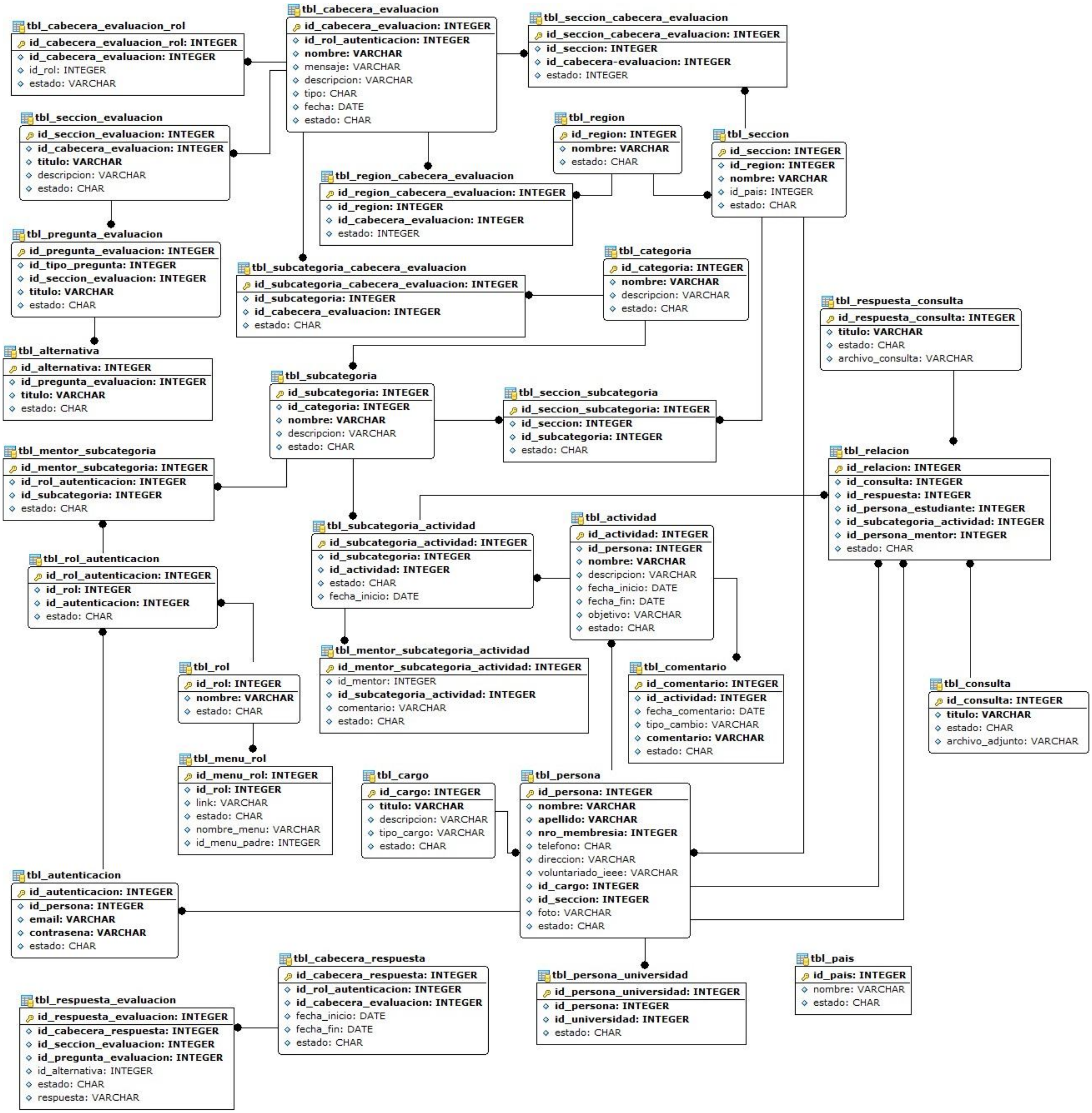

Figura 4.10 Modelo Relacional de la Base de datos

CAPITULO 5

5. IMPLEMENTACIÓN Y PRUEBAS

5.1. Preparación del entorno de desarrollo

Para preparar el entorno de desarrollo se debe tener identificadas algunas características técnicas de la solución.

- La aplicación está diseñada bajo la Arquitectura J2EE
- El motor de base de datos es MySql
- La solución propuesta está basada en el desarrollo de una aplicación web utilizando el modelo MVC de los patrones identificados como mejores prácticas en el desarrollo de software empresarial.

Bajo estas premisas dentro del entorno de desarrollo debemos considerar la instalación y tener a la mano las siguientes herramientas:

- Apache-Tomcat 6.0.14
- Jdk 1.6.0_03
- Eclipse 3.3.1
- MyEclipse 6.0.1
- MySQL Server 5.0
- MySQL Manager EMS 3.0

Adicionalmente debemos tener el último respaldo del sistema y la base de datos; para esto nos referimos al respaldo de la base YYY y al proyecto TTT.

Para instalar las herramientas considerar primero el JDK, luego copiar la carpeta eclipse en la unidad C: en Windows e instalar el MyEclipse. Luego se instala el Servidor Apache-Toncat. Finalmente se debe instalar MySQL Server 5.0 y MySQL Manager EMS.

Una vez instaladas las herramientas procedemos a ejecutar el EMS para administrar la base de datos y MyEclipse para agregar el proyecto.

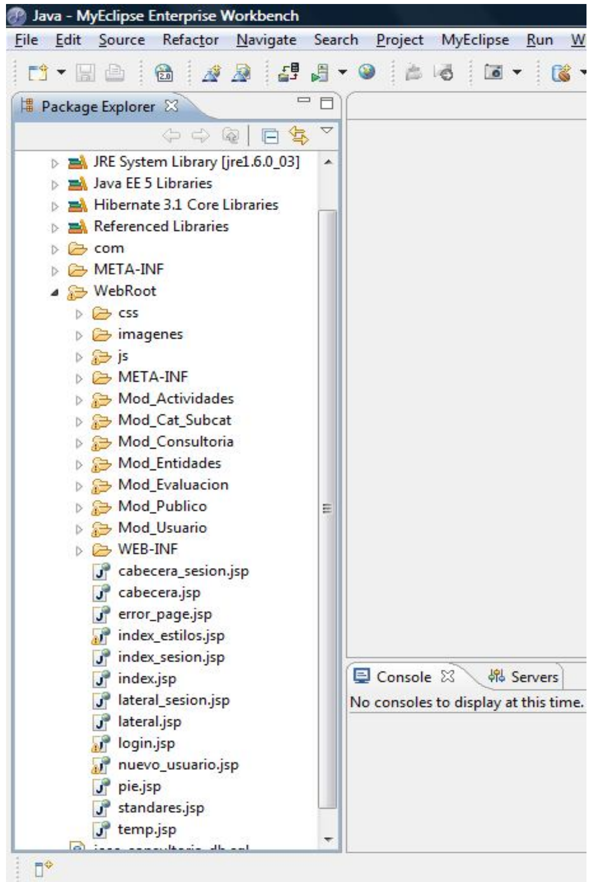

Figura 5.1 Pantalla que muestra el proyecto importado en Eclipse

Otro punto importante de revisar es la configuración del Servidor en el Eclipse para poder ejecutar el proyecto.

En el menú MyEclipse, escogemos la opción *Preferences* y se nos mostrará una pantalla donde seleccionamos la opción de Servers, vamos a Tomcat y escogemos Tomcat 6.x, en esta pantalla registraremos la ruta donde instalamos el servidor Apache-Tomcat.

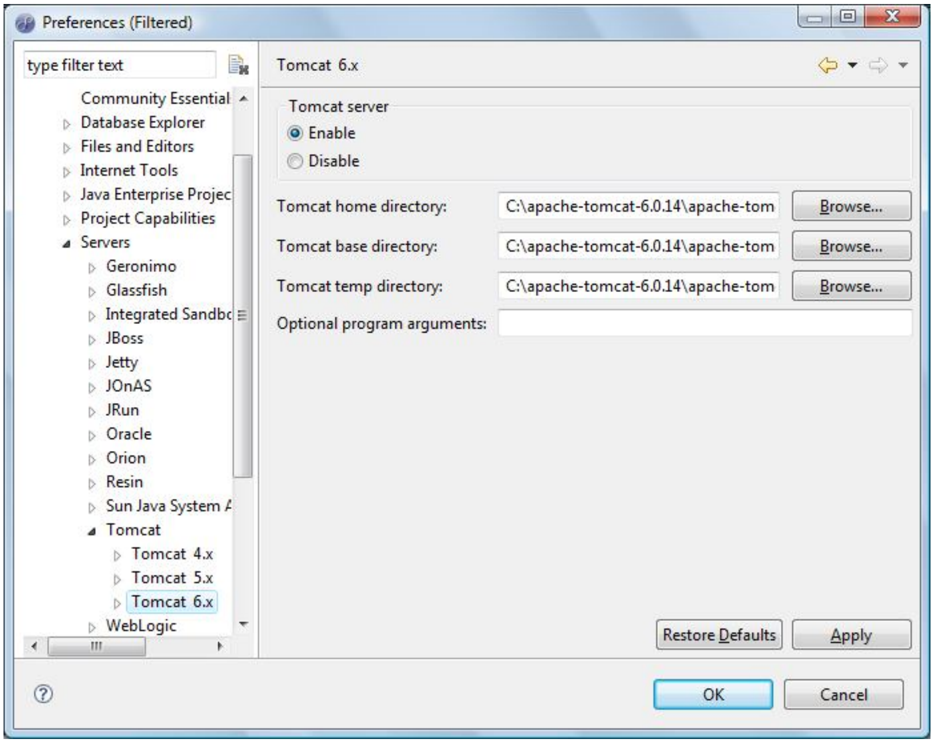

Figura 5.2 Pantalla que muestra la configuración del servidor Apache-Tomcat

Adicionalmente, esta misma pantalla, seleccionamos el servidor Tomcat y dentro de este seleccionamos la opción JDK para configurar la instancia del JDK

Para configurar el JDK damos clic en el botón Add y registramos el JDK. Finalmente todo queda listo para empezar a probar el proyecto y hacerle modificaciones.

Puntos importantes de recordar:

- Al instalar el JDK preferible configurar las variables de entorno, sobre todo JAVA_HOME

- Todo lo mencionado se realiza si es la primera vez que se están configurando la máquina para desarrollo, caso contrario la siguiente vez solo se abre el Eclipse y se empieza a modificar y probar.

5.2. Proceso de implementación de los módulos del sistema

El sistema fue dividido en los siguientes módulos para su implementación:

- Módulo de Administración
- Módulo de consultorías
- Módulo de Evaluaciones

Como el esquema a utilizar es el Modelo-Vista-Controlador, este serviría de guía para la implementación de cada módulo, considerando aquellos objetos que son parte del modelo, de las vistas y controladores de peticiones. La implementación resultaría más manejable puesto que se sigue el patrón seleccionado para su diseño.

Luego solo había que identificar aquellas funcionalidades que integraban los módulos para finalmente tener el sistema completo.

A continuación se muestra el proceso por cada módulo y un detalle de su implementación:

MÓDULO DE ACTIVIDADES

Éste modulo puede ser accedido por el estudiante o por el administrador de Sección dependiendo de la opción que cada uno vaya a ejecutar, los menús se cargarán dependiendo del tipo de rol que tenga el usuario que está iniciando sesión.

El estudiante es la única persona que puede crear Actividades, luego crear la actividad, éste debe escoger las consultorías en las que necesita asesoría, esto es, seleccionar categorías y Subcategorías. Estas actividades y consultorías quedan en un estado pendiente hasta aprobación del Administrador de Sección, a quien le llega un correo informándole de la nueva creación. Al crearlas, automáticamente el sistema propone un mentor para cada subcategoría, en caso de que no hubiese, el sistema envía un correo al Administrador de Sección para su conocimiento.

MODULO DE CONSULTORIAS

Una vez creada y aprobada la actividad, se puede dar inicio al intercambio de preguntas y respuestas entre el mentor y el estudiante.

El esquema general de todo el proceso de consultoría se muestra en el siguiente diagrama de procesos

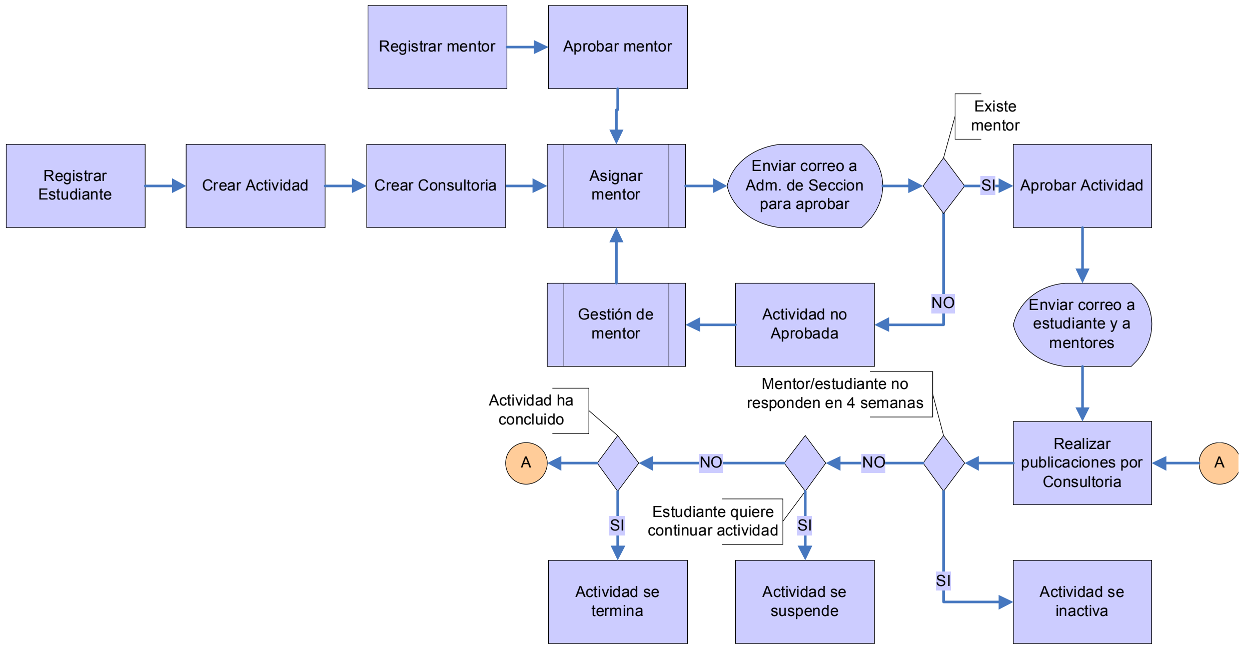

Fig. 5.5 Diagrama de procesos de Consultoría

En la implementación de la consultoría interviene el módulo de actividad, pues si no existe una actividad creada, no se pueden realizar consultorías al respecto.

La secuencia que se usó para crear esta entidad, al igual que todas es la que se describe a continuación:

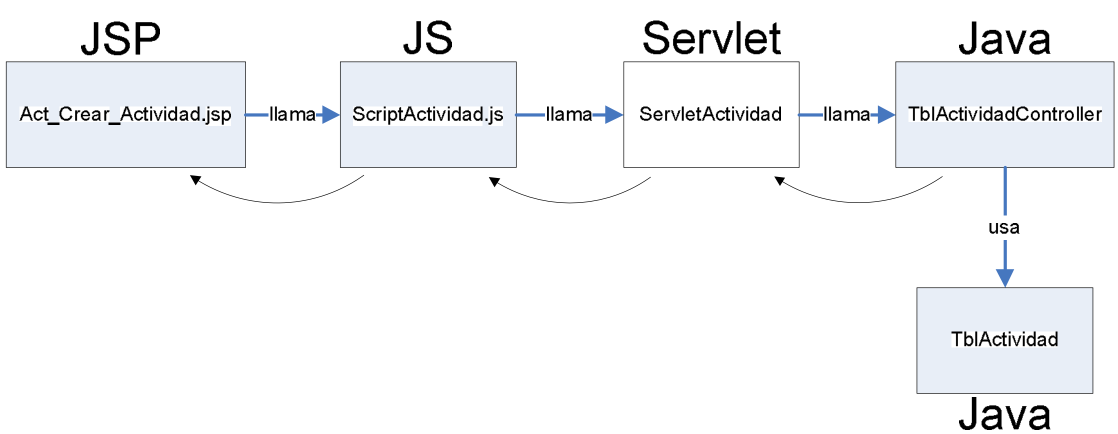

Fig. 5.6 Diagrama de secuencia de creación de Actividad

Siguiendo los lineamientos de la metodología orientada a objetos de desarrollo en capas, la página JSP contendrá sólo el front-end de la creación del objeto Actividad, aquí no hay código que se comunique directamente con la clase ni con la base de datos. Ésta página llama al script que leerá la información ingresada usando Prototype y a la vez hará la conexión con el servlet. La función del ScriptActividad.js sería la siguiente:

```
function ingresarActividad(contextPath)
{
      //alert("Entro al ScripActividad - Funcion: ingresarActividad")
      if($F("txt_fecha_fin") >= $F("txt_fecha_inicio"))
      {
      new Ajax.Request(contextPath + "/servlet/ServletActividad",
      {
              method: "post",
              parameters: "?accion=ingresarActividad&nombre=" +
$F("txt_nombre") + "&descripcion=" + $F("txt_descripcion") +
"&fecha_inicio=" + $F("txt_fecha_inicio") + "&fecha_fin=" +
$F("txt_fecha_fin") + "&objetivo=" + $F("txt_objetivo"),
              onSuccess: function(transport)
              {
                  $("frm_actividad").reset();
                  alert("La Actividad ha sido ingresada con éxito.
Favor a continuación ingrese la o las Consultorías relacionadas a su
Actividad");
                        var id_actividad =
transport.responseXML.getElementsByTagName("id")[0];
                        window.location = contextPath +
"/servlet/ServletActividad?accion=ingresarActividadConsultorias&id_act
ividad=" + id_actividad.firstChild.nodeValue;

              }
      });
      }
      else
      {
            alert("La Fecha Fin debe ser mayor o igual a la Fecha
Inicio de la Actividad. Favor confirmar las fechas ingresadas.");
            $("txt_fecha_inicio").clear();
            $("txt_fecha_fin").clear();
      }
      }
```

Prototype es un framework escrito en JavaScript que se orienta al desarrollo sencillo y dinámico de aplicaciones web. Es una herramienta que implementa las técnicas AJAX. [12]. Para usarlo se debe bajar la librería e incluir en las páginas JSP la línea *<script type="text/JavaScript" src="path/to/prototype.js"></script>.*

En el código anterior para la creación de la actividad, recibimos las variables de la página en JSP con el comando "$F(nombrevariable)", $F() es una función que recibe el ID de un objeto de la página y devuelve su valor. También tenemos la función *getElementByTagName* que recibe una clase como parámetro y devuelve un vector con los elementos que tienen como atributo *className* de la clase. Desde aquí también se puede manipular los objetos de la página html, limpiando formularios, creando objetos como tablas, listas, etc.

Cuando la transacción pasa al ServletActividad.java, invoca a una acción que trabaja como cualquier otro Servlet, recibiendo los parámetros con *request.getParameter.* Luego éste realiza la invocación al *ActividadController* para realizar la consulta usando el modelo de persistencia de Hibernate, el cual quedaría asi:

```
public Integer InsertarActividad(TblActividad actividad)
	{
		Transaction transaction = null;

		try
		{
			System.out.println("Entró al Controller
TblActividadController - InsertarActividad");
			Session session =
HibernateSessionFactory.getSession();
			transaction = session.beginTransaction();
			Integer id_actividad =
(Integer)session.save(actividad);
			//session.save(actividad);
			transaction.commit();
			session.close();
			System.out.println("Salió al Controller
TblActividadController - InsertarActividad");
			return  id_actividad;
		}

		catch (Exception e)
		{
			e.printStackTrace();
```

```
            if (transaction!=null)
            {
                  transaction.rollback();
            }
            return null;
      }
}
```

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

Hibernate es software libre, distribuido bajo los términos de la licencia GNU LGPL. Hibernate ofrece también un lenguaje de consulta de datos llamado HQL (Hibernate Query Language), [13]

En el controlador usamos las sentencias HQL para armar las consultas, es un lenguaje muy parecido a SQL, incluso con sentencias parecidas.

MÓDULO DE EVALUACIONES

Este módulo corresponde exclusivamente al Administrador de Región, quien será el único que pueda crear Evaluaciones de cualquier tipo.

A continuación el diagrama de procesos para la creación de las evaluaciones:

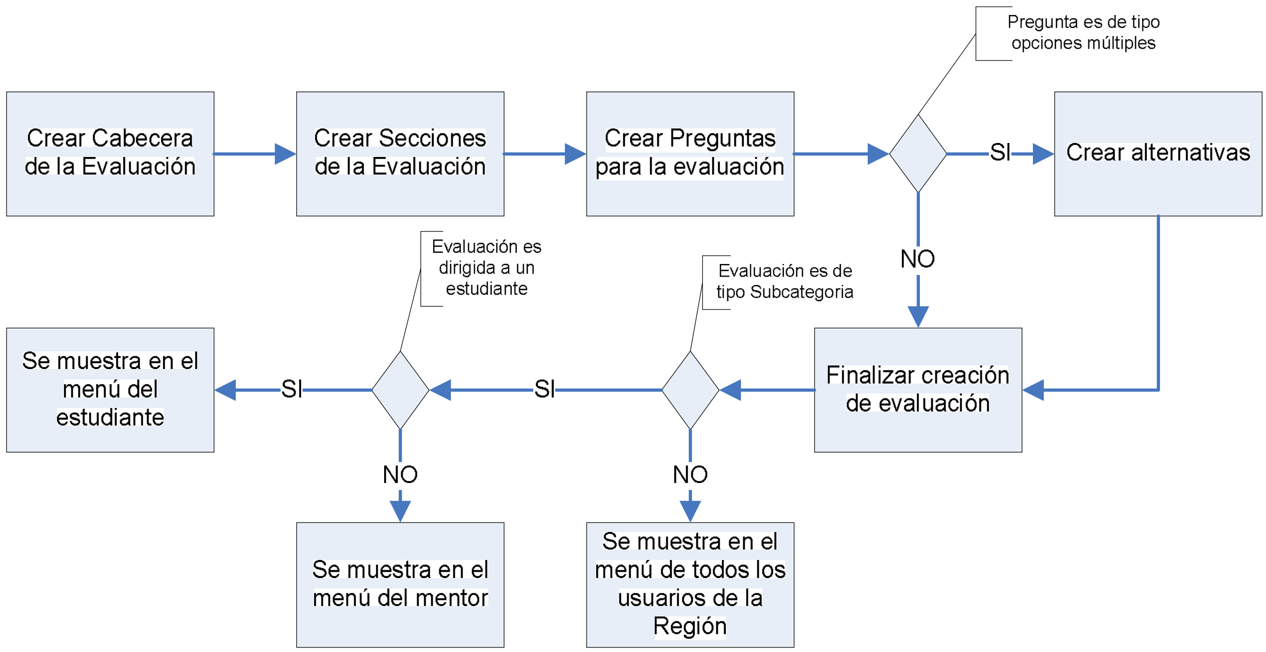

Figura 5.7 Diagrama de procesos de módulo Evaluación

El proceso describe el orden en el que se deben realizar las creaciones de los objetos para llegar a obtener una evaluación completa. Una vez creados, los usuarios a los que va dirigida pueden llenar estas encuestas.

MÓDULO DE ADMINISTRACIÓN

En el módulo de administración básicamente integramos las entidades básicas que deben crearse para la administración del sistema, esto es:

- Cargos IEEE
- Secciones IEEE
- Regiones IEEE
- Universidades IEEE
- Usuarios del sistema

- Categorías y Subcategorías

La implementación de estas entidades se realizó bajo la misma lógica de la entidad *Actividad,* por lo que podemos hacer referencia a ésta.

5.3. Plan de pruebas

Para desarrollar las pruebas del sistema se utilizarán la clasificación de pruebas por defecto y entre ellas las pruebas de caja negra.

Las pruebas funcionales o de caja negra son un enfoque para llevar a cabo pruebas donde estas se derivan de la especificación del programa o componente. El sistema es una “caja negra” cuyo comportamiento sólo se puede determinar estudiando las entradas y salidas relacionadas. Otro nombre para estas es pruebas funcionales debido a que el probador solo le interesa la funcionalidad y no la implementación del software [10].

Para realizar las pruebas utilizaremos la lógica del siguiente gráfico:

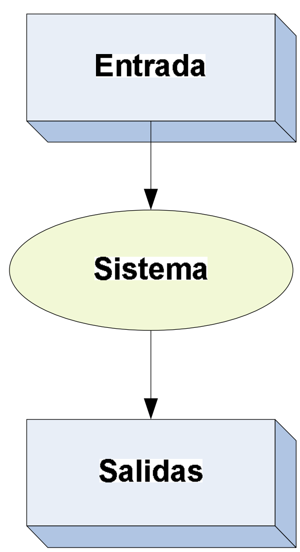

Figura 5.8 Flujo de pruebas de funcionalidades más representativas

Para la ejecución de las pruebas se utilizaron entradas más representativas, las cuales listamos a continuación:

a) Responder Evaluación
b) Crear consultoría
c) Realizar consulta dentro de consultoría
d) Responder consulta dentro de consultoría
e) Responder Evaluación
f) Estudiante crea nueva actividad
g) Cambiar estudiante responsable de actividad
h) Consultar directorio GOLD-Región
i) Administrador de Región aprueba/rechaza usuario Administrador de Sección
j) Consultar Consultorías anteriores de toda la Región

El sistema debe responder a las entradas de la siguiente forma:

a) Debe mostrar un mensaje que la evaluación fue ingresada exitosamente, que los datos fueron almacenados en la base de datos
b) El sistema debe mostrar los datos de la consultoría creada
c) El sistema debe mostrar la pregunta que haga el estudiante en un actividad al mentor, relacionada a una sub-categoría.
d) El sistema debe mostrar la respuesta que un mentor haga a una consulta de un estudiante en una sub-categoría.

e) El sistema debe mostrar la evaluación que se respondió en determinada actividad, indicando previamente que los datos se ingresaron correctamente.
f) El sistema debe mostrar la nueva actividad creada por el estudiante e indicar con un mensaje que todo se realizó correctamente.
g) El sistema debe mostrar los datos del nuevo estudiante que lidera el desarrollo de una actividad
h) El sistema debe mostrar el directorio completo de Coordinadores GOLD de todas las Secciones de la Región 9 del IEEE.
i) El sistema debe mostrar que un administrador de Sección ha sido aceptado como usuario con privilegios administrativos en el sistema
j) El sistema debe mostrar un listado de las consultorías anteriores relacionadas a la que actualmente se está revisando en una actividad.

5.4. Análisis de resultados

Para realizar las pruebas se pidió la colaboración de voluntarios IEEE tener los roles de administrador GOLD y estudiantes. Se definió que se utilizaría 5 actividades para desarrollar las pruebas, cada actividad con 3 categorías asociadas u cada categoría con un mentor.

Para pruebas a, c, d, e, g, h, i, j se logró el 100% de efectividad entre lo que se esperaba que ocurriera y lo que realmente ocurrió.

Para las pruebas b y f, el sistema respondió como se esperaba, solamente que no se consideraron para los estudiantes pre-condiciones en el uso de estas funcionalidades, puesto que para crear una consultoría o actividad, el estudiante debió haber finalizado su actividad anterior para que el sistema le permita utilizar estas funcionalidades exitosamente,

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

Luego del desarrollo de esta tesis, contamos con una aplicación web disponible para los miembros IEEE que administra la comunicación y asesoría entre un miembro estudiantil y un mentor (Miembro GOLD) del IEEE. A partir de esto podemos concluir que:

1. Existe mayor beneficio al desarrollar una aplicación distribuida en capas usando patrones de diseño de J2EE, pues facilita el mantenimiento de la aplicación.

2. Se utilizó un ramework de persistencia que mejoró la comunicación con la base de datos, pensamos que las aplicaciones que en su código utilizan comunicación directa con el motor de base de datos pueden

llegar a ser más lentas que aquellas que usan una capa intermedia que realice este trabajo.

3. Al implementar el módulo de consultorías, se logró automatizar el almacenamiento de la comunicación entre 2 miembros IEEE en calidad de mentoría, que son tan necesarios en la comunidad Latinoamericana del IEEE.

RECOMENDACIONES

1. Se recomienda aprovechar el sistema para mantener una comunicación interactiva entre un estudiante y un mentor para obtener el conocimiento necesario para realizar eventos exitosos.
2. Es recomendable establecer métricas para medir la calidad del producto de software que se está desarrollando.
3. Se recomienda que si se va a añadir funcionalidad al portal se considere seguir utilizando el uso de patrones de la arquitectura J2EE.
4. Recomendamos aumentar funcionalidad a la aplicación, brindando otros servicios a los miembros estudiantiles tales como base de conocimiento, suscripciones RSS, etc.
5. Recomendamos realizar actualizaciones de las herramientas en el servidor frecuentemente con nuevos parches que brinden más seguridad a la información que se está almacenando.

6. Implementar una capa adicional a las vistas y al modelo para tener el sistema en varios idiomas, actualmente solo soporta el español porque será implementado para Latinoamérica.

7. Diseñar un módulo de datamining para analizar el liderazgo e todos los proyectos tanto de estudiantes como profesionales

ANEXOS

ANEXO A

DETALLE DE CASOS DE USO DE LA APLICACIÓN

A continuación se presenta el detalle de cada uno de los casos de uso planteados, en un formato formal con algunos campos de interés para el proyecto.

No existe una plantilla estándar para documentar los detalles de casos de uso. Sin embargo, existe un acuerdo en cuanto a las secciones fundamentales que deben ser consideradas en un formato detallado, el cual intentamos utilizar en las descripciones que se muestran a continuación. [11]

ID Caso de Uso	**CU-001**
Caso de Uso:	**Iniciar sesión**
Descripción:	El usuario requiere iniciar su sesión para acceder a las distintas opciones que el sistema le ofrece.
Actores:	Miembro estudiantil, Mentor, Administradores y

	revisores

Precondiciones:
- Haberse suscrito en el sistema. (Caso de uso CU-057)

Escenario principal de éxito (Flujo normal):
1. El miembro estudiantil, mentor, administrador o revisor ingresa su nombre de usuario (correo electrónico) y su contraseña
2. El miembro estudiantil, mentor, administrador o revisor elige la opción “Ingresar”.
3. El sistema verifica que el usuario existe en la base de datos y que los datos ingresados son correctos.
4. El sistema muestra las diferentes opciones a las que tiene acceso el usuario ya sea como miembro estudiantil, mentor, administrador o revisor.

Extensiones (o Flujo Alternativo):

4a. El miembro estudiantil, mentor, administrador o revisor ingresa datos erróneos o incompletos
1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):
- Ingresar al sistema con acceso a ciertas opciones que dependen de el rol que tiene dentro del sistema.

ID Caso de Uso	**CU-002**
Caso de Uso:	**Cerrar sesión**
Descripción:	El usuario requiere cerrar su sesión para salir de las opciones que el sistema le ofrece
Actores:	Miembro estudiantil, Mentor, Administradores y revisores

Precondiciones:
- Haber iniciado sesión en el portal. (Caso de uso CU-001)

Escenario principal de éxito (Flujo normal):
1. El miembro estudiantil, mentor, administrador o revisor escoje la opción “Logout”
2. El sistema muestra la pantalla principal sin las opciones a las que tiene acceso el usuario ya sea como miembro estudiantil, mentor, administrador o revisor.

Extensiones (o Flujo Alternativo):

1a. La base de datos se encuentra abajo.
1. El sistema presenta un aviso de error

Poscondiciones (Garantías de éxito):
- Salir del sistema con éxito.

ID Caso de Uso	**CU-003**
Caso de Uso:	**Administrador de Región crea una evaluación**
Descripción:	El usuario requiere crear una encuesta para evaluar las consultorías del sistema.
Actores:	Administrador de Región

Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber escogido la opción Evaluación del menú general ▪ Si la evaluación a crear es de tipo Subcategoría, debe haber creado una subcategoría (Caso de uso CU-015) **Escenario principal de éxito (Flujo normal):** **1.** El usuario **2.** revisor. **Extensiones (o Flujo Alternativo):** **4a.** El miembro estudiantil, mentor, administrador o revisor ingresa datos erróneos o incompletos **3.** El sistema presenta un aviso indicando que los datos ingresados son inválidos **4.** Volver al punto 1 del flujo normal **Poscondiciones (Garantías de éxito):** ▪ Ingresar al sistema con acceso a ciertas opciones que dependen de el rol que tiene dentro del sistema.

ID Caso de Uso	**CU-004**
Caso de Uso:	**Administrador de Región crea una sección de una evaluación**
Descripción:	El usuario requiere crear una sección de una encuesta para evaluar las consultorías del sistema.
Actores:	Administrador de Región
Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber escogido la opción Evaluaciones del menú general ▪ Haber creado una cabecera de evaluación (Caso de uso CU-003) **Escenario principal de éxito (Flujo normal):** **1.** El usuario escoge la opción "Crear nueva evaluación" **2.** El usuario ingresa los datos de la cabecera de la evaluación exitosamente. **3.** El sistema presenta los campos: nombre y descripción. **4.** El usuario ingresa la información requerida. **5.** El usuario escoge la opción para *Guardar y continuar*. **6.** El sistema verifica la información registrada. **7.** El sistema crea la nueva sección y muestra un mensaje que la sección se creó con éxito. **Extensiones (o Flujo Alternativo):** **2.a** No se creó la cabecera de la evaluación **1.** Referirse al CU-003 **2.** Volver al punto 1 del flujo normal **4.a.** La información solicitada está incorrecta o incompleta. **1.** El sistema muestra un mensaje de error. **2.** Volver al punto 1 del flujo normal **Pos-condiciones (Garantías de éxito):** ▪ Ingresar una sección con éxito, el sistema mostrará la lista de las secciones creadas exitosamente.	

ID Caso de Uso	**CU-005**
Caso de Uso:	**Administrador de Región crea una pregunta de una sección de evaluación**
Descripción:	El usuario requiere crear preguntas dentro de una sección de una encuesta para evaluar las consultorías del sistema.
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción Evaluaciones del menú general
- Haber creado una sección de evaluación (Caso de uso CU-006)

Escenario principal de éxito (Flujo normal):

1. El usuario escoge la opción "Crear nueva evaluación"
2. El usuario ingresa los datos de la cabecera de la evaluación exitosamente.
3. El usuario ingresa los datos de la sección de la evaluación exitosamente.
4. El sistema muestra la lista de secciones de la evaluación
5. El usuario escoge la opción *Preguntas*
6. El sistema presenta los campos: Titulo y Tipo.
7. El usuario ingresa la información requerida.
    a. Si el usuario escoge el tipo "Múltiple" o "Simple", el sistema muestra los campos *Alternativa 1, Alternativa 2, Alternativa 3, Alternativa 4, Alternativa 5*.
    b. El usuario ingresa las alternativas de la pregunta
8. El usuario escoge la opción para *Ingresar*.
9. El sistema verifica la información registrada.
10. El sistema crea la nueva pregunta y muestra un mensaje que la pregunta se creó con éxito.

Extensiones (o Flujo Alternativo):

7.a La información solicitada está incorrecta o incompleta.

1. El sistema muestra un mensaje de error.
2. Volver al punto 1 del flujo normal

7.b.a El usuario ha ingresado sólo 1 alternativa y se requieren mínimo 2

1. El sistema muestra un mensaje de error.
2. Volver al punto 1 del flujo normal

Pos-condiciones (Garantías de éxito):

- Ingresar una pregunta con éxito, el sistema mostrará la lista de las preguntas creadas exitosamente.

ID Caso de Uso	**CU-006**
Caso de Uso:	**Administrador de Región crea una alternativa de una pregunta de evaluación**
Descripción:	El usuario requiere crear alternativas dentro de una pregunta de una encuesta para evaluar las consultorías del sistema, siempre que ésta sea multiple.
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)

▪ Haber escogido la opción Evaluaciones del menú general ▪ Haber creado una pregunta de evaluación (Caso de uso CU-004) **Escenario principal de éxito (Flujo normal):** **11.** El usuario escoge la opción “Crear nueva evaluación” **12.** El usuario ingresa los datos de la cabecera de la evaluación exitosamente. **13.** El usuario ingresa los datos de la sección de la evaluación exitosamente. **14.** El sistema muestra la lista de secciones de la evaluación **15.** El usuario escoge la opción *Preguntas* **16.** El usuario ingresa una pregunta de tipo “múltiple” o “abierta” **17.** El sistema muestra los campos Alternativa 1,2,3,4,5 **18.** El usuario ingresa al menos 2 alternativas. **19.** El usuario escoge la opción para *Ingresar*. **20.** El sistema verifica la información registrada. **21.** El sistema crea la nueva pregunta y muestra un mensaje que la pregunta se creó con éxito. **Extensiones (o Flujo Alternativo):** **7.a** La información solicitada está incorrecta o incompleta. **3.** El sistema muestra un mensaje de error. **4.** Volver al punto 1 del flujo normal **7.b.a** El usuario ha ingresado sólo 1 alternativa y se requieren mínimo 2 **3.** El sistema muestra un mensaje de error. **4.** Volver al punto 1 del flujo normal **Pos-condiciones (Garantías de éxito):** ▪ Ingresar una pregunta con éxito, el sistema mostrará la lista de las preguntas creadas exitosamente.

ID Caso de Uso	**CU-007**
Caso de Uso:	**Administrador de Región modifica una evaluación**
Descripción:	El usuario requiere modificar una encuesta para evaluar las consultorías del sistema.
Actores:	Administrador de Región
Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber escogido la opción *Evaluaciones* del menú general ▪ Haber creado al menos una evaluación (Caso de uso CU-004) **Escenario principal de éxito (Flujo normal):** **1.** El sistema muestra la lista de las evaluaciones creadas por ese usuario. **2.** El usuario escoge la opción *Ver Detalle* **3.** El sistema muestra la evaluación completa. **4.** El usuario escoge la Opción *Modificar* **5.** El sistema presenta los campos: titulo, mensaje, descripción y tipo con la información que había sido ingresada. **6.** El usuario modifica la información requerida. **7.** El usuario escoge la opción para guardar. **8.** El sistema verifica la información registrada. **9.** El sistema modifica la evaluación y muestra un mensaje confirmando	

que la evaluación fue modificada sin problemas.

10. El usuario escoge *Ok*

Extensiones (o Flujo Alternativo):

6.a. La información solicitada está incorrecta o incompleta. El sistema muestra un mensaje de error.

Pos-condiciones (Garantías de éxito):

- El sistema muestra la evaluación completa con los cambios hechos.

ID Caso de Uso	**CU-008**
Caso de Uso:	**Administrador de Región crea categoría**
Descripción:	El usuario requiere crear una categoría para clasificar las actividades y consultorías
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Categorias y SubCat* del menú general

Escenario principal de éxito (Flujo normal):

1. El usuario escoge la opción *Nueva Categoría*
2. El sistema presenta los campos: nombre y descripción.
3. El usuario ingresa la información requerida.
4. El usuario escoge la opción para *Guardar.*
5. El sistema verifica la información registrada.
6. El sistema crea la nueva categoría y muestra un mensaje que la categoría se creó con éxito

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Ingresar una categoría exitosamente.

ID Caso de Uso	**CU-009**
Caso de Uso:	**Administrador de Región modifica categoría**
Descripción:	El usuario requiere modificar una categoría
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Categorias y SubCat* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema presenta la lista de categorías que pertenecen a esa Región
2. El usuario escoge la opción *Modificar.*
3. sistema presenta los campos: categoría, descripción y estado.
4. El usuario modifica la información requerida.
5. El usuario escoge la opción para *Actualizar.*
6. El sistema verifica la información registrada.
7. El sistema modifica la categoría y muestra un mensaje que la

categoría se modificó con éxito

Extensiones (o Flujo Alternativo):

4.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Modificar una categoría exitosamente. El sistema muestra una lista de las categorías de la Región del usuario, con los cambios realizados.

ID Caso de Uso	**CU-010**
Caso de Uso:	**Administrador de Región crea subcategoría**
Descripción:	El usuario requiere crear una subcategoría para clasificar las actividades y consultorías
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Categorias y SubCat* del menú general

Escenario principal de éxito (Flujo normal):

7. El usuario escoge la opción *Nueva Subcategoría*
8. El sistema presenta los campos: Categoría, Subcategoría y descripción.
9. El usuario ingresa la información requerida.
10. El usuario escoge la opción para *Guardar.*
11. El sistema verifica la información registrada.
12. El sistema crea la nueva subcategoría y muestra un mensaje que la subcategoría se creó con éxito

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Ingresar una subcategoría exitosamente.

ID Caso de Uso	**CU-011**
Caso de Uso:	**Administrador de Región modifica subcategoría**
Descripción:	El usuario requiere modificar una subcategoría
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Categorias y SubCat* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema presenta la lista de subcategorías que pertenecen a esa Región
2. El usuario escoge la opción *Modificar.*
3. sistema presenta los campos: subcategoría, descripción y estado.

4. El usuario modifica la información requerida.
5. El usuario escoge la opción para *Actualizar.*
6. El sistema verifica la información registrada.
7. El sistema modifica la subcategoría y muestra un mensaje que la subcategoría se modificó con éxito

Extensiones (o Flujo Alternativo):

4.a. El usuario ingresa datos erróneos o incompletos

3. El sistema presenta un aviso indicando que los datos ingresados son inválidos
4. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Modificar una subcategoría exitosamente. El sistema muestra una lista de las subcategorías de la Región del usuario, con los cambios realizados.

ID Caso de Uso	**CU-012**
Caso de Uso:	**Administrador de Sección asigna categorías o Subcategorías a su sección**
Descripción:	El usuario requiere modificar una categoría
Actores:	Administrador de Sección

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Categorias y SubCat* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema presenta la lista de categorías y subcategorías activas para la Región a la que pertenece la Sección del usuario. Las subcategorías activas de la sección aparecen marcadas.
2. El usuario selecciona o selecciona las Subcategorías que desea para su Sección.
3. El usuario escoge la opción *Ingresar.*
4. El sistema verifica la información registrada.
5. El sistema modifica las Subcategorías disponibles para la Sección de usuario y muestra un mensaje que se realizó con éxito

Extensiones (o Flujo Alternativo):

2.a. El usuario desactiva Subcategorías que están activas para la Sección dentro de una consultoría activa

1. El sistema presenta un aviso indicando que no se puede desactivar dicha subcategoria
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Asignar subcategorías exitosamente.

ID Caso de Uso	**CU-013**
Caso de Uso:	**Recomendar nueva categoría**
Descripción:	El usuario requiere recomendar una categoría
Actores:	Mentor, Revisores, Administrador de Sección

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)

▪ Haber escogido la opción *Categorias y SubCat* del menú general **Escenario principal de éxito (Flujo normal):** **6.** El sistema presenta la lista de categorías y subcategorías activas para la Región a la que pertenece la Sección del usuario. **7.** El usuario escoge la opción *Recomendar Categoría.* **8.** El sistema muestra los campos Nombres y descripción. **9.** El usuario escoge la opción *Recomendar.* **10.** El sistema verifica la información registrada. **11.** El sistema informa el requerimiento al Administrador de Región. **Extensiones (o Flujo Alternativo):** **2.a.** El usuario desactiva Subcategorías que están activas para la Sección dentro de una consultoría activa **3.** El sistema presenta un aviso indicando que no se puede desactivar dicha subcategoria **4.** Volver al punto 1 del flujo normal **Poscondiciones (Garantías de éxito):** ▪ Asignar subcategorías exitosamente.

ID Caso de Uso	**CU-014**
Caso de Uso:	**Recomendar nueva subcategoría**
Descripción:	El usuario requiere recomendar una subcategoría
Actores:	Mentor, Revisores, Administrador de Sección
Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber escogido la opción *Categorias y SubCat* del menú general **Escenario principal de éxito (Flujo normal):** **12.** El sistema presenta la lista de categorías y subcategorías activas para la Región a la que pertenece la Sección del usuario. **13.** El usuario escoge la opción *Recomendar Categoría.* **14.** El sistema muestra los campos categoría, Nombres y descripción. **15.** El usuario escoge la opción *Recomendar.* **16.** El sistema verifica la información registrada. **17.** El sistema informa el requerimiento al Administrador de Región. **Extensiones (o Flujo Alternativo):** **2.a.** El usuario no ingresa la información completa **5.** El sistema presenta un aviso indicando que no se puede recomendar dicha subcategoria **6.** Volver al punto 1 del flujo normal **Poscondiciones (Garantías de éxito):** ▪ Asignar subcategorías exitosamente.	

ID Caso de Uso	**CU-015**
Caso de Uso:	**Administrador de Región aprueba o rechaza sugerencias de categorías**
Descripción:	El usuario requiere aprobar o rechazar sugerencia de categoría
Actores:	Administrador de Región
Precondiciones:	

▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber escogido la opción *Categorias y SubCat* del menú general **Escenario principal de éxito (Flujo normal):** **1.** El sistema presenta la lista de categorías que pertenecen a esa Región. Las sugerencias de Categorías se muestran con estado *Pendiente.* **2.** El usuario escoge la opción *Modificar.* **3.** sistema presenta los campos: categoría, descripción y estado. **4.** El usuario modifica la información requerida. **5.** El usuario escoge la opción para *Actualizar.* **6.** El sistema verifica la información registrada. **7.** El sistema modifica la categoría y muestra un mensaje que la categoría se modificó con éxito **Extensiones (o Flujo Alternativo):** **4.a.** El usuario ingresa datos erróneos o incompletos **1.** El sistema presenta un aviso indicando que los datos ingresados son inválidos **2.** Volver al punto 1 del flujo normal **Poscondiciones (Garantías de éxito):** ▪ Modificar una categoría exitosamente. El sistema muestra una lista de las categorías de la Región del usuario, con los cambios de estado realizados.

ID Caso de Uso	**CU-016**
Caso de Uso:	**Administrador de Región aprueba o rechaza sugerencias de Subcategorías**
Descripción:	El usuario requiere aprobar o rechazar sugerencia de Subcategorías
Actores:	Administrador de Región
Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber escogido la opción *Categorias y SubCat* del menú general **Escenario principal de éxito (Flujo normal):** **8.** El sistema presenta la lista de subcategorías que pertenecen a esa Región. Las sugerencias de Subcategorías se muestran con estado *Pendiente.* **9.** El usuario escoge la opción *Modificar.* **10.** El sistema presenta los campos: Categoría, Subcategoria, descripción y estado. **11.** El usuario modifica la información requerida. **12.** El usuario escoge la opción para *Actualizar.* **13.** El sistema verifica la información registrada. **14.** El sistema modifica la subcategoría y muestra un mensaje que la subcategoría se modificó con éxito **Extensiones (o Flujo Alternativo):** **4.a.** El usuario ingresa datos erróneos o incompletos **3.** El sistema presenta un aviso indicando que los datos ingresados son inválidos **4.** Volver al punto 1 del flujo normal	

Poscondiciones (Garantías de éxito): ▪ Actualizar el estado de una subcategoría exitosamente. El sistema muestra una lista de las subcategorías de la Región del usuario, con los cambios de estado realizados.

ID Caso de Uso	**CU-017**
Caso de Uso:	**Administrador crea universidades**
Descripción:	El usuario requiere crear universidades
Actores:	Administradores
Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber escogido la opción *Universidad* del menú general **Escenario principal de éxito (Flujo normal):** **1.** El usuario escoge la opción *Nueva Universidad* **2.** El sistema presenta los campos: nombre y datos de sitio web. **3.** El usuario ingresa la información requerida. **4.** El usuario escoge la opción para *Ingresar.* **5.** El sistema verifica la información registrada. **6.** El sistema crea la nueva universidad y muestra un mensaje que la universidad se creó con éxito **Extensiones (o Flujo Alternativo):** **3.a.** El usuario ingresa datos erróneos o incompletos **3.** El sistema presenta un aviso indicando que los datos ingresados son inválidos **4.** Volver al punto 1 del flujo normal **Poscondiciones (Garantías de éxito):** ▪ Ingresar una universidad exitosamente.	

ID Caso de Uso	**CU-018**
Caso de Uso:	**Administrador modifica universidades**
Descripción:	El usuario requiere modificar universidades
Actores:	Administradores
Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber escogido la opción *Universidad* del menú general **Escenario principal de éxito (Flujo normal):** **1.** El sistema presenta la lista de las universidades creadas para la sección del usuario. **2.** El usuario escoge la opción *Ver Detalle* **3.** El sistema presenta los campos: nombre y datos de sitio web. **4.** El usuario ingresa la información requerida. **5.** El usuario escoge la opción para *Actualizar.* **6.** El sistema verifica la información registrada. **7.** El sistema modifica la universidad y muestra un mensaje que la universidad se modificó con éxito **Extensiones (o Flujo Alternativo):** **3.a.** El usuario ingresa datos erróneos o incompletos **5.** El sistema presenta un aviso indicando que los datos	

ingresados son inválidos

6. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Modificar una universidad exitosamente.

ID Caso de Uso	**CU-019**
Caso de Uso:	**Administrador de Región crea cargos**
Descripción:	El usuario requiere crear cargos IEEE
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Cargos* del menú general

Escenario principal de éxito (Flujo normal):

7. El usuario escoge la opción *Nuevo Cargo*
8. El sistema presenta los campos: nombre, tipo y descripción.
9. El usuario ingresa la información requerida.
10. El usuario escoge la opción para *Ingresar.*
11. El sistema verifica la información registrada.
12. El sistema crea el nuevo cargo y muestra un mensaje que el cargo se creó con éxito

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Crear un cargo exitosamente.

ID Caso de Uso	**CU-020**
Caso de Uso:	**Administrador de Región modifica cargos IEEE**
Descripción:	El usuario requiere modificar cargos IEEE
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Cargos* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema presenta la lista de categorías que pertenecen a esa Región
2. El usuario escoge la opción *Modificar.*
3. sistema presenta los campos: categoría, descripción y estado.
4. El usuario modifica la información requerida.
5. El usuario escoge la opción para *Actualizar.*
6. El sistema verifica la información registrada.
7. El sistema modifica la categoría y muestra un mensaje que la categoría se modificó con éxito

Extensiones (o Flujo Alternativo):

4.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos

ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Modificar cargos exitosamente. El sistema muestra una lista de los cargos de la Región del usuario, con los cambios realizados.

ID Caso de Uso	CU-021
Caso de Uso:	**Administrador de Región crea Secciones**
Descripción:	El usuario requiere crear Secciones IEEE
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Secciones* del menú general

Escenario principal de éxito (Flujo normal):

1. El usuario escoge la opción *Crear Nueva Sección*
2. El sistema presenta los campos: nombre, región, país, datos de sitio web.
3. El usuario ingresa la información requerida.
4. El usuario escoge la opción para *Ingresar.*
5. El sistema verifica la información registrada.
6. El sistema crea la nueva sección y muestra un mensaje que la sección se creó con éxito

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos
3. El sistema presenta un aviso indicando que los datos ingresados son inválidos
4. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Crear una sección exitosamente.

ID Caso de Uso	CU-022
Caso de Uso:	**Administrador de Región modifica secciones IEEE**
Descripción:	El usuario requiere modificar secciones IEEE
Actores:	Administrador de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Secciones* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema presenta la lista de secciones que pertenecen a esa Región
2. El usuario escoge la opción *Ver Detalle.*
3. El sistema presenta los campos: nombre, región, país, datos de sitio web.
4. El usuario modifica la información requerida.
5. El usuario escoge la opción para *Actualizar.*
6. El sistema verifica la información registrada.
7. El sistema modifica la Sección y muestra un mensaje que la Sección se modificó con éxito

Extensiones (o Flujo Alternativo): **4.a.** El usuario ingresa datos erróneos o incompletos **3.** El sistema presenta un aviso indicando que los datos ingresados son inválidos **4.** Volver al punto 1 del flujo normal **Poscondiciones (Garantías de éxito):** ▪ Modificar secciones IEEE exitosamente. El sistema muestra una lista de las secciones de la Región del usuario, con los cambios realizados.

ID Caso de Uso	**CU-023**
Caso de Uso:	**Administrador de Región consulta respuestas a una evaluación**
Descripción:	El usuario requiere modificar consultar las respuestas a las evaluaciones que ha creado
Actores:	Administrador de Región
Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber creado al menos una evaluación (Caso de uso CU-003) ▪ Haber escogido la sección *Evaluación* del menú principal **Escenario principal de éxito (Flujo normal):** **1.** El sistema presenta la lista de evaluaciones que pertenecen a esa usuario como creador **2.** El usuario escoge la opción *Ver Respuestas* **3.** El sistema presenta una tabla con la información de las personas que respondieron la encuesta **4.** El usuario selecciona una persona **5.** El sistema muestra la encuesta totalmente respondida por ese usuario con éxito **Extensiones (o Flujo Alternativo):** **Poscondiciones (Garantías de éxito):** ▪ Mostrar la información de las respuestas a encuestas realizadas.	

ID Caso de Uso	**CU-024**
Caso de Uso:	**Responder evaluación**
Descripción:	El usuario requiere responder las evaluaciones que se le asignen
Actores:	Estudiante, mentor, administrador de sección
Precondiciones: ▪ Haber iniciado sesión en el portal. (Caso de uso CU-001) ▪ Haber creado al menos una evaluación (Caso de uso CU-003) ▪ Haber escogido la sección *Evaluación* del menú principal **Escenario principal de éxito (Flujo normal):** **1.** El sistema presenta la lista de evaluaciones que pertenecen a esa usuario comopendientes de responder. **2.** El usuario escoge la opción *Responder* **3.** El sistema la encuesta lista para ser respondida **4.** El usuario responde todas las preguntas y presiona *Aceptar*	

5. El sistema sistema verifica la información registrada.
6. El sistema muestra un mensaje que la Encuesta se respondió con éxito.

Extensiones (o Flujo Alternativo):

4.a El usuario no respondió todas las preguntas

1. El sistema presenta un mensaje indicando que faltan preguntas por responder
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Mensaje de éxito al responder la encuesta completamente.

ID Caso de Uso	**CU-025**
Caso de Uso:	**Crear Consultoría**
Descripción:	El usuario requiere crear consultorías
Actores:	Estudiante

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber creado al menos una actividad (Caso de uso CU-032)
- Haber escogido la sección Mis *Actividades* del menú principal

Escenario principal de éxito (Flujo normal):

1. El sistema presenta la lista de actividades que pertenecen a ese usuario como creador.
2. El usuario escoge la opción *Crear consultoría*
3. El sistema presenta una tabla con la información de las categorías y subcategorías que aún están disponibles.
4. El usuario selecciona una categoría
5. El sistema muestra las subcategorías de esa categoría
6. El usuario selecciona una subcategoría y escoge *Aceptar*
7. El sistema procesa la información ingresada

Extensiones (o Flujo Alternativo):

1.a El usuario no ha creado actividades

1. El sistema no muestra nada, se debe crear una actividad (CU-032)
2. Volver el punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Mensaje de éxito al crear la consultoría completa

ID Caso de Uso	**CU-026**
Caso de Uso:	**Realizar consulta dentro de consultoría**
Descripción:	El usuario requiere crear consultas en las consultorías
Actores:	Estudiante

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber creado al menos una consultoría (Caso de uso CU-025)
- Haber escogido la sección Mis *Actividades* del menú principal
- Haber escogido la sección *Consultorías Activas* del menú principal

Escenario principal de éxito (Flujo normal):

1. El sistema presenta la información de la consultoría activa

seleccionada y muestra el campo *Pregunta.*
2. El usuario ingresa la pregunta y escoge *Enviar*.
3. El sistema verifica la información ingresada.
4. El sistema muestra la pregunta ingresada

Extensiones (o Flujo Alternativo):

2.a El usuario no ha ingresa pregunta
1. El sistema no muestra nada, se debe crear una pregunta
2. Volver el punto 1 del flujo normal

Poscondiciones (Garantías de éxito):
- Mensaje de éxito al crear una consulta en la consultoria

ID Caso de Uso	**CU-027**
Caso de Uso:	**Responder consulta dentro de consultoría**
Descripción:	El usuario requiere responder consultas en las consultorías
Actores:	Mentor

Precondiciones:
- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber sido asignado a una consultoría
- Haber escogido la sección *Mis Consultorías* del menú principal

Escenario principal de éxito (Flujo normal):
1. El sistema presenta la información de las consultoría activa del usuario
2. El usuario escoge la consultoría donde quiere responder una pregunta.
3. El sistema muestra el campo *Respuesta.*
4. El usuario ingresa la respuesta y escoge *Enviar*.
5. El sistema verifica la información ingresada.
6. El sistema muestra la respuesta ingresada

Extensiones (o Flujo Alternativo):

2.a El usuario no ha ingresado respuesta
1. El sistema no muestra nada, se debe crear una respuesta
2. Volver el punto 1 del flujo normal

Poscondiciones (Garantías de éxito):
- Mensaje de éxito al crear una respuesta en la consultoría

ID Caso de Uso	**CU-028**
Caso de Uso:	**Modificar consulta dentro de consultoría**
Descripción:	El usuario requiere modificar consultas en las consultorías
Actores:	Estudiante

Precondiciones:
- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber creado al menos una consultoría (Caso de uso CU-025)
- Haber escogido la sección Mis *Actividades* del menú principal
- Haber escogido la sección *Consultorías Activas* del menú principal

Escenario principal de éxito (Flujo normal):
1. El sistema presenta la información de la consultoría activa

seleccionada y muestra el campo *Pregunta.*
2. El usuario selecciona la pregunta y escoge *Editar*.
3. El usuario muestra la pregunta que previamente se ingresó.
4. El usuario modifica la información de la pregunta
5. El sistema verifica la información ingresada.
6. El sistema muestra la pregunta modificada

Extensiones (o Flujo Alternativo):

2.a El usuario no ha ingresado pregunta
1. El sistema no muestra nada, se debe crear una pregunta
2. Volver el punto 1 del flujo normal

Poscondiciones (Garantías de éxito):
- Mensaje de éxito al editar la pregunta

ID Caso de Uso	**CU-029**
Caso de Uso:	**Modificar estado de la consultoría**
Descripción:	El usuario requiere modificar estado de la consultorías
Actores:	Estudiante

Precondiciones:
- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber creado al menos una consultoría (Caso de uso CU-025)
- Haber escogido la sección Mis *Actividades* del menú principal
- Haber escogido la sección *Consultorías Activas* del menú principal

Escenario principal de éxito (Flujo normal):
1. El sistema presenta la información de la consultoría activa seleccionada y muestra el campo *Pregunta.*
2. El usuario selecciona la pregunta y escoge *Editar*.
3. El usuario muestra la pregunta que previamente se ingresó.
4. El usuario modifica la información de la pregunta
5. El sistema verifica la información ingresada.
6. El sistema muestra la pregunta modificada

Extensiones (o Flujo Alternativo):

2.a El usuario no ha ingresado pregunta
3. El sistema no muestra nada, se debe crear una pregunta
4. Volver el punto 1 del flujo normal

Poscondiciones (Garantías de éxito):
- Mensaje de éxito al editar la pregunta

ID Caso de Uso	**CU-030**
Caso de Uso:	**Estudiante crea nueva actividad**
Descripción:	El usuario requiere crear actividades
Actores:	Estudiantes

Precondiciones:
- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Mis Actividades* del menú general

Escenario principal de éxito (Flujo normal):
1. El usuario escoge la opción *Nueva Actividad*
2. El sistema presenta los campos: nombre, objetivo, descripción, fecha inicio, fecha fin.

3. El usuario ingresa la información requerida.
4. El usuario escoge la opción para *Guardar.*
5. El sistema verifica la información registrada.
6. El sistema crea la nueva actividad y muestra un mensaje que la actividad se creó con éxito

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Ingresar una actividad exitosamente.

ID Caso de Uso	**CU-031**
Caso de Uso:	**Administrador de Sección aprueba actividad**
Descripción:	El usuario requiere aprobar actividades creadas en el sistema por miembros estudiantiles.
Actores:	Administradores de Sección

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Actividades* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema muestra la lista de actividades de la sección en cuestión.
2. El usuario escoge la actividad que desee activar.
3. El usuario actualiza el estado de la actividad.
4. El usuario escoge la opción para *Actualizar.*
5. El sistema verifica la información registrada.
6. El sistema aprueba la actividad y muestra un mensaje de que será informado al miembro estudiantil.

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

7. El sistema presenta un aviso indicando que los datos ingresados son inválidos
8. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Actualizar el estado de una actividad.

ID Caso de Uso	**CU-032**
Caso de Uso:	**Cambiar miembro Estudiantil responsable de actividad**
Descripción:	El usuario requiere cambiar miembros estudiantiles responsables de una actividad.
Actores:	Administradores de Sección

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Actividades* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema muestra la lista de actividades de la sección en cuestión.

2. El usuario escoge la actividad que desee modificar.
3. El usuario actualiza el estudiante encargado de la actividad.
4. El usuario escoge la opción para *Actualizar.*
5. El sistema verifica la información registrada.
6. El sistema modifica la actividad y muestra un mensaje de que será informado al miembro estudiantil.

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

9. El sistema presenta un aviso indicando que los datos ingresados son inválidos
10. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Actualizar el estudiante responsable de una actividad.

ID Caso de Uso	CU-033
Caso de Uso:	**Registro de usuario en el sistema**
Descripción:	El usuario requiere ser creado en el sistema.
Actores:	Administradores de sección, estudiante, mentor, administrador de región.

Precondiciones:

- Haber escogido la opción *Nuevo Usuario* del menú principal

Escenario principal de éxito (Flujo normal):

1. El sistema muestra las diferentes opciones para tipos de usuario a crear.
2. El usuario escoge el tipo de usuario a crear.
3. El usuario ingresa los datos, dependiendo del tipo de usuario.
   3.1. Si es usuario tipo mentor deberá ingresar las subcategorías en las que se encuentra habilitado.
4. El usuario escoge la opción para *Ingresar.*
5. El sistema verifica la información registrada.
6. El sistema realiza las siguientes acciones dependiendo del tipo de usuario a crear:
   6.1. Si es mentor se envía un correo al Administrador de Sección para su aprobación.
   6.2. Si es miembro Estudiantil, se enviará una notificación al correo del estudiante para indicarle su contraseña.
   6.3. Si es Administrador de Sección se enviará un correo al administrador de región para su aprobación.

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Ingresar datos de usuario al sistema de manera exitosa.

ID Caso de Uso	CU-034
Caso de Uso:	**Administrador de Sección aprueba/rechaza usuario**

	mentor
Descripción:	El usuario requiere aprobar o rechazar usuarios de tipo mentor.
Actores:	Administradores de Sección

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Mentores* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema muestra la lista de los usuarios tipo mentor de la sección en cuestión.
2. El usuario escoge la el mentor que desee activar o inactivar.
3. El sistema muestra el detalle del mentor.
4. El usuario actualiza el estado del mentor.
5. El usuario escoge la opción para *Actualizar.*
6. El sistema verifica la información registrada.
7. El sistema modifica el mentor y muestra un mensaje de que será informado al mentor.

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Actualizar el estado de un mentor.

ID Caso de Uso	**CU-035**
Caso de Uso:	**Administrador de Región(AR) aprueba/rechaza usuario Administrador de Sección (AS)**
Descripción:	El usuario requiere aprobar o rechazar usuarios de tipo Administrador de Sección.
Actores:	Administradores de Región

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Usuarios* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema muestra la lista de los usuarios tipo Administrador de Sección de la Región del usuario AR.
2. El usuario escoge la el AS que desee activar o inactivar.
3. El sistema muestra el detalle del AS.
4. El usuario actualiza el estado del AS.
5. El usuario escoge la opción para *Actualizar.*
6. El sistema verifica la información registrada.
7. El sistema modifica el AS y muestra un mensaje de que será informado al mentor.

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Actualizar el estado de un Administrador de Sección.

ID Caso de Uso	**CU-036**
Caso de Uso:	**Modificar datos de usuario**
Descripción:	El usuario requiere modificar sus datos.
Actores:	Administradores de región,

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Mis Datos* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema muestra el detalle del usuario con opción a modificar.
2. El usuario actualiza lo que desee.
3. El usuario escoge la opción para *Actualizar.*
4. El sistema verifica la información registrada.
5. El sistema modifica el usuario y muestra un mensaje de que será informado al mentor.

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Actualizar los datos de un usuario.

ID Caso de Uso	**CU-037**
Caso de Uso:	**Solicitar cambio de rol**
Descripción:	El usuario requiere cambiar el rol que tiene
Actores:	Miembro estudiantil, mentor

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Mis datos* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema muestra el detalle del usuario.
2. El usuario escoge la opción *Solicitar ascenso*.
3. El sistema muestra el detalle de las opciones a llenar para el ascenso.
4. El usuario escoge la opción para *Aceptar.*
5. El sistema verifica la información registrada.
6. El sistema muestra un mensaje de que será informado al Administrador de Sección.

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

1. El sistema presenta un aviso indicando que los datos ingresados son inválidos
2. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Actualizar el estado de un Administrador de Sección.

ID Caso de Uso	CU-038
Caso de Uso:	**Consultar Directorio Gold-Region**
Descripción:	El usuario requiere consultar el directorio de los miembros GOLD de la región
Actores:	Publico

Precondiciones:

- Haber escogido la opción *Directorio GOLD* del menú principal

Escenario principal de éxito (Flujo normal):

1. El sistema muestra la lista de las regiones disponibles.
2. El usuario escoge la Región que desee consultar.
3. El sistema muestra la lista de los usuarios GOLD de la Región
4. El usuario escoge el detalle del GOLD que desee.
5. El sistema muestra el detalle del GOLD.

Extensiones (o Flujo Alternativo):

3.a. No existen GOLDs en la Región

Poscondiciones (Garantías de éxito):

- Mostrar la lista de GOLDs de la Región.

ID Caso de Uso	CU-039
Caso de Uso:	**Consultar Directorio Gold-Sección**
Descripción:	El usuario requiere consultar el directorio de los miembros GOLD de la sección
Actores:	Publico

Precondiciones:

- Haber escogido la opción *Directorio GOLD* del menú principal

Escenario principal de éxito (Flujo normal):

1. El sistema muestra la lista de las regiones disponibles.
2. El usuario escoge la Región que desee consultar.
3. El sistema muestra la lista de los usuarios GOLD de la Región
4. El usuario escoge el detalle del GOLD que desee.
5. El sistema muestra el detalle del GOLD.

Extensiones (o Flujo Alternativo):

3.a. No existen GOLDs en la Sección seleccionada

Poscondiciones (Garantías de éxito):

- Mostrar la lista de GOLDs de la Sección.

ID Caso de Uso	CU-040
Caso de Uso:	**Cambiar contraseña**
Descripción:	El usuario requiere cambiar su contraseña
Actores:	Administradores de Región, mentor, administrador de sección, estudiante

Precondiciones:

- Haber iniciado sesión en el portal. (Caso de uso CU-001)
- Haber escogido la opción *Mis datos* del menú general

Escenario principal de éxito (Flujo normal):

1. El sistema muestra el detalle del usuario.

2. El usuario escoge la opción *Cambiar contraseña.*
3. El sistema muestra los campos *contraseña anterior*, *nueva contraseña* y *repetir nueva contraseña*.
4. El usuario ingresa los datos solicitados.
5. El usuario escoge la opción para *Actualizar.*
6. El sistema verifica la información registrada.
7. El sistema modifica el usuario y muestra un mensaje de éxito.

Extensiones (o Flujo Alternativo):

3.a. El usuario ingresa datos erróneos o incompletos

3. El sistema presenta un aviso indicando que los datos ingresados son inválidos
4. Volver al punto 1 del flujo normal

Poscondiciones (Garantías de éxito):

- Actualizar la contraseña de un usuario.

BIBLIOGRAFIA

[1] IEEE Webmaster, About IEEE, http://www.ieee.org/web/aboutus/home/index.html, consultado el 25 de septiembre de 2008.

[2] Java 2 Platform, Enterprise Edition (J2EE) Overview, http://java.sun.com/j2ee/overview.html,consultado el 12 de Agosto de 2009..

[3] Juan Antonio Palos, Introducción al Servidor de aplicaciones iPlanet, http://www.programacion.com/java/tutorial/ipintro/4/, consultado del 15 de septiembre de 2009.

[4] Javier Eguíluz Perez, Introducción a AJAX, http://www.librosweb.es/ajax/, consultado el 12 de agosto de 2009.

[5] http://sherekan.com.ar/2008/04/19/introduccion-a-ajax/, Blog de programación, consultado el 15 de septiembre de 2009.

[6] Wium Lie H., Bos B., *Cascading Style Sheets, designing for the Web*, (2nd edition, 1999, Addison Wesley, ISBN 0-201-59625-3), capítulo 2.

[7] http://www.w3.org/Style/CSS/

[8] Diseño Orientado a objetos, http://es.wikipedia.org/wiki/Diseño_orientado_a_objetos, consultado el 15 de septiembre de 2009.

[9] Core J2EE Patterns: Best Practices and Design Strategies, http://java.sun.com/blueprints/corej2eepatterns, Página official de Sun acerca de estándares desarrollados para J2EE.

[10] Sommerville I., Ingeniería de Software, 6ta edición, capítulo 20, página 443

[11] Casos de uso, http://en.wikipedia.org/wiki/Use_case, consultado el 15 de septiembre de 2009.

[12] Prototype, http://es.wikipedia.org/wiki/Prototype, consultado el 08 de octubre de 2009.

[13] Hibernate, http://es.wikipedia.org/wiki/Hibernate, consultado el 08 de octubre de 2009.