UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA DE ADMINISTRACIÓN DE PLANOS Y DOCUMENTOS CAD DE INGENIERÍA EN EL IMP

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

ERWIN MORALES ESPINOSA

Director de Tesis: M. en C. Jesús Sosa Iglesias
Codirector de Tesis: Ing. Carlos A. Román Zamitiz

CIUDAD UNIVERSITARIA

Septiembre de 2003

A mis padres, Porfirio & Evelia:

Por haberme dado la vida, así como todo el amor y apoyo incondicional en todo momento, tanto moral como material.
Gracias por la mejor herencia que pudieron darme: mi educación, los valores que me han inculcado y el constante ejemplo de seguir siempre adelante.

A mis hermanos, Paty & Saint:

El apoyo de ustedes ha sido pieza fundamental de mi desarrollo, les agradezco enormemente todo lo que han hecho por mi.

A mi familia:

Abuelito(a)s, mis tíos, primos, cuñados(Paty y Alberto), y muy en especial a la familia Cigarroa Espinosa. Ustedes ocupan una parte muy especial dentro de mi.

A mis amigos:

Viridiana Borralles, Carlos, Myriam(Folle), Israel N. Turcio, Maricarmen Hernández, Itzel y Omar Rodríguez. Gracias por tocar y enriquecer mi existencia.

A Liz, sabes que gran parte de este trabajo también te pertenece, gracias por permitirme crecer como persona junto contigo.

A Lulú, gracias por todo tu apoyo, compañía, amor y enseñanzas. Jt'a bcp súper.

Al movimiento Scout y al grupo 2 Misol-ha porque mas que una actividad sabatina, fue y seguirá formando parte de mis cimientos como persona.

A la Facultad de Ingeniería y mi Universidad por las enseñanzas y experiencias vividas dentro de ella.

Erwin.

Agradecimientos:

Agradecimiento especial al M. en C. Jesús Sosa Iglesias por todo el tiempo que dedicó para la culminación de este trabajo de Tesis, así como por el conocimiento que me ha transmitido en cursos y consultas durante mi estancia en el Instituto Mexicano del Petróleo, sin duda éste trabajo no hubiera llegado a su fin sin su valiosa asesoría.

A Jorge Moreno Hdez, por el apoyo recibido dentro del IMP y en el campo laboral durante la realización de éste trabajo.

A la M. en C. Ma. de Lourdes Martínez Bermúdez por su apoyo dentro del IMP.

A todos mis profesores y sinodales de examen profesional, en especial a Carlos Román, Alberto González, Marcial Contreras, J. Manuel Gómez.

Índice

Índice de Figuras y List

Figuras

Listados

RESUMEN

Dentro del área de Ingeniería del Instituto Mexicano del Petróleo se llevan a cabo trabajos de diseño y planeación de instalaciones petroleras, por lo que se genera una gran cantidad de conocimiento plasmado en planos generados por medio de Diseño Asistido por Computadora entre otros formatos como documentos creados por procesador de texto, hoja de cálculo, imágenes, audio, video, etc. Esto genera la necesidad de contar con un recurso computacional que satisfaga las necesidades que surgen cuando se requiere que dicha información sea compartida entre los miembros de los distintos grupos de personas que laboran en ésta área.

Este trabajo de tesis tiene como objetivo diseñar e implementar un sistema computacional que cubra cierto tipo de necesidades dentro del área bajo un ambiente de Intranet con el fin de dar acceso a dicho recurso desde cualquier computadora conectada a la Intranet institucional. El sistema ha sido diseñado utilizando herramientas de modelado(UML) siguiendo estándares de programación orientada a objetos y la metodología de desarrollo Objectory, bajo una arquitectura de tres capas. La construcción del sistema se realizó utilizando el lenguaje de programación orientado a objetos Java, utilizando componentes de la versión J2EE(Java 2 Enterprise Edition), en base al diseño antes realizado.

De igual manera, se describe la metodología de pruebas realizada al sistema, la cual se ajustó a la metodología empleada en la fase de desarrollo(Objectory). Ésta se basa en emplear a los casos de uso generados a partir de los requerimientos del sistema para llevar la planeación del desarrollo y en el caso de las pruebas, tener una referencia de los elementos o sub-módulos que se deben someter a prueba y que es lo que se espera de ellos.

El resultado de todo este proceso es la construcción de un sistema que ha sido diseñado, construido y probado bajo una metodología establecida que garantiza la calidad del software creado para solucionar los problemas de distribución de conocimiento que se tienen dentro del área de Ingeniería del Instituto Mexicano del Petróleo.

Introducción

Este trabajo de tesis tiene como objetivo principal la creación de un sistema de administración de documentos y planos CAD para el área de Ingeniería del Instituto Mexicano del Petróleo, institución que sustenta una importante plataforma para la investigación científica y el desarrollo tecnológico al servicio de la industria petrolera en general.

Los trabajos que se realizan dentro de dicha área del IMP son básicamente del ramo de desarrollo tecnológico, en su mayoría la planeación de plataformas marinas y algunas otras estructuras que requieren del diseño asistido por computadora, además de la planeación alterna. Estas tareas generan una gran cantidad de conocimiento el cual debe tener una buena distribución para optimizar todos los procesos que ahí se realizan, por lo que se desarrolló este sistema como una medida para mejorar dicha distribución.

En el capítulo 1 titulado Justificación del sistema de administración de documentos se realiza un análisis de los procesos actuales de distribución del conocimiento dentro del área de Ingeniería del IMP, determinando sus errores a corregir así como los aciertos con que cuenta y que pueden ser retomados por el sistema a construir. De esta manera se propone la construcción de dicho sistema, el esquema de construcción y los objetivos de dicho sistema generados a partir de las necesidades existentes.

En el segundo capítulo que lleva por titulo Análisis Previo de un Sistema de Administración de Documentos(SAD) se propone la metodología Objectory como metodología de desarrollo del software propuesto en el primer capítulo de éste trabajo de tesis. Se realiza también un análisis de requerimientos, los cuales se plasman en la creación de los casos de uso del sistema, definiendo los actores que interactuarán con el sistema y utilizando UML para expresar gráficamente los diagramas de casos de uso. Así mismo se propone la tecnología en que se basa la construcción del

sistema y se propone como la arquitectura base del sistema un modelo en tres capas regido por el patrón Modelo Vista Controlador(MVC).

Por su parte, en el capítulo tercero llamado Diseño del Sistema de Administración de Documentos, como su nombre lo indica se realiza el modelado de las tres capas que comprenden la arquitectura del sistema, por una parte se modela la capa de datos mediante la realización del modelo Entidad Relación que rige a la base de datos del sistema. Respecto a la capa de lógica y procesos se modela la estructura lógica del sistema con la ayuda de UML, desde la definición y creación de los diagramas de paquetes, a la definición de los objetos que realizarán los procesos de control y de negocio con la capa de datos, expresados gráficamente en los diagramas de clases clasificados por paquetes así como la definición de las interacciones entre los objetos concebidos mediante la creación de diagramas de secuencias los cuales se realizan en base a los casos de uso generados en el capítulo dos. Finalmente se describe el diseño de la interfaz de usuario, la tecnología a emplear en este caso y el fin que persigue.

El cuarto capítulo, Implementación del Sistema de Administración de Documentos, se describen los procesos que se siguieron para la construcción del sistema en base al diseño creado y que es descrito en el capítulo tres. De la misma manera, esta dividido según la arquitectura propuesta de tres capas mediante la descripción de la creación de la base de datos por parte de la capa de datos, la descripción detallada de los objetos creados textualmente y de manera gráfica mediante el uso de diagramas de clases para la parte de lógica y control, y finalmente se describe la creación de las interfaces de usuario basadas en la tecnología JSP, las cuales son clasificadas respecto a los casos de uso del sistema.

El plan de pruebas implementado durante el desarrollo del sistema esta descrito en el quinto capítulo de éste trabajo, pruebas durante el ciclo de desarrollo, dentro del cual se describen también las pruebas realizadas al sistema y que éstas fueron en base a la metodología Objectory, la cual rige la fase de desarrollo del sistema. Al finalizar estas pruebas, se tiene la certeza de que se obtuvo un producto con calidad considerable y que dichas pruebas se realizaron de una manera metodologica establecida.

Para finalizar, el capítulo seis contiene las conclusiones finales de éste trabajo, así como los logros obtenidos y los posibles trabajos a futuro por realizar sobre el Sistema de Administración de Documentos para obtener mejores resultados a los obtenidos dentro de éste proyecto.

CAPÍTULO

1

Justificación del Sistema de Administración de D

En este capítulo, se realiza un análisis del sistema de Administración de documentos que se utiliza dentro del Instituto Mexicano del Petróleo. Se podrá conocer mas acerca de la forma en que se realizan los trabajos dentro del área, con el fin de comprender mas los procesos que se llevan a cabo y así resaltar las necesidades que se tienen, de la misma manera se describirán los procesos de distribución de información entre las personas que trabajan conjuntamente dentro del área, así como sus deficiencias y aciertos.

Como consecuencia del análisis, se determinará que este método empleado es insuficiente para un mejor aprovechamiento de los recursos humanos y técnicos que se tienen, es decir se tiene un mal sistema de administración del conocimiento que no ayuda a mejorar y acelerar los procesos de producción de nuevos productos dentro del área.

Respondiendo a estas necesidades se planteará posteriormente la creación de un sistema de administración de planos y documentos CAD dentro del área el cual tenga ciertas características que mejoren la administración del conocimiento. Para tener una idea de cuales son los recursos que se podrían utilizar, se hará una breve descripción de los recursos técnicos con que el Instituto cuenta, en especial los referentes a hardware y software.

Finalmente después de haber definido el problema a resolver y sabiendo con que recursos se cuentan, se definirán una serie de objetivos preliminares con los que dicho sistema debe cumplir, así como un objetivo genérico que describirá en pocas palabras lo que el producto final debe de realizar al final de su realización.

1.1 Marco Conceptual

Al interior de las actuales organizaciones o empresas, el conocimiento es una de las palabras y conceptos clave para el desarrollo de las mismas. Si éste no se esta generando continuamente, puede darse el caso de estancamientos que lleven a un fracaso total de la organización, ya que dentro del esquema actual de interacción entre todo tipo de organizaciones, la competitividad es de suma importancia para tomar un lugar entre la preferencia del grupo o grupos a los que estén enfocados los productos o servicios que éstas ofrezcan.

Por otro lado, en sí el conocimiento por si solo no representaría una significación importante, sino que debe tener ciertos aspectos que transformen este conocimiento generado en conocimiento realmente útil y aplicable en la mayor cantidad de casos posibles dentro de la misma empresa o incluso al exterior de ésta con el fin de lograr un desarrollo integral.

Observando al conocimiento desde otro aspecto, éste es un recurso volátil, no tangible, difícil de concretar y retener, además de que los cambios son extremadamente continuos, de tal manera que resulta difícil conservarlo por mucho tiempo como un recurso totalmente útil.

En México, existen muchas instituciones que se dedican a la generación y aplicación de nuevo conocimiento y entre las mas importantes se encuentra el Instituto Mexicano del Petróleo(IMP).

El Instituto Mexicano del Petróleo desde su creación ha sido una importante base para la investigación científica y el desarrollo de tecnologías al servicio principalmente de las industrias petrolera, petroquímica básica, petroquímica derivada y química entre otras. Además es una institución que se moderniza día con día que tiene como propósito asegurar el fortalecimiento de la investigación y desarrollo tecnológico mediante programas y proyectos de investigación de vanguardia, orientando sus esfuerzos hacia soluciones con servicios integrados brindados principalmente a Petróleos Mexicanos, su principal cliente, y hacia el interior de la organización.

El IMP esta organizado de manera que los trabajos de investigación, técnicos, referentes a exploración y producción, medio ambiente, capacitación, administrativos y de comercialización se distribuyen entre nueve grandes subdirecciones ejecutivas, las cuales tienen actividades específicas que en su conjunto hacen posible que dicha institución trabaje de una manera ordenada y con la competitividad que es requerida.

Entre estas grandes subdirecciones, una de las mas destacadas debido al conocimiento que genera, es la subdirección de Ingeniería, la cual tiene entre sus líneas de investigación:

- El desarrollo de metodologías, sistemas y programas para el diseño, dibujo e ingeniería asistida por computadora.

- El diseño de instalaciones petroleras, como plantas de proceso, petroquímicas y plataformas marinas.

- La realización de diseño conceptual, de detalle de equipo de proceso, de intercambio de calor, combustión e instrumentación, fijando sus condiciones de operación mas adecuadas.

El producto de estas líneas de investigación consiste en un conjunto de diversos documentos en distintos formatos(doc, pdf, etc), dibujos y planos CAD de detalle requeridos para la adquisición de equipo y

materiales y para la propia construcción de la obra, así como el servicio de diseño térmico, hidráulico, mecánico y estructural del equipo de proceso requerido en plantas industriales y de petróleo.

1.2 Administración de Documentos en Ingeniería del IMP

Para iniciar a describir el actual sistema con el cual se distribuye la información entre las personas que laboran dentro de la institución, es necesario primero mencionar el tipo de trabajo que se realiza y como se realiza.

Como se mencionó en el marco conceptual, la principal actividad de ingeniería es el diseño de nuevas plantas petroquímicas o bien el diseño de la extensión de alguna ya existente. Estas actividades generan una gran cantidad de dibujos y documentos que representan los planos de construcción y de especificación de las plantas de proceso, estos dibujos y documentos antiguamente los realizaban los dibujantes a “mano”, pero con la inserción de las tecnologías CAD(Diseño asistido por computadora), todo el esquema antiguo se vio forzado a cambiar por esta nueva modalidad, en la cual los dibujantes ahora solo diseñan con la ayuda de la computadora y el software adecuado para estos fines.

Dentro de este tipo de software especializado se encuentran paquetes como Autocad, MicroStation, IntelliCAD, AbisCAD, Solid EDGE, TrueSpace 3D, pero dentro del Instituto Mexicano del Petróleo por conveniencia propia se manejan principalmente el Autocad y Microstation. Aunque no solamente la información que se genera proviene de este tipo de software, en este caso si seria la principal fuente de la información.

Adicional a los planos creados sobre las herramientas CAD, dentro de esta área se comparten archivos de texto, archivos generados con procesadores de palabras, hojas de calculo, archivos de texto estático como postscript o pdf, archivos de hipertexto como puede ser html, archivos de configuración en texto plano, archivos de configuración en formato xml, archivos con imágenes, archivos de sonido y en ocasiones muy especiales archivos multimedia que contienen audio y video, entre muchos otros tipos de archivos.

Por otra parte, dentro de esta área, como dentro de todo el IMP, los trabajos que se realizan, pertenecen a diferentes proyectos, algunos proyectos son para su uso dentro del Instituto, y otros proyectos se realizan para empresas externas, que en su mayoría es Petróleos Mexicanos. Así, las distintas especialidades de Ingeniería que esta trabajando sobre un mismo proyecto deben tener comunicación completa con otras especialidades y departamentos de manera que los trabajos que alguna persona realiza deben de estar al alcance de los demás participantes, a menos que se impongan restricciones que no lo permitan.

Actualmente, esta distribución de la información se realiza a través de la red compartiendo carpetas dentro del sistema operativo Windows o bien a través de su transporte en medios de almacenamiento como disquetes, correo electrónico y CD’s entre otros.

Uno de los problemas que de esto deriva es la complicada localización de algún archivo, pues la persona que comparte los recursos debe de indicar a los demás en que computadora que esta conectada a la red se encuentra el archivo además de la carpeta dentro del mismo equipo. Aunado a esto, dentro del IMP, no hay una uniformidad de sistemas operativos, debido a que algunas maquinas mas antiguas no soportan los sistemas operativos actuales, o bien no hay compatibilidad entre ellos.

Existen computadoras principalmente con Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, las cuales no tienen muchos problemas de comunicación entre ellas, salvo que el acceso desde cualquier computadora hacia alguna que tenga su Windows basado en tecnología NT, es mas complejo pues las restricciones de seguridad se incrementan, causando molestias para compartir

carpetas.

Por otra parte, no solo se cuenta con computadoras con sistema operativo Windows, se trabaja también sobre estaciones de trabajo Sillicon Graphics bajo el sistema operativo IRIX, algunos equipos con computadoras de Sun Microsystems con Solaris como sistema operativo y un numero reducido de PC's con alguna distribución Linux como red hat Linux o mandrake, operando sobre ellas. Estos últimos sistemas operativos son variantes de UNIX, el cual no tiene compatibilidad directa con ninguna versión de Windows, lo que hace aun mas complejo la compartición de archivos.

Dejando de lado los problemas que acarrea el compartir archivos por medio de la red por medio de los sistemas operativos, podríamos considerar algunas otras alternativas como es instalar un servidor FTP(File Transfer Protocol) en el cual los usuarios carguen y descarguen los archivos que realizan y de la misma manera se separarían para ordenarlos un poco mas. Esto acarrearía algunos beneficios como el uso de la red y de casi cualquier computadora que pudiera funcionar como este tipo de servidor, los problemas radican en la dificultad para el uso del software cliente de este tipo de servidor, principalmente para montar archivos al servidor, así como la falta de orden que podrían causar los usuarios de dicho servicio.

El método mas convencional para transferir los archivos, es por medio de dispositivos de almacenamiento como son disquetes, CD ROM, cinta magnética, correo electrónico etc. Como se supone ya, estos medios son los menos seguros ya que los materiales con los cuales están hechos en ocasiones son muy delicados y pueden resultar dañados en el proceso de su transporte, se pueden cometer errores de copiado, errores de lectura, etc, que finalmente afectan el proceso de la distribución de la información y el conocimiento.

Los trabajos realizados dentro de esta área, como se mencionó, son planos sobre los cuales trabajan varias personas, hay personas especializadas en realizar ciertas tareas sobre estos planos lo que provoca esta situación, por lo que es necesario llevar un conjunto de versiones desde su inicio hasta la terminación del proyecto, esto comúnmente la gente lo realiza modificando el nombre del archivo y colocándolo en la carpeta que se tiene prevista para este fin.

1.3 Deficiencias de la administración de documentos

En la sección anterior se mencionaron las diferentes maneras en que se puede distribuir la información entre las personas que laboran dentro del área de Ingeniería Asistida por Computadora, que realmente son muy básicas, llenas de deficiencias como las dificultades de uso, y en algunos puntos con aciertos, como el nivel de seguridad que brinda resguardar la información bajo un equipo con sistema operativo Unix o alguna de sus variantes, o el manejo de usuarios que se cuenta con la tecnología NT de Windows y Unix.

Dentro de las anteriores formas que se mencionaron, en las que se distribuyen los archivos, hay varios factores en común que las hacen deficientes, las cuales engloban un punto clave para el buen manejo de la información: la Administración. Administración se entiende como el manejo eficiente de asuntos, en este caso información, siguiendo determinadas reglas.

Los métodos anteriormente descritos tienen varias deficiencias las cuales hacen difícil una integral distribución de la información, estas deficiencias se citan a continuación:

- No se puede tener una referencia clara de que persona coloca los archivos sobre un fólder compartido o bien quien fue la persona que transporto por medio de unidades de almacenamiento extraíbles algún archivo, lo que causa en ocasiones mucho conflicto en el caso de que se encuentren errores o dudas sobre el contenido de las modificaciones.

No se tiene un manejo automático de las versiones, esto lo realizan los dibujantes cambiando el nombre del archivo lo que acarrea problemas debido a olvidos o descuidos que provocan la sobre escritura de versiones muy importantes haciéndolas irrecuperables, así mismo no se puede tener una idea clara de cual ha sido el avance conforme al tiempo de un proyecto.

Errores en la ubicación de los archivos, el problema de la sobre escritura se presenta nuevamente, ya que usuarios pueden sobrescribir archivos que por alguna cuestión tenían nombres iguales pero no por eso contenían lo mismo, provocando una perdida importante de información por errores humanos, descuidos o bien por la falta de orden dentro de las carpetas que se comparten.

Es muy difícil llevar un control del avance sobre los proyectos pues no se tienen mecanismos para determinar precisamente las fechas de colocación de archivos, y a la vez se pueden presentar problemas de los ya citados con anterioridad, que harían esta tarea aun mas complicada.

Complicados mecanismos de búsqueda de archivos, ya sea por su nombre, características o bien por descripciones, las cuales no se pueden realizar al compartir archivos directamente a otra computadora. Esto acarrea una lenta distribución, ya que cuando el volumen de archivos es grande se torna complicado el realizar esta tarea.

Incompatibilidad entre sistemas operativos, aunque principalmente se utiliza AutoCAD para realizar los planos, siendo que este corre sobre el sistema operativo Windows, existen estaciones de trabajo corriendo sobre el sistema operativo IRIX, una variante de UNIX, así como computadoras bajo distribuciones Linux, entre las cuales hay una gran diferencia en cuanto a compatibilidad se refiere, no es la misma facilidad de transferencia de archivos entre dos equipos con Windows, que la que existe entre dos equipos UNIX, y menos aun entre un Windows y un UNIX. La única manera es con la utilización de software de compatibilidad entre estos dos sistemas, o bien la utilización de un servidor FTP, el cual tendría los mismos problemas que ya se mencionaron anteriormente.

No se puede tener una descripción de los archivos que se almacenan, lo que dificulta su identificación, aunque en muchas ocasiones el nombre del archivo es representativo de su contenido, hay ocasiones en que es necesaria una descripción mas detallada para los fines que convengan.

Afortunadamente, estos sistemas de distribución de la información, no solamente tienen deficiencias, sino que cuentan con algunas cualidades explotables, las cuales se pueden tomar en cuenta para incrementar la versatilidad y utilidad del sistema de administración de documentos que se planea realizar. A continuación se mencionan algunos de estos aciertos que pueden ser tomados en cuenta:

El principal punto a considerar es el nivel de seguridad que los sistemas operativos proveen, los cuales van desde la autentificación para dejar o no ingresar a un usuario al sistema, hasta los permisos que se le pueden asignar a cada archivo o carpeta. Principalmente si tomamos como referencia a los sistemas Unix o basados en este mismo, a los cuales se les pueden asignar permisos de lectura, escritura, ejecución, propiedad individual, propiedad por grupo, etc.

El segundo punto importante a considerar, es el orden jerárquico en que se formula el sistema de archivos de los sistemas operativos actuales, en los cuales se forman carpetas sobre las cuales se pueden anidar mas carpetas formando un sistema de archivos que se asemeja a un árbol eneario, ideal para una fácil navegación entre estas y la facilidad de jerarquizar la información contenida en ellas.

La capacidad de identificación del contenido y la cantidad de información medida en bytes que los archivos tienen. Gráficamente o por medio de las extensiones de identificación de los archivos, dentro de cualquier sistema operativo actual es posible determinar fácilmente de que tipo de archivo se trata con solo observar detalles como el icono que lo identifica.

Respaldos, en el caso de la transferencia de archivos por medio de unidades de almacenamiento extraíbles(disquetes, CD-ROM, cinta magnética, etc) o incluso esta haciendo de alguna manera un respaldo, muy útil en caso de imprevistos respecto al equipo o a un error humano, por ejemplo una sobre escritura o la eliminación accidental de un archivo.

Como se observa, existen puntos a favor y en contra del actual sistema de administración de documentos que se maneja dentro de esta área, pero el fin de la tecnología es que los humanos podamos realizar las tareas que tenemos a cargo de una manera mas fácil y eficiente. De esta manera, este listado de puntos a favor y en contra del actual sistema debería de tener una tendencia a listar mas aciertos que deficiencias dentro de sus operaciones para considerarse eficiente, además de que estos aciertos deberían de estar coordinados y relacionados entre sí, y en este caso los aciertos están aislados o bien estos mismos se relacionan con alguna deficiencia, principalmente las dificultades de manejo, o en algunos otros casos, el tiempo que se requiere para realizar alguna operación.

De esta manera se muestra que el sistema usado en la actualidad no es eficiente ya que las deficiencias que presentan son importantes y aunque se ha podido llevar normalmente la actividad durante este tiempo bajo estos procesos, es necesario que sean superadas las deficiencias para poder llegar a mejores niveles de productividad, siguiendo lo mas cercanamente posible prácticas de administración del conocimiento eficientes que hagan las tareas mas fáciles para la gente que labora en el área, y a la vez mayormente controlables por las personas que se encargan de la coordinación de los proyectos.

1.4 Propuesta para mejorar la Administración de Documentos en el IMP

Como se ha venido diciendo, es necesaria una solución que resuelva las necesidades del área ya citadas, con la utilización de herramientas computacionales que no requieran de una gran inversión económica, tratando de explotar al máximo los recursos con que se cuentan dentro del instituto.

Por esto, se propone en este trabajo de tesis, el diseño y construcción de un sistema computacional que integre el mayor posible número de soluciones que cubran las necesidades que se tienen de forma completa y con la menor utilización de recursos posible.

Como es de esperarse, la tecnología esta inmersa dentro del Instituto Mexicano del Petróleo, y dentro de esta institución el uso de la computadora como herramienta de trabajo es esencial, y el área de Ingeniería Asistida por Computadora no es la excepción. Se puede decir que la gran parte de los trabajadores dentro de esta área dominan el uso de la computadora y adicionalmente a esto, cada trabajador tiene asignado un equipo el cual puede y debe utilizar para realizar las tareas que se le asignan.

Estos equipos con que se cuentan, están basados en varias plataformas, como ya se mencionó, y la solución a desarrollar deberá contemplar este punto, realizando una aplicación la cual pueda ser accesada desde cualquier tipo de plataforma.

Dentro de los recursos a utilizar, de los mas importantes es la intranet con que se cuenta. Esta es una red de área local(LAN) que utiliza tecnología basada en Web, en otras palabras protocolos abiertos de comunicación (TCP/IP), sistemas estándares de transferencia de archivos(HTTP y FTP), correo electrónico (SMTP, POP3, IMAP), y mensajería entre otros, los cuales se pueden utilizar desde casi cualquier plataforma existente(Windows, UNIX y sus variantes, MAC, etc). Dicho de otra manera, la intranet con

que se cuenta en el instituto es una red privada que utiliza los recursos que han sido desarrollados para Internet para distribuir información y aplicaciones, pero con la diferencia de que solo pueden tener acceso a ella un grupo de usuarios controlado, los cuales solo pueden tener acceso en el interior de la institución o bien desde Internet pero con los debidos modos de acceso, como son las redes privadas virtuales entre otros. Esta red esta conectada a Internet hacia afuera, pero no cualquier persona puede ingresar a esta red desde Internet.

Por su parte, la tecnología Web agrupa una serie de tecnologías relativamente reciente y estándares que hacen posible el buen funcionamiento de Internet o bien de las intranets, entre estas tecnologías, se encuentra el HTML, lenguaje básico para hacer archivos de texto con hipervínculos(hipertexto), los cuales se publican en un servidor web y que se transporta a trabes de la red por medio del protocolo http a diferentes usuarios, que a su vez estos leen dichos archivos por navegadores (browsers).

Esto nos brinda la ventaja de que las aplicaciones desarrolladas bajo este esquema serán mejor utilizadas y con mayor sencillez pues el usuario final únicamente requiere conocer el uso de un navegador web, sin la necesidad de instalar aplicaciones cliente en cada equipo y conocer su funcionamiento. Además, casi todas las plataformas ya traen un navegador incluido en su instalación inicial, haciéndola de esta manera una solución fácil, económica y confiable.

La facilidad se obtiene debido al conocimiento que ya se tiene sobre el software cliente empleado(el navegador web), la economía al ahorrarse el desarrollo, implementación e instalación de nuevo software sobre los equipos de los usuarios finales. Por ultimo la confiabilidad radica en que los estándares utilizados han sido ya probados ampliamente y son continuamente mejorados en cuanto a estabilidad, seguridad y eficiencia.

Adicionalmente a la eficiencia de estas tecnologías, existen esquemas de distribución de la computación las cuales utilizan de una mejor manera el tiempo de procesamiento de los equipos encargados de realizar las tareas de distribución “repartiendo” el trabajo entre varias instancias ya sea entre otros equipos en constante comunicación por medio de la red, o bien dentro del mismo procesador repartiendo el trabajo en procesos distintos. Así entonces se podría utilizar estas tecnologías puesto que se cuentan con los recursos computacionales para desarrollarlo.

En cuanto a la tecnología de desarrollo, esta institución tiene la posibilidad de adquirir licencias de software de desarrollo, se podrían adquirir suites de desarrollo en Web como las implementaciones de desarrollo hechas por Microsoft, IBM, Macromedia, entre otras; pero desde el punto de vista del autor, la compra de este software sería innecesaria debido a que existen plataformas mas robustas las cuales se adquieren sin ningún cargo económico para la institución. De esta manera, lenguajes de desarrollo en Web robustos como Java[java], PHP, Perl; servidores de bases de datos como MySQL, mSQL, postgreSQL; herramientas de diseño de software en UML[uml] como ArgoUML; servidores Web como Apache[apch], e incluso sistemas operativos sobre las cuales se pueden desarrollar las aplicaciones como las diferentes distribuciones de Linux[lnx] o BSD[bsd], pueden obtenerse de manera gratuita en Internet, o bien ya se encuentran dentro de la intranet del IMP, como los recursos contenidos dentro del web dedicado al software libre de la institutución (linux.imp.mx).

En suma, se cuenta con una infraestructura sobre la cual una aplicación basada en Web dispuesta de manera distribuida y desarrollada bajo software libre(freeware) sería la mejor opción a elegir pues es la manera mas eficiente de aprovechar con lo que se cuenta sin un desembolso económico considerable.

1. 5 Objetivo del sistema a desarrollar

Teniendo una vision mas clara de las deficiencias y problemas a resolver, así como de los recursos con que la institución cuenta, se puede formular ahora los objetivos del Sistema de Administración de Planos y Documentos CAD de Ingeniería en el IMP, para de esta manera delimitar los alcances de este trabajo. Para dar una visión general del sistema, que lo pueda definir en unas palabras definiremos un objetivo genérico, y posteriormente para delimitar los alcances del sistema, se enlistara una serie de objetivos mas puntuales con los que debe cumplir el sistema.

1.5.1 Objetivo Genérico

Realizar un sistema de información que incorpore un centro de información con estructura distribuida; que administre recursos de información específicamente documentos y planos CAD, apoyándose eficazmente en herramientas computacionales, en especial la tecnología Web; creando servicios y puntos de acceso, todo orientado hacia el usuario final, la investigación, la creación y consolidación de conocimiento. Juntando así las islas de conocimiento dentro del área y almacenamiento de datos, abriendo las puertas para nuevos servicios reutilizables, creando una administración eficiente de información en documentos y planos CAD para apoyar la toma de decisiones y alcanzar metas propuestas, como un componente mas de la Administración del Conocimiento dentro del área de Ingeniería del IMP.

1.5.2 Objetivos Específicos

Dentro de los objetivos de este trabajo de tesis están comprendidos los siguientes puntos:

- Implementar un repositorio único, centralizado y seguro para todos los documentos electrónicos del área de Ingeniería del Instituto Mexicano del Petróleo(archivos CAD, documentos generados por procesadores de palabras, planillas electrónicas, presentaciones, etc.)
- Posibilitar el acceso distribuido a la información relacionada a un determinado tema en pocos segundos reduciendo significativamente los costos asociados, ganando en productividad.
- Eliminar totalmente los costos originados por información diseminada en diferentes departamentos, diferentes ubicaciones y/o diferentes sistemas.
- Simplificar el acceso a la información usando tecnología Web.
- Terminar con el extravío de archivos y documentos.
- Posibilitar búsquedas de documentos por las distintas categorías dispuestas.
- Posibilitar la creación de reportes y estadísticas de la actualización y consulta de los documentos y/o archivos depositados.
- Seguridad: Puede ser implementada por usuario o grupos de usuarios. El acceso puede ser restringido a diversos tipos de documentos y/o categorías
- Notificación por e-mail de actualizaciones de documentos a los integrantes de los grupos realizados.

- Obtener una vision previa de los documentos con datos significativos que describan el contenido y cometido de cada documento y/o archivo.

- Contar con capacidades de personalización de información para cada uno de los usuarios del sistema, para provocar mayor eficiencia en la utilización de éste.

- Habilitar una navegación sencilla e intuitiva dentro de las carpetas de documentos del repositorio central.

En general, se pueden considerar estos puntos, los cuales se extenderán y entenderán en el capítulo que se refiere al análisis previo al desarrollo del sistema de administración de documentos, pero con esto se puede tener una visión general de los alcances del sistema, así como de sus limitaciones.

Para la realización de este proyecto, es necesario llevar una metodología que conduzca por un buen camino su desarrollo, la cual guíe el desarrollo del software mediante técnicas que lleven a un resultado satisfactorio basado en un producto de software de calidad. Para obtenerlo, es necesaria una buena planeación de todos los procesos para la realización del software, desde la delimitación de los requerimientos del sistema, hasta la arquitectura en que se basará la aplicación. En el siguiente capítulo de este trabajo de tesis se definirá esta metodología , tratando de definir punto por punto lo necesario para realizar esta tarea de análisis y diseño de un sistema de Administración de Documentos(SAD).

CAPÍTULO

2

Análisis Previo de un Sistema de Administración Documentos(SAD)

En este capítulo se define la metodología a utilizar en el diseño y construcción del SAD, esto con el fin de tener una planeación clara del sistema de Administración de Documentos y Planos CAD, señalando las características y fases de este proceso y como se manejarán dentro de todo el proyecto. Posteriormente se realiza un análisis de requerimientos en el cual se definen mediante casos de uso[cus] el funcionamiento del sistema así como los roles de usuario que interactuarán con él.

Posteriormente, se realiza una descripción breve de la tecnología que se propone emplear dentro del proyecto en cuanto a hardware y software. Para finalizar este capítulo se propone una arquitectura generalizada la cual será la base para un correcto proceso de diseño.

2.1 Metodología

Podemos denominar como un proceso a un conjunto de eventos ordenados parcialmente y destinados a alcanzar un objetivo. En Ingeniería de Software el objetivo básicamente es construir un producto de software o la mejora de algún producto que ya existe, en términos generales de procesos de ingeniería, el objetivo es desarrollar o mejorar un proceso de desarrollo.

Para realizar un correcto diseño de una aplicación, es necesario contar con mecanismos que nos permitan llevar una secuencia de acciones a tomar durante todo el proceso de la creación del software que se desea, desde la toma de los requerimientos del sistema, hasta su implementación después de haber sido comprobada su eficacia. Un mecanismo que indica el seguimiento que se

debe aplicar al proceso de creación de software es el proceso para el desarrollo de software Objectory, el cual se describirá a continuación, así como una breve planeación de todo el proceso de creación del sistema de Administración de documentos y planos CAD.

2.2 Proceso para el Desarrollo de Software Objectory

Objectory[oby] es una metodología para el desarrollo de software bajo el esquema de programación orientada a objetos con el fin de obtener como producto final un software de calidad, el cual esta basado en casos de uso, es decir la descripción de elementos o usuarios que son externos al sistema (llamados actores), la cual describe la funcionalidad del sistema. Estos a su vez se aplican durante todas las fases de desarrollo, incluyendo el análisis, diseño, implementación y pruebas.

Estos casos de uso representan un flujo o flujos definidos de eventos dentro del sistema, estos son iniciados por actores y describen la continuidad de eventos que los actores provocan al utilizar el software creado. Por su parte, los actores son las únicas entidades que interactúan con los casos de uso, y estos pueden ser otro sistema, equipos externos o bien un humano. Estos actores representan mas un rol de usuario que un usuario físico, ya que varios usuarios físicos pueden tener un papel similar.

Por otra parte, cuando un sistema software como el propuesto, que se desarrolla desde cero, el desarrollo es el proceso de crear un sistema desde sus requisitos y a través de la creación de los casos de uso. Pero una vez que el sistema toma forma, el desarrollo siguiente es el proceso de amoldar el sistema a los nuevos o modificados requisitos. Esto se aplica durante todo el ciclo de vida del sistema.

De la manera que ya se mencionó, la metodología que se sigue dentro de este trabajo de tesis, se basa en los casos de uso debido básicamente a que estos se convierten en una vía estándar para generar la documentación de la estructura del sistema consecuente, además de que brindan una excelente ayuda en la planeación de mecanismos iterativos para el desarrollo del software, es la guía básica a través del ciclo de vida del software y finalmente sirve como referencia para la creación de mecanismos de pruebas.

En el siguiente diagrama se muestra la fuerte influencia que ejercen los casos de uso sobre los procesos más importantes del diseño, implementación y prueba del software.

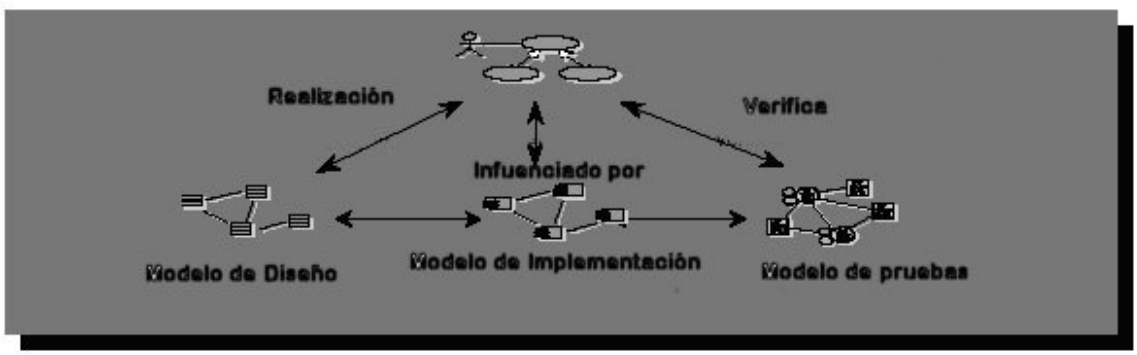

Fig. C2 –1 Influencia de los casos de uso sobre los procesos de diseño, implementación y prueba del software

2.3 Estructura del ciclo de vida

A grandes rasgos, la estructura del ciclo de vida es la organización dinámica del método de desarrollo de software a lo largo del tiempo de creación. A su vez, el ciclo de vida se puede descomponer en ciclos los cuales cada uno trabaja sobre una nueva generación de un producto. El método de desarrollo empleado (Objectory) divide todo el ciclo de desarrollo en cuatro fases consecutivas, que indican desde la planeación del software, hasta las pruebas y su implementación, estas cuatro fases son las siguientes:

Inicio
Elaboración
Construcción
Transición

Cada una de estas fases tiene un fin específico y un punto en el tiempo en el cual se deben tomar decisiones críticas, por esto ciertos puntos clave se deben de lograr al finalizar cada fase. A continuación se describe cada una de estas fases, así como la manera en que se desarrollo para la creación del sistema de administración de documentos y planos CAD.

- **Fase de Inicio**

  En esta fase se estableció el estudio de factibilidad del sistema en el área donde se planea implantar, el área de Ingeniería Asistida por Computadora del IMP, así como la definición de la naturaleza de la aplicación. Para esto se tuvieron que identificar las entidades externas con las cuales el sistema debe interactuar(actores), así como la naturaleza de esta interacción a alto nivel. Esto consiste en la identificación de las personas que utilizarán el sistema y cuales son sus necesidades, de la misma manera se buscó la posible interacción de otros posibles sistemas o entidades que pudieran hacer uso de el producto final. Aquí es donde es necesario identificar todos los casos de uso posibles. Mediante el estudio de factibilidad se identifican los criterios de éxito del sistema, los riesgos que se tienen y una estimación de los recursos necesarios para lograr la finalización del mismo.

- **Fase de Elaboración:**

  El fin de esta fase de elaboración fue el analizar el dominio del problema, estableciendo una base arquitectónica y desarrollando un plan de proyecto. En la base arquitectónica se tomó en cuenta el sistema completo, lo cual implica el haber recibido el mayor número posible de casos de uso de la fase anterior, tomando en cuenta las restricciones que pueda tener el sistema.

  Al finalizar la fase de elaboración se examinó si los objetivos del sistema están claros, el entorno en que se desarrollan, la elección de la arquitectura y su factibilidad.

- **Fase de Construcción:**

  Dentro de esta fase de construcción, iterativa e incrementalmente se desarrollo el producto completo, y a la finalización de esta se obtuvo un producto preparado para la transición de ser un proyecto a un producto empleado en un entorno real de trabajo, con usuarios reales, lo cual implica describir los casos de usos restantes, junto con la complementación

de la implementación y las pruebas al producto creado. En esta fase se contempla desde el modelado iterativo de todo el sistema, así como la codificación de este modelo.

Al finalizar esta fase, se decidió que el software, el entorno y los usuarios ya estaban preparados para el inicio de sus operaciones.

- **Fase de Transición**:

  Durante la fase de transición, se trasladó el software al entorno real de trabajo y una vez que el producto fue puesto en manos de usuarios, los cuales generaron reportes para realizar desarrollos adicionales para corregir problemas no detectados en la fase de pruebas y refinar detalles que pueden quedar en el proceso de la construcción.

  Al terminar esta fase se debe decidir si los objetivos del ciclo de vida fueron alcanzados y en que medida, este punto concluye con puntos específicos con los cuales se tiene una base de cuales serían los trabajos a futuro en este sistema o en algún otro desarrollo que se incorpore de alguna manera con este proyecto.

2. 4 Análisis de Requerimientos

Los requerimientos de un sistema son la pieza fundamental para el desarrollo, sobre ellos se basa la planeación del proyecto y los recursos que se emplearán en el mismo, las verificaciones que se realizarán al final de su creación, la planeación de las estrategias de pruebas a las que será sometido el sistema y son el fundamento para el ciclo de vida del proyecto, estos son la base fundamental para crear la documentación del sistema.

Estos requerimientos deben estar adecuadamente formulados, de manera que sea posible realizar una comprobación o verificación de ellos al obtener el producto final, a su vez, estos deben de describir la característica especifica que el sistema debe tener y deben de ser lo mas abstractos y concisos posible.

Estos requerimientos se deben de enfocar a describir las necesidades del cliente, en nuestro caso las personas que laboran en el área de Ingeniería Asistida por Computadora, por lo cual estos se tuvieron que obtener de una persona que conociera el funcionamiento del área y sus necesidades.

Requerimientos

Dentro de los requerimientos que se tienen para realizar el diseño de este sistema se tienen:

- Implementación de un repositorio centralizado para los documentos electrónicos que se realizan en el área.
- Acceso a este repositorio mediante cualquier equipo de computo conectado a la Intranet del Instituto Mexicano del Petróleo.
- Posibilidad de realizar búsquedas de documentos por distintas categorías propuestas.
- Restricciones de acceso a diferentes roles de usuarios, así como a grupos de usuarios y categorías.

- Notificación sobre actualizaciones de documentos, así como actualizaciones a los datos de cualquier usuario.
- Descripción previa de los documentos contenidos en el repositorio.
- Navegación sencilla e intuitiva sobre la organización de los documentos, por medio de un sistema que emule un sistema de archivos genérico compuesto de carpetas y archivos organizados jerárquicamente.
- Capacidad de administración de los usuarios que accedan al sistema, así como del contenido.
- Control de versiones

2.5 Casos de Uso del sistema

Los casos de uso representan la forma en como un Cliente (Actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en como los elementos interactúan (operaciones o casos de uso).

Actores

Un actor es un papel o rol que un usuario o un agente externo juega con respecto al sistema. Cuando se dice que un actor representa un papel, nos referimos a que este tiene un conjunto de capacidades asociadas.

En el sistema de Administración de documentos se toman en cuenta cuatro tipos de roles diferentes de usuarios, estos diferentes roles, estos son:

- Administrador
- Moderador
- Publicador
- Lector

Estos roles indican ciertas operaciones que el usuario que este bajo cierto rol puede realizar, estas se muestran a continuación:

Lector

Este rol de usuario describe a usuarios del sistema que tienen acceso a los documentos del grupo al cual pertenecen, pueden observar y descargar los documentos que están bajo los directorios que corresponden a su grupo. Así mismo tienen capacidad de modificar sus datos de acceso e identificación en el sistema, siendo notificados de esto por medio de correo electrónico, así como de actualizaciones de documentos pertenecientes a su grupo y que el publicador, moderador o administrador haya decidido notificar.

Publicador

Este rol, tiene las mismas características que el lector, pero además de descargar documentos, éste también puede cargar documentos sobre el servidor y decidir si desea notificar a los miembros de su grupo de la carga de documentos nuevos o nuevas versiones de un documento existente.

Moderador

Este rol además de contener las características de Publicador, tiene la capacidad de modificar el contenido y la información de las carpetas y archivos que pertenecen al grupo al que esta inscrito. Este rol tiene la función de controlar el contenido de los documentos que se publican en el espacio dedicado a su grupo.

Administrador

El usuario que se encuentra bajo este rol, como lo indica su nombre, se encarga de llevar la administración de todo el sistema, el contenido de éste y de sus usuarios. En cuanto al sistema, este tipo de usuario puede cargar y descargar documentos en cualquier ubicación, no importando el grupo ni la carpeta, pues tiene acceso absoluto al sistema. Respecto al contenido, un usuario bajo este rol puede modificar y eliminar cualquier archivo, versión, o carpeta en cualquier lugar. En cuanto a los usuarios, tiene la capacidad de crear nuevos usuarios y grupos, así como negar el acceso a cierto usuario por medio de una opción que bloquea al usuario en cuestión.

2.6 Diagramas de casos de uso

A continuación se muestran los casos de uso del sistema, un diagrama por cada actor para evitar confusión.

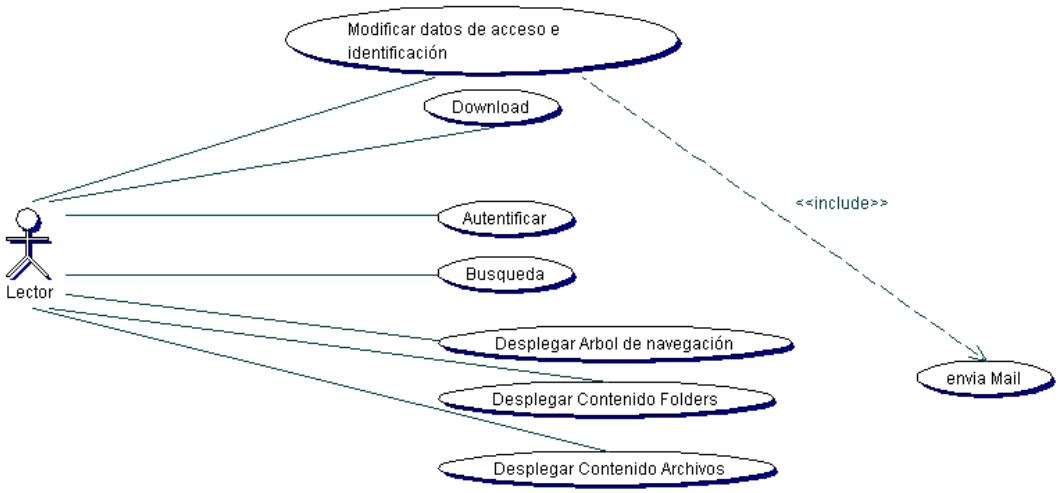

Fig. C2-2. Casos de Uso correspondientes al actor Lector

En la figura C2-2 se muestran los distintos casos de uso del actor Lector, el actor que tiene relación con el menor número de casos de uso dentro del sistema, pues sus funciones son restringidas únicamente a descargar archivos que pertenezcan a su grupo, así como las funcionalidades que tiene como usuario del sistema las cuales consisten en la modificación de sus datos de acceso e identificación dentro del sistema así como su autentificación dentro del mismo. Estos casos de uso se describen a continuación.

Autentificar
Para ingresar al sistema es necesario que los usuarios proporcionen un nombre de usuario y contraseña. Además es necesario que se identifiquen los usuarios para distinguir entre los cuatro tipos de usuario, cual es el que le corresponde.

Búsqueda
Búsqueda genérica dentro de los datos contenidos en las tablas de la base de datos, las cuales pueden ser para localizar archivos y/o carpetas de acuerdo a las descripciones de éstas.

Desplegar Árbol navegación
Despliegue del árbol de navegación para facilitar la navegación

Desplegar contenido archivos
Despliegue de los datos correspondientes a los archivos y sus versiones, como fecha de publicación, usuario que publico, tamaño, y tipo de archivo o versión del archivo.

Desplegar contenido fólder
Despliegue de los datos correspondientes a los folders, fecha de creación, usuario creador, archivos y carpetas contenidos, etc.

Download
Carga de los archivos contenidos en el sistema.

Envía mail
Envió de información a los usuarios por medio de correo electrónico, el usuario no interactúa directamente con este caso de uso, sino que es utilizado por otros casos de uso.

Modificar datos de acceso e identificación
Modificación de los datos de identificación como nombre, correo electrónico, número de gafete, etc, así como cambio de contraseña.

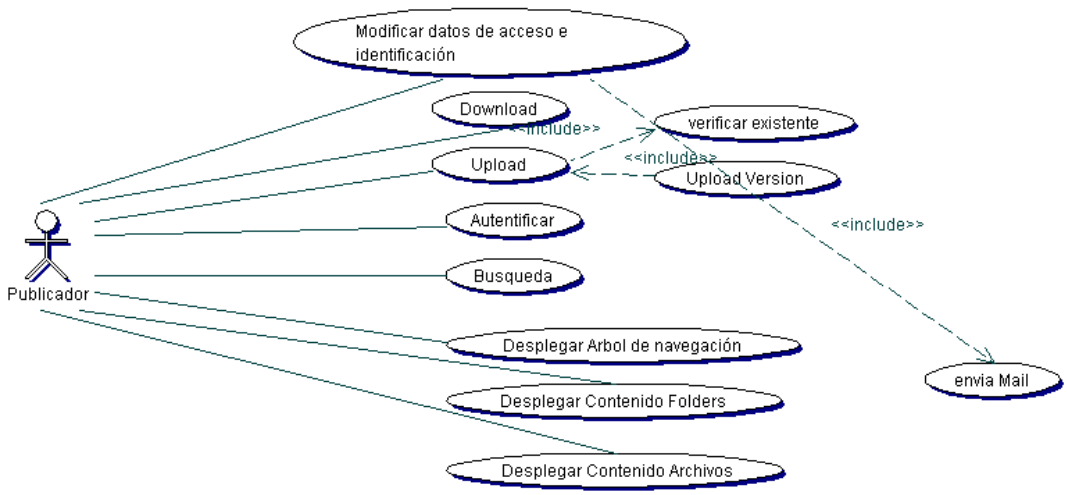

Fig. C2-3 Casos de Uso correspondientes al actor Publicador

En la figura anterior(C2-3), se muestra el diagrama de casos de uso del actor publicador, el cual tiene menos restricciones sobre el sistema que el actor lector. Este actor interactúa con los mismos casos de uso que el lector, mas el caso de uso Upload, que a su vez incluye al caso de uso verificar existente y Upload versión. A continuación se muestra una descripción de estos casos de uso.

Upload
Carga de archivos al servidor, o bien de nuevas versiones si existiera un archivo con el mismo nombre del que se desea subir sobre ese folder.

Upload Versión
Carga de versiones de un archivo en caso de que exista un archivo con el mismo nombre sobre el mismo folder.

Verificar archivo existente
Verifica si un archivo ya esta contenido dentro de esa carpeta.

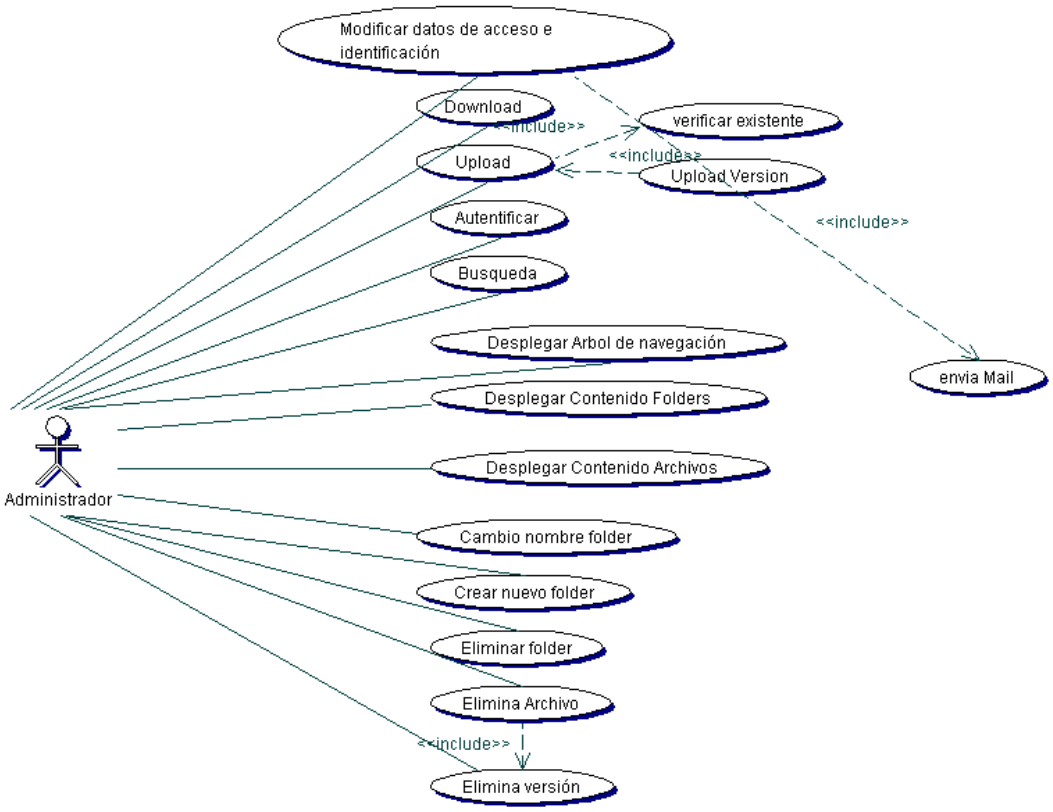

Fig. C2-4 Casos de Uso correspondientes al actor Moderador

En esta figura(C2-4), se despliega el diagrama de casos de uso del actor Moderador, el cual contiene algunos casos de uso que representan la condición de este actor sobre los actores lector y publicador. De la misma manera, se muestra una descripción breve de los casos de uso con los cuales este actor interactúa y que no se han mencionado en los actores pasados.

Cambio nombre de fólder
Renombrar el folder en el cual esta el usuario ubicado

Crear nuevo folder
Creación de un nuevo folder bajo el grupo del folder padre.

Elimina folder
Eliminación de folders y su contenido(folders hijos, archivos y versiones)

Elimina archivo
Eliminación de un archivo con las posibles versiones que contenga

Elimina versión
Eliminación de determinada versión de un archivo registrado en el sistema.

En este diagrama(C2-5) se muestran todos los casos de uso posibles en el sistema, a los cuales tiene acceso directo en su mayoría, el actor administrador, a continuación se enlistan los casos de uso restantes y que son exclusivos de este actor.

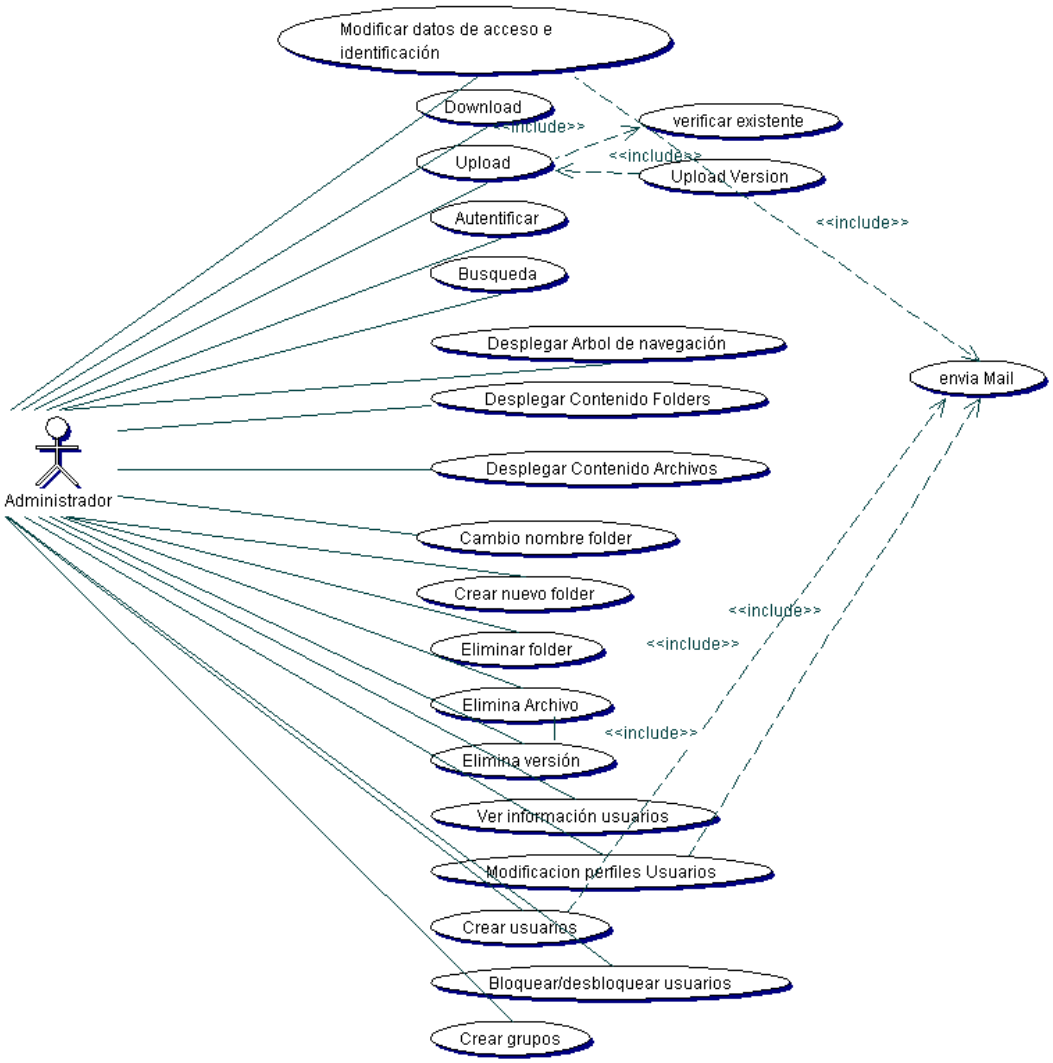

Fig. C2-5. Casos de Uso correspondientes al actor Administrador.

Bloquear o desbloquear usuarios
Deshabilitar la posibilidad de ingresar al sistema a cualquier usuario por las razones que convengan, sin eliminar a los usuarios, así mismo permitir el ingreso nuevamente, después de haber sido bloqueado.

Crea usuario
Creación de nuevas cuentas de usuario, las cuales tendrán acceso al sistema, asignación de nombre de usuario, contraseña, perfil y notificación por e-mail del alta de su cuenta.

Crear grupos
Creación de nuevos grupos de usuarios

Modifica perfiles de usuarios
Modificación de perfiles de los usuarios del sistema, cambio de grupo y cambio de permisos.

2.7 Recursos Utilizados

Dentro de este punto se consideran los recursos que se proponen para al realización del proyecto.

Recursos de Software

Cabe destacar que el software para la creación de este proyecto, se ha elegido principalmente software de libre distribución en Internet(freeware), con el fin de evitar la compra de licencias o recursos que solo aumentan el costo del proyecto y no así su calidad.

Los recursos de software a utilizar se muestran a continuación:

- Java Developpement Kit(JDK) 1.4.0[jdk14] como entorno de desarrollo en lenguaje Java de Sun Microsystems.
- Servidor de base de datos relacional MySQL 3.23.49[myslq] para Linux x86.
- Contenedor de Servlets y JSP Tomcat 4.0 del proyecto Jakarta de Apache.
- API's varias: Java Mail[jmail], procesamiento(DOM[dom], SAX[sax]) y conversión(XSLT[xslt]) de XML[xml], o'reilly (MultipartDataRequest), JDBC-MySQL [jdbc], etc.
- Sistema operativo Red Hat Linux 7.2[rhlin] y utilerías(vi, pico, gedit, Emacs, etc.).
- Herramientas de Modelado de software ArgoUML y Together.

Recursos de Hardware

Dentro de los recursos que se recomiendan utilizar, se comprenden dos tipos, los empleados en el desarrollo y los propuestos para la utilización de la aplicación trabajando con usuarios reales:

Desarrollo:

- Computadora Personal(PC) con procesador Pentium II a 256Mhz o mayor, 256Mb en RAM.

Utilización real:

- Computadora Personal(PC) con procesador Pentium IV a 1.3GHz o mayor, 512 Mb en RAM

Recursos Documentales

- Documentación sobre los recursos de software a utilizar: Java, JSP, Servlets, JDBC, API's varias, Linux, MySQL, SQL, Apache http Server. Ya sea en tutoriales obtenidos sin costo en Internet o bibliografía especializada en los temas mencionados, si se desea mas información referirse a la sección de bibliografía.

2.8 Tecnología a emplear

A continuación se muestra las diferentes tecnologías propuestas para utilizarse dentro de la realización del proyecto.

Programación orientada a objetos

La programación orientada a objetos[poo] es la base de los actuales métodos de programación, y cada día son utilizados con mayor frecuencia, pues la portabilidad, la eficiencia y velocidad de desarrollo que se muestra al utilizar estas tecnologías lo hacen mucho mas atractivo que otras tecnologías para el desarrollo de software. Como su nombre lo indica, esta basado en la utilización de objetos, que se pueden definir como un conjunto complejo de datos y programas que poseen estructura y forman parte de una organización. Tal organización permite que estos objetos a su vez se agrupen en clases dentro de las cuales existen relaciones de diferentes tipos entre los objetos que tiene un comportamiento similar. Los lenguajes orientados a objetos deben cumplir con tres características básicas, estas son: 1. Debe estar basado en objetos, 2. Basados en clases y 3. Capaz de tener herencia de clases.

Java

Java es un lenguaje de programación orientada a objetos con el cual se pueden realizar todo tipo de desarrollos el cual se ha extendido en la actualidad y cada día cobra mayor importancia en el mundo de la programación ya que cumple con las características básicas y mucho mas que se requieren para considerársele como un Lenguaje orientado a objetos.

Esta desarrollado por Sun Microsystems y siempre ha estado enfocado a cubrir las necesidades tecnológicas de ultimo tiempo. Una de las principales características por las que Java se ha desarrollado tanto en cuanto a utilización y mejoras es que es un lenguaje independiente de la plataforma, además de ser de distribución libre. Esto nos indica que al realizar algún desarrollo con Java, no existirá la limitante de la plataforma y éste podrá funcionar en cualquier sistema operativo o equipo en el mercado. Esto lo consigue debido a que se ha creado una Máquina virtual para cada plataforma, la cual actúa como puente entre el sistema Operativo y los programas.

Además, últimamente Java esta incursionando con grandes pasos al campo del desarrollo de aplicaciones para dispositivos móviles, lo cual augura un futuro prometedor para el desarrollo de aplicaciones sobre este lenguaje orientado a objetos.

La aplicación a desarrollar estará basada totalmente en el lenguaje orientado a objetos Java, debido a su optimo desempeño en cualquier tipo de aplicaciones y su amplio desarrollo para la creación de aplicaciones web.

Tecnologías Web basadas en Java(JSP, Servlets, JavaBeans)

Las tecnologías empleadas en web para Java, son básicamente tres y son los Java Servlets, las páginas JSP y los JavaBeans. Por su parte los Servlets[servl] nos brindan una excelente forma de manejar las peticiones que provienen de un navegador Web hacia un servidor http, así como las respuestas de dicho servidor, mediante el empleo de lenguaje Java. Estas estructuras se emplean dentro del proyecto para tareas como la carga de parámetros del servidor, así como controlar el direccionamiento de paginas.

La tecnología JSP[jsp], debido a sus características, permite la combinación de código HTML estático con un contenido dinámico de una manera mucho más rápida y sencilla que en un Servlet. Estas, aunque tiene una apariencia de HTML sencillo, al cargarse al servidor Web, se convierten automáticamente en Servlets. En el desarrollo de esta aplicación, las páginas JSP, se utilizan principalmente para el despliegue de las interfaces gráficas al usuario.

El formato de las clases construidas de acuerdo al estándar de los JavaBeans[jb] permiten un descubrimiento automático de las propiedades y la información que estas manejan, de la misma manera, esto permite que los componentes creados de esta manera, sean completamente reutilizables e ideales para manejar la lógica de negocio en las aplicaciones Web, como es el caso de esta aplicación. El uso de esta tecnología permitirá en un futuro un mejor mantenimiento y una fácil extensión de las utilidades de la aplicación.

XML

XML desde su creación en 1998, ha revolucionado la manera de estructurar, describir y compartir la información. La idea de los creadores del XML, fue la de crear un lenguaje muy general que sirva para utilizarse para muchas cosas y por muchas cosas, así como para ser transportado por cualquier medio. La potencia de la forma de trabajar del XML se basa en que se identifica y etiqueta el contenido olvidándose de la forma de presentarlo, pues existen herramientas y medios muy potentes para presentarlo como se desee. Dentro de este proyecto, se hace uso de la generación y transformación de código XML para generar la interfaz de navegación.

2.9 Arquitectura del Sistema de Administración de Documentos

Definir la arquitectura del sistema, es un punto muy importante en el desarrollo, ya que es en base a esto la forma en como se realizará el desarrollo, las actividades de diseño están centradas alrededor de la noción de la arquitectura del software determinando que instancias deben de realizar que trabajos y la manera en que éstas interactuarán. En este proyecto de tesis, se toma como referencia la arquitectura multinivel de 3 capas[tresc].

Modelo de 3 capas

La arquitectura típica en que se basa en el modelo de 3 capas es: presentación, lógica de negocio y datos.

Presentación

La capa de presentación permite al usuario visualizar y acceder al sistema a través de una interfaz de usuario. En un sistema web esta capa está compuesta por distintas páginas web que presentan los datos en un navegador (Internet Explorer, Nestcape Navigator, Opera, etc) principalmente, en este caso la presentación se realiza por medio de páginas JSP.

Lógica

En esta capa se encuentran las reglas y lógica de procedimientos necesarios para realizar las operaciones del sistema. Esta capa interactúa entre la base de datos o algún otro servicio(mail, LDAP, etc), y la presentación ante algún requerimiento de búsqueda o acceso de datos por parte de los usuario. Para el desarrollo de aplicaciones en tres capas sobre java, esta capa se realiza mediante el empleo de la tecnología de JavaBeans, debido a sus propiedades de acceso a datos y fácil manipulación y acceso por parte de las interfaces gráficas.

Datos

Esta capa es en donde se encuentran los datos relacionados con el sistema, definidos en una base de datos con sus respectivas tablas y registros, los cuales pueden ser accedidos y manipulados por los usuarios. De la misma manera, en esta capa se encuentran los posibles servicios a los que puede acceder la lógica de negocio, tales como e-mail, LDAP, dominios NT, dominios UNIX, etc.

Diagrama de sistema propuesto en 3 capas

El siguiente diagrama muestra la arquitectura de 3 capas en la que estará basado el sistema de Administración de Documentos.

Fig. C2-6 Arquitectura de 3 Capas para el Sistema de Administración de documentos y planos CAD para el IMP

MVC

Muy ligado a la arquitectura de tres capas, se encuentra el Modelo Vista Controlador(MVC)[mvc] debido a su enfoque de componentes en que esta basada la tecnología de tres capas que se ha ido adquiriendo en los últimos años dentro del desarrollo de aplicaciones web. Este patrón MVC se implementa en la capa de presentación, y el objetivo del MVC es separar la presentación de la lógica de negocio.

Este patrón descompone una aplicación interactiva en tres grandes bloques:

El modelo, contiene los datos y la funcionalidad de la aplicación, además de ser independiente de la representación de los datos.

Las vistas, muestran la información al usuario de una cierta forma.

El controlador asociado, que generalmente es un Servlet, recibe como entrada las peticiones de los usuarios y determina la vista a usar para presentar los resultados.

El navegador del usuario, en su petición http, genera una solicitud que es atendida por el controlador. Éste analiza sus entradas y llama a los objetos correspondientes del Modelo. El Modelo se encarga de ejecutar la solicitud y generar los resultados que mostrarán posteriormente las Vistas.

Según el resultado que retorne el Modelo, el Controlador derivará la generación de la página interfaz a una o más JSP (Vistas), que podrán consultar los Modelos con el fin de realizar su tarea adecuadamente

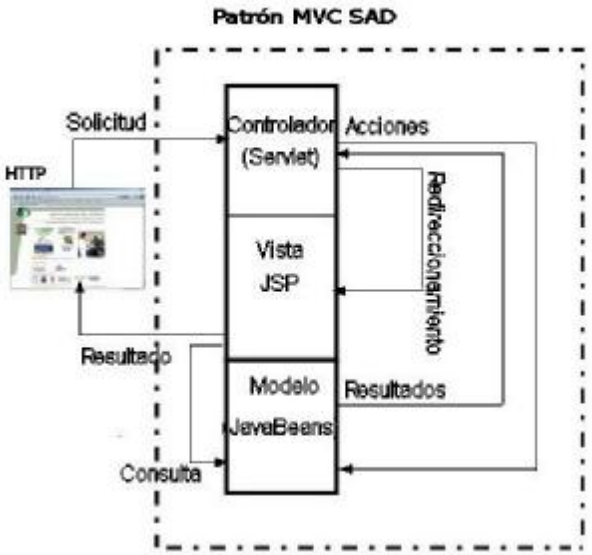

Fig. C2-7 Esquema del patrón MVC para el SAD

En la figura C2-7 se muestra gráficamente las interacciones entre los tres distintos bloques que componen el patrón MVC, en éste se puede observar claramente que el cliente http, al realizar una solicitud o petición se dirige directamente al controlador, el cual interactúa con el Modelo para obtener ciertos resultados con los cuales redireccionará a una página de vista(JSP), la cual puede consultar datos al bloque de modelo y posteriormente enviar el resultado pedido por el cliente http.

Hasta este punto de este trabajo de tesis, se han obtenido los antecedentes, la problemática a resolver, así como un análisis de ésta. Como resultado de este análisis se ha propuesto una solución al problema basado en la deficiencia del servicio de Administración de Documentos del área de Ingeniería Asistida por computadora y en general de la subdirección de Ingeniería del IMP. Solución que se basa en el empleo de tecnologías Web siguiendo los requerimientos pedidos, bajo una arquitectura de tres capas, que ya se definió dentro de éste capítulo.

El siguiente paso es realizar el diseño del Sistema de Administración de Documentos, el cual consiste en generar el modelado de los datos que se manejaran dentro de el sistema como parte de la capa de datos. Por otra parte, se tendrá que realizar el diseño de la lógica del sistema mediante el diseño de las clases, y finalmente el diseño de la interfaz de usuario.

CAPÍTULO

3

Diseño del Sistema de Administración de Docum

Después del el análisis de los requerimientos que se ha realizado, es necesaria una etapa de diseño del Sistema de Administración de Documentos para poder alcanzar de manera integral los objetivos de éste. Esta etapa esta dividida en cuatro fases y el objetivo de éste capítulo es desarrollar la manera en que se desarrollan.

Como se observará posteriormente, el análisis junto con el análisis de requerimientos son las etapas mas decisivas en cuanto a toma de decisiones y de éstas depende el éxito o fracaso en la creación de software, así como la dificultad que se tendrá cuando se deseen realizar cambios o mantenimiento al sistema.

3.1 Esquema general de diseño

El diseño del sistema de Administración de Documentos y planos, es el primer paso de la fase de desarrollo, el objetivo de esto es reproducir un modelo o representación de la entidad que se construirá posteriormente sobre la base de la traducción de los requerimientos ya obtenidos en el capítulo de análisis. El diseño es el proceso en el que se establece la calidad del desarrollo del software y sirve como base de la codificación, prueba y mantenimiento, sin éste, el sistema a construir podría ser inestable y con pocas posibilidades de realizar cambios.

La metodología de diseño utilizada se basa en cuatro elementos necesarios para realizarlo. Estos son el Diseño de Datos, el Diseño de paquetes o estructura del sistema, el diseño de clases de objetos, y de interacción entre objetos y finalmente el diseño de la interfaz gráfica.

3.2 Diseño de la base de datos

Sobre la base de los requerimientos y el análisis, se diseño la base de datos con la ayuda de herramientas CASE que permiten realizar diagramas entidad-relación(E-R) con lo cual se describe de manera gráfica las relaciones entre los datos.

Los diagramas principales para llevar a cabo el análisis y diseño del sistema. Presentan las clases del sistema con sus relaciones estructurales y de herencia, forman parte de la vista estática del sistema y en éstos se definen las características de cada una de las clases, colaboraciones, relaciones de dependencia y generalización.

Los diagramas de secuencias representan la interacción que existe entre los objetos y clases diseñadas a través del diagrama de clases, de la misma manera se representa como y en que momento de la ejecución del software va a existir cierta relación entre los distintos objetos y clases creados.

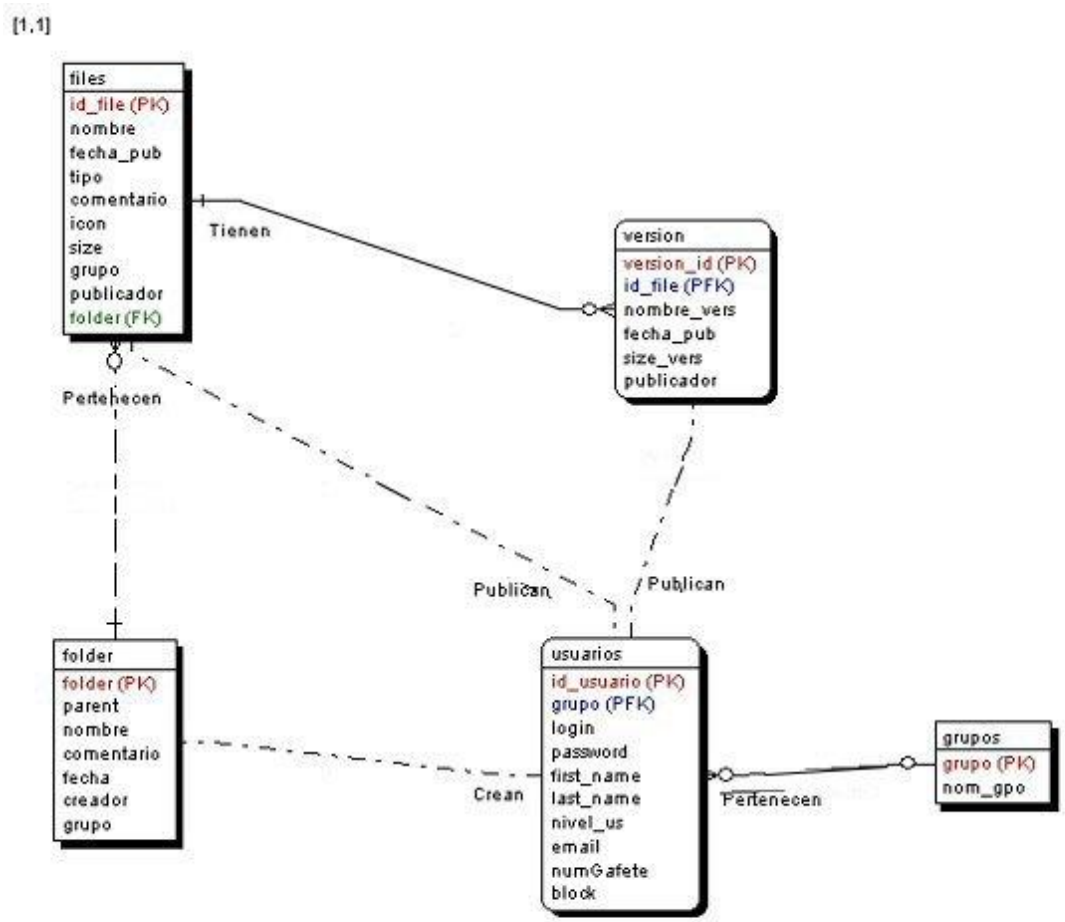

Fig. C3-1 Diagrama entidad-relación

En la figura C3-1, se muestran las cinco tablas que se utilizan en el Sistema de Administración de documentos, así como las relaciones que existen entre ellas. A continuación se da una descripción breve de su contenido, así como de su utilidad dentro del SAD.

Entidad: files

En esta tabla se almacenan los datos de los archivos que se publican dentro del sistema. El nombre, tipo, fecha en que fue publicado, tamaño, icono representativo y folder al que pertenece son los datos necesarios para una identificación total del archivo.

Entidad: folder

Se almacenan aquí los datos de los folders que sirven como contenedores de los distintos archivos y versiones que se publican en el sistema. Se almacena el nombre, una descripción breve del folder, la fecha de creación, la persona que la crea, el grupo al que pertenece, y finalmente el identificador del folder padre de éste. Este ultimo atributo es necesario para la creación de la interfaz de navegación, la cual consiste en un árbol jerárquico de navegación del cual se tendrá mas información posteriormente.

Entidad: version

La tabla versión, almacena la información nuevas versiones de los archivos que se publican en el SAD, su identificador de versión, archivo asociado, nombre de dicha versión, fecha de publicación, y persona que la publicó.

Entidad: Usuarios

Contiene toda la información necesaria de los usuarios que tienen la posibilidad de ingresar al sistema. Esta información se compone de un identificador de usuario, su nombre de usuario, contraseña, nombre y apellidos con el fin de identificar a las personas que publican dentro del sistema, el perfil del usuario, grupo al que pertenece, correo electrónico para hacer notificaciones, número de gafete del empleado y un atributo que funciona como bandera para indicar si este usuario tiene acceso al sistema o bien ha sido deshabilitado por algún administrador.

Entidad: Grupos

Para tener una clasificación de los usuarios de acuerdo a su proyecto, división, etc., es necesario agruparlos. Dentro de esta tabla se tienen un identificador por grupo, así como su nombre.

3.3 Diseño de la estructura del sistema

El objetivo general del diseño del sistema es generar un modelo o representación técnica del software que se desea y sobre el cual se sienta la calidad del producto final. Para lograr este objetivo es preciso crear una estructura del sistema mediante el uso de estándares de diseño que enfoquen de manera completa los procesos que se desarrollarán.

Dentro de este proceso de diseño de estructura del SAD, se toman en cuenta básicamente tres tipos de diagramas que están basados en el estándar UML(Unified Modeling Lenguage), el primero es el diagrama de paquetes. En estos diagramas se representan agrupaciones de clases que tienen un fin común llamados paquetes y aunque UML no tiene reglas especificas sobre la composición de éstos, las clases contenidas en cierto paquete deben de tener una funcionalidad común dentro de la estructura del sistema. En el caso del SAD, se cuenta con un diseño compuesto por cinco paquetes(auth, arbol, folder, utils y users) en los cuales se ha subdividido la funcionalidad del sistema.

3.3.1. Diagrama de paquetes

En el diseño del nuestro sistema, los paquetes ofrecen un mecanismo general para llevar a cabo la organización de los subsistemas agrupando a los elementos del modelado. Estos paquetes corresponden a un submodelo del sistema.

En el siguiente diagrama de paquetes se muestran gráficamente los cinco paquetes con que cuenta el sistema, cuya funcionalidad se describirá posteriormente.

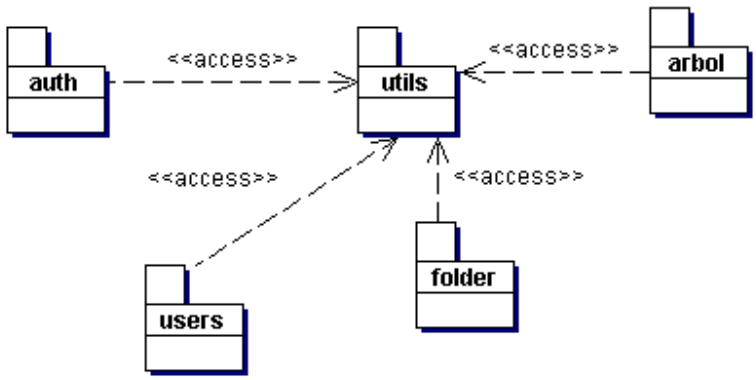

Fig. C3-2 Diagrama de paquetes del SAD

Los cinco paquetes con que cuenta el SAD y su funcionalidad son:

- auth

  Este paquete contiene clases que básicamente tienen la funcionalidad de la autentificación de usuarios, mediante éstas clases se accede al nivel de datos para obtener la información requerida para tener acceso al sistema(nombre de usuario y contraseña).

- users

  En este paquete se lleva toda la funcionalidad que facilita al administrador realizar el control de los usuarios que se encuentran dados de alta en el sistema, así como realizar altas de nuevos usuarios, bloquear usuarios, etc. De la misma manera dentro de este paquete se pueden realizar las validaciones respecto al grupo al que pertenece cada usuario, así como el manejo de los grupos(creación, eliminación, etc).

- folder

  Dentro del paquete folder se realiza toda la funcionalidad de las modificaciones, publicaciones y despliegue al usuario final de los archivos y carpetas contenidas dentro del sistema.

- arbol

  Mediante el acceso a la base de datos que contiene la información de los diferentes folders y sus archivos contenidos, se realizan archivos xml los cuales después de un procesamiento xslt forman un árbol de navegación que permite tener una navegación entre los folders, más intuitiva y fácil.

- utils

Este paquete contiene clases que realizan tareas complementarias de los demás paquetes tales como las conexiones con la base de datos y el envío de correo electrónico para realizar las notificaciones a los usuarios entre algunas y otras más sencillas.

3.4 Diagramas de clases

Como ya se dijo, éste tipo de diagramas es la base del diseño del sistema pues se muestran las interacciones entre las diferentes entidades que componen el sistema. A continuación se presentan estos diagramas de clases divididos según el paquete al que pertenecen, es decir tenemos cinco diagramas de clases los cuales están relacionados entre sí según lo muestra el diagrama de paquetes.

3.4.1 Diagrama de clases del paquete "auth"

Como ya se dijo, en este paquete se realizan las tareas de autentificación para el acceso al sistema de administración de documentos por medio del uso del Servicio de Autentificación y Autorización en Java(JAAS)[jaas], este paquete esta compuesto por tres objetos donde se realiza la lógica de negocio, *ControlAuth*, *PassiveCallbackHandler* y *RdbmsLoginModule*, dos clases que se han implementado para obtener objetos del tipo *Principal* en *RdbmsPrincipal* y *Credential* en *RdbmsCredential*, que conforman propiedades específicas del paquete java.security, donde se insertan los permisos y en general cualquier información referente a los usuarios que logran una autentificación exitosa en el sistema mediante el servicio JAAS, y que además se integran al objeto *LoginContext* que aparece en el diagrama de clases de este paquete.

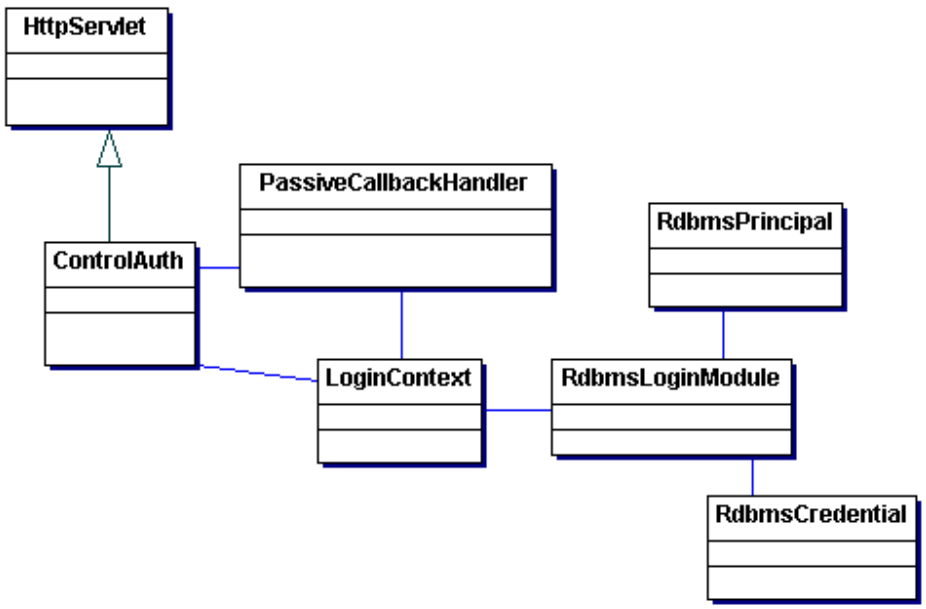

Fig. C3-3 Diagrama de clases del paquete auth

Objeto ControlAuth

Encargado de realizar el control entre la interfaz gráfica y la lógica de negocio contenida en las clases que realizan la autentificación JAAS. Consiste de un Servlet, el cual recibe por parte el cliente web las peticiones y éste redirecciona a la pagina JSP conveniente con las propiedades del usuario que están contenidas en la base de datos, las cuales le dan ciertos privilegios dependiendo del nivel de usuario y grupo al que pertenezca. Dentro de este Servlet se crea un objeto *PassiveCallbackHandler*, el cual se utiliza junto a las propiedades contenidas en el archivo <root-webapp>/WEB-INF/jaas.config para crear al objeto *LoginContext*.

Objeto RdbmsLoginModule

Esta clase realiza la lógica de negocio de autentificación de los usuarios con respecto a los datos contenidos en la base de datos, creando objetos *Credential* y *Principal* a partir de los datos asociados a los usuarios contenidos en dicha base de datos. Este objeto es llamado indirectamente para crear un objeto *LoginContext* en el Servlet controlador *ControlAuth*. Para crear dicho objeto se tienen que recibir dos parámetros, el primero es la propiedad que se obtiene del archivo jaas.config y que contiene los datos de acceso a la base de datos que contiene ésta información, así como el nombre y paquete de la clase que realizará dicho proceso, que en el este caso es el objeto *RdbmsLoginModule*.

Objeto PassiveCallbackHandler

Recibe como parámetros el nombre de usuario y contraseña de los usuarios, para crear un objeto *CallbackHandler* usado para la construcción del objeto loginContext en el Servlet controlador.

Objeto RdbmsCredential y Objeto RdbmsPrincipal

Son objetos *Credential* y *Principal* a los cuales se les asignan las propiedades asociadas a los usuarios autentificados y que resultan útiles pues de dichos objetos se obtienen todos los datos del usuario autentificado.

3.4.2 Diagrama de clases del paquete “users”

A través de este paquete se obtiene la información referente a los usuarios que están activos en el sistema, así como la inserción de éstos, y algunas otras modificaciones relacionadas con el manejo de usuarios.

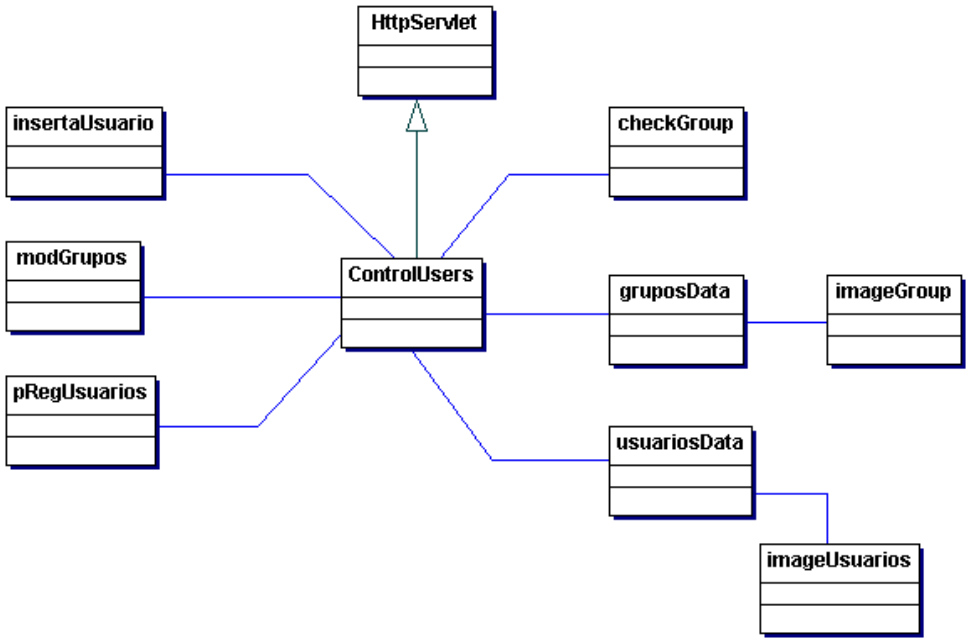

Fig. C3-4 Diagrama de clases del paquete users

Objeto ControlUsers

Para realizar los procesos referentes al manejo de usuarios siguiendo la estructura MVC, es necesario contar con un controlador. Este objeto al igual que el controlador de autentificación extiende de la clase HttpServlet y recibe las peticiones del navegador web para cargar propiedades y redireccionar a la página JSP correspondiente.

Objeto insertaUsuario

Este objeto se encarga de la mayoría de las operaciones referentes a la inserción y modificación de propiedades de los usuarios que se encuentran dados de alta en el sistema, cuenta con métodos para insertar usuarios, actualizar información de los usuarios, bloquear y desbloquear usuarios, modificar permisos de usuarios y cambios de grupo.

Objeto modGrupos

Al clasificar a los usuarios por grupos, es necesaria la creación de nuevos grupos y de la misma manera, en caso que un grupo deje de tener uso, puede eliminarse para evitar acceso a información que ya no es actual. Este objeto contiene métodos con los cuales se pueden hacer modificaciones sobre los grupos.

Objeto pRegUsuarios

Este se encarga de las validaciones del lado del servidor en el momento de dar de alta a nuevos usuarios, o bien en la actualización de los datos de los usuarios.

Objeto checkGroup

Este objeto es utilizado para la validación de acceso de los usuarios a las carpetas y/o archivos, con este mecanismo únicamente los usuarios que pertenecen a cierto grupo pueden acceder a las carpetas que pertenecen a este grupo.

Objeto gruposData

Obtiene información sobre los grupos de usuarios, como nombres, lista de usuarios pertenecientes al mismo.

Objeto imageGrupos

Es una clase espejo de la tabla grupos de la base de datos y es instanciada desde el objeto gruposData con el fin de colocar la información obtenida sobre un objeto más transportable.

Objeto usuariosData

Obtiene información de los usuarios, los datos personales, de acceso, privilegios en el sistema así como los archivos y versiones que han publicado los usuarios.

Objeto imageUsuarios

Tiene el mismo fin que el objeto imageGrupos, pero con la diferencia de que es la clase espejo de la tabla usuarios de la base de datos.

3.4.3 Diagrama de clases del paquete “arbol”

En este paquete se encarga de la lógica para despliegue del árbol de navegación el cual facilita el acceso a los diferentes folders y archivos contenidos y ordenados jerárquicamente. Para realizar este proceso, se dividió el proceso en dos partes. La primera la realiza la clase *ConsTree* junto con *OutilTree*, las cuales generan un archivo XML a partir de los datos contenidos en la tabla *folders* y *files,* esto con el fin de utilizar las tecnologías que han surgido últimamente de XML y XSLT para separar la lógica de las aplicaciones y la presentación. Por otra parte la clase *Cooltree* se encarga de generar a partir del árbol jerárquico generado con XML, archivos .js(javaScript) los cuales contienen un script que forma el árbol gráficamente en el navegador web. Este proceso se describirá posteriormente con mas detalle, en la implementación del sistema.

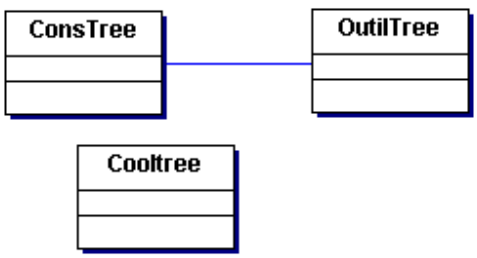

Fig. C3-5 Diagrama de clases del paquete arbol

Objeto ConsTree

Obtiene de la base de datos la información para crear un archivo XML, instanciando a *OutilTree*. Es capaz de formar un archivo XML bien formado, el cual se procesa posteriormente con XSLT para crear el javaScript para generar la interfaz gráfica del árbol de navegación.

Objeto OutilTree

Sus métodos crean la estructura de un archivo XML, las etiquetas de apertura y cerradura de cada nodo del archivo XML.

Objeto Cooltree

Se encarga de la transformación del archivo XML, para generar archivos .js(javaScript)[js] con una estructura prediseñada la cual despliega el arbol de navegación en el navegador web. Este objeto recibe como entradas el archivo .xml que se desea procesar, y una hoja de estilo .xls con las reglas de transformación que se le aplicarán.

3.4.4 Diagrama de clases del paquete "folder"

Como se mencionó, este paquete es el encargado del cometido central de la aplicación, es decir la publicación de los archivos. Contiene objetos que permiten la visualización de los contenidos de los folders, así como la publicación de archivos, creación y modificación de folders y el manejo de las versiones de cada archivo que se publica en el sistema. A continuación se describen las clases contenidas dentro de este paquete.

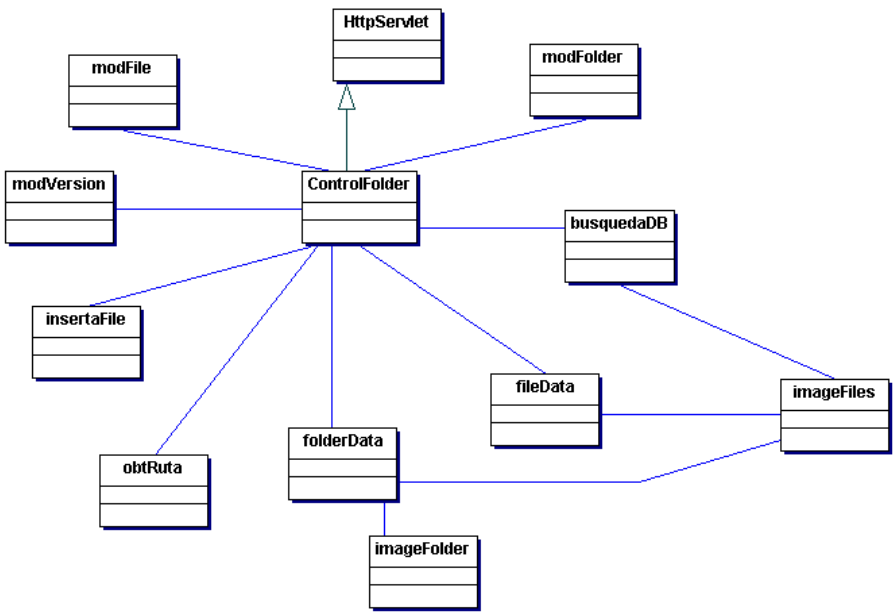

Fig. C3-6 Diagrama de clases del paquete folder

Objeto ControlFolder

Extiende de la clase *HttpServlet* y esta encargado de llevar la lógica requerida para el redireccionamiento de las paginas JSP de acuerdo a la peticiones que realice el cliente http o navegador. Es la parte que realiza las funciones de controlador en el esquema modelo-vista-controlador.

Objeto modFolder

Sus métodos permiten la creación, borrado y renombrado de los folder del sistema de archivos que se maneja dentro del sistema, estas modificaciones se hacen sobre la base de datos y no sobre los datos binarios del servidor, lo que permite la recuperación de archivos que se han borrado en el sistema. Por lo que se tiene siempre un respaldo de la información que alguna vez se cargó en el SAD.

Objeto folderData

Se obtiene los archivos y folders que contiene un folder en especifico a través del acceso a la tabla de la base de datos correspondiente a los folders y su contenido. Se accede a esta clase generalmente desde la página JSP que despliega el contenido de los folders.

Objeto imageFolder

Es la clase que representa lógicamente a la tabla folder de la base de datos. Es accesada por la clase *folderData* para obtener de la capa de datos la información de los folders que se requiera.

Objeto insertaFile

Esta encargada de registrar nuevos archivos y versiones de éstos en la base de datos, así como de realizar el proceso lógico de comprobación de la existencia o ausencia de un archivo en determinado folder para determinar si se debe registrar una versión nueva o bien un nuevo archivo.

Objeto modFile

Realiza modificaciones en el registro de los archivos que se publican en el sistema, como renombrados y eliminación de archivos así como de sus versiones.

Objeto imageFiles

Objeto que representa lógicamente a la tabla “files” de la base de datos.

Objeto modVersion

Modificaciones a las versiones publicadas, principalmente la eliminación de éstas ya sea versión por versión, o cuando se eliminan los registros de archivos con todas las versiones que se contienen.

Objeto obtRuta

Se utiliza principalmente para el despliegue de la ruta de subdirectorios en que se encuentra ubicado un usuario y se basa en hacer una búsqueda recursiva en la tabla que contiene la información de los folders.

Objeto busquedaDB

Realiza búsquedas entre los nombres de los archivos y folders publicados en el sistema de administración de documentos, los cuales están alojados en los registros de la base de datos con el fin de una localización más rápida de los documentos alojados en el sistema.

3.4.5 Diagrama de clases del paquete "utils"

Este paquete esta encargado de realizar tareas extra que apoyan a los demás paquetes, en casos como la comunicación con otras fuentes de datos y de información como la base de datos y servidor STMP por medio del API de Java Mail.

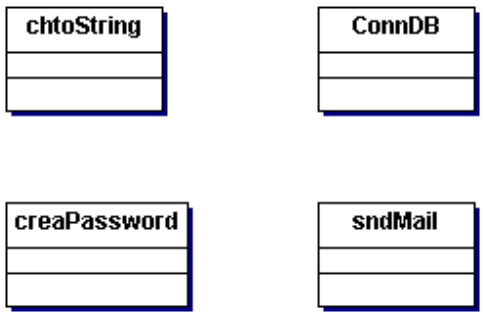

Fig. C3-7 Diagrama de clases del paquete utils

Objeto chtoString

Su funcionamiento es simple, pero realiza lógica utilizada repetidamente en las páginas JSP. Básicamente cambia los datos numéricos que se tienen en la base de datos a las cadenas que están representadas por dichos datos.

Objeto creaPassword

Realiza la lógica de creación de un password o contraseña aleatoria de 8 dígitos que se obtiene al insertar un usuario nuevo, con el fin de evitar colocar una contraseña generalizada a todos los usuarios, aunque estos lo pueden modificar posteriormente.

Objeto sndMail

Basado en el API de Java Mail, esta utilería permite el envío de mensajes de correo electrónico utilizados al actualizar los datos de los usuarios y al crear nuevos usuarios, es utilizada al hacer modificaciones importantes en el sistema de archivos del SAD, como cambio de nombre de carpetas y grupos entre algunas otras.

Objeto ConnDB

Este maneja las conexiones con la base de datos y es la clase mas utilizada por el resto de las clases que componen el sistema de administración de documentos. Tiene métodos para realizar peticiones a la base de datos y actualizaciones a la misma. Es adaptable a la un pool de conexiones, lo cual optimiza el manejo de las conexiones a la capa de datos del sistema.

De esta manera se han descrito los diferentes paquetes y clases que conforman el SAD. Con esto se tienen mas bases para realizar los diagramas de secuencias, los cuales nos muestran las interacciones entre los distintos objetos del sistema.

3.5 Diagramas de Secuencias

El diagrama de secuencias es uno de los diagramas más efectivos para el modelado de las interacciones entre los objetos de los sistemas. Estos diagramas se modelan para cada caso de uso, por lo que los diagramas que se mostraran posteriormente están muy relacionados con los diagramas de casos de uso descritos en el capítulo dos.

Por su parte los diagramas de casos de uso muestran una vista de negocio del escenario donde se desenvuelve el sistema, mientras el diagrama de secuencia contiene detalles de implementación de dicho escenario, incluyendo objetos y clases que se emplean para su implementación, así como los mensajes pasados entre los objetos

Normalmente se examina la descripción de los casos de uso para determinar cuales son los objetos necesarios para la implementación del escenario, si se tiene modelada la descripción de cada caso de uso como una secuencia de varios pasos, resulta sencillo seguir estos pasos para la creación de los objetos que son necesarios en el sistema en cuestión. Dicha descripción se realizo en el capítulo dos, por lo que dichas descripciones sirven de base para la creación de los diagramas de secuencias y en general el descubrimiento de los objetos necesarios, así como el tipo de información y el momento en que son requeridas.

Es importante mencionar que en los diagramas de secuencias se crean nombres de las operaciones de negocio que se tienen que realizar para la comunicación e interacción de los objetos, los cuales sirven de referencia en el momento de la implementación de los objetos, clases y sus operaciones.

Como se ha mencionado, corresponde a cada caso de uso el modelado de un diagrama de secuencias, por lo que a continuación se muestran los diagramas de secuencias correspondientes a los casos de uso obtenidos en el análisis del sistema, así como una breve descripción del proceso que describen.

Para la realización de estos diagramas, fue necesaria la utilización de una herramienta CASE de modelado en UML, dicha herramienta es una plataforma de modelado, construcción y publicación de software llamada Togheter versión 6.1.

3.5.1 Diagrama de secuencias del caso de uso Autentificar

La seguridad es importante dentro del sistema, pues es necesario que se haga uso del sistema de manera controlada y con una identificación completa de los usuarios. Cuando un usuario ingresa al sistema, la primera acción a realizar es la de ingresar su nombre de usuario y contraseña a través del navegador web, estos datos son recibidos por el servidor mediante el Servlet que se encarga de controlar las peticiones referentes a la autentificación de usuarios(*ControlAuth*). Este se encarga de crear un objeto *PassiveCallbackHandler*, cuya clase implementa a la clase *CallbackHandler*, dicho objeto para ser creado recibe como parámetros el nombre de usuario y contraseña. De la misma manera, en el Servlet *ControlAuth* se crea un objeto *LoginContext* que recibe como parámetros el objeto *CallbackHandler* creado anteriormente y el nombre de la propiedad donde obtener al objeto *LoginContext* que invocará al objeto *LoginModule* que realizará la lógica de autentificación de los

datos recibidos por el Servlet, contra los contenidos en la base de datos, y en caso de ser una autentificación exitosa, recoger de la base de datos, la información asociada con los usuarios e insertarla en objetos *Credential* y *Principal*. , En nuestro caso es la clase *RdbmsLoginModule* quien por medio del objeto *ConnDB* realiza el proceso de obtención de datos desde la base de datos.

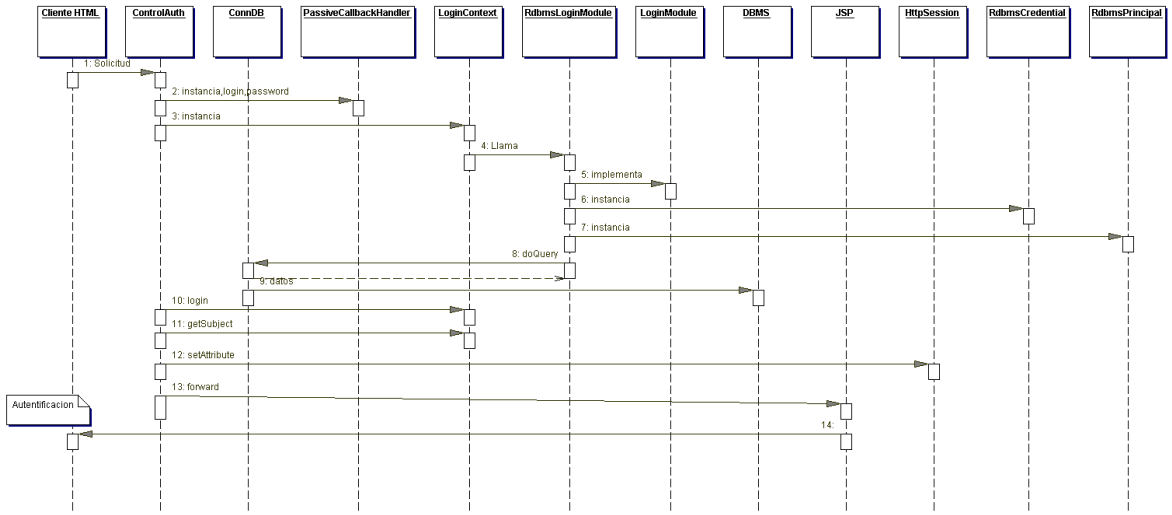

Fig. C3-8 Diagrama de secuencias del caso de uso Autentificar

En caso de lograr una exitosa autentificación con el sistema, el Servlet extrae la información asociada al usuario después de ejecutar el método *login()* de la clase *LoginContext*, datos que se adjuntan a la sesion creada para el usuario autentificado.

Dentro de los datos asociados al usuario se encuentra que tipo o nivel de usuario es, éste dato se utiliza para finalmente direccionar a los usuarios a la interfaz web JSP de inicio que le corresponde.

3.5.2 Diagrama de secuencias del caso de uso “Bloquear-desbloquear Usuario”

El caso de uso referente al bloqueo de usuarios es solo accesible al administrador del sistema y tiene la funcionalidad de permitir y denegar el acceso a usuarios por cualquier motivo que se amerite sin eliminar las referencias de publicaciones de dichos usuarios, ya que al eliminar a algún usuario del sistema, se perderían las referencias de publicación en los archivos, dejando incompleta la información adicional referente a los documentos publicados.

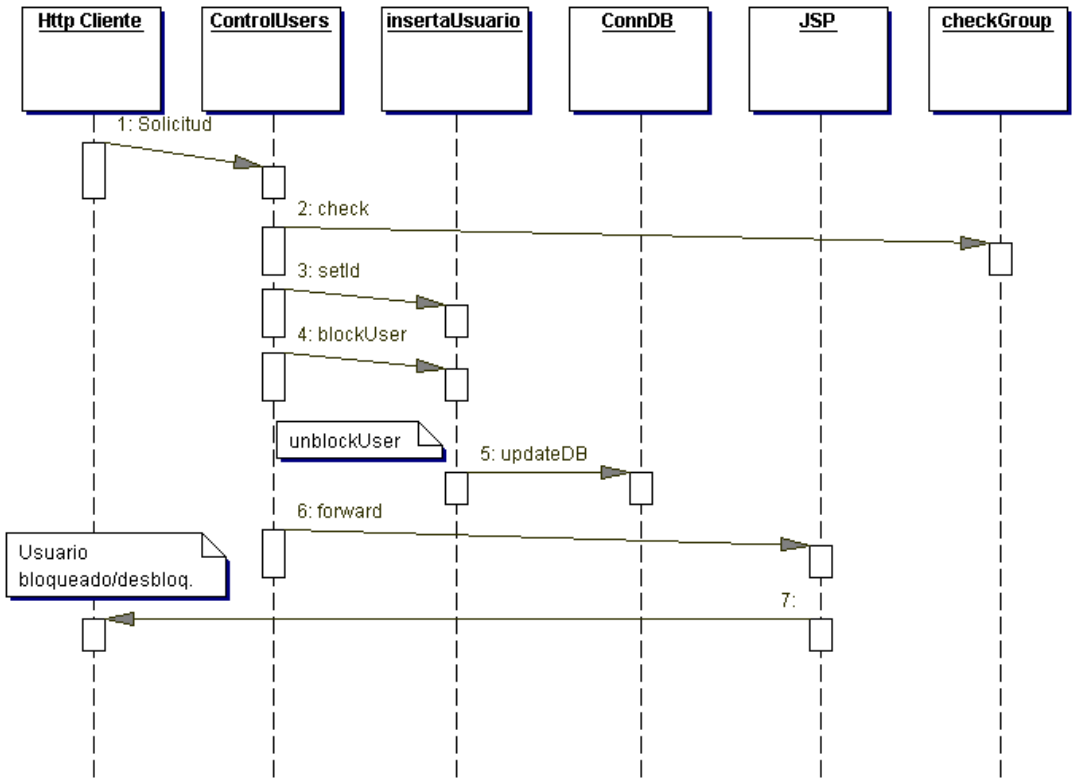

Fig. C3-9 Diagrama de secuencias del caso de los casos de uso Bloquear y desbloquear usuario

En este caso el usuario parte de la página JSP a la cual solo tienen acceso usuarios con privilegios de administración ya que se realiza inicialmente una comprobación del tipo de usuario que se trata según los atributos cargados a la sesión en el momento de la autentificación. Dicho usuario envía una petición de bloqueo o bien de desbloqueo de algún usuario la cual es recibida por el Servlet controlador llamado *ControlUsers* el cual después de comprobar que el rol del usuario es el de un administrador, crea un objeto *insertaUsuario* al cual le envía el identificador único del usuario que se desea bloquear / desbloquear. Posteriormente ejecuta el método de bloqueo(*blockUser*) o bien desbloqueo(*unblockUser*) según sea el caso, el objeto *insertaUsuario* utiliza también un objeto ConnDB con el cual se conecta a la base de datos para realizar las actualizaciones debidas sobre el campo “*block”* de la tabla usuarios de la base de datos del sistema. Una vez realizadas las

actualizaciones, el Servlet controlador envía al usuario nuevamente a la página JSP de administración de usuarios, la cual contiene las actualizaciones realizadas con anterioridad.

3.5.3 Diagrama de secuencias del caso de uso "Búsqueda"

El caso de uso "búsqueda" es accesible para todos los usuarios ya que este es una herramienta para la localización de los archivos contenidos en el sistema, y desde los usuarios administradores hasta los lectores tienen la posibilidad de realizar búsquedas sobre los archivos registrados en la base de datos.

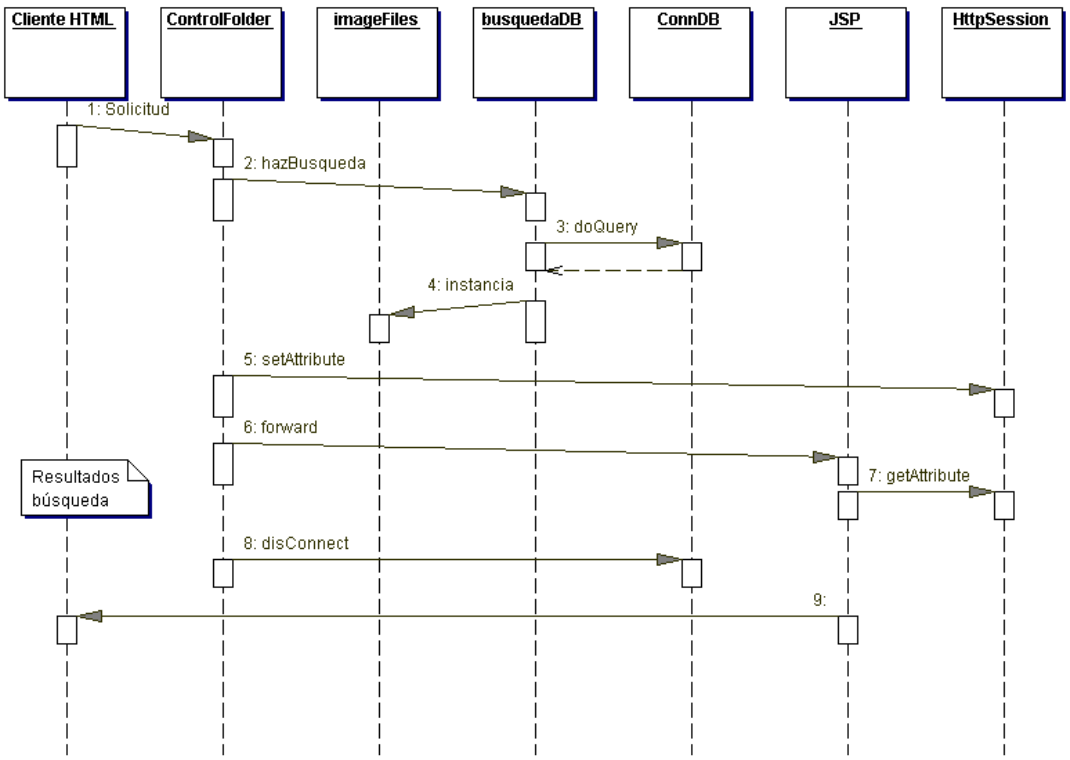

Fig. C3-10 Diagrama de secuencias del caso de uso Búsqueda

La interfaz de búsqueda se encuentra en todos las páginas que presentan contenido de archivos y folders, y desde esta interfaz se hace la petición especificando el tipo de búsqueda y la cadena a buscar, por lo que esta solicitud la recibe el Servlet controlador(*ControlFolder*) la cual crea un objeto *busquedaDB,* llamando al método *hazBusqueda.* Este método realiza una consulta mediante el método *doQuery* de la clase *ConnDB*, hacia la tabla files de la base de datos, el cual contiene la información referente a todos los archivos cargados en el sistema.

El método *hazBusqueda* regresa un objeto de tipo *Vector* el cual crea insertando objetos *imageFiles* sobre los cuales se coloca la información obtenida en la base de datos a través de sus métodos getXXX y setXXX. Dicho objeto *Vector* es colocado en la sesión a través del método *setAttribute* de

HttpSession para que posteriormente el Servlet controlador redireccione por medio del método *forward* a otra página JSP que extraerá los resultados del objeto Vector recibido a través de la sesión creada con *HttpSession* mediante el método *getAttribute*.

Finalmente la página JSP que recibe el objeto Vector con los resultados de la búsqueda, los despliega con ligas hacia su ubicación dentro del sistema de archivos del SAD en caso de que dichos archivos localizados correspondan al grupo al que pertenece el usuario que realiza esta búsqueda.

3.5.4 Diagrama de secuencias del caso de uso "Cambio nombre folder"

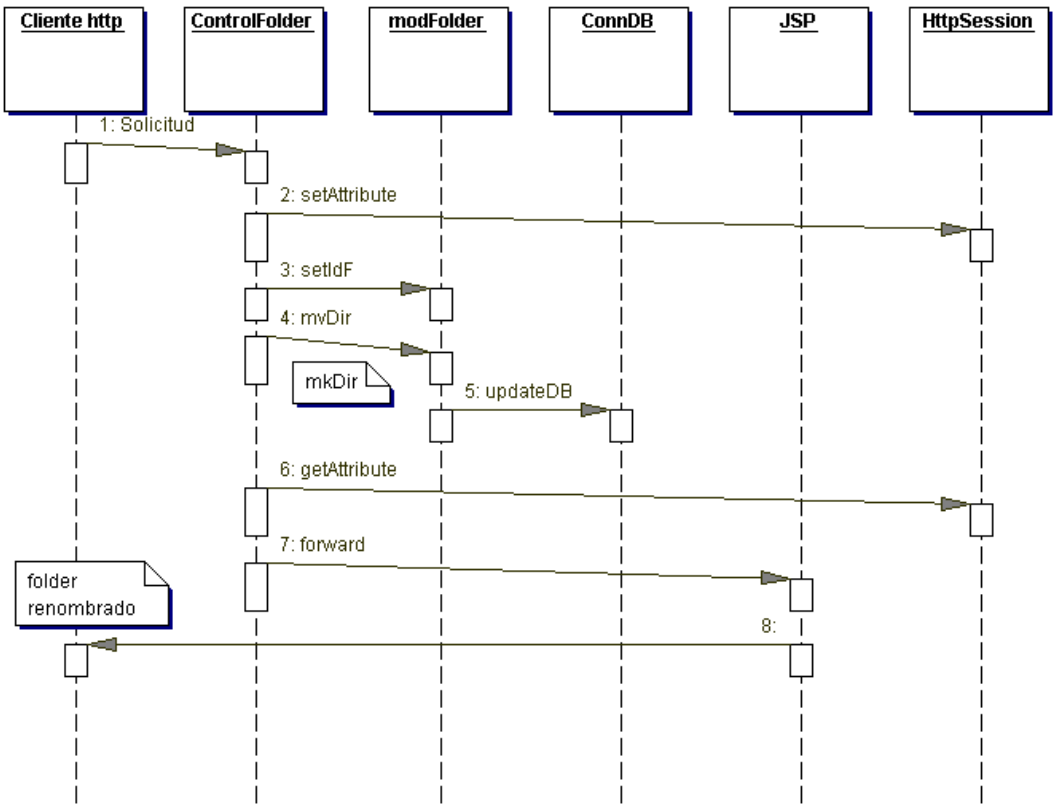

Fig. C3-11 Diagrama de secuencias del caso de uso Cambio nombre folder y crear folder

Es necesario que en ocasiones que se realicen modificaciones sobre el contenido del sistema, por lo que el cambio de nombres y descripciones de las carpetas es indispensable, en este caso de uso se contempla dicha acción.

Para realizar esta acción, el Servlet controlador(*ControlFolder*) recibe una solicitud de cambio de nombre de algún folder, posteriormente el controlador se encarga de colocar en la sesión los datos correspondientes al folder que se desea renombrar, de la misma manera crea un objeto *modFolder*, al que le coloca el identificador del folder a modificar, así como el nuevo nombre por medio del método *setIdF()* y *setNewName()* respectivamente, seguido del método mvDir, el cual modifica el

nombre del folder en la base de datos. Finalmente el Servlet controlador con el apoyo de la recuperación del identificador del folder, redireccionan al usuario a la página JSP donde hizo la petición, pero con las actualizaciones de nombre y comentarios de archivo correspondientes.

La figura C3-11 muestra el proceso de modificación de nombre y creación de folders, ya que es un proceso en el que están relacionadas las mismas clases y en general es un proceso muy parecido.

3.5.5 Diagrama de secuencias del caso de uso "Crea nuevo folder"

Uno de los objetivos principales del sistema de Administración de documentos es la clasificación intuitiva de los documentos contenidos, por esto se hace necesaria la creación de nuevos folders o carpetas con el fin de tener una buena distribución de los documentos.

La realización de este proceso es muy similar al del cambio de nombre de folders, con la diferencia de que el Servlet controlador coloca en el objeto *modFolder* el identificador del folder padre y no el folder que se desea cambiar, de igual manera se coloca el nuevo nombre mediante el método *setNewName()* para indicar el nombre del nuevo folder. Como última diferencia, se ejecuta el método *mkDir()* el cual realiza un proceso de actualización a la base de datos a través de la creación de un objeto *ConnDB*.

3.5.6 Diagrama de secuencias del caso de uso "Crea grupo"

La creación de nuevos grupos es necesaria para la clasificación de los nuevos usuarios, o bien de los usuarios que ya pertenecen a un grupo y se hace necesaria su pertenencia a algún otro. Esta actividad la pueden realizar únicamente usuarios con privilegios de administrador como ya se mostró en el diagrama de casos de uso correspondiente al tipo de usuario administrador.

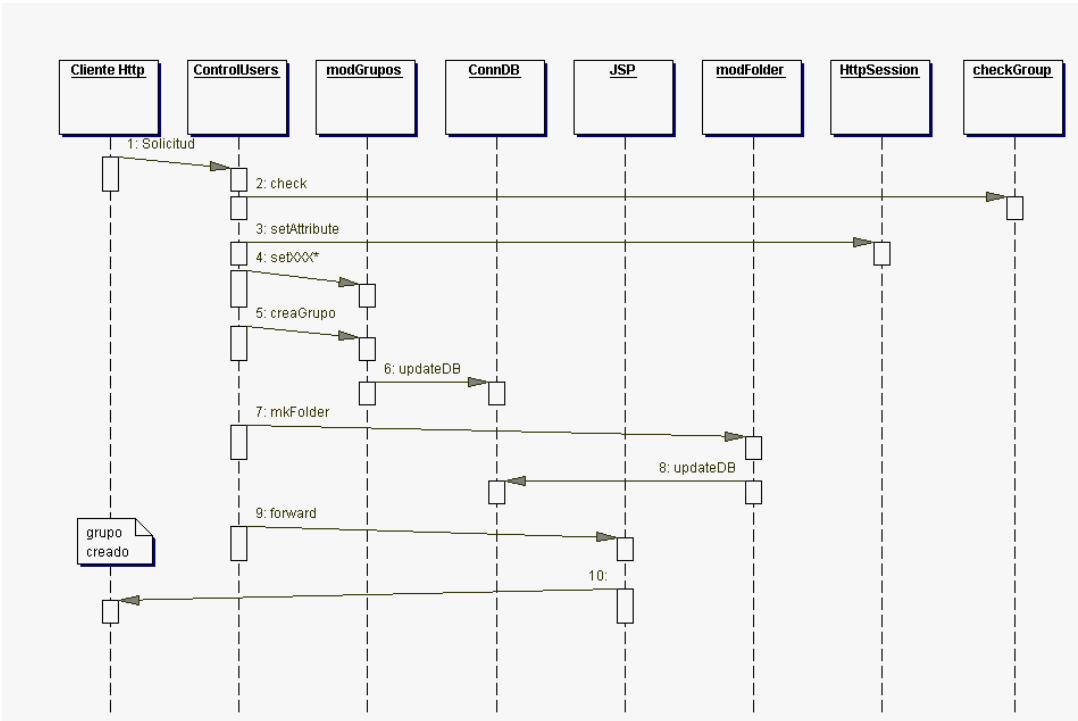

Fig. C3-12 Diagrama de secuencias del caso de uso Crea Grupo

El usuario administrador hace la solicitud de creación de un nuevo grupo, la cual es recibida por el Servlet controlador de usuarios (*ControlUsers*), en primera instancia se comprueba el nivel del usuario mediante el uso del método *check* de la clase *checkGroup,* posteriormente el Servlet crea un objeto *modGrupos* al cual le adjunta mediante métodos de tipo setXXX los valores correspondientes al la creación del nuevo grupo, como el identificador del usuario creador y el nombre del grupo para que posteriormente ejecute el método *creaGrupo* el cual crea un objeto ConnDB que es quien realiza las inserciones sobre la tabla “grupos” de la base de datos.

Finalmente, *ControlUsers* redirecciona al usuario a la página de administración de usuarios para poder continuar con tareas relacionadas con la gestión de los usuarios que pueden ingresar al sistema. Estas acciones se muestran gráficamente en la figura C3-12.

3.5.7 Diagrama de secuencias del caso de uso “Despliega Árbol ”

La realización de este caso de uso es una herramienta a la que tienen acceso los cuatro diferentes tipos de usuarios, ya que tiene la funcionalidad de hacer más fácil la navegación entre los folders y subfolders que componen el sistema de archivos que se maneja en todo momento. Cabe señalar que este árbol de navegación se muestra en la mayoría de las interfaces que están relacionadas con el manejo de archivos y solo en las referentes a la administración de usuarios no se despliega pues no tendría una funcionalidad necesaria, consumiendo recursos en tiempo de ejecución y memoria.

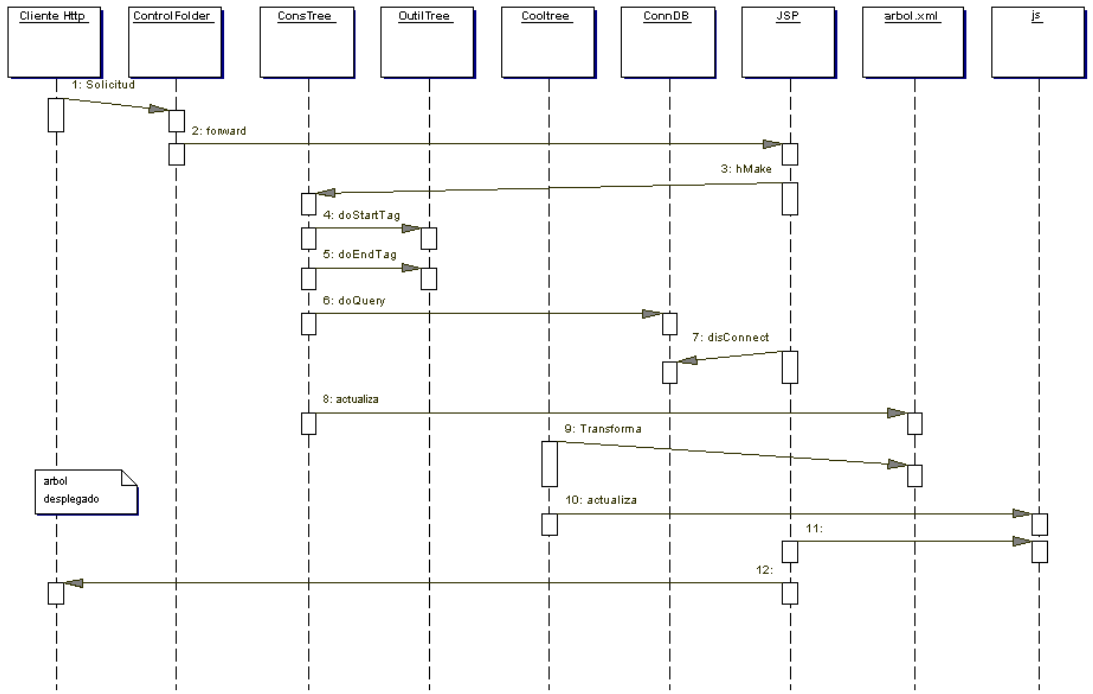

Fig. C3-13 Diagrama de secuencias del caso de uso Despliega Árbol

Al inicio el cliente http hace la petición al servidor web para desplegar alguna página JSP que contenga elementos relacionados con el manejo de archivos, y entre las otras tareas de despliegue, esta reenvía a una página JSP que es la encargada de ejecutar los métodos correspondientes a la

creación de los archivos js que contienen el script para la realización gráfica del arbol de navegación.

Como se ha dicho, desde la página JSP se ejecuta el método *hMake()* de la clase *ConsTree*, este método con la colaboración de la clase *OutilTree* a través de sus métodos *doStartTag()* y *doEndTag()* y de la clase *ConnDB* con el método *doQuery()* forman un archivo XML que contiene la estructura de directorios contenida en la base de datos, pero que por la clase de estructuración de la información en las bases de datos relacionales, no podría formarse dicha estructura de "arbol" que se necesita para tener una concepción mas natural del sistema de archivos.

La información que se extrae proviene de las tablas "files" y "folders" de la base de datos y se genera dicho archivo XML mediante un algoritmo que contiene elementos recursivos para formar de manera estructurada y anidada los elementos que componen el directorio.

Se eligió la estructuración de un archivo XML ya que entre sus propiedades se encuentra la formación correcta de sus elementos, es decir no están permitidos errores de estructuración y cada etiqueta que se abre tiene que ser cerrada en algún momento de manera correcta. Además de que esta formado de una manera muy similar al árbol de navegación que se desea obtener, un nodo puede contener subnodos y estos a su vez a otros de su misma clase.

Una vez teniendo el archivo XML en memoria, se actualiza en el servidor sobrescribiéndolo cada que ocurre un cambio en la estructura del sistema de archivos. Posteriormente, en la segunda fase de este proceso, se tiene que realizar la transformación del formato XML a un medio con el cual se puede desplegar el árbol de manera amigable en el navegador web.

Dicho proceso se realiza a través de una transformación XSLT, por medio de la clase *Cooltree* la cual tiene un método llamado *hazParse()* que analiza el documento fuente, en este caso es el archivo llamado arbol.xml y mediante la base de una hola de estilo xsl realiza el cambio de formato a un archivo js(java script) el cual se actualiza al realizar este proceso. Finalmente, éste se envía a través del servidor web junto con otros archivos que tiene la función de dar la configuración y la forma en que el cliente web o navegador los debe de procesar mediante javaScript para obtener el resultado esperado. Esta configuración trata de ser funcional en todos los navegadores web que existen en el mercado o al menos con los dos mas comerciales(MS I Explorer y Netscape).

Este proceso de transformación desde la base de datos, hasta un formato capaz de ser leído en cualquier navegador web se muestra gráficamente en la figura C3-13 de manera sencilla a través de un diagrama de secuencias.

3.5.8 Diagrama de secuencias del caso de uso "Despliega archivo "

Este caso de uso es generalizado para todos los usuarios ya que es una de las funciones primordiales del sistema de administración de documentos, la publicación de los documentos, así como sus propiedades como el usuario publicador, su tamaño, tipo y fecha en que fue colocado en el servidor. Es necesario obtener variada información de mas de una tabla de la base de datos, debido a que se verifica el grupo del usuario, el tipo de usuario, los datos propios del archivo, las versiones del archivo que se esta observando que se tienen guardadas en el servidor, así como sus propiedades independientes.

Cuando el cliente http o navegador web hace una petición al servidor web solicitando el despliegue de este caso de uso, el Servlet controlador(*ControlFolder*) recibe la petición y coloca en la sesión por medio del método *setAttribute* de la clase *HttpSession*, el identificador del archivo que se desea desplegar, seguido de una redirección a través del método *forward* del Servlet controlador.

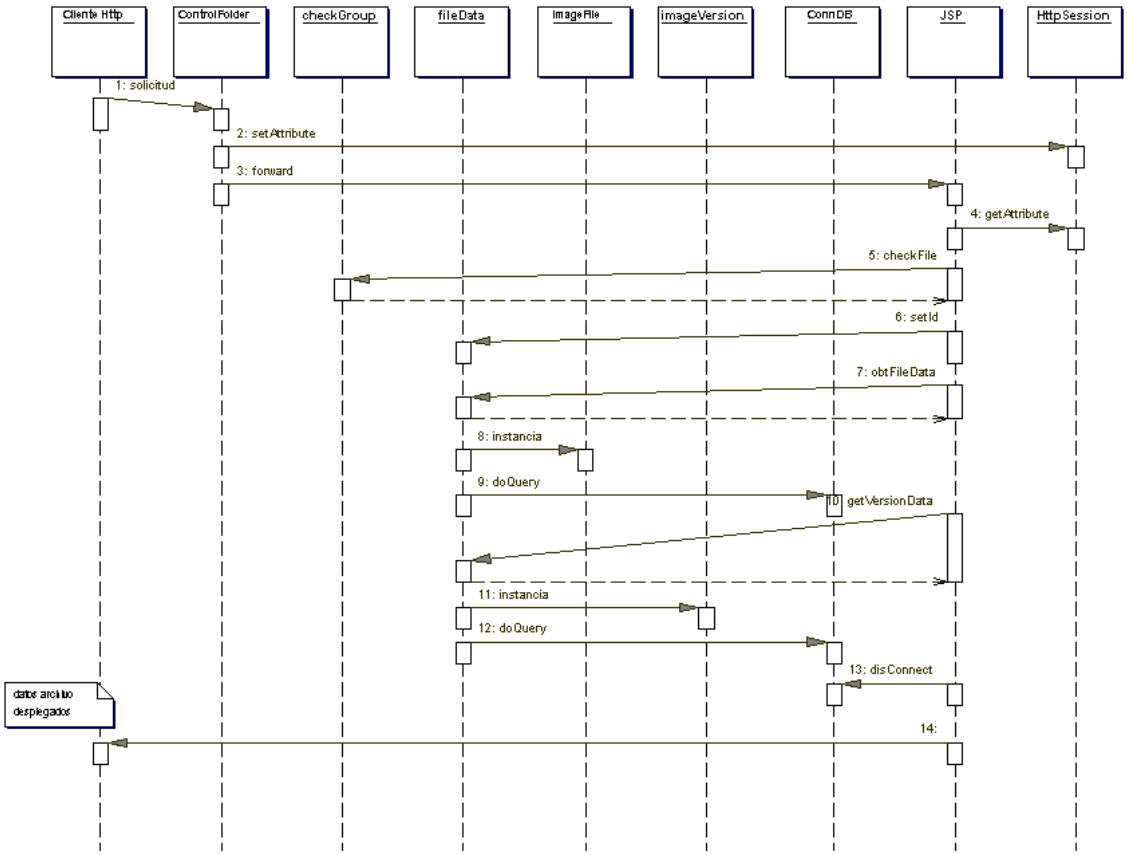

Fig. C3-14 Diagrama de secuencias del caso de uso Despliega archivo

La petición es recibida por una página JSP llamada *fileD.jsp* la cual verifica por el método *checkFile* de la clase *checkGroup* si el usuario corresponde al grupo en el que fue publicado el archivo que se desea desplegar. En caso de que se le niegue esta información al usuario, éste es enviado a una página de error, pero en caso contrario, la página despliega la información del archivo requerida creando un objeto *fileData* y colocando por medio del método *setId* el identificador del archivo que fue obtenido previamente de la sesión. Posteriormente se ejecuta el método *obtFileData()* el cual regresa un objeto *Vector* el cual contiene objetos del tipo *imageFiles* de los cuales se obtienen sus propiedades por medio de la realización de querys a la base de datos a través de la clase *ConnDB.* De una manera similar se obtienen las versiones que se tienen registradas de dicho archivo, pero a través del método *getVersionData* de la clase *fileData* pero a través de la creación y colocación de

objetos *imageVersion*. Finalmente se ejecuta desde la página JSP un método *disConnect()* de la clase *ConnDB* para liberar la conexión, *Statement* y *ResultSet* creados para realizar los querys.

De esta manera se presentan la información solicitada, además de interfaces para realizar peticiones de modificaciones sobre los archivos y descripciones detalladas de las versiones cargadas en el servidor.

3.5.9 Diagrama de secuencias del caso de uso "Despliega folder"

Para realizar el despliegue del contenido de los diferentes folders, el Servlet controlador redirecciona la petición reenviada hacia una página JSP llamada folderD.jsp, encargada de publicar la información del folder, los subfolders contenidos, los archivos contenidos y las interfaces para las modificaciones a los folders(cambio de nombre, cambio de grupo y eliminación de folder), según el tipo de usuario, por ejemplo a un usuario lector únicamente se le presentan los datos del folder y los archivos contenidos, mientras que a un usuario moderador se le presentan todas las posibilidades que se mencionaron anteriormente.

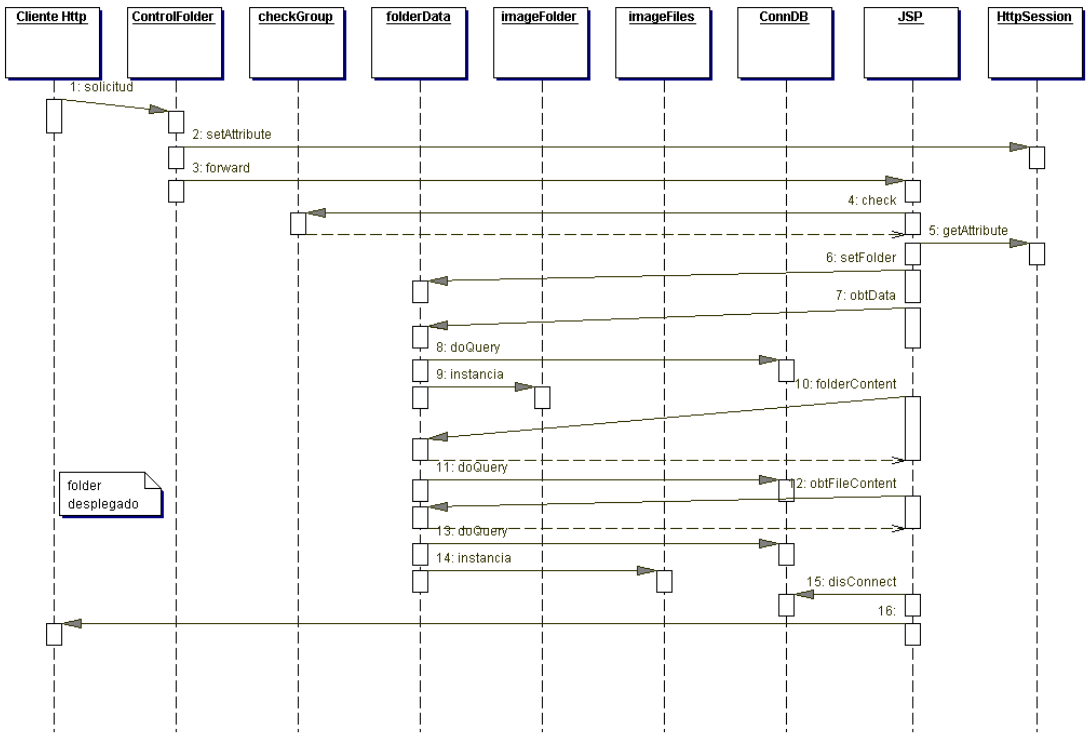

Fig. C3-15 Diagrama de secuencias del caso de uso Despliega folder

El proceso de adquisición de los datos es muy similar al realizado en la obtención de los datos de los archivos, con la diferencia de que las clases utilizadas, así como los métodos son los propios de los folders. Este proceso, así como las clases y métodos utilizados se muestran en el diagrama de secuencias de la figura C3-15.

3.5.10 Diagrama de secuencias del caso de uso “Elimina archivo”

Al utilizar este caso de uso se tiene que eliminar la información propia del archivo, asi como las versiones asociadas a éste. Por lo que están implicadas las tablas “archivos” y “versiones” de la base de datos.

Para eliminar el archivo, se realiza esta petición desde el cliente web, por lo que el Servlet controlador recibe la petición y coloca en la sesión el identificador del archivo que se desea eliminar para redireccionar al usuario a una página JSP que contiene la información de dicho archivo, así como las versiones que se eliminaran junto con él, y la confirmación de la eliminación de la base de datos de los documentos que puedan estar contenidos en el servidor y que tengan relación con el documento principal o inicial.

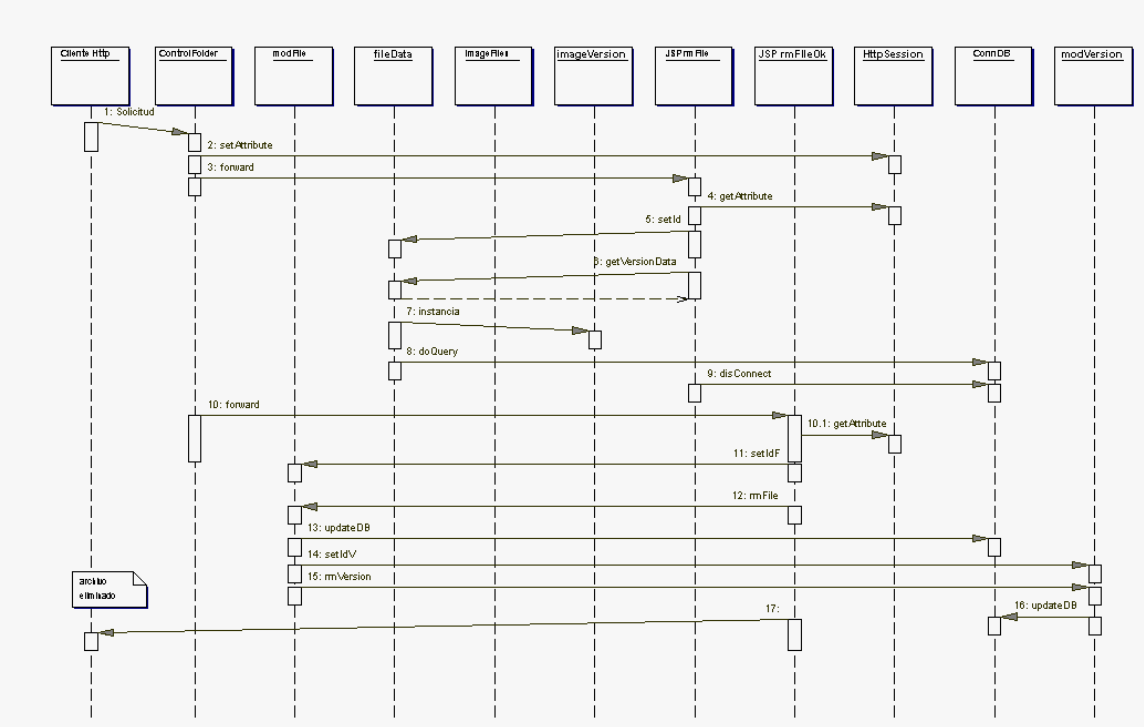

Fig. C3-16 Diagrama de secuencias del caso de uso Elimina Archivo

Como se ha visto los procesos de actualización y modificación de los diferentes procesos relacionados con los archivos, versiones y folders son muy similares, por lo que se puede tomar como referencia el diagrama de secuencias de la figura C3-16 para observar el proceso de la eliminación del archivo, el cual termina con el redireccionamiento del usuario a la página JSP que despliega el contenido del folder al que pertenecía, o bien si por alguna razón accidental o intencional algún usuario no autentificado o con un nivel inferior al de moderador intentara eliminar cierto archivo, se le redireccionaría a una página de error que le indica la falla que esta cometiendo.

3.5.11 Diagrama de secuencias del caso de uso “Elimina folder”

La diferencia con respecto a las demás modificaciones de este proceso, es la eliminación de los subfolders y archivos que pueda contener el folder que se desea eliminar, así como los archivos que puedan contener sus hijos. Por lo que se realiza un proceso iterativo y recursivo dentro de la clase *modFolder*, dentro del método *remFolderList*.

En general el proceso se lleva a cabo obteniendo de la petición el identificador del folder que se desea eliminar, mostrando posteriormente una página de confirmación de la eliminación donde se presentan los folders, subfolders y archivos que se eliminaran si se procede con la operación. Una vez confirmada, se ejecuta el método *remFolderList* que recibe como parámetro el identificador del folder, eliminando todos los elementos que se tengan a partir de dicho folder.

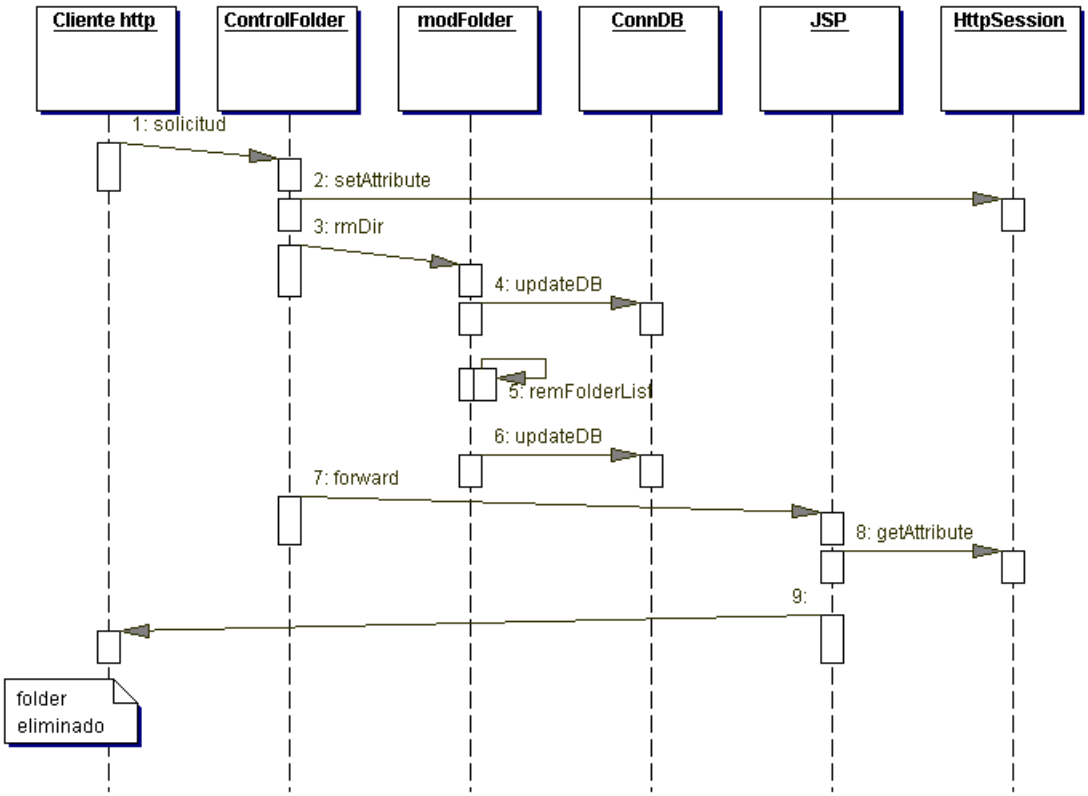

Fig. C3-17 Diagrama de secuencias del caso de uso Elimina Folder

Para finalizar, el usuario es enviado a la página de despliegue del folder padre del eliminado, cabe señalar que el único folder no eliminable por el usuario es el folder raíz ya que si se permitiera esto, se perdería toda la información contenida en el sistema por algún error de los usuarios finales del sistema.

Esta herramienta es solo accesible a los usuarios con rol de administradores, quienes pueden eliminar folders en el lugar que deseen, mientras que los usuarios moderadores solo pueden realizar esta operación dentro de las carpetas que pertenecen a su grupo, aunque de hecho no tiene acceso a la información de otros grupos, puede darse el caso de un acceso no permitido o accidental.

3.5.12 Diagrama de secuencias del caso de uso "Elimina versión"

Es uno de los procesos más sencillos pues solo se tiene que modificar la tabla de "versión" para eliminar los registros de la versión que se desea eliminar. Como lo indica el diagrama de secuencias de la figura C3-18, se utilizan objetos *modFolder* que realizan dicha funcionalidad y un objeto *ConnDB* quien realiza la conexión con la base de datos, así como la actualización de los registros contenidos. Este proceso esta implícito en la eliminación de archivos, como se puede ver en el diagrama C3-16 correspondiente al caso de uso "eliminar archivo".

Al final el usuario es enviado a la página del archivo original, donde tiene la oportunidad de continuar consultando archivos, o bien realizar otra modificación.

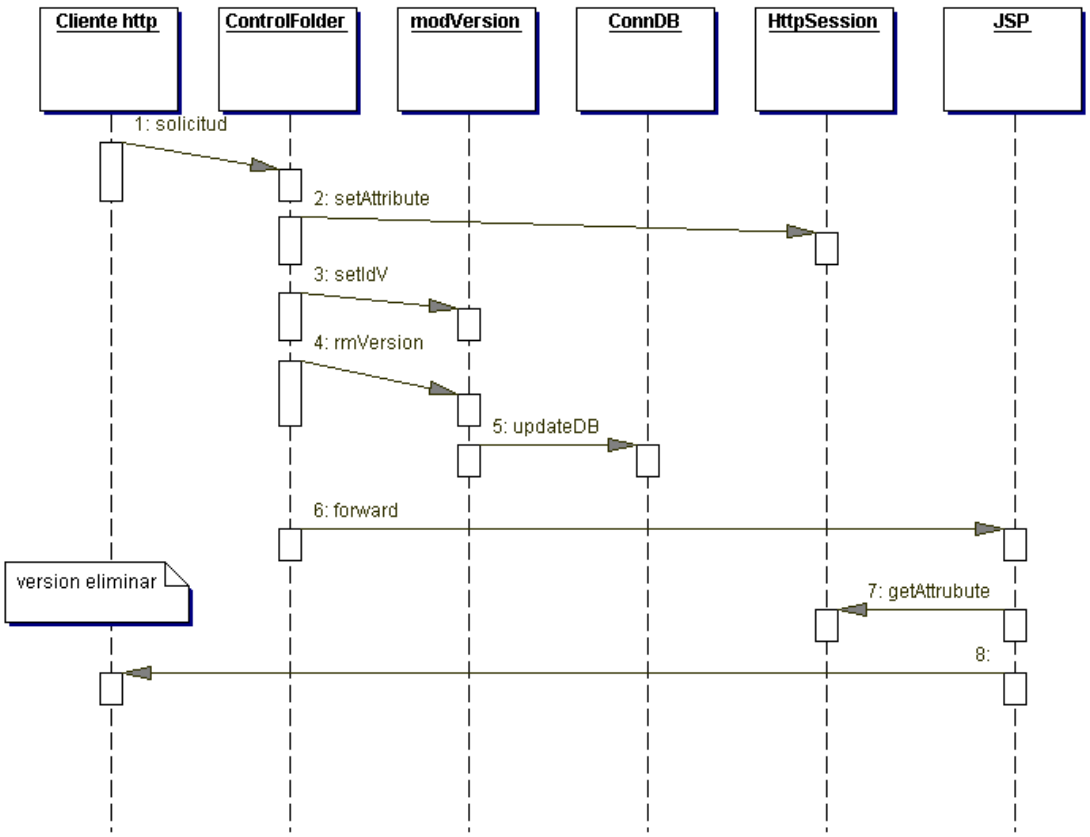

Fig. C3-18 Diagrama de secuencias del caso de uso Elimina Versión

3.5.13 Diagrama de secuencias del caso de uso “envía Mail”

Dentro de los casos de uso que realizan actividades fuera de la comunicación con la base de datos se encuentra el caso de uso de envía Mail, o envía correo electrónico. Este caso de uso utiliza el API de Java Mail y el activation framework de J2EE para realizar las conexiones con el servidor STMP del Instituto Mexicano del Petróleo(imp.mx), con el fin de enviar los mensajes que se requieran dentro del sistema, como es la actualización del perfil de algún usuario, la actualización de sus datos de acceso(contraseña), alta de usuario, publicación de archivos, etc.

Para realizar esta acción, algún Servlet o página JSP que requiera de enviar un mensaje envía la petición al Servlet controlador el cuál crea un constructor de la clase *sndMail*(en caso de que el Servlet lo necesite, el mismo creará el constructor). Sobre dicho constructor se colocan las propiedades del mensaje a enviar, como el destinatario, titulo y cuerpo del mensaje, para posteriormente éste sea enviado por medio de la ejecución del método *enviaMail()*.

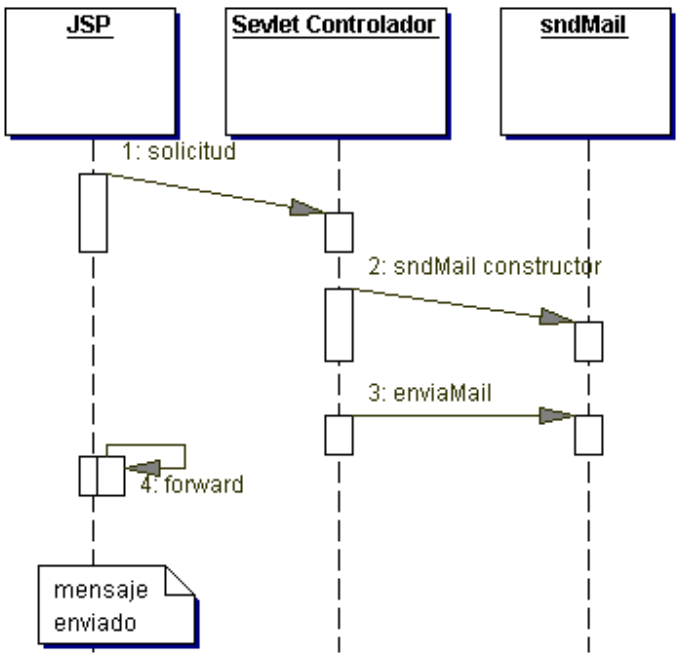

Fig. C3-19 Diagrama de secuencias del caso de uso envía Mail

Dicho método es quien utiliza en sí las utilidades brindadas por el API Java Mail, el cual se apoya sobre el JAF(Java activation framework), java mail es una extensión de los paquetes de java que tiene todas las herramientas para enviar, recibir y gestionar correo electrónico.

3.5.14 Diagrama de secuencias del caso de uso “inserta usuario”

Para realizar la inserción de nuevos usuarios, es necesario actualizar la tabla de usuarios de la base de datos a partir de los datos obtenidos en la forma incluida en la interfaz gráfica desplegada en el cliente http o navegador web, el cual deberá contener el nombre de usuario, contraseña generada automáticamente por el servidor, nombre del usuario, apellidos, correo electrónico, numero de gafete, así como el rol de usuario que se le asignará. Estos datos pasan a una validación que realiza la clase *pRegUsuarios* donde se comprueba que no falte ningún dato de los pedidos, así como la duplicidad de nombres de usuario para el acceso al sistema, esto mediante la ejecución del método *checkUser* de la clase *insertaUsuario* quien tiene que realizar un query a la base de datos mediante la creación de un objeto ConnDB, al cual se le deberán cerrar las conexiones y *statements* creados mediante el método *disConnect*.

En caso de que la validación resultara negativa, el usuario es enviado a la misma interfaz gráfica pero con el despliegue de los errores que cometió en cada campo erróneo.

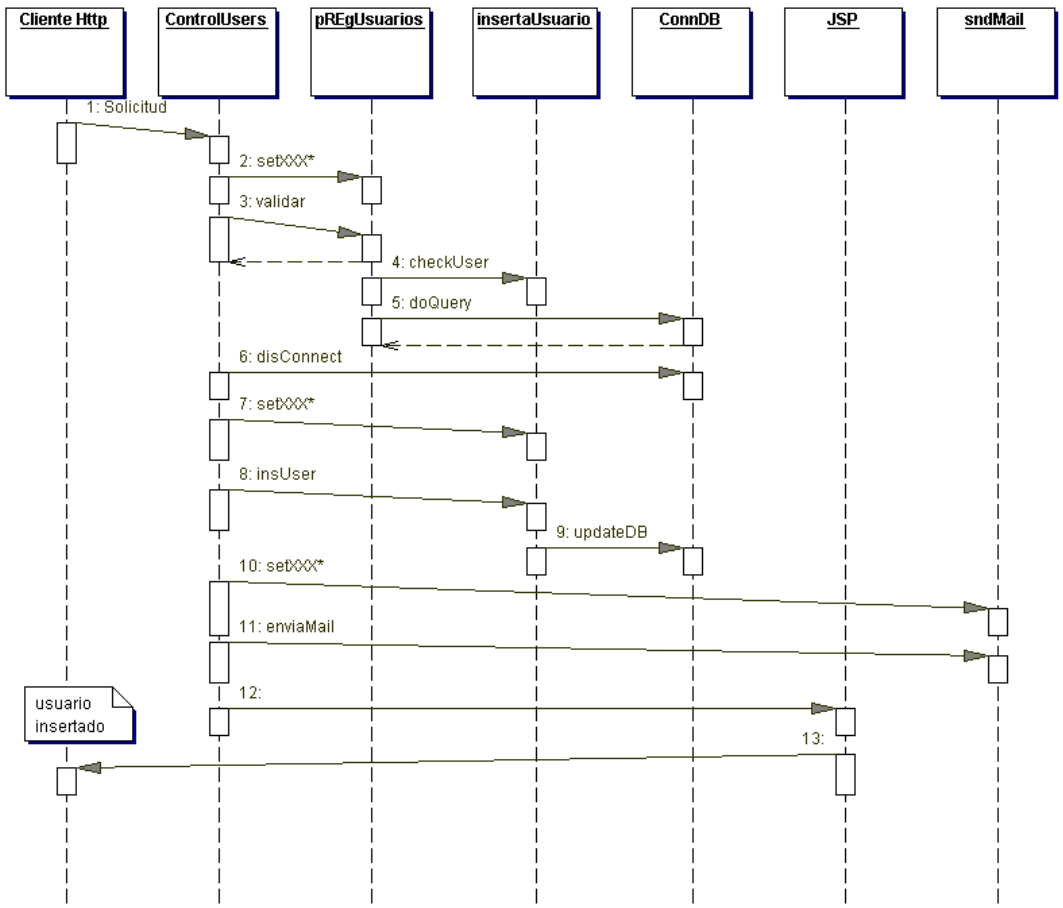

Fig. C3-20 Diagrama de secuencias del caso de uso inserta Usuario

Una vez validados los datos, el Servlet controlador obtiene el grupo al que pertenecerá el usuario mediante una página JSP que contiene la interfaz de inserción de dicha propiedad así como la confirmación de las anteriores, y al recibirlas de nuevo a través del método *getParameter* del *request*, crea un constructor de la clase *insertaUsuario*, el cual contiene los datos a insertar correspondientes al nuevo usuario, para posteriormente ejecutar el método *insUser()*, de la misma manera, éste Servlet crea un constructor de la clase *sndMail* con el mail del usuario creado y un cuerpo de mensaje que contiene el nombre de usuario y su contraseña para tener acceso al SAD.

Finalmente el usuario administrador es redireccionado hacia la página principal de administración, para poder realizar mas tareas administrativas.

3.5.15 Diagrama de secuencias del caso de uso "modificar datos de acceso"

El diagrama C3-21 muestra la secuencia de operación del caso de uso modificar datos de acceso, el cual es accesible a todo usuario que este dado de alta en el sistema y se utiliza para hacer cambios en los datos de los usuarios por los mismos usuarios, a diferencia del alta de los mismos, que la realizan únicamente usuarios con privilegios de administrador.

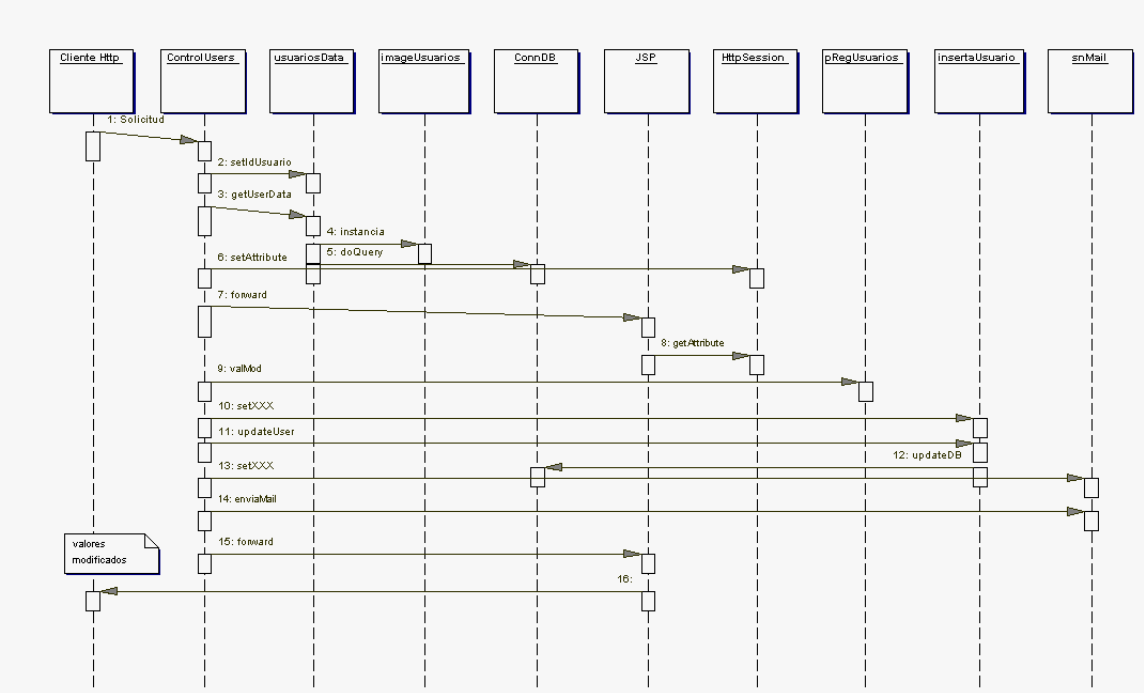

Fig. C3-21 Diagrama de secuencias del caso de uso modificar datos de acceso

Dentro de la interfaz de cambio de datos se puede modificar el nombre, apellidos, correo electrónico, número de gafete, y cambio de contraseña. Estos datos se muestran de forma editable en los campos de texto, con excepción de la contraseña.

Al enviar la solicitud de cambio de datos, éstos datos son procesados para realizar una validación de su contenido por medio del método *valMod* de la clase *pRegUsuarios* y en caso de que ocurra algún error con los datos insertados, el usuario es devuelto a la misma página pero con la diferencia de que se muestran los errores cometidos antes de enviar la solicitud.

De la misma manera que en la inserción de nuevos usuarios, la información actualizada se envía a la dirección de correo electrónico proporcionada por el usuario, o bien la dirección que contenía por default. La figura C3-21 muestra de forma mas detallada y precisa el proceso antes descrito.

3.5.16 Diagrama de secuencias del caso de uso "modificar valores de usuario"

Dentro de las tareas de administración de usuarios, se encuentra el cambio de valores de los usuarios, es decir, el cambio de datos referentes al grupo y tipo de usuario debido a que es necesario en ocasiones el traslado de usuarios entre grupos o bien la elevación o disminución de privilegios a los usuarios, ya sea por errores en la captura inicial o alguna otra situación.

La figura C3-22 muestra a detalle dicho proceso que comienza con la solicitud de dicho caso de uso por parte del cliente http manipulado por algún usuario administrador desde la pantalla de administración de usuarios, de esta manera el Servlet controlador adjunta a la sesión el identificador del usuario que se desea modificar, el cual posteriormente se coloca dentro de un objeto *usuariosData* por medio de un método del tipo *setXXX()*, ejecutando posteriormente el método *getUserData*, el cual instancia a un objeto *imageUsuarios*, en el cual inserta las propiedades obtenidas a partir de la petición a la base de datos e inserta a un objeto del tipo *Vector.*

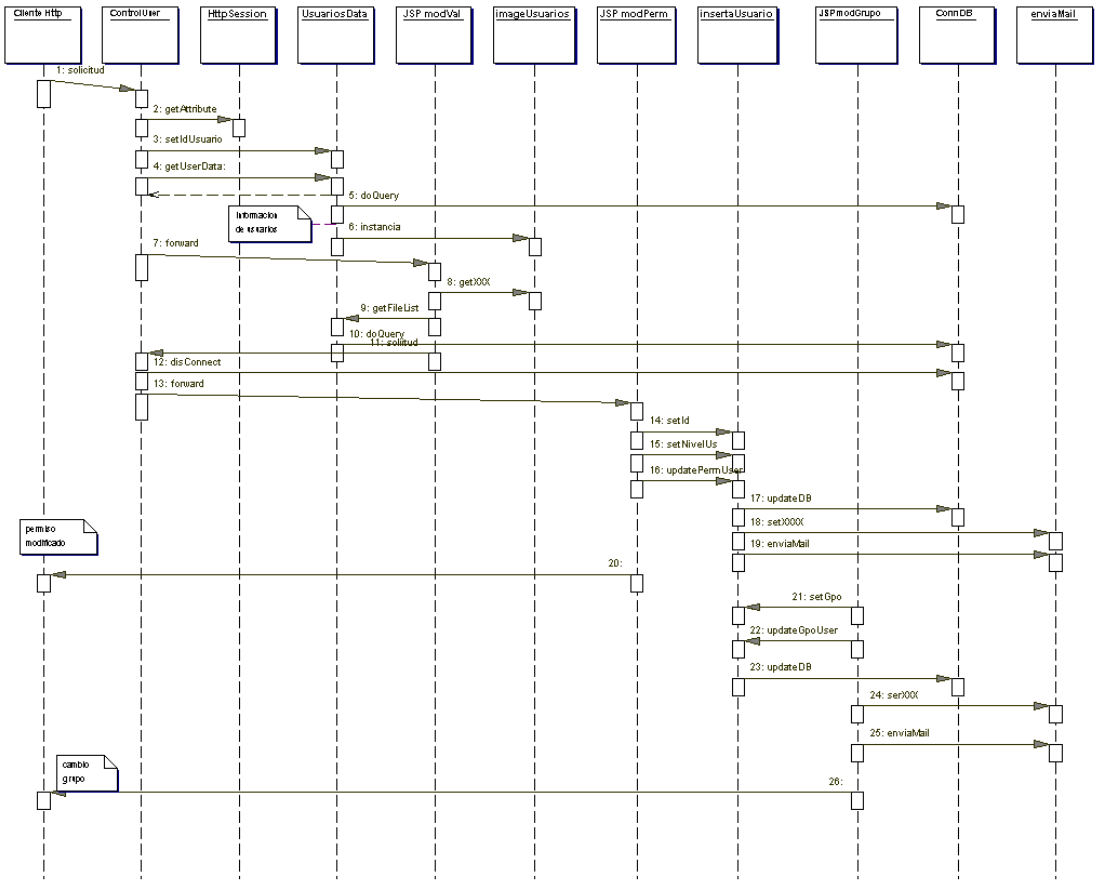

Fig. C3-22 Diagrama de secuencias del caso de uso modificar valores de usuario

Este último objeto es recuperado a través de la sesión y desplegado en una página JSP que incluye los datos de dicho usuario, su rol de usuario actual, grupo actual y un listado de los archivos que ha publicado en el SAD que obtiene mediante el método *getFileList* de la clase *usuariosData*, que también realiza consultas a la base de datos.

La segunda fase del proceso es la actualización del rol de usuario y/o grupo al que pertenece, en el caso del cambio de rol de usuario, se crea un objeto de tipo *insertaUsuario* al cual se le asignan las propiedades de identificador de usuario(*setId*) y (*setNivelUs*) para posteriormente ejecutar las actualizaciones con el método *updatePermUser* el cual realiza la actualización de la base de datos.

Si se elige la opción de cambio de grupo se crea un objeto *insertaUsuario* y se le adjunta el identificador del usuario y el identificador del grupo al que pertenecerá el usuario, cambios que se realizan con el método *updateGpoUser*, el cual realiza las actualizaciones propias de la modificación al sistema.

En ambos casos se realiza el envío de correo electrónico por medio del caso de uso envía mail, que se ha explicado con anterioridad. Finalmente el usuario administrador es direccionado hacia la interfaz que contiene la administración de documentos.

3.5.17 Diagramas de secuencias de los casos de uso “upload archivos” y “upload versiones”

Los casos de uso ilustrados de manera secuencial en el diagrama de secuencias de la figura C3-23 es en realidad el proceso de publicación del SAD, éste proceso es el que se encarga de la publicación y verificación de los archivos existentes dentro de un folder, para llevar un control de versiones automático.

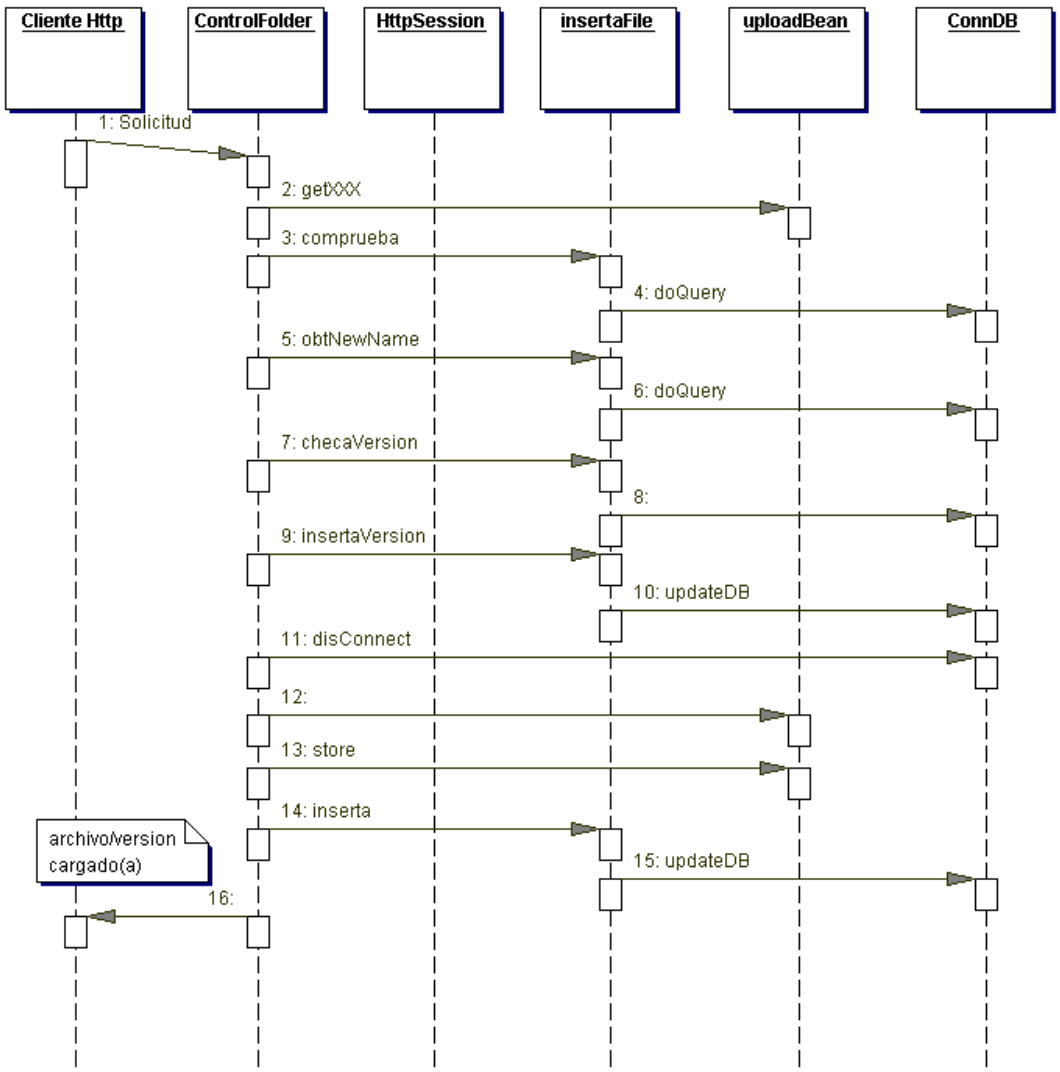

Fig. C3-23 Diagrama de secuencias del caso de uso del upload de archivos y folders

El cliente web envía la solicitud de carga de archivo, el cual contiene al archivo, un comentario breve y el identificador del folder que lo contiene, este es recibido por el Servlet controlador mediante un *MultipartFormDataRequest,* el cual asemeja a un request normal pero tiene algunas otras propiedades como la obtención de archivos cargados y su información relacionada.

Posteriormente el Servlet crea un objeto *uploadBean* con el cual se obtienen a través de métodos del tipo *getXXX* las propiedades del archivo cargado, y por medio del *MultipartFormDataRequest* los datos ya mencionados. Una vez obtenido el nombre del archivo, se verifica a través de la creación de un objeto *insertaFile* con método *comprueba*, si hay algún documento con el mismo nombre dentro de ese mismo folder, según los registros de la base de datos. En caso de que no haya ningún archivo con un nombre similar únicamente lo escribe dentro del servidor y lo registra en la base de datos. En el caso contrario, mediante la ejecución del método *checaVersion* el Servlet obtiene el numero secuencial de versión en que se encuentra dicho archivo y obtiene un nuevo nombre de la versión. Finalmente lo inserta en la base de datos con su nuevo nombre por medio del método *inserta* de la clase *insertaFile* y guarda al archivo en el servidor con dicho nombre alterado, identificándolo por su número de versión con el método *store* del *uploadBean*.

Finalmente el usuario es direccionado nuevamente a la página JSP que da una confirmación de la carga del archivo mostrando los datos esenciales del archivo, el comentario que colocó y si se trata de la versión de un archivo existente o si se trata de un archivo nuevo.

Los diagramas de secuencia que se han mostrado, junto con los diagramas de clases y el modelado de los datos forman parte importante del diseño del sistema, pues nos brindan una ayuda irremplazable al modelar con anticipación los procesos que se realizaran en la fase de implementación del sistema. Aunque implícitamente, en los diagramas de secuencia se tiene que hablar de cómo se realizaran las interfaces de usuario, queda pendiente aún por definir esta parte tan importante del sistema, ya que será la presentación hacia el usuario y es donde el usuario tiene que interactuar, por lo que debe de resultar lo mas amigable y sencillo posible para poder obtener un mejor aprovechamiento del sistema, no importando que nivel de conocimientos en materia de uso de herramientas web tengan los usuarios finales.

3.6 Diseño de la interfaz de usuario

Para la realización de estas interfaces, es necesaria la utilización de las páginas JSP(Java Server Pages), solución del lenguaje Java para la creación de objetos con las características deseadas para su publicación en web, teniendo contacto con ellas por medio de los navegadores web comerciales como Internet Explorer, Netscape Comunicator, Mozilla, Opera, etc.

Con las páginas JSP se pueden realizar aplicaciones web ejecutadas desde variados servidores web, los cuales pueden estar en casi cualquier plataforma conocida, debido a que están basadas en el lenguaje multiplataforma Java. Las JSP están compuestas básicamente de código HTML y en ocasiones XML combinados con etiquetas especiales que contienen sintaxis java de manera especial, por lo que para su elaboración no se necesitan mas recursos que los utilizados en la creación de una página HTML.

Las páginas JSP corren del lado del servidor por lo que éstas tienen que ser compiladas como un programa escrito en Java, pero con la diferencia que ésta compilación la realiza el servidor contenedor de JSP en tiempo real, al ejecutar alguna de éstas y como resultado de dicha compilación se obtiene una clase que finalmente es un Servlet, con las capacidades y beneficios que éstos últimos tienen.

El número de páginas JSP que se requieren para la realización de una aplicación web varía dependiendo del número de pantallas o interfaces que se le muestren al usuario y la posibilidad de rehusar estas mismas. Para el sistema de Administración de Documentos se realizaron, en algunos casos de uso, mas de una página JSP ya que algunos contienen varios pasos por los cuales el usuario debe pasar; solo en algunos casos se reutilizaron páginas JSP, las cuales despliegan información muy similar pero en diferente situación.

Como se ha mencionado, el contenedor que se utiliza para la realización del SAD, es el Jakarta-tomcat-3.2.3, y las páginas alojadas en dicho contenedor de Servlets y jsp se encuentran en el contexto llamado SAD que se encuentra debajo del directorio llamado webapps, mientras que los paquetes que contienen los JavaBeans y Servlets controladores están en la carpeta *'/webserver-root/webapps/SAD/WEB-INF/classes'* .

Esta colocación de los elementos sigue el estándar para la creación de Archivos de Aplicaciones Web(WAR) [J2EEAT], y es generalizado para todos los servidores contenedores de JSP, lo que garantiza que estos archivos pueden ser transportados a cualquier servidor web que siga la especificación de J2EE para la publicación de aplicaciones web basadas en la tecnología de los Servlets.

Respecto a los nombres usados en las páginas JSP, se eligieron identificadores representativos de la misión de presentación que tienen dentro de la aplicación web, así como la funcionalidad con respecto a los JavaBeans que accedan, con el fin de crear un ambiente propicio para el mantenimiento de la misma aplicación.

Para aumentar la sencillez del código, se escribió en algunas ocasiones la funcionalidad de una página, en varias JSP con lo cual se tiene un funcionamiento aislado, el cual se conjunta en una sola interfaz, con lo cual también se induce a la reutilización de pequeños componentes de presentación. Ejemplos claros y muy utilizados dentro de la aplicación, son el árbol de navegación y el encabezado que contiene los valores del usuario que se encuentra dentro del sistema, éstos elementos aparecen en la mayoría de las pantallas y sería innecesario colocar el mismo código JSP en todas las páginas que contengan dicho, ahorrando tiempo de mantenimiento y corrección de errores.

De esta manera se concluye la fase de diseño del sistema, lo cual da pie para la continuación del proyecto continuando con la fase de realización de la implementación en la cual se construirán los elementos descritos dentro del diseño. La primera actividad es la creación de la base de datos en base al modelo obtenido anteriormente sobre el servidor de bases de datos relacionales MySQL, con la utilización del cliente de mysql. Posteriormente se realizará la construcción de las clases en base al diseño y la funcionalidad con que se diseñaron, de la misma manera se realizarán las interfaces web al usuario, con lo cual se tendrán los elementos para hacer la integración de los diferentes componentes de la aplicación dentro del servidor contenedor de Servlets y JSP, es decir las clases Java, Servlets, JSP, y archivos .jar que se utilizan en el desarrollo de la aplicación web.

CAPÍTULO

4

Implementación del Sistema de Administración d Documentos

Dentro de éste capítulo se comprende la construcción del sistema utilizando el patrón de diseño MVC, por una parte la creación de la base de datos que resguardará la información manejada en el sistema, los Java Beans(Modelo) y Servlets(Controlador), y por último las interfaces gráficas de usuario(Vista).

4.1 Creación de la base de datos

Por ser el estándar de uso en base de datos, la base de datos utilizada en éste sistema se implanta en un servidor de bases de datos relacional(RDMS). Dicha estandarización provee de un lenguaje unificado para la manipulación y consulta de datos bajo bases de datos de éste tipo, lenguaje que comprende una serie de instrucciones generales para todos los servidores de éste tipo y cuyo nombre es SQL.

SQL tiene una serie de sentencias que deben de funcionar en todos los servidores relacionales, ya que es una condición necesaria para que algún servidor se le denomine como “relacional”. Servidores como Oracle(en sus diferentes versiones), Informix, DB2, Sybase, PostgreSQL, mSQL, SAPDB, MySQL, MS SQL Server, Access, etc. forman la basta colección de opciones que existen en el mercado para el manejo de bases de datos.

La mayoría de los mencionados se tratan de Software propietario, lo cual tiene el inconveniente de tener costo por su uso, así como en ocasiones su documentación. Como se mencionó en el primer capítulo de éste trabajo de tesis, uno de los objetivos esenciales del desarrollo de este sistema es la utilización de software libre el cual no eleve los costos de producción y mantenimiento del sistema. Por esta razón las opciones se limitan descartando a servidores que tienen gran poder de procesamiento de datos como Oracle y Sybase pero que tienen la mencionada restricción.

Por esta razón, para el manejo de los datos en el sistema se eligió entre algunos otros(mSQL,

PostgreSQL) al servidor de bases de datos relacional MySQL, que aunque no es completamente un producto freeware, si se permite su uso en cualquier condición además de que funciona sobre las plataformas más comunes dentro de las posibilidades del autor(Win32 y Linux x86).

A continuación se describirá el proceso para la creación de la base de datos sobre MySQL.

4.1.1 Creación de la base de datos en MySQL

Para la creación de la base de datos sobre MySQL, se utiliza un script SQL(SAD.sql) para la creación de las tablas sobre la base de datos el cual se muestra a continuación.

```
drop table files;
create table files(id_file int not null primary key AUTO_INCREMENT,
                                        nombre varchar(50) not null,
                                        folder int not null,
                                        fecha_pub date,
                                        tipo varchar(30),
                                        comentario varchar(100),
                                        icon varchar(60),
                                        size varchar(20),
                                        grupo int not null,
                                        publicador int not null);

drop table folder;
create table folder(folder int not null primary key AUTO_INCREMENT,
                            parent int not null,
                            nombre varchar(30),
                            comentario text,
                            fecha date,
                            creador int,
                            grupo int not null);
drop table grupos;
create table grupos(grupo int not null primary key AUTO_INCREMENT,
                            nom_gpo varchar(30) not null UNIQUE);
drop table usuarios;
create table usuarios(id_usuario int not null primary key AUTO_INCREMENT,
                            login varchar(16) not null UNIQUE,
                            password varchar(16) not null,
                            first_name varchar(64),
                            last_name varchar(64),
                            grupo int not null,
                            nivel_us int  not null,
                            email varchar(64) not null,
                            numGafete varchar(10) not null,
                            block int not null);
drop table version;
create table version (version_id int not null primary key AUTO_INCREMENT,
                            id_file int not null,
                            nombre_vers varchar(50),
                            fecha_pub date not null,
                            size_vers varchar(20),
                            publicador int not null);

insert into usuarios
(login,password,first_name,last_name,grupo,nivel_us,email,numGafete,block)
                                                    values
('admin','admin','Administrador','Admin',0,1,'admin_sad@imp.mx','00000',0);

insert into folder (parent,nombre,comentario,fecha,creador,grupo)
                                        values
(0,'Raiz','Folder Raiz Solo accesible para administradores, sobre esta carpeta se crean las
carpetas de los diferentes grupos',NOW(),1,0);
```

Listado C4-1 Script para la creación de la base de datos.

La ejecución de este script crea las tablas files, folder, grupos, versión y usuarios además de que crea un usuario de tipo administrador con nombre de usuario y contraseña admin, además de dar de alta el folder raíz al cual solo es posible ingresar si el usuario es administrador.

Dando por hecho que el servidor y el cliente de mysql están instalados en el equipo la forma de ejecutarlo en entorno Linux/UNIX es la siguiente:

```
#mysql –h host –u sad –p SADDB < SADDB.sql
Password:***
```

Donde sad es el nombre de usuario que se debe crear en MySQL con password 'imp', SAADB la base de datos que se debe crear y SADDB.sql el script sql contenido dentro del contexto de la aplicación en el directorio <root-webapp>/Ut.

Recapitulando lo escrito en el capítulo tres, de diseño de la aplicación, donde se definieron las tablas y campos de la base de datos, tenemos cinco entidades relacionales para realizar la funcionalidad de la aplicación, que se enumeran a continuación:

Tabla files

Almacena la información relacionada con los archivos cargados en el servidor, un identificador de archivo(id_file), campo no nulo que tiene la función de llave primaria por lo que se le asigna la funcionalidad de 'AUTO_INCREMENT', función exclusiva de mysql que no pertenece en los estándares de SQL. Tiene un campo en el cual guarda el nombre del archivo, este campo no debe ser nulo y es alfanumérico; el folder al que pertenece, fecha de publicación obtenida del servidor de base de datos MySQL a través de la función NOW() propia de MySQL, tipo de archivo(tipo MIME) el cual se obtiene a partir del *MultiPartDataRequest* que se utiliza al cargar el archivo al servidor, comentario que escribe el publicador al momento de cargar el archivo, icono según el tipo de archivo el cual se define a partir del tipo MIME obtenido con anterioridad y que asigna un icono asociado o en caso de no estar contenido en los existentes coloca uno de archivo desconocido. En el campo size se inserta la cantidad de bytes que mide el archivo, este dato también se obtiene al cargar el archivo. Sobre el campo grupo se inserta el valor numérico del grupo al que pertenece dicho documento, y finalmente en el campo publicador se coloca el identificador del usuario que publico dicho documento.

Tabla folder

Esta tabla contiene los datos asociados a los folders, con el fin de tener una identificación y estructuración del sistema de archivos manejado dentro del sistema. La llave primaria de esta tabla es el campo folder, que almacena valores enteros y tiene asignada la propiedad de MySQL 'AUTO_INCREMENT' aumentando en uno el valor que se inserta en este campo al momento de realizar inserciones en los otros campos.

El campo parent recibe valores enteros y no acepta valores nulos debido a que esta es la referencia del folder insertado, indica cual es el folder del que proviene, es decir su folder padre, del cual únicamente la raíz no tiene dicha referencia y se asigna como él mismo. Este campo también es útil para la creación de la interfaz gráfica de navegación entre los directorios. Los folders incluidos deben de contar con identificación por parte del usuario, por lo que la tabla usa un campo de nombre del folder y un comentario que su creador adjunta al momento de crearlas, así como la fecha de creación, creador y grupo al que pertenece.

Esta tabla tiene una relación fuerte con la tabla de files, pues la llave primaria de ésta, es una llave secundaria(FK) de la tabla files, debido a que un folder puede contener muchos archivos, y a un archivo le corresponde pertenecer solamente a una carpeta.

Tabla versión

Bajo el soporte de esta tabla se realiza el control de versiones sobre los documentos que se cargan sobre el servidor, después del proceso de reconocimiento de documentos con el mismo nombre de archivo y en caso positivo, se almacena el documento cargado con un nombre nuevo que indica el número consecutivo de versión, y los datos se guardan en esta tabla junto con un número consecutivo que es el identificador de la versión.

Estos datos son el nombre real del archivo, como fue guardado en el servidor, el identificador del documento al que esta referenciado, la fecha de publicación, tamaño e identificador del usuario que la colocó en el servidor.

Esta tabla tiene una relación fuerte con la tabla "files" ya que la llave primaria de files funciona como llave foránea en ésta tabla (id_file), así mismo, con la tabla de usuarios pues utilizamos el identificador de usuario para indicar quien publico las versiones.

Tabla usuarios

La tabla de usuarios almacena los datos importantes sobre éstos. Cada usuario cuenta con un identificador de usuario(o numero de usuario) el cual tiene la función de ser llave primaria y bajo este identificador se puede conocer que usuario dio de alta un archivo, una versión y creo un folder.

Por su parte el campo de login almacena los nombres de usuario(login) de éstos. Este campo es 'unique' (no pueden repetirse dos cadenas) y es el identificador alfanumérico de cada usuario. El password es necesario para darle cierta seguridad al usuario de que nadie ingresará a su cuenta sin saber la contraseña.

Los campos de first_name y last_name, corresponden al nombre de pila del usuario, útiles para la identificación dentro del sistema por parte de los demás usuarios. El siguiente campo indica cual es el grupo al que pertenecen el usuario, éste es llave foránea de la tabla grupos.

El contenido del campo nivel_us determina cual es el nivel o rol de usuario, se asigna un valor numérico distinto a cada uno de los cuatro niveles de usuario(Administrador, Moderador, Publicador, Lector). Dentro de los datos importantes del usuario se encuentra el e-mail ya que es el medio de comunicación entre el sistema y el usuario, el campo dedicado a esto se llama 'email' y acepta valores alfanuméricos. El campo numGafete almacena el número de trabajador, esto tiene el fin de que posteriormente pueda validarse este dato y se tenga una certeza de que los usuarios registrados sean únicamente trabajadores el IMP.

Finalmente, el campo block guarda un valor numérico de 1 ó 0 en caso de que el usuario este bloqueado o desbloqueado.

Tabla grupos

Como se mencionó en la justificación del proyecto, dentro del área de Ingeniería del IMP se realizan una gran cantidad de proyectos al mismo tiempo por lo que se requiere que el SAD tenga la capacidad de aislar de alguna manera el acceso a los documentos por medio de grupos de usuarios que únicamente puedan acceder a los documentos que estén dentro del grupo al que están adscritos. La tabla grupos contiene únicamente dos campos, el primero es el identificador de grupos y el segundo corresponde al nombre asignado al grupo.

4.2 Realización de las clases en Java

Después de haber construido la base de datos que resguardará los datos utilizados generados por el sistema, es necesario crear la lógica de negocio que accederá a dichos datos, así como la lógica de control de la aplicación. Esta lógica se realiza con la creación de clases Java, que en específico es la programación de Servlets en caso del control de la aplicación y JavaBeans en el caso del acceso a la información contenida a la base de datos, el envío de mensajes electrónicos y otras tareas que tienen que realizarse por medios mas confiables evitando la programación de lógica mas elaborada y compleja sobre la presentación de la aplicación, es decir sobre las páginas JSP.

A continuación se muestran las clases construidas mas importantes, clasificadas por paquetes con el fin de simplificar la descripción de su funcionamiento y construcción, de la misma manera como se clasificaron en la fase de diseño.

4.2.1 Clases del paquete Auth

Dentro de éste paquete se consideran las siguientes clases:

- *PassiveCallbackHandler*
- *RdbmsLoginModule*
- *RdbmsPrincipal*
- *RdbmsCredential*
- *ControlAuth*

La figura C4-1 muestra los diagramas de las clases mencionadas con anterioridad.

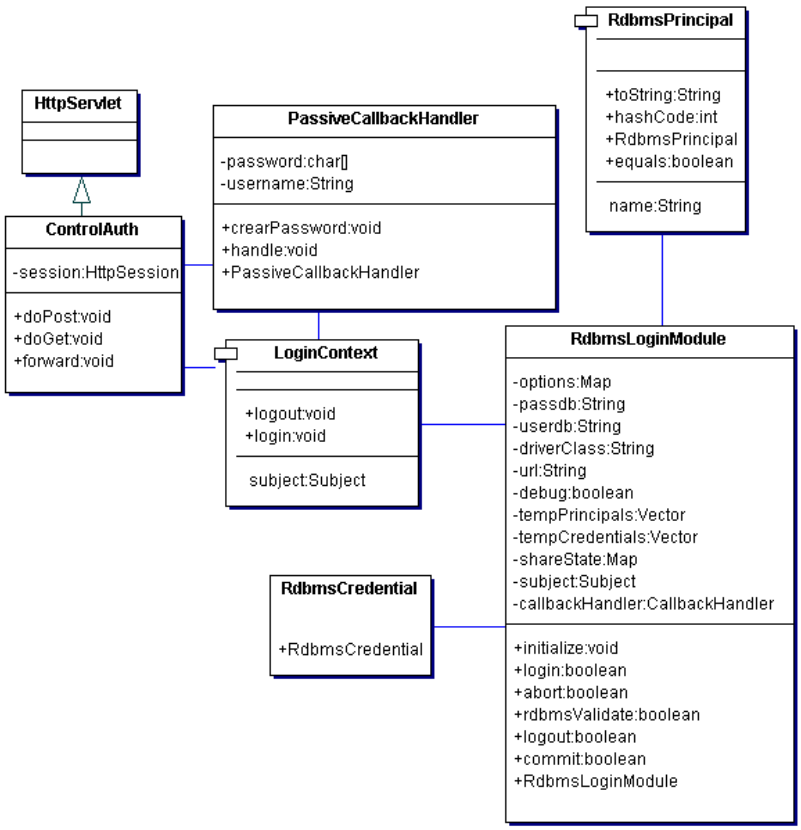

Fig. C4-1 Diagrama de clases del paquete auth.

La clase *ControlAuth* es un Servlet que se encarga de controlar el proceso de autentificación, extiende de la clase *HttpServlet* y éste debe de hacer la sustitución de los métodos *doGet()* y *doPost()* para hacer uso de las peticiones que se realicen al servidor desde los navegadores web, por medio de POST o GET. El método *forward()* se encarga de redireccionar al usuario a la pagina de entrada en caso de una exitosa autentificación, o bien a la página de autentificación en caso contrario. Tiene un atributo llamado *sesion* el cual es de tipo *HttpSession*, y dentro de éste objeto se colocan todos los atributos que se desean guardar en la sesión creada.

Las clases *PassiveCallbackHandler*, *RdbmsLoginModule, RdbmsPrincipal, RdbmsCredential*, conforman el bloque de autentificación por medio del Servicio de autentificación y autorización para Java. *PassiveCallbackHandler y LoginContext* son creadas por el Servlet controlador y estas a su vez instancian indirectamente a *RdbmsLoginModule, RdbmsPrincipal y RdbmsCredential*. De las cuales *RdbmsLoginModule* realiza el proceso de autentificación contra la base de datos y *RdbmsPrincipal* y *RdbmsCredential* son empleadas para transportar los datos obtenidos de la base de datos para su extracción en el Servlet.

4.2.2 Clases del paquete folder

Este paquete contiene las siguientes clases, que se describirán a continuación en dicho orden y en base a los diagramas que se presentarán. Los diagramas se dividirán para su mejor apreciación, al final de éste apartado se desplegará el diagrama de clases completo.

- *ControlFolder*
- *modFile*
- *modVersion*
- *modFolder*
- *busquedaDB*
- *fileData*
- *insertaFile*
- *obtRuta*
- *folderData*
- *imageFiles*
- *imageFolder*

En la figura C4-2 se muestran los diagramas de las clases modFile, modVersion y ControlFolder.

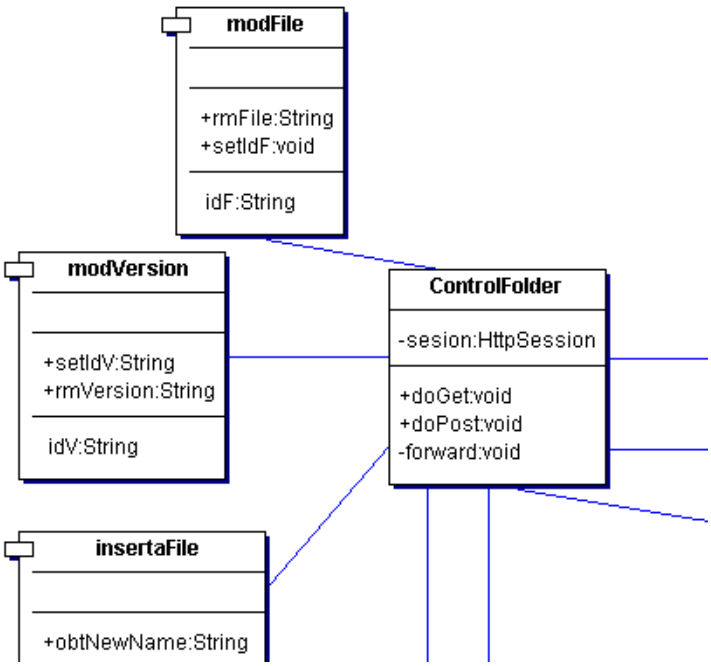

Fig. C4-2 Diagramas de clases modFile, modVersion y ControlFolder.

La clase *ControlFolder* es el Servlet controlador de los procesos destinados a la publicación, eliminación y modificación de los documentos dentro del SAD, así como la manipulación, creación y eliminación de los folders que contienen dichos documentos, realizando los procesos de lógica de manejo de la aplicación. Esta clase extiende de la clase *HttpServlet* y debe de sustituir a los métodos públicos *doGet()* y *doPost()* para hacer uso de las peticiones GET y POST que provengan de los clientes o navegadores web. El método *forward* es el encargado de redireccionar al usuario entre las páginas de interfaz de usuario, cabe señalar que éste Servlet como los demás usados dentro de la aplicación desarrollan el método *forward* por medio de la ejecución del método *RequestDispatcher* de *ServletContext*.

Por su parte la clase *modFile* es un JavaBean que tiene la propiedad *idF* lo cual incluye a un atributo *String* llamado idF y el método de acceso *getIdF(),* además cuenta con el método de acceso *setIdF()*. Contiene también el método *rmFile* que se encarga de eliminar archivos, este a su vez crea un objeto *modVersion* y llama al método *rmVersion* para eliminar a todas las versiones que tienen como referencia de archivo al que se pretende eliminar.

La clase *modVersion*, como se ha dicho, esta encargada de eliminar versiones especificas que están cargadas sobre el sistema. Se trata de otro JavaBean con la propiedad *String idV* , su respectivo método de acceso *setIdV()* y un método ya mencionado, llamado *rmVersion*, el cual realiza la tarea de eliminación de versiones.

La figura C4-3 muestra los diagramas de las clases modFolder, busquedaDB y fileData.

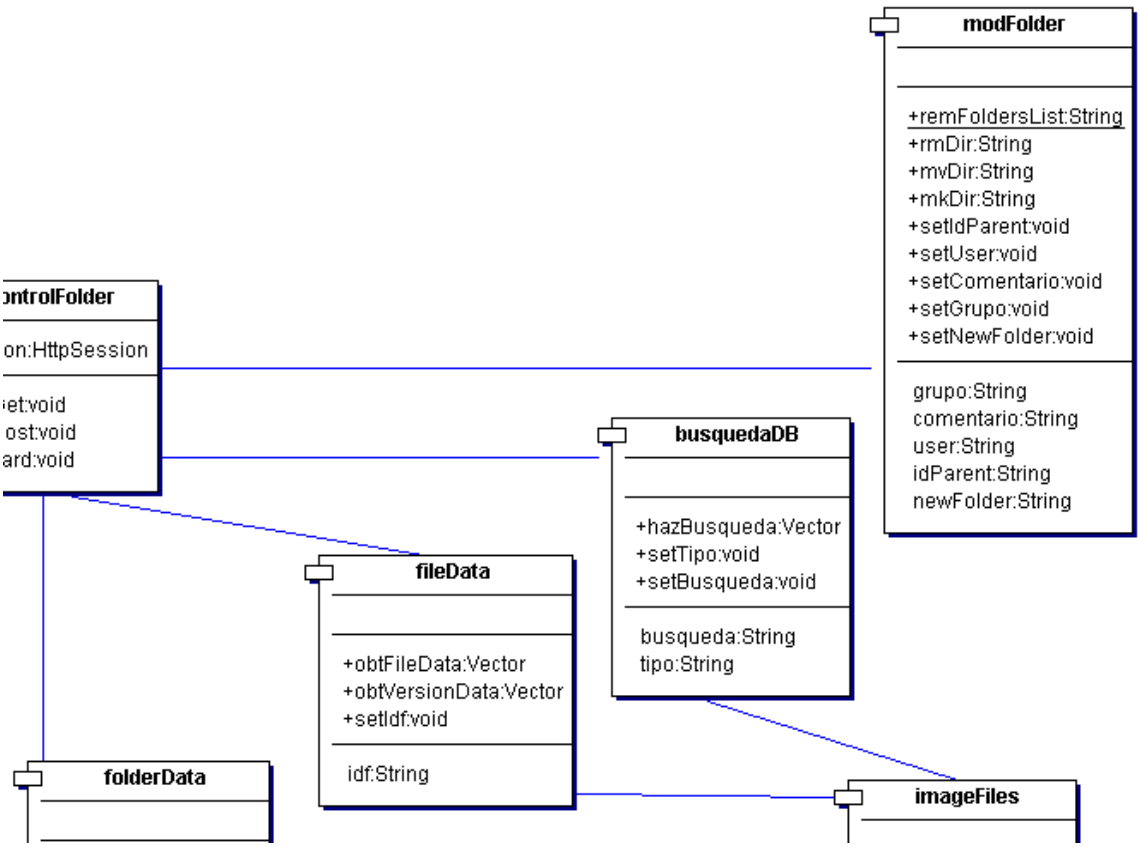

Fig. C4-3 Diagramas de clases fileData, busquedaDB y modFolder.

La clase *modFolder* es un JavaBean que tiene las propiedades *newFolder, idParent, user, comentario y grupo*(que contiene sus atributos String en todos los casos y sus métodos getXXX), así como sus métodos de acceso setXXX. Dentro de la funcionalidad de la clase se encuentra el método *mkDir()*, que realiza las respectivas conexiones y modificaciones a la base de datos(por medio de la clase *ConnDB*) para crear nuevos folders sobre el SAD.

La siguiente funcionalidad en cuanto a modificaciones la realiza el método *rmDir()*, el cual elimina la información de algún folder en especifico, y con la asistencia del método *remFoldersList()* se eliminan todos los archivos contenidos dentro de la carpeta indicada. Este proceso es recursivo, ya que al eliminar un folder, se deben de eliminar los hijos de éste, así como el conjunto de archivos y versiones contenidas en ellos.

El método restante, *mvDir()*, se encarga de modificar el nombre de los folders, únicamente realizando una actualización a los registros de la base de datos.

La clase *busquedaDB* es un JavaBean que tiene las propiedades *búsqueda* de tipo *String* y la propiedad *tipo* también de tipo *String*. La clase tiene la función de hacer búsquedas sobre la base de datos para obtener datos relacionados con cadenas de caracteres que los usuarios teclean por medio de la interfaz de usuario, realizando esta búsqueda sobre los nombres y comentarios de los archivos y folders contenidos en la aplicación. Este proceso lo realiza por medio del método

hazBusqueda() el cual regresa un objeto *Vector* que contiene objetos *imageFiles* con los datos de los archivos que coinciden con la búsqueda.

La clase *fileData* obtiene la información relacionada con los documentos desde la base de datos para ser desplegados por medio de las interfaces de usuario a la ejecución del método *obtFileData()*, de igual manera, el método *obtVersionData()* obtiene los datos de las versiones para el mismo fin. Por tratarse de un JavaBean tiene la propiedad *String id* y el método de acceso *setId().*

La figura C4-4 muestra los diagramas de las clases insertaFile, folderData y obtRuta.

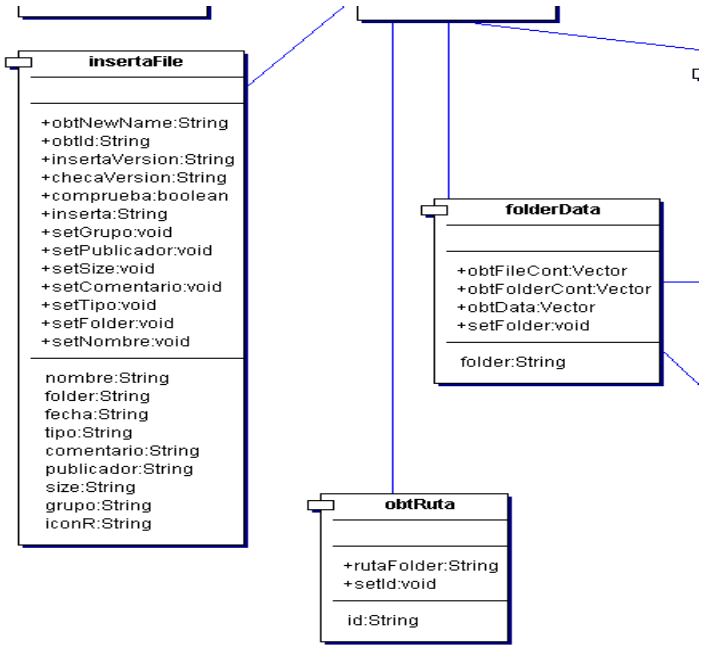

Fig. C4-4 Diagramas de clase de insertaFile, obtRuta y folderData.

Una de las clases corazón del sistema es el JavaBean *insertaFile*, ya que realiza la lógica de negocio de una de las funcionalidades mas notorias del sistema, es decir, la publicación de documentos. Como ya se dijo, se trata de un JavaBean que tiene las propiedades de bean *nombre, folder, fecha, tipo, comentario, publicador, size, grupo e iconR,* todas de tipo *String,* lo cual comprende sus respectivos atributos private, y sus métodos de acceso *getXXX()* y *setXXX()*.

Dentro de los métodos que contiene esta clase se encuentran *inserta*(), quien devuelve un objeto String de confirmación, y realiza la lógica de inserción de nuevos registros a la base de datos, ya sean documentos nuevos o versiones de documentos existentes en la base de datos del sistema. Dicha selección se realiza por medio de la ejecución del método *comprueba()* el cual regresa un objeto bolean, en este caso un valor verdadero si el documento que se desea publicar en cierto folder ya existe y por lo tanto es necesario catalogarlo como versión e insertarlo en la tabla como tal. Ahora bien es necesario conocer en número consecutivo de versión con que el documento anterior esta registrado, dicha tarea se realiza con el método *chekaVersion()* quien regresa el número siguiente de versión la cual corresponderá al documento en cuestión.

Debido a que en el servidor se almacenan los archivos separados por carpeta, no sería lo mejor sobrescribir los documentos cada que se publica una versión nueva en el sistema, por lo que se debe de modificar el nombre del archivo para así almacenarlo sin la pérdida de datos, contando con los documentos completos. Dicha modificación en el nombre físico como es guardado el

documento, así como el nombre que se coloca dentro de la base de datos, lo realiza el método *obtNewName()* el cual regresa un objeto *String* que contiene el nuevo nombre a utilizar.

La clase *obtRuta* únicamente se utiliza para obtener la ruta donde se encuentra algún documento o folder especifico, dicha clase es un JavaBean que cuenta con una propiedad llamada *id* y sus respectivos métodos *getId* y *setId* y un método llamado *rutaFolder* que regresa un objeto *String* que contiene la información requerida en una cadena de caracteres. Este proceso se realiza de manera recursiva buscando en la base de datos al padre de la carpeta donde se encuentra ubicado el usuario, y a su vez el padre de éste, hasta llegar a la carpeta raíz.

Por último, el método *insertaVersion()* realiza la lógica de inserción de nuevos registros en la tabla versión de acuerdo con los datos obtenidos anteriormente.

La clase *folderData* es un JavaBean que tiene la propiedad *folder,* la cual es un objeto String y tiene sus respectivos métodos de acceso *getFolder()* y *setFolder()*. Básicamente este bean obtiene los datos respectivos de un folder que se le especifique a través de su propiedad. Los métodos *obtFolderCont(), obtFileCont() y obtData()* buscan el contenido del folder, sus subdirectorios, archivos y versiones regresándolo a través de objetos del tipo *Vector* que contienen objetos de tipo *imageFiles* e *imageFolders*.

4.2.3 Clases del paquete árbol

En este paquete están comprendidas las clases ConsTree, OutilTree y Cooltree, que como ya se explicó realizan la función de despliegue del árbol de navegación. La figura C4-5 muestra los diagramas de las clases ConsTree, OutilTree y Cooltree.

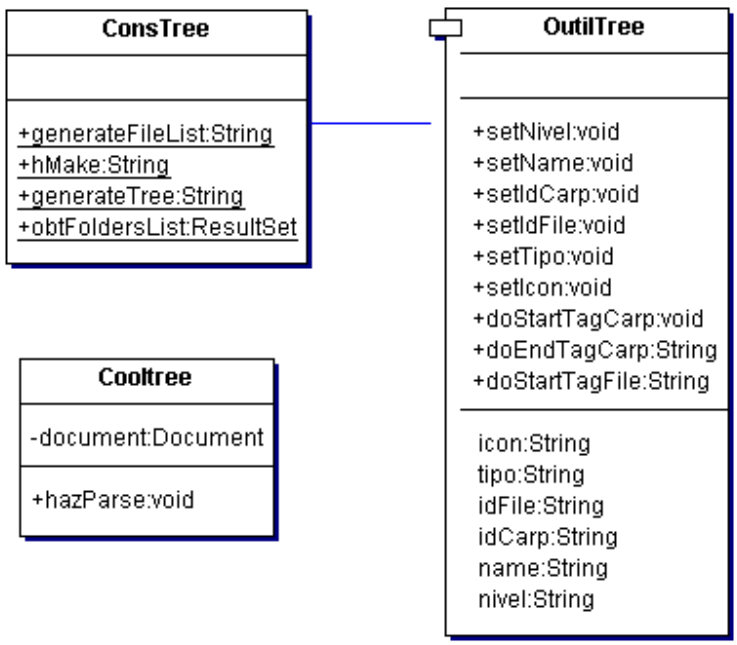

Fig. C4-5 Diagramas de clase del paquete arbol.

En la explicación de los diagramas de secuencias se mostró el proceso que se sigue para la realización del árbol de navegación por lo que se explicaran únicamente los elementos que comprenden las clases que realizan este proceso.

La primera clase en actuar en este caso es *ConsTree* con la participación de *OutilTree*. *ConsTree* consta de cuatro métodos, el principal de ellos es el método *hMake()* la cual regresa un objeto *String* que expresa el documento XML que representa al sistema de directorios y que a su vez escribe dicho documento XML en el directorio <root-webapp>/WEB-INF/classes/res ejecutando al método *generateTree()* y *getFolderList()* que regresa un objeto *ResultSet*, los cuales mediante un proceso en el cual el primero ejecuta al segundo y a su vez se ejecutan a ellos mismos recursivamente para profundizar a través de todo el sistema de directorios que se estructura a partir base de datos.

A su vez, *generateTree()* ejecuta al método *generateFileList()* quien regresa objetos de tipo *String* que representan los documentos contenidos en la carpeta que se este obteniendo información. Por su parte la clase (JavaBean) *OutilTree* se instancia desde *ConsTree* insertándole las propiedades *nivel, nombre* e *idCarp* en el momento de generar las etiquetas XML correspondientes a los directorios dentro del método *generateTree()* y las propiedades *icon, tipo, idFile* y *name* para generar las etiquetas XML correspondientes al los documentos dentro del método *generateFileList()*. A su vez, se ejecutan los métodos del bean *OutilTree doStartTagCarp()* y *doEndTagCarp()* quienes regresan objetos *String* para generar las etiquetas XML de inicio y final de directorios, y el método *doStartTagFile()* quien construye el par de etiquetas XML correspondientes a los documentos publicados.
La segunda fase del proceso de construcción la realiza la clase *Cooltree* con la transformación del documento XML generado por la clase *ConsTree*, y con base a las transformaciones indicadas en una hoja de estilo(Cxsl.xsl) que se encuentra ubicada en <raiz-webapp>/WEB-INF/classes/res. Dicha transformación es generada por el uso del API de transformaciones XSLT que emplea parsers contenidos en el paquete javax.xml.parsers, eventos SAX(Simple API for XML) y DOM(Document Oriented Model), así como transformaciones contenidas dentro del paquete javax.xml.transform.

Esta clase de transformación cuenta únicamente con un atributo de clase llamado document, el cual es un objeto DOM (org.w3c.dom.Document), y un método llamado *hazParse()*, quien se encarga de todo el proceso de transformación y de la escritura del documento que resulta de dicha acción sobre el directorio <raiz-webapp> /js/, un archivo javaScript(js) que contiene los datos necesarios para que éste sea interpretado por el navegador Web y despliegue el resultado deseado sin la necesidad de utilizar subprogramas como Applets Java dentro de la interfaz Web, las cuales dependen del cliente en mayor medida que la tecnología javaScript.

4.2.4 Clases del paquete users

Dentro de este paquete se encuentran las clases que definen a los objetos que se asocian con la lógica de negocio referente al manejo de los usuarios dentro del SAD, estas clases son:

- *checkGroup*
- *ControlUsers*
- *gruposData*
- *imageGroup*
- *imageUsuarios*
- *insertaUsuario*
- *modGrupos*
- *pRegUsuarios*
- *usuariosData*

A continuación se muestra en la figura C4-6 los diagramas de las clases *checkGroup, gruposData, imageGruop y ControlUsers*, posteriormente se describe brevemente el contenido y funcionamiento de éstas clases.

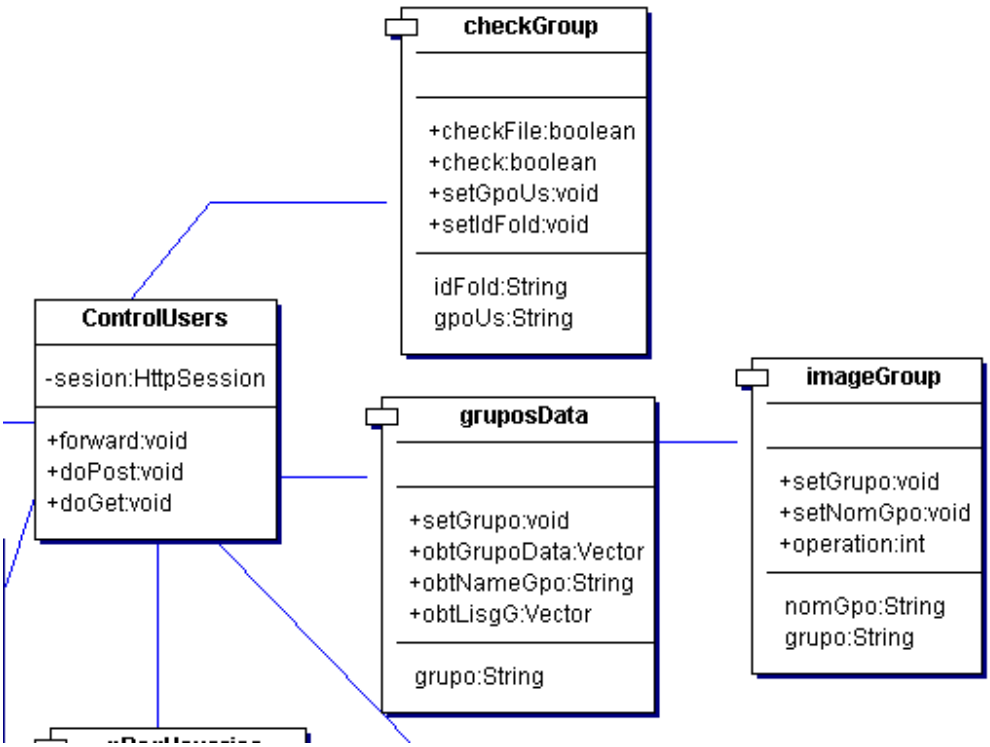

Fig. C4-6 Diagramas de clase del paquete users, checkGroup, ControlUsers, gruposData e imageGroup.

La clase *ControlUsers* consiste en un Servlet controlador, por lo que extiende de la clase *HttpServlet*, el funcionamiento es muy similar a los controladores que ya se han descrito en éste capítulo y contempla a los mismos métodos considerados: el método *forward()* para realizar las redirecciones, y realiza las sustituciones de los métodos *doGet()* y *doPost()* para gestionar las peticiones que provienen de los clientes web.

Por su parte la clase *checkGroup* mostrada en la figura C4-6, es un JavaBean que tiene las propiedades de bean *idFold* y *gpoUs*, de tipo String, con sus respectivos métodos de acceso. Dentro de su funcionalidad están los métodos *checkFile()* y *check().* Los cuales sirven para determinar si un folder o un documento corresponden al grupo que pertenece el usuario que desea acceder a ver el contenido, realizar modificaciones o bien de descargar los documentos contenidos en el folder que desea.

El método *checkFile()* regresa un objeto del tipo bolean verdadero en caso de que el usuario si corresponda al mismo grupo al que pertenece el documento. Estos métodos se ejecutan desde las interfaces de usuario para lograr un control en todo momento de las personas que accedan a la información contenida.

Al igual que el anterior, el método *check()* retorna un objeto bolean, pero éste método verifica la misma concordancia entre el usuario y el directorio que desea observar o sobre el que quiere realizar modificaciones.

Cabe señalar que en los roles de usuario se comprende un tipo de usuario administrador, el cual tiene acceso a todos los directorios y documentos del sistema, por lo que éstos pertenecen a un grupo especial. Al realizar la verificación dentro de éstos métodos, dichos usuarios siempre obtienen un valor de verdadero en dicha operación, teniendo la posibilidad de ingresar a cualquier sitio del sistema de directorios y realizar las modificaciones que crea convenientes.

La clase *gruposData* es un JavaBean con una propiedad de bean *String* llamada *grupo*, y tres métodos que realizan tareas de obtención de datos referentes a los grupos contenidos en la tabla grupos de la base de datos, el método *obtGrupoData()* regresa un objeto *Vector* que contiene un objeto *imageGrupos* con los datos de un grupo en específico, al igual que el método *obtListG()*, objeto *Vector* el cual contiene el mismo número de objetos *imageGrupos* que grupos registrados dentro del sistema.

En la figura C4-7 se muestran los diagramas de las clases gruposData, imageGroup, pRegUsuarios, usuariosData e imageUsuarios, descritos posteriormente.

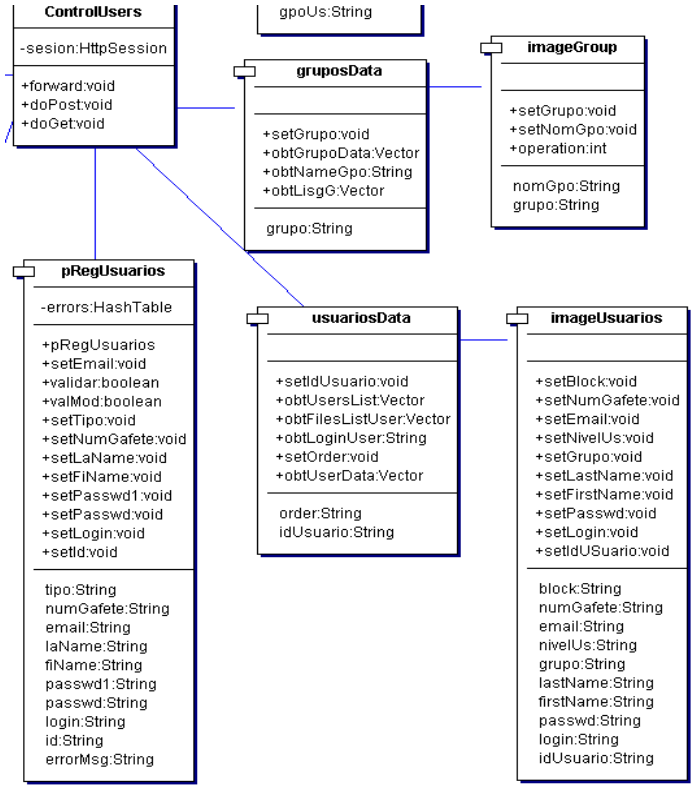

Fig. C4-7 Diagramas de clase del paquete users, usuariosData, imageUsuarios y pRegUsuarios

El JavaBean *usuariosData* tiene como propiedades al objetos String *order* e *idUsuario* con sus respectivos métodos de acceso *get* y *set*, además de los métodos *obtUsersList(), obtFilesListUser(), obtLoginUser()* y *obtUserData().*

El cometido principal de ésta clase es la obtención de datos varios desde la tabla usuarios en la base de datos, los cuales se despliegan a través de las interfaces gráficas, éstos procesos se llevan a cabo específicamente con los métodos *obtLoginUser()* y *obtUserData()*. El método *obtUsersList()* es utilizado para listar los usuarios que están activos en el sistema, ya sea en forma general o bien por grupo, éste regresa un objeto *Vector* el cual contiene objetos *imageUsuarios* los cuales contienen las propiedades de los usuarios guardados dentro de la base de datos. Por su parte el método *obtFilesListUser()* obtiene de la tabla files de la base de datos, los documentos que el usuario ha publicado en algún momento, si es que él ha realizado esta acción. Esto nos es útil para llevar un historial de los documentos que cada usuario coloca dentro del sistema de directorios del SAD.

En el proceso alta de usuarios es necesario comprobar que los valores que se están introduciendo sean correctos, por lo que se debe realizar una validación de dichos datos. Para los procesos de validación existen dos alternativas, la primera y mas sencilla se puede realizar del lado del cliente o navegador web por medio del uso de código javaScript, y la segunda y mas complicada, pero más segura debido a que se realiza únicamente con código Java puro se realiza del lado del servidor.

Este sistema se trató de realizar utilizando únicamente programación Java, utilizando al máximo los beneficios y estabilidad que nos da éste lenguaje en nuestras aplicaciones y únicamente en el caso del despliegue de la representación gráfica del sistema de directorios, se utilizó código javaScript. En el caso de las validaciones del lado del servidor, el JavaBean *pRegUsuarios* realiza dicha validación dentro de su método *validar()* que regresa un objeto bolean indicando el éxito o fracaso del procesamiento de una solicitud de alta de usuario. Dentro de esta verificación de datos, se compara el nombre de usuario que se intenta dar de alta con los existentes dentro de la base de datos, ya que no se pueden ni deben de repetir nombres de usuario pues causaría graves conflictos. Por otro lado se verifica dentro de éste mismo método que los valores obtenidos en la forma no sean cadenas vacías, o que tengan el formato adecuado, como el número de gafete que debe ser numérico y el correo electrónico que debe de contener ciertos caracteres(como la "arroba"[@]) para ser tomada en cuenta como una dirección válida.

Dicho proceso de validación se explica en el diagrama de secuencias del caso de uso "alta de Usuario" contenido en el capítulo anterior.

El JavaBean *insertaUsuario* mostrado en la figura C4-8, contiene métodos que realizan varias de las tareas mas importantes de éste paquete, la inserción y actualización de usuarios. Como se trata de un bean, contiene sus respectivas propiedades las cuales consisten en los datos necesarios para dar de alta un usuario, y los métodos de acceso correspondientes a dichas propiedades.

El método *insUser()* inserta en la base de datos a nuevos usuarios, después de haberse realizado la debida validación de los datos a insertar, *updateUser()* se encarga de realizar actualizaciones de datos que pueden realizar los usuarios, datos personales que no afectan sus permisos de usuario o bien el grupo al que pertenecen, como su nombre, número de gafete y correo electrónico. Contrariamente, los métodos *updateGpoUser(), updatePermUser(), blockUser() y unblockUser()* modifican valores que si afectan estos puntos por lo que solo tienen acceso a ellos los usuarios con privilegios de administración. El método *updateGpoUser()* modifica el valor del grupo al que esta adscrito el usuario, *updatePermUser()* modifica permisos cambiando el rol del usuario, y únicamente los usuarios administradores no pueden ser afectados por estas modificaciones. Finalmente *blockUser() y unblockUser()* modifican el valor de block en la tabla usuarios de la base de datos, para permitir o denegar el acceso al usuario que se modifique.

Dentro de la mayoría de éstas acciones se notifica al usuario por medio del uso de la clase sndMail del paquete util, la cual envía un correo electrónico al usuario con los cambios que se le han realizado a su cuenta del SAD.

La última clase por describir dentro de éste paquete es *modGrupos* que es donde se crean y eliminan grupos de usuarios del sistema, otra de las herramientas administrativas para el control de usuarios. Se trata de otro JavaBean con dos propiedades de tipo *String* que modifica la tabla grupos de la base de datos para anexar o eliminar registros referentes a los grupos de usuarios.

En la figura C4-8 se muestran los diagramas de las clases insertaUsuario, modGrupos, y ControlUsers del paquete users.

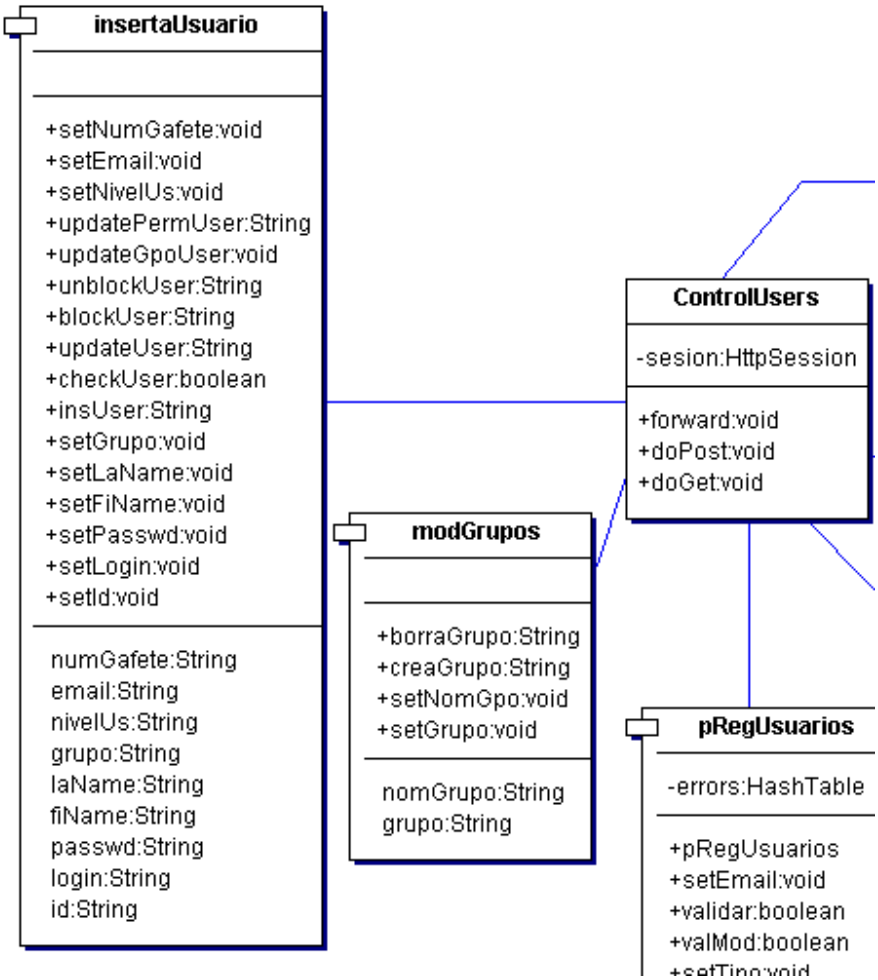

Fig. C4-8 Diagramas de clase del paquete users, insertaUsuario y modGrupos.

4.2.5 Clases del paquete utils

El paquete utils contiene clases que se instancian en clases de otros paquetes, pues sus tareas son auxiliares dentro de los procesos, desde el despliegue de información en la interfaz gráfica, hasta las muy útiles y necesarias conexiones y gestión de éstas con la base de datos.

En la figura C4-9 se muestran los diagramas de clase de los objetos correspondientes al paquete utils. Las clases mas sencillas son *chtoString* y *creaPassword*, la primera únicamente cambia los valores del nivel de usuario que se obtienen de la base de datos como un dato numérico, a un texto que nos dice mas acerca del nivel(Administrador, Moderador, Publicador o Lector). Este proceso se realiza varias veces dentro de toda la aplicación.

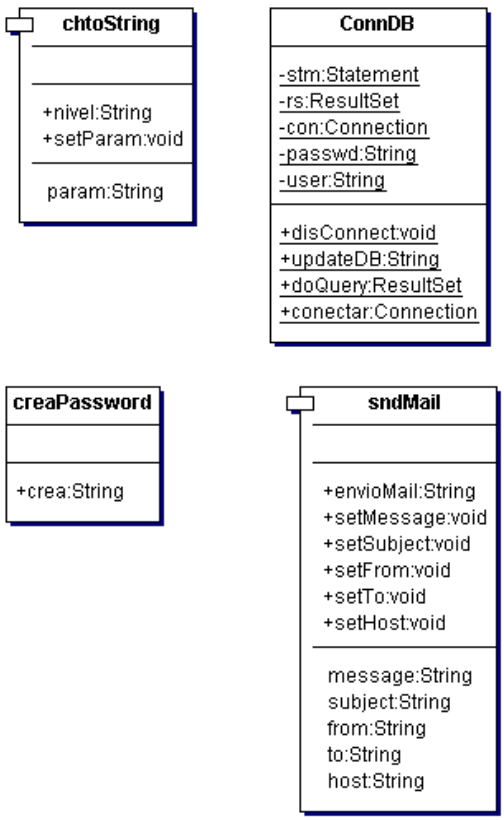

Fig. C4-9 Diagramas de clase del paquete utils.

Por su parte *creaPassword* genera una contraseña de ocho caracteres y que se utiliza en la alta de nuevos usuarios. Debido a que los usuarios no tienen la posibilidad de darse de alta por su propia mano con el fin de restringir la cantidad de usuarios y la seguridad de los documentos y en general de la información del sistema, únicamente los usuarios administradores pueden dar de alta a nuevos usuarios, por lo que se genera una contraseña diferente y aleatoria para cada usuario, la que se envía por correo electrónico junto con sus datos de acceso.

La clase *sndMail* tiene la misión de enviar los mensajes de correo electrónico que sean necesarios

dentro del sistema de administración de documentos a través del uso del API Java Mail que da toda la infraestructura para enviar este tipo de mensajes a través de un servidor SMTP, que en el caso del SAD, corresponde al servidor imp.mx de correo electrónico.

Finalmente, la clase mas importante del paquete debido a su utilización es *ConnDB*. Esta maneja las conexiones con el servidor de bases de datos MySQL y realiza las actualizaciones y peticiones por parte de las otras clases del SAD que lo requieran, de modo que se puede decir que es la clase mas instanciada del sistema. Los cuatro métodos que contiene pueden realizar la conexión con la base de datos, la actualización de registros, inserción de registros, consulta de registros y la eliminación de las conexiones abiertas anteriormente con el fin de no mantener conexiones abiertas innecesariamente.

Hasta este punto, se ha realizado la descripción de las clases que se codificaron para llevar a cabo el funcionamiento completo del SAD según los requerimientos y casos de uso que se recabaron en la fase de diseño del sistema, y dentro del desarrollo de la aplicación los nuevos requerimientos que se generan implícitamente.

En seguida, se encuentra descrita la parte correspondiente a la creación de las interfaces de usuario creadas para este sistema, con lo cual se completa la construcción del sistema de Administración de documentos.

4.3 Implementación de la interfaz Web

Por el hecho de que el sistema debe ser accesado por la Intranet del IMP, la interfaz de usuario se programó y construyó a partir de la tecnología Java Server Pages, para la publicación de páginas Web, que consiste en insertar fragmentos de código Java sobre páginas HTML con una sintaxis específica y especial, no muy diferente a la programación de una clase Java común.

La tecnología JSP(Java Server Pages) utiliza código Java que se compila y ejecuta del lado del servidor web, por lo que el cliente web solo recibe e interpreta como código HTML, de ésta manera el usuario en ningún momento se percata de todos los procesos que tienen que ser necesarios para la presentación de la pantalla que está observando, aunque pueda tener acceso al código fuente de dicha página desde su navegador web.

Para la realización de éstas interfaces no es necesaria la utilización de grandes recursos o interfaces de programación(IDE), pues únicamente es necesario un editor de texto sobre el cual se escriben las aplicaciones al igual que en el caso de la programación de las clases Java. En la construcción del SAD, básicamente se emplearon el editor *gedit*, vi, y pico sobre Linux y en las ocasiones que se tuvo que trabajar sobre plataforma Windows se utilizó el notepad, y para la edición de los botones e imágenes utilizadas, el editor de imágenes gipm sobre Linux y Paint Shop Pro sobre Windows.

A continuación se describe un poco de la acción que realizan algunas de las interfaces de usuario mas importantes contenidas en el SAD, así como su funcionamiento e interacción con sus similares. Dichas interfaces están clasificadas por casos de uso para facilitar su comprensión y los casos de uso que se realizan por detrás de la aplicación no se muestran en esta sección.

4.3.1 Interfaces de usuario del caso de uso Autentificar

El primer contacto del usuario con el sistema es la pantalla de login, que se encuentra en la dirección http://<url-servidor-web>:puerto/SAD donde <url-servidor-web> corresponde a la dirección IP o nombre de dominio donde se encuentra alojada la aplicación, puerto el número de puerto por el que escucha el servidor. La figura C4-10 muestra ésta pantalla, en la que se le solicita al usuario que proporcione su nombre de usuario y clave(o contraseña) para tener acceso al sistema.

Fig. C4-10 Página inicial y de acceso al sistema.

En el caso especifico de la pantalla incluida en la figura C4-10 el usuario admin se encuentra tratando de ingresar al sistema, por lo que tiene que escribir su nombre de usuario(admin) y clave(admin).

Este formulario HTML, es enviado por el método POST hacia el Servlet ControlAuth, que determina el tipo o nivel de usuario que ingresa al sistema a través del proceso de autentificación y obtención de datos de sesión explicados en la parte correspondiente a la construcción del sistema.

Dependiendo del tipo de usuario, es conducido a las diferentes interfaces correspondientes a los cuatro diferentes roles de usuario, éstas interfaces son *admin.jsp* mostrada en la figura C4-11, *moderador.jsp* mostrada en la figura C4-12, *pub.jsp* mostrada en la figura C4-13 y *lect.jsp* mostrada en la figura C4-14.

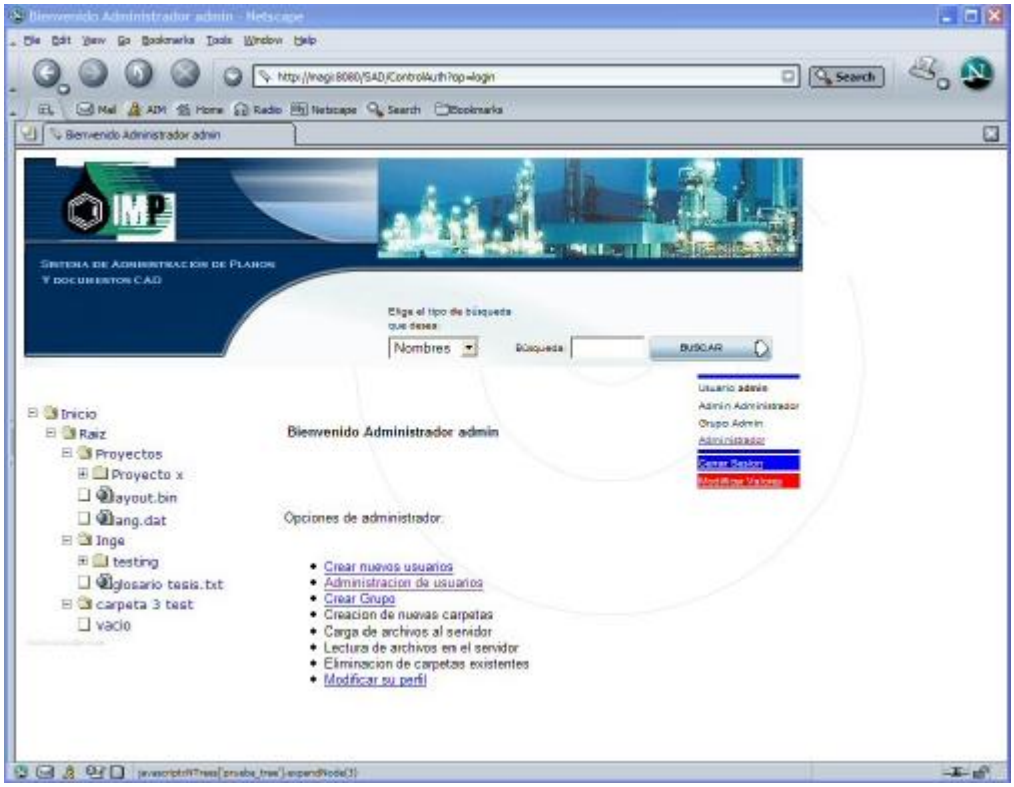

Fig. C4-11 Interfaz de bienvenida del tipo de usuario Administrador.

Fig. C4-12 Interfaz de bienvenida del tipo de usuario Moderador.

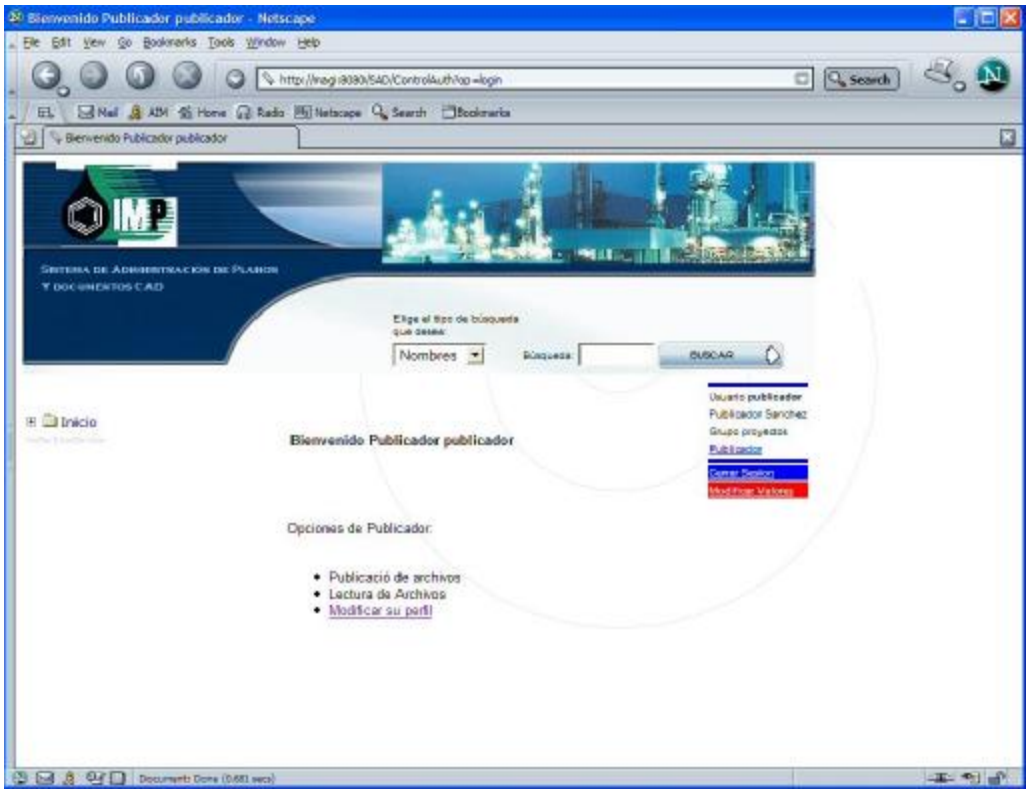

Fig. C4-13 Interfaz de bienvenida del tipo de usuario Publicador.

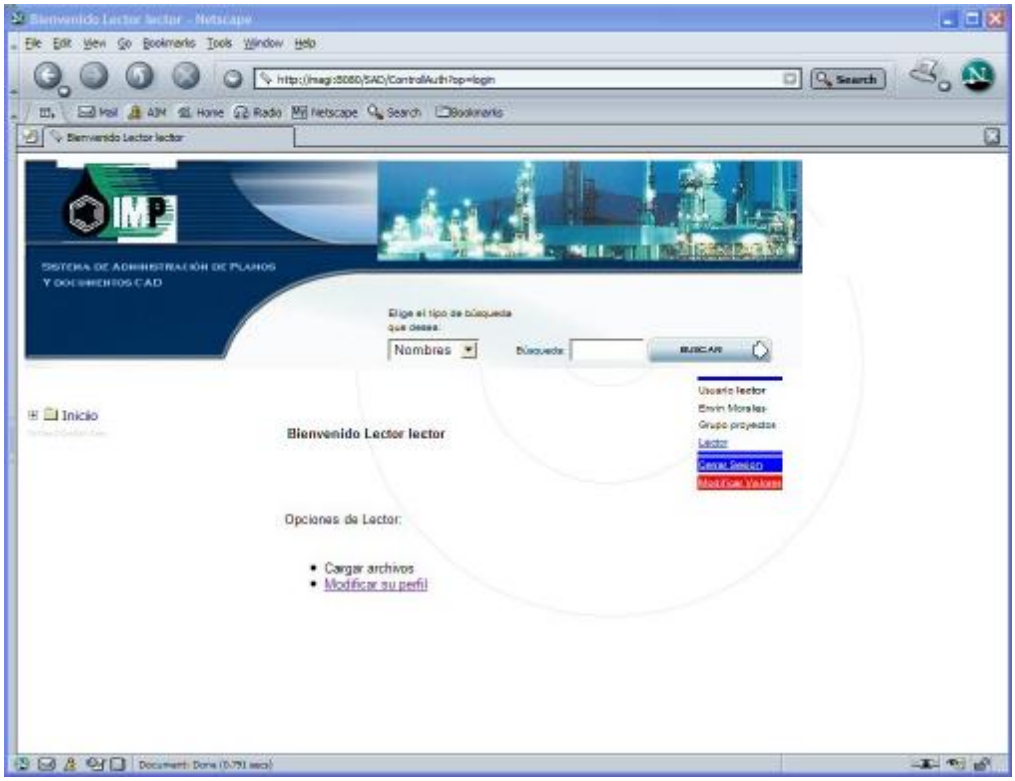

Fig. C4-14 Interfaz de bienvenida del tipo de usuario lector.

Como se observa, dichas interfaces contienen las acciones que puede realizar cada usuario, desde el acceso a todas las acciones del sistema por parte de los usuarios de tipo administrador, hasta la restricción de casi todas las acciones del tipo de usuario lector, quien únicamente tiene permisos para observar y descargar archivos contenidos dentro de los directorios que pertenezcan al grupo al que se encuentra adscrito.

4.3.2 Interfaces de usuario del caso de uso Búsqueda

Este caso de uso es accesible para todos los usuarios, se teclea una cadena de caracteres que contengan algún contenido representativo a lo que se desea encontrar dentro del sistema y el tipo de búsqueda que se realizará, de esta manera la aplicación busca dentro de la base de datos todas las coincidencias que correspondan a la cadena de caracteres mencionada dentro de los nombres y comentarios que identifican a cada archivo y que además correspondan al grupo de usuarios al que esta asignado dicho usuario.

La figura C4-15 muestra la interfaz de búsqueda que se muestra en casi todas las páginas dentro del SAD y que es el formulario de inicio de búsqueda. Corresponde este pequeño fragmento de código a la página JSP llamada *busqueda.jsp.*

Después de enviar la petición de búsqueda, es procesada por el Servlet controlador correspondiente y se obtienen los resultados en caso de que existan. La figura C4-15 contiene el resultado de una búsqueda en la selección inferior de la pantalla.

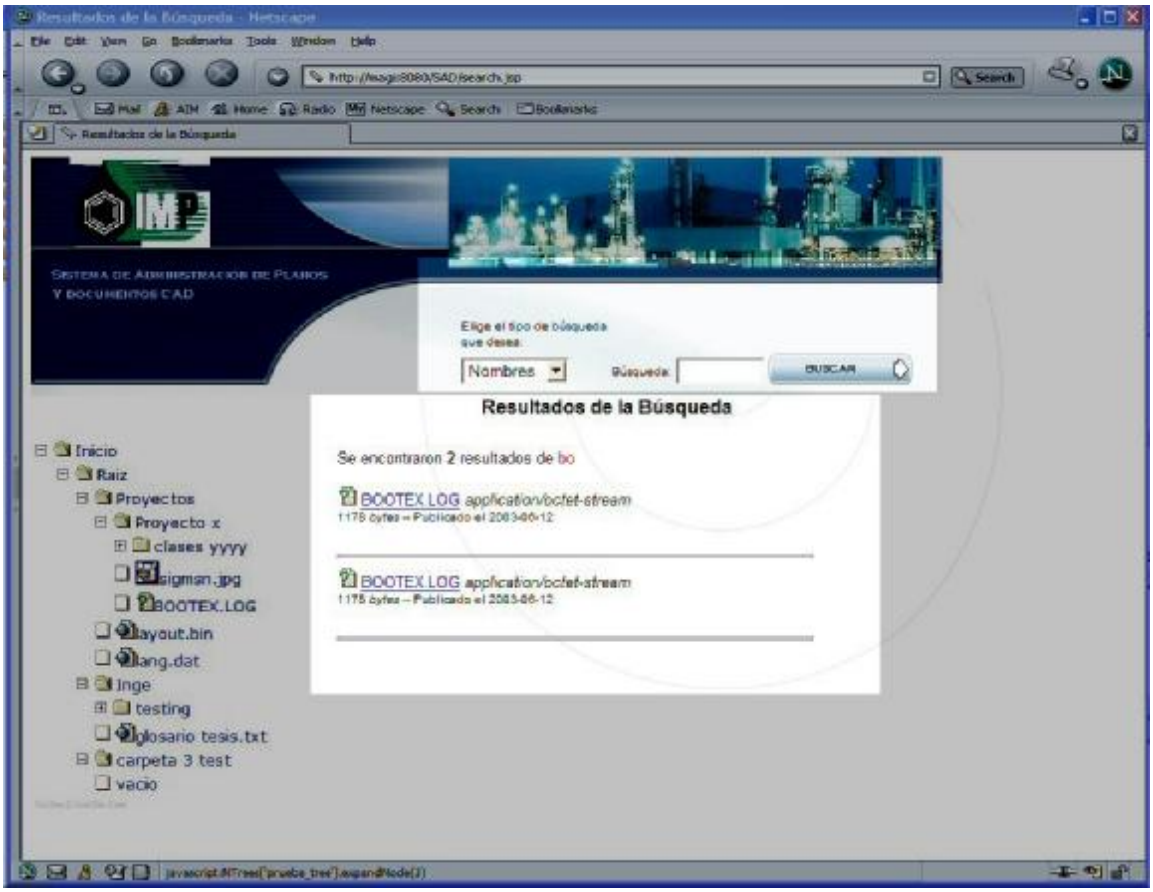

Fig. C4-15 Interfaz de búsqueda de archivos en el sistema.

4.3.3 Interfaces de usuario del caso de uso Desplegar Árbol de navegación

La interfaz de navegación presentada de manera de un árbol jerárquico y construida a partir del contenido de la tabla folders de la base de datos por las clases *ConsTree, OutilTree y Cooltree*, se presenta al usuario en las páginas web donde el usuario tiene contacto con información relacionada a los directorios o a los documentos publicados. Como se mencionó, está basada en la tecnología javaScript y se genera a partir de tres archivos js, de los cuales el primero llamado *tree_nodes.js* que contiene la programación fuerte para lograr las acciones del arbol jerárquico. El segundo archivo llamado *tree_format* contiene la configuración de dicho arbol, tamaños, posiciones e iconos que debe contener. Finalmente el último y no menos importante corresponde al generado por la aplicación que contiene la estructura del árbol y es llamado *cooltree.js* que esta contenido junto con los dos antes mencionados dentro de directorio js/ que se encuentra debajo del directorio raíz de la aplicación.

La figura C4-16 muestra la ubicación y un ejemplo de un árbol jerárquico desplegado mostrando el contenido de algunos directorios.

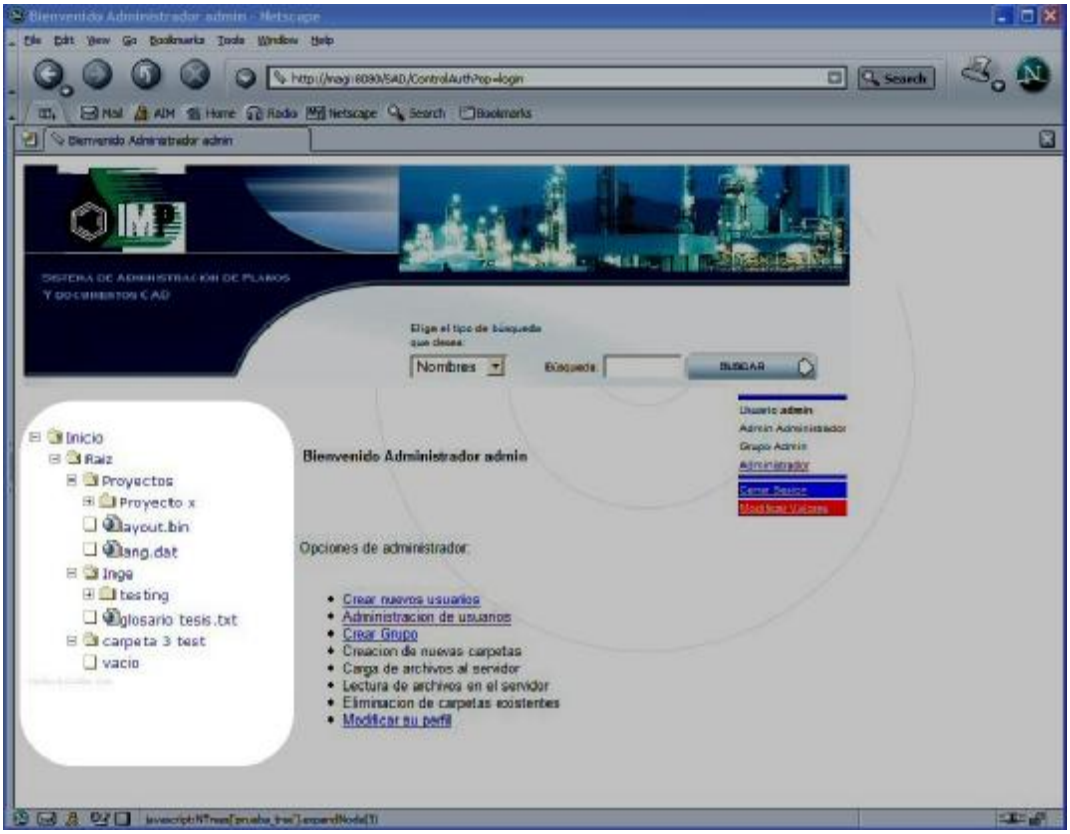

Fig. C4-16 Interfaz del caso de uso Desplegar Arbol de navegación.

4.3.4 Interfaces de usuario del caso de uso Desplegar contenido archivos

La página JSP encargada del despliegue del contenido de los archivos es *fileD.jsp.* Esta página despliega el nombre del documento, quien lo publica, una descripción breve, la fecha en que fue publicada, la ruta de directorios donde esta ubicado, la opción de descargar el archivo y las versiones que se encuentran en el servidor, así como el usuario que realizo dicha publicación.

Adicionalmente, para los usuarios administradores y moderadores de grupo, se cuenta con la opción de eliminar el documento, así como sus versiones.

Dentro de ésta página JSP son llamados varios de los métodos correspondientes a clases que forman parte del paquete folder. Un ejemplo de ésta pagina se muestra en la figura C4-17 la cual corresponde a una página de usuario administrador.

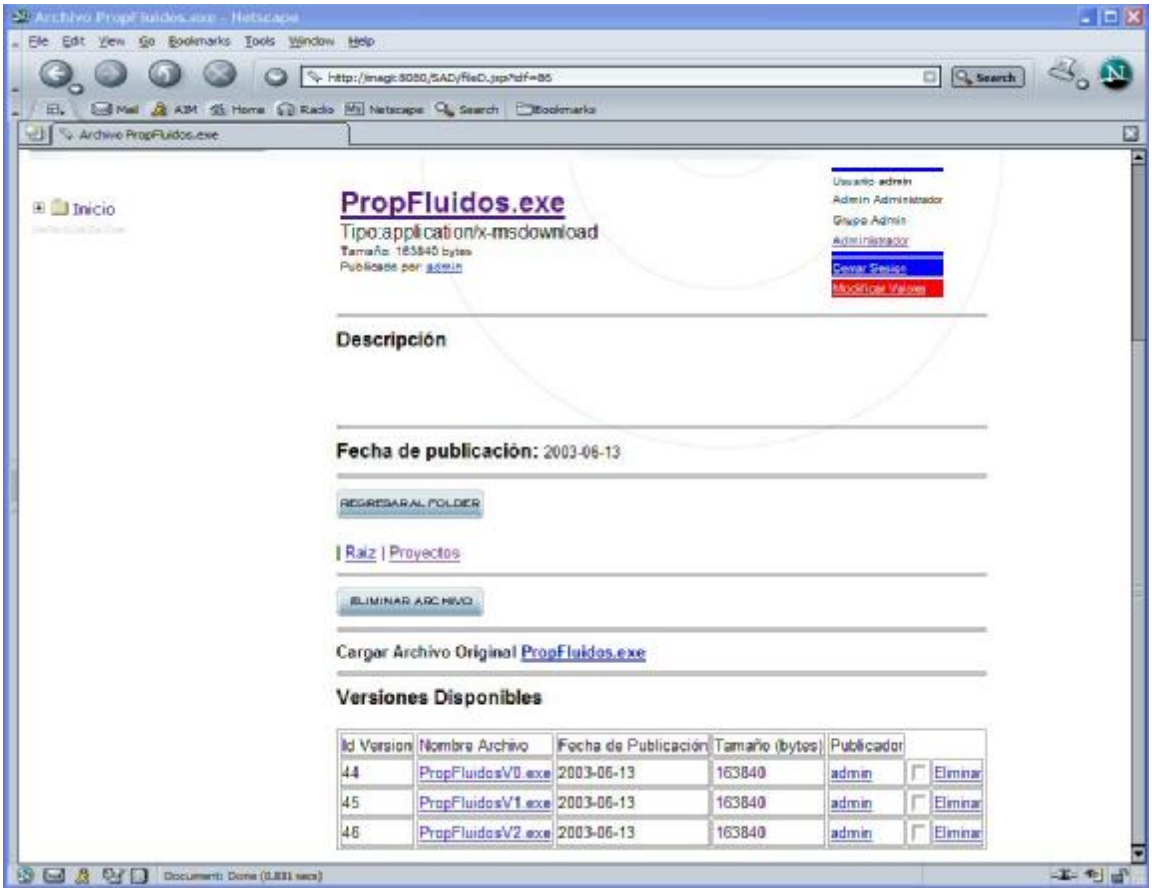

Fig. C4-17 Interfaz del caso de uso Desplegar contenido de archivos.

En la figura C4-17 se puede observar que el usuario autentificado tiene la capacidad de eliminar tanto documentos como versiones del archivo PropFluidos.exe.

4.3.5 Interfaces de usuario del caso de uso Desplegar contenido de folders

La figura C4-18 muestra el contenido de un folder ejemplo con su información correspondiente: nombre, descripción, fecha de creación, ruta en donde se encuentra, y su contenido en subdirectorios y archivos cargados sobre éste. Así como las modificaciones que se le pueden realizar: carga de nuevos documentos, creación de subdirectorios, renombrar y eliminar subdirectorios.

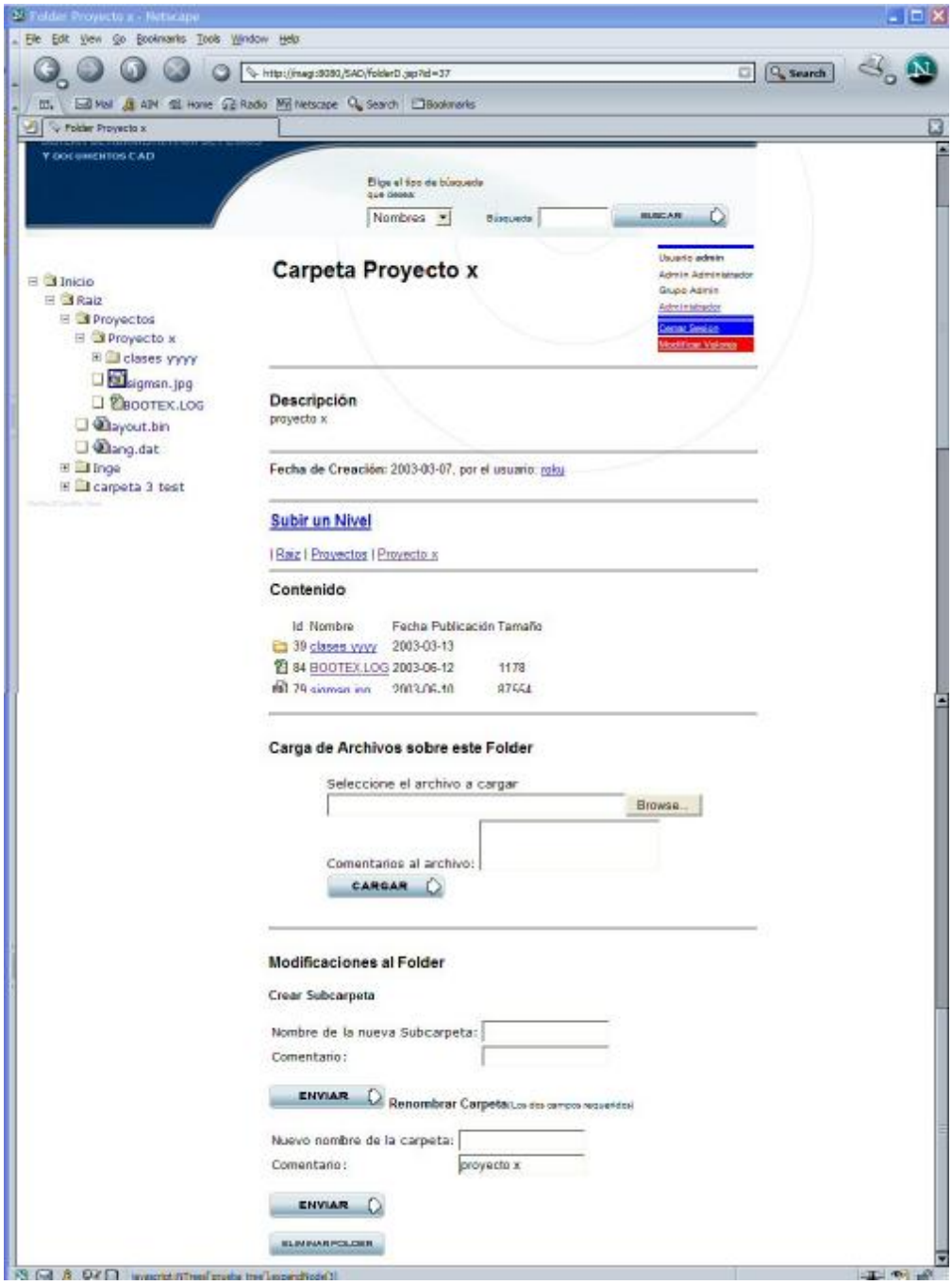

Fig. C4-18 Interfaz de usuario del caso de uso desplegar contenido de folders.

Esta interfaz es una de las mas completas pues llama a métodos de clases incluidas en el paquete folder y tiene comunicación con varios casos de uso, carga de documentos, renombre de folder, creación de nuevos subdirectorios, eliminación y despliegue de información entre algunos otros.

4.3.6 Interfaces de usuario del caso de uso modificar datos de acceso e identificación

Como se mencionó, es necesario que los usuarios tengan la opción de modificar sus datos así como la clave de acceso o contraseña. Para tener acceso a esta sección únicamente es necesario seguir la liga que se encuentra en la mayoría de las páginas de la interfaz web, en la figura C4-19 se señala dicho punto de acceso a la interfaz mostrada en la figura C4-20.

La interfaz habilitada para estas modificaciones corresponde a la página JSP *modValores.jsp.* En esta interfaz se realizan validaciones del lado del servidor para descartar que los usuarios coloquen valores no autorizados o valores nulos dentro de los campos de texto correspondientes a sus datos, en las figuras C4-20 y C4-21 se muestran algunas pantallas donde fueron colocados datos erróneos y se muestra la respuesta del servidor a esto.

Fig. C4-19 Enlace a la interfaz de modificación de valores.

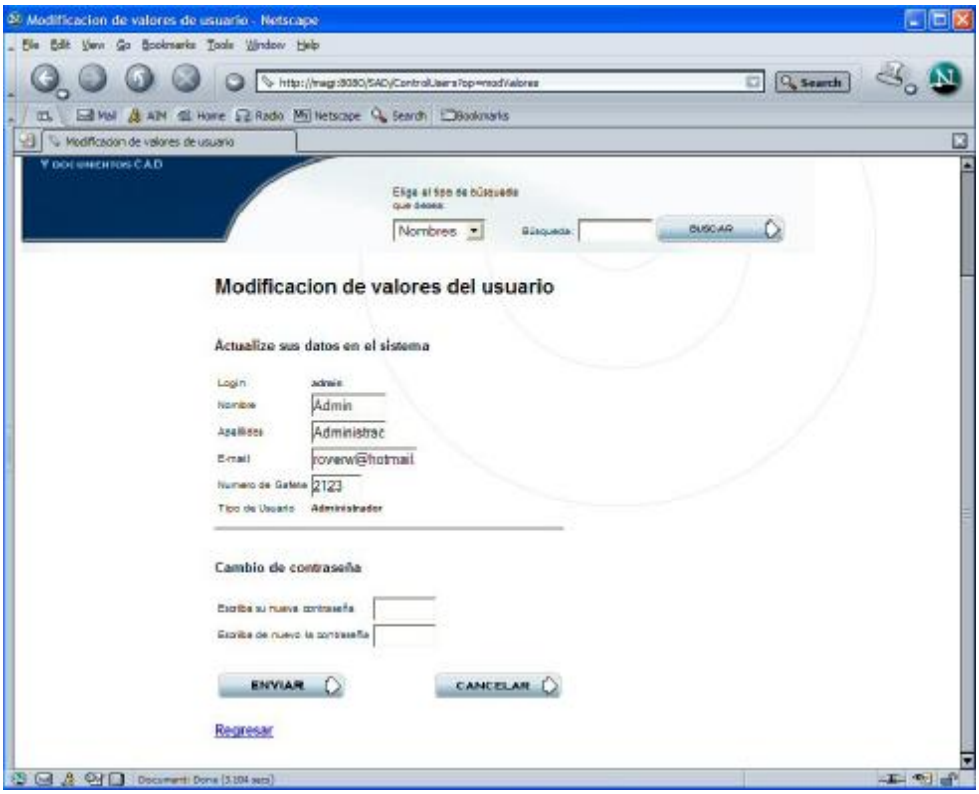

Fig. C4-20 Interfaz completa de modificación de valores.

Específicamente, en la figura C4-21 se omitió la escritura sobre el campo correspondiente al nombre del usuario, y en ésta se muestra la respuesta de la validación que realiza el servidor respecto a esta falta.

Fig. C4-21 Error en la escritura del nombre del usuario.

Por su parte la figura C4-22 muestra el error arrojado por el servidor al intentar escribir una dirección de correo electrónico con formato inadecuado.

Fig. C4-22 Error en la escritura del correo electrónico y su respuesta.

Al realizar las modificaciones de forma correcta, se envía un correo electrónico al usuario donde se le notifican los cambios que realizo y su nueva contraseña. Finalmente, el usuario es redireccionado a su página de inicio.

4.3.7 Interfaces de usuario del caso de uso Upload(Carga de documentos)

La carga de documentos dentro del sistema de administración de documentos se realiza sobre el directorio donde se desea colocar por lo que forma parte de la interfaz de usuario del despliegue de información de folders o directorios.

Como se indico en la construcción de las clases Java, el Servlet controlador es el encargado de recibir la petición del cliente sobre el cual se envía el archivo a cargar sobre el servidor, se guarda el nombre y datos importantes del documento o archivo en la base de datos, guardando también el archivo binario sobre un directorio físico dedicado al directorio donde se colocó, para posteriormente redireccionar al usuario a la página del directorio donde decidió publicar el documento con los cambios respectivos a la carga del nuevo archivo o versión.

Las figuras C4-23 y C4-24 muestran la página del folder antes de cargar un archivo y cuando éste ya fue publicado en el sistema.

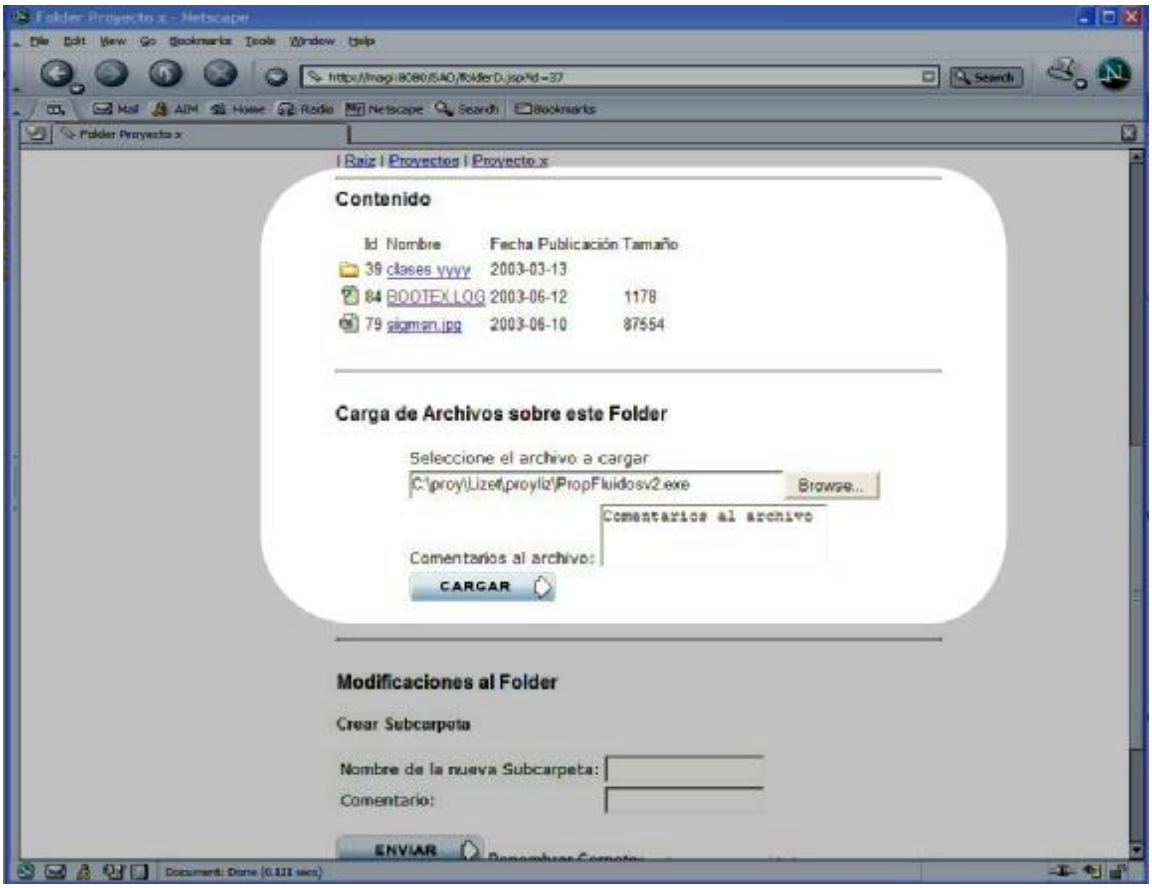

Fig. C4-23 Contenido de un directorio antes de cargar un archivo.

Fig. C4-24 Contenido de un directorio después de cargar un archivo.

4.3.8 Interfaz de usuario del caso de uso Cambio de nombre de folder

Esta interfaz esta contenida también en la página de información de los directorios que se despliega para usuarios administradores y moderadores de grupo. Consiste en un campo de texto para escribir el nuevo nombre del folder en que se encuentra el usuario y otro campo de texto dedicado a los comentarios que pueda realizar dicho usuario, respecto al nuevo nombre de la carpeta.

En la figura C4-25 se muestra dicha interfaz acompañada de la interfaz de creación de nuevos subdirectorios y la eliminación del directorio actual.

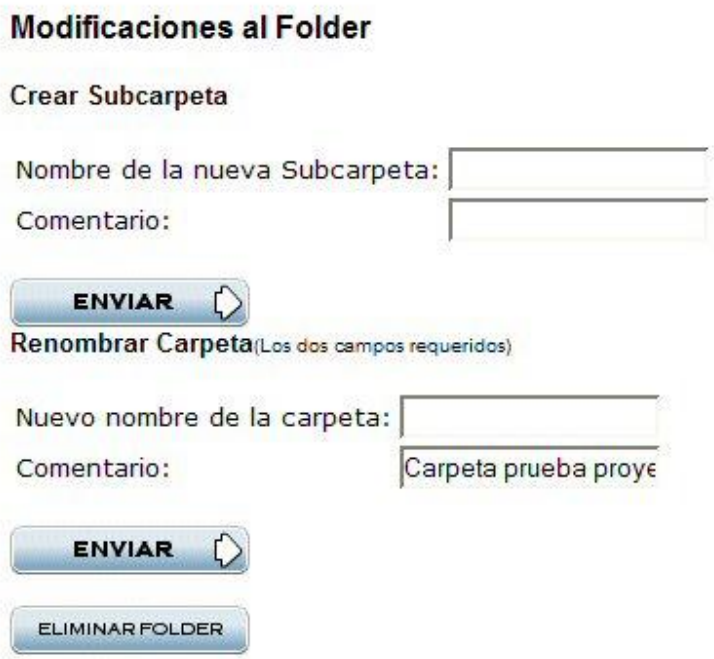

Fig. C4-25 Interfaces de modificaciones a los directorios.

4.3.9 Interfaz de usuario del caso de uso Crear nuevo folder

Como se puede observar en la figura C4-26 la interfaz de creación de nuevos subdirectorios consiste en únicamente dos campos de texto sobre los cuales se escribe el nombre del nuevo directorio a crear y un comentario al respecto. Cabe señalar que los folders creados pertenecen al grupo al que pertenece el folder padre, y solo en caso del directorio raíz, al que únicamente tienen acceso los usuarios administradores del sistema se elige el grupo al que pertenecerá el directorio.

La petición de creación es recibida por el Servlet controlador, quien ejecuta los métodos correspondientes para poder colocar el nuevo folder sobre la base de datos para su registro y posteriormente el usuario es redireccionado hacia la misma página pero ya con los cambios aplicados.

En la figura C4-26 se muestra la creación de un directorio sobre la raíz de éstos por parte de un usuario administrador.

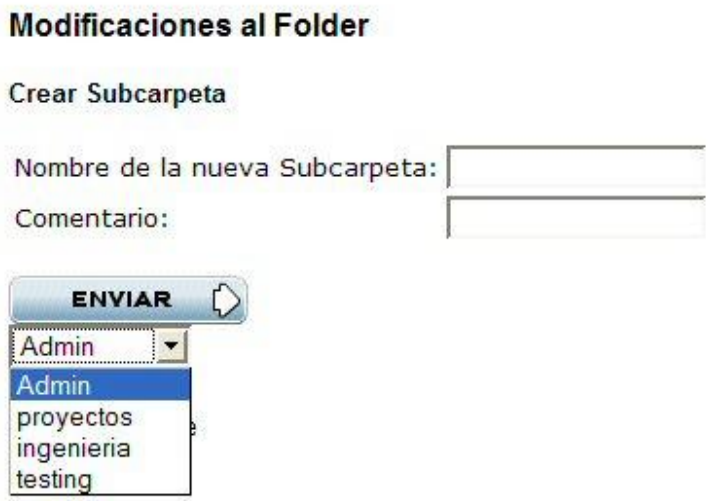

Fig. C4-26 Creación de un subdirectorio sobre el directorio raíz.

4.3.10 Interfaz de usuario del caso de uso Elimina folder

Para eliminar un folder, así como su contenido es necesario contar con permisos de administrador o bien de moderador de grupo(por lo que solo puede alterar el contenido de su grupo). El enlace o liga correspondiente a ésta acción se encuentra al final de la descripción y contenido de los directorios del SAD como es visible en la figura C4-27.

Después de seguir la liga de este botón, el usuario es enviado a una página de confirmación donde el usuario aun tiene la opción de regresar y no eliminar ni el folder ni su contenido. En caso de que el usuario de clic en “eliminar”, el directorio será eliminado irremediablemente de la base de datos, aunque los archivos físicos continúan guardándose en el servidor para atender a cualquier contingencia. En la figura C4-28 se muestra esta interfaz de confirmación para eliminar un directorio.

Fig. C4-27 Ubicación del botón para la eliminación de folders

En la siguiente figura, se muestra la interfaz de confirmación de eliminación de directorios perteneciente al caso de uso elimina folder.

Fig. C4-28 Confirmación para la eliminación de directorios.

4.3.11 Interfaz de usuario del caso de uso Elimina archivo

Para realizar la eliminación de archivos, el usuario debe posicionarse en los detalles del documento que quiere eliminar del sistema. Si se trata de un usuario administrador o moderador de grupo, estará activo el botón de eliminar archivo. Dicho botón lleva al usuario a otra interfaz de confirmación donde se le desplegará una lista de las versiones que serán eliminadas con el archivo en cuestión. La figura C4-29 muestra la ubicación del botón “Eliminar archivo” .

Fig. C4-29 Ubicación del botón “Eliminar archivo” dentro de la interfaz de archivo

La siguiente figura, C4-30, muestra la interfaz de confirmación perteneciente al caso de uso Elimina archivo, la cual presenta los datos principales del archivo así como las versiones que contiene y que serán eliminadas junto con el archivo o documento.

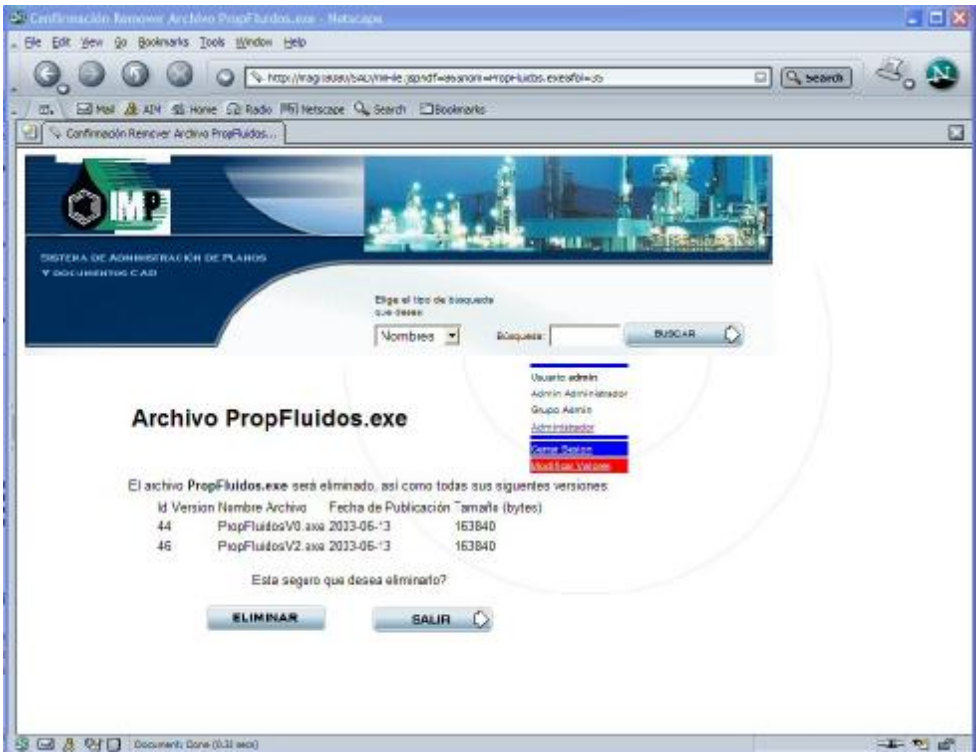

Fig. C4-30 Interfaz de confirmación para eliminar archivo y sus versiones

4.3.12 Interfaz de usuario del caso de uso Bloquear o desbloquear usuarios

La utilidad de bloquear o desbloquear usuarios es exclusiva de los usuarios administradores y sirve para denegar la entrada a usuarios que por alguna razón ya no deben de ingresar al sistema. Esta se encuentra en la interfaz de administración de usuarios y únicamente se tiene que dar un clic en el ícono de usuario activo o usuario bloqueado para cambiar su estatus. Se muestra en la figura C4-31 los íconos de usuarios bloqueados y desbloqueados así como una referencia en la parte superior izquierda para mayor referencia.

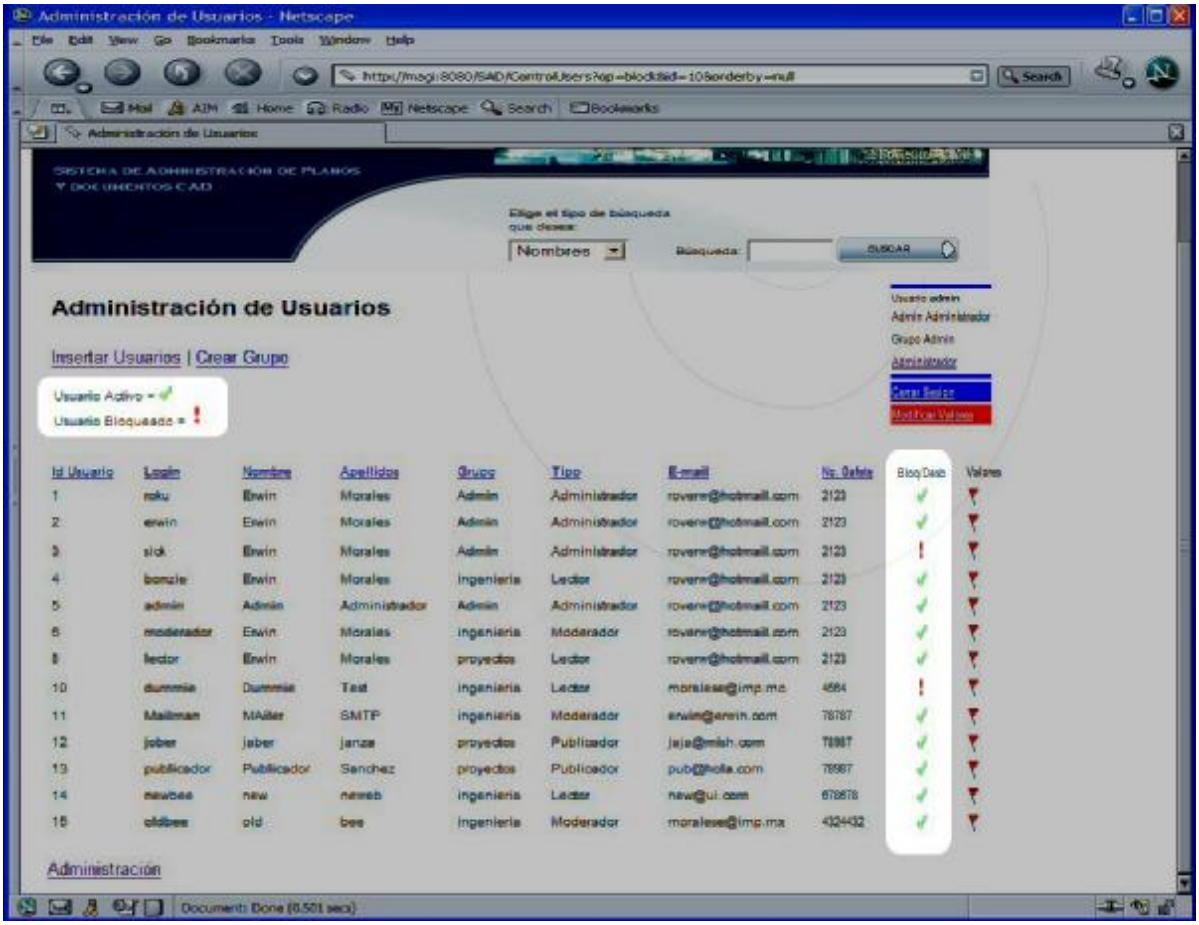

Fig. C4-31 Interfaz de bloqueo/desbloqueo de usuarios.

Dentro de ésta interfaz de administrador de usuarios se encuentran los datos de los usuarios registrados en el sistema: identificador de usuario, nombre de usuario, nombre, apellidos, grupo, tipo, correo electrónico, número de gafete y los accesos a la opción mencionada recientemente(bloqueo de usuarios) y a la modificación de los valores de usuario, con la opción de realizar ordenamientos por los datos mencionados.

4.3.13 Interfaz de usuario del caso de uso crea usuario

El alta de usuarios lo realizan únicamente administradores del sistema, debido a que se pueden manejar varios grupos y roles de usuario. Resultaría complicado de controlar si los usuarios se inscribieran al sistema automáticamente pues al registrarlos es necesario especificar estos datos tan importantes e implicaría doble acción de registro si los usuarios se registraran en un principio y los administradores le asignaran grupo y rol posteriormente.

Aunado a esto, entre los trabajos a futuro que se explicaran posteriormente, se encuentra la autentificación de usuarios usando como referencia el directorio de usuarios del dominio NT manejado en el área de ingeniería del IMP.

En esta interfaz, mostrada en la figura C4-32, los administradores tienen que escribir el nombre de usuario, nombre, apellidos, correo electrónico, número de gafete y tipo de usuario, el password es creado automáticamente por la clase *creaPassword*. Estos datos son validados del lado del servidor, dichas validaciones consisten en que no exista otro usuario con el mismo nombre de usuario y que los datos sean escritos en los campos de texto.

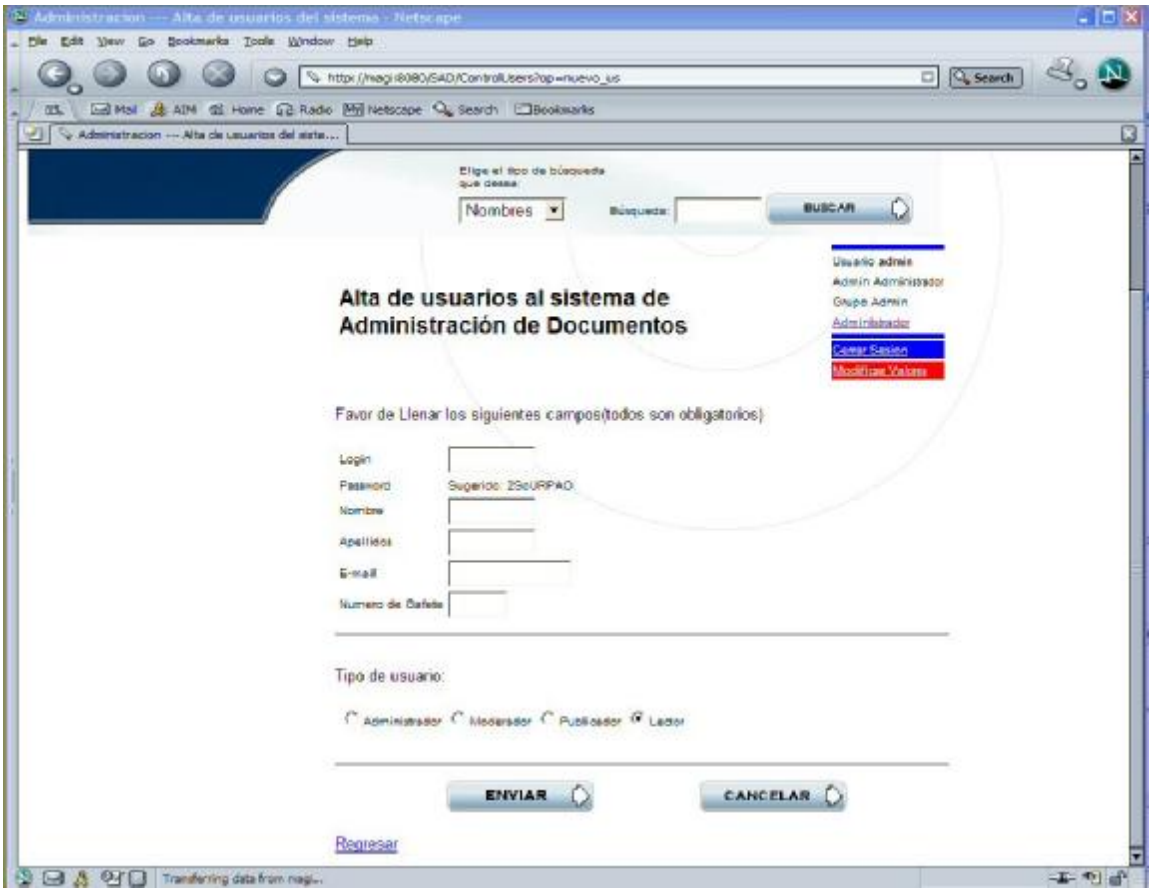

Fig. C4-32 Interfaz de alta de usuarios en el sistema.

Al enviar la forma pulsando en el botón de enviar, se despliega la segunda interfaz de alta de usuarios donde se verifican los datos y se determina el grupo al que pertenecerá el usuario. Esta se muestra en la figura C4-33 y al confirmar dando clic en el botón de enviar el usuario es registrado y notificado por correo electrónico acerca de su nueva cuenta de usuario.

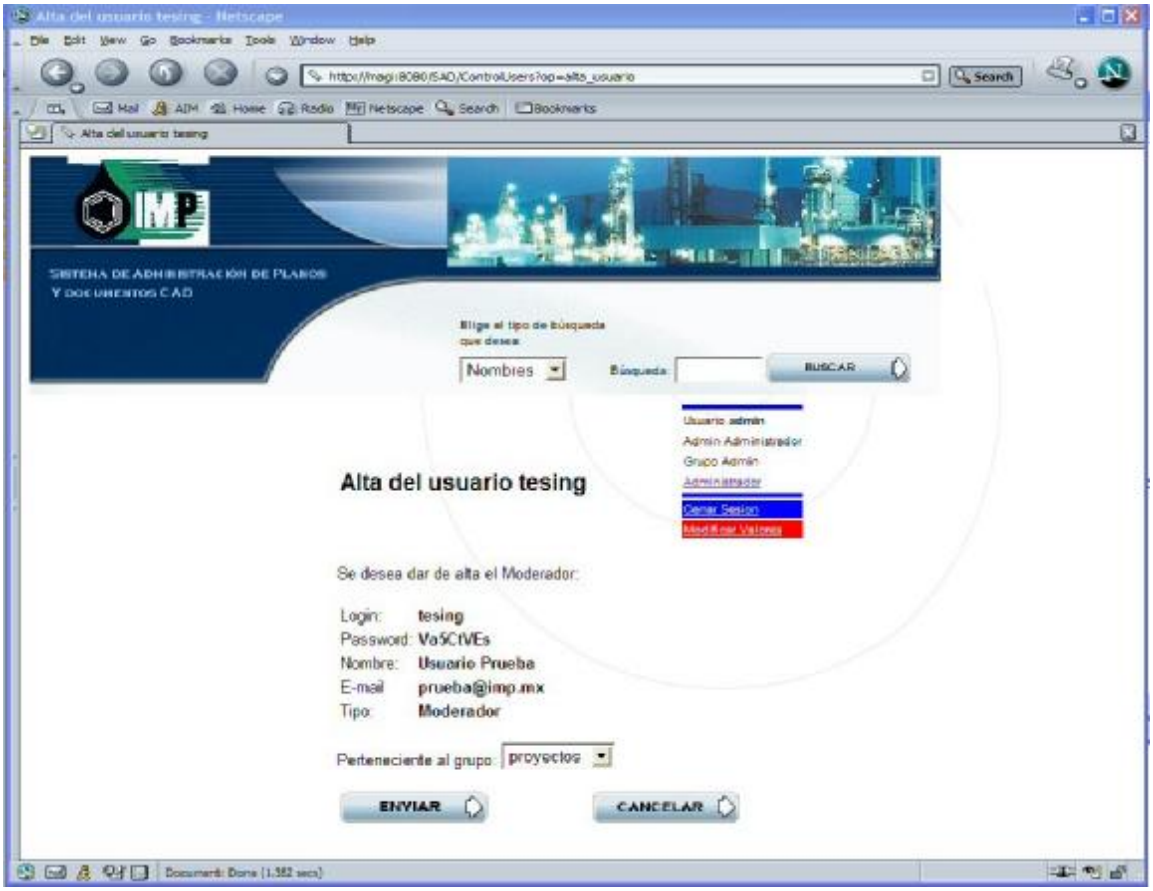

Fig. C4-33 Segunda interfaz de alta de usuarios.

4.3.14 Interfaz de usuario del caso de uso Crear grupos

Continuando con los casos de uso asociados con los usuarios administradores, la creación de nuevos grupos de usuarios se accede desde la pantalla principal de los usuarios administradores o desde la interfaz de administración de usuarios, y consiste en un campo de texto donde se coloca el nombre del grupo a crear únicamente, el sistema por otra parte crea el folder correspondiente a dicho grupo debajo del directorio raíz del sistema.

Dentro de la misma interfaz se encuentra la eliminación de grupos. El usuario tiene que elegir el grupo a eliminar en una caja de selección que contiene los nombres de los grupos y pulsar sobre el botón de eliminar para realizar la modificación.

La figura C4-34 corresponde a la interfaz expuesta donde se realizan las modificaciones de creación y eliminación de grupos.

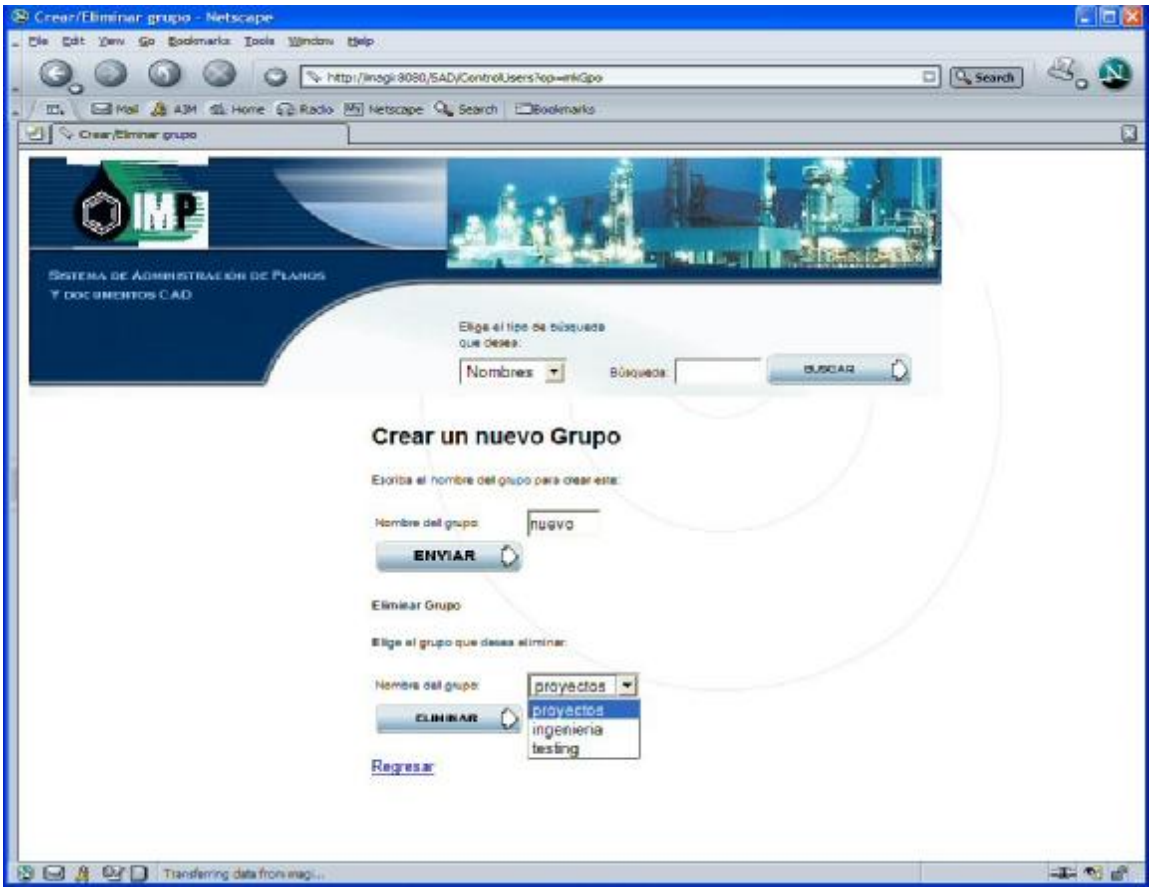

Fig. C4-34 Interfaz de creación y eliminación de grupos

4.3.15 Interfaz de usuario del caso de uso Modifica perfiles de usuarios

Dentro de la interfaz de administración de usuarios, se encuentra un icono que lleva a la correspondiente a la modificación del perfil de usuario con la única restricción de que el usuario a modificar no sea administrador.

Dentro de esta interfaz se pueden modificar el rol de usuario, incrementando sus permisos o bien disminuyéndolos por medio del cambio de valor en los botones de radio desplegados. Así mismo el grupo al que pertenecen, donde se debe de cambiar el valor de la caja de selección al grupo donde se desea reasignar al grupo.

Además de la oportunidad de hacer cambios, se muestran los actuales valores del usuario y su historial de publicaciones. Esta interfaz corresponde a la figura C4-35.

En caso de que el usuario sea un administrador, la interfaz que le desplegara será diferente y prácticamente no se pueden modificar los valores que en caso de los otros tres tipos de usuarios se pueden realizar. La interfaz para usuarios administradores se muestra en la figura C4-36.

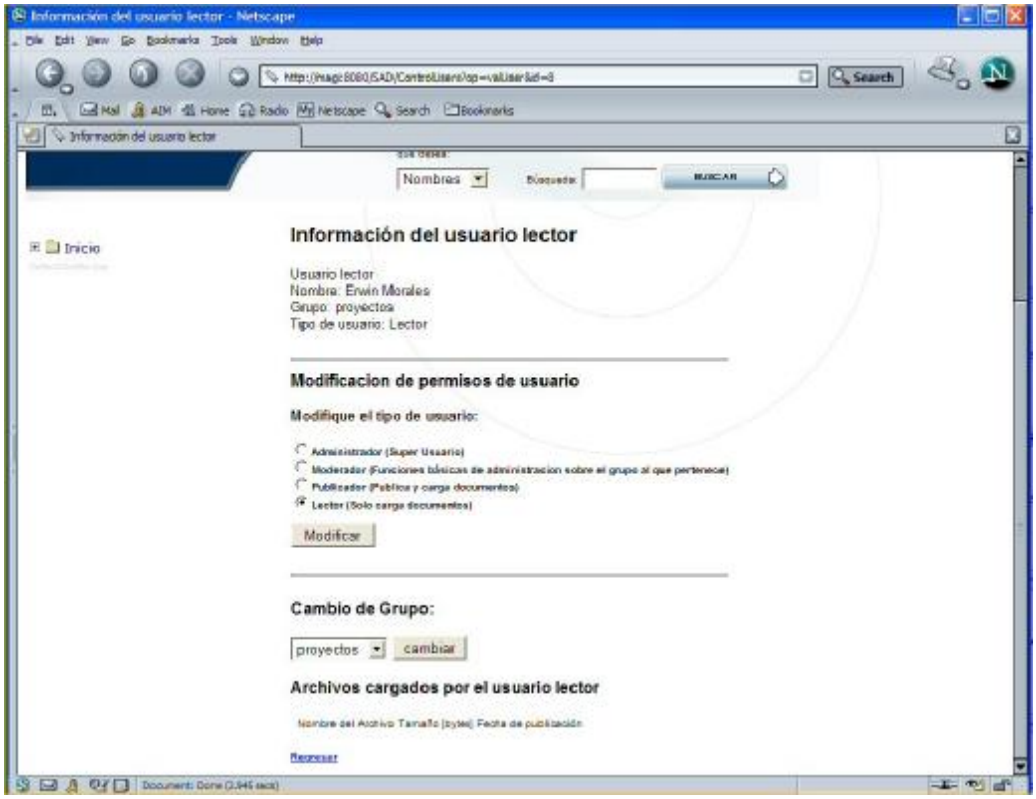

Fig. C4-35 Interfaz de cambio de valores de usuarios.

Dentro de dicha interfaz se puede observar que la mayoría de los datos no pueden ser editados como en el caso del resto de perfiles o roles de usuario.

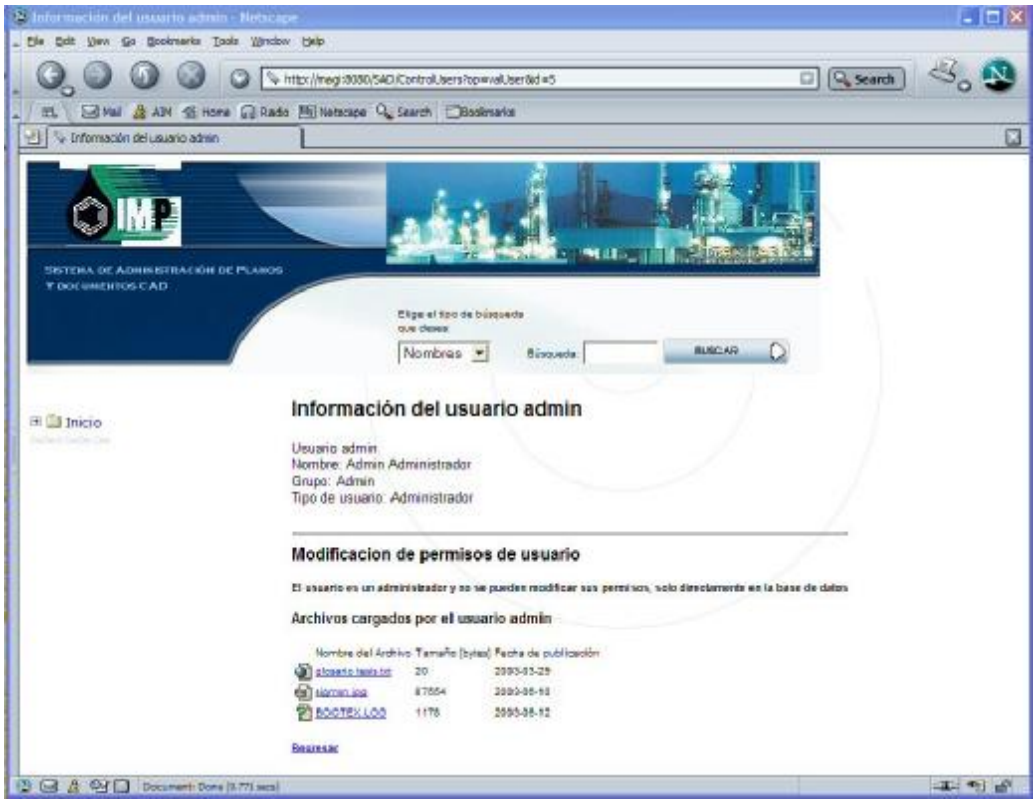

Fig. C4-36 Interfaz desplegada en caso de tratarse de un usuario administrador.

Dentro de este apartado se mostraron las diferentes interfaces gráficas de usuario creadas para la operación de sistema, con lo que se concreta la construcción del sistema de administración de documentos pues al crear las interfaces se integran las tres capas comprendidas en el diseño del sistema bajo el modelo MVC, donde los Servlets creados junto con las clases del sistema, tienen la función de controlar los eventos que se provocan en las interfaces de usuario, eventos que son soportados y ejecutados por los métodos correspondientes a los JavaBeans ya construidos, quienes realizan toda la lógica de negocio con la capa de datos soportada por el servidor de bases de datos relacional.

En el capítulo siguiente, correspondiente a las pruebas realizadas al sistema, se explicara el modelo y tipo de pruebas realizadas al sistema para comprobar su optimo funcionamiento, realizando una revisión de los objetivos planteados en el análisis del sistema con respecto a los puntos logrados en la realización del proyecto.

CAPÍTULO

5

Pruebas Durante el Ciclo de Desar

Dentro de éste capítulo se describen las pruebas realizadas al sistema de administración de documentos, así como la metodología que se empleó para realizar dicha actividad, la cual representa una parte importante dentro del ciclo de desarrollo de la aplicación, pues es con las pruebas como se tiene un conocimiento de la calidad del software y si éste cumple con las especificaciones planteadas al inicio de éste trabajo de tesis.

La prueba del desarrollo es relativamente independiente de la metodología de desarrollo utilizada para construirlo, aunque las pruebas están comprendidas implícitamente dentro de las etapas de desarrollo en el análisis previo realizado en el capítulo dos de éste trabajo de tesis.

Para esto existen varios tipos de pruebas aplicadas durante las diferentes actividades del proceso de desarrollo, las cuales requieren de tiempo y costo extra llegando a significar entre un 30% y 50% del costo total de desarrollo, por lo que dicho modelo requiere ser planificado con anticipación y a la par del desarrollo del sistema. Cabe señalar que las pruebas no son la última actividad realizada en el desarrollo de éste proyecto, no se puede obtener un software de calidad únicamente mediante pruebas finales y depuraciones al sistema en base a éstas. Las pruebas se realizaron en paralelo al desarrollo del sistema, teniendo pruebas finales únicamente como certificación final de la calidad del sistema construido y no como la oportunidad de localizar mas errores, ya que encontrar errores en ésta etapa es bastante problemático pues se tendría que regresar a las etapas anteriores para poder resolverlos de raíz, es considerado una mejor actividad evitar defectos que removerlos.

5.1 Tipos y niveles de pruebas existentes

Los tipos de pruebas existentes se dividen en pruebas de validación y verificación, en el caso de la verificación se revisa si el resultado obtenido corresponde con la especificación del sistema, es decir, si se esta construyendo el sistema correctamente, algo por si sólo no garantiza la satisfacción de los usuarios finales. En el caso de la validación se revisa si el resultado es realmente lo que los usuarios finales desean o dicho de otra manera si el sistema se esta construyendo de manera que tanto la especificación como el resultado sean correctos.

Existen también algunas técnicas de pruebas que se utilizan para realizar la verificación de la funcionalidad de los sistemas, entre estas técnicas se encuentran las siguientes:

Prueba de Regresión, tiene como propósito verificar el sistema luego de haber realizado cambios, como correcciones de faltas.

Prueba de Operación, verifica el sistema en operación por un periodo largo de tiempo bajo condiciones normales de uso. Este tipo de prueba mide la confiabilidad del sistema.

Prueba de escala completa, tiene como propósito verificar el sistema en su carga máxima asignando parámetros a su valor límite, interconectando el sistema con un máximo número de equipos y usuarios simultáneos, un ejemplo de ésta técnica es la prueba de estrés.

Prueba de rendimiento, tiene como propósito medir la capacidad de procesamiento del sistema bajo diferentes cargas, incluyendo espacio de almacenamiento y utilización de procesador.

Prueba de sobrecarga, tiene el propósito de observar como se comporta el sistema cuando se le aplica una sobrecarga y va mas allá de las pruebas de escala completa y rendimiento. Aunque no se puede esperar que el sistema soporte este tipo de pruebas, éste debería ejecutar correctamente, sobrevivir a picos de carga evitando que ocurra una catástrofe.

Prueba negativa, tiene el propósito de medir el estrés del sistema en situaciones inesperadas como en casos de uso que normalmente no serían invocados simultáneamente. El sistema se usa intencionalmente y sistemáticamente de manera incorrecta.

Prueba basada en requisitos o prueba de casos de uso, su propósito es hacer pruebas basadas directamente en la especificación de los requisitos. Pueden utilizarse los mismos casos de uso originales como casos de prueba. También pueden hacerse pruebas para verificar las especificaciones de rendimiento o de escala completa. Se busca verificar que el sistema final cumple con las especificaciones funcionales descritas por los casos de uso originales.

Pruebas ergonómicas, busca probar aspectos ergonómicos del sistema. Se prueba si las interfaces son consistentes con los casos de uso a los cuales corresponden o entre diferentes casos de uso, si los menús son lógicos y legibles, si los mensajes del sistema son visibles, si se puede entender los mensajes de falla, etc.

En cuanto a los niveles de pruebas, existen principalmente tres niveles para la aplicación de las diversas técnicas de pruebas existentes, se clasifican según la profundidad de las pruebas. A continuación se enlistan estos tres tipos así como sus características.

Prueba de Unidad: Se prueba únicamente unidades del sistema, típicamente una clase, paquete o subsistema, es la prueba de mas bajo nivel y en el caso de la programación orientada a objetos el nivel más primitivo en donde se realizan este tipo de pruebas es el nivel de clases.

Prueba de Integración: Se verifica que las unidades trabajen juntas correctamente, pruebas de unidad e integración pueden ser hechas mediante casos de uso, los cuales pueden ser aplicados a clases, paquetes de servicio, subsistemas y el sistema completo.

Prueba de Sistema: Se prueba el sistema completo o la aplicación como tal, se ejecutan varios casos de uso en paralelo y se somete el sistema a diferentes cargas.

5.2 Metodología de pruebas empleada

Debido a que en la realización de las pruebas, estas pueden terminar sin ser realmente cuantificables, es complicado realizar una metodología que resulte apropiada para probar de una manera integral los sistemas. Por esto se aplicó para este proyecto una metodología que fue propuesta en el Objectory Process de Rational debido a que se trata del proceso que realiza las pruebas durante el ciclo de vida del desarrollo del software además de que corresponde a la metodología de desarrollo empleada para éste proyecto.

Esta metodología esta basada en realizar las pruebas en base a los casos de uso propuestos en el análisis del sistema y trata de comprobar el correcto funcionamiento del sistema desde los niveles de clase hasta la comprobación de la correcta interacción de éstas. Además éstas pruebas se realizan bajo un ciclo de vida de pruebas establecida que esta descrito en la siguiente sección.

5.2.1 Ciclo de vida de las pruebas

Al igual que el ciclo de vida del desarrollo del software, las pruebas pueden estar definidas en iteraciones que van reformulando éstas dependiendo de los errores localizados mediante la realización de las pruebas. La figura C5-1 muestra el ciclo de vida de las pruebas, que comprende a las diferentes etapas dicho ciclo de vida, iniciando con la planeación de las pruebas, pasando posteriormente al diseño de la prueba. Después de haber diseñado las pruebas a realizar se procede a implementar las pruebas, pruebas de integración en base a los casos de uso y desde la visión general del sistema para finalmente pasar a la evaluación de la prueba. Este es un proceso iterativo en el que se revisaron continuamente si los objetivos de las pruebas fueron cubiertos en su totalidad y en caso de que ocurriera lo contrario se hubieran tenido que reformular como se muestra en la figura.

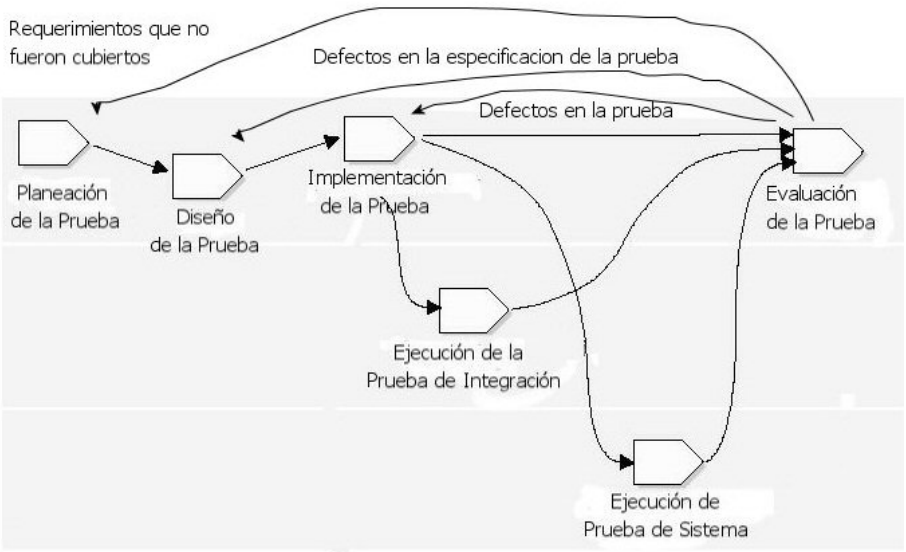

Fig. C5-1 Ciclo de vida de las pruebas.

En base a lo anterior, se definieron una serie de pruebas al sistema, que se describen en la sección siguiente, donde se muestra el resultado de la planeación, diseño y ejecución de las pruebas realizadas en base a los casos de uso, o bien en grupos de casos de uso que se consideran como un módulo que realiza ciertos procesos similares.

5.2.2 Definición y ejecución de pruebas

A. Prueba del caso de uso Autentificar

PLAN DE LA PRUEBA CASO DE USO AUTENTIFICAR	
Introducción:	La realización del proceso de autentificación de un usuario que pertenece al sistema.
Estrategia	Autentificar un usuario tomando los siguientes casos posibles: • El usuario es valido(nombre de usuario y contraseña correctas). • La cuenta de usuario no es valida(nombre de usuario no existente). • Cuenta de usuario valida, contraseña no valida. • Envío de datos nulos, espacios vacíos. Verificando el resultado obtenido en cada caso.
Recursos	Cliente Web(navegador)
Programa de actividades	• Realizar la petición desde el navegador web a la página de inicio del sistema. • Ingresar los valores solicitados(nombre de usuario y contraseña). • Pulsar en el botón de Aceptar. • Verificar resultados.

DISEÑO DE LA PRUEBA CASO DE USO AUTENTIFICAR	
Caso de Prueba	**Procedimiento de prueba**
El usuario es valido(nombre de usuario y contraseña correctas).	Teclear nombre de usuario valido Teclear contraseña correspondiente a dicho usuario. Pulsar sobre el botón de aceptar. Verificar resultado.
La cuenta de usuario no es valida(nombre de usuario no existente).	Teclear nombre de usuario no valido. Teclear cualquier cadena. Pulsar sobre el botón de aceptar. Verificar resultado.
Cuenta de usuario valida, contraseña no valida.	Teclear nombre de usuario valido Teclear contraseña que no correspondiente a dicho usuario. Pulsar sobre el botón de aceptar. Verificar resultado.
Envío de datos nulos, espacios vacíos.	Pulsar sobre el botón de aceptar. Verificar resultado.

EJECUCIÓN DE PRUEBAS CASO DE USO AUTENTIFICAR		
Caso de Prueba	**Error**	**Éxito**
El usuario es valido(nombre de usuario y contraseña correctas).		Éxito
La cuenta de usuario no es valida(nombre de usuario no existente).		Éxito
Cuenta de usuario valida, contraseña no valida.		Éxito
Envío de datos nulos, espacios vacíos.		Éxito

B. Caso de uso Búsqueda

PLAN DE LA PRUEBA CASO DE USO BÚSQUEDA	
Introducción:	Realización de la búsqueda de un documento existente en la base de datos.
Estrategia	Realizar una búsqueda tomando los siguientes casos posibles: • Búsqueda de un archivo existente, tecleando su nombre completo. • Búsqueda de un archivo existente, tecleando una parte de su nombre. • Búsqueda de un archivo que no existe en la base de datos. • Envío de datos nulos para la búsqueda, espacios vacíos. Verificando el resultado obtenido en cada caso.
Recursos	Cliente Web(navegador)
Programa de actividades	• Realizar la petición desde el navegador web en la interfaz de búsqueda incluida en las paginas correspondientes a las descripciones de carpetas y documentos. • Ingresar la cadena de búsqueda. • Pulsar en el botón de “Búsqueda”. • Verificar resultados de búsqueda obtenidos.

DISEÑO DE LA PRUEBA CASO DE USO BÚSQUEDA	
Caso de Prueba	**Procedimiento de prueba**
Búsqueda de un archivo existente, tecleando su nombre completo.	Teclear nombre de documento valido. Pulsar sobre el botón de búsqueda. Verificar resultado deseado, archivo(s) encontrado.
Búsqueda de un archivo existente, tecleando una parte de su nombre.	Teclear porción de nombre de documento valido. Pulsar sobre el botón de búsqueda. Verificar resultado deseado, archivo(s) encontrado.
Búsqueda de un archivo que no existe en la base de datos.	Teclear nombre de documento no valido. Pulsar sobre el botón de búsqueda. Verificar que no haya resultados para esa búsqueda.
Envío de datos nulos para la búsqueda, espacios vacíos.	Pulsar sobre el botón de búsqueda. Verificar que no haya resultados para esa búsqueda.

EJECUCIÓN DE PRUEBAS CASO DE USO BÚSQUEDA		
Caso de Prueba	**Error**	**Éxito**
Búsqueda de un archivo existente, tecleando su nombre completo.		Éxito
Búsqueda de un archivo existente, tecleando una parte de su nombre.		Éxito
Búsqueda de un archivo que no existe en la base de datos.		Éxito
Envío de datos nulos para la búsqueda, espacios vacíos.		Éxito

C. Caso de uso Desplegar Árbol de navegación

PLAN DE LA PRUEBA CASO DE USO DESPLEGAR ÁRBOL DE NAVEGACIÓN	
Introducción:	Acceso a las interfaces web que contienen el árbol jerárquico observando su comportamiento
Estrategia	Despliegue del arbol de navegación con las siguientes actividades: • Expansión de todos los nodos del árbol jerárquico. • Repliegue de todos los nodos del árbol jerárquico. • Expansión de un nodo del árbol. • Repliegue de un nodo del árbol. • Acceder a algún directorio por medio del árbol. • Acceder a algún documento por medio del árbol . • Anexar un nuevo directorio al sistema. • Anexar un nuevo documento al sistema. Verificando el resultado obtenido en cada caso reflejado en el árbol.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta válida de administrador. • Realizar la petición desde el navegador web en las interfaces de descripción de folder y documento. • Extender y replegar el árbol de navegación. • Crear una carpeta y publicar un documento. • Verificar resultados de búsqueda obtenidos.

DISEÑO DE LA PRUEBA CASO DE USO DESPLEGAR ÁRBOL DE NAVEGACIÓN	
Caso de Prueba	**Procedimiento de prueba**
Expansión de todos los nodos del árbol jerárquico.	Pulsar con el puntero del ratón en los símbolos de expansión del arbol repetidamente.
Repliegue de todos los nodos del árbol jerárquico.	Pulsar con el puntero del ratón en los símbolos de repliegue del arbol repetidamente.
Expansión de un nodo del árbol.	Pulsar con el puntero del ratón en los símbolos de expansión del arbol.
Repliegue de un nodo del árbol.	Pulsar con el puntero del ratón en los símbolos de repliegue del arbol.
Acceder a algún directorio por medio del árbol.	Pulsar con el puntero del ratón en algún icono de directorio para acceder a el.
Acceder a algún documento por medio del árbol .	Pulsar con el puntero del ratón en algún icono de documento para acceder a el.

Anexar un nuevo directorio al sistema.	Crear un nuevo directorio en un folder cualquiera para comprobar que se anexa al árbol en el lugar correcto.
Anexar un nuevo documento al sistema.	Publicar un nuevo documento en un directorio cualquiera, verificando que se anexo al árbol.

EJECUCIÓN DE PRUEBAS CASO DE USO DESPLEGAR ÁRBOL DE NAVEGACIÓN		
Caso de Prueba	**Error**	**Éxito**
Expansión de todos los nodos del árbol jerárquico.		Éxito
Repliegue de todos los nodos del árbol jerárquico.		Éxito
Expansión de un nodo del árbol.		Éxito
Repliegue de un nodo del árbol.		Éxito
Acceder a algún directorio por medio del árbol.		Éxito
Acceder a algún documento por medio del árbol .		Éxito
Anexar un nuevo directorio al sistema.		Éxito
Anexar un nuevo documento al sistema.		Éxito

D. Caso de uso Desplegar contenido de Carpeta

PLAN DE LA PRUEBA CASO DE USO DESPLEGAR CONTENIDO DE CARPETA	
Introducción:	Acceso a las interfaces web que muestran el contenido de las carpetas.
Estrategia	Acceso a interfaz de contenido de carpeta, con las siguientes variantes: • Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de administrador. • Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de moderador. • Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de Publicador. • Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de Lector. Verificando que las opciones a realizar correspondan con el tipo de usuario, así como la pertenencia al grupo que pertenece la carpeta.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta válida de administrador, moderador, publicador y lector perteneciente a un grupo de usuario. • Realizar la petición desde el navegador web en las interfaces de descripción de carpeta. • Verificar la pertenencia al grupo correspondiente de la carpeta a la que se desea acceder. • Verificar que las opciones sobre la carpeta correspondan a los permisos del usuario.

DISEÑO DE LA PRUEBA CASO DE USO DESPLEGAR CONTENIDO DE CARPETA	
Caso de Prueba	**Procedimiento de prueba**
Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de administrador.	Ingresar al sistema con cuenta de administrador y verificar el acceso a todas las carpetas, no importando grupo ni posición, además de contar con todas las opciones de modificación a carpetas.
Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de moderador.	Ingresar al sistema con cuenta de moderador de grupo y verificar el acceso a las carpetas que pertenecen al grupo y la denegación a las de grupos ajenos, así como verificar todas las opciones de modificación a carpetas.
Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de Publicador.	Ingresar al sistema con cuenta de publicador y verificar el acceso a las carpetas que pertenecen a su grupo y la denegación de acceso a los grupos ajenos, así como verificar la posibilidad de publicar documentos.
Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de Lector.	Ingresar al sistema con cuenta de lector y verificar el acceso y denegación a las carpetas.

DISEÑO DE LA PRUEBA CASO DE USO DESPLEGAR CONTENIDO DE CARPETA		
Caso de Prueba	**Error**	**Éxito**
Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de administrador.		Éxito
Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de moderador.		Éxito
Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de Publicador.		Éxito
Acceso a carpetas pertenecientes a todos usuarios de los grupos con perfil de Lector.		Éxito

D. Caso de uso Descarga de documentos

PLAN DE LA PRUEBA CASO DE USO DESCARGA DE DOCUMENTOS	
Introducción:	Carga de un documento publicado desde un cliente web.
Estrategia	Cargar un documento existente de cualquier interfaz de descripción de documento. Verificando que el archivo cargado sea el especificado, así como su tipo, y tamaño. Cargar una versión existente y verificar su existencia.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido. • Posicionarse en la descripción de un documento publicado. • Pulsar sobre la liga de “Descargar Documento Original” • Verificar nombre y tamaño de archivo descargado. • Pulsar sobre el nombre de alguna versión disponible. • Verificar la correspondencia entre la información publicada y el archivo obtenido.

EJECUCIÓN DE PRUEBAS CASO DE USO DESCARGA DE DOCUMENTOS		
Caso de Prueba	**Error**	**Éxito**
Cargar un documento existente de cualquier interfaz de descripción de documento.		Éxito

E. Caso de uso Envía Mail

Para probar el funcionamiento de éste caso de uso se consideró como parte del caso de uso "modificar datos de acceso e identificación", dentro de su programa de pruebas.

F. Caso de uso Modificar datos de acceso e identificación.

PLAN DE LA PRUEBA CASO DE USO MODIFICAR DATOS DE ACCESO Y MOD.	
Introducción:	Proceso de modificación de los datos de identificación de un usuario, como nombre, apellidos, correo electrónico, contraseña y número de gafete.
Estrategia	Modificar los datos del usuario con las siguientes variantes: • Nombre del usuario en nulo(vacío). • Apellidos del usuario en nulo(vacío). • Correo electrónico en nulo(vacío). • Correo electrónico con dirección errónea. • Campos de contraseña en nulos. • Contraseña de confirmación diferente a la inicial. • Botón de cancelar.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido. • Ingresar en la liga de "Modificar valores". • Teclear los valores según la estrategia de pruebas. • Pulsar con el puntero del ratón sobre el botón de "Enviar" o "Cancelar según el caso. • Verificar los resultados en el sistema. • Verificar resultados en la bandeja de correo electrónico correspondiente al especificado.

DISEÑO DE LA PRUEBA CASO DE USO MODIFICAR DATOS DE ACCESO Y MOD.	
Caso de Prueba	**Procedimiento de prueba**
Nombre del usuario en nulo(vacío).	Modificar los datos solicitados(nombre, apellido, correo electrónico y ficha) dejando en vacío al correspondiente al nombre. Pulsar sobre Enviar. Verificar que el sistema despliegue un error de falta de nombre.
Apellidos del usuario en nulo(vacío).	Modificar los datos solicitados(nombre, apellido, correo electrónico y ficha) dejando en vacío al correspondiente a los apellidos. Pulsar sobre Enviar.

	Verificar que el sistema despliegue un error de falta de apellidos.
Correo electrónico en nulo(vacío).	Modificar los datos solicitados(nombre, apellido, correo electrónico y ficha) dejando en vacío al correspondiente al correo electrónico. Pulsar sobre Enviar. Verificar que el sistema despliegue un error de falta de correo electrónico.
Correo electrónico con dirección errónea.	Modificar el valor del correo electrónico eliminándole la arroba y el(los) punto(s). Pulsar sobre enviar Verificar que el sistema despliegue un error de sintaxis de correo electrónico.
Campos de contraseña en nulos.	Pulsar sobre enviar Verificar que los cambios no afectaron a la contraseña.
Contraseña de confirmación diferente a la inicial.	Teclear dos contraseñas distintas en cada campo de contraseñas. Verificar que el sistema arroje un error de contraseñas distintas.
Botón de cancelar.	Verificar que se regrese a la interfaz pasada.
Datos correctos.	Ingresar datos correctos en todos los campos Verificar en el sistema si se realizaron los cambios. Verificar en el correo electrónico si el mensaje de confirmación llegó y los datos son correctos.

EJECUCIÓN DE LAS PRUEBAS CASO DE USO MODIFICAR DATOS DE ACCESO Y MOD.		
Caso de Prueba	**Error**	**Éxito**
Nombre del usuario en nulo(vacío).		Éxito
Apellidos del usuario en nulo(vacío).		Éxito
Correo electrónico en nulo(vacío).		Éxito
Correo electrónico con dirección errónea.		Éxito
Campos de contraseña en nulos.		Éxito
Contraseña de confirmación diferente a la inicial.		Éxito
Botón de cancelar.		Éxito
Datos correctos.		Éxito

G. Upload (Carga de documentos al servidor)

PLAN DE LA PRUEBA	
Introducción:	Proceso de carga de documentos al servidor en una carpeta específica.
Estrategia	Emplear la interfaz de carga de archivos con las siguientes variantes: • Pulsar sobre el botón "cargar" con un archivo nulo(campo vacío), con el campo de comentario vacío. • Cargar un archivo, con el campo de comentario en nulo. • Cargar un archivo sin extensión. • Cargar un archivo de tamaño común(~ 100Kb). • Cargar un archivo mediano(~ 1024 Kb). • Cargar un archivo grande(~41502720 Kb). • Cargar un archivo con nombre grande(>250 caracteres). • Cargar un archivo con el mismo nombre de alguno que ya se encuentre publicado(nueva versión). • Cargar un archivo al servidor con los cuatro distintos perfiles de usuario, uno a la vez. • Cargar un archivo en un grupo al que no pertenece el usuario con los perfiles Moderador, publicador y lector.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido con perfil de administrador, moderador o publicador. • Ingresar a cualquier folder accesible para el usuario. • Teclear los valores y elegir documentos según la estrategia de pruebas. • Pulsar con el puntero del ratón sobre el botón de "Cargar" • Verificar los resultados en el sistema

DISEÑO DE LA PRUEBA	
Caso de Prueba	**Procedimiento de prueba**
Pulsar sobre el botón "cargar" con un archivo nulo(campo vacío), con el campo de comentario vacío.	Pulsar con el puntero del ratón el botón de cargar sin haber elegido algún archivo ni escrito ningún comentario. Verificar que no ocurran errores ni se publique un documento nulo.
Cargar un archivo, con el campo de comentario en nulo.	Elegir un archivo a cargar en la ventana de navegación que se despliega al presionar "browse" Verificar que el archivo se publique y descargue sin ningún problema.
Cargar un archivo sin extensión.	Elegir un archivo a cargar en la ventana de navegación que se despliega al presionar "browse" que cumpla con el requisito de no tener extensión. Verificar que el archivo se publique y descargue sin ningún problema.
Cargar un archivo de tamaño común(~ 100Kb)	Elegir un archivo a cargar en la ventana de navegación que se despliega al presionar "browse" que cumpla con el requisito de tener aproximadamente 100 Kb de tamaño.

	Verificar que el archivo se publique y descargue sin ningún problema.
Cargar un archivo mediano(~ 1024 Kb)	Elegir un archivo a cargar en la ventana de navegación que se despliega al presionar "browse" que cumpla con el requisito de tener aproximadamente 1 Mb de tamaño. Verificar que el archivo se publique y descargue sin ningún problema.
Cargar un archivo grande(~41502720 Kb)	Elegir un archivo a cargar en la ventana de navegación que se despliega al presionar "browse" que cumpla con el requisito de tener aproximadamente 40 Mb de tamaño. Verificar que el archivo se publique y descargue sin ningún problema.
Cargar un archivo con nombre grande(>250 caracteres)	Elegir un archivo a cargar en la ventana de navegación que se despliega al presionar "browse" que cumpla con el requisito de tener aproximadamente 250 caracteres en su nombre. Verificar que el archivo se publique y descargue sin ningún problema.
Cargar un archivo con el mismo nombre de alguno que ya se encuentre publicado(nueva versión)	Elegir un archivo a cargar en la ventana de navegación que se despliega al presionar "browse" que cumpla con el requisito de tener el mismo nombre de algún archivo publicado en el sistema. Verificar que el archivo se encuentra en las versiones correspondientes a su mismo nombre y se descargue sin ningún problema..
Cargar un archivo al servidor con los cuatro distintos perfiles de usuario, uno a la vez.	Ingresar al sistema con una cuenta distinta a la vez, administrador, moderador, publicador y lector. Ingresar a una carpeta que pertenezca al grupo del usuario autentificado. Elegir un archivo a cargar en la ventana de navegación que se despliega al presionar "browse". Verificar que el archivo se publique en el sistema, con excepción del usuario Lector, al que no le debe de aparecer la opción de "Cargar Documento".
Cargar un archivo en un grupo al que no pertenece el usuario con los perfiles Moderador, publicador y lector.	Ingresar al sistema con una cuenta distinta a la vez, moderador, publicador y lector. Ingresar a una carpeta que no pertenezca al grupo del usuario autentificado. Verificar que el usuario no tenga acceso a esa carpeta.

DISEÑO DE LA PRUEBA		
Caso de Prueba	**Error**	**Éxito**
Pulsar sobre el botón "cargar" con un archivo nulo(campo vacío), con el campo de comentario vacío.		Éxito
Cargar un archivo, con el campo de comentario en nulo.		Éxito
Cargar un archivo sin extensión.		Éxito

Cargar un archivo de tamaño común(~ 100Kb).		Éxito
Cargar un archivo mediano(~ 1024 Kb).		Éxito
Cargar un archivo grande(~41502720 Kb).		Éxito
Cargar un archivo con nombre grande(>250 caracteres).		Éxito
Cargar un archivo con el mismo nombre de alguno que ya se encuentre publicado(nueva versión).		Éxito
Cargar un archivo al servidor con los cuatro distintos perfiles de usuario, uno a la vez.		Éxito
Cargar un archivo en un grupo al que no pertenece el usuario con los perfiles Moderador, publicador y lector.		Éxito

H. Caso de uso Cambio nombre de folder

PLAN DE LA PRUEBA CASO DE USO CAMBIO DE NOMBRE DE FOLDER	
Introducción:	Proceso de modificación de nombre de una carpeta contenida en el sistema.
Estrategia	Realizar el proceso de modificación de cambio de nombre de carpeta con las siguientes variantes: • Nuevo nombre de la carpeta en nulo(vacío). • Nuevo nombre de carpeta válido y comentario en nulo(vacío). • Nuevo nombre de la carpeta válido y comentario. Realizar el proceso de modificación de cambio de nombre de carpeta con usuarios de tipo publicador y lector.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido con perfil de administrador o moderador(excepto última prueba) • Ingresar a cualquier carpeta accesible al usuario. • Teclear los valores según la estrategia de pruebas. • Pulsar con el puntero del ratón sobre el botón de “Enviar”. • Verificar los resultados en el sistema.

DISEÑO DE LA PRUEBA CASO DE USO CAMBIO DE NOMBRE DE FOLDER	
Caso de Prueba	**Procedimiento de prueba**
Nuevo nombre de la carpeta en nulo(vacío).	Pulsar sobre el botón “Enviar” de la interfaz de cambio de nombre de carpeta sin escribir nuevo nombre de carpeta y comentario indistinto. Verificar resultados.
Nuevo nombre de carpeta válido y comentario en nulo(vacío).	Escribir un nombre de carpeta nuevo Pulsar sobre el botón “Enviar” de la interfaz de cambio de nombre de carpeta sin escribir comentario. Verificar resultados.
Nuevo nombre de la carpeta válido y comentario.	Escribir nombre de carpeta válido y comentario. Pulsar sobre el botón “Enviar” de la interfaz de cambio de nombre de carpeta. Verificar resultados.
Realizar el proceso de modificación de cambio de nombre de carpeta con	Ingresar al sistema con un usuario con perfil de Publicador o Lector.

usuarios de tipo publicador y lector.	Posicionarse en una carpeta válida para el usuario. Verificar que la opción de modificar nombre de folder no se encuentra en sus opciones.

EJECUCIÓN DE PRUEBAS CASO DE USO CAMBIO DE NOMBRE DE FOLDER		
Caso de Prueba	**Error**	**Éxito**
Nuevo nombre de la carpeta en nulo(vacío).		Éxito
Nuevo nombre de carpeta válido y comentario en nulo(vacío).		Éxito
Nuevo nombre de la carpeta válido y comentario.		Éxito
Realizar el proceso de modificación de cambio de nombre de carpeta con usuarios de tipo publicador y lector.		Éxito

I. Caso de uso Crear nuevo Folder

PLAN DE LA PRUEBA CASO DE USO CREAR NUEVO FOLDER	
Introducción:	Proceso de creación de una subcarpeta sobre cualquier carpeta existente y válida para el usuario.
Estrategia	Realizar el proceso de creación de una nueva carpeta con las siguientes variantes: • Nombre de la carpeta en nulo(vacío) y comentario indistinto(vacío o con contenido). • Nombre de carpeta nueva válido y comentario en nulo(vacío). • Nuevo nombre de la carpeta válido y comentario valido. Realizar el proceso de modificación de crear nuevo folder con usuarios con perfil de publicador y lector.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido de tipo administrador o moderador. • Ingresar a cualquier carpeta accesible al usuario. • Teclear los valores según la estrategia de pruebas en la interfaz de creación de nuevas carpetas. • Pulsar con el puntero del ratón sobre el botón de “Enviar”. • Verificar los resultados en el sistema.

DISEÑO DE LA PRUEBA CASO DE USO CREAR NUEVO FOLDER	
Caso de Prueba	**Procedimiento de prueba**
Nombre de la carpeta en nulo(vacío) y comentario indistinto(vacío o con contenido).	Pulsar sobre el botón “Enviar” de la interfaz de creación de carpeta sin escribir nombre de carpeta nueva y comentario indistinto. Verificar resultados.
Nombre de carpeta nueva válido y comentario en nulo(vacío).	Escribir un nombre de carpeta nueva Pulsar sobre el botón “Enviar” de la interfaz de cambio de nombre de carpeta sin escribir comentario. Verificar resultados.
Nuevo nombre de la carpeta válido y comentario valido.	Escribir nombre de carpeta válido y comentario. Pulsar sobre el botón “Enviar” de la interfaz de creación de carpeta.

	Verificar resultados
Realizar el proceso de modificación de crear nuevo folder con usuarios con perfil de publicador y lector.	Ingresar al sistema con un usuario con perfil de Publicador o Lector. Posicionarse en una carpeta válida para el usuario. Verificar que la opción de creación de un nuevo folder no se encuentra en sus opciones.

EJECUCIÓN DE PRUEBAS CASO DE USO CREAR NUEVO FOLDER		
Caso de Prueba	**Error**	**Éxito**
Nombre de la carpeta en nulo(vacío) y comentario indistinto(vacío o con contenido).		Éxito
Nombre de carpeta nueva válido y comentario en nulo(vacío).		Éxito
Nuevo nombre de la carpeta válido y comentario valido.		Éxito
Realizar el proceso de modificación de crear nuevo folder con usuarios con perfil de publicador y lector.		Éxito

J. Caso de uso Elimina Folder

PLAN DE LA PRUEBA CASO DE USO ELIMINA FOLDER	
Introducción:	Proceso de eliminación de una carpeta válida para el usuario.
Estrategia	Realizar el proceso de eliminación de una carpeta con usuarios con perfil publicador y lector. Realizar el proceso de eliminación de carpeta con usuarios con perfil publicador y lector. Realizar el proceso de eliminación de carpeta con usuarios con perfil moderador, publicador y lector en una carpeta que no pertenezca a su grupo.
Recursos	Cliente Web(navegador).
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido con el perfil indicado. • Ingresar a cualquier carpeta accesible al usuario, en su caso. • Pulsar con el puntero del ratón sobre el botón de “Eliminar Folder”. • Confirmar la eliminación en la pantalla resultante. • Verificar los resultados en el sistema, verificar que no exista dicha carpeta ni sus hijos.

DISEÑO DE LA PRUEBA CASO DE USO ELIMINA FOLDER	
Caso de Prueba	**Procedimiento de prueba**
Realizar el proceso de eliminación de una carpeta.	Pulsar sobre el botón “Eliminar Folder” de la interfaz de descripción de carpeta y confirmar la petición en la siguiente pantalla.
Realizar el proceso de eliminación de carpeta con usuarios con perfil publicador y lector.	Verificar que la opción de eliminar folder no se encuentra dentro de sus opciones de carpeta
Realizar el proceso de eliminación de carpeta con usuarios con perfil moderador, publicador y lector en una carpeta que no pertenezca a su grupo.	Verificar que la carpeta no es accesible para el usuario.

DISEÑO DE LA PRUEBA CASO DE USO ELIMINA FOLDER		
Caso de Prueba	**Error**	**Éxito**
Realizar el proceso de eliminación de una carpeta.		Éxito
Realizar el proceso de eliminación de carpeta con usuarios con perfil publicador y lector.		Éxito
Realizar el proceso de eliminación de carpeta con usuarios con perfil moderador, publicador y lector en una carpeta que no pertenezca a su grupo.		Éxito

K. Caso de uso Elimina Archivo

PLAN DE LA PRUEBA CASO DE USO ELIMINA ARCHIVO	
Introducción:	Proceso de eliminación de un documento válido para el usuario.
Estrategia	Realizar el proceso de eliminación de un documento con usuarios con perfiles siguientes: • Administrador y moderador de grupo • Publicador y Lector Realizar el proceso de eliminación de un documento sobre una carpeta que no pertenece al grupo un usuario con perfil moderador, publicador y lector.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido del perfil indicado. • Ingresar a cualquier carpeta accesible al usuario(si es el caso). • Ingresar a cualquier documento perteneciente a dicha carpeta. • Pulsar con el puntero del ratón sobre el botón de “Eliminar Archivo”. • Confirmar la eliminación en la pantalla resultante. • Verificar los resultados en el sistema, verificar que no exista dicho documento ni sus versiones.

DISEÑO DE LA PRUEBA CASO DE USO ELIMINA ARCHIVO	
Caso de Prueba	**Procedimiento de prueba**
Realizar el proceso de eliminación de un documento, con perfil Administrador y moderador.	Pulsar sobre el botón “Eliminar Archivo” de la interfaz de descripción de documento y confirmar la petición en la siguiente pantalla.
Realizar el proceso de eliminación de un documento, con perfil Administrador y moderador.	Verificar que la operación no se puede realizar, ya que no existe la opción de eliminar archivo.
Realizar el proceso de eliminación de un documento sobre una carpeta que no pertenece al grupo del usuario.	Verificar que la(s) carpeta(s) no son accesibles para el usuario.

EJECUCIÓN DE PRUEBAS CASO DE USO ELIMINA ARCHIVO		
Caso de Prueba	**Error**	**Éxito**
Realizar el proceso de eliminación de un documento, con perfil Administrador y moderador.		Éxito
Realizar el proceso de eliminación de un documento, con perfil Administrador y moderador.		Éxito
Realizar el proceso de eliminación de un documento sobre una carpeta que no pertenece al grupo del usuario.		Éxito

L. Caso de Uso elimina versión

PLAN DE LA PRUEBA CASO DE USO ELIMINA VERSIÓN	
Introducción:	Proceso de eliminación de una versión de un documento válido para el usuario.
Estrategia	Realizar el proceso de eliminación de una versión de un documento con usuario con perfil de Administrador o moderador. Realizar el proceso de eliminación de una versión de un documento con usuario con perfil de Publicador o Lector.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido del perfil indicado. • Ingresar a cualquier carpeta accesible al usuario. • Ingresar a cualquier documento perteneciente a dicha carpeta. • Pulsar con el puntero del ratón sobre la liga de “Eliminar Versión”. • Verificar los resultados en el sistema, verificar que no exista dicha versión en la descripción del documento.

DISEÑO DE LA PRUEBA CASO DE USO ELIMINA VERSIÓN	
Caso de Prueba	**Procedimiento de prueba**
Realizar el proceso de eliminación de una versión un documento con usuario con perfil de Administrador o moderador.	Pulsar sobre la liga de “Eliminar Version” de la interfaz de descripción de documento. Verificar resultados.
Realizar el proceso de eliminación de una versión de un documento con usuario con perfil de Publicador o Lector.	Verificar que la operación es no valida para dicho usuario pues no se encuentra entre sus opciones.

EJECUCIÓN DE PRUEBAS CASO DE USO ELIMINA VERSIÓN		
Caso de Prueba	**Error**	**Éxito**
Realizar el proceso de eliminación de una versión un documento con usuario con perfil de Administrador o moderador.		Éxito
Realizar el proceso de eliminación de una versión de un documento con usuario con perfil de Publicador o Lector.		Éxito

M. Bloquear o desbloquear usuarios.

PLAN DE LA PRUEBA CASO DE USO BLOQUEAR O DESBLOQUEAR USUARIOS	
Introducción:	Proceso bloqueo o desbloqueo de un usuario en el sistema
Estrategia	Realizar el proceso conforme: • Bloquear usuario • Desbloquear usuario. Bajo una cuenta de perfil Administrador de sistema. Realizar el proceso conforme: • Bloquear usuario • Desbloquear usuario. Bajo una cuenta de perfil diferente al de administrador.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido del perfil solicitado. • Ingresar a la administración de usuarios. • Pulsar en el icono de bloqueo / desbloqueo del usuario que se desea modificar(ó). • Verificar en la administración de usuarios la modificación. • Prueba de autentificación del usuario que se modifico su acceso al sistema. • Verificar que los usuarios bloqueados no puedan ingresar al sistema y que los que se encuentran en estado de disponible no tengan ningún problema para ingresar.

DISEÑO DE LA PRUEBA CASO DE USO BLOQUEAR O DESBLOQUEAR USUARIOS	
Caso de Prueba	**Procedimiento de prueba**
Bloquear Usuario, perfil administrador.	Pulsar sobre el icono que indica usuario disponible para cambiarlo de estado. Verificar en el sistema si se realizó el cambio. Verificar en la autentificación de usuarios que no tiene acceso al sistema.
Desbloquear usuario, perfil administrador.	Pulsar sobre el icono que indica usuario no disponible para cambiarlo de estado. Verificar en el sistema si se realizó el cambio. Verificar en la autentificación de usuarios que no tiene acceso al sistema.
Bloquear Usuario, perfil distinto al de administrador.	Verificar que la opción no es posible de realizar pues no es accesible para dicho usuario.
Desbloquear usuario, perfil distinto al de administrador.	Verificar que la opción no es posible de realizar pues no es accesible para dicho usuario.

EJECUCIÓN DE PRUEBAS CASO DE USO BLOQUEAR O DESBLOQUEAR USUARIOS		
Caso de Prueba	**Error**	**Éxito**
Bloquear Usuario, perfil administrador.		Éxito
Desbloquear usuario, perfil administrador.		Éxito
Bloquear Usuario, perfil distinto al de administrador.		Éxito
Desbloquear usuario, perfil distinto al de administrador.		Éxito

N. Caso de uso Crear Usuario

PLAN DE LA PRUEBA CASO DE USO CREAR USUARIO	
Introducción:	Proceso de creación de un nuevo usuario del sistema.
Estrategia	Emplear la interfaz de creación de usuarios con las siguientes variantes: • Intentar crear un usuario con todos los campos que se piden vacíos. • Intentar crear un usuario con el campo de nombre de usuario vacío. • Intentar crear un usuario con el campo de nombre vacío. • Intentar crear un usuario con el campo de apellidos vacío. • Intentar crear un usuario con el campo de correo electrónico vacío. • Intentar crear un usuario con el campo de correo electrónico erróneo. • Intentar crear un usuario con el campo de número de gafete vacío. • Intentar crear un usuario con nombre de usuario existente en el sistema. • Intentar crear un usuario con todos los datos válidos. Intentar ingresar a la interfaz de creación de usuarios con un usuario de tipo moderador, publicador y lector.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido de tipo administrador si es el caso. • Ingresar al módulo de administración de usuarios. • Ingresar a la liga de “Insertar Usuarios” • Capturar datos solicitados según las variantes de la estrategia. • Verificar los resultados en el sistema. • Verificar en la bandeja de correo electrónico las confirmaciones enviadas por el sistema.

DISEÑO DE LA PRUEBA CASO DE USO CREAR USUARIO	
Caso de Prueba	**Procedimiento de prueba**
Intentar crear un usuario con todos los campos que se piden vacíos.	Pulsar sobre el botón de enviar de la primera pantalla de creación de usuarios sin llenar los campos pedidos. Verificar que arroje errores por falta de campos.
Intentar crear un usuario con el campo de nombre de usuario vacío.	Llenar con datos válidos todos los campos pedidos en la interfaz de creación de nuevos usuarios con excepción del nombre de usuario. Pulsar sobre la tecla de “Enviar”.

	Verificar que arroje un error de falta de nombre de usuario.
Intentar crear un usuario con el campo de nombre vacío.	Llenar con datos válidos todos los campos pedidos en la interfaz de creación de nuevos usuarios con excepción del nombre del usuario. Pulsar sobre la tecla de “Enviar”. Verificar que arroje un error de falta de nombre del usuario.
Intentar crear un usuario con el campo de apellidos vacío.	Llenar con datos válidos todos los campos pedidos en la interfaz de creación de nuevos usuarios con excepción del campo de apellidos. Pulsar sobre la tecla de “Enviar”. Verificar que arroje un error de falta de apellidos.
Intentar crear un usuario con el campo de correo electrónico vacío.	Llenar con datos válidos todos los campos pedidos en la interfaz de creación de nuevos usuarios con excepción del correo electrónico Pulsar sobre la tecla de “Enviar”. Verificar que arroje un error de falta de correo electrónico.
Intentar crear un usuario con el campo de correo electrónico erróneo.	Llenar con datos válidos todos los campos pedidos en la interfaz de creación de nuevo usuario con un formato de correo electrónico incorrecto(sin “arroba”, sin punto). Pulsar sobre la tecla de “Enviar”. Verificar que arroje un error de .
Intentar crear un usuario con el campo de número de gafete vacío.	Llenar con datos válidos todos los campos pedidos en la interfaz de creación de nuevos usuarios con excepción del número de gafete. Pulsar sobre la tecla de “Enviar”. Verificar que arroje un error de falta de numero de gafete.
Intentar crear un usuario con nombre de usuario existente en el sistema.	Llenar con datos válidos todos los campos pedidos en la interfaz de creación de nuevos usuarios con un nombre de usuario que se sabe que existe en el sistema(p.e. “admin”, usuario por default del sistema). Pulsar sobre la tecla de “Enviar”. Verificar que arroje un error de falta de duplicidad de nombres de usuario.
Intentar crear un usuario con todos los datos válidos.	Llenar con datos válidos los campos pedidos en la interfaz de creación de nuevos usuarios con datos válidos y nombre de usuario que no exista dentro de la base de datos del SAD. Pulsar la tecla “Enviar”. Elegir grupo en la nueva interfaz obtenida. Verificar en el sistema la correcta autentificación del usuario creado. Verificar en la bandeja de entrada del correo electrónico especificado en el alta del usuario, si los datos de confirmación de cuenta son recibidos y correctos.
Intentar ingresar a la interfaz de	Verificar que para el usuario no es posible ingresar a

creación de usuarios con un usuario de tipo moderador, publicador y lector.	dicha interfaz pues no tiene los permisos debidos.

EJECUCIÓN DE PRUEBAS CASO DE USO CREAR USUARIO		
Caso de Prueba	**Error**	**Éxito**
Intentar crear un usuario con todos los campos que se piden vacíos.		Éxito
Intentar crear un usuario con el campo de nombre de usuario vacío.		Éxito
Intentar crear un usuario con el campo de nombre vacío.		Éxito
Intentar crear un usuario con el campo de apellidos vacío.		Éxito
Intentar crear un usuario con el campo de correo electrónico vacío.		Éxito
Intentar crear un usuario con el campo de correo electrónico erróneo.		Éxito
Intentar crear un usuario con el campo de número de gafete vacío.		Éxito
Intentar crear un usuario con nombre de usuario existente en el sistema.		Éxito
Intentar crear un usuario con todos los datos válidos.		Éxito
Intentar ingresar a la interfaz de creación de usuarios con un usuario de tipo moderador, publicador y lector.		Éxito

O. Caso de uso Crear y Eliminar Grupos

PLAN DE LA PRUEBA CASO DE USO CREAR Y ELIMINAR GRUPOS	
Introducción:	Proceso de creación y eliminación de grupos de usuarios.
Estrategia	• Realizar el proceso de creación de un grupo de usuarios colocando valor nulo en el nombre del nuevo grupo. • Realizar el proceso de creación de un grupo de usuarios con el nombre de un grupo existente. • Realizar el proceso de creación de un grupo de usuarios colocando un nombre valido y no existente en la base de datos. • Realizar el proceso de eliminación de un grupo de usuarios. • Realizar el proceso de eliminación de un grupo bajo una cuenta con perfil de moderador, publicador y lector.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido de tipo administrador, si es el caso. • Ingresar al módulo de administración de usuarios. • Ingresar a la liga de “Crear Grupo”. • Escribir o seleccionar el nombre del grupo que se desea crear o eliminar respectivamente. • Pulsar en el botón de “Enviar” o “Eliminar” según sea el caso. • Verificar los resultados en el sistema.

DISEÑO DE LA PRUEBA CASO DE USO CREAR Y ELIMINAR GRUPOS	
Caso de Prueba	**Procedimiento de prueba**
Realizar el proceso de creación de un grupo de usuarios colocando valor nulo en el nombre del nuevo grupo.	Pulsar el botón de enviar en la interfaz de creación de nuevos grupos sin escribir ningún valor en el campo de nombre de grupo. Verificar que no se cree un grupo nuevo con valor nulo.
Realizar el proceso de creación de un grupo de usuarios con el nombre de	Pulsar el botón de enviar en la interfaz de creación de nuevos grupos escribiendo el nombre de un grupo

un grupo existente.	de usuarios existente en el SAD. Verificar que el sistema arroje un error de grupo existente.
Realizar el proceso de creación de un grupo de usuarios colocando un nombre valido y no existente en la base de datos.	Pulsar el botón de enviar en la interfaz de creación de nuevos grupos escribiendo el nombre de un grupo de usuarios valido(no existente) en el SAD. Verificar que el sistema adjunte el nuevo grupo al sistema. Crear un usuario que pertenezca al grupo e ingresar con dicho usuario al folder raiz del grupo.
Realizar el proceso de eliminación de un grupo de usuarios.	Elegir un grupo entre la lista y pulsar sobre el botón "Eliminar". Verificar que el sistema haya eliminado el grupo de la lista.
Realizar el proceso de eliminación de un grupo bajo una cuenta con perfil de moderador, publicador y lector.	Verificar que ésta opción no aparezca en sus opciones y no sea posible de realizar por los permisos con que cuenta el usuario.

EJECUCIÓN DE PRUEBAS CASO DE USO CREAR Y ELIMINAR GRUPOS		
Caso de Prueba	**Error**	**Éxito**
Realizar el proceso de creación de un grupo de usuarios colocando valor nulo en el nombre del nuevo grupo.		Éxito
Realizar el proceso de creación de un grupo de usuarios con el nombre de un grupo existente.		Éxito
Realizar el proceso de creación de un grupo de usuarios colocando un nombre valido y no existente en la base de datos.		Éxito
Realizar el proceso de eliminación de un grupo de usuarios.		Éxito
Realizar el proceso de eliminación de un grupo bajo una cuenta con perfil de moderador, publicador y lector.		Éxito

P. Caso de uso Modifica perfiles de usuarios

PLAN DE LA PRUEBA CASO DE USO MODIFICA PERFILES DE USUARIO	
Introducción:	Proceso de modificación de perfiles de usuarios.
Estrategia	Realizar el proceso de modificación de perfil de usuario con dos tipos de usuario diferentes, y en las dos diferentes actividades, ingresando al sistema con un usuario de perfil administrador. • Eligiendo a un usuario administrador y tratando de modificar su tipo de usuario. • Eligiendo a un usuario administrador y tratando de modificar su grupo de pertenencia. • Eligiendo a un usuario de diferente tipo que administrador y modificar su tipo de usuario. • Eligiendo a un usuario de diferente tipo que administrador y modificar su grupo de pertenencia. Realizar el proceso de modificación de perfil de usuario a cualquier usuario,

	ingresando al sistema con un usuario de perfil moderador, publicador y lector.
Recursos	Cliente Web(navegador)
Programa de actividades	• Ingresar al sistema con una cuenta de usuario válido del perfil que se indique. • Ingresar al módulo de administración de usuarios. • Dar clic sobre el icono correspondiente a modificar valores( ). • Cambiar el tipo de usuario por medio de los botones de radio o bien el grupo de pertenencia de la lista, según sea el caso de la prueba. • Pulsar en el botón de "Modificar" o "Cambiar" según sea el caso. • Verificar los resultados en el sistema.

DISEÑO DE LA PRUEBA CASO DE USO MODIFICA PERFILES DE USUARIO	
Caso de Prueba	**Procedimiento de prueba**
Eligiendo a un usuario administrador y tratando de modificar su tipo de usuario.	Desde el módulo de administración de usuarios, elegir a un usuario administrador y pulsar sobre el icono de modificar valores( ). Verificar que ningún valor de tipo de usuario puede ser modificado en éste caso.
Eligiendo a un usuario administrador y tratando de modificar su grupo de pertenencia.	Desde el módulo de administración de usuarios, elegir a un usuario administrador y pulsar sobre el icono de modificar valores( ). Verificar que ningún valor de grupo puede ser modificado en éste caso.
Eligiendo a un usuario de diferente tipo que administrador y modificar su tipo de usuario.	Desde el módulo de administración de usuarios, elegir a un usuario moderador, publicador o lector y pulsar sobre el icono de modificar valores( ). Cambiar el tipo de usuario a uno diferente al suyo. Verificar que el sistema haya hecho los cambios. Autentificarse con esa cuenta de usuario y comprobar que los permisos con que cuenta corresponden al nuevo valor de tipo de usuario.
Eligiendo a un usuario de diferente tipo que administrador y modificar su grupo de pertenencia.	Desde el módulo de administración de usuarios, elegir a un usuario moderador, publicador o lector y pulsar sobre el icono de modificar valores( ). Cambiar el grupo de usuario a uno diferente al suyo. Verificar que el sistema haya hecho los cambios. Autentificarse con esa cuenta de usuario y comprobar que los permisos con que cuenta corresponden al nuevo grupo de usuarios.
Realizar el proceso de modificación de perfil de usuario a cualquier usuario, ingresando al sistema con un usuario de perfil moderador, publicador y lector.	Verificar que esta opción no es posible de realizar por usuarios con estos perfiles, debido a que no tienen acceso a las opciones ni tienen los permisos para realizarlas.

DISEÑO DE LA PRUEBA CASO DE USO MODIFICA PERFILES DE USUARIO		
Caso de Prueba	**Error**	**Éxito**
Eligiendo a un usuario administrador y tratando de modificar su tipo de		Éxito

usuario.		
Eligiendo a un usuario administrador y tratando de modificar su grupo de pertenencia.		Éxito
Eligiendo a un usuario de diferente tipo que administrador y modificar su tipo de usuario.		Éxito
Eligiendo a un usuario de diferente tipo que administrador y modificar su grupo de pertenencia.		Éxito
Realizar el proceso de modificación de perfil de usuario a cualquier usuario, ingresando al sistema con un usuario de perfil moderador, publicador y lector.		Éxito

5.3 Comentarios acerca de los resultados obtenidos

Una actividad indispensable en todo proceso de desarrollo de software, es la prueba, si esta actividad no se llevara a cabo, se tendría un desconocimiento total sobre la calidad de los productos.

Después de haber concluido los resultados de las pruebas realizadas siguiendo la metodología para pruebas del proceso Objectory realizando pruebas clasificadas por los casos de uso definidos del sistema, en el cual se detectaron los errores que se cometieron en la construcción del sistema y que a su vez fueron corregidos dentro del mismo proceso, se asegura la calidad del proyecto construido ya que en el resultado final de dicho proceso se demuestra que todas las pruebas realizadas durante el ciclo de vida del desarrollo realizadas al sistema resultaron exitosas.

De esta manera, con el sistema completo y probado es posible continuar con el capítulo final de éste trabajo de tesis, que comprende los beneficios y alcances obtenidos al realizar la construcción de este sistema, los trabajos a futuro y por último las conclusiones finales a este trabajo.

CAPÍTULO

6

Conclusione

6.1 Conclusiones finales

La contribución de éste trabajo de tesis dentro del área de Ingeniería Asistida por Computadora del Instituto Mexicano del Petróleo radica en un mejoramiento de los procesos de distribución, captación, almacenamiento y seguimiento del conocimiento generado dentro del área modificando los procesos “manuales” de distribución de la información, a la utilización de un repositorio centralizado que utiliza como base de comunicación la intranet de la institución y la tecnología Web, dando una accesibilidad desde cualquier computadora conectada a la Intranet y que cuente con un navegador web.

Para la construcción del sistema de administración de documentos se emplearon tecnologías en rápido crecimiento y demanda para la programación sobre Intranet/Internet por medio de programación en el lenguaje orientado a objetos Java. Estas tecnologías permiten un acceso mas sencillo a las aplicaciones construidas pues emplean al navegador web como programa cliente, evitando la construcción e instalación de aplicaciones cliente para acceder a las aplicaciones de tipo cliente-servidor.

Para el diseño del sistema se empleo el Lenguaje de Modelado Unificado, el cual se utilizó para visualizar y especificar gráficamente cada una de las partes que comprenden el desarrollo del sistema de Administración de Documentos, con los procesos de negocio y funciones del sistema

que éste contiene por medio de diagramas de casos de uso, diagramas de secuencias y diagramas de clases y paquetes.

Para el desarrollo del sistema, se empleó la metodología Objectory para la creación de software, la cual, tiene como finalidad la posibilidad de obtener un producto de calidad. Dicha metodología basa su estrategia en los casos de uso del sistema que describen la funcionalidad del sistema y que sirven como base para todas las fases del desarrollo.

El Sistema de Administración de Documentos y Planos CAD, esta construido bajo el patrón de diseño Modelo Vista Controlador, el cual separa las tareas de lógica de negocio, control y presentación en tres entidades diferentes, lo cual permite una mayor claridad en la estructura del sistema. Esta estructura fue construida gracias al uso de JavaBeans para la parte del Modelo de la aplicación, Java Servlets en la parte de control y por último Java Server Pages para la parte de vista consistente en las interfaces de usuario. Estas tecnologías tienen claras ventajas sobre sus competidoras en el ámbito de las aplicaciones web, ya que constituyen una especificación que puede ser extendida hasta los límites que el desarrollador tenga capacidad y no una implementación de una tecnología existente. Además debido a la naturaleza del lenguaje Java, las aplicaciones creadas bajo éste esquema pueden ser ejecutadas en la mayoría de las plataformas existentes dentro de la escena computacional actual.

Se pudo comprobar que para la realización de un sistema de calidad no es necesario la adquisición de costosas licencias de software, ya que con el uso del software libre, se pueden realizar todas las tareas de desarrollo incluso con un mejor rendimiento y una mayor seguridad. El ejemplo mas claro de éste tipo de tecnologías es el mismo entorno de desarrollo que los creadores del lenguaje Java ofrecen a los desarrolladores para crear aplicaciones de tipo standalone o aplicaciones web sin ningún costo.

Referente a la calidad del sistema, se realizó un programa de pruebas integral basado en la metodología Objectory debido a que fue la metodología utilizada en el desarrollo del proyecto y que además garantiza una alta calidad en el producto final, apegado en su totalidad a los requerimientos que los usuarios demandan mediante pruebas realizadas en base a los casos de uso obtenidos al inicio de la planeación del sistema.

6.2 Trabajos a Futuro

Después de haber desarrollado este sistema de administración de documentos, y aunque se cubrieron los objetivos planteados en el inicio de éste trabajo de tesis, el sistema aún esta en la posibilidad de ser mejorado en diversos aspectos importantes como la seguridad, el mejor aprovechamiento de los recursos por medio de computación distribuida, la diversificación de medios de acceso, herramientas que mejoren la interfaz hombre-sistema, las mejoras y actualizaciones en patrones de diseño, entre algunas otras.

A continuación se enlistan algunas tecnologías que se podrían aplicar para un mejoramiento del SAD.

Autentificación JAAS de acuerdo al dominio NT.

Esta implementación consistiría en basar la autentificación en el/los dominio/s NT con que se cuentan. Esto tendría como consecuencia útil, que los usuarios del área de Ingeniería del IMP no tendrían que recordar otro nombre de usuario u otra contraseña para utilizar el sistema.

Al realizar la autentificación por este medio, el funcionamiento del sistema se tendría que modificar de acuerdo a los roles de usuario que se manejen dentro del dominio entre otras modificaciones, así mismo éstos roles y datos en general no se podrían modificar a conveniencia de los administradores del sistema, por lo que algunos módulos referentes a la administración de usuarios e incluso posibles tareas de los usuarios del sistema dejarían de ser funcionales, como el cambio de rol de usuario, cambio de contraseñas y datos de acceso para los usuarios.

Integrar a tomcat con el servidor web apache

Aunque el servidor Jakarta Tomcat es un servidor que soporta cargas importantes de trabajo, es necesario contar con una infraestructura mas sólida para sostener la aplicación, evitando a toda costa las caídas del sistema debidas a la baja capacidad de aceptar peticiones por parte de dicho servidor.

Una de las alternativas mas comunes en el caso de Tomcat 4.1, es la cooperación entre dicho servidor y el servidor web Apache[ApCH] por medio del uso del conector *mod_webapp* disponible en el sitio de apache, y que después de su configuración permite la participación de ambos servidores como uno solo, haciendo capaz al servidor Web apache, que solo tiene permitido inicialmente despachar páginas con HTML estático, convertirse en un motor procesador de tecnología Servlet y JSP.

Uso de EJB

Construir los actuales Java Beans bajo el esquema de Enterprise Java Beans[EJB] para lograr un mejor aprovechamiento de los recursos, bajo un esquema de computación distribuida para escalar la aplicación a entornos de mayor demanda de usuarios.

Indexación de documentos contenidos en el servidor con la herramienta Lucene

Lucene[JakL] está dentro de los proyectos de Jakarta Apache y entre otras características se encuentra la de indexar el contenido de documentos depositados en un servidor de aplicaciones. Para el caso del SAD, esta herramienta incrementaría enormemente sus posibilidades de búsquedas específicas dentro del contenido de los documentos a diferencia de la limitada búsqueda con que cuenta el sistema, el cual se basa únicamente en los nombres de archivo y comentarios que los usuarios colocaron al publicarlo.

Uso de Java Server Faces para separar aún más la lógica de negocio de la presentación

Aunque el sistema SAD esta basado en el modelo de programación MVC, y la lógica de negocio, de control y las interfaces de presentación están separadas, bajo este esquema aun se tiene que manejar cierto nivel de código java sobre las interfaces de presentación.

La propuesta de las Java Server Faces[JSF], nos brinda un nuevo esquema basado en el uso de la tecnología de TagLibs(librerias de etiquetas) separando completamente la lógica de las interfaces de usuario, utilizando sobre éstas únicamente código XML en su mayoría, creando y utilizando componentes que pueden ser reutilizables dentro de la misma u otras aplicaciones. Aunque ésta especificación aun esta en proceso de maduración ya puede ser implementada, pero con sus reservas debidas.

ANEXO

A

Instalación y Configuración del

La estructura de directorios con la que esta construido el SAD, sigue el modelo para la construcción de Archivos de Aplicaciones Web(WAR) [J2EEAT], que consiste en el directorio padre, el cual contiene las páginas estáticas y las páginas JSP. Debajo de éste directorio está localizado el directorio llamado *img/* que contiene las imágenes utilizadas en las interfaces de usuario, el directorio js, que contiene código que se ejecuta del lado del cliente consistente en archivos javaScript, el directorio css/ que contiene la hoja de estilo utilizada para dar un mejor formato a las interfaces de usuario y la carpeta Ut/ que contiene utilidades como el script para la creación de la base de datos en el servidor de bases de datos relacional MySQL.

Continuando con el estándar de estructura de directorios para paquetes tipo WAR, debajo del directorio raíz(llamado SAD), se encuentra el directorio WEB-INF, el cual contiene el archivo jaas.config, éste es el archivo de configuración para el módulo de autentificación de usuarios. El archivo web.xml que contiene la configuración de la aplicación web y que posteriormente se detallará. Dentro de éste directorio se encuentran los subdirectorios lib/ y classes/. El primero contiene todos los archivos JAR o librerías llamadas por las clases que corren del lado del servidor, a su vez el subdirectorio classes contiene todas las clases que corren del lado del servidor, en éste caso se trata de Servlets y JavaBeans, los cuales se encuentran distribuidos en 5 paquetes para su mejor distinción y organización.

Esta estructura descrita es totalmente portable entre los diferentes ambientes que se encuentran dentro de la especificación de los Java Servlet, o mas conocidos como contenedores de Servlets. Estos archivos son paquetes similares a los archivos JAR, y se les puede considerar un ejecutable para las aplicaciones web.

Después de describir la estructura básica con que esta construido el sistema de administración de planos y documentos, podemos comenzar a describir como fue la instalación sobre el Contenedor de Servlets y JSP Tomcat del proyecto Jakarta Apache, dentro de un sistema que no tenía cargado ninguno de los elementos necesarios para poder instalar este sistema.

1. Obtención e Instalación del JDK(Java Developer Kit)

2. Obtención e Instalación del Jakarta Tomcat 4.1[jT4.1].

3. Creación del contexto SAD sobre el subdirectorio <TOMCAT_HOME>/webapps/

Este punto consiste en la creación de la estructura de directorios descrita anteriormente .

4. Instalación de páginas JSP y páginas estáticas HTML sobre el directorio creado recientemente:
<TOMCAT_HOME>/webapps/SAD

5. Instalación de los archivos de configuración web.xml y jaas.config sobre el directorio <TOMCAT-HOME>/webapps/SAD/WEB-INF.

6. Instalación de los archivos JAR necesarios para la ejecución de las propias clases en el directorio <TOMCAT_HOME>/webapps/SAD/WEB-INF/lib. Estos consisten en los archivos JAR necesarios para el envío de correo electrónico, el driver JDBC necesario para realizar las conexiones con MySQL, y el archivo JAR uploadbean.jar necesario para la carga de documentos al servidor.

7. Instalación del paquete Sys, que contiene todas las clases necesarias para el correcto funcionamiento del SAD, bajo el subdirectorio <TOMCAT_HOME>/webapps/SAD/WEB-INF/classes.

8. Configuración del archivo web.xml ubicado en <TOMCAT_HOME>/webapps/SAD/WEB-INF/ para la configuración de los valores necesarios para que la aplicación envíe correo electrónico con el fin de realizar las notificaciones correspondientes a los usuarios. A continuación se muestra un fragmento de éste archivo para ejemplificar ésta configuración.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>

<servlet>.....</servlet>

<servlet>
<servlet-name>ControlUsers</servlet-name>
<servlet-class>Sys.users.ControlUsers</servlet-class>
<load-on-startup>0</load-on-startup>

<init-param>
<param-name>from</param-name>
<param-value>moralese@imp.mx</param-value>
</init-param>

<init-param>
<param-name>host</param-name>
<param-value>imp.mx</param-value>
</init-param>
</servlet>
<servlet-mapping>
...
</servlet-mapping>
</webapp>
```

Listado A.1 Ejemplo de configuración del archivo web.xml para establecer valores de mail.

9. Configuración del archivo web.xml para establecer el directorio donde se guardarán todos los documentos publicados en el sistema. En el siguiente listado se ejemplifica que el directorio que resguarda los documentos es /var/uploadSAD/.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>

<Servlet>.....</Servlet>

< servlet>
        <servlet-name>ControlFolder</servlet-name>
        <servlet-c lass>Sys.folder.ControlFolder</servlet-class>
                <load-on-startup>0</load-on-startup>

        <init-param>
            <param-name>path</param-name>
           <param-value>/var/uploadSAD </param-value>
        </init-param>
        </servlet>

<servlet-mapping>
...
</servlet-mapping>
</webapp>
```

Listado A.2 Ejemplo de configuración del archivo web.xml para establecer valores folder de carga

de archivos.

10. Configuración del archivo jaas.config, especificando el nombre de usuario, contraseña y nombre de base de datos a la que deberá contectarse para realizar la autentificación de usuarios del sistema. En el siguiente listado se muestra dicha configuración para los valores actuales del SAD.

```
SAD {
  Sys.auth.RdbmsLoginModule required debug="false"
url="jdbc:mysql://127.0.0.1:3306/arbol"
driver="org.gjt.mm.mysql.Driver" userdb="sad" passdb="imp";
};
```

Listado A.3 Ejemplo de configuración del archivo jaas.config.

Bibliografí

A continuación se muestran las referencias realizadas dentro del cuerpo de éste trabajo de tesis, así como bibliografía o enlaces de Internet útiles para el tema.

Referencia

apch **Apache Web Server**

http://httpd.apache.org/

cus **Casos de Uso UML**

Martin Fowler, Kendall Scott, UML gota a gota, cap 6, Pearson Educ. 1999.
http://www.dcc.uchile.cl/~psalinas/uml/casosuso.html

dom **Document Object Model**

http://www.w3.org/DOM/
Chang,Scardina,Karun Manual de Oracle XML Osborne-McGraw Hill Pag 259 Apendice

Ejb **Enterprise Java Beans**

http://java.sun.com/products/ejb

J2EEAT **J2EE Addendum Tutorial**

http://java.sun.com/j2ee/tutorial/

jaas **Java Authentication and Authorization Service**

http://java.sun.com/products/jaas/
http://www.javaworld.com/javaworld/jw-09-2002/ jw-0913-jaas.html

JakL **Jakarta Apache Lucene project**

http://jakarta.apache.org/lucene

java **Lenguaje de Programación Java**

http://java.sun.com/
http://java.sun.com/docs/books/tutorial/

jb **Java Beans**

http://java.sun.com/products/javabeans/
http://developer.java.sun.com/developer/ onlineTraining/Beans/Beans1/

jdbc **JDBC Data Access API**

Hobbs Ashton, Aprendiendo programación para bases de datos con JDBC Prentice Hall, 1998.
http://java.sun.com/products/jdbc
http://java.sun.com/docs/books/tutorial/jdbc/
http://mmmysql.sourceforge.net/ (driver JDBC-MySQL)

jdk14 **Java Developer Kit v.1.4.1**

http://java.sun.com/j2se/1.4.1

jmail **API de Java Mail**

http://java.sun.com/products/javamail/
http://www.javaworld.com/javaworld/jw-06-1999/ jw-06-javamail.html

js **JavaScript**

Powell, "JavaScript Manual de Referencia", McGraw Hill

JSF **Java Server Faces**

http://java.sun.com/j2ee/javaserverfaces
http://www.javaworld.com/javaworld/ jw-11-2002/jw-1129-jsf.html
javangelist.snipsnap.org/space/Java+Server+Faces

jsp **Java Server Pages**

http://java.sun.com/products/jsp
Marty Hall, "Servlets y JavaServer Pages", Prentice Hall, 2001

jT4.1 **Jakarta Tomcat 4.1**

http://jakarta.apache.org/tomcat
http://java.sun.com/products/jsp/tomcat/

mvc **Modelo Vista Controlador**

polaris.dit.upm.es/~jcduenas/patrones/Modelo.htm
st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html
http://ootips.org/mvc-pattern.html

mysql **Servidor de Bases de Datos MySQL**

MySQL Edición Especial DuBois Paul, Prentice may 2001
http://www.mysql.org

oby **Objectory Process**

http://www.iscn.at/select_newspaper/object/rational.html
http://www.iam.unibe.ch/~scg/Resources/UML/Mirror/PDF/objectory11.pdf

poo **Programación orientada a objetos**

Deitel y Deitel Cómo programar en JAVA. Prentice Hall 1999
http://www.javahispano.org/download/articulos/intro_poo.pdf

rhlin **Distribución Linux Red Hat Linux**

Lopez Camacho V, Linux, guía de instalación y administración McGraw Hill 2001
http://www.redhat.com
http://www.linux.org
http://www.linuxdoc.org/

sax **Simple API for XML**

Chang,Scardina,Karun Manual de Oracle XML Osborne-McGraw Hill Pag 259 Apendice
http://dev.w3.org/cvsweb/perl/modules/W3C/SAX/
http://www.saxproject.org/

servl **Servlets**

http://java.sun.com/products/servlet/
http://java.sun.com/docs/books/tutorial/servlets/
Marty Hall, "Servlets y JavaServer Pages", Prentice Hall, 2001

tresc **Arquitectura Tres Capas**

http://www.lsi.us.es/~tdg/res/3-tier-SP/

uml **United model language**

www.uml.org
www.rational.com/uml/index.jsp

XML **XML Extensible Mark Lenguage**

http://www.w3.org/TR/REC-xml
Buiding Web Services with Java Graham, Simenoinov SAMS pag. 33
Chang,Scardina,Karun Manual de Oracle XML Osborne-McGraw Hill Pag 259 Apendice

xslt **XSLT**

Chang,Scardina,Karun Manual de Oracle XML Osborne-McGraw Hill Pag 259 Apendice
www.w3.org/TR/xslt
www.w3.org/TR/xslt