Universidad de las Ciencias Informáticas

Facultad 5

Título

Desarrollo de un sistema de administración web para la línea de desarrollo Seguridad del Centro de Desarrollo de Informática Industrial.

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Autores

Yadeilis Durán Pérez

Yanet Torres Sales

Tutores

Ing. Enrique Reyes Bermúdez

Ing. Alejandro Manuel Rubinos Carvajal

Ciudad de La Habana, 2010

´´Día llegará en que pueda llevar el hombre consigo como hoy, el tiempo en el reloj un aparato diminuto que sea capaz de tener la luz, el color y la fuerza.´´

José Martí

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de este trabajo y reconocemos a la Universidad de las Ciencias Informáticas los derechos primordiales del mismo con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Yadeilis Durán Pérez

_____

FIRMA DE LA AUTORA

Ing. Alejandro Manuel Rubinos Carvajal

_____

FIRMA DEL TUTOR

Yanet Torres Sales

_____

FIRMA DE LA AUTORA

Ing. Enrique Reyes Bermúdez

_____

FIRMA DEL TUTOR

DATOS DE CONTACTO

Ing. Enrique Reyes Bermúdez.
Universidad de las Ciencias Informáticas, Habana, Cuba.
Email: ereyes@uci.cu

Ing. Alejandro Manuel Rubinos Carvajal.
Universidad de las Ciencias Informáticas, Habana, Cuba.
Email: amrubinos@uci.cu

AGRADECIMIENTOS

De Yadeilis:

A mis padres, quienes me infundieron la ética y el rigor que guían mi transitar por la vida, por apoyarme en todo cuanto hizo falta para que yo me sintiera tranquila y con ánimos para seguir adelante. Muchas gracias papi y mami, el apoyo constante de ustedes, a pesar de la distancia, ha sido fundamental para que yo pudiera llegar hasta aquí.

A mi hermano, por inyectarme su fuerza y su valor para conseguir este objetivo, por darme todo su cariño, su optimismo y comprensión.

A toda mi familia, en especial a mis tíos Rogelio y Belkis y a mis abuelos paternos, por ser para mí como unos padres, por estar siempre pendientes de mi, por ayudarme, a mi tía por cuidar a mis padres en mi ausencia, por brindarme todo su cariño y confianza, a mi tío por ese inmenso amor que me ha mostrado durante el transcurso de estos años, por estar siempre pendiente de mi, a mis abuelitos, porque se que anhelan tanto como yo este título, por ser tan amorosos y especiales.

A mis segundos padres y su familia, Francisca y Omar, por quererme como una hija mas, por estar siempre preocupados por mí, por su inmenso amor.

A mi novio, Reinier, por haber sido mi fiel compañero durante estos cinco años, por compartir los bueno y malos momentos, por su apoyo y amor.

A mi tutor Enrique, por su asesoramiento científico y estímulo para seguir creciendo intelectualmente, por su predisposición permanente e incondicional en aclarar mis dudas, por su sincera amistad.

A mi tutor Ignais, por haber asumido el guiarme y ayudarme en la ausencia de enrique, por brindarme todo su conocimiento y estar siempre disponible.

A todos los profesores que han forjado en mi todos los conocimientos y valores adquiridos, en especial a Matilde, que ha sido además, una gran amiga.

A mi compañera de tesis y amiga Yanet, por haber compartido conmigo durante todos estos años, por haber logrado una amistad que perdurara por siempre.

A mis compañeros y amigos de los grupo 5109, 5303 y 5506, a las muchachitas que hemos compartido apartamento, a todos ellos, quienes me acompañaron en esta trayectoria de aprendizaje y conocimientos, compartiendo buenos y malos momentos.

A mi gente de Guantánamo, las vecinas Mireya, Deisy y Mama Chela, y a mis amigos, Kiko, Masi y Elsa, por ser mis fieles amigos y no olvidarse de mí.

A la Revolución y la Universidad, por haberme brindado la posibilidad de ostentar hoy, el título de Ingeniera en Ciencias Informáticas.

A todos sinceramente, muchas gracias.

De Yanet:

A mi madre por darme las fuerzas para terminar cada día, por no dejar de llamarme ni un solo día en estos 5 años, por saber comprenderme y entenderme, por hacer tuya toda una carrera de 5 años.

A mis abuelos que quiero tanto, a los que están y a los que se fueron antes de tiempo. Están siempre en mi corazón.

A mi papa Rolando y a Magaly, por estar siempre ahí cuando los necesito y apoyarme. Por ser mis maestros de la vida.

A mis hermanos Roly y Claudia, mis hermanitos pequeños, estoy muy orgullosa de ustedes. Sigan siempre por el camino de la virtud.

A Haroldo por ser mi paz y mi tranquilidad, por querer tanto a mi madre y ser la persona especial que eres, mejor que tu no quiero a nadie.

A Guillermo, por llegar en partida doble, por ser el mejor amigo del mundo y el mejor hermano que la vida me ha podido dar. Por ser incondicionalmente mi confidente. Por estar ahí siempre que lo necesito y comprenderme siempre. Por seguir a mi lado aun cuando soy insoportable. Se que nos fajamos mucho, que siempre peleo por todo y que de vez en cuando me dan mis ataques de celos, pero esa es mi forma de decirte que "te quiero un montón" me queda corto. Me has dado el mejor momento de mi vida, conocerte.

A Josefa, Blaya y Luis Ángel por hacerme sentir siempre como en casa, por hacerme parte de la familia sin la cual mi vida no estaría completa.

A mis amigos de siempre Arturo, Lianet y Liset, por cuidarme siempre y apoyarme en todo. Por darme todos estos años de amistad incondicional, los cuales seguirán para siempre.

A mis amigas Celina, Lizandra y Dailén por brindarme su amistad incondicional y sincera, han sido de las mejores cosas que me han pasado en la universidad, nunca las olvidare, las quiero mucho.

A Yayi, mi compañera de tesis insoportable y una de mis mejores amigas, por recorrer la mayor parte de la universidad juntas y vivir momentos buenos y malos.

A mi tutor Kike por brindarme sus conocimientos, su amistad y su ayuda. Por ser el mejor tutor del mundo, mejor que tu, nadie.

A mi tutor Ignais por cuidar de la retaguardia en ausencia de Kike, por ayudarnos en todo, por nunca decirnos "no", por hacer de esta tesis, tu tesis.

A todas las personas que de una u otra han formado parte de mi vida en todos estos 5 años de universidad.

DEDICATORIA

De Yadeilis:

A mis padres, por su comprensión y ayuda en momentos malos y buenos. Me han enseñado a encarar las adversidades sin perder nunca la dignidad ni desfallecer en el intento. Me han dado todo lo que soy como persona, mis valores, mis principios, mi perseverancia y mi empeño y todo ello con una gran dosis de amor y sin pedir nunca nada a cambio, por haber confiado siempre en mi, por apoyarme en todas las decisiones que he tomado, por ser mi guía, mi ejemplo, por ser los seres mas maravillosos de este mundo.

A mi hermano, por ser mi ángel guardián, por estar a mi lado incondicionalmente durante estos cinco años, por haber sido mas que mi hermano, mi amigo, mi padre y todo para mi, por ofrecerme constantemente su inmenso amor, por darme animo, por cuidarme, por impulsarme a seguir adelante, por decirme siempre tranquila, todo saldrá bien, por hacerme sentir cada día mas orgullosa de él, por ser un ejemplo de perseverancia.

A mi negrito, mi sobrino, que aunque sé que falta mucho para que le llegue este momento, espero pueda ser orgullo y guía para él, por ser, Alejandrito, lo más hermoso que tengo en esta vida.

Para todos ustedes que son lo más importante de mi vida, es este título, los amo.

De Yanet:

A mi madre, Hilda Barbarita por ser la mejor madre del mundo, en una escala del 1 al 10 tienes un 100000. Por ser mí mejor amiga, por hacerme fuerte y perseverante, por enseñarme que en esta vida nada cae del cielo y que hay que luchar fuerte por conseguir lo que uno quiere. Por dedicarme todo tu tiempo y hacerme el centro de tu vida, por enseñarme a no caer ante las adversidades y a levantarme de nuevo para poder seguir. Por hacerme mujer más rápido de lo que pensé y ser mi ejemplo en la vida. Por confiar siempre en mí y nunca dudar. Te amo….

A mi abuela "Aya", por ser la vieja más linda del mundo. Por quererme tanto y hacer tuya mí lucha. Por darme esos consejos que tanto me enseñan. Por entenderme siempre, por valorarme y respetarme. Por enseñarme a ser mejor persona y nunca dudar de mí. Eres lo mejor de mi vida. Hoy se cumple nuestro mayor sueño. Te quiero mucho.

A mis abuelos "mami Hilda", "papi Ángel" y "abuelo Pilito", donde estén se que están orgullosos de mí porque hoy cumplí el sueño de sus vidas, verme hecha una profesional.

Resumen

Hoy en día los ataques a sistemas críticos son cada vez más frecuentes y la mayoría de las veces son llevadas a cabo por personas que trabajan dentro de las propias empresas. Estas personas dado que pueden moverse con facilidad dentro del centro, tienen acceso a atacar los recursos de más importancia, de allí la necesidad de tener un sistema que de alguna manera minimice esos riesgos. Para dar solución a dicho problema se realiza el siguiente trabajo de diploma que tiene como objetivo principal desarrollar un sistema web para administradores de seguridad sobre la plataforma de software libre capaz de brindarles confianza a los administradores de seguridad durante su trabajo en el sistema.

Para cumplir el objetivo se realizó un estudio de las diferentes actividades que se desarrollan por parte de los administradores de seguridad. Se investigaron las principales tecnologías de desarrollo y bibliotecas existentes en el mundo y se hace una propuesta fundamentada de las más importantes. Se presenta además la modelación del sistema a partir de diferentes diagramas y se realiza la implementación del mismo. Por último, se documentan las diferentes pruebas realizadas logrando finalmente la aceptación del producto.

Palabras Claves

Administrador, seguridad, sistema, software libre.

Abstract

The attacks to critical systems become nowadays more frequents and most of the times these are performed by employees working inside of the same enterprises .These people can attack the most important resources due to the fact that they can easily move inside their workplaces. There for the necessity of having a system that somehow minimizes those risks. To give solution to this problem it`s realized the following term paper, which has as a main objective to develop a web system to security administrators on an open source , able to bring confidence to security administrators during their work in the system.

To fulfill the objective it was realized a study to the different tasks to be developed by security administrators. Where the main technologies of development and bookstores around the world were researched and where a based proposal of the most important is made. Besides it is presented an example of the system from different diagrams and it´s realized the implementation of the same. Finally, the different tests realized are pointed out; accomplishing the acceptation of the product.

Key words

Administrator, security, system, open source.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La seguridad es un tema novedoso y en constante evolución a nivel mundial, tanto para fabricantes de tecnologías, diseñadores de la ingeniería de sus aplicaciones, como para las organizaciones profesionales que se orientan en el planteamiento de su normativa y regulación (Villalón, 2002).

Según la propia fuente, podemos entender como seguridad una característica de cualquier sistema (informático o no) que nos indica que ese sistema está libre de todo peligro, riesgo o daño, y que es, en cierta manera, lo menos vulnerable posible.

La seguridad informática no es un bien medible, en cambio, sí podríamos desarrollar diversas herramientas para cuantificar de alguna forma nuestra inseguridad informática, las cuales garantizarían y simplificarían la administración de seguridad ya que minimizan la misma e incrementan la automatización, la productividad y la eficiencia operativa, por otra parte, dan un impulso a la auto administración simplificando el monitoreo y la administración de los recursos y aplicaciones de seguridad a través del uso de interfaces de usuarios automatizadas. Los sistemas automatizados, debido a la alta sensibilidad de sus recursos, de los datos que manejan y las operaciones riesgosas que se pueden realizar, requieren de una funcionalidad que les garantice a los administradores de seguridad el control y monitoreo del acceso de los usuarios al sistema, garantizando robustez en el mismo (Villalón, 2002).

En la Universidad de las Ciencias Informáticas la producción de software y servicios informáticos se basa en la integración de los procesos de formación, investigación y producción en torno a una temática para convertirla en una rama productiva. En la facultad 5 se encuentra el “Centro de Desarrollo de Informática Industrial” (CEDIN) del cual la línea de desarrollo Seguridad forma parte, dicha línea propone una solución dirigida a cubrir los niveles básicos de seguridad en cuanto a autenticación, autorización y auditorías.

En la actualidad existe en la línea de desarrollo Seguridad una aplicación de escritorio para Administradores de Seguridad la cual permite entre otras funcionalidades, monitorear las sesiones activas y las alarmas que se disparan, posibilita la búsqueda de usuarios teniendo en cuenta diferentes criterios de búsqueda, se pueden mostrar los logs guardados en el servidor de seguridad además de proveer el control de diferentes variables como el número máximo de sesiones abiertas que puede tener un usuario. Dicha aplicación brinda supervisión y control de los procesos de seguridad en tiempo real, sin embargo, tal tendencia trae consigo que el monitoreo de la información por parte de los administradores de seguridad requiera de su presencia física en la computadora donde se ejecuta dicha aplicación.

Como los navegadores son una potente herramienta con las que cuentan los sistemas operativos, debido a que son fáciles de utilizar, requieren menos memoria y sobre todo poseen compatibilidad multiplataforma, sistemas para administradores de seguridad de este tipo podrían ser accesibles desde cualquier punto de la red sin tener que distribuir e instalar el mismo. Teniendo en cuenta lo anteriormente planteado se deriva la necesidad de contar con una herramienta informática para administradores de seguridad que mediante la web permita su control total desde cualquier localización de trabajo remoto.

Ante esta situación se puede definir el siguiente **problema científico**: ¿Cómo garantizar el control total del sistema para administradores de seguridad en la línea de desarrollo Seguridad del Centro de Desarrollo de Informática Industrial desde la web?

Para ello el **objeto de estudio** lo constituye: Componentes y herramientas de desarrollo de interfaz de usuario.

Para dar respuesta al problema de la investigación se definió que el **objetivo general** consistiera en: Desarrollar un módulo de visualización web para el sistema de administración de la línea de desarrollo Seguridad del Centro de Desarrollo de Informática Industrial.

Identificándose que el **campo de acción** que abarca esta investigación sería: Componentes de desarrollo de interfaz de usuario en la web.

Conforme a este planteamiento se derivan las siguientes **tareas**:

- Estudio de la aplicación de escritorio para administradores de seguridad de la línea de desarrollo Seguridad.
- Obtención de los requisitos de la aplicación de escritorio de la línea de desarrollo Seguridad desarrollado con la biblioteca Gtk a través de su documentación.
- Selección de las herramientas y tecnologías de desarrollo web mediante el estudio del estado del arte.
- Realización de la documentación correspondiente al análisis utilizando los diferentes diagramas de modelado.
- Realización de la documentación correspondiente al diseño utilizando los diferentes diagramas de modelado.

- Análisis de los patrones de diseño con el objetivo de obtener un diseño flexible y reutilizable y aplicable a las posibles mejoras de la arquitectura de la aplicación.
- Desarrollo de la aplicación que cumpla las funcionalidades del sistema para administradores de seguridad con los requerimientos especificados.
- Documentación de pruebas para próximas iteraciones.

Diseño metodológico

Durante todo el período de indagación se utilizaron un conjunto de métodos científicos de investigación. Estos métodos se clasifican en:

- **Teóricos:**

  Posibilitan el conocimiento del estado del arte del fenómeno, su evolución en una etapa determinada, su relación con otros fenómenos, así como su aislamiento como objeto estudiado.

Dentro de los teóricos se emplearon los siguientes:

- **Analítico sintético:** Para analizar todo el contenido en los documentos revisados, sintetizar, clasificar y evaluar la información valiosa, y ponerlo en práctica en el momento de analizar el sistema de administración de la línea de desarrollo Seguridad y llegar a conceptos y generalizaciones.
- **Análisis-Histórico-Lógico:** Con el objetivo de conocer las diferentes herramientas y tecnologías necesarias para el desarrollo del sistema web destacando sus ventajas y desventajas obteniendo cuál o cuáles de ellos utilizar.
- **Inductivo-Deductivo:** Permite llegar al final del análisis con un grupo de conocimientos generalizados, manifestándose la necesidad que existe en la línea de desarrollo Seguridad del Centro de Desarrollo de Informática Industrial, de tener bien presente las ventajas que ofrecen los sistemas para administradores de Seguridad web.

- **Empíricos**:

Estos métodos permiten extraer de los fenómenos analizados las informaciones que se necesitan sobre ellos a través de observaciones, del uso de técnicas de opinión y la propia experimentación.

Dentro de los empíricos se emplearon los siguientes:

- **Observación:** Se han consultado varios documentos mientras se llevaba a cabo la investigación para el estudio de las diferentes herramientas y tecnologías web a utilizar.
- **Experimento:** Con el objetivo de facilitar la construcción de los diferentes casos de prueba que se realizan en algunos escenarios de los entregables.
- **Modelación:** Para la elaboración de los diferentes diagramas necesarios para el entendimiento y mantenimiento del producto.

  Los diagramas son:

  - Diagramas de Casos de Usos
  - Diagramas de Clases
  - Diagramas de Interacción
  - Diagramas de Componentes
  - Diagrama de Despliegue

La siguiente investigación estará estructurada de la siguiente forma:

Capítulo 1: “Fundamentación teórica”: En este capítulo se explicará en detalles las funcionalidades del sistema para administradores de seguridad y sus posibles mejoras. Se explican además las metodologías y tecnologías actuales a considerar para realizar la selección de aquellas que se van a utilizar para el desarrollo de la aplicación web para administradores de seguridad teniendo en cuenta también los requisitos del usuario.

Capítulo 2: “Características del Sistema”**:** En este capítulo se realizará la descripción de la aplicación web para administradores de seguridad a desarrollar para lograr una mayor claridad y comprensión por parte de los desarrolladores. Los puntos que se abordarán son los siguientes: descripción del modelo de dominio, especificación de los requisitos funcionales y no funcionales, representación de los actores y casos de uso del sistema a través de un diagrama de casos de uso del sistema, así como la descripción textual de cada uno de ellos.

Capítulo 3: "Diseño del Sistema"**:** El contenido que se aborda en este capítulo estará relacionado con el diseño del sistema. El mismo incluye la descripción de los estilos arquitectónicos y patrones de diseño seleccionados, además de explicar cómo se adaptan y se representan cada uno de ellos en el diseño del sistema. Por otra parte, se realizarán todos lo diagramas de clases y de interacción que den soporte a cada uno de los requisitos del sistema, incluyendo los no funcionales.

Capítulo 4: "Implementación y Prueba": En el desarrollo de este capítulo se describirá la implementación de la aplicación web para administradores de seguridad usando componentes. Además, se podrá apreciar con una breve descripción, las interfaces que le dan solución a las diferentes funcionalidades de la misma, además de dar una descripción de las pruebas realizadas a la herramienta web.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se explicará en detalles las funcionalidades del sistema para administradores de seguridad y sus posibles mejoras. Se explican además las metodologías y tecnologías actuales a considerar para realizar la selección de aquellas que se van a utilizar para el desarrollo de la aplicación web para administradores de seguridad teniendo en cuenta también los requisitos del usuario.

1.1 Seguridad

La seguridad informática consiste en garantizar que los recursos del sistema de información (material informático o programas) de una organización sean utilizados de la manera que se decidió y que el acceso a la información allí contenida así como su modificación sólo sea posible a las personas que se encuentren acreditadas y dentro de los límites de su autorización (CRYPTEX, 2010).

La seguridad de la Información se ha convertido en un área clave en el mundo interconectado de hoy. Día a día, en los principales medios de comunicación se repiten los ataques de virus, hackers y otros peligros tecnológicos. Desde el ámbito corporativo y gubernamental, la búsqueda de profesionales en Seguridad Informática se ha duplicado y la tendencia sigue en aumento.

A grandes rasgos se entiende que mantener un sistema seguro consiste básicamente en garantizar tres aspectos: confidencialidad, integridad y disponibilidad. Algunos estudios integran la seguridad dentro de una propiedad más general de los sistemas, la confiabilidad, entendida como el nivel de calidad del servicio ofrecido. Consideran la disponibilidad como un aspecto al mismo nivel que la seguridad y no como parte de ella, por lo que dividen esta última en sólo las dos facetas restantes, confidencialidad e integridad (Villalón, 2002).

Según la propia fuente, la confidencialidad dice que los objetos de un sistema han de ser accedidos únicamente por elementos autorizados a él, y que esos elementos autorizados no van a convertir esa información en disponible para otras entidades; la integridad significa que los objetos sólo pueden ser modificados por elementos autorizados, y de una manera controlada, y la disponibilidad indica que los objetos del sistema tienen que permanecer accesibles a elementos autorizados.

La información de una empresa o lugar en específico se puede ver afectada por muchos factores, incidiendo básicamente en los aspectos de confidencialidad, integridad y disponibilidad de la misma. Desde el punto de vista del usuario, uno de los problemas más importantes puede ser el que está

relacionado con el delito o crimen informático, por factores externos e internos. Una persona no autorizada podría: clasificar y desclasificar los datos, filtrar información, alterar la información, borrar la información, usurpar datos, hojear información clasificada, entre otros (Villalón, 2002).

1.1.1 Activos de Seguridad

Los activos son los elementos que la seguridad informática tiene como objetivo proteger. Son tres elementos que los conforman:

- **Información:** Es el objeto de mayor valor para una organización, el objetivo es el resguardo de la información, independientemente del lugar en donde se encuentre registrada, en algún medio electrónico o físico.
- **Equipos que la soportan:** Software, hardware y organización.
- **Usuarios:** Individuos que utilizan la estructura tecnológica y de comunicaciones que manejan la información (Taringa, 2009).

Según la propia fuente, el activo más importante que se posee es la información y, por lo tanto, deben existir técnicas que la aseguren, más allá de la seguridad física que se establezca sobre los equipos en los cuales se almacena. Estas técnicas las brinda la seguridad lógica que consiste en la aplicación de barreras y procedimientos que resguardan el acceso a los datos y sólo permiten acceder a ellos a las personas autorizadas para hacerlo.

Los medios para conseguirlo son:

1. Restringir el acceso (de personas de la organización y de las que no lo son) a los programas y archivos.
2. Asegurar que los operadores puedan trabajar pero que no puedan modificar los programas ni los archivos que no correspondan (sin una supervisión minuciosa).
3. Asegurar que se utilicen los datos, archivos y programas correctos en/y/por el procedimiento elegido.
4. Asegurar que la información transmitida sea la misma que reciba el destinatario al cual se ha enviado y que no le llegue a otro.

5. Asegurar que existan sistemas y pasos de emergencia alternativos de transmisión entre diferentes puntos.

6. Organizar a cada uno de los empleados por jerarquía informática, con claves distintas y permisos bien establecidos, en todos y cada uno de los sistemas o aplicaciones empleadas.

7. Actualizar constantemente las contraseñas de accesos a los sistemas de cómputo.

RSA[1], la división de seguridad de EMC[2], ha presentado los resultados de su Encuesta Global de Seguridad del Consumidor Online 2010 que refleja la opinión de más de 4.500 consumidores de todo el mundo en relación con su nivel de concienciación sobre las amenazas online existentes, sus preocupaciones sobre la seguridad de su información personal o su predisposición a compartirla en la red, así como de sus deseos de contar con una mejor protección de sus datos personales (CRYPTEX, 2010).

Según la propia fuente, el estudio de RSA muestra que los consumidores que utilizan la banca online (86%) comparten mayores preocupaciones sobre el robo de sus datos personales que aquellos que utilizan portales relacionados con la salud (64%) o webs gubernamentales (68%). Como resultado de estas preocupaciones, más de la mitad de los encuestados aseguraron que son poco propensos a compartir información o a interactuar en estas webs.

Los consumidores estuvieron de acuerdo en que sus datos personales deberían estar mejor protegidos que por un simple nombre de usuario y contraseña en las webs de redes sociales (59%), de servicios médicos (64 %), gubernamentales (70%) y de la banca online (80%). Nueve de cada diez consumidores están deseando utilizar unas medidas de seguridad más fuertes que las actuales, según CRYPTEX (2010).

1.1.2 Administradores de Seguridad

Existen diferentes conceptos de Administrador de Seguridad, entre los más acertados se encuentran los siguientes:

[1] División de Seguridad de EMC, principal proveedor de soluciones de seguridad para la aceleración de los negocios, ayudando a las organizaciones líderes mundiales a conseguir el éxito a través de la resolución de los desafíos de seguridad más complejos y sensibles.

[2] Empresa que integra la American Fortune 500 y S&P 500 fabricante de software y sistemas para administración y almacenamiento de información.

Un administrador de seguridad no es más que la persona que es responsable de la definición o aplicación de una o más partes de una política de seguridad, pero existen diferentes usuarios que sin tener autorización intentan realizar dichas funciones (CRYPTEX, 2010).

El administrador de seguridad es el responsable de la seguridad física y lógica de los recursos de un sistema. Es el que administra el programa de seguridad de los datos que se manejan (Soluciones_Seguras, 2009).

Según la propia fuente, un administrador de seguridad informática es un individuo especialmente hábil. Esta persona debe ser capaz de:

- Reconocer las exposiciones de seguridad actual y potencial.
- Desarrollar soluciones en un entorno de cambios constantes de tecnología de los ordenadores (sistemas virtuales, tiempo real, etc.…).
- Fijar los diseños estándares de seguridad para los programas de aplicación.
- Establecer los procedimientos administrativos y el criterio de desarrollo de programación para los sistemas de seguridad futura.
- Negociar e incluir a la dirección media y principal en las emisiones de análisis de riesgo, respecto a la seguridad contra la utilidad de los recursos del ordenador.

Además, un Administrador de Seguridad debe:

Asegurar

- que la conectividad sea estable y sin abusos.
- que la presencia externa esté disponible.
- que los atacantes no pueden entrar.
- que los usuarios no abusan de la salida.

Mantener

- una copia de respaldo de todos sus equipos.

- un plan de contingencia para las eventualidades más comunes.
- un historial de las actividades de los usuarios.
- un áudito de las actividades de administración.

Optimizar

- el uso del Internet en la oficina.
- el uso de los recursos de la empresa (Soluciones_Seguras, 2009).

1.1.2.1 Funciones y Obligaciones de un Administrador de Seguridad

La función del administrador de seguridad entiende, lógicamente, de todo lo relacionado con las cuestiones de seguridad inherentes y exigibles al sistema informático.

Funciones:

1. Llevar a cabo las actividades de administración de la seguridad mediante la gestión de perfiles y controles de acceso.
2. Verificar que las medidas de seguridad física y lógica protegen los datos personales y mantener informado al responsable de seguridad.
3. Colaborar con la identificación de los datos especialmente susceptibles de protección, según los modelos de clasificación que existan.
4. Analizar posibles transgresiones e irregularidades en los accesos.
5. Evaluar la seguridad de paquetes, aplicaciones, productos y dispositivos, antes de su adquisición o implantación.
6. Dar soporte técnico en materia de seguridad, a los desarrolladores, técnicos y usuarios en general.
7. Proveer un apoyo administrativo directo para la instalación de sistemas de seguridad para asegurar el uso seguro de todos los sistemas en línea y de información.
8. Establecer objetivos para el desarrollo futuro de los sistemas de seguridad para la evolución de los sistemas en línea y de información.

9. Administración del sistema de password del sistema en el que se trabaje.
10. Revisión periódica del logging del sistema.

Obligaciones:

1. Conocer la normativa interna en materia de seguridad y especialmente la referente a protección de datos de carácter personal.
2. Cumplir lo dispuesto en la normativa interna vigente en cada momento.
3. Conocer las consecuencias que se pudieran derivar y las responsabilidades en que pudiera incurrir en caso de incumplimiento de la normativa, que podrían derivar en sanciones (Rivas, 1988).

1.2 Metodologías a utilizar, tendencias y tecnologías actuales

Para la realización de una aplicación informática se deben definir los tipos de tecnologías a utilizar, así como las herramientas y las metodologías que serán de mayor utilidad para su implementación. A continuación se explicarán brevemente los detalles de cada una de estas tecnologías seleccionadas para el desarrollo e implementación de la aplicación.

1.2.1 Sistema operativo

Un sistema operativo puede ser contemplado como una colección organizada de extensiones software del hardware, consistente en rutinas de control que hacen funcionar al computador y proporcionan un entorno para la ejecución de programas. Estos programas utilizan las facilidades proporcionadas por el sistema operativo para obtener acceso a recursos del sistema informático como el procesador, archivos y dispositivos de entrada/salida (E/S). De esta forma, el sistema operativo constituye la base sobre la cual pueden escribirse los programas de aplicación, los cuales invocarían sus servicios por medio de llamadas al sistema. Por otro lado, los usuarios pueden interactuar directamente con él a través de órdenes concretas. En cualquier caso, actúan como interfaz entre los usuarios/aplicaciones y el hardware de un sistema informático (El_Rincon_de_Linux, 2009).

- **GNU/Linux**

GNU/Linux es un sistema operativo, compatible Unix[3]. Dos características muy peculiares lo diferencian del resto de sistemas que podemos encontrar en el mercado, la primera, es que es libre, esto significa que no tenemos que pagar ningún tipo de licencia a ninguna casa desarrolladora de software por el uso del mismo, la segunda, es que el sistema viene acompañado del código fuente.

Características de GNU/Linux

- **Multitarea:** GNU/Linux utiliza la llamada multitarea preventiva, la cual asegura que todos los programas que se están utilizando en un momento dado serán ejecutados, siendo el sistema operativo el encargado de ceder tiempo de microprocesador a cada programa.
- **Multiusuario:** Muchos usuarios usando la misma máquina al mismo tiempo.
- **Multiprocesador:** Soporte para sistemas con más de un procesador está disponible para Intel[4], AMD[5] y SPARC[6].
- **Funciona en modo protegido 386.**
- **Protección de la memoria entre procesos**, de manera que uno de ellos no pueda colgar el sistema.
- **Carga de ejecutables por demanda:** GNU/Linux sólo lee del disco aquellas partes de un programa que están siendo usadas actualmente.
- **Política de copia en escritura para la compartición de páginas entre ejecutables:** Esto significa que varios procesos pueden usar la misma zona de memoria para ejecutarse. Cuando alguno intenta escribir en esa memoria, la página (4Kb de memoria) se copia a otro lugar. Esta política de copia en escritura tiene dos beneficios: aumenta la velocidad y reduce el uso de memoria.

[3] (registrado oficialmente como **UNIX®**) es un sistema operativo portable, multitarea y multiusuario.
[4] Integrated Electronics Corporation es el más grande fabricante de chips semiconductores basado en ingresos.
[5] **(Advanced Micro Devices, Inc.)**Es una de las compañías más grandes del mundo en producción de microprocesadores compatibles x86 (junto a Intel) y uno de los más importantes fabricantes de CPUs, GPUs, chipsets y otros dispositivos semiconductores.
[6] **(Scalable Processor ARChitecture)** es una arquitectura RISCbig-endian. Es decir, una arquitectura con un conjunto reducido de instrucciones.

- **Memoria virtual usando paginación (sin intercambio de procesos completos) a disco:** A una partición en el sistema de archivos, con la posibilidad de añadir más áreas de intercambio sobre la marcha.
- **Librerías compartidas de carga dinámica:** DLL's[7] y librerías estáticas.
- **Todo el código fuente está disponible**, incluyendo el núcleo completo y todos los drivers, las herramientas de desarrollo y todos los programas de usuario; además todo ello se puede distribuir libremente. Hay algunos programas comerciales que están siendo ofrecidos para GNU/Linux actualmente sin código fuente, pero todo lo que ha sido gratuito sigue siendo gratuito.
- **Control de tareas POSIX[8].**
- **Soporte para muchos teclados nacionales o adaptados** y es bastante fácil añadir nuevos dinámicamente.
- **Consolas virtuales múltiples:** Varias sesiones de logging a través de la consola entre las que se puede cambiar con las combinaciones adecuadas de teclas (totalmente independiente del hardware de video). Estas consolas se crean dinámicamente y puedes tener hasta 64.
- **Sistema de archivos de CD-ROM** que lee todos los formatos estándar de CD-ROM.
- **TCP/IP**[9], incluyendo Telnet[10], NFS[11], etc.
- **Software cliente y servidor NetWare**[12] (El_Rincon_de_Linux, 2009)**.**

[7] Biblioteca de enlace dinámico (dinamic-link library) es el término con el que se refiere a los archivos con código ejecutable que se cargan bajo demanda de un programa por parte del sistema operativo.
[8] **(Portable Operating System Interface; la X viene de UNIX)**, son una familia de estándares de llamadas al sistema operativo. Persiguen generalizar las interfaces de los sistemas operativos para que una misma aplicación pueda ejecutarse en distintas plataformas.
[9] Es un conjunto de protocolos de red en los que se basa Internet y que permiten la transmisión de datos entre redes de computadoras.
[10] **(TELecommunication NETwork)** es el nombre de un protocolo de red (y del programa informático que implementa el cliente), que sirve para acceder mediante una red a otra máquina, para manejarla remotamente como si estuviéramos sentados delante de ella.
[11] **Network File System** (Sistema de archivos de red) es un protocolo de nivel de aplicación, según el Modelo OSI.
[12] Es un Sistema operativo de red. Es una de las plataformas de servicio más fiable para ofrecer acceso seguro y continuado a la red y los recursos de información, sobre todo en cuanto a servidores de archivos.

Como distribución de este sistema operativo se encuentra Debian, este es un sistema operativo (S.O.) libre. Debian utiliza el núcleo GNU/Linux (el corazón del sistema operativo), pero la mayor parte de las herramientas básicas vienen del Proyecto GNU; de ahí el nombre GNU/Linux (Debian, 2010).

1.2.2 Metodologías de desarrollo

Se define como metodología al conjunto de estrategias, métodos o actividades intencionadas, organizadas, secuenciadas e integradas, que permitan el logro de aprendizajes significativos y de calidad (Lagos, 2008).

Hoy en día a los desarrolladores de grandes aplicaciones se les dificulta realizar una planificación organizada del trabajo, mantener las políticas del trabajo en equipo y a la vez asumir los adelantos en la informatización de los procesos productivos. Es por ello que organizaciones y empresas requieran cada vez más de aplicaciones confiables y de mejor calidad tanto para su desarrollo como para su mantenimiento. Para contribuir con las necesidades de estas organizaciones y empresas la comunidad desarrolladora de software ha estandarizado un conjunto de procedimientos ya definidos con el fin de complementar una metodología de trabajo a seguir para el desarrollo de productos software.

- **Rational Unified Process (RUP)**

"Un proceso de desarrollo de software define quién hace qué, cómo y cuándo. RUP define cuatro elementos: los roles, que responden a la pregunta ¿Quién?, las actividades que responden a la pregunta ¿Cómo?, los productos, que responden a la pregunta ¿Qué? y los flujos de trabajo de las disciplinas que responden a la pregunta ¿Cuándo?"(Jacobson, 2000)

Según la propia fuente, RUP (Proceso Unificado de Desarrollo) es el proceso más apropiado para el desarrollo de grandes proyectos, dado que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas. RUP es un proceso bien definido, estructurado y adaptable específicamente a cada proyecto. Se define a partir de tres características esenciales: estar dirigido por los Casos de Uso, centrado en la arquitectura y ser iterativo e incremental.

RUP es un proceso iterativo e incremental que se encarga de dividir el trabajo en partes más pequeñas o en mini proyectos, permitiendo que se logre un equilibrio entre la arquitectura y casos de uso durante cada mini proyecto. En cada iteración del proyecto o mini proyecto como se ha definido se obtiene un incremento, provocando un aumento del producto (Jacobson, et al., 2004).

RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al final de cada ciclo.

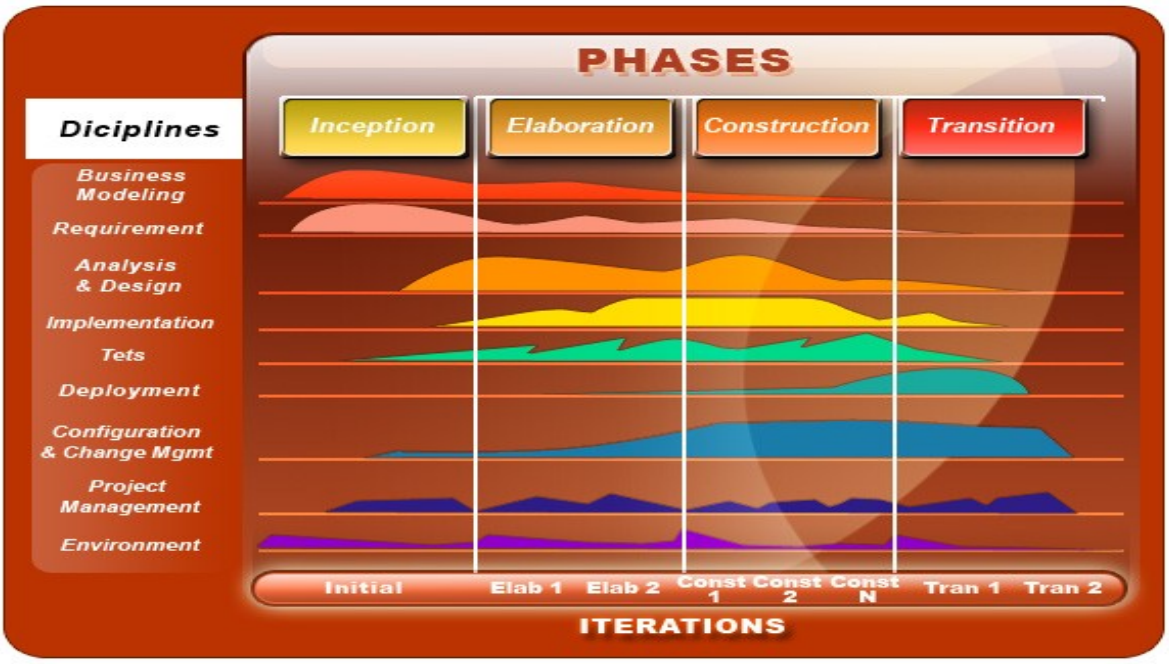


Figura 1 Metodología de desarrollo de software: RUP

Este proceso de desarrollo está definido por 4 fases fundamentales:

- **Conceptualización (Concepción o Inicio)**: Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.
- **Elaboración**: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido.
- **Construcción**: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtiene uno o varios entregables del producto que han pasado las pruebas. Se ponen estos entregables a consideración de un subconjunto de usuarios.
- **Transición**: El entregable ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores (Jacobson, 2000).

Según la propia fuente, cada fase del RUP concluye con un hito bien definido, en el cual se deben tomar determinadas decisiones y alcanzar las metas clave antes de pasar a la siguiente fase. Los hitos para cada una de las fases son:

- **Inicio**: Visión de los objetivos.
- **Elaboración**: Prototipo de la arquitectura.
- **Construcción**: Capacidad operacional inicial.
- **Transición**: Liberación del producto.

Los artefactos son productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables (Jacobson, 2000).

Roles y Artefactos que propone RUP

RUP define un conjunto de roles, agrupados por participación en tareas relacionadas estos son: Analistas, Desarrolladores, Gestores, Apoyos, Especialista en Prueba y Otros Roles Adicionales.

De acuerdo con las necesidades y características de la investigación los roles que se desarrollarán serán analista, diseñador y programador perteneciente a los grupos de analistas y desarrolladores definido por el Proceso Unificado de Desarrollo (RUP) respectivamente.

- **Analista**: Es el responsable del conjunto de requisitos que están modelados en los casos de usos específicos, es su responsabilidad también delimitar el sistema, encontrando los actores y los casos de uso y asegurando que el modelo de casos de uso es completo y consistente.

Los artefactos que serán producidos por el analista serán los siguientes:

- Modelo de Caso de Uso
- Glosario de Términos
- Actor del Sistema
- Caso de Uso del Sistema

- Vista de Caso de Uso

➢ **Diseñador de Sistema:** Es el responsable del diseño de una parte del sistema, el cual contiene los requisitos, la arquitectura y el desarrollo de proceso para el proyecto.

Los artefactos que serán producidos por el diseñador se describen a continuación:

- Clase del Análisis
- Diagrama de clases del Análisis
- Clase del Diseño
- Realización de caso de uso
- Subsistema de Diseño

➢ **Implementador:** Es el responsable de desarrollar y probar componentes, en acuerdo con las normas adoptadas en el proyecto para la integración de subsistemas más grandes.

Los artefactos que serán realizados por el implementador se describen a continuación:

- Diagrama de componentes

La selección de una metodología de desarrollo para lograr un producto con mayor calidad, es una de las dificultades que presentan los desarrolladores de software actualmente. El Proceso Unificado de Software (RUP) y Programación Extrema (XP) son los procesos más utilizados y mejor documentados a nivel mundial (Jacobson, 2000).

¿Por qué RUP?

Teniendo en cuenta que RUP es un proceso de desarrollo de software que se caracteriza por su empleo en proyectos de larga duración y con un alto grado de complejidad, ha sido seleccionado para el desarrollo de este proyecto como la metodología de trabajo a utilizar. Para la elección de este proceso no solo se han tenido en cuenta las ventajas que nos brinda, sino que también es la metodología de trabajo más utilizada en la institución a la cual pertenece el equipo de desarrolladores. La metodología utilizada para el desarrollo de la herramienta es RUP con UML como lenguaje de modelado. Además, es la metodología escogida para el trabajo en el centro.

1.2.2.1 Lenguaje de modelado

RUP utiliza como lenguaje de modelado: Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language)

"UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema software" (Roger, et al., 2005).

Figura 2 Lenguaje Unificado de Modelado

Según la propia fuente, UML utiliza herramientas de modelado visual que facilita la gestión de dichos modelos, permitiendo ocultar o exponer detalles cuando sea necesario. Ayuda a mantener la consistencia entre los artefactos del sistema: requisitos, diseños e implementaciones. UML también intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implementar un lenguaje de modelado común para todos los desarrolladores se crea una documentación también común, de forma tal que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

UML se ha convertido en un estándar para representar y modelar la información con la que se trabaja en las fases de análisis y especialmente en la fase de diseño.

Está compuesto por tres clases de bloques de construcción formadas por los elementos, que son abstracciones de cosas reales y ficticias; las relaciones, que permiten relacionar los elementos y por último están los diagramas que representan las colecciones de elementos con sus relaciones. UML es además un método formal de modelado aportando como ventajas: mayor rigor en la especificación, pues

permite realizar una verificación y validación del modelo realizado, y automatizar determinados procesos (Roger, et al., 2005).

1.2.3 Herramientas CASE

En las últimas décadas se ha trabajado en el área de desarrollo de sistemas para encontrar técnicas que permitan incrementar la productividad y el control de calidad en cualquier proceso de elaboración de software. Hoy en día la tecnología Computer Aided Software Engineering (CASE, por sus siglas en inglés) reemplaza al papel y al lápiz por el ordenador para transformar la actividad de desarrollar software en un proceso automatizado.

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) son un conjunto de programas y ayuda que dan asistencia a los analistas, ingenieros de software y desarrolladores durante todos los pasos del ciclo de vida de desarrollo de un software (INFORMATICA, 1999).

- **Visual Paradigm**

Es una herramienta CASE para visualizar y diseñar elementos de software, para ello hace uso del lenguaje de modelado UML. Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Facilita la interoperabilidad con otras herramientas CASE como el Rational Rose y puede ser integrada con herramientas Java. Posee además gran facilidad de uso, acelera el desarrollo de aplicaciones, ya que sirve de puente visual entre arquitectos, analistas y diseñadores de sistemas de información, haciendo el trabajo más fácil y dinámico (Giraldo, y otros, 2005).

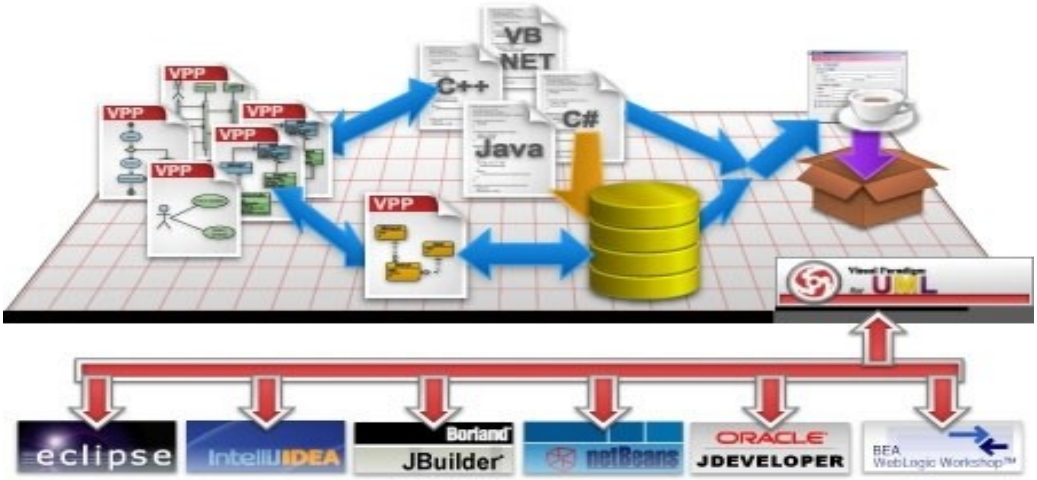


Figura 3 Visual Paradigm

Según la propia fuente, Visual Paradigm es una herramienta fácil de usar que soporta la última notación UML 2.1, ingeniería inversa, generación de código, importación desde la herramienta Rational Rose, exportación/importación XMI[13], generador de informes, editor de figuras, integración con Microsoft Visio[14], plug-in, integración IDE con Visual Studio[15], IntelliJ IDEA[16], Eclipse, NetBeans[17] y otros. Entre sus nuevas características se incluyen el modelado colaborativo con CVS[18] y Subversion[19], interoperabilidad con modelos UML2 a través de XMI, etc.

En nuestra universidad el Visual Paradigm es una herramienta libre ya que la universidad cuenta con la licencia para su utilización.

Características (Giraldo, y otros, 2005)

- Producto de calidad.
- Soporta aplicaciones web.

[13] Estándar para el intercambio de meta modelos usando XML.
[14] Software de dibujo vectorial para Microsoft Windows.

[15] Entorno de desarrollo integrado para sistemas operativos Windows.
[16] Entorno inteligente para desarrollar aplicaciones Java, cliente y servidor.
[17] Entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas.
[18] **Concurrent Versions System** es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros que forman un proyecto y permite que distintos desarrolladores colaboren.
[19] Software de sistema de control de versiones.

- Las imágenes y reportes generados, no son de muy buena calidad.
- Varios idiomas.
- Generación de código para Java y exportación como HTML.
- Fácil de instalar y actualizar.
- Compatibilidad entre ediciones.

Para el desarrollo de nuestra aplicación se selecciona la herramienta CASE Visual Paradigm debido a que es multiplataforma y brinda muchas facilidades para el diseño de los diagramas necesarios y su documentación. Apoya a un conjunto de idiomas tanto en la generación de código como en la ingeniería inversa de los mismos, permitiendo la importación y exportación de archivos XML de diferentes versiones. Emplea una rápida respuesta y con poca memoria, requisito que le permite manejar grandes y complicadas estructuras de un proyecto en una forma muy eficiente y, sin embargo, sólo requiere de una configuración de escritorio. Posee además gran facilidad de uso, acelera el desarrollo de aplicaciones, ya que sirve de puente visual entre arquitectos, analistas y diseñadores de sistemas de información, haciendo el trabajo más fácil y dinámico. Además, en el centro se está trabajando con Visual Paradigm ya que tenemos la licencia.

1.2.4 Lenguajes de Programación

Los lenguajes de programación son herramientas que nos permiten crear programas y software. Entre ellos tenemos Delphi, Visual Basic, Pascal, Java, entre otros. Los lenguajes de programación de una computadora en particular se conocen como código de máquinas o lenguaje de máquinas. Estos facilitan la tarea de programación, ya que disponen de formas adecuadas que permiten ser leídas y escritas por personas, a su vez resultan independientes del modelo de computador a utilizar (Lenguajes_de_Programacion, 2009).

- **Programación Web**

La programación web, parte de las siglas WWW, que significan World Wide Web o telaraña mundial. Para realizar una página con la programación web se deben tener claros tres conceptos fundamentales los cuales son, el URL (Uniform Resource Locators), es un sistema con el cual se localiza un recurso dentro

de la red, este recurso puede ser una página web, un servicio o cualquier otra cosa(Lenguajes_de_Programación, 2009).

Según la propia fuente, el siguiente concepto dentro de la programación web, es el protocolo encargado de llevar la información que contiene una página web por toda la red de Internet, como es el HTTP (Hypertext Transfer Protocol). Y por último el lenguaje necesario cuya funcionalidad es la de representar cualquier clase de información que se encuentre almacenada en una página web, este lenguaje es el HTML (Hypertext Markup Language).

Con la introducción del Internet y de la web en concreto, se han abierto infinidad de posibilidades en cuanto al acceso y uso de información desde cualquier parte del mundo. Con los avances en tecnología cada vez se demandan aplicaciones más rápidas, ligeras y robustas que permitan ser usadas sin importar el lugar u horario. En los últimos años la programación web ha incrementado aceleradamente su desarrollo, pues hoy en día ya se pueden hacer complejos sistemas que antes solo eran posibles hacer vía soluciones cliente/servidor como soluciones de escritorio.

Una aplicación web es un sistema informático que los usuarios utilizan accediendo a un servidor web a través de Internet o de una intranet. Las aplicaciones web son populares debido a la practicidad del navegador web como cliente ligero. La facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software en miles de potenciales clientes es otra razón de su popularidad. Aplicaciones como los web mails[20], wikis[21], weblogs[22], tiendas en línea son ejemplos bien conocidos de aplicaciones web (Gobierno_de_Canarias, 2010).

Ventajas:

- No requieren instalación, pues usan tecnología web, lo cual nos permite el aprovechamiento de todas las características del Internet.
- Son fáciles de usar (no requieren conocimientos avanzados de computación).
- Alta disponibilidad, ya que puede realizar consultas en cualquier parte del mundo donde tenga acceso a Internet y a cualquier hora.

[20] Un correo web es un cliente de correo electrónico, que provee una interfaz web por la que acceder al correo electrónico.
[21] Sitio web cuyas páginas web pueden ser editadas por múltiples voluntarios a través del navegador web. Los usuarios pueden crear, modificar o borrar un mismo texto que comparten.
[22] Sitio web periódicamente actualizado que recopila cronológicamente textos o artículos de uno o varios autores, apareciendo primero el más reciente, donde el autor conserva siempre la libertad de dejar publicado lo que crea pertinente.

- Datos centralizados y fácil integración de datos de múltiples fuentes.
- Menos requerimientos de memoria: Las demandas de memoria RAM (Random AccessMemory, Memoria de Acceso Aleatorio) por parte del usuario final son más razonables que en los programas instalados localmente.
- Compatibilidad multiplataforma ya que varias tecnologías incluyendo PHP, Java, Flash[23], ASP[24] y Ajax permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales.
- Múltiples usuarios concurrentes, las aplicaciones basadas en web pueden realmente ser utilizadas por múltiples usuarios al mismo tiempo. No hay más necesidad de compartir pantallas o enviar instantáneas cuando múltiples usuarios pueden ver e incluso editar el mismo documento de manera conjunta.

Desventajas:

- Acceso limitado, la necesidad de conexión permanente y rápida a Internet hacen que el acceso a estas aplicaciones no esté al alcance de todos. La interactividad no se produce en tiempo real, en las aplicaciones web cada acción del usuario conlleva un tiempo de espera excesivo hasta que se obtiene la reacción del sistema.
- Elementos de interacción muy limitados. En comparación con el software de escritorio, las posibilidades de interacción con el usuario que ofrecen las aplicaciones web (mediante formularios principalmente) son muy escasas, aunque se ha avanzado mucho con la introducción del Ajax en la construcción de páginas web.
- Diferencias de presentación entre plataformas y navegadores. La falta de estándares ampliamente soportados dificulta el desarrollo de las aplicaciones.
- La seguridad de tus datos, depende de la seguridad de la aplicación web y del servidor donde este alojado.

[23] Aplicación multimedia usada para aportar animación, vídeo e interactividad a las páginas Web. Adobe Flash es muy usado en anuncios y juegos Web.
[24] Tecnología de Microsoft del tipo "lado del servidor" para páginas web generadas dinámicamente.

- La estabilidad de la aplicación está sujeta al número de visitas en un mismo lapso de tiempo (sobre carga del servidor) (Montalvo, 2008).

- **Extend**

Ext es una biblioteca Java Script del lado del cliente, ligera y de alto rendimiento, compatible con la mayoría de navegadores que nos permite crear páginas e interfaces web dinámicas. Ext puede ejecutar en contra de cualquier plataforma de servidor que puede procesar las solicitudes POST y devolver datos estructurados. Las opciones más comunes son PHP, Java, .NET [25]y muchos más. Ext contiene algunos adaptadores como Ajax, animación, manipulación DOM[26], manejo de eventos, entre otros (Barrios, 2007).

Según la propia fuente, como marco de un cliente, Ext puede ejecutar en contra de cualquier plataforma de servidor que puede procesar las solicitudes POST y devolver datos estructurados. Las opciones más comunes son PHP, Java, .NET y muchos más. Hay marcos de lado servidor disponible para las plataformas más populares que harán mucho más fácil cuando se programa en el marco del modelo de aplicaciones AJAX que utiliza Ext., además es compatible con la mayoría de los navegadores web más populares como son Internet Explorer 6 +,Firefox 1.5, Safari 2 y Ópera 9.

Hemos incluido los tres archivos siguientes que Ext requiere para correr en nuestra página:

- **ext-all.css:** Un archivo stylesheet que controla las de baratijas de Ext. Este archivo siempre debe ser incluido tal cual, sin modificaciones. Cualquier cambio al CSS en este archivo rompería futuras mejoras.

- **ext-base.js:** Este archivo proporciona la funcionalidad principal de Ext, es el motor del mismo, es el archivo que nosotros cambiaríamos si quisiéramos usar otra biblioteca como jQuery[27], con la Ext.

- **ext-all-debug.js/ext-all.js:** Todas las baratijas se encuentran en este archivo. La versión de ajuste es usada para el desarrollo, y luego cambiado hacia fuera para la versión de no ajuste para la producción.

[25] Framework de Microsoft que hace un énfasis en la transparencia de redes, con independencia de plataforma de hardware y que permita un rápido desarrollo de aplicaciones.
[26] Interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.
[27] Biblioteca o framework de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web.

Una vez que estos archivos se disponen en el lugar, podemos comenzar a usar la biblioteca Ext.

- **C++**

C++ es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido. Posteriormente se añadieron facilidades programación genérica, que se sumó a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje multiparadigma (Programación, 2009).

Según la propia fuente, actualmente existe un estándar, denominado ISO C++, al que se han adherido la mayoría de los fabricantes de compiladores más modernos. Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales. C++ permite trabajar tanto a alto como a bajo nivel.

A partir de dichas razones y por estudios realizados sobre los diferentes lenguajes de programación posibles a utilizar, en el presente trabajo se seleccionó para realizar la herramienta web el lenguaje de programación C++.

- **Wt** (WT, 2009)

Wt (pronunciado “witti” o “Web Toolkit”) es una biblioteca en C++ y servidor de aplicaciones para desarrollar y desplegar aplicaciones web interactivas. Posee marcos a base de página (basado en PHP, JSP[28]/JSF[29], el Rubí sobre Carriles, etc.) que no hacen la abstracción de las tecnologías subyacentes (HTML/XHTML, JavaScript, CSS, AJAX, Formas, DHTML, SVG/VML/Canvas[30]). La estructura del uso de la aplicación sigue sobre todo el paradigma HTML de hoy en día, esto quiere decir que no sólo se tiene que poner en práctica un regulador para indicar como un usuario se mueve de una página a otra, pero

[28] **(Java Server Page)** Página de Servidor Java. Se refiere a un tipo especial de páginas HTML, en las cuales se insertan pequeños programas que corren sobre Internet (comúnmente denominados scripts), se procesan en línea para finalmente desplegar un resultado final al usuario en forma de HTML.
[29] Framework para construir interfaces de usuario en aplicaciones web.
[30] Especificación, lenguaje XML de programación abierto y etiqueta o elemento en HTML, respectivamente, que permiten la generación de gráficos.

usando técnicas de AJAX avanzadas se puede diseñar y mantener la comunicación de cliente-servidor, además es un importante adelanto en la creación de aplicaciones dinámicas de JavaScript.

El desarrollo de aplicaciones con Wt es escrito en solo un lenguaje de compilación (C++) del cual la biblioteca genera HTML/XHTML necesario, Java script, CGI[31], SVG/VML/Canvas y el código de AYAX. La filosofía de trabajo de Wt es bastante sencilla, básicamente se crea el código en Java usando cualquier entorno de desarrollo (IDE) de Java y el compilador lo traduce a HTML y Java Script.

El API es de widgets céntricos y ofrece una completa abstracción de los detalles específicos de la aplicación web.

Beneficios del uso de Wt:

- Desarrollo tanto de aplicaciones web como de aplicaciones de escritorio.
- Proporciona un juego extenso de widgets, que trabajan independientemente de la disponibilidad Java Script.
- Especificación tanto para cliente como validación del lado del servidor y manejo de acontecimientos.
- Genera estándares HTML y código XHTML.
- Evita problemas de seguridad comunes ya que tiene el control completo de la presentación y activamente se elimina filtrando etiquetas activas y atributos, ni lógica de negocio de exposiciones, que se queda en el servidor.
- API simple para acontecimientos iniciados por el servidor.

Desventajas del uso de Wt:

- Es difícil añadir **Wt** sobre código ya existente.
- Carece de soporte a Java 5.
- Carece de widgets avanzados (aunque sí Ext.).

[31] **(Common Gateway Interface)** es de las primeras formas de programación web dinámica. Es utilizado comúnmente para contadores, bases de datos, motores de búsqueda, formularios, generadores de email automático, rotadores y mapas de imágenes, juegos en línea y otros.

Figura 4 Modelo de aplicación Wt

El paradigma seguido de estos marcos web es ilustrado en la Figura 5. En cada paso, el navegador web solicita una nueva página al servidor web, y puede enviar dentro de la petición un número de valores. Al final del servidor, la aplicación web procesa la petición, identifica la sesión, y realiza la lógica de negocio. Finalmente, la aplicación genera la página de respuesta. La página de respuesta puede contener no sólo HTML, sino también Java Script para realzar la interactividad de la aplicación web.

La figura 4a el modelo de página dinámico tradicionalmente usado por marcos de aplicaciones web, contrastó con 4b el modelo conductor de acontecimiento tradicionalmente usado por Wt.

1.2.5 Entorno de desarrollo integrado IDE

Un entorno de desarrollo integrado (IDE) es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien, poder utilizarse para varios. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un

constructor de interfaz gráfica GUI[32]. Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes (Babylon, 2007).

- **Eclipse** (Eclipse, 2010)

Eclipse es principalmente una plataforma de programación, usada para crear entornos integrados de desarrollo (del Inglés IDE). Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto.

Pese a que Eclipse está escrito en su mayor parte en Java, se ejecute sobre máquina virtual de Java, es neutral y a su vez, adaptable a cualquier tipo de lenguaje de programación, por ejemplo C/C++, Cobol, C#, XML, etc. La característica clave de Eclipse es la extensibilidad y que presenta además una gran estructura formada por muchos plug-in que van conformando la funcionalidad final.

El proyecto Eclipse PHP Development Tools (PDT) es un conjunto de herramientas que mejora la productividad de los desarrolladores de PHP. Esta es la primera vez que se trabaja en función de esta gran comunidad por parte de Eclipse y se piensa continuar su desarrollo para brindar una herramienta completamente funcional que facilite la extensibilidad de los proyectos en PHP y supere las herramientas de desarrollo actuales para PHP.

El Eclipse maneja los recursos del usuario, organizados en uno o más proyectos. Cada proyecto corresponde a un directorio en el directorio de trabajo de eclipse y contienen archivos y carpetas. Se configura en perspectivas que configuran los editores de código y las vistas. A diferencia de muchas aplicaciones escritas en Java, Eclipse tiene el aspecto y se comporta como una aplicación nativa. No está programada en Swing[33], sino en SWT[34] y Jface[35], que emula los gráficos nativos de cada sistema operativo. Este plug-in facilita el uso de un sistema de control de versiones para manejar los recursos en un proyecto del usuario y define el proceso necesario para guardar y recuperar de un repositorio.

[32] **(Graphical User Interfaces)** Componente de una aplicación informática que el usuario visualiza y a través de la cual opera con ella. Está formada por ventanas, botones, menús e iconos, entre otros elementos.
[33] Biblioteca gráfica para Java. Incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, desplegables y tablas.
[34] **(Standard Widget Toolkit)** es un conjunto de componentes para construir interfaces gráficas en Java, (widgets) desarrollados por el proyecto Eclipse.
[35] Conjunto de widgets para realizar interfaces de usuario construido sobre SWT. Fue desarrollado por IBM para facilitar la construcción del entorno de desarrollo Eclipse, pero su uso no está limitado a éste.

Hoy en día los entornos de desarrollo proporcionan un marco de trabajo para la mayoría de los lenguajes de programación existentes en el mercado (por ejemplo C, C++, C#, Java, Python y Visual Basic entre otros). Además es posible que un mismo entorno de desarrollo tenga la posibilidad de utilizar varios lenguajes de programación, como es el caso de Eclipse (Entornos_de_Desarrollo_Integrado, 2008).

Debido a todas las ventajas que presenta y las facilidades que tiene para la programación web se ha decidido utilizar como lenguaje de programación para el desarrollo de nuestra aplicación por cuestiones de preferencias Eclipse.

1.2.6 Herramienta para Pruebas de Software automatizadas

OpenLoad es una herramienta para pruebas de carga automatizadas que surgió de la necesidad de automatizar pruebas de carga en aplicaciones web. Es la primera solución rápida de optimización de rendimiento basada en navegador, es fácil de usar y su objetivo consiste en proporcionar respuestas de rendimiento en tiempo real simulando un número de usuarios conectados haciendo solicitudes a una misma aplicación. Esto es muy útil porque permite evaluar el impacto de inmediato antes que la aplicación sea puesta a disposición de los usuarios.

Para el correcto funcionamiento de la herramienta es necesario especificar los siguientes aspectos:

Comando openload [opciones] [dirección URL] [número de usuarios a simular]

Dirección URL: URL de la página que se desea probar.

Número de usuarios a simular: # de clientes a simular conectados concurrentemente a la página. Por defecto la herramienta trae 5

- **Ventajas del uso de OpenLoad**

El verificador de carga OpenLoad brinda facilidad de uso, exactitud y escalabilidad de un solo desarrollo integrado y ambiente de la prueba. Este minimiza substancialmente el tiempo requerido para las pruebas de carga y pone a punto la eficiencia de la web basado en aplicaciones y servicios por simplificación de procesos, verificando la conducta funcional esperada a través de la regresión automatizada, probando y apuntando con precisión la eficiencia de los cuellos de botella dentro de las aplicaciones web; ofreciendo al desarrollo de aplicaciones y pruebas de equipos, la habilidad de realizar rápida y productivamente la optimización de aplicaciones web.

1. Tiene poderosas capacidades de pruebas de regresión incorporadas que le permiten comprobar

fácilmente los resultados esperados con carga para un solo usuario y más.

2. Permite variar la entrada del número de usuarios y las opciones para el trabajo con la herramienta.
3. Posibilita hacer las pruebas desde diferentes puntos de la red.
4. Ofrece facilidad de uso sin precedentes, la exactitud y la escalabilidad de un desarrollo integrado y único entorno de prueba.

Conclusiones parciales del capítulo 1

Después de hacer un análisis del estado del arte de las metodologías y tecnologías necesarias y actuales en el mundo se decide utilizar GNU/Linux como Sistema Operativo en su distribución Debian, como lenguaje de programación C++ vinculado con la biblioteca Wt que incluye diversos elementos de Extend y como IDE por todas las ventajas que presenta el Eclipse. También se ha definido RUP como la metodología de desarrollo de software a utilizar y el Visual Paradigm como herramienta CASE en que se realizará la modelación de la aplicación aprovechando las posibilidades que brinda UML.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

En este capítulo se realizará la descripción de la aplicación web para administradores de seguridad a desarrollar para lograr una mayor claridad y comprensión por parte de los desarrolladores. Los puntos que se abordarán son los siguientes: descripción del modelo de dominio, especificación de los requisitos funcionales y no funcionales, representación de los actores y casos de uso del sistema a través de un diagrama de casos de uso del sistema, así como la descripción textual de cada uno de ellos.

Modelo de Dominio

Un modelo del dominio captura los tipos más importantes de objetos en el contexto del sistema representando los objetos existentes o eventos que sucedan en el entorno en el que se trabaja (Jacobson, et al., 2004).

Según la propia fuente, muchos de los objetos del dominio pueden obtenerse de una especificación de requisitos o mediante la entrevista con los expertos del dominio. Entre los objetivos fundamentales de este modelo se encuentra la comprensión y descripción de las clases más importantes dentro del contexto del sistema, así como la ayuda para los usuarios, clientes y desarrolladores a la utilización de un vocabulario común. Cuando no existen procesos definidos RUP propone realizar un modelo del dominio, que es un subconjunto del modelo de negocio.

2.1 Descripción del Modelo de Dominio

El Modelo del Dominio se describe mediante diagramas de UML, especialmente mediante diagramas de clases, estos muestran a los clientes, usuarios, revisores y a otros desarrolladores las clases del dominio y cómo se relacionan mediante asociaciones. En la figura 6 se representa el modelo de dominio realizado.

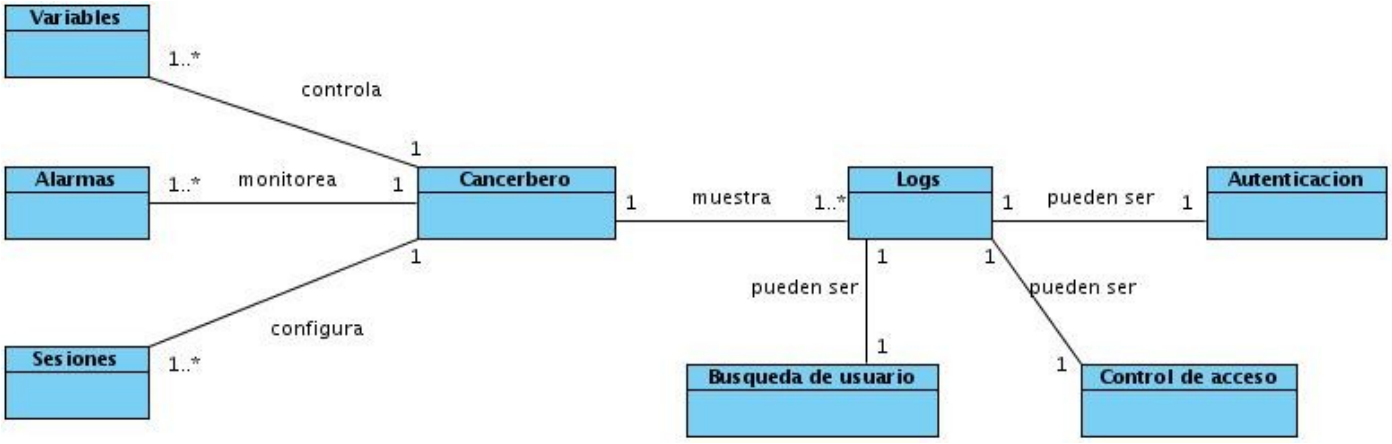


Figura 5 Modelo de Dominio

2.1.1 Clases conceptuales del dominio

1. **Cancerbero:** Herramienta web para administradores de seguridad.
2. **Variables:** Datos que necesita el servidor para el sistema de detección de intrusos.
3. **Alarmas:** Dispositivo que avisa de alguna particularidad fuera de lo normal en el sistema.
4. **Sesiones:** Sesiones de trabajo que se crean a los usuarios que entran al sistema.
5. **Logs:** Archivo que registra movimientos y actividades del programa en ejecución**.**

    5.1 **Autenticación:** Procedimiento de comprobación de una entidad del sistema.

    5.2 **Control de acceso:** Controla el acceso a la red con algunas políticas implementadas.

    5.3 **Búsqueda de usuarios:** Búsqueda de usuarios que se encuentren conectados al sistema.

2.2 Definición de los Requisitos Funcionales

Los Requisitos Funcionales se definen como las capacidades o condiciones que el sistema debe cumplir. Estos se mantienen invariables sin importar con que propiedades o cualidades se relacionen. La herramienta web a desarrollar a través de la presente investigación debe cumplir con los requisitos funcionales siguientes:

RF 1 Autenticar

RF 2 Cambiar contraseña

RF 3 Mostrar logs

RF 4 Administrar sesiones

RF 4.1 Eliminar sesiones

RF 4.2 Mostrar detalle de usuario

RF 5 Administrar alarmas

RF 5.1 Eliminar alarma

RF 6 Administrar variables

2.3 Definición de los Requisitos no Funcionales

- **Usabilidad**

El sistema podrá ser usado por cualquier tipo de persona que posea conocimientos básicos en el manejo de la computadora y los sistemas operativos GNU/Linux y Windows.

- **Apariencia e interfaz externa**

La herramienta tendrá una interfaz amigable, interactiva, de uso fácil y sencillo.

- **Restricciones de diseño e implementación**

- Será una aplicación web.
- El análisis y diseño de la aplicación será basado en la Metodología RUP con el uso del lenguaje de modelado UML.
- Se utilizara la programación web.
- Se usará como herramienta CASE el Visual Paradigm para el modelado de los artefactos que se generan en cada uno de los flujos de trabajo definidos por RUP.
- Se utilizarán el Eclipse como entorno de desarrollo garantizando la calidad de todo el ciclo de

desarrollo del producto.

- **Soporte**

Se debe ofrecer servicios de mantenimiento y actualización.

- **Portabilidad**

Será un sistema multiplataforma, se podrá disponer en cualquier sistema operativo que disponga de navegadores web.

- **Software**

Para el uso de esta herramienta se debe tener instalado algún navegador de internet.

- **Hardware**

Para el funcionamiento de la herramienta se requieren máquinas con los siguientes requisitos:

- Procesador Pentium 3[36] o superior.
- 256 Mb de RAM.

- **Eficiencia**

Se recomienda una computadora como mínimo con 256 MB de RAM.

- **Rendimiento**

La herramienta está diseñada sobre una aplicación web y como característica relevante debe tener la rapidez de respuesta entre las operaciones que se realicen en la aplicación.

- **Seguridad**

La herramienta dispondrá de una contraseña de seguridad para tener acceso al sistema que solo conocerá el administrador de seguridad que trabaje con ella.

[36] Microprocesador de arquitecturai686 fabricado por Intel, el cual es una modificación del Pentium Pro.

➢ **Requerimiento de Confiabilidad**

- Se debe garantizar que el sistema esté disponible las 24 horas del día.
- Las actividades de mantenimiento, supervisión y edición del sistema no deben interferir en el correcto desempeño del resto del sistema, ni afectar la ejecución de la aplicación.

2.4 Modelo de Casos de Usos del Sistema

El Modelo de Casos de Uso describe lo que hace el sistema para cada tipo de usuario representados por uno a varios actores, los cuales representan a individuos o sistemas externos que colaboran con este (Jacobson, y otros, 2004). Para el desarrollo de esta herramienta se ha identificado el siguiente actor.

Actor(s)	Descripción
Administrador	Es la persona que interactúa con el sistema, es la encargada de llevar la administración de la aplicación en el centro donde se encuentre en funcionamiento, es el único autorizado a realizar todas las operaciones que estén en la herramienta llevando a cabo un mejor control de los usuarios que están conectados al sistema.

Según la propia fuente, cada forma en que los actores usan el sistema se representa con un caso de uso. Los casos son "fragmentos" de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. De manera más precisa, "*un caso de uso especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores, incluyendo alternativas dentro de la secuencia*".

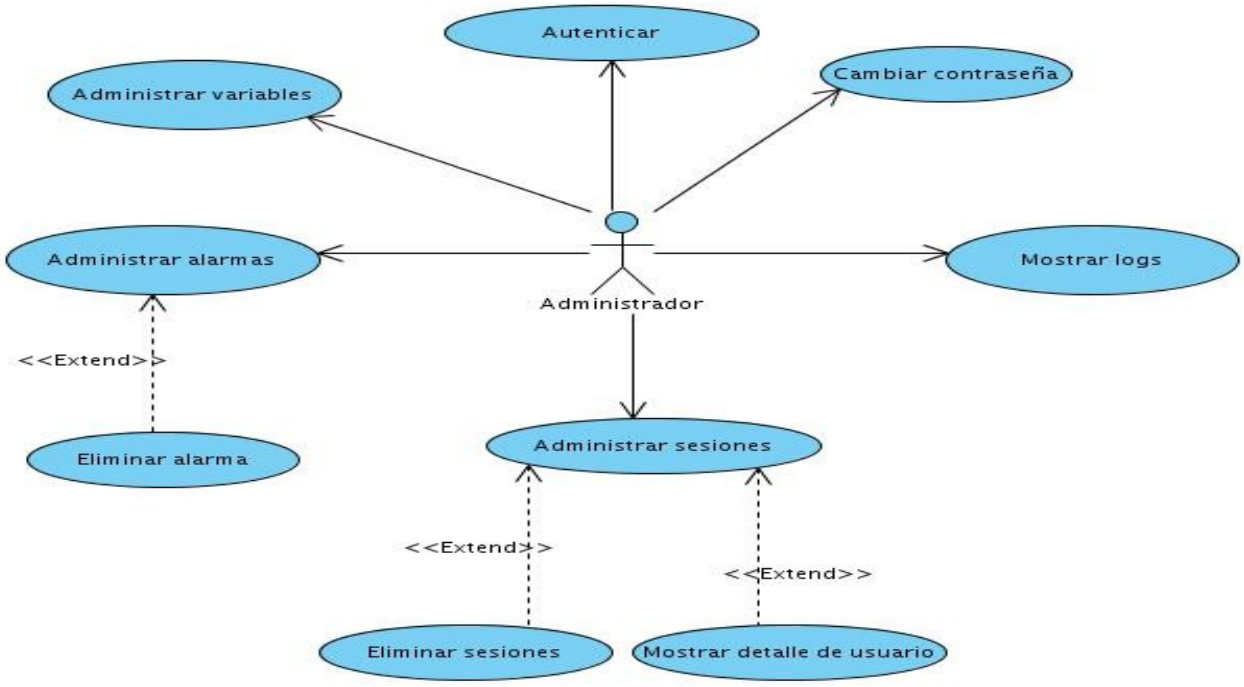


Figura 6 Diagrama de Casos de Uso del Sistema

2.4.1 Patrones de Caso de Uso del Sistema

Los patrones de casos de uso permiten con mayor facilidad reflejar los requisitos reales de un sistema, así como un fácil trabajo y mantenimiento del mismo. Se presentan a modo de herramientas que permiten resolver los problemas que se les planteen a los desarrolladores de una forma ágil y sistemática (Jacobson, et al., 2004).

El patrón utilizado en el desarrollo de la herramienta web para administradores de seguridad es: Concordancia (Commonality)

- **Concordancia (Commonality)**

Extrae una subsecuencia de acciones que aparecen en diferentes lugares del flujo de casos de uso y es expresado por separado (UCI, 2008).

- ***Adición***

En el caso de este patrón alternativo, la subsecuencia común de casos de uso, extiende los casos de uso Eliminar Sesiones y Mostrar Detalles de Usuario del caso de uso Administrar sesiones

compartiendo la subsecuencia de acciones. Los otros casos de uso modelan el flujo que será expandido con la subsecuencia.

2.5 Descripción textual de los casos de uso del Sistema

RF 1 Autenticar

Caso de Uso:	**Autenticar**
Actores:	Administrador (inicia)
Resumen:	Este caso de uso le permite al administrador autenticarse mediante su usuario y contraseña.
Referencias	RF1
Prioridad	crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. Inicia el caso de uso cuando el usuario decide ejecutar la aplicación.	2. El sistema solicita al usuario su autenticación.
3. El administrador suministra su nombre de usuario y la contraseña.	4. El sistema realiza la autenticación con los datos suministrados por el usuario.
Flujo Alterno 1"Autenticación fallida"	
Acción del Actor	**Respuesta del Sistema**
	1.1 Si la autenticación no fue satisfactoria el sistema muestra información al usuario sobre las posibles causas. 1.2 El sistema retorna al paso 2.

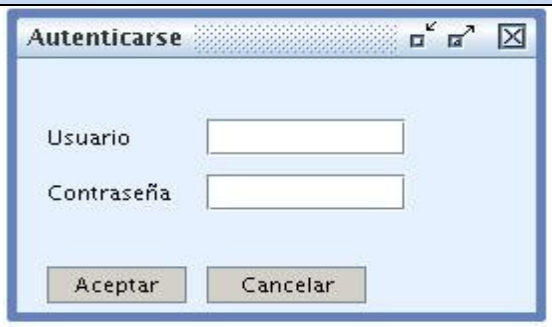
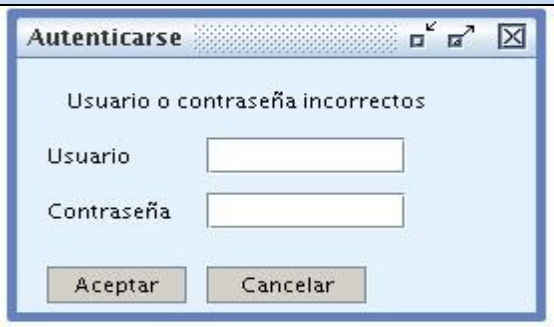
Pos condiciones	Se autenticó el usuario.
Prototipo de interfaz de usuario	
 	

Tabla 1 Descripción CUS Autenticar

RF 2 Cambiar contraseña

Caso de Uso:	**Cambiar contraseña**
Actores:	Administrador (inicia)
Resumen:	Este caso de uso le permite al usuario cambiar su contraseña.
Referencias	RF2
Precondiciones	1. El administrador debe estar autenticado en el sistema.
Prioridad	crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso comienza cuando el usuario selecciona la opción “Cambiar contraseña”.	2. El sistema solicita al usuario su contraseña actual.
3. El usuario suministra su contraseña actual.	4. El sistema realiza la verificación de la contraseña actual.

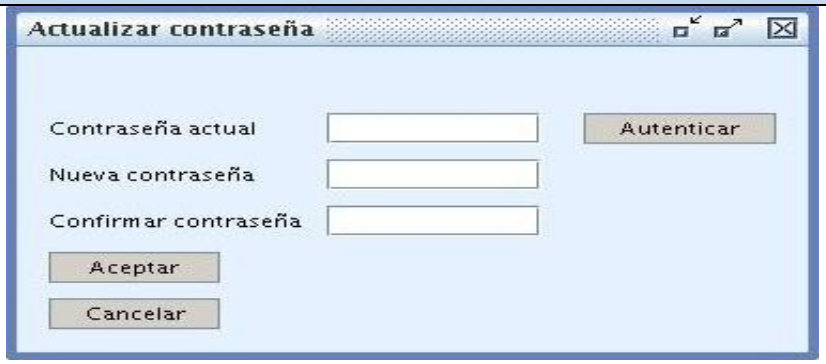
	5. El sistema solicita al usuario su contraseña nueva y la confirmación de la misma.
6. El usuario suministra la nueva contraseña.	7. El sistema realiza la actualización de la contraseña.
	Flujo Alterno 2.1"Comprobación de contraseña fallida"
	2.1.1 Si la verificación resulta fallida el sistema retorna al paso 4.
Pos condiciones	Se cambió la contraseña.
	Prototipo de interfaz de usuario
	

Tabla 2 Descripción CUS Cambiar Contraseña

RF 3 Mostrar logs

Caso de Uso:	**Mostrar logs**
Actores:	Administrador(inicia)
Resumen:	Este caso de uso permite realizar una búsqueda avanzada de los logs de seguridad.
Referencias	RF3
Precondiciones	1. El administrador debe estar autenticado en el sistema.
Prioridad	crítico

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción “Mostrar Logs”.	2. El sistema muestra las opciones: 2.1. Autenticación (ver Sesión 1 “Autenticación”) 2.2. Control de acceso (ver Sesión 2 “Control de acceso”) 2.3. Buscar usuarios (ver Sesión 3 “Buscar usuarios”)
3. El usuario escoge una de las opciones mostradas por el sistema.	4. El sistema muestra los campos según la opción escogida.
Sesión 1 “Autenticación”	
	2.1.1 El sistema muestra los campos nick, nombre, id de usuario, descripción, nodo, fecha inicio y fecha fin para filtrar la búsqueda de los logs de autenticación.
2.1.2 El usuario completa los campos que le interese filtrar. 2.1.3 El usuario selecciona la opción “Buscar”.	
	2.1.4 El sistema muestra los resultados de la búsqueda realizada.
Sesión 2 “Control de acceso”	
	2.2.1 El sistema muestra los campos nick, nombre, id de usuario, descripción, id del recurso, fecha inicio y fecha fin para filtrar la búsqueda de los logs de autenticación.
2.2.2 El usuario completa los campos que le interese filtrar. 2.2.3 El usuario selecciona la opción	

"Buscar".	
	2.2.4 El sistema muestra los resultados de la búsqueda realizada.
Sesión 3 "Buscar usuarios"	
	2.3.1 El sistema muestra los campos nick, nombre, id de usuario, perfil, id perfil, id recurso y supervisor para filtrar la búsqueda de los logs de autenticación.
2.3.2 El usuario completa los campos que le interese filtrar. 2.3.3 El usuario selecciona la opción "Buscar".	
	2.3.4 El sistema muestra los resultados de la búsqueda realizada.
2.3.5 El usuario selecciona la opción "Mostrar detalles de usuario"(Ver CU-Mostrar Detalles de Usuario)	
Prototipo de interfaz de usuario	
	

Tabla 3 Descripción CUS Mostrar Logs

RF 4 Administrar sesiones

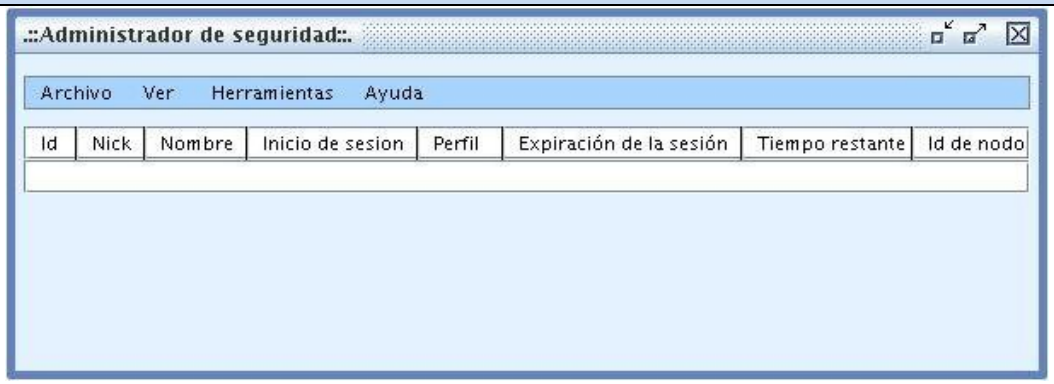
Caso de Uso:	**Administrar sesiones**
Actores:	Administrador (inicia)
Resumen:	Este caso de uso permite administrar las sesiones que se encuentran abiertas.
Referencias	RF4, RF4.1, RF4.2
Precondiciones	1. El administrador debe estar autenticado en el sistema.
Prioridad	crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso comienza cuando el usuario selecciona la opción "Administrar Sesiones".	2. El sistema muestra el id, usuario y tiempo que lleva abierta la sesión.
Pos condiciones	Se mostraron los datos de las sesiones que están abiertas.
Prototipo de interfaz de usuario	
	

Tabla 4 Descripción CUS Administrar Sesiones

RF 4.1 Eliminar sesiones

Caso de Uso:	**Eliminar sesiones**
Actores:	Administrador(inicia)
Resumen:	Este caso de uso permite eliminar las sesiones que se seleccionen.
Referencias	RF4, RF4.1, RF4.2
Precondiciones	1. El administrador debe estar autenticado en el sistema.
Prioridad	crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
4.1.1 El usuario selecciona la sesión a eliminar dándole doble click encima. 4.1.2 El usuario selecciona la opción “Eliminar Sesiones”.	
	4.1.3 El sistema elimina la sesión seleccionada.
Pos condiciones	Se eliminó la sesión abierta.
Prototipo de interfaz de usuario	

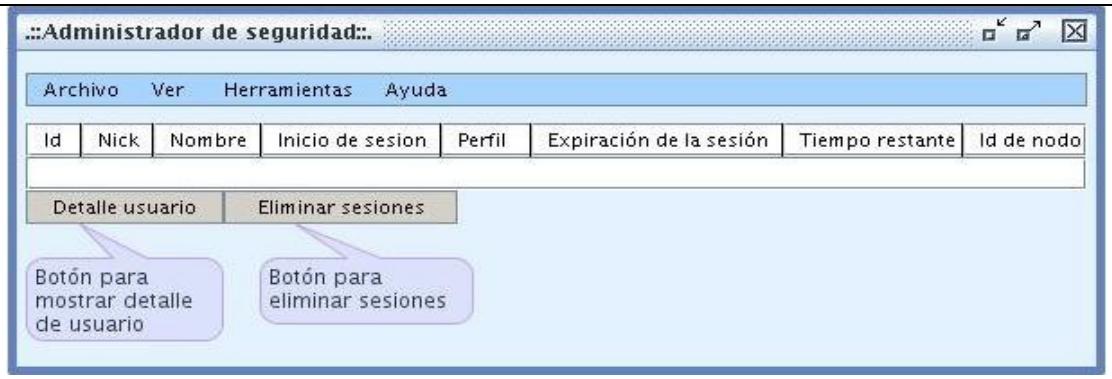


Tabla 5 Descripción CUS Eliminar Sesiones

RF 4.2 Mostrar detalle de usuario

Caso de Uso:	**Mostrar detalle de usuario**
Actores:	Administrador (inicia)
Resumen:	Este caso de uso permite mostrar los detalles del usuario.
Referencias	RF4, RF4.1, RF4.2
Precondiciones	1. El administrador debe estar autenticado en el sistema. 2. Debe haber un usuario seleccionado.
Prioridad	crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
4.2.1 El usuario selecciona la sesión de la que se quiere saber los detalles dándole doble click encima. 4.2.2 El usuario selecciona la opción “Mostrar detalles de	

usuario".	
	4.2.3 El sistema muestra los detalles del usuario de la sesión seleccionada, id, perfil, responsabilidad, usuario, teléfono, supervisor, correo y dirección.
Pos condiciones	Se mostraron los detalles del usuario seleccionado.
Prototipo de interfaz de usuario	
	

Tabla 6 Descripción CUS Mostrar Detalle de Usuario

RF 5 Administrar alarmas

Caso de Uso:	**Administrar alarmas**
Actores:	Administrador (inicia)
Resumen:	Este caso de uso permite administrar las alarmas que se activaron.
Referencias	RF5, RF5.1
Precondiciones	1. El administrador debe estar autenticado en el sistema.
Prioridad	crítico
Flujo Normal de Eventos	

Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el usuario selecciona la opción "Administrar Alarmas".	2. El sistema muestra el id de la alarma, descripción, id del usuario, usuario, id del recurso, fecha, incidencia, estado y tipo de alarma.
Pos condiciones	Se administran las alarmas que se activaron.
Prototipo de interfaz de usuario	
	

Tabla 7 Descripción CUS Administrar Alarmas

RF 5.1 Eliminar alarma

Caso de Uso:	**Eliminar alarma**
Actores:	Administrador(inicia)
Resumen:	Este caso de uso permite eliminar las alarmas que se seleccionen.
Referencias	RF5, RF 5.1
Precondiciones	1. El administrador debe estar autenticado en el sistema.
Prioridad	crítico
Flujo Normal de Eventos	

Acción del Actor	Respuesta del Sistema
5.1.1 El usuario selecciona la alarma a eliminar dándole doble click encima. 5.1.2 El usuario selecciona la opción “Eliminar Alarma”.	
	5.1.3 El sistema elimina la alarma seleccionada.
Pos condiciones	Se eliminó la alarma.
Prototipo de interfaz de usuario	
	

Tabla 8 Descripción CUS Eliminar Alarma

RF 6 Administrar variables

Caso de Uso:	**Administrar variables**
Actores:	Administrador (inicia)
Resumen:	Este caso de uso permite configurar las variables.
Referencias	RF6
Precondiciones	1. El administrador debe estar autenticado en el sistema.

Prioridad	crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el usuario selecciona la opción “Preferencias”.	2. El sistema muestra la vista para configurar las variables.
3. El usuario selecciona el tipo de variable que desea configurar.	4. El sistema muestra las variables relacionadas con el tipo de configuración seleccionada.
Pos condiciones	Se controlaron las variables deseadas.
Prototipo de interfaz de usuario	

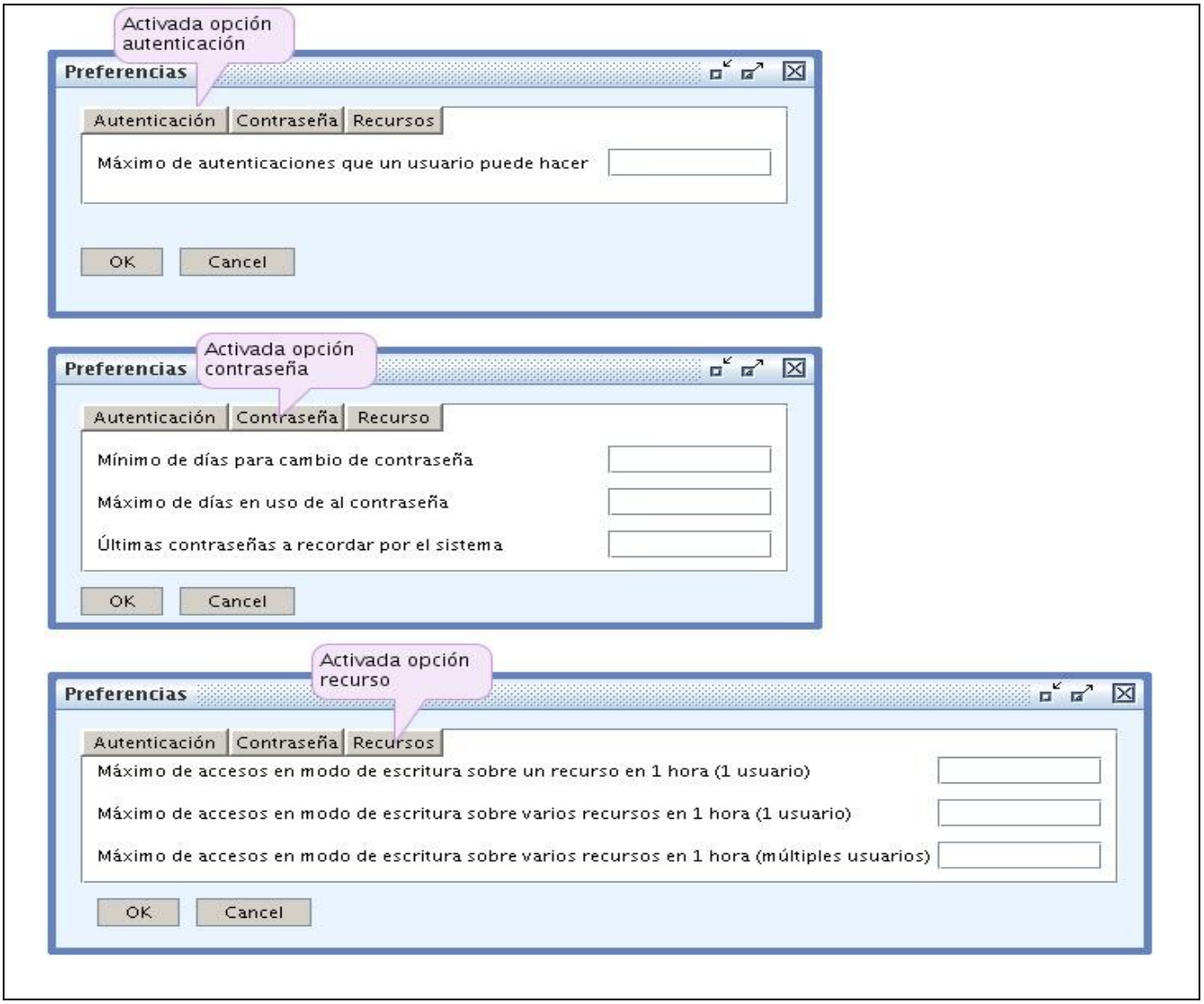


Tabla 9 Descripción CUS Administrar Variables

Conclusiones parciales del capítulo 2

En el desarrollo de este capítulo se le dio cumplimiento a la segunda tarea planteada, identificándose 9 funcionalidades necesarias para el diseño de la aplicación web para administradores de seguridad. Estas se encuentran agrupadas en 6 casos de uso arquitectónicamente significativos que identifican las funcionalidades.

CAPÍTULO 3: DISEÑO DEL SISTEMA

El contenido que se aborda en este capítulo estará relacionado con el diseño del sistema. El mismo incluye la descripción de los estilos arquitectónicos y patrones de diseño seleccionados, además de explicar cómo se adaptan y se representan cada uno de ellos en el diseño del sistema. Por otra parte, se realizarán todos lo diagramas de clases y de interacción que den soporte a cada uno de los requisitos del sistema, incluyendo los no funcionales.

3.1 Descripción del estilo arquitectónico utilizado

"Una arquitectura es el conjunto de decisiones significativas sobre la organización del sistema de software, la selección de los elementos estructurales y sus interfaces, con los que se compone el sistema, junto con su comportamiento tal como se especifica en las colaboraciones entre esos elementos, la composición de esos elementos estructurales y de comportamiento en subsistemas progresivamente más amplios, y el estilo de arquitectura que guía esta organización –estos elementos y sus interfaces, sus colaboraciones, y su composición." (Booch, y otros, 1999)

- **Arquitectura Cliente - Servidor**

El objetivo principal de la Arquitectura del Software es aportar elementos que ayuden a la toma de decisiones y, al mismo tiempo, proporcionar conceptos y un lenguaje común que permitan la comunicación entre los equipos que participen en un proyecto. Para conseguirlo, la Arquitectura del Software construye abstracciones, materializándolas en forma de diagramas (blueprints) comentados (Casanovas, 2004).

Según la propia fuente, en una aplicación web la arquitectura es sencilla, esta es conocida como la arquitectura cliente servidor donde sus componentes principales son el Servidor web, una Red y un Navegador o cliente además de la aplicación en el Servidor, que es la que permite al sistema manejar lógica de negocio y tener un estado. En este modelo las aplicaciones se dividen de forma que el servidor contiene la parte que debe ser compartida por varios usuarios, y en el cliente permanece sólo lo particular de cada usuario.

Siempre que un cliente requiere un servicio lo solicita al servidor correspondiente y éste le responde proporcionándolo. Normalmente, pero no necesariamente, el cliente y el servidor están ubicados en distintos procesadores. Los clientes se suelen situar en ordenadores personales y/o estaciones de trabajo

y los servidores en procesadores departamentales o de grupo. Para la comunicación de los procesos con la red se emplea un tipo de equipo lógico denominado middleware que controla las conversaciones. Su función es independizar ambos procesos (cliente y servidor). La interfaz que presenta es la estándar de los servicios de red que hace que los procesos "piensen" en todo momento que se están comunicando con una red, según Casanovas (2004).

En la herramienta web para administradores de seguridad se implementará la parte de la vista del cliente, la cual tomará los datos del servidor de seguridad.

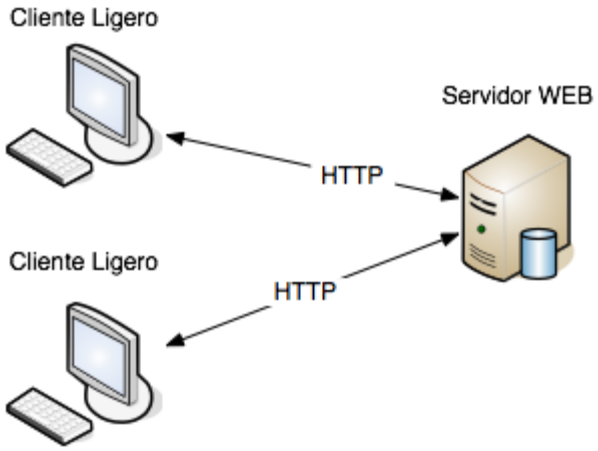


Figura 7 Arquitectura Cliente-Servidor

- **Arquitectura Modelo -Vista -Controlador**

El patrón de diseño Modelo-Vista-Controlador (MVC) se utiliza para el diseño de aplicaciones que presenten interfaces sofisticadas. Básicamente consiste en estructurar el programa en tres componentes que se relacionan entre ellos de una manera muy concreta (Alvarez, 2008):

- El **modelo** es el núcleo de la aplicación. Su única responsabilidad respecto a la vista o al controlador es avisar de cambios importantes mediante eventos. Además, debe llevar un registro de las vistas y controladores del sistema, para que sea capaz de notificar a estos sobre los cambios que pueda producir un agente externo a los datos.
- La **vista** es la encargada de mostrar al usuario el progreso del funcionamiento del modelo. Cuando el modelo cambia la vista solicita el nuevo estado y se actualiza. Debe tener además un registro del controlador que por lo general es instanciado por esta fase. Presenta el modelo en un formato que pueda interactuar con el usuario, por lo general es la interfaz del usuario.

- El **controlador** es una especie de capa intermedia que conecta la vista con el modelo. De acuerdo con los mensajes recibidos de la vista (la acción del usuario), modifica a ésta, y se pone en contacto con el modelo.

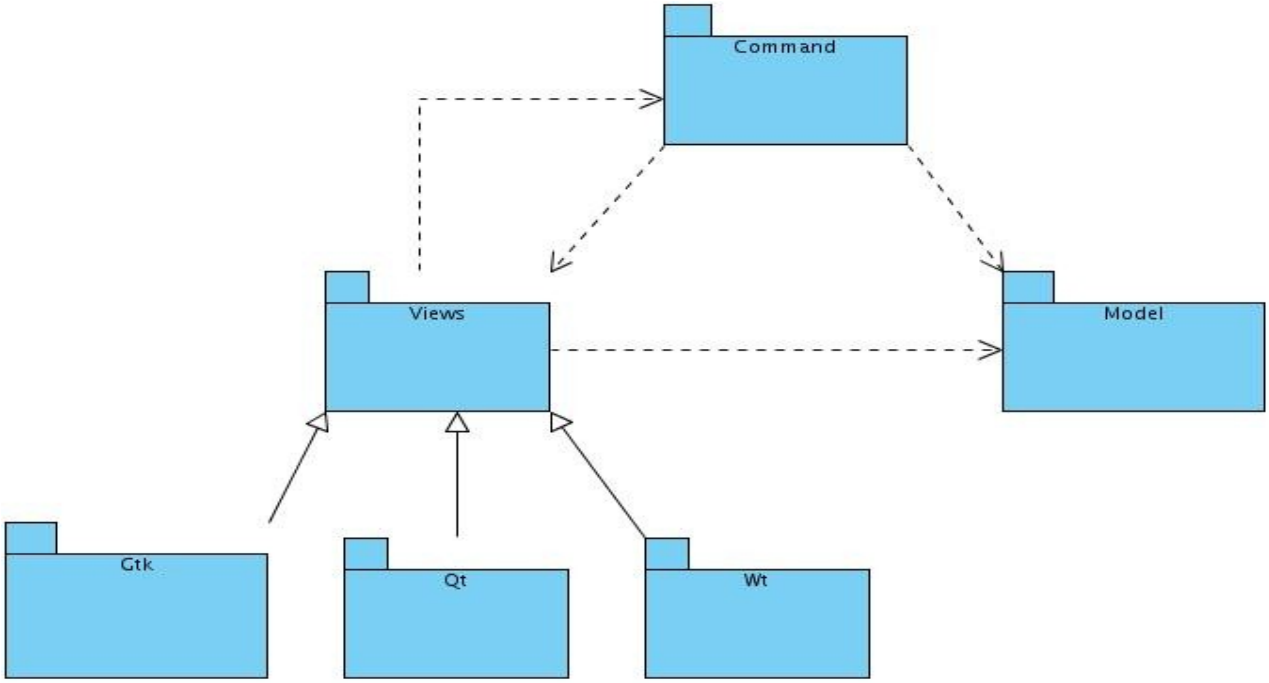


Figura 8 Estructura del patrón Modelo-Vista-Controlador

La experiencia en el desarrollo de aplicaciones complejas ha demostrado que la lógica de una interfaz de usuario cambia con más frecuencia que los almacenes de datos y la lógica del negocio. Si se realizara un diseño donde se mezclen los componentes de la interfaz y del negocio, puede traer como consecuencia que al modificar la interfaz también se debe modificar los componentes del negocio haciendo el trabajo más complejo y con un mayor riesgo de que ocurran errores.

El uso de este patrón posibilita realizar el diseño de una aplicación donde se desacople la vista del modelo, con la finalidad de mejorar la reusabilidad. De esta forma las modificaciones en las vistas impactan en menor medida en la lógica de negocio o de datos.

La estructura de la aplicación de escritorio cuenta con el patrón Modelo-Vista-Controlador mostrado en la figura 8. El trabajo con la biblioteca Wt, nos permite reutilizar esta estructura ya implementada y solo tener que agregar el paquete de la vista para la aplicación web, el cual es compatible con el resto del sistema.

3.2 Descripción de los patrones de diseño utilizados

El diseño, es un modelo del sistema, donde el mismo se debe describir con suficiente detalle como para ser implementado. Existen muchos esquemas y estructuras de solución que se usan repetidas veces en función del contexto del problema, la cuales se han convertido en patrones.

"Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, para describir después el núcleo de la solución a ese problema, de tal manera que esa solución pueda ser usada más de un millón de veces sin hacerlo ni siquiera dos veces de la misma forma(Lago, 2007)*."*

Según la propia fuente, el uso de patrones de diseño ayuda a la obtención de una aplicación de mayor extensibilidad y reutilización, estos gozan de algunas cualidades las cuales se describen a continuación.

- **Encapsulamiento y abstracción:** Cada patrón encapsula un problema bien definido y su solución en un dominio particular.
- **Extensión y variabilidad:** Cada patrón debería ser abierto por extensión o parametrización por otros patrones, de tal forma que pueden aplicarse juntos para solución un gran problema.
- **Generatividad y composición:** Cada patrón una vez aplicado genera un contexto resultante, el que concuerda con el contexto inicial de uno o más de uno de los patrones del catálogo.

Para realizar el diseño de la aplicación se han utilizado algunos de estos patrones existentes, los cuales se relacionan a continuación.

3.2.1 Patrón Abstract Factory

Proporciona una interfaz para crear familias de objetos relacionadas o dependientes, sin especificar su clase concreta (Gamma, y otros, 1995).

Según la propia fuente, el uso de este patrón consiste en declarar una interfaz de operaciones AbstractFactory que crea una familia de productos que también son definidos en clases abstractas, toda la interacción del cliente será mediante las interfaces declaradas. Para la creación de nuevos productos se realiza a través de la interfaz sin necesidad de saber que Factory está siendo utilizada ni cual Producto se recibe tal como se muestra en la figura 9.

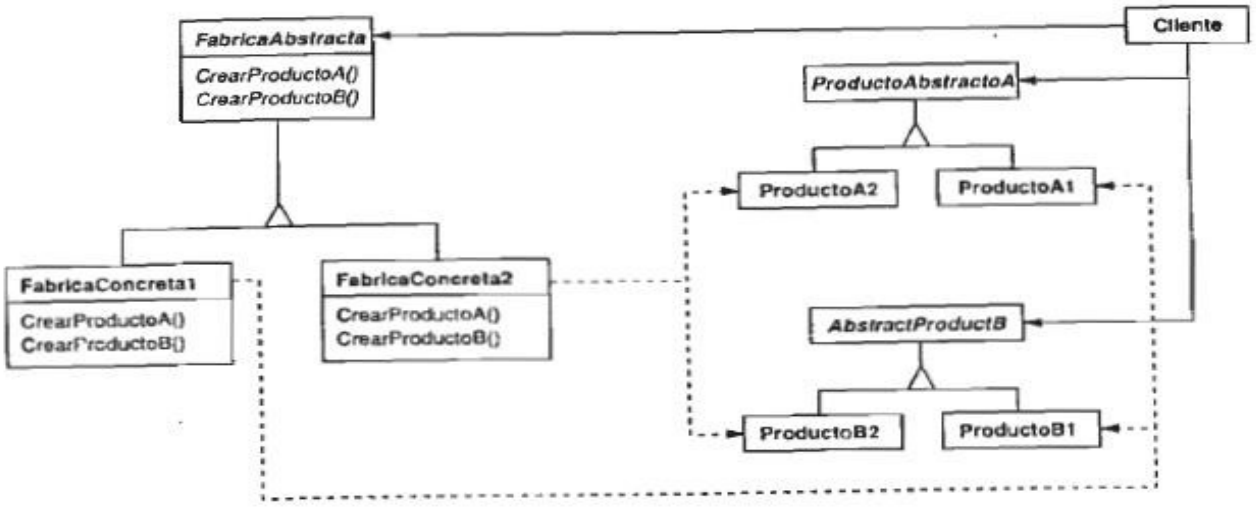


Figura 9 Estructura del patrón de diseño Abstract Factory

Como se puede apreciar este patrón es muy aplicado en sistemas que son independientes de cómo se crean, componen y representan sus productos. Con el uso del mismo se incrementa la flexibilidad del diseño, resultando fácil cambiar de familia de productos y promueve la consistencia entre productos, según Gamma (1995).

En la aplicación este patrón se evidencia en la clase *Factory* al crear una familia de instancias de las vistas del sistema a partir de la cual la clase *WFactory* se encarga concretamente de crear las vistas de la aplicación web, las que luego son utilizadas por la clase *ViewsManager*. Esta estructura forma la vista del patrón MVC.

```
class Factory: public Views::Factory
{
public:
        Factory () {};
        virtual ~Factory () {};
        Views::MainWindowsView* createMainWindowsView ();
        Views::PasswordAuthenticationView* createPasswordAuthenticationView ();
        Views::MessageBox* createMessageBox ();
```

```
        Views::ChangePasswordView* createChangePasswordView ();
        Views::UserDetailView* createUserDetailView();
        Views::PreferencesView* createPreferencesView();
        Views::PasswordView* createPasswordView();
        Views::LogsView* createLogsView ();
};
```

Este método crea una instancia de la vista principal.

Views: MainWindowsView Factory::createMainWindowsView()*

Este método crea una instancia de la vista Autenticarse.

*Views::PasswordAuthenticationView * Factory::createPasswordAuthenticationView ()*

Este método crea una instancia de la vista para mostrar mensajes.

Views::MessageBox Factory::createMessageBox()*

Este método crea una instancia de la vista Cambiar Contraseña.

Views::ChangePasswordView Factory::createChangePasswordView()*

Este método crea una instancia de la vista Detalle de Usuario.

Views::UserDetailView Factory::createUserDetailView()*

Este método crea una instancia de la vista Preferencias.

Views::PreferencesView Factory::createPreferencesView()*

Este método crea una instancia de la vista para el trabajo con las contraseñas.

Views::PasswordView Factory::createPasswordView()*

Este método crea una instancia de la vista de Logs de Seguridad.

Views::LogsView Factory::createLogsView ()*

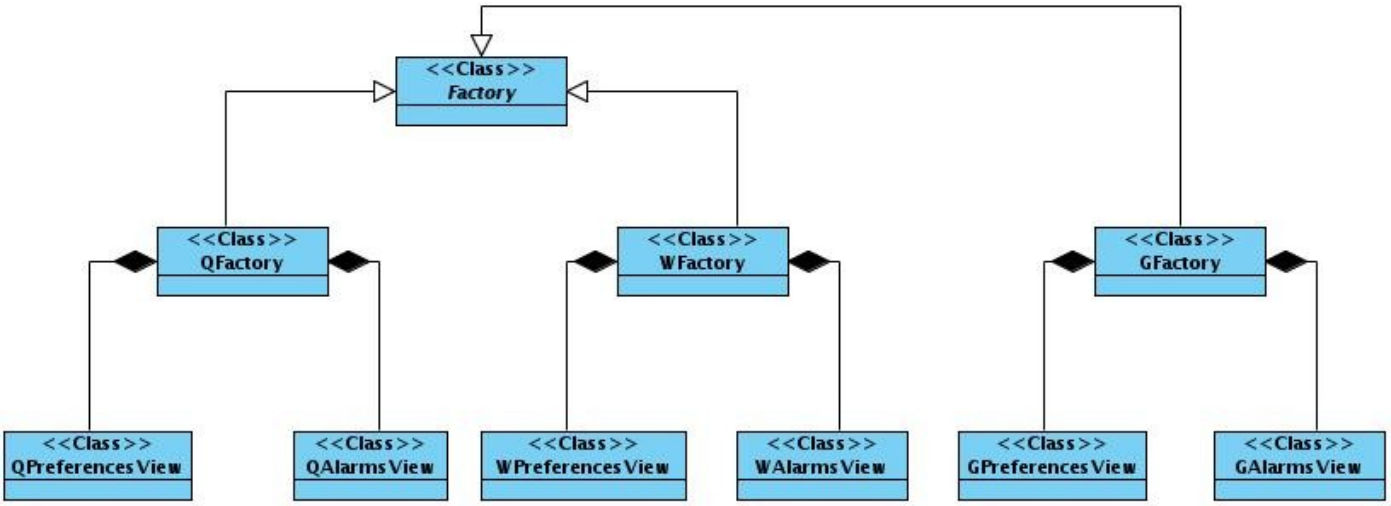


Figura 10 Representación del Patrón Abstract Factory en la aplicación

3.2.2 Patrón Command

Encapsula una petición en un objeto, permitiendo así parametrizar a los clientes con diferentes peticiones, hacer cola o llevar un registro de las peticiones y poder deshacer las operaciones (Gamma, y otros, 1995).

Según la propia fuente, el uso de este patrón consiste en crear un objeto OrdenConcreta y especificar su receptor almacenándolo en un objeto Invocador y este envía la petición a la interfaz. Cuando las órdenes se pueden deshacer, OrdenConcreta guarda el estado para deshacer la orden. El objeto OrdenConcreta invoca operaciones de su receptor para llevar a cabo la petición tal como se muestra en la figura 11.

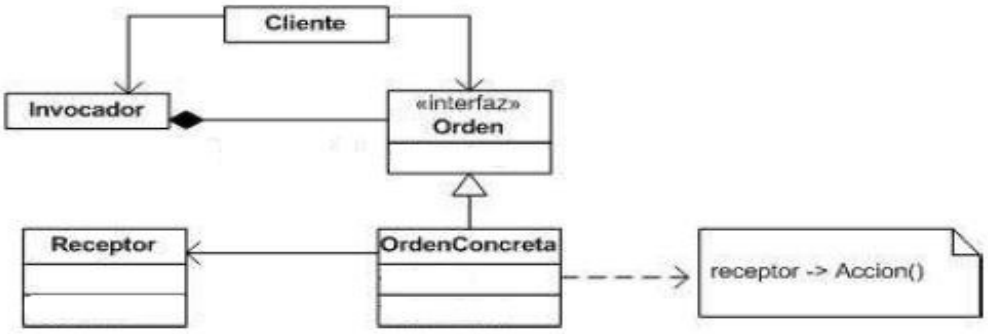


Figura 11 Estructura del patrón de diseño Command

Como se puede apreciar este patrón es muy utilizado cuando se quiere Parametrizar objetos con una acción a realizar, Especificar, poner en cola y ejecutar peticiones en diferentes instantes de tiempo,

Permitir registrar los cambios de manera que se puedan volver a aplicar en caso de una caída del sistema y ofrecer un modo de modelar transacciones (encapsular un conjunto de cambios sobre unos datos), según Gamma (1995).

En la aplicación este patrón comunica la vista con el modelo haciendo función de controlador. Las clases *Authentication, Load, LogOut, Save, ShowChangePassword, ShowLogs, ShowMainWindow, ShowMessage, ShowPassword, ShowPasswordAuthentication, ShowPreferences, ShowUserDetail* heredan de la clase *Command* la cual delega en cada una de estas las responsabilidades específicas a realizar para garantizar el correcto funcionamiento del sistema.

```
class Command
{
public:
        Command();
        virtual ~Command ();
        virtual void exec() = 0;
protected:
        Model::ModelManager* modelManager ();
};
```

Este método de ejecución es el que se re implementa en cada una de las hijas de la clase *Command*, cuando la misma delega las diferentes responsabilidades a cumplir:

```
virtual void exec() = 0;
```

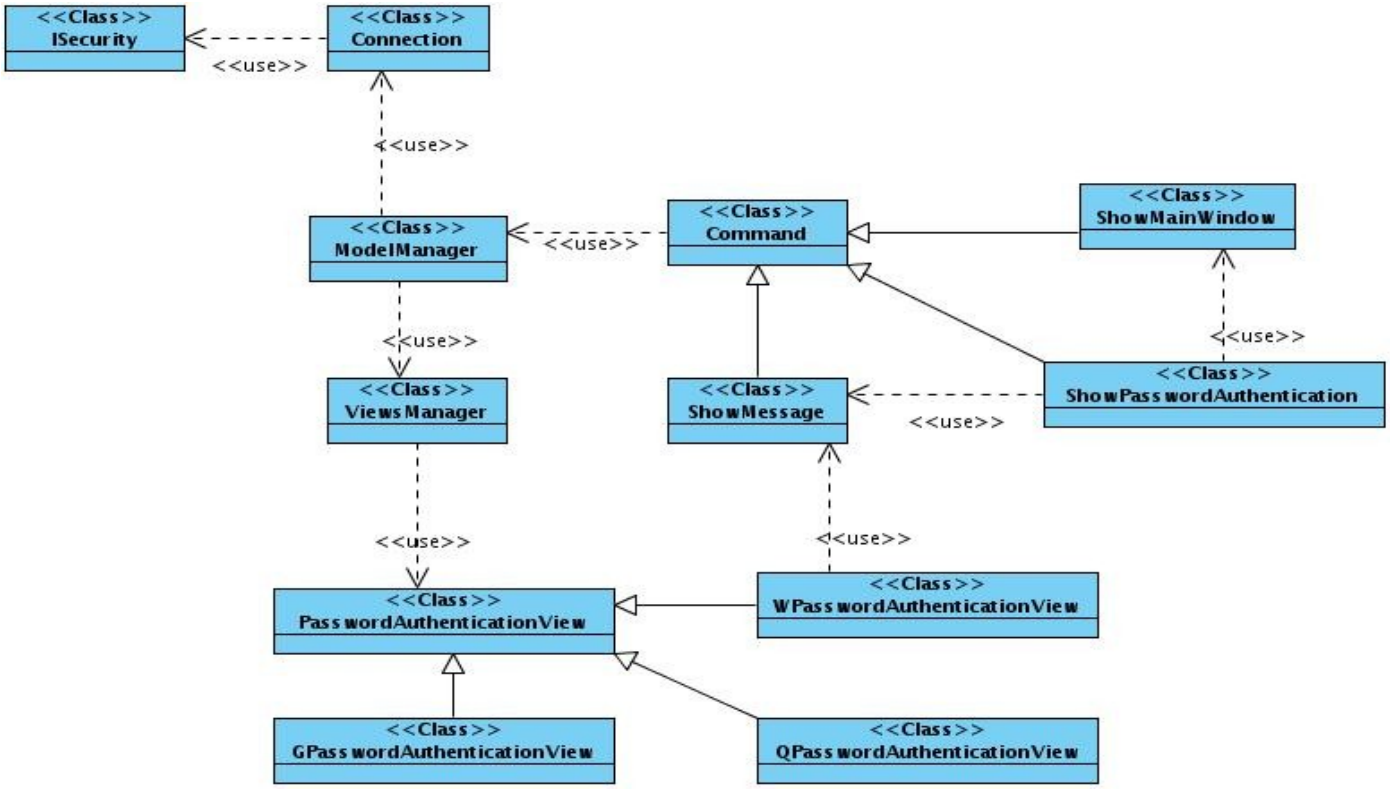


Figura 12 Representación del Patrón Command en la aplicación

3.2.3 Patrón Singleton

Garantiza que una sola clase sólo tenga una instancia y proporciona un punto de acceso global a ella.

El uso de este patrón consiste en definir una operación Instancia que permite que los clientes acceder a su única instancia. Instancia es una operación de clase que puede ser responsable de crear su única instancia. Los clientes acceden a la instancia de un Singleton exclusivamente a través de la operación Instancia de esta (Gamma, y otros, 1995).

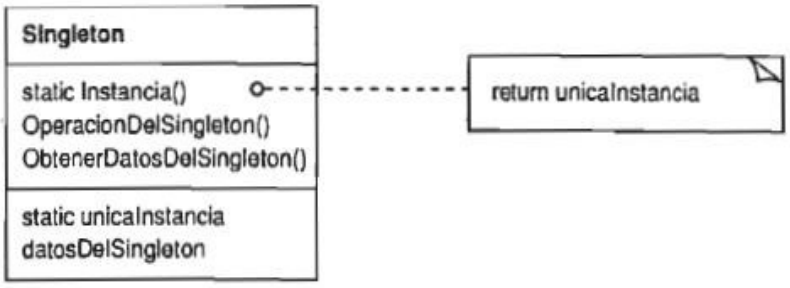


Figura 13 Estructura del patrón de diseño Singleton

Según la propia fuente, este patrón es muy utilizado ya que entre sus ventajas está la de acceso controlado a la única instancia, espacio de nombres reducido, permite el refinamiento de operaciones y la representación, permite un número variable de instancias y es más flexible que las operaciones de clases.

En la aplicación este patrón se evidencia al garantizar que solo se pueda crear una instancia única de la clase *ModelManager,* así como crear un punto de acceso global a esta por las demás clases del sistema. *Singleton* forma parte del modelo dentro del patrón MVC.

```
template <typename T>
class Singleton
{
public:
        static T* getInstance ()
        {
                static T* instance = new T ();
                return instance;
        }
};
```

Figura 14 Representación del Patrón Singleton en la aplicación

3.3 Diagrama de clases del diseño

Los diagramas de clases son un tipo de diagrama estático que describe la estructura de un sistema. Estos son utilizados durante el proceso de diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema (Esquivel, 2009).

Según la propia fuente, en el diseño del sistema los diagramas de clases son la manera de describir gráficamente las especificaciones de las clases de software y las interfaces en una aplicación. Los mismos contienen la siguiente información:

- Clases, asociaciones y atributos.

- Interfaces, con sus operaciones y constantes.
- Métodos.
- Información sobre los tipos de los atributos.
- Navegabilidad.
- Dependencias.

Como se puede apreciar un diagrama de clase contiene las definiciones de las entidades de la aplicación a diferencia de un modelo conceptual que representa conceptos del mundo real.
En el Anexo 3 se ven representados todos los diagramas de clases del diseño de la aplicación.

3.4 Diagramas de interacción

La creación de los Diagramas de Interacción de un sistema es una de las actividades más importantes en el desarrollo de un sistema, pues al construirlos se toman decisiones clave acerca de su funcionamiento futuro. Un diagrama de interacción consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos (Guerrero, 2008).Existen dos tipos de diagramas de interacción, ambos basados en la misma información, pero cada uno enfatizando un aspecto particular: Diagramas de Secuencia y Diagramas de Colaboración.

Según la propia fuente, un aspecto que se debe tener presente en el momento de realizar un diagrama de interacción es la correspondencia que debe existir con los diagramas de clases de diseño, razón por la que ambas tareas se llevan en paralelo. Para el desarrollo del diseño del sistema se seleccionó el diagrama de secuencia como la forma de representar el comportamiento dinámico del sistema.

En el Anexo 4 se ven representados todos los diagramas de secuencia del diseño.

Conclusiones parciales del capítulo 3

Al finalizar Capítulo “Diseño del Sistema”, se le ha dado cumplimiento a una de las tareas trazadas, el diseño de la herramienta web para administradores de seguridad. Se hace uso de tres patrones de diseño y un estilo arquitectónico descrito en secciones anteriores, los cuales contribuyen a lograr una arquitectura estable y sólida y a crear un plano del modelo de implementación.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

En el desarrollo de este capítulo se describirá la implementación de la aplicación web para administradores de seguridad usando componentes. Además se podrá apreciar con una breve descripción, las interfaces que le dan solución a las diferentes funcionalidades de la misma. Además de dar una descripción de las pruebas realizadas a la herramienta web.

La implementación se comienza con los resultados del diseño e implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares (Jacobson, et al., 2004). Teniendo como propósito principal el desarrollo de la arquitectura y el sistema como un todo.

4.1 Diagrama de componentes

El diagrama de componentes se utiliza para modelar la vista estática de un sistema, mostrando la organización y las dependencias entre un conjunto de componentes (Romero, 2002).

Según la propia fuente, es muy importante tener en cuenta que para su creación no es necesario que un diagrama incluya todos los componentes del sistema, normalmente estos se realizan por partes donde cada una describe un apartado del sistema. En él se representan las librerías, tablas, archivos, ejecutables y documentos que formen parte del mismo. Uno de los principales usos que se les da es para observar que componentes pueden compartirse entre sistemas o entre diferentes partes de uno mismo.

Llegado el momento de la implementación de la herramienta, todos los diagramas generados en fases anteriores se materializan en un sistema integrando las partes necesarias para la obtención de un producto final. Estas relaciones de dependencia se modelan a través del diagrama de componentes, como se muestra en el Anexo 5.

El diagrama de componentes del sistema de administración web se estructuró en paquetes, donde a cada clase definida en el diseño se le asignó un componente representando además las bibliotecas que necesita el sistema para su correcto funcionamiento. Los paquetes se definieron según los identificados en el diseño.

4.2 Diagrama de despliegue

Los Diagramas de Despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la

disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria. Los estereotipos permiten precisar la naturaleza del equipo: Dispositivos, Procesadores, Memoria (Marca, et al., 2000).

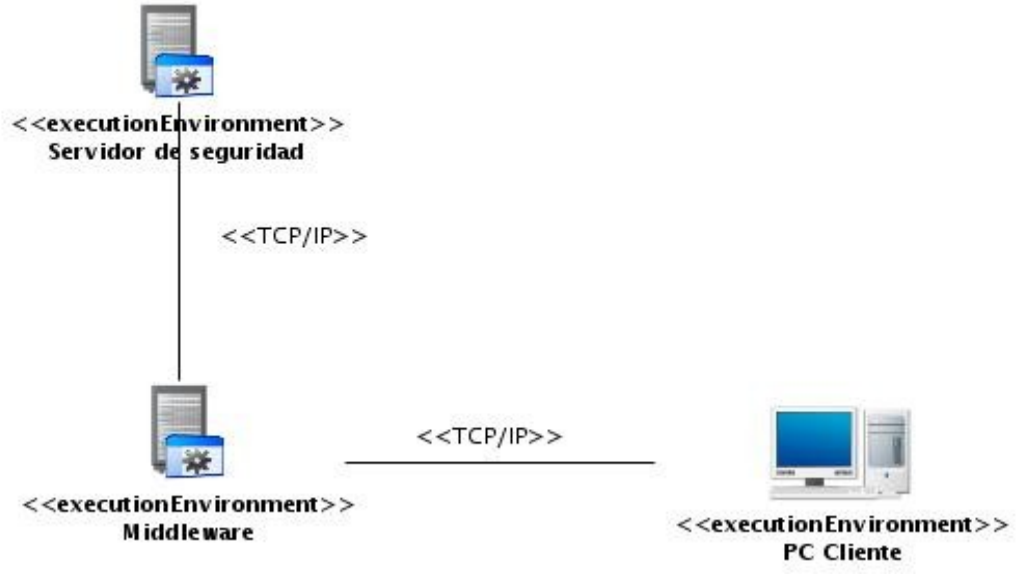


Figura 15 Diagrama de despliegue de la herramienta

Cancerbero: Herramienta web para administradores de seguridad.

Servidor de seguridad: Es una aplicación en C++ que publica a través del uso del subsistema Middleware las interfaces necesarias para que las funcionalidades con que cuenta, las cuales son Subsistema Autenticación, Subsistema Control de Accesos y Subsistema de detección de intrusos, puedan ser invocadas por cualquier cliente.

Middleware: Subsistema de Comunicación del SCADA Nacional Guardián del ALBA.

4.3 Pantallas de la Herramienta de Modelación Gráfica desarrollada

La aplicación desarrollada consta de una interfaz principal. En la parte superior se encuentra ubicada la barra de herramienta la cual contiene funcionalidades básicas de la aplicación como cambiar contraseña, mostrar logs, administrar sesiones, administrar alarmas, administrar variables entre otras, como se representa en la figura 16.

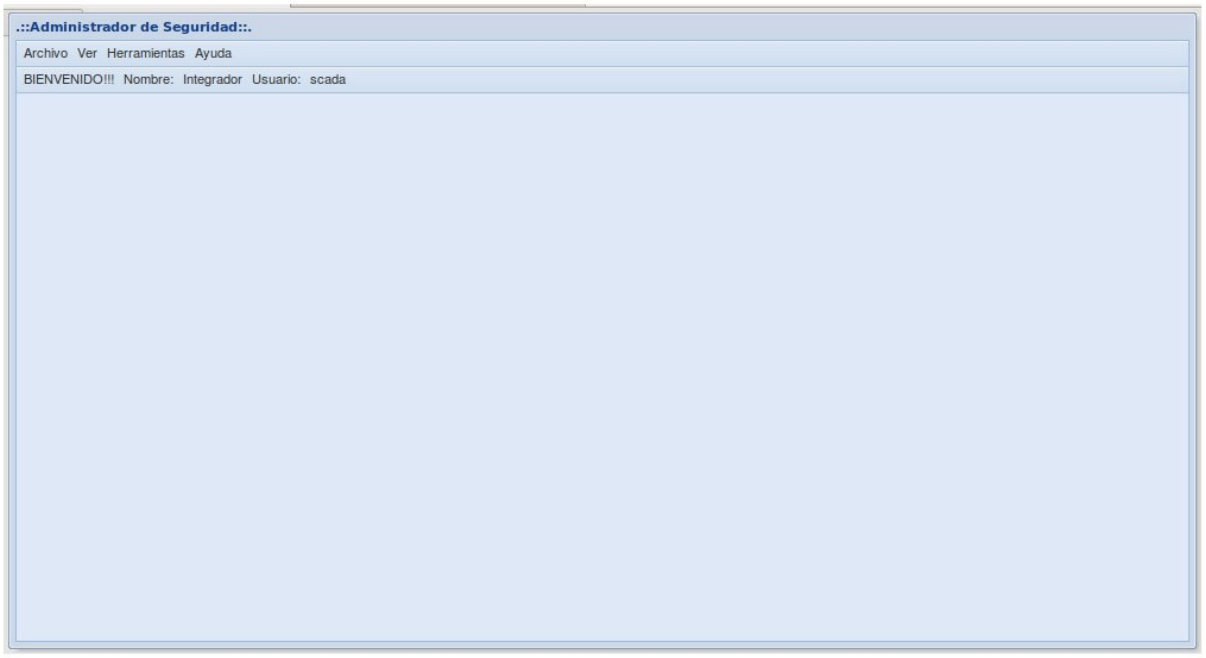


Figura 16 Interfaz principal de la aplicación

A continuación se describen cada una de las funcionalidades que presenta la herramienta desarrollada.

Autenticación

Para poder acceder a la aplicación web para administradores de seguridad primeramente es necesario autenticarse en el sistema a través de un usuario y contraseña.

Figura 17 Interfaz Autenticar

Cambiar contraseña

Dentro de las funcionalidades que se le permite al administrador de seguridad está la de cambiar contraseña, para esto es necesaria la contraseña actual que es comprobada durante el proceso de cambio y una nueva contraseña, a la cual se le va midiendo la fortaleza a medida que se va escribiendo. Esta no será cambiada hasta que la fortaleza sea de 80 o más.

Figura 18 Interfaz Cambiar contraseña

Mostrar logs

Esta funcionalidad permite realizar una búsqueda avanzada de los logs de seguridad en cuanto a autenticación o control de accesos, y búsqueda de los usuarios que se encuentran registrados en el sistema. En cada una de las opciones de búsqueda el administrador de seguridad puede especificar diferentes criterios de búsqueda, en caso de no introducir ningún dato el sistema muestra todos los logs según la opción escogida y en caso de introducir un criterio de búsqueda erróneo el sistema muestra los resultados de la búsqueda vacios.

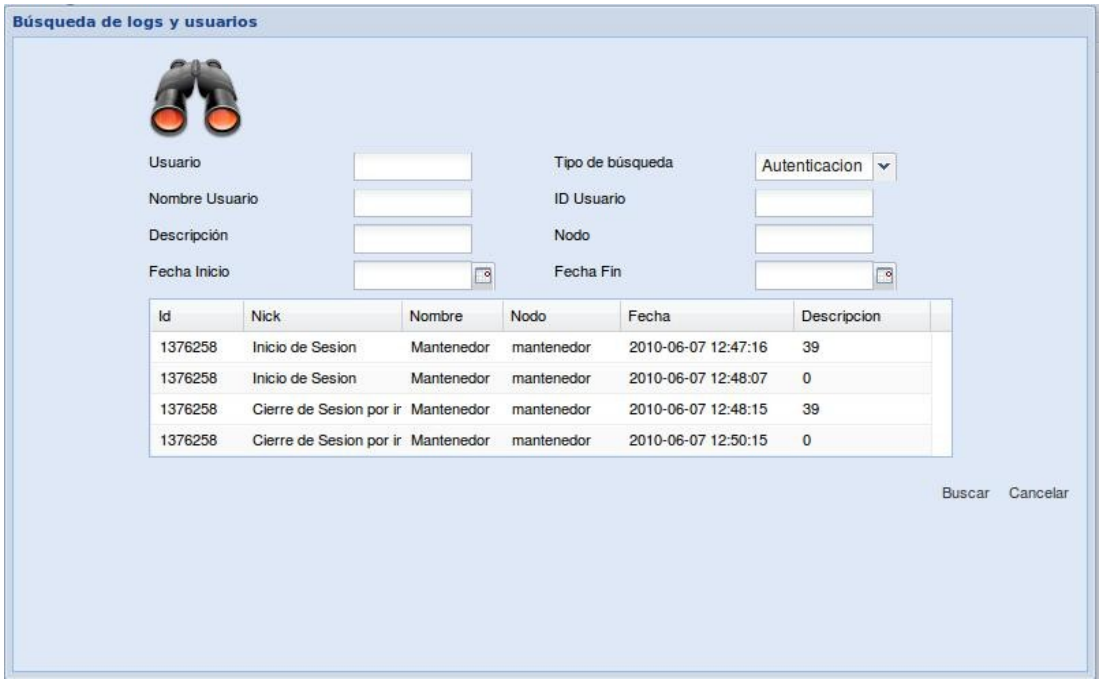


Figura 19 Interfaz Búsqueda de logs y usuarios por autenticación

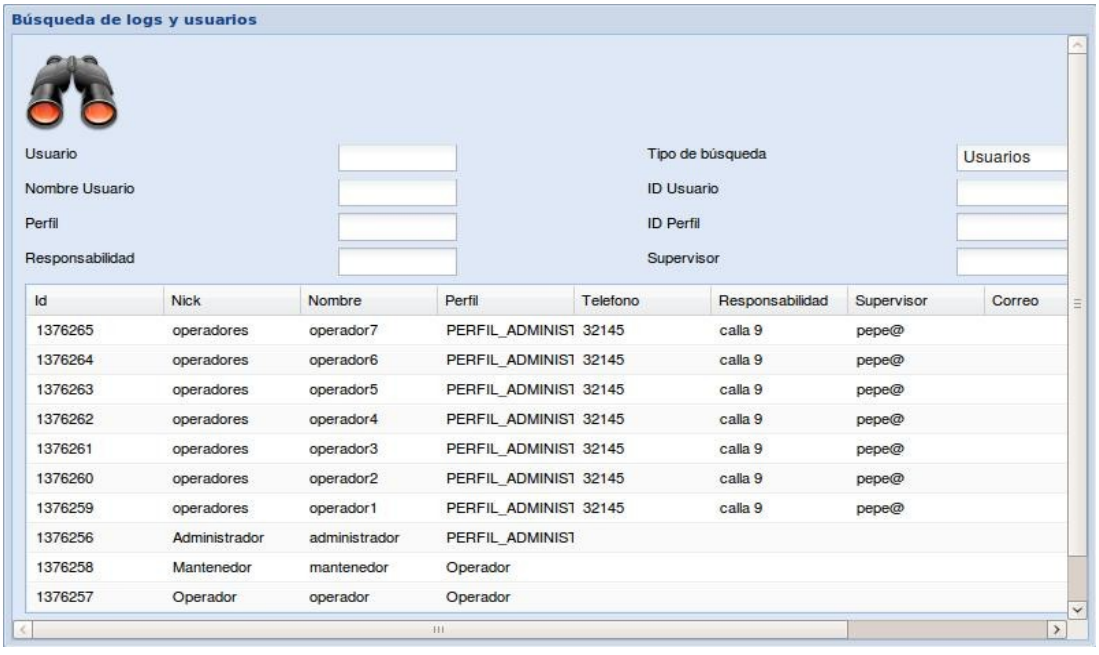


Figura 20 Interfaz Búsqueda de logs y usuarios por búsqueda de usuarios

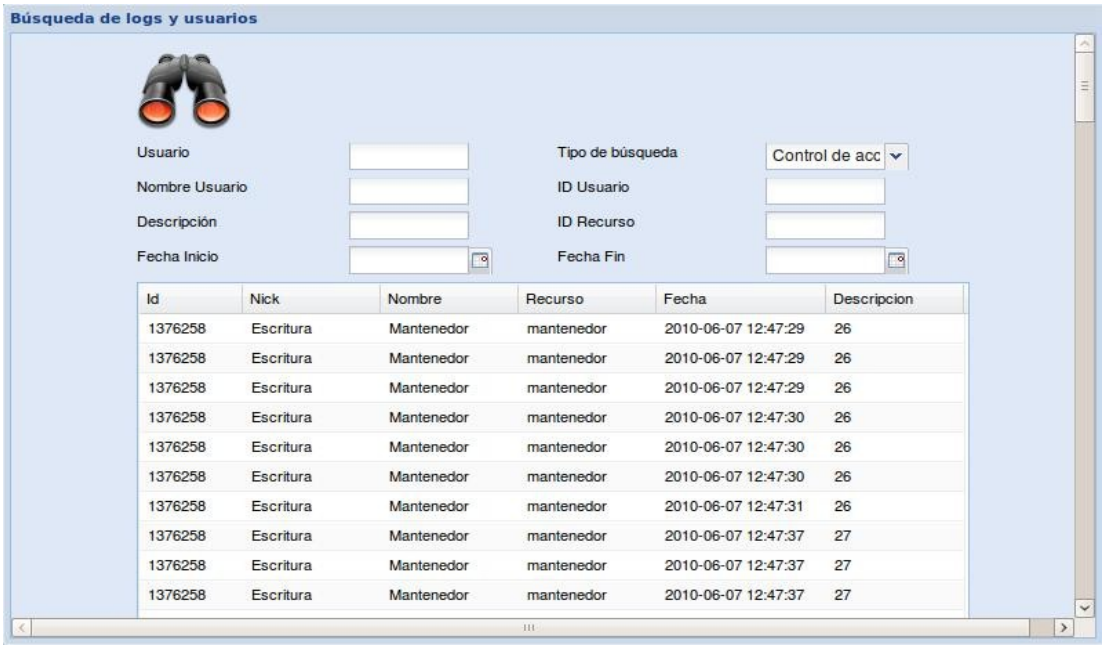


Figura 21 Interfaz Búsqueda de logs y usuarios por control de acceso

Mostrar detalles de usuario

Esta funcionalidad le permite al administrador de seguridad conocer los detalles de los usuarios que se encuentran conectadas al sistema. Al escoger cualquier usuario mostrado en las sesiones o en la búsqueda se muestran los detalles del mismo.

Figura 22 Interfaz Mostrar detalles de usuario

Administrar sesiones

Esta funcionalidad le brinda la posibilidad al administrador de seguridad de conocer todos los datos de las sesiones de trabajo creadas en el sistema dándole la posibilidad de eliminar la sesión que él estime pertinente, en caso que la sesión eliminada sea en la que se está trabajando el sistema automáticamente emite un mensaje de error para que la aplicación sea cerrada.

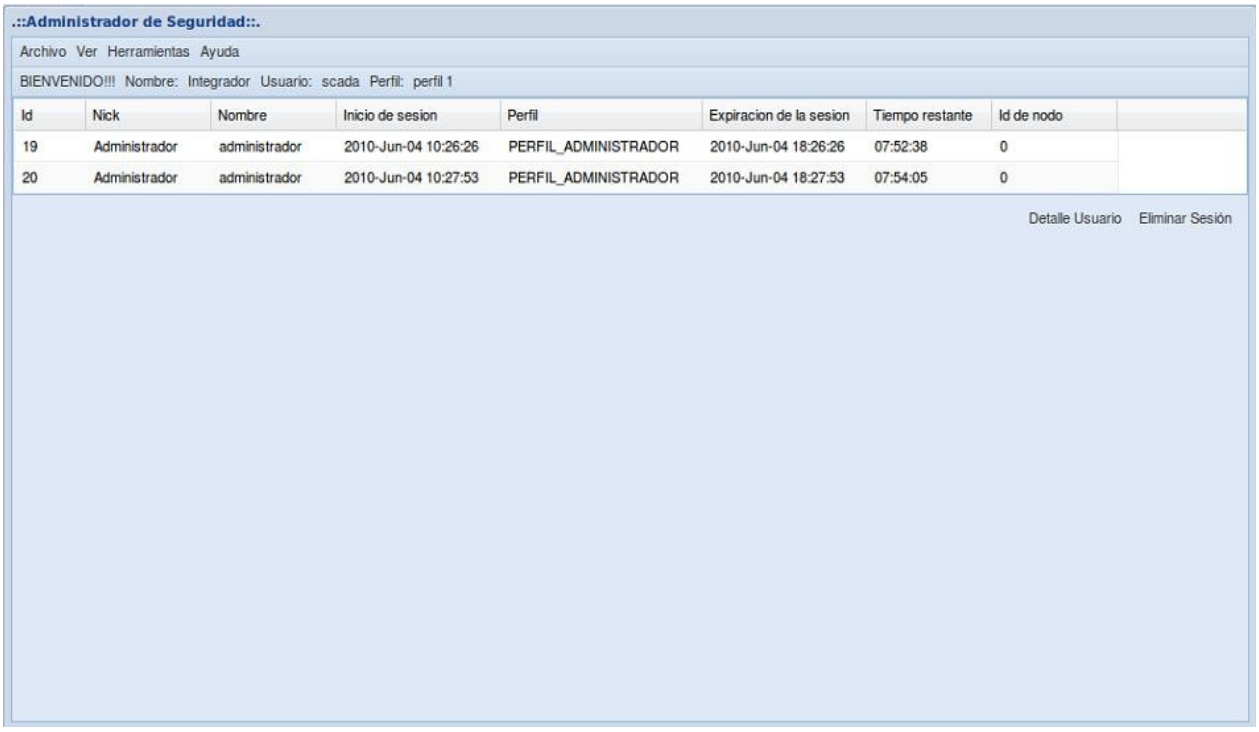


Figura 23 Interfaz Administrar sesiones

Administrar variables

Esta funcionalidad permite cargar todos los valores de las variables que están en el servidor de seguridad, las cuales pueden referirse a la autenticación, la contraseña y los recursos. Brinda además la posibilidad de cambiar estas variables en caso de ser necesario o que el administrador desee.

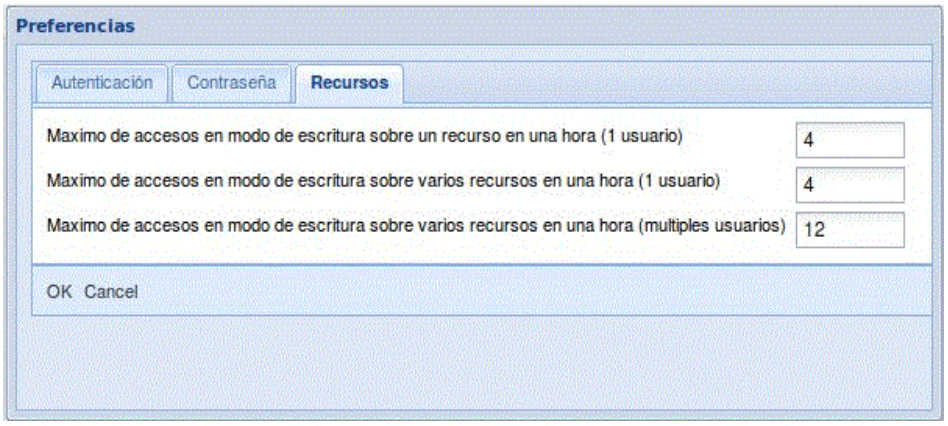
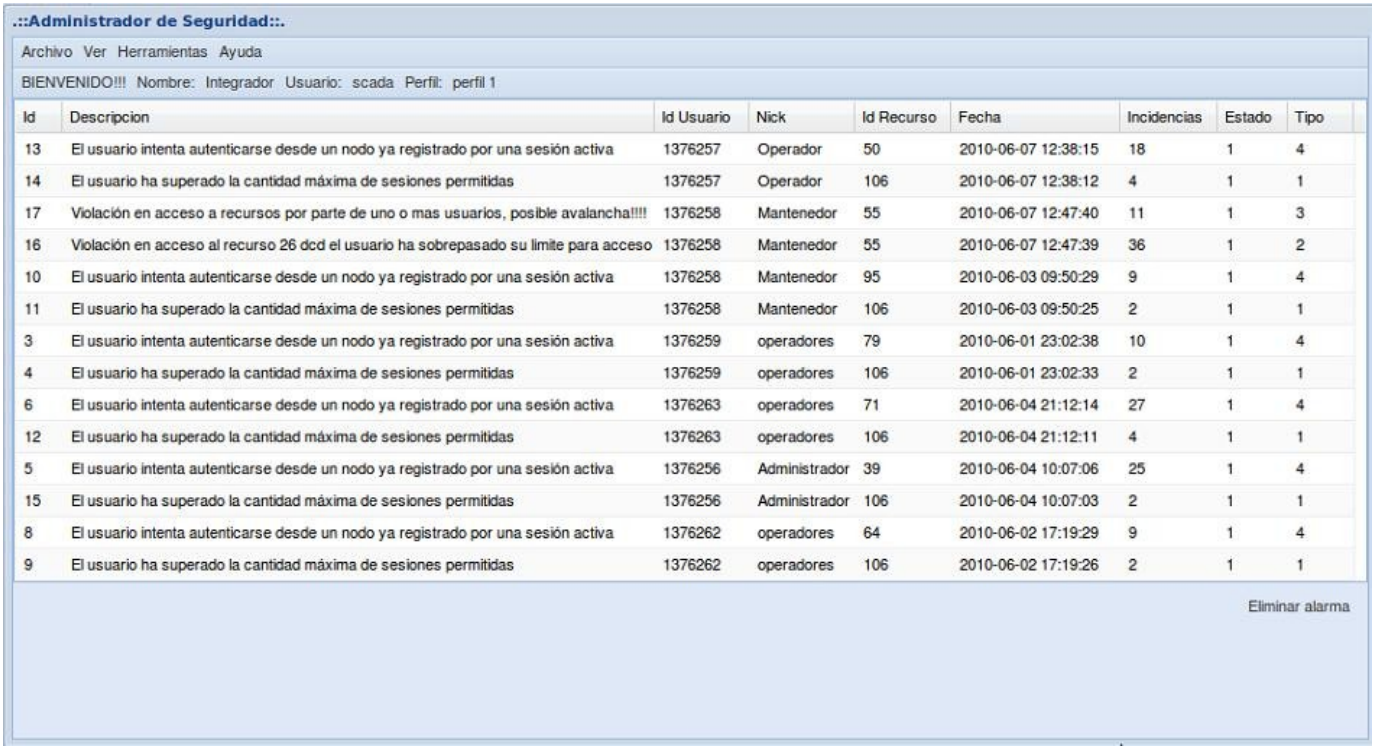


Figura 24 Interfaz Administrar variables

Administrar alarmas

Esta funcionalidad le brinda la posibilidad al usuario de conocer todas las alarmas que se disparan de las sesiones abiertas en ese momento dentro del sistema dándole la posibilidad de eliminar la alarma que él estime pertinente.



Id	Descripcion	Id Usuario	Nick	Id Recurso	Fecha	Incidencias	Estado	Tipo
13	El usuario intenta autenticarse desde un nodo ya registrado por una sesión activa	1376257	Operador	50	2010-06-07 12:38:15	18	1	4
14	El usuario ha superado la cantidad máxima de sesiones permitidas	1376257	Operador	106	2010-06-07 12:38:12	4	1	1
17	Violación en acceso a recursos por parte de uno o mas usuarios, posible avalancha!!!!	1376258	Mantenedor	55	2010-06-07 12:47:40	11	1	3
16	Violación en acceso al recurso 26 dcd el usuario ha sobrepasado su limite para acceso	1376258	Mantenedor	55	2010-06-07 12:47:39	36	1	2
10	El usuario intenta autenticarse desde un nodo ya registrado por una sesión activa	1376258	Mantenedor	95	2010-06-03 09:50:29	9	1	4
11	El usuario ha superado la cantidad máxima de sesiones permitidas	1376258	Mantenedor	106	2010-06-03 09:50:25	2	1	1
3	El usuario intenta autenticarse desde un nodo ya registrado por una sesión activa	1376259	operadores	79	2010-06-01 23:02:38	10	1	4
4	El usuario ha superado la cantidad máxima de sesiones permitidas	1376259	operadores	106	2010-06-01 23:02:33	2	1	1
6	El usuario intenta autenticarse desde un nodo ya registrado por una sesión activa	1376263	operadores	71	2010-06-04 21:12:14	27	1	4
12	El usuario ha superado la cantidad máxima de sesiones permitidas	1376263	operadores	106	2010-06-04 21:12:11	4	1	1
5	El usuario intenta autenticarse desde un nodo ya registrado por una sesión activa	1376256	Administrador	39	2010-06-04 10:07:06	25	1	4
15	El usuario ha superado la cantidad máxima de sesiones permitidas	1376256	Administrador	106	2010-06-04 10:07:03	2	1	1
8	El usuario intenta autenticarse desde un nodo ya registrado por una sesión activa	1376262	operadores	64	2010-06-02 17:19:29	9	1	4
9	El usuario ha superado la cantidad máxima de sesiones permitidas	1376262	operadores	106	2010-06-02 17:19:26	2	1	1

Figura 25 Interfaz Administrar alarmas

4.4 Pruebas

Las pruebas son una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos específicos, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente (UCI, 2009-2010).

Según la propia fuente, la Prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo.

- **Niveles de pruebas** (UCI, 2009-2010)**.**

- **Prueba de Desarrollador:** Es la prueba diseñada e implementada por el equipo de desarrollo. Tradicionalmente estas pruebas han sido consideradas solo para la prueba de unidad, aunque en la actualidad en algunos casos pueden ejecutar pruebas de integración. Se recomienda que estas pruebas cubran más que las pruebas de unidad.

- **Prueba independiente:** Es la prueba que es diseñada e implementada por alguien independiente del grupo de desarrolladores. El objetivo de estas pruebas es proporcionar una perspectiva deferente y en un ambiente más rico que los desarrolladores. Una vista de la prueba independiente es la prueba independiente de los stakeholder, que son pruebas basadas en las necesidades y preocupaciones de los stakeholders.
- **Prueba de unidad:** Es la prueba enfocada a los elementos testeables más pequeño del software. Es aplicable a componentes representados en el modelo de implementación para verificar que los flujos de control y de datos están cubiertos, y que ellos funcionen como se espera. La prueba de unidad siempre está orientada a caja blanca.
- **Prueba de integración:** Es ejecutada para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso. Se prueba un paquete o un conjunto de paquetes del modelo de implementación. Estas pruebas descubren errores en las especificaciones de las interfaces de los paquetes, además debe ser responsabilidad de desarrolladores y de independientes, sin solaparse las pruebas.
- **Prueba de Sistema:** Son las pruebas que se hacen cuando el software está funcionando como un todo. Es la actividad de prueba dirigida a verificar el programa final, después que todos los componentes de software y hardware han sido integrados.

- **Prueba de aceptación:** Prueba de aceptación del usuario es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido.

➢ **Tipos de Prueba** (UCI, 2009-2010)**.**

Cada tipo de prueba tiene un objetivo específico y una técnica que lo soporte.
La siguiente lista muestra los tipos de pruebas basado en dimensiones de calidad:

Dimensión de Calidad **Riesgo de calidad**	**Tipos de Prueba**
Funcionalidad	**Función:** Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso. **Seguridad:** Asegurar que los datos o el sistema solamente es accedido por los actores deseados. **Volumen**: Enfocada en verificando las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la BD.
Usabilidad	**Usabilidad:** Prueba enfocada a factores humanos, estéticos, consistencia en la interfaz de usuario, ayuda sensitiva al contexto y en línea, asistente documentación de usuarios y materiales de entrenamiento.
Fiabilidad	**Integridad:** Enfocada a la valoración de la robustez (resistencia a fallos). **Estructura:** Enfocada a la valoración a la adherencia a su diseño y formación. Este tipo de prueba es hecho a las aplicaciones web asegurando que todos los enlaces están conectados, el contenido deseado es mostrado y no hay contenido huérfano. **Stress:** Enfocada a evaluar cómo el sistema responde bajo condiciones anormales. (extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible, recursos compartidos no disponible)
Performance	**Benchmark:** es un tipo de prueba que compara el

(Rendimiento)	rendimiento de un elemento nuevo o desconocido a uno de carga de trabajo de referencia conocido. **Contención**: Enfocada a la validación de las habilidades del elemento a probar para manejar aceptablemente la demanda de múltiples actores sobre un mismo recurso (registro de recursos, memoria, etc) **Carga:** Usada para validar y valorar la aceptabilidad de los límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba permanece constante. La variación en carga es simular la carga de trabajo promedio y con picos que ocurre dentro de tolerancias operacionales normales. **Performance profile:** Enfocadas a monitorear el tiempo en flujo de ejecución, acceso a datos, en llamada a funciones y sistema para identificar y direccional los cuellos de botellas y los procesos ineficientes.
Soportabilidad	**Configuración:** Enfocada a asegurar que funciona en diferentes configuraciones de hardware y software. Esta prueba es implementada también como prueba de rendimiento del sistema **Instalación:** Enfocada a asegurar la instalación en diferentes configuraciones de hardware y software bajo diferentes condiciones (insuficiente espacio en disco, etc.)

Tabla 10 Tipos de prueba

A partir de los diferentes niveles y tipos de pruebas analizados, se deciden realizar pruebas de sistema específicamente de caja negra así como pruebas de rendimiento específicamente de carga.

A continuación se muestra el desarrollo de algunas pruebas de sistema que fueron realizadas.

- **Caso de Uso Autenticar**

Descripción general: Este caso de uso permite al usuario autenticarse con el sistema a partir de su usuario y su contraseña.

Caso de prueba #1: Autenticar

1. **Descripción:** Este caso de prueba le permite al usuario autenticarse con el sistema.
2. **Flujo central:**

   2.1 El usuario inserta su usuario y su contraseña.

   2.2 El usuario selecciona la opción Aceptar.

   2.3 El sistema realiza la validación de los datos insertados y lleva a cabo la autenticación.
3. **Iteraciones**

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba
Introduce todos los campos correctamente, Ejemplos usuario: administrador contraseña: 123456		El sistema realiza la validación de los datos correctamente y lleva a cabo la autenticación.	Satisfactorio
	El usuario deja campos vacíos Ejemplo: usuario: administrador contraseña:	El sistema muestra un mensaje de error alertando que los campos de usuario o contraseña son incorrectos.	Satisfactorio
	El usuario introduce valores incorrectos. Ejemplo: usuario: admin contraseña: 123	El sistema muestra un mensaje de error alertando que los campos de usuario o contraseña son incorrectos.	Satisfactorio

	El usuario introduce usuario con caracteres no válidos Ejemplo: usuario: /--	El sistema muestra un mensaje de error informando de que el usuario o la contraseña no pueden contener los siguientes caracteres: /, ', --.	Satisfactorio

Tabla 11 Caso de Prueba Autenticar

- **Caso de Uso Cambiar contraseña**

Descripción general: Este caso de uso permite al usuario cambiar su contraseña para acceder al sistema.

Caso de prueba #2: Cambiar contraseña

1. **Descripción:** Este caso de prueba le permite al usuario cambiar su contraseña para acceder al sistema.
2. **Flujo central**

   2.1 El usuario introduce su contraseña actual para autenticarse en el sistema.

   2.2 El usuario introduce la nueva contraseña así como la confirmación de la misma.

   2.3 El sistema realiza la validación de los datos insertados y lleva a cabo el cambio de contraseña.
3. **Iteraciones**

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba
Introduce el campo de la contraseña correctamente, Ejemplo contraseña: 123456		El sistema realiza la validación del dato correctamente y lleva a cabo la autenticación.	Satisfactorio
	El usuario deja el campo vacío Ejemplo: contraseña:	El sistema muestra un mensaje de error alertando que la contraseña es incorrecta.	Satisfactorio

	El usuario introduce valores incorrectos. Ejemplo: contraseña: qwert	El sistema muestra un mensaje de error alertando que la contraseña es incorrecta.	Satisfactorio
Introduce el campo nueva contraseña correctamente, con una fortaleza mayor que el 80%, Ejemplo contraseña: scada123		El sistema realiza la validación del dato correctamente y solicita la reafirmación de la nueva contraseña.	Satisfactorio
	Introduce el campo nueva contraseña incorrecta, no tiene una fortaleza mayor de 80%. Ejemplo contraseña: scada	El sistema realiza la validación del dato y muestra un mensaje de error notificando que la fortaleza de su nueva contraseña es muy débil.	Satisfactorio
	El usuario introduce una nueva contraseña y su confirmación diferentes. Ejemplos nueva contraseña: 4ut0m4t1c4 confirmación de la contraseña: 4ut0m4t1c4;	El sistema muestra un mensaje de error informando al usuario que la nueva contraseña y su confirmación no coinciden.	Satisfactorio

Tabla 12 Caso de Prueba Cambiar Contraseña

- **Caso de Uso Mostrar logs**

Descripción general: Este caso de uso permite realizar una búsqueda avanzada de los logs de seguridad.

Caso de prueba #3: Buscar usuarios.

1. **Descripción:** Este caso de prueba le permite al usuario realizar la búsqueda de usuarios conectados al sistema.
2. **Flujo central:**
   2.1 El usuario introduce su nombre, usuario o id para realizar la búsqueda de usuario y mostrar algún log de seguridad.
   2.2 El sistema realiza la validación de los datos insertados y muestra los resultados de la búsqueda.
   2.3 En caso de que no se inserte ningún valor en los campos la búsqueda muestra todos los resultados existentes.

3. Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba
Introduce los datos de los campos correctamente, Ejemplo Usuario: administrador Nombre usuario: Administrador		El sistema realiza la validación de los datos correctamente y lleva a cabo la búsqueda mostrando los resultados.	Satisfactorio
El usuario deja los campos vacios.		El sistema lleva a cabo la búsqueda y muestra todos los datos de usuarios guardados en el servidor.	Satisfactorio
	El usuario introduce valores incorrectos. Ejemplo: Usuario: admin Nombre usuario: admin	El sistema muestra los campos de los resultados de la búsqueda vacíos.	Satisfactorio

Tabla 13 Caso de Prueba Buscar Usuarios

Caso de prueba #4: Buscar logs por autenticación

1. **Descripción:** Este caso de prueba le permite al usuario mostrar los logs guardados en el servidor de seguridad por autenticación.
2. **Flujo central:**

   2.1 El usuario introduce su nombre, usuario o id para realizar la búsqueda y mostrar algún log de seguridad por autenticación.

   2.2 El sistema realiza la validación de los datos insertados y muestra los resultados de la búsqueda.

   2.3 En caso de que no se inserte ningún valor en los campos la búsqueda muestra todos los resultados existentes.

3. **Iteraciones**

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba
Introduce los datos de los campos correctamente, Ejemplo Usuario: administrador Nombre usuario: Administrador		El sistema realiza la validación de los datos correctamente y lleva a cabo la búsqueda mostrando los resultados.	Satisfactorio
El usuario deja los campos vacíos.		El sistema lleva a cabo la búsqueda y muestra todos los datos de logs por autenticación guardados en el servidor.	Satisfactorio
	El usuario introduce valores incorrectos. Ejemplo: Usuario: admin Nombre usuario: admin	El sistema muestra los campos de los resultados de la búsqueda vacíos.	Satisfactorio

Tabla 14 Caso de Prueba Buscar Logs por Autenticación

Caso de prueba #5: Buscar logs por control de acceso

1. **Descripción:** Este caso de prueba le permite al usuario mostrar los logs guardados en el servidor por control de acceso.
2. **Flujo central:**
   2.1 El usuario introduce su nombre, usuario o id para realizar la búsqueda de usuario y mostrar3. Iteraciones algún log de seguridad.
   2.2 El sistema realiza la validación de los datos insertados y muestra los resultados de la búsqueda.
   2.3 En caso de que no se inserte ningún valor en los campos la búsqueda muestra todos los resultados.
3. **Iteraciones**

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba
Introduce los datos de los campos correctamente, Ejemplo Usuario: administrador Nombre usuario: Administrador		El sistema realiza la validación de los datos correctamente y lleva a cabo la búsqueda mostrando los resultados.	Satisfactorio
El usuario deja los campos vacíos.		El sistema lleva a cabo la búsqueda y muestra todos los datos de usuarios guardados en el servidor.	Satisfactorio
	El usuario introduce valores incorrectos. Ejemplo: Usuario: admin Nombre usuario: Admin	El sistema muestra los campos de resultados de la búsqueda vacíos.	Satisfactorio

Tabla 15 Caso de Prueba Buscar Logs por Control de Acceso

Caso de prueba #6: Administrar variables

1. **Descripción:** Este caso de prueba le permite al usuario configurar las variables que se necesitan en el servidor.

2. **Flujo central:**

2.1 El usuario introduce los nuevos valores de las variables.

2.2 El sistema realiza la validación de los datos insertados y lleva a cabo el cambio de variables.

3. Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba
Introduce el valor de las variables correctamente, Ejemplo Máximo de autenticaciones que un usuario puede hacer: 6		El sistema realiza la validación del dato correctamente y lleva a cabo la actualización de la variable.	Satisfactorio
	El usuario deja campos vacíos Ejemplo: Máximo de autenticaciones que un usuario puede hacer:	El sistema muestra un mensaje de error alertando que el valor introducido no es un número.	Satisfactorio
	El usuario introduce valores incorrectos. Ejemplo: Máximo de autenticaciones que un usuario puede hacer: ae-	El sistema muestra un mensaje de error alertando que el valor introducido no es un número.	Satisfactorio

Tabla 16 Caso de Prueba Administrar Variables

Para la realización de las pruebas de carga se utilizó una herramienta automatizada lo cual brinda la posibilidad de que ni el desarrollador ni una persona externa sean los que generen los casos de prueba, de esta forma, se evita el problema de que el propio desarrollador se haga " trampas al solitario", esto quiere decir, que como él realizó el desarrollo de la unidad está ansioso de que ésta sea correcta y quizás genere casos de prueba triviales en los cuales la unidad no falla y se evitan además los altos costos al involucrar una segunda persona; por otro lado, se evita también el alto tiempo que insume generar los casos de prueba de forma manual y asegurando una mejor calidad del sistema desarrollado.

- **Pruebas de carga**

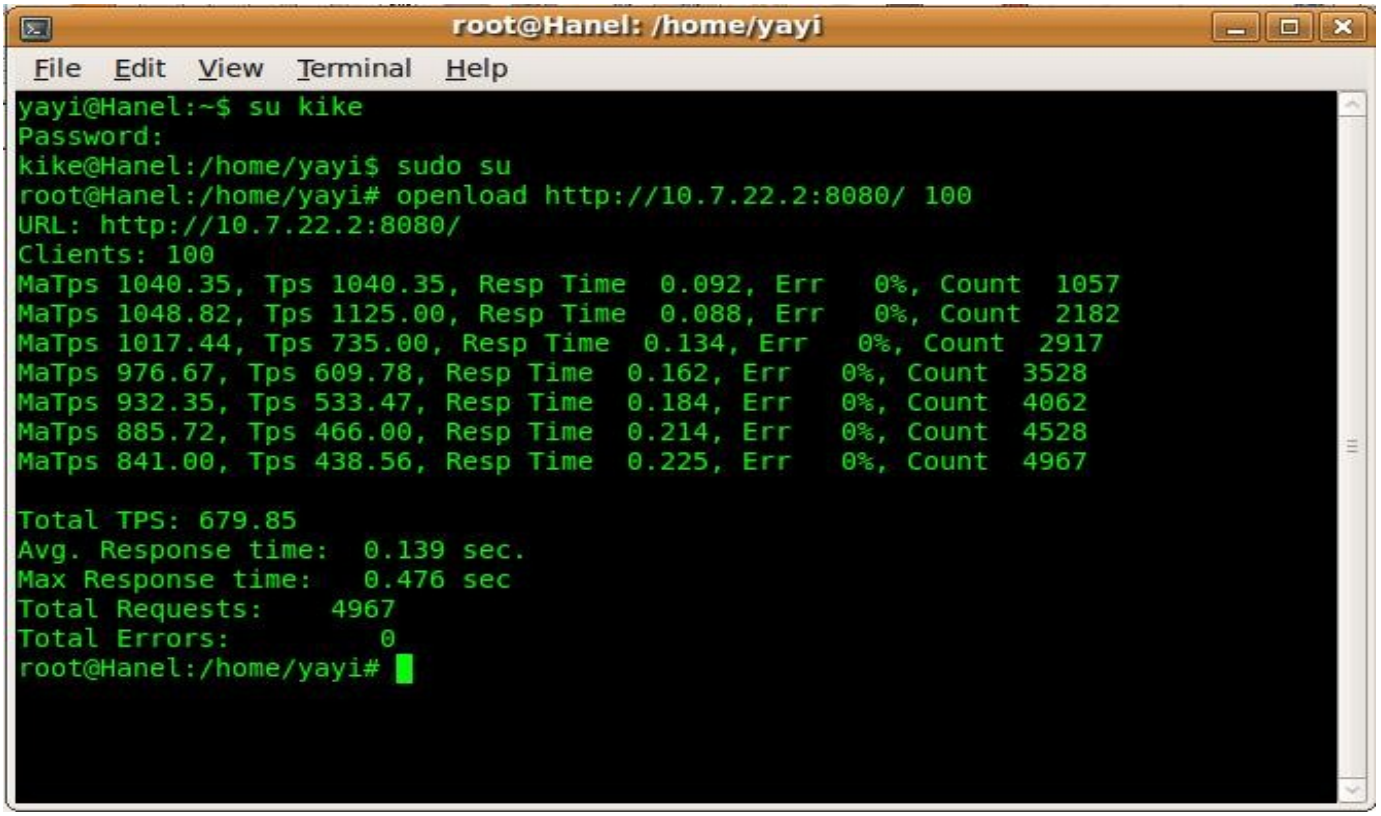


Figura 26 Ejecución de las pruebas de carga

Donde:

- **MaTps:** Promedio de 20 segundos de TPS.
- **Tps:** (transacciones por segundo) es el número de peticiones realizadas durante ese segundo.
- **Resp Time:** Tiempo medio de respuesta en segundos para el segundo transcurrido.
- **Err:** El porcentaje de respuestas con errores.
- **Count:** Solicitudes completadas.
- **Total TPS:** Promedio de toda la prueba, (Total de las solicitudes completadas) / (Total tiempo transcurrido).
- **Avg. Response time:** Tiempo medio de respuesta total en cuestión de segundos.
- **Max Response time:** Tiempo de respuesta más alto durante la prueba.
- **Total Requests:** Número total de solicitudes completadas.

- **Total Errors:** Total de respuestas con errores.

Para la validación de la solución propuesta se realizaron 9 casos de pruebas, a partir de los cuales se emitió como resultado final que 5 casos de pruebas no presentan errores y los 4 restantes si presentan.

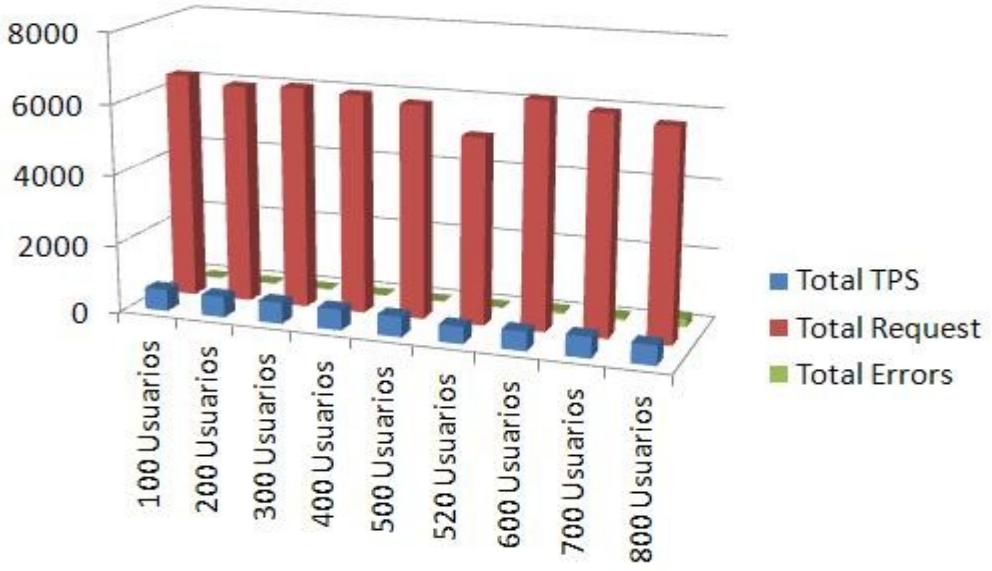


Figura 27 Resultado de las pruebas de carga

En el Anexo 6 se muestra cada uno de los casos de pruebas ejecutados.

Conclusiones parciales del capítulo 4

Al culminar el capítulo “Implementación y Prueba”, se le ha dado cumplimiento a las últimas tareas trazadas para el desarrollo de la aplicación, quedando totalmente implementada la herramienta web para administradores de seguridad. Se realizaron varias pruebas de unidad para garantizar que se hayan cumplido todos los requisitos funcionales.

Conclusiones Generales

Con la realización de este trabajo se logró cumplir con los objetivos propuestos en el mismo así como arribar a las siguientes conclusiones:

- Con el objetivo de comprender y describir las clases más importantes dentro del contexto del sistema se realizó un modelo del dominio, pues los procesos del negocio no se encontraban bien definidos.
- El prototipo desarrollado incluye las funcionalidades definidas para el sistema de administración de seguridad, logrando el acceso al mismo desde diferentes ambientes de trabajo.
- La realización de este trabajo contribuye al fortalecimiento de la línea de desarrollo Seguridad, brindando un nuevo sistema de administración de fácil acceso a través del uso de los navegadores web.

Recomendaciones

Al concluir este trabajo se recomienda:

- Proponer a la comunidad agregar las mejoras realizadas a la biblioteca Wt detectadas a partir del desarrollo de nuestra aplicación
- Continuar con el mejoramiento del sistema web para administradores de seguridad a partir de los nuevos cambios que se les realicen a la biblioteca Wt por la comunidad.
- Proponer la tecnología utilizada a otros módulos del centro que tengan la necesidad de extender sus sistemas a la web.

Referencias Bibliográficas

Alvarez, Francisco José Zumbado. 2008. Sistema computarizado para el manejo de inventario de bodega. Sistema computarizado para el manejo de inventario de bodega.Costa Rica : s.n., 2008.

Babylon. 2007. Entorno de desarrollo integrado. Babylon. [En línea] 2007. [Citado el: 18 de Enero de 2010.] Disponible en: [http://www.babylon.com/definition/entorno_de_desarrollo/Spanish]

Barrios, Ricardo. 2007. Tutorial:Introduction to Ext (Spanish). ExtJS. [En línea] 7 de Marzo de 2007. [Citado el: 10 de Febrero de 2010.] Disponible en: [http://www.extjs.com/learn/Tutorial:Introduction_to_Ext_%28Spanish%29]

Booch, G, Rumbaugh, J y Jacobson, I. 1999.1999.

Calero, M. 2003.Una explicación de la programación extrema (XP). Madrid : s.n., 2003.

Casanovas, Josep. 2004. Usabilidad y arquitectura del software. desarrolloweb.com. [En línea] 9 de Septiembre de 2004. [Citado el: 20 de Febrero de 2010.] Disponible en: [http://www.desarrolloweb.com/articulos/1622.php]

CRYPTEX. 2010. Report: RSA 2010 Global Online Consumer Security Survey. CRYPTEX. [En línea] 2 de Febrero de 2010. [Citado el: 1 de Marzo de 2010.] Disponible en: [http://seguridad-informacion.blogspot.com/2010/02/report-rsa-2010-global-online-consumer.html]

Debian. 2010. ¿ Que es Debian ? Debian. [En línea] 2010. [Citado el: 23 de Marzo de 2010.] Disponible en: [http://www.debian.org/intro/about]

E_intelligent. 2005. Que ventajas tiene Linux sobre Windows. E_intelligent. [En línea] 2005. [Citado el: 25 de Enero de 2010.] Disponible en: [http://www.entmexico.com/hosting/windows-o-linux.html]

Eclipse. 2010. ¿Que es Eclipse? Eclipse. [En línea] 2010. [Citado el: 3 de Marzo de 2010.] Disponible en: [http://plataformaeclipse.com/]

El_Rincon_de_Linux. 2009. Sobre Linux. El Rincon de Linux. [En línea] 2009. [Citado el: 20 de Febrero de 2010.] Disponible en: [http://www.linux-es.org/sobre_linux]

E Entornos_de_Desarrollo_Integrado. 2008. Entornos de Desarrollo Integrado. Sitio Web de la EU, de Ingenieria Tecnica de Oviedo. [En línea] 2008. [Citado el: Enero de 22 de 2010.] Disponible en: [http://petra.euitio.uniovi.es/~i1667065/HD/documentos/Entornos%20de%20Desarrollo%20Integrado.pdf]

Esquivel, Ing: Karina. 2009. Blog de WordPress.com. TEMA 9: DIAGRAMA DE CLASE EN UML. [En línea] 2009. [Citado el: 28 de Febrero de 2010.] Disponible en: [http://kesquivel.files.wordpress.com/2009/08/tema-9_diagrama_clase.ppt]

Gamma, Erich, y otros. 1995.Patrones de Diseño. España : Cofas,S.A, 1995.

Giraldo, Luis y Zapata, Yuliana. 2005.*HERRAMIENTAS DE DESARROLLO DE INGENIERIA DE SW PARA LINUX.* s.l. : Monitoria de Ingesoft, 2005.

Gobierno_de_Canarias. 2010. Aplicaciones Web. Gobierno de Canarias. [En línea] 2010. [Citado el: 28 de Abril de 2010.] Disponible en: [http://www.gobiernodecanarias.org/medusa/contenidosclicescuela2.0/index.asp@r0=tutoriales&r3=1.html]

GSInnova. Rational Software Architect. GSInnova. [En línea] [Citado el: 25 de Enero de 2010.] Disponible en: [http://www.rational.com.ar/herramientas/softwarearchitect.html]

Guerrero, Luis. 2008.UML - Diagramas de interacción. Chile : Departamento de Ciencias de la Computación, 2008.

INFORMATICA, INSTITUTO NACIONAL DE ESTADISTICA E. 1999.*Herramientas Case.* Jesus Maria : INEI, 1999.

Jacobson, I. 2000.*El proceso unificado de desarrollo de software.* Madrid : Person Educaciónón S.A, 2000.

Jacobson, Ivar y Booch, Grady y Rumbaugh, James. 2004.*El Proceso Unificado de Desarrollo de Software.* 2004. pág. 255. Vol. I.

—. 2004.*El Proceso Unificado de Desarrollo de Software, Implementacion.* 2004.

—. 2004.*El Proceso Unificado del Software, Casos de Uso.* 2004.

—. 2004.*El Proceso Unificado del Software, Modelo de dominio.* 2004.

Lago, Ramiro. 2007. Patrones de diseño software. Proactiva. [En línea] Abril de 2007. [Citado el: 25 de Marzo de 2010.] Disponible en: [http://www.proactiva-calidad.com/java/patrones/index.html]

Lagos, H. 2008. Proyecto Curricular. Gestión Unidad Técnica Pedagógica. [En línea] 29 de Julio de 2008. [Citado el: 1 de Febrero de 2010.] Disponible en: [http://hugolagospedagogia.blogspot.com/2008/07/proyecto-curricular.html]

Lenguajes_de_Programacion. 2009. Lenguajes de Programacion. *Lenguajes de Programacion.* [Online] 2009. [Cited: marzo 5, 2010.] Disponible en: [http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml]

Lenguajes_de_Programación. 2009. Programación Web. Lenguajes de Programación. [En línea] 2009. [Citado el: 17 de Marzo de 2010.] Disponible en: [http://www.lenguajes-de-programacion.com/programacion-web.shtml]

Marca, Hugo y Quisberti, Nancy. 2000.*ANALISIS Y DISEÑO DE SISTEMAS II, Diagrama de despliegue.* 2000.

Montalvo, Carlos. 2008. Aplicaciones web Ventajas y Desventajas? Calin Soft. [En línea] 2008. [Citado el: 3 de Noviembre de 2009.] Disponible en: [http://www.calinsoft.com/2008/08/aplicaciones-web-ventajas-y-desventajas.html]

Programación. 2009. ¿ Que es la Programación ? Programación. [En línea] 2009. [Citado el: 8 de Febrero de 2010.] Disponible en: [http://www.masterhacks.20m.com/custom4_1.html]

Rivas, Alonso. 1988.*Auditoria Informática.* Madrid : Ediciones Diaz de Santos, S.A, 1988.

Roger y Pressman. 2005.*Ingeniería del Software.* La Habana : Félix Varela, 2005.

Romero, Oscar. 2002. Diagrama de componentes. Programacion en castellano. [En línea] 25 de Noviembre de 2002. [Citado el: 9 de Abril de 2010.] Disponible en: [http://www.programacion.com/articulo/introduccion_a_uml_181/6]

Soluciones_Seguras. 2009. El rol del administrador de seguridad. Soluciones Seguras. [En línea] 2009. [Citado el: 15 de febrero de 2010.] Disponible en: [http://www.revistaitnow.com]

Taringa. 2009. Seguridad Informatica...Para tener en cuenta. Taringa. [En línea] 2009. [Citado el: 25 de Abril de 2010.] Disponible en: [http://www.taringa.net/posts/info/4020941/Seguridad-Informatica___Para-tener-en-cuenta.html]

UCI. 2008.*Conferencia 2 Ciclo Requerimientos.* Ciudad Habana : s.n., 2008.

Velasco, Perla. 2001. Prueba_de_Componentes_de_Software_basadas_en_el_Modelo_JavaBeans. *Prueba_de_Componentes_de_Software_basadas_en_el_Modelo_JavaBeans.* TLAXCALA : s.n., 2001.

Villalón, Antonio. 2002.*SEGURIDAD EN UNIX Y REDES.* 2002.

WT. 2009. Other benefits of using Wt. WT. [En línea] 2009. [Citado el: 23 de Febrero de 2010.] Disponible en: [http://www.webtoolkit.eu/wt]

UCI. (2009-2010). Fase de construccion, Disciplina prueba. Ciudad Habana.

Bibliografía

Alvarez, Francisco José Zumbado. 2008. Sistema computarizado para el manejo de inventario de bodega. Sistema computarizado para el manejo de inventario de bodega.Costa Rica : s.n., 2008.

Babylon. 2007. Entorno de desarrollo integrado. Babylon. [En línea] 2007. [Citado el: 18 de Enero de 2010.] Disponible en: [http://www.babylon.com/definition/entorno_de_desarrollo/Spanish]

Barrios, Ricardo. 2007. Tutorial:Introduction to Ext (Spanish). ExtJS. [En línea] 7 de Marzo de 2007. [Citado el: 10 de Febrero de 2010.] Disponible en: [http://www.extjs.com/learn/Tutorial:Introduction_to_Ext_%28Spanish%29]

Booch, G, Rumbaugh, J y Jacobson, I. 1999.1999.

Calero, M. 2003.Una explicación de la programación extrema (XP). Madrid : s.n., 2003.

Casanovas, Josep. 2004. Usabilidad y arquitectura del software. desarrolloweb.com. [En línea] 9 de Septiembre de 2004. [Citado el: 20 de Febrero de 2010.] Disponible en: [http://www.desarrolloweb.com/articulos/1622.php]

CRYPTEX. 2010. Report: RSA 2010 Global Online Consumer Security Survey. CRYPTEX. [En línea] 2 de Febrero de 2010. [Citado el: 1 de Marzo de 2010.] Disponible en: [http://seguridad-informacion.blogspot.com/2010/02/report-rsa-2010-global-online-consumer.html]

Debian. 2010. ¿ Que es Debian ? Debian. [En línea] 2010. [Citado el: 23 de Marzo de 2010.] Disponible en: [http://www.debian.org/intro/about]

E_intelligent. 2005. Que ventajas tiene Linux sobre Windows. E_intelligent. [En línea] 2005. [Citado el: 25 de Enero de 2010.] Disponible en: [http://www.entmexico.com/hosting/windows-o-linux.html]

Eclipse. 2010. ¿Que es Eclipse? Eclipse. [En línea] 2010. [Citado el: 3 de Marzo de 2010.] Disponible en: [http://plataformaeclipse.com/]

El_Rincon_de_Linux. 2009. Sobre Linux. El Rincon de Linux. [En línea] 2009. [Citado el: 20 de Febrero de 2010.] Disponible en: [http://www.linux-es.org/sobre_linux]

E Entornos_de_Desarrollo_Integrado. 2008. Entornos de Desarrollo Integrado. Sitio Web de la EU, de Ingenieria Tecnica de Oviedo. [En línea] 2008. [Citado el: Enero de 22 de 2010.] Disponible en: [http://petra.euitio.uniovi.es/~i1667065/HD/documentos/Entornos%20de%20Desarrollo%20Integrado.pdf]

Esquivel, Ing: Karina. 2009. Blog de WordPress.com. TEMA 9: DIAGRAMA DE CLASE EN UML. [En línea] 2009. [Citado el: 28 de Febrero de 2010.] Disponible en: [http://kesquivel.files.wordpress.com/2009/08/tema-9_diagrama_clase.ppt]

Gamma, Erich, y otros. 1995.Patrones de Diseño. España : Cofas,S.A, 1995.

Giraldo, Luis y Zapata, Yuliana. 2005.*HERRAMIENTAS DE DESARROLLO DE INGENIERIA DE SW PARA LINUX.* s.l. : Monitoria de Ingesoft, 2005.

Gobierno_de_Canarias. 2010. Aplicaciones Web. Gobierno de Canarias. [En línea] 2010. [Citado el: 28 de Abril de 2010.] Disponible en: [http://www.gobiernodecanarias.org/medusa/contenidosclicescuela2.0/index.asp@r0=tutoriales&r3=1.html]

GSInnova. Rational Software Architect. GSInnova. [En línea] [Citado el: 25 de Enero de 2010.] Disponible en: [http://www.rational.com.ar/herramientas/softwarearchitect.html]

Guerrero, Luis. 2008.UML - Diagramas de interacción. Chile : Departamento de Ciencias de la Computación, 2008.

INFORMATICA, INSTITUTO NACIONAL DE ESTADISTICA E. 1999.*Herramientas Case.* Jesus Maria : INEI, 1999.

Jacobson, I. 2000.*El proceso unificado de desarrollo de software.* Madrid : Person Educaciónón S.A, 2000.

Jacobson, Ivar y Booch, Grady y Rumbaugh, James. 2004.*El Proceso Unificado de Desarrollo de Software.* 2004. pág. 255. Vol. I.

—. 2004.*El Proceso Unificado de Desarrollo de Software, Implementacion.* 2004.

—. 2004.*El Proceso Unificado del Software, Casos de Uso.* 2004.

—. 2004.*El Proceso Unificado del Software, Modelo de dominio.* 2004.

Lago, Ramiro. 2007. Patrones de diseño software. Proactiva. [En línea] Abril de 2007. [Citado el: 25 de Marzo de 2010.] Disponible en: [http://www.proactiva-calidad.com/java/patrones/index.html]

Lagos, H. 2008. Proyecto Curricular. Gestión Unidad Técnica Pedagógica. [En línea] 29 de Julio de 2008. [Citado el: 1 de Febrero de 2010.] Disponible en: [http://hugolagospedagogia.blogspot.com/2008/07/proyecto-curricular.html]

Lenguajes_de_Programacion. 2009. Lenguajes de Programacion. *Lenguajes de Programacion.* [Online] 2009. [Cited: marzo 5, 2010.] Disponible en: [http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml]

Lenguajes_de_Programación. 2009. Programación Web. Lenguajes de Programación. [En línea] 2009. [Citado el: 17 de Marzo de 2010.] Disponible en: [http://www.lenguajes-de-programacion.com/programacion-web.shtml]

Marca, Hugo y Quisberti, Nancy. 2000.*ANALISIS Y DISEÑO DE SISTEMAS II, Diagrama de despliegue.* 2000.

Montalvo, Carlos. 2008. Aplicaciones web Ventajas y Desventajas? Calin Soft. [En línea] 2008. [Citado el: 3 de Noviembre de 2009.] Disponible en: [http://www.calinsoft.com/2008/08/aplicaciones-web-ventajas-y-desventajas.html]

Programación. 2009. ¿ Que es la Programación ? Programación. [En línea] 2009. [Citado el: 8 de Febrero de 2010.] Disponible en: [http://www.masterhacks.20m.com/custom4_1.html]

Rivas, Alonso. 1988.*Auditoria Informática.* Madrid : Ediciones Diaz de Santos, S.A, 1988.

Roger y Pressman. 2005.*Ingeniería del Software.* La Habana : Félix Varela, 2005.

Romero, Oscar. 2002. Diagrama de componentes. Programacion en castellano. [En línea] 25 de Noviembre de 2002. [Citado el: 9 de Abril de 2010.] Disponible en: [http://www.programacion.com/articulo/introduccion_a_uml_181/6]

Soluciones_Seguras. 2009. El rol del administrador de seguridad. Soluciones Seguras. [En línea] 2009. [Citado el: 15 de febrero de 2010.] Disponible en: [http://www.revistaitnow.com]

Taringa. 2009. Seguridad Informatica...Para tener en cuenta. Taringa. [En línea] 2009. [Citado el: 25 de Abril de 2010.] Disponible en: [http://www.taringa.net/posts/info/4020941/Seguridad-Informatica___Para-tener-en-cuenta.html]

UCI. 2008.*Conferencia 2 Ciclo Requerimientos.* Ciudad Habana : s.n., 2008.

Velasco, Perla. 2001. Prueba_de_Componentes_de_Software_basadas_en_el_Modelo_JavaBeans. *Prueba_de_Componentes_de_Software_basadas_en_el_Modelo_JavaBeans.* TLAXCALA : s.n., 2001.

Villalón, Antonio. 2002.*SEGURIDAD EN UNIX Y REDES.* 2002.

WT. 2009. Other benefits of using Wt. WT. [En línea] 2009. [Citado el: 23 de Febrero de 2010.] Disponible en: [http://www.webtoolkit.eu/wt]

UCI. (2009-2010). Fase de construccion, Disciplina prueba. Ciudad Habana.

Anexos

Anexo 1 Diagramas de clases del análisis

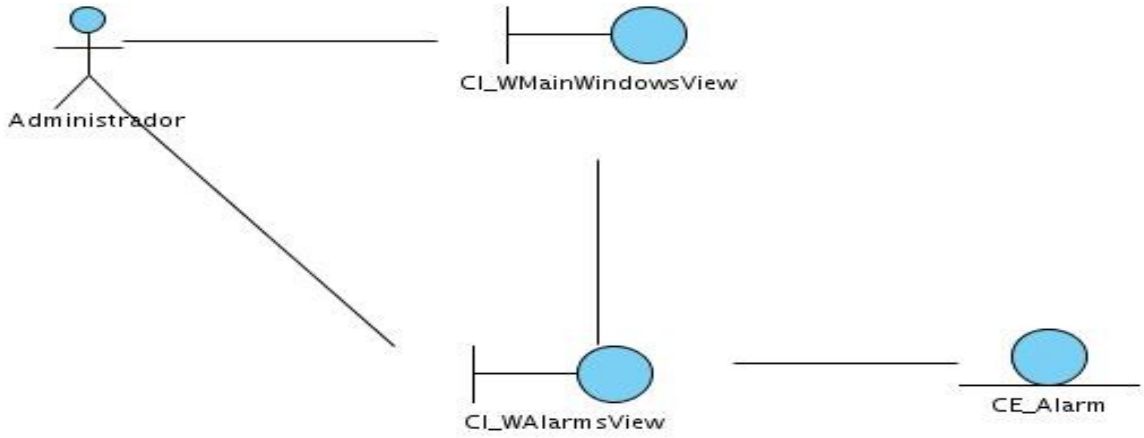


Figura 28 Diagrama de Clases del Análisis CU Admin alarmas

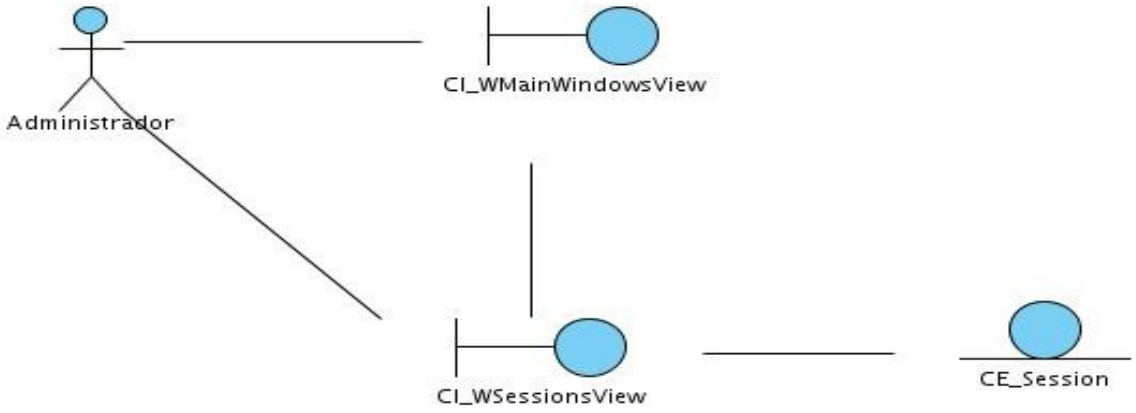


Figura 29 Diagrama de Clases del Análisis CU Admin sesiones

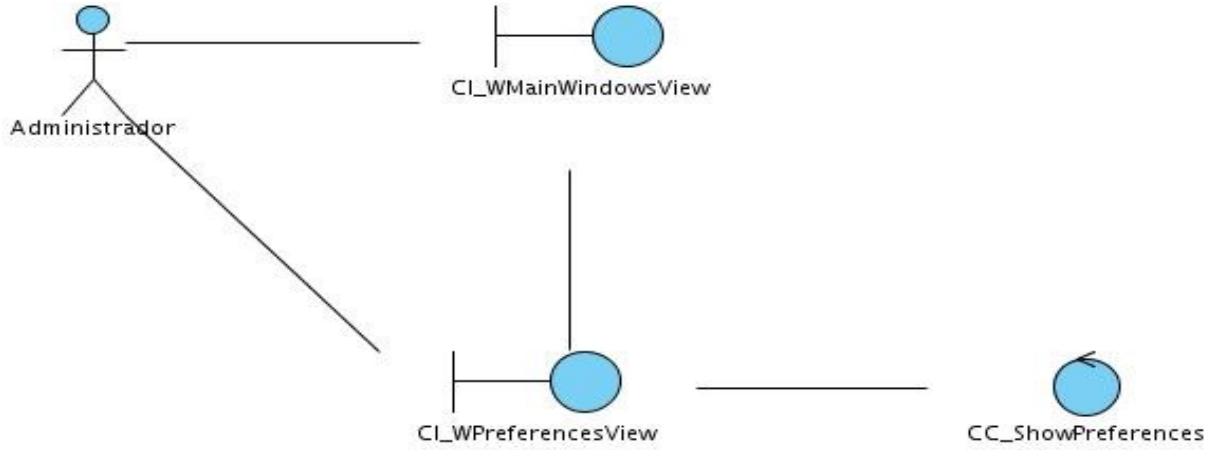


Figura 30 Diagrama de Clases del Análisis CU Admin variables

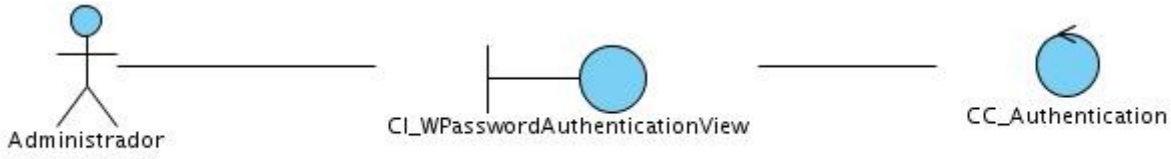


Figura 31 Diagrama de Clases del Análisis CU Autenticar

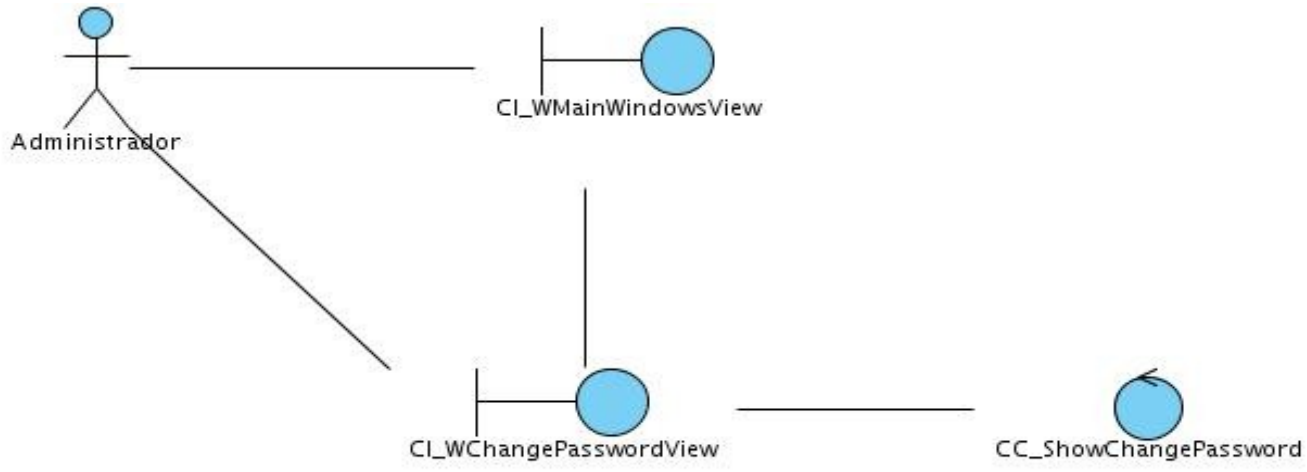


Figura 32 Diagrama de Clases del Análisis CU Cambiar contraseña

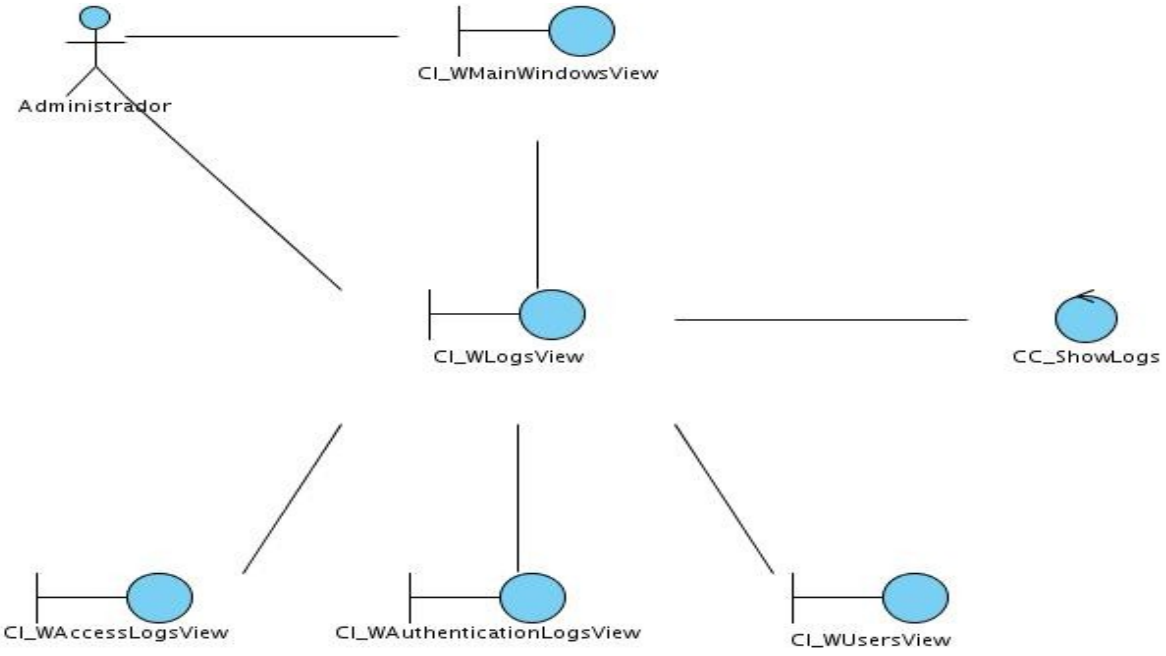


Figura 33 Diagrama de Clases del Análisis CU Mostrar logs

Anexo 2 Diagramas de colaboración del análisis

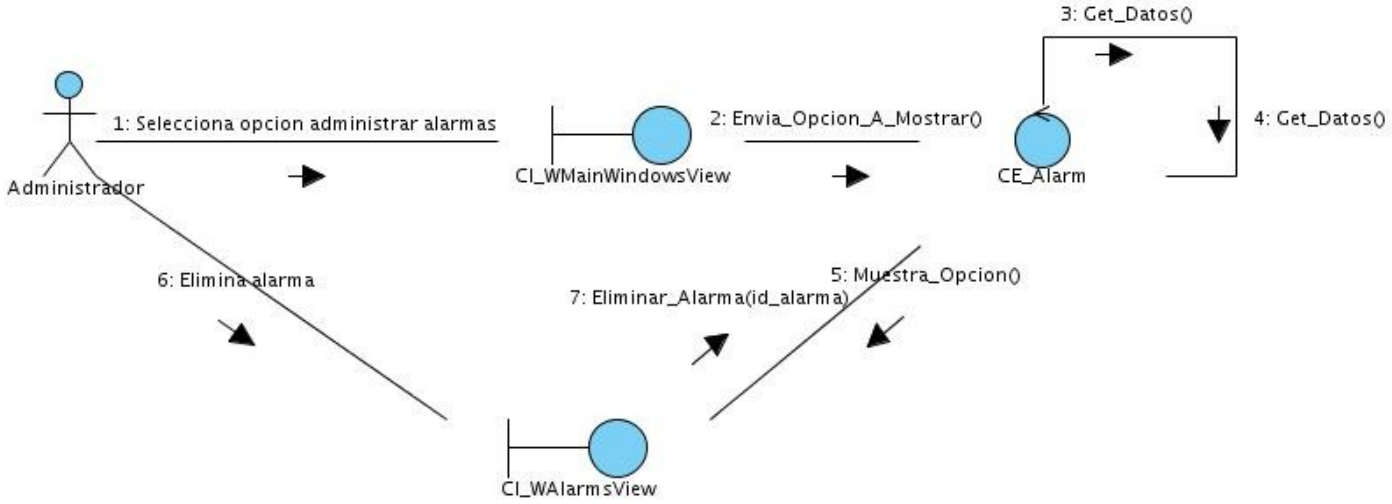


Figura 34 Diagrama Colaboración del Análisis CU Admin alarmas

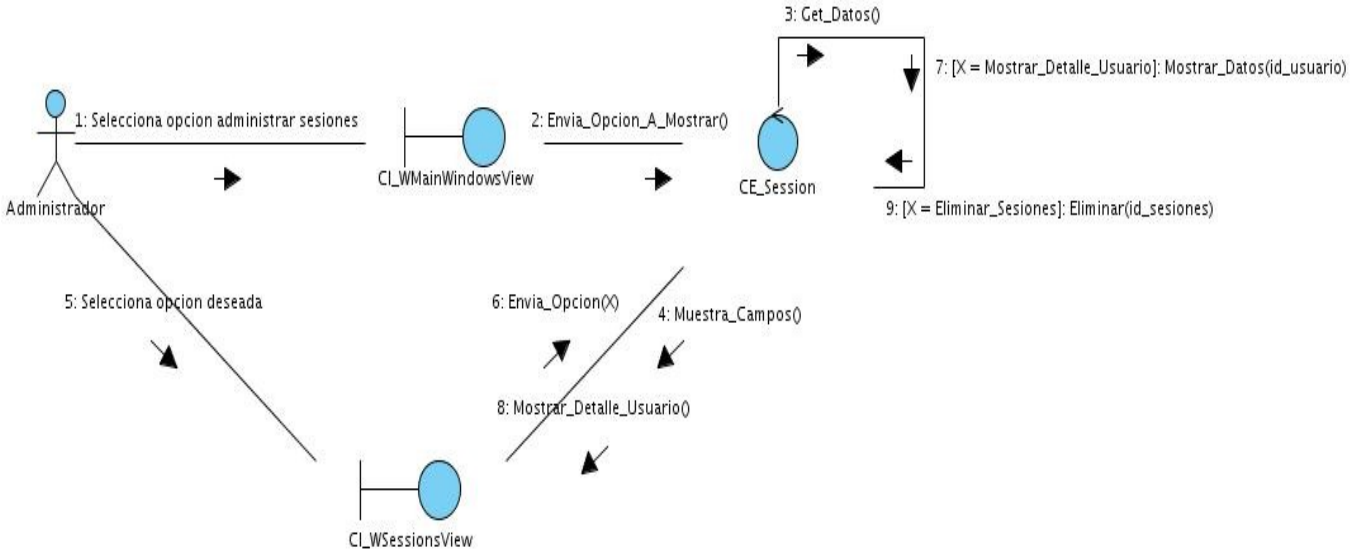


Figura 35 Diagrama Colaboración del Análisis CU Admin sesiones

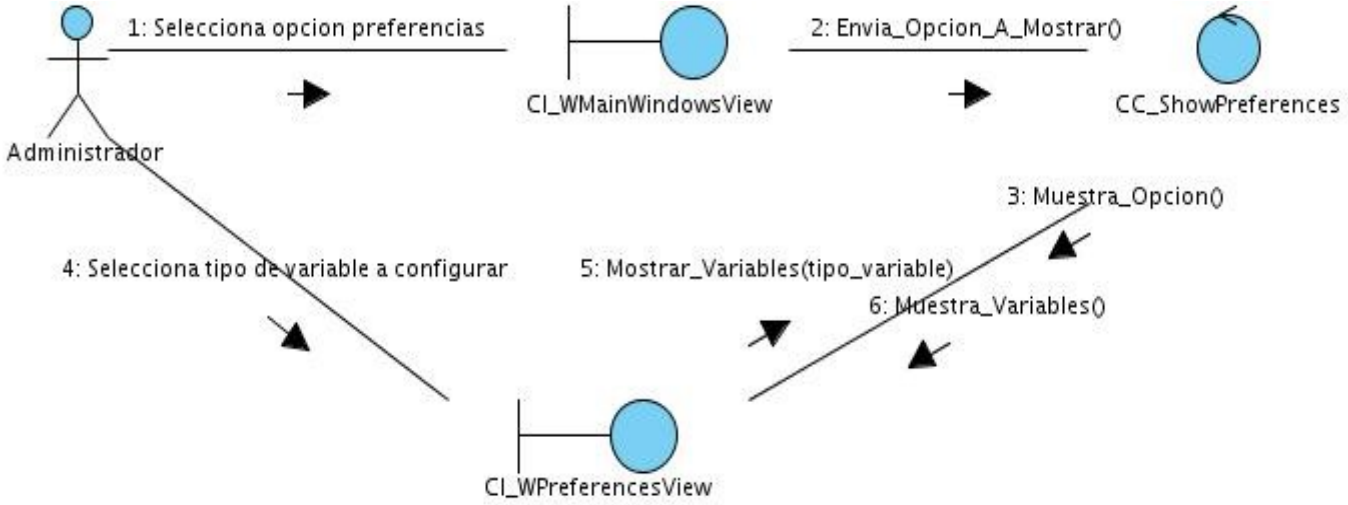


Figura 36 Diagrama Colaboración del Análisis CU Admin variables

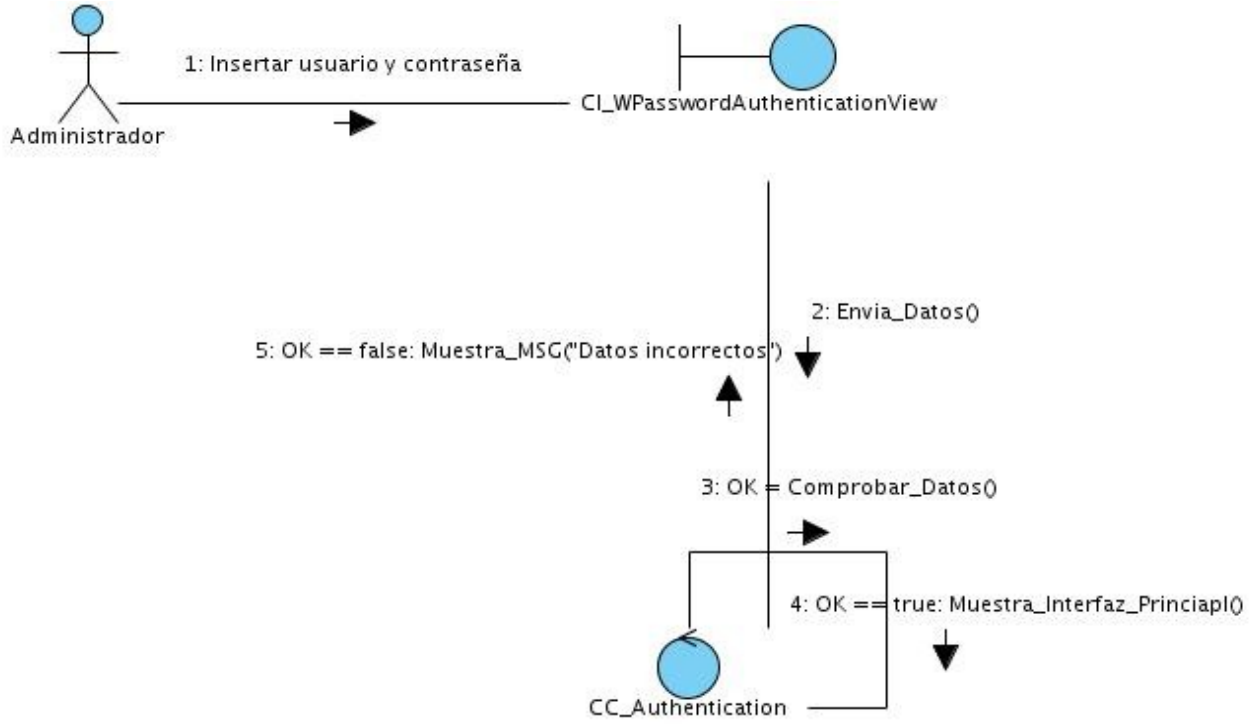


Figura 37 Diagrama Colaboración del Análisis CU Autenticar

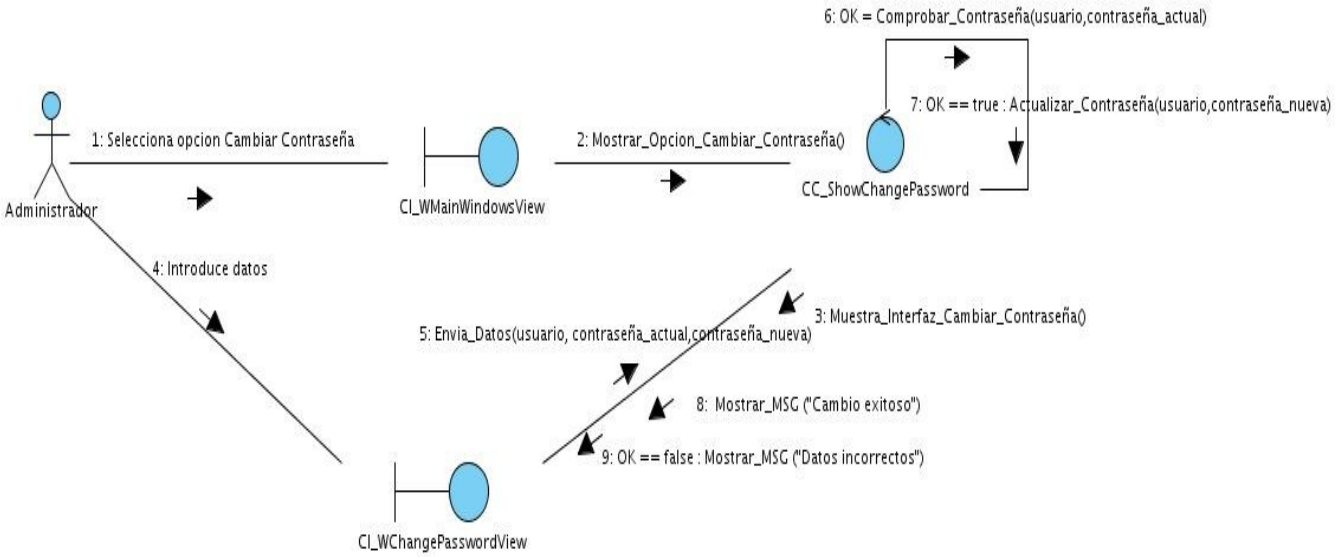


Figura 38 Diagrama Colaboración del Análisis CU Cambiar contraseña

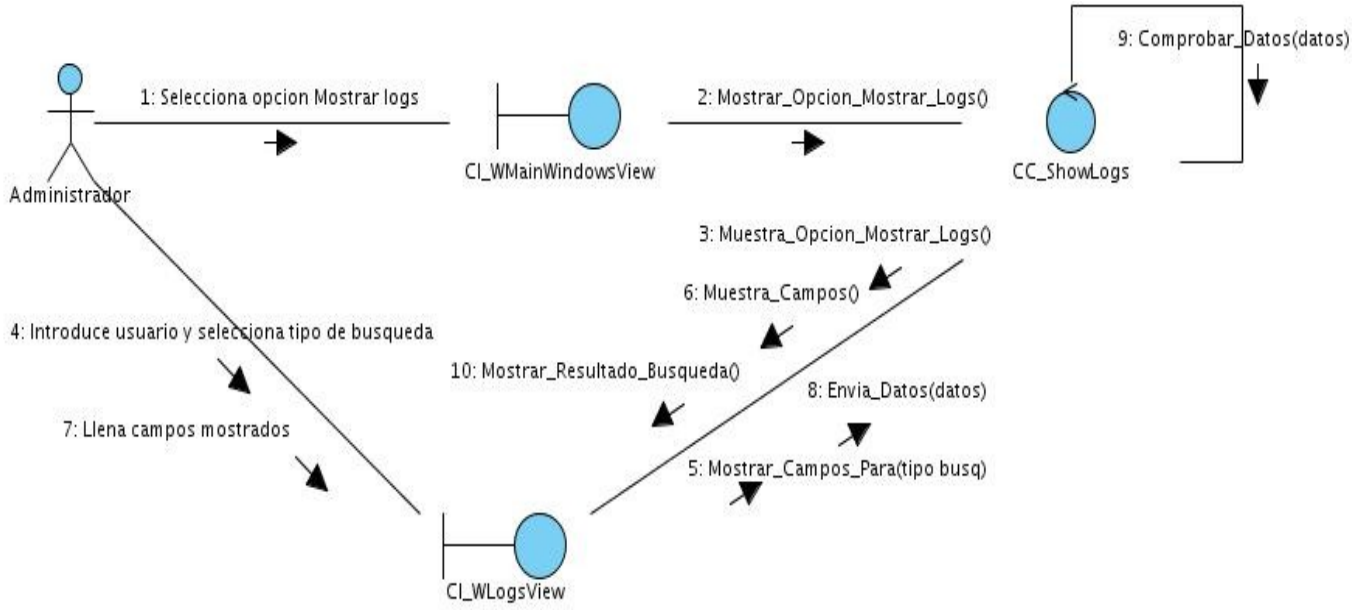


Figura 39 Diagrama Colaboración del Análisis CU Mostrar logs

Anexo 3 Diagramas de clases del diseño

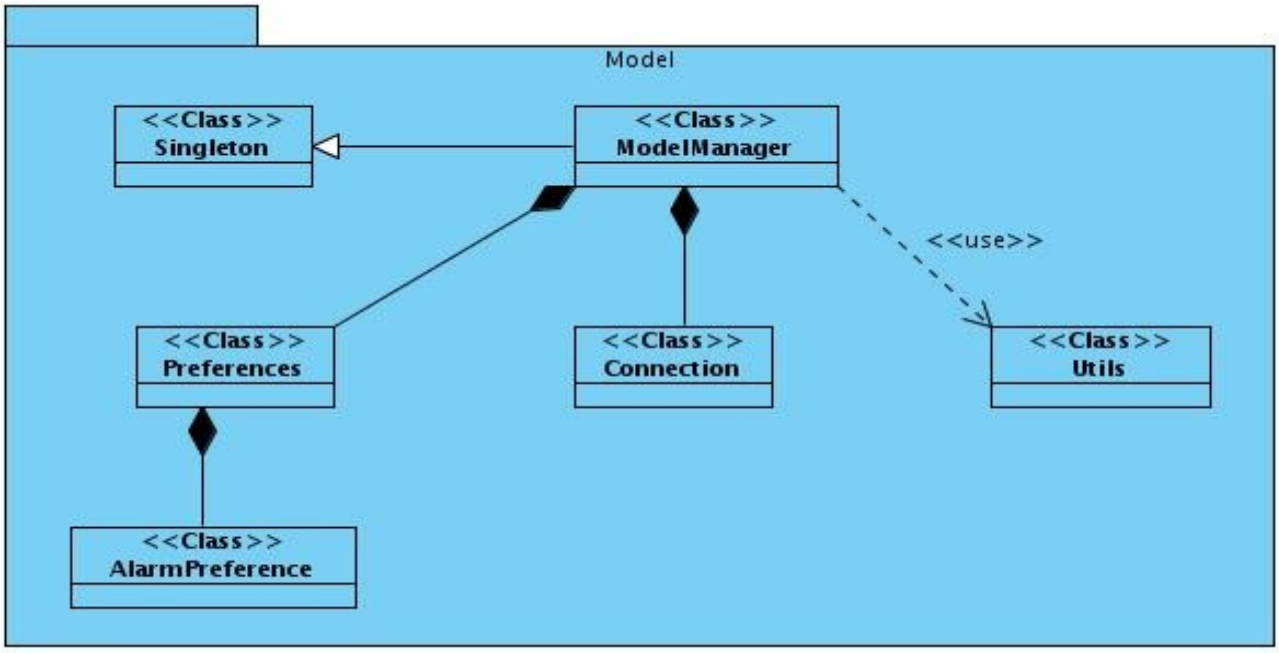


Figura 40 Diagrama Clases del Diseño Paquete Modelo

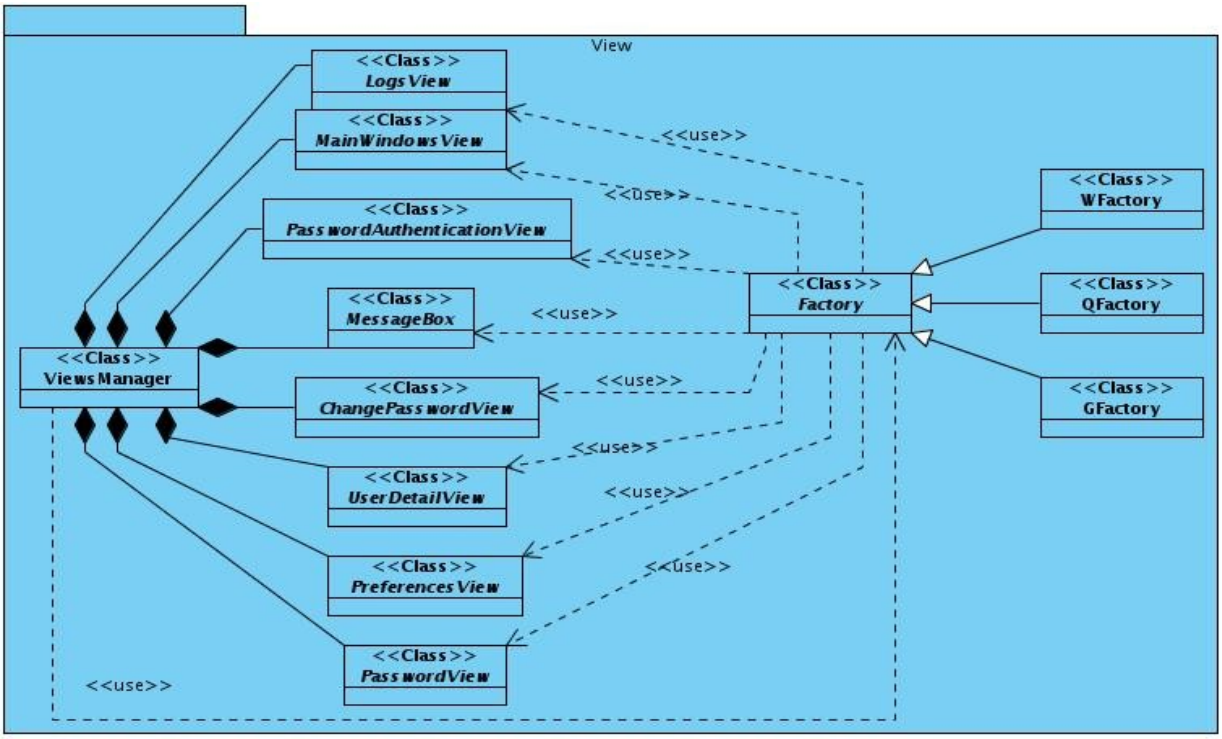


Figura 41 Diagrama Clases del Diseño Paquete Vista

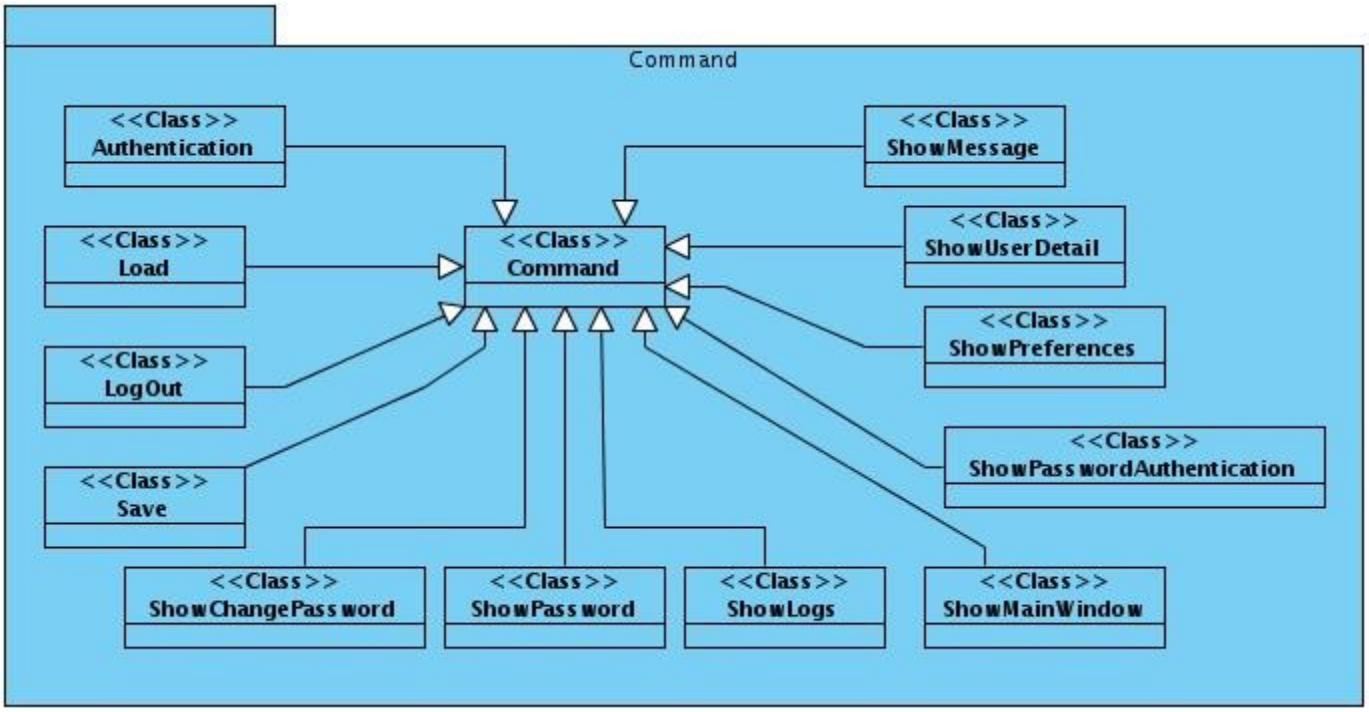


Figura 42 Diagrama Clases del Diseño Paquete Comando

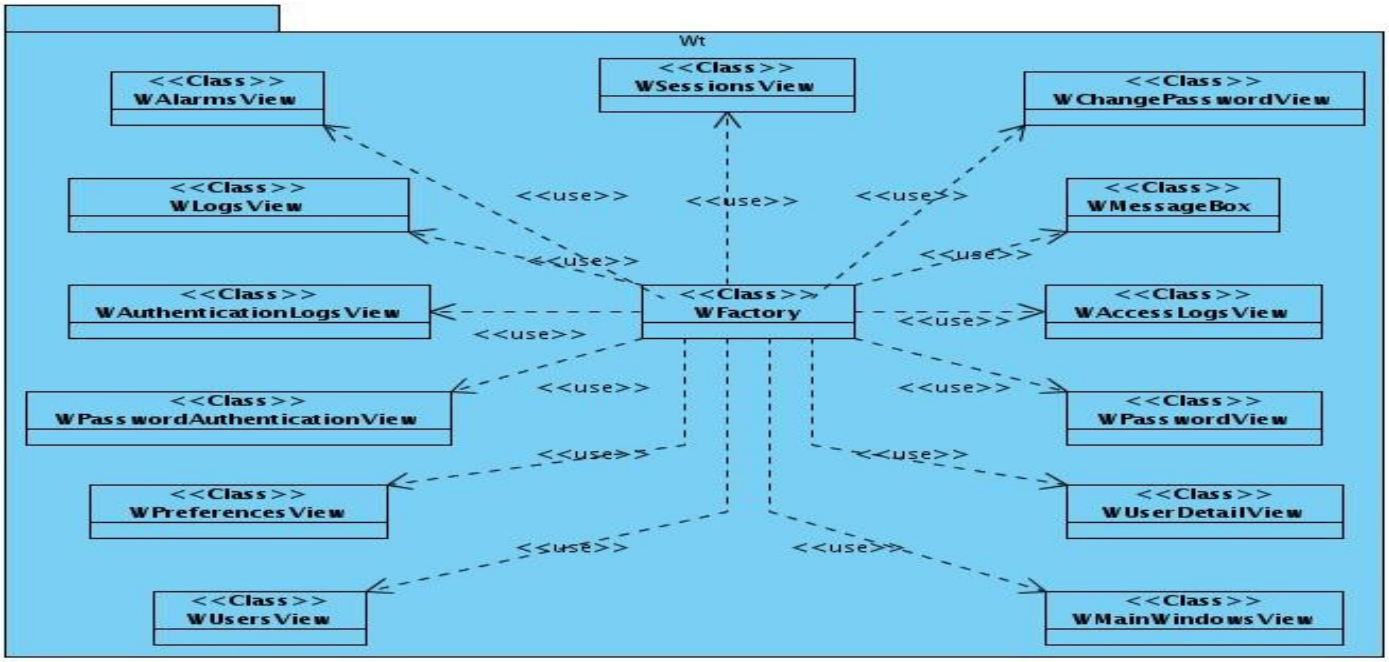


Figura 43 Diagrama Clases del Diseño Paquete Wt

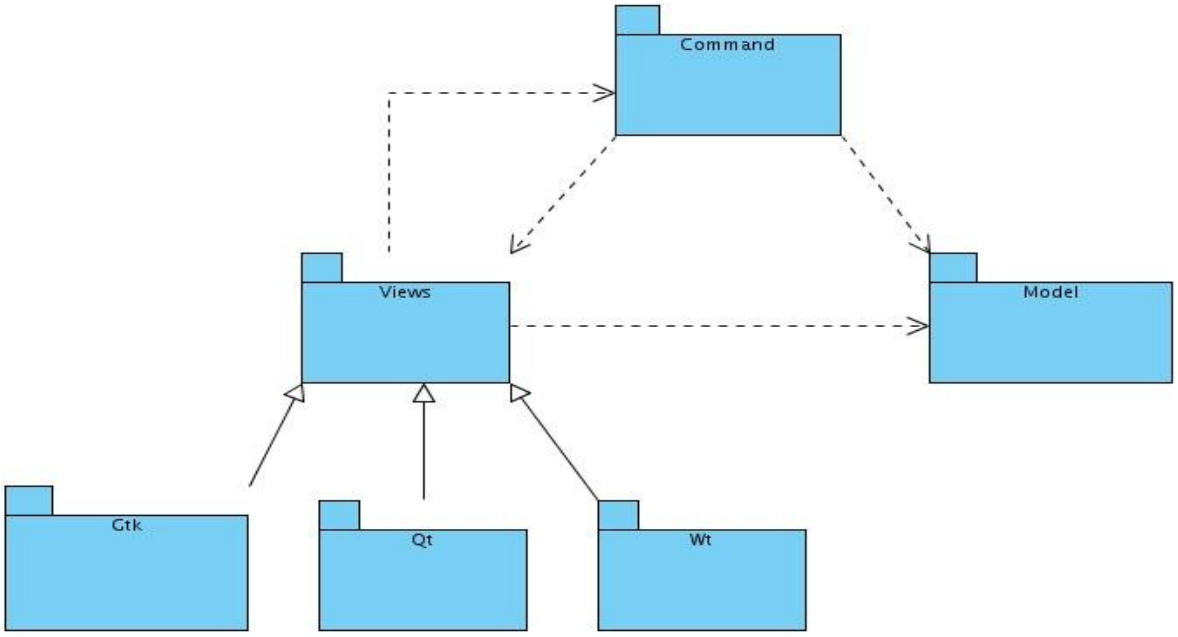


Figura 44 Diagrama Clases del Diseño por paquetes

Anexo 4 Diagramas de secuencia del diseño

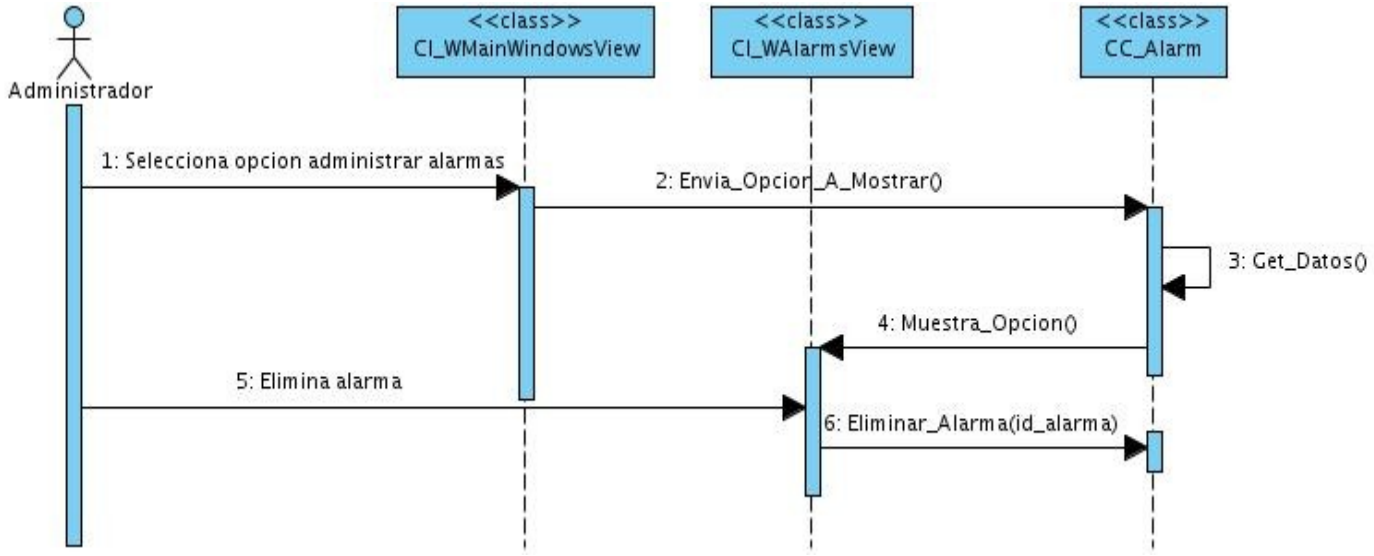


Figura 45 Diagrama Secuencia del Diseño CU Admin alarmas

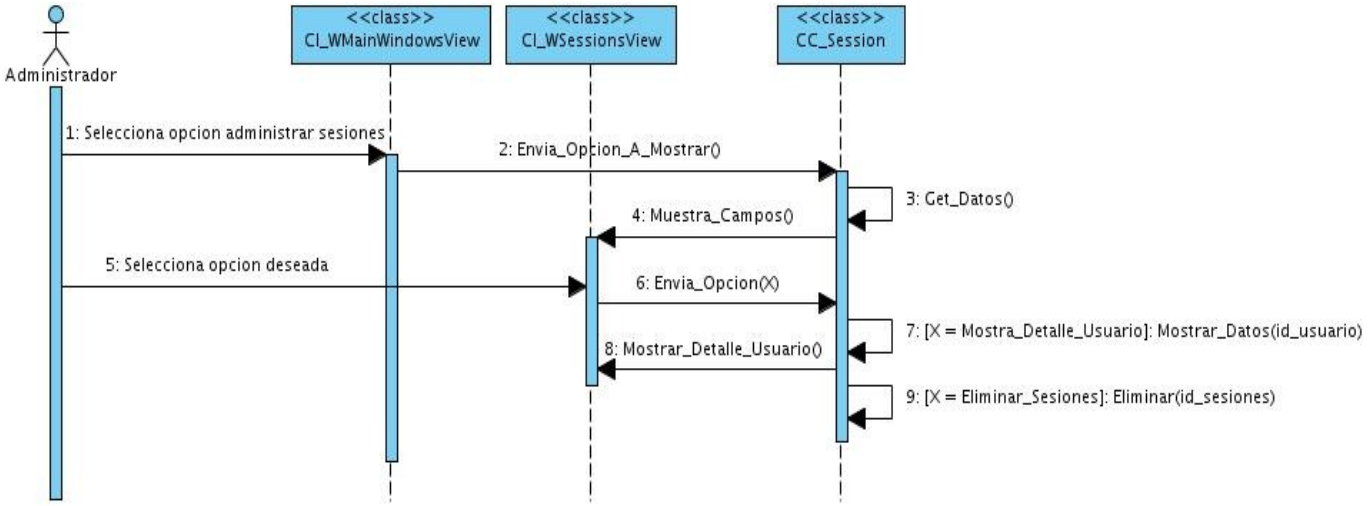


Figura 46 Diagrama Secuencia del Diseño CU Admin sesiones

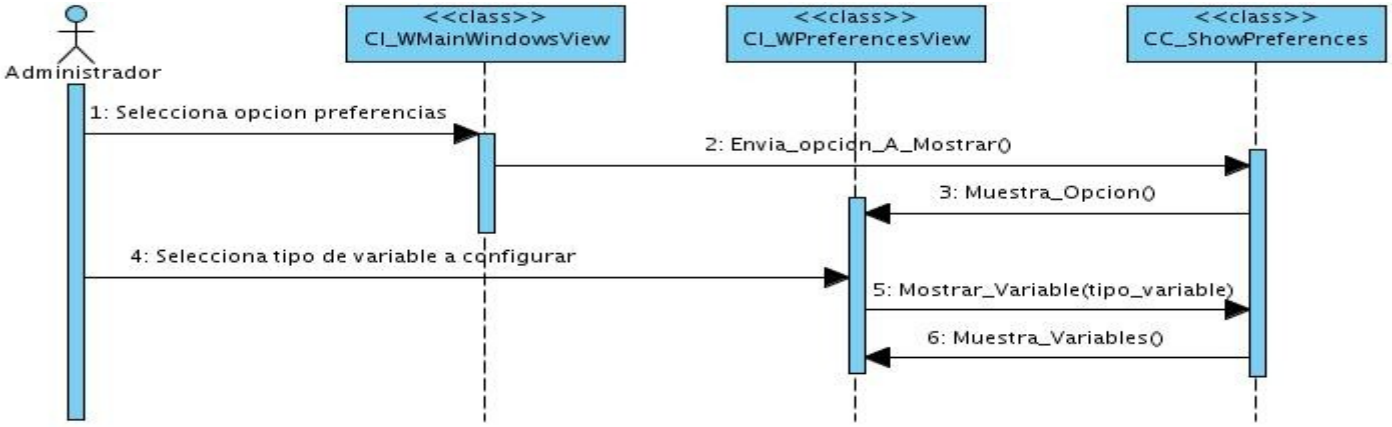


Figura 47 Diagrama Secuencia del Diseño CU Admin variables

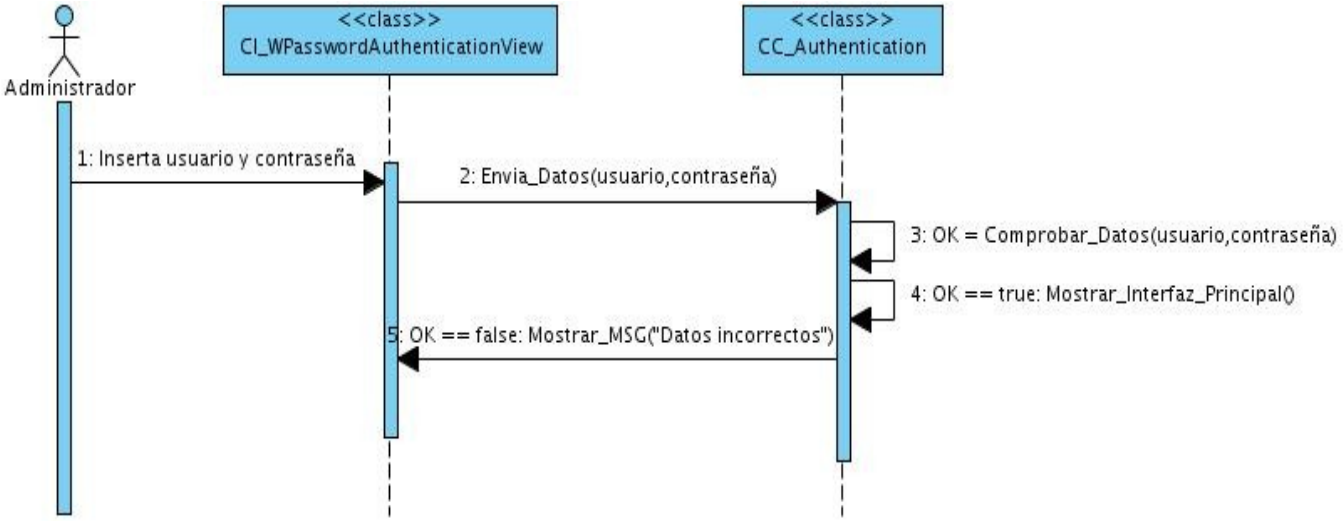


Figura 48 Diagrama Secuencia del Diseño CU Autenticar

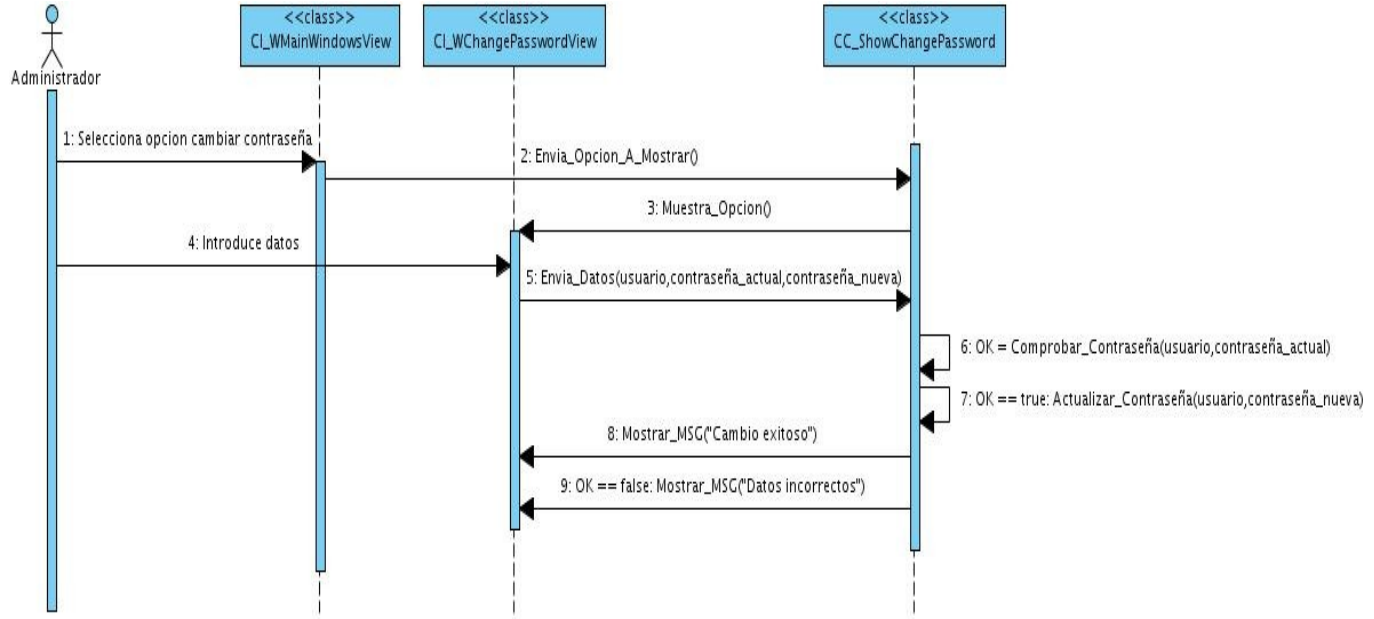


Figura 49 Diagrama Secuencia del Diseño CU Cambiar contraseña

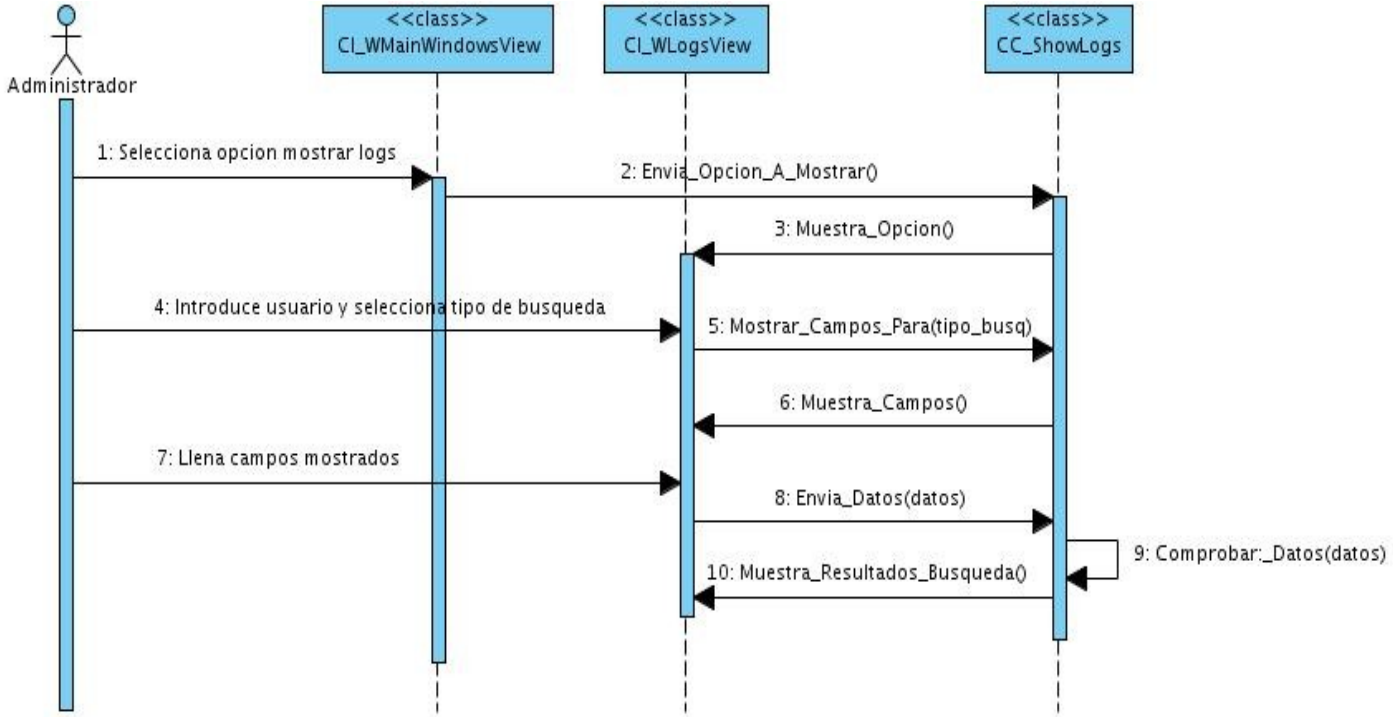


Figura 50 Diagrama Secuencia del Diseño CU Mostrar logs

Anexo 5 Diagrama de componentes por paquetes

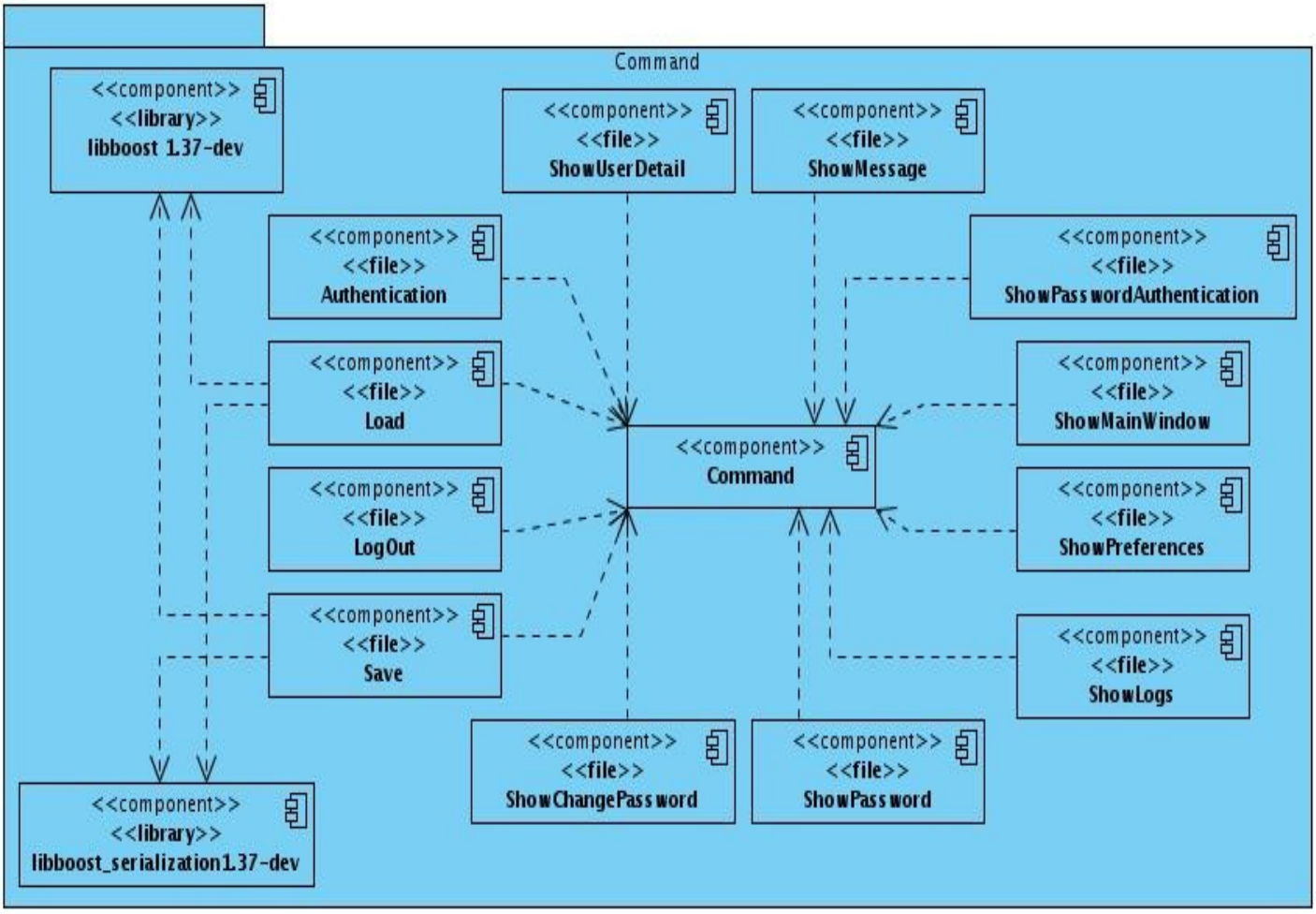


Figura 51 Diagrama Componentes Paquete Command

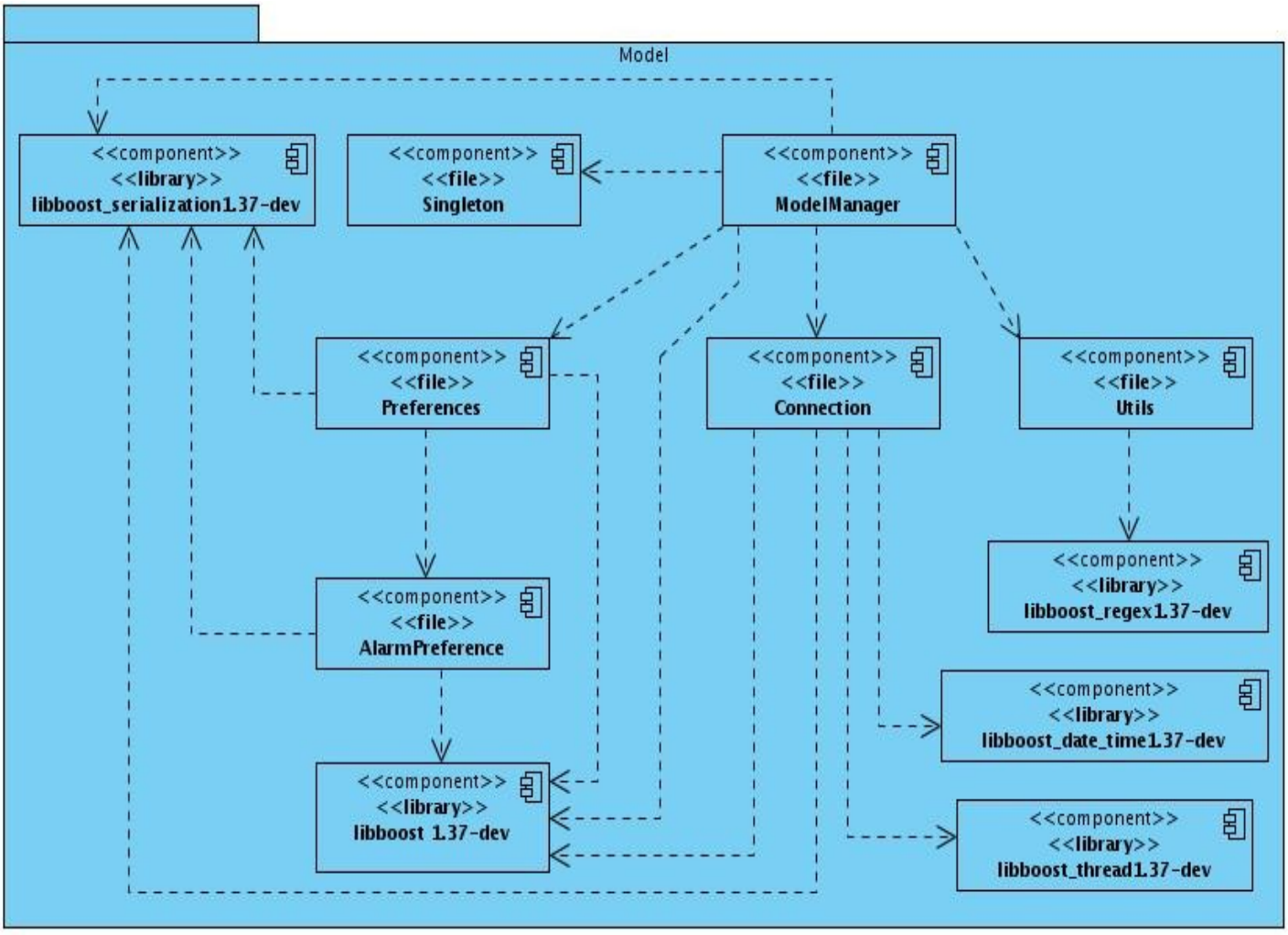


Figura 52 Diagrama Componentes Paquete Model

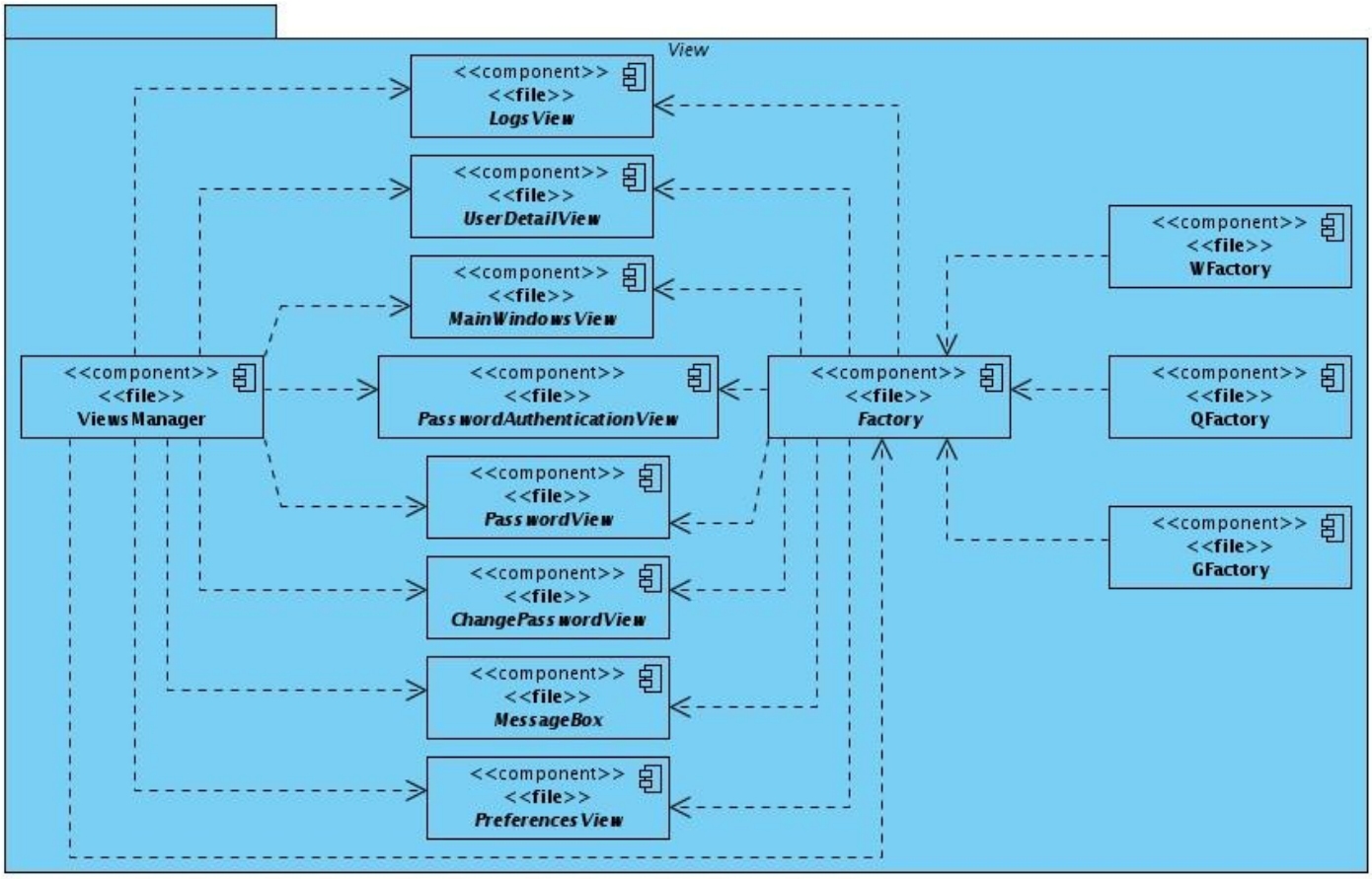


Figura 53 Diagrama Componentes Paquete View

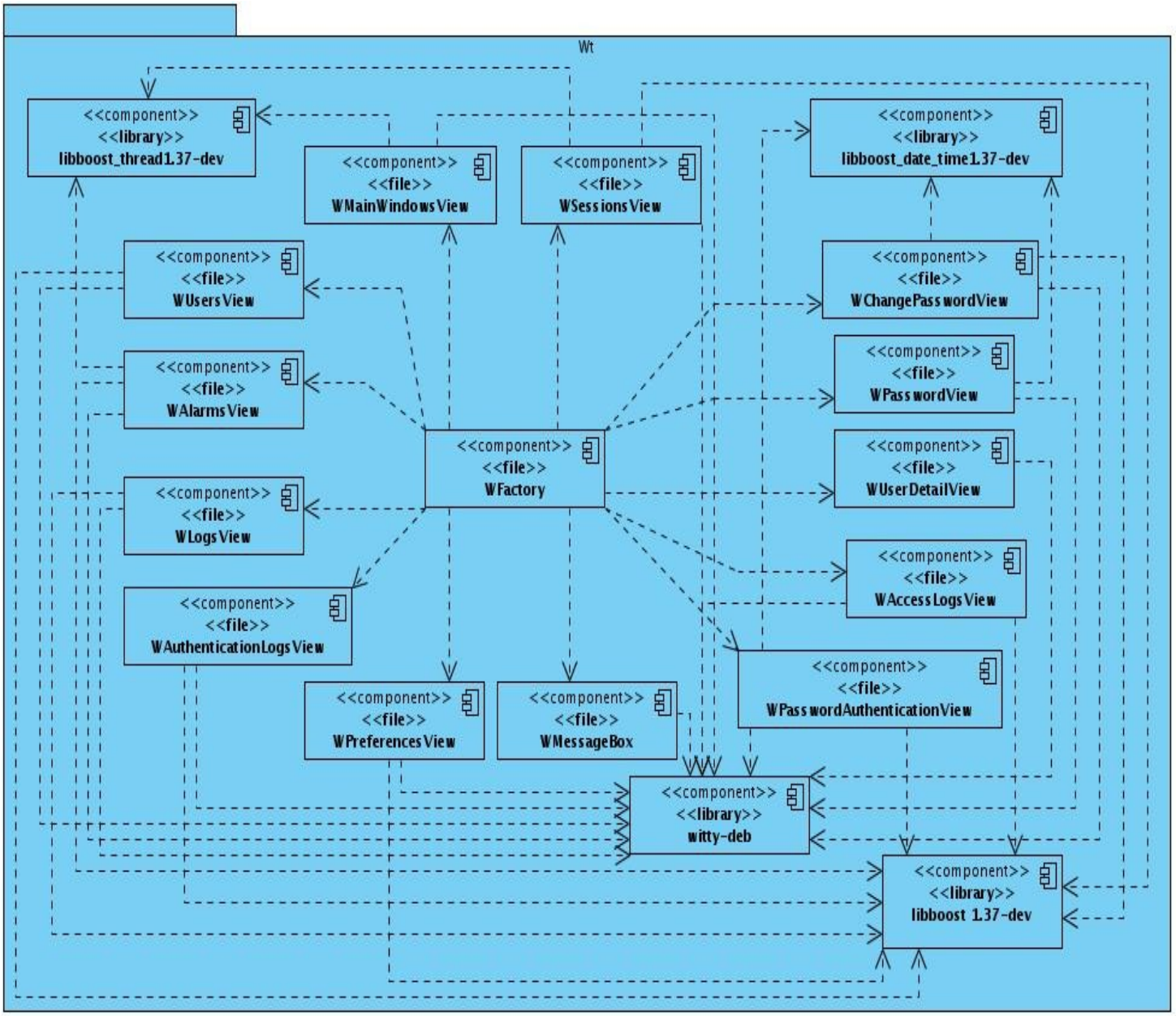


Figura 54 Diagrama Componentes Paquete Wt

Anexo 6 Casos de prueba

Figura 55 Caso de prueba 1

Figura 56 Caso de prueba 2

Figura 57 Caso de prueba 3

Figura 58 Caso de prueba 4

Figura 59 Caso de prueba 5

Figura 60 Caso de prueba 6

Figura 61 Caso de prueba 7

Figura 62 Caso de prueba 8

Figura 63 Caso de prueba 9

Glosario de Términos

Online: (En línea) El concepto se utiliza en el ámbito de la informática para nombrar a algo que está conectado o a alguien que está haciendo uso de una red, generalmente Internet**.**

Password: Serie secreta de caracteres que permite a un usuario tener acceso a un archivo, a un ordenador, o a un programa. En sistemas multiusos, cada usuario debe incorporar su contraseña antes de que el ordenador responda a los comandos.

Gtk: Es una biblioteca del equipo GTK+, la cual contiene los objetos y funciones para crear la interfaz gráfica de usuario. Maneja widgets como ventanas, botones, menús, etiquetas, deslizadores, pestañas, etc.

Hackers: Término para designar a alguien con talento, conocimiento, inteligencia e ingenuidad, especialmente relacionada con las operaciones de computadora, redes, seguridad, etc.

Driver: Software o programa que nos sirve de intermediario entre un dispositivo de hardware y el sistema operativo.

Logs: Registro oficial de eventos durante un periodo de tiempo en particular. Usado para registrar datos o información sobre quién, que, cuando, donde y porque un evento ocurre para un dispositivo en particular o aplicación, registra movimientos y actividades de un determinado programa.

XML: (Extensible Markup Language) Conjunto de reglas para definir etiquetas semánticas que nos organizan un documento en diferentes partes. Metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados.

XP: Metodología más destacada dentro de los procesos ágiles de desarrollo de software, utilizada para el desarrollo de proyectos cortos.

RUP: (Proceso Unificado de desarrollo de Software) Es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

UML: Lenguaje Unificado de Modelado, por sus siglas en inglés, Unified Modeling Language.

Herramientas CASE: Conjunto de programas y ayuda que dan asistencia a los analistas, ingenieros de software y desarrolladores durante todos los pasos del ciclo de vida de desarrollo de un software.

Visual Paradigm: Herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue.

Java: Lenguaje de programación.

Plug-in: Módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

IDE: Entorno de programación que ha sido empaquetado como un programa de aplicación.

Eclipse: Plataforma de programación, usada para crear entornos integrados de desarrollo (del Inglés IDE)

XMI: Estándar para el intercambio de meta modelos usando XML.

RSA: Herramienta de diseño y desarrollo integrados que potencia el desarrollo orientado al modelado con UML para la creación de aplicaciones y servicios con buena arquitectura.

Stakeholders: Interesados directos e indirectos de una empresa que teniendo algún tipo de interés en las operaciones empresariales, le brindan su apoyo y ante los cuales la organización es responsable.

C++: Lenguaje de programación hibrido.

Framework: Esquema (un esqueleto, un patrón) para el desarrollo y/o la implementación de una aplicación.

HTML: (Lenguaje de Marcado de Hipertexto) Es el lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

RAM: La memoria de acceso aleatorio (en inglés: random-access memory cuyo acrónimo es RAM) es la memoria desde donde el procesador recibe las instrucciones y guarda los resultados.

PHP: Lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting) pero

actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

Ajax: (Asynchronous JavaScript And XML) técnica de desarrollo web para crear aplicaciones interactivas.

Extend: Biblioteca Java Script del lado del cliente ligera y de alto rendimiento, compatible con la mayoría de navegadores que nos permite crear páginas e interfaces web dinámicas.

C: Lenguaje de programación orientado a la implementación de Sistemas Operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

Wt: Biblioteca en C++ y servidor de aplicaciones para desarrollar y desplegar aplicaciones web interactivas.

XHTML: Lenguaje de marcado predominante para la elaboración de páginas web.

JavaScript: Lenguaje de scripting basado en objetos no tipeado y liviano, utilizado para acceder a objetos en aplicaciones. Principalmente, se utiliza integrado en un navegador web permitiendo el desarrollo de interfaces de usuario mejoradas y páginas web dinámicas. Es un lenguaje basado en prototipos, con entrada dinámica y con funciones de primera clase.

CSS: Lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). Es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.

DHTML: Conjunto de técnicas que permiten crear sitios web interactivos utilizando una combinación de lenguaje HTML estático, un lenguaje interpretado en el lado del cliente (como JavaScript), el lenguaje de hojas de estilo en cascada (CSS) y la jerarquía de objetos de un DOM.

API: Conjunto de convenciones internacionales que definen cómo debe invocarse una determinada función de un programa desde una aplicación.

Widgets: Pequeña aplicación o programa, usualmente presentado en archivos o ficheros pequeños que son ejecutados por un motor de widgets o Widget Engine.

GNU/Linux: Término empleado para referirse a la combinación del núcleo o kernel libre similar a Unix denominado **Linux**, que es usado con herramientas de sistema GNU.

Debian: Comunidad conformada por desarrolladores y usuarios, que mantiene un sistema operativo GNU basado en software libre. El sistema se encuentra precompilado, empaquetado y en un formato sencillo para múltiples arquitecturas de computador y para varios núcleos.

Middleware: Software de conectividad que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas. Funciona como una capa de abstracción de software distribuida, que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo y red).

Scada: Aplicación de software especialmente diseñada para funcionar sobre computadores en el control de producción, proporcionando comunicación con los dispositivos de campo y controlando el proceso de forma automática desde la pantalla del operador, proveyendo toda la información asociada al proceso.