Universidad de las Ciencias Informáticas

Facultad 1

"Diseño e implementación de un módulo de administración para el Sistema Unificado de Gestión de la Fuerza de Trabajo Calificada"

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autores:

Yailin Molina González

Evelio Rafael Martínez Peña

Tutores:

Ing. Alexander Rodríguez Mompié

Ing. Carlos Tonet Groero Carmona

Ciudad de la Habana, junio 2010

"Año 52 de la Revolución"

"Seamos realistas y hagamos lo imposible."
Ernesto Che Guevara.

Dedicatoria:

Yailin

A:

Mi familia y mi gran amor, especialmente a mis padres y mi hermana.

Evelio

A:
Mis padres por tanto amor…
A mamita porque es única.
A mi especial familia y amigos.

Yailin

A mi familia por su apoyo incondicional a lo largo de estos cinco años.

A mis padres por toda su dedicación, comprensión, sacrificio y esmero, por confiar tanto en mí. Papi hoy tu sueño sí se cumple completo, te lo prometí el día 10 de octubre del 2005 cuando entré por primera vez a la escuela y hoy te lo estoy haciendo realidad. Mami ya ves que tanto sacrificio sí valió la pena, hoy tienes a tus dos hijas, como tanto quisiste, hechas profesionales. Gracias por guiarnos por un buen camino, por el ejemplo y educación que nos dieron. Los amo mucho y me siento muy orgullosa de ustedes.

A mi hermana, gracias por saber desempeñar tan bien tu papel de hermana mayor, por ayudarme, por quererme, por comprenderme, por el buen ejemplo que me diste, sin ti, esto no hubiera sido posible.

A mis segundos padres, Luiky y Mayi por tanta ayuda, dedicación y comprensión, nunca duden del cariño y el amor tan grande que les tengo.

A mis tías Adela, Pilar y Noris, por quererme mucho y brindarme su apoyo incondicional, en todo.

A mis tíos Gonzalo, Pupi y Robe, por estar pendiente de mí.

A mis abuelitos lindos y queridos, Isabel, Esther y Dionisio, los quiero mucho.

A mis primos Marlén, Yuli, Aidiel, Orly y Julio, gracias por estar tan pendiente de mí y brindarme su apoyo.

A mi amor por estar siempre a mi lado, apoyándome en todo, por demostrarme que se sí podía, aunque yo pensara lo contrario, por hacer que este sueño se cumpliera de forma más fácil, por la comprensión, dedicación y esmero, por hacerme feliz, te amo mi vida.

A mi suegra (Dulce), por ayudarme tanto y estar tan pendiente de mí.

A la Rubia por ayudarme tanto con el documento.

A mis compañeros de proyecto, especialmente Alain y Evelio.

A mis compañeros de cuarto por soportarme a lo largo de estos años, Elizabeth, María del Carmen, Pedro, Dayana, Yaima y Yordan.

A mis tutores Alexander y Carlos Tonet, por su ayuda y apoyo incondicional.

A Lorenzo y Alberto por darme fe y confianza en los momentos que más lo necesitaba, por ayudarme a que este sueño se hiciera realidad.

A todas las personas que de una forma u otra contribuyeron a que este sueño se hiciera realidad.

Por último, quisiera agradecerle a la revolución y al Compañero Fidel Castro por darme la posibilidad de estudiar en esta universidad tan maravillosa.

Evelio

Mi agradecimiento y amor infinito a mis padres que son mi vida y principal apoyo, sin ellos no hubiera llegado tan lejos.

A la Revolución, por permitirme alcanzar mis sueños de ser un profesional.

A mis tutores, Carlos Tonet y Alexander R. Mompie por su apoyo siempre necesario y acertado.

A mis compañeros de proyecto que tanto me ayudaron, en especial a Yailin y Alain.

A Maye por estar a mi lado en cada momento.

A todos esos amigos que he ido conociendo durante estos cinco años y me han dado su ayuda y confiado su amistad.

A Argel y Eloísa que me guiaron hacia esta carrera y que me han ayudado, los considero como mi familia.

A mi hermana y mi sobrina.

A mi abuela Luisa, y a mis abuelos que ya no están.

A mis tíos Magdalena, Mirtha, Ana, Omar, Rafael, Rodolfo.

A mis primos todos, especialmente a Ñico, Marcia y Mayelin que siempre me han acompañado.

A mis vecinos y amigos, especialmente a Blanca.

A todos, los que de una forma u otra me han apoyado tanto en todo.

Declaración de autoría

Declaramos que somos los únicos autores del trabajo titulado: **Diseño e Implementación del Módulo Administración para el Sistema Unificado de Gestión de Fuerza de Trabajo Calificada "GeForza"**, y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo, con carácter exclusivo.

Para que así conste firmamos la presente a los ___ días del mes de Junio del año 2010.

Yailin Molina González

Evelio Rafael Martínez Peña

Ing. Carlos Tonet Groero Carmona

Ing. Alexander Rodríguez Mompié

Resumen:

El presente trabajo titulado: Diseño e implementación de un módulo de administración para el Sistema Unificado de Gestión de la Fuerza de Trabajo Calificada ”GeForza”, pretende alcanzar una homogeneización de los clasificadores utilizados por los diferentes subsistemas del departamento Fuerza de Trabajo Calificada (FTC) del Ministerio de Economía y Planificación (MEP) y lograr la seguridad del sistema. Para el desarrollo del sistema, se tuvo en cuenta las necesidades reales del departamento, se estudiaron con profundidad las aplicaciones con las que se gestionaba la información en el mismo. Para que este fuera lo más eficiente posible, se realizó un amplio estudio de los procesos implicados en el tema, con el objetivo de garantizar el buen funcionamiento del sistema. A la hora de seleccionar las herramientas, metodología, lenguajes y tecnologías se tuvo en cuenta la política del país y la universidad de emigrar a software libre. Se modelaron los procesos involucrados en el negocio a través de un modelado de dominio, se realizó un correcto levantamiento de requisitos, un modelado del sistema, así como un adecuado análisis, diseño e implementación del mismo, finalmente se hicieron pruebas para verificar la calidad, seguridad, confiabilidad, y disponibilidad, con el objetivo de obtener la aceptación del cliente. Obteniendo un Módulo de Administración que garantiza la homogeneización de los clasificadores utilizados por los diferentes subsistemas del departamento FTC y la seguridad del sistema “GeForza”.

Palabras clave:

Clasificadores, homogeneización, administración.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Desde tiempos remotos el hombre siempre ha buscado la forma de optimizar su forma de trabajar, hacer la mayor cantidad de operaciones en el menor tiempo posible. Al paso de los años ha ido renovando su forma de pensar y de protegerse, esto lo ha obligado a superarse cada día más.

Después del triunfo de la revolución en Cuba, la máxima dirección del país puso gran empeño en la superación integral del hombre, llevando a cabo una serie de tareas y poniendo muchos recursos para que esto fuera posible, como por ejemplo: la campaña de alfabetización, la creación de los Joven Club de Computación, bibliotecas comunitarias, casas de cultura comunitarias, cursos de superación para jóvenes las Sede Universitarias Municipales, entre otras.

En las últimas décadas, con el gran adelanto de la informatización en todas las esferas de la sociedad y el perfeccionamiento acelerado de las Tecnologías de la Información y las Comunicaciones (TIC), la mayoría de las empresas e instituciones necesitan informatizar los procesos de trabajo que desarrollan para ganar competitividad, eficiencia y tiempo. Por esta razón uno de los objetivos estratégicos del Ministerio de Economía y Planificación (MEP) es informatizar todos sus procesos.

El Ministerio de Economía y Planificación (MEP) es el Organismo encargado de planificar, ejecutar, dirigir y controlar la aplicación de la Política del Estado y el Gobierno en materia de Economía, Planificación, Estadística y Diseño , además de las funciones comunes de todos los Organismos de Administración Central del Estado (OACE), tiene sus funciones específicas para ello.

En el MEP, existe una dirección encargada del Desarrollo Social de Cuba, que entre otras tareas es responsable de planificar y controlar el personal calificado entrante y saliente de los diferentes niveles de enseñanza, que luego de prepararse en una materia van a ocupar una plaza en un centro de trabajo. El departamento Fuerza de Trabajo Calificada (FTC) pertenece a esta dirección, y es el encargado de lo antes mencionado, es decir, de lograr el máximo equilibrio entre la formación y el empleo de los estudiantes de la Educación Superior, Técnica y Profesional, acorde a las reales necesidades de los territorios y el país, atendiendo las indicaciones del nivel superior, en relación con esta temática.

El MEP, cuenta con un pequeño grupo de informáticos que desarrollan software para informatizar todos los procesos que se llevan a cabo en el ministerio, trayendo como consecuencia que muchos de sus departamentos cuenten con software implementados en herramientas antiguas, dificultando la eficiencia de los procesos y la interacción de los trabajadores con dichos software, el efecto de esta situación es el retraso o la inexactitud de la información, y la ineficiencia de las entidades.

Estos factores lo tenían en común la mayoría de las entidades que tienen la suerte de estar informatizadas en Cuba, pues no todas, han caminado por el sendero hacia el desarrollo informático. Los altos dirigentes del país se percataron, por los estudios realizados por el departamento de FTC de la necesidad que existía de formar informáticos, por esta razón el compañero Fidel Castro Ruz, en aquel entonces, Presidente de Cuba, del Consejo de Estado y de Ministros, decide fundar la Universidad de las Ciencias Informáticas (UCI).

La UCI surge en el 2002 con la misión de formar profesionales altamente calificados, producir software y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación. Como centro de producción de software desempeña un papel significativo, ya sea por la idea de construir un modelo de ciudad digital, como por el gran número de proyectos que desde la Universidad contribuyen a la informatización del país.

El departamento FTC, al percatarse que la realización de sus tareas no se estaba realizando de forma eficiente, contacta con la UCI para la realización de un sistema que permita realizar su trabajo de forma rápida y correcta.

El equipo de desarrollo luego de realizar un estudio minucioso de los sistemas con los que se estaban llevando a cabo las tareas en el departamento de FTC, se plantea la siguiente **situación problémica**:

- ✓ Carencia de un módulo que permita la homogeneización de los clasificadores utilizados por los diferentes sistemas utilizados en el departamento de FTC.
- ✓ Ausencia total de la seguridad básica en los sistemas.
- ✓ Desactualización de la información utilizada por los sistemas del departamento de FTC.
- ✓ Utilización de sistemas implementados en software privativos, con herramientas y lenguajes caducos.

Luego de un profundo análisis de las necesidades reales existentes y con el fin de solucionar la situación anterior, se plantea el siguiente **problema científico**:

Necesidad de implementar el módulo de administración para el Sistema Unificado de Gestión de la Fuerza de Trabajo Calificada “GeForza”, que logre la homogeneidad de los clasificadores utilizados en el departamento Fuerza de Trabajo Calificado (FTC) del Ministerio de Economía y Planificación (MEP).

Por lo tanto, el **objetivo general** del trabajo consiste en: Desarrollar el módulo de administración para el Sistema Unificado de Gestión de la Fuerza de Trabajo Calificada “GeForza”.

A partir del análisis del objetivo general se identificaron los siguientes **objetivos específicos:**

- ✓ Diseñar el módulo de administración para el Sistema Unificado de Gestión de la Fuerza de Trabajo Calificada “GeForza”.

- ✓ Implementar el módulo de administración para el Sistema Unificado de Gestión de la Fuerza de Trabajo Calificada “GeForza”.
- ✓ Implementar funcionalidades que permitan conocer las acciones ejecutadas sobre la base datos.
- ✓ Realizar el estudio de factibilidad del sistema.
- ✓ Realizar pruebas de calidad al sistema después de concluida su implementación.

Para alcanzar una solución satisfactoria que elimine total o parcialmente el problema planteado, se necesita realizar un estudio exhaustivo del **objeto de estudio**:

El proceso de integración y manipulación de la información gestionada por el departamento de FTC del MEP.

En síntesis, el **campo de acción** que abarca la investigación es: *Módulo de administración del sistema GeForza.*

Para dar cumplimiento a los objetivos planteados anteriormente, se trazaron una serie de **tareas de investigación:**

- ✓ Realizar un estudio del estado del arte de sistemas de administración
- ✓ Estudiar la Metodología RUP en sus distintas fases. Definir requisitos funcionales del sistema.
- ✓ Definir requisitos no funcionales del sistema.
- ✓ Estudiar las herramientas a utilizar para el desarrollo del proyecto
- ✓ Estudiar e investigar las funcionalidades e importancia de la arquitectura MVC que proponen los marcos de trabajo ExtJS y Zend Framework.
- ✓ Proponer un módulo de administración que cumpla con las necesidades del sistema GeForza. Diseñar el módulo de administración a partir del levantamiento de requisitos funcionales realizado.
- ✓ Estudiar e investigar sobre los marcos de trabajo ExtJS y Zend Framework.
- ✓ Implementar mediante herramientas libres un módulo de administración para GeForza.
- ✓ Realizar un estudio de factibilidad del sistema.

Hipótesis

“Con la implementación del módulo de administración para el Sistema Unificado de Gestión de la Fuerza de Trabajo Calificada “GeForza”, se logrará una homogeneización de los clasificadores utilizados por los diferentes sistemas del departamento Fuerza de Trabajo Calificada (FTC) del Ministerio de Economía y Planificación (MEP).”

Variables:

Variable independiente: Módulo de administración.

Variable dependiente: Homogeneización de los clasificadores.

La operacionalización de las variables se podrá encontrar en el anexo # 1.

En el desarrollo de la presente investigación se utilizarán un conjunto de **métodos científicos** que servirán de guía y facilitarán un mejor entendimiento de lo que está sucediendo, como es el caso de los métodos empíricos y los teóricos. A continuación se explica en detalles el por qué de su selección.

Dentro de los métodos teóricos se encuentran:

Histórico-Lógico: Se realizará un estudio de los sistemas actúales que se utilizan en el departamento de FTC para definir los principales errores y deficiencias en vista de perfeccionar las vulnerabilidades.

Modelación: El mismo se pone en práctica en el trabajo al realizar el análisis de la realidad, mediante diversos modelos y diagramas que ayudan a comprender mucho más el objeto en su totalidad.

Analítico-Sintético: Posibilitó el análisis del proceso de administración para determinar con exactitud cómo este deberá funcionar. Como resultado, se toman todas las características principales para lograr modelar un sistema que logre una integración eficaz y una armonía de los nomencladores.

Los **métodos empíricos** utilizados para obtener información sobre el objeto de estudio son:

Observación: Se utiliza para investigar los procesos externamente sin tener que llegar a la esencia de los mismos, lo que ayudó al planteamiento del problema científico, además de permitir conocer bien el proceso delimitado como objeto de estudio, lo cual ayuda a tener un conocimiento más detallado de lo que se quiere, lo que hace falta hacer y cómo hay que hacerlo.

Entrevista: Se realizarán entrevistas a cada uno de los integrantes del departamento Fuerza de Trabajo Calificada del Ministerio de Economía y Planificación (MEP), con el objetivo de conocer las principales fallas del sistema actual así como las nuevas funcionalidades que requieren.

Estudio Documental: Se realizará un estudio a partir de los documentos entregados por el cliente y otros que sean necesarios para estimular la autosuperación en temas de interés para el desarrollo de la aplicación.

Posibles resultados: al finalizar el proceso de investigación y desarrollo se espera como resultado un Módulo de Administración para el Sistema Unificado de Gestión de la Fuerza de Trabajo Calificada “GeForza” y los artefactos generados en el proceso de diseño.

Este documento está estructurado por cinco capítulos, el contenido de los cuales se describe a continuación.

Capítulo 1: Fundamentación Teórica, se realiza un estudio de las tendencias actuales de los módulos de administración. Se detallan las tecnologías, metodologías, herramientas y lenguajes utilizados definidas por la dirección del proyecto y aprobadas por el cliente.

Capítulo 2: Descripción de la solución propuesta, Se hace un estudio de los procesos del departamento de FTC, se modelan dichos procesos, se proponen mejorías a los procesos del cliente, se definen los requerimientos funcionales y no funcionales del software, estos se agrupan en casos de uso y se hace la descripción de los mismos.

Capítulo 3: Construcción de la solución propuesta, se modelan los diagramas de clases del análisis y del diseño. Se hace referencia a los patrones utilizados para la construcción del diseño. Se muestra el modelo lógico y físico de datos, así como un estudio de la factibilidad del sistema.

Capítulo 4: Implementación y prueba, se presentan todo lo relacionado con la implementación del sistema, diagrama de despliegue, diagrama de componentes, los estándares utilizados en la implementación y los resultados de las pruebas realizadas al sistema.

Además, el trabajo incluye Resumen, Conclusiones, Recomendaciones, Referencias bibliográficas, Bibliografía, Glosario de términos y Anexos.

CAPÍTULO 1

FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En el siguiente capítulo se realiza un estudio del arte de los diferentes módulos administrativos que existen en la actualidad. Se describen además las características del lenguaje de programación, sistema gestor de base de datos, metodología de desarrollo y herramientas propuestas para dar solución al problema.

1.2. Definiciones

¿Qué es la administración?

"Administración es el proceso de planear, organizar, dirigir y controlar el uso de recursos para lograr objetivos". (CHIAVENATO, 1999)

¿Qué es un sistema?

"Un sistema es una reunión o conjunto de elementos relacionados". (GICH, 1998)

¿Qué es información?

"Los datos están constituidos por los registros de los hechos, acontecimientos, transacciones, etc. Por el contrario, la información implica que los datos estén procesados de tal manera que resulten útiles o significativos para el receptor de los mismos, por lo que en cierto modo, los datos se pueden considerar la materia prima para obtener información" (Carrera, 2002)

¿Qué es un Sistema de Información?

"Un conjunto formal de procesos que, operando sobre una colección de datos estructurada según las necesidades de la empresa, recopilan elaboran y distribuyen la información (o parte de ella) necesaria para las operaciones de dicha empresa y para las actividades de dirección y control correspondientes (decisiones) desempeñar su actividad de acuerdo con su estrategia de negocio". (IBIDEM)

¿Administrador de un Sistema Informático?

El significado preciso varía. En las organizaciones con un sistema muy grande y complicado, generalmente dividen al personal informático según su especialidad. En este caso un administrador de un Sistema Informático es la persona que tiene la responsabilidad de asegurar el correcto funcionamiento y mantenimiento del mismo, o algún aspecto de éste.

¿Qué es un clasificador?

Proceso que permite una ordenación de elementos, según un determinado criterio.

1.3. Proceso de administración

Toda aplicación web que cuente con más de un subsistema debe tener implícito un Módulo o Sistema de administración, el cual es el encargado de la actualización, de la seguridad básica y homogeneización de esta aplicación entre otras funcionalidades. Generalmente estas aplicaciones tienen que cumplir con las siguientes responsabilidades:

- ✓ Realizar copias de seguridad.
- ✓ Instalar y configurar el nuevo hardware y software.
- ✓ Agregar, borrar y modificar información de las cuentas de usuarios, restablecer contraseñas, entre otras.
- ✓ Responder consultas técnicas.
- ✓ Responsable de la seguridad.
- ✓ Responsable de la configuración del sistema.
- ✓ Resolución de problemas.

Estos sistemas deben contar con un proceso de autenticación que es el encargado de verificar que alguien o algo es quien o lo que dice ser. En redes de equipos públicos y privados la autenticación se lleva a cabo comúnmente a través de contraseñas de inicio de sesión.

Este proceso se lleva a cabo de la siguiente manera. Todos los clientes son autenticados cuando se conectan a un servidor, después de eso el cliente es de confianza y a su vez cuando un cliente ejecuta alguna acción, ésta es chequeada, entra una política específica, donde cada uno de los controles y acciones son permitidos o denegados para ese usuario. Esta especificidad asume que el usuario está autenticado tanto tiempo como dure la conexión.

El proceso de autorización, es muy importante en estos sistemas ya que la autorización se realiza una vez que se ha autenticado al usuario. Este proceso consiste en determinar lo que el usuario puede y no puede hacer en una aplicación.

Para que estos sistemas sean seguros deben contar con el uso y las técnicas de criptografía. La criptografía es una tecnología importante para crear aplicaciones web seguras. Esta ciencia existe desde hace muchos años, aunque mucho antes de que se hubieran inventado los equipos informáticos. Sin embargo, con la llegada de estos últimos, la criptografía ha sido capaz de crear códigos prácticamente indescifrables.

Los algoritmos criptográficos se basan en un método matemático que se emplea para cifrar y descifrar un mensaje. Generalmente funciona empleando una o más claves (números o cadenas de caracteres) como parámetros del algoritmo, de modo que sean necesarias para recuperar el mensaje a partir de la versión cifrada, algunos de los algoritmos más representativos en la actualidad son:

a) Algoritmos simétricos de cifrado

- ✓ Cifrados de Producto
- ✓ El algoritmo DES (Estándar de cifrado de datos). Variantes
- ✓ El algoritmo IDEA (algoritmo internacional del cifrado de datos)
- ✓ Modos de Operación para Algoritmos Simétricos
- ✓ Criptoanálisis de Algoritmos Simétricos

b) Algoritmos asimétricos de cifrado

- ✓ El Algoritmo RSA (cifra asimétrica definida en la estándar de cifrado de datos)

La tecnología es hoy por hoy algo que cambia continuamente, y esta ayuda a los individuos a ser más productivos, y así pensar en organismos sociales que se unan para llegar a una meta en común y lograr objetivos previamente establecidos, para lograr unificarse y llegar a formar empresas, que necesitan ser parte de esta evolución tecnológica.

Sin embargo, cada vez más, es indispensable el uso de Control de Accesos para proteger cualquier tipo de información, ya sea personal o empresarial, ya que mucha de esta, es confidencial y debe ser protegida, para evitar que la información pueda ser robada o sufrir daños.

Por lo anterior, es importante que los administradores de sistemas tengan listas de control de acceso para proteger la información que es requerida. Es importante tener en cuenta que cada usuario debe tener una contraseña para acceder a dichos sistemas, la cual, le permitirá tener permisos para manejar información. Si un usuario, es cambiado de puesto, es seguro que a éste, le sean asignados nuevos permisos, y por tanto nueva contraseña, para lograr más seguridad.

Hoy en día existen diferentes tipos de controles de acceso en lo que se pueden situar:

MAC (Mandatory Access Control, Control de Acceso Obligatorio por sus siglas en inglés), DAC (Discretionary Access Control, Control de Acceso Discrecional por sus siglas en inglés) y RBAC (Role-based Access Control).

El control de acceso basado en roles, es una tecnología que llegó para satisfacer las principales necesidades, en cuanto a control de acceso se refiere.

Al inicio, los sistemas manejaron dos tipos diferentes de control de accesos, uno llamado DAC, en éste el usuario es quien decide como proteger el sistema, mediante controles de acceso impuestos por el sistema, y el otro MAC, en este tipo de control de acceso, el sistema es quién protege los recursos.

Sin embargo, actualmente este tipo de problema ha disminuido, gracias a que la seguridad está siendo llevada a cabo por RBAC. La administración de la seguridad, consiste en que los roles deben asignarse adecuadamente a los diferentes tipos de personas, según sus capacidades y puestos de cada una de ellas. Básicamente, la gran evolución que se está teniendo con RBAC es porque funciona como una mezcla de DAC y MAC, puesto que contiene de cierta forma la flexibilidad para el Control de Accesos que tiene DAC y la rigidez que se tiene con MAC.

Una de las características principales que muestran el manejo de los sistemas de Control de Accesos es que el administrador del sistema es quien maneja los datos, para así satisfacer las necesidades de la empresa u organización, tal como lo hace RBAC.

En RBAC, los permisos se encuentran asociados con los roles y los usuarios son miembros de los roles, de esta manera, se adquieren permisos a los roles, es por eso que la administración en el Control de Acceso es más sencilla. Los roles son creados en forma centralizada para las distintas funciones de trabajos en una organización, cada usuario tiene asignado uno o varios roles. Es por eso que cada usuario puede ser asignado de un rol a otro. Cada usuario tiene diferentes permisos, dependiendo del rol que esté llevando a cabo.

Para que un sistema sea confiable, debe tenerse en cuenta cuáles podrían ser los caminos para llegar a la confiabilidad, los niveles que existen en los usuarios y los permisos o privilegios que dichos usuarios puedan llegar a tener.

1.4. Estudio de módulos de administración

Módulos de administración en Cuba.

Módulo de Administración del sistema de la Aduana

El sistema informático instalado en las instituciones de Aduana de Cuba cuanta Módulo de Administración.

Dentro de las funcionalidades principales de este módulo se encuentran: gestionar el nivel de acceso de cada uno de los responsables de ejecutar las tareas en el sistema y actualizar la Base de Datos (BD).

Para garantizar el nivel de acceso del sistema se ha organizado el mismo de la siguiente forma, este cuenta con muchos usuarios y estos están asociados a muchos grupos de usuarios del sistema, el sistema está organizado por módulos, donde cada módulo cuenta con una serie de funcionalidades que están asociadas a cada uno de los grupos del sistema. En fin, para garantizar el nivel de acceso de un usuario, el sistema busca a qué grupos del mismo está asociado y a su vez a qué

funcionalidades de los módulos están vinculados, después de obtener esta lista de funcionalidades el sistema automáticamente las habilita para el uso del usuario.

Para un mejor funcionamiento de la Aduana en Cuba, este cuenta con un sistema central que se alimenta de muchos subsistemas que están desplegados en diferentes partes del país, por razones de conectividad todos los sistemas no están conectados al sistema central, debido a esta razón dichos subsistema tienen que generar un fichero, donde se recojan todas las acciones ejecutadas en el sistema hasta el momento, para poder actualizar el sistema central; esta es la principal razón por el cual el Módulo de Administración de la Aduana brinda el servicio de cargar un fichero para actualizar cualquier parte de la BD del Sistema Central.

Módulo de Administración del sistema Info@tletas

El Instituto Nacional de Deportes y Recreación (INDER), cuenta con una aplicación web Info@tletas la cual se encarga de la administración de los expedientes técnico-acumulativos de atletas y entrenadores nacionales e internacionales. Esta aplicación cuenta con diversos módulos entre ellos un módulo de administración, él se encarga de gestionar (inserción, modificación y/o eliminación) los datos de centros, deportes, especialidades deportivas, pruebas deportivas, países, provincias, municipios y competencias deportivas. Su seguridad está basada en roles de usuario, Seguridad, direcciones, infraestructuras productivas, Bitácoras o log.

Módulos de administración en la UCI.

Módulo de Administración del sistema CEDRUX.

El proyecto colaborativo entre el Ministerio de las Fuerzas Armadas Revolucionarias (FAR) y la UCI, implementaron un sistema llamado CEDRUX, dicho sistema cuenta con un Módulo de Administración encargado de gestionar la seguridad de cada una de las funcionalidades que éste pone a disposición de los usuarios.

El nivel de acceso está organizado de la siguiente forma: el sistema cuenta con usuarios que pueden estar asociados a uno o muchos roles y estos a su vez con un número determinado de funcionalidades del sistema. Para lograr la autenticación y configuración del sistema en dependencia del usuario que se autentique se realiza el siguiente proceso, el sistema identifica el usuario y los roles a los cuales está vinculado para obtener la lista de funcionalidades y activarlas al usuario que se ha autenticado.

Módulo Administración del Sistema para la Gestión de la Información de estudiantes y profesores de la Facultad 6.

Este módulo se encarga de llevar el control de los usuarios que entran a la aplicación, los roles que desempeñan, el registro de las actividades que realizan y los permisos, restringiendo de esta forma el acceso a los servicios que brinda el sitio y garantizando a las vez la seguridad de la información que se maneja en el mismo. Este módulo cuenta con las siguientes secciones:

- ✓ Usuarios.
- ✓ Permisos de usuarios.
- ✓ Roles de usuarios.
- ✓ Registro de actividades de Usuarios.

Conclusión del estudio realizado:

Después de terminar el estudio se demuestra que los módulos de administración en las aplicaciones web, juegan funciones que enmarcan lo siguiente:

- ✓ Gestión de Usuarios.
- ✓ Gestión de Roles.
- ✓ Asignación de Roles.
- ✓ Asignación de restricciones por roles.
- ✓ Gestión de clasificadores o valores que van hacer usados por el resto de los módulos donde está anclado el módulo de administración.

En el caso particular del Módulo de Administración del Sistemas CEDRUX, luego del estudio realizado se pudo ver que cumplía con la política actual de la UCI y del país de emigrar a software libre, además de tener gran parte de las funcionalidades definidas para el Módulo de Administración del sistema GeForza las cuales son:

- ✓ Gestionar roles.
- ✓ Gestionar usuarios.
- ✓ Gestionar campos del perfil de usuarios.
- ✓ Gestionar perfil de usuarios.
- ✓ Gestionar trazas.
- ✓ Cambiar contraseña.
- ✓ Autenticar usuarios.
- ✓ Asignar roles.

Por lo que se decidió trabajar sobre ese marco de trabajo dejando intacto los módulos de seguridad y trazas. Teniendo esto, exclusivamente era necesario implementar las funcionalidades que logren

homogeneización entre los clasificadores utilizados por los diferentes subsistemas del sistema GeForza.

Esto fue posible ya que la dirección del proyecto había tomado la decisión de trabajar la misma arquitectura que se definió para el sistema Cedrux, además de contar con toda la documentación y el porta con todo su soporte.

1.5. Software Libre, migración y tendencias actuales

Se denomina Software Libre, al software que brinda libertad a los usuarios sobre su producto adquirido, pues una vez obtenido puede ser usado, modificado, copiado, estudiado y redistribuido libremente, de modo más preciso se refiere a cuatro libertades de los usuarios del software:

La libertad para ejecutar el programa con cualquier propósito (llamada "libertad 0").

La libertad para estudiar y modificar el programa ("libertad 1").

La libertad de copiar el programa de manera que puedas ayudar a tu vecino ("libertad 2").

La libertad de mejorar el programa y hacer públicas tus mejoras de forma que se beneficie toda la comunidad ("libertad 3").

Es importante señalar que las libertades 1 y 3 obligan a que se tenga acceso al código fuente.

El software libre es el software que garantiza, sin costo adicional, la libertad de usar el programa con cualquier propósito; de estudiar cómo funciona y adaptarlo según las necesidades. La libertad de distribuir copias, de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad que lo usa se beneficie. La migración hacia el software libre evita la dependencia de los proveedores y se reciben mejores servicios y productos. Cuba y en particular la Universidad de las Ciencias Informáticas (UCI), han sustentado la posibilidad de migrar hacia software libre, por las ventajas que representa con respecto a los de tipo propietario. Desde el ámbito político, primeramente representa la no utilización de productos informáticos que demanden la autorización de sus propietarios (licencias) para su explotación. En el presente Cuba se encuentra a merced de la empresa norteamericana Microsoft, que tiene la capacidad legal de reclamar a Cuba que no siga utilizando un sistema operativo de su propiedad, basada en leyes de propiedad industrial por las cuales también Cuba se rige; esto provocaría una interrupción inmediata del programa de informatización de la sociedad, que como parte de la batalla de ideas está desarrollando el país. Así mismo, pudiera implementarse una campaña de descrédito a la isla, abogando el uso de la piratería informática por parte de las instituciones estatales cubanas.

Además, el software libre representa la alternativa para los países pobres y, es por concepción, propiedad social. Económicamente, su utilización no implica gastos adicionales por concepto de

cambio de plataforma de software, por cuanto es operable en el mismo soporte de hardware con que cuenta el país. La adquisición de cualquiera de sus distribuciones, puede hacerse de forma gratuita, descargándolas directamente de Internet o en algunos casos a muy bajos precios.

Por último, desde el punto de vista tecnológico permite su adaptación a los contextos de aplicación, al contar con su código fuente, lo cual garantiza un mayor por ciento de efectividad, además la corrección de sus errores de programación y obtención de las actualizaciones y nuevas versiones.

1.6. *Herramientas, metodología, lenguajes y tecnología*

1.6.1. Aplicación web

Una Aplicación Web es una interfaz diseñada para cubrir las necesidades de un negocio y gestionar su información (la información puede ser de dominio público o restringido a ciertas personas a través de un nombre de usuario y contraseña) con el objetivo de que cualquier persona pueda consultarla e interactuar con ella a través de la red. Se pueden adaptar a muchas situaciones, su objetivo es mejorar la forma de trabajo y la productividad de una empresa o grupo de personas de una manera sencilla.

Beneficios

- ✓ Permiten reunir las diferentes áreas de una entidad.
- ✓ Tener mayor control de datos y mejor seguridad en las diferentes secciones del sitio web.
- ✓ Permite tener información siempre actualizada.
- ✓ Otorga la flexibilidad de determinar niveles de acceso según la confidencialidad de los datos así como la posibilidad de realizar transacciones en línea.

Ventajas de una aplicación web

- ✓ Ahorra tiempo: Se pueden realizar diferentes tareas, sin necesidad de descargar ni instalar ningún programa.
- ✓ No hay problemas de compatibilidad: Basta tener una computadora mínimamente actualizado para poder utilizarlas.
- ✓ No ocupan espacio en el disco duro.
- ✓ Actualizaciones inmediatas: Al conectarse se está utilizando siempre la última versión que se haya lanzado.
- ✓ Consumo de recursos bajo: Dado que la aplicación se encuentra instalada en otro ordenador, muchas de las tareas que realiza el software no consumen recursos de la computadora que se está usando.
- ✓ Multiplataforma: No importa el sistema operativo para que esta funcione correctamente.

- ✓ Portables: Son independiente de la computadora donde se utilicen, sólo es necesario disponer de acceso a Internet.
- ✓ La disponibilidad suele ser alta: Porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.
- ✓ Los virus no dañan los datos: Estos están guardados en el servidor de la aplicación.

1.6.2. Patrón de Arquitectura

Modelo Cliente-Servidor.

Esta arquitectura consiste básicamente en que el cliente realiza peticiones a otro programa -el servidor- que le da respuesta.

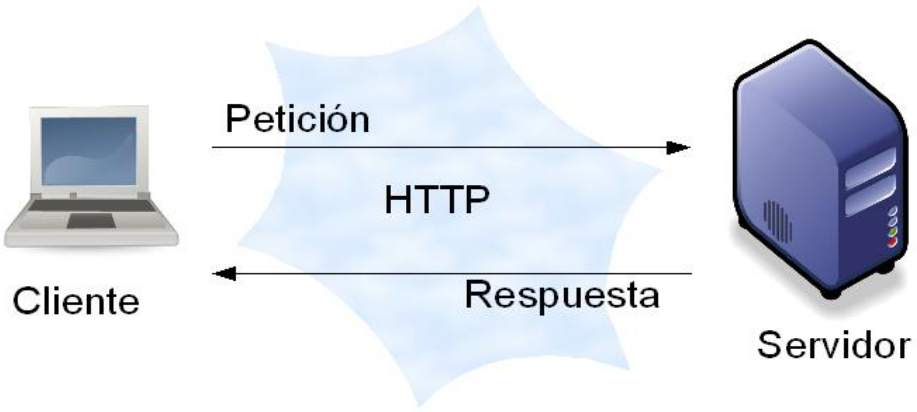

Figura 1. Modelo Cliente-Servidor

Cliente: Es el que comienza un requerimiento de servicio

Que es un servidor: Es cualquier recurso de automatización dedicado a responder a las exigencias del cliente.

Ventajas del modelo Cliente-Servidor:

- ✓ Se reducen los costos de producción de software y se disminuyen los tiempos requeridos, ya que para la fabricación de una nueva aplicación pueden usarse los servidores que estén disponibles, reduciéndose el desarrollo a la elaboración de los procesos del cliente, según los requerimientos deseados.
- ✓ Facilita el suministro de información a los usuarios, proporcionando una mayor consistencia a la información de la organización, al contar con un control centralizado de los elementos compartidos.

- ✓ Permite llevar más fácilmente la información a donde se necesita y contribuye a aumentar su precisión, pues se puede obtener de la fuente (el servidor) y no de una copia en papel o en medio magnético.
- ✓ La habilidad de integrar sistemas heterogéneos es inherente al modelo Cliente-Servidor, pues los clientes y los servidores pueden existir en múltiples plataformas y tener acceso a datos de cualquier sitio de la red.
- ✓ Favorece la adaptación a cambios en la tecnología, pues facilita la migración de las aplicaciones a otras plataformas y, al aislar claramente las diferentes funciones de una aplicación, hace más fácil incorporar nuevas tecnologías en esta.

Modelo Vista Controlador (MVC).

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Aparta esencialmente la lógica del negocio de la lógica de la presentación, factor que posibilita la simplificación del trabajo y el mantenimiento de los sistemas.

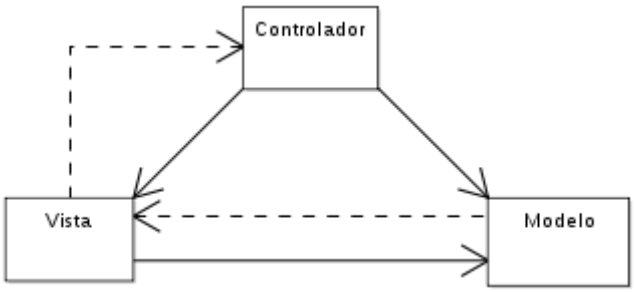

Figura 2. Patrón de Arquitectura MVC

El correcto uso de este patrón dispone, cómo se observa en la figura anterior, de tres capas básicas: el Modelo, la Vista y el Controlador.

Modelo: No es más que el Sistema de Gestor de Base de Datos (SGBD).

Vista: Es la página definición de la sigla HTML (HyperText Markup Language) y el código que provee los datos dinámicos a la página, es decir la capa donde se le muestra la información al usuario.

Controlador: Es la lógica de negocio

Ventajas:

- ✓ Hay un claro alejamiento entre los dispositivos de un programa; lo cual permite implementarlos por separado.
- ✓ La conexión entre el Modelo y sus Vistas es dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.
- ✓ Centralización de los aspectos de seguridad y transaccionalidad, que serían responsabilidad del modelo.
- ✓ No replicación de lógica de negocio en los clientes: esto permite que las modificaciones y mejoras sean automáticamente aprovechadas por el conjunto de los usuarios, reduciendo los costes de mantenimiento.
- ✓ Mayor sencillez de los clientes.

1.6.3. Metodología de desarrollo de software

Una metodología de desarrollo de software no es más que el conjunto de pasos y procedimientos que deben seguirse para desarrollar software.

Todo desarrollo de software es riesgoso y difícil de controlar, pero si no se lleva una metodología de por medio, lo que se obtienen son clientes insatisfechos con el resultado y desarrolladores aún más insatisfechos. Lo más importante antes de elegir la metodología que se usará para la implementación de un software, es determinar el alcance que tendrá y luego de ahí ver cuál es la que más se acomoda a la aplicación. Las metodologías fueron diseñadas bajo un conjunto de filosofías, fases, procedimientos, reglas, técnicas, herramientas, documentación y aspectos de formación para lograr la calidad, que es el principal objetivo estratégico en las organizaciones. Para el desarrollo del sistema informático propuesto, se escogió una de las metodologías más utilizadas en la universidad.

Proceso Unificado de Desarrollo (RUP)

Es un marco de desarrollo de software iterativo e incremental, que está compuesto por cuatro fases denominadas Inicio, Elaboración, Construcción y Transición. Está basado en componentes e interfaces bien definidas y utiliza el Lenguaje Unificado de Modelado (UML) para visualizar, especificar, construir y documentar un sistema de software.

Características de RUP:

- ✓ Dirigido por los casos de uso: los casos de usos son una forma abstracta de representar lo que los usuarios finales necesitan, por lo cual ellos son los que deben guiar el proceso de desarrollo. Esto se garantiza con la obtención de los diferentes modelos que son el resultado de los flujos de trabajo que propone RUP.

- ✓ Centrado en la arquitectura: la arquitectura de un proyecto muestra una visión común del sistema completo, visión en la que deben estar de acuerdo tanto el equipo de desarrollo como los clientes. Es por ello que la arquitectura describe los elementos del modelo que son más importantes para la construcción del sistema, así como los cimientos para comprenderlo y desarrollarlo de forma económica.
- ✓ Iterativo e incremental: RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y las cuales se definen según el nivel de madurez que alcanzan los productos que se van obteniendo con cada actividad ejecutada. La terminación de cada fase ocurre en el hito correspondiente a cada una, donde se evalúa que se hayan cumplido los objetivos de la fase en cuestión.
- ✓ Enfocado en los riesgos: El Proceso Unificado requiere que el equipo del proyecto se centre en identificar los riesgos críticos en una etapa temprana del ciclo de vida. Los resultados de cada iteración, en especial los de la fase de Elaboración, deben ser seleccionados en un orden que asegure que los riesgos principales son considerados primero.

1.6.4. Lenguaje Unificado de Modelado (UML)

UML es un grupo de especificaciones de notación orientadas a objeto, las cuales están compuestas por distintos diagramas, que representan las diferentes etapas del desarrollo de un proyecto de software. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad y está respaldado por el Grupo de Gestión de Objetos (OMG de su significado en inglés Object Management Group). Ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. Este lenguaje indica cómo crear y leer los modelos, pero no dice cómo desarrollarlos, esto último es el objetivo de las metodologías de desarrollo. Los objetivos de UML son muchos, pero se pueden sintetizar sus funciones:

- ✓ Visualizar: permite expresar gráficamente un sistema de forma que otra persona lo pueda entender.
- ✓ Especificar: permite especificar las características de un sistema antes de su construcción.
- ✓ Construir: a partir de los modelos especificados se pueden construir los sistemas diseñados.
- ✓ Documentar: los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

1.6.5. Herramienta case para el modelado

Hoy en día existen múltiples herramientas de modelado, entre ellas se puede encontrar: ArgoUML, BoUML, Umbrello, Visual Paradigm y Rational Rose Enterprise Edition. Las dos últimas son propietarias pero en el caso de Visual Paradigm la universidad tiene la licencia para su uso, además que permite la generación de código PHP que es el lenguaje propuesto por sus amplias características para el desarrollo del sistema.

Visual Paradigm.

Visual Paradigm es una herramienta de modelado que utiliza el lenguaje de modelado estándar UML, permite la generación de códigos e ingeniería inversa. Con una clase de diseño bien especificada, Visual Paradigm puede generar código hasta en quince lenguajes de programación entre ellos PHP, Java, C++, Esquemas XML, XML, .NET exe/dll, CORBA IDL.

Visual Paradigm actualmente cumple con las políticas de migración a Software Libre en Cuba, ya que es una herramienta multiplataforma que se puede utilizar tanto en Linux como en Windows. Su interfaz es muy intuitiva, de fácil aprendizaje para los desarrolladores. Permite la generación automática de diagramas a partir de descripciones de casos de usos, por ejemplo, diagramas de secuencia, permitiendo la agilidad en el trabajo del analista.

Visual Paradigm ofrece:

- ✓ Entorno de creación de diagramas para UML 2.0.
- ✓ Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Capacidades de ingeniería directa (versión profesional) e inversa.
- ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- ✓ Disponibilidad de múltiples versiones, para cada necesidad.
- ✓ Disponibilidad de integrarse en los principales IDE.
- ✓ Disponibilidad en múltiples plataformas.

1.6.6. Lenguaje de programación para el desarrollo web.

PHP

Constituye un lenguaje *script* de alto nivel interpretado del lado del servidor. Es un lenguaje de programación (originario del nombre *PHP Tools* o *Personal Home Page Tools*) que sirve principalmente para proporcionar características dinámicas a una página Web. Al ser ejecutado del lado del servidor, PHP permite acceder a los recursos internos del mismo o a otros externos, como por

ejemplo a una base de datos, siendo el resultado normalmente una página HTML con los datos y acciones enviadas al cliente.

Ventajas

- ✓ Es un lenguaje multiplataforma.
- ✓ Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una Base de Datos.
- ✓ El código fuente escrito en PHP es invisible al navegador y al cliente, es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- ✓ Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- ✓ Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- ✓ Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- ✓ Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- ✓ Permite aplicar técnicas de programación orientada a objetos.
- ✓ Biblioteca nativa de funciones sumamente amplia e incluida.
- ✓ No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- ✓ Tiene manejo de excepciones (desde PHP5).

PHP 5

El 13 de julio de 2004, fue lanzado PHP 5, utilizando el motor Zend Engine II (o Zend Engine 2). La versión más reciente de PHP es la 5.2.6 (1 de mayo de 2008), que incluye todas las ventajas que provee el nuevo Zend Engine 2 como:

Mejor soporte para la Programación Orientada a Objetos, que en versiones anteriores era extremadamente rudimentario, con PHP Data Objects.

- ✓ Mejoras de rendimiento.
- ✓ Mejor soporte para MySQL con extensión completamente reescrita.
- ✓ Mejor soporte a XML (XPath, DOM, etc.).
- ✓ Soporte nativo para SQLite.
- ✓ Soporte integrado para SOAP.

- ✓ Integradores de datos.

1.6.7. Marco de trabajo

Marco de trabajo (Framework): no es más que una estructura de soporte definida mediante la cual otro proyecto de software puede ser desarrollado y organizado. Pueden incluir soporte de programas, bibliotecas y lenguajes interpretados entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio. Son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional. Es el esqueleto sobre el cual varios objetos son integrados para una solución dada. No es más que una base de programación que atiende a sus descendientes (manejado de una forma estructural y/o en cascada) posibilitando cualquier respuesta ante las necesidades de sus miembros, o secciones de una aplicación web.

EXT.JS

Ext es un nuevo marco de trabajo de javascript del lado del cliente para el desarrollo de aplicaciones web, basado originalmente en YUI pero que actualmente es independiente del marco de trabajo que se utilice (incluso puede usarse sin el marco de trabajo). Ofrece una gran cantidad de widgets para crear interfaces de usuario complejas. PHP-Ext es una librería open source que permite potenciar la capa UI de Javascript en aplicaciones. Para ello ofrece una serie de librerías (compatibles con PHP 4 y 5) para integrar Ext JS en un sistema. Funciona como un mapeado en clases de la librería JS. Entre las posibilidades que ofrece se encuentra con la creación de formularios, combos, grids o menús. Además, ayuda a la comunicación entre el cliente y el servidor mediante JSON y XML.

Tiene un sistema dual de licencia: Comercial y Open Source. Este marco de trabajo puede correr en cualquier plataforma que pueda procesar POST y devolver datos estructurados (PHP, Java, .NET y algunas otras). Basa toda su funcionalidad en Java Script a través de librerías ya muy conocidas. En tiempo de ejecución carga y crea todos los objetos HTML a través del uso intenso de DOM (Modelo de Objetos del Documento).

Ventajas:

- ✓ Código reutilizable.
- ✓ Independiente o adaptable a marco de trabajo diferentes.
- ✓ Orientada a la programación de interfaces tipo escritorio en la web.

- ✓ El API es homogeneizado independientemente del adaptador usado. Los controles siempre se verán igual.
- ✓ Soporte comercial.
- ✓ Una extensa comunidad de usuarios.
- ✓ La orientación a objetos intensa te hará modular todos tus scripts.
- ✓ El diseño está completamente separado de la funcionalidad.
- ✓ Funciones comunes como validación, combobox editables, ventanas arrastrables (con minimizar y maximizar) y grillas editables, son muy fáciles de implementar.
- ✓ Utilización de AJAX y JSON como mecanismos de comunicación con el servidor.
- ✓ Implementación basada en patrones de diseño.
- ✓ Amplia librería de componentes gráficos fácilmente extensibles.
- ✓ Buena y amplia documentación, así como también su comunidad.

Doctrine.

Doctrine es un potente y completo sistema ORM (Mapeador Relacional de Objetos) para PHP 5.2+ con un DBAL (Capa de Abstracción de Base Datos) incorporado.

Entre muchas otras cosas tienes la posibilidad de exportar una base de datos existente a sus clases correspondientes y también a la inversa, es decir, convertir clases (convenientemente creadas siguiendo las pautas del ORM) a tablas de una base de datos. Por otro lado, como la librería es bastante grande, esta tiene un método para ser “compilada” al pasar a producción. Su principal ventaja radica en poder acceder a la base de datos utilizando la programación orientada a objetos (POO) debido a que doctrine utiliza el patrón Active Record para manejar la base de datos, tiene su propio lenguaje de consultas y trabaja de manera rápida y eficiente. Es fácilmente integrado a los principales marcos de trabajo de desarrollo utilizados actualmente.

Zend Framework.

Los marcos de trabajo por lo general presentan una estructura organizada que obliga a los programadores a seguir estándares y a trabajar de manera organizada. El uso de estas aplicaciones ha demostrado que la organización de la programación influye notablemente en la calidad de las aplicaciones.

Zend Framework es uno de los más utilizados para PHP y utiliza el estilo MVC como base de su funcionamiento. Es fácilmente integrable a las aplicaciones debido a su composición ya que contiene diferentes clases de gran utilidad como, por ejemplo en la búsqueda dinámica de ficheros a incluir. Orientado a componentes. Acoplable a variadas tecnologías de desarrollo. Cuenta con un importante

mecanismo de manejo de controladores y vistas por lo que se propone tenerlo en cuenta para el diseño de estos dos componentes de la arquitectura.

1.6.8. Servidor web

Básicamente, un servidor web sirve contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante HTTP.

Apache 2.0.

Posee facilidad de configuración, robustez, estabilidad y transparencia. Apache está desarrollado por la Fundación de Software Apache (Apache Software Foundation), es una tecnología gratuita de código fuente abierto con licencia descendiente de las licencias BSD, por lo que se puede hacer uso de su código fuente. Lo que hace a este servidor web universal, es que corre en una multitud de sistemas operativos. Es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor Web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este. Brinda soporte para varios lenguajes: PHP, JAVA, Perl y librerías ASP. Apache permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto. Un gran número de aplicaciones web están montadas en servidores Apache.

1.6.9. Entorno Integrado de Desarrollo (IDE).

Un Entorno de Integrado Desarrollo (IDE de su significado en inglés Integrated Development Environment), es un conjunto de herramientas que ha sido creado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Un IDE provee un marco de trabajo amigable para la mayoría de los lenguajes de programación y en algunos casos puede funcionar como un sistema en tiempo de ejecución en donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto. Entre los entornos integrados de desarrollo se encuentran los siguientes:

- ✓ Eclipse.
- ✓ VisualStudio.NET.
- ✓ NetBeans.
- ✓ SharpDevelop.
- ✓ Zend Studio.
- ✓ NetBeans.

NetBeans IDE es una herramienta para programadores de código abierto escrito en Java usando la plataforma NetBeans. Soporta el desarrollo de todos los tipos de aplicación Java, tales como: JEE *(Java Platform, Enterprise Edition),* Web, EJB *(Enterprise JavaBeans)* y aplicaciones móviles. Las funciones de este IDE son provistas por módulos donde cada uno provee una función bien definida: soporte de Java, edición o soporte para el sistema de control de versiones. NetBeans contiene todos los módulos necesarios para el desarrollo de aplicaciones Java en una sola descarga, facilitándole al usuario comenzar a trabajar inmediatamente. La plataforma NetBeans es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes. Ofrece servicios comunes permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma se encuentran:

- ✓ Administración de las interfaces de usuario.
- ✓ Administración de las configuraciones del usuario.
- ✓ Administración de almacenamiento.
- ✓ Administración de ventanas.
- ✓ Marco de trabajo basado en asistentes.

1.6.10. Sistema Gestor de Base de Datos (SGBD).

Los sistemas que gestionan datos, así como las transacciones entre estos, constituyen una potente herramienta, la cual está estrechamente vinculada a sistemas de software robustos que necesiten de la persistencia, localización y explotación de un volumen amplio de datos. El propósito general de los gestores de base de datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante, para un buen manejo de datos. Los principales objetivos de los gestores de base de datos son:

- ✓ Abstracción de la información: Los gestores ahorran a los usuarios detalles acerca del almacenamiento físico de los datos.
- ✓ Independencia: Consiste en la capacidad de modificar el esquema físico o lógico de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.
- ✓ Respaldo y recuperación: Los gestores deben proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada en ellos, y de restaurar a partir de estas copias datos perdidos.
- ✓ Seguridad: La información almacenada en una base de datos puede llegar a tener un gran valor, los gestores deben garantizar que esta información se encuentre segura frente a usuarios malintencionados, que intenten acceder a la misma con el objetivo de manipularla o destruirla.

Los gestores de base de datos cuentan con un complejo sistema de permisos y grupos de usuarios, que permiten otorgar diversas categorías de permisos.

En la actualidad existe gran cantidad de gestores, cada uno con características distintas, así como ventajas y desventajas a la hora de ser usado, los dos principales gestores libres son: PostgreSQL y MySQL, ambos muy populares. Para la realización de este trabajo se decidió usar PostgreSQL.

Postgre SQL.

Es un sistema de gestión de bases de datos objeto-relacional basado en el proyecto *Ingres*, de la universidad de *Berkeley*, California. Pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, PostgreSQL incluye características de la orientación a objetos como pueden ser: la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional, además de otras específicas del gestor, como lo son: un mejor soporte para *sub-selects*, *triggers*, vistas y procedimientos almacenados. Entre sus características más específicas se encuentran:

- ✓ Implementación del estándar SQL92/SQL99.
- ✓ Soporte para distintos tipos de datos, como son: datos de tipo fecha, datos sobre redes (MAC, IP), cadenas de bits, etc., así como la creación de tipos de datos propios.
- ✓ Al igual que con los tipos de datos, PostgreSQL permite la declaración de funciones propias, así como la definición de dispensadores.
- ✓ Soporte en la mayoría de los sistemas operativos más utilizados incluyendo, Linux, varias versiones de UNIX, BeOS y Windows.

1.7. Fundamentación de herramientas, metodologías, lenguajes y tecnología a utilizar

La dirección del proyecto junto con el equipo de arquitectura, realizó una investigación de las tendencias actuales en el desarrollo de software donde se decide utilizar la misma arquitectura del Sistema CEDRUX. Ya que esta arquitectura se utiliza como referencia para la programación web en PHP en la UCI, trabajándose con estas herramientas por las siguientes razones:

Dado el entorno en que se tiene que utilizar la aplicación se considera que sería oportuno realizar una aplicación web, utilizando a apache 2.0 o superior como servidor web, ya que la política que se está llevando a cabo en Cuba en estos momentos es migrar a software libre, además de que apache cuenta con la robustez necesitada para satisfacer las solicitudes de los clientes. Se propone por causas similares utilizar a PHP 5.0 o superior como lenguaje de desarrollo.

Se considera a RUP dentro de las metodologías estudiadas la más eficiente para el modelado del sistema. Dado que el alcance del proyecto abarca procesos complejos, el tiempo acordado con el cliente para el desarrollo del software es suficiente para generar los artefactos y actividades definidos por RUP, además, se necesita una buena documentación y organización de los procesos.

Luego de analizar diferentes herramientas para el modelado del sistema se llega a la conclusión que Visual Paradigm es la más conveniente. Aunque es una herramienta privada, la universidad posee su licencia lo que facilita los trámites en la producción del software.

Se usarán varios marcos de trabajo, entre ellos: Doctrine para la capa de abstracción de base de datos, pues brinda un gran rendimiento en ejecución y admite escribir de forma segura consultas muy complejas, otros de los marcos de trabajo que se utilizarán son: Ext JS, para programar la capa de presentación, ya que permite crear interfaces de usuarios muy amigables y eficaces, y Zend Framework, para programar la capa de negocio, pues posibilita diseñar su estructura de componentes con pocas dependencias, posibilitando que los desarrolladores utilicen componentes individualmente los cuales forman una potente y extensible aplicación Web al ser mezclados.

Como entorno integrado de desarrollo se usará NetBeans porque es un producto de código abierto que le permite a los equipos de desarrollo utilizar las mejores prácticas y estándares de la industria del software para la productividad general del grupo, permite que las aplicaciones se desarrollen a partir de un conjunto de módulos o componentes de software, además es la herramienta ideal para el desarrollo con PHP.

Como sistema gestor de Base de Datos se utiliza PostgreSQL, pues es uno de los gestores de bases de datos más utilizados actualmente, es capaz de manejar una enorme cantidad de datos, permitiendo gran conjunto de accesos simultáneos de los usuarios, brindando seguridad y estabilidad de los mismos, facilita el trabajo con procedimientos almacenados y consultas.

1.8. Conclusiones

En este capítulo se realizó un estudio relacionado con los Módulos de Administración, dando una panorámica general de cómo estos se implementaban hoy en día, se pusieron ejemplos de los mismos, lo que tributó conocimientos e ideas para la realización del sistema. Se realizó un análisis de las tecnologías informáticas a emplear a lo largo del desarrollo de la solución del problema, haciendo énfasis en las tendencias de migración a Software Libre, se eligieron los lenguajes PHP 5.2.8 y UML como lenguaje de modelado: marco de trabajo: Ext JS, Doctrine 1.1 y Zend Framework 1.7.6, como servidor web: Apache 2.2.9, como herramienta de modelado: Visual Paradigm, metodología a utilizar: RUP y Sistema Gestor de Base de Datos PostgreSQL 8.3.

DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA

2.1. Introducción

En este capítulo se presenta el negocio a través de un modelado del dominio. Además, se representa una propuesta del software a desarrollar, especificando los requerimientos funcionales y no funcionales y a partir de ellos la definición de los casos de uso del sistema y sus descripciones.

2.2. Modelo del dominio

Un modelo del dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las “cosas" que existen o los eventos que suceden en el entorno donde estará el sistema.

Después de haber realizado un estudio de los procesos que intervienen en el módulo de administración y de haber realizado entrevistas con los clientes expertos en el proceso, se diseñó un modelo del dominio, debido a que estos ofrecieron un negocio definido y para que el equipo de desarrollo entendiera lo que se quería se decidió hacer este modelado. Un modelo del dominio representa cosas del mundo real y para poder identificar los conceptos se hace necesario investigar el dominio del problema. Se especifican las relaciones que existen entre los principales conceptos teniendo en cuenta que el sistema retroalimentara a los otros sistemas.

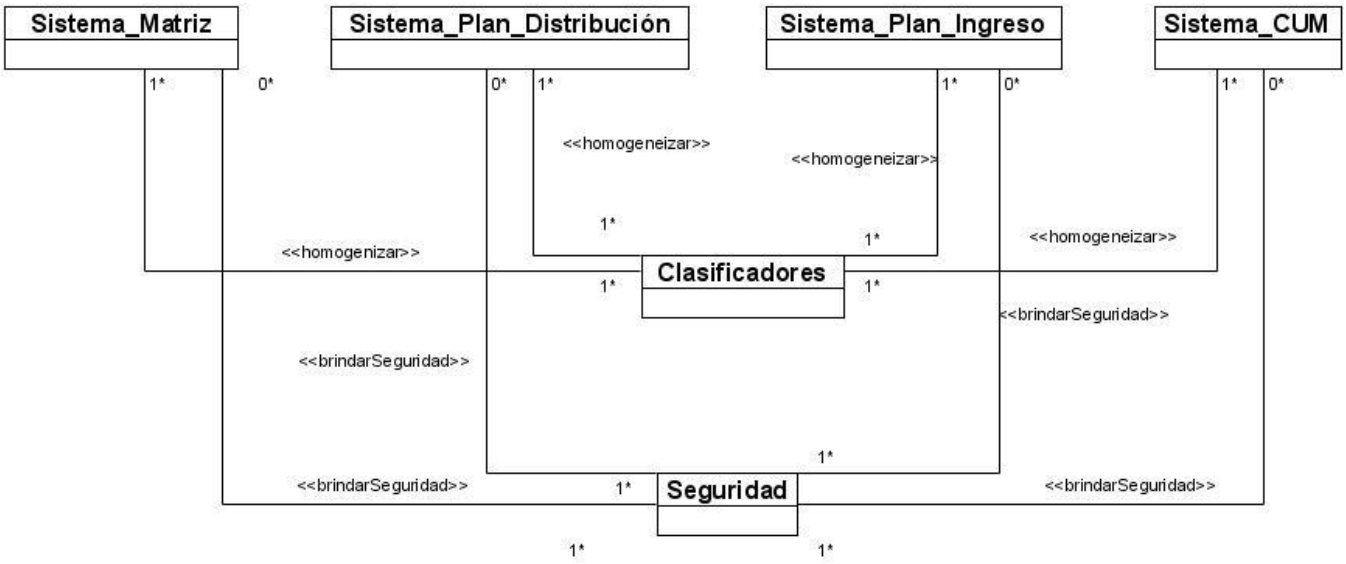

Figura 3. Modelo de Dominio.

Definición de las clases del Modelo de Dominio.

Clasificador: Proceso que permite una ordenación de elementos, según un determinado criterio.

Sistema Matriz

Es el sistema principal el cual es el encargado de captar toda la información y brindársela a los demás sistemas para que se lleve a cabo la planificación de los recursos humanos calificados del país.

Plan de Ingreso a la Educación Superior y Técnico Medio

Es un subsistema de GeForza, el cual es el encargado capturar los datos necesarios para que se elabore de forma correcta el plan de ingreso a la educción media y superior.

Plan de Distribución de Graduados

Es un subsistema de GeForza, el cual es el encargado capturar los datos necesarios para que se elabore de forma correcta la distribución de graduados del curso regular diurno.

Plan de Ingreso a las Centros Universitarios Municipales (CUM)

Es un subsistema de GeForza, el cual es el encargado de gestionar los datos necesarios para elabore el plan de ingreso a los Centros Universitarios Municipales (CUM).

2.3. Especificación de los Requerimientos del software

Un requerimiento es una necesidad documentada sobre el contenido, forma o funcionalidad de un producto o servicio. Condición o capacidad que un usuario necesita para poder resolver un problema o lograr un objetivo. Condición o capacidad que debe exhibir o poseer un sistema para satisfacer un contrato, estándar, especificación, u otra documentación formalmente impuesta. Una condición o

capacidad que debe ser conformada por el sistema (RUP). Algo que el sistema debe hacer o una cualidad que el sistema debe poseer.

2.3.1. Requerimientos Funcionales

Los requerimientos funcionales (RF) especifican acciones que el sistema debe ser capaz de realizar, sin tomar en consideración ningún tipo de restricción física. Es decir, especifican el comportamiento de entrada y salida del sistema y surgen de la razón fundamental de la existencia del producto.

A continuación se podrán apreciar los requisitos abreviados del sistemas en el anexo # 2 se podrán ver bien especificados.

RF-1: Realizar autenticación de usuario

1.1. Configurar el sistema en dependencia de las funcionalidades que pueda realizar

1.2. Verificar si el usuario es válido

1.3. Verificar si la contraseña es válida

RF-2: Tramitar Roles

2.1. Agregar roles

2.2. Modificar roles

2.3. Eliminar roles

2.4. Regular acciones

RF-3: Tramitar Usuario

3.1. Agregar usuario

3.2. Modificar usuario

3.3. Eliminar usuario

RF-4: Tramitar trazas

4.1. Mostrar trazas de usuarios

4.2. Eliminar trazas de usuario

RF-5: Modificar clave de usuario.

5.1. Solicitar datos de usuario

5.2. Contraseña anterior

5.3. Nueva contraseña

5.4. Confirmar contraseña

RF-6: Tramitar Entidades

6.1. Agregar Entidades

6.2. Modificar Entidades

6.3. Eliminar Entidades

6.4. Buscar Entidad

RF-7: Tramitar organismos

7.1. Agregar organismo

7.2. Modificar organismo

7.3. Eliminar organismo

7.4. Buscar organismo

RF-8: Tramitar carrera o especialidad

8.1. Agregar carrera o especialidad

8.2. Modificar carrera o especialidad

8.3. Eliminar carrera o especialidad

8.4. Buscar carrera o especialidad

RF-9: Tramitar provincia

9.1. Agregar provincia

9.2. Modificar provincia

9.3. Eliminar provincia

RF-10: Tramitar municipio

10.1. Agregar municipio

10.2. Modificar municipio

10.3. Eliminar municipio

10.4. Buscar municipio

RF-11: Tramitar correlacionador de carreras

11.1. Agregar correlacionador de carreras

11.2. Modificar correlacionador de carreras

11.3. Eliminar correlacionador de carreras

11.4. Buscar correlacionador de carreras

RF- 12: Tramitar Ramas de las Ciencias

12.1. Agregar ramas de las ciencias

12.2. Modificar ramas de las ciencias

12.3. Eliminar ramas de las ciencias

RF-13: Tramitar familias de especialidades

13.1. Agregar familias de especialidades

13.2. Modificar familias de especialidades

13.3. Eliminar familias de especialidades

RF-14: Tramitar Tipos

14.1. Agregar Tipos

14.2. Modificar un Tipo

14.3. Eliminar un Tipo

RF-15: Tramitar Empresa

15.1. Agregar empresa

15.2. Modificar empresa

15.3. Eliminar empresa

15.4. Buscar empresa

RF-16: Tramitar centros priorizados

16.1. Agregar centro priorizados

16.2. Modificar centro priorizados

16.3. Eliminar centro priorizados

16.4. Código del centro priorizado

RF-17: Tramitar fuentes de ingreso

17.1. Agregar fuentes de ingreso

17.2. Modificar fuentes de ingreso

17.3. Eliminar fuentes de ingreso

17.4. Buscar fuentes de ingreso

RF-18: Tramitar Edades

18.1. Agregar edades

18.2. Modificar edades

18.3. Eliminar edades

RF-19: Tramitar programas de la revolución

19.1. Adicionar programas de la revolución

19.2. Modificar programas de la revolución

19.3. Eliminar programas de la revolución

19.4. Buscar programas de la revolución

RF-20: Tramitar Eficiencia Nacional

20.1. Adicionar eficiencia nacional

20.2. Modificar eficiencia nacional

20.3. Eliminar eficiencia nacional

20.4. Buscar eficiencia nacional

RF-21: Tramitar eficiencia provincial

21.1. Adicionar eficiencia provincial

21.2. Modificar eficiencia provincial

21.3. Eliminar eficiencia provincial

21.4. Buscar eficiencia provincial

RF-22: Exportar PDF

22.1. Muestra un PDF con todas sus opciones

RF-23: Exportar XML

23.1. Exportar todos los modelos y reportes que desee el usuario. Para esto se dará la posibilidad de escoger el (los) mismo (s).

RF 24: Importar XML

24.1. Restaurar en la base de datos todos los datos obtenidos a partir de un archivo XML anteriormente seleccionado

2.3.2. Requisitos no Funcionales

Los requerimientos no funcionales (RNF) son las propiedades o cualidades que el producto debe tener para que este sea atractivo, usable, rápido y confiable.

RNF 1. Usabilidad

- ✓ A los administradores finales de la aplicación se les debe dar un adiestramiento básico en el uso de la aplicación. Estas personas deben tener un nivel de acceso amplio en la aplicación para poder darle respuesta a cada incidente ocurrido.

RNF 2. Fiabilidad

- ✓ Permitir la transferencia de información desde las distintas versiones (MEP u organismo) hacia las otras.
- ✓ Permitir la transferencia de información de manera fragmentada y que después se pueda consolidar en la base de datos del MEP.

RNF 3. Seguridad

Políticas de seguridad por usuarios y roles: el sistema debe contar con un grupo de políticas de accesibilidad a las diferentes funcionalidades del mismo en dependencia del nivel de autorización que presente un usuario determinado.

- ✓ El sistema garantizará la autenticación como primera acción. Esta consistirá en suministrar un nombre de usuario único y una contraseña que debe ser de conocimiento exclusivo de la persona que se autentica.
- ✓ El sistema registrará todas las operaciones realizadas por los usuarios autenticados.
    - o Código y nombre de la persona.
    - o Fecha y hora de inicio y de terminación de la operación.

- Operación realizada y oficina desde donde se realizó la operación.
- Estación de trabajo desde la que operó.

RNF 4. Eficiencia

- ✓ El sistema debe funcionar con un máximo rendimiento pero ajustado a bajas prestaciones de las computadoras debido que no todos los organismos poseen tecnología de punta.
- ✓ La aplicación debe garantizar un acceso fácil y rápido, contando con un menú que satisfaga las necesidades de los usuarios.

RNF 5. Soporte

- ✓ El sistema deberá presentar un manual de usuario, permitiendo con ello un correcto uso de sus funcionalidades y brindarle al usuario una mayor experiencia del trabajo con el mismo.
- ✓ La documentación del sistema debe estar actualizada en todos los aspectos, fases de trabajo y ciclos de desarrollo del mismo, permitiendo con ello un respaldo tanto ingenieril como legal del desarrollo de dicho sistema.
- ✓ El sistema contará con un grupo de soporte y asesoría al cliente del producto destinado a brindar asesoría y soporte técnico al mismo.

RNF 6. Restricciones de diseño

- ✓ Lenguaje de programación: PHP 5.
- ✓ Como gestor de base de datos se utilizará PostgreSQL 8.3 o superior.
- ✓ Los marcos de trabajo que se utilizarán son: Zend Framework para la capa de negocio, Ext JS para la capa de presentación y Doctrine para la capa de acceso a datos
- ✓ Como Entorno Integrado de Desarrollo (IDE) se empleará NetBeans 6.8.
- ✓ El servidor de aplicaciones será Apache-2.0 o superior
- ✓ El modelado UML se hará con Visual Paradigm 6.1.
- ✓ Se utilizará como metodología de desarrollo de software RUP.
- ✓ Las interfaces destinadas al usuario, se programarán en JavaScript.

RNF 7. Hardware

- ✓ Se necesitará una impresora, para la impresión de los reportes del sistema.
- ✓ Las PCs clientes deben tener las siguientes características: memoria RAM de 256 MB o superior, 20 GB o más de disco duro, 1.2 GHz o más de procesador.
- ✓ Todas las PCs clientes deben estar conectadas a la red para poder acceder a la aplicación.
- ✓ Se requiere de un servidor para Bases de Datos con las siguientes características:
  - Servidor Xeon a 3.0 GHz.
  - 1 GB de memoria RAM.

- Dos discos duros de 36 y 250 GB, este último con dos particiones.

RNF 8. Software

- ✓ Para el cliente: se usará un sistema operativo Windows 95, cualquier versión superior o GNU/Linux.
- ✓ Las computadoras cliente del sistema y ubicadas en el dominio de la organización, deben tener el navegador Mozilla Firefox.
- ✓ Para el servidor: sistema operativo Windows Server 2000 o superior o Linux; Ubuntu Server 7.10 o superior.
- ✓ Servidor Web: Apache 2.0 o superior.
- ✓ Gestor de base de datos: PostgreSQL.

RNF 9. Apariencia o interfaz externa

- ✓ Interfaz web: la interfaz deberá ser sencilla, amigable con colores suaves a la vista y sin cúmulo de imágenes u objetos que distraigan al cliente del objetivo de su empleo, brindando facilidades que permitan interactuar con el sistema de forma fácil y rápida. Permitiendo que personas con pocas habilidades informáticas manejen el sistema.

RNF 10. Requisitos de licencia

- ✓ El proyecto utiliza la política de software libre donde todas las herramientas que utilizan son libres. Para la herramienta Visual Paradigm se utiliza la licencia que la universidad adquirió.

RNF 11. Requisitos legales, de derecho de autor y otros.

- ✓ La Universidad de las Ciencias de Informáticas tiene el derecho de autor sobre el Sistema Unificado de Gestión de Fuerza de Trabajo Calificada.

2.4. Modelo del sistema

A continuación se podrá apreciar el modelado del sistema en un diagrama de casos de uso donde se representa gráficamente los procesos que se llevarán a cabo en el sistema y su interacción con los actores.

2.4.1. Actor del sistema

El actor es una entidad externa del sistema que de alguna manera participa en la historia del caso de uso. Por lo regular estimula el sistema con eventos de entrada o recibe algo de él.

Actor	Descripción
Administrador	Es el rol que tiene la responsabilidad de asegurar el correcto funcionamiento y mantenimiento del sistema.

Tabla #1: Descripción del actor del sistema.

2.4.2. Definición de los casos de usos del sistema

Un caso de uso es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado significativo.

Listado de casos de uso del sistema:

CU-1 Gestionar Organismo.

CU-2 Gestionar División Política Administrativa.

CU-3 Gestionar Entidad.

CU-4 Gestionar Nomenclador.

CU-5 Gestionar Programas de la Revolución.

CU-6 Gestionar Correlacionador de Carreras.

CU-7Gestionar Carrera o Especialidades.

CU-8 Gestionar Fuente de Ingreso.

CU-9 Gestionar Eficiencia.

CU-10 Importa XML.

CU-11 Exportar XML.

Dentro de los casos de usos definidos en el portal y reutilizados en el módulo de administración de GeForza se encuentran:

CU-12 Gestionar Usuario.

CU-13 Gestionar Roles de Usuario.

CU-14 Gestionar Trazas de Usuario.

CU-15 Cambiar Contraseña.

2.4.3. Diagrama de casos de usos del sistema

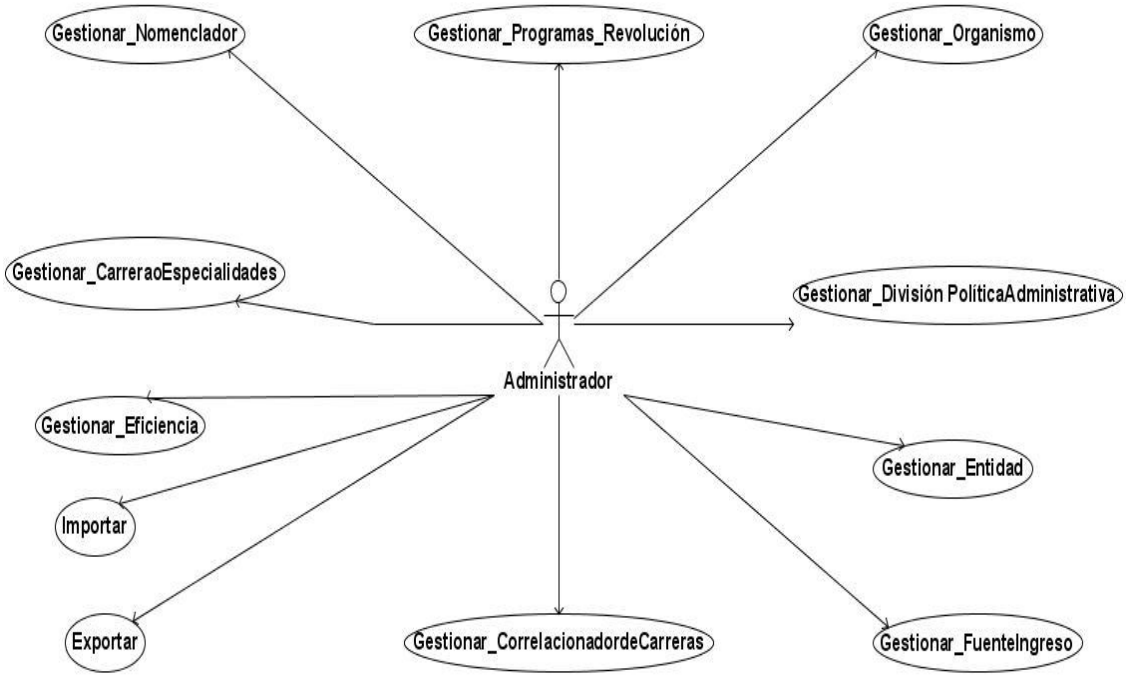

Figura 4. Diagrama de Caso de Uso del Sistema

2.4.4. Descripciones abreviadas de los casos de uso del sistema

Las descripciones completas se podrán encontrar en el Anexo # 3.

Caso de uso: Gestionar Organismo

Caso de Uso:	Gestionar Organismo
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar Organismo. A partir de ahí tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina, busca o imprime, según la opción deseada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF : 7, 22
Prioridad	Crítico.

Tabla # 2: Caso de uso del sistema: Gestionar Organismo.

Caso de uso: Gestionar Entidades

Caso de Uso:	Gestionar Entidad
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar Entidad. A partir de ahí el administrador tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina, busca o imprimir según la opción seleccionada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 6, 15, 16, 22
Prioridad	Crítico.

Tabla # 3: Caso de uso del sistema: Gestionar Entidad.

Caso de uso: Gestionar Carreras o Especialidades

Caso de Uso:	Gestionar Carrera o Especialidades.
Actores:	Administrador.
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar Carrera. A partir de ahí tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina, busca o imprimir según la opción seleccionada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 8, 22
Prioridad	Crítico.

Tabla # 4: Caso de uso del sistema: Carreras o Especialidades

Caso de uso: Gestionar División Política Administrativa.

Caso de Uso:	Gestionar División Política Administrativa.
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar División Política Administrativa. A partir de ahí tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina, busca o imprime según la opción seleccionada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y

	el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 9, 10, 22
Prioridad	Secundario

Tabla # 5: Caso de uso del sistema: Gestionar División Política Administrativa

Caso de uso: Gestionar Nomenclador

Caso de Uso:	Gestionar Nomenclador
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar Nomenclador. A partir de ahí tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir un nomenclador. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina o imprime según la opción seleccionada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 12, 13, 14, 18, 22

Tabla # 6: Caso de uso del sistema: Gestionar Nomenclador

Caso de uso: Gestionar Programas de la Revolución

Caso de Uso:	Gestionar Programas de la Revolución.
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar Programas de la Revolución. A partir de ahí tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina, busca, o imprimir según la opción seleccionada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 19, 22
Prioridad	Secundario

Tabla # 7: Caso de uso del sistema: Gestionar Programas de la Revolución

Caso de uso: Gestionar Correlacionador de Carreras.

Caso de Uso:	Gestionar Correlacionador de Carreras.
Actores:	Administrador

Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar Correlacionador. A partir de ahí tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina, busca o imprime según la opción deseada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 11, 22
Prioridad	Secundario

Tabla # 8: Caso de uso del sistema: Gestionar Correlacionador de Carreras

Caso de uso: Gestionar Eficiencia

Caso de Uso:	Gestionar Eficiencia.
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar Eficiencia. A partir de ahí tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina, busca o imprime según la opción seleccionada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 20, 21, 22
Prioridad	Secundario

Tabla # 9: Caso de uso del sistema: Gestionar Eficiencia

Caso de uso: Gestionar Fuente de Ingreso

Caso de Uso:	Gestionar Fuente de Ingreso.
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Gestionar Eficiencia. A partir de ahí tiene la opción de Adicionar, Modificar, Eliminar, Buscar o Imprimir. Posteriormente introduce los datos necesarios para realizar la operación deseada y el sistema adiciona, modifica, elimina, busca o

	imprime según la opción seleccionada, finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 17, 22
Prioridad	Secundario

Tabla #10: Caso de uso del sistema: Gestionar Fuente de Ingreso

Caso de uso Exportar XML.

Caso de Uso:	Exportar XML.
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Exportar XML., finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 23
Prioridad	Auxiliar

Tabla # 11: Caso de uso del sistema: Exportar XML

Caso de uso Importar XML.

Caso de Uso:	Importar XML.
Actores:	Administrador
Resumen:	El caso de uso comienza cuando el Administrador selecciona la opción Importar XML., finaliza el caso de uso cuando se realizan todas las operaciones deseadas y el administrador selecciona la opción salir.
Precondiciones:	El actor debe estar autenticado en el sistema.
Referencias	RF: 24
Prioridad	Auxiliar

Tabla # 12: Caso de uso del sistema: Importar XML

2.5. Conclusiones

Con la realización de este capítulo se inició el desarrollo de la propuesta de solución, se analizaron los procesos involucrados en el negocio, así como la obtención de un listado de funcionalidades que debe tener el sistema (expresados en los requisitos funcionales) donde se adquirieron finalmente los casos de uso y los artefactos principales correspondientes a este flujo de trabajo tales como la especificación de requisitos, diagramas de casos de uso y descripción de los casos de uso. Con la determinación de

los casos de uso, se está en condiciones de comenzar la elaboración del mismo, específicamente, de realizar el análisis y diseño del sistema, partiendo siempre, por supuesto, de las funcionalidades consideradas.

CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA

3.1. Introducción

En el siguiente capítulo se hace el análisis y diseño del Módulo de Administración del sistema GeForza, lo cual es de vital importancia para el desarrollo de software, pues constituye la vista lógica de la arquitectura. Se realizarán los diagramas de clases del análisis, diseño e interacción, los cuales garantizan un mejor entendimiento para el posterior desarrollo del módulo, siempre ajustando el resultado de estos diagramas a la arquitectura definida para el sistema y diseñados con la herramienta y lenguaje seleccionado. Además de realizar un estudio de la factibilidad del sistema. Para ello se realizará la estimación del esfuerzo y el tiempo de desarrollo del sistema en horas-hombre, utilizando el método de estimación por puntos de casos de uso

3.2. Análisis

El análisis es la etapa del flujo de trabajo análisis y diseño, que se encarga de refinar y estructurar los requisitos identificados anteriormente, con el objetivo de lograr una mejor comprensión y descripción de los mismos, que facilite estructurar el sistema en su totalidad.

Para ello se identifican un conjunto de clases que se definen como Clases del Análisis, las cuales se centran en los requisitos funcionales y son evidentes en el dominio del problema porque representan conceptos y relaciones del dominio. Tienen atributos y entre ellas se establecen relaciones de asociación, agregación / composición, generalización / especialización y tipos asociativos.

RUP propone clasificar a las clases en:

- ✓ Clase Interfaz: se utilizan para modelar la interacción entre el sistema y los actores.
- ✓ Clase Controladora: representan coordinación, secuencia, y control de otros objetos y se usan con frecuencia para encapsular el control de un caso de uso en concreto.
- ✓ Clase Entidad: se utilizan para modelar información que posee una vida larga y que a menudo es persistente. Modelan la información y el comportamiento asociado de algún fenómeno o concepto, como una persona, un objeto, o un suceso del mundo real.

3.2.1. Diagrama de clases del análisis

Una clase de análisis y sus objetos normalmente participan en varias realizaciones de casos de uso, y algunas de las responsabilidades, atributos, y asociaciones de una clase concreta suelen ser sólo relevantes para una única realización de caso de uso. Por tanto, es importante durante el análisis coordinar todos los requisitos sobre una clase y sus objetos que pueden tener diferentes casos de uso. A continuación se muestra el diagrama de clases del análisis de los caso de usos más significativos los demás se podrán ver en el Anexo # 4.

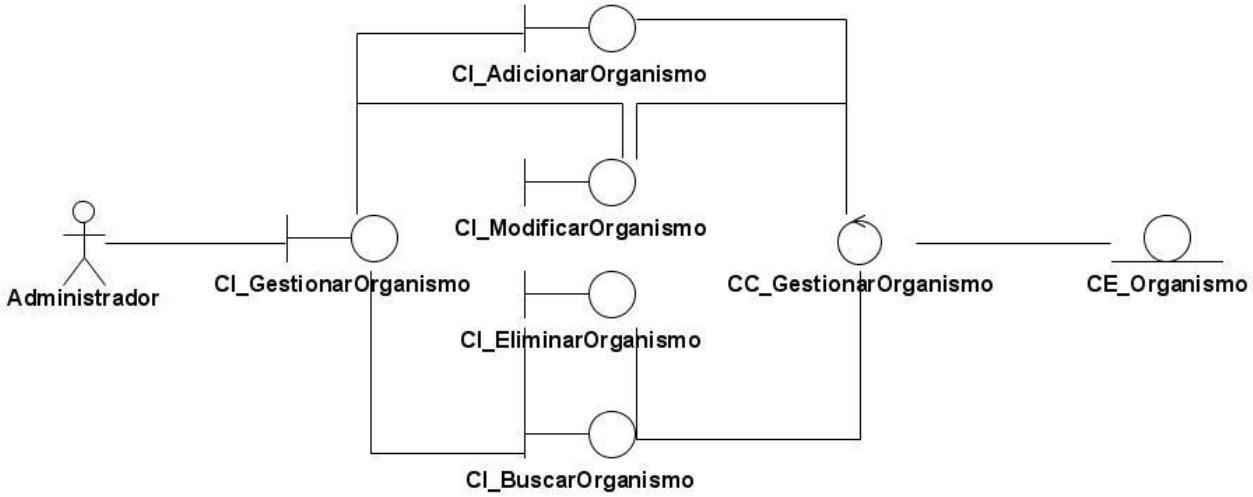

Figura 5: Diagrama de clases del análisis caso de uso: Gestionar Organismo.

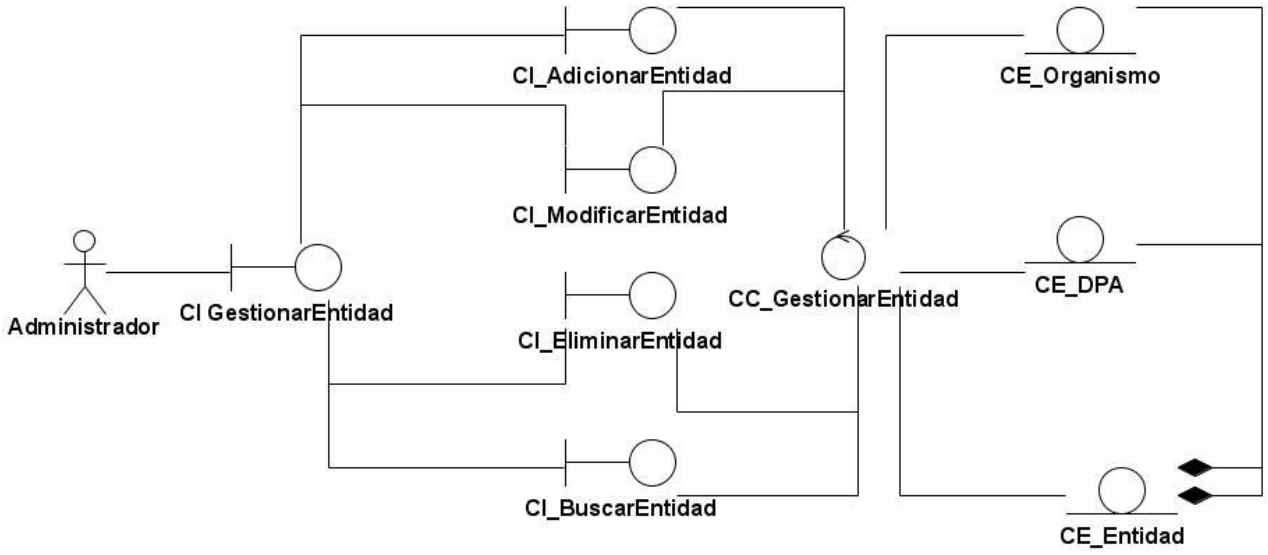

Figura 6: Diagrama de clases del análisis caso de uso: Gestionar Entidad.

3.2.2. Diagramas de Colaboración

Los diagramas de interacción representan una vista dinámica del sistema y constituyen la secuencia de acciones que ocurren cuando el actor comienza el caso de uso, así como los mensajes que se envían entre cada una de las clases. Los mismos se pueden clasificar en: diagramas de colaboración y diagramas de secuencia.

Durante la realización del análisis se utilizan los diagramas de colaboración, su principal objetivo es mostrar las interacciones entre objetos organizados entorno a objetos y los enlaces entre ellos.

En el Anexo # 5 se podrán apreciar los diagramas de colaboración para cada escenario de los casos de uso más significativos.

3.3. Diseño

Una entrada esencial del diseño es el resultado del modelo de análisis. El diseño será utilizado para visualizar la implementación y para soportar las técnicas de programación gráfica ya que se decidió realizar ingeniería inversa en esta fase.

3.3.1. Patrones de Diseño

Los patrones de diseño son una solución a un problema de diseño. Se consideran procedimientos para solucionar problemas del mismo tipo y son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software; Los desarrolladores lo usan como una forma de reutilizar la experiencia, clasificando las soluciones con términos de común denominación.

Para el diseño de la propuesta de solución del subsistema, se tiene en cuenta un patrón tradicional del diseño Web: arquitectura Modelo-Vista-Controlador, que implementa Zend Framework.

Modelo Vista Controlador (MVC) es un patrón que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos:

- ✓ Modelo: Encapsula los datos y las funcionalidades. Es independiente de cualquier representación de salida y/o comportamiento de entrada. Se encarga de la abstracción de la lógica relacionada con los datos, permitiendo que la vista y las acciones sean independientes.
- ✓ Vista: Muestra la información al usuario en forma de página web y le permite interactuar con ella.
- ✓ Controlador: Se encarga de procesar las interacciones del usuario y realiza los cambios necesarios en el modelo o en la vista. Mantiene aislado al modelo y a la vista de los detalles del protocolo utilizado para las peticiones.

Se utilizan además, patrones de asignación de responsabilidades (GRASP) y patrones Gang of Four (GOF).

Patrones GRASP usados

Experto: este patrón se tiene en cuenta para la asignación de responsabilidades a las clases de forma tal que las mismas contengan la información necesaria para poder ejecutar una acción específica. Las clases que brinda el marco de trabajo Ext JS se encargarán de visualizar las interfaces ya que cuentan con la información para crear los diferentes componentes visuales, las clases controladoras del Zend Framework manejarán las peticiones del cliente, y las clases que genera y utiliza el Doctrine serán las encargadas del acceso a datos pues contienen y representan los datos que manejará el sistema. El uso de este patrón permitirá a los objetos valerse de su propia información para hacer lo que se les pide, favorece la existencia de mínimas relaciones entre las clases, lo que permite contar con un sistema robusto y fácil de mantener.

Creador: este patrón se tiene en cuenta para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto sólo pueda ser creada por el objeto que contiene la información necesaria para ello. Existe un único script PHP que se encarga de instanciar al controlador frontal, este último es el encargado de instanciar las clases controladoras y estas, a su vez, instancian objetos de la clase Zend_View. El uso de este patrón permite crear las dependencias mínimas necesarias entre las clases, favoreciendo al mantenimiento del sistema.

Bajo acoplamiento: este patrón brinda como solución asignar responsabilidades de manera que las clases no dependan fuertemente unas de otras. De esta, forma las clases son fáciles de entender por separadas, fáciles de reutilizar y no se afectan por cambios de otros componentes. Dicho patrón se tiene en cuenta debido a la importancia de realizar un diseño de clases independientes que soporten los cambios.

Alta cohesión: este patrón propone asignar la responsabilidad de manera que la complejidad se mantenga dentro de límites manejables asumiendo solamente las responsabilidades que deben manejar, evadiendo un trabajo excesivo. Su utilización mejora la claridad y facilidad con que se entiende el diseño, simplifica el mantenimiento y las mejoras de funcionalidad, genera un bajo acoplamiento, soporta mayor capacidad de reutilización.

Controlador: este patrón se tiene en cuenta para realizar las asignaciones en cuanto al manejo de los eventos del sistema y definir sus operaciones. Zend Framework contribuye a la utilización de este patrón ya que define un Controlador Frontal (Front Controller) que implica que todas las solicitudes son

dirigidas a un único script PHP que se encarga de instanciar al controlador frontal y redirigir las llamadas.

Patrones GOF

Decorator (Envoltorio): este patrón permite añadir funcionalidad a una clase dinámicamente. Zend framework implementa dicho patrón en la clase Zend_View, encargada de asignarle responsabilidades a objetos de manera dinámica y configurarlos con nuevos atributos.

Singleton (Instancia única): este patrón garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Zend Framework tiene una instancia única del controlador frontal disponible mediante este patrón para lograr una vía de entrada única a las solicitudes.

Facade (Fachada): este patrón proporciona una interfaz para crear familias de objetos relacionados o dependientes sin especificar su clase concreta. Permite configurar en tiempo de ejecución un sistema con una familia u otra de objetos. Además garantiza que un conjunto de clases se usen a la vez. Zend Framework provee una API para el acceso a datos conformada por un conjunto de clases que implementa dicho patrón.

Factory (Fabricación): este patrón simplifica los accesos a las clases de la capa de acceso a datos proporcionando un objeto que todas las clases de capas superiores utilizarán para acceder a las clases contenidas en la capa del modelo. Define una interface de más alto nivel que permite usar el sistema más fácil. El objetivo de la aplicación de este patrón es reducir la dependencia entre clases. Se utilizará una clase intermediaria entre las clases controladoras de Zend Framework y las de acceso a datos de Doctrine, la que brindará, de las operaciones de acceso a datos, sólo las que necesiten los controladores para su funcionamiento, lo que reduce la dependencia de estos entre las múltiples clases de acceso a datos existentes en el sistema.

3.3.2. Diagrama de clases del diseño

Un diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación. La forma habitual de modelar estos diagramas no es factible cuando lo que se quiere diseñar es una aplicación web como es el caso, por tanto, se utiliza una extensión de UML para Web, que se ajusta a la arquitectura de este tipo de sistemas.

Los diagramas de clases del diseño contienen la siguiente información:

- ✓ Clases, asociaciones y atributos.
- ✓ Páginas y enlaces entre ellas.

A continuación se mostraran los diagramas de clase del diseño de los casos de usos más significativos, los demás se podrán ver en el Anexo # 5.

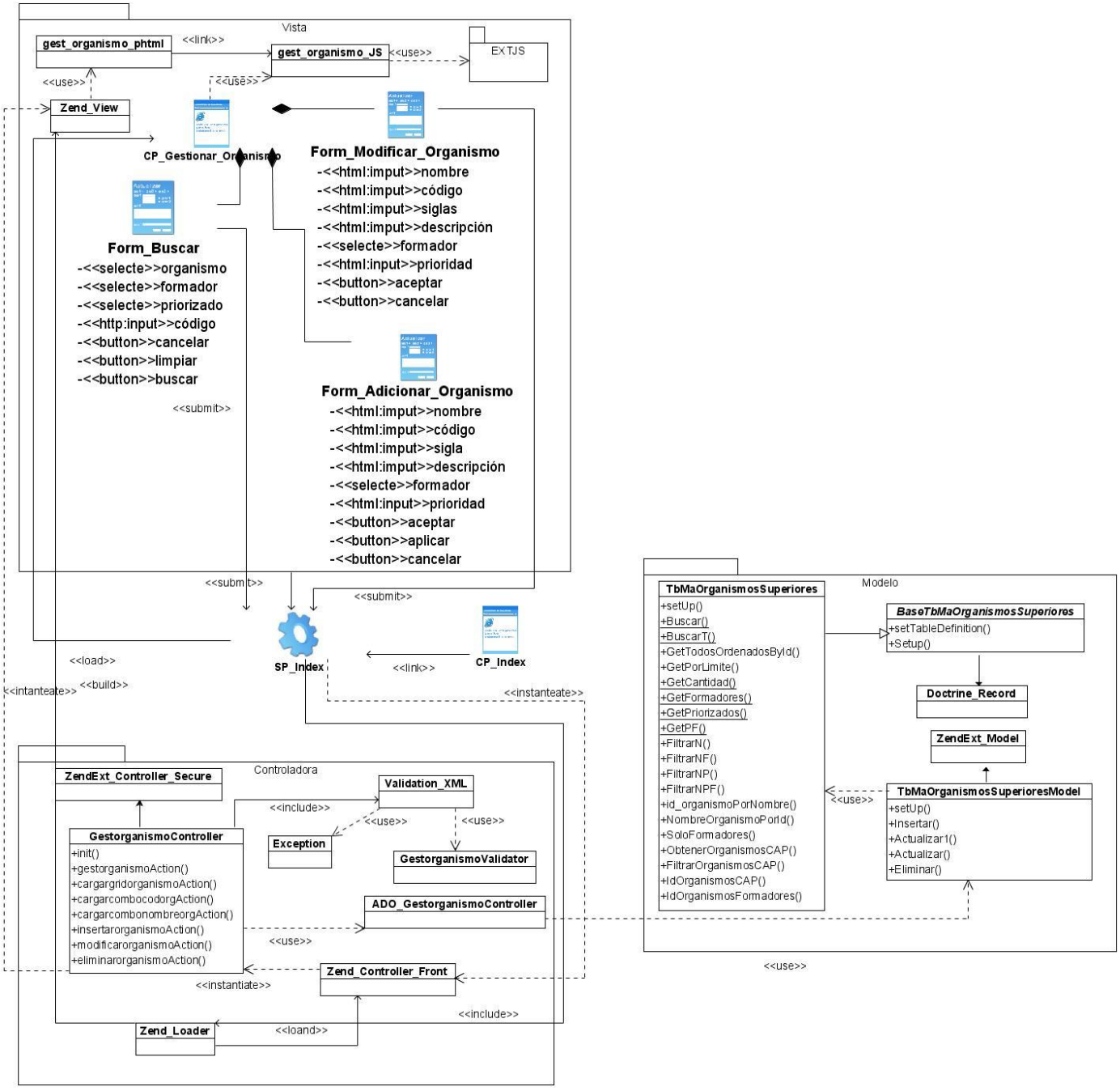

Figura 7: Diagrama de clases del diseño caso de uso: Gestionar Organismo.

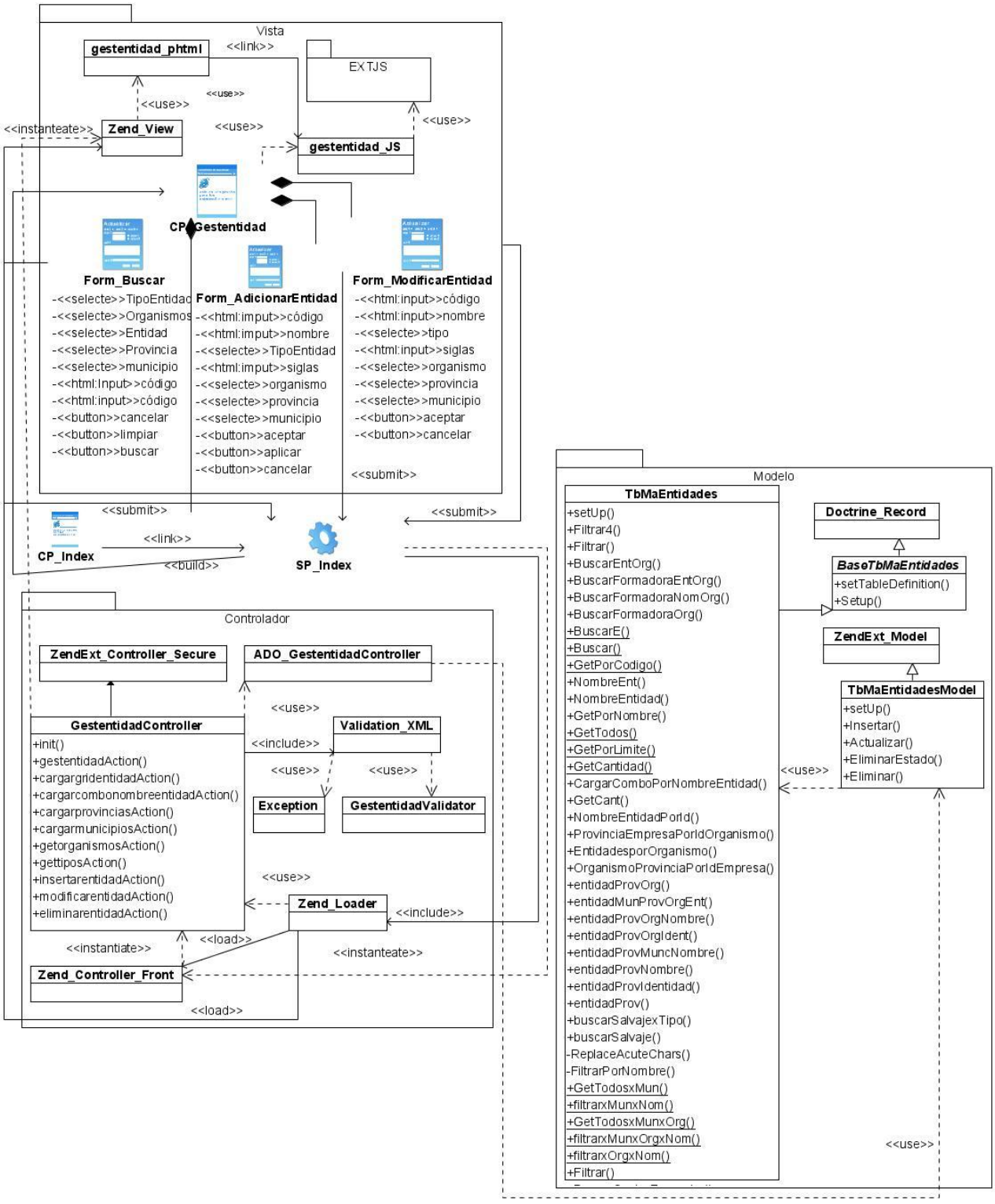

Figura 8: Diagrama de clases del diseño caso de uso: Gestionar Organismo.

3.3.3. Diagramas de secuencias

Los diagramas de secuencia, como se ha mencionado anteriormente, son un tipo de diagrama de interacción. Estos diagramas muestran como se relacionan los objetos entre ellos mediante secuencias de mensajes que se envían entre sí.

En el Anexo # 6 se encuentran los Diagramas de Secuencias de cada sección de los casos de usos más significativos.

3.4. Modelo datos

El modelo de datos describe la representación lógica y física de los datos persistentes usados por la aplicación. El modelo lógico y la descripción de cada tabla se podrán ver en el Anexo # 7.

3.4.1. Modelo Físico

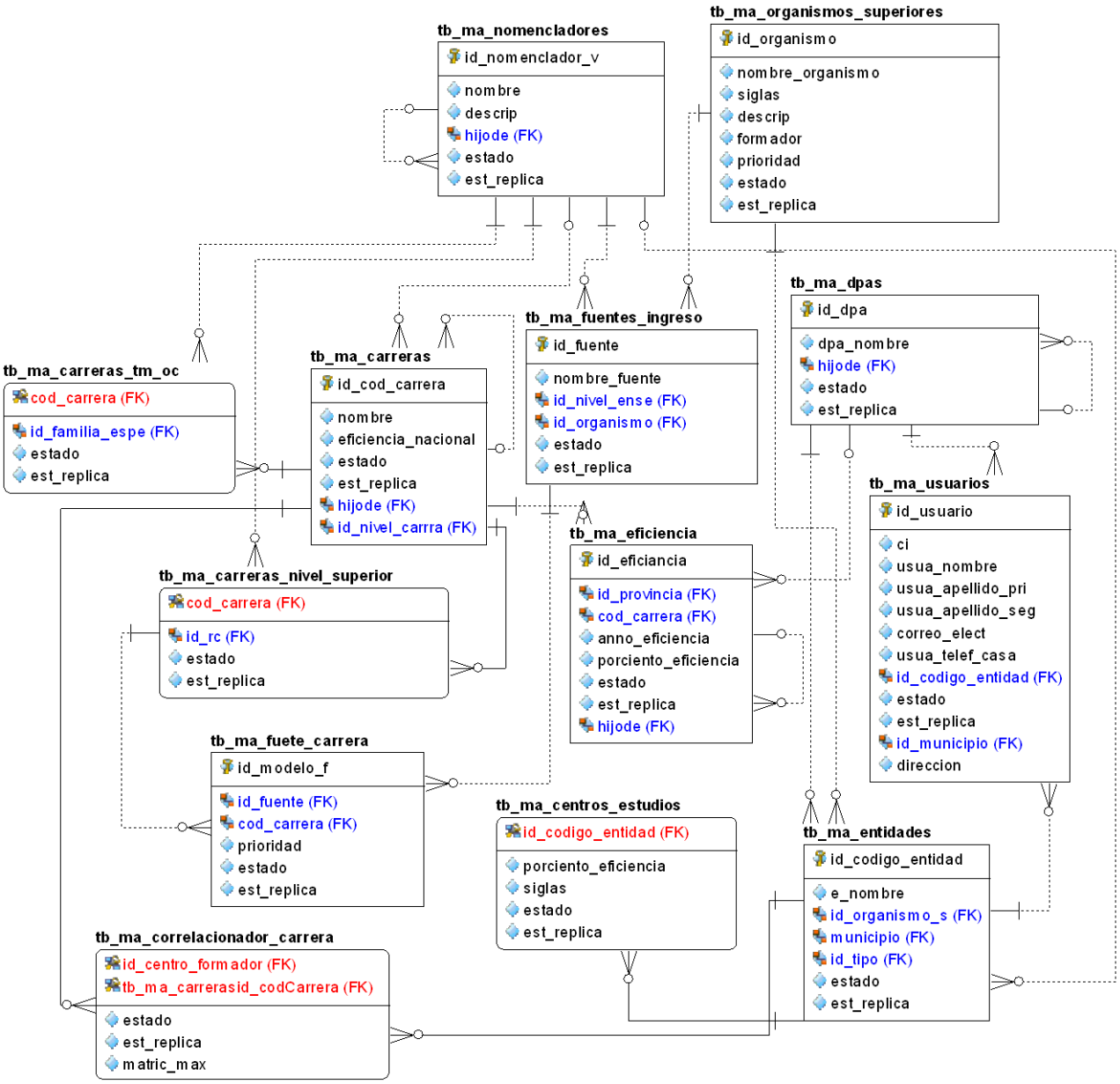

Figura 9: Modelo Físico.

3.5. Estimación del esfuerzo

“La estimación mediante el análisis de puntos de casos de uso es un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores”.

Mediante esta técnica existe una probabilidad de estimar el esfuerzo que el equipo de desarrollo empleará para elaborar el sistema, a partir de las características de sus requisitos, expresados en los casos de uso, por lo que es decisión de la universidad por las características de los proyectos la utilización de esta técnica. A continuación, se realiza la estimación del desarrollo del módulo donde se detallan los pasos a seguir para la aplicación de este método.

El primer paso para la estimación consiste en el cálculo de los puntos de casos de uso sin ajustar.

Se calcula a partir de la siguiente ecuación:

UUCP = UAW + UUCW donde:

UUCP: puntos de casos de uso sin ajustar

UAW: factor de peso de los actores sin ajustar

UUCW: factor de peso de los casos de uso sin ajustar

Factor de peso de los actores sin ajustar (UAW)

"Este valor se calcula mediante un análisis de la cantidad de actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los actores se establece teniendo en cuenta en primer lugar si se trata de una persona o de otro sistema y en segundo lugar la forma en la que el actor interactúa con el sistema".

Actores	Descripción	Complejidad	Factor de Peso
Administrador	Una persona que interactúa con el sistema mediante una interfaz gráfica.	Complejo	3

Tabla #13. Factor de peso de los actores sin ajustar (UAW).

UAW = $\sum(A_i \times FP_i)$

Cantidad de actores de tipo complejo: 1

UAW = 1 * 3

UAW = 3

Factor de peso de los casos de uso sin ajustar

"Este valor se calcula mediante un análisis de la cantidad de casos de uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los casos de uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como

una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia y está representada por uno o más pasos del flujo de eventos principal del caso de uso, pudiendo existir más de una transacción dentro del mismo caso de uso"

Caso de Uso	Descripción	Complejidad	Factor de Peso
CUS Gestionar Organismos	7 transacciones	Medio	10
CUS Gestionar Entidades	7 transacciones	Medio	10
CUS Gestionar Nomencladores	7 transacciones	Medio	10
CUS Gestionar Programas de la Revolución	7 transacciones	Medio	10
CUS Gestionar DPA	7 transacciones	Medio	10
CUS Gestionar Correlacionar de Carreras	7 transacciones	Medio	10
CUS Gestionar Eficiencia	7 transacciones	Medio	10
CUS Gestionar Carrera o Especialidades	7 transacciones	Medio	10
CUS Gestionar Fuentes de Ingreso	7 transacciones	Medio	10
CUS Importar	2 transacciones	Simple	5
CUS Exportar	2 transacciones	Simple	5

Tabla # 14. Factor de peso de los casos de uso sin ajustar.

Factor de peso de los casos de uso sin ajustar (UUCW):

UUCW = $\sum (CU_i \times FP_i)$

UUCW = Sumatoria de los casos de uso por su factor de peso.

UUCW = 9 * 10 + 2 * 5

UUCW = 100

Finalmente, los puntos de casos de uso sin ajustar resultan

UUCP = UAW + UUCW

UUCP = 3 + 100

UUCP = 103

El segundo paso es calcular los puntos de casos de uso ajustados.

UCP = UUCP * TCF * EF donde,

UCP: puntos de casos de uso ajustados

UUCP: puntos de casos de uso sin ajustar

TCF: factor de complejidad técnica

EF: factor de ambiente

Factor de complejidad técnica

Para calcular este coeficiente se analizan factores que pueden determinar la complejidad técnica del sistema, a estos factores se les asocia un valor de 0 a 5 que determina el vínculo del mismo con las características deseadas para el sistema.

Factor	Descripción	Peso	Valor Asignado	Peso * Valor
T1	Sistema distribuido	2	0	0
T2	Objetivos de performance o tiempo de respuesta	1	3	3
T3	Eficiencia del usuario final	1	1	1
T4	Procesamiento interno complejo	1	0	0
T5	El código debe ser reutilizable	1	3	3
T6	Facilidad de instalación	0.5	1	0.5
T7	Facilidad de uso	0.5	5	2.5
T8	Portabilidad	2	3	6
T9	Facilidad de cambio	1	2	2
T10	Concurrencia	1	3	3
T11	Incluye objetivos especiales de seguridad	1	3	3
T12	Provee acceso directo a terceras partes	1	5	5
T13	Se requieren facilidades especiales de entrenamiento a usuarios	1	1	1

Tabla # 15. Factor de complejidad técnica

El Factor de complejidad técnica resulta:

TCF = 0.6 + 0.01 * Σ (Peso * Valor asignado)

TCF = 0.6 + 0.01 * (0+3+1+0+3+0.5+2.5+6+2+3+3+5+1)

TCF = 0.6 + 0.01 * 30

TCF = 0.6 + 0.3

TCF = 0.9

Factor de ambiente

Para calcular este coeficiente se analizan una serie de factores que pueden determinar el tiempo requerido para el desarrollo del sistema, teniendo en cuenta aspectos como habilidades,

conocimientos, etc. de los involucrados en la realización del sistema. A estos factores se les asocia un valor de 0 a 5 que determina el vínculo del mismo con las características deseadas para el sistema.

Factor	**Descripción**	Peso	Valor Asignado	Peso * Valor
E1	Familiaridad con el modelo de proyecto utilizado	1.5	5	7.5
E2	Experiencia en la aplicación	0.5	3	1.5
E3	Experiencia en orientación a objetos	1	4	4
E4	Capacidad del analista líder	0.5	2	1
E5	Motivación	1	5	5
E6	Estabilidad de los requerimientos	2	2	4
E7	Personal part-time	-1	0	0
E8	Dificultad del lenguaje de programación	-1	1	-1

Tabla # 16. Factor de ambiente.

El Factor de ambiente resulta:

EF = 1.4 – 0.03 * Σ (Peso * Valor asignado)

EF = 1.4 – 0.03 * Σ (7.5+1.5+4+1+5+4+0-1)

EF = 1.4 – 0.03 * 22

EF = 1.4 – 0.66

EF = 0.74

Finalmente, los puntos de casos de uso ajustados resultan:

UCP = UUCP * TCF * EF

UCP = 103 * 0.9 * 0.74

UCP = 68.598

El esfuerzo en horas-hombre viene dado por:

E = UCP * CF donde,

E: esfuerzo

UCP: puntos de casos de uso ajustados

CF: factor de conversión

Total EF = 1 + 0 Total

EF = 1

CF = 20 horas-hombre (si Total EF ≤ 2)

CF = 28 horas-hombre (si Total EF = 3 ó Total EF = 4)

CF = abandonar o cambiar proyecto (si Total EF ≥ 5)

Para este tipo de proyecto 20 horas-hombre/punto de casos de uso, es decir, un punto de caso de uso toma 20 horas-hombre.

E = UCP * CF

E = 81.918 * 20

E = 1371.96 horas-hombres

Se considera que este esfuerzo representa un porcentaje del esfuerzo del flujo de trabajo de implementación. Para una estimación más completa de la duración total del proyecto hay que agregar a la estimación del esfuerzo obtenida por los puntos de casos de uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo de software. Teniendo en cuenta los siguientes valores porcentuales para la distribución del esfuerzo entre las diferentes actividades de un proyecto, que estadísticamente se considera aceptable, se obtiene:

Actividad	Porcentaje	Horas-hombre
Análisis	10.00 %	342.99
Diseño	20.00 %	685.98
Programación	40.00 %	1371.96
Pruebas	15.00 %	514.85
Sobrecarga (otras actividades)	15.00 %	514.85
Total	100.00 %	3429.2

Tabla # 17 Distribución del esfuerzo

Esfuerzo Total (horas-hombres) 3429.2. Para la etapa de análisis y diseño se requiere de un esfuerzo de 1028.97 horas-hombres, si se considera que trabajan 2 personas, 80 horas como promedio en la semana, este sistema debe terminarse en 21,436875 semanas, lo que representa un aproximado de 21 semanas, cinco meses y una semana aproximadamente.

3.6. Costo *del proyecto*

Ahora se van a analizar los costes asociados al desarrollo del producto.

C = **CHM** * **E**T Donde:

C: costo.

CHM: cantidad de hombre-mes.

ET: esfuerzo total del proyecto.

Cálculo de la cantidad de hombres-mes (**CHM**)

CHM =**CH** * **SBM** Donde:

CH: cantidad de hombres trabajando en el proyecto

SBM: salario básico por hombre-mes

SBM = $100

CH = 2 Por lo que:

CHM = 2 * 100

CHM = 200 pesos/mes.

Entonces:

C = 200 * 5.1

C = **$ 1020.**

Por tanto para el desarrollo del producto se obtiene un costo de 1300 pesos.

3.7. Beneficios tangibles e intangibles

Tangibles

- ✓ El desarrollo de un módulo de administración que contribuirá con la seguridad y la homogeneización de los clasificadores utilizados por los diferentes subsistemas del sistema GeForza.

Intangibles

- ✓ Gestión la información de forma rápida y segura, puesto que todo el flujo de información debe realizarse mediante la red.
- ✓ Impresión de información desde el sistema.
- ✓ Búsquedas rápidas y eficientes.
- ✓ Contará con una documentación que sustentará su desarrollo para posteriores versiones.
- ✓ Mayor seguridad en el intercambio de información a través de la red y con usuarios autenticados.
- ✓ Trabajo más rápido y efectivo, asegurando mayor calidad.
- ✓ Solución a los problemas encontrados en la aplicación que se usa actualmente.
    - o Homogeneización de los clasificadores.
    - o Implementación en software libre.
    - o Interfaz sencilla y amigable.
    - o Información segura.

3.8. Conclusiones

Como resultado de la elaboración de este capítulo se logró la terminación de la etapa de análisis y diseño del subsistema, se realizaron los diagramas de clases del análisis y del diseño para cada caso de uso identificado. Se realizaron además los diagramas de interacción (diagramas de secuencia y

diagrama de colaboración) para cada escenario de los casos de uso. Como parte de este capítulo se describe la arquitectura y los patrones de diseño utilizados, se muestran los modelos físicos y lógicos de datos propuestos para satisfacer las necesidades del subsistema, por último, se realiza una estimación del esfuerzo donde se obtuvo el esfuerzo total, el tiempo de duración del módulo y el costo total.

IMPLEMENTACIÓN Y PRUEBA

4.1. Introducción

Este capítulo es el último del presente trabajo, el cual muestra la construcción de los artefactos esenciales de los flujos de trabajo Implementación y Prueba. Donde se muestra el diagrama de despliegue, los diagramas de componentes y el modelo de prueba con sus respectivos casos de prueba.

4.2. Modelo despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo.

A continuación se muestra la distribución física del proyecto “GeForza” teniendo en cuenta la arquitectura del software y del hardware.

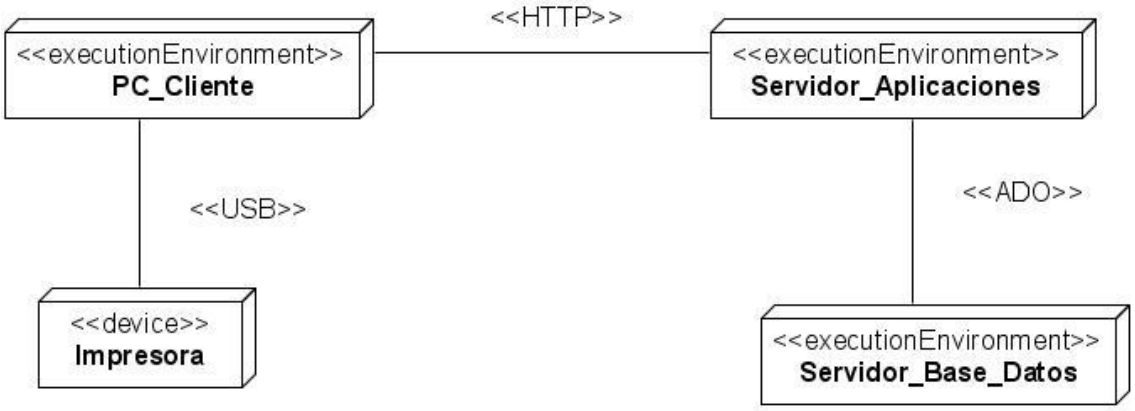

Figura 10: Modelo Despliegue.

4.3. Implementación

Este flujo de trabajo enmarca el comienzo de la fase de Construcción. Esta fase tiene como propósito dejar listo un producto software en su versión operativa inicial (versión beta). A esta versión le incumbe tener la calidad requerida para su uso y cumplir con los requisitos de software determinados en el segundo capítulo.

El flujo de implementación tiene como propósito:

- ✓ Definir la organización del sistema en términos de Subsistemas de Implementación organizados en capas.
- ✓ Implementar los elementos de diseño en términos de “Elementos de Implementación” (ficheros Fuentes, binarios, ejecutables y otros).
- ✓ Probar los componentes desarrollados independientemente como unidades.
- ✓ Integrar los resultados producidos por desarrolladores independientes o equipos en un sistema ejecutable.

4.3.1. Diagrama de Componentes

Un diagrama de componentes se representa como un grafo de componentes unidos por medio de relaciones de dependencia (compilación, ejecución), pudiendo mostrarse las interfaces que estos soporten. Cada componente representa una parte modular del sistema, desplegable y reemplazable que encapsula implementación y un conjunto de interfaces y proporciona la realización de los mismos.

Estos diagramas son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos. Dicho diagrama contiene los nodos que forman la topología hardware sobre la que se ejecuta el sistema y la distribución de las partes del sistema en ellos.

A continuación se podrá apreciar los diagramas de componentes de los casos de usos del sistema más significativos, los demás se podrán ver en el anexo # 8.

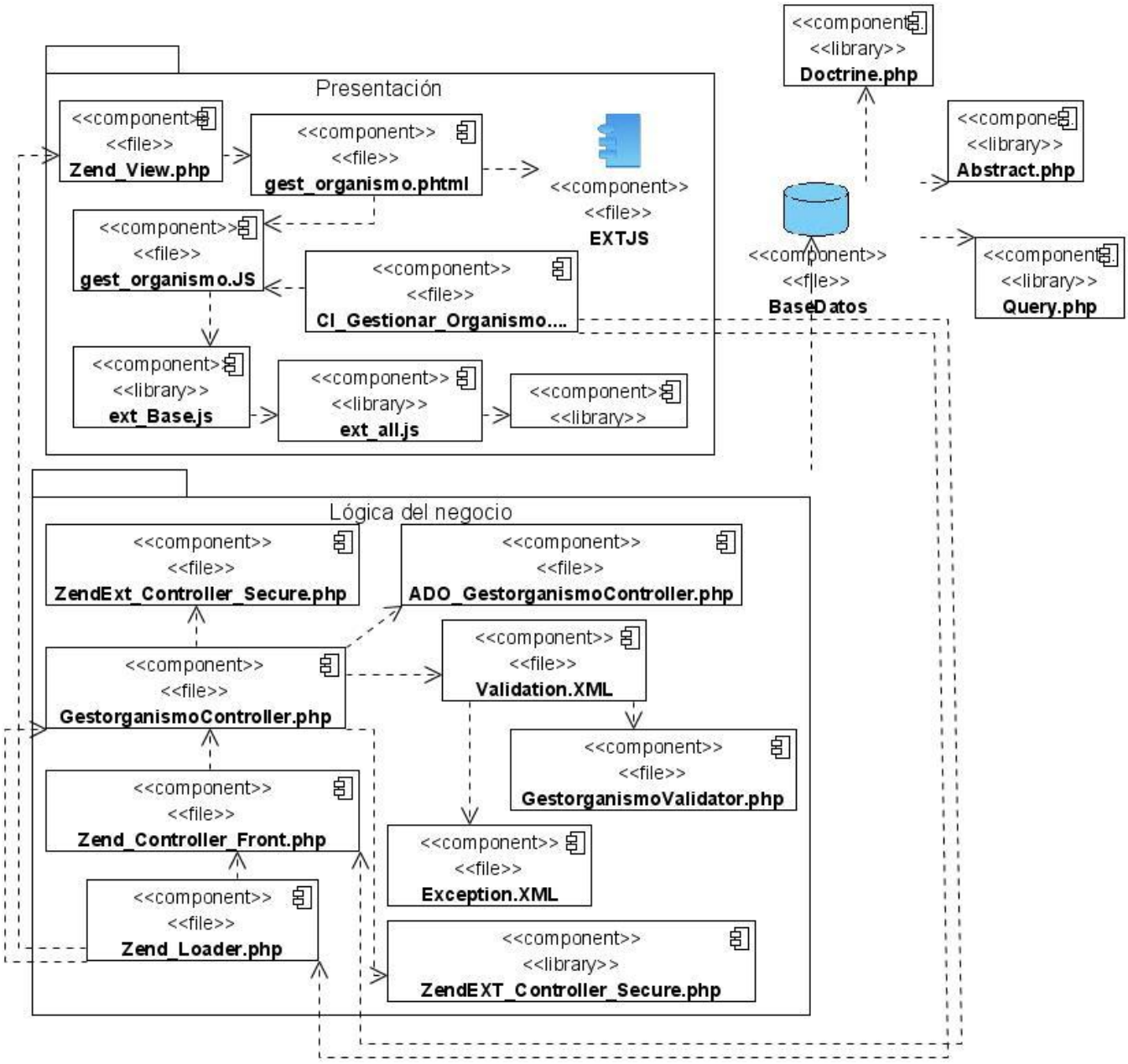

Figura # 11: Diagrama de componentes caso de uso: Gestionar Organismo.

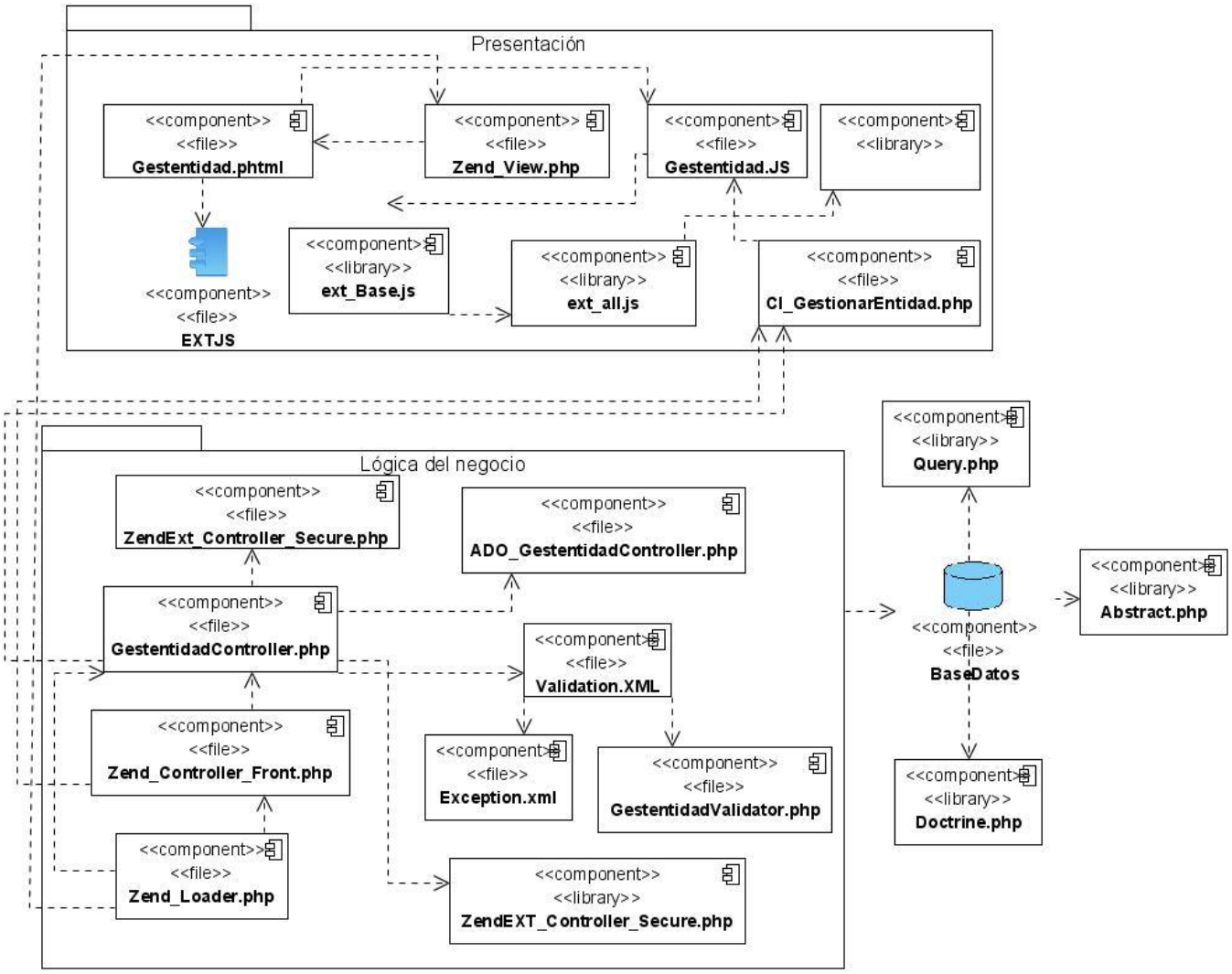

Figura # 12: Diagrama de componentes caso de uso: Gestionar Entidad.

4.4. Tratamiento de errores

Para el tratamiento de errores se validan los datos que son introducidos por los usuarios al sistema. Se valida que el formato de los datos sea el esperado y que no se omita información de importancia para el procesamiento de solicitudes. En estos casos se informa al usuario mediante un mensaje que ha realizado alguna operación incorrecta. Cada usuario tendrá acceso restringido según su rol y se le permitirá únicamente ver las funcionalidades para las cuales obtenga permisos.

4.5. Seguridad

Para garantizar la seguridad de la aplicación se realizó un estudio de las tendencias actuales y métodos más utilizados y seguros de hoy en día, como se demuestra en el capítulo I. Como resultado

de dicho estudio se decidió utilizar el módulo de seguridad del sistema Cedrux en la aplicación. Obteniendo de este modo un sistema seguro que garantice la confiabilidad de la información, la cual básicamente está diseñada de la siguiente manera.

El subsistema de seguridad está dividido en 4 módulos: Configurar nomencladores, Configurar sistemas, Configurar servidores y Configurar usuarios.

El módulo Configurar nomencladores permite el manejo de los dominios, base de datos, gestores de base de datos, esquemas, idiomas, temas, escritorios, expresiones y claves, para el manejo posterior de los servidores, sistemas y usuarios.

El módulo Configurar servidores permite manejar los datos de los servidores, gestores de base de datos, base de datos y esquemas de base de datos.

El módulo Configurar sistemas posee las funcionalidades correspondientes al manejo de los sistemas, las funcionalidades, acciones, servicios que brinda o consume sus funciones y parámetros de las mismas.

El módulo Configurar usuarios permite la gestión de los usuarios, roles, perfiles de usuario y los campos del perfil de usuario.

4.5.1. Auditoría de sistemas

Las trazas permiten en el desarrollo de software crear un mecanismo de registro oficial de eventos durante un período de tiempo en particular, además de registrar datos o información sobre quién, que, cuando, donde y por qué un evento ocurre para un dispositivo en particular o aplicación. Todo esto permite monitorear las actividades de la aplicación o dispositivo, donde se puede obtener una buena oportunidad para determinar eventos y tomar la acción necesaria para corregir el problema o iniciar una investigación en caso de un incidente de seguridad.

La auditoría de las aplicaciones se realiza mediante la utilización del componente Traza. Este componente permite el registro de los eventos que ocurren en el sistema permitiendo de esta forma poder corregir problemas que se presenten, el mismo puede ser instalado en un servidor independiente para garantizar la confidencialidad de los datos. Se podrían controlar eventos como:

- ✓ Inicio de una acción: Se dispara un evento al comienzo de una acción.
- ✓ Terminación de una acción: Se dispara un evento al finalizar una acción.
- ✓ Error en una acción: Se dispara un evento cuando en una acción ocurre un error.
- ✓ Autenticar usuario: Se dispara un evento por cada URL visitada.
- ✓ IoC Externo: Se dispara un evento cuando ocurre integración entre diferentes subsistemas.
- ✓ IoC Interno: Se dispara un evento cuando ocurre integración entre diferentes componentes de un subsistema.

- ✓ Excepciones: Se dispara un evento cuando ocurre una excepción en el sistema.
- ✓ Rendimiento: Es el tiempo de ejecución de una acción.
- ✓ Error IoC Externo: Se dispara un evento cuando ocurre un error en la integración entre diferentes subsistemas.
- ✓ Error IoC Interno: Se dispara un evento cuando ocurre un error en la integración entre diferentes componentes de un subsistema.

4.5.2. Lista de chequeo

Entidad	Parámetros	
Entidad 1	**Sistema operativo.**	Utilización del SO Linux en los servidores.
	Tecnologías en los servidores.	Servidor web Apache
		Servidor de base de datos Postgres 8.3
		Navegador Web Mozilla Firefox 2.0.17 ó superior.
		No tener instaladas extensiones de Mozilla Firefox.
		Lenguaje de programación PHP 5.2.4
		Actualización de los parches de seguridad.
		Opciones de explorar directorio desactivadas
		Restringir acceso por direcciones ip.
		Apache debe funcionar bajo su propia cuenta y grupo de usuario.
		Módulos innecesarios deshabilitados: mod_imap, mod_include, mod_info, mod_status, mod_cgi, mod_autoindex.

	Configuración del servidor web.	Los includes del lado servidor desactivados**.**
		Ejecución de CGI desactivada.
		Desactivar error, notice, alert, warn y demás opciones que brinden información sensible.
		No permitir que apache siga enlaces simbólicos. Desactivar todas las opciones.
		Desactivar la ayuda para los archivos .htaccess
		Disminuye el valor máximo de tiempo de espera. Timeout 60
		Limitar el tamaño máximo de peticiones LimitRequestBody 2097152
		Ejecuta Apache en un entorno Chroot.
	Configuración de PHP.	Desactivar el acceso a ficheros remotos.
		Register globals desactivado.
		Acceso restringido a los ficheros puede acceder PHP.
		Modo seguro activado.
		Límites controlados max_execution_time = 30 max_input_time = 60 ; memory_limit = 16M ; upload_max_filesize = 2M

		post_max_size = 8M ;
		Control de acceso a ficheros mediante Apache.
		Carga dinámica de módulos desactivada.
		Evitar el acceso a la Shell.
		Deshabilitar información sobre php en las cabeceras del servidor.
		Funciones de integración de php/Apache desactivadas.
		Creación de un nuevo directorio para almacenamiento de las sesiones de usuario.
		Al directorio donde se almacenan las sesiones sólo puede tener acceso el usuario Apache.
	Principios básicos de seguridad en la BD.	El administrador de la BD y el administrador del sistema operativo (ASO) no pueden ser la misma persona.
		Los privilegios del usuario operador y los del ABD no deben ser los mismos.
		El número mínimo de caracteres de las contraseñas de los usuarios será de 7 caracteres.
		Los usuarios no deben compartir sus cuentas ni sus contraseñas.
		Habilitar sólo los servicios y puertos requeridos.
		La cuenta de root y de administrador sólo la debe tener el personal requerido.
		Restringir las conexiones remotas usando el fichero pg_hba.conf.

		La contraseña del usuario no debe ser igual al nombre del usuario.
		La contraseña debe tener fortaleza requerida con la utilización de letras, números y caracteres especiales.
		La contraseña debe cambiarse como máximo cada tres meses.
		Uso de un firewall que mantenga protegido al servidor.
		El firewall configurado para que sólo tenga acceso al servidor la red o subred requerida.
		Habilitar en el postgresql.conf que postgres escuche todas las peticiones, tanto locales como remotas. listen_addresses = 'ALL'
		Activar la encriptación de contraseñas de la base de datos a la hora de conectarse en el postgresql.conf. password_encryption = on
		No se debe permitir en la configuración de las conexiones en el fichero pg_hba.conf: DATABASE ALL USER ALL METHOD TRUST
		Crear un usuario de base de datos por cada sistema.
		Para cada usuario de base de datos especificar

		en el pg_hba.confç: BD, rango de ip, y método de encriptación MD5.
		Uso del auditor de código fuente (CRC) para el control de la integridad de las aplicaciones.
		Ofuscación de código.
	Mecanismos de control de integridad.	Cálculo de CRC.
	Ofuscador de código.	Deben crearse dominios que contengan exactamente las estructuras a las que tiene acceso uno o varios usuarios.
	CRC	Deben registrarse la estructura de los sistemas hasta el nivel de acción para garantizar la seguridad hasta la base.
	Seguridad de las aplicaciones	Deben crearse roles que contengan exactamente los sistemas, funcionalidades y acciones a las cuales tiene acceso uno o varios usuarios.
		La configuración de las claves de usuarios debe contener signos, letras y números, contar con un número mayor de 7 caracteres y caducar en un tiempo de 60 días.
		Existencia de un plan de contingencia.
		Existencia de un plan de seguridad informática.

	Seguridad informática.	Control de acceso a los locales y las PCs.
		Existencia de una política de salva de seguridad.

Tabla # 18 Lista de chequeo

4.6. Estándares de implementación

En la Vista

Para esta capa se definen como estándares los siguientes:

- ✓ En el nombre de las clases debe comenzar con gest en el caso de los casos de uso gestionar, seguido por el nombre del caso de uso. Ejemplo: gestorganismo, gestentidad.
- ✓ El nombre de la clase debe comenzar con gen en el caso de los casos de uso generar, seguido por el nombre del caso de uso. Ejemplo: gennomencladores.
- ✓ Teniendo en cuenta el componente que se utilizará el nombre del mismo estará definido de la siguiente forma:
    - Los botones se nombrarán: btn+Nombre. Ejemplo: btnAddOrganismo.
    - Los stores se nombrarán: st+Nombre. Ejemplo: stProv stOrg.
    - Los combobox se nombrarán: cb+Nombre. Ejemplo: cbProvincia.
    - Los texfields se nombrarán: txf+Nombre. Ejemplo: txfOrganismo.
    - Los gridpanels se nombrarán: gp+Nombre. Ejemplo: gpNomenclador.
    - Las ventanas se nombrarán: win+Nombre. Ejemplo: winAddDemanda

En el Negocio

Para esta capa se definen:

- ✓ El nombre de la clase comenzará con Gest seguida por el nombre si el caso de uso es un gestionar y la palabra Controller. Ejemplo: GestorganismoController.
- ✓ El nombre de la clase comenzará con Gen seguida por el nombre si el caso de uso es un generar y la palabra Controller. Ejemplo: GennomencladoresController.
- ✓ El nombre de los atributos será en letra minúscula y en caso de ser más de una palabra se escribirá sin separación. Ejemplo: datosfinal.
- ✓ El nombre de las funciones será nombre seguido por la palabra Action. Ejemplo: cargarcomboprovinciaAction.

- ✓ Los servicios se nombrarán Gest seguido del nombre y la palabra Service. Ejemplo: GestorganismoService.
- ✓ Las validaciones se nombrarán Gest seguido del nombre y la palabra Validator. Ejemplo: GestorganismoValidator.

En el Modelo

Para esta capa se utilizan:

- ✓ El nombre de las entidades se define por tb seguido de las siglas del módulo y el nombre separando cada una de las palabras por un "_". Ejemplo: tb_ma_organismos.
- ✓ El nombre de los atributos será en letra minúscula y en caso de ser más de una palabra serán separadas por un "_". Ejemplo: id_código.

4.7. Pruebas

Las pruebas de software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación. (Pressman, 2002).

En el desarrollo de un software, el proceso de prueba es clave a la hora de detectar errores o fallas. Las pruebas son de gran importancia en la garantía del software, una selección cuidadosa de los datos de prueba puede ofrecer mucha confianza en cuanto al desempeño que posee el programa. Esto, asociado a un determinado mecanismo de comprobación de errores, puede producir software más confiable. El objetivo principal de la realización de una prueba es obtener un conjunto de pruebas que tengan la mayor posibilidad de descubrir los defectos del software.

4.7.1. Métodos de prueba.

Existen dos métodos de prueba fundamentales:

- ✓ Prueba de caja blanca (prueba de caja de cristal): se centran en la estructura de control del programa. Se obtienen casos de prueba que aseguren que durante la prueba se han ejecutado, por lo menos una vez, todas las sentencias del programa y que se ejercitan todas las condiciones lógicas. (Pressman, 2002)
- ✓ Prueba de caja negra (prueba de comportamiento): son diseñadas para validar los requisitos funcionales sin fijarse en el funcionamiento externo del programa. Se centran en el ámbito de información del programa, de forma que se proporcione una cobertura completa de prueba. No son una alternativa a las técnicas de pruebas de caja blanca. (Pressman, 2002)

Para desarrollar la prueba de caja negra existen varias técnicas, entre ellas están:

- ✓ Los métodos de pruebas basados en grafos: exploran las relaciones entre los objetos del programa y su comportamiento.
- ✓ La partición equivalente: esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- ✓ El análisis de valores límite: prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- ✓ La prueba de la tabla ortogonal: suministra un método sistemático y eficiente para probar sistemas con un número reducido de parámetros de entrada.

Se utilizará el método de caja negra, aplicando la técnica de particiones equivalentes que está definido en la estrategia de pruebas trazada por el proyecto Fuerza de Trabajo Calificada.

4.7.2. Casos de prueba.

Un caso de prueba específica una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las que ha de probarse. (Jacobson, 2000)

A continuación se podrán apreciar los casos de pruebas de los casos de usos más significativos, los demás se podrán ver en el anexo # 9.

Caso de prueba del caso de uso: Gestionar Organismo

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC1: Adicionar Organismo	1: Adicionar un organismo correctamente.	Se adiciona un nuevo organismo al sistema.	-Se selecciona la opción “Adicionar”. - Llenar todos los campos - Seleccionar “Aplicar”. - Seleccionar “Aceptar”. - Muestra un mensaje indicando que el organismo fue creado con éxito.

	2: Adicionar un organismo dejando campos en blanco.	No se adiciona un nuevo organismo al sistema.	-*Se selecciona la opción "Adicionar".* - Llenar los campos dejando campos en blanco. - Seleccionar "Aplicar" *.- Muestra un mensaje de error "Datos incorrectos, por favor revise otra vez"* -Seleccionar "Aceptar"
	3: Adicionar un organismo pasando datos incorrectos	Se adiciona un nuevo organismo al sistema.	-*Se selecciona la opción "Adicionar".* - Llenar los campos pasando datos incorrectos. - *Seleccionar "Aplicar".* *-Se muestra un mensaje indicando los campos incorrectos.* -Seleccionar "Aceptar"
	4: Cancelar la operación	No se adiciona un nuevo organismo al sistema.	-*Se selecciona la opción "Adicionar".* - Llenar los campos. - Seleccionar "Cancelar".
SC2: Modificar Organismo	1: Modificar Organismo correctamente	Se modifica un organismo al sistema.	*-Se selecciona un organismo* *-Se selecciona la opción "Modificar"* *- Se cambian los datos deseados.* *- Seleccionar ¨Aceptar.¨* *- Muestra un mensaje indicando que el organismo fue modificado con éxito.*

	2: Modificar Organismo pasando datos incorrectos	No se modifica el organismo al sistema.	*-Se escoge un organismo* *-Se selecciona la opción "Modificar "* *- Se cambian los datos deseados pasando datos incorrectos o dejando campos vacíos.* *-Seleccionar ¨Aceptar.¨* -Se muestra un mensaje indicando los campos incorrectos. -Seleccionar "Aceptar"
	3: Cancelar operación	No se modifica organismo al sistema.	*-Se escoge un organismo* *-Se selecciona la opción "Modificar".* *- Se cambian los datos deseados.* - Seleccionar "Cancelar".
SC3: Eliminar Organismo	1: Eliminar organismo correctamente.	Se elimina un organismo al sistema.	*-Se selecciona un Organismo.* *-Se selecciona la opción "Eliminar".* *-Muestra un mensaje de comprobación:"* Seguro que desea eliminar el organismo*"* -Seleccionar la opción "Aceptar" -Muestra un mensaje indicando que se eliminó el organismo.
	2: Cancelar la operación.	No se elimina un organismo al sistema.	-Se selecciona un Organismo. -Se selecciona la opción "Eliminar". -Se selecciona la opción "Cancelar*".*

SC4: Buscar Organismo	1: Buscar organismo correctamente	Se busca un organismo en el sistema.	- *Se selecciona la opción Buscar.* - *Introduce uno o varios criterios de búsqueda.* *-Selecciona "Buscar".* *-Se muestran los datos del(los) organismo(s) buscado(s).*
	2: Buscar organismo pasando el criterio inexistente.	No se busca un organismo en el sistema.	- Se introduce el criterio -Selecciona "Buscar". - Muestra un mensaje indicando que el criterio de búsqueda introducido no genera resultado. - Seleccionar "Aceptar".
	3: Cancelar la búsqueda.	No se busca un organismo en el sistema.	- Se selecciona la opción Búsqueda avanzada. - Introduce uno o varios criterios de búsqueda. -Selecciona "Cancelar".
	4: Limpiar búsqueda.	Se limpian los datos resultantes de la búsqueda.	-Seleccionar la opción" Limpiar búsqueda" -Vuelve a la ventana principal de Gestionar Organismo mostrando todos los organismos existentes.

Tabla # 19 Caso de prueba del caso de uso: Gestionar Organismos

Caso de prueba del caso de uso: Gestionar Entidad

Nombre de la sección	**Escenarios de la sección**	**Descripción de la funcionalidad**	**Flujo central**
SC1: Adicionar Entidad	*1: Adicionar una entidad correctamente.*	*Se adiciona una nueva entidad al sistema.*	*-Se selecciona la opción "Adicionar".* *- Llenar todos los campos* *- Seleccionar "Aplicar".* *- Seleccionar "Aceptar".*

			- Muestra un mensaje indicando que la entidad fue creada con éxito.
	2: Adicionar una entidad dejando campos en blanco.	*No se adiciona una nueva entidad al sistema.*	*-Se selecciona la opción "Adicionar".* - Llenar los campos. - Seleccionar "Aplicar" *.- Muestra un mensaje de error "Datos incorrectos, por favor revise otra vez"* -Seleccionar "Aceptar"
	3: Adicionar una entidad pasando codigo existente	*No se adiciona una nueva entidad al sistema.*	*- Se selecciona la opción "Adicionar".* - Llenar los campos. - Seleccionar "Aplicar" *.- Muestra un mensaje indicando que el código ya existe.* -Seleccionar "Aceptar"
	4: Cancelar la operación	*No se adiciona una nueva entidad al sistema.*	*-Se selecciona la opción "Adicionar".* - Llenar los campos. - Seleccionar "Cancelar".
SC2: Modificar Entidad	*1: Modificar entidad correctamente*	*Se modifica una entidad al sistema.*	*-Se selecciona una entidad* *-Se selecciona la opción "Modificar"* *- Se cambian los datos deseados.* *- Seleccionar ¨Aplicar.¨* *- Seleccionar ¨Aceptar.¨* *- Muestra un mensaje indicando*

			que la entidad fue modificada con éxito.
	2: Modificar entidad pasando datos incorrectos	*No se modifica la entidad al sistema.*	*-Se selecciona una entidad* *-Se selecciona la opción "Modificar "* *- Se cambian los datos deseados* *-Seleccionar ¨Aceptar.¨* *- Muestra un mensaje indicando cuales campos están vacios o incorrectos.* -Seleccionar "Aceptar"
	3: Cancelar operación	*No se modifica entidad al sistema.*	*-Se selecciona una entidad* *-Se selecciona la opción "Modificar".* *- Se cambian los datos deseados.* *- Seleccionar "Cancelar".*
SC3: Eliminar Entidad	*1: Eliminar entidad correctamente.*	*Se elimina una entidad al sistema.*	*-Se selecciona una entidad.* *-Se selecciona la opción "Eliminar".* *- Muestra un mensaje de confirmación preguntando que si está seguro de eliminar la entidad.* *-Seleccionar la opción "Aceptar"* *-Muestra un mensaje indicando que se eliminó la entidad.*
	2: Cancelar la operación.	*No se elimina una entidad al sistema.*	*-Se selecciona una entidad.* *-Se selecciona la opción "Eliminar".* *- Muestra un mensaje de confirmación preguntando que si está seguro de eliminar la*

			entidad. *-Se selecciona la opción "Cancelar".*
SC4: Buscar Entidad	*1: Buscar entidad correctamente*	*Se busca una entidad en el sistema.*	*-Se selecciona la opción "Búsqueda avanzada"* *- Se introduce(n) el(los) criterio(s) de búsqueda.* *-Selecciona "Buscar".* *-Se muestran los datos de la entidad.*
	2: Buscar entidad pasando criterio inexistente.	*No se muestra entidad buscada.*	*-Se selecciona la opción "Búsqueda avanzada"* *- Se introduce(n) el(los) criterio(s) de búsqueda.* *-Selecciona "Buscar".* *- Muestra un mensaje de error indicando que la búsqueda no arroja resultados.* *- Se selecciona "Aceptar".*
	3: Cancelar la búsqueda.	*No se realiza búsqueda.*	*-Se selecciona la opción "Búsqueda avanzada"* *- Se introduce(n) el(los) criterio(s) de búsqueda.* *-Selecciona "Cancelar"*
	4: Limpiar búsqueda.	*Se limpian los datos resultantes de la búsqueda.*	*-Seleccionar la opción" Limpiar búsqueda"* *-Se regresa a la ventana principal para Gestionar entidades mostrando todas las entidades existentes.*

Tabla # 20: Caso de prueba del caso de uso: Gestionar Organismos

4.8. Conclusiones

En este capítulo se realizan los artefactos principales de los flujos de trabajo Implementación y Prueba. Donde se obtienen los artefactos principales de cada flujo los cuales son: diagrama de despliegue y componentes y modelo de prueba con sus respectivos casos de prueba, los cuales permitieron chequear cada una de las funcionalidades del sistema.

CONCLUSIONES

Esta investigación tiene un papel fundamental, porque a partir de ella se deriva la implementación, prueba e implantación de un piloto del módulo de Administración para el Sistema Unificado de Gestión de Fuerza de Trabajo Calificada “GeForza”, el cual garantiza la homogeneización de los clasificadores utilizados por los diferentes sistemas del departamento Fuerza de Trabajo Calificada (FTC) del Ministerio de Economía y Planificación (MEP).

- ✓ Se realizó un estudio de las herramientas informáticas que posibilitaría la implementación de la solución propuesta.
- ✓ Se logró analizar y diseñar un módulo administración sobre tecnologías libres que garantiza seguridad de la información gestionada en el proceso de administración del sistema GeForza.
- ✓ Logró concebirse, modelarse y especificarse un módulo de administración para el sistema GeForza.
- ✓ Se logró la homogeneización de los clasificadores utilizados por los diferentes módulos del sistema GeForza.
- ✓ Se logró documentar todo el proceso de desarrollo.

Con el presente trabajo se demuestra la solución del problema a resolver relacionado con el módulo de administración para el sistema GeForza.

RECOMENDACIONES

Una vez cumplido con los objetivos de la investigación, teniendo en cuenta las experiencias obtenidas en la misma, se recomienda:

- ✓ Implementar el caso de uso Bitácora el cual permite en vez de eliminar de manera definitiva las tablas de la base de datos poder cambiar su estado a oculto y mediante este caso de uso gestionar estas, pudiendo restaurarlas o eliminarlas.
- ✓ Implementar caso de uso Gestionar Trazas, que cumpla de forma más sencilla con las necesidades del cliente.

BIBLIOGRAFÍA

Rumbaugh, J.; Jacobson, I. y Booch, G.;"El Proceso Unificado de Desarrollo de Software". Addison Wesley. Madrid, 2000, nº

PRESSMAN, R. Ingeniería del software. Un enfoque práctico.2002. McGraw-Hill/Interamericana de España.

PostgreSQL.com. Kit de prensa de PostgreSQL 8.3 Disponible en:
URL: http://www.postgresql.org/about/press/presskit83

Joan Garnet. Doctrine: ORM Open Source para PHP 5.2 +. Disponible en:
URL: http://www.joangarnet.com/blog/?p=415

Ext JS en Español. Comunidad de desarrollo Disponible en:
URL: http://extjs.es/

Tutorial de Apache .Servidor Web Disponible en:
URL: http://www.ayuda-internet.net/tutoriales/desarrollo/apache/index.html

UML: Diagramas. UML ¿Qué es UML? Disponible en:
URL: http://www.ingenierosoftware.com/analisisydiseno/uml.php

Visual Paradigm. Calidad, construir aplicaciones más rápido y mejor. Disponible en:
URL: http://translate.google.com.cu/translate?hl=es&sl=en&u

RUMBAUGH, James, JACOBSON, Ivar; BOOCH, Grady, "El lenguaje unificado de modelado".2000. Addison Wesley

AUTORES, C. D. La definición de software libre. Disponible en:
http://www.gnu.org/philosophy/free-sw.es.html

GARBEE, B. Conferencia Internacional de software libre. 21 de Enero 2006, Disponible en:
http://www.opensourceworldconference.com/malaga06/es/modules/news/article.php?storyid=354

AUTORES, C. D. Conferencia #2 Ingeniería de Software. Fase de Inicio. Modelo del Negocio. 2007_2008, Disponible en:
http://teleformacion.uci.cu/mod/resource/view.php?id=11553

AUTORES, C. D. Conferencia #4 Ingeniería de Software. Flujo de Requerimiento. 2008_2009, Disponible en:
http://teleformacion.uci.cu/mod/resource/view.php?id=12103

AUTORES, C. D. Clase Teórica-Práctica 6. Técnicas de Estimación. Ingeniería de Software 1. 2008_2009, Disponible en:

http://teleformacion.uci.cu/mod/resource/view.php?id=12606

IVAR JACOBSON, G. B., JAMES RUMBAUGH. “El Proceso Unificado de Desarrollo del Software”. Editorial Félix Varela, 2004. Vol. 1.

INFORMÁTICA, C. D. A. F. D. Patrones del “*Gang of Four*”. Unidad Docente de Ingeniería del Software.

LARMAR.C. UML y Patrones. Introducción al análisis y diseño orientado a objetos. Editorial Félix Varela, 2004. Vol. 1.

PEDRO HERNÁNDEZ VALVERDE, N. J. J., ANA MARIA MORENO SÁNCHEZ CAPUCHINO. El Proceso Unificado de Rational (RUP) y su relación con las técnicas y métodos de la ingeniería de usabilidad de software. 2004_2005, Disponible en: http://is.ls.fi.upm.es/doctorado/Trabajos20042005/Hernandez.pdf

PROENZA, Y. E. A. Diseño Avanzado de Aplicaciones Web. EXT – Zend Framework y Doctrine, 2009, Disponible en:

Revista de Software Libre ATIX # 2, http://atix.opentelematics.org

CARRERA, R. “Apuntes de la materia de sistemas integrales de información”.2002.

GICH, J. “Teoría General de sistemas”. México. 1981.

IBIDEM. nº

CHIAVENATO, I. “Introducción a la Teoría General de la Administración”. Colombia 1999.

GLOSARIO DE TÉRMINOS

OACE: Organismos de Administración Central del Estado.

CAP: Consejo de Administración Provincial.

MEP: Ministerio de Economía y Planificación.

FTC: Fuerza de Trabajo Calificada.

GeForza: Sistema Unificado de la Gestión de la Fuerza de Trabajo Calificada.

RF: Requerimiento funcional.

RNF: Requerimiento no funcional.

TIC: Tecnologías de la Información y las Comunicaciones.

DPA: División Político Administrativa.

MVC: Modelo Vista Controlador.

IDE: Entorno Integrado de Desarrollo.

SGBD: Sistema Gestor de Base de Datos.

UML: Lenguaje Unificado de Modelado.

UCI: Universidad de las Ciencias Informáticas

MAC: Mandatory Access Control. Control de Acceso Obligatorio por sus siglas en inglés.

DAC: Discretionary Access Control. Control de Acceso Discrecional por sus siglas en inglés.

RBAC: Role-based Access Control. Control de Acceso Basado en Roles.

BD: Base de Datos.

INDER: Instituto Nacional de Deportes y Recreación.

FAR: Fuerzas Armadas Revolucionarias.

RUP: Proceso Unificado de Desarrollo.

ORM: Mapeador Relacional de Objetos.

POO: Programación Orientada a Objetos.

RUP: Proceso Unificado de Rational.

CUM: Centros Universitarios Municipales.

OC: Obrero Calificado.

TM: Técnico Medio.

NS: Nivel Superior.

PI: Plan de Ingreso.

PD: Plan Distribución.