Universidad de las Ciencias Informáticas
Facultad 3

Título: Sistema para la administración de posgrado en el centro CEIGE

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor(es): Leandro José Capdesuñer García

Tutor(es): Henry Raúl González Brito

Yusmara Buchillón Hernández

La Habana, junio de 2012

Año 53 de la Revolución

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Autor:

_____

Leandro José Capdesuñer García

Tutores:

_____ _____

Ing. Henry Raúl González Brito Ing. Yusmara Buchillón Hernández

AGRADECIMIENTOS

Quiero agradecer:

A mis padres que siempre han estado presentes y me han sabido guiar por el camino correcto los quiero mucho.

A mi novia yubi también te quiero mucho, quien siempre me ha estado apoyando y ayudándome en todo momento.

A mi familia los que están aquí presente y todos los que por diferentes razones no pueden estar.

A mis tutores por la ayuda y los consejos especialmente a yusmara por lo mucho que me ayudo.

Al tribunal y la oponente por los consejos y recomendaciones y los consejos en cada corte.

A mis amigos del aula, del edificio, del barrio todos los que han compartido siempre conmigo.

A todos los presentes muchas gracias.

DEDICATORIA

Este trabajo va dedicado a todas las personas que hicieron posible que se realizara el mismo en especial a mis padres, familiares, novia, tutores, amigos, en fin todos los que de una manera u otra fueron parte del mismo.

RESUMEN

El Centro de Informatización de la Gestión de Entidades (CEIGE) requiere de la gestión de la formación de posgrado de todos sus profesionales, ya que existen deficiencias en la realización de esta tarea, debido a la carencia de una aplicación capaz de realizar la gestión de las actividades necesarias para la actualización de los datos de posgrado de los profesionales del mismo.

El presente trabajo de diploma contempla el desarrollo de un sistema automatizado para gestionar la formación de posgrado del CEIGE a través de diversos aspectos: Se realiza un estudio del estado del arte de varios sistemas de formación de posgrado, además de definir las herramientas, tecnologías y la metodología que guiará el proceso de desarrollo de la aplicación.

Se realizó este trabajo con el objetivo principal de proponer un sistema informático que ayude en la gestión de las actividades de posgrados, se describe el modelado del sistema a través de la especificación de los requisitos, así como la propuesta del diseño, y la construcción del mismo a partir de los resultados del diseño. Se realizaron las pruebas al sistema para examinar el correcto funcionamiento de la misma comprobando la agilización y facilitación del proceso de gestión de las actividades de posgrado, garantizando una mayor calidad.

Palabras Claves: sistema, gestión, posgrado.

TABLA DE CONTENIDOS

ÍNDICE DE IMÁGENES

ÍNDICE DE TABLAS

INTRODUCCIÓN

La sociedad moderna se caracteriza por el cambio acelerado, permanente y continuo. Abrirse paso en la economía global constituye una realidad invariable para los países en vías de desarrollo como Cuba; al no contar el mismo con grandes recursos naturales, recurrir a la formación de un capital humano que investigue, desarrolle e innove pasa a ser un elemento de vital importancia. Disponer de recursos humanos calificados se convierte en la esencia del gran desafío de la capacidad de administrar el conocimiento, ubicándolo en un renglón principal de la economía.

La Universidad de las Ciencias Informáticas (UCI) se integra a la vida económica del país, conservando su esencia de enseñar, educar, dirigir a los futuros ingenieros y convirtiéndose en una de las principales empresas productoras de software del país. Para lograrlo se ha dividido en estructuras productivas especializadas en diversos temas (Salud, Gestión de entidades, entre otras). Como parte de este proceso se cuenta con un amplio claustro de profesores, integrado en gran medida por egresados de la misma institución. En función de esto la Universidad dedica tiempo en la superación de los mismos tanto en el quehacer productivo como en la docencia, ofreciendo cursos de posgrados, entrenamientos, talleres y diplomados, se centra también en su superación académica con maestrías, especialidades y doctorados, así como eventos científicos e investigativos, elevando el prestigio de la universidad.

Actualmente en la universidad existen varias facultades, las cuales tienen definidas estrategias para gestionar la información posgraduada. En la Facultad 3, específicamente en el Centro de Informatización de la Gestión de Entidades (CEIGE) existen deficiencias en el control de esta información, provocando que el proceso de posgrado se realice con retrasos, deficiencias en la toma de decisiones y en ocasiones pérdida de información, algunos factores así lo indican:

1. No existe un control adecuado de los cursos que reciben los profesionales del centro, esto provoca que no se conozca que profesionales han pasado determinados cursos y cuáles no, impidiendo que se puedan tomar decisiones respecto a las matrículas de los cursos que se ofertan.
2. No existe un control adecuado de los programas académicos en los cuales están matriculados los estudiantes de posgrado, esto afecta la planificación de los departamentos porque jefes de proyectos y departamentos no manejan la misma información sobre este aspecto.

3. No existe un control adecuado del estado de completamiento de los créditos de los programas académicos de posgrado, lo que provoca que no se conozca el estado que presenta en las maestrías y diplomados, dificultando la solicitud de plazas en cursos ofertados o planificados.
4. Se desconoce el cumplimiento real de los cursos obligatorios que tiene que pasar los adiestrados, lo que afecta la planificación y toma de decisión respecto al proceso de superación posgraduada de los adiestrados.
5. No existe un control adecuado de los cursos que el centro ha impartido lo que dificulta consolidar la información que se utiliza para reportar los resultados de posgrado anuales del centro.
6. Existen dificultades en el control del proceso de la categorización docente, por esto motivo se hace difícil el seguimiento y control del proceso de categorización ya que no se conoce el estado de los profesores.
7. No se tiene un control de la superación de los profesionales

Teniendo en cuenta lo anteriormente planteado se identifica el siguiente **Problema a resolver:**
¿Cómo contribuir a mejorar el control de los procesos de posgrado de los profesionales del CEIGE?
Por tanto el **objeto de estudio** que guiará la investigación se centrará en Los procesos de desarrollo de software aplicado a sistemas de gestión de posgrado, y se precisa en el **campo de acción:** La Informatización y control del proceso de superación posgraduada del centro CEIGE.
Para dar cumplimiento al presente trabajo de diploma se plantea como **objetivo general:** Desarrollar un sistema web para el control de las principales actividades del proceso de posgrado en el centro CEIGE
Del objetivo general, se desglosaron los siguientes **objetivos específicos:**

1. Definir el marco teórico de la investigación.
2. Realizar el análisis y diseño del proceso de posgrado en el centro CEIGE.
3. Implementar el Sistema para la administración de posgrado en el centro CEIGE.
4. Validar los resultados obtenidos.

Dando cumplimiento a estos objetivos se espera materializar **la idea a defender** siguiente: La implementación de un sistema de control del proceso de posgrado del centro CEIGE permitirá gestionar la información referente a los estudiantes de posgrado de forma rápida y segura, permitiendo un control adecuado de los cursos, los programas académicos en los cuales están matriculados, estado de completamiento de los créditos y proceso de la categorización docente.

Métodos de investigación científica

Métodos teóricos

Histórico-Lógico: En la investigación se utiliza este método ya que se realiza un análisis de la evolución que ha tenido en Cuba y en el mundo el tema de la formación posgraduada. Además luego de realizar las consultas en diferentes literaturas acerca de los procesos de formación personal, se pueden hacer comparaciones en aras de seleccionar de cada criterio estudiado, los aspectos que más aportan al tema que es objeto de la investigación.

Métodos empíricos

Observación: Este método se utilizará para realizar una evaluación de la situación problemática en cuestión.

Entrevista: Permitió realizar conversaciones planificadas entre el cliente y el analista. Fue utilizado para obtener información que serviría para la especificación de los requisitos del sistema.

Estructura del Trabajo

Capítulo 1: Fundamentación teórica: En este capítulo se realiza un estudio del estado del arte revisando algunos de los sistemas más utilizados para la gestión de los procesos de posgrado exponiendo las ventajas y las desventajas de cada uno. Se realizará un análisis de las metodologías, herramientas y lenguajes que existen en la actualidad y que pudieran ser útiles en el desarrollo de la solución justificando la selección de los mismos.

Capítulo 2: Análisis y diseño: Se realizará el análisis y diseño donde se obtendrán los artefactos necesarios para la posterior implementación del proceso entre los que se puede mencionar la identificación, especificación y validación de los requisitos y la especificación de los diagramas de clases del diseño, así como las métricas utilizadas para su validación. En este capítulo además se determinará la arquitectura y los patrones de diseño a utilizar en la implementación.

Capítulo 3: Implementación y pruebas: Se llevará a cabo la implementación de la aplicación, obteniéndose los artefactos correspondientes como son los diagramas de componentes y el de despliegue, además posteriormente se le realizarán las pruebas pertinentes. También se realizará un análisis de los beneficios de la aplicación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo primeramente se hará mención a algunos conceptos básicos para una mejor comprensión del trabajo. Se realizará un estudio crítico y valorativo de los sistemas utilizados para la gestión de posgrado a nivel mundial y en Cuba. Con el objetivo de obtener ventajas y desventajas para la realización del Sistema para la administración de posgrado en el centro CEIGE. Se incluirá además, un estudio de las tendencias, tecnologías, métodos y herramientas a utilizar para su desarrollo.

1.2 Conceptos Básicos

Formación posgraduada

Se define como formación posgraduada, toda la actividad referida a estudios realizados luego del título de grado, es decir que forma parte de estudios superiores o de tercer ciclo y comprende especialización, maestría, doctorado y posdoctorado teniendo como antecedente obligatorio el pregrado. En Cuba se le confiere una importancia enorme a la formación posgraduada, con el fin de tener cada día profesionales más capaces de afrontar los problemas del mundo contemporáneo, por este motivo se aboga por la construcción de un humano más integral y comprometido con la sociedad. (Pérez, 2001)

La formación posgraduada se divide en varios aspectos entre los cuales se definen como fundamentales:

Superación profesional

Curso: provee formación sobre todo, a los graduados universitarios, comprende contenido sobre temas específicos, que posibilitan la instrucción, actualización o especialización de los profesionales que lo reciben. Tiene una extensión mínima de 2 créditos.

Entrenamiento: forma a los profesionales sobre todo en un enfoque de desarrollo o especialización de habilidades o destrezas, en el uso de nuevas tendencias o tecnologías de nuevo surgimiento. Tiene una extensión mínima de 2 créditos.

Diplomado: tiene como objetivo la formación de los profesionales en un marco de área particular de desempeño, se concibe como un conjunto de cursos y/o entrenamientos, el mismo culmina con la realización y defensa de un trabajo final ante un tribunal. Tiene una extensión mínima de 15 créditos.

Superación académica

Maestría: proporciona una vasta cantidad de conocimientos y cultura científica, una mayor capacidad de desarrollo docente, científico y de innovación. Se compone de una tesis o trabajo investigativo final a discutir ante un tribunal y tiene como extensión mínima 70 créditos.

Especialidad: enfoca la preparación de los profesionales, fundamentalmente en el adiestramiento para la competencia laboral, que exige el desempeño en un determinado puesto. Tiene como extensión mínima 100 créditos.

Doctorado: se otorga a profesionales que desarrollan un trabajo de alto nivel científico, los cuales logran una extensa especialización en una rama determinada de las ciencias. Se defiende un trabajo final con un gran aporte científico e innovador ante un tribunal competente, el cual contiene la solución a un problema científico de gran envergadura. (Calleja, 2003)

Posgrado

Podría decirse que el posgrado es la última fase de la educación formal e incluye los estudios de especialización, maestría y doctorado. Las características de los posgrados dependen de cada país o institución.

Son los estudios de especialización luego de haber pasado la licenciatura o pregrado, forma parte del tercer ciclo de estudio; la última fase de la educación formal. Es el nivel más alto del sistema de educación superior, dirigido a promover la educación permanente de los graduados universitarios. En la educación de posgrado concurren uno o más procesos formativos y de desarrollo, no solo de enseñanza aprendizaje, sino también de investigación, innovación, creación artística y otros, articulados armónicamente en una propuesta docente-educativa pertinente a este nivel. (MES, 2006)

Programas

Es la previa declaración de lo que se piensa hacer en alguna materia; el sistema y distribución de las materias de un curso o asignatura

Proyecto o planificación ordenada de las distintas partes o actividades que componen una cosa que se va a realizar sistema y distribución de las materias que forman una asignatura o un curso escolar

1.3 Algunos sistemas usados en la gestión de posgrado en el mundo

1.3.1 Sistema de Posgrados de la ESPOL

EL Sistema de Posgrados de la ESPOL, ofrece maestrías, especializaciones y diplomados en una amplia variedad de tópicos, con personal docente de alto nivel, tanto nacionales como extranjeros.

Los estudiante de posgrado, pueden utilizar el sistema de información para consultar sus calificaciones y horarios de clases, exámenes, profesores, próximos cursos a iniciarse, etc. (ESP del Litoral Guayaquil, 2007)

- Permite al alumno inscribirse en los diferentes posgrados que están en oferta.
- Brinda un calendario a los estudiantes sobre el proceso de matrícula, los costos y la modalidad.
- Los estudiantes tienen la posibilidad de informarse sobre los requisitos de admisión, las titulaciones que se otorgan, promociones disponibles y la malla curricular.
- Los posgrados están organizados por las diferentes áreas a las que pertenecen para lograr la organización de los mismos.

1.3.2 Postgraduate Students Management System (PSMS)

El Sistema de Gestión de Estudiantes de Posgrado (PSMS) facilita a los estudiantes de posgrado el registro en los cursos del programa de maestría. Los estudiantes que aparecen en el segundo semestre en adelante pueden tomar ventaja de este servicio de registro en línea. Además los estudiantes de posgrado pueden generar su formulario de inscripción semestral, no es necesario realizar más formularios de registros de forma manual. Los estudiantes de posgrado también pueden ver la información completa de su historial personal y académico en línea. (University of Engineering & Technology, 2012)El mismo permite:

- Se permite el revisado del horario de registro de primavera 2012
- Los resultados de programas de posgrado
- Ver el calendario académico
- A raíz de los programas de posgrado se ofrecen publicar folleto de posgrado

1.3.3 Sistema de Información de Estudios de Posgrado. Universidad Central de Venezuela

El Sistema de Información de los Estudios de Posgrado en la Universidad Central de Venezuela (Sidep-UCV) pone en servicio sus plataformas de documentación, información e interacción y constituye un salto cualitativo en el manejo de los Sistemas de Información del Posgrado.

Sidep-UCV brinda acceso a la oferta académica de cursos, ofrece directorios sobre profesores/expertos y sobre cursantes/egresados, brinda acceso a sus centros de documentación y a sus bases de datos. Propicia la interrelación entre los egresados y la universidad, entre los miembros de la colectividad Universidad Central de Venezuela y las comunidades académicas del mundo.

El sistema brinda las siguientes ofertas:

- Sistema de Información sobre los Cursos de Posgrado en la UCV.
- Directorio de "Oportunidades de Financiamiento de Estudios de Posgrado, existentes en la UCV y Entes Nacionales".
- Normativa Nacional e Institucional del Posgrado.
- Lista de información sobre el posgrado (inscripciones, oferta de cursos) y también puede participar en una lista de reflexión sobre posgrado y desarrollo.
- Noticias y eventos.
- Sistema de Información Bibliográfico con la Producción Intelectual del Posgrado: Libros, Tesis, Monografías, Artículos y con bibliografía nacional e internacional sobre la investigación en educación avanzada 1998/1997/1941-1996.
- Centro de Información y Documentación de Estudios de Posgrado (Cipost).
- Publicaciones electrónicas (texto completo) basadas en los resultados de investigación del Centro de Estudios e Investigaciones sobre Educación Avanzada (Ceisea) Universidad Central de Venezuela.
- Ponencias (texto completo) presentadas en las Jornadas de Análisis de Posgrado de la UCV. (1998).
- Convenios y Programas de la UCV.
- Otros sitios de interés.

Tiene una herramienta de búsqueda mediante la cual usted podrá acceder al Catálogo de cursos por Facultad, por Área del conocimiento y Palabra clave.

- Búsqueda por Facultad:

Para realizar su búsqueda en la base de datos de cursos de posgrado, hará un enlace a la lista de cursos por Facultad, allí seleccionará el curso y obtendrá seguidamente todos los datos que puedan ser de su interés con relación a éste.

- Búsqueda por Área de Conocimiento

Para realizar la búsqueda en el Catálogo de Cursos de Posgrado, seleccione el área del conocimiento, allí se obtiene una respuesta a su búsqueda que se traduce en todos los cursos que puedan ser descritos dentro del área, en esa lista se selecciona el curso y se obtiene seguidamente todos los datos que puedan ser de interés con relación a éste.

- Búsqueda por Palabra Clave

Para realizar la consulta se escribe un término que describa su parámetro de búsqueda. Como resultado de la consulta se encuentra un listado de los cursos de posgrado, en esa lista se selecciona el Curso de interés y se obtiene todos los datos relacionados con éste.

1.3.4 El Sistema de Estudios de Posgrado (SEP)

El Sistema de Estudios de Posgrado (SEP) se dedicada a impulsar y facilitar el desarrollo de los programas de maestría, doctorado y especialidades. Es un sistema encargado de organizar, orientar, impulsar y administrar los programas de posgrado de la Universidad de Costa Rica. El desarrollo de los programas se orienta hacia la especialización de profesionales en diferentes disciplinas del saber, con un enfoque transdisciplinario y orientación transcultural. Se apoya, además, en la investigación para contribuir a generar conocimientos y a reformular conceptos. (Universidad Central de Venezuela, 2010)

Brinda una variada oferta de posgrado:

- Propone diferentes cursos, maestrías, doctorados y especialidades para una mejor preparación de los profesionales.
- Permite al alumno inscribirse en los diferentes posgrados que están en oferta.
- Presenta un detallado programa acerca de las convocatorias de admisión, incluyendo la modalidad del posgrado.
- Brinda un calendario a los estudiantes sobre el proceso de matrícula y los costos por etapas así como todo lo referente a la documentación necesaria para dicho proceso.
- Los estudiantes tienen la posibilidad de informarse acerca de los horarios relacionados con los diferentes posgrados de su interés.
- Brinda información referida a los costos y pagos para el correcto ingreso a los posgrados.

1.4 Algunos sistemas usados en la gestión de posgrado en Cuba

1.4.1 Sistema de Gestión Académica de Posgrado

El proyecto "Gestión de Posgrado" desarrolla el producto nombrado "Sistema de Gestión Académica de Posgrado", el cual gestiona los procesos de formación posgraduada tanto de profesionales de la Universidad de las Ciencias Informáticas como fuera de ella. Los procesos involucrados abarcan la gestión de las actividades de formación académica y de superación profesional las que, a su vez, incluyen las formas organizativas: curso, entrenamiento, diplomado, maestría, doctorado y especialidad. La solución comprende los procesos de gestión de actividades tales como: inscripción, convocatoria,

matrícula y control docente de cada una de las formas organizativas así como reportes, estadísticas y gráficos. De igual manera incluye la gestión documental asociada a la actividad de posgrado y su integración con el Archivo Universitario.

El sistema garantizará dichas formas organizativas con el mínimo posible de trabajo manual y por ende automatizará y agilizará procesos que hoy día sólo pueden ser ejecutados por una sola persona en un mismo espacio de tiempo y lugar. Los usuarios del sistema tendrán acceso al mismo de forma simultánea desde cualquier punto del centro u institución donde se implante dicho sistema

1.4.2 Sistema para la Gestión de la Formación de Posgrado en el Centro de Informatización de la Gestión de Entidades

En el 2010 se realizó la tesis Sistema para la Gestión de la Formación de Posgrado en el Centro de Informatización de la Gestión de Entidades implementada con el framework Symfony con el objetivo de implementar un nuevo sistema que le proporcione la posibilidad al profesional de mantener su información de posgrado y perfil profesional actualizado, que facilite el trabajo al responsable de gestionar esta información, además de poder informarse acerca de los diversos temas relacionados a la formación de posgrado. La solución comprende los procesos actualizar del perfil, gestionar ediciones de actividad de posgrado y realizar reportes.

1.5 Valoración de los sistemas de posgrado estudiados

Los sistemas de posgrado anteriormente estudiados constituyen herramientas significativas en sus respectivas áreas, pero tienen algunas limitaciones, como es el caso del Sistema de Estudios de Postgrado (SEP) el cual realiza el proceso de inscripción a través del correo, vía a veces no segura, no tiene un control de los estudiantes matriculados en los diferentes cursos de posgrados ofertados, no ofrece facilidades de la gestión de reportes relacionada con datos de los estudiantes, es decir la cantidad de cursos vencidos, los créditos acumulados, etc. Por otra parte, el Sistema SPOL pone a disposición de todos los interesados, un variado servicio que garantiza en gran medida las necesidades de los mismos, pero hay que efectuar un pago para la matrícula a los cursos disponibles. En algunos casos los sistemas requieren del pago de costosas licencias para su utilización, no se gestiona la administración de los programas de posgrados y las ediciones. En la UCI se han desarrollado dos sistemas para la gestión de los procesos de posgrado, el Sistema de Gestión Académica de Postgrado que actualmente se utiliza en la secretaría de posgrado de la universidad y el Sistema para la Gestión de la Formación de Posgrado en

el Centro de Informatización de la Gestión de Entidades, algunas de las deficiencias que estos sistemas presentan son: que no realizan el proceso de cambio de categoría docente de los profesionales, no muestran el historial de las ediciones de cursos asociadas a los profesionales y el estado de las mismas, no existe un control de la cantidad de ediciones concluidas, activas y abandonadas de cada profesional.

Luego de hacer un análisis de los sistemas estudiados se llega a la conclusión, que los mismos no cumplen con algunas funcionalidades propuestas por el cliente, el estudio de las funcionalidades y tecnologías de los sistemas existentes llevaría consigo mayor cantidad de tiempo que la creación de uno nuevo, además, ninguno de estos sistemas se encuentran implementados sobre el marco de trabajo Sauxe, uno de los requisitos propuestos por el cliente, por lo que es más factible hacer un sistema nuevo, que responda específicamente a las necesidades del proceso de posgrado en el CEIGE, que adicionar nuevas funcionalidades a los ya existente.

1.6 Proceso de Desarrollo de Software

1.6.1 Modelo de desarrollo

Para guiar el desarrollo del Sistema para la administración de posgrado en el centro CEIGE se empleará el **modelo de desarrollo orientado en componentes** creado por un equipo de desarrollo del CEIGE, basado en buenas prácticas y principios de varias metodologías, ya sean ágiles como XP[1] y SCRUM o pesada como RUP[2]. Está orientado a las necesidades y artefactos que son generados durante el desarrollo de cualquier software; define todos los roles involucrados y sus responsabilidades, las diferentes actividades que realizan, junto con el flujo de estas y los artefactos que se deben generar. (Equipo de Producción, 2009)

La utilización de este modelo se debe a que es el establecido por el centro para el desarrollo de todos sus productos además de las características que presenta:

- **Centrado en la arquitectura**

A través de la arquitectura se orientan las prioridades del desarrollo, se resuelven las necesidades tecnológicas y de soporte, determina la línea base, los elementos de software estructurales a partir de los existentes en la arquitectura de negocio, interviene en la gestión de cambios y diseña la evolución e integración del producto.

1 Programación Extrema del inglés Extreme Programming.
2 Proceso Unificado de Rational del inglés Rational Unified Process.

- **Orientado a componentes**

Según el nivel de significado arquitectónico de los componentes, son orientadas las iteraciones.

- **Iterativo e incremental**

El equipo de arquitectura, los clientes y la alta gerencia, planifican y coordinan las iteraciones, estas constituyen el desarrollo de componentes, los cuales son integrados al término de la iteración, permitiendo la evolución incremental del producto.

- **Ágil y adaptable al cambio**

Los clientes y funcionales son involucrados en el proyecto, por lo que poseen parte de la responsabilidad del éxito del mismo. Semanalmente se concilian, discuten y aprueban los cambios. El desarrollo de las partes formaliza solo las características principales de la solución, priorizándose de esta manera los talleres y las comunicaciones.

- Son utilizados solamente los artefactos necesarios para documentar el producto.
- Se modela el negocio por procesos y no por casos de uso.

1.7 Herramientas y tecnologías utilizadas

1.7.1 Herramientas CASE

Las herramientas CASE (*Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador*) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras. Sistema de software que intenta proporcionar ayuda automatizada a las actividades del proceso de software. Los sistemas CASE a menudo se utilizan como apoyo al método. (Scribd, 2010)

Visual Paradigm for UML 6.1

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Visual Paradigm es una herramienta CASE creada para soportar el ciclo de vida

completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas entre otras opciones. Constituye una herramienta de software libre de gran utilidad para el analista. Dentro de sus características se aprecia que soporta BPMN y UML versión 2.1. Muestra también: (Visual Paradigm, 2008)

- Diagramas de Procesos de Negocio.
- Modelado colaborativo con CVS y subversión.
- Generador de informes para generación de documentación.
- Distribución automática de diagramas - Reorganización de las figuras y conectores de los diagramas UML.
- Dibujo de diagramas UML con plantillas (stencils) de MS Visio.
- Editor de figuras.

El Visual Paradigm for UML es un producto que facilita la organización, la visualización, diseño, integración y despliegue mediante diagramas. La herramienta ayuda al equipo de desarrollo de software a mejorar la construcción del modelo del proceso de desarrollo de software, maximizando y acelerando la producción del equipo y las contribuciones individuales.

1.7.2 Lenguaje Unificado de Modelado (UML)

UML (Unified Modeling Lenguaje) o Lenguaje Unificado de Modelado es un lenguaje basado en una notación gráfica la cual permite: especificar, construir, visualizar y documentar los objetos de un sistema programado. UML se quiere convertir en un lenguaje estándar con el que sea posible modelar todos los componentes del proceso de desarrollo de aplicaciones, ha sido ampliamente aceptado debido al prestigio de sus creadores. Hay que tener en cuenta que el estándar UML no es un proceso, no es una metodología de desarrollo, sino una notación, un lenguaje. (Romero, 2010)

Será utilizado por las siguientes ventajas:

- Es una consolidación de muchas notaciones y conceptos más usados orientados a objetos.
- Utiliza un conjunto de símbolos para representar gráficamente los diversos componentes en relación con el sistema.

- Puede ser utilizado para el modelado de negocio de procesamiento y modelado de requisitos.
- Es independiente del lenguaje de programación.
- Realiza la documentación de todos los artefactos que son generados durante el proceso de desarrollo.
- Se aprende con facilidad.
- Es un lenguaje consolidado.
- Tecnología orientada a objetos.
- El cliente participa en todas las etapas del proyecto.
- Corrección de errores viables en todas las etapas.

1.7.3 Base de datos

Una base de datos es una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular.

PostgreSQL 8.3

Es un sistema de gestión de bases de datos, objeto-relacional (ORDBMS), basado en el proyecto Postgres de la Universidad de Berkeley. Está considerado como el sistema de base de datos de código abierto más avanzado del mundo, publicado bajo la licencia BSD[3]. (Toledo, 2010)

A continuación se enumeran las principales ventajas de este gestor de bases de datos por lo cual será utilizado: Implementación del estándar SQL92/SQL99.

- Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP), cadenas de bits, etc. También permite la creación de tipos propios.
- Incorpora una estructura de datos de arreglos.

[3]**BSD:** es la licencia de software otorgada principalmente para los sistemas BSD ("Berkeley Software Distribution, en español, distribución de software berkeley"). Es una licencia de software libre. Esta licencia tiene menos restricciones en comparación con otras estando muy cercana al dominio público, permite el uso del código fuente en software no libre.

- Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
- Permite la declaración de funciones propias, así como la definición de disparadores.
- Está distribuido bajo licencia BSD.
- Su código fuente está disponible libremente.
- Utiliza un modelo cliente/servidor.
- Funciona correctamente con grandes cantidades de datos y una alta concurrencia de usuarios accediendo al mismo tiempo.
- Posee acceso encriptado vía SSL.

PgAdmin III

PgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia de código abierto. Está escrita en C++ usando la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda usar en Linux, FreeBSD, Solaris, Mac OS X y Windows. Es capaz de gestionar versiones a partir de PostgreSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de PostgreSQL entre ellas: Pervasive Postgres, EnterpriseDB, Mammoth Replicator y SRA PowerGres. Incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados, soporte para el motor de replicación Slony-I y mucho más. La conexión al servidor puede hacerse mediante conexión TCP/IP[4] o Unix Domain Sockets (en plataformas *nix), y puede encriptarse mediante SSL[5] para mayor seguridad. (Guía Ubuntu, 2008).

Esta herramienta será utilisada para la gestión de la base de datos del Sistema de administración de posgado del CEIGE.

[4] **TCP/IP:** (Transmission Control Protocol/Internet Protocol) Protocolo de control de transmisión/Protocolo de Internet.

[5] **SSL:** (Secure Sockets Layer) Protocolo de Capa de Conexión Segura.

1.7.4 IDE de desarrollo

Un entorno de desarrollo integrado o IDE (acrónimo en inglés de integrated development environment), es un programa informático compuesto por un conjunto de herramientas de programación o puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios. (IDE, 2011)

NetBeans 7.0.1

Es un entorno de desarrollo integrado (IDE), modular, de base estándar (normalizado), escrito en el lenguaje de programación Java. El proyecto NetBeans consiste en un IDE de código abierto y una plataforma de aplicación, las cuales pueden ser usadas como una estructura de soporte general (framework) para compilar cualquier tipo de aplicación. Provee varias características y mejoras nuevas, tales como mejores características de edición JavaScript, soporte para usar estructuras Spring de soporte web. (Netbeans, 2011)

sera utilizado como IDE de desarrollo por las ventajas que ofrece:

- Altamente configurable.
- Es multiplataforma.
- Soporte JavaScript
  - Sintaxis Resaltada
  - Completamiento de Código y Análisis de Tipeo
  - Soluciones Rápidas (Quick Fixes) y Verificación de Sintaxis
  - Refactorización

1.8 Tecnologías Web

Las tecnologías web son un conjunto de herramientas que facilitan lograr mejores resultados a la hora del desarrollo de un sitio web.

1.8.1 Lenguajes, notaciones y tecnologías de programación

Lenguaje de Marcas de Hipertexto (HTML)

El Lenguaje de Marcas de Hipertexto (Hyper Text Markup Language), no es más que un conjunto de etiquetas o comandos, complementados en la mayoría de los casos por extensiones que permiten dar formato a un archivo, con el objetivo básico de crear un documento que pueda ser visualizado en

ambiente Internet en forma de Página Web y que esta, además, pueda, por medio de dichas etiquetas, tener la estructura o forma deseada por quien la diseñó. Las etiquetas HTML tienen, por lo general, una etiqueta de apertura y una de cierre; aunque existen algunas excepciones en las que solo basta con colocar la de apertura. (Grupo de autores, 2010)

JavaScript

JavaScript es un lenguaje interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C. Al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de herencia, es más bien un lenguaje basado en prototipos, ya que las nuevas clases se generan clonando las clases base y extendiendo su funcionalidad. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del DOM5. JavaScript es un lenguaje con muchas posibilidades, utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes, orientados a objetos mucho más complejos. Con JavaScript se puede crear diferentes efectos e interactuar con los usuarios. JavaScript es soportado por la mayoría de los navegadores como Internet Explorer, Netscape y Mozilla Firefox. (Valade, 2004)

JavaScript y XML asíncronos (Ajax)

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript y XML asíncronos), es una técnica de desarrollo web para crear aplicaciones interactivas. Estas se ejecutan en el cliente, es decir, en el navegador del usuario, y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma, es posible realizar cambios sobre la misma página sin necesidad de recargarla, aumentando la interactividad, velocidad y usabilidad en la misma.

AJAX es una combinación de tres tecnologías ya existentes:

- XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.
- DOM accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y Script, para mostrar e interactuar dinámicamente con la información presentada.

- El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto iframe en lugar del XMLHttpRequest para realizar dichos intercambios.

AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente. (ProgramacionWeb, 2009)

Lenguaje de Etiquetado Extensible (XML)

El Lenguaje de Etiquetado Extensible (Extensible Markup Language, XML por sus siglas en inglés) es muy simple, pero estricto, pues juega un papel fundamental en el intercambio de una gran variedad de datos. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones. Dicha tecnología es un conjunto de módulos que ofrece servicios útiles a las demandas más frecuentes por parte de los usuarios. XML sirve para estructurar, almacenar e intercambiar información. Su función principal es describir datos y no mostrarlos como es el caso de HTML. (Colectivo editorial, 2008)

Hipertext Preprocesor (PHP 5.2.3)

PHP es el acrónimo de Hipertext Preprocesor. Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación. Se escribe dentro del código HTML, lo que lo hace realmente fácil de utilizar. Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor web.
Esto hace que cualquier sistema pueda ser compatible con el lenguaje y significa una ventaja importante.

Al ser un lenguaje libre dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas: Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de Acrobat Reader), analizar código XML. Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes. (The PHP Group, 2011)

Su empleo para el desarrollo del Sistema para la administración de posgrado en el centro CEIGE se debe a que el marco de trabajo Sauxe, sobre el cual se implementará el sistema, fue realizado con este lenguaje de programación, debido a sus ventajas:

- Posee gran cantidad de documentación y librerías.
- Provee diferentes niveles de seguridad, que pueden ser configurados en el archivo .ini.

- Es de código abierto por lo cual no hay que pagar para la obtención de actualizaciones anuales, ni por el uso de este.
- La interacción dinámica entre la página web y el usuario.
- La creación de aplicaciones para servidores e independientes del navegador.
- Utilizando el mismo código fuente, funciona sobre múltiples plataformas.
- Se pueden leer y escribir archivos, crear imágenes, conectarse a servidores remotos, realizar consultas a base de datos.

1.8.2 Tipo de arquitectura

Una Arquitectura de Software, también denominada Arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información. (Machado, 2010)

Estilo arquitectónico Modelo-Vista-Controlador (MVC).

MVC es un estilo arquitectónico de diseño de arquitectura de software usado principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de los conceptos para que el desarrollo este estructurado de una mejor manera, facilitando la programación en diferentes capas de manera paralela e independiente. MVC sugiere la separación del software en 3 estratos.

Modelo: Es la representación de la información que maneja la aplicación. El modelo en sí son los datos puros que puestos en un contexto del sistema proveen de información al usuario o a la aplicación misma.

Vista: Es la representación del modelo en forma gráfica disponible para la interacción con el usuario. En el caso de una aplicación web la "Vista" es la página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones.

Controlador: Es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando el Modelo en caso de ser necesario.

El **Modelo** es el responsable de:

- Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.

- Definir las reglas de negocio (la funcionalidad del sistema).
- Llevar un registro de las vistas y controladores del sistema.
- Si se encuentra ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

El **Controlador** es responsable de:

- Recibir los eventos de entrada.
- Contiene reglas de gestión de eventos, del tipo "SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas.

Las **Vistas** son responsables de:

- Recibir datos del modelo y mostrarlos al usuario.
- Tienen un registro de su controlador asociado (normalmente porque además lo instancia).
- Pueden dar el servicio de "Actualización", para que sea invocado por el controlador o por el modelo (cuando es un modelo activo que informa de los cambios en los datos producidos por otros agentes).

En la siguiente figura se muestra la estructura del patrón arquitectónico Modelo-Vista-Controlador.

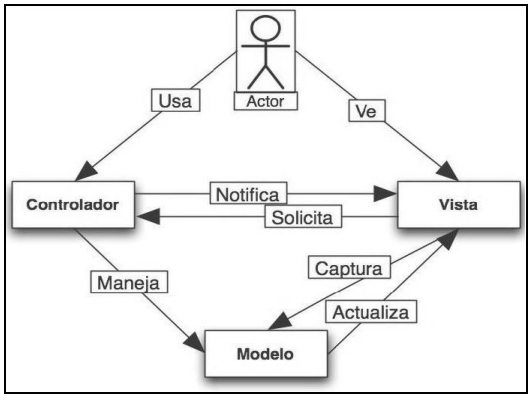

Figura 1: Estructura Modelo-Vista-Controlador

Tanto la vista como el controlador dependen del modelo, el cual no depende de las otras clases. Esta separación permite construir y probar el modelo, independientemente de la representación visual.

Ventajas:

Soporte de múltiples vistas: Dado que la vista se halla separada del modelo y no hay dependencia directa del modelo con respecto a la vista, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos simultáneamente. Por ejemplo, múltiples páginas de una aplicación web pueden utilizar el mismo modelo de objetos mostrado de maneras diferentes.

Adaptación al cambio: Los requisitos de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas de negocios. Los usuarios pueden preferir distintas opciones de representación, o requerir soporte para nuevos dispositivos como teléfonos celulares o PDAs[6]. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación generalmente no afecta al modelo.

Desventajas:

Costo de actualizaciones frecuentes: Si el modelo experimenta cambios frecuentes, por ejemplo, podrían desbordar las vistas con una lluvia de requisitos de actualización.

1.8.3 Servidor Web

Apache 2.0

Apache es el servidor web hecho por excelencia, su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. Este servidor de aplicaciones web ofrece una serie de ventajas: (Ciberaula, 2010)

- Corre en una multitud de sistemas operativos, lo que lo hace prácticamente universal.
- Es una tecnología gratuita de código fuente abierta. Esto le da una transparencia al software de manera que si se desea ver lo que está instalando como servidor, se puede saber, sin ningún secreto o puertas traseras.
- Es un servidor altamente configurable de diseño modular.
- 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.
- La configuración de mensajes de error.

[6] **PDA**:(Personal Digital Assistant) Asistente Digital Personal.

1.8.4 Frameworks

Un framework es una estructura conceptual y tecnológica de soporte definida, normalmente, con artefactos o módulos de software concretos, en base a la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas 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. (Patxi, 2007)

Zend Framework

Es un framework para desarrollo de aplicaciones Web y servicios Web con PHP. Brinda soluciones para construir sitios Web modernos, robustos y seguros. Además, es código abierto y trabaja con PHP 5. (Patxi, 2007)

Presenta entre otras las siguientes características:

- Simplifica la gestión de archivos de configuración.
- Proporciona los componentes que forma la infraestructura del patrón Modelo-Vista-Controlador.
- Proporciona una capa de acceso a base de datos, construida sobre PDO[7] pero ampliándola con diferentes características.
- Cuenta con módulos para manejar archivos PDF, canales RSS, Web Services (Amazon, Flickr, Yahoo)
- Proporciona mecanismos de filtrado y validación de entradas de datos.
- Clientes para servicios Web, incluidos Google Data APIs y StrikeIron.

Doctrine Framework

Doctrine es un ORM[8] para PHP 5.2.3 y posterior. Además de todas las ventajas que conlleva un ORM, uno de sus puntos fuertes es su lenguaje DQL (Doctrine Query Language) inspirado en el HQL[9] de Hibernate.

[7] **PDO:** (PHP Data Objects) Capa de abstracción de acceso a datos para PHP.
[8] **ORM:** Object Relation Mapper.
[9] **HQL:** (Hibernate Query Language).

Cuando se trabaja con Doctrine, es necesario informar a su motor interno del cual es el modelo de la aplicación, para ello es posible hacer ingeniería inversa de la base de datos existente, o si comienza la aplicación desde 0, crear el modelo en la sintaxis específica que propone Doctrine y luego generar toda la base de datos. (Doctrine, 2010)

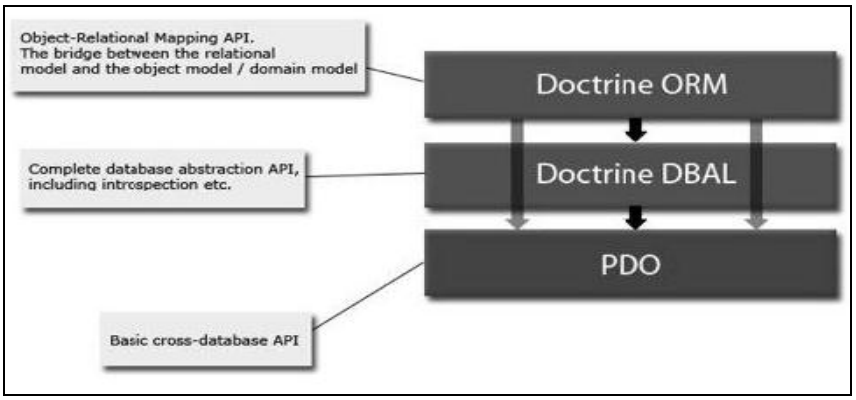

Figura 2: Estructura de Doctrine ORM.

Un ORM o (Object Relational Mapper) es una técnica de programación que permite convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional, es decir, las tablas de la base de datos pasan a ser clases y los registros objetos que se pueden manejar con facilidad. (Doctrine, 2010)

Utilizar un ORM tiene una serie de ventajas que facilita enormemente tareas comunes y de mantenimiento:

- **Reutilización:** La principal ventaja que aporta un ORM es la reutilización permitiendo llamar a los métodos de un objeto de datos desde distintas partes de la aplicación e incluso desde diferentes aplicaciones.
- **Encapsulación:** La capa ORM encapsula la lógica de los datos pudiendo hacer cambios que afectan a toda la aplicación únicamente modificando una función.
- **Portabilidad:** Utilizar una capa de abstracción permite cambiar en mitad de un proyecto de una base de datos MySQL a una Oracle sin ningún tipo de complicación. Esto es debido a que no utiliza una sintaxis MySQL, Oracle o SQLite para acceder al modelo, sino una sintaxis propia del ORM utilizado que es capaz de traducir a diferentes tipos de bases de datos.

- **Seguridad:** Los ORM suelen implementar mecanismos de seguridad que protegen la aplicación de los ataques más comunes como inyecciones de código SQL.
- **Mantenimiento del código:** Gracias a la correcta ordenación de la capa de datos, modificar y mantener los códigos es una tarea sencilla.

ExtJs Framework 3.3

Ext JS es un framework que soporta JavaScript para construir aplicaciones complejas en internet, además de ser la base para Ext. Designer, el cual es una aplicación de escritorio que te permite construir aplicaciones web (Inc, 2011). En la implementación del Sistema de administración de posgrado en el centro CEIGE será empleado para la construcción de todas las interfaces de la aplicación debido a las ventajas que ofrece:

- Crear aplicaciones RIA [10] mediante javaScript, utilizando componentes predefinidos.
- Ser utilizado sobre cualquier navegador.
- Proveer un balance entre Cliente-Servidor, ya que distribuye la carga, permitiendo que el servidor gestione más clientes al mismo tiempo.
- Poseer un API [11]fácil de usar.
- Contar con licencias comerciales y de código abierto.
- Desarrollar interfaces parecidas a las aplicaciones de escritorio.
- La orientación a objetos intensa hará modular todos los scripts.
- El diseño está completamente separado de la funcionalidad.
- Funciones comunes como validación, combo boxes editables, ventanas arrastrables (con minimizar y maximizar), grid editables, son muy fáciles de implementar.

1.8.5 Sauxe Framework

El framework Sauxe 1.5.4 se estructura de manera tal que facilita la reutilización de los diferentes componentes y es de fácil entendimiento para quienes desarrollen sobre el mismo. Sauxe utiliza ExtJS

10 Aplicaciones de Internet Enriquecidas del inglés Rich Internet Applications
11 Interfaz de Programación de Aplicaciones del inglés Application Programming Interface.

para implementar la capa de presentación, se apoya en Zend-Ext, una extensión de Zend Framework para el desarrollo de la lógica del negocio y para la gestión de los datos utiliza Doctrine. Este utiliza como estilo arquitectónico Modelo Vista Controlador. También tiene como propósito insertar la programación orientada a aspectos así como la inversión de controles. El framework en su nueva configuración define que la conexión a la base de datos será configurada en un archivo de tipo XML almacenado en la carpeta de recursos comunes del proyecto. Es válido aclarar que este cuenta con un componente de transacciones mediante el cual serán salvados automáticamente los datos de modificaciones e inserciones. Es decir, ya no será necesaria la implementación por parte del programador de las consultas de inserción, este solamente deberá programar la obtención de los campos de la presentación y dentro del Sauxe el Doctrine es el responsable de que los mismos sean guardados en la base de datos. (Hernandez, 2011)

1.8.6 Navegador Web

Mozilla Firefox 8.0

Es un navegador de Internet libre y de código abierto. Es usado para visualizar páginas web. Incluye corrector ortográfico, búsqueda progresiva, marcadores dinámicos y un sistema de búsqueda integrado que utiliza el motor de búsqueda que desee el usuario. Además se pueden añadir funciones a través de complementos desarrollados por terceros. Las características de Mozilla Firefox son las siguientes: (Mozilla Firefox, 2009)

- Multiplataforma.
- Cuenta con una protección antiphishing, antimalware e integración con el antivirus.
- La navegación por pestañas.
- Bloqueador de ventanas emergentes.
- Múltiples Extensiones.

Utiliza el sistema SSL para proteger la comunicación con los servidores web, utilizando fuerte criptografía cuando se utiliza el protocolo HTTPS[12].

[12] **HTTPS:** (Hypertext Transfer Protocol Secure) Protocolo de transferencia de hipertexto seguro.

1.9 Pruebas de software

Las pruebas del software son un elemento crítico para la garantía de calidad del software y representa una visión final de las especificaciones, del diseño y de la codificación. Una vez generado el código fuente, el software debe ser probado para descubrir y corregir, el máximo de errores posibles antes de su entrega al cliente. Glen Myers en su libro estableció varias normas que pueden servir acertadamente como objetivos de las pruebas, estas son:

- La prueba es el proceso de ejecución de un programa con la intención de descubrir un error.
- Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.
- Una prueba tiene éxito si descubre un error no detectado hasta entonces.

Los datos que se van recogiendo a medida que se lleva a cabo la prueba proporcionan una buena indicación de la fiabilidad del software y de alguna manera, indican la calidad del software como un todo. (Pressman, 2006)

1.9.1 Pruebas de Caja blanca

La prueba de caja blanca, denominada a veces prueba de caja de cristal es un método de diseño de casos de prueba que utiliza la estructura de control del diseño procedimental para obtener los casos de prueba. (Pressman, 2006)

1.9.2 Pruebas de Caja negra

Las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software. O sea, la prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. (Pressman, 2006)

Los métodos empleados en la caja negra para asegurar la calidad de la aplicación desarrollada fueron:

1.9.2.1 Partición equivalente

La partición equivalente es un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal

descubre de forma inmediata una clase de errores que de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. (Pressman, 2006)

1.9.2.2 Análisis de valores límite

Los errores tienden a darse más en los límites del campo de entrada que en el centro. Por ello, se ha desarrollado el análisis de valores límite (AVL) como técnica de prueba. El análisis de valores límite es una técnica de diseño de casos de prueba que complementa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el AVL lleva a la elección de casos de prueba en los extremos de las clases. En lugar de centrarse solamente en las condiciones de entrada, el AVL obtiene casos de prueba también para el campo de salida. (Pressman, 2006)

1.10 Conclusiones

Se definieron primeramente conceptos claves para el entendimiento de los propósitos planteados en este documento. Luego de analizar los sistemas de gestión de los posgrados a nivel mundial, se arriba a la conclusión de que ninguno cubría todas las necesidades que requiere el sistema. Por tal motivo es vital la elaboración de un sistema que gestione los procesos relacionados con los posgrados de CEIGE y capaz de proporcionar respuestas rápidas y de forma eficiente para todos sus usuarios. La dirección del proyecto decide utilizar, luego de realizar una valoración de todas las ventajas y facilidades que presentan las metodologías y tecnologías el uso de estas para el desarrollo de los módulos, componentes y subsistemas que a continuación se refieren, UML para la realización de los diferentes diagramas efectuados durante el negocio, el análisis y diseño, el Modelo de Desarrollo de Software Orientado a Componentes, el mismo permitirá la generación de artefactos de vital importancia en el análisis y el diseño como son: Modelo de proceso de negocio, Modelo conceptual, Prototipo de interfaz de usuario, Especificación de requisitos, Diagrama de clases y Descripción del diseño de clases, como herramienta para el modelado se utiliza Visual Paradigm, como propuesta para el desarrollo se utilizará PHP v5.2 como lenguaje de programación, como herramienta para la programación a través de los lenguajes del lado del cliente y del servidor Netbeans, como servidor de aplicaciones Web Apache v 2.0 y como Sistema Gestor de Base de Datos Postgre SQL v 8.3. Se realizó un estudio de la arquitectura propuesta para la solución del sistema, así como las pruebas para validar el correcto funcionamiento del mismo.

CAPÍTULO 2: ANÁLISIS Y DISEÑO

2.1 Introducción

Para lograr una visión general de la herramienta que se desea desarrollar. Se hace una descripción de los procesos del negocio que serán automatizados, será enunciada la propuesta del sistema. Se presentan además, las reglas que tiene que cumplir el negocio así como también se muestra el modelo de objetos. Las funcionalidades que brinda el sistema teniendo como resultado los requisitos funcionales y no funcionales del sistema que darán solución a los problemas existentes.

2.2 Propuesta de Sistema

El Sistema para la administración de posgrado en el centro CEIGE será capaz de gestionar la información referente a los estudiantes de posgrado de forma rápida y segura permitiendo un control adecuado de los cursos de los estudiantes de posgrado, los programas académicos en los cuales pueden estar matriculados, estado de completamiento de los créditos y proceso de la categorización docente. Un control adecuado de las ediciones y estados de los estudiantes de posgrado respecto a las mismas. Se podrá matricular un conjunto de profesores en un programa y en una edición, mostrar el listado de profesionales que están realizando el cambio de categoría. Mostrar el historial de ediciones de cursos asociadas a un profesional mostrando el estado en cada una de ellas.

Por tanto, el sistema debe permitir adicionar, modificar, y eliminar los Cursos, Programas, Ediciones.

2.3 Modelación de los procesos de negocio

La modelación de los procesos de negocio permite realizar una exploración del dominio del problema, con el fin de lograr comprensión por parte del equipo de desarrollo de los procesos que se realizan en la formación posgraduada y la relación que existe entre estos. De esta forma se van determinando necesidades operacionales, obteniéndose finalmente un entendimiento del negocio para dar paso a la fase inicial del sistema, lográndose a través de los objetivos siguientes: (Unidad de Compatibilización Integración y Desarrollo, 2009)

- Realizar un estudio de los procesos existentes con el fin de contribuir con el principio de reutilización.
- Detallar las características del negocio a través de la descripción de los procesos.
- Verificar que se haya realizado un buen análisis del negocio

2.3.1 Mapa de procesos del negocio

Un proceso de negocio es un conjunto de tareas relacionadas lógicamente que se llevan a cabo para lograr como resultado un negocio definido. Los procesos describen como es realizado el trabajo en la organización y se caracterizan por ser observables, medibles, mejorables y repetitivos.

Para comprender los procesos del negocio se realizó un previo estudio de los procesos de posgrado así como entrevistas con los clientes para garantizar un mejor entendimiento con el desarrollador. Como resultado final fue elaborada matriz de relación entre los procesos de negocio y posteriormente validado por el cliente.

Tabla 1: Matriz de relación entre los procesos de negocio

		Entradas	
		Administración de persona	Gestión de la formación posgraduada
Salidas	Administración de persona		Expediente de la persona
	Gestión de la formación posgraduada		Curso Programa Edición

2.3.2 Descripción del proceso de negocio

Tabla 2: Descripción del proceso de negocio Gestión de la formación posgraduada

Descripción del proceso Gestión de la formación posgraduada. Objetivo	Realizar las creaciones de los programas y cursos necesarios para la creación de las ediciones que pueden pasar los estudiantes de posgrado
Evento(s) que lo genera(n)	NA
Pre condiciones	NA
Marco legal	NA
Clientes externos	Dirección de investigación y posgrado del CEIGE

Entradas	NA
Flujo de eventos	
Flujo básico Realizar cuadres financieros	
1.	Definir los programas para los estudiantes de posgrado
2.	Definir los cursos por cada programa de posgrado
3.	Definir las ediciones de los cursos y los programas
Post-condiciones	
1.	Se obtienen los programas para los estudiantes de posgrado
2.	Se obtienen los cursos por cada programa de posgrado
3.	Se obtienen las ediciones de los cursos
Salidas	
Programas, Cursos, Ediciones	
Flujos paralelos	
N/A	
Salidas	
N/A	
Flujos alternos	
N/A	
Salidas	
N/A	
Asuntos pendientes	

Diagrama del proceso Gestión de la formación posgraduada

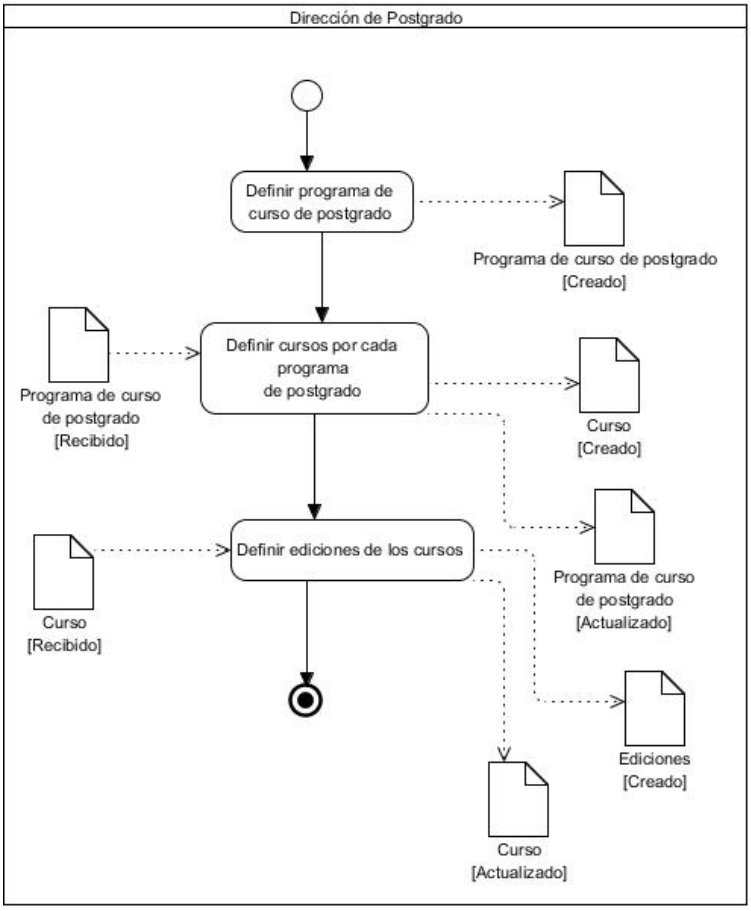

Figura 3: Diagrama del proceso Gestión de la formación posgraduada

2.3.3 Modelo Conceptual

Para descomponer el dominio del problema hay que identificar los conceptos, los atributos y las asociaciones del dominio que se juzgan importantes. El resultado puede expresarse en un modelo conceptual, que no es más que una representación visual de los conceptos u objetos del mundo real, significativos para un problema o área de interés. (Unidad de Compatibilización Integración y Desarrollo, 2009)

El modelo conceptual del proceso

El modelo conceptual del proceso Gestión de la formación posgraduada fue confeccionado con todas las clases conceptuales identificadas en este proceso junto con los atributos y relaciones existentes entre las mismas (consultar Figura 4)

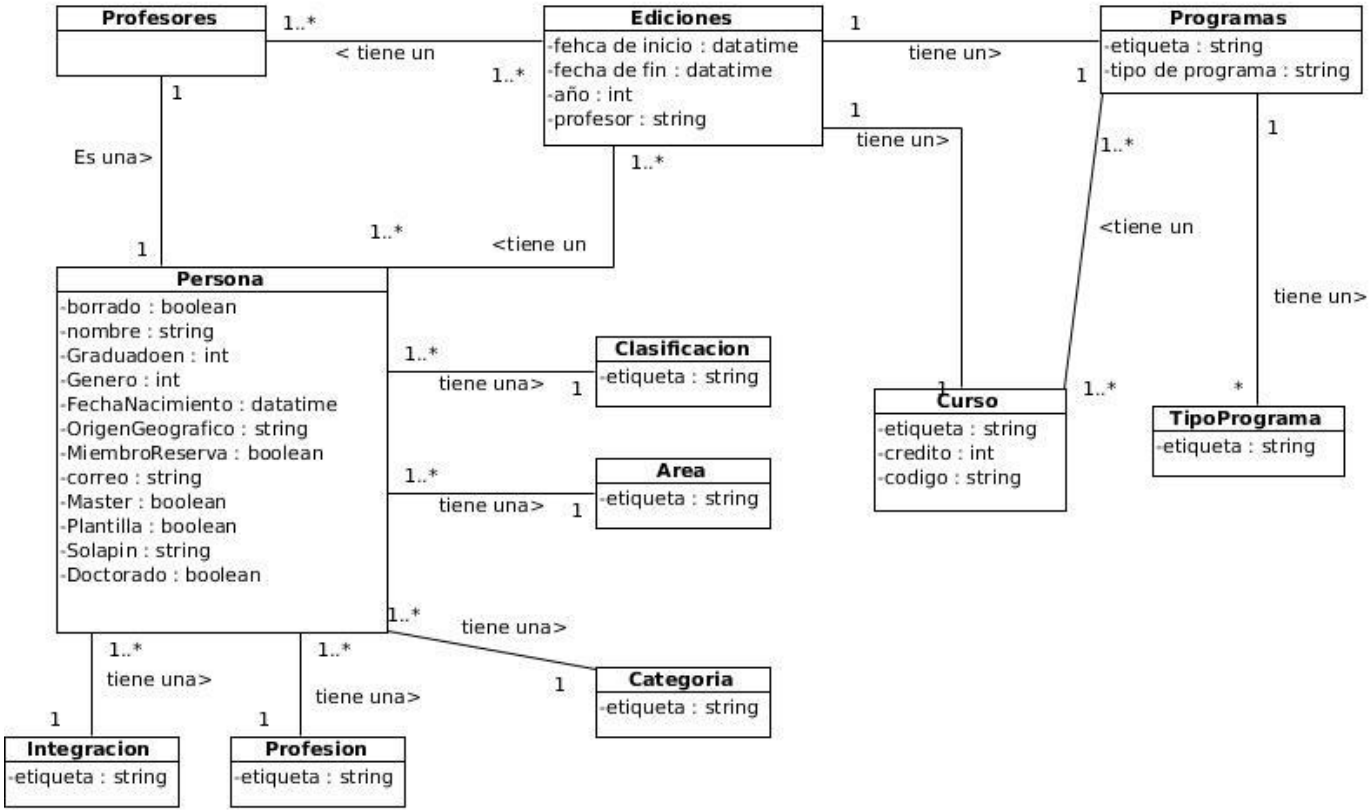

Figura 4: Modelo conceptual del proceso Gestión de la formación posgraduada

2.4 Requisitos

Un requisito es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de este. (Sommerville, 2005) Pueden dividirse en dos categorías: requerimientos funcionales y requerimiento no funcionales. Los requerimientos funcionales son los que definen las funciones que el sistema será capaz de realizar, describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Es importante que se describa el ¿Qué? y no el ¿Cómo? se deben hacer esas transformaciones. (La Ingeniería de Requerimientos y su importancia en el desarrollo de proyectos de software, 2005)

La captura de requisitos es la actividad mediante la cual el equipo de desarrollo de un sistema de software extrae, de cualquier fuente de información disponible, las necesidades que debe cubrir dicho sistema. El proceso de captura de requisitos puede resultar complejo, principalmente si el entorno de trabajo es desconocido para el equipo de analistas, y depende mucho de las personas que participen en él. Por la

complejidad que todo esto puede implicar, la ingeniería de requisitos ha trabajado desde hace años en desarrollar técnicas que permitan hacer este proceso de una forma más eficiente y precisa. (Escalona, y otros, 2002)

Antes de comenzar el proceso de captura de los requisitos del Sistema para la administración de posgrado en el centro CEIGE se identificaron los proveedores válidos de requisitos teniendo en cuenta algunos criterios como: conocimiento del negocio, disponibilidad de tiempo, habilidades de comunicación, conocimiento de informática en aras de identificar a las personas con mayor dominio del para así obtener los requerimientos a implementar.

2.4.1 Definición de requisitos

El propósito de la definición de requisitos es especificar las condiciones o capacidades que el sistema debe cumplir y las restricciones bajo las cuales debe operar, logrando un entendimiento entre el equipo de desarrollo y el cliente, y especificando las necesidades reales de forma que cumpla sus expectativas.

2.4.2 Técnicas para la captura de requisitos

Se aplicaron las siguientes técnicas para la captura de los requisitos:

Entrevista: A través de encuentros planificados con el cliente se realizaron preguntas relacionadas con el proceso de Posgrado en función de obtener las necesidades que tenían de informatización. Esto proporcionó como resultado que fueran identificados de forma satisfactoria los requisitos por los que se guiaría el desarrollo del módulo, a través del cual se realizaría la gestión de este proceso. (Pérez Olmos, 2009)

Tormenta de ideas: En cada encuentro fueron debatidos los requisitos identificados, fluyendo variadas ideas acerca de los mismos, lo cual logró como resultado que se obtuviera una visión más amplia de lo que se quería implementar, favoreciéndose de esta forma el avance de futuras etapas en el desarrollo del sistema, a través del cual se llevaría a cabo la gestión de Posgrado. (Pérez Olmos, 2009)

Sistemas existentes: utilizada para el análisis de distintos sistemas ya desarrollados que estén relacionados con el sistema a ser construido y observar las distintas salidas que los sistemas producen (listados, consultas, etc.), para analizar las nuevas ideas que pueden surgir sobre la base de estas. (Pérez Olmos, 2009)

2.4.3 Identificación y clasificación de requisitos

Requisitos funcionales

RF.1 Agrupación de requisito Gestionar programas

1.1 Adicionar programas

1.2 Modificar programas

1.3 Eliminar programas

1.4 Listar programas

1.5 Mostrar listado de cursos asociados

RF.2 Agrupación de requisito Gestionar cursos

2.1 Adicionar cursos

2.2 Modificar cursos

2.3 Eliminar cursos

2.4 Listar cursos

RF.3 Agrupación de requisito Gestionar ediciones

3.1 Adicionar ediciones

3.2 Modificar ediciones

3.3 Eliminar ediciones

3.4 Listar ediciones

3.5 Buscar ediciones

RF.4 Agrupación de requisito Gestionar persona

4.1 Adicionar persona

4.2 Modificar persona

4.3 Eliminar persona

4.4 Listar persona

4.5 Buscar persona

RF.5 Agrupación de requisito Gestionar clasificación

5.1 Adicionar clasificación

5.2 Modificar clasificación

5.3 Eliminar clasificación

5.4 Listar clasificación

RF.6 Agrupación de requisito Gestionar área de trabajo

6.1 Adicionar área de trabajo

6.2 Modificar área de trabajo

6.3 Eliminar área de trabajo

6.4 Listar área de trabajo

RF.7 Agrupación de requisito Gestionar categoría docente

7.1 Adicionar categoría docente

7.2 Modificar categoría docente

7.3 Eliminar categoría docente

7.4 Listar categoría docente

RF.8 Agrupación de requisito Gestionar integración

8.1 Adicionar integración

8.2 Modificar integración

8.3 Eliminar integración

8.4 Listar integración

RF.9 Agrupación de requisito Gestionar profesión

9.1 Adicionar profesión

9.2 Modificar profesión

9.3 Eliminar profesión

9.4 Listar profesión

RF.10 Mostrar listado de profesionales que están realizando el cambio de categoría

RF.11 Asociar profesional a curso

RF.12 Quitar profesional a curso

RF.13 Mostrar historial de ediciones

RF.14 Mostrar profesionales por cursos

RF.15 Mostrar profesionales por programa

RF.16 Mostrar profesionales por edición

RF.17 Mostrar cantidad de ediciones concluidas, activas y abandonadas de cada profesional

Requisitos no funcionales

- **Requisitos de Interfaz:**

La interfaz debe ser amigable y fácil de usar.

El sistema debe seguir el estándar para el diseño de interfaces definido.

Los errores cometidos por el usuario les serán notificados.

- **Requisitos de Usabilidad:**

La herramienta propuesta será usada por personas que no necesariamente tienen habilidades con el trabajo en la computadora de manera que no puede ser una dificultad para el usuario el uso de ella.

- **Requisitos de Rendimiento:**

La herramienta propuesta debe ser rápida y el tiempo de respuesta debe ser el mínimo posible.

- **Requisitos de Portabilidad:**

La herramienta propuesta podrá ser usada bajo cualquier sistema operativo, para su implementación se usará PHP y PostgreSQL que son multiplataforma.

- **Requisitos de Seguridad:**

Autenticación (Contraseña de acceso).
Se concederá acceso al sistema a partir de un nombre de usuario y una contraseña.

El sistema concederá acceso a cada usuario autenticado solo a las funciones que le estén permitidas, de acuerdo a la configuración del sistema.
Verificación sobre las acciones irreversibles (eliminaciones).

- **Confiabilidad:**

La información manejada por el sistema estará protegida de accesos no autorizados y divulgación

El sistema debe garantizar protección contra acciones no autorizadas o que puedan afectar la integridad de los datos.

- **Integridad:**

La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción y estados inconsistentes, de la misma forma será considerada igual a la fuente o autoridad de los datos.

- **Requisitos de Software:**

En el equipo servidor, independientemente del sistema operativo, se necesitará el servidor web Apache2 y como gestor de bases de datos PostgresSql 8.3 o superior. En el equipo cliente sólo se requerirá del navegador Mozilla Firefox 8.0 o superior.

2.4.4 Especificación de requisitos

A continuación se muestra la descripción del requisito funcional Gestionar programas.

Descripción del requisito Gestionar edición

Tabla 3: Descripción del requisito gestionar edición

Precondiciones	Debe haberse autenticado ante el sistema Deben existir cursos, profesores y programas en el sistema
Flujo de eventos	
Flujo básico	
1	El sistema permite gestionar una edición
2	Se introducen los datos de la edición Curso Programa Profesor Año Fecha de inicio Fecha de fin
3	El sistema valida (ver validación 1) los datos introducidos.

4	Si los datos son correctos el sistema los registra.	
5	El sistema confirma el registro de los datos.	
6	Concluye el requisito.	
Pos-condiciones		
1	Se registró en el sistema una nueva edición	
Flujos alternativos		
Flujo alternativo 4.a Información errónea		
1	El sistema señala los datos erróneos y permite corregirlos.	
2	El usuario corrige los datos.	
3	Volver al paso 3 del flujo básico.	
Pos-condiciones		
1	NA	
Flujo alternativo 4.b Información incompleta.		
1	El sistema señala los datos vacíos y permite corregirlos.	
2	El usuario corrige los datos.	
3	Volver al paso 3 del flujo básico.	
Pos-condiciones		
1	NA	
Flujo alternativo *.a El usuario cancela la acción		
1	Concluye el requisito.	
Pos-condiciones		
1	NA	
Validaciones		
1	Se validan los datos según lo establecido en el Modelo conceptual	
Relaciones	**Requisitos Incluidos**	NA
	Extensiones	NA
Conceptos	**Fuente de Datos**	Visibles en la interfaz: Tipo Dirección
Requisitos especiales	NA	
Asuntos pendientes	NA	

Prototipo elemental de interfaz gráfica de usuario

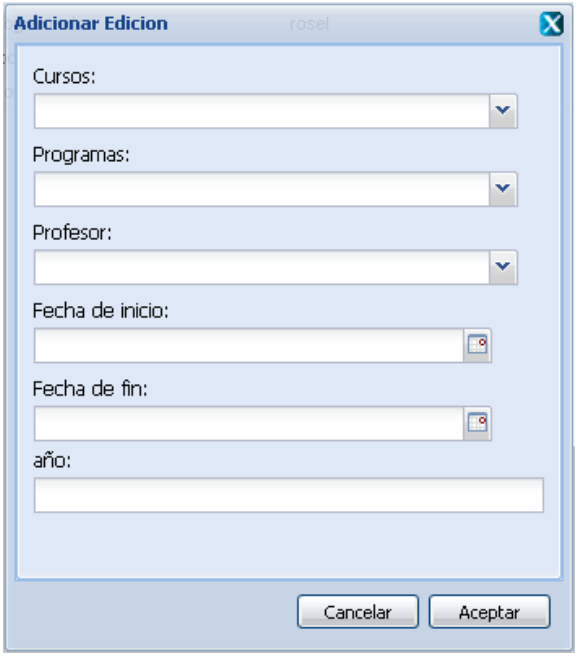

Figura 5: Prototipo de interfaz de usuario: Adicionar edición

2.4.5 Validación de los requisitos

La validación de los requisitos se realiza con la finalidad de comprobar que los requisitos identificados sean precisos, consistentes, realistas, verificables, definan lo que el usuario desea del producto final, que los errores que hayan sido detectados sean corregidos y el resultado del trabajo cumpla con los estándares establecidos para el proceso, el proyecto y el producto. (Pressman, 2006)

Para la validación de los requisitos fueron aplicadas las siguientes técnicas:

Revisión técnica formal: La revisión de las especificaciones de los requisitos del proceso Gestión de la formación posgraduada fue realizada con el director de Investigación y Posgrado del CEIGE. Con la utilización de esta técnica se validó que no existieran errores en el contenido o malas interpretaciones, información incompleta, inconsistencias y que los requisitos no fueran contradictorios, imposibles o inalcanzables, dando como resultado que fueran aprobados los que estaban descritos de forma correcta, clara y consistente.

Prototipos: A partir de las especificaciones de requisitos fueron conformados prototipos de interfaz de usuario. Con esto se validó que los requisitos estaban en concordancia con las necesidades plasmadas por el cliente. El empleo de esta técnica ofreció como resultados que el cliente tuviera una idea de la

estructura de la interfaz de usuario y se favoreciera la comunicación con el mismo, ya que tenía una visión inicial del módulo a través del cual se gestionaría el proceso Gestión de la formación posgraduada.

Generación de casos de prueba (test de requisitos): Fueron definidos y diseñados casos de pruebas para cada requerimiento especificado, con el objetivo de verificar el cumplimiento de los mismos. La utilización de esta técnica ofreció los siguientes resultados: fueron identificados los posibles escenarios de los requisitos, así como los juegos de datos de los campos determinados en estos, se validaron y aprobaron los que estaban bien enunciados, descritos y consistentes.

Con la aplicación de estas técnicas se demostró que los requisitos estaban descritos de forma correcta, respondían a las necesidades planteadas por el cliente y no existían ambigüedades entre los mismos.

2.5 Diseño

2.5.1 Mecanismos de diseño

Los mecanismos de diseño se utilizan con el objetivo de simplificar los diagramas de clases. Cada diseñador acorde a sus necesidades establece sus propios mecanismos de diseño, teniendo en cuenta los patrones y estilos seleccionados. (Cabrera Pereira., y otros, 2009)

En el Sistema para la administración de posgrado en el CEIGE se definieron los siguientes mecanismos:

2.5.1.1 Mecanismo de diseño para las páginas clientes.

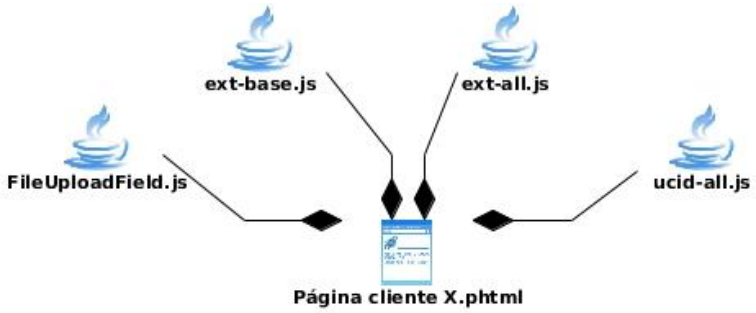

Figura 6: Mecanismo de diseño para las páginas cliente

ext-all.js: Se encuentra entre las clases que posee el Ext Js y es la encargada de crear los componentes visuales de la vista.

ucid-all.js: Encargada de mostrar la interfaz estándar.

ext-base.js: Es la encargada de manejar las solicitudes, respuestas y componentes de Ext Js.

FileUploadField.js: Su función es la de cargar los ficheros.

2.5.1.2 Mecanismo de diseño para las clases controladoras.

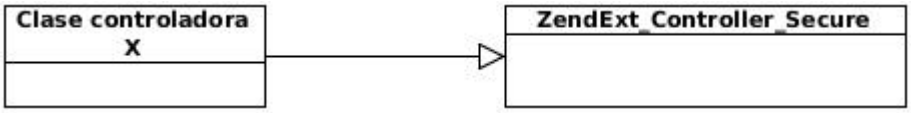

Figura 7: Mecanismo de diseño para las clases controladoras

Todas las clases controladoras que se definieron en el diseño del Sistema para la administración de posgrado en el CEIGE heredan de la clase ZendExt_Controller_Secure, ya que posee funcionalidades comunes para todas.

2.5.1.3 Mecanismo de diseño para el nombre de las clases.

A partir de las agrupaciones de requisitos funcionales que tuvieran relación entre sí fueron elaboradas las clases del diseño del Sistema para la administración de posgrado en el CEIGE. Las clases fueron denominadas según las funcionalidades que estas realizan. Las que agrupan los requisitos adicionar, modificar y eliminar, se nombran AMENombreDeLaClase, las que agrupan listar y buscar, LBNombreDeLaClase, mientras las que tienen como función mostrar las relaciones son nombradas según los datos que muestran.

2.5.1.4 Mecanismos de diseño para las clases modelo.

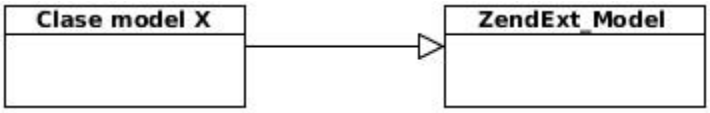

Figura 8: Mecanismo de diseño para las clases modelo

Todas las clases modelo que fueron definidas en el diseño heredan de la clase ZendExt_Model, la cual posee las principales funciones para el manejo de los datos.

2.5.2 Modelo de Diseño.

El modelo de diseño proporciona detalles acerca de las estructuras de datos, las arquitecturas, las interfaces y los componentes del software que son necesarios para implementar el sistema. (Pressman, 2006)

2.5.3 Diagramas de clases del diseño con estereotipos web

Durante el diseño, se modela el sistema de manera que soporte todos los requisitos funcionales. Este modelo además sirve de abstracción de la implementación del sistema y es utilizado como entrada fundamental de las actividades de implementación y para soportar las técnicas de programación gráfica de la aplicación. También se declaran los métodos y atributos teniendo en cuenta el lenguaje de programación en el que se desarrolla el sistema.

Como resultado del estudio del análisis de definieron 3 diagramas de clases del diseño a las que se le aplicaron patrones de arquitectura, de asignación de responsabilidades y de diseño. Como patrón arquitectónico se utilizó el MVC, mediante el cual se realiza, como muestra la Figura 6: Diagrama de clases del diseño Gestionar ediciones, un diseño que desacople la vista del modelo, con la finalidad de mejorar la reusabilidad. De esta forma las modificaciones en la vista impactan en menor medida en la lógica de negocio o de datos. Por otra parte se refleja los patrones de GRASP, Alta cohesión y Bajo acoplamiento que como se puede ver en la figura antes mencionada, existe poca interacción entre las clases y las mismas no están cargadas de responsabilidades, proporcionando una alta reutilización, mejoras en la claridad y facilidad para entender el diseño y los cambios que se realicen no afecten a otros componentes.

A continuación se encuentran los diagramas de clases del diseño para cada agrupación de los requisitos, de forma tal que se facilite la comprensión de las relaciones entre los distintos componentes.

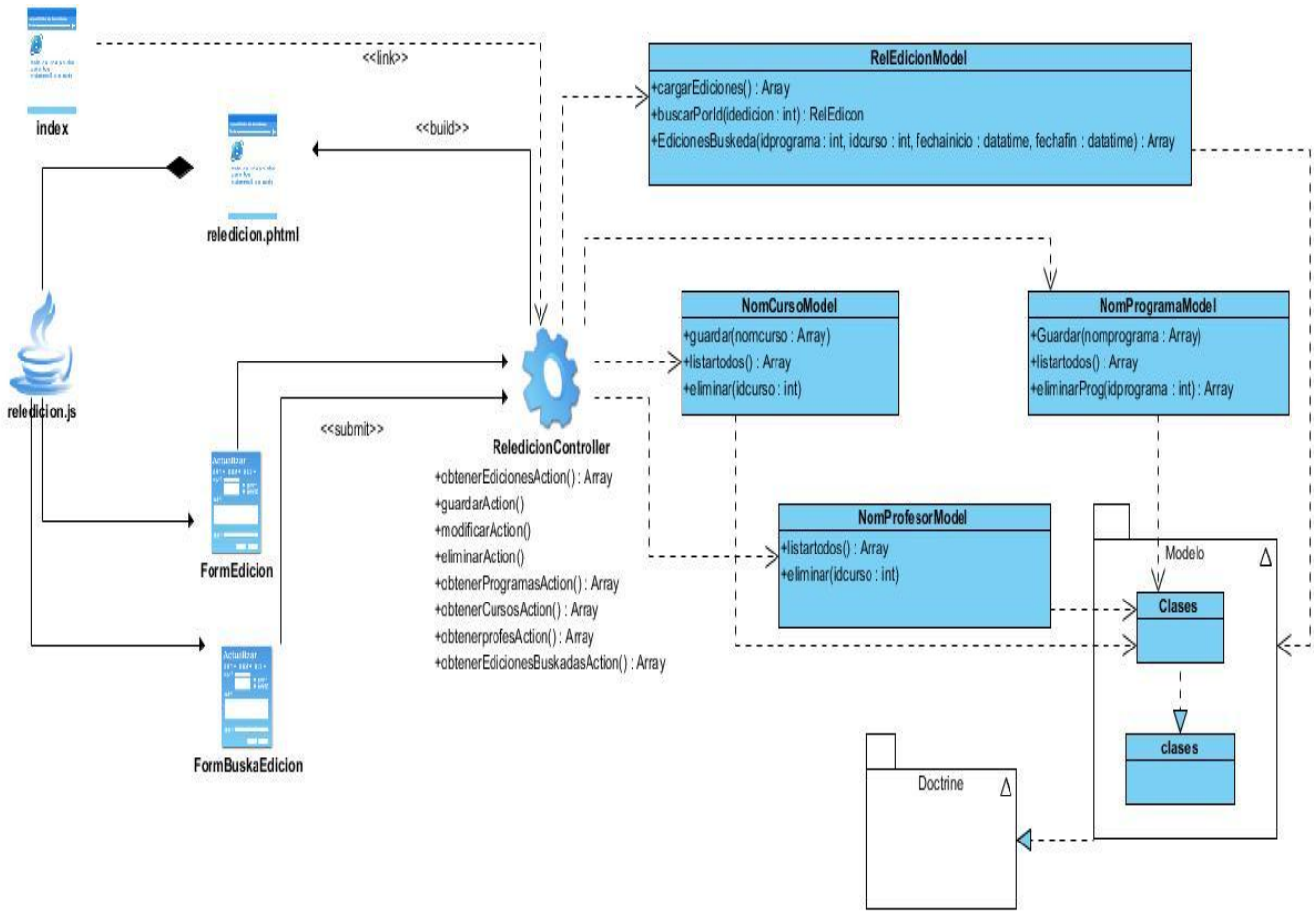

Figura 9: Diagrama de clases del diseño Gestionar ediciones

2.5.4 Descripción de las clases del diseño

La descripción de las clases del diseño tiene como objetivo de poseer un mayor entendimiento del funcionamiento de las mismas.

Tabla 4: Descripción de la clase ReledicionController

ReledicionController	
Tipos de clase:	**Controladora**
Nombre	guardarAction()

Descripción	Crear un objeto de tipo Reledicion con los datos introducidos para guardarlo en la Base de Datos
Nombre	modificarAction()
Descripción	Modifica los datos de un objeto de tipo Reledicion por los introducidos
Nombre	eliminarAction()
Descripción	Elimina una Edición seleccionado por el usuario
Nombre	obtenerEdicionesAction()
Descripción	Retorna un arreglo con todos las ediciones existentes
Nombre	obtenerProgramasAction()
Descripción	Retorna un arreglo con todos los programas existentes
Nombre	obtenercursosAction()
Descripción	Retorna un arreglo con todos los cursos existentes
Nombre	obtenerprofesAction()
Descripción	Retorna un arreglo con todos los profesores existentes
Nombre	obtenerEdicionesBuskadasAction()
Descripción	Retorna un arreglo con las ediciones seleccionadas según los parámetros (idprograma y/o idcurso y/o fechainicio, fechafin)

Tabla 5: Descripción de la clase ReledicionModel

ReledicionModel

Tipos de clase:	**Modelo**
Nombre	addEdicion($Rel)
Descripción	recibe un objeto de tipo Reledicion con los datos introducidos por el usuario y lo guarda en la base de datos
Nombre	eliminarEdicion($idedi)
Descripción	Elimina una Edición especificado por el idedicion de la base de datos
Nombre	cargarEdiciones()
Descripción	Retorna un arreglo con todas las ediciones existentes en la base de datos
Nombre	buscarPorId($idedi)
Descripción	Retorna una edición especificada por el id de la misma
Nombre	obtenerEdicionesBuskeda($idprograma,$idcurso,$fechainicio,$fechafin)
Descripción	Retorna un arreglo con las ediciones seleccionadas según los parámetros (idprograma y/o idcurso y/o fechainicio, fechafin)

2.5.5 Patrones de diseño

Los Patrones de Diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Existen un innumerable número de estos, todos adecuados a dar soluciones a problemáticas existentes.

2.5.5.1 Patrón “Modelo-Vista-Controlador” (MVC)

La implementación de este patrón en dicho módulo se realizó de la siguiente manera:

La Vista engloba la capa y la lógica de presentación, en la misma se maneja todo el flujo web. Le brinda al usuario la posibilidad de interactuar con el Controlador mediante los mensajes de eventos y le permite visualizar las respuestas a sus peticiones.

El Controlador por su parte representa el gestor de eventos, el cual está compuesto por las clases controladoras. Este atiende los pedidos que llegan desde la Vista accediendo al paquete *models*.

El Modelo se encuentra dividido en los paquetes *bussines*, el cual encierra las clases donde se realiza la lógica del negocio, y *domain*, donde están comprendidas las clases del dominio, las que se encargan de leer y/o escribir datos en las tablas de la base de datos.

2.5.5.2 Singleton

Este patrón fue utilizado en las clases del dominio, donde los métodos contenidos en estas se hicieron de forma estática, con lo cual se garantizaba que pudieran ser accedidos sin crear una instancia de las mismas. Por ejemplo: para acceder al método buscarPorId() de la clase del NomCurso, desde la clase NomCursoModel, solo es necesario escribir NomCursoModel::buscarPorId().

2.5.5.3 Controlador

Es un patrón de diseño web utilizado como único punto de entrada a la aplicación. Este patrón se puede ver con la creación de las clases controladoras para los diferentes procesos, que se realizan para la gestión de los Cursos, Programas y Ediciones se crearon las clases ReledicionController, NomProgramaController, NomCursoController.

2.5.5.4 Experto

Es un patrón de asignación de responsabilidades (GRASP, del inglés General Responsibility Assignment Software Patterns) que delega las responsabilidades a quién contiene la información necesaria para cumplirlas. Fue utilizado en todas las clases definidas.

2.5.5.5 Mediador

Dada la existencia de relaciones entre las clases de mucho a mucho fue necesaria la utilización de este patrón para la creación de la tabla generada por esta relación. Ejemplo: de las clases NomCurso y NomPrograma se creó la clase RelCursoPrograma generada de la relación mucho a mucho existente.

2.5.5.6 Alta cohesión

Su empleo está dado en que fueron asignadas responsabilidades a las clases de forma tal que la cohesión siguiera siendo alta, o sea, cada clase se encargará de realizar solamente las funciones que estén en correspondencia con la responsabilidad que posea.

2.5.6 Métricas para evaluar el diseño propuesto

Las métricas son una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado. Permiten descubrir y corregir problemas potenciales antes de convertirse en defectos catastróficos. Se emplean con el objetivo de llevar un control de la calidad del producto que se está desarrollando, evaluar la efectividad del proceso y mejorar la calidad del trabajo. (Pressman, 2006)

Una vez realizado el diseño del Sistema para la administración de posgrado en el centro CEIGE, se dio paso a su validación, para esto se utilizaron las métricas orientadas a objetos, específicamente a las clases, ya que las medidas y métricas para una clase individual, la jerarquía y las colaboraciones de estas permiten al ingeniero de software evaluar la calidad del diseño propuesto. (Pressman, 2006)

2.5.6.1 Métricas propuestas por Lorenz y Kidd

En su libro sobre las métricas orientadas a objetos, dividen las métricas basadas en clases en cuatro amplias categorías: tamaño, herencia, valores internos y valores externos. Las orientadas al tamaño para las clases orientadas a objetos se centran en el recuento de atributos y operaciones para cada clase individual y los valores promedio para el sistema orientado a objetos como un todo. (Pressman, 2006)

La métrica empleada fue **Tamaño operacional de clase (TOC),** pues se ajusta para evaluar al diseño realizado y es fácil de usar. Debido a la cantidad de clases que posee el módulo, solo se le aplicó a las clases modelos y controladoras de los diagramas realizados, estas clases son las más significativas y tienen la mayor cantidad de operaciones en el sistema.

Para la evaluación de las clases fueron utilizados umbrales para el tamaño general, la responsabilidad, la complejidad y la reutilización de las clases.

Tabla 6: Umbrales para la TOC

Clasificación	Valores de los umbrales
Pequeño	<=Promedio de operaciones(PO)

Medio	>PO y <=2*PO
Grande	>2*PO

Tabla 7: Tamaño operacional de clase según sus atributos y operaciones

No	Nombre	Cantidad de atributos	Cantidad de operaciones	Tamaño
1	NomcursoController	**0**	**5**	**Pequeño**
2	NomprogramaController	**0**	**11**	**Medio**
3	NomtipoprogramaController	**0**	**4**	**Pequeño**
4	RelcursoprogramaController	**0**	**2**	**Pequeño**
5	ReledicionController	**0**	**9**	**Medio**
6	NomcursoModel	**0**	**5**	**Pequeño**
7	NomprogramaModel	**0**	**7**	**Medio**
8	NomtipoprogramaModel	**0**	**5**	**Pequeño**
9	RelcursoprogramaModel	**0**	**8**	**Medio**
10	ReledicionModel	**0**	**8**	**Medio**

Nombre: es el nombre de las clases seleccionadas para la aplicación de la métrica.

Cantidad de atributos: Es la cantidad de atributos que tienen las clases.

Cantidad de operaciones: es la cantidad de operaciones que tiene las clases.

Tamaño: Es el valor obtenido según el promedio de operaciones de las clases.

Tabla 8: Tamaño de las clases según sus atributos y operaciones

Clasificación	Cantidad de clases	Responsabilidad de las clases	Complejidad de implementación de las clases	Reutilización de las clases
Pequeño	5	Baja	Baja	Alta
Medio	5	Medio	Medio	Medio
Grande	0	Alta	Alta	Baja

Tabla 9: Tamaño de las clases según sus atributos y operaciones

Cantidad de clases	Cantidad de clases pequeñas	Cantidad de clases grandes	Cantidad de clases medias	Promedio de atributos	Promedio de operaciones
10	5	0	5	0	6

Haciendo un análisis de los resultados obtenidos en el empleo de esta técnica, se puede concluir que el diseño realizado es simple, tiene una calidad aceptable, pues la mayoría de las clases que fueron analizadas se encuentran dentro de la categoría de pequeñas o medianas, los valores de calidad definidos por la métrica utilizada no fueron afectados, demostrándose en los pequeños valores de TOC alcanzados, la implementación de forma general es sencilla, se disminuye en gran medida la responsabilidad de las clases y las pruebas pueden ser realizadas con facilidad.

2.6 Modelo de datos

El modelo de datos es un conjunto de conceptos, reglas y convenciones que permite describir y manipular los datos de un cierto mundo real que se desea almacenar en la base datos. (Rivera, 2011)

En el modelo de datos elaborado para el proceso de posgrado se visualizan las entidades con sus atributos, así como las relaciones entre ellas, estas son las encargadas de almacenar todos los datos necesarios para que el sistema pueda ofrecer las funcionalidades que fueron identificadas durante la captura de requisitos. A continuación se presenta el modelo de datos realizado:

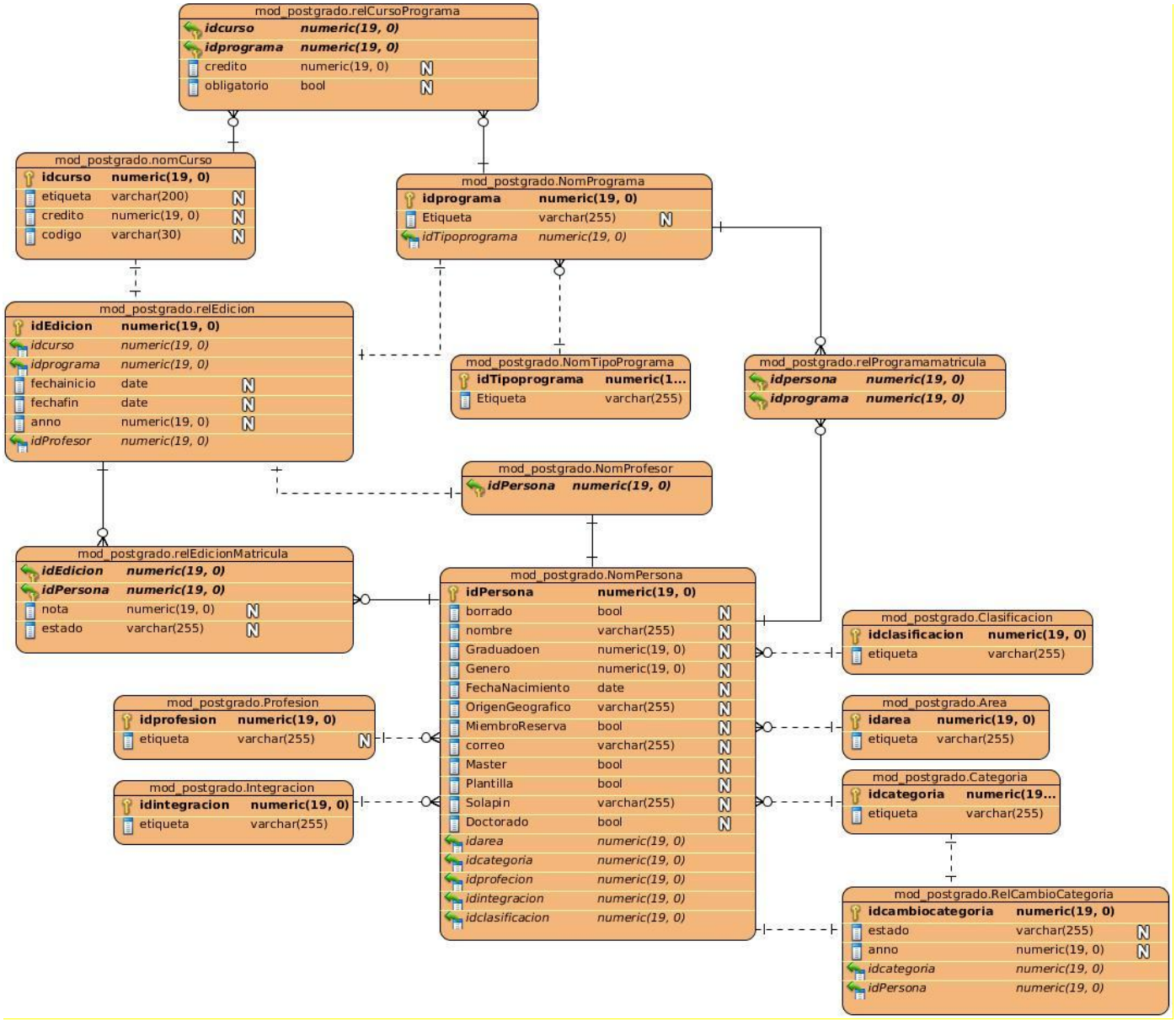

Figura 10: Modelo de datos

2.7 Conclusiones

Con la modelación del negocio del proceso Gestión de la formación posgraduada, fueron generados una serie de artefactos, que crearon las condiciones para la captura de los requisitos funcionales de este proceso, los cuales fueron identificados, especificados y validados, permitiendo con esto darle paso a la etapa de diseño, donde quedaron expuestos los artefactos de esta etapa, así como las métricas empleadas para realizar su validación, que ofrecieron como resultado que el diseño estaba realizado de

forma simple y con una calidad aceptable. Se exponen las cuestiones referentes a los patrones de diseño utilizados, la utilización de los mismos permitió que el diseño sea sencillo, lo que implica una alta reutilización de las clases, baja responsabilidad y complejidad. Estos artefactos generados en el capítulo fueron revisados mediante revisiones técnicas formales con el cliente, garantizando con esto la calidad de los mismos.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS

3.1 Introducción

A partir de los resultados obtenidos del diseño, se comenzará la implementación y las pruebas del sistema. Se describe cómo realizar una exitosa implementación a partir del modelo del diseño en términos de componentes. Se mostrará el diagrama de despliegue, y para realizar la validación del diseño y la implementación se aplicarán métricas del diseño así como pruebas de caja blanca y caja negra.

3.2 Implementación

3.2.1 Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Se realizan con el objetivo de poseer una vista de forma general del sistema a partir de las dependencias e integraciones de los componentes y módulos. En la Imagen 8 se muestra el diagrama de componentes del Sistema de administración de posgrado del centro CEIGE. En este se representa la integración entre los componentes, o sea los servicios que brindan unos y son consumidos por otros y viceversa.

A continuación se muestra el Diagrama de componentes del Sistema de administración de posgrado del centro CEIGE:

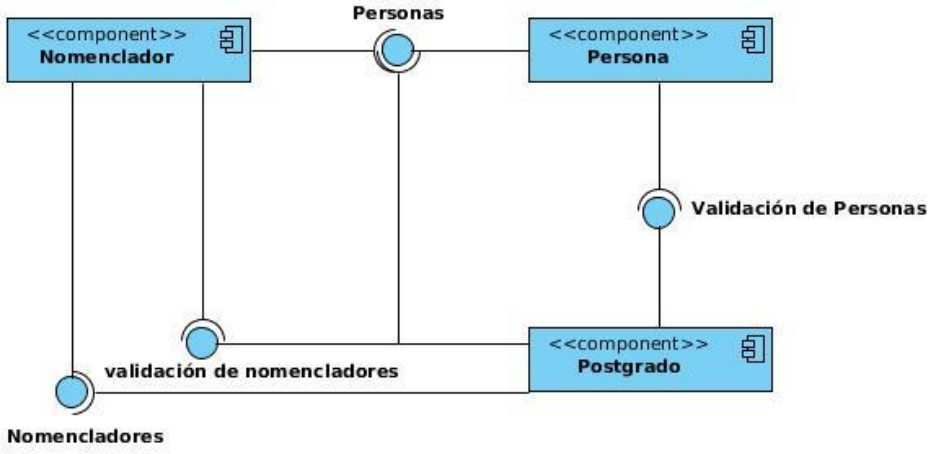

Figura 11: Diagrama de componente

3.2.2 Modelo de despliegue

Los diagramas de despliegue visualizan la disposición física de los distintos recursos computacionales que componen un sistema y el reparto de los componentes sobre dichos nodos. (Universidad de Castilla La Mancha, 2011-2012)

Para el modelo realizado se definió una arquitectura cliente-servidor de la siguiente manera:

Cliente: Accede a la aplicación a través de un computador, donde es ejecutada mediante el navegador Mozilla Firefox, sobre cualquier sistema operativo.

Servidor Web: El servidor de aplicaciones empleado donde radica la lógica de negocio de la aplicación es el Servidor Web Apache2 utilizando el lenguaje PHP.

Servidor de Base de datos: Servidor de Datos PostgreSQL 8.3.

A continuación se presenta el modelo de despliegue elaborado para el Sistema para la administración de posgrado en el centro CEIGE.

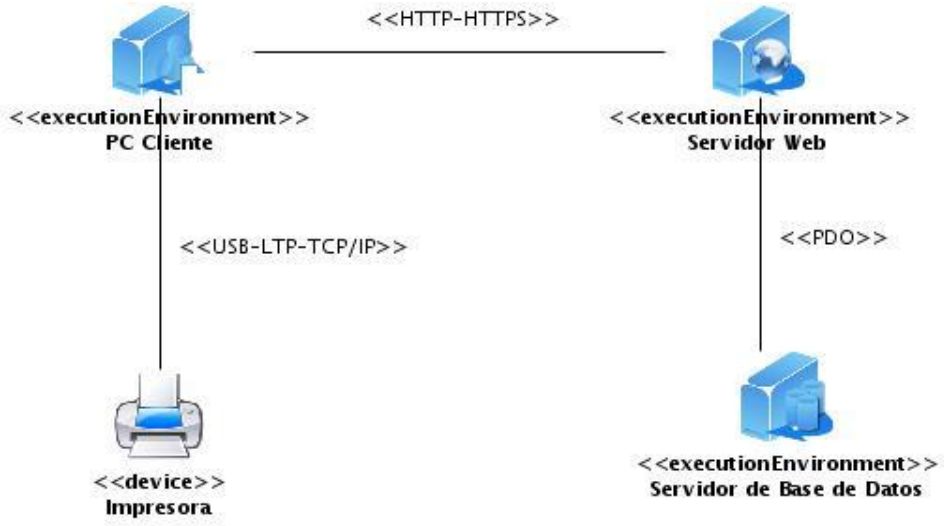

Figura 12: Modelo de despliegue

3.2.3 Estándares de codificación

Los estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, la comprensión y mantenimiento

del código. (Departamento de Electrónica, Sistemas e Informática Coordinación Docente de Programación)

A continuación se especifican los estándares de codificación que fueron utilizados en el desarrollo del Sistema para la Administración de posgrado:

3.2.3.1 Nomenclatura de las clases

- Para las clases controladoras el nombre comenzará con la primera letra en mayúscula y el resto en minúscula, además estará dado según la función que realiza y se le adicionará al final “Controller”. Ejemplo: NomcursoController.
- El nombre de las clases del modelo que se encuentran dentro de la carpeta “bussines” comenzará con la primera letra en mayúscula y el resto en minúscula. A este nombre se le agregará al final “Model” y al inicio se le colocará “Nom”, en caso de que sea un nomenclador, en el caso que sea una relacion sera “Rel” por ejemplo: NomcursoModel, RelcursoprogramaModel.
- Las clases del dominio que se encuentran dentro de la carpeta “domain”, son generadas mediante un mapeo a la base de datos realizado por la herramienta Doctrine Generator v 3.0, la cual fue desarrollada en el CEIGE y son nombradas igual que las tablas de la base de datos. Ejemplo: Nomcurso.
- Las clases contenidas en la carpeta “generated” son generadas mediante un mapeo a la base de datos realizado por la herramienta Doctrine Generator v 3.0, la cual fue desarrollada en el CEIGE y delante del nombre se le coloca “Base”.
- El nombre de las clases de la vista comenzará con la primera letra en mayúscula y el resto en minúscula, en caso de que este sea compuesto, se utilizará la notación PascalCasing. A este nombre se le agregaran al principio, las letras en mayúscula de las funciones que se realizarán a través de estas clases. Ejemplo: AMEpersona (adicionar, modificar y eliminar).

3.2.3.2 Nomenclatura de los métodos o funciones

El nombre de los métodos o funciones de una clase comenzará en minúscula, en caso de que sea compuesto se utilizará la notación CamelCasing, o sea, seguido del primer nombre, comenzará el segundo con mayúscula. En caso de que sea una función de una clase controladora se le agregará al final la palabra “Action”. Ejemplo: guardarAction, obtenertProgramasAction, buscar.

3.2.3.3 Nomenclatura de las variables

Las variables serán nombradas comenzando en minúscula, en caso de que el nombre de estas sea compuesto, se utilizará la notación CamelCasing, o sea, seguido del primer nombre, comenzará el segundo en mayúscula. En la capa de la Vista, las variables que contengan componentes que formen parte de la interfaz, se les colocará delante un sufijo que indique el tipo de componente que contiene. Ejemplo: winEdiciones, btnAdicionarCurso, storeCurso, smCurso, gridCurso.

3.2.4 Publicación de servicios entre componentes

Para cada uno de los componentes fueron publicados en sus clases Services, los mismos métodos que son utilizados en estos para la recuperación de los datos en las clases del modelo y del dominio, realizándose de esta manera con el objetivo de reutilizar código ya existente. Los servicios fueron implementados de modo que reciban muchos o ningún parámetro, siempre que fuera posible, disminuyendo con esto la cantidad de servicios a publicar. Algunas de las ventajas de la utilización de los servicios son:

- Permite desacoplar las clases de sus dependencias de manera de que las mismas puedan ser reemplazadas o actualizadas con muy pocos o casi ningún cambio en el código fuente de sus clases.
- Permite escribir clases que dependan de clases cuyas implementaciones no son conocidas en tiempo de compilación.
- Permite testar las clases aisladamente sin sus dependencias.
- Permite desacoplar sus clases de ser responsables de localizar y gestionar el tiempo de vida de sus dependencias.

El sistema cuenta con las clases Services, AreaService, CategoriaService, NompersonaService y NomprofesorService, las cuales contienen todos los servicios que son consumidos por los restantes componentes del Sistema, con los que se tienen una interrelación.

A continuación se muestran los servicios que son brindados por el componente:

Tabla 10: Servicios que brinda el componente Persona

Servicios	Componentes que lo consumen	Descripción
Listar_Profesores	Nomencladores Posgrado	Retorna un listado de todos los profesores en el sistema.
Buscar_Porid	Nomencladores Posgrado	Busca un profesor según un identificador pasado por parámetros y retorna los datos de la misma.
Buscar_PoridPersonas	Nomencladores Posgrado	Busca una persona según un identificador pasado por parámetros y retorna los datos de la misma.
Obtener_TodasPersonas	Nomencladores Posgrado	Retorna un listado de todas las personas en el sistema.
Buscar_PoridArea	Posgrado	Busca un área según un identificador pasado por parámetros y retorna los datos de la misma.
Obtener_Categorias	Posgrado	Retorna un listado de todas las Categorías en el sistema.

3.3 Pruebas de software

3.3.1 Pruebas de caja blanca

Para utilizar esta técnica primeramente se hace necesario el cálculo de la complejidad ciclomática del algoritmo que vaya a ser analizado, para lo cual se deben enumerar las sentencias de código de este y a partir de ahí elaborar el grafo de flujo de esta funcionalidad. Las sentencias enumeradas del método GuardarPersonaAction() que se muestra en la Figura 15

```
function GuardarPersonaAction() {

    $nombre = $this->_request->getPost('nombre');1
    $solapin = $this->_request->getPost('solapin');1
    $plantilla = $this->_request->getPost('plantilla');1
    $miembroreserva = $this->_request->getPost('miembroreserva');1
    $master = $this->_request->getPost('master');1
    $profesion = $this->_request->getPost('idprofesion');1
    $integracion = $this->_request->getPost('idintegracion');1
    $clasificacion = $this->_request->getPost('idclasificacion');1
    $categoria = $this->_request->getPost('idcategoria');1
    $area = $this->_request->getPost('idarea');1
    $graduadoen = $this->_request->getPost('graduadoen');1
    $genero = $this->_request->getPost('genero');1
    $doctorado = $this->_request->getPost('doctorado');1
    $correo = $this->_request->getPost('correo');1
    $borrado = $this->_request->getPost('borrado');1
    $origengeografico = $this->_request->getPost('origengeografico');1
    $fechanacimiento = $this->_request->getPost('fechanacimiento');1
    $generos = 0;1

    if ($genero == 'Masculino') {2
        $generos = 1;3
    }else 4
        $generos = 2;5

    if ($plantilla == on) { 6
        $plantilla = true; 7
    }else 8
        $plantilla = false; 9

    if ($miembroreserva == on) {10
        $miembroreserva = true; 11
    }else 12
        $miembroreserva = false; 13

    if ($master == on) { 14
        $master = true; 15
    }else 16
```

Figura 13: Código fuente de la funcionalidad GuardarPersonaAction ()

```
    $master = false; 17

if ($doctorado == on) { 18
    $doctorado = true; 19
}else 20
    $doctorado = false; 21

if ($borrado == on) { 22
    $borrado = true; 23
}else 24
    $borrado = false; 25

$obj = new Nompersona(); 26

$obj->nombre = $nombre; 27
$obj->solapin = $solapin;27
$obj->plantilla = $plantilla;27
$obj->miembroreserva = $miembroreserva;27
$obj->master = $master;27
$obj->idprofesion = $profesion;27
$obj->idintegracion = $integracion;27
$obj->idclasificacion = $clasificacion;27
$obj->idcategoria = $categoria;27
$obj->idarea = $area;27
$obj->graduadoen = $graduadoen;27
$obj->genero = $generos;27
$obj->doctorado = $doctorado;27
$obj->correo = $correo;27
$obj->borrado = $borrado;27
$obj->origengeografico = $origengeografico;27
$obj->fechanacimiento = $fechanacimiento;27

$persona = new NompersonaModel(); 28
$persona->Insertar($obj); 29
$this->showMessage('La Persona fue insertada satisfactoriamente.'); 30
```

Figura 14: Código fuente de la funcionalidad GuardarPersonaAction ()

A continuación en la figura 12 se muestra el grafo de flujo asociado a la funcionalidad.

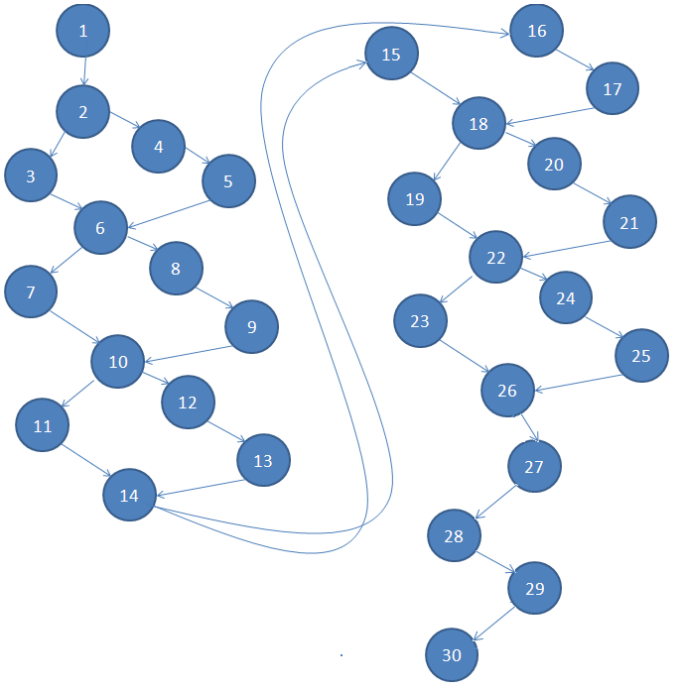

Figura 15: Grafo de flujo asociado al algoritmo GuardarPersonaAction ()

3.3.1.1: Cálculo de la complejidad ciclomática a partir de un segmento de código

La complejidad ciclomática de un código determinado puede ser calculada de tres maneras diferentes. Al calcular la complejidad del método GuardarPersonaAction () fueron utilizadas las tres formas posibles con el objetivo de verificar que el cálculo se había realizado correctamente, si los resultados obtenidos en cada una era el mismo. Las fórmulas para realizar dicho cálculo son:

1. V (G) = (A - N) + 2

V (G)= (35 -30)+2

V (G)= 7

Siendo A la cantidad total de aristas del grafo y N la cantidad de nodos.

2. V (G) = P +1

V (G) = 6+1

V (G) = 7

Siendo P la cantidad de nodos predicado (son aquellos de los cuales parten dos o más aristas)

3. V (G) = R

V (G) = 7

Siendo R la cantidad de regiones que posee el grafo.

En cada una de las fórmulas V (G) ha representado el valor del cálculo. A partir de los resultados obtenidos en cada uno, se puede determinar que la complejidad ciclomática del código analizado es 7, que a su vez es el número de caminos posibles a circular el flujo y el límite superior de casos de prueba que se le pueden aplicar a dicho código.

A continuación se muestran los caminos básicos por donde puede circular el flujo:

Camino básico # 1: 1-2-3-6-7-10-11-14-15-18-19-22-23-26-27-28-29-30

Camino básico # 2: 1-2-4-5-6-7-10-11-14-15-18-19-22-23-26-27-28-29-30

Camino básico # 3: …-6-8-9-10-11-14-15-18-19-22-23-26-27-28-29-30

Camino básico # 4: …-10-12-13-14-15-18-19-22-23-26-27-28-29-30

Camino básico # 5: …-14-16-17-18-19-22-23-26-27-28-29-30

Camino básico # 6: …-18-20-21-22-23-26-27-28-29-30

Camino básico # 7: …-22-24-25-26-27-28-29-30

Los tres puntos suspensivos (…) de los caminos del 3 en adelante indican que cualquier vía antes de la estructura de control es aceptable.

Para cada uno de los caminos obtenidos se realiza un caso de prueba. Los casos de prueba realizados son los siguientes:

3.3.1.1.1 Caso de prueba para el camino básico #1

Camino: 1-2-3-6-7-10-11-14-15-18-19-22-23-26-27-28-29-30

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se reciben los datos de las personas y las variables $plantilla, $miembroreserva, $master, $doctorado y $borrado serán verdaderas y $genero = ‘Masculino’.

Entrada: nombre=Leandro&solapin=e12108&correo=ljcapdesuner@uci.cu&fechanacimiento=05-07-2012&graduadoen=2007&genero=Masculino&origengeografico=Centro&idprofesion=1&idintegracion=1&idclasificacion=1&idcategoria=1&idarea=1&borrado=on&plantilla=on&miembroreserva=on&master=on&doctorado=on

Resultados esperados: Se espera que sea adicionada una nueva persona en el sistema.

3.3.1.1.2 Caso de prueba para el camino básico #2

Camino: 1-2-4-5-6-7-10-11-14-15-18-19-22-23-26-27-28-29-30

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se reciben los datos de las personas y las variables $miembroreserva, $master, $doctorado y $borrado serán verdaderas, $plantilla será falsa y $genero = 'Masculino'.

Entrada: nombre=Leandro&solapin=e12108&correo=ljcapdesuner@uci.cu&fechanacimiento=05-07-2012&graduadoen=2007&genero=Masculino&origengeografico=Centro&idprofesion=1&idintegracion=1&idclasificacion=1&idcategoria=1&idarea=1&borrado=on&miembroreserva=on&master=on&doctorado=on

Resultados esperados: Se espera que sea adicionada una nueva persona en el sistema.

3.3.1.1.3 Caso de prueba para el camino básico #3

Camino: …-6-8-9-10-11-14-15-18-19-22-23-26-27-28-29-30

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se reciben los datos de las personas y las variables $plantilla, $master, $doctorado y $borrado serán verdaderas, $miembroreserva será falsa y $genero = 'Masculino'.

Entrada: nombre=Leandro&solapin=e12108&correo=ljcapdesuner@uci.cu&fechanacimiento=05-07-2012&graduadoen=2007&genero=Masculino&origengeografico=Centro&idprofesion=1&idintegracion=1&idclasificacion=1&idcategoria=1&idarea=1&borrado=on& plantilla=on&master=on&doctorado=on

Resultados esperados: Se espera que sea adicionada una nueva persona en el sistema.

3.3.1.1.4 Caso de prueba para el camino básico #4

Camino: …-10-12-13-14-15-18-19-22-23-26-27-28-29-30

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se reciben los datos de las personas y las variables $plantilla, $ miembroreserva, $doctorado y $borrado serán verdaderas, $master será falsa y $genero = 'Masculino'.

Entrada: nombre=Leandro&solapin=e12108&correo=ljcapdesuner@uci.cu&fechanacimiento=05-07-2012&graduadoen=2007&genero=Masculino&origengeografico=Centro&idprofesion=1&idintegracion=1&idclasificacion=1&idcategoria=1&idarea=1&borrado=on& plantilla=on&miembroreserva=on&doctorado=on

Resultados esperados: Se espera que sea adicionada una nueva persona en el sistema.

3.3.1.1.5 Caso de prueba para el camino básico #5

Camino: …-14-16-17-18-19-22-23-26-27-28-29-30

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se reciben los datos de las personas y las variables $plantilla, $miembroreserva, $master y $borrado serán verdaderas, $doctorado será falsa y $genero = 'Masculino'.

Entrada: nombre=Leandro&solapin=e12108&correo=ljcapdesuner@uci.cu&fechanacimiento=05-07-2012&graduadoen=2007&genero=Masculino&origengeografico=Centro&idprofesion=1&idintegracion=1&idclasificacion=1&idcategoria=1&idarea=1&borrado=on& plantilla=on&master=on&master=on

Resultados esperados: Se espera que sea adicionada una nueva persona en el sistema.

3.3.1.1.6 Caso de prueba para el camino básico #6

Camino: …-18-20-21-22-23-26-27-28-29-30

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se reciben los datos de las personas y las variables $plantilla, $miembroreserva, $master y $doctorado serán verdaderas, $borrado será falsa y $genero = 'Masculino'.

Entrada: nombre=Leandro&solapin=e12108&correo=ljcapdesuner@uci.cu&fechanacimiento=05-07-2012&graduadoen=2007&genero=Masculino&origengeografico=Centro&idprofesion=1&idintegracion=1&idclasificacion=1&idcategoria=1&idarea=1&doctorado=on& plantilla=on&master=on&master=on

Resultados esperados: Se espera que sea adicionada una nueva persona en el sistema.

3.3.1.1.7 Caso de prueba para el camino básico #7

Camino: …-22-24-25-26-27-28-29-30

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se reciben los datos de las personas y las variables $plantilla, $miembroreserva, $master, $doctorado y $borrado serán verdaderas y $genero = 'Femenino'.

Entrada: nombre=Leandro&solapin=e12108&correo=ljcapdesuner@uci.cu&fechanacimiento=05-07-2012&graduadoen=2007&genero=Femenino&origengeografico=Centro&idprofesion=1&idintegracion=1&idclasificacion=1&idcategoria=1&idarea=1&borrado=on&plantilla=on&miembroreserva=on&master=on&doctorado=on

Resultados esperados: Se espera que sea adicionada una nueva persona en el sistema.

Con la aplicación de los casos de prueba expuestos anteriormente se comprobó que el flujo de trabajo de las funcionalidades es correcto, ya que se probó que cada sentencia es ejecutada al menos una vez, cumpliéndose así las condiciones de la prueba.

3.3.2 Pruebas de caja negra

Para cada uno de los requisitos funcionales fueron definidos casos de prueba, los cuales son los siguientes:

Caso de prueba para el requisito Adicionar curso

Condiciones de ejecución

- Se debe identificar y autenticar ante el sistema, además debe tener los permisos para ejecutar esta acción.
- Se debe seleccionar la opción del menú: **SGP/Nomencladores/Curso**
- Se debe haber registrado al menos un Área, profesión, Clasificación, Categoría e Integración en el sistema.

Tabla 11: Descripción del caso de prueba para el requisito Adicionar edición

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Adicionar Ediciones	El sistema permite Adicionar Ediciones	EP 1.1: Adicionar Ediciones introduciendo datos válidos.	– Se introducen los datos de la Edición correspondiente. – Se presiona el botón

			Aceptar. – Se muestra un mensaje de información. – Se presiona el botón **Aceptar**.
		EP 1.2: Adicionar Ediciones introduciendo datos inválidos.	– Se introducen los datos inválidos de la Edición – Se presiona el botón **Aceptar**. – Se muestra un mensaje de información. – Se presiona el botón **Aceptar**.
		EP 1.3: Adicionar Ediciones dejando campos vacíos.	– Se introducen los datos dejando algún campo en blanco. – Se presiona el botón **Aceptar**. – Se muestra un mensaje informando del error. – Se presiona el botón **Aceptar**.
		EP 1.4: Cancelar.	– Se introducen o no los datos del Edición – Se presiona el botón **Cancelar**.

La aplicación desarrollada fue probada por el cliente, donde fue comprobado el correcto funcionamiento de la misma a partir de los diseños de los casos de prueba.

Para más información sobre el caso de prueba para el requisito Adicionar edición, consultar el documento entregable CG-SM-DR-227. Los casos de prueba elaborados para los restantes requisitos funcionales se encuentran en los documentos entregables: CG-SM-DR-230, CG-SM-DR-229, CG-SM-DR-231, CG-SM-DR-228, CG-SM-DR-225, CG-SM-DR-349, CG-SM-DR-351, CG-SM-DR-352, CG-SM-DR-350, CG-SM-

DR-359, CG-SM-DR-362, CG-SM-DR-361, CG-SM-DR-363, CG-SM-DR-360, CG-SM-DR-226, CG-SM-DR-138, CG-SM-DR-144, CG-SM-DR-140, CG-SM-DR-141, CG-SM-DR-139.

La aplicación desarrollada fue probada por el cliente, donde fue comprobado el correcto funcionamiento de la misma a partir de los diseños de los casos de prueba. Como constancia de los satisfactorios resultados obtenidos, se emitió un acta de liberación de software.

3.4 Conclusiones

Durante el desarrollo del capítulo fueron expuestos los artefactos generados en la etapa de implementación, a partir del resultado del diseño realizado en el capítulo anterior, se realizaron los diagramas de componentes en el que se representaron las relaciones entre los mismos y los servicios que son brindados por unos y consumidos por otros y viceversa, y el diagrama de despliegue que visualiza la disposición física de los distintos recursos computacionales que componen el sistema, se presentaron las pruebas estructurales y funcionales que les fueron aplicadas, las que dieron como resultado que poseía un correcto funcionamiento, contaba con las condiciones de calidad necesarias y satisfacía las necesidades plasmadas por el cliente.

CONCLUSIONES

Se desarrolló un estudio de algunos sistemas que gestionan la información de posgrados existentes en el mundo y en Cuba, tomándolos como punto de referencia para un mayor entendimiento del problema en cuestión y aprovechando las ideas positivas de los mismos.

Se efectuó el análisis de la solución que permitió tener una visión más clara de la gestión del proceso de posgrado mediante los requisitos funcionales de este proceso, los cuales fueron identificados, especificados y validados, demostrando que estaban descritos de forma correcta, respondían a las necesidades del cliente y no existían ambigüedades entre los mismos.

Para el diseño del sistema se tuvo en cuenta el patrón arquitectónico Modelo-Vista- Controlador, facilitándose una mayor reutilización del código y organización de las funcionalidades del sistema. Se exponen las cuestiones referentes a los patrones de diseño utilizados, la utilización de los mismos permitió que el diseño sea sencillo, lo que implica una alta reutilización de las clases, baja responsabilidad y complejidad. Se aplicó la métrica tamaño operacional de clases que se obtuvo como resultado de la misma que el diseño estaba realizado de forma simple y con una calidad aceptable.

El módulo fue implementado y validado, posibilitando que se le diera solución a las necesidades plasmadas por los clientes. A través de la realización de las pruebas de caja negra se validó las funcionalidades de la aplicación obtenida a partir de los diseños de los casos de prueba y en las pruebas de caja blanca se probó que cada sentencia es ejecutada al menos una vez. La implementación de forma general es sencilla, se disminuye en gran medida la responsabilidad de las clases y las pruebas pueden ser realizadas con facilidad.

RECOMENDACIONES

Se recomienda que en el desarrollo de futuros sistemas para la gestión de posgrado sea utilizado el presente trabajo como referencia.

Continuar agregándole funcionalidades al Sistema para la administración de posgrado en el centro CEIGE, para su mejor control del proceso de posgrado en el centro.

Poner en marcha el Sistema para la administración de posgrado en el centro CEIGE y luego expandirlo a los demás centros que presenten deficiencias con el control del proceso.

TRABAJOS CITADOS

Colectivo editorial. 2008. W3C Consortium. [Online] enero 9, 2008. [Cited: diciembre 1, 2011.] http://www.w3c.es/divulgacion/guiasbreves/tecnologiasxml.

Escuela Superior Politécnica del Litoral Guayaquil. 2008. Escuela Superior Politécnica del Litoral - Guayaquil. [Online] 2008. [Cited: noviembre 26, 2011.] http://www.csi.espol.edu.ec/ui/es/content/sistema/sistema.aspx?op=toshow&id=106.

Grupo de autores. 2010. HTML.net. [Online] 2010. [Cited: noviembre 28, 2011.] http://es.html.net/tutorials/css/lesson1.php.

Guía Ubuntu. 2008. Guía Ubuntu. [Online] marzo 2008. [Cited: noviembre 26, 2011.] http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.

Mozilla Firefox. 2009. GetFirefox. [Online] 2009. [Cited: diciembre 5, 2011.] http://www.getfirefox.es/firefox-features.

The PHP Group. 2011. PHP: Hipertext Preprocesor. [Online] febrero 2011. [Cited: diciembre 3, 2011.] http://php.net/manual/es/intro-whatis.php.

Unidad de Compatibilización Integración y Desarrollo. 2009. *Proceso de Desarrollo y Gestión de Proyectos de Software.* La Habana : s.n., 2009.

Universidad Central de Venezuela. 2010. Sistema de informacion de estudios de postgrado. [Online] 2010. [Cited: enero 10, 2012.] http://www.postgrado.ucv.ve/.

Visual Paradigm. 2008. freedownloadmanager.org. [Online] 2008. [Cited: novimbre 22, 2011.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

Arias Chaves, Michael. 2005. *LA INGENIERÍA DE REQUERIMIENTOS Y SU IMPORTANCIA EN EL DESARROLLO DE PROYECTOS DE SOFTWARE.* 2005.

BONILLA, L. B. 2010. La estrategia genérica de diferenciación para la excelencia académica en un sistema de estudios de posgrado a distancia: El caso del Sistema de Estudios de Posgrado de la UNED Costa Rica. [Online] 2010. [Cited: diciembre 10, 2011.] http://www.sep.ucr.ac.cr/.

Cabrera Pereira, Leisniel Ignacio y Hernández Gómez, Maylin. 2009. *Análisis y Diseño del Módulo de Cobros y Pagos del sistema integral de gestión CEDRUX.* La Habana : s.n., 2009.

Calleja, José Manuel Ruiz. 2003. La maestría como proceso docente educativo para la formación de investigadores. [Online] noviembre 2003. [Cited: noviembre 20, 2011.]

http://www.sappiens.com/castellano/articulos.nsf/Educadores/La_maestr%C3%ADa_como_proceso_docente_educativo_para_la_formaci%C3%B3n_de_investigadores./C5216E1C37CE2A17C1256DD50075D379!opendocument.

Ciberaula. 2010. Ciberaula.com. *Una Introducción a APACHE.* [Online] 2010. [Cited: diciembre 7, 2011.] http://linux.ciberaula.com/articulo/linux_apache_intro/.

Departamento de Electrónica, Sistemas e Informática Coordinación Docente de Programación. [Online] [Cited: marzo 25, 2012.]

Doctrine. 2010. Doctrine-Project.org. [Online] 2010. [Cited: noviembre 29, 2011.] http://www.doctrine-project.org..

Equipo de Producción. 2009. *Modelo de Desarrollo Orientado a componentes del proyecto ERP-CUBA.* La Habana : s.n., 2009.

Escalona, María José y Koch, Nora. 2002. Entorno Virtual de Aprendizaje. [Online] 2002. [Cited: 4 25, 2012.] http://eva.uci.cu.

Escuela Internacional de postgrado Peru. 2011. [Online] 2011. [Cited: diciembre 10, 2011.] Peruhttp://www.esip.edu.pe/.

Hernández, A. Coello. 2002. *El paradigma cuantitativo de la investigación científica.* La Habana : Editorial universitaria, 2002.

Hernandez, Juan Darien Macias. 2011. Dragones. [Online] 2011. [Cited: diciembre 3, 2011.]

IDE. 2011. Wikipedia. [Online] 2011. [Cited: noviembre 30, 2011.] http://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado.

Inc, Sencha. 2011. Sencha. [Online] 2011. http://www.sencha.com/learn/extjs/?4x.

Machado, Daymel. 2010. *Implementación del módulo Distribución de productos.* La Habana : s.n., 2010.

MES. 2006. REGLAMENTO DE LA EDUCACION DE POSGRADO DE LA REPUBLICA DE CUBA. [Online] 2006. [Cited: noviembre 20, 2011.] http://www.uvs.sld.cu/archivos/reglamento-de-posgrado-mes-cuba.pdf..

NED University of Engineering & Technology. 2012. NED UNIVERSITY OF ENGINEERING & TECHNOLOGY. [Online] 2012. [Cited: enero 12, 2012.] http://www.neduet.edu.pk:8080/postgrad/Login.jsp.

Patxi. 2007. Eslomas.com [En línea] 3 de julio de 2007. *Frameworks de Zend para el desarrollo de aplicaciones PHP.* [Online] julio 3, 2007. [Cited: noviembre 26, 2011.] http://www.eslomas.com/index.php/archives/2007/07/03/framework-de-zend-para-el-desarrollo-de-aplicaciones-php/.

Pérez Olmos, Yoisy. 2009. *Técnicas y herramientas de la ingeniería de requisitos adecuadas para simuladores virtuales.* La Habana : s.n., 2009.

Pressman, Roger S. 2006. *Ingeniería del Software Un enfoque práctico.* 2006.

ProgramacionWeb. 2009. ProgramacionWeb.net. [Online] 2009. [Cited: noviembre 25, 2011.] http://www.programacionweb.net/cursos/curso.php?num=2.

Rivera, Javier Fernandez. 2011. [Online] 2011. [Cited: abril 20, 2012.] http://aurea.es/wp-content/uploads/modelodedatos.pdf..

Romero, Oscar Casasola. 2010. Programacion en Castellano. [Online] agosto 27, 2010. [Cited: noviembre 23, 2011.] http://www.programacion.com/articulo/introduccion_a_uml_181.

Scribd. 2010. Scribd . [Online] 2010. [Cited: noviembre 22, 2011.] http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.

Sommerville. 2005. *Requerimientos.* 2005.

Toledo, Álbaro. 2010. UPtoDOWN. [Online] 2010. [Cited: diciembre 5, 2011.] http://postgresql.uptodown.com/.

Unidad de Compatibilización Integración y Desarrollo. 2009. *Proceso de Desarrollo y Gestión de Proyectos de Software.* Ciudad de la Habana : s.n., 2009.

Universidad de Castilla La Mancha. 2011-2012. Departamento de Sistemas Informáticos. [Online] 2011-2012. [Cited: 04 15, 2012.] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.