Universidad de las Ciencias Informáticas.

Facultad 1.

Título: Sistema de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1.

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

Autores: Yisel González Fernández.
Yanitza Benitez Campo.

Tutores: Ing. Victor Luis Borroto Alvariño.
Ing. Julio Cardoso Ventura.

Ciudad de la Habana, junio 2010.
"Año 52 de la Revolución".

AGRADECIMIENTOS

Yisel González Fernández:

- *A nuestro Comandante Fidel por tener la idea maravillosa de crear esta universidad de excelencia.*
- *A la Revolución por darme la oportunidad de estudiar en ella.*
- *A la UCI por forjarme como profesional y persona del bien.*
- *A los profesores que de una forma u otra estuvieron ahí aportando su conocimiento.*
- *A mis maravillosos tutores que sin ellos este trabajo no se hubiese hecho con la calidad necesaria, por estar ahí cada vez que los necesitaba.*
- *A los amigos de ayer, hoy y siempre que hemos compartido juntos los buenos y malos momentos de la vida en especial a ese piquete de insoportables que nunca olvidaré (Lorian, Anelkis, Yilo, Yanis, Yanitza, Aliuska, Dai, el Z, Lily, Kendry, Boloy, Rey), a mi grupo 1108 y 1401.*
- *A la Yani, mi compañera de tesis por aguantarme en estos años y, más aún, en el transcurso de esta tesis.*
- *A Villiva por regañarme tanto, por guiarme, por estar siempre dispuesto a ayudarme en todo lo que he necesitado y sobre todo por quererme tanto. Gracias por el apoyo incondicional que siempre me has brindado.*
- *A mi madrina Cary, a Manolo y a Regla, por preocuparse tanto por mí y por cómo me iban los estudios, por complacerme siempre en todos mis antojos. Gracias por tratarme con ese cariño que solo ustedes saben hacerlo.*
- *A Yeni, mi prima yo te quiero muchísimo y lo sabes, cómo no agradecerte a ti si tú eres lo máximo, gracias por aguantar todas mis malcriadeces y por brindarme tu apoyo en todo momento, de todas tú eras la única que confiaste en mí y mira no te defraude, hoy tu prima es ingeniera.*
- *A mi tío Pedro y a mi tía Niurka (que aunque no te diga tía quiero que sepas que te quiero mucho), gracias por sacarme de mis apuros y por quererme tanto, por ayudarme siempre en todo lo que he necesitado de ustedes.*
- *A mis suegros que han sido como mis padres, gracias por los consejos y por sacarme esa sonrisa, gracias por darme fuerza cuando yo pensaba que ya todo estaba perdido.*

AGRADECIMIENTOS

- *A Aliu, gracias michi por estar conmigo ahí en las buenas y en las malas, sufriendo mis problemas junto conmigo y por tus consejos.*
- *A mi abuela Tomasa por ayudarme desde allá arriba, aunque no estés físicamente aquí con nosotros yo sé que tu también te sientes orgullosa de tu nieta y a mi abuela Gloria gracias por estar ahí cada vez que te necesito, gracias por tus consejos y sobre todo por pedir siempre cosas buenas para mí.*
- *A todas las personas que no confiaron en mí, les doy gracias también porque ellos me enseñaron hacer fuerte y día tras día me dieron fuerza para demostrarle que yo si puedo dar más de lo que ellos esperan.*

A cuatro personas que el hecho de que las deje para último no significa que sean las que menos tengo que agradecerle, todo lo contrario:

- *A mi pochito lindo, mi amor, mi amigo, mi esposo, mi alma gemela, mi compañero de guerrilla, de tantos momentos lindos, pero tantos momentos amargos, tú sabes que sin ti este sueño no hubiese sido realidad, gracias por tu paciencia, por tu ayuda incondicional a cualquier hora, por tus consejos tan sabios, por guiarme en todo momento, por darme fuerza cuando pensaba que todo estaba perdido, por enseñarme a ver la vida de otra manera, gracias por quererme y comprenderme tanto, gracias por aceptarme con mis defectos y con mis virtudes, me siento afortunada de tenerte a mi lado, sin dudas te amo con la vida.*
- *A mi papá, papi agradecerte a ti es algo que en esta tesis no cabe, necesitaría 3 ó 4 más para agradecerte todo lo que día tras día haces por mí, eres el mejor padre del mundo, tú me has dado tantos buenos consejos y me has enseñado tantas cosas, siempre buscando todas las maneras posibles de hacerme el estudio fácil y la vida agradable. Eres mi ejemplo y orgullo en la vida profesional, nunca has dejado de cuidarme y preocuparte por mis cosas, gracias por depositar en mí tu confianza, nunca te defraudaré, esta tesis es para que te sientas orgulloso de tu hija que tanto te quiere, por esto han sido todos estos años de estudio. Papi gracias por tu crianza, gracias a ella hoy soy la persona que soy.*

AGRADECIMIENTOS

- *A mi mamá, mi amiga de todos los años, gracias mami por tanto amor, por tus consejos, por tu dedicación día tras día, por enseñarme a ver que el arcoíris tiene más de 7 colores y que por muy difíciles que se vean las cosas todo tiene solución, gracias por tus regaños cuando lo necesitaba, tú has sido mi impulso, mi fuerza en todo momento y mi eterno apoyo, la mujer más hermosa e increíble del mundo, a ti te agradeceré siempre. "Mamita al fin lo logramos tu hijita ya es Ingeniera".*
- *A mi hermanita, mi princesita linda, mi niña malcriada, aunque siempre estemos fajadas solo quiero que sepas que te quiero con la vida, que a ti también te agradezco todo lo que haces por mí, por tus consejos cada vez que yo llamaba llorando y solo el teléfono era la vía para desahogarme y con tus voz calma mis oídos oían "no llores mi hermana que eso es bobería", y tratabas de sacarme esa sonrisa que estaba perdida, es mucho, pero mucho lo que yo te quiero.*

De corazón gracias a todos.

AGRADECIMIENTOS

Yanitza Benitez Campo:

- *A mi mamá por ser mi fuerza, mi luz, mi guía, mi apoyo, por darme todo el amor y el cariño que siempre he necesitado, por ser la personita que me ha cuidado durante estos 23 años luchando como nadie podría imaginárselo, luchando contra todo y contra todos para darme todo lo que siempre me hizo falta. A ti mi mamuchi te dedico este sueño que es de las dos. Gracias por todo lo lindo que has hecho por tu hijita durante todo este tiempo, gracias también por haber sabido burlarte de la soledad que nos invadió desde el día que vine a estudiar tan lejos, por todo esto y por las otras cosas maravillosas que solo tú sabes hacer, las cuales serían imposibles de mencionar en este momento. Mil veces gracias.*
- *A mi puchuti, mi brují, mi esposo por quererme tanto, cuidarme y aguantarme durante estos 4 años, gracias por todo el apoyo y el amor que me has dado, por comprenderme, por aconsejarme, por guiarme cuando pensaba que no había más salidas para mis problemas, por ser incondicional, por estar ahí cuando más te necesitaba, gracias mi amor te quiero con todo mi corazoncito.*
- *A mi papá, a ti papi por haberme hecho entender que errar es de humanos, por haberme demostrado que de verdad eres mi padre y que quieres estar ahí para mí cuando te necesite, gracias por cuidar a mi hermana durante todo este tiempo, por ser la personita en la que ella confía, gracias por todas estas cosas maravillosas que estás haciendo.*
- *A mi segundo padre Israel, por cuidar, respetar y darle todo el cariño que mí mamita necesitaba durante todo este tiempo, gracias por todo, quiero que sepas que para mi eres como un padre.*
- *A mis suegros (Ramón, Anita, Alba) por haberme acogido como a una hija más y por haberme cuidado en todo este tiempo.*
- *A mi ángel, quien más que mi tía Marcia, mi tiita eres un ángel en verdad porque esas cosas maravillosas que tú haces una persona no las podría hacer, gracias por todos tus consejos, te quiero mucho.*
- *A mis cuñis (Raydel, Michel, Joaco) por aguantar todos mis berrinches y por ser tan locos.*

AGRADECIMIENTOS

- *A mis tutores por guiarme, ayudarme y apoyarme durante todo el trabajo de mi tesis, gracias.*
- *A mi compañera de tesis Yisel que me ayudó, se estresó, me peleó y luchó junto conmigo durante todo el transcurso de la tesis para lograr el sueño de las dos (ser ingenieras y demostrarle a todos que una estudiante de politécnico y una de la ESPA pueden llegar a ser ingenieras), Gracias.*
- *A mis amigos que compartieron conmigo durante todo este tiempo en la universidad, a mis súper grupos el 1401 y 1504, a las chicas del piquete (Lily, Yisel, Aliuska, Daimaris, Yanis).*
- *A mis compañeras de cuarto Yaité, Malena, Darianna y Lily por compartir juntas todo este tiempo de convivencia.*

DEDICATORIA

A mis padres por ser mi máxima fuente de inspiración, y por brindarme siempre su apoyo.

A mi hermana por aguantar mis malcriadeces.

A mi esposo por tanto amor y dedicación.

Yisel González Fernández.

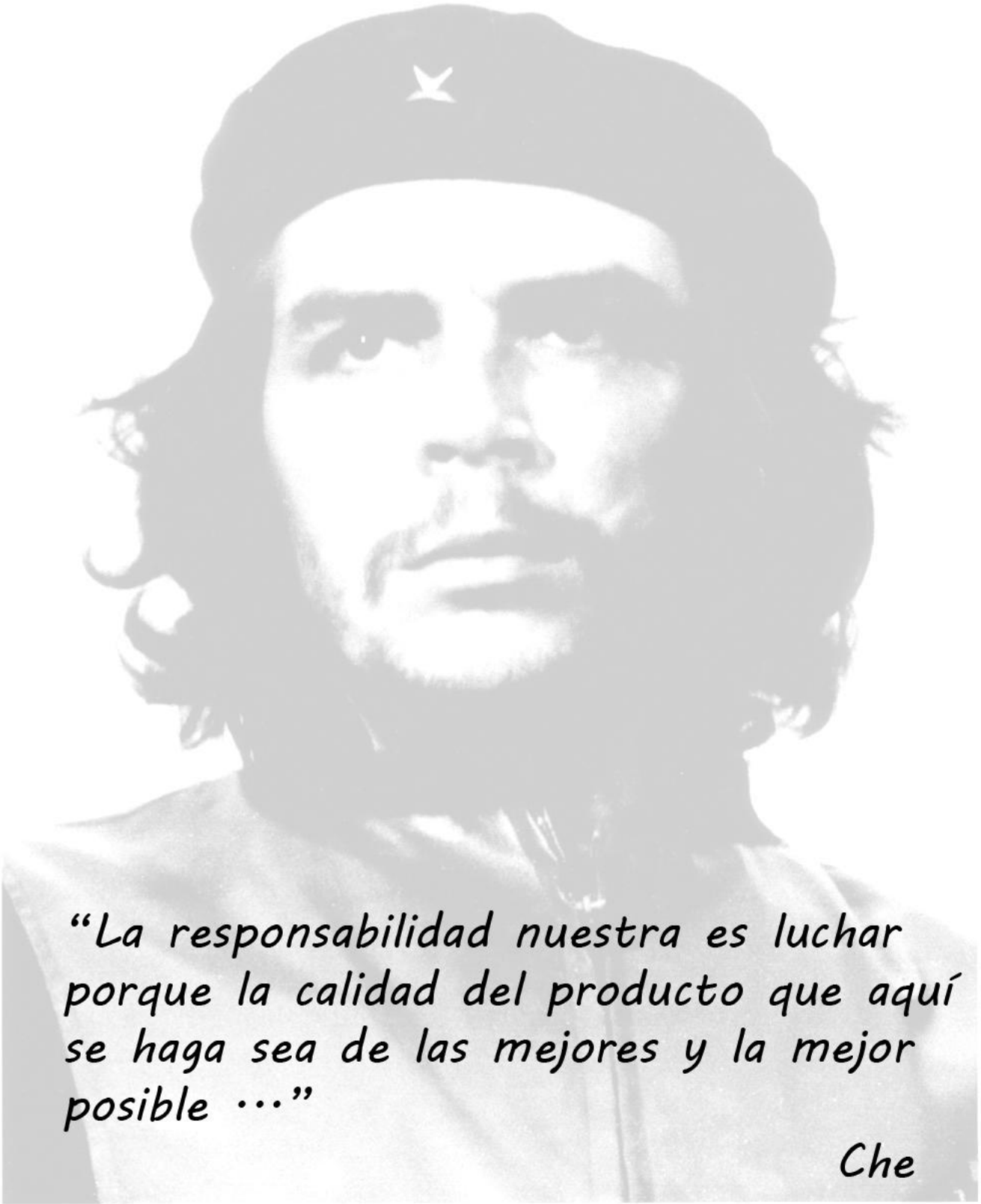
DEDICATORIA

A mi madre por ser mi rayito de esperanza, mi guía y por darme todo su apoyo.

A mi tía Marcia por aconsejarme en todo este tiempo y guiarme por el buen camino.

A mi brujito (Ray) por todo su amorcito, su tiempo y dedicación.

Yanitza Benitez Campo.


"La responsabilidad nuestra es luchar porque la calidad del producto que aquí se haga sea de las mejores y la mejor posible ..."
Che

Resumen.

El título del presente trabajo es Sistema de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1. Teniendo como necesidad desarrollar un sistema para el apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1 de la Universidad de las Ciencias Informáticas (UCI) y que a su vez agilice la gestión de los procesos involucrados en estos.

En el mismo se realizó un estudio detallado de diferentes sistemas que gestionan recursos humanos tanto en Cuba, a nivel Mundial y en la Universidad de las Ciencias Informáticas, tomando las mejores soluciones, para adaptarlas al Sistema de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes de la facultad 1. Se realizó un análisis de las herramientas, técnicas, metodologías actuales que serán usadas para construir la solución del problema planteado, además de la fundamentación del uso de la metodología escogida para realizar el trabajo.

Se realizó una descripción de los principales procesos que se identificaron para automatizar el sistema. Se confeccionaron los prototipos de interfaz de la aplicación acorde a las necesidades del sistema. Finalmente, se aplicaron una serie de pruebas al sistema, obteniendo un producto con la calidad requerida.

Con la implementación de este sistema informático se dará un vuelco en el proceso de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1 de la UCI, ya que este permite una mejor gestión y control de los mismos, la agilización de tiempo, el trabajo óptimo y la fácil interacción del usuario con el sistema.

Palabras Claves: Gestión, Procesos, Sistema, Automatizar.

Índice de contenido.

Índice de figuras.

Índice de tablas.

Introducción.

El cruel e injusto bloqueo económico impuesto por el gobierno de los Estados Unidos afecta todas las esferas de Cuba, es por ello que el área de la informática y las comunicaciones no está ajena a estos efectos.

La historia del bloqueo a Cuba en materia de informática es casi tan antigua como el triunfo de la Revolución Cubana y se remonta a finales de 1960, cuando el gobierno norteamericano no autorizó el embarque de dos computadoras IBM-1401, compradas por Cuba a la IBM y quedaron varadas para siempre en el puerto de Nueva York, pese a que para ese entonces todavía la IBM tenía oficialmente una filial en La Habana.

Así se inició una escalada de agresiones y limitaciones sin precedentes en cuanto a la negación de acceso a la tecnología a un país, que llevó a que Cuba, por ejemplo, solo pudiera conectarse a Internet en 1996 y en virtud de una aprobación especial del Departamento de Estado.

Cuba pese a lo anterior mencionado se ha visto inmersa en un profundo y novedoso proceso de transformaciones educacionales y sociales como programas de la Batalla de Ideas, a partir del cual se emprendieron y se emprenden nuevos programas destinados a elevar el nivel cultural de la población y su calidad de vida. De esta manera, surge así la Universidad de las Ciencias Informáticas (UCI) con el doble objetivo de informatizar el país y desarrollar la industria del *software* (sistema, aplicación, en español) para contribuir al desarrollo económico del mismo. Es por ello que desde su creación como proyecto ha volcado sus esfuerzos en asistir con soluciones prácticas los problemas sociales tanto dentro como fuera de la misma, por lo que ha informatizado muchos de los servicios que esta posee y brinda la comunidad universitaria.

Existe un crecimiento notable en el ingreso anual de estudiantes al Movimiento de Alumnos Ayudantes para contribuir a la enseñanza y la formación profesional de estudiantes de años inferiores, todo este proceso se ha convertido en una de las disciplinas con mayor grado de prioridad dentro de esta importante institución,

tratándose así de un reto para los que tienen la responsabilidad de llevarla a la práctica en las distintas facultades de la Universidad.

Para un mejor funcionamiento en el movimiento de alumnos ayudantes en la Facultad 1, se llevan a cabo una serie de procesos relacionados con el apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en dicha facultad, pero en una investigación se detectaron una serie de problemas que hace de estas actividades un trabajo engorroso. Primeramente, el proceso de solicitud y la aceptación del alumno se hacen manualmente lo que ha traído como consecuencia que el trabajo sea más extenso y la aceptación sea mucho más lenta, además cabe la posibilidad que toda la documentación que genera cada uno de los procesos que se llevan a cabo para la selección, seguimiento y control de los alumnos ayudantes se pierda debido a que donde se encuentra corra riesgos de seguridad, también existe la posibilidad que la compartimentación de la información este limitada para el personal que lo necesite en el momento deseado, las orientaciones y el seguimiento de las tareas orientadas al alumno ayudante por parte de sus tutores se realizan vía *email (correo electrónico, en español),* provocando así que muchas veces no llegue las especificaciones de las tareas al destinatario y se desconozca el desarrollo y cumplimiento de las mismas, lo que genera una ***Situación Problémica***.

Todo lo anteriormente planteado origina el siguiente ***Problema Científico:***

¿Cómo agilizar la gestión de los procesos de ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1 de la UCI?

En consecuencia, el ***Objeto de Estudio*** de dicho trabajo son los procesos referentes al ingreso, seguimiento y control del movimiento de alumnos ayudantes y más específicamente su ***Campo de Acción*** son los procesos relacionados con la gestión de las actividades de ingreso, seguimiento y control de este movimiento en la facultad antes mencionada.

Para dar respuesta a la Situación Problémica planteada anteriormente se trazó el siguiente ***Objetivo General****,* desarrollar un sistema para el apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1 de la UCI y que a su vez agilice la gestión de los procesos involucrados en estos,

utilizando una metodología y lenguajes que corresponda con las nuevas concepciones de la informatización del país y de la Universidad.

Partiendo de la necesidad que actualmente la facultad presenta con la gestión de los procesos llevados a cabo en el ingreso, control y seguimiento del Movimiento de Alumnos Ayudantes, se trazó la siguiente ***Idea a defender***, implementación de un sistema para el apoyo al ingreso, seguimiento y control del Movimiento de Alumnos Ayudantes en la facultad 1 de la UCI, contribuyendo así a agilizar la gestión de las actividades de dicho movimiento.

Para lograr un eficiente desarrollo de la investigación y darle cumplimiento al objetivo general trazado se pone en práctica las siguientes ***Tareas Científicas:***

1. Describir el estado del arte de la problemática a tratar.
2. Confeccionar el marco teórico – conceptual de la investigación a partir de una búsqueda y revisión bibliográfica.
3. Valorar las técnicas de programación, lenguajes y *framework* propuestos por la Dirección de Informatización de la Universidad
4. Generar los artefactos del flujo de diseño del sistema a desarrollar.
5. Implementar el sistema para agilizar los procesos de ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1 de la UCI.
6. Diseñar los casos de uso de pruebas que validen y certifiquen el correcto funcionamiento del sistema desarrollado.

Para dar cumplimiento a estas tareas se emplean métodos empíricos y teóricos de la investigación científica. Dentro de los ***Métodos Empíricos*** se realizan una serie de ***Entrevistas*** a las personas involucradas en los procesos de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes, para tener un mejor conocimiento sobre los requisitos que debe tener el sistema. Se emplea como método la ***Observación Científica***, la cual permite la evaluación centrada en los procesos relacionados con los objetivos propuestos.

Como ***Métodos Teóricos*** se utiliza el método ***Analítico-Sintético*** pues se hace una breve investigación y estudio detallado de documentos que generan los procesos de

las actividades del movimiento de alumnos ayudantes para así identificar el problema existente en la facultad, sus causas y lograr un profundo diseño.
Por medio del Método ***Histórico-Lógico*** se estudia la evolución y desarrollo de la gestión de actividades en el movimiento de alumnos ayudantes a nivel de facultad y la UCI, además de estudiar los cambios que surgieron y dieron lugar a dicha situación.

La estructura del documento está constituida de la siguiente manera:
Capítulo 1. Fundamentación Teórica: Este capítulo contiene una base teórica para entender el problema planteado. Se hace un estudio de las tendencias actuales de diferentes *softwares* que gestionen Recursos Humanos a nivel mundial, Cuba y la UCI. Así como los diferentes lenguajes, tecnologías y metodologías a usar.
Capítulo 2. Características del Sistema: Expone las funcionalidades o requisitos del sistema que son objeto de solución de acuerdo con los servicios que se desean implementar.
Capítulo 3. Diseño del Sistema: Se muestran los diagramas de clases del diseño y sus descripciones, junto con los correspondientes diagramas de interacción, así como los patrones a usar.
Capítulo 4. Implementación y Prueba: Se muestra el diagrama de componente y la matriz de integración externa, se realizan una serie de pruebas para validar la implementación.

Capítulo 1: Fundamentación teórica.

1.Introducción.

En este capítulo se realizó un estudio acerca de los aspectos teóricos para la elaboración y concepción del Trabajo de Diploma. Se analizaron y se definieron los conceptos para comprender el dominio del problema. Se realizó una investigación del lenguaje de modelado a usar, además la metodología de desarrollo para guiar el avance del sistema y por último se analizaron las herramientas, metodologías, *framework* y los lenguajes seleccionados para llevar a cabo la elaboración del sistema.

1.1 ¿Qué son los Alumnos Ayudantes?

Los alumnos ayudantes son aquellos estudiantes de alto aprovechamiento docente, previamente seleccionados en las carreras, tanto en las sedes centrales como en las sedes universitarias, que se distinguen por mostrar ritmos de asimilación más rápidos, aptitudes favorables para el aprendizaje de alguna o algunas disciplinas del plan de estudio y para la investigación científica o el trabajo de desarrollo técnico.

1.2 Tendencias actuales de los Sistemas de Gestión de los Recursos Humanos.

Las entidades eligen el *software* de recursos humanos que más se adapte a sus necesidades. Cuando se habla de profesionalizar la gestión de los recursos humanos, se asocia muchas veces este concepto a contar con un *software* capaz de mejorar el desempeño del área.

El mercado informático cuenta con una diversidad de soluciones que parece crecer año tras año. Cada organización detecta las necesidades propias y adquiere el *software* que más se adapte a sus necesidades, que posea el soporte apropiado para cada negocio.

Los *software* de gestión de recursos humanos están orientados a satisfacer diferentes necesidades de las empresas en aras de gestionar al potencial humano

dentro de las mismas, sus principales y más comunes funcionalidades son la planificación, selección del personal, detectar las necesidades de capacitación, administrar los cursos de capacitación de la entidad, entre otros aspectos importantes.

1.3 Tendencias históricas de los Sistemas de Gestión de los Recursos Humanos:

1.3.1 Sistemas de Gestión de los Recursos Humanos en el mundo.

A nivel mundial se han desarrollado varios sistemas de Gestión de los Recursos Humanos, a continuación se presentan ejemplos de algunos sistemas:

Cezanne Software: es un proveedor de soluciones avanzadas del capital humano que ayuda a las organizaciones a mejorar, gestionar, recompensar y retener su recurso más importante: las personas.

El objetivo de la compañía es ofrecer unas soluciones flexibles y centradas en las personas que mejoren la realización cotidiana de las estrategias de gestión del Capital Humano.

Sus productos incluyen aplicaciones para la gestión del rendimiento del empleado, planes de sucesión y carreras, formación y desarrollo, gestión de las personas, selección, análisis salarial, planificación retributiva, revisión salarial, encuestas y diseño de organigramas.

Sus aplicaciones de Recursos Humanos (RRHH) están diseñadas específicamente para su utilización en entorno web. Se puede acceder fácilmente tanto con una conexión Internet segura como por medio de la red interna y ofrecen un nivel avanzado de seguridad, alcance y facilidad necesaria para su uso global en la compañía.

Cezanne Software cuenta en la actualidad con más de 750 clientes en todo el mundo. Un ejemplo de ello son: Adverto, Almirall, Atento, Banesco, Boehringer Ingelheim, Borsa Italiana, British American Tobacco, British Telecom, Carrefour,

CAF, Coremain, EDP, HC Energía, Ibercaja, Illy, Intercontinental Hotels Group, Invercaixa, Jose de Mello, Grupo Balzola, Luxotica, NEC, Nextel, RAI, Sacyr Vallehermoso, Schweppes, Securitas Direct, UCI, Unicoop y Vodafone (Cezanne Software, 2010).

PlanningPME: es un agradable entorno con las funciones y los recursos necesarios para que permite planificar tareas, proyectos y todo tipo de actividades de un modo sencillo y muy visual. Básicamente, PlanningPME permite programar tareas en forma de bloques de colores identificando parámetros como, por ejemplo, el nombre de la tarea, la fecha de inicio y de entrega, el tipo de tarea o su periodicidad (única, diaria, semanal, mensual, etc.). Además de tareas, PlanningPME permite gestionar recursos, es decir, se puede crear un equipo de personas, asignar tareas e incluso compartir toda la planificación en red o mediante publicaciones en formato HTML. Sus funciones adicionales permiten gestionar clientes, importar/exportar tareas a Excel o MS Project, lanzar búsquedas por multicriterios, generar estadísticas e imprimir memorias. En definitiva, PlanningPME es una interesante solución para planificar tareas, gestionar proyectos y mejorar, en consecuencia, el rendimiento y la eficiencia del equipo (PlanningPME, 2010)**.**

HRmgr: es una herramienta de gestión de recursos humanos. Está diseñada para ser simple de usar aunque no lo suficientemente poderosa para realizar el trabajo que se requiere. Los datos pueden ser almacenados en la base de datos cubriendo un amplio rango de información de un empleado. Además de su nombre, dirección, fecha de nacimiento, supervisor, título de trabajo, HRmgr rastrea la información relativa a la asistencia, logros, beneficios, revisiones, sueldos, información de inmigración y del departamento de trabajo y contactos en caso de emergencia. Además de la información en listados, se pueden imprimir reportes para una fácil referencia y guardar copias en sus archivos. Debido a la gran cantidad de personal e información privada que será almacenada, este programa fue equipado con protección por contraseñas. HRmgr también provee *Backup (*copia de seguridad, en español*)*, restauración, importación, exportación y utilidades de base de datos. Este

programa debe funcionar bien para el almacenamiento de la información de recursos humanos y para las necesidades de organizaciones de cualquier tipo, ya sean pequeños o de medianos negocios. La versión sin registrar está limitada a cinco registros de empleados. No lográndose con la gran mayoría de las características que presenta, que su uso sea completamente accesible por personas que requieran de la información (HRmgr, 2010).

HRCorporate. Versión 4.2: es una solución integral de recursos humanos que permite atraer, desarrollar y retener el talento humano que una organización requiere para lograr mejores resultados de negocio. Este *software* fue diseñado para grandes organizaciones, cuenta con 14 módulos completamente integrados sobre una plataforma 100 % Web. Cada módulo presenta funcionalidades especializadas por procesos de recursos humanos que se enlazan al resto de la solución, que se implantan y configuran de acuerdo con los requerimientos de cada cliente. Esta aplicación es creada para una institución en específico, aunque posee muy buenas funcionalidades no tienen la flexibilidad de adaptarse a las exigencias propuesta por la facultad 1(HRCorporate.Version 4.2, 2010).

1.3.2 Sistemas de Gestión de los Recursos Humanos en Cuba.

En Cuba también se han desarrollado varios sistemas de Gestión de los Recursos Humanos, a continuación se presentan ejemplos de algunos sistemas:

ASSETS NS: es un Sistema de Gestión Integral estándar y parametrizado que permite el control de los procesos de Compras, Ventas, Producción, Taller, Inventario, Finanzas, Contabilidad, Presupuesto, Activos Fijos, Útiles y Herramientas y Recursos Humanos. Dispone, además, de métodos novedosos para administración y planificación de inventarios. Es un sistema flexible, amigable, con ayuda en línea que puede ser instalado en una microcomputadora o sobre varias, funcionando en ambiente multiusuario incluidas estaciones remotas. Así mismo,

proporciona opciones de seguridad que le permiten limitar el acceso a los diferentes procesos del sistema de acuerdo con el perfil de cada usuario.

En el mismo se facilita el uso de la parametrización para adaptarse a las exigencias de cada entidad en particular, garantizando que sus reportes tengan la forma y el contenido que el usuario les defina. Este sistema presenta diferentes módulos, dentro de estos se encuentra el módulo de Recursos Humanos (Versión 3.1 Access 97) el cual está concebido para calcular las nóminas y controlar los recursos laborales de una entidad.

Desde Recursos Humanos se pueden controlar íntegramente los recursos laborales: empleados, estructura organizativa de la entidad y plantilla. Siempre que se introducen altas, bajas y otros movimientos, se actualiza automáticamente el registro de empleados (Catálogo de Empleados) y se generan los reportes correspondientes (con el formato oficial). Es posible modificar plantillas, introducir cambios en la estructura organizativa, crear nuevos cargos y realizar conversiones de plazas (ASSETS NS, 2010).

Fastos: Sistema para la gestión de los recursos humanos conformado por módulos que ejecuta los procesos y operaciones de: registro de empleados, control de la plantilla, control de asistencia, informes y modelos.

Sistema diseñado para generar y controlar toda la información correspondiente a la nómina en una organización. Es capaz de asimilar una amplia gama de sistemas de pago, pues se configura de acuerdo con la forma de cada cliente.

- Registro de los empleados: se guardan los datos de los empleados, así como informaciones referentes a los reportes de vacaciones, certificados médicos, licencias, resolución.
- Control de la plantilla: Permite establecer la estructura organizativa de las plazas de la entidad.
- Control de asistencia: lo cual incorpora el control de claves de asistencias, turnos de trabajos, horarios, tarjeta de asistencia e incidencias de cada empleados. Permite acoplar relojes (RTA 600) para actualizar la información de la tarjeta de asistencia de forma automática.

- Informes y modelos: Permite obtener un total de 56 informes, por ejemplo cierre del periodo, análisis de fondo de tiempo, estadísticos, entre otros, y 11 modelos.
- Control de la capacitación respecto a: Acciones de Capacitación, Estudios Realizados, Cursos, Eventos, Experiencia docente, Publicaciones, Conocimientos, Idiomas extranjeros, Otros Aspectos, Plan de desarrollo, Informes.
- Control de la información de los cuadros: Se establece el registro de los cuadros, dirigentes y reserva, referente a evaluaciones, inspecciones, sanciones, necesidades de capacitación, entre otros (Fastos, 2010).

RH-CITMA versión 1.0: pretende obtener las ventajas de la identificación de los procesos de la empresa para la gestión de algunas Actividades de Recursos Humanos.

Para lograr lo anterior ha sido diseñado como sistema compuesto por diferentes módulos relacionados entre sí dentro de los que se encuentran:

- Procesos
- Selección del Personal
- Sistema de Sugerencias
- Inventario de Personal

Este *software* ha sido diseñado desde el primer momento para convertirse (en versiones posteriores) en un *software* para la gestión de la Calidad o incluso un sistema de Gestión Integrado que incluya la calidad, la seguridad y salud del trabajo y el medio ambiente, las buenas prácticas de manufactura y el HACCP para el caso de empresas productoras de alimentos, incluyendo siempre las facilidades de la gestión de los recursos humanos presente en esta versión.

En el diseño y confección del mismo se emplearon los *software* Visual Studio - Visual Basic v 6.0 Service pack 6, Base de datos Access 2003, Shalom Help Maker versión v 0.5.2

El *software* RH-CITMA tiene como ventajas:

- Integra la mayoría de las actividades de la gestión de recursos humanos basado en un enfoque de procesos.
- Facilita la identificación y control de todos los procesos de la empresa.

El *software* RH-CITMA tiene como desventaja:

- Requiere de la persona que lo utilizará un dominio de enfoques de calidad y gestión por procesos.
- No incluye la parte mejoramiento de Procesos.
- Su Funcionamiento es más fácil en una empresa que posea un sistema de Gestión de la Calidad y los procesos identificados (RHCITMA, 2010).

1.3.3 Sistemas de Gestión de los Recursos Humanos en la UCI.

En la universidad se cuenta con un Sistema para el control de la información y evaluación de los profesores en los departamentos docentes de la facultad 8. Este sistema permite llevar el control de la información de los profesores y medios básicos que pertenecen a cada departamento, además brinda a los jefes de departamento la posibilidad de evaluar a los profesores según las actividades realizadas y del cumplimiento del plan de trabajo anual.

Este sistema permite además:

- Gestionar actividades del profesor
- Generar reportes
- Asignar evaluación
- Gestionar plan individual del profesor
- Gestionar usuarios
- Crear cuenta de usuario
- Crear plan de trabajo
- Realizar evaluaciones mensual y trimestral del profesor
- Realizar evaluación anual del profesor

En cuanto a un sistema que gestione los procesos referentes al movimiento de alumnos ayudantes, en la Universidad no se ha realizado ningún *software* que permita gestionar dicha información, a pesar de que existen aplicaciones que tienen

semejanzas con el *software* a modelar. Luego de haber realizado el análisis de los sistemas automatizados se puede concluir que los mismos poseen un gran número de características que le permiten solucionar ágilmente un amplio espectro de problemas, sin embargo, se detecta la no existencia de un *software* que agrupe la gestión de un sistema de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1, a partir de lo cual se ratifica la necesidad del desarrollo del mismo. Al analizar los ejemplos anteriores se decide tomar algunos aspectos positivos que ayuden a la confección de la aplicación.

1.4 Metodología de Desarrollo de Software.

Una metodología de desarrollo de *software* es un conjunto de técnicas, herramientas, procedimientos y soporte documental que permite a los desarrolladores definir los elementos necesarios para la construcción de un nuevo producto de *software*.

Mediante la metodología de desarrollo de *software* se van indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. Además, detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla (1).

1.4.1 Proceso de Desarrollo y Gestión de Proyectos de Software.

Un proceso de desarrollo de *software* tiene como objetivo la producción eficiente de un producto de *software* que satisfaga los requisitos de un cliente con una planificación y una estimación de recursos predecibles. Los elementos de un proceso y sus relaciones deben responder Quién debe hacer Qué, Cuándo y Cómo. Esto se logra modelando las interacciones y relaciones que suceden entre las personas (roles), las actividades que estas desarrollan y los artefactos que se crean o actualizan durante el proceso.

El modelo de desarrollo de *software* propuesto describe la secuencia de actividades de alto nivel para la construcción y desarrollo de soluciones. Se logra con la

combinación entre los modelos basado en Componentes, el Iterativo y el Incremental. Se emplearán las técnicas de prototipado, si son requeridas, para los requerimientos del usuario de los que no existe una visión clara por parte de estos, con el objetivo de desarrollar una definición mejorada de los requisitos del usuario para el sistema.

Desarrollo iterativo e incremental: Es un enfoque en el que el ciclo de vida está compuesto por iteraciones, estos son pequeños procesos compuestos de varias actividades cuyo objetivo es entregar una parte del sistema parcialmente completo, probado, integrado y estable. Todo el *software* es integrado en cada entrega de cada iteración hasta obtener el producto de *software* completo en la última iteración. En cada iteración se obtiene como resultado un incremento.

Desarrollo basado en componentes: Nos lleva a alcanzar un mayor nivel de reutilización de *software*, aún en contextos distintos de aquellos para los que fue diseñado. Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes ensamblados. Cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema. Dado que un componente puede ser construido y luego mejorado continuamente, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo.

1.4.2 Notación de Modelado de Procesos de Negocio.

La Notación para el Modelado de Procesos de Negocio (*Business Process Management Notation)* es una notación para la representación de procesos de negocio, creada por BPMI (*Business Process Management Initiative*) con el objetivo de proveer una notación comprensible por todos los interesados en el proceso de negocio, desde los expertos de negocio hasta los desarrolladores técnicos.

BPMN es un estándar nuevo para el flujo de procesos del modelado de negocios y los servicios web, define un Diagrama de Procesos de Negocio basado en la técnica de diagramado de flujos que ajusta modelos gráficos de operación de procesos de negocio.

La Notación para el Modelado de Procesos de Negocio se aplicará para la modelación de negocio del sistema en la elaboración del mapa de procesos dado el enfoque que este representa hacia los procesos de negocio con el objetivo de definir cómo se desarrolla la gestión de los procesos que forman parte de la solución así como representar el flujo de información de los mismos.

Con la creciente certeza de que la agilización, la optimización y la gestión de los procesos son las claves del éxito, los desarrolladores de *software* han comenzado a usar la Notación de Modelado de Procesos de Negocio (*Business Process Modeling Notation*, BPMN por sus siglas en inglés) la cual proporciona un lenguaje común para que las partes involucradas puedan comunicar los procesos de forma clara, completa y eficiente. Siendo esta metodología de tipo ágil y orientada a objeto.

BPMN está diseñado para cubrir varios tipos de modelado y permite la creación, tanto de segmentos de proceso, como procesos de negocio de comienzo a fin, y en diferentes niveles de representatividad. Además, se usa para comunicar una amplia variedad de información a diferentes audiencias.

La notación de modelado de procesos de negocio tiene como objetivos primarios:

- Proveer una notación entendible para cualquiera desde el analista del negocio, el desarrollador técnico y hasta la persona propia del negocio.
- Crear un puente estandarizado entre el diseño de procesos de negocio y su implementación.
- Asegurar que los lenguajes para la ejecución de procesos de negocio puedan ser visualizados con una notación común.

Lo que arroja a las ventajas siguientes:

- Define la notación y semántica de un BPD (*Business Process Diagram, diagrama de proceso de negocio en español.*)
- Provee la capacidad de entender los procedimientos internos en una notación gráfica y da a las organizaciones la habilidad de comunicarlos de una manera estándar.

- Mejora las capacidades de las notaciones de proceso de negocio tradicional para manejar inherentemente los conceptos de procesos de negocio *business to business* (comunicaciones de comercio electrónico, en español).

El ciclo de diseño, análisis, ejecución y gestión de los procesos requiere que diferentes partes interactúen con los procesos, en diferentes momentos y de diferentes formas. BPMN ha sido creado para proporcionar una misma notación que sea comprensible tanto para los analistas como para los profesionales de las tecnologías de la información. Con BPMN los usuarios pueden crear diagramas usando una interfaz intuitiva que gestiona automáticamente muchas de las tareas de dibujo (2).

1.5 Lenguaje de Modelado.

1.5.1 UML.

El Lenguaje Unificado de Modelado, en sus siglas en inglés (UML), es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de *software*. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

UML es un lenguaje de propósito general para el modelado orientado a objetos. UML es también un lenguaje de modelamiento visual que permite una abstracción del sistema y sus componentes (UML, 2010).

Ventajas de UML:

- Propone un estándar para el intercambio técnico de modelos y diseños Código reutilizable.
- Es una consolidación de muchas de las notaciones y conceptos más usados orientados a objeto.
- Facilita el ahorro de tiempo en el desarrollo del *software*.
- Se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real.

1.6 Herramienta de Modelado.

1.6.1 Visual Paradigm.

Visual Paradigm para UML 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 crear diagramas de clases, generar código desde diagramas y documentación.

Visual Paradigm se integra al Entorno de Desarrollo Integrado (*Integrated Development Environment*, IDE por sus siglas en inglés) de Eclipse. Está diseñado para desarrollar *software* con Programación Orientada a Objetos, reduce la duración del ciclo de desarrollo brindando ayuda tanto a arquitectos, analistas, diseñadores como a desarrolladores. Busca también automatizar tareas tediosas que pueden distraer a los desarrolladores.

Ventajas de Visual Paradigm:

- Es multiplataforma
- Permite el modelamiento de los Requisitos
- Permite el Modelado de Procesos de Negocio
- Permite el modelamiento de bases de datos (3).

1.7 Lenguajes de Programación.

1.7.1 PHP.

Es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica.

Ventajas de PHP:

- Muy sencillo de aprender.
- Permite las técnicas de Programación Orientada a Objetos (POO).

- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad.
- Lee y manipula datos desde diversas fuentes, incluyendo datos que pueden ingresar los usuarios desde formularios HTML.
- El precio para utilizar PHP es cero, por lo que es gratuito y se puede descargar desde www.php.net
- Ofrece una solución simple y universal para las paginaciones dinámicas de la web de fácil programación.
- Es multiplataforma. Funciona en toda máquina que sea capaz de compilar su código, entre ellas diversos sistemas operativos para PC y diversos Unix. El código escrito en PHP en cualquier plataforma funciona exactamente igual en otra.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- Completamente orientado a la web (PHP, 2010).

1.7.2 JavaScript.

JavaScript es un lenguaje de programación utilizado para crear pequeños programas encargados de realizar acciones dentro del ámbito de una página web. Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado. Permite ejecutar instrucciones como respuesta a las acciones del usuario, se puede crear páginas interactivas con programas como: calculadoras, agendas, o tablas de cálculo. Es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Además, JavaScript pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente (4).

Ventajas de JavaScript:

- Es un código “interpretado” por el cliente.
- Es un lenguaje abierto.

- Es un código orientado a objetos.
- Es un código integrado a HTML.

1.7.3 HTML.

El lenguaje de marcas HTML (del inglés: *HyperText Marckup Language*, en español Lenguaje de Marcado de Hipertexto), se utiliza para definir las páginas web y permite describir hipertexto, es decir, texto presentado de forma estructurada y agradable, con enlaces que conducen a otros documentos o fuentes de información relacionadas, y con inserciones multimedia (gráficos, fotos, sonido.etc). Básicamente se trata de un conjunto de etiquetas que sirven para definir el texto y otros elementos que compondrán una página web, es decir, se utiliza para crear las páginas web y les indica a los navegadores cómo deben mostrar el contenido.

El lenguaje HTML contiene dos partes:

- Contenido, que es el texto que se verá en la pantalla de un ordenador
- Etiquetas y atributos que estructuran el texto de la página web en encabezados, párrafos, listas, enlaces, etc. y normalmente no se muestra en pantalla (5).

1.7.4 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. Siendo su función principal describir datos y no mostrarlos como es el caso de HTML (6).

1.8 Herramientas de programación.

1.8.1 NetBeans.

NetBeans es una aplicación de código abierto diseñada para el desarrollo de aplicaciones fácilmente portable entre las distintas plataformas, haciendo uso de la tecnología Java. Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones Web, control de versiones, colaboración entre varias personas, creación de aplicaciones compatibles con teléfonos móviles y funcionalidades ampliables mediante la instalación de packs.

El proyecto NetBeans consta de un entorno de desarrollo integrado (IDE) de código abierto y una plataforma de aplicaciones que permiten a los desarrolladores crear rápidamente la Web, empresa, escritorio y móviles aplicaciones, utilizando la plataforma *Java*, así como PHP, JavaScript y Ajax, Ruby on Rails, C / C++ y Groovy. Cuenta con el apoyo de una vibrante comunidad de desarrolladores y ofrece una variada selección de terceros *plug-ins*, además es entorno de desarrollo integrado para Windows, Mac, Linux y Solaris.

Ventajas de NetBeans:

- Código abierto, patrocinado por Sun.
- La integración de múltiples herramientas y protocolos proporciona razones para la migración.
- Facilidad de uso durante todo el ciclo de desarrollo.
- Maneja la complejidad de la arquitectura orientada al servicio (SOA).
- El soporte al modelado mejora la productividad del desarrollador.
- Facilidad de uso con Windows y *software* libre.
- Perfecto entorno de desarrollo (7).

1.8.2 Spket.

Spket es un plugin para Eclipse y Aptana que provee un conjunto de utilidades para la edición de JavaScript, sobre todo para la edición de clases que extienden el *framework* JavaScript ExtJS o que usan la librería. Spket provee un editor de código

JavaScript muy parecido al editor Java de Eclipse, es decir, incluye autocompletado de código, resaltado de texto, muestra de errores, entre otros.

1.9 Tecnologías.

1.9.1 Servidor Web Apache.

Es un servidor web gratuito, potente y que ofrece un servicio estable y sencillo de mantener y configurar. Es indiscutiblemente uno de los mayores logros del *Software* Libre. El servidor Apache se desarrolla dentro del proyecto HTTP *Server* (http), de la *Apache Software Foundation* (Fundación de software Apache, en español).

Ventajas de Apache:

- Es multiplataforma, aunque idealmente está preparado para funcionar bajo Linux.
- Muy sencillo de configurar.
- Es *Open-Source* (código abierto, en español).
- Muy útil para proveedores de Servicios de Internet que requieran miles de sitios pequeños con páginas estáticas.
- Amplias librerías de PHP *y* Perl a disposición de los programadores.
- Posee diversos módulos que permiten incorporarle nuevas funcionalidades, estos son muy simples de utilizar.
- Es capaz de utilizar lenguajes como PHP, TCL, *Pitón*, entre otros (Apache, 2010).

1.9.2 Gestor de Base Datos PostgresSQL.

PostgresSQL es un servidor de base de datos relacional orientado a objetos de software libre. Como muchos otros proyectos, es de código abierto, el desarrollo de PostgresSQL no es manejado por una

sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo.

Algunas de sus principales características son la alta concurrencia. Mediante un sistema denominado MVCC (Acceso concurrente multiversión, por sus siglas en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros

accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último que se actualizó.

Ventajas de PostgresSQL:

- Corre en casi todos los principales sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows, etc.
- Soporte nativo para los lenguajes más populares: PHP, C, C++, Perl, Python, etc.
- Gran escalabilidad. Es capaz de ajustarse al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, permitiéndole soportar una mayor cantidad de peticiones simultáneas de manera correcta (en algunos casos, se dice que ha llegado a soportar el triple de carga de lo que soporta MySQL).
- 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, permite la creación de tipos propios de datos.
- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.
- Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de base de datos de alto nivel, tales como Oracle (8).

1.9.3 Administrador de Base Datos PGAdmin 3.

PgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgresSQL, siendo la más completa y popular con licencia *Open Source*. 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 PostgresSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de PostgresSQL como Pervasive Postgres, EnterpriseDB, Mammoth Replicator y SRA PowerGres.

PgAdmin III está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas.

El interfaz gráfico soporta todas las características de PostgresSQL y facilita enormemente la administración. La aplicación también 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 o Unix Domain Sockets (en plataformas *nix), y puede encriptarse mediante SSL para mayor seguridad (9).

1.9.4 Gestor de Configuración Tortoise SVN.

TortoiseSVN es un cliente gratuito de código abierto para el sistema de control de versiones *Subversion* (controlador de versiones, en español). TortoiseSVN maneja ficheros y directorios a lo largo del tiempo. Los ficheros se almacenan en un repositorio central. Esto permite que pueda recuperar versiones antiguas de sus ficheros y examinar la historia de cuándo y cómo cambiaron sus datos, y quién hizo el cambio.

Ventajas de *TortoiseSVN*:

- Integración con el shell de *Windows*: TortoiseSVN se integra perfectamente en el *shell* de *Windows* (por ejemplo, el explorador). Esto significa que puede seguir trabajando con las herramientas que ya conoce.
- Iconos sobreimpresionados: El estado de cada carpeta y fichero versionado se indica por pequeños iconos sobreimpresionados. De esta forma, puede ver fácilmente el estado en el que se encuentra su copia de trabajo.
- Fácil acceso a los comandos de Subversion: Todos los comandos de Subversion están disponibles desde el menú contextual del explorador. TortoiseSVN añade su propio submenú allí (10).

1.9.5 Navegador Web Firefox.

Mozilla Firefox 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.

Ventajas de Firefox:

- Es multiplataforma.
- Cuenta con una protección *antiphishing* (detección de robo de contraseña o tarjeta de crédito, en español), *antimalware* (detección y eliminación de virus, en español) e integración con el antivirus.
- Permite la navegación por pestañas.
- Bloqueador de ventanas emergentes (Mozilla Firefox, 2010).

1.10 Framework.

1.10.1 Zend Framework.

Zend Framework se trata de un *framework* para desarrollo de aplicaciones Web y servicios Web con PHP. Brinda soluciones para construir sitios web modernos, robustos y seguros. Además, es de código abierto y trabaja con PHP 5.

Ventajas de *Zend Framework*:

- Trabaja en 3 capas, usando el Modelo Vista Controlador.
- Cuenta con módulos para manejar archivos en formato de documento portátil, canales de sindicación de noticias y servicios web.
- Incluye objetos de las diferentes bases de datos, por lo que es extremadamente simple para consultar la base de datos.
- Completa documentación y pruebas de alta calidad.
- Robustas clases para autenticación y filtrado de entrada.
- Clientes para servicios web (11).

1.10.2 Zend_Ext Framework.

Es un *framework* de código abierto, diseñado para PHP 5 o superior. Se deriva de Zend Framework por lo que cumple con todas sus características. Posee un controlador vertical para el control de las acciones realizadas por las vistas hacia el controlador, un motor de reglas para las validaciones en el servidor, y se le agregó el IOC para la integración entre los módulos o componentes. Tiene incorporado el ORM Doctrine Framework para el trabajo en la capa de abstracción de la base de datos y el ExtJS Framework para el desarrollo de las vistas.

1.10.3 Doctrine Framework.

Framework Doctrine es un potente y completo sistema de mapas de relaciones de objetos (*Object Relational Mapper*, ORM por sus siglas en inglés) para PHP 5.2 o superior con una base de datos con capas de abstracción incorporada.

Entre muchas otras cosas tienes la posibilidad de exportar una base de datos existente a sus clases correspondientes y también a la inversa, es decir, convertir clases a tablas de una base de datos. Su principal ventaja radica en poder acceder a la base de datos utilizando la programación orientada a objetos debido a que doctrine utiliza el patrón Active Record para manejar la base de datos, tiene su propio lenguaje de consultas y trabaja de manera rápida y eficiente. Es fácilmente integrado a los principales *frameworks* de desarrollo utilizados actualmente, por lo que se propone su uso (12).

1.10.4 ExtJS.

ExtJS es una librería construida con JavaScript que proporciona una interfaz cuya potencia radica en la rica colección de componentes para el diseño de interfaces del lado del cliente.

Tiene incluidos la mayoría de los controles de los formularios Web incluyendo tablas para mostrar datos y elementos semejantes a la programación *desktop (escritorio, en español)* como los formularios, paneles, barras de herramientas, menús y muchos otros. Dentro de su librería de componentes incluye componentes para el manejo de datos, lectura de XML, lectura de datos JSON e implementaciones basadas en AJAX. Presenta el uso de JavaScript con una programación orientada a objetos.

Ventajas de ExtJS:

- Tiene un alto *performance* (rendimiento, en español), *UI customizables (interfaz de usuario personalizables, en español)*.
- Un modelo de Componentes muy bien diseñado y extensible.
- Una API muy fácil de usar.
- Licencias comerciales y *Open Source*.
- Compatibilidad de *Browsers (navegadores, en español).*

- Código reutilizable.
- Independiente o adaptables a *framework* diferentes (13).

1.11 IOC.

Inversión de control (Inversion of Control, IOC por sus siglas en inglés,) es un concepto junto a unas técnicas de programación en las que el flujo de ejecución de un programa se invierte respecto a los métodos de programación tradicionales, en los que la interacción se expresa de forma imperativa haciendo llamadas a procedimientos o funciones. Tradicionalmente el programador especifica la secuencia de decisiones y procedimientos que pueden darse durante el ciclo de vida de un programa mediante llamadas a funciones. En su lugar, en la inversión de control se especifican respuestas deseadas a sucesos o solicitudes de datos concretas, dejando que algún tipo de entidad o arquitectura externa lleve a cabo las acciones de control que se requieran en el orden necesario y para el conjunto de sucesos que tengan que ocurrir (14).

1.12 Estilo arquitectónico MVC

El Modelo Vista Controlador (*Model View Controller*, MVC por sus siglas en inglés) es un estilo arquitectónico que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. El estilo arquitectónico MVC se ve frecuentemente en aplicaciones web, donde la vista es la interface de usuario y el código es el que provee de datos dinámicos a la página; el modelo es el Sistema de Gestión de Base de Datos y la lógica de negocio; y el controlador es el responsable de recibir los eventos de entrada desde la vista.

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.
- Define las reglas de negocio (la funcionalidad del sistema).
- Lleva un registro de las vistas y controladores del sistema.
- Si estamos 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, estas acciones pueden suponer peticiones al modelo o a las vistas.

Las Vistas son responsables de:

- Recibir datos del modelo mediante el controlador y las muestras al usuario.
- Tienen un registro de su controlador asociado.
- Pueden dar el servicio de "Actualización ()", para que sea invocado por el controlador o por el modelo cuando es un modelo activo (MVC, 2010).

1.13 Conclusiones parciales del capítulo 1.

Como conclusión al capítulo 1 se puede decir que se realizó un estudio detallado de los conceptos necesarios para el entendimiento del *software*, se tomaron decisiones importantes para la realización del sistema, tales como la elección de los lenguajes de programación, como por ejemplo del lado del servidor utilizaremos PHP 5.2.4 y del lado del cliente JavaScript, como Gestor de Base de Datos utilizaremos Postgres 8.4.1, como servidor web utilizaremos Apache 2.2.9, como Administrador de Base Datos será PgAdmin 3, el Navegador Web a utilizar será Firefox, los *framework* a utilizar serán Doctrine, Zend Framework, Zend_Ext, Extjs.

Capítulo 2: Características del Sistema.

2. Introducción.

Para realizar un *software* con una buena calidad se debe tener dominio de todos los procesos relacionados en el mismo, se deben definir bien claras sus características para así desarrollar un sistema con la mejor calidad posible. En el siguiente capítulo se muestran detalladamente los procesos relacionados con el negocio. Se da una propuesta del sistema que se pretende obtener. Se analizan la captura de requisitos del *software*, se especifican los requisitos funcionales y no funcionales. Se visualiza y se describe el modelo de negocio, esto permite una buena comprensión del mismo.

2.1 Objeto de estudio.

2.1.1 Problema y Situación Problémica.

La facultad 1 de la Universidad de las Ciencias Informáticas necesita para un mejor funcionamiento y organización un sistema que sea capaz de llevar a cabo el apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes de dicha facultad, ya que todo el proceso de solicitud y aceptación del alumno se hacen manualmente lo que ha traído como consecuencia que el trabajo sea más extenso y la aceptación sea mucho más lenta, además, puede que toda la documentación que genera cada uno de los procesos que se llevan a cabo para la selección, seguimiento y control de los alumnos ayudantes se pierda debido a que donde se encuentra corre riesgos de seguridad, otra deficiencia es que todas las orientaciones y el seguimiento de las tareas orientadas al alumno ayudante por parte de la cátedra se realizan vía *email (correo electrónico, en español)* provocando así que muchas veces no llegue las especificaciones de las tareas al destinatario y se desconozca el desarrollo y cumplimiento de las mismas.

2.1.2 Objeto de Automatización.

En la facultad 1 se maneja toda la información referente al ingreso, seguimiento y control del movimiento de alumnos ayudantes. Gestionar todos estos datos se hace difícil y tedioso ya que se realiza manualmente aumentando así la posibilidad de cometer errores, de tener información repetida y también se desperdicia tiempo por el hecho de no tener la misma centralizada. La realización de un sistema para gestionar los procesos que genera todas estas actividades, que recopile la información que se maneja en los mismos y que se automaticen los procesos descritos posteriormente, es la propuesta que tiene este trabajo.

Los procesos a automatizar serán:

- Ratificar alumnos ayudantes.
- Solicitar alumnos ayudantes.
- Aprobar al estudiante.
- Asignar tutor a los alumnos ayudantes.
- Asignar tareas a los alumnos ayudantes.

2.2 Información que se maneja.

La información que se maneja en este trabajo es la RESOLUCIÓN No. 210 del año 2007, más específico el Capítulo 5 y sus artículos 214, 215, 216, 217, 218, 219, 220, 221 y 222 respectivamente. Además, se maneja el documento donde se explica todo lo referente al proceso de ratificación y captación de alumnos ayudantes.

2.3 Propuesta del Sistema.

En la presente investigación, con el propósito de darle cumplimiento al problema científico planteado, de acuerdo con los estudios realizados y atendiendo a las necesidades del movimiento de alumnos ayudantes se propone la implementación de una aplicación web haciendo uso de los *framework* Doctrine Framework, Zend_Ext Framework, Zend Framework utilizado para el desarrollo de aplicaciones web en PHP y el *framework* Extjs utilizado para presentación en JavaScript. Todos estos integrados en un marco de trabajo desarrollado en el centro UCID (Unidad de compatibilización, integración y desarrollo de proyectos para la defensa).

El sistema permitirá realizar solicitudes para ser alumno ayudante a los estudiantes y a los profesores que deseen que un estudiante sea alumno ayudante de su asignatura, además de cancelar dicha solicitud. Una vez realizada la solicitud el sistema permitirá gestionar la aceptación al alumno, si este cumple los requisitos se acepta la solicitud, de lo contrario se aplaza o se rechaza la misma, dándole a conocer al estudiante el motivo del rechazo o aplazamiento de la solicitud.

Si la solicitud es aceptada se le asignará un tutor al estudiante, un grupo o grupos docentes donde el mismo impartirá las clases. Permitirá además asignarles tareas a los estudiantes así como darle seguimiento y una evaluación sistemáticamente a la misma. Contendrá conjuntamente un sistema de alertas para notificarle al usuario cuando una tarea está a punto de finalizar según la fecha de culminación de la misma. Conjuntamente con todas estas actividades se podrá mostrar y realizar búsquedas de datos que el usuario solicite.

La confidencialidad de la información estará gestionada a través del sistema de seguridad integral que propone el marco de trabajo del centro UCID, y la integridad de la información será protegida mediante las salvas periódicas que se le realicen al servidor de base de datos asegurando así que no se pierda la misma en caso de desastre o una mala manipulación.

2.4 Modelo de negocio.

El modelo del negocio es uno de los modelos útiles previos al desarrollo de un *software*. Su propósito es lograr una mejor comprensión del problema que el *software* tiene que resolver. Permite comprender la estructura y la dinámica de la organización en la cual se va a implantar un sistema además de los problemas actuales de dicha organización e identificar las mejoras potenciales. Asegura que los consumidores, usuarios finales y desarrolladores tengan un entendimiento común de la organización, derivando los requerimientos del sistema que va a soportar la misma (15).

2.4.1 Modelo de los procesos del Negocio.

La modelación de proceso 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 actualmente en la entidad y la relación que existe entre estos. De esta forma, se van determinando necesidades operacionales, así como restricciones que presenta la entidad, obteniéndose finalmente un entendimiento del negocio para dar paso a la fase inicial del sistema, lográndose a través de los objetivos siguientes:

- 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 (16).

2.4.2 Procesos de Negocio.

Una parte de vital importancia dentro del modelado del negocio es la definición de los procesos del negocio y su descripción, ya que permite tener una información más detallada de los mismos.

En la presente investigación se definieron cinco procesos principales, éstos son:

- Ratificar alumnos ayudantes.
- Solicitar alumnos ayudantes.
- Aprobar al estudiante.
- Asignar tutor a los alumnos ayudantes.
- Asignar tareas a los alumnos ayudantes.

2.4.3 Mapa de Procesos.

El mapa de procesos es un artefacto que muestra la interacción que se establece entre los procesos de negocio fundamentales que han sido identificados con sus respectivas entradas y salidas (Mapa de proceso, 2010).

2.4.3.1 Descripción del Mapa de Proceso del Negocio.

El proceso de ingreso, seguimiento y control del movimiento de alumnos ayudantes se basa en un conjunto de procedimientos que se realizan con el objetivo de obtener un buen resultado en la captación de estudiantes para el movimiento. Todo este proceso se realiza en 5 etapas:

1. Ratificar alumnos ayudantes.

2. Solicitar alumnos ayudantes.
3. Aprobar al estudiante.
4. Asignar tutor a los alumnos ayudantes.
5. Asignar tareas a los alumnos ayudantes.

- **Ratificar alumnos ayudantes.**

El proceso de ratificación a los alumnos ayudantes pertenecientes al movimiento se realiza mediante una evaluación por parte del tutor y un integrante del consejo FEU de la facultad 1 de acuerdo con los requisitos que estos deben cumplir, si esta evaluación que corresponde a su trayectoria de años anteriores es positiva se ratifica como alumno ayudante, sino deja de pertenecer a dicho movimiento y se actualiza el listado donde se encuentran registrados.

- **Solicitar alumnos ayudantes.**

Para realizar el proceso de solicitud de alumnos ayudantes la FEU lanza una convocatoria de acuerdo con la cantidad de estudiantes que necesiten por cada asignatura para pertenecer al movimiento de alumnos ayudantes, la solicitud puede ser realizada por parte de todos aquellos estudiantes que deseen pertenecer al movimiento de alumnos ayudantes de la facultad 1 y por parte de los profesores que tengan alguna propuesta de algún estudiante que pueda pertenecer a dicho movimiento.

- **Aprobar al estudiante.**

La secretaria docente le solicita al encargado de la FEU las solicitudes efectuadas por parte de los profesores como de los estudiantes. Luego realiza una evaluación detallada a cada estudiante de acuerdo con su integralidad, su índice académico, la nota de la asignatura en la que el estudiante desea impartir clase y si tiene exámenes de premios, entregándole al mismo los resultados de la evaluación. Luego la FEU se reúne con la Dirección de la Facultad y discuten los resultados de la evaluación entregada por parte de la secretaria docente y si el estudiante cumple con todos los requisitos propuestos es aceptado y pasa a pertenecer al Movimiento de Alumnos Ayudantes de la Facultad 1, sino es rechazado. Luego se confecciona el listado de los alumnos ayudantes se envía a la Dirección de Economía y se

realiza la presentación de los resultados al Consejo Docente de la Facultad donde todos lo que pertenecen a dicho consejo conocen a los nuevos alumnos ayudantes y se envía a cada estudiante aceptado la notificación de su ingreso.

- **Asignar tutor a los alumnos ayudantes.**

A cada alumno ayudante el jefe de departamento le asigna un tutor en dependencia de la asignatura que este se encuentre impartiendo.

- **Asignar tareas a los alumnos ayudantes.**

Una vez asignado el tutor al alumno ayudante, el tutor se encarga de darle seguimiento a los alumnos ayudantes que se encuentren asesorando, además de asignarles tareas, las cuales el tutor se las evaluará y las mismas deberán cumplirse de acuerdo con el cronograma de tarea establecido.

A continuación se presenta el mapa de procesos:

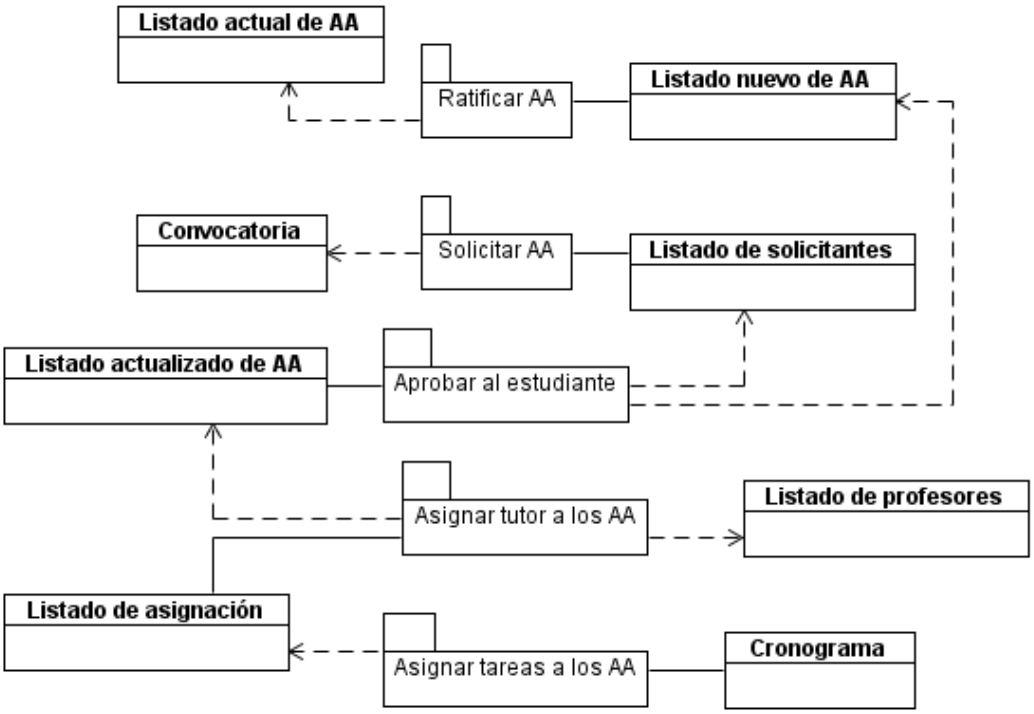


Figura 1: Mapa de procesos.

A continuación se muestran los diagramas de proceso de los cinco procesos más importantes:

- Ratificar alumnos ayudantes.

- Solicitar alumnos ayudantes.
- Aprobar al estudiante.
- Asignar tutor a los alumnos ayudantes.
- Asignar tareas a los alumnos ayudantes.

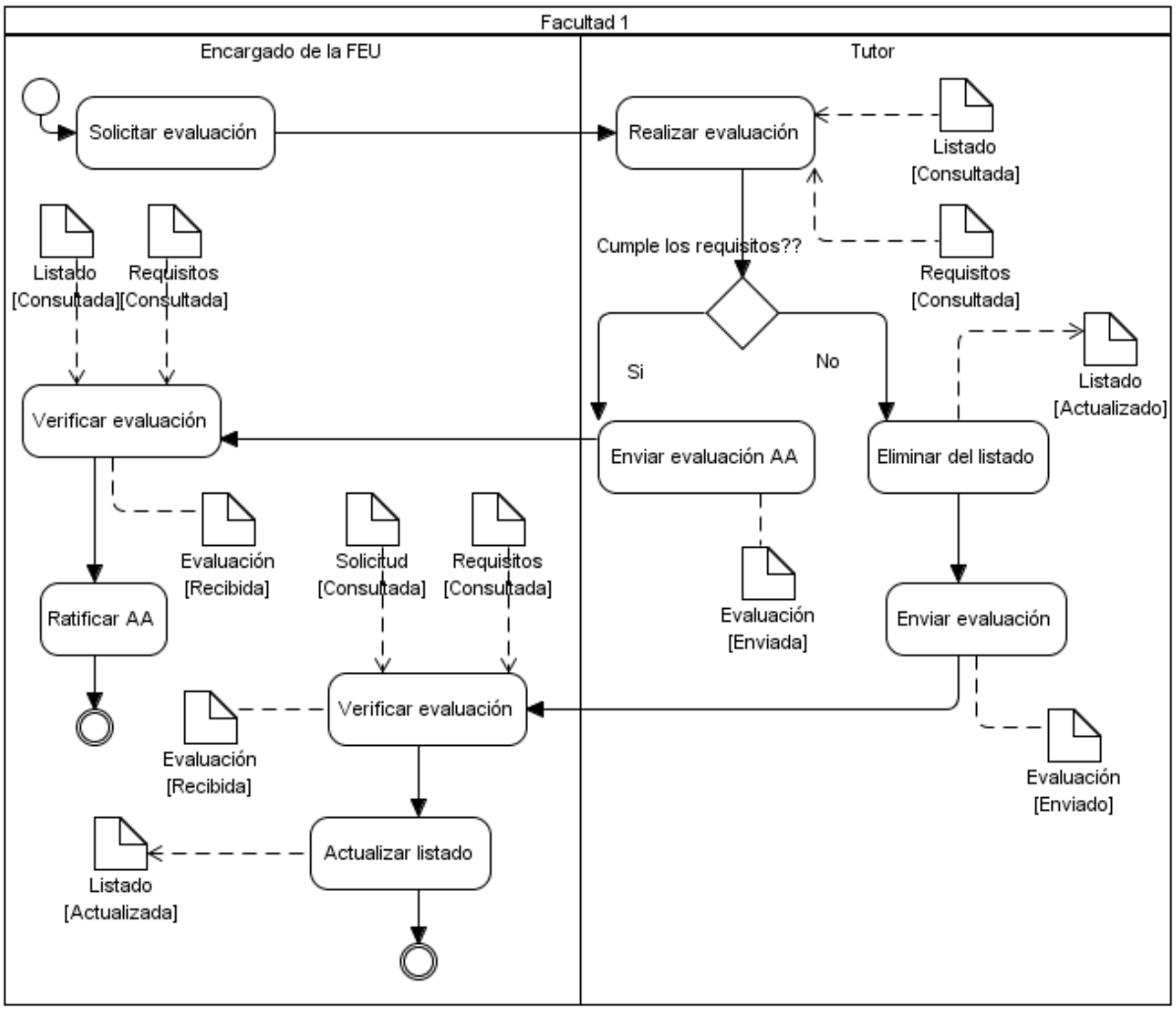


Figura 2: Diagrama de proceso: Ratificar alumnos ayudantes.

Ver descripción del proceso ratificar alumnos ayudantes en el Anexo 1.

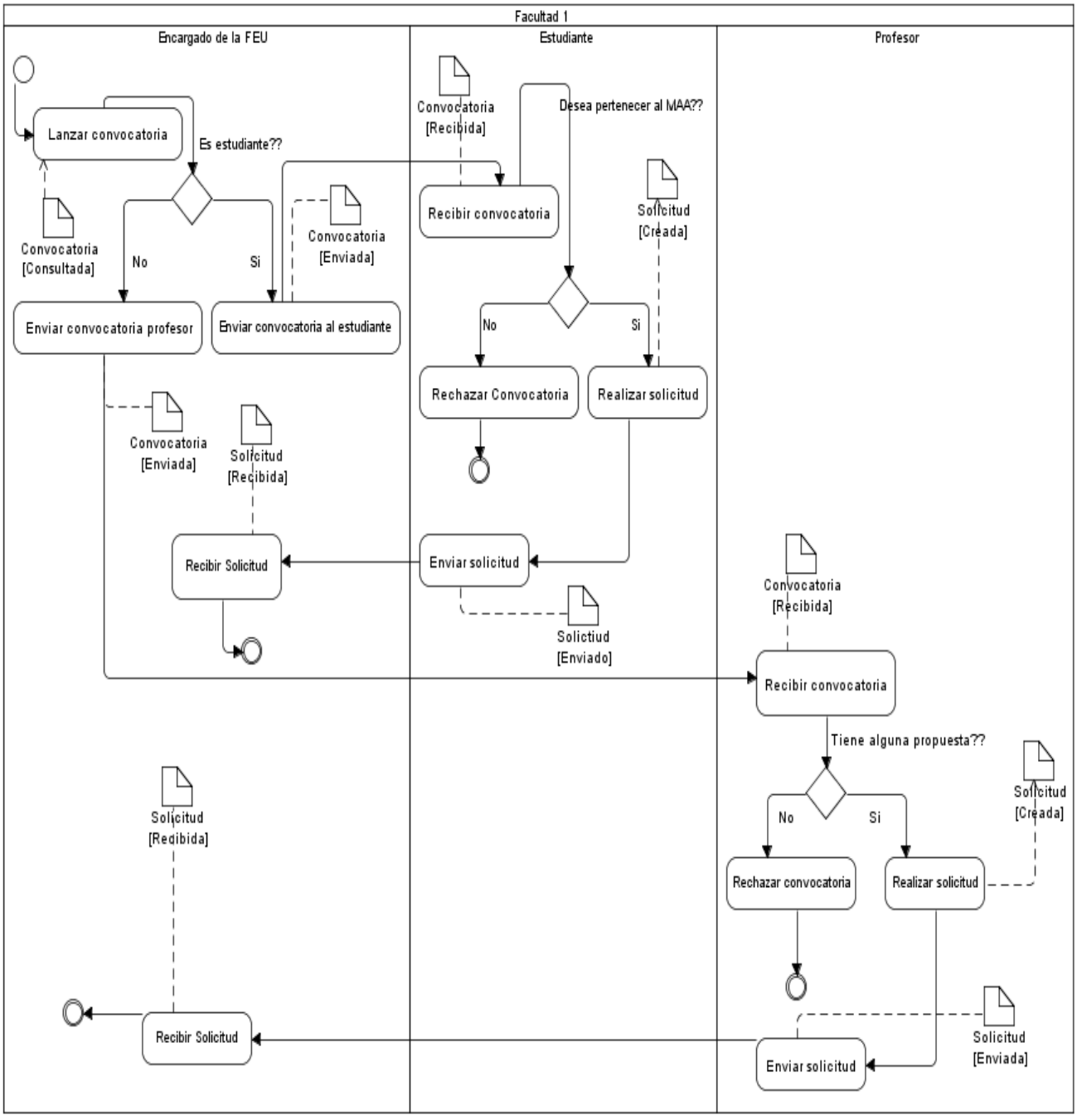


Figura 3: Diagrama de proceso: Solicitud de alumnos ayudantes.

Ver descripción del proceso solicitud de alumnos ayudantes en el Anexo 2.

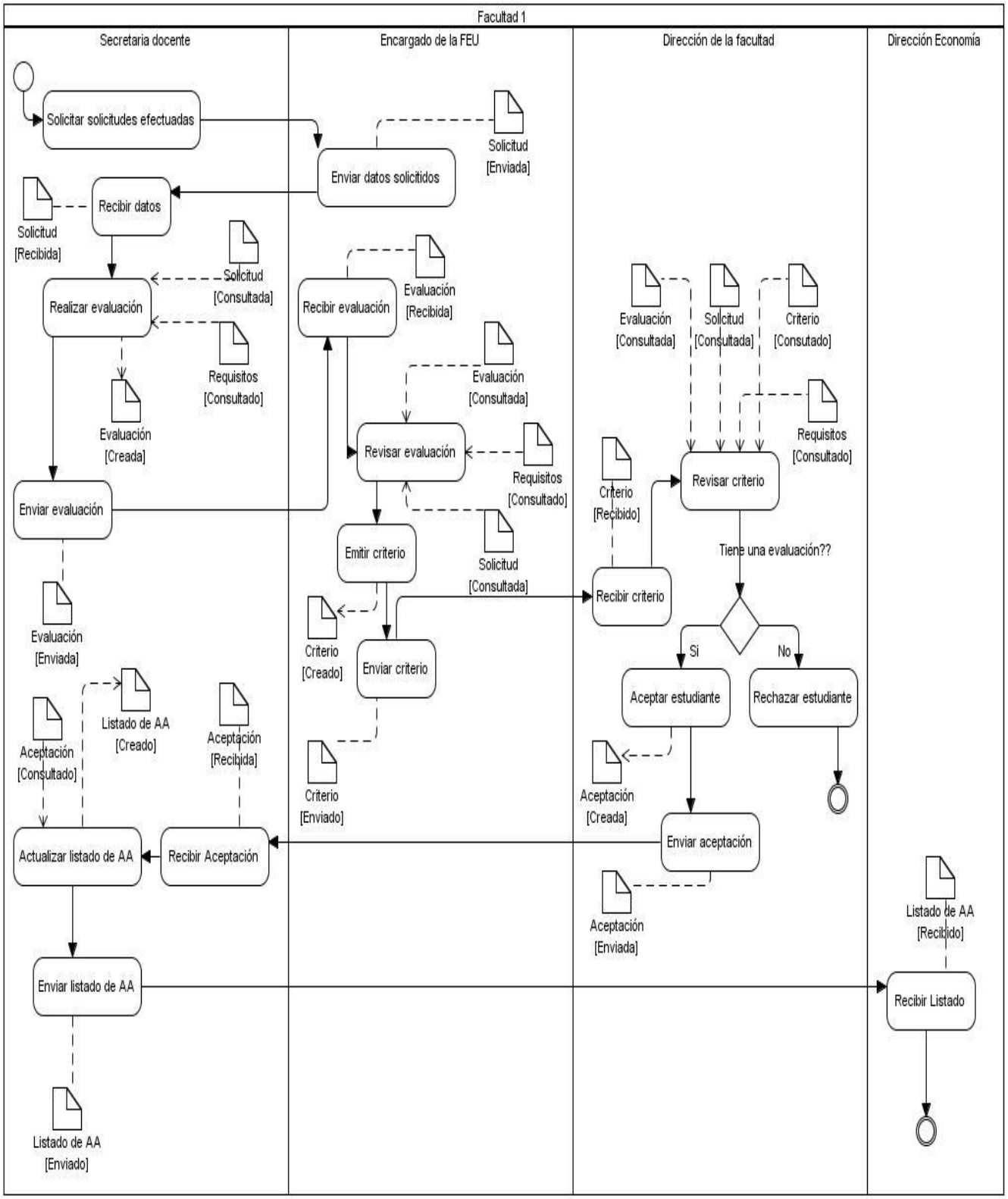


Figura 4: Diagrama de proceso: Aprobar al estudiante.

Ver descripción del proceso aprobar al estudiante en el Anexo 3.

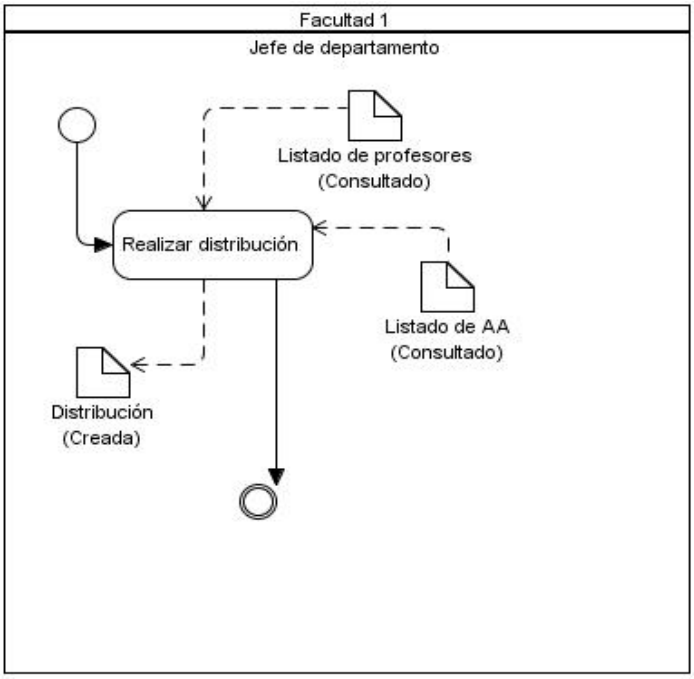


Figura 5: Diagrama de proceso. Asignar tutor a los alumnos ayudantes.

Ver descripción del proceso asignar tutor al alumno ayudante. Anexo 4.

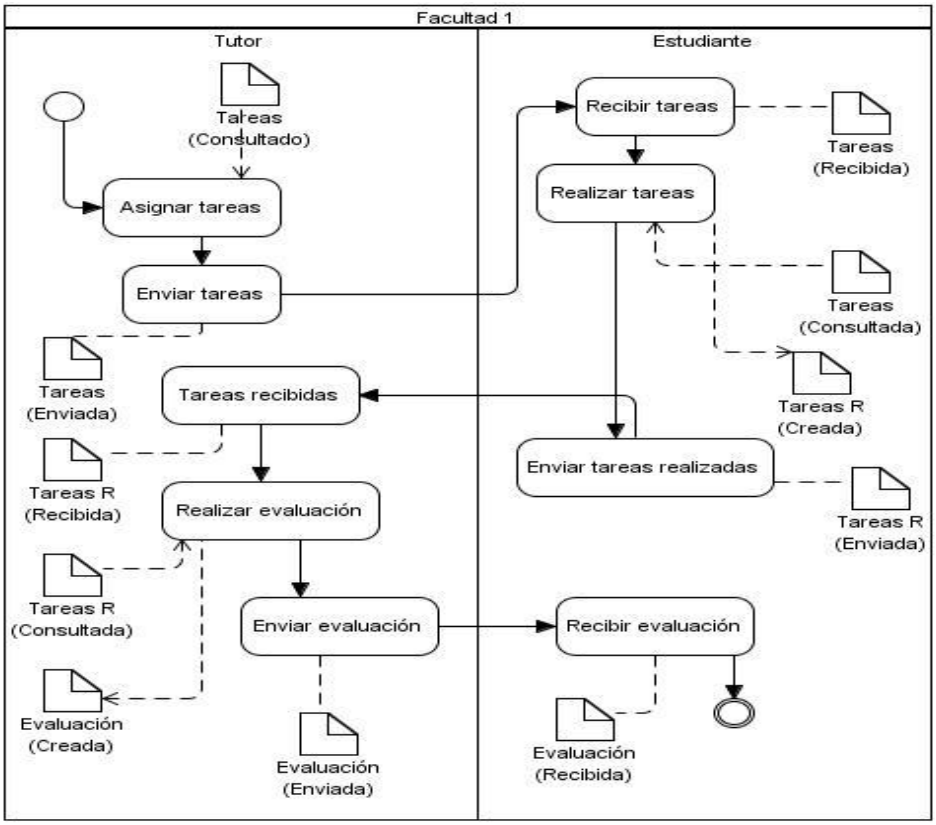


Figura 6: Diagrama de proceso: Asignar tareas a los alumnos ayudantes.

Ver descripción del proceso asignar tareas a los alumnos ayudantes en el Anexo 5.

2.5 Modelo Conceptual

En el Modelo Conceptual se identifican los conceptos (Objetos) significativos del negocio y estos se representan mediante clases. Se relacionan los conceptos representándose como una asociación y se especifica el tipo de relación que existe en lenguaje natural. Se identifican los atributos de cada concepto y se incluyen en la entidad correspondiente. Se identifican las posibles relaciones con conceptos que no pertenezcan al negocio y se representan dentro de un paquete que identifique dicho negocio (Modelo conceptual, 2010).

A continuación se muestra el modelo conceptual:

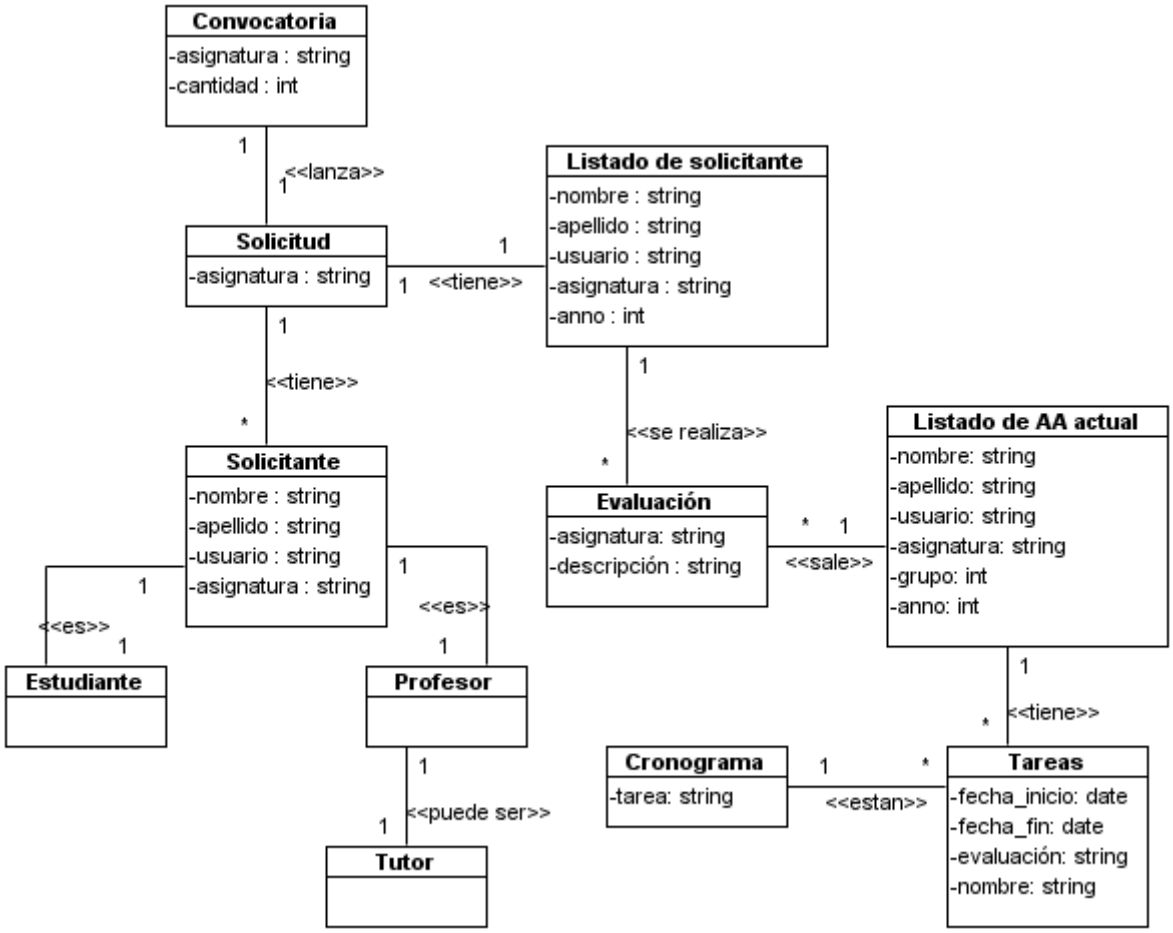


Figura 7: Modelo conceptual.

2.6 Especificación de los requisitos de software.

Los requerimientos representan las necesidades de los usuarios y los objetivos del sistema. Deben ser concisos, completos y especificar claramente todo lo que se necesita llevar a cabo, ya sean entradas, salidas válidas o no; todas las posibles respuestas y situaciones. No deben ser ambiguos, sólo deben tener una interpretación. Deben ser verificables, es decir; se debe poder chequear que cada requisito es cumplido por el *software*. Además, los requisitos deben ser lo más adaptables y modificables posible. Los requerimientos deben ser capaces de satisfacer todos y cada uno de los objetivos del sistema (17).

2.6.1 Requisitos funcionales.

Los requisitos funcionales son aquellas capacidades o condiciones que el sistema debe cumplir. Ellos definen que es lo que el sistema debe hacer y permiten identificar las funcionalidades requeridas. No alteran la funcionalidad del producto, es decir, los requisitos funcionales se mantienen invariables sin importarle con que propiedades o cualidades se relacionen (18).

Requisitos funcionales identificados:

RF 1. Gestionar departamento.

1.1 Adicionar departamento.

1.2 Modificar departamento.

1.3 Eliminar departamento.

1.4 Buscar departamento.

RF 2. Gestionar semestre.

2.1 Adicionar semestre.

2.2 Modificar semestre.

2.3 Eliminar semestre.

2.4 Buscar semestre.

2.5 Asociar año(s) a semestre.

RF 3 Gestionar asignatura.

3.1 Adicionar asignatura.

3.2 Modificar asignatura.

3.3 Eliminar asignatura.

3.4 Buscar asignatura.

RF 4 Gestionar convocatoria.

4.1 Adicionar convocatoria.

4.2 Modificar convocatoria.

4.3 Eliminar convocatoria.

4.4 Buscar convocatoria.

RF 5 Gestionar Solicitud de Alumnos ayudantes.

5.1 Realizar solicitud.

5.2 Cancelar solicitud.

RF 6 Gestionar Aceptación de Alumnos ayudantes.

6.1 Aprobar solicitud.

6.2 Rechazar solicitud.

6.3 Aplazar solicitud.

6.4 Buscar solicitud.

6.5 Mostrar motivo de rechazo o aplazo.

RF 7 Gestionar profesores por departamento.

7.1 Asignar profesor al departamento.

7.2 Eliminar profesor del departamento.

7.3 Asignar asignatura al profesor.

RF 8 Gestionar grupos docentes.

8.1 Asociar grupo(s) a profesor y a alumnos ayudantes.

8.2 Modificar grupos de profesor o alumnos ayudantes.

8.3 Eliminar grupos de profesor o alumnos ayudantes.

RF 9 Gestionar Tutor de Alumnos ayudantes.

9.1 Asignar tutor o tutores a alumnos ayudantes.

9.2 Modificar tutor.

9.3 Eliminar tutor.

9.4 Mostrar tutores de cada alumno ayudante.

RF 10 Gestionar Tareas

10.1 Crear tareas.

10.2 Asignar tarea

10.3 Modificar tareas.

10.4 Eliminar tareas.

10.5 Evaluar tareas.

10.6 Buscar tarea.

10.7 Mostrar tareas de cada alumno ayudante.

10.8 Mostrar alerta al usuario cuando alguna tarea en la cual esté involucrado este por finalizar.

RF 11 Gestionar reportes:

11.1 Mostrar listado de los alumnos ayudantes dado un filtro.

Ver prototipo de interfaz de usuario en el Anexo 6.

2.6.2 Requisitos no funcionales.

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Son esas características que posibilitan que el producto sea atractivo, usable, rápido, confiable, etc. En muchos casos los requisitos no funcionales son fundamentales en el éxito del producto. Normalmente, están vinculados a requisitos funcionales, es decir, una vez que se conozca lo que el sistema debe hacer, se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser (19).

Requisitos no funcionales identificados:

RNF 1: Apariencia o interfaz externa:

1.1 Debe tener el logo de la Facultad 1 de la UCI.

1.2 Las funcionalidades deben estar explícita para lograr una mayor navegabilidad.

1.3 Se debe utilizar colores claros y un tamaño de fuente apropiado para una lectura fácil.

RNF 2: Usabilidad:

2.1 La interfaz será fácil de usar, sencilla, asequible para todo tipo de usuarios, de forma tal que aún los usuarios sin conocimientos básicos de informática puedan interactuar con ella de forma intuitiva.

2.2 El idioma de las interfaces deben ser en español.

RNF 3: Rendimiento:

3.1 Debe ser eficiente al gestionar las solicitudes, permitiendo alcanzar los resultados deseados sin hacer un uso extensivo de la navegación por el sitio.

3.2 Debe estar disponible las 24 horas del día.

3.3 Debe soportar a 1000 usuarios conectados simultáneamente a la base de datos central en cualquier momento dado.

3.4 El tiempo de espera de cada respuesta a una petición debe estar por debajo de los 6 segundos.

RNF 4: Soporte:

4.1 El sistema debe ser de fácil instalación y configuración, con vista a poder darle un mantenimiento asequible en caso de fallos.

RNF 5: Portabilidad:

5.1 Su implementación sobre PHP propicia que sea una aplicación Multiplataforma.

5.2 Deberá correr no sólo sobre Windows sino también sobre Linux para que se pueda llevar a cabo esta acción sin necesidad de efectuar cambios significativos.

5.3 Es compatible con el navegador web Firefox.

RNF 6: Seguridad:

6.1 El sistema podrá ser usado por cualquier persona que acceda a él y requiriéndose conocimientos básicos de computación y trabajo en Web.

6.2 Se debe establecer un sistema de roles para usuarios diferentes como la secretaria, los estudiantes, los responsables de la FEU, tanto el que atiende docencia como el que atiende el movimiento de alumnos ayudantes de la facultad 1, los jefes de departamento, los tutores y la vicedecana de formación. Cada uno de ellos tendrá acceso solo a ejecutar las acciones en las funciones, controlando a través de un sistema de traza todo lo que se realice desde el sistema.

RNF 7: Software:

7.1 El usuario sólo debe tener instalado el navegador web Firefox y los sistemas operativos Windows o Linux

7.2 Servidor web Apache.

7.3 Servidor de Base de Datos Postgres.

RNF 8: Hardware:

- Del lado del cliente:

8.1 Memoria RAM con 256 MB o más.

8.2 Un microprocesador Pentium 4 a más de 1.6 GHz

8.3 Capacidad de disco duro 10GB como mínimo.

- Del lado del servidor:

8.4 Memoria RAM con 512 MB o más.

8.5 Un microprocesador Pentium 4 a más de 2.0 GHz

8.6 Capacidad de disco duro 20GB como mínimo.

8.7 Todos los nodos involucrados en la funcionalidad de la aplicación deben estar conectados a una red que requiere como mínimo 100 Mbps de velocidad.

Ver descripción de los requisitos funcionales en el Anexo 7.

2.7 Beneficios y aportes del sistema.

Con el sistema implementado se logró responder con rapidez y agilidad los procesos referentes al ingreso, seguimiento y control del movimiento de alumnos ayudantes de la facultad 1. Además, se logró manejar todos estos procesos de forma automatizada, contribuyendo así que el trabajo se simplifique.

Por otra parte, se podrá acceder al sistema desde cualquier lugar de la universidad garantizando de esta manera la obtención de datos e información en el momento deseado o necesitado. El sistema además tiene como beneficio que reduce considerablemente el trabajo y evita además, la pérdida de documentos.

2.8 Conclusiones parciales del capítulo 2.

A manera de conclusión se puede citar que en este capítulo 2 se detallaron los procesos que actualmente ocurren en el movimiento de alumnos ayudantes de la facultad 1, se ha realizado una descripción de la propuesta de solución y además, se tuvieron en cuenta una serie de requisitos funcionales y no funcionales que se especificaron que eran importantes para la calidad del sistema que a su vez permite una mejor comprensión del objeto a automatizar.

Capítulo 3: Diseño del Sistema.

3. Introducción.

En este capítulo se abordarán los temas referentes al diseño, se modelarán los artefactos que permitirán dar continuación a la implementación del sistema, como los diagramas de secuencia y de clases del diseño a los cuales se les aplican patrones de diseño para hacer el trabajo más eficiente. Se realizó el modelo de datos representando las tablas de la base de datos vinculadas a la propuesta de solución, así como sus relaciones.

3.1 Patrones de diseño.

Los patrones de diseño ayudan a evitar que los cambios en el sistema se realicen de una forma específica asegurándonos que se afecte lo menos posible. Cada patrón de diseño deja a cierto aspecto del sistema que varíe de forma independiente de otros aspectos, haciendo al sistema robusto a un tipo particular de cambio (20).

3.1.1 Patrones Grasp.

Los patrones Grasp describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. Existen cinco patrones Grasp principales ellos son: Experto, Creador, Alta Cohesión, Bajo Acoplamiento, Controlador y cuatro adicionales que son: Polimorfismo, Fabricación Pura, Inducración y No Hables con Extraños (Grasp, 2010).

De estos se utilizarán los siguientes:

3.1.1.1 Creador.

Problema: ¿Quién debería ser responsable de crear una nueva instancia de alguna clase?

Solución: Este patrón como su nombre lo indica es el que crea, el guía la asignación de responsabilidades relacionadas con la creación de objetos, se asigna la responsabilidad de que una clase B cree un objeto de la clase A solamente cuando:

- B contiene a A
- B es una agregación (o composición) de A
- B almacena a A
- B tiene los datos de inicialización de A (datos que requiere su constructor)
- B usa a A.

Aplicación: La clase GestdepartamentoController, es la responsable de crear instancias de la clase NomDepartamentoModel para realizar las funciones de adicionar, modificar, eliminar y buscar datos en la base de datos e instancias de la clase NomDepartamentoModel en caso de necesitar cargarlos.

3.1.1.2 Experto.

Problema: ¿Quién debería ser el responsable de conocer la información?

Solución: La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada.

Hay que tener en cuenta que esto es aplicable mientras estemos considerando los mismos aspectos del sistema:

- Lógica de negocio
- Persistencia a la base de datos
- Interfaz de usuario

Aplicación: El sistema implementa este patrón en forma de servicios; ejemplo de su aplicación: la clase GestconvocatoriaController.php que al necesitar datos del nomenclador Asignatura este obtiene la información a través de la clase NomAsignatura, la cual es la encargada del acceso a datos de este nomenclador.

3.1.1.3 Alta Cohesión.

Problema: ¿Cómo mantener la complejidad dentro de límites manejables?

Solución: Asignar una responsabilidad de modo que la cohesión siga siendo alta.

Aplicación: Cada elemento del diseño realiza una labor única dentro del sistema, no desempeñada por el resto de los elementos, aunque las convocatorias están relacionadas con las asignatura y los departamentos, no se decide la implementación de una clase general sino la creación de clases para cada proceso

de gestión (GestasignaturaController.php y GestdepartamentoController.php) evitando la sobrecarga, compresión y reutilización de funcionalidades.

3.1.1.4 Bajo Acoplamiento.

Problema: ¿Cómo dar soporte a una mínima dependencia y a un aumento de la reutilización?

Solución: Una clase con bajo acoplamiento no depende de “muchas otras” clases. Las clases con alto acoplamiento recurren a muchas clases y no es conveniente, además son más difíciles de mantener, entender y reutilizar.

Aplicación: Es la idea de mantener las clases más independientes posibles, es decir, que no dependan de otras clases para ejercer sus funcionalidades. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases

3.1.1.5 Patrón Controlador.

Problema: ¿Quién debería encargarse de atender un evento del sistema?

Solución: Asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema, a una clase que represente una de las siguientes opciones:

- El “sistema” global (controlador de fachada).
- La empresa u organización global (controlador de fachada).
- Algo en el mundo real que es activo (por ejemplo, el papel de una persona) y que pueda participar en la tarea (controlador de tareas).

Aplicación: En el sistema cada requisito funcional cuenta con una clase GestasignaturaController que es la encargada de manejar los eventos y la lógica del negocio del sistema relacionado al mismo.

3.2 Diagrama de clase del diseño.

Un diagrama de clase del diseño es un diagrama estático que describe la estructura de un sistema mostrando sus clases, operaciones y las relaciones existentes entre ellos. El mismo contiene clases asociadas, métodos, navegabilidad y dependencias.

Tipos de clases:

- Controladoras: Las clases controladoras o *Controller* permiten la gestión de información entre la vista y modelo, en respuesta a acciones del usuario.
- Modelo:
  - Negocio: Las clases del negocio definen la lógica del negocio de la aplicación.
  - Dominio: Las clases del dominio son las responsables de acceder mediante consultas a la base de datos.
- Vista: Se encarga de mostrar la información recibida del controlador al usuario (21).

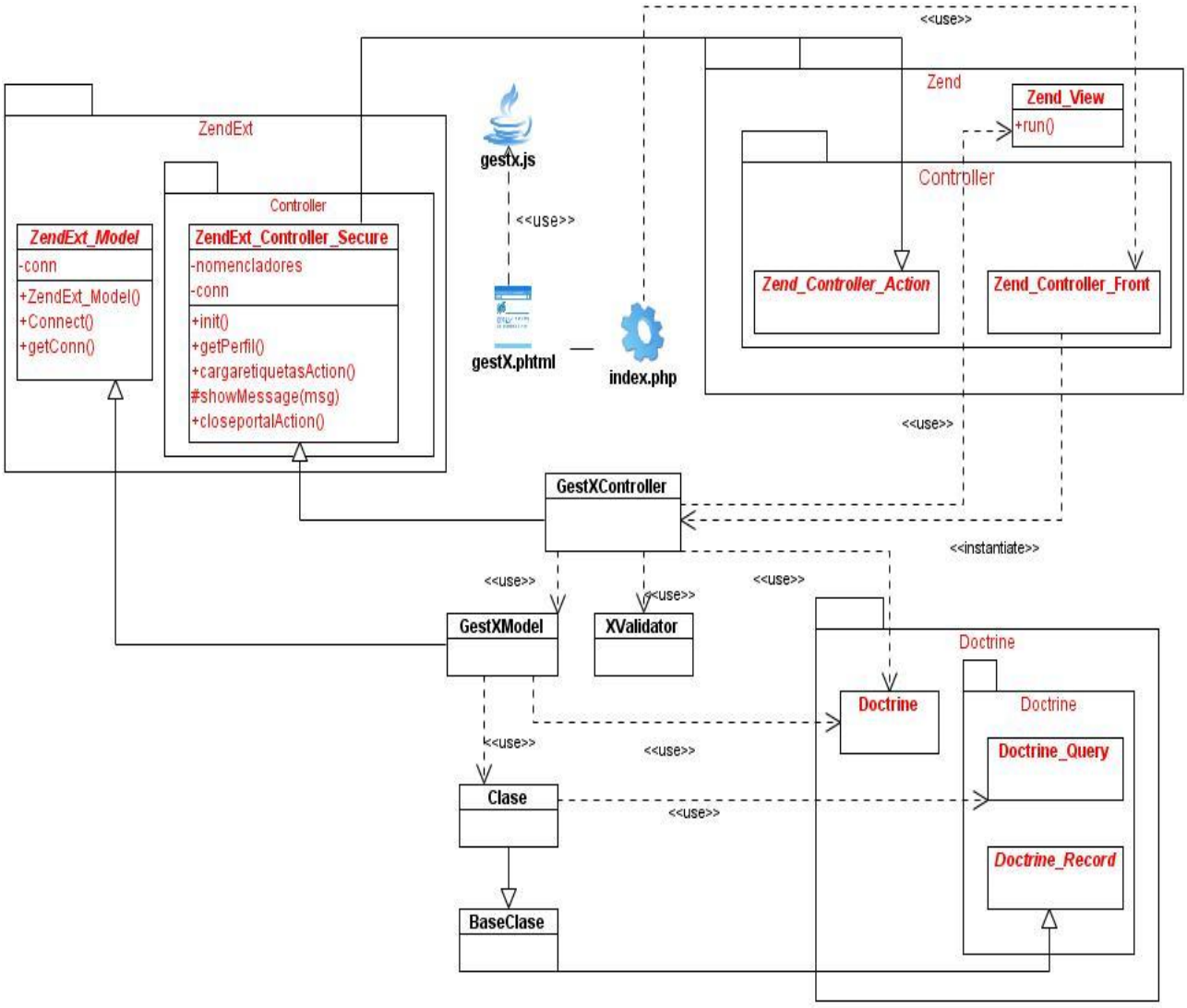


Figura 8: Diagrama de clases genérico.

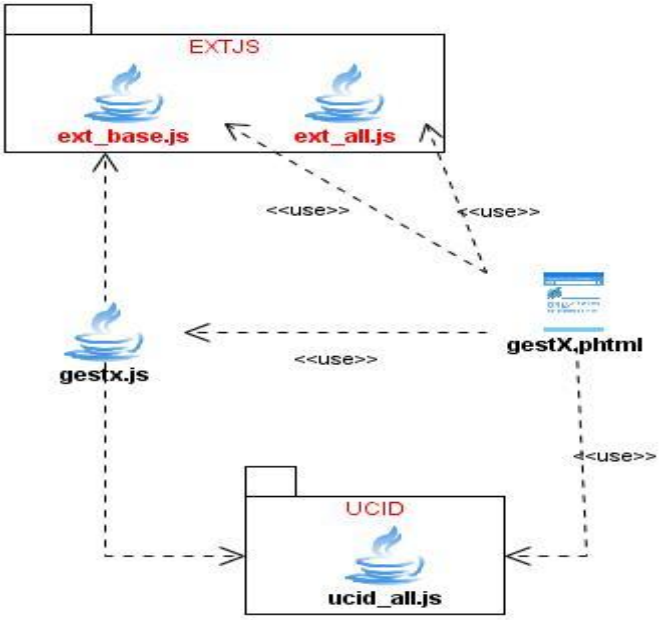


Figura 9: Diagrama de clases genérico para Extjs.

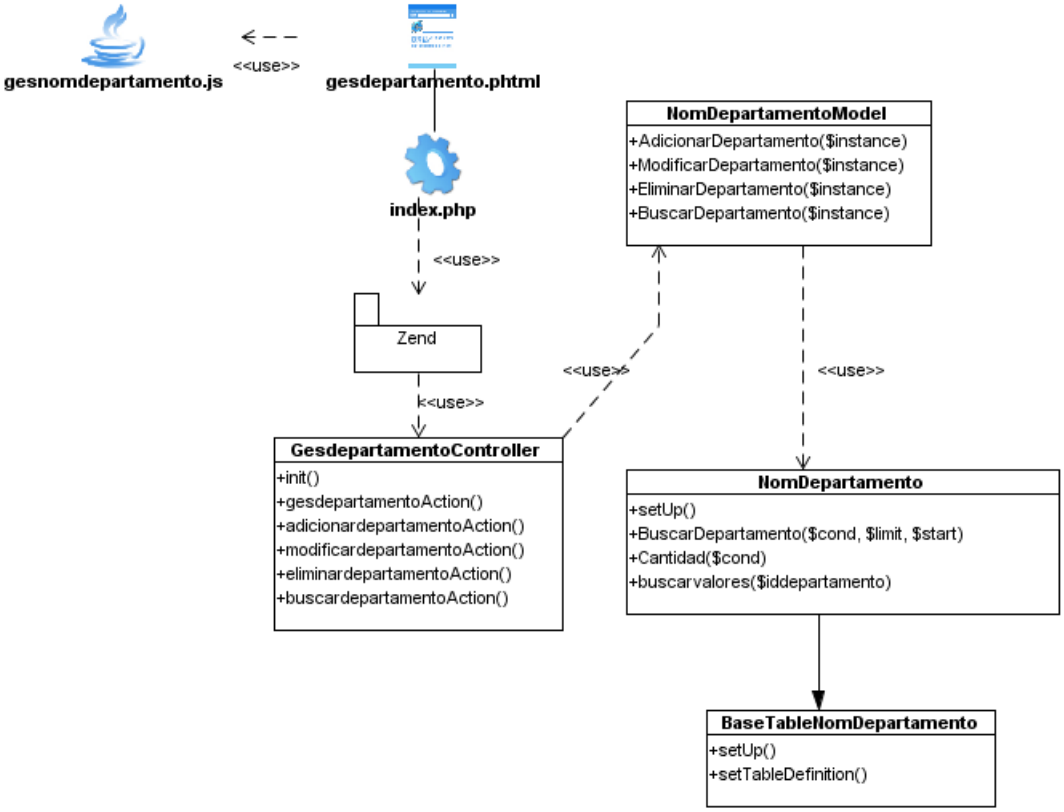


Figura 10: Diagrama de clase del diseño Gestionar departamento.

Ver otros diagramas de clases del diseño en el Anexo 8 y la descripción de los diagramas de clase del diseño en el Anexo 9.

3.3 Diagramas de interacción.

Los diagramas de interacción muestran cómo los objetos se comunican entre ellos mediante la transferencia de mensajes. Son diagramas de interacción los de secuencia, los cuales destacan la ordenación temporal de los mensajes y los de colaboración, permitiendo representar enlaces y mensajes entre objetos (Interacción, 2010).

Ver diagrama de secuencia en el Anexo 10.

3.4 Diagrama de despliegue.

Un diagrama de Despliegue muestra cómo y dónde se desplegará el sistema. Las máquinas físicas y los procesadores se representan como nodos, y la construcción interna puede ser representada por nodos o artefactos embebidos. Como los artefactos se ubican en los nodos para modelar el despliegue del sistema, la ubicación es guiada por el uso de las especificaciones de despliegue (23).

El sistema está conformado por PC que se comunican con el servidor web Apache, este es comunicado con un gestor de base de datos PostgresSQL y un servidor de autenticación LDAP.

Figura 11: Diagrama de despliegue.

3.5 Tratamiento de errores.

El tratamiento de errores es un paso fundamental para lograr un buen funcionamiento del sistema. Los usuarios están expuestos a hacer errores en la inclusión y alteración de los datos y para ello se debe tener en cuenta todas las posibles fallas que pueda tener el usuario. El tratamiento de errores en esta aplicación se realiza con el sistema de captura de errores, permitiendo que una vez que ocurre una excepción el cliente es redireccionado a una página de error con su mensaje correspondiente.

3.6 Seguridad.

En el mundo de hoy en donde mantener la información, bajo la seguridad e integridad requerida, es de una importancia vital, se hace necesaria la existencia de mecanismos que permitan una protección en cuanto a los datos del sistema. A partir de los principios de poder lograr la integridad, confiabilidad y disponibilidad, de la idea antes expuesta, se tuvo en cuenta para el acceso a la aplicación, una previa autenticación en el sistema donde los únicos que pueden accederlo sean los responsables de la FEU, tanto el que atiende docencia, como el que atiende el Movimiento de Alumnos Ayudantes de la facultad 1, los profesores, el vicedecano de formación, los jefes de departamentos, estudiantes y la secretaria docente de la facultad 1. De esta manera, las personas antes mencionadas según el privilegio o los permisos asignados, podrán acceder únicamente a las tareas o acciones que le corresponde ejecutar. A partir de los requisitos funcionales y no funcionales y tal como se había planteado, el sistema poseerá varios usuarios. De este modo se define la seguridad y la credencial o credenciales que se requieran para la ejecución de cada una de las acciones. El encargado de asignar los permisos o asignar los roles a cada usuario es responsable de la FEU que atiende el Movimiento de Alumnos Ayudantes.

3.7 Diagrama Entidad- Relación de la BD.

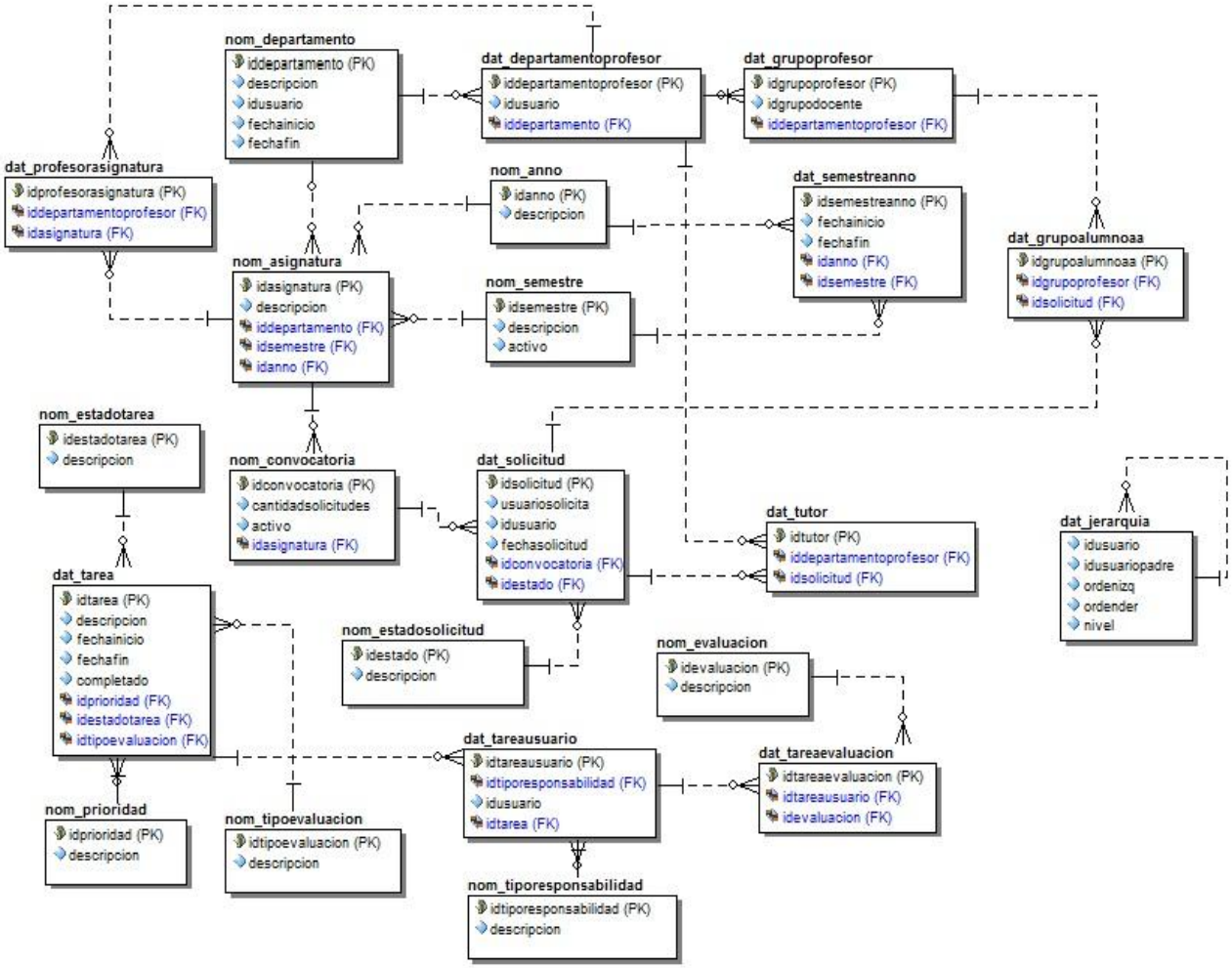


Figura 12: Diagrama Entidad Relación de la Base de Datos.

Ver descripción de las tablas del diagrama Entidad Relación de la Base de Datos en el Anexo 11.

3.8 Conclusiones parciales del capítulo 3.

Como conclusión se puede citar que en el capítulo 3 se detallaron los diagramas de clase del diseño y los de interacción respectivamente. Se mostró el diagrama de Entidad - Relación de la base de datos, el diagrama de despliegue, el diagrama de componente, así como los patrones de diseño que permitieron seguir una línea de trabajo en la realización del diseño.

Capítulo 4: Implementación y Pruebas.

4. Introducción

En el presente capítulo serán abordados todos los temas referentes a la implementación y pruebas del sistema. En el mismo se muestra el diagrama de componentes y la matriz de integración. Además serán abordados todos los temas referentes a las pruebas del sistema.

4.1 Diagrama de componente.

Un diagrama de componentes representa los segmentos de aplicaciones, controladores embebidos, etc. que conformarán un sistema. Un diagrama de componentes tiene un nivel de abstracción mucho más elevado que un diagrama de clase. Usualmente un componente se implementa por una o más clases en tiempo de ejecución. Estos son bloques de construcción, como así eventualmente un componente puede comprender una gran porción de un sistema. (Componente, 2010).

En el diagrama de componente que a continuación se puede observar, el componente Gestionar tareas recibe el árbol jerárquico del componente Configuración, este a su vez le brinda los tutores y los departamentos al componente Gestionar información, además le brinda las asignaturas al componente Convocatoria. El componente Solicitud recibe del componente Gestionar información las asignaturas que imparte un profesores y del componente Convocatoria las convocatorias que hayan sido lanzadas. Por último el componente Proceso de selección recibe las solicitudes realizadas del componente Solicitud.

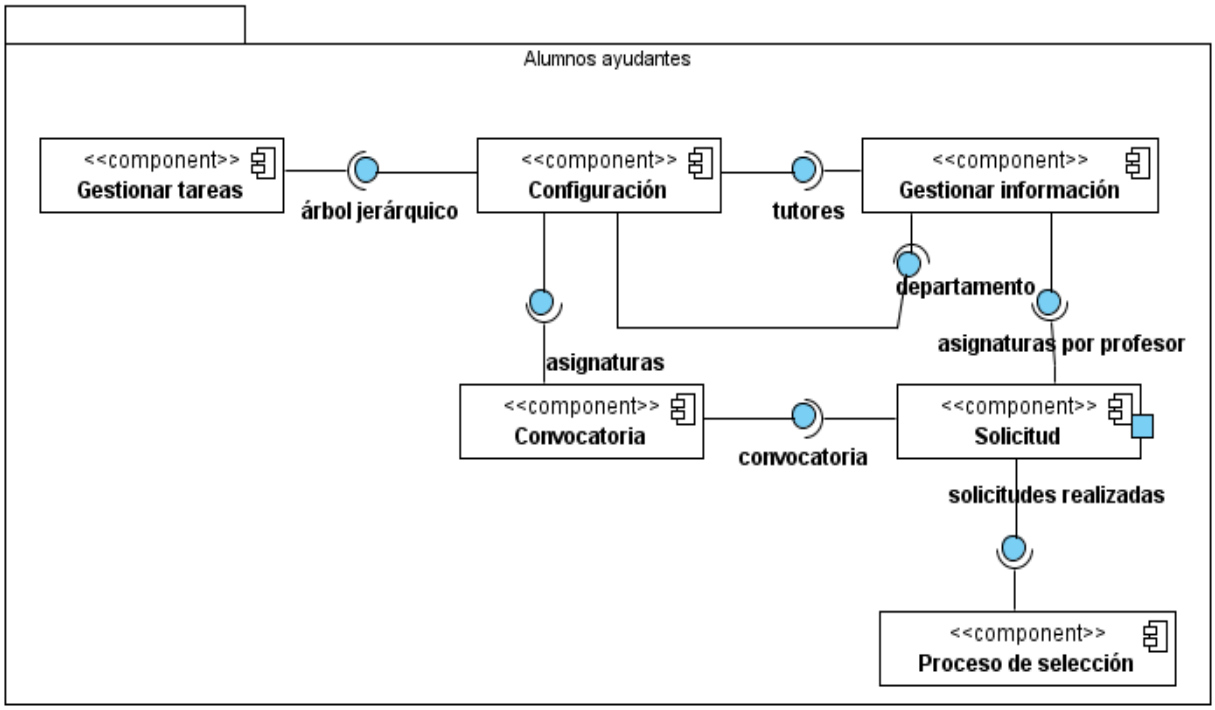


Figura 13: Diagrama de componente.

4.2 Matriz de integración.

La matriz de integración de componentes contiene todos los componentes definidos en el subsistema, de forma matricial. En las intercepciones se especifican los servicios que consume el componente con respecto a su correspondiente. En la siguiente matriz de integración se especifica la integración entre los componentes externos del sistema.

	Componentes externos
Componentes Internos	**Seguridad**
Configuración	BuscarUsuarios
Convocatoria	BuscarUsuarios
Solicitud	BuscarUsuarios

Proceso de selección	BuscarUsuarios
Gestionar información	BuscarUsuarios
Tareas	BuscarUsuarios

Tabla 1: Matriz de integración externa.

4.3 Pruebas de software.

El objetivo de las pruebas, expresado de forma sencilla, es encontrar el mayor número posible de errores con una cantidad razonable de esfuerzo, aplicado sobre un lapso de tiempo realista. Se debe ejecutar el programa antes de que llegue al cliente, con la intención de descubrir todos los errores, de manera que el cliente no experimente la frustración asociada con un producto de baja calidad. Con el propósito de encontrar el mayor número posible de errores, las pruebas deben conducirse sistemáticamente.

Otro de sus objetivos son que:

- 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.
- Reducir costos de mantenimiento.
- Obtener información concreta acerca de fallas, que pueda usarse como apoyo en la mejora de procesos y en la de los desarrolladores.

Los objetivos anteriores suponen un cambio dramático de punto de vista. Quitan la idea que normalmente se tiene de que una prueba tiene éxito si no descubre errores. El objetivo es diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y de esfuerzo. Si la prueba se lleva a cabo con éxito (de acuerdo con el objetivo anteriormente establecido), descubrirá errores en el *software*. Como ventaja secundaria, la prueba demuestra hasta que punto las funciones del *software* parecen funcionar de acuerdo con las especificaciones y parecen alcanzarse los requisitos de rendimiento (Pruebas, 2010).

4.3.1 Pruebas de Caja Negra.

Las pruebas de caja negra se centran en lo que se espera de un módulo, es decir, intentan encontrar casos en que el módulo no se atiene a su especificación. Por ello se denominan pruebas funcionales, y el probador se limita a suministrarle datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo el módulo por dentro.

Las pruebas de caja negra están especialmente indicadas en aquellos módulos que van a ser interfaz con el usuario (en sentido general: teclado, pantalla, ficheros, canales de comunicaciones, etc.)

Las pruebas de caja negra se apoyan en la especificación de requisitos del módulo. De hecho, se habla de "cobertura de especificación" para dar una medida del número de requisitos que se han probado. Es fácil obtener coberturas del 100% en módulos internos, aunque puede ser más laborioso en módulos con interfaz al exterior. En cualquier caso, es muy recomendable conseguir una alta cobertura en esta línea (Prueba de Programas, 2010).

Figura 14: Prueba de caja negra.

4.3.1.1 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. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así, el número total de casos de prueba que hay que desarrollar.

Una clase de equivalencia representa un conjunto de estados válidos o no válidos para condiciones de entrada. Típicamente, una condición de entrada es un valor

numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica.

El objetivo de partición equivalente es reducir el posible conjunto de casos de prueba en uno más pequeño, un conjunto manejable que evalúe bien el *software*. Se toma un riesgo porque se escoge no probar todo. Así que se necesita tener mucho cuidado al escoger las clases (Método de Prueba de Caja Negra, 2010)

4.3.1.2 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 como técnica de prueba. El análisis de valores límite lleva a una elección de casos de prueba que ejerciten los valores límite.

El análisis de valores límite es una técnica de diseño de casos de prueba que completa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el análisis de valores límite lleva a la elección de casos de prueba en los extremos de la clase. En lugar de centrarse solamente en las condiciones de entrada, el análisis de valores límite obtiene casos de prueba también para el campo de salida.

4.3.2 Aplicación de pruebas de caja negra.

Para el desarrollo de este tipo de prueba se utilizará el requisito Gestionar departamento. El objetivo del mismo es persistir en la base de datos la información referente a los departamentos que se han creado en la entidad dando la posibilidad de modificar, adicionar, eliminar y buscar el departamento.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Gestionar Departamento.	El sistema permite adicionar departamentos persistiendo la información en la BD y buscarla en caso necesario.	1.1: Adicionar un departamento introduciendo los datos correctamente.	– El usuario presiona el botón **Adicionar.** – El sistema muestra el formulario a llenar para adicionar el

			departamento. – El usuario introduce los datos correctamente y presiona el botón **Aceptar** o **Aplicar**. – El sistema muestra la notificación “Se ha adicionado satisfactoriamente el departamento”. – El usuario presiona el botón **Aceptar** de la notificación. – El sistema muestra que se insertó el departamento

		1.2: Adicionar un departamento introduciendo datos inválidos o campos en blanco	– El usuario presiona el botón **Adicionar.** – El sistema muestra el formulario a llenar para adicionar el departamento. – El usuario introduce los datos incorrectos y presiona el botón **Aceptar** o **Aplicar** – El sistema muestra la notificación de los campos incorrectos y permite volver a llenar los datos del departamento a adicionar.
		1.4: Cancelar.	– El usuario presiona el botón **Adicionar.** – El sistema muestra el formulario a llenar para adicionar el departamento. – El usuario presiona el botón **Cancelar.** – El sistema cierra el formulario a llenar y limpia los campos.

Tabla 2: Aplicación de prueba de caja negra: Escenario Adicionar departamento.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Gestionar Departamento.	El sistema permite modificar departamentos persistiendo la información en la BD y buscarla en caso necesario.	2.1: Modificar un departamento introduciendo los datos correctamente.	– El usuario selecciona el departamento que se desea modificar – El usuario presiona el botón **Modificar.** – El sistema muestra los datos del departamento que se desea modificar en un formulario. – El usuario introduce los datos que desea modificar y presiona el botón **Aceptar** o **Aplicar.** – El sistema verifica que se hayan modificado los datos. – Si el usuario no modificó los datos el sistema muestra un mensaje “No se ha modificado los valores del departamento. ¿Está seguro de que desea continuar? Si el usuario presiona el

			botón **Aceptar**, se cierra el formulario, si presiona el botón **Cancelar** se muestra nuevamente el formulario para que el usuario pueda seguir modificando. – Si el usuario modificó los datos el sistema muestra un mensaje “El departamento fue modificado satisfactoriamente” y el usuario presiona el botón **Aceptar** de la notificación.

	2.2: Modificar un departamento introduciendo datos inválidos o en blanco al presionar el botón **Modificar.**	– El usuario selecciona el departamento que se desea modificar – El usuario presiona el botón **Modificar.** – El sistema muestra los datos del departamento que desea modificar en un formulario. – El usuario introduce los datos que desea modificar incorrectamente y presiona el botón **Aceptar** o **Aplicar.** – El sistema muestra la notificación de los campos incorrectos y permite volver a llenar los datos del departamento a modificar.
	2.3: Cancelar.	– El usuario selecciona el departamento que desea modificar – El usuario presiona el botón **Modificar.** – El sistema muestra los datos del departamento que

			desea modificar en un formulario. – El usuario introduce o no los datos en los campos y presiona el botón **Cancelar.** – El sistema cierra el formulario a llenar y limpia los campos

Tabla 3: Aplicación de prueba de caja negra: Escenario Modificar departamento.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Gestionar Departamento.	El sistema permite eliminar departamentos persistiendo la información en la BD y buscarla en caso necesario.	3.1: Eliminar un departamento introduciendo los datos correctamente.	– El usuario selecciona el departamento que desea eliminar. – El usuario presiona el botón **Eliminar.** – El sistema muestra la notificación "¿Está seguro de que desea eliminar el departamento seleccionado?" – El usuario presiona el botón **Aceptar** de la notificación – El sistema elimina el departamento y muestra un mensaje confirmando que se eliminó.

Tabla 4: Aplicación de prueba de caja negra: Escenario Eliminar departamento.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Gestionar Departamento.	El sistema permite buscar departamentos persistiendo la información en la BD y buscarla en caso necesario.	4.1: Buscar un departamento introduciendo los datos correctamente	– El usuario introduce los datos que desea buscar y presiona el botón **Buscar.** – El sistema muestra los datos deseados.
		4.2: Buscar un departamento introduciendo datos inválidos	– El usuario introduce los datos incorrectos que desea buscar y presiona el botón **Buscar.** – El sistema muestra la notificación “No se encontraron departamentos que coincidan con su criterio de búsqueda”. ¿Desea volver a mostrar los departamentos? – El usuario presiona el botón **Aceptar** de la notificación. – El sistema permite volver a buscar el departamento.

Tabla 5: Aplicación de prueba de caja negra: Escenario Buscar departamento.

4.4 Niveles de pruebas.

4.4.1 Pruebas de aceptación.

El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema que determine su aceptación, desde el punto de vista de su funcionalidad y rendimiento.

Las pruebas de aceptación son definidas por el usuario del sistema y preparadas por el equipo de desarrollo, aunque la ejecución y aprobación final corresponden al usuario (22).

4.5 Conclusiones parciales del capítulo 4.

En este capítulo se hace un bosquejo de distintos aspectos con gran significado para las pruebas de *software*, resaltándose la importancia de las mismas en el desarrollo de un proyecto. Se aborda acerca de las pruebas de caja negra y de aceptación, obteniendo como resultado que el sistema desarrollado cumple con el funcionamiento esperado y satisface todos los requisitos funcionales.

Conclusiones generales.

En el desarrollo de esta investigación se realizó un estudio de las tendencias históricas y actuales de *software* que gestionan recursos humanos a nivel mundial, en Cuba y en la Universidad. Se estudiaron las diferentes metodologías, herramientas, lenguajes de programación y *framework* que existen en la actualidad para así seleccionar la más conveniente para el desarrollo de la aplicación web.

Se describieron los procesos que conllevan las actividades relacionadas en el movimiento de alumnos ayudantes de la facultad 1 de la UCI, se realizó una descripción de la propuesta de solución y además se tuvieron en cuenta una serie de requisitos funcionales y no funcionales que se decidió que eran importantes para la calidad del sistema, que a su vez permiten una mejor comprensión del objeto a automatizar

Se estudiaron los diferentes patrones de diseño y se seleccionaron los más convenientes, lo que permitió seguir una línea de trabajo en la realización del diseño para posteriormente implementarlo e integrarlo a la facultad 1 de la UCI.

Finalmente, se realizaron un conjunto de pruebas a la aplicación web en busca de posibles vulnerabilidades del mismo, con el objetivo de minimizar los posibles errores.

Como conclusión general se puede afirmar que el trabajo cumplió el objetivo trazado para el desarrollo del mismo, satisfaciendo las necesidades del cliente. Se logró la implementación de una aplicación web que podrá ser usada por cualquier persona que tenga los conocimientos básicos de la computación y que brinda las funciones necesarias para la gestión de las actividades en el movimiento de alumnos ayudantes de la facultad 1 de la UCI.

Recomendaciones.

Luego de haber cumplido con los objetivos propuestos mediante la realización del trabajo, obteniendo así como resultado una solución que permite gestionar las actividades que se llevan a cabo en el movimiento de alumnos ayudantes de la facultad 1 de UCI, se recomienda:

- Proseguir con la investigación e identificar nuevos requisitos para actualizar el sistema desarrollado, garantizando mayor funcionalidad y aceptación del cliente.
- Implantar el sistema en otras facultades.
- Para la evaluación del alumno ayudante, se recomienda integrar la aplicación con el subsistema Evaluación del sistema informativo de atención a los cuadros, el cual fue realizado en la UCID.

Bibliografía referenciada.

1. *Metodología de desarrollo.* http://www.rhernando.net/modules/tutorials/doc/ing/met_soft.html.

2. *BPMN.* http://pana10.files.wordpress.com/2007/12/bpmn1.ppt.

3. *Visual Paradigm.* http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.

4. *JavaScript.* http://max-alva.webs.com/javascript.htm.

5. *HTML.* http://www.euskalnet.net/jaoprogramador/xml/xml05.htm.

6. *XML.* http://www.w3c.es/divulgacion/guiasbreves/tecnologiasXML.

7. *Netbeans.* http://ayuda-java.blogspot.com/2007/07/qu-es-netbeans.html.

8. *PostgresSQL.* https://forja.rediris.es/docman/view.php/312/461/Postgres-Programmer.pdf.

9. *Pgadmin3.* http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.

10. *TortoiseSVN.* http://svnbook.red-bean.com/nightly/es/svn-book.pdf.

11. *Zend Framework.* http://www.carlosleopoldo.com/post/zend-framework-una-ntroduccion/.

12. *Doctrine Framework.* http://www.doctrine-project.org/.

13. *Ext JS.* http://extjs.com/blog/2007/09/28/ext-20-alpha-release/.

14. *IOC.* http://www.hachisvertas.net/blog/01/2008/03/27/inversion-de-control-ioc-inyeccion-de-dependencias-di.

15. *Modelo de Negocio.* http://www.monografias.com/trabajos23/modelos-de-negocio/modelos-de-negocio.shtml.

16. *Modelo de proceso de Negocio.* http://www.teraloc.com/portal/index.php?option=com_content&view=article&id=51&Itemid=92.

17. *Requisitos.* http://lsi.ugr.es/~mvega/docis/requeintro.pdf.

18. *Requisitos F.* http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional.

19. *Requisitos NF.* http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional.

20. *Patrones de diseño.* http://msdn.microsoft.com/es-es/library/bb972240.aspx.

21. *Diagrama de clase del diseño.* http://eisc.univalle.edu.co/materias/Material_Desarrollo_Software/DISCLASES_A12.pdf.

22. *Prueba Aceptación.* http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node55.html.

23. *Despliegue.* http://www.sparxsystems.com.ar/download/ayuda/index.html?deploymentdiagram.htm.

Bibliografía consultada.

(2010). Obtenido de PlanningPME: http://www.planningpme.es/.

(2010). Obtenido de HRmgr: http://www.hr-manager.net/Spain.

(2010). Obtenido de HRCorporate.Version 4.2: http://www.losrecursoshumanos.com/contenidos/1114--sale-a-la-venta-hr-corporate-solucion-para-los-procesos-de-rrhh.html.

(2010). Obtenido de ASSETS NS: http://assets.co.cu/assets.asp.

(2010). Obtenido de Fastos: http://www.desoft.cu/Productos1/FastosPagus/tabid/442/Default.aspx.

(2010). Obtenido de RHCITMA: http://www.monografias.com/trabajos30/gestion-procesos/gestion-procesos.shtml.

(2010). Obtenido de UML: http://mayi.polanco.googlepages.com/TRABAJODEINGSOFTWAREII.doc.

(2010). Obtenido de PHP: http://php.ciberaula.com/articulo/introduccion_php/.

(2010). Obtenido de Apache: http://linux.ciberaula.com/articulo/linux_apache_intro/.

(2010). Obtenido de Mozilla Firefox: http://firefoxperu.blogspot.com/2005/12/qu-es-mozilla-firefox.html.

(2010). Obtenido de MVC: http://www.proactiva-calidad.com/java/patrones/mvc.html.

(2010). Obtenido de Mapa de proceso: http://www.slideshare.net/samespinosa/mapa-de-procesos-1053479.

(2010). Obtenido de Modelo conceptual: http://www.fosonline.org/images/Documents/Medidas/PDF4Tracy/D77927.pdf.

(2010). Obtenido de Grasp: http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=grasp.

(2010). Obtenido de Interacción: http://www.dcc.uchile.cl/~psalinas/uml/interaccion.html.

(2010). Obtenido de Componente: http://www.sparxsystems.com.ar/download/ayuda/index.html?componentdiagram.htm

(2010). Obtenido de Pruebas: http://lsi.ugr.es/~arroyo/inndoc/doc/pruebas/pruebas_d.php.

(2010). Obtenido de Prueba de Programas:

http://www. lab.dit.upm.es/~lprg/material/apuntes/pruebas/ testing.htm#S22.

(2010). Obtenido de Método de Prueba de Caja Negra:

http://www. udistrital.edu.co/comunidad/grupos/ arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20software/node28.html.

Glosario de términos.

AA: Alumnos ayudantes.

BD: Base de datos.

BPMN: Notación para el Modelado de Procesos de Negocio.

CEIGE: Centro de Informatización para la Gestión de Entidades.

FEU: Federación estudiantil universitaria.

Framework: Es un conjunto de prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar.

HTML: Lenguaje de Marcado de Hipertexto.

HTTP: Protocolo de transferencia de hipertexto.

IDE: Entorno de desarrollo integrado.

IOC: Inversión de control.

MVC: Modelo Vista Controlador.

MAA: Movimiento de Alumnos Ayudantes.

Procesos: Es un conjunto de actividades o eventos que se realizan o suceden con un fin determinado.

RF: Requisito funcional.

RNF: Requisito no funcional.

SOA: Arquitectura orientada a servicios.

SSL: Seguridad de la Capa de Transporte.

Software: Sistema, aplicación web.

UML: Lenguaje Unificado de Modelado.

UCID: Unidad de compatibilización integración y desarrollo de productos informáticos para la defensa

UCI: Universidad de las Ciencias Informáticas.

VP: Visual Paradigm.

XML: Lenguaje de Etiquetado Extensible.