Universidad de las Ciencias Informáticas

Facultad 2

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

Sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2.

Autores: Leskenia Acosta Londres

Manuel Alejandro Argudín Chirino

Tutores: Ing. Marisel Rivera Alarcón

Lic. Roxana Pérez Rubido

La Habana, Cuba

Junio, 2014

"Año 56 de la Revolución"

DECLARACIÓN DE AUTORÍA

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

Para que así conste, firmamos la presente a los ____ días del mes de junio del 2014.

Manuel Alejandro Argudín Chirino

_____

Firma del Autor

Leskenia Acosta Londres

_____

Firma del Autor

Ing. Marisel Rivera Alarcón

_____

Firma del Tutor

Lic. Roxana Pérez Rubido

_____

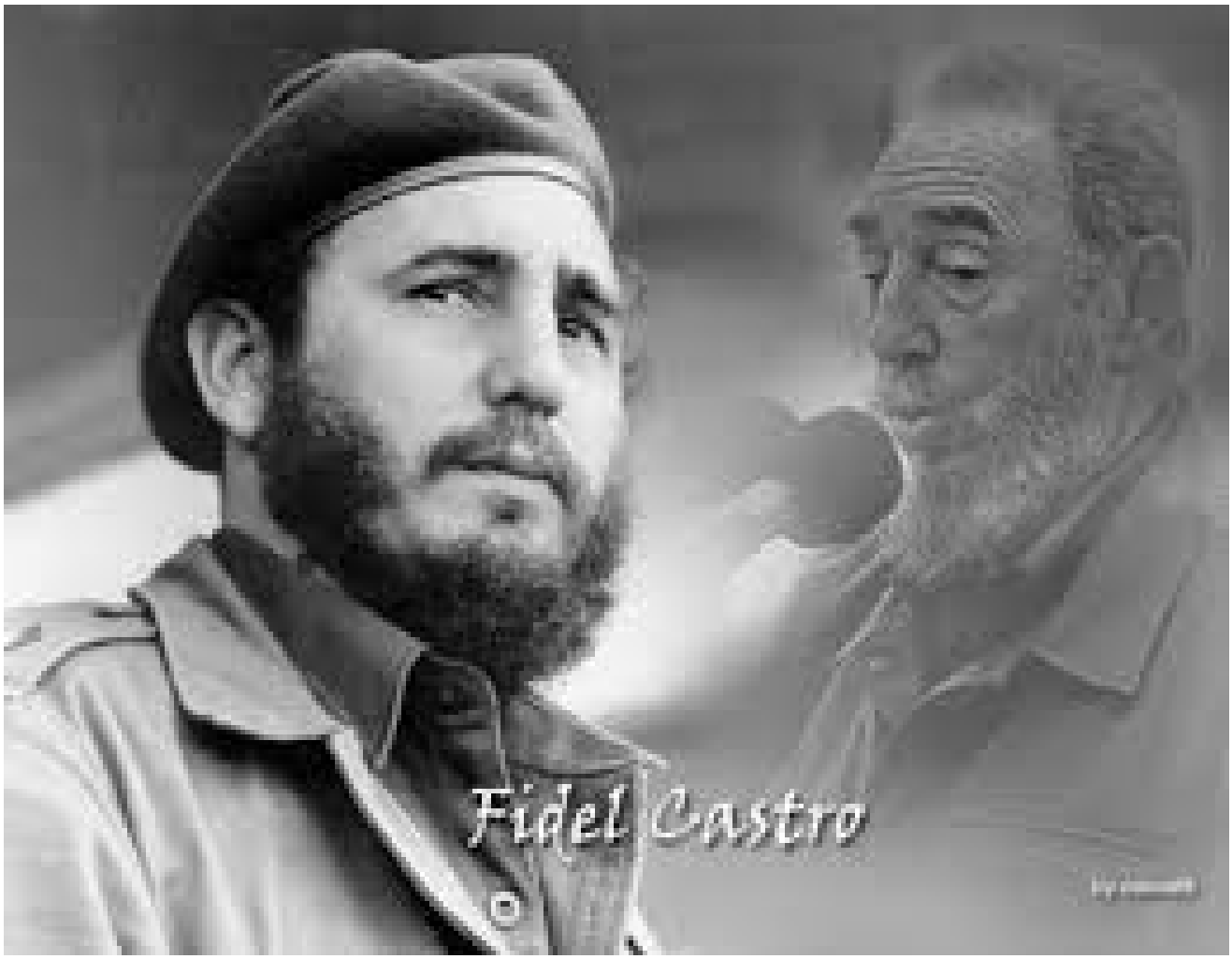
Firma del Tutor

Datos de contacto

Tutores:

Ing. Marisel Rivera Alarcón (malarcon@uci.cu) Ingeniera en Ciencias Informáticas con 5 años de experiencia como profesora. Pertenece al Departamento de Programación de la Facultad 2 y trabaja en el proyecto Prótesis perteneciente al Departamento SES del Centro CESIM.

Lic. Roxana Pérez Rubido (rubido@uci.cu) Licenciada en Ciencias de la Computación con 7 años de experiencia como profesora. Se desempeña actualmente como Jefa de Departamento de Ciencias Básicas de la Facultad 2. Ha impartido las asignaturas Matemática Discreta 1 y 2 y Álgebra Lineal.



"Pero ninguna idea triunfa así, fácilmente. Para que una idea triunfe hay que empezar a pensarla bien, hay que predicarla, hay que defenderla, hay que persuadir a mucha gente, y entonces al final la idea triunfa."

Fidel Castro.

Leskenia: Hoy es un día especial para mí, he podido lograr mi sueño y el de otras personas, enfrentarme al mundo como una profesional. En esto años de sacrificios, sueños, alegrías, tristezas son innumerables las personas a las que tengo que agradecerles por ayudarme a llegar hasta aquí.

A Fidel y la Revolución Cubana, por permitirles de igual manera a las personas con discapacidades físicas integrarse a los estudios y formarlos profesionalmente.

A Dios por mi vida, por mi familia, por haberme permitido vencer todas las barreras.

A mis amigos por aceptarme, soportarme, quererme, apoyarme y aconsejarme: Glauver, Odelis, Diannellys, Dayana, Yaneisi, Adyana, Yenet, Rodelmis. A mis amigos del alma a los que hoy puedo llamar hermanos, porque sin ellos no habría sido posible para mí vivir en estos duros años en la UCI: Leslie, Dora de la Caridad, Daimy, David, Damian, Yusnelys, Barbaro, Rafael. Siéntanse parte importante de este logro, los amo.

A las tutoras, oponente y el tribunal por todas las recomendaciones, críticas constructivas y los momentos que me dedicaron para que el trabajo saliera adelante y con calidad. A mi compañero de tesis que sin él no habría sido posible el acabado de la tesis.

A mis compañeros de grupos: 7103, 7203, 7502, mis profesores y en especial: Rosa de la Caridad, Hugo Vargas, Reinier Alonso. A todos muchas gracias.

A mi novio Mario Ernesto por su amor, comprensión, dedicación y sobre todo por darme su apoyo incondicional en mis malos momentos, gracias por darme la fuerza y convertirme en una persona más luchadora, te amo.

A la familia de Rafael en especial a Milagro, a la familia de Pedro, a la madre de mi novio Thaislen, personas hermosas que conocí y me han hecho sentir parte de su familia, gracias por su amor y cariño, gracias por todo, un besote bien grande.

A mi papi que ha sido mi inspiración, porque gracias a ti he aprendido a ser fuerte y a enfrentar todos los retos que la vida me ha puesto, y más que ser una profesional mi empeño ha sido demostrarte que nada me impide seguir adelante. Espero que te sientas orgulloso de mí.

AGRADECIMIENTOS

A mi mamita gracias por darme la vida, gracias por tu amor, gracias por confiar en mí, gracias por tu apoyo, gracias por estar conmigo en todo momento, gracias por existir, para ti es este regalo.
A mis adorados abuelitos por ayudarme mucho y ser como mis padres, gracias por el amor y la enseñanza que me han dado, por los mimos cuando pequeña y hasta ahora, los amo.
A toda mi familia gracias por estar siempre pendiente de mí y ayudarme cuando he necesitado.
A todos aquellos que de una forma u otra han contribuido a que hoy este aquí, les agradezco eternamente.

AGRADECIMIENTOS

Manuel Alejandro: En este día me gustaría agradecer a todas esas personas que de una forma u otra han aportado su grano de arena en mi formación como futuro profesional. A mi familia en general por la paciencia y preocupación que siempre tuvieron hacia mí: A todos los quiero mucho.

A la persona más importante de mi vida, mi madre: gracias por ser siempre ese ángel que cuidaba de mí. Por tu apoyo incondicional cuando incluso yo creía que la batalla estaba perdida. Por levantarme siempre, sonreírme y decirme con mucha calma:"Tranquilo que todo va a salir bien".

A mi padre Juan Carlos: viejo de verdad que de todos mis amigos eres el mejor. Gracias por todo el soporte que me has dado toda mi vida, a pesar de no estar todo el tiempo juntos. Gracias por darme a esas dos gordas lindas que aunque casi nunca las veo, son otro motivo para seguir adelante y ser cada día mejor.

A Mimi y Papá Chirino: por ser los primeros padres que conocí. Mis abuelos del alma, no sabría como agradecerles todo el cariño que me han brindado. Gracias por no apartarse de mi lado cuando más lo he necesitado, por ser incluso mi cama cuando los sueros no me dejaban tocar un colchón.

A Juan Carlos (Nene): gracias por ser mi segundo papá, por ser mi amigo y hacer la vida de mi mamá tan feliz; por ser mi confidente y quererme como a tu hijo.

A Tata y Ale: por ser más allá de mis tíos, mis hermanos mayores y darme unos primos tan lindos como Rafael y Alexa. Por toda la preocupación y los consejos.

A Marqui por ser ese hermano varón que nunca pude tener. Mi primo del alma, compañero de buenos y malos ratos. Por tener el corazón más grande que he conocido.

A mis mejores amigos para lo bueno y lo malo (más aún): Julito, Yoslandi, Abelito, Andrés, Ismary, Yaima, Dito, Choy, Carlitos,Gabi, Osmín, Felito, Rasciel y Tito; sin ustedes y su apoyo no podría considerarme nada. Ustedes son mis hermanos y no

importa cuán lejos estemos siempre recordaré el haberlos conocido como una de las cosas más linda que me ha ocurrido en la vida.

A mis compañeros de grupo por tantas anécdotas juntos. Por compartir conmigo estos duros cinco años de alegrías y tristezas.

A mi compañera por ser ejemplo de sacrificio y abnegación. Por permitirme compartir con ella este proceso para lograr hacernos ingenieros.

A mis tutoras Maricel y Roxana, no solo por fungir como tutoras, sino por ser mis amigas. Gracias por ser tan lindas y dedicarme su tiempo cuando surgía una duda. Por toda la ayuda y la paciencia.

A Eduardo y Nolberto por toda su ayuda, aun cuando estaban bien ocupados. A pesar de conocerlos hace muy poco me gustaría considerarlos mis amigos.

DEDICATORIA

Con todo el amor y el cariño a mis padres y abuelos, por su incondicional e infinito amor, por la confianza que siempre depositaron en mí, por su preocupación constante, por inspirarme y guiarme en todo momento a ser todo lo que soy, por no defraudarme nunca, por respetarme siempre, en fin por amarlos tanto dedico todo mi esfuerzo de estos años y todo mi empeño por no defraudarlos. Este triunfo también es de ustedes.

A mi especial pareja por amarme y por creer siempre en mí.

A mi familia en general por siempre preocuparse por mí.

A mis amigos por ser especiales conmigo y demostrarme lo importante que es viajar en este mundo acompañado de personas como ellos.

Leskenia

Le dedico el presente Trabajo de Diploma a mi familia por ser el motor impulsor de todas mis acciones.

A mis abuelos Martha (Mimi) y Jesús (Papá), por todo el cariño y amor de este mundo. Mis viejitos del alma los amo con todo mi corazón. Una vida no me alcanza para demostrárselo.

A la mujer más linda del mundo: Gretel, mi mamita linda, te adoro. Ojalá y algún día mis hijos se sientan tan orgullosos de su padre como yo me siento de ti. Eres mi todo.

A mi papá Juan Carlos, por ser el modelo de hombre que algún día quisiera llegar a ser.

A Anyel, Anyélica, Alexa, Rafael y Marcos, nunca dejarán de ser mis niños, no importa cuánto pase el tiempo.

A mis Amigos por ser lo más lindo que me ha pasado en la vida.

Manuel Alejandro

Resumen

En la Facultad 2 de la Universidad de las Ciencias informáticas (UCI), actualmente la información asociada a la gestión de los principales procesos relativos al Movimiento de Alumnos Ayudantes (MAA): captación de los estudiantes, seguimiento y control de las tareas, proceso de baja y evaluación, son tratados manualmente, e influye negativamente en la calidad de la misma, lo que lo convierte en un proceso ineficiente. El presente trabajo se propone desarrollar una aplicación informática que gestione la información de los procesos asociados al MAA de la Facultad 2 de la UCI.

El desarrollo de las funcionalidades se basa en tecnologías web, multiplataforma y utiliza el patrón arquitectónico Modelo Vista Controlador. Emplea PHP v.5.3.8 como lenguaje de programación, *NetBeans* 7.4 como entorno de desarrollo y *PostgreSQL* 9.1 como sistema gestor de bases de datos. El proceso de desarrollo estuvo guiado por la metodología XP.

Se obtuvo como resultado un sistema que mejora la gestión de los procesos asociados al MAA, que permite centralizar la información relacionada con los alumnos ayudantes (AA) garantizando mayor organización y control de sus procesos; y minimizar los posibles errores humanos al contar con un formato estándar para la realización de la evaluación anual.

Palabras claves: aplicación informática, movimiento de alumnos ayudantes, sistemas de gestión.

Índice

ÍNDICE

Introducción

Desde los albores de la humanidad, la información y el uso que se le da a la misma, han cumplido un papel trascendental en el desarrollo del hombre como ser social, incitando así el interés en dominar y conocer los fenómenos que afectaban su vida. Debido a esta necesidad de conocer su medio, el hombre buscó vías a fin de almacenar la información y preservarla para futuras generaciones; pero no es hasta el inicio de la Era de la Información (1950), que esta cobra un marcado sentido en la vida diaria del hombre.

En esta Era de la Información surge un nuevo concepto que hoy en día se encuentra presente en casi todos los sectores de la sociedad denominado: Tecnologías de la Información y las Comunicaciones (TIC), tecnologías cuyas aplicaciones suponen un gran cúmulo de oportunidades, pero a la vez una gran cantidad de desafíos. Los avances obtenidos en materia de tecnologías de información, en los últimos años han sido significativos, posibilitándose numerosas innovaciones en disímiles ramas de la sociedad, así como la convergencia con otras tecnologías.

La República de Cuba, país sometido a un férreo bloqueo desde los inicios de la Era de la Información, no ha desestimado esfuerzos para mantenerse a la par del desarrollo de las TIC. Muchas han sido las instituciones fundadas en el país para garantizar dicho desarrollo. En el sector de la educación se encuentra la UCI, universidad creada al fragor de la Batalla de Ideas en el año 2002, cuyos principales objetivos son: formar profesionales comprometidos con su Patria y altamente calificados en la rama de la Informática; producir aplicaciones y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación y servir de soporte a la industria cubana del *software*.

Uno de los pilares de dicha universidad es el fuerte y desarrollado MAA con el que cuenta, cuyo objetivo es formar a los estudiantes que lo integran, como docentes o futuros investigadores, y así contribuir a satisfacer las necesidades de la universidad y los centros de investigación científica (MES, 2007). Por esta razón, cada año se realiza un proceso de captación de estudiantes con las actitudes y aptitudes necesarias para ingresar a las filas de este movimiento, dándole seguimiento a su quehacer dentro del mismo.

INTRODUCCIÓN

La gestión de los procesos relativos a los alumnos ayudantes: captación de los estudiantes, seguimiento y control de las tareas, evaluación de los alumnos ayudantes, control de las bajas, generación de reportes y sus respectivos subprocesos; generan mucha información que es tratada manualmente lo que trae consigo que el trabajo sea muy lento, engorroso e ineficiente y que aumente la posibilidad de tener información duplicada o que se deteriore. Además, puede provocar que la información que generan estos procesos se pierda debido a que esta no se encuentra segura y centralizada. Otras de las deficiencias presentes es que la orientación y seguimiento de las tareas son enviadas a los alumnos ayudantes mediante el correo, provocando que puedan no llegar las especificaciones de las mismas y también, retardo o incumplimientos en su ejecución. Al no seguirse un estándar en el manejo, presentación y seguimiento de la información, la evaluación anual no sigue los mismos patrones en todos los departamentos docentes propiciando que se puedan cometer errores humanos en el momento de emitir dicha evaluación.

El plan de trabajo del AA contiene tareas generales, sin fecha de inicio y fin, influyendo de forma negativa en el seguimiento y control de su ejecución por parte de los tutores. Al no contar con un histórico de las tareas asignadas, la evaluación anual del AA es elaborada sin tener en cuenta lo que se le orientó y ejecutó, provocando que sea evaluado con poco rigor. La solicitud de baja se hace por escrito recibiéndola el tutor quién debe informar al Vicedecano de Formación, esto puede provocar demoras en la notificación y por ende que se efectúe un pago adicional indebido a estudiantes que ya no ejercen como AA.

De acuerdo a la **situación problemática** planteada anteriormente, surge como **problema a resolver**: ¿cómo mejorar la gestión de la información de los procesos asociados al Movimiento de Alumnos Ayudantes de la Facultad 2 para que apoye la toma de decisiones de los profesores en el proceso de evaluación?

Luego del análisis de la situación actual, la investigación enmarca su **objeto de estudio** en: la gestión de la información de los procesos asociados a la vida académica, centrándose el **campo de acción** en: la gestión de la información de los procesos asociados al Movimiento de Alumnos Ayudantes de la Facultad 2 de la UCI.

INTRODUCCIÓN

Definiéndose como **objetivo general**: desarrollar una aplicación informática que gestione la información de los procesos asociados al Movimiento de Alumnos Ayudantes de la Facultad 2 de la UCI y apoye la toma de decisiones de los profesores en el proceso de evaluación.

La **idea a defender** en el presente trabajo es la siguiente: “si se desarrolla una aplicación informática que gestione la información de los procesos asociados al Movimiento de Alumnos Ayudantes de la Facultad 2 de la UCI, se mejorará la gestión de dicha información, apoyando la toma de decisiones de los profesores en el proceso de evaluación”.

Para dar solución a los objetivos planteados se han definido las siguientes **tareas de la investigación***.*

- Consultar como se realizan los procesos relacionados con la gestión de los alumnos ayudantes en la Facultad 2.
- Realizar el análisis de sistemas internacionales y nacionales utilizados para la gestión de la información, estableciendo similitudes con la investigación en curso.
- Definir herramientas, tecnologías y metodología a utilizar para el desarrollo del Sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2.
- Obtener los artefactos propuestos por la metodología de desarrollo definida.
- Implementar la aplicación informática propuesta utilizando las herramientas y tecnologías definidas.
- Realizar pruebas de software para garantizar el correcto funcionamiento de la aplicación informática implementada.

Los principales **métodos de investigación** empleados en función de dar solución a estas tareas son:

Métodos teóricos:

- ✓ **Método Histórico-Lógico*:*** se utiliza para el estudio de los sistemas de gestión de información académica, obteniéndose los aspectos positivos y negativos de los mismos para guiar el desarrollo de un sistema que mejore la gestión de información de alumnos ayudantes en la Facultad 2.

- ✓ **Método Analítico-Sintético:** se utiliza para lograr extraer los elementos esenciales de la bibliografía consultada durante la descripción de los conceptos asociados al dominio del problema.
- ✓ **Método Modelación:** se utiliza para explicar la realidad con la creación de diagramas. Mediante su utilización se podrán elaborar diferentes tipos de diagramas que brindarán información clara sobre el tema de estudio.

Métodos Empíricos:

- ✓ **Entrevista:** se realizó a la Vicedecana de formación de la Facultad 2 y a Ernesto Alejandro Yero, responsable de vida académica en la universidad por parte de la FEU, para obtener información mediante preguntas realizadas sobre los procesos asociados a los alumnos ayudantes. El modelo de la misma se encuentra en el ANEXO 1.

Con el desarrollo de la aplicación y su posterior despliegue se esperan los siguientes beneficios:

1. La información relacionada con los alumnos ayudantes estará centralizada, garantizando mayor organización y control de sus procesos.
2. Al contar con un formato estándar para realizar la evaluación anual se minimizan los posibles errores humanos que pueden ser cometidos en el momento de emitir la evaluación anual.
3. Los procesos de baja, evaluación anual, seguimiento y control de las tareas asignadas se agilizarán, provocando satisfacción a los usuarios que hacen uso de estos servicios.

El presente trabajo está dividido en tres capítulos que se describen a continuación:

Capítulo 1: Fundamentación teórica del sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

Incluye el estudio del tema a tratar, tanto a nivel internacional como nacional, así como una descripción de conceptos que son esenciales para la comprensión de este trabajo. Además una fundamentación de las tecnologías, herramientas y metodología seleccionadas para utilizar en la solución del problema planteado.

Capítulo 2: Características y diseño del sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

Se realiza una descripción general de la aplicación, así como sus elementos más importantes. Se describen las Historias de Usuario (HU) estableciéndose una prioridad para cada una, luego se realiza la estimación del esfuerzo necesario para su implementación Se procede a la planificación de la etapa de implementación donde se generan el plan de iteraciones y el plan de entregas, se definen los roles del sistema, se describen las tarjetas Clase-Responsabilidad-Colaboración (CRC), se modela el diagrama de dato Entidad-Relación, se establecen patrones de diseño y el patrón arquitectónico.

Capítulo 3: Implementación y prueba del sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

Se detalla brevemente, en las tareas de ingeniería, las funcionalidades a implementar en cada HU previamente descrita. Se establecen los estándares de codificación. Se modela el diagrama de despliegue. Se procede a la realización de pruebas de aceptación del sistema y se muestra el resultado obtenido en las mismas.

Capítulo 1: Fundamentación teórica del sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

En el presente capítulo, se brinda información acerca de los principales conceptos referentes a los sistemas de gestión de información y los alumnos ayudantes. Además se describe la justificación del conjunto de herramientas, tecnologías y metodología utilizadas para desarrollar la aplicación.

1.1. Conceptos generales

Información académica: información relativa al ciclo de vida del estudiante en una academia.

Sistemas de gestión: son estructuras interactivas formadas por personas, equipos y métodos, destinados a crear un flujo de información capaz de proporcionar un fundamento adecuado para la toma de decisiones o sencillamente la solución a problemas específicos.

¿Qué es la gestión de la información?

La gestión consiste en un conjunto de actividades coordinadas para dirigir y controlar una organización; la información es la forma social de existencia del conocimiento consolidada en una fuente determinada.

El concepto de gestión de la información tiene varias definiciones dadas por diferentes autores en sus obras:

- Según L. Woodman en su obra *Information management in large organizations* plantea que: *«la gestión de información es todo lo que tiene que ver con obtener la información correcta, en la forma adecuada, para la persona indicada, al costo correcto, en el momento oportuno, en el lugar indicado para tomar la acción precisa.» (Woodman, 1985)*
- Según Gloria Ponjuán en la obra "*Papel del profesional de la información en la gestión del conocimiento"*, la gestión de la información no es más que: *« [...] el proceso mediante el cual se obtienen, despliegan o utilizan recursos básicos (económicos, físicos, humanos, materiales) para manejar información dentro y para la sociedad a la que sirve. Tiene como elemento básico la gestión del ciclo de vida de este recurso y*

ocurre en cualquier organización. Es propia también de unidades especializadas que manejan este recurso en forma intensiva, llamadas unidades de información.» (Dante, 1998)

Sistemas de Gestión de Información: garantizan la disponibilidad de las herramientas y procedimientos organizativos necesarios para la generación, adquisición, procesamiento, almacenamiento, búsqueda, recuperación, transmisión y uso de la información interna y externa de interés para el trabajo de la organización. (Farrucha, 2006)

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. (MES, 2007)

1.2. Estudio del estado del arte

El desarrollo de la informática es un elemento fundamental en el crecimiento de un país. Esta idea alcanza un auge mayor cuando disímiles empresas e instituciones educacionales se centran en obtener mejores resultados informatizando distintas áreas en busca de agilizar sus procesos o servicios.

En la actualidad los sistemas de gestión se hacen necesarios en el quehacer diario de cualquier institución, ya que los mismos reducen los costos de una gran cantidad de operaciones, tanto monetarios como humanos. Para guiar el desarrollo del presente trabajo se realizó un estudio tanto a nivel internacional como en Cuba de diferentes sistemas de gestión de información relacionada con el estudiantado de una universidad con el objetivo de encontrar similitudes y diferencias en la forma en la que en ellos gestionan la información respecto a la solución que se desea dar en el presente trabajo.

1.2.1. Estudio de sistemas de gestión de información académica en el mundo

En el ámbito internacional se analizaron dos sistemas de gestión de información académica:

SISGIAA: Sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

Sistema de Gestión Universitaria de la Universidad Católica de Salta: sistema que permite administrar una universidad desde el punto de vista estructural hasta el académico. Todos los subsistemas que lo componen están controlados por el subsistema de seguridad. La interfaz está pensada como un *web Browser*, facilitando su instalación y la posibilidad de operación a través de Internet, lo que disminuye así los costos de instalación y administración. Está implementado en Visual Basic.NET y ASP.Net. (Globalsis, 2010)

Sistema de Gestión Académica: herramienta que facilita la gestión de los procesos académicos en beneficio de los estudiantes, egresados, docentes y administrativos de una institución docente. Dentro de las características del sistema se encuentran:

- Aporta medios de gestión y control de la información que se desprende de los procesos misionales.
- Facilita los procesos de Autoevaluación con miras a la Acreditación.
- Facilita herramientas informáticas que permiten llevar registro, control y seguimiento al ciclo de vida del estudiante. (TECNOLOGÍA)

1.2.2. Estudio de sistemas de gestión de información académica en Cuba

A nivel nacional se analizó el siguiente sistema de gestión de información académica:

Sistema de Información Docente de la Educación Superior Cubana (SIGENU): surge con el fin de suplir la necesidad de automatizar los procesos relacionados a la gestión académica de las Instituciones de Educación Superior (IES). Está compuesto por módulos que gestionan la información de un estudiante desde su matrícula hasta que se gradúa o causa baja definitiva. Entre las funcionalidades principales del sistema de gestión se encuentra la inscripción de un estudiante, registro de asignaturas a cursar, registro de evaluaciones, control de bajas, emisión de reportes oficiales entre otros. El proyecto se desarrolla con herramientas de software libre con el paradigma de modelado MDA (*Model-Driven*, *Architecture*) sobre la plataforma J2EE (Java 2 *Platform*, *Enterprise Edition*). El producto se encuentra en explotación sobre el Gestor de Base de Datos PostgreSQL, aunque su diseño e implementación es independiente del Gestor de bases de Datos y del Sistema Operativo. (SIGENU, 2004)

En la UCI se pueden citar sistemas implementados que se encargan de gestionar información referente a los estudiantes, entre ellos Gestión Universitaria y DataFEU.

Gestión Universitaria: sistema desarrollado por la Dirección de Informatización de la Universidad que consta de subsistemas para la gestión de la información generada en diferentes áreas de la UCI. Pregrado, lleva el control del expediente de los estudiantes: plan de estudio, trámites docentes, registro de asignaturas y resumen de evaluaciones. Por su parte, Residencia, como su nombre lo indica, brinda información de donde reside cada estudiante, profesor y familiares, apoyado de otros datos de interés. Está desarrollado con el uso de tecnologías y herramientas libres con el uso del marco de trabajo GUUD (marco desarrollado en la UCI que combina los marcos CodeIgniter y JQuery).

DataFEU: sistema de gestión para la automatización de los principales procesos llevados a cabo por la FEU en la UCI y apoyar a sus principales dirigentes en el proceso de toma de decisiones. Fue desarrollado sobre los marcos de trabajo Symfony y ExtJS empleando MySQL como servidor de bases de datos, ajustándose a la soberanía tecnológica por la que aboga Cuba hoy. Se centra principalmente en el proceso de integralidad de los estudiantes de la UCI registrando la trayectoria de cada uno de ellos desde el 1ero al 5to año de la carrera. Además, se apoya en las evaluaciones de los estudiantes en las diferentes esferas de la vida universitaria (docencia, producción y extensión), en la participación en eventos desarrollados a los diferentes niveles y en los méritos y distinciones obtenidos. Contempla las sanciones y elementos negativos de los cuales pueden ser objeto. El sistema cuenta con módulos para la seguridad y la administración; así como también un módulo para la generación de reportes en formato PDF con el objetivo de extraer estadísticas y centralizar la información hacia los factores de la Universidad. (Escoba*r et al.*, 2012)

Como parte del estudio del arte se analizaron trabajos de diploma desarrollados en años anteriores acerca del tema en cuestión:

Sistema de Gestión del Movimiento de Alumnos Ayudantes desarrollado en la Facultad 2 en el año 2010. Entre los procesos que informatiza se encuentran:

- Autenticar usuario.
- Gestionar solicitud y aprobación de ayudantía.

SISGIAA: Sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

- Evaluar AA.
- Notificar cambios y generar reportes del listado general de los AA existentes en el sistema. (Dominguez y Díaz, 2010)

Sistema de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la Facultad 1 desarrollado en el 2010.

- Gestionar solicitud y aprobación de ayudantía.
- Gestionar tareas.
- Generar reportes del listado general de los AA existentes en el sistema dado un filtro. (Fernández y Campo, 2010)

Solución informática para la gestión de alumnos ayudantes en la gestión académica de pregrado sistema desarrollado en el 2012 en la Facultad 1. Este sistema a juicio de los desarrolladores del presente trabajo es uno de los que mayor similitud presenta con el trabajo que se desea desarrollar en la Facultad 2

- Gestionar solicitud y aprobación de ayudantía.
- Gestionar evaluación semestral y anual de los AA.
- Mostrar listados: AA, Cargos, Solicitudes.
- Proponer estudiante para ayudantía. (Rojas, 2012)

1.2.3. Valoración del estudio del estado del arte

Luego de analizados los sistemas anteriores en el ámbito internacional y nacional, se puede concluir que los desarrollados fuera de la UCI gestionan un gran número de funcionalidades que solucionan un amplio espectro de problemas; sin embargo no existe un *software* que tramite la información referente a los AA. De los trabajos de diploma desarrollados en la UCI, se puede concluir que a pesar de que todos gestionan de una u otra manera procesos relacionados a los AA, no existe una aplicación que gestione todos los procesos que se desean informatizar en la Facultad 2. Es por ello que se requiere de un sistema que unifique en una sola aplicación todos los procesos descritos en dichos trabajos. Por lo anterior se decide implementar una aplicación web para la gestión de los procesos referentes a los AA en la Facultad 2.

SISGIAA: Sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

1.3. Herramientas, tecnologías, metodologías

Para el desarrollo de la aplicación web se realizó un estudio de las principales herramientas, tecnologías y metodologías que guían la producción del software en la actualidad, con el objetivo de definir cuáles utilizar, teniendo en cuenta las características de la aplicación que se desea desarrollar.

1.3.1. Herramientas y lenguaje de modelado

Visual Paradigm v8.0: herramienta *CASE*[1] empleada para visualizar y diseñar elementos de software, para ello utiliza el lenguaje UML (por sus siglas en inglés *Unified Modeling Language*), proporciona a los desarrolladores una plataforma que les permite diseñar un producto con calidad de forma rápida. Facilita la interoperabilidad con otras herramientas *CASE* como *Rational Rose*. Se integra con diversos *IDE* (Entorno de Desarrollo Integrado) como: *NetBeans* (de *Sun*), Eclipse (de IBM[2]), JDeveloper (de Oracle), JBuilder (de Borland). Está disponible en varias ediciones: *Enterprise*, *Professional*, *Community*, *Standard*, *Modeler* y *Personal*. Genera código y realiza ingeniería inversa para diferentes lenguajes de programación como: Java, C++, PHP, XML. En adición se genera código para C#, Visual Basic.net, *ODL* (*Object Definition Languaje*), *Flash Action Script*, *Delphi*, *Perl* y *Python*. Se integra con el *Visio* para importar imágenes del mismo y realizar los diagramas de despliegue. Además exporta e importa los diagramas en el estándar XML.

Ventajas de Visual Paradigm:

- Ofrece entorno de creación de diagramas para *UML* 2.1.
- Disponibilidad en múltiples plataformas.
- Disponibilidad de integrarse a los principales *IDEs*.
- Producto de calidad.
- Fácil de instalar y actualizar.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.

1 **CASE**, Computer Aided Software Engineering (Ingeniería de Software Asistida por Computadoras).

2 **IBM,** International Business Machines.

- Soporta una gama de lenguajes en la Generación de Código e Ingeniería Inversa en Java, *C*++, *PHP*, Esquema de *XML*, *Ada* y *Python*.
- La Generación de Código soporta C #, *Visual Basic*.NET, Lenguaje de Definición de Objeto (*ODL*), *Flash Action Script*, *Delphi*, *Perl*, *Objetivo-C*, y *Ruby*. (Paradigm, 2010)

Enterprise Architect v7.1: es una herramienta de diseño y análisis *UML*, que provee beneficios significativos para ayudar a construir modelos de sistemas de software rigurosos, donde es posible mantener la trazabilidad de manera consistente desde el análisis de requerimientos hasta los artefactos de análisis y diseño, a través de la implementación y el despliegue. *Enterprise Architect* soporta este proceso en un ambiente fácil de usar, rápido y flexible. Es una herramienta multi-usuario, basada en Windows, diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad.

Principales características con las que cuenta:

- Velocidad, estabilidad y rendimiento.
- Trazabilidad de extremo a extremo.
- Construido sobre la base de UML 2.1.
- Perfiles y soporte de tecnología.
- Crear elementos del modelo UML para un amplio alcance de objetivos.
- Ubicar esos elementos en diagramas y paquetes.
- Documentar los elementos que ha creado. (*SPARXSYSTEMS)*

Para determinar la herramienta de modelado a usar para explicar la realidad de los procesos del negocio con la creación de diagramas, el equipo de trabajo partió de la siguiente pregunta para determinar cuál de los estudiados era más factible ¿Qué se necesita hacer en la herramienta? Resultando que los estudiados permiten el modelado BPMN (dado sus siglas en inglés, *Business Process Modeling Notation*). Por lo que se determinó por afinidad de los desarrolladores usar Visual Paradigm para el modelado de los procesos del negocio a informatizar. A continuación se describe el lenguaje de modelado integrado en las herramientas *CASE* antes mencionadas.

SISGIAA: Sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

Lenguaje Unificado de Modelado (UML por sus siglas en inglés, *Unified Modeling Language*): es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es el estándar para representar y modelar la información con la que se trabaja en las fases de análisis y diseño de un sistema informático. Tiene una notación gráfica muy expresiva que permite representar en mayor o menor medida todas las fases de un proyecto informático: desde el análisis con los casos de uso, el diseño con los diagramas de clases, objetos hasta la implementación y configuración con los diagramas de despliegue. UML ofrece una amplia variedad de diagramas que permiten visualizar el sistema desde varias perspectivas, estos son: Diagrama de Casos de Uso, Diagrama de Clases, Diagrama de Objetos, Diagrama de Secuencia, Diagrama de Colaboración, Diagrama de Estados, Diagrama de Actividad, Diagrama de Componentes, Diagrama de Despliegue. Para modelar el comportamiento dinámico del sistema son utilizados los diagramas de actividad, secuencia y colaboración. (Booch *et al*., 1999)

BPMN es la notación de Modelado usada en el presente trabajo para representar los procesos del negocio mediante el siguiente nivel de modelado:

Modelos de proceso: son diagramas de flujo extendido con suficiente información para que el proceso pueda ser analizado, simulado, y/o ejecutado.

Notación de Modelado de Procesos de Negocios (BPMN por sus siglas en inglés, *Business Process Modeling Notation*): es el nuevo estándar para el modelado de procesos de negocio y servicios web. Es una notación a través de la cual se expresan los procesos de negocio en un diagrama de procesos de negocio (BPD), agrupa la planificación y gestión del flujo de trabajo, así como el modelado y la arquitectura.

Características de BPMN

- Proporciona un lenguaje gráfico común, con el fin de facilitar su comprensión a los usuarios de negocios.
- Integra las funciones empresariales.
- Utiliza una Arquitectura Orientada por Servicios (SOA), con el objetivo de adaptarse rápidamente a los cambios y oportunidades del negocio.

SISGIAA: Sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

- Combina las capacidades del software y la experiencia de negocio para optimizar los procesos y facilitar la innovación del negocio. (White, 2004)

1.3.2. Tecnologías y lenguajes de programación del lado del servidor

JAVA: lenguaje de programación orientado a objeto independiente de plataforma, desarrollado con la idea original de usarlo para la creación de páginas web. Permite el desarrollo de aplicaciones bajo el esquema Cliente-Servidor y tiene muchas similitudes con los lenguajes C y C++. (Gosling, 2000)

Su diseño se basa en las siguientes características:

- Simple: elimina la complejidad de los lenguajes como C y da paso al contexto de los lenguajes modernos orientados a objetos.
- Familiar: la sintaxis de Java es similar a la de C y C++.
- Alto rendimiento: java es considerado de alto rendimiento por ser tan veloz en el momento de correr los programas y por ahorrase muchas líneas de código.
- Robusto: el sistema de java maneja la memoria de la computadora (apuntadores, memoria que no se esté utilizando).
- Seguro: presenta ciertas políticas que evitan se puedan programar virus.
- Dinámico: java no requiere que compiles todas las clases de un programa para que este funcione.(Gosling, 2000)

PHP 5.3.8: lenguaje de programación interpretado, diseñado para la confección de páginas web dinámicas. El mismo es ejecutado e interpretado en el servidor, generando código HTML[3] y enviándolo al cliente. Este lenguaje cuenta con muchas ventajas dentro de las que se encuentran:

- Sencillo de aprender.
- El rendimiento de sistemas implementados en PHP es muy superior al de otros implementados en otros lenguajes, por lo que el usuario experimenta una sensación de mayor rapidez y usabilidad.

3 **HTML**: Hipertext Markup Language

- Permite técnicas de programación orientada a objetos (POO).
- Capacidad de conexión con la mayoría de los gestores de bases de datos que se utilizan en la actualidad.
- El precio para su uso es cero, por lo que es gratuito y se puede descargar desde el sitio www.php.net.
- Es multiplataforma, funciona en toda máquina 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 que en otra. (CIBERAULA)

Los desarrolladores del presente trabajo optan por la utilización de PHP v.5.3.8 como lenguaje de programación del lado del servidor, ya que se basan en las principales ventajas con la que cuenta este lenguaje: la curva de aprendizaje con respecto a Java es menor y el tiempo de respuesta de las aplicaciones implementadas en el mismo es menor a otros lenguajes. También se basan en el hecho de que la comunidad PHP en la Universidad es bastante amplia, con un sitio para la publicación de todo tipo de documentación referente a este lenguaje. A continuación se procede a hacer un estudio de los principales marcos de trabajo PHP existentes:

ZendFramework v2.0.6: es un *framework*[4] de código abierto para desarrollar aplicaciones web y servicios web con PHP5. Usa código 100% orientado a objetos en su implementación. La estructura de los componentes de *Zend Framework* es algo único; cada componente está construido con una baja dependencia de otros componentes. Esta arquitectura débilmente acoplada permite a los desarrolladores utilizar los componentes por separado. A menudo se refiere a este tipo de diseño como uso a voluntad.

Aunque se pueden utilizar de forma individual, los componentes de la biblioteca estándar de *Zend Framework* conforman un potente y extensible *framework* de aplicaciones web al combinarse. *Zend Framework* ofrece un gran rendimiento, una robusta implementación Modelo-Vista-Controlador (MVC), una abstracción de base de datos fácil de usar, un componente de formularios que implementa la prestación de formularios HTML y validación y filtrado para que los desarrolladores puedan consolidar todas las operaciones usando de una manera sencilla la interfaz orientada a objetos. Otros componentes, como *Zend_Auth* y *Zend_Acl*, proveen

4 **Framework,** Marco de Trabajo (en inglés).

autentificación de usuarios y autorización diferentes a las tiendas de certificados comunes. También existen componentes que implementan bibliotecas de cliente para acceder de forma sencilla a los *web services* más populares. (Alle*n et al.*, 2009)

Symfony v2.2.1: *Framework* PHP de tipo *full-stack* construido con varios componentes independientes creados por el proyecto *Symfony.* Es un proyecto PHP de *software* libre que permite crear aplicaciones y sitios web rápidos y seguros de forma profesional. Su código, y el de todas sus librerías y componentes que incluye, se publican bajo licencia MIT (*Massachusetts Institute of Technology*) de software libre. La documentación del proyecto también es libre e incluye varios libros y decenas de tutoriales específicos. Entre las principales ventajas que posee este marco de desarrollo se encuentran:

- Soporta múltiples bases de datos sin necesidad de cambiar nada en el código: utiliza Doctrine2, una capa de abstracción que le permite interactuar con varias bases de datos.
- El uso de este *framework* hace que se corrijan errores que existirían si se usara PHP puro: evita la redundancia de código así como lo que algunos llamarían código espagueti, porque se separa el código HTML del código PHP.
- Facilidad de instalación y configuración: ha sido probado tanto en plataformas Windows, como derivadas de Unix.
- Automatiza tareas comunes para que el programador pueda enfocarse por completo en las especificaciones. (Porebsk*i et al., et al*. 2011)

CodeIgniter: es un producto de código libre, libre de uso para cualquier aplicación. Como cualquier otro *framework*, contiene una serie de librerías que sirven para el desarrollo de aplicaciones web y además marca una forma específica de codificar las páginas de dichas aplicaciones y clasificar sus diferentes *scripts*. Lo anterior sirve para que el código esté organizado y sea más fácil de crear y mantener. *CodeIgniter* implementa el proceso de desarrollo MVC, utilizado tanto para hacer sitios web como programas tradicionales. (Alvarez, 2009)

Una vez analizados los anteriores marcos de trabajo se decidió trabajar con Symfony v2.2.1, debido a que cuenta con un grupo de ventajas sobre los demás *framework* PHP, tales como:

- Comunidad: tiene la comunidad de usuarios activos más grande de todos los *frameworks*. Centenares de personas en todo el mundo ayudan a programar el *framework*.
- Documentación: su documentación es muy extensa, en general buena y se actualiza con mucha frecuencia.
- Innovación: Symfony2 ha sido el primer *framework* PHP en popularizar conceptos importantes como la inyección de dependencias.
- Bundles: Symfony2 ya cuenta con unos dos mil *bundles* (un concepto similar a los *plugins* de otras aplicaciones).
- Ecosistema: los mejores proyectos de PHP están creados por programadores que orbitan alrededor del proyecto Symfony y su ecosistema (*Composer*, *Doctrine*, *Propel*, *Capifony*, *Assetic*, *Twig*, *Imagine*, *Behat*, *Monolog*)
- Profesional: es serio, pero no aburre. Programar aplicaciones Symfony2 es suficientemente serio para utilizarlo en el ámbito profesional, pero no lo bastante como para convertir la vida del programador en un infierno de aburrimiento.
- Seguridad: les importa realmente la seguridad. (Symfony, 2012)

Por otra parte se tuvo en cuenta lo popular que es este *framework* dentro de la comunidad de programadores en la UCI, ya que en el sitio www.php.uci.cu se puede encontrar gran cantidad de información referente a Symfony y al desarrollo de aplicaciones web con su utilización.

1.3.3. Tecnologías y lenguajes de programación del lado del cliente

HTML5: quinta versión de HTML que remplaza a HTML4 corrigiendo problemas con los que se encontraban los desarrolladores. Con la aparición del mismo se introducen cambios en la semántica que hacen que la estructura de la web sea más coherente y de fácil entendimiento. Se utiliza para describir texto presentado de forma estructurada y agradable, con enlaces que conducen a otros documentos o fuentes de información relacionadas, y con inserciones de multimedia (gráfico, imágenes, sonido, video). 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. (De Luca, 2011) Este lenguaje viene integrado a Symfony v2.2.1, por lo que no se hace necesaria su instalación y acoplamiento al *framework*.

JavaScript: lenguaje de programación que surgió con el objetivo inicial de programar ciertos comportamientos sobre las páginas web, respondiendo a la interacción del usuario. Hoy es el motor de las aplicaciones más conocidas en el ámbito de Internet: *Google*, *Facebook*, *Twitter*. La Web 2.0 se basa en el uso de *Javascript* para implementar aplicaciones enriquecidas que son capaces de realizar todo tipo de efectos, interfaces de usuario y comunicación asíncrona con el servidor por medio de *Ajax*[5]. (Flanagan, 2002) Para el trabajo desde el lado del cliente se estudiaron dos marcos de trabajo *JavaScript* con el objetivo de seleccionar el adecuado para el desarrollo de la aplicación propuesta.

JQuery v1.11: marco de trabajo *JavaScript* que brinda una infraestructura que facilita la creación de aplicaciones complejas del lado del cliente. El mismo ofrece un grupo de funcionalidades basadas en *JavaScript* y permite hacer cambios en el código HTML sin necesidad de recargarlos haciendo uso de DOM[6] y Ajax. Posee licencia para uso en cualquier tipo de plataforma, personal o comercial. Es un marco de trabajo serio, bien documentado, estable y con un gran equipo de desarrolladores a cargo de la mejora y actualización del mismo. (Chaffer y Swedberg, 2010)

ExtJS v4: de acuerdo a la definición de la www.extjs.com es una librería *Javascript* que permite construir aplicaciones complejas en Internet. Esta librería incluye:

- Componentes UI[7] del alto *performance* y personalizables.
- Modelo de componentes extensibles.
- Un API[8] fácil de usar.
- Licencias *Open source* y comerciales.

Permite crear aplicaciones complejas utilizando componentes predefinidos así como un manejador de *layouts* similar al que provee *Java Swing*, gracias a esto provee una experiencia consistente sobre cualquier navegador, evitando el tedioso problema de validar que el código

5 **Ajax**: JavaScript asíncrono y XML por sus siglas en Inglés
6 **DOM**: Modelo de Objetos del Documento por sus siglas en Inglés.
7 **UI,** User interface (Interfaz de Usuario).
8 **API**: Interfaz de programación de aplicaciones por sus siglas en Inglés.

escrito funcione bien en cada uno (*Firefox*, *Internet Explorer*, *Safari*). Cuenta con una serie de desventajas que podrían hacer tedioso el trabajo de dicho *framework*. Entre ellas:

- Problemas con los motores de búsqueda porque indexan el contenido web estático por lo que los textos cargados de manera dinámica no serán encontrados.
- No se pueden usar fuera de línea, por su naturaleza web estas aplicaciones no pueden ser usadas en el cliente como cualquier otra aplicación.
- No existe una forma fácil de realizar *binding* entre los componentes visuales con el respectivo modelo, lo cual genera que el programador tenga que escribir más código para validar y enlazar los formularios. (Orchar*d et al.*, 2009)

Luego de analizados estos marcos de trabajo, se decidió hacer uso de *JQuery* para la realización de la aplicación, ya que el mismo es más ligero que *ExtJS*, posibilitando así un mayor rendimiento del sistema.

1.3.4. Gestores y clientes de bases de datos

PostgreSQL v9.1: sistema de base de datos de código abierto líder, con una comunidad global de miles de usuarios, colaboradores, decenas de empresas y organizaciones. El proyecto *PostgreSQL* se basa en más de 25 años de ingeniería, iniciando en la Universidad de California, Berkeley, y tiene un ritmo sin precedentes de desarrollo en la actualidad. El conjunto de características de *PostgreSQL* no sólo coincide con los mejores sistemas de bases de datos privativas, sino que los supera en funciones de bases de datos avanzadas, extensibilidad, seguridad y estabilidad.

Ventajas de *PostgresSQL*:

- Es una base de datos 100% ACID[9]
- Soporta distintos tipos de datos además del soporte para los tipos base: datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC[10], IP[11]), cadenas de bits. También permite la creación de tipos propios.

9 **ACID,** acrónimo de **A**tomicity, **C**onsistency, **I**solation and **D**urability: Atomicidad, Consistencia, Aislamiento y Durabilidad en español.

10 **MAC**: control de acceso al medio por sus siglas en Inglés

- Incluye herencia entre tablas, es por ello que se le incluye entre los gestores objeto-relacionales.
- Soporta un subconjunto de SQL92[12] mayor que el que soporta MySQL y tiene ciertas características orientadas a objetos.
- Copias de seguridad (*Online/hot backups*)
- *Unicode*
- *Multi-Version Concurrency Control* (MVCC)
- Multiples métodos de autentificación
- Acceso encriptado vía SSL [13](Douglas y Douglas 2003; Postgresql)

SQLite: es un motor de base de datos SQL embebido. A diferencia de la mayoría de las otras bases de datos SQL, no tiene un proceso servidor independiente. Lee y escribe directamente en archivos de disco ordinarios. Una base de datos completa de SQL con varias tablas, índices, *triggers* y vistas, está contenida en un archivo de disco único. El formato de archivo de base de datos es multiplataforma - se puede copiar libremente una base de datos entre sistemas de 64 bits, de 32 bits o entre arquitecturas *big-endian* y *little-endian* -. Estas características hacen de *SQLite* una opción popular como formato de archivo de aplicación. Las transacciones son *ACID* aunque interrumpidas por fallos del sistema o fallas de energía. (Newman, 2004)

MySQL: es un sistema de administración de bases de datos DBMS (*Database Management System*) para bases de datos relacionales. Fue escrito en C y C++, destaca por su gran adaptación a diferentes entornos de desarrollo, permitiendo su interacción con los lenguajes de programación más utilizados como *PHP*, *Perl*, *Java* y su integración en distintos sistemas operativos. Entre las principales ventajas con las que cuenta, se destacan:

- Mayor rendimiento, mayor velocidad tanto al conectar con el servidor como al servir *selects*.
- No hay límites en el tamaño de los registros.
- Aunque haya fallos en el sistema, no suele perder información ni corromper los datos.

Por otra parte presenta como principal desventaja:

11 **IP**: Internet protocol por sus siglas en Inglés.
12 **SQL**: estándar de lenguaje de consulta estructurado por su siglas en Inglés.
13 **SSL:** capa de conexión segura por su siglas en Inglés.

- No soporta transacciones, *roll-backs* ni *subselects*. (Kofler, 2001)

Los tres gestores de bases de datos analizados pueden ser utilizados; pero por mayor dominio de la tecnología, se escogió *PostgreSQL*. Una vez escogido el gestor de la base de datos se decidió utilizar como cliente para el mismo *pgAdmin*.

pgAdmin III v.1.14.1: es la plataforma de desarrollo más popular y rica en características de administración de código abierto para *PostgreSQL*. La aplicación puede utilizarse en *Linux*, *FreeBSD*, *Solaris*, *Mac OSX* y las plataformas de *Windows* para administrar PostgreSQL 7.3 y superiores. Está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL sencillas hasta el desarrollo de bases de datos complejas.

Características de *pgAdmin***:**

- Interfaz gráfica compatible con todas las características de *PostgreSQL* que facilita la administración.
- La conexión con el servidor se puede hacer a través de TCP/IP o *Unix Domain Sockets* (en plataformas * unix), y puede ser encriptado SSL para la seguridad.
- No se requieren controladores adicionales para comunicarse con el servidor de base de datos. (PGADMIN)

1.3.5. Servidor de aplicaciones

Servidor web Apache: es un software libre y el servidor web más popular. Algunos sondeos realizados demuestran que más del 70% de los sitios web en internet están manejados por Apache, haciéndolo más extensamente usado que todos los otros servidores web juntos. Este es un proyecto de la Fundación de Software Apache, con el objetivo de suministrar un servidor seguro, eficiente, y extensible que proporcione servicios HTTP[14] en sincronía con los estándares actuales. Apache es flexible, rápido, eficiente, continuamente actualizado y adaptado a los nuevos protocolos HTTP.

Características de Apache Web Server:

14 **HTTP**: protocolo de transferencia de hipertexto por sus siglas en inglés.

- Modular: puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la API de programación de módulos, para el desarrollo de los específicos.
- Extensible: gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP.
- Multiplataforma. (Kabir, 1998)

1.3.6. Entorno de Desarrollo Integrado (IDE)

NetBeans v7.4: entorno de desarrollo integrado para *Windows*, *Mac*, *Linux* y *Solaris*, de código abierto patrocinado por *Sun*. Tiene como principales ventajas:

- 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 por sus siglas en inglés).
- El soporte al modelado mejora la productividad del desarrollador.
- Perfecto entorno de desarrollo. (Boudreau, 2002)

PhpStorm 7: Entorno de desarrollo propio para lenguaje PHP, soporta las versiones del lenguaje a partir de la versión 5.3, desarrollado por *JetBrains*. Proporciona autocompletado de código, refactorización, prevención de errores *on-the-fly*, soporta mezclas de idiomas. Interpreta los tipos de variables y clases internas (incluso cuando cambian de tipo o *callbacks*) de forma rápida e inteligente, lo cual representa un incremento en productividad. Brinda soporte y apoyo a los estándares PSR-0, PSR-1 y PSR-2. Presenta descargas para *Linux*, *MacOSX*, *Windows*, **BSD* y *Unix*. Es de carácter privativo por lo que no es libre ni gratuito. Para poder hacer uso del mismo hay que adquirir por medio de la compra su licencia. (Gajda, 2013)

Por cuestiones de libertades en el software, se decidió hacer uso de *NetBeans* como IDE, ya que *PhpStorm* es de carácter privativo y los desarrolladores del presente trabajo desean hacer uso de tecnologías libres para la confección de la solución propuesta.

1.3.7. Metodología de desarrollo

Para la realización de la solución propuesta, se llevó a cabo un estudio de tres metodologías de desarrollo de software, con el fin de contar con una guía rectora en la obtención del objetivo trazado.

Extreme Programming (XP): metodología ágil para el desarrollo de software, muy popular, se centra en las necesidades del cliente con el objetivo de lograr un producto de calidad y aminorar el tiempo de entrega de la solución. Su principal objetivo es el logro de una relación de comunicación armoniosa entre el cliente y el equipo de desarrollo para así llegar a un producto informático que cumpla con las peticiones del primero y garantizar el éxito en el desarrollo de software. Promueve el trabajo en equipo estableciendo un buen clima. Esta metodología es la adecuada para proyectos pequeños y con requisitos imprecisos y muy cambiantes.

Se basa en:

- **Pruebas Unitarias**: son las pruebas realizadas a los principales procesos, de tal manera que se adelanta en algo hacia el futuro, se pueden hacer pruebas de las fallas que pudieran ocurrir.
- **Re fabricación:** se desarrolla en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- **Programación en pares**: consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento.

Ventajas del uso de XP:

- Se obtiene gran optimización en el código, ya que el mismo es generado para una arquitectura específica. Esto trae consigo un ahorro de hardware y tiempo de ejecución.
- Las posibilidades de fracasar el proyecto son muy bajas, ya que el cliente participa continuamente en el mismo y ante cualquier cambio de última hora, se vuelven a replantear los objetivos.

- Los errores son encontrados tempranamente ya que el producto se prueba continuamente. (Beck, 2000)

Proceso Unificado de Desarrollo (RUP): es una metodología de desarrollo de software basado en componentes e interfaces bien definidas, y junto con UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP se divide en 4 fases, dentro de las cuales se realizan varias iteraciones según el proyecto: Fase de Inicio, Fase de Elaboración, Fase de Construcción, Fase de Transición. Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso).

Sus características principales son:

- Unifica los mejores elementos de metodologías anteriores.
- Preparado para desarrollar grandes y complejos proyectos.
- Orientado a Objetos.
- Utiliza el UML como lenguaje de representación visual.
- Permite ingeniería inversa. (Belloso Cicilia, 2009)

SCRUM: es una metodología en la que se aplican de manera regular un conjunto de mejores prácticas para trabajar colaborativamente en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. En *SCRUM* se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, *SCRUM* está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Los principales beneficios que proporciona *SCRUM* son:

- Entrega periódica de resultados de los requisitos más prioritarios ya completados, lo cual proporciona las siguientes ventajas:
    - Gestión regular de las expectativas del cliente y basada en resultados tangibles.
    - Resultados anticipados (*time to market*).
    - Flexibilidad y adaptación respecto a las necesidades del cliente, cambios en el mercado.
    - Mitigación sistemática de los riesgos del proyecto.
- Productividad y calidad.
- Alineamiento entre el cliente y el equipo de desarrollo.
- Equipo motivado. (PROYECTOSAGILES)

Dado la falta de recursos humanos, poca envergadura del proyecto y el corto tiempo de entrega del mismo, se decidió el uso de las metodologías ágiles porque a juicio de los desarrolladores son las que más se ajustan al presente trabajo. Las metodologías ágiles son básicamente para proyectos pequeños, siempre están preparadas para cambios durante el desarrollo y sobre todo al cliente se le considera parte del equipo, además porque una de sus mejores características es que presentan de manera continua versiones de software funcionando. El uso de la metodología ágil XP respecto a *SCRUM* está basado en la diferencia entre ambas mostradas en la **Tabla 1**.

SCRUM	XP
Las iteraciones de entregas son de dos a cuatros semanas y se conocen como *sprint*.	Las iteraciones de entregas son de una a tres semanas.
Al finalizar un *sprint*, las tareas que se han realizado en él y en las que el cliente ha mostrado su conformidad, no se vuelven a tocar en ningún momento. “Lo que se termina, funciona y está bien, se aparta y no se toca.”	Las tareas que se van terminando en las diferentes entregas al cliente son susceptible a modificaciones durante el trascurso del todo el proyecto, incluso después que funcione correctamente.
El equipo de desarrollo de *SCRUM* trata de seguir el orden de prioridad que marca el cliente, pero si ven que es mejor modificar dicho orden para el desarrollo de las tareas, pueden hacerlo.	El equipo de desarrollo de XP sigue estrictamente el orden de prioridad marcado por el cliente, (aunque el equipo de desarrollo le ayude a decidir, ellos son los que mandan.)
Se basa en la administración del proyecto.	Se centra más en la programación o creación del proyecto.
Cada miembro del equipo *SCRUM* trabaja de forma individual.	Los miembros del equipo programan en parejas.

Tabla 1 Comparación entre metodologías ágiles (CHAVES *et al*. 2012)

En este capítulo se hizo un estudio del arte a nivel internacional y nacional de sistemas de gestión de información. Además, se estudiaron varias tecnologías, herramientas y metodologías; definiéndose XP como metodología de desarrollo; *Visual Paradigm* como herramienta de modelado, UML como lenguaje de modelado, *Netbeans como* entorno de desarrollo, como lenguajes de programación PHP, HTML 5 y *JavaScript*, como marcos de trabajo Symfony2 y *JQuery*, como gestor de base de datos PosgreSQL, como cliente para el gestor de base de datos pgAdmin y como servidor de aplicaciones web Apache.

Capítulo 2: Características y diseño del sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

En el presente capítulo se describe la aplicación propuesta y se exponen las características fundamentales de la misma. También se definen la arquitectura y los patrones de diseños utilizados; así como los actores con los que interactúa el sistema. Se muestran, además, los artefactos generados mediante la metodología XP.

2.1. Objeto de automatización

En el análisis del problema se identificaron los principales procesos que desarrolla el negocio en la actividad de gestión de los alumnos ayudantes para dar solución al problema existente actualmente. Seguidamente se describen dichos procesos que sustentan el negocio:

- Proceso Solicitar Alumno Ayudante
- Proceso Aprobar Alumno Ayudante
- Proceso Dar baja a un Alumno Ayudante
- Proceso Evaluar a un Alumno Ayudante

2.1.1. Descripción de los procesos

Proceso Solicitar Alumno Ayudante

Solicitar ayudantía surge a partir de la necesidad en los departamentos docentes de la Facultad 2. Los jefes de departamentos, al inicio de cada semestre, solicitan una cantidad de alumnos ayudantes, por cada asignatura al Vicedecano de Formación. La información es enviada al estudiante que atiende vida académica por la FEU, quien es el encargado de trasmitirla al resto del estudiantado. La misma se envía por correo o se publica en el sitio de la Facultad 2. Una vez divulgadas las convocatorias, si un estudiante desea presentar la solicitud de ayudantía en una o varias asignaturas lo notifica personalmente o vía correo electrónico al responsable de vida académica. Luego de cumplirse la fecha límite de la convocatoria publicada queda registrado en un documento físico, Word o una hoja Excel el listado de los aspirantes. Proceso muy extenso que desde que inicia hasta que termina requiere de mucho intercambio de información entre los involucrados.

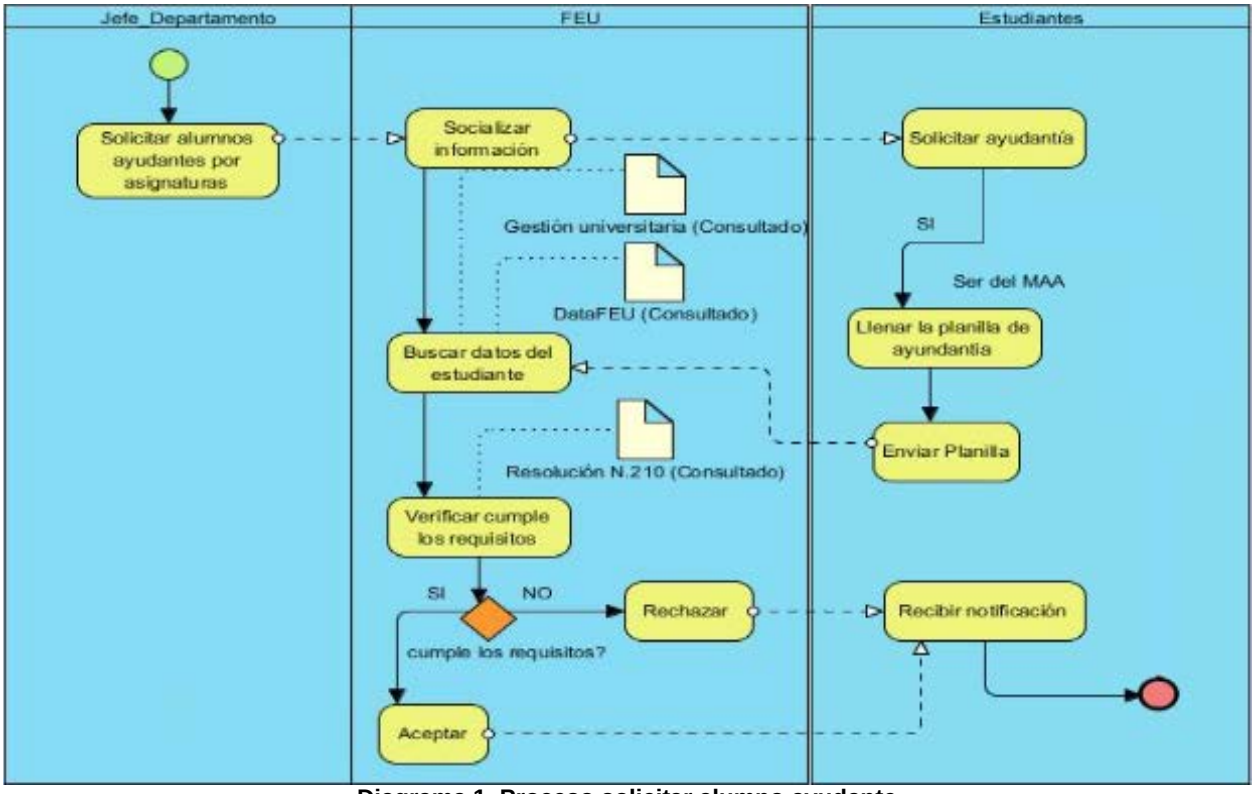


Diagrama 1. Proceso solicitar alumno ayudante

Proceso Aprobar Alumno Ayudante

Para el proceso de aprobación de los alumnos ayudantes se hace necesario revisar determinados datos por cada estudiante presente en la lista de solicitudes como: índice académico, nota de la asignatura en la que desean ser alumnos ayudantes, entre otros datos de interés. La encargada de manejar esta información es la secretaria docente haciendo uso de documentos físicos como planilla de integralidad, expediente del estudiante o de sistemas informáticos de la Universidad, por ejemplo Gestión Universitaria. Una vez realizada la recopilación de los datos de los aspirantes se procede al análisis de cada solicitud para verificar si cumplen con los requisitos para ingresar al movimiento y así escoger las mejores propuestas de acuerdo con la cantidad de alumnos ayudantes que se necesiten por asignaturas. Luego se les informa a los estudiantes el resultado de sus solicitudes (aceptada o denegada) quedando registrado en un documento la lista de estudiantes que formarán parte del Movimiento. Al alumno ayudante se le asigna un profesor tutor que será el encargado de velar por el cumplimiento de sus tareas; el grupo que deberá atender y la asignatura en la que se desempeñará.

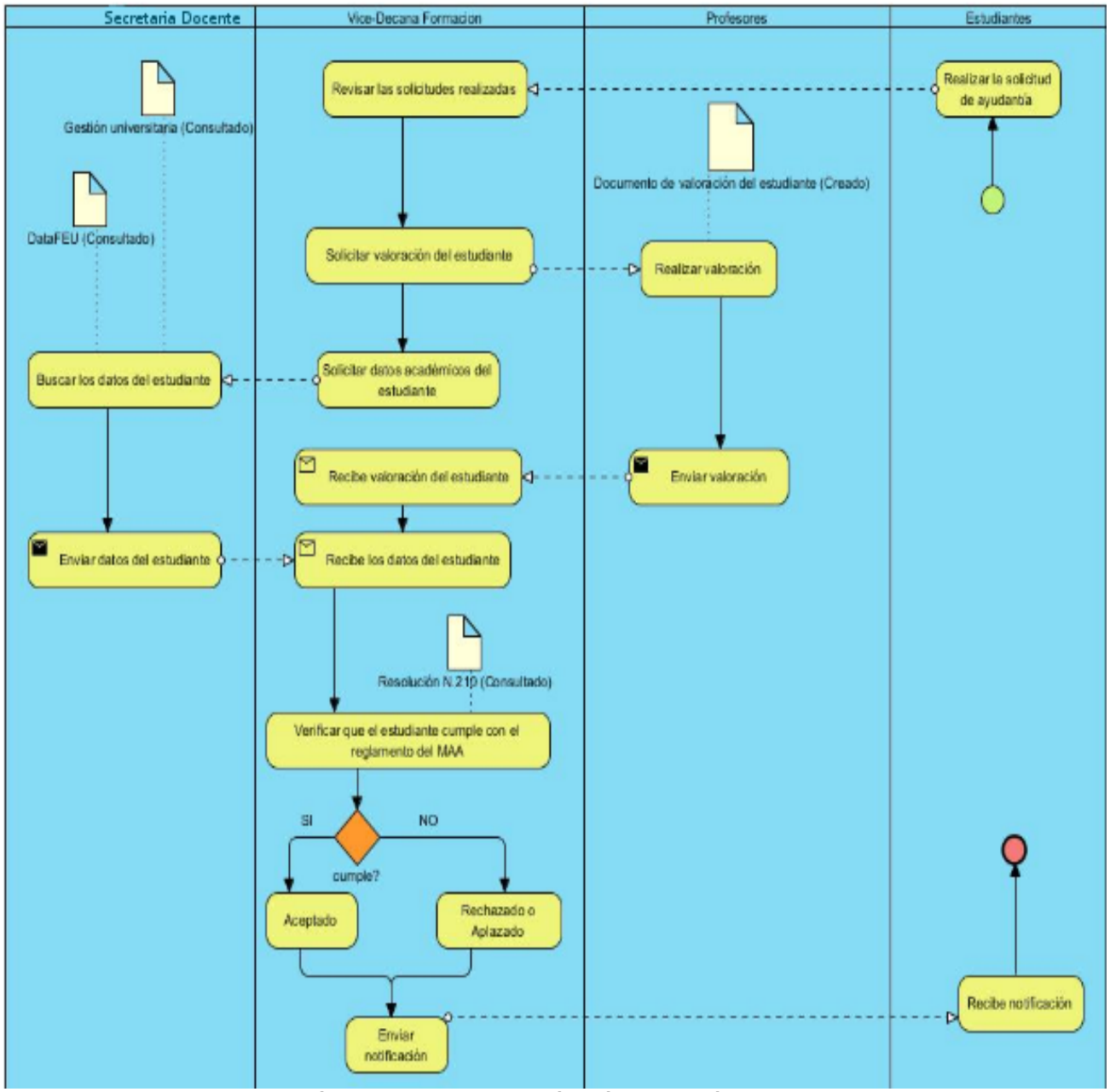


Diagrama 2. Proceso aprobar alumno ayudante

Proceso Dar Baja a un Alumno Ayudante

Este proceso puede ser producto a dos posibles casos: Baja voluntaria y Baja por pérdida de requisitos. El primero se produce debido a una decisión propia del alumno por motivos personales; el segundo, sucede cuando un alumno ayudante deja de cumplir con los requisitos planteados en los estatutos del Movimiento y es necesario separarlo del mismo. En cualquiera de los dos casos, la baja del estudiante queda registrada en un documento físico, Word u hoja Excel, confeccionado por el Vicedecano de Formación de la Facultad.

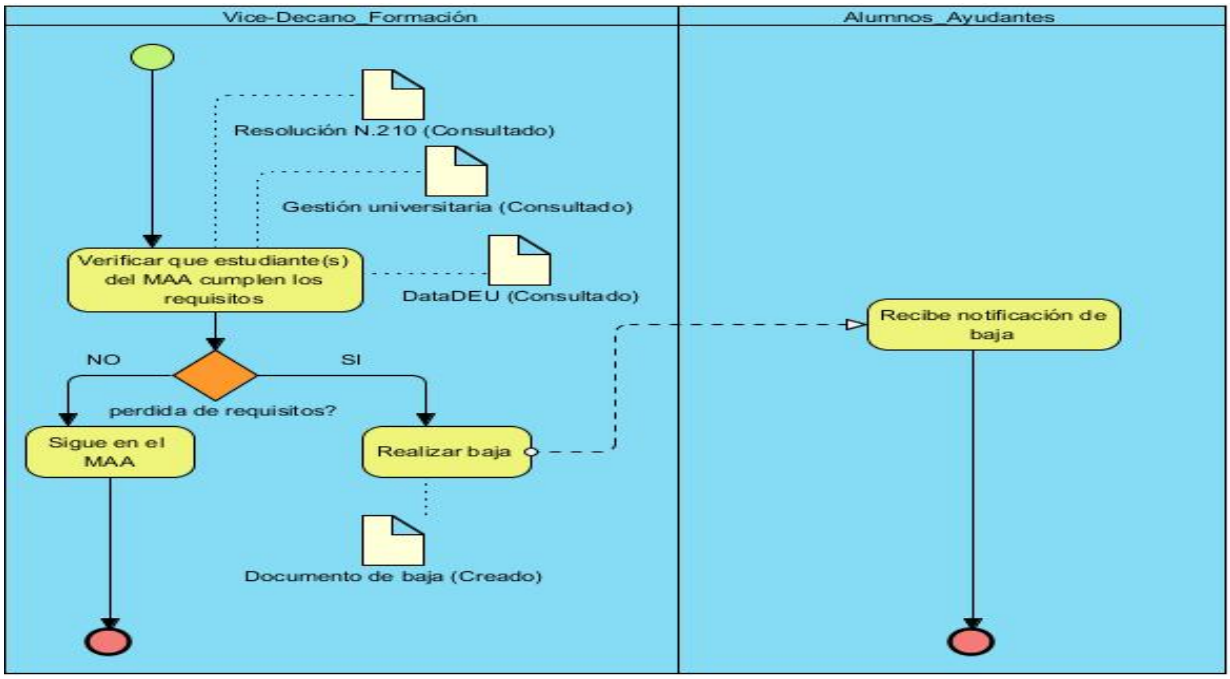


Diagrama 3. Proceso alumno ayudante causa baja por pérdida de requisitos

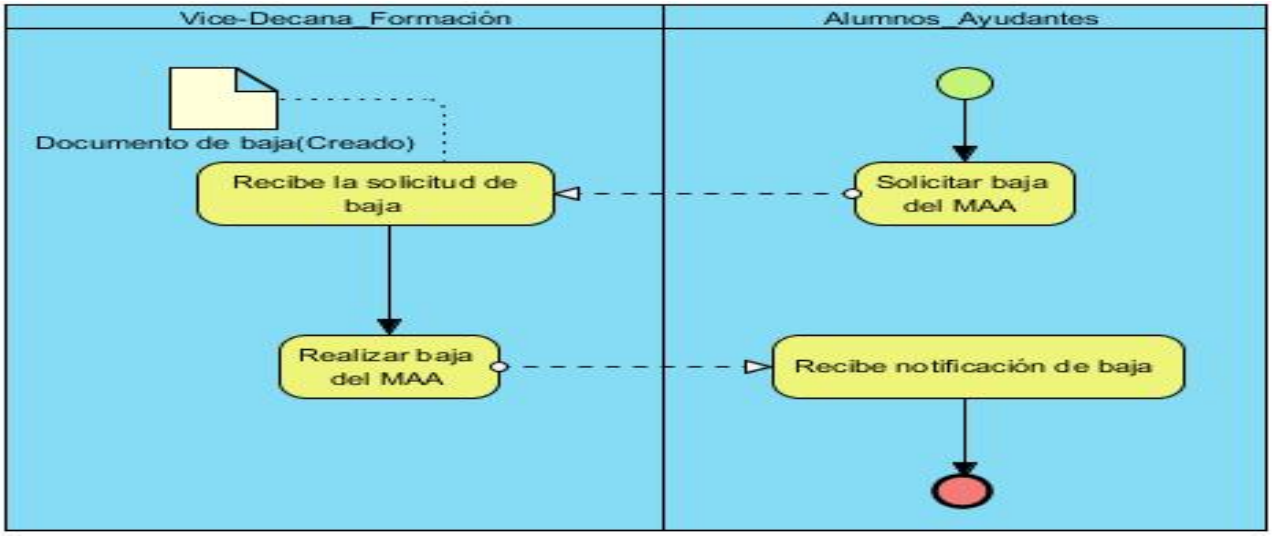


Diagrama 4. Proceso alumno ayudante solicita baja del MAA por motivos propios

Proceso Evaluar Alumno Ayudante

En este proceso el profesor tutor orienta las actividades que deben cumplir sus alumnos ayudantes mediante un Plan de Trabajo semestral, el cual es analizado y aprobado en conjunto con el estudiante. Una vez concluido el semestre el profesor tutor evaluará el cumplimiento del mismo a partir de las tareas realizadas durante este período. Emitirá un documento al concluir el curso donde quede reflejada esta evaluación y las tareas orientadas ejecutadas o no.

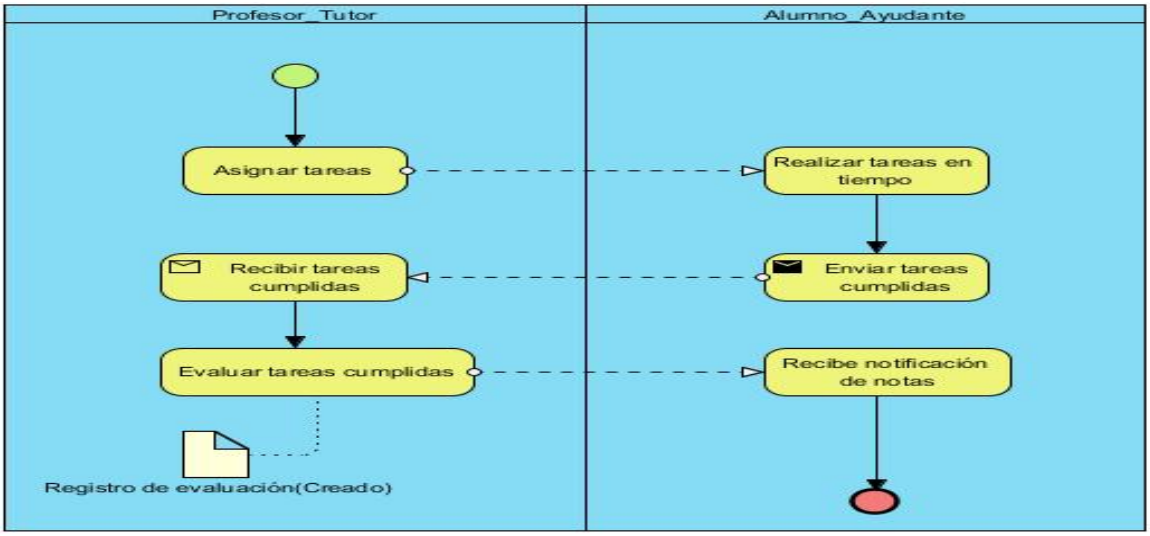


Diagrama 5. Proceso evaluar al alumno ayudante

2.2. Descripción de la solución propuesta

Con el fin de resolver las deficiencias descritas en la situación problemática, se propone realizar un sistema informático web que gestione la información de los procesos asociados al MAA. El mismo permitirá:

- A los estudiantes: realizar y modificar solicitudes de ayudantía de acuerdo a las convocatorias publicadas; ver las tareas que le son asignadas y la evaluación obtenida una vez realizadas.
- A los profesores tutores: además de asignar y evaluar tareas a sus alumnos ayudantes les permitirá proponer la entrada y salida de algún estudiante al MAA y generar la evaluación anual y el plan de trabajo.
- A los Jefes de Departamentos: además de las funcionalidades de un profesor les permitirá publicar las necesidades de alumnos ayudantes de sus departamentos, gestionar las asignaturas y los profesores de sus departamentos.
- Al Vicedecano de Formación: le permitirá gestionar las publicaciones de las convocatorias, aceptar o denegar las solicitudes, dar baja a algún alumno ayudante, gestionar las asignaturas, los departamentos docentes, los Jefes de Departamentos y los grupos docentes. La aplicación contará con un sistema de alertas para notificar al estudiante si ha sido aceptado como AA, si ha sido dado de baja, si se le asignó alguna y cuando esta está por terminar. Permitirá, además, generar el listado de los alumnos ayudantes.

2.3. Planeación

La planeación es la etapa inicial de todo proyecto en XP. Durante este punto se interactúa con el cliente y el resto del equipo de desarrollo para explorar e identificar los requisitos del usuario en historias de usuarios y los requerimientos del sistema, así como la familiarización del equipo de trabajo con las herramientas y tecnologías seleccionadas para la implementación del sistema. Se identifican el número y el tamaño de las iteraciones, el cliente establece la prioridad de cada historia de usuario, y correspondientemente los programadores realizan la estimación del esfuerzo que costará implementar cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. (Letelier y Penades, 2006)

2.3.1. Requisitos del usuario

Son declaraciones de los servicios y funciones que proveerá el sistema. Estos requerimientos dependen del tipo de software que se desarrolle, de los posibles usuarios del software y del enfoque general tomado por la organización al redactar requerimientos. Estos requisitos al tiempo que avanza el proyecto de software se convierten en los algoritmos, la lógica y gran parte del código del sistema. (Letelier y Penades, 2006)

A continuación se muestra un listado de los requisitos definidos para la aplicación, son descritos más adelante en las Historias de Usuario (HU), forma en la que se especifican en la metodología XP:

- HU1 Gestionar Departamentos.
- HU2 Gestionar Jefes de Departamentos.
- HU3 Gestionar Tutor.
- HU4 Gestionar Asignaturas.
- HU5 Gestionar Semestre.
- HU6 Gestionar Grupos.
- HU7 Gestionar Alumnos Ayudantes.
- HU8 Gestionar Convocatoria.
- HU9 Gestionar Solicitud de AA.
- HU10 Gestionar Aprobación de solicitud.
- HU11 Gestionar Evaluaciones AA.
- HU12 Gestionar Actividades de los AA.
- HU13 Realizar Reportes.
- HU14 Gestionar Necesidades de AA.
- HU15 Notificaciones.
- HU16 Proponer Estudiante para Ayudantía.
- HU17 Autenticar Usuario

2.3.2. Requisitos del sistema

Son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o

confiable, por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares. Son fundamentales en el éxito del producto y normalmente están vinculados a requerimientos funcionales. (Sommerville, 2005)

A continuación los requisitos del sistema identificados:

Interfaz

RS1 La interfaz debe tener una fuente apropiada, establecer contraste entre el color de las interfaces y la fuente para lograr una buena lectura. El sistema contará con dos interfaces principales: la interfaz de entrada al sistema y la interfaz de navegación. En ambas interfaces deben predominar los colores claros como el azul, el blanco y el gris, mientras la fuente debe ser de color negro.

Usabilidad

RS2 La aplicación debe tener en todas sus interfaces el idioma español.

RS3 La aplicación debe presentar una interfaz de autenticación fácil para que los usuarios puedan acceder sin problemas.

RS4 Las interfaces deben ser amigables, sencillas para que los usuarios puedan interactuar con ella sin dificultad.

Rendimiento

RS5 La aplicación debe ser eficiente en respuesta a una solicitud realizada, permitiendo alcanzar los resultados deseados sin hacer uso extensivo en la navegación del sitio.

Soporte

RS6 El código de la aplicación debe ser comprensible, de esta forma el sistema resultará fácil de entender y mantener en caso de posibles fallos.

Portabilidad

RS7 La aplicación debe ser multiplataforma.

RS8 La aplicación debe ser compatible con varios navegadores web: *Firefox* en su versión 24 o superior, *Chrome* en su versión 31 o superior e Internet *Explorer* en su versión 9 o superior.

Seguridad

RS9 La aplicación debe contar con una política de seguridad o sistemas de roles para cada usuario diferente que interviene en el proceso en dependencia del nivel jerárquico que cumpla su rol. El cual solo les permitirá acceder a las funcionalidades permitidas de acuerdo al rol que desempeñe dentro del mismo.

RS10 La aplicación debe permitir su acceso mediante la autenticación, consumiendo los servicios Ldap.

Software

RS11 La aplicación debe permitir el acceso desde un navegador, sin importar el sistema operativo ya que será desarrollada con tecnologías y herramientas multiplataforma. Para montar el servidor de la aplicación se propone hacerlo desde una PC con *Ubuntu* 12.04 como sistema operativo.

Hardware

RS12 La PC cliente debe contar con 512 Mb de RAM y 20 Gb de disco duro como mínimo e instalado un navegador web para la utilización del sistema.

RS13 El servidor en el que se montará la aplicación debe contar con 2 Gb de RAM y 80 Gb de disco duro como mínimo.

2.3.3. Definición de roles

Los actores del negocio son aquellas personas que interactúan con el negocio para beneficiarse de sus resultados.

Actores que intervienen en el negocio	Rol que desempeñan en el sistema
Vice-Decano de Formación	Representa al Vicedecano de Formación de la Facultad (administrador del sistema) posee todos los privilegios y el dominio total del sistema. Puede gestionar la convocatoria, los alumnos ayudantes, los departamentos, los jefes de departamentos, las asignaturas y los grupos, es quien aprueba o deniega la solicitud de ayudantía.
Jefe de departamento	Representa al Jefe de Departamento de la Facultad. Gestiona las necesidades de alumnos ayudantes de cada asignatura de su departamento, adiciona los profesores de su departamento, además puede realizar las funciones del Profesor.
Profesor	Representa al Profesor Tutor de la facultad, puede planificar y evaluar las tareas de sus alumnos ayudantes, solicitar la baja de un alumno ayudante, generar el plan de trabajo y la evaluación anual del AA y proponer a un estudiante para la ayudantía.
Alumno ayudante	Representa al alumno ayudante de la Facultad que podrá ver sus evaluaciones, su plan de actividades, puede ver las convocatorias publicadas en caso de que desee solicitar cambio de asignatura, solicitar su baja del MAA.
Estudiante	Representa un estudiante de la Facultad, con acceso solo a ver la convocatoria publicada, gestionar la ayudantía en caso deseado y esperar la notificación.

Tabla 2 Actores que intervienen en el negocio

2.3.4. Historias de Usuario

Las HU son las técnicas utilizadas por la metodología XP para representar y dar a conocer los requerimientos que proveerá el software al equipo de desarrollo. El tratamiento de las historias de usuario es muy dinámico y flexible, deben ser fichas escritas con un leguaje preciso y entendible que enuncie brevemente las características que debe cumplir el sistema. Estas HU pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas a lo largo del ciclo de vida del proyecto y deben ser definidas por iteraciones. Son usadas para estimar tiempos de desarrollo de la parte de la aplicación que describen. También se utilizan en la fase de pruebas, para verificar si el programa cumple con lo especificado. (Beck y Andres, 2005; Letelier y Penades, 2006)

Respecto de la información contenida en la historia de usuario, existen varias plantillas sugeridas pero no hay un consenso al respecto. La presente investigación asume la presentada por Beck[15] en su libro *Extreme Programming Explained* (Beck y Andres, 2005) con los siguientes aspectos:

Número: número asignado a la HU.

Nombre de HU: nombre de la HU.

Usuario: usuario del sistema que utiliza la HU.

Prioridad en el negocio: nivel de prioridad de la HU en el negocio.

Riesgo de desarrollo: nivel de riesgo si no se realiza la HU.

Puntos estimados: estimación que hace el equipo de desarrollo del tiempo necesario para implementar la HU. El 1 equivale a una semana ideal de trabajo -5 días hábiles trabajando 40 horas; o sea, 8 horas diarias-. Cuando el valor es 0.5 equivale a 2 días y medio de trabajo, 20 horas en total.

Puntos reales: el tiempo real en el que se realizó la HU.

Descripción: breve descripción de lo que realizará la HU.

A continuación una representación de las historias de usuario que se identificaron. Las demás se encuentran en el ANEXO 2

Historia de usuario	
N.: 1	**Nombre:** Gestionar Departamento
Usuario: Vicedecano de Formación	
Prioridad en el Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración Asignada: 1	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Vicedecano de Formación, modificar, eliminar y buscar un Departamento en el sistema.	
Observación: No hay	

Tabla 3 HU Gestionar Departameno.

15 **Kent Beck:** uno de los creadores de las metodologías de desarrollo de software de programación extrema

Historia de usuario	
N.: 7	**Nombre:** Gestionar AA
Usuario: Vicedecano de Formación	
Prioridad en el Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración Asignada: 1	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Vicedecano de Formación adicionar, modificar y eliminar un alumno ayudante en el sistema.	
Observación:	

Tabla 4 HU Gestionar AA

Historia de usuario	
N.: 11	**Nombre:** Gestionar solicitud de AA
Usuario: Estudiante	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 2	**Esfuerzo Estimado:** 1
Descripción: Permitirá al rol Estudiante realizar, modificar y cancelar su solicitud de ayudantía.	
Observación:	

Tabla 5 HU Gestionar solicitud de AA

Historia de usuario	
N.: 14	**Nombre:** Evaluar AA
Usuario: Profesor**,** Jefe de Departameno.	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 3	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Profesor y Jefe de Departamento evaluar a un AA, permitiéndole además modificar dicha evaluación.	
Observación: El profesor debe guiarse y mantener las pautas del modelo de evaluación de un AA.	

Tabla 6 HU Evaluar AA

2.3.5. Estimación de esfuerzo por Historia de Usuario

Para llevar a cabo el desarrollo del sistema es necesario estimar el esfuerzo para cada historia de usuario identificada. Esta estimación incluye todo el esfuerzo asociado a la implementación de las HU. (Beck y Andres, 2005)

En la siguiente tabla se muestra la estimación del esfuerzo para cada una de las historias de usuario.

Historias de Usuarios	Estimación
Gestionar Departamento	0.5
Gestionar Jefe de Departamento	0.5
Gestionar Tutor	0.5
Gestionar Asignaturas	0.5
Gestionar Semestre	0.5
Gestionar Grupos	0.5
Gestionar AA	0.5
Necesidad de AA	0.5
Convocatoria de AA	0.5
Gestionar Solicitud de AA	1
Aprobar Solicitud de AA	1
Proponer Estudiante para Ayudantía	0.5
Evaluar AA	0.5
Notificaciones	0.5
Actividades de los AA	1
Reportes	1
Autenticar Usuario	0.5

Tabla 7 Estimación del esfuerzo

2.4. Plan de publicaciones

Una vez identificadas las HU y estimado el esfuerzo para cada una de ellas, el equipo de desarrollo determina junto con el cliente la planificación de la fase de implementación. Esta planificación refleja en cuántas iteraciones el grupo de trabajo realizará su labor. Al final de cada iteración se obtiene un producto que luego se muestra al usuario para verificar que lo desarrollado hasta el momento es lo que él desea.

2.4.1. Plan de iteraciones

Para la implementación de esta aplicación se han definido tres iteraciones. En la **primera iteración** se implementarán las HU con prioridad media, la realización de las mismas va dando idea de cómo quedará la aplicación aunque todavía estará en sus inicios. En la **segunda iteración** se implementarán las HU con prioridad de negocio alta dando fin a algunas de las funcionalidades más críticas. En la **tercera iteración,** luego de haberse implementado las funcionalidades fijadas en iteraciones anteriores, se elaboran las HU que dan fin al desarrollo de la aplicación. ANEXO 3.

2.4.2. Plan de duración de iteraciones

Este plan se encarga de mostrar las historias de usuario que serán implementadas en cada una de las iteraciones, así como la duración estimada de cada una y el orden en que se implementarán. Esto dará paso a determinar el plan de entrega final de la aplicación. A continuación se muestran los resultados.

Iteraciones	Historias de usuarios a implementar	Duración de las Iteraciones
Iteración 1	Gestionar Departamento Gestionar Jefe de Departamento Gestionar Asignaturas Gestionar Semestre Gestionar Grupos Gestionar AA Gestionar Tutor	3.5 semanas
Iteración 2	Necesidad de AA Convocatoria de AA Gestionar Solicitud de AA Aprobar Solicitud de AA Proponer Estudiante para Ayudantía	3.5 semanas
Iteración 3	Actividades de los AA Evaluar AA Notificaciones Reportes Autenticar Usuario	3.5 semanas

Tabla 8 Plan de duración de las iteraciones

2.4.3. Plan de Entregas

Plan de entregas ideado para la fase de implementación. Con el plan de entrega se muestra al usuario una aproximación de cuándo recibirá cada una de las versiones de la herramienta, teniendo en cuenta que la implementación comienza en el mes de marzo. En la **Tabla 9** se muestra el Plan de Entrega de la aplicación.

Sistema	Final 1ra iteración 4ta semana de marzo	Final 2da iteración 4ta semana de abril	Final 3ra iteración 4ta semana de mayo
Gestión de información de los Alumnos Ayudantes en la Facultad 7	**Versión 0.1**	**Versión 1.0**	**Versión 1.1**

Tabla 9 Plan de entrega

2.5. Diseño

El diseño crea una estructura que organiza la lógica del sistema, un buen diseño permite que el sistema crezca, pero la metodología XP sugiere que hay que conseguir diseños simples y sencillos. Es por ello que se debe hacer todo lo menos complicado posible para conseguir un diseño entendible e implementable que a la larga costará menos tiempo y esfuerzo a desarrollar. Entre los elementos más importantes que menciona XP referentes al diseño se encuentran las tarjetas CRC. (Beck y Andres, 2005)

2.5.1. Tarjetas Clase-Responsabilidad-Colaborador

La principal funcionalidad de las tarjetas CRC es ayudar a dejar los hábitos de la programación procedural clásica para centrarse en el enfoque orientado a objetos. En esta etapa se busca describir las responsabilidades que tiene que cumplir cada clase y las colaboraciones entre ellas para poder implementar las historias de usuario. A medida que se van obteniendo las responsabilidades y las colaboraciones se anotan en la tarjeta CRC. (Beck y Andres, 2005)

A continuación una representación de las tarjetas CRC definidas en el sistema, las demás se encuentran en el ANEXO 4.

CRC Aa	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla aa de la Base Datos responsable de manejar datos referente a la lista de alumnos ayudantes con los que cuenta el sistema.	Grupo ProfesorTutor Curso Semestre Asignaturas Evaluaciones SolicitudBaja

Tabla 10 CRC Aa

2.6. Patrones

2.6.1. Patrón arquitectónico

El **Modelo Vista Controlador (MVC)**: es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador.

El **Modelo**: es la representación de la información con la cual el sistema opera, gestiona todos los accesos a dicha información, tanto consultas como actualizaciones (lógica de negocio). Envía a la 'vista' aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al 'modelo' a través del 'controlador'.

El **Controlador**: responde a eventos, usualmente acciones que el usuario invoca, implica cambios en el 'modelo' cuando se hace alguna solicitud sobre la información y también puede enviar comandos a su 'vista' asociada, el 'controlador' hace de intermediario entre la 'vista' y el 'modelo'.

La **Vista**: presenta el 'modelo' (información y lógica de negocio) en un formato adecuado para interactuar con el usuario (usualmente la interfaz de usuario) por tanto requiere de dicho 'modelo' la información que debe representar como salida.

Principales ventajas del MVC:

- Clara separación entre el modelo, la vista y el controlador.
- Facilidad para la realización de pruebas unitarias de los componentes.
- Reutilización de los componentes.
- Simplicidad en el mantenimiento de los sistemas. (González y Romero, 2012)

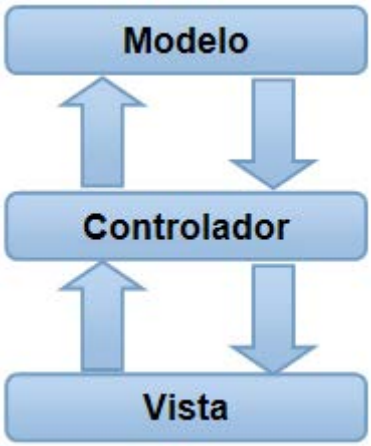


Figura 1 ***Patrón arquitectónico MVC***

2.6.2. Patrones de diseño

Los patrones de diseño son aquellos que expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software. Un patrón de diseño identifica clases, instancias, roles, colaboraciones y distribución de responsabilidades. Es la solución estándar para un problema común de programación. Permite un lenguaje de programación a alto nivel. Es la forma práctica de describir aspectos de la organización de un programa.

Patrones GRASP (dado sus siglas en inglés *General Responsibility Assignment Software Patterns*) entre los que se encuentran:

Experto: determina cuál es la clase que debe asumir una responsabilidad a partir de la información que posee cada una. Le asigna una responsabilidad al experto en información, o sea es la clase que cuenta con la información necesaria para cumplir la responsabilidad.

Este patrón se evidencia en la definición de las clases de acuerdo a las funcionalidades que deben realizar a partir de la información manejada dentro del componente, como por ejemplo las clases controladoras.

```php
class ActividadesController extends Controller {

    /**
     * Lists all Actividades entities.
     *
     * @Route("/", name="actividades")
     * @Method("GET")
     * @Template()
     */
    public function indexAction() {
        $em = $this->getDoctrine()->getManager();

        $entities = $em->getRepository('SISGIAA1Bundle:Actividades')->findAll();
        $entity = new Actividades();
        $usuario = $this->get('security.context')->getToken()->getUser();
        $rol = 'ProfesorTutor';
        if ($this->get('security.context')->isGranted('ROLE_Jefe de Departamento')) {
            $rol = 'JDpto';
        }
```

Figura 2 ***Patrón experto***

Creador: es el responsable de crear una nueva instancia de alguna clase. Le asigna a una clase B la responsabilidad de crear las instancias de una clase A.

En la clase controladora AaController.php se encuentra definido el objeto $ldap = new \SISGIAA1\SISGIAA1Bundle\Util\Ldap(); de las clases LDAP.php, evidenciando de este modo que la clase AaController.php es "creador" del objeto de la LDAP.php.

```php
    public function createAction(Request $request) {
        $em = $this->getDoctrine()->getManager();
        $ldap = new \SISGIAA1\SISGIAA1Bundle\Util\Ldap();
        $entity = new Aa();
        $form = $this->createForm(new AaType(), $entity);
        $form->bind($request);

        $user = $entity->getUsuario();
        $comprobar = $ldap->findAction('user', $user);
       // print_r($comprobar);exit;
```

Figura 3 ***Patrón Creador***

Bajo acoplamiento: Medida de la fuerza con que una clase está conectada a otras clases. Resuelve cómo dar soporte a una dependencia escasa y a un aumento de la reutilización. Asigna una responsabilidad para mantener bajo acoplamiento.

Este patrón es evidenciado en cada uno de los módulos del sistema, cada clase controladora maneja la información de las clases entidades que están relacionadas con las funcionalidades que se especifican en dicha clase controladora para lograr un bajo acoplamiento de clases.

```php
class EvaluacionesController extends Controller
{
    /**
     * Lists all Evaluaciones entities.
     *
     * @Route("/", name="evaluaciones")
     * @Method("GET")
     * @Template()
     */
    public function indexAction()
    {
        $em = $this->getDoctrine()->getManager();

        $entities = $em->getRepository('SISGIAA1Bundle:Evaluaciones')->findAll();

        return array(
            'entities' => $entities,
        );
    }
```

Figura 4 ***Patrón bajo acoplamiento***

```php
class Evaluaciones
{
    /**
     * @var integer
     *
     * @ORM\Column(name="id", type="integer", nullable=false)
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="IDENTITY")
     */
    private $id;

    /**
     * @var string
     *
     * @ORM\Column(name="evaluacion", type="string", length=10, nullable=false)
     */
    private $evaluacion;
```

Figura 5 ***Patrón bajo acoplamiento***

Patrones GoF (dado sus siglas en inglés *Gang of Four*) se descubren como una forma indispensable de enfrentarse a la programación.

Decorador: permite agregar responsabilidad a un objeto de forma dinámica. (Díaz *et al,*. 2005)

Symfony lo implementa a través de la utilización de una plantilla global que en este caso es la plantilla base, que almacena el código HTML que es común a todas las páginas de la aplicación.

```
{% extends '::base.html.twig' %}

{% block body -%}
<div id="site_content">
    <h1>Listado de Alumnos Ayudantes</h1>
```

Figura 6 ***Decorador.***

2.7. Modelo de datos

Los modelos de datos determinan la estructura de la información, con el objetivo de mejorar la comunicación y la precisión en aplicaciones que usan e intercambian datos. Los modelos de datos son esenciales para el desarrollo de sistemas de información, ya que a través de ellos puede conseguirse la compatibilidad necesaria para manejar cantidades colosales de datos. (Elmasr*i et al.*, 2002)

2.7.1. Modelos de Datos basados en Objetos

Entidad-relación: datos organizados en conjuntos interrelacionados de objetos (entidades) con atributos asociados. Se denomina así debido a que precisamente permite representar relaciones entre entidades (objetivo del modelado de datos). El modelo debe estar compuesto por:

- Entidades
- Atributos
- Relaciones
- Cardinalidad
- Llaves (Elmasr*i et al.*, 2002)

El diagrama de Entidad-Relación de la base de datos realizado en el presente trabajo se encuentra en el ANEXO 5.

En este capítulo se describieron las principales características del sistema a través de los artefactos generados por la metodología XP. Se realizó la descripción y modelado de los procesos a automatizar. Se realizó el levantamiento de requisitos del software a través de las HU, se estimó el esfuerzo necesario para la implementación de cada HU y se confeccionó el plan de iteraciones en el orden en que estas HU fueron implementadas de acuerdo a la prioridad establecida entre ellas y el plan de entrega del sistema. Se capturaron los requerimientos no funcionales. Se construyó el modelo de la base de datos y se definió la arquitectura del sistema bajo el patrón MVC.

Capítulo 3: Implementación y prueba del sistema para la gestión de información de los Alumnos Ayudantes en la Facultad 2

En este capítulo se abordan las especificaciones de la fase Implementación. Se definen los estándares de codificación a utilizar, las tareas de ingeniería y se muestra el diagrama de despliegue. Además, se definen las pruebas a aplicar al sistema para demostrar su correcto funcionamiento.

3.1. Estándares de codificación

El propósito fundamental de los estándares de codificación es que el proyecto tenga una arquitectura y un estilo consistente, de forma tal que resulte fácil de entender y más fácil de mantener. El mayor problema que trata de enfrentar esta práctica, es el de intentar entender el formato y el estilo utilizado en el código escrito por otros desarrolladores.

Es importante tener un buen estándar de codificación, porque de esta forma se deberá:

- Clarificar más que confundir.
- Promover la intención del código.
- Permitir que los programas se acerquen lo mejor posible al lenguaje natural.
- Incorporar las mejores prácticas de la codificación. (PHP-FIG)

Los estándares definidos para la implementación de este proyecto fueron los establecidos por los creadores del *framework* Symfony 2 que siguen los estándares PSR-0, PSR-1 y PSR-2. Los mismos plantean:

- Añadir un solo espacio después de cada delimitador coma.
- Añadir un solo espacio alrededor de los operadores (==, &&).
- Definir una clase por archivo.
- Añadir una coma después de cada elemento del arreglo en un arreglo multilínea, incluso después del último.
- Añadir una línea en blanco antes de las declaraciones *return*, a menos que el valor devuelto solo sea dentro de un grupo de declaraciones (tal como una declaración *if*).

- Usar llaves para indicar la estructura del cuerpo de control, independientemente del número de declaraciones que contenga;
- Declarar las propiedades de clase antes que los métodos.
- Declarar primero los métodos públicos, luego los protegidos y finalmente los privados.
- Utilizar paréntesis al crear instancias de clases, independientemente del número de argumentos que el constructor tenga.
- Utilizar la técnica *lowerCamelCase* para escribir el nombre de métodos y funciones.
- Sufijar las interfaces con Interface.
- Sufijar las características con *Trait*.
- Sufijar las excepciones con *Exception*.
- Utilizar *namespaces* para todas las clases.

Ejemplo:

```
public function showAction($id)
    {
        $em = $this->getDoctrine()->getManager();
        $entity = $em->getRepository('SISGIAA1Bundle:Rol')->find($id);
        if (!$entity) {
            throw $this->createNotFoundException('Unable to find Rol entity.');
        }
        $deleteForm = $this->createDeleteForm($id);
        return array('entity' => $entity, 'delete_form' => $deleteForm->createView(),
        );
    }
```

3.2. Tareas de Ingeniería

Las HU se descomponen, cada una, en varias tareas de ingeniería que serán desarrolladas por el equipo de trabajo aplicando la práctica de la programación en parejas. Estas tareas son para el uso estricto de los programadores, por lo que pueden ser escritas en lenguaje técnico y no necesariamente entendible por el cliente. A continuación una representación de las tareas de programación definidas, el resto se encuentra en el ANEXO 6.

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU1_Gestionar Departamento
Nombre Tarea: Insertar departamento	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 28/02/2014	**Fecha Fin:** 01/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan la inserción de los datos de un departamento.	

Tabla 11 Insertar Departamento

Tarea de Ingeniería	
Nro. Tarea: 2	**HU**: HU1_Gestionar Departamento
Nombre Tarea: Eliminar un departamento	
Tipo de Tarea: Desarrollada	**Puntos estimados**: 0.5
Fecha Inicio: 01/03/2014	**Fecha Fin**: 02/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan eliminar al departamento seleccionado teniendo en cuenta que para eliminarlo debe existir.	

Tabla 12 Eliminar Departamento

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU1_Gestionar Departamento
Nombre Tarea: Modificar un departamento	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 02/03/2014	**Fecha Fin:** 03/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan modificar el departamento seleccionado.	

Tabla 13 Modificar Departamento

3.3. Pruebas

El instrumento adecuado para determinar el status de la calidad de un producto de software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas tanto a los componentes del software como al sistema en su totalidad, con el objetivo de medir el grado en que este cumple con los requerimientos. (Sommerville, 2005)

Pruebas (test): «una actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto» (Sommerville, 2005)

La metodología XP se basa en las pruebas unitarias y de aceptación. Las pruebas unitarias verifican el funcionamiento de una clase y son ejecutadas por los programadores; las pruebas de aceptación validan el cumplimiento de las funcionalidades del software y son realizadas por el cliente a partir de las HU.

Un sistema se dice completamente aceptable si quedan satisfechos todos los requisitos funcionales especificados por el cliente, teniendo en cuenta además los requisitos no funcionales, así como los distintos recursos de dicho módulo (Joskowicz, 2008).

3.3.1. Pruebas de aceptación

En la presente investigación se realizaron pruebas de aceptación. Para ello se definieron 57 casos de prueba. La **Tabla 15** muestra los casos de pruebas de aceptación para la HU Gestionar Departamento, el resto de los casos de prueba se encuentran en el ANEXO 7.

Caso de Prueba de Aceptación	
Código: HU1_P1	**Historia de Usuario:** HU1_Gestionar Departamento
Nombre: Insertar un departamento	
Descripción: Probar que se puedan insertar los departamentos en la BD correctamente.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Vicedecano de Formación Insertar datos válidos del departamento Llenar todos los campos de la inserción El departamento no debe existir en la Base de datos	
Entrada / Pasos de Ejecución: Se inserta un departamento con datos válidos	
Resultado Esperado: El departamento se inserte en la base de datos	
Evaluación de la Prueba: Satisfactoria	

Caso de Prueba de Aceptación	
Código: HU1_P2	**Historia de Usuario:** HU1_Gestionar Departamento
Nombre: Eliminar un departamento	
Descripción: Probar que se puedan eliminar los departamentos de la BD.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Vicedecano de Formación El departamento a eliminar debe existir en la Base de datos	
Entrada / Pasos de Ejecución: Se selecciona un departamento a eliminar	
Resultado Esperado: El departamento se elimine de la base de datos correctamente.	
Evaluación de la Prueba: Satisfactoria	

Caso de Prueba de Aceptación	
Código: HU1_P3	**Historia de Usuario:** HU1_Gestionar Departamento
Nombre: Modificar un departamento	
Descripción: Probar que se puedan modificar los departamentos en la BD.	

Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Vicedecano de Formación El departamento a modificar debe existir en la Base de datos Llenar los campos a modificar
Entrada / Pasos de Ejecución: Se selecciona un departamento a modificar
Resultado Esperado: Se modifique el departamento en la base de datos correctamente.
Evaluación de la Prueba: Satisfactoria

Tabla 14 PA Gestionar Departamento (1ra iteración)

3.4. Resultados de las pruebas

A partir de los siguientes datos se realizaron las pruebas al sistema:

- 17 Historias de Usuario.
- 57 Tareas de Ingeniería.
- 57 Casos de Prueba de Aceptación.

La aplicación de las pruebas de aceptación se realizó siguiendo lo establecido por la metodología XP de forma iterativa para comprobar la correcta implementación de las historias de usuario. Se aplicaron un total de 57 casos de prueba distribuidos en 3 iteraciones. A continuación se refleja en la *Figura 7*, la cantidad total de funcionalidades probadas y las no conformidades obtenidas por cada iteración.

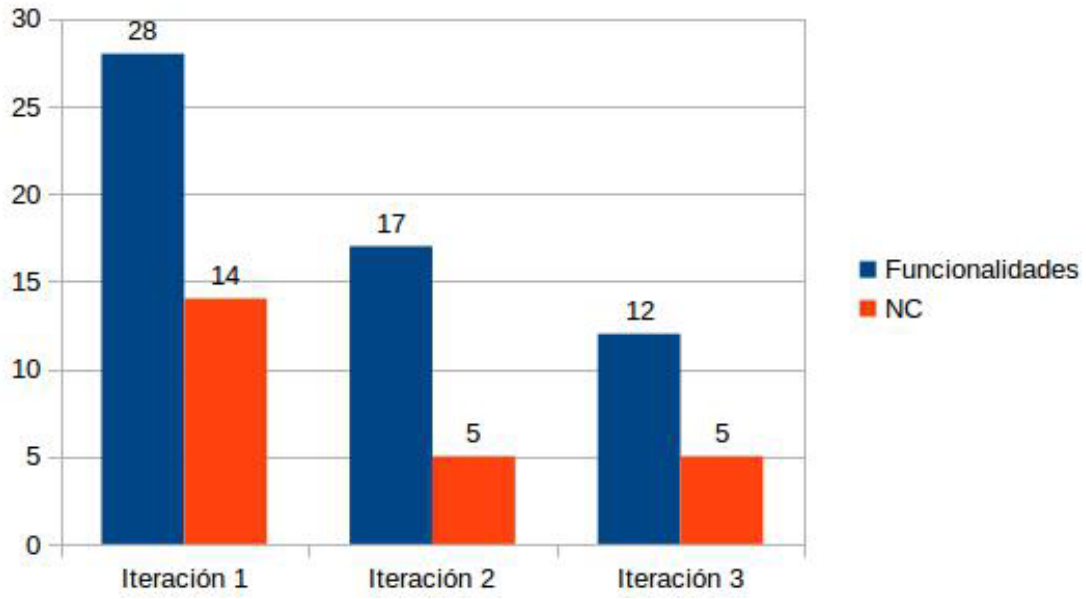


Figura 7 ***Resultados de las pruebas de aceptación***

A continuación se describen por iteración los errores detectados:

Iteración 1

Se comprobaron 28 funcionalidades, detectándose las siguientes 14 no conformidades:

5 errores ortográficos

5 errores de interfaz

4 errores de validación de campos

Iteración 2

Se comprobaron 17 funcionalidades, detectándose las siguientes 5 no conformidades:

1 error de interfaz

2 errores ortográficos

2 errores de validación de campos

Iteración 3

Se comprobaron 12 funcionalidades, detectándose las siguientes 5 no conformidades:

5 errores de interfaz

3.5. Diagrama de Despliegue

El Diagrama de Despliegue provee un modelo detallado de la forma en la que los componentes se desplegarán a lo largo de la infraestructura del sistema. Detalla las capacidades de red, las especificaciones del servidor, los requisitos de *hardware* y otra información relacionada al despliegue del sistema propuesto. Es un mapa específico de la instalación física del sistema. (Booc*h et al.*, 1999)

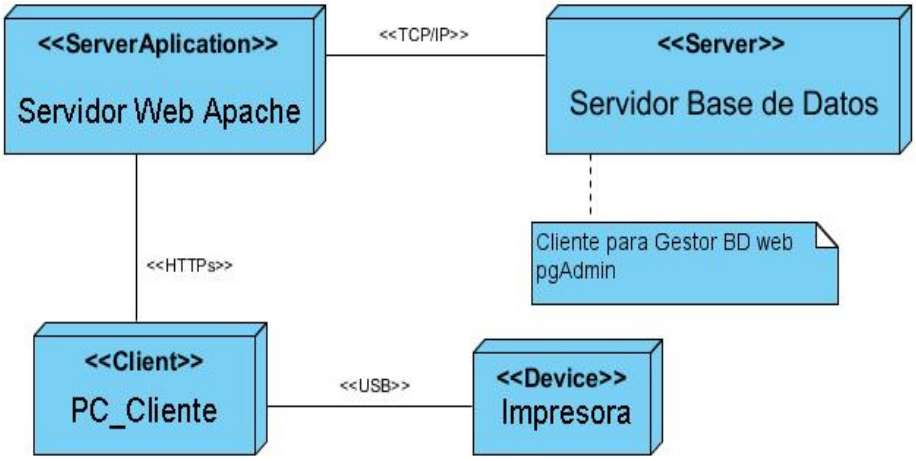


Diagrama 6. Diagrama de despliegue

En el presente capítulo se especificaron los elementos relacionados con la implementación del sistema: estándares de codificación y tareas de ingeniería; y la descripción de las pruebas aplicadas al sistema. Se mostró, además, el diagrama de despliegue que indica la situación física de los componentes desarrollados para la instalación del sistema.

Conclusiones Generales

Después de realizado el estudio de los sistemas de gestión de información académica existentes a nivel nacional e internacional, se lograron establecer los elementos teóricos que guiaron el desarrollo de la aplicación web “Sistema de Gestión de la Información de Alumnos Ayudantes de la Facultad 2”.

A partir del estudio de diferentes metodologías, herramientas y lenguajes de programación se definieron las más convenientes para el desarrollo de la aplicación web “Sistema de Gestión de la Información de Alumnos Ayudantes de la Facultad 2”.

En la especificación de los requisitos de usuario a través de las HU se obtuvieron los artefactos necesarios para el desarrollo de la aplicación web Sistema de Gestión de la Información de Alumnos Ayudantes de la Facultad 2.

Se logró la implementación de una aplicación web que gestiona la información de los procesos asociados al Movimiento de Alumnos Ayudantes de la Facultad 2 de la UCI, que apoya la toma de decisiones de los profesores en los procesos de evaluación, permitiendo organizar y mantener el control de la información de los estudiantes vinculados a dicho movimiento.

Recomendaciones

Las recomendaciones de la investigación están dirigidas a sugerir acciones para completar el producto obtenido. Por lo que para mejorar la solución propuesta se recomienda para versiones posteriores:

- El sistema realiza notificaciones pero se recomienda realizar un módulo de notificaciones para que una vez autentificado, se muestren las notificaciones propias del usuario.
- Ampliar la realización de reportes a otros formatos, como hojas de cálculo y documentos Word.
- Integrar el sistema a otros ya existentes en la universidad para facilitar la obtención de información académica necesaria para el proceso de ratificación del AA.

Referencias Bibliográficas

ALVAREZ, M. A. CodeIgniter, 2009. [Disponible en: http://www.desarrolloweb.com/articulos/codeigniter.html 20.

ALLEN, R.; N. LO, et al. Zend Framework in action. Manning, 2009. p. 1933988320

BECK, K. Extreme programming explained: embrace change. Addison-Wesley Professional, 2000. p. 0201616416

BECK, K. and C. ANDRES. Extreme Programming Explained. Segunda Edición. Estados Unidos, John Wait, 2005. 186 p. 0321278658

BELLOSO CICILIA, C. I. Monografía sobre la metodología de desarrollo de software, Rational Unified Process (RUP). Facultad de Ingeniería. El Salvador, Universidad Don Bosco, 2009. p.

BOOCH, G.; J. RUMBAUGH, et al. El lenguaje unificado de modelado. Addison-Wesley, 1999. p.

BOUDREAU, T. NetBeans: the definitive guide. " O'Reilly Media, Inc." 2002. p. 0596002807

CIBERAULA. Introducción, definición y evolución de PHP. Disponible en: http://php.ciberaula.com/articulo/introduccion_php/ 17.

CHAFFER, J. and K. SWEDBERG. Jquery reference guide: a comprehensive exploration of the popular javascript library. Packt Publishing Ltd, 2010. p. 1849510059

CHAVES, D. G.; J. M. MESA, et al. Diferencias entre Scrum y XP, 2012. [Disponible en: http://www.slideshare.net/deborahgal/diferencias-entre-scrum-y-xp-12219336 36.

DANTE, G. P. Gestión de información en las organizaciones: principios, conceptos y aplicaciones. Universidad de Chile, 1998. 222 p. 9789567782000

DE LUCA, D. N. HTML5: entienda el cambio, aproveche su potencial. USERSHOP, 2011. p. 987177379X

DÍAZ, M. P.; S. MONTERO, et al. Ingeniería de la web y patrones de diseño. Pearson Prentice Hall, 2005. p. 8420546097

REFERENCIAS BIBLIOGRÁFICAS

DOMÍNGUEZ., Y. H. and R. H. DÍAZ. Sistema de Gestión del Movimiento de Alumnos Ayudantes. Facultad 2., Universidad de las Ciencias Informáticas, 2010. p.

DOUGLAS, K. and S. DOUGLAS. PostgreSQL: a comprehensive guide to building, programming, and administering PostgresSQL databases. SAMS publishing, 2003. p. 0735712573

ELMASRI, R.; S. B. NAVATHE, et al. Fundamentos de sistemas de bases de datos. Addison-Wesley, 2002. p. 8478290516

ESCOBAR, K. R.; Y. P. VÁZQUEZ, et al. Sistema de Gestión de los Procesos del la Federación Estudiantil Universitaria en la Universidad de las Ciencias Informáticas., 2012.

FARRUCHA, M. P. Sistema de Gestión del Conocimiento para una Consultoría de Inteligencia Empresarial. INTEMPRES2006, Ciudad de la Habana, 2006. p.

FERNÁNDEZ., Y. G. and Y. B. CAMPO. Sistema de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1. Facultad 1. Ciudad de la Habana, Universidad de las Ciencias Informáticas, 2010. p.

FLANAGAN, D. JavaScript: the definitive guide. " O'Reilly Media, Inc." 2002. p. 0596000480

GAJDA, W. Instant PhpStorm Starter. Packt Pub., 2013. p. 1849693951

GLOBALSIS, S. I. Sistema de Gestión Universitaria, 2010. [Disponible en: http://www.globalsis.com.ar/index.php?mod=universitaria 06.

GONZÁLEZ, Y. D. and Y. F. ROMERO Patrón Modelo-Vista-Controlador Revista Telem@ tica, 2012, 11(1): 47-57.

GOSLING, J. The Java language specification. Addison-Wesley Professional, 2000. p. 0201310082

JOSKOWICZ, J. Reglas y prácticas en eXtreme Programming *Universidad de Vigo. España*, 2008.

KABIR, M. J. Apache Server Bible. IDG Books Worldwide, Inc., 1998. p. 0764532189

KOFLER, M. What Is MySQL? , Springer, 2001. p. 1893115577

REFERENCIAS BIBLIOGRÁFICAS

LETELIER, P. and M. C. PENADES Metodologías Ágiles para el desarrollo de software: eXtreme Programming (XP), 2006.

MES. Resolución 210, La Habana, 2007.

NEWMAN, C. SQLite (Developer's Library). Sams, 2004. p. 067232685X

ORCHARD, L. M.; A. PEHLIVANIAN, et al. Professional JavaScript Frameworks: Prototype, YUI, ExtJS, Dojo and MooTools. Wrox Press Ltd., 2009. p. 047038459X

PARADIGM, V. Visual paradigm for uml Visual Paradigm for UML-UML tool for software application development, 2010.

PGADMIN. PgAdmin PosgreSQL Tools. Disponible en: http://www.pgadmin.org/ 29.

PHP-FIG. Guía de estilo de codificación. Disponible en: http://www.php-fig.org/psr/psr-2/es/ 49.

POREBSKI, B.; K. PRZYSTALSKI, et al. Building PHP Applications with Symfony, CakePHP, and Zend Framework. John Wiley and Sons, 2011. p. 1118067924

POSTGRESQL. Características, limitaciones y ventajas. Disponible en: http://postgresql-dbms.blogspot.com/p/limitaciones-puntos-de-recuperacion.html 26.

PROYECTOSAGILES. Qué es SCRUM. Disponible en: http://www.proyectosagiles.org/que-es-scrum 35.

ROJAS, I. L. R. Solución informática para la gestión de alumnos ayudantes en la gestión académica de pregrado. Facultad 1. La Habana, Universidad de las Ciencias Informáticas, 2012. p.

SIGENU. Sistema de Información Docente de la Educación Superior Cubana, 2004. [Disponible en: http://sigenu.mes.edu.cu 07.

SOMMERVILLE, I. Ingeniería de software. Séptima Edición. PEARSON EDUCACIÓN, S.A., Madrid, 2005. 712 p. 8478290745

SPARXSYSTEMS. Enterprise Architect - Herramienta de diseño UML. Disponible en: http://www.sparxsystems.com.ar/products/ea.html 13.

REFERENCIAS BIBLIOGRÁFICAS

SYMFONY. ¿Qué framework deberías elegir para tu próximo proyecto?, 2012. [Disponible en: http://symfony.es/noticias/2012/09/10/que-framework-deberias-elegir-para-tu-proximo/ 21.

TECNOLOGÍA, C. Sistema de gestión académica-SINU. Disponible en: http://www.casewaresa.com/web/index.php?option=com_content&view=article&id=16&Itemid=119 05.

WHITE, S. A. Introduction to BPMN IBM Cooperation, 2004, 2(0): 0.

WOODMAN, L. Information management in large organizations. en: Information management from strategies to action. London, ASLIB, 1985. 95-114.p.

Bibliografía

ALVAREZ, M. A. CodeIgniter, 2009. [Disponible en: http://www.desarrolloweb.com/articulos/codeigniter.html 20.

ALLEN, R.; N. LO, et al. Zend Framework in action. Manning, 2009. p. 1933988320

BECK, K. Extreme programming explained: embrace change. Addison-Wesley Professional, 2000. p. 0201616416

BECK, K. and C. ANDRES. Extreme Programming Explained. Segunda Edición. Estados Unidos, John Wait, 2005. 186 p. 0321278658

BELLOSO CICILIA, C. I. Monografía sobre la metodología de desarrollo de software, Rational Unified Process (RUP). Facultad de Ingeniería. El Salvador, Universidad Don Bosco, 2009. p.

BOOCH, G.; J. RUMBAUGH, et al. El lenguaje unificado de modelado. Addison-Wesley, 1999. p.

BOUDREAU, T. NetBeans: the definitive guide. " O'Reilly Media, Inc." 2002. p. 0596002807

CIBERAULA. Introducción, definición y evolución de PHP. Disponible en: http://php.ciberaula.com/articulo/introduccion_php/ 17.

CHAFFER, J. and K. SWEDBERG. Jquery reference guide: a comprehensive exploration of the popular javascript library. Packt Publishing Ltd, 2010. p. 1849510059

CHAVES, D. G.; J. M. MESA, et al. Diferencias entre Scrum y XP, 2012. [Disponible en: http://www.slideshare.net/deborahgal/diferencias-entre-scrum-y-xp-12219336 36.

DANTE, G. P. Gestión de información en las organizaciones: principios, conceptos y aplicaciones. Universidad de Chile, 1998. 222 p. 9789567782000

DE LUCA, D. N. HTML5: entienda el cambio, aproveche su potencial. USERSHOP, 2011. p. 987177379X

DÍAZ, M. P.; S. MONTERO, et al. Ingeniería de la web y patrones de diseño. Pearson Prentice Hall, 2005. p. 8420546097

BIBLIOGRAFÍA

DOMÍNGUEZ., Y. H. and R. H. DÍAZ. Sistema de Gestión del Movimiento de Alumnos Ayudantes. Facultad 2., Universidad de las Ciencias Informáticas, 2010. p.

DOUGLAS, K. and S. DOUGLAS. PostgreSQL: a comprehensive guide to building, programming, and administering PostgresSQL databases. SAMS publishing, 2003. p. 0735712573

ELMASRI, R.; S. B. NAVATHE, et al. Fundamentos de sistemas de bases de datos. Addison-Wesley, 2002. p. 8478290516

ESCOBAR, K. R.; Y. P. VÁZQUEZ, et al. Sistema de Gestión de los Procesos del la Federación Estudiantil Universitaria en la Universidad de las Ciencias Informáticas., 2012.

FARRUCHA, M. P. Sistema de Gestión del Conocimiento para una Consultoría de Inteligencia Empresarial. INTEMPRES2006, Ciudad de la Habana, 2006. p.

FERNÁNDEZ., Y. G. and Y. B. CAMPO. Sistema de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1. Facultad 1. Ciudad de la Habana, Universidad de las Ciencias Informáticas, 2010. p.

FLANAGAN, D. JavaScript: the definitive guide. " O'Reilly Media, Inc." 2002. p. 0596000480

GAJDA, W. Instant PhpStorm Starter. Packt Pub., 2013. p. 1849693951

GALLEGO Vázquez, José Antonio. Desarrollo web con PHP y MySQL. Anaya, 2003.

GLOBALSIS, S. I. Sistema de Gestión Universitaria, 2010. [Disponible en: http://www.globalsis.com.ar/index.php?mod=universitaria 06.

GONZÁLEZ, C. D. Base de Datos PostgreSQL, SQL avanzado y PHP. 2008.

GONZÁLEZ, Y. D. and Y. F. ROMERO Patrón Modelo-Vista-Controlador Revista Telem@ tica, 2012, 11(1): 47-57.

GOSLING, J. The Java language specification. Addison-Wesley Professional, 2000. p. 0201310082

JOSKOWICZ, J. Reglas y prácticas en eXtreme Programming *Universidad de Vigo. España*, 2008.

BIBLIOGRAFÍA

KABIR, M. J. Apache Server Bible. IDG Books Worldwide, Inc., 1998. p. 0764532189

KOFLER, M. What Is MySQL? , Springer, 2001. p. 1893115577

LARMAN, Craig. UML y Patrones. Ed, Prentice Hall, 1ra Edición, México. [en línea] < http://eva.uci.cu/mod/resource/view.php?id=8500&subdir=/UML_y_Patrones> [consultada: 22 marzo 2014]

LETELIER, P. and M. C. PENADES Metodologías Ágiles para el desarrollo de software: eXtreme Programming (XP), 2006.

MES. Resolución 210, La Habana, 2007.

NEWMAN, C. SQLite (Developer's Library). Sams, 2004. p. 067232685X

ORCHARD, L. M.; A. PEHLIVANIAN, et al. Professional JavaScript Frameworks: Prototype, YUI, ExtJS, Dojo and MooTools. Wrox Press Ltd., 2009. p. 047038459X

PARADIGM, V. Visual paradigm for uml Visual Paradigm for UML-UML tool for software application development, 2010.

PGADMIN. PgAdmin PosgreSQL Tools. Disponible en: http://www.pgadmin.org/ 29.

PHP-FIG. Guía de estilo de codificación. Disponible en: http://www.php-fig.org/psr/psr-2/es/ 49.

POREBSKI, B.; K. PRZYSTALSKI, et al. Building PHP Applications with Symfony, CakePHP, and Zend Framework. John Wiley and Sons, 2011. p. 1118067924

POSTGRESQL. Características, limitaciones y ventajas. Disponible en: http://postgresql-dbms.blogspot.com/p/limitaciones-puntos-de-recuperacion.html 26.

PRESSMAN, Roger. Técnicas de prueba del software. Ingeniería del Software Un enfoque práctico.6ta edición. McGraw-Hill, 2005. Pp.418-461

PROYECTOSAGILES. Qué es SCRUM. Disponible en: http://www.proyectosagiles.org/que-es-scrum 35.

ROJAS, I. L. R. Solución informática para la gestión de alumnos ayudantes en la gestión académica de pregrado. Facultad 1. La Habana, Universidad de las Ciencias Informáticas, 2012. p.

BIBLIOGRAFÍA

SIGENU. Sistema de Información Docente de la Educación Superior Cubana, 2004. [Disponible en: http://sigenu.mes.edu.cu 07.

SOMMERVILLE, I. Ingeniería de software. Séptima Edición. PEARSON EDUCACIÓN, S.A., Madrid, 2005. 712 p. 8478290745

SPARXSYSTEMS. Enterprise Architect - Herramienta de diseño UML. Disponible en: http://www.sparxsystems.com.ar/products/ea.html 13.

SYMFONY. ¿Qué framework deberías elegir para tu próximo proyecto?, 2012. [Disponible en: http://symfony.es/noticias/2012/09/10/que-framework-deberias-elegir-para-tu-proximo/ 21.

TECNOLOGÍA, C. Sistema de gestión académica-SINU. Disponible en: http://www.casewaresa.com/web/index.php?option=com_content&view=article&id=16&Itemid=119 05.

WHITE, Stephen A; MIERS, Derek. Guía de Referencia y Modelado BPMN. Comprendiendo y utilizando BPMN, Future Strategies Inc.

WHITE, S. A. Introduction to BPMN IBM Cooperation, 2004, 2(0): 0.

WOODMAN, L. Information management in large organizations. en: Information management from strategies to action. London, ASLIB, 1985. 95-114.p.

Glosario de términos

Aplicación: cualquier programa que corra en un sistema operativo y que haga una función específica para un usuario. Por ejemplo, procesadores de palabras, bases de datos, agendas electrónicas.

Base de Datos: es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

Interfaz: es la conexión entre dos ordenadores o máquinas de cualquier tipo dando una comunicación entre ambas.

Cliente: persona, organización o grupo de personas que encarga la construcción de un sistema, ya sea empezando desde cero, o mediante el refinamiento de versiones sucesivas.

Facultad: es una institución docente donde se imparten estudios superiores especializados en alguna materia o rama del saber, generalmente constituyen una subdivisión de una universidad.

Rol: papel, cometido o función que tiene o desempeña que interpreta un actor.

Usuario: humano que interactúa con un sistema.

Iteración: significa el acto de repetir un proceso con el objetivo de alcanzar una meta deseada, objetivo o resultado. Cada repetición del proceso también se le denomina una "iteración", y los resultados de una iteración se utilizan como punto de partida para la siguiente iteración.

Multiusuario: significa que soporta el acceso concurrente de varios usuarios.

Módulo: bloques de código que proveen extra funcionalidad o mejoras.

Navegador: es el programa que nos ofrece acceso a Internet. Debe ser capaz de comunicarse con un servidor y comprender el lenguaje de todas las herramientas que manejan la información de Web.

Página Web: es un documento HTML/XHTML accesible generalmente mediante el protocolo HTTP de Internet.

Servidor: se encarga de proporcionar al navegador los documentos y medios que este solicita. Utiliza un protocolo HTTP para atender las solicitudes de archivos por parte de un navegador.

Sistema Operativo: es el software básico de una computadora que provee una interfaz entre el resto de programas del ordenador, los dispositivos hardware y el usuario.

Windows: familia de sistemas operativos creados por *Microsoft Corporation* desde 1985 hasta la fecha.

Software: programas de sistemas, utilerías o aplicaciones expresados en un lenguaje de máquina.

Anexos

ANEXO 1 Modelo de la entrevista

Entrevista a posibles clientes o usuarios del Sistema de Gestión de Información para los Alumnos Ayudantes de la Facultad 2: Vicedecana de formación y Responsable de vida académica de la FEU a nivel UCI.

¿Cuáles son las pautas para decidir si un estudiante puede ser o no alumno ayudante?

¿Cuáles documentos se revisan para determinar si un estudiante es idóneo para ser alumno ayudante?

¿Cómo se realizan los procesos de captación de los alumnos ayudantes y el de baja?

¿Cómo se orienta y da seguimiento a las tareas asignadas a los alumnos ayudantes?

¿Cómo se realiza la evaluación de los alumnos ayudantes? ¿En qué período se hace?

¿Cuáles son los problemas presentes en la gestión de la información de los AA actualmente según su criterio?

¿Crees necesario un sistema gestor que facilite los procesos respecto a la información manejada de los alumnos ayudantes?

¿Qué características crees debe cumplir un sistema que facilite la gestión de la información de los AA en la facultad?

ANEXO 2 Historias de Usuario

Historia de usuario	
N.: 2	**Nombre:** Gestionar Jefe de Departamento
Usuario: Vicedecano de Formación	
Prioridad en el Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración Asignada: 1	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Vicedecano de Formación del sistema adicionar, modificar y eliminar un jefe de Departamento en el sistema.	
Observación: No hay	

Tabla 15 HU Gestionar Jefe de Dpto.

Historia de usuario	
N.: 3	**Nombre**: Gestionar Tutor
Usuario: Vicedecano de Formación y Jefes de Departamento	
Prioridad en el Negocio: Media	**Riesgo en Desarrollo**: Bajo
Iteración Asignada: 1	**Esfuerzo Estimado**: 0.5
Descripción: Permitirá al usuario con rol Vicedecano de Formación y Jefe de departamento del sistema asignar, modificar y eliminar profesores por asignaturas en el sistema.	
Observación: No hay	

Tabla 16 HU Gestionar tutor

Historia de usuario	
N.: 4	**Nombre:** Gestionar Asignaturas
Usuario: Vicedecano de Formación	
Prioridad en el Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración Asignada: 1	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Vicedecano de Formación del sistema adicionar, modificar y eliminar una asignaturas en el sistema.	
Observación: No hay	

Tabla 17 HU Gestionar asignatura

Historia de usuario	
N.: 5	**Nombre:** Gestionar Semestre
Usuario: Vicedecano de Formación	
Prioridad en el Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración Asignada: 1	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Vicedecano de Formación del sistema adicionar, modificar y eliminar en el sistema.	
Observación: No hay	

Tabla 18 HU Gestionar semestre

Historia de usuario	
N.: 6	**Nombre:** Gestionar Grupos
Usuario: Vicedecano de Formación	
Prioridad en el Negocio: Media	**Riesgo en Desarrollo:** Bajo
Iteración Asignada: 1	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Vicedecano de Formación adicionar, modificar y eliminar grupo en el sistema.	
Observación: No hay	

Tabla 19 HU Gestionar grupos

Historia de usuario	
N.: 8	**Nombre:** Gestionar Convocatoria de Ayudantía
Usuario: Vicedecano de Formación	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 2	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Vicedecano de Formación adicionar, modificar y mostrar en el sistema la convocatoria de alumno ayudante, además le permitirá eliminar la convocatoria realizada en el caso que lo desee.	
Observación: No hay	

Tabla 20 HU Convocatoria de AA

Historia de usuario	
N.: 10	**Nombre:** Gestionar Aprobar Solicitud de AA
Usuario: Vicedecano de Formación	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 2	**Esfuerzo Estimado:** 1
Descripción: Permitirá al rol Vicedecano de Formación aceptar o denegar la solicitud de ayudantía del estudiante y enviar una notificación al solicitante si la acción realizada fue aceptar.	
Observación: No hay	

Tabla 21 HU Aprobar solicitud de AA

Historia de usuario	
N.: 12	**Nombre:** Gestionar Actividades de los AA
Usuario: Profesor, Jefe de Departamento.	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 3	**Esfuerzo Estimado:** 1
Descripción: Permitirá al usuario con rol Profesor y Jefe de Departamento asignar, eliminar, modificar y evaluar las actividades a realizar por sus AA.	
Observación: No hay	

Tabla 22 HU Actividades de los AA

Historia de usuario	
N.: 13	**Nombre:** Reportes
Usuario: Vicedecano de Formación, Profesor, Jefe de Dpto.	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 3	**Esfuerzo Estimado:** 1
Descripción: Les permitirá a los usuarios con roles Vicedecano de Formación, Profesor y Jefe de Departamento elaborar una serie de reportes cuando les sean necesarios según los privilegios establecidos en el sistema. Reportes: estructura del MAA, solicitudes de ayudantía, solicitudes de bajas, actividades asignadas, plan de trabajo por AA, acta de baja de un AA.	
Observación: Los reportes podrán ser generados cuando sean necesarios	

Tabla 23 HU Reportes

Historia de usuario	
N.: 14	**Nombre:** Gestionar Necesidades de AA
Usuario: Jefe de Departamento.	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 2	**Esfuerzo Estimado:** 0.5
Descripción: Permitirá al usuario con rol Jefe de Departamento adicionar, modificar, eliminar y mostrar las necesidades de AA por asignatura de un Departamento.	
Observación: No hay	

Tabla 24 HU Necesidad de AA

Historia de usuario	
N.: 15	**Nombre:** Notificaciones
Usuario: Sistema	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 3	**Esfuerzo Estimado:** 0.5
Descripción: El sistema envía notificación de la solicitud de ayudantía al estudiante (aceptada), envía notificación de estado o fecha de culminación de las tareas a los AA, envía notificación al profesor y al AA cuando se le cambia el estado de la tarea, cuando se añade un AA, cuando se le asigna una tarea al AA.	
Observación:	

Tabla 25 HU Notificaciones

Historia de usuario	
N.: 16	**Nombre:** Proponer Estudiante para Ayudantía
Usuario: Profesor**,** Jefe de Departamento.	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 2	**Esfuerzo Estimado:** 0.5
Descripción: Brindará la posibilidad al usuario con rol Profesor o Jefe de Departamento de proponer a un estudiante que ellos consideren que tiene las aptitudes y actitudes para ingresar a las filas del MAA.	
Observación:	

Tabla 26 HU Proponer estudiante para ayudantía

Historia de usuario	
N.: 17	**Nombre:** Autenticar Usuario
Usuario: Estudiantes, Alumnos Ayudantes, Vicedecano de Formación, Profesor, Jefe de Departamento.	
Prioridad en el Negocio: Alta	**Riesgo en Desarrollo:** Medio
Iteración Asignada: 3	**Esfuerzo Estimado:** 1
Descripción: Permitirá a los usuarios autenticarse en el sistema para acceder a las funcionalidades del mismo de acuerdo a los permisos establecidos para sus roles.	
Observación: Para autenticarse en el sistema debe ser con el usuario uci.	

Tabla 27 HU Autenticar Usuario

ANEXO 3 Tabla de iteraciones

Iteraciones	Historias de usuarios a implementar	Prioridad en el negocio
1	Gestionar Departamento	Media
1	Gestionar Jefe de Departamento	Media
1	Gestionar Tutor	Media
1	Gestionar Asignaturas	Media
1	Gestionar Semestre	Media
1	Gestionar Grupos	Media
1	Gestionar Alumno Ayudante	Media
2	Necesidad de Alumno Ayudante	Alta
2	Convocatoria de Alumno Ayudante	Alta
2	Gestionar Solicitud de Alumno Ayudante	Alta
2	Aprobar Solicitud de Alumno Ayudante	Alta
2	Proponer Estudiante para Ayudantía	Alta
3	Gestionar Actividades de los AA	Alta
3	Evaluar Alumno Ayudante	Alta
3	Notificaciones	Alta
3	Reportes	Alta
3	Autenticar Usuario	Alta

Tabla 28 Plan de iteraciones

ANEXO 4 Tarjetas CRC

CRC Actividades	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla actividades de la Base Datos responsable de manejar los datos de las actividades asignadas a los alumnos ayudantes.	Evaluaciones EstadoAct Aa

Tabla 29 CRC Actividades

CRC Asignaturas	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla asignaturas de la Base Datos responsable de manejar los datos de las asignaturas con las que trabaja el sistema.	Departamento

Tabla 30 CRC Asignaturas

CRC Convocatoria	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla convocatoria de la Base Datos responsable de manejar los datos de las convocatorias publicadas en el sistema.	Asignaturas

Tabla 31 CRC Convocatoria

CRC Curso	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla curso de la Base Datos responsable de manejar datos de los cursos con los que se trabaja en el sistema.	

Tabla 32 CRC Curso

CRC Departamento	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla departamento de la Base Datos responsable de manejar datos de los departamentos docentes con los que trabaja el sistema	

Tabla 33 CRC Departamento

CRC EstadoAct	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla EstadoAct de la Base Datos responsable de manejar el estado de las actividades asignadas a los alumnos ayudantes.	

Tabla 34 CRC EstadoAct

CRC Evaluaciones	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla evaluaciones de la Base Datos responsable de almacenar los datos de las evaluaciones de los AA y las tareas asignadas.	

Tabla 35 CRC Evaluaciones

CRC Grupo	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla grupo de la Base Datos responsable de almacenar los datos de los grupos existentes en el sistema.	

Tabla 36 CRC Grupo

CRC JDpto	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla JDpto de la Base Datos responsable de almacenar los datos de los jefes de departamentos definidos en el sistema.	Departamento

Tabla 37 CRC JDpto

CRC ProfesorTutor	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla profesor tutor de la Base Datos responsable de almacenar los datos de los tutores asignados en el sistema.	JDpto Asignaturas

Tabla 38 CRC ProfesorTutor

CRC Categoria	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla Categoria de la Base Datos responsable de almacenar las categorías a comparar con las brindadas por el LDAP para así otorgar los roles del sistema.	Rol

Tabla 39 CRC Categoria

CRC Rol	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla Rol de la Base Datos responsable de manejar los datos de los roles con que trabaja la aplicación.	Categoria

Tabla 40 CRC Rol

CRC Semestre	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla semestre de la Base Datos responsable de controla los datos de los semestres existentes.	

Tabla 41 CRC Semestre

CRC SolicitudBaja	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla solicitud de baja de la Base Datos responsable de almacenar los datos relacionados a las solicitudes de baja.	

Tabla 42 CRC SolicitudBaja

CRC SolicitudEntrada	
Responsabilidades	**Colaboradores**
Clase entidad que representa la tabla solicitud de entrada de la Base Datos responsable de almacenar los datos relacionados con las solicitudes de ayudantía.	Asignaturas

Tabla 43 CRC SolicitudEntrada

ANEXO 5 Diagrama Entidad-Relación

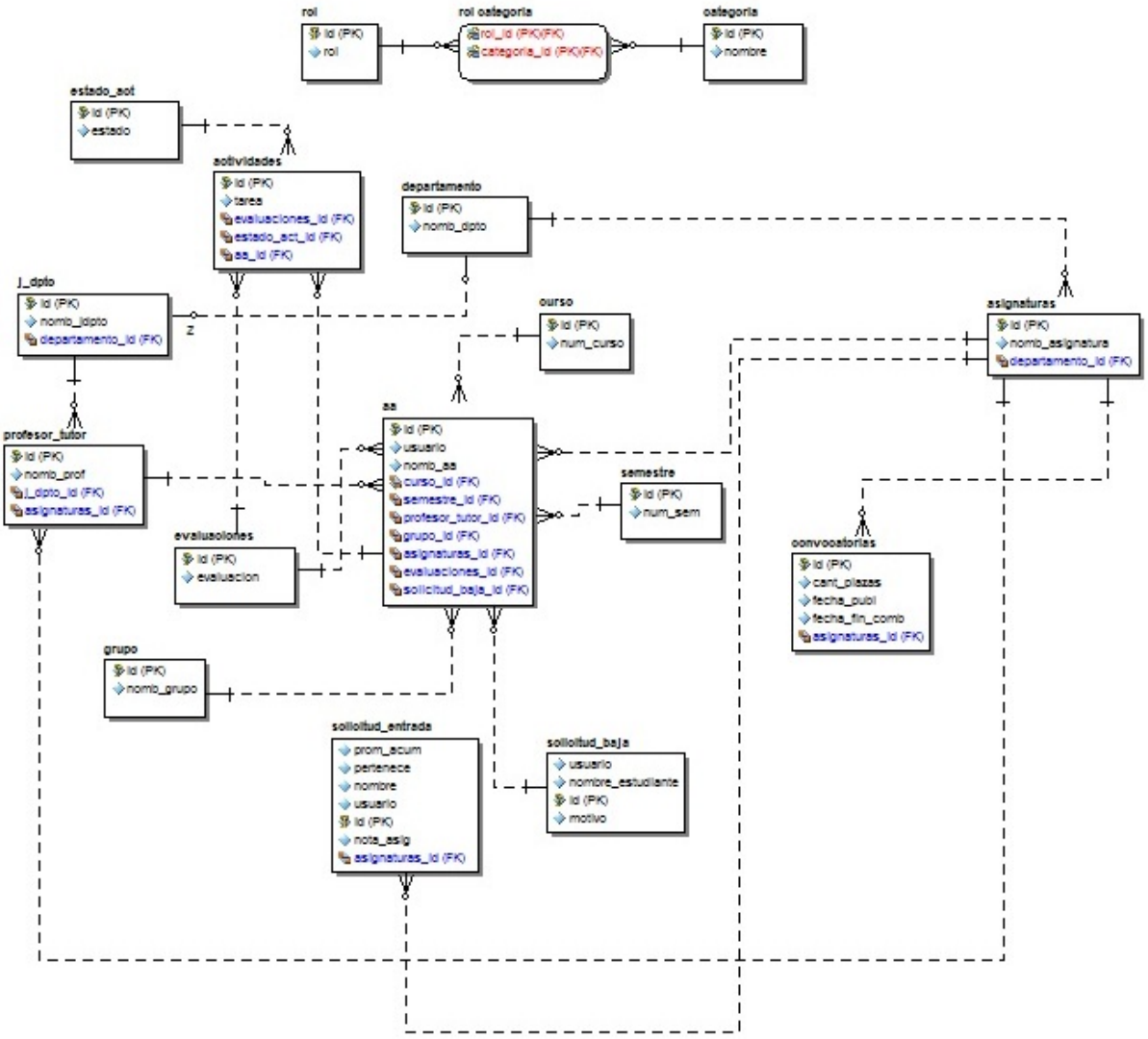


Diagrama 7.Diagrama de base de datos

ANEXO 6 Tareas de ingeniería

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU2_ Gestionar jefe de Departamento
Nombre Tarea: Modificar jefe de departamento	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 04/03/2014	**Fecha Fin:** 05/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan modificar los datos referentes a un Jefe de departamento seleccionado.	

Tabla 44 Modificar jefe de Departamento (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU3_Gestionar tutor
Nombre Tarea: Insertar profesor	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 06/03/2014	**Fecha Fin:** 07/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten la captura de los datos de un profesor (nombre, apellidos, asignatura) para la inserción del mismo en la base de datos del sistema.	

Tabla 45 Insertar profesor (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU3_ Gestionar tutor
Nombre Tarea: Eliminar profesor	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 07/03/2014	**Fecha Fin:** 08/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten eliminar un profesor seleccionado, teniendo en cuenta que para eliminarlo debe primero existir en la base de datos para luego borrarlo.	

Tabla 46 Eliminar profesor (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU3_ Gestionar tutor
Nombre Tarea: Modificar profesor	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 10/03/2014	**Fecha Fin:** 11/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten modificar los datos de un profesor seleccionado.	

Tabla 47 Modificar profesor (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU4_Gestionar Asignaturas
Nombre Tarea: Insertar asignatura	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 12/03/2014	**Fecha Fin:** 13/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten insertar una asignatura al sistema.	

Tabla 48 Insertar Asignatura (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU4_Gestionar Asignaturas
Nombre Tarea: Eliminar asignatura	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 12/03/2014	**Fecha Fin:** 13/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Se implementan las funcionalidades que permiten eliminar del sistema una asignatura seleccionada.	

Tabla 49 Eliminar Asignatura (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU4_Gestionar Asignaturas
Nombre Tarea: Modificar asignatura	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 13/03/2014	**Fecha Fin:** 14/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten modificar una asignatura seleccionada teniendo en cuenta que para modificarla debe primero existir en la base de datos.	

Tabla 50 Modificar Asignatura (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU5_Gestionar Semestre
Nombre Tarea: Insertar semestre	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 15/03/2014	**Fecha Fin:** 16/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten insertar un semestre.	

Tabla 51 TI Insertar semestre (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU5_Gestionar Semestre
Nombre Tarea: Modificar semestre	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 16/03/2014	**Fecha Fin:** 17/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten modificar datos de un semestre seleccionado.	

Tabla 52 TI Modificar semestre (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU6_Gestionar Grupos
Nombre Tarea: Crear grupo	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 18/03/2014	**Fecha Fin:** 19/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten capturar los datos de un grupo, para luego ser mostrado.	

Tabla 53 TI Crear grupo (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU6_Gestionar Grupos
Nombre Tarea: Eliminar grupo	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 19/03/2014	**Fecha Fin:** 20/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten eliminar un grupo seleccionado teniendo en cuenta que para eliminarlo debe primero crearlo para luego borrar dicho grupo.	

Tabla 54 TI Eliminar grupo (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU6_Gestionar Grupos
Nombre Tarea: Modificar grupo	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 20/03/2014	**Fecha Fin:** 21/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten modificar datos de un grupo seleccionado.	

Tabla 55 TI Modificar grupo (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU7_Gestionar Alumno Ayudante
Nombre Tarea: Insertar alumno ayudante	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 24/03/2014	**Fecha Fin:** 25/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten insertar los datos de un alumno ayudante en el perfil y luego ser mostrado.	

Tabla 56 TI Insertar alumno ayudante (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU7_Gestionar Alumno Ayudante
Nombre Tarea: Modificar alumno ayudante	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 25/03/2014	**Fecha Fin:** 26/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten modificar datos de un alumno ayudante seleccionado.	

Tabla 57 TI Modificar alumno ayudante (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU7_Gestionar Alumno Ayudante
Nombre Tarea: Eliminar alumno ayudante	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 26/03/2014	**Fecha Fin:** 27/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten eliminar un alumno ayudante seleccionado.	

Tabla 58 TI Eliminar alumno ayudante (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU8_Gestionar profesor tutor
Nombre Tarea: Insertar profesor tutor	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 28/03/2014	**Fecha Fin:** 29/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten insertar los datos de un profesor tutor seleccionado.	

Tabla 59 TI Insertar profesor tutor (1ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU14_Gestionar necesidades de AA
Nombre Tarea: Publicar necesidades de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 31/03/2014	**Fecha Fin:** 01/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten insertar las necesidades de alumno ayudante en los departamentos docentes.	

Tabla 60 TI Publicar necesidades de AA (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU8_Gestionar convocatoria de AA
Nombre Tarea: Publicar convocatoria de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 02/04/2014	**Fecha Fin:** 02/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten publicar o adicionar la convocatoria de alumno ayudante en el sistema.	

Tabla 61 TI Publicar convocatoria de AA (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU8_Gestionar convocatoria de AA
Nombre Tarea: Modificar convocatoria de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 03/04/2014	**Fecha Fin:** 03/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten modificar datos de la convocatoria publicada en el sistema en caso que lo desee. Para poder modificarla ya debe existir en el sistema.	

Tabla 62 TI Modificar convocatoria de AA (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU8_Gestionar convocatoria de AA
Nombre Tarea: Eliminar convocatoria de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 04/04/2014	**Fecha Fin:** 05/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten eliminar la convocatoria publicada en el sistema en caso deseado.	

Tabla 63 TI Eliminar convocatoria de AA (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 4	**HU:** HU10_Gestionar convocatoria de AA
Nombre Tarea: Mostrar convocatoria de AA por asignatura	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 08/04/2014	**Fecha Fin:** 08/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades para que pueda ser mostrada en el sistema la convocatoria de ayudantía.	

Tabla 645 TI Mostrar convocatoria de AA por asignatura (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU9_Gestionar solicitud de AA
Nombre Tarea: Realizar solicitud de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 09/04/2014	**Fecha Fin:** 10/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten al estudiante realizar e insertar en el sistema las solicitudes de ayudantía deseadas.	

Tabla 65 TI Realizar solicitud de AA (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU9_Gestionar solicitud de AA
Nombre Tarea: Modificar la solicitud de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 10/04/2014	**Fecha Fin:** 11/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten modificar la solicitud de ayudantía realizada.	

Tabla 66 TI Modificar solicitud de AA (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU9_Gestionar solicitud de AA
Nombre Tarea: Eliminar la solicitud de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 11/04/2014	**Fecha Fin:** 12/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten cancelar la solicitud de ayudantía realizada.	

Tabla 67 TI Eliminar solicitud de AA (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU10_Gestionar aprobación de solicitud de AA
Nombre Tarea: Aceptar solicitud de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 14/04/2014	**Fecha Fin:** 15/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten que una solicitud de ayudantía realizada sea aceptada, notificándolo. La misma ya debe existir en el sistema.	

Tabla 689 TI Aceptar la solicitud de ayudantía (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU10_Gestionar aprobación de solicitud de AA
Nombre Tarea: Denegar solicitud de AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 16/04/2014	**Fecha Fin:** 17/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permite denegar una solicitud de ayudantía realizada, notificándolo. La misma ya debe existir en el sistema.	

Tabla 69 TI Denegar solicitud de AA (2da iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU16_Proponer Estudiantes para AA
Nombre Tarea: Proponer estudiantes para AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 21/04/2014	**Fecha Fin:** 23/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten a los roles Profesor y Jefe de Departamento proponer a estudiantes para alumnos ayudantes.	

Tabla 70 Proponer Estudiantes para AA (2da iteración

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU11_ Gestionar Evaluación AA
Nombre Tarea: Evaluar alumno AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 24/04/2014	**Fecha Fin:** 25/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten a los roles Profesor, Jefe de Departamento y Vicedecano de Formación emitir una evaluación a un alumno ayudante de acuerdo a las tareas cumplidas.	

Tabla 71 TI Evaluar AA (3ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU11_ Gestionar Evaluación AA
Nombre Tarea: Modificar evaluación AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 28/04/2014	**Fecha Fin:** 29/04/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permiten los roles Profesor, Jefe de Departamento y Vicedecano de Formación cambiar la evaluación del AA.	

Tabla 72 TI Modificar Evaluación AA (3ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU15_ Gestionar Notificaciones
Nombre Tarea: Enviar notificaciones	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 01/05/2014	**Fecha Fin:** 05/05/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades para que el sistema pueda enviar notificaciones cuando se requiera; por ejemplo, cuando ha pasado el plazo para la realización de una tarea a signada a un AA y aun no se ha realizado.	

Tabla 73 TI Enviar Notificaciones (3ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU12_ Gestionar Actividades de los AA
Nombre Tarea: Asignar actividades a los AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 06/05/2014	**Fecha Fin:** 08/05/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan asignar las diferentes tareas a los alumnos ayudantes.	

Tabla 74 TI Asignar actividades a los AA (3ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU12_ Gestionar Actividades de los AA
Nombre Tarea: Eliminar actividades a los AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 09/05/2014	**Fecha Fin:** 12/05/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan eliminar alguna tarea asignada a un alumno ayudante.	

Tabla 75 TI Eliminar actividades a los AA (3ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 3	**HU:** HU12_ Gestionar Actividades de los AA
Nombre Tarea: Modificar actividades a los AA	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 12/05/2014	**Fecha Fin:** 13/05/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan modificar, cambiar alguna tarea asignada a un alumno ayudante.	

Tabla 76 TI Modificar actividades de los AA (3ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 1	**HU:** HU13_ Gestionar Reportes
Nombre Tarea: Listar Reportes	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 1
Fecha Inicio: 14/05/2014	**Fecha Fin:** 19/05/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan elaborar diferentes reportes: estructura del MAA, solicitudes de ayudantía, solicitudes de bajas, actividades asignadas, plan de trabajo por AA, acta de baja de un AA, acta de Evaluación Anual.	

Tabla 77 Listar Reportes (3ra iteración)

Tarea de Ingeniería	
Nro. Tarea:1	**HU:** HU17_Autenticar Usuario
Nombre Tarea: Autenticar	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio:19/05/2014	**Fecha Fin:** 21/05/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan la autenticación de los usuarios al sistema mediante Ldap. Luego de autenticarse el usuario solo podrá hacer uso de las funcionalidades a las que tiene acceso.	

Tabla 78 TI Autenticar (3ra iteración)

Tarea de Ingeniería	
Nro. Tarea: 2	**HU:** HU2_ Gestionar jefe de Departamento
Nombre Tarea: Eliminar jefe de departamento	
Tipo de Tarea: Desarrollada	**Puntos estimados:** 0.5
Fecha Inicio: 04/03/2014	**Fecha Fin:** 05/03/2014
Programador responsable: Manuel Alejandro Argudín Chirino y Leskenia Acosta Londres	
Descripción: Implementar las funcionalidades que permitan eliminar los datos referentes a un Jefe de departamento seleccionado.	

Tabla 79 Eliminar jefe de Departamento (1ra iteración)

ANEXO 7 Pruebas de aceptación

Caso de Prueba de Aceptación	
Código: HU2_P1	**Historia de Usuario**: HU2_Gestionar jefe de Departamento
Nombre: Modificar un jefe de Departamento	
Descripción: Probar que se pueda modificar un jefe de dpto.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Vicedecano de Formación. Llenar todos los campos de la modificación del jefe de departamento. La persona a nombrar como jefe de departamento debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Se modifica un jefe de departamento.	
Resultado Esperado: Se modifique un jefe de departamento correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 80 PA Modificar un jefe de Departamento (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU3_P1	**Historia de Usuario**: HU3_ Gestionar tutor
Nombre: Insertar un profesor	
Descripción: Probar que se pueda insertar un profesor en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por los roles Vicedecano de Formación y Jefe de Departamento. Insertar datos válidos del profesor a insertar. Llenar todos los campos de la inserción. El profesor no debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Se envian los datos del nuevo profesor.	
Resultado Esperado: Se inserte un profesor en el sistema correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 81 PA Insertar un profesor (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU3_P2	**Historia de Usuario**: HU3_ Gestionar tutor
Nombre: Modificar un profesor	
Descripción: Probar que se pueda modificar un profesor en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por los roles Vicedecano de Formación y Jefe de Departamento. Llenar todos los campos de la modificación del profesor. El profesor debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Se envien los nuevos datos del profesor.	
Resultado Esperado: Se actualicen correctamente los datos pertenecientes al profesor en el sistema.	
Evaluación de la Prueba: Satisfactoria	

Tabla 82 PA Modificar un profesor (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU3_P4	**Historia de Usuario**: HU3_ Gestionar profesor tutor
Nombre: Eliminar un profesor	
Descripción: Probar que se pueda eliminar un profesor en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por los roles Vicedecano de Formación y Jefe de Departamento. El profesor a eliminar debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Seleccionar un profesor a eliminar en el sistema.	
Resultado Esperado: Se elimine un profesor del sistema correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 83 PA Gestionar Profesor por Asignatura (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU4_P1	**Historia de Usuario:** HU4_Gestionar Asignaturas
Nombre: Insertar asignaturas	
Descripción: Probar que se pueda insertar una asignatura en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Vicedecano de Formación. Insertar datos válidos de la asignatura a insertar. Llenar todos los campos de la inserción. La asignatura a insertar no debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Se envian los datos de la nueva asignatura.	
Resultado Esperado: Se pueda insertar una asignatura en el sistema correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 84 PA Insertar Asignatura (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU4_P2	**Historia de Usuario:** HU4_Gestionar Asignaturas
Nombre: Eliminar una asignatura	
Descripción: Probar que se pueda eliminar una asignatura en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol administrador. La asignatura a eliminar debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Seleccionar la asignatura a eliminar.	
Resultado Esperado: Se pueda eliminar una asignatura en el sistema correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 85 PA Eliminar Asignatura (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU4_P3	**Historia de Usuario:** HU4_Gestionar Asignaturas
Nombre: Modificar asignaturas.	
Descripción: Probar que se pueda modificar una asignatura en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Vicedecano de Formación. Llenar todos los campos del formulario de modificación de la asignatura. La asignatura debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Seleccionar una asignatura a modificar.	
Resultado Esperado: Se pueda modificar una asignatura en el sistema correctamente.	
Evaluación de la Prueba: Satisfactoria.	

Tabla 86 PA Modificar Asignatura (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU7_P1	**Historia de Usuario:** HU7_Gestionar Alumno Ayudante
Nombre: Insertar un Alumno Ayudante.	
Descripción: Probar que se pueda insertar un AA en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por los roles Vicedecano de Formación y Jefe de Departamento. Llenar todos los campos con datos validos del estudiantes. El AA a insertar no debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Seleccionar la opción insertar un AA.	
Resultado Esperado: Se pueda se inserte un AA correctamente en el sistema.	
Evaluación de la Prueba: Satisfactoria	

Tabla 87 PA Insertar AA (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU7_P2	**Historia de Usuario:** HU7_Gestionar Alumno Ayudante
Nombre: Modificar un Alumno Ayudante	
Descripción: Probar que se pueda modificar un AA en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Vicedecano de Formación. Llenar todos los campos de la modificación del AA. El AA a modificar debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Seleccionar la opción modificar un AA.	
Resultado Esperado: Se pueda editar un AA correctamente en el sistema.	
Evaluación de la Prueba: Satisfactoria	

Tabla 88 PA Modificar AA (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU7_P3	**Historia de Usuario:** HU7_Gestionar Alumno Ayudante
Nombre: Eliminar a un Alumno Ayudante	
Descripción: Probar que se pueda eliminar un AA en el sistema.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Vicedecano de Formación. El AA a eliminar debe existir en la Base de datos.	
Entrada / Pasos de Ejecución: Seleccionar la opción eliminar un AA.	
Resultado Esperado: Se pueda eliminar un AA correctamente en el sistema.	
Evaluación de la Prueba: Satisfactoria	

Tabla 89 PA Eliminar AA (1ra iteración)

Caso de Prueba de Aceptación	
Código: HU8_P1	**Historia de Usuario:** HU8_Gestionar convocatoria de AA
Nombre: Insertar convocatoria de AA	
Descripción: Probar que se pueda adicionar la convocatoria de AA.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por los roles Vicedecano de Formación y Jefe de Departamento. Llenar los campos para publicar la convocatoria de AA. La convocatoria de AA a publicar no debe existir en el sistema.	
Entrada / Pasos de Ejecución: Seleccionar la opción insertar la convocatoria de AA.	
Resultado Esperado: Se pueda publicar la convocatoria de AA correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 90 PA Insertar convocatoria de AA (2da iteración)

Caso de Prueba de Aceptación	
Código: HU9_P1	**Historia de Usuario:** HU9_Gestionar solicitud de AA
Nombre: Realizar solicitud de ayudantía	
Descripción: Probar que se puede realizar la solicitud de ayudantía.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Estudiante. Insertar datos válidos de la solicitud. Llenar todos los campos para realizar la solicitud.	
Entrada / Pasos de Ejecución: Se realiza la solicitud.	
Resultado Esperado: Se pueda hacer la solicitud de la ayudantía correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 91 PA Realizar Solicitud de AA (2da iteración)

Caso de Prueba de Aceptación	
Código: HU11_P1	**Historia de Usuario:** HU11_ Gestionar Evaluación AA
Nombre: Evaluar alumno ayudante	
Descripción: Probar que se pueda evaluar a un estudiante.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por los roles Profesor y Jefe de Departamento. Llenar los campos para la evaluación. El estudiante a evaluar debe existir en la base de datos.	
Entrada / Pasos de Ejecución: Se selecciona la opción evaluar.	
Resultado Esperado: Se pueda evaluar al estudiante correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 92 PA Evaluar de AA (3ra iteración)

Caso de Prueba de Aceptación	
Código: HU14_P2	**Historia de Usuario:** HU14_ Gestionar Evaluación AA
Nombre: Modificar evaluación alumno ayudante	
Descripción: Probar que se pueda modificar la evaluación de un estudiante.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por los roles Profesor y Jefe de Departamento. Llenar los campos para modificar la evaluación. Debe existir en la base de datos la evaluación a modificar.	
Entrada / Pasos de Ejecución: Se selecciona la opción modificar evaluación.	
Resultado Esperado: Se pueda modificar la evaluación al estudiante correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 93 PA Modificar Evaluación de AA (3ra iteración)

Caso de Prueba de Aceptación	
Código: HU15_P1	**Historia de Usuario:** HU15_Gestionar Notificaciones
Nombre: Notificaciones	
Descripción: Probar que el sistema envíe las notificaciones.	
Condiciones de Ejecución: Ejecutada por el sistema. Que existan AA, solicitudes, tanto de ayudantía como de baja y que existan tareas asignadas los AA en el sistema.	
Entrada / Pasos de Ejecución:	
Resultado Esperado: Que el sistema envíe las notificaciones correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 94 PA Notificaciones (3ra iteración)

Caso de Prueba de Aceptación	
Código: HU12_P1	**Historia de Usuario:** HU12_Gestionar Actividades de los AA
Nombre: Asignar actividades a los AA	
Descripción: Probar que se puedan asignar actividades a los alumnos ayudantes.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Profesor y Jefe de Departamento. Deben existir los alumnos ayudantes en el sistema. Entrar los datos válidos para insertar actividad.	
Entrada / Pasos de Ejecución: Se selecciona la opción asignar tarea.	
Resultado Esperado: Se pueda asignar actividades a los AA correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 95 PA Asignar actividades de AA (3ra iteración)

Caso de Prueba de Aceptación	
Código: HU12_P2	**Historia de Usuario:** HU12_Gestionar Actividades de los AA
Nombre: Eliminar actividades a los AA	
Descripción: Probar que se puedan eliminar actividades a los alumnos ayudantes.	
Condiciones de Ejecución: La interfaz debe ser ejecutada por el rol Profesor y Jefe de Departamento. Deben existir las actividades de alumnos ayudantes en el sistema.	
Entrada / Pasos de Ejecución: Selección de la opción eliminar.	
Resultado Esperado: Se pueda eliminar correctamente.	
Evaluación de la Prueba: Satisfactoria	

Tabla 96 PA Eliminar Actividades de AA (3ra iteración)

Caso de Prueba de Aceptación	
Código: HU17_P1	**Historia de Usuario:** HU17_ Autenticar Usuario
Nombre: Autenticar	
Descripción: Probar que el sistema autentique para cada usuario.	
Condiciones de Ejecución: Se introducen los datos de autenticación del usuario del dominio UCI. Deben estar definidos los permisos de usuarios.	
Entrada / Pasos de Ejecución: Acceder a la autenticación.	
Resultado Esperado: Que el sistema muestre a cada usuario solo lo que esté permitido.	
Evaluación de la Prueba: Satisfactoria	

Tabla 97 PA Autenticar (3ra iteración)