Universidad de Matanzas sede “Camilo Cienfuegos”
Facultad de Ciencias Técnicas
Departamento de Informática

SISTEMA WEB PARA LA GESTIÓN DE PROYECTOS E INVESTIGACIONES COMO RESPUESTA A LA DEMANDA DEL TERRITORIO

Trabajo de Diploma en opción al Título de Ingeniero Informático

Autor: Anniel Marrero Santana

Tutora: MSc. Liz Pérez Martínez

Junio, 2018

"A veces cuando innovas, cometes errores. Es mejor admitirlos rápidamente, y seguir adelante apostando por tus otras innovaciones."

Steve Job

"Estoy convencido que la mitad de lo que separa a los emprendedores con éxito y los que no tienen éxito es la pura perseverancia."

Steve Job

A mis padres, mi hermano, mi novia, mis primos, en general a toda mi familia porque todo este esfuerzo es por y para ellos

Agradecimientos

Este trabajo de diploma ha sido posible gracias al apoyo de muchas personas, a las cuales quisiera expresarle mi más sincera gratitud.

A mis padres Grisel y Alexce, porque a ellos les debo la vida, porque siempre me han brindado su apoyo, por su amor desinteresado, su ayuda incondicional, porque con su cariño contribuyen a mi superación cultural.

A mi hermano Abiel, mi familia que siempre me han apoyado.

A Liz Pérez por aceptar tutorar mi tesis, por su experiencia, sus conocimientos, su paciencia, por su ayuda en todo momento.

A mis compañeros de aula, especialmente: Isseli, Lisdany, Ernesto Torres, Collazo.

A todos muchas gracias.

Declaración de Autoría

Yo, Anniel Marrero Santana, declaro que soy el único autor de este trabajo de diploma en opción al grado de Ingeniero Informático, y autorizo a la Universidad de Matanzas sede “Camilo Cienfuegos” a hacer uso del mismo, para la finalidad que estimen conveniente.

Y para que así conste, firmo la presente a los 14 días del mes de junio del año 2018.

_____ _____

Firma del Autor Firma del Tutor

Opinión del Tutor

DATOS PERSONALES DEL TUTOR

Nombre y apellidos: Liz Pérez Martínez.
Centro de trabajo: Universidad de Matanzas.
Organismo a que pertenece: Ministerio de Educación Superior – MES.
Cargo que ocupa: Vicedecana Facultad de Ciencias Técnicas.
Especialidad de la que es graduado: Ingeniería Informática. Universidad de Matanzas, 2012.
Categoría docente o investigativa: Asistente.
Grado científico: Master en Ciencias, Universidad de Matanzas, 2015.

DATOS DE LA TESIS Y EL DIPLOMANTE

Nombre y apellidos: Anniel Marrero Santana.
Centro de estudio: Universidad de Matanzas sede “Camilo Cienfuegos”.
Título de la Tesis: Sistema web para la gestión de proyectos e investigaciones como respuesta a la demanda del territorio.

OPINION SOBRE EL TRABAJO

La tesis presentada posee gran actualidad, pues intenta resolver un problema real presente en la Universidad de Matanzas, y además contribuye a la informatización de nuestra sociedad.
El tutor de este trabajo de diploma considera que, durante su ejecución, la estudiante mostró las cualidades que a continuación se detallan:

- Autonomía, abordó temáticas que no recibió en su formación como ingeniero y encontró soluciones eficientes, demostrando así su capacidad de autosuperación.
- Madurez investigativa, el estudiante supo adecuar en su investigación tecnologías actuales de gran impacto a nivel mundial. Fue consecuente con los aspectos tanto metodológicos como de la investigación científica propiamente.
- Rigor y tenacidad, manifestado desde el tratamiento de los conceptos estudiados, su diseño y hasta llegar al desarrollo de la aplicación.

Una gran cantidad de clases y métodos, un diseño bien concebido, pruebas correctamente realizadas y otros recursos desarrollados para culminar su investigación, unido a una excelente planificación del tiempo dieron una gran calidad al trabajo obtenido.

Por todo lo anteriormente señalado, considero que el estudiante Anniel Marrero Santana reúne los requisitos para el título de Ingeniero Informático.

Espero, que su futuro como profesional colme todas sus expectativas y le depare metas que amplíen su capacidad como Ingeniero Informático.

_____

MSc. Liz Pérez Martínez
Dpto. Informática
Universidad de Matanzas
Junio/2018

Resumen

El sector de la educación se suma a los nuevos tiempos, marcados por el desarrollo vertiginoso de las nuevas tecnologías, el ensayo y diseño asistido por computadora. La Universidad de Matanzas (UM) no queda exenta a esta realidad, ésta organización cuenta con una aplicación web disponible en http://www.umcc.cu, donde expone información referente a la misma y ofrece varios servicios. A pesar de ello la institución no cuenta con un sitio que promocione su cartera de productos y/o servicios para la solución de problemas de investigación ni posee una herramienta que facilite la gestión de la información referente a esta temática, que ahorre recursos tangibles que se almacenan en archivadoras cuyos volúmenes de papeles crecen. Situación a la que se suma el hecho de no disponer de un medio propio para la divulgación de noticias, eventos, publicaciones y otras informaciones. Para solucionar este problema se realizó un estudio sobre la gestión de las propuestas de las empresas y los problemas de investigación asociados a las líneas de investigación de la Universidad, para ser solucionados a partir de un proceso de asignación de tareas en las cuales involucran tanto profesores como estudiantes de la UM. Existen tareas que tributan a trabajo diploma, lo cual permite crear un banco de problemas, que permitirá que los estudiantes lo consulten y puedan solicitar su participación. Este estudio quedó concretado con la construcción de un sistema web que implementa las necesidades descritas.

Abstract

The education´s sector is present in the new times, marked by the development of the technologies, the essay and design assisted by computers. The University of Matanzas (UM) has a web application available in http://www.umcc.cu, here we can found information and many services. Although, the institution does not have a web site that publish its products and services for given solutions to the investigation´s problems nor has a tool for management this information, that save important resource that are saved in amount of papers. Was realize a big study about the management of the challenge of the enterprise and the investigation´s problems, associated to the investigation´s line of the University for given solution to the problematic question, that solution apprehend many task in which students and professor will participate. There is task that throw in diploma´s work, that will create a bank of problems that allow to students request a participation. This study was concreted in the construction of the system web that meet the describe needs.

Contenido

Índice de Tablas

Índice de Ilustraciones

Introducción

El acelerado desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC) ha transformado de forma objetiva nuestra sociedad. Cada día, la informática adquiere más relevancia en la vida de las personas y en las instituciones administrativas y sociales. El desarrollo de las TIC ha revolucionado el mundo, dando paso a grandes adelantos en los distintos sectores de la actividad humana.

Cuba apuesta hoy al desarrollo de la informática como variante de avance tecnológico. Los especialistas están llamados a jugar un papel protagónico en el futuro económico y social del país. Actualmente, se sigue perfeccionando el trabajo y ampliando el radio de acción de las nuevas tecnologías en beneficio de todas las personas. Se planifican metas ambiciosas que están a la altura de los países del primer mundo.

Desde el surgimiento de las redes de comunicación interconectadas (internet), las organizaciones se han apoyado en el uso de aplicaciones web como un instrumento de difusión e intercambio de recursos, información y servicios, convirtiéndolas en la carta de presentación de las mismas y en muchos casos supone la primera toma de contacto entre el cliente y la organización.

El sector de la educación también se suma a los nuevos tiempos, marcados por el desarrollo vertiginoso de las nuevas tecnologías, el ensayo y diseño asistido por computadora se utiliza cada vez más en los proyectos de gran envergadura, pues permite un gran ahorro de tiempo y recursos, premisa que persiguen las instituciones dedicadas a este fin.

La Universidad de Matanzas (UM) no queda exenta a esta realidad, ésta organización cuenta con una aplicación web publicada en Internet disponible en http://www.umcc.cu, donde expone información referente a la misma y ofrece varios servicios como la promoción de productos académicos, de tal modo que los usuarios puedan solicitarlos online, también brinda la posibilidad de acceso al correo electrónico para los profesores, un foro, un libro de visitas, permite un intercambio fluido con los usuarios, entre otros; además cuenta con una intranet que está expandida por las diferentes áreas de la universidad y a la vez algunas de estas áreas poseen su propio sitio web en la red interna, así mismo los diferentes departamentos y facultades subordinados a esta organización.

Pese a la existencia de estos servicios, la institución no cuenta con un sitio que promocione su cartera de productos y/o servicios para la solución de problemas de investigación. En otras palabras, no poseen una herramienta que la identifique como una institución destinada más allá de fines educativos, que facilite la gestión de la

información referente a esta temática, que ahorre recursos tangibles que se almacenan en archivadoras cuyos volúmenes de papeles crecen a medida que pasan los cursos, situación que resulta muy incómoda y retardada a la hora de buscar y gestionar una información específica, conocer estadísticas o simplemente cuando se quiere divulgar y promocionar información referente a las ofertas de investigación y disponibilidad de afrontar nuevas investigaciones. Situación a la que se suma el hecho de no disponer de un medio propio para la divulgación de noticias, eventos, publicaciones y otras informaciones de interés tanto para estudiantes, docentes e investigadores, como para organizaciones fuera del esquema universitario.

Todo lo anterior se traduce en un modelo de gestión de la información desordenado, descentralizado, con un elevado consumo de recursos tangibles e intangibles, que dificulta la toma de decisiones y tributa a la pérdida monetaria por falta de comunicación.

La **situación problémica** descrita anteriormente evidencia la necesidad de disponer de una aplicación web que permita divulgar la cartera de productos y servicios de la UM, así como sus disponibilidades de investigaciones, y que posibilite un vínculo con la gestión empresarial y las propias necesidades de esta.

Lo que deriva en el siguiente **problema científico,** ¿cómo gestionar las propuestas de las empresas en la UM y la asignación de temas de tesis a los estudiantes?

Para regir nuestra investigación nos hemos trazado la siguiente **hipótesis**, **s**i se desarrolla un sistema web se podrán gestionar las propuestas de las empresas y la asignación de temas de tesis a los estudiantes de manera eficaz.

El **objeto de estudio** es la gestión de las propuestas de las empresas en la UM y la asignación de temas de tesis a los estudiantes.

El **campo de acción** es la informatización del proceso de gestión de las propuestas de las empresas en la UM y la asignación de temas de tesis a los estudiantes.

Como **objetivo general** de la investigación se tiene, desarrollar un sistema web para la gestión de proyectos e investigaciones como respuesta a las demandas del territorio a partir de las propuestas de las empresas y la asignación de temas de tesis a los estudiantes.

Para dar cumplimiento al objetivo general se trazan los siguientes **objetivos específicos:**

- ✓ Analizar los fundamentos teóricos que caracterizan el estado del arte de la problemática planteada, así como las tecnologías para el diseño e implementación de la propuesta de solución.
- ✓ Diseñar la propuesta de solución empleando la metodología de desarrollo de software seleccionada.

- ✓ Implementar la propuesta de solución empleando las tecnologías seleccionadas.
- ✓ Validar la propuesta de solución mediante pruebas.

Se empleó la metodología de desarrollo de software Programación eXtrema o *eXtreme Programming* (XP) formulada por Kent Beck, autor de *Extreme Programming Explained: Embrace Change* (1999). Estametodologías enfatiza la adaptabilidad sobre la previsibilidad, considera que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Plantea que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

Para el desarrollo de la investigación se utilizaron diversos **métodos y técnicas** tales como:

- ✓ Dentro de los métodos teóricos:
  - Método de **análisis histórico – lógico**: permitió estudiar la trayectoria y desarrollo de los sistemas de gestión existentes.
  - Método de **análisis y síntesis**: este se precisó durante la revisión bibliográfica y el análisis de los resultados, permitiendo descomponer lo complejo en sus partes y cualidades, la división del todo en sus múltiples relaciones para luego unir las partes analizadas, descubrir las relaciones y características generales entre ellas.
  - Método **inductivo - deductivo**: su uso fue necesario tanto en la revisión bibliográfica, como en el análisis de los resultados, permitiendo arribar a conclusiones que se infirieron a partir de propiedades y relaciones existentes entre los elementos que conforman el fenómeno objeto de estudio.
- ✓ Como métodos empíricos, utilizados por medio de las siguientes técnicas:
  - **Observación**: permitió entender el proceso de solución a problemas de investigación en la UM.
  - **Entrevistas**: aportaron datos esenciales a la investigación puesto que se entrevistaron a los implicados en el proceso objeto de estudio. Fue útil en distintos momentos de la investigación; fundamentalmente al inicio, cuando se realizó el levantamiento de requisitos para efectuar una exploración preliminar

del problema a investigar y justo antes de concebir la interfaz del software, para de esta forma desarrollarla a gusto del cliente.

Entre los **aportes** de la investigación se destacan:

- el **teórico-investigativo**, al integrar los procedimientos tradicionales más utilizados por autores relacionados con el tema, a través de diferentes fases, etapas y pasos que permiten orientar metodológicamente la secuencia de acciones lógicas a desarrollar; y los elementos a tener en cuenta para la continuidad de la investigación,
- el **práctico**, al desarrollar una herramienta informática que asista la gestión de la información referente a la gestión de proyectos e investigaciones de la UM como respuesta a las demandas del territorio.

Los aportes anteriores de una u otra manera tienen una implicación **social**, pues con la implantación del software para la gestión de la información se hace más sencilla la actividad de todos los implicados; reduciendo además, los errores humanos tan comunes en la manipulación de la información.

El **resultado esperado** de este trabajo es contar con una herramienta desarrollada en ambiente web fácil de manipular y administrar, la que permitirá ahorrar tiempo, más confiabilidad y seguridad en la interactividad entre los usuarios. La gestión de los datos posibilitará una mejor organización de la información, logrando de forma segura la integridad, extracción, manipulación y persistencia de los datos.

Atendiendo a lo planteado anteriormente, la tesis queda estructurada en introducción, tres capítulos, conclusiones, recomendaciones y referencias bibliográficas, según sigue:

- ✓ Una Introducción, donde se caracteriza la situación problemática y se fundamenta el problema científico a resolver.
- ✓ Capítulo 1: “Marco Teórico-Referencial.” Donde se exponen los principales conceptos referentes a la investigación. Así como la selección de la metodología y tecnologías para el desarrollo de la aplicación propuesta como solución al problema planteado.
- ✓ Capítulo 2: “Descripción de la solución propuesta.” Se aplican los principales artefactos de la metodología de desarrollo de software seleccionada para el diseño de la propuesta de solución.
- ✓ Capítulo 3: “Construcción y validación.” Donde se analiza el resultado obtenido y se valida la propuesta de solución.
- ✓ Un apartado de conclusiones donde se verifica el cumplimiento de los objetivos trazados al inicio de la investigación.

- ✓ Las recomendaciones en la cual se plasman una serie de propuestas encaminadas a la continuidad de esta investigación.
- ✓ Y las referencias de la bibliografía citada.

Capítulo I: Marco teórico-referencial

1.1 Introducción

En el presente capítulo se realiza un análisis del estado del arete de la temática objeto de estudio, sobre la gestión de las propuestas de las empresas y los problemas de investigación asociados a las líneas de investigación de la UM, para ser solucionados a partir de un proceso de asignación de tareas en las cuales involucran tanto profesores como estudiantes. Además, se hace mención a las diferentes tendencias tecnológicas que intervienen en el desarrollo de la propuesta de solución, así como a la metodología de desarrollo de software empleada.

1.2 Flujo del proceso involucrado en el campo de acción

Mediante el software las empresas o los profesores de la UM pueden plantear sus problemáticas para ser resueltas por la Universidad, mediante una cartera de servicios que ofrece, estas solicitudes pasan a ser analizadas para su posterior aprobación, lo que creará una solución general que será dividida en tareas las cuales serán solicitadas por estudiantes o profesores. Constituye otro objetivo que los estudiantes desde años tempranos puedan consultar las tareas que tributan a temas de tesis para interesarse por uno u otro, no esperando al final de la carrera para seleccionarlos. En el sistema interactúan tanto las empresas, estudiantes y profesores de la Universidad.

1.3 Fundamento y antecedentes del trabajo

1. Se encontró el sistema EVA entorno virtual de aprendizaje en la UCI, que gestiona la información referida a la evaluación del aprendizaje autónomo. Este sistema es un espacio educativo alojado en la web, un conjunto de herramientas informáticas que posibilitan la interacción didáctica de manera que el alumno pueda llevar a cabo las labores propias de la docencia como son conversar, leer documentos, realizar ejercicios, formular preguntas al docente, trabajar en equipo… etc. Todo ello de forma simulada sin que medie una interacción física entre docentes y

alumnos. Cuando hablamos de Entorno Virtual de Aprendizaje (EVA) o en ingles Virtual learning environment (VLE), también conocido por las siglas LMS (Learning Management System), a todos se nos viene a la cabeza Moodle, el más conocido y extendido EVA del mercado. El más extendido entre otras cosas porque se trata de un programa de código abierto, es decir de licencias gratuita. Nacido a la vera de las universidades, cuna del software libre.

2. Trello es una herramienta gratuita que te permite organizar tus tareas y proyectos a través de tableros. Tareas, ideas y recursos se organizan en columnas obteniendo una gran visibilidad de la dimensión del trabajo. Trello se basa en el sistema Kaban para la gestión de tareas, en donde en una pizarra blanca se introducen las diferentes tareas con post-its. Cada tarea está separada en una columna diferente según su estado en el proceso de finalización. Un tablero representa un proyecto a completar que contiene una colección de tareas (tarjetas) que están organizadas en columnas verticales. Cada tablero puede ser privado, público o sólo compartirse con otros miembros del equipo. Cuando invitas a otro compañero a un tablero, puedes darle permisos de edición o simplemente permisos de lectura.

3. ApowerMirror se puede enumerar en una de las mejores aplicaciones para profesores, ya que les permite mostrar todo su contenido móvil en una pantalla más grande. Con esto, su aula será mucho más interesante al agregar sus recursos y mostrarlos a su clase, ayudando a sus estudiantes a aprender más y vincular la teoría con una mejor práctica. Este programa se puede aplicar para Mac y Windows PC. En cuanto a dispositivos móviles, los dispositivos Android pueden obtener la aplicación en Google Play, mientras que los usuarios de iPhone pueden utilizar con la aplicación de escritorio a través de AirPlay.

4. EDMODO. Es una plataforma educativa con funciones similares a una red social, su estética es similar a Facebook sin embargo su enfoque es totalmente educativo. Fue fundada en Chicago Illinois con el fin de llevar

la educación al ambiente del siglo XXI. El registro es completamente gratuito y ofrece tres tipos de cuentas “Profesor”, “Estudiante” y “Padre”. Al crear una cuenta como profesor puedes dar de alta grupos organizados por grados, al crear un nuevo grupo la plataforma asigna un código que se debe proporcionar a los estudiantes para unirse al mismo. Una vez creados los grupos y vinculando las cuentas de los Estudiantes a cada grupo EDMODO permite crear diferentes tipos de asignaciones que el alumno debe resolver, desde tareas escritas, encuetas, pruebas de opción múltiple, etc. Cuenta también con una especie de blog, muy similar a estar en un grupo de Facebook pero completamente controlado por el profesor por los que nadie puede unirse sin que el profesor lo autorice, volviéndose completamente seguro para los estudiantes. Este blog o foro es ideal para cuestionar al profesor sobre cualquier dudad generada en clase.

Ninguno de los sistemas existentes resuelve el problema detectado en esta investigación dado que fueron diseñados para negocios específicos, con características propias que, aunque muestran algunas similitudes con nuestra propuesta, se alejan en gran medida del resultado esperado. Además, tampoco son de distribución libre, por lo que la Universidad de Matanzas no puede disponer de ellos.

1.4 Metodología de desarrollo de software: Programación eXtrema

Programación extrema (*Extreme Programming*, XP por sus siglas en inglés) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo (Joskowicz, 2008).

Según (Goto, y otros, 2014), XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios.

Su creador, (Beck, 2000) la define como especialmente adecuada para proyectos con requisitos imprecisos donde existe un alto riesgo técnico.

A continuación, se muestran los principales artefactos de la metodología XP:

1.4.1 Las Historias de Usuario

Las historias de usuario son descripciones cortas y simples de una funcionalidad, escritas desde la perspectiva de la persona que necesita un producto de software, por lo general el usuario, área de negocio o cliente.

Es la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas (Jeffries, y otros, 2001).

1.4.2 Roles

Los roles, según varios autores, deben ser:

- ✓ Programador: escribe las pruebas unitarias y produce el código del sistema. Responsable de decisiones técnicas, de construir el sistema, sin distinción entre analistas, diseñadores o codificadores (Hurtado, y otros, 2005).
- ✓ Cliente: escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio (Calero, 2003).
- ✓ Encargado de pruebas *(Tester):* ayuda al cliente a escribir las pruebas funcionales (Reynoso, 2012). Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas (Beck, 2000).
- ✓ Encargado de seguimiento (*Tracker*): proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración (Cubel, 2012).

- ✓ Entrenador (*Coach*): es responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.
- ✓ Consultor: es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.
- ✓ Gestor (*Big boss*): es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación.

1.4.3 Proceso

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se pierde calidad en el software o no se cumplen los plazos establecidos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración.

(Beck, 2000) caracterizó al ciclo de vida ideal de XP consiste de seis fases: Exploración, Planificación de la Entrega (Release), Iteraciones, Producción, Mantenimiento y Muerte del Proyecto.

1.4.4 Prácticas

La principal suposición que se realiza en XP, es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de las siguientes prácticas (Wake, 2000).

- ✓ El juego de la planificación. Hay una comunicación frecuente el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo

requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración.

- ✓ Entregas pequeñas. Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más tres meses.
- ✓ Metáfora. El sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema (conjunto de nombres que actúen como vocabulario para hablar sobre el dominio del problema, ayudando a la nomenclatura de clases y métodos del sistema).
- ✓ Diseño simple. Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.
- ✓ Pruebas. La producción de código está dirigida por las pruebas unitarias. Estas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.
- ✓ Refactorización (*Refactoring*). Es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. Se mejora la estructura interna del código sin alterar su comportamiento externo (Poppendieck, 2003).
- ✓ Programación en parejas. Toda la producción de código debe realizarse con trabajo en parejas de programadores. Esto conlleva ventajas implícitas (menor tasa de errores, mejor diseño, mayor satisfacción de los programadores).
- ✓ Propiedad colectiva del código. Cualquier programador puede cambiar parte del código en cuando lo desee.
- ✓ Integración continua. Cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día.
- ✓ 40 horas por semana. Se debe trabajar un máximo de 40 horas por semana. No se trabajan horas extras en dos semanas seguidas. Si esto ocurre, probablemente está ocurriendo un problema que debe corregirse. El trabajo extra desmotiva al equipo (Gittins, y otros, 2001).

- ✓ Cliente in-situ. El cliente tiene que estar presente y disponible todo el tiempo para el equipo. Éste es uno de los principales factores de éxito del proyecto XP. El cliente conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y los programadores pueden resolver de manera inmediata cualquier duda asociada. La comunicación oral es más efectiva que la escrita.
- ✓ Estándares de programación. XP enfatiza que la comunicación de los programadores es a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación para mantener el código legible.

1.5 Tendencias tecnológicas

Es necesario para el desarrollo de un producto informático que satisfaga una necesidad existente, el empleo de diferentes herramientas y tecnologías con demostrada idoneidad. Ya que, según el problema a resolver, es de suma importancia una correcta elección de las mismas.

1.5.1 Patrón de diseño: Modelo Vista Controlador

El modelo-vista-controlador (MVC) es un patrón de arquitectura de software que separa los datos y la lógica del 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, es decir, por un lado, define componentes para la representación de la información, y por otro lado para la interacción del usuario. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento (Rivera, 2015).

El Modelo Vista Controlador es aplicable al desarrollo de cualquier aplicación independientemente del lenguaje de programación elegido. MVC consiste en dividir el código de una aplicación en capas (Fernández, 2012):

- ✓ **Modelo**: es todo acceso a datos y las funciones que llevan lo que llaman "lógica de negocio", o sea datos y reglas de negocio. Lleva un registro de las vistas y controladores del sistema. Cada acceso a datos se pone en su función individual porque, de esta forma, si se cambia de gestor de bases de datos este cambio sólo afecta a estas funciones, no al resto de la aplicación. Tener

el modelo bien delimitado permite la existencia de varias aplicaciones que compartan el mismo.

- ✓ **Vista:** en una aplicación Web, es el HTML y lo necesario para convertir datos en HTML. O sea, muestra la información del modelo al usuario. Tienen un registro de su controlador asociado (normalmente porque además lo instancia). Pueden dar el servicio de "Actualización", para que sea invocado por el controlador o por el modelo. Tener la vista separada del controlador permite cambiar la aplicación para que genere, en lugar de HTML, algo distinto (por ejemplo, WML), sin tener que tocar más que una parte completamente delimitada del código.
- ✓ **Controlador:** es lo que une la vista y el modelo. Por ejemplo, son las funciones que toman los valores de un formulario, consultan la base de datos (a través del modelo) y producen valores, que la vista tomará y convertirá en HTML. En resumen, gestiona las entradas del usuario. Contiene reglas de gestión de eventos, del tipo “SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas. De este modo, el código que “hace algo” está perfectamente separado del código dedicado a crear HTML.

1.5.2 Frameworks

JQuery

Es una poderosa biblioteca de Java Script que ayuda a desarrolladores y diseñadores a agregar elementos interactivos y dinámicos a sus sitios, limando inconsistencias en los navegadores y reduciendo grandemente el tiempo de desarrollo (Chaffer, y otros, 2010).

ASP.NET

Constituye un entorno para aplicaciones web desarrollado y comercializado por Microsoft. Es usado por programadores y diseñadores para construir sitios web dinámicos, aplicaciones web y servicios web XML (Microsoft, 2008). Apareció en enero de 2002 con la versión 1.0 del .NET Framework, y es la tecnología sucesora de la tecnología Active Server Pages (ASP). ASP.NET está construido sobre el Common Language Runtime, permitiendo a los programadores escribir código ASP.NET usando cualquier lenguaje admitido por el .NET Framework (Otegem, 2015).

Uno de los objetivos más importantes considerados para creación de la tecnología .NET es el de otorgar un gran rendimiento. Es por ello que se plantean las principales características que combina .NET para convertirse en una de las mejores opciones para el desarrollo de aplicaciones según (Brianza, 2006).

- ✓ **Eficiencia:** para que la tecnología tenga éxito al ser utilizada por las Empresas, el personal especializado debe estar capacitado para migrar las aplicaciones y así evitar un rendimiento deficiente al ejecutar el código, ya que el *Common Language Runtime* (CLR) ejecuta el código de manera especial. El CLR compila en un lugar particular todos los códigos de aplicaciones en códigos naturales de máquinas, esto con la finalidad de asegurar un rendimiento óptimo. La conversión mencionada puede realizarse en el momento en que se ejecuta la aplicación o bien cuando se instala la aplicación por primera vez. La compilación realizada en el proceso hace uso de todas las características del microprocesador disponible en diferentes plataformas, logrando así superar a las aplicaciones tradicionales de Windows.
- ✓ **Soporte de lenguajes**: esta característica es quizás considerada como una de las más importantes para beneficio de los desarrolladores, debido que ASP.NET ofrece la posibilidad de escribir códigos en diversos lenguajes.ASP.NET soporta la programación en lenguajes como, Visual Básic.Net y C#.
- ✓ **Código y contenido por separado**: en la versión tradicional ASP, se presenta el inconveniente de tener que crear la interfaz de usuario y el código ASP de manera conjunta, es decir se realiza una combinación de código de imágenes, botones y tablas de HTML y secciones en VBScript o JScript poco prácticas. La versión actual ASP .NET soluciona el problema separando la interfaz de usuario con el código.
- ✓ **Compatibilidad con navegadores**: permite la creación de páginas Web que funcionan correctamente en todos los navegadores. Esta mejora se incluye gracias a los controles de servidor que posee la nueva versión. En el momento en el que un control es procesado, automáticamente se realiza un chequeo del tipo de navegador que se está ejecutando y se genera una página adecuada para el navegador.
- ✓ **Código Compilado**: la compilación no interpreta el código como lo hace la versión ASP. Sino que dentro del entorno *New Generation Windows Services*

(NGWS) el código es compilado *just- in- time*, aumentando el rendimiento mediante el soporte nativo y servicios de caché.

- ✓ **Controles de Servidor**: dentro de los aspectos importantes de ASP.NET, se encuentra su librería de clases, la cual brinda el programador una herramienta para creación de aplicaciones multiplataforma, permitiendo también un considerable ahora en las líneas de código empleado.

Este propio autor (Brianza, 2006) plantea que ASP. NET ofrece varias ventajas importantes referentes a los modelos de programación Web empleados antes del surgimiento de esta tecnología:

- ✓ **Mejor rendimiento**. A diferencia de sus predecesores es capaz de aprovechar las ventajas del enlace anticipado, la compilación *just-in-time*, la optimización nativa y los servicios de caché desde el primer momento. Por lo tanto, existe un incremento importante del rendimiento desde el inicio, donde se comienza a escribir el código.
- ✓ **Compatibilidad con herramientas de primer nivel**. El ambiente de trabajo se complementa con un diseño y una caja de herramientas muy completas. Los controles de servidor de arrastra y colocar y la implementación automática ejemplifican la eficacia de las herramientas empleadas.
- ✓ **Eficacia y flexibilidad**. Se encuentra disponible para los programadores de aplicaciones web, debido a que la biblioteca de clases, la mensajería y las soluciones de acceso a datos se encuentran accesibles desde la Web de manera uniforme. Es posible elegir el lenguaje que mejor se adapte a la aplicación para desarrollar o incluso implementar varios lenguajes.
- ✓ **Simplicidad**. Facilita la realización de tareas como el envío de formularios, autenticación del cliente y la implementación y configuración de sitios. El ambiente de trabajo permite generar interfaces de usuario, que separan la lógica de aplicación del código de presentación, además de controlar eventos de forma sencilla a través del modelo de procesamiento de formulario de tipo Visual Basic. CLR simplifica la programación con servicios de código administrado como el recuento de referencia automático y el recolector de elementos no utilizados.
- ✓ **Facilidad de uso**. Utiliza una configuración jerárquica basada en texto, lo cual simplifica la aplicación de la configuración al entorno del servidor y a las aplicaciones Web. La información de configuración es almacenada como texto

sin formato, por lo que se puede aplicar la nueva configuración sin la ayuda de herramientas de administración local. Una aplicación ASP. NET se implementa en un servidor de forma sencilla mediante una copia de los archivos necesarios, por lo que no se requiere reinicio de servidor ni para implementar o reemplazar el código compilado en ejecución.

- ✓ **Escalabilidad y disponibilidad.** Fue diseñado tomando en cuenta la escalabilidad con características específicamente a medida, con el fin de mejorar el rendimiento en entornos agrupados y de múltiples procesadores. También controla y administra los procesos de tal manera que, si uno no se comporta adecuadamente, se pueda crear un nuevo proceso ayudando a mantener la aplicación disponible.
- ✓ **Posibilidad de personalización y extensibilidad**. Cuenta con una arquitectura que permite a los programadores insertar código en el nivel adecuado. Y es posible extender o sustituir un subcomponente del motor de tiempo de ejecución con un componente escrito personalizado.
- ✓ **Seguridad**. Con la configuración por aplicación y la autenticación de Windows, es posible obtener una seguridad completa de las aplicaciones

Entity Framework y Object Relational Mapping

*Entity Framework (*EF) es un *Object Relational Mapping* (ORM) que permite a los desarrolladores de .NET trabajar con datos relacionales usando objetos específicos del dominio (Pereira, 2014). A su vez, un *Object Relational Mapping* (ORM) es una técnica de programación para convertir las tablas de una base de datos, en una serie de entidades que simplifiquen las tareas básicas de acceso a los datos para el programador (Fundora, 2012).

Bootstrap

Es un *framework* originalmente creado por Twitter, que permite crear interfaces Web con CSS y JavaScript, cuya particularidad es la de adaptar la interfaz del sitio Web al tamaño del dispositivo en que se visualice. Esta técnica de diseño y desarrollo se conoce como *responsive design* (en inglés) o diseño adaptativo (García, 2015).

El beneficio de usar *responsive design* en un sitio Web, es principalmente que el sitio Web se adapta automáticamente al dispositivo desde donde se acceda. Lo que se usa con más frecuencia es el uso de *Media Queries* (consultas), que es un módulo de CSS3 que permite la representación de contenido para adaptarse a condiciones como la resolución de la pantalla y si trabajas las dimensiones de tu contenido en porcentajes, puedes tener una web fluida capaz de adaptarse a casi cualquier tamaño de forma automática (Carpenter, y otros, 2000).

1.5.3 Entorno de desarrollo: *Visual Studio* 2015

Visual Studio (*Community Edition*), no es una versión recortada de un producto comercial y es gratis no solamente para el sector educativo, sino para desarrolladores individuales o en equipos de hasta cinco personas. El entorno de desarrollo integrado (IDE) de esta versión de Visual Studio soporta múltiples tipos de proyectos en un solo archivo solución dentro del IDE y tiene todas las características de productividad y extensibilidad del IDE, lo que significa que se puede usar Xamarin, ReSharper, VsVim y otras extensiones VSIX (López, 2014). Permite crear aplicaciones inteligentes, de manera rápida y con mejoras a las funciones populares, como la navegación de código, refactorización, y los arreglos de código, le ahorran tiempo y esfuerzo, sin importar el idioma o la plataforma.

Toda la experiencia de depuración y prueba incluye mejoras para ayudar a detectar y abordar problemas lo antes posible. Características como *Live Unit Testing, Exception Helpers* y *Run to Click* refuerzan el bucle de desarrollo de operaciones al reducir los riesgos de regresión y exponer inmediatamente la causa principal de los problemas nuevos.

1.5.4 Ajax

Ajax es una técnica de desarrollo web para lograr aplicaciones más interactivas, permitiendo actualizar parte de una página con información que se encuentra en el servidor sin tener que refrescar completamente la página. De modo similar podemos enviar información al servidor. Algunas de sus ventajas son: utiliza tecnologías ya existentes; soportada por la mayoría de los navegadores modernos; interactividad ya que usuario no tiene que esperar hasta que lleguen

los datos del servidor; portabilidad y mayor velocidad, esto debido que no hay que retornar toda la página nuevamente (Fernández, 2012).

1.5.5 Lenguajes de programación

C-Sharp

C# (pronunciado si sharp en inglés) es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma.NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270). C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común (Kovacs, 2007).

Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma.NET, similar al de Java, aunque incluye mejoras derivadas de otros lenguajes. El nombre C Sharp fue inspirado por la notación musical, donde '#' (sostenido, en inglés sharp) indica que la nota (C es la nota do en inglés) es un semitono más alto, sugiriendo que C# es superior a C/C++. Además, el signo '#' se compone de cuatro signos '+' pegados.

Aunque C# forma parte de la plataforma .NET, ésta es una API, mientras que C# es un lenguaje de programación independiente diseñado para generar programas sobre dicha plataforma. Ya existe un compilador implementado que provee el marco Mono - DotGNU, el cual genera programas para distintas plataformas como Windows, Unix, Android, iOS, Windows Phone, Mac OS y GNU/Linux.

HTML5

Hyper Text Markup Language, versión 5 es la quinta revisión del lenguaje de programación de la *World Wide Web*. Esta nueva versión reemplaza al X-HTML, corrige los problemas que los desarrolladores web encuentran, así como rediseñan el código, la presente versión se actualiza en función de las nuevas necesidades que demanda la web en la actualidad (canton, 2010).

Puede ser considerado como piedra angular de la web semántica. Presenta una serie de ventajas frente al HTML tradicional, como la capacidad de ordenar semánticamente el contenido del documento con etiquetas como *nav, header, section, footer*; incrustar directamente elementos multimedia como audios,

videos y canvas 2d y 3d; un nuevo grupo de tipos de entrada de datos para formularios con validación sin JavaScript; soporte de etiquetas para manejo de grandes cantidades de datos (*Datagrid, Details, Menu y Command*) que permiten generar tablas dinámicas que pueden filtrar, ordenar y ocultar contenido en el cliente, (en dependencia del nivel de implementación de HTML5 del navegador utilizado) (Consortium, 2014)

CCS

Hojas de Estilo en Cascada (*Cascading Style Sheets*), es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. Se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores web controlar el estilo y el formato de múltiples páginas Web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento.

JavaScript

Es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos,[3] basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente (*client-side*), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas aunque existe una forma de JavaScript del lado del servidor(*Server-side* JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente *widgets*) es también significativo (Domínguez, 2004).

1.6 Conclusiones parciales del capítulo

Luego de estudiar los antecedentes del proceso objeto de estudio y los aspectos generales de la entidad, así como las herramientas y tecnologías y la metodología de desarrollo de software, se concluye que:

1. Los sistemas existentes no resuelven la situación problémica planteada.
2. La combinación de las herramientas de desarrollo *Microsoft Visual Studio 2015* y *ReSharper* 8.0, utilizando las tecnologías antes mencionadas y la metodología XP, es la apropiada para desarrollar la aplicación que dará solución al problema de esta investigación.
3. El uso de últimas versiones de herramientas componentes y estándares le brinda al sistema un mayor tiempo de vida útil.
4. Se demuestra la necesidad de la utilización de software libre como una tendencia a asumir.
5. Para el desarrollo de las etapas de la Ingeniería de Software es factible utilizar XP.

En sentido general se ha contribuido a la mejor comprensión del objeto de estudio y se han establecido las bases para las siguientes fases de la investigación.

Capítulo II: Análisis, diseño y construcción de la solución propuesta

2.1 Introducción

En el presente capítulo se describen los principales elementos de la propuesta utilizando la Metodología de Programación Extrema (XP) y se presenta al equipo de trabajo. De igual forma se realiza el análisis del funcionamiento a través de las Historias de Usuarios (HU) y se muestran las iteraciones; así como el plan de entrega. Además, se efectúa un estudio de factibilidad mediante análisis de costo y beneficios, determinándose los beneficios tangibles e intangibles del desarrollo de la propuesta.

2.2 Etapa de planificación

La planificación es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario, y, asociadas a éstas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entregas, o "*Release Plan*" (Joskowicz, 2008).

Esta etapa persigue como propósito de lograr una eficiente organización de la formulación inicial del problema y brindar una solución deseada se realiza la planificación. Para ello se desarrollan las Historias de Usuario (UH), que tienen su sustento en las exigencias del cliente hacia el software. Estas constituyen el punto inicial a partir del cual se desarrolla el resto del proyecto. Se estima las entregas y el tiempo necesarios teniendo en cuenta retrasos inesperados o cambios que podrían ocurrir durante su confección.

2.3 Equipo y roles de trabajo

De acuerdo a la metodología, la conformación del equipo de trabajo puede variar dependiendo del proyecto que se esté ejecutando. Se reafirma el principio de 40 horas de trabajo semanales, un tiempo prudencial de descanso cada dos o tres aproximadamente. La simplicidad es la base de la programación extrema. Se simplifica el diseño para agilizar el desarrollo y facilitar el mantenimiento. Unos diseños complejos del código junto a sucesivas modificaciones por parte de

diferentes desarrolladores hacen que la complejidad aumente exponencialmente. La programación debe ser en pequeñas versiones poco a poco, en caso de algún fallo se efectúa la reprogramación del código sin que presente problemas la funcionalidad inicial. Es fundamental la retroalimentación con el cliente, al estar el cliente integrado en el proyecto, su opinión sobre el estado del proyecto se conoce en tiempo real.

En la tabla 2.1 se describe la aplicación al equipo de trabajo formado por dos programadores el encargado de prueba y dos gestores.

Tabla 2.0.1 Equipo de trabajo y roles

Miembros	**Roles**
Anniel Marrero Santana	Programador, Encargado de Pruebas, Encargado de Seguimento
Liz Pérez Martínez	Entrenador, Gestor, Consultor
Decana de la Facultad de Ciencias Económicas	Cliente, encargado de Pruebas

Fuente: Elaboración propia.

2.4 Historias de Usuario iniciales

Las Historias de usuarios sustituyen a los documentos de especificación funcional, y a los casos de uso. Estas historias se encuentran escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. Las historias de usuario poseen los detalles mínimos que se requieren para que los programadores puedan realizar una estimación poco riesgosa del tiempo que llevará su desarrollo.

La escala equivalente a la prioridad en el negocio es:

- ✓ **Alta**: Se le asigna a las Historias de Usuario que responden a funcionalidades esenciales en el desarrollo del proyecto, el cliente las define como primordiales.
- ✓ **Media**: Otorgadas a las Historias de Usuario no tienen una afectación directa sobre el proyecto en curso que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación directa sobre el proyecto que se esté desarrollando.

- ✓ **Baja**: Se le otorga a las Historias de Usuario que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura y no tienen nada que ver con el proyecto en desarrollo.

La escala nominal de riesgo en desarrollo es:

- ✓ **Alta:** Cuando para la implementación de la Historia de Usuario se considera la posible existencia de errores que lleven a inoperatividad del código.
- ✓ **Media**: Cuando pueden aparecer errores en la implementación de la Historia de Usuario que puedan retrasar la entrega de la versión.
- ✓ **Baja**: Cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto.

En la tabla 2.2 se muestra una relación de las HU que se plantearon inicialmente.

Tabla 2.0.2 Historias de Usuario iniciales

No	Nombre	Prioridad	Riesgo	Iteraciones	Ptos estimados
1	Diseño de la interfaz de usuario.	Alta	Alto	1	1
2	Gestionar usuarios.	Alta	Medio	1	1
3	Gestionar Trazas	Alto	Alto	1	1
4	Gestionar Propuesta Empresa	Alto	Alto	2	1
5	Gestionar Propuesta Investigación	Alto	Alto	2	1
6	Gestionar Soluciones	Alto	Alto	2	1
7	Gestionar Tareas	Alto	Alto	3	1
8	Gestionar Solicitud Estudiante	Alto	Alto	3	1
9	Gestionar Solicitud Profesor	Alto	Alto	3	1
10	Gestionar Profesores	Alto	Alto	4	1
11	Gestionar Línea de Investigación	Alto	Alto	4	1
12	Gestionar Facultad	Alto	Alto	4	1
13	Gestionar Departamento	Alto	Alto	5	1
14	Gestionar Tipo Alcance	Alto	Alto	5	1
15	Gestionar Grado Científico	Alto	Alto	5	1

16	Gestionar Categoría Docente	Alto	Alto	6	1
17	Mostrar notificaciones	Alto	Alto	6	1

Fuente: Elaboración propia.

A continuación, se muestran algunas descripciones de HU:

Tabla 2.0.3 HU 1

Historia de usuario	
Número: 1	**Usuario:** todos
Nombre de Historia: Diseño de la interfaz de usuario	
Prioridad en el negocio: Alta	**Riesgo en desarrollo**: Alta
Puntos estimados: 1	**Iteración asignada**: 1
Programador responsable: Anniel Marrero Santana	
Descripción: Se muestra la vista de la interfaz principal en dependencia del rol del usuario que se ha registrado en el sistema.	

Fuente: Elaboración propia.

Tabla 2.0.4 HU 2

Historia de usuario	
Número: 3	**Usuario**: Administrador
Nombre de Historia: Gestionar Usuarios	
Prioridad en el negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Anniel Marrero Santana	
Descripción: Se permite al administrador gestionar los usuarios y asignar los roles correspondientes	

Fuente: Elaboración propia.

Tabla 2.0.5 HU 4

Historia de usuario	
Número: 5	**Usuario:** Todos

Nombre de Historia: Gestionar Propuesta Empresa	
Prioridad en el negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Anniel Marrero Santana	
Descripción: Permite a los usuarios buscar, ordenar, ver detalles así como crear, editar, eliminar las Propuesta Empresa, según su rol.	

Fuente: Elaboración propia.

Tabla 2.0.6 HU 5

Historia de usuario	
Número: 6	**Usuario:** Todos
Nombre de Historia: Gestionar Propuesta Investigación	
Prioridad en el negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Anniel Marrero Santana	
Descripción: Permite a los usuarios buscar, ordenar, ver detalles así como crear, editar, eliminar las Propuesta Investigación, según su rol.	

Fuente: Elaboración propia.

2.5 Planificación de iteraciones

Las funcionalidades son desarrolladas en esta fase y se genera al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración. Como las historias de usuario no tienen suficiente detalle como para permitir su análisis y desarrollo, al principio de cada iteración se realizan las tareas necesarias de análisis, recabando con el cliente todos los datos que sean necesarios. El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo. Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance.

En la ilustración 2.1 se muestran las tres iteraciones y la cantidad de semanas en los que se demora el desarrollo de cada tarea.

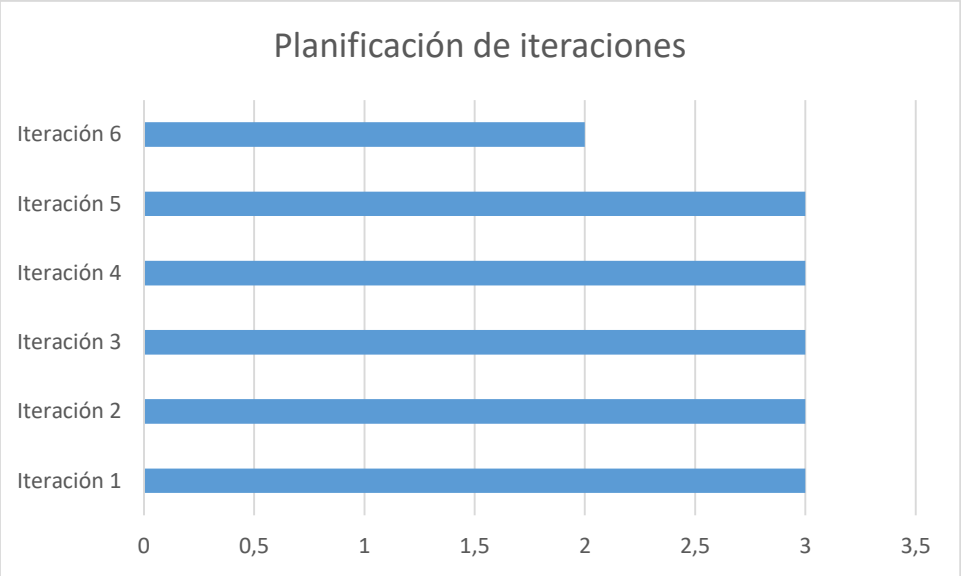

Ilustración 2.1 Planificación de las Iteraciones en semanas

Fuente: Elaboración propia.

2.6 Etapa de diseño

En XP solo se diseñan aquellas historias de usuario que el cliente ha seleccionado para la iteración actual por dos motivos: por un lado, se considera que no es posible tener un diseño completo del sistema y sin errores desde el principio. El segundo motivo es que, dada la naturaleza cambiante del proyecto, el hacer un diseño muy extenso en las fases iniciales del proyecto para luego modificarlo, se considera un desperdicio de tiempo.

Es importante resaltar que esta tarea es permanente durante la vida del proyecto partiendo de un diseño inicial que va siendo corregido y mejorado en el transcurso del proyecto.

2.6.1 Tareas a desarrollar

La tabla 2.7 muestra un resumen de la cantidad de tareas correspondientes a cada HU.

Tabla 2.0.7 Tareas de Iteración

No	Nombre HU	No	Tarea de Iteración	Iteraciones
1	Diseño de la interfaz de usuario.	1	Interfaz principal	1
2	Gestionar usuarios.	4	Gestionar usuario	1
		5	Autentificarse	

3	Gestionar Trazas	6	Listar historial Usuarios	1
4	Gestionar Propuesta Empresa	7	Crear	2
		8	Editar	
		9	Eliminar	
5	Gestionar Propuesta Investigación	10	Crear	2
		11	Editar	
		12	Eliminar	
6	Gestionar Soluciones	13	Crear	2
		14	Editar	
		15	Eliminar	
7	Gestionar Tareas	16	Crear	3
		17	Editar	
		18	Eliminar	
8	Gestionar Solicitud Estudiante	19	Crear	3
		20	Edita	
		21	Eliminar	
9	Gestionar Solicitud Profesor	22	Crear	3
		23	Editar	
		24	Eliminar	
10	Gestionar Profesor	25	Crear	4
		26	Editar	
		27	Eliminar	
11	Gestionar Facultad	28	Crear	4
		29	Editar	
		30	Eliminar	
12	Gestionar Departamento	31	Crear	4
		32	Editar	
		33	Eliminar	
13		34	Crear	5

	Gestionar Líneas de Investigación	35	Editar	
		36	Eliminar	
14	Gestionar Tipo Alcance	37	Crear	5
		38	Editar	
		39	Eliminar	
15	Gestionar Grado Científico	40	Crear	5
		41	Editar	
		42	Eliminar	
16	Gestionar Categoría Docente	43	Crear	6
		44	Editar	
		45	Eliminar	
17	Mostrar Notificaciones	46	Mostrar Notificaciones	6

Fuente: Elaboración propia.

A continuación, se relacionan algunas tareas de iteración a realizar durante el desarrollo de este proyecto:

Tabla 2.0.8 TI 7

Tarea de iteración	
Número: 7	**No Historia: 4**
Nombre de Tarea: Crear Propuesta de Empresa	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 0.3
Programador responsable: Anniel Marrero Santana	
Descripción: Permite a los usuarios crear sus propuestas	

Fuente: Elaboración propia.

Tabla 2.0.9 TI 14

Tarea de iteración	
Número: 14	**No Historia: 7**
Nombre de Tarea: Editar Solución	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 0.3
Programador responsable: Anniel Marrero Santana	

Descripción:
Permite al Admin o AdminInvestigacion editar una solución

Fuente: Elaboración propia.

Tabla 2.10 TI 18

Tarea de iteración	
Número: 18	**No Historia: 8**
Nombre de Tarea: Eliminar Tareas	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 0.3
Programador responsable: Anniel Marrero Santana	
Descripción: Permite eliminar una tarea que no tenga ni estudiantes ni profesores participando en ella.	

Fuente: Elaboración propia.

Tabla 2.11 TI 46

Tarea de iteración	
Número: 46	**No Historia: 17**
Nombre de Tarea: Mostrar Notificaciones	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 1
Programador responsable: Anniel Marrero Santana	
Descripción: Permite Mostrar Notificaciones al admin de investigación para ser revisadas.	

Fuente: Elaboración propia.

2.6.2 Tarjetas de Clase, Responsabilidad y Colaboración.

Se muestras algunas tarjetas de Clase, Responsabilidad y Colaboración (CRC) creadas:

Tabla 2.12 Tarjeta CRC Usuarios

Usuarios
Superclase:

Subclase:	
Descripción: En esta clase se guardan los elementos referidos a los usuarios.	
ATRIBUTOS	
IdUsuario	String
NombreUsuario	String
Contraseña	String
Rol	String

Fuente: Elaboración propia.

Tabla 2.13 Tarjeta CRC Elementos

Propuesta Entidad	
Superclase:	
Subclase:	
Descripción: En esta clase se guardan los elementos referidos a la **Propuesta Entidad**	
Atributos	
Id	Int
Nombre Empresa	String
Teléfono	String
Correo	String
Situación Problémica	String
Descripción	String
Pago Servicio	Bool
Sector	(comboBox)

Fuente: Elaboración propia.

2.7 Análisis de factibilidad

Al desarrollarse un software es preciso saber si será factible o no su producción, por lo que hay que tener en cuenta los costos y beneficios que traerá consigo. Es necesario también realizar una estimación del esfuerzo, el tiempo de desarrollo y la cantidad de personas que participarán para poder determinar eficazmente si resulta beneficioso el desarrollo, aunque es importante señalar que solo se habla de una estimación.

2.7.1 Estimación del costo de desarrollo

En el momento de la planificación del proyecto, es casi imposible determinar el número de líneas de código que poseerá la aplicación, siendo el método tradicional para calcular los costos COCOMO (Modelo Constructivo de Costos, por su acrónimo del inglés *COnstructive COst MOdel*) poco idóneo en este caso, ya que está orientado a la magnitud del producto final, midiendo el tamaño del proyecto, en líneas de código principalmente. Dentro de sus principales inconvenientes podemos citar:

- ✓ Los resultados no son proporcionales a las tareas de gestión ya que no tiene en cuenta los recursos necesarios para realizarlas.
- ✓ Se puede desviar de la realidad si se indica mal el porcentaje de líneas de comentarios en el código fuente.
- ✓ Es un tanto subjetivo, puesto que está basado en estimaciones y parámetros que pueden ser "vistos" de distinta manera por distintos analistas que usen el método.
- ✓ Se miden los costes del producto, de acuerdo a su tamaño y otras características, pero no la productividad.

Sin embargo, ya que se emplea XP como metodología para el desarrollo del software, se tiene un estimado del tiempo de desarrollo del mismo, aprovechando que la fórmula de Bohem comprende este parámetro (Costo = Cantidad de Hombres * Salario Medio * Tiempo de Desarrollo), es posible obtener una estimación del salario del autor. Haciendo los cálculos pertinentes tenemos, considerando como valor promedio para el salario mensual por hombres $ 365.00, un desarrollador y un tiempo de desarrollo estimado de 4 meses, al sustituir y calcular se obtiene un costo de $ 1460.

2.7.2 Beneficios tangibles e intangibles

Con la implantación del software para la gestión de la información se hace más sencilla la actividad de todos los implicados; reduciendo además, los errores humanos tan comunes en la manipulación de la información.

La herramienta desarrollada en un ambiente web fácil de manipular y administrar, permitirá ahorrar tiempo, más confiabilidad y seguridad en la interactividad entre los usuarios. La gestión de los datos posibilitará una mejor organización de la

información, logrando de forma segura la integridad, extracción, manipulación y persistencia de los datos.

Se gestionan las propuestas de las empresas y problemas de investigación, son ofertadas las tareas que tributan a una solución en particular a estudiantes y profesores de la UM de una manera interactiva, eficiente y eficaz.

2.7.3 Análisis de costos y beneficios

Se determinó como factible el desarrollo y puesta en marcha del sistema propuesto, pues este reduce significativamente el tiempo de búsqueda y facilita la gestión de la información concerniente a la UM y a las empresas del territorio. Dado que centraliza la misma de forma tal que los estudiantes, profesores, trabajadores y demás interesados tengan acceso a esta a través de las redes informáticas de la universidad e internet.

Con estos beneficios y la estimación de costos realizada anteriormente, se evidencia que la solución propuesta presenta una buena relación entre costo y beneficios.

2.8 Conclusiones del capítulo

Una vez descrita la propuesta de solución al problema científico de esta investigación, se concluye que:

1. Se dispone de un equipo de trabajo capaz de solucionar el problema existente.
2. La planificación inicial es necesaria para tener una visión general del problema que se enfrenta y pensar desde el inicio en soluciones eficaces.
3. Las HU iniciales permitieron especificar los requisitos funcionales de la aplicación.
4. El cliente conformó las HU de acuerdo a sus necesidades.
5. La planificación de las iteraciones se realiza según las características de las HU, teniendo en cuenta la prioridad y el riesgo de desarrollo.
6. El plan de entrega de los módulos de la aplicación se correspondió con el final de cada iteración.
7. La estimación del costo del software se realizó a partir de la fórmula de Bohem, apreciándose un valor racional del mismo.

8. El beneficio de este sistema es tanto para la UM, como para el sector empresarial.
9. El sistema aporta un beneficio mayor que su costo de producción.

Capítulo III Validación de la Solución Propuesta

3.1 Introducción

En este capítulo se comprueba el cumplimiento de los requerimientos iniciales, con este propósito se desarrollan los casos de prueba, las estrategias de prueba y de forma similar se estudian las pruebas de caja negra a través de las cuales se efectúan las pruebas de aceptación y se analizan los resultados obtenidos.

Se presenta un software que responde a los objetivos planteados en la introducción de este trabajo, y este capítulo estará dedicado esencialmente a la validación dicho software.

3.2 Pruebas

La prueba de software se puede definir como una "actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas, registrándose los resultados obtenidos. Seguidamente se realiza un proceso de evaluación en el que los resultados obtenidos se comparan con los resultados esperados para localizar fallos en el software". (Pressman, 2010)

3.2.1 Pruebas de aceptación

Las pruebas de aceptación son las realizadas por el cliente y usuarios finales de la aplicación. Permiten probar las funcionalidades que exige el cliente. Luego de haber superado las pruebas de aceptación podrá considerarse que la aplicación es apta para el uso y despliegue dentro del proyecto.

Las pruebas persiguen como objetivo, llevar a cabo el proceso de ejecución de un programa con la intención de descubrir un error. Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar al menos un error no descubierto hasta entonces (Pressman, 2010).

Como resultado de las pruebas de aceptación se obtendrá una valoración del funcionamiento del sistema, que puede ser satisfactoria o no. Los campos que comprenden estas pruebas son los siguientes:

- ✓ Código: Constituye un indicador de la prueba realizada y es sugerente al nombre de la prueba a la que hace referencia.
- ✓ HU: Contiene el nombre de la historia de usuario a la que hace referencia la prueba a realizar.

- ✓ Nombre: Nombre que se le asigna a la prueba a realizar.
- ✓ Descripción: Se detalla la funcionalidad que se desea probar.
- ✓ Condiciones de Ejecución: Muestra las condiciones necesarias y, que deben ser satisfechas antes de efectuar la ejecución del caso prueba para desarrollar el mismo y obtener los resultados esperados.
- ✓ Entradas / Pasos de Ejecución: Se describen los pasos seguidos en el desarrollo de la prueba, para ello se consideran todas las entradas que hace el usuario, con el propósito de conocer si se obtiene el resultado esperado.
- ✓ Resultado esperado: Se define el resultado que se espera obtener con la prueba realizada.
- ✓ Evaluación de la prueba: Una vez efectuada la prueba y obtenido su resultado se expone una valoración de la misma. Se clasifican en:
    - Satisfactoria: Cuando el resultado de la prueba es exactamente el esperado por el usuario.
    - Parcialmente satisfactoria: cuando el resultado no es completamente el esperado por el cliente o usuario de la aplicación y muestra resultados erróneos o fuera de contexto.
    - No satisfactoria: Cuando el resultado de la prueba realizada genera un error de codificación en la aplicación o muestra como resultado elementos no deseados o fuera de contexto, trayendo como consecuencia que la funcionalidad requerida por el cliente no tenga resultado, lo que invalida también la HU.

Las pruebas de aceptación se llevan a cabo al efectuar los pasos siguientes:

1. Se redactan los casos de prueba teniendo en cuenta el orden de las historias de usuario y los niveles de prioridad dados a las funcionalidades.
2. Se planifica con el cliente de cuándo y cuáles pruebas serán llevadas a cabo.
3. Se reúnen los miembros del proyecto seleccionados para realizar las pruebas.
4. Se completan cada uno de los campos de las tablas de las pruebas de aceptación con el resultado de la prueba.

A continuación, se muestran las pruebas de aceptación, alternadas con vistas de los mensajes de errores que fueron detectados a medida que se fueron realizando las mismas o que forman parte de la validación:

Tabla 3.1 PA 1

Pruebas de aceptación	
Número Caso de Prueba: 1	**No Historia**:1
Nombre Caso de Prueba: Test Interfaz Principal	
Descripción: Verificar que se muestren las interfaces visuales implementadas	
Condiciones de ejecución: Esté corriendo la aplicación.	
Entradas: Interfaces de la aplicación.	
Resultado esperado: Se muestren las interfaces visuales de la aplicación.	
Evaluación: Prueba satisfactoria	

Fuente: Elaboración propia.

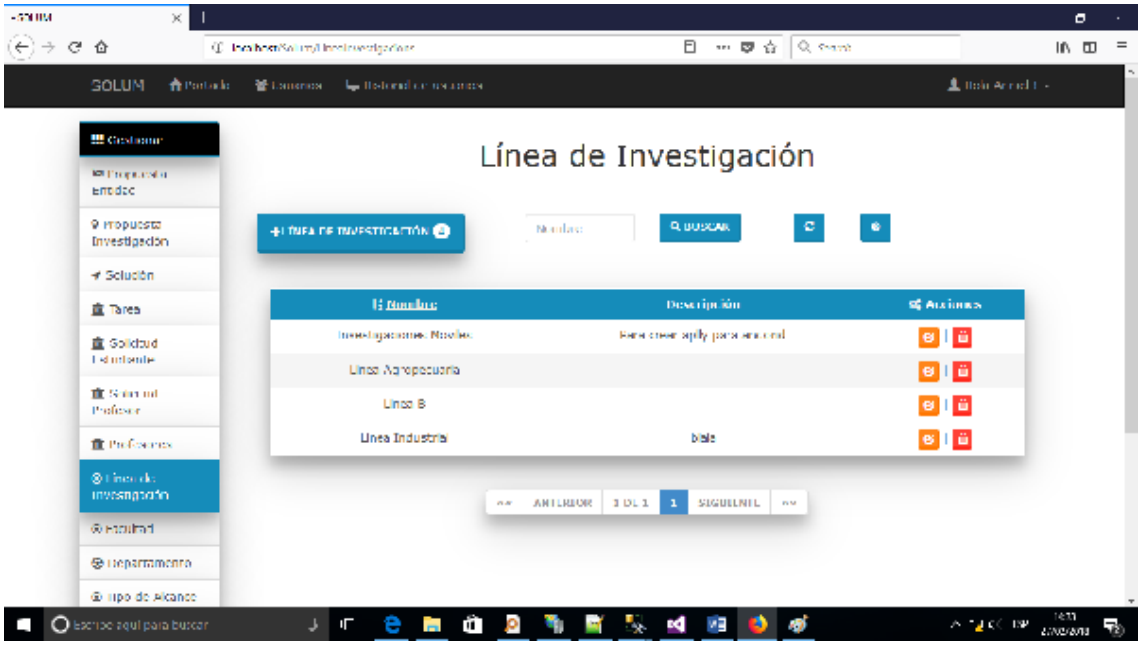

Ilustración 3.1 Resultado de caso de prueba No. 1

Fuente: Elaboración propia.

Tabla3.2 PA 2

Pruebas de aceptación	
Número Caso de Prueba: 2	**No Historia:** 2

Nombre Caso de Prueba: Test Base de Datos
Descripción: Verifica el funcionamiento de la base de datos
Condiciones de ejecución: Estar conectado a la base de datos.
Entradas: Valores para leer o escribir en la base de datos por ejemplo: propuesta entidad, usuario, contraseña, etc.
Resultado esperado: Se muestran o guardan los datos correctamente.
Evaluación: Prueba satisfactoria

Fuente: Elaboración propia.

Tabla 3.3 PA 3

Pruebas de aceptación	
Número Caso de Prueba: 3	**No Historia**: 3
Nombre Caso de Prueba: Test Autentificar	
Descripción: Verificar que se autentifique un usuario correctamente en el sistema.	
Condiciones de ejecución: Estar conectado a la Base de Datos	
Entradas: Nombre de usuario y contraseña	
Resultado esperado: Se otorgan al usuario correspondiente los permisos que le corresponden a su rol.	
Evaluación: Prueba satisfactoria	

Fuente: Elaboración propia.

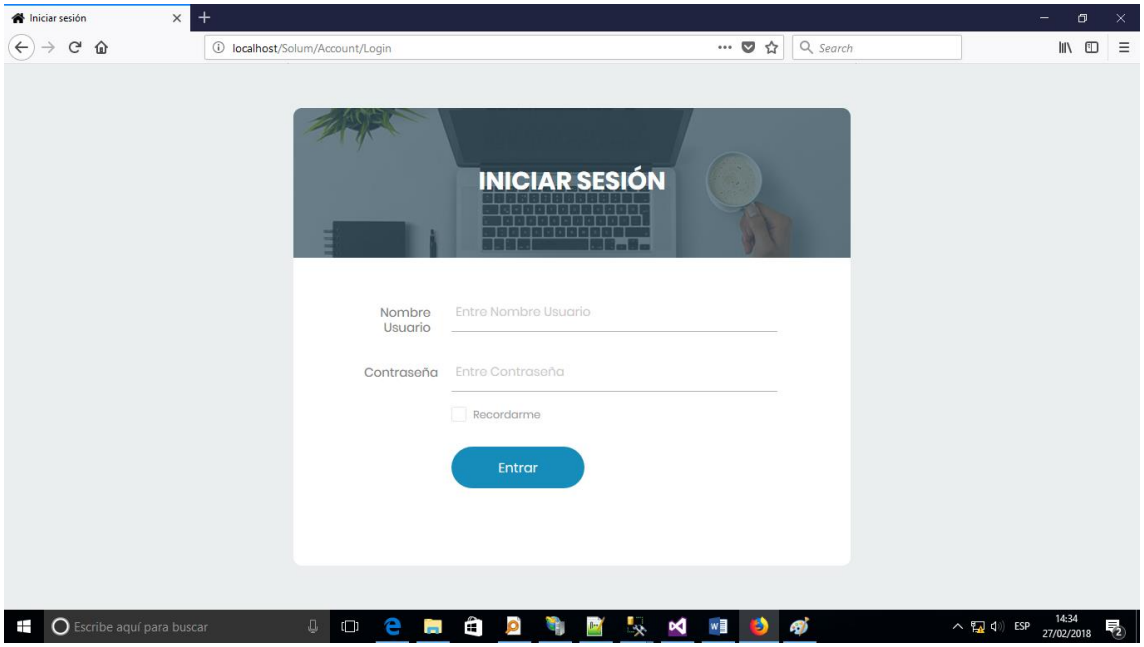

Ilustración 3.2 Resultado de caso de prueba No. 3

Fuente: Elaboración propia.

Tabla 3.4 PA 4

Pruebas de aceptación	
Número Caso de Prueba: 4	**No Historia:**4
Nombre Caso de Prueba: Test gestionar propuesta Entidad	
Descripción: Verificar que se pueda editar o eliminar una propuesta Entidad correctamente.	
Condiciones de ejecución: Estar conectado a la base de datos.	
Entradas: Elementos de la propuesta Entidad.	
Resultado esperado: Se edita o elimina correctamente la propuesta Entidad	
Evaluación: Prueba satisfactoria	

Fuente: Elaboración propia.

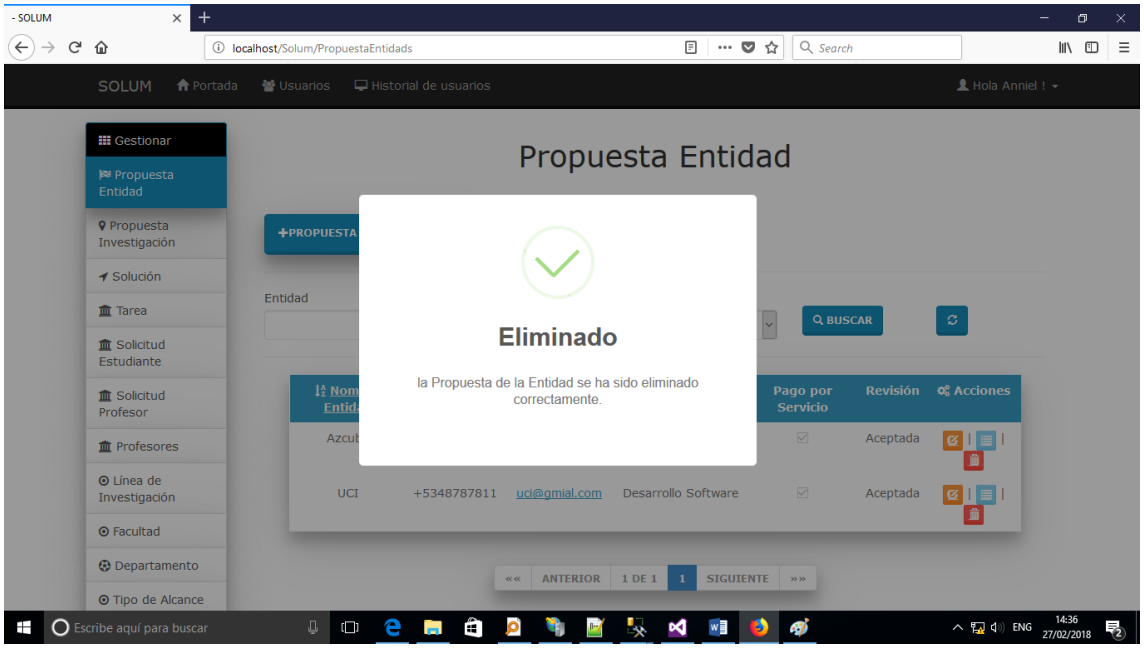

Ilustración 3.3 Resultado de caso de prueba No. 4

Fuente: Elaboración propia.

3.2.2. Pruebas de caja blanca

La implementación de este tipo de pruebas requiere habilidades de programación, un conocimiento del *framework* de desarrollo y un cierto conocimiento funcional que permita conocer qué misión tienen determinadas clases y métodos. Son pruebas que examinan minuciosamente los detalles internos de la realización (estructura interna del componente).

3.2.3. Pruebas unitarias

Las pruebas unitarias consisten en identificar y corregir problemas en una parte del código de una aplicación. El objetivo de esta prueba es evaluar de manera individual la correcta operación de cada módulo de software.

Después de desarrollar todo un proceso de pruebas con un nivel medio de sencillez se lograron resultados satisfactorios, pues tras la detección de diferentes errores, obtenidos fundamentalmente con las realizadas, se solucionaron varios problemas que impedían el cumplimiento de los requisitos fundamentales del sistema en cuestión.

Las primeras pruebas fueron planeadas y ejecutadas en módulos individuales del programa y a medida que fueron avanzando se desplazaron a módulos integrados, hasta que finalmente llegaron al sistema completo y se logró obtener un software cuyas funciones se encuentra en correspondencia con las especificaciones acordadas y que además cumple con los requerimientos de rendimiento.

El desarrollo del sistema cumple las expectativas trazadas al inicio del proyecto y satisface al cliente en su totalidad ya que se logra facilitar este proceso.

3.3 Análisis de los resultados

Terminado el proceso de desarrollo, en el cual se cumplió todo el cronograma de trabajo gracias a la constante colaboración entre cliente y desarrollador, se obtuvo un sistema que cumple a cabalidad con las expectativas del cliente. Influyó en este resultado el uso de poderosas herramientas de desarrollo como Microsoft Visual Studio 2015, que permitió reducir el tiempo de desarrollo estimado. Al disponer de más tiempo en el cronograma, desarrollador y cliente se dedicaron a probar y perfeccionar las funcionalidades del sistema, obteniendo como resultado, una poderosa y valiosa herramienta al servicio de la UM y de las empresas del territorio.

Como resultados finales se obtuvo una aplicación web, con una apariencia agradable y fácil de usar. La planificación inicial se cumplió a un 100 %, se utilizaron herramientas actuales para su desarrollo, y el plan de entrega fue cumplido con éxito. El cliente quedó complacido con el trabajo y se espera que se logren alcanzar mayores resultados.

A continuación se muestran imágenes de las interfaces de usuario:

Ilustración 3.4 Formulario para crear propuesta Entidad

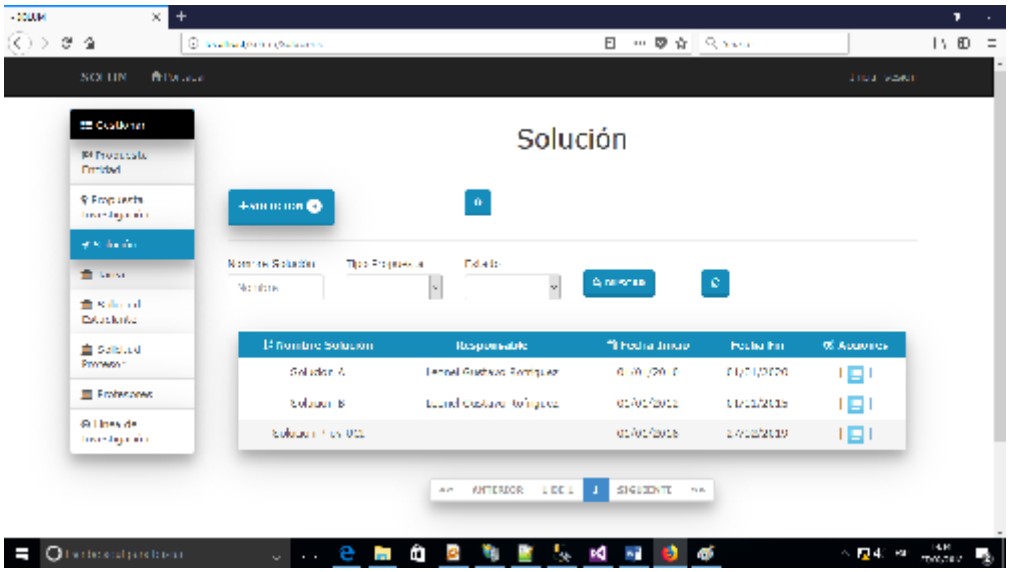

Ilustración 3.5 Listado de soluciones

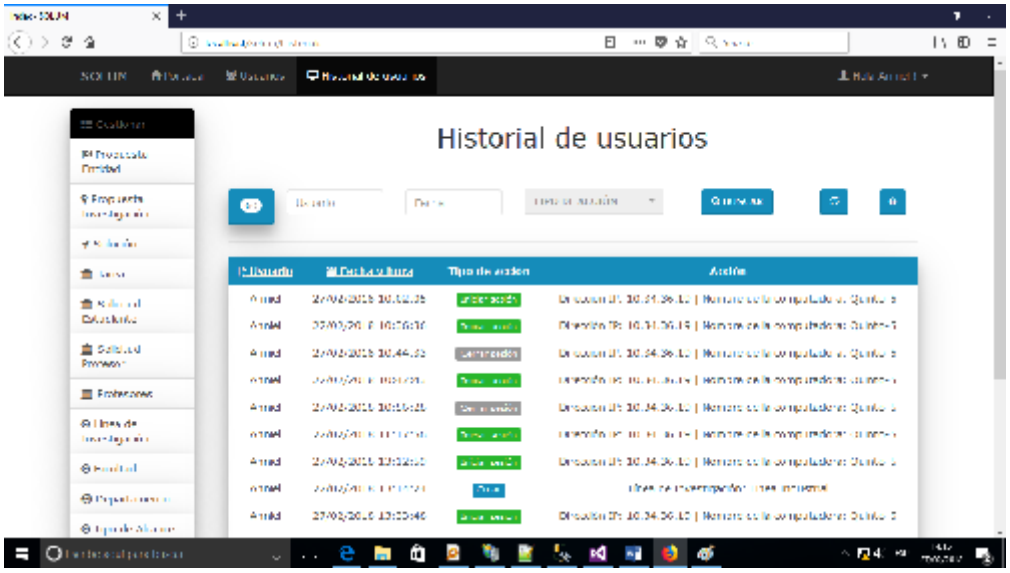

Ilustración 3.6 Interfaz de usuario de administrador

3.4 Conclusiones del capítulo

Una vez construida la propuesta de solución al problema científico de esta investigación a partir de la metodología XP, y analizado los resultados de la misma a partir de las pruebas funcionales, se concluye que:

1. La planificación fue acertada, pues permitió el desarrollo del sistema según el cronograma y cumpliendo con los objetivos del cliente al iniciar esta investigación.
2. La realización de pruebas de aceptación durante el desarrollo del sistema permitió tanto al cliente como al desarrollador encontrar defectos y errores en el sistema y su rápida corrección.
3. A partir del diseño de la propuesta y con el uso de las tecnologías adecuadas para su implementación se obtuvo un sistema que cumple a cabalidad con las expectativas del cliente.

Conclusiones

Como resultado de esta investigación quedaron satisfechos los objetivos trazados arribando a las siguientes conclusiones:

1. El estudio realizado sobre los antecedentes, el estado actual de la temática, la bibliografía y documentos relacionados con el objeto de estudio, permitió contar con los elementos necesarios para dar solución a la problemática planteada.
2. Los sistemas automatizados encontrados, vinculados al tema no le dan solución al problema planteado por lo que no es factible su utilización.
3. Se utilizaron las herramientas de software más factibles para la construcción de la solución.
4. Se implementó el sistema cumpliendo con el cronograma de desarrollo planteado al cliente.
5. Se realizó la estimación de costo de implementación del sistema y el estudio de factibilidad, arrojando como resultado la factibilidad de la realización del sistema informático.
6. La realización de las pruebas de aceptación al sistema permitió la detección de errores y la rápida corrección de los mismos.

Recomendaciones

Desde el punto de vista del alcance del presente trabajo y teniendo en cuenta el momento de desarrollo del mismo, se proponen las siguientes recomendaciones:

1. Continuar el desarrollo del sistema añadiéndole a este, otros requerimientos funcionales para aumentar el alcance del mismo.
2. Extender los beneficios del software a otras universidades del país.

Referencias Bibliográficas

Beck, K. 2000. *Extreme Programming Explained: Embrace Change. s.l.* s.l. : Addison-Wesley, 2000.
Brianza, M. 2006. *ASP .NET orientado al desarrollo de aplicaciones web.* Universidad Autónoma del Estado de Hidalgo. México : s.n., 2006.
Calero, M. 2003. Una explicación de la programación extrema(XP). [Online] 2003. [Cited: Mayo 3, 2017.] www.apolosoftware.com..
canton, A. 2010. *Manual de HTML5 en español.* 2010.
Carpenter, J. and Bithell, J. 2000. *Bootstrap confidence intervals: when , which,what?.A practical guide for medical statisticians.* s.l. : Statistics in medicine, 2000.
Chaffer, J. and Swedberg, K. 2010. *query reference guide: a comprehensive exploration of the popular javascript library.* s.l. : Packt Publishimg Ltd, 2010.
Consortium, Word Wide Web. 2014. Borrador actual de especificaciones de HTML5. [Online] 2014. [Cited: noviembre 20, 2017.] http://dev.w3.org/html5/spec/Overview..
Cubel, N. 2012. *Extreme Programming Laboratorio de Sistemas de Información .* Valencia : Universidad Politécnica de Valencia, 2012.
Domínguez, M. 2004. *Todo Programación.* Madrid : Editorial Iberprensa, 2004.
Fernández, D G. 2012. *Sistema Informático en plataforma libre para el desarrollo de las técnicas de comercio electrónico en la Tecnología.* 2012.
Fundora, l. 2012. *Herramienta web para la gestión de compra de materias.* Ingeniería Informática, Universidad de Matanzas. 2012. Trabajo de Diploma.
García, L. 2015. *Portal web educativo para el desarrollo de habilidades comunicativas en el idioma inglés.* Universidad de Ciencias Informáticas. 2015. Trabajo de Diploma.
Gittins, R. and Hope, S. 2001. *A study of Human Solutions in eXtreme Programming .* Bangor : Shool of Informatics University of Wales, 2001.
Goto, et al. 2014. *An Extreme Programming Method for Innovative Software Based on Systems Design and its Practical Study.* Japón : s.n., 2014. Vol. 5.
Hurtado, J., Julio, A. and Bastiarrica, C. 2005. Modelo de Procesos, Calidad y Mejoramiento . *Proyecto SIMEP-SW.* 2005.
Jeffries, R, Anderson, A and Hendrickson, C. 2001. *Extreme Programming Installed. s.l.* s.l. : Addison Wesley, 2001.
Joskowicz, J. 2008. *Reglas y Prácticas en Extreme Preogramming.* 2008.
Kovacs, J. 2007. C#.Net History Lesson. [Online] 2007. [Cited: noviembre 20, 2017.] http://jameskovacs.com/2007/09/07/cnethistory-lesson..
López, M. 2014. Visual Studio es ahora de código libre y gratuito. *UNOCERO.* [Online] noviembre 15, 2014. [Cited: noviembre 27, 2017.] https://www.unocero.com/author/morsa/.
Otegem, M. 2015. *Interview with Scott Guthrie, creator of ASP.NET.* 2015.
Pereira, A. 2014. *Software de gestión para el control y evaluación del entrenamiento deportivo.* Facultad de Ciencias de la Cultura Física y el Deporte, Universidad de Matanzas. 2014. Tesis para optar al grado de máster en Ciencias de la Cultura Física, el Deporte .
Poppendieck, M. 2003. *Lean Software Development: An Agile Toolkit for Software Development Managers.* s.l. : Addison Wesley, 2003.
Pressman, R. 2010. *Ingeniería de Software. Un enfoque práctico.* 2010.
Reynoso, B. 2012. *Métodos Ágiles en Desarrollo de Software,Introducción a la Arquitectura de Software. .* Buenos Aires : Universidad de Buenos Aires , 2012.

REFERENCIAS BIBLIOGRÁFICAS

Rivera, Y. 2015. *Sistema de control de Activos Fijos en la Entidad de Cultura.* Informática, Universidad de Matanzas. Matanzas : s.n., 2015. Trabajo de Diploma.
Wake, W. 2000. *Extreme Programming Explored.* 2000.