Universidad de las Ciencias Informáticas

Facultad 3

Sistema para la gestión de Proyectos Extensionistas

Trabajo de Diploma en opción al título de Ingeniería en Ciencias Informáticas

Autores:

Cesar Borges Prado

Francis Pozo González

Tutores:

MsC. Mailen Edith Escobar Pompa

Ing. Jorge Jesús Hidalgo Ruiz

La Habana, junio del 2019

"Año 61 del Triunfo de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser los autores del presente documento 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 _____ del año _____.

_____ _____

Autor: Cesar Borges Prado Autor: Francis Pozo González

_____ _____

MSc. Mailen Edith Escobar Pompa Ing. Jorge Jesús Hidalgo Ruiz

“Sueña y serás libre en espíritu… lucha y serás libre en vida”

Ernesto “Ché” Guevara

DEDICATORIA

César

A má, a pá… a toda mi familia y amigos.

Francis

A mis amigos y amigas.

A mi familia entera… gracias por darme la vida.

AGRADECIMIENTOS

Cesar

A Dasiel y Nai, aparecieron de la nada y con nuestras altas y bajas han sabido quedarse, gracias por estar ahí.

Para el tipo nuevo de la banda: Ale.

Má, eres una madre increíblemente genial, Vero y yo tenemos la mayor suerte del mundo por tenerte.

A mi hermanita del alma pues que decirte tampoco me pudo tocar una mejor.

Papá que lastima no puedas estar aquí conmigo. Gracias por tu sacrificio, tu entrega y tus ganas de sacarnos adelante.

Para mi tía loca, tiquismiquis, creativa y futura psicóloga.

A mi queridísima tutora Mailen gracias por escogernos, sé que te hemos dado trabajo, pero valió la pena, eres una persona increíble.

A mis compañeros de año sé que no llegue a conocerlos a todos, pero espero que les vaya genial en sus vidas profesionales y personales.

Francis

Me gustaría dar las gracias a todas aquellas personas que de una forma u otra me han acompañado durante todo este tiempo:

Quiero agradecer de manera muy especial a mi tutora… usted es la mejor. Gracias por permitirnos molestarla tanto y por su sabroso café.

A cada uno de los profesores, gracias por las enseñanzas. Estaré eternamente agradecido.

A mi maestra, la profe Dariela.

A mis amig@s (a los que están y a los que no) por los buenos momentos que compartimos, especialmente a es@s que se han convertido en mis herman@s… No tengo palabras para expresar todo lo que siento por ustedes.

A mis amores… gracias por darme calorcito y por ayudarme a entender en un 1% a las mujeres.

A mi familia entera por su amor… por ser mi hogar y por darme todo lo que tengo…

¡A todos… MUCHAS GRACIAS!

RESUMEN

La extensión universitaria constituye un proceso orientado a promover la cultura general e integral de la comunidad universitaria y su entorno social, la cual tiene entre sus salidas a los proyectos extensionistas. El objetivo del presente trabajo es desarrollar un sistema informático que contribuya a la mejorar las formas de organización y control de la información referente a estos proyectos.

Como parte de las tareas realizadas, se detalla el estado del arte referente a la extensión universitaria y las formas organizativas relacionadas con la gestión de proyectos. Se realizó un estudio de sistemas similares con el objetivo de identificar funcionalidades a tener en cuenta en la propuesta de solución. En correspondencia con la metodología de desarrollo de software XP, se identificaron, modelaron y describieron los procesos de negocio. Para la validación del sistema se realizaron pruebas de software, evidenciándose la correspondencia entre los objetivos propuestos y los resultados obtenidos.

PALABRAS CLAVE: extensión universitaria, proyecto extensionista, sistema informático.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El Ministerio de Educación Superior (MES) es la organización rectora de los estudios universitarios cubanos, la cual está integrada por una red de instituciones académicas y entidades de ciencia e innovación tecnológica distribuidas en todo el país. Este sistema, bajo la dirección del Partido Comunista de Cuba (PCC) y el Gobierno, fue creado en julio de 1976 con la misión de formar profesionales altamente calificados y comprometidos con la Revolución, para garantizar una elevada formación política, social, ideológica y cultural de la sociedad cubana (MES, 2004).

Para potenciar la formación cultural de los estudiantes, el MES puso en marcha el Programa Nacional de Extensión Universitaria (PNEU) en el año 2004, asumiéndola como *"un proceso orientado a la labor educativa, que promueva y eleve la cultura general integral de la comunidad universitaria y su entorno social"* (PNEU, 2004). Este programa, reconoce que entre las salidas del proceso extensionista se encuentran: los programas, proyectos, actividades, acciones y tareas, las cuales guardan una relación entre sí. El proyecto extensionista, en particular, se ha convertido en un eje articulador de la extensión universitaria y elemento clave para dinamizarla y promover las transformaciones que se requieren en este proceso (PNEU,2004).

La Universidad de las Ciencias Informáticas (UCI), como parte de esta red de instituciones académicas, impulsó el desarrollo de la actividad extensionista desde sus primeros años, de conjunto con la ejecución de la Batalla de Ideas y las nuevas tareas del MES. Esta institución promueve el desarrollo profesional a través de un modelo de formación caracterizado por el vínculo estudio-trabajo, la realización de tareas investigativas y actividades de carácter extensionista. El claustro de profesores y especialistas, trabaja por consolidar un entorno caracterizado por un ambiente intelectual, ético y estético, en el que la comunidad universitaria participe en las actividades docentes, culturales, deportivas, políticas y recreativas, para asumir de manera autónoma y responsable un estilo de vida culto y saludable (UCI, 2001).

La Facultad 3 de la universidad, se ha destacado por su labor en la actividad extensionista, obteniendo resultados relevantes a nivel de universidad. La Facultad cuenta con 5 proyectos propios: Desarrollo del profesional, Laboratorio de Hardware, Proyecto de Matemática y Física (MAFIS), ¿Quién quiere ser ingeniero? y Cinco Palmas.

El ciclo de realización de cada uno de estos proyectos comienza cuando un profesor, estudiante, departamento u otra disciplina de la universidad identifica una problemática que pueda ser resuelta a través de un proyecto extensionista, y solicita un permiso para su realización. Luego de todo el proceso administrativo, se ejecuta el proyecto si este finalmente es aprobado.

Una vez concluido, es necesario almacenar las evidencias las cuales dan prueba de la calidad, fecha, lugar y tipo de actividad que se haya realizado. Entre las evidencias que puede generar un proceso de extensión están los videos, las fotografías, las encuestas y toda la documentación referente a dicha actividad. Actualmente no existe una plataforma donde se puedan alojar todo lo relacionado con las evidencias, lo que trae consigo que parte de la información de los proyectos se pierda al no ser recogida en su debido momento. A raíz de ese problema, se ve afectada la disponibilidad y centralización de la información.

Dentro de la documentación asociada a los proyectos pueden encontrarse las plantillas de planificación, las encuestas y otros archivos importantes. Parte de esta es almacenada en grandes volúmenes y formatos, su mayoría en papel, lo que puede provocar pérdidas de información por deterioro o extravío, además de ser un riesgo considerable para la integridad y seguridad de los datos. Además de esto, algunos de los documentos de planificación han cambiado su estructura, lo que obliga a modificar las formas de gestionar la información.

Por otra parte, para llevar a cabo la planificación de las actividades de cada proyecto, el vicedecano de extensión y residencia debe reunirse personalmente con los coordinadores de los mismos para dar las orientaciones, lo que puede traer consigo una pérdida considerable de tiempo y retrasar otras tareas importantes.

La conservación de los resultados de los proyectos extensionistas constituye una parte importante de la memoria histórica de la institución y son valorados en los procesos de acreditación de la carrera y de la universidad, dando cumplimiento a la variable seis del MES referente al impacto social. Por ello, se considera importante su correcta gestión.

Debido a lo antes expuesto el **problema de investigación** queda formulado de la siguiente forma: ¿Cómo mejorar la disponibilidad y centralización de la información de los proyectos extensionistas?

Constituye el **objeto de estudio** de la presente investigación: la gestión de procesos en la Dirección de Extensión y Residencia de la UCI, enmarcando como **campo de acción** la

gestión de proyectos extensionistas en la Dirección de Extensión y Residencia de la Facultad 3.

Se ha trazado como **objetivo general**: Desarrollar un sistema informático que permita mejorar el control, centralización y disponibilidad de la información de los proyectos extensionistas.

A partir del objetivo general, se derivan los siguientes **objetivos específicos**:

- Elaborar el marco teórico de la investigación mediante el estudio de los sistemas para la gestión de información.
- Desarrollar el sistema para la gestión de la información de los proyectos extensionistas.
- Verificar el correcto funcionamiento de la herramienta y de las variables de la investigación.

Tareas a cumplir para el desarrollo de la investigación.

- Revisión bibliográfica acerca de las soluciones existentes para determinar fortalezas e insuficiencias en las mismas y valorar la viabilidad de su uso.
- Elaboración del marco teórico del tema de investigación.
- Estudio de las herramientas para el desarrollo de aplicaciones web.
- Análisis para definir las herramientas, métodos y metodologías de desarrollo de software.
- Identificación de los requisitos funcionales y no funcionales a tener en cuenta para el desarrollo del sistema para la gestión de proyectos extensionistas.
- Definición de las estrategias, técnicas y métodos como marco de la solución a desarrollar.
- Ejecución de las estrategias, técnicas y métodos para la implementación de la solución a desarrollar.
- Aplicación de las pruebas de software al sistema de gestión de proyectos extensionistas.
- Validación de las variables de la propuesta de solución.

Métodos de investigación científica.

Se definen en esta sección los métodos de investigación que ayudan a identificar la estrategia para el diseño de los experimentos. El método científico se puede clasificar en teóricos y empíricos, los cuales están dialécticamente relacionados.

Métodos teóricos:

- El método histórico-lógico: En el estudio de la evolución histórica y tendencias actuales de la Dirección de Proyectos Extensionistas.
- El método analítico-sintético: En el estudio de las tecnologías y las herramientas, para identificar las ventajas que puedan ser aprovechadas en el desarrollo de la presente solución.
- La modelación: Representación de las clases para el análisis y diseño de la solución propuesta mediante diagramas.

Métodos empíricos:

- La entrevista: Obtener información referente a los procesos que tienen lugar en el área de extensión universitaria, para formular los requisitos del sistema.

Estructura capitular de la investigación.

La presente investigación está estructurada en 3 capítulos:

- Capítulo 1: Fundamentación Teórica. Contiene el marco teórico-referencial de la investigación, donde se exponen los principales conceptos del problema, que son necesarios para entender los procesos de gestión de los proyectos extensionistas.
- Capítulo 2: Análisis y diseño de la solución propuesta. Se describen los procesos del negocio y los conceptos asociados al mismo. Además, se realiza el análisis y diseño de la propuesta de solución atendiendo a los requisitos funcionales y no funcionales.
- Capítulo 3: Implementación, pruebas y validación de la propuesta de solución. Se describe la fase de implementación del proyecto y la estrategia seguida para aplicar las pruebas al sistema.
- Posteriormente se presentan las conclusiones generales, recomendaciones, referencias bibliográficas y los anexos de la presente investigación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se analizan los conceptos que resultarán útiles para un mejor entendimiento de los términos asociados al problema planteado, además de un estudio de los sistemas similares y cómo tributan estos a la presente solución. Posteriormente, se fundamentan las tecnologías, herramientas y metodologías utilizadas en el desarrollo del sistema de gestión de proyectos extensionistas.

1.1. Conceptos fundamentales asociados al dominio del problema:

Como punto de partida de la presente investigación, se requiere un entendimiento de los conceptos enunciados a continuación.

Sistema de gestión: Un sistema de gestión o software de gestión, hace referencia a las aplicaciones informáticas en las que se apoyan las organizaciones, empresas o proyectos para desarrollar sus actividades. Estos sistemas soportan las funciones administrativas y de gestión, así como las comunicaciones; añadiendo valor a los productos y servicios de las organizaciones, empresas o proyectos (Encyclopedia of Computer Science, 2003).

Proyecto extensionista: Eje articulador de la gestión de la extensión universitaria y elemento clave para dinamizarla y promover las transformaciones que se requieren en este proceso. Constituye una unidad más operativa dentro del proceso de planeación del trabajo sociocultural universitario, pues ofrece tratamiento a situaciones y problemas específicos. Es una unidad mínima de asignación de recursos, que a través de un conjunto concreto de actividades, acciones y tareas pretende modificar o transformar una parcela de la realidad sociocultural disminuyendo o eliminando un déficit o solucionando un problema (PNEU, 2004).

Extensión universitaria: El Ministerio de Educación Superior de Cuba, define la extensión como: "*el proceso que tiene como objetivo promover la cultura en la comunidad intrauniversitaria y extrauniversitaria, para contribuir a su desarrollo cultural*" (PNEU, 2004).

Cultura: Toda producción material y espiritual del hombre, se expresa en todo el sistema de la actividad humana. Conjuntos de valores espirituales y materiales creados por la humanidad en el curso de la historia o nivel de desarrollo alcanzado por la sociedad en la

instrucción, la ciencia, la literatura, el arte, la moral, la filosofía y las instituciones correspondientes (PNEU, 2004).

Comunidad: Puede ser definida como el espacio físico ambiental, geográficamente delimitado, donde tiene lugar un sistema de interacciones sociopolíticas y económicas que producen un conjunto de relaciones interpersonales sobre la base de necesidades. Este sistema resulta portador de tradiciones, historia e identidades propias que se expresan en identificación de intereses y sentido de pertenencia que diferencian al grupo que integra dicho espacio ambiental de los restantes (PNEU, 2004).

Trabajo Cultural Comunitario: Es un conjunto de acciones estructuradas como proceso, gestado por y para la comunidad a partir de sus demandas, que favorecen una transformación comunitaria desde y hacia la cultura, como herramienta de impacto en la población y espacio al que va destinado con un impacto perdurable. Requiere de una base económica para su desarrollo, proponiéndose el cumplimiento de objetivos que respeten la identidad y diversidad del espacio y de las personas potenciando los saberes y habilidades del conjunto de manera especial en comunidades vulnerables (PNEU, 2004).

Participación: Es la acción y efecto de participar con carácter incluyente en un proceso (tomar o recibir parte de algo, compartir, noticiar). Es la capacidad ciudadana de involucrarse en las decisiones políticas, sociales y económicas de un país o región (PNEU, 2004).

Programa extensionista: Es un conjunto de acciones interrelacionadas y coordinadas, con el fin de alcanzar objetivos determinados dentro de los límites de una política dada, de un presupuesto y períodos de tiempo determinados. Pueden ser más o menos globales de acuerdo con el grado de generalidad que asumen. El programa constituye la forma organizativa más general para la extensión universitaria (PNEU, 2004).

Actividades extensionistas: Son aquellas que como parte de la planeación del trabajo sociocultural universitario ofrecen tratamiento a una situación y problema específico. Se identifican como tales el conjunto de acciones y tareas que tienen un carácter similar o están estrechamente relacionadas y que se orientan al cumplimiento de un objetivo específico o a parte del mismo. Las actividades pueden formar parte o no de un proyecto, su planificación puede ser independiente de estos, pero siempre estarán en el marco de la política que establece un programa (PNEU, 2004).

Acciones extensionistas: Están integradas por un conjunto de tareas que se identifican fuertemente entre sí por sus características similares y su orientación a propósitos muy específicos. Las acciones pueden existir en casos excepcionales independientes a las actividades y proyectos, pero siempre responden a la política y estrategias del programa (PNEU, 2004).

Tareas extensionistas: Representan la célula constitutiva del proceso extensionista y del trabajo sociocultural universitario, por ser la expresión más pequeña del mismo que mantiene sus características y relaciones fundamentales. Por eso la subdivisión de una tarea implicaría el desmembramiento del proceso y la pérdida de sus cualidades. Constituyen los eslabones primarios que permiten el alcance de los resultados esperados y la solución del problema (PNEU, 2004).

1.2. Análisis de soluciones similares

En este epígrafe se analizarán los sistemas de gestión de proyectos a nivel nacional, internacional y los que fueron producidos en la universidad, con el objetivo de identificar fortalezas y debilidades de los mismos.

AvaiBook Sports

Es un sistema que gestiona eventos deportivos de forma gratuita para que organizadores de eventos, empresas de cronometraje, clubes deportivos, circuitos de carreras, puedan mantener un control y gestión de las diferentes modalidades deportivas (AvaiBook Sports, 2014). Permite gestionar cualquier evento deportivo, inscribirse de forma *online* en el evento, promocionar y difundir el evento, gestionar los premios, exportar el listado de los participantes inscritos en el evento en formato Excel, utilizando los filtros que proporciona. El sistema es totalmente compatible con cualquier sistema de cronometraje. Permite utilizar redes sociales como Facebook y Twitter para la difusión de los eventos.

Por sus características queda descartado su uso como solución al problema, teniendo en cuenta que la solución solo se enmarca en la gestión de eventos deportivos y no integra todos los procesos que se gestionan en los proyectos extensionistas.

Sistema de gestión patrimonial (SIGPAC)

La Sección de Patrimonio Cultural es una unidad especializada, orientada a gestionar políticas públicas e implementar actividades que promuevan la salvaguarda del patrimonio cultural inmaterial colombiano, apoyando principalmente el fortalecimiento de las manifestaciones y expresiones vivas de las culturas locales (Báez, 2006). La herramienta permite gestionar los diferentes programas patrimoniales, promocionar y difundir los eventos que se desarrollan en el ámbito de la gestión del patrimonio cultural.

Este sistema queda descartado como solución al problema, teniendo en cuenta que solo se enmarca en la gestión patrimonial y no integra todos los procesos que se gestionan en los proyectos extensionistas.

1.2.1. Soluciones a nivel nacional

CubaLiteraria

La Editorial Electrónica CubaLiteraria es un portal del Instituto Cubano del Libro, que divulga el acontecer de la literatura cubana dentro y fuera de la Isla. Conforman su espacio informativo: ensayos, artículos noticiosos, columnas de autor, entrevistas, homenajes, convocatorias, entre otros perfiles.

Por sus características queda descartado su uso como solución al problema, teniendo en cuenta que se dedica solamente a la gestión de eventos culturales, la manera en que gestiona dichos eventos, no se ajusta a las formas de gestionar los proyectos extensionistas.

1.2.2. Soluciones a nivel de universidad

Gestión de la participación y los resultados obtenidos por los estudiantes en las actividades extracurriculares

Esta solución fue desarrollada en el 2008 para realizar la primera fase de un sistema que permitiera gestionar la información relacionada con la participación y resultados de los estudiantes en las distintas actividades extracurriculares. Consiste en una aplicación informática basada en una plataforma Web, que viabiliza la gestión de los procesos de planificación, organización y control que permita el manejo óptimo de los recursos humanos. Esta solución fue desarrollada debido a que no existía un software que permitiera aplicar las tecnologías de la informática y las comunicaciones para agilizar la generación y el flujo

de información de las actividades extracurriculares. La investigación solamente presenta las fases de análisis y diseño de este sistema (Laffita, 2008).

Debido a que en este proyecto solo se exponen las bases para la construcción de un sistema, sin llegar a implementarlo, se descarta como posible solución.

Desarrollo de una Aplicación Web para la Gestión de los procesos de Extensión y Residencia de la Facultad 15

Se desarrolla una aplicación informática basada en una plataforma Web para llevar a cabo la gestión de los procesos de planificación, organización y control en el área extensión y residencia de la Facultad 15 y que permita el manejo óptimo de los recursos humanos. Esta solución propone un sistema que facilita el manejo de toda la información relacionada con los procesos que se realizan en el área de extensión y residencia. Se centra en las funcionalidades que se requieren, debe permitir insertar, modificar, así como buscar y guardar la información (García, Carrillo, 2010).

Debido a que el sistema fue implementado hace varios años y que no genera la plantilla de los proyectos extensionistas en el nuevo formato se hace necesario desarrollar un nuevo sistema con funcionalidades que se ajusten a las necesidades actuales.

Sistema de información para el área de residencia y extensión universitaria de la Facultad 6

Esta investigación propone un sistema de información para el área de la residencia y extensión universitaria de la Facultad 6. Se basa en gestionar parte de la información relacionada con esta área en esa facultad. El sistema pretende llevar el control estadístico de la guardia estudiantil, cuartelería, TSU, paradas de beca, así como de las actividades de extensión universitaria como: juegos deportivos, festival de artistas aficionados, entre otras (Legrá, 2011).

Este sistema no responde a algunas de las necesidades actuales, entre las cuales se encuentra la gestión de las evidencias con el uso del nuevo formato de plantilla de inscripción, por lo que no puede tomarse como solución a la problemática propuesta en la presente investigación.

Solución para la gestión de información de la Extensión Universitaria en el área de Cultura Física en la Universidad de las Ciencias Informáticas

Esta solución permite la estandarización, organización, control de las acciones y sus resultados de forma pertinente. Realiza la consulta, intercambio y reporte de datos confiables para sustentar la toma de decisiones en el área de Cultura Física en la universidad (Gómez, Céspedes, 2013).

Este sistema no responde a las necesidades de gestión de la información de los proyectos extensionistas pues fue desarrollado para funcionar en un contexto limitado y no para tratar con información mucho más genérica como la que se maneja en las amplias variedades de proyectos extensionistas, y por tanto no genera la plantilla actualizada para la inscripción de estos proyectos.

Solución para la gestión de información de los procesos de Extensión Universitaria en el área de Extensión Cultural de la Universidad de las Ciencias Informáticas

En esta investigación se desarrolló un módulo en el Sistema Informático de Extensión Universitaria para la gestión de información del área de Extensión Cultural que permite la organización, control de las acciones y sus resultados de forma pertinente. También realiza consultas, permite obtener reportes de datos y contribuir a la gestión de la información. Practica el uso de tecnologías libres y la posibilidad de poder utilizarse en diferentes plataformas (Cabezas, 2013).

Esta solución no genera una plantilla de inscripción que contenga la información de un proyecto específico. No es factible como solución debido a que está específicamente orientado a la participación de los estudiantes en las actividades culturales, obligando a desarrollar otros sistemas para gestionar los demás procesos extensionistas (dígase actividades de los proyectos).

Sistema Web de la Dirección de Extensión Cultural de la Universidad de las Ciencias Informáticas

Consiste en el desarrollo de un sistema web para le gestión de la información de los procesos de la Dirección de Extensión Cultural de la UCI y que permita una comunicación interna y externa que propicie el diálogo, potencie la participación, posibilite la difusión y divulgación de la cultura y el quehacer universitario y social, contribuyendo así a la elevación del nivel cultural de la comunidad, su residencia y su entorno social. Se centra en el desarrollo de un sistema que brinda a los usuarios que accedan a este, la posibilidad de hacer consultas personalizadas a través de distintos filtros de búsqueda de la información de un artista aficionado o una actividad (Vázquez, 2014).

Este sistema no responde a las necesidades actuales, pues se concentra en una única área de extensión por tanto no sería de utilidad para gestionar todos los proyectos de esta índole de los cuales se necesite gestionar la información y evidencias.

Aplicación web para la gestión de los procesos extensionistas de la Facultad 1

En este trabajo se desarrolló una aplicación web para la gestión de los procesos extensionistas de la Facultad 1, con el objetivo de lograr un mejor control y seguimiento de los procesos de extensión universitaria. El sistema cuenta con varios módulos, a través de los cuales, los usuarios pueden acceder, y si su rol lo permite, modificar varias instancias de los datos (Leyva, López, 2015).

Atendiendo a su estructura y funcionalidades, este trabajo se refiere particularmente a gestionar las actividades extensionistas de manera general, por lo cual no es aplicable en el contexto específico de los proyectos extensionistas. Esta solución tampoco se ajusta a las necesidades actuales, ya que no genera la plantilla de inscripción con la información de los proyectos extensionistas.

1.2.3. Consideraciones sobre las soluciones similares

La Tabla 1 muestra el análisis de los sistemas homólogos, teniendo en cuenta como parámetros fundamentales: las plataformas para las que fueron desarrolladas, el uso de tecnologías libres, la gestión de evidencias de cada proyecto y si generan la plantilla con el nuevo formato (Ver Anexo 1).

Tabla 1: Resumen de los sistemas homólogos analizados. Elaboración propia

Autores	Plataformas	Uso de Tecnologías Libres	Gestión de Evidencias	Plantilla Actualizada
AvaiBook Sports	Linux, Windows, MacOS	Privativo	Sí	No
Sistema de gestión patrimonial (SIGPAC)	Linux, Windows, MacOS	Privativo	Sí	No
CubaLiteraria	Linux, Windows	Sí	No	No

Liumila Laffita Nicot, 2008	Propuesta de solución teórica			
ImilsyGarcía Brito y Diana Carrillo Feria, 2010	Linux, Windows, MacOS	Sí	No	No
LipsyMariem Legrá Legrá,2011	Linux,Windows	Sí	No	No
Kirenia Cabeza Matos y Javier Anias Santos, 2013	Linux, Windows, MacOS y Solaris	Sí	Sí	No
Celia Rosa Gómez Hernándezy Leandro Lázaro Céspedes Lara, 2013	Linux, Windows, MacOS	Sí	Sí	No
Adonis Vázquez Campos, 2014	Linux, Windows, MacOS	Sí	Sí	No
Lidia del Carmen Leyva Feijoó y Alejandro López Rodríguez, 2015	Linux, Windows, MacOS y Solaris	Sí	Sí	No

El estudio de los sistemas similares permitió comprobar que estos no se ajustan a las formas organizativas actuales en la gestión de proyectos extensionistas, sin embargo, poseen elementos comunes que pueden ser tomados en cuenta en el sistema propuesto, tales como:

- La gestión de documentación.
- El control de acceso basado en roles.
- La gestión de proyectos.
- La gestión de los recursos humanos de un proyecto.
- La gestión de las distintas fases y actividades de un proyecto.
- Una visualización de informes o reportes sobre los proyectos registrados.
- Una visualización de las actividades de cada proyecto.

1.3. Metodología de desarrollo de software

Las metodologías de desarrollo de software surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental para desarrollar un software. Para ello, se hace necesario llevar a cabo un estricto control de los flujos de trabajo, de no ser así, es muy posible que una vez finalizado el proyecto aún existan inconformidades o salidas con resultados inesperados. Para evitar este tipo de problemas, es de vital importancia escoger la metodología de desarrollo de software que más se ajuste a las condiciones de trabajo de los desarrolladores y a las necesidades del cliente.

Las Metodologías de Desarrollo de Software se clasifican en dos grandes grupos (Figueroa, Roberth, Solís, Cabrera, 2010):

Metodologías tradicionales (pesadas)

Las metodologías tradicionales están orientadas al control de los procesos, estableciendo rigurosamente las actividades a desarrollar, herramientas a utilizar y notaciones que se usarán (Figueroa, Roberth, Solís, Cabrera, 2010). Este tipo de metodología, puede manifestarse en diversas modalidades, tales como:

Proceso Racional Unificado (RUP [1]): Tiene como objetivo ordenar y estructurar el desarrollo de software, en la cual se tienen un conjunto de actividades necesarias para transformar los requisitos del usuario en un sistema. Es un proceso basado en los modelos en Cascada y por Componentes, el cual presenta las siguientes características: es dirigido por los casos de uso, es centrado en la arquitectura, así como iterativo e incremental, lo cual es fundamental para el proceso de desarrollo de software (Figueroa, Roberth, Solís, Cabrera, 2010).

Marco de Soluciones de Microsoft (MSF [2]): Esta es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. Se centra en los modelos de proceso y de equipo, deja en un segundo plano las elecciones tecnológicas. Compuesta de varios modelos encargados de planificar las diferentes partes implicadas en el desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el

[1] *Rational Unified Process*

[2] *Microsoft Solution Framework*

modelo de Aplicación. La Metodología MSF se adapta a proyectos de cualquier dimensión y de cualquier tecnología (Figueroa, Roberth, Solís, Cabrera, 2010).

Metodologías ligeras (ágiles)

Este tipo de metodologías van orientadas a la interactuación con el cliente y el desarrollo incremental del software, mostrando versiones parcialmente funcionales del software al cliente en intervalos cortos de tiempo, para que pueda evaluar y sugerir cambios en el producto según se va desarrollando (Callejas, Baquero, 2007). Entre los ejemplos más representativos de este grupo se encuentran: la Programación Extrema, SCRUM y el Proceso Unificado Ágil (AUP).

Proceso Unificado Ágil (AUP[3]): Es una versión simplificada del RUP. Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando conceptos que aún se mantienen válidos en RUP y técnicas ágiles. Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva: inicio, elaboración, construcción y transición. En el proceso AUP las disciplinas de: Modelado de Negocio, Requisitos y Análisis y Diseño, se agrupan en una sola, mientras que las restantes (Implementación, Pruebas, Despliegue, Gestión de Configuración, Gestión y Entorno) coinciden con las de RUP (Callejas, Baquero, 2007).

SCRUM: Es una metodología ágil útil para empresas de desarrollo de software orientadas a varios clientes. Consiste en un modelo de asignación de tareas diarias basado en reuniones rápidas y control de la evolución de los procesos. Es muy bueno para llevar un seguimiento de las tareas que se están llevando a cabo y saber en qué puntos se ha atascado el equipo. Además, la profundidad de las tareas que se asignan en SCRUM tiende a ser incremental, y esto coincide exactamente con el devenir normal de un proceso de desarrollo (Callejas, Baquero, 2007).

Metodología de desarrollo para la Actividad productiva de la UCI (AUP-UCI): Como parte de un proceso de estandarización llevado a cabo en los centros productivos de la UCI, se aprobó este tipo de metodología. Al no existir una metodología de software universal, se decide hacer una variación de la metodología y las disciplinas de AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI, quedando de la

[3] *Agile Unified Process*

siguiente forma: gestión de la configuración, la planeación y el control del proyecto (Sánchez, 2015).

A partir de que el modelado de negocio propone tres variantes a utilizar en los proyectos (Casos de Uso del Negocio (CUN), Descripción de Procesos del Negocio (DPN) o Modelo Conceptual (MC)) y existen tres formas de encapsular los requisitos (Casos de Uso del Sistema (CUS), Historias de Usuario (HU) o Descripción de Requisitos por Proceso (DRP)), surgen cuatro escenarios en la metodología AUP en su variante UCI para modelar el sistema en los proyectos, manteniendo en dos de ellos el MC, quedando de la siguiente forma (Santana, 2015):

- Escenario No 1: proyectos que modelen el negocio con CUN solo pueden modelar el sistema con CUS.
- Escenario No 2: proyectos que modelen el negocio con MC solo pueden modelar el sistema con CUS.
- Escenario No 3: proyectos que modelen el negocio con DPN solo pueden modelar el sistema con DRP.
- Escenario No 4: proyectos que no modelen negocio solo pueden modelar el sistema con HU.

Programación Extrema o *Extreme Programming* (XP): 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 estableciendo una fuerte retroalimentación con el cliente. Es adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Esta metodología, evidencia principios tales como el desarrollo incremental, la participación activa del cliente, el interés en las personas y no en los procesos como elemento principal, y aceptar el cambio y la simplicidad. Tomó su nombre por las prácticas reconocidas en el desarrollo de software y por la participación del cliente en niveles extremos. XP incluye las fases de: planeación, diseño, codificación y pruebas.

Fases del ciclo XP (Pressman, 2010)

Planeación o Planificación del proyecto: Entre las acciones que se ejecutan en esta fase se encuentra definir las historias de usuario con el cliente y crear el plan de iteraciones

donde se indica en qué iteración serán implementadas las historias de usuarios previamente definidas.

Diseño: Sugiere utilizar diseños sencillos y simples pues de esta manera costará menos tiempo y esfuerzo desarrollarlo posteriormente.

Codificación: Se definen los estándares de codificación que se utilizarán para facilitar la comprensión del código.

Pruebas: El funcionamiento del sistema se lleva a cabo a través de las historias de usuario definidas en la primera fase, el cual sugiere que se realicen pruebas funcionales que sirven para evaluar las distintas tareas realizadas.

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, puesto que se perderá calidad en el software o no se cumplirán los plazos. 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.

Justificación de selección de la metodología

Luego de haberse estudiado en profundidad las características y particularidades, tanto de los enfoques ágiles como de los tradicionales, se decidió que para el desarrollo de la presente solución se adoptará una metodología ágil, basándose en la necesidad de realizar entregas parciales de la solución en el menor tiempo posible, dándole prioridad a las funcionalidades más esenciales y aprovechando la flexibilidad de las tareas asignadas.

Como metodología de desarrollo, fue seleccionada la Programación Extrema (XP). Para ello, se tuvo en cuenta que:

- El equipo de desarrollo es pequeño (2 personas), el cual se encuentra en constante comunicación con el cliente.
- Los requisitos de software pueden cambiar durante el proceso de implementación.
- El software puede ser implementado en un tiempo relativamente corto.

1.4. Herramientas y tecnologías

Para el desarrollo exitoso de un producto de software es fundamental la correcta selección de las herramientas a emplear, lo que influye directamente en la calidad del producto final y el nivel de esfuerzo del equipo de desarrollo para obtenerlo.

A continuación, se describe cada una de las tecnologías utilizadas en la solución propuesta.

1.4.1. Herramientas de Ingeniería del Software Asistidas por Computadoras o CASE Tools[4]

Las herramientas CASE son herramientas que sirven para auxiliar a los desarrolladores. Estas han surgido para dar solución a varios problemas inherentes al diseño del software, principalmente para la mejora de la calidad del desarrollo de sistemas de mediano y gran tamaño (Meza, 2011).

Visual Paradigm

Es una herramienta CASE profesional que soporta el ciclo de vida completo del desarrollo de software, análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El lenguaje de modelado UML ayuda a una rápida construcción de aplicaciones de calidad. Permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación. La herramienta CASE también proporciona abundantes tutoriales, demostraciones interactivas y proyectos UML (Visual Paradigm for UML, 2014).

Para el desarrollo de la solución propuesta se utilizó la herramienta Visual Paradigm en su versión 8.0.

Entre las ventajas que brinda Visual Paradigm se destacan:

- La navegación intuitiva entre el código y el modelo.
- Un poderoso generador de documentación y reportes UML PDF/HTML/MS Word.
- Soporte completo de notaciones UML.
- Diagramas de diseño automático sofisticado.
- Análisis de texto y soporte de tarjeta CRC.
- Proporciona soportes a varios lenguajes en generación de código e ingeniería inversa a través de plataformas Java, C++, entre otros lenguajes de programación.

Lenguaje Unificado de Modelado (UML) 2.0

UML es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. El mismo permite realizar una

[4] *Computer Aided Software Engineering*

descripción altamente detallada del sistema, de procesos del negocio, de componentes de software reutilizables, entre otros (Alhir, 2003).

1.4.2. Herramienta de modelado de interfaces

Axure RP

Es una aplicación ideal para crear prototipos y especificaciones muy precisas para páginas web. Se trata de una herramienta especializada en la tarea, así que cuenta con todo lo que se puede necesitar para crear los prototipos de forma más eficiente. Axure RP permite componer la página web visualmente, añadiendo, quitando y modificando los elementos con suma facilidad, demostrando su grado de especialización en las anotaciones. En este punto, permite especificar el estado de cada elemento, el beneficio esperado, el riesgo, la estabilidad, a quién va dirigido y a quién se le asignará la tarea. Otra característica a destacar del Axure RP es que permite un diseño colaborativo. Las características analizadas fueron tomadas en cuenta a la hora de su elección además de ser el Axure RP la herramienta utilizada por el Grupo de arquitectura de Información de la universidad para esta función (Axure, 2018).

1.4.3. Entorno de desarrollo integrado (IDE[5])

Un entorno de desarrollo integrado o entorno de desarrollo interactivo, es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software (Hernández, 2010).

Microsoft Visual Studio Code

Es un entorno de desarrollo integrado para Windows, Linux y MacOS. Es compatible con múltiples lenguajes de programación, tales como C++, C#, Visual Basic .NET, F#, Java, Python, Ruby y PHP. Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno compatible con la plataforma .NET (a partir de la versión .NET 2002). Así, se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, entre otros (Visual Studio, 2019).

[5] *Integrated development environment*

1.4.4. Lenguajes de programación

Un lenguaje de programación es un idioma artificial diseñado para controlar el comportamiento de una máquina como las computadoras, para expresar algoritmos con precisión (Arias, 2011). Pueden usarse con diferentes fines: para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Están constituidos por un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Su principal ventaja es que pueden adaptarse fácilmente para ejecutarse en diferentes tipos de equipos. En el presente acápite, se hace referencia a los lenguajes de programación utilizados en la investigación.

Lenguajes del lado cliente

Lenguaje de Marcación de Hipertexto (HTML[6]) v5.0

El Lenguaje de Marcado de Hipertexto es el lenguaje orientado al desarrollo web. Está compuesto por etiquetas que marcan el principio y el final de cada elemento del documento. Permite representar el contenido en forma de texto, así como complementar el texto con objetos, como el caso de las imágenes. Los documentos HTML deben tener la extensión HTML o HTM14, para que puedan ser visualizados en los navegadores (Hernández, 2010).

Lenguajes del lado del servidor

PHP v.5.4.3

PHP es un lenguaje de script incrustado dentro del HTML. Es un lenguaje de programación para la creación de páginas web dinámicas. Permite la creación de aplicaciones con interfaz gráfica, conexiones a servidores de base de datos (Oracle, MySQL) y puede ser ejecutado en sistemas como Unix, Windows y Linux. PHP ofrece un extenso conjunto de funciones para la explotación de bases de datos lo cual facilita determinadas acciones sin tener que generar programas realizados en un lenguaje distinto al HTML (Cervero, 2011).

[6] *Hypertext Mark-Up Language HTML*

1.4.5. Framework[7] de desarrollo

Un marco de trabajo o *framework* es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar. Permite simplificar el desarrollo de un sistema mediante la utilización de distintos tipos de lenguajes, el uso de componentes, librerías y estructuras de acuerdo a los diferentes patrones existentes. Aporta agilidad y facilidad a los programadores, pues encapsula operaciones complejas en instrucciones sencillas y posee una organización bien estructurada del proyecto informático en desarrollo (Hernández, 2010)**.**

Bootstrap 4

Es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales. Bootstrap proporciona un conjunto de hojas de estilo que proveen definiciones básicas de estilo para todos los componentes de HTML. Esto otorga una uniformidad al navegador y da una apariencia moderna para el formateo de los elementos de texto, tablas y formularios (Hernández, 2010).

Symfony 3.4

Es un marco estable para desarrollar aplicaciones, está publicado bajo una licencia de software libre y es fácil de instalar y configurar en la mayoría de las plataformas. Es sencillo de usar y al mismo tiempo es flexible. Su arquitectura interna está completamente desacoplada, lo que permite adaptarse fácilmente a los diferentes proyectos. Posee la ventaja de ser multiplataforma y compatible con la mayoría de gestores de bases de datos como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Hace uso del patrón arquitectónico MVC y sigue la mayoría de las mejores prácticas y patrones de diseño para la web.

[7] entorno de trabajo o marco de trabajo

1.4.6. Plataforma de desarrollo Web

Una plataforma web tiene cuatro componentes básicos: un sistema operativo, un servidor web, una base de datos y un lenguaje de programación. Buena parte de las empresas comerciales presentan los cuatro componentes bien sea empaquetados, o de fácil incorporación e integración entre ellos, ventaja que no se tiene en el mundo abierto ya que cada uno produce por su lado (Rodríguez, 2013).

Apache 2.4

Apache es el servidor web hecho por su excelencia, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. La licencia Apache es descendiente de la licencia BSD, la cual permite modificar con el código fuente. Actualmente más del 60 por ciento de los administradores de toda la Web utilizan Apache. Se trata una de las plataformas de servidores Web de código abierto más poderosas del mundo (Rodríguez, 2013).

1.4.7. Sistema Gestor de Bases de Datos

Un sistema gestor de base de datos (SGBD), es un tipo de software específico dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan; lo que es equivalente a una agrupación de programas que sirven para definirla, construirla y manipularla, permitiendo almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Brindan facilidades eficientes y un grupo de funciones con el objetivo de garantizar la confidencialidad, la calidad, la seguridad y la integridad de los datos que contienen, así como un acceso fácil y eficiente a los mismos (Chacón, 2011).

MySQL v.5.2.24

Es un sistema gestor de bases de datos relacionales rápido y flexible. Es idóneo para la creación de bases de datos con acceso desde páginas web dinámicas, así como para la creación de cualquier otra solución que implique el almacenamiento de datos, posibilitando realizar múltiples y rápidas consultas. Está desarrollado en C# y C++, facilitando su integración en otras aplicaciones desarrolladas también en esos lenguajes. Su gran rapidez y sencillez se debe, en gran medida, a la infinidad de librerías y herramientas que permiten su uso en distintos lenguajes de programación (Chacón, 2011).

Conclusiones del Capítulo 1

- El estudio de los principales conceptos asociados al problema permitió adquirir y fomentar los conocimientos necesarios para entender los procesos de gestión de proyectos extensionistas.
- En el análisis de los sistemas homólogos se pudo determinar que algunos satisfacen una parte de las necesidades existentes, pero no existe una herramienta capaz de generar la plantilla de inscripción actualizada para los proyectos extensionista, por lo que resulta importante la creación de un nuevo sistema que satisfaga esta necesidad.
- La metodología de desarrollo escogida se adapta al entorno de trabajo escogido por los desarrolladores, justificada por las características del equipo de desarrollo y las prioridades del negocio.
- El estudio de las características de las herramientas y tecnologías a utilizar permitió justificar su selección para el desarrollo de la presente investigación.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA SOLUCIÓN PROPUESTA

Introducción

En el siguiente capítulo se describen las características de la propuesta de solución. En este sentido, se realizará una descripción general del posible resultado, luego se procede a realizar la Ingeniería de Requisitos abarcando los requisitos funcionales y no funcionales. Se describirá de igual forma la disciplina de análisis y diseño a través de los patrones de diseño, la historias de usuario y las tarjetas CRC[8], haciendo uso de la metodología XP.

2.1. Modelado del Negocio

El propósito principal de esta fase en el proceso de desarrollo de software es realizar una exploración del dominio del problema, especificando los conceptos y las relaciones que existen entre estos. A continuación, se detalla una breve descripción del negocio y el modelo de dominio generado para mayor entendimiento del mismo.

2.1.1. Breve descripción del negocio

El proceso de inscripción de proyectos extensionistas, consta de una serie de procedimientos:

1. El profesor, estudiante, departamento u otra disciplina de la Universidad identifica una problemática que deba ser resuelta mediante la realización de un proyecto extensionista y presenta las ideas básicas al Vicedecano de Extensión Universitaria.
2. El Vicedecano de Extensión Universitario, pide criterios del proyecto extensionista presentado al órgano asesor para estos temas que reside en el Departamento de Gestión Extensionista de la Dirección de Extensión Universitaria.
3. El órgano asesor para proyectos extensionistas entrega por escrito los criterios referentes al proyecto extensionista presentado al Vicedecano de Extensión Universitario o responsable del área en cuestión.
4. El Vicedecano de Extensión Universitario o responsable del área en cuestión solicita presentar en el Consejo de Dirección de su área el proyecto extensionista.
5. El Consejo de Dirección del área a la que se subordinará el proyecto extensionista aprobará o no el proyecto, escuchando a todas las partes que este órgano colectivo de dirección entienda necesarias.

[8] Clase, Responsabilidad y Colaboración

6. Si el proyecto extensionista es aprobado por el Consejo de Dirección del área será luego obligación de este órgano colectivo de dirección que el proyecto rinda cuenta su marcha al menos con carácter semestral.

Para centralizar, disponer y controlar la información referente a estos proyectos, se decidió desarrollar un sistema informático que permitirá a los usuarios interactuar con la información de los proyectos extensionistas, además de potenciar la participación y el intercambio entre cada uno de los interesados promoviendo el trabajo sociocultural de la universidad. Para lograr este objetivo, se ofrecen una serie de utilidades, siendo los coordinadores los encargados de realizar cualquier tipo de modificación de acuerdo a las necesidades.

2.1.2. Modelo de Dominio

Un modelo de dominio es la representación visual de conceptos u objetos relacionados entre sí. Es el mecanismo fundamental para comprender el problema y establecer conceptos afines. Se utiliza como alternativa para la comprensión por parte del cliente de la estructura del negocio y se describe mediante diagramas de clases (Márquez, 2009).

Independientemente de que la metodología *XP* no genere un modelo de dominio como artefacto de esta, se vio la necesidad de realizarlo, en aras de un mejor entendimiento del negocio. Vale destacar que un modelo de dominio se utiliza para representar clases conceptuales del mundo real, y no componentes de software.

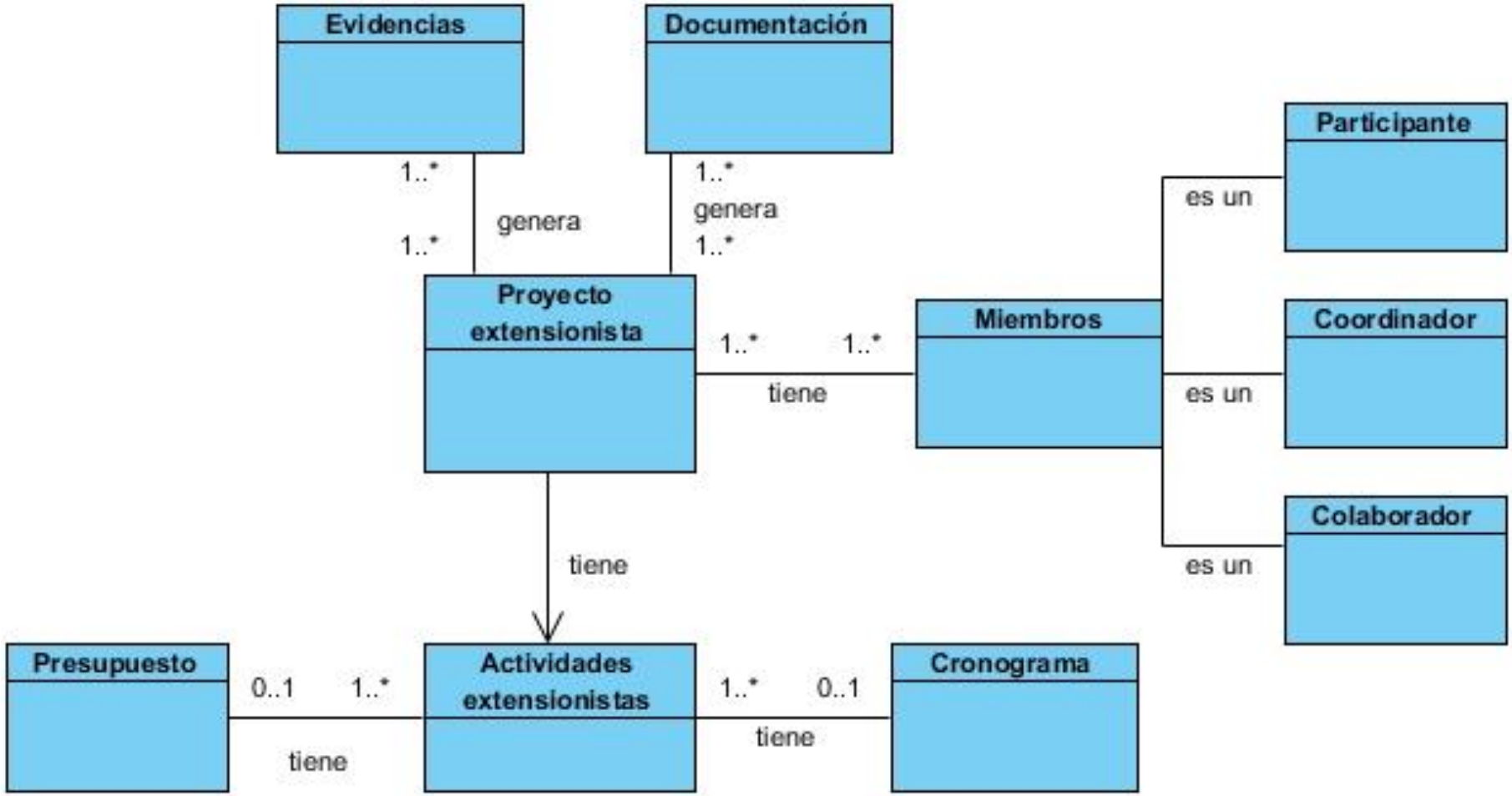

Figura 1: Modelo de dominio. Elaboración propia

2.1.3. Conceptos del dominio

Proyecto extensionista: Eje articulador de la gestión de la extensión universitaria y elemento clave para dinamizarla y promover las transformaciones que se requieren en este proceso. Constituye una unidad más operativa dentro del proceso de planeación del trabajo sociocultural universitario, pues ofrece tratamiento a situaciones y problemas específicos. Es una unidad mínima de asignación de recursos, que a través de un conjunto concreto de actividades, acciones y tareas pretende modificar o transformar una parcela de la realidad sociocultural disminuyendo o eliminando un déficit o solucionando un problema (PNEU, 2004).

Cronograma del proyecto: incluye las fechas planificadas de inicio y finalización del proyecto y las actividades, acciones y tareas extensionistas que se requieren para alcanzar los resultados previstos en el plazo acordado.

Documentación del proyecto: constituye la representación física o electrónica de la información que se genera como resultado de la gestión de los procesos que tienen lugar durante el ciclo de vida del proyecto.

Evidencia(s) de proyecto: se refiere a la documentación, archivos de audio y video, imágenes, productos y otros materiales que se generan durante el ciclo de vida del proyecto y se conservan como constancia de la realización del mismo.

Presupuesto: se refiere a la previsión del dinero o recursos necesarios para un determinado proyecto o actividad.

Miembro(s): persona asociada a un proyecto extensionista, que puede ser asignada como seguidor o responsable a una actividad, acción o tarea en el sistema.

Coordinador(es): se refiere a la(s) persona(s) encargada(s) de gestionar el trabajo del proyecto y orientarlo hacia el cumplimiento de sus. Aplica conocimientos administrativos y gerenciales, habilidades, herramientas y técnicas para un alto rango de tareas con el fin de obtener el resultado deseado para el proyecto, en el momento oportuno.

Colaborador(es): se refiere a los miembros que participan directamente en la gestión, aportando conocimientos, habilidades, técnicas o recursos.

Participante(s): se refiere a los miembros que participan directa o indirectamente en una actividad, pero sin tener cargos en la gestión.

2.2. Breve descripción del sistema propuesto

El sistema que se propone tiene como principal objetivo gestionar los procesos de planificación y control, para ello los usuarios del sistema dispondrán de funcionalidades que les permitirán:

- Gestionar los proyectos extensionistas, respetando las formas organizativas del proceso extensionista. Los usuarios autorizados podrán acceder, obtener y modificar información de los proyectos extensionistas.
- Gestionar los miembros de un proyecto extensionista, pudiendo asignarles uno o varios roles.
- Generar y visualizar el listado de las actividades de un proyecto.
- Generar reportes sobre el estado de las actividades, proyectos y recursos necesarios.
- Generar la plantilla de inscripción de proyectos.
- Gestionar las evidencias de cada proyecto.

2.3. Disciplina de requisitos.

El propósito de esta fase es la definición de requisitos que especifican las condiciones o capacidades que el sistema debe cumplir y las restricciones bajo las cuales debe operar, logrando un entendimiento entre el equipo de desarrollo y el especialista funcional, y especificando las necesidades reales de forma que satisfaga sus expectativas (UCID: SOFTWARE-DEFENSA, 2012).

Durante la fase de obtención de los requisitos de software, se realizó un grupo de reuniones y entrevistas no formales y no estructuradas entre el cliente y el equipo de desarrollo, en las cuales se formularon una serie de preguntas (Ver Anexo 2), en busca de un mayor entendimiento de la problemática existente, y a partir de ahí poder identificar los requisitos del software. En el presente acápite se describen los requisitos funcionales y no funcionales obtenidos para el desarrollo del sistema.

2.3.1. Requisitos funcionales del sistema (RF)

La obtención de requisitos es el proceso mediante el cual los interesados en un sistema de software descubren, revelan y exponen sus peticiones. Estos definen las condiciones o capacidades que el sistema será capaz de realizar y describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Estos requisitos, al tiempo que avanza el proyecto de software, se convierten en los algoritmos, la lógica y gran parte del código del sistema (UCID: SOFTWARE-DEFENSA, 2012).

La Tabla 2 muestra los requisitos funcionales del sistema. Se decidió agrupar las tareas en forma de paquetes con el objetivo de organizarlas según su prioridad y facilitar el trabajo al equipo de desarrollo, siguiendo el esquema de trabajo XP.

Tabla 2: Requisitos funcionales del sistema. Elaboración propia

Identificador	Nombre del requisito	Prioridad	Complejidad
Paquete “Plantilla de proyecto”			
RF1	Generar plantilla	Alta	Media
Paquete “Evidencias”			
RF2	Insertar evidencia	Alta	Alta
RF3	Eliminar evidencia	Alta	Alta
Paquete “Proyecto”			
RF4	Crear nuevo proyecto	Alta	Media
RF5	Actualizar proyecto	Alta	Media
RF6	Eliminar proyecto	Alta	Media
RF7	Mostrar proyecto	Alta	Media
RF8	Cerrar proyecto	Alta	Media
RF9	Generar reporte de actividades del proyecto	Alta	Media
Paquete “Actividad”			
RF10	Crear nueva actividad	Media	Media
RF11	Actualizar actividad	Media	Media
RF12	Eliminar actividad	Media	Media
RF13	Listar actividades	Media	Media

RF14	Generar reporte de recursos	Media	Media
Paquete “Roles”			
RF15	Nuevo rol	Baja	Media
RF16	Eliminar rol	Baja	Media
Paquete “Miembros”			
RF17	Insertar miembro	Baja	Baja
RF18	Eliminar miembro	Baja	Baja
RF19	Otorgar permisos	Baja	Media
RF20	Denegar permisos	Baja	Media

2.3.2. Requisitos no funcionales del sistema (RNF)

Los requerimientos no funcionales de un sistema son propiedades, cualidades o restricciones que el producto debe cumplir (sin referirse directamente a las funciones específicas), tales como restricciones de tiempo, estándares de desarrollo, especificaciones de diseño, entre otros (Pressman, 2010).

A continuación, se describen las características no funcionales que debe tener el sistema a implementar:

RNF1: Apariencia o interfaz externa. El sistema deberá poseer una interfaz gráfica uniforme, ajustándose al diseño del portal Dragones.

RNF2: Seguridad. La seguridad del sistema está basada en niveles de acceso, establecida por roles que serán asignados a los usuarios que interactúan con el sistema. Esto garantizará que la información almacenada solo sea modificada o visualizada solo por los usuarios autorizados.

RNF3: Usabilidad. El sistema requiere conocimientos básicos, en este caso será utilizado por los coordinadores de los proyectos, quienes manejarán los datos de forma administrativa.

Se utilizará el idioma español para los formularios, tablas, reportes y textos de la interfaz.

La navegabilidad no debe ser compleja, para garantizar que las funcionalidades sean rápidamente accesibles por el usuario.

El sistema debe ser escalable. De esta forma podrán agregarse nuevas funcionalidades, sin ser afectadas las que ya están funcionando.

RNF4: Disponibilidad. El sistema debe estar disponible las 24 horas del día, sin incluir los días dedicados al mantenimiento.

RNF5: Portabilidad. El sistema debe ajustarse a distintos tipos de navegadores (Firefox, Chrome u otros).

2.3.3. Validación de Requisitos de Software

La validación de requisitos examina las especificaciones para asegurar que todas las especificidades del sistema han sido establecidas sin ambigüedad, sin inconsistencias, sin omisiones, que los errores detectados hayan sido corregidos, y que el resultado del trabajo se ajusta a los estándares establecidos para el proceso, el proyecto y el producto (Pressman, 2010). El proceso de validación se considera de vital importancia dentro de la disciplina de requisitos, ya que:

- Asegura que los ingenieros de software entiendan el marco de trabajo y los objetivos.
- Verifica que los requerimientos sean comprensibles, constantes y finitos.
- Reduce los costos, al detectar errores antes del desarrollo o después que el sistema esté en uso.

Para validar los requerimientos, es preciso utilizar una de las técnicas empleadas para esta tarea. En la presente investigación se adoptó la técnica de realización de los prototipos, la cual permitió representar visualmente las condiciones solicitadas por el cliente. En el 0 se hace referencia a las Historias de Usuario, donde se detallan los prototipos obtenidos en la propuesta de solución.

2.4. Historias de Usuario (HU)

Las HU representan una breve descripción del comportamiento del sistema, empleando una terminología sin lenguaje técnico. Se realizan para cada característica principal del sistema (requisitos funcionales). Estas sustituyen a los documentos de especificación funcional, y a los “casos de uso”. Estas “historias” son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. La diferencia más importante entre estas historias y los tradicionales documentos de especificación funcional se encuentra en el nivel de detalle requerido (Joskowicz, 2008).

La Tabla 3: Historia de Usuario "Crear nueva plantilla" muestra la HU correspondiente a la funcionalidad “Crear nueva plantilla”. El resto de las HU se encuentran en el Anexo 3 del presente documento.

Tabla 3: Historia de Usuario "Crear nueva plantilla". Elaboración propia

Historia de Usuario	
Identificador: RF1	**Nombre:** Crear nueva plantilla
Usuario: Coordinador	**Iteración asignada:** 1
Prioridad en el negocio: Alta	**Puntos estimados:**8
Riesgo en desarrollo: Alto	**Puntos Reales:**8
Descripción: Genera una plantilla de proyecto con todos sus datos.	
Prototipo de Interfaz	

Descripción de los campos de una HU

Identificador: Un identificador es un conjunto de caracteres alfanuméricos de cualquier longitud que sirve para enumerar o marcar las entidades. Los identificadores pueden ser combinaciones de letras y números.

Nombre: El nombre es la designación o denominación verbal que se le da a un concepto tangible o intangible, concreto o abstracto, para distinguirlo de otros.

Usuario: Un usuario es “quien usa algo”, con el objetivo de consumir un servicio u obtener algún beneficio.

Iteración asignada: El software desarrollado en una unidad de tiempo es llamado una iteración. Cada iteración del ciclo de vida del proyecto retoma cada una de las fases de la metodología.

Puntos estimados y reales: Es una representación de la cantidad de esfuerzo que supone desarrollar una HU. Se definió una escala de 1 a 10, considerándose que:

- De 1 a 4, la HU requiere un bajo esfuerzo de desarrollo.
- De 5 a 7, la HU requiere un mediano esfuerzo de desarrollo.
- De 8 a 10, la HU requiere un alto esfuerzo de desarrollo.

Riesgo en desarrollo: Riesgo es una medida de la magnitud de los daños frente a una situación que pueda afectar los flujos de trabajo durante el desarrollo de software.

Para establecer las complejidades de cada uno de los requisitos funcionales se definió una escala de riesgos:

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

Descripción: Explica cómo funciona el requisito descrito en la HU.

Observaciones: Alguna especificación o explicación adicional.

Imagen de la Interfaz: Es una muestra de la interfaz que representa la HU.

2.5. Tareas de la Ingeniería o Task Card

Las Tareas de la Ingeniería (TI) dividen en escenarios las HU previamente confeccionadas para el desarrollo del sistema, donde a cada HU le puede corresponder más de una TI. Describen con más profundidad las características del sistema y se consideran la entrada de trabajo para el programador (Calero, 2013).

La siguiente tabla muestra una de las TI correspondiente a la HU “Crear nueva plantilla”. Se realizaron un total de 21 TI, el resto se encuentra en el Anexo 4 del presente documento.

Tabla 4: Tarea de Ingeniería:

Tarea de Ingeniería	
Número de la Tarea: 1	**HU:** RF1
Nombre de la Tarea: Crear plantilla de proyecto extensionista	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.8 semanas
Fecha inicio: 11/2/2019	**Fecha Fin: 15/2/2019**
Programador responsable	
Descripción	

2.6. Tarjetas CRC

Las tarjetas CRC (Clase, Responsabilidad, Colaboración) se crean en la segunda fase de la metodología XP y representan las clases a la que pertenece un objeto determinado. Estas tarjetas se dividen en tres secciones que contienen la información del nombre de la clase, sus responsabilidades y sus colaboradores. En ellas se expresa el diseño del sistema (Pressman, 2010):

- Clase: representa una entidad, como por ejemplo una persona o evento.
- Responsabilidades: representa las actividades que realizan, atributos y métodos.
- Colaboradores: representa las clases con las que trabaja en conjunto para llevar a cabo sus responsabilidades.

Los pasos a seguir para llenar las tarjetas son los siguientes:

1. Encontrar clases.
2. Encontrar responsabilidades.
3. Definir colaboradores.
4. Disponer las tarjetas.

La Tabla 5, muestra la tarjeta CRC correspondiente a la clase “Plantilla”.

Tabla 5: Tarjeta CRC de la clase "Proyecto". Elaboración propia

Clase: Proyecto	
Descripción: Contiene una tabla con los campos de una plantilla de proyecto.	
Responsabilidad(es): Adicionar, actualizar, modificar y eliminar datos de los proyectos.	**Colaborador(es)** Actividad, Evaluación.

2.7. Plan de iteraciones

Un Plan de Iteraciones (PI) es una planificación donde los desarrolladores y clientes establecen los tiempos de implementación ideales de las HU. Las iteraciones contarán con una breve descripción de lo que se implementará en ellas, el orden en que se van a implementar las HU y la duración total de cada una.

La Tabla 6 muestra el PI elaborado para la confección del sistema para la gestión de proyectos extensionistas.

Tabla 6: Plan de Iteraciones. Elaboración propia

Iteración	Descripción de la iteración	Orden de la HU a implementar	Duración total
1.	En esta iteración se implementarán las HU con prioridad Alta.	HU1-HU9	8 semanas
2.	En esta iteración se implementarán las HU con prioridad Media.	HU10-HU14	4 semanas
3.	En esta iteración se implementarán las HU con prioridad Baja	HU15-HU21	4 semanas

2.8. Plan de entregas

En esta sección se documenta lo acordado en las reuniones con el cliente, midiendo la prioridad de cada HU y atendiendo a las necesidades más inmediatas. El plan de entregas reflejado en la Tabla 7**.**

Tabla 7: Plan de entregas. Elaboración propia

Plan de entregas del Sistema para la gestión de proyectos extensionistas			
Iteración	**Entregable**	**Fecha inicio**	**Fecha Fin**
Iteración 1	Sistema para la gestión de proyectos extensionistas	8/2/2019	8/4/2019
Iteración 2	Sistema para la gestión de proyectos extensionistas	9/4/2019	16/5/2017
Iteración 3	Sistema para la gestión de proyectos extensionistas	17/5/2019	27/6/2017

2.9. Arquitectura y patrones

Para lograr un mayor entendimiento de la estructura del sistema, en el presente acápite se describe el diseño arquitectónico sobre el cual está enmarcado el producto final de la investigación. Para ello, se hace referencia a la forma en la que la aplicación queda estructurada.

2.9.1. Arquitectura de software

La arquitectura de software se refiere a la estructura general del software y las formas en que esta proporciona una integridad al sistema. En su forma más simple, la arquitectura es la estructura u organización de los componentes del programa (módulos), la manera en que estos componentes interactúan, y la estructura de datos que utilizan los componentes (Pressman, 2010).

Debido a que se escogió a Symfony como marco de trabajo, se decidió utilizar la arquitectura definida por este framework, dígase Modelo-Vista-Controlador (MVC).

En líneas generales, MVC es una propuesta de diseño de software utilizada para implementar sistemas donde se requiere el uso de interfaces de usuario. Surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado, donde se potencie la facilidad de mantenimiento, reutilización del código y la separación de conceptos (Alvarez, 2014).Esta arquitectura está conformada por tres niveles:

- El Modelo que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.
- La Vista, o interfaz de usuario, que representa visualmente la información que se envía al cliente y los mecanismos interacción con éste.
- El Controlador, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.

El diagrama de componentes presentado en la Figura 3, explica cómo se evidencia el uso de la arquitectura MVC en el desarrollo de la presente solución.

2.9.2. Patrones de diseño

Los patrones constituyen una guía para el diseño del software. Su objetivo es la solución de problemas que ocurren repetidamente dentro de un contexto muy bien definido. Además,

deben ser reusables, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. Son de gran utilidad para describir las mejores prácticas, buenos diseños, y encapsulan la experiencia permitiendo su reutilización. Definen la estructura de un sistema de software, los cuales a su vez se componen de subsistemas con sus responsabilidades, también tienen una serie de directivas para organizar los componentes del mismo sistema, con el objetivo de facilitar la tarea del diseño de tal sistema (Sommerville, 2011).

Patrones *GRASP*[9]

Los Patrones Generales de Software para Asignar Responsabilidades (GRASP), describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. A continuación, se relacionan los patrones GRASP utilizados en el sistema (Larman, 2003):

Creador: Consiste en asignar a una clase determinada la responsabilidad de crear una o más instancias de otra. La clase creadora contiene, agrega, registra, utiliza o posee datos de inicialización de objetos de alguna otra clase (Larman, 2003). En el sistema, la clase reporte_recurso.php es la encargada de crear instancias de la entidad index_actividad.php.

```
$objExcel = new PHPExcel();
$objExcel ->getProperties()->setCreator("Codigos")->setDescription("Reporte");
```

Figura 2: Patrón Creador. Elaboración propia

[9] *General Responsibility Assignment Software Pattern*

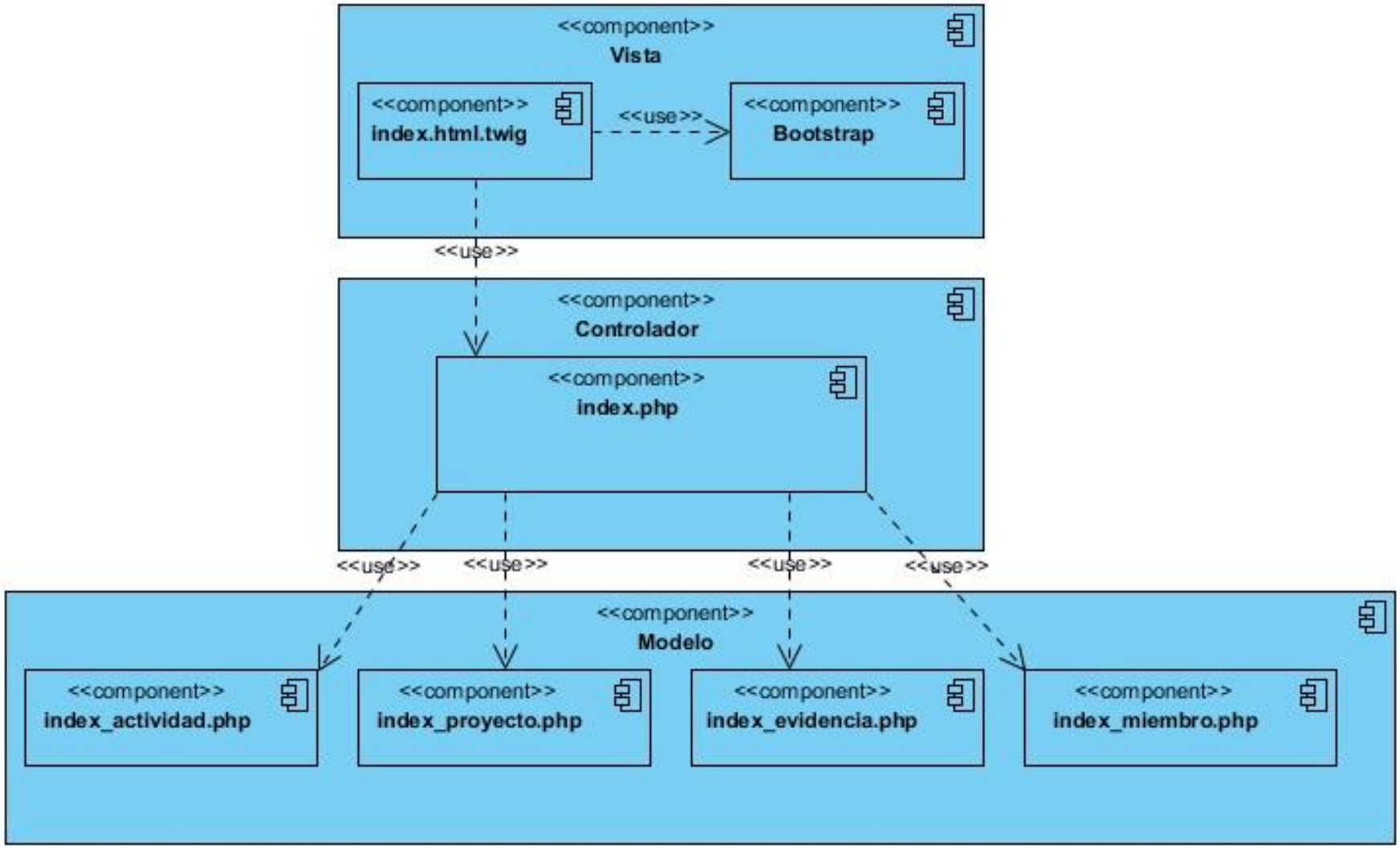

Figura 3: Diagrama de componentes. Elaboración propia

Alta cohesión: Propone que la información que almacena una clase debe de ser coherente y debe estar, en la medida de lo posible relacionada con la clase. Al realizar un cambio en una clase con alta cohesión, todos los métodos que pueden verse afectados, estarán a la vista, en el mismo archivo. Incrementa la claridad, la reutilización y la facilidad de comprensión del diseño (Larman, 2003).

Bajo acoplamiento: Este patrón expresa que entre las clases deberán existir pocas ataduras, es decir, estas estarán lo menos relacionadas posible de forma tal que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases, incrementando la reutilización y disminuyendo la dependencia entre las clases (Larman, 2003). El uso de este patrón se evidencia en la poca relación existente entre las clases.

Patrones GoF[10]

Los patrones *GoF* deben su nombre al grupo compuesto por sus creadores: *Erich Gamma, Richard Helm, Ralph Jhonson y John Vlisides*. Estos patrones se destacan por proporcionar elementos reusables en el diseño de sistemas de software y por su amplia estandarización de diseño. En el desarrollo de la presente investigación, se utilizaron los siguientes patrones:

Singleton: Es un patrón de diseño de software que restringe la creación de instancias de una clase a un objeto. Esto es útil cuando se necesita exactamente un objeto para coordinar acciones en todo el sistema (Larman, 2013). Este patrón se puede evidenciar en la clase index_actividad.php, como se muestra en la Figura 4.

```php
<?php
include 'gestion/conexion.php';
$sql = "SELECT * from proyecto WHERE activo ='t' ";
$resultado = mysql_query($sql);
$band = false;
?>
```

Figura 4: Patrón Singleton. Elaboración propia

[10] *Gang of Four*

Conclusiones del capítulo 2

- El empleo de las técnicas para la obtención de requisitos posibilitó comprender, identificar y describir los requisitos funcionales y no funcionales que deberá cumplir la solución.
- La realización del modelo de dominio permitió obtener un mejor entendimiento de las actividades realizadas en la gestión de los proyectos extensionistas.
- Se explicaron los elementos fundamentales correspondientes a la fase de planificación y diseño del sistema que se propone.
- El empleo de los patrones de diseño *GRASP* y *GoF* permitió identificar la reutilización y estandarización del diseño del sistema.
- Con la utilización apropiada de los patrones de diseño seleccionados y tomando en cuenta las prioridades del cliente se creó el plan de entregas, determinando el tiempo de desarrollo del sistema.

CAPÍTULO 3: VALIDACIÓN, IMPLEMENTACIÓN Y PRUEBAS

Introducción

En este capítulo se describe la validación de las variables de la investigación, la fase de implementación y posteriormente las pruebas al sistema desarrollado.

3.1. Validación del diseño

La validación del diseño mediante las métricas de diseño, permite medir de forma cuantitativa la calidad de los atributos internos del software, de forma tal que pueden ayudar al desarrollador a juzgar la calidad de un diseño a nivel de componente (Pressman, 2010).

Para la validación del sistema en cuestión se utilizaron las métricas siguientes:

- Métrica de relaciones entre clases (RC).
- Métrica de tamaño operacional de las clases (TOC).

3.1.1. Métrica relaciones entre clases.

Utilizando la métrica RC el resultado es dado por el número de relaciones de uso que se establecen entre una clase y las demás clases existentes y se evalúa a partir de atributos de calidad; acoplamiento, complejidad de mantenimiento, reutilización y cantidad de pruebas (Pressman, 2010).

Cada atributo de calidad se describe a continuación:

- **Acoplamiento:** un aumento del RC implica un aumento del acoplamiento de la clase.
- **Complejidad de mantenimiento:** un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
- **Reutilización:** un aumento del RC implica una disminución en el grado de reutilización de la clase.
- **Cantidad de pruebas:** un aumento del RC implica un aumento de la cantidad de pruebas necesarias para probar una clase.

Para aplicar la métrica RC es necesario categorizar cada una de las clases según la cantidad de relaciones que esta contenga. Para obtener un nivel óptimo de relación entre clases, el valor obtenido al aplicar dicha métrica debe ser directamente proporcional al acoplamiento y a la complejidad de mantenimiento; además debe ser inversamente

proporcional al nivel de reutilización del código. A continuación, se describe el atributo de calidad con sus categorías y criterios de evaluación:

Tabla 8: Atributos de calidad de la métrica RC. Elaboración propia

Atributo	**Categoría**	**Criterio**
Acoplamiento	Ninguno	RC = 0
	Bajo	RC = 1
	Medio	RC = 2
	Alto	RC > 2
Complejidad de Mantenimiento	Baja	RC <= Promedio
	Media	Promedio <=RC<= 2* Promedio
	Alto	RC >2* Promedio
Reutilización	Baja	RC >2* Promedio
	Media	Promedio< RC <= 2 *Promedio
	Alto	RC <= Promedio
Cantidad de pruebas	Baja	RC <= Promedio
	Media	Promedio<= RC <2*Promedio
	Alto	RC >= 2*Promedio

La aplicación del instrumento de evaluación de la métrica RC en el diseño del sistema propuesto, arrojó los resultados expuestos los siguientes gráficos:

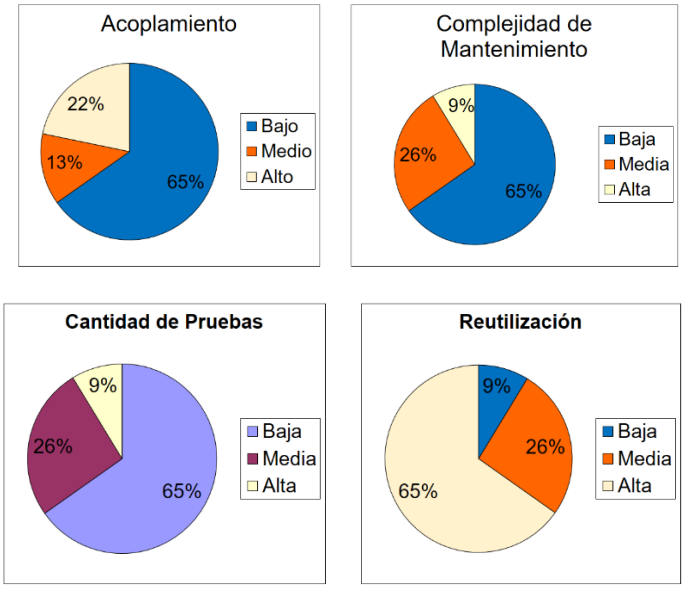

Figura 5: Resultados de la métrica RC. Elaboración propia

Luego de evaluar los resultados obtenidos de cada uno de los atributos de calidad, se obtuvieron los siguientes resultados:

- El 65% de las clases tienen un bajo acoplamiento.
- El 65% de las clases poseen una baja complejidad de mantenimiento
- El 65% de las clases tienen una baja de cantidad de pruebas
- El 65% de las clases poseen una alta reutilización.

Como se puede observar en la figura el acoplamiento posee un nivel bajo por lo que existe poca dependencia entre las clases, lo que posibilita una alta probabilidad de reutilización. También se puede apreciar que existe un medio nivel de complejidad de mantenimiento por lo que a la hora de mejorar los métodos y demás operaciones no es necesario realizar una gran cantidad de pruebas.

3.1.2. Métrica tamaño operacional de clases.

Las métricas basadas en el Tamaño Operacional de Clases están dadas por el número de métodos asignados a una clase, así como la cantidad de atributos q esta posea y el promedio que presenta el sistema en su totalidad. Se evalúa a partir de los siguientes atributos de calidad (Pressman, 2010).

- **Responsabilidad:** un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.

- **Complejidad de implementación:** un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
- **Reutilización:** un aumento del TOC implica una disminución del grado de reutilización de la clase.

Una vez analizado el indicador tamaño de clase, si el valor resultante tiende al crecimiento, es probable que la clase posea un alto grado de Responsabilidad; en consecuencia, el nivel de Reutilización sería mínimo y la implementación altamente compleja. A continuación, se describe el atributo de calidad con sus categorías y criterios de evaluación:

Tabla 9: Atributos de calidad de la métrica TOC. Elaboración propia

Atributo de calidad	**Categoría**	**Criterio**
Responsabilidad	Baja	TOC < =Promedio (Promedio)
	Media	TOC Entre Promedio y 2* Promedio
	Alta	TOC >2*Promedio
Complejidad de implementación	Baja	TOC < =Promedio
	Media	Entre Promedio y 2* Promedio
	Alta	TOC >2*Promedio
Reutilización	Baja	TOC >2*Promedio
	Media	TOC Entre Promedio y 2* Promedio
	Alta	TOC < =Promedio

La aplicación del instrumento de evaluación de la métrica RC en el diseño del sistema propuesto, arrojó los resultados expuestos en la siguiente figura:

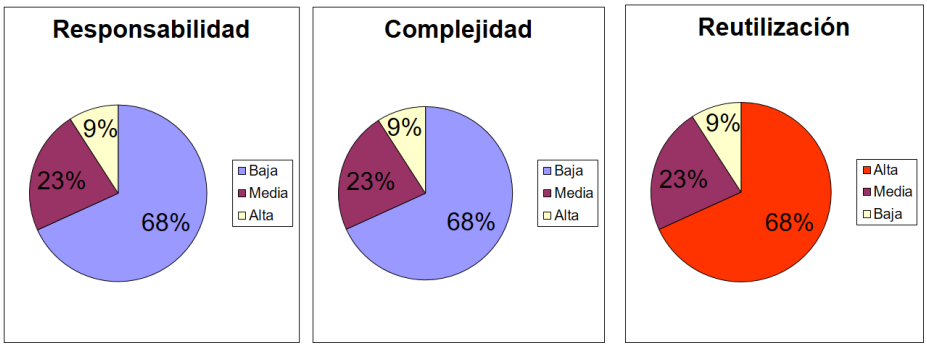

Figura 6: Resultados de la métrica TOC. Elaboración propia

Una vez evaluados los resultados obtenidos de cada uno de los atributos de calidad, se obtuvieron los siguientes resultados:

- El 68% de las clases poseen una baja responsabilidad de prueba.
- El 68% de las clases poseen una baja complejidad de prueba.
- El 68 % de las clases poseen una alta reutilización.

Obteniendo un bajo porciento de responsabilidad y complejidad de las clases se demuestra la calidad del diseño. Unidos a un alto grado de reutilización de las clases se facilita en gran medida la implementación de las mismas.

3.2. Implementación

Se seleccionan las HU para ser implementadas durante el transcurso de la iteración a la que pertenecen. Por estas razones, se lleva a cabo una revisión del plan de iteraciones y se modifican en caso de ser necesario. El propósito general de la implementación es desarrollar la arquitectura y el sistema como un todo. La aplicación que se desarrolle debe tener la calidad requerida para su uso y cumplir con los requisitos de software determinados en el segundo capítulo.

3.2.1. Estándares de codificación

Los estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código. Algunas de las ventajas que presenta el uso de estos son (Pressman, 2010):

- ❖ Legibilidad del código para que otros desarrolladores puedan comprender el sistema.

- ❖ Facilidades de mantenimiento y actualización del sistema, con código claro y bien documentado.

Comentarios

Es conveniente dejar información que pueda ser leída para entender qué se hizo en un fragmento de código. Los comentarios son escritos correctamente y claros en una sola línea.

```php
<?php //Funcionalidad insertar evidencia
$ida = $_GET['ida'];
$idp = $_GET['idp'];
?>
```

Figura 7: Comentarios. Elaboración propia

Sentencias

Las sentencias describen acciones algorítmicas que puede ejecutar un programa. Son elementos básicos en los que se divide el código según el lenguaje de programación que se utilice. Cada sentencia simple termina en punto y coma. En las sentencias compuestas la llave que inicia comienza en la línea siguiente al igual que la llave que termina y guardan la misma identación. La sentencia *while* se rige por la siguiente estructura:

```php
while($i<10){
    $objExcel->getActiveSheet()->setCellValue('A'.$fila,$i);
    $objExcel->getActiveSheet()->setCellValue('B'.$fila,$i);
    $objExcel->getActiveSheet()->setCellValue('C'.$fila,$i);
    $objExcel->getActiveSheet()->setCellValue('D'.$fila,$i);

    $fila++;
    $i++;
}
```

Figura 8: Sentencias. Elaboración propia

Camel Case

Para una mejor estandarización en el código del sistema se emplea el estándar de codificación CamelCase, y se puede dividir en dos tipos (Acedo, 2017):

- Upper Camel Case, cuando la primera letra de cada una de las palabras es mayúscula

- Lower Camel Case, igual que la anterior con la excepción de que la primera letra es minúscula.

En el caso del sistema en cuestión se utiliza Lower Camel Case, como se muestra en la siguiente figura:

```php
<?php
include 'gestion/conexion.php';
$sql = "SELECT * from proyecto WHERE activo ='t' ";
$resultado = mysql_query($sql);
$band = false;
?>
```

Figura 9: Lower Camel Case. Elaboración propia

3.3. Pruebas de Software

Las pruebas de software son un proceso iterativo que implican ejercer una implementación del software con datos de prueba. Durante su aplicación se examinan las salidas del software y su entorno operacional para comprobar que funciona tal y como se quiere. Las pruebas son una técnica dinámica de valoración de calidad de software (Sommerville, 2011).

Estrategia de pruebas

La estrategia de prueba describe el enfoque y los objetivos generales de la actividad de prueba. Incluye los niveles a que se realizarán las pruebas, el tipo de prueba a ser ejecutada y los métodos que se emplearán. Se especifican además las técnicas de prueba y herramientas que se usarán, así como los casos de prueba diseñados para lograr los objetivos. Debe permitir comenzar a evaluar por los componentes más simples y pequeños e ir avanzando progresivamente hasta probar todo el software en su conjunto (Sommerville, 2011).

3.3.1. Pruebas de caja blanca

Las pruebas de caja blanca se realizan sobre las funcionalidades internas de un módulo de software, y se encargan de comprobar los caminos lógicos, ciclos (bucles) y condiciones que debe cumplir el programa. Con el empleo de este método es posible desarrollar casos de prueba que garanticen la ejecución, al menos una vez, de los caminos independientes (Pressman, 2010). Para aplicar este método se decidió utilizar la técnica de ruta básica.

Técnica de ruta básica

La técnica de ruta básica tiene como objetivo comprobar que cada camino se ejecute independiente de un componente o programa, obteniéndose una medida de la complejidad lógica del diseño. Esta técnica de prueba debe ser utilizada para evaluar la efectividad de los métodos asociados a una clase, con el objetivo de asegurar que cada camino independiente sea ejecutado por lo menos una vez en el sistema (Pressman, 2010).

Pressman propone como estrategia para aplicar la ruta básica, realizar un análisis de la complejidad ciclomática[11] de cada procedimiento que componen las clases del sistema; una vez concluido este paso se selecciona el método con valor de contener errores, además de que ofrece una medida del número de pruebas que deben diseñarse para validar la correcta implementación de una determinada función. Esta métrica se calcula sobre un grafo y se puede realizar mediante tres formas distintas:

1. **V(G)** = R

2. **V(G)** = E – N + 2

3. **V (G)** = P + 1

Donde:

- G: Grafo de flujo
- R: Número de regiones.
- E: Número de aristas
- V(G): Complejidad ciclomática.
- N: Total de nodos.
- P**:** Número de nodos predicados[12].

[11] Métrica de software que proporciona una medición cuantitativa de la complejidad lógica de un programa.

[12] Nodo a partir del cual pueden tomarse dos o más caminos distintos

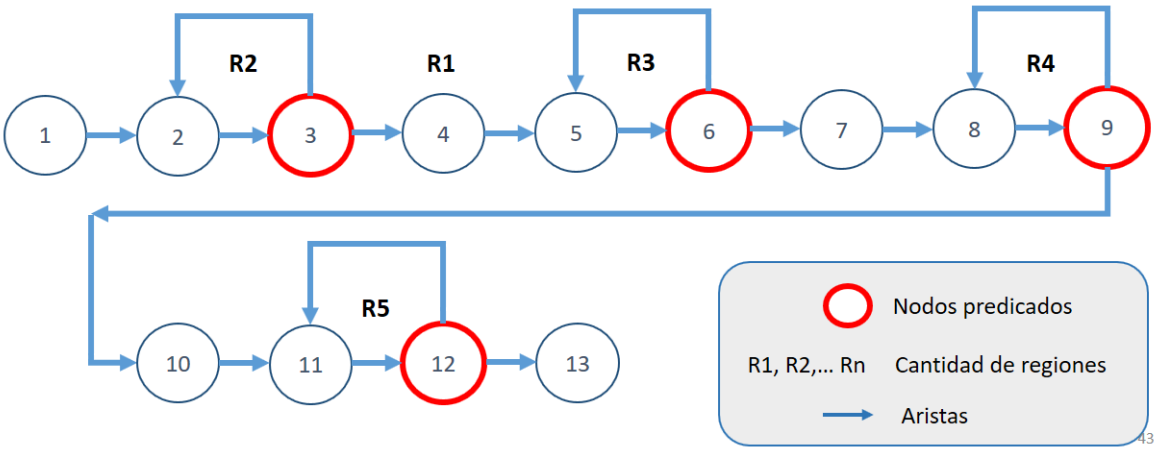

Figura 10: Grafo de flujo del método Generar_plantilla.php. Elaboración propia

El cálculo de la complejidad ciclomática arroja los siguientes resultados:

1. **V(G) = R** = **5**

2. **V(G) = E – N + 2** = 16 – 13 + 2 = **5**

3. **V (G) = P + 1** = **5**

Una vez calculada la complejidad ciclomática, el valor obtenido representa el límite superior de pruebas que deberán aplicarse (Pressman, 2010). Para cada camino fue diseñado un caso de prueba. En el Anexo 6 puede observarse el caso de prueba para el camino 1-2-3-4-5-6-7-8-9-10-11-12-13.

Resultados al aplicar la técnica de camino básico

La técnica fue aplicada a todos los métodos del sistema. Para comprobar la fiabilidad del código fuente se realizaron tres iteraciones completas, una por cada entrega a realizar, en busca de errores de codificación. En ninguna de las tres iteraciones se obtuvieron no conformidades, por lo que se puede concluir que luego de realizar la prueba de caja blanca, donde se diseñaron y ejecutaron los casos de prueba, se logró probar todos los caminos del código no encontrándose errores en el mismo.

3.3.2. Pruebas de caja negra.

La prueba de caja negra (o funcionales) se refiere a las pruebas que se llevan a cabo en la interfaz del software. Una prueba de caja negra examina algunos aspectos fundamentales

de un sistema con poca preocupación por la estructura lógica interna del software (Pressman, 2010).

Conociendo una función específica para la que fue diseñado el producto, se pueden diseñar pruebas que demuestren que dicha función está bien realizada. Dichas pruebas son llevadas a cabo sobre la interfaz del software, es decir, de la función, proporcionando unas entradas y estudiando las salidas para ver si concuerdan con las esperadas.

Para la aplicación de esta técnica a la solución obtenida, se confeccionaron casos de prueba por cada Historia de usuario (HU). A continuación, se representa el caso de prueba para la HU "Crear nuevo proyecto".

Tabla 10: Caso de prueba para la historia de usuario Crear nuevo proyecto. Elaboración propia

Caso de prueba de caja negra		
Código de caso de prueba: 01	**Nombre de historia de usuario:** Crear nuevo proyecto	
Nombre de la persona que realiza el caso de prueba: Cesar Borges Prado		
Descripción de la prueba: Comprobar que el sistema crea correctamente un nuevo proyecto, teniendo en cuenta cada condición y los distintos campos rellenados con el formato correspondiente.		
Condiciones de ejecución: El cliente debe acceder al hacer un uso correcto de sus privilegios de rol.		
Acción	**Entrada**	**Resultados esperados**
Hacer clic en la opción de crear proyecto	Texto	Se crea correctamente un proyecto y el sistema guarda los datos introducidos por el usuario
Evaluación de prueba: Satisfactoria		

Resultado de las pruebas de caja negra:

En la aplicación de las pruebas de caja negra se realizaron tres iteraciones, en las que se detectaron varias no conformidades que en la mayoría de los casos estuvieron dadas por errores de interfaz, de validación y ortografía. En la primera iteración se encontraron un total

de 16 no conformidades, de las cuales fueron analizadas y corregidas 11 de estas. En una segunda iteración fueron encontradas 11 no conformidades dándosele solución a todas. En la tercera se obtuvo un total de 0 no conformidades dándose por concluidas las pruebas. En la Figura 11 se muestra una gráfica con un resumen estadístico del número de no conformidades encontradas en cada iteración.

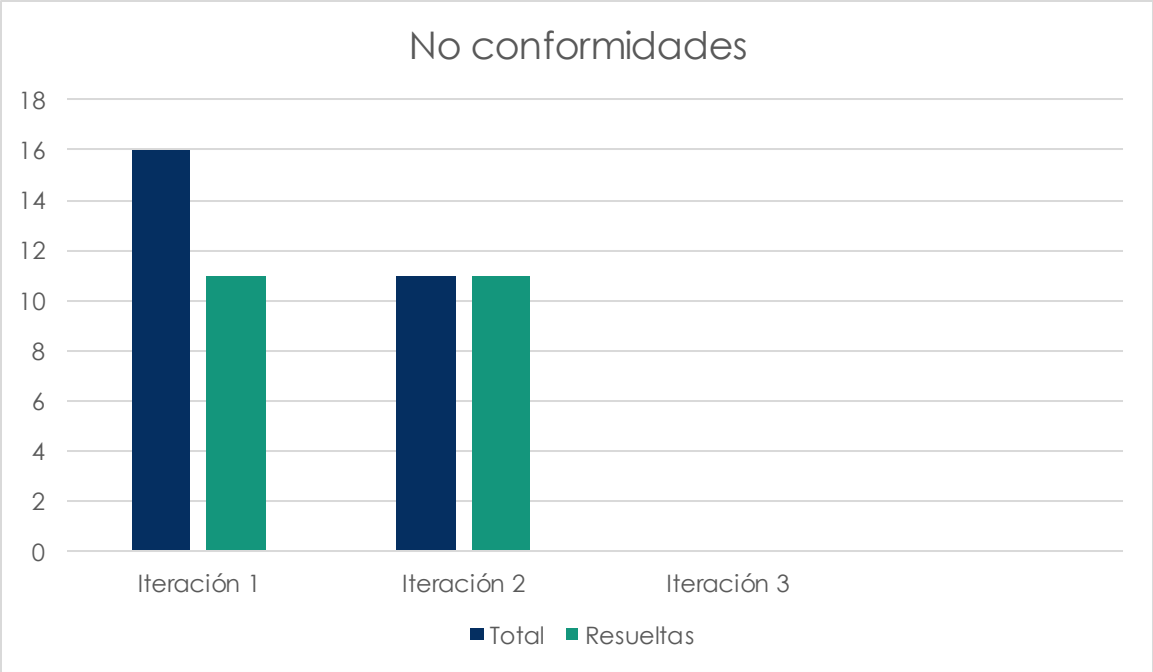

Figura 11: Gráfica de no conformidades. Elaboración propia

3.3.3. Pruebas de aceptación.

La prueba de aceptación es generalmente desarrollada y ejecutada por el cliente en conjunto con el equipo de desarrollo. Esta tiene como objetivo determinar cómo el sistema satisface sus criterios de aceptación, validando los requisitos identificados para el desarrollo del software, incluyendo la documentación y procesos de negocio. Está considerada como la fase final del proceso, para crear un producto confiable y apropiado para su uso (Los Andes Training, 2017). Con su empleo se persigue evaluar la disposición del sistema para su despliegue y posterior uso. Entre los datos que componen una prueba de este tipo están:

Iteración: número de la iteración realizada para la prueba.

Nombre del rasgo a probar: nombre del rasgo al que se le realiza la prueba.

No. del rasgo a probar: número del rasgo al que se le realiza la prueba.

Descripción del rasgo a probar: descripción breve del rasgo que se prueba.

Precondiciones: condiciones necesarias para poder realizar el caso de prueba y obtener los resultados esperados.

Pasos: pasos lógicos a seguir durante el desarrollo de la prueba para la obtención del resultado esperado.

Resultados esperados: descripción breve de los resultados esperados luego de realizar la prueba.

Evaluación: clasificación de la prueba dependiendo de la comparación del resultado obtenido con el resultado esperado.

Para mostrar la estrategia seguida durante las pruebas de aceptación y los resultados alcanzados, se tomó el requisito “Crear nuevo proyecto” a modo de ejemplo. A continuación, se describen las pruebas realizadas.

Tabla 11: Caso de prueba para el requisito "Crear nuevo proyecto" Iteración 1. Elaboración propia

Iteración No.: 1	
Nombre del requisito a probar: Crear nuevo proyecto.	**Número del requisito a probar:** RF4
Descripción del requisito a probar: el sistema debe tener una opción “crear un nuevo proyecto” donde se introducirá la información sobre el proyecto según formato descrito en el **¡Error! No se encuentra el origen de la referencia.**. La información de esta opción se podrá guardar siempre que estén llenas todas las solicitudes.	
Precondiciones: el usuario debe estar previamente registrado y autenticado en el sistema.	
Pasos: 1. El usuario acceder al sistema. 2. Ir a la pestaña agregar. 3. Llenar los campos correspondientes.	
Resultados esperados: se agrega el proyecto satisfactoriamente a la lista de proyectos.	
Evaluación: fallida	

Observaciones: permite que la fecha final del proyecto sea menos que la fecha inicial.

Tabla 12: Caso de prueba para el requisito "Crear nuevo proyecto" Iteración 2. Elaboración propia

Iteración No.: 2	
Nombre del requisito a probar: Crear nuevo proyecto.	**Número del requisito a probar:** RF4
Descripción del requisito a probar: el sistema debe tener una opción "crear un nuevo proyecto" donde se introducirá la información sobre el proyecto según formato descrito en el **¡Error! No se encuentra el origen de la referencia.**. La información de esta opción se podrá guardar siempre que estén llenas todas las solicitudes.	
Precondiciones: el usuario debe estar previamente registrado y autenticado en el sistema.	
Pasos: 1. El usuario acceder al sistema. 2. Ir a la pestaña agregar. 3. Llenar los campos correspondientes.	
Resultados esperados: se agrega el proyecto satisfactoriamente a la lista de proyectos.	
Evaluación: satisfactoria	
Observaciones: se creó el proyecto correctamente	

Resultado de las pruebas de aceptación:

En la ejecución de las pruebas de aceptación el sistema fue revisado por los clientes. Se realizaron dos iteraciones. En la primera se identificaron 5 errores, 4 de interfaz y 1 de validación, los cuales fueron corregidos en una segunda iteración, emitiéndose el acta de liberación por parte de los clientes.

3.4. Validación de las variables de la investigación

Partiendo del punto de que el problema a resolver definido por los autores de la presente investigación es: ¿Cómo mejorar la disponibilidad y centralización de la información de los

proyectos extensionistas?, se presenta cómo se resolvieron las deficiencias identificadas en la situación problemática con la solución propuesta a través de un análisis de cómo se realizaba el proceso antes del desarrollo del sistema y después de la utilización del sistema.

Antes

El proceso de gestión de la información de los proyectos extensionistas en la Facultad 3 se realiza de forma manual con apoyo de las computadoras, sin embargo, se presentan problemas que atentan contra una correcta gestión de esta información:

- Dentro de la documentación asociada a los proyectos pueden encontrarse las plantillas de los proyectos, encuestas y otros archivos importantes. Parte de esta es almacenada en grandes volúmenes y formatos, su mayoría en papel, lo que puede provocar pérdidas de información por deterioro o extravío.
- La confección y actualización de la plantilla del proyecto extensionista se realiza de forma manual por los coordinadores de proyecto, teniendo que trabajar en el documento cada vez que es solicitada una actualización.
- Para llevar a cabo la realización de las actividades de cada proyecto el vicedecanato debe reunirse con los coordinadores y despachar la necesidad de recursos que son necesarios para la realización de la actividad, lo que trae consigo pérdida de tiempo y que se atrasen otras actividades importantes.
- Las evidencias de las actividades de los proyectos extensionistas no están centralizadas. Cada coordinador del proyecto guarda las suyas, no existiendo un lugar donde estén todas y que sea accesible desde la dirección del vicedecanato de extensión y residencia. Parte de estas evidencias se pierden al no ser recogidas en el momento.

Después

El uso de un sistema para la gestión de los proyectos extensionistas de la Facultad 3 con el fin de gestionar toda la información que se genera dentro de los proyectos brinda los siguientes beneficios:

- El sistema permite gestionar toda la información asociada a los proyectos extensionistas, tanto la plantilla del proyecto, como las actividades que en él se realicen con sus respectivas evidencias, lográndose tener la información oportuna de cada proyecto.

- Cuenta con una funcionalidad que permite exportar la planilla del proyecto actualizada con los datos que en él se encuentran, por lo que una vez que se comience a hacer uso del mismo no se hace necesario realizar una actualización manual de la planilla de los proyectos.
- El sistema permite realizar reportes que le permitirán al vicedecano de extensión y residencia acceder a los recursos que cada proyecto necesita y darle seguimiento al accionar de cada proyecto, sin necesidad de para ello tener que comunicarse con los coordinadores.
- Las evidencias van a ser guardadas en el sistema como parte de la información de los proyectos, estando centralizadas por la herramienta, lo que va a permitir a los usuarios poder acceder a ellas siempre que sea necesario, quedando guardada una memoria histórica del accionar de los proyectos extensionistas en la facultad.

Conclusiones del capítulo 3

Teniendo en cuenta los objetivos trazados en la investigación y en aras de dar cumplimiento al problema propuesto, se arribó a las siguientes conclusiones:

- Se implementaron las funcionalidades requeridas, utilizando los estándares de codificación y las buenas prácticas de programación.
- El uso de las métricas TOC y RC arrojó resultados satisfactorios, demostrándose que el diseño propuesto presenta buena calidad.
- Se realizaron pruebas de caja blanca, caja negra y de aceptación, lo que posibilitó verificar el correcto funcionamiento del sistema, corrigiéndose las no conformidades detectadas.
- La validación de las variables de la investigación arrojó como resultado que el sistema propuesto brinda una solución a la problemática planteada.

CONCLUSIONES GENERALES

Con la culminación de la presente investigación se obtuvo como resultado, la implementación del Sistema para la gestión de proyectos extensionistas. A continuación, se exponen las conclusiones generales:

- El estudio realizado como parte de la investigación sirvió de apoyo en la toma de decisiones con vista al desarrollo de la propuesta de solución, demostrando que no existe un sistema informático que permita gestionar los proyectos extensionistas, teniendo en cuenta las relaciones y nexos de derivación que establece el Programa Nacional de Extensión Universitaria, para las formas organizativas del proceso extensionista.
- La realización del modelado del negocio y la descripción de los procesos identificados, sirvió de guía para el posterior diseño de la propuesta de solución.
- El empleo de la metodología, las herramientas, tecnologías y lenguajes de desarrollo seleccionados soportaron todo el proceso de desarrollo de la propuesta de solución. Además, los artefactos generados en correspondencia con la metodología de desarrollo XP, facilitaron la comprensión e implementación del sistema desarrollado.
- La realización de las pruebas al sistema desarrollado, permitió detectar y corregir los errores presentes en el mismo.

RECOMENDACIONES

- Se recomienda desarrollar otras funcionalidades e incorporarlas al sistema, con el fin de potenciar y enriquecer las prestaciones de este.

BIBLIOGRAFÍA REFERENCIADA

1. Ministerio de Educación Superior de la República de Cuba, 2004. Accedido 6 de diciembre de 2018. http://www.mes.gob.cu/es.
2. Programa Nacional de extensión Universitaria, 2004. Accedido 25 de Febrero del 2019 https://www.mes.gob.cu/sites/default/files/documentos/EU_PNEU.pdf
3. Bienestar Universitario, Universidad de las Ciencias Informáticas, 2001. Accedido 6 de diciembre de 2018. http://www.uci.cu/vida-universitaria/bienestar-universitario.
4. Escobar Arraigada, Claudio, 2006.'WordPress' y la creación de un sitio Web dinámico: metodología de instalación y puesta en marcha. Chile A.G : Héctor Gómez Fuentes, Director Departamento de Gestión de Información, 2006.
5. UCI, Inscripción de proyectos extensionistas universitarios.doc.
6. Encyclopedia of Computer Science. 2003. *Management information systems (MIS).* 4ta edición,: s.n., 2003. ISBN: 0-470-86412-5.
7. Laffita Nicot, Liumila. 2008. Trabajo de diploma: Gestión de la participación y los resultados obtenidos por los estudiantes en las actividades extracurriculares.
8. García Brito, Imilsy y Carrillo Feria, Diana. 2010. Trabajo de diploma: Desarrollo de una Aplicación Web para la Gestión de los procesos de Extensión y Residencia de la Facultad 15.
9. Lipsy Mariem Legrá Legrá 2011. Trabajo de diploma: Sistema de información para el área de residencia y extensión universitaria de la Facultad 6.
10. Gómez Hernández, Celia Rosa y Céspedes Lara, Leandro Lázaro. 2013. Trabajo de diploma: Solución para la gestión de información de la Extensión Universitaria en el área de Cultura Física en la Universidad de las Ciencias Informáticas.
11. Cabeza Matos, Kirenia y Arias Santos, Javier. 2013. Trabajo de diploma: Solución para la gestión de información de los procesos de Extensión Universitaria en el área de Extensión Cultural de la Universidad de las Ciencias Informáticas.
12. Vázquez Campos, Adonis. 2014. Trabajo de diploma: Sistema Web de la Dirección de Extensión Cultural de la Universidad de las Ciencias Informáticas.
13. Leyva Feijoó, Lidia del Carmen y López Rodríguez, Alejandro. 2015. Trabajo de diploma: Aplicación web para la gestión de los procesos extensionistas de la Facultad 1.
14. «Mauro Callejas Cuervo, Oscar Yovany Baquero Moreno. 2007. *Herramientas libres para modelar software.pdf* .»

15. Modelo XP para desarrollo de proyecto. [En línea], 2011. [Consultado: 2 marzo 2019]. Disponible en: http://es.slideshare.net/johitaamiga/modelo-xp-para-desarrollo-de-proyecto.
16. MEZA, Mirna. Herramientas Case: INTRODUCCIÓN. Herramientas Case [En línea]. sábado, de abril de 2011. [Consultado: 11 de marzo del 2019]. Disponible en: http://fdsherramientascase.blogspot.com/2011/04/i-n-t-r-o-d-u-c-c-i-o-n.html
17. Visual Paradigm for UML (ME) - (Paradigma Visual para UML (ME)) (Visual Paradigm for UML (ME)) por Visual Paradigm International Ltd. - reporte y descarga. [En línea]. [Consultado: 11 de marzo del 2019]. Disponible en: http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/
18. Alhir, Sinan Si . 2003. Learning UML. s.l. : O'Reilly, 2003.
19. Axure RP, 2018 http://www.axure.com/
20. Figueroa, Roberth G., Solís, Camilo J.,Cabrera, Armando A., 2010 Metodologías Tradicionales vs. Metodologías Ágiles.».
21. ARIAS PATIÑO, J. D.; LOZANO ISAZA, M. A., et al. (2011). Desarrollo de un sistema computacional para el aprendizaje de programación estructurada" SCAPE". 2011.
22. HERNÁNDEZ, C.; ROSENDO, L., et al. (2010). Estándares de Diseño Web. Ciencias de la Información, 2010, vol. 41, nº 2, p. 69-71. Disponible en: http://cinfo.idict.cu/index.php/cinfo/article/viewArticle/34.
23. *Symfony*. (2019). Obtenido de SymfonyWorld: https://symfony.com
24. Manual de CSS. [En línea] [Citado el: 7 de abril de 2019.] http://www.manualdecss.com/
25. Libros Web. [En línea] [Citado el: 10 de junio de 2019.] www.librosweb.es/javascript.
26. Rodriguez, K. (2013). *Apache Server*. Obtenido de http://webapache.blogspot.com/
27. CERVERO, J. C. (2011). Content Management System. 2011, vol. 1, nº Disponible en: http://upcommons.upc.edu/pfc/handle/2099.1/11459.
28. ARRAIGADA, C. E. y LLANCAO, J. L. (2006). " WordPress" y la creación de un sitio Web dinámico: metodología de instalación y puesta en marcha. Serie Bibliotecología y Gestión de Información, 2006, vol. 10. Disponible en: http://eprints.rclis.org/7102/1/serie10.pdf.
29. Chacón, Elizabeth de los Milagros Ramírez. Sistema de gestión de calidad del grupo de Calidad de FORTES. Módulo Pruebas. La Habana : s.n., 2011.

30. Marqués, Mercedes. *Apuntes de Ficheros y Bases de Datos.* Castellón de la Plana, España : Ingeniería Técnica en Informática de Gestión de la Universidad Jaume, 2009.
31. UCID: SOFTWARE-DEFENSA. 2012. PRODESOFT-Proceso de Desarrollo y Gestión de Proyectos de Software. 2012. Vol. 1.5.
32. Luttrell, Joe. y Dobson, Michael. LESSONS FROM HISTORY. Functional versus Non Functional Requirements and Testing. 2013. [En línea]. [Consultado: 11 de marzo del 2019]. Disponible en: http://www.lessons-fromhistory.com/node/83.
33. Joskowicz, José . Reglas y Prácticas en eXtreme Programming [En línea]. 2 octubre 2008. [Consultado: 11 de marzo del 2019] Disponible en: http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf
34. PENADÉS, María del Carmen. Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). 2008.
35. Anónimo. Artefactos XP [En línea]. [Consultado el 9 de mayo de 2019] Disponible en http://www.abacovirtual.edu.pe/chiclayo/filedocente/140318-CEI2012203 MB-000029-19072012-173247.pdf
36. Manuel Calero Solís. Programación-extrema. [En línea] 2013. [Citado el: 16 de 05 de 2013.] http://programacion-extrema.wikispaces.com/7.
37. Pressman, Roger S. (2010). *Ingeniería del software. Un enfoque práctico. 6ta Edición.* New York.
38. Acedo, J. I. (1 de Septiembre de 2017). *Apuntes de Programacion*. Obtenido de Estándares de nomenclatura: Snake Case, Kebab Case, Camel Case: http://programacion.jias.es/2017/09/estandares-de-nomenclatura-snake-case-kebab-case-camel-case/
39. Sommerville, Ian. (2011). *Sommerville_Parte_II_Requerimientos 7ma Edición.* Madrid(España).
40. Larman, Craig. UML y Patrones. Introducción al análisis orientado a objetos y al proceso unificado. 2003. s.l: Pearson Educación, 2003. ISBN 9788420534381.
41. Los Andes Training. [En línea] 23 de Agosto de 2017. [Consultado 5 de junio de 2019] Disponible en: https://losandestraining.com/2017/08/23/que-son-las-pruebas-de-aceptacion/.
42. Visual Studio, 2019 https://visualstudio.microsoft.com/es/