Universidad de las Ciencias Informáticas

Facultad 3

Centro de Calidad

Título: *Sistema para gestionar actividades de calidad en proyectos de desarrollo*

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas.

Autor: Noel Harrinso Hidalgo Reyes.

Tutores: **Ing**. Roberto Menejías García, **MsC**. AymaraMarinDiaz, **DrC**. Yaimí Trujillo Casañola

Junio, 2020

"Año 62 de la Revolución"

Declaración de autoría

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____del año_____.

Noel Harrinso Hidalgo Reyes

Autor

_____

DrC. Yaimí Trujillo Casañola

Tutor

_____

Ms.C. AymaraMarínDíaz

Tutor

_____

Ing. Roberto Menejías García

Tutor

_____

Agradecimientos

A Dios por estar aquí hoy, por la salud y por las muchas bendiciones,

A mi familia por ser un punto de apoyo, los consejos, las experiencias, por el amor incondicional que me han otorgado. A tíaYayi y tío Luis, tíaOdalis y tío Armando,tíaYaquelin y tío Luis Ángel, tía Concha y tío Orlando, y tía Tere, que son mis segundas madres y mis segundos padres, sacrificados hasta el final por mí, los quiero, a Naylin, a Pedri, gracias, a Duly y su familia, a Nayari y su familia, y a todos los demás que son especiales para mi vida, gracias, no quisiera más familia que esta.

A mis padres, por estar ahí con ese amor que los caracteriza y los disimiles consejos que me enseñaron para lograr mis metas y objetivos en la vida.Los quiero.

A mi hermano, por las vivencias, por alegrarme y escucharme en los momentos más difíciles.

A mis amigos y compañeros, que siempre estuvieron ahí conmigo, en las buenas y malas, en las apretadas, agradecido por todo lo que han hecho por mí., a Carli, Lucio, Franci, Yudiel, mi primo Ernesto, Riki, Yeili, Sucel, Adriana, Carlo, Yoandri Freire, Melisa, Jaime, Dariel, Yaniel, mi primo Aciel, Enmanuel y el resto de la zona, los de Pilón y los de Miraflores, un abrazo para todos.

A la familia de Irina, por los mejores y más divertidos momentos de mi vida, por los consejos y preocupación, Deysi, un beso grande, Adanay, Eblis, Kukita, Evelin, Kevin, Yani,Ismaray, Yamira,Yoelvis y Mayedo, gracias.

A mis tutoras, por su ayuda inmensa, y su confianza.

A mi querida Irina, quien ha transitado conmigo por los duros y más alegres caminos, por llevar siempre conmigo el peso de la experiencia, por llorar y reír conmigo, por todo, gracias a mi amor, te amo con todo mi corazón…..

Dedicatoria

A Dios siempre presente.

A mi familia y a mis padres, a Irina, y todo aquel que siempre me dio su mano

A esta Patria que nos dió Fidel, a mi nación, a mi Cuba.

Resumen

Realizar de forma adecuada actividades que garanticen la calidad del producto en los proyectos en desarrollo de software, de forma continua permite disminuir el esfuerzo de corrección de defectos durante la ejecución de los proyectos.

La investigación propone el desarrollo de un sistema de aplicación web que utilice el razonamiento basado en casos (RBC), con el objetivo de sugerir actividades de calidad a realizar en las diferentes fases de un proyecto en desarrollo, basado en el resultado de experiencias anteriores y el criterio de expertos en el área.

Para realizar la validación de la propuesta de solución se utilizan técnicas de caja blanca, caja negra, método Delphi y pruebas de aceptación, para verificar el adecuado funcionamiento de la solución.

PALABRAS CLAVE

Actividades, calidad, proyectos, software,disminución, defecto, razonamiento.

Tabla de contenido

Tabla de contenido

Tabla de contenido

Índice de Figuras

Índice de Tablas

Introducción

INTRODUCCIÓN

La ingeniería de software es una ciencia que es desarrollada y aplicada mundialmente, debido al contexto globalizado que demandan las Tecnologías de la Información y las Comunicaciones (TICs). La ingeniería de software (ISW) comprende todos los aspectos de la producción del software desde las primeras etapas de la especificación del sistema, hasta el mantenimiento de éste después de su uso. Desde sus inicios ha aplicado un enfoque sistemático, disciplinado y organizado con el objetivo de alcanzar un software de alta calidad que sea producido en tiempo, con el costo planificado y que funcionen los requisitos pactados con el cliente (1).

La calidad de un producto está relacionada con el "Grado en que un conjunto de características inherentes de un objeto cumple con los requisitos" (2).Se refiere a la capacidad que posee un objeto, entidad, componente, para satisfacer necesidadesimplícitasoexplicitas. Es decir, en el fondo es uncumplimientoderequisitos(3).

El concepto de calidad de software (CSW)es abordado por numerosos autores entre los que destacaPressman, donde argumenta que la calidad está asociada al cumplimiento de exigencias establecidas por consumidores y por documentación técnica elaborada a través de la ingeniería del software durante el ciclo de vida de un proyecto (1). Por tanto,se asume que la calidad está vinculada a las características que debe cumplir un producto según su finalidad y exigencias establecidas previamente por el cliente.

Un producto para la aceptación y conformidad del cliente depende de si tiene la calidad requerida, es decir, si la expectativa coincide con la experiencia real que viva el cliente."La sociedad demanda productos y servicios informáticos que den respuesta a las necesidades del mercado, donde cada vez se es más riguroso a la competitividad existente" (4).La producción no satisface la demanda y los costos son altos, en la mayoría de los casos, por no aplicar las buenas prácticas de la Ingeniería y Gestión del Software. Estas prácticas son de vital importancia, pues les da seguridad a los usuarios y clientes de estos productos, proporcionándoles a los mismos sus pedidos a tiempo y con buena calidad (5).

Los usuarios verán una mayor calidad en un producto de software en la medida que éste responda a los requerimientosde la eficiencia del desempeño, tenga facilidad de uso, disponga de una

correcta ayuda y la documentación del usuario final sea realmente útil, entre otros. Estas apreciaciones de calidad hacia un determinado producto, elevarán el nivel de confianza para la organización desarrolladora, lo que puede elevar su posición en el mercado (6).

Basado en los conceptos deISW y CSW los que sonfundamentales para el desarrollo de software, se puede decir que se busca mejorar la calidad del producto final, desde la calidad del proceso y del propio producto. Donde, la calidad de un producto de software está directamente relacionada a su cantidad de defectos. Los defectos de un producto de software deben ser detectados lo antes posible y así evitar el re-trabajo (7). Existen evidencias que demuestran que entre el 40% y el 50% del esfuerzo de un proyecto, se emplea en re-trabajo que podría ser evitado (4).

La calidad ha evolucionado y se ha convertido en un importante punto diferenciador en la industria del software. Para alcanzar la calidad requerida de un software es determinante desarrollar un grupo de actividades que la garantice, como las que se refieren a realizar revisiones de adherencia a procesos y productos, con el objetivo de la evaluar la adherencia a los procesos y productos de trabajo. Además, verificar la correcta transición de una fase a la siguiente. Realizar revisiones técnicas formales (RTF) que tiene como objetivos la revisión técnica de los productos de trabajo que se identifiquen en la institución como entregable al cliente o aquellos solicitados o por necesidad de la institución por deficiencias detectadas en análisis realizados (4).

Y finalmente las pruebas que sus objetivos son comprobar que el producto a desarrollar cumple con los requisitos establecidos por el cliente, teniendo en cuenta las recomendaciones de buenas prácticas propuestas por los modelos, estándares y guías más recomendados a nivel internacional (4).

Para los miembros de los proyectos y a gerencia de las organizaciones que desarrollan software es muy difícil decidir, ¿cuándo realizar estas actividades, en qué momento de la ejecución del proyecto es mejor realizarlos o con qué frecuencia revisar que se cumplan estas actividades?, son decisiones que necesitan tomar las direcciones de los proyectos.

La toma de decisiones basado en datos, en datos históricos, en criterios, en análisis puede ser utilizado en todas las fases del ciclo de vida de un proyecto, con el objetivo de establecer criterios

que garanticen la calidad de los productos o servicios, puesto que de ello depende la satisfacción del cliente para el cual se obtiene el producto final (8).

Conseguir que el éxito sea un referente en la gestión empresarial depende de la habilidad de tomar buenas decisiones. Las decisiones inteligentes hacen que los proyectos prosperen, en cambio las decisiones tomadas a la ligera bloquearán y perjudicarán el rendimiento del equipo de trabajo del proyecto(9). Hacer lo correcto en el momento adecuado evitaría costos mayores porque un producto sin defectos, no se tiene que rehacer o mejorar, por tanto, no hay pérdida de tiempo, ni costos por ese concepto. La toma de decisiones es una parte fundamental en la vida de los proyectos e influyen completamente en el producto final.

En Cuba, en aras de ampliar el mercado, mejorar la productividad, aumentar la satisfacción de los clientes y garantizar la calidad de los productos que se desarrollan, se han dado pasos estratégicos en la industria, un ejemplo es la ejecución de programas de mejora (4), en organizaciones como la Universidad de las Ciencias Informáticas (UCI), XETID, DATYS, entre otros.

A pesar de los esfuerzos realizados aún se evidencian problemas en los productos finales y en la satisfacción de los clientes. En análisis realizadosen las estadísticas sobre todos los proyectos de la UCI del año 2014se muestra que solo un 10 % de los proyectos se finalizaron con éxito, un 9% del total de los proyectos resultaron detenidos y los proyectos cerrados con atraso representaron el 28 % (el comportamiento en el 2019 sigue siendo el mismo). La principal causa de este comportamiento según los especialistas de la dirección fue el esfuerzo dedicado a la corrección de defectos en etapas avanzadas del desarrollo y que no existe un sistema que fomente la utilización de las actividades de calidad en la toma de decisiones a partir de experiencias anteriores.Al realizar un análisis de los datos, se pudo constatar que varios de los defectos detectados se introdujeron en etapas tempranas del desarrollo del software, al no tener conocimiento suficiente sobre las actividades de calidad que buscan encontrar defectos durante el desarrollo (4).Esto evidencia que, el aseguramiento de la calidad del software se diseña para cada aplicación antes de comenzar a desarrollarla y no después (10).

Introducción

La tendencia en la UCI ha demostrado que en el desarrollo de los proyectos existen varios casos que culminan correctamente de acuerdo a los requisitos, pero que no cumplen en tiempo con los hitos planificados desde el inicio, lo que reporta valores negativos para los proyectos venideros (9).En cuanto a factores que determinan la calidad de un producto como que cumpla con las características operativas, la capacidad para soportar los cambios, y la adaptabilidad a nuevos entornos (10), la no aplicación de manera correcta y constante de las actividades de calidad en los proyectos, es una de las causas fundamentales de que los productos finales sean liberados en una 3era o 4ta iteración y no satisfagan adecuadamente la demanda de los clientes (4).

Transformar la información y/o datos en conocimiento es útil, pero resulta engorroso realizar esta acción de forma manual cuando existe un gran cúmulo o es necesario la intervención para ello de expertos sobre determinado conocimiento, ambas opciones provocan pérdida de tiempo y aumento del costo de cualquier proyecto. El término inteligencia artificial (IA) se refiere a la capacidad de emular las funciones inteligentes del cerebro humano (11). Los sistemas basados en el conocimiento constituyen paradigmas computacionales de la Inteligencia Artificial (IA). Utilizan conocimiento sobre un dominio específico. La solución que se obtiene es similar a la obtenida por una persona experimentada en el dominio del problema (12). Por lo tanto,resulta ventajoso el empleo de técnicas de inteligencia artificial (IA) para simular las actividades de expertos y apoyar el proceso de toma de decisiones, entre las cuales destacan como sistemas basados en conocimiento: las redes neuronales artificiales, los algoritmos genéticos, los sistemas basados en reglas, los sistemas basados en probabilidades y los sistemas basados en casos.

No existe una herramienta que sugiera actividades o acciones para asegurar la calidad basado en datos históricos de proyectos finalizados para evitar errores, fallos o defectos durante las etapas tempranas de un proyecto en desarrollo; quepermita crear conocimiento a través de experiencias previamente obtenidas, útiles para la toma decisiones en cada fase del ciclo de vida de un proyecto, que incida en el resultado final del producto, y que sea aplicado para mejorar la calidad de los procesos del desarrollo del software, desde el inicio hasta el final del proyecto. Que tenga como objetivo minimizar o disminuir el re-trabajo a los miembros del equipo de desarrollo.

Partiendo del análisis realizado se plantea el siguiente **problema de investigación**:

Introducción

¿Cómo disminuir el esfuerzo de corrección de defectos durante el ciclo de vida de los proyectos de desarrollo de software teniendo en cuenta el conocimiento y experiencia de casos anteriores?

A partir del problema de investigación planteado se define como **objeto de estudio**: corrección de defectos durante el ciclo de vida de los proyectos de desarrollo de software. Enmarcado en el **campo de acción**: Las actividades de calidad durante un proyecto en desarrollo teniendo en cuenta conocimiento y experiencia de casos anteriores.

Objetivo general: Desarrollar un Sistema Basado en el Conocimiento que permita identificar y contribuir a la disminución del esfuerzo de corrección de defectos durante el ciclo de vida de los proyectos de desarrollo de software.

De dicho objetivo general se derivan los siguientes **objetivos específicos:**

- ✓ Elaborar el marco teórico de la investigación, a partir del análisis a nivel nacional e internacional de las herramientas existentes donde sugieran utilizar actividades de calidad que aplique sistema basado en conocimiento.
- ✓ Diseñaruna herramienta que mediante el uso de un sistema basado enconocimiento permita sugerir las actividades de calidad adecuadas a las características y necesidades del proyecto.
- ✓ Validar la herramienta que mediante el uso delsistema basado en conocimiento permita sugerir las actividades de calidad para contribuir a la disminución del esfuerzo de corrección de defectos durante el ciclo de vida de los proyectos de desarrollo.

Para dar cumplimiento a los objetivos específicos se trazan las siguientes **Tareas de Investigación**:

- ✓ Análisis de diferentes soluciones existentes en un marcodonde se recomiende actividades de calidad en los proyectos de desarrollo de software.
- ✓ Análisis de los sistemas basados en conocimientocomo técnicas de la inteligencia artificial.
- ✓ Definir el sistema basado en conocimiento a utilizar para lograr el objetivo general.

- ✓ Analizar los datos de proyectos para identificar el conocimiento asociado alrededor de ellos que deba persistir en el sistema.
- ✓ Análisis de herramientas, tecnologías y metodología de desarrollo de software.
- ✓ Selección de las herramientas, las tecnologías y la metodología de desarrollo de software a utilizar en el desarrollo del sistema.
- ✓ Definición de la estrategia para la realización de pruebas al sistema.
- ✓ Utilizar método de caja blanca para la validación del código del sistema.
- ✓ Utilizar método de caja negra para la validación de las funcionalidades del sistema.
- ✓ Utilizar pruebas internas y de aceptación.

De acuerdo a todo lo antes expuesto se propone la **idea a defender**:

El desarrollo de una herramienta basada en el conocimiento permitirá identificar y contribuir a la disminución del esfuerzo de corrección de defectos durante el ciclo de vida de los proyectos de desarrollo de software.

Como **resultado esperado** se obtendrá una herramienta que disminuirá el esfuerzo de corrección de defectos durante el ciclo de vida de los proyectos de desarrollo de software.

Métodos Científicos de Investigación

Para el desarrollo de la investigación es necesario la fundamentación a través de los métodos científicos, siendo utilizados en la presente los siguientes:

Métodos Teóricos:

Histórico-Lógico: Se enfocó el estudio de los sistemas basados en conocimientos, y elementos de calidad aplicados en proyectos de desarrollo de software que como resultado obtienen la disminución del esfuerzo en la corrección de defectos, desde un enfoque histórico lógico de forma general. Permite determinar las principales causas del desenlace del problema, así como las posibles soluciones a poner en práctica, a partir del análisis de estudios anteriormente realizados.

Análisis y síntesis: Permite desarrollar un estudio sobre las actividades de calidad en los proyectos y las experiencias adquiridas durante la actividad productiva en la organización, además de enfatizar en las buenas prácticas asociadas a este proceso. Se realiza una síntesis de los conceptos y definiciones sobre el tema luego del análisis de la bibliografía.

Sistémico: En función de organizar la información encontrada de manera dispersa y obtener una conceptualización más clara sobre sistemas basados en conocimientos y su relación con la toma de decisiones que disminuya el esfuerzo de corrección de defectos en un proyecto.

Métodos Empíricos:

Entrevista: Se conocerá las especificaciones y funcionalidades de las herramientas similares existentes, además de las deficiencias que tienen para ser solucionada en la herramienta basada en conocimiento a desarrollar, a partir de las necesidades del cliente.

Para facilitar su comprensión, el documento está estructurado en tres capítulos, conclusiones, figuras y anexo, a continuación, se realiza una breve descripción:

Capítulo 1: Se analizan e investigan las actividades de calidad en los proyectos de desarrollo teniendo en cuenta conocimiento y experiencia de casos anteriores. Se referencian conceptos básicos, metodología y herramientas.

Capítulo 2: Se describen las necesidades del negocio, así como la especificación de requisitos de información, funcionales y no funcionales. Se realiza el análisis, diseño e implementación de la herramienta basada en conocimientos.

Capítulo 3: Se explica cómo se podrían utilizar las técnicas de caja blanca y caja negra para validar la correcta implementación de las funcionalidades. Se describe cómo utilizar el método Delphi para valorar la propuesta y las técnicas y métodos para llevar a cabo las pruebas de aceptación.

Capítulo 1: Fundamentación Teórica

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En el presente capítulo se realiza un estudio sobre los conceptos asociados a la investigación como ingeniería y calidad de software, proceso de desarrollo de software, estudio de los sistemas basados en conocimiento y selección de la técnica a utilizar; elementos útiles para una mejor comprensión de la propuesta de solución.Además, se estudian las actividades de desarrollo de software en la UCI y las actividades de calidad asociadas.Se define la metodología, herramientas y lenguajes para el desarrollo de la herramienta a proponer.

1.2. Ingeniería y Calidad de Software

Ingeniería de software

Para llegar a establecer un concepto y enfocar el estudio sobre los procesos de la ingeniería de software, es preciso abordar conceptos de varios autores, sobre este tema.

Para Fritz Bauer la Ingeniería del software es el establecimiento y uso de principios sólidos de ingeniería, orientados a obtener un software económicoque seafiableytrabajedemaneraeficienteenmáquinasreales (13).Según M. Shaw y D. Garlan, es la aplicación de métodos y conocimientocientíficoparacrear solucionesprácticasyrentablesparaeldiseño,construcción, operación y mantenimiento del software y los productos asociados, al servicio de las personas(14).

F. J. García-Peñalvo explica, que la ingeniería del Software no solo se refiere a los procesos técnicos para el desarrollo delsoftware, sinoqueademásincluyelasactividadesdegestióndelproyecto,así como el desarrollo de herramientas, métodos y teorías para el soporte del desarrollo del software(15).

Para lograr el objetivo general es prudente para esta investigación tomar el concepto de ISW según Fritz Bauer,el mismo implícitamente comprendede ingenieríaen las actividades de gestión de un proyecto,durante toda su vida de desarrolloorientados a obtener un software económico, o sea que no exceda el tiempo pactado entre el cliente y el proveedor; fiable significa que sea un producto con garantías de usabilidad,y que trabajedemaneraeficiente, que sea una solución

práctica a un problema real, que sea funcional en concordancia con su objeto social al servicio de las personas. Este concepto está en armonía con elementos que definen la calidad del software.

Procesode software

Es relevante estudiar el proceso de software como elemento base que define las actividades a realizar en la línea de tiempo en el cual es concebido el software, para lo que a continuación algunos autores establecen conceptos:

El proceso se define como el conjunto de actividades, métodos, prácticas y transformaciones usadas para desarrollar y mantener los productos de software y sus productos de trabajo asociados (16).El proceso de desarrollo de software debe tener como propósito fundamental una producción de calidad que reúna los requisitos y satisfaga las necesidades del cliente y/o del usuario al que va dirigido (17).

Según García Peñalvo, el proceso de desarrollo de software suele denominarse *ciclo de vida* del software. Los proyectos softwarese desarrollan en una serie de fases. Van desde la concepción del software y su desarrollo inicial hasta su puesta en funcionamiento y posterior retirada por otra nueva generación de software (18).

Se asume por la correspondencia de los términos empleados el concepto de proceso de desarrollo de software según García Peñalvo, el cual establece que existen fases por el cual transita el proyecto y refleja un hilo conductor del proceso con un inicio, un desarrollo y un cierre de proyecto, así como un posterior mantenimiento o retirada del software.

Fases del proceso de desarrollo de software

Para cumplir el objetivo propuesto es necesario conocer las fases del ciclo de vida del proyecto.En el proceso de construcción de sistemas informáticos se pueden distinguir tres fases genéricas

- ✓ La definición (Análisis de Sistemas; Análisis de Requisitos)
- ✓ El desarrollo (Diseño; Codificación; Prueba)
- ✓ El mantenimiento (Correctivo; Adaptativo; Perfectivo; Preventivo)(18).

Estas fases genéricas abarcan todos los procesos que se realizan en la producción de software independientemente de la metodología que se utilice, divididas en dos grupos, las ágiles y las tradicionales o robustas.

Utilizando las fases de definición, desarrollo y mantenimiento, las cuales pueden ser relacionadas o emparejadas con el inicio, ejecución y cierre de un proyecto, es posible una esquematización para el análisis y diseño de la herramienta que se propone realizar, la cual basada en una información introducida por el usuario sobre en qué fase se encuentra su proyecto, entre otros elementos a introducir, como la clasificación del producto,debe realizar una búsqueda de información de proyectos similares en la misma fase. Identificar las actividades de calidad realizadas y evaluar los elementos negativos y sugerir en cada caso las actividades de calidad a realizar para mitigar posibles resultados negativos en el proyecto en desarrollo.

1.3. Calidad

Es necesario para el contexto que se desarrolla la investigación definir qué es la calidad, y calidad de software, que representa en la industria del software.Calidad: “Grado en que un conjunto de características inherentes de un objeto cumple con los requisitos.” Definiendo al requisito como: “Necesidad o expectativa establecida, generalmente implícita u obligatoria” (2).

Establecer conceptos de calidad para comprender el objetivo que se desea lograr. Calidad eshacer las cosas bien, es igual a cumplir un compromiso, marco de entendimiento. Depende de un marco de referencia establecido para el proyecto (contexto, valores, alcance, objetivos).Calidad en proyectos tecnológicos conviene verla como una decisión y un objetivo. Es la forma de evitar que cada uno aplique sus propios criterios, significa: definir un marco, estándar, norma que establezca cómo medir la calidad, asegurarla y evaluarla(19).

Este último concepto abarca elementos básicos dentro de los proyectos, como el hecho de establecer el alcance y objetivos, y la medición de la calidad, asegurarla y evaluarla, definir un marco de trabajo que permita garantizarla. Asumiendo este concepto, es importante realizar un análisis del cómo y cuándo aplicar estos elementos para lograr un producto correcto tanto para el cliente como para los desarrolladores y que sea eficiente.

Capítulo 1: Fundamentación Teórica

La calidad está altamente relacionada con los defectos en los productos y al alejarse su detección del momento en que se introducen, aumentan los tiempos para su corrección. A pesar de que la tendencia es a desarrollar pruebas, se afirma que es importante invertir en encontrar errores desde el comienzo del proceso de desarrollo (20).A partir de este concepto de Pressman, se responde a la interrogante del cuándo, y es que estimula a que se realice pruebas desde el comienzo del proceso de desarrollo, para evitar que aumenten los tiempos planificados del proyecto, en corregir defectos.

Las tendencias actuales consideran a la calidad como un factor estratégico. Ya no se trata de una actividad inspectora sino preventiva: planificar, diseñar, fijar objetivos, educar e implementar un proceso de mejora continua, la gestión estratégica de la calidad hace de esta una fuente de ventajas competitivas que requiere del esfuerzo colectivo de todas las áreas y miembros de la organización'' (4).

La calidad se planifica, se diseña y se incorpora antes de que comience la ejecución del proyecto.En todo proyecto es sumamente importante dedicar tiempo a la gestión de calidad para principalmente:

- Prevenir errores y defectos.
- Evitar realizar de nuevo el trabajo = ahorrar tiempo y dinero.
- Tener un cliente satisfecho (19).

La calidad con el fin de fortalecer la industria del software enfocando la ingeniería y la calidad de software en la integración y mejora de procesos, denota que la mejora de procesos del software (MPS) se centra en elevar la madurez del proceso de desarrollo y como consecuencia la calidad del producto final" (4). Por tanto, al aplicar los componentes de la calidad como un proceso de mejora continua y centrándose la organización en elevar estos niveles en sus tareas dentro del proyecto de manera preventiva, la organización adquiere un mayor nivel de calidad en sus resultados/productos.

Las organizaciones desean entregar productos y servicios en menos tiempo, más baratos y con una terminación que satisfaga a los usuarios finales. Al mismo tiempo, con la evolución de las

tecnologías, casi todas las organizaciones se han visto abocadas a construir productos y servicios cada vez más complejos, elevar los niveles de reusabilidad a través del desarrollo y adquisición de componentes (4). Una organización con elevada calidad en los productos que provee, le garantiza competitividad (buen posicionamiento) dentro del mercado del software.

Actividades de calidad

Para garantizar la calidad en la industria del software se realizan varias actividades que tienen como objetivo que los productos cumplan con los requisitos que se realicen en tiempo con el costo planificado, que tengan la calidad requerida para el cliente. Según Marín lasactividades de calidad son aquellas acciones que se llevan a cabo para prevenir, controlar y mejorar la calidad de los productos (4). A continuación, se listan las actividades de calidad a raíz del estudio realizado por Marín: "(…) un marco de trabajo para la gestión de las actividades de calidad, que parte de las mejores prácticas propuestas en estándares, guías y modelos internacionales y de las experiencias de las organizaciones."

- ✓ Revisiones
- ✓ Auditorías a la configuración
- ✓ Revisiones entre pares
- ✓ Validaciones
- ✓ Verificaciones
- ✓ Pruebas
- ✓ Revisiones técnicas formales

Para complementar estas actividades se utilizan**técnicas y métodos**,los queresultan efectivos para estas actividades propuestos en los modelos de calidad, se listan a continuación:

- ➢ Listas de chequeo
- ➢ Pruebas de aceptación
- ➢ Entrevistas
- ➢ Revisión documental
- ➢ Prototipado (4).

Capítulo 1: Fundamentación Teórica

En análisis realizado en ¨Marco de trabajo para disminuir los defectos en proyectos de desarrollo de software¨ con expertos a nivel nacional sobre las actividades de calidad desarrolladas en la industria del software cubano, a través de cuestionarios arrojaron que el 100% de los expertos coincidieron en que deben realizarse **revisiones** y **pruebas**, y el 87% que se deben llevar a cabo **revisiones técnicas formales**, siendo estas las actividades de mayor porciento de inclusión por parte de los expertos (4). Este resultado concluyó con la propuesta de utilizar tres actividades de calidad específicamente. El que será tomado por el autor debido a su pertinencia con respecto al problema planteado en la investigación, y en aras de lograr el objetivo general se utilizaran estas 3 actividades como la base para el desarrollo de la solución, las cuales se listan y describen a continuación:

- Revisiones de adherencia a procesos y productos.
- Pruebas.
- Revisiones Técnicas Formales. (4).

De estas actividades es útil que la solución a implementar guarde como artefactos de entrada y salida de datos, elementos como el nombre, el objetivo de la revisión, indicaciones para su ejecución, productos de trabajo, la frecuencia con la que se realiza, y el resultado.

Revisiones de adherencia a procesos y productos

Objetivos: la evaluación de la adecuación a los procesos y productos de trabajo institucionalizados. Además, se realizarán para verificar la correcta transición de una fase a la siguiente. Indicaciones para su ejecución:

- ✓ Entrega de documentación requerida.
- ✓ Disponibilidad del equipo de proyecto para ejecutar la actividad.

Frecuencia: se propone que se realicen al menos una al finalizar cada fase. Productos de trabajo: todos los productos de trabajo que se creen a partir de la utilización de la metodología de desarrollo utilizada.

Productos de trabajo:todos los productos de trabajo que se creen a partir de la utilización de la metodología de desarrollo utilizada.

Resultados esperados: Resolución de los defectos detectados en el tiempo establecido por el equipo revisor (4).

Revisiones Técnicas Formales (RTF)

Objetivos: la revisión técnica de los productos de trabajo que se identifiquen en la institución como entregables al cliente o aquellos solicitados por los clientes o por necesidad de la institución por deficiencias detectadas en análisis realizados.Indicaciones para su ejecución:

- ✓ Entrega de documentación requerida.
- ✓ Disponibilidad de los roles involucrados del equipo de proyecto para ejecutar la actividad.

Frecuencia: se realizarán al culminar cada uno de los productos de trabajo que resulten de interés a la organización.

Productos de trabajo: Cronograma o plan de proyecto, requisitos funcionales, requisitos no funcionales, arquitectura, base de datos y codificación.

Resultados esperados: Resolución de los defectos detectados en el tiempo conciliado con el equipo experto (4).

Pruebas

Objetivos: comprobar que el producto a desarrollar cumple con los requisitos establecidos por el cliente, teniendo en cuenta las recomendaciones de buenas prácticas propuestas por los modelos, estándares y guías más recomendados a nivel inter nacional. Indicaciones para su ejecución:

- ✓ Artefactos a probar terminados.
- ✓ Entrega de documentación requerida.

Estos artefactos pueden ser: requisitos funcionales y no funcionales firmados por el cliente, casos de pruebas, entre otros.

Frecuencia: se realizarán al culminar cada uno de los productos de trabajo que resulten de interés a la organización.

Productos de trabajo: requisitos funcionales, requisitos no funcionales, producto, manual de usuario y manual de instalación.

Resultados esperados: Resolución de los defectos detectados en cada uno de los tipos de pruebas realizadas (4).

Luego de describir las actividades de calidad a utilizar, así como la definición de los elementos de entrada y salida de datos, y basados en las fases del ciclo de vida de un proyecto o en las fases genéricas del proceso de desarrollo de software, se concluye que estas actividades de calidad pueden ejecutarse en cualquier etapa del proyecto. Teniendo en cuenta que siempre se cumplan las indicaciones para su ejecución en cada una de ellas, como primicia para poder ejecutarse.

1.4. Sistemas Basados en Conocimiento (SBC)

Para que el sistema ayude a disminuir el esfuerzo de corrección de defectos en un proyecto en desarrollo, el mismo debe sugerir actividades de calidad en cualquier etapa, apoyándose en experiencias similares anteriores. Para que la solución pueda realizar un análisis y sugerencia por sí solo, es necesario hacer uso de la Inteligencia Artificial (IA).

La IA se refiere a la capacidad de emular las funciones inteligentes del cerebro humano (11), donde ayuda a la toma de decisiones a través de la utilización del conocimiento y la experiencia acumulada, ajustándola a la nueva situación y aprendiendo de cada escenario lo que se considere relevante, retroalimentándose con cada nuevo conocimiento (21).De las técnicas de la IA es más idónea utilizar los sistemas basado en el conocimiento atendiendo en este tipo de circunstancias la necesidad del problema a resolver teniendo en cuenta que, los Sistemas Basados en Conocimiento son programas para computadoras que simulan las cadenas de razonamiento que realiza un experto para resolver un problema de su dominio. Para conseguirlo, se dota al sistema de un conjunto de principios o reglas que infieren nuevas evidencias a partir de la información previamente conocida (21).

Los Sistemas Basado en Conocimiento son programas para computadoras que utilizan conocimiento para un dominio de aplicación determinado para obtener una solución a un problema en este dominio. Son un modelo computacional de más alto nivel que el paradigma de la programación convencional (22).

También entre otras presentan las siguientes propiedades fundamentales, que responden a la necesidad que representa la forma adecuada lograr el objetivo general de esta investigación partiendo de que son:

- ✓ Aplicables a problemas complejos que requieren excesivos recursos computacionales o temporales.
- ✓ Pueden ser utilizados por usuarios no expertos.
- ✓ Recrean el razonamiento humano.
- ✓ Representan el conocimiento humano, el cual, si bien es difícil de caracterizarlo, una vez almacenado y procesado adecuadamente se constituye en algo invaluable(23).

Este tipo de técnica utiliza la madurez de las empresas, los estándares definidos, ejemplos anteriores, resultados obtenidos, experiencias en el campo donde se quiere implantar, entre otros. Este permite el empleo de dicho contenido para dar apoyo a la toma de decisiones en la búsqueda de determinados comportamientos, así como resultados a ocurrir (21).

Los SBC están compuestos por diferentes formas de representar como son las Redes Neuronales Artificial (RNA), los Sistemas Basados en Reglas (SBR) y el Razonamiento Basado en Caso (RBC

Los SBR permiten definir reglas IF-THEN con una serie de ventajas:

- ➢ Las reglas se entienden fácilmente
- ➢ Las reglas son independientes
- ➢ Las reglas encapsulan pequeñas porciones de conocimiento que conjuntamente permiten modelar un problema complejo
- ➢ Las reglas pueden colocarse en cualquier orden en un programa. En estos sistemas el conocimiento se representa como hechos y las reglas permiten manipular los hechos (24).

A pesar de la potencia de estos sistemas tienen ciertas limitaciones:

- ➢ Es extremadamente difícil obtener un conjunto de reglas correcto. Este problema es conocido como el cuello de botella de elicitacion del conocimiento y hace muy difícil la recolección del conocimiento en ciertos dominios.

- Estos sistemas no funcionan adecuadamente cuando se carece de un modelo explícito de conocimiento (24).

Las RNA parte dela adquisición del conocimiento incluye la selección de los ejemplos, el diseño de su topología y el entrenamiento de la red para hallar el conjunto de pesos. Facilitan el trabajo con información incompleta y brindan algoritmos poderosos de aprendizajes para crear la base de conocimiento; pero requieren de muchos ejemplos y son cajas negras que no explican cómo se alcanza la solución. Estas generalmente utilizan pesos como forma de representar el conocimiento y el cálculo de niveles de activación de las neuronas como métodos de solución de problemas.(21).

El razonamiento basado en conocimiento (RBC) es una técnica de Inteligencia Artificial(IA), que conlleva a los SBC, que a diferencia de cualquier programa convencional, este tipo de sistemas hace uso de una base de conocimientos, la cual contiene información almacenada sobre un problema en particular, normalmente proporcionada por un experto y un motor de inferencia que es el encargado de razonar sobre soluciones posibles para el problema planteado a través de búsquedas de información en la base de conocimientos, con lo cual se lograría emitir una conclusión razonable comparable al pensamiento de un humano (23).

El RBC representa un nuevo método para resolver esencialmente problemas no estructurados en el cual el razonamiento se realiza a partir de una memoria asociativa que usa un algoritmo para determinar una medida semejanza entre dos objetos. En este paradigma la base del comportamiento inteligente de un sistema radica en recordar situaciones similares existentes en el pasado (25).

Con el objetivo de lograr disminuir el esfuerzo de corrección de defectos en los proyectos de software es necesario realizar una evaluación inicialmente entre el proyecto en ejecución con respecto a proyectos similares anteriores para utilizar de los mismos las mejores prácticas y que sea posible identificar errores que se cometieron previamente, para que no se vuelva a incidir en ello. De las técnicas de los SBC anteriormente descritas es más acorde a utilizar dada la problemática planteada la técnica Razonamiento Basado en Caso (RBC).

Capítulo 1: Fundamentación Teórica

Razonamiento basado en casos

Dentro de las técnicas de inteligencia artificial, el Razonamiento Basado en Casos (RBC) es un paradigma de solución de problemas que difiere de otros enfoques y técnicas por su capacidad de utilizar el conocimiento específico adquirido en situaciones previas y utilizarlo en la situación presente. El nuevo problema se resuelve buscando en su memoria, un caso similar a este que haya sido resuelto en el pasado. Además, incrementa su conocimiento, almacenando el nuevo caso para ser usado en situaciones futuras(26).

La arquitectura básica de un RBC consiste en una base de casos, un procedimiento para buscar casos similares y un procedimiento de adaptación para ajustar las soluciones de los problemas similares a los requerimientos del nuevo problema (26).

Base de conocimiento

La base de conocimiento contiene la información sobre un determinado problema del cual el sistema es experto, esta base almacena una representación de los conceptos y relaciones de las tareas. Este conocimiento se representa normalmente usando lógica de predicados o lógica proposicional, los elementos que constituyen el conocimiento se clasifican en hechos, heurísticas y reglas (23). También se almacena el conocimiento sobre el dominio de aplicación en forma de estructuras conceptuales, reglas de producción u otra forma de representación del conocimiento (25).

La base de conocimiento para la solución del problema contendrá de cada proyecto de software desarrollado: el nombre, la clasificación del producto, el tiempo de duración del cronograma sin actividades de calidad, tiempo de duración del proyecto, actividades de calidad base, defectos más comunes de ese tipo de producto y recomendaciones de actividades de calidad para mitigar eso defectos. También contendrá los indicadores de calidad usados en la producción de software, siendo los siguientes elementos a tener en cuenta de cada uno: nombre del parámetro, valor del parámetro, descripción del indicador y nombre del indicador. Además, tendrá las medidas para disminuir cada indicador afectado, pruebas de calidad y cantidad de defectos por prueba.

Motor de inferencia

Es la máquina de razonamiento del sistema, la cual compara el problema insertado con los que están almacenados en la base de datos y como resultado infiere una respuesta con el mayor grado de semejanza a la que se busca (21).

Explora la base de conocimiento con el propósito de obtener una salida. En este motor se implementa el método de resolución del problema que, utilizando los hechos, las reglas y las heurísticas, decide las acciones a tomar para llegar a una solución (23).

El motor de inferencia del sistema de RBC para la solución planteada,a partir de la introducción de la descripción de un proyecto, seintroduceen la base de conocimiento en búsqueda de proyectos similares de los cuales obtendrá las actividades de calidad, pruebas de calidad, los indicadores de calidad afectados, las medidas para disminuir cada indicador afectado, y cantidad de defectos por prueba y devuelve un grupo de casos semejantes.Luego de valorar la información infiere una respuesta o un resultado (adapta la solución) compuesta por diferentes acciones a tomar, específicamente recomendaciones de actividades de calidad (definidas anteriormente en ese capítulo) a realizar para mitigar defectos correspondientes a la fase en la que se encuentra el proyecto. Guarda esa solución lo que permite que el sistema se retroalimente y aprenda.

Interfaz

La interfaz de usuario permite la comunicación entre el sistema y el usuario, dando la posibilidad de interactuar con la base de caso, plantear nuevos problemas y consultar los resultados inferidos (21).

El ciclo de vida de un RBC está formado esencialmente por los cuatro procesos siguientes:

- Recuperar el caso o casos pasados más similares, pues retoma la experiencia de un problema anterior que se cree es similar al nuevo.

- Reutilizar la información y conocimiento de este caso o casos recuperados para resolver el nuevo problema. Esto consiste en copiar o integrar la solución del caso o casos recuperados.

- Revisar la solución propuesta.

- Guardar la nueva solución una vez ha sido confirmada o validada. Se guardan aquellas partes de la experiencia de manera tal que sea útil para resolver problemas futuros (26).

En un sistema con RBC se incluirá información acerca de la especificación del problema y atributos del medio que describen el entorno del problema. La descripción del problema debe incluir:

- Las metas que se deben conseguir para resolver el problema.

- Las restricciones de estas metas.

- Las características de la situación del problema y las relaciones entre sus partes.

Otra información de vital importancia que se debe almacenar es la descripción de la solución, que será usado cuando se encuentre una situación similar (26).

Dependiendo de cómo el sistema razone con los casos, esta descripción puede incluir únicamente los hechos que llevan a la solución o información sobre pasos adicionales en el proceso de obtención de la solución. Además, la descripción de la solución también podrá incluir:

- Las justificaciones de las decisiones tomadas en la solución.

- Soluciones alternativas que no fueron elegidas y porqué.

- Soluciones no admisibles y la razón por la que fueron rechazadas.

- Expectativas acerca del resultado de la solución(26).

Es importante incluir una medida del éxito, si en la base de casos se han logrado soluciones con diferentes niveles de éxito o fracaso. También se debe incluir información acerca del resultado.

- Si el resultado cumple o no con las expectativas y una explicación del por qué.

- Si el resultado fue un éxito o un fracaso.

- Estrategia de reparación (26).

El conocimiento que almacena un caso es específico, no abstracto, y todo el conocimiento relacionado (es decir, conocimiento aplicable en una circunstancia concreta) se encuentra cerca en la base de casos, esto quiere decir que el conocimiento necesario para resolver un problema específico se encuentra agrupado en unos pocos casos o incluso en uno.

En los sistemas RBC la base de casos es la memoria de casos almacenados. A la hora de construir la memoria se debe tener en cuenta los siguientes aspectos:

- La estructura y representación de los casos.

- El modelo de memoria usado para organizar los casos.

- Los índices empleados para identificar cada caso (26).

Luego de realizar un análisis sobre los Sistemas Basados en el Conocimiento, resulta el RBC, el de mayor perspectiva a utilizar para la solución del problema planteado, dada sus características como que esté compuesto por una base de casos (proyectos anteriores) y un procedimiento para buscar casos similares. Esencialmente, con estas dos características modela el objetivo de la investigación, concluyendo que cumple con el objetivo buscadoy es la opción a utilizar en la solución propuesta.

1.5. Toma de decisiones

La toma de decisiones es un proceso el cual comienza por una necesidad de información, se recopilan los datos, se analizan y se selecciona un curso de acción entre varias alternativas(27).

En la toma de decisiones existen dos tipos, las decisiones de certidumbre (el decisor cuenta con toda la información sobre la decisión, conoce su resultado final y cómo se va a comportar) y las decisiones de incertidumbre, son aquellas que el decisor toma sin un conocimiento completo de la situación y cada estrategia corresponde a diversos resultados (27). Éste ultimotipo se corresponde con las situaciones que se presentan en el proceso de desarrollo de un proyecto, donde las decisiones se basan en diferentes resultados obtenidos previamente. Entre los métodos que pertenecen a la toma de decisiones en incertidumbre se encuentra el *Método de expertos, Método Delphi, Método de Jerarquías Analíticas (AHP).*

El método de Expertos se hace necesario en el análisis de sistemas complejos en el concurso de expertos, con su poder de análisis, conocimiento, razonamiento, experiencia e intuición, pueden disminuir la complejidad de la problemática a resolver al cuantificar sus criterios en un rango de valores (27).

La prosperidad de una empresa radica básicamente en las decisiones que tome (9). En el proceso de desarrollo de software constantemente se toman decisiones a la hora de realizar cualquier actividad o ejecutar un proceso ingenieril. Es por ello que la solución propuesta debe sugerir un grupo de actividades de calidad a realizar por los miembros del proyecto que la consulten, basado

en los datos obtenidos de las experiencias de proyectos anteriores, similares al que será insertado inicialmente que se encuentra en ejecución, con el objetivo de mejorar los resultados.

Las actividades de calidad a sugerir forman parte del resultado obtenido del concurso de un experto, en este caso por la MSc. Aymara Marín Díaz, la cual es autora de "Marco de trabajo para gestionar actividades de calidad en los proyectos de desarrollo de software", e*n el marco de trabajo se proponen: actividades de calidad, técnicas, y métodos para llevarlas a cabo (4)*, la base de casos contendrá las actividades propuestas por la autora, y las sugerencias estarán en correspondencia con lo planteado por la misma en el marco de trabajo.

1.6. Metodologías para el desarrollo de software

Las metodologías para el desarrollo de software se basan en la utilización de modelos, técnicas y herramientas utilizados durante el ciclo de vida de un software. Las metodologías ágiles, están más orientadas a la generación de código con ciclos muy cortos de desarrollo, se dirigen a equipos de desarrollo pequeños, hacen especial hincapié en aspectos humanos asociados al trabajo en equipo e involucran activamente al cliente en el proceso(28).

Las metodologías ágiles presentana partir del concepto anterior varias características que se semejan alproyecto a ejecutar para la solución del problema planteado,ejemplo, el equipo de desarrollo es pequeño (una sola persona); la comunicación y retroalimentaciónes constante con el cliente. Por tanto, es lógico seleccionar una metodología ágil para desarrollar el proyecto.

La metodología que se propone para el desarrollo delsistema es la AUP (Proceso Unificado Ágil),pero en su adaptación a los procesos de desarrollo de software de la UCI. A continuación, aspectos fundamentales de la metodología.

AUP-UCI

El ciclo de vida de los proyectos se recrea en tres fases: fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes tres fases de AUP en una sola, a la que llamaremos Ejecución y se agrega la fase de Cierre(29). A continuación, se describe la fase de Ejecución:

*Fase de Ejecución:*En esta fase de la AUP-UCI se definen: elaboración, construcción y transición. Se ejecutan las actividades requeridas para desarrollar la herramienta, incluyendo el ajuste de los

planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, se obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto (29). En esta fase estará comprendido el desarrollo del sistema como solución al problema de la investigación planteado.

Disciplinasde AUP-UCI

Las disciplinas definidas que utiliza esta metodología se desarrollan en la etapa de Ejecución de la metodología. Donde se realizan iteraciones y se obtienen resultados incrementales. Las disciplinas a implementar serán: Requisitos, Análisis y diseño, Implementación, Pruebas internas y Pruebas de aceptación (9).

Escenarios de la metodología

En la metodología AUP-UCI están definidos 4 escenarios para modelar el negocio, donde cada escenario está diseñado para los 4 casos básicos en los que puede encontrarse cualquier proyecto que se desee desarrollar. El escenario donde se desarrollará el proyecto es el número 4, el mismo aplica para los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio bien definido. El cliente estará acompañando el desarrollo del proyecto para convenir los detalles de los requisitos, así como poder probarlos y validarlos. Este escenario es utilizado en proyectos que no modelen negocio donde solo se pueden modelar los requisitos con Historias de Usuarios(29).

1.7. Herramientas y lenguajes para el desarrollo de software

Lenguaje de ModeladoUML 2.0

UML (UnifiedModelingLanguage) o Lenguaje de Modelación Unificado es un lenguaje gráfico para detallar, construir, visualizar y documentar las partes o artefactos en el proceso de desarrollo de software (30). UML es un lenguaje visual de modelado para visualizar, especificar, construir, y documentar los artefactos en un sistema de software, es el lenguaje que estandariza la forma de crear diagramas, el significado preciso, y las relaciones existentes entre ellos (31). Los diagramas

de diseño de clases, componente y despliegue que propone la metodología a utilizar, serán realizados utilizando este lenguaje.

Visual Paradigmfor UML 8.0

Visual Paradigm es una herramienta CASE para modelamiento UML muy potente, gratuita, fácil de instalar, utilizar y actualizar. Te permite dibujar todo tipo de diagramas UML, revertir código fuente a modelos UML, generar código fuente desde los diagramas UML, y mucho más.Incluye los objetos de UML además de diagramas de casos de uso, diagramas de clase, diagramas de componentes, entre otros. Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software (32).El modelamiento de los diagramas para el desarrollo del software de la propuesta de solución se realizarán usando esta herramienta.

Lenguajes de programaciónutilizados

HTML 5

Se utiliza el lenguaje de programación web HTML5, el cual propone estándares para cada elemento de la web y también un propósito claro para cada una de las tecnologías involucradas. Tiene los elementos necesarios para ubicar contenido estático o dinámico, además, provee los elementos estructurales, CSS se encuentra involucrado en cómo volver toda esa estructura utilizable y atractiva a la vista, y JavaScript tiene todo el poder necesario para proveer dinamismo, seguridad y construir aplicaciones web completamente funcionales (33).

CSS 3

Para crear los estilos web se utiliza CSS que es un lenguaje que trabaja con HTML para proveer estilos visuales a los elementos del documento como el tamaño, color, borde, fondo, entre otros. Es la forma de separar la estructura de la presentación. CSS ha crecido y ganado importancia, pero siempre desarrollado en paralelo, enfocado en las necesidades de los diseñadores y apartado del proceso de evolución de HTML(33).

JavaScript 1.7

JavaScript es un leguaje interpretado usado para múltiples propósitos. Posee motores de interpretación, creados para acelerar el procesamiento del código (33). Es un lenguaje compacto, y

basado en objetos, diseñado para el desarrollo de aplicaciones cliente-servidor a través de internet. Tiene la particularidad que se encuentra insertado dentro del mismo documento HTML (34).

PHP 7

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Entre sus características están la inclusión de la visibilidad, las clases abstractas, clases y métodos finales, manejo de excepciones, interfaces, clonación y tipos sugeridos. Forma parte de las tecnologías LAMP (Linux, Apache, MySQL, PHP/Python/Perl), siendo esta plataforma una de las más extendidas en cuanto software open source para desarrollo web se refiere (35). Se utilizó este lenguaje en conjunto con los antes mencionados e integrándose con el marco de trabajo Symfony para crear la propuesta de solución.

Marco de trabajo para desarrollar la propuesta de solución

Symfony 3

Es un framework para PHP para el desarrollo de aplicaciones web de forma rápida, sencilla, y está diseñado para optimizar el desarrollo de aplicaciones web a través de diversas características clave. Separa las reglas de negocio de la aplicación, la lógica del servidor y las vistas de presentación. Admite la utilización de los mejores componentes actualmente disponibles (36).

Características:

- Se basa en el patrón MVC. Symfony guarda todas las clases y archivos relacionados con el modelo. La vista está formada por plantillas que se guardan en varios directorios repartidos por todo el proyecto.
- Al controlador pertenecen los frontales quienes realmente delegan todo el trabajo en las acciones. Las agrupaciones lógicas de acciones se denominan módulos.
- URLs inteligentes: que permiten que las direcciones de las páginas web sean parte de la interfaz y resulten amigables a los motores de búsqueda.
- Una gestión de listas más amigables al usuario: gracias a la paginación automática, ordenamiento y filtrado de resultados.
- Es multiplataforma

- Fácil de instalar y configurar: ha sido probado con éxito en plataformas Windows y derivadas de Unix.(36)

Se utiliza el frameworkSymfony debido a que permite la arquitectura de tipo Modelo-Vista-Controlador para el desarrollo de aplicaciones web. Proporciona elementos para la optimización del desarrollo web lo que permite reducir el tiempo de desarrollo de una aplicación web compleja. Su característica multiplataforma y facilidad de instalación y configuración, son ventajas necesarias y útiles entre otras, por los que se selecciona este framework para desarrollar la solución.

Gestión y administración de bases de datos

PostgreSQL 9.5

PostgreSQL es un gestor de bases de datos relacionales organizado en diferentes niveles. El puerto por defecto es5432 y es el mismo que se asume para la propuesta de solución. Una base de datos (database) es una colección de esquemas (schema), quienes,a su vez, finalmente contienen los objetos que se conocen habitualmente en el contexto de técnicas de almacenamiento de datos, tales como tablas (table), vistas(view), secuencias(sequence) (37).

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando(38).

PgAdmin III

PgAdmin III es una plataforma de administración y desarrollo Open Source. La aplicación puede utilizarse en plataformas Linux, FreeBSD, Solaris, MacOS y Windows. PgAdmin está diseñado para responder a las necesidades de los usuarios, desde escribir consultas SQL sencillas hasta desarrollar bases de datos complejas. La interfaz gráfica puede ejecutarse en el escritorio o en un servidor web. La aplicación incluye una sintaxis que resalta el editor de SQL (39).PgAdmin es la interfaz gráfica que permite la administración de la base de datos en este caso vinculada con PostgreSQL9.4 que es el gestor de bases de datos que se utiliza en la propuesta de solución.

Entorno de Desarrollo Integrado (IDE)

PhpStorm 2017.1.5

Se utiliza el PhpStorm como un Entorno de Desarrollo Integrado (IDE)inteligente para desarrollar aplicaciones en Pre-procesador de hipertextos (PHP), proporcionandoherramientas esenciales como refactorización, análisis de código y comprobación de errores.Las principales novedades en PhpStorm 2017.1.5 incluyen:

- Soporte para PHP 5.3 en adelante.
- Depuración con cero configuraciones con todos los navegadores.
- Editores de Lenguaje de Consulta Estructurado (SQL) con resultados editables.
- Soporte para Lenguaje Marcado de Hipertextos (HTML5) (40).

Servidor de Aplicaciones

Apache 2.5

El servidor web Apache se basa en un concepto muy simple de división del sistema en módulos, representando estos la división del trabajo dentro del sistema, valga la redundancia. Este servidor tiene un Núcleo (core) que lleva a cabo todas las funcionalidades importantes del sistema, y los módulos, externos a este núcleo, añaden funcionalidades al servidor web, configurándose cada uno de ellos por separado. Se obtiene globalmente un sistema muy flexible, que permite ser configurado y optimizado acorde a necesidades mediante la implementación de módulos externos (41).

1.8. Conclusiones parciales

El desarrollo del marco teórico de la investigación a partir de los conceptos y elementos definidos permitió realizar una caracterización dela herramienta que se propone realizar.Se seleccionaron los lenguajes, herramientas y metodología con los que se le dará solución el problema planteado en la presente investigación.

Capítulo 2: Desarrollo de la solución

CAPÍTULO 2: DESARROLLO DE LA SOLUCIÓN

2.1. Introducción

En este capítulo se realiza una descripción de los elementos fundamentales del sistema, con el objetivo de tener una visión acerca del producto final. Partiendo de la metodología definida se describen los procesos realizados durante las disciplinas de Requisitos, Análisis y Diseño e Implementación.

2.2. Necesidades del negocio

La necesidad de negocio de una organización puede deberse a una necesidad de formación, a una demanda del mercado, a un avance tecnológico, a un requisito legal o a una norma gubernamental (42).La Dirección de Calidad de Software de la Universidad de Ciencias Informáticas, necesita disminuir el esfuerzo de corrección de defectos en los proyectos en desarrollo, basándose en experiencias de casos anteriores proponga actividades de calidad a realizar en el proyecto durante su ejecución.

Las actividades de calidad sugeridas tienen como objetivo evitar el re-trabajo al equipo de desarrollo. Por tanto,tomando como guíael concepto anteriormente mencionado, y en correspondencia con la necesidad del cliente se puede catalogar a la propuesta de solución como un avance tecnológico.

2.3. Requisitos

Los requisitos son quizás la base más importante en la construcción de productos de software porque a través de éstos se puede lograr un entendimiento común entre las partes interesadas sobre el sistema que se va a implementar (43).Las coordinaciones necesarias entre las partes en cuanto al sistema son a través de la definiciónde los requisitos, logrando así un mejor entendimiento para el diseño e implementación del mismo.

Técnicas para la obtención de requisitos de software

Para realizar la búsqueda de los requisitos fue necesario la utilización de técnicas para la obtención o levantamiento de requisitos. Las técnicas utilizadas son:

Capítulo 2: Desarrollo de la solución

Entrevista: Se llevaron a cabo reuniones y entrevistas con el cliente, en este caso, la DrC. YaimíTrujillo Casañola y la MsC. Aymara Marín Días, pertenecientes a la Dirección de Calidad de Software de la Universidad, quienes tienen una vastaexperiencia laboral en la gestión de calidad en proyectos de desarrollo de software.

Tormenta de ideas: En cada encuentro se debatieron los requisitos identificados, obteniendo como resultado una visión más amplia de las funcionalidades a implementar, favoreciéndose de esta forma el avance en el desarrollo del sistema.

Requisitos Funcionales:

Los requisitos funcionales corresponden a las características deseadas de un sistema (44). Describiendo los pasos a seguir, el qué y el cómo deben ser las funciones de un sistema, dependiendo del tipo de software y el enfoque y objetivos que adopta la organización.

Los requisitos funcionales del sistema se muestran a continuación:

RF1: Insertar Proyecto.

RF2: Listar Proyectos.

RF3: Ver datos de proyecto. Mostrando: Clasificación de producto, tiempo de duración, actividades de calidad base, defectos más comunes a partir de ese tipo de producto, recomendaciones actividades de calidad para mitigar esos defectos.

RF4: Proponer actividades de calidad y defectos potenciales.

RF5: Registrar resultados de las pruebas.

RF6: Recomendar actividades para contención de fase (Inteligencia Artificial).

RF7: Identificar indicadores de calidad afectados.

RF8: Mostrar resultados.

RF9: Registrar actividad.

RF10: Determinar estado de los indicadores de calidad. Insertar los indicadores de costos de calidad y determinar si son adecuados o no.

RF11: Adicionar parámetro.

Descripción del requisito funcional: Insertar Proyecto.

A continuación, se describe mediante una historia de usuario el requisito:

Numero: 1	**Nombre del Requisito:**Insertar Proyecto.
Programador: Noel H. Hidalgo Reyes	**Iteración Asignada:**2
Prioridad: Alta	**Tiempo Estimado:**3 días
Riesgo en Desarrollo:	**Tiempo Real:** 60 horas
Descripción:	
Prototipo de Interfaz: 	

Figura 1: Historia de usuario de la funcionalidad Insertar proyecto

Requisitos No Funcionales

Los requisitos no funcionales pueden estar caracterizados por diferentes factores tales como pueden ser los atributos de calidad, de seguridad, de disponibilidad, restricciones del sistema,

metas, características de usabilidad, etc. (44). Los requisitos no funcionales no están directamente relacionados con los usuarios, ellos corresponden a la parte emergentes como, fiabilidad, usabilidad, rendimiento, seguridad y restringir características del sistema.

RNF1 Usabilidad

RNF1.1 Inteligibilidad: El idioma de todas las interfaces de la aplicación será español.

RNF1.2 Operatividad:

i. Los errores cometidos por el usuario les serán notificados.

ii. El sistema expondrá el menú general desde cualquiera de sus páginas.

RNF1.3 Facilidad de aprendizaje:

i. Los botones siempre aparecen del lado derecho de la interfaz.

ii. La forma de llenar los formularios es de arriba hacia abajo.

iii. Los menús aparecen en el orden en que se lleva a cabo el negocio.

RNF1.4 Capacidad de atracción:

i. El sistema ofrece una interfaz fácil de operar para el cliente.

ii. Diseño sencillo, con pocas entradas, permitiendo que no sea necesario mucho entrenamiento para que los usuarios puedan utilizar el módulo (42).

RNF2 Fiabilidad:

i. Ante el fallo de una funcionalidad del sistema, el resto de las funcionalidades que no dependen de esta deberán seguir funcionando.

ii. El sistema impondrá campos obligatorios para garantizar la integridad de la información que se introduce por el usuario y no permitirá la entrada de datos incorrectos (42).

RNF3 Mantenibilidad

i. La codificación del sistema será estándar y las funcionalidades serán comentadas. Además, el sistema debe permitir adicionar o modificar funcionalidades (42).

RNF4 Portabilidad

RNF4.1 Adaptabilidad: El sistema deberá adaptarse al entorno operativo siempre y cuando este cumpla con las características de la aplicación y debe coexistir con otros sistemas sin dificultad.

RNF5 Rendimiento

RNF5.1: Comportamiento en el tiempo: El sistema en las operaciones de modificación o búsqueda de datos, deberá tener respuestas a peticiones del cliente en un tiempo máximo de 5 segundos (esta cifra no incluye los retardos por concepto de tráfico de red) (42).

RNF6 Hardware

RNF6.1: Estaciones de trabajo: Las estaciones de trabajo deberán poseer como mínimo un procesador Pentium IV a 2.20 GHZ, una RAM de 1GB y una tarjeta de red.

RNF6.2: Servidor de aplicaciones: El servidor de aplicaciones deberá poseer como mínimo un Core 2 Duo a 2.33 GHz, una RAM de 4 GB, disco duro 250 GB, una tarjeta de red y una fuente de alimentación de corriente.

RNF7 Software

RNF7.1: Navegador web: Las estaciones de trabajo deberán poseer el navegador web Mozilla Firefox 34.0 o una versión superior y el sistema operativo Linux/Windows.

RNF7.2: Servidor de aplicaciones web: El servidor de aplicaciones web deberá ser Apache 2.0 y el sistema operativo Linux.

Técnicas de validación de requisitos

Se define a las técnicas de validación de requisitos como el procedimiento que permite a los usuarios detectar y resolver conflictos entre requisitos, verificando cada uno de los criterios de validación (no ambigüedad, consistencia, concisión, verificabilidad, trazabilidad, fiabilidad, entre otros), algunas de estas técnicas pueden utilizarse en forma individual o combinada (45).

Capítulo 2: Desarrollo de la solución

Existen varias técnicas para la validación de requisitos de software como: Técnica de prototipado y Diseño de casos de prueba que son descritas a continuación.

Prototipado: el uso de prototipos es considerado como una de las técnicas más eficientes del levantamiento de requisitos, en tanto que este método es útil cuando existen dudas acerca de los objetivos de una aplicación y cuando las opiniones del usuario son necesarias en una fase temprana del proceso de especificación (46).

Diseño de casos de prueba (DCP): el objetivo principal del diseño de casos de prueba es obtener un conjunto de pruebas que tengan la mayor probabilidad de descubrir los defectos del software (47).

Se comprobó la coherencia de las necesidades expuestas al validarse con el cliente los diseños de casos de prueba y los prototipos de interfaz de usuario, permitiéndole al cliente una mejor visión de la propuesta de solución aplicando las técnicas mencionadas para la validación de los requisitos. Para así permitir la realización de correcciones durante el proceso de análisis, diseño e implementación de las funcionalidades del sistema.

2.4. Diseño del sistema

Luego de especificados los requisitos funcionales y no funcionales del sistema que llevara por nombre Disminución del Esfuerzo de Corrección en Proyectos en Desarrollo (**DECProD**) se procede a realizar el diseño del mismo. El diseño debe corresponderse con los requisitos funcionales y no funcionales descritos anteriormente.

Arquitectura del sistema

EL modelo de arquitectura a utilizar en el desarrollo del sistema es el modelo-vista-controlador (MVC).

Es una arquitectura de software que separa los componentes internos que interactúan en una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. Se trata de un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones y sobre varios de lenguajes y plataformas de desarrollo de aplicaciones web (48).

Vista:Compone la información que se envía al cliente y los mecanismos interacción con éste (48),presentando el modelo en un formato adecuado para la interacción.

Modelo: Contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia (48), enviando a la vista parte de información que en cada momento se solicita para mostrar.

Controlador: 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 (49), cuando se hace alguna solicitud sobre información este responde a eventos y permite peticiones al modelo.

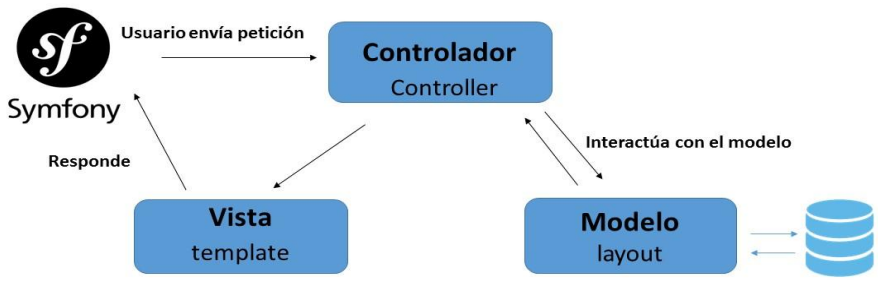


Figura 2: Patrón MVC en el sistema DECProD.

En la Figura 2 se observa la utilización del patrón MVC,según el marco de trabajo Symfony 3, ejemplo las clases controladoras, modelo y es representada las vista ya desde la apreciación hacia el usuario como interfaz.

Capítulo 2: Desarrollo de la solución

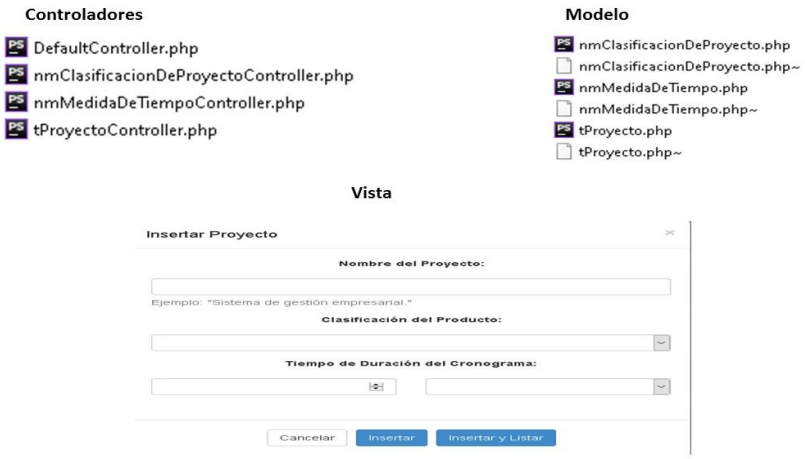


Figura 3: Aplicación del patrón MVC en el sistema DECProD. Fuente: elaboración propia.

Diagrama de clases del diseño

Un diagrama de clase cuyo pilar se centra en el Lenguaje Unificado de Modelado (UML, por sus siglas en inglés), es la representación del sistema y la relación que existe entre el, tal como herencia, agregación, asociación y composición (49).

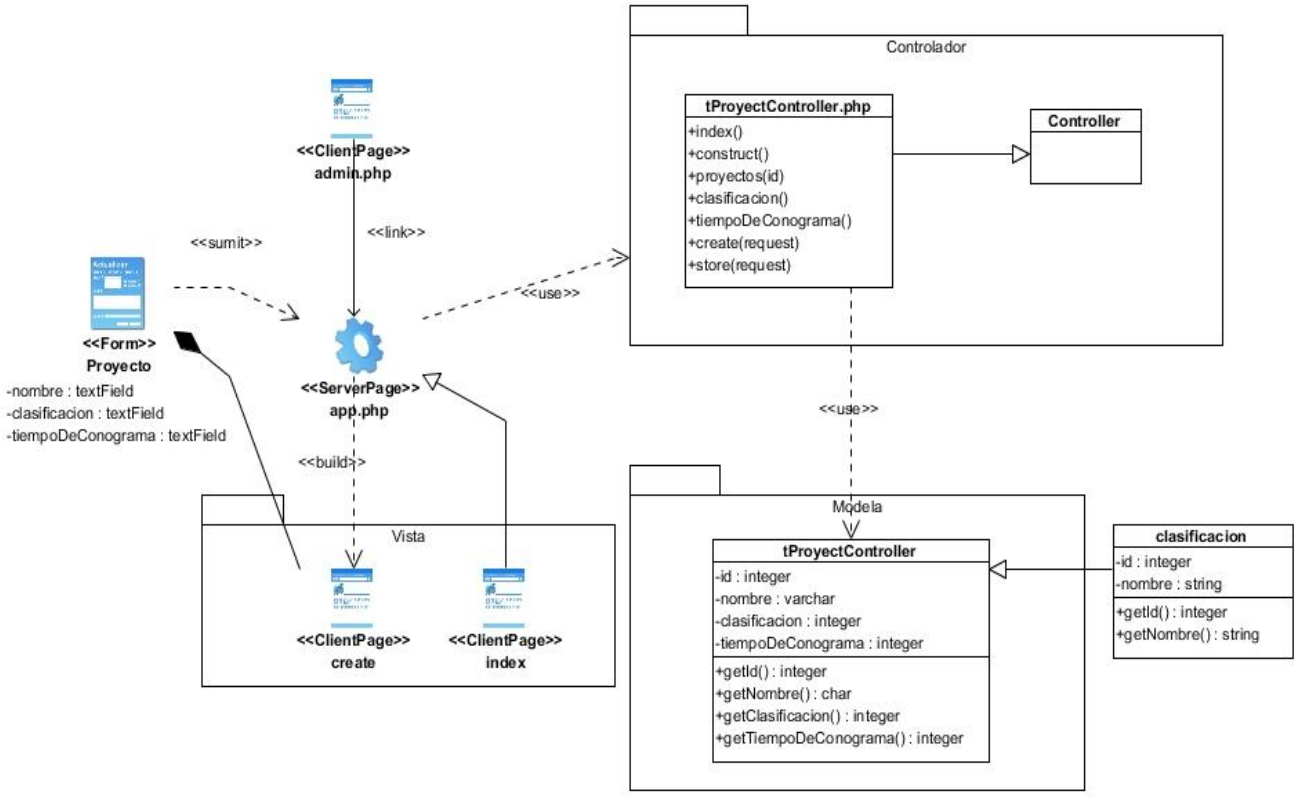


Figura 4: Diagrama de clases del diseño

Patrones de diseño de software

Los patrones de diseño para el desarrollo de software constituyen uno de los avances más utilizados en la tecnología orientada a objetos. Los patrones cumplen la función de capturar, recopilar y transmitir la experiencia del diseñador. El concepto de patrones, como técnica para el desarrollo, viene a encapsular de alguna forma la experiencia y conocimiento adquirido con el correr de los años, de forma que pueda ser transmitido, y se logre con ellos un lenguaje común dentro de la disciplina (50).En el diseño del sistema se utilizaron los patrones GRASP.

Experto: asigna una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

Creador: guía la asignación de responsabilidades relacionadas con la creación de objetos.

Bajo acoplamiento: medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo acoplamiento no depende de muchas otras.

Alta cohesión: medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.

Controlador: objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define el método de su operación (51).

Figura 5: Patrón GRASP aplicado a la clase de diseño

Modelo de datos

El modelo de datos está basado en una percepción del mundo real consistente en objetos básicos llamados entidades y de relaciones entre estos objetos. Es uno de los diferentes modelos de datos semánticos, pues representa el significado de los datos (51).

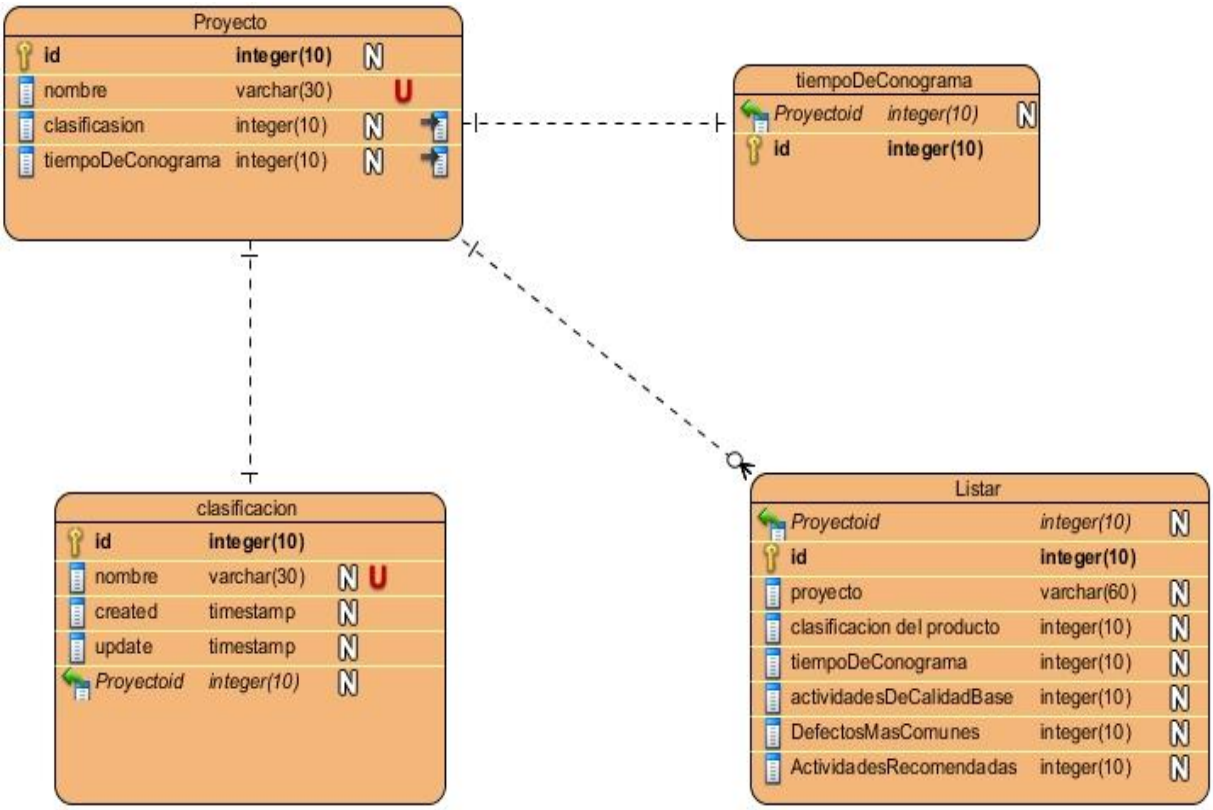


Figura 6: Modelo de datos de la solución propuesta. Fuente: elaboración propia

2.5. Implementación del sistema

Diagrama de componentes

Es una parte física de un sistema (módulo, base de datos, programa ejecutable, etc.).Se puede decir que un componente es la materialización de una o más clases, porque una abstracción con atributos y métodos pueden ser implementados en los componentes. Se representa como un rectángulo en el que se escribe su nombre y en él se muestran dos pequeños rectángulos al lado izquierdo. Los componentes se pueden agrupar en paquetes, así como los objetos en clases, además puede haber de ellos relaciones de dependencia como generalización, asociación, agregación, realización (52).

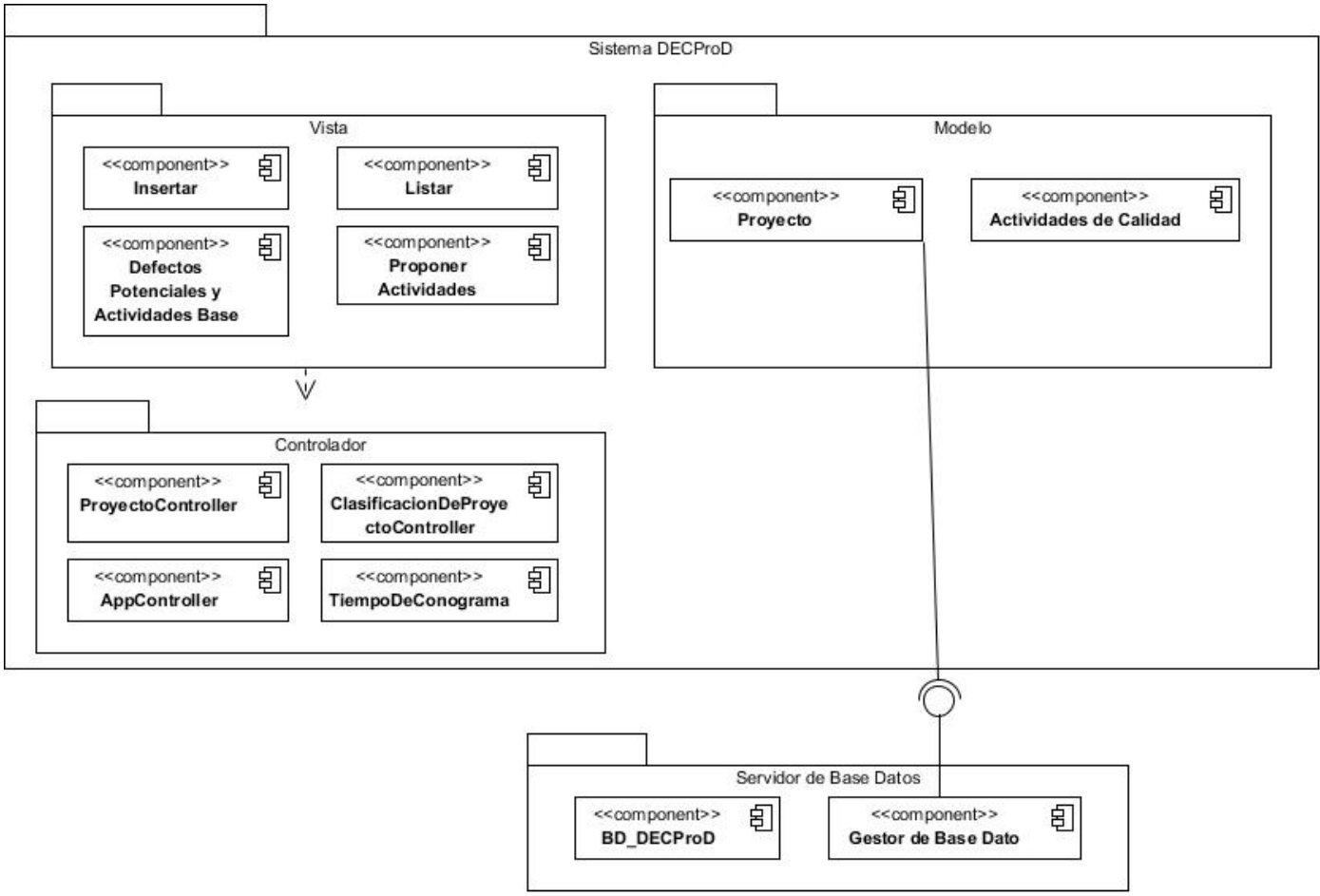


Figura 7: Diagrama de Componentes

Diagrama de despliegue

Los diagramas de despliegue describen la topología física del sistema: la estructura de las unidades hardware y el software que se ejecuta en cada unidad (53).

Capítulo 2: Desarrollo de la solución

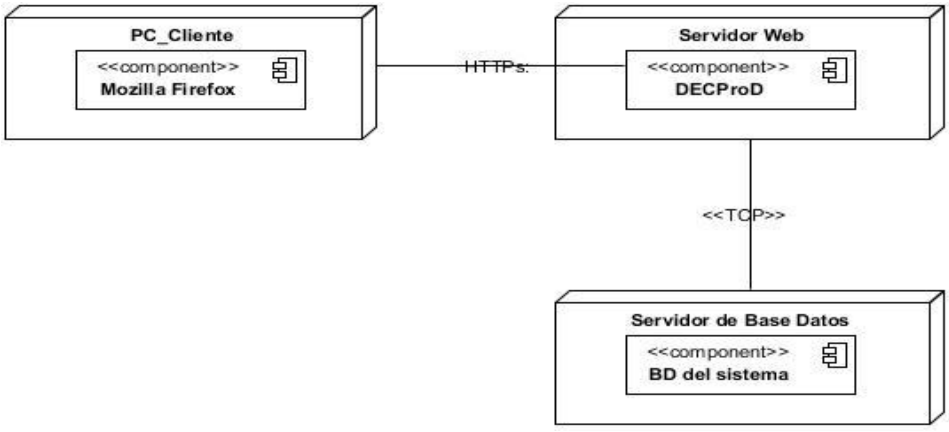


Figura 8: : Diagrama de Despliegue

Estándares de codificación

Los estándares de codificación se pueden definir como “un conjunto de reglas, normas o patrones destinados a establecer uniformidad en el proceso de generación de un código. Son pautas de programación enfocadas a la estructura y apariencia física del código para facilitar su lectura, comprensión y mantenimiento”

Notación de camello (del inglés *Camel Case*): este estilo de notación se emplea para escribir la primera letra de la identificación con minúsculas, y la inicial de cada una de las palabras concatenadas se escribe con mayúscula. Existen dos tipos de *Camel Case*, el estilo *UpperCamel Case*, define que la primera letra de cada una de las palabras que identifican a un elemento de un proyecto es mayúscula y el *LowerCamel Case* cuando la primera letra es minúscula. El estilo de notación utilizado en la declaración de variables y métodos es LowerCamel Case.

- Los nombres de las clases controladoras terminan con la palabra Controller.
- Las variables que poseen nombres de palabras compuestas se escriben juntas.
- Se hizo uso de los comentarios para la explicación del código siempre que fue necesario (9).

Capítulo 2: Desarrollo de la solución

Descripción de la implementación

2.6. Conclusiones parciales

En este capítulo se presentó las necesidades del negocio y en correspondencia se especificaron los requisitos funcionales y no funcionales del software que debe cumplir el proyecto. Se definió la arquitectura, diseño y modelación, donde se obtuvo como resultado un diseño sencillo y acorde a las necesidades. Además, se seleccionaronlas herramientas a trabajar en el proyecto tanto para la modelación como para la implementación, realizándose todos los procesos con éxito.

Capítulo 3: Validación de la solución

CAPÍTULO 3: VALIDACIÓN DE LA SOLUCIÓN

3.1. Introducción

En el presente capítulo se describe cómo realizar la validación de la solución propuesta. Se explica la forma deaplicar las técnicas de caja blanca y caja negra para validar la correcta implementación de las funcionalidades. Se describe cómo utilizar el método Delphi para valorar la propuesta y las técnicas y métodos para llevar a cabo las pruebas de aceptación.

3.2. Validación del software

La verificación y la validación es el nombre que se le da a los procesos de comprobación y análisis que aseguran que el software que se desarrolla esta acorde a su especificación y cumple con las necesidades de los clientes (54)

“Verificación: Proceso de evaluación de un sistema o componente para determinar si un producto de una determinada fase de desarrollo satisface las condiciones impuestas al inicio de la fase”. “Validación: Proceso de evaluación de un sistema o componente durante o al final del proceso de desarrollo para determinar cuándo se satisfacen los requerimientos especificados” (55).

Se debe ejecutar un grupo de pruebas para realizar la verificación y validación de los requisitos tantos funcionales como no funcionales, con el objetivo de encontrar errores o fallos en el sistema. Las pruebas a realizar serán las pruebas de caja blanca para la validación del código, las pruebas de caja negra serán aplicadas a los requisitos funcionales sobre la interfaz de software y sus respuestas en cada posible suceso. A continuación, se describen ambos tipos de pruebas.

Pruebas de caja blanca

Las pruebas de caja blanca se basan en el diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivarlos. Mediante la prueba de la caja blanca el ingeniero del software puede obtener casos de prueba que:

- Garanticen que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo, programa o método.
- Ejerciten todas las decisiones lógicas en las vertientes verdadera y falsa.

- Ejecuten todos los bucles en sus límites operacionales.
- Ejerciten las estructuras internas de datos para asegurar su validez (56).

Técnica camino básico

La prueba del camino básico es una técnica de prueba de la Caja Blanca propuesta por Tom McCabe. Esta técnica permite obtener una medida de la complejidad lógica de un diseño y usar esta medida como guía para la definición de un conjunto básico. La idea es derivar casos de prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control. Para obtener dicho conjunto de caminos independientes se construye el Grafo de Flujo asociado y se calcula su complejidad ciclomática. Los pasos que se siguen para aplicar esta técnica son:

- A partir del diseño o del código fuente, se dibuja el grafo de flujo asociado.
- Se calcula la complejidad ciclomática del grafo.
- Se determina un conjunto básico de caminos independientes.
- Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico(56).

Pruebas de caja negra

Las pruebas de Caja Negra o pruebas Funcionales se realizan sobre la interfaz del software, comprobando las entradas y salidas de datos.

Estas pruebas se realizan para comprobar que cada función es operativa, utilizando el artefacto **diseño de caso de prueba,** que tienen como objetivo introducir juegos de datos que ayuden a la ejecución de los casos y facilite que el sistema se ejecute en todas sus variantes (57).

Método de partición de equivalencia

Éste método divide el campo de entrada en clases de datos de los que se pueden derivar casos de pruebas. La partición equivalente se dirige a una definición de casos de prueba que

descubran clases de errores, reduciendo así el número total de casos de usos de prueba que hay que desarrollar.

En palabras más simple, el método divide el dominio de los valores de entrada en un número finito de clases de equivalencia. De esta manera se puede asumir que una prueba realizada con un valor representativo de cada clase es equivalente a una prueba realizada con cualquier otro valor de la misma clase (54).

“Puede utilizarse paralograr objetivos de cobertura de entrada y salida, con entradas humanas, vía interfaces a un sistema, o parámetros de interfaz de las pruebas de integración”(55).

Descripción del caso de prueba*Insertar Proyecto*.

Condiciones de ejecución

Nombre de la sección.	**Escenario de la sección**	**Descripción de la funcionalidad**	**Flujo central**
SC1: Insertar Proyecto	EC 1.1 :Insertar Proyectos correctamente	El sistemadebe mostrar un interfaz que posibilita insertar los datos de un proyecto.	1. Se muestra la interfaz Insertar Proyecto con los campos a insertar (nombre), (clasificación del producto), (tiempo de duración del cronograma sin actividades de calidad). 2. Se inserta el dato solicitado 3. Se oprime el botón Insertar. 4. El sistema

			comprueba que l dato sea correcto, y muestra el menaje “Proyecto Insertado correctamente”.
	EC 1.2: Insertar proyecto dejando campo vacíos.	El sistema debe mostrar un mensaje informando que existen campos vacíos.	1. Se muestra la interfaz Insertar Proyecto con los campos a insertar (nombre), (clasificación del producto), (tiempo de duración del cronograma sin actividades de calidad). 2. Se inserta el dato solicitado 3. Se oprime el botón Insertar. 4. El sistema muestra un mensaje indicando que el campo esta vacío.
	EC 1.3: Insertar Proyecto insertando datos incorrectos.	El sistema debe mostrar un mensaje indicando que el valor insertado no es correcto.	1. Se muestra la interfaz Insertar Proyecto con los campos a insertar (nombre), (clasificación del producto), (tiempo de

			duración del cronograma sin actividades de calidad). 2. Se inserta el dato incorrecto. 3. Se oprime el botón Insertar. 4. El sistema muestra un mensaje indicando que el campo tiene un valor incorrecto.

Tabla 1: Diseño de caso de prueba de la funcionalidad Insertar Proyecto

Método Delphi

El método Delphi, es unprocedimiento de consulta a expertos que permite recoger opinionesgrupales, consensuadas y fidedignas. Se recurre a este método cuando los criteriosdeevaluación necesitan ser clarificados, por tratarse de un campo novedososobre el que no existen suficientes datos previos que permitan perfilar lainterpretación del objeto de estudio (58).

Este método presenta ventajas pues, se logra obtener una conformidad del resultado en el desarrollo de los cuestionarios sucesivos, o sea un mayor consenso. También desde el punto de vista de que todo esto se hace sin interacción entre los expertos y con fundamentos estadísticos se puede decir que logra recoger información generalmente rica y abundante sobre el problema estudiado y puede ser utilizado en varios campos de acción. Los resultados son cualitativamente apreciables (59).

Capítulo 3: Validación de la solución

El método Delphi, tiene un amplio uso en varias áreas del conocimiento a nivel internacional y a nivel nacional ha sido empleado fundamentalmente en investigaciones educativas y médicas, aunque en los últimos años se ha empleado en investigaciones de Ciencias Informáticas con el objetivo de socializar, externalizar y combinar el conocimiento de los expertos. El método aprovecha los elementos comunes en el grupo de expertos, preserva el anonimato mediante el uso de flujos de comunicación y permite la participación de expertos que se encuentren geográficamente dispersos(60). Es un método que va a contribuir a validar la propuesta de solución, basados en el criterio de expertos y utilizando sus conocimientos para obtener resultados fidedignos basados en fundamentos estadísticos.

Pruebas de aceptación

Estas pruebas son generalmente desarrolladas y ejecutadas por el cliente o un especialista de la aplicación y es conducida a determinar cómo el sistema satisface sus criterios de aceptación validando los requisitos que han sido relevados para el desarrollo, incluyendo la documentación y proceso de negocio.

Evalúan la predisposición del sistema para el despliegue y uso, aunque no es necesariamente el nivel final de la prueba(54).

“Es el proceso decomparar el programa contra sus requerimientos iniciales y las necesidades reales de los usuarios. Realizado generalmente por el cliente o el usuario final”(55).

3.3. Conclusiones Parciales

En el presente capítulo se realizó un plan para ejecutar las pruebas de validación del sistema para la Disminución del Esfuerzo de Corrección de Defectos en Proyectos en Desarrollo (**DECProD**). Para validar el sistema se describieron pruebas, específicamente la técnica del Camino básico con la cual se podrá comprobar que el flujo de trabajo de las funcionalidades sea el correcto, así como la técnica de partición de equivalencia, el método Delphi y prueba de aceptación.

Conclusiones Generales

CONCLUSIONES GENERALES

El análisis realizado en las diferentes bibliografías y referentes sobre los sistemas basados en casos (SBC), evidenció la necesidad de desarrollar un sistema de razonamiento basado en casos (RBC), para disminuir el esfuerzo de corrección de defectos en proyectos en desarrollo de software.

Se diseñó la solución al sistema modelando como implementar DECProD cumpliendo con los requisitos funcionales y no funcionales definidos.

Se describió un plan de pruebas para ejecutar la validación del sistema a partir de la aplicación de técnicas que garanticen el correcto funcionamiento del sistema y demuestran la satisfacción del cliente.

Recomendaciones

RECOMENDACIONES

1. Implementar la técnica de Inteligencia Artificial Razonamiento Basado en Caso(RBC), para que el sistema DECProD sugiera actividades de calidad para disminuir el esfuerzo de corrección de los proyectos en desarrollo.
2. Ejecutar al sistema RBC una vez terminado el plan de prueba que se describe en el capítulo 3 de esta tesis.

Referencias Bibliográficas

REFERENCIAS BIBLIOGRÁFICAS

1. PRESSMAN, R.S., El proceso. Ingeniería del Software: Un Enfoque Práctico. OJEDA-MARTÍN, R, 1998: p. 17-36.

2. ISO 9000:2015 “Sistemas de Gestión de la Calidad. Fundamentos y Vocabulario.

3. Ibáñez Galle, Javier. Ingeneria en Gestión de la Calidad. ISO-9000. Universidad Federico Santa María- Santiago. 2018. https://s3.amazonaws.com/academia.edu.documents/57316455/Clase_1-08-03-18-P_1-JIG-UFSM_STGO.pdf?response-content disposition=inline%3B%20filename%3DSISTEMA_GESTION_DE_LA_CALIDAD.pdf&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWOWYYGZ2Y53UL3A%2F20200128%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20200128T162500Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signature=818c74a448c9d35949c3212e8fbccb671063af033cd64e701dc5f75d5bc19d02

4. Marin Diaz, Aymara. MARCO DE TRABAJO CON UN ENFOQUE POR COMPONENTES PARA GESTIONAR ACTIVIDADES DE CALIDAD . La Habana : s.n., 2018.

5. Sommerville, I. and M.I.A. Galipienso, Ingeniería del software. 2005: Pearson Educación.

6. "UML y Patrones. Introducción al análisis y diseño orientado a objetos". Larman., C. pp 35-58., México : Prentice Halll, 1999.

7. Cañizares González, R., Repositorio de recursos educativos para las Instituciones de Educación Superior, in Departamento de producción de herramientas educativas. 2012, Universidad de las Ciencias Informáticas. p. 191.

8. PALENZUELA., ING. FILIBERTO LÓPEZ. Modelo para la toma de decisiones en los proyectos de software basado en los criterios de expertos. June 2013.

9. Tamayo, Irina Elsa Mayedo. Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

10. Lovelle, juan Manuel Cuevas. Análisis y diseño orientado a objetos. Oviedo, España : Servitec, abril, 2003.

11. Sebastián Badaró, Leonardo Javier Ibañez y Martín Jorge Agüero. Sistemas Expertos: Fundamentos, Metodologías y Aplicaciones. 2013 Universidad de Palermo, Facultad de Ingeniería.

Referencias Bibliográficas

12. Martínez Sánchez. Dra Natalia, García Lorenzo. Dra María Matilde, García Valdivia. Dra Zoila Zenaida. Modelo para diseñar sistemas de enseñanza-aprendizaje inteligentes utilizando el razonamiento basado en casos. 2009 Departamento de Ciencias de la Computación. Universidad Central de las Villas. Cuba

13. Fritz Bauer, First NATO Software Engineering Conference, Garmisch (Germany), 1968[16, 17].

14. M. Shaw y D. Garlan, Software architecture: Perspectives on an emerging discipline. Englewood Cliffs, NJ, USA: Prentice-Hall, 1996.

15. F. J. García-Peñalvo, "Ingeniería del Software," cap. 7, Universidad de Salamanca, 2018

16.Tello, Pablo Gabriel, Evaluación de Calidad de un Producto de Software. Facultad de Informática. Universidad de la Plata. Abril, 2016.

17. García Sánchez, Eduardo; Vite Chávez, Osbaldo; Navarrate Sánchez, Miguel Ángel;García Sánchez, Miguel Ángel; Torres Cosío, VerónicaMetodología para el desarrollo de software multimedia educativo MEDESMECPU-e, Revista de Investigación Educativa, núm. 23, julio-diciembre,2016,pp.216-226 Veracruz, México https://www.redalyc.org/pdf/2831/283146484011.pdf

18.García Peñalvo, Francisco José. Vázquez Ingelmo, Andrea. PROCESO. Departamento de Informática y AutomáticaUniversidad de Salamanca, 2019. https://repositorio.grial.eu/bitstream/grial/1511/3/3.%20Proceso.pdf

19. García Mata, Carolina. Gestión de la Calidad, Riesgos y Evaluación.Universidad Internacional deRioja.
https://s3.amazonaws.com/academia.edu.documents/56403607/01_Gestion_Calidad.pdf?response-content-
disposition=inline%3B%20filename%3DGestion_de_la_Calidad_Riesgos_y_Evaluaci.pdf&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWOWYYGZ2Y53UL3A%2F20200128%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20200128T163023Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-
Signature=955c00b1e83767f0275fd4a794b52c4f3a0003b32fed2a7ac2361638a19acf0a

20. Pressman, R.S., Software Engineering: A Practitioner's Approach, 7/e, RS Pressman & Associates. Inc., McGraw-Hill, ISBN. Vol. 73375977. 2010. Larman, C., UML y Patrones. Introducción al análisis y diseño orientado a objetos. Ed. 1999, Prentice Halll. HUMPHREY, W., Introducción al Proceso Software Personal. 1997. Marin Diaz, Aymara. MARCO DE TRABAJO

Referencias Bibliográficas

CON UN ENFOQUE POR COMPONENTES PARA GESTIONAR ACTIVIDADES DE CALIDAD . La Habana : s.n., 2018.

21.Cordero Morales Dasiel, Ruiz Constanten Yadira, Torres Rubio Yoanny. Sistema de Razonamiento Basado en Casos para la identificación de riesgos de software. 2013, http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992013000200010

22. López Rivero Elaine . Sistema Basado en Casos para contribuir a la disminución de los costos de la calidad de los proyectos del Centro de Telemática. 2013. http://repositorio.uci.cu.

23.Proaño Escalante, Rodrigo A.. Saguay Chafla, Ciro N. Jácome Canchig, Segundo B. Sandoval Zambrano, Fanny. Sistemas basados en conocimiento como herramienta de ayuda en la auditoría de sistemas de información. Enfoque UTE vol.8 supl.1 Quito feb. 2017. http://scielo.senescyt.gob.ec/scielo.php?script=sci_arttext&pid=S1390-65422017000100148

24. Lozano Laura, Fernández Javier , Razonamiento Basado en Casos: Una Visión General. https://www.infor.uva.es/~calonso/IAI/TrabajoAlumnos/Razonamiento%20basado%20en%20casos.pdf

25. de la Cruz Figueroa Luis Felipe, Fernández Rodríguez Ricardo, González Rangel Miguel Ángel . Hacia herramientas de inteligencia artificial en la enseñanza médica. Enfoque preliminar. 2018 . http://scielo.sld.cu/pdf/rcim/v10n1/rcim08118.pdf

26. Alvarez Cancio, Michel. Cruz de la Osa, Reyder. Hernández López, Iván. Aplicación de un sistema basado en casos para la identificación de opacidad en la cápsula posterior mediante imágenes del pentacam. RCIM vol.8 supl.1 Ciudad de la Habana 2016.

27. ING. FILIBERTO LÓPEZ PALENZUELA. Modelo para la toma de decisiones en los proyectos de software basado en los criterios de expertos.pdf. June 2013. Tamayo, Irina Elsa Mayedo. Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

28. HERNÁNS CHENPNE, MARCELO. Diseño de una Metodología Ágil de Desarrollo de Software.pdf. [online]. 2004. [Accessed 16 November 2016]. Available from: ftp://ucistore.uci.cu/documentacion/Ingenieria%20Software/eXtreme%20Programming/books/Dise%C3%B1o%20de%20una%20Metodolog%C3%ADa%20%C3%81gil%20de%20Desarrollo%20de%20Software.pdfFacultad de Ingeniería. Universidad de Buenos Aires. Tamayo, Irina Elsa Mayedo.

Referencias Bibliográficas

Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

29. Rodríguez Sánchez, Ing. Tamara. Metodología de desarrollo para la Actividad productiva de la UCI.pdf. [online].Available from: https://excriba.prod.uci.cu/proxy/alfresco/api/node/content/workspace/SpacesStore/a622adab-eac5-4fb3-ba08-a266767fff5f/Metodologia%20UCI.pdf?a=true

30. GÁMEZ BATISTA,ING. YALICE, MORENO VEGA, DR. VALERY y MARTÍNEZ MÁRQUEZ, MSC. YOAN. Consideraciones generales para el diseño de una herramienta interactiva de simulación de procesos.pdf. [online]. 2008. Tamayo, Irina Elsa Mayedo. Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

31. GONZALO GÉNOVA, JOSÉ M. FUENTES y MARIA C. VALIENTE. Evaluación comparativa de herramientas CASE para UML.pdf. http://s3.amazonaws.com/academia.edu.documents/45998726/Evaluacion_comparativa_de_herramientas_CASE_para_UML.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1487694932&Signature=PeJdsuX9RTPil0PKwh58Z%2B2wIbM%3D&response-content-disposition=inline%3B%20filename%3DEvaluacion_de_herramientas_CASE_para_UML.pdfDepartamento de Informática, Universidad Carlos III de Madrid. Tamayo, Irina Elsa Mayedo. Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

32. DENYS PÉREZ-BORROTO LORENZO. Subsistema de registro de equipos médicos para el sistema informatizado de salud.pdf [online]. La Habana: UNIVERSIDAD DE LAS C IENCIAS INFORMÁTICAS, 2011. Tamayo, Irina Elsa Mayedo. Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

33.JUAN DIEGO GAUCHAT. El gran libro de HTML5, CSS3 y Javascript [online]. primera. marcombo ediciones técnicas, 2012.

34.MIQUEL ÁNGEL SÁNCHEZ MAZA. Javascript [online]. INNOVA 2001. Tamayo, Irina Elsa Mayedo. Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

Referencias Bibliográficas

35. LANZA, Michele y MARINESCU, Radu. Object-oriented metrics in practice: using software metrics to characterize, evaluate, y improve the design of object-oriented systems; with 8 tables. Berlin: Springer, 2006. ISBN 978-3-540-24429-5. Tamayo, Irina Elsa Mayedo. Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

36. ANDINO GUEVARA, CRISTINA XIMENA, DESARROLLO DE UN SISTEMA ACADÉMICO ORIENTADO A LA WEB PARA LA UNIDAD EDUCATIVA JUAN DE VELASCO UTILIZANDO SYMFONY Y MYSQL, 2016, http://dspace.espoch.edu.ec/handle/123456789/6274.

37.CLAVADETSCHER, Charles. Autorización en PostgreSQL. [online]. 2015.

38.Sobre PostgreSQL | www.postgresql.org.es. [online]. Available from: http://www.postgresql.org.es/sobre_postgresql

39. PgAdmin: PostgreSQL administration y management tools. [online]. Available from: https://www.pgadmin.org/

40. Themes | PhpStorm Themes & Color Styles. [online]. Available from: http://www.phpstorm-themes.com/

41.Castillo Dayer, Juan Antonio. Diseño e implementación de una aplicación web para la administración de recursos de investigación del área de ingeniería telemática. [online]. 2008. Available from: http://repositorio.upct.es/hyle/10317/257. Tamayo, Irina Elsa Mayedo. Sistema de reportes basados en indicadores para contribuir a la toma de decisiones en la actividad productiva de la UCI. La Habana : s.n., junio, 2017.

42.PROJECT MANAGEMENT INSTITUTE. Guia? de los fundamentos de la dirección? de proyectos (Guía del PMBOK) 3era. 3era Edición. Newton Square, PA: Project Management Institute, 2004. ISBN 978-1-930699-73-1.

43.Mazo, Raúl , Carlos Andrés Jaramillo, Raúl. Hacia una nueva plantilla para la especificación de requisitos en lenguaje natural semi-estructurado. GiDITIC, Universidad Eafit, Medellín, Colombia Gerencia de innovación, SQA S.A., Medellín, Colombia.2019. https://hal.archives-ouvertes.fr/hal-02502483/

44. Mg. Navarro, Mirta E., Mg. Moreno, Marcelo P., Lic. Aranda, Juan, Lic. Parra, Lorena,Lic. Rueda, Jose R., Cruz Pantano, Juan. Integración de Arquitectura de Software en el Ciclo de Vida de las Metodologías Ágiles. Una Perspectiva Basadaen Requisitos. Departamento de Informática -

Referencias Bibliográficas

F.C.E.F. y N. - U.N.S.J. Complejo Islas Malvinas. Cereceto y Meglioli. 5400. Rivadavia. San Juan, 2017. http://sedici.unlp.edu.ar/bitstream/handle/10915/62077/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

45. LARMAN, Craig, HERNÁNDEZ RODRÍGUEZ, Luz María y CÁRDENAS ANAYA, Humberto. UML y patrones: introducción al análisis y diseño orientado a objetos. Naucalpan de Juárez: Prentice Hall, 1999. ISBN 978-970-17-0261-1.

46.Bolaños Benítez, Juan Sebastián, Medina Erazo, Luis Fernando. Proceso de validación para requisitos no funcionales bajo el enfoque de gestión del conocimiento. Universidad del Cuca, Facultad de Ingeniería Electrónica y telecomunicaciones, Departamento de Sistemas, Popayán 2017.
http://repositorio.unicauca.edu.co:8080/bitstream/handle/123456789/1738/PROCESO%20DE%20VALIDACI%C3%93N%20PARA%20REQUISITOS%20NO%20FUNCIONALES%20BAJO%20EL%20ENFOQUE%20DE%20GESTI%C3%93N%20DEL%20CONOCIMIENTO.pdf?sequence=1&isAllowed=y

47. Medina Cruza, Javier, Pineda Ballesteros, Eliecer, Téllez Acuña, Freddy Ronaldo. Requerimientos de software: prototipado, software heredado y análisis de documentos. Universidad Nacional Abierta y a Distancia - UNAD Magíster en Ingeniería e Informática, 2019. http://rcientificas.uninorte.edu.co/index.php/ingenieria/article/view/11452/214421444111

48. ROGER PRESSMAN. Ingeniería del Software. Un enfoque práctico.pdf [online]. 6th. Ed. McGraw-Hill, [no date]. [Accessed 20 marzo 2020]. Available from: ftp://ucistore.uci.cu/documentacion/Ingenieria%20Software/Ingenieria.del.Software.Roger.Pressman.6th.Ed.McGraw-Hill.pdf

49.LAURA MURILLO, RAMIRO PEDRO. ARQUITECTURA PERVASIVA CON TECNOLOGÍAS WebRTC HÍBRIDAS PARA EL DESARROLLO DE UN FRAMEWORK MODELO VISTA CONTROLADOR DE TIEMPO REAL. PUNO, PERÚ 2019. http://repositorio.unap.edu.pe/bitstream/handle/UNAP/12281/Ramiro_Pedro_Laura_Murillo.pdf?sequence=3&isAllowed=y

50.Blanco Hernández, Nancy, Montané Jiménez Luis G., Mezura Godoy, Carmen. SISTEMAS GROUPWARE PARA EL DISEÑO DE DIAGRAMA DE CLASES UML EN AMBIENTES TÁCTILES.

Referencias Bibliográficas

Pistas Educativas 127 (CITEC 2017), diciembre 2017, México, Tecnológico Nacional de México en Celaya. http://www.itc.mx/ojs/index.php/pistas/article/view/1081/877

51.García Ojalvo, MsC. Irina, Sepúlveda Lima, DrC. Roberto,. Abelló Ugalde, MsC. Isidro A. Sistema automatizado distribuido de ingreso a la educación superior SADIES. Uso de patrones de diseño. Ministerio de Educación Superior, Vol. 7, No. 6, Año. 2018. Calle 23 No. 667 esq. a E. Vedado, Cuba, 10400, revistacongreso@mes.gob.cu. http://revista.congresouniversidad.cu/index.php/rcu/article/view/1090/992

52.López Álvarez, Roniel. Herramienta de software para la toma de decisiones en los análisis de factibilidad. Universidad de las Ciencias Informáticas. La Habana, julio del 2018. http://habana.qfa.uam.es/~lmc/PREMIOS%202019/Ciencia%20T%E9cnicas/CD%20161/ACC%20Marieta/Tesis%20Completas/Tesis%20de%20Diploma%20Roniel-Lopez-Alvarez.pdf

53.MALDONADO VÁSQUEZ, JOAO ALFONSO. ANÁLISIS Y DISEÑO DE SISTEMAS ORIENTADO A OBJETOS PARA UNA BIBLIOTECA. UNIVERSIDAD JOSÉ CARLOS MARIÁTEGUI VICERRECTORADO DE INVESTIGACIÓN FACULTAD DE INGENIERÍA Y ARQUITECTURA. MOQUEGUA PERÚ 2017. http://repositorio.ujcm.edu.pe/bitstream/handle/ujcm/700/Joao_trabajosuficiencia_titulo_2017.pdf?sequence=1&isAllowed=y

54.AGÜERO, Esteban Ulises – BIAGETTI, Alejandro Nicolás. INTEGRACIÓN DE GESTIÓN DE PRUEBAS A LA ARQUITECTURA DE INTEGRACIÓN CONTINUA DESARROLLADA PARA EL SOFTWARE CIENTÍFICO TÉCNICO. https://rdu.iua.edu.ar/bitstream/123456789/403/1/TFG-Ag%C3%BCero-Biagetti-
vF8.pdf

55.Andrés Mera Paz Julián. Análisis del proceso de pruebasde calidad de software. 2016. https://repository.ucc.edu.co/bitstream/20.500.12494/962/1/Pruebas.pdf

56. VIERA IPANAQUE NAZARET TAYLER ALDAIR. IMPLEMENTACIÓN DE UN GENERADOR DE CASOS DE PRUEBA PARA PROCEDIMIENTOS EN LENGUAJE JAVA. 2019. https://revistas.udem.edu.co/index.php/ingenierias/article/view/1646/2547

57.Polanco Garay, Leodanny Wuilber; Moré Soto, Dailien; Saborit Figueroa, Alejandro Miguel. CONTROL DE ACTIVOS FIJOS POR EL PROYECTO INFOTEAM. http://www3.fi.mdp.edu.ar/riipro/journal/index.php/IJOPM/article/view/337/609

Referencias Bibliográficas

58.Gutierrez Artache,Juncal, EL USO DEL MÉTODO DELPHI COMO HERRAMIENTA DE EVALUACIÓN CONSENSUADA EN LA DIDÁCTICA DE LA TRADUCCIÓN: EL PERFIL DEL TRADUCTOR LOCALIZADOR, Universidad de Granada -España, 2017

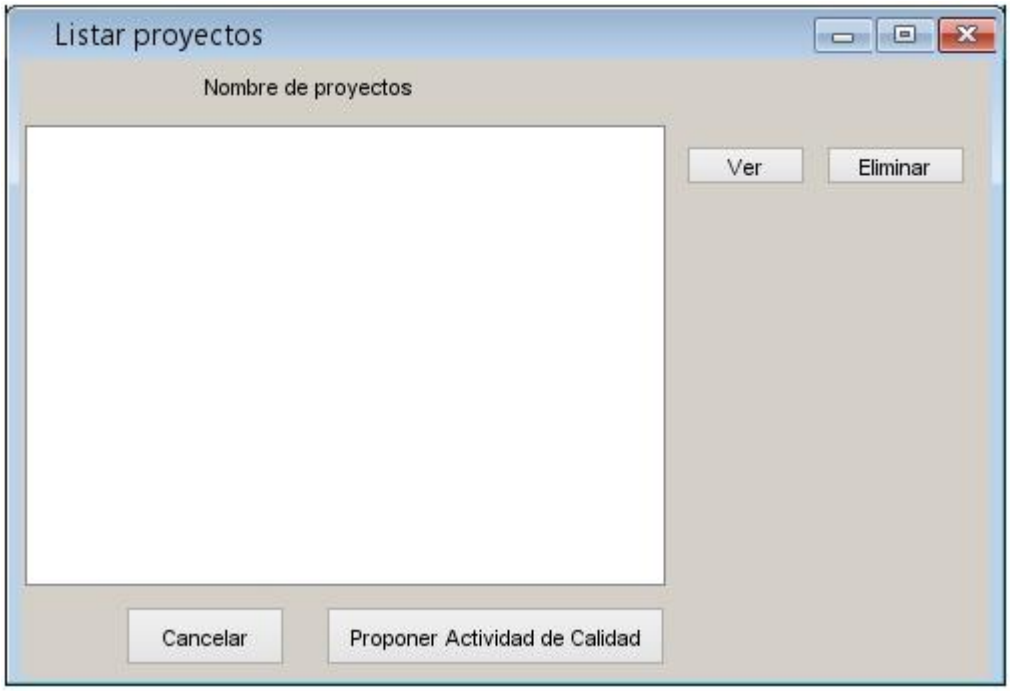
59.Hernández,R. Díaz,M. PROCEDIMIENTO PARA EVALUAR PROYECTOS Y ESTABLECER UN ORDEN DE PRIORIDADES PARA SU EJECUCIÓN, Universidad de las Ciencias Informáticas. Cuba,

60 Trujillo Casañola, Yaimi, MODELO PARA VALORAR LAS ORGANIZACIONES DESARROLLADORAS DE SOFTWARE AL INICIAR LA MEJORA DE PROCESOS", in Universidad de las Ciencias Informáticas, Ciudad de La Habana. 2014, Universidad de las Ciencias Informáticas.

.

ANEXOS

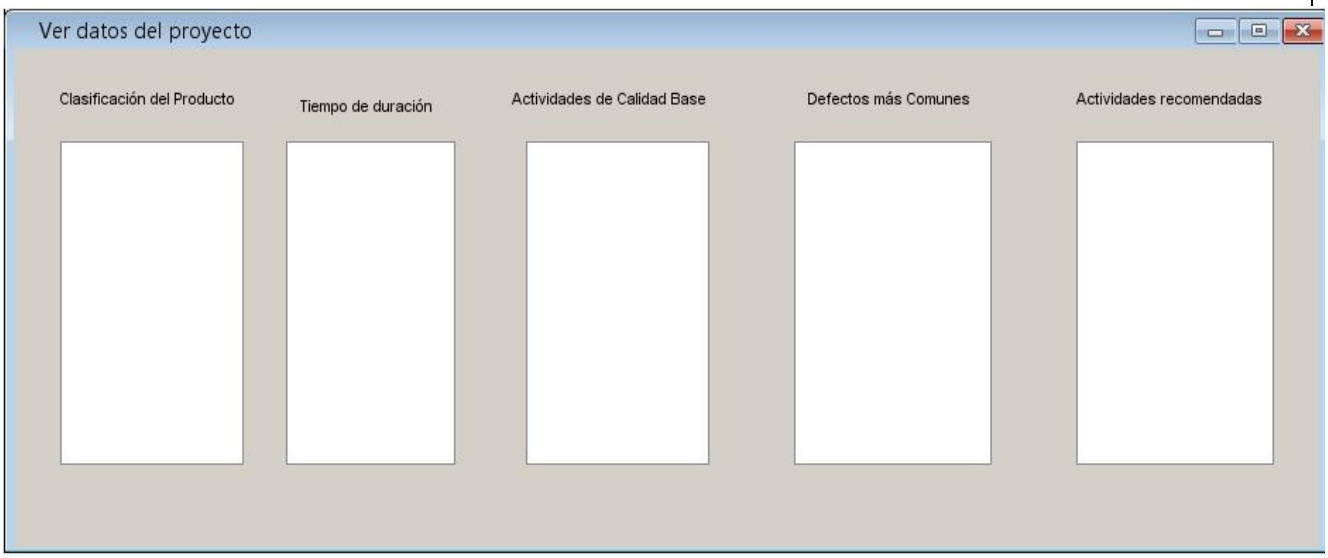
Anexo1: Historia de usuario *Listar proyectos*

Numero: 2	**Nombre del Requisito:**Listar proyecto.
Programador: Noel H. Hidalgo Reyes	**Iteración Asignada:**2
Prioridad: Alta	**Tiempo Estimado:**2 días
Riesgo en Desarrollo:	**Tiempo Real:** 48 horas
Descripción:	
Prototipo de Interfaz: 	

Anexo2:Historia de usuario *Ver datos de proyecto*

Numero: 3	**Nombre del Requisito:**Ver datos de proyecto.
Programador: Noel H. Hidalgo Reyes	**Iteración Asignada:**2

Prioridad: Alta	**Tiempo Estimado:**3 días
Riesgo en Desarrollo:	**Tiempo Real:** 60 horas
Descripción:	
Prototipo de Interfaz:	



Anexo3:Diseño caso de prueba *Listar proyectos*

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC 2: Listar proyectos	EC 2.1:Listar proyectos	El sistema procede a mostrar los proyectos con los datos solicitados.	1. Se selecciona la opción insertar proyecto. 2. Luego de haber introducidos los datos correctamente selecciona

			la opción guardar y listar. 3. El sistema muestra la interfaz Listar proyectos con todos los proyectos solicitados.

Anexo4:Diseño caso de prueba *Ver datos de proyecto*

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC 3: Ver datos de proyecto	EC 1: Ver datos de proyecto	El sistema muestra los datos de los proyectos solicitados.	1. Se muestra la interfaz listar proyectos. 2. Se selecciona la opción Ver. 3. El sistema muestra la interfaz con los datos solicitados