Universidad de las Ciencias Informáticas

Facultad 3

“Componente para la revisión de expedientes de proyecto en el Centro de Calidad, Estándares y Seguridad de la empresa XETID”

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

Autora:

Elizabeth de la Caridad Batista Martínez

Tutores:

MSc. Handy Hernández Dalmau

Ing. Riandrys Góngora Román

Ing. Mariela Milagros Bony Fernández

La Habana, 2020

“Año 62 de la Revolución”

Declaración de autoría

Declaro ser la autora de la presente tesis y se reconoce a la Universidad de las CienciasInformáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste se firma la presente a los ___ días del mes de _____ del año _____.

Elizabeth de la Caridad Batista Martínez

_____

Firma de la autora

MSc. Handy Hernández Dalmau

_____

Firma del Tutor

Ing. Riandrys Góngora Román

_____

Firma del Tutor

Ing. Mariela Milagros Bony Fernández

_____

Firma de la Tutora

… "La mejor forma de predecir el futuro es implementarlo" …

David Heinemeier Hansson.

Agradecimientos

Agradezco en primer lugar a mi mamita Niurka por ser tan especial en mi vida y darme los mejores consejos de este mundo, además de estar siempre a mi lado, en los buenos y malos momentos, tú eres mi inspiración para ser mejor cada día.

Le agradezco muchísimo a mi papá Reynaldo por acompañarme durante toda mi carrera brindándome su apoyo y la comprensión de un padre hacia un hijo en su carrera profesional.

A mi hermano, por ser el único que tengo y que lo amo cantidad.

A mi familia en general, que siempre han sido especiales conmigo.

A mis profesores, quienes me han ayudado a llegar a este momento, a través, de sus conocimientos y dedicación.

A mis amigas del alma Susel, Arlety, Sandra, Mónica y Amanda que en estos 5 años de carrera me han ayudado y han estado ahí para todo lo que me hacía falta tanto de la carrera como familiar.

A mis amigos Michel, Martin, Reynier González, Alian y en especial a mi novio Sergio por apoyarme incondicionalmente durante todos estos años.

A todos mis compañeros del aula y amistades de la universidad por formar parte de mi vida.

Gracias a Dios por haberme permitido llegar al final de esta meta

A todos muchas gracias…

Dedicatoria

En especial a mis padres, quienes más han influido en mi formación profesional y siempre se han enorgullecido de mí. Por haber confiado en mí y darme el apoyo en esos momentos en que realmente no tenía donde encontrarlo y sobre todo por sus sacrificios para que yo llegara hasta aquí.

A mi hermano Reinier por darme todo su apoyo y amor en todos estos años.

A mis abuelos Gladys, William, Reinerio que desde el cielo me guiaron y en especial a mi abuela Mercedes que en todos estos años siempre estuvo pidiéndole a Dios por llegar hacer una profesional.

A mis sobrinos y primitas para que les sirva de ejemplo y logren ser alguien en la vida, que se preparen como futuros profesionales y logren sus sueños.

A mi madrina Teresita por haberme apoyado en todo momento en los buenos y malos.

A mi familia por poner sus expectativas, dedicación y esfuerzos a mis servicios.

No quiero dejar de dedicarle este Trabajo de Diploma a todas mi amigas y amigos.

Resumen

El proceso de documentación de software ha alcanzado un auge considerable en los últimos tiempos, lo que garantiza un software con mayor calidad. La Empresa de Tecnologías de la Información para la Defensa (XETID) no se ha mantenido al margen de los métodos de documentación en sus proyectos productivos. Es por ello que la XETID realiza la gestión de sus proyectos mediante un expediente de proyecto.

Debido a que muchos de los proyectos tienen diferentes políticas de desarrollo, la aplicación de un mismo expediente no cubre todas las necesidades existentes en cada proyecto de los diferentes centros de la empresa. El Centro de Calidad Estándares y Seguridad (CCES) también se ve afectada en este aspecto, ya que no cuenta con un sistema para revisión de los expedientes de proyectos. Por lo que proponer un componente que revise en un menor tiempo la estructura de los expedientes de proyectos, la nomenclatura de cada carpeta y archivo según el expediente que va estar cargado en la base de datos, si se encuentra algún error se describe como una no conformidad y se va a exportar en un Excel, es el objetivo de este trabajo de diploma.

En la investigación se sistematizan los elementos más significativos a tener en cuenta al documentar, centrando el estudio inicialmente en definir, qué es un expediente de proyecto. Se analizan, además, aspectos claves para la creación del mismo, tales como, la metodología en la que está centrado el proceso de desarrollo de los expedientes de proyecto. Por tal motivo se realiza un análisis de las principales características del expediente que está aplicándose en ellos.

Índice General

Índice de Figuras

Introducción

La informática ha aportado cambios significativos en la vida de la sociedad. Su impacto a nivel global se ve reflejado en todas las esferas y actividades sociales y cada día juega un papel fundamental en la vida cotidiana del hombre. Esto ha propiciado que los avances tecnológicos centren sus objetivos en mejorar la calidad y ampliar sus aplicaciones, principalmente en la producción del software. (Díaz, Pérez y Florido, 2011).

Casi la totalidad de los trabajos, trámites, procesos que se realizan en la actualidad están vinculados a la informática, que se convierte en una nueva fuente económica para muchos países. Esto ha propiciado que los avances tecnológicos centren sus objetivos en mejorar la calidad y ampliar sus aplicaciones, principalmente en la producción de software.

En Cuba, la informática se está convirtiendo en una fuente de desarrollo económico-social y para esto se viene realizando un trabajo significativo en materia de la informatización de la sociedad. La formación y capacitación de nuevos profesionales en las diferentes ramas de la informática constituyen un eslabón fundamental en el desarrollo de la industria del software a nivel nacional, pero para ello tiene que lograr una mayor calidad en sus productos, ya que el exceso de información constituye uno de los principales problemas. Es frecuente encontrar un gran número de información redundante e innecesaria mezclada con la que es realmente importante. Además de que existen empresas que se encargan de realizar estas pruebas, otras que las hacen a sus propios productos tal es el caso de la Empresa de Tecnologías de la Información para la Defensa (XETID), en la cual se encuentra el Centro de Calidad Estándares y Seguridad (CCES).

La Empresa de Tecnologías de la Información para la Defensa (XETID), la cual también tiene como objetivo el desarrollo de programas informáticos, velando celosamente por la calidad de los productos que desarrollan. Para garantizar que los proyectos obtengan productos de alta calidad, la XETID posee un centro dirigido especialmente a la revisión de los proyectos, el Centro de Calidad, Estándares y Seguridad. Este centro tiene como misión el certificar y evaluar productos informáticos de producción nacional o importada, procesos de desarrollo y organizaciones para la industria de software en las FAR y su visión es colaborar en la obtención de un elevado nivel de calidad de los productos y alcanzar así una alta satisfacción de los clientes.

Dentro del proceso de revisión de los proyectos se encuentra la revisión de los expedientes de proyecto, la cual se ejecuta teniendo en cuenta una lista de verificación que contiene una serie de preguntas que guían al especialista en la revisión. Por cada pregunta de la

lista de verificación el especialista debe verificar su cumplimiento en el expediente de proyecto que esté revisando y luego redactar la no conformidad correspondiente, en caso de haber detectado alguna. Este proceso requiere tiempo, precisión y destreza en la revisión de cada detalle, teniendo en cuenta que cada expediente está organizado por carpetas que están enumeradas y además poseen un nombre que ayuda a identificar su contenido. A su vez, una carpeta puede contener dentro otras carpetas o documentos, que también tienen un orden y un nombre definido en el expediente plantilla. Cada uno de estos elementos debe ser verificado por el especialista, tomando como punto de partida las preguntas en la lista de verificación.

Una vez concluida la revisión de un expediente, el especialista debe elaborar un resumen donde se refleje el total de no conformidades detectadas, el impacto de cada una de ellas, cuáles son las carpetas y documentos cuya numeración y nombre no coinciden con el expediente plantilla, cuáles son las carpetas y documentos que no se encuentran en el expediente revisado, entre otros elementos.

Otro elemento de importancia dentro de estas revisiones, es la diversidad de los expedientes a revisar, ya que la empresa XETID cuenta con expedientes de proyectos de desarrollo de portales para proyectos que brindan servicios, entre otros.

De manera general, la revisión de un expediente de proyecto demanda un alto nivel de experiencia por parte del especialista, que debe interpretar correctamente cada pregunta de la lista de verificación y conocer bien las especificaciones de cada tipo de expediente de proyecto, para saber en qué parte del expediente buscar la evidencia apropiada. Además, la revisión implica tiempo, precisión y destreza por parte del especialista, al tener que revisar pregunta a pregunta de la lista de verificación y cada carpeta y documento de un expediente, para luego elaborar los resultados. Este proceso puede ser ejecutado por un mismo especialista para varios expedientes de proyecto al mismo tiempo, por lo que puede tornarse en un trabajo tedioso y engorroso. La inexperiencia, el cúmulo de documentación y el nivel de detalle en la revisión pueden conllevar a revisiones incompletas donde se pasen por alto detalles de importancia para el resultado final.

Para propiciar una mejor ejecución del proceso, se necesita que la revisión del expediente sea ágil, sencilla, flexible al cambio y que pueda ser realizada por cualquier especialista independientemente de su experiencia, o incluso por el propio proyecto como herramienta de auto-evaluación, sin que ello implique demora en su proceso de desarrollo. Por lo expuesto anteriormente se define como **problema a resolver:** ¿Cómo contribuir a laimplementación de un componente que permita la revisión de la estructura y formato de los expedientes de proyectos en el Centro de Calidad, Estándares y Seguridad de la

empresa XETID?; teniéndose como **objeto de estudio:**Estructura y formato de los expedientes de proyecto en el Centro de Calidad, Estándares y Seguridad de la empresa XETID, enmarcando el **campo de acción:**Revisión de la estructura y formato de los expedientes de proyecto en el Centro de Calidad, Estándares y Seguridad de la empresa XETID.Para darle solución a la problemática planteada se define como **objetivo general:**Desarrollar un componente informático que permita revisar la estructura y formato de los expedientes deproyecto en el Centro de Calidad, Estándares y Seguridad de la empresa XETID.

Con vista a darle cumplimiento al objetivo planteado se han definido los siguientes **Objetivos específicos:**

1. Realizar el estudio del estado del arte para lograr un mejor entendimiento del objeto abordado en la presente investigación.
2. Elaborar el levantamiento y descripción de los requisitos para identificar las funcionalidades y otras particularidades que debe tener el componente a desarrollar.
3. Realizar el análisis y diseño de la propuesta de solución como aproximación a la implementación del componente informático.
4. Implementar el componente para contribuir a la revisión de la estructura y formato de los expedientes deproyectos en el Centro de Calidad, Estándares y Seguridad de la empresa XETID.
5. Ejecutar las pruebas de software para comprobar el correcto funcionamiento del componente desarrollado.

Como resultado de la investigación se obtendrá un componente informático para la revisión de la estructura y formatode los expedientes de proyectos, permitiendo a los especialistas de calidad identificar las no conformidades en un menor plazo de tiempo y brindar una información más completa y exacta.

Como **métodos de investigación** científica se emplearon los siguientes:

Teóricos:

- **Análisis y síntesis:** Para analizar y extraer los conceptos más importantes que se relacionan con el objeto de estudio.
- **Análisis histórico-lógico:** Para analizar toda la evolución del problema que se estará estudiando además de soluciones existentes que permitirán recrear un mejor desglose del problema.

- **Modelación:** Para la creación de modelos, propuestas y alternativas, donde serán representadas las características y relaciones fundamentales del problema que se estará estudiando.

Empíricos:

- **La Entrevista:** Como método empírico, es una conversación planificada entre el investigador y el entrevistado para obtener información. Su uso constituye un medio para el conocimiento cualitativo de los fenómenos o sobre características personales del entrevistado y puede influir en determinados aspectos de la conducta humana por lo que es importante una buena comunicación.

La investigación se encuentra estructurada de la siguiente forma:

Capítulo1: "Fundamentación teórica":en este capítulo se relacionan los conceptos y aspectos teóricos que sustenta el desarrollo del componente de expediente de proyecto, se realiza el estudio del estado del arte para lograr un mejor entendimiento del objeto abordado en la presente investigación. Además, se explican las distintas herramientas y tecnologías que se van a usar, así como la metodología de desarrollo de software.

Capítulo2. "Análisis y diseño": se realiza una propuesta de la solución y se definen los requisitos funcionales y no funcionales para la misma, así como los artefactos establecidos en la metodología utilizada, necesarios para realizar el análisis y diseño del componente.

Capítulo3: "Implementación y pruebas": en este capítulo se definen los estándares de codificación del lenguaje y otros artefactos empleados durante la implementación del componente, así como los casos de pruebas para validar el correcto funcionamiento del componente implementado.

CAPÍTULO 1. Fundamentación Teórica

En el presente capítulo se definen los principales conceptos sobre los cuales se encuentra enfocada la investigación, como expediente de proyecto, revisión técnica formal. Se hace un estudio de las herramientas que se utilizan con el fin de alterar en el correcto funcionamiento del componente. Además, sedescriben lasprincipales tecnologías, metodología y herramientasaemplearen el desarrollo de la solución.

1.1 Conceptos asociados a la investigación.

Hoy en día, el desarrollo de componentes informáticos, es una parte crítica de todo negocio. Esto se debe a la calidad insuficiente de la recopilación de información para la creación de estos componentes desde sus inicios.

Para resolver el problema mencionado anteriormente, en la actualidad, se hace uso de un expediente de proyecto para mejorar el proceso de gestión de la documentación, o sea, la forma en que deben recogerse las características del producto de software desde su fase inicial de desarrollo, hasta el momento que está en manos de los usuarios finales. Toda empresa o institución desarrolladora de productos informáticos, tratan de llevar junto al desarrollo del software; el llenado de aquellos artefactos que respondan al modelo de calidad y a la metodología escogida para regir su trabajo, mediante un expediente de proyecto. De este modo se garantiza el éxito del desarrollo de los componentes informáticos.

Con lo descrito anteriormente se define como un expediente de proyecto:

Sistema informático, que a su vez es un conjunto de documentos y otros elementos que registran las diferentes etapas que se llevan a cabo durante el proceso de desarrollo de un proyecto informático. Este expediente incluye dos partes: una de archivo, de los datos que hacen referencia a las actividades del proyecto ya cerradas, y otra activa, que es la que el jefe de proyecto actualiza día a día. (Álvarez Amarelys,2012).

Según Yanko Hernández Valdés, asesor de la Dirección de Calidad, en la UCI, define a un expediente de proyecto como la forma de registro de todos los sucesos, elementos usados y cualquier otro tipo de acontecimiento que sea necesario documentar a lo largo del ciclo de vida del software. Su objetivo esregistrar todo el ciclo de forma tal que no se pierdan elementos generados para garantizar la no duplicación de esfuerzos y acceder fácilmente a cualquier necesidad de información. (Álvarez Amarelys,2012).

1.1.1 ¿Cuál es la documentación que por lo general se guarda en un expediente de proyecto? ¿Cómo se organiza esta información dentro del expediente de proyecto?

El objetivo del expediente de proyecto es documentar toda la información de los proyectos garantizando de esta forma el cumplimiento de los requerimientos mínimos de Calidad. El expediente se encuentra organizado por centros, dentro se encuentra toda la documentación referente a la Gestión de proyectos, así como la Ingeniería de dominio. (Empresa de tecnologías de la información para la defensa (XETID), 2019).

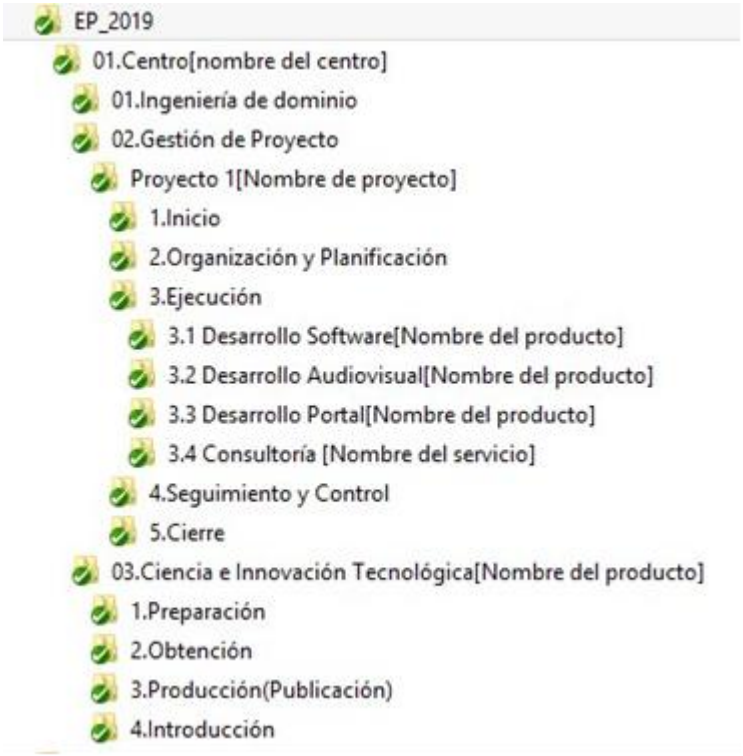

Figura 1-1: Estructura del expediente de proyecto.

Sistema informático.

Un sistema informático resulta de la interacción entre los componentes físicos que se denominan hardware y los lógicos que se denominan software. A estos hay que agregarles el recurso humano, parte fundamental de un sistema informático.(Álvarez Pérez,2012).

Expediente

Un expediente es el conjunto de papeles y documentos que corresponden a un determinado asunto o negocio. También puede tratarse de la serie ordenada de actuaciones administrativas o judiciales. (Álvarez Pérez,2012).

Proyecto

Proyecto es un plan de trabajo, con acciones sistemáticas, o sea, coordinadas entre sí, valiéndose de los medios necesarios y posibles, en busca de objetivos específicos a alcanzar en un tiempo previsto.

1.2 ¿Cómo se revisa un expediente de proyecto?

La empresa XETID está estructurada por centros de desarrollo, estos a su vez se dividen en departamentos que agrupan proyectos afines a una temática,uno de ellos es el centro de Calidad, Estándares y Seguridad que está estructurado en dos líneas de servicios: de revisiones y pruebas de software.

- 1-Los especialistas de las líneas de servicios de revisiones realizan a los expedientes de proyectos la**Revisión Técnica Formal** ya que es un proceso de revisión riguroso, su objetivo es llegar a detectar lo antes posible, los posibles defectos o desviaciones en los productos que se van generando a lo largo del desarrollo. Esta característica fuerza a que se adopte esta práctica únicamente para productos que son de especial importancia, porque de otro modo podría frenar la marcha del proyecto a los expedientes de proyectos durante el ciclo de vida de desarrollo para garantizar la correcta adecuación de los mismos a las metodologías y estándares definidos en la empresa. (Hernández y Fernández, 2012).
- 2-El equipo de pruebas de software es el encargado de revisar las pruebas a los productos del software durante un ciclo de vida garantizando la calidad de los mismos antes de su comercialización al cliente.

1.2.1 Herramientas para la revisión de expedientes de proyectos. SVN VS GIT

Los llamados sistemas de control de versiones fueron creados con el fin de detectar cambios en los documentos o archivos y se encargan de guardar todas las versiones anteriores, incluyendo el registro de fecha y hora, así como el identificador del usuario de un archivo para que los datos puedan ser recuperados y restaurados en cualquier momento. De esta forma, es posible determinar qué usuario ha realizado cambios en un punto determinado. Los objetivos generales de este tipo de sistemas consisten en coordinar el acceso compartido de varios usuarios a los archivos y permitir el desarrollo simultáneo de varias bifurcaciones o *branches*.

Generalmente, los sistemas de control de versiones se utilizan para el desarrollo de software, para aplicaciones de oficina y para gestores de contenido. Dos de los más conocidos son Apache Subversion (SVN) y Git, los cuales pueden ser instalados internamente en el servidor propio o externamente en el servidor de algún proveedor de alojamiento web.(Digital Guide IONOS, 2020.)

Propiedades	**SVN**	**GIT**
Control de versiones	Centralizada	Distribuida
Repositorio	Un repositorio central donde se generan copias de trabajo.	Copias locales del repositorio en las que se trabaja directamente.
Autorización de acceso	Dependiendo de la ruta de acceso.	Para la totalidad del directorio.
Seguimiento de cambios	Basado en archivos.	Basado en contenido.
Historial de cambios	Solo en repositorio completo, las copias de trabajo incluyen únicamente la versión más reciente.	Tanto el repositorio como las copias de trabajo individuales incluyen el historial completo.

Aunque muchos usuarios se preguntan cuál de los dos programas de control de versiones es mejor, no existe una respuesta general. La elección del sistema de control de versiones más adecuado para uno u otro proyecto dependerá de los objetivos específicos. Ambos sistemas difieren en su estructura y en el proceso de trabajo resultante.

1.3 Metodología de desarrollo de software(Prodesoft)

El desarrollo de un producto de software va unido a un ciclo de vida compuesto por 5 fases: inicio, modelación, construcción, explotación experimental y despliegue, que comprenden todas las actividades, desde el momento en que surge la idea de crear un nuevo producto

de software, hasta aquel en que el producto deja definitivamente de ser utilizado por el último de sus usuarios.

Durante la **fase de Inicio** se logra una visión preliminar de la problemática a resolver y se definen los recursos relevantes para la ejecución del proyecto. Es decir, se describen los objetivos y el alcance del proyecto, se identifican los involucrados y ejecutores (entidades involucradas), se estima de manera general las actividades a realizar durante todo el ciclo de desarrollo del proyecto (Cronograma General), se establece la estrategia a seguir pararealizar la modelación del negocio y la captura de requisitos y de ser necesario se estiman los recursos materiales que deberán ser adquiridos.

En la **fase de Modelación** se capturan las partes esenciales del sistema, donde se identifican los procesos de negocio fundamentales y se aceptan los requisitos funcionales, obteniéndose la línea base de la arquitectura y una estrategia de construcción de la aplicación aprobada por los implicados en el proyecto. El hito fundamental de esta fase es la liberación de la arquitectura de sistema, datos y despliegue.

En la **fase de Construcción** se aclaran los requisitos restantes y se completa el desarrollo del sistema sobre una base estable de la arquitectura. Las fases anteriores sólo dieron una arquitectura básica que es aquí refinada de manera incremental conforme se construye el producto. En esta fase todas las características, componentes, y requisitos deben ser integrados, implementados, y probados en su totalidad, obteniendo una versión liberada del producto.

Durante la **fase de Explotación Experimental** se convierte la versión liberada del producto en una solución estable, donde se eliminan los errores que surgen durante las pruebas y se obtiene una certificación funcional y de seguridad del producto. En la fase de Despliegue se instala y configura el sistema para un ambiente de producción real, se capacita al personal que usará la aplicación y se continúa dando soporte durante la explotación del sistema, culminando de ser preciso con transferencias tecnológicas.

Figura 1-2: Etapas del ciclo de vida del Producto.

1.4 Lenguajes y herramientas para el modelado de la solución

1.4.1 Lenguajes de modelado.UMLv2.0

El UML (Lenguaje Unificado de Modelado) nació como una notación estándar de la construcción de modelos; mediante la notación de UML, se aprenden técnicas del análisis y el diseño orientados a objeto.

Sus funciones son resumibles en cuatro fundamentales:

- Visualizar: permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- Especificar: permite especificar cuáles son las características de un sistema antes de su construcción.
- Construir: a partir de los modelos especificados se pueden construir los sistemas diseñados.
- Documentar: los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

Los lenguajes de programación se han ido desarrollando y en la actualidad son utilizados para la creación de sitios web.

1.4.2 Herramientas de Modelado

El uso de las herramientas de modelado es muy importante para realizar el análisis del sistema. Estas ofrecen mejor precisión al trabajo mediante el modelo realizado, ya que es una forma de comprender en lo que se está trabajando.

Las herramientas CASE (Ingeniería de Software Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras. La tecnología CASE supone la [automatización] del desarrollo del software, contribuyendo a mejorar la calidad y la productividad en el desarrollo de sistemas de información.

Visual Paradigm v5.0

Para el desarrollo de este trabajo se utiliza la herramienta Visual Paradigm, ya que la misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación.

Esta herramienta permite:

- Disponibilidad en múltiples plataformas (Windows, Linux).
- Diseño enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo
- Disponibilidad de múltiples versiones, para cada necesidad.
- Licencia: comercial.
- Soporta aplicaciones Web.
- Compatibilidad entre ediciones.
- Ingeniería inversa - Código a modelo, código a diagrama.
- Generación de código - Modelo a código, diagrama a código.
- Diagramas de flujo de datos.
- Generación de bases de datos
- Transformación de diagramas de Entidad-Relación en tablas de Base de Datos.
- Ingeniería inversa de bases de datos-Desde Sistemas Gestores de Bases de Datos (DBMS) existentes a diagramas de Entidad-Relación.

- Generador de informes.

1.4.3 Lenguajes de Programación

PHP v5.2.11

“Hypertext Preprocessor" es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. El cliente solamente recibe el resultado de su ejecución en el servidor, sin ninguna posibilidad de determinar que código ha producido el resultado recibido.

El servidor web puede ser incluso configurado para que procese todos los ficheros con PHP soporten el uso de otros servicios y que usen protocolos como IMAP, SNMP, NNTP, POP3, HTTP y derivados. También se pueden abrir sockets de red directos e interactuar con otros protocolos.

Ventajas de PHP

- Es un lenguaje multiplataforma.
- Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una Base de Datos.
- El código fuente escrito en PHP es invisible al navegador y al cliente, ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados extensiones).
- Posee una amplia documentación en su página oficial (www.php.org), entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables, aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Tiene manejo de excepciones (desde PHP5).
- Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora

de programar (muchos otros lenguajes tampoco lo hacen), aun estando dirigido a alguna en particular, el programador puede aplicar en su trabajo cualquier técnica de programación y/o desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del patrón de diseño Modelo Vista Controlador (o MVC), que permiten separar el tratamiento y acceso a los datos, la lógica de control y la interfaz de usuario en tres componentes independientes.

JavaScript v1.7

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java.

XML

XML (Extensible Markup Language) es un lenguaje usado para estructurar información en cualquier fichero que contenga texto, como por ejemplo ficheros de configuración de un programa o una tabla de datos. XML es un estándar abierto y libre, creado por el Consorcio Word Wide Web, en colaboración con un panel que incluye representantes de las principales compañías productoras de software. Este lenguaje de marcas generalizado fue propuesto en 1996, y apareció una primera especificación en el año 1998. Desde entonces su uso ha tenido un crecimiento acelerado, que se espera que continúe durante los próximos años. En comparación con otros sistemas existentes para estructurar documentos, XML tiene la ventaja de poder ser más exigente en cuanto a organización se refiere, lo cual tiene como resultado final documentos mejor estructurados. Por esta razón ha ganado muchísima popularidad en los últimos años.

1.4.4 Sistemas gestores de Base de Datos

Se denomina Sistemas gestores de Base de Datos (SGBD), el software que permite la utilización y/o actualización de los datos almacenado en una o varias bases(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez.

Un SGBD es el software mediante el cual se puede manejar información almacenada, debido a que ofrece las herramientas necesarias para este tipo de trabajo. Facilita al administrador de la base de datos el control de todo el contenido y los programas que acceden a estos, asegurando de esta forma la seguridad de los mismos.

La forma de organizar las BD puede ser diferente en cada caso en particular, pero existen un conjunto de objetivos generales que deben cumplir los SGBD, de forma que faciliten el diseño de las aplicaciones. Entre los objetivos fundamentales se pueden encontrar:

- Independencias de los datos y los programas de aplicación.
- Minimización de la redundancia.
- Integración y sincronización de la BD.
- Integridad de los datos.
- Seguridad y protección de los datos.
- Facilidad de manipulación de la información y control centralizado.

PostgreSQL v9.3

Este SGBD es un software que permite la definición de bases de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación. Un SGBD relacional facilita a los usuarios describir los datos que serán almacenados en la Base de Datos junto con un grupo de operaciones para manejar los datos.

Para este trabajo se utiliza en SGBD PostgreSQL, desarrollado en la Universidad de California, en el Departamento de Ciencias de la Computación de Berkeley. Es un gestor de bases de datos de código abierto, brinda un control de concurrencia multi-versión (MVCC por sus siglas en inglés) que permite trabajar con grandes volúmenes de datos; soporta gran parte de la sintaxis SQL y cuenta con un extenso grupo de enlaces con lenguajes de programación.

1.4.5 IDE de desarrollo

NetBeans v7.3

NetBeans es un proyecto de código abierto muy usado en el mundo. Se fundó en junio del 2000. Este IDE es un entorno de desarrollo, es decir una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación, ese proyecto es un producto libre y gratuito sin restricciones de uso, bajo la licencia de GNU GPL v2.

Marco de trabajo

Un framework es un conjunto de herramientas, librerías, convenciones y buenas prácticas que pretenden encapsular las tareas repetitivas en módulos genéricos fácilmente reutilizables.

Entre las características más esenciales se encuentran:

- Propone un modelo de colaboración.
- Reutiliza el diseño y el código.
- Incorpora el conocimiento dominio para el que el frameworks fue diseñado.

Los frameworks se construyen para dar flexibilidad y generalidad, tratando de cubrir un dominio entero en vez de problemas determinados. Este enfoque produce un Generador de Aplicación más complejo y más extensible que en los sistemas de software tradicionales.

Zeolides v2.2.0

Conjunto de componentes de software librerías, herramientas y tecnologías libres integradas que permiten el desarrollo ágil y basado en componentes de aplicaciones web empresariales (múltiples propósitos, gran complejidad y grandes volúmenes de datos), centrando el desarrollo en el negocio, los requerimientos y las interfaces de usuario. Este framework presenta una estructura que facilita la comprensión por parte de los desarrolladores que trabajen sobre el mismo. En este ambiente se utiliza ExtJS para implementar la capa de presentación y utiliza el patrón arquitectónico Modelo Vista Controlador (MVC). El empleo de Zeolides está basado en las ventajas que posee:

- Los identificadores de cada tupla de las tablas son generados automáticamente en la Base de Datos como una secuencia.
- El antiguo método de try y catch para el lanzamiento de excepciones desaparece, en el nuevo marco con registrar las excepciones en el manager de excepciones, el framework es capaz de realizar un mejor tratamiento de las mismas.

- No es necesario en cada método de inserción de información a la Base de Datos ejecutar el método correspondiente en la clase del negocio correspondiente, el framework es capaz de que una vez tomados los datos de la presentación salvarlos en la Base de Datos directamente.

Ext JS v2.2

Este framework está bajo la licencia GPLv3 y multiplataforma, la versión utilizada es de las más avanzadas de Ext JS. Posibilita el uso del patrón Modelo Vista Controlador (MVC) para la vista. En las versiones anteriores de ExtJS el usuario del framework estructuraba su aplicación a su manera, problema que a partir de la versión 4.0 se está intentando solucionar, al definir una arquitectura para que todo el que use la plataforma y desarrolle a partir de ella lo haga de la misma manera, pudiéndose compartir los segmentos de código y ser más entendible por el que use el código.

Servidor para aplicaciones web

Apache v2.2

Un servidor web notable por su utilidad es ApacheHTTP Server comúnmente conocido como Apache. Es publicado bajo la licencia Apache, esta permite la distribución de derivados de código abierto a partir de su código fuente original. Es desarrollado y mantenido por una comunidad abierta de desarrolladores bajo los auspicios de la *Apache Software Foundation*.

Apache presenta características significativas entre las que se encuentran: bases de datos de autenticación y negociado de contenido. La arquitectura que utiliza es cliente servidor, donde el cliente realiza la petición al servidor y este atiende dicha petición enviando al usuario la respuesta a la solicitud.

Entre algunas de las ventajas que posee se puede encontrar: es modular, flexible eficiente. Es utilizado para mostrar páginas web con contenidos estáticos y dinámicos. Cuenta con una amplia aceptación en la red por lo que se puede conseguir ayuda fácil. Por todas las características y ventajas que ofrece, es el servidor web que se escoge para el desarrollo del sistema, pues entre todas ventajas se puede mencionar además que es un servidor web HTTP para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP y la noción de sitio virtual. Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido.

1.5 Conclusiones

El estudio de un grupo de definiciones asociados al expediente de proyecto y a su vez el análisis de los sistemas homólogos permitió un buen entendimiento de la investigación. Durante este capítulo fueron establecidas las herramientas, proceso de desarrollo y tecnologías a usar para la creación del componente de revisión de expedientes de proyectos. Además, la tecnología Prodesoft permitirá guiar todo el proceso de desarrollo, facilitando la implementación de la futura solución.

CAPÍTULO 2. Análisis y diseño

En el presente capítulo se realiza la descripción de la propuesta de solución y se detallan los principales requerimientos de la solución. Además, se explica el uso de los patrones de diseño empleados para la implementación del componente y el patrón arquitectónico a aplicar en el mismo. Por otra parte, se describen las historias de usuario para cada requisito funcional identificado.

2.1 Propuesta de Solución

El desarrollo del componente propuesto tiene como principal objetivo revisarla estructura de los expedientes de proyectos en el centro de Calidad, Estándares y Seguridad de la empresa Xetid, que cuente con seguridad, que sea ágil permitiéndole a los especialistas de la empresa poder revisar los expedientes en un menor tiempo posible y de encontrar las no conformidades.Se propone que el componente informático realice varias funcionalidades, revise la estructura de los expedientes de proyectos, la nomenclatura de cada carpeta y archivo según el expediente que va estar cargado en la base de datos, si se encuentra algún error se describe como una no conformidad y se va a exportar en un Excel.

2.2 Modelo conceptual y sus conceptos

El modelo conceptual es un proceso abstracto para desarrollar una vista alternativa de la situación del problema, y entonces, volver al mundo verdadero, y finalmente, probar al modelo; también muestra lo que debería suceder al lograr los objetivos especificados y es construido utilizando la teoría de sistemas para describir lo que el sistema necesita hacer. (Delegado, Machado, 2002).

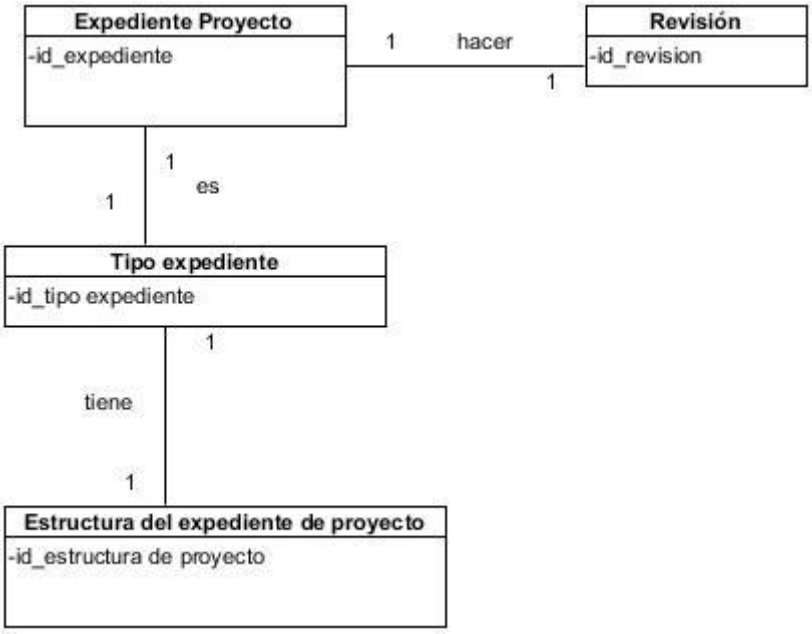

Figura 2-1: Modelo conceptual de expediente de proyecto.

Principales conceptos

Tipo expediente: puede definir los **tipos de expediente**que permiten clasificar los diferentes expedientesque se van a revisar

Los tipos de proyectos pueden ser:

- Desarrollo de Software
- Desarrollo de Portales
- Desarrollo Audiovisuales
- Consultoría

Estructura del expediente de proyecto: Todo expediente de proyecto está estructurado por secciones de archivos y carpetas.

Revisiones: Se les realizara a los expedientes de proyectos.

2.3 Requisitos

Los analistas de sistemas pueden clasificar los requisitos identificados en dos grandes grupos: los requisitos funcionales y los requisitos no funcionales. A menudo, los requisitos de sistemas de software se clasifican en funcionales y no funcionales, o como requisitos del dominio.

2.3.1 Requisitos Funcionales

Los requisitos funcionales hacen referencia a la descripción de las actividades y servicios que un sistema debe proveer. Normalmente este tipo de requisitos están vinculados con las entradas, las salidas de los procesos y los datos a almacenar en el sistema.

Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requisitos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer.

Los requisitos funcionales de un sistema describen lo que el sistema debe hacer. Los cuales dependen del software a desarrollar y en su mayor parte se los redactan de una forma abstracta. Pero estos describen con detalle la función de este, sus entradas, salidas y excepciones. En el caso del sistema para la revisión de los expedientes de proyectos en el Centro de Calidad, Estándares y Seguridad de la empresa XETID se dividió en varias áreas de trabajo una para gestionar estándares, otra para visualizar, descargar, revisar el expediente y por último generar las no conformidades. (Rojas, Pérez y Delgado, 2019).

RF1: Gestionar nomenclador de configuración o de estructura.

Descripción: Permite Adicionar, Modificar y Eliminar nomenclador de los expedientes de proyectos.

RF1.1 Adicionar nomenclador.

Descripción: Permite adicionar nomenclador de los expedientes de proyectos.

RF1.2 Modificar nomenclador.

Descripción: Permite modificar nomenclador de los expedientes de proyectos.

RF1.3 Eliminar nomenclador

Descripción: Permite eliminar nomenclador de los expedientes de proyectos.

RF2: Descargar expediente de proyecto.

Descripción: Permite descargar el expediente de proyecto.

RF3: Visualizar el expediente de proyecto a revisar.

Descripción: Permite visualizar el expediente de proyecto.

RF4: Revisar el expediente de proyecto.

Descripción: Permite revisar el expediente de proyecto.

RF5: Generar no conformidades.

Descripción: Permite generar no conformidades que se lleguen a generar en un Excel.

RF6: Exportar no conformidades.

Descripción: Permite exportar no conformidades.

2.4 Prototipos de interfaz de usuario y Especificación de requisitos

Especificación del requisito Adicionar nomenclador del Expediente de Proyecto

Conceptos tratados	**Conceptos**	**Atributos**
	• Expediente • Estructura de expediente de proyectos • Tipo de expediente • Revisión	Id_denominación
Descripción	1. Se selecciona el menú principal la opción Calidad. 2. Se selecciona la opción Tipo de expediente. 3. Se muestra la interfaz con la opción Adicionar. 4. Se Selecciona la opción Adicionar. 4.1. Si muestra la interfaz con la opción Adicionar el tipo de expediente. 5. Se selecciona el tipo de expediente. 6. Se muestra la interfaz con la estructura del tipo de expediente 7. Se selecciona la opción Revisar. 8. Si no deseas guardarlo, se selecciona el botón cancelar y se cancela la operación.	

	9. Si deseas guardarlo y si se selecciona la opción aceptar se cierra la interfaz, guarda el tipo de expediente adicionado y se actualiza el listado de la interfaz. 10. Si deseas guardar y no deseas aceptar, se selecciona la opción aplicar, se guarda la descarga, se actualiza el listado de la interfaz y se mantiene abierta la interfaz.
Complejidad	Alta
Validaciones	Para las acciones Aceptar Los botones se habilitarán cuando se haya introducido los valores correctos, en el campo. El sistema valida los datos según lo descrito en el documento Modelo conceptual_Expediente de Proyecto. Si ocurre un error al intentar acceder a la Base de Datos, el sistema muestra el mensaje: "*Los datos de conexión son incorrectos.* " La opción Adicionar del espacio de trabajo muestra como mensaje "*Adicionar prueba* ***(Alt+I)***". El botón Cancelar de la interfaz muestra como mensaje "*Cancelar acción*". Si existen campos con valores incorrectos se marca en rojo y se muestra un mensaje de los definidos en el documento Si un campo no puede ser nulo y se deja vacío se muestra el mensaje: "*Este campo es obligatorio*".
Post-condiciones	Se ha adicionado el tipo de expediente

Post-requisito	No procede.

Figura 2-2: Interfaz Adicionar nomenclador.

Especificación del requisito Modificar Nomenclador del Expediente de Proyecto

Conceptos tratados	**Conceptos**	**Atributos**
	• Expediente • Estructura de expediente de proyectos • Tipo de expediente • Revisión	id_denominacion
Descripción	1. Se selecciona en el menú principal la opción Calidad. 2. Se selecciona la opción Tipo de Expediente. 3. Se muestra la interfaz con la opción Modificar. 4. Se selecciona la opción Modificar. 5. Se muestra la interfaz el tipo de expediente a modificar. 6. Se modifica el tipo de expediente.	

	7.Sise desea guardar, se selecciona la opción aceptar, se cierra la interfaz, se modifica los datos y se actualiza el listado de la interfaz. 8.Si no se desea guardar, se selecciona la opción cancelar, se cancela la operación y se cierra la interfaz. 9. Se muestra el siguiente mensaje:" *El tipo de expediente ha sido modificado satisfactoriamente"*
Complejidad	Alta
Validaciones	El sistema valida los datos según lo descrito en el documento Expediente de Proyecto conceptual. Si ocurre un error al intentar acceder a la Base de Datos, el sistema muestra el mensaje: "*Los nomencladores no se pudieron modificar debido a un error interno.*", y se mantiene abierta la interfaz modificar nomenclador. La opción Modificar del espacio de trabajo Verificar tipo de expediente muestra como mensaje "*Modificar tipo de expediente*". El botón Cancelar de la interfaz Modificar tipo de expediente muestra como mensaje "*Cerrar ventana sin guardar los datos*". El botón Aceptar de la interfaz Modificar tipo de expediente muestra como mensaje *"Guardar datos y cerrar ventana".* El botón Cancelar de la interfaz Modificartipo de expediente muestra como mensaje "*Cerrar ventana sin guardar los datos*". El botón Aceptar de la interfaz Modificartipo de expediente muestra como mensaje *"Guardar datos y cerrar ventana".* Si un campo no puede ser nulo y se deja vacío se muestra el mensaje: *"Este campo es obligatorio".*
Post-condiciones	Se ha modificado untipo de expediente de proyecto.
Post-requisito	No procede.

Figura 2-3: Interfaz modificar nomenclador.

Especificación de requisito Eliminar Tipo de Expediente de Proyecto.

Conceptos tratados	**Conceptos**	**Atributos**
	• Expediente • Estructura de expediente de proyectos • Tipo de expediente • Revisión	Id_denominacion
Descripción	1. Se selecciona en el menú principal la opción Calidad. 2. Se selecciona la opción Tipo de expediente. 3. Se muestra la interfaz con la opción eliminar. 4. Se muestra la opción el listado de lo que se quiere eliminar. 5. Se selecciona la opción eliminar. 6. Se muestra el siguiente mensaje de confirmación:" *¿Está seguro de que quiere eliminar el tipo de expediente seleccionado*? 7. Si se desea confirmar, se selecciona la opción aceptar, se cierra el mensaje y se elimina el expediente. 8. Si no se desea confirmar, se selecciona la opción cancelar, se cancela la operación y se cierra la interfaz. 9. Se muestra el siguiente mensaje de información:" *El tipo de expediente de proyecto ha sido eliminado*".	

Complejidad	Alta
Validaciones	La opción Eliminar de la barra de herramientas del espacio de trabajo se habilitará cuando se seleccione un estándar en la lista de estándares. Si ocurre un error al intentar acceder a la Base de Datos, el sistema muestra el mensaje: "*El tipo de expediente no se puedeeliminar debido a un error interno.*", y se mantiene abierta la interfaz. La opción Eliminar del espacio de trabajo muestra como mensaje *"Eliminar tipo de expediente de proyecto".*
Post-condiciones	Se ha eliminado untipo de expediente de proyecto.

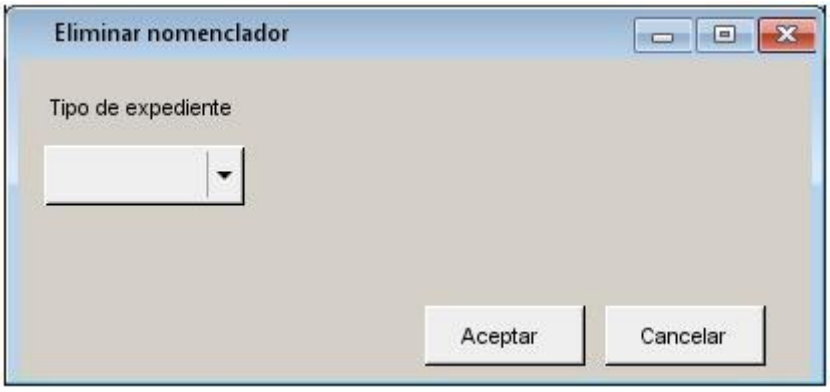

Figura 2-4: Interfaz eliminar nomenclador.

2.3.2 Requisitos no Funcionales

Los requisitos no funcionales describen otras prestaciones, características y limitaciones que debe tener el sistema para alcanzar el éxito. Los requisitos no funcionales engloban características como rendimiento, facilidad de uso, presupuestos, tiempo de entrega, documentación, seguridad y auditorías internas.

Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requisitos no funcionales a menudo se aplican al sistema en su totalidad. Normalmente apenas se aplican a características o servicios individuales del sistema. (Molina, Granda y Velázquez, 2019).

Hardware

Características	PC_Cliente	Servidor de Aplicaciones	Servidor de Base de Datos
Memoria RAM	1GB	8GB	8GB
Capacidad de Almacenamiento	10GB	150GB	80GB
Capacidad de Procesamiento	3.0GHZ	16GHZ	16GHZ
Capacidad de Transmisión de datos	1GB	1GB	1GB

Software

PC_Cliente	Servidor de Aplicaciones	Servidor de Bases de Datos
Mozilla Firefox 28 o superior, recomendado 28. Ofimática. Soporte para formato PDF.	Sistemas operativos Debian server 7.0. Servidor Web Apache 2.4 o superior, con modulo PHP 5.	PostgreSQL 9.3 como Sistema Gestor de Base de Datos.

Rendimiento/Eficiencia

- Los tiempos de respuesta y velocidad de procesamiento de la información serán rápidos, no mayores de 5 segundos para las actualizaciones y 10 para las recuperaciones excepto consultas complejas.
- El tiempo máximo de espera estará en dependencia de las operaciones realizadas, generalmente no deben exceder los 30 segundos.

Aplicación

La cantidad máxima de usuarios concurrentes en el sistema estará regulada principalmente por las presentaciones de los servidores tanto el servidor Web como el de Bases de datos, a mayores presentaciones aumentará la cantidad de usuarios soportados por el sistema. Para los requerimientos mínimos descritos en este documento de hardware podrán interactuar con el sistema un total de 200 usuarios sin afrontar dificultades de rendimiento y hasta 250 usuarios obteniendo tiempos de respuesta fuera de los rangos establecidos anteriormente.

Base de Datos

El sistema pedirá una Base de Datos con una cantidad de 50 peticiones simultaneas de una consulta a dicha Base de Datos, con 2 o 3 minutos de ser estresado por completo el sistema en peticiones. El tiempo que tardara la Base de Datos para dar respuestas a las consultas realizadas, según cada nivel de complejidad (Alta, Media y Baja).

- Bajas 10 segundos.
- Media 30 segundos.
- Alta 1 minuto.

El posible rendimiento de la Base de Datos ante las condiciones que se especifican en los acápites de hardware y software.

Confiabilidad

La información del sistema estará disponible todo el tiempo que el usuario solicite, deberá prever contingencias que pueden afectar la presentación estable y permanente del servicio al desarrollo de aplicaciones. Por ejemplo, prever casos de excepciones y errores por parte de los programadores como usuarios finales del producto final.

Seguridad

La información estará protegida contra accesos no autorizados utilizando mecanismos de validación que puedan garantizar el cumplimiento de esto: cuenta, contraseña y nivel de acceso, de manera que cada uno pueda tener disponible solamente las opciones relacionadas con su actividad y tenga datos de acceso propios, garantizando así la confidencialidad.

El sistema contara con 2 roles (administrador y especialista).

Integridad: la información manejada por el software será objeto de cuidadosa protección contra la corrupción y estados inconsistentes, mediante el mantenimiento de las bases de datos, así como la salva de la información se realizará a través de cada componente, de igual manera el origen y autoridad de los datos.

Disponibilidad: A los programadores de las aplicaciones como usuarios finales del Marco de Trabajo se les garantizará el acceso a la información y los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado de manera que se les permitirá a los usuarios tener atribución respecto a sus funciones de trabajo.

Sólo tendrán acceso al componente, las personas calificadas para interactuar con el marco de trabajo según conocimientos y competencias.

Confidencialidad: La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

Usabilidad

- El sistema empleará barras de progreso para indicar el estado de los procesos que por su complejidad requieran de un tiempo de procesamiento apreciable por los usuarios.
- Se utiliza el idioma español para todo el sistema.
- El software tendrá siempre visible la opción de Ayuda, lo que posibilitará un mejor aprovechamiento de sus funcionalidades.
- Deberá emplearse el uso de tooltips y teclas calientes. Además, se debe hacer uso del paginado en aquellas interfaces que lo requieran.

Restricciones de Diseño

El producto de software final debe diseñarse sobre una arquitectura cliente-servidor. Debe emplear los estándares establecidos en la entidad. Debe usar como lenguaje del lado del servidor, PHP y del lado del cliente JavaScript.

Interfaz

Para el desarrollo del sistema se utilizaron los estándares empleados en la empresa. La distribución mínima de la pantalla es de 800x600. Se utilizan los protocolos de comunicación https. Soporta los periférico mouse y teclado. Utiliza PostgreSQL 9.3, PHP 5.4, JQuery 1.8, EXT JS 2.2 y como operativo Unix.

Información que comparten los componentes:

- Información e Incidentes se comparten información entre sí. Ambos con Actores fuentes y Seguimiento y Flujo de Trabajo.

Portabilidad

El sistema será multiplataforma (Linux y Windows fundamentalmente). Después de instalado puede desinstalarse, presentando un fichero para el mismo, utilizando el Manual de instalación.

Políticos Culturales

El sistema solo debe contener palabras en el idioma español y debe respetar los términos empleados por la organización.

Legales

Requerimientos de licencias y patentes

Utilizará la licencia General Public License (GPL).

Aplicación de Estándares

El sistema estará regido por diferentes estándares establecidos:

- Estándar para la modelación de procesos de negocio y gestión de requisitos.
- Estándar de codificación.
- Estándar de documentación.
- Estándar de mensajes para las acciones 1.7.
- Estándar de teclas calientes 1.4

2.4 Diseño de la arquitectura propuesta

De acuerdo al modelo conceptual, la especificación de requisitos, los prototipos de interfaz usuario definidos para el sistema de revisión de los expedientes de proyectos en el Centro de Calidad, Estándares y Seguridad de la empresa Xetid, se especificó la integración del sistema con otros componentes, obteniendo el siguiente grafico de diagrama de componentes:

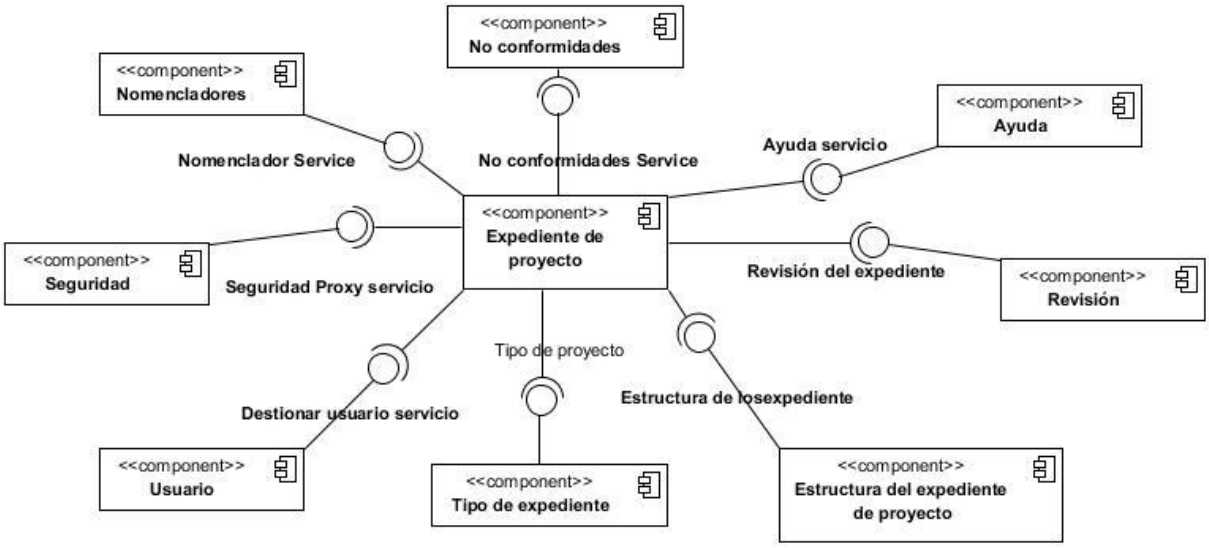

Figura 2-5: Diagrama de componentes de expediente de proyecto.

2.5 Patrones de Diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces, que brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. (Tedeschi, 2013).

En el módulo web, se utilizaron los siguientes patrones GRASP para asignar responsabilidades y los patrones GOF conocidos como los patrones de la pandilla de los cuatro:

2.5.1 Patrones GRASP

Patrón Experto

Es la clase que tiene la información necesaria para cumplir la responsabilidad. Expresa que los objetos deben hacer las cosas relacionadas con la información que poseen. Esto soporta un bajo acoplamiento, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento.

Patrón Bajo Acoplamiento

Asigna la responsabilidad de manera que el acoplamiento permanezca bajo. Permite crear clases más independientes, más reutilizables, lo que implica mayor productividad.

Patrón Controlador

Asigna la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase.

Patrón Creador

Guía la asignación de responsabilidades con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento.

Patrón Alta Cohesión

Asigna responsabilidades de manera que la cohesión se mantenga alta. Mide el grado en que están relacionadas las responsabilidades de una clase.

2.5.2 Patrones GOF

Patrón Facade (Fachada)

Viene de una única interfaz accediendo a un sistema completo, actuando como único punto de acceso, y hace que éste sea más fácil de utilizar.

Patrón Mediator (Mediador)

Define un objeto que encapsula cómo interactúan un conjunto de objetos. Promueve un bajo acoplamiento al evitar que los objetos se refieran unos a otros explícitamente.

2.6 Patrón arquitectónico.

El diseño del sistema está basado en la arquitectura Modelo Vista Controlador (MVC). Este patrón de arquitectura separa presentación e interacción de los datos del sistema. El sistema se estructura en tres componentes lógicos que interactúan entre sí. El componente **Modelo** maneja los datos del sistema y las operaciones asociadas a esos datos. El componente **Vista** (browser) define y gestiona cómo se presentan los datos al usuario. El componente **Controlador** (componentes del lado del servidor que manejan los requerimientos de HTTP) dirige la interacción del usuario y pasa estas interacciones a Vista y Modelo.

Tanto la vista como el controlador dependen del modelo, el cual no depende de las otras clases. Esta separación permite construir y probar el modelo independientemente de la representación visual. (Somerville, 2011).

Figura 2-6: Patrón arquitectónico Modelo Vista Controlador.

2.7 Conclusiones

A lo largo del capítulo se analizaron los requerimientos de la solución propuesta y en base a ellos, se generaron los artefactos: Modelo conceptual, levantamiento de requisitos, especificacion de requisitos, Prototipo de interfaz de usuario, diagrama de componente y diagrama de entidad-relación. Estos artefactos constituyen los cimientos, ya que traducen las necesidades del cliente en funcionalidades, recogen las tareas que debe realizar cada desarrollador agrupadas por iteración y el tiempo máximo del que dispone para ejecutarlas.
La determinación de los patrones arquitectónicos y de diseño usados permitió lograr una mayor organización en los elementos que conforman la solución.Se crearon los prototipos de interfaz con los que cuenta la solución permitió conocer el comportamiento de cada una de las clases en un alto nivel.

CAPÍTULO 3. Implementación y pruebas de los componentes de seguridad y licencia.

En el presente capítulo se analiza desde el punto de vista de la calidad, el cumplimiento del problema a resolver del trabajo de diploma, se establecen criterios asociados a la implementación y validación del sistema propuesto. En la realización de diversos sistemas informáticos juega un papel fundamental el uso de las técnicas de evaluación dinámica o pruebas. Para lograr esto debes llevar a cabo estrategias a la hora de evaluar dinámicamente el software partiendo del estudio y puesta en práctica del método de caja negra. Para ellos se describe de forma detallada todo el proceso de dichas pruebas y se muestra el análisis del resultado final.

3.1 Diagrama de despliegue

Cuando se trata de hardware y el software del sistema, se utiliza los diagramas de despliegue para razonar sobre la tipología de procesadores y dispositivos sobre los que reejecuta el software. Los diagramas de despliegue se utilizan para visualizar los aspectos estáticos de estos nodos físicos y sus relaciones y para especificar sus detalles para la construcción.

Un diagrama de despliegue es un diagrama que muestra la configuración de los nodos que participan en la ejecución y de los componentes que residen en ellos, gráficamente, un diagrama de despliegue es una colección de nodos y arcos.

El modelo de despliegue puede describir diferentes configuraciones de red, incluidas las configuraciones para pruebas y para simulación, además de representar una correspondencia entre la arquitectura de software y la arquitectura del sistema (hardware).

La figura muestra el diagrama de despliegue del sistema que se propone:

Figura 3-1: Diagrama de Despliegue.

PC_Cliente: espacio de trabajo por el cual el cliente interactúa con la aplicación. Esta computadora debe tener instalado el Sistema Operativo Windows o GNU/ Linux, como navegador Web Internet Explorer o Mozilla Firefox 2.0 como mínimo.

Servidor Web: se utilizará el servidor Web apache más el lenguaje de programación PHP. Contiene toda la información referente a la aplicación.

Servidor Base de datos: se utilizará PostgreSQL como base de datos que permitirá almacenar toda la información de la aplicación.

3.2 Implementación

Sedefinenestándaresdecodificaciónporqueunestilodeprogramaciónhomogéneo en un proyecto; permiteque todos los participantes lo puedan entenderen menos tiempo y que el código en consecuencia sea mantenible.

Programa fuente es una noción que se emplea como sinónimo de código fuente. Se trata de las instrucciones que un programa informático transmite a unacomputadora para que pueda ejecutarse. Dichas instrucciones son líneas de texto escritas en un lenguaje de programación (una estructura capaz de impartir instrucciones informáticas a partir de una determinada base semántica y sintáctica).

La legibilidad del código fuente, repercute directamente en el entendimiento que pueda tener otro programador del mismo, aspecto crucial ya que todo software tiene que someterse constantemente a mantenimiento y mejora de sus funcionalidades, por lo que en la implementación de los módulos de Planes y Actividades se utilizaron los siguientes estándares de codificación propuestos en los documentos “Estándar de Codificación para PHP” v 2.0, y “Estándar de Diseño de Interfaces para las aplicaciones de Gestión” v 1.0.

Algunos de los estándares reflejados en estos documentos son:

- El nombre de una clase comienza con la primera letra mayúscula y en caso de un nombre compuesto se empleará.
- El nombre de los atributos de una clase comienza con la primera letra en minúscula.
- Los nombres de las clases pertenecientes al dominio deben de empezar con la palabra “Dat” y el de las clases del modelo deben tener la palabra “Model” al final.

En las clases controladoras:

- Los nombres de los métodos serán definidos con la palabra “Action” al final.

- Cada método será público.
- El nombre de cada clase controladora debe terminar con la palabra “Controller”.
- Se utilizará el método Post para las llamadas.

3.3 Pruebas

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

Las pruebas, vistas desde el marco de un proceso de desarrollo de software, son los diferentes procesos que se deben realizar durante un desarrollo, con el objetivo de asegurar que este completo, correcto, tenga calidad, entre otros factores de gran importancia. (Sánchez, 2015).

Consisten en llevar a cabo la verificación dinámica de un componente, programa o sistema, mediante el uso de métodos, técnicas y herramientas especializadas, las cuales permiten detectar y corregir errores, problemas e inconsistencias durante el proceso de desarrollo.

Tipos de prueba

La Técnica de **pruebas de caja negra**, consiste en ver el programa que queremos probar como una caja negra despreocupándonos del comportamiento interno y concentrando el esfuerzo en encontrar el comportamiento incorrecto, de acuerdo a las especificaciones de dicho programa, teniendo solo en cuenta las entradas y salidas de dicho programa.Las herramientas básicas son observar la funcionalidad y contrastar con la especificación. Un tipo de pruebas funcionales, las pruebas de seguridad, estudian las funciones (por ejemplo, un cortafuego) asociadas a la detección de amenazas procedentes de personas ajenas y malintencionadas, tales como virus. Otro tipo de pruebas funcionales, las pruebas de interoperabilidad, evalúan la capacidad del producto de software de interactuar con uno o más componentes o sistemas especificados. (Quijano, 2018).

La Técnica de **pruebas de caja blanca**, al contrario de las pruebas de caja negra consiste en verificar la estructura interna de un programa componente con independencia de la funcionalidad establecida para el mismo. Por tanto, no se comprueba la corrección de los resultados, sólo si éstos se producen. Las pruebas no funcionales incluyen, pero sin

limitarse a ello, pruebas de rendimiento, carga, estrés, pruebas de usabilidad, prueba de mantenibilidad, pruebas de fiabilidad y pruebas de portabilidad. (Salazar, 2015).

Para la validación del sistema, se realizaron diversas pruebas de las antes mencionadas, tales como:

- **Pruebas funcionales haciendo uso de los diversos diseños de casos de prueba**: Las pruebas funcionales son aquellas que se aplican al producto final, y permiten detectar en qué puntos el producto no cumple sus especificaciones, es decir, comprobar su funcionalidad. Para realizarlas se debe hacer una planificación que consiste en definir los aspectos a examinar y la forma de verificar su correcto funcionamiento, punto en el cual adquieren sentido los casos de prueba.
- **Pruebas de código mediante el servidor de integración continua Jenkins:**Antes de ser ejecutado el sistema se le realizan pruebas estáticas, que no son más que pruebas a un componente o sistema a nivel de especificación o implementación sin ejecutar el software, por ejemplo, revisiones o análisis estático de código. Para esto se cuenta con el servidor de integración continua Jenkins.
- **Pruebas de seguridad:** Al igual que las pruebas funcionales y las observadas anteriormente, los usuarios de negocios primero tienen que querer la seguridad como un aspecto necesario de la calidad general. Una parte crítica de la creación de este compromiso es comunicarse con ellos acerca de los riesgos y beneficios, y establecer objetivos mutuamente acordados. Las actividades que se pueden realizar para hacer las pruebas de seguridad son diversas y se orientan a varios ámbitos, especialmente en lo relativo a asegurar el funcionamiento y disponibilidad de los servicios web y contenidos publicados.

En el proceso de pruebas se realizaron 3 iteraciones, resultando en la primera iteración 4 cantidades de no conformidades, las cuales se resolvieron todas. En la segunda iteración 2 cantidades de no conformidades y en la tercera iteración ninguna no conformidad.

3.4 Conclusiones

En el capítulo que concluye se determinó el diagrama de despliegue que rige como va a funcionar el componente y la comunicación entre los nodos. Con la aplicación de las pruebas se pudo detectar, documentar y corregir las no conformidades existentes en el componente implementado.Por último, se determinaron los tipos de pruebas de software a emplear, teniendo en cuenta que el producto es un componente para un sistema informático.

Conclusiones Generales

Finalizada la presente investigación se puede concluir que se desarrollaron todas las tareas a fin de cumplir los objetivos propuestos, resaltando que:

- Con la investigación del estudio del arte asociado a los sistemas homólogos permitió el desarrollo de una propuesta de solución.
- Con el estudio de los elementos más significativos de los expedientes de proyectos, se demostró que existe un estándar al documentar y que este proceso se desarrolla de forma particular en cada empresa u organización.
- Al utilizar la metodología de desarrollo de software(Prodesoft) se facilitó la adaptabilidad del expediente de proyecto para el desarrollo del componente para la empresa XETID.
- La aplicación de las pruebas permitió evaluar el funcionamiento de las operaciones implementadas de forma correcta.
- Con la puesta en práctica de todos los elementos analizados, se desarrolló un componente para las revisiones de los expedientes de proyecto acorde a las necesidades de los especialistas del área de Calidad de la empresa XETID.

Recomendaciones

Para dar continuidad a la presente investigación, se recomienda:

- El componente para la revisión de los expedientes de proyecto sea utilizado en todas las áreas de la empresa XETID y en otras empresas del país.
- Implementar la versión de que el componente revise todo el formato de cada artefacto que contenga cada carpeta.

Referencias Bibliográficas

(1) ÁLVAREZ DE ZAYAS, C., 1995. Metodología de la investigación científica. Santiago de Cuba, 88 pág.

(2) DELGADO DAPENA, M. y MACHADO, N., 2002. Modelo Conceptual de un proyecto de Software utilizando el razonamiento Basado en Casos. Facultad de Ingeniería Industrial, Universidad Tecnológica de La Habana José Antonio Echeverría, Cujae. [en línea]. [Consulado 18 marzo 2020]. Disponible en:http://rii.cujae.edu.cu/index.php/revistaind/article/view/228.

(3) DIAZ LAZO, J., PEREZ GUTIERREZ, A. y FLORIDO BACALLAO, R., 2011. Impacto de las tecnologías de la información y las comunicaciones (TIC) para disminuir la brecha digital en la sociedad actual. *cultrop* [en línea]. [Consulta: 7 febrero 2020]., vol.32, n.1 [citado 2020-08-27], pp.81-90. Disponible en: <http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S0258-59362011000100009&lng=es&nrm=iso>. ISSN 0258-5936.

(4) Digital Guide IONOS, 2020. Git vs SVN: ¿cuál es el mejor sistema de control de versiones? [en línea]. [Consulta: 20 marzo 2020]. Disponible en:https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/git-vs-svn-una-comparativa-del-control-de-versiones/

(5) Empresa de tecnologías de la información para la defensa (XETID)., 2019. Especificaciones de expedientes de proyecto. La Habana, 19 pág.

(6) Empresa de tecnologías de la información para la defensa (XETID)., 2019. Estándar de documentación. La Habana, V 3.0.0, 9 pág.

(7) Empresa de tecnologías de la información para la defensa (XETID). Proceso de Desarrollo de Software. La Habana, V 1.7 ,112 pág.

(8) Empresa de tecnologías de la información para la defensa (XETID). Proceso de Desarrollo de Software, XETID. La Habana, V 2.0, 128 pág.

(9) FERNANDEZ, R., 2008. Propuesta de un expediente, para los proyectos productivos del Polo de Software Libre, de la Facultad 10. Universidad de las Ciencias Informáticas. Cuba, 87pág.

(10) Uniwebsidad, sa. Git- Manual de usuario. Versión 1. [en línea]. [Consulta: 20 marzo 2020]. Disponible en:https://uniwebsidad.com/libros/jobeet-1-4/capitulo-1/subversion

(11) GitHub Docs., 2020 ¿Cuáles son las diferencias entre Subversion y Git?[en línea]. [Consulta: 14 marzo 2020]. Disponible en:

https://help.github.com/es/github/importing-your-projects-to-github/what-are-the-differences-between-subversion-and-git

(12) GONZALEZ CASTELLANOS, R.A., YLL LAVIN, M. y CURIEL LORENZO, L.D., 2003. Metodología de la Investigación Científica para las Ciencias Técnicas. 1ra parte Diseño Teórico y Formulación del proyecto de investigación. Universidad de Matanzas, 59 pág.

(13) INEGI., 2011. Políticas para el desarrollo de un Sistema Informático.

(14) LARMAN, C., 2003. UML y PATRONES. Una introducción al análisis y diseño orientado a objetos y al proceso unificado. Madrid: PEARSON EDUCACIÓN, S.A. ISBN 84-205-3438-2.

(15) LOPEZ CHAVEZ, D.L. y ALFONSECA RAMIREZ, P., 2018. Pruebas de seguridad utilizando herramientas para la detención de vulnerabilidades.Dirección de Calidad de Software. Universidad de las Ciencias Informáticas. [en línea]. [Consulta: 28 marzo 2020], 7 pág.

(16) MARTIN GONZALEZ, Y. y RIOS HILARIO, A.B., 2005.Application of "Functional Requirements for Bibliographic Records" (FRBR) [en linea], catalogues.*ACIMED* [en línea], vol.13, n.4 [citado 2020-08-27], pp.1-1. Disponible en: <http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352005000400005&lng=es&nrm=iso>. ISSN 1024-9435.

(17) PÉREZ PORTO, J. y MERINO, M. Definiciones: Definición de expediente. [en línea]. [Consulta: 24 febrero 2020]. Publicado: 2010. Actualizado: 2012. Disponible en: (https://definicion.de/expediente/).

(18) RIL VALENTIN, E.B., RODRIGUEZ PUENTE, R., PINERO PEREZ, P. Y. y MARTINEZ NORIEGAS, H. A.,2013. Descubrimiento de conocimiento en lecciones aprendidas documentadas en los procesos de cierre de proyectos informáticos. *Rev cuba cienc informat* [online]., vol.7, n.3 [citado 2020-08-27], pp.45-57. Disponible en: <http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992013000300005&lng=es&nrm=iso>. ISSN 2227-1899.

(19) SOMMERVILLE, I., 2011. Software engineering. Ninth Edition. Massachusetts: Pearson Education, Inc. ISBN 9780137035151.

(20) TEDESCHI, N., 2013. ¿Qué es un patrón de diseño? [en línea]. [Consulta: 27 marzo 2020]. Disponible en: http://msdn.microsoft.com/es-es/library/bb972240.aspx

(21) TOLL PALMA, Y.C., RIL GIL, Y., sa. Propuesta de manual de procedimiento para pruebas de sistema. Universidad de las Ciencias Informáticas. [en línea]. [Consulta: 25 marzo 2020], 10 pág.

(22) VEGA MINIENT, Y., RAMOS NAVARRO, M., MUSTELIER SANCHIDRIAN, D. y PIÑERO PEREZ, Y., mayo 2014. Modelando el proceso de desarrollo del Software de la UCI con cbSPEM. Revista Cubana de Ciencias Informática,11 pág.

(23) HERNÁNDEZ LEÓN, R. A. y COELLO GONZÁLEZ, S., 2011. El proceso de investigación científica. Ciudad de La Habana: Editorial Universitaria, ISBN 978-959-161307-3, 110 pág.

(24) HERNANDEZ GONZALEZ, L.A., FERNANDEZ PEÑA, J.M., 2012. Herramienta para la automatización de revisiones técnicas formales como apoyo al desarrollo orientado a objetos. Reto. Universidad Veracruzana. [en línea]. [Consulta: 14 marzo 2020], 6 pág.

(25) ALVAREZ PEREZ, A., 2012.Expedientes de proyecto para el desarrollo exitoso de almacenes de datos en la Facultad Regional Mártires de Artemisa. Artemisa, 80 pág.

(26) CABRERA HERNANDEZ, M., 2013. Catálogo para la generación de diagramas de componentes del Sistema de Información para la Salud en Cuba. *RCIM* [en línea], vol.5, n.1 [citado 2020-08-27], pp.30-41. Disponible en: <http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1684-18592013000100005&lng=es&nrm=iso>. ISSN 1684-1859.

(27) SALAZAR MARTINEZ, E. enero 2015. Procedimientos para realizar pruebas de caja blanca [en línea]. [Consulta: 22 marzo 2020]. Disponible en: http://www.informatica-juridica.com/trabajos/procedimiento-realizar-pruebas-caja-blanca/

(28) SANCHEZ PEÑO, J.M., junio 2015. Pruebas de Software. Fundamentos y Técnicas. Escuela Técnica Superior de Ingeniería y Sistemas de Telecomunicación. Universidad Técnica de Madrid,130 pág.

(29) REYES PEREZ, J.M.,2016. CEAD. Componente de revisión de arquitectura de datos de la Empresa de Tecnología de la Información para la Defensa (XETID). Ciudad de la Habana, 69 pág.

(30) HUAQUISTO CÁCERES, S., 2016. Análisis de eficiencia en proyectos de inversión pública: Un estudio de casos en proyectos ejecutados por administración directa. Revista de investigaciones altoandinas 18 (1) 61-68 pág.

(31) QUIJANO, J. marzo 2018. ¿Qué pruebas debemos hacerle a nuestro software y para qué? [en línea]. [Consulta: 11 abril 2020]. Disponible en:

https://www.genbeta.com/desarrollo/que-pruebas-debemos-hacerle-a-nuestro-software-y-para-que

(32) LEYVA, A., 2018. Patrones de diseño de software. [en línea]. [Consulta: 6 abril 2020]. Disponible en: https://devexperto.com/patrones-de-diseno-software

(33) Subversion. [en línea]. [Consulta: 14 marzo 2020]. Disponible en: http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/325

(34) MOLINA HERNANDEZ, Y., GRANDA DIHIGO, A. y VELAZQUEZ CINTRA, A., 2019. Los requisitos no funcionales de software. Una estrategia para su desarrollo en el Centro de Informática Médica. *Rev cub cienc informat* [en línea], vol.13, n.2 [citado 2020-08-27], pp.77-90. Disponible en: <http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992019000200077&lng=es&nrm=iso>. ISSN 2227-1899.

(35) MELES, J. Apunte patrones y software UML 2.0. [en línea]. [Consulta: 20 marzo 2020]. Disponible en: https://es.scribd.com/document/253891624/Apunte-Patrones-GRASP, 22 pág.

(36) ROJAS ROBERT, D. M., PEREZ MORALES, Z. y DELGADO DAPENA, M. D.,2019. Generador de valores interesantes para casos de pruebas unitarias. *Ing. Ind.* [en línea], vol.40, n.2 [citado 2020-09-14], pp.183-193. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59362019000200183&lng=es&nrm=iso