Universidad de las Ciencias Informáticas

FACULTAD 6

Título:

Sistema de Información para la gestión de la formación y calidad de software en DATEC.

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

Autor: Alberto García Tosca

Tutor(es): MSc. Reynaldo Alvarez Luna.
Ing. Claudia García Suárez del Villar.
Ing. Clara Elena Brizuela Figueredo.

La Habana, Junio de 2014

"Año 56 de la Revolución"

Declaración de Autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Alberto García Tosca

_____

Firma del Autor

MSc. Reynaldo Alvarez Luna.

_____

Firma del Tutor

Ing. Claudia García Suárez del Villar.

_____

Firma del Tutor

Ing. Clara Elena Brizuela Figueredo.

_____

Firma del Tutor

Datos de Contacto

Autor: Alberto García Tosca
Universidad de las Ciencias Informáticas
E-mail: atosca@estudiantes.uci.cu

Tutor: MSc. Reynaldo Alvarez Luna
Máster en Ciencias
E-mail: rluna@uci.cu

Tutora: Ing. Claudia García Suárez del Villar
Ingeniero en Ciencias Informáticas
E-mail: cgarcias@uci.cu

Tutora: Ing. Clara Elena Brizuela Figueredo
Ingeniero en Ciencias Informáticas
E-mail: cebrizuela@uci.cu

Agradecimientos

Si existe una persona a la que debo agradecer es a mi madre, que por su apoyo, paciencia, ayuda y consejos es tan merecedora de este título como yo. Tengo que agradecerle no permitirme nunca que me rindiera, ni siquiera que apartara la vista del objetivo. Gracias a ti madre, hoy somos ingenieros. También le agradezco a mi padre que aunque vivamos lejos siempre ha sabido estar cerca en los momentos necesarios para ayudar o aconsejar como seguir y salir adelante. A mi abuela por su preocupación constante sobre mis resultados en las parciales, aunque no tuviera en ese momento. Al resto de mi familia, que de diversas formas siempre aportaron algo para que este día pudiera ser posible, especialmente a mi tío Carlos. Gracias a ti tío puedo decir hoy “Coroné patrón”.

Le agradezco a mis amigos, los viejos y los hechos en estos 5 años, que durante la carrera han compartido conmigo en los buenos y malos momentos, especialmente al Gaby que ha sido mi buen amigo en esta campaña, en las fáciles y en las difíciles. Al resto del piquete de los animales.

A mis tutores Reynaldo, Claudia y Clara por su paciencia para conmigo, por saber guiarme y aconsejarme en este último año. A la secretaria Fifi, que fue la primera persona que me brindó su ayuda desde el primer día que ingresé a la escuela y hasta hoy nunca me ha faltado su apoyo. A todos los profesores que a lo largo de la carrera han contribuido a este logro. A todos ellos mis más sinceros agradecimientos.

A todos aquellos que por falta de tiempo no pueda nombrar, gracias por haber sido parte, de una forma u otra, de este trayecto de 5 años.

Dedicatoria

Le dedico este momento tan importante de mi vida a mi familia que siempre me ha brindado su apoyo incondicional. En especial a mi madre y mi padre, que son los verdaderos autores de este trabajo.

Resumen

La Información y el Conocimiento, juegan un papel determinante en la sociedad, debido a que ambos conforman la base para el cumplimiento de objetivos y metas. Para la obtención de los máximos beneficios que estos pueden brindar, se hace necesaria una gestión acorde a las necesidades. En este sentido los Sistemas de Información (SI) han ido convirtiéndose poco a poco a través de los años, en una valiosa herramienta dentro de las organizaciones. La aplicación de un SI, asegura una mejoría en la eficiencia, así como el aumento de la productividad.

La gestión de los procesos de formación y calidad de software del Centro de Tecnologías de Gestión de Datos de la Universidad de Ciencias Informáticas, genera informaciones que suelen almacenarse utilizando software como Microsoft Word y Excel, lo que atenta contra la integridad de los datos y dificulta el proceso de gestión de estos y la elaboración de informes. A raíz de estos problemas, se expone en el presente documento un sistema que centraliza la información de los procesos antes mencionados. Utilizando tecnologías web actualizadas y brindando a los usuarios, altos niveles de usabilidad en la gestión de la información, la solución además garantiza la integración al Sistema para la Gestión de Proyectos a través de servicios para garantizar el manejo único de la información, garantizándose así la disponibilidad, integridad y confidencialidad de la información.

Abstract

The information and knowledge play a key role in society, because both are the basis for the achievement of objectives and goals. To obtain the maximum benefits they can provide, it is necessary a management according to the needs. In this sense, information systems have been becoming gradually over the years, a valuable tool within organizations. Applying an IS ensures improved efficiency and increased productivity.

The management of Training and Software Quality process in the Center of Data Management Technologies at the University of Information Sciences generates information typically stored using software such as Microsoft Word and Excel, which undermines the integrity of the data and hinders the process of managing these and reporting. Because of these problems, is presented in this paper a system that centralizes information of the aforementioned processes. It was developed using updated web technologies and providing to the users a high levels of usability in information management, the solution also ensures the integration with the system for project management through services to ensure the only access to the information management, thus ensuring availability, integrity and confidentiality of information.

Índice de Contenidos

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

La información en los procesos de toma de decisiones, a pesar de ser un factor tan vital y necesario, en muchas ocasiones es ignorada u olvidada por la gerencia de la organización. No prestar debida atención a este aspecto sería un punto vulnerable de las organizaciones y pudiera provocar la extinción de las mismas. Actualmente, tanto la información como el conocimiento juegan un papel determinante en la sociedad, debido a que ambos conforman la base para el cumplimiento de objetivos y metas. Estos no serían alcanzados con solo contar con la información correcta. Para la obtención de los máximos beneficios que puede brindar esta, se hace necesario gestionarla acorde a las necesidades.

La Gestión de la Información es el conjunto de procesos que interactúan con la información desde su generación o captura hasta su disposición final, siempre garantizando la disponibilidad, la integridad y la confidencialidad de la misma. Comprendiendo además su selección, combinación, filtrado y difusión a los interesados. El no disponer de una adecuada gestión de la información empaña los elementos relacionados con la eficiencia y la eficacia, componentes imprescindibles para alcanzar el éxito en la entidad.

En este sentido los Sistemas de Información (SI) han ido convirtiéndose poco a poco a través de los años, en una valiosa herramienta dentro de las organizaciones, mejorando la eficiencia y aumentando la productividad en las mismas. Un SI es una combinación de fuentes de información junto con una serie de mecanismos para su gestión, aprovechando al máximo los recursos de información de las organizaciones en aras de una continua evolución y de facilitar la toma de decisiones, posibilitando una mejoría en la posición competitiva de la organización.

Para las organizaciones cubanas, el contar con un SI que facilite el análisis oportuno de la información sería un factor de éxito en el perfeccionamiento empresarial. El desarrollo de estos sistemas es asignado, en su mayoría, a la Universidad de Ciencias Informáticas (UCI). La UCI como institución desarrolladora de software, utiliza una estructura de proyectos agrupados en centros de desarrollo con temáticas afines que vinculan estudiantes, profesores y especialistas en la producción. El Centro de Tecnologías de Gestión de Datos (DATEC), forma parte de dicha estructura, siendo un centro especializado en tecnologías de Gestión de Datos.

Entre los procesos que atiende DATEC se encuentran los procesos de calidad de software y formación. El proceso de calidad de software controla la gestión de los proyectos y la calidad de los mismos. De igual modo el proceso de formación controla la gestión de la superación profesional, la formación profesional y los recursos humanos. En la actualidad DATEC se enfrenta a una serie de problemas que entorpecen el trabajo que se desarrolla con la información generada de la gestión de los procesos mencionados.

En el área de la calidad de los proyectos se controla el estado de los proyectos y el proceso de calidad de los mismos. Cada proyecto cuenta con una serie de documentos que reflejan su estado e historial, esta información suele ser actualizada con frecuencia, lo cual se ve dificultado por la ausencia de centralización de la misma.

El área de la formación profesional gestiona el proceso de formación de los estudiantes vinculados a la producción, mediante las prácticas profesionales hasta la culminación de su formación con el desarrollo del trabajo de diploma. Al igual que en las otras áreas, la gestión de esta información genera gran número de documentos que dificultan la manipulación, el análisis y el acceso a ella. Estos problemas también se evidencian en el área de los recursos humanos, la cual realiza el control y seguimiento de las actividades laborales realizadas por cada uno de los recursos humanos del centro.

Por último y no menos importante se encuentra el área de la superación profesional, en esta área se gestiona la planificación y el control de las participaciones de los trabajadores en actividades, eventos y cursos de superación, lo que también genera un cúmulo elevado de documentos que traen consigo los problemas anteriormente mencionados.

De manera general en los procesos de formación y calidad de software del centro DATEC se gestiona la información sin un control eficiente que garantice la confidencialidad de la misma. Tampoco existe un lugar donde se pueda publicar y obtener esta información de forma fácil y centralizada garantizando su disponibilidad. Además estas informaciones suelen almacenarse utilizando software como Microsoft Word y Excel, lo que atenta contra la integridad de los datos y dificulta el proceso de gestión de estos, su recuperación y la elaboración de informes.

A raíz de la problemática planteada, surge el siguiente **problema de la investigación** ¿Cómo centralizar la información gestionada en los procesos de formación y calidad de software en el Centro de Tecnologías

de Gestión de Datos? Donde el **objeto de estudio** está dirigido a los Sistemas de Información, enmarcado en el **campo de acción:** Sistemas de Información en el Centro de Tecnologías de Gestión de Datos. Enfocado en dar solución al problema planteado se define como **objetivo general:** Desarrollar un sistema que centralice la información de la gestión de la calidad de software y la formación en el Centro de Tecnologías de Gestión de Datos.

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

- Identificar los requisitos funcionales del sistema a desarrollar a partir de la información gestionada en los procesos de formación y calidad de software del centro DATEC.
- Realizar el diseño del Sistema de Información para la gestión de la formación y calidad de software en DATEC.
- Implementar el Sistema de Información para la gestión de la formación y calidad de software en DATEC.
- Realizar pruebas al Sistema de Información para la gestión de la formación y calidad de software en DATEC.

Las preguntas científicas se elaboran para lograr un mejor entendimiento del problema. Las respuestas de las siguientes preguntas científicas convergerán a la solución del problema planteado:

- ¿Cuál es la información gestionada en los procesos de formación y calidad de software en DATEC?
- ¿Qué sistemas aportan información a la gestión de los procesos de formación y calidad de software en DATEC?
- ¿Cómo garantizar la integración y funcionalidad del sistema de información a implementar?

Para dar cumplimiento a los objetivos enunciados anteriormente se proponen las siguientes **tareas de investigación:**

- Identificación de los elementos teóricos fundamentales relacionados con los sistemas de información para guiar el desarrollo de la investigación.
- Identificación de la metodología, herramientas y tecnologías a utilizar en el desarrollo del Sistema.
- Análisis de la información gestionada en los procesos de formación y calidad de software en DATEC para la identificación de requisitos.

- Definición de los requisitos funcionales y no funcionales para el desarrollo del Sistema de Información.
- Diseño del Sistema de Información para los procesos de formación y calidad de software.
- Implementación de las funcionalidades identificadas.
- Diseño de los casos de pruebas para determinar el correcto funcionamiento de los requisitos.
- Realización de pruebas funcionales al sistema.

Para el desarrollo de esta investigación particular se utilizará un enfoque sistémico, debido al análisis de los sistemas legados que pueden aportar información y viceversa, se utilizarán como métodos teóricos:

La modelación es justamente el método mediante el cual se crean abstracciones con el objetivo de explicar la realidad. Particularmente en la presente investigación dichas abstracciones constituyen los modelos de arquitectura, los diagramas de clase entre otros artefactos que se generarán con vistas a explicar el negocio actual y la solución.

Para la recogida de la información acerca del dominio o negocio de la aplicación se ha definido la utilización de **la entrevista** como técnica fundamental durante el proceso de identificación de los requisitos.

El presente trabajo de diploma ha sido estructurado de la siguiente manera: resumen, introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía y anexos. A continuación se resume el contenido fundamental de los capítulos:

CAPÍTULO 1: FUNDAMENTO TEÓRICO

En este capítulo se describen los principales conceptos relacionados con los Sistemas de Información, los cuales son vitales para la solución del problema planteado anteriormente. Se establecen las tecnologías, metodologías y herramientas a utilizar argumentando su selección.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA SOLUCIÓN PROPUESTA

En este capítulo se enfatiza en la propuesta de solución para el problema de la investigación. Mediante el modelo de dominio se describe la estructura que se desea representar en el sistema a desarrollar. Luego

de profundizar los conceptos fundamentales que describen el dominio de la aplicación, se establecen los requisitos funcionales y no funcionales, agrupándolos para la obtención del diagrama de casos de uso. De igual forma se diseñarán los diagramas de clases y de interacción para los casos de uso arquitectónicamente significativos. Se describen los patrones de diseño utilizados en la solución y la distribución física del sistema mediante el modelo de despliegue.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN

En este capítulo se hace énfasis en las disciplinas de implementación y pruebas. Se analizará el Modelo de Implementación así como una descripción de como los elementos del modelo de diseño se implementan en términos de componentes. De igual modo se realizarán las pruebas al sistema arrojando los principales resultados obtenidos a través de los casos de pruebas realizados a cada caso de usos.

CAPÍTULO 1: FUNDAMENTO TEÓRICO

En este capítulo se efectuará un estudio de los Sistemas de Información, abordándose los conceptos fundamentales relacionados con sus clasificaciones y las funciones que realizan. Se establecen las tecnologías, metodologías y herramientas a utilizar en el desarrollo de la aplicación que se implementará.

1.1. Gestión de la Información

La Gestión de la Información es el proceso que incluye "… operaciones como la extracción, manipulación, tratamiento, depuración, conservación, acceso y/o colaboración de la información adquirida por una entidad a través de diferentes fuentes y que gestiona el acceso y los derechos de los usuarios sobre la misma" (1).

En general se define a la gestión de la información como el proceso que incluye la obtención, manipulación, análisis, utilización y difusión de la información a los interesados, permitiendo así, explotar eficientemente los recursos de información con que cuenta la organización para apoyar a los directivos en la toma de decisiones. Por lo tanto, la gestión de la información implica (2):

- Selección de la información requerida.
- Recogida y análisis de los datos.
- Registro y recuperación de la información cuando sea necesario.
- Utilizarla y divulgarla.

1.2. Sistemas de Información

La denominación de Sistema de Información (SI) es muy ambigua en tanto que engloba gran cantidad de definiciones y enfoques, dependiendo de la perspectiva que adopte el autor de la investigación. A continuación, una selección de definiciones sobre los SI.

Según J. López un SI es "un conjunto de personas, maquinaria y procedimientos que integrados hacen posible a los individuos trabajar con ofertas y demandas que aparecen en el trabajo cotidiano" (3). K. Samuelson señala que un "sistema de información es la combinación de recursos humanos y materiales que resultan de las operaciones de almacenar, recuperar y usar datos con el propósito de una gestión eficiente en las operaciones de las organizaciones" (4). Los autores K. Laudon y J. Laudon definen a un

Sistema de Información como un organismo que recolecta, procesa, almacena y distribuye información. Son indispensables para ayudar a los gerentes a mantener el orden en la compañía, a analizar todo lo que por ella pasa y a crear nuevos productos que favorezcan a la organización (5).

El objetivo primordial de un SI es apoyar la toma de decisiones en la organización. Es importante señalar que existen dos tipos de SI, los formales y los informales; los primeros utilizan como medio para llevarse a cabo estructuras sólidas como ordenadores, los segundos son más artesanales y usan medios más antiguos como el papel y el lápiz (6).

De esta manera puede definirse un SI como un conjunto de personas, sistemas y procesos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio. Independientemente de las definiciones planteadas o el enfoque de los autores, todo SI mantiene un conjunto de flujos de información, de entrada y de salida, destinado a solucionar un problema informativo a cualquier escala.

Los SI canalizan la información desde las fuentes a los receptores. Estos procesan la información comunicándola, haciéndola accesible, organizándola y sometiéndola a mediación informática.

1.2.1. Clasificaciones y características de los Sistemas de Información

Los Sistemas de Información pueden clasificarse como:

- Los transaccionales, se caracterizan por estar diseñado para recolectar, almacenar, modificar y recuperar todo tipo de información que es generada por los procesos operativos en una organización, puede integrar las grandes bases de información institucional para ser utilizada posteriormente por los funcionarios de nivel operativo de la organización en la toma de decisiones.
- Los de apoyo a las decisiones, generalmente son desarrollados por el usuario final con el objetivo de proporcionar información de soporte para los mandos intermedios y la alta gerencia en el proceso de toma de decisiones.
- Los estratégicos, su función principal no es apoyar la automatización de los procesos operativos ni proporcionar información para apoyar la toma de decisiones, son desarrollados para uso interno, para lograr a través de su implantación y uso ventajas que los competidores no posean, apoyando al nivel alto de la organización.

A continuación se presentan características para todo Sistema de Información (7):

- Disponibilidad de la información. Suministro de información de manera selectiva.
- Variedad en la forma de presentación de la información.
- Generalidad, como las funciones para atender a las diferentes necesidades.
- Flexibilidad, capacidad de adaptación.
- Fiabilidad, para que el sistema opere correctamente.
- Seguridad, protección contra pérdidas.
- Amigabilidad, para el usuario.

Funciones de los Sistemas de Información

- Entrada de Información: Es el proceso mediante el cual el SI obtiene los datos que necesita para procesar la información. Las entradas pueden ser manuales o automáticas. En las manuales la información es proporcionada directamente por el usuario, mientras que en las automáticas los datos o información provienen de otros sistemas o módulos.
- Almacenamiento de información: A través de esta propiedad el sistema puede recordar la información guardada en el proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos. La unidad típica de almacenamiento son los discos magnéticos o discos duros, los discos flexibles o disquetes y los discos compactos.
- Procesamiento de Información: Es la capacidad del SI para efectuar una secuencia de operaciones preestablecida sobre datos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite transformar datos fuente en información que facilite la toma de decisiones.
- Salida de Información: Es la capacidad de un SI para sacar la información procesada o bien datos de entrada al exterior. Es importante aclarar que la salida de un SI puede constituir la entrada a otro sistema o módulo.

El SI que se obtendrá es clasificado como transaccional, ya que garantizará la gestión de la información de los procesos que son llevados a cabo en DATEC.

1.2.2. Sistemas de Información en DATEC

GESPRO Suite de Gestión de Proyectos

Este SI es utilizado en DATEC para la gestión de los proyectos, recoge información sobre las principales áreas de este proceso guardando información sobre los proyectos, los recursos humanos y otros tipos de recursos. Agrupa tanto a los sistemas de información, como a los sistemas de inteligencia empresarial o de negocios, con el propósito fundamental de apoyar la toma de decisiones. Su misión es crear bienes y servicios informáticos relacionados con la gestión de datos.

GESPRO no gestiona información referente al proceso de Formación, específicamente las áreas de superación y formación profesional. Una posible propuesta de solución al problema planteado sería integrar la gestión del proceso de Formación a este sistema, pero por políticas de seguridad informática se restringe la integración de nuevos módulos de gestión de información al GESPRO. Por otra parte, entre los servicios que ofrece destacan con singular importancia para la solución propuesta los referentes a los recursos humanos y a los proyectos, permitiendo utilizar la información existente de estos procesos en otro sistema que si permita la gestión de la información del proceso de Formación.

Generador Dinámico de Reportes

El Generador Dinámico de Reportes (GDR), es una aplicación web que permite a los usuarios abstraerse de los conocimientos relacionados con los gestores de bases de datos. GDR no gestiona información alguna, sino que, contribuye a la toma de decisiones pues permite generar reportes de forma rápida e interactiva.

Es una solución abierta y extensible para los informes administrados. Su arquitectura flexible permite a los programadores de software y a las empresas integrar el sistema con sus portales empresariales o aplicaciones personalizadas (8). El sistema está compuesto por 6 módulos: Diseñador de modelos, Diseñador de reportes, Diseñador de consultas, Visor de reportes, Administrador de reportes y Seguridad.

Luego del estudio realizado se propone desarrollar un nuevo sistema que permita gestionar la información del proceso de Formación y centralizarla con la del proceso de Calidad de Software existente en el GESPRO. Con este propósito se utilizarán como método de captación de la información para el nuevo sistema los servicios que brinda GESPRO referentes a los recursos humanos y a los proyectos. Luego de gestionada la información de los procesos de Calidad de Software y Formación, aprovechando la capacidad de integración de GDR, se decide integrarlo al nuevo sistema para el apoyo a la toma de

decisiones a partir de la generación rápida e interactiva de reportes.

En el siguiente epígrafe se describen los aspectos relacionados con la metodología de desarrollo de software seleccionada para conducir el desarrollo del Sistema de Información para la gestión de la formación y calidad de software en DATEC.

1.3. Metodologías de desarrollo de software

Una metodología de desarrollo de software es el entorno que se utiliza para estructurar, planificar y controlar el proceso de desarrollo de un sistema informático. La metodología da solución a los problemas existentes en la producción de software, englobando procedimientos, técnicas, documentación y herramientas que se utilizan en la creación de un producto de software. La importancia del uso de estas radica en que se trata de separar el aplicativo en procesos, y a la vez cada proceso en funciones, y para cada función determinar un tiempo aproximado de desarrollo. Para guiar el proceso de desarrollo del Sistema de Información para la gestión de la formación y calidad de software en DATEC se hará uso de la metodología Proceso Unificado Abierto (OpenUP) (9).

1.3.1. OpenUp

La metodología OpenUP presenta las mismas características de Proceso Unificado de Rational (RUP, por sus siglas en inglés, *Rational Unified Process*). Contiene el desarrollo iterativo, casos de uso y escenarios de conducción de desarrollo, gestión de riesgos y el enfoque centrado en la arquitectura (10).

OpenUp es un proceso de desarrollo de software de código abierto diseñado para pequeños equipos organizados que quieren tomar una aproximación ágil del desarrollo. Es un proceso iterativo que es Mínimo, Completo, y Extensible. Se valora la colaboración y el aporte de los *stakeholders*[1] sobre la formalidad y los entregables innecesarios (11).

OpenUP estructura el ciclo de vida de un proyecto en cuatro fases: inicio, elaboración, construcción y transición (Fig. 1). El ciclo de vida del proyecto brinda a los interesados un mecanismo de supervisión y

[1] Stakeholder: Usuario, grupo de personas, organización u otra entidad que posee un interés directo o indirecto en un sistema, que puede afectar o afectarse por las acciones u objetivos del mismo.

dirección para controlar los fundamentos del proyecto, su ámbito, la exposición a los riesgos, el aumento de valor y otros aspectos (10).

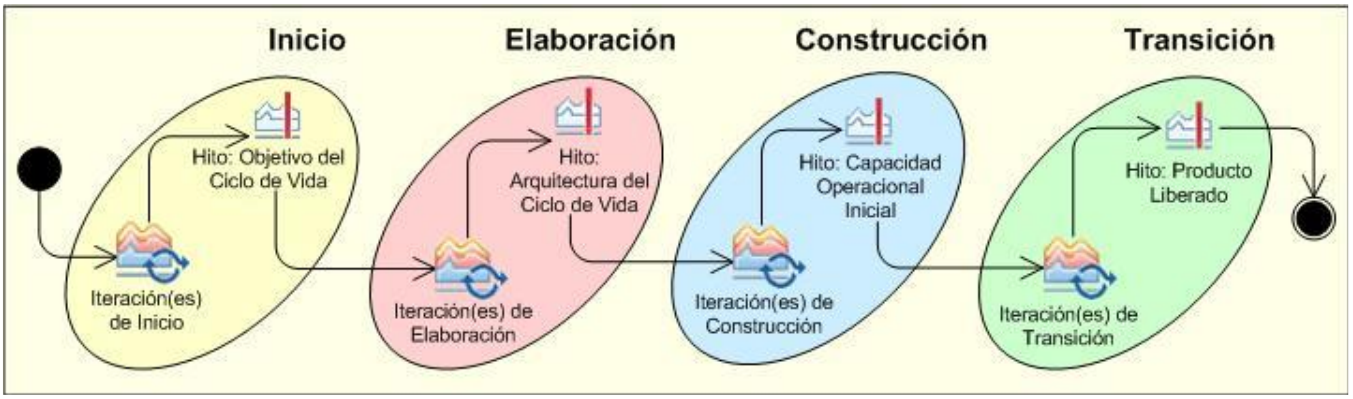


Fig. 1. Ciclo de vida de OpenUp (36)

Principios básicos de OpenUP (12):

- Colaboración para alinear los intereses comunes, difundir el conocimiento sobre el proyecto.
- Prioridades competitivas que maximicen el valor de los participantes en el proyecto.
- Evolucionar para obtener retroalimentación continua en aras de una mejora constante.
- Articulación de la arquitectura para minimizar riesgos y favorecer la planificación del desarrollo.

1.4. Lenguaje de Modelado

El Lenguaje de Modelación Unificado (UML, por sus siglas en inglés, *Unified Model Lenguaje*) surge a partir de la necesidad existente de un lenguaje gráfico, que estandarizara la manera de especificar y documentar un sistema de software. Y que además, incluyera aspectos conceptuales tales como procesos de negocios y funciones del sistema.

UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje interactivo de modelado de sistemas orientado a objetos de notación para el desarrollo de software (13). Se caracteriza por visualizar, especificar, construir y documentar los artefactos de un sistema. Está compuesto de diversos elementos gráficos que se combinan para conformar diagramas, así

como las reglas para combinar tales elementos. Estos diagramas, clasificados en estáticos y dinámicos, son usados para dar solución al problema que se plantea (9).

Este lenguaje como bien su nombre lo dice, modela los artefactos del sistema, facilitando así el entendimiento a las personas que no cuentan con conocimientos profundos de informática. Convirtiéndolo en un lenguaje de fácil comprensión, que a su vez posibilita una mejoría en la comunicación entre clientes y desarrolladores.

Beneficios que aporta este lenguaje (9):

- Mejor entendimiento del riesgo del proyecto antes de construir el sistema.
- Mejores tiempos totales de desarrollo (de 50% o más).
- Poder especificar la estructura y comportamiento del sistema y comunicarlo a todos los integrantes del proyecto.
- Documentar las decisiones de la arquitectura del proyecto.
- Obtener el plano del sistema.

1.5. Herramientas CASE

Las herramientas de Ingeniería de Software Asistida por Computadora (CASE, por sus siglas en inglés, *Computer Aided Software Engineering*), son un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software. Estas facilitan tareas como el diseño del proyecto, el cálculo de costos, la implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras (14).

Beneficios que aporta el uso de las herramientas CASE:

- Facilidad para la revisión de aplicaciones: Contribuyen a modificar el sistema por medio de las especificaciones más que por los ajustes al código fuente.
- Soporte para el desarrollo de prototipos de sistemas: Permiten que los ajustes necesarios al diseño se realicen con rapidez para alterar la presentación y las características de la interface.

- Generación de código: Se disminuye el tiempo de preparación de un programa, asegurando además una estructura estándar y consistente para este. Esto influye en el mantenimiento y disminuye la ocurrencia de errores, garantizando de este modo una mayor calidad en el producto.
- Mejora en la habilidad para satisfacer los requerimientos del usuario: Las descripciones mediante los diagramas y prototipos contribuyen a un intercambio de ideas más efectivo con los usuarios.
- Soporte interactivo para el proceso de desarrollo: Las herramientas CASE soportan pasos interactivos, anticipando que los detalles del sistema serán revisados con mayor frecuencia y consistencia.

Existen un variado número de herramientas CASE, cada una con sus propias particularidades, que van desde la complejidad, variedad de opciones, la dedicación a diferentes tipos de proyectos hasta los costos; algunas de estas son *Rational Rose, Umbrello, Argo UML, Visual Paradigm for UML.*

A continuación se detallan los aspectos fundamentales de la herramienta CASE *Visual Paradigm*.

1.5.1. Visual Paradigm

Visual Paradigm fue creada para el ciclo vital completo del desarrollo del software, automatizándolo y acelerándolo, partiendo de la captura de requisitos, análisis, diseño e implementación, permitiendo la generación de código, ingeniería inversa y generación de informes (7). Tiene dentro de sus características que es multiplataforma, portable y posee gran facilidad de uso. Presenta licencia gratuita y comercial. Su diseño está centrado a casos de uso y enfocado al negocio que genera un software de mayor calidad (15).

Tiene la capacidad de crear el esquema de clases a partir de una base de datos y a la vez crear la definición de base de datos a través del esquema de clases. Incorpora soporte para el trabajo en equipo, lo que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros (16).

Teniendo en cuenta la capacidad que presenta Visual Paradigm para representar todo tipo de diagramas durante el proceso de desarrollo de un software, que su diseño se centra en casos de uso, su portabilidad

y facilidad de uso, se elige esta herramienta de modelado en su versión 8.0 para el desarrollo de la aplicación. Por otro lado, es la herramienta definida dentro de la línea del centro.

1.6. Lenguajes de Programación

Un lenguaje de programación es un idioma que constituye el sistema de comunicación entre el hombre y la máquina, mediante el cual se transmiten a ésta las instrucciones en un formato comprensible. Los lenguajes de programación se clasifican en lenguajes de bajo nivel o alto nivel. Los lenguajes de bajo nivel se aproximan al código maquina mientras que los de alto nivel se asemejan más al lenguaje natural o humano (17).

A partir del aumento de las necesidades de las plataformas y la demanda de los usuarios, han surgido diferentes lenguajes de programación específicos para el desarrollo web. Inicialmente este problema con las plataformas y las demandas era solucionado mediante lenguajes estáticos. Con el avance de nuevas tecnologías aparecen nuevos problemas, y con estos, nuevos lenguajes dinámicos que los solucionarán. Estos lenguajes permitirían una interacción con los usuarios y el uso de sistemas de Bases de Datos.

En la actualidad existen dos grupos fundamentales de lenguajes para programación Web. Un grupo relativo a los que son utilizados del lado del cliente y en el otro se encuentran los del lado del servidor.

Del lado del cliente se encuentran, principalmente Java Script y Visual Basic Script (VBScript). Las páginas creadas con estos lenguajes son enviadas al usuario, de forma que el navegador es el encargado de interpretar y mostrarlas por pantalla. Lógicamente, es necesario prestar especial atención al navegador utilizado; aunque todos deben estar basados en los estándares del W3C *Consortium*[2], no dejan de estar presentes los posibles problemas en la visualización. Siendo así, lo que funciona con un navegador puede no hacerlo con otro.

Por otra parte, entre los lenguajes del lado del servidor se encuentran algunos como PHP, ASP, JSP, PERL. En conjunto son lenguajes reconocidos, ejecutados e interpretados por el propio servidor. Estos se

[2] W3C Consortium: Comité de generación de estándares para la Word Wide Web, similar a la ISO, se encarga de generar versiones estándares de HTML, SGML (*Standard Generalized Mark-up Lenguaje*), XML (*Extensible Markup Language*), XHTML, las plantillas de estilos CSS (*Cascade Style Sheets*).

encargan del control funcional del sistema, desarrollo de la lógica del negocio, de la conexión con las bases de datos, el procesamiento de la información y el envío de los resultados al cliente en un formato comprensible para él.

Para la implementación del sistema propuesto se decide utilizar JavaScript como lenguaje de programación del lado del cliente y PHP del lado del servidor. Los siguientes subepígrafes abordan aspectos fundamentales de estos lenguajes.

1.6.1. Lenguaje JavaScript

El gran desarrollo de la web es producto de lenguajes de programación como JavaScript, en cuanto a la posición y presentación de los contenidos es el avance más significativo en páginas web dinámicas y exactas (18).

Este es un lenguaje interpretado, por lo que para la ejecución de los programas no es necesaria la compilación. La mayoría de los navegadores en sus últimas versiones interpretan el código JavaScript integrado en las páginas web, siendo estos los encargados de interpretar y ejecutar los códigos escritos en este lenguaje. JavaScript cuenta con una implementación del Modelo en Objetos para la Representación de Documentos (DOM, por sus siglas en inglés, *Document Object Model*), para la interacción con las páginas.

JavaScript se utiliza principalmente en su forma del lado del cliente y ha sido implementado para permitir mejoras en la interfaz de usuario y páginas web dinámicas. Guarda parecidos con otros lenguajes como C o PHP, tanto en su formato como en su sintaxis. Aunque por supuesto, tiene sus propias características definitorias:

- Lenguaje de guiones: Puede realizar diversas tareas y funcionalidades que le sean especificadas al introducir su código fuente dentro de los elementos HTML.
- Orientado a Objetos: No distingue entre tipos de objetos. La herencia se realiza a través del mecanismo de prototipado y los métodos y propiedades pueden ser añadidos a cualquier objeto dinámicamente.
- Manejador de eventos: esto permite que durante el tiempo que el usuario visualice la página con el script en cuestión, JavaScript podrá reaccionar ante cualquier evento que se le indique.

- Independiente de plataforma: un script escrito en cualquier plataforma podrá ser ejecutado en otras. (19)

1.6.2. Lenguaje PHP 5.4

PHP (acrónimo recursivo de *Hypertext Preprocessor*) es un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios Web, puede ser incrustado en HTML. Es un lenguaje híbrido que toma las mejores características de otros idiomas para crear un lenguaje potente, de fácil uso y aprendizaje. A esto se añaden valores como el hecho de ser un proyecto de código abierto, gratuito y multiplataforma. Convirtiéndolo en el favorito para una amplia comunidad de programadores (20).

Una de las versiones más actualizada de esta tecnología es PHP 5.4. Entre sus principales características destacan su rapidez, facilidad de aprendizaje; al igual que las anteriores versiones presenta soporte multiplataforma tanto de diversos sistemas operativos, como servidores HTTP y de bases de datos.

PHP 5.4 permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres y Oracle. Además cuenta con la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como Linux, Mac OS X y Windows. Presenta un módulo para Apache, por lo que permite la interacción con este servidor web.

Este lenguaje ofrece las siguientes ventajas (21):

- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con PostgreSQL.
- Posee una amplia documentación en su página oficial, sobresaliendo la explicación y ejemplificación de todas las funciones del sistema en un único archivo de ayuda.
- Permite el manejo de excepciones (20).
- Capacidad de expandir su potencial utilizando módulos.
- Es un lenguaje multiplataforma, caracterizado por su rapidez y fácil aprendizaje.

- Programación segura y confiable ya que el código fuente es invisible tanto para el navegador como para el cliente porque es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador.
- Posee interfaces distintas para cada tipo de servidor.
- Permite aplicar técnicas de programación orientada a objetos. Clases y herencia.
- Posee una biblioteca nativa de funciones sumamente amplia e incluida.

1.7. Marcos de trabajo

Un marco de trabajo (*framework*) es un conjunto de componentes personalizables e intercambiables estructurados con el fin de acelerar el proceso de desarrollo de nuevos sistemas, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. Un *framework* puede ser considerado como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta (22).

1.7.1. Symfony 2.2

Symfony es uno de los frameworks para PHP más populares entre los usuarios y las empresas en la actualidad. Su objetivo es optimizar el desarrollo de aplicaciones web, permitiendo a los programadores crear código de más calidad y más fácil de mantener. Está basado en el paradigma de la programación orientada a objetos (10).
Utiliza el patrón arquitectónico Modelo-Vista-Controlador (MVC), separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación. Es independiente de la plataforma y el sistema gestor de bases de datos, siendo compatible con la mayoría de estos últimos (23).

Symfony fue diseñado para ajustarse a las siguientes características (12):

- Fácil de instalar y configurar en la mayoría de plataformas.
- Doctrine es un mapeo objeto-relacional (ORM, por sus siglas en inglés, *Object-Relational mapping*) que incluye Symfony, el cual permite manejar la información de la base de datos como si fueran objetos de PHP, disminuyendo el esfuerzo y tiempo de desarrollo del sistema.
- Utiliza programación orientada a objetos.
- Sigue la mayoría de las mejores prácticas y patrones de diseño para la web.

- Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.

1.7.2. ExtJS 4.1

ExtJS es un *framework* de JavaScript que permite el desarrollo de aplicaciones enriquecidas para Internet del lado del cliente. Permite crear aplicaciones complejas utilizando componentes ya definidos y flexibiliza el manejo de componentes de la página como el DOM y peticiones AJAX (24).

Principales características de ExtJS:

- Alto rendimiento en ejecución debido a la optimización de código JavaScript.
- Controles de usuario personalizables.
- Modelo orientado a componentes, bien diseñado y extensible.
- Posee una Interfaz de Programación de Interface (API) intuitiva y fácil de utilizar.
- Es distribuido bajo licencias de código abierto y comercial.

1.8. Entorno de Desarrollo Integrado NetBeans

NetBeans es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Es un producto libre y gratuito sin restricciones de uso (20).

La Plataforma NetBeans permite programar en distintos lenguajes, siendo ideal para el lenguaje Java, además ofrece un excelente entorno para la programación en PHP. Incluye soporte para Symfony y provee una estructura para los proyectos, propone un esqueleto para organizar el código fuente y el editor conjuntamente integra los lenguajes como HTML, JavaScript, CSS y PHP (25).

1.9. Sistema Gestor de Bases de Datos

Un Sistema Gestor de Base de Datos (SGBD), es una colección de programas que proporcionan al usuario los medios necesarios para garantizar todas las características de una Base de Datos (9). Ayuda en la definición y manipulación de los datos, controla la seguridad y privacidad de estos, manteniendo la integridad de los mismos dentro de la base de datos.

1.9.1. PostgreSQL 9.1

PostgreSQL destaca entre los gestores de bases de datos de código abierto más avanzados en la actualidad. Puede ser utilizado, modificado y distribuido gratuitamente, ya sea con fines privados, comerciales o académicos (20). Ofrece control de concurrencia multiversión, permitiendo que mientras un proceso escribe en una tabla, otros puedan acceder a la misma sin necesidad de utilizar bloqueos. Cumple completamente con las características ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) para realizar transacciones seguras. Posee interfaces nativas para varios lenguajes entre los que se encuentra PHP, lo que lo convierte en multiplataforma (26).

PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, soporte multiusuario, transacciones, optimización de consultas y arreglos (20).

Otras características que presenta PostgreSQL (15):

- Copias de seguridad en línea.
- Replicación asíncrona.
- Transacciones anidadas.
- Integridad referencial.

Partiendo de los beneficios que presenta PostgreSQL y en correspondencia con lo establecido en la arquitectura del centro DATEC, además de los resultados medibles y probados en la mayoría de los proyectos del centro, se decide utilizarlo en su versión 9.1 como Gestor de Bases de Datos para el sistema que a implementar.

1.9.2. Aplicación gráfica para gestionar PostgreSQL

PgAdmin III es una aplicación gráfica para administrar el gestor de bases de datos PostgreSQL, destacando entre las más completas y populares bajo licencia de código abierto (27). Su diseño permite a los usuarios escribir desde simples consultas SQL hasta el desarrollo de complejas bases de datos. Su interfaz gráfica facilita la administración debido a que soporta todas las características de PostgreSQL. También incluye un editor SQL y un editor de código de la parte del servidor. Permite que la conexión al servidor pueda hacerse mediante conexión TCP/IP. Por las facilidades que ofrece esta herramienta, fue seleccionada para la administración del gestor de bases de datos que se utilizará en la aplicación.

1.10. Servidor Web Apache

Apache es el servidor web por excelencia, su flexibilidad en la configuración, la robustez y estabilidad lo hacen muy confiable para millones de servidores en el mundo. La licencia Apache desciende de la BSD. Esta licencia permite hacer lo que se pretenda con el código fuente con la única condición de que sea reconocido su trabajo, esto le da transparencia al software de manera que se pueda saber que se instala como servidor, sin secretos ni puertas traseras.

Entre sus características sobresalen (20)

- Es prácticamente universal, al poder correr en una multitud de sistemas operativos.
- Gran extensibilidad, se pueden añadir módulos para ampliar las capacidades con que cuenta Apache. Existe una rica variedad de módulos, que permiten desde generar contenido dinámico con PHP, Java o Perls, hasta monitorizar el rendimiento del servidor.
- Apache trabaja con lenguajes como Perl, PHP y otros lenguajes de script.
- Cuenta con mensajes de errores altamente configurables.

1.11. Conclusiones parciales del capítulo

A modo de conclusión, en el presente capitulo se caracterizaron los elementos fundamentales a utilizar en el desarrollo y cumplimiento de los objetivos planteados. Se profundizó en el estudio de los Sistemas de Información, determinando la necesidad de implementación de un Sistema de Información como solución al problema planteado. Basado en los resultados arrojados por la investigación, se decide utilizar como metodología de desarrollo OpenUp y como herramienta de modelado Visual Paradigm 6.4. Para el desarrollo del sistema se utilizará JavaScript como lenguaje de programación del lado del cliente y PHP 5.4 del lado del servidor. Fueron seleccionados ExtJS 4.1 y Symfony 2.2 como marcos de trabajo para el desarrollo del sistema de información. Además, por el soporte para los lenguajes de programación y marcos de trabajo seleccionados, se utilizará NetBeans 7.2 como entorno integrado de desarrollo. Como gestor de base de datos se eligió PostgreSQL 9.1 y para su administración la herramienta PgAdminIII. El servidor Web que se decidió utilizar es el Apache por su potencia, fiabilidad y compatibilidad con PHP y PostgreSQL.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA SOLUCIÓN PROPUESTA

En este capítulo se enfatiza en la propuesta de solución para el problema de la investigación. Mediante el modelo de dominio se describe la estructura que se desea representar en el sistema a desarrollar. Luego de profundizar los conceptos fundamentales que describen el dominio de la aplicación, se establecen los requisitos funcionales y no funcionales, agrupándolos para la obtención del diagrama de casos de uso. De igual forma se diseñarán los diagramas de clases y de interacción para los casos de uso arquitectónicamente significativos. Se describen los patrones de diseño utilizados en la solución y la distribución física del sistema mediante el modelo de despliegue.

2.1. Modelo de Dominio

El modelo de dominio facilita la comprensión del entorno de trabajo con el cual el sistema va interactuar. Captura y expresa el entendimiento alcanzado y puede ser utilizado como punto de partida para el diseño del sistema.

A continuación se presenta el Modelo de Dominio (Fig. 2) del Sistema de Información para la gestión de la formación y calidad de software en DATEC, además de una descripción de sus conceptos.

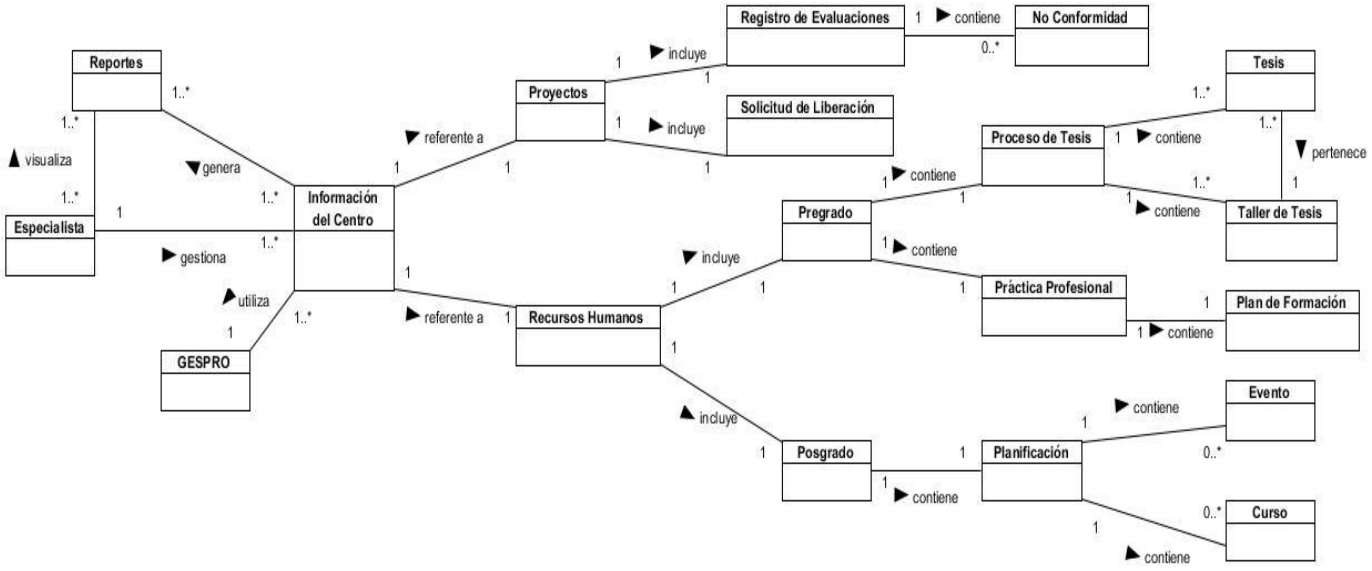


Fig. 2. Modelo de Dominio

Conceptos del Modelo de Dominio:

Especialista: Entidad que representa al especialista que utiliza el sistema acorde a los permisos asignados y visualiza reportes. Los especialistas gestionan la información generada en el centro formando parte de esta información.

Reportes: Entidades que representan los archivos generados por el sistema a partir de la información obtenida de la base de datos.

Información del Centro: Entidad que representa la información que es generada por los procesos de las áreas de Calidad de Proyecto, Investigación y Posgrado, Planificación y Formación Profesional.

GESPRO: Sistema que gestiona la información de los proyectos y recursos humanos del centro, el especialista hace uso de este sistema para gestionar información del centro.

Proyectos: Entidad que representa la información relacionada con el proceso de calidad de los proyectos existentes en el centro durante todo el ciclo de vida de los mismos.

Registro de Evaluaciones: Entidad que representa el documento que archiva todas las evaluaciones realizadas a un proyecto durante el ciclo de vida de este. Estas evaluaciones se clasifican en auditorias o revisiones.

No conformidad: Entidad que representa las no conformidades, estas son errores detectados durante las evaluaciones y archivadas para posteriormente realizarle el seguimiento.

Solicitud de Liberación: Entidad que representa el documento necesario para solicitar la liberación de un proyecto.

Recursos Humanos: Entidad que representa la información relacionada con los trabajadores y estudiantes que pertenecen al centro**.**

Pregrado: Entidad que representa una especificación de la información de los recursos humanos, relacionada con el proceso de formación de un profesional durante toda la etapa de estudio de la disciplina.

Proceso de Tesis: Entidad que representa el proceso al cual se somete un estudiante como culminación de los estudios, en este proceso se lleva el control de las Tesis, Talleres de Tesis y las evaluaciones obtenidas por los estudiantes en cada uno de los talleres.

Tesis: Entidad que representa el trabajo de diploma al cual se enfrenta el estudiante al final de la carrera como culminación de los estudios. Las Tesis están compuestas por los tutores que son profesores que guían y apoyan el trabajo de los autores que son estudiantes. Cada Tesis pertenece a un Taller de Tesis, en el cual se evaluará el trabajo desarrollado.

Taller de Tesis: Entidad que representa los talleres de tesis, los cuales están compuestos por un grupo de profesores calificados que actúan como tribunal en la evaluación de las Tesis. Un Taller de Tesis puede contener varias Tesis diferentes.

Práctica Profesional: Entidad que representa la información relacionada con las prácticas profesionales que realizan los estudiantes durante el transcurso de la carrera.

Plan de Formación: Entidad que representa el documento que archiva el plan de formación del estudiante durante el transcurso del curso.

Posgrado: Entidad que representa una especificación de la información de los recursos humanos, relacionada con el proceso de investigación y posgrado. Incluye toda la información relacionada con la superación del profesional.

Planificación: Entidad que representa la planificación de actividades que tienen los profesionales durante su superación. Estas actividades están compuestas por eventos y cursos de adiestramiento.

Cursos: Entidad que representa los cursos que debe culminar el profesional en el proceso de superación. Existe una cantidad obligatoria de cursos necesarios, y otros cursos que pueden ser opcionales para el profesional.

Eventos: Entidad que representa los eventos en los cuales debe participar el profesional de acuerdo a su planificación.

2.2. Especificación de los requisitos del sistema

Los requisitos de un sistema describen las condiciones o capacidades que deben ser alcanzadas por un sistema, necesarias para un usuario resolver un problema o lograr un objetivo. Son propiedades o restricciones determinadas de forma precisa que deben satisfacerse. Se pueden clasificar en: funcionales y no funcionales (28). Partiendo de la descripción de las clases representadas en el Modelo de Dominio y las necesidades del cliente, fueron determinados los requisitos funcionales y no funcionales del sistema.

2.2.1. Requisitos funcionales

Los requisitos funcionales definen el comportamiento interno de un software, son condiciones que el sistema ha de cumplir. Están encaminados a describir qué es lo que el sistema debe hacer (28). Se identificaron los siguientes requisitos funcionales:

RF1 Insertar usuario.
RF2 Importar usuario.
RF3 Editar usuario.
RF4 Eliminar usuario.
RF5 Visualizar usuario.
RF6 Autenticar usuario.
RF7 Asignar rol a usuario.
RF8 Crear notificación.
RF9 Visualizar notificación.
RF10 Eliminar notificación.
RF11 Insertar proyecto.
RF12 Importar proyecto.
RF13 Editar proyecto.
RF14 Eliminar proyecto.
RF15 Visualizar proyecto.
RF16 Insertar registro.
RF17 Editar registro.
RF18 Insertar nuevo cambio.
RF19 Visualizar control de cambios.
RF20 Insertar Evaluación.
RF21 Editar Evaluación.
RF22 Eliminar Evaluación.
RF23 Mostrar Evaluación.
RF24 Insertar No Conformidad.
RF25 Editar No Conformidad.
RF26 Eliminar No Conformidad.

RF27 Visualizar No Conformidades.

RF28 Realizar seguimiento a las No Conformidades.

RF29 Solicitar liberación de proyecto.

RF30 Insertar Planificación del usuario.

RF31 Editar Planificación del usuario.

RF32 Eliminar Planificación del usuario.

RF33 Visualizar Planificación del usuario.

RF34 Chequeo del cumplimiento de la planificación.

RF35 Insertar datos del evento.

RF36 Insertar constancia del cumplimiento del evento.

RF37 Aprobar cumplimiento del evento.

RF38 Insertar cursos de adiestramiento.

RF39 Editar cursos de adiestramiento.

RF40 Eliminar cursos de adiestramiento.

RF41 Visualizar cursos de adiestramiento.

RF42 Asignar cursos de adiestramiento.

RF43 Seleccionar cursos de adiestramiento.

RF44 Insertar constancia del cumplimiento del curso de adiestramiento.

RF45 Aprobar cumplimiento del curso de adiestramiento.

RF46 Chequeo del cumplimiento de los cursos de adiestramiento.

RF47 Insertar plan de formación del estudiante.

RF48 Eliminar plan de formación del estudiante.

RF49 Visualizar plan de formación del estudiante.

RF50 Insertar evaluación mensual del estudiante.

RF51 Visualizar evaluación mensual del estudiante.

RF52 Insertar taller de tesis.

RF53 Editar taller de tesis.

RF54 Eliminar taller de tesis.

RF55 Visualizar taller de tesis.

RF56 Insertar Tesis.

RF57 Editar Tesis.

RF58 Eliminar Tesis.
RF59 Visualizar Tesis.
RF60 Insertar evaluación de tesis.
RF61 Visualizar evaluaciones de tesis.
RF62 Resumir proceso de formación.
RF63 Insertar Reporte.
RF64 Editar Reporte.
RF65 Eliminar Reporte.
RF66 Visualizar Reporte.
RF67 Exportar Reporte.

2.2.2. Requisitos no funcionales

Los requerimientos no funcionales constituyen cualidades o propiedades que el producto debe tener. No son parte de la razón fundamental del producto pero si son fundamentales en el éxito y la aceptación de este (28).

Se identificaron los siguientes requisitos no funcionales:

Requisitos de Usabilidad:

RNF1 Tipo de Aplicación Informática: El componente deberá ser una aplicación web.
RNF2 De acuerdo a los estándares de diseño la interfaz debe simular un entorno de escritorio que facilite el trabajo con la aplicación a los usuarios del sistema.

Requisitos de Portabilidad:

RNF3 El Sistema será multiplataforma (Windows y Linux).

Requisitos de Seguridad:

RNF4 Los datos almacenados en la base de datos deberán estar protegidos ante accesos no autorizados utilizando para ello mecanismos de autenticación así como encriptación de las contraseñas.

RNF5 La conexión entre la PC Cliente y el Servidor deberá ser mediante el protocolo seguro de transferencia de hipertexto (HTTPS, por sus siglas en inglés, *Hypertext Transfer Protocol Secure*), evitando que la información sensible pueda ser usada por un atacante que haya conseguido interceptar la transferencia de datos.

RNF6 Requisitos de Eficiencia:

- El sistema debe permitir la concurrencia de al menos 100 usuarios.
- El tiempo promedio de respuesta del sistema debe ser aceptable (entre 2 y 8 segundos).

RNF7 Requisitos de Software.

Servidor Web:

- Sistema operativo: Multiplataforma (Windows y Linux).
- Servidor web Apache 2.0 o versión superior.

Servidor de Base de Datos:

- Sistema operativo: Multiplataforma (Windows y Linux).
- Sistema Gestor de Base de Datos: PostgreSQL 9.1.
- PostgreSQL debe estar correctamente configurado para aceptar conexiones vía TCP/IP.

PC Cliente:

- Sistema operativo: Multiplataforma (Windows y Linux).
- Navegador web: Mozilla Firefox 22.0 o versión superior.

RNF8 Requisitos de Hardware.

Servidor Web:

- Microprocesador: Intel Pentium (IV) 1.7 GHz.
- Memoria RAM: 1 GB o superior.
- 10 GB de espacio en disco duro.

Servidor de Base de Datos:

- Microprocesador: Intel Pentium (IV) 1.7 GHz.
- Memoria RAM: 1 GB o superior.
- 80 GB de espacio en disco duro.

PC Cliente:

- Microprocesador: Intel Pentium (IV) 1.7 GHz.

- Memoria RAM: 256MB o superior.

Requisitos de Interfaz:

RNF9 Restricciones de Diseño e Implementación

- El sistema deberá ser implementado en los lenguajes de programación PHP y JavaScript.
- Se utilizarán los marcos de trabajo de desarrollo Symfony y ExtJS.
- Se empleará el entorno de desarrollo NetBeans 7.2 y el sistema gestor de base de datos PostgreSQL 9.1.

2.3. Diagrama de casos de uso del sistema

El Diagrama de casos de uso del sistema se centra fundamentalmente en estructurar los requisitos funcionales mediante casos de uso; definiendo su relación con los actores que interactúan con el sistema.

Actores de Sistema

Los actores de un sistema son agentes externos que interactúan con él, ya sean personas u otros sistemas. Además son generalmente responsables de realizar actividades que serán automatizadas en el sistema. A continuación se describen los actores del sistema (ver Tabla 1) y se realiza la representación UML de los mismos (Fig. 3).

Tabla 1: Actores del Sistema

Actor	Descripción
Usuario	Es una generalización de los actores del sistema que contiene la funcionalidad de autenticarse en el mismo.
Administrador	Es el administrador primario del sistema, encargado de administrar usuarios y asignar los roles.
Administrador de la Calidad	Es el encargado de gestionar la información referente al proceso de Calidad de Proyecto. El Administrador de la Calidad gestiona los proyectos, sus registros de evaluaciones, las evaluaciones en sí y las no conformidades encontradas.

Administrador de Investigación y Posgrado	Es el encargado de gestionar de manera general la información referente al proceso de Investigación y Posgrado. El Administrador de Investigación y Posgrado gestiona los eventos, cursos y realiza la gestión de la planificación de los usuarios.
Administrador de Formación Profesional	Es el encargado de gestionar de manera general la información referente al proceso de Formación Profesional. El Administrador de Formación Profesional gestiona las tesis, los talleres de tesis y realiza la gestión del proceso de tesis y la práctica profesional.
Especialista	Es el Usuario con permisos básicos para el acceso y trabajo con el sistema. Se encarga de la administración del plan de formación de los estudiantes y sus evaluaciones. En su mayoría estará compuesto por trabajadores del centro.
Estudiante	Es el Usuario con permisos bajos para el acceso y trabajo con el sistema. En su mayoría estará compuesto por estudiantes que pertenecen al centro.
GDR	El Sistema de Información utiliza los servicios del Sistema Generador Dinámico de Reportes para la obtención de nueva información mediante los reportes.

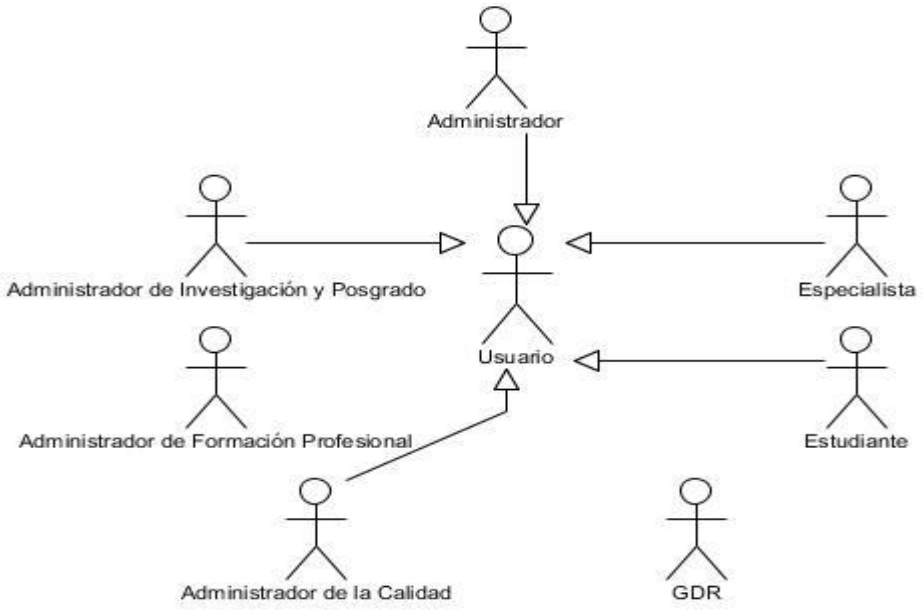


Fig. 3. Actores del Sistema

Diagrama de Casos de Uso

Los Casos de Uso (CU) describen el comportamiento del sistema mediante acciones y reacciones desde el punto de vista del usuario. Son el conjunto de operaciones o tareas específicas que se realizan tras una orden de un actor o la invocación de otro caso de uso. Describen la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para realizar una tarea específica (25).

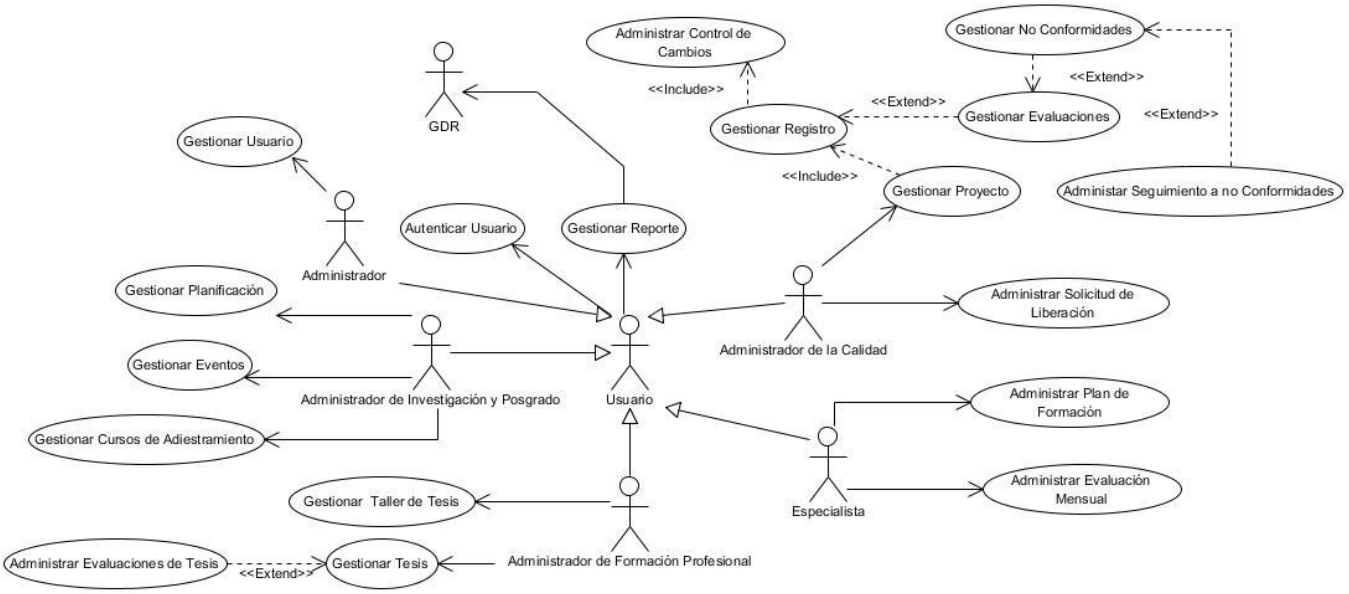


Fig. 4. Diagrama de Casos de Uso del Sistema

2.3.1. Patrones de Caso de Uso utilizados en la solución

Los patrones de caso de uso capturan mejores prácticas para modelar casos de uso. Entre los beneficios del uso de los patrones se encuentran el aumento de la productividad, la reutilización de elementos existentes y la disminución de tiempo invertido en resolver problemas ya resueltos. En la solución son empleados los siguientes patrones:

CRUD Completo

El patrón CRUD Completo propone formar un caso de uso a partir de los requisitos funcionales relacionados con las acciones de altas, bajas, cambios y consultas a alguna entidad del sistema. Ejemplo

de ello se evidencia en los casos de usos Gestionar Usuario, Registro, Taller de Tesis, Tesis representados en el diagrama de caso de uso del sistema.

CRUD Parcial

El patrón CRUD Parcial propone formar un caso de uso a partir de los requisitos funcionales relacionados con algunas de las acciones de un CRUD Completo. Ejemplo de ello se evidencia en los casos de usos Administrar Plan de Formación, Administrar Evaluación Mensual, Administrar Evaluaciones de Tesis, Administrar Control de Cambios y Administrar Seguimiento a No Conformidades representados en el diagrama de caso de uso del sistema.

Extensión Concreta o Inclusión

Extensión concreta: Consiste en una relación extendida entre dos casos de uso. Este patrón se aplica cuando un flujo puede extender el flujo de otro caso de uso así como ser realizado en sí mismo. Ejemplo de ello se evidencia en la figura que a continuación se muestra:

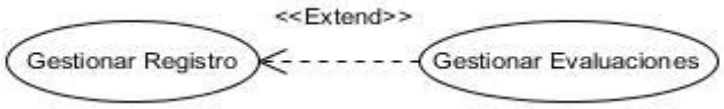


Fig. 5. Extensión Concreta

Inclusión concreta: Se incluye una relación del caso de uso base al caso de uso de inclusión. El último puede ser instalado en sí mismo. El caso de uso base puede ser concreto o abstracto. Ejemplo de ello se evidencia en la figura que a continuación se muestra:

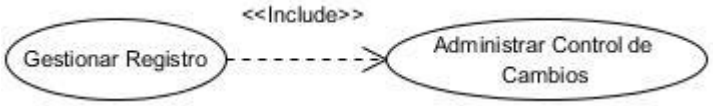


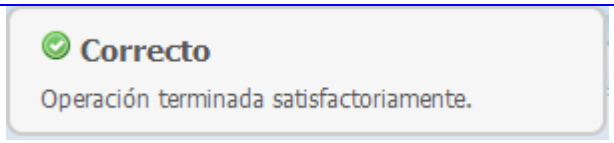
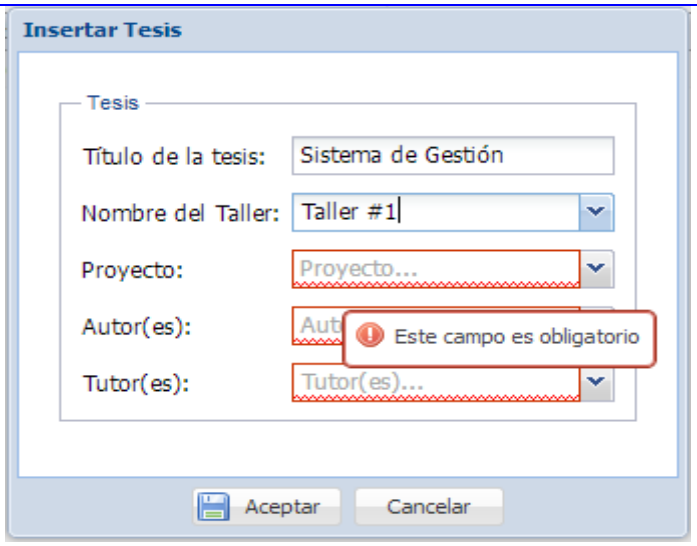
Fig. 6. Inclusión Concreta

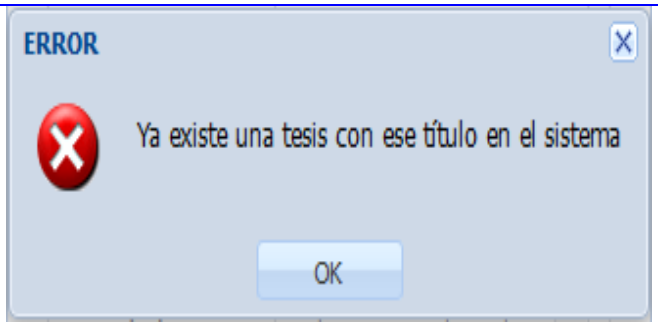
2.3.2. Especificación del CU Gestionar Tesis

Tabla 2: Especificación del CU Gestionar Tesis

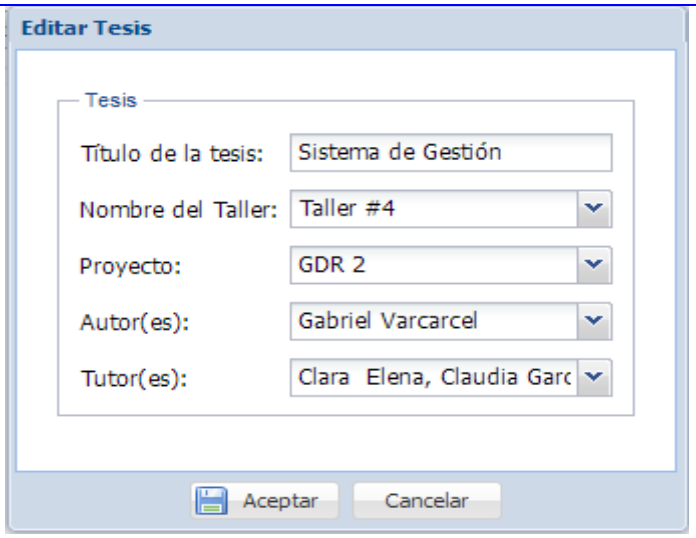
Caso de Uso:	Gestionar Tesis
Actores:	Administrador de Formación y Posgrado
Resumen:	El caso de uso se inicia cuando el actor selecciona una de las opciones asociadas al caso de uso (insertar, editar, eliminar y visualizar las tesis). El caso de uso termina una vez finalizada alguna de las opciones mencionadas anteriormente.
Precondiciones:	El sistema debe estar instalado y ejecutándose correctamente. El actor debe estar autenticado con los permisos necesarios.
Referencias	RF56, RF57, RF58, RF59
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el actor Administrador de Formación y Posgrado selecciona la pestaña Gestión de Tesis.	2. El sistema muestra el listado de las tesis y activa las opciones correspondientes a la gestión de tesis. El actor Administrador de Formación y Posgrado puede: -Insertar Tesis, ver sección "**Insertar Tesis**". -Editar Tesis, ver sección "**Editar Tesis**". -Eliminar Tesis, ver sección "**Eliminar Tesis**". -Visualizar Tesis, ver sección "**Visualizar Tesis**".
Sección "Insertar Tesis"	
Acción del Actor	**Respuesta del Sistema**
1. El actor Administrador de Formación y Posgrado selecciona la opción Insertar.	2.El sistema muestra la interfaz para insertar tesis, solicitando los siguientes datos: - Título. - Proyecto. - Taller. - Autor. - Tutor.
3. El actor Administrador de Formación y Posgrado introduce los datos correspondientes.	4. El sistema valida los datos introducidos, luego de la validación activa el botón aceptar.

5. El actor Administrador de Formación y Posgrado selecciona la opción aceptar.	6. El sistema adiciona la tesis, recarga el listado de tesis y muestra el mensaje “Operación terminada satisfactoriamente”, terminando así el caso de uso.
Prototipo de Interfaz	
	
Flujo Alterno al paso 4 “ Validación”	
Acción del Actor	**Respuesta del Sistema**
	4a. El sistema muestra los campos con errores de validación subrayados en rojo.
4b. El actor Administrador de Formación y Posgrado se posiciona sobre el campo que tiene errores en los datos.	4c. El sistema muestra el mensaje de error de acuerdo al campo donde se posicione el actor Administrador de Formación y Posgrado.
Prototipo de Interfaz	
	
Flujo Alterno al paso 1 ,5 “ Cancelar”	
Acción del Actor	**Respuesta del Sistema**
a. El actor Administrador de Formación y Posgrado decide no adicionar una tesis y cancela la operación.	b. El sistema cancela la operación y cierra la interfaz correspondiente a la adición de tesis.

Flujo Alterno al paso 6 “ Error al insertar tesis”	
Acción del Actor	**Respuesta del Sistema**
	6a. En caso de que exista alguna tesis con el mismo título del que se intenta adicionar el sistema muestra el mensaje de error “Ya existe una tesis con ese título en el sistema”.
Prototipo de Interfaz	
	
Sección “Editar Tesis”	
Acción del Actor	**Respuesta del Sistema**
1. El actor Administrador de Formación y Posgrado selecciona la tesis que desea editar del listado de tesis que se muestra y selecciona la opción editar tesis.	2. El sistema muestra una interfaz con los datos de la tesis seleccionada previamente.
3. El actor Administrador de Formación y Posgrado realiza los cambios correspondientes en los campos de la interfaz mostrada previamente por el sistema.	4. El sistema valida los campos editados, luego de la validación activa el botón editar.
5. El actor Administrador de Formación y Posgrado selecciona la opción aceptar.	6. El sistema edita la tesis seleccionada, recarga el listado para mostrar la actualización realizada y muestra el mensaje “Operación terminada satisfactoriamente”, terminando así el caso de uso.

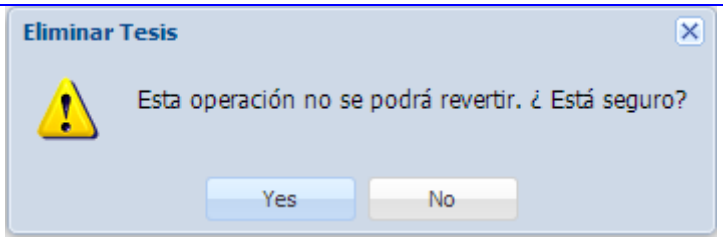
Prototipo de Interfaz



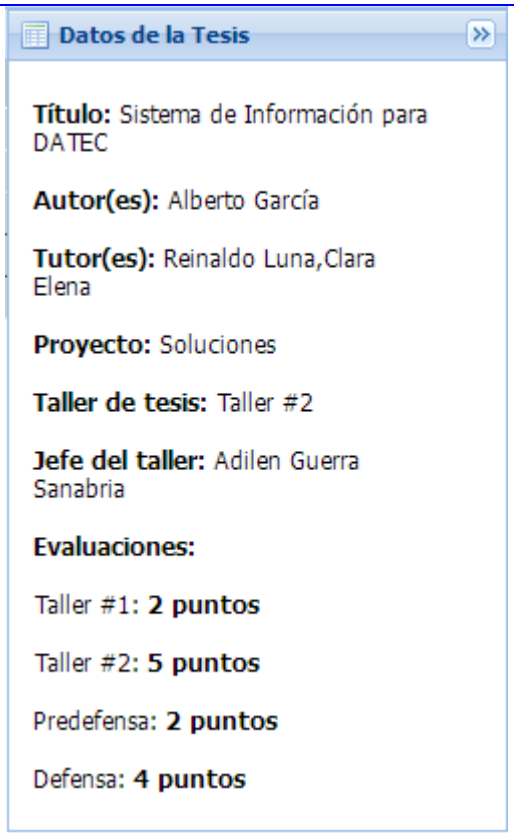
Flujo Alterno al paso 4 " Validación"

Acción del Actor	Respuesta del Sistema
	4a. El sistema muestra los campos con errores de validación subrayados en rojo.
4b. El actor Administrador de Formación y Posgrado se posiciona sobre el campo que tiene errores en los datos.	4c. El sistema muestra el mensaje de error de acuerdo al campo donde se posicione el actor Administrador de Formación y Posgrado.

Prototipo de Interfaz

Flujo Alterno al paso 1 ,5 “ Cancelar”	
Acción del Actor	**Respuesta del Sistema**
a. El actor Administrador de Formación y Posgrado decide no editar la tesis y cancela la operación.	b. El sistema cancela la operación y cierra la interfaz correspondiente a editar tesis.
Sección “Eliminar Tesis”	
Acción del Actor	**Respuesta del Sistema**
1. El actor Administrador de Formación y Posgrado selecciona la tesis que desea eliminar del listado de tesis y selecciona la opción eliminar tesis.	2. El sistema muestra una interfaz preguntando “Esta operación no se podrá revertir ¿Está seguro?”
3. El actor Administrador de Formación y Posgrado selecciona la opción sí del cuadro de diálogo mostrado por el sistema.	4. El sistema elimina la tesis seleccionada, recarga el listado de tesis y muestra el mensaje “Operación terminada satisfactoriamente”, terminando así el caso de uso.
Prototipo de Interfaz	
	
Flujo Alterno al paso 3 “ Selección de la opción no”	
Acción del Actor	**Respuesta del Sistema**
3a. El actor Administrador de Formación y Posgrado decide no realizar la operación de eliminar y selecciona la opción no.	3b. El sistema cierra la interfaz mostrada en el paso 2, terminando así el caso de uso.
Sección “Visualizar Tesis”	
Acción del Actor	**Respuesta del Sistema**
1. El actor Administrador de Formación y Posgrado selecciona una tesis del listado realizando un doblecilck sobre esta.	2. El sistema muestra una interfaz con el listado de los datos relacionados a la tesis seleccionada, terminando así el caso de uso.

Prototipo de Interfaz



2.4. Modelo de Diseño

El Modelo de Diseño se utiliza para documentar el diseño de un sistema. Es un modelo de objeto que describe la realización de los casos de uso y sirve como una abstracción del Modelo de Implementación y del código fuente. Se utiliza como entrada esencial para las actividades en los flujos de trabajo Implementación y Prueba. Es un artefacto integral que abarca todas las clases del diseño y sus relaciones, e incluye los diagramas de clases y de interacción del diseño (29).

2.4.1. Diagrama de Clases del Diseño

El Diagrama de Clases es el diagrama principal para el análisis y diseño. Un diagrama de clases sirve para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser

asociativas, de herencia, de uso. La definición de clases incluye definiciones para atributos y operaciones (30). A continuación se muestra el diagrama de clases diseño del caso de uso Gestionar Tesis:

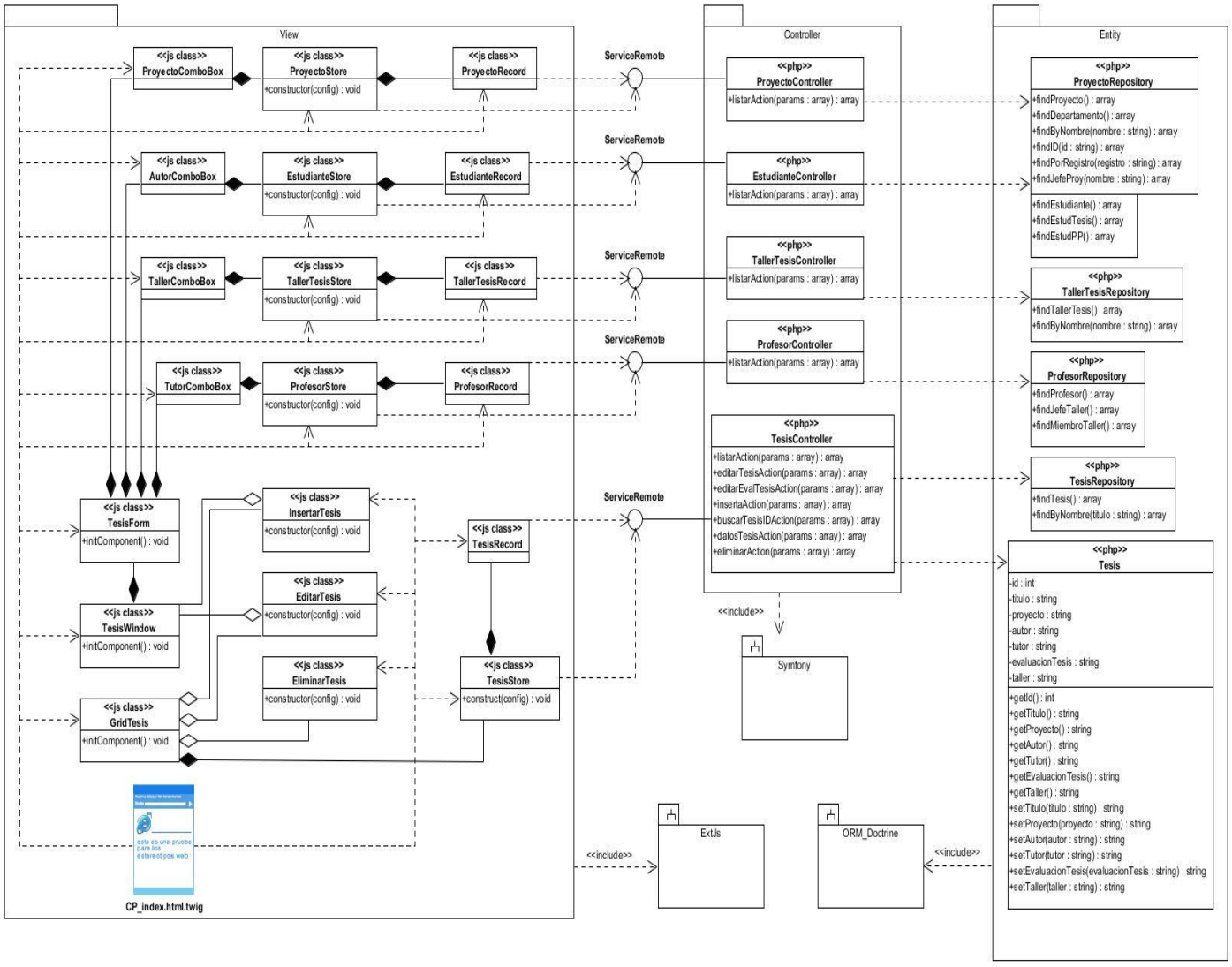


Fig. 7. Diagrama de clases del diseño del CU Gestionar Tesis

El Diagrama de clases del diseño (Fig. 7) incluye todas las clases con los métodos y atributos correspondientes al CU Gestionar Tesis. En este diagrama el paquete *View* contiene las clases de la que conforman la vista, que incluye componentes como formularios y ventanas, los cuales pertenecen al *framework* ExtJS. El paquete *Controller* contiene las clases del controlador y el paquete *Entity* las

relacionadas con el modelo, evidenciándose así el uso del patrón Modelo-Vista-Controlador. Todas las clases de la vista se encuentran incluidas en la plantilla CP_index.html.twig.

La clase GridTesis es la encargada de construir la tabla donde se muestran las tesis existentes en la base de datos, para lo que compone a la clase TesisStore que a su vez compone la clase TesisRecord, estas se encargan mediante un servicio remoto de conectar la tabla con la clase controladora. La clase GridTesis también agrega a las clases encargadas de las acciones que se realizan a las tesis existentes, estas clases son (InsertarTesis, EditarTesis, EliminarTesis).

Las clases InsertarTesis y EditarTesis agregan a la clase TesisWindow que compone la clase TesisForm siendo esta la que contiene el formulario para la acción seleccionada. TesisForm está compuesto por cuatro combobox que componen sus clases Store y Record respectivamente, además de garantizar la conexión del combobox con las controladoras necesarias.

Las clases controladoras contienen las funcionalidades requeridas por las clases Store y Record para el envío de los datos a las vistas. La Clase TesisControler es la encargada de las consultas a la base de datos con los métodos (listarAction, insertarAction, editarAction, eliminarAction, buscarTesisIDAction, datosTesisAction) y utilizando la clase del modelo TesisRepository con sus métodos (findTesis, findByNombre) para el trabajo con la clase Tesis. La clase Tesis representa la entidad Tesis en la base de datos y está compuesta por sus atributos y métodos.

2.4.2. Patrones de diseño utilizados en la solución

Patrones de diseño GRASP

Creador: El patrón creador asigna la responsabilidad de la creación de nuevos objetos o clases a la clase que contenga la información necesaria. De este depende que el diseño pueda soportar un bajo acoplamiento, mayor claridad, encapsulación y reutilización (31). Este patrón se utiliza en la adición de las Tesis, donde el controlador TesisController, es el encargado de crear instancias de la entidad Tesis y asigna valores a los atributos de dicha entidad.

Controlador: El patrón controlador intermedia entre una determinada interfaz y su algoritmo de implementación, de tal forma que recibe los datos y los envía a las distintas clases según el método llamado. Con mayor número de controladores aumenta la cohesión y disminuye el acoplamiento (31). En

la solución este patrón se pone de manifiesto en la utilización de diversos controladores como por ejemplo: UsuarioController, ProyectoController, RegistroController, TallerTesisController y TesisController.

Alta cohesión: En la solución se observa el uso de este patrón en las clases del modelo, Proyecto, Registro, ControlCambio, Evaluacion, NoConformidad, TallerTesis, Tesis y EvaluacionTesis las cuales poseen la estructura necesaria para almacenar la información de una manera coherente y de acuerdo a la responsabilidad que tenga asignada cada clase. Además todos sus elementos trabajan juntos para proporcionar un comportamiento bien delimitado.

Bajo acoplamiento: Es la idea de tener las clases lo menos ligadas entre sí para en caso de producirse una modificación en alguna de ellas, repercuta lo menos posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. En la solución se observa el uso de este patrón en las clases del modelo Usuario, Proyecto, TallerTesis, Tesis y Evaluacion.

Polimorfismo: Se debe hacer uso del polimorfismo siempre que las alternativas o comportamientos relacionados varían según el tipo (clase), se asigna la responsabilidad para el comportamiento utilizando operaciones polimórficas a los tipos para los que varía el comportamiento. En la solución este patrón se pone de manifiesto mediante la utilización de operaciones polimórficas en los controladores: UsuarioController, EstudianteController, ProfesorController y TecnicoController.

Patrones de diseño GOF

Decorador: Añade y elimina funcionalidades en tiempo de ejecución. Presenta una mayor flexibilidad que la herencia, evitando la excesiva utilización de esta y en algunos casos la herencia múltiple. En la solución se evidencia la utilización de este patrón cuando la clase base.html.twig, definida en el diagrama de clases del diseño, contiene todo el código HTML y a su vez decora todas las plantillas a usar, en dependencia de la petición del usuario.

2.5. Diagrama de Secuencia

Un diagrama de secuencia es modelado para cada CU y muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo. Contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario y mensajes intercambiados entre los objetos

(32). Un diagrama de secuencia del sistema es un artefacto que muestra los eventos de entrada y salida de una aplicación.

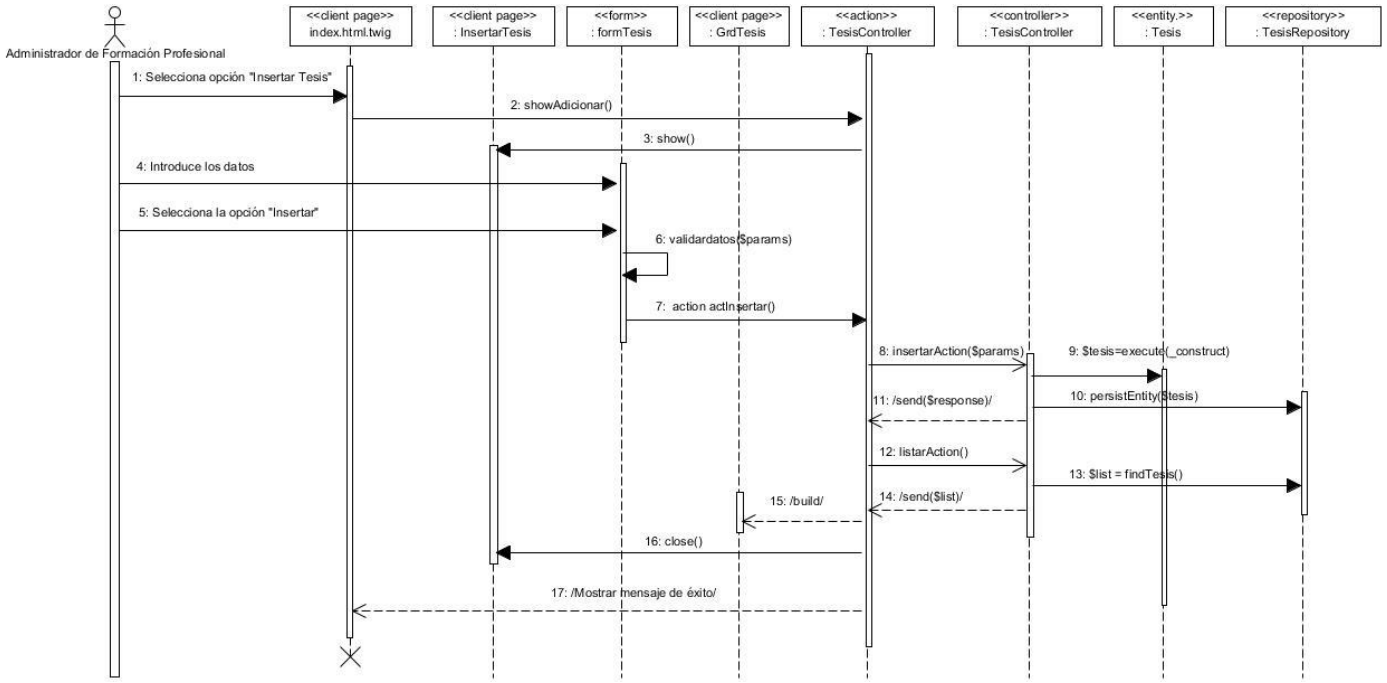


Fig. 8. Diagrama de Secuencia del CU Gestionar Tesis sección Insertar Tesis

En el Diagrama de Secuencia (Fig. 8) el actor Administrador de Formación Profesional selecciona la opción “Insertar Tesis”, luego la página cliente (CP_index.html.twig) se encarga de mandar a ejecutar de la acción TesisController el método showAdicionar(), esta llamada muestra la vista InsertarTesis que contiene un formulario creado en la clase formTesis, luego el actor introduce los datos en dicho formulario y da clic en el botón “Aceptar”. La clase formTesis valida el formulario y ejecuta la acción actInsertar(), esta a su vez ejecuta el método insertarAction() de la controladora TesisController pasándole como parámetros los valores del formulario. La controladora TesisController ejecuta el método _construct() de la entidad Tesis creando un nuevo objeto de esta clase. Luego la propia clase TesisController ejecuta el método persistEntity() del repositorio TesisRepository insertando el objeto creado a la base de datos y envía a la acción TesisController la información referente a la inserción. La acción TesisController captura esta respuesta y ejecuta el método listarAction() de la controladora TesisController que a su vez obtiene mediante el método findTesis() del repositorio TesisRepository las tesis existentes en la base de datos. La

clase controladora TesisController envía el resultado de la consulta a la acción TesisController para esta construir la vista GrdTesis y mostrar las existencias de la base de datos. Finalmente la acción TesisController ejecuta el método close() de la vista InsertarTesis y muestra un mensaje de éxito en la página cliente (CP_index.html.twig) terminando así el escenario “Insertar Tesis”.

2.6. Modelo de Datos

El modelo de datos aporta la base conceptual para el diseño de aplicaciones que hacen un uso intensivo de datos, siendo este la estructura o representación física de las tablas de la base de datos. De igual forma, permiten describir las restricciones de integridad y las operaciones de manipulación de los datos. (10). A continuación se muestra el modelo de datos del Sistema de Información para la gestión de la formación y calidad de software en DATEC:

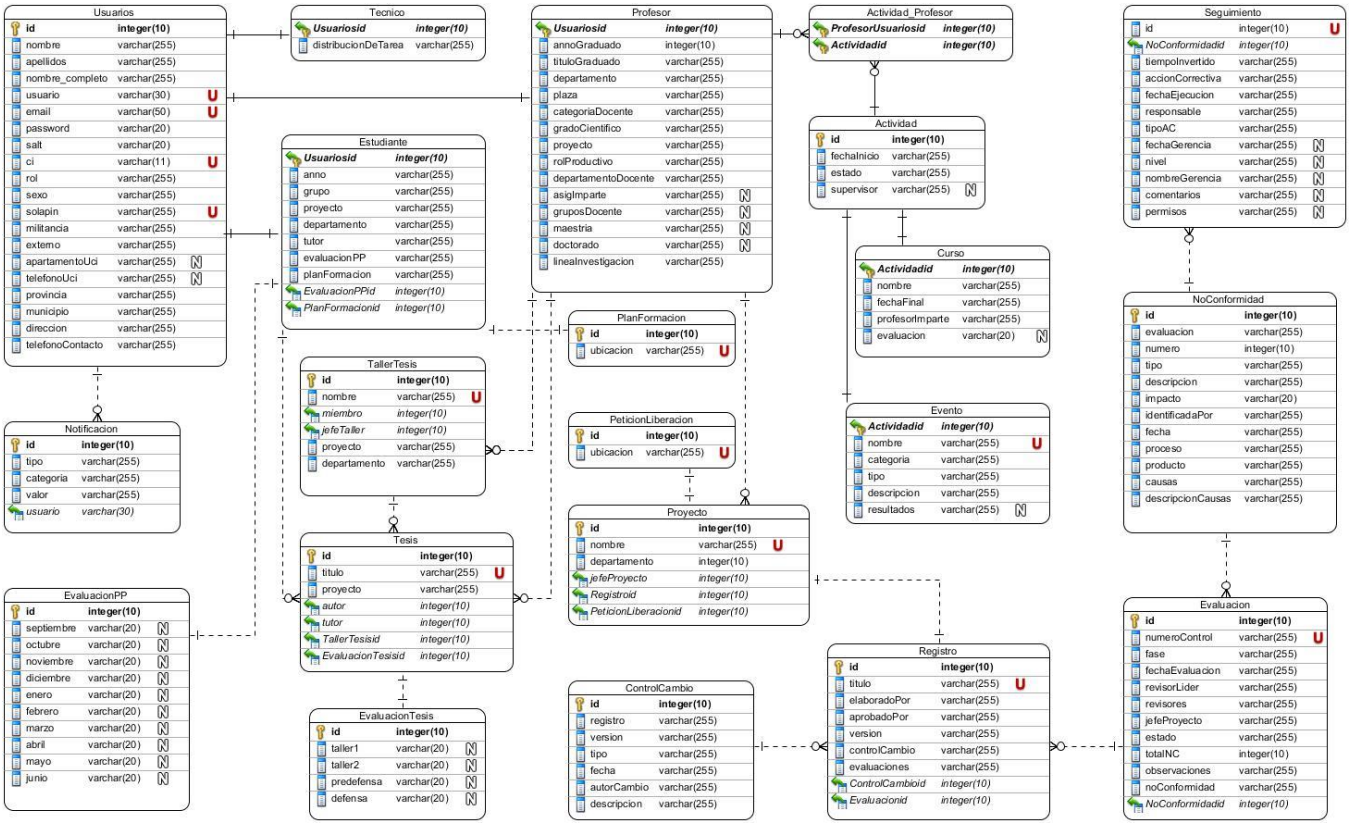


Fig. 9. Modelo de Datos del Sistema de Información para la gestión de la formación y calidad de software en DATEC

Descripción de las principales tablas del Modelo de Datos

Tabla 3: Descripción de la tabla Tesis del modelo de datos

Nombre: Tesis		
Descripción: Almacena las tesis existentes en el sistema.		
Atributo	**Tipo**	**Descripción**
id	integer(10)	Identificador auto-incremental.
titulo	varchar(255)	Título de la Tesis.
proyecto	varchar(255)	Proyecto al que pertenece la Tesis.
autor	varchar(255)	Autor(es) de la Tesis.
tutor	varchar(255)	Tutor(es) de la Tesis.

Tabla 4: Descripción de tabla Usuario del modelo de datos

Nombre: Usuario		
Descripción: Almacena los usuarios existentes en el sistema.		
Atributo	**Tipo**	**Descripción**
id	integer(10)	Identificador auto-incremental.
nombre	varchar(255)	Nombre del usuario.
apellidos	varchar(255)	Apellidos del usuario.
nombre_completo	varchar(255)	Nombre completo del reporte.
usuario	varchar(30)	Nombre de usuario para acceder al sistema.
email	varchar(50)	Email del usuario.

password	varchar(50)	Password del usuario.
salt	varchar(20)	Valor aleatorio utilizado para codificar el password.
ci	varchar(11)	Carnet de Identidad del usuario.
rol	varchar(255)	Rol del usuario en el sistema.
sexo	varchar(255)	Sexo del usuario.
solapin	varchar(255)	Solapín del usuario.
militancia	varchar(255)	Define si el estudiante es militante o no.
externo	varchar(255)	Define si el estudiante es externo o no.
apartamentoUci	varchar(255)	Apartamento del usuario en la UCI.
telefonoUci	varchar(255)	Teléfono del usuario en la UCI.
provincia	varchar(255)	Provincia de residencia del usuario.
militancia	varchar(255)	Municipio de residencia del usuario.
direccion	varchar(255)	Dirección de residencia del usuario.
telefonoContacto	varchar(255)	Teléfono para el contacto con el usuario.

2.7. Modelo de Despliegue

Los Diagramas de Despliegue muestran las relaciones físicas y reparto de componentes de los distintos nodos que componen un sistema. Además proporciona un modelo minucioso de la estructura en la que los componentes se desplegarán a lo largo de la infraestructura del sistema. Permite detallar las capacidades

de red, las especificaciones del servidor, los requisitos de hardware y otra información relacionada al despliegue del sistema propuesto (10).

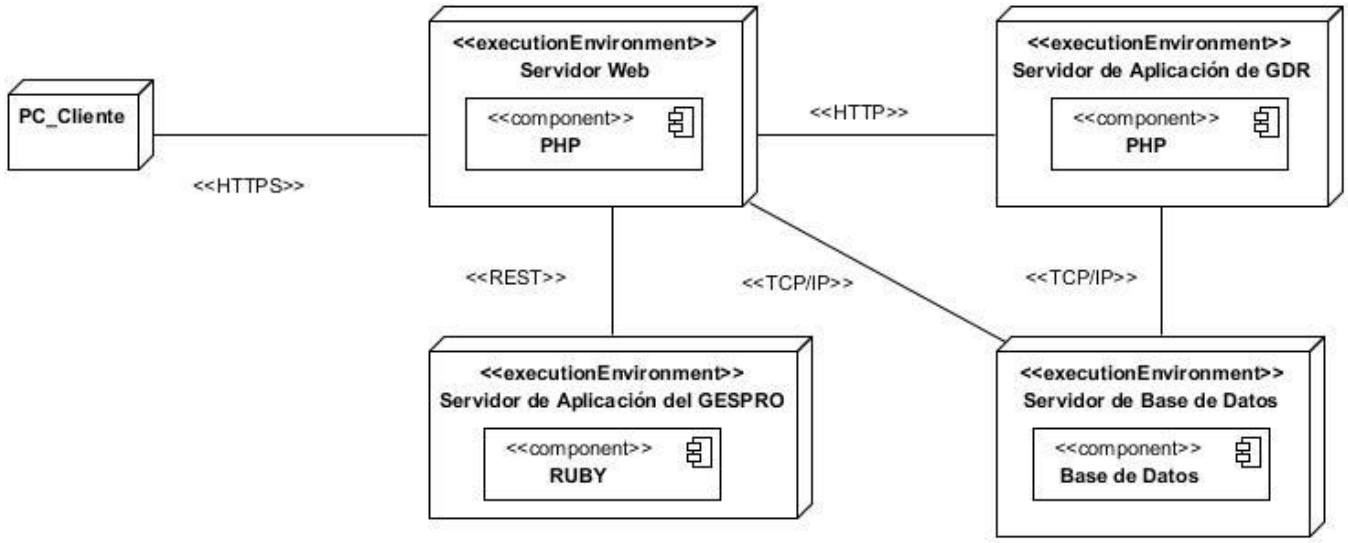


Fig. 10. Diagrama de Despliegue

(PC Cliente) Estaciones de trabajo con la aplicación cliente: Hace referencia a la estación de trabajo que el usuario utilizará para acceder a la aplicación Web.

Servidor de Aplicación: Servidor de aplicación donde se ha publicado la aplicación. Es la herramienta principal para ejecutar la lógica de negocio en el lado del servidor. Tiene la responsabilidad de ejecutar el código de las páginas servidor. Utiliza el protocolo seguro de transferencia de hipertexto (HTTPS, por sus siglas en inglés, *Hypertext Transfer Protocol Secure*), como protocolo de comunicación para establecer la conexión segura con la PC Cliente. Para la conexión con el servidor de aplicación de GDR utiliza el protocolo de transferencia de hipertexto (HTTP, por sus siglas en inglés, *Hypertext Transfer Protocol*) evitando que la información sensible pueda ser usada por un atacante que haya conseguido interceptar la transferencia de datos. La conexión con el servidor de aplicación del GESPRO se realiza mediante el protocolo de transferencia de estado representacional (REST, por sus siglas en inglés, *Representational State Transfer*). La conexión entre el Servidor Web y el Servidores de Base de Datos será mediante el Protocolo de Control de Transmisión/Protocolo de Internet (TCP/IP, por sus siglas en inglés, *Transmission Control Protocol/Internet Protocol*) garantizando el orden y la entrega de datos.

Servidor de base de datos: Es donde la aplicación se va a conectar para extraer la información de la base de datos. Debe tener instalado el Sistema Gestos de Base de Datos (SGBD) PostgreSQL.

2.7.1. Modelo de Paquetes

El diagrama muestra cómo está estructurado el sistema utilizando los paquetes para agrupar elementos. Cada paquete puede contener o no, otros paquetes o clases, que tienen interfaces y realizan cierta funcionalidad. Permiten dividir un modelo para agrupar y encapsular sus elementos en unidades lógicas individuales. Los paquetes pueden ser simples estructuras conceptuales o pueden estar reflejados en la implementación (33).

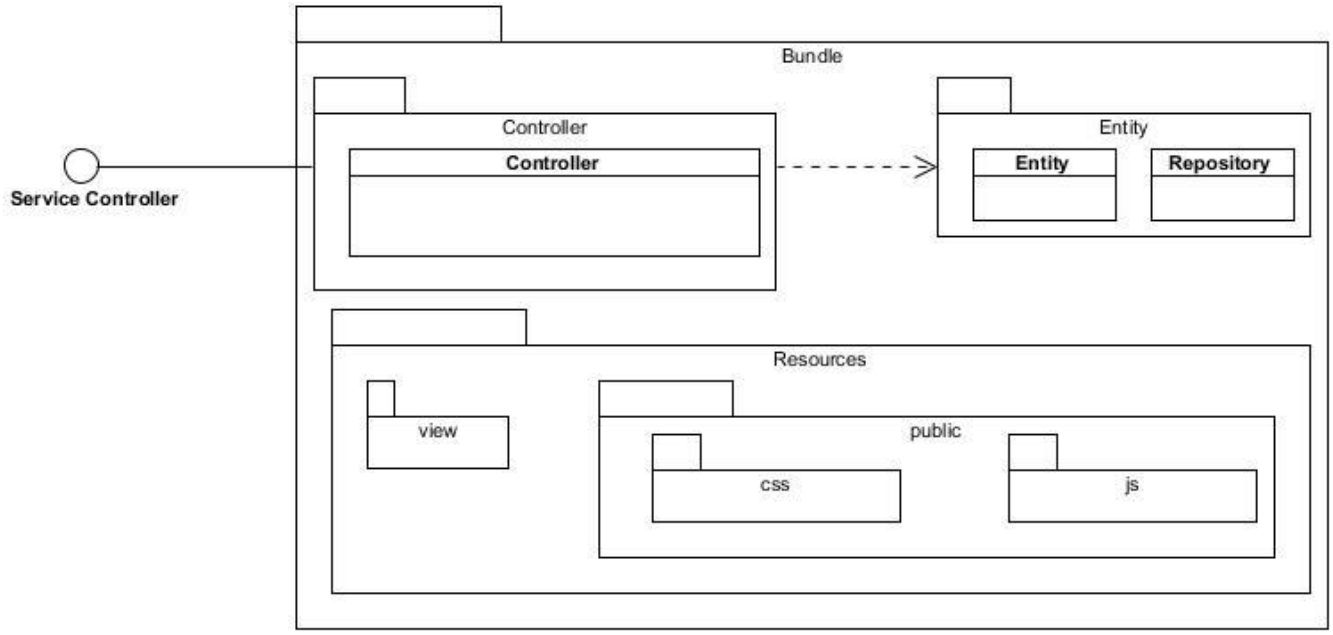


Fig. 11. Diagrama de Paquete

El *Bundle* de la aplicación estará compuesto por una carpeta denominada *Controller* donde se almacenarán todos los controladores del sistema, los cuales en dependencia de las acciones que realicen usarán los repositorios y las entidades correspondientes.

En la carpeta *Entity* se almacenarán las entidades, estas se encargarán de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes del tipo de gestor de

bases de datos. Junto a estas también se almacenarán los repositorios, los cuales organizarán las sentencias del lenguaje de consulta *Doctrine* de una entidad en cuestión.

La carpeta *Resources* contendrá las carpetas *view* y *public*. Dentro de la carpeta *view* se encontrarán las plantillas *Twig*, estas mostrarán los diferentes componentes del sistema. En la carpeta *public* se ubicarán diferentes recursos como hojas de estilos, imágenes y archivos *JavaScript*, cada uno en su carpeta correspondiente.

Se debe destacar que no se utilizará a *Symfony* en la elaboración de la presentación dado que esta se implementará mediante *ExtJS* 4.1, por lo que se define una organización (Fig. 12) para los componentes de presentación.

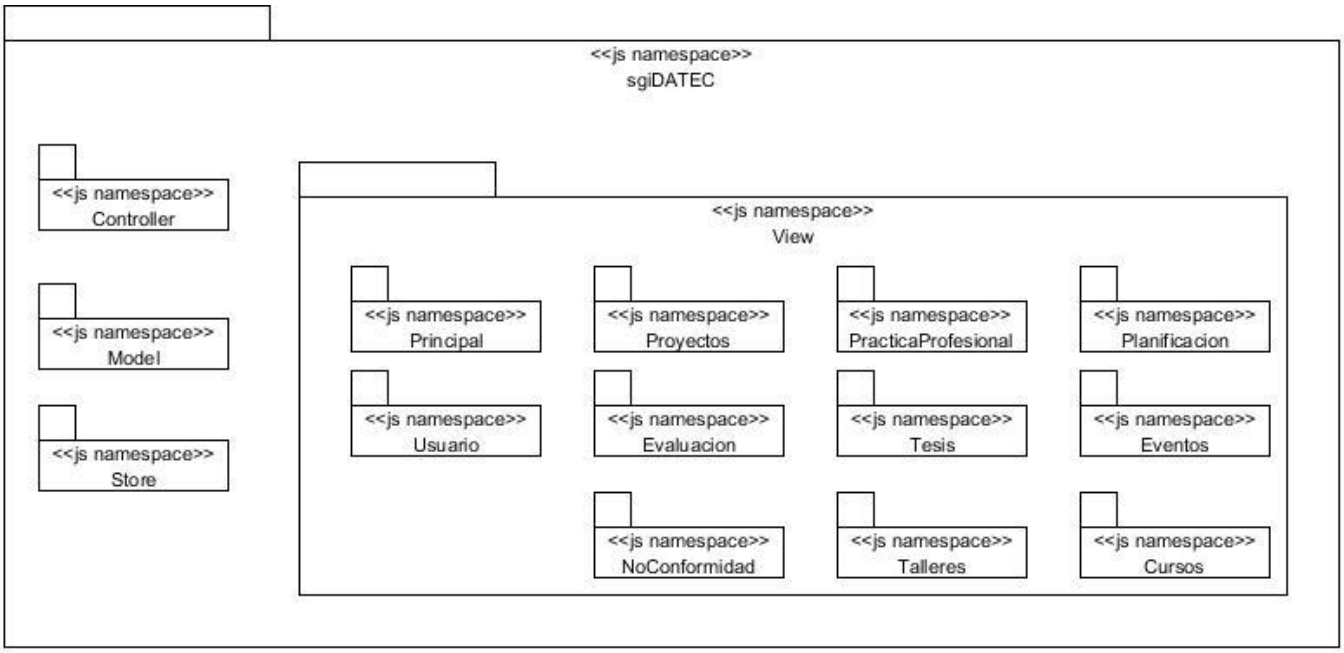


Fig. 12. Diagrama de Paquete del lado del cliente

En la carpeta *Controller* se almacenarán las clases controladoras que contienen las acciones que se realizan en el sistema. Las carpetas *Store* y *Model* contienen respectivamente los *stores* y *record* que realizan las diferentes peticiones a los controladores en la parte del servidor, en la carpeta *View* se almacenarán las vistas del sistema, cada una en su carpeta correspondiente de acuerdo a la función que cumpla dentro del sistema.

2.8. Conclusiones parciales del capítulo

En el presente capitulo se elaboró el Modelo de Dominio como apoyo para la comprensión de los principales conceptos del negocio. Fueron identificados 67 requisitos funcionales y 9 no funcionales que detallan las características de la aplicación, la función requerida, su comportamiento y rendimiento. Se agruparon los requisitos funcionales en casos de uso; estos fueron relacionados mediante un diagrama de casos de uso del sistema y se realizó una descripción detallada logrando así un mayor acercamiento a lo que el sistema deberá cumplir. Se determinó el patrón arquitectónico a seguir, así como los patrones de diseños utilizados en la realización de los diagramas de clases del diseño. Se generaron los diagramas de secuencias para cada uno de los escenarios, brindando una visión de cómo el usuario interactúa con la aplicación. A partir de la confección del diagrama de entidad-relación se obtuvo la estructura de la base de datos. Se confeccionó el diagrama de despliegue proporcionando una descripción de cómo se realizará el despliegue de los componentes.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN

El presente capítulo es el correspondiente a las disciplinas de implementación y pruebas. Se analizará el Modelo de Implementación así como una descripción de como los elementos del modelo de diseño se implementan en términos de componentes. De igual modo se realizarán las pruebas al sistema arrojando los principales resultados obtenidos a través de los casos de pruebas realizados a cada caso de usos.

3.1. Modelo de Implementación

El Modelo de Implementación describe cómo los elementos del modelo de diseño, se implementan en términos de componentes, ficheros de código fuente o ejecutables. Describe también cómo se organizan los componentes de acuerdo a los mecanismos de estructuración disponibles en el entorno de implementación, lenguajes de programación empleados, y la dependencia entre los componentes. Facilita la organización del producto y lo hace más entendible a los desarrolladores (34).

3.1.1. Diagrama de Componentes

Los diagramas de componentes se encuentran dentro del Modelo de Implementación. Un diagrama de componentes es la representación de las divisiones por componentes de un sistema de software y las dependencias entre ellos. Los componentes físicos incluyen: archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables y paquetes. Un componente es una parte de un sistema modular, desplegable y reemplazable que se encuentra en la computadora. Su potencial radica en la reusabilidad que ofrece su utilización (34).

A continuación se expone el diagrama de componentes correspondiente al CU Gestionar Tesis, el cual está compuesto por 3 paquetes de componentes (*View, Controller, Entity*). El paquete *View* agrupa los componentes que son utilizados en las interfaces, con los cuales interactúa directamente el usuario. El paquete *Controller* agrupa los componentes que se relacionan con la lógica del negocio, los cuales dan respuesta a las peticiones del usuario mediante la interacción con el paquete *Entity*. Por su parte este, cuenta con un conjunto de componentes relacionados con las clases de acceso a datos, las cuales son utilizadas por el paquete *Controller* para dar respuestas a la vista.

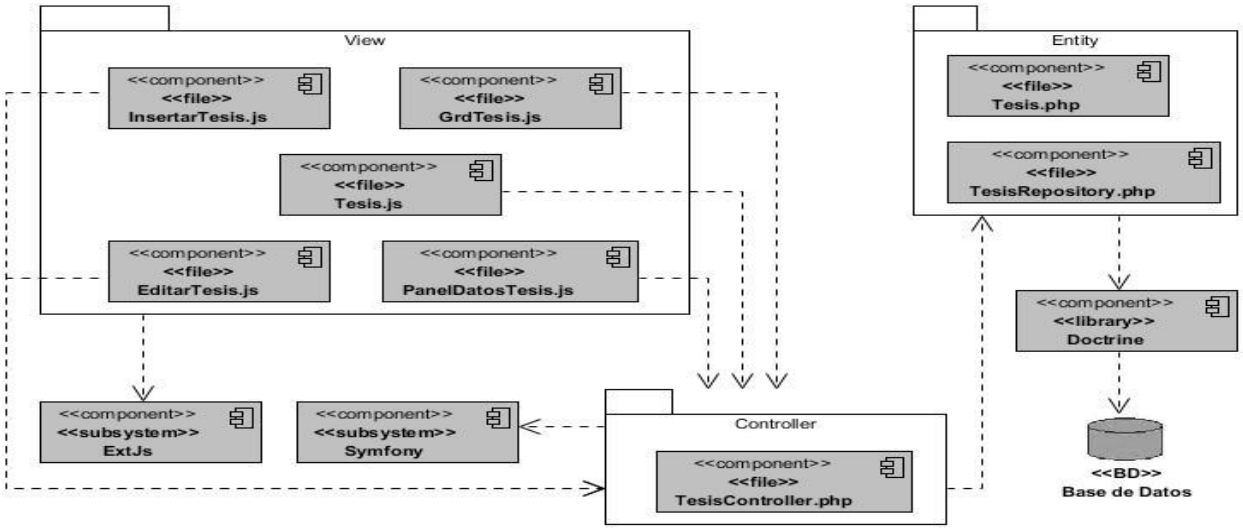


Fig. 13. Diagrama de Componentes del CU Gestionar Tesis

3.2. Código Fuente

El código fuente de un software es un conjunto de instrucciones que conforman un bloque de texto, escrito según las reglas sintácticas de algún lenguaje de programación. En el código fuente está escrito por completo el funcionamiento del programa. El código fuente no es ejecutable directamente por el computador, debe convertirse a lenguaje de máquina mediante compiladores, ensambladores o intérpretes (35). La escritura y organización del código fuente de un programa se define mediante los estándares de codificación. A continuación se detallan los estándares de codificación utilizados en la solución.

3.2.1 Estándares de codificación

Un estándar de codificación comprende todos los aspectos de la generación de código. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador lo hubiera escrito, de manera que facilite la comprensión del sistema para otros programadores y aumente la mantenibilidad del código. Al comenzar el desarrollo del software, deben establecerse estos estándares de codificación para asegurar la coordinación entre los programadores del proyecto. La adopción de un estándar de codificación sólo es viable si se sigue desde el principio hasta el final del proyecto de software (36).

Estándares de codificación utilizados para PHP (37)**:**

- La etiqueta de cierre ?> debe ser omitida en los ficheros que sólo contengan código PHP.
- El código debe usar 4 espacios como indentación, no tabuladores.
- Los nombres de clases pueden contener sólo caracteres alfanuméricos. Los números están permitidos en los nombres de clase, pero desaconsejados en la mayoría de casos.
- El nombre de una clase siempre empieza con mayúscula, en caso de estar compuesta por varias palabras se escribe una seguida de la otra y poniendo cada letra inicial en mayúscula.
- Para el nombre de clases, funciones y variables no está permitido el uso de barras bajas (_) ni de los espacios.
- Los nombres de funciones y variables deben empezar siempre con una letra minúscula utilizando la forma "camelCase".
- Se recomienda en sentido general la elocuencia, los nombres de las funciones deben ser elocuentes como para describir su propósito y comportamiento.

Estándares de codificación utilizados para JavaScript (38)**:**

- Los ficheros se nombrarán siempre en minúsculas para evitar errores con aquellas plataformas ‘sensitive-case’. Los ficheros tienen que concluir con la extensión .js, y no deben incluir signos de puntuación excepto – (guión medio) o _ (guión bajo).
- Las variables y las propiedades deben declararse utilizando la forma ‘camelCase’. No deben ir precedidas de un guion bajo para indicar que se tratan de variables privadas.
- Las clases deben nombrarse siguiendo la forma ‘*StudlyCaps*’ para diferenciarlos a simple vista del resto de estructuras.
- Las constantes deben nombrarse en mayúsculas y separando palabras con un guión bajo.
- Los métodos de los objetos deben ser nombrados utilizando la forma ‘*camelCase*‘.

Ejemplo de código fuente:

A continuación se muestra un ejemplo del código fuente de la funcionalidad “Visualizar Tesis”, el cual le permite al usuario poder visualizar los datos de una tesis seleccionada.

```
78        mostrarDatosTesis: function(grid, record) {
79            var me = this;
80            var panel = me.getPanelDatosTesis();
81            var myMask = new Ext.LoadMask(panel, {
82                msg: "Cargando..."
83            });
84            myMask.show();
85            Actions.Formacion_Tesis.datosTesis(record.data, {
86                success: function(options, asd) {
87                    if (options.success) {
88                        panel.updateDetail(options.datos);
89                        panel.expand();
90                        myMask.hide();
91                    } else {
92                        myMask.hide();
93                    }
94                },
95                failure: function(options, asd) {
96                    Ext.MessageBox.show({
97                        title: 'ERROR',
98                        msg: 'No se pudo establecer la conexión con el Servidor.',
99                        icon: Ext.MessageBox.ERROR,
100                       buttons: Ext.Msg.OK
101                   });
102                   myMask.hide();
103               }
104           });
105       }
```

Fig. 14. Fragmento del código " Visualizar Tesis " del CU Gestionar Tesis

3.3. Pruebas del software

El proceso de pruebas del software es el instrumento adecuado para evaluar la calidad del producto. Durante este proceso se realizan pruebas al sistema en su totalidad o a componentes del software, con el objetivo de identificar posibles fallos durante el proceso de desarrollo. De forma estructurada mediante Técnicas de prueba, se especifican los casos de prueba que se utilizarán en las pruebas. Los objetivos, métodos y técnicas del proceso de pruebas se describen en el plan de pruebas (39).

Durante todo el ciclo de desarrollo del software se aplican las pruebas con diferentes objetivos y distintos niveles de trabajo. Los niveles de pruebas verifican y validan un producto de software en diferentes ángulos con la capacidad de determinar no conformidades en relación con el uso que se le vaya a dar al sistema, así como del cumplimiento de todas las descripciones y respuestas del sistema en diferentes entornos (40). En la evaluación del sistema se ejecutan las pruebas a nivel de Pruebas de Desarrollador, el cual está diseñado e implementado por el equipo de desarrollo.

La tipo de prueba seleccionado para evaluar el correcto desempeño del sistema es la de Pruebas de

Funcionalidad. Estas pruebas permiten asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados (23).

Mediante la técnica de Caja Negra, también conocido como Pruebas de Comportamiento, se probará usando datos válidos e inválidos por cada requisito funcional, para confirmar que los resultados esperados ocurran cuando se usen datos válidos y se desplieguen los mensajes apropiados de error. La técnica de Caja Negra se lleva a cabo sobre la interfaz del software. Ejercitan todos los requisitos funcionales mediante un conjunto de condiciones de entrada, con el objetivo de detectar funciones incorrectas o ausentes, errores de estructura de datos o base de datos externas, errores de rendimiento y de inicialización (30).

3.3.1. Diseño de Caso de Prueba

Un caso de prueba se diseña según las funcionalidades descritas en los casos de uso, con el propósito de lograr una comprensión común de las condiciones específicas que la solución debe cumplir. Cada especificación de caso de uso es recogida en una planilla de caso de prueba, dividido en secciones y escenarios, describiendo sus funcionalidades y cada variable. También se registran las revisiones realizadas al caso de prueba, junto con todo aquello que no corresponde a la calidad del software.

En la confección de los casos de prueba de Caja Negra se utilizó el criterio de la técnica de Partición de Equivalencia. Esta técnica intenta dividir el dominio de entrada de un programa en un número finito de variables de equivalencia, de modo que se pueda inferir razonablemente que una prueba realizada con un valor representativo de cada variable es equivalente a una prueba realizada con cualquier otro valor de dicha variable. Las variables de equivalencia representan un conjunto de estados válidos y no válidos para las condiciones de entrada de un programa (10).

A continuación se presentan las tablas de la sección probada para el caso de uso Gestionar Tesis.

Tabla 5: Secciones de prueba para el caso de uso Gestionar Tesis

Nombre de la sección	Descripción de la funcionalidad
SC1 Insertar Tesis	El usuario decide insertar una nueva tesis, el sistema envía un mensaje indicando que la petición se realizó exitosamente, terminando así el CU.

A continuación se detallan las variables que se encuentran asociadas al caso de uso Gestionar Tesis.

Tabla 6: Descripción de las variables

No.	Nombre del campo	Clasificación	Valor Nulo	Descripción
1	Título de la tesis	Campo de Texto	No	Título con el cual se identificará la tesis. Alfanumérico. Admite 0-9 a-z A-Z
2	Nombre del taller	ComboBox	No	Será un taller que se encuentre almacenado en la base de datos.
3	Proyecto	ComboBox	No	Será un proyecto que se encuentre almacenado en la base de datos.
4	Autor(es)	ComboBox	No	Serán usuarios estudiantes que se encuentren almacenados en la base de datos.
5	Tutor(es)	ComboBox	No	Serán usuarios profesores que se encuentren almacenado en la base de datos.

Mediante las anteriores representaciones se realizó la matriz de datos que permitió evaluar y probar la información introducida por el usuario, enfrascándose únicamente en la sección que se toma como referencia. Con el empleo de la técnica de partición de equivalencia y utilizando un juego de datos correctos e incorrectos, se lograron registrar los resultados de las pruebas.

Tabla 7: Matriz de datos

Escenario	Descripción	1	2	3	4	5	Respuesta del sistema	Flujo Central	Resultado
EC 1.1 Insertar	Se adicionan correctamente	V	V	V	V	V	Se adiciona correctamente la	1- Se inicia la	Satisfactorio

Tesis	los datos de la tesis.						tesis en la BD. Muestra un mensaje de éxito en el almacenamiento.	Actividad. 2- Se llenan los datos correctamente. 3- Se selecciona aceptar. 4- Insertar los elementos.	
EC 1.2 Insertar Tesis con campos en blanco	Se adicionan los datos de la tesis dejando campos en blancos. Se muestra un mensaje indicando el error.	N/A	V	V	V	V	Se muestra un mensaje indicando que todos los campos son obligatorios.	1- Se inicia la Actividad. 2- Se escoge la opción aceptar. 3- Se muestra un mensaje de error.	Satisfactorio
		V	N/A	V	V	V			
		V	V	N/A	V	V			
		V	V	V	N/A	V			
		V	V	V	V	N/A			
		N/A	N/A	N/A	N/A	N/A			
EC 1.3 Insertar Tesis con datos no válidos	Se adicionan los datos de la tesis con valores inválidos. Se muestra un mensaje indicando el error.	I	N/A	N/A	N/A	N/A	Se muestra un mensaje indicando que debe insertar correctamente los datos.	1- Se inicia la Actividad. 2- Se llenan los datos. 3- Se escoge la opción aceptar. 4- Se muestra un mensaje de error.	Satisfactorio

EC 1.4 Insertar Tesis con cantidad de autores inválida	Se adicionan los datos de la tesis con número inválido de autores. Se muestra un mensaje indicando el error.	V	V	V	I	V	Se muestra un mensaje indicando el límite de autores en una tesis.	1- Se inicia la Actividad. 2- Se llenan los datos. 3- Se escoge la opción aceptar. 4- Se muestra un mensaje de error.	Satisfactorio
EC 1.5 Insertar Tesis con cantidad de tutores inválida	Se adicionan los datos de la tesis con número inválido de tutores. Se muestra un mensaje indicando el error.	V	V	V	V	I	Se muestra un mensaje indicando el límite de tutores en una tesis.	1- Se inicia la Actividad. 2- Se llenan los datos. 3- Se escoge la opción aceptar. 4- Se muestra un mensaje de error.	Satisfactorio

Tras aplicar las pruebas funcionales al sistema desarrollado se comprobó la correcta validación de los campos, verificando que cada cual aceptará exclusivamente los caracteres válidos. En el proceso de pruebas se detectaron 17 No Conformidades, las cuales fueron corregidas y gestionadas correctamente.

Además de validar la completitud de los requisitos con las pruebas funcionales al software, se realizaron pruebas para evaluar el rendimiento de estos requisitos definidos, bajo condiciones prefijadas en los requisitos no funcionales.

Para medir el rendimiento del sistema se utilizó el tipo de Pruebas de Carga y Estrés, que se realiza generalmente para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperadas. La carga puede ser el número esperado de usuarios concurrentes utilizando la aplicación y que realizan un número específico de transacciones durante el tiempo que dura la carga. Esta prueba

puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación. Las pruebas de rendimiento se realizaron utilizando la herramienta JMeter 2.9, para simular las peticiones a la aplicación de varios usuarios conectados concurrentemente.

Tabla 8: Tiempo medio de respuesta para distintas cantidades de usuarios conectados concurrentemente

Cantidad de usuarios concurrentes	Tiempo medio de respuesta (s)
100	7.3
150	10.1

Como resultado se obtuvo que la aplicación responde en 7.3 segundos a las peticiones de los 100 usuarios concurrentes definidos en los RNF; siendo este óptimo ya que los tiempos de respuesta inferiores a los 8 segundos son considerados permisibles. Además se midió el tiempo de respuesta de la aplicación para la concurrencia de 150 usuarios; obteniéndose un tiempo de respuesta de 10.1 segundos. Para obtener tiempos de respuesta adecuados para 150 o más usuarios es necesario mejorar las prestaciones de hardware de los servidores.

Luego de las verificaciones a nivel de desarrollador el sistema fue puesto en producción para ser validado por los clientes, expresando estos su conformidad con el sistema a través de la carta de aceptación disponible en el Anexo 1.

3.4 Conclusiones parciales del capítulo

El presente capítulo arrojó como resultado la obtención de los diagramas de componentes en concordancia con la estructura que presentan los diagramas de clases del diseño. Las relaciones de dependencia existentes entre los principales componentes del sistema guiaron la implementación de acuerdo a los patrones de diseño utilizados. Se brindó una especificación de los estándares de codificación utilizados y un fragmento del código como representación de su aplicación. Las pruebas de carga y estrés aplicadas mediante la herramienta JMeter permitieron medir el rendimiento del sistema ante las peticiones de una cantidad representativa de usuarios conectados concurrentemente. Para evaluar la calidad de la aplicación se validó la completitud de los requisitos con las pruebas funcionales al software. Se realizaron las pruebas de caja negra utilizando la técnica de particiones de equivalencia, obteniéndose 17 no conformidades significativas, las cuales fueron resueltas en su totalidad, obteniéndose un producto libre de errores y listo para su ejecución.

Conclusiones

Una vez culminada la investigación se puede afirmar que se alcanzaron los objetivos trazados arribando a las siguientes conclusiones:

- La realización del estudio de los principales conceptos relacionados a los sistemas de información y las nuevas tecnologías, posibilitó sentar las bases teóricas para el desarrollo del Sistema de Información para la gestión de la formación y calidad de software en DATEC.
- Las entrevistas realizadas permitieron la comprensión del negocio y la identificación de los requisitos; aspectos claves para las fases de análisis y diseño de la solución.
- Se realizó la implementación del Sistema de Información para la gestión de la formación y calidad de software en DATEC cumpliendo con los 67 requisitos funcionales identificados en las fases de análisis y diseño.
- La verificación de la calidad del sistema se realizó mediante las pruebas de caja negra y pruebas de carga y estrés, siendo solucionadas las no conformidades detectadas. Posteriormente la aceptación del sistema por parte del cliente complementó los resultados obtenidos en el proceso de pruebas.

Recomendaciones

- Diseñar reportes que contribuyan a la toma de decisiones utilizando el Generador Dinámico de Reportes integrado al sistema.
- Extender la solución al resto de los procesos gestionados en el centro para lograr una mayor centralización de la gestión de la información.

Referencias Bibliográficas

1. Curto, MSc. Joseph. Qué es la Gestión de la Información? *Information Management.* [En línea] 2006. www.informationmanagement.worpress.com.

2. *El desarrollo de los sistemas de información y documentación.* López Yepes, Dr José. 1991, Vol. Cuadernos EUBD.

3. Codina, Luis. *La investigación en sistemas de información.* Zaragoza : Universidad de Zaragoza, 1996.

4. *Information Systems and networks.* Dr. Kjell Samuelson Amsterdam: North Holland, 1977. Amsterdam: North Holland : s.n., 1977.

5. Emery, James C. *Sistemas de información para la dirección. El recurso estratégico crítico. .* Madrid : Ediciones Díaz de Santos, 1990.

6. J.Burch. *Diseño de Sistemas de Información: Teoría y Práctica.* 2014.

7. *Sistemas de información gerencial - Administración de la empresa digital.* Kenneth, Laudon, Jane. s.l. : Pearson Educación- Prentice Hall, 2006.

8. Bartle, Phil. Potención Conmunitaria. [Online] [Cited: 2 23, 2009.]. [En línea] 2009. [Citado el: 23 de febrero de 2014.] www.scn.org/mpfc/modules/mon-miss.html.

9. Campillo, D. Sistema de Gestión de Información para el control de Proyectos en Albet S.A. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2009.

10. García, Suárez del Villar Claudia. Herramienta para importar los reportes desde la versión 1.7 del Generador Dinámico de Reportes a su versión 2.0. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2012.

11. OpenUP. *openUP/Basic.* [En línea] 2007. http://epf.eclipse.org/wikis/openupsp/.

12. Pressman, R. *Software Engineering: a Practitioner's Approach. .* s.l. : The McGraw-Hill Companies, Inc., 2010.

13. Marqués, María Mercedes. Herramientas CASE. s.l. : http://www3.uji.es/~mmarques/f47/apun/node75.html.

14. Leyva Herbella Isnel, Soto Góngora Reynier. Módulo Farmacia del Sistema de Información Hospitalaria a las HIS. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas .* La Habana : s.n., 2009.

15. Sitio Oficial del Visual Paradigm. . [En línea] 2009. [Citado el: 20 de noviembre de 2013.] http://www.visual-paradigm.com/product/vpuml.

16. López, José Joaquín. *Informática y comunicaciones en la empresa.* Madrid : s.n., 2004.

17. Mayo Leyva, Yandier. Implementación de las validaciones en JavaScript para la AGR. Adaptación al framework ExtJS. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2008.

18. Fiallo Ana Yensi, Gonzalez de la Nuez J. Desarrollo del módulo Gestión de Tesis del Subsistema de Gestión Académica de Pregrado en la Universidad de las Ciencias Informáticas. *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2010.

19. PHP.net. [En línea] http://php.net/manual/es/intro-whatis.php.

20. Estrada Grillo, Victor Manuel. Módulo de Reportes Estándares para SIGE. *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2013.

21. Villa, Crysfel. Ext JS: La herramienta que revolucionó el front-end. . [En línea] 2009. www.sg.com.mx..

22. Pecos, Daniel. PostGreSQL vs. MySQL. . [En línea] http://www.netpecos.org/docs/mysql_postgres/index.html.

23. PgAdmin_III. [En línea] 2013. www.guia-ubuntu.org..

24. Tecnología y Synergix. [En línea] 2009. [Citado el: 20 de febrero de 2014.] http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/. .

25. García, Joaquín. Desarrollo de Software Orientado a Objetos. [En línea] 27 de septiembre de 2003. [Citado el: 03 de marzo de 2014.] http://www.ingenierosoftware.com/analisisydiseno/casosdeuso.php.

26. Pressman, Roger. *Ingeniería del software. Un enfoque práctico.* 2005.

27. Arizaca, Eliza. *Análisis y Diseño del sistema II.* 2013.

28. Lic Sergio Sanchez Rios. Metodologías de análisis y diseño VII. [En línea] http://www.slideshare.net/SergioRios/unidad-9-patrones-de-diseo.

29. Tello, Jesús Cáceres. *Diagrama de Secuencia. .* Alcalá : Dpto. Ciencias de la Computación, Universidad de Alcalá, 2012.

30. Santiago Salgado Rios. Repositorio Digital, ESPE. *Repositorio Digital, ESPE. .* [En línea] 2011. http://repositorio.espe.edu.ec/bitstream/21000/6316/1/AC-SISTEMAS-ESPE-047042.pdf..

31. Maza, Miguel A. *Internet, Javasript. : Innovación y Cualificación.* España : s.n., 2012.

32. *Marco de trabajo para aplicaciones web de código abierto en instituciones universitarias. .* Alarcón, Carlos Andrés Guerrero. s.l. : Revistas Memorias, 2012.

33. *UML Diagramas de Paquetes.* Gutierrez, Demián. s.l. : Universidad de los Andes, 2009.

34. Jacobson, Ivar, Grady, Booch y Rumbaugh, James. *El proceso unificado de desarrollo de software. .* 2009.

35. Diccionario de Informática. [En línea] http://www.alegsa.com.ar/Dic/codigo%20fuente.php.

36. Microsoft. 2013. [En línea] Microsoft Developer Network. , 2010.

37. Guía de Estilo de Codificación. . [En línea] http://www.php-fig.org/psr/psr-2/es/.

38. Benítez, Carlos. Guía de Estilo. Nomenclatura en archivos JavaScript. [En línea] 2012. http://www.etnassoft.com/2012/10/23/guia-de-estilo-nomenclatura-en-archivos-javascript/.

39. Gestión de Calidad y Pruebas de Software. [En línea] http://www.pruebasdesoftware.com/laspruebasdesoftware.htm .

40. Verdecia Fonseca, Miguel E. Visor de Reportes para Móviles con sistema operativo Android. Trabajo para optar por el título de Ingeniero en Ciencias Informáticas . La Habana : s.n., 2013.

41. Infante Frómeta, Jenny. Propuesta de Arquitectura de para Sistema Generador de Reportes. Universidad de las Ciencias Informáticas. La Habana : s.n., 2009.

Bibliografía

Curto, MSc. Joseph. Qué es la Gestión de la Información? *Information Management.* [En línea] 2006. www.informationmanagement.worpress.com.

El desarrollo de los sistemas de información y documentación. López Yepes, Dr José. 1991, Vol. Cuadernos EUBD.

Codina, Luis. *La investigación en sistemas de información.* Zaragoza : Universidad de Zaragoza, 1996.

Information Systems and networks. Dr. Kjell Samuelson Amsterdam: North Holland, 1977. Amsterdam: North Holland : s.n., 1977.

Emery, James C. *Sistemas de información para la dirección. El recurso estratégico crítico. .* Madrid : Ediciones Díaz de Santos, 1990.

J.Burch. *Diseño de Sistemas de Información: Teoría y Práctica.* 2014.

Sistemas de información gerencial - Administración de la empresa digital. Kenneth, Laudon, Jane. s.l. : Pearson Educación- Prentice Hall, 2006.

Administración de Sistemas de Información. Kenneth, Laudon, Jane. s.l. : Ed. Prentice Hall, 1998.

Los sistemas de información y la integración de sus sistemas de gestión normalizados. . Díaz, José Pino. s.l. : Universidad de Granada.

Bartle, Phil. Potención Conmunitaria. [Online] [Cited: 2 23, 2009.]. [En línea] 2009. [Citado el: 23 de febrero de 2014.] www.scn.org/mpfc/modules/mon-miss.html.

Barroso, Lic J. Propuesta de pautas para el diseño de un Sistema de Gestión de Información en la empresa ECIMETAL. Tesis de Licenciatura. Cuba, Departamento de Bibliotecología y Ciencia de la Información. La Habana : Universidad de La Habana, 2009.

Los Sistemas de Gestión de Información, piedra angular de la Estrategia integral de gerencia. Lara, MSc. Lourdes Portela. 2004.

Campillo, D. Sistema de Gestión de Información para el control de Proyectos en Albet S.A. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2009.

García, Suárez del Villar Claudia. Herramienta para importar los reportes desde la versión 1.7 del Generador Dinámico de Reportes a su versión 2.0. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2012.

OpenUP. *openUP/Basic.* [En línea] 2007. http://epf.eclipse.org/wikis/openupsp/.

Establecimiento de una Metodología de Desarrollo de Software para la Universidad de Navojoa Usando OpenUP. Carmina Lizeth Torres Flores, Germán Harvey Alférez Salinas. s.l. : Universidad de Navojoa, 2008.

Pressman, R. *Software Engineering: a Practitioner's Approach. .* s.l. : The McGraw-Hill Companies, Inc., 2010.

Marqués, María Mercedes. Herramientas CASE. s.l. : http://www3.uji.es/~mmarques/f47/apun/node75.html.

Leyva Herbella Isnel, Soto Góngora Reynier. Módulo Farmacia del Sistema de Información Hospitalaria a las HIS. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas .* La Habana : s.n., 2009.

Sitio Oficial del Visual Paradigm. . [En línea] 2009. [Citado el: 20 de noviembre de 2013.] http://www.visual-paradigm.com/product/vpuml.

López, José Joaquín. *Informática y comunicaciones en la empresa.* Madrid : s.n., 2004.

Mayo Leyva, Yandier. Implementación de las validaciones en JavaScript para la AGR. Adaptación al framework ExtJS. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2008.

Jorge Hera, Sergio. Interfaz de programación de aplicaciones para la transformación de esquemas de modelos de datos a Notación de Objetos JavaScript. *Trabajo para optar por el título de Ingeniero en Ciencias Informáticas .* La Habana : s.n., 2011.

Fiallo Ana Yensi, Gonzalez de la Nuez J. Desarrollo del módulo Gestión de Tesis del Subsistema de Gestión Académica de Pregrado en la Universidad de las Ciencias Informáticas. *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2010.

PHP.net. [En línea] http://php.net/manual/es/intro-whatis.php.

Estrada Grillo, Victor Manuel. Módulo de Reportes Estándares para SIGE. *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* La Habana : s.n., 2013.

Villa, Crysfel. Ext JS: La herramienta que revolucionó el front-end. . [En línea] 2009. www.sg.com.mx..

Pecos, Daniel. PostGreSQL vs. MySQL. . [En línea] http://www.netpecos.org/docs/mysql_postgres/index.html.

PgAdmin_III. [En línea] 2013. www.guia-ubuntu.org..

Tecnología y Synergix. [En línea] 2009. [Citado el: 20 de febrero de 2014.] http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/. .

García, Joaquín. Desarrollo de Software Orientado a Objetos. [En línea] 27 de septiembre de 2003. [Citado el: 03 de marzo de 2014.] http://www.ingenierosoftware.com/analisisydiseno/casosdeuso.php..

Pressman, Roger. *Ingeniería del software. Un enfoque práctico.* 2005.

Arizaca, Eliza. *Análisis y Diseño del sistema II.* 2013.

Lic Sergio Sanchez Rios. Metodologías de análisis y diseño VII. [En línea] http://www.slideshare.net/SergioRios/unidad-9-patrones-de-diseo.

Tello, Jesús Cáceres. *Diagrama de Secuencia. .* Alcalá : Dpto. Ciencias de la Computación, Universidad de Alcalá, 2012.

Santiago Salgado Rios. Repositorio Digital, ESPE. *Repositorio Digital, ESPE. .* [En línea] 2011. http://repositorio.espe.edu.ec/bitstream/21000/6316/1/AC-SISTEMAS-ESPE-047042.pdf..

Maza, Miguel A. *Internet, Javasript. : Innovación y Cualificación.* España : s.n., 2012.

38. *Marco de trabajo para aplicaciones web de código abierto en instituciones universitarias. .* Alarcón, Carlos Andrés Guerrero. s.l. : Revistas Memorias, 2012.

Symfony para todos. [En línea] http://symfony.cubava.cu/introduccion/symfony-2/.

UML Diagramas de Paquetes. Gutierrez, Demián. s.l. : Universidad de los Andes, 2009.

Marca Huallpara, Hugo M. *Análisis y Diseño de Sistemas II, Diagramas de Despliegues.* 2012.

Jacobson, Ivar, Grady, Booch y Rumbaugh, James. *El proceso unificado de desarrollo de software. .* 2009.

Diccionario de Informática. [En línea] http://www.alegsa.com.ar/Dic/codigo%20fuente.php.

Microsoft. 2013. [En línea] Microsoft Developer Network. , 2010.

Guía de Estilo de Codificación. . [En línea] http://www.php-fig.org/psr/psr-2/es/.

Benítez, Carlos. Guía de Estilo. Nomenclatura en archivos JavaScript. [En línea] 2012. http://www.etnassoft.com/2012/10/23/guia-de-estilo-nomenclatura-en-archivos-javascript/.

Gestión de Calidad y Pruebas de Software. [En línea] http://www.pruebasdesoftware.com/laspruebasdesoftware.htm .

Verdecia Fonseca, Miguel E. Visor de Reportes para Móviles con sistema operativo Android. Trabajo para optar por el título de Ingeniero en Ciencias Informáticas . La Habana : s.n., 2013.

Infante Frómeta, Jenny. Propuesta de Arquitectura de para Sistema Generador de Reportes. Universidad de las Ciencias Informáticas. La Habana : s.n., 2009.

Anexos

Anexo 1: Acta de aceptación del cliente

UCI
Universidad de las Ciencias Informáticas

Centro de Tecnologías de Gestión de Datos
DATEC

La Habana, ___ de _____ del 2014

"Año 56 de la Revolución".

ACTA DE ACEPTACIÓN

De una parte, el Centro de Tecnologías de Gestión de Datos, en lo sucesivo DATEC, de la Universidad de las Ciencias Informáticas (UCI), representado en este acto por: Ing. Alberto Mendoza Garnache, Ing. Clara Elena Brizuela, y de **otra parte** el estudiante: Alberto García Tosca.

Primero: Que en cumplimiento de los requisitos se ha efectuado el desarrollo del Sistema de información para la gestión de la formación y calidad de software en DATEC.
CONSIDERANDO: Que los hitos realizados cumplen con los requisitos establecidos, han sido desarrollados con la calidad requerida y bajo las condiciones pactadas y aprobadas por **Las Partes**.

CONSIDERANDO: Que los hitos que se han ejecutado cumplen con los requerimientos establecidos

POR TANTO: Las Partes acuerdan formalizar mediante la presente Acta, la aceptación del producto: **Sistema de información para la gestión de la formación y calidad de software en DATEC**

Y para que así conste, se extiende la presenta **Acta** en dos (2) ejemplares, rubricados por **Las Partes**.

Alberto García Tosca.
Entrega

Jefe Dpto. Integración de Soluciones
Ing. Alberto Mendoza Garnache

Asesora de Planificación y Control de DATEC
Ing. Clara Elena Brizuela Figueredo

Reciben