

UNIVERSIDAD MAYOR DE SAN ANDRÉS

FACULTAD DE CIENCIAS PURAS Y NATURALES

CARRERA DE INFORMÁTICA



PROYECTO DE GRADO

**“SISTEMA DE GESTIÓN Y SEGUIMIENTO ACADÉMICO
CASO: INSTITUTO BÍBLICO DE CAPACITACIÓN
INTERNACIONAL”**

**PARA OPTAR AL TÍTULO DE LICENCIATURA EN INFORMÁTICA
MENCIÓN: INGENIERÍA DE SISTEMAS INFORMÁTICOS**

POSTULANTE: IVAN JOSUE QUENTA VARGAS

TUTOR: Ph.D. MARISOL TELLEZ RAMIREZ

NUESTRA SEÑORA DE LA PAZ – BOLIVIA

2023

INDICE

Contenido

CAPÍTULO 1 MARCO INTRODUCTORIO.....	5
1.1 Introducción.....	5
1.2 Antecedentes	6
1.2.1 Antecedentes Institucionales	6
1.2.2 Antecedentes de Proyectos Similares.....	9
1.3 Definición del problema	11
1.4 Objetivos.....	13
1.4.1 Objetivo General.....	13
1.4.2 Objetivos Específicos.....	13
1.5 Justificación	13
1.6 Alcances y limites.....	14
1.6.1 Alcances	14
1.6.2 Límites	15
1.7 Metodología.....	15
1.8 Herramientas.....	18
CAPITULO 2 MARCO TEÓRICO	18
2.1 Introducción.....	18
2.2 Ingeniería de software.....	18
2.3 Metodologías de desarrollo ágiles	19
2.4 Scrum	21
2.5 Kanban	23
2.6 Scrumban.....	25
2.6.1 Flujo de trabajo.....	26
2.6.2 Características	26
2.6.3 Backlog en scrumboard.....	28

2.7 Beneficios y conclusiones de Scrumban	29
2.8 Metodología UWE	29
2.8.1 Fases de desarrollo	31
2.8.2 Modelo de requisitos	32
2.8.3 Modelo de contenido	33
2.8.4 Modelo de navegación	34
2.8.5 Modelo de presentación	35
2.8.6 Modelo de proceso	37
2.9 Herramientas de desarrollo	38
2.9.1 Herramientas de Frontend	38
2.9.1.1 Framework Angular	38
2.9.1.2 PrimeNG	39
2.9.2 Herramientas de backend	39
2.9.2.1 Framework Flask	39
2.9.2.2 JWT (Json-Web-Token)	40
2.9.2.3 Postman	41
2.9.2.4 PostgreSQL	42
2.10 Calidad de software	43
2.10.1 Métricas de software	44
2.10.2 Métricas de calidad	45
2.10.2.1 Norma ISO 9126	47
CAPITULO 3 MARCO APLICATIVO	48
3.1 Introducción	48
3.2 Especificaciones	49
3.2.1 Construcción de tablero	51
3.3 Fase Pregame	54
3.2.1 Conceptos y exploración	54
3.2.2 Roles Scrumban	55
3.4 Producción	55
3.4.1 Primera iteración	55
3.4.1.1 Modelado de datos	56

3.4.1.2 Modelado conceptual	58
3.4.2 Segunda iteración	59
3.4.2.1 Casos de uso – Gestión académica.....	60
3.4.2.3 Modelado de navegación – Gestión Académica	63
3.4.2.4 Modelado de presentación – Gestión Académica	64
3.4.3 Tercer iteración	65
3.4.3.1 Casos de uso – Gestión de pagos	66
3.4.3.2 Modelado de navegación – Gestión de pagos	68
3.4.4 Cuarta iteración	69
3.4.4.1 Casos de uso – Gestión de roles y permisos.....	70
3.4.4.2 Descripción de casos de uso – Gestión de roles y permisos.....	70
3.4.4.4 Modelo de navegación – Gestión de roles y permisos	71
3.5 Entrega del producto.....	73
3.5.1 Diseño de interfaces	73
3.6 Métricas.....	73
3.6.1 Funcionalidad.....	73
3.6.2 Usabilidad	74
CAPITULO 4 CONCLUSIONES Y RECOMENDACIONES	75
4.1 Conclusiones	75
4.2 Recomendaciones	76
1. CRONOGRAMA.....	77
2. BIBLIOGRAFÍA.....	79
ANEXOS	81

CAPÍTULO 1 MARCO INTRODUCTORIO

1.1 Introducción

En la actualidad, el uso de las Tecnologías de la Información y Comunicación en diversas organizaciones, ya sean públicas o privadas, adquiere una relevancia significativa.

Dentro del contexto de la educación, el término Tecnologías de la Información y Comunicación (TIC) se refiere a las diversas herramientas y recursos informáticos, digitales, audiovisuales y multimedia que utilizan las instituciones y su comunidad en general (docentes, estudiantes, directivos y personal administrativos) para llevar a cabo los procesos pedagógicos, académicos y organizacionales (Garcés, 2021).

Debido a los procesos manuales y tediosos que conlleva el Instituto Bíblico de Capacitación Internacional en el área administrativa y académica, el manejo de las TIC sería un apoyo considerable.

Las TIC pueden desempeñar un papel fundamental en la educación, la mejora de la enseñanza y el aprendizaje de alta calidad. Además, también pueden contribuir con la gestión, dirección y administración más eficientes de una institución educativa. (Unesco, 2016).

En consecuencia, debido al impacto de las TIC, el avance de la sociedad impulsa a que las instituciones y organizaciones promuevan la automatización en los procesos cotidianos. Estos recursos tecnológicos permiten mejorar las funciones y garantizar la confiabilidad en la gestión de la información.

Por estas razones, resulta fundamental la implementación de un sistema informático que se ajuste en función a las necesidades de la institución. Esto actúa como un facilitador en términos de información, tiempo y eficiencia.

En respuesta a estas necesidades, el “Instituto Bíblico de Capacitación Internacional”, cuya finalidad es la formación de pastores, obreros y líderes ministeriales, ha iniciado el proceso de desarrollo de un sistema de información, Este sistema tiene la finalidad resolver las dificultades que han surgido en áreas como la administración, seguimiento académico, registros, notas y cobro de materias, entre otras. Con esta iniciativa, la institución busca reducir tiempo y trabajo humano, de este modo permitiendo que el personal administrativo, docente y estudiantes dispongan de la información de manera eficiente y segura en el menor tiempo posible.

En este contexto, el presente proyecto abordará los estudios pertinentes para lograr la automatización de los procesos de gestión de la institución a través de una solución informática.

1.2 Antecedentes

Actualmente se ha presenciado un notable aumento en el registro y almacenamiento de la información en la institución. Conscientes de esta realidad, la institución ve la necesidad de mejorar la gestión y administración de dicha información. Por lo tanto, es imperativo implementar un sistema de gestión y seguimiento académico.

1.2.1 Antecedentes Institucionales

El Instituto Bíblico de Capacitación Internacional, tiene como objetivo brindar apoyo a líderes y no líderes, incluyendo a pastores, obreros y líderes de poca experiencia ministerial, que desean mejorar su conocimiento bíblico y capacitación ministerial. El enfoque de la institución está en el desarrollo efectivo de su ministerio y en satisfacer para todo aquello el profundizar su comprensión de la palabra de dios a través de los estudios teológicos. El instituto IBCI es

perteneciente a las iglesias evangélicas en América Latina, ha estado en funcionamiento durante 28 años bajo la dirección del director nacional Lic. W. Hugo Carrasco. (Sainz, 2023).

El Instituto Bíblico de Capacitación Internacional, está dirigido por director regional el Lic. Adolfo G. Flor Sainz y Pr. Alyne Monteiro Costa de Flor.

El instituto bíblico de capacitación internacional ofrece un programa por tres niveles, con un total de 10 materias agrupadas por nivel, hace un total de 30 materias que dan con la duración de 3 años de estudio, al completar cada nivel, los estudiantes reciben los siguientes certificados:

- Primer nivel, certificado básico en biblia y teología.
- Segundo nivel, diploma en biblia y teología.
- Tercer nivel, bachiller en biblia y teología.

El sistema de estudio utilizado es el modular, el cual se basa en el estudio de una materia por mes, con una duración de 4 semanas. Además, la institución implementa otros sistemas de estudio, tal como:

- Sistema presencial: Las clases se imparten de manera presencial en la zona Villa Adela, El Alto, en la iglesia ICAD El Paraíso.
- Sistema virtual: Las clases se llevan a cabo en formato virtual, a través de la plataforma de Zoom, brindando flexibilidad y acceso remoto a los estudiantes.
- Sistema de materias sueltas, este sistema permite tomar materias específicas a través de un convenio con las iglesias locales que solicitan dichas asignaturas, brindando opciones personalizadas de estudio.
- Sistema de curso ministerial: Este sistema está diseñado exclusivamente para pastores y/o encargados de ministerios que están en pleno ejercicio ministerial. A través de un convenio entre IBCI y las iglesias locales, se ofrece la oportunidad de estudiar materias

específicas, permitiendo a los participantes completar su experiencia ministerial con conocimientos teológicos (Sainz, 2023).

Misión: Formar y capacitar a obreros y pastores, con sana doctrina y ética cristiana, basada en principios y verdades de las sagradas escrituras para responder a las necesidades de la iglesia y de la sociedad.

Visión: Enviar a todos los obreros y pastores capacitados para un servicio de excelencia en la iglesia local, nacional e internacional, enteramente preparados para toda buena obra edificando y extendiendo el reino de Dios.

El siguiente organigrama del Instituto Bíblico de Capacitación Internacional tiene la siguiente organización, como se muestra en la figura 1.

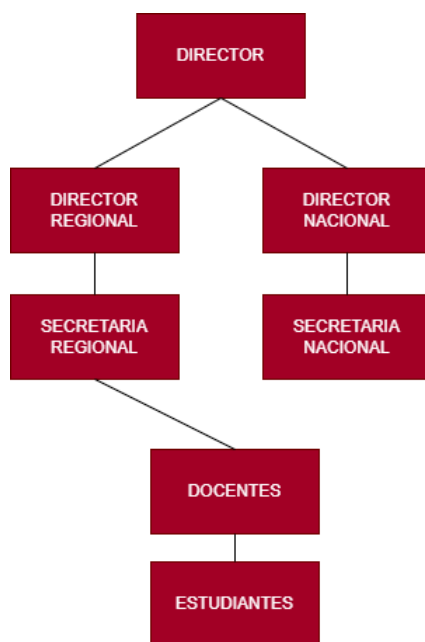


Figura 1 Estructura Orgánica Instituto Bíblico de Captación Internacional

Fuente (IBCI, 2023)

1.2.2 Antecedentes de Proyectos Similares

- Universidad Mayor de San Andrés

Título: “Sistema de Seguimiento Académico y Control Disciplinario”

Autor: Magaly Norah Salazar Martinez

Año: 2015

Este sistema permite un seguimiento académico y control disciplinario efectivos al contar con una organización adecuada, un control más eficiente y una optimización del tiempo en los procesos de administración de la institución. Incluye la implementación de módulos de administración, usuario, control disciplinario y seguimiento académico para los estudiantes de una Unidad Educativa (Salazar, 2015).

- Universidad Mayor de San Andrés

Título: “Sistema de Seguimiento Académico y gestión administrativa. Caso: Unidad de Postgrado en informática - UMSA”

Autor: Nestor Calixto Chura Aguilar

Año: 2014

Se desarrolló un sistema de seguimiento académico y gestión administrativa para mejorar los procesos de la Unidad de Postgrado en Informática de la UMSA. Se utilizó la metodología Extreme Programming (XP) y se combinaron las metodologías WebML y diagramas UML para la documentación. Se aplicaron métricas de calidad basadas en el estándar ISO 9126 y se realizó un análisis de seguridad según la ISO 17799. Además, se realizó un análisis de costo beneficio utilizando el modelo COCOMO II para determinar el costo total del software desarrollado (Chura, 2014).

- Universidad Católica Los Ángeles Chimbote

Título: Análisis de un Sistema Web de Seguimiento Académico para la I.E.P. “San Marcos”

Autor: Tanya Anai Carmen Herrera

Año: 2021

Este trabajo se enfocó en el análisis de un Sistema Web de Seguimiento Académico para la I.E.P. "San Marcos" en Tambogrande, Piura, con el objetivo de mejorar la gestión de la información académica y la calidad del servicio. Se utilizó una metodología cuantitativa, descriptiva y no experimental, con una muestra de 20 agentes educativos de una población de 393 personas. Se aplicaron encuestas y cuestionarios. Los resultados mostraron que los padres de familia y el personal administrativo están satisfechos con el sistema actual, aunque el conocimiento de las TIC varía entre ellos. Todos los encuestados expresaron la necesidad de un Sistema Web de Seguimiento Académico. Se concluye que la hipótesis propuesta es válida y el sistema automatizado beneficiará a los agentes educativos al tener una mejor organización de la información del estudiante y ofrecer un servicio eficiente y seguro. (Carmen, 2021).

- Universidad de San Martín de Porres

Título: “Diseño de un Sistema Web para el seguimiento y evaluación de los alumnos con carta de permanencia en la facultad de ciencias contables, económicas y financieras de la Universidad de San Martín de Porres”

Autores: Einstein Manuel Novoa Tafur, Julio César Rodríguez Postigo

Año: 2015

Este proyecto buscó diseñar un sistema web para el seguimiento y evaluación de alumnos con carta de permanencia en la Facultad de Ciencias Contables, Económicas y

Financieras de la Universidad de San Martín de Porres. Se utilizó la metodología ágil SCRUM y se logró reducir tiempos, disminuir la deserción estudiantil mediante alertas académicas y mejorar la calidad de atención académica. El sistema automatiza la gestión académica, permitiendo un seguimiento continuo del desempeño y generando informes para una mejor coordinación entre profesores y personal administrativo. (Novoa y Rodríguez, 2015).

1.3 Definición del problema

El Instituto Bíblico de Capacitación Internacional, ubicado en la ciudad de El Alto, actualmente lleva a cabo todos sus procesos de manera manual, utilizando informes físicos para almacenar información sobre inscripciones y registro de pagos.

Sin embargo, este enfoque manual conlleva varias dificultades. Por ejemplo, cuando se solicitan notas o pagos pendientes, el proceso de recopilación es extremadamente lento, llevando al menos 30 minutos. Además, otras solicitudes de los estudiantes también se ven afectadas por esta situación, lo que provoca frustración entre ellos. Además, el hecho de que toda la documentación se encuentre en formato físico aumenta el riesgo de pérdida de los documentos, lo cual agrava aún más la situación.

Por otra parte, los informes financieros también presentan problemas debido al desorden de los documentos, lo que retrasa el proceso de búsqueda y consulta de información.

En resumen, el Instituto Bíblico de Capacitación Internacional de El Alto enfrenta desafíos significativos debido a su enfoque manual en los procesos actuales. La lentitud en la recopilación de datos, la posible pérdida de documentación y el desorden en los informes

financieros son solo algunos ejemplos de los problemas que afectan tanto al personal como a los estudiantes.

El Instituto Bíblico de Capacitación Internacional presenta los siguientes problemas:

- El proceso de administración de datos de los estudiantes nuevos presenta demoras, ya que se realiza de forma manual, lo que aumenta el riesgo de cometer errores en la transcripción de la información.
- La información de los estudiantes se encuentra almacenada en diferentes carpetas, lo que dificulta su localización y puede provocar dificultades para encontrar la información necesaria, generando retrasos en la búsqueda de datos de los estudiantes.
- La falta de disponibilidad inmediata de la información. Debido a que los estudiantes retoman las materias después de periodos de 3, 6 o incluso 9 meses, su información queda almacenada y esto puede dificultar la localización de la documentación del estudiante cuando es necesaria.
- Se identifica una falta de monitoreo hacia los estudiantes, especialmente aquellos que abandonan o se retiran parcialmente de las materias. Existe la necesidad de realizar un seguimiento y notificar la apertura de nuevas materias para inscribirse. Esta falta de información sobre los estudiantes que no continúan con el curso dificulta el seguimiento adecuado y la comunicación oportuna, como el envío de notificaciones.

Por lo tanto, el problema principal planteado es: ¿Cómo mejorar el proceso de gestión y seguimiento en el Instituto Bíblico de Capacitación Internacional?

1.4 Objetivos

1.4.1 Objetivo General

Desarrollar un sistema de gestión y seguimiento académico para el instituto bíblico de capacitación internacional, para gestionar los procesos en organización, control y monitoreo.

1.4.2 Objetivos Específicos

- Establecer requerimientos y requisitos basados en las especificaciones del Instituto Bíblico de Capacitación Internacional (IBCI) para proponer mejoras en la administración.
- Diseñar una interfaz de usuario intuitiva, atractiva y funcional, que cumplan con las especificaciones y requisitos establecidos por el IBCI.
- Diseñar y desarrollar módulos para la gestión de la información del IBCI.
- Implementar medidas de seguridad, mediante niveles de acceso, para garantizar la protección de la información.
- Realizar pruebas de calidad, según parámetros del estándar ISO 9126.

1.5 Justificación

El Instituto Bíblico de Capacitación Internacional en la parte administrativa, tiene como objetivo brindar una mejor atención a los estudiantes, pero es necesario mejorar la forma en que se registran los estudiantes, las notas y los pagos.

Implementar un sistema de gestión y seguimiento académico permitirá al personal tener un mayor control sobre la información de los estudiantes y docentes, al mismo tiempo que reducirá el tiempo necesario para realizar registros, consultas y búsquedas de información de cada estudiante.

Este sistema automatizado ayudará a mejorar la eficiencia y permitirá brindar una atención más rápida y precisa a los estudiantes. Además, se evitará la carga de trabajo manual para el personal y los docentes en lo que respecta al seguimiento de los estudiantes.

1.6 Alcances y límites

1.6.1 Alcances

Los alcances se reflejarán en la especificación de los módulos, y la gestión de información se encargará de proporcionar la información necesaria a estudiantes, docentes y personal administrativo.

- El módulo de registro de estudiantes, se dedicará exclusivamente al registro de información de los estudiantes.
- El módulo de materias, permitirá el registro de nuevas materias o la asignación de materias a un nivel específico, así como la visualización de un informe que muestra todas las materias que se deben cursar.
- El módulo de administración de notas, permitirá el registro de notas por materia.
- El módulo de monitoreo de estudiantes, comprenderá de un seguimiento personalizado de los estudiantes, permitiendo el envío de mensajes de notificación sobre la apertura de nuevas materias.
- El módulo de control de pagos, permitirá visualizar los registros de pagos realizados, registrar nuevos pagos y mostrar una lista de pagos pendientes.
- El módulo de reportes, proporcionará una visualización de todos los informes generados por cada módulo en diferentes formatos, como PDF y/o Excel.

- El módulo de material de apoyo, proporcionará textos, libros, manuales, entre otros para que los estudiantes tengan el material, adicional y complementario en su avance.
- El módulo de certificados, se proporcionará exclusivamente los certificados obtenidos en los niveles aprobados o curso adicionales por los estudiantes.
- El módulo de tesorería, se encargará del control y verificación de los montos generados.

1.6.2 Límites

- El proyecto se enfocará en el análisis, diseño y desarrollo de los procesos necesarios, teniendo en cuenta los requerimientos y el estudio de la situación actual.
- Las interfaces del sistema estarán limitadas en función de los niveles de autorización de los usuarios, garantizando que solo los tipos de usuarios autorizados tengan acceso.
- El sistema estará disponible para el uso de docentes, estudiantes y personal administrativo.
- El desarrollo se realizará exclusivamente para el Instituto IBCI, ubicado en la ciudad de El Alto, sin contemplar su implementación en otras sucursales.

1.7 Metodología

El sistema de información de gestión y seguimiento académico se desarrolla utilizando una metodología ágil llamada Scrumban, que combina aspectos más destacados de los métodos de desarrollo Scrum y Kanban. Scrumban es una metodología que se origina al combinar la naturaleza prescriptiva de Scrum para agilizar el desarrollo y aprovechar las mejoras de procesos de Kanban (Gamboa, 2014). Esta metodología es especialmente adecuada para proyectos de mantenimiento o también los proyectos en los que los requerimientos del software cambian con frecuencia o pueden surgir errores de programación inesperados durante todo el ciclo de

desarrollo. En estos casos, los sprints de Scrum no son convenientes, ya que los errores y obstáculos que aparecen a lo largo del proyecto son difíciles de determinar, lo que dificulta estimar el tiempo necesario para cada tarea. Por lo tanto, es más conveniente adoptar el flujo de trabajo continuo del modelo Kanban. Sin embargo, se pueden mantener las reuniones diarias de Scrum para monitorear regularmente el progreso del producto.

Scrumban combina las ventajas de las dos metodologías ágiles Scrum y Kanban al aprovechar la visualización del proceso por etapas y la asignación del equipo en una pizarra blanca con tarjetas que representan las tareas pendientes. Estas tareas son seleccionadas por el personal adecuado hasta que se completan en la etapa final. Además, esta metodología establece límites tanto para las tareas en las que trabaja el equipo en conjunto como para las tareas individuales. (Taberner, 2015).

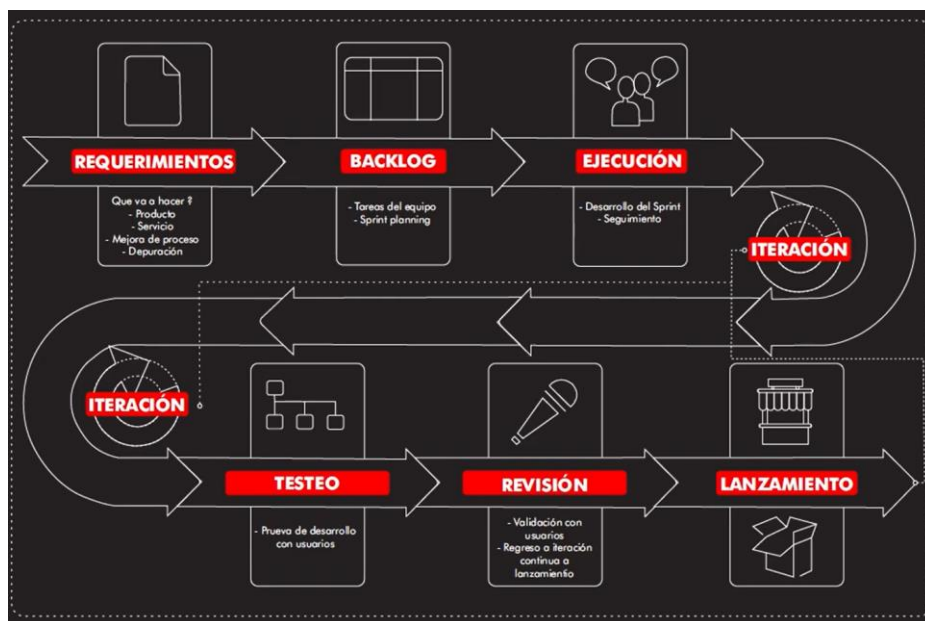


Figura 2 Ciclo de la metodología Scrumban.

Fuente: (Thinking, 2020)

Adicionalmente, Scrumban establece restricciones en la cantidad de tareas que se pueden trabajar tanto en equipo como individualmente. Por lo tanto, se establece una política que permite a un miembro del equipo manejar recursivamente hasta dos tareas simultáneamente. Esto se debe a la necesidad de flexibilizar la aceptación de nuevas tareas en caso de bloqueos o circunstancias particulares. También el número de tareas en proceso a un nivel global se asigna al equipo de desarrollo, para mantener el orden y constancia del desarrollo del trabajo. (Taberner, 2015).

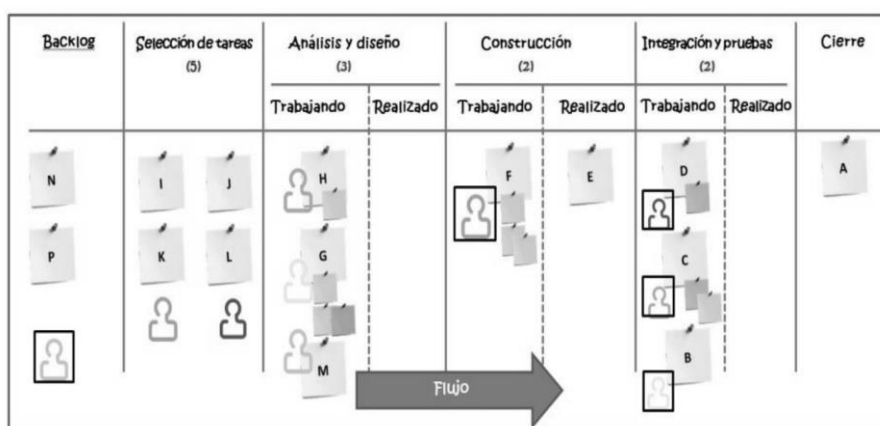


Figura 3 Flujo de trabajo de Scrumban.

Fuente: (Guzmán, Castañeda Islas, & Pedroza Méndez, 2014)

También se contemplará la metodología de modelado UWE (UML-based web engineering - UML basado en ingeniería web), que se enfoca en aplicaciones personalizadas o adaptativas. Esta metodología diferencia claramente entre las etapas de obtener, definir y validar los requisitos. El resultado final de la recopilación de requisitos utilizando UWE es la creación de un modelo de casos de uso junto con documentación adicional que describe a los usuarios del sistema, las reglas de adaptación, los casos de uso y la interfaz.

1.8 Herramientas

CAPITULO 2 MARCO TEÓRICO

2.1 Introducción

En este capítulo, se abordará los conceptos esenciales de las metodologías que se utilizarán para fundamentar el presente proyecto, en base al planteamiento de los problemas y objetivos, realizando la búsqueda, extracción y recopilación de información necesaria.

2.2 Ingeniería de software

La Ingeniería de Software es una disciplina tecnológica con múltiples facetas, que está compuesta por varias capas, como puede ser apreciada en la Figura 1, requiere un sólido compromiso organizacional con la calidad como base fundamental. En este contexto, filosofías como la Administración Total de la Calidad, Six Sigma y otras similares desempeñan un papel crucial al fomentar una cultura de mejora continua. Es esta cultura de mejora constante la que impulsa el desarrollo de enfoques cada vez más efectivos en la Ingeniería de Software.

Figura 2.1

Capas de la Ingeniería de Software



Nota. La Ingeniería de Software es una disciplina compuesta por capas. Tomado de (Pressman, 2010).

El pilar fundamental de la Ingeniería de Software se encuentra en su capa de proceso. La capa de proceso es el actuar de un elemento cohesionador que integra diferentes capas tecnológicas, que permite el desarrollo metódico y oportuno del software. Este proceso define una estructura establecida para obtener de manera eficaz la tecnología de Ingeniería de software. Además, el proceso de desarrollo de software sirve como base para la gestión de proyectos de software, estableciendo el marco en el cual se aplican enfoques técnicos, se generan productos de trabajo (como modelos, documentos, datos, informes, formatos, entre otros), se garantiza la calidad y se gestiona adecuadamente el cambio.

Los métodos en el campo de la Ingeniería de Software ofrecen la experiencia técnica necesaria para la creación de software. Estos métodos son conjunto de actividades, que abarcan desde la comunicación inicial hasta el análisis de requisitos, la creación de modelos de diseño, la programación, las pruebas y el soporte posterior.

Las herramientas de Ingeniería de Software proporcionan apoyo automatizado o semiautomatizado para respaldar tanto el proceso como los métodos utilizados en el desarrollo de software. Cuando se integran estas herramientas de manera que la información generada pueda ser aprovechada por otra, se crea un sistema conocido como “Ingeniería de Software Asistida por Computadora”, que facilita y mejora el desarrollo de software. (Pressman, 2010, p. 44).

2.3 Metodologías de desarrollo ágiles

Para comenzar, es esencial comprender las definiciones de 'metodología' y 'desarrollo'. La palabra 'metodología' se compone de tres términos de origen griego: 'metá' (que significa 'más

allá'), 'odós' ('camino'), y 'logos' ('estudio'). A la luz de esto, podemos definir la 'metodología' como los enfoques y métodos utilizados para identificar el camino más apropiado a seguir (Rivas et al., 2015, p. 982).

Hacer mención de las metodologías ágiles conlleva la necesidad de aludir a las prácticas convencionales de desarrollo de software, dado que las metodologías ágiles surgieron como una respuesta a las metodologías tradicionales. Estos dos enfoques poseen rasgos fundamentales que son opuestos entre sí, y su aplicación óptima encuentra su lugar en contextos dispares.

Las metodologías tradicionales en el ámbito del desarrollo de software se rigen por un enfoque centrado en la planificación. Inician un proyecto de desarrollo con un proceso exhaustivo de obtención de requisitos antes de proceder a las etapas de análisis y diseño. Esta estrategia busca asegurar la entrega de resultados de alta calidad dentro de un marco de tiempo definido, mientras las metodologías ágiles se destacan por su flexibilidad, permitiendo su adaptación a las necesidades específicas de cada equipo y proyecto. Estos proyectos ágiles se dividen en unidades más pequeñas a través de una lista organizada de características. Cada proyecto se aborda de forma independiente, desarrollando un conjunto de características en un breve período, generalmente de dos a seis semanas. La comunicación con el cliente es constante, llegando al punto de contar con un representante del cliente durante todo el proceso de desarrollo. Estos proyectos se caracterizan por su alta colaboración y su capacidad para adaptarse a los cambios; de hecho, la adaptación a cambios en los requisitos se considera una característica esperada y deseada, al igual que la entrega continua de resultados al cliente y la retroalimentación constante por parte de este. Tanto el producto como el proceso se someten a mejoras de manera regular. (Navarro Cadavid et al., 2013, p. 31).

Tabla 2.1

Metodologías tradicionales vs metodologías Ágiles

Metodologías tradicionales	Metodologías ágiles
Predictivos	Adaptativos
Orientados a procesos	Orientados a personas
Proceso rígido	Proceso flexible
Se concibe como un proyecto	Un proyecto es subdividido en varios
Poca comunicación con el cliente	proyectos más pequeños
Entrega de software al finalizar el desarrollo	Comunicación constante con el cliente
Documentación extensa	Entregas constantes de software
	Poca documentación

Nota. Características de metodologías tradicionales y ágiles.

2.4 Scrum

La metodología Scrum es ampliamente reconocida como una de las metodologías líderes en el desarrollo ágil de software a nivel global. Su origen se remonta a la década de 1980, cuando Ikujiro Nonaka e Hirotaka Takeuchi realizaron análisis de la importancia del trabajo en equipo en el desarrollo de productos y la autonomía que se debe otorgar a los equipos (Takeuchi & Nonaka, 1986, citado por Rodríguez & Vicente, 2015, p.133). Posteriormente, a comienzo de los años 90, Jeff Sutherland y Ken Schwaber retomaron estos conceptos y formalizaron un marco de trabajo con reglas específicas, diseñadas principalmente para el desarrollo de productos de software complejos (Schwaber & Sutherland, 2012, citado por Rodríguez & Vicente, 2015, p.133).

Scrum, como una metodología de desarrollo ágil, se fundamenta en la idea de crear ciclos cortos para el desarrollo, que generalmente son iteraciones, y en Scrum se les conoce como “Sprints” (Trigás, 2012, p. 33).

Para comprender el ciclo de desarrollo de Scrum, debemos conocer las 5 principales fases del ciclo de desarrollo ágil.

1. Concepto. Se define las características del producto de forma general y la asignación sobre el desarrollo del producto.
2. Especulación. Se realiza la toma de decisiones basadas en la información recopilada y se define los límites que guiarán el desarrollo del producto, como los costos y los plazos. Se construye el producto a partir de las ideas principales y se verifica cómo se afectan las partes realizadas al entorno. Esta fase se repite en cada iteración.
 - a. Desarrollar y revisar los requisitos generales.
 - b. Mantener la lista de las funcionalidades que se esperan.
 - c. Establecer las fechas de las versiones, hitos e iteraciones.
3. Exploración. Se amplía el producto incorporando las funcionalidades propuestas en la fase de Especulación.
4. Revisión. El equipo revisa todo lo que se ha construido y lo compara con el objetivo deseado.
5. Cierre. Se entrega una versión del producto, además que el cierre no implica la finalización del proyecto, ya que aún habrá cambios continuos, conocidos como “mantenimiento”, que acercarán el producto final al estado deseado.

Figura 2.2

Ciclo de desarrollo ágil



Nota. Las 5 fases importantes del desarrollo ágil. Tomado de (Trigás, 2012)

En la actualidad, Scrum se utiliza en diversos ámbitos y organizaciones, siendo especialmente relevante en el desarrollo de software. Esta metodología es objeto de opiniones encontradas, ya que su característica más destacada, las reuniones de equipo frecuentes, puede ser tanto apreciada como cuestionada por los profesionales involucrados. Esto se debe a que, en el contexto de los equipos de desarrollo de software, la implementación de Scrum puede ser un desafío debido a la falta de flexibilidad inherente, que puede variar dependiendo del proyecto en particular.

2.5 Kanban

Kanban es una forma simple de concebirlo expresándose como: “Con Kanban, puedes gestionar el trabajo”. Este enfoque se extiende a la administración de diversos servicios profesionales, incluso aquellos relacionados con el trabajo del conocimiento. La adopción de este método implica una perspectiva integral hacia los servicios, centrada en su mejora desde la óptica de los clientes.

El método Kanban implica la visualización del trabajo y su flujo a lo largo del proceso de trabajo, lo que facilita una gestión eficaz de las operaciones comerciales, incluyendo la identificación y gestión de riesgos asociados a la prestación de servicios a los clientes. Con el tiempo, el uso de Kanban permite el desarrollo de una capacidad adaptativa para responder de manera más ágil y eficiente a los cambios en las necesidades y expectativas de los clientes o en el entorno empresarial.

Kanban a menudo se confunde con una metodología o un marco de trabajo. En el contexto de la Ingeniería de Software, una metodología se refiere a una aproximación para definir los procesos de desarrollo de software y la gestión de proyectos. El término “metodología” puede ser engañoso, ya que en realidad se trata del estudio de métodos. Las metodologías en este campo suelen contener flujos de trabajo y procesos bien definidos y prescriptivos, que incluyen roles y responsabilidades específicas. En consecuencia, tienden a ser altamente específicas para un dominio particular, como el desarrollo de software.

Por otra parte, un marco de trabajo es un espacio de metodología incompleta. Se compone de estructuras generales destinadas a tener una aplicación más amplia, pero que requieren personalización y adaptación para encajar en un contexto específico (Group, 2021).

El método Kanban se basa en un mecanismo que implica la gestión de todas las tareas mediante un tablero Kanban. En este tablero, se limita la cantidad de tareas o trabajos que se pueden llevar a cabo al mismo tiempo. Estas tareas se organizan de mayor a menor en términos de la cantidad de trabajo que implican en el sistema. El objetivo principal de este enfoque es mejorar el flujo de valor hacia los clientes.

En el marco del método Kanban, puede haber uno o más individuos trabajando para lograr la entrega de un producto. El cliente solicita el trabajo en función de las necesidades identificadas

y es el cliente quien aprueba y acepta el trabajo una vez que sea ha completado. (Anderson & Carmichael, 2016, p. 12).

Figura 2.3

Ejemplo de tablero Kanban



Nota. Ejemplo de los componentes de un tablero Kanban. Tomado de (Anderson & Carmichael, 2016),

2.6 Scrumban

Scrumban combina los conceptos clave de un equipo trabajando en conjunto para completar un trabajo, como en Scrum, con la gestión de la carga de trabajo necesaria para alcanzar una eficiencia óptima, tal como se ve en Kanban. Esta metodología busca proporcionar alto alcance y visibilidad en el proceso de desarrollo. (Swisher, 2014, citado por Flores Leyva, 2014, p.42). El escenario de la metodología Scrumban conjunta los elementos comunes de estas dos metodologías como: tablero y tareas, además de tomar los elementos únicos de Kanban como el límite de trabajo en proceso y prioridades y los elementos de Scrum, el backlog, los sprints y las

revisiones diarias y mensuales. (Bicheler, Gómez A. & Palacin R. , 2012, citado por Flores Leyva, 2014).

2.6.1 Flujo de trabajo

Scrumban combina los conceptos clave de un equipo trabajando en conjunto para completar un trabajo, como en Scrum, con la gestión de la carga de trabajo necesaria para alcanzar una eficiencia óptima, tal como se ve en Kanban. Esta metodología busca proporcionar alto alcance y visibilidad en el proceso de desarrollo. (Swisher, 2014, citado por Flores Leyva, 2014, p.42). Scrumban tiene un flujo de trabajo híbrido que principalmente es abordado por equipo de trabajo que prefieren una estructura similar a la metodología Scrum ya que permite una entrega regular y tiene la facilidad adaptarse a pequeñas desviaciones causadas por tareas urgentes. Además, incorpora ciertos elementos de la metodología Kanban.

Scrumban combina la estructura de Scrum con la flexibilidad de Kanban, permitiendo favorecerse de los beneficios de ambos enfoques. A diferencia de otros métodos, La metodología Scrumban dispone una guía de procesos, pero también brinda la libertad de que el equipo puede asignar de manera personalizada para mejorar su forma de trabajar (Bhavna, 2022).

2.6.2 Características

El enfoque Scrumban combina las ventajas de ambas metodologías y las aplica en diferentes momentos durante el desarrollo de un proyecto. Scrum se utiliza para agilizar la fase de diseño del producto o servicio, especialmente en tareas que se ejecutan una sola vez y se revisan solo para mejoras en los entregables. No obstante, una vez que se inicia la producción en cadena, Kanban toma un papel más protagónico, contribuyendo a mantener una fluidez en todo el proceso de producción. A medida que el proyecto se acerca a la fase de producción pura, Scrum

cede ante Kanban. En esta etapa, Scrumban se enfoca en el tiempo de entrega promedio y el tiempo de ciclo de producción.

- Mejora continua en el proceso. (Kanban)
- Perfeccionamiento continuo en el valor entregado al cliente. (Scrum)
- Adaptación al cambio y flexibilidad.
- Efectividad y eficacia en los procesos.
- Utilización de las materias primas, con el mínimo de desperdicio.
- Agilidad en el tiempo.

Complementando, Scrum se enfoca en proporcionar valor constante a los usuarios y clientes finales de manera efectiva, mientras que Kanban se centra en mejorar continuamente los procesos. (Mancuzo, 2021).

A continuación, se presenta un cuadro de comparación entre las metodologías ágiles más relevantes, en la siguiente Tabla 2.

Tabla 2.2

Comparación entre metodologías

	Scrum	Kanban	Scrumban
Principio	Mejora constante del producto	Desarrollo continuo del proceso	Mejora el producto y el proceso
Etapas	Diseño	Producción	Diseño y producción
Proceso	Iterativo - Incremental (Cada sprint tiene un principio y fin)	Ciclo continuo: El flujo es constante, no termina nunca	Iterativo e incremental, forma continua
Estimación	Predeterminadas	Sin estimaciones	Trabajo continuo

Cambios	Debe esperar al siguiente Sprint	Se agendan en el tablero	Se agenda según aparecen en el tablero
Backlog	Lista predefinida del Sprint	Individualidades en cada tarjeta	Tarjetas JIT (Just in Time)

Nota. Comparación de las principales características entre las metodologías. Tomado de (Mancuzo, 2021)

2.6.3 Backlog en scrumboard

El proceso iterativo e incremental de Scrumban se basa en planificar Sprints en intervalos regulares, sincronizados con revisiones y retrospectivas. Sin embargo, el objetivo de la planificación es llenar los espacios disponibles a medida que se despejan o surgen nuevas necesidades de productos. Esto contrasta con el enfoque de Scrum puro, donde se completan todos los elementos antes de comenzar el proceso con todos los espacios disponibles. Además, en Scrumban, no se determina previamente el número de espacios, sino que se generan a medida que se necesitan. Esto conlleva una reducción significativa en los costos generales y en la planificación de Sprints. (Mancuzo, 2021).

Para comprender mejor este sistema que fusiona dos metodologías ágiles, es fundamental conocer cómo gestionar el backlog en el Scrumboard. Algunos elementos clave a considerar son:

- Evitar crear o analizar demasiadas historias de usuario para reducir el desperdicio.
- Asegurar un nivel de análisis adecuado antes de iniciar el desarrollo.
- Acumular solo cuando se ha recibido un nuevo pedido en el tablero.

2.7 Beneficios y conclusiones de Scrumban

Indudablemente, la combinación de Kanban y Scrum potencia las características de ambos modelos. Scrumban tiene como objetivo principal establecer una gestión de proyectos que sea efectiva, eficiente, ágil y orientada hacia el valor y la integración del cliente. Los beneficios que se pueden obtener mediante la implementación de Scrumban, aprovechando la sinergia de ambos enfoques, incluyen:

- Calidad mejorada: Se enfoca en la calidad constante en todo el proceso.
- Just-in-time: Decisiones y tareas se toman justo en el momento en que son necesarias.
- Plazos de ejecución más cortos: Utiliza el método Kaizen de mejora continua.
- Minimización del desperdicio: Elimina cualquier actividad que no aporte valor al cliente.
- Mejora de procesos: Incorpora elementos de Scrum al flujo de Kanban cuando es necesario.

La metodología Scrumban combina la estructura organizativa de Scrum con los métodos basados en tarjetas, solicitudes, ciclos y visualización de Kanban. Para implementar con éxito esta metodología, es esencial contar con un software de gestión de proyectos que permita la visualización de tableros Kanban. Herramientas como Monday, Trello o Notion facilitarán la creación de tableros Scrumban con columnas personalizadas, gráficos y métricas ágiles para una gestión más eficiente del proyecto. (Mancuzo, 2021).

2.8 Metodología UWE

Según el sitio oficial de UWE, esta metodología se caracteriza por proporcionar una notación especializada en el ámbito de desarrollo de aplicaciones web, junto con un proceso de desarrollo basado en modelos y herramientas de apoyo. Una de las particularidades más destacadas de

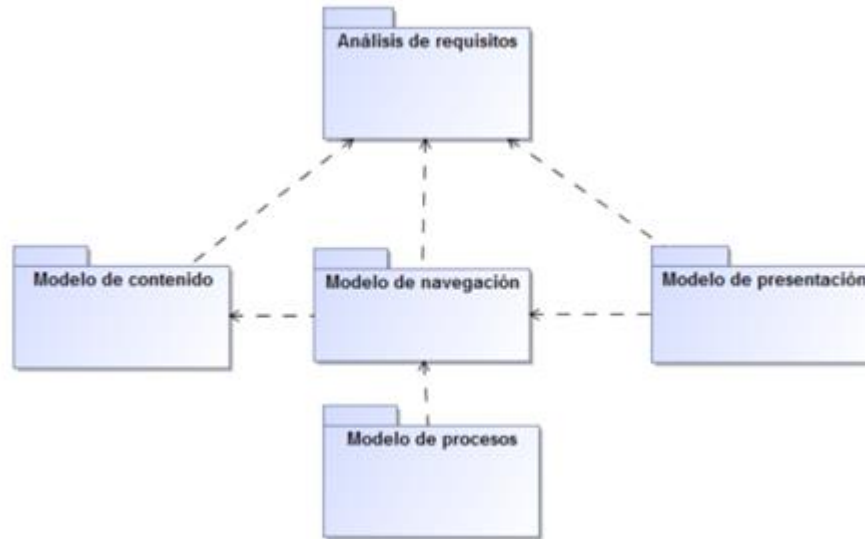
UWE es su adhesión a estándares, que no se limita únicamente al uso de UML como lenguaje común, sino que también incluye el empleo de XMI para el intercambio de modelos, MOF para el meta-modelado, principios derivados del enfoque MDA, el lenguaje QVT para transformación de modelos y XML.

Las razones fundamentales para preferir los mecanismos de extensión de UML en lugar de técnicas de modelado propietarias son varias: la aceptación generalizada de UML en el desarrollo de sistemas de software, la capacidad de definir un lenguaje de modelado específico para el dominio web mediante lo que se conoce como perfil UML y el amplio soporte de herramientas de modelado visual a través de las actuales herramientas CASE basadas en UML.

En el contexto de UWE, se opta por utilizar la notación UML en su forma más pura y los tipos de diagramas UML siempre que sea factible durante el análisis y diseño de aplicaciones web, es decir, sin introducir extensiones innecesarias. Sin embargo, para abordar las particularidades propias de la web, como los nodos y enlaces en la estructura de hipertexto, el perfil UWE incorpora estereotipos, valores etiquetados y restricciones específicas para los elementos del modelo. La extensión UWE abarca aspectos relacionados con la navegación, presentación, procesos empresariales y adaptación, entonces en un término corto la metodología UWE, es la notación se concibe como una extensión "ligera" de UML.

Figura 2.4

Modelo de UWE.



Nota. Representación gráfica del modelo UWE. Tomado de (Nieves Guerrero et al., 2014, p.138).

2.8.1 Fases de desarrollo

UWE (Unified Web Engineering) es una metodología que se utiliza para especificar de manera más efectiva una aplicación web durante su proceso de creación. Esta metodología se basa en el uso de UML (Unified Modeling Language) para sus modelos y métodos, lo que facilita la transición y comprensión en el desarrollo de aplicaciones web.

UWE establece una clara definición de cómo construir cada uno de los elementos del modelo en el proceso de desarrollo. Su implementación implica las siguientes etapas y modelos:

- **Análisis de Requisitos:** Esta etapa se encarga de plasmar los requisitos funcionales de la aplicación web mediante un modelo de casos de uso. Aquí se identifican las principales interacciones entre el usuario y el sistema.
- **Modelo de Contenido:** En esta fase, se define con detalle los conceptos involucrados en la aplicación web utilizando un diagrama de clases. Esto permite tener una comprensión profunda de la estructura de datos y objetos necesarios.

- **Modelo de Navegación:** El modelo de navegación representa cómo los usuarios se mueven a través de la aplicación web. Incluye la definición de índices, menús, consultas y otros elementos de navegación.
- **Modelo de Presentación:** En esta etapa, se representan las interfaces de usuario mediante vistas abstractas. Esto ayuda a visualizar cómo se verá la aplicación para los usuarios.
- **Modelo de Proceso:** Este modelo representa las actividades y procesos que se relacionan con cada clase de proceso en la aplicación web. Ayuda a comprender cómo funcionan las distintas partes del sistema.

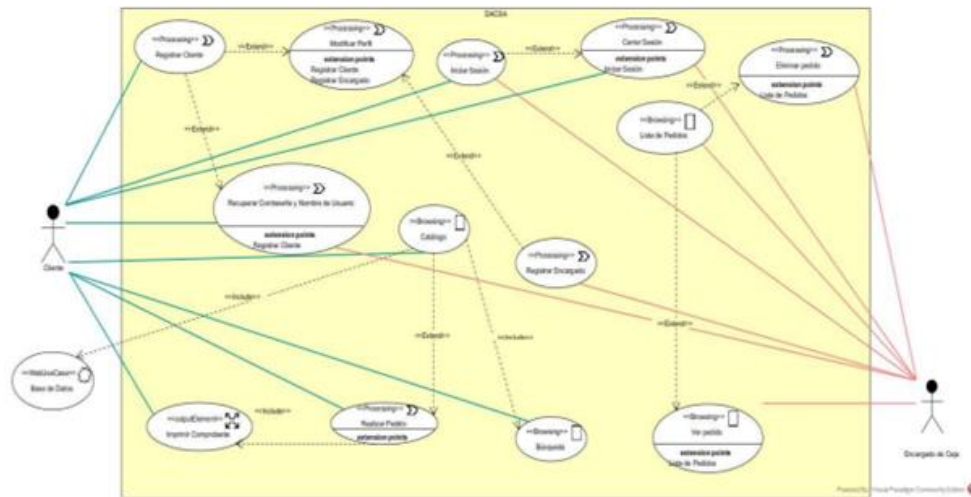
Como se observa, la metodología UWE provee distintos modelos que permite describir una aplicación Web desde varios puntos de vista abstractos, dichos modelos están relacionados tal como se muestra en la Figura 4. Cada uno de estos modelos se representa como paquetes UML, dichos paquetes son procesos relacionados que pueden ser refinados en iteraciones sucesivas durante el desarrollo del UWE. (Nieves Guerrero et al., 2014)

2.8.2 Modelo de requisitos

La primera etapa en el desarrollo de aplicaciones web es la identificación de los requisitos, y en UWE, estos requisitos se especifican mediante el modelado de requisitos funcionales, que se lleva a cabo a través del modelado de casos de uso utilizando UML, como se muestra en la Figura 2.5, a continuación.

Figura 2.5

Diagrama de Requerimientos de Modelo de Requisitos.



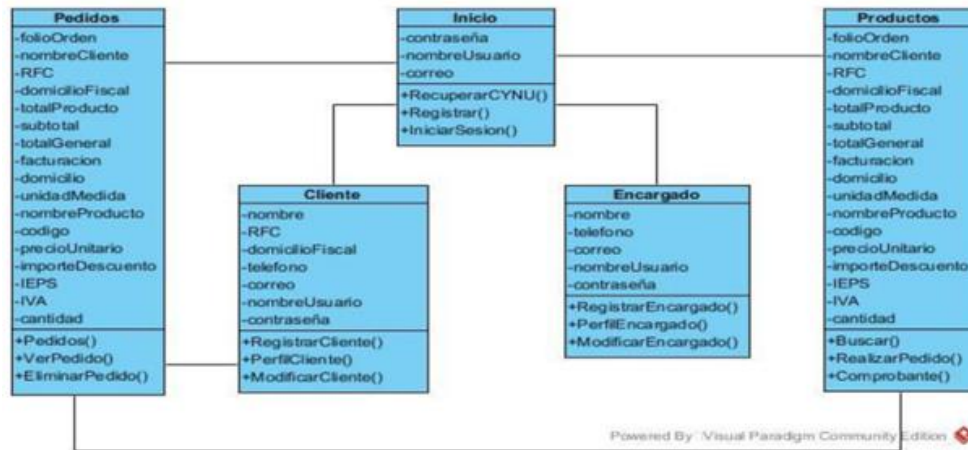
Nota. Casos de uso de la metodología UWE. Tomado de (Instituto de Estudios Superiores del Valle de Orizaba, S. C., 2019)

2.8.3 Modelo de contenido

El modelo de contenido tiene como objetivo brindar una representación visual de la información en el dominio de datos que es relevante para la aplicación web. Este tipo de información se describe utilizando un diagrama UML estándar de clases, como se ilustra en la Figura 2.6. En este contexto, se detallan las clases necesarias específicas para el caso de estudio que se está presentando.

Figura 2.6

Diagrama de Clases



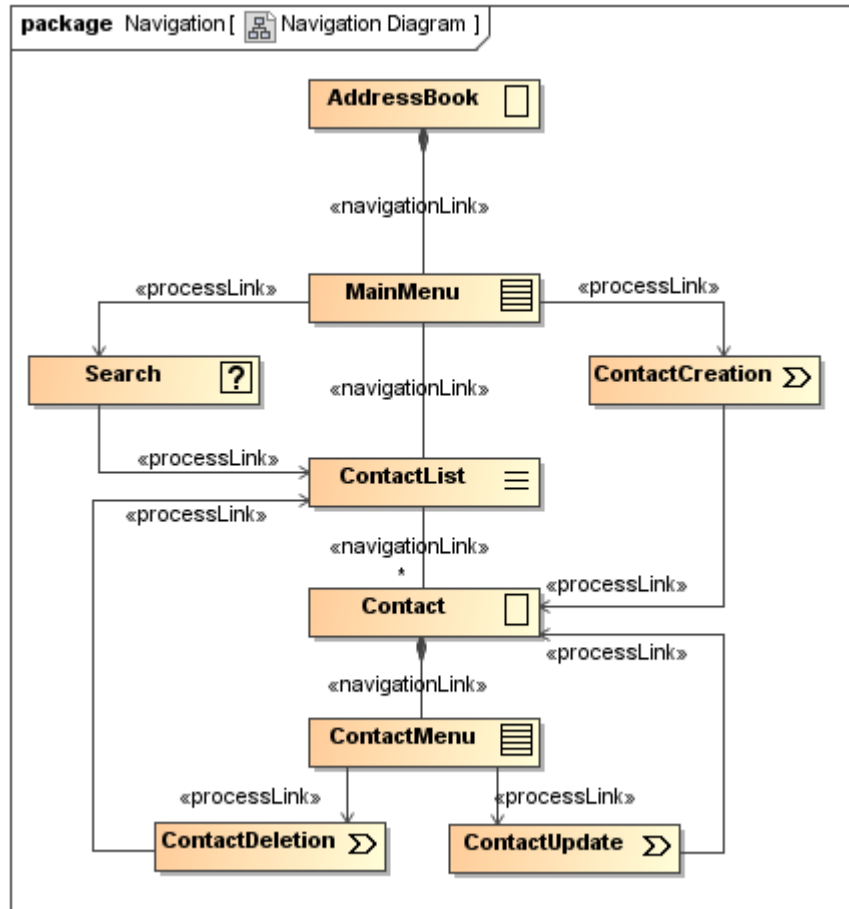
Nota. Ejemplo de un diagrama de clases. Tomado de (Instituto de Estudios Superiores del Valle de Orizaba, S. C., 2019)

2.8.4 Modelo de navegación

Este modelo se fundamenta en la combinación del modelado de requisitos y el modelado de contenido. En el caso de las clases del modelo de contenido que tienen relevancia para la navegación, se integran en el modelo de navegación junto con sus asociaciones, representando así los navigationClass y navigationLinks.

Figura 2.7

Diagrama de clases navegacional



Nota. Modelo de Navegación de UWE. Tomado de (Guerrero, Pech y Mendez, 2014)

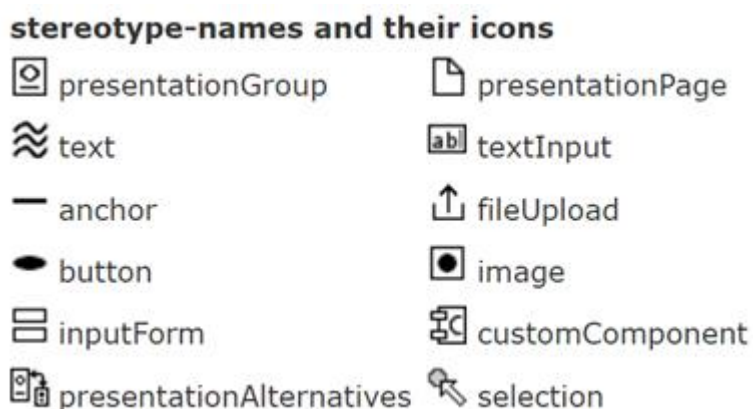
2.8.5 Modelo de presentación

El modelo de presentación ofrece una perspectiva abstracta y describe la estructura fundamental de la interfaz de usuario (UI) de una aplicación web. Este modelo se basa en el modelo de navegación, y los elementos que representa se utilizan para presentar los nodos de navegación. Cada atributo del elemento <navigationClass> se representa en el modelo de presentación mediante un elemento de la interfaz de usuario correspondiente. Por ejemplo, un elemento "next" se utiliza para representar el atributo "titulo" del <navigationClass>, mientras que un elemento "image" se usa para representar el atributo "foto".

Generalmente, el contenido de diferentes <navigationNodes> se presenta en una página web, que se denomina <presentationPage> en UWE. En este modelo, se representan las clases de navegación y procesos que están asociadas con cada página web. Estos elementos son introducidos por la metodología UWE en este modelo de desarrollo.

Figura 2.8

Modelo de presentación

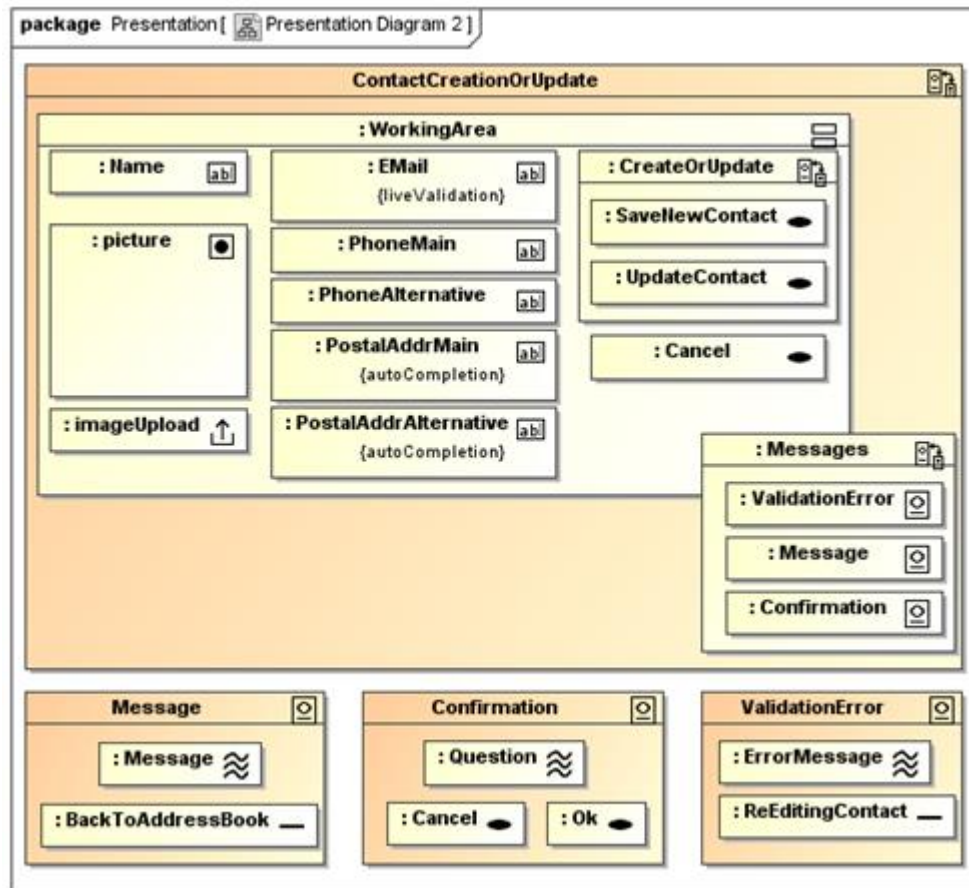


Nota. Estereotipo del Modelo de Presentación Tomado por (Guerrero, Pech y Mendez, 2014)

En la Figura 2.9, se presenta el enfoque de trabajo del modelo de presentación en el contexto del modelado UWE, que se emplea en el desarrollo del proyecto.

Figura 2.9

Modelo de presentación



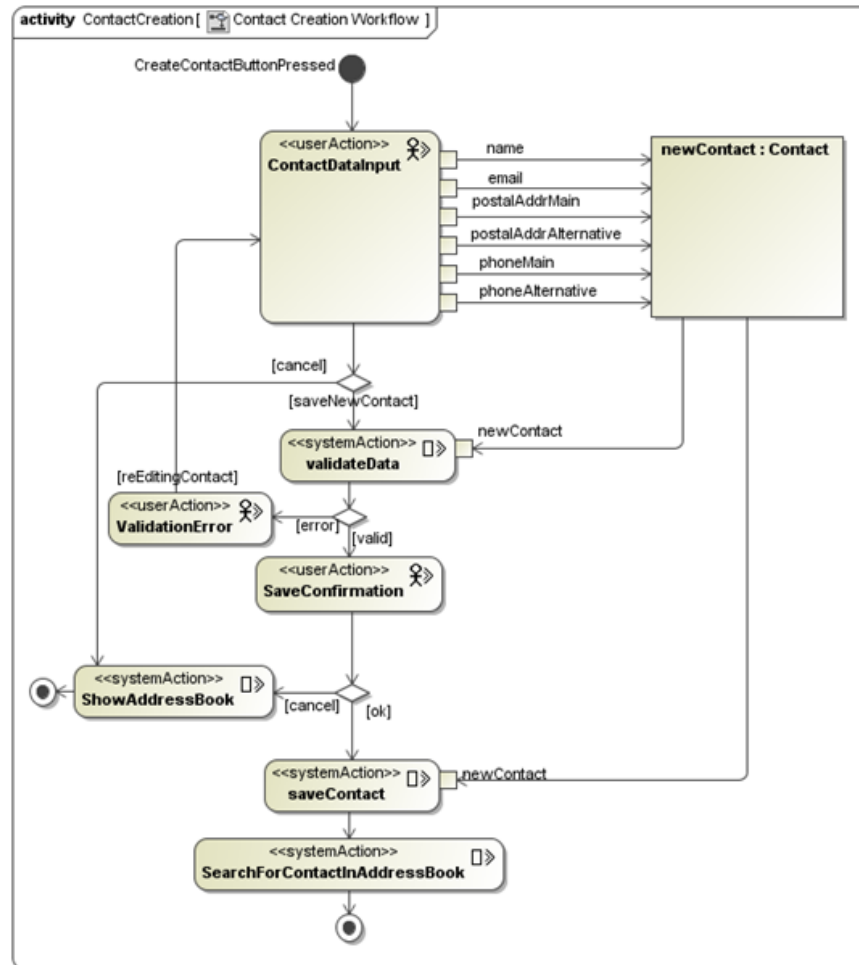
Nota. Ejemplo de Modelo de Presentación Tomado por (Guerrero, Pech y Mendez, 2014)

2.8.6 Modelo de proceso

El sitio (Web Engineering Group, 2014) sostiene que este modelo proporciona los componentes esenciales para representar los procesos de negocio en un modelo UWE. A diferencia del modelo de navegación, que se encarga de representar la estructura estática de la aplicación web, el modelo de proceso aborda la parte dinámica de la misma. Este modelo incluye ciertos estereotipos, como se ilustra en la Figura 2.10.

Figura 2.10

Modelo de proceso



Nota. Ejemplo de modelo de flujo de proceso de UWE, Tomado por (Guerrero, Pech y Mendez, 2014)

2.9 Herramientas de desarrollo

En esta sección se presenta las herramientas de desarrollo que se utilizarán en el presente proyecto.

2.9.1 Herramientas de Frontend

2.9.1.1 Framework Angular

Angular es una plataforma de desarrollo y un framework de diseño que se utiliza para crear aplicaciones de una sola página altamente eficientes y avanzadas. Este framework se basa en el

patrón de diseño Modelo-Vista-Controlador (MVC), que es fundamental en el desarrollo de aplicaciones de una sola página. Esta característica permite aprovechar la reutilización de funciones en distintas partes de la aplicación mediante el uso de componentes, lo que simplifica su mantenimiento y mejora su legibilidad gracias a su estructura bien definida. Además, Angular facilita la incorporación de nuevas funcionalidades y la realización de pruebas unitarias de manera más eficiente.

2.9.1.2 PrimeNG

PrimeNG es una biblioteca de código abierto diseñada para Angular, que brinda a los desarrolladores una amplia gama de componentes de interfaz de usuario. Estos componentes abarcan desde tablas y menús hasta botones y calendarios, lo que facilita la creación de aplicaciones web atractivas y de fácil uso.

Una de las ventajas más destacadas de PrimeNG es su capacidad para integrarse con los estilos de Material Design de Google. Además, ofrece compatibilidad con Bootstrap, lo que permite diseñar aplicaciones con una apariencia moderna y coherente, de acuerdo con las preferencias y necesidades específicas del proyecto. Además, la biblioteca es altamente personalizable, lo que significa que los desarrolladores pueden adaptar los componentes según los requisitos precisos de su proyecto.

2.9.2 Herramientas de backend

2.9.2.1 Framework Flask

Flask es un microframework diseñado para agilizar la creación de aplicaciones web. Su enfoque es proporcionar solo las funciones esenciales, permitiendo a los desarrolladores añadir características adicionales según sea necesario durante el proceso de desarrollo. Flask es una

opción liviana en el ámbito de los marcos de aplicaciones WSGI y puede ser utilizado tanto en el lado del servidor como en el lado del cliente, si es necesario. Entre sus características se incluyen un depurador interactivo, un objeto completo de solicitud, un sistema de enrutamiento para gestionar los puntos finales, utilidades para el manejo de etiquetas de entidad, control de caché, fechas, cookies, entre otros elementos esenciales. Además, Flask ofrece un servidor WSGI enroscado para facilitar el desarrollo local y un cliente de prueba para simular las solicitudes HTTP. En su núcleo, Flask se basa en las bibliotecas Werkzeug y Jinja.

Jinja, por otro lado, es una dependencia clave que tiene Flask. Funciona como un motor de plantillas completo, ofreciendo una serie de características avanzadas, como la ejecución segura en un entorno aislado, una sólida protección contra ataques XSS, la capacidad de heredar plantillas y una fácil depuración, entre otras. Además, el código que se escribe en las plantillas HTML se compila como código Python.

Dado que Flask suele ser considerado un marco para prototipos, no incluye una capa de abstracción para la base de datos ni proporciona validación o seguridad, lo que brinda a los implementadores total libertad para agregar los requisitos necesarios. Además, existen extensiones disponibles para Flask que incluyen bibliotecas como gunicorn para el servidor, SQLAlchemy para la base de datos, Alembic para la gestión de migraciones de bases de datos, Celery y Redis para la ejecución de tareas asíncronas, Flask-WTF para validar formularios y Flask-limiter para limitar las solicitudes web. Flask es compatible con Python 3 y versiones posteriores, y se puede instalar fácilmente con el gestor de paquetes oficial de Python, pip.

(Ghimire, 2020)

2.9.2.2 JWT (Json-Web-Token)

JWT (JSON Web Token) es un estándar que se encuentra documentado en el RFC 7519.

Este estándar establece un mecanismo para transmitir la identidad de un usuario de manera segura entre dos partes. Además, permite incluir una serie de reclamaciones o privilegios en el token. Estos privilegios se codifican en objetos de tipo JSON y se incorporan en el cuerpo o payload de un mensaje que está firmado digitalmente.

Figura 2.11

Estructura de token JWT

The image shows the JWT.io website interface. On the left, under the 'Encoded' tab, a JWT token is pasted: `eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36P0k6yJV_adQssw5c`. On the right, under the 'Decoded' tab, the token's structure is displayed. The header is `{ "alg": "HS256", "typ": "JWT" }`. The payload is `{ "sub": "1234567890", "name": "John Doe", "iat": 1516239022 }`. The signature verification section shows the formula: `HS256(base64UrlEncode(header) + "." + base64UrlEncode(payload), your-256-bit-secret)`, with a checkbox for 'secret base64 encoded'.

Nota. Ejemplo de la estructura de token con JWT. Tomado del sitio oficial JWT.

2.9.2.3 Postman

Postman es una herramienta ampliamente empleada, principalmente en el contexto de pruebas de API REST, aunque ofrece funcionalidades adicionales que van más allá de la simple evaluación de este tipo de sistemas.

Gracias a esta herramienta, se pueden llevar a cabo diversas actividades, que incluyen la prueba, consumo y depuración de API REST. Además, Postman brinda la capacidad de monitorizar

estas APIs, escribir pruebas automatizadas para las mismas, documentarlas, crear simulaciones y mocks, entre otras funcionalidades (López, 2019).

2.9.2.4 PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos de código abierto altamente confiable que brinda un sólido soporte para diversas funciones de SQL, como claves foráneas, subconsultas, disparadores, y la capacidad de definir tipos y funciones personalizadas por el usuario. Además de cumplir con los estándares SQL, PostgreSQL enriquece el lenguaje SQL al ofrecer una amplia gama de funciones que permiten manejar y optimizar cargas de trabajo de datos de manera meticulosa. Características Clave de PostgreSQL

PostgreSQL se destaca por sus características únicas que la convierten en una base de datos ampliamente preferida, siendo la segunda más utilizada después de MySQL.

- **Fiabilidad y Cumplimiento de Normas:** PostgreSQL ofrece una sólida semántica ACID para transacciones y soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados en varios lenguajes. Admite diversos tipos de datos, desde INTEGER y VARCHAR hasta TIMESTAMP y BOOLEAN. También puede manejar objetos binarios de gran tamaño, como imágenes o vídeos. Su fiabilidad se respalda con una extensa comunidad de soporte y su capacidad de tolerancia a fallos gracias al registro de escritura anticipada.
- **Extensiones:** PostgreSQL incluye potentes características como recuperación puntual, Control de Concurrencia Multiversional (MVCC), tablespaces, controles de acceso granulares, replicación asincrónica, un planificador/optimizador de consultas avanzado y registro de escritura anticipada. El Control de Concurrencia Multiversional permite lectura y escritura concurrente de tablas, minimizando bloqueos y conflictos.

- Escalabilidad: PostgreSQL es altamente escalable, admitiendo Unicode, conjuntos de caracteres internacionales y múltiples codificaciones de caracteres. Puede manejar una gran cantidad de usuarios concurrentes y cantidades significativas de datos. Además, es compatible con múltiples sistemas operativos, como Linux, Microsoft Windows, OS X, FreeBSD y Solaris.
- Carga Dinámica: El servidor PostgreSQL permite a los usuarios cargar código personalizado sobre la marcha. Esto significa que los usuarios pueden especificar archivos de código objeto, como bibliotecas compartidas que implementen nuevas funciones o tipos, y PostgreSQL los cargará según sea necesario. Esta capacidad de adaptación en tiempo real lo hace especialmente adecuado para implementar rápidamente nuevas estructuras de almacenamiento y aplicaciones.

2.10 Calidad de software

La calidad de Software se compone de las características distintivas que se definen según su valor y viabilidad. Esta excelencia se relaciona con la eficacia, versatilidad, exactitud, exactitud, confiabilidad, capacidad de mantenimiento, adaptabilidad, facilidad de uso, seguridad y la integridad del software (García León & Beltrán, 1995).

El principal objetivo que apunta en los sistemas de información está orientado a:

- Mejorar la calidad del producto
- Proveer técnicas aplicadas para automatizar el manejo de datos
- Validar y controlar formalmente la calidad del trabajo realizado
- Cumplir los objetivos de la organización en cuanto a productividad de subsistemas de cómputo.

Según se menciona en el libro 'Ingeniería de Software' de Pressman (1993), en la industria del software se pueden identificar requisitos relacionados con la satisfacción del cliente en cuanto a productos o servicios de software, la eficiencia en la gestión de recursos en proyectos de desarrollo de software y una adecuada asignación de recursos humanos. En lo que concierne a la calidad del software, se ha empleado históricamente una métrica que evalúa la cantidad de defectos por cada mil líneas de código como una de las primeras aproximaciones para su evaluación.

2.10.1 Métricas de software

Es una aplicación que se trata de una práctica constante de evaluaciones en el transcurso del desarrollo de software y de sus resultados, con el propósito de proporcionar datos pertinentes oportunamente, con el fin de mejorar tanto los procedimientos como los productos. En lo que concierne a las métricas de software.

En el texto 'Métricas de calidad de software' de Pereira, Ayaach, Quintero, Granadillo y Bustamante (2012), se presenta una clasificación de las métricas de Software, que se detalla en la Tabla 2.3.

Tabla 2.3

Clasificación de Métricas de Software

Métrica de Software	Descripción
De complejidad	Métricas que definen la medición de la complejidad en volumen, tamaño, anidaciones y configuración.

De calidad	Métricas que definen la calidad del software en exactitud, estructuración o modularidad, prueba y mantenimiento.
De competencia	Métricas que intentan valorar o medir las actividades de productividad de los programadores con respecto a su certeza, rapidez, eficiencia y competencia.
De desempeño	Métricas que miden la conducta de módulos y sistemas de un software bajo la supervisión de SO o hardware.
Estilizadas	Métricas de experimentación y de preferencia: estilo de convenciones, limitaciones, etc.

Nota. Recuperado de (García León & Beltrán, 1995).

2.10.2 Métricas de calidad

Las métricas del proceso de desarrollo de software y las métricas del producto representan una herramienta cuantitativa que brinda a los profesionales del software una comprensión detallada de la eficiencia del proceso de desarrollo. Estas métricas también desempeñan un papel crucial en la identificación de áreas de mejora y en la implementación de soluciones para optimizar el proceso de desarrollo de software. (Pressman, 2003).

Existen diferentes modelos que permiten medir la calidad del software, se presenta en la siguiente Tabla 2.4.

Tabla 2.4*Modelos de Calidad de Software*

Calidad de software	Descripción
Modelo MCCALL (1977)	<p>Se describe como la calidad en un concepto elaborado mediante relaciones jerárquicas entre factores de calidad, en base a tres criterios:</p> <ul style="list-style-type: none">• Características operativas• Capacidad de cambios• Adaptabilidad a nuestros entornos
Modelo de FURPS (1987)	<p>Se desarrollado por Hewlett-Packard (HP), se utiliza para establecer métricas de calidad para todas las actividades del proceso de desarrollo de un software, inclusive de un sistema de información.</p>
Modelo de DROMEY (1996)	<p>Se resalta el hecho de que la calidad del producto es altamente determinada por los componentes del mismo (Incluyendo documentos de requerimientos, guías de usuarios, diseños y código).</p>
Normas ISO 9000: ISO/IEC 9126	<p>El estándar que establece que cualquier componente de la calidad del software puede ser descrito en términos de una o más de las siete características básicas:</p> <ul style="list-style-type: none">• Funcionalidad• Confiabilidad• Usabilidad• Eficiencia• Mantenibilidad• Portabilidad• Satisfacción;

cada una de ellas se detalla a través de un conjunto de subcaracterísticas que permiten profundizar en la evaluación de la calidad de productos de software.

Modelo sistemático de calidad (MOSCA) Es un conjunto de categorías, características y métricas asociadas que miden la calidad de un proceso de software con un enfoque sistémico.

Nota. Recuperado de (Pressman, 2003).

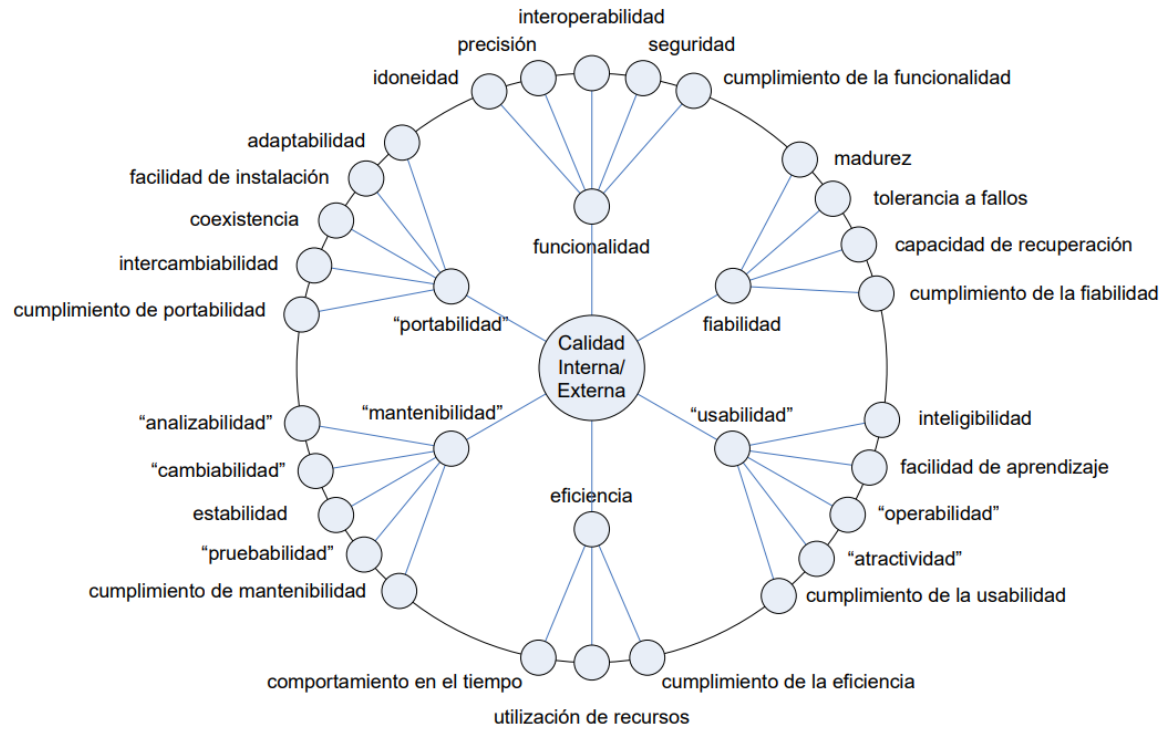
2.10.2.1 Norma ISO 9126

En 1991, la Organización Internacional de Normalización (ISO) presentó su primer modelo de calidad destinado a evaluar productos de software bajo la norma ISO 9126:1991. A lo largo de los años, este estándar sufrió múltiples revisiones, evolucionando hasta llegar a la norma ISO/IEC 9126 actual titulada "Ingeniería de Software. Calidad del Producto". Esta norma ISO/IEC 9126 introduce un conjunto de características, subcaracterísticas y atributos diseñados para analizar la calidad de un producto de software. En concreto, define seis propiedades principales: funcionalidad, fiabilidad, usabilidad, eficiencia, mantenibilidad y portabilidad. (Marcos et al., 2008)

Estas propiedades se dividen en subcategorías, como se representa en la Figura 2.12.

Figura 2.12

Características de la calidad interna y externa según la ISO/IEC 9126.



Nota. Tomado de (Marcos et al., 2008)

CAPITULO 3 MARCO APLICATIVO

3.1 Introducción

En este capítulo, se abordará el análisis y el diseño del Sistema de Gestión y Seguimiento Académico, siguiendo el enfoque y la ejecución de la metodología Scrumban, la cual se explicó en detalle en el capítulo anterior.

Es importante destacar que Scrumban no es una metodología rígida, sino que, al igual que otras metodologías ágiles, se adapta de forma constante a las circunstancias que surgen durante el desarrollo del proyecto.

El proceso de inicio del proyecto consistirá en realizar el análisis y la selección de los requisitos necesarios. En cada iteración o sprint, se elegirán partes específicas del sistema del proyecto para su implementación, teniendo en cuenta tanto el proceso de modelado como el tiempo de trabajo necesario.

Además, en el desarrollo del sistema, se empleará la metodología de modelado UWE, la cual también se presentó en el capítulo anterior.

3.2 Especificaciones

La Especificación de Requisitos de Software (ERS) es una descripción exhaustiva del comportamiento del sistema que se va a desarrollar. Durante la fase de especificaciones, se recopilaron las disposiciones y requisitos necesarios para la ejecución del proyecto. En esta etapa, se revaluó la utilidad de las historias de usuario en la pila de productos (Product backlog), que constituye el componente principal de este proyecto. Además, siguiendo la metodología de modelado UWE (Uml-Based Web Engineering), se decidió reemplazar las historias de usuarios con diagramas de casos de uso especificados en UWE.

Una vez definidos la mayoría de los casos de uso para la construcción del sistema, se llevaron a cabo reuniones de planificación con la organización interesada. Posteriormente, se seleccionaron algunos casos de uso para su desarrollo.

Para cada iteración (Sprint), se detallan en tablas las diferentes actividades necesarias para el desarrollo e implementación del sistema. Cada una de estas actividades corresponde a una tarea en el backlog product.

A través de una reunión con las partes interesadas, se logró definir una lista de tareas que contribuyeron a establecer los requisitos del sistema a un nivel general. La idea central para el desarrollo del sistema consiste en crear una herramienta que gestione la información de la

organización, mediante los siguientes módulos: académico, materia, curso, pago, calificación, asistencia y reportes. Esto permitirá realizar el registro de la información relacionada entre estos módulos mencionados de los diferentes roles del sistema y, por ende, la interacción con los diversos módulos a implementar.

Además, el sistema proporcionará diferentes tipos de reportes dependiendo del módulo. Por ejemplo, para el módulo de pago, se generará un reporte de lista de pagos, historial de pagos, entre otros reportes descargables. Asimismo, habrá reportes específicos para cada rol; por ejemplo, el rol de estudiante y docente contará con reportes que mostrarán las materias en las que están involucrados, ya sea como docente o estudiante, por curso y materia. Esto se hace con el fin de facilitar la obtención de información en tiempo real sobre el estado de pagos, notas y otros aspectos relevantes de los roles respectivos, así como la gestión de ciertas funciones relacionadas con el área académica.

A continuación, se presenta el backlog de producto, que contiene los requisitos y características principales del sistema.

Tabla 3.1

Fases Tabla de requerimientos

Id	Descripción	Prioridad
1	Diseño de la base de datos relacional	Alta
2	Implementación de la Base de Datos en PostgreSQL	Alta
3	Creación del proyecto Frontend en Framework Angular y Git.	Alta
4	Creación del proyecto Backend en Framework Flask y Git.	Alta
5	Creación de funciones y procedimientos almacenados en la Base de Datos.	Media
6	Elaboración de diferentes API en Backend.	Alta

7	Modelación de estructura del sistema en Backend y Frontend.	Alta
8	Gestión de creación de roles de usuario.	Alta
9	Gestión de registro de estudiantes, docentes, administrativos, invitados y otros.	Media
10	Gestión de registro de inscripciones.	Alta
11	Gestión de registro de materias, cursos.	Alta
12	Gestión de registro de pagos.	Alta
13	Gestión de notificaciones de pagos pendientes.	Alta
14	Gestión de notificaciones de cursar a materias pendientes.	Alta
15	Gestión de registro de notas.	Alta
16	Gestión de registro de asistencia.	Media
17	Implementación de reportes tipo PDF en los módulos necesarios.	Media
18	Implementación de Autenticación segura mediante el uso de JWT (Json Web Token).	Alta
19	Gestión de registro de Roles y Permisos de usuario.	Alta
20	Diseñar e implementar interfaz de usuario.	Alta

Nota. Elaboración Propia

3.2.1 Construcción de tablero

La integración de un tablero en la metodología Scrumban es una de las herramientas clave adoptadas de Kanban. En este enfoque, no hay una definición preestablecida de los campos que deben utilizarse en el tablero, lo que proporciona flexibilidad para adaptarlo a las necesidades del equipo y del proyecto. Teniendo en cuenta esta flexibilidad, se ha diseñado una versión

personalizada del tablero utilizando la herramienta Notion. Aunque este tablero tiene varios modos u opciones de uso, se gestiona y consume la misma información ingresada.

Este tablero consta de diferentes modos de uso: tablero tipo Kanban, tablero tipo Scrumban, tablero tipo listado. Estos modos están divididos en:

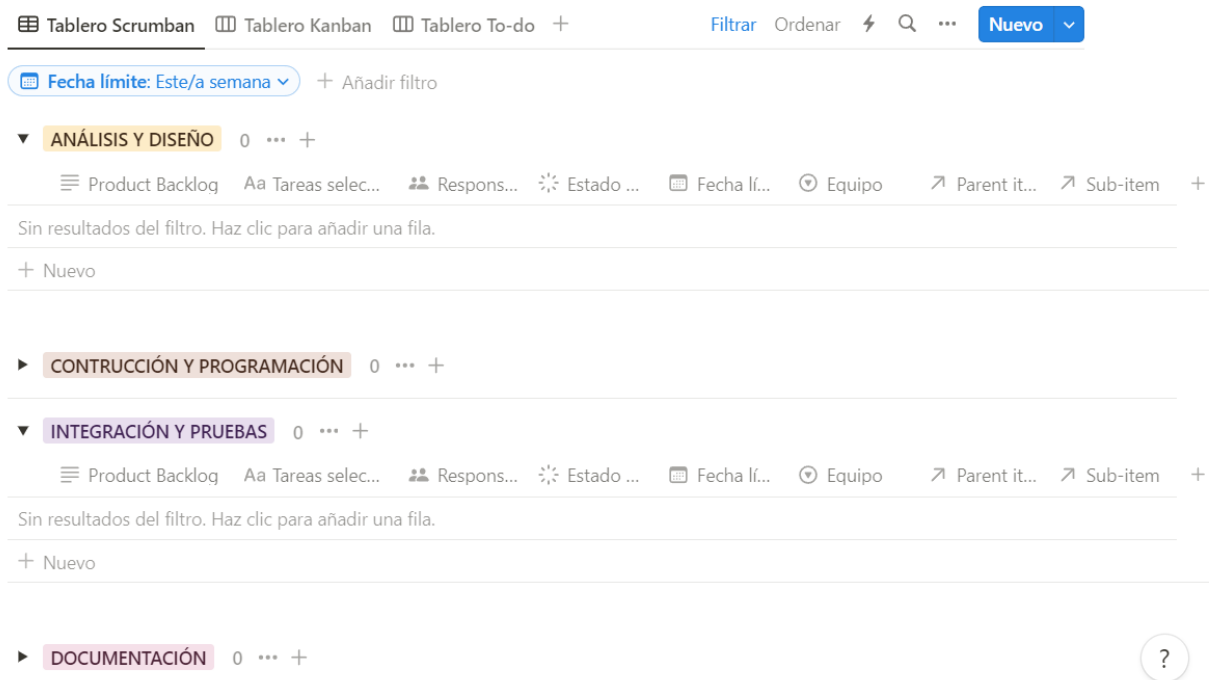
- Análisis y diseño
- Construcción y Programación
- Integración y Pruebas
- Documentación

Figura 3.1

Diseño de tablero Scrumban.

Tablero Scrumban

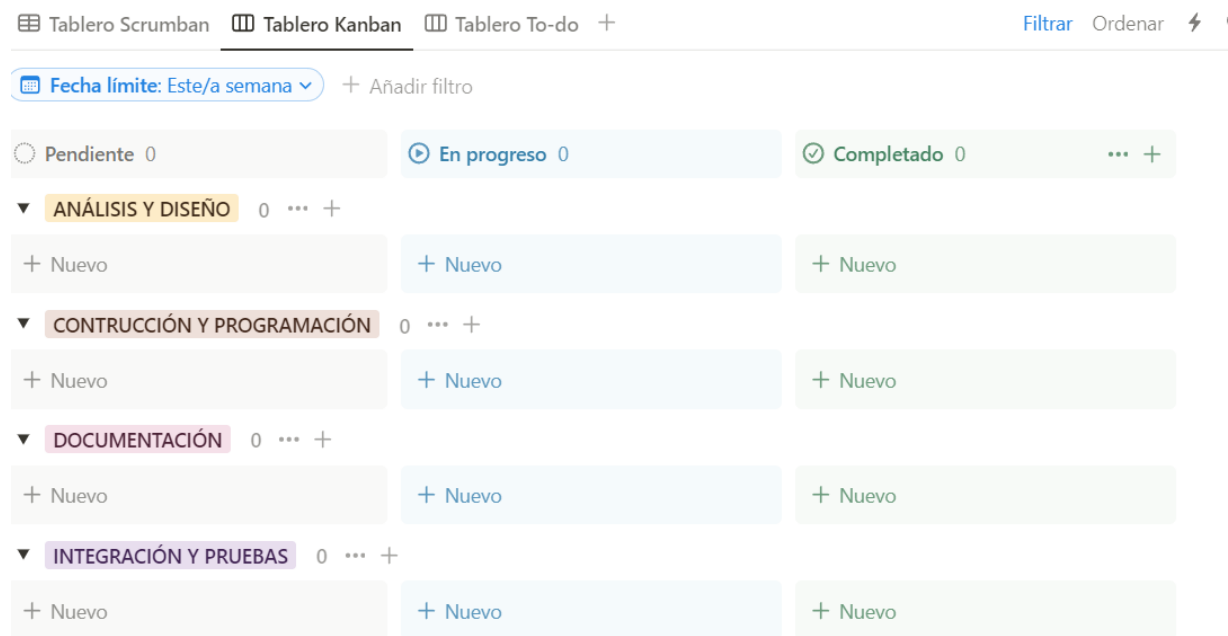
↓ Haz clic en las distintas pestañas de la base de datos para ver otras vistas.



Nota. Elaboración propia

Figura 3.2

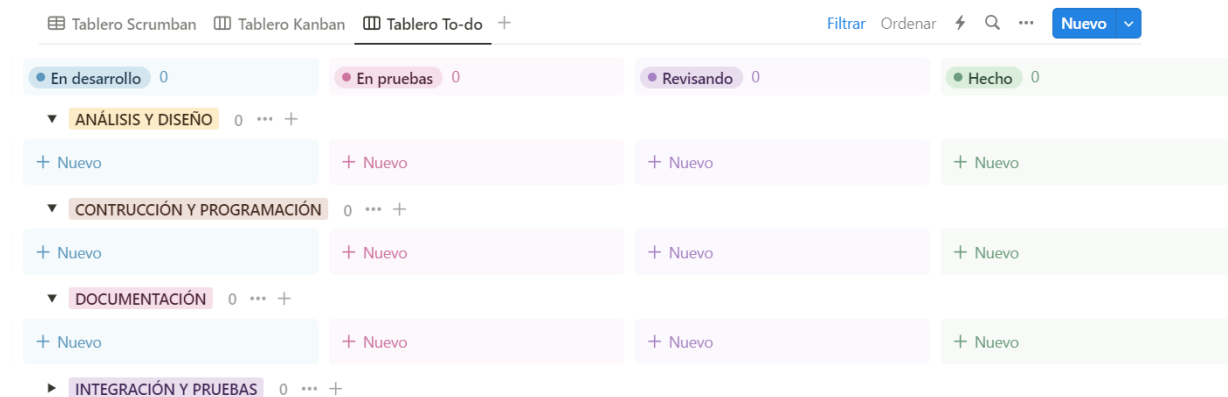
Diseño de tablero Kanban.



Nota. Elaboración propia

Figura 3.3

Diseño de tablero tipo listado.



Nota. Dividida en diferentes secciones: En desarrollo, En pruebas, Revisando y Hecho.

Elaboración Propia.

3.3 Fase Pregame

La fase inicial es fundamental, y ejemplos como el PREGAME en SCRUM o la fase de exploración en Kanban resaltan su importancia. En SCRUM, la fase inicial se centra en el PREGAME, que implica la construcción del Product-Backlog. En cambio, en la metodología Kanban, su fase inicial, la fase de exploración, se dedica a identificar a los usuarios implicados en el desarrollo del producto, en este caso, el sistema. Además, durante esta fase se abordan los requerimientos, tareas y la planificación del producto.

3.2.1 Conceptos y exploración

Los conceptos y exploración es la fase inicial reviste una importancia fundamental en la que implica la construcción del Product-Backlog. Durante esta etapa, se llevaron a cabo tres tareas de manera gradual para la obtención de los requisitos, los cuales se detallan en la siguiente tabla.

Tabla 3.2

Tareas realizadas para la obtención de requisitos

Tarea	Descripción
Entrevistas personales	Se realizan entrevistas con las personas involucradas en la organización, como estudiantes, docentes y administrativos.
Observación	Se observan las complicaciones y dificultades con el acceso y administración de su información entre los involucrados de la organización.
Documentación	Se revisa la documentación proporcionada de la organización.

Nota. Elaboración Propia

3.2.2 Roles Scrumban

En la siguiente tabla, se muestra las fases de trabajo en el tablero Kanban, el límite WIP y los responsables que componen el equipo.

Tabla 3.3

Fases Tabla de requerimientos

Fase	WIP (Work In Progress)	Responsable
Pedido	-	Flow Manager: Director IBCI
Análisis	4	Analista: Ivan Josue Quenta Vargas
Diseño	3	Diseñador: Ivan Josue Quenta Vargas
Implementador	3	Desarrollador: Ivan Josue Quenta Vargas
Producción	-	Flow Manager: Ivan Josue Quenta Vargas

Nota. Elaboración Propia

3.4 Producción

3.4.1 Primera iteración

A continuación, se detallará las tareas elegidas según prioridad.

Tabla 3.4

Fases Tabla de requerimientos

Iteración: 1		
Duración 2 semanas		
Nro	Descripción de la tarea	Tipo
1	Organización de la iteración.	Planificación

2	Modelado de requerimientos mediante casos de uso.	Desarrollo
3	Diseño del diagrama entidad relación para la base de datos.	Desarrollo
4	Modelado conceptual mediante diagrama de clases	Desarrollo
5	Diseño del modela de navegaciones	Desarrollo
6	Diseño del modelo de presentación	Desarrollo
7	Desarrollo de la interfaz gráfica para interfaz principal del sistema (Dashboard)	Desarrollo
8	Codificación de la base de datos en PostgreSQL	Alta

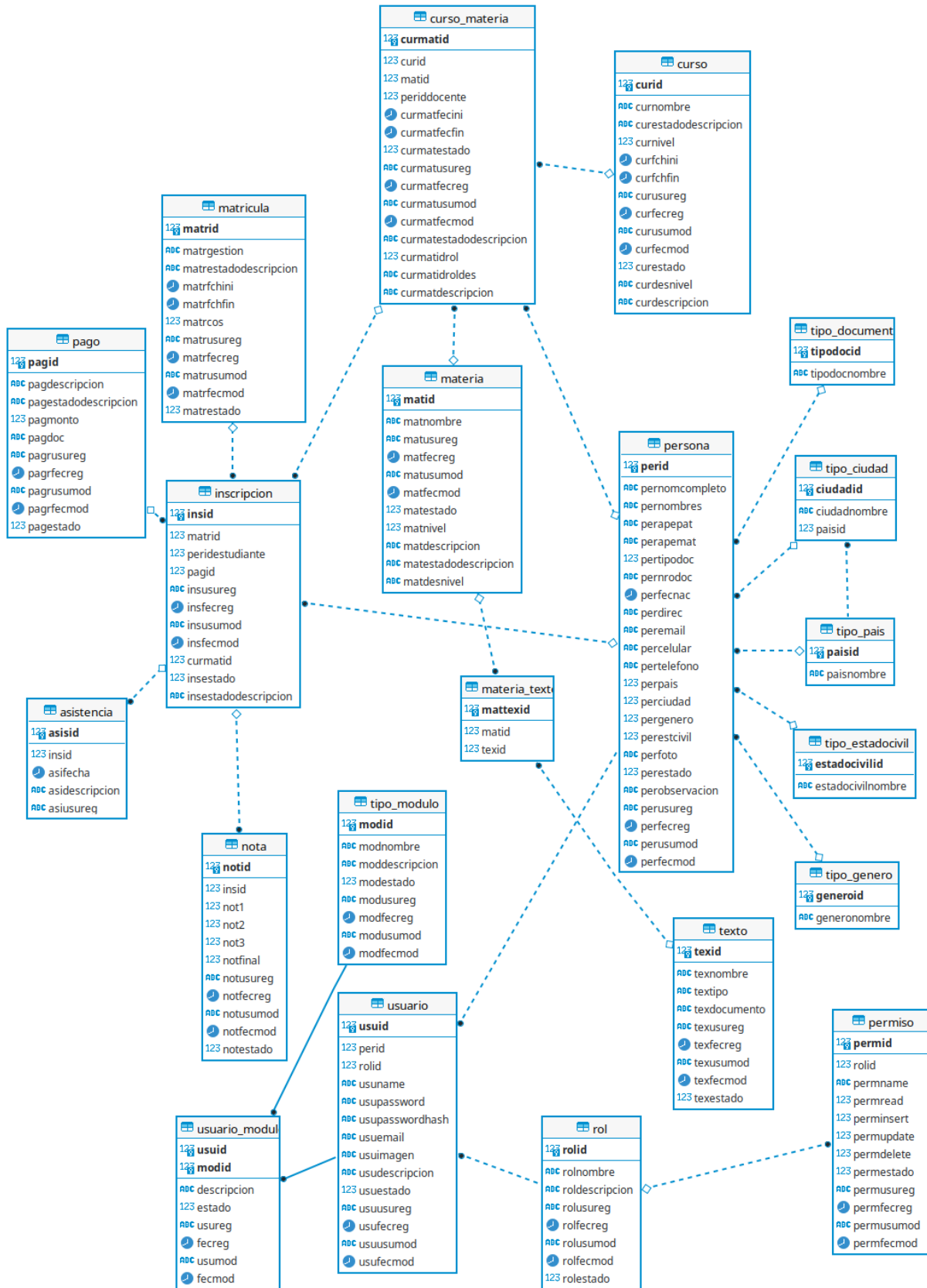
Nota. Pila de productos primera iteración. Elaboración Propia.

3.4.1.1 Modelado de datos

Para el desarrollo del sistema se emplea una base de datos relacional codificada en PostgreSQL cuyo modelo relacional se presenta a continuación.

Figura 3.4

Modelo Relacional



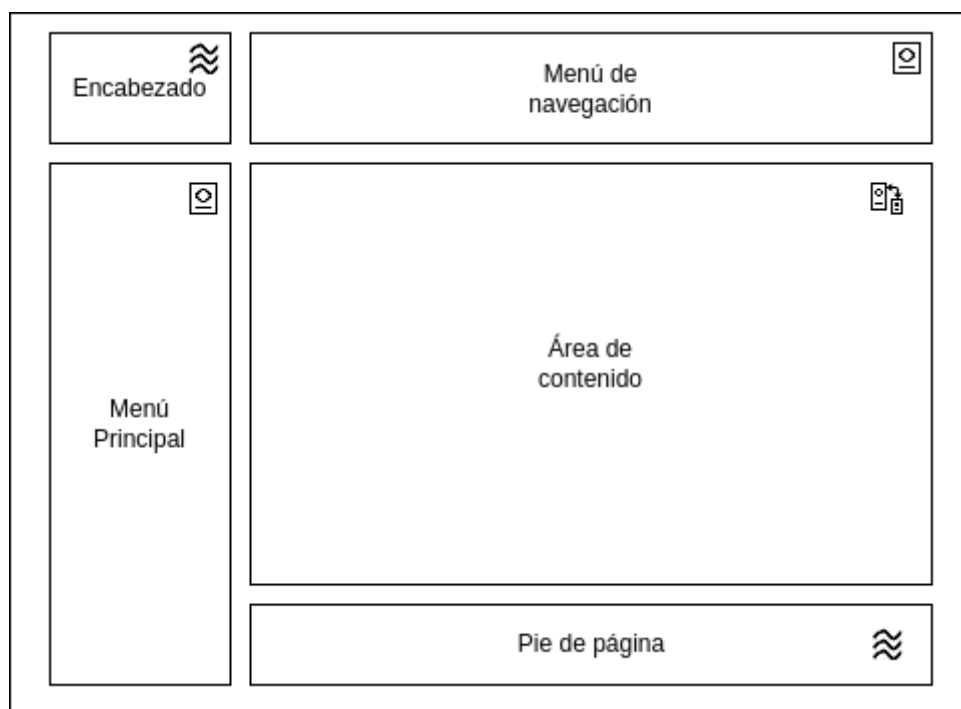
Nota. Diseño del modelo Relacional de la base de datos. Elaboración Propia.

3.4.1.2 Modelado conceptual

A continuación, se muestra el modelo general de presentación de la página principal.

Figura 3.5

Modelo de presentación



Nota. Elaboración Propia.

Con el fin de la primera iteración se ha realizado el análisis retrospectivo correspondiente con el fin de mejorar el proceso en la siguiente iteración, las funcionalidades correspondientes al incremento de la iteración son: la interfaz gráfica, el modelado del sistema y una base de datos funcional.

3.4.2 Segunda iteración

A continuación, se detalla las tareas seleccionadas según la prioridad en este sprint y considerando el análisis retrospectivo en el anterior sprint. Para esta iteración se ha tomado en cuenta todos los módulos encargados de por la sección administrativa, información relacionada con los estudiantes, los cursos, niveles y materias de que registran en la institución.

Tabla 3.5

Fases Tabla de requerimientos

Sprint: 2		
Duración 2 semanas		
Nro	Descripción de la tarea	Tipo
1	Planificación de la iteración	Planificación
2	Complementación de los casos de uso	Desarrollo
3	Complementación de diagrama de clases	Desarrollo
4	Implementación de funciones y procedimientos almacenadas para la manipulación de la base de datos	Desarrollo
5	Codificación de API REST de la sección académica.	Desarrollo
6	Diseño del modelo de presentación	Desarrollo
7	Desarrollo de la interfaz gráfica para interfaz de rol administrativo	Desarrollo

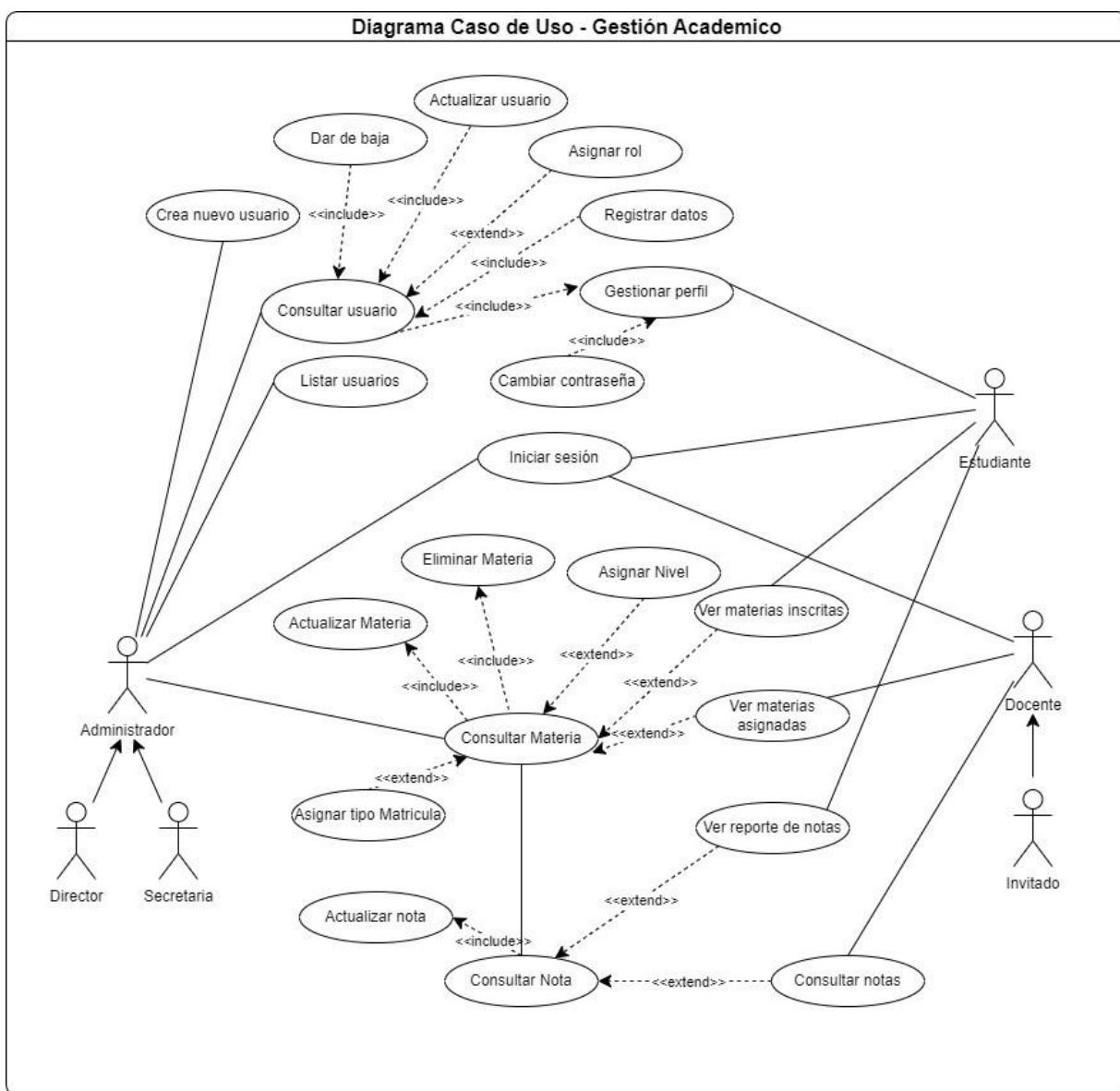
Nota. Pila de productos segunda iteración. Elaboración Propia.

3.4.2.1 Casos de uso – Gestión académica

A continuación, se muestra el caso de uso gestión académica en contiene las operaciones como ALTA, BAJA y modificaciones del registro de los alumnos.

Figura 3.6

Diagrama Casos de Uso para Gestión Académica.



Nota. Elaboración propia

3.4.2.2 Descripción caso de uso – Gestión Académica

A continuación, se detallamos los diseños para gestión académica, el cual contempla el manejo de la información de los usuarios, materias, matrícula, cursos y las inscripciones de los estudiantes.

A continuación, se muestra el caso de uso detallado para registrar al estudiante.

Tabla 3.6

Descripción de case de uso “Administrar registro de estudiantes”

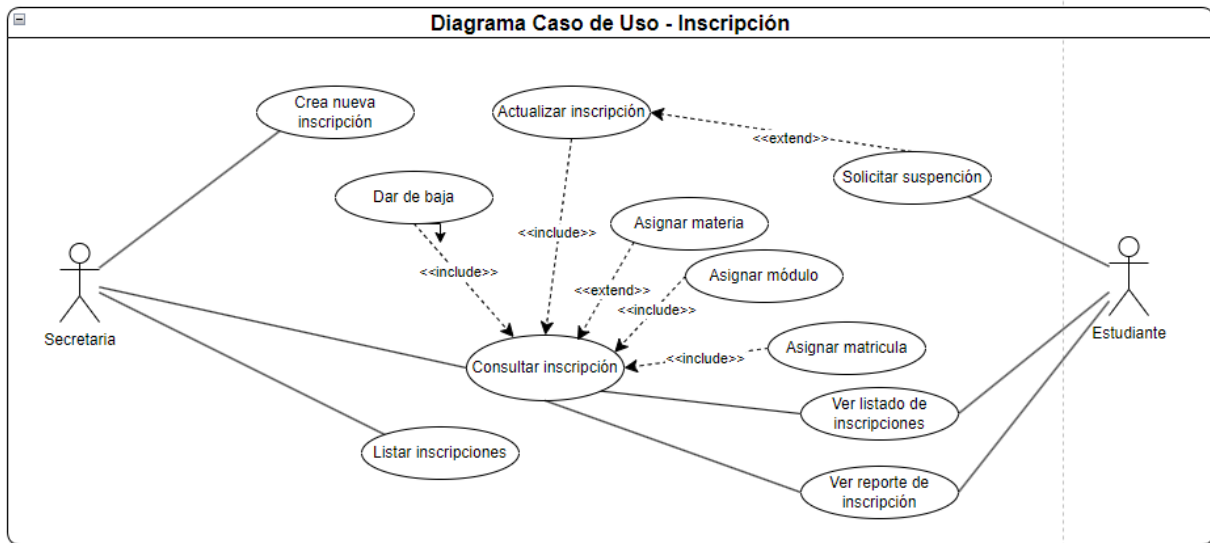
Caso de Uso: Administrar registro de estudiantes	
Actor	Planificación de la iteración
Descripción	Administrar el ABM (Alta, Baja y Modificación)
Precondición	El administrativo debe introducir los datos del estudiante,
Prioridad	Primaria

Nota. Elaboración Propia.

A continuación, se muestra el caso de uso de la inscripción de estudiantes.

Figura 3.7

Diagrama Casos de Uso Inscripción.



Nota. Elaboración propia

Tabla 3.7

Descripción de case de uso “Administrar Inscripciones”

Caso de Uso: Administrar inscripciones	
Actor	Administrativo, Estudiante
Descripción	Administrar el ABM (Alta, Baja y Modificación) y listado de inscripciones de estudiantes a los diferentes cursos, niveles
Precondición	El estudiante debe estar registro previamente en la base de datos.
Prioridad	Primaria

Nota. Elaboración Propia.

Tabla 3.8

Descripción de case de uso “Registrar la suspensión o el retorno de cursos”

Caso de Uso: Administrar inscripciones	
Actor	Administrativo, Estudiante
Descripción	Cambia el estado de una inscripción vigente, pendiente, abandono o finalizado. Este estado es de gran importancia, ya que dependerá el manejo de la información en otras operaciones como el control de pago de cursos.
Precondición	El administrativo debe registra en detalle las razones de la suspensión o el retorno, previamente verificados solicitando por la institución.
Prioridad	Primaria

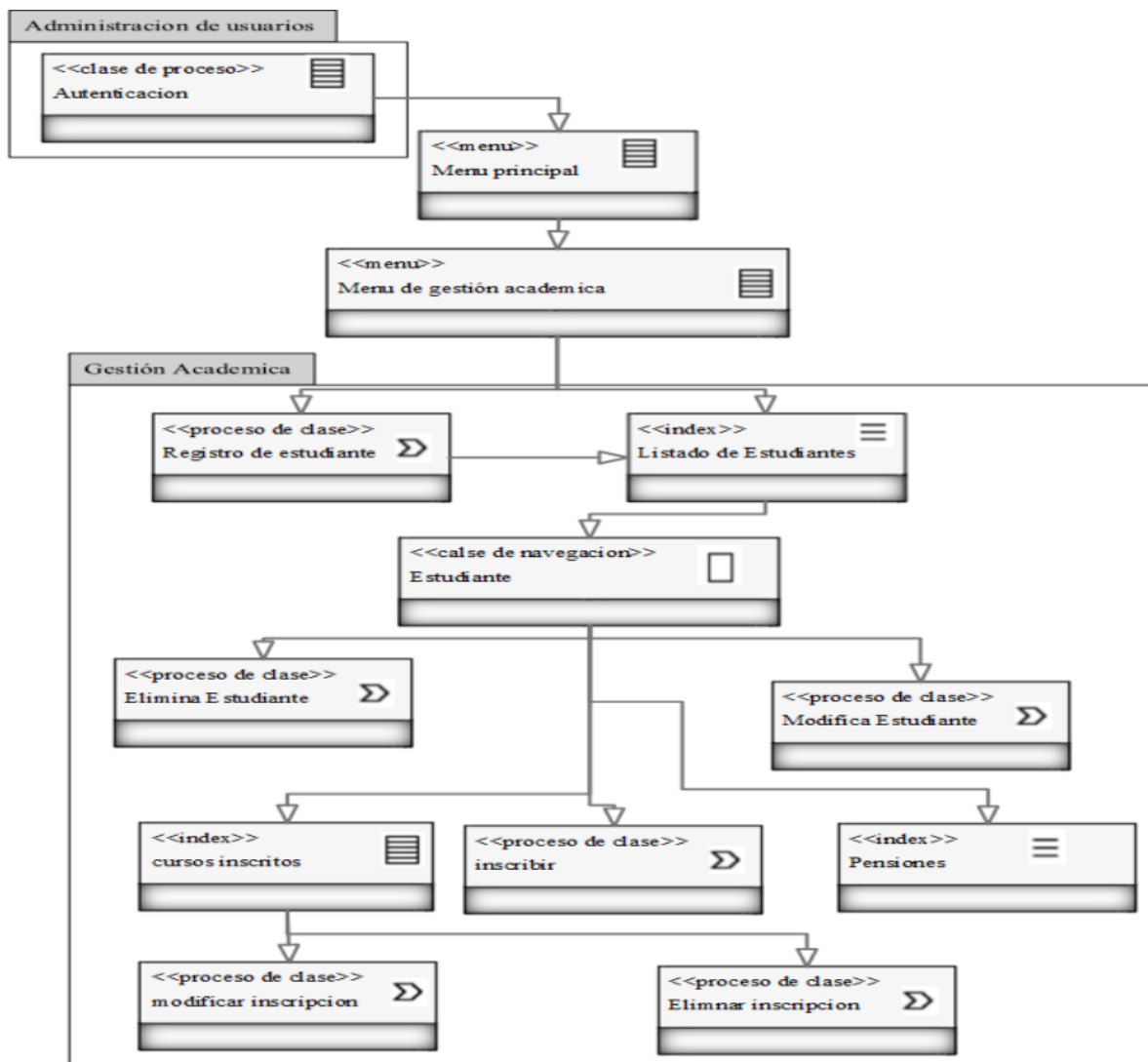
Nota. Elaboración Propia.

3.4.2.3 Modelado de navegación – Gestión Académica

A continuación, se muestra el modelo navegacional para la gestión académica.

Figura 3.8

Modelo de navegación Gestión Académica



Nota. Elaboración Propia.

3.4.2.4 Modela de presentación – Gestión Académica

A continuación, se muestra el modelo de representación correspondiente a la gestión académica.

Figura 3.9

Modelo de presentación Gestión Académica

(Imagen)

Nota. Elaboración Propia.

Las funcionalidades correspondientes al incremento de la interacción con los módulos encargados de la administración de la información relacionada con los estudiantes, los módulos encargados de la administración de la información.

3.4.3 Tercer iteración

A continuación, se detalla las tareas seleccionadas según la prioridad en este sprint y considerando el análisis retrospectivo en el anterior sprint.

Tabla 3.9

Fases Tabla de requerimientos

Sprint: 3		
Duración 2 semanas		
Nro	Descripción de la tarea	Tipo
1	Planificación de la iteración	Planificación
2	Complementación de los casos de uso	Desarrollo
3	Complementación de diagrama de clases	Desarrollo
4	Implementación de funciones y procedimientos almacenadas para pago de inscripción de la base de datos	Desarrollo
5	Codificación de API REST de la sección académica.	Desarrollo
6	Diseño del modelo de presentación de pagos.	Desarrollo
7	Codificación de los métodos para el manejo del módulo encargado de la gestión de pagos.	Desarrollo
8	Desarrollo de la interfaz gráfica para interfaz de rol administrativo	Desarrollo

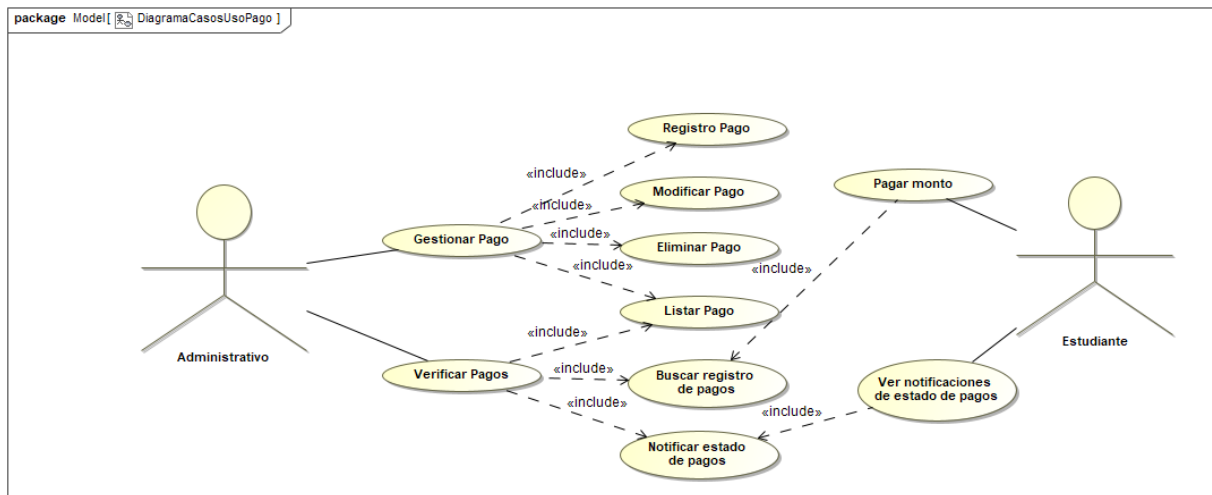
Nota. Pila de productos segunda iteración. Elaboración Propia.

3.4.3.1 Casos de uso – Gestión de pagos

A continuación, se muestra el diagrama de caso de uso general denominado gestión de pago de materias, encargado del control, registro, verificación del pago, en cuento a una inscripción.

Figura 3.9

Diagrama Casos de Uso para Gestión de Pagos.



Nota. Elaboración propia

3.4.3.2 Descripción caso de uso – Gestión de pagos

Los casos de uso que se detallan a continuación están diseñados para la administración del pago de inscripciones, matriculas de los estudiantes el cual contempla el registro de pagos de pensiones y verificación y estado de los pagos pendientes.

A continuación, se muestra la descripción del caso de uso “Gestión de pagos”.

Tabla 3.10

Descripción de case de uso “Verificar pagos”

Caso de Uso: Verificar Pagos	
Actor	Administrativo, Estudiante
Descripción	Le permite visualizar los pagos de materias por estudiante, se podrá observar el estado de pago, como vencidos, pagado o pendiente.
Precondición	El administrativo debe registrar los pagos de materias y matricula.
Prioridad	Primaria

Nota. Elaboración Propia.

Tabla 3.11

Descripción de case de uso “Registrar Pagos”

Caso de Uso: Registrar Pagos	
Actor	Administrativo
Descripción	El registro de pagos de materia o matricula se realizará por estudiante.
Precondición	El estudiante debe pagar el monto correspondiente a la materia o matricula.
Prioridad	Primaria

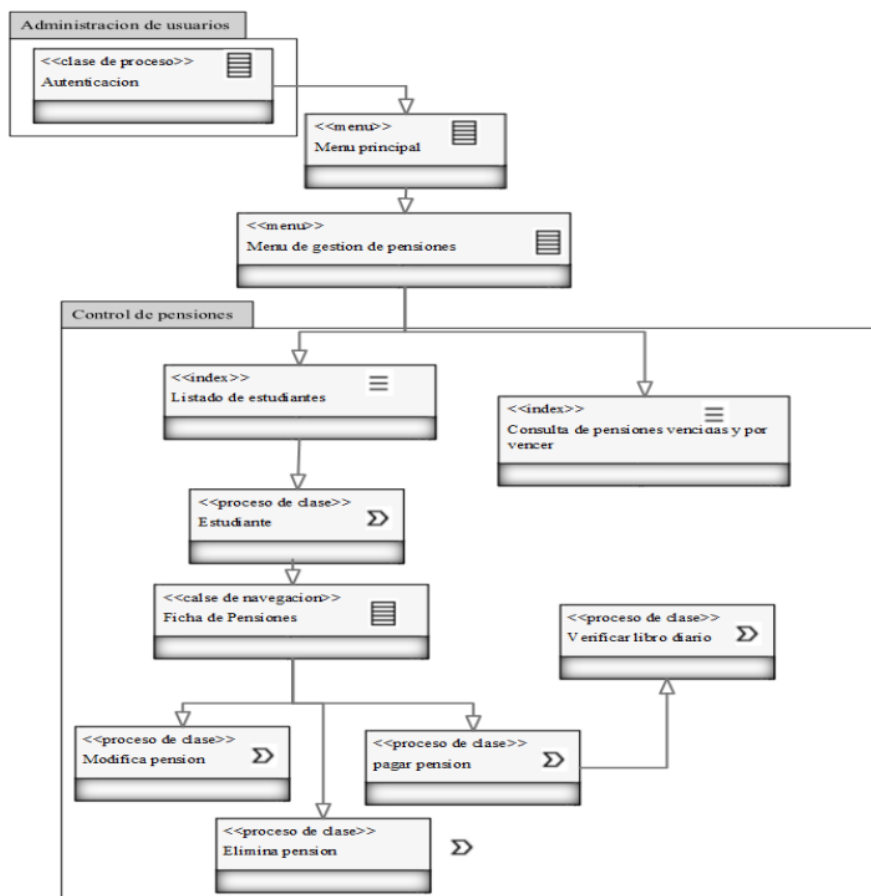
Nota. Elaboración Propia.

3.4.3.2 Modelado de navegación – Gestión de pagos

A continuación, se muestra el modelo navegacional para la gestión de pagos de materias y matrícula.

Figura 3.9

Modelo de navegación – Gestión de Pagos.



Nota. Elaboración propia.

Al final de la tercera iteración las funcionalidades correspondientes al incremento de la iteración son: Los módulos encargados de la gestión de pagos de cada estudiante y los módulos encargados de la verificación de pagos pendientes y por vencer.

3.4.4 Cuarta iteración

A continuación, se detalla las tareas seleccionadas según la prioridad en este sprint y considerando el análisis retrospectivo en el anterior sprint.

Tabla 3.12

Fases Tabla de requerimientos

Sprint: 4		
Duración 2 semanas		
Nro	Descripción de la tarea	Tipo
1	Planificación de la iteración	Planificación
2	Complementación de los casos de uso módulo registro de roles y permisos	Desarrollo
3	Implementación de funciones y procedimientos almacenadas para administración de roles y permisos de usuario	Desarrollo
4	Codificación de las Apis necesarias para la administración de registro de roles y permisos.	Desarrollo
5	Diseño del modelo de presentación de roles y permisos.	Desarrollo
6	Desarrollo de la interfaz gráfica para gestión de roles y permisos,	Desarrollo

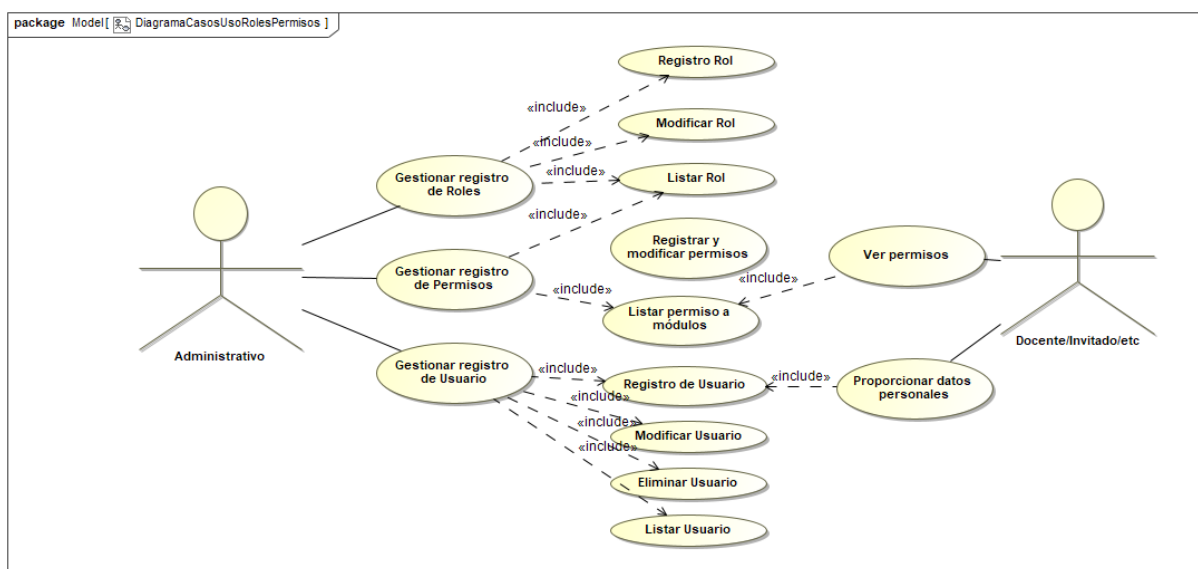
Nota. Pila de productos cuarta iteración. Elaboración Propia.

3.4.4.1 Casos de uso – Gestión de roles y permisos

A continuación, se muestra el diagrama de caso de uso general denominado gestión de registros, listado de roles y permisos.

Figura 3.10

Diagrama Casos de Uso para Gestión de Roles y Permisos.



Nota. Elaboración propia

3.4.4.2 Descripción de casos de uso – Gestión de roles y permisos

Los casos de uso que se detallan a continuación están diseñados para la gestión de registro de roles y permisos, el cual necesita del registro de usuarios adicionalmente. Esto se realizará en caso de cualquier tipo de rol, por ejemplo, docente, estudiante, administrativo, invitado, etc.

A continuación, se muestra la descripción del caso de uso “Gestión de Roles y permisos”.

Tabla 3.10

Descripción de case de uso “Gestión de registro de Roles y Permisos”

Caso de Uso: Gestión de Roles y Permisos

Actor	Administrativo, Estudiante, Docente, Invitado
Descripción	El administrativo puede agregar, modificar, eliminar a los usuarios de diferentes roles de la base de datos.
Precondición	El administrativo debe registrar los roles y permisos del usuario.
Prioridad	Primaria

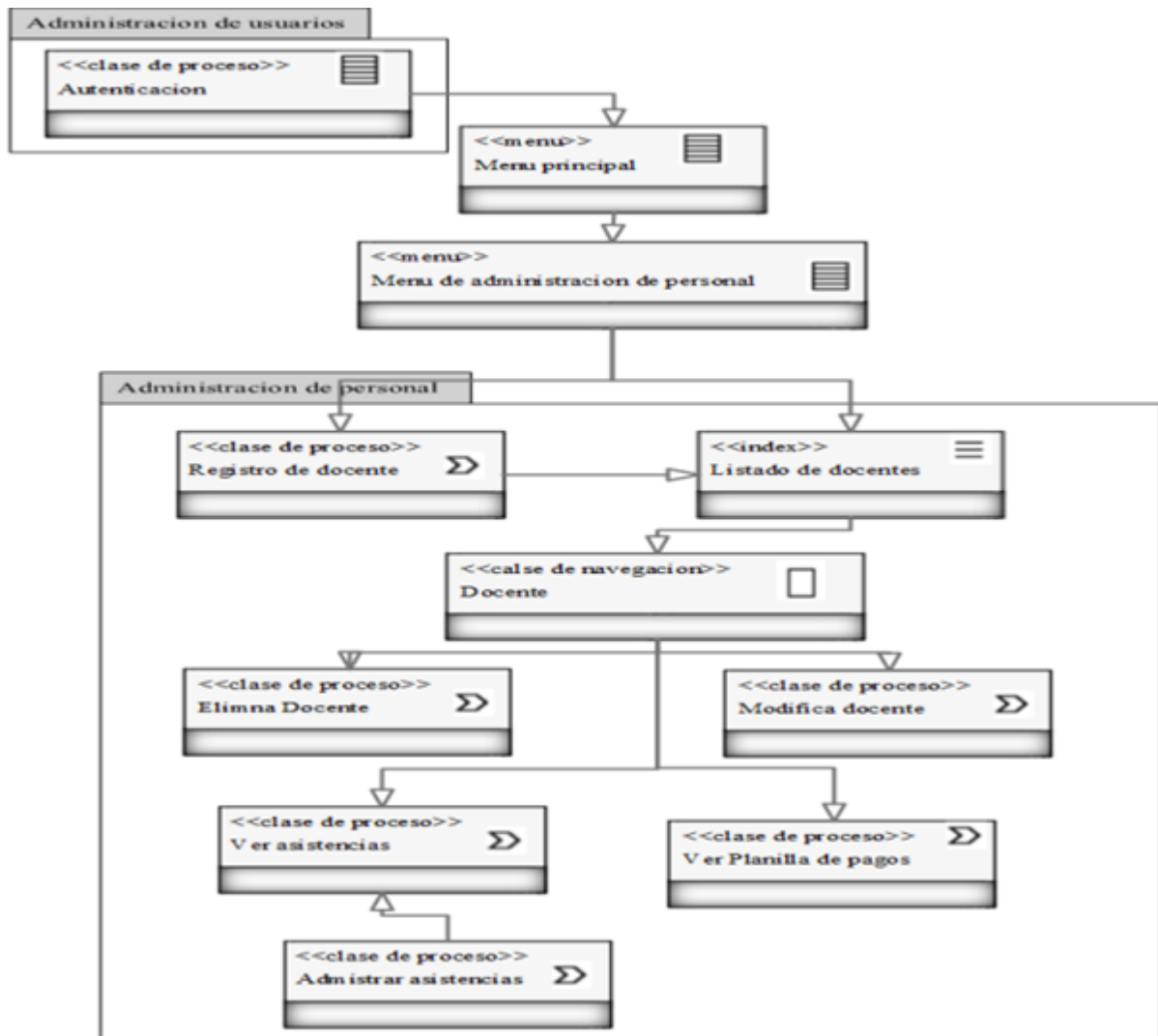
Nota. Elaboración Propia.

3.4.4.4 Modelo de navegación – Gestión de roles y permisos

A continuación, se muestra el modelo navegacional para la gestión de pagos de materias y matrícula.

Figura 3.11

Modelo de navegación – Gestión de Roles y Permisos.



Nota. Elaboración propia

3.5 Entrega del producto

3.5.1 Diseño de interfaces

El diseño de interfaz es una de las partes muy importantes del desarrollo del sistema, pues es la primera vista que el usuario tendrá del sistema, además que aquí es donde se observa la diferentes complicaciones y errores de diseño.

Figura 3.12

Interfaz de Login.

(Imagen)

Nota. Elaboración propia

Figura 3.13

Interfaz de Usuarios.

(Imagen)

Nota. Elaboración propia

3.6 Métricas

En la actualidad la calidad y los criterios de calidad de un software es indispensables, entre los más destacados tenemos la ISO 9126, el cual se aplicará en el presente proyecto.

3.6.1 Funcionalidad

En característica de la funcionalidad se utiliza la técnica de pruebas exploratorias, con el propósito de asegurar la funcionalidad y la estabilidad del software.

1. Prueba exploratoria por iteraciones:

Los casos de prueba fueron diseñado acorde a estándares establecidos en la misma técnica de pruebas. Se detallan en las siguientes tablas: 1, 2 y 3, estas correspondientes a cada iteración.

(Tabla 1)

(Tabla 2)

(Tabla 3)

3.6.2 Usabilidad

La usabilidad o facilidad de uso (FU), se calcula de la siguiente manera.

(formula)

$$FU = \frac{[(\frac{\sum x_i}{n}) * 100]}{n}$$

Se calcula (xi) y E(xi), utilizando la escala de evaluación, según de las tablas anteriores y basándonos en los datos de la tabla (tabla 4)

(Tabla 4)

Calculando FU:

(formular)

$$FU = \frac{[(\frac{90}{10}) * 100]}{10}$$

$$FU = 90 \Rightarrow 90\%$$

Por lo tanto la facilidad de uso es del 90 por ciento.

CAPITULO 4 CONCLUSIONES Y RECOMENDACIONES

Al finalizar el proyecto se tiene las siguientes conclusiones y recomendaciones.

4.1 Conclusiones

Con la realización del presente proyecto se concluye que:

- Se disminuyo el uso de papel en el papeleo.
- Se logró implementar un módulo para facilitar la gestión del área académica.
- Se logró implementar un módulo para facilitar la gestión de usuarios del sistema.
- Se logró centralizar la información de los registros competentes.
- Se pudo brindar una solución efectiva a la gestión de pensiones permitiendo a los administrativos controlar el pago de materias.
- A partir de la implementación del sistema se consiguió un proceso automatizado de registro y procesamiento de las transacciones económicas.

4.2 Recomendaciones

Finalizando el presente proyecto se hacen las siguientes recomendaciones:

- Incorporar normas y políticas dentro del instituto para el uso del sistema.
- Para el uso del sistema es recomendable cambiar las contraseñas de los administrativos mensualmente, por razones de seguridad del sistema.
- Realizar copias de seguridad de la base de datos si es posible semanalmente, ya que la información que contiene es muy para el funcionamiento del sistema.

1. CRONOGRAMA

ACTIVIDAD	CRONOGRAMA DE AVANCE DE PROYECTO DE GRADO																							
	JULIO				AGOSTO				SEPTIEMBRE				OCTUBRE				NOVIEMBRE				DICIEMBRE			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Establecer los requerimientos y requisitos en conformidad con el Instituto IBCI. Redacción de Marco Teórico.																								
Diseñar una base de datos, para el registro de estudiantes, docentes, materias, notas, pagos, niveles. Redacción de Marco Aplicativo.																								
Diseñar una interfaz de usuario sencilla de usar y funcional de acuerdo a las especificaciones y requerimiento del instituto IBCI.																								
Diseñar el módulo de seguridad, mediante asignación de roles y niveles de acceso, de esta forma poder resguardar la información de los usuarios.																								
Implementar un módulo de administración de notas.																								
Implementar un módulo de monitoreo del avance de los estudiantes.																								

Implementar un módulo de control y seguimiento de pagos de inscripción.																							
Redacción de capítulo calidad y seguridad.																							
Implementar un módulo para el despliegue de reportes de impresión y control.																							
Redacción del Capítulo Análisis Costo Beneficio																							
Capacitación del funcionamiento del sistema al personal de la institución.																							
Redacción del Capítulo Conclusiones y Recomendaciones																							

2. BIBLIOGRAFÍA

Unesco. (2016). *Las tecnologías de la información y la comunicación (TIC) en la educación*.

Obtenido de https://unesdoc.unesco.org/ark:/48223/pf0000129533_spa

Echeverry Tobón, L. M., & Delgado Carmona, L. E. (2007). *Caso práctico de la metodología ágil XP al desarrollo de software*.

SALAZAR, M. N. (2015). *Sistema de Seguimiento Académico y Control Disciplinario*. La Paz, Bolivia: Facultad de Ciencias Puras y Naturales.

CHURA, N. C. (2014). *Sistema de Seguimiento Académico y gestión administrativa. Caso: Unidad de Postgrado en informática - UMSA*. La Paz, Bolivia: Facultad de Ciencias Puras y Naturales.

CARMEN, T. A. (2021). *Análisis de un Sistema Web de Seguimiento Académico para la I.E.P. "San Marcos"*. Piura, Perú. Escuela Profesional de ingeniería de sistemas.

NOVOA, E. M. & RODRÍGUEZ, J. C. (2015). *Diseño de un Sistema Web para el seguimiento y evaluación de los alumnos con carta de permanencia en la facultad de ciencias contables, económicas y financieras de la Universidad de San Martín de Porres*. Lima, Perú. Universidad de San Martín de Porres.

ARTECH, C. (2012). *Visión General de Genexus*. Montevideo, Uruguay. Recuperado de <https://www.genexus.com/es/noticias/leer-noticia/wp-vision-general>

Garcés, M. E. (2021). *Universidad Tecnológica de Bolívar*. Obtenido de <https://www.utb.edu.co/blog/las-nuevas-tecnologias-de-informacion-y-comunicacion-tic-aplicadas-en-contextos-sociales-y-educativos/>

IBCI. (2023). *Instituto Bíblico de Capacitación Internacional*.

Sainz, L. A. (2023). Instituto Bíblico de Capacitación Internacional. (U. I. Vargas, Entrevistador)

Gamboa, J. C. (2014). *Aumento de la productividad en la gestión de proyectos, utilizando una metodología ágil aplicada en una fábrica de software en la ciudad de Guayaquil*. Revista Tecnológica-ESPOL, 27(2).

Taberner, B. (2015). *Informatización de una Pyme aplicando metodología ágil: un caso real. (Proyecto fin de grado)*. Universidad Politécnica de Valencia, Valencia, España.

Thinking, D. (2020). *Que es la metodología "SCRUMBAN" Temporada 12 Ep 1*. Obtenido de https://www.youtube.com/watch?v=_jGJT6sMkCQ

Guzmán, D. I., Castañeda Islas, U., & Pedroza Méndez, B. (2014). *Metodología ágil scrumban en el proceso de desarrollo y mantenimiento de software de la norma moprosoft*. pág. 103.

ANEXOS

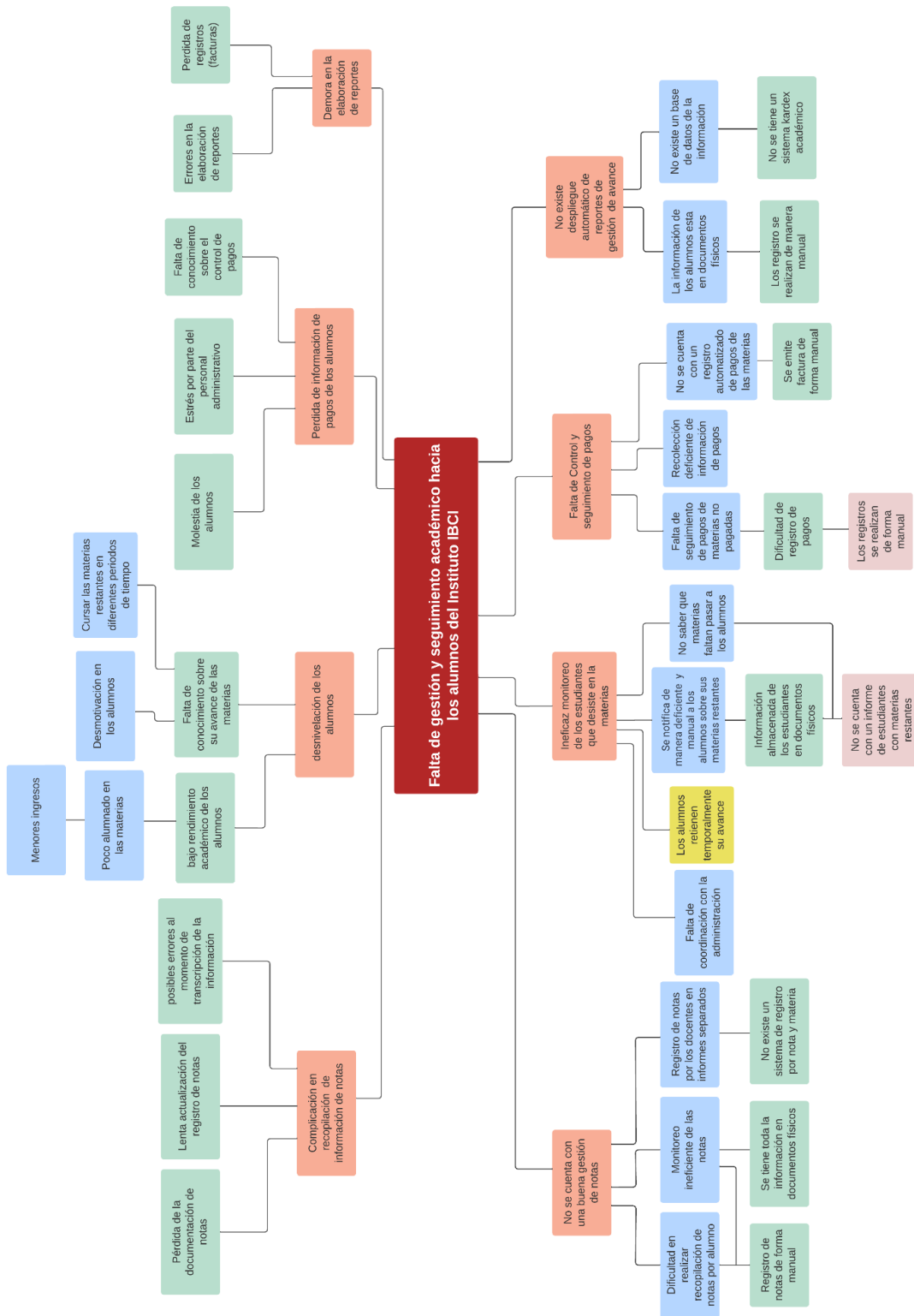
ANEXO ANALISIS DE SITUACIÓN

Resumen Narrativo	Indicadores	Medios de Verificación	Supuestos
FIN Contribuir a mejorar la gestión y el seguimiento de los alumnos en el resguardo de su información, además de una rápida actualización de estas y menores errores de transcripción. Aumentando el rendimiento académico y a su vez el monitoreo de su avance. También se visualizará su información mediante reportes e informes de control para satisfacer las necesidades del personal administrativo, docente y estudiantes del IBCI.	Disminuir en la cantidad de estudiantes que abandonan en los diferentes niveles del instituto bíblico de IBCI captación internacional al cabo de 1 año.	Formulario de conformidad, Análisis de crecimiento por parte de la administración.	El instituto bíblico de IBCI capacitación internacional, está en total conformidad de mejorar la gestión y seguimiento de los estudiantes dentro de la institución.
PROPÓSITO Desarrollar un sistema de gestión y seguimiento académico para los estudiantes del instituto bíblico de capacitación internacional.	Desarrollar un sistema de gestión y seguimiento académico, que contará con 5 módulos según la ISO 9126, hasta 31 de noviembre de 2023.	Uso de software por parte del personal administrativo, docente y estudiantes del IBCI, además de la defensa pública del proyecto de grado realizado.	Obtener el respaldo de la dirección del IBCI, por lo cual se proporcionará la información relacionada a los procesos de inscripción, administración, calificación y gestión de pagos.
PRODUCTOS C-1: Requerimientos y requisitos establecidos de acuerdo con el IBCI. C-2: Una base de datos diseñada, de acuerdo a las especificaciones del IBCI, para el registro de información. C-3: Interfaz de usuario funcional y sencilla de usar de acuerdo a las especificaciones y requerimientos del IBCI. C-4: Módulo de seguridad diseñado, mediante asignación de roles de acceso, de esta forma poder resguardar la información de los usuarios.	1. Se mantendrán reuniones con la dirección, el personal docente y estudiantes del IBCI para contemplar un análisis de los requerimientos. desde 01/07/23 hasta 21/07/23. 2. Se realizará, al menos 10 tablas, en un diagrama entidad relación para implementarlo en la base de datos, desde 09/07/23 hasta 05/08/23. 3. Se creará la interfaz de usuario según los requerimientos, desde 05/08/23 hasta 22/08/23.	1. Entrevistas con los usuarios donde se describirán los requerimientos y propuestas descritas por el presente trabajo. 2. Inicio del desarrollo del sistema de información. 3. Base de datos implementada para el funcionamiento del software.	El IBCI está de acuerdo en facilitar la información de los informes de registros de la información de los estudiantes y mostrar cómo se realizan los procesos de inscripción, pagos y registro de notas. Se tiene disponibilidad de tiempo y compromiso por parte del personal del IBCI y estudiantes para realizar las entrevistas y encuestas.

<p>C-5: Módulo de administración de notas, que permita el registro y seguimiento de notas de los estudiantes.</p> <p>C-6: Módulo de monitoreo de estudiantes, que permita monitoreo del avance y envío de mensajes de notificaciones a los estudiantes.</p> <p>C-7: Módulo de control y seguimiento de pagos, para los registros de inscripciones y pagos de materias.</p> <p>C-8: Módulo de reportes, para el despliegue de reportes de impresión y control de acuerdo a los requerimientos del IBCI.</p> <p>C-9: Capacitación del funcionamiento del sistema al personal de la institución.</p>	<p>4. Se realizarán niveles de y asignan roles, desde 25/08/23 hasta 16/09/23.</p> <p>5. Se realizará el módulo de notas con sus interfaces, desde 17/09/23 hasta 14/10/23.</p> <p>6. Se realizará el módulo de monitoreo de estudiantes con 3 interfaces de usuario respectivas, desde 17 de septiembre hasta el 14 de octubre de 2023.</p> <p>7. Se realizará el módulo de administración de pagos con 3 interfaces de usuario respectivas, desde 2 hasta el 30 de octubre de 2023.</p> <p>8. Se realizará reportes por módulo, bajo el control y aprobación del IBCI, desde 15/10/23 hasta 15/11/23.</p>	<p>4. Demostración a la dirección del Instituto IBCI de las diferentes interfaces de usuario de los módulos.</p> <p>5. Pruebas de seguridad realizadas durante la implementación del software.</p> <p>6. Los reportes se desplegarán en PDF.</p>	<p>Se tiene toda la información necesaria para realizar un diseño de base de datos, que cumpla con las necesidades del instituto IBCI.</p> <p>El IBCI brindará las características que se deberá seguir en el diseño de las interfaces.</p> <p>La disponibilidad de tiempo necesario para la capacitación del personal administrativo, docente y estudiantes del IBCI, sobre el funcionamiento del sistema desarrollado.</p>
<p>ACTIVIDADES</p> <p>C-1:</p> <ul style="list-style-type: none"> - Identificar requerimientos y requisitos. - Realizar entrevistas, encuestas y reuniones con el personal de la institución. <p>C-2:</p> <ul style="list-style-type: none"> - Implementar, analizar y ajustar el diagrama E-R. <p>C-3:</p> <ul style="list-style-type: none"> - Diseñar los mockups, con un diseño sencillo y eficiente. - Implementar interfaz de usuarios para cada módulo. <p>C-4:</p> <ul style="list-style-type: none"> - Definir los roles del sistema. - Diseñar las vistas de accesos. <p>C-5:</p> <ul style="list-style-type: none"> - Realizar proceso de registro de notas. - Realizar reporte de notas para todos los usuarios. <p>C-6:</p> <ul style="list-style-type: none"> - Realizar despliegue de lista de estudiantes inscriptos, por materia y por nivel. 	<p>C-1: -</p> <p>Componente 2: \$ 370</p> <p>Componente 3: \$ 450</p> <p>Componente 4: \$ 450</p> <p>Componente 5: \$ 250</p> <p>Componente 6: \$ 470</p> <p>Componente 7: \$ 470</p> <p>Componente 8: \$ 200</p> <p>Componente 9: \$ 310</p> <p>Total \$ 2970</p>	<p>1. Informe de las entrevistas realizadas.</p> <p>2. Implementación de la base de datos acorde a los requerimientos del Instituto IBCI.</p> <p>3. Pruebas de generación y autenticidad del usuario.</p> <p>4. Formulario test de aceptación de usuario por módulo.</p> <p>5. Informe de la captación a los usuarios.</p>	<p>Compromiso de disponibilidad de tiempo con el personal del instituto IBCI.</p> <p>La disponibilidad de tiempo necesaria de la directora del Instituto IBCI.</p> <p>La conformidad con el diseño de la base de datos.</p> <p>La aprobación del diseño de interfaz de usuario por el Instituto IBCI.</p> <p>La aprobación del módulo de administración de notas, monitoreo, pagos, y reportes.</p> <p>El compromiso y disponibilidad de tiempo para el programa de capacitación del sistema desarrollado</p>

<ul style="list-style-type: none"> - Realizar proceso de envío de notificaciones de apertura de materias. - Generar reporte de alumnos sin continuidad en las materias para el usuario administrativo. - Realizar reporte de materias, récord académico y historial académico. <p>C-7:</p> <ul style="list-style-type: none"> - Realizar proceso de búsqueda de información de pagos por materia y nivel. - Realizar lista de materias pendientes a pagar. - Generar reportes/informes de control de los pagos. <p>C-8:</p> <ul style="list-style-type: none"> - Generar reportes de notas, inscripción, materias, pagos, docentes, entre otros. <p>C-9:</p> <ul style="list-style-type: none"> - Realizar videos de capacitación del sistema. - Implementar manual de usuario. - Realizar capacitación al personal docente, administrativo y estudiante. 			
---	--	--	--

ANEXO ÁRBOL DE PROBLEMAS



ANEXO ÁRBOL DE OBJETIVOS

