ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Maestría en Sistemas de Información Gerencial

"DISEÑO DE UNA PLATAFORMA WEB COLABORATIVA CON HERRAMIENTAS TECNOLÓGICAS PARA MEJORAR LOS REQUERIMIENTOS DE USUARIOS PARA PROYECTOS DE DESARROLLO DE SOFTWARE EN UNA EMPRESA DE TELECOMUNICACIONES"

TRABAJO DE TITULACIÓN

PREVIO A LA OBTENCIÓN DEL TÍTULO DE

MAGISTER EN SISTEMAS DE INFORMACIÓN GERENCIAL

PRESENTADO POR:

ING. KEVIN REINALDO BAQUE PUYA

ING. FABRICIO BOLÍVAR BERMEO ROMERO

GUAYAQUIL – ECUADOR

AÑO 2023

AGRADECIMIENTO

Agradezco a Dios por la fortaleza y guía espiritual que me ha brindado. A mis padres, hermanos, esposa e hija, quienes han sido mi mayor motivación. También, agradezco a mi empresa por la oportunidad de crecimiento que ha enriquecido mi trayectoria profesional.

Ing. Kevin Baque Puya

A Dios, por brindarme la oportunidad de cada día ver la luz y poder cumplir mis estudios de cuarto nivel.

A mi familia, por su apoyo incondicional en este camino, con su ayuda me pudieron dar fuerzas en momentos difíciles.

Ing. Fabricio Bermeo Romero

DEDICATORIA

A Dios, por guiar mis pasos y brindarme fuerza inquebrantable. A mis padres, hermanos, fuente inagotable de amor y sabiduría, por su constante apoyo y sacrificio. A mi esposa, compañera de vida y cómplice en cada desafío. A mi hija, luz y alegría que inspira mi esfuerzo diario. Este logro es dedicado a ustedes, quienes han sido mi pilar fundamental en este viaje. Con profundo agradecimiento y amor.

Ing. Kevin Baque Puya

El presente trabajo se lo dedico a mi madre y a mi hermana que son ejemplo de vida, a mi hijo por ser quien día a día me hace sentir una persona importante y ser el motor de nuestras vidas.

Ing. Fabricio Bermeo Romero

TRIBUNAL DE SUSTENTACIÓN

Firmado electrónicamente por:
JUAN CARLOS GARCIA PLUA

M.Sc. Juan Carlos García Plúa

TUTOR

Firmado electrónicamente por:
LENIN EDUARDO FREIRE COBO

M.Sc. Lenin Eduardo Freire Cobo

REVISOR

DECLARACIÓN EXPRESA

“La responsabilidad del contenido de esta Tesis de Grado, me corresponde exclusivamente, y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL.”

(Reglamento de graduación de la ESPOL)

Firmado electrónicamente por:
KEVIN REINALDO
BAQUE PUYA

Ing. Kevin Reinaldo Baque Puya

Firmado electrónicamente por:
FABRICIO BOLIVAR
BERMEO ROMERO

Ing. Fabricio Bolívar Bermeo Romero

RESUMEN

Iniciamos con un análisis detallado del proceso actual, donde se identifican desafíos en la documentación de requerimientos para proyectos de desarrollo de software en una empresa de telecomunicaciones. Estos desafíos incluyen la falta de claridad en las definiciones de procesos, dificultades en la reunión de involucrados y la necesidad de agilizar el proceso.

Como solución, proponemos un modelo "to-be" que se centra en la creación de la Plataforma Web Colaborativa (PWC). Esta solución se fundamenta en estándares de calidad, proporcionando análisis en tiempo real y una experiencia de usuario intuitiva.

El diseño arquitectónico de la PWC se articula en torno a componentes claves como el sistema de autenticación y el sistema de notificaciones. Este diseño busca garantizar la seguridad, accesibilidad y eficacia de la plataforma. Los prototipos, por otro lado, materializan conceptos cruciales que establecen una base sólida para recibir retroalimentación temprana.

En conjunto, este enfoque integral sienta las bases para una transformación cultural y tecnológica en el desarrollo de software. La PWC no es simplemente una herramienta, sino un cambio significativo que redefine la manera en que la empresa afronta los proyectos de desarrollo de software, optimizando procesos, mejorando la comunicación y elevando la eficiencia en cada etapa del ciclo de vida del software.

ÍNDICE GENERAL

ABREVIATURAS Y SIMBOLOGÍA

CCB	Tablero de control de cambios.
DERCAS	Documento de Especificación de Requerimientos por el Cliente.
PWC	Plataforma web colaborativa.
RCM	Gestión de cambios del requisito.

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En el complejo tejido del desarrollo de software en empresas de telecomunicaciones, la correcta documentación de requerimientos se presenta como el epicentro de una eficiente gestión de proyectos. En el corazón de una destacada compañía del sector, surge una problemática que resuena en numerosas organizaciones: el desafío crítico de capturar de manera precisa y ágil los requisitos para proyectos de desarrollo de software.

Este proceso, vital para el éxito de los proyectos, se ve afectado por una serie de obstáculos que comprometen su eficacia. La falta de claridad en las definiciones, las dificultades para coordinar a los actores involucrados y la urgencia de optimizar los tiempos de elaboración de documentos han delineado un panorama que exige una solución.

Es en este contexto que se centraliza el proyecto presente, guiada por la visión de superar estos desafíos mediante la introducción de la Plataforma Web Colaborativa (PWC). Esta propuesta no solo busca subsanar las deficiencias actuales, sino que aspira a rediseñar por completo la manera en que la empresa encara el desarrollo de software. A lo largo de esta introducción, exploraremos la raíz de esta problemática, analizaremos los obstáculos presentes en el proceso actual y presentaremos una visión que busca cambiar la narrativa del desarrollo de software

CAPÍTULO 1

GENERALIDADES

En el siguiente capítulo, se presentarán los antecedentes de la empresa, así como los desafíos que actualmente enfrenta en uno de sus procesos de requerimientos para el desarrollo de software por parte de los usuarios encargados. Además, se expondrá la propuesta de solución y la metodología que se empleará. Asimismo, se establecerán tanto el objetivo general como los objetivos específicos que abarcará este proyecto.

1.1. ANTECEDENTES

En el ámbito del desarrollo de software en empresas de telecomunicaciones, los requerimientos de usuarios han surgido como un factor crítico para el éxito de los proyectos. La importancia de comprender y documentar de manera correcta los requerimientos de los usuarios ha sido destacada por varios estudios en ingeniería de software.

La ingeniería de requerimientos se ha consolidado como una disciplina esencial en el proceso de desarrollo de software, según lo señalado por diversos expertos [1]. La calidad y claridad de los requerimientos del usuario impactan directamente en la calidad del software resultante [2]. Además, la falta de comprensión adecuada de los requerimientos afecta a los tiempos de los ingenieros de software ya que conlleva a retrabajos y que el proyecto no finalice en la fecha estimada [2].

Las plataformas web, definidas como sistemas instalados en hardware que ofrecen servicios de software [3], han demostrado ser entornos propicios para la colaboración y la gestión eficiente de información.

A pesar de estos avances, la literatura también destaca desafíos persistentes en la elaboración de requerimientos de usuarios. La falta de metodologías claras y efectivas en la recopilación y análisis de

requerimientos puede llevar a malentendidos y errores en la implementación del software [1].

1.2. DESCRIPCIÓN DEL PROBLEMA

En el departamento de sistemas de una empresa de telecomunicaciones con una amplia estructura organizativa, que comprende más de cinco mil empleados distribuidos en aproximadamente ochenta departamentos, se identifica un desafío crítico en el proceso de desarrollo de software. La empresa enfrenta obstáculos significativos en la documentación precisa y completa de los requerimientos que realiza los usuarios encargados de elaborar los documentos para la implementación de nuevos proyectos de desarrollo de software. Actualmente, el proceso de documentación presenta notables deficiencias, no contiene la información completa, precisa lo que complica entender los procesos o definiciones realizadas y efectuar un análisis completo para la implementación del proyecto, esto provoca que el documento tenga que ser enviado nuevamente al usuario para su corrección desembocando en tener muchas reuniones para realizar definiciones no contempladas tanto en los procesos internos o externos donde se involucran a otros departamentos de la empresa que tienen afectación en el requerimiento solicitado lo que provoca que no se inicie el proyecto en la fecha estimada.

1.3. SOLUCIÓN PROPUESTA

En respuesta a las deficiencias identificadas en la documentación de usuarios en el departamento de sistemas de una empresa de telecomunicaciones, propongo una solución: el diseño de una Plataforma Web Colaborativa (PWC). Inspirándonos en los estándares de calidad de IEEE [4], esta plataforma estará diseñada meticulosamente para capturar de manera detallada y precisa los requerimientos del usuario. Siguiendo las pautas de colaboración de Parker y sus colegas [5], la PWC permitirá discusiones en tiempo real y aclaraciones instantáneas, mejorando así la comprensión mutua entre los usuarios y los desarrolladores. Para garantizar la calidad de los requerimientos, implementaremos herramientas analíticas avanzadas basadas en la metodología de STARTS [6], que proporcionarán análisis en tiempo real sobre la documentación, ofreciendo retroalimentación instantánea y detallada.

La PWC contará con una interfaz de usuario intuitiva y fácil de usar, fomentando la participación de usuarios técnicos y no técnicos. Además, se priorizará la seguridad de los datos sensibles, estableciendo un nuevo estándar para futuros proyectos en el departamento de sistemas. Esta solución está diseñada no solo para mejorar la eficiencia y calidad del proceso de desarrollo de software,

sino también para promover la colaboración efectiva entre todas las partes involucradas.

Estoy convencido de que esta solución representa una respuesta integral a los desafíos identificados. Mi enfoque se basa en una comprensión profunda de los estándares de la industria, como se evidencia en las referencias [4], [5] y [6].

1.4. OBJETIVO GENERAL

Diseñar una plataforma Web Colaborativa basada en tecnologías Angular y microservicios en Java, alojada en Amazon Web Services (AWS) para optimizar el proceso de documentación de los usuarios en proyectos de desarrollo de software del departamento de sistemas de una empresa de telecomunicaciones. El propósito es mejorar la calidad y la comprensión de los requerimientos del usuario, reducir la falta de información y fomentar la participación de las áreas afectadas en el proceso de especificación de requerimientos, con el fin de lograr un desarrollo de software más eficiente y satisfactorio para todas las partes involucradas.

1.5. OBJETIVOS ESPECÍFICOS

- Investigar las mejores prácticas y soluciones implementadas en plataformas web colaborativas similares, analizando sus éxitos y

desafíos para aplicar lecciones aprendidas en el diseño del proyecto.

- Realizar un análisis detallado de las necesidades y requerimientos de los usuarios encargados de elaborar proyectos, incluyendo entrevistas y encuestas, para comprender completamente sus expectativas y desafíos.
- Diseñar la arquitectura de la plataforma web colaborativa, centrándose en la implementación de microservicios en Java y Angular para la interfaz de usuario, priorizando la facilidad de uso, la accesibilidad y la integración fluida con otros sistemas para crear una experiencia de usuario intuitiva y eficaz.

1.6. METODOLOGÍA

El alcance de este proyecto implica llevar a cabo un estudio no experimental de tipo transversal empleando entrevista y encuestas diseñadas específicamente para capturar información relevante de los usuarios encargados de elaborar los requerimientos para los nuevos proyectos de desarrollo de software.

El proceso de levantamiento de información se realizará de manera cuidadosa y colaborativa, utilizando reuniones online para garantizar la participación de los usuarios clave. Durante estas interacciones, se

recopilarán datos detallados sobre los requisitos y desafíos actuales que enfrentan en el proceso de documentación de requerimientos.

Los datos recopilados se analizarán utilizando técnicas estadísticas y herramientas de análisis de datos, permitiendo la identificación de correlaciones y áreas críticas en el proceso de documentación. Estos análisis proporcionarán una base sólida para el diseño de la Plataforma Web Colaborativa.

1.7. RESULTADOS ESPERADOS

- Recopilar información detallada y específica sobre los desafíos enfrentados por los usuarios encargados en el proceso de documentación de requerimientos.
- Prototipo funcional de la plataforma web colaborativa, demostrando su viabilidad y funcionalidad para mejorar la documentación de requerimientos.
- Minimizar significativamente la necesidad de correcciones y modificaciones posteriores al trabajo del equipo de desarrollo, gracias a una documentación de requerimientos más precisa y completa desde el principio.
- Aumentar la calidad general de los proyectos mediante la implementación de requerimientos más claros y comprensibles, reduciendo la probabilidad de errores.

- Mitigar errores críticos y disminuir las posibilidades de fallos en el ambiente de producción al garantizar que los requerimientos estén correctamente documentados y entendidos por todas las partes involucradas en el proceso de desarrollo.

CAPÍTULO 2

MARCO TEÓRICO

En este capítulo, se presenta un marco teórico integral que sienta las bases necesarias para la propuesta de diseño de la plataforma web colaborativa, abordando aspectos como la definición de plataformas web, la metodología de requerimientos de usuarios, el ciclo de vida del desarrollo de software y casos de estudio relevantes. Estos fundamentos teóricos son cruciales para abordar los desafíos identificados en la elaboración de requerimientos. La comprensión profunda de estos elementos teóricos contribuirá a mejorar la calidad en la entrega final de los proyectos, la reducción del retrabajo y la minimización de errores en el ambiente de producción.

2.1. PLATAFORMA WEB

2.1.1. DEFINICIÓN

Una plataforma web se define como un sistema instalado en hardware que proporciona servicios de software [7]. Se trata de un entorno en línea que habilita a los usuarios para acceder, compartir, gestionar información y recursos a través de internet. Utiliza tecnologías web que ofrecen una interfaz de usuario accesible a través de navegadores web estándar, posibilitando el acceso desde cualquier dispositivo con conexión a internet [3]. De acuerdo con otros estudios, como el de [8], una plataforma web facilita el funcionamiento de módulos específicos de hardware y software. Estas plataformas se definen mediante estándares que ayudan a establecer una arquitectura de hardware. Esta arquitectura no solo define la estructura y el comportamiento de la plataforma web, sino también aspectos como usabilidad, funcionalidad, rendimiento, flexibilidad, reutilización de código, fácil comprensión, restricciones y los compromisos tanto económicos como tecnológicos. Cabe destacar que la arquitectura aborda las distintas etapas que conforman el ciclo de vida del software.

2.1.2. COMPONENTES Y ELEMENTOS

Las plataformas web incorporan interfaces de usuario intuitivas para brindar una experiencia amigable a los usuarios. Asimismo, estas plataformas utilizan bases de datos robustas para garantizar el almacenamiento seguro de la información. Además, integran algoritmos sofisticados que facilitan la interacción entre los usuarios y sistema. Además, se implementan sistemas de seguridad avanzados para salvaguardar los datos y las transacciones realizadas en la plataforma [5].

En un estudio sobre la implementación de una plataforma web para la gestión y control documental [3], se identificaron los siguientes componentes clave:

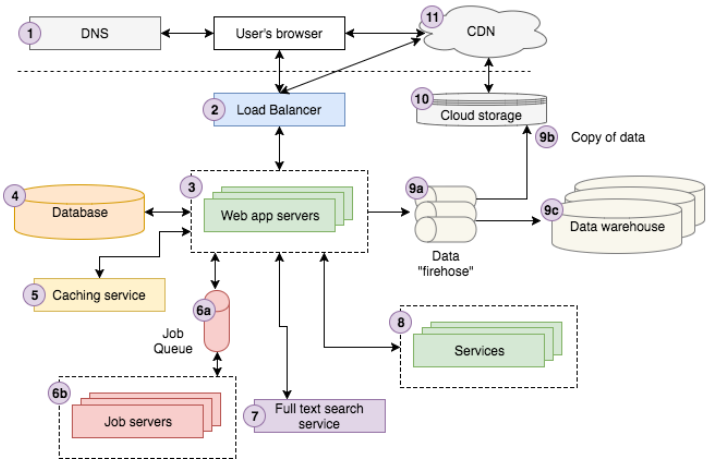


Figura 2.1: Descripción de la arquitectura de aplicaciones web

Fuente: Arquitectura Web [9]

2.1.3. BENEFICIOS

Las plataformas web ofrecen una variedad de beneficios, desde mejorar la colaboración y reducir errores hasta permitir una gestión eficiente del tiempo y fomentar la adaptabilidad continua a medida que evolucionan las necesidades del proyecto y los usuarios, otros autores como [5] destaca los beneficios fundamentales de las plataformas web, como la capacidad para conectar diversos grupos de usuarios de manera eficiente y efectiva, facilitar transacciones y colaboraciones sin problemas, proporcionar experiencias altamente personalizadas a los usuarios.

En resumen, algunos de estos beneficios incluyen:

- **Accesibilidad Global:** Las plataformas web ofrecen una interfaz accesible mediante navegadores estándar desde cualquier dispositivo que tenga conexión a internet. Esto asegura que los interesados y los equipos de desarrollo puedan acceder y colaborar en tiempo real sin importar su ubicación geográfica, facilitando una comunicación efectiva a nivel global.
- **Colaboración Eficiente:** Fomentan la colaboración entre equipos multidisciplinarios y partes interesadas. Los

usuarios, desarrolladores y otros participantes pueden intercambiar ideas, revisar documentos y debatir detalles del proyecto en un entorno digital colaborativo. Esto impulsa la co-creación y contribuye a evitar malentendidos al mantener a todos los involucrados en la misma sintonía.

- **Reducción de Errores y Ambigüedad:** Al ofrecer un espacio centralizado para la documentación y revisión de requerimientos, las plataformas web contribuyen a minimizar errores y ambigüedades en los requisitos del usuario. La información se presenta de manera estructurada, facilitando la comprensión y disminuyendo las interpretaciones erróneas, lo que conlleva a una reducción de errores en las fases subsiguientes del desarrollo.
- **Gestión Eficiente del Tiempo:** La administración de requerimientos mediante plataformas web facilita una comunicación más precisa y ordenada, generando ahorro de tiempo en reuniones presenciales. Además, al tener la información centralizada y disponible en cualquier momento, se reduce el tiempo invertido en la búsqueda de documentación o datos pertinentes.

- **Adaptabilidad y Escalabilidad:** Estas plataformas presentan una notable flexibilidad y capacidad de expansión, permitiendo a las organizaciones incorporar nuevas funciones o realizar ajustes de acuerdo con las demandas cambiantes del proyecto. Su versatilidad asegura que el sistema pueda desarrollarse en paralelo con la evolución del proyecto.

2.2. METODOLOGÍA DE LOS REQUERIMIENTOS DE USUARIOS

2.2.1. EVOLUCIÓN DE LAS METODOLOGÍAS

La gestión de cambios constituye un desafío complejo en el proceso de ingeniería de requisitos, dada la dificultad de identificar requisitos del sistema que sean integrales y reflejen situaciones contemporáneas, al mismo tiempo que se adapten a necesidades en constante cambio. Las variaciones en las demandas de los clientes, las modificaciones en el mercado, la competencia y las regulaciones gubernamentales ejercen una influencia significativa en la alteración de los requisitos del sistema. De acuerdo con otros estudios, como se indica en [10], el cambio de requisitos se define como la tendencia de los requisitos a cambiar con el tiempo en reacción a las necesidades

cambiantes de los clientes, las partes interesadas, las empresas y el entorno laboral.

Es fundamental destacar que uno de los desafíos más críticos que las empresas de desarrollo de software enfrentan es la constante modificación en la redefinición de los procesos. En este sentido, la Gestión del Cambio de Requisitos (RCM) es muy importante para la implementación de los proyectos de software.

La evolución de las metodologías empleadas para los requisitos de usuario se fundamenta en la idea de que el RCM es un proceso impulsado por la colaboración. Un RCM efectivo requiere una comunicación y coordinación significativas entre las partes involucradas.

La falla en la gestión del cambio de requisitos puede resultar en elevados costos de desarrollo, retrasos en los cronogramas, requisitos inestables y pruebas de usuario interminables, lo que, en última instancia, conduce al fracaso total del proyecto.

Se han creado diversos modelos y marcos para la Gestión del Cambio de Requisitos (RCM) con el propósito de asistir a las organizaciones de desarrollo de software en la eficiente gestión de los cambios en los requisitos de software. Así como [11] diseñó un marco que detalla los procedimientos de inicio,

evaluación y selección de cambios en la RCM. Este enfoque ofrece información específica sobre cómo llevar a cabo la gestión del cambio, teniendo en cuenta factores críticos de tiempo y costos necesarios para la implementación de los cambios requeridos. De modo similar [12] presentó un modelo formal de proceso para la gestión de cambios, compuesto por seis etapas principales: "iniciar", "recibir", "evaluar", "aprobar o desaprobar", "implementar" y "configurar". Aunque este modelo guía las actividades de gestión de cambios, carece de algunas fases esenciales, como la verificación y lotes [13].

El modelo RCM de [14] fue introducido para abordar los elementos fundamentales del proceso de gestión de cambios de requisitos. Este modelo identifica seis fases principales: "solicitud de cambio", "rechazo", "lote", "implementación" y "actualización". Cada cambio se ingresa en el tablero de control de cambios (CCB), que valida el cambio utilizando todas las fases del modo [14]. Sin embargo, una desventaja de este modelo es la falta de un mecanismo de comunicación tanto para el cliente como para los proveedores. Los modelos detallados anteriormente nos ayudan a gestionar los cambios de requisitos exigidos, desarrollar software de calidad y satisfacer las expectativas del cliente [13], [14], [15].

2.2.2. CATEGORÍA DE LOS REQUERIMIENTOS

La metodología de los requerimientos de usuarios desempeña un papel central en el desarrollo de software, ya que establece el proceso mediante el cual se recopilan, analizan y documentan los requisitos del usuario, una metodología efectiva garantiza la comprensión clara y precisa de las necesidades de los usuarios, lo que es esencial para el éxito de cualquier proyecto de software. De acuerdo con [1] los requerimientos de usuarios se dividen en dos categorías fundamentales:

- **Requerimientos Funcionales:** Los requerimientos funcionales son las funciones que el sistema debe realizar, se detallan los cambios en el sistema con sus diferentes entradas para producir sus respectivas salidas. Debemos mencionar lo importante de definir de una forma clara el ¿Qué? y no el ¿Cómo? se deben realizar estos cambios que se necesitan en el proyecto. Estos requerimientos al tiempo que se avanza y posterior a la finalización del proyecto, se convierten en los algoritmos, las reglas del negocio y código del sistema.
- **Requerimientos No Funcionales:** Los requerimientos no funcionales son particularidades que pueden limitar el

sistema, por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares, etc.

Es fundamental tener en cuenta que un requerimiento debe cumplir con ciertos criterios para ser efectivo:

- **Especificado por escrito:** Todo requerimiento debe estar claramente definido por escrito, similar a un contrato o acuerdo entre las partes involucradas en el proyecto.
- **Etapa de pruebas:** Un requerimiento debe ser verificable para determinar si se ha cumplido adecuadamente. La capacidad de prueba garantiza que el requerimiento sea medible y cumpla con los criterios establecidos.
- **Conciso:** Se refiere que sea fácil de leer y entender. La documentación debe ser simple y clara para aquellos que lo consulten en el futuro, facilitando la comprensión y la implementación.
- **Completo:** Un requerimiento está completo si no necesita detalles adicionales en su redacción. Debe tener toda la información necesaria para fácil comprensión sin la necesidad de explicaciones adicionales.

- **Consistente:** Que el requerimiento sea consistente, que no sea discordante con otro requerimiento. La coherencia en los requisitos es esencial para evitar conflictos durante el desarrollo.
- **No ambiguo:** Un requerimiento no debe ser ambiguo y debe ser interpretado de una forma única. En la definición de su lenguaje no debe provocar confusiones y debe ser comprensible para todos los stakeholders involucrados en el proyecto.

Cumplir con estos criterios garantiza que los requerimientos de usuarios sean claros, comprensibles y, lo que es más importante, alcanzables durante el desarrollo del software. Estos principios fundamentales son pilares en la metodología de los requerimientos de usuarios y son esenciales para el éxito del proyecto de software.

2.3. CICLO DE VIDA DEL DESARROLLO DE SOFTWARE

2.3.1. DEFINICIÓN

El ciclo de vida del desarrollo de software es un proceso estructurado que describe las etapas a través de las cuales un software evoluciona desde la concepción de la idea hasta su entrega y mantenimiento. Según [16] el ciclo de vida del software está constituido por el conjunto de todas las etapas que preceden y suceden a la producción de software

Figura 2.2: Descripción del ciclo de vida del desarrollo

Fuente: Autor

2.3.2. MODELOS DE CICLO DE VIDA

La existencia de diferentes modelos de ciclos de vida del software no permite comprender que no exista uno ideal o que no tenga limitaciones, sin embargo, se torna muy importante que los proyectos se puedan implementar con un ciclo de vida bien definido [17].

Todos los sistemas de información pasan por una serie de fases desde su inicio hasta su final. Según [18] el ciclo de vida comprende las siguientes etapas:

- **Planificación:** es la etapa de coordinar un proyecto, se definen las tareas iniciales de su fase inicial aquí se debe incluir y tener claro el alcance que tendrá el proyecto, realizar un estudio sobre la viabilidad, el análisis de los riesgos asociados al proyecto, una estimación del valor del proyecto, una planificación estimada y la entrega de recursos a las distintas etapas del proyecto.
- **Análisis:** En esta etapa es donde se necesita una comprensión clara, precisa y total de los requerimientos del sistema. Es una etapa muy importante ya que se debe obtener y entender con precisión lo que se necesita, caso contrario en otra etapa del proyecto no se logrará

obtenerlo. En esta etapa vamos a encontrarnos con muchos temas o definiciones que no estarán claras por parte del usuario o cliente, por lo que se torna necesario realizar reuniones para aclarar lo solicitado.

- **Diseño:** aquí dibujaremos, especificaremos los diferentes modelos que vamos a utilizar, que nos permiten la implementación de manera efectiva del sistema, es decir, se preparan las diferentes alternativas para el desarrollo del sistema y se defina la estructura general que debemos seguir (diseño de la arquitectura). Es importante tener en cuenta que la propuesta inicial de diseño no siempre es la adecuada, por lo que debe refinarse y, por lo que no es necesario iniciar de cero, podemos utilizar patrones de diseño que nos permitan evitar errores que otros han cometido.
- **Implementación:** luego de tener contexto y claridad sobre la funcionalidad que debe tener el proyecto y tomar las decisiones sobre los distintos componentes (diseño) que se utilizaran, iniciaremos su desarrollo. Antes de iniciar a programar el sistema o crear la base de datos, es fundamental haber comprendido la problemática o la necesidad que pretende resolver y haber aplicado

principios básicos de diseño que nos permita construir un sistema de alta calidad. Aquí nos toca seleccionar las herramientas adecuadas, el entorno de desarrollo, el lenguaje de programación a utilizar para el sistema que vamos a construir. En el momento de programar debemos tener en cuenta que el código sea lo más sencillo de comprender. Para que el código sea legible, debe tomarse en cuenta aspectos como seleccionar algoritmos y estructuras de datos adecuados, mantener la lógica de la aplicación de manera sencilla, documentar cada proceso del sistema, facilitar la interpretación visual, entre otros.

- **Pruebas:** en esta fase nos enfocaremos en detectar los errores que se cometen en las etapas anteriores del proyecto (y procurar corregirlos). Existen diferentes tipos de pruebas:
  - **Pruebas de unidad:** sirven para validar el funcionamiento de un fragmento de código o componente específico.
  - **Pruebas de integración:** se realizan cuando se necesitan unir varios o todos los componentes que conforman el sistema, nos ayudara a detectar errores en sus interfaces.

  - **Pruebas alfa:** las realiza el usuario final después de finalizado el sistema, el objetivo es ayudar a pulir aspectos de la interfaz de usuario.
  - **Pruebas beta:** se realizan cuando el sistema no es un producto a la medida, sino que se venderá como un producto en el mercado. Son realizadas por usuarios finales, ajenos al equipo de desarrollo.

- **Test de aceptación:** se utilizan en sistemas a la medida, si es aceptado de manera exitosa, con esto se finaliza la etapa de implementación e inicia la fase de mantenimiento. Por último, a lo largo del ciclo de vida, se realizan revisiones de todos los productos generados a lo largo del proyecto, desde la especificación de requerimientos hasta el código de los diferentes módulos que integran la aplicación.
- **Instalación o despliegue:** antes de poner el sistema en modo productivo, se debe planificar el ambiente o entorno donde deberá funcionar el sistema, evaluando las características de hardware y software: equipos necesarios y su configuración física, redes de interconexión entre equipos internos y externos, sistemas

operativos actualizados, bibliotecas, componentes suministrados por terceros, entre otros.

- **Uso y mantenimiento:** detallaremos los tres tipos de mantenimiento que pueden darse:
  - **Mantenimiento correctivo:** solventar los errores que se detecten en la vida útil del sistema.
  - **Mantenimiento adaptativo:** adecuar al sistema para futuras necesidades (por ejemplo, incrementar características hardware, actualizaciones en software).
  - **Mantenimiento perfectivo:** se detectan mejoras que nos permitirán mejorar o agregar nuevas funcionalidades al sistema.

Los enfoques tradicionales de ciclo de vida se fundamentan en una metodología estructurada, lo que implica que cada tarea se divide en unidades más pequeñas e independientes. Algunos de los ejemplos más reconocidos incluyen los siguientes:

- **Modelo en Cascada:** Según [12] modelo que se debe avanzar en orden, de una etapa a otra si se finaliza con éxito las tareas de verificación y validación propias de la etapa. Si se detectan errores, se debe regresar a la fase inmediatamente. Este modelo tradicional, exige una

aproximación secuencial al proceso de desarrollo de software, sin embargo, se torna muy complicado debido a que los proyectos reales raramente siguen el flujo secuencial de actividades que propone este modelo. Normalmente, es difícil conocer todos los requerimientos del proyecto desde el inicio, no existe una versión operativa del sistema hasta las etapas finales del proyecto, por lo que cualquier rectificación de cualquier decisión tomada de manera equivocada, supondrá un coste adicional, tanto económico como temporal.

- **Modelo Iterativo-Incremental:** Este modelo proviene del modelo en cascada, nos ayuda a minimizar el riesgo que surge entre lo solicitado por el usuario y el producto final, por malentendidos durante la etapa de levantamiento de información. Por cada iteración se finalice se deberá entregar al cliente una versión mejorada o más funcionalidades para el producto.
- **Modelo en Espiral:** Este modelo hace énfasis en la gestión de riesgos y definen cuatro actividades principales:
    a. Planificación: determinar objetivos, alternativas y restricciones del proyecto

b. Análisis de riesgos: analizar alternativas, identificar y resolver riesgos
c. Ingeniería: Desarrollo del producto
d. Evaluación: Aceptación por parte del cliente y dar una puntación a los resultados obtenidos frente a la siguiente iteración.

- **Modelo en V:** Modelo similar al modelo de cascada diseñado por Alan Davis, la diferencia con el modelo mencionado, se agrega la etapa de retroalimentación entre las etapas de análisis y mantenimiento, y entre las de diseño y debugging.
- **Modelo Orientado a Objetos:** Esta técnica fue presentada en la década del 90. Al igual que la filosofía del paradigma de la programación orientada a objetos, en esta metodología cada funcionalidad, o requerimiento solicitado por el usuario, es considerado un objeto. Los objetos están representados por un conjunto de propiedades, a los cuales denominamos atributos, por otra parte, al comportamiento que tendrán estos objetos los denominamos métodos. (Universidad del Papaloapan, 2015).

2.4. ARQUITECTURA DE SOFTWARE

2.4.1. DEFINICIÓN

La arquitectura es un modelo para un sistema que nos proporciona una abstracción asociada para gestionar la complejidad de un sistema, estableciendo comunicación y mecanismo de coordinación entre elementos, el resultado debe tener una estructura definida para cumplir con los requisitos técnicos, los requerimientos de operación, así mismo buscar optimizar los atributos de calidad como son el rendimiento y seguridad. Se define una resolución estructurada para cumplir con todos los requisitos técnicos y necesidades operativas, optimizando al mismo tiempo los atributos de calidad comunes como el rendimiento y seguridad.

2.4.2. PATRONES DE DISEÑO

Las características describen las expectativas del código en niveles operativos y técnicos. El desempeño y baja tolerancia a fallos, cuantificabilidad y la responsabilidad de la unidad de área es la clave de características. También se le conoce como atributo de calidad o patrones. Los microservicios son uno entre varios patrones de diseño de código Patrón basado en eventos, sin servidor Patrón y mucho más. El patrón de microservicios

recibió su nombre al ser adoptado por Amazon y Netflix y mostrando su gran impacto.

- **Arquitectura sin servidor:** La arquitectura sin servidor se divide en dos principales grupos. El primero es "Backend como servicio", que es conocido como 'Baas'. Otro es "Funciones como servicio" (FaaS).” Lo cual se conoce como FaaS. La arquitectura sin servidor le ayudará a ahorrar mucho tiempo Cuidado y corrección de errores de implementación y servidores tareas regulares.
- **Arquitectura basada en eventos:** Esta arquitectura cuenta con los productores de eventos y clientes de eventos. El mayor plan es desvincularse de los componentes de su sistema y cada mitad van para ser desencadenado una vez que un evento notable de la otra mitad se ha activado. Como ejemplo, el sistema de tienda en Internet tiene dos componentes. Primero un módulo de venta y segundo un módulo de comerciante. si un cliente realiza una venta, el módulo de adquisición generaría una ocasión de “pedido Pendiente” Desde el vendedor módulo es interesante dentro del “orden Evento "Pendiente", estará escuchando, en caso de que alguno esté motivado. Una vez que el módulo de vendedor reciba este evento,

ejecutar algunas tareas o incluso provocar otro evento para pedir mucha mercancía a un comerciante.

- **Arquitectura de microservicios:** La arquitectura de microservicios es una arquitectura famosa. Hoy en día depende del desarrollo de pequeños autónomos servicios estándar donde cada servicio resuelve una desventaja o realiza una tarea singular y estos. Los módulos se comunican entre sí a través de API bien definida para servir al objetivo empresarial.

2.4.3. MODELO DE ARQUITECTURA DE SOFTWARE C4

El modelo C4 nos ayuda a los equipos de desarrollo de software a describir y comunicar la arquitectura de software, tanto durante las sesiones de diseño inicial como cuando documentó retrospectivamente una base de código existente. Es una forma de crear mapas de tu código, a varios niveles de detalle, como ejemplo podemos relacionar a Google Maps cuando necesitamos acercarnos y salir de un área que tenemos como objetivo inspeccionar.

El modelo C4 tiene cuatro tipos de diagramas, de ahí el origen de su nombre, ya que son las iniciales del nombre de cada

componente, cada uno contiene su propio nivel de detalle y un objetivo diferente, en este modelo C4 tenemos:

- **Diagrama de contexto del sistema [Context]:** El diagrama de Contexto de un sistema (primer nivel), es el inicio para diagramar y documentar un sistema de software, esto nos ayuda tener una visión general sobre el sistema a diseñar. En esta etapa, lo importante es tener un diagrama que el usuario final pueda entender sobre la estructura/componentes del sistema mas no un detalle técnico sobre la solución del sistema.
- **Diagrama de contenedores [Containers]:** Diagrama de segundo nivel, una vez que entendemos el sistema y su ajuste al entorno de TI en general, el siguiente paso se considera tener más detalles describiendo los contenedores de nuestro sistema, como se comunican e interactúan. Un "contenedor" es una aplicación web junto al servidor, una aplicación de una sola página, aplicación de escritorio, aplicación móvil, esquema de base de datos, sistema de archivos, etc. Esencialmente, un contenedor es una unidad que se ejecuta por separado (por ejemplo, un espacio de proceso separado) que ejecuta código o almacena datos. El diagrama de contenedor muestra la

forma de alto nivel de la arquitectura de software y cómo responsabilidades se distribuyen a través de ella. También muestra las principales opciones tecnológicas y cómo se comunican los contenedores unos con otros. Es un diagrama de enfoque de tecnología simple y de alto nivel que es útil para desarrolladores de software y personal de soporte/operaciones por igual.

- **Diagrama de componentes [Components]:** Llegamos al nivel tres, donde podemos detallar y descomponer cada contenedor para poder identificar/describir los principales bloques estructurales de construcción y sus interacciones. El diagrama de componentes muestra cómo un contenedor se compone de un número de "componentes", lo que cada uno de ellos componentes son, sus responsabilidades y los detalles de la tecnología y la implementación, es posible que un sistema tengas más de un diagrama de componentes.
- **Diagrama de código [Code]:** En este último nivel, podemos aplicar una ‘radiografía’ para cada componente que tiene nuestra solución, para visualizar cómo se implementa como código, utilizando diagramas de clase UML, diagramas de relación de entidad o similares, lo

ideal es generar este código automáticamente mediante herramientas (herramienta de modelado IDE o UML), y debería considerar mostrar sólo aquellos atributos y métodos que le permiten contar la historia que desea contar. Este nivel de detalle no se recomienda y es una vista opcional.

2.5. CASOS DE ESTUDIOS APLICADOS

En esta sección vamos a mencionar casos aplicados sobre el flujo de trabajo que pueden tener desde conjuntos de software, flujos de trabajo manuales que son documentados y codificados, es así, donde encontramos a la biblioteca de la Universidad del Norte de Texas (UNT) donde administran un Repositorio Institucional llamado 'UNT Scholarly Works', el objetivo fue mejorar los flujos de los procesos para que sean flexibles y extensibles, así como de proporcionar un flujo de trabajo claro para que su personal procese el contenido enviado.

La biblioteca de la UNT trabaja con un repositorio institucional para recopilar, preservar y poner a disposición documentación académica y creativa de sus profesores y estudiantes. El repositorio institucional, UNT Scholarly Works (https://digital.library.unt.edu/scholarlyworks/), se ha implementado como una colección en la Biblioteca Digital de la UNT (https://digital.library.unt.edu/) junto con otras colecciones como:

- Tesis y Disertaciones Electrónicas de la UNT.
- Repositorio de Datos de la UNT.
- Trabajos de Graduado de la UNT.
- Trabajos de Pregrado de la UNT.
- UNT Scholarly Works.

La forma de recopilación la tuvieron durante mucho tiempo y creció hasta que la situación se tornó inmanejable (alrededor de 6.100 documentos para procesarse), la forma de recopilación se lo han hecho por diferentes medios de comunicación y flujos de trabajo basados en correo electrónico para recibir nuevos documentos, y un flujo de trabajo basado en carpetas utilizadas para proyectos digitales en la División de Bibliotecas Digitales de la UNT.

¿Como surgió el cambio? El Ingreso de un nuevo administrador y la necesidad de conocer los flujos de trabajo, en la revisión de los procesos se presentaron inconvenientes debido a que la documentación no era exhaustiva ni detallada como se lo necesitaba, otro punto a considerar para el cambio fue durante la pandemia COVID-19, ya que trasladó todo el trabajo presencial al trabajo remoto, es aquí donde se vio la necesidad de contar con flujos de trabajo documentados para que el personal tenga acceso y pueda trabajar sin edición de los contenidos sin interferir entre ellos.

UNT recurrió a una herramienta de gestión de proyectos, este es ‘GitLab’, que es una aplicación de gestión de código fuente y planificación de proyectos basada en la web que se basa en el sistema de control de versiones distribuido, que se lo utilizó para rastrear elementos y vincular sus procesos. Además de GitLab, la UNT implemento una wiki departamental que le permite tener su documentación más actualizada a medida que cambian sus flujos de trabajo y sus procesos.

El modelo que utilizó UNT, es la de flujos de procesos basados en carpeta, el componente principal que utilizó es el sistema de seguimiento de problemas, así mismo utilizo el componente para la creación de tickets llamado ‘IsSues’, que se lo utiliza como herramienta de seguimiento, con formatos establecidos, adicional a cada ‘problema’ se le asigna una o varias etiquetas, esto para facilitar la segmentación de los tipos de problemas, el objetivo es que cada proyecto tenga una carpeta creada en una unidad de red compartida, cada paso del flujo de trabajo se utiliza como un nombre de carpeta.

Podemos concluir que este software le permitió a UNT codificar aún más su flujo de trabajo utilizando las herramientas y funciones presentes en los proyectos de GitLab, como problemas, etiquetas y una wiki, sus esfuerzos se centraron en métodos fáciles de ajustar para organizar el

contenido, como los flujos de trabajo basados en carpetas, pueden realizar cambios en sus flujos de trabajo sin gastos generales adicionales. Así mismo, se encuentran trabajando con un sistema interno que les permite aprovechar la infraestructura existente para sus necesidades [19]

CAPÍTULO 3

DEFINICIÓN DE LA SITUACIÓN ACTUAL

En este capítulo, nos sumergimos en la evaluación detallada del estado actual del proceso de levantamiento de requerimientos en los proyectos de desarrollo de software dentro de una empresa de telecomunicaciones. Este análisis abarcará la definición y aplicación de una encuesta, así como el posterior análisis de los resultados obtenidos. Además, se examina el proceso de levantamiento de información, donde se detalla la recopilación y evaluación de datos esenciales para la elaboración del modelo (AS-IS)

3.1. DOCUMENTO DE REQUERIMIENTO

En la actualidad el documento de requerimiento para la solicitud de implementación de un nuevo proyecto lo llamaremos DERCAS, este documento se la ha asignado un identificador ‘FOR SIS 02’ y está diseñado para el uso exclusivo de una empresa de telecomunicación, las partes que constituyen este documento son:

- **Portada:** Se detallan los siguientes puntos:
    - Logotipo que identifica a la empresa de telecomunicaciones.
    - Un texto centrado que indica: ‘Levantamiento de requerimiento de software’.
    - Fecha de creación del documento.
    - Fecha ultima de modificación del documento.

Posterior a la portada, contiene 10 secciones que la detallamos a continuación:

- **Sección 1 - Objetivo:**

  En este apartado se detalla un texto que indica: ‘El propósito de este documento es registrar el levantamiento de información de los requerimientos de software en las diferentes aplicaciones del Grupo 'XYZ’ por parte de los usuarios solicitantes’.

- **Sección 2 - Información del requerimiento:**

El sponsor en esta parte debe llenar lo siguiente:

- **Empresa:** Nombre de la empresa a la que pertenece y va dirigido el proyecto.
- **Proyecto:** El nombre que le dará a su proyecto.
- **Cliente:** El nombre del sponsor.
- **Patrocinador principal:** Persona que sugirió el proyecto, puede ser que sea el mismo cliente.
- **Líder de proyecto:** Persona encargada de la implementación del proyecto.

- **Sección 3 - Propósito:**

  El sponsor debe detallar el objetivo principal del proyecto.

- **Sección 4 - Alcance de Requerimiento:**

  En esta sección el solicitante deberá indicar lo siguiente:

  - **Descripción del alcance:** Se incluye una corta descripción del alcance del requerimiento, beneficios que tendrá en su departamento con la implementación, metas.
  - **Requiere control de cambios:** En este apartado nos ayuda a llevar un control cuando la implementación tenga un impacto sobre sistemas de información o procesos existentes y se debe involucras a las Jefaturas o Gerencias afectadas.

- **Sección 5 - Funcionalidad:**

  El sponsor deberá listar las principales funcionalidades requeridas para su proyecto.

- **Sección 6 - Tipos de usuario:**

  Aquí se debe especificar como se van a clasificar los usuarios que van a utilizar el software o los nuevos procesos a implementar, puede ser por roles, por frecuencia de uso o nivel de experiencia del usuario.

- **Sección 7 - Reglas de negocio:**

  Un punto muy importante es el detalle de las validaciones/condiciones que el sponsor deberá detallar para el correcto funcionamiento del proyecto.

- **Sección 8 - Requerimientos funcionales:**

  Desde este punto y en adelante, este documento lo complementa el equipo asignado a la implementación del proyecto, y se debe detallar todas las actividades que se deberán realizar para el cumplimiento del proyecto, pueden ser una lista por cada acción a realizar.

- **Sección 9 - Requerimientos de interfaces externas:**

  Clasificaremos esta sección en:

- **Interfaces de Usuario:** En este punto se indican las características de cada interfaz con el usuario, ejemplos de pantallas que se van a utilizar.
- **Interfaces de Hardware:** Listar los dispositivos donde se podrá utilizar el nuevo software como computadoras, dispositivos móviles, impresoras, etc.
- **Interfaces de Software:** Interacción del software con componentes externos (base de datos, sistemas operativos, librerías)
- **Interfaces de comunicación:** Se detalla las funciones de comunicación que requiere el software, tales como: protocoles de comunicación de red, plantillas para envío de correos, formato de mensajería, requerimientos de encriptación.

- **Sección 10 - Control de Cambios:**

  Aquí llevaremos el control sobre las actualizaciones que se realicen al documento, donde llenaremos la siguiente información:

  - **Versión:** Numero incremental que inicia desde 1 cuando se entrega el documento.
  - **Acción:** Creación/Modificación del documento.

- **Fecha del cambio:** fecha en que se realiza la actualización.
- **Autor:** nombre de la persona responsable de la actualización.

3.2. DEFINICIÓN DE LA ENCUESTA

El presente proyecto en la metodología a usar contempla la definición de una encuesta a realizarse a los jefes departamentales que conforma al holding de la empresa que ven la necesidad de automatizar o mejorar los procesos actuales que tenemos en nuestros sistemas internos. A continuación, se detalla la encuesta utilizada en el proyecto:

Tabla 1: Encuesta

Fuente: Autor

Pregunta	Tipo
1. ¿Cuál es su nivel de satisfacción de elaborar un DERCAS (Documento de Especificación de Requerimientos por el Cliente)?	Cerrada
2. Seleccione los problemas que usted tiene al elaborar un DERCAS:	Cerrada
3. ¿Consideras necesario que el proceso de levantamiento de información se automatice?	Cerrada
4. ¿Cuáles son los beneficios que usted cree que debe tener este nuevo software en la gestión de requerimientos?	Opción múltiple
5. ¿Cómo describirías la comunicación actual en el proceso de desarrollo de software en la empresa?	Cerrada

6. ¿Qué herramientas o métodos utilizas actualmente para colaborar con otros miembros del equipo en la definición de requerimientos?	Abierta
7. Actualmente, ¿Cuántas horas usted estima que invierte semanalmente en la elaboración y revisión de un DERCAS?	Cerrada
8. ¿Cuántas horas usted está dispuesto a invertir semanalmente para realizar un DERCAS?	Cerrada
9. ¿Cuántos días te toma agendar reuniones con otros departamentos para realizar un DERCAS?	Cerrada
10. ¿Tienes alguna sugerencia o comentario adicional que pueda mejorar el proceso actual de gestión de requerimientos en la empresa?	Abierta

3.3. RESULTADO DE LA ENCUESTA

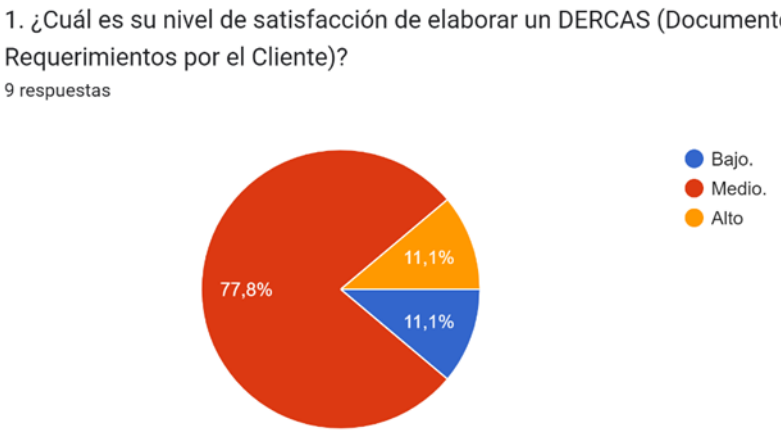


Figura 2.3: Resultado de pregunta 1

Fuente: Autor

Análisis e interpretación: La mayoría de los encuestados muestra un nivel de satisfacción medio al elaborar dercas, indicando una oportunidad de mejora en el proceso actual.

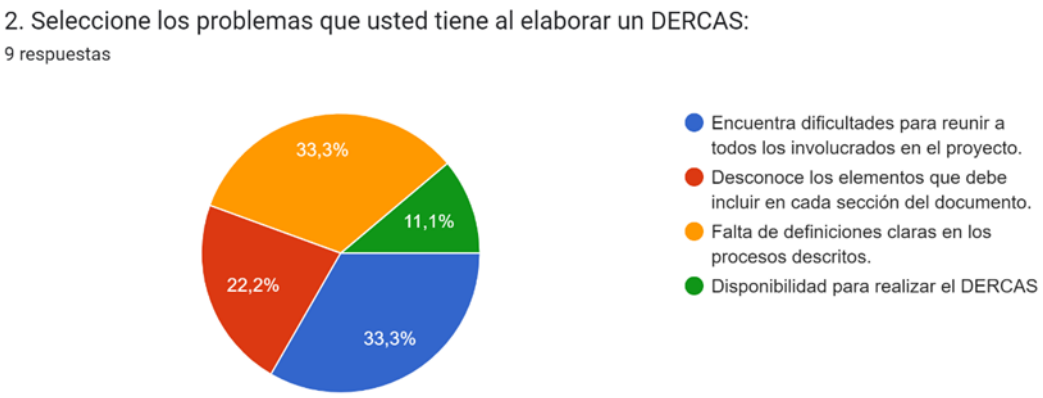


Figura 2.4: Resultado de pregunta 2

Fuente: Autor

Análisis e interpretación: Los problemas más destacados incluyen la dificultad para reunir a los involucrados y la falta de claridad en las definiciones de procesos.

Figura 2.5: Resultado de pregunta 3

Fuente: Autor

Análisis e interpretación: Existe un consenso unánime entre los encuestados sobre la necesidad de automatizar el proceso de levantamiento de información.

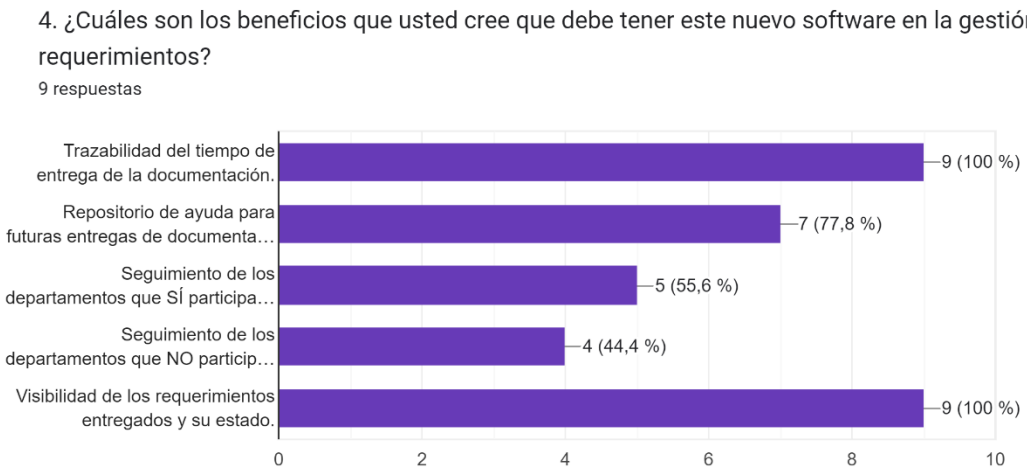


Figura 2.6: Resultado de pregunta 4

Fuente: Autor

Análisis e interpretación: Los encuestados esperan beneficios significativos, destacando la importancia de la trazabilidad y la visibilidad en el proceso.

Figura 2.7: Resultado de pregunta 5

Fuente: Autor

Análisis e interpretación: La mayoría percibe la comunicación actual como regular, señalando una oportunidad para mejorar la eficacia de las interacciones.

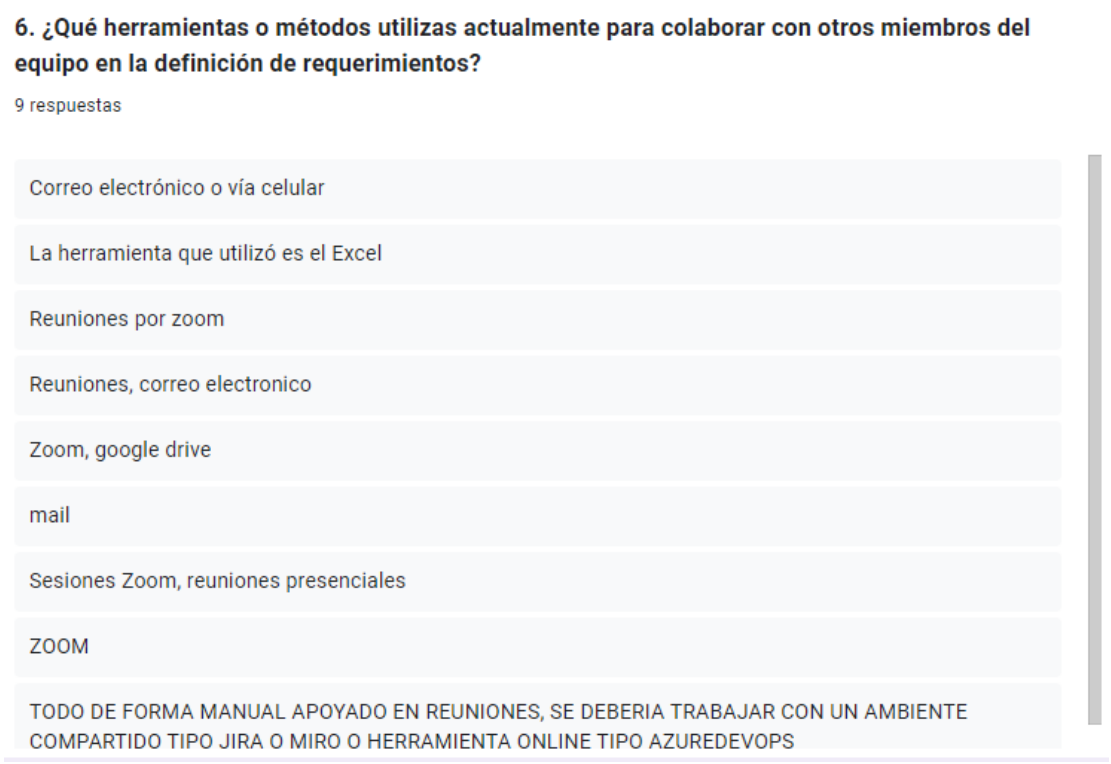


Figura 2.8: Resultado de pregunta 6

Fuente: Autor

Análisis e interpretación: La diversidad de herramientas utilizadas destaca la necesidad de consolidar y simplificar el proceso.

Figura 2.9: Resultado de pregunta 7

Fuente: Autor

Análisis e interpretación: La mayoría invierte hasta 4 horas semanales, indicando que simplificar el proceso puede ahorrar tiempo significativo.

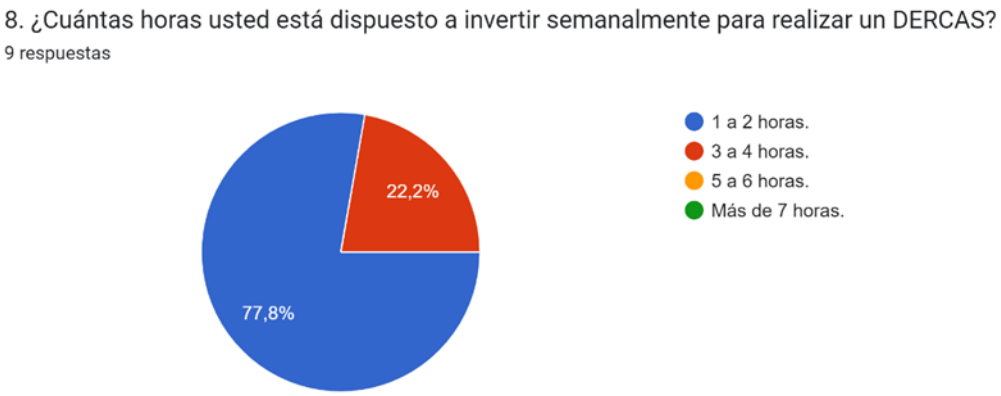


Figura 2.10: Resultado de pregunta 8

Fuente: Autor

Análisis e interpretación: Los encuestados muestran disposición a invertir tiempo, pero preferiblemente en intervalos más cortos.

Figura 2.11: Resultado de pregunta 9

Fuente: Autor

Análisis e interpretación: La mayoría experimenta demoras de hasta 5 días en agendar reuniones, lo que sugiere la necesidad de un proceso más ágil.

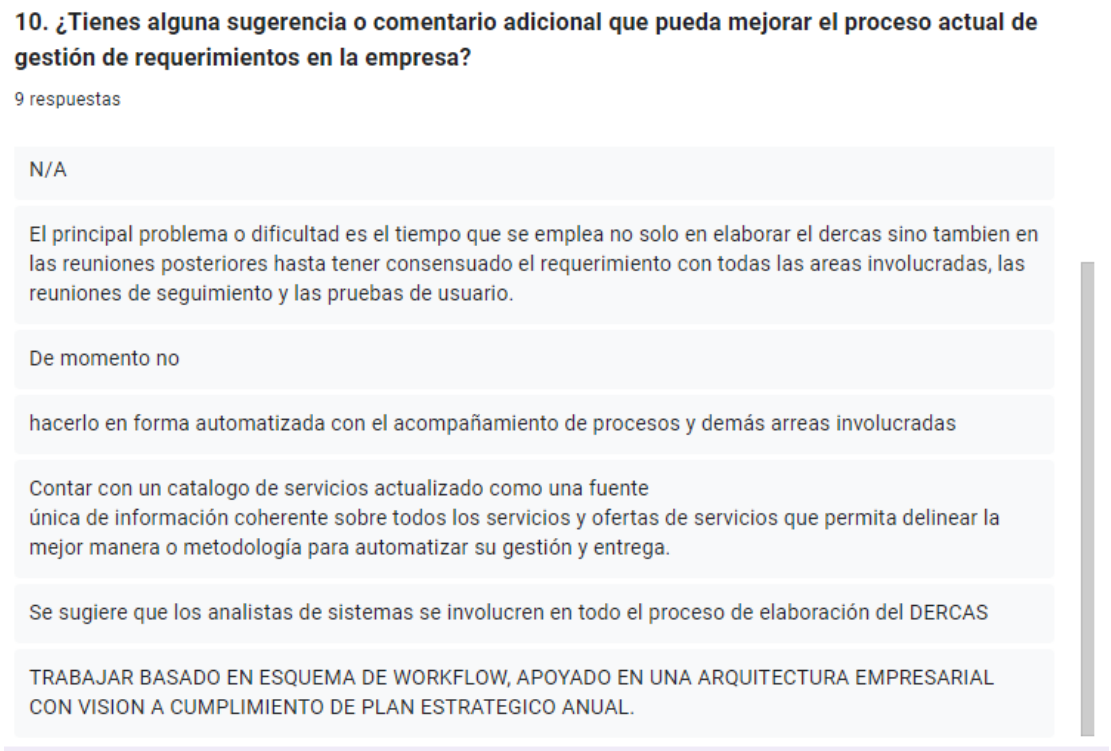


Figura 2.12: Resultado de pregunta 10

Fuente: Autor

Análisis e interpretación: Se destaca la necesidad de registrar todas las actividades y resultados de todas las áreas involucradas.

Se sugiere trabajar de forma automatizada con herramientas como Jira o Miro.

La participación de los analistas de sistemas en todo el proceso es recomendada

3.4. LEVANTAMIENTO DE INFORMACIÓN DEL PROCESO ACTUAL

En esta etapa fundamental, se detalla el proceso empleado para recopilar información esencial sobre la gestión actual de elaboración de requerimientos por parte de los usuarios encargados.

Para visualizar el estado actual del proceso, se implementó el modelo "As-Is". Este modelo permitió identificar procesos que se podrán mejorar, comprendiendo cómo se desarrollan los procesos.

La elaboración de una encuesta estratégica con preguntas claves se diseñó para obtener resultados de los usuarios encargados de elaborar requerimientos de proyectos de desarrollo de software. Esta encuesta se presentó como una herramienta crucial para comprender las percepciones, dificultades y sugerencias de quienes participan en la elaboración de requerimientos. Adicionalmente, se llevó a cabo una reunión con los usuarios responsables de proyectos para exponer la problemática existente y la propuesta de automatización. Esto permitió alinear expectativas, presentar la solución propuesta y recoger retroalimentación directa.

Esta fase del proceso sienta las bases para una comprensión profunda de los desafíos actuales, permitiendo una transición eficaz hacia la propuesta de diseño de la Plataforma Web Colaborativa.

3.5. DEFINICIÓN DEL MODELO AS-IS

En esta sección, se detalla exhaustivamente el proceso actual de levantamiento de requerimientos, llevado a cabo en colaboración entre el equipo de desarrollo de software y los usuarios responsables de la elaboración del DERCAS (Documento de Especificación de Requerimientos por el Cliente). La representación de este proceso incluirá aspectos clave para comprender su dinámica y estructura. A continuación, se describen las secciones que abarcará esta representación:

3.5.1. IDENTIFICACIÓN DE ACTORES

Destacaremos los roles esenciales en el proceso, identificando quiénes son los actores principales que participan en el levantamiento de requerimientos.

Tabla 2: Identificación de actores

Fuente: Autor

Área	Rol
Desarrollo de software	Coordinador de proyectos.
	Líder de proyectos.
	Analista.
	Ingeniero de software.

• Comercial • Financiero • Producto	Jefaturas de área.

3.5.2. RELACIÓN DE ACTORES CON EL PROCESO

Analizaremos la interacción y la relación entre los diferentes actores, delineando cómo se comunican y colaboran a lo largo del proceso de levantamiento de requerimientos.

Tabla 3: Actores y su relación con el proceso

Fuente: Autor

Actor	Descripción	Interés en el proceso	Responsabilidades
Jefaturas de áreas	Responsables de la toma de decisiones a nivel departamental	Alto	Contribuir con la creación y revisión del dercas
Coordinador de proyectos	Coordina la asignación de dercas a líderes de proyectos.	Alto	Coordinar la asignación de dercas a líderes de proyectos
Líder de proyectos	Supervisa el proceso de elaboración del dercas	Alto	Supervisar y liderar el proceso de elaboración del dercas y colaborar en la asignación de tareas y revisión de documentos
Analista	Analiza y comprende los requisitos del dercas, y se comunica con el	Alto	Analizar y comprender los requisitos del dercas

	usuario para clarificaciones		Colaborar en la creación de prototipos y la revisión de documentos.
Ingeniero de software	Responsable de entender y desarrollar los requisitos del dercas	Medio	Participar en la revisión de requisitos y el proceso de desarrollo del dercas

3.5.3. OBJETOS DE NEGOCIO

Identificamos los elementos que son parte del flujo de procesos para la etapa de levantamiento de información en un proyecto de desarrollo de software.

Tabla 4: Objetos de negocio

Fuente: Autor

Nombre	Tipo	Descripción	Roles
Dercas	BO	Documento en la cual se especifican los requerimientos de los usuarios para el desarrollo de un proyecto de software	Jefaturas de áreas
Prototipo	BO	Representa el diseño de los requerimientos del usuario	Analista

3.5.4. MATRIZ DE CASOS DE USO

Presentamos la matriz que ilustrará los diversos casos de uso asociados con el proceso de levantamiento de requerimientos, brindando una visión completa de las interacciones entre actores y objetos de negocio.

Tabla 5: Matriz de casos de uso

Fuente: Autor

ID	Actividad	Tipo	Descripción	Rol	Objeto Negocio
A01	Crear dercas	Manual	El usuario crea el documento	Jefaturas de áreas	Dercas
A02	Enviar dercas	Usuario	El usuario envía el documento al coord. de proyectos	Jefaturas de áreas	Dercas
A03	Evaluar la asignación del dercas	Manual	El coord. De proyectos, revisa el dercas, para saber a qué líder va a asignar	Coord. de proyectos	Dercas
A04	Asignar equipo	Usuario	Coord, de proyectos asigna el dercas al líder de proyectos	Coord. de proyectos	
A05	Revisar dercas	Manual	El líder de proyectos revisa el dercas, con el objetivo de explicar al ing. De software y analista	Líder de Proyectos	Dercas

A06	Consultar Disponibilidad	Usuario	Consulta disponibilidad para agendar una reunión	Líder de proyectos	
A07	Indicar Disponibilidad	Usuario	Indica disponibilidad	Analista	
A08	Indicar Disponibilidad	Usuario	Indica disponibilidad	Ingeniero de software	
A09	Agendar Reunión	Usuario	Envío de convocatoria al equipo	Líder de proyectos	
A10	Revisar dercas con el equipo	Manual	Revisión del dercas en conjunto con el equipo	Líder de proyectos , Analista, Ingeniero de software	Dercas
A11	Consultar Disponibilidad	Usuario	Consulta la disponibilidad al usuario	Analista	
A12	Indicar Disponibilidad	Usuario	El Usuario indica la disponibilidad	Jefaturas de áreas	
A13	Agendar Reunión	Usuario	Envía convocatoria	Analista	
A14	Revisar dercas	Manual	Se realiza la lectura del dercas para saber los requerimientos del usuario en conjunto con el equipo de desarrollo	Líder de proyectos , Analista, Ingeniero de software	Dercas
A15	Solicitar la actualización del dercas	Usuario	Cuando el dercas no está correcto en su totalidad se solicita la actualización	Analista	Dercas
A16	Modificar dercas	Manual	El usuario modifica el dercas	Jefaturas de áreas	Dercas

			en base a las reuniones		
A17	Enviar dercas actualizado	Usuario	El usuario envía por correo electrónico la actualización del dercas	Jefaturas de áreas	Dercas
A18	Creación o modificación de prototipos	Usuario	El analista en diseña o modifica prototipos	Analista	Prototipos
A19	Consultar Disponibilidad	Usuario	Consulta la disponibilidad al usuario	Analista	
A20	Indicar Disponibilidad	Usuario	El Usuario indica la disponibilidad	Jefaturas de áreas	
A21	Agendar Reunión	Usuario	Envía convocatoria	Analista	
A22	Presentar Prototipos	Usuario	El analista presenta el prototipo	Analista	Prototipos

3.5.5. REGISTRO DE EXCEPCIONES

Se detallan posibles situaciones excepcionales o desviaciones del proceso estándar, proporcionando una comprensión más completa de las contingencias que podrían surgir.

Tabla 6: Registro de excepciones

Fuente: Autor

ID	Excepción	Actividad	Descripción	Acciones Correctivas	Objeto Negocio
E1	El dercas no está bien redactado	A01	La redacción deficiente del dercas dificulta el análisis del proyecto para el equipo de desarrollo.	Proporcionar una guía detallada para la elaboración del dercas.	Dercas
E2	El usuario no informó a las áreas afectadas	A02	El usuario en la creación del dercas no involucró a las áreas afectadas en sus requerimientos	En caso de que los requerimientos afecten a otras áreas, deberá ser obligatorio la notificación a las otras áreas	Dercas
E3	No hay disponibilidad por parte del usuario	A11, A12, A13, A19, A20, A21	Falta de disponibilidad por parte del usuario para consultas y reuniones.	Establecer fechas definidas para revisar el proyecto solicitado	

3.5.6. MODELO AS-IS DEL PROCESO DE LEVANTAMIENTO DE REQUERIMIENTOS

En el Anexo A, se encuentra detallado el Modelo As-Is, el cual brinda una visión integral y detallada del proceso actual de levantamiento de requerimientos en una empresa de telecomunicaciones. Este modelo se basa en la información recopilada a través de diversas fuentes, entre ellas, la Matriz de Casos de Uso. Al analizar la Matriz de Casos de Uso, se revela la complejidad y las interacciones específicas que caracterizan el proceso. Desde la creación del Documento de Especificación de Requerimientos por el Cliente (DERCAS) por parte de los usuarios hasta su revisión y asignación por parte de coordinadores y líderes de proyectos. Los distintos roles, como Jefaturas de Áreas, Coordinadores de Proyectos, Líderes de Proyectos, Ingenieros de Software y Analistas, desempeñan funciones cruciales a lo largo de este proceso.

La consulta de disponibilidad, la programación de reuniones y la revisión colaborativa del dercas con el equipo de desarrollo son aspectos destacados que subrayan la necesidad de coordinación y comunicación constante. Además, se identifican excepciones que pueden surgir, como problemas en la redacción del dercas o

falta de información sobre áreas afectadas, lo que destaca la importancia de abordar estos desafíos en la propuesta del Modelo To-Be.

Este análisis profundo, plasmado en el Modelo As-Is presente en el Anexo A, proporciona una comprensión detallada del estado actual del proceso. A partir de esta base, se desarrollará la propuesta del Modelo To-Be, que busca mejorar la eficiencia y la calidad del proceso de levantamiento de requerimientos en el contexto de la empresa de telecomunicaciones.

3.6. ANÁLISIS DE RESULTADOS DEL LEVANTAMIENTO DE INFORMACIÓN

El análisis de los resultados del levantamiento de información revela una serie de aspectos cruciales en el actual proceso de desarrollo de software en una empresa de telecomunicaciones.

Entre los problemas identificados se encuentra la dificultad de los usuarios para reunir a todos los involucrados en el proyecto, el desconocimiento de los elementos que deben incluirse en cada sección del dercas, así como la falta de definiciones claras en los procesos descritos. Adicionalmente, se destaca la problemática de la disponibilidad para llevar a cabo el proceso de elaboración del dercas.

Una conclusión unánime entre los encuestados es la necesidad urgente de automatizar el proceso de levantamiento de información.

Este consenso refleja una clara demanda de eficiencia y mejora en la gestión de requerimientos, subrayando la importancia de optimizar los procedimientos existentes.

En cuanto a las expectativas y beneficios esperados de la nueva plataforma propuesta, los encuestados han señalado la trazabilidad del tiempo de entrega de la documentación y la visibilidad de los requerimientos y su estado como aspectos críticos. Además, existe un interés destacado en un repositorio de ayuda para futuras entregas de documentación y el seguimiento de los departamentos participantes.

La diversidad de herramientas utilizadas, desde correos electrónicos hasta reuniones presenciales y herramientas en línea, destaca la complejidad del proceso actual. La mayoría de los encuestados invierte hasta 4 horas semanales, subrayando la importancia de optimizar el tiempo dedicado a la elaboración de dercas.

En relación con el agendamiento de reuniones con otros departamentos, se observa que puede llevar hasta 5 días, evidenciando la necesidad de un proceso más ágil y eficiente.

Las sugerencias adicionales proporcionadas incluyen la necesidad de registrar todas las actividades, trabajar con herramientas automatizadas, y la participación de analistas de sistemas en todo el proceso.

Estos valiosos aportes ofrecen una orientación clave para la definición de los requisitos de la Plataforma Web Colaborativa, asegurando una solución que aborde de manera integral las áreas de mejora identificadas.

CAPÍTULO 4

DISEÑO DE LA SOLUCIÓN

En esta etapa crucial del proyecto, se forja el camino hacia una transformación significativa en el proceso de gestión de requerimientos. La implementación de la Plataforma Web Colaborativa (PWC) marca el hito que redefine la eficiencia y la eficacia en la creación de los Documentos de Especificación de Requerimientos por el Cliente (DERCAS). A lo largo de este capítulo, se delinean los elementos fundamentales que compondrán el futuro de la gestión de requerimientos en una empresa de telecomunicaciones

4.1. DEFINICIÓN DEL MODELO TO-BE

En esta sección, nos aventuramos hacia la visión "To-Be", un horizonte donde las limitaciones actuales del proceso de gestión de requerimientos se transforman en oportunidades. Detallamos cómo la PWC abordará las barreras existentes, optimizando la colaboración.

4.1.1. IDENTIFICACIÓN DE ACTORES

Destacaremos los roles esenciales en el proceso, identificando quiénes son los actores principales que participan en el nuevo proceso de levantamiento de requerimientos.

Tabla 7: Identificación de actores en el modelo to-be

Fuente: Autor

Área	Rol
Plataforma Web Colaborativa	Herramienta tecnológica que facilita la gestión de los proyectos (nuevo)
Desarrollo de software	Coordinador de proyectos (se mantiene).
	Analista (se mantiene).
• Comercial • Financiero • Producto	Jefaturas de área (se mantiene).

4.1.2. RELACIÓN DE ACTORES CON EL PROCESO

Analizaremos la interacción y la relación entre los diferentes actores, delineando cómo se comunican y colaboran a lo largo del nuevo proceso de levantamiento de requerimientos.

Tabla 8: Actores y su relación en el modelo to-be

Fuente: Autor

Actor	Descripción	Interés en el proceso	Responsabilidades
Plataforma Web colaborativa	Herramienta tecnológica que facilita la gestión y seguimiento de los proyectos	Alto	Presentar de forma intuitiva los campos necesarios para guardar la información necesaria con el objetivo de crear el dercas
Jefaturas de áreas	Responsables de la toma de decisiones a nivel departamental	Alto	Contribuir con la creación y revisión del dercas en la plataforma web colaborativa
Coordinador de proyectos	Coordina la asignación de dercas a líderes de proyectos.	Medio	Coordinar la asignación de dercas a líderes de proyectos
Analista	Analiza y comprende los requisitos del dercas, y se comunica con el usuario para clarificaciones	Alto	Analizar y comprender los requisitos del dercas Colaborar en la creación de prototipos y la revisión de documentos.

4.1.3. OBJETOS DE NEGOCIO

Identificamos los elementos que son parte del flujo de procesos para la etapa de levantamiento de información en un proyecto de desarrollo de software.

Tabla 9: Objeto de negocio en el modelo to-be

Fuente: Autor

Nombre	Tipo	Descripción	Roles
Proyecto	BO	Representa el proyecto de desarrollo de software	Jefaturas de áreas
Reunión	BO	Interacción planificada entre miembros del equipo para revisar y acordar sobre aspectos del proyecto	Jefaturas de áreas, PWC, Analista
Dercas	BO	Documento en la cual se especifican los requerimientos de los usuarios para el desarrollo de un proyecto de software	Jefaturas de áreas,PWC
Prototipo	BO	Representa el diseño de los requerimientos del usuario	Analista

4.1.4. MATRIZ DE CASOS DE USO

Presentamos la matriz que ilustrará los diversos casos de uso asociados con el proceso de levantamiento de requerimientos, brindando una visión completa de las interacciones entre actores y objetos de negocio.

Tabla 10: Matriz de caso de uso en el modelo to-be

Tabla 11: Registro de excepciones en el modelo to-be

Fuente: Autor

ID	Excepción	Actividad	Descripción	Acciones Correctivas	Objeto Negocio
E1	Error de inicio de Sesión	A01	Error de inicio de sesión en la Plataforma Web Colaborativa	Verificar credenciales y asegurar conexión a Internet	
E2	Fallo de generación de Dercas	A06	Fallo en la generación del DERCAS	Revisar los datos de entrada y reiniciar el proceso de generación	Dercas
E3	Fallo en lógica de negocio para la asignación automática	A15	Incapacidad para identificar automáticamente áreas que requieren actualización en el DERCAS	Revisar la lógica de identificación y mejorar la notificación	Dercas
E4	Error en creación y	A20	Error en la creación/modi	Revisar los datos de	Dercas

	modificación de prototipos		ficación de prototipos	entrada y la lógica de diseño de prototipos	
E5	Fallo en la presentación de prototipo	A25	Problema durante la presentación del prototipo	Verificar el estado del prototipo y realizar pruebas previas a la presentación	Prototipo

4.1.5. MODELO TO-BE

En el Anexo B, se encuentra detallado el Modelo "to-be" propuesto implica la implementación de una Plataforma Web Colaborativa (PWC) para mejorar el proceso de desarrollo de software en una empresa de telecomunicaciones. Los usuarios, especialmente las Jefaturas de áreas, acceden a la PWC para crear nuevos proyectos, proporcionando detalles clave y requisitos iniciales. La PWC automatiza la generación del Documento de Especificación de Requerimientos por el Cliente (DERCAS) utilizando la información del proyecto.

La plataforma facilita la colaboración eficiente entre roles como Analistas, Jefaturas de áreas y Coordinadores de proyecto, al proporcionar herramientas para revisar, solicitar reuniones y actualizar la información del proyecto. Además, notifica

automáticamente sobre datos inconsistentes y áreas que requieren actualización en el DERCAS.

La PWC permite la creación y modificación de prototipos, con la posibilidad de solicitar reuniones específicas para revisar los prototipos con el equipo. Automatiza el agendamiento de reuniones, optimizando el tiempo dedicado a la colaboración. Ofrece validación instantánea de la información ingresada y asesoramiento sobre el uso de la plataforma.

Se establece un registro de excepciones para monitorear y abordar posibles problemas durante la ejecución de actividades, con acciones correctivas especificadas. Este modelo to-be busca optimizar la eficiencia y calidad del proceso de desarrollo de software, promoviendo la colaboración efectiva y proporcionando herramientas para abordar desafíos identificados en el modelo as-is.

4.2. DISEÑO ARQUITECTÓNICO

El diseño arquitectónico es el esqueleto de la PWC, define las bases fundamentales para sustentar su funcionamiento. En este proceso, se detallan los componentes principales, sus interacciones y la estructura global de la plataforma. La arquitectura propuesta, guiada por la metodología C4, no solo persigue la eficiencia y escalabilidad, sino que también prioriza la seguridad. Este enfoque garantiza la integridad de la información sensible a lo largo de cada fase del proceso de gestión de requerimientos.

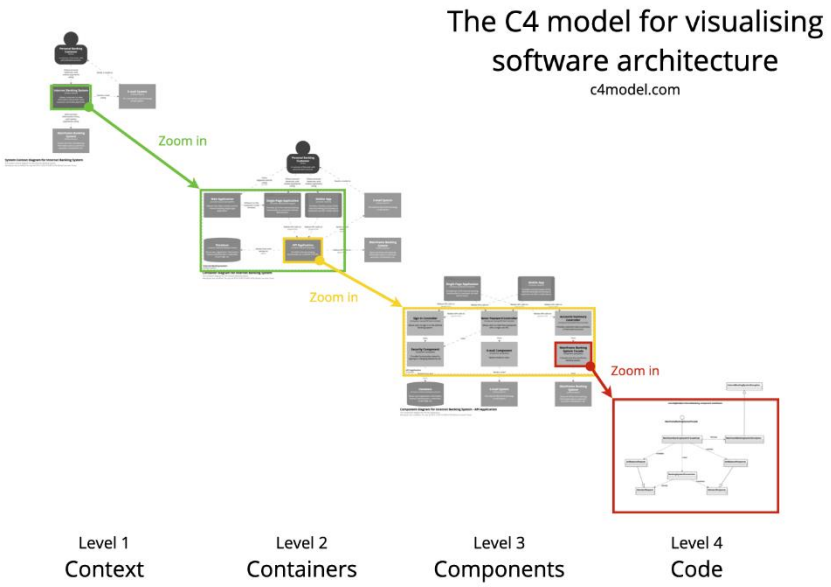


Figura 2.13: Modelo C4 para visualizar la arquitectura de software

Fuente: Modelo C4 [20]

A continuación, se presenta un resumen destacando los elementos clave abordados en los diferentes diagramas, brindando así una visión estructurada y detallada del diseño arquitectónico.

4.2.1. DIAGRAMA DE CONTEXTO

Este es el lienzo inicial que grafica las interacciones entre la PWC y su entorno. Identificamos relaciones claves con los sistemas de autenticación y de notificaciones, estableciendo el escenario para una arquitectura bien integrada y eficaz.

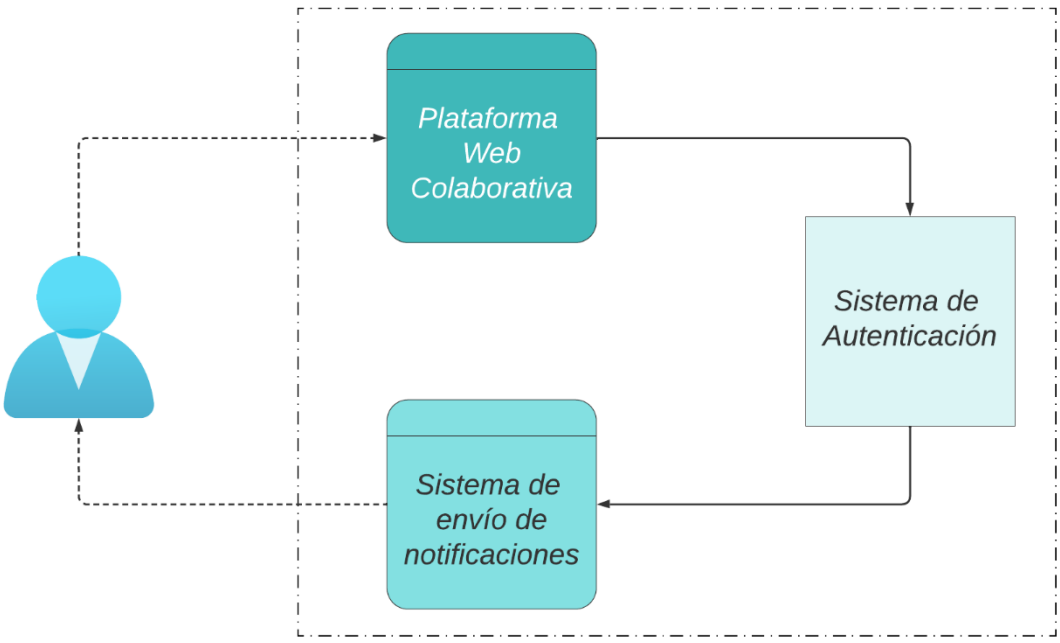


Figura 2.14: Diagrama de contexto

Fuente: Autor

4.2.2. DIAGRAMA DE CONTENEDOR

Avanzamos hacia el Diagrama de Contenedor, desglosando los módulos y componentes principales de la PWC. Cada contenedor representa una unidad lógica que contribuye a la funcionalidad general del sistema. Este nivel de detalle nos permite comprender cómo interactúan y se comunican los distintos componentes, sentando las bases para un diseño robusto y escalable.

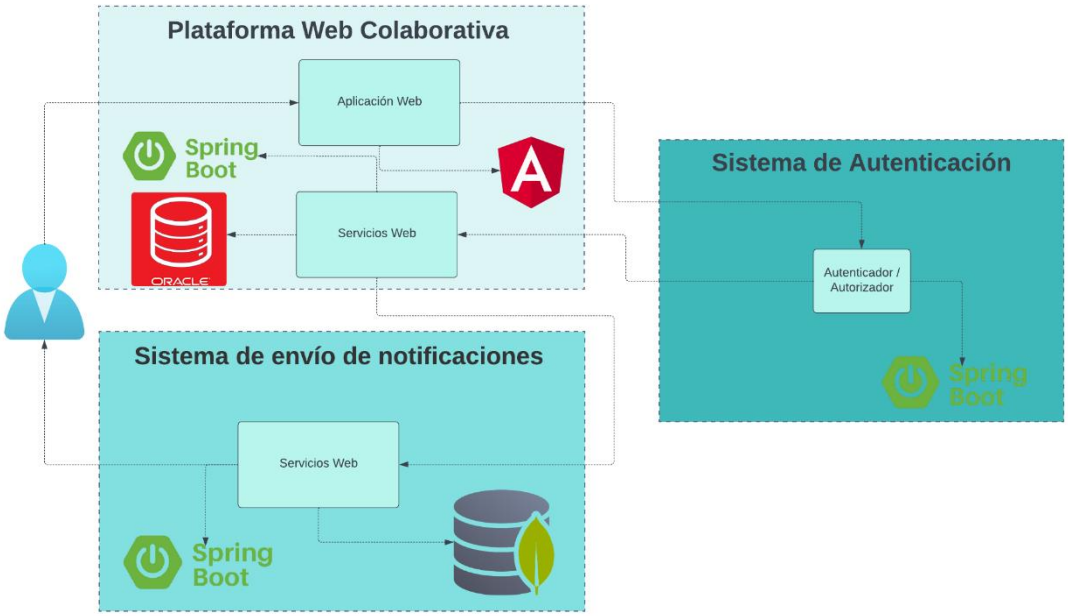


Figura 2.15: Diagrama de contenedor

Fuente: Autor

4.2.3. DIAGRAMA DE COMPONENTES

En este nivel, el Diagrama de Componente nos sumerge más profundamente. Aquí, cada elemento esencial de software se desglosa en sus partes constituyentes, ofreciendo una visión detallada de las relaciones y dependencias. Este nivel de abstracción es esencial para la implementación efectiva de la PWC, asegurando coherencia y modularidad.

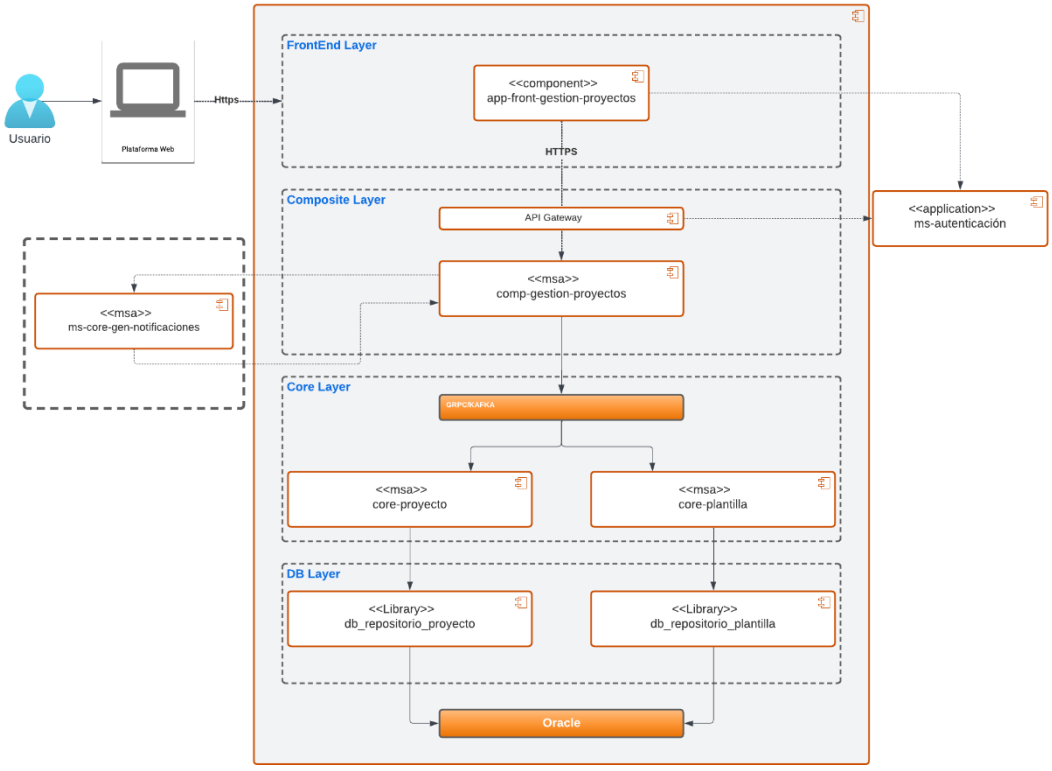


Figura 2.16: Diagrama de componentes

Fuente: Autor

4.2.4. DIAGRAMAS DE SECUENCIA

La dinámica de las interacciones entre los actores y la PWC cobra vida en los Diagramas de Secuencia. Estos diagramas proporcionan una perspectiva detallada de cómo los componentes de la plataforma facilitarán las diversas operaciones críticas. A través de ejemplos concretos, ilustramos cómo la PWC simplificará la comunicación entre los usuarios, reduciendo malentendidos y mejorando la eficiencia en la gestión de requerimientos.

- **Iniciar sesión:** En el diagrama de secuencia de iniciar sesión, se representa el flujo detallado de interacciones entre el usuario y la PWC, junto con el Sistema de Autenticación. Este proceso es fundamental para proporcionar un acceso seguro y personalizado a la plataforma.

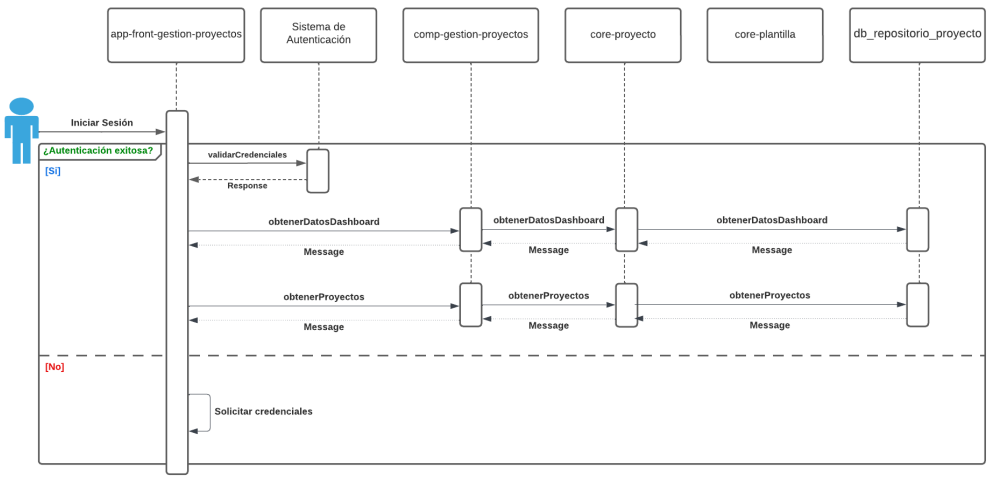


Figura 2.17: Diagrama de Secuencia: Iniciar Sesión

Fuente: Autor

- **Crear Proyecto:** en el diagrama de secuencia para crear un proyecto, se detallan las interacciones entre el usuario y la Plataforma Web Colaborativa (PWC) durante el proceso de creación de un nuevo proyecto.

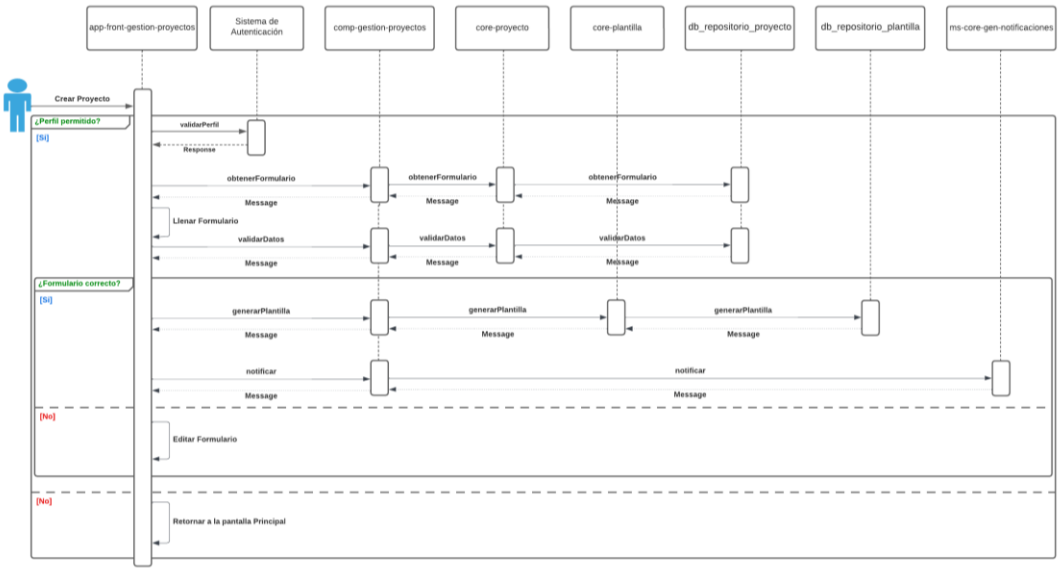


Figura 2.18: Diagrama de Secuencia: Crear Proyecto

Fuente: Autor

- **Agendar Reunión:** en el diagrama de secuencia para agendar una reunión, se describen las interacciones automáticas de la PWC para programar una reunión.

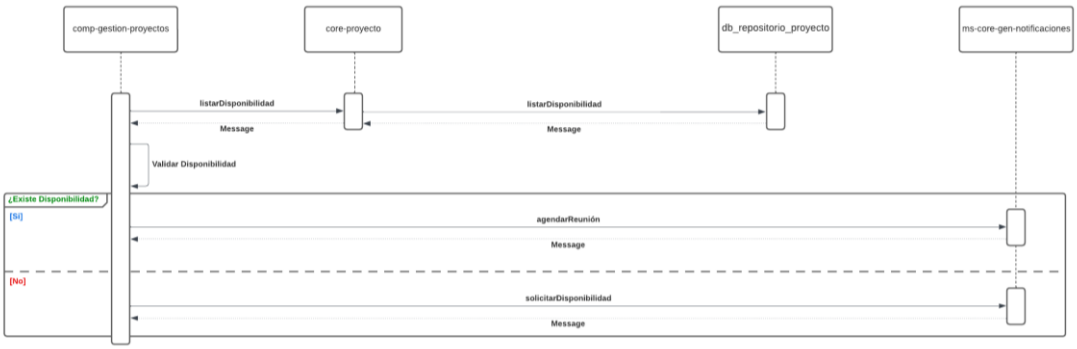


Figura 2.19: Diagrama de Secuencia: Agendar Reunión

Fuente: Autor

4.3. ELABORACIÓN DE PROTOTIPO

En este segmento, concretamos la visión de la Plataforma Web Colaborativa (PWC) a través de prototipos que encapsulan las funcionalidades de inicio de sesión, creación de proyectos y agendamiento de reuniones. Este paso esencial no solo acerca nuestra conceptualización a la realidad, sino que también establece un terreno sólido para recibir retroalimentación temprana y fundamentar futuras iteraciones y refinamientos:

- **Iniciar sesión:** Después de un inicio de sesión exitoso, los usuarios son llevados a un Dashboard personalizado. Este panel centralizado proporciona un resumen de sus proyectos.

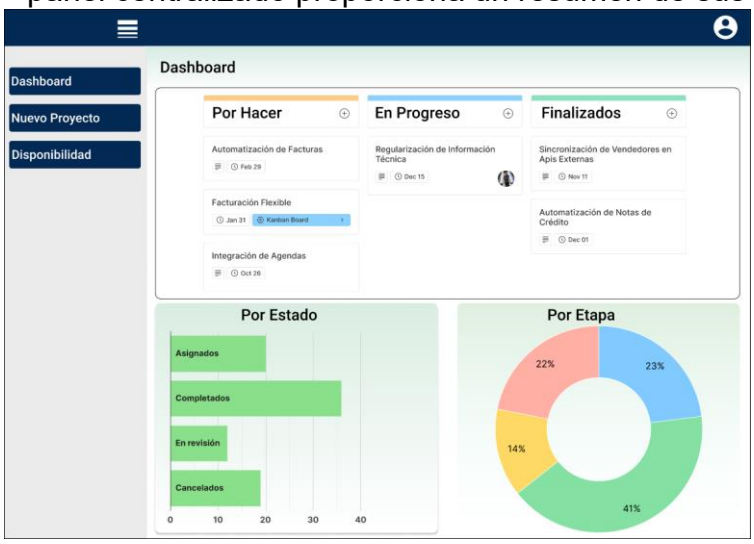


Figura 2.20: Prototipo: Dashboard

Fuente: Autor

- **Crear Proyecto:** Introducimos un formulario interactivo que guía al usuario en la introducción de detalles del proyecto.

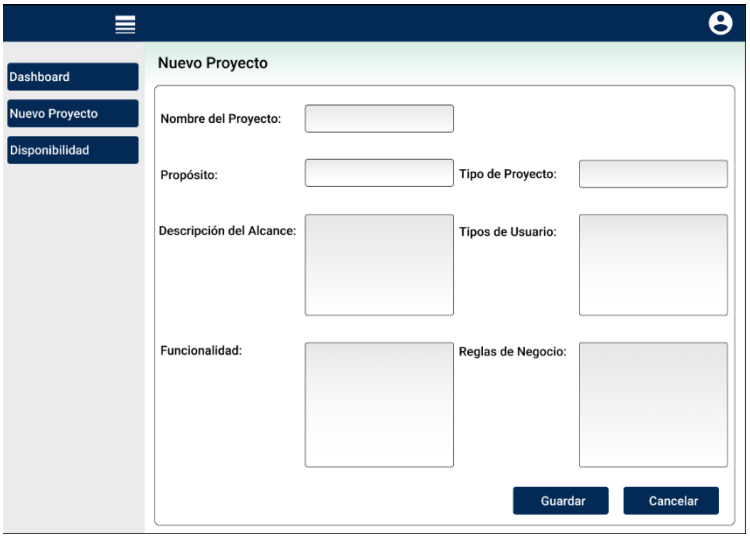


Figura 2.21: Prototipo: Crear Nuevo Proyecto

Fuente: Autor

- **Validar Información y asesoramiento:** La PWC verifica la integridad y consistencia de la información ingresada y proporciona asesoramiento en tiempo real para mejorar la calidad de los datos, así también envía notificaciones a los usuarios involucrados del proyecto.

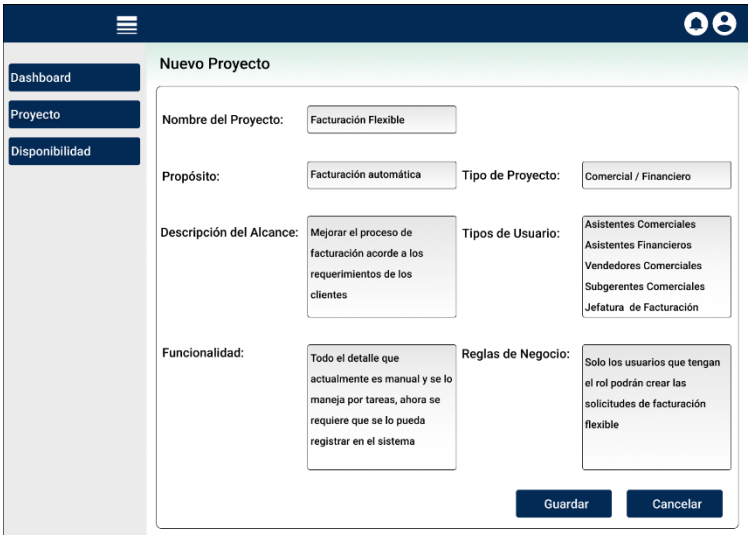


Figura 2.22: Prototipo: Ingresar Información del Nuevo Proyecto

Fuente: Autor

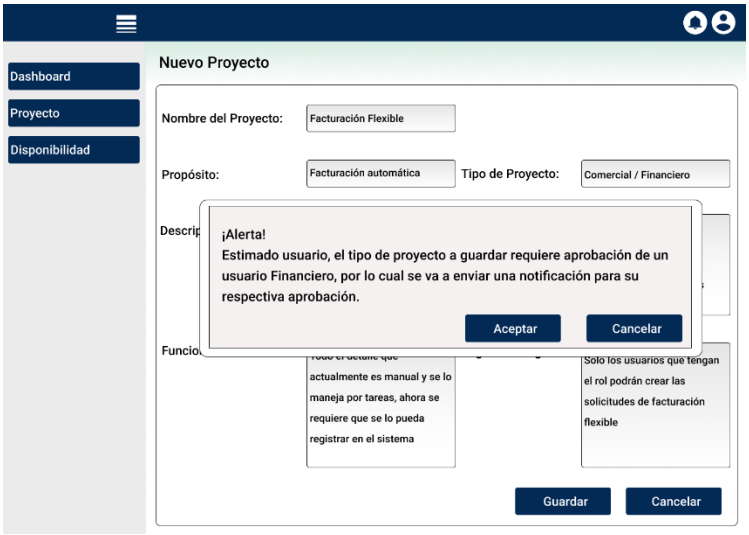


Figura 2.23: Prototipo: Validación de Usuarios

Fuente: Autor

- **Notificar los Datos Inconsistentes:** Si se detectan datos inconsistentes, la PWC notifica al usuario sobre los campos que necesitan corrección, como, por ejemplo:

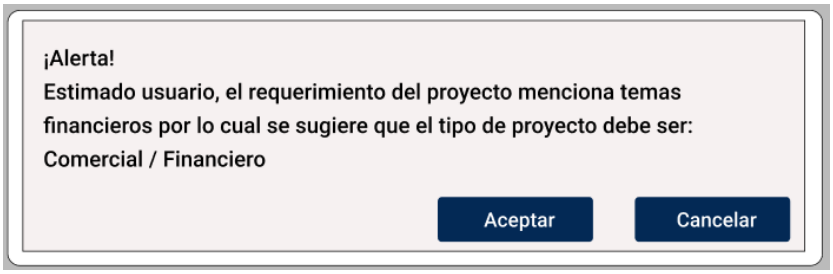


Figura 2.24: Prototipo: Validación del Tipo de Proyecto

Fuente: Autor

- **Editar Información del Proyecto:** Las jefaturas de áreas pueden editar la información del proyecto en cualquier momento para actualizar datos.

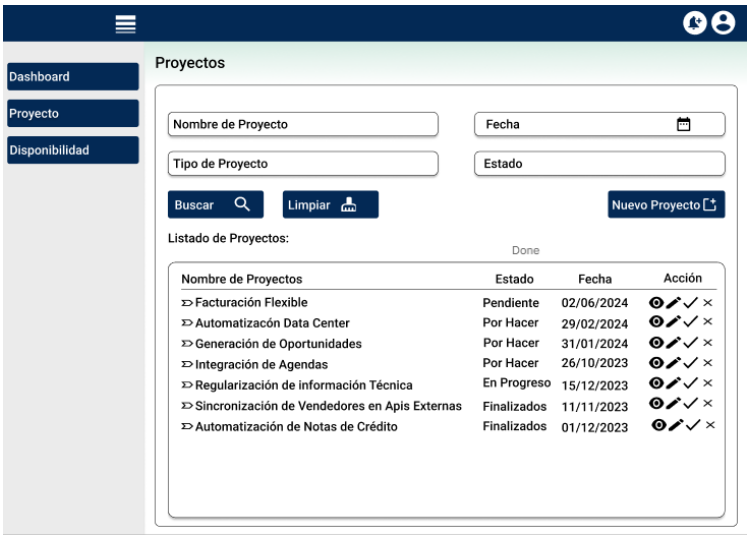


Figura 2.25: Prototipo: Visualización de Proyectos

Fuente: Autor

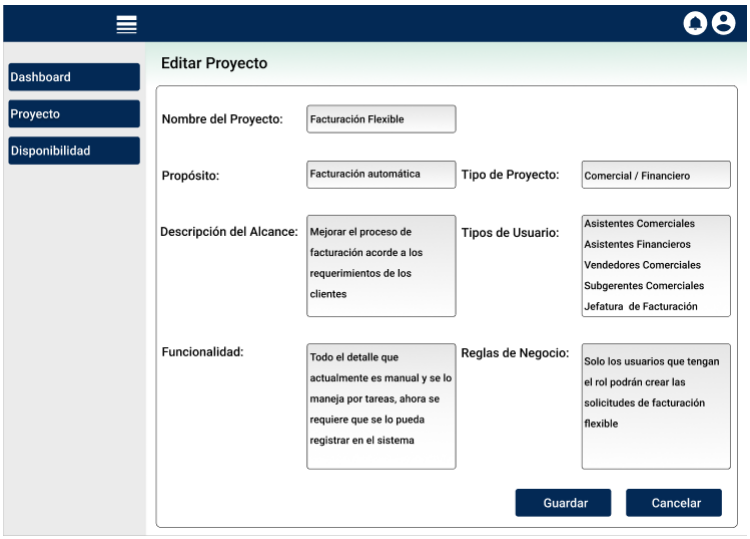


Figura 2.26: Prototipo: Editar de Proyecto

Fuente: Autor

- **Generación del Dercas:** Una vez que la información del proyecto es validada, la plataforma genera automáticamente el Dercas enviándolo por correo al coordinador de proyectos.

Estimado personal,

El presente correo es para informar la Aprobación de un nuevo Proyecto:

Nombre Proyecto:	Facturación Flexible
Tipo:	Comercial / Financiero
Creador por:	Alexandra Santana
Cargo:	Subgerente Comercial
Fecha de Creación:	24/05/2024

Detalle del Proyecto:

Proposito	Descripción del alcance	Tipos de Usuario	Funcionalidad
Facturación automática	Mejorar el proceso de facturación acorde a los requerimientos de los clientes	Asistentes Comerciales, Asistentes Financieros, Vendedores Comerciales, Subgerentes Comerciales, Jefatura de Facturación	Todo el detalle que actualmente es manual y se lo maneja por tareas, ahora se requiere que se lo pueda registrar en el sistema

Dercas

Figura 2.27: Prototipo: Visualización de Proyectos

Fuente: Autor

- **Evaluar la asignación de Dercas:** El Coordinador de proyecto revisa el Dercas generado para asignar al analista correspondiente

- **Asignar Analista:** El Coordinador de proyecto asigna al analista encargado de revisar y trabajar en el Dercas.

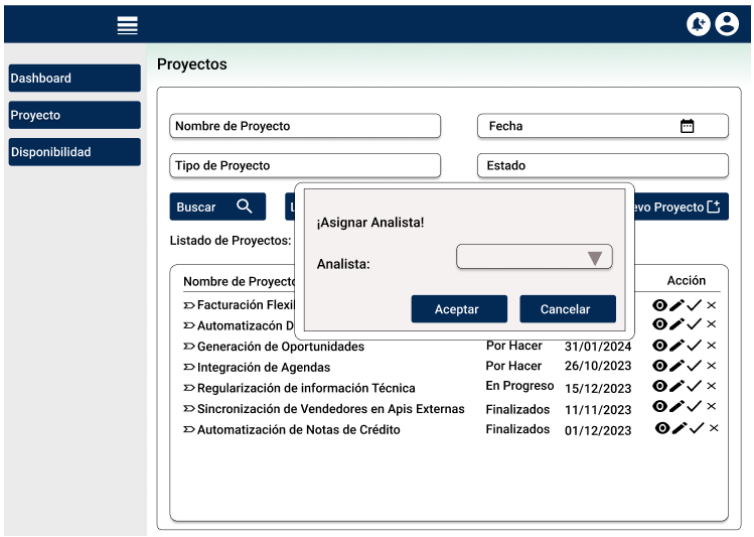


Figura 2.28: Prototipo: Asignar Analista

Fuente: Autor

Una vez revisado el documento, el Analista procede a solicitar una reunión con las Jefaturas de áreas para aclarar cualquier duda o requerimiento adicional. Para esto, la PWC cuenta con un componente de calendario interactivo que permite consultar la disponibilidad de todos los participantes involucrados. Este calendario facilita la selección de fechas y horarios adecuados, evitando conflictos de agenda.

- **Agendar Reunión:** El componente de calendario interactivo facilita la selección de fechas y horarios, mientras que la indicación de disponibilidad proporciona sugerencias intuitivas. Después de agendar una reunión, los usuarios son guiados a una pantalla de confirmación que destaca los detalles clave y próximos pasos.

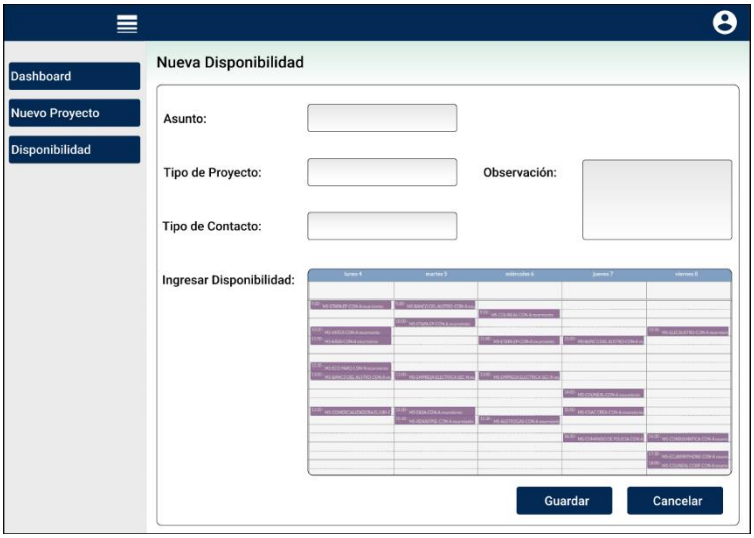


Figura 2.29: Prototipo: Ingresar Disponibilidad

Fuente Autor

Continuando con la gestión del Analista se procede a solicitar una reunión con las Jefaturas de áreas para aclarar cualquier duda o requerimiento adicional. Las Jefaturas de áreas indican su disponibilidad a través del mismo calendario interactivo mencionado anteriormente. Durante la reunión agendada, el Analista y las Jefaturas de áreas se reúnen para revisar el

Dercas en conjunto, resolviendo cualquier inconsistencia o falta de información. Si se identifican escenarios no descritos o reglas de negocios faltantes, el Analista solicita la actualización del Dercas, y las Jefaturas de áreas proceden a modificar el documento según las observaciones y sugerencias realizadas. La PWC valida nuevamente la información actualizada, ofreciendo asesoramiento en tiempo real para garantizar la calidad y consistencia de los datos. Si aún existen inconsistencias, la plataforma notifica a los usuarios para que realicen las correcciones pertinentes.

Una vez el Dercas es actualizado y validado correctamente, la plataforma notifica al Analista que puede proceder con la revisión final. En esta etapa, el Analista realiza cualquier ajuste necesario y, si todo está en orden, inicia la creación o modificación de prototipos basados en los requerimientos detallados en el Dercas Para presentar estos prototipos, el Analista solicita una nueva reunión con las Jefaturas de áreas. Utilizando nuevamente el calendario interactivo, la disponibilidad de los participantes se consulta y se agendan las reuniones necesarias.

Durante la reunión de presentación de prototipos, el Analista muestra los diseños a las Jefaturas de áreas, utilizando herramientas integradas de la PWC para anotaciones y comentarios en tiempo real. Esto permite a todos los participantes colaborar eficazmente, asegurando que los prototipos cumplan con las expectativas y requerimientos del proyecto.

Este flujo de trabajo, respaldado por la plataforma web colaborativa y su calendario interactivo, no solo optimiza la comunicación y coordinación entre los diferentes roles, sino que también mejora significativamente la calidad y precisión de los documentos y prototipos, asegurando un proceso de desarrollo de proyectos más ágil y eficiente.

CAPÍTULO 5

EVALUACIÓN Y ANÁLISIS DE RESULTADOS

Este capítulo desempeña un papel crítico al permitirnos refinar y perfeccionar la plataforma web colaborativa antes de su implementación. A través de un análisis riguroso y la aplicación de las mejores prácticas de diseño, nos aseguramos de que la solución propuesta no solo cumpla con las expectativas, sino que también sobresalga en su capacidad para potenciar eficazmente los procesos de gestión de proyectos en la empresa.

5.1. EVALUACIÓN DEL DISEÑO DE LA PLATAFORMA WEB COLABORATIVA

Abordaremos la evaluación detallada del diseño de la Plataforma Web Colaborativa (PWC), una herramienta concebida para transformar el proceso de documentación de requerimientos en el departamento de sistemas de una empresa de telecomunicaciones. La evaluación se llevará a cabo a los siguientes stakeholders:

- Jefatura de Sistemas valora positivamente el enfoque en la eficiencia del proceso de documentación y destaca la importancia de reducir tiempos para el éxito de los proyectos. Además, la atención a la seguridad y la alineación con estándares de calidad son aspectos bien recibidos. No obstante, se sugiere mejorar la claridad en aspectos técnicos específicos durante la presentación para lograr una comprensión más completa y detallada.
- Los usuarios valoran la interfaz intuitiva y fácil de usar, así como la implementación de herramientas analíticas y retroalimentación detallada. La necesidad de una capacitación inicial más detallada se identifica como punto de mejora.
- Arquitecto y analista de sistema destacan la robustez del diseño arquitectónico, especialmente la inclusión de sistemas clave como autenticación y notificaciones. Reconocen el potencial

impacto positivo en los resultados del desarrollo, pero sugieren explorar una integración más fluida con las herramientas existentes utilizadas por el equipo.

Este feedback detallado proveniente de cada tipo de usuario se considerará integralmente para ajustar y perfeccionar la Plataforma Web Colaborativa, asegurando que responda de manera óptima a las necesidades y expectativas de todos los involucrados en el proceso de desarrollo de software.

5.2. MEJORAS Y SUGERENCIAS

Durante el proceso de presentación se identificaron diversas áreas de mejora y sugerencias valiosas que se detallan a continuación:

- **Claridad en Aspectos Técnicos:** Se propone desarrollar material complementario o sesiones adicionales para aclarar puntos técnicos específicos que generaron inquietud en la Jefatura de Sistemas. Esto garantizará una comprensión más completa y profunda del diseño y la implementación de la PWC.
- **Capacitación Detallada:** Para abordar la necesidad de una capacitación inicial más detallada identificada por los usuarios, se recomienda la elaboración de un programa de capacitación exhaustivo. Este programa deberá cubrir no solo la

funcionalidad básica de la plataforma, sino también las características avanzadas y herramientas analíticas disponibles.

- **Integración Fluida:** Respondiendo a la sugerencia del arquitecto y analista de Sistemas, se explorará la implementación de una integración más fluida con las herramientas existentes del equipo de desarrollo. Esto se abordará mediante un análisis detallado de las herramientas utilizadas actualmente y la adaptación de la PWC para garantizar una transición sin problemas.
- **Comunicación Transparente:** Se impulsará una comunicación más transparente y abierta durante todo el proceso de implementación. Esto incluirá actualizaciones regulares sobre el progreso del proyecto, así como espacios para la discusión y aclaración de dudas.
- **Optimización Continua:** Se establecerá un proceso de revisión continua que permita la optimización constante de la PWC. Esto incluirá la recopilación periódica de feedback, evaluación de métricas clave y ajustes según las necesidades cambiantes de los usuarios y del entorno.

Estas mejoras y sugerencias se integrarán de manera proactiva en la fase de implementación de la Plataforma Web Colaborativa.

5.3. ANÁLISIS DE RESULTADOS

Aunque la implementación completa de la Plataforma Web Colaborativa (PWC) aún no se ha realizado, es posible estimar los resultados esperados en base al diseño propuesto y a la retroalimentación recibida durante la fase de diseño. Se proyecta que la introducción de la PWC tendrá un impacto significativo en varios aspectos clave del proceso de desarrollo de software en la empresa de telecomunicaciones.

La eficiencia operativa mejorará sustancialmente gracias a la reducción en el tiempo dedicado a la elaboración de documentos de especificación de requerimientos. Se espera que la colaboración en tiempo real y las herramientas de análisis automatizado agilicen el proceso de captura y validación de requerimientos.

Además, se anticipa una mejora significativa en la calidad de la documentación de requerimientos. La PWC facilitará la revisión y edición colaborativa de documentos, lo que resultará en una reducción de errores y omisiones, así como en una mayor claridad en las definiciones de procesos.

La colaboración interdepartamental se verá favorecida, permitiendo una comunicación más fluida y una coordinación más efectiva entre los equipos técnicos y no técnicos. Esto reducirá los tiempos de espera

para la programación de reuniones y mejorará la integración entre departamentos.

Aunque se anticipan resultados positivos, se reconocen desafíos potenciales, como la necesidad de una capacitación detallada para algunos usuarios y la optimización continua de la integración con herramientas existentes. También se debe garantizar la seguridad y privacidad de los datos sensibles desde el diseño inicial de la plataforma.

En resumen, el diseño de la PWC tiene el potencial de ser un paso significativo hacia la mejora del proceso de desarrollo de software en la empresa de telecomunicaciones. Sin embargo, se enfatiza la importancia de mantener un enfoque de mejora continua y una evaluación detallada durante la implementación para asegurar que la plataforma alcance su máximo potencia.

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES:

1. La realización del presente trabajo ha sido un proceso enriquecedor y revelador, abordando de manera integral los desafíos identificados en el proceso actual de desarrollo de software en la empresa de telecomunicaciones. A lo largo de los distintos capítulos, se ha explorado y desarrollado un modelo TO-BE representado por la Plataforma Web Colaborativa (PWC), con el objetivo de superar las limitaciones presentes en el modelo AS-IS.
2. La implementación exitosa del modelo TO-BE con la PWC marca un hito significativo en la evolución del proceso de desarrollo de software. Este proyecto no solo resuelve problemas identificados, sino que

también establece un estándar para la innovación continua. La transformación lograda refuerza la idea de que la adaptación de tecnologías avanzadas y enfoques colaborativos no solo es esencial sino también imperativa para enfrentar los desafíos dinámicos de la industria del desarrollo de software.

RECOMENDACIONES:

1. **Continuidad en la Capacitación:** se recomienda implementar programas continuos de capacitación para garantizar que todos los usuarios, estén completamente familiarizados con las funcionalidades de la Plataforma Web Colaborativa (PWC). Esto asegurará un uso óptimo de la herramienta y maximizará sus beneficios.
2. **Monitoreo de Indicadores Clave:** se recomienda establecer un sistema de monitoreo continuo de indicadores clave de rendimiento, como el tiempo de implementación de proyectos y la calidad de la documentación de requerimientos. Estos datos proporcionarán una retroalimentación cuantitativa sobre la efectividad de la PWC a lo largo del tiempo y ayudarán en futuras mejoras.
3. **Feedback Regular y Canales Abiertos:** fomentar un sistema de feedback regular mediante canales abiertos de comunicación entre los usuarios y los responsables de la PWC es esencial. Esto permitirá identificar oportunidades de mejora, abordar desafíos emergentes y mantener la relevancia de la plataforma en un entorno en constante evolución.

BIBLIOGRAFÍA

[1] M. A. Chaves, «La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software», *InterSedes: Revista de las Sedes Regionales*, vol. 6, n.º 10, pp. 1-13, 2005.

[2] C. T. R. Barajas, «Impacto de los requerimientos en la calidad de software», *Tecnología Investigación y Academia*, vol. 5, n.º 2, pp. 161-173, 2017.

[3] E. P. Gomez Ruiz, «Implementación de un sistema de información bajo plataforma web para la gestión y control documental de la empresa corporación Jujedu EIRL–Talara; 2017», 2017.

[4] «IEEE Standard Glossary of Software Engineering Terminology», *IEEE Std 610.12-1990*, pp. 1-84, 1990, doi: 10.1109/IEEESTD.1990.101064.

[5] G. Parker, M. Van Alstyne, y S. P. Choudary, *Platform revolution: how networked markets are transforming the economy - and how to make them work for you*, First published as a Norton paperback 2017. New York London: W. W. Norton & Company, 2017.

[6] K. Southwell *et al.*, «Requirements definition and design», *The STARTS Guide,* vol. 1, pp. 177-313, 1987.

[7] *Ingeniería del software: un enfoque práctico*, Séptima edición. México; Bogotá; Buenos Aires; Caracas; Guatemala; Madrid: McGraw-Hill Education, 2013. [En línea]. Disponible en: http://cotana.informatica.edu.bo/downloads/ld-Ingenieria.de.software.enfoque.practico.7ed.Pressman.PDF

[8] I. Jacobson, G. Booch, y J. Rumbaugh, «El Proceso Unificado de Desarrollo de Software, 2000», *Adisson-Wesley. Prólogo, Capítulos*, pp. 1-5, 2000.

[9] J. Fulton, «Web Architecture 101», [En línea]. Disponible en: https://medium.com/storyblocks-engineering/web-architecture-101-a3224e126947

[10] N. Nurmuliani, D. Zowghi, y S. Powell, «Analysis of requirements volatility during software development life cycle», en *2004 Australian Software Engineering Conference. Proceedings.*, IEEE, 2004, pp. 28-37.

[11] M. A. Akbar *et al.*, «Improving the Quality of Software Development Process by Introducing a New Methodology–AZ-Model», *IEEE Access*, vol. 6, pp. 4811-4823, 2018, doi: 10.1109/ACCESS.2017.2787981.

[12] M. W. Bhatti, F. Hayat, N. Ehsan, A. Ishaque, S. Ahmed, y E. Mirza, «A methodology to manage the changing requirements of a software project», en *2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM)*, 2010, pp. 319-322. doi: 10.1109/CISIM.2010.5643642.

[13] A. A. Khan, S. Basri, P. Dominic, y others, «A process model for requirements change management in collocated software development»,

en *2012 IEEE Symposium On E-Learning, E-Management And E-Services*, IEEE, 2012, pp. 1-6.

[14] D. C. Ince, *Introduction to software quality assurance and its implementation*. McGraw-Hill, Inc., 1995.

[15] D. Damian, «Stakeholders in global requirements engineering: Lessons learned from practice», *IEEE software*, vol. 24, n.º 2, pp. 21-27, 2007.

[16] J. Pradel Miquel, J. A. Raya Martos, B. Campderrich Falgueras, X. Sánchez Porras, C. Fuertes Royo, y R. Albiñana Bertomeu, «Ingeniería del software, febrero 2013», 2013.

[17] H. Cubero, «Procesos de ingeniería de software», 2020.

[18] F. Berzal, «El ciclo de vida de un sistema de información», [En línea]. Disponible en: http://flanagan.ugr.es/docencia/2005-2006/2/apuntes/ciclovida.pdf

[19] W. R. Johnson Freeman, M. E. Phillips, y K. K. Phillips, «Administrar un flujo de trabajo de repositorio institucional con GitLab y un sistema de depósito basado en carpetas». [En línea]. Disponible en: https://journal.code4lib.org/articles/15650?utm_source=rss&utm_medium=rss&utm_campaign=managing-an-institutional-repository-workflow-with-gitlab-and-a-folder-based-deposit-system

[20] S. Brown, «El modelo C4 para visualizar la arquitectura de software», [En línea]. Disponible en: https://c4model.com/

GLOSARIO

- **Plataforma Web Colaborativa (PWC):** Sistema en línea que permite la colaboración en tiempo real entre usuarios, facilitando la creación, revisión y gestión de documentos y proyectos.
- **Documento de Especificación de Requerimientos por el Cliente (DERCAS):** Documento que describe los requisitos funcionales y no funcionales de un sistema de software, elaborado por el sponsor principal o analista de negocio.
- **Microservicios:** Arquitectura de software que consiste en dividir una aplicación en pequeños servicios independientes y modularizados, que se pueden desarrollar, desplegar y escalar de forma independiente.
- **Amazon Web Services (AWS):** Plataforma de servicios en la nube ofrecida por Amazon, que proporciona una amplia gama de servicios de computación, almacenamiento, bases de datos, entre otros.
- **Dashboard:** Panel de control visual que muestra información clave de manera resumida y fácilmente comprensible, permitiendo a los usuarios tomar decisiones informadas.
- **Prototipo:** Versión inicial de un producto o sistema, generalmente desarrollada con el propósito de obtener retroalimentación temprana y validar conceptos antes de la implementación final.

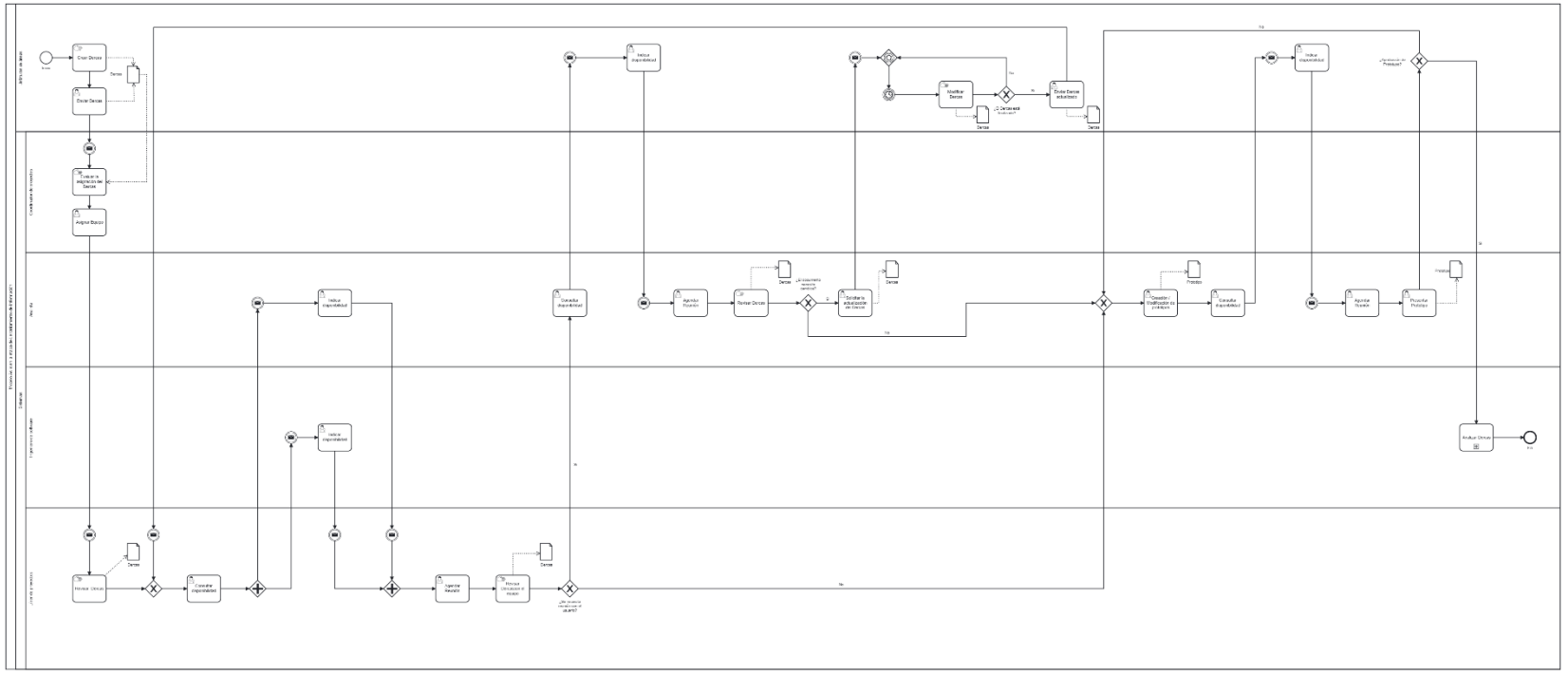
- **Diagrama de Secuencia:** Diagrama que representa la interacción entre objetos en un sistema a través del tiempo, mostrando la secuencia de mensajes intercambiados entre ellos.
- **Diagrama de Contexto:** Diagrama que muestra las interacciones entre un sistema y sus entornos externos, identificando los actores y los límites del sistema.
- **Diagrama de Contenedor:** Diagrama que representa los componentes de un sistema y sus relaciones, ayudando a visualizar la estructura y arquitectura del sistema.
- **Diagrama de Componentes:** Diagrama que muestra los componentes de un sistema y las dependencias entre ellos, proporcionando una vista detallada de la estructura interna del sistema.
- **Sistema de Autenticación:** Componente que gestiona el proceso de verificación de la identidad de un usuario, permitiendo el acceso seguro a un sistema o plataforma.
- **Sistema de Notificaciones:** Componente que se encarga de enviar notificaciones o alertas a los usuarios sobre eventos importantes o cambios en el sistema.
- **Roles:** Funciones o responsabilidades asignadas a usuarios dentro de un sistema o proyecto, determinando sus permisos y actividades autorizadas.

- **Gestión de Requerimientos:** Proceso que involucra la captura, análisis, documentación y gestión de los requisitos de un sistema de software, asegurando que cumplan con las necesidades y expectativas del cliente.
- **Iteración:** Proceso de repetir una serie de pasos o actividades en un ciclo continuo, con el fin de mejorar un producto o sistema mediante la retroalimentación y la corrección de errores.

ANEXOS

Anexo A

Modelo As-Is



Anexo B

Modelo To-Be