TEC | Tecnológico de Costa Rica

Área Académica de Administración de Tecnologías de Información

Propuesta de mejora del proceso de aseguramiento de la calidad del *software* para un Grupo Financiero

Trabajo Final de Graduación para optar al grado de Licenciatura en Administración de Tecnología de Información

Elaborado por: Fabián Navarro Martínez

Prof. Tutor: Máster. Agustín Francesa Alfaro

Cartago, Costa Rica

II Semestre

Noviembre, 2023

Hoja de Aprobación

Área Académica de Administración de Tecnologías de Información

GRADO ACADÉMICO: LICENCIATURA

Los miembros del Tribunal Examinador del Área Académica de Administración de Tecnologías de Información, recomendamos el siguiente Trabajo Final de Graduación del estudiante Fabián Antonio Navarro Martínez sea aceptado como requisito parcial para optar al grado académico de Licenciatura en Administración de Tecnologías de Información.

JOSE AGUSTIN FRANCESA ALFARO (FIRMA)
2023.11.28 13:31:39-06'00'

Agustín Francesa Alfaro

Profesor Tutor

TEC Tecnológico de Costa Rica

Firmado digitalmente por GABRIEL UMAÑA CENTENO (FIRMA)
Fecha: 2023.11.28 13:54:52 -06'00'

Gabriel Umaña Centeno

Lector Académico

Firmado digitalmente por VICTOR GERARDO VARGAS RAMIREZ (FIRMA)
Fecha: 2023.11.29 11:28:49 -06'00'

Víctor Vargas

Lector de Industria

TEC | Tecnológico de Costa Rica

Firmado digitalmente por YARIMA TATIANA SANDOVAL SANCHEZ (FIRMA)
Fecha: 2023.11.28 15:05:41 -06'00'

Yarima Sandoval Sánchez

Coordinadora Trabajo Final de Graduación

Dedicatoria

Quiero dedicar este trabajo final de graduación a todas aquellas personas que han sido parte de este proceso directa e indirectamente. Primeramente agradecerle a Dios por haberme permitido llegar a este punto que alguna vez soñé cuando ingresé a la carrera y que finalmente estoy alcanzando.

Seguidamente dedicarle este proyecto al pilar más importante en mi vida el cual es mi familia, especialmente a mi padre, madre, hermanos y sobrinas. También a aquellos que ya no están en mi vida pero que estoy seguro siempre han estado presentes aunque no me de cuenta.

Finalmente a modo de agradecimiento, referirme a todos los amigos que han estado presentes a lo largo de la carrera con los cuales he pasado muy buenos momentos. También al profesor Agustín Francesa que me acompañó y guió a lo largo de este proceso con su conocimiento. Por último, a la organización que me brindó un espacio para desarrollar mi proyecto y ha sido un punto vital en mi actual formación y crecimiento.

Resumen

Navarro Martínez, Fabián (2023). Propuesta de mejora en el proceso de aseguramiento de la calidad del *software* para un Grupo Financiero. [Trabajo Final de Graduación, Tecnológico de Costa Rica].

La presente investigación tiene como objetivo principal proponer una mejora al proceso de aseguramiento de calidad del *software* para un Grupo Financiero. La principal problemática que se desea resolver es la deficiencia en el proceso que se realiza actualmente dentro del departamento, ya que dicho proceso es desarrollado de forma empírica sin estar estandarizado ni alineado con las buenas prácticas existentes para el desarrollo y aseguramiento de la calidad del *software*.

La investigación tiene un enfoque cualitativo y tiene como principales fuentes de recopilación de información técnicas esenciales como lo son las entrevistas, la observación y revisión documental.

El procedimiento metodológico se basa fundamentalmente en cinco fases principales las cuales son el análisis de la situación actual del proceso, que tiene como objetivo generar un diagrama *as-is* con todos sus elementos necesarios, seguidamente se presenta una fase para el rediseño del proceso, en esta fase se estudian algunas buenas prácticas para la generación del proceso deseado por la organización.

Las siguiente fase consiste en la selección de una herramienta que sirva de apoyo al proceso propuesto y que se adecue a las necesidades de la organización. La cuarta fase corresponde con la elaboración de un posible plan de implementación para el proceso propuesto, para ello se utiliza como guía lo establecido en el PMBOK con relación a los dominios de desempeño en la gestión de proyectos. Finalmente, se encuentra la fase cinco que tiene como propósito realizar un análisis de costo-beneficio para evaluar la viabilidad en términos financieros de una posible implementación del proceso propuesto en la organización.

Palabras clave: Aseguramiento de calidad del *software*, procesos de negocio, rediseño de procesos.

Abstract

Navarro Martínez, Fabián (2023). Proposal for improvement in the *software* quality assurance process for a Financial Group. [Final Graduation Project, Technological Institute of Costa Rica].

The main objective of this research is to propose an improvement to the *software* quality assurance process for a Financial Group. The main problem that we want to solve is the deficiency in the process that is currently carried out within the department, since said process is developed empirically without being standardized or aligned with existing good practices for the development and quality assurance of *software*.

The research has a qualitative approach and its main sources of information collection are essential techniques such as interviews, observation and documentary review.

The methodological procedure is fundamentally based on five main phases which are the analysis of the current situation of the process, which aims to generate an as-is diagram with all its necessary elements, followed by a phase for the redesign of the process, in this phase, some good practices are studied for generating the process desired by the organization.

The next phase consists of selecting a tool that supports the proposed process and that adapts to the needs of the organization. The fourth phase corresponds to the development of a possible implementation plan for the proposed process, for which the provisions established in the PMBOK in relation to the performance domains in project management are used as a guide. Finally, there is phase five, which aims to carry out a cost-benefit analysis to evaluate the viability in financial terms of a possible implementation of the proposed process in the organization.

Key words: *Software* quality assurance, business processes, process redesign.

Tabla de Contenidos

Índice de Figuras

Índice de Tablas

Nota Aclaratoria

Género[1]:

La actual tendencia al desdoblamiento indiscriminado del sustantivo en su forma masculina y femenina va contra el principio de economía del lenguaje y se funda en razones extralingüísticas. Por tanto, deben evitarse estas repeticiones, que generan dificultades sintácticas y de concordancia, que complican innecesariamente la redacción y lectura de los textos.

Este documento se redacta de acuerdo con las disposiciones actuales de la Real Academia Española con relación al uso del "género inclusivo". Al mismo tiempo se aclara que estamos a favor de la igualdad de derechos entre los géneros

[1] Recuperado de: http://www.rae.es/consultas/los-ciudadanos-y-las-ciudadanas-los-ninos-y-las-ninas

1. Capítulo 1: Introducción

1.1. Descripción General

El objetivo del presente documento es brindar una solución a una problemática relacionada con la deficiencia en el proceso de aseguramiento de la calidad del *software* en la organización estudiada.

Actualmente no se sigue un proceso estándar para gestionar este tema de la calidad, por ende, este proceso se realiza de una manera subjetiva, es decir cada desarrollador realiza sus pruebas de manera distinta y en algunos casos no se cubren escenarios que tienen afectaciones en el ambiente de producción.

Debido a lo anterior, se desea realizar un análisis del proceso actual para detectar sus deficiencias, con el fin de efectuar mejoras a través de un estándar que brinde una guía con buenas prácticas, con respecto al aseguramiento de la calidad del *software*.

A lo largo del documento se presentan diversos capítulos que permitirán entender la propuesta que busca solucionar la problemática identificada.

1.2. Antecedentes

A continuación, se presentan una serie de secciones que brindan un poco de contexto sobre la organización en la que se realiza el Trabajo Final de Graduación e información sobre algunos proyectos similares que se han desarrollado tanto a nivel interno como externo.

1.2.1. Descripción de la organización

El desarrollo del Trabajo Final de Graduación se efectuará en un Grupo Financiero que ofrece productos y servicios en la región de Centroamérica, específicamente en Costa Rica, Nicaragua, El Salvador, Honduras, Guatemala y Panamá, además de otros países del Caribe, como Gran Caimán y Bahamas (Grupo Financiero, 2023).

Según se indica en el sitio web del Grupo Financiero (2023), la organización cuenta con alrededor de setenta años de experiencia y ofrece sus productos y servicios a un total cercano de 4.2 millones de clientes en toda la región, esto se logra gracias a la participación de cerca de 19 000 colaboradores originarios de los diversos países antes mencionados y de algunos países de Norteamérica, principalmente.

El inicio de sus operaciones se dio el día 5 de julio de 1952 en Nicaragua, en donde se fundó el primer banco; en la década de los años 70 se convirtió en la institución pionera del negocio de tarjetas de crédito en toda la región, de esta manera, fue ampliando gradualmente sus operaciones a lo largo de los demás países anteriormente mencionados (Grupo Financiero, 2023).

El segundo país en donde ofreció sus productos fue en Costa Rica, en donde el Grupo Financiero a mediados de los años 80 realizó la adquisición de un banco de índole privado,

considerado como uno de los más importantes del país. Gradualmente, el Grupo Financiero extendió sus operaciones a cada uno de los países y fue en la década de los años 90 en que se convirtió en el primer ente con presencia en toda Centroamérica (Grupo Financiero, 2023).

Como siguiente hito en la historia del Grupo Financiero, en el año 2004 se encuentra el inicio de sus operaciones en el ámbito de tarjetas de crédito en la región de Norteamérica, específicamente en México. A través de una alianza estratégica con otra compañía en el año 2005 se da la adquisición del 49.99% del capital del Grupo Financiero por parte de esta última compañía. En el mismo periodo, se adquirió un banco de índole privada en Honduras por parte del Grupo Financiero.

Siguiendo la línea cronológica, para el año 2007, el Grupo Financiero continuó con su estrategia de expansión en donde se adquirió una corporación financiera en Costa Rica, además de incrementar su patrimonio y operaciones al adquirir un programa de promoción que apoyaba a la pequeña y microempresa en El Salvador (Grupo Financiero, 2023).

Dos años más tarde, la compañía que ya había adquirido el 49,99% del capital del Grupo Financiero decidió incrementar su participación en la alianza establecida, entonces pasó de un 49,99% a un 75% de participación accionaria, de esta forma se convirtió en el principal accionista. No obstante, esta compañía decidió dejar de lado sus operaciones en el sector de la banca privada, esto se debió a un cambio de estrategia a nivel mundial que adoptó la compañía para centrar sus actividades en el sector industrial (Grupo Financiero, 2023).

Debido a lo anterior, en julio del 2010, un conglomerado colombiano conformado por la participación de varias entidades financieras decidió llevar a cabo la adquisición del 100% de las acciones del Grupo Financiero a través de un proceso de compraventa de acciones (Grupo Financiero, 2023).

A pesar del cambio de control accionario, la estrategia de negocios y la identidad del Grupo Financiero se mantienen y a raíz de la adquisición ha sido posible ofrecer productos con un valor agregado a los clientes, compartir experiencias, aprovechar las sinergias y las mejores prácticas de ambas partes y, sobre todo, compartir la visión de negocios, lo que hace que el Grupo Financiero siga siendo hoy una organización caracterizada por el mejoramiento continuo, la pasión por la excelencia, la innovación y la creatividad. (Grupo Financiero, 2023).

En la actualidad, el Grupo Financiero, atento a los cambios en el entorno y la evolución permanente, ha desarrollado una estrategia de transformación a través de la banca digital, que en la actualidad es utilizada por más de 1.9 millones de clientes (Grupo Financiero, 2023).

De acuerdo con el Grupo Financiero (2023), su estrategia se basa en tres aspectos:

- Valor económico: basado en maximizar el valor económico generado, hacer banca radicalmente transparente y ser líder en soluciones financieras simples, digitales y de triple valor.

- Valor social: relacionado con ser líderes en inversión social estratégica, hacer banca inclusiva y equitativa en género, promover el desarrollo y bienestar de sus colaboradores y también erradicar la pobreza.
- Valor ambiental: estrategia basada en impulsar el bienestar ambiental de la región, pensando en ser un banco carbono positivo.

A continuación se presentan la misión, visión y valores de Grupo Financiero:

Misión

La misión del Grupo Financiero es:

Facilitar con excelencia el intercambio y financiamiento de bienes y servicios, a través de sistemas de pago y soluciones financieras innovadoras y rentables que contribuyan a generar riqueza, a crear empleo y a promover el crecimiento económico sostenible y solidario de los mercados donde operamos. (Grupo Financiero, 2023).

Visión

La visión del Grupo Financiero es:

Ser la organización financiera preferida de todas las comunidades que servimos por nuestra conectividad con personas y empresas, por nuestra confiabilidad, espíritu innovador, solidez y claro liderazgo en los sistemas de pago de la Región. (Grupo Financiero, 2023).

Valores

Los valores del Grupo Financiero (2023) son:

- Integridad.
- Respeto.
- Excelencia.
- Responsabilidad.
- Innovación.

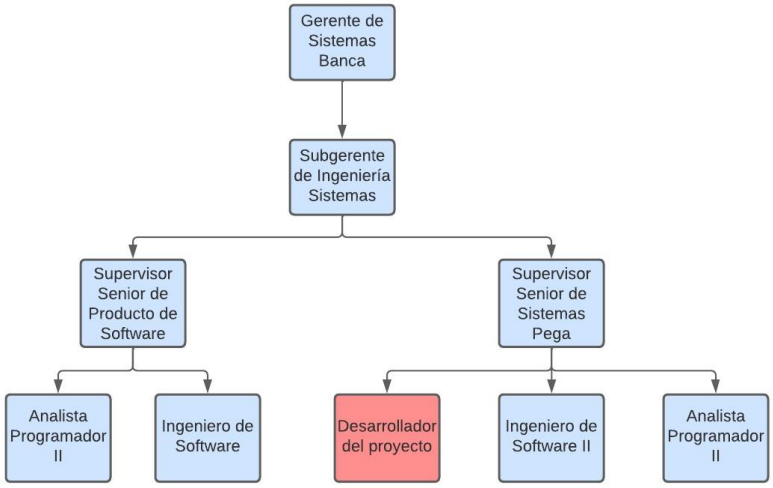
Equipo de trabajo

En la presente sección se describe el equipo que está involucrado en el desarrollo del proyecto y que tendrá una gran relevancia para los resultados esperados, ya que claramente representan una fuente de información de primer nivel.

En la **Figura 1** se observa la estructura organizativa del departamento en el cual se desarrolla el proyecto, en donde se identifica claramente una distribución jerárquica. Es importante indicar que el organigrama presentado muestra solo las personas y roles que tendrán relevancia para el desarrollo del proyecto, por ende, se omiten otros puestos que no tendrán un impacto directo en la propuesta.

Figura 1

Equipo del Proyecto



Nota. La figura muestra el organigrama del equipo del proyecto.

El proyecto se desarrolla en la gerencia de Sistemas Banca y Procesos, en donde se utiliza una herramienta de optimización y automatización de procesos llamada Pega, esta será analizada posteriormente en el documento.

Dentro de los roles mostrados en el organigrama, los que tienen relevancia para el desarrollo del proyecto son el Supervisor Senior de Sistemas Pega, el Supervisor Senior de Producto de *Software*, los analistas programadores y los Ingenieros de *Software*. Cabe destacar que por cuestiones de entendimiento del organigrama solo se muestra un rol relacionado para Analista Programador I y II, pero en realidad hay alrededor de 13 personas que ocupan estos puestos involucrados en el proyecto.

En este caso, el Supervisor Senior de Sistemas Pega es el que tiene el rol de patrocinador del proyecto, además de ser el medio de contacto directo con los demás miembros del equipo y miembro principal del grupo llamado Centro de Excelencia Operacional (COE). Dentro de sus responsabilidades se encuentran dar seguimiento a los equipos encargados del desarrollo de proyectos en la plataforma Pega, impulsar proyectos innovadores sobre dicha plataforma y resolver incidentes relacionados con la plataforma.

Asimismo, se encuentra el Supervisor Senior de Producto de *Software*, él se encarga de dar trazabilidad a los diversos ambientes involucrados en el ciclo de vida del lanzamiento del *Software*, específicamente a los ambientes de desarrollo, aseguramiento de la calidad (QA), *staging* y producción. Además, se encarga de la gestión de las diversas licencias necesarias para realizar las funciones en el departamento, también de la supervisión del buen estado general de la plataforma

y de los servicios utilizados, y forma parte de la mesa de aprobación de cambios (CAB) de pases a producción.

Finalmente, se tienen los roles de analistas programadores e ingenieros de *software*, estos están enfocados en la tarea de desarrollo de las diversas funcionalidades necesarias en la plataforma Pega.

1.2.2. Trabajos similares realizados dentro y fuera de la organización

A continuación, de acuerdo con Quesada (comunicación personal, 24 de mayo del 2023), se mencionan dos proyectos desarrollados en el Grupo Financiero, específicamente en el departamento de Sistemas Banca y Procesos relacionados con la mejora en los estándares de desarrollo y el aseguramiento en la calidad del *software*:

- Creación del *Chapter* de Integración y el *Chapter* de Arquitectura.

Este es un espacio que se usa para tratar temas relacionados con el desarrollo eficiente del *software* y el planteamiento de buenas prácticas por parte de los desarrolladores para tratar de disminuir la brecha sobre el tema de la calidad. Este proyecto fue propuesto por parte del equipo del Centro de Excelencia Operacional (COE).

- Proyecto para la implementación de la herramienta *Code review*.

De igual forma, la utilidad llamada *Code review* fue un proyecto impulsado por el Centro de Excelencia Operacional en el que se desarrolló una funcionalidad dentro de la plataforma Pega, este tiene como objetivo que el *software* desarrollado por el programador sea revisado por el líder técnico del equipo al cual pertenece el desarrollador. El líder técnico del equipo es una persona que cuenta con la experiencia necesaria para analizar y determinar a un alto nivel si las funcionalidades desarrolladas son factibles para viajar a los otros ambientes, específicamente a producción. (H. Quesada, comunicación personal, 24 de mayo del 2023).

De forma externa a la organización se pueden mencionar los siguientes proyectos realizados relacionados con el tema de aseguramiento de la calidad del *software*:

- Propuesta de mejora al proceso de aseguramiento y control de calidad para los proyectos de *software* de Inclutec.

Este proyecto se basa en una propuesta para la mejora del aseguramiento y control de calidad de *software* basada en normas y buenas prácticas de calidad que se utilizan actualmente en la industria del *software* para adaptarlo a las necesidades de un grupo de interés, como es el caso de Inclutec (Palacio, 2020).

Algunos de los estándares y buenas prácticas mencionados y utilizados en dicho proyecto son el SWEBOK que brinda contenido útil con respecto a la calidad del *software* y la ingeniería de *software* en general. Además del anterior se utilizaron los siguientes marcos de referencia:

- ✓ IEEE 730-2014, utilizado como guía para la planeación e implementación del proceso de aseguramiento de la calidad del desarrollo de *software*.
- ✓ ISQTB: brindó buenas prácticas relacionadas con *software testing*.
- ✓ ISO 25010:2011: se empleó para el proceso de aseguramiento de calidad, mediante el uso de modelos de calidad para el desarrollo de productos de *software*.

Una de las principales actividades de la organización es el desarrollo de *software*, por esta razón el proyecto se enfoca en esa área. Se requiere realizar un análisis que permita identificar requerimientos de aseguramiento y control de calidad en los procesos actuales y tener alternativas para la gestión efectiva del cumplimiento de los requerimientos (Palacio, 2020).

- Desarrollo de una guía de un marco de referencia de calidad para la metodología de desarrollo de *software* ágil Scrum.

En dicho proyecto se estudió el proceso de pruebas dentro de la metodología de desarrollo ágil Scrum, se recopiló información y se realizó un análisis del ciclo de vida del *software* bajo Scrum, que, a pesar de tener ciertas pautas, normas y buenas prácticas dadas por la metodología, evidencia la necesidad de definir un marco de referencia formal para administrar el proceso de calidad del *software*. Con ese antecedente se planteó una propuesta de marco de referencia de calidad, estructurada y definida, que apoye a la metodología antes mencionada (Rojas, 2016).

Dicha propuesta tiene un soporte teórico apoyado en el CMMI y en estándares internacionales de calidad, para que de esta manera se impulse a la metodología Scrum, por consiguiente, brinde una guía adecuada para un proceso de calidad que sea escalable y que permita aportar un valor agregado al producto de *software*. (Rojas, 2016).

Este modelo CMMI es acompañado de algunos estándares para la propuesta de mejorar el proceso de calidad del *software*, entre ellos el ISO 29110, este fue utilizado para definir los componentes para las pruebas y el proceso de documentación, el ISO 25010 y los estándares IEEE 1028 y el IEEE 829 como guía para el definir las características del proceso de pruebas y documentación de dicho proceso.

1.3. Planteamiento del problema

A continuación, se hace una descripción de la situación problemática que se presenta en la organización y que se toma como un reto para el desarrollo del trabajo final de graduación. Además, se mencionan los principales beneficios esperados producto de la realización del proyecto.

1.3.1. Situación problemática

De forma general, la principal problemática que se presenta en el departamento de Sistemas Banca y Procesos es la deficiencia en el proceso de aseguramiento de la calidad del *software*, ya

que constantemente se tienen que devolver pases enviados a producción, por temas de errores relacionados con la calidad en el *software* desarrollado.

Este incumplimiento con la calidad de las funcionalidades desarrolladas se debe principalmente a no seguir un proceso estándar para el aseguramiento de la calidad del *software*, aplicando pautas para prevenir defectos y garantizar que se cumplan con buenas prácticas, específicamente aplicadas al desarrollo de *software*. A continuación se detalla el contexto de la problemática.

El desarrollo del proyecto se realiza en el departamento de Sistemas Banca y Procesos del Grupo Financiero. En dicho departamento en el año 2018 se llevó a cabo la implementación de una herramienta para la gestión de procesos de negocio, basada en el concepto de *low code* o programación sin código, llamada Pega. (H. Quesada, comunicación personal, 24 de mayo del 2023).

El inicio de la implementación de la herramienta Pega se hizo en el año 2018 con 3 equipos regionales atendiendo 3 proyectos o procesos, para el año 2020 ya no eran 3 equipos de 15 desarrolladores, sino que se contaba con 8 equipos y alrededor de 40 desarrolladores, finalmente, para el año 2023 se cuenta con 30 equipos y un total de 150 desarrolladores. (H. Quesada, comunicación personal, 24 de mayo del 2023).

A raíz del crecimiento exponencial en el número de equipos se comenzaron a crear también equipos locales, además de los regionales, los cuales trabajaban en las mejoras de los procesos propios de cada país. Al carecer de un proceso estándar para el aseguramiento de la calidad en el *software*, surgieron de forma constante errores en los diversos desarrollos enviados a producción, esto sumado al incremento exponencial en el número de pases enviados a dicho ambiente. (H. Quesada, comunicación personal, 24 de mayo del 2023).

El hecho de estar enviando constantemente pases a producción representa una probabilidad elevada de ser impactados por problemas relacionados con la calidad de *software*, situación que se presentó con frecuencia en el año 2022, cuando ciertos productos que estaban en el ambiente de producción resultaron afectados por pases defectuosos enviados a este ambiente, lo cual tuvo un impacto negativo, en cuanto a retrabajo para el negocio, esto se reflejaba en aspectos como tiempo y costos. (H. Quesada, comunicación personal, 24 de mayo del 2023).

Esta fue una de las principales causas que alimenta la problemática, en la cual muchas funcionalidades enviadas al ambiente de producción presentan defectos relacionados con una baja calidad en el *software* desarrollado por los diferentes equipos, tanto a nivel regional como local. (H. Quesada, comunicación personal, 24 de mayo del 2023).

Un aspecto que ha dificultado enfocarse en el tema de aseguramiento de la calidad del *software* es el tiempo, específicamente se atribuye a las prioridades establecidas dentro de la organización, teniendo como principal preocupación entregar resultados rápidos al negocio

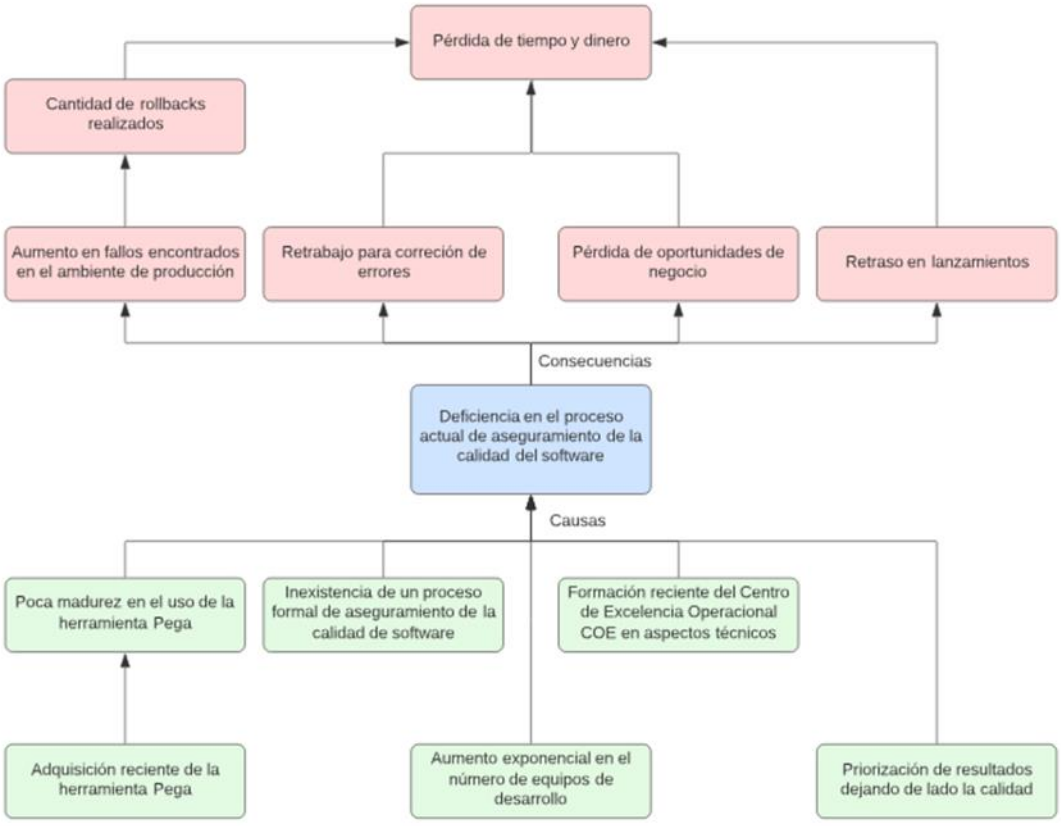
brindando menor importancia al tema de la calidad de los productos desarrollados. (H. Quesada, comunicación personal, 24 de mayo del 2023).

Según Quesada (comunicación personal, 24 de mayo del 2023), a raíz de esta problemática, se han identificado consecuencias de índole negativa como:

- Retrabajo relacionado con el desarrollo de *software* para la corrección de defectos encontrados en el ambiente de producción.
- Retraso en el lanzamiento de funcionalidades previstas en los plazos establecidos por el área de negocio.
- Desalineación con los objetivos planteados por la organización para el periodo.
- Pérdida de oportunidades de negocio producto del retraso en los plazos de entrega.

En la **Figura 2** se muestra el árbol de problemas desarrollado para el proyecto, en este se identifican las principales causas y consecuencias de la problemática identificada.

Figura 2
Árbol del problema



Nota. La figura muestra el árbol del problema elaborado para el proyecto.

1.3.2. Justificación del proyecto

Con respecto a la investigación realizada inicialmente y con los datos recopilados, se determina que existe una oportunidad de mejora para el proceso actual de aseguramiento de la calidad del *software* desarrollado en el departamento de Sistemas Banca y Procesos de la organización. El aseguramiento de la calidad del *software* es un tema que cobra gran relevancia para una organización que busca ofrecer los mejores productos y servicios para sus clientes, ya que un buen producto o servicio brindado es la base para atraer a nuevos clientes y generar un sentimiento de lealtad para los existentes en la organización.

Desde hace algunos años la organización ha experimentado problemas en la calidad del *software* desarrollado, principalmente debido a la falta de un proceso formal establecido en la organización para el aseguramiento de la calidad, basado en buenas prácticas o marcos de referencia actualizados ofrecidos por la industria.

El desarrollo de la propuesta establecida en la presente investigación busca mejorar el estado actual del proceso aplicado, mediante una mejora en la gestión de aseguramiento de la calidad del *software*, a su vez se busca alinear con los objetivos propuestos por la organización con respecto a la estandarización y uso de buenas prácticas.

Otro de los aspectos por los cuales este proyecto toma relevancia es por la visión que tiene el COE con respecto a la automatización de pruebas, para ello se busca establecer una base con respecto a lo necesario para primeramente realizar las pruebas manuales y después la automatización de dichas pruebas.

Los aspectos mencionados anteriormente ayudarían a minimizar la problemática que afecta a la organización, principalmente el ambiente de producción, pues tiene una repercusión directa para los equipos que deben hacer un esfuerzo extra para corregir errores producto de un proceso formal de aseguramiento de la calidad del *software*.

1.3.3. Beneficios esperados o aportes del Trabajo Final de Graduación

En esta sección se indican una serie de beneficios tanto directos como indirectos, producto de la realización del proyecto en la organización.

Beneficios directos:

1. Estandarización del proceso de aseguramiento de calidad del *software*.
2. Utilización de buenas prácticas, estándares y marcos de referencia ofrecidos por la industria para gestionar y documentar el ciclo de vida del desarrollo del *software* en la organización, específicamente en el departamento de Sistemas Banca y Procesos
3. Propicia una menor cantidad de errores al usuario final mediante su identificación en etapas tempranas.

4. Se contará con métricas e indicadores que permitirán a la organización tener visibilidad y trazabilidad sobre la calidad del *software* desarrollado y el impacto que el plan de aseguramiento de la calidad brinde al departamento.
5. Evaluación de los beneficios obtenidos a través de una posible implementación de la mejora del proceso de aseguramiento de la calidad del *software* en la organización.
6. Alineación de la organización en cuanto a la búsqueda de la estandarización en sus procesos.
7. Reducción del número de errores encontrados en el ambiente de producción, por consiguiente disminuirá la cantidad de pases devueltos para los equipos.

Beneficios indirectos:

1. Madurez en el proceso de aseguramiento de la calidad del *software* en la organización.
2. Reconocimiento y visibilidad del trabajo realizado por el Centro de Excelencia Operacional (COE) en búsqueda de la mejora continua en las actividades realizadas en el departamento de Sistemas Banca y Procesos.
3. Actualización en el departamento con respecto a la implementación de buenas prácticas de la industria del desarrollo de *software*.

1.4. Objetivos del Trabajo Final de Graduación

A continuación, se indican los diferentes objetivos propuestos para el proyecto, para la redacción de estos se utilizó el formato SMART y también la taxonomía propuesta por Benjamín Bloom del año 1956.

Objetivo General

Proponer una mejora en el proceso de aseguramiento de la calidad del *software* para el cumplimiento de estándares de calidad requeridos por el departamento de Sistemas Banca y Procesos, durante el segundo semestre del año 2023.

Objetivos Específicos

1. Analizar la situación actual del proceso de desarrollo y aseguramiento de la calidad de *software* dentro del departamento para el entendimiento de la forma en que se realiza actualmente dicho proceso.
2. Rediseñar el proceso de desarrollo y aseguramiento de la calidad del *software* en el departamento para la estandarización de este utilizando las buenas prácticas de la industria.
3. Desarrollar una propuesta de implementación del proceso de aseguramiento de calidad del *software* para la gestión de los recursos asociados a este.

1.5. Alcance

En la sección del alcance se mencionan qué actividades se llevarán a cabo para cumplir con cada uno de los objetivos propuestos inicialmente, a continuación, se muestra el alcance por objetivo.

Objetivo 1: Analizar la situación actual del proceso de desarrollo y aseguramiento de la calidad de *software* dentro del departamento para el entendimiento de la forma en que se realiza actualmente dicho proceso.

Con respecto a este objetivo, se pretende realizar un análisis del proceso actual realizado por algunos equipos de desarrollo, esto servirá como insumo para generar el diagrama *as-is* del proceso utilizando notación BPMN 2.0.

Objetivo 2: Rediseñar el proceso de desarrollo y aseguramiento de la calidad del *software* en el departamento para la estandarización de este utilizando las buenas prácticas de la industria.

Se analizarán algunas buenas prácticas de la industria relacionados con el proceso de aseguramiento de la calidad del *software* para identificar el estado deseado para el proceso, esto permitirá elaborar el diagrama *to-be* con las mejoras identificas de acuerdo con la situación actual y las buenas prácticas.

Objetivo 3: Desarrollar una propuesta de implementación del proceso de aseguramiento de calidad del *software* para la gestión de los recursos asociados a este.

La primera actividad relacionada con este objetivo es realizar un plan de implementación que contemple aspectos como alcance, tiempo, riesgos, involucrados y otros aspectos relacionados con la implementación del nuevo proceso de desarrollo y aseguramiento de calidad del *software*.

Como segundo punto se realiza un análisis costo-beneficio de la propuesta para evaluar si en términos financieros es viable implementarla dentro de la organización. Para ello se realizan dos tipos de análisis, primero se calcula el retorno sobre la inversión del primero y después el periodo de recuperación de la inversión.

1.6. Supuestos

Los supuestos del proyecto corresponden a ciertas condiciones que se deben considerar como verdaderas y que permitirán establecer la base sobre la cual se desarrollará el proyecto, aunque claramente la veracidad de las condiciones podría cambiar en cualquier momento. Los siguientes son supuestos establecidos para el proyecto actual:

- Los requerimientos establecidos al inicio del proyecto no cambiarán por parte del patrocinador del proyecto.

- Se cuenta con la disponibilidad de las personas tanto en el entorno de la organización como fuera de ella para desarrollar el proyecto.
- La información brindada por la organización será de carácter real y relevante para la ejecución de la investigación.
- El visto bueno dado por la subgerencia es definitivo y no cambiará repentinamente, esto asegura que se pueda proseguir con la elaboración de la propuesta.

1.7. Entregables

A continuación, se mencionan los principales entregables que se obtendrán producto del desarrollo del proyecto en la organización.

Entregables del producto

Los entregables que se planean brindar a la organización como desarrollo de la propuesta de este proyecto son:

- Plan de implementación: este documento contendrá la forma en que el proceso propuesto podrá ser implementado dentro del departamento, indicando recursos necesarios, involucrados, cronograma, entre otros aspectos importantes.
- Plan para el aseguramiento de la calidad del *software*: en este documento se describirá el contenido con respecto a cómo se gestionará el propuesto de desarrollo y aseguramiento de calidad del *software* en la organización.

Entregables académicos

Los entregables que se producirán como parte de la ejecución del proyecto son:

- Minutas: las minutas se muestran en la sección de apéndices y servirán para documentar las reuniones realizadas a los largo del proyecto. La plantilla para documentar las minutas es la siguiente **Apéndice A. Plantilla de minutas.**
- El presente documento también forma parte de los entregables académicos.

1.8. Limitaciones

Las limitaciones son aquellas restricciones que pueden tener una repercusión en la ejecución efectiva del proyecto y pueden ser de carácter interno como externo. En esta sección se indican las principales limitaciones para el desarrollo del proyecto y que se deben tener en cuenta ya que pueden repercutir negativamente en el alcance de este.

Las limitaciones identificadas son:

- Acceso a la información necesaria para la ejecución de las diversas fases del proyecto, principalmente para el análisis de la situación actual, en la cual el principal insumo será la información proveniente de la organización.
- Disponibilidad por parte de los diversos involucrados al trabajar bajo la metodología Scrum, pues el tiempo está limitado a realizar las labores propias de cada rol, esto podría afectar la disponibilidad de las personas, por ende, se podría carecer del acceso a la información necesaria en el momento indicado, principalmente a través del uso de herramientas de recopilación de información, como las entrevistas.
- Al ser una plataforma de desarrollo relativamente nueva en la compañía, no se cuenta con un gran volumen de información que permita realizar análisis exhaustivos sobre proyectos relacionados con temas como mejora del ciclo de vida del desarrollo de *software* en la plataforma.
- Algunos estándares relacionados con la gestión de la calidad del *software* son de versión de paga, esto representaría la imposibilidad de acceder a información oportuna para la realización del proyecto.

1.9. Exclusiones del proyecto

Las principales exclusiones que se consideran para este proyecto son las siguientes:

- Capacitaciones: el tema de las capacitaciones se excluye del alcance del proyecto, sin embargo, se toma en cuenta para realizar el cálculo de los costos relacionados con la implementación de la propuesta, ya que dicha actividad es necesaria para garantizar que el alcance del proceso propuesto será comprendido por los principales involucrados.
- Proceso de diseño de *software:* dentro de las recomendaciones brindadas para el proceso *to-be* de aseguramiento de calidad del *software* se indica que se deben incluir actividades para este proceso de diseño de *software*, pero por cuestiones de alcance y limitaciones de tiempo no se toma en cuenta esta actividad para el proceso *to-be*.

A continuación se procede a desarrollar el siguiente capítulo correspondiente al marco conceptual del proyecto.

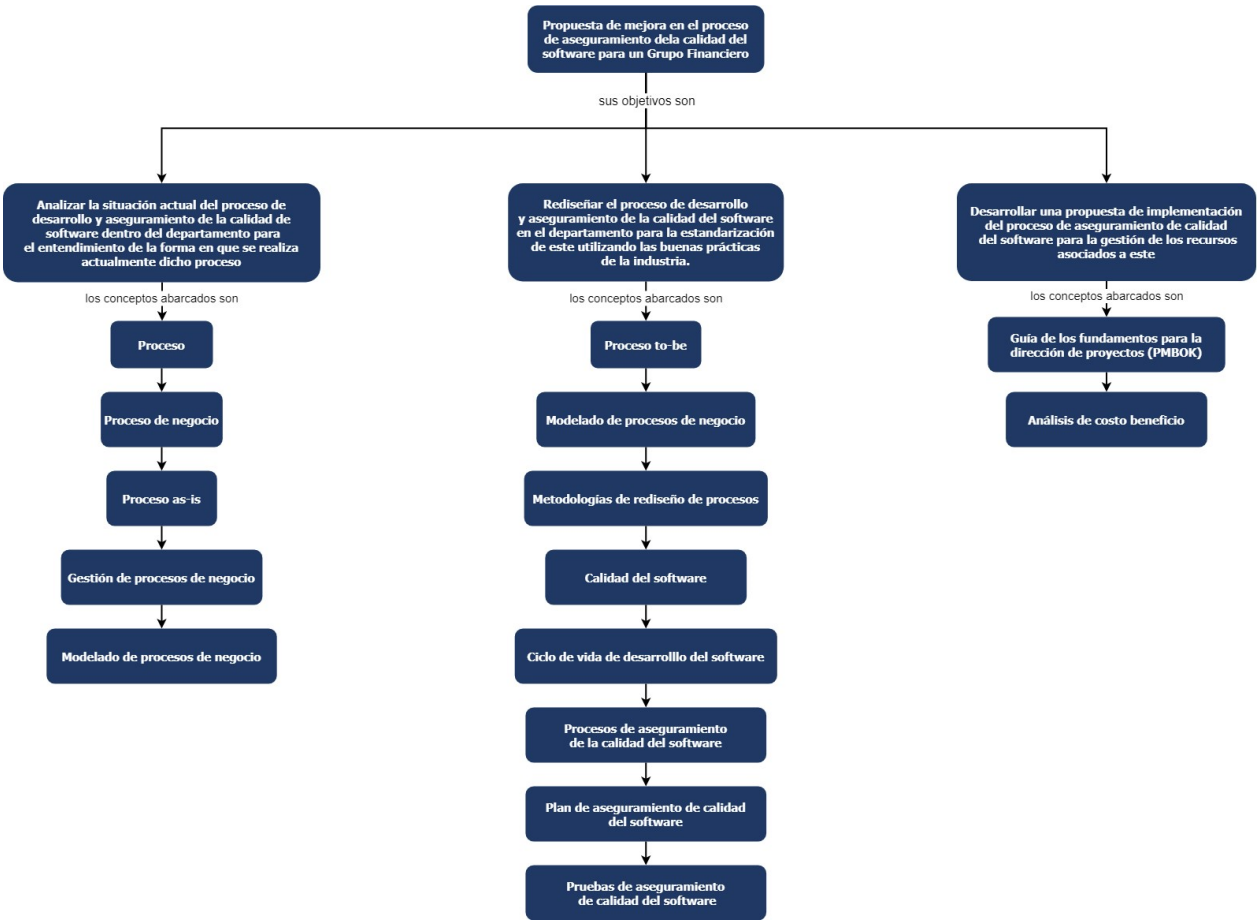
2. Capítulo 2: Marco Conceptual

El marco conceptual ofrece una base de conocimiento que permite comprender las secciones desarrolladas en la presente investigación. En este se indican los términos e información correspondiente a cada temática pertinente al tópico de estudio. Se pretende tener un orden lógico que facilite la compresión del lector.

Con el objetivo de facilitar dicha compresión lectora, se elaboró un mapa de conceptos que muestre a un alto nivel la relación entre los términos y los objetivos de la investigación. Este mapa de conceptos se muestra en la **Figura 3**.

Figura 3

Mapa de conceptos de la investigación



Seguidamente se desarrollan los conceptos y definiciones que tienen relevancia para el proyecto actual.

2.1. Proceso

Primeramente, de acuerdo con el enfoque de este proyecto, es importante comprender qué es un proceso. Según Galindo (2018) “Un proceso es el conjunto de pasos o etapas para llevar a cabo una actividad” (p.91).

De acuerdo con Reyes (2022), se entiende como un proceso una serie o secuencia de pasos que permiten alcanzar un objetivo determinado. Es importante resaltar que un proceso es una secuencia de pasos que debe seguirse en un orden específico, si no se hace de esta manera no se obtendrá el resultado deseado.

2.2. Proceso de negocio

Según Moreno (2023), un proceso de negocio es un conjunto de actividades o tareas estructuradas y coordinadas que una organización realiza para lograr un objetivo específico, por ejemplo, proporcionar un servicio o elaborar un producto. Este implica varios pasos, recursos y personas que trabajan de forma organizada para garantizar el funcionamiento eficiente de la organización y crear valor para los clientes o partes interesadas.

Solano (2019) indica que los procesos de negocio ocurren en todos los niveles de la organización y que estos son la base para el desarrollo de conceptos relacionados como la Gestión de los Procesos de Negocio (BPM) y Automatización Robótica de Procesos (RPA).

2.3. Clasificación de los procesos de negocio

De acuerdo con Solano (2019), los procesos de negocio se pueden clasificar por su propósito y el área de la organización a la que impactan en:

- Procesos operacionales o primarios: estos están relacionados con el negocio central y la cadena de valor de la empresa. Se encargan de generar y entregar valor al cliente en forma de productos o servicios.
- Procesos de apoyo o secundarios: sirven de apoyo a los procesos operacionales y no generan valor a los clientes de manera directa.
- Procesos de gestión: tienen como función medir, monitorear y controlar las actividades relacionadas con los procedimientos y sistemas. De igual forma estos no crean valor directamente al cliente.

2.4. Proceso *as-is*

Según Pierce (2022), un proceso *as-is* define la situación actual, sea de la organización, un departamento o un subproceso y retrata el estado presente del proceso, tal cual se hace hoy en día. El objetivo de un *as-is* es plantear la realidad, ya sea un flujo desordenado, no estandarizado, con tiempos muertos, sin responsables claros o por el contrario un proceso relativamente ordenado. La idea con respecto a este tipo de procesos es identificar los puntos de mejora.

Para diagramar un proceso *as-is* es necesario involucrarse con los usuarios, las personas que realizan el proceso diariamente y conocen el paso a paso, las herramientas que necesitan, los departamentos que se involucran, básicamente toda la información pertinente.

2.5. Proceso *to-be*

Según Pierce (2022), el proceso *to-be* define la situación futura, sea de la organización, un departamento o un subproceso. El diagrama *to-be* retrata el estado mejorado del proceso, a donde se quiere llegar.

Para determinar a quiénes se debe de involucrar depende de los puntos de mejora identificados. Para el análisis deben participar personas que entiendan el proceso *as-is* y tengan la capacidad para cuestionar la situación actual, con el fin de encontrar puntos de mejora.

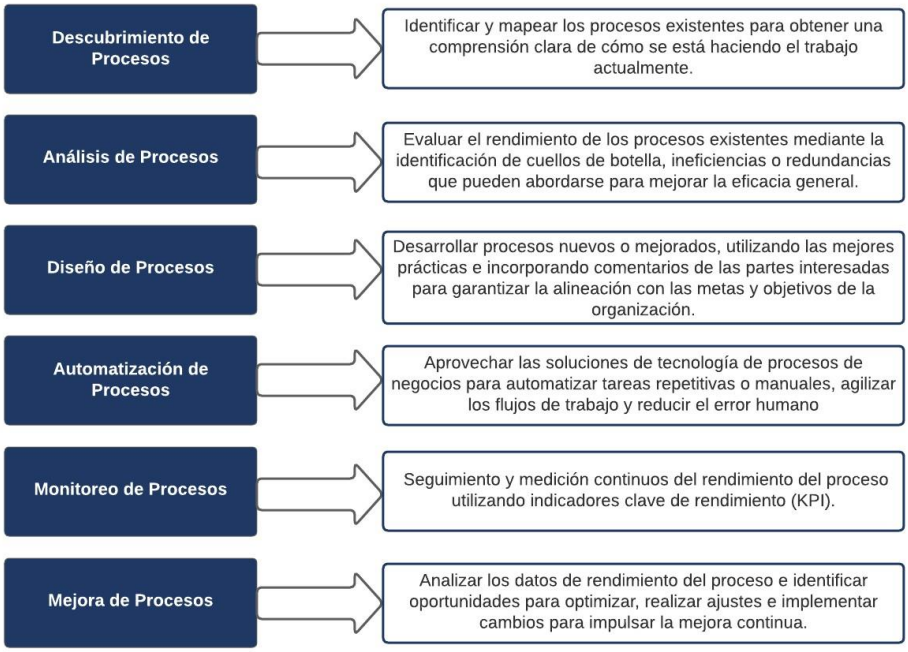
2.6. Gestión de procesos de negocio

De acuerdo con Moreno (2023), la gestión de procesos de negocio, como se conoce en inglés *Business Process Management* (BPM), hace referencia a una metodología estructurada dedicada a la creación, revisión, ejecución, observación y mejora de las operaciones y flujos de trabajo de la organización.

En la **Figura 4**, se muestran los aspectos clave de la Gestión de Procesos de Negocios.

Figura 4

Aspectos clave de la Gestión de Procesos de Negocio



Nota. Tomado de *Procesos de Negocios: La guía completa*, por Moreno, 2023.

2.7. Modelado de procesos de negocio

Según Bizagi (s.f.), el modelamiento de los procesos de negocio es una técnica que se utiliza para documentar, diseñar y optimizar los procesos de una organización.

El modelado de procesos proporciona una representación visual de las etapas y el flujo de un proceso, asimismo, permite que las partes interesadas puedan comprender fácilmente las actividades que le conforman. Además, permite a las organizaciones representar y comunicar los procesos oficiales, mejorar los operaciones y planificar los proyectos de automatización de los procesos de negocio.

2.7.1. BPMN

De acuerdo con Lucidchart (s.f.), la notación del modelado de procesos de negocio (*Business Process Modeling Notation*, BPMN) es un método de diagrama de flujo que modela los pasos de un proceso de negocio planificado de principio a fin. Un aspecto clave de la gestión de procesos de negocio (BPM) es que representa visualmente una secuencia detallada de los flujos de información y las actividades empresariales necesarias para finalizar un proceso.

Elduayen (2022) brinda otra definición "Notación de Modelado de Procesos (BPMN) es una representación gráfica estándar de los procesos y flujos de trabajo de una empresa. Es un diagrama de flujo que representa a los participantes, las opciones y la dirección de dichos procesos mediante gráficos e imágenes estandarizados" (párr. 1).

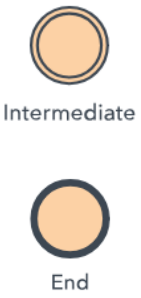
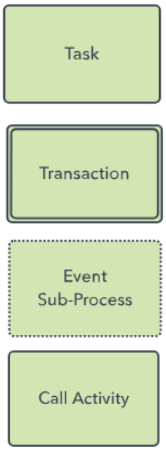
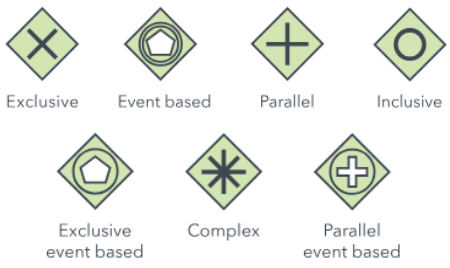
Actualmente, la notación BPMN utiliza la versión 2.0 que según Lucidchart (s.f.), se basa en los siguientes cuatro elementos:

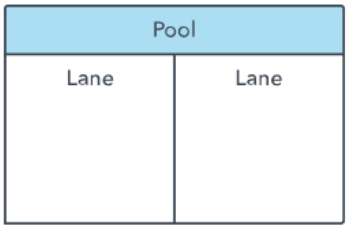
- Objetos de flujo: eventos, actividades y portales.
- Objetos de conexión: flujo de secuencia, flujo de mensaje y asociación.
- Carriles: piscina o carril.
- Artefactos: objeto de datos, grupo y anotación.

Estos elementos se describen en detalle en la **Tabla 1**.

Tabla 1
Elementos de la notación BPMN 2.0

Elemento	Descripción	Representación
Evento	Según Lucidchart (s.f), es un disparador que inicia, modifica o finaliza un proceso. Se muestran con círculos que contienen otros	

Elemento	Descripción	Representación
	símbolos en función del tipo de evento. Se clasifican como "lanzar" o "capturar", según su función.	
Actividad	Según Lucidchart (s.f), es una tarea particular llevada a cabo por una persona o sistema. Se muestra con un rectángulo con bordes redondeados. Pueden volverse detalladas con subprocesos, bucles, compensaciones y múltiples instancias.	
Compuerta lógica	Según Lucidchart (s.f.), es un punto decisivo que puede modificar la ruta en función de las condiciones o los eventos. Se muestran como diamantes. Pueden ser exclusivos o inclusivos, paralelos, complejos o basarse en datos o eventos.	
Flujo de secuencia	Según Lucidchart (s.f), este flujo muestra el orden de las actividades que se realizarán. Se representa con una línea recta con una flecha.	

Elemento	Descripción	Representación
Flujo de mensajes	Según Lucidchart (s.f), muestra mensajes que fluyen en límites organizativos, como los departamentos. No se deben conectar eventos o actividades dentro de una piscina. Se representa con una línea discontinua que contiene un círculo al principio y una flecha al final.	
Asociación	Según Lucidchart (s.f), se muestra con una línea punteada y asocia un artefacto o texto a un evento, actividad o puerta de enlace.	
Carril y piscina	Según Lucidchart (s.f), una piscina representa a los principales participantes de un proceso. Los carriles dentro de una piscina muestran las actividades y los flujos para un determinado rol o participante, definiendo quién es responsable de qué partes del proceso.	
Artefacto	Según Lucidchart (s.f), estos muestran información adicional para aportar el nivel necesario de detalle al diagrama. Hay tres tipos	

Elemento	Descripción	Representación
	de artefactos: objeto de datos, grupo y anotación. Un objeto de datos muestra los datos necesarios para una actividad. Un grupo muestra una agrupación lógica de actividades. Una anotación brinda una explicación completa de una parte del diagrama.	 

Nota. Tomado de Lucidchart (s.f).

2.8. Estandarización de procesos

De acuerdo con Obando (2022), la estandarización de procesos consiste en la unificación de los procedimientos, metodologías y operaciones dentro de una organización, con el fin de crear un modelo reproducible de trabajo y cumplir con parámetros definidos de calidad y eficiencia.

Según Obando (2022), la estandarización de procesos tiene como objetivo principal asegurarse de que se está cumpliendo con los estándares procedimentales y de calidad definidos por una organización. Estas estandarizaciones deben ser integrales e incluir todas las etapas de cumplimiento de un procedimiento, desde su inicio y desarrollo hasta la obtención de resultados. Otro aspecto importante es que la estandarización de procesos es un paso predecesor si se desea optar por la automatización de estos.

2.9. Metodología de rediseño de procesos según Dumas

Dumas et al. (2018), en su libro *Fundamentals of Business Process Management* brindan una serie de fases que son necesarias para realizar reingeniería para los procesos de negocio. Según Dumas et al. (2018), las fases del ciclo de vida de reingeniería de procesos son:

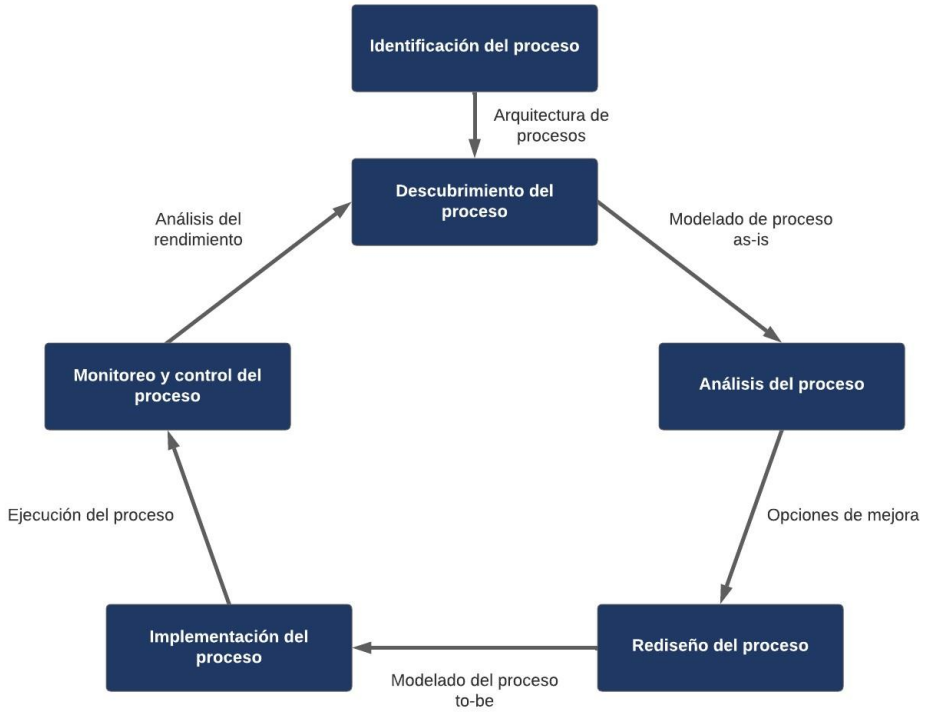
- Identificación del proceso: en esta fase se identifican los procesos de negocio y se relacionan entre sí. Se hace referencia a las actividades de gestión para identificar estos procesos y seleccionar procesos específicos para la mejora. Una de las salidas de esta fase es la denominada arquitectura de procesos.
- Descubrimiento del proceso: en este punto se realiza el levantamiento del proceso *as-is*, este contiene el entendimiento detallado y documentado del proceso de

negocio tal como funciona en el momento de levantamiento de este. Una de las actividades que se realiza en esta fase es la del modelado, que ayuda a expresar y formalizar el proceso.

- Análisis del proceso: una vez que se tiene el proceso *as-is*, se realiza un estudio de este identificando aspectos y oportunidades de mejora.
- Rediseño del proceso: este es el proceso mejorado que se aspira implementar, también conocido como proceso *to-be*. De igual forma, se utiliza el modelado en notación BPMN para representar el proceso *to-be*.
- Implementación del proceso: como su nombre lo indica, en esta fase se tratará de implementar y llevar a la práctica el nuevo proceso diseñado.
- Monitoreo y control del proceso: la finalidad de esta fase es supervisar el proceso para verificar que se ejecuta tal y como se definió inicialmente. Además, se pretende medir el funcionamiento real del proceso, pudiendo de esta forma identificar posibles oportunidades de mejora.

En la **Figura 5**, se muestran las fases del ciclo de vida de un proceso de negocio propuesto por Dumas et al. (2018).

Figura 5
Fases del ciclo de vida de un proceso según Dumas et al



Nota. Adaptado de *Fundamentals of Business Process Management*, por Dumas et al., 2018.

2.10. Metodología de rediseño de procesos según Madison

Madison (2005), en su libro *Process Mapping, Process Improvement, and Process Management*, brinda una metodología para el rediseño de procesos que consta de diez pasos organizados en cuatro fases distintas las cuales son: inicio, análisis del proceso, rediseño del proceso y mejora continua.

De acuerdo con Madison (2005), los diez pasos para la metodología propuesta son:

1. Introducción al proceso de rediseño: se desarrollan actividades para poder seleccionar los procesos que serán rediseñados. Se crea un diagrama de flujo macro con el objetivo de tener una vista completa del proceso y así definir el alcance de este.
2. Formación del equipo de proceso: se selecciona un grupo de personas que será encargado de dirigir la mejora de los procesos.
3. Diagrama "*As-is*": este representa el estado actual sobre el cual se aplicará el rediseño. Para ello se utiliza la técnica de los lentes de frustración.
4. Entrevista con el cliente: Madison (2005) propone entrevistar al cliente y obtener información sobre lo que desea, requiere y necesita obtener del proceso.
5. Benchmarking y buenas prácticas: se realizan comparaciones con buenas prácticas presentes en la industria, con el fin de obtener las mejores prácticas para aplicar al nuevo rediseño.
6. Primer corte del rediseño: se realiza una propuesta sobre el proceso ideal, logrando así de forma consensuada obtener un nuevo proceso basado en los insumos de los primeros pasos y la opinión de cada miembro.
7. Revisión de la gerencia y pruebas: se presenta a la gerencia el nuevo diseño y se comparten ideas sobre la implementación y la gestión de riegos.
8. Diseño final y comunicación a personal y clientes: se comunica el nuevo diseño ya revisado por la gerencia a personal y clientes. Este paso se relaciona con la gestión del cambio.
9. Implementación del rediseño: se aplica el proceso rediseñado. Se puede iniciar con un plan piloto y posteriormente con fases estratégicas de implementación de manera progresiva. Este paso finaliza cuando todo el proceso es aplicado.
10. Instalación de métricas y mejora continua: se implementan mecanismos de control y medición, para así supervisar y tener una mejora continua.

2.10.1. Análisis de lentes de procesos

Según Madison (2005), estos lentes permiten realizar un diagnóstico de los procesos desde la perspectiva de los diversos involucrados, esto para brindar el enfoque adecuado al proceso que será rediseñado, de acuerdo con la necesidad que se presente en dicho momento.

De acuerdo con lo anterior, Madison (2005) indica que el análisis se puede realizar desde la perspectiva de cuatro lentes diferentes, los cuales son:

- Lente de frustración: el objetivo principal de este lente es aprender las distintas frustraciones de las personas que trabajan dentro del proceso a la hora de efectuar su trabajo.
- Lente del tiempo: este lente se centra en la reducción del tiempo para la entrega de valor y satisfacción del cliente. Se busca identificar y eliminar acciones repetidas que sean redundantes en el proceso.
- Lente de costo: su objetivo es identificar las partes del proceso que generen más gasto para intentar reducirlo.
- Lente de calidad: en este lente se analizan aspectos relacionados con la forma en que se desarrollan los procesos de inicio a fin para obtener las salidas esperadas. Este se relaciona con la entrega de valor y con la ventaja competitiva que se pueda generar.

2.11. Calidad

Existen diversas definiciones con respecto al tema de calidad, según Orozco (2023), “la calidad se refiere a aquellos productos o servicios que cuentan con características que cubren las expectativas del cliente, ofreciendo un máximo de conformidad con un margen mínimo de errores y defectos” (p.2).

Según la Organización Internacional de Normalización (*International Organization for Standarization*, en adelante ISO) (2015), “la calidad de los productos y servicios de una organización está determinada por la capacidad para satisfacer a los clientes, y por el impacto previsto y el no previsto sobre las partes interesadas pertinentes” (p.2).

2.12. Aseguramiento de la calidad

El Aseguramiento de la calidad garantiza que los enfoques, técnicas, métodos y procesos diseñados para los proyectos se implementen correctamente. Las actividades de aseguramiento de la calidad monitorean y verifican que los procesos utilizados para gestionar y crear los entregables se hayan seguido y estén operativos (*Software* Testing Help, 2023).

El Aseguramiento de la calidad es un proceso proactivo y es de naturaleza preventiva. Reconoce fallas en el proceso y debe completarse antes que el control de calidad (*Software* Testing Help, 2023).

2.13. Control de la calidad

Este proceso se centra en identificar un defecto. El control de calidad garantiza que los enfoques, técnicas, métodos y procesos diseñados en el proyecto se sigan correctamente (*Software* Testing Help, 2023).

Las actividades de control de calidad monitorean y verifican que los entregables del proyecto cumplan con los estándares de calidad definidos. El Control de Calidad es un proceso reactivo y de naturaleza de detección que reconoce los defectos (*Software* Testing Help, 2023).

2.14. Diferencia entre aseguramiento de calidad y control de calidad

Existe la creencia de que los términos de aseguramiento y control de la calidad son lo mismo y tienen el mismo alcance. Ambos conceptos tienen una relación muy estrecha y se enfocan en la gestión de la calidad, pero tienen un origen diferente.

Como se mencionó anteriormente, el aseguramiento de la calidad se centra en la prevención de defectos, mientras que el control de la calidad tiene como objetivo identificar defectos. En la **Tabla 2** se muestra una comparativa entre las principales diferencias con respecto a cada término.

Tabla 2

Comparación entre aseguramiento y control de la calidad

Aseguramiento de la calidad	Control de la calidad
Tiene como objetivo principal prevenir los defectos.	Su objetivo es identificar y corregir los defectos.
Técnica para la gestión de la calidad.	Método para verificar la calidad.
Todos los miembros del equipo son responsables.	Es responsabilidad del equipo de pruebas.
Asegura que se esté haciendo lo correcto.	Asegura que los resultados de lo que se ha realizado sean los esperados.
Se centra en todo el ciclo de vida de desarrollo de *software*.	Se centra en el ciclo de vida de las pruebas de *software*.
Orientado al proceso.	Orientado al producto.

Nota. Tomado de *Difference Between Quality Assurance And Quality Control (QA Vs QC)*, por *Software* Testing Help, 2023.

2.15. Calidad del *software*

De acuerdo con la IEEE 729-83, la calidad del *software* es el grado con el cual el cliente o usuario percibe que el *software* satisface sus expectativas.

Según la ISO/IEC 25010, cuando se habla de calidad del *software* se hace referencia al grado en que un producto de *software* satisface los requerimientos de sus usuarios, finalmente, esta es la manera en la cual aporta valor.

Por su parte para Pressman (2010) la calidad del *software* se refiere a la concordancia entre los requisitos funcionales y de rendimiento establecidos con los estándares de desarrollo documentados y con las características implícitas que se espera de todo *software* desarrollado profesionalmente.

2.16. Características de la calidad del *software*

Según la ISO/IEC 25010, en su modelo de calidad del producto de *software*, este se encuentra compuesto por ocho características o atributos de calidad los cuales se muestran en la **Tabla 3** a modo de resumen.

Tabla 3

Características y sub características de calidad del producto de software

Característica	Sub característica	
Adecuación Funcional	• Completitud funcional • Corrección funcional • Pertinencia funcional	
Eficiencia de desempeño	• Comportamiento temporal • Utilización de recursos • Capacidad	
Compatibilidad	• Coexistencia • Interoperabilidad	
Usabilidad	• Reconocibilidad de la adecuación • Aprendizabilidad • Operabilidad • Accesibilidad	• Protección contra errores de usuario • Estética de la interfaz de usuario
Fiabilidad	• Madurez • Disponibilidad	• Tolerancia a fallos • Capacidad de recuperación
Seguridad	• Confidencialidad • Integridad • No repudio	• Responsabilidad • Autenticidad

Mantenibilidad	• Modularidad • Reusabilidad • Analizabilidad	• Capacidad para ser modificado • Capacidad para ser probado
Portabilidad	• Adaptabilidad • Capacidad para ser instalado	• Capacidad para ser reemplazado

Nota. Adaptado de *ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, por ISO/IEEE, 2011.

A continuación, se procede a explicar cada uno de estos características o atributos de calidad.

2.16.1. Adecuación funcional

Según la ISO/IEC 25010, este atributo representa la capacidad del producto *software* para proporcionar funciones que satisfacen las necesidades declaradas e implícitas del usuario cuando el producto se usa en las condiciones especificadas. La adecuación funcional se conforma a su vez por tres subcaracterísticas, la cuales son:

- Completitud funcional: grado en el que un conjunto de funcionalidades cubre los objetivos especificados por el usuario.
- Corrección funcional: se refiere a la capacidad del producto de *software* para proveer los resultados correctos con el nivel de precisión requerido.
- Pertinencia funcional: capacidad del producto de *software* para proporcionar un conjunto apropiado de funciones para los objetivos especificados por el usuario.

2.16.2. Eficiencia de desempeño

Según la ISO/IEC 25010, la eficiencia de desempeño se refiere al desempeño en relación con la cantidad de recursos que son utilizados bajo ciertas condiciones. A su vez este atributo se compone de las siguientes subcaracterísticas:

- Comportamiento temporal: tiempo de respuesta y procesamiento cuando un sistema realiza ciertas funciones bajo ciertas condiciones determinadas, en relación con un banco de pruebas.
- Utilización de recursos: se refiere a los tipos de recursos y cantidad utilizadas cuando el *software* lleva a cabo alguna función bajo ciertos criterios específicos.
- Capacidad: es el grado en que los límites máximos de un parámetro de un producto de *software* cumplen con los requerimientos.

2.16.3. Compatibilidad

Según la ISO/IEC 25010, la compatibilidad es la capacidad de dos o más sistemas para intercambiar información y llevar a cabo ciertas funciones requeridas cuando comparten el mismo entorno de *hardware* o *software*. Sus subcaracterísticas son:

- Coexistencia: capacidad del producto de *software* para coexistir con otro producto independiente en un ambiente común compartiendo recursos comunes.
- Interoperabilidad: capacidad de dos o más sistemas para intercambiar y utilizar la información intercambiada.

2.16.4. Fiabilidad:

Según la ISO/IEC 25010, esta característica se refiere a la capacidad de un sistema para desempeñar funciones específicas bajo ciertas condiciones y periodo de tiempo determinados. La fiabilidad reúne las siguientes subcaracterísticas:

- Madurez: capacidad del sistema para poder satisfacer las necesidades de fiabilidad en condiciones normales.
- Disponibilidad: capacidad del sistema para estar operativo y accesible cuando es requerido.
- Tolerancia a fallos: capacidad del sistema para operar según lo previsto en presencia de fallos tanto de *hardware* o de *software*.
- Capacidad de recuperación: capacidad del producto de *software* para recuperar datos y restablecer el estado deseado del sistema, en caso de interrupción o fallo.

2.16.5. Seguridad

Según la ISO/IEC 25010, la seguridad es la capacidad de protección de la información y los datos de manera que personas o sistemas no autorizados no puedan leerlos o modificarlos. Las subcaracterísticas de este atributo son:

- Confidencialidad: capacidad de protección contra el acceso de datos e información no autorizados.
- Integridad: capacidad del sistema para prevenir accesos o modificaciones no autorizados a datos o programas existentes.
- No repudio: capacidad de demostrar las acciones o eventos que han tenido lugar, de manera que dichas acciones o eventos no puedan ser repudiados posteriormente.

- Responsabilidad: capacidad de rastrear de forma inequívoca las acciones de una entidad.
- Autenticidad: capacidad de demostrar la identidad de un sujeto o un recurso.

2.16.6. Mantenibilidad

Según la ISO/IEC 25010, es la capacidad del producto de *software* para ser modificado efectiva y eficientemente, debido a necesidades evolutivas, correctivas o perfectivas. Las subcaracterísticas de este atributo son:

- Modularidad: capacidad de un sistema que permite que un cambio en un componente tenga un impacto mínimo en los demás.
- Reusabilidad: capacidad de un activo que permite ser utilizado en más de un sistema de *software* o en la construcción de otros activos.
- Analizabilidad: facilidad con la que se puede evaluar el impacto de un cambio sobre el resto del *software*, diagnosticar las deficiencias o causas de fallos en el *software*, o identificar las partes por modificar.
- Capacidad para ser modificado: es la capacidad del producto que permite que sea modificado de forma efectiva y eficiente sin introducir defectos o degradar el desempeño.
- Capacidad para ser probado: facilidad con la que se pueden establecer criterios de prueba para un sistema o componente y con la que se pueden llevar a cabo las pruebas para determinar si se cumplen dichos criterios.

2.16.7. Portabilidad

Según la ISO/IEC 25010 es la capacidad del producto de ser transferido de forma efectiva y eficiente de un entorno de *hardware* o *software*, operacional o de utilización a otro. Las siguientes son las subcaracterísticas relacionadas con la portabilidad:

- Adaptabilidad: capacidad del sistema que le permite ser adaptado de forma efectiva y eficiente a diferentes entornos determinados de *hardware* o de *software*.
- Capacidad para ser instalado: es la facilidad con la que el producto se puede instalar o desinstalar de forma exitosa.
- Capacidad para ser reemplazado: capacidad del producto de *software* para ser utilizado en lugar de otro con el mismo propósito y en el mismo entorno.

2.16.8. Usabilidad

Según la ISO/IEC 25010 es la capacidad del producto *software* para ser entendido, aprendido, usado y atraer al usuario. Esta característica se compone de las siguientes subcaracterísticas:

- Reconocibilidad de la adecuación: capacidad del producto de *software* que permite al usuario entender si el *software* es adecuado para sus necesidades.
- Capacidad de aprendizaje: capacidad del producto que permite al usuario aprender sobre su aplicación.
- Operabilidad: capacidad del producto que le permite al usuario operarlo y controlarlo con facilidad.
- Protección contra errores de usuario: capacidad del producto para proteger a los usuarios de cometer errores.
- Estética de la interfaz de usuario: capacidad de la interfaz de usuario del producto/ sistema de agradar y satisfacer la interacción con el usuario.
- Accesibilidad: capacidad del producto que permite que sea utilizado por usuarios que presentan diferentes características de accesibilidad o discapacidades.

2.17. Aseguramiento de la calidad del *software*

Según Maceira (2023), el aseguramiento de calidad en el ámbito del *software* es la creación de un sistema de control para prevenir errores durante la etapa de desarrollo del *software* y reducir la cantidad de defectos en la etapa de prueba. En términos generales se hace referencia a la construcción del proceso de pruebas.

Maceira (2023) indica que el aseguramiento de la calidad del *software* ayuda a verificar el cumplimiento de todos los escenarios comerciales y los requisitos del usuario, así como a identificar todos los posibles problemas y errores en los productos de *software*.

De igual forma, Maceira (2023) hace una diferenciación entre el aseguramiento de calidad y control de la calidad del *software*, indicando que el control de calidad es la implementación del proceso de pruebas para la búsqueda y establecimiento de defectos, mientras que el aseguramiento de la calidad busca prevenir errores durante la etapa de desarrollo.

2.18. Ciclo de Deming o PDCA

Según Sivakumar (2023), el ciclo de Deming se puede definir como un conjunto de cuatro pasos repetitivos, conectados lógicamente que ayudan a la mejora continua de la calidad y al aprendizaje.

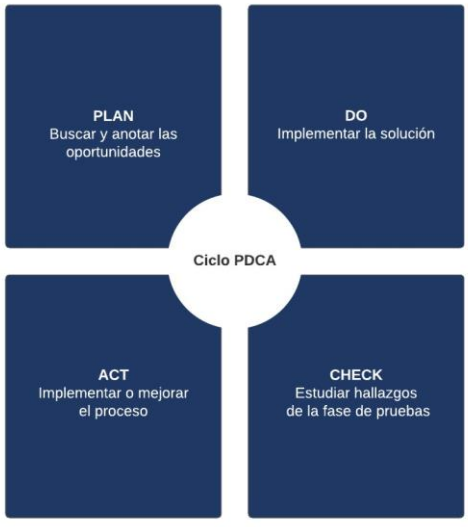
El ciclo de Deming también se conoce como ciclo PDCA (por sus siglas en inglés *Plan, Do, Check, Act*) o ciclo PDSA (por sus siglas en inglés *Plan, Do, Study, Act*) y es una versión modificada del Ciclo Shewart (*Plan, Do see*) que ha existido desde que fue introducido por el experto en estadística Walter. A. Shewart en la década de 1920.

Sivakumar (2023), explica las fases del ciclo de Deming de la siguiente forma:

- *Plan:* esta etapa implica planificar el objetivo final y el proceso a seguir para alcanzarlo. En caso de que se esté buscando mejorar o resolver un problema en un proceso existente, aquí es donde se reúne toda la información y los pasos necesarios que podrían ayudar a resolver el problema.
- *Do:* es necesario implementar las correcciones y soluciones que se materializaron durante la fase de planificación. Se recomienda implementar los cambios a menor escala, a modo de prueba, para saber si la solución resulta útil o no.
- *Check (Study):* en esta fase se verifican los resultados obtenidos de las pruebas en busca de cambios y mejoras. Si todavía surgen problemas, hay que descubrir las causas y encontrar una solución.
- *Act:* fase final en donde se implementan en su totalidad los cambios recomendados y los procesos probados. En caso de que se requieran ejecutar más cambios se deben tomar observaciones de este paso y se debe de reiniciar el ciclo una vez más.

Este ciclo de Deming o PDCA debe seguirse hasta que se cumplan todas las expectativas necesarias y no haya necesidad de realizar más cambios en el futuro. En la **Figura 6** se muestra resumido este ciclo.

Figura 6
Ciclo de Deming o PDCA



Nota. Adaptado de *Deming Cycle: Plan-Do-Check-Act (PDCA) Cycle Explained*, por Sivakumar, 2023.

2.19. Proceso de ingeniería de requerimientos de software

De acuerdo con Gupta (2023), este proceso define, documenta y mantiene los requisitos en el proceso de diseño de ingeniería. Proporciona un mecanismo eficaz para comprender los deseos de los clientes, analizar sus necesidades, evaluar la viabilidad de los requisitos proporcionados, negociar una solución razonable a los problemas que surgen, especificar la solución, validar las especificaciones y gestionar los requisitos.

Según Gupta (2023), el proceso de ingeniería de requerimientos se modela con las siguientes actividades:

- Obtención y análisis de requisitos

Es un proceso de interacción con clientes y usuarios finales para observar los requisitos del dominio y así decidir qué servicios debe proporcionar el sistema y las demás limitaciones. Algunas técnicas para la recopilación de requerimientos son las siguientes: entrevistas, lluvias de ideas, creación de prototipos, entre otras.

- Especificación de requerimientos

Gupta (2023) indica que es el proceso de anotar los requisitos del sistema y los requisitos del usuario en un documento. Las condiciones deben ser claras, comprensibles, completas y coherentes. Es trabajo del analista escribir los requisitos para el equipo de desarrollo en lenguaje técnico para que puedan entenderse fácilmente y sean beneficiosos.

- Priorización de requerimientos

De acuerdo con Gupta (2023), esta es una fase en la cual se trabaja de la mano con el cliente para establecer una serie de prioridades con respecto a los requerimientos recopilados, esto permitirá enfocarse en trabajar en los requerimientos que tienen mayor urgencia según la clasificación realizada.

- Verificación y validación de requisitos

Según Gupta (2023), la verificación de requisitos se refiere al conjunto de tareas realizadas en el *software* para garantizar que una función específica se implemente correctamente en él.

Por otra parte, la validación de requisitos es un proceso que garantiza que los requisitos especificados satisfagan las necesidades de los clientes. Algunas actividades para la validación de requerimientos son: revisiones de requisitos, creación de prototipos, generación de casos de prueba, entre otros.

2.20. Proceso de pruebas de aseguramiento de calidad del software

Existen varios enfoques o buenas prácticas para realizar el proceso de pruebas aseguramiento de calidad del *software*, a continuación, se describen algunos que serán tomados en cuenta para la realización del presente proyecto.

- Proceso pruebas de aseguramiento de la calidad del *software* según el SWEBOK

Según el SWEBOK (2014), las actividades de este proceso son las siguientes:

- Planeación

Se deben planificar las actividades de prueba. Los aspectos clave de la planificación de pruebas incluyen la coordinación del personal, la disponibilidad de los equipos de prueba, la creación y el mantenimiento de toda la documentación relacionada con las pruebas y la planificación para posibles resultados indeseables.

Si se mantiene más de una línea base del *software*, entonces una consideración importante de planificación es el tiempo y el esfuerzo necesarios para garantizar que el entorno de prueba tenga la configuración adecuada.

- Generación de casos de prueba

Esta actividad se basa en el nivel de prueba que se realizará y las técnicas de prueba particulares. Los casos de prueba deben estar bajo el control de la gestión de la configuración del *software* e incluir los resultados esperados para cada prueba.

- Preparación del ambiente de pruebas

Se refiere a que el entorno utilizado para las pruebas debe ser compatible con las otras herramientas de ingeniería de *software* adoptadas. Debería facilitar el desarrollo y control de casos de prueba, así como el registro y recuperación de resultados esperados, *scripts* y otros materiales de prueba.

- Ejecución

Según el SWEBOK (2014), la ejecución de las pruebas debe incorporar un principio básico de la experimentación científica, todo lo que se haga durante las pruebas debe realizarse y documentarse con suficiente claridad para que otra persona pueda replicar los resultados. Esto quiere decir que las pruebas deben realizarse de acuerdo con procedimientos documentados utilizando una versión claramente definida del *software* bajo prueba.

- Evaluación de los resultados de las pruebas

Se deben evaluar los resultados de las pruebas para determinar si estas han sido exitosas o no. En la mayoría de los casos, se dice que un resultado exitoso significa que el *software* funcionó como se esperaba y no tuvo resultados inesperados importantes. No todos los resultados

inesperados son necesariamente fallas, en ocasiones se determina que son simplemente ruido.

Antes de que se pueda eliminar una falla, se necesita un esfuerzo de análisis y depuración para aislarla, identificarla y describirla. Cuando los resultados de las pruebas son particularmente importantes, se puede convocar una junta de revisión formal para evaluarlos.

- Reporte de problemas/ Registro de pruebas

Las actividades de prueba se pueden ingresar en un registro de pruebas para identificar cuándo se realizó una prueba, quién la realizó, qué configuración de *software* se utilizó y otra información de identificación relevante.

Los resultados de las pruebas inesperados o incorrectos se pueden registrar en un sistema de notificación de problemas, cuyos datos forman la base para la posterior depuración y solución de los problemas que se observaron como fallas durante las pruebas. Además, las anomalías no clasificadas como fallos podrían documentarse en caso de que posteriormente resulten más graves de lo que se pensaba.

- Seguimiento de defectos

Los defectos se pueden rastrear y analizar para determinar cuándo se introdujeron en el *software*, por qué se crearon y cuándo se pudieron haber observado por primera vez en el *software*.

- Proceso pruebas de aseguramiento de la calidad del *software* según Jeff Tian

Tian (2005), indica en su libro *Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement*, que existen tres tipos de actividades principales relacionadas con el proceso de aseguramiento de *software* y el proceso de pruebas, estas actividades son:

- Planeación y preparación de las pruebas

Esta actividad comprende las siguientes subactividades:

  - Establecer metas.
  - Seleccionar estrategias y técnicas generales.
  - Reunir información.
  - Construir modelos de pruebas.
  - Preparar casos de prueba individuales.
  - Preparación del conjunto de pruebas generales.
  - Preparación del procedimiento de pruebas.

- Ejecución y medición de pruebas

Esta actividad está compuesta por los siguientes pasos:

- Asignar y ajustar el tiempo y los recursos de las pruebas.
- Ejecutar las pruebas.
- Comprobar los resultados de las pruebas e identificar fallos del sistema.

- Análisis de resultados de pruebas y actividades de seguimiento

Esta actividad consiste en el posible diagnóstico de problemas y reparación de defectos como seguimiento de ejecuciones de pruebas individuales, y análisis de resultados de pruebas generales para decisiones de gestión e iniciativas de mejora de la calidad.

- Automatización de pruebas

Tiene como objetivo automatizar algunas tareas manuales con el uso de herramientas de *software*. La demanda de automatización de pruebas es fuerte, porque las pruebas puramente manuales de principio a fin pueden ser tediosas y propensas a errores. Es importante mencionar también que no es posible realizar pruebas completamente automatizadas.

- Proceso pruebas de aseguramiento de la calidad del *software* según Thomas Hamilton

De acuerdo con Hamilton (2023), y lo descrito en una publicación en su sitio *web* Guru99.com existen seis fases que forman parte del ciclo de vida de pruebas en el *software*, estas fases son las siguientes:

- Análisis de requerimientos

Las actividades que se llevan a cabo en esta fase son:

- Identificar los tipos de pruebas a realizar.
- Recopilar detalles sobre las prioridades y el enfoque de las pruebas.
- Preparar una Matriz de Trazabilidad de Requisitos (RTM).
- Identificar los detalles del entorno de prueba donde se supone que se llevarán a cabo las pruebas.

- Planificación de las pruebas

En esta fase se realizan las siguientes actividades:

- Preparación de un plan de pruebas para varios tipos de pruebas.
- Selección de herramientas de prueba.
- Estimación del esfuerzo de prueba.

- Planificación de recursos y determinación de roles y responsabilidades.

- Desarrollo de los casos de prueba

Sus principales actividades son:

- Crear los casos de prueba.
- Revisión de los casos de prueba.
- Crear datos de prueba si es que existe un entorno de pruebas disponible.

- Configuración del ambiente de pruebas

En esta fase se realizan las siguientes actividades:

- Comprender la arquitectura requerida, la configuración del entorno y preparar la lista de requisitos de hardware y *software* para el entorno de prueba.
- Configuración del entorno de prueba y datos de prueba.

- Ejecución de las pruebas

Para esta fase se indican las siguientes actividades:

- Ejecutar las pruebas según el plan.
- Documentar los resultados de las pruebas y registrar los defectos de los casos fallidos.
- Asignar los defectos a los casos de prueba en la Matriz de Trazabilidad de Requisitos.
- Volver a probar las correcciones de defectos.
- Seguimiento de los defectos hasta el cierre.

- Cierre del ciclo de pruebas

Sus principales actividades son:

- Evaluar los criterios de finalización del ciclo en función del tiempo, la cobertura de la prueba, el costo, el *software*, los objetivos comerciales críticos y la calidad.
- Preparar métricas de prueba basadas en los parámetros anteriores.

- Documentar el aprendizaje del proyecto.
- Elaborar informe de cierre de pruebas.
- Análisis de resultados de pruebas para conocer la distribución de defectos por tipo y gravedad.

- Proceso pruebas de aseguramiento de la calidad del *software* según Spillner et al.

De acuerdo con Spillner et al. (2014), las principales actividades del proceso de aseguramiento de la calidad del *software* son:

- Planificación y control de las pruebas

Se deben planificar los recursos asociados a las pruebas al inicio de este proceso y elaborar un plan de pruebas que documente todos los aspectos necesarios sobre la ejecución de pruebas.

También se debe realizar una priorización de las pruebas que se deben ejecutar de acuerdo con las necesidades del momento. Adicionalmente, se deben configurar las herramientas e infraestructura para la ejecución de las pruebas.

- Análisis y diseño de las pruebas

En esta fase primeramente se especifica lo que debería ser probado, para ello, dicha especificación debe ser concreta y clara para desarrollar los casos de prueba. Se debe especificar cuales casos de prueba satisfacen requerimientos específicos y además establecer cuales precondiciones se deben cumplir para desarrollar con éxito los casos de prueba.

Otro aspecto importante es que se deben establecer los resultados y comportamientos esperados de las pruebas antes de ejecutarlas.

- Implementación y ejecución de las pruebas

Cuando las precondiciones establecidas para la ejecución de los casos de prueba se cumplen, se puede iniciar con dichas pruebas. En esta fase también toma importancia la prioridad de ejecución de los casos de pruebas establecidos en el plan de pruebas.

Para facilitar la comprensión y eficiencia en la ejecución de los casos de prueba se deben agrupar estos casos en conjuntos de pruebas. Cuando las condiciones para la ejecución de los casos de prueba están listas se procede a comenzar las pruebas, para ello se pueden realizar las pruebas manuales o utilizar herramientas de *software* para facilitar este proceso.

La última actividad en esta fase es documentar los resultados de las pruebas ejecutadas y si existen defectos encontrados en el *software* se deben corregir y realizar pruebas nuevamente.

- Evaluación y reporte de las pruebas

Se debe definir si el ciclo de pruebas se ha completado de acuerdo con aspectos como el criterio de cobertura de las pruebas. Si este criterio se cumple se puede decir que el proceso terminó correctamente, entonces se debe de realizar un resumen sobre el plan de pruebas ejecutado.

- Actividades de cierre de las pruebas

Se debe realizar una evaluación del proceso de pruebas ejecutado para encontrar opciones de mejora a dicho proceso y contribuir a la mejora continua.

2.21. International Software Testing Qualifications Board

Según Faro (2022), el Comité Internacional de Certificaciones de Pruebas de Software (*International Software Testing Qualifications Board*, ISTQB), es una entidad que nació con el objetivo de definir un esquema de certificación internacional para la calidad del *software*. Suministra el plan de estudios y el glosario que definen lo que establecen las guías para la acreditación y evaluación de los profesionales del *testing* a cargo de los comités de cada país.

Entre otras funciones, permite formarse y certificarse en tres módulos distintos y en tres niveles diferentes. Los distintos niveles de certificación ISTQB son inicial, avanzado y experto y los módulos son *core*, ágil y especialista.

2.22. Metodología de priorización MoSCoW

De acuerdo con VanZandt (2023), la metodología de priorización MoSCoW se define como un marco organizacional que ayuda a aclarar y priorizar características o requisitos para un proyecto determinado. Al crear límites para las prioridades, los equipos pueden limitar su enfoque y crear metas directas y alcanzables.

Siguiendo la idea de VanZandt (2023), MoSCoW es un acrónimo que representa las cuatro categorías en las que se pueden clasificar las distintas características. Estas categorías son:

- *Must have* (Debe tener): son los requisitos esenciales que deben incluirse en el proyecto o producto. Si alguno de estos requisitos no se cumple, el proyecto o producto no puede considerarse exitoso.
- *Should have* (Debería tener): son requisitos importantes que deberían incluirse si es posible. No son absolutamente críticos pero agregan un valor significativo al proyecto o producto.
- *Could have* (Podría tener): son requisitos que es bueno tener, pero no son críticos. Pueden considerarse si el tiempo y los recursos lo permiten, pero pueden aplazarse si es necesario.
- *Won´t have* (No tendrá): son requisitos que están explícitamente fuera del alcance del proyecto o producto actual. Estos requisitos no están actualmente bajo consideración.

2.23. Ciclo de vida de desarrollo del *software*

Según González (2022), el ciclo de vida de desarrollo de *software* es un desglose de todas las fases implicadas en el proceso de desarrollo de *software*. Cada equipo de desarrollo de *software* sigue un proceso de desarrollo establecido, sin embargo, existen algunos principios básicos comunes a todas las estrategias de ciclo de vida de desarrollo de *software*.

Siguiendo con la idea de González (2022), las fases comunes del ciclo de vida de desarrollo de *software* son:

- Análisis de requisitos: fase en la que el equipo comprenda los requisitos de su proyecto. En esta fase no sólo intervienen los desarrolladores, también puede ser necesaria la ayuda de los analistas de negocio.
- Fase de planificación: en este apartado se dividen las tareas del proyecto, así como se realiza una estimación de los costes, análisis de riesgos, asignación de recursos y personal, entre otros.
- Fase de diseño: el desarrollador o el equipo de desarrolladores deben diseñar cuidadosamente su producto de *software*.
- Fase de codificación: es donde los desarrolladores empiezan realmente a crear *software*. Si se ha elegido el enfoque tradicional, aquí es donde se empieza a escribir código, por el contrario, si se ha elegido un enfoque diferente como *low-code* o *no-code* aquí es donde empiezan a utilizar dichas plataformas de desarrollo.
- Fase de pruebas: el *software* debe ser sometido a una fase de pruebas previas para garantizar que funcionará y que va a satisfacer las necesidades de los usuarios. Esta fase permite identificar errores y solucionarlos a tiempo.
- Fase de despliegue: es cuando el *software* se implementa en las plataformas seleccionadas.
- Fase de mantenimiento: todo *software* requiere mantenimiento, es necesario actualizarlo constantemente, solucionar los posibles problemas que puedan surgir y mantenerlo al máximo de sus posibilidades.

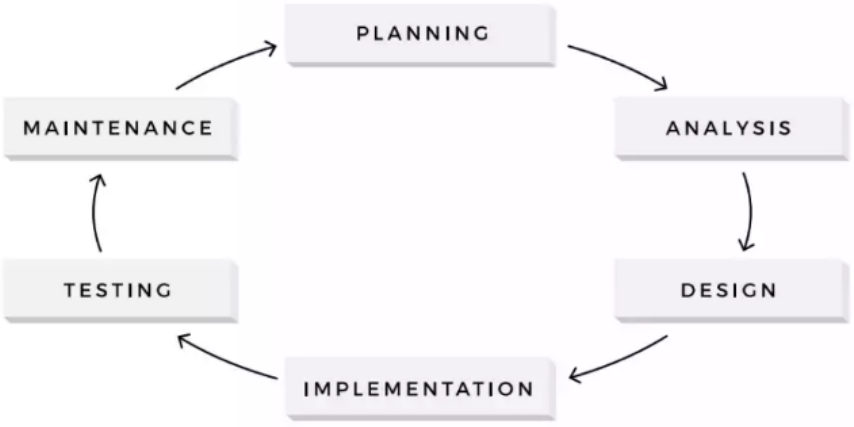
2.23.1. Metodologías de desarrollo de *software*

De acuerdo con Altexsoft (s.f.), existen algunas metodologías que siguen diferentes enfoques para realizar el ciclo de vida de desarrollo de *software*, a continuación, se mencionan tres de estas metodologías.

- Metodología en cascada: representa el ciclo de vida de desarrollo de *software* tradicional, incluye seis fases consecutivas: planificación, análisis, diseño, implementación, pruebas y mantenimiento.

Figura 7

Metodología de desarrollo de software en cascada

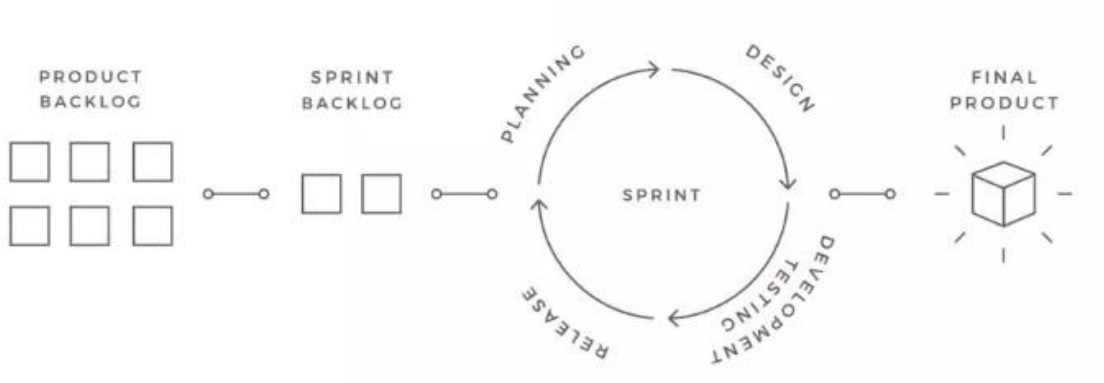


Nota. Tomado de *Quality Assurance, Quality Control and Testing — the Basics of Software Quality Management*, por Altexsoft, s.f.

- Metodología ágil: divide el proceso de desarrollo en partes, iteraciones o *sprints* pequeños. Esto permite a los evaluadores trabajar en paralelo con el resto del equipo durante todo el proceso y corregir las fallas y errores inmediatamente después de que ocurren.

Figura 8

Metodología ágil de desarrollo de software



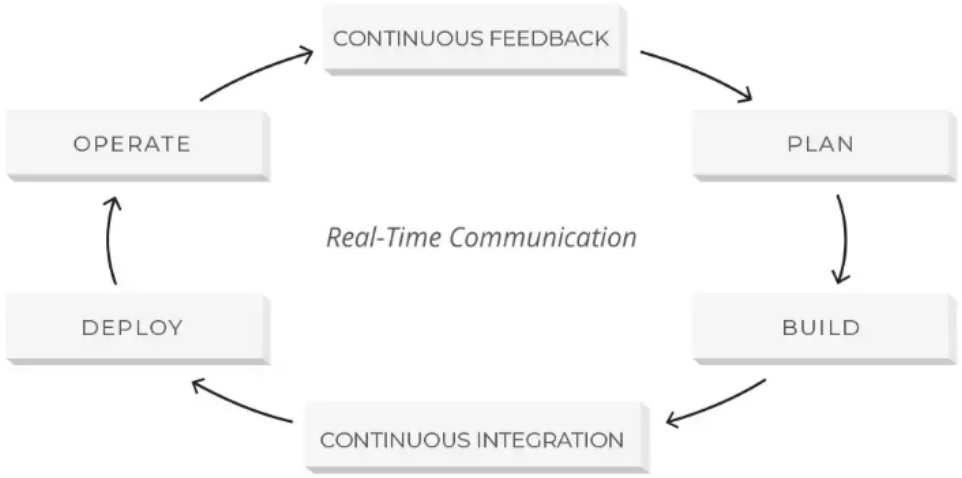
Nota. Tomado de *Quality Assurance, Quality Control and Testing — the Basics of Software Quality Management*, por Altexsoft, s.f.

El objetivo principal de este proceso es ofrecer nuevas funciones de *software* de forma rápida con calidad. Por lo tanto, este enfoque es menos costoso; además, corregir los errores en las primeras etapas del proceso de desarrollo, antes de que se acumulen más problemas, es significativamente barato y requiere de menos esfuerzo.

- DevOps: esta es una nueva metodología de desarrollo de *software* que requiere un alto nivel de coordinación entre varias funciones de la cadena de entrega, a saber, desarrollo, control de calidad y operaciones.
- DevOps incluye el concepto de desarrollo continuo en el que el código, escrito y versionado con el control de versiones, se creará, implementará, probará e instalará en el ambiente de producción que esté listo para ser consumido por el usuario final.

Figura 9

Metodología DevOps



Nota. Tomado de *Quality Assurance, Quality Control and Testing — the Basics of Software Quality Management*, por Altexsoft, s.f.

2.24. Plan de aseguramiento de calidad del *software*

Según el SWEBOK (2015), este plan define las actividades y tareas empleadas para garantizar que el *software* desarrollado para un producto específico satisfaga los requisitos establecidos del proyecto y las necesidades del usuario dentro de las restricciones de costo y cronograma del proyecto y sea proporcional a los riesgos del proyecto. El plan de aseguramiento de calidad del *software* primero garantiza que los objetivos de calidad estén claramente definidos y comprendidos.

2.25. Pruebas de aseguramiento de calidad de *software*

De acuerdo con Turrado (2020), las pruebas de *software* son un conjunto de procesos con los que se pretende probar un sistema o aplicación en diferentes momentos para comprobar su correcto funcionamiento.

Por su parte IBM (s.f.), indica que "la prueba de *software* es el proceso de evaluar y verificar que un producto o aplicación de *software* hace lo que se supone que debe hacer. Los beneficios de las pruebas incluyen la prevención de errores, la reducción de los costos de desarrollo y la mejora del rendimiento" (párr. 1).

2.25.1. Tipos de pruebas de aseguramiento de calidad de *software*

Según Hamilton (2023), existen cuatro niveles de pruebas de *software*, estás pruebas son:

- Pruebas unitarias: este tipo ayuda a probar cada módulo o unidad de *software* por separado. El objetivo es probar cada parte del *software* separándola y comprobando que el componente cumple o no con la funcionalidad.
- Pruebas de integración: en este nivel de pruebas se combinan y prueban diferentes módulos de *software* en grupo para garantizar que el sistema integrado esté listo para la prueba. La integración de pruebas verifica el flujo de datos de un módulo a otros módulos.
- Pruebas de sistema: se realizan en un sistema completo e integrado. Permite verificar el cumplimiento del sistema según los requisitos. Pone a prueba la interacción general de los componentes. La prueba del sistema suele ser la prueba final para verificar que el sistema cumple con las especificaciones.
- Pruebas de aceptación: determina si se cumplen los requisitos de una especificación o contrato según su entrega.

2.26. Matriz de responsabilidades RACI

De acuerdo con Good (2023), una matriz RACI es un documento que aclara qué personas o grupos son responsables de la finalización exitosa de un proyecto y los roles que cada uno desempeñará a lo largo del proyecto. El acrónimo RACI significa los diferentes tipos de responsabilidad: responsable, aprobador, consultado e informado.

- Responsable: suelen ser responsables de desarrollar y completar los entregables del proyecto. Las partes responsables suelen ser miembros prácticos del equipo que hacen contribuciones directas para la finalización del proyecto.
- Aprobador: es responsable de supervisar la finalización general de la tarea, no obstante, es posible que no sea la persona que en realidad realiza el trabajo.

- Consultado: estas personas brindan orientación que a menudo es un requisito previo para otras tareas del proyecto, por ejemplo, brindar orientación legal sobre un proyecto durante todo el proceso.
- Informado: son aquellas que necesitan permanecer al tanto de la comunicación durante todo el proyecto. Estas personas no tienen que ser consultadas ni ser parte de la toma de decisiones, pero se les debe informar de todas las actualizaciones del proyecto.

2.27. Análisis costo-beneficio

Según MacNeil (2022), el análisis de costo-beneficio es una herramienta de toma de decisiones que sirve para elegir con qué acciones vale la pena avanzar. Ofrece una perspectiva cuantitativa del problema para tomar decisiones basadas en evidencia y no en opiniones subjetivas o prejuicios.

Por su parte Bello (2021), indica que un análisis de coste-beneficio es el proceso de comparar los costes y los beneficios u oportunidades estimados asociados con la decisión de un proyecto. El objetivo de este estudio es determinar si tiene sentido desde una perspectiva comercial.

Seguidamente, Bello (2021) alude a que este tipo de análisis implica sumar todos los costos de un proyecto o decisión. Al resultado se le resta esa cantidad de los beneficios totales proyectados. Si los beneficios proyectados superan los costes se puede argumentar que la decisión es buena, en caso contrario no se recomienda continuar con la implementación.

De esta forma se culmina con la explicación de los principales conceptos que serán utilizados en los siguientes capítulos y que permitirán comprender las ideas que se expondrán seguidamente. Una vez culminado este capítulo se prosigue con el capítulo 3 marco metodológico del proyecto.

3. Capítulo 3: Marco Metodológico

En este capítulo se explica la forma en que el proyecto se desarrolla. Es decir, se detallan secciones como el tipo de investigación, el enfoque y diseño de la investigación, de dónde provendrán los datos e información para llevar a cabo el proyecto, así como los principales sujetos de investigación. Además de lo anterior, dentro de este apartado se plantean las fases de desarrollo, explicadas en la sección del procedimiento metodológico.

3.1. Tipo de investigación

De acuerdo con Hernández et al. (2014), la investigación es útil para distintos fines: crear nuevos sistemas y productos; resolver problemas económicos o sociales; ubicar mercados, diseñar soluciones y hasta evaluar si se ha realizado algo correctamente o no. En resumen, la investigación está presente en casi todos los aspectos de la vida.

La investigación científica es un tipo de investigación formal y estructurada y se caracteriza por ser sistemática, empírica y crítica. Cuando se dice que es sistemática se hace referencia a que hay una disciplina para realizar la investigación y que los hechos no se dejan a la casualidad. El término crítico quiere decir que la investigación se evalúa y es mejorada de manera constante. Por último, cuando se dice que la investigación científica es empírica es porque se recolectan y analizan datos.

La investigación científica se puede clasificar de dos formas diferentes, como investigación básica, que tiene como objetivo producir conocimiento y teorías; o como investigación aplicada, la cual busca resolver problemas. De acuerdo con lo anterior y con el desarrollo de este proyecto, se dice que el presente estudio es una investigación de tipo aplicada.

3.2. Enfoque de la investigación

De acuerdo con Mata (2019), cuando se habla del enfoque de la investigación se hace referencia a la naturaleza del estudio y abarca el proceso investigativo en todas sus etapas, desde la definición del tema y el planteamiento del problema de investigación, así como el desarrollo del aspecto teórico, la definición de la metodología y la recolección y análisis de los datos.

Según Hernández et al. (2014), existen tres tipos de enfoques para una investigación, estos son el cuantitativo, el cualitativo y el mixto.

El enfoque cuantitativo se caracteriza por utilizar la recolección de datos para probar hipótesis con base en mediciones numéricas, hace uso del análisis estadístico y es probatorio. En este la recolección de datos se fundamenta en la medición, estos se representan mediante números y se deben analizar con métodos estadísticos. Otra de las características importantes de este enfoque es que se intentan generalizar los resultados encontrados en un grupo o segmento denominado "muestra" con respecto a una colectividad denominada "universo o población".

Con respecto al enfoque cualitativo se puede decir que el investigador plantea un problema, pero no sigue un proceso definido. Este enfoque se basa en un proceso inductivo caracterizado por explorar y describir, para luego generar perspectivas teóricas. Además, en este la recolección de datos se basa en métodos no estandarizados, consiste en obtener las perspectivas y puntos de vista de los participantes, en donde el investigador hace preguntas abiertas, se recaba información a través de medios verbales y no verbales, así como visuales, para luego analizar dicha información.

Otro aspecto importante que caracteriza esta investigación es que se utilizan técnicas para la recolección de información como entrevistas, observación, revisión de documentos, discusiones en grupo, evaluación de experiencias personales, entre otras.

Por último, como lo indica su nombre el enfoque mixto implica combinar aspectos de los enfoques cuantitativos y cualitativos en un mismo estudio.

Según la información anterior y la naturaleza de esta investigación, se determina que el enfoque de investigación será el cualitativo, en donde se analizarán desde la perspectiva de diversos actores y sus vivencias.

3.3. Alcance de la investigación

Según Hernández et al. (2014), el alcance de la investigación indica el resultado o lo que se obtendrá a partir de esta y condiciona el método que se seguirá para obtener dichos resultados. De acuerdo con el mismo autor, existen cuatro tipos diferentes de alcances: el exploratorio, el correlacional, el descriptivo y el explicativo.

El objetivo del alcance exploratorio es examinar un tema o problema de investigación poco estudiado, el cual se ha abordado de una forma superficial. También es útil para establecer las bases para investigaciones futuras o sugerir afirmaciones y postulados.

El alcance correlacional tiene como finalidad establecer o conocer el grado de relación que se presenta entre dos o más conceptos, variables o categorías en cierto contexto. Tiene un valor explicativo.

Con respecto al alcance descriptivo, Hernández et al. (2014) mencionan que este busca especificar las propiedades o características de personas o grupos de estas, procesos, objetos o cualquier otro fenómeno que se someta a un análisis.

Por último, se encuentra el alcance explicativo, este se encarga de determinar las causas de eventos y fenómenos físicos o sociales, además de explicar por qué y cómo ocurre un fenómeno o por qué se relacionan dos o más variables.

Según las definiciones anteriores, el alcance que se ajusta al presente proyecto es el descriptivo, ya que dentro de la definición brindada por los autores se dice que este busca especificar las características de procesos o cualquier otro fenómeno que se someta a un análisis, justo este proyecto está orientado a analizar un proceso dentro de la organización.

3.4. Diseño de la investigación

De acuerdo con Hernández et al. (2014), el diseño de la investigación es un plan o estrategia que se desarrolla para obtener la información que se requiere en una investigación y poder responder al problema.

Los tipos de diseño de investigación para el enfoque cualitativos son:

- Diseños etnográficos: estudian a grupos y sistemas sociales para analizar e interpretar las prácticas presentes en dichos sistemas. También estudia elementos culturales.
- Diseños fenomenológicos: explora y describe las experiencias de las personas con respecto a un fenómeno.
- Diseños investigación-acción: este aborda problemáticas que necesitan resolverse y que pretende lograr un cambio. Busca soluciones específicas.
- Diseños narrativos: se basa en historias sobre procesos, hechos, eventos y experiencias, siguiendo una línea de tiempo.
- Teoría fundamentada: responde a la relación entre categorías de un proceso o fenómeno. Es una teoría que explica un fenómeno o responde al planteamiento del problema.

El diseño de investigación que se adecua al presente proyecto es el investigación-acción, ya que la finalidad de este es brindar una solución a una problemática presente, en este caso en una organización.

3.5. Fuentes de datos e información

Las fuentes de datos son instrumentos para el conocimiento, búsqueda y acceso a la información, pueden ser documentos de varios tipos y brindan datos sobre un tema o área específica, asimismo, según su nivel pueden ser clasificadas en fuentes primarias, secundarias o terciarias.

Según Guzmán (s.f.), las fuentes de información primarias contienen información nueva y original que no ha sido modificada y que son producto de un trabajo intelectual. Además, la información proviene de una fuente directa, sea de una persona, institución o algún otro medio.

Las fuentes de datos primarias que se tendrán en cuenta para el desarrollo del presente proyecto son:

- Entrevistas y cuestionarios.
- Documentación original de la organización.
- Estándares y marcos de referencia de la industria.

Las fuentes de información secundarias permiten conocer hechos o fenómenos a partir de documentos o datos recopilados por otros. Se consideran como una reinterpretación de las fuentes primarias.

De igual forma, se piensa acudir a las siguientes fuentes de datos secundarias:

- Trabajos de graduación similares al actual.
- Artículos científicos disponibles en Internet.
- Libros, revistas y foros.

3.6. Sujetos de investigación

Los sujetos de investigación son aquellos involucrados que tendrán un impacto directo o indirecto en el desarrollo del proyecto.

En la **Tabla 4,** se describen los principales sujetos de investigación presentes, así como alguna información relevante sobre cada uno.

Tabla 4
Sujetos de investigación

Sujeto de investigación	Rol en la organización	Importancia en la investigación
Desarrollador de *software* / Programador	Todo aquel colaborador o colaboradora que se dedica al desarrollo de nuevas funcionalidades o productos de *software*.	Se puede considerar como causante indirecto de la problemática presentada en la organización.
Propietario del producto	Encargado de verificar que los requerimientos que se le brindan se transfieran y entiendan correctamente por el desarrollador de *software*.	Encargado de verificar que el producto de *software* cumpla con las especificaciones indicadas por el negocio. Encargado de verificar que los pases a producción se apliquen de manera correcta.
Supervisores de producto de *software* / Supervisor de Sistemas Pega	Encargados de velar por que se cuente con todas las condiciones necesarias para el desarrollo de nuevos productos según los estándares establecidos por la organización.	Dentro de esta investigación figuran como involucrados directos ya que supervisan que el proyecto se realice de acuerdo con lo requerido por la organización, además de ser

Sujeto de investigación	Rol en la organización	Importancia en la investigación
		parte del Centro de Excelencia Operacional.

3.7. Variables o categorías de la investigación

Las variables de la investigación permiten verificar que el cumplimiento de los objetivos se está logrando, esto con el apoyo de los diversos indicadores. A continuación, en la **Tabla 5** se describen las principales variables que se quieren analizar en la investigación con respecto a cada objetivo específico establecido.

Tabla 5

Variables de la investigación

Objetivo específico	Variable de investigación	Definición conceptual	Indicadores	Técnica de investigación
Analizar la situación actual del proceso de desarrollo y aseguramiento de la calidad de *software* dentro del departamento para el entendimiento de la forma en que se realiza actualmente dicho proceso.	Situación actual del proceso de desarrollo y aseguramiento de la calidad de *software*.	Se hace referencia a la forma en que actualmente los equipos de trabajo realizan el proceso de desarrollo y aseguramiento de la calidad del *software*. Esto incluye el paso a paso, instrumentos utilizados, documentación desarrollada en el proceso y tipos de prueba de *software* realizadas.	• Existencia de pruebas de aseguramiento de calidad. • Proceso actual diagramado.	• Entrevistas. • Observación. • Modelado de procesos BPMN.
	Deficiencias existentes en el proceso actual.	Aspectos que no se están realizando correctamente con respecto al proceso de desarrollo y aseguramiento de calidad del *software*.	• Nivel de conocimiento de los desarrolladores sobre pruebas de *software*. • Nivel de documentación de pruebas. • Nivel de completitud del proceso.	• Entrevistas. • Lentes de Madison

Objetivo específico	Variable de investigación	Definición conceptual	Indicadores	Técnica de investigación
Rediseñar el proceso de desarrollo y aseguramiento de la calidad del *software* en el departamento para la estandarización de este utilizando las buenas prácticas de la industria	Proceso mejorado de desarrollo y aseguramiento de calidad del *software*.	Proceso que contemple las buenas prácticas sugeridas por la industria con respecto al desarrollo y aseguramiento de la calidad del *software*.	• Buenas prácticas de la industria que apliquen al nuevo proceso. • Proceso mejorado diagramado.	• Revisión documental. • Modelado de procesos BPMN.
	Plan de aseguramiento de calidad del *software*.	Documentación sobre aquellas actividades, roles, tipos de pruebas, documentación requerida y otros aspectos que contempla un plan de aseguramiento de la calidad del *software* que complemente el proceso deseado	• Tipos de pruebas requeridas. • Procedimiento y estándares para la realización de pruebas.	• Revisión documental.
Desarrollar una propuesta de implementación del proceso de aseguramiento de calidad del *software* para la gestión de los recursos asociados a este.	Plan de implementación del proceso propuesto de aseguramiento de la calidad del *software*.	Documentación que indique la forma en que se van a gestionar los recursos relacionados con la implementación del plan de aseguramiento de la calidad. Este contempla aspectos como gestión del alcance, tiempo, involucrados, recursos, entre otros.	• Guías para la gestión de proyectos. • Herramientas y técnicas para la gestión de recursos.	• Revisión documental.

Objetivo específico	Variable de investigación	Definición conceptual	Indicadores	Técnica de investigación
	Análisis costo-beneficio.	Análisis que indique la viabilidad de la implementación del proyecto, contemplando los beneficios cuantitativos y cualitativos de este.	• Beneficios de la propuesta. • Costos de implementación esperados.	• Revisión documental. • Entrevistas.

3.8. Técnicas e instrumentos de recolección de datos

En esta sección se indican los instrumentos utilizados en la presente investigación para la recolección de datos para su posterior análisis.

De acuerdo con Machuca (2022), las técnicas de recolección de datos son herramientas que sirven para recopilar información de diferentes fuentes, hacer evaluaciones y tomar mejores decisiones. A continuación, se describen los principales instrumentos utilizados para el proyecto.

3.8.1. Revisión documental

Según Machuca (2022), esta técnica se encuentra dentro del grupo de métodos cualitativos y consiste en recopilar información empleando una revisión de diferentes fuentes documentales. En esta técnica de recolección de datos el investigador debe tener un buen criterio de análisis y selección de fuentes, por ello es importante verificar la validez de los documentos que se revisarán antes de utilizarlos.

Siguiendo con lo expuesto por Machuca (2022), esta técnica emplea los siguientes pasos:

1. Selección de fuentes: recopilación de todo el material que podría ser de interés para el desarrollo de la investigación. Estas fuentes podrían ser documentos impresos, electrónicos, material gráfico y audiovisual.
2. Revisar y organizar la información: una vez que se tengan seleccionadas las fuentes de información, se deben revisar y clasificar los datos relevantes que se encuentren.
3. Análisis de datos: este último paso consiste en analizar el material recopilado y determinar las conclusiones de la investigación.

Para el presente estudio la principal fuente de recopilación de información serán los documentos electrónicos, específicamente libros digitales, revistas electrónicas y artículos de Internet. La plantilla utilizada para realizar revisión documental se muestra en el **Apéndice C. Plantilla de revisión documental.**

3.8.2. Entrevista

De acuerdo con Machuca (2022), esta técnica consiste en recolectar información utilizando una serie de preguntas que ayudan al entrevistador a conocer aspectos como el comportamiento y preferencias del entrevistado.

Existen diversas maneras de realizar una entrevista, esta puede ejecutarse de forma individual o grupal, de manera presencial o virtual y utilizando medios como llamadas telefónicas o videollamadas. Existen varios tipos de entrevista, Machuca (2022) identifica las siguientes:

- Entrevista estructurada: siguen un formato previamente establecido.

- Entrevista semiestructurada: en esta las preguntas siguen un orden preestablecido, pero puede variar en el transcurso de la entrevista, además, el entrevistador puede agregar preguntas si lo considera necesario.
- Entrevista no estructurada: la estructura de esta no es preestablecida y las preguntas dependen del criterio del entrevistador.

En el **Apéndice B. Plantilla de entrevista**, se muestra la plantilla que se utilizará para esta técnica de recolección de información.

3.8.3. Observación

De acuerdo con Díaz (2023), la técnica de observación es una herramienta para recopilar información de manera objetiva. Esta permite generar conocimiento amplio sobre un tema en particular y puede ser utilizada para realizar estudios de campo, investigar comportamientos, entender mejor los problemas y sus causas, conocer las condiciones y necesidades específicas de un grupo, entre otros.

Asimismo Díaz (2023), menciona que existen los siguientes tipos de observación:

- Observación estructurada: se centra en la recolección de datos mediante el uso de instrumentos con una lista predefinida de elementos o preguntas. Esta ofrece un mayor control sobre la recolección de datos, ya que el observador está limitado a los elementos listados, lo que a su vez permite tener datos precisos.
- Observación no estructurada: se refiere a la recolección de datos basada en los juicios y las opiniones del observador. La ventaja de este tipo de observación es la flexibilidad y el nivel de compresión que permite del tema.

La plantilla para la documentación de observación se encuentra en el **Apéndice D. Plantilla de observación.**

3.9. Matriz de cobertura de las variables

En la **Tabla 6** se encuentra representada la forma en que se distribuyen los diferentes instrumentos de recolección de datos para cada una de las variables propuestas.

Tabla 6
Matriz de cobertura de las variables

Variable de investigación	Entrevista	Revisión documental	Modelado BPMN	Observación
Situación actual del proceso de desarrollo y aseguramiento de la calidad de *software*.	X		X	X

Variable de investigación	Entrevista	Revisión documental	Modelado BPMN	Observación
Deficiencias existentes en el proceso actual.	X	X		
Proceso mejorado de desarrollo y aseguramiento de calidad del *software*.	X	X	X	
Plan de aseguramiento de calidad del *software*.	X	X		
Plan de implementación del plan de aseguramiento de la calidad.	X	X		
Análisis costo- beneficio.	X	X		

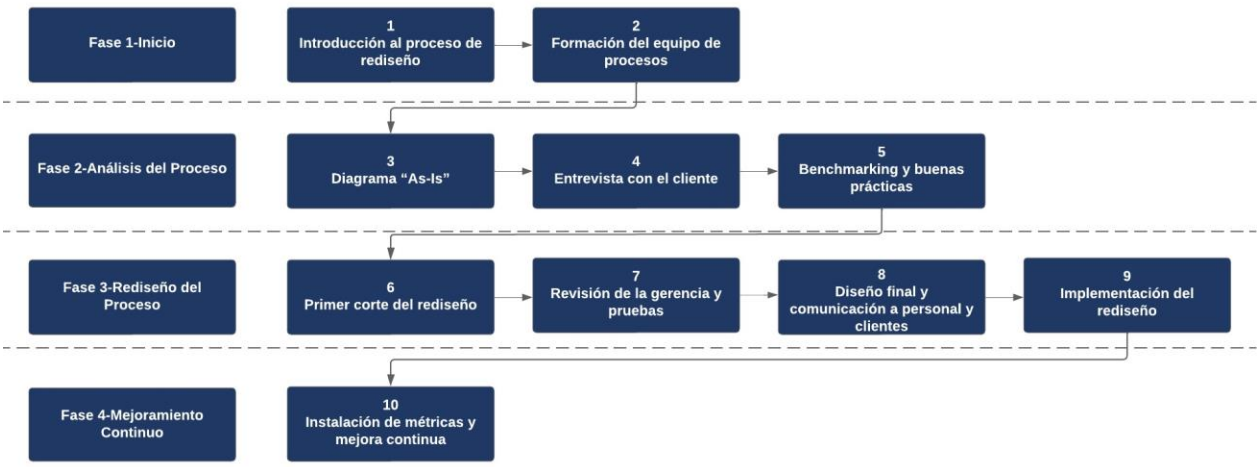
3.10. Procedimiento metodológico de la investigación

El procedimiento metodológico brinda una guía de pasos o enfoques estructurados que deben de seguirse con el objetivo de abordar un problema de forma sistemática y efectiva. Este debe permitir la realización de un trabajo ordenado y que sea repetible en situaciones similares.

De acuerdo con la naturaleza de este proyecto, se necesita una metodología que permita abordar el rediseño de procesos, para ello se evalúan dos propuestas brindadas por Marlon Dumas y Dan Madison. Aunque las dos metodologías presentan principios similares, se decide optar por la metodología de Dan Madison del año 2005 (ver **Figura 10**), esto debido a que al tratarse de la mejora de un solo proceso se adecua al contexto presentado en el proyecto, sin embargo, se realizarán algunas adaptaciones necesarias.

Figura 10

Metodología de Reingeniería de Procesos de Dan Madison



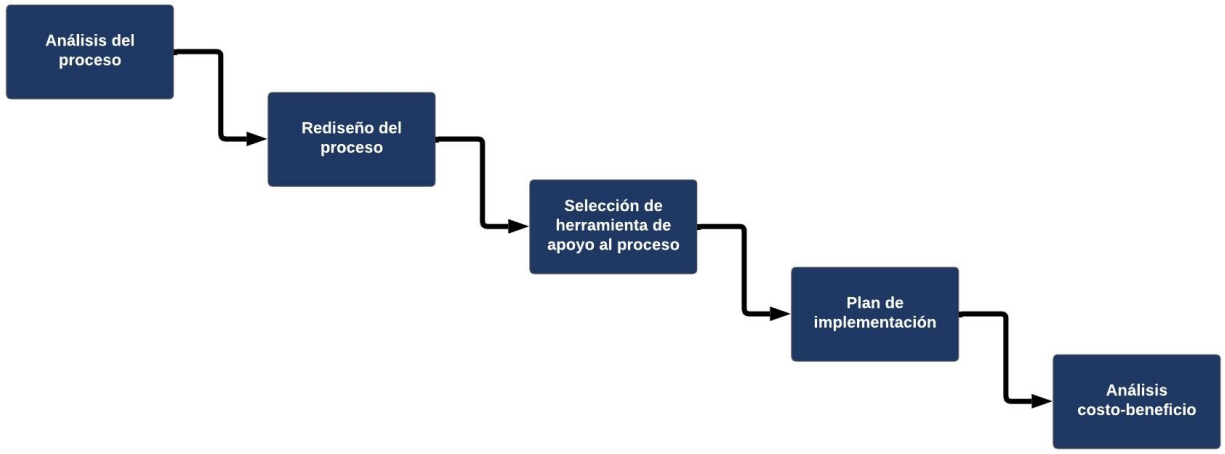
Nota. Adaptado de *Process Mapping, Process Improvement and Process Management: A Practical Guide to Enhancing Workflow and Information Flow*, por Madison, 2005, Paton Professional.

El proyecto está constituido por cinco fases principales, en donde las primeras dos fases están relacionadas y se apoyarán con la metodología de rediseño de procesos seleccionada, en donde se analizará el proceso actual y se redefinirá el proceso deseado con los puntos de mejora encontrados.

La fase tres se relaciona con el desarrollo del plan de aseguramiento de calidad del *software* para el departamento, siguiendo con la fase cuatro orientada a definir la forma en que se pondrá en marcha el proyecto a través del plan de implementación. Finalmente, se encuentra la fase cinco, en esta se realizará el análisis costo-beneficio del proyecto.

En la **Figura 11** se muestra a un alto nivel el procedimiento metodológico del proyecto a nivel de fases de ejecución.

Figura 11
Fases de desarrollo del proyecto



3.10.1. Fase 1: Análisis del proceso

Mediante esta fase se plantea describir de qué forma el proceso de desarrollo y aseguramiento de la calidad del *software* se realiza actualmente en la organización y se relaciona directamente con el objetivo específico uno del proyecto. Para esta primera fase se hace una relación directa con los puntos tres y cuatro de la metodología de rediseño de procesos propuesta por Madison.

A continuación, se detallan las actividades correspondientes a esta fase.

3.10.1.1. Diagramación del proceso *as-is*

En este punto se representa el estado actual del proceso al que se le aplicará el rediseño y se asocia al paso tres de la metodología de Madison. Para realizar el diagrama del proceso *as-is* se utilizará el estándar conocido como Notación de Modelado de Procesos (BPMN), en su versión 2.0. Para ello se utilizarán técnicas como entrevistas y observación para diagramar y documentar el proceso.

3.10.1.2. Identificación de puntos de mejora

Según Madison (2005), este paso se apoya en los denominados lentes de procesos, estos permiten realizar un diagnóstico de los procesos para que se le pueda brindar el enfoque adecuado al nuevo proceso que será elaborado.

El autor define tres lentes que pueden ser analizados, según el objetivo del proyecto se realizará un análisis del proceso desde la perspectiva de la calidad que se relaciona con la entrega de valor.

Para la aplicación de este lente de calidad, se procederá a realizar entrevistas al supervisor de producto de *software* y al supervisor de sistemas Pega, como resultado se elaborará un diagrama de Ishikawa para plasmar las causas directas con respecto a la problemática de calidad del proceso e identificar los puntos de mejora.

3.10.2. Fase 2: Rediseño del proceso

El objetivo final de esta fase es diseñar y modelar el nuevo proceso que contendrá las mejoras identificadas con respecto a lo que dictan las mejores prácticas de la industria, con respecto al desarrollo y aseguramiento de la calidad del *software*.

3.10.2.1. Revisión de buenas prácticas

Esta actividad tiene relación directa con el paso cinco realizado en la metodología de rediseño propuesta por Madison (2005), en este paso lo que se busca es evaluar y comparar buenas prácticas presentes en la industria con el objetivo de aplicarlas al nuevo proceso que será planteado.

3.10.2.2. Diseño del proceso *to-be*

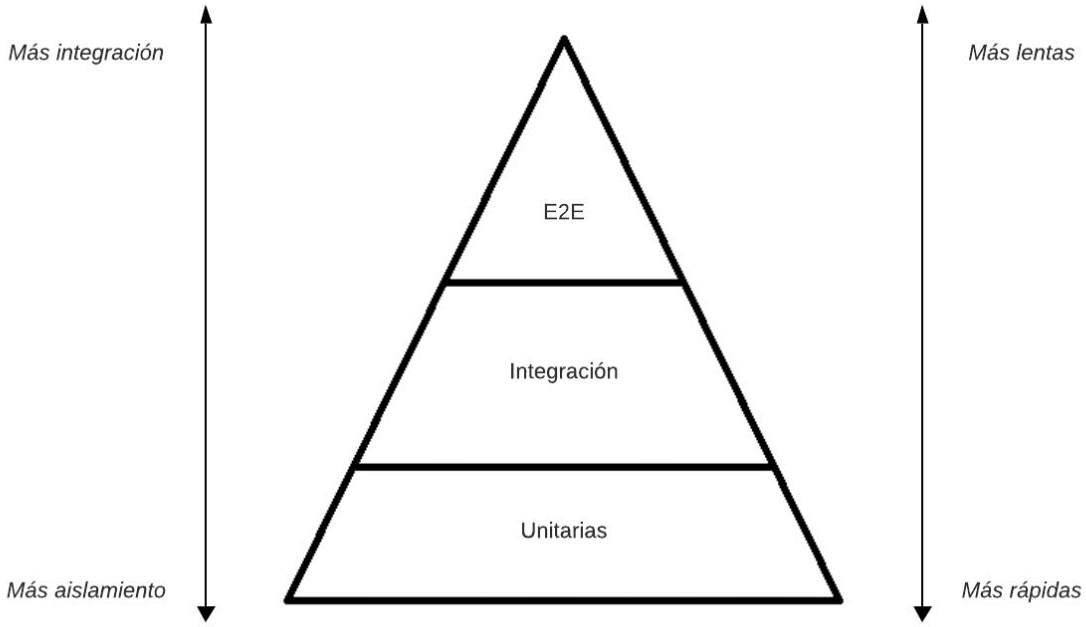
Este paso consiste en representar la propuesta del proceso rediseñado en un diagrama en notación BPMN 2.0 para posteriormente presentarlo a los interesados, con el objetivo de obtener su visto bueno y/o retroalimentación.

Con esta actividad se cumplen con los pasos seis y siete de la metodología propuesta por Madison (2005).

3.10.2.3. Plan de aseguramiento de calidad del *software*

En este punto se elabora la documentación sobre el proceso de desarrollo y aseguramiento de calidad del *software* propuesto. El nuevo proceso de aseguramiento de la calidad del *software* se centrará en la ejecución de pruebas de *software*. Dichas pruebas de *software* contemplarán tres tipos principales, según Cohn (2009), un plan de pruebas efectivo consta de tres niveles de pruebas, las pruebas unitarias, las pruebas de integración y las pruebas *end-to-end* (pp. 311-312). En la **Figura 12** se plasma la idea brindada por Cohn (2009).

Figura 12
Pirámide de pruebas



Nota: Adaptado de *Succeeding with Agile: Software Development Using Scrum* (p.312), por Mike Cohn, 2009, Addison-Wesley Professional.

3.10.3. Fase 3: Selección de herramienta de apoyo al proceso

El objetivo de esta fase es seleccionar una herramienta de *software* que sirva como apoyo para complementar el nuevo proceso de desarrollo y aseguramiento de calidad del *software*. Esta fase se divide en los siguientes puntos.

3.10.3.1. Recopilación de requerimientos

En este punto se procede a recopilar los principales requerimientos con que debe cumplir la herramienta de gestión de casos de prueba de *software*.

3.10.3.2. Priorización de requerimientos

Una vez recopiladas las principales características que debe reunir la herramienta, se deben priorizar dichas características utilizando el método de priorización de requerimientos *MoSCoW*.

3.10.3.3. Investigación de herramientas disponibles

En este punto se realiza una investigación de herramientas de *software* disponibles en el mercado, para la gestión de pruebas de *software*, que puedan cumplir con los requerimientos.

3.10.3.4. Selección final de la herramienta

Como último punto dentro de esta fase se indica realizar un evaluación final de las herramientas preseleccionadas y proceder a seleccionar la herramienta final que será propuesta como complemento al proceso mejorado de desarrollo y aseguramiento de la calidad del *software*.

3.10.4. Fase 4: Plan de implementación

Esta fase tiene una relación directa con el paso siete de la metodología propuesta por Madison, aunque en este caso tiene variación, ya que este alude a la implementación del nuevo proceso, pero por cuestiones de tiempo y alcance del proyecto no se realizará la implementación como tal, sino que se elaborará un plan de implementación por fases que permita sobrellevar el tema de resistencia al cambio.

Para llevar a cabo la elaboración del plan de implementación se utilizará como guía lo indicado en el PMBOK, específicamente en su guía de los fundamentos para la dirección de proyectos, tomando como referencia su séptima edición.

De los ocho dominios de desempeño mencionados para el proyecto, se tomarán en cuenta los siguientes, esto con el fin de gestionar aspectos como el tiempo, los involucrados, el alcance, el riesgo y comunicación. En la **Tabla 7** se indican los principales dominios que se abarcarán en el plan de implementación del nuevo proceso de desarrollo y aseguramiento de la calidad del *software*.

Tabla 7
Dominios de desempeño del proyecto

Dominio de desempeño	Descripción
Interesados	Se ocupa de las actividades y funciones asociados con los interesados. Implica trabajar con los interesados para mantener la alineación, relaciones positivas y satisfacción.

Dominio de desempeño	Descripción
Planificación	Organiza, elabora y coordina el trabajo del proyecto a lo largo de la totalidad de este. Aborda las actividades y funciones necesarias para la entrega de los resultados del proyecto.
Incertidumbre	Se ocupa de las actividades y funciones asociadas con el riesgo y la incertidumbre.

Nota. Tomado de *Guía de los fundamentos para la dirección de proyectos* (pp.7-129), por Project Management Institute, 2021, Project Management Institute, Inc.

3.10.5. Fase 5: Análisis costo-beneficio del proyecto

Esta última fase evalúa si es factible la implementación del proyecto en términos financieros para la organización.

3.10.5.1. Determinación de costos e ingresos

Este punto consiste en realizar un análisis para identificar los principales costos que genera la implementación del proyecto. Una vez identificados los costos se les asignará un valor monetario. De igual forma se realizará un análisis similar para los ingresos, con el fin de plasmar ambos en términos monetarios.

3.10.5.2. Cálculo de indicadores financieros

En esta sección se realiza el cálculo aproximado del retorno sobre la inversión del proyecto así como del periodo de recuperación de la inversión para saber si el proyecto es rentable, en términos financieros, para la organización.

3.11. Operacionalización de las variables o categorías

Este apartado es de utilidad para comprender aspectos como conceptos, indicadores, técnicas y herramientas que permitan entender cómo se realizará la presente investigación y con qué herramientas se podrán abarcar las variables establecidas para cada objetivo específico del proyecto.

Toda esta información se encuentra resumida en la **Tabla 8**.

Tabla 8

Operacionalización de las variables de investigación

Objetivo específico	Fase del procedimiento metodológico	Variable de investigación	Instrumento utilizado	Sujetos de investigación
Analizar la situación actual del proceso de desarrollo y aseguramiento de la calidad de *software* dentro del departamento para el entendimiento de la forma en que se realiza actualmente dicho proceso.	Fase 1: Análisis del proceso	Situación actual del proceso de desarrollo y aseguramiento de la calidad de *software*. Deficiencias existentes en el proceso actual.	Apéndice B. Plantilla de entrevista Apéndice C. Plantilla de revisión documental Herramienta de modelado de procesos.	• Supervisor de producto de *software*. • Desarrolladores de *software*. • Supervisor de sistemas Pega.
Rediseñar el proceso de desarrollo y aseguramiento de la calidad del *software* en el departamento para la estandarización de este utilizando las buenas prácticas de la industria.	Fase 2: Rediseño del proceso	Proceso mejorado de desarrollo y aseguramiento de calidad del *software*. Plan de aseguramiento de calidad del *software*.	Apéndice B. Plantilla de entrevista. Apéndice C. Plantilla de revisión documental. Herramienta de modelado de procesos.	• Supervisor de producto de *software*. • Supervisor de sistemas Pega.
	Fase 3: Selección de herramienta de apoyo al proceso	Herramientas para la gestión de casos de prueba de *software*.		

Objetivo específico	Fase del procedimiento metodológico	Variable de investigación	Instrumento utilizado	Sujetos de investigación
Desarrollar una propuesta de implementación del proceso de aseguramiento de calidad del *software* para la gestión de los recursos asociados a este	Fase 4: Plan de implementación Fase 5: Análisis costo-beneficio	Plan de implementación del plan de aseguramiento de la calidad. Análisis costo-beneficio.	Apéndice C. Plantilla de revisión documental.	• Supervisor de producto de *software*. • Supervisor de sistemas Pega.

3.12. Tabla resumen del procedimiento metodológico o trazabilidad

En la **Tabla 9**, se muestra la matriz de trazabilidad del procedimiento metodológico establecido para el proyecto.

Tabla 9

Matriz de trazabilidad

Objetivo específico	Procedimiento metodológico	Análisis de resultados	Propuesta de solución	Conclusiones	Recomendaciones
Analizar la situación actual del proceso de desarrollo y aseguramiento de la calidad de *software* dentro del departamento para el entendimiento de la forma en que se realiza actualmente dicho proceso.	3.10.1.1 Diagramación del proceso as-is 3.10.1.2 Identificación de puntos de mejora	4.1.1 Diagrama del proceso *as-is* 4.1.2 Identificación de puntos de mejora	No aplica	Capítulo 6: Conclusiones (Ver conclusiones para Objetivo específico 1)	Capítulo 7: Recomendaciones
Rediseñar el proceso de desarrollo y aseguramiento de la calidad del *software* en el departamento para la estandarización de este utilizando las buenas prácticas de la industria.	3.10.2.1 Revisión de buenas prácticas 3.10.2.2 Diseño del proceso to-be 3.10.2.3 Plan de aseguramiento de calidad del software 3.10.3.1 Recopilación de requerimientos	4.2.1 Revisión de buenas prácticas de la industria 4.2.2 Diseño del proceso *to-be* 4.2.3 Plan de aseguramiento de la calidad del *software* 4.3.1 Recopilación de requerimientos 4.3.2 Priorización de requerimientos	5.1.1 Diseño del proceso *to-be* 5.1.2 Plan de aseguramiento de calidad del *software* 5.2.1 Selección final de la herramienta	Capítulo 6: Conclusiones (Ver conclusiones para Objetivo específico 2)	Capítulo 7: Recomendaciones

Objetivo específico	Procedimiento metodológico	Análisis de resultados	Propuesta de solución	Conclusiones	Recomendaciones
	3.10.3.2 Priorización de requerimientos 3.10.3.3 Investigación de herramientas disponibles 3.10.3.4 Selección final de la herramienta	4.3.3 Investigación de herramientas disponibles			
Desarrollar una propuesta de implementación del proceso de aseguramiento de calidad del *software* para la gestión de los recursos asociados a este.	3.10.4 Fase 4: Plan de implementación 3.10.5.1 Determinación de costos e ingresos 3.10.5.2 Cálculo de indicadores financieros	4.4 Fase 4: Plan de implementación 4.5 Fase 5: Análisis de costo-beneficio del proyecto	5.3 Fase 4: Plan de implementación 5.4.1 Determinación de costos e ingresos 5.4.2 Cálculo de indicadores financieros	Capítulo 6: Conclusiones (Ver conclusiones para Objetivo específico 3)	Capítulo 7: Recomendaciones

Una vez descrita la metodología que se empleará para el proyecto, se prosigue con el siguiente capítulo correspondiente al análisis de resultados y síntesis de la información obtenida al aplicar los diversos instrumentos diseñados.

4. Capítulo 4: Análisis de Resultados

En el presente capítulo de análisis de resultados se detallan los instrumentos descritos en el marco metodológico para la recolección de información. Una vez aplicados los instrumentos se deben sintetizar los datos recopilados para facilitar la compresión de la información.

Como se indicó anteriormente, las principales técnicas utilizadas para la recolección de la información fueron las entrevistas, la revisión documental y la observación, aplicándolas a los principales involucrados del proceso de desarrollo y aseguramiento de la calidad del *software* dentro del departamento de sistemas banca y procesos, esto con el fin de recopilar información sobre el proceso actual y las opciones de mejora para el proceso planteado, por ende para la propuesta de solución.

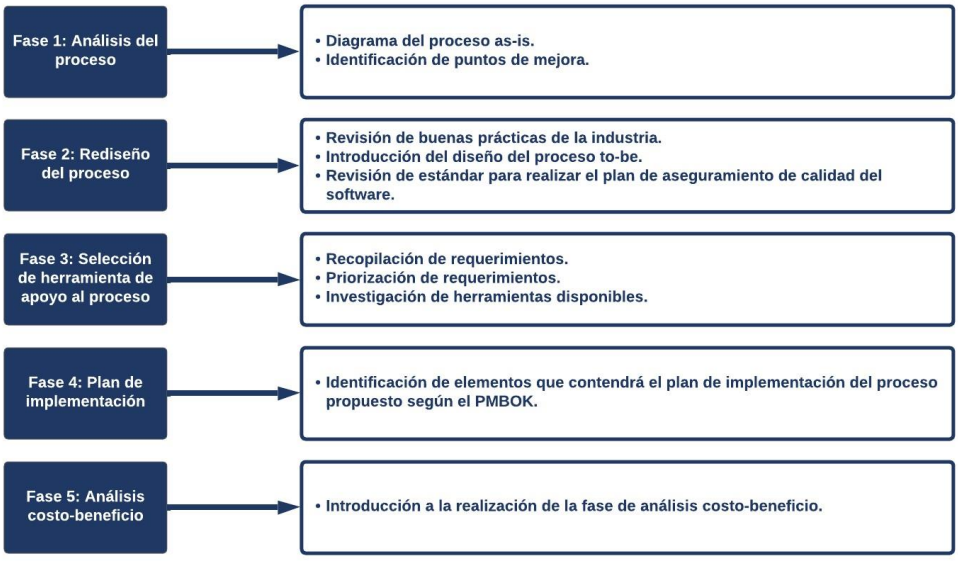
A continuación se presentan las actividades realizadas para la recolección de información para las fases del capítulo:

- Entrevistas al supervisor de sistemas Pega.
- Entrevistas al supervisor de producto de *software*.
- Entrevistas a desarrolladores de *software*.
- Encuestas a desarrolladores.
- Observación del proceso de desarrollo y aseguramiento de la calidad del *software*.
- Revisión documental de buenas prácticas para el desarrollo de pruebas de *software*.

En la **Figura 13** se muestran los contenidos que serán abarcados en este capítulo de acuerdo con las fases establecidas en el **Procedimiento metodológico de la investigación.**

Figura 13

Contenido abarcado en el capítulo 4 según el procedimiento metodológico



4.1. Fase 1: Análisis del proceso

El objetivo de esta fase es describir la forma en que el proceso de desarrollo y aseguramiento de calidad del *software* se realiza actualmente, para ello se identifica cada una de las actividades realizadas así como los roles que intervienen en dicho proceso.

Una vez que se tienen los elementos descritos anteriormente y se entiende cómo se realiza el proceso, se procede a efectuar el diagrama *as-is* utilizando la notación BPMN 2.0.

4.1.1. Diagrama del proceso *as-is*

Para conocer el estado actual del proceso dentro del departamento y poder representarlo a través de un diagrama en notación BPMN 2.0 es necesario aplicar primeramente una serie de entrevistas con los involucrados directos del proceso. El resultado de estas entrevistas se describe a continuación.

4.1.1.1. Entrevista a supervisor de sistemas Pega

Esta fue una entrevista de tipo estructurada en la que se busca entender el funcionamiento del proceso desde el punto de vista de este supervisor y comparar su intervención con el resto de los involucrados.

En la **Tabla 10**, se muestran las preguntas realizadas en esta entrevista.

Tabla 10
Preguntas dirigidas a supervisor de sistemas Pega

N° Pregunta	Pregunta
1	¿Cómo cree que se realiza el proceso de desarrollo y QA de *software* actualmente en los equipos? (En cada sprint, desde el *planning* hasta cuando se llevan las funcionalidades a QA).
2	¿Existe un proceso de QA estandarizado en el departamento?, (es decir, cada equipo realiza las mismas actividades de la misma forma, por ejemplo, tipos de pruebas realizadas, escenarios cubiertos, documentación, etc.)
3	¿Cuáles son los principales roles que forman parte del ciclo de vida de desarrollo y aseguramiento de la calidad?
4	¿Cuáles actividades relacionadas con aseguramiento de la calidad cree que realizan los equipos?
5	¿Cuáles cree que son los principales puntos de dolor que enfrenta la organización/departamento con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*?

N° Pregunta	Pregunta
6	¿Existe un departamento que se especialice en la gestión de la calidad del *software*, en este caso en Pega?
7	¿Cómo se podrían "alivianar" esos puntos de dolor con respecto al tema de QA del *software*?

Las respuestas obtenidas al aplicar esta entrevista se encuentran en el **Apéndice J. Entrevista 6.**

Los principales hallazgos obtenidos fueron los siguientes:

- Las principales actividades realizadas dentro del proceso de desarrollo y aseguramiento de la calidad del *software* son: planificación del *sprint*, inicio del *sprint*, inicio de desarrollo de las funcionalidades requeridas en cada historia de usuario, despliegue de las funcionalidades hacia el ambiente de QA, revisión de las historias por parte del dueño del producto (*Product owner*), despliegue de funcionalidades al ambiente de *Staging*, pruebas en *Staging*, despliegue al ambiente de producción.
- No existe un proceso estandarizado de aseguramiento de calidad de *software*, pero se está buscando implementar un proceso estandarizado. A largo plazo se estaría buscando la automatización del proceso de pruebas y también en la creación de un departamento de QA.
- Los principales roles involucrados en el proceso de desarrollo y aseguramiento de la calidad del *software* son el dueño del producto (*Product owner)*, el Scrum Máster, el líder técnico del equipo y los desarrolladores.
- No existe ningún rol asociado directamente al aseguramiento de calidad del *software.* Tampoco existe un departamento o equipo de trabajo dedicado exclusivamente al aseguramiento de la calidad del *software.*
- Las principales actividades de aseguramiento de calidad que realizan los desarrolladores de *software* son pruebas unitarias, pruebas de aceptación y pruebas de certificación *end-to-end*.

4.1.1.2. Entrevista a supervisor de producto de *software*

De igual forma, para entender el proceso actual es importante conocer el punto de vista que tiene el supervisor de producto de *software*, para ello se aplica la misma entrevista que se efectuó al supervisor de sistemas Pega. En la **Tabla 11**, se muestran las preguntas realizadas en esta entrevista.

Tabla 11
Preguntas dirigidas al supervisor de producto de software

N° Pregunta	Pregunta
1	¿Cómo cree que se realiza el proceso de desarrollo y QA de *software* actualmente en los equipos? (En cada sprint, desde el *planning* hasta cuando se llevan las funcionalidades a QA).
2	¿Existe un proceso de QA estandarizado en el departamento?, (es decir, cada equipo realiza las mismas actividades de la misma forma, por ejemplo, tipos de pruebas realizadas, escenarios cubiertos, documentación, etc.)
3	¿Cuáles son los principales roles que forman parte del ciclo de vida de desarrollo y aseguramiento de la calidad?
4	¿Cuáles actividades relacionadas con aseguramiento de la calidad cree que realizan los equipos?
5	¿Cuáles cree que son los principales puntos de dolor que enfrenta la organización/departamento con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*?
6	¿Existe un departamento que se especialice en la gestión de la calidad del *software*, en este caso en Pega?
7	¿Cómo se podrían "alivianar" esos puntos de dolor con respecto al tema de QA del *software*?

En el **Apéndice K. Entrevista 7** se muestran las respuestas obtenidas al aplicar este instrumento. A continuación, se sintetiza el contenido obtenido.

- El proceso de desarrollo de *software* es estandarizado, no así las actividades relacionadas con el aseguramiento de la calidad del *software*.
- No existe un proceso estandarizado de aseguramiento de la calidad del *software*, pero existe la necesidad de trabajar en uno.
- Indicó que los principales roles involucrados en el proceso de desarrollo y aseguramiento de la calidad de *software* pertenecen a los desarrolladores y el propietario del producto (*product owner*).
- Actualmente los equipos solo realizan pruebas de sistema y de aceptación.

4.1.1.3. Entrevista a desarrolladores

Estas ayudan a describir un enfoque preciso sobre el proceso de desarrollo y aseguramiento de la calidad del *software*. Esta entrevista se intentó aplicar a desarrolladores de diversos equipos

de desarrollo de *software* en Pega para obtener una perspectiva completa del proceso, se detalla en la **Tabla 12**.

Tabla 12
Preguntas dirigidas a desarrolladores de software

N° Pregunta	Pregunta
1	¿Podría describir el proceso de desarrollo que realiza el equipo?
2	¿Qué actividades de aseguramiento de la calidad del *software* considera que se realizan dentro del equipo?
3	¿Existe algún manual o estándar en la organización para realizar las pruebas de *software* o saber cómo se deben realizar?
4	¿Realizan documentación de las funcionalidades y pruebas realizadas?
5	¿Quién se encarga de diseñar los diversos tipos de prueba realizados?
6	¿Cómo determina la forma en que se deben realizar las pruebas?
7	Según su conocimiento, ¿Qué tipos de pruebas de *software* cree que se realizan en el equipo?
8	¿Qué mejoras realizaría al proceso de desarrollo y aseguramiento de calidad actual?
9	Del 1 al 5, donde 1 es poca y 5 mucha, ¿Actualmente cuánta importancia se le da al proceso de QA en la organización?
10	De los siguientes tipos de pruebas de *software*: Pruebas unitarias, pruebas de integración, pruebas *end-to-end*, ¿Conoce el alcance de cada una de ellas?
11	¿Cree que se debe incrementar el conocimiento en la organización/departamento sobre el tema de QA?
12	¿Alguna observación adicional?

Las respuestas obtenidas al aplicar esta entrevista se encuentran en los siguientes apéndices:

Apéndice E. Entrevista 1, Apéndice F. Entrevista 2, Apéndice G. Entrevista 3, Apéndice H. Entrevista 4, Apéndice I. Entrevista 5.

Los principales hallazgos obtenidos fueron los siguientes:

- Las principales actividades en común que se realizan en los equipos con respecto al proceso de desarrollo y aseguramiento de *software* son:
    - Recopilación de requerimientos de negocio.
    - Creación del *backlog*.
    - Refinamiento de las historias.

- Planeación del *sprint*.
- Inicio del *sprint*.
- Desarrollo de historias de usuario.
- Pruebas unitarias.
- *Code review*.
- Despliegue de las funcionalidades al ambiente de QA.
- Pruebas en QA.
- Pruebas de aceptación.
- Documentación de las historias.
- Despliegue de las funcionalidades al ambiente de *Staging*.
- Pruebas en el ambiente de *Staging*.
- Despliegue de las funcionalidades al ambiente de producción.

Es importante indicar que estas actividades se realizan de forma secuencial según su orden en la lista anterior.

Adicionalmente se obtuvieron los siguientes resultados una vez aplicadas las entrevistas a los desarrolladores de *software*.

- Como se muestra en la **Figura 14**, el 100% de los entrevistados indicó que no existe un estándar para realizar los diversos tipos de pruebas de *software* necesarios para garantizar el correcto funcionamiento de las funcionalidades desarrolladas.

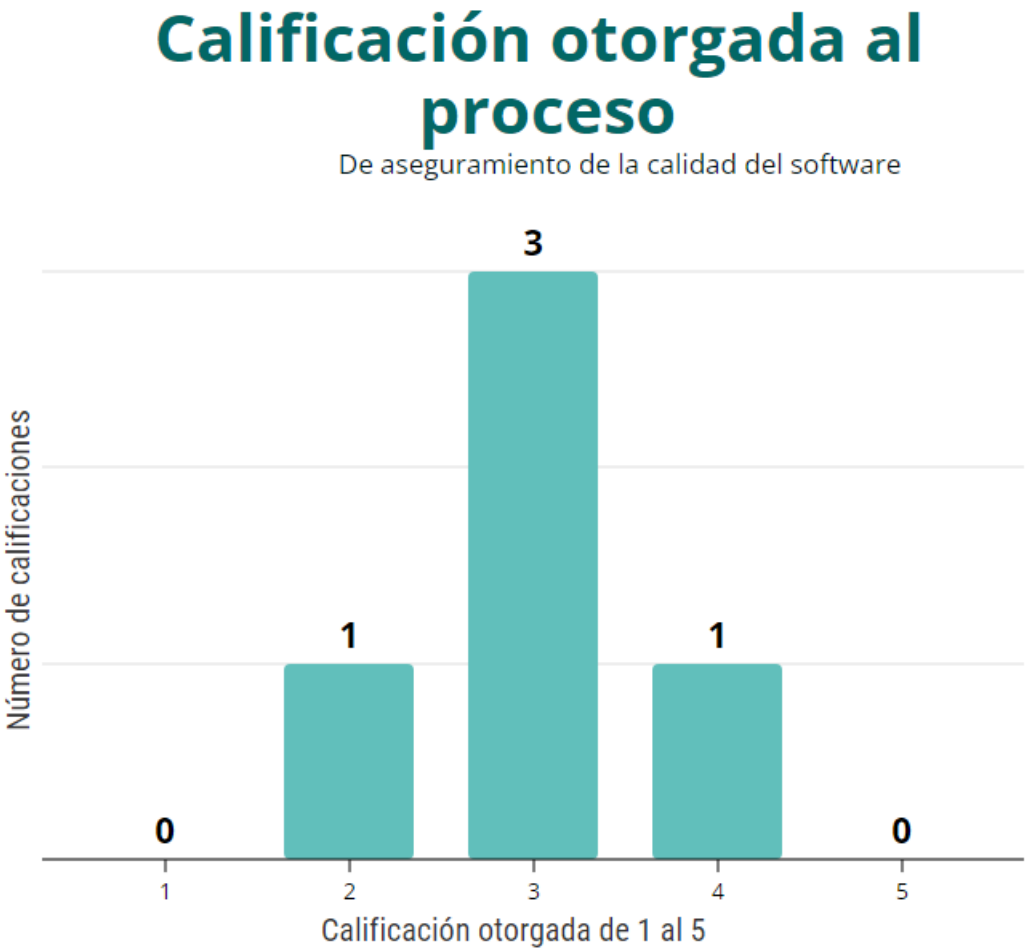
Figura 14

Resultados sobre existencia de un estándar para prueba

- En promedio, los desarrolladores de *software* le asignaron una calificación de 3 puntos al proceso de aseguramiento de la calidad realizado en el departamento, en donde un 1 representa la calificación más baja y 5 la calificación más alta posible, esto se evidencia en la **Figura 15**.

Figura 15

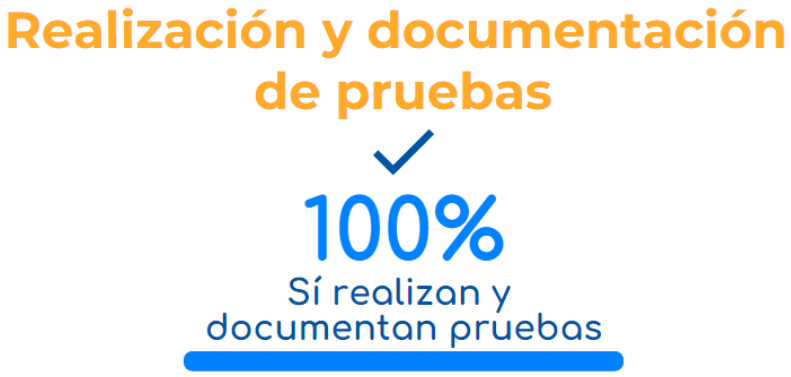
Resultados sobre calificación brindada al proceso de QA



- Como se aprecia en la **Figura 16**, el 100% de los entrevistados indicó que sí realizan diversos tipos de pruebas de *software* y documentan dichas pruebas. De igual forma, la totalidad de los desarrolladores indicó que ellos mismos son los encargados de diseñar los diversos tipos de prueba que creen convenientes.

Figura 16

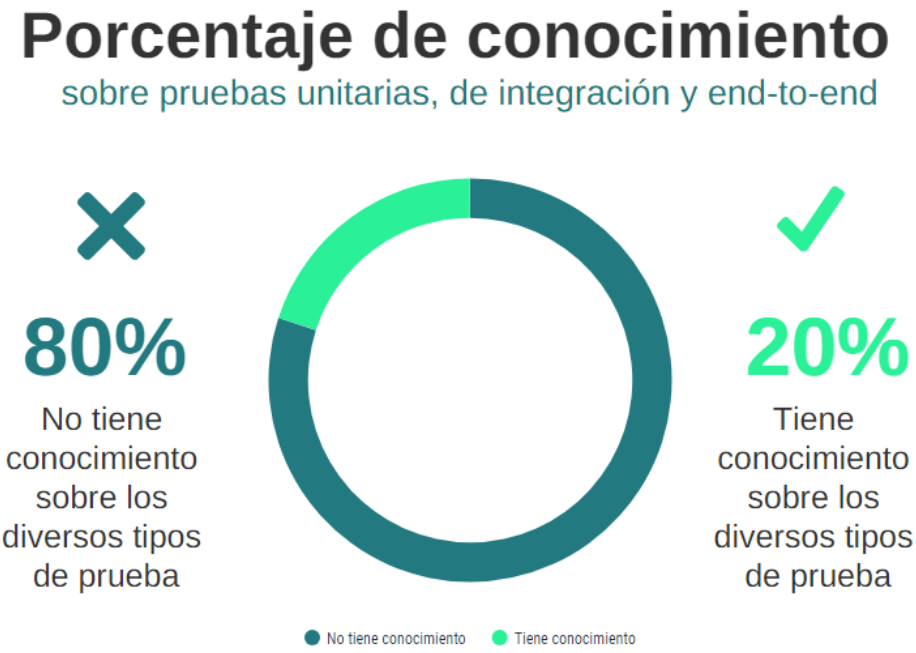
Resultados sobre realización de pruebas de software



- El 80% de los entrevistados indicó que no tienen total claridad sobre el alcance que tienen las pruebas unitarias, las pruebas de integración y las pruebas *end-to-end*. Esto se muestra en el gráfico presentado en la **Figura 17**. Además, el 100% de los desarrolladores indicó que debe aumentar el nivel de conocimiento con respecto al tema de QA en el departamento.

Figura 17

Resultados sobre conocimiento de tipos de pruebas



4.1.1.4. Observación del proceso

Adicionalmente a las entrevistas se utilizó la técnica de observación para representar el proceso *as-is* de desarrollo y aseguramiento de la calidad del *software* como parte de las recomendaciones brindadas en el **Apéndice V. Minuta 5**. A través de esta técnica se logró comprender y describir la manera en que se realiza el proceso desde otra perspectiva, además, se logró complementar lo descrito anteriormente por los participantes entrevistados. A continuación, se mencionan los principales roles y actividades ejecutadas durante el proceso, esto servirá para complementar el entendimiento del diagrama *as-is* del proceso.

En la **Tabla 13** se describen los principales involucrados que se identificaron en el proceso.

Tabla 13
Involucrados en el proceso

Involucrado	Funciones realizadas
Propietario del producto (*Product owner)*	• Recopilar los requerimientos por parte del área de negocio. • Priorizar y administrar el *backlog* del equipo. • Creación de las historias de usuario. • Gestionar el ciclo de vida del producto en general. • Presidir los refinamientos. • Realizar pruebas de aceptación con los desarrolladores.
Desarrolladores de *software*	• Elaborar las diferentes funcionalidades requeridas. • Realizar las pruebas de *software* de las funcionalidades desarrolladas. • Participar en las sesiones de planeación del *sprint.* • Participar en las sesiones de refinamiento del equipo.
Líder técnico	• Brindar apoyo a todos los compañeros del equipo. • Documentar y enviar los pases a solicitud de aprobación para el despliegue en el ambiente de producción. • Realizar el *code review*. • Realizar las labores descritas para los desarrolladores de *software*. • Participar en las sesiones del *chapter* de arquitectura.
Scrum Master	• Gestionar la comunicación con los otros equipos. • Eliminar los impedimentos que interfieran en el trabajo del equipo. • Organizar las diversas reuniones requeridas por el equipo. • Presidir la reunión de retrospectiva del equipo al final del *sprint*.

Involucrado	Funciones realizadas
COE	• Equipo habilitador para los equipos de desarrollo de *software*. • Aprobar o rechazar el despliegue de funcionalidades críticas hacia el ambiente de producción. • Apoyar a los equipos en temas relacionados a la mejora continua e implementación de buenas prácticas con respecto al desarrollo de *software.*

En la **Tabla 14**, se describen las actividades que fueron identificadas en el proceso. Es importante indicar que en dicha tabla se encuentran las actividades que se consideran relevantes en el proceso y que fueron identificadas a través de la observación (Ver **Apéndice L Primera observación del proceso actual** y **Apéndice M. Segunda observación del proceso actual**).

Tabla 14
Actividades desarrolladas en el proceso

Actividad	Descripción
Recopilación de requerimientos de negocio.	En esta actividad el *product owner* se encarga de recopilar los requerimientos descritos por el usuario de negocio.
Creación del *backlog*	El *product owner* se encarga de crear las historias de usuario e incluirlas dentro del *backlog* del equipo.
Refinamiento de las historias	Esta es una sesión en la que participan el *product owner* y los desarrolladores de *software* en la que se analizan las historias de usuario del *backlog* y se les asigna la puntuación correspondiente a la estimación del esfuerzo que implicaría desarrollarlas.
Planeación del *sprint*	Esta es una actividad que se realiza con la participación del *product owner*, los desarrolladores y el *scrum master* en la que se organizan las historias de usuario que formarán parte del *sprint*. A cada desarrollador se le asignan las historias de usuario necesarias.
Desarrollo de historias de usuario	Una vez iniciado el *sprint*, cada desarrollador se encarga de elaborar las funcionalidades correspondientes a las historias de usuario asignadas.

Actividad	Descripción
Pruebas unitarias	Una vez codificadas las funcionalidades, el desarrollador realiza pruebas para verificar que se cumpla con lo solicitado en la historia de usuario asignada.
Code review	Esta es una revisión de las funcionalidades que se realiza entre el líder técnico del equipo y el desarrollador. El líder técnico con su criterio de experto aprueba o brinda opciones de mejora al código elaborado.
Despliegue de las funcionalidades al ambiente de QA	Las funcionalidades codificadas por los desarrolladores son organizadas en un *release* que viaja al ambiente de QA.
Pruebas en QA	Una vez que las funcionalidades desarrolladas se encuentran en al ambiente de QA, se deben realizar las pruebas de *software* en dicho ambiente. Esta actividad la realiza el desarrollador.
Pruebas de aceptación	Cuando el desarrollador certifica que las pruebas en QA han sido satisfactorias, procede a realizar las pruebas de aceptación con el *product owner*.
Documentación de las historias	Una vez obtenido el visto bueno en las pruebas de aceptación, el desarrollador debe documentar las funcionalidades realizadas junto con la evidencia de las pruebas. En este punto se adjunta la documentación a la historia de usuario y se cambia el estado de la historia a finalizada.
Despliegue de las funcionalidades al ambiente de *Staging*	Esta actividad consiste en pasar las funcionalidades que se encuentran en el ambiente de QA al ambiente de *staging* para proceder a realizar las pruebas en dicho ambiente.
Pruebas en el ambiente de *Staging*	De igual forma en que se hizo en los ambientes de desarrollo y QA, se deben realizar las mismas pruebas en el ambiente de *staging* para certificar la correctitud de las funcionalidades desarrolladas.

Actividad	Descripción
Despliegue de las funcionalidades al ambiente de producción	En este paso se implementan las funcionalidades desarrolladas por el equipo en el *release* al ambiente productivo.

Asimismo, las principales herramientas que intervienen en el proceso de desarrollo y aseguramiento de la calidad del *software* se describen en la **Tabla 15**.

Tabla 15

Herramientas utilizadas en el proceso

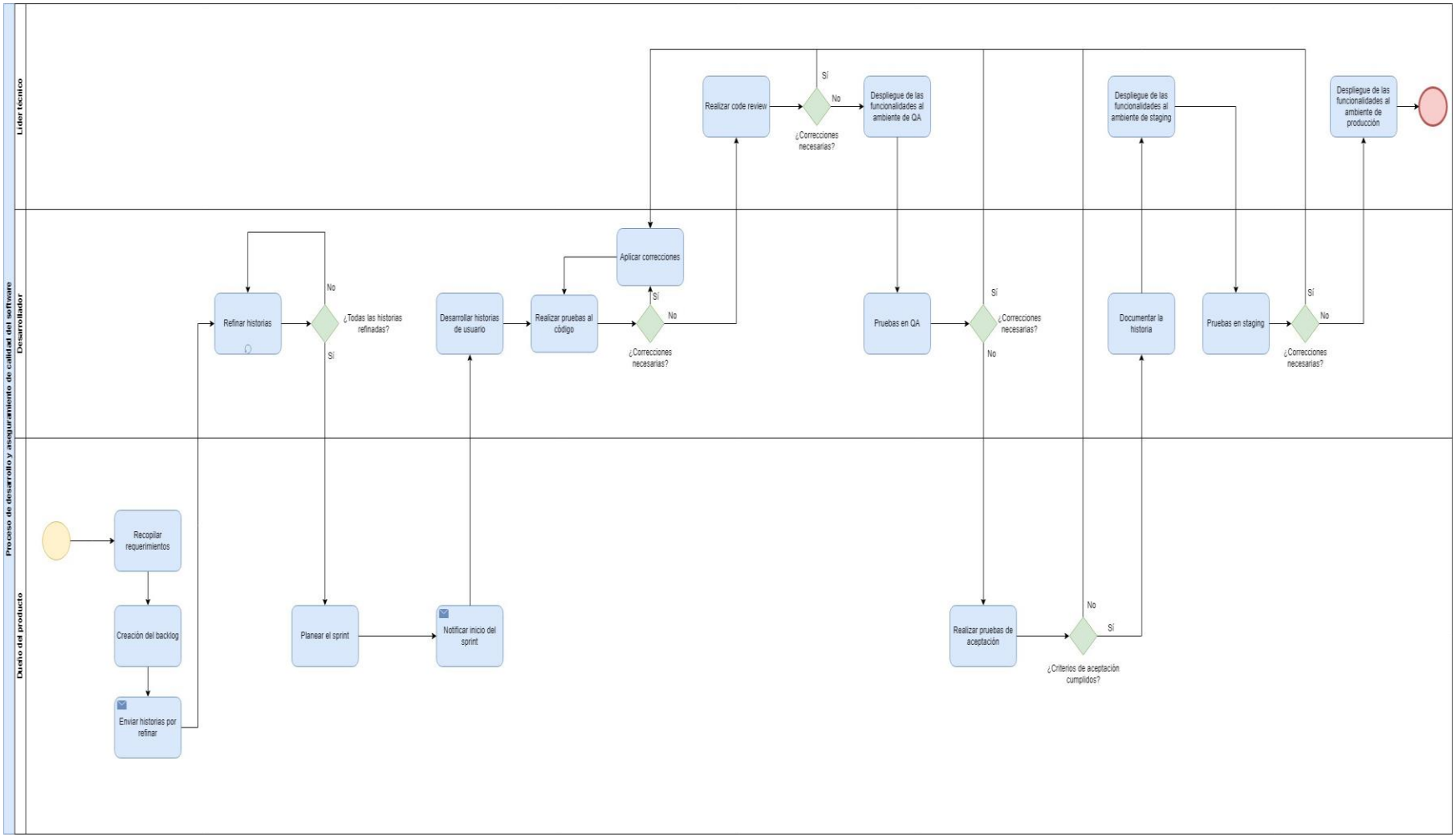
Herramienta	Uso
Pega	Esta es la herramienta de desarrollo de tipo *low code* que utilizan los desarrolladores para programar las diversas funcionalidades.
Jira	En esta herramienta se gestiona el backlog con las historias pendientes del equipo y los *releases* relacionados con cada *sprint.*
Confluence	Utilizada para generar y adjuntar documentación con respecto a los proyectos en general. Funciona como una base de conocimiento para los involucrados dentro del proceso.

Una vez aplicados los diversos instrumentos para conocer el estado actual del proceso de desarrollo y aseguramiento de la calidad del *software*, se procede a realizar el diagrama *as-is* del proceso, intentando representar la totalidad de las actividades descritas por los diferentes involucrados a través de las entrevistas, además del proceso de observación descrito en el punto **Observación del proceso.**

A continuación, en la **Figura 18**, se muestra el diagrama del proceso *as-is* de desarrollo y aseguramiento de la calidad del *software* elaborado, tomando como insumo los resultados de las técnicas aplicadas anteriormente.

Figura 18

Diagrama as-is del proceso de desarrollo y aseguramiento de la calidad del software



Para complementar la información mostrada en el diagrama anterior, en la **Tabla 16** se describen las diversas actividades que forman parte de este proceso.

Tabla 16
Descripción de las actividades del proceso as-is

Actividad	Descripción
Recopilar requerimientos	El dueño del producto realiza sesiones con los involucrados por parte del negocio para recopilar las necesidades de estos en forma de requerimientos.
Creación del backlog	El dueño del producto crea las historias de usuario con base en los requerimientos recopilados en las sesiones con la parte del negocio.
Enviar historias por refinar	El dueño del producto informa sobre las historias de usuario que se van a refinar con el objetivo de incluirlas en los próximos *sprints*.
Refinar historias	Los desarrolladores verifican los criterios de aceptación de cada historia del *backlog* para estimar los puntos de esfuerzo que tendrá la historia. Es necesario indicar que estos puntos estimados podrían cambiar una vez finalice el desarrollo de la historia.
Planear el sprint	El dueño del producto incluye en el *sprint* algunas historias de usuario registradas en el *backlog* del equipo.
Notificar inicio del sprint	El dueño del producto (*Product Owner)* notifica al equipo de desarrollo sobre el inicio del *sprint* planificado anteriormente.
Desarrollar historias de usuario	En esta actividad se inicia con el desarrollo de las funcionalidades para cumplir con los criterios de aceptación de la historia.
Realizar pruebas al código	Se realizan pruebas a las funcionalidades desarrolladas según el conocimiento que tengan los desarrolladores con respecto a las pruebas de *software* y a los escenarios de prueba que ellos determinen necesarios.
Aplicar correcciones	El desarrollador debe realizar correcciones al código en caso de que las funcionalidades

Actividad	Descripción
	desarrolladas no cumplieran con el objetivo deseado.
Realizar code review	El líder técnico del equipo realiza una revisión del código producido por el desarrollador de *software* para validar si puede desplegarse en los siguientes ambientes (QA, *staging* y desarrollo).
Despliegue de las funcionalidades al ambiente de QA	Se envían las funcionalidades desarrolladas al ambiente de QA.
Pruebas en QA	Se realizan pruebas de las funcionalidades en el ambiente de QA.
Realizar pruebas de aceptación	Se realizan pruebas con el dueño del producto (*Product Owner)* para verificar que las funcionalidades desarrolladas cumplen con los criterios de aceptación de la historia.
Documentar la historia	El desarrollador realiza la documentación de la historia a través de un documento que contiene varias secciones como lo son la descripción de la historia y las evidencias de cambios realizados.
Despliegue de las funcionalidades al ambiente de staging	Se envían las funcionalidades desarrolladas al ambiente de *staging*.
Pruebas en staging	Se realizan pruebas de las funcionalidades en el ambiente de *staging*.
Despliegue de las funcionalidades al ambiente de producción	Se envían las funcionalidades desarrolladas al ambiente de producción.

4.1.2. Identificación de puntos de mejora

Dentro de los aspectos que menciona Madison (2005), como parte de la metodología de rediseño de procesos se puede optar por aplicar tres tipos de lentes para el diagnóstico del proceso, estos son el lente de frustración, el lente del tiempo y el lente de calidad.

Para el proceso de desarrollo y aseguramiento de la calidad del *software* se aplicará el lente de calidad, ya que en este se verifican problemáticas relacionadas con la forma en que se realizan los procesos de inicio a fin para obtener los resultados esperados. Con este análisis se pretende obtener puntos de mejora que se le puedan aplicar al nuevo proceso planteado.

Para realizar este análisis se listan una serie de problemas o defectos relacionados con la calidad del proceso, para ello, se utiliza como insumo los resultados una vez efectuadas las reuniones con los supervisores de producto y supervisor de sistemas Pega (ver **Apéndice R. Minuta 1** y **Apéndice S. Minuta 2**), además las que se efectuaron a los desarrolladores de *software.*

En la **Tabla 17** se encuentra el listado de defectos recopilados.

Tabla 17
Lista de defectos del proceso

ID del defecto	Defecto
01	Los desarrolladores omiten escenarios de pruebas.
02	No existe un proceso estandarizado de aseguramiento de la calidad del *software.*
03	Los desarrolladores se encargan de diseñar las pruebas y verificar la calidad del *software*.
04	El proceso de aseguramiento de calidad es totalmente empírico.
05	No existen plantillas para documentación de pruebas.
06	Los desarrolladores independientemente del equipo al que pertenezcan realizan documentación de sus historias y pruebas de una forma distinta.
07	El nivel de conocimiento de los desarrolladores con respecto a temas de aseguramiento de la calidad es muy limitado.
08	No existe un departamento dedicado a la gestión de la calidad del *software*.

Es importante mencionar que, de los defectos descritos anteriormente, uno de los críticos es la omisión de escenarios de pruebas, principalmente si al enviar las funcionalidades al ambiente de producción no se han probado correctamente los escenarios es posible que alguno de ellos se reproduzca y cause gran afectación a la operativa de la organización.

A continuación, en la **Tabla 18** se realiza un ordenamiento de los defectos según su criticidad y los efectos que representan para la organización.

Tabla 18

Impacto de los defectos encontrados

Defecto	Efecto	Impacto
No existe un departamento dedicado a la gestión de la calidad del software.	No hay un criterio de un equipo experto con respecto a la calidad del *software* producido, por ende, nadie comprueba que se esté produciendo *software* defectuoso.	Alto, ya que es necesario que la organización cuente con un departamento o equipo que verifique la calidad del *software* producido.
Los desarrolladores omiten escenarios de pruebas.	Se omiten escenarios de prueba que se pueden presentar en el ambiente productivo, lo cual afectará la operativa de la organización.	Alto, debido a que puede afectar directamente la calidad de los servicios brindados.
Los desarrolladores se encargan de diseñar las pruebas y verificar la calidad del software.	Este defecto se relaciona con el defecto 01, ya que probablemente si un desarrollador no tiene mucho conocimiento sobre tipos de pruebas de *software* este vaya a omitir posibles escenarios de prueba.	Alto, se pueden omitir algunos tipos de prueba necesarios para cumplir con el aseguramiento del *software*, además que no es correcto que los propios desarrolladores cumplan con el rol que debería ejecutar el departamento de QA.
El nivel de conocimiento de los desarrolladores con respecto a temas de aseguramiento de la calidad es muy limitado.	Es perjudicial debido a que los mismos desarrolladores son los encargados realizar el proceso de aseguramiento de la calidad del *software*, y si su nivel de conocimiento del tema es bajo, probablemente el nivel de pruebas realizado sea superficial o no sea el correcto.	Alto, ya que se necesita tener el conocimiento necesario para ejecutar un proceso tan importante como lo es el aseguramiento de la calidad del *software*. Esto podría repercutir negativamente en la satisfacción del cliente.
No existe un proceso estandarizado de aseguramiento de la calidad del software.	No hay una base para o guía para realizar el proceso, por ende, este será entendido y ejecutado de forma diferente por los desarrolladores de cada uno de los equipos.	Medio, ya que la organización debería de guiarse por estándares que dicten la forma en que se realizan los procesos y sean

Defecto	Efecto	Impacto
		comprendidos de la misma forma por todos.
El proceso de aseguramiento de calidad es totalmente empírico.	Esto significa que el proceso de aseguramiento de la calidad del *software* es entendido de forma diferente por cada desarrollador, por ende, probablemente los tipos de pruebas ejecutados sean diferentes o se omitan tipos de pruebas.	Medio, aunque actualmente el proceso es empírico, sí hay evidencia de que es realizado por los desarrolladores. Podría afectar a la organización igualmente en la calidad del *software* enviado a producción.
No existen plantillas para documentación de pruebas.	Dificulta y retrasa la agilidad con la que los desarrolladores ejecutan el proceso de desarrollo y aseguramiento de la calidad del *software*.	Bajo, no representa una amenaza directa a la operativa de la organización.
Los desarrolladores independientemente del equipo al que pertenezcan realizan documentación de sus historias y pruebas de una forma distinta.	No se contribuye al objetivo que tiene la organización con respecto a la estandarización de sus procesos.	Bajo, de igual forma no afecta la forma en que se brindan los productos o servicios a los clientes.

En la **Figura 19** se muestra un diagrama de Ishikawa que permite visualizar de forma resumida las causas de estas problemáticas organizadas según su categoría.

Figura 19
Resumen del lente de calidad

Una vez realizado este análisis se tiene un insumo sobre las principales necesidades o puntos de mejora que deben satisfacerse en el nuevo proceso de desarrollo y aseguramiento de calidad del *software.*

4.2. Fase 2: Rediseño del proceso

El objetivo de esta fase es llevar a cabo una serie de actividades para poder diseñar y representar las mejoras que se le realizarán al proceso de desarrollo y aseguramiento de calidad del *software*. Para ello, una de las actividades importantes es la revisión documental para la evaluación de estándares y buenas prácticas relacionadas con el proceso de aseguramiento de calidad del *software*, así como para el desarrollo del *software* en general.

Es importante indicar que una de las actividades que se llevará a cabo en esta fase es la evaluación de las buenas prácticas y estándares disponibles en la industria, una vez realizado este punto, se procede a elaborar el nuevo proceso de desarrollo y aseguramiento de la calidad del *software*. También se realiza un análisis sobre una serie de herramientas que son útiles y representan un complemento para el nuevo proceso planteado.

4.2.1. Revisión de buenas prácticas de la industria

En esta actividad se aplica la técnica de revisión documental (ver **Apéndice N. Resultado de revisión documental #1, Apéndice O. Resultado de revisión documental #2, Apéndice P. Resultado de revisión documental #3, Apéndice Q. Resultado de revisión documental #4**) para recopilar información sobre lo que algunos estándares dictan como lo esencial que debe de contener un proceso de aseguramiento de la calidad del *software*.

Primeramente y como recomendación recibida se decide aplicar al proceso *to-be* una fase relacionada directamente con el proceso de aseguramiento de la calidad del *software*. Esta fase es la relacionada con el proceso de ingeniería de requerimientos de *software*, que es donde debe iniciar el proceso de aseguramiento de calidad.

Para ello, en el punto **2.19 Proceso de ingeniería de requerimientos de software** se especifican las principales actividades involucradas en este proceso. Según los autores, el proceso de ingeniería de requerimientos de *software* sigue de forma general las siguientes actividades:

- Obtención y análisis de requisitos
- Especificación de requerimientos
- Priorización de requerimientos
- Verificación y validación de requisitos

De acuerdo con diversos autores, existe una fase en el proceso de aseguramiento de la calidad del *software* relacionada con el diseño del *software*, pero por cuestiones de alcance del proyecto se excluye esta fase del proceso *to-be* previsto. Esta observación se encuentra en la sección de **Exclusiones del proyecto**, además se realiza una recomendación sobre este punto en la sección **Capítulo 7: Recomendaciones**.

Siguiendo con las actividades visualizadas en el proceso *to-be* se encuentra el proceso de pruebas de aseguramiento de calidad del *software*. Existen varios enfoques que se describirán de forma resumida a continuación. Es importante indicar que la explicación completa de cada uno de estos procesos y sus actividades se encuentran en el marco conceptual, específicamente en el punto **2.20 Proceso de pruebas de aseguramiento de calidad del software.**

- Proceso de aseguramiento de la calidad del *software* según el SWEBOK

Según el SWEBOK (2014), las actividades de este proceso son las siguientes:

  - Planeación
  - Generación de casos de prueba
  - Preparación del ambiente de pruebas
  - Ejecución
  - Evaluación de los resultados de las pruebas
  - Reporte de problemas/ Registro de pruebas
  - Seguimiento de defectos

- Proceso de aseguramiento de la calidad del *software* según Jeff Tian

Tian (2005), indica en su libro *Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement*, que existen las siguientes actividades principales relacionadas con el proceso de aseguramiento de *software* y el proceso de pruebas, estas actividades son:

  - Planeación y preparación de las pruebas
  - Ejecución y medición de pruebas
  - Análisis de resultados de pruebas y actividades de seguimiento
  - Automatización de pruebas
- Proceso de aseguramiento de la calidad del *software* según Thomas Hamilton

De acuerdo con Hamilton (2023), y lo descrito en una publicación en su sitio web Guru99.com existen seis fases que forman parte del ciclo de vida de pruebas en el *software*, estas fases son las siguientes:

  - Análisis de requerimientos
  - Planificación de las pruebas
  - Desarrollo de los casos de prueba
  - Configuración del ambiente de pruebas
  - Ejecución de las pruebas
  - Cierre del ciclo de pruebas
- Proceso de aseguramiento de la calidad del *software* según Spillner et al.

De acuerdo con Spillner et al. (2014), las principales actividades del proceso de aseguramiento de la calidad del *software* son:

  - Planificación y control de las pruebas
  - Análisis y diseño de las pruebas
  - Implementación y ejecución de las pruebas
  - Evaluación y reporte de las pruebas
  - Actividades de cierre de las pruebas

4.2.2. Diseño del proceso *to-be*

Esta actividad se encuentra explicada en el capítulo cinco pues se considera que forma parte de la propuesta de solución del proyecto. En esa sección se detallan todos los aspectos relacionados con el proceso planteado de desarrollo y aseguramiento de la calidad del *software.*

4.2.3. Plan de aseguramiento de la calidad del *software*

De acuerdo con W&B Asset Studio (2023), un Plan de Aseguramiento de la Calidad del *Software* (en adelante SQAP, por sus siglas en inglés *Software Quality Assurance Plan*) es un documento para el proceso de desarrollo de *software* que detalla cómo se gestionará la calidad del

software durante todo el ciclo de vida de un proyecto. Este plan sirve como guía para todo el equipo de desarrollo, delineando estrategias, procesos y actividades necesarias para garantizar que el *software* cumpla con los estándares de calidad deseados.

Para desarrollar el Plan de Aseguramiento de Calidad del *Software* se utiliza lo propuesto por el estándar IEEE Std 730-1998, para planes de aseguramiento de calidad de *software*, este estándar contiene los siguientes contenidos:

- Propósito (Sección 1 del SQAP).
- Documentos de referencia (Sección 2 del SQAP).
- Gestión (Sección 3 del SQAP).
- Documentación (Sección 4 del SQAP).
- Estándares, prácticas, convenciones y métricas (Sección 5 del SQAP).
- Revisiones y auditorías (Sección 6 del SQAP).
- Pruebas (Sección 7 del SQAP).
- Reporte de problemas y acciones correctivas (Sección 8 del SQAP).
- Herramientas, técnicas y metodologías (Sección 9 del SQAP).
- Código de control (Sección 10 del SQAP).
- Control de medios (Sección 11 del SQAP).
- Control de proveedores (Sección 12 del SQAP).
- Recopilación, mantenimiento y retención de registros (Sección 13 del SQAP).
- Entrenamiento (Sección 14 del SQAP).
- Gestión del riesgo (Sección 15 del SQAP).

El desarrollo del Plan de Aseguramiento de Calidad de *Software* se desarrolla en el capítulo V correspondiente a la propuesta de solución del proyecto (Ver **Plan de aseguramiento de calidad del *software***).

4.3. Fase 3: Selección de herramienta de apoyo al proceso

Una de las recomendaciones brindadas por diversos autores con respecto al proceso de aseguramiento de la calidad del *software* se refiere a utilizar una herramienta para la gestión de los casos de prueba, específicamente para facilitar las fases de diseño, ejecución y reporte de las pruebas.

Debido a esto, se decide realizar una reunión con el supervisor de sistemas Pega para evaluar la idea de optar por la adquisición de una herramienta de este tipo y recopilar las necesidades y requerimientos que se deben tomar en cuenta para el proceso de selección de la herramienta.

A continuación, se realizan las actividades correspondientes con la selección de la herramienta para la gestión de casos de prueba.

4.3.1. Recopilación de requerimientos

Para seleccionar la herramienta de gestión de casos de prueba es necesario contar con una serie de requerimientos que permitan delimitar la cantidad de herramientas por evaluar. Para recopilar los requerimientos con que debe cumplir la herramienta de *software* se realizó una reunión con el supervisor de sistemas Pega, la cual se encuentra documentada en el **Apéndice T. Minuta 3.** Los requerimientos recopilados se pueden consultar en la **Tabla 19**.

Tabla 19
Requerimientos de la herramienta de software

ID Requerimiento	Descripción
Req1	La herramienta debe permitir crear casos de prueba.
Req2	La herramienta debe permitir ejecutar casos de prueba manualmente.
Req3	La herramienta debe permitir crear conjuntos de pruebas.
Req4	La herramienta debe poder integrarse con Jira.
Req5	La herramienta debe ser preferiblemente de versión gratuita.
Req6	La herramienta debe permitir la escalabilidad con respecto al número de usuarios permitidos.
Req7	La herramienta debe permitir la autenticación mediante inicio de sesión único (*Single Sign-On*).
Req8	La herramienta debe permitir asociar casos de prueba a historias de usuario.

4.3.2. Priorización de requerimientos

Una vez recopilados los requerimientos que debe cumplir la nueva herramienta se debe proceder a priorizarlos para definir los criterios esenciales para la selección de la nueva herramienta de gestión de casos de prueba.

Para esta priorización se utiliza la metodología de clasificación de requerimientos MoSCoW. En la **Tabla 20** se muestra la priorización de los requerimientos para la selección de la herramienta.

Tabla 20
Priorización de requerimientos para la herramienta

ID Requerimiento	Descripción	Priorización
Req1	La herramienta debe permitir crear casos de prueba.	*Must*
Req2	La herramienta debe permitir ejecutar casos de prueba manualmente.	*Must*
Req3	La herramienta debe permitir crear conjuntos de pruebas.	*Must*
Req4	La herramienta debe poder integrarse con Jira.	*Must*
Req5	La herramienta debe ser preferiblemente de versión gratuita.	*Should*
Req6	La herramienta debe permitir la escalabilidad con respecto al número de usuarios permitidos.	*Must*
Req7	La herramienta debe permitir la autenticación mediante inicio de sesión único (*Single Sign-On*).	*Should*
Req8	La herramienta debe permitir asociar casos de prueba a historias de usuario.	*Should*

Una vez realizada la priorización de los requerimientos se procede a identificar las herramientas disponibles para ser evaluadas.

4.3.3. Investigación de herramientas disponibles

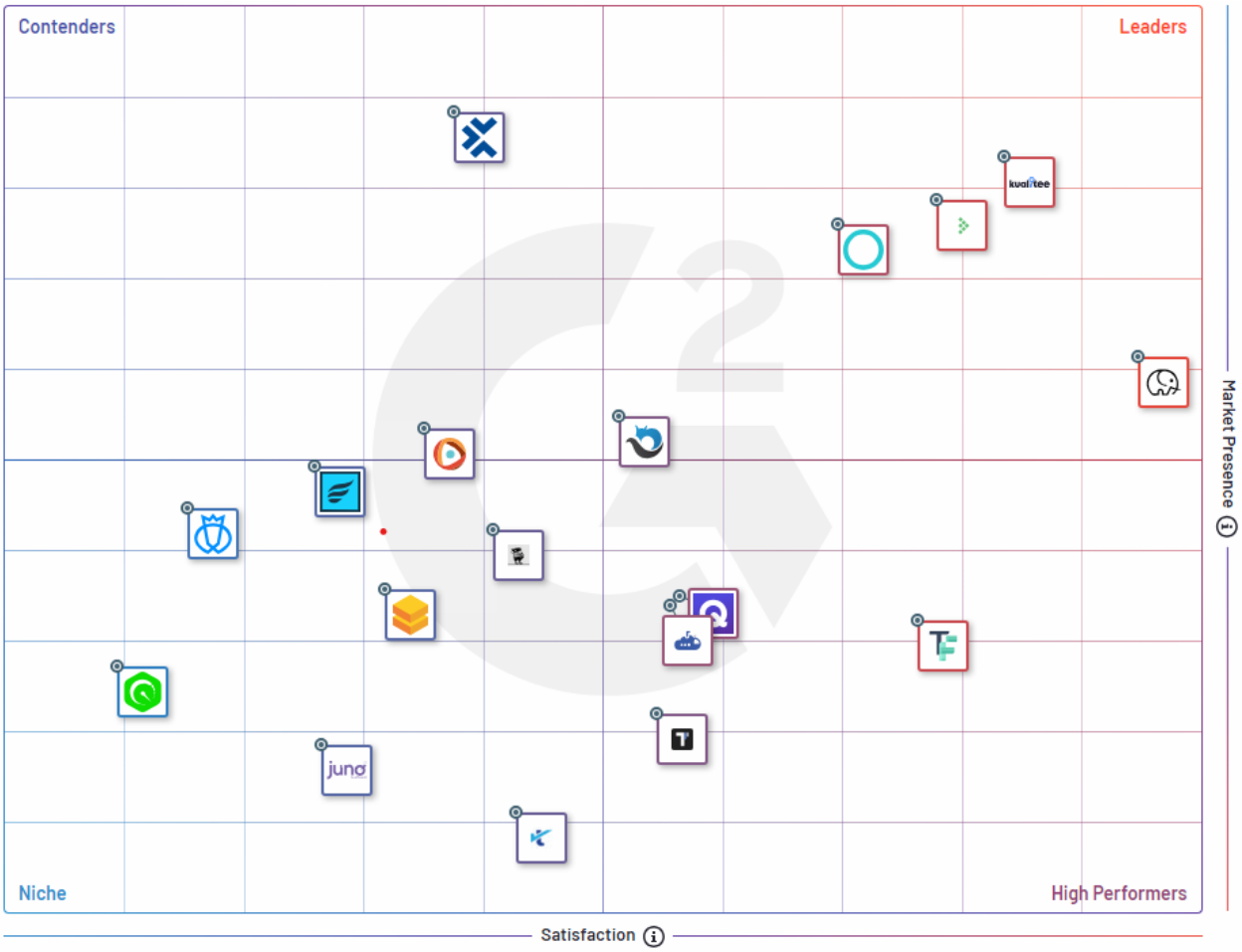
Para este punto se procede a realizar la investigación documental con el fin de determinar cuáles herramientas de gestión de casos de prueba están disponibles en el mercado para ser evaluadas. Para ello se revisan diversas fuentes para tener más disponibilidad de opciones.

Según G2 (s.f.), las tres mejores herramientas para la gestión de casos de prueba son las siguientes: Tuskr, Kualitee y TestRail

Estas se posicionan en el cuadrante de herramientas líderes con respecto a la gestión de pruebas. En la **Figura 20** se observa el cuadrante de G2 para herramientas de gestión de pruebas para el año 2023.

Figura 20

Cuadrante de G2 para herramientas de gestión de pruebas



Nota. Tomado de *Best Test Management Tools for Medium-Sized Businesses*, por G2, s.f.

También se toman en cuenta las siguientes herramientas disponibles en el *Marketplace* de Atlassian para la gestión de pruebas, estas se evalúan debido a que permiten la integración directa con Jira. Para la selección de estas herramientas se tomaron en cuenta dos filtros principales:

1. Las mejores calificadas.
2. Que permitan el alojamiento en la nube

De acuerdo con lo anterior las herramientas que se evaluarán son:

- AIO Tests
- AgileTest

Por lo tanto, se define la siguiente lista de herramienta de gestión de pruebas que se evaluarán:

- Tuskr
- Kualitee
- TestRail
- AIO Tests
- AgileTest

La evaluación y selección final de la herramienta se realizará como parte del capítulo de propuesta de solución, específicamente en la **Fase 3: Selección de herramienta de apoyo al proceso**.

4.4. Fase 4: Plan de implementación

El plan de implementación del proyecto contiene la gestión de una serie de aspectos relacionados con la gestión de proyectos, según lo establecido por el PMBOK. En este plan se gestionan una serie de recursos básicos que generalmente los proyectos contemplan y que forman parte directa del proceso de desarrollo y aseguramiento de la calidad del *software*.

A continuación, se mencionan los principales aspectos que contempla el plan de implementación del proyecto.

- Interesados

Los interesados son aquellos individuos, grupos u organizaciones que pueden verse afectados por una decisión, actividad o resultado de un proyecto. Las dos principales actividades relacionadas con este dominio son la identificación de los interesados y el análisis de estos.

A continuación, en la **Tabla 21** se muestran los principales interesados identificados para el proyecto.

Tabla 21
Interesados del proyecto

Interesado	Impacto	Expectativas
Desarrollador de *software*	Alto	Brindar información sobre las actividades realizadas en el proceso de desarrollo y aseguramiento de *software.* Desarrollar las diversas funcionalidades de *software* requeridas.
Líder técnico	Medio	Encargado de apoyar el proceso de desarrollo y especialmente el de aseguramiento de la calidad del

Interesado	Impacto	Expectativas
		software a través del asesoramiento en la creación de casos de prueba.
Proveedor de herramienta de gestión de casos de prueba	Medio	Encargado de distribuir las licencias para la herramienta de gestión de casos de prueba.
Supervisor de sistemas Pega	Medio	Brindar el visto bueno sobre el proceso planteado y trabajar en su mejora continua.
Analista de QA	Alto	Encargado de apoyar de forma general en el proceso de aseguramiento de la calidad del *software*.
Propietario del producto (*Product owner*)	Medio	Responsable de gestionar los requerimientos de negocio y las historias de usuario asociados a estos.

Las diferentes categorías de impacto de los involucrados en el proceso se describen en la **Tabla 22**.

Tabla 22
Impacto de los involucrados en el proceso

Impacto	Definición
Bajo	No tiene una relación directa con el proceso, por ende, sus actividades no tienen ningún impacto en este.
Medio	Es un participante que participa indirectamente en al proceso de aseguramiento de la calidad del *software*, indica actividades de mejora que se deben aplicar al proceso.
Alto	Es un participante directo del proceso de aseguramiento de la calidad del *software*, es decir, ejecuta las actividades directas del proceso y los resultados de este dependen del nivel de efectividad con que aplica las actividades del proceso.

- Planificación

Este dominio aborda las actividades iniciales y los recursos necesarios para la ejecución del proceso planteado que forman parte del plan piloto.

Dentro de los elementos que se toman en cuenta para la fase de planificación están:

- Cronograma

Para la elaboración del cronograma se toma en cuenta una de las variables de riesgo pensadas para el proyecto como lo es la resistencia al cambio. Por ello se establece que el proyecto será implementado de manera incremental iniciando por la realización de las pruebas unitarias, posteriormente se solicitará la realización de pruebas de integración y finalmente se incorporarán las pruebas *end-to-end* al proceso de aseguramiento de la calidad del *software*.

De esta forma, se establece la siguiente distribución para la realización de las actividades:

- Tiempo de adquisición de la herramienta de apoyo: 1 semana.
- Capacitación del equipo de analistas de QA: 1 semana.
- Capacitación de los desarrolladores de *software*: 2 días.
- Ejecución de pruebas unitarias: 12 semanas.
- Ejecución de pruebas de integración: 12 semanas.
- Ejecución de pruebas end-to-end: 12 semanas.

- Recursos

Los recursos indican los elementos necesarios para implementar de una forma exitosa el proyecto. Para este proyecto se identificaron los siguientes recursos necesarios para la implementación de este y se encuentran resumidos en la **Tabla 23**.

Tabla 23
Recursos necesarios para el proyecto

Recurso	Descripción
Herramienta para gestión de casos de prueba	Herramienta de *software* que servirá de apoyo para el proceso de aseguramiento de calidad gracias a su utilidad para gestionar los casos de prueba de *software* en general.
Analista de QA	Persona encargada de guiar y apoyar la ejecución del proceso de aseguramiento de calidad del *software* a través de su conocimiento en la industria y habilidades técnicas.

- Incertidumbre

El concepto de incertidumbre tiene una relación directa con los riesgos según el PMBOK, el primer paso que se debe llevar a cabo para la gestión de la incertidumbre es identificar los riesgos; para el presente proyecto se determinó a través de una lluvia de ideas una serie de riesgos

que podrían tener un impacto directo sobre las actividades planeadas. En la **Tabla 24** se identifican los riesgos planteados.

Tabla 24
Riesgos del proyecto

ID del riesgo	Riesgo	Descripción
R1	Resistencia al cambio	Temor, incomprensión, o inadaptación de algunos involucrados, estos pueden mostrar rechazo a estos nuevos cambios.
R2	Retraso en la planificación propuesta	Tiempo adicional que se puede tomar para implementar el proyecto.
R3	Dificultad en el uso de herramientas propuestas	Existe la posibilidad de que la herramienta de *software* propuesta sea nueva para los desarrolladores de *software* y por ende se les dificulte su uso.
R4	Problemas de comunicación	Este riesgo se refiere a la existencia de problemas para comunicar la implementación del proyecto a los involucrados.
R5	Problemas de integración de la herramienta	Se debe verificar si la herramienta de *software* propuesta cuenta con el visto bueno para ser instalada en los equipos.
R6	Aumento en el costo de la herramienta	El precio de la herramienta de *software* propuesta puede aumentar con respecto a lo presupuestado.
R7	Viabilidad económica de realizar el proyecto.	Existe la posibilidad de que la propuesta planteada no se adapte al presupuesto de la organización.

El siguiente paso dentro de este proceso es realizar un análisis de los riesgos identificados, para ello primeramente se debe identificar la gravedad y la probabilidad de ocurrencia de los riesgos.

Según Asana (2022), la gravedad mide qué tan graves serán las consecuencias de cada riesgo. Para clasificar la gravedad de los riesgos, Asana (2022) brinda una escala de categorización que se describe en la **Tabla 25**.

Tabla 25

Escala de gravedad de ocurrencia de riesgos

Nivel de gravedad	Gravedad	Descripción
1	Insignificante	El riesgo generará pocas consecuencias si ocurriera.
2	Menor	Las consecuencias del riesgo se gestionarán con facilidad.
3	Moderada	Las consecuencias del riesgo tardarán en mitigarse.
4	Importante	Las consecuencias de este riesgo serán significativas y pueden causar daños a largo plazo.
5	Catastrófica	Las consecuencias de este riesgo serán muy perjudiciales y puede resultar difícil recuperarse.

Nota. Tomado de *Matriz de riesgos: cómo evaluar los riesgos para lograr el éxito del proyecto*, por Asana, 2022.

Seguidamente se debe calcular la probabilidad de ocurrencia del riesgo. Para ello, Asana (2022) establece una estrategia de categorización de probabilidad de ocurrencia de riesgos que se muestra en la **Tabla 26**.

Tabla 26

Escala de probabilidad de ocurrencia de riesgos

Nivel de probabilidad	Probabilidad	Descripción	Porcentaje de probabilidad de ocurrencia
1	Muy improbable	El hecho de que este riesgo ocurra es una posibilidad remota.	Menos del 5%
2	No es probable	Existe una gran probabilidad de que este riesgo no ocurra.	Entre el 6% y el 15%
3	Posible	Es posible que este riesgo pudiera ocurrir o no.	Entre el 16% y el 59%
4	Probable	Existe una gran probabilidad de que este riesgo ocurra.	Entre el 60% y el 89%
5	Muy probable	Es bastante seguro que este riesgo ocurrirá en algún momento.	Más del 90%

Nota. Tomado de *Matriz de riesgos: cómo evaluar los riesgos para lograr el éxito del proyecto*, por Asana, 2022.

Seguidamente se debe calcular el impacto de la materialización de los riesgos. Asana (2022), brinda un cálculo para el impacto de los riesgos en el proyecto. En la **Tabla 27** se muestra el cálculo mencionado. Es importante mencionar que esta clasificación se obtiene al multiplicar la gravedad por el impacto.

Tabla 27
Escala de impacto de ocurrencia de riesgos

Clasificación	Color	Descripción
Bajo (1-6)	Verde	Estos no tendrán consecuencias significativas para el proyecto
Medio (7-12)	Amarillo	Pueden causar contratiempos en el proyecto.
Alto (13-25)	Rojo	Pueden causar el fracaso en la implementación del proyecto.

Nota. Tomado de *Matriz de riesgos: cómo evaluar los riesgos para lograr el éxito del proyecto*, por Asana, 2022.

Seguidamente, se procede a realizar el análisis de riesgos para el presente proyecto utilizando como insumo la información anterior. En la **Tabla 28** se muestra el resumen del análisis.

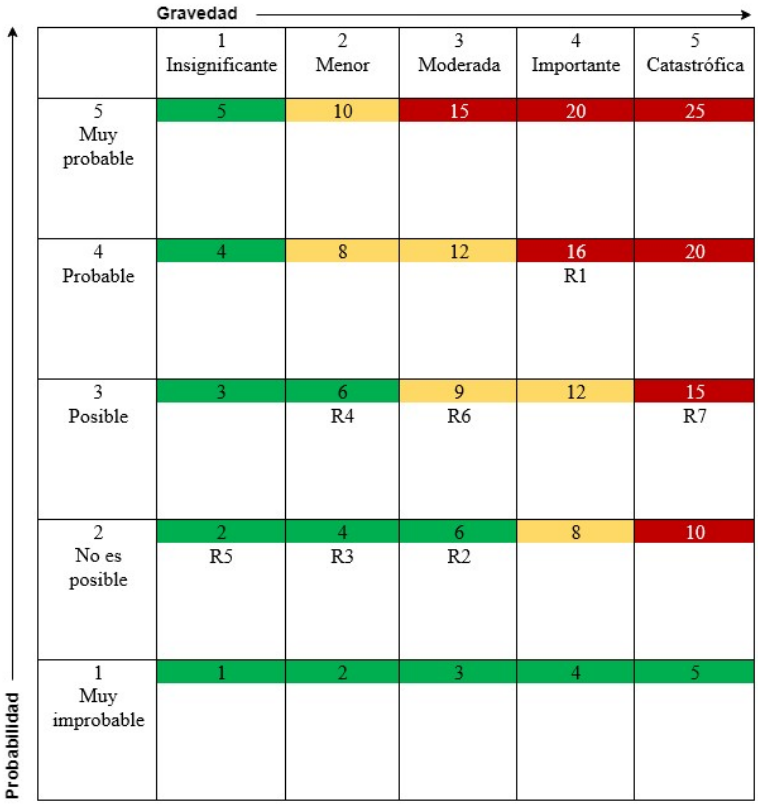
Tabla 28
Análisis de riesgos del proyecto

ID del riesgo	Gravedad	Probabilidad	Impacto (Gravedad x Probabilidad)	Color
R1	4	4	16	Rojo
R2	3	2	6	Verde
R3	2	2	4	Verde
R4	2	3	6	Verde
R5	1	2	2	Verde
R6	3	3	9	Amarillo
R7	5	3	15	Rojo

Finalmente, se realiza la matriz de riesgos para el proyecto (ver **Figura 21**), para ello se toma como base la plantilla de matriz de riesgos propuesta por Asana (2022), esta plantilla se encuentra en el **Anexo I .Plantilla para matriz de riesgos**.

Figura 21

Matriz de riesgos del proyecto



Nota. Adaptado de *Matriz de riesgos: cómo evaluar los riesgos para lograr el éxito del proyecto*, por Asana, 2022.

Otro aspecto importante con respecto a la gestión de riesgos es tener un plan de respuesta para estos, para los riesgos de este proyecto se presenta el plan de riesgos que se muestra en la **Tabla 29**.

Tabla 29

Análisis de riesgos del proyecto

ID del riesgo	Riesgo	Acción propuesta
R1	Resistencia al cambio	Para ellos se propone un plan de implementación por fases que permita adaptar a los involucrados en el proceso de cambio.
R2	Retraso en la planificación propuesta	Se debe contar con un margen de tiempo para la

ID del riesgo	Riesgo	Acción propuesta
		implementación del proyecto por si se dificulta realizar la implementación en la fecha propuesta.
R3	Dificultad en el uso de herramientas propuestas	Se debe brindar capacitación sobre el uso de la herramienta de *software* que facilite el uso de esta.
R4	Problemas de comunicación	Se contempla la gestión de la comunicación en el plan de implementación del proyecto, para ello se especifican los medios necesarios para comunicar el proyecto.
R5	Problemas de integración de la herramienta	Se debe contar con el visto bueno por parte del área de seguridad para realizar la implementación de la herramienta en los equipos de la organización.
R6	Aumento en el costo de la herramienta	Se tienen varias opciones de herramientas disponibles para adquirir, por ende, si el costo de la herramienta aumentara se evaluará migrar a otra.
R7	Viabilidad económica de realizar el proyecto.	Se evaluarán diversas opciones para la implementación del proyecto con el supervisor de sistemas Pega, considerando la que se ajuste al presupuesto disponible para la implementación del proyecto.

4.5. Fase 5: Análisis de costo-beneficio del proyecto

Para esta sección se realiza lo siguiente: revisión documental para obtener datos relacionados con el análisis de costos del proyecto, por ejemplo, salarios, cargas sociales, costos de licencias y otra información relevante para este cálculo.

Además, se realiza una reunión con el supervisor de producto de *software* para determinar los costos asociados con la creación de un equipo que se dedique al aseguramiento de la calidad del *software*.

Asimismo, se realiza una reunión con el supervisor de sistemas Pega para tratar de conocer los posibles ingresos que se obtendrían con la implementación del proceso en la organización, esto con el fin de realizar el análisis ROI para identificar si el proyecto es viable de realizar en términos económicos.

El resultado de esta fase se encuentra desarrollado en el Capítulo V en el apartado **Fase 5: Análisis de costo-beneficio del proyecto**.

Después de haber realizado el análisis de resultados correspondiente al Capítulo IV, se procede a desarrollar el capítulo que le brinda valor a la organización, es decir, la propuesta de solución para la problemática planteada. Esta propuesta se desarrolla en la siguiente sección.

5. Capítulo 5: Propuesta de solución

En el capítulo actual se desarrolla la propuesta que busca brindar una solución a la problemática vivida en el departamento con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*.

Para ello, se utiliza como insumo la información recopilada en el capítulo anterior de análisis de resultados, a partir del estudio realizado se desarrollan las siguientes secciones que se muestran en la **Figura 22**.

Figura 22

Contenido abarcado en el capítulo 5 según el procedimiento metodológico

5.1. Fase 2: Rediseño del proceso

El propósito de este punto es diseñar el nuevo proceso de desarrollo y aseguramiento de la calidad del *software*. Se enfatiza en este último pues el análisis de estándares y buenas prácticas se orienta al ciclo de vida de dicho proceso.

La finalidad de esta fase es brindar al departamento un proceso estandarizado para el desarrollo y aseguramiento de la calidad del *software*, en donde se especifiquen las actividades que componen el proceso y su correspondiente descripción, además de los roles que forman parte de este.

Además de lo anterior, se brindan una serie de instrumentos recomendados que sirven de apoyo para el proceso de desarrollo y aseguramiento de la calidad del *software* y que complementan la solución propuesta para la organización.

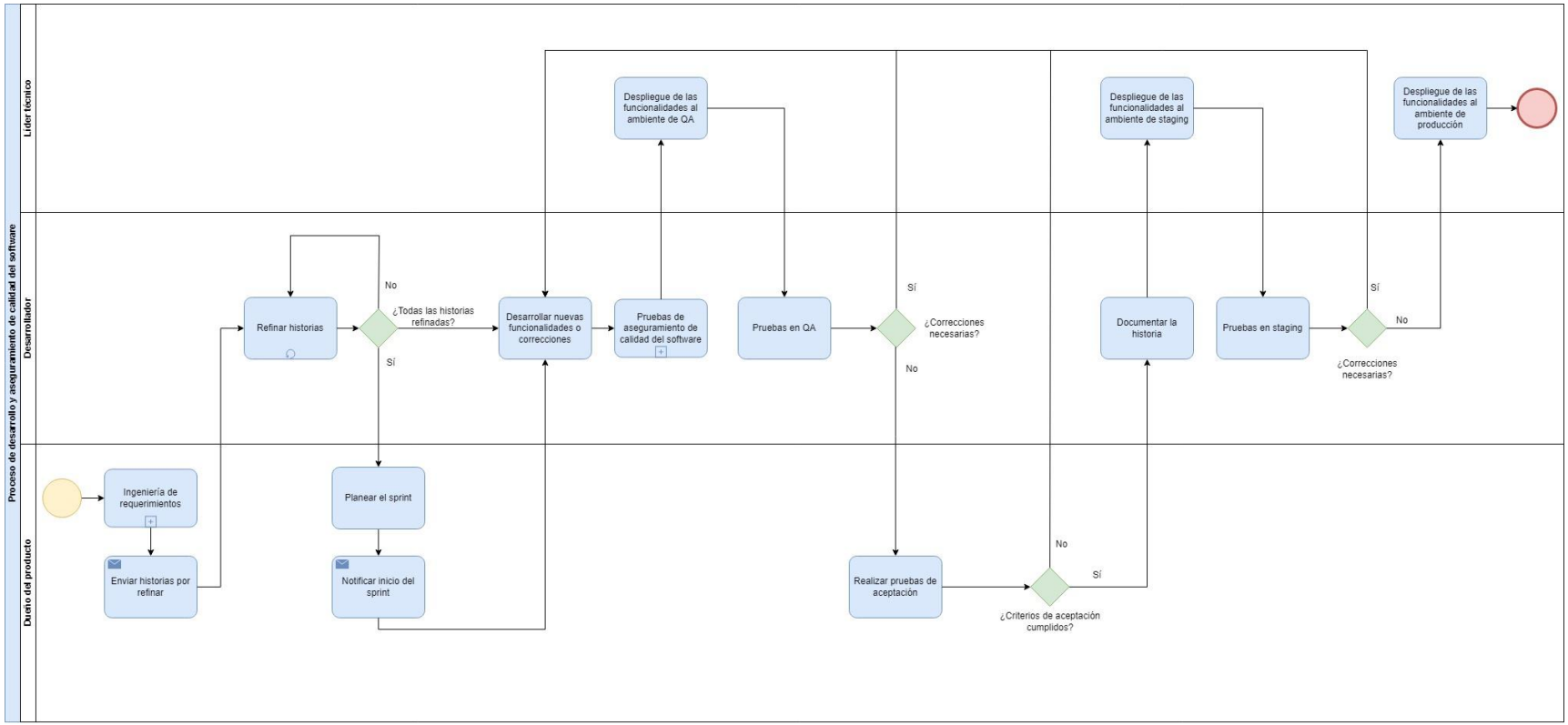
5.1.1. Diseño del proceso *to-be*

El diseño del nuevo proceso se basa en la evaluación de estándares y buenas prácticas enfocado en el punto de **Revisión de buenas prácticas de la industria**, además de recomendaciones brindadas por el supervisor de sistemas Pega y con el supervisor de producto de *software*, por medio de diversas reuniones documentadas en las minutas **Apéndice T. Minuta 3** y **Apéndice U. Minuta 4.**

De manera general, el proceso de desarrollo y aseguramiento de la calidad del *software* se muestra en la **Figura 23**. Es importante recalcar que dentro de este proceso general se encuentra el subproceso de ingeniería de requerimientos de *software* y el proceso de pruebas de aseguramiento de calidad del *software* como subprocesos para facilitar la representación del diagrama. Más adelante se encuentran explicados estos subprocesos con profundidad.

Figura 23

Diagrama del proceso to-be de desarrollo y aseguramiento de calidad de software



En la **Tabla 30** se describen las diversas actividades que forman parte del proceso *to-be* de desarrollo y aseguramiento de la calidad del *software*; asimismo, con respecto al proceso *as-is* generado en la sección **Diseño del proceso *to-be*** se destaca que se eliminan algunas actividades y se incluye el subproceso de aseguramiento de calidad del *software*, este subproceso se describe posteriormente en esta misma sección.

Tabla 30
Descripción de las actividades del proceso to-be

Actividad	Descripción
Subproceso de ingeniería de requerimientos de *software*	En este subproceso se encuentran todas las actividades relacionadas con el proceso de ingeniería de requerimientos de *software*. Desde la recopilación de requerimientos hasta la creación del backlog.
Enviar historias por refinar	El dueño del producto informa sobre las historias de usuario que se van a refinar con el objetivo de incluirlas en los próximos *sprints*.
Refinar historias	Los desarrolladores verifican los criterios de aceptación de cada historia del *backlog* para estimar los puntos de esfuerzo que tendrá la historia. Es necesario indicar que estos puntos estimados podrían cambiar una vez finalice el desarrollo de la historia.
Planear el sprint	El dueño del producto incluye en el *sprint* algunas historias de usuario registradas en el *backlog* del equipo.
Notificar inicio del sprint	El dueño del producto (*Product Owner)* notifica al equipo de desarrollo sobre el inicio del *sprint* planificado anteriormente.
Desarrollar nuevas funcionalidades o correcciones	En esta actividad se realiza el desarrollo de las nuevas funcionalidades requeridas para la historia de usuario o se realizan las correcciones según lo identificado al ejecutar las pruebas.
Subproceso de pruebas de aseguramiento de la calidad del *software*	En este subproceso se incluyen todas las actividades relacionadas con el proceso de aseguramiento de la calidad del *software*, este se encuentra divido en tres fases: planeación de

Actividad	Descripción
	las pruebas, diseño de las pruebas y ejecución de las pruebas.
Despliegue de las funcionalidades al ambiente de QA	Se envían las funcionalidades desarrolladas al ambiente de QA.
Pruebas en QA	Se realizan pruebas de las funcionalidades en el ambiente de QA.
Realizar pruebas de aceptación	Se realizan pruebas con el dueño del producto (*Product Owner)* para verificar que las funcionalidades desarrolladas cumplen con los criterios de aceptación de la historia.
Documentar la historia	El desarrollador realiza la documentación de la historia a través de un documento que contiene varias secciones como lo son la descripción de la historia y las evidencias de cambios realizados.
Despliegue de las funcionalidades al ambiente de staging	Se envían las funcionalidades desarrolladas al ambiente de *staging*.
Pruebas en staging	Se realizan pruebas de las funcionalidades en el ambiente de *staging*.
Despliegue de las funcionalidades al ambiente de producción	Se envían las funcionalidades desarrolladas al ambiente de producción.

Como se puede apreciar en la **Figura 23**, el proceso de ingeniería de requerimientos de *software* y el proceso de pruebas de aseguramiento de la calidad del *software* se muestran como un subproceso, por ende, a continuación, se detallan y se explican cada fase de estos.

A continuación se procede a explicar cada subproceso mencionado anteriormente, iniciando con el proceso de ingeniería de requerimientos de *software*. Este proceso se muestra en la **Figura 24**, a continuación se describen las principales actividades desarrolladas en este proceso.

- Informar sobre sesión de descripción del problema: este paso es el que inicia este proceso, y se basa en la comunicación sobre la necesidad de analizar una problemática que tiene el negocio y que desea resolver, el actor del lado de negocio es el responsable de organizar esta sesión.
- Ejecutar técnicas para la recopilación de requerimientos: cuando se realiza la sesión para analizar el problema se ejecutan las técnicas por parte del *product owner* para recopilar los requerimientos del negocio.
- Analizar los requerimientos recopilados: en esta actividad el *product owner* realiza un análisis de los requerimientos recopilados para resolver cualquier inquietud o ambigüedad

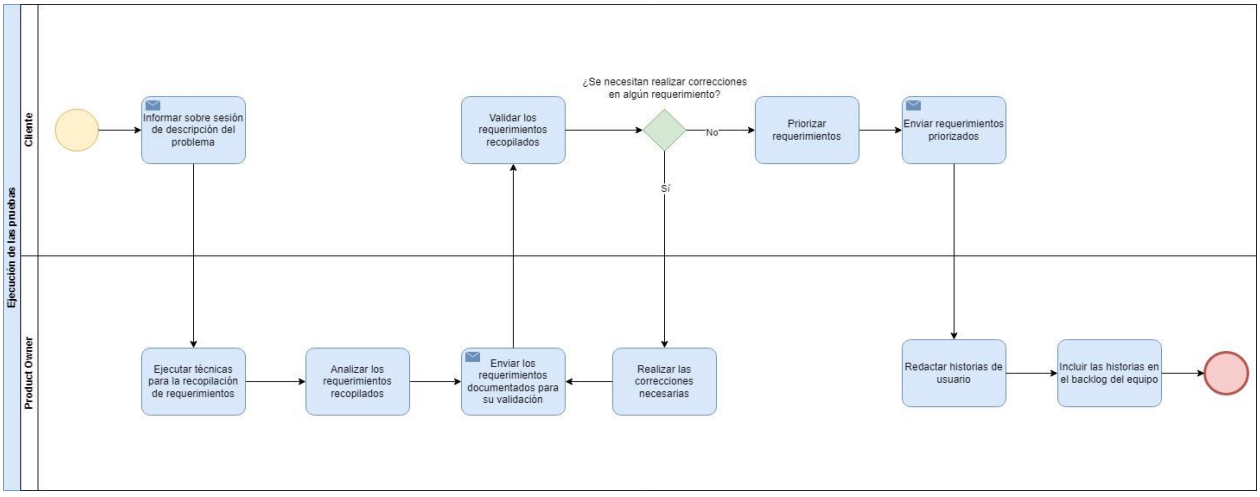
sobre los requerimientos y las envía al cliente del lado de negocio para validar que se cumplan los objetivos que busca el cliente.

- Validar los requerimientos recopilados: el cliente por parte del negocio valida que los requerimientos que redactó el *product owner* cumplen con la necesidad del negocio y resuelven la problemática establecida inicialmente. Si existiera una incongruencia entre lo deseado por el cliente y el requerimiento redactado por el *product owner* se envían las correcciones necesarias y el *product owner* es responsable de realizar los ajustes necesarios para cumplir con las expectativas del cliente.
- Priorizar requerimientos: una vez finaliza el proceso de validación de los requerimientos el cliente es responsable de realizar una priorización de los requerimientos para hacer saber al *product owner* la urgencia del trabajo que se debe planificar. Una vez se realiza esta priorización se envían la lista de requerimientos priorizados el *product owner*.
- Redactar historias de usuario: con la lista de requerimientos debidamente establecida se procede a redactar las historias de usuarios para cada requerimiento. Seguidamente estas historias de usuario se agregan al backlog del equipo.

Es importante indicar que más adelante en el proceso de desarrollo y aseguramiento de calidad del *software* se encuentra una actividad para la ejecución de pruebas de aceptación con el *product owner* del equipo. En esta actividad se verifica que las funcionalidades desarrolladas cumplan efectivamente con lo especificado por parte del cliente en la sesión de recopilación de requerimientos. Todo este proceso de ingeniería de requerimientos es verificado por el analista de QA para asegurarse que los requerimientos estén correctamente definidos.

Figura 24

Proceso de ingeniería de requerimientos



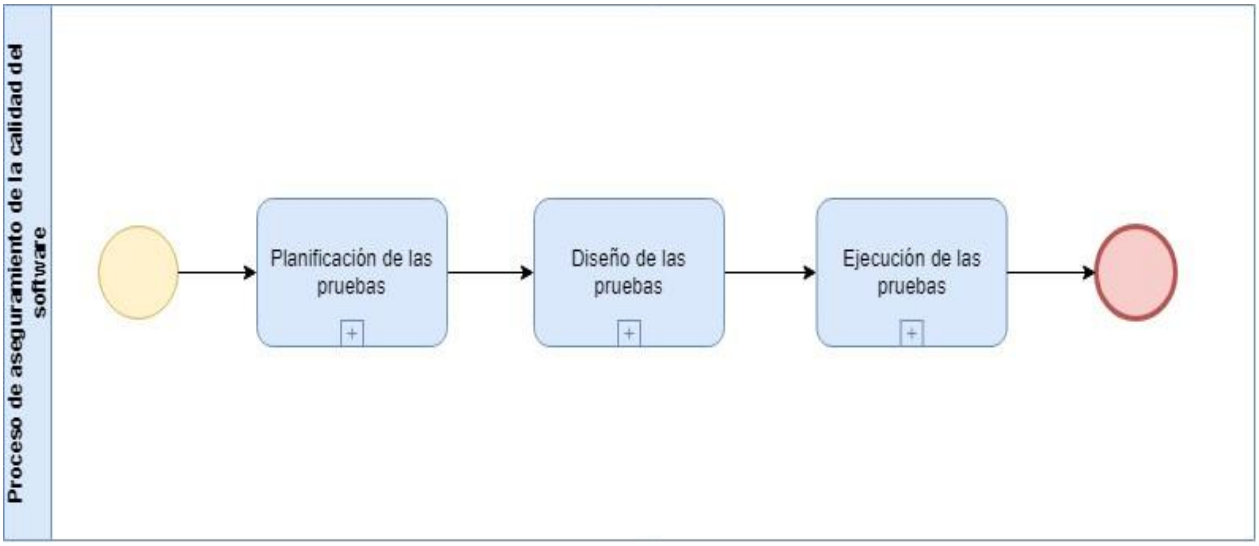
En la **Figura 25** se diagrama el proceso de pruebas de aseguramiento de la calidad de *software* estandarizado propuesto para el departamento de Sistemas Banca y Procesos, el cual servirá de guía para que los distintos equipos se apoyen y alineen con los esfuerzos que plantea el

Centro de Excelencia Operacional (COE) para mejorar la calidad de los productos ofrecidos a los clientes y la búsqueda de la estandarización en sus procesos.

Como resultado de la investigación realizada de buenas prácticas se concluye que las principales actividades que componen el proceso son las siguientes:

- Planificación de las pruebas de *software.*
- Diseño de las pruebas de *software*.
- Ejecución de las pruebas de *software*.

Figura 25
Proceso de aseguramiento de calidad del software



Un aspecto importante es que el proceso planteado se alinea con lo establecido en el **Ciclo de Deming o PDCA**, el cual establece las fases con respecto a la metodología de gestión y tiene como objetivo la mejora continua de los procesos, estas fases son: planificar, hacer, revisar y actuar (*Plan, Do, Check, Act).*

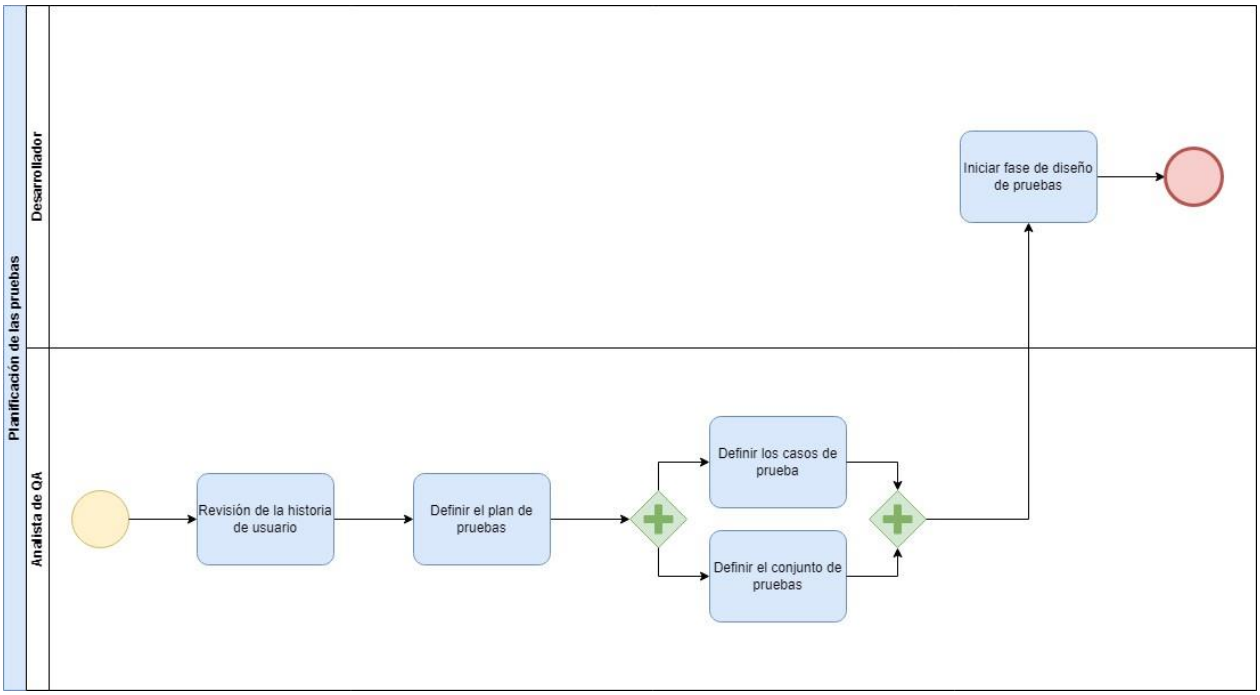
5.1.1.1. Subproceso de planificación de las pruebas de *software*

En la **Figura 26** se pueden observar las principales actividades que conforman este subproceso de planificación de las pruebas y que tiene como principal objetivo analizar las historias de usuario, definir el ciclo de pruebas y los casos de prueba necesarios por tipo de prueba (unitarias, integración, *end-to-end*) para cada historia de usuario.

Mediante este subproceso de planificación de las pruebas de *software* se tiene como objetivo redactar en un alto nivel los diferentes casos de prueba que serán importantes para cada

historia de usuario y serán agrupados en conjuntos de pruebas, estos posteriormente serán elaborados completamente en la fase de diseño de las pruebas como se apreciará adelante.

Figura 26
Subproceso de planificación de las pruebas de software



En la **Tabla 31** se muestra la descripción de cada una de las actividades propuestas para este subproceso.

Tabla 31
Actividades del subproceso de planificación de las pruebas de software

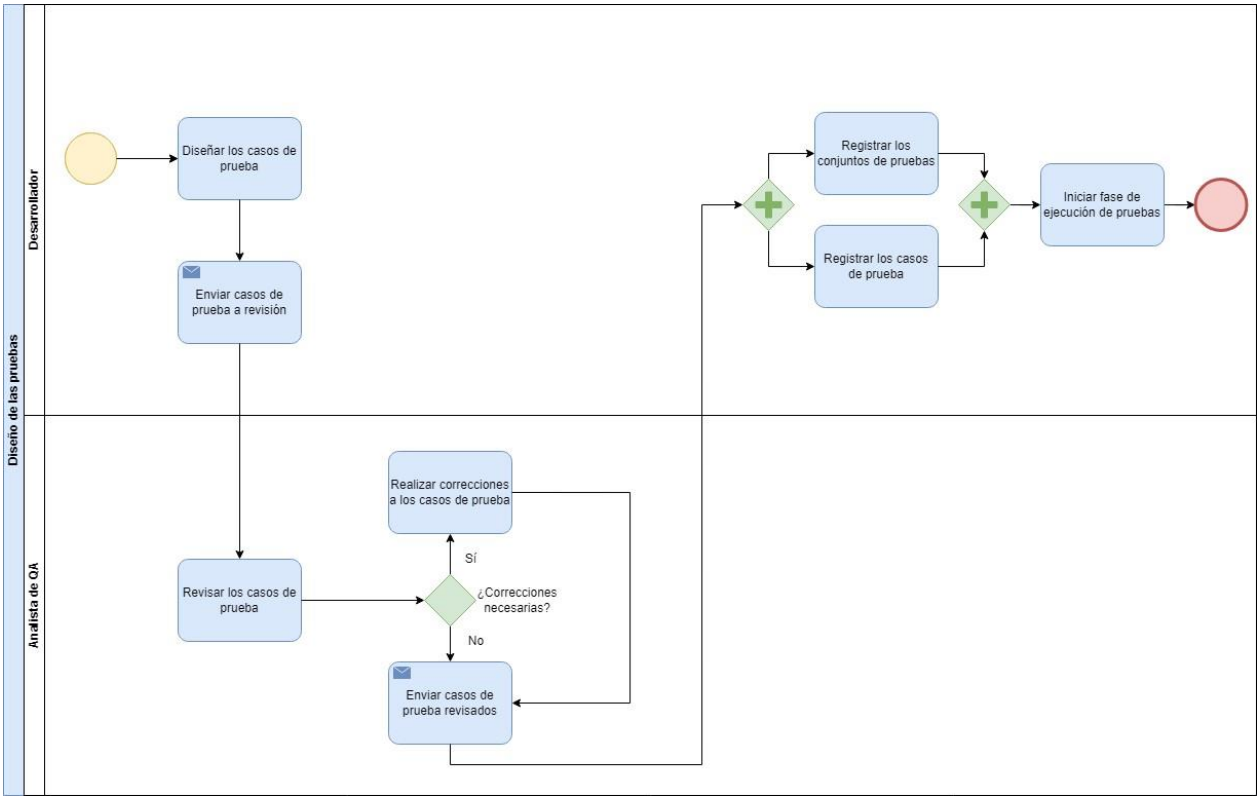
Actividad	Descripción
Revisión de historia de usuario	En esta actividad se revisa la historia de usuario para verificar los criterios de aceptación de la historia e identificar el plan de pruebas necesario para cubrir las funcionalidades desarrolladas en la historia de usuario.
Definir el plan de pruebas	Este es un documento que contiene la estrategia que será aplicada para ejecutar el plan de pruebas, este contiene entre otras cosas la historia de usuario asociada, la descripción de la historia de usuario, la configuración del ambiente de pruebas, los criterios de cobertura del plan de pruebas, los casos de prueba

Actividad	Descripción
	agrupados en conjuntos de pruebas y el detalle sobre la priorización de las pruebas.
Definir los casos de prueba	En este paso se define a un alto nivel los casos de prueba necesarios.
Definir el conjunto de pruebas	Se organizan los casos de prueba en conjuntos de prueba para facilitar su orden de ejecución.
Iniciar la fase relacionada con el diseño de las pruebas	Una vez definidos los puntos anteriores, se debe realizar el subproceso de diseño de las pruebas.

5.1.1.2. Subproceso de diseño de las pruebas de *software*

En la **Figura 27** se pueden observar las principales actividades que conforman este subproceso de diseño de las pruebas, tienen como principal objetivo realizar el diseño de los casos de prueba, necesarios para la siguiente fase de ejecución y para cada tipo de prueba especificado.

Figura 27
Subproceso de diseño de las pruebas de software



En la **Tabla 32** se muestra la descripción de cada una de las actividades propuestas para este subproceso.

Tabla 32
Actividades del subproceso de diseño de las pruebas de software

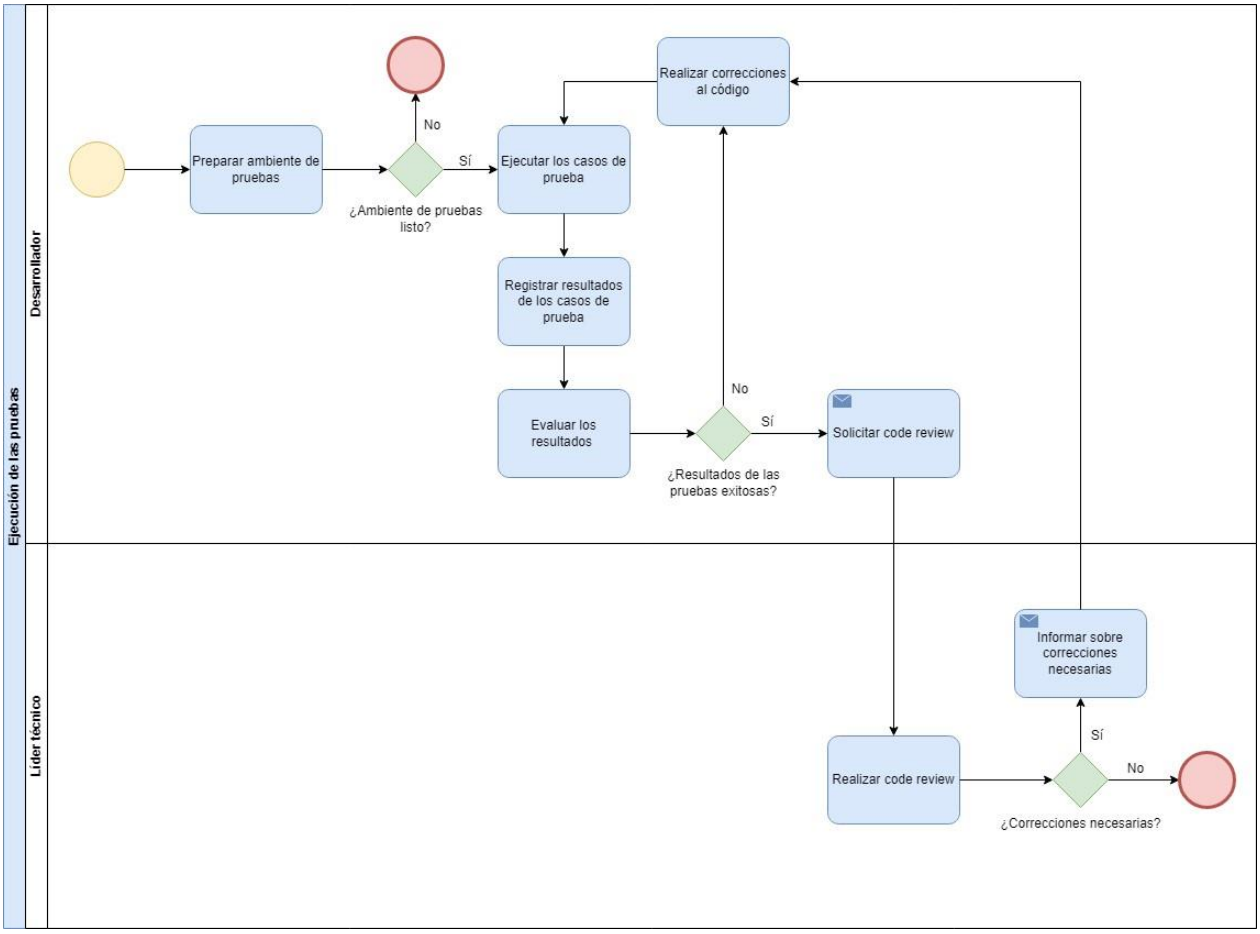
Actividad	Descripción
Diseñar los casos de prueba	Este paso consiste en diseñar los casos de prueba. Para ello se debe completar la información relacionada con la plantilla para casos de prueba (Ver **Apéndice W. Plantilla para documentación de casos de prueba**). Estos casos de prueban deben contener entre otras cosas el identificador del caso de prueba, el objetivo del caso de prueba, la descripción del caso de prueba, los resultados esperados y el estado de la prueba ejecutada.
Revisión de los casos de prueba	Se envían los casos de prueba al analista de aseguramiento de calidad del *software* para validar el alcance y la completitud de estos. De ser necesario, se realizan correcciones a los casos de prueba.
Envío de los casos de prueba	Una vez validados los casos de prueba por parte del analista de aseguramiento de calidad de *software* se deben enviar a los desarrolladores para que revisen los cambios aplicados, si es que se hubieran hecho correcciones.
Registrar los casos de prueba	El desarrollador se encarga de registrar los casos de prueba en la herramienta de gestión de casos de prueba para seguidamente continuar con el subproceso de ejecución de las pruebas. Este paso se realiza si existe una herramienta para la gestión de pruebas.
Registrar los conjuntos de pruebas	De igual forma el desarrollador debe crear los conjuntos de pruebas en la herramienta de gestión de pruebas para seguidamente vincular los casos de prueba creados con anterioridad a los conjuntos de pruebas definidos.

5.1.1.3. Subproceso de ejecución de las pruebas de *software*

En la **Figura 28** se pueden observar las principales actividades que conforman este subproceso de ejecución de las pruebas, tiene como principal objetivo ejecutar los casos de prueba diseñados en la fase anterior. Se inicia primeramente con los casos de prueba relacionados con las pruebas unitarias, luego los de las pruebas de integración y finalmente con las pruebas *end-to-end*.

Una vez ejecutadas las diferentes pruebas se procede a documentar su resultado, se determina si se deben realizar correcciones al código desarrollado y seguidamente se vuelven a ejecutar las pruebas en las que no se obtuvieron los resultados deseados. Este ciclo se repite para cada caso de prueba, hasta corroborar que en todos los casos ejecutados obtuvieron los resultados esperados.

Figura 28
Subproceso de ejecución de las pruebas de software



En la **Tabla 33** se muestra la descripción de cada una de las actividades propuestas para este subproceso.

Tabla 33

Actividades del subproceso de ejecución de las pruebas de software

Actividad	Descripción
Preparar ambiente de pruebas	Antes de comenzar con la ejecución de las pruebas se debe verificar que el ambiente de pruebas esté configurado correctamente con las precondiciones establecidas para poder ejecutar los casos de prueba correctamente.
Ejecutar los casos de prueba	Se procede a ejecutar los casos de prueba establecidos en el plan de pruebas iniciando con las pruebas unitarias, luego las de integración y finalmente con las pruebas *end-to-end.*
Registrar los resultados de los casos de prueba	Se deben registrar los resultados obtenidos al realizar los casos de prueba y el estado del caso de prueba (fallido o aprobado).
Realizar correcciones	Si el resultado del caso de prueba no fue exitoso se deben realizar las correcciones necesarias hasta obtener los resultados deseados del caso de prueba.
Realizar *code review*	El *code review* es una revisión que realiza el líder técnico del equipo para certificar que las funcionalidades desarrolladas son óptimas para continuar con el despliegue a los otros ambientes. Si el líder técnico indica que se deben realizar correcciones al código, el desarrollador debe proceder a realizarlas y efectuar pruebas nuevamente hasta certificar que se cumple con lo especificado en los casos de prueba.

Los anteriores son los subprocesos identificados como parte de la mejora al proceso sugerido de aseguramiento de calidad de *software* para el departamento, estos conforman el proceso general de desarrollo de *software*; se espera que todos los equipos pongan en práctica estas fases para los desarrollos que serán realizados en los *sprints* planificados y que son parte de la estandarización que se desea alcanzar como punto de mejora, con respecto al proceso ejecutado actualmente.

5.1.1.4. Roles involucrados en el proceso

En la **Tabla 34** se describen los principales roles que forman parte del nuevo proceso, es importante recalcar que la mayoría de los roles se mantiene, tomando en cuenta solamente la incorporación de un rol de analista de aseguramiento de la calidad del *software* como apoyo para el proceso con los conocimientos técnicos que debe poseer, según las necesidades buscadas por el departamento.

Tabla 34

Roles y responsabilidades de los involucrados en el proceso

Rol	Descripción
Desarrollador	Es la persona encargada de desarrollar las nuevas funcionalidades a lo largo del *sprint*, este debe realizar las siguientes actividades durante el proceso: • Desarrollo de mejoras o nuevas funcionalidades. • Registro de los casos de prueba dentro de la herramienta de gestión de casos de prueba. • Ejecución de los casos de prueba. • Registro del resultado de los casos de prueba. • Corrección de errores. • Documentación técnica sobre la historia de usuario asignada.
Líder técnico	El líder técnico es una persona dentro de cada equipo de desarrollo que apoya al resto de desarrolladores en el proceso de desarrollo del *software*. • Realizar el *code review* de las funcionalidades de cada historia realizada por el desarrollador de *software*. • En este caso se analiza la posibilidad de que el líder técnico del equipo se encargue de ejecutar las funciones del analista de QA en caso de que la opción de adquirir talento humano para este

Rol	Descripción
	nuevo rol no sea viable para el departamento.
Analista de QA	Este es el nuevo rol propuesto para reforzar el proceso de aseguramiento de la calidad del *software* en el departamento. Sus actividades en el nuevo proceso serán los siguientes: • Revisión y análisis de las historias de usuario de cada *sprint* trabajado. • Creación del conjunto de pruebas para la historia de usuario. • Creación de los casos de prueba requeridos. • Revisión de los casos de prueba diseñados por el desarrollador de *software*.
Propietario del producto (*Product owner)*	De igual forma dentro del proceso de desarrollo de *software* el propietario del producto se encarga de recopilar los requerimientos por parte del área de negocio y convertirlos en historias de usuario que serán atendidas por los desarrolladores a través del ciclo del *sprint*.

De igual forma es importante definir las habilidades necesarias que deberá reunir la persona que sea tomada en cuenta para asumir el nuevo rol de analista de aseguramiento de calidad del *software*, por ende, en la **Tabla 35** se resumen las principales habilidades sugeridas para el rol.

Tabla 35
Habilidades sugeridas para el rol de analista de QA

Habilidad	Nivel de conocimiento requerido
Conocimiento en el proceso de despliegue de *software* y seguimiento de defectos de *software*.	Alto
Conocimiento sobre pruebas manuales de *software*.	Alto
Experiencia con metodologías de pruebas de *software*.	Medio

Habilidad	Nivel de conocimiento requerido
Experiencia con el proceso de aseguramiento de calidad de *software*.	Alto
Habilidad para elaborar y aplicar metodologías de pruebas incluyendo la redacción de planes de pruebas y casos de prueba.	Alto
Conocimiento y experiencia en metodologías ágiles (scrum).	Medio
Experiencia con el uso de Jira.	Medio
Conocimientos en programación y bases de datos.	Bajo
Deseable conocimientos en automatización de pruebas de *software*.	Bajo
Conocimiento con la herramienta Pega.	Medio
Conocimientos sobre el uso de herramientas de gestión de casos de prueba.	Alto
Habilidades de comunicación verbal y escrita.	Medio

Matriz de asignación de responsabilidades RACI

La matriz RACI permite identificar los roles y responsabilidades para el proceso de desarrollo del aseguramiento de la calidad del *software*, de esta forma se indica la manera en que se distribuyen las tareas identificadas en el proceso planteado para la organización.

En la **Tabla 36** se muestran las actividades del proceso propuesto, así como la categorización de la responsabilidad asignada a cada rol.

Tabla 36
Matriz de asignación de responsabilidades RACI

Actividad	Desarrollador	Líder técnico	Analista de QA	Propietario del producto
Proceso de desarrollo de *software*				
Ingeniería de requerimientos	I	I	A	R
Enviar historias por refinar	I	I	I	R
Refinar historias	R	C	I	C
Planear el sprint	I	C	I	R
Notificar inicio del sprint	I	I	I	R

Actividad	Desarrollador	Líder técnico	Analista de QA	Propietario del producto
Desarrollar nuevas funcionalidades o correcciones	R	C	I	I
Despliegue de las funcionalidades al ambiente de QA	I	R	I	I
Pruebas en QA	R	I	I	I
Realizar pruebas de aceptación	I	I	I	R
Documentar la historia	R	I	I	I
Despliegue de las funcionalidades al ambiente de staging	I	R	I	I
Pruebas en staging	R	I	I	I
Despliegue de las funcionalidades al ambiente de producción	I	R	I	I
Proceso de aseguramiento de la calidad de *software*				
Subproceso de planificación de las pruebas				
Revisión de historia de usuario	I	C	R	C
Definir el plan de pruebas	I	I	R	I
Definir los casos de prueba	I	I	R	I
Definir el conjunto de pruebas	I	I	R	I
Subproceso de diseño de las pruebas				
Diseñar los casos de prueba	R	I	C	I
Revisión de los casos de prueba	I	I	R	I
Envío de los casos de prueba	I	I	R	I
Registrar los casos de prueba	R	I	C	I
Registrar los conjuntos de pruebas	R	I	C	I
Subproceso de ejecución de las pruebas				
Preparar ambiente de pruebas	R	C	C	I
Ejecutar los casos de prueba	R	I	C	I
Registrar los resultados de los casos de prueba	R	I	C	I
Realizar correcciones	R	C	I	I
Crear documentación de la historia	R	I	I	I

5.1.1.5. Artefactos propuestos para el proceso

Como parte del proceso planteado para la organización se sugiere utilizar una plantilla para los casos de prueba que se deben ejecutar. Los elementos que contiene esta plantilla son los siguientes:

- ID del caso de prueba: el caso de prueba debe estar representado por una identificación única y seguir una convención, por ejemplo, CP_Descripción de la prueba.
- Prioridad: se refiere a la prioridad de ejecución de la prueba. Se definen los siguientes valores: bajo, medio o alto.
- Fecha: fecha en que se ejecuta la prueba.
- Descripción de la prueba: resumen del propósito de la prueba.
- Precondiciones: requisito o requisitos que deben cumplirse antes de la ejecución del caso de prueba. Para ejecutar el caso de prueba, se enumeran todas las condiciones previas.
- Pasos por ejecutar: se enlistan detalladamente todos los pasos que deben ejecutarse y el orden específico para completar el escenario de prueba.
- Datos de prueba: conjunto de datos de prueba como entrada para el caso de prueba.
- Resultados esperados: se menciona el resultado esperado de la prueba.
- Resultado actual: se indica el resultado real una vez ejecutado el caso de prueba.
- Estado: se indica el estado final de la prueba. Los valores establecidos son aprobado o reprobado.
- Comentarios: cualquier nota u observación que brinde valor con respecto al caso de prueba ejecutado.

La plantilla propuesta para la documentación de casos de prueba puede encontrarse en el **Apéndice W. Plantilla para documentación de casos de prueba.**

5.1.2. Plan de aseguramiento de calidad del *software*

Del contenido descrito en la sección **Plan de aseguramiento de la calidad del *software***, para el plan propuesto se toman en cuenta las siguientes secciones:

- Propósito
  Este documento tiene como objetivo definir el plan de aseguramiento de la calidad del *software* para el Departamento de Sistemas Banca y Procesos del Grupo

Financiero. Este documento funciona como estándar para la realización del proceso de desarrollo y aseguramiento de calidad del *software* en los equipos de desarrollo Pega y las actividades del proceso se deberán realizar para cada historia de usuario trabajada durante el *sprint*.

- Documentación de referencia
  - SWEBOK.
  - Estándar IEEE para la elaboración de planes de aseguramiento de calidad de *software*.
  - Libro *Software* testing foundations.
  - Libro Succeeding with agile: *Software* Development Using scrum.
  - Libro *Software* Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement.
- Gestión:
  - Responsabilidades:
    - Desarrollador:
      - Desarrollo de mejoras o nuevas funcionalidades.
      - Registro de los casos de prueba dentro de la herramienta de gestión de casos de prueba.
      - Ejecución de los casos de prueba.
      - Registro del resultado de los casos de prueba.
      - Corrección de errores.
      - Documentación técnica sobre la historia de usuario asignada.
    - Líder técnico:
      - Realizar el code review de las funcionalidades de cada historia realizada por el desarrollador de *software*.
      - En este caso se analiza la posibilidad de que el líder técnico del equipo se encargue de ejecutar las funciones del analista de QA en caso de que la opción de adquirir talento humano para este nuevo rol no sea viable para el departamento.
    - Analista de QA:
      - Revisión y análisis de las historias de usuario de cada sprint trabajado.
      - Creación del conjunto de pruebas para la historia de usuario.
      - Creación de los casos de prueba requeridos.
      - Revisión de los casos de prueba diseñados por el desarrollador de *software*.
  - Tareas

    La descripción de las tareas se encuentra descrita en la sección de Herramientas, técnicas y metodologías.

- Herramientas y metodologías
  - Herramientas: las principales herramientas que se utilizarán en el proceso son:
    - Pega para el desarrollo de *software*.
    - AgileTest para la gestión de casos de prueba.
  - Metodología: la metodología que se debe seguir para el proceso de desarrollo y aseguramiento de calidad del *software* es la misma que se encuentra indicada en la sección **Diseño del proceso *to-be***.

5.2. Fase 3: Selección de herramienta de apoyo al proceso

En esta sección se realiza la escogencia final de la herramienta que servirá de apoyo para el proceso planteado de desarrollo y aseguramiento de calidad del *software*. Para ello se elabora una metodología para evaluar las herramientas preseleccionadas y la que obtenga la mayor puntuación final será la seleccionada.

5.2.1. Selección final de la herramienta

Como se mencionó en la sección **Fase 3: Selección de herramienta de apoyo al proceso**, se recomienda apoyar el proceso de aseguramiento de calidad del *software* con diferentes herramientas que permitan gestionar los casos de pruebas, los planes de pruebas, entre otros aspectos relacionados con la ejecución de las pruebas de aseguramiento de la calidad.

Debido a esto, se debe seleccionar una herramienta para la gestión de pruebas de las sugeridas en el punto **4.3.3 Investigación de herramientas disponibles**, el proceso de selección se desarrolla a continuación.

5.2.1.1. Metodología de evaluación

Es necesario establecer ciertos parámetros para decidir si las herramientas propuestas cumplen o no con los requerimientos identificados en el punto de Priorización de requerimientos, para ello se crea un sistema de calificación que se muestra en la **Tabla 37**.

Tabla 37
Sistema de calificación para los requerimientos

Grado de cumplimento	Descripción	Puntaje asignado
No cumple	La herramienta no reúne las condiciones para cumplir con el requerimiento indicado.	0
Parcial	La herramienta reúne parcialmente las condiciones para cumplir con el requerimiento especificado.	1

Grado de cumplimento	Descripción	Puntaje asignado
Total	La herramienta reúne las condiciones necesarias para cumplir con el requerimiento especificado.	2

Además, con el objetivo de dar relevancia a la priorización de requerimientos en la que se utiliza la metodología MoSCoW, se asigna una puntuación a cada valor como se describe seguidamente:

- *Must have:* 4 puntos.
- *Should have:* 3 puntos.
- *Could have:* 2 puntos
- *Won´t have:* 1 punto

Para cada requerimiento que sea categorizado según la clasificación anterior, se multiplicará la puntuación obtenida por los valores establecidos anteriormente según corresponda. Por ejemplo, si una herramienta cumple con un requerimiento totalmente y además ese requerimiento es categorizado como *must* obtendrá una calificación final de 2 * 4 = 8 puntos para dicho requerimiento. Estos puntajes relacionados con la priorización utilizando la metodología MoSCoW será aplicada únicamente en la sección **Resumen de la evaluación**. Además para las herramientas que cumplan totalmente con el requerimiento de que sean gratuitas se les sumarán 3 puntos adicionales.

5.2.1.2. Evaluación de las herramientas preseleccionadas

Una vez establecido el sistema de calificación para los requerimientos de la herramienta se procede a evaluar las cinco herramientas preseleccionadas en el punto Investigación de herramientas disponibles. Para cada una de ellas se evalúa cada requerimiento y se le otorga el puntaje obtenido.

Tuskr

Para realizar la evaluación de esta herramienta se utiliza la información disponible según Tuskr (s.f) en su sitio web. En la **Tabla 38** se muestra la evaluación correspondiente a la herramienta Tuskr.

Tabla 38
Evaluación de la herramienta Tuskr

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
Req1	La herramienta debe permitir crear casos de prueba.	*Must*	Existen diversos planes que permiten crear desde 1000	2

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
			hasta 250 mil casos de prueba.	
Req2	La herramienta debe permitir ejecutar casos de prueba manualmente.	*Must*	Las casos puede ser ejecutados manualmente.	2
Req3	La herramienta debe permitir crear conjuntos de pruebas.	*Must*	Sí es posible crear conjuntos de pruebas.	2
Req4	La herramienta debe poder integrarse con Jira.	*Must*	La herramienta con más de 400 integraciones diferentes, entre ellas Jira.	2
Req5	La herramienta debe ser preferiblemente de versión gratuita.	*Should*	La herramienta cuenta con una versión gratuita pero solamente para un máximo de 5 usuarios. El precio mínimo por usuario es de $9 por mes.	1
Req6	La herramienta debe permitir la escalabilidad con respecto al número de usuarios permitidos.	*Must*	La herramienta permite actualizar el plan actual por uno con mayor capacidad de usuarios.	2
Req7	La herramienta debe permitir la autenticación mediante inicio de sesión único (*Single Sign-On*).	*Should*	La herramienta permite la autenticación mediante inicio de sesión único, aunque sólo está disponible en el plan *Enterprise.*	1
Req8	La herramienta debe permitir asociar casos de	*Should*	Sí cuenta con integración con	2

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
	prueba a historias de usuario.		historias de usuario de Jira.	

Kualitee

Para realizar la evaluación de esta herramienta se utiliza la información disponible según Kualitee (s.f) en su sitio web. En la **Tabla 39** se muestra la evaluación correspondiente a la herramienta Kualitee.

Tabla 39

Evaluación de la herramienta Kualitee

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
Req1	La herramienta debe permitir crear casos de prueba.	*Must*	Permite crear casos de prueba ilimitados.	2
Req2	La herramienta debe permitir ejecutar casos de prueba manualmente.	*Must*	Se pueden ejecutar casos de prueba manuales y automáticos.	2
Req3	La herramienta debe permitir crear conjuntos de pruebas.	*Must*	Sí es posible crear conjuntos de pruebas.	2
Req4	La herramienta debe poder integrarse con Jira.	*Must*	Se puede integrar con Jira, GitHub, Asana, entre otras herramientas.	2
Req5	La herramienta debe ser preferiblemente de versión gratuita.	*Should*	El precio mínimo por usuario es de $15 por mes.	0
Req6	La herramienta debe permitir la escalabilidad con respecto al número de usuarios permitidos.	*Must*	Se pueden agregar o eliminar usuarios a demanda.	2
Req7	La herramienta debe permitir la autenticación	*Should*	Cuenta con soporte para realizar *Single Sign-On.*	2

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
	mediante inicio de sesión único (*Single Sign-On*).			
Req8	La herramienta debe permitir asociar casos de prueba a historias de usuario.	*Should*	Sí permite la integración con historias de usuario de Jira.	2

TestRail

Para esta evaluación se utiliza como insumo la información disponible según TestRail (s.f) en su sitio web. En la **Tabla 40** se muestra la evaluación correspondiente a la herramienta TestRail.

Tabla 40

Evaluación de la herramienta TestRail

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
Req1	La herramienta debe permitir crear casos de prueba.	*Must*	Permite crear casos de prueba ilimitados.	2
Req2	La herramienta debe permitir ejecutar casos de prueba manualmente.	*Must*	Permite ejecutar casos de prueba manuales.	2
Req3	La herramienta debe permitir crear conjuntos de pruebas.	*Must*	Sí es posible crear conjuntos de pruebas.	2
Req4	La herramienta debe poder integrarse con Jira.	*Must*	Cuenta con integración con Jira en sus planes *professional* y *enterprise*.	2
Req5	La herramienta debe ser preferiblemente de versión gratuita.	*Should*	El precio mínimo por usuario es de $37 por mes.	0
Req6	La herramienta debe permitir la escalabilidad con respecto al número de usuarios permitidos.	*Must*	Permite aumentar el número de usuario según sea requerido.	2

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
Req7	La herramienta debe permitir la autenticación mediante inicio de sesión único (*Single Sign-On*).	*Should*	Cuenta con soporte para realizar inicio de sesión único en su versión *enterprise*.	1
Req8	La herramienta debe permitir asociar casos de prueba a historias de usuario.	*Should*	Permite enlazar casos y conjuntos de pruebas a historias de Jira.	2

AIO Tests

Para realizar la evaluación de esta herramienta se utiliza la información disponible según AIO Tests (s.f) en su sitio web, así como en el sitio web de Atlassian Marketplace. En la **Tabla 41** se muestra la evaluación correspondiente a la herramienta AIO Tests.

Tabla 41
Evaluación de la herramienta AIO Tests

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
Req1	La herramienta debe permitir crear casos de prueba.	*Must*	Sí permite crear casos de prueba.	2
Req2	La herramienta debe permitir ejecutar casos de prueba manualmente.	*Must*	Se pueden ejecutar los casos de prueba de forma manual.	2
Req3	La herramienta debe permitir crear conjuntos de pruebas.	*Must*	Sí es posible crear conjuntos de pruebas.	2
Req4	La herramienta debe poder integrarse con Jira.	*Must*	Es una integración directamente para Jira.	2
Req5	La herramienta debe ser preferiblemente de versión gratuita.	*Should*	Versión gratuita para hasta 10 usuarios, a partir de los 11 usuarios el	1

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
			precio varía por usuario por mes.	
Req6	La herramienta debe permitir la escalabilidad con respecto al número de usuarios permitidos.	*Must*	Se pueden agregar más instancias según se necesite.	2
Req7	La herramienta debe permitir la autenticación mediante inicio de sesión único (*Single Sign-On*).	*Should*	Realizado simultáneamente con Jira.	2
Req8	La herramienta debe permitir asociar casos de prueba a historias de usuario.	*Should*	Permite enlazar los casos de prueba con incidencias en Jira.	2

AgileTest

Para esta evaluación se utiliza como insumo la información disponible según AgileTest (s.f) en el sitio web de Atlassian Marketplace. En la **Tabla 42** se muestra la evaluación correspondiente a la herramienta AgileTest.

Tabla 42
Evaluación de la herramienta AgileTest

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
Req1	La herramienta debe permitir crear casos de prueba.	*Must*	Es posible crear casos de prueba.	2
Req2	La herramienta debe permitir ejecutar casos de prueba manualmente.	*Must*	Se pueden ejecutar los casos de prueba de forma manual y registrar los resultados obtenidos.	2
Req3	La herramienta debe permitir crear conjuntos de pruebas.	*Must*	Sí, se pueden agrupar casos de prueba en *suites* de prueba.	2

ID Requerimiento	Descripción	Priorización	Comentarios	Puntaje obtenido
Req4	La herramienta debe poder integrarse con Jira.	*Must*	Su desarrollador Devsamurai es un socio de soluciones de Atlassian, por ende, la aplicación cuenta con integración con Jira.	2
Req5	La herramienta debe ser preferiblemente de versión gratuita.	*Should*	Cuenta con una versión totalmente gratuita lista para agregar a Jira.	2
Req6	La herramienta debe permitir la escalabilidad con respecto al número de usuarios permitidos.	*Must*	Al ser una aplicación gratuita solo debe instalarse como complemento a Jira.	2
Req7	La herramienta debe permitir la autenticación mediante inicio de sesión único (*Single Sign-On*).	*Should*	Realizado simultáneamente con Jira.	2
Req8	La herramienta debe permitir asociar casos de prueba a historias de usuario.	*Should*	Sí permite asociar los casos de prueba con incidencias en Jira.	2

Una vez realizada la evaluación de cada una de las herramientas contempladas dentro de la lista posible, se procede a mostrar un resumen con los puntajes obtenidos por cada herramienta.

5.2.1.3. Resumen de la evaluación

Una vez asignadas las puntuaciones a cada una de las herramientas preseleccionadas se realiza el cálculo del puntaje total para cada una de ellas, esto con el fin de identificar cuál es la que obtiene el puntaje máximo y por ende será la herramienta que se priorice para su adquisición como parte del proceso propuesto.

En la **Tabla 43** se muestra cada uno de los requerimientos recopilados junto con la puntuación obtenida por cada herramienta, así como su puntuación final.

Tabla 43

Resumen de la evaluación de las herramientas

ID Requerimiento	Descripción	Priorización	Tuskr	Kualitee	TestRail	AIO Tests	AgileTest
			Puntaje Obtenido				
Req1	La herramienta debe permitir crear casos de prueba.	*Must*	8	8	8	8	8
Req2	La herramienta debe permitir ejecutar casos de prueba manualmente.	*Must*	8	8	8	8	8
Req3	La herramienta debe permitir crear conjuntos de pruebas.	*Must*	8	8	8	8	8
Req4	La herramienta debe poder integrarse con Jira.	*Must*	8	8	8	8	8
Req5	La herramienta debe ser preferiblemente de versión gratuita.	*Should*	3	0	0	3	9
Req6	La herramienta debe permitir la escalabilidad con respecto al número de usuarios permitidos.	*Must*	8	8	8	8	8
Req7	La herramienta debe permitir la autenticación mediante inicio de sesión único (*Single Sign-On*).	*Should*	3	6	3	6	6
Req8	La herramienta debe permitir asociar casos de prueba a historias de usuario.	*Should*	6	6	6	6	6
		Puntaje Total	52	52	49	55	61

5.2.1.4. Selección de la herramienta

De acuerdo con el resumen de los puntajes obtenidos por cada herramienta, documentado en la **Tabla 43**, se aprecia que la herramienta con el máximo puntaje obtenido es AgileTest, ya que cumple con todos los requerimientos especificados, especialmente con los priorizados como *must*, que son los que a solicitud del supervisor de sistemas Pega debe cumplir la nueva herramienta.

Otro aspecto que favorece la selección de la herramienta es el tema de costos, en esta fase se analizó un total de cinco herramientas; asimismo, dentro de los criterios de selección se encuentra un requerimiento sobre el costo de la herramienta.

En la **Tabla 44** se muestran los costos anuales para cada herramienta, cabe destacar que para este cálculo se muestra el costo anual para un total de 150 licencias, debido a que esta es la cantidad de desarrolladores que conforman el departamento de sistemas, banca y procesos.

Tabla 44

Costo de las herramientas

Herramienta	Costo total anual en dólares	Costo total anual en colones
Tuskr	$43,500	₡23.058,915
Kualitee	$16,200	₡8.587,458
TestRail	$34,157	₡18.106,284
AIO Tests	$1,900	₡1.007,171
AgileTest	$0	₡0

Nota. Los precios se muestran en dólares americanos y para realizar la conversión a colones se toma el tipo de cambio actual de ₡530,09 según el Banco Central de Costa Rica (s.f) para la fecha del 12 de octubre del 2023.

Como se puede apreciar, el costo de la herramienta AgileTest es de ₡0, esto reafirma que es la idónea para el proceso propuesto.

Otro de los aspectos que justifica la selección de dicha herramienta es la calificación brindada por los consumidores en el sitio *web* de Atlassian Marketplace, se le brindó una calificación de 3.9 estrellas de 4 posibles, además de contar con gran cantidad de comentarios positivos sobre la funcionalidad de la herramienta.

También cuenta con algunas características que la hacen compatible para ser seleccionada, por ejemplo, cuenta con la certificación *cloud fortified* que ofrece seguridad, confiabilidad y soporte adicionales. Además de lo anterior, cuenta con documentación y una guía de usuario con todos los elementos necesarios para comprender la forma en que funcionan sus características básicas hasta las avanzadas.

En resumen, la herramienta AgileTest además de ser gratuita cumple con todos los requerimientos especificados y es una excelente opción como herramienta inicial para ser utilizada como apoyo al proceso propuesto de aseguramiento de calidad del *software.*

5.3. Fase 4: Plan de implementación

Como bien lo indica el nombre del presente proyecto, este se enfoca en una propuesta de mejora para el proceso de desarrollo y aseguramiento de la calidad del *software*, por ende, en el alcance del proyecto no se contempla su implementación como tal, sin embargo sí se abarca la elaboración de un plan de implementación que sirva como referencia o punto de partida si es que la organización se declina por continuar con la fase de implementación.

De acuerdo con lo establecido en el punto **Fase 4: Plan de implementación**, se propone elaborar este plan para el departamento utilizando lo que dicta el PMBOK en su versión 7, con respecto a los dominios de desempeño de un proyecto. En total, esta guía o conjunto de buenas prácticas para la gestión de proyectos brinda ocho dominios de desempeño los cuales son: interesados, equipo, enfoque de desarrollo y ciclo de vida, planificación, trabajo del proyecto, entrega, métricas y finalmente el domino de la incertidumbre. En la sección señalada anteriormente se indican cuáles de esos ocho dominios se tomaron en cuenta para desarrollar el presente plan de implementación del proyecto.

A continuación, se describen las secciones que contendrá el entregable sobre el plan de implementación sugerido.

- Dominio de desempeño de la entrega

En esta sección se hace referencia a la gestión del alcance del proyecto, es decir, el trabajo que se abarca en el presente proyecto para la organización.

- Dominio de desempeño de interesados

En este apartado se identifican los principales involucrados que forman parte del proyecto planteado, además se identifica la importancia de estos para que el proyecto se pueda implementar exitosamente.

Tabla 45
Lista de involucrados

Interesado	Impacto	Expectativas
Desarrollador de *software*	Alto	Brindar información sobre las actividades realizadas en el proceso de desarrollo y aseguramiento de *software.* Desarrollar las diversas funcionalidades de *software* requeridas.

Interesado	Impacto	Expectativas
Líder técnico	Medio	Encargado de apoyar el proceso de desarrollo y especialmente el de aseguramiento de la calidad del *software* a través del asesoramiento en la creación de casos de prueba.
Proveedor de herramienta de gestión de casos de prueba	Medio	Encargado de vender las licencias para la herramienta de gestión de casos de prueba.
Supervisor de sistemas Pega	Medio	Brindar el visto bueno sobre el proceso planteado y trabajar en su mejora continua.
Analista de QA	Alto	Encargado de apoyar de forma general en el proceso de aseguramiento de la calidad del *software*.
Propietario del producto (Product owner)	Medio	Responsable de gestionar los requerimientos de negocio y las historias de usuario asociados a estos.

- Dominio de desempeño de planificación
  Este punto contiene varios aspectos relevantes para la gestión del proyecto, según el PMBOK (2021), en su versión siete aquí se detallan los siguientes:

  - Cronograma

Para la elaboración del cronograma se toma en cuenta una de las variables de riesgo planteadas para el proyecto como lo es la resistencia al cambio. Por ello se establece que el proyecto será implementado de manera incremental iniciando por la realización de las pruebas unitarias, posteriormente se solicitará la realización de pruebas de integración y finalmente se incorporarán las pruebas end-to-end al proceso de aseguramiento de la calidad del *software*.

Con respecto al tema de implementación de las pruebas de *software* se establece iniciar con las pruebas unitarias, para ello se propone un plan piloto en el cual se realicen en un periodo de tres meses hasta que los desarrolladores alcancen un nivel de madurez, con respecto a estas, y se cumplan un serie de indicadores clave de rendimiento, se mencionan a continuación:

- Porcentaje de reglas testeadas: se debe obtener un porcentaje del 100% de reglas testeadas en cada historia de usuario.
- Porcentaje de casos de prueba aprobados: del total de casos de prueba diseñados para pruebas unitarias se deben ejecutar el 100% de ellos.

A continuación, se indica la forma de calcular estos indicadores clave:

Tabla 46
Indicadores para las pruebas unitarias

Indicador clave de rendimiento	Fórmula
Porcentaje de reglas testeadas	$\left(\frac{Número\ de\ reglas\ testeadas}{Número\ total\ de\ reglas\ de\ la\ historia}\right) * 100$
Porcentaje de casos de prueba aprobados	$\left(\frac{Número\ de\ casos\ de\ prueba\ exitosos}{Número\ total\ de\ casos\ de\ prueba\ ejecutados}\right) * 100$

Seguidamente, como parte de la planificación establecida en el plan piloto se deben continuar con las pruebas de integración, estas se realizan conjuntamente con las pruebas unitarias y se propone un periodo de tres meses para que los equipos de desarrollo las implementen gradualmente y se alcancen los resultados deseados.

Para las pruebas de integración de *software* se cuenta con los siguientes indicadores clave de rendimiento:

Tabla 47
Indicadores para las pruebas de integración

Indicador clave de rendimiento	Fórmula
Porcentaje de componentes probados	$\left(\frac{Número\ de\ componentes\ probados}{Número\ total\ de\ componentes\ de\ la\ aplicación}\right) * 100$
Porcentaje de casos de prueba aprobados	$\left(\frac{Número\ de\ casos\ de\ prueba\ exitosos}{Número\ total\ de\ casos\ de\ prueba\ ejecutados}\right) * 100$
Porcentaje de casos de prueba fallados	$\left(\frac{Número\ de\ casos\ de\ prueba\ fallados}{Número\ total\ de\ casos\ de\ prueba\ ejecutados}\right) * 100$

Una vez integradas las pruebas unitarias y las pruebas de integración, se propone continuar con las pruebas end-to-end de igual forma en un plazo de tres meses para completar los tres tipos de pruebas establecidas en la pirámide de pruebas de *software*.

Para las pruebas end-to-end se proponen los siguientes indicadores clave de rendimiento:

Tabla 48
Indicadores para las pruebas end-to-end

Indicador clave de rendimiento	Fórmula
Porcentaje de casos de prueba aprobados	$\left(\frac{Número\ de\ casos\ de\ prueba\ exitosos}{Número\ total\ de\ casos\ de\ prueba\ ejecutados}\right) * 100$
Porcentaje de casos de prueba fallados	$\left(\frac{Número\ de\ casos\ de\ prueba\ fallados}{Número\ total\ de\ casos\ de\ prueba\ ejecutados}\right) * 100$
Porcentaje de casos de prueba cubiertos	$\left(\frac{Número\ de\ casos\ de\ prueba\ ejecutados}{Número\ total\ de\ casos\ de\ prueba\ planificados}\right) * 100$

Una vez completado el tiempo total del plan piloto se propone el siguiente indicador clave de rendimiento para evaluar si la estrategia de pruebas ha sido efectiva y ha contribuido a la reducción de defectos en el ambiente productivo. La fórmula para calcular dicho indicador contiene los siguientes datos:

A: Número de incidentes antes de la propuesta.

B: Número de incidentes después de la propuesta.

Tabla 49
Indicador general de efectividad

Indicador clave de rendimiento	Fórmula
Porcentaje de reducción de incidentes en el ambiente productivo	$\left(\frac{A-B}{A}\right) * 100$

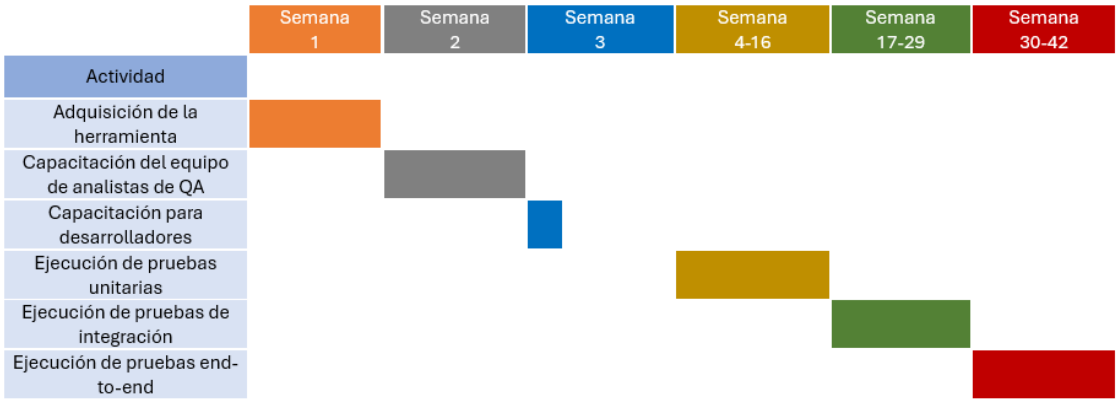
De esta forma, se propone la siguiente distribución para la realización de las actividades como parte del plan piloto, la duración aproximada de estas actividades se validó con el supervisor de sistemas Pega (ver **Apéndice T. Minuta 3**):

- Tiempo de adquisición de la herramienta de apoyo: 1 semana.
- Capacitación del equipo de analistas de QA: 1 semana.
- Capacitación de los desarrolladores de *software*: 2 días.
- Ejecución de pruebas unitarias: 12 semanas.

- Ejecución de pruebas de integración: 12 semanas.
- Ejecución de pruebas end-to-end: 12 semanas.

Seguidamente, en la **Figura 29** se encuentra el diagrama de Gantt para la implementación del proyecto.

Figura 29
Cronograma del proyecto



- Recursos

Para el proyecto se identificaron los siguientes recursos necesarios para su implementación exitosa.

Tabla 50
Recursos del plan de implementación

Recurso	Descripción	Cantidad
Herramienta para gestión de casos de prueba	Herramienta de *software* que servirá de apoyo para el proceso de aseguramiento de calidad gracias a su utilidad para gestionar los casos de prueba de *software* en general.	1 licencia por usuario, para un total de 150 usuarios.
Equipo de analistas de QA	Persona encargada de guiar y apoyar la ejecución del proceso de aseguramiento de calidad del *software* a través de su conocimiento y habilidades técnicas.	1 equipo de 4 analistas de QA

- Presupuesto

En este punto se establecen los costos aproximados incurridos con respecto a la implementación del proyecto.

Tabla 51

Presupuesto aproximado para el proyecto

Concepto	Monto en colones
Salario del desarrollador del proyecto	₡7.084,000
Salario del equipo de analistas de QA	₡42.504,000
Capacitaciones	₡926,800
Herramienta de gestión de pruebas	₡0
Configuración y capacitación en el uso de la herramienta	₡81,180
Otros costos	₡575,000
Total	₡51.170,980

- Comunicación

Para efectos de garantizar una comunicación efectiva entre los involucrados en el proyecto, se establecen los siguientes canales de comunicación oficiales utilizados en la organización:

- Microsoft teams.
- Correo electrónico de Outlook.

Para iniciar con la implementación del proyecto se creará un canal en la plataforma de Microsoft Teams en el cual se agregarán a todos los involucrados y se les comunicará con un mes de anticipación los cambios planteados al proceso. Además, la atención de dudas en general se realizará por medio de este canal.

- Dominio de desempeño de la incertidumbre

En esta sección se realiza la gestión de los principales riesgos identificados para el proyecto. De manera general se identificaron los siguientes riesgos.

Tabla 52

Riesgos identificados en el plan de implementación

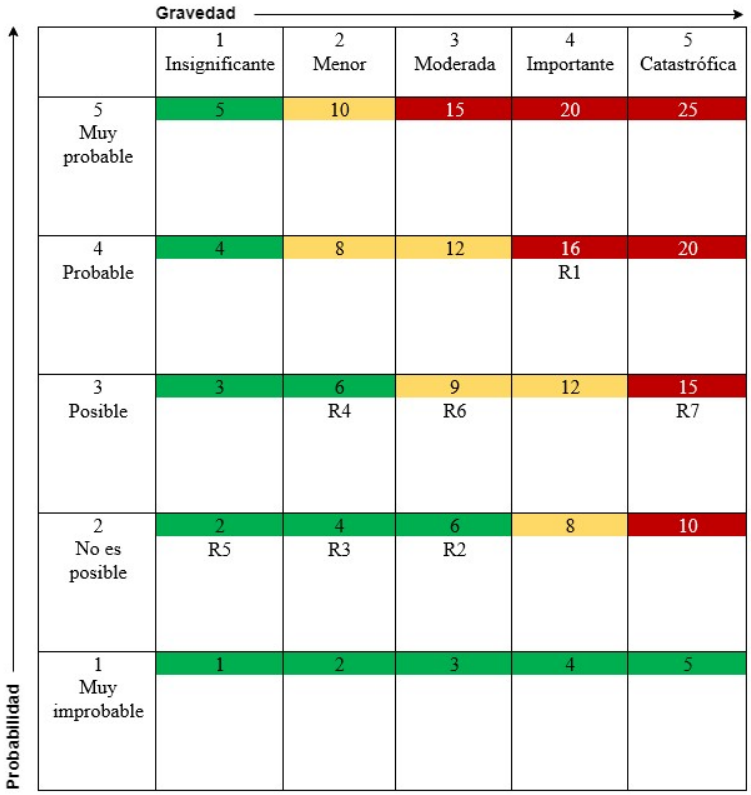
ID del riesgo	Riesgo	Descripción
R1	Resistencia al cambio	Temor, incomprensión, o inadaptación de algunos involucrados, estos pueden mostrar rechazo a estos nuevos cambios.
R2	Retraso en la planificación propuesta	Tiempo adicional que se puede tomar para implementar el proyecto.
R3	Dificultad en el uso de herramientas propuestas	Existe la posibilidad de que la herramienta de *software* propuesta sea nueva para los

ID del riesgo	Riesgo	Descripción
		desarrolladores de *software* y por ende se les dificulte su uso.
R4	Problemas de comunicación	Este riesgo se refiere a la existencia de problemas para comunicar la implementación del proyecto a los involucrados.
R5	Problemas de integración de la herramienta	Se debe verificar si la herramienta de *software* propuesta cuenta con el visto bueno para ser instalada en los equipos.
R6	Aumento en el costo de la herramienta	El precio de la herramienta de *software* propuesta puede aumentar con respecto a lo presupuestado.
R7	Viabilidad económica de realizar el proyecto.	Existe la posibilidad de que la propuesta planteada no se adapte al presupuesto de la organización.

En la **Figura 30** se muestra la matriz de riesgos del proyecto.

Figura 30

Matriz de riesgos del plan de implementación



Para estos riesgos identificados se elabora un plan de respuesta como se muestra en la **Tabla 53**.

Tabla 53
Plan de acción para los riesgos

ID del riesgo	Riesgo	Acción propuesta
R1	Resistencia al cambio	Para ellos se propone un plan de implementación por fases que permita adaptar a los involucrados en el proceso de cambio.
R2	Retraso en la planificación propuesta	Se debe contar con un margen de tiempo para la implementación del proyecto por si se dificulta realizar la implementación en la fecha propuesta.
R3	Dificultad en el uso de herramientas propuestas	Se debe brindar capacitación sobre el uso de la herramienta de *software* que facilite el uso de esta.
R4	Problemas de comunicación	Se contempla la gestión de la comunicación en el plan de implementación del proyecto, para ello se especifican los medios necesarios para comunicar el proyecto.
R5	Problemas de integración de la herramienta	Se debe contar con el visto bueno por parte del área de seguridad para realizar la implementación de la herramienta en los equipos de la organización.
R6	Aumento en el costo de la herramienta	Se tienen varias opciones de herramientas disponibles para adquirir, por ende, si el costo de la herramienta aumentara se evaluará migrar a otra.
R7	Viabilidad económica de realizar el proyecto.	Se evaluará con el supervisor de sistemas Pega diversas opciones para la implementación del proyecto, evaluando la que se ajuste al presupuesto disponible para la implementación del proyecto.

5.4. Fase 5: Análisis de costo-beneficio del proyecto

Una vez analizados y descritos todos los aspectos que formarán parte de la nueva propuesta para el proceso de desarrollo y aseguramiento de la calidad del *software*, se necesita realizar un análisis desde la perspectiva financiera sobre la viabilidad económica del proyecto, aspecto que será de utilidad para la organización, para tomar la decisión final con respecto a continuar con la fase de implementación del proyecto.

A continuación, se procede a realizar un desglose y cálculo de los costos identificados para el proyecto, así como de los posibles ingresos que serán obtenidos producto de la mejora en el proceso actual.

Es importante indicar que algunos de los datos utilizados para realizar este cálculo son de origen confidencial, por ende, se realizan tan solo algunos cálculos para no exponer dicha información por solicitud de la organización.

5.4.1. Determinación de costos e ingresos

Primeramente, se procede a realizar el cálculo de los costos asociados para el proyecto, seguidamente se calculan los ingresos posibles para finalmente realizar el análisis financiero de la propuesta.

5.4.1.1. Costos

A continuación, se presentan los principales costos identificados producto de la implementación del proyecto.

- Desarrollador del proyecto

Este costo viene a representar el salario pagado al estudiante por concepto de la realización del proyecto. Como se indicó anteriormente, por cuestiones de confidencialidad, para representar el salario pagado al estudiante se realizó un cálculo utilizando el promedio de salarios pagados en tres puestos de diferentes categorías para desarrolladores de *software*, con lo cual se obtuvo que el salario mensual pagado al estudiante es de aproximadamente ₡1.771,000 mensuales con el monto incorporado correspondiente a las cargas sociales.

En la **Tabla 54** se muestra el costo total referente al salario del estudiante a lo largo de las 16 semanas correspondientes a la duración del proyecto.

Tabla 54
Costos por salario del estudiante

Total de salario mensual	Cantidad de meses de duración del proyecto	Costo total
₡1.771,000	4	₡7.084,000

- Equipo de analistas de QA

Como parte de las mejoras propuestas para el proceso actual de desarrollo y aseguramiento de la calidad del *software* se encuentra la implementación de un departamento dedicado al tema de aseguramiento de la calidad.

Como se evidencia en el **Apéndice U. Minuta 4**, este tema fue conversado con el supervisor de producto de *software* para validar si la idea era factible. Resultado de la reunión este indica que

sí es una de las posibles opciones que tienen contempladas, pero en lugar de crear un departamento como tal piensan en la vertiente de formar un equipo de aseguramiento de calidad con personas propias del departamento, dedicadas a medio tiempo a tareas relacionadas con estos temas.

Complementando lo mencionado anteriormente, en esa misma sesión se indica que el equipo estará formado inicialmente por 4 personas, debido a esto, los costos relacionados con los salarios del equipo de analistas de QA se realiza tomando como base esta cantidad de sujetos. Otro punto importante con respecto a este cálculo es que el salario de cada analista se realiza similar a lo efectuado con el salario del estudiante que desarrolla el proyecto, es decir, se calcula un promedio con tres categorías de salarios para el puesto de desarrollador de *software*.

Por lo tanto, como cada analista estará dedicado un 50% de la jornada a este rol el salario aproximado para cada analista de aseguramiento de calidad de *software* será de ₡885,500 mensuales, a este monto ya se le sumó el correspondiente a las cargas sociales.

En la **Tabla 55** se muestran los costos totales por concepto de salarios para el equipo de analistas de aseguramiento de la calidad del *software*.

Tabla 55
Costos por salarios del equipo de analistas de QA

Total de salario mensual	Cantidad de analistas de aseguramiento de calidad	Costo Total anual
₡885,500	4	₡42.504,000

- Capacitaciones

Con respecto a este criterio es necesario recalcar que el equipo de analistas de aseguramiento de calidad de *software* por conformar necesita recibir capacitación con respecto al tema de aseguramiento de la calidad, por ende, se propone que el equipo reciba una capacitación de cuatro horas por día durante una semana, por parte de un experto en el tema de aseguramiento de la calidad del *software*.

Para ello, se procede a investigar cuál es el salario aproximado de un analista de aseguramiento de calidad de *software* senior. Según Glassdoor (2023), en su sitio web expone que el salario para un ingeniero *senior* de aseguramiento de la calidad del *software* es de aproximadamente ₡1.460,000 mensuales. Para estimar los salarios este sitio *web* se basa en la cantidad de salarios de esa categoría que son enviadas de forma anónima al sitio *web* de Glassdoor.

Para estimar el costo por el concepto de capacitaciones para los integrantes del equipo de aseguramiento de calidad, se debe calcular el aproximado de lo que gana por hora el ingeniero *senior* de aseguramiento de la calidad del *software*, una vez se tiene este monto se toma como costo por hora por concepto de capacitación de un experto en el tema.

En la **Tabla 56** se muestran las conversiones realizadas para calcular el costo total por concepto de capación al equipo de analistas de QA.

Tabla 56
Costo de capacitación para el equipo de analistas de QA

Costo por hora	Total de horas de capacitación	Costo total
Salario mensual: ₡1.460,000 Salario diario: ₡1.460,000/30 = ₡48,666 Salario por hora: ₡48.666/8 = ₡6,083	4 horas por día * 5 días = 20 horas	Cantidad de horas * Costo por hora 20 * ₡6,083 = ₡121,000

Siguiendo con el tema de costos por capacitaciones, se debe estimar el costo relacionado con las capacitaciones brindadas a los desarrolladores de *software* que tendrán que adaptarse al nuevo proceso de desarrollo y aseguramiento de la calidad del *software*.

Para lo anterior, se propone hacer cinco sesiones de 3 horas mediante la plataforma *teams*, de igual forma con el equipo de aseguramiento de calidad de *software* creado, para explicar todos los aspectos de la nueva metodología. Entonces por las tres sesiones será un total de 15 horas de capacitación, el costo por hora de capacitación se toma del costo por hora del salario de una analista de aseguramiento de calidad calculado anteriormente en el punto de cálculo de costos para el equipo de analistas de QA.

Una vez aclarado lo anterior, en la **Tabla 57** se muestran los costos relacionados con la capacitación para los desarrolladores de *software*.

Tabla 57
Costo de capacitación para los desarrolladores

Costo por hora	Total de horas de capacitación	Costo total
Salario mensual: ₡1.771,000 Salario diario: ₡1.771,000/30 = ₡59,000 Salario por hora: ₡59,000/8 = ₡7,380	15 horas	Cantidad de horas * Costo por hora 15 * ₡7,380 = ₡110,700

Una vez analizados los datos anteriores por concepto de capacitación tanto para el equipo de analistas de QA como para los desarrolladores de *software*, se concluye que el costo total por concepto de capacitaciones será de aproximadamente ₡231,700.

Al ser el tema de aseguramiento de calidad de *software* un proceso importante dentro del departamento, además de la capacitación inicial se proponen realizar otras tres capacitaciones para el resto del año, por lo tanto en el año se brindarán un total de cuatro capacitaciones que tendrán un costo aproximado de ₡926,800.

- Herramienta de *software*

Como se analizó en la **Fase 3: Selección de herramienta de apoyo al proceso**, como punto de mejora para el proceso planteado se evalúa la opción de adquirir una herramienta de gestión de pruebas de *software*.

De acuerdo con la sección de **Selección de la herramienta**, se realiza un análisis de los costos de cada herramienta y se determina que la seleccionada, es decir AgileTest tiene un costo de ₡0, por ende no representa un costo para el proyecto.

- Costos de capacitación y configuración de la herramienta

Se deben tomar en cuenta los costos relacionados con la capacitación para el uso y la configuración de la herramienta para la gestión de casos de pruebas. Con respecto al tema de configuración de la herramienta se deben realizar algunas configuraciones y personalizaciones según lo necesite la organización. Al ser una herramienta genérica para la gestión de casos de prueba, esta viene preconfigurada para ser utilizada sin realizar muchas modificaciones, por lo que se estima que estas configuraciones podrían tardar aproximadamente un día en configurarse para todas las instancias, lo que podría tener un costo aproximado de ₡59,040, esto tomando en cuenta las ocho horas productivas y que el costo por hora según los cálculos realizados es de aproximadamente ₡7,380.

Con respecto a los costos por concepto de capacitación sobre el uso de la herramienta se estima que se debe realizar una sesión de capacitación de una duración de alrededor de tres horas, lo que tendría un costo aproximado de ₡22,140.

De esta forma, los costos totales por concepto de configuración de la herramienta y capacitación en el uso de esta serían de ₡81,180.

- Otros costos

Como parte de la propuesta también se deben evaluar otros costos indirectos que se deben contemplar y que influyeron para el desarrollo del proyecto. Estos costos son aproximados y se obtuvieron de sitios *web* que comercializan dichos productos. En la siguiente lista se muestran los costos identificados para esta categoría:

  - Laptop Lenovo ThinkPad L14: ₡450,000.
  - Monitor externo Gigabyte: ₡100,000.
  - Mouse Dell: ₡10,000.
  - Teclado Havit: ₡15,000.

De esta forma el monto total por concepto de otros costos es de ₡575,000.

- Costos totales

Una vez realizados todos los cálculos correspondientes a costos individuales del proyecto se pueden obtener los costos totales, como se muestra en la **Tabla 58**.

Tabla 58
Costos totales del proyecto

Concepto	Monto en colones
Salario del desarrollador del proyecto	₡7.084,000
Salario del equipo de analistas de QA	₡42.504,000
Capacitaciones	₡926,800
Herramienta de gestión de pruebas	₡0
Configuración y capacitación en el uso de la herramienta	₡81,180
Otros costos	₡575,000
Total	₡51.170,980

5.4.1.2. Ingresos

Una vez realizado el cálculo de los costos del proyecto se procede a elaborar el cálculo de los posibles ingresos que se obtendrán con el proyecto. Para estimar los posibles ingresos producto de la realización del proyecto se realiza una reunión con el supervisor de sistemas Pega que se encuentra documentada en el **Apéndice T. Minuta 3**, en esta se indica que una posible forma de identificar los ingresos es calculando los costos en que incurre la organización producto de la resolución de incidentes presentados en el ambiente de producción causados por la falta de pruebas ejecutadas por los equipos.

En dicha sesión se analiza que en promedio por semana se registran 15 incidentes, estos se relacionan con un impacto negativo de los pases instalados, asimismo, corresponden a causas de escenarios de pruebas no contemplados que impactan las funcionalidades que los otros equipos ya han certificado en el ambiente de producción.

De acuerdo con el supervisor de sistemas Pega, para resolver cada incidente presentado en el ambiente de producción se deben destinar aproximadamente 18 horas efectivas para resolución, este total de horas se divide entre el tiempo que toma identificar el incidente, el tiempo en que se encuentra el defecto en la funcionalidad implementada en producción, las horas de trabajo para la corrección del defecto y pruebas en ambientes menores, sumado al tiempo sobre los protocolos que toma hacer el nuevo despliegue hacia producción y las pruebas de certificación en dicho ambiente.

El costo por esas 10 horas se puede calcular utilizando como parámetro el monto por hora que se le paga a un desarrollador, este es de aproximadamente ₡7,380, cálculo que se realizó anteriormente tomando en cuenta el salario más las cargas sociales.

En la **Tabla 59** se muestran los cálculos realizados para obtener el monto aproximado para los ingresos del proyecto.

Tabla 59
Costos aproximados para resolver incidentes

Cantidad de incidentes por semana	Total de horas para resolver los incidentes por semana	Costo semanal para resolver incidentes	Costo mensual para resolver incidentes	Costo anual para resolver incidentes
15	15 incidentes *10 horas = 150 horas	150 horas * ₡7,380 = ₡1.107,000	₡1.107,000 * 4 = ₡4.428,000	₡4.428,000 * 12 = ₡53.136,000

Como se mencionó anteriormente, estos costos para resolver incidentes se tomarán como posibles ingresos para el proyecto, debido a que esta fue la forma disponible en este momento para calcularlos.

5.4.2. Cálculo de indicadores financieros

En este punto se realizan algunos análisis desde la perspectiva financiera para determinar si el proyecto es viable económicamente. Los cálculos que se realizan son el cálculo del retorno sobre la inversión (ROI) y el plazo de recuperación de la inversión.

Primeramente se necesitan calcular los flujos de efectivo de cada año del proyecto, en este caso se toman tres años como proyección para evaluar su factibilidad. Como se puede observar en la **Tabla 60**, el salario del desarrollador del proyecto se toma sólo en cuenta para el año 1, ya que solo en ese año se incurre en ese gasto, además la herramienta no tiene costo según lo demostrado en la sección de cálculo de **Costos**.

Con respecto a los costos por concepto del salario del equipo de analistas de QA se establece que para el primer año será de aproximadamente ₡38.742,750 ya que se descuenta el monto por el tiempo de desarrollo de la propuesta (16 semanas) y el tiempo de adquisición de la herramienta (1 semana), lo que representa un aproximado de ₡3.761,250. Para el resto de los años, sería un monto fijo de ₡42.504,000 por año.

Adicionalmente, se planean realizar cuatro capacitaciones tanto a los desarrolladores de *software* como al equipo de analistas de QA por año, por ende se contempla este monto para cada año.

Tabla 60
Flujos de efectivo

Concepto	Año 1	Año 2	Año 3
Salario del desarrollador del proyecto	₡7.084,000	-	-
Salario del equipo de analistas de QA	₡38.742,750	₡42.504,000	₡42.504,000
Capacitaciones	₡926,800	₡926,800	₡926,800
Herramienta de gestión de pruebas	₡0	-	-
Configuración y capacitación en el uso de la herramienta	₡81,180	-	-
Otros costos	₡575,000	-	-
Total	₡47.409,730	₡43.430,800	₡43.430,800
Total general		₡134.271,330	

- Retorno sobre la inversión

Para realizar el cálculo del ROI se toma el ingreso calculado anteriormente, una observación importante es que para el primer año del proyecto no se cuentan los ingresos para las primeras 19 semanas debido a que en estas se realizan las actividades de desarrollo del proyecto, adquisición de la herramienta de *software* y la capacitación del equipo de analistas y desarrolladores, por ende en ese tiempo no se percibirán posibles ingresos, sino sería hasta que se implemente el proceso de aseguramiento de calidad de *software* con sus respectivos tipos de pruebas.

Siendo así los ingresos para el primer año serían de aproximadamente de ₡32.103,000 y para el resto de los años de ₡53.136,000 por año. De esta forma, los posibles ingresos para los primeros tres años de implementación del proyecto serían de ₡32.103,000 + ₡106.272,000, para un total de ₡138.375,000.

Para el cálculo de la fórmula del retorno sobre la inversión se tienen los siguientes datos:

- Ingresos: ₡138.375,000
- Inversión: ₡134.271,330

Con estos datos se puede calcular el ROI utilizando la fórmula:

- ROI = [(ingresos – inversión) / inversión] x 100
- ROI = [(₡138.375,000– ₡131.000,030) / ₡131.000,030] x 100
- ROI = 3,05%

Como se puede notar, el resultado del cálculo del ROI fue de un 3,05% lo que significa que el proyecto será rentable para la organización en un término de tres años.

- Plazo de recuperación de la inversión

Esta fórmula permite identificar el tiempo que tardará la organización en recuperar el monto invertido al inicio del proyecto. La fórmula para calcular este plazo es la siguiente:

PRI = a + (b – c) / d

En donde:

- A: año anterior al que se recupera la inversión.
- B: inversión inicial del negocio.
- C: flujo de efectivo anual del año anterior al que se recupera la inversión.
- D: flujo de efectivo anual del año en el que se recupera la inversión.

Para el proyecto actual se cuenta con los siguientes datos:

- A: 2
- B: ₡134.271,330
- C: ₡85.239,000
- D: ₡53.136,000

De acuerdo con estos datos, el periodo de recuperación de la inversión sería el siguiente:

- PRI = a + (b – c) / d
- PRI = 2 + (₡134.271,330 – ₡85.239,000) / ₡53.136,000
- PRI = 2,9227704381

Por lo tanto, el periodo de recuperación de la inversión para el proyecto es de aproximadamente 2,92 años, es decir 2 años y 11 meses y 2 días.

Una vez realizado el análisis costo-beneficio para el proyecto, se culmina el contenido del capítulo V correspondiente a la propuesta de solución del proyecto. De esta forma, seguidamente se desarrolla el capítulo VI que contiene las conclusiones del proyecto.

6. Capítulo 6: Conclusiones

En este capítulo se muestran las conclusiones obtenidas una vez finalizada la presente investigación. Estas se encuentran agrupadas según los objetivos planteados.

Objetivo específico 1: Analizar la situación actual del proceso de desarrollo y aseguramiento de la calidad de *software* dentro del departamento para el entendimiento de la forma en que se realiza actualmente dicho proceso.

- No existe un proceso formal para el desarrollo y aseguramiento de la calidad del *software*, por lo tanto tampoco existe documentación con respecto a dicho proceso, esto evidencia que existe la necesidad de incorporar un proceso estandarizado que esté correctamente documentado.
- Se identificaron un total de ocho defectos o puntos de mejora con respecto al proceso actual de desarrollo y aseguramiento de calidad del *software*, esto se logra gracias a las entrevistas realizadas tanto a los desabolladores de *software*, líderes técnicos, supervisor de producto de *software* y supervisor de sistemas Pega.
- El ochenta por ciento de los desarrolladores entrevistados indicó que no tienen conocimiento sobre los diversos tipos de pruebas de *software* mencionadas en las entrevistas aplicadas, es importante mencionar que los tipos de pruebas que fueron consultadas fueron: pruebas unitarias, pruebas integrales y pruebas *end-to-end.*
- Al diagramar el proceso *as-is* de desarrollo y aseguramiento de la calidad del *software* se evidencia que con respecto a la fase de recopilación de requerimientos no se siguen buenas prácticas para este proceso, por lo cual es importante para tomar en cuenta para el proceso *to-be*.

Objetivo específico 2: Rediseñar el proceso de desarrollo y aseguramiento de la calidad del *software* en el departamento para la estandarización de este utilizando las buenas prácticas de la industria.

- Para la generación del proceso *to-be* se analizaron un total de cuatro buenas prácticas o metodologías con respecto al proceso de pruebas de aseguramiento de la calidad del *software*, todas estas buenas prácticas compartían la mayoría de las actividades. Además se incluyeron buenas prácticas para realizar el proceso de ingeniería de requerimientos de *software*, estos debido a las recomendaciones brindadas.
- Para la selección de la herramienta de *software* que apoyará al proceso propuesto se evaluaron un total de cinco opciones posibles, pero se seleccionó la herramienta AgileTest ya que obtuvo el puntaje más alto posible, además que cumplía el principal requerimiento con respecto al costo de la herramienta.
- Al proceso propuesto de pruebas de aseguramiento de calidad del *software* se le agregaron tres subprocesos los cuales son: planeación de las pruebas, diseño de las

pruebas y ejecución de las pruebas. Además se incluyó un subproceso para el tema de ingeniería de requerimientos de *software*.

- Se incluyó un equipo con cuatro integrantes con el rol de analista de QA para gestionar el tema de aseguramiento de la calidad del *software* en el departamento, esto con respecto a lo previsto por el supervisor de producto de *software* y lo que busca el departamento de sistemas banca y procesos de la organización.

Objetivo específico 3: Desarrollar una propuesta de implementación del proceso de aseguramiento de calidad del *software* para la gestión de los recursos asociados a este.

- Para el plan de implementación del nuevo proceso se pusieron en práctica tres dominios de desempeño propuestos por el PMBOK para la gestión de proyectos, estos son: involucrados, planificación e incertidumbre. Estos tres dominios de desempeño se tomaron en cuenta debido a que tiene un enfoque en actividades relacionadas con la etapa de planificación del proceso, los demás dominios hacen un énfasis en la ejecución de un proyecto como tal.
- Se obtuvo un retorno sobre la inversión favorable de 3,05% en un cálculo realizado utilizando como proyección una cantidad de tres años a partir del inicio de la implementación del proceso propuesto. Además con respecto al cálculo del periodo de recuperación de la inversión se obtuvo un periodo de aproximadamente de 2 años y 11 meses y 2 días.
- Para la identificación de los costos del proyecto se realizaron una serie de reuniones con el supervisor de producto de *software* para tomar en cuenta de manera general todos aquellos costos del proyecto y obtener montos aproximados para los cálculos posteriores.
- Con respecto al cálculo de los ingresos por concepto de una posible implementación del proyecto la única forma viable que se encontró para realizar dicho cálculo fue realizar una suposición con respecto a los costos en que se incurren por solucionar temas relacionados con la deficiencia en el proceso de pruebas de aseguramiento de la calidad del *software*. Lo anterior se debe a la imposibilidad de calcular tiempos de ejecución y realizar posibles simulaciones de ejecución de actividades.

Una vez presentadas las conclusiones obtenidas para los objetivos del proyecto, se procede a desarrollar el capítulo VII correspondiente a las recomendaciones generales propuestas para el proyecto, estas se redactan en función de las conclusiones descritas anteriormente.

7. Capítulo 7: Recomendaciones

A continuación, se indican algunas recomendaciones que podrían aplicarse en un futuro al proceso de desarrollo y aseguramiento de calidad del *software* para el departamento de Sistemas Banca y Procesos.

- Primeramente se sugiere que se ejecute el plan de implementación elaborado para el proceso propuesto y evaluar la efectividad del nuevo proceso diseñado para el departamento, esto permitirá saber si con el plan piloto propuesto será factible realizar una mayor inversión con respecto a lo planteado o si por el contrario se deben costos.
- Se recomienda contemplar la posibilidad de agregar o actualizar los indicadores claves de rendimiento propuestos, esto con el fin de validar correctamente si el proceso recomendado está brindando los beneficios esperados para el departamento. De igual forma se recomienda actualizar estos indicadores cuando el proceso haya alcanzado una mayor madurez y la gestión del cambio esté gestionado, de esta forma se notará de una forma medible si el proceso ha proporcionado los beneficios esperados.
- Se recomienda si fuera posible evaluar el nivel de madurez del departamento con respecto al proceso de aseguramiento de calidad del *software* a través de un modelo de verificación como el TMMI (*Testing Maturity Model Integrated*) que permite comprobar el nivel de mejora en el proceso de pruebas de *software*. Este modelo de evaluación de la madurez permitirá determinar si el proceso se encuentra gestionado o no, además de brindar una calificación en una escala del 1 al 5.
- Se sugiere analizar la posibilidad de adquirir más talento humano que forme parte del equipo de QA en el departamento, además de brindar capacitaciones continuas a los desarrolladores para aumentar su conocimiento con respecto al tema de pruebas y aseguramiento de la calidad del *software*.
- Adicionalmente, se recomienda certificar a los integrantes del equipo de aseguramiento de calidad con certificaciones internacionales, por ejemplo las brindadas por el ISTQB (*International Software Testing Qualifications Board*), siempre que sea posible realizar dicha inversión, esto brindará formalidad al tema de aseguramiento de la calidad en el departamento y reforzará los conocimientos de los involucrados en el proceso.
- Se recomienda realizar los esfuerzos e inversión necesarios para buscar la mejora continua para el proceso de desarrollo y aseguramiento de la calidad del *software*, así como trabajar en la automatización de pruebas de *software* tan pronto como sea posible para reducir tiempos y costos en el proceso. Esto ayudará a abarcar una de las posibilidades de mejora identificadas y a su vez atacar uno de los defectos encontrados.

- Se recomienda contratar talento humano que se dedique a tiempo completo al tema de aseguramiento de la calidad del *software*, ya que esto permitirá dar el apoyo necesario a todos los equipos de desarrollo de *software*. Además se recomienda incluir un rol para analista de requerimientos de *software* que puedan apoyar el proceso propuesto de ingeniería de requerimientos.
- Como recomendación se propone como punto de mejora para el proceso propuesto implementar un proceso dedicado al aseguramiento de calidad en la etapa de diseño del *software,* ya que las buenas prácticas de la industrian contemplan este proceso como parte de los aspectos del aseguramiento de calidad del *software.* De igual forma, se recomienda hacer esfuerzos para mejorar el proceso propuesto de ingeniería de requerimientos de *software*.
- Se propone incluir un proceso final para el control de calidad del *software* que permita evaluar la calidad del producto final elaborado, ya que el proceso actual tiene como objetivo realizar el aseguramiento de calidad, que tiene como función el prevenir defectos durante la etapa de desarrollo de *software*.

8. Capítulo 8: Referencias

AgileTest. (s.f). *Test Management for Jira.* https://agiletest.app/

Angeli, J. (2018). *¿Qué es el mapeo de procesos AS IS/TO BE?*. Neomind. https://www.neomind.com.br/es/blog/que-es-el-mapeo-de-procesos-as-is-to-be/#:~:text=El%20Mapeo%20de%20procesos%20AS%20IS%20%2F%20TO%20BE%20es%20una,actividades%20del%20d%C3%ADa%20a%20d%C3%ADa.

Asana. (2022). *Matriz de riesgos: cómo evaluar los riesgos para lograr el éxito del proyecto (incluye ejemplos)*. Asana. https://asana.com/es/resources/risk-matrix-template

Atlassian. (s.f). *AIO Tests: QA Testing and Test Management in Jira.* Atlassian. https://marketplace.atlassian.com/apps/1222843/aio-tests-qa-testing-and-test-management-in-jira?hosting=cloud&tab=overview

Atlassian. (s.f). *Agile Test & Enterprise Test Management for Jira.* *https://marketplace.atlassian.com/apps/1230085/agile-test-management-for-jira?hosting=cloud&tab=overview*

Banco Central de Costa Rica. (s.f). *Tipo cambio de compra y de venta del dólar de los Estados Unidos de América.* Banco Central de Costa Rica. https://gee.bccr.fi.cr/indicadoreseconomicos/Cuadros/frmVerCatCuadro.aspx?idioma=1&CodCuadro=%20400

Cohn, M. (2009). *Succeeding with Agile: Software Development Using Scrum*. (1. ª ed.). Addison-Wesley Professional.

Dumas, M., Rosa, L. M., Mendling, J. & Reijers, H. A. (2018). *Fundamentals of Business Process Management (2.a ed.)*. Springer.

Eurofins. (2023). *El ciclo Deming: en qué consiste y cómo ayuda en la gestión y mejora de procesos.* Eurofins. https://www.eurofins-environment.es/es/el-ciclo-deming-que-consiste-y-como-ayuda-gestion-procesos/

Faro, I. (2022). *Los estándares de calidad del software más importantes.* Iberus Blog. https://www.hiberus.com/crecemos-contigo/los-estandares-de-calidad-del-*software*-mas-importantes/

Faro, I. (2022). *Niveles de certificación en ISTQB. ¿Cuáles son?*. Iberus Blog. https://www.hiberus.com/crecemos-contigo/niveles-de-certificacion-en-istqb-cuales-son/

G2. (s.f). *Best Test Management Tools for Medium-Sized Businesses.* G2. https://www.g2.com/categories/test-management/mid-market

Glassdoor. (2023). *Sueldos para Senior Software Test Engineer en Costa Rica.* Glassdoor. https://www.glassdoor.com.mx/Salaries/costa-rica-senior-*software*-test-engineer-salary-SRCH_IL.0,10_IN57_KO11,40.htm

Grupo Financiero. (2023).

Good, F. (2023). *What Is a RACI Matrix?*. Project Management. https://project-management.com/understanding-responsibility-assignment-matrix-raci-matrix/

Hamilton, T. (2023). *STLC (Software Testing Life Cycle) Phases, Entry, Exit Criteria.* Guru99. https://www.guru99.com/*software*-testing-life-cycle.html

Hamilton, T. (2023). *What is Software Testing? Definition.* Guru99. https://www.guru99.com/*software*-testing-introduction-importance.html

Hernández, R., Fernández, C., & Baptista, P. (2014). *Metodología de la investigación*. McGrawHill Education.

IEEE Computer Society. (2014). *Guide to the Software Engineering Body of Knowledge (SWEBOK(R)): Version 3.0*. IEEE Computer Society Press.

Institute of Electrical and Electronics Engineers. (2014). *IEEE 730-2014 - IEEE Standard for Software Quality Assurance Processes.* IEEE.

ISO/IEC. (s.f). *ISO/IEC 25010.* ISO 25000. https://iso25000.com/index.php/normas-iso-25000/iso-25010

Kualitee. (s.f). *One testing platform to rule them all.* Kualitee. https://www.kualitee.com/

Lucidchart. (s.f). *Qué es la notación de modelado de procesos de negocio.* Lucidchart. https://www.lucidchart.com/pages/es/que-es-la-notacion-de-modelado-de-procesos-de-negocio

Machuca, F. (2022). *8 técnicas de recolección de datos: descubre un mundo más allá de la encuesta.* Crehana. https://www.crehana.com/blog/transformacion-digital/tecnicas-recoleccion-de- datos/

Madison, D. (2005). *Process Mapping, Process Improvement and Process Management: A Practical Guide to Enhancing Work Flow and Information Flow*. Paton Professional.

Mata, L. (2019). *El enfoque de investigación: la naturaleza del estudio.* Investigalia. *https://investigaliacr.com/investigacion/el-enfoque-de-investigacion-la-naturaleza-del-estudio/#:~:text=Cuando%20hablamos%20de%20enfoque%20de,el%20desarrollo%20de%20la%20perspectiva*

Moreno, J. (2023). *Procesos de Negocios: La guía completa.* Flokzu. https://flokzu.com/es/procesos-negocios-guia-completa/

Page, S. (2015). The Power of Business Process Improvement: 10 Simple Steps to Increase Effectiveness, Efficiency, and Adaptability. AMACOM.

Palacio, D. (2020). *Propuesta de una metodología de aseguramiento y control de calidad para los proyectos de software de Inclutec* [Trabajo Final de Graduación, Tecnológico de Costa Rica].

Pierce, A. (2022). *Mejora de procesos utilizando As Is & To Be.* Imagineer. https://blog.imagineer.co/es/proceso-de-negocio/proceso-de-negocio/mejora-de-procesos-utilizando-as-is-to-be

Project Management Institute. (2022). *Guía de los fundamentos para la dirección de proyectos*. (7.ª ed.). Project Management Institute, Inc.

Reyes, E . (2022). *Definición de proceso según autores.* Emprendedor Inteligente. https://www.emprendedorinteligente.com/definicion-de-proceso-segun-autores/

Rojas Pavón, A. R. (2016). *Desarrollo de una guía de un marco de referencia de calidad para la metodología de desarrollo de software ágil Scrum* (Bachelor's thesis, PUCE).

Software Testing Help (2023). *What Is Software Quality Assurance (SQA): A Guide For Beginners. Software* Testing Help. https://www.*software*testinghelp.com/*software*-quality-assurance/

Solano, F. (2019). *¿Qué es un proceso de negocio?*. Imagineer. https://blog.imagineer.co/es/proceso-de-negocio/proceso-de-negocio/que-es-un-proceso-de-negocio

Spillner, A., Linz, T., & Schaefer, H. (2014). *Software testing foundations*. (4th ed.). Santa Barbara: Rocky Nook.

Sydle. (2023). *Procesos de negocio: ¿Qué son y cómo modelarlos? Ejemplos.* Sydle. https://www.sydle.com/es/blog/que-son-procesos-de-negocio-610afc74504afa7e3653c2c3

Sydle. (2023). *Ciclo PDCA: ¿cuáles son los pasos y cómo funciona? Conoce algunos ejemplos*. Sydle. https://www.sydle.com/es/blog/ciclo-pdca-61ba2a15876cf6271d556be9

TestRail. (s.f). *Orchestrate Your Entire QA Process in One Centralized Platform.* TestRail. https://www.testrail.com/platform/

Tiann. J. (2005). *Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement.* (1.ª ed.). Wiley-IEEE Press.

Tuskr. (s.f). *A complete test management tool.* Tuskr. https://tuskr.app/

VanZandt. (2023). What is a MoSCoW Analysis? Definition, Use Guide, and Analysis. IdeaScale. https://ideascale.com/blog/moscow-analysis-definition/#

9. Capítulo 9: Apéndices

Apéndice A. Plantilla de minutas

Minuta de Reunión	
Información General	
Número de Minuta:	
Fecha:	
Participantes:	
Objetivos	
Temas tratados	
Observaciones/Comentarios	

Apéndice B. Plantilla de entrevista

Entrevista	
Número de entrevista:	
Fecha:	
Objetivo:	
Participantes	
Rol dentro del proyecto	
Preguntas	
Pregunta	Respuesta

Apéndice C. Plantilla de revisión documental

Revisión documental	
Objetivo	
Fuente	
Principales hallazgos	

Apéndice D. Plantilla de observación

Observación del proceso	
Fecha	
Objetivo	
Proceso observado	
Principales hallazgos	
Comentarios	

Apéndice E. Entrevista 1

Entrevista	
Número de entrevista: 01	
Fecha: 14/09/2023	
Objetivo: Conocer el punto de vista de los desarrolladores con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*	
Participantes	
Rol dentro del proyecto	Desarrollador de *software* / Líder técnico
Preguntas	
Pregunta	Respuesta
¿Podría describir el proceso de desarrollo que realiza el equipo?	Se revisan las acciones de mejora o nuevas funcionalidades de negocio, luego se puntúan. Se ingresa un backlog, se crean los repositorios o branches para trabajar las reglas. Luego se desarrolla la funcionalidad, se prueban y se realiza merge de las reglas.
¿Qué actividades de aseguramiento de la calidad del *software* considera que se realizan dentro del equipo?	Pruebas del programador, pruebas en QA y *Staging*, pruebas de negocio, sesiones técnicas para revisar las funcionalidades y *code review*.
¿Existe algún manual o estándar en la organización para realizar las pruebas de *software* o saber cómo se deben realizar?	No.
¿Realizan documentación de las funcionalidades y pruebas realizadas?	Sí.
¿Quién se encarga de diseñar los diversos tipos de prueba realizados?	El desarrollador.
¿Cómo determina la forma en que se deben realizar las pruebas?	De acuerdo con los criterios de aceptación de la historia de usuario.
Según su conocimiento, ¿Qué tipos de pruebas de *software* cree que se realizan en el equipo?	Pruebas de escritorio y pruebas end to end.
¿Qué mejoras realizaría al proceso de desarrollo y aseguramiento de calidad actual?	Por el momento no identifico alguno.
Del 1 al 5, donde 1 es poca y 5 mucha, ¿Actualmente cuánta importancia se le da al proceso de QA en la organización?	3

De los siguientes tipos de pruebas de *software*: Pruebas unitarias, pruebas de integración, pruebas end-to-end, ¿Conoce el alcance de cada una de ellas?	Sí.
¿Cree que se debe incrementar el conocimiento en la organización/departamento sobre el tema de QA?	Sí
¿Alguna observación adicional?	No

Apéndice F. Entrevista 2

Entrevista	
Número de entrevista: 02	
Fecha: 14/09/2023	
Objetivo: Conocer el punto de vista de los desarrolladores con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*	
Participantes	
Rol dentro del proyecto	Desarrollador de *software*
Preguntas	
Pregunta	Respuesta
¿Podría describir el proceso de desarrollo que realiza el equipo?	Se refinan las historias durante el sprint previo, se decide qué historias se incorporarán y trabajarán durante el sprint de acuerdo a la capacidad del equipo, se da seguimiento cada día durante la daily a primera hora. Cada desarrollador es responsable de registrar su avance en la plataforma de jira. A grandes rasgos, el esqueleto de desarrollo de una historia standard consta de Análisis, Desarrollo, Pruebas Funcionales, Code review (a cargo de otro miembro del equipo), Merge, Deployment, Pruebas en QA, pruebas de aceptación, y documentación de la historia. Una vez finalizadas las historias incluidas en el release, se prepara el viaje a STG y producción

	del código. Se hace el deployment del código al entorno de STG, donde se realizan pruebas para garantizar su funcionamiento en Producción. Una vez el pase está aprobado y hay evidencia exitosa de pruebas, se agenda el pase a producción y se realiza la instalación por parte de los encargados. En la instalación se encuentra el lead técnico, el PO y de ser necesario, los desarrolladores involucrados.
¿Qué actividades de aseguramiento de la calidad del *software* considera que se realizan dentro del equipo?	Pruebas funcionales/unitarias, code review, pruebas en STG, pruebas en producción.
¿Existe algún manual o estándar en la organización para realizar las pruebas de *software* o saber cómo se deben realizar?	No.
¿Realizan documentación de las funcionalidades y pruebas realizadas?	Sí.
¿Quién se encarga de diseñar los diversos tipos de prueba realizados?	El lead técnico, y los desarrolladores.
¿Cómo determina la forma en que se deben realizar las pruebas?	El desarrollador, PO y lead técnico valoran qué escenarios de prueba se necesitan validar para la historia.
Según su conocimiento, ¿Qué tipos de pruebas de *software* cree que se realizan en el equipo?	Pruebas funcionales.
¿Qué mejoras realizaría al proceso de desarrollo y aseguramiento de calidad actual?	Necesitamos un team dedicado de QA. Tenemos un sesgo al ser quienes desarrollamos el código, de ver el "como sí" funcione.
Del 1 al 5, donde 1 es poca y 5 mucha, ¿Actualmente cuánta importancia se le da al proceso de QA en la organización?	3. Es muy común ver equipos completos sin QA's asignados.
De los siguientes tipos de pruebas de *software*: Pruebas unitarias, pruebas de integración,	No.

pruebas end-to-end, ¿Conoce el alcance de cada una de ellas?	
¿Cree que se debe incrementar el conocimiento en la organización/departamento sobre el tema de QA?	Sí.
¿Alguna observación adicional?	No.

Apéndice G. Entrevista 3

Entrevista	
Número de entrevista: 03	
Fecha: 16/09/2023	
Objetivo: Conocer el punto de vista de los desarrolladores con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*	
Participantes	
Rol dentro del proyecto	Desarrollador de *software* / Líder técnico
Preguntas	
Pregunta	Respuesta
¿Podría describir el proceso de desarrollo que realiza el equipo?	Iniciamos con el planning del sprint. Ejecutamos las tareas acordadas durante el sprint. Hacemos pruebas unitarias de los desarrollos. Realizamos code review de las tareas. El siguiente sprint pasa por pruebas de QA y STG. Finalmente, al tener el VB de las pruebas en QA y STG se realiza el pase a producción.
¿Qué actividades de aseguramiento de la calidad del *software* considera que se	El code review es parte fundamental para certificar la calidad de los desarrollos

realizan dentro del equipo?	elaborados por el equipo. Se realiza para todas las tareas trabajadas.
¿Existe algún manual o estándar en la organización para realizar las pruebas de *software* o saber cómo se deben realizar?	No que yo sepa.
¿Realizan documentación de las funcionalidades y pruebas realizadas?	Sí.
¿Quién se encarga de diseñar los diversos tipos de prueba realizados?	Cada desarrollador tiene la obligación de probar las diferentes funcionalidades o casuísticas desarrolladas.
¿Cómo determina la forma en que se deben realizar las pruebas?	Probando todas las posibles acciones que puede realizar el usuario.
Según su conocimiento, ¿Qué tipos de pruebas de *software* cree que se realizan en el equipo?	Pruebas unitarias, pruebas de verificación de Logs y pruebas de integración.
¿Qué mejoras realizaría al proceso de desarrollo y aseguramiento de calidad actual?	Darle más enfoque de pruebas en el aplicativo de otros países cuando se actualicen reglas de FW, ya que estas son compartidas en otros países.
Del 1 al 5, donde 1 es poca y 5 mucha, ¿Actualmente cuánta importancia se le da al proceso de QA en la organización?	4
De los siguientes tipos de pruebas de *software*: Pruebas unitarias, pruebas de integración, pruebas end-to-end, ¿Conoce el alcance de cada una de ellas?	No.
¿Cree que se debe incrementar el conocimiento en la organización/departamento sobre el tema de QA?	Sí.
¿Alguna observación adicional?	No.

Apéndice H. Entrevista 4

Entrevista	
Número de entrevista: 04	
Fecha: 16/09/2023	
Objetivo: Conocer el punto de vista de los desarrolladores con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*	
Participantes	
Rol dentro del proyecto	Desarrollador de *software*
Preguntas	
Pregunta	Respuesta
¿Podría describir el proceso de desarrollo que realiza el equipo?	Se realizan las siguientes actividades: -Análisis y diseño -Desarrollo e implementación -Pruebas unitarias -Code review -Despliegues -Pruebas certificación QA -Release
¿Qué actividades de aseguramiento de la calidad del *software* considera que se realizan dentro del equipo?	Pruebas de *software*, entendimiento profundo de los requerimientos de los stakeholders.
¿Existe algún manual o estándar en la organización para realizar las pruebas de *software* o saber cómo se deben realizar?	No.
¿Realizan documentación de las funcionalidades y pruebas realizadas?	Sí.

¿Quién se encarga de diseñar los diversos tipos de prueba realizados?	Cada desarrollador tiene la obligación de probar las diferentes funcionalidades o casuísticas desarrolladas.
¿Cómo determina la forma en que se deben realizar las pruebas?	Según los escenarios funcionales y los escenarios relacionados con el manejo de errores.
Según su conocimiento, ¿Qué tipos de pruebas de *software* cree que se realizan en el equipo?	Pruebas manuales.
¿Qué mejoras realizaría al proceso de desarrollo y aseguramiento de calidad actual?	Implementación de pruebas automáticas.
Del 1 al 5, donde 1 es poca y 5 mucha, ¿Actualmente cuánta importancia se le da al proceso de QA en la organización?	2
De los siguientes tipos de pruebas de *software*: Pruebas unitarias, pruebas de integración, pruebas end-to-end, ¿Conoce el alcance de cada una de ellas?	No.
¿Cree que se debe incrementar el conocimiento en la organización/departamento sobre el tema de QA?	Sí.
¿Alguna observación adicional?	No.

Apéndice I. Entrevista 5

Entrevista	
Número de entrevista: 05	
Fecha: 16/09/2023	
Objetivo: Conocer el punto de vista de los desarrolladores con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*	
Participantes	
Rol dentro del proyecto	Desarrollador de *software* / Líder técnico

Preguntas	
Pregunta	Respuesta
¿Podría describir el proceso de desarrollo que realiza el equipo?	Se hace el planning del sprint en donde a cada desarrollador se le asigna una historia de usuario, cuando el desarrollador finaliza la funcionalidad realiza el code review con el líder técnico y seguidamente realiza pruebas del código, después procede a hacer el despliegue de la funcionalidad a los ambientes de QA, staging y por último al ambiente de producción.
¿Qué actividades de aseguramiento de la calidad del *software* considera que se realizan dentro del equipo?	Pruebas unitarias y code review.
¿Existe algún manual o estándar en la organización para realizar las pruebas de *software* o saber cómo se deben realizar?	No.
¿Realizan documentación de las funcionalidades y pruebas realizadas?	Sí.
¿Quién se encarga de diseñar los diversos tipos de prueba realizados?	Cada desarrollador las diseña según su criterio.
¿Cómo determina la forma en que se deben realizar las pruebas?	Se revisan los criterios de aceptación de la historia de usuario y se hacen las pruebas según se necesite.
Según su conocimiento, ¿Qué tipos de pruebas de *software* cree que se realizan en el equipo?	Pruebas unitarias principalmente..
¿Qué mejoras realizaría al proceso de desarrollo y aseguramiento de calidad actual?	Capacitaciones a los equipos para mejorar el nivel de profundidad de las pruebas que se realizan.
Del 1 al 5, donde 1 es poca y 5 mucha, ¿Actualmente cuánta importancia se le da al proceso de QA en la organización?	2
De los siguientes tipos de pruebas de *software*: Pruebas unitarias, pruebas de integración,	No, solamente de las pruebas unitarias.

pruebas end-to-end, ¿Conoce el alcance de cada una de ellas?	
¿Cree que se debe incrementar el conocimiento en la organización/departamento sobre el tema de QA?	Sí, en general en el equipo carece de conocimiento sobre temas de QA.
¿Alguna observación adicional?	No.

Apéndice J. Entrevista 6

Entrevista	
Número de entrevista: 06	
Fecha: 14/09/2023	
Objetivo: Conocer el punto de vista de los desarrollador del supervisor de sistemas Pega con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*	
Participantes	
Rol dentro del proyecto	Supervisor de sistemas Pega
Preguntas	
Pregunta	Respuesta
¿Cómo cree que se realiza el proceso de desarrollo y QA de *software* actualmente en los equipos? (En cada sprint, desde el planning hasta cuando se llevan las funcionalidades a QA).	Utilizamos metodología SCRUM de 2 semanas, los PO priorizan las historias de su backlog y lo comparte con el equipo, seguido a ello el equipo se compromete en la cantidad de las historias y se iniciar con el desarrollo de las historias, finalizado el desarrollo de cada historia es desplegado hacia el ambiente de QA, en ese momento el PO es quién acepta las historia en dicho ambiente. El ciclo se atención para desarrollar las historias se vuelve iterativo e incremental hasta finalizar con el conjunto de historias. Si el conjunto de historias resuelve el release se procede con el despliegue hacia ambiente de Staging, por lo que el equipo certifica las pruebas en dicho ambiente, si las pruebas son satisfactorias se lleva el release a

	revisión del administrador de cambios para coordinar la instalación del pase en Producción.
¿Existe un proceso de QA estandarizado en el departamento?, (es decir, cada equipo realiza las mismas actividades de la misma forma, por ejemplo, tipos de pruebas realizadas, escenarios cubiertos, documentación, etc.)	No existe un proceso de QA estandarizado, estamos evaluando la posibilidad de crear un área de QA y con ello la metodología de trabajo a utilizar, de momento nos enfocamos únicamente en la creación de pruebas unitarias en el desarrollo que se implementa en cada una de las diferentes historias. No hacemos automatización de pruebas, pero también está en el *roadmap* llegar a ese punto.
¿Cuáles son los principales roles que forman parte del ciclo de vida de desarrollo y aseguramiento de la calidad?	Los roles que forman parte del ciclo de vida de desarrollo son: Product owner, Scrum Master, Líder Técnico, Desarrolladores; adicionalmente existen las figuras de amigos del equipo, quiénes pueden ser el SME, el Líder y Arquitecto de *Software*, el Arquitecto de Negocio, Desarrollador de plataformas externas. No tenemos identificados los roles de aseguramiento de calidad actualmente, pero al construir el área probablemente se estarían definiendo nuevos roles.
¿Cuáles actividades relacionadas con aseguramiento de la calidad cree que realizan los equipos?	Se realizan pruebas unitarias por parte de los desarrolladores en ambiente Dev, pruebas de aceptación por parte de los PO en ambiente de QA y las pruebas de certificación del release end to end en ambiente de Staging.
¿Cuáles cree que son los principales puntos de dolor que enfrenta la organización/departamento con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*?	La demanda de la gran cantidad de requerimientos desarrollados y crecimiento acelerado de las aplicaciones no va alineado a la calidad de las pruebas que deben realizarse para que no exista afectación cuando los releases son instalados en ambiente de Producción, por tanto, el Negocio suele verse afectado con incidentes provocados por la calidad en los desarrollos.

¿Existe un departamento que se especialice en la gestión de la calidad del *software*, en este caso en Pega?	No existe, pero hemos estamos trabajando en disponer un área de QA que permita la gestión de la calidad de los desarrollos que actualmente se desarrollan por los equipos.
¿Cómo se podrían "alivianar" esos puntos de dolor con respecto al tema de QA del *software*?	Creando el área de QA que permita gestionar la calidad del *software* y considerando incluir todos los diferentes tipos de pruebas que mejoren la calidad de los desarrollos, así de una metodología basado en un proceso que contemple todos los escenarios de pruebas y aprobaciones; finalmente, lograr la automatización en el desarrollo de pruebas automáticas para agilizar los desarrollos sin afectar los tiempos del Negocio.

Apéndice K. Entrevista 7

Entrevista	
Número de entrevista: 07	
Fecha: 17/09/2023	
Objetivo: Conocer el punto de vista del supervisor de producto de *software* con respecto al proceso de desarrollo y aseguramiento de la calidad del *software*	
Participantes	
Rol dentro del proyecto	Supervisor de producto de *software*
Preguntas	
Pregunta	Respuesta
¿Cómo cree que se realiza el proceso de desarrollo y QA de *software* actualmente en los equipos? (En cada sprint, desde el planning hasta cuando se llevan las funcionalidades a QA). Descríbalo si es posible.	El proceso de desarrollo está bastante estandarizado excepto por las actividades de aseguramiento de la calidad, no está bien planificado y prácticamente todas las pruebas son manuales

¿Existe un proceso de QA estandarizado en el departamento?, (es decir, cada equipo realiza las mismas actividades de la misma forma, por ejemplo, tipos de pruebas realizadas, escenarios cubiertos, documentación, etc).	No existe un proceso estandarizado.
¿Cuáles son los principales roles que forman parte del ciclo de vida de desarrollo y aseguramiento de la calidad del *software*?	Desarrollador: Persona que construye la funcionalidad en la aplicación según los criterios de aceptación de las historias. También se encarga de crear la prueba unitaria por cada historia/objeto. Product owner: Encargado de definir y priorizar las historias de usuario, se encarga de las pruebas de aceptación, las cuales se ejecutan de manera manual.
¿Cuáles actividades relacionadas con aseguramiento de la calidad cree que realizan los equipos?	Actualmente los equipos realizan pruebas de sistema y de aceptación principalmente. Dichas pruebas se realizan manualmente.
¿Existe un departamento que se especialice en la gestión de la calidad del *software*, en este caso en Pega?	No existe. Se está proponiendo la creación de un equipo de Calidad del *Software* para la plataforma Pega.

Apéndice L Primera observación del proceso actual

Observación del proceso	
Fecha	20/09/2023
Objetivo	Describir con mayor profundidad la forma en que se realiza el proceso de desarrollo y aseguramiento de calidad del *software* para identificar sus principales actividades.
Proceso observado	Desarrollo y aseguramiento de calidad del *software* dentro del departamento de Sistemas Banca y Procesos.
Principales hallazgos	

Se determina de manera general que las principales actividades del proceso actual son: • Recopilación de requerimientos de negocio. • Creación del backlog • Refinamiento de las historias • Planeación del sprint • Desarrollo de historias de usuario • Pruebas unitarias • Code review • Despliegue de las funcionalidades al ambiente de QA • Pruebas en QA • Pruebas de aceptación • Documentación de las historias • Despliegue de las funcionalidades al ambiente de Staging • Pruebas en el ambiente de Staging • Despliegue de las funcionalidades al ambiente de producción
Comentarios
El proceso observado se realiza dentro del equipo del desarrollador del proyecto, este se realiza tomando en cuenta la participación del desarrollador en el ciclo del *sprint.*

Apéndice M. Segunda observación del proceso actual

Observación del proceso	
Fecha	21/09/2023
Objetivo	Describir con mayor profundidad la forma en que se realiza el proceso de desarrollo y aseguramiento de calidad del *software* para identificar roles y otros aspectos que sean relevantes que aporten valor para describir el proceso actual.
Proceso observado	Desarrollo y aseguramiento de calidad del *software* dentro del departamento de Sistemas Banca y Procesos.
Principales hallazgos	
Se determina de manera general que los principales roles que intervienen dentro del proceso en el equipo son: • Desarrollador de *software* • Líder técnico • *Product Owner* • *Scrum Master* Además se determina que dentro del proceso se utilizan las siguientes herramientas:	

• Jira para describir las historias de usuario. • Confluence para adjuntar la documentación de las historias desarrolladas dentro del sprint. • Pega como herramienta para el desarrollo de *software*.
Comentarios
El proceso observado se realiza dentro del equipo del desarrollador del proyecto, se toman en cuenta los roles generales que intervienen en el proceso, así como las herramientas que se supone todos los otros equipos de desarrollo deberían utilizar.

Apéndice N. Resultado de revisión documental #1

Revisión documental	
Objetivo	Revisar el contenido de dos metodologías existentes para la mejora de procesos.
Fuente	Libros: • Process Mapping, Process Improvement and Process Management: A Practical Guide to Enhancing Work Flow and Information Flow.
Principales hallazgos	

Se determina que de manera general las fases y pasos que forman parte de la metodología de rediseño de procesos propuesta por Madison (2005) son:
• Fase 1:
  ○ Introducción al proceso de rediseño.
  ○ Formación del equipo de procesos.
• Fase 2:
  ○ Diagrama “As-Is”.
  ○ Entrevista con el cliente.
  ○ Benchmarking y buenas prácticas.
• Fase 3:
  ○ Primer corte del rediseño.
  ○ Revisión de la gerencia y pruebas.
  ○ Diseño final y comunicación a personal y clientes.
  ○ Implementación del rediseño.
• Fase 4:
  ○ Instalación de métricas y mejora continua.
  ○	

Además se identificaron los siguientes lentes para el diagnóstico de procesos y la forma en que estos influyen en el rediseño del proceso: • Lente de frustración. • Lente de tiempo. • Lente de calidad.

Apéndice O. Resultado de revisión documental #2

Revisión documental	
Objetivo	Revisar buenas prácticas o metodologías para la elaboración del proceso *to-be* de aseguramiento de calidad del *software*.
Fuente	Libros: • Guide to the *Software* Engineering Body of Knowledge (SWEBOK(R)): Version 3.0. • *Software* Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement. • *Software* testing foundations. Sitios web: • https://www.guru99.com/*software*-testing-life-cycle.html
Principales hallazgos	
Se revisan un total de cuatro metodologías para el rediseño del proceso de aseguramiento de la calidad del *software*, de manera general los autores indican que el proceso está compuesto por las siguientes fases: • Fase de análisis de requerimientos. • Fase de planeación de las pruebas. • Fase de diseño de las pruebas. • Fase de ejecución de las pruebas. Las fases mencionadas anteriormente se encuentran compuestas por actividades diferentes según lo descrito por cada autor.	

Apéndice P. Resultado de revisión documental #3

Revisión documental	
Objetivo	Revisar herramientas de gestión de casos de prueba disponibles y sus características.
Fuente	Sitios web: • https://www.aiotests.com/ • https://agiletest.app/ • https://www.testrail.com/platform/ • https://tuskr.app/ • https://www.kualitee.com/
Principales hallazgos	
Se revisaron los sitios web las herramientas preseleccionadas para obtener las características con las que cuenta cada herramienta. Para cada herramienta se revisó lo siguiente: • Posibilidad de creación de casos de prueba. • Capacidad para crear conjuntos de pruebas. • Capacidad para integrarse con Jira. • Costos de los planes disponibles para cada herramienta. • Opciones de escalabilidad con respecto a los planes ofrecidos y el número de usuarios que permite. • Métodos de autenticación de la herramienta. • Otras funcionalidades que podrían agregar valor.	

Apéndice Q. Resultado de revisión documental #4

Revisión documental	
Objetivo	Revisar material disponible para la realización del plan de implementación del proceso propuesto y para el plan de aseguramiento de calidad del *software*.
Fuente	Libros: • IEEE 730-2014 - IEEE Standard for *Software* Quality Assurance Processes. • Guía de los fundamentos para la dirección de proyectos (PMBOK).
Principales hallazgos	

- Con respecto al plan de implementación del proceso propuesto, el PMBOK en su versión siete brinda ocho dominios de desempeño para la gestión de proyectos, los cuales son:
  - Involucrados.
  - Equipo.
  - Enfoque de desarrollo y del ciclo de vida.
  - Planificación.
  - Trabajo del proyecto.
  - Entrega.
  - Medición.
  - Incertidumbre.

  De estos dominios se plantea la idea de trabajar solamente con los dominios de involucrados, planificación e incertidumbre.

- Para el plan de aseguramiento de calidad del *software* se revisa el estándar IEEE 730-2014 para revisar el contenido general de un plan de este tipo, además se revisaron ejemplos de planes disponibles realizados por algunos actores para identificar las secciones que son factibles de implementar para el plan propuesto.

Apéndice R. Minuta 1

Minuta de Reunión	
Información General	
Número de Minuta:	01
Fecha:	28/09/2023
Participantes:	• Hermes Quesada • Fabián Navarro
Objetivos	
• Revisar el contenido del capítulo 4 relacionado con el análisis de la situación actual para verificar las actividades descritas en el proceso actual.	

• Obtener feedback sobre los principales "dolores" del proceso actual
Temas tratados
• Se hizo una revisión de alto nivel de las actividades descritas en el proceso actual y se quitaron algunas actividades que según la perspectiva de Hermes no realizaban todos los equipos de desarrollo. • Hermes describió los principales dolores que tiene el departamento con respecto al proceso de desarrollo en Pega, además brindó recomendaciones sobre cómo mejorar el proceso actual. Se identificaron los principales defectos que tiene el proceso actual y cómo estos se materializan y tienen efectos negativos en el ambiente de producción y la operativa de la organización
Observaciones/Comentarios
• Se sugiere por parte de Hermes acudir a Gil Alberto Hernández para revisar las actividades detalladas y obtener su feedback. • De igual forma se recomendó solicitar una sesión con Esteban Martínez para verificar cuáles actividades serían factibles para agregar al proceso de aseguramiento de la calidad del *software*, ya que él tiene cierto nivel de conocimiento con respecto a pruebas de *software* gracias a la experiencia obtenida en experiencias laborales en auditoría de *software.*

Apéndice S. Minuta 2

Minuta de Reunión	
Información General	
Número de Minuta:	02
Fecha:	02/10/2023
Participantes:	• Gil Alberto Hernández • Fabián Navarro
Objetivos	

• Revisar las actividades que establecimos como parte del proceso actual. • Entender la perspectiva de Gil Alberto sobre la problemática con respecto al proceso de desarrollo y aseguramiento de la calidad. • Investigar las acciones que visualiza el COE para mitigar la problemática con respecto a los incidentes que se presentan en el ambiente de producción.
Temas tratados
• Se revisó con Gil Alberto las actividades que previamente habían sido definidas con Hermes sobre el proceso de desarrollo y aseguramiento de la calidad. • Entender la perspectiva de Gil Alberto sobre la problemática de falta del aseguramiento de la calidad en el proceso de desarrollo en Pega y analizar el seguimiento que realiza sobre estadísticas con respecto a los incidentes que se presentan en el ambiente de producción y las principales causas de estos incidentes y cuál es la visión que maneja el Centro de Excelencia Operacional para mitigar estas causas.
Observaciones/Comentarios
• Se determinó que la estandarización del proceso e implementar diferentes tipos de pruebas de *software* como obligatorias para los equipos es una opción viable para mitigar la problemática que se presenta con respecto a los incidentes ocurridos en producción. • Se analizaron las acciones que ha forzado el COE como respuesta a la creciente aparición de incidentes en producción por pases de Pega.

Apéndice T. Minuta 3

Minuta de Reunión	
Información General	
Número de Minuta:	03

Fecha:	06/10/2023
Participantes:	• Hermes Quesada • Fabián Navarro

Objetivos

- Revisar las actividades del proceso *to-be* planteados.
- Evaluar la opción de tomar en cuenta una herramienta de gestión de pruebas como complemento para el proceso.
- Discutir la forma en que se pueden obtener los ingresos para el proyecto.

Temas tratados

- Se le explicó a Hermes el avance actual del proyecto y lo realizado en el capítulo de propuesta de solución del proyecto, se explicaron los estándares y buenas prácticas que fueron utilizados para elaborar las actividades del nuevo proceso propuesto.
- Se le preguntó a Hermes su opinión con respecto a tomar en cuenta una herramienta de gestión de pruebas de *software* como parte de las opciones de mejora que se le pueden incluir al proceso propuesto. También se aprovecha la sesión para recopilar los principales requerimientos con que debería evaluarse la herramienta que se tomaría en cuenta para el proceso propuesto.
- Con respecto a al tema de cálculo de ingresos del proyecto se evalúa con Hermes la forma en que se pueden calcular dichos ingresos y se determina que una forma viable podría ser tomando en cuenta los RFC´s relacionados a incidentes en producción producto de la falta de escenarios de pruebas por parte de los equipos, para ello se recomienda obtener un promedio de RFC´s semanales y el esfuerzo en tiempo aproximado para resolver los incidentes, con ello se obtendrá los posibles ingresos que se obtendrían al implementar el proyecto.

Observaciones/Comentarios

- Con respecto al análisis de requerimientos de la herramienta se acuerda con Hermes que una de las principales condiciones con las que debe cumplir es que se pueda integrar con Jira, además de evaluar la opción de buscar herramientas gratuitas si fuerza posible.
- También se acuerda realizar una sesión con Gil Alberto Hernández para conversar sobre la idea de conformar en equipo dedicado al tema de aseguramiento de la calidad del *software*.

Apéndice U. Minuta 4

Minuta de Reunión	
Información General	
Número de Minuta:	04
Fecha:	09/10/2023
Participantes:	• Fabián Navarro • Gil Alberto Hernández
Objetivos	
• Realizar una revisión sobre las actividades del proceso *to-be* propuesto. • Comprender la idea de formar un departamento de aseguramiento de la calidad del *software*.	
Temas tratados	

• Se realizó una revisión general del proceso propuesto y sus actividades para saber si es factible implementar todas las actividades para el proceso. • Se le pregunta a Gil Alberto la opción que tienen contemplada de crear un departamento de aseguramiento de calidad. Para ello se preguntan los detalles sobre la creación de este departamento, específicamente temas relacionados con número de personas de equipo y presupuesto disponible para la contratación de este.
Observaciones/Comentarios
• Con respecto a las actividades propuestas para el nuevo proceso, Gil Hernández sugiere omitir la fase de análisis de requerimientos, ya que esta es realizada por el *product owner* de cada equipo y en la cual los desarrolladores no tienen participación directa. • Se establece por porte de Gil Alberto que no será un departamento, sino que se contempla formar en equipo dedicado de alrededor de cuatro personas del mismo departamento para dedicarse a este tema inicialmente con una jornada de medio tiempo.

Apéndice V. Minuta 5

Minuta de Reunión	
Información General	
Número de Minuta:	05
Fecha:	20/09/2023 – 10/10/2023
Participantes:	• Fabián Navarro • Agustín Francesa
Objetivos	
• Obtener retroalimentación sobre el contenido desarrollado en los capítulos 2,4 y 5.	

Temas tratados
• El profesor Agustín revisa el contenido que se ha desarrollado en el capítulos 2 y recomienda agregar más contenido con respecto a los temas de aseguramiento de calidad del *software*. • Con respecto al capítulo cuatro se le realizan algunas preguntas al profesor con respecto a las técnicas utilizadas para describir el proceso actual, específicamente referida a la observación. El profesor recomienda complementar dicha técnica con otras técnicas para describir el proceso actual para no describirlo de manera muy subjetiva. • Se realizaron preguntas sobre el tema de cálculo de costos y beneficios económicos del proyecto y sobre la inclusión de una nueva fase para la evaluación de la implementación de una herramienta de gestión de pruebas de *software*.
Observaciones/Comentarios
• El profesor Agustín anota las observaciones y correcciones necesarias en el documento colaborativo creado. • Esta comunicación se realiza vía WhatsApp por temas de facilidad de comunicación. • Se obtuvo la retroalimentación del profesor lector de igual forma a través de comentarios en el documento compartido. • Se decide incluir una nueva fase para realizar el análisis de implementación de una herramienta para la gestión de pruebas de *software* para el proceso propuesto. • Se decide organizar una sesión para la revisión del documento.

Apéndice W. Plantilla para documentación de casos de prueba

Grupo Financiero
Documentación de casos de prueba

Datos del caso de prueba

Historia de usuario relacionada		**ID del caso de prueba**		**Estado del caso de prueba**	
Prioridad (Baja/Media/Alta)		**Fecha de ejecución**			
Descripción del caso de prueba					

Precondiciones

Datos de prueba

# Paso	Detalle del paso	Resultado esperado	Resultado actual	Estado

Comentarios

Apéndice X. Aceptación de minutas por parte de la contraparte de la empresa

Aprobación de minutas contraparte de la organización

Yo Hermes Quesada Rodríguez confirmo la veracidad de las siguientes minutas correspondientes al trabajo final de graduación del estudiante Fabián Navarro Martínez, el cual lleva como nombre "Propuesta de mejora del proceso de aseguramiento de la calidad del software para un Grupo Financiero".

Minuta	Fecha
Minuta 01	27/04/2023
Minuta 02	24/05/2023
Minuta 04	08/08/2023
Minuta 07	26/09/2023
Minuta 08	28/09/2023
Minuta 09	02/10/2023
Minuta 10	06/10/2023
Minuta 11	09/10/2023
Minuta 14	03/11/2023

Firma: _____

Apéndice Y. Aceptación de minutas por parte del profesor tutor

Aprobación de minutas profesor tutor

Yo Agustín Francesa Alfaro confirmo la veracidad de las siguientes minutas correspondientes al trabajo final de graduación del estudiante Fabián Navarro Martínez, el cual lleva como nombre "Propuesta de mejora del proceso de aseguramiento de la calidad del software para un Grupo Financiero".

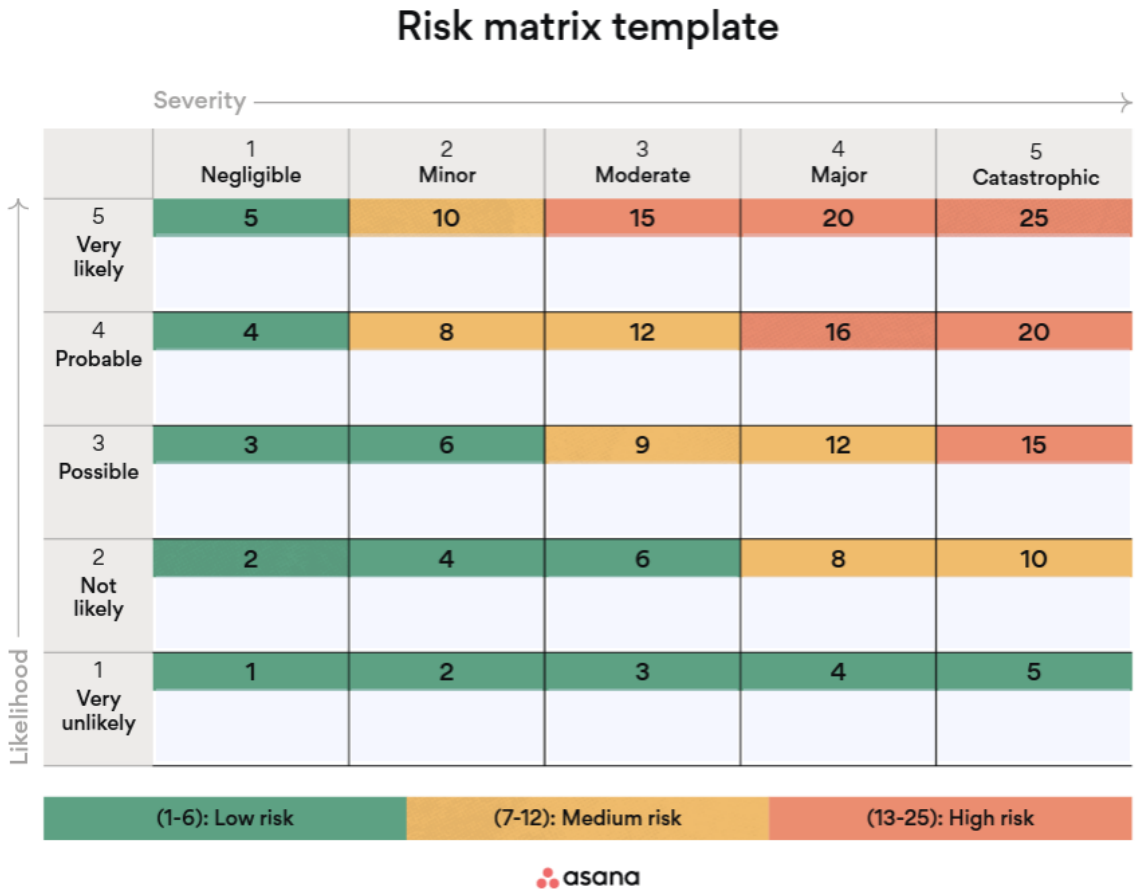
Minuta	Fecha
Minuta 04	08/08/2023
Minuta 07	26/09/2023
Minuta 14	03/11/2023

Firma: 

JOSE AGUSTIN FRANCESA ALFARO (FIRMA)
2023.11.03 17:38:22-06'00'

10. Capítulo 10: Anexos

Anexo I .Plantilla para matriz de riesgos



Anexo II. Carta de revisión filológica

Alajuela, 05 de noviembre de 2023

A quien interese:

Yo, Gisela Alfaro Chaves, cédula de identidad 2-0701-0506 profesional en Filología Española y en Enseñanza del Castellano y la Literatura, perteneciente al Colegio de Licenciados y Profesores en Letras, Filosofía, Ciencias y Artes; leí y corregí el proyecto final de graduación:

Propuesta de mejora del proceso de aseguramiento de la calidad del software para un Grupo Financiero

Documento realizado por el estudiante Fabián Navarro Martínez, cédula: 305060245; con el fin de optar por el grado académico de Licenciatura en Administración de Tecnología de Información, del Instituto Tecnológico de Costa Rica.

Por este motivo, se revisaron y corrigieron aspectos como la construcción de párrafos, organización discursiva, vicios del lenguaje trasladados al campo escrito, ortografía, puntuación, barbarismos, coherencia, cohesión y otros elementos relacionados con el campo filológico.

Realizadas las correcciones, doy fe de que el documento está listo para ser presentado.

Se suscribe de ustedes cordialmente,

Gisela Alfaro Chaves, céd 207010506
Carné de colegiada 67138