Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

DISEÑO DE INVESTIGACIÓN PARA PROPUESTA DE UN SISTEMA DE GESTIÓN DE MEDICIÓN PARA EL CONTROL Y ACTUALIZACIÓN EN UNA EMPRESA DE DESARROLLO DE SOFTWARE

Noé David Gómez Xicol
Asesorado por el M.A. Ing.Brian Enrique Chicol Morales

Guatemala, junio de 2022

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

DISEÑO DE INVESTIGACIÓN PARA PROPUESTA DE UN SISTEMA DE GESTIÓN DE MEDICIÓN PARA EL CONTROL Y ACTUALIZACIÓN EN UNA EMPRESA DE DESARROLLO DE SOFTWARE

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

NOÉ DAVID GÓMEZ XICOL
ASESORADO POR ELM.A. ING.BRIAN ENRIQUE CHICOL MORALES

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, JUNIO DE 2022

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANA	Inga. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Kevin Vladimir Cruz Lorente
VOCAL V	Br. Fernando José Paz González
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Pedro Antonio Aguilar Polanco
EXAMINADOR	Ing. César Augusto Fernández Cáceres
EXAMINADOR	Ing. Marlon Francisco Orellana López
EXAMINADOR	Ing. Pedro Pablo Hernández Ramírez
SECRETARIA	Inga. Lesbia Magalí Herrera López

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

DISEÑO DE INVESTIGACIÓNPARAPROPUESTA DE UN SISTEMA DE GESTIÓN DE MEDICIÓN PARA EL CONTROL Y ACTUALIZACIÓN EN UNA EMPRESA DE DESARROLLO DE SOFTWARE

Tema que me fuera asignado por la Dirección de Escuela de Estudios de Postgrado con fecha abril de 2022.

Noé David Gómez Xicol

ESCUELA DE ESTUDIOS DE
POSTGRADO
FACULTAD DE INGENIERÍA

EEPFI-PP-0565-2022
Guatemala, 26 de abril de 2022

Director
Carlos Gustavo Alonzo
Escuela De Ingenieria En Sistemas
Presente.

Estimado Ing. Alonzo

Reciba un cordial saludo de la Escuela de Estudios de Postgrado de la Facultad de Ingeniería.

El propósito de la presente es para informarle que se ha revisado y aprobado el Diseño de Investigación titulado: **DISEÑO DE INVESTIGACIÓN PARA PROPUESTA DE UN SISTEMA DE GESTION DE MEDICION PARA EL CONTROL Y ACTUALIZACION EN UNA EMPRESA DE DESARROLLO DE SOFTWARE**, el cual se enmarca en la línea de investigación: **Gerencia Estratégica - Cuadro de mando integral**, presentado por el estudiante **Noe David Gomez Xicol** carné número **200924998**, quien optó por la modalidad del "PROCESO DE GRADUACIÓN DE LOS ESTUDIANTES DE LA FACULTAD DE INGENIERÍA OPCIÓN ESTUDIOS DE POSTGRADO". Previo a culminar sus estudios en la Maestría en ARTES en Gestion Industrial.

Y habiendo cumplido y aprobado con los requisitos establecidos en el normativo de este Proceso de Graduación en el Punto 6.2, aprobado por la Junta Directiva de la Facultad de Ingeniería en el Punto Décimo, Inciso 10.2 del Acta 28-2011 de fecha 19 de septiembre de 2011, firmo y sello la presente para el trámite correspondiente de graduación de Pregrado.

Atentamente,

"Id y Enseñad a Todos"

Brian Enrique Chicol Morales
INGENIERO ELÉCTRICO Col. 16356
Ma. INGENIERÍA DE MANTENIMIENTO

Mtro. Brian Enrique Chicol Morales
Asesor(a)

UNIVERSIDAD DE SAN CARLOS
ESCUELA DE POSTGRADO
FACULTAD DE INGENIERIA
DE GUATEMALA

Mtro. Hugo Humberto Rivera Perez
Coordinador(a) de Maestría

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
DIRECCIÓN
ESCUELA DE POSTGRADO

Mtro. Edgar Darío Alvaréz Cotí
Director
Escuela de Estudios de Postgrado
Facultad de Ingeniería

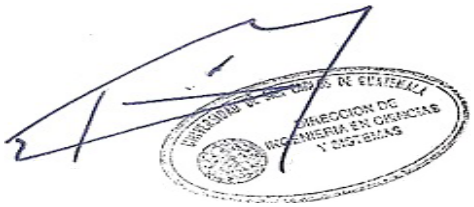
UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERÍA

EEP-EICS-0565-2022

El Director de la Escuela De Ingenieria En Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del Asesor, el visto bueno del Coordinador y Director de la Escuela de Estudios de Postgrado, del Diseño de Investigación en la modalidad Estudios de Pregrado y Postgrado titulado: **DISEÑO DE INVESTIGACIÓN PARA PROPUESTA DE UN SISTEMA DE GESTION DE MEDICION PARA EL CONTROL Y ACTUALIZACION EN UNA EMPRESA DE DESARROLLO DE SOFTWARE**, presentado por el estudiante universitario **Noe David Gomez Xicol**, procedo con el Aval del mismo, ya que cumple con los requisitos normados por la Facultad de Ingeniería en esta modalidad.

ID Y ENSEÑAD A TODOS



Ing. Carlos Gustavo Alonzo
Director
Escuela De Ingenieria En Sistemas

Guatemala, abril de 2022

Decanato
Facultad de Ingeniería
24189101- 24189102
secretariadecanato@ingenieria.usac.edu.gt

LNG.DECANATO.OI.475.2022

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería en Ciencias y Sistemas, al Trabajo de Graduación titulado: **DISEÑO DE INVESTIGACIÓN PARA PROPUESTA DE UN SISTEMA DE GESTIÓN DE MEDICIÓN PARA EL CONTROL Y ACTUALIZACIÓN EN UNA EMPRESA DE DESARROLLO DE SOFTWARE**, presentado por: **Noé David Gómez Xicol,** después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANA
FACULTAD DE INGENIERÍA

Inga. Aurelia Anabela Cordova Estrada
Decana

Guatemala, junio de 2022

AACE/gaoc

ACTO QUE DEDICO A:

Dios	Por haberme permitido realizar una más de mis metas y ese pilar en mi vida.
Mis padres	Por su apoyo incondicional, amor, sus consejos, sus regaños, sus cuidados y por tanto.
Mis hermanos	Josué, Lesly y Ahmid Gómez por ese apoyo, esa fuerza y esas palabras que a veces sin decirlas las entendía.
Mis abuelos	Cristina Acabal de Gómez (q. e. p. d.), Graciela Gómez, por sus consejos, palabras motivadoras y oraciones.
Esposa	Por su amor, comprensión, motivación y apoyo incondicional.
Familia y amigos	Brenda y Yovani Yoque, Hugo Ahmid Gómez, Dorca, Paula y Hugo David Gómez, por su apoyo, consejos y sus oraciones.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por ser mi alma mater y formarme como profesional.
Facultad de Ingeniería	Por todo el aprendizaje obtenido dentro de sus aulas, no solo en el área profesional sino también en lo personal.
Mis padres	Por su apoyo incondicional, porque siempre han querido ser ese trampolín para que yo logre mis metas.
Mis hermanos	Josué, Lesly, Fernando Gomez por acompañarme en este camino largo y motivarme siempre.
Mis amigos y compañeros	Por darme ese empuje que necesitaba para llegar cumplir con esta meta.
A mi asesor	Por el acompañamiento y el conocimiento compartido para la realización de este trabajo de investigación.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
=	Igual que
%	Porcentaje
Q	Quetzales

GLOSARIO

BSC	*Balance scorecard.*
Confiabilidad	Habilidad que tiene un sistema o componente de realizar sus funciones requeridas bajo condiciones específicas en periodos de tiempo determinados.
Correctitud	Propiedad que distingue a un algoritmo de un procedimiento efectivo.
DevOps	*Developer and Operations*.
Dod	Departamento de defensa de EUA.
Eficiencia	Capacidad del producto de software para adherirse a estándares o convenciones definidos para que el software cumpla con los lineamientos establecidos.
Flexibilidad	Medida el programa es susceptible de ser cambiado.
Interoperabilidad	Capacidad que tienen los sistemas y/o equipos no solo de intercambiar información si no de interpretarla y procesarla en un formato amigable a su usuario.
IV&V	Verificación y validación independientes.

KPIs	Indicadores de desempeño.
Mantenibilidad	Es la facilidad con la que se modifica, mejora y/o adapta un producto software.
Portabilidad	Característica que posee un software para ejecutarse en diferentes plataformas.
Reusabilidad	Poder volver a usar parte de dicho software en otro proyecto.
SMART	Acrónimo formado por las siglas en ingles de requerimientos para definir un kpi.
SQA	Aseguramiento de la Calidad del Software.
Testeabilidad	Es el grado en el cual un artefacto de software (sistema, modulo, clase, entre otros.) soporta *testing* en un contexto de prueba dado.
Usabilidad	Atributo de calidad que mide lo fáciles que son de usar las interfaces.

RESUMEN

En el área de desarrollo de software los objetivos a cumplir en principio podemos listar la calidad de software, manejo de versiones del software y los indicadores que apoyan a medir la productividad en equipos de trabajo.

Para la realización una gestión adecuada en el área de desarrollo de software es necesario definir un modelo de gestión de calidad. Se cuenta con antecedentes y un marco teórico que dan un panorama amplio para poder definirlo, en cuanto a la calidad del software existen métricas que permiten definir adoptar un modelo adecuado y a su vez definir la estrategia adecua para el aseguramiento de la calidad.

El diseño tiene por objetivo proponer un sistema de medición para el control y actualización en una empresa de Desarrollo de software, definir indicadores se utilizará la herramienta BSC y para la gestión de calidad y control de versiones de software recomendar una herramienta de *software* para este propósito.

El diseño de investigación se realiza a través de un enfoque mixto debido a que las variables son medidas de forma cualitativa y cuantitativa, es un diseño no experimental debido a que se cuentan con datos históricos, se realiza un estudio transversal con los datos proporcionados con lo que se busca evidenciar las causas que dan pie a la problemática.

1. INTRODUCCIÓN

La medición de productividad en el desarrollo de software es un tema de debate al ser el producto final intangible, esto provoca que existan diferentes formas de medición de las cuales cada empresa define sus métricas conforme a análisis previos.

Cuando no se tienen métricas acordes a las necesidades y objetivos de la empresa puede que no se obtenga un resultado objetivo y eso genere perdidas a la empresa.

Uno de los factores importantes en el desarrollo de software es el control de calidad en el productivo final. Al hablar de calidad de software la mayoría de las veces se piensa en la funcionalidad de este y en cierta forma es correcto, pero existen otras variables que permiten evaluar, entre las cuales se puede mencionar: eficiencia, calidad en el código, software escalable, adaptable, entre otras.

Actualmente la necesidad de gestionar y controlar el desarrollo de *software* en cada una de sus fases se ha acrecentado, por lo que contar con un sistema que permita realizar esto y que a su vez proporcione un panorama general del avance de los proyectos de software es indispensable.

Se presentan temas sobre la gestión específicamente *balance scorecard*, sus perspectivas, tablero de control de mando, indicadores claves de desempeño e indicadores de desempeño en el desarrollo de *software* estos temas servirán para fundamentar la investigación. También se estudiarán los temas de calidad

en el software y sobre Devops metodología que permite gestionar e integrar el desarrollo de software.

Se tiene por objetivo definir un modelo de medición de resultados y control de calidad en una empresa de desarrollo de software para mejorar la productividad del área y contribuir con el crecimiento de la empresa. Para alcanzar el objetivo primero se determina un modelo adecuado para medición de productividad de la empresa y con ello minimizar el tiempo de ocio en los miembros del equipo.

En el presente diseño de investigación en el que se cubrirán los objetivos mencionados. En la primera parte se encuentra una recopilación de fuentes bibliográficas que fundamentaran la investigación. A continuación, se detalla la metodología, definición de variables y fases de esta. En el siguiente capítulo se detallan las herramientas y técnicas de análisis que se utilizaran. El capítulo siguiente cuenta con un detalle de los recursos necesarios para la investigación con los cuales se analizará la factibilidad de esta. Por último, se presenta un cronograma que contiene las actividades con sus respectivos tiempos que serán necesarias para realizar la investigación.

2. ANTECEDENTES

Balance scoredcard inicia como un complemento de las mediciones financieras, como evolución en las nuevas organizaciones. Los indicadores del BSC no se basan solo en mediciones financieras, sino que se han adoptado 4 nuevos procesos los cuales son los siguientes: traducir la visión de la empresa, comunicar y vincular, planificación de negocios y el ultimo Feedback y aprendizaje (Kaplan y Norton, 2004).

Una de las virtudes de BSC es proporcionarnos una fotografía para comprobar cómo se ha alcanzado la estrategia de la organización a corto, medianos y largo plazo (Fernández, 2001).

> La evaluación de la gestión no es solo un instrumento más, como hipótesis de partida entendemos que es necesario plantear la evaluación de la gestión no sólo como un instrumento más, sino como un componente que favorezca un proceso de cambio integral orientado a una gestión estratégica de los recursos y que esté alineada a los objetivos que le dan sentido institucional (Ghiglione, 2021, p. 6).

Callejas, Alarcón y Alvarez (2017), desarrollan para la revista Entramado un artículo llamado *Modelos de calidad del* software, un estado del arte en el cual se mencionan los modelos de gestión de calidad en el software de acuerdo a el nivel de software.

> El modelo DevOps busca cambiar la mentalidad entre los equipos de desarrollo y operaciones para aportar valor a la organización desde el área

> de tecnología. Para ello, busca optimizar y automatizar al máximo las tareas que se ejecutan a lo largo del ciclo de vida de construcción de software (Muñoz, Ordóñez y Bucheli, 2019, p. 82).

La finalidad de DevOps es reducir la cantidad de tiempo entre el repositorio local hasta el repositorio de control de versión después de realizado un commit, da garantía y a su vez que sea implementado en el ambiente de producción, la alta calidad en el software (Muñoz, Ordóñez y Bucheli, 2019).

Delgado y Díaz (2021), desarrollan para la revista Cubana de Ciencias un artículo llamado *Modelos de desarrollo de* software en el cual se analizan cada uno de los modelos de desarrollo de software propuestos en el artículo y se realiza un estudio en donde a través de encuestas a diferentes desarrolladores de software seleccionan el modelo más adecuado. Cada modelo se puede adaptar de acuerdo las necesidades de cada empresa o proyecto. Es importante definir un modelo de desarrollo para poder aplicar las métricas en el desarrollo de software.

Según el caso de estudio planteado por Morales (2021), en la revista ODIGOS en el artículo con nombre *Propuesta de automatización para el seguimiento de ventas en microempresas* se realiza el desarrollo de un proyecto de software se utiliza Azure DevOps para la gestión de tareas, medición de resultado y repositorio de software y a su vez se utiliza la metodología ágil scrum para la gestión del proyecto. Esto es un ejemplo de solución para la medición de resultados, control de versiones, control de calidad en donde se utiliza una metodología ágil y como herramienta de gestión Azure DevOps.

Realizar mediciones es una parte fundamental en cualquier área de la ingeniería. “En la Ingeniería de Software, la productividad se define, con

frecuencia, desde un punto de vista económico, y se entiende como la efectividad del esfuerzo productivo, es decir, la tasa de producción por unidad de entrada" (Hernández, Martínez, Jiménez y Jiménez, 2019, p. 67).

3. PLANTEAMIENTO DEL PROBLEMA

Medición de resultados no certeros para toma de decisiones, entrega de *software* de mala calidad y múltiples versiones de aplicaciones

3.1. Contexto general

La empresa BPO Guatemala S.A le brinda diversos servicios a la empresa MS que es una administradora de seguros en otro país. Los servicios que le brinda son: *mail room*, digitación de documentos, procesamiento de reclamos de seguro, digitalización y visualización de formularios, reclamos y facturas, también todos los datos obtenidos son procesados y se generan reportes para uso de la empresa.

En los últimos años los directores del área han observado que la medición de productividad del área no ha sido totalmente exacta debido a que la forma de medir es con base a puestas en producción, más específicamente proyectos que han sido concluidos exitosamente. Esto no permite tener una visibilidad clara de los resultados a la variabilidad de complejidad de los proyectos. A su vez se ha identificado que existe una diferencia abismal entre los ambientes de producción, pruebas y desarrollo; la razón es porque no se cuenta con un control de versiones adecuado, esto afecta también en la calidad del software ya que en ocasiones se realizan cambios sobre versiones de software desactualizada e incluso versiones no oficiales.

3.2. Descripción del problema

Los KPIs propuestos para la medición de resultados del equipo de desarrollo de software no han sido los adecuados esto ha generado una medición de resultado no certeros para la toma de decisiones.

Por otro lado, se tiene un proceso de control de calidad mal implementado y un repositorio de código fuente no centralizado lo que provoca una entrega de *software* de mala calidad y múltiples versiones de aplicaciones.

En consecuencia, se realizan proyectos de desarrollo de software y no se tiene una forma de medir los resultados de cada proyecto, los proyectos no pasan un control de calidad en la codificación del proyecto, es decir no se controla la calidad del código fuente, se cuenta con varias versiones del mismo proyecto guardadas en diferentes ubicaciones.

La empresa ha evidenciado una reducción de utilidades en un 10 % en el año y se ha identificado que una de las áreas que ha influido es el área de tecnología y desarrollo de software.

3.3. Formulación del problema

Se realiza una formulación de preguntas centrales con los cuales se busca tener una visión más amplia de la problemática que se tiene en la empresa en mención y que servirán para buscar una solución a la misma.

3.3.1. Pregunta central

¿Como el diseño de un sistema de gestión de medición para el control y actualización en una empresa de desarrollo de software mejorara la productividad del área y contribución para el crecimiento de la empresa?

3.3.2. Preguntas auxiliares

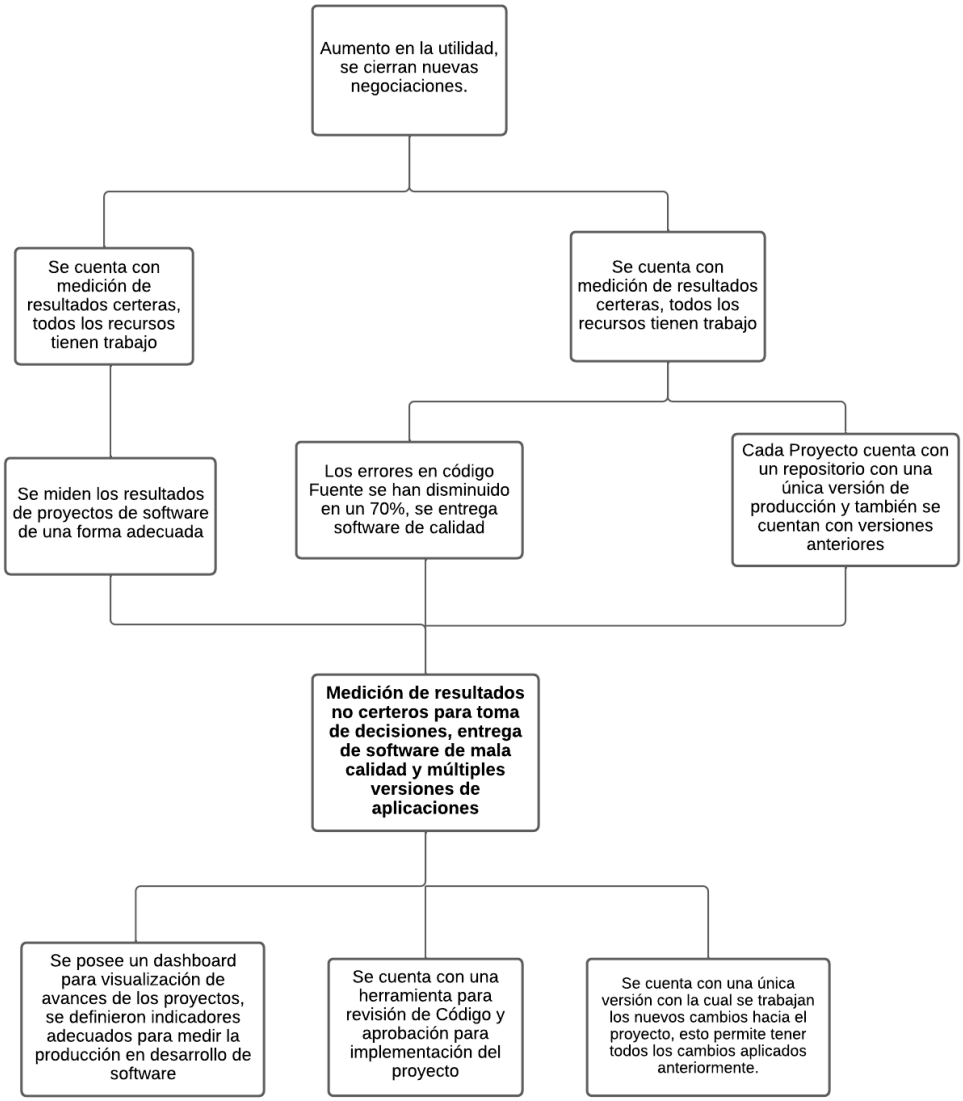
- ¿Cuál es el modelo adecuado para medir la productividad en el desarrollo de software para minimizar el tiempo de no productividad en el equipo de trabajo?

- ¿Cuál es la forma adecuada de gestionar el control de calidad dentro del desarrollo de software para reducir la cantidad de errores en el software?

- ¿Qué herramienta es la adecuada para llevar el control de versiones, avances de proyectos y control de calidad de software?

Como referencia se agrega la figura para descripción de la problemática actual y tener una mejor visión de esta.

Figura 1. **Árbol de problema**



Fuente: elaboración propia.

3.4. Delimitación del problema

La investigación se llevará a cabo se toma en cuenta datos de referencia de la empresa BPO Guatemala S.A específicamente en el área de desarrollo de software.

Se abordará la problemática sobre la falta de mediciones específicas dentro del área para la productividad, control de calidad en el software entregado y las versiones del software.

4. JUSTIFICACIÓN

La realización de la presente investigación se justifica en la línea de investigación de cuadro de control de mando, sistemas de modelo de gestión y calidad del área de gestión estratégica y de sistemas integrados de gestión de la maestría de gestión industrial. Con esta investigación se busca recomendar definir un modelo de medición de resultados y control de calidad en una empresa de desarrollo de software para mejorar la productividad del área y contribuir con el crecimiento de la empresa.

Con este trabajo se propondrán indicadores para realizar la medición de resultados en el área de desarrollo de software, con lo que aportara a la toma de decisiones para aumentar la productividad del área de desarrollo de software.

Adicionalmente, se obtendrá a través de la investigación y análisis la forma correcta de gestionar la calidad en el software y poder asegurar que el software producido cuente con las características importantes: funcionalidad, usabilidad, seguridad, flexibilidad y portabilidad el resultado será la reducción de errores en el software y con esto también se aporta al incremento de productividad dentro de la empresa.

Este trabajo a su vez beneficiará la industria guatemalteca en su rama de desarrollo de software debido a que pueden aplicar las propuestas de gestión de control de calidad y modelo de medición de productividad.

5. OBJETIVOS

5.1. General

Definir un modelo de medición de resultados y control de calidad en una empresa de desarrollo de software que mejorará la productividad del área y contribuirá para el crecimiento de la empresa.

5.2. Específicos

- Determinar el modelo adecuado para medición de productividad en una empresa de desarrollo de software para, minimizar el tiempo de ocio en los miembros del equipo.

- Proponer un sistema de control de calidad en el desarrollo de software para, reducir la cantidad de errores en el software.

- Recomendar una herramienta adecuada para, control de avances de proyectos, control de calidad en el software y versión de software.

6. NECESIDADES A CUBRIR Y ESQUEMA DE LA SOLUCIÓN

A partir del presente trabajo se realizará una propuesta para solucionar la problemática de medición de resultados no certeros para toma de decisiones, entrega de software de mala calidad y múltiples versiones de aplicaciones. Como resultado se tendrá una propuesta de definición de modelo de medición de resultados y control de calidad en una empresa de desarrollo de software para mejorar la productividad del área y contribuir con el crecimiento de la empresa.

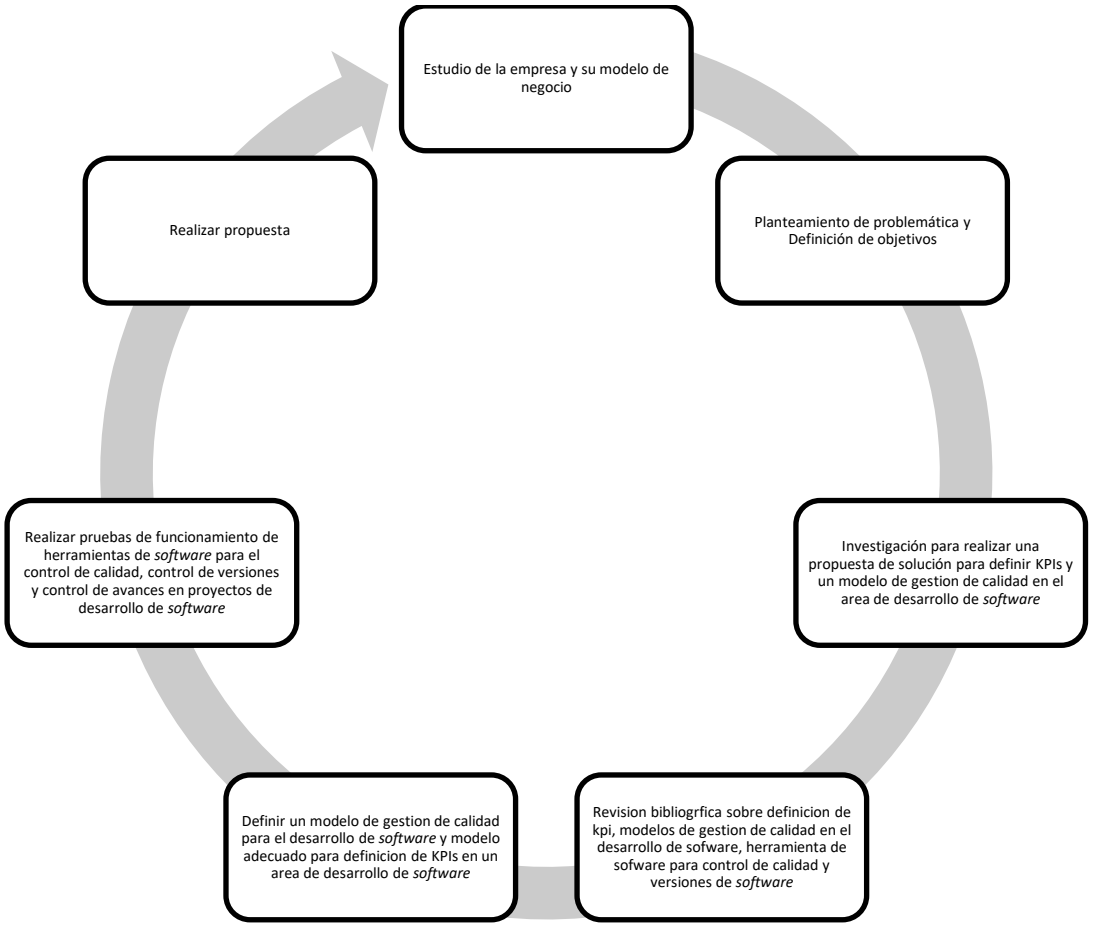
Con este trabajo se determinará un modelo adecuado para la medición de productividad en una empresa de desarrollo de software a partir de esto se busca la optimización de asignación de tareas con los diferentes miembros del equipo y así minimizar el tiempo de ocio de los miembros del equipo como resultado la productividad del área se aumentará.

Asimismo, se propondrá un sistema de control de calidad que garantice la funcionalidad del *software*, así como la escalabilidad, usabilidad, seguridad, flexibilidad y portabilidad. Los beneficios con la propuesta es la reducción de errores en el *software* esto conlleva a una reducción en nuevos proyectos sobre el mismo sistema y evitar *fixes* para solucionar errores en el código.

6.1. Esquema de solución

Se realiza el siguiente esquema de solución para la problemática abordada en donde se define la forma en que se realizara el proceso para búsqueda de solución de la problemática en cuestión.

Figura 2. **Esquema de solución**



Fuente: elaboración propia.

7. MARCO TEÓRICO

A continuación, se definen los temas que servirán de base para buscar una solución a la problemática.

7.1. *Balanced scorecard*

Balanced scorecard también llamado cuadro de control de mando durante los últimos años ha despertado el interés de muchos altos directivos e incluso ha sido adoptado por muchas empresas las cuales buscan, pero ¿Por qué aumenta el interés por *balanced scorecard*? Sin importar que los modelos de planificación y gestión de las empresas no sean tan conocidos, *balanced scorecard* aporta soluciones a los problemas que tienen las empresas y da la tranquilidad que buscan los directivos (Fernández, 2001).

Fernández (2001), define *balanced scorecard* como un: “modelo de gestión que traduce la estrategia en objetivos relacionados, medidos a través de indicadores y ligados a unos planes de acción que permiten alinear el comportamiento de los miembros de la organización” (p. 81).

Las organizaciones en la actualidad se desarrollan y compiten en ambientes complejos; por tal motivo, para poder sobrevivir deben enfocarse en sus objetivos estratégicos y definir los mecanismos que deben utilizar para lograrlos (Ghiglione, 2021).

Busca direccionar las decisiones estratégicas con los objetivos de la empresa, se definen indicadores basados en una visión estratégica proyectados

a plazo largo, con los que se puedan medir desde una perspectiva objetiva y subjetiva.

Sus objetivos: definir, identificar o transformar la estrategia y la visión de la empresa; comunicar y vincular los objetivos e indicadores estratégicos de la compañía; planificar, establecer nuevos objetivos y alinear las iniciativas estratégicas empresariales; aumentar el *feedback* y la formación estratégica en cada área de la compañía.

Las empresas actuales basan su fuerza competitiva en términos como: eficiencia, eficacia, calidad y rapidez.

Los parámetros financieros permiten medir los resultados ya obtenidos en el pasado, pero para crear valor futuro hemos de analizar y tener en cuenta a nuestros clientes, empleados y tecnología entre otros. Esto a su vez favorecerá en la creación un modelo de gestión con el cual se busque lograr las metas definidas.

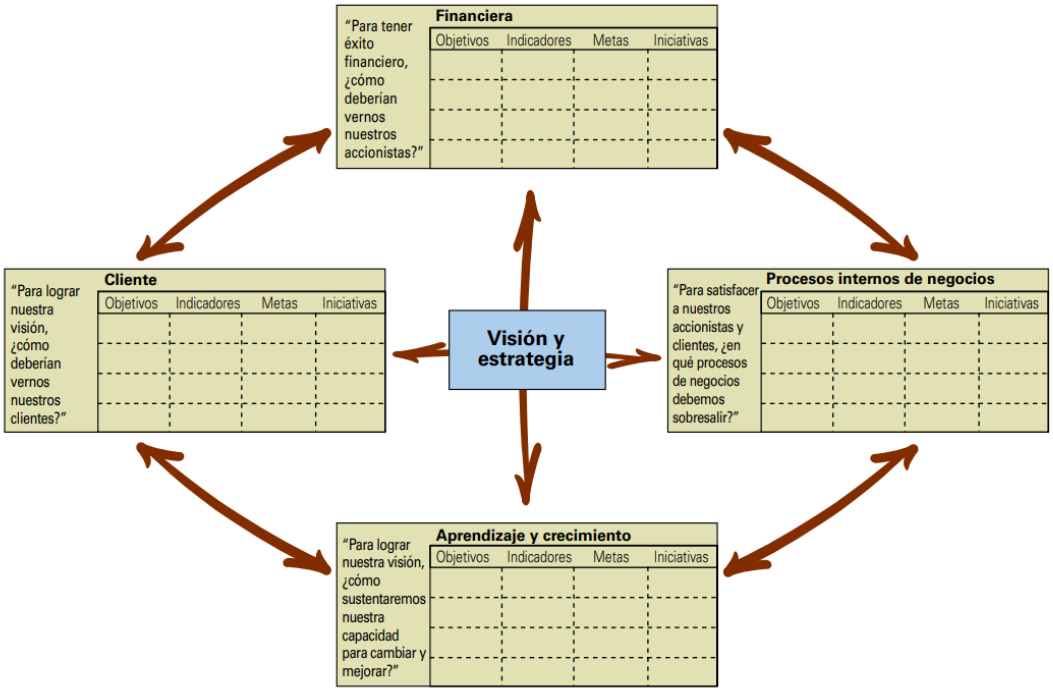
7.1.1. Perspectivas de un *balance scorecard*

Balanced scorecard no solo se enfoca en la perspectiva financiera además de esta toma en cuenta 3 perspectivas que son importantes para realizar una gestión que estén alineadas para alcanzar los objetivos.

Las 4 perspectiva dentro de las que trabaja *balanced scorecard* son: la financiera, los clientes, los procesos internos y aprendizaje y crecimiento (Kaplan y Norton, 2004).

La figura 3 muestra las 4 perspectivas de enfoque de *balanced scorcard*, cada una cuenta con metas, objetivo e indicadores a su vez cuenta con iniciativas.

Figura 3. **Cuatro perspectivas**



Fuente: Kaplan y Norton. (1996). *Translating Strategy into Action – The Balanced Scorecard*.

La perspectiva financiera es un conjunto de la visión de los accionistas de la compañía junto con la generación de valor de la empresa. Da respuesta a la pregunta ¿Cuáles deberían ser los indicadores que con un alto rendimiento reflejen el crecimiento de la empresa desde el punto de vista financiero? La perspectiva financiera tiene como objetivo primordial generar valor para la sociedad, en compañías privadas con ánimos de lucro (Dávila, 1999).

La perspectiva del cliente se centra en el posicionamiento que tiene la compañía dentro del mercado y segmentos de mercados en el que quiere competir (Dávila, 1999).

Esta perspectiva se apoya en dos indicadores el primero es la cuota de mercado y un índice de comparación de precios de la compañía con sus competidores, con estos se verá reflejado o no el posicionamiento de la compañía. La perspectiva interna analiza indicadores de procesos internos que son claves para que la compañía pueda posicionarse y así crear una estrategia para lograr sus objetivos (Dávila, 1999).

Para que una compañía pueda crecer debe aprender y la última perspectiva nos habla de esto, la perspectiva de aprendizaje y crecimiento se centra en que para poder crecer es necesario tener un modelo de negocio apropiado, la importancia de invertir en los recursos y en mejoras de los procesos (Dávila, 1999).

7.1.2. Tablero de comando integral

Funciona como un sistema que organiza y a su vez presenta datos los cuales sirven para tener una mejor gestión y a su vez elegir el rumbo de la organización, es utilizado por los altos mandos. A través del tablero de mando integral se pueden visualizar alertas que funcionan en base a indicadores con los que se evalúa de forma integral la gestión. Es la manera de representar la gestión de la compañía de una forma integral.

A su vez el tablero de control integral tiene como principales objetivos:

> Medir los avances en cumplimiento de la misión, la visión, los valores, los objetivos de la organización; alinear los indicadores y las metas de la dirección con la cadena de valor de la organización y los indicadores y metas de las áreas; integrar el plan estratégico con los planes operativos de las áreas; crear Tableros de Control de cada área y alinearlos con el Tablero de Control de la dirección; identificar los diferentes tipos de indicadores existentes en un proceso; sincronizar los objetivos y metas de la dirección general con las demás áreas; orientar los esfuerzos hacia la satisfacción de las necesidades de los usuarios, empleados, proveedores y la comunidad (Fleitman, 2008, p. 273).

Establecer los indicadores de gestión de una forma correcta y objetiva es la clave para el éxito o fracaso para implementar un sistema de mejora continua y evaluación de desempeño. Los indicadores propuestos deben reunir especificaciones que se alinean con los objetivos de la compañía con respecto a la perspectiva con la cual se elabora. Es necesario que se defina el objetivo a alcanzar, el cual tendrá que estar enfocado en la forma de medición de resultados alineado con los objetivos que se tiene la compañía (Ghiglione, 2021).

Cada compañía implementara cada uno de los tipos de tableros con respecto a las necesidades de esta, se toma en cuenta la visión y estrategias adoptadas, los diferentes tableros son:

> Tablero de control directivo: es aquel que permite monitorear los resultados de la empresa en su conjunto y hacer foco en los diferentes temas claves en que puede segmentarse. Su monitoreo es de aproximadamente cada mes. Puede incluir indicadores de todos los sectores para los directivos claves o sectorizado para un directivo.

Tablero de control estratégico: nos brinda la información interna y externa necesaria para conocer la situación y evitar llevarnos sorpresas desagradables importantes respecto al posicionamiento estratégico y a largo plazo de la empresa.

Tablero de control integral: información relevante para que la alta dirección de una empresa pueda conocer la situación integral de su empresa. Engloba a las tres perspectivas anteriores (Ghiglione, 2015, p. 31).

7.1.3. Indicadores claves de desempeño

Los indicadores de desempeño son una herramienta que en la actualidad se han convertido en indispensables dentro de las compañías. Migueles (2014), lo define como:

> Las siglas K.P.I. vienen de las palabras *key performance indicators* que traducido al español se define como Indicadores claves de desempeño. Son métricas financieras o no financieras usadas para ayudar a una organización a definir y medir el progreso que se da según las metas planteadas, se utilizan para cuantificar objetivos que reflejan el rendimiento de una organización, y que generalmente se recogen en su plan estratégico. Es un tema del que se suele hablar con frecuencia en el mundo de la gestión (p. 15).

Utilizar K.P.Is proporciona las siguientes ventajas:

“Medición constante; adaptación del negocio; motivación en los colaboradores; tranquilidad de los inversionistas. Esto debido a que existe una

forma certera de visualizar el éxito o fracaso del trabajo dentro de las compañías" (Núñez, 2018, párr. 4-7).

7.1.3.1. ¿Cómo definir los kpis?

Para definir los kpis se deben tomar en cuenta 2 cosas: los objetivos y la información que se obtendrá. Los kpis proporcionan un punto intermedio para entender los objetivos de una forma porcentual y así poder visualizar de una forma clara cómo se comporta la compañía con respecto a la estrategia (Núñez, 2018).

Para definir los kpis se utiliza el modelo S.M.A.R.T y tomar en cuenta lo siguiente: ¿Cuáles son los objetivos de la estrategia? ¿Qué estrategia se ha definido? ¿Qué tácticas/acciones se van a desarrollar? ¿Qué datos/resultados a nivel cuantitativo y cualitativo se pueden obtener de dichas acciones? (Núñez, 2018).

El modelo S.M.A.R.T son siglas en ingles que a continuación se presentan junto con la forma de utilizar este modelo para definir kpi.

- *Specific* (específico). Sea específico con respecto a lo que espera lograr.
    - *Measurable* (mensurable). Use parámetros cuantificables para facilitar el seguimiento del rendimiento de su proyecto
    - *Attainable* (alcanzable). Asegúrese de que sus objetivos sean realistas. Los castillos en el aire son geniales, sin embargo, usted desea establecer hitos alcanzables.

- *Relevant* (relevante). ¿Son sus métricas pertinentes a su proyecto? Si ha identificado sus objetivos esperados, esto debería facilitar la determinación de si los KPI son relevantes o no.

- *Time-bound* (con límite de tiempo). Establezca un marco de tiempo. Tenga un principio y un final para que pueda establecer líneas de base e hitos. Este intervalo establecido también lo ayudará a identificar cosas como la estacionalidad, las migraciones, los lanzamientos de productos y más (Microsoft 365 Team, 2019, párr. 5-9).

De esta forma se pueden definir los kpis y se asegura que estén alineados con los objetivos y estrategia de la compañía.

7.1.3.2. KPIs del desarrollador de *software*

Existen infinidad de métricas para el desarrollo de *software* las hay desde el punto de vista de calidad, usabilidad, productividad, entre otras.

Para medir la productividad del desarrollo de *software* una de las principales mediciones es la que se realiza al desarrollador. Es la forma de evaluar su trabajo y a partir de esto incentivar o llamar la atención depende los resultados de este. A continuación, Black (2020), lista las principales metricas que se utilizan para medir el trabajo de los desarrolladores.

Tiempo de entrega (*lead time*): el tiempo de entrega es el tiempo que tarda algo de principio a fin.

Cantidad de código: los equipos de desarrollo pueden mirar esta métrica de *software*, también llamada miles de líneas de código (KLOC), para determinar el tamaño de una aplicación. Si este KPI es alto, podría indicar que los desarrolladores fueron productivos en sus esfuerzos de programación.

Trabajo en curso (WIP): en un contexto de ingeniería de *software,* WIP es un trabajo de desarrollo en el que el equipo ha comenzado a trabajar y que ya no está en el backlog.

Velocidad ágil: para calcular la velocidad, un equipo de desarrollo de *software* ágil analiza los *sprints* anteriores y cuenta el número de historias de usuario o puntos de historia completados a lo largo del tiempo

Tasa de éxito de la meta del *sprint:* esta métrica de *software* calcula el porcentaje de elementos que completó el equipo de desarrollo en el *backlog* del *sprint*. Es posible que un equipo no termine el 100 % del trabajo durante un sprint determinado.

Número de versiones de *software:* los equipos ágiles y de DevOps dan prioridad a los lanzamientos de *software* continuos y frecuentes. Con este KPI, los equipos pueden realizar un seguimiento de la frecuencia con la que lanzan *software*, ya sea mensual, semanal, diaria, por hora o en cualquier otro período de tiempo, y si esa cadencia finalmente ofrece suficiente valor comercial (párr. 9-14).

Cabe mencionar que no son las únicas métricas que existen, cada compañía puede definir propias de acuerdo con sus necesidades.

7.2. Calidad en el *software*

La industria del *software* ha ido en crecimiento conforme el avance tecnológico y esto se debe a que en todos los sectores el uso de aplicaciones ha aumentado. Las aplicaciones de *software* cada vez brindan más facilidad a la hora de gestionar, administrar, comercializar para cada tipo de actividad existe un *software* que hace fácil o minimiza el trabajo.

Callejas, Alarcón y Álvarez(2017),han afirmado lo siguente con respecto a que es *software* :

> El *software* es una de las herramientas de mayor utilidad en la optimización de procesos en las organizaciones, con el propósito de contar y ofrecer optimización, eficiencia y satisfacción de necesidades, razón por la cual el *software* debe contar con criterios que garanticen su calidad (p. 237).

Las organización dedidcadas al desarrollo de *software* muchas veces no le dan la importancia que debe tener la calidad de *software* y esto a su vez impacta en el funcionamiento en la operaciones de la empresa que solicito el mismo. Callejas, Alarcón y Álvarez(2017), definen calidad de *software* como:

> El término calidad de *software* se refiere al grado de desempeño de las principales características con las que debe cumplir un sistema computacional durante su ciclo de vida, dichas características de cierta manera garantizan que el cliente cuente con un sistema confiable, lo cual aumenta su satisfacción frente a la funcionalidad y eficiencia del sistema construido (p. 237).

También, existe una defición estandarizada de el termino calidad la cual es por parte del IEEE (1990), y es la siguiente: el grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario.

De manera que, al entender las definiciones de calidad de *software* y comprender que es *software* se puede concluir que para obtener calidad de *software* se debe tomar en cuenta los requerimientos y especificicaciones del cliente ademas de con una adecuado control de calidad el IEEE (1990), conjunto de actividades diseñados para evaluar la calidad de el desarrollo o producción de productos en este caso aplicado a *software*.

7.2.1. Modelos de calidad del *software*

En el sitio EcuRed (2017), se define un modelo calidad como: “conjunto de buenas prácticas para el ciclo de vida del *software,* enfocado en los procesos de gestión y desarrollo de proyectos” (párr. 3).

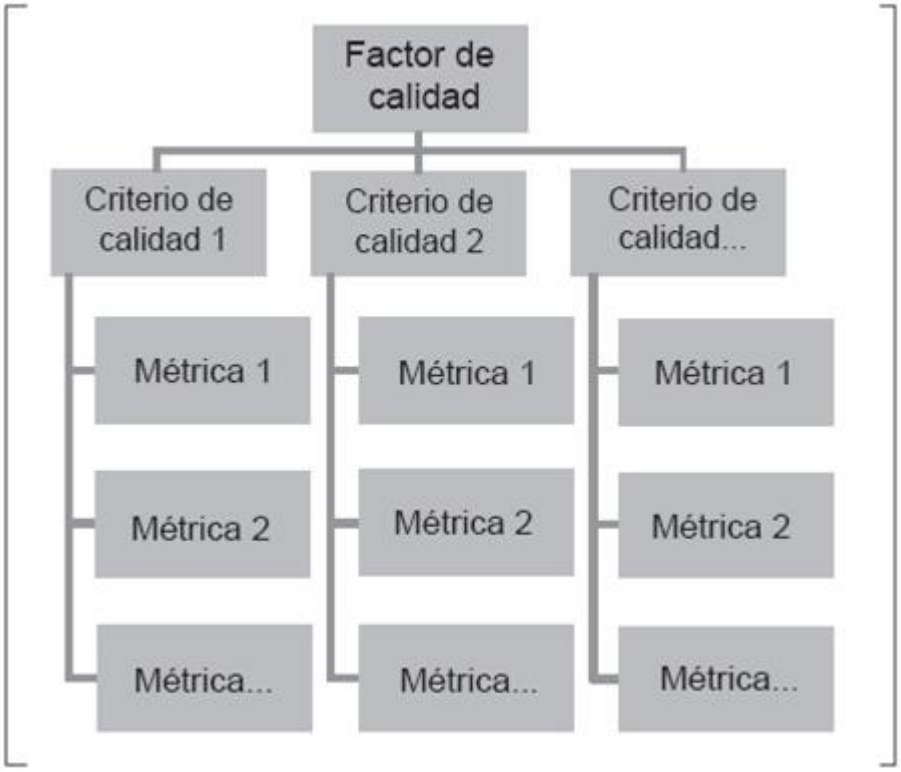
Otra definición de modelo de calidad dada por Scalone (2006) refiere:

> Los modelos de calidad son aquellos documentos que integran la mayor parte de las mejores prácticas, proponen temas de administración en los que cada organización debe hacer énfasis, integran diferentes prácticas dirigidas a los procesos clave y permiten medir los avances en calidad (p. 28).

A partir de la definición de modelo de calidad es importante que dentro de la metodología de desarrollo que se adopte en la compañía se elija el modelo adecuado según los procesos, prácticas y las mediciones que se requieran.

Entre los modelos de calidad existentes se pueden mencionar los siguientes: Boehm, McCall, ISO 9126 entre otros, cada uno tiene un enfoque distinto cada uno conformado por factores de calidad que son la base principal del modelo, en la siguiente figura se detalla cómo se descompone un factor y que a su vez puede formarse por varios criterios.

Figura 4. **Estructura de la calidad del *software***



Fuente: Callejas, Alarcón y Álvarez. (2017). *Modelos de calidad del software, un estado del arte*.

7.2.2. Modelos de calidad del *software*

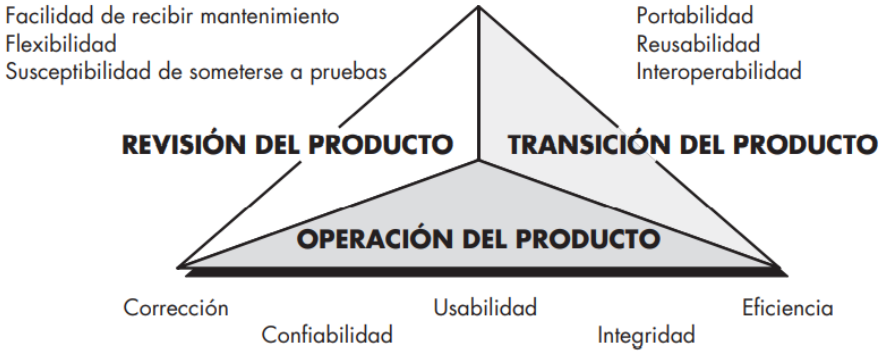
Para facilitar el control y medición de calidad en el *software* es necesario definir factores. Cada modelo está compuesto por diferentes factores con respecto al enfoque de cada uno. El concepto genérico de calidad para su fácil entendimiento se descompone en factores.

Se pueden clasificar en dos grupos grandes los cuales son los siguientes: factores que pueden ser medidos directamente (errores/KDL/unidad de tiempo) y factores que solo pueden ser medidos indirectamente (la facilidad de uso o de mantenimiento). En ambos casos para medir la calidad se debe compara el *software* con alguna referencia y llegar a un nivel aceptable de medición (Pressman, 2010).

7.2.2.1. Factores de calidad según Mccall

Pressman (2010), menciona que los factores de calidad del *software,* mostrados en la figura 4, tienen 3 aspecto principales de un producto *software:* sus características operativas, su capacidad de cambios y su adaptabilidad a nuevos entornos.

Figura 5. **Factores de calidad de Mccall**



Fuente: Pressman. (2010). *Ingeniería del software, un enfoque práctico*.

Tabla I. **Factores de calidad**

Punto de vista	Factor
Revisión del producto	Mantenibilidad
	Flexibilidad
	Testeabilidad
Transición del producto	Portabilidad
	Reusabilidad
	Interoperabilidad
Operación del producto	Correctitud
	Confiabilidad
	Eficiencia
	Integridad
	Usabilidad

Fuente: elaboración propia.

7.2.2.2. Factores de calidad según ISO 9126

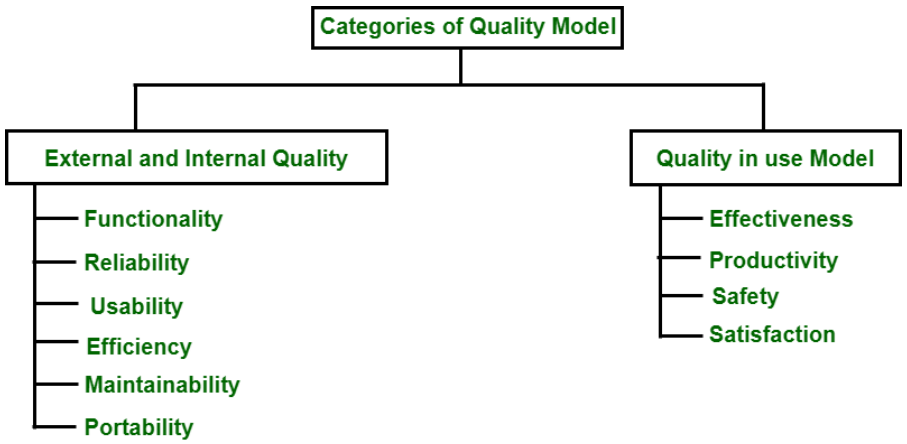
El sitio GeeksforGeeks (2021) definen ISO 9126 de la siguiente forma:

ISO / IEC 9126 es un estándar internacional propuesto para garantizar la calidad de todo el *software* - productos intensivos que incluye sistemas como la seguridad crítica donde en caso de falla de la vida útil del *software* estará en peligro. ISO, es decir, la Organización Internacional de Normalización e IEC, es decir, la Organización Eléctrica Internacional, han desarrollado los estándares

ISO / IEC 9126 para la ingeniería de *software* - calidad del producto para proporcionar una especificación y un modelo de evaluación integrales para la calidad del producto de *software.*

ISO 9126 divide en 2 modelos de calidad cada uno cuenta con factores con los que se evalúa el *software* de acuerdo con el enfoque, los modelos son las siguientes: calidad interna y externa y calidad en uso en la figura 5 se detalla cada una.

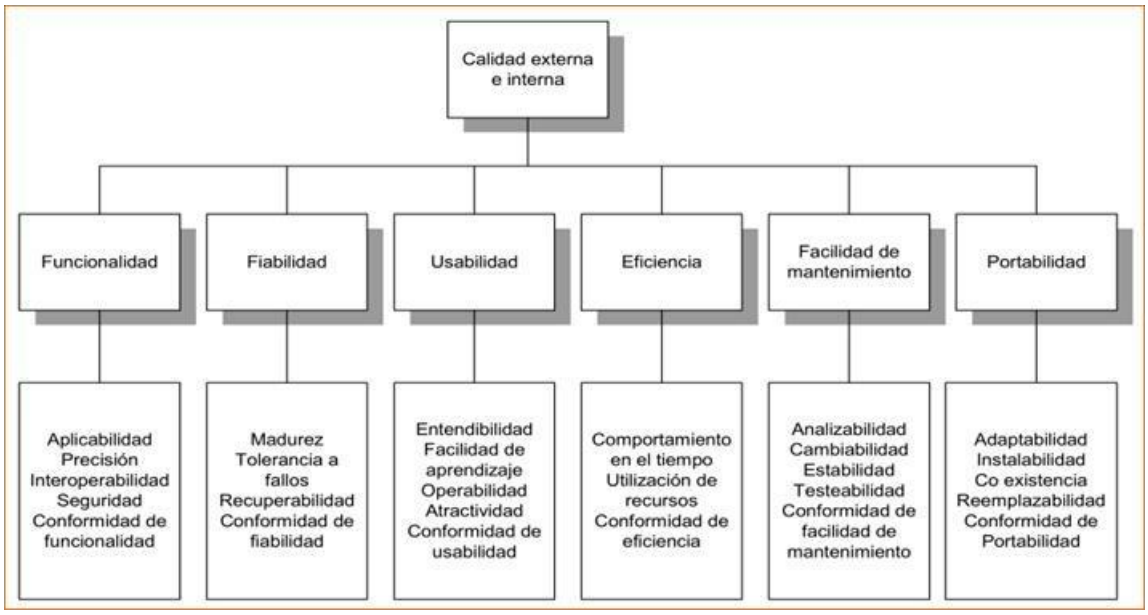
Figura 6. **Modelos de calidad**



Fuente: GeeksforGeeks. (2021). ISO/*IEC 9126 in Software Engineering*. Recuperado de https://www.geeksforgeeks.org/iso-iec-9126-in-software-engineering/?ref=gcse. Consultado: 20 de agosto, 2021.

El modelo de calidad externa e interna puede representarse en la figura 6 en donde se muestra cada factor que considera este modelo y cada subfactor que es tomado en cuenta para asegurar la calidad y en la tabla II se describe cada factor o característica para este modelo.

Figura 7. **Modelo de calidad del producto de *software* para la calidad externa e interna**



Fuente: Informática. (2020). *ISO 9126*. Recuperado de https://sites.google.com/site/informaticamcprats/iso-9126. Consultado: 21 de agosto, 2021.

Tabla II. **Características de la calidad interna y externa, definido en ISO/IEC 9126-1**

Característica	Definición
Funcionalidad	La capacidad del producto *software* para proveer las funciones que satisfacen las necesidades explícitas e implícitas cuando el *software* se utiliza bajo condiciones específicas.
Fiabilidad	La capacidad del producto *software* para mantener un nivel especificado de funcionamiento cuando se está utilizando bajo condiciones especificadas.
Usabilidad	La capacidad del producto *software* de ser entendido, aprendido, usado y atractivo al usuario, cuando es usado bajo las condiciones especificadas.
Eficiencia	La capacidad del producto *software* para proveer un desempeño apropiado, de acuerdo con la cantidad de recursos utilizados y bajo las condiciones planteadas.
Facilidad de mantenimiento	Capacidad del producto *software* para ser modificado. Las modificaciones pueden incluir correcciones, mejoras o adaptación del *software* a cambios en el entorno, y en requerimientos y especificaciones funcionales.
Portabilidad	La capacidad del *software* para ser trasladado de un entorno a otro.

Fuente: Informática. (2020). *ISO 9126*. Recuperado de https://sites.google.com/site/informaticamcprats/iso-9126. Consultado: 21 de agosto, 2021.

La calidad en uso es definida por la norma ISO/IEC 9126-1según Anciniega (2017), como:

> La perspectiva del usuario de la calidad del producto *software* cuando éste es usado en un ambiente específico y un contexto de uso específico. Ésta mide la extensión para la cual los usuarios pueden conseguir sus metas en un ambiente particular, en vez de medir las propiedades del *software* en sí mismo (párr. 11).

Existen diferentes tipos de usuario y cada usuario tendrá una forma diferente de dar por satisfecho el funcionamiento del *software,* por ejemplo: un operario del sistema tiene un panorama distinto en comparación a un administrador del sistema, indiferente del tipo de usuario todos son tomados con la misma importancia.

La figura 6 se representa el modelo de calidad en uso y a su vez se muestran los factores que componen, los cuales son: efectividad, productividad, integridad, satisfacción a su vez en la tabla III se define cada factor.

Figura 8. **Modelo de calidad del producto *software* para la calidad en uso**

Fuente: Informática. (2020). *ISO 9126*. Recuperado de https://sites.google.com/site/informaticamcprats/iso-9126. Consultado: 21 de agosto, 2021.

Cada compañía utiliza un modelo que se adecue a sus necesidades y con el que se sientan cómodos trabajar, no existe el modelo perfecto incluso algunas compañías adoptan sus propios modelos de calidad en el *software* que pueden ser una mezcla de otros modelos de calidad.

Tabla III. **Características de la calidad en uso, definido en ISO/IEC 9126-1**

Característica	**Definición**
Efectividad	La capacidad del producto *software* para permitir a los usuarios lograr las metas especificadas con precisión y completitud en un contexto de uso específico.
Productividad	La capacidad del producto *software* para permitir a los usuarios emplear cantidades apropiadas de recursos en relación con la efectividad lograda en un contexto de uso específico.
Integridad	La capacidad del producto *software* para lograr niveles aceptables de riesgo de daño a las personas, negocio, *software*, propiedad o entorno en un contexto de uso específico.
Satisfacción	La capacidad del producto *software* para satisfacer a los usuarios en un contexto de uso específico.

Fuente: Informática. (2020). *ISO 9126*. Recuperado de https://sites.google.com/site/informaticamcprats/iso-9126. Consultado: 21 de agosto, 2021.

7.2.3. Aseguramiento de la calidad

Durante la década de los 50, los sistemas del DoD (del inglés *Deparment of Defense of USA*) le dieron lugar a incorporar *Software*. Inicialmente estos proyectos eran puestos en marcha de forma empírica es decir sin ninguna planificación lo cual desencadenaba en problemas de tiempo, dinero y a su vez de calidad. Se plantea la solución a esta problemática con la verificación y validación independientes (IV&V del inglés *independent verification and validation*). Este era un proceso que tenía por objetivo evaluar la calidad y funcionamiento correcto de una forma rigurosa, se realizaba durante todas las fases de construcción del *software*.

SQA (*Software Quality Assurance*) surge de la metodología verificación y validación independientes (IV&V), la cual es una versión evolucionada de la misma.

Para asegurar la calidad, Fernández, García y Beltrán (1995), mencionan que “para controlar la calidad del *software* es necesario, ante todo, definir los parámetros, indicadores o criterios de medición”. (p. 41). Hacen referencia a Tom de Marco, en donde el menciona que usted no puede controlar lo que no se puede medir.

7.3. Devops

Devops surge de la necesidad de unificar esfuerzo, culturas, prácticas y metodologías entre los roles dentro del desarrollo de *software* (Desarrollo, operaciones IT, Ingeniería de la calidad y seguridad) con el objetivo de aumentar la capacidad de producir con una mejor calidad se reduce el tiempo de entrega en los servicios/productos (Microsoft azure, 2020).

Otra definición de devops es la que dan Ebert, Gallardo, Hernantes y Serrano(2016):

“DevOps se trata de desarrollo, aprovisionamiento rápidos y flexibles en los procesos de negocios. Integra eficientemente el desarrollo, la entregay operaciones, se facilita así una conexión ágil y fluida de estossitios tradicionalmente separados” (p. 94).

La union de dos terminos en ingles (Develpment / Operations) se combinan para formar el termino DevOps. A su vez al llevarlo a la vida real Devops se centra en instaurar una cultura en donde los equipos que tienen como fin el desarrollo de aplicaciones de *software* trabajen en conjunto y perseguir los objetivos. De esta manera se puedan coordinar y colobarar para entregar un producto de calidad, en menos tiempo y se invierten menos recursos.

Al aplicar la cultura devops dentro de una organización traerá beneficios como se menciona en Microsoft Azure (2020):

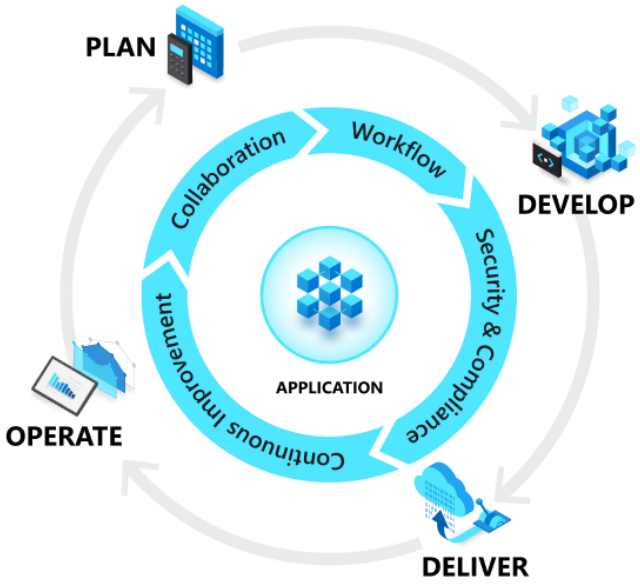
> Los equipos que adoptan la cultura, las prácticas y las herramientas de DevOps mejoran el rendimiento y crean productos de más calidad en menos tiempo, lo que aumenta la satisfacción de los clientes. Esta mejora de la colaboración y la productividad es fundamental también para alcanzar objetivos de negocio como estos:reducción del tiempo de comercialización; adaptación al mercado y a la competencia; mantenimiento de la estabilidad y la confiabilidad del sistema; mejora del tiempo medio de recuperación (párr. 3).

7.3.1. Ciclo de vida DevOps

DevOps influye en el ciclo de vida de las aplicaciones a lo largo de las fases de planeamiento, desarrollo, entrega y uso. Cada fase depende de las demás y las fases no son específicas de un rol. En una auténtica cultura de DevOps, todos los roles están implicados de algún modo en todas las fases (Microsoft Azure, 2020).

En la figura 10, Microsoft (2020), describe las 4 fases del ciclo de vida devops se toma en cuenta que en este diagrama presentado se omiten las fases de monitores y feedback, estas fases se dan después de la fase de desarrollo cuando la aplicación ya ha sido entrega.

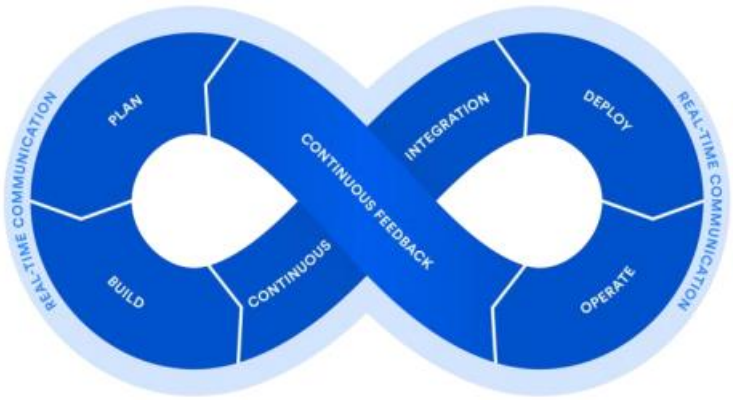
Figura 9. **Ciclo de vida**



Fuente: Microsoft Azure. (2020). *¿Qué es DevOps*? Recuperado de https://azure.microsoft.com/es-es/overview/what-is-devops/#devops-overview. Consultado el 25 de agosto de 2021.

En la figura 11 se detallan las fases del ciclo de vida devops de una aplicación, cada etapa que conforma el ciclo se repite como un bucle, al implementar devops se persigue la mejora continua con lo que se crea un ciclo infinito.

Figura 10. **Bucle ciclo de vida**



Fuente: Durández. (2020). *Diseño e implementación de un proceso Devops con control de calidad y seguridad del código*.

Cada fase que conforma el ciclo de vida devops es una parte importante de esta metodología por lo que se detalla lo que dijo Fernández (2018) a continuación:

> Plan: se basa en habilitar los mecanismos necesarios para dar la posibilidad a cualquier usuario de aportar de forma continua sus ideas y que estas puedan ser transformadas en requerimientos u objetivos, siendo estos priorizados e incluidos en próximas iteraciones en forma de historias de usuario; DevOps ayuda a la construcción de *software* se da paso a mecanismos que sean capaces de favorecer la creación de entornos de

desarrollo repetibles es la fase de construcción; la integración continua (CI) es una técnica que tiene como objetivo detectar los posibles problemas que pueda tener el *software* de forma temprana, permitiendo así solucionarlos; uno de los grandes objetivos de DevOps es facilitar el paso de un *software* que se desarrolla una versión funcional por eso en la fase de despliegue provee herramienta que lo facilitan.

En la fase de monitoreo se busca una versión de *software* sea funcional, es fundamental realizar un seguimiento de esta: monitorización del rendimiento de la aplicación y servidor, seguimiento de problemas, incidentes y cambios; la cultura DevOps fomenta la utilización de herramientas que facilitan exista un flujo continuo de comunicación para obtener retroalimentación (párr. 5-8).

7.3.2. Integración continua

Integración continua en el desarrollo de *software* surge como solución a los problemas de: pérdida de código causada por no contar con un repositorio central; múltiples versiones; dificultad para encontrar un error; demora en la validación y publicación de actualizaciones al *software*; reducción en la calidad del producto a entregar.

Figura 11. **Integración continua - Entrega continua (AWS)**

Fuente: Amazon Web Services, Inc. (2018). *¿Qué es la integración continua*? Consultado el 28 de agosto de 2021. Recuperado de https://aws.amazon.com/es/devops/continuous-integration/.

De los problemas anteriormente descritos surge la necesidad de subsanarlos. Con la integración continua, como puede empezar a intuirse por su nombre, los desarrollos del equipo se combinan cada poco tiempo. Estos cambios se fusionan o integran, normalmente en un repositorio del código compartido con control de versiones. Cuando un desarrollador integra los cambios de su código con el del resto del equipo en ese repositorio remoto, normalmente se disparan las pruebas automatizadas que detectan la correcta integración de esos cambios sobre el código actual.

7.3.3. Control de versiones

Control de versiones es la práctica de administrar el código por versiones, permite dar un seguimiento de las revisiones y del historial de cambios para facilitar la revisión y la recuperación del código. Esta práctica suele implementarse con sistemas de control de versiones, como Git, que permite que varios desarrolladores colaboren para crear código. Estos sistemas proporcionan un proceso claro para fusionar mediante combinación los cambios en el código

que tienen lugar en los mismos archivos, controlar los conflictos y revertir los cambios a estados anteriores (Microsoft Azure, 2020).

El uso del control de versiones es una práctica de DevOps fundamental que ayuda a los equipos de desarrollo a trabajar juntos, dividir las tareas de programación entre los miembros del equipo y almacenar todo el código para poder recuperarlo fácilmente si fuese necesario (Microsoft Azure, 2020).

Las principales características son: manejo de conflictos: en un equipo de desarrollo es común que se trabaje sobre el mismo proyecto y esto ocasione conflictos a la hora de guardar los cambios. Con esta herramienta se puede controlar los conflictos, los cambios se agregan conforme se agregan al repositorio y se actualiza la versión local que cada miembro del equipo tiene.

Histórico: uno de los problemas comunes en el desarrollo de *software* se da al perder una versión de un archivo y esto no permita regresar a una versión anterior del sistema. El contar con un histórico garantiza el fácil acceso a versiones anteriores, tener un historial de las modificaciones que se han realizado.

Copias de seguridad: contar con un repositorio central asegura que se pueda contar con copias de seguridad del código y que pueden ser accesibles en cualquier momento.

8. PROPUESTA DE ÍNDICE DE CONTENIDOS

9. METODOLOGÍA

En el siguiente capítulo se realiza una presentación del tipo de estudio, el diseño, los alcances y para variables para la presente investigación.

9.1. Características del estudio

El estudio tendrá un enfoque mixto debido a que cuenta con variables que serán medidas de forma cualitativa y cuantitativa.

El alcance será explicativo debido a que se analizar las causas de la baja productividad, falta de calidad en el desarrollo de *software* y a su vez de dará una propuesta de un sistema que permita gestionar, medir y actualizar los procesos de desarrollo de *software*.

El diseño adoptado será no experimental, pues, la información de un sistema de gestión de medición para el control y actualización de desarrollo de *software* serán datos reales y no serán modificados. A su vez será un estudio transversal ya que se tomarán datos históricos que permitan evidenciar la falta de calidad y baja productividad en el desarrollo de *software* dentro del área de desarrollo de *software*.

9.2. Unidades de análisis

El análisis a través de un *software* que se desarrolla para un cliente de proyecto de *software* implementado de donde se obtendrá la información sobre

implementaciones del *software*, análisis de productividad del equipo de desarrollo Ragnarok.

9.3. Variables

A continuación, se presentan las variables para el desarrollo de la investigación.

Primera pregunta auxiliar ¿Cuál es el modelo adecuado para medir la productividad en el desarrollo de *software* para minimizar el tiempo de no productividad en el equipo de trabajo?

Tabla IV. **Variables primera pregunta auxiliar**

Variable	Definición teórica	Definición operativa
Productividad	Medición de tiempo y recursos invertidos en el desarrollo de *software*	Medición de horas hombre H y monetarias Q
Tiempo	Cantidad de espacio entre entregas de *software*	Horas H

Fuente: elaboración propia.

Segunda pregunta auxiliar ¿Cuál es la forma adecuada de gestionar el control de calidad dentro del desarrollo de *software* para reducir la cantidad de errores en el *software*?

Tabla V. **Variables segunda pregunta auxiliar**

Variable	Definición teórica	Definición operativa
Gestionar	Acciones para controlar	Procesos de desarrollo de *software*, indicadores 75 – 85 %
Control de calidad	Forma de asegurar la calidad en el *software*	Indicadores de funcionamiento del *software*75 – 85 %
Errores	Mal funcionamiento en el *software* desarrollado	Indicadores de funcionamiento del *software*75 – 85 %

Fuente: elaboración propia.

Tercere pregunta auxiliar ¿Qué herramienta es la adecuada para llevar el control de versiones, avances de proyectos y control de calidad de *software*?

Tabla VI. **Variables tercera pregunta auxiliar**

Variable	Definición teórica	Definición operativa
Herramienta	Sistema para controlar y gestionar	indicadores 75 – 85 %
Control de versiones	Repositorio para gestionar y versionar el código de *software*	Indicador de portabilidad 75-85 %
Avance de proyectos	Medición de productividad en el desarrollo de *software*	Indicadores de avance de proyectos

Fuente: elaboración propia.

9.4. Fases de estudio

A continuación, se detallan las fases con las que contara la investigación.

9.4.1. Fase 1: Búsqueda bibliográfica

En la primera fase se realizará una consulta de todas las bibliografías posibles relacionadas al tema, para enriquecer los conocimientos sobre gestión de calidad de *software*, como medir la productividad en el *software* y herramientas o metodologías que permitan gestionar el desarrollo de *software*.

9.4.2. Fase 2: Observación y recolección de datos

En la segunda fase se realizará una recolección de datos de los últimos 6 meses en el área de desarrollo de *software*, para enriquecer los conocimientos sobre las causas de error en el *software*, la disminución en la calidad del *software* y la baja productividad en el área se tomará como base el equipo Ragnarock y el proyecto de un cliente especifico.

9.4.3. Fase 3: Análisis de información

En la tercera fase se realizará el análisis de información para identificar las causas principales en la disminución de productividad e identificar las fases de desarrollo de *software* que toman más tiempo. A su vez los análisis correspondientes para identificar en qué fase se dan más errores en el *software*.

Para esta fase se plantea utilizar herramientas como: tablas de datos y medidas de tendencia central.

9.4.4. Fase 4: Interpretación de información

En la cuarta fase se realizará una interpretación de resultados con los cuales se podrán identificar las causas en la disminución de productividad en el área de desarrollo de *software*, la disminución en la calidad del *software* entregado a nuestro cliente. Definir un modelo de gestión que permita dar solución a la problemática.

9.4.5. Fase 5: Redacción de propuesta

Luego de la interpretación de los resultados se redactará una propuesta dé solución a la problemática específicamente que brinde y determine el modelo adecuado para medición de productividad en una empresa de desarrollo de *software* para minimizar el tiempo de no productividad en los miembros del equipo ragnarock que a su vez proponga un sistema de control de calidad en el desarrollo de *software* para reducir la cantidad de errores en el *software* y que se pueda recomendar una herramienta adecuada para control de avances de proyectos, control de calidad en el *software* y versión de *software* .

10. TÉCNICAS DE ANÁLISIS DE INFORMACIÓN

En el presente capitulo se presentan las técnicas de análisis utilizadas para la presente investigación.

Para la toma de datos de productividad en el área de desarrollo de *software* se realizará a través de datos históricos; los cuales se componen de tiempo de desarrollo, tiempo de pruebas, tiempo de implementación y tiempo de soporte. Con los datos obtenidos se realizará un análisis de media y moda para establecer el tiempo actual en cuanto a tiempo en el desarrollo. Para presentar el resultado se utilizarán un diagrama de pie con el que se evidenciara que proceso es el que más tiempo toma.

Al tener datos históricos de seis meses se procederá a realizar un análisis de medidas de tendencia central se realizará el cálculo de media aritmética, moda y sus respectivas desviaciones. Se utilizará una tabla de datos para agrupar e identificar los tipos de errores y comparar los tipos de errores y así evidenciar las más comunes.

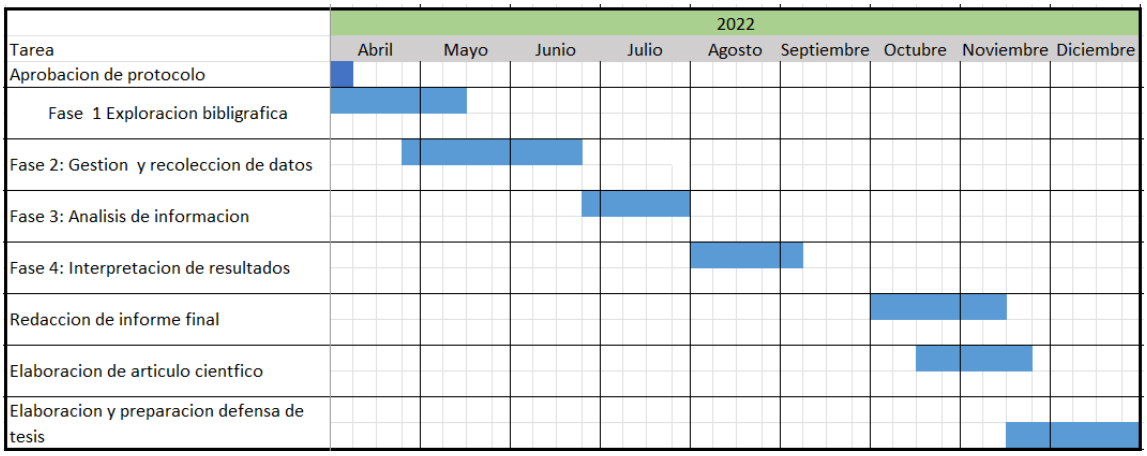
Se analizarán herramientas que ayuden a minimizar la cantidad de errores en el *software* y así aumentar la calidad en el *software*. Para esto se realizará un proceso de observación entre las diferentes herramientas: Azure GIT de Microsoft y AWS Devops se realizará proceso de pruebas para realizar una tabla de datos de comparación.

11. CRONOGRAMA

A continuación, se muestra un detalle de las actividades que se realizaran para la presente investigación.

Se observa que la división de tiempo está dada por meses y cada mes se divide en 4 semanas de trabajo.

Figura 12. **Cronograma de actividades**



Fuente: elaboración propia.

12. FACTIBILIDAD DEL ESTUDIO

Para la realización del presente trabajo de investigación se utilizarán recursos propios, los cuales se detallan a continuación.

Tabla VII. **Recursos necesarios para la investigación**

Descripción	Tipo de recurso	Costo Q.	Cantidad	Total Q.	Porcentaje
Depreciación de equipo de computo	Equipo tecnológico	250	15	3750	35.89 %
Honorarios Tesista	Humano	2500	1	2500	23.92 %
Honorarios asesores	Humano	2500	1	2500	23.92 %
Hojas bond	Papelería	100	1	100	0.96 %
Impresiones	Papelería	100	1	100	0.96 %
Internet	Suministro	500	1	500	4.78 %
Energía eléctrica	Suministro	500	1	500	4.78 %
Imprevistos	Imprevistos	350	1	500	4.78 %
Subtotal				7950	100 %
Descuento	Honorarios Tesista		1	2500	
Total				5450	

Fuente: elaboración propia.

Los recursos con los que se cuentan son suficientes para cubrir el costo de la investigación por lo que se considera factible ser realizada.

13. REFERENCIAS

1. Amazon Web Services, Inc. (20 de septiembre, 2018). *¿Qué es la integración continua*? [Mensaje de un blog]. Recuperado de https://aws.amazon.com/es/devops/continuous-integration/.

2. Black, R. (7 de septiembre, 2020). *23 métricas de desarrollo de software que monitorear hoy*. [Mensaje de un blog]. Recuperado de https://www.computerweekly.com/es/consejo/23-metricas-de-desarrollo-de-*software*-que-monitorear-hoy.

3. Callejas, M., Alarcón, A. y Álvarez, A. (20 de junio, 2017). Modelos de calidad del *software*, un estado del arte. *RevistaEntramado*,*13*(1), 236-250.

4. Dávila, A. (11 de septiembre, 1999). Nuevas herramientas de control: el cuadro de mando integral. *Revista de Antiguos Alumnos*, 34-42.

5. Delgado, L. y Díaz, L. (31 de marzo, 2021). Modelos de Desarrollo de *Software*. *Revista Cubana de Ciencias Informáticas*, *15*(1), 37-51.

6. Durández, E. (2020). *Diseño e implementación de un proceso devops con control de calidad y seguridad del código*. (Tesis de licenciatura). Universidad Carlos III de Madrid, España.

7. Ebert, C., Gallardo, G., Hernantes, J. y Serrano, N. (5 de mayo, 2016). DevOps. *IEEE Software*, *16*(1), 94-100.

8. EcuRed. (18 de mayo, 2017). *Modelo de calidad*. [Mensaje de un blog]. Recuperado de https://www.ecured.cu/Modelo_de_calidad.

9. Fernández, A. (5 de mayo, 2001). El Balanced Scorecard: ayudando a implantar la estrategia. *Revista de Antiguos Alumnos*, *1*(1), 81-83.

10. Fernández, O., García, D. y Beltrán, A. (28 de diciembre, 1995). Un enfoque actual sobre la calidad del *software*. *Revista ACIMED*,*3*(3), 40-42.

11. Fernández, R. (20 de diciembre, 2018). *DevOps: 6 fases clave para el proceso del desarrollo de software*. [Mensaje de un blog]. Recuperado de https://www.izertis.com/es/-/blog/devops-6-fases-clave-para-el-proceso-del-desarrollo-de-*software*.

12. Fleitman, J. (2008). *Evaluación integral para implantar modelos de calidad*. México: Editoria Pax.

13. GeeksforGeeks. (29 de octubre, 2021). ISO/*IEC 9126 in Software Engineering*. [Mensaje de un blog]. Recuperado de https://www.geeksforgeeks.org/iso-iec-9126-in-*software*-engineering/?ref=gcse.

14. Ghiglione, F. (2015). Gestión de RR. HH del personal de planta permanente de la Honorable Cámara de Diputados (provincia de La Pampa). Desafíos para una adecuada evaluación de desempeño. (Tesis de maestría). Universidad Nacional de La Pampa, Argentina.

15. Hernández, G., Martínez, Á., Jiménez, R. y Jiménez, F. (19 de noviembre, 2019). Métricas de productividad para equipo de trabajo de desarrollo ágil de *software*: una revisión sistemática. *Revista Tecnológicas*, *22*(1), 63-81.

16. IEEE Computer Society. (1990). *IEEE Standard Glossary of Software Engineering Terminology*. Estados Unidos: The institute of Electrical and Electronics Engineers.

17. Informática. (9 de octubre, 2020). *ISO 9126*. [Mensaje de un blog]. Recuperado de https://sites.google.com/site/informaticamcprats/iso-9126.

18. Kaplan, R. y Norton, D. (1996). *Translating Strategy into Action – The Balanced Scorecard*. Estados Unidos: Harvard Business.

19. Kaplan, R. y Norton, D. (2004). *Strategy maps: converting intangible assets into tangible outcomes*. Estados Unidos: Harvard Business.

20. Microsoft Azure. (5 de agosto, 2020). ¿Qué es DevOps? [Mensaje de un blog]. Recuperado de https://azure.microsoft.com/es-es/overview/what-is-devops/#practices.

21. Microsoft 365 Team. (9 de octubre, 2019). *Indicadores clave de rendimiento (KPI): qué son y cómo usarlos*. [Mensaje de un blog]. Recuperado de https://www.microsoft.com/es-ww/microsoft-365/business-insights-ideas/resources/what-are-kpis-and-how-to-use-them.

22. Migueles, W. (2014). Indicadores clave de rendimiento en proyectos de desarrollo de *software* a partir de un objetivo organizacional. (Tesis de maestría). Pontifica Universidad Católica de Valparaíso, Chile.

23. Morales, J. (10 de octubre, 2021). Propuesta de automatización para el seguimiento de ventas en microempresas. *RevistaODIGOS*,*2*(3), 77-98.

24. Muñoz, D., Ordóñez, H. y Bucheli, V. (1 de junio, 2019). Lineamientos para la implementación del modelo CALMS de DevOps en mipymes desarrolladoras de *software* en el contexto sur colombiano. *Revista Guillermo de Ockham*, *18*(1), 81-91.

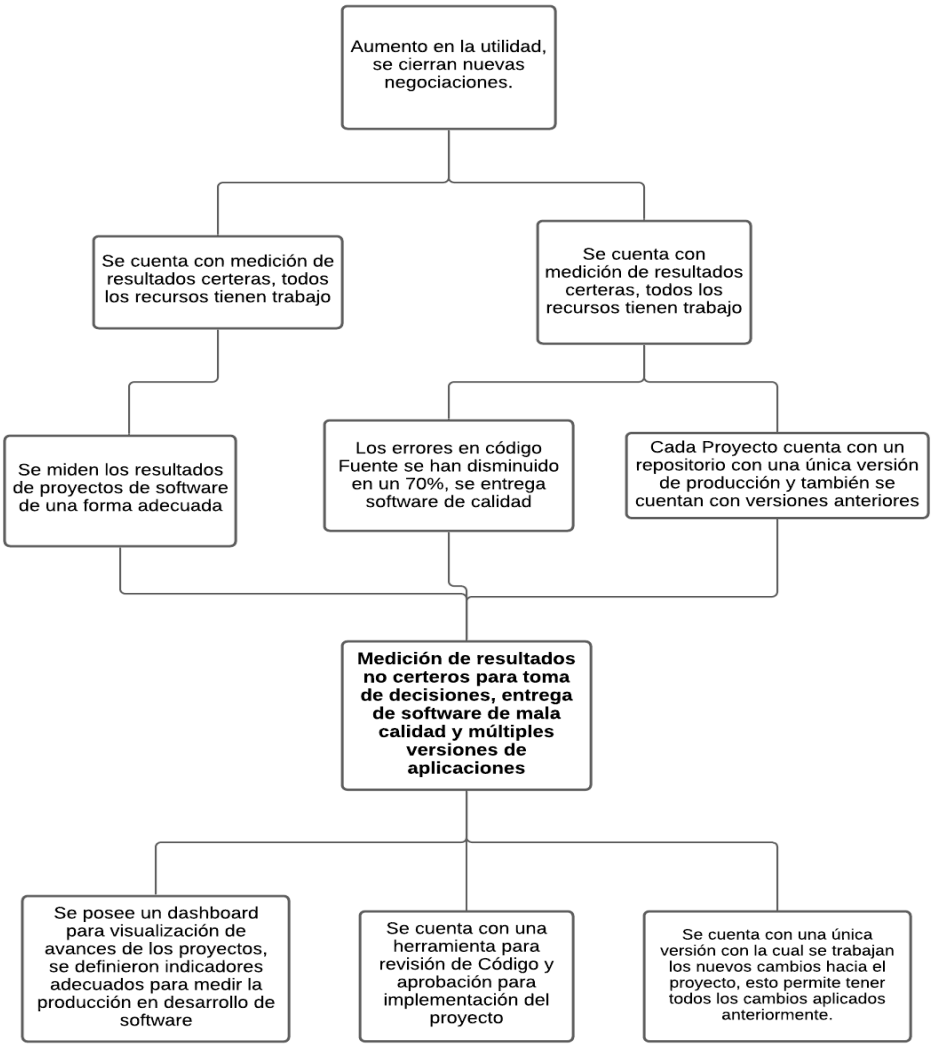
25. Núñez, V. (1 de noviembre, 2018). *¿Qué son los indicadores KPI y qué tipos existen?* [Mensaje de un blog]. Recuperado de https://vilmanunez.com/indicadores-kpi.

26. Pressman, R. (2010). *Ingeniería del software, un enfoque práctico*.Estados Unidos: McGraw-Hill.

27. Scalone, F. (2006). Estudio comparativo de los modelos y estándares de calidad del *software*. (Tesis de maestría). Universidad Tecnológica Nacional, Argentina.

14. APÉNDICES

Apéndice 1. **Árbol de objetivos**



Fuente: elaboración propia.

Apéndice 2. **Matriz de coherencia**

Título de la investigación	Planteamiento del problema de investigación	Preguntas de investigación	Objetivos
Propuesta de un sistema de medición para el control y actualización en una empresa de Desarrollo de *software*	medición de resultados no certeros para toma de decisiones, entrega de *software* de mala calidad y múltiples versiones de aplicaciones	Principal ¿Como el diseño de un sistema de gestión de medición para el control y actualización en una empresa de desarrollo de *software* mejorara la productividad del área y contribución para el crecimiento de la empresa? Auxiliares ¿Cuál es el modelo adecuado para medir la productividad en el desarrollo de *software* para minimizar el tiempo de no productividad en el equipo de trabajo? ¿Cuál es la forma adecuada de gestionar el control de calidad dentro del desarrollo de *software* para reducir la cantidad de errores en el *software*? ¿Qué herramienta es la adecuada para llevar el control de versiones, avances de proyectos y control de calidad de *software?*	General Proponer un sistema de medición para el control y actualización en una empresa de Desarrollo de *software* Específicos Determinar el modelo adecuado para medición de productividad en una empresa de desarrollo de *software.* Implementación de sistema de control de calidad en el desarrollo de *software.* Determinar herramienta adecuado para control de avances de proyectos, control de calidad en el *software* y versión de ***software*.**

Fuente: elaboración propia.