UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE INDUSTRIAS AGROPECUARIAS Y CIENCIAS AMBIENTALES

CARRERA DE COMPUTACIÓN

Tema: "Sistema de Gestión de Configuración de Software Usando DevOps"

Trabajo de Integración Curricular previo a la obtención del título de Ingeniera en Ciencias de la Computación

AUTORA: Valencia Mafla Angela Victoria

TUTOR: Ing. Guano Cárdenas Carlitos Alberto MSc.

Tulcán, 2026.

CERTIFICADO DEL TUTOR

Certifico que la estudiante Valencia Mafla Angela Victoria con el número de cédula 0401905112 ha desarrollado el Trabajo de Integración Curricular: "Sistema de Gestión de Configuración de Software Usando DevOps"

Este trabajo se sujeta a las normas y metodología dispuesta en la Codificación del Reglamento de Régimen Académico y de Estudiantes de la UPEC, por lo tanto, autorizo la presentación de la sustentación para la calificación respectiva.

Firmado electrónicamente por:
CARLITOS ALBERTO GUANO CARDENAS
Validar únicamente con FirmaEC

_____

Ing. Guano Cárdenas Carlitos Alberto MSc.

TUTOR

Tulcán, enero de 2026

AUTORÍA DE TRABAJO

El presente Trabajo de Integración Curricular constituye un requisito previo para la obtención del título de Ingeniera en la Carrera de Computación de la Facultad de Industrias Agropecuarias y Ciencias Ambientales

Yo, Valencia Mafla Angela Victoria con cédula de identidad número 0401905112 declaro que la investigación es absolutamente original, auténtica, personal y los resultados y conclusiones a los que he llegado son de mi absoluta responsabilidad.

Valencia Mafla Angela Victoria

AUTORA

Tulcán, enero de 2026

ACTA DE CESIÓN DE DERECHOS DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Yo Valencia Mafla Angela Victoria declaro ser autora de los criterios emitidos en el Trabajo de Integración Curricular: "Sistema de Gestión de Configuración de Software Usando DevOps" y eximo expresamente a la Universidad Politécnica Estatal del Carchi y a sus representantes de posibles reclamos o acciones legales.

Valencia Mafla Angela Victoria

AUTORA

Tulcán, enero de 2026

AGRADECIMIENTO

A la Universidad Politécnica Estatal del Carchi

por brindarme una educación de calidad y permitirme formarme como una profesional de calidad.

A mi tutor Ing. Carlitos Guano MSc.

por impartirme sus conocimientos y permitir una evaluación constante del proyecto logrando así culminarlo.

Al Msc. Juan Pablo López

por su colaboración con el proyecto y la disposición brindada.

A todos los que conforman TICS

por formar parte del proyecto y su colaboración en cada fase de este.

A mis docentes

por brindarme conocimiento a través de cada clase y dedicarse con pasión a la docencia.

DEDICATORIA

A mi padre

por brindarme el amor y valores que guían mi vida.

A mi madre

por su consistencia y cariño.

A mis hermanos

por no dejarme sola en los momentos más difíciles.

A pantera

por ser mi compañero y mi mascota preciada.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

El presente trabajo denominado "Sistema de gestión de configuración de software usando DevOps (Desarrollo y Operaciones de Tecnologías de la Información)" tiene como propósito mejorar el control de versiones la trazabilidad de cambios y la administración de actualizaciones de los módulos virtuales de los sistemas integrados de la Universidad Politécnica Estatal del Carchi (UPEC). La investigación usó un enfoque mixto fundamentado en diferentes técnicas como la recolección de información que permitió el diagnóstico de los problemas principales presentes en los procesos actuales del desarrollo como la ausencia de lineamientos estandarizados, el uso limitado de herramientas de control de versiones y la dependencia a los procesos de forma manual afectando la trazabilidad. Tomando como base el diagnóstico inicial, se eligieron las mejores herramientas y prácticas compatibles con el contexto institucional. Considerando el criterio de necesidad técnica dentro de los sistemas, se diseñó una guía metodológica estructurada donde se definen roles, se realizan los flujos del control de versiones, procedimientos para la gestión del cambio, lineamientos para la actualización del módulo y actividades de soporte basadas en automatización; sobre todo en buenas prácticas, estandarización de procesos y DevOps. Como resultado de la investigación se elaboró una guía para la aplicación de prácticas DevOps, la cual fue evaluada por el personal de la unidad de desarrollo perteneciente a la dirección de TIC de la UPEC quienes confirmaron la pertinencia aplicabilidad y la contribución a la mejora de los procesos actuales dentro de la unidad con esto se concluyó que la propuesta es un instrumento viable para fortalecer la gestión de configuración facilitando la adopción progresiva de prácticas DevOps y reducción de errores derivados del manejo manual de versiones y actualizaciones.

Palabras Claves: SGCS, DevOps, Control de versiones, Trazabilidad, Automatización.

ABSTRACT

The present work, entitled "Software Configuration Management System Using DevOps (Information Technology Development and Operations)," aims to improve version control, change traceability, and the management of updates for the virtual modules of the integrated systems at the Universidad Politécnica Estatal del Carchi (UPEC). The research employed a mixed-method approach, supported by various data collection techniques that made it possible to diagnose the main problems in the current development processes. Among these issues were the lack of standardized guidelines, the limited use of version control tools, and the dependence on manual procedures; factors that significantly affect traceability. Based on this initial diagnosis, the tools and practices most suitable for the institutional context were selected. Accordingly, and considering the technical needs within the systems, a structured methodological guide was designed. This guide defines roles, establishes workflows for version control, details procedures for change management, outlines guidelines for module updates, and includes support activities based on automation, with a strong emphasis on best practices, process standardization, and DevOps principles As a result of the study, a guide for the implementation of DevOps practices was developed and subsequently evaluated by the development unit staff of the UPEC TIC Department. The evaluators confirm edits relevance, applicability, and contribution to improving the unit's current processes. Therefore, it is concluded that the proposal is a viable instrument for strengthening configuration management, facilitating the gradual adoption of DevOps practices, and reducing errors arising from the manual handling of versions and updates.

Keywords: SGCS, DevOps, Version Control, Trace ability, Automation.

INTRODUCCIÓN

La gestión eficiente del desarrollo y mantenimiento de software es fundamental para que la calidad y la seguridad de los sistemas informáticos se vean garantizados en las instituciones educativas. La unidad de formación académica de tercer nivel denominada Universidad Politécnica Estatal del Carchi (UPEC) cuenta con diversos módulos virtuales que soportan procesos académicos y administrativos, sin embargo, la inexistencia de un sistema formal y estandarizado de administración de composición del software genera dificultades en el control de versiones, seguimiento de cambios y actualización de estos módulos.

La adopción de prácticas DevOps es una solución para optimizar el flujo de existencia del software, al integrar desarrollo, operaciones y automatización mediante herramientas específicas. Un sistema para la administración configuración basado en DevOps permite controlar eficazmente las versiones, mejorar la trazabilidad de los cambios y facilitar la administración de actualizaciones, lo que produce en un desarrollo más ágil, seguro y confiable.

Una guía metodológica con estructura que se adapte a las necesidades específicas de la institución de educación superior Universidad Politécnica Estatal del Carchi (UPEC), especificando los pasos para la creación de un Sistema de gestión de configuración de software (SGCS) enfocado en el uso de DevOps, a su vez analizando las herramientas para que se acople al entorno universitario.

En el documento se busca orientar y abordar la necesidad dentro de la institución mediante un análisis de situación actual real de la unidad de desarrollo la identificación de las principales limitaciones en la gestión de configuración y la selección de prácticas DevOps que se ajusten a una realidad qué se puede ver con algunos de los insumos se podrá realizar una guía metodológica que propone un modelo estructurado para poder estandarizar los procesos el control de versiones asegurar la trazabilidad y mejorar la administración de todas las actualizaciones dentro de los módulos virtuales es así que el documento puede ofrecer fundamentación teórica proceso investigativo resultados obtenidos y la propuesta metodológica que permita fortalecer la gestión de software en la institución.

I. EL PROBLEMA

1.1. PLANTEAMIENTO DEL PROBLEMA

En el desarrollo de software la estandarización de procesos es importante porque garantiza la calidad de los productos desarrollados existen algunos enfoques tradicionales como el modelo en cascada que demuestran ser limitados en entornos donde se ocupan requerimientos que cambian constantemente es por eso que las investigaciones recientes señalan que estos modelos rígidos dificultan la adaptación y afecta al éxito de los proyectos. (Offerman et al., 2022) señalan que muchas organizaciones trabajan con métodos tradicionales que dificultan la colaboración entre desarrollo y operaciones, ya que "separating these two capabilities can hinder communication and collaboration" (p. 1). Lo que significaría que al Separar estas 2 capacidades genera gran comunicación y buena colaboración entre los diferentes equipos.

DevOps ha surgido no solo como una metodología, sino como una transformación en la cultura y en las técnicas normativas actuales que permite integrar el desarrollo y operaciones, potenciando la colaboración interdisciplinaria, la automatización de procesos y la entrega continua. Es así como "DevOps es un conjunto de prácticas y conceptos para realizar un cambio cultural en las organizaciones... proporcionando aplicaciones y servicios a gran velocidad... fundamental para alcanzar los objetivos del negocio" (Aranguren & Ruiz, 2023, p. 18).

Además de las limitaciones propias de las metodologías tradicionales, persisten desafíos técnicos en los procedimientos para la elaboración de programas informáticos a nivel global. Uno de los más críticos es la ausencia de trazabilidad en los componentes que conforman un sistema, lo que impide establecer relaciones claras entre versiones que transitan por diferentes entornos. Esta situación compromete la fiabilidad del producto final, ya que puede derivar en que se instalen en producción versiones no verificadas correctamente "No existe trazabilidad del software: resulta imposible establecer una relación entre los distintos componentes de software que pasan de un ambiente a otro... Esta falta de trazabilidad podría tener

como consecuencia la instalación, en el ambiente de producción, de componentes que no han sido probados" (Rodríguez, 2021, p. 3).

A ello se suma que muchos procesos de actualización de versiones se siguen realizando de forma manual, lo cual incrementa significativamente la posibilidad de errores humanos y genera incertidumbre operativa "La mayor parte del tiempo, el traspaso de versiones de un ambiente a otro implica la copia manual de archivos... por lo que la posibilidad de cometer errores siempre queda abierta" (Rodríguez, 2021, p. 3).

En el contexto ecuatoriano, las organizaciones enfrentan importantes desafíos relacionados con la gestión eficiente de sus sistemas de información y procesos para la formación de los programas. La persistencia de métodos obsoletos y poco estructurados que dificultan la trazabilidad, el control y la mejora continua se muestra como un problema principal. Se señala que "La gestión documental también representa uno de los inconvenientes es que las organizaciones enfrentan hoy, debido a la ineficiencia y la obsolescencia de los sistemas y métodos que utilizan"(Ruales, 2024, p.4).

La desinformación sobre la estructura y estandarización impide alcanzar niveles óptimos de calidad y sostenibilidad en los proyectos tecnológicos del país. "Es escasamente habitual que las compañías de Ecuador dispongan un procedimiento establecido para la administración de configuraciones de software, mucho menos conforme a un estándar, dado que es conocido que desarrollan software utilizando metodologías ágiles que privilegian las entregas funcionales por encima de la documentación y el mantenimiento." (Villacres, 2023, p. 12).

En la Dirección de Tecnologías e Información de la Comunicación de la Universidad Estatal del Carchi, específicamente en la unidad de desarrollo de software se presenta la limitada aplicación prácticas formales en la gestión de configuración de software. Los procesos de desarrollo de sistemas realizados por el equipo tienen dificultades en sus mecanismos de estandarización de control de versiones, trazabilidad de cambios y la documentación haciendo que se difícil tener una evolución controlada de los sistemas.

1.2. FORMULACIÓN DEL PROBLEMA

El escaso conocimiento sobre los sistemas de gestión de configuración de software (GCS) en la Universidad Politécnica Estatal del Cachi limita la adopción adecuada de prácticas de Desarrollo y Operaciones en Tecnologías de la información lo que desencadena la pérdida de integridad de proyectos de desarrollo, fallos de control de versiones y dependencia de procesos manuales durante el 2025.

1.3. JUSTIFICACIÓN

En América del Sur, la adopción del enfoque DevOps está ganando popularidad para la entrega continua y la cooperación entre los equipos de desarrollo y operaciones. En Colombia, se ha identificado que "DevOps emerge como una herramienta clave puesto que tiene el potencial de agilizar el proceso de desarrollo y entrega de software" (Diaz, 2024, p.15).

Los sistemas de gestión para la configuración de un software benefician a personas que busquen estandarizar los procesos dentro de un proyecto, brindando herramientas eficaces para garantizar la calidad del producto. Asimismo, "la gestión de la configuración es una disciplina especializada... que permite ofrecer la versión correcta de la solución al cliente y reducir la pérdida de trabajo" (Zúñiga et al., 2020, p. 467).

En Cuenca-Ecuador, se ha identificado que "las pequeñas organizaciones logran nivel 1 de capacidad por implementar sistemas de versiones basados en Git", dando como resultado la "automatización de procesos, trazabilidad y comunicación efectiva en los equipos de desarrollo" (Zúñiga et al., 2020, p. 470).

Esta investigación cimienta bases para trabajos investigativos futuros que se enfoquen en la implementación de sistema de gestión de configuración enfocados al uso prácticas DevOps, permitiendo ampliar el conocimiento investigativo y práctico en la estandarización de procesos para espacios educativos de tercer nivel.

Se beneficia de forma directa al equipo de desarrolladores de la dirección de TIC ya que el control de versiones mejora y se reduce los errores técnicos dando como resultado una optimización de procesos e indirectamente se da una

mejora a la comunidad universitaria ya que se podrá obtener plataformas institucionales más estables y una mejor interacción de usuario.

1.4. OBJETIVOS Y PREGUNTAS DE INVESTIGACIÓN

1.4.1. Objetivo General

Desarrollar una guía metodológica para la implementación de un Sistema de Gestión de Configuración de Software con el uso de DevOps, orientado al control de versiones, la trazabilidad de cambios y la administración de actualizaciones de los módulos virtuales de los sistemas integrados de la Universidad Politécnica Estatal del Carchi (UPEC).

1.4.2. Objetivos Específicos

- Analizar las necesidades actuales en la gestión de versiones y actualizaciones para la identificación de oportunidades de mejora de los módulos virtuales de la UPEC.
- Identificar herramientas DevOps para un sistema de gestión de configuración de software basado en compatibilidad con la UPEC.
- Diseñar la guía metodológica que oriente la implementación del sistema de gestión de configuración de software usando DevOps.

1.4.3. Preguntas de Investigación

- ¿Cuáles son las principales necesidades que tiene la UPEC en gestión de versiones, trazabilidad y actualización de módulos virtuales?
- ¿Qué herramientas DevOps son adecuadas para un sistema de gestión de configuración de software en la UPEC?
- ¿Cómo debe estructurarse una guía metodológica para un sistema de gestión de configuración de software usando DevOps?

II. FUNDAMENTACIÓN TEÓRICA

2.1. ANTECEDENTES DE LA INVESTIGACIÓN

En el campo de ingeniería de programas informáticos, las estructuras de gestión la configuración ha crecido como una parte crucial para el control y el seguir de cambios en el desarrollo de proyectos. A lo largo de los años, los sistemas de gestión de configuración han sido reconocidos como un pilar principal para que se garantice la integridad, trazabilidad y reproducción de productos en el software. Aun así, la integración de sistema de gestión de configuración con metodologías emergentes como DevOps están en constante incomprensión, debido a la escasa literatura investigativa que asuma su implementación y uso. Por esto, se ha señalado que "el Supply Chain Management (SCM) se menciona en la literatura sobre DevOps como algo importante o como parte de DevOps, pero a menudo se omiten los detalles sobre su implementación o uso" (Hochbergs & Nilsson, 2020, p. 3), así se evidencia la necesidad de mayor investigación para especificar su rol y estandarizar lineamientos de integración.

Para el desarrollo de un proyecto de software se debe optimizar procesos para que la calidad del producto se mantenga durante el ciclo de la vida del proyecto, de la misma manera con la evolución de la tecnología y el impulso de las metodologías emergentes se debe promover el uso práctico de DevOps y la estandarización de procesos a través de la gestión de configuración.

En términos globales, los Sistemas Gestión de Configuración permiten establecer una infraestructura estandarizada para que los equipos cooperen de manera ágil y con eficiencia. La unificación entre DevOps y Sistema de Gestión de Configuracioón mejora el nivel de trazabilidad, control de versiones y eficiencia operativa en el ciclo de vida del software. Como se ha mencionado, "para lograr esto e incorporar el objetivo general de DevOps, y al mismo tiempo mantenerse al ritmo del mercado, el flujo de trabajo desde la idea hasta, por ejemplo, un producto terminado, debe ser lo más eficiente y eficaz posible" (Hochbergs & Nilsson, 2020, p. 2). Este punto de vista promueve la idea de SGC como facilitador de los objetivos de DevOps en entornos dinámicos y altamente

competitivos La evolución del desarrollo de software está promoviendo la adopción de metodologías que integren la eficiencia operativa, automatización y colaboración entre equipos. Por esto, DevOps ha surgido como un nuevo enfoque.

La Gestión de la Configuración del Software se ubica como un pilar, ya que permite controlar los cambios y versiones del proyecto de software, dando facilidad para la coordinación y trazabilidad en entornos complicados, "la Gestión de la Configuración del Software (SCM, por sus siglas en inglés) es una tarea que implica identificar, organizar y controlar los cambios y modificaciones en el software […] y permitir la colaboración mediante la coordinación del trabajo con principios y prácticas, procesos y soporte de herramientas" (Hochbergs & Nilsson, 2020, p. 2). Esta relación entre DevOps y SGC resulta básica para cumplir con los ciclos del desarrollo de proyectos.

Tabla 1. Aportes del documento

Tema	Aporte
Falta de definición clara de DevOps	El documento indica que DevOps *no tiene una definición unificada*, provocando interpretaciones distintas entre organizaciones. (p. 7–10)
SCM en DevOps	La literatura menciona SCM como parte de DevOps, pero rara vez explica *cómo* se implementa. (p. 7)
SCM independiente del método	El SCM es "agnóstico a la metodología" y debería poder adaptarse a DevOps. (p. 9)
Prácticas DevOps más relacionadas con SCM	El documento mapea CI/CD, automatización, monitoreo, testeo y planificación con actividades SCM. (p. 38–45)
Importancia del control de cambios	El control de cambios requiere CCB, impacto, revisión y aprobación. (p. 35–36)
Auditorías y baselines	Las auditorías verifican que la configuración coincide con la documentación y requisitos. (p. 21–22)
DevOps requiere automatización, CI/CD y pruebas continuas	El documento remarca que DevOps depende de automatizar pruebas, integraciones, despliegues y monitoreo. (p. 25–33)

Aunque la gestión de configuración de software ha alcanzado un nivel de madurez considerable para productos, estos enfoques resultan insuficientes para abordar la complejidad y diversidad de las SPL, lo que evidencia la necesidad de desarrollar una estrategia integral y adaptativa. Es así como, "existen muy pocos enfoques sobre la gestión de configuración en SPL, aunque la gestión de configuración está lo bastante madura para productos convencionales fabricados a proporción, estos enfoques resultan carentes para ofrecer una respuesta global. En otras palabras, aún es necesaria una aproximación integral a la administración de configuración en líneas de productos de software" (Espinel et al., 2024, p.1485).

Tabla 2. Aportes del documento

Tema	Aporte
La Gestión de Configuración (CM) es crítica en todo desarrollo	"Un elemento integral en cualquier actividad de desarrollo de software" (p. 1485).
Mayor complejidad en entornos distribuidos	El crecimiento de sistemas distribuidos y basados en componentes aumenta los desafíos de CM (p. 1485).
Necesidad de una visión unificada del CM	"Todavía no existe una visión común y coherente del CM en SPL" (p. 1484).
Trazabilidad y control de versiones	La trazabilidad y el versionamiento son "cruciales" para mantener consistencia e integridad (p. 1485).
Las actividades de CM mal cubiertas	Hay más investigación en *build management* y *version management*, y muy poco en *change management* y *23eléase management* (p. 1506).
Clasificación de actividades de CM	El esquema clasifica CM en: *Change Management, Version Management, Build Management, Release Management* (p. 1489).
CM en SPL sigue en fase inmadura	El área "todavía está en su fase formativa" (p. 1506).
Variabilidad, dependencias y consistencia son retos centrales	La dificultad de manejar dependencias y mantener consistencia (p. 1485).

A nivel operativo, Los SGC en entornos donde las herramientas integradas de extremo a extremo no han sido adecuadamente implementadas y las plataformas de gestión de proyectos no se utilizan de buena manera, el seguimiento y control de aspectos críticos se vuelven procesos lentos y propensos a errores. La falta de automatización y coordinación genera estructuras de trabajo con baja eficiencia, dificultando la adopción plena de prácticas DevOps. Se puede decir que, "la automatización no se utilizaba de manera intensiva y herramientas de gestión de proyectos como Jira no se usaban de forma efectiva, el seguimiento y control de la información relacionada con la publicación, versión, cambios, así como la entrega frecuente o incluso continua y el mantenimiento actualizado de los entornos, tomaban mucho tiempo, por lo que nuevamente surgía una estructura con baja eficiencia y viabilidad" (Bildirici et al., 2023, p. 2).

Tabla 3. Aportes del documento

Tema	Aporte
Problemas enfoque tradicional de CM	Identifica problemas en falta de agilidad, poca colaboración, ausencia de automatización y errores manuales (p. 1–2).
Integración DevOps con Ágil con CM mejora la eficiencia	Integrar DevOps y Agile con CM mejora tiempos, precisión, control y calidad (p. 1).
SCM tradicional es rígido y lento	Los modelos tradicionales dificultan reaccionar a cambios y necesidades del entorno (p. 2).
Git y Gitflow pilar del nuevo modelo	Gitflow facilita ramas aisladas, versiones claras y control del código (p. 6).

Infraestructura como Código y Configuración como Código	La nueva arquitectura usa IaC y CaC para controlar entornos y evitar drift (p. 6–7).
DevOps	Todas las actividades de CM coinciden con principios DevOps, como monitoreo, pruebas continuas y control (p. 7).

2.2. MARCO TEÓRICO

2.2.1. Teoría de Sistemas

La teoría de sistemas ofrece herramientas para enfrentar los desafíos actuales, "la aparición de nuevos tipos de sistemas complejos, como los sistemas habilitados por Inteligencia Artificial (IA), ha desafiado las prácticas tradicionales de la SE" (Shadab et al., 2023, p. 1), es necesario una base teórica sólida que permita una evolución metodológica y conceptual en la disciplina.

La diferencia entre sistemas cerrados y abiertos constituye una parte fundamental en la teoría de sistemas. Comprender el funcionamiento de los sistemas con su entorno permite encontrar soluciones más adaptables y eficientes, particularmente en contextos complejos como los sistemas inteligentes. Los teóricos clásicos definieron los sistemas abiertos como los "que tienen interacciones externas, con un límite entre lo interno y lo externo" (Shadab et al., 2023, p. 3), destacando la importancia de los flujos de información, materia y energía en la dinámica sistémica.

La teoría general de sistemas, formalizada por Ludwig Von Bertalanffy en 1968, constituye un marco fundamental para el estudio de sistemas naturales y artificiales. Esta constante evolución es vital para construir una teoría más robusta y congruente a través del tiempo.

Los sistemas complejos tienen la capacidad de interconectar elementos de diferentes diciplinas, en teoría de sistemas los problemas se analizan como un genérico para soluciones más acordes.

2.2.2. Sistematización de procesos

La sistematización de procesos es un enfoque metodológico que va a permitir ordenar, documentar y mejorar las actividades internas en una organización y así asegurar que estas se desarrollen de manera eficiente y alineada con los objetivos planteados con esto en mente una organización es entendida como un conjunto de actividades que generan valor para los usuarios externos. Según (Jiménez et al., 2024, p. 73), La esencia del sistema de gestión por procesos

formalizados es radicalmente entender y mejorar continuamente cómo se deben realizar las actividades dentro de la organización para lograr resultados.

2.2.3. Sistemas de Gestión

Dentro de los sistemas de gestión es importante tomar en cuenta que la administración junto con la organización va a permitir el control de recursos dentro de nuestra entidad o los procesos que se llevan a cabo dentro del desarrollo. Lo que va a permitir que nosotros cumplamos con una meta ya establecida es esta habilidad para cumplir con los objetivos y ajustarse a los cambios realizados es crucial por lo tanto considerar a la planificación como un eje central tomando en cuenta a la tecnología de la información para así preservar la estabilidad de nuestra compañía. Según (Lares et al., 2023, p. 260), la planeación estratégica de tecnologías de la información es fundamental para garantizar la sostenibilidad de las empresas en contextos cambiantes.

Es decir, "un SG es una combinación de subsistemas de una empresa en una estructura integral, para transformar entradas y facilitar el logro de sus objetivos" (Lagos et al., 2023, p. 5). La puesta en marcha de un SG brinda ventajas, dado que incrementa la eficacia y perfecciona la administración de recursos y riesgos.

2.2.4. Gestión del Cambio

Cuando en una empresa hay modificaciones se requiere entender desde dónde vienen y cómo se va a difundir jerárquicamente es por eso importante entender que el aprendizaje no debe guardarse. Es crucial entender que el aprendizaje no nace en la formación formal, sino que se origina en la vida cotidiana y en la interacción entre los integrantes de la organización. Esta visión facilita la creación de acciones más eficaces para ajustarse a un ambiente en permanente cambio. Como señala el texto, "la exploración de los procesos de cambio y aprendizaje ayudan a definir acciones en el campo de la gestión y abren nuevas perspectivas para el entendimiento del desarrollo de las organizaciones" (Rincón & Gómez, 2023, p. 42).

Las experiencias que se dan dentro de una comunidad son capaces de moldear la manera en la que una persona puede asimilar y experimentar la organización es así que afectará directamente la actitud ante el cambio cómo se ajustará a nuevos ámbitos. Mediante la cultura se expresan los valores,

creencias y costumbres que favorecen u obstaculizan los procesos de cambio. "la cultura organizacional es el lazo que une el cambio como factor externo y la transformación organizacional como factor interno" (Vera et al., 2023, p. 18).

Adoptar una visión sistémica y estratégica permite vincular las tendencias externas con los procesos internos, facilitando intervenciones efectivas y alineadas con los objetivos organizacionales. La cultura organizacional actúa como el eje articulador entre las dinámicas sociales y los procesos de cambio. Como lo afirma (Vera et al., 2023, p. 21), "una visión sistémica permite ubicar y gestionar la transformación organizacional a partir de tendencias de cambio del entorno", resaltando la necesidad de estudiar la cultura desde múltiples enfoques.

El modelo ADKAR tiene cinco hitos fundamentales para el éxito del cambio, iniciando por la conciencia, que implica que tanto el individuo como la organización conozcan la necesidad del cambio. Esta etapa permite identificar aspectos susceptibles de mejora y crea la disposición para avanzar en el proceso, también "es el primer hito en el cual el individuo (o la organización) toma conciencia." (Manzano & Ripoll, 2022, p. 54), el proceso de cambio comienza desde la comprensión y aceptación de la realidad actual.

2.2.5. Ingeniería de Software

Se reconoce a la ciencia de programas informáticos como una disciplina que combina saberes de informática, matemáticas y administración para la creación de productos y procesos de software. Dentro de este método se ve la planificación, la organización y la supervisión de los proyectos lo que permite la transformación en una mezcla de lo técnico y lo administrativo esta dualidad es la garantía para la calidad de software ya que "no es posible producir software de alta calidad sin la aplicación de principios y procesos de naturaleza gerencial" (Montilva & Barrios, 2021, p. 20).

Al pasar los años dentro de la ingeniería de software se puede ver reconocido como una parte fundamental a las ciencias de la informática por qué se ha demostrado que es esencial dentro de la educación académica de los expertos en un principio se incorporó en los currículos como una disciplina destinada a retos de desarrollo y la conservación como se indica en el plan de estudios sugerido por entidades internacionales, "la enseñanza de la ingeniería del

software es una actividad más de los programas de estudio en ciencias de la computación" (Montilva & Barrios, 2021, p. 25), Se evidencia su relación con la computación y su relevancia en la formación técnica universitaria.

La implementación de los principios Lean facilita la identificación y eliminación de los siete residuos. Estos desechos abarcan trabajos no finalizados y funciones superfluas hasta demoras y fallos que impactan de manera negativa en el proceso de desarrollo, "para el éxito de la metodología es fundamental la identificación de las características de los desechos" (Gordon & Delgado, 2023, p. 124), Solo a través de su identificación es posible poner en marcha medidas correctivas que potencien la productividad y la satisfacción del cliente.

Si vamos al ámbito de la ingeniería de software se puede decir que el progreso tecnológico se ve íntimamente ligado ya que su uso comprende desde los sistemas de información hasta las plataformas de ventas online esto requiere el dominio de campos como la arquitectura, el diseño, el modelado y la garantía de la calidad del software. De hecho, "en las industrias y en cualquier tipo de organización, la ingeniería de software y sistemas de información son transversales en las necesidades de todo tipo de proyectos" (Luján, 2024, p. 7).

2.2.6. Metodología en Software

Cuando hablamos del enfoque que tiene el software se puede decir que es un procedimiento ordenado y eficaz así que para una organización se facilita la implementación y supervisión de cada fase de un proyecto. La metodología es una investigación que guía el enfoque en el tratamiento de problemas y soluciones en la ingeniería de software.

Si nos centramos en metodología en software permite que la calidad técnica sea garantizada cuando hablamos de un producto se consideran los elementos como la rapidez, protección, y utilidad. En los 50 algunos desarrolladores se centraban en la codificación e ignoraban la interpretación de las necesidades del usuario lo cual resultaba en productos de baja calidad.

Se puede basar en modelos de ciclo de vida que es estructuran bajo el proceso en etapas para garantizar un producto con alta calidad esto se conoce como un enfoque metodológico ya que tiene etapas que van desde los requerimientos hasta la conservación del producto. La implementación de una

metodología apropiada promueve la claridad y exactitud en cada fase del ciclo, optimizando la comunicación y el triunfo del proyecto.

Cuando se incluyen técnicas ágiles en el momento de creación de software se utiliza una nueva cultura instruccional qué va de acuerdo con los fundamentos de una comunicación eficaz dentro del equipo. Según el análisis realizado, "las estrategias efectivas incluyen la educación y capacitación sobre los beneficios de las metodologías ágiles" (Chacha et al., 2024, p. 14), lo que permite vencer retos. Estos elementos facilitan una adopción progresiva y exitosa que incremente la eficiencia, la calidad del producto y la satisfacción del cliente.

Tabla 4. Metodología tradicional y la metodología ágil

Características	Metodología Tradicional	Metodología Ágil
Enfoque	Estructura de fases rígidas	Incremental y flexible
Documentación	Extensa y formalizada	Suficiente y adaptada al equipo
Comunicación	Formal, jerárquica	No formal, abierta y continua dentro del equipo
Ventajas	Roles, responsabilidades y entregables definidos	Satisfacción del cliente y rapidez
Desventajas	Poco flexible, lento para adaptarse a cambios	Requiere equipo maduro y compromiso cultural

2.2.7. Configuración de Software

La configuración de software permite mantener el control y la integridad de los productos durante su desarrollo. Se gestionan de forma sistemática los cambios, se establecen líneas base y se asegura la trazabilidad de cada componente del sistema. Se define como, "la Gestión de la Configuración de Software (GCS) […] proporciona métodos y herramientas para identificar y controlar el software a lo largo de su desarrollo y uso" (Villacres, 2023, p. 8). Con ello se garantiza la calidad del producto final y la coherencia entre sus versiones.

También respalda la gestión de proyectos y la garantía de calidad. Logra controlar la evolución del producto, asegurando su integridad a lo largo del tiempo y facilitando la trazabilidad de los cambios. "La administración de la configuración de software es un procedimiento dentro del ciclo de vida del software que proporciona apoyo para la administración de proyectos, las tareas de desarrollo y mantenimiento." (Villacres, 2023, p. 6). Ofrece ventajas a los equipos de desarrollo y a los usuarios finales, asegurando que el producto satisface los requerimientos detallados.

Se considera que la gestión de configuración es un soporte para la administración de los proyectos beneficiando también a los clientes y usuarios finales. La gestión de configuración permite "controlar los cambios en esas características, registrar e informar sobre el procesamiento de cambios y el estado de implementación" (Villacres, 2023, p. 7). También incluye la administración de los componentes y sus versiones para evitar conflictos entre los cambios realizados por diferentes desarrolladores. Por ello la gestión de versiones organiza y controla las líneas de código y las líneas base del sistema. "La gestión de versiones es el proceso de administrar líneas de código y líneas base" (Cubero, 2020, p. 7).

La gestión de configuración de software se basa en estándares reconocidos como el IEEE 828-2005, que establecen directrices para la administración adecuada de la configuración. Para esto, los sistemas de gestión de versiones utilizan un repositorio público junto con espacios de trabajo privados, donde los desarrolladores pueden trabajar de forma aislada antes de integrar sus cambios. Así se muestra que, "los sistemas de gestión de versiones utilizan repositorio público y un espacio de trabajo privado" (Cubero, 2020, p. 4), asegurando un control eficiente y seguro durante el desarrollo colaborativo.

En un plan de GCS, es posible fusionar herramientas de software y procedimientos manuales, fusionando diversos componentes esenciales como la estructura de la biblioteca, la gestión de accesos, el monitoreo de documentos y la gestión de código. Normalmente, estas herramientas son específicas para GCS. "Se puede combinar las herramientas de software y procedimientos manuales en el plan de GCS" (Villacres, 2023, p. 10), lo que permite una gestión integral y adaptada a las necesidades del proyecto.

2.2.8. IEEE 828

El estándar IEEE 828 establece que hay requisitos formales para la gestión de configuración dentro de los sistemas y el software tiene como propósito definir diferentes lineamientos que garanticen que los elementos de la configuración sean gestionados de forma sistemática y controlada y así mismo sea trazable durante todo el ciclo de vida de un sistema la norma que están haciendo nombrada enfatiza que es necesario mantener políticas procedimientos roles y responsabilidades así como auditorías y procesos de cambios.

Da un marco estructurado para que las organizaciones obtengan ayuda a la hora de cumplir requerimientos de seguridad y de estandarización en especial cuando se trata de manejar entornos críticos donde la pérdida de trazabilidad o cambios que no están controlados pueden generar riesgos y vulnerabilidades tal como ha señalado la gestión de configuración es obligatoria para garantizar que los sistemas mantengan "configuraciones base documentadas, control de cambios, análisis de impacto, auditorías y restricciones de acceso para asegurar la estabilidad y seguridad del entorno tecnológico" (Epa, 2022, pp. 1, 3).

Por ello se dice que la norma se convierte en un estándar como referente para poder implementar un sistema de gestión de configuración de software que sea robusto asegurado que los componentes del software se mantengan constantes y alineados con requisitos.

2.2.9. Componentes de sistema de gestión de configuración de software

Un sistema de gestión de configuración de software tiene una estructura a partir de algunos componentes esenciales que van a permitir identificar y controlar a la misma vez que registrar y supervisar los elementos que conforman un sistema de software dentro de la literatura reciente hay que resaltar que la gestión de configuración moderna se basa en bajo 3 pilares que son la identificación de ítems de configuración el control de cambios y la contabilidad del estado. Estos tres componentes constituyen "elementos fundamentales para que la gestión de configuración pueda garantizar la integridad y el control de los activos tecnológicos"(Oluwatoyin et al., 2023, p. 145).

El componente llamado identificación consiste en que se debe clasificar y definir de manera única cada uno de los ítems ligados a la configuración CI esto incluye el código fuente la documentación las librerías la infraestructura y cualquier otro artefacto perteneciente o relevante para el sistema la identificación precisa va a permitir establecer líneas de base y facilita la trazabilidad durante el ciclo de vida del sol el segundo componente es el control de los cambios este es el encargado de gestionar y evaluar cualquier modificación que se hace dentro de un sistema a este proceso se debe involucrar un análisis de impacto y una revisión formal en la documentación de decisiones y aplicación controlada de cambios. Explican que este control busca

"evitar modificaciones no autorizadas o inconsistentes que puedan comprometer la integridad del sistema" (Oluwatoyin et al., 2023, p. 145).

El registro de estado es el tercer componente que hace referencia al que al mantenimiento de historiales versiones y estados actuales de la configuración este registro permite realizar auditorías facilitar la resolución de incidentes y asegurar que existe transparencia en esta literatura también se puede añadir un cuarto elemento transversal que es la verificación o la auditoría de configuración qué hace la evaluación de las consistencia entre la línea de base aprobada y el estado real del sistema estas auditorías permiten detectar errores y asegurar el cumplimiento de los procedimientos de forma adecuada en conjunto todos los componentes que conforman un sistema de gestión de configuración de software que sea robusto y capaz de mantener la estabilidad de un software toma decisiones dentro del desarrollo y la operación de sistemas complejos.

2.2.10. Versionado semántico

El personamiento semántico es un esquema de numeración de versiones que es ampliamente adoptado en la industria del software porque permite comunicar de una manera clara el impacto qué hay en los cambios introducidos en tu en un sistema este esquema estructura tiene una estructura donde cada número refleja un nivel distinto de compatibilidad por ejemplo incrementos en el mayor indican cambios Incompatibles en el menor funcionalidades nuevas y en el path correcciones que no están siendo alteradas lo importancia del versionamiento semántico está en el que permite a los equipos anticipar los efectos de actualizar una dependencia permitiendo la facilidad de la estabilidad la trazabilidad y la planificación del ciclo de vida del software en el documento se resalta que SemVer "se ha convertido en una práctica ampliamente recomendada para comunicar de forma explícita la magnitud y compatibilidad de los cambios introducidos en una librería o componente"(Ochoa et al., 2021, p. 2).

Este análisis en el estudio puede evidenciar que el versionado semántico además de cumplir un rol descriptivo puede también cumplir un rol predictivo ya que los desarrolladores pueden depender de su consistencia para decidir si una actualización puede romper versiones que influyen directamente en la

confianza de un ecosistema de dependencias destacando que "los consumidores de paquetes ajustan sus restricciones de versión con base en la expectativa de estabilidad que promete SemVer" (Ochoa et al., 2021, p. 3).

Es por ello que un sistema de gestión de configuración de software el versiona miento semántico funciona como un mecanismo central para que se pueda controlar la evolución de un producto documentar los cambios y garantizar la compatibilidad entre sus diferentes componentes el uso sistemático contribuye a la tranquilidad y a la calidad del software especialmente en entornos DevOps donde la frecuencia de cambios es alta.

2.2.11. Git

El programa conocido como Git es un sistema que permite controlar versiones para gestionar de manera eficiente la evolución de un código fuente y asimismo coordinar el trabajo entre múltiples desarrolladores el diseño de este está orientado para ofrecer una gran velocidad integridad de datos y soporte para los flujos de trabajo distribuidos y no lineales las características que lo convierten en una herramienta más utilizada en proyectos de desarrollo moderno. Según (Shchirov, 2022, p. 348), Git es "un sistema de control de versiones libre y de código abierto, diseñado para manejar desde proyectos pequeños hasta muy grandes con rapidez y eficiencia".

La principal fortaleza de este programa es que permite trabajar mediante diferentes ramas lo que hace que el desarrollar nuevas funcionalidades corregir errores o experimentar sin afectar el código principal sea mejor el documento destaca que las ramas ofrecen "un entorno aislado para cada cambio del código, garantizando que la rama principal mantenga siempre una versión estable y de calidad de producción" (Shchirov, 2022, p. 348). Estas características facilitan en de enorme manera la implementación de metodologías ágiles y debo donde se requiere que las interacciones sean rápidas y controladas. Esta característica facilita enormemente la implementación de metodologías ágiles y DevOps, donde se requieren iteraciones rápidas y controladas.

Otro componente fundamental dentro de esta arquitectura distribuida es que a diferencia de los sistemas centralizados cada desarrollador puede tener una copia completa de los repositorios con el historial lo que permite trabajar sin

conexión y realizar commits locales y así mejorar la resiliencia a fallos cómo puede afirmar el autor, "cada desarrollador obtiene su repositorio local con un historial completo de commits, lo que hace que Git sea rápido y elimina bloqueos comunes en sistemas centralizados" (Shchirov, 2022, p. 349).

También incorpora diferentes mecanismos que fortalecen y fomentan la colaboración como los pull request que facilitan revisiones de código las discusiones y fusiones seguras esto permite que los equipos de gran magnitud puedan mantener los estándares de calidad y reduzcan la probabilidad de errores cómo dice en el texto los pull requests "permiten a los desarrolladores iniciar discusiones sobre sus cambios antes de integrarlos al repositorio principal" (Shchirov, 2022, p. 349).

También Git puede ser compatible con entornos de integración continua y entrega continua haciendo que sea fácil automatizar los despliegues y las pruebas cada vez que se realiza un cambio esta integración acelera los ciclos de entrega y reduce la cantidad de errores dentro de la producción contribuyendo a la eficiencia del proceso de desarrollo en el documento se enfatiza que Git "se integra exitosamente con entornos de integración y entrega continua, permitiendo automatizar despliegues desde ramas específicas" (Shchirov, 2022, p. 349).

2.2.12. Auditorias y baselines

Las auditorías dentro de un sistema de gestión de configuración de software es un mecanismo fundamental para que se garantice la integridad consistencia y la trazabilidad del sistema mismo las líneas de base representan también conjuntos de elementos de configuración aprobados normalmente en un momento específico en conjunto con los cambios autorizados conforman la configuración actual del sistema. Una línea base es exactamente información de configuración que se designa formalmente en un momento específico durante todo el ciclo de vida de un desarrollo de producto estas líneas base son puntos de referencia que permiten controlar versiones y así no tener pérdidas de integridad y mantener un registro claro de la evolución de un sistema.

Las auditorías de configuración son actividades que se llevan de manera sistemática porque permiten verificar que los elementos de la configuración y las líneas base estén alineados con los requisitos definidos y con la

documentación que se tiene aprobada una auditoría permite que cada guiten o conjunto de ítems cumplan los estándares especificados estas auditorías asegurarán la conformidad técnica y revisarán la precisión del inventario en la base de datos de la configuración la eficacia de los procesos de cambio.

La Configuration Management Procedure, la verificación y auditoría de configuración es un proceso obligatorio para validar "la integridad de los procesos, sistemas, ítems y baselines bajo control de la gestión de configuración" (Epa, 2022, p. 4).

Cuando se realizan revisiones periódicas es más fácil detectar desviaciones y corregir inconsistencias para asegurar que todo el cambio haya sido aprobado antes de la inclusión en el entorno productivo el documento también enfatiza que todas las auditorías deben registrarse generando así un historial completo de acciones hallazgos y actividades de cierre esto garantizará la trazabilidad y proporciona evidencia formal para un análisis posterior o evaluaciones externas la obligación es mantener estos registros para fortalecer la transparencia y la responsabilidad del ciclo de vida del software.

2.2.13. Rastreo de cambios

El rastreo de cambios es también conocido como la trazabilidad qué es la capacidad de seguir y documentar lo interacciona entre los artefactos de un software a lo largo de todo el ciclo de vida mientras se desarrollan esta trazabilidad va a permitir conocer un cambio en cada uno de los requisitos del módulo componentes o el documento y cómo va a impactar en otras partes del sistema garantizando así que hay control transparencia y una buena gestión del proceso de evolución del software la trazabilidad se define como "la capacidad de describir y seguir la vida de un requisito en dirección hacia adelante y hacia atrás, desde su origen hasta su desarrollo, especificación, implementación, despliegue y mantenimiento"(Gul et al., 2021, p. 347).

Con esta descripción en mente se puede decir que la trazabilidad no es más que un mecanismo de registro sino que también un proceso de conexión para las piezas que conforman un sistema permitiendo así entender que las dependencias y riesgos y efectos colaterales están unidas este rastreo de cambios es esencial para realizar análisis de impacto los cuales consisten en determinar qué partes del sistema se verán afectadas cuando se modifica

algún requerimiento o componente se explica que la falta de trazabilidad adecuada conduce a "estimaciones incorrectas, reprocesos, retrasos e incluso fallos de proyecto"(Gul et al., 2021, p. 347).

2.2.14. Transformación Digital

Hoy en día, las instituciones se encuentran con retos gigantes debido a las aceleradas modificaciones tecnológicas. La adaptación implica modificar la estructura, los procedimientos y la cultura de la entidad. Algunas entidades actuales consideran que la transformación digital es una prioridad y no un obstáculo que se deba tomar como una tendencia. Dicen que, "requiere una comprensión profunda de la compleja interacción de la tecnología, la cultura, el liderazgo, la estrategia y otros dominios. Deben atravesarse aguas inexploradas de innovación y disrupción." (Omol, 2024, p. 1).

La era digital no es solo algo efímero, sino que en realidad se puede considerar que es algo crucial en nuestra modernidad porque potencia la innovación y la competitividad entre los negocios. Sin irnos más allá podemos considerar que es una nueva etapa en la evolución de la tecnología que va a permitir tener oportunidades para mejorar procesos y mejorar la experiencia de usuario y así crear valor en el mercado. "El viaje de la transformación digital se ha entrelazado cada vez más en las corrientes más agresivas de la estrategia organizacional e impacta todos los aspectos del negocio contemporáneo" (Omol, 2024, p. 1).

El internet de las cosas nos ha brindado un progreso rápido en estos últimos años lo que ha permitido que las redes sociales transformen completamente la comunicación entre las comunidades y las personas es por esto que los recursos como la inteligencia artificial y la realidad aumentada cuentan con un impacto importante dentro de la educación y el tiempo libre esto evidencia que la transformación digital no es una técnica sino también un tema social y cultural. Según Paul et al., p. (2024, p. 2), estos campos son esenciales para comprender la dinámica de la transformación digital y las potenciales rutas de investigación que este sector deberá investigar.

Cuando tratamos el tema de la digitalización lleva a la modificación de operaciones y procesos porque se basa en la incorporación de las tecnologías de la información lo que crea nuevos modelos de negocio, potencia el

rendimiento y fomenta la innovación de forma constante. "Además, la TD no está vinculada únicamente a la tecnología, sino también a una mejora en el modelo de negocio, la colaboración y la cultura." (Reis & Melão, 2023, p. 3)

2.2.15. Tecnologías de la Información

La información y comunicación tienen herramientas pertenecientes a un grupo que simplifican la administración y el almacenamiento con el intercambio y la transmisión de datos estas tecnologías van a revolucionar la administración y la comunicación en cada uno de los entornos aplicadas. Entonces, "las TIC están presentes en todos los niveles de la sociedad actual, desde las más grandes corporaciones multinacionales hasta particulares" (Olarte et al., 2023, p. 2).

En centros de enseñanza secundaria, las TIC promueven experiencias educativas para los alumnos. Adicionalmente, su utilización aumentó a causa de la pandemia del COVID-19, que impulsó la incorporación de estas herramientas en los procesos de enseñanza. No obstante, aún persisten desafíos para conseguir una implementación adecuada, lo que requiere un trabajo conjunto de profesores e instituciones. Como se subraya en el estudio, "su importancia ha crecido considerablemente a raíz de la pandemia del COVID-19." (Peralta et al., 2023, p. 1)

El uso de las Tecnologías de la Información y la Comunicación en la educación ha provocado una transformación profunda en los procesos de enseñanza-aprendizaje, "el uso de TIC en educación contribuye de manera didáctica e innovadora, al cambio y transformación, desde el sumo compromiso de capacitación del profesorado, así mismo, en la formación inclusiva y tecnológica del estudiantado" (Mollo et al., 2023, p. 4). La necesidad de una preparación constante permite enfrentar los retos de una educación virtual de manera eficaz.

2.2.16. Metodologías Emergentes

Siempre existe avances dentro del desarrollo de software y las técnicas no son la excepción es por eso por lo que se aprecia el ajuste y rapidez como una habilidad cuando se existe demandas muy variantes dentro de un proyecto. La reingeniería y el enfoque de beneficio mutuo buscan impulsar el desarrollo y así asegurar un buen desempeño, fundamentándose en estudios robustos y bases teóricas que unen todas las etapas del ciclo de vida del software. Es así que,

"cada una de estas metodologías busca optimizar, acelerar y garantizar un funcionamiento completamente correcto, dependiendo del ámbito y la aplicación específica" (Villa et al., 2023, p. 3).

Las metodologías emergentes buscan el ajuste del ambiente que están en constante cambio y busca satisfacer también las necesidades de un usuario y así fomentar un entorno laboral que sea adecuado. Se podría decir que "las metodologías ágiles se enfocan en el buen ambiente de trabajo y apoyo laboral entre compañeros, ya que deducen que con este tipo de método los trabajadores pueden rendir mejor en sus tareas y tener un resultado final óptimo" (Villa et al., 2023, p. 8).

Aunque es considerado más una cultura podemos decir que DevOps podría llegar a convertirse en una metodología de trabajo que se transforma en el modo en que los equipos colaboran y entregan aplicaciones ya que fomentan la integración de despliegues rápidos y fiables es así que este nuevo método produce una comunicación eficaz y entrega ágil con buena calidad. En este sentido, "DevOps fomenta una colaboración cercana y una comunicación efectiva entre los equipos, lo que facilita una entrega continua más ágil, eficiente y confiable de software con altos estándares de calidad" (Alexandra et al., 2024, p. 4).

Tabla 5. Comparación metodologías anteriores contra metodología emergente

Características	**Metodologías Anteriores (Tradicional y ágil)**	**Metodologías emergentes**
Adaptabilidad al cambio	Tradicionales: baja. Ágiles: alta, con ciclos cortos.	Muy alta, cambios constantes incluso en etapas avanzadas.
Calidad del producto	Tradicionales: asegurada al final. Ágiles: garantizada durante cada iteración.	Calidad continua mediante pruebas automáticas, integración y retroalimentación constante.
Relación con el cliente	Tradicionales: contacto limitado. Ágiles: comunicación frecuente y retroalimentación activa.	Involucramiento constante, con monitoreo de necesidades y mejora del producto en tiempo real.
Uso de tecnología	Tradicionales: herramientas básicas. Ágiles: herramientas de gestión.	Herramientas avanzadas de automatización, integración, despliegue y monitoreo.
Objetivo principal	Tradicionales: cumplimiento de requerimientos definidos. Ágiles: satisfacción del usuario y entrega funcional.	Optimización de procesos, velocidad, calidad continua y entrega de valor constante.

2.2.17. DevOps

DevOps por sí mismo representa un gran cambio en la cultura de las organizaciones ya que este promueve la cooperación entre los equipos y la automatización de los diferentes procesos que están en la distribución constante del software este procedimiento puede llevar varios retos ya que aún se necesita orientación precisa sobre cómo implementar las buenas prácticas, algunas compañías incurren en gastos extras al usar herramientas incorrectas debido a una deficiente guía resulta importante estandarizar las prácticas sugeridas a través de modelos qué funcionen como fundamento para su rendimiento y uso "DevOps es un esfuerzo colaborativo y multidisciplinario dentro de una organización para automatizar la entrega continua de nuevas versiones de software, garantizando su corrección y fiabilidad." (Marques & Correia, 2023, p. 1).

Dentro de algunas instituciones ya se ha visto este nuevo método como una transformación en cómo se trabaja y cómo se crea ya que se pone en práctica la preservación del software y promueve la cooperación continua entre los diferentes grupos que trabajan en el desarrollo y las operaciones es así que no sólo se conlleva a modificaciones técnicas, sino que se transforma la cultura dentro de las organizaciones para finalizar con el incremento de la eficacia y la calidad del producto entregado. "DevOps es una cultura, un proceso y una tecnología que promueve una colaboración estrecha entre el desarrollo de software y las operaciones e integra diversas disciplinas" (Alejos et al., 2024, p. 3), lo cual resalta su naturaleza multidimensional y su impacto transversal dentro de las organizaciones.

Si se utiliza DevOps dentro de las técnicas ágiles se puede conseguir una gran mejora en la cooperación entre los equipos durante todo el proceso de ciclo de vida del software cuando no se unifican provocan que las organizaciones encuentren restricciones debido a la ausencia de comunicación de equipos. Por esto, "DevOps pretende involucrar al área de desarrollo de software en la implementación y operación del software desarrollado" (Alejos et al., 2024, p. 2), lo que enfatiza su función como enlace entre disciplinas que tradicionalmente funcionaban de manera distinta, y enfatiza la importancia de potenciar las habilidades del equipo para alcanzar un auténtico cambio ágil.

La posibilidad de una fusión entre las técnicas ágiles y DevOps puede transformar el desarrollo de los sistemas de información ya que se va a posibilitar la entrega más veloz y se podrá enfocar en el usuario si tenemos esta integración se promoverá también la cooperación más rápida entre los diferentes programadores recreando así la optimización de la calidad del software. Así las organizaciones promueven un cambio cultural orientado a la mejora continua. "DevOps reduce la brecha entre los desarrolladores, las operaciones y el usuario final" (Alejos et al., 2024, p. 13).

La literatura actual revela una poca investigación sobre cómo las prácticas DevOps influyen en la mantenibilidad del producto, lo que representa una oportunidad importante para futuros estudios. Evaluar y adaptar métricas de mantenibilidad a estos entornos es crucial para garantizar productos sostenibles a lo largo de los años. "La mantenibilidad del software está estrechamente relacionada con varias prácticas de DevOps" (García et al., 2024, p. 12). Se debe analizar cómo este enfoque puede mejorar la calidad estructural del software.

Tabla 6. Ciclo de vida DevOps

Fase	Descripción	Herramientas
Planificación	Se definen los requerimientos del producto, se planifican las versiones y tareas del equipo.	Trello, Azure Boards
Desarrollo	Codificación del software según los requisitos establecidos.	Github, Gitlab
CI	El código se integra frecuentemente en un repositorio común con pruebas automáticas.	Jenkins, CicleCI
Testing	Se realizan pruebas automatizadas para validar la calidad y funcionalidad del software.	Selenium, TestNG
CD	El software se empaqueta y prepara para ser desplegado automáticamente en producción.	GitLab CI/CD
Implementación	Se despliega el software en el entorno de producción de forma controlada y automatizada.	Ansible, Kurbenetes
Operación	Se monitorea el funcionamiento del software, el rendimiento del sistema y los errores.	Nagios, Datadog
Monitoreo y retroalimentación	Se recopila retroalimentación del usuario y del sistema para mejorar el producto en futuras iteraciones.	New Relic

La estandarización de prácticas DevOps pueden apoyarse en modelos visuales para describir el flujo integral de procesos según los estándares internacionales

por eso la siguiente figura representa una síntesis basada en los lineamientos de la norma ISO/IEC 12207.

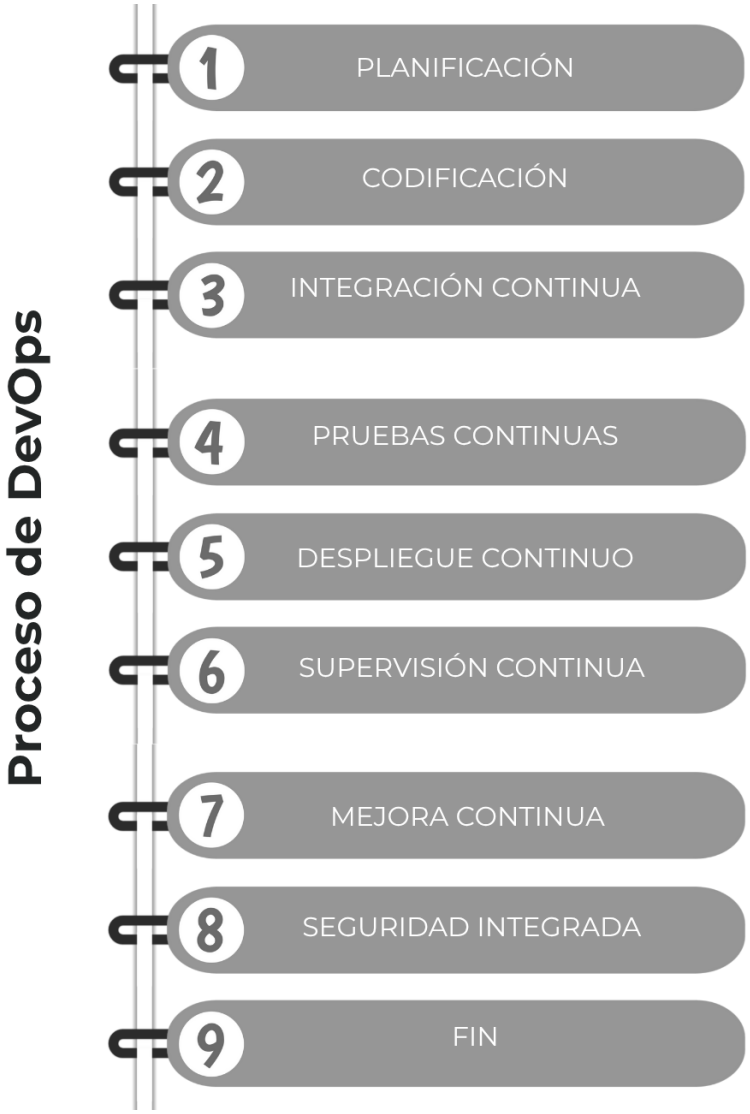


Figura 1. Proceso de DevOps

2.2.18. Cultura DevOps CALMS

DevOps es una cultura que tiene como pilar fundamental la adopción efectiva de los enfoques de desarrollo de software DevOps surge como una solución a la necesidad que ha existido de integrar de forma colaborativa los equipos de desarrollo y los de operaciones para reducir discordancia facilitar los ciclos de entrega y garantizar la estabilidad de un sistema dentro de la cultura de DevOps existe un modelo CALMS que sigue de las siglas en el documento se destaca que "un ambiente de cultura organizacional basado en la colaboración y comunicación entre los equipos de desarrollo y operaciones"(Muñoz et al., 2020, p. 82).

Este modelo cultural está definido a que se bajen las barreras entre los equipos pero es lograr promover objetivos compartidos y tener responsabilidades conjuntas dentro de la documentación se hace énfasis en que "el éxito de DevOps depende en gran medida de la colaboración entre los desarrolladores y el personal de operaciones de TI" (Muñoz et al., 2020, p. 84).

El automatizar es esencial para que se bajen el número de errores manuales y así acelerar el ciclo de entrega que permite que los equipos se den enfoque en el mayor valor que tiene una institución. "DevOps depende en gran medida de la automatización […] desde la integración del código hasta la implementación" (Muñoz et al., 2020, p. 85).

Este principio lo que busca es minimizar que haya desperdicios y facilitar los procesos ya que las actividades deben mantenerse en lo más mínimo pero que sigan siendo útiles así evitando que exista la documentación excesiva para las funcionalidades que aún no son necesarias y ciclos largos de desarrollo.

El modelo planteado en la documentación dice que se puede monitorear los indicadores de los rendimientos la velocidad la calidad y la experiencia de usuario también se señala que hay diferentes métricas que son claves como "frecuencia de liberaciones, tasa de éxito de implementación, tasas de error de aplicación y tiempo medio de resolución de problemas"(Muñoz et al., 2020, p. 87).

Compartir conocimiento y promover transparencia es vital para una cultura DevOps. Dice en el texto que el "compartir es esencial para mejorar la

comunicación y hacer que las personas trabajen juntas"(Muñoz et al., 2020, p. 87).

2.2.19. CI/CD

La integración continua o en inglés titulado continuous integration y la entrega continúa o continuous delivery es una práctica fundamental dentro de esta nueva cultura planteada llamada DevOps qué es orientada a la sistematización aceleramiento y garantizarían la calidad de un proceso de desarrollo de software y despliegue del mismo el objetivo central es permitir que los equipos tengan cambios de forma frecuente se ejecuten pruebas automatizadas y se desplieguen versiones confiables sin interrupciones estas prácticas DevOps se puede decir que son pilares para lograr que los ciclos de entrega sean cortos y los tiempos de espera de respuesta sean rápidos logrando así un producto de software más estable dentro del documento se enfatiza que las prácticas "han transformado fundamentalmente el desarrollo moderno de software, permitiendo a las organizaciones alcanzar niveles sin precedentes de eficiencia y confiabilidad en la entrega"(Damarapati, 2025, p. 1).

Estas 2 prácticas se relacionan con las métricas críticas del desempeño de la cultura según lo que dice el artículo las organizaciones con prácticas maduras CI/CD logran "frecuencias de despliegue múltiples por día y tiempos de entrega de cambios inferiores a una hora" (Damarapati, 2025, p. 1)

Se señala que organizaciones con implementaciones sólidas de CI/CD reportan "tiempos medios de recuperación (MTTR) menores a una hora y tasas de fallos inferiores al 15%" (Damarapati, 2025, p. 2).

2.2.20. Infraestructura como código

La infraestructura como código es una práctica que va a permitir gestionar y configurar la infraestructura tecnológica mediante diferentes archivos que declaren el código en lugar de procedimientos manuales este paradigma transforma el modo de administración de servidores redes y servicios haciendo posible la automatización y el control de versiones sobre los componentes de la infraestructura al igual que se hace con los softwares.

Según Morris, p. (2021, p. 3), IaC consiste en "gestionar y automatizar la infraestructura mediante prácticas de ingeniería de software, utilizando archivos de configuración que describen el estado deseado de los sistemas"

La infraestructura tradicional sí hace gestionando manualmente y es propensa a cometer errores y que las configuraciones sean inconsistentes así como también que los procesos sean poco escalables en contraste la infraestructura como código Evita la configuración por Clic y permite que los diferentes entornos sean replicables auditables y automatizados por ello en la documentación se señala que "eliminar la configuración manual que conduce a inconsistencias y a la aparición del 'drift' de configuración entre entornos"(Morris, 2021, p. 12).

2.2.21. Pipeline DevOps

Dentro de esta práctica conocida como DevOps el pipeline es una cadena de procesos automatizados que permiten integrar probar y analizar el software de manera continua y que sea confiable se constituye como el corazón operativo del DevOps ya que puede tener todas las etapas de un ciclo de entrega garantizando que haya estabilidad rapidez y que sea repetible en cada liberación de código la principal función de un pipeline es que se transforme los cambios de códigos en despliegues funcionales mediante la automatización completa y así reducir la intervención manual y los errores humanos se puede decir que "automatiza completamente el ciclo de vida del software utilizando herramientas estándar de la industria como Jenkins, Docker y Kubernetes" (Chauhan et al., 2025, p. 11).

Los métodos manuales de despliegue representan un problema significativo, produciendo "cuellos de botella, errores humanos e inconsistencias entre entornos"(Chauhan et al., 2025, p. 12).

2.2.22. Calidad del software

Cuando se habla de calidad en el software se hace referencia a que el conjunto de atributos que determinan el grado de que un sistema cumpla con los requisitos funcionales o no funcionales incluyendo la confiabilidad la seguridad y la eficiencia que se tiene también la usabilidad la calidad no depende únicamente en el código sino también en los procesos automatizados que intervienen en la construcción del mismo la integración pruebas y

despliegues es por esto que la calidad se ve influenciada por las prácticas CI y se dé en el documento se analiza que al ser crítico dentro del ciclo de vida del software se puede convertir en un punto vulnerable que afecta de forma directa a la calidad. Según Pan et al., p. (2024, p. 1), "los CI/CD pipelines sufren de código malicioso y vulnerabilidades severas, y las personas no han sido plenamente conscientes de sus superficies de ataque y los impactos correspondientes".

Mas del 97% de los repositorios analizados utilizan scripts CI/CD con versiones desactualizadas, lo que introduce riesgos significativos. En palabras de los autores: "83.56% de los usos de scripts referencian versiones obsoletas, mientras que 97.86% de los repositorios usan al menos una versión antigua" (Pan et al., 2024, p. 8).

III. METODOLOGÍA

3.1. ENFOQUE METODOLÓGICO

3.1.1. Enfoque

Se busca una combinación dentro del enfoque para poder unir dos técnicas que son la cuantitativa y la cualitativa Y así poder tener una visión completa de lo que es un sistema de gestión de configuración aplicando DevOps. Como donde se indica que "SCM es agnóstico a la metodología y debería poder adaptarse a DevOps"(Hochbergs & Nilsson, 2020, p. 9).

En el enfoque cuantitativo es necesario la obtención de los datos numéricos a través de un instrumento en este caso las encuestas aplicadas al personal de la unidad de tics, con el fin de conocer si los indicadores como el control de versiones, la trazabilidad de cambios y la gestión de actualizaciones están siendo afectados. considerando que "DevOps y sus herramientas permiten reducir tiempos de ciclo y aumentar calidad mediante control de versiones y trazabilidad"(Cui, 2024, p. 2).

En el marco del componente cualitativo es necesario comprender qué información es complementaria a nuestra investigación por lo cual se ha escogido el instrumento de checklist basado en una norma ISO/IEC que nos permitirá recolectar información verídica para conocer cómo se maneja la unidad de desarrollo y así identificar los problemas y las soluciones que se deben aplicar a cada uno de los procesos de configuración. Esto coincide con la idea de que la gestión de configuración "implica identificar, organizar y controlar cambios y modificaciones en el software"(Hochbergs & Nilsson, 2020, p. 2).

La integración de ambos enfoques favorece el contraste y la validación de los datos obtenidos, fortaleciendo la confiabilidad de los resultados y proporcionando una perspectiva más completa del contexto institucional. Se concluye que la combinación de métodos mejora la interpretación de procesos y "demostró mejoras en tiempo de compilación, versiones más precisas y control total del sistema"(Bildirici et al., 2023, p. 1).

3.1.2. Tipo de Investigación

3.1.2.1. Investigación Descriptiva

La investigación descriptiva es una investigación de tipo cuantitativo ya que nos va a permitir detallar las condiciones y a caracterizar las condiciones actuales en las que se pueden ver las versiones y los cambios dentro de los módulos desarrollados por la unidad de desarrollo dentro de la universidad UPEC el principal objetivo consiste en identificar qué necesidades deficiencias y oportunidades de mejora tienen cada uno de los procesos de configuración de software se pueden utilizar instrumentos para determinar esto.

3.1.2.2. Investigación no experimental

La investigación no experimental va a permitir se manipulen las variables dentro de lo establecido sin afectar completamente el área de estudio y se podrá analizar qué fenómenos están actuando dentro del contexto del área o la unidad esto va a permitir comprender la situación real de la unidad de desarrollo sin tener que manipular sus procesos garantizando así que la información sea verídica y las condiciones sean auténticas.

3.1.2.3. Investigación-Acción

La investigación–Acción, busca comprender y mejorar las prácticas actuales de gestión de configuración de software en el área a investigar. Este tipo de estudio deja analizar la situación existente, identificar problemas y proponer acciones de mejora que contribuyan al fortalecimiento de los procesos institucionales.

3.1.2.4. Investigación aplicada

La finalidad de la investigación aplicada es dar una solución práctica e implementar dentro del área de TICS, esta investigación utiliza conocimientos teóricos y metodológicos para intervenir en un problema específico y así generar mejoras en el entorno.

3.1.2.5. Investigación de campo

La investigación de campo se ve dentro de la recolección de datos en el entorno real dónde se realizan los procesos de software, esta investigación va a permitir obtener información mediante la observación encuestas entrevistas o la interacción con los actores involucrados dentro de la unidad.

3.1.2.6. Investigación documental

La investigación documental procura la revisión a conveniencia de fuentes teóricas, técnicas y las normativas que proporcionen un sustento teórico y metodológico al trabajo en sí, esta investigación va a permitir un análisis y sintetizar la información para hacer una comparación técnica y construir nuestra propuesta.

3.2. IDEA A DEFENDER

Una guía metodológica para un Sistema de Gestión de Configuración de Software permitirá la adopción de prácticas DevOps a la Universidad Politécnica Estatal del Carchi mejorando el control de versiones, integridad de proyectos y actualizaciones manuales en sus módulos virtuales.

3.3. DEFINICIÓN Y OPERACIONALIZACIÓN DE LAS VARIABLES

3.3.1. Definición de las variables

Variable independiente (VI). - Sistema de gestión de configuración de software

Variable dependiente (VD). – DevOps

3.3.2. Operacionalización de las variables

Tabla 7. Operacionalización de las variables

Variable	Definición	Dimensión	Indicadores	Técnica	Instrumento
VI	Conjunto de procedimientos, herramientas y políticas que permiten identificar, controlar y supervisar los componentes del software, asegurando la coherencia, trazabilidad y mantenimiento de las versiones a lo largo del ciclo de vida del proyecto (ISO/IEC 26563:2022).	Flujo de trabajo Git y ramas Integración y pruebas	Control de versiones, estructura de ramas, frecuencia de commits, revisiones de código, resolución de conflictos. Aplicación de integración continua (CI), automatización de compilaciones, validación de cambios, frecuencia de pruebas unitarias e integración.	Encuesta	Cuestionario validado
VD	"DevOps es un conjunto de prácticas y conceptos para realizar un cambio cultural en las organizaciones… proporcionando aplicaciones y servicios a gran velocidad… fundamental para alcanzar los objetivos del negocio" (Aranguren & Ruiz, 2023, p. 18)	Herramientas y mejoras	Uso de herramientas DevOps (Git, Jenkins, Docker, etc.), nivel de automatización de despliegues, tiempos de actualización, trazabilidad de cambios y comunicación entre equipos.	Checklist	Lista de verificación ISO/IEC 26563:2022

3.4. MÉTODOS UTILIZADOS

3.4.1. Métodos

3.4.1.1. Método PHVA

La metodología de mejora continua conocida como PHVA o planificar, hacer, verificar y actuar cuyo reconociendo es respaldado bajo norma ISO/IEC 29110, este método permite gestionar y mejorar los procesos en organizaciones.

Tabla 8. Aplicación de PHVA

Fase	Implicación	Aplicación
Planificar	Identificar problema, definición de objetivos y selección de instrumentos	Diagnóstico de la gestión y configuración actual, diseño de encuesta, entrevista y checklist ISO/IEC
Hacer	Ejecución de actividades y comparación con estándares	Aplicar encuestas y checklist al equipo de desarrollo
Verificar	Analizar los datos y comparar estándares	Análisis cuantitativo y cualitativo, identificación de problemas
Actuar	Diseño e implementaciones de mejora	Elaboración de guía metodológica usando DevOps

3.4.1.2. Método cuantitativo

Para la recolección de datos, se aplicó una encuesta estructurada dirigida a los seis desarrolladores de la Unidad de Desarrollo de Software y al personal técnico de la Dirección de TIC de la UPEC. El cuestionario fue validado por tres profesionales del área de ingeniería de software, quienes verificaron la pertinencia, coherencia y claridad de los ítems. Los resultados obtenidos se sistematizaron mediante herramientas estadísticas descriptivas que permitieron identificar patrones y niveles de cumplimiento dentro de los procesos actuales.

Además, se utilizó un checklist técnico basado en la norma ISO/IEC 26563:2022, el cual fue revisado y validado por tres expertos para asegurar su alineación con los criterios internacionales de gestión de configuración de software.

3.4.1.3. Método cualitativo

Se conoce como método cualitativo ya que en el diseño de la comprensión de prácticas actuales y así el diagnóstico de problemas que ya existen con esto se propone mejoras concretas a cada uno de los problemas hallados dentro del proceso de gestión de configuración de software es adecuado decir que este método nos permite integrar la observación el análisis y el hacer dentro de un proceso junto y que pertenece a la mejora continua de la institución.

Las entrevistas no estructuradas al principio permiten un análisis dónde se ve la percepción de las experiencias de las personas dentro de la unidad de desarrollo con respecto al control de versiones y la falta de automatización de los procesos es ahí donde podemos ver la necesidad de las prácticas DevOps esta información se hace referencia a lo que es cualitativo por lo cual se debe complementar.

3.4.1.4. Método proyectivo

A partir del análisis de los resultados cuantitativos y cualitativos se elaboró una guía metodológica para la implementación de un sistema de gestión de configuración de software basado en DevOps. Este método permitió transformar el diagnóstico técnico y organizacional en una propuesta práctica orientada a optimizar los procesos de desarrollo y mantenimiento de software dentro de la UPEC.

La guía integra tres secciones principales flujo de trabajo Git y ramas, integración y pruebas, y herramientas y mejoras, que se corresponden con las dimensiones operativas establecidas en la operacionalización de las variables.

3.4.2. Técnicas

3.4.2.1. Técnicas e instrumentos de recolección de datos

De acuerdo con la naturaleza mixta del estudio, se utilizaron diferentes técnicas de recolección de información.

Tabla 9. Técnicas e instrumentos de recolección de datos

Tipo de dato	Técnica	Características	Instrumento
Cuantitativo	Encuesta estructurada	Permite recopilar información estandarizada sobre las prácticas actuales de gestión de configuración de software, control de versiones y uso de herramientas DevOps.	Cuestionario validado.
Cuantitativo	Checklist técnico	Evalúa el grado de cumplimiento de los requisitos de la norma ISO/IEC 26563:2022 relacionados con la gestión de configuración.	Lista de verificación ISO/IEC 26563:2022
Cualitativo	Entrevista no estructurada	Permite explorar experiencias, percepciones y opiniones del personal técnico sobre los procesos actuales y la posible implementación de DevOps.	Guía de entrevista abierta.
Cualitativo	Revisión Documental	Analiza documentos, políticas internas, registros de proyectos y repositorios de código existentes en la Dirección de TIC.	Análisis documental.

3.4.2. 2. Diagrama de flujo metodológico

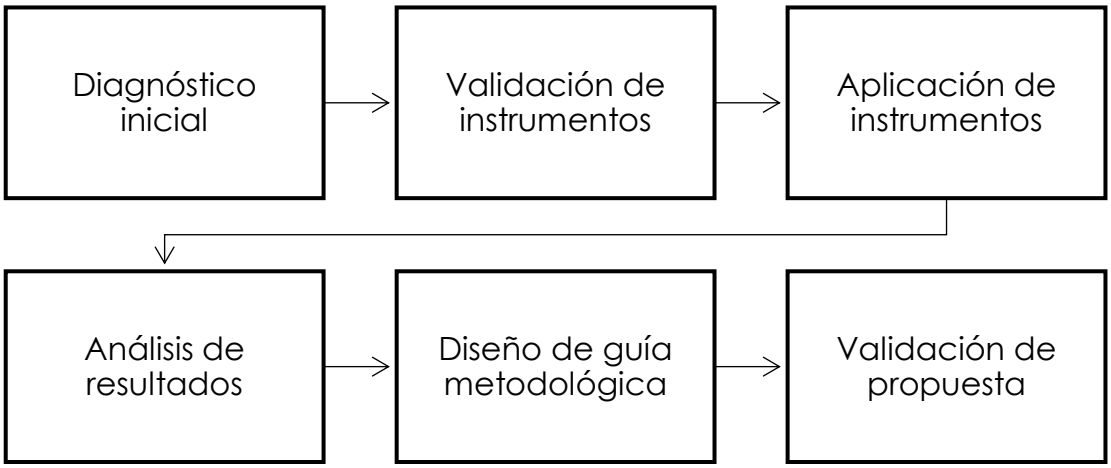


Figura 2. Flujograma metodológico

3.5. ANÁLISIS ESTADÍSTICO

3.5.1. Análisis estadístico

El análisis estadístico de la investigación se desarrolló con el propósito de interpretar los datos obtenidos a partir de las encuestas, entrevistas y checklist aplicados al personal técnico de la Dirección de Tecnologías de la Información y Comunicación (TIC) y la Unidad de Desarrollo de Software de la Universidad Politécnica Estatal del Carchi (UPEC).

La investigación se llevó a cabo durante el período académico correspondiente al año 2025, en el entorno institucional de la UPEC, con la participación de seis desarrolladores y personal técnico vinculado al mantenimiento de los módulos virtuales universitarios. La naturaleza del estudio es no experimental, pues no se manipulan las variables, sino que se analizan los fenómenos tal como ocurren en su contexto real.

3.5.2. Población y muestra

La población de estudio corresponde al personal técnico y desarrolladores pertenecientes a la Dirección de TIC y la Unidad de Desarrollo de Software de la UPEC. Debido al tamaño reducido de la población (seis participantes), se emplea un muestreo no probabilístico por conveniencia, ya que los participantes fueron seleccionados por su vínculo directo con los procesos de gestión de software y su disponibilidad para colaborar en la investigación.

3.5.3. Tipo de datos y variables

Los datos obtenidos se clasifican en cualitativos y cuantitativos, de acuerdo con su naturaleza y escala de medición.

Tabla 10. Datos y variables

Tipo de variable	Escala de medición	Características
Cualitativas nominales	Nominal	Datos categóricos que identifican atributos como el tipo de herramienta utilizada (Git, Jenkins, Docker), cumplimiento de buenas prácticas y procesos aplicados. No permiten operaciones aritméticas.
Cualitativas ordinales	Ordinal	Reflejan niveles de cumplimiento o valoración (por ejemplo: bajo, medio, alto) sobre el control de versiones, trazabilidad y automatización. Permiten establecer un orden jerárquico entre categorías.
Cuantitativas discretas	Intervalo	Representan conteos o frecuencias (número de versiones activas, frecuencia de pruebas, commits por semana). Permiten el uso de medidas de tendencia central y dispersión.
Cuantitativas continuas	Razón	Expresan proporciones o tiempos (porcentaje de automatización, tiempo de despliegue, frecuencia de actualizaciones). El cero tiene valor real. Se aplican medidas como media y desviación estándar.

3.5.4. Resultados de encuesta

1. **¿Qué estrategia de branching (ramificado) usan en tu equipo?**

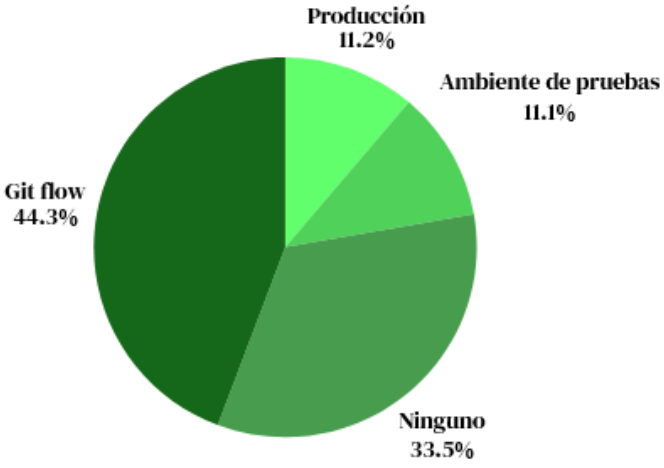


Figura 3. Resultado de la pregunta 1

Existe un conocimiento parcial de estrategias de ramificación (principalmente Git Flow), todavía persisten prácticas manuales o no estandarizadas que limitan la automatización y la trazabilidad de los procesos de desarrollo.

2. **En una escala del 1 al 5, ¿qué tan clara es la política de commits (mensajes, estructura)?**

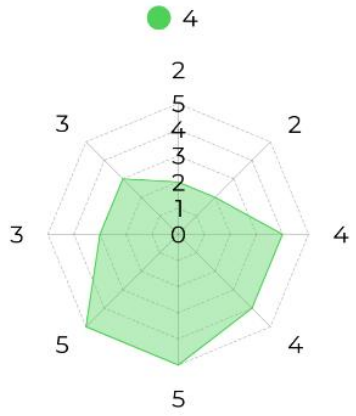


Figura 4. Resultado de la pregunta 2

Los desarrolladores perciben una claridad aceptable en la estructura y redacción de los mensajes de commit, lo que indica la existencia de lineamientos básicos para registrar los cambios en el repositorio. Sin embargo, la variabilidad observada en algunos puntos de la gráfica refleja que no todos los miembros del equipo aplican de manera uniforme dichas políticas, lo cual puede generar inconsistencias en la trazabilidad y en la comprensión del historial de versiones.

3. **¿Con qué frecuencia realizas pull requests (solicitudes de integración) o merge requests (fusión de ramas)?**

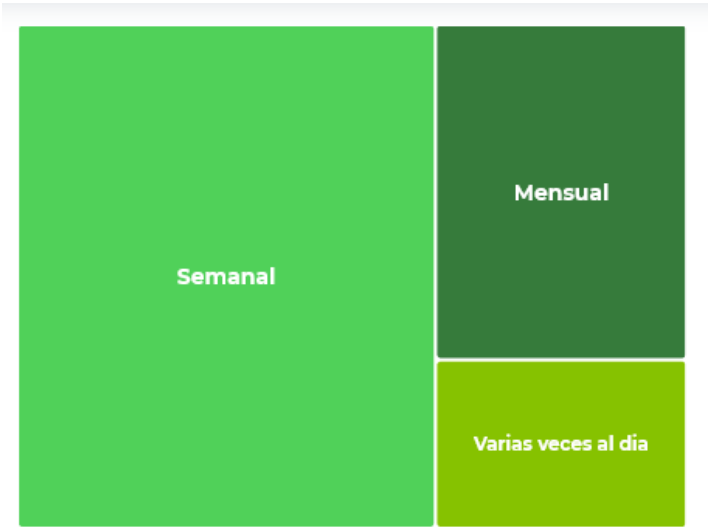


Figura 5. Resultado de la pregunta 3

Estos resultados sugieren que, el equipo ya aplica estrategias de integración periódica, la frecuencia semanal aún limita la detección temprana de errores y la automatización de despliegues. De acuerdo con las buenas prácticas DevOps, un flujo de integración continua ideal implica fusiones diarias o incluso múltiples veces al día, con el fin de mantener el código estable y reducir conflictos.

4. **¿Tu equipo ejecuta pruebas automatizadas antes de integrar código?**

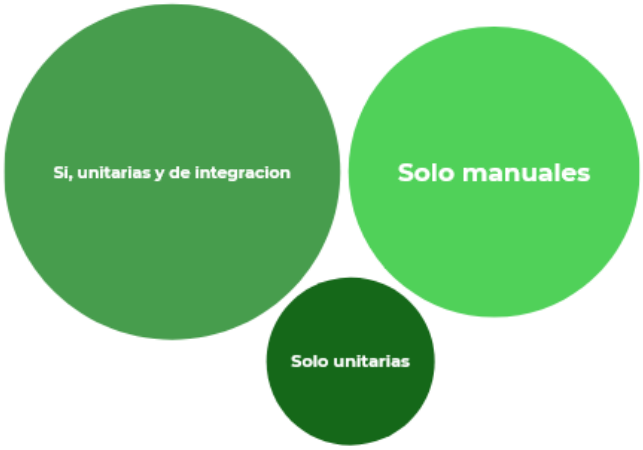


Figura 6. Resultado de la pregunta 4

Los resultados evidencian que, aunque existe una tendencia positiva hacia la implementación de pruebas automatizadas, aún no se ha consolidado una cultura institucional que integre completamente las fases de testing automatizado dentro del flujo de integración y entrega continua (CI/CD).

5. **En escala del 1 al 5, ¿qué tan rápido detectan errores gracias a la integración continua (CI)?**

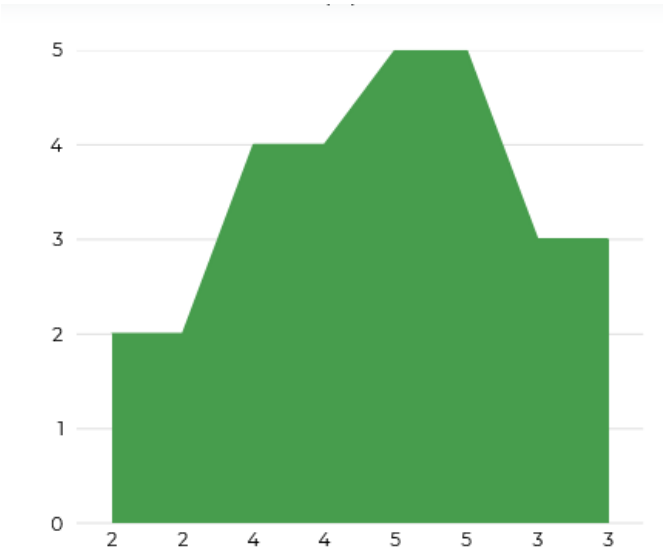


Figura 7. Resultado de la pregunta 5

La Figura muestra que la mayoría de los desarrolladores calificó con valores entre 4 y 5 la rapidez en la detección de errores mediante la integración continua (CI), lo que refleja una percepción positiva sobre la eficiencia de este proceso.

El equipo de desarrolladores que suelen aplicar prácticas ahí puede identificar fallos de forma oportuna en el proceso del desarrollo es así que se puede mejorar la estabilidad y calidad del software dado como producto final pero al ver que hay niveles con respuesta en el 2 y el 3 puede ser una sugerencia de que aún hay diferencias en el grado de automatización y aplicación de esta práctica.

6. **¿Qué porcentaje aproximado de tu tiempo de desarrollo dedicas a configurar o mantener pipelines (procesos de CI/CD)?**

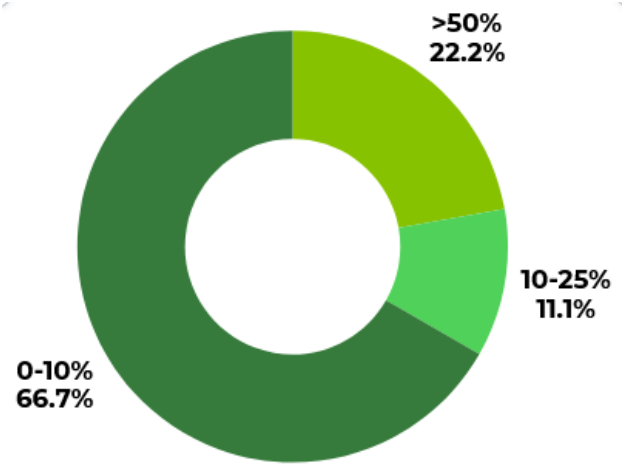


Figura 8. Resultado de la pregunta 6

La adopción limitada y que no es uniforme muestra que las prácticas de automatización dentro de la unidad de desarrollo si bien partes del equipo pueden implementar procesos de integración y despliegue continuo la mayoría aún depende de procesos manuales que no están estandarizados.

7. **En una escala del 1 al 5, ¿con qué grado de satisfacción evalúas las herramientas CI/CD (integración y entrega continua) actuales?**

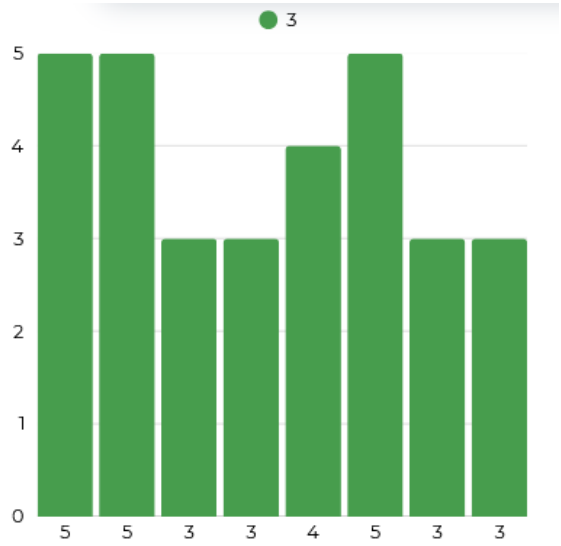


Figura 9. Resultado de la pregunta 7

Dentro de la unidad de desarrollo las herramientas que comprenden a la integración y entrega continua son buscadas y valoradas de forma buena, aunque se debe establecer y fortalecer su adopción institucional para así unir los criterios para maximizar la efectividad dentro del desarrollo del producto.

8. **¿Qué funcionalidad consideras más necesaria para agilizar tu trabajo?**

Figura 10. Resultado de la pregunta 8

La Figura muestra que la mayoría de los desarrolladores considera que la integración con testing automatizado es la funcionalidad más necesaria para optimizar su flujo de trabajo. Se destaca la necesidad de monitoreo en tiempo real, reflejando el interés del equipo por contar con herramientas que permitan supervisar el estado del sistema y detectar fallos de manera inmediata. Finalmente, un grupo menor señaló el despliegue más rápido como búsqueda de eficiencia en la entrega del software.

3.5.5. Resultado del checklist

la funcionalidad más relevante, lo que indica una

C: CUMPLE P: PARCIAL NC: NO CUMPLE

Tabla 11. Checklist diagnóstico

Nº	Requisito	Nivel de Necesidad	Estado Actual	Observaciones
1	Identificación de todos los módulos virtuales y sus versiones	Alta	P	Identificación manual, sin trazabilidad
2	Uso de herramientas de control de versiones (Git, GitLab, etc.)	Alta	NC	No se usan herramientas formales
3	Documentación de cambios realizados en los sistemas (historial de versiones)	Alta	NC	Cambios no se documentan o se hacen de forma informal
4	Flujo definido para solicitud, aprobación e implementación de cambios	Alta	P	Existe flujo informal, sin estandarización
5	Existencia de repositorio centralizado para código y documentación técnica	Alta	NC	Información dispersa en múltiples lugares
6	Definición de roles y responsabilidades para la gestión de configuración	Media	P	Tareas asignadas de forma no formal
7	Realización de respaldos periódicos y restauración en caso de fallos	Media	NC	No existe protocolo ni herramientas para backup
8	Capacitación al personal TIC sobre DevOps, CI/CD y control de versiones	Alta	NC	Conocimiento general, sin formación técnica
9	Uso de metodologías ágiles o híbridas en los proyectos TIC	Media	P	Métodos mixtos sin formalización
10	Auditoría interna sobre versiones, cambios y configuraciones	Media	NC	No se realizan auditorías
11	Selección y uso de herramientas DevOps según las capacidades institucionales	Alta	P	Conocimiento superficial de herramientas
12	Adaptación de soluciones al entorno técnico actual de la UPEC	Media	C	Infraestructura general es compatible
13	Registro y trazabilidad de cambios por usuario y fecha	Alta	NC	No hay sistema de seguimiento
14	Integración continua o despliegue automatizado	Alta	NC	Procesos totalmente manuales

3.5.6. Comparativa de resultados

1. **Control de versiones y flujo de trabajo**

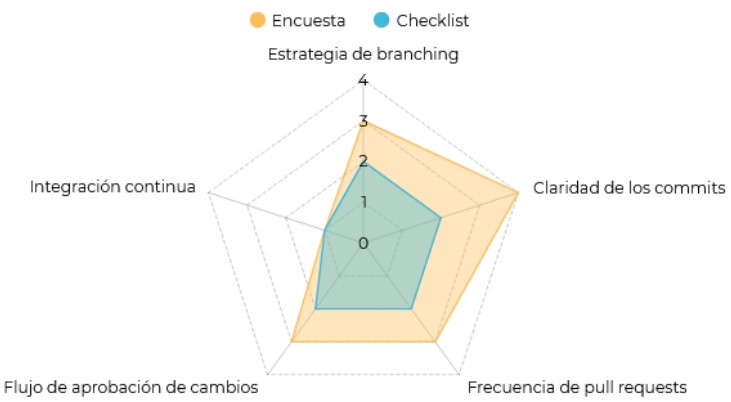


Figura 11. Control de versiones y flujo de trabajo

La comparación evidencia una discrepancia entre la percepción del equipo y el cumplimiento real según la norma. Aunque los desarrolladores consideran que los commits son claros, la falta de trazabilidad formal y de un sistema que registre los cambios por usuario y fecha refleja una brecha significativa en la gestión del control de versiones. Esto resalta la necesidad de fortalecer la documentación y el seguimiento de cambios dentro del flujo de trabajo.

2. **Integración continua y pruebas automatizadas**

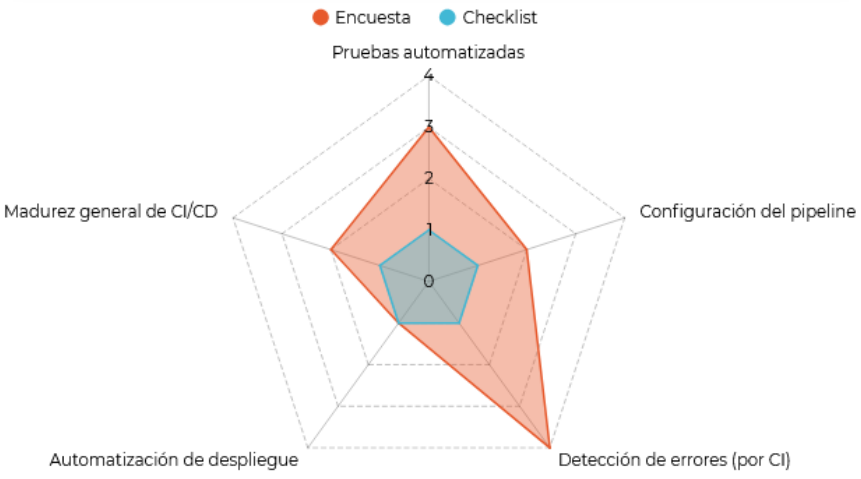


Figura 12. Integración continua y pruebas automatizadas

Los resultados reflejan que, aunque algunos miembros del equipo aplican pruebas unitarias y perciben efectividad en la detección de errores, la evaluación técnica demuestra que no existen procesos formales de integración

continua ni despliegue automatizado. La automatización es mínima y los flujos siguen siendo manuales, evidenciando una brecha significativa entre la práctica percibida y el cumplimiento de la norma ISO/IEC 26563:2022.

3. Gestión y selección de herramientas DevOps

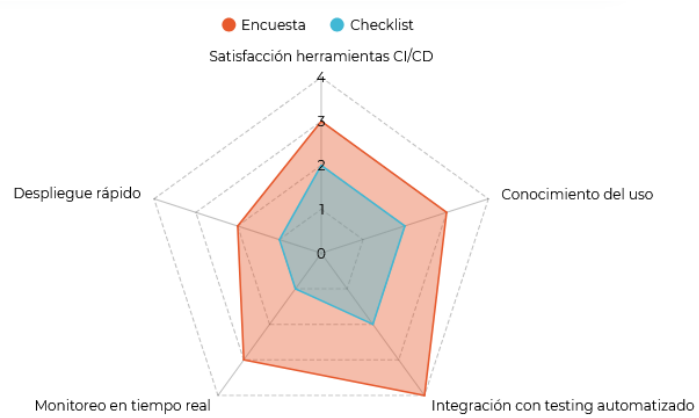


Figura 13. Gestión y selección de herramientas DevOps

Como se puede observar en el diagrama tipo radar el equipo de desarrollo tiene una satisfacción buena con las herramientas tipo CI/CD se puede ver que reconocen la necesidad de integrar funciones avanzadas como el testing de automatización y el monitoreo en tiempo real se puede también mostrar que la evaluación frente al checklist es superficial ya que la adopción parcial de las herramientas DevOps indica que hay una brecha entre la percepción de la unidad de desarrollo y el nivel de madurez real que usan las tecnologías de integración y despliegue.

4. Capacitación y madurez del equipo

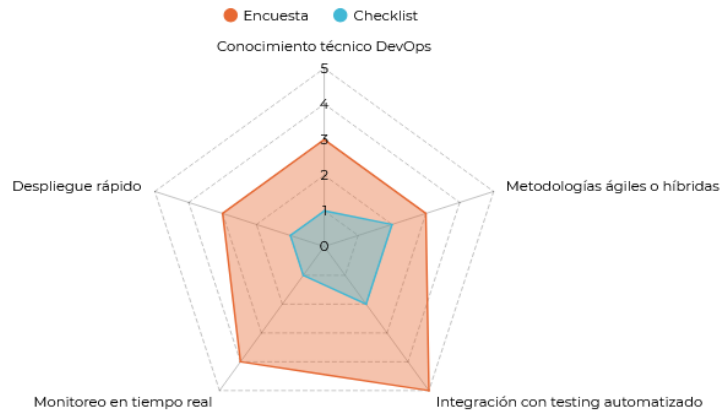


Figura 14. Capacitación y madurez del equipo

Se pone en evidencia que la unidad de desarrollo percibe correctamente las áreas funcionales que se pueden optimizar dentro de su trabajo dando prioridad a la integración con testing automatizado y el monitoreo real en comparación con el checklist se refleja que hay una brecha en la formación técnica en DevOps y en aplicación de buenas prácticas de metodologías ágiles esto muestra que existe conciencia de las necesidades aunque aún no se dispone de las capacidades ni procesos estandarizados para implementarlos.

IV. RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS

4.1.1. Identificación de problemas SGCS

4.1.1.1. La escasa trazabilidad en la identificación de cambios

En la unidad de desarrollo se registran los cambios realizados de forma manual sin ningún método formal o estandarizado que permita rastrear las modificaciones que se hacen no se sabe quiénes ejecutan los cambios o a qué momento se lo hace esto da como resultado la falta de trazabilidad que pone en riesgo el control y puede tener errores que produzcan el riesgo de las pérdidas de versiones.

4.1.1.2. Inexistencia de herramientas formales de control de versiones

Los desarrolladores no utilizan soluciones generales profesionales como git lo que hace que sea un límite en la aplicación correcta de la gestión de versiones y que no haya protección del código fuente haciendo que haya un mal control de versiones y esto genera que haya veces sobre escritura pérdida de cambios y diferentes conflictos en la integridad de código.

4.1.1.3. Documentación técnica insuficiente y no versionada

La documentación que se tiene sobre los cambios y los procesos de desarrollo no tiene una estructura adecuada ni estandarizada y no se actualiza en función a las modificaciones de los sistemas lo cual permite que la transferencia no sea correcta y afecta que haya una buena mantenibilidad de software creando dependencia en el conocimiento de cada uno de los desarrolladores.

4.1.1.4. Flujo de trabajo informal y no estandarizado

La unidad de desarrollo no tiene pipeline definido y no hay un flujo de git formal cada una de las personas se aplica a diferentes prácticas al desarrollo al hacer esto los componentes tienen falta de estandarización y provoca diferentes inconsistencias cuando se hacen las entregas y a veces confusión en la integración y baja consistencia operativa.

4.1.1.5. Ausencia de un repositorio Centralizado

Todo se almacena en el código que está disperso en diferentes máquinas carpetas sin un repositorio único con los códigos al tener un repositorio que no está haciendo centralizado se puede impedir que el trabajo sea colaborativo y se aumenta que haya pérdidas de información también se llega a dificultar la auditoría de código.

4.1.1.6. Rol y responsabilidades no formalizados

No se existen roles definidos para la gestión de configuración integración pruebas y despliegue las funciones se asignan informalmente lo que provoca que se genere duplicidad de tareas, falta de responsabilidad clara y dificultad de implementación de cualquier proceso.

4.1.1.7. Ausencia de políticas de respaldo y recuperación

No existe un procedimiento de backup para respaldar versiones, las configuraciones o documentos críticos dentro del sistema, sin estas políticas, si hay un error humano se puede ocasionar pérdidas en el código y afectar la continuidad de la operación.

4.1.1.8. Deficiencias de la formación técnica del personal

El personal aun no cuenta con capacitación formal en prácticas DevOps por lo cual hay falta de conocimiento para adopción de buenas prácticas y esto aumenta el riesgo de errores operativos.

4.1.1.9. Uso de métodos híbridos sin estructura formal

Se aplican elementos de distintas metodologías, aunque cabe aclarar que se trabaja bajo una modalidad técnicamente llamada scrum no se sigue formalmente todas las fases.

4.1.1.10. Falto de auditorías del proceso de configuración

No se realizan auditorías internas para verificar que se cumplan los procedimientos, los estándares y las buenas prácticas de gestión de configuración por lo cual es imposible detectar incumplimientos lo que afecta la calidad de software y la confiabilidad del proceso.

4.1.1.11. Adopción superficial de herramientas y practicas DevOps

Se utilizan algunas herramientas de DevOps pero no se aplican prácticas completas lo cual limita el rendimiento del equipo.

4.1.1.12. Inexistencia de un sistema de seguimiento de cambios por usuario

El proceso no permite identificar qué desarrollador realizó un cambio, ni si se puede medir el impacto de lo que hizo lo cual afecta la trazabilidad y fallos en el código.

4.1.1.13. Procesos manuales y ausencia de CI/CD

Las integraciones pruebas y despliegue se realizan manualmente sin herramientas de automatización como Jenkins o Github actions los procesos manuales aumentan la probabilidad de errores y retrasan el tiempo de entrega lo que va en contra de lo que es consiste en DevOps.

4.1.2. Árbol de problemas

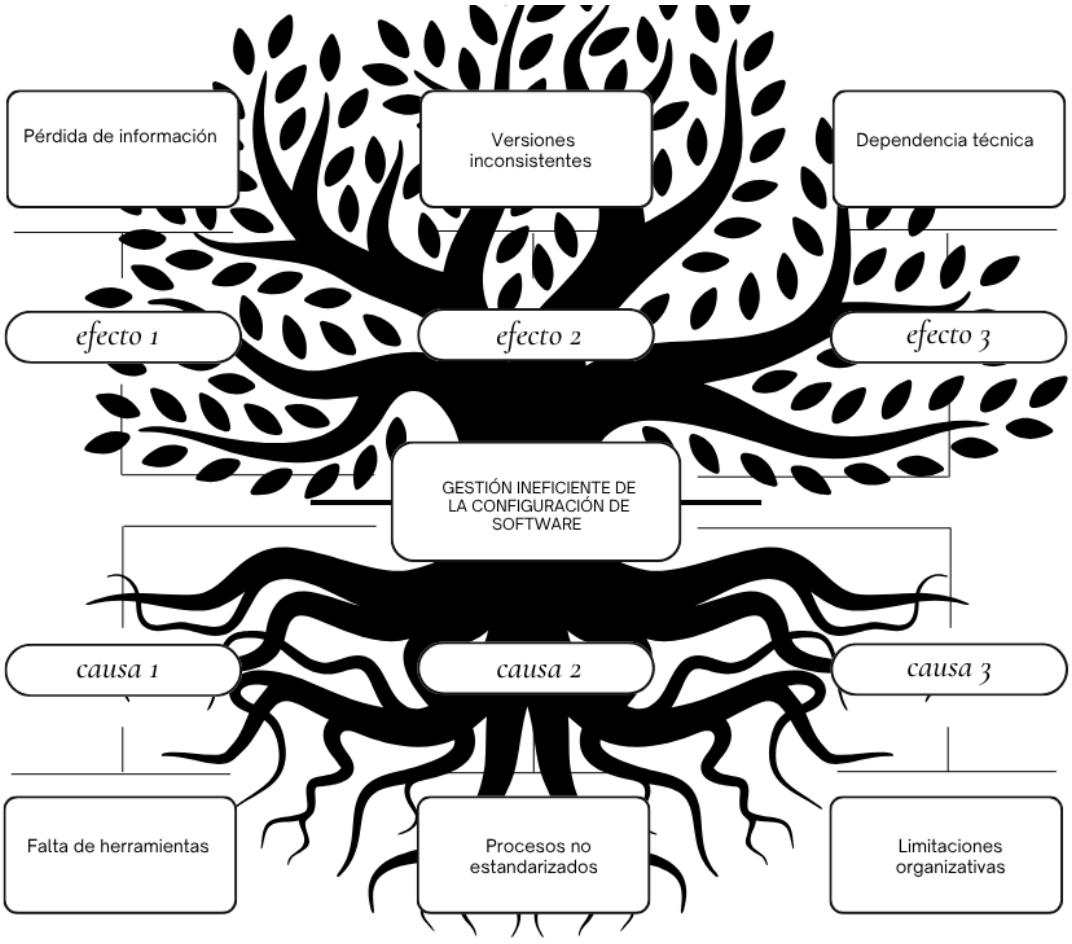


Figura 15. Árbol de problemas

4.1.3. Mapeo de problemas con procesos del SGCS

Basado en lo establecido por ISO/IEC 26563. Se realiza el mapeo de la siguiente manera.

Tabla 12. Mapeo de problemas a procesos de gestión

Problema	Proceso de gestión afectado
La identificación se realiza manualmente y sin trazabilidad.	Identificación de la configuración
No se usan herramientas formales de control de versiones.	Control de la configuración
Los cambios no se documentan ni se versionan correctamente.	Identificación y control de la configuración
Flujo informal sin estandarización.	Control de la configuración
No hay un repositorio único; la información está dispersa.	Identificación y control de la configuración
No existen roles formalmente establecidos; las tareas se reparten informalmente.	Gestión organizacional / planificación de la configuración
No existen políticas ni herramientas para backup.	Estado de la configuración / control de la configuración
Falta de formación técnica formal; solo conocimiento general.	Gestión de soporte / mejora continua
Se aplican métodos mixtos sin estructura ni seguimiento formal.	Planificación del proceso / gestión del cambio
No se realizan auditorías para verificar la conformidad.	Auditoría de la configuración
Conocimiento superficial y adopción parcial de herramientas.	Gestión de herramientas / integración continua
No existe un sistema de seguimiento de cambios por usuario.	Control de la configuración / auditoría de la configuración
Procesos manuales sin CI/CD implementado.	Gestión de la integración y entrega continua

4.1.4. Integración de Prácticas DevOps a procesos SGCS

4.1.4.1. Identificación de la configuración

La práctica DevOp conocida como "Iac o Infraestructura como código", permitirá la definición y el versionamiento de los elementos dentro de la configuración correspondientes a archivos o dependiente dependencias mediante la reproducción de scripts.

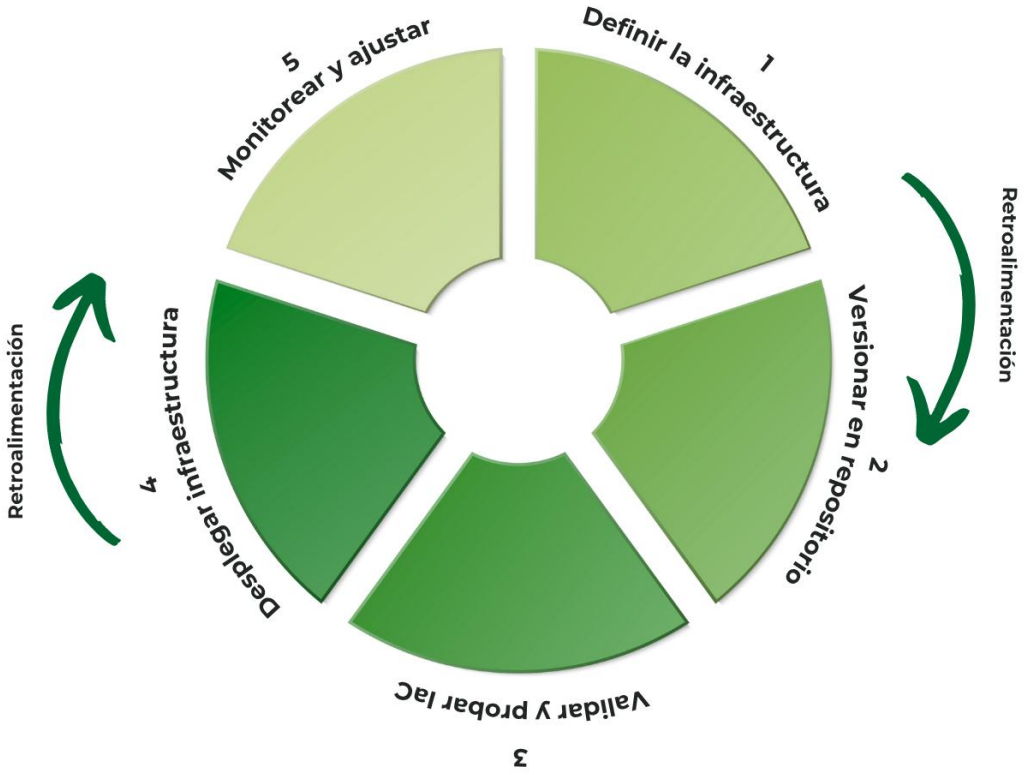


Figura 16. Flujo IaC

4.1.4.2. Control de la configuración

La práctica DevOps "control de versiones" podrá asegurar la trazabilidad de cambios dentro de los artefactos de software mediante las ramas y los commits controlados.

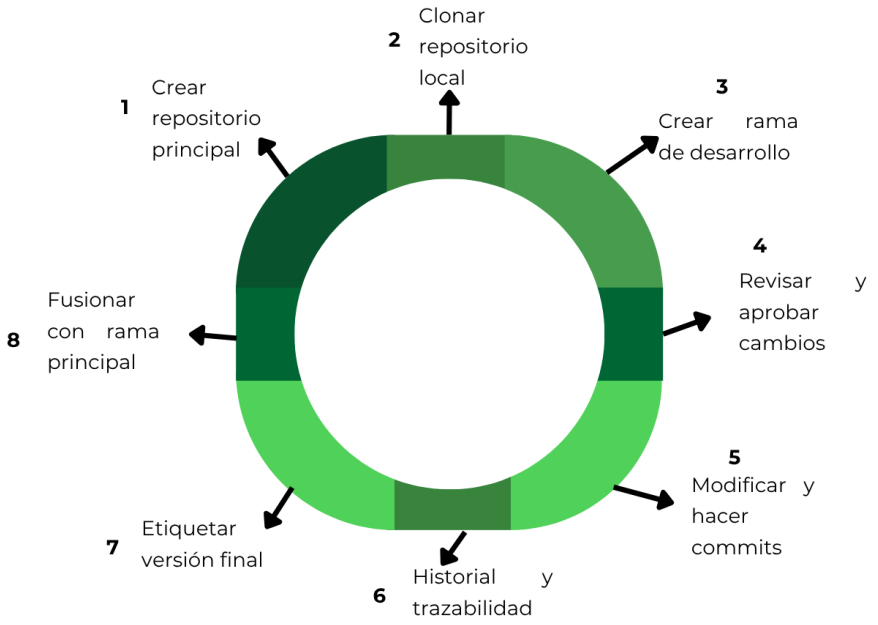


Figura 17. Flujo control de versiones

4.1.4.3. Identificación y control de la configuración

Los "repositorios centralizados" permitirán la unificación de las versiones y así facilitar la revisión colaborativa de las configuraciones.

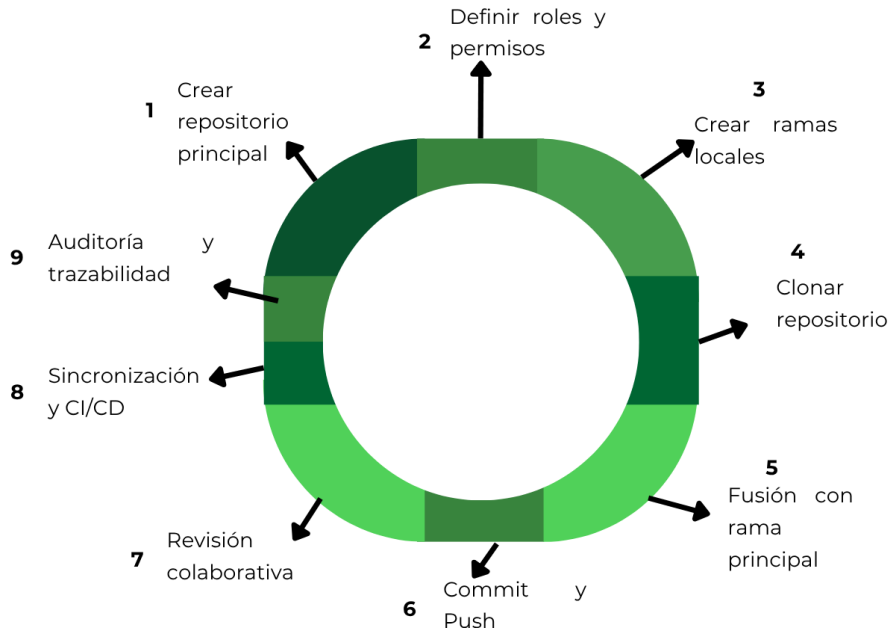


Figura 18. Flujo repositorios centralizados

4.1.4.4. Gestión organizacional

La "gestión ágil con herramientas CI/CD" es una práctica DevOp que permite la planificación, la automatización, la construcción y el despliegue de configuración.

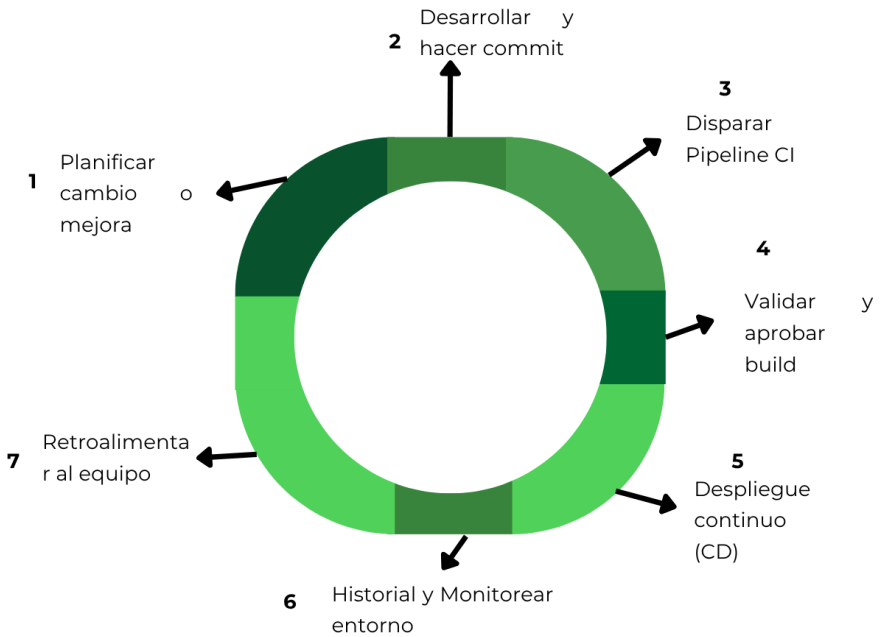


Figura 19. Flujo CI/CD

4.1.4.5. Planificación del proceso

La práctica "gestión de cambios automatizada con pipeline" controla las aprobaciones y los despliegues controlados cuando se realizan modificaciones.

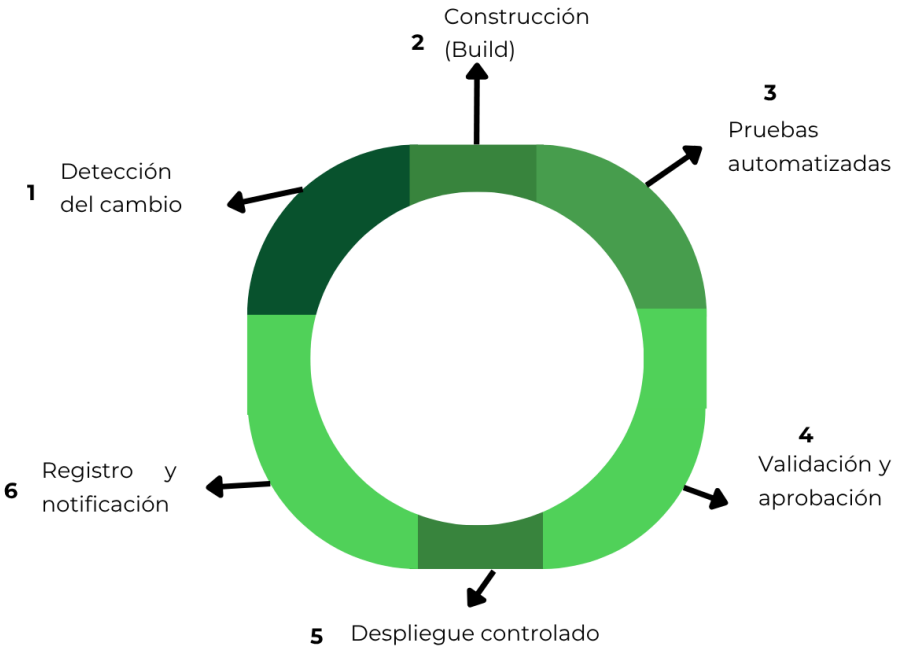


Figura 20. Flujo gestión de cambios

4.1.4.6. Estado de la configuración

El "monitoreo y la observabilidad" como práctica DevOp va a permitir la visualización del estado actual en vivo de los entornos y las configuraciones en tiempo real.

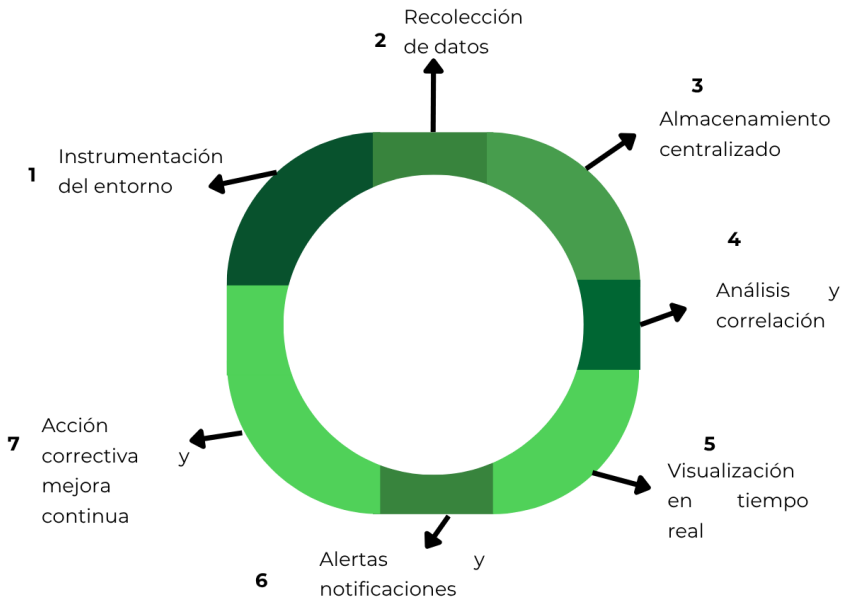


Figura 21. Flujo monitoreo y observabilidad

4.1.4.7. Gestión de soporte

La práctica de DevOp "feedback" continuo se analiza mediante métricas de desempeño para implementar mejoras iterativas durante el proceso.

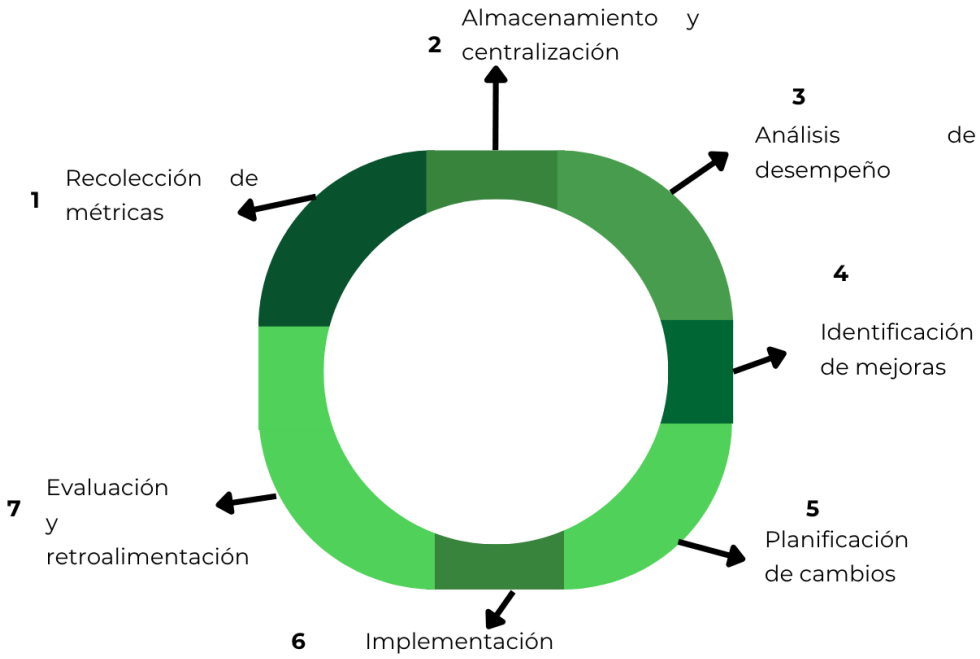


Figura 22. Flujo feedback

4.1.4.8. Auditoria de la configuración

La "trazabilidad automatizada" permite registrar los cambios y acciones en los entornos automáticamente lo que asegurará el cumplimiento.

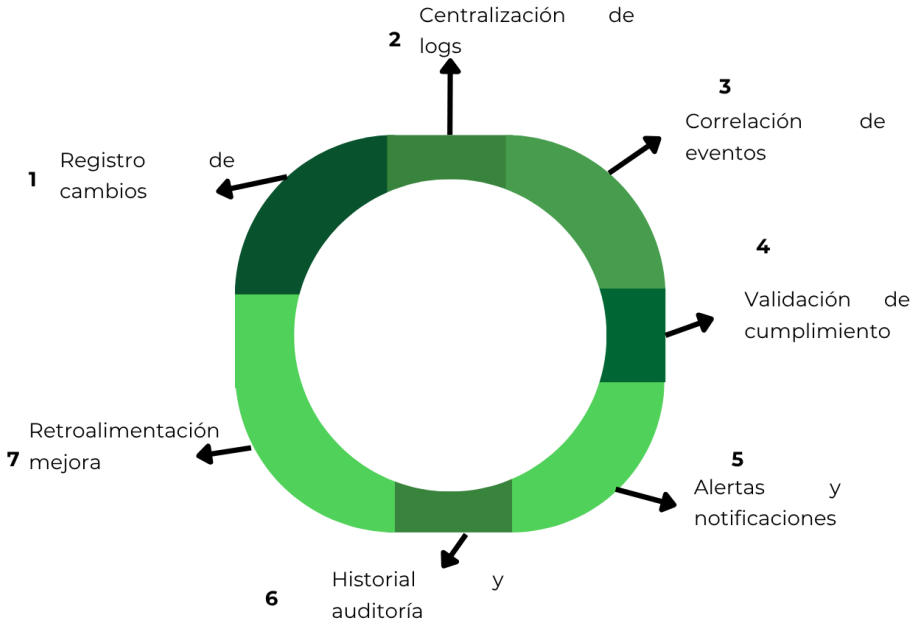


Figura 23. Flujo de trazabilidad automatizada

4.1.4.9. Gestión de herramientas

La práctica "integración continua o CI" automatiza las pruebas y las validaciones de los elementos de configuración antes de que se realice el despliegue.

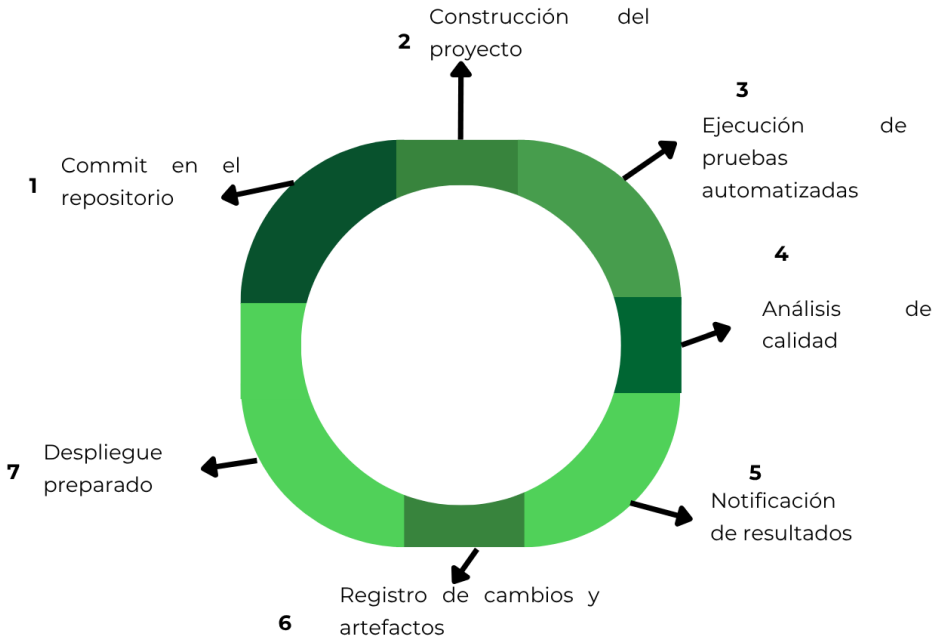


Figura 24. Flujo integración continua

4.1.5. Herramientas correspondientes a prácticas DevOps

4.1.5.1. Infraestructura como código (IaC)

Las herramientas correspondientes para Iac son:

- Terraform
- Ansible
- Aws CloudFormation

Figura 25. Herramientas IaC

Terraform y Ansible son herramientas de código abierto gratuitas. AWS CloudFormation pertenece a AWS y puede tener costos, ofrece créditos académicos mediante AWS Educate y AWS Academy.

4.1.5.2. Control de versiones

Las herramientas correspondientes para control de versiones son:

- Git
- Github
- Gitlab
- Bitbucket

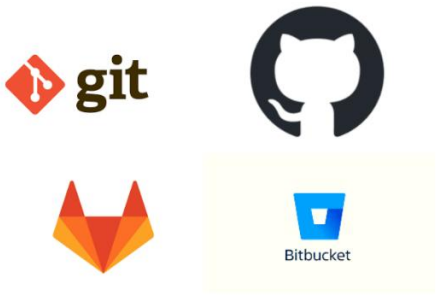


Figura 26. Herramienta control de versiones

Git es gratuito y de código abierto. GitHub ofrece beneficios educativos mediante GitHub Education. GitLab dispone de versión gratuita y convenios académicos. Bitbucket brinda licencias sin costo para instituciones educativas.

4.1.5.3. Repositorios centralizados

Las herramientas correspondientes para repositorios centralizados son:

- Github entreprise
- Gitlab
- Bitbucket server
- Subversion

Figura 27. Repositorios centralizados

GitLab CE es gratuito y de código abierto. GitHub Enterprise y Bitbucket Server son versiones comerciales, aunque ofrecen planes académicos. Subversion (SVN) es una herramienta libre usada en entornos corporativos que tienen control estricto.

4.1.5.4. Gestión ágil con herramientas CI/CD

Las herramientas correspondientes para gestión ágil con herramientas CI/CD son:

- Jenkins
- Gitlab CI/CD
- Azure DevOps
- CircleCI

Figura 28. Herramienta CI/CD

Jenkins y GitLab CI/CD son herramientas gratuitas y de código abierto. Azure DevOps y CircleCI ofrecen planes pagos, disponen de versiones gratuitas y beneficios académicos para universidades.

4.1.5.5. Gestión de cambios automatizado con pipelines

Las herramientas correspondientes para gestión de cambios automatizados con pipelines son:

- Jenkins pipelines
- Github actions
- ArgoCD
- Spinnake

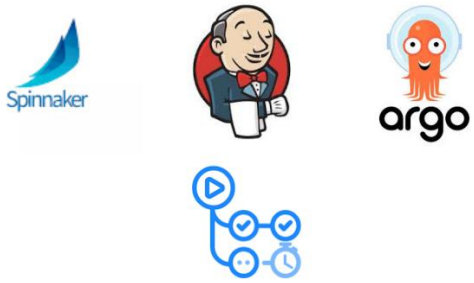


Figura 29. Herramientas cambios con pipelines

Jenkins Pipelines, GitHub Actions, ArgoCD y Spinnaker son herramientas gratuitas y de código abierto. GitHub ofrece planes educativos mediante GitHub Education para uso académico.

4.1.5.6. Monitoreo y observabilidad

Las herramientas correspondientes para monitoreo y observabilidad son:

- Oracle aplication perfomance monitoring
- Grafana
- ELK stack
- Prometheus

Figura 30. Herramienta de monitoreo y observabilidad

Prometheus, Grafana y ELK Stack son herramientas gratuitas y de código abierto para monitoreo y análisis. Oracle APM es una solución comercial y ofrece versiones académicas y pruebas gratuitas.

4.1.5.7. Feedback continuo

Las herramientas correspondientes para feedback continuo son:

- Slack
- Azure DevOps

- Jira
- Trello
- SonarQube

Figura 31. Herramienta feedback continuo

SonarQube es gratuito en su versión comunitaria. Jira y Trello ofrecen planes pagos y educativos de Atlassian for Education. Slack y Azure DevOps disponen de versiones gratuitas y licencias académicas para instituciones.

4.1.5.8. Trazabilidad automatizada

Las herramientas correspondientes para trazabilidad automatizada son:

- Gitlab
- Jira
- Azure boards
- Splunk
- ELK stack

Figura 32. Herramienta de trazabilidad

GitLab y ELK Stack son de código abierto. Jira, Azure Boards y Splunk son soluciones comerciales, pero ofrecen planes académicos para instituciones educativas.

4.1.5.9. Integración continua (CI)

Las herramientas correspondientes para CI son:

- Jenkins
- TravisCI
- CircleCI
- Github actions
- Gitlab CI

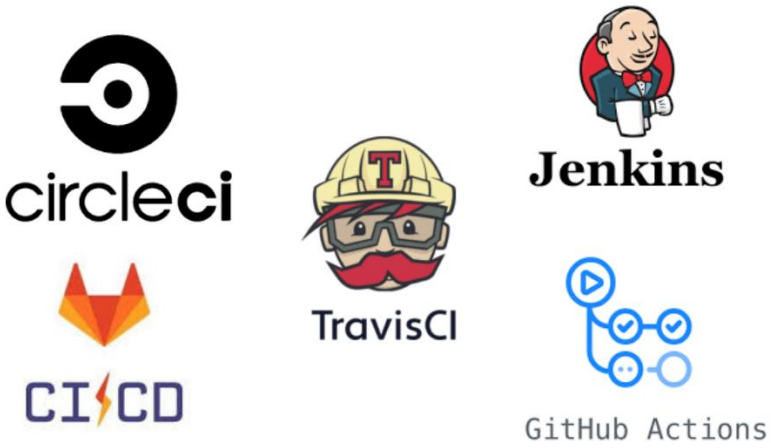


Figura 33. Herramienta CI

Jenkins y GitLab CI son de código abierto. GitHub Actions, Travis CI y CircleCI ofrecen versiones gratuitas y planes educativos para universidades.

4.1.6. Análisis de Percepciones del Equipo de Desarrollo y Dirección TIC sobre la Propuesta

4.1.6.1. Prácticas DevOps con más impacto

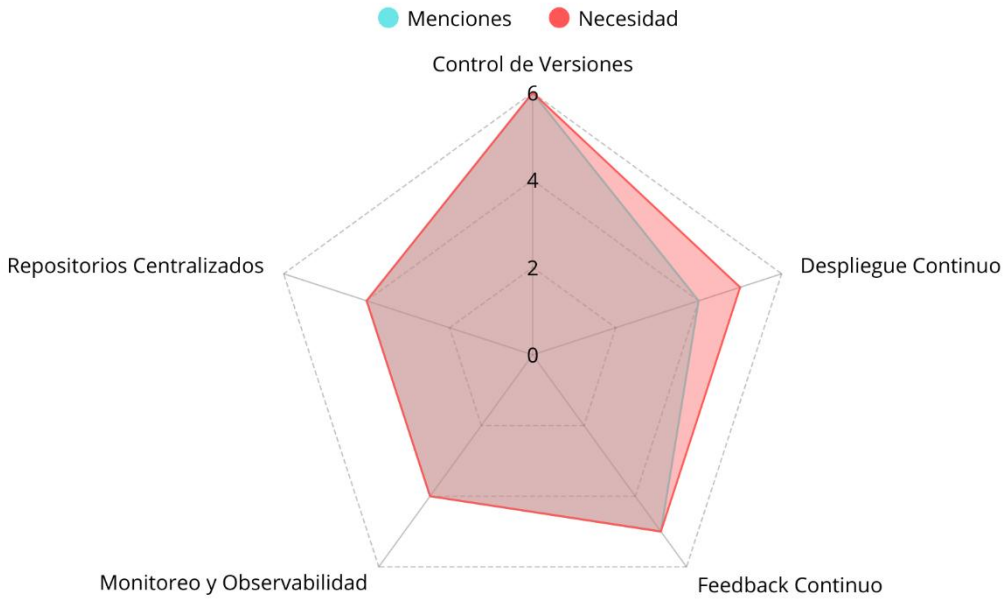


Figura 34. DevOps con más impacto

El control de versiones obtuvo el mayor puntaje tanto en menciones (6) como en necesidad (6), evidenciando su estado crítico dentro de los procesos actuales de la UPEC.

El feedback continuo y el despliegue continuo también presentan una alta valoración, indicando que la automatización del flujo de trabajo es vista como una mejora prioritaria.

El monitoreo/observabilidad y los repositorios centralizados, aunque menos mencionados, mantienen una necesidad significativa, lo cual sugiere oportunidades de fortalecimiento en la trazabilidad y estabilidad del ciclo de desarrollo.

4.1.6.2. Conceptos técnicos

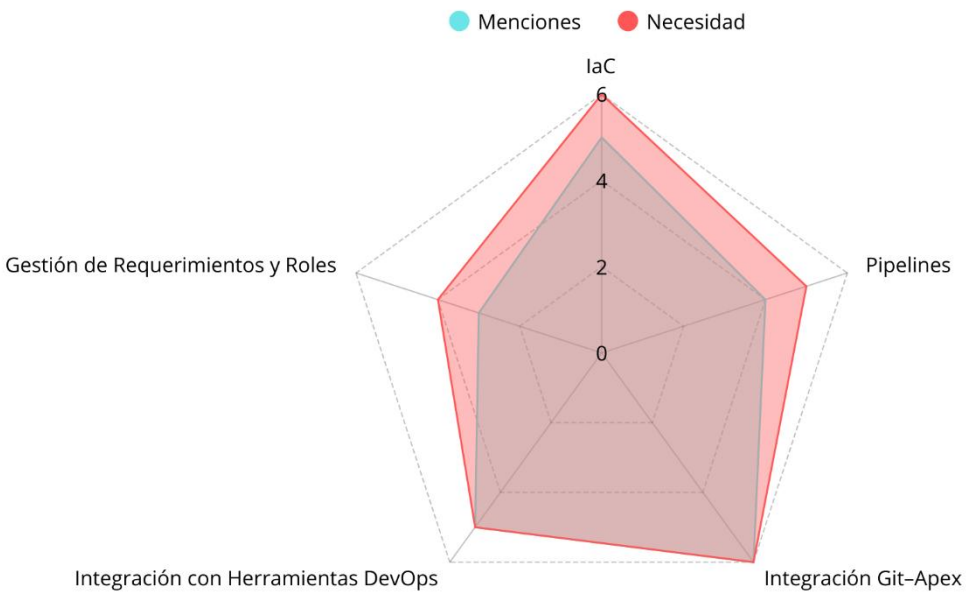


Figura 35. Conceptos técnicos

El análisis revela que la integración entre Git y Oracle Apex es el concepto más problemático, debido a las limitaciones del entorno low-code y la ausencia de archivos tradicionales que puedan gestionarse mediante control de versiones.

IaC y los pipelines de CI/CD también se destacan como conceptos nuevos para la mayoría del personal, lo cual evidencia la necesidad de capacitación adicional.

La integración de herramientas como Azure, Jenkins, Terraform o Slack genera dudas técnicas por la falta de documentación adaptada a la realidad de la UPEC.

Finalmente, se identificó la necesidad de clarificar el rol de la gestión de requerimientos dentro del flujo DevOps institucional.

4.1.6.3. Aplicabilidad de prácticas DevOps al entorno UPEC

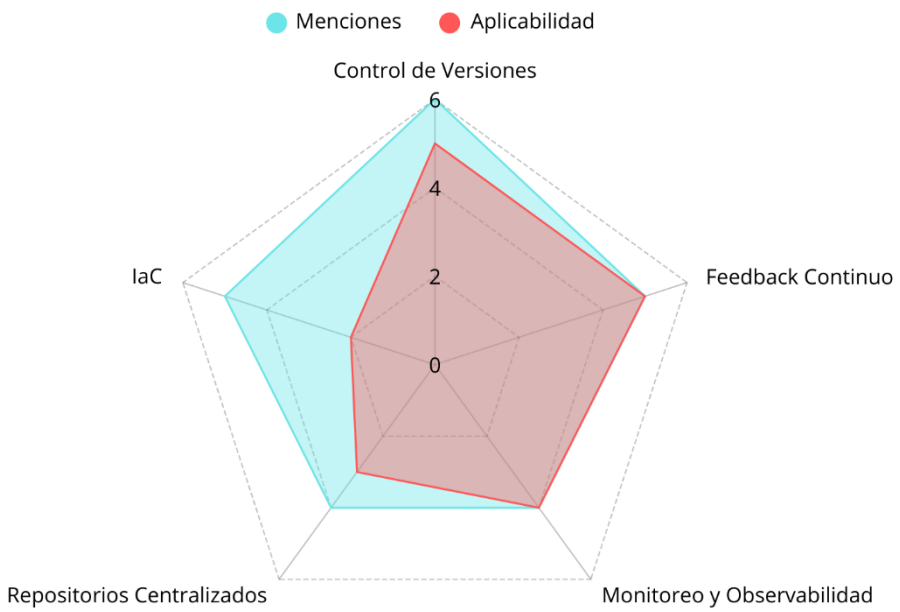


Figura 36. Aplicabilidad de prácticas DevOps al entorno UPEC

El análisis evidencia que el control de versiones y el feedback continuo son las prácticas con mayor aplicabilidad en la UPEC, aun considerando las limitaciones del entorno de Oracle Apex. El monitoreo y la observabilidad también resultan viables gracias a las herramientas nativas disponibles (Oracle APM).

En contraste, los repositorios centralizados presentan dificultades debido a la ausencia de integración nativa con Apex, lo cual limita su implementación completa.

Finalmente, IaC es la práctica con menor aplicabilidad actual, ya que la infraestructura de la UPEC se encuentra preconfigurada y no opera bajo modelos de despliegue automatizado basados en código.

4.1.6.4. Limitaciones técnicas y de organización identificadas

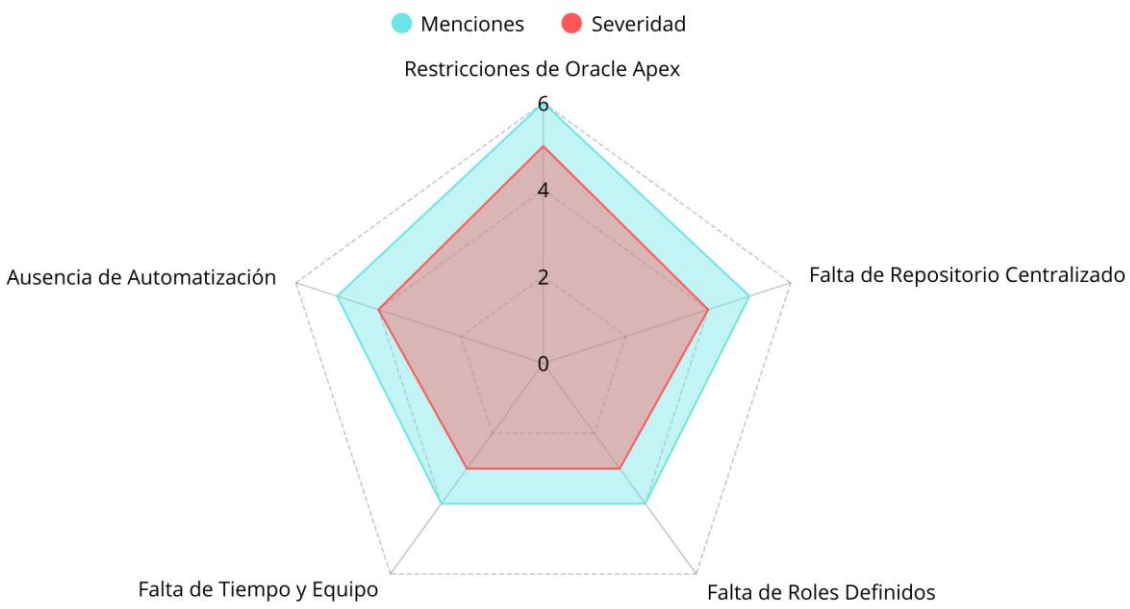


Figura 37. Limitaciones técnicas y de organización identificadas

De acuerdo con las entrevistas, la principal limitación para adoptar prácticas DevOps es la naturaleza del entorno Oracle Apex, que no permite un manejo tradicional de archivos, repositorios ni automatización de despliegues.

También se identificó la ausencia de un repositorio centralizado, la falta de roles claramente definidos dentro del flujo de trabajo y la insuficiencia de personal, lo que reduce la capacidad de aplicar prácticas más avanzadas.

Finalmente, la carencia de automatización para pruebas y despliegues dificulta el cumplimiento del ciclo DevOps completo.

4.1.6.5. Expectativas de la unidad de desarrollo

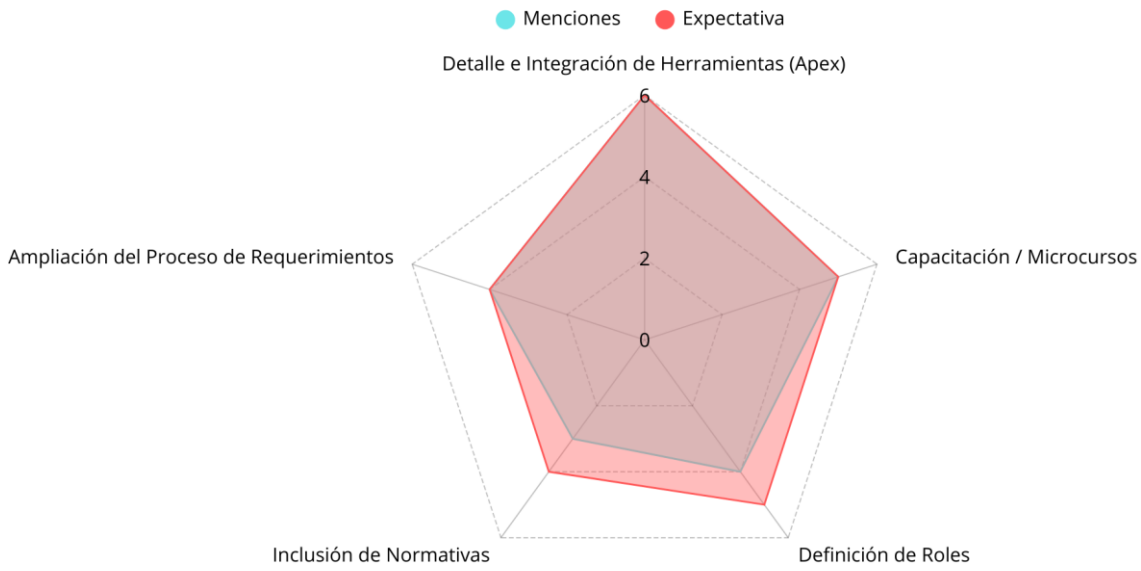


Figura 38. Expectativas de la unidad de desarrollo

El análisis muestra que la unidad de desarrollo espera principalmente una guía con mayor detalle sobre cómo integrar herramientas DevOps dentro del entorno Oracle Apex, debido a las particularidades técnicas de su plataforma. También se identificó una fuerte expectativa de capacitación y microcursos para facilitar la adopción de la metodología.

La definición de roles operativos y la inclusión de normativas legales y técnicas relevantes se consideran elementos necesarios para formalizar la implementación. Finalmente, los entrevistados resaltaron la importancia de reforzar el apartado de gestión de requerimientos para adaptarlo mejor a los procesos actuales.

4.1.6.6. Formato y estructura de la guía

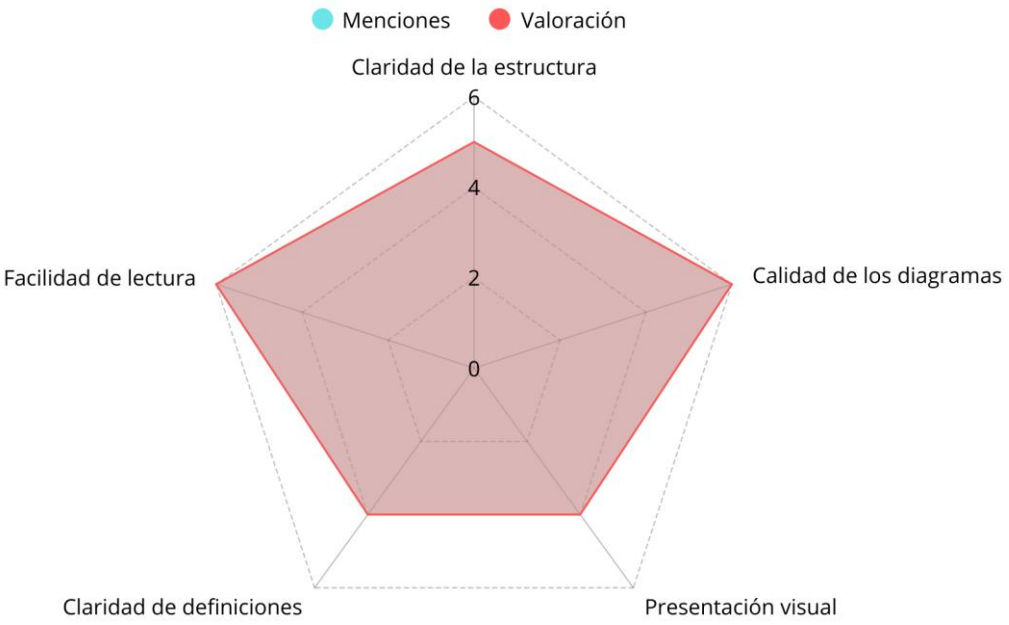


Figura 39. Formato y estructura de la guía

En general, la guía recibió una valoración altamente positiva en términos de estructura, claridad gráfica y usabilidad. Los entrevistados destacaron que los diagramas permiten comprender con mayor facilidad las prácticas DevOps y el flujo de trabajo, lo que incrementa la utilidad del documento como herramienta de consulta rápida.

Aunque el diseño visual fue bien recibido, se sugirió adecuarlo más al manual de marca institucional. También se identificó la necesidad de clarificar ciertas definiciones dentro de los gráficos. En conjunto, el formato fue percibido como didáctico, accesible y bien organizado.

4.1.6.7. Cambios de nivel de compresión sobre DevOps

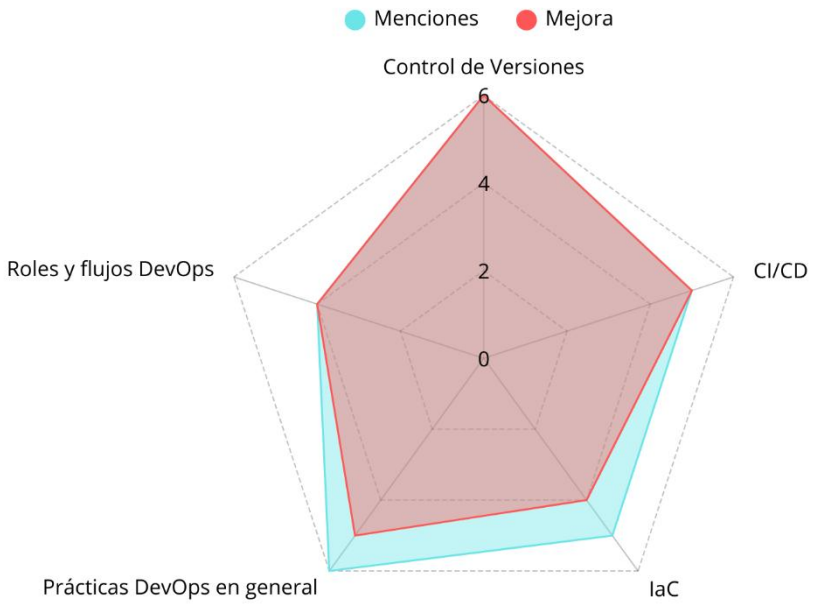


Figura 40. Cambios de nivel de compresión sobre DevOps

Los resultados muestran un incremento significativo en la comprensión del control de versiones, el despliegue continuo y el funcionamiento general de DevOps. El personal manifestó haber adquirido nociones más sólidas sobre el flujo DevOps, así como sobre la importancia de roles y procesos asociados.

Aunque conceptos como IaC siguen siendo percibidos como complejos debido a las limitaciones del entorno Oracle Apex, la guía permitió una mejora clara en su entendimiento. En general, se evidencia un aumento considerable del nivel de conocimiento técnico en DevOps dentro de la unidad de desarrollo.

4.1.6.8. Retroalimentación

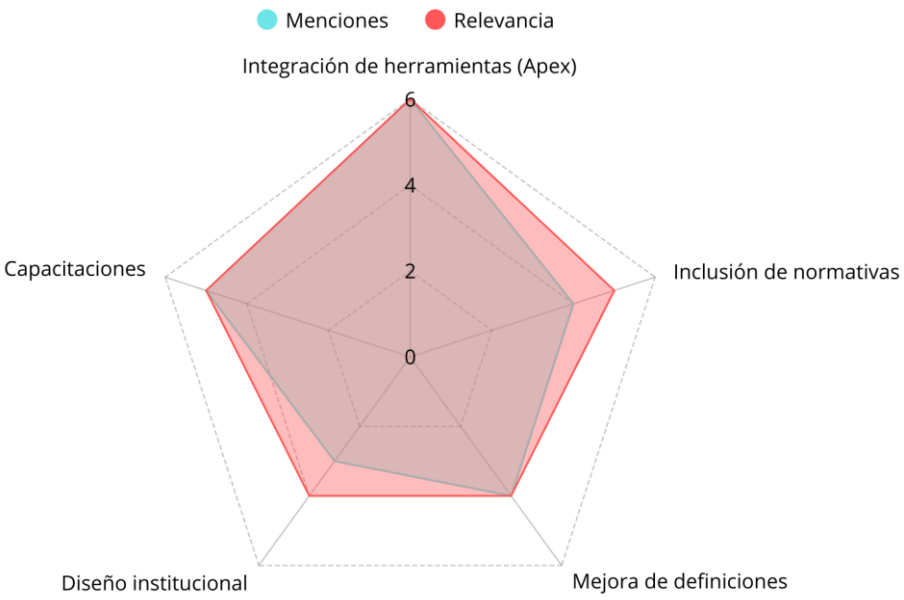


Figura 41. Retroalimentación

La retroalimentación recibida se concentró especialmente en la necesidad de ampliar el apartado de herramientas y clarificar la integración técnica con el entorno Oracle Apex, aspecto identificado como crítico para la implementación de prácticas DevOps. También se observó la importancia de incluir normativas legales y técnicas relevantes, así como mejoras en definiciones específicas.

Otros aportes se relacionan con la alineación visual al manual de marca institucional y la sugerencia de implementar procesos de capacitación o microcursos para facilitar la adopción de la guía en la unidad.

4.1.7. Optimización de la Guía a partir de las Percepciones del Personal Técnico y Directivo

4.1.7.1. Ajustes a las prácticas DevOps según su impacto prioritario

Las prácticas DevOps de mayor relevancia para el equipo de desarrollo son control de versiones, el feedback continuo y el despliegue continuo. En función de ello, la guía fue ajustada para fortalecer especialmente estos tres componentes, incorporando pasos operativos detallados, diagramas de flujo más explícitos y ejemplos adaptados al entorno de Oracle Apex.

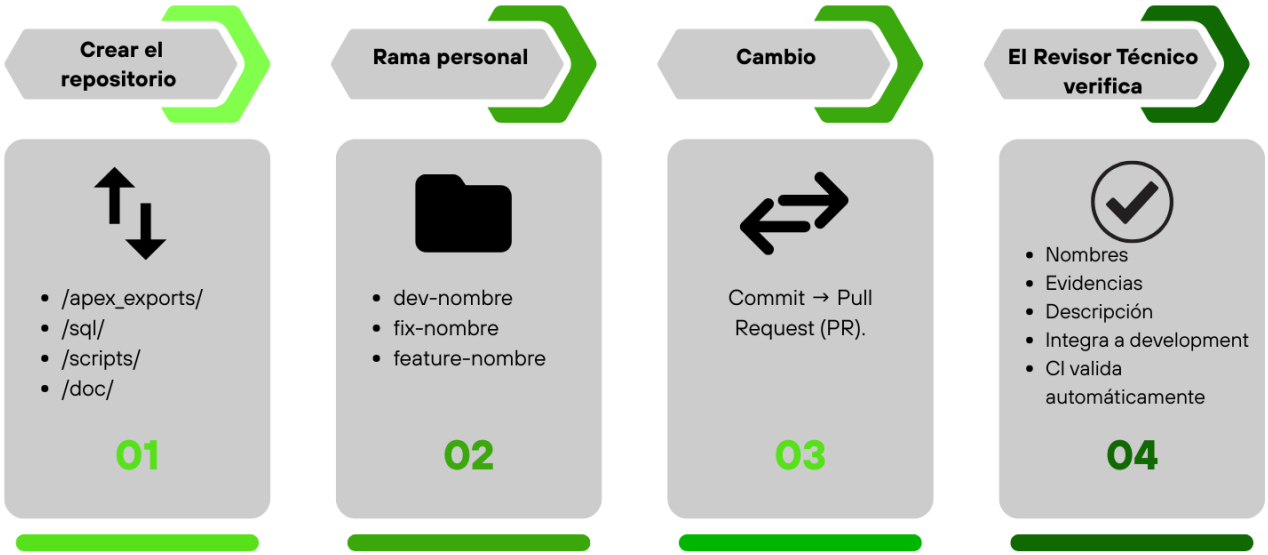


Figura 42. Aplicación de control de versiones

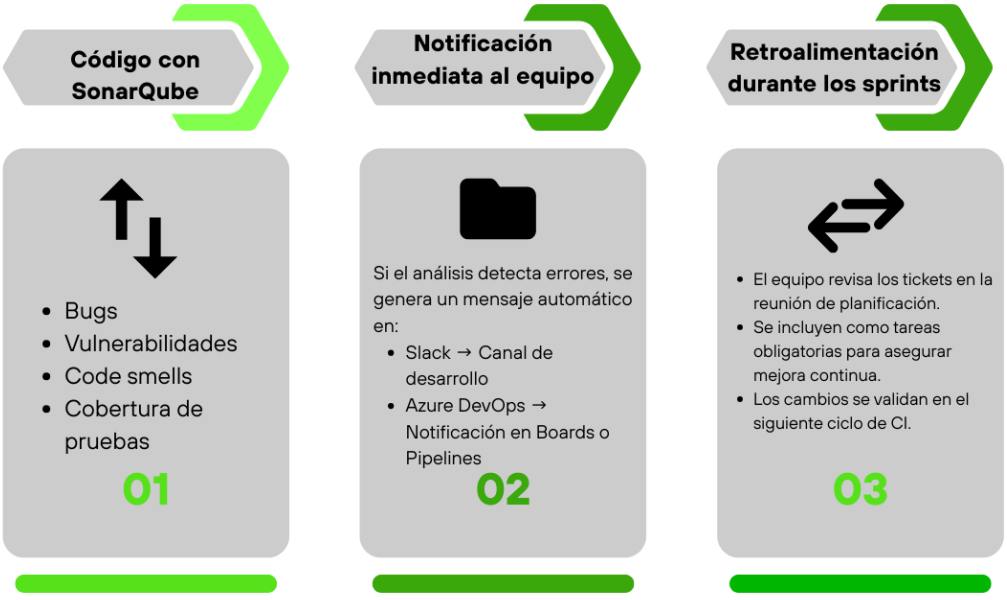


Figura 43. Aplicación para feedback continuo

Se reforzaron las secciones relacionadas con monitoreo y observabilidad y repositorios centralizados, debido a su potencial de mejora en trazabilidad y colaboración.

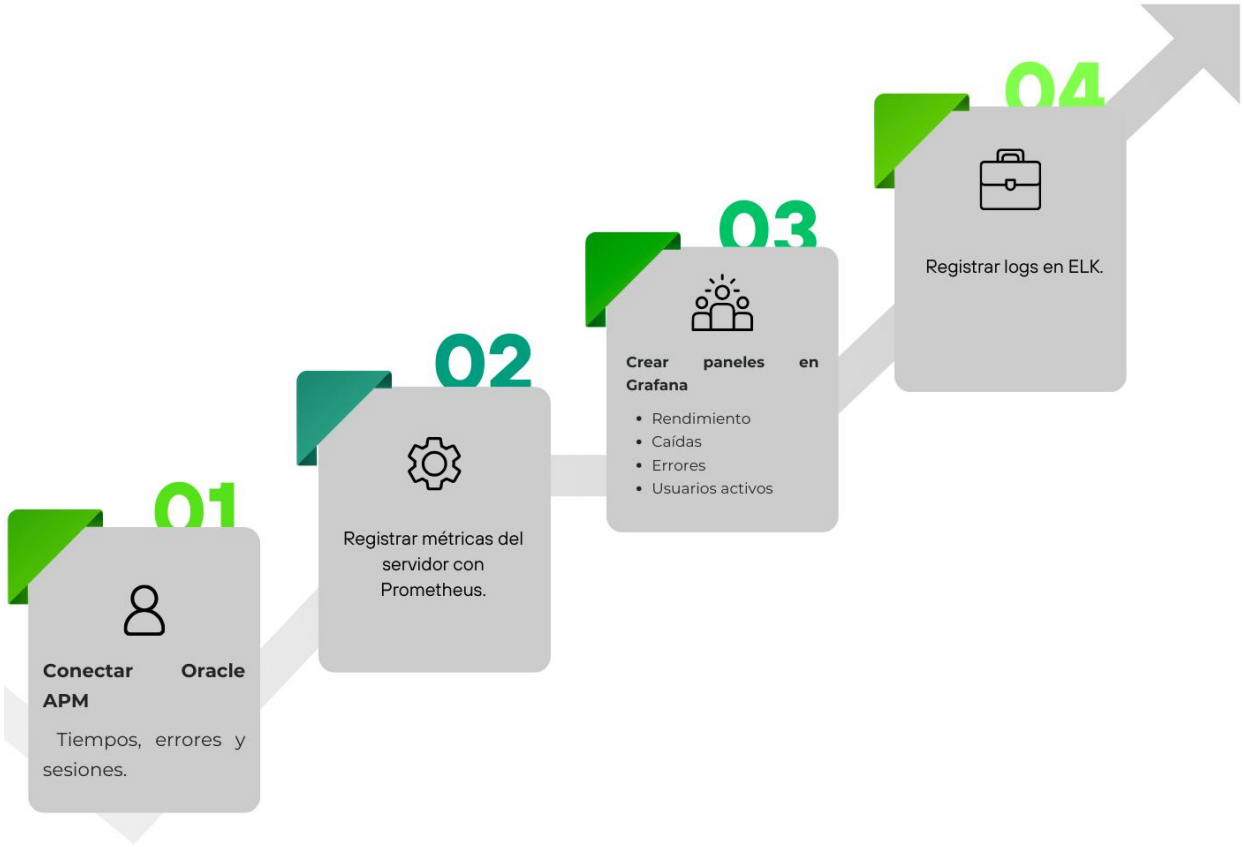


Figura 44. Aplicación para monitoreo

Asi la guía prioriza aquellas prácticas DevOps con mayor impacto inmediato y real dentro de los procesos de la UPEC, alineándose directamente con las necesidades identificadas por el personal técnico.

4.1.7.2. Fortalecimiento de conceptos técnicos

Varios conceptos técnicos fueron identificados como poco claros o difíciles de aplicar al contexto institucional, particularmente la integración entre Git y Oracle Apex, el uso de pipelines CI/CD e IaC. Por esta razón, se realizaron mejoras significativas en la guía, ampliando las explicaciones y simplificando el lenguaje técnico sin perder rigurosidad conceptual.

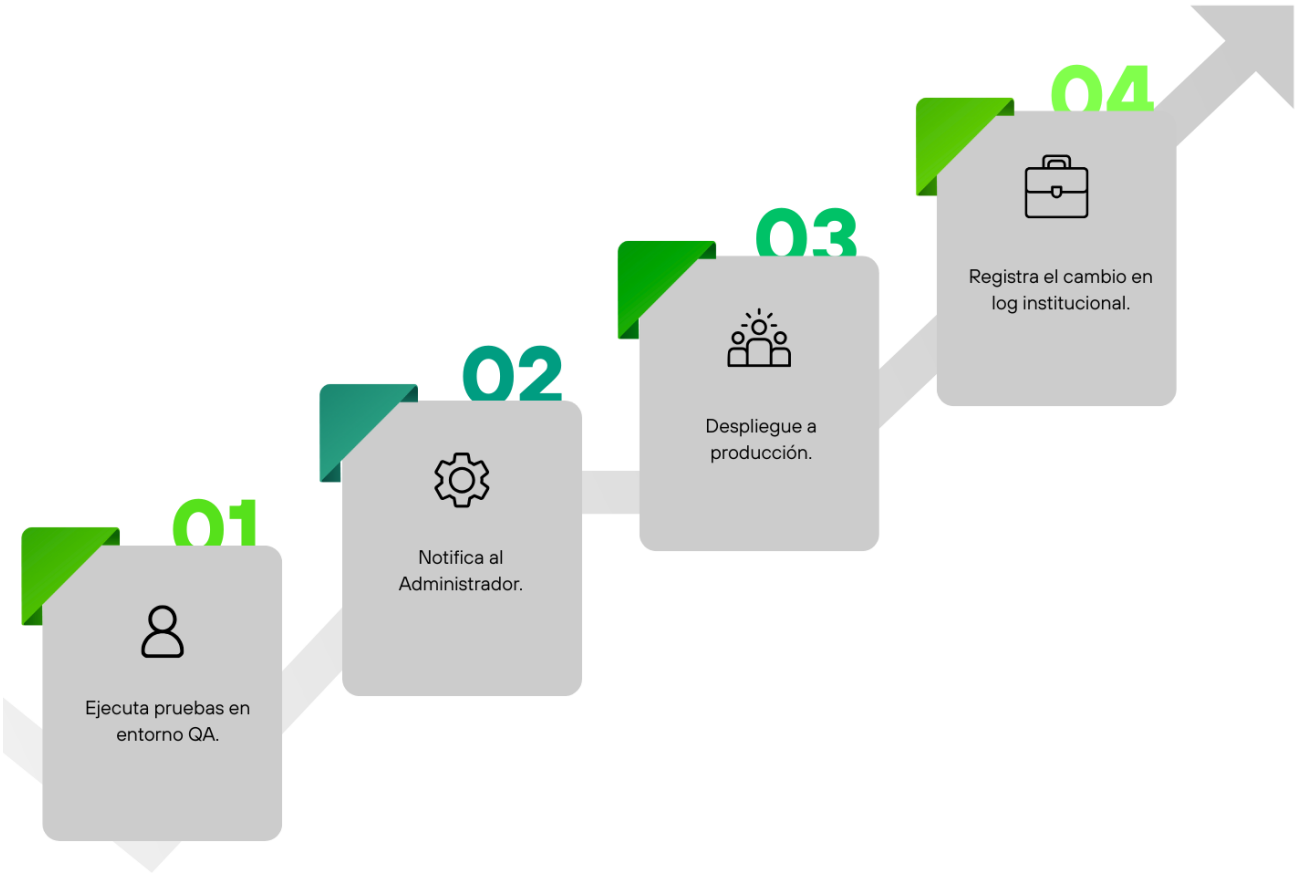
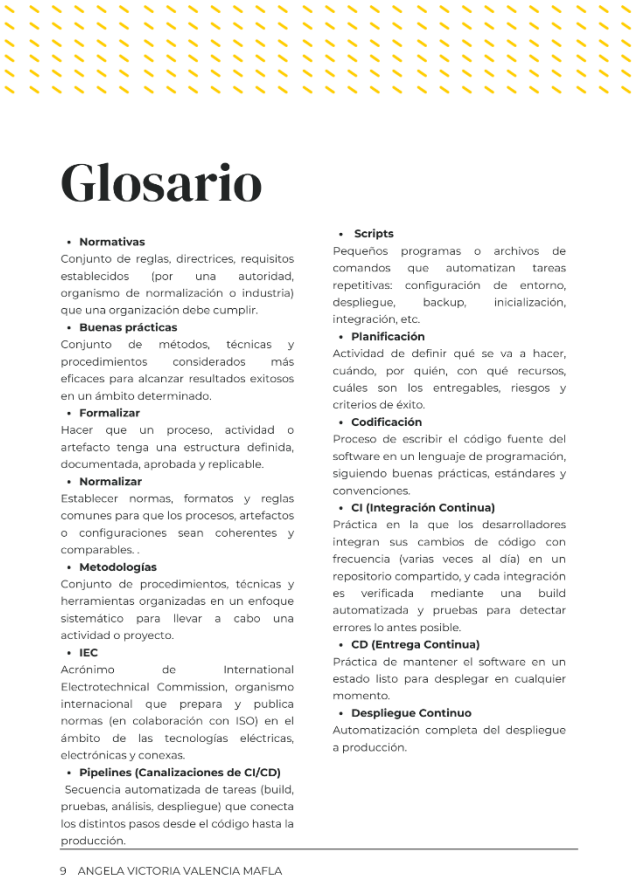


Figura 45. Aplicación de pipeline

Se pusieron ejemplos, diagramas más sencillos, glosario y pasos específicos de cómo adaptar las prácticas DevOps al entorno low-code de Apex. También se añadieron notas aclaratorias sobre la integración de herramientas como GitHub, GitLab, Azure DevOps, Jenkins, Terraform y Slack, con énfasis en cómo pueden ser utilizadas dentro de las limitaciones actuales de la UPEC.



Glosario

- **Normativas**

Conjunto de reglas, directrices, requisitos establecidos (por una autoridad, organismo de normalización o industria) que una organización debe cumplir.

- **Buenas prácticas**

Conjunto de métodos, técnicas y procedimientos considerados más eficaces para alcanzar resultados exitosos en un ámbito determinado.

- **Formalizar**

Hacer que un proceso, actividad o artefacto tenga una estructura definida, documentada, aprobada y replicable.

- **Normalizar**

Establecer normas, formatos y reglas comunes para que los procesos, artefactos o configuraciones sean coherentes y comparables. .

- **Metodologías**

Conjunto de procedimientos, técnicas y herramientas organizadas en un enfoque sistemático para llevar a cabo una actividad o proyecto.

- **IEC**

Acrónimo de International Electrotechnical Commission, organismo internacional que prepara y publica normas (en colaboración con ISO) en el ámbito de las tecnologías eléctricas, electrónicas y conexas.

- **Pipelines (Canalizaciones de CI/CD)**

Secuencia automatizada de tareas (build, pruebas, análisis, despliegue) que conecta los distintos pasos desde el código hasta la producción.

- **Scripts**

Pequeños programas o archivos de comandos que automatizan tareas repetitivas: configuración de entorno, despliegue, backup, inicialización, integración, etc.

- **Planificación**

Actividad de definir qué se va a hacer, cuándo, por quién, con qué recursos, cuáles son los entregables, riesgos y criterios de éxito.

- **Codificación**

Proceso de escribir el código fuente del software en un lenguaje de programación, siguiendo buenas prácticas, estándares y convenciones.

- **CI (Integración Continua)**

Práctica en la que los desarrolladores integran sus cambios de código con frecuencia (varias veces al día) en un repositorio compartido, y cada integración es verificada mediante una build automatizada y pruebas para detectar errores lo antes posible.

- **CD (Entrega Continua)**

Práctica de mantener el software en un estado listo para desplegar en cualquier momento.

- **Despliegue Continuo**

Automatización completa del despliegue a producción.

9 ANGELA VICTORIA VALENCIA MAFLA

Figura 46. Glosario de términos

4.1.7.3. Aplicabilidad al entorno UPEC

Se evidenció la necesidad de adaptar la guía a las condiciones reales de trabajo en la UPEC, considerando la plataforma Oracle Apex. Con base en ello, se realizaron modificaciones orientadas a prácticas DevOps que son totalmente aplicables, parcialmente aplicables o requieren ajustes adicionales.

La guía ahora incluye recomendaciones específicas para la aplicación de control de versiones, feedback continuo y monitoreo mediante Oracle APM.

4.1.7.4. Mejoras en el formato, diseño y estructura de la guía

La guía presentaba un formato claro y didáctico pero era necesario realizar ajustes en términos de diseño institucional, definiciones y diagramas. Se actualizó la estética general del documento para alinearla al manual de marca de la UPEC, ajustando colores, tipografías y estilos de encabezados.

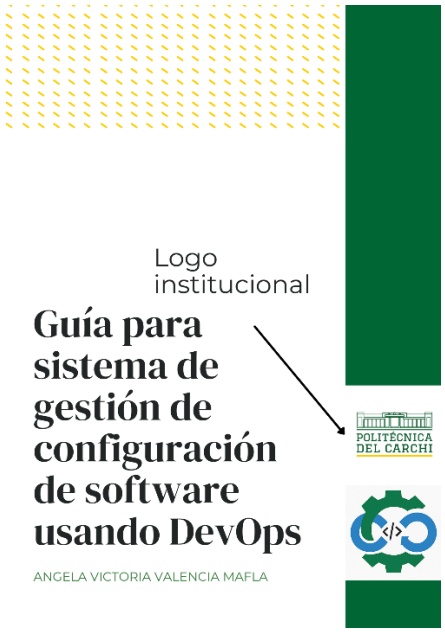


Figura 47. Uso de marca institucional

Figura 48. Uso de colores institucionales

Se reescribieron definiciones bajo los diagramas para la claridad, se reorganizaron secciones para facilitar la lectura y se añadió una estructura más progresiva que guía al lector paso a paso en cada práctica DevOps.

Figura 49. Uso de nota en herramientas CI/CD

Figura 50. Uso de nota en herramientas repositorios centralizados

4.2. DISCUSIÓN

Un sistema de gestión de configuración de software es crítico dentro de cualquier desarrollo de sistemas por ello se puede evidenciar que la falta de estandarización limita que haya madurez en los entornos complejos y también que haya una compleja integración con metodologías emergentes como DevOps es así Que los resultados obtenidos con la investigación es señalan diferentes confirmaciones puestas en los antecedentes como las hechas por Hochbergs & Nilsson Señalan que en la literatura se llega a mencionar la relación que existe sobre los sistemas de gestión de configuración de software y la práctica de desarrollo y operaciones dice que estas prácticas raramente se explican de manera práctica esta falta de claridad teórica se ve reflejada en los hallazgos que tenemos en los resultados ya que el 44.3% de los desarrolladores utilizan Gitflow mientras que un 33.5% no aplica ninguna estrategia de forma formal es así que se muestra que las prácticas estructuradas y métodos improvisados Coincide con lo que dicen los autores respecto a la ausencia de lineamientos y necesidades de especificar cómo se debe operar un sistema de gestión dentro del enfoque DevOps.

Dentro de la investigación de Espinel destacó que la gestión de configuración en los entornos complejos no tiene una visión integral y madura sobre la trazabilidad causando diferentes inconsistencias y en la gestión de cambio esta realidad se puede ver también dentro de la UPEC ya que se identifican limitaciones en la documentación falta de control de cambios y ausencia en registros formales que permitan producir versiones de software aunque los desarrolladores reconocen la importancia del versionamiento y la trazabilidad es parcial esto confirma la necesidad de un enfoque unificado.

También podemos ver que en la investigación de Bildirici argumenta que los entornos donde no hay adecuadas herramientas de automatización e integración continua son lentos e ineficientes lo cual evidencia que en la UPEC existe la disposición hacia herramientas de CI/CD pero que el 66.7% de los desarrolladores indican menos del 10% de su tiempo a configurar pipelines y a un a aún existe una fuerte dependencia de pruebas manuales esto también puede demostrar que la automatización y la integración continua está en una fase temprana tal como los autores describen para organizaciones con modelos tradicionale

V. CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- El análisis realizado permitió identificar que la UPEC tiene dificultades en la gestión de versiones y actualizaciones de los módulos virtuales de la unidad de desarrollo en el área de tics porque tiene escases de estandarización y dependencias a procesos manuales esto se ve como resultado en limitaciones afectan la trazabilidad el control de cambios y la identificación entre integridad de proyectos mediante el uso de prácticas DevOps se puede presentar una oportunidad para optimizar estos procesos automatizando e integrando la mejora continua y así también la el trabajo entre equipos de desarrollo y operaciones.
- Las herramientas como Git, Jenkins, sonarqube son herramientas tipo DevOps que se adecúan al entorno institucional ya que cuentan con compatibilidad con los sistemas ya existentes esta combinación va a permitir que las tecnologías posibiliten la implementación de un flujo más eficiente bajo el control de versiones y pruebas automatizadas y así se fortalece la calidad del software institucional.
- Se diseñó la guía metodológica basada en tres secciones principales que son el flujo de trabajo bajo git y ramas, la integración y pruebas y las herramientas de mejora que van orientadas hacia la creación de sistemas de gestión de configuración de software adaptándose a las necesidades de mejora de la UPEC también se pueden ver integradas herramientas y lineamientos técnicos que promueven la estandarización de procesos y contribuyen a una gestión más rápida y colaborativa entre los desarrolladores.

5.2. RECOMENDACIONES

- Las prácticas DevOps se deben implementar de forma constante pero progresiva haciendo uso de la gestión de versiones dentro de los módulos de la UPEC dando como prioridad la automatización de procesos críticos y la capacitación al personal técnico a fin de que se bajen los errores de manuales y asegurar que hay trazabilidad de cambio.
- Se recomienda que debe adoptarse herramientas interoperables como git y jenkins y deben integrarse bajo un mismo flujo de trabajo así se puede optimizar el control de versiones y consolidar los procesos de integración y entrega continua.
- Se puede abordar la guía metodológica desarrollada en esta investigación como un documento referente para promover el uso en los proyectos de software del UPEC además se aconseja revisar y actualizar la evolución de las herramientas aplicadas dentro del área de tics específicamente en la unidad de desarrollo.

VI. REFERENCIAS BIBLIOGRÁFICAS

Alejos, A., Luza Montero, C., & Valencia Vivas, M. (2024). DevOps y agilidad: Una revisión sistemática. *Revista Científica: BIOTECH AND ENGINEERING*, *4*(2). https://doi.org/10.52248/eb.Vol4Iss2.153

Alexandra, E., Segura, V., Verónica, ;, & Segura, T. V. (2024). Tendencias y prácticas actuales en Ingeniería del Software. *Revista Científica Multidisciplinar*. https://orcid.org/0000-0002-1440-

Aranguren, M., & Ruiz, A. (2023). *INCORPORACIÓN DE LA METODOLOGÍA DEVOPS EN LA CONSTRUCCIÓN DE SOFTWARE DE LA APLICACIÓN NAVEGA SEGURO*. https://hdl.handle.net/10983/30899

Bildirici, F., Keziban, A., & Codal, S. (2023). *DEVOPS AND AGILE METHODS INTEGRATED SOFTWARE CONFIGURATION MANAGEMENT: HAVELSAN EXPERIENCE*. https://doi.org/10.48550/arXiv.2306.13964

Chacha, F., Cordero, D., & Ramón, G. (2024). Desafíos y estrategias en la implementación de metodologías agiles en proyectos informáticos. *Runas. Journal of Education and Culture*, *5*(10), e240189. https://doi.org/10.46652/runas.v5i10.189

Chauhan, D., Jain, C., Singh, V., & Yadav, A. P. (2025). DESIGN AND IMPLEMENTATION OF A CI/CD DEVOPS PIPELINE FOR AUTOMATED AND CONTINUOUS SOFTWARE DEPLOYEMENT. *INTERNATIONAL JOURNAL OF DevOps*, *2*(1), 11–30. https://doi.org/10.34218/ijdo_02_01_002

Cubero, H. (2020). *GESTIÓN DE LA CONFIGURACIÓN DE SISTEMAS DE SOFTWARE*.

Cui, J. (2024). *The Role of DevOps in Enhancing Enterprise Software Delivery Success through R&D Efficiency and Source Code Management*.

Damarapati, A. (2025). CI/CD Best Practices: Building Reliable Pipelines. *European Journal of Computer Science and Information Technology*, *13*(10), 1–10. https://doi.org/10.37745/ejcsit.2013/vol13n10110

Diaz, J. (2024). *HACIA UN LENGUAJE DE PATRONES DEVOPS PARA LAS STARTUPS COLOMBIANAS*. https://hdl.handle.net/1992/75306

Epa, U. (2022). *Configuration Management Procedure*.

Espinel Mena, G. P., Carrillo Medina, J. L., Galarza, E. E., & Matias Urbieta, M. (2024). A Systematic Mapping of Configuration Management Activities in Software Product Line. *Journal of Universal Computer Science*, *30*(11), 1484–1510. https://doi.org/10.3897/jucs.110887

García, G., Peña Olivero, N., & Avila George, H. (2024). DevOps and Software Product Quality Measurement: Preliminary Findings. *RISTI - Revista Iberica de Sistemas e Tecnologias de Informacao*, *2024*(53), 37–52. https://doi.org/10.17013/risti.53.37-52

Gordon, R., & Delgado, C. (2023). *INGENIERÍA DE SOFTWARE. APORTES DE LA METODOLOGÍA LEAN PARA SISTEMAS DE INFORMACIÓN EN PANAMÁ*. https://orcid.org/0000-0001-8468-4910

Gul, L., Imtiaz, S., Munir, S., Munir, M., & Khan, M. A. (2021). Integrated Traceability Approach for an Effective Impact Analysis. *Mehran University Research Journal of Engineering and Technology*, *40*(2), 346–357. https://doi.org/10.22581/muet1982.2102.09

Hochbergs, E., & Nilsson, L. (2020). *Software Connguration Management in a DevOps context*. https://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=9024735&fileOId=9024736

Jiménez, D., Zambrano, C., Negrin, E., & Zambrano, I. (2024). Diseño de un sistema de gestión por procesos para la Asociación de Mujeres Comunitarias del cantón Tosagua, Ecuador. *Revista San Gregorio*, *1*(59), 72–78. https://doi.org/10.36097/rsan.v1i59.3188

Lagos, N., Gómez, L., Londoño, D. C., Moreno, I. C., & Camacho Camacho, H. (2023). *Metodología para la integración de sistemas de gestión: revisión de literatura*. *15*. https://doi.org/10.15332/24631140

Lares, G., Romero, B., Lara, I., Lara, J., & Canobbio, C. (2023). *Ecosistemas Organizacionales*.

Luján, L. (2024). *Tópicos de ingeniería de software y sistemas de información*.

Manzano, I., & Ripoll, J. (2022). *La vigencia del modelo Kotter en la gestión del cambio: un análisis empírico*.

Marques, P., & Correia, F. F. (2023). *Foundational DevOps Patterns*. http://arxiv.org/abs/2302.01053

Mollo, J., Lázaro, R., & Crespo, R. (2023). Implementación de Nuevas Tecnologías de Información y Comunicación para la Educación Superior: Revisión sistemática. *Revista Científica Ciencia & Sociedad*, *3*.

Montilva, J., & Barrios, J. (2021). *Ingeniería del Software: Un enfoque basado en procesos*.

Morris, K. (2021). *Infrastructure as Code Dynamic Systems for the Cloud Age*.

Muñoz, D. A., Ordóñez, H., & Bucheli, V. (2020). Guideline to implement the CALMS model (DevOps) at mipymes in the software development organizations of the South of Colombia. *Revista Guillermo de Ockham*, *18*(1), 81–91. https://doi.org/10.21500/22563202.4270

Ochoa, L., Degueule, T., Falleri, J.-R., & Vinju, J. (2021). *Semantic versioning and impact of breaking changes in Maven Central*. https://hal.archives-ouvertes.fr/hal-03378089

Offerman, T., Blinde, R., Stettina, C. J., & Visser, J. (2022). *A Study of Adoption and Effects of DevOps Practices*. https://doi.org/10.1109/ICE/ITMC-IAMOT55089.2022.10033313

Olarte, M., Flores Mayta, D. J., Rios Vera, K. J., Quispe Ambrocio, A. D., & Seguil-Ormeño, N. A. (2023). Tecnologías de la Información y Comunicación (TIC) en la gestión empresarial: Un análisis cienciométrico. *Revista de Investigación en Comunicación y Desarrollo*, *14*(4), 388–400. https://doi.org/10.33595/2226-1478.14.4.899

Oluwatoyin, F., Azeez Olanipekun, H., Olubukola, R. A., Ololade, G. F., & Monisola, O. (2023). CONFIGURATION MANAGEMENT IN THE MODERN ERA: BEST PRACTICES, INNOVATIONS, AND CHALLENGES. *Computer Science & IT Research Journal*, *4*(2), 140–157. https://doi.org/10.51594/csitrj.v4i2.613

Omol, E. J. (2024). Organizational digital transformation: from evolution to future trends. En *Digital Transformation and Society* (Vol. 3, Número 3, pp. 240–256). Emerald Publishing. https://doi.org/10.1108/DTS-08-2023-0061

Pan, Z., Shen, W., Wang, X., Yang, Y., Chang, R., Liu, Y., Liu, C., Liu, Y., & Ren, K. (2024). *Ambush from All Sides: Understanding Security Threats in Open-Source Software CI/CD Pipelines*. https://doi.org/10.1109/TDSC.2023.3253572

Paul, J., Ueno, A., Dennis, C., Alamanos, E., Curtis, L., Foroudi, P., Kacprzak, A., Kunz, W. H., Liu, J., Marvi, R., Nair, S. L. S., Ozdemir, O., Pantano, E., Papadopoulos, T., Petit, O., Tyagi, S., & Wirtz, J. (2024). Digital transformation: A multidisciplinary perspective and future research agenda. *International Journal of Consumer Studies*, *48*(2). https://doi.org/10.1111/ijcs.13015

Peralta, L., Gaona Portal, M. del P., Luna Acuña, M. L., & Bazán Linares, M. V. (2023). Las tecnologías de la información y la comunicación (TIC) en educación secundaria: Una revisión sistemática. *Revista Andina de Educación*, *7*(1), 000711. https://doi.org/10.32719/26312816.2023.7.1.1

Reis, J., & Melão, N. (2023). Digital transformation: A meta-review and guidelines for future research. En *Heliyon* (Vol. 9, Número 1). Elsevier Ltd. https://doi.org/10.1016/j.heliyon.2023.e12834

Rincón, H., & Gómez, D. (2023). *Cambio y aprendizaje organizacional, revisión documental*.

Rodríguez, R. (2021). *DISEÑO E IMPLEMENTACIÓN DE UN PROCESO DE GESTIÓN DE LA CONFIGURACIÓN PARA UN BANCO*. https://repositorio.uchile.cl/bitstream/handle/2250/181949/Dise%C3%B1o-e-implementacion-de-un-proceso-de-gestion-de-la-configuracion-para-un-banco.pdf?sequence=1

Ruales, I. (2024). *ANÁLISIS, DISEÑO Y DESARROLLO DE UN SISTEMA DE AUTOMATIZACIÓN DE LA GESTIÓN DOCUMENTAL EN EMPRESAS A TRAVÉS DE UNA INTERFAZ GRÁFICA INTUITIVA*. http://dspace.ups.edu.ec/handle/123456789/27200

Shadab, N., Cody, T., Salado, A., & Beling, P. (2023). *A Systems-Theoretical Formalization of Closed Systems*. http://arxiv.org/abs/2311.10786

Shchirov, P. (2022). *GIT VERSION CONTROL SYSTEM*. https://tproger.ru/translations/beginner-git-cheatshet

Vera, J., Reinoso, J., & Ramírez, E. (2023). *GESTIÓN DE LA CULTURA Y CAMBIO ORGANIZACIONAL*. https://www.researchgate.net/publication/368984248

Villa, M., Centeno, C., Gavilanez, A., & Cadena, J. (2023). Herramientas para el desarrollo de software. Una revisión. *Revista Científica de Informática ENCRIPTAR.*, *6*.

Villacres, S. (2023). *Guía de aplicación del proceso de gestión de la configuración de software*. http://bibdigital.epn.edu.ec/handle/15000/24478

Zúñiga, M. R., Hurtado, G. P., & Pérez, J. M. (2020). La capacidad de la gestión de la configuración en las pequeñas organizaciones de desarrollo de software. *ConcienciaDigital*, *3*(3), 466–485. https://doi.org/10.33262/concienciadigital.v3i3.1339

VII. ANEXOS

Anexo 1. Certificado del abstract por parte de CIDEN

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI FOREIGN AND NATIVE LANGUAGES CENTER

ABSTRACT- EVALUATION SHEET

NAME: Angela Victoria Valencia Mafla
DATE: Martes, 16 de diciembre de 2025
Topic: "Sistema de gestión de configuración de software usando DevOps."
MARKS AWARDED **QUANTITATIVE AND QUALITATIVE**

VOCABULARY AND WORD USE	Use new learnt vocabulary and precise words related to the topic	Use a little new vocabulary and some appropriate words related to the topic	Use basic vocabulary and simplistic words related to the topic	Limited vocabulary and inadequate words related to the topic
	EXCELLENT: 2 ☐	GOOD: 1,5 ■	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
WRITING COHESION	Clear and logical progression of ideas and supporting paragraphs.	Adequate progression of ideas and supporting paragraphs.	Some progression of ideas and supporting paragraphs.	Inadequate ideas and supporting paragraphs.
De	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
ARGUMENT	The message has been communicated very well and identify the type of text	The message has been communicated appropriately and identify the type of text	Some of the message has been communicated and the type of text is little confusing	The message hasn't been communicated and the type of text is inadequate
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
CREATIVITY	Outstanding flow of ideas and events	Good flow of ideas and events ■	Average flow of ideas and events	Poor flow of ideas and events
	EXCELLENT: 2 ☐	GOOD: 1,5	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
SCIENTIFIC SUSTAINABILITY	Reasonable, specific and supportable opinion or thesis statement	Minor errors when supporting the thesis statement	Some errors when supporting the thesis statement	Lots of errors when supporting the thesis statement
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
TOTAL/AVERAGE	9 - 10: EXCELLENT 7 - 8,9: GOOD 5 - 6,9: AVERAGE 0 - 4,9: LIMITED	TOTAL 9		

1

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI- FOREIGN AND NATIVE LANGUAGES CENTER

Informe sobre el Abstract de Artículo Científico o Investigación.

Autor: Angela Victoria Valencia Mafla
Fecha de recepción del abstract: Lunes, 15 de diciembre de 2025
Fecha de entrega del informe: Martes, 16 de diciembre de 2025

El presente informe validará la traducción del idioma español al inglés si alcanza un porcentaje de: 9 – 10 Excelente.

Si la traducción no está dentro de los parámetros de 9 – 10, el autor deberá realizar las observaciones presentadas en el ABSTRACT, para su posterior presentación y aprobación.

Observaciones:

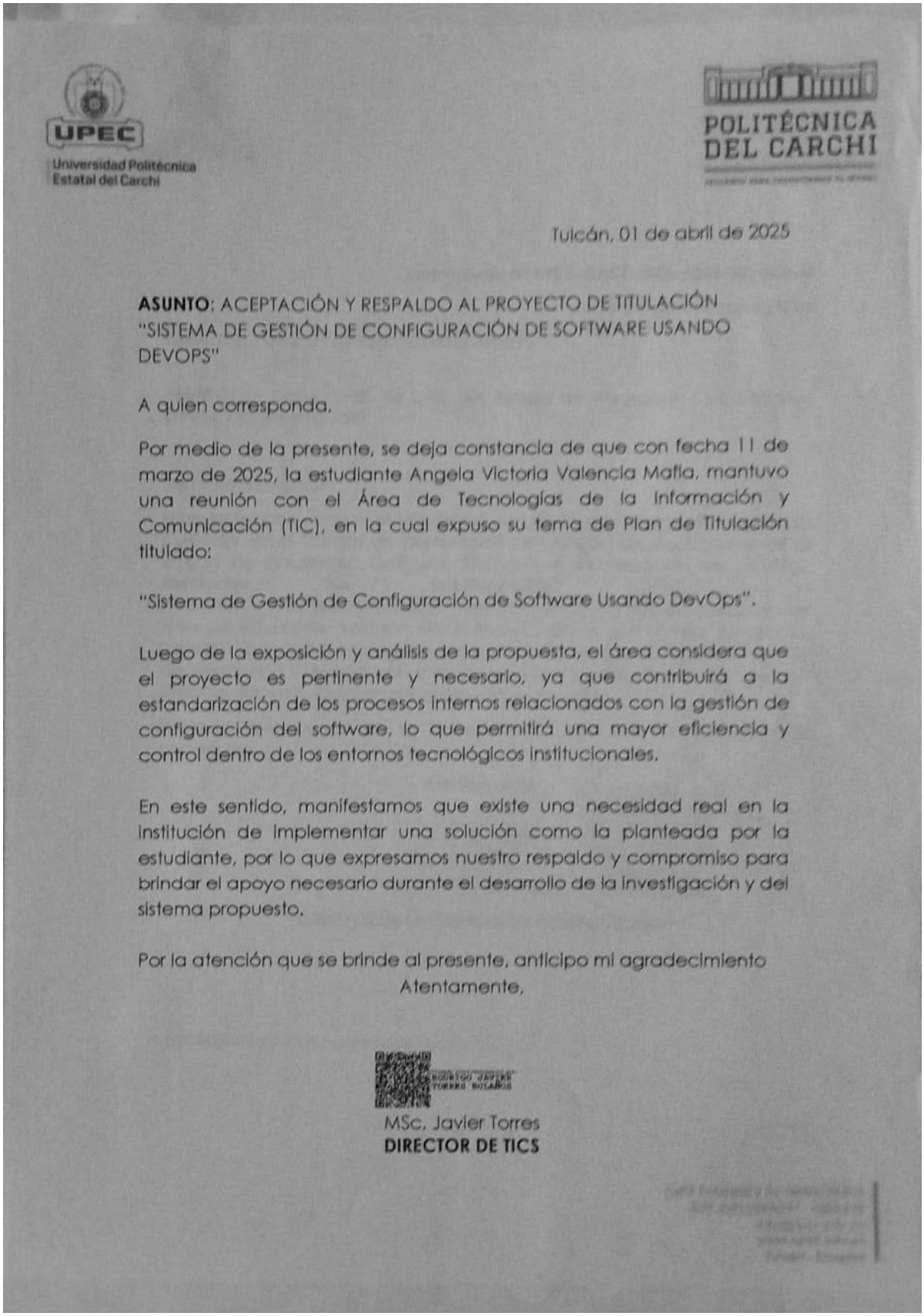
Después de realizar la revisión del presente abstract, éste presenta una apropiada traducción sobre el tema planteado en el idioma Inglés. Según la rúbrica de evaluación de la traducción en Inglés, ésta alcanza un valor de 9; por lo cual se valida dicho trabajo.

Atentamente

Firmado electrónicamente por: MARTHA ARACELLY VIVEROS ALMEIDA
Validar únicamente con FirmaEC

MA. Martha Viveros
Responsable del
CIDEN

Anexo 2. Acta de aceptación de proyecto



UPEC
Universidad Politécnica Estatal del Carchi

POLITÉCNICA DEL CARCHI

Tulcán, 01 de abril de 2025

ASUNTO: ACEPTACIÓN Y RESPALDO AL PROYECTO DE TITULACIÓN "SISTEMA DE GESTIÓN DE CONFIGURACIÓN DE SOFTWARE USANDO DEVOPS"

A quien corresponda.

Por medio de la presente, se deja constancia de que con fecha 11 de marzo de 2025, la estudiante Angela Victoria Valencia Mafla, mantuvo una reunión con el Área de Tecnologías de la Información y Comunicación (TIC), en la cual expuso su tema de Plan de Titulación titulado:

"Sistema de Gestión de Configuración de Software Usando DevOps".

Luego de la exposición y análisis de la propuesta, el área considera que el proyecto es pertinente y necesario, ya que contribuirá a la estandarización de los procesos internos relacionados con la gestión de configuración del software, lo que permitirá una mayor eficiencia y control dentro de los entornos tecnológicos institucionales.

En este sentido, manifestamos que existe una necesidad real en la institución de implementar una solución como la planteada por la estudiante, por lo que expresamos nuestro respaldo y compromiso para brindar el apoyo necesario durante el desarrollo de la investigación y del sistema propuesto.

Por la atención que se brinde al presente, anticipo mi agradecimiento

Atentamente,

MSc. Javier Torres
DIRECTOR DE TICS

Anexo 3. Entrevista inicial

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

ENTREVISTA

1. DATOS INFORMATIVOS

1.1.Estudiante: Valencia Mafla Angela Victoria

1.2.Entrevistado: Wilson Andrés Zabala Villareal

1.3.Objetivo: Recopilar información clave sobre los procesos, herramientas, problemas y expectativas del área de desarrollo de software de la UPEC, para un sistema de gestión de configuración de software usando DevOps

2. INTRODUCCIÓN

Estimado,

El objetivo de esta entrevista es conocer más a fondo cómo actualmente se gestiona la configuración de software en el área de desarrollo, qué problemas enfrentan y cuáles son las expectativas respecto a la implementación de buenas prácticas y herramientas que ayuden a mejorar la calidad, la seguridad y la continuidad en el desarrollo. La información recopilada servirá como insumo para mi trabajo de titulación relacionado con un sistema de gestión de la configuración de software usando DevOps. No hay respuestas correctas o incorrectas, lo importante es conocer la realidad del área.

3. DESARROLLO

3.1. Organización y roles

1. ¿Cómo está conformado actualmente el equipo de desarrollo de software?
2. ¿Existen roles definidos para cada integrante o todos cumplen funciones similares?
3. ¿Cómo se coordina el trabajo entre los diferentes miembros del equipo?
4. ¿Qué metodologías de trabajo han utilizado hasta ahora (XP, Scrum, otros)?

3.2. Procesos y documentación

1. ¿Cuentan con procesos documentados para la gestión del software?
2. ¿Existen manuales, actas o guías que se puedan consultar?

3. ¿Cómo se gestionan los requisitos de software y los cambios que solicita el usuario final?
4. ¿Qué tan frecuente es la actualización de la documentación de proyectos?

3.3. Herramientas y servidores

1. ¿Qué herramientas utilizan actualmente para el control de versiones?
2. ¿Por qué no se usa GitLab, aunque tienen un servidor configurado?
3. ¿Se emplean otras herramientas de integración o automatización en el desarrollo?
4. ¿Qué sistemas de bases de datos manejan en los proyectos (Oracle, MySQL, otros)?

3.4. Gestión de configuración y versiones

1. ¿Cómo realizan el versionado de las aplicaciones y bases de datos?
2. ¿Existen registros formales de las versiones de los proyectos?
3. ¿Han tenido problemas al migrar proyectos de un equipo a otro?
4. ¿Qué dificultades se presentan al gestionar cambios en el software?

3.5. Problemas actuales

1. ¿Qué problemas considera más críticos en la gestión de la configuración de software?
2. ¿Han tenido pérdidas de páginas o funcionalidades al realizar cambios?
3. ¿Qué tan frecuentes son los incidentes relacionados con la base de datos o los backups?
4. ¿Cómo se manejan los incidentes cuando se presentan?

3.6. DevOps y automatización

1. ¿Tienen conocimiento o experiencia previa con metodologías DevOps?
2. ¿Cómo perciben la posibilidad de implementar CI/CD en el área de desarrollo?
3. ¿Qué beneficios creen que aportaría adoptar DevOps en sus procesos?

4. ¿Qué limitaciones o barreras anticipan para aplicar DevOps en el equipo actual?

3.7. Expectativas y mejoras

1. ¿Qué esperan lograr con un sistema de gestión de la configuración más organizado?
2. ¿Cómo creen que se puede mejorar la calidad y seguridad de las aplicaciones?
3. ¿Qué esperan respecto a la reducción de errores e incidentes?
4. ¿Qué expectativas tienen sobre la integración de usuarios finales en los procesos?

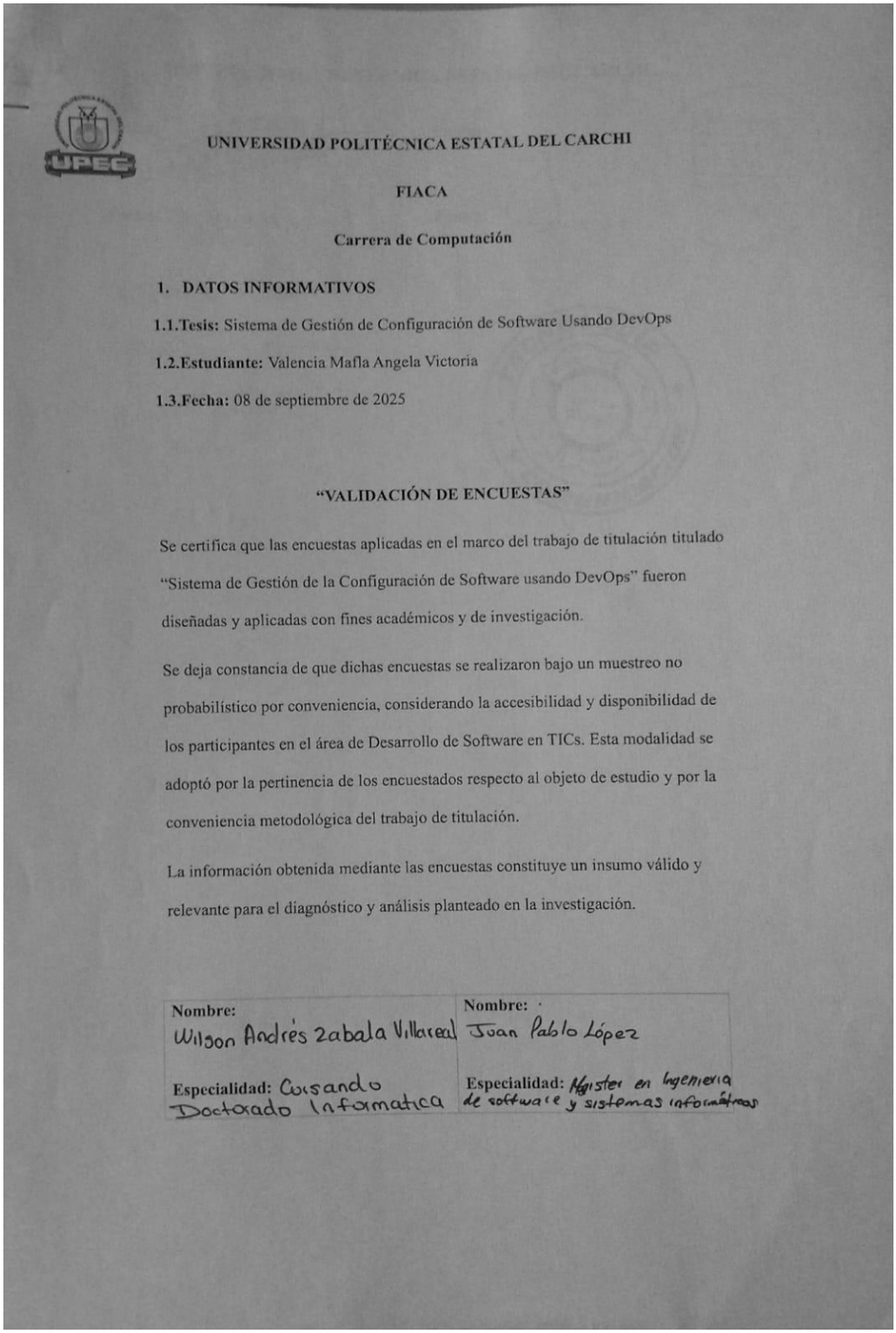
Wilson Andrés Zabala Villareal

Coordinador de desarrollo de Software

UPEC

POLITÉCNICA DEL CARCHI
Unidad de Desarrollo de Software y Análisis de Datos

Anexo 4. Validación de instrumentos (Encuesta)



UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FIACA

Carrera de Computación

1. DATOS INFORMATIVOS

1.1.Tesis: Sistema de Gestión de Configuración de Software Usando DevOps

1.2.Estudiante: Valencia Mafla Angela Victoria

1.3.Fecha: 08 de septiembre de 2025

"VALIDACIÓN DE ENCUESTAS"

Se certifica que las encuestas aplicadas en el marco del trabajo de titulación titulado "Sistema de Gestión de la Configuración de Software usando DevOps" fueron diseñadas y aplicadas con fines académicos y de investigación.

Se deja constancia de que dichas encuestas se realizaron bajo un muestreo no probabilístico por conveniencia, considerando la accesibilidad y disponibilidad de los participantes en el área de Desarrollo de Software en TICs. Esta modalidad se adoptó por la pertinencia de los encuestados respecto al objeto de estudio y por la conveniencia metodológica del trabajo de titulación.

La información obtenida mediante las encuestas constituye un insumo válido y relevante para el diagnóstico y análisis planteado en la investigación.

Nombre: Wilson Andrés Zabala Villareal	**Nombre:** Juan Pablo López
Especialidad: Cursando Doctorado Informatica	**Especialidad:** Magister en Ingeniería de software y sistemas informáticos

Carrera de Computación

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

Firma:	**Firma:**
Fecha: 08-09-2025	**Fecha:** 09-09-2025

POLITÉCNICA DEL CARCHI
DIRECCIÓN DE TIC

Wilson. Ortega Herrera
Tecnología y Comunicación

TECNOLOGÍA Y COMUNICACIÓN
BOMBEROS TULCÁN

Anthony Quisaguano

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

RECOLECCIÓN DE DATOS

1. DATOS INFORMATIVOS

1.1.Estudiante: Angela Victoria Valencia Mafla

1.2.Fecha: 10 de septiembre 2025

1.3.Tesis: Sistema de Gestión de Configuración de Software Usando DevOps

1.4.Objetivo: Diagnosticar el estado actual de la gestión de configuración de software y la adopción de prácticas DevOps en la UPEC.

"DESARROLLO DE SISTEMAS"

2. INTRODUCCIÓN

Estimado/a Desarrollador/a,

Esta encuesta tiene por objetivo conocer tus prácticas diarias en control de versiones, pruebas y despliegues, así como tu experiencia con herramientas DevOps. Tus respuestas guiarán las recomendaciones técnicas de la guía.

3. DESARROLLO

3.1. Sección A: Flujo de trabajo Git y ramas

1. ¿Qué estrategia de branching (ramificado) usan en tu equipo?

Git Flow	X
Trunk-based development	
Feature branching	
Ninguno	
Otro:	

2. En una escala del 1 al 5, ¿qué tan clara es la política de commits (mensajes, estructura)?

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

1 = Nada clara 5 = Muy clara

1	2	3	4	5
			X	

3. ¿Con qué frecuencia realizas pull requests (solicitudes de integración) o merge requests (fusión de ramas)?

Varias veces al día	
Diario	X
Semanal	
Mensual	

3.2. Sección B: Integración y pruebas

1. ¿Tu equipo ejecuta pruebas automatizadas antes de integrar código?

Sí, unitarias y de integración	
Sólo unitarias	X
Sólo manuales	
No	

2. En escala del 1 al 5, ¿qué tan rápido detectan errores gracias a la integración continua (CI)?

1 = Muy lento 5 = Muy rápido

1	2	3	4	5
			X	

3. ¿Qué porcentaje aproximado de tu tiempo de desarrollo dedicas a configurar o mantener pipelines (procesos de CI/CD)?

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

0–10 %	
10–25 %	X
25–50 %	
> 50 %	

3.3.Sección C: Herramientas y mejoras

1. En una escala del 1 al 5, ¿con qué grado de satisfacción evalúas las herramientas CI/CD (integración y entrega continua) actuales?

1 = Muy insatisfecho 5 = Muy satisfecho

1	2	3	4	5
		X		

2. ¿Qué funcionalidad consideras más necesaria para agilizar tu trabajo?

Despliegue más rápido	
Mejor gestión de secretos	
Integración con testing automatizado	X
Monitoreo en tiempo real	

Anexo 5. Validación de instrumentos (Checklist)

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FIACA

Carrera de Computación

1. DATOS INFORMATIVOS

1.1.Tesis: Sistema de Gestión de Configuración de Software Usando DevOps

1.2.Estudiante: Valencia Mafla Angela Victoria

1.3.Fecha: 8 de septiembre de 2025

"Checklist para la Guía del Sistema de Gestión de Configuración de Software Usando DevOps"

2. INTRODUCCIÓN

Este documento presenta un checklist diagnóstico, elaborado con base en las respuestas obtenidas en entrevistas y encuestas aplicadas al personal clave del área TIC, incluyendo autoridades, coordinadores y programadores. Su propósito es identificar el estado actual de la gestión de configuración de software en la UPEC y establecer un punto de partida para el diseño e implementación de una guía basada en prácticas DevOps, que permita modernizar y automatizar los procesos de control de versiones, trazabilidad de cambios y actualizaciones de sistemas.

La estructura del checklist está alineada con la norma ISO/IEC 26563:2022 – Software and systems engineering: Methods and tools for product line configuration management, que especifica procesos, capacidades metodológicas y soporte de herramientas para la gestión de configuración en contextos de familias de productos o sistemas.

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

3. OBJETIVO

Diagnosticar el estado actual de la gestión de configuración de software en la Universidad Politécnica Estatal del Carchi, como base para el diseño de una guía que permita su implementación bajo principios DevOps.

4. CHECKLIST DE DIAGNÓSTICO

C: CUMPLE P: PARCIAL NC: NO CUMPLE

Nº	Requisito	Nivel de Necesidad	Estado Actual	Observaciones
1	Identificación de todos los módulos virtuales y sus versiones	Alta	P	Identificación manual, sin trazabilidad
2	Uso de herramientas de control de versiones (Git, GitLab, etc.)	Alta	NC	No se usan herramientas formales
3	Documentación de cambios realizados en los sistemas (historial de versiones)	Alta	NC	Cambios no se documentan o se hacen de forma informal
4	Flujo definido para solicitud, aprobación e implementación de cambios	Alta	P	Existe flujo informal, sin estandarización

5	Existencia de repositorio centralizado para código y documentación técnica	Alta	NC	Información dispersa en múltiples lugares
6	Definición de roles y responsabilidades para la gestión de configuración	Media	P	Tareas asignadas de forma no formal
7	Realización de respaldos periódicos y restauración en caso de fallos	Media	NC	No existe protocolo ni herramientas para backup
8	Capacitación al personal TIC sobre DevOps, CI/CD y control de versiones	Alta	NC	Conocimiento general, sin formación técnica
9	Uso de metodologías ágiles o híbridas en los proyectos TIC	Media	P	Métodos mixtos sin formalización
10	Auditoría interna sobre versiones, cambios y configuraciones	Media	NC	No se realizan auditorías
11	Selección y uso de herramientas DevOps según las capacidades institucionales	Alta	P	Conocimiento superficial de herramientas

12	Adaptación de soluciones al entorno técnico actual de la UPEC	Media	C	Infraestructura general es compatible
13	Registro y trazabilidad de cambios por usuario y fecha	Alta	NC	No hay sistema de seguimiento
14	Integración continua o despliegue automatizado	Alta	NC	Procesos totalmente manuales

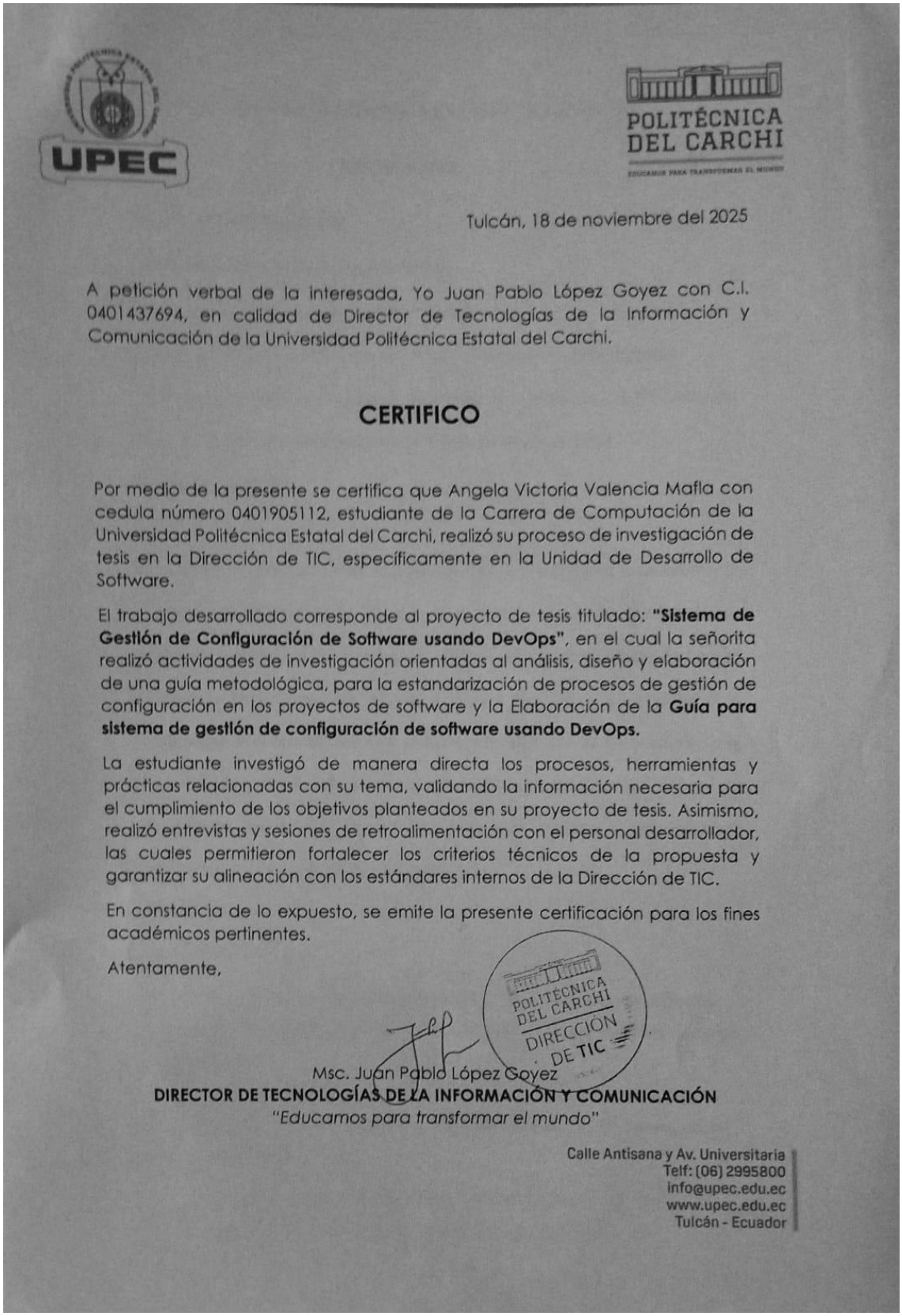
5. Validación

Nombre: Marco Yandún	**Nombre:** Andrés Hidalgo
Especialidad: Docente Computación	**Especialidad:** Docente de Computación
Firma:	**Firma:**
Fecha: 8-Sep-2025	**Fecha:** 08/09/2025

Wilson Ortiz Hurtado
Tecnología en Comunicación

TECNOLOGÍA Y COMUNICACIÓN
BOMBEROS TULCÁN

Anexo 6. Certificado dado por TIC aprobación de guía



UPEC

POLITÉCNICA DEL CARCHI

Tulcán, 18 de noviembre del 2025

A petición verbal de la interesada, Yo Juan Pablo López Goyez con C.I. 0401437694, en calidad de Director de Tecnologías de la Información y Comunicación de la Universidad Politécnica Estatal del Carchi.

CERTIFICO

Por medio de la presente se certifica que Angela Victoria Valencia Mafla con cedula número 0401905112, estudiante de la Carrera de Computación de la Universidad Politécnica Estatal del Carchi, realizó su proceso de investigación de tesis en la Dirección de TIC, específicamente en la Unidad de Desarrollo de Software.

El trabajo desarrollado corresponde al proyecto de tesis titulado: **"Sistema de Gestión de Configuración de Software usando DevOps"**, en el cual la señorita realizó actividades de investigación orientadas al análisis, diseño y elaboración de una guía metodológica, para la estandarización de procesos de gestión de configuración en los proyectos de software y la Elaboración de la **Guía para sistema de gestión de configuración de software usando DevOps.**

La estudiante investigó de manera directa los procesos, herramientas y prácticas relacionadas con su tema, validando la información necesaria para el cumplimiento de los objetivos planteados en su proyecto de tesis. Asimismo, realizó entrevistas y sesiones de retroalimentación con el personal desarrollador, las cuales permitieron fortalecer los criterios técnicos de la propuesta y garantizar su alineación con los estándares internos de la Dirección de TIC.

En constancia de lo expuesto, se emite la presente certificación para los fines académicos pertinentes.

Atentamente,

POLITÉCNICA DEL CARCHI DIRECCIÓN DE TIC

Msc. Juan Pablo López Goyez
DIRECTOR DE TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN
"Educamos para transformar el mundo"

Calle Antisana y Av. Universitaria
Telf: (06) 2995800
info@upec.edu.ec
www.upec.edu.ec
Tulcán - Ecuador

Anexo 7. Guía para sistema de gestión de configuración de software usando DevOps