



Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América

Facultad de Ingeniería de Sistemas e Informática

Escuela Profesional de Ingeniería de Software

**Integración y despliegue continuo para la gestión de
cambios de componentes de las aplicaciones web en
una entidad financiera**

TRABAJO DE SUFICIENCIA PROFESIONAL

Para optar el Título Profesional de Ingeniero de Software

AUTOR

Aarón Esteban CASTILLO RODRÍGUEZ

ASESOR

Dr. Cayo Víctor LEÓN FERNÁNDEZ

Lima, Perú

2025



Reconocimiento - No Comercial - Compartir Igual - Sin restricciones adicionales

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Usted puede distribuir, remezclar, retocar, y crear a partir del documento original de modo no comercial, siempre y cuando se dé crédito al autor del documento y se licencien las nuevas creaciones bajo las mismas condiciones. No se permite aplicar términos legales o medidas tecnológicas que restrinjan legalmente a otros a hacer cualquier cosa que permita esta licencia.

Referencia bibliográfica

Castillo, A. (2025). *Integración y despliegue continuo para la gestión de cambios de componentes de las aplicaciones web en una entidad financiera*. [Trabajo de suficiencia profesional de pregrado, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Escuela Profesional de Ingeniería de Software]. Repositorio institucional Cybertesis UNMSM.

Metadatos complementarios

Datos de autor	
Nombres y apellidos	Aarón Esteban Castillo Rodríguez
Tipo de documento de identidad	DNI
Número de documento de identidad	71523029
URL de ORCID	https://orcid.org/0009-0008-7229-7233
Datos de asesor	
Nombres y apellidos	Cayo Victor León Fernández
Tipo de documento de identidad	DNI
Número de documento de identidad	07001405
URL de ORCID	https://orcid.org/0000-0003-1704-8214
Datos del jurado	
Presidente del jurado	
Nombres y apellidos	Raúl Marcelo Armas Calderón
Tipo de documento	DNI
Número de documento de identidad	07156168
Miembro del jurado 1	
Nombres y apellidos	Nilo Eloy Carrasco Ore
Tipo de documento	DNI
Número de documento de identidad	09342780
Datos de investigación	
Línea de investigación	No aplica
Grupo de investigación	No aplica
Agencia de financiamiento	Sin financiamiento
Ubicación geográfica de la investigación	País: Perú Departamento: Lima Provincia: Lima

	Distrito: Cercado de Lima Jr. Carlos Amezaga No. 375 Universidad Nacional Mayor de San Marcos Latitud: -12.0564232 Longitud: -77.0843327
Año o rango de años en que se realizó la investigación	2024
URL de disciplinas OCDE	2.02.04 -- Ingeniería de sistemas y comunicaciones https://purl.org/pe-repo/ocde/ford#2.02.04



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
Escuela Profesional de Ingeniería de Software

Acta Virtual de Sustentación
del Trabajo de Suficiencia Profesional

Siendo las 18:00 horas del día 13 de enero del año 2025, se reunieron virtualmente los docentes designados como Miembros de Jurado del Trabajo de Suficiencia Profesional, presidido por el Ing. Armas Calderón Raúl Marcelo (Presidente), Mg. Carrasco Ore Nilo Eloy (Miembro) y el Dr. León Fernández Cayo Víctor (Miembro Asesor), usando la plataforma Meet (<https://meet.google.com/seg-hqdz-yxf>), para la sustentación virtual del Trabajo de Suficiencia Profesional intitulado: **“INTEGRACIÓN Y DESPLIEGUE CONTINUO PARA LA GESTIÓN DE CAMBIOS DE COMPONENTES DE LAS APLICACIONES WEB EN UNA ENTIDAD FINANCIERA”**, por el Bachiller **Castillo Rodríguez Aarón Esteban**; para obtener el Título Profesional de Ingeniero de Software.

Acto seguido de la exposición del Trabajo de Suficiencia Profesional, el Presidente invitó al Bachiller a dar las respuestas a las preguntas establecidas por los miembros del Jurado.

El Bachiller en el curso de sus intervenciones demostró pleno dominio del tema, al responder con acierto y fluidez a las observaciones y preguntas formuladas por los señores miembros del Jurado.

Finalmente habiéndose efectuado la calificación correspondiente por los miembros del Jurado, el Bachiller obtuvo la nota de **19 (DIECINUEVE)**.

A continuación el Presidente de Jurados el Ing. Armas Calderón Raúl Marcelo, declara al Bachiller **Ingeniero de Software**.

Siendo las 18:53 horas, se levantó la sesión.

Presidente

Ing. Armas Calderón Raúl Marcelo

Miembro

Mg. Carrasco Ore Nilo Eloy

Miembro Asesor

Dr. León Fernández Cayo Víctor



CERTIFICADO DE SIMILITUD

Yo, LEÓN FERNÁNDEZ CAYO VICTOR en mi condición de asesor acreditado con la Resolución Decanal N° 001059-2024-D-FISI/UNMSM y Oficio N° 0017-PTTSP-FISI-2024 del Trabajo de Suficiencia Profesional cuyo título es Integración y Despliegue continuo para la gestión de cambios de componentes de las aplicaciones web en una entidad financiera, presentado por el bachiller CASTILLO RODRÍGUEZ AARÓN ESTEBAN para optar el título de Ingeniero de Software CERTIFICO que se ha cumplido con lo establecido en la Directiva de Originalidad y de Similitud de Trabajos Académicos, de Investigación y Producción Intelectual. Según la revisión, análisis y evaluación mediante la plataforma PIDS, el documento evaluado cuenta con el 8% de similitud, nivel **PERMITIDO** para continuar con los trámites correspondientes y para su **publicación en el repositorio institucional.**

Se emite el presente certificado en cumplimiento de lo establecido en las normas vigentes, como uno de los requisitos para la obtención del título.

Lima, 8 de enero de 2025

Firma del Asesor

Nombres y Apellidos del asesor: LEÓN FERNÁNDEZ CAYO VICTOR

DNI 07001405

UNIVERSIDAD MAYOR DE SAN MARCOS
FACULTAD DE INGENIERIA DE SISTEMAS E
INFORMATICA
ESCUELA PROFESIONAL DE INGENIERIA DE
SOFTWARE

**Integración y Despliegue continuo para la gestión
de cambios de componentes de las aplicaciones
web en una entidad financiera**

Autor: Castillo Rodríguez, Aarón Esteban

Asesor: Dr. León Fernández, Cayo Victor

Título: Trabajo de Suficiencia profesional para optar el Título Profesional de
Ingeniero de Software

Fecha: Noviembre de 2024

RESUMEN

El presente trabajo aborda la optimización de la gestión de cambios en componentes de aplicaciones web que se realizó dentro de una entidad financiera, un sector donde la rapidez y seguridad en el desarrollo digital son esenciales para mantener la competitividad. La propuesta consistió en implementar una plataforma de integración y despliegue continuo (CI/CD) que reduzca los tiempos de despliegue y eleve la calidad del software, alineándose con las normativas regulatorias y mejores prácticas como DevOps. La metodología empleada fue SCRUM e incluyó tres etapas principales: análisis y diagnóstico de los procesos actuales, implementación de herramientas clave como Jenkins, Bitbucket, SonarQube y Artifactory, y capacitación de un equipo piloto en el uso de estas tecnologías. La solución se aplicó en un equipo piloto de la entidad financiera, conformado por desarrolladores, operadores y especialistas en seguridad. Los resultados incluyen una reducción del 60% en los tiempos de despliegue, mejoras en la calidad del código y una mayor eficiencia operativa, además de optimizar costos relacionados con tiempos de operación y aprobaciones manuales. Este proyecto establece un modelo replicable que refuerza la agilidad, seguridad y sostenibilidad en el desarrollo de software, recomendando su expansión a otros equipos para maximizar su impacto en la organización.

Palabras clave: DevOps, Integración Continua, Despliegue continuo, Aplicaciones Web

MAJOR NATIONAL UNIVERSITY OF SAN MARCOS
FACULTY OF SYSTEMS AND INFORMATICS ENGINEERING

PROFESSIONAL SCHOOL OF SOFTWARE
ENGINEERING

**Continuous Integration and Deployment for change management of
web application components in a financial institution**

Autor: Castillo Rodríguez, Aarón Esteban

Advisor: Dr. León Fernández, Cayo Victor

Title: Professional Sufficiency Work to opt for the Professional Title of Systems
Engineer

Date: November 2024

ABSTRACT

This work addresses the optimization of change management in web application components that was performed within a financial institution, a sector where speed and security in digital development are essential to maintain competitiveness. The proposal consisted of implementing a continuous integration and deployment (CI/CD) platform to reduce deployment times and increase software quality, aligned with regulatory standards and best practices such as DevOps. The methodology used was SCRUM and included three main stages: analysis and diagnosis of current processes, implementation of key tools such as Jenkins, Bitbucket, SonarQube and Artifactory, and training of a pilot team in the use of these technologies. The solution was applied in a pilot team of the financial institution, made up of developers, operators and security specialists. The results include a 60% reduction in deployment times, improvements in code quality and greater operational efficiency, in addition to optimizing costs related to operation times and manual approvals. This project establishes a replicable model that reinforces agility, security and sustainability in software development, recommending its expansion to other teams to maximize its impact on the organization.

Keywords: DevOps, Continuous Integration, Continuous Deployment, Web Applications

TABLA DE CONTENIDOS

RESUMEN.....	IV
ABSTRACT	V
TABLA DE CONTENIDOS	VI
INTRODUCCION	1
CAPÍTULO I - TRAYECTORIA PROFESIONAL.....	3
CAPÍTULO II - CONTEXTO EN EL QUE SE DESARROLLÓ LA EXPERIENCIA.....	6
2.1 EMPRESA - ACTIVIDAD QUE REALIZA	6
2.2 VISIÓN	6
2.3 MISIÓN	6
2.4 ORGANIZACIÓN DE LA EMPRESA	7
2.5 ÁREA, CARGO Y FUNCIONES DESEMPEÑADAS	8
2.6 EXPERIENCIA PROFESIONAL REALIZADA EN LA ORGANIZACIÓN	8
CAPÍTULO III – ACTIVIDADES DESARROLLADAS	10
3.1 SITUACION PROBLEMÁTICA	10
3.1.1 Fase de pre-despliegue	10
3.1.2 Fase de despliegue:.....	13
3.1.3 Fase de post-despliegue:.....	16
3.1.4 Cierre del Pase a Producción	18
3.1.5 Definición del problema	18
3.2 SOLUCIÓN	19
3.2.1 Objetivos.....	19
3.2.2 Alcance.....	20
3.2.3 Etapas y metodología	21
3.2.3 Fundamentos teóricos utilizados.....	31
3.2.4 Implementación de las áreas, procesos, sistemas y buenas practicas	34
3.3 EVALUACIÓN	49
3.3.1 Evaluación Costo-Beneficio	49
3.3.2 Interpretación del VAN y del TIR.....	52
3.3.3 Ahorro Anual Proyectado.....	54
3.3.4 Flujo de Caja Proyectado.....	54
3.3.5 Cálculo de VAN (Valor Actual Neto)	55
3.3.6. Cálculo de TIR (Tasa Interna de Retorno)	55
3.3.7. Conclusión	55
CAPÍTULO IV. REFLEXIÓN CRÍTICA DE LA EXPERIENCIA	56
4.1. INTRODUCCIÓN.....	56
4.2. EXPERIENCIA ADQUIRIDA.....	56
4.3. LIMITACIONES Y DESAFÍOS	56
4.4. ANÁLISIS CRÍTICO DE LAS METODOLOGÍAS Y PROCESOS.....	57
4.5. ENSEÑANZAS Y RECOMENDACIONES PARA FUTURAS IMPLEMENTACIONES	57
4.6. CONCLUSIÓN	58
CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES	59
5.1. CONCLUSIONES.....	59
5.2. RECOMENDACIONES	59
5.3. REFERENCIAS BIBLIOGRÁFICAS.....	60
GLOSARIO	62

INDICE DE TABLAS

TABLA 1 EXPERIENCIAS LABORALES	3
TABLA 2 CERTIFICACIONES	4
TABLA 3 IDIOMAS	4
TABLA 4 TABLA DE ACTIVIDADES EN FASE DE PRE-DESPLIEGUE	12
TABLA 5 RESPONSABILIDADES EN LA FASE DE DESPLIEGUE	15
TABLA 6 ACTIVIDADES EN LA FASE DE POST-DESPLIEGUE.....	17
TABLA 7 TAREAS Y ACTIVIDADES DEL SPRINT 0	22
TABLA 8 TAREAS Y ACTIVIDADES DEL SPRINT 1	23
TABLA 9 ACTIVIDADES DEL SPRINT 3.....	27
TABLA 10 ACTIVIDADES DEL SPRINT 4.....	28
TABLA 11 ACTIVIDADES DEL SPRINT 5.....	29
TABLA 12 ACTIVIDADES DEL SPRINT 6.....	30
TABLA 13 COSTO POR CADA ROL EN EL PROYECTO.....	49
TABLA 14 COSTO POR CADA RECURSO DE INFRAESTRUCTURA USADO EN EL PROYECTO	50
TABLA 15 TIEMPOS DE DESPLIEGUE ANUAL POR PROYECTO.....	50
TABLA 16 AHORRO DE TIEMPO ANUAL ESTIMADO POR PROYECTO	51
TABLA 17 AHORRO DE TIEMPO ANUAL POR 5 PROYECTOS	51
TABLA 18 COSTO DE INGENIEROS EN FASE DE IMPLEMENTACIÓN.....	52
TABLA 19 COSTO DE SERVIDORES INFRAESTRUCTURA EN ETAPA DE IMPLEMENTACIÓN	52
TABLA 20 COSTOS OPERATIVOS DE PERSONAL PARA SOPORTE DE LA PLATAFORMA	53
TABLA 21 COSTOS OPERATIVOS DE INFRAESTRUCTURA PARA SOPORTE DE LA PLATAFORMA.....	53
TABLA 22 AHORRO ANUAL PROYECTADO POR CADA PERFIL	54
TABLA 23 FLUJO DE CAJA PROYECTADO	54

INDICE DE FIGURAS

FIGURE 1 ESQUEMA ORGANIZATIVO DE LA ENTIDAD FINANCIERA.....	7
FIGURE 2 FASE DE PRE-DESPLIEGUE.....	13
FIGURE 3 FASE DE DESPLIEGUE.....	16

INTRODUCCION

En cualquier país, el sector financiero se destaca por ser uno de los sectores más competitivos y desafiantes donde se usa tecnología de punta. En el ámbito actual, la elaboración de productos digitales que integren velocidad, excelencia y protección es un elemento fundamental que incide en el éxito de las instituciones financieras. Dentro de esta gama de productos, las aplicaciones web han adquirido un papel crucial al posibilitar a los usuarios llevar a cabo transacciones bancarias en cualquier momento y lugar.

Sin embargo, para asegurar la vigencia en el mercado, el cumplimiento de las regulaciones gubernamentales y la protección de los datos sensibles de los usuarios es fundamental realizar un mantenimiento y una actualización constantes de las aplicaciones web. El objetivo del presente informe profesional es implementar una plataforma de integración y despliegue continuo para la gestión de cambios de componentes de las aplicaciones web en una entidad financiera.

En la actualidad, uno de los principales desafíos que enfrenta el sector financiero es la optimización del tiempo de entrega de software, también referido como "lead time". La disponibilidad de esta métrica se ve influenciada por varios factores, tales como la realización de procesos manuales, la burocracia interna y las aprobaciones manuales, lo cual puede resultar en la falta de disponibilidad de los servicios y retrasos innecesario si no se siguen prácticas adecuadas.

En este contexto, las prácticas de integración y despliegue continuos, impulsadas por la metodología DevOps, representan una solución eficaz para optimizar el tiempo de entrega del software. En la actualidad, la implementación de una plataforma tecnológica basada en principios DevOps se considera un elemento fundamental para el éxito y la competitividad de las entidades financieras.

A continuación, se expone la estructura del trabajo de suficiencia profesional.

- En el capítulo I se analiza la trayectoria profesional, evidenciando la experiencia laboral adquirida durante el ejercicio de la profesión a lo largo del tiempo.

- Seguidamente, en el capítulo II se detalla el contexto en el cual tuvo lugar la experiencia, incluyendo información relevante sobre la empresa y la práctica profesional llevada a cabo en dicha organización.
- El objetivo del capítulo III se describen las actividades llevadas a cabo en este trabajo, tales como la exposición de la problemática, la definición de los objetivos, la aplicación de la metodología empleada y la presentación de los resultados obtenidos con la solución propuesta.
- En el capítulo VI se presenta un análisis crítico de la experiencia y la contribución del autor, así como las necesidades abordadas, la experiencia previa y la formación necesaria.
- Finalmente, en el capítulo V encontraran las conclusiones y recomendaciones.

CAPÍTULO I - TRAYECTORIA PROFESIONAL

El bachiller Aarón Castillo Rodríguez tiene más de 10 años de experiencia en tecnología, destacando en DevOps, ingeniería de confiabilidad de sitios (SRE) y arquitecturas en la nube. Ha liderado equipos en proyectos estratégicos, mejorando la eficiencia y reduciendo costos en los sectores financiero y retail (**ver Tabla 1**)

Ha implementado plataformas tecnológicas para empresas como Belcorp, Intercorp Retail y Everis Perú, estableciendo procesos de Integración y Despliegue Continuo (CI/CD) que redujeron tiempos de entrega.

Además, ha diseñado arquitecturas escalables en AWS, GCP y Azure, y dirigido migraciones de plataformas de comercio electrónico. Actualmente, trabaja en una reconocida cadena de cafeterías internacional, desarrollando herramientas que optimizan los procesos de desarrollo de software.

Tabla 1 Experiencia Laboral

Empresa	Puesto	Fecha	Logros
EPAM	Senior Platform Engineer	Julio 2023 - Actualidad	- Implementación de medición automática de métricas DORA y optimización de pipelines CI/CD.
Belcorp	Senior Site Reliability Engineer	Dic 2021 - junio 2023	- Liderazgo del equipo SRE y migración exitosa a VTEX, logrando reducir costos de infraestructura de \$7200 a \$700.
Intercorp Retail	Senior DevOps Specialist	Mar 2021 - Dic 2021	- Optimización de costos en \$3000 mensuales y automatización de despliegues en GCP para proyectos clave de retail.
Everis Perú	Lead DevOps Architect	2018 - 2021	Reducción del tiempo de entrega de días a horas mediante CI/CD para más de 100 equipos de productos.

Anda Technologies	Backend Developer	Jul 2016 - Mar 2018	Implementación de servicios backend y plataforma CI/CD, mejorando la eficiencia del equipo de desarrollo.
Telemovil GPS	Software Developer	Mar 2014 - Jul 2016	Diseño e implementación de servicios backend y migración exitosa a infraestructura basada en AWS.

Nota: Elaborado por el autor

Como punto adicional, el bachiller ha obtenido diferentes certificaciones relacionadas a su campo laboral según se muestra en la Tabla 2.

Tabla 2 Certificaciones

Periodo	Certificación	Institución
2021	Azure Fundamentals	Microsoft
2021	Azure Solutions Architect Expert	Microsoft
2020	Certified Digital Transformation Professional	Global Trust Association
2019	DevOps Master	EXIN
2019	Lean IT Foundation	EXIN

Nota: Elaborado por el autor

Finalmente, el bachiller tiene dominado el lenguaje Inglés tal cual se muestra en la Tabla 3.

Tabla 3 Idioma dominados

Periodo	Idioma	Nivel
----------------	---------------	--------------

2024	Ingles	B2
-------------	--------	----

Nota: Elaborado por el autor

CAPÍTULO II - CONTEXTO EN EL QUE SE DESARROLLÓ LA EXPERIENCIA

2.1 Empresa – Actividad que realiza

El proyecto denominado Integración y Despliegue Continuo, se desarrolló en una institución financiera peruana con presencia a nivel nacional e internacional. Ofrece una diversidad de servicios financieros que abarcan la financiación para el comercio internacional, el arrendamiento, los seguros y la consultoría financiera. La principal unidad estratégica de negocios se enfoca en proporcionar soluciones financieras completas tanto para empresas como para personas.

2.2 Visión

La empresa tiene como objetivo convertirse en la organización que ofrece la experiencia más satisfactoria a sus clientes, a través de un enfoque sencillo, cercano y oportuno. Además, la empresa busca convertirse en el empleador de preferencia en Perú, motivando y promoviendo el desarrollo de los profesionales más destacados. En el ámbito de la gestión, la visión de la empresa es establecerse como un punto de referencia destacado a nivel regional en el campo de la gestión empresarial, reforzando su posición de liderazgo en el sector financiero de Perú.

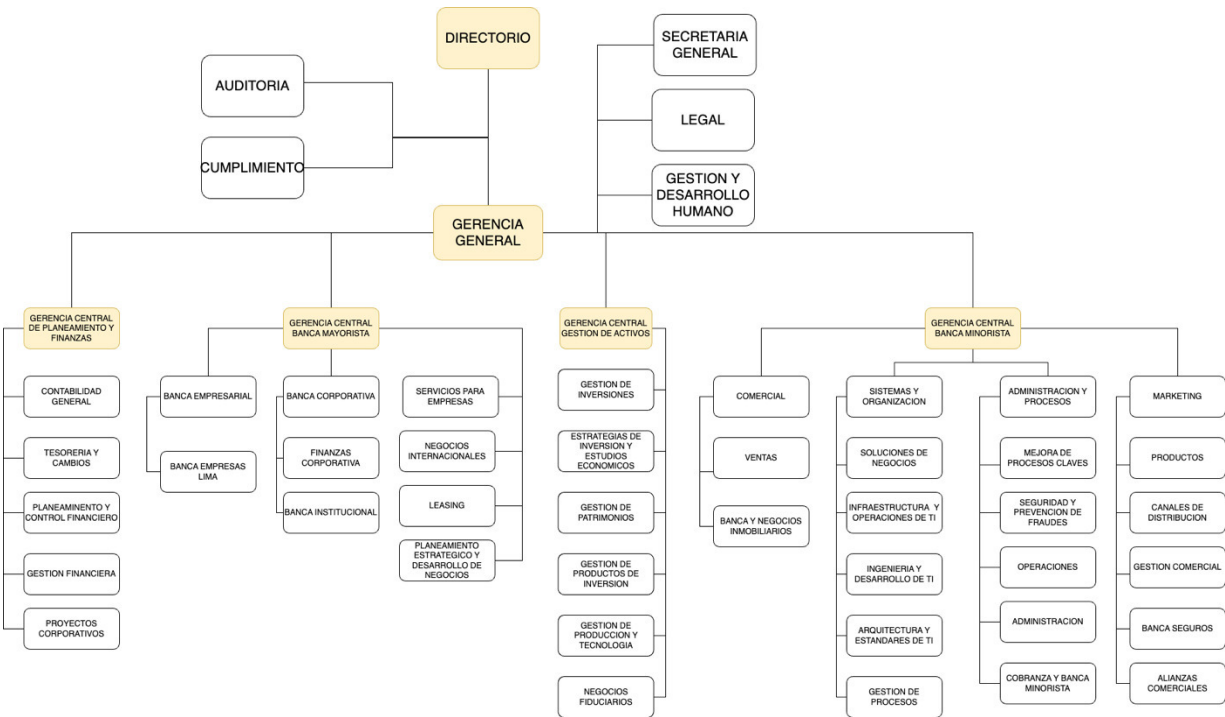
2.3 Misión

La empresa tiene como objetivo ser aliada estratégica de sus clientes, colaboradores y del país. La empresa se enfoca en asegurar una experiencia de desarrollo y cuidado para sus empleados. Ofrece productos y servicios financieros adaptados a las necesidades de los clientes. Además, contribuye al crecimiento económico y al bienestar de los ciudadanos del país.

2.4 ORGANIZACIÓN DE LA EMPRESA

La estructura orgánica de la empresa es de tipo jerárquico que comprende un directorio, gerencia general y los demás órganos de apoyo y línea como se muestra en la Figura 1. El desarrollo de negocios, las finanzas, la tecnología y los riesgos son áreas fundamentales, siendo crucial el desarrollo de negocios para promover soluciones financieras innovadoras. Dentro de la estructura organizativa, este proyecto se desarrolló en el área de DevSecOps, que pertenece al departamento de Ingeniería y Desarrollo de TI, cuya actividad principal es la automatización y optimización de los procesos de desarrollo y operación de software.

Figura 1 Esquema Organizativo de la entidad financiera



Nota: Preparado por el autor

2.5 ÁREA, CARGO Y FUNCIONES DESEMPEÑADAS

El autor del presente trabajo demostró tener competencias y habilidades en el campo especializado de DevSecOps, desempeñándose primero como ingeniero de DevOps y luego ascendió al cargo de ingeniero líder de DevOps cuyos roles y responsabilidades fueron lo siguiente:

- Implementar, mantener y administrar más de 10 herramientas dentro de la plataforma integral de desarrollo, seguridad y operaciones de la empresa.
- Diseñar, desarrollar e implementar pipelines de integración y despliegue continuo para más de 8 tecnologías utilizadas en el entorno bancario, garantizando su correcto funcionamiento y realizando labores de mantenimiento periódico.
- Brindar capacitación y asesoramiento a más de 20 equipos de desarrollo de producto en las mejores prácticas de DevOps, tales como el control de versiones, implementaciones continuas, automatización de pruebas y fomento de la colaboración entre equipos.
- Gestionar de manera eficiente la infraestructura en la nube basada en Azure, con más de 15 servidores dedicados a respaldar la plataforma integral de desarrollo, seguridad y operaciones (DevSecOps).
- Encabezar de manera técnica al equipo, asignando responsabilidades y garantizando el cumplimiento oportuno de los plazos establecidos.

2.6 EXPERIENCIA PROFESIONAL REALIZADA EN LA ORGANIZACIÓN

Durante su prolongada estancia en esta respetada organización, el autor se vio confrontado con una amplia gama de desafíos y obstáculos relacionados con la compleja tarea de implementar exitosamente soluciones DevSecOps en un exigente entorno bancario caracterizado por su estricta regulación y normativas.

El autor empezó como analista DevOps durante 2 años y luego paso los próximos 2 años como líder del equipo de DevSecOps. Este trabajo se desarrolló cuando el autor tenía el rol de analista DevOps en el segundo año a través de la consultora donde laboraba.

Como líder destacado de un equipo altamente capacitado, tuvo la importante responsabilidad de coordinar de manera efectiva la implementación estratégica de diversas herramientas innovadoras y procesos especializados, los cuales lograron optimizar de forma notable y sustancial el ciclo completo de desarrollo y despliegue de software. Este valioso aporte contribuyó de manera directa y significativa a potenciar la eficiencia operativa y el rendimiento general de la entidad en cuestión. Además, desempeñó un papel fundamental al trabajar en estrecha colaboración y coordinación con los diversos equipos de producto, brindando asesoramiento técnico especializado y liderando de manera efectiva proyectos estratégicos y fundamentales para la actualización y mejora continua de la infraestructura tecnológica de la empresa.

CAPÍTULO III – ACTIVIDADES DESARROLLADAS

3.1 SITUACION PROBLEMATICA

Antes de la puesta en marcha de la plataforma de integración y despliegue continuo, el proceso de implementación de modificaciones de software en la institución financiera enfrentaba diversos desafíos que incidían en la eficacia y la rapidez del procedimiento.

El despliegue se estructuraba en tres etapas fundamentales: una fase preparatoria (pre-despliegue), una fase de ejecución (despliegue) y una fase de seguimiento (post-despliegue). No obstante, la finalización integral del ciclo demandaba de tres a cinco días, dependiendo de la cantidad de modificaciones a ser implementadas.

La ejecución manual del proceso, la necesidad de obtener aprobaciones en cada fase y la participación de diversas áreas como el Equipo de Producto, Seguridad de Tecnologías de la Información, Operaciones TI y Gobierno de TI de la Información, generaban dificultades para cumplir con los plazos establecidos.

3.1.1 Fase de pre-despliegue

En esta fase se ejecutan los pasos previos de preparación antes del despliegue, el flujo de estos pasos se grafican en la Figura 2. El resumen se muestra en la Tabla 4. A continuación se detallan cada uno de los puntos:

3.1.1.1 Preparación del Paquete de Liberación

- **Integración de cambios:** Se consolidan los cambios del código realizado por los diferentes equipos de desarrollo.
- **Recopilación de información sobre librerías:** Se detalla el listado de las librerías de software utilizadas para la compilación.
- **Revisión de pares:** Se lleva a cabo una revisión por parte de un compañero para asegurar la calidad y funcionalidad del código.
- **Validación de la calidad del código:** Se ejecutan herramientas de análisis estático para asegurar el cumplimiento de estándares de calidad.

- **Validación de la compilación:** Se verifica que el código se pueda compilar correctamente sin errores.

3.1.1.2 Creación del Manual de Instalación y Configuración del Sistema

- **Formato de despliegue:** Llenado de la información clave (nombre del proyecto, compilación, descarga de librerías).
- **Cambio de configuración del código:** Descripción de los pasos necesarios para ajustar el código al entorno productivo.
- **Modificación de secretos:** Instrucciones para actualizar contraseñas, claves API, y otros datos sensibles.
- **Configuración del entorno:** Pasos para preparar el entorno de ejecución del proyecto, como la configuración de servidores o bases de datos.

3.1.1.3 Aprobación del Líder Técnico

- **Justificación técnica:** Explicación de la razón detrás del cambio técnico.
- **Despliegue a alto nivel:** Presentación de los pasos generales del despliegue planificado.

3.1.1.4. Congelamiento del Código Fuente

- **Subida a la herramienta de gestión:** El paquete de liberación se sube a la herramienta de control de versiones y se completa la información requerida.

3.1.1.5. Creación del Evento en el Calendario de Producción

- **Reserva en el calendario:** Se asegura un espacio en el calendario de producción, evitando superposiciones con otros despliegues.
- **Asignación de personal clave:** Se agenda la participación de personas clave, como el equipo de Agile Ops, para el pase a producción.

3.1.1.6. Aprobación de Seguridad TI

- **Revisión de artefactos:** El equipo de Seguridad TI revisa los artefactos generados para asegurar que cumplan con los requisitos de seguridad establecidos.

3.1.1.7 Revisión del Manual de Instalación y Configuración

- **Validación por parte de Operaciones TI:** El equipo de Operaciones TI revisa la documentación para familiarizarse con los pasos de instalación y preparación antes del despliegue.

3.1.1.8 Aprobación del Gobierno de TI

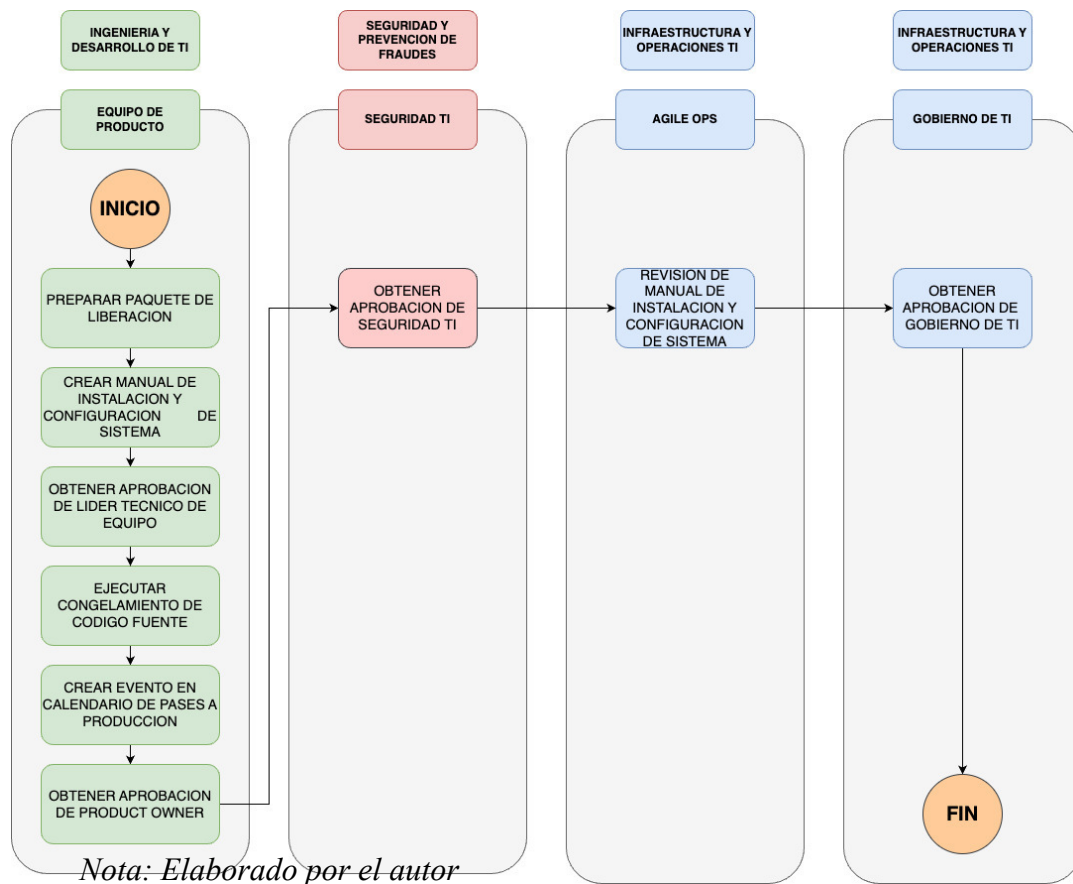
- **Revisión de despliegues programados:** El equipo de Gobierno de TI revisa el calendario de despliegues y emite su aprobación o sugerencias si es necesario.

Tabla 4 Tabla de actividades en fase de pre-despliegue

Actividad	Responsable	Descripción
Preparación del paquete de liberación	Equipo de desarrollo	Consolidación de cambios, revisión de pares, validación de calidad y compilación.
Creación del manual de instalación	Equipo de desarrollo	Documentación del proceso de configuración y despliegue.
Aprobación del líder técnico	Líder técnico	Evaluación técnica de los cambios y despliegue a alto nivel.
Congelamiento del código fuente	Equipo de desarrollo	Subida del paquete de liberación a la herramienta de gestión de cambios.
Creación del evento en el calendario	Operaciones TI	Reservar espacio en el calendario y asignar a las personas clave para el despliegue.
Aprobación de Seguridad TI	Seguridad TI	Revisión de artefactos y validación de conformidad con las políticas de seguridad.
Revisión del manual de instalación	Operaciones TI	Validación del manual de instalación para el despliegue.
Aprobación del Gobierno de TI	Gobierno de TI	Revisión y aprobación final del calendario de despliegues.

Nota: Elaborado por el autor

Figura 2 Fase de pre-despliegue



3.1.2 Fase de despliegue:

La fase de despliegue se ejecuta en el día previamente acordado en el calendario de pases a producción, con la participación de dos equipos clave: Operaciones TI y el Centro de Operaciones de Seguridad. Estos equipos son los únicos autorizados para realizar modificaciones en el entorno productivo, según las políticas estrictas de la entidad financiera como se muestra en la Figura 3. El resumen de estas actividades se muestra en la Tabla 5.

Es importante destacar que:

1. Solo Operaciones TI y el Centro de Operaciones de Seguridad pueden realizar modificaciones en producción.

2. Otros equipos (desarrolladores, propietarios de producto, etc.) no están autorizados a intervenir en el proceso de producción.
3. Los cambios no planificados son sancionados según las normativas internas.

A continuación, se detallan los pasos que deben seguirse de acuerdo con el manual de instalación y configuración:

3.1.2.1 Obtener el manual de instalación y configuración del sistema

- Se descarga el manual de control de versiones para seguir los procedimientos establecidos.

3.1.2.2 Obtener credenciales controladas por el Centro de Operaciones de Seguridad

- Se aseguran las credenciales como contraseñas de servidores, claves API, archivos de configuración sensibles, y otros secretos de aplicación.

3.1.2.3 Obtener los artefactos necesarios

- Se reúnen los artefactos de despliegue, incluyendo código fuente, librerías y otros componentes requeridos.

3.1.2.4 Instalar la nueva versión de la aplicación web

- La nueva versión se despliega en los servidores de producción asignados, siguiendo los parámetros establecidos.

3.1.2.5 Configurar la aplicación según el manual de configuración

- Se realiza la configuración de la aplicación en producción de acuerdo con los detalles técnicos especificados en el manual.

3.1.2.6 Notificar sobre cualquier incidente

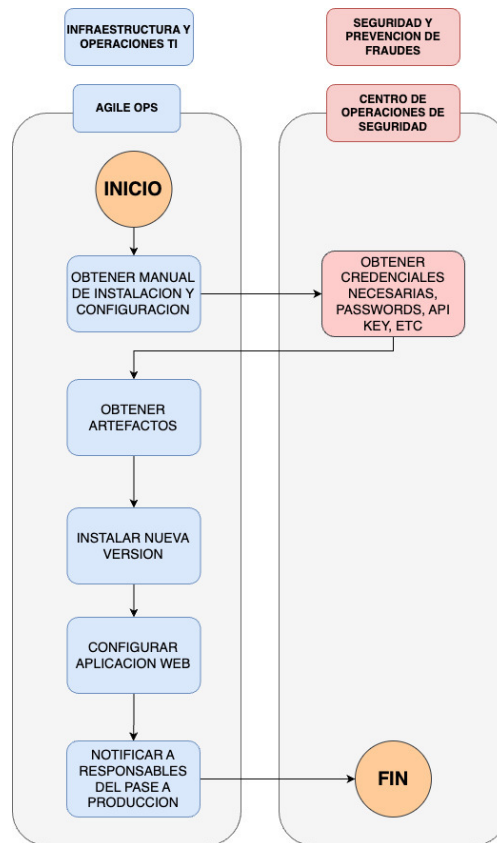
- Se notifican inmediatamente los problemas surgidos durante el despliegue a los responsables de Operaciones TI y el Centro de Operaciones de Seguridad.

Tabla 5 Responsabilidades en la fase de despliegue

Actividad	Responsable	Descripción
Obtener el manual de instalación	Operaciones TI	Descargar el manual de instalación desde el sistema de control de versiones.
Obtener credenciales de seguridad	Centro de Operaciones de Seguridad	Acceder a las contraseñas, claves API y secretos controlados por Seguridad.
Obtener artefactos de despliegue	Operaciones TI	Reunir el código fuente y las librerías necesarias para el despliegue.
Instalar la nueva versión de la aplicación	Operaciones TI	Desplegar la nueva versión en los servidores de producción.
Configurar la aplicación en producción	Operaciones TI	Configurar la aplicación según el manual de configuración.
Notificar problemas	Operaciones TI	Comunicar cualquier incidencia durante el pase a producción.

Nota: Elaborado por el autor

Figure 3 Fase de despliegue



Nota: Elaborado por el autor

3.1.3 Fase de post-despliegue:

Una vez completados todos los pasos detallados en el **manual de instalación y configuración**, se inicia la fase de post-despliegue. Esta fase es crucial para garantizar la estabilidad del sistema y confirmar que los cambios implementados han sido efectivos y no han afectado negativamente el rendimiento. El resumen de estas actividades se muestra en la Tabla 6.

3.1.3.1 Pruebas de regresión

- El equipo de producto es responsable de validar que el sistema sigue funcionando correctamente desde la perspectiva del usuario final. Las pruebas de regresión incluyen:

- Verificar que todas las funcionalidades anteriores sigan operando sin problemas.
- Se tiene que asegurar que las nuevas funcionalidades se hayan integrado de manera correcta en el sistema.

3.1.3.2 Monitorización de métricas de infraestructura:

- El equipo de Operaciones TI monitorea las métricas clave del sistema para asegurar que el despliegue no haya impactado negativamente el entorno. Las métricas que se controlan son:
 1. **Cantidad de peticiones correctamente atendidas:** Se verifica que el sistema continúe respondiendo adecuadamente a las solicitudes de los usuarios.
 2. **Cantidad de errores:** Se supervisan los errores generados por el sistema para identificar posibles fallos en el código o la configuración.
 3. **Latencia:** Se mide el tiempo de respuesta del sistema para garantizar que no haya aumentos significativos que afecten la experiencia del usuario.

Tabla 6 Actividades en la fase de post-despliegue

Actividad	Responsable	Descripción
Pruebas de regresión	Equipo de Producto	Validar el correcto funcionamiento del sistema desde el punto de vista del usuario.
Monitorización de peticiones	Operaciones TI	Verificar que el sistema responde adecuadamente a las solicitudes de los usuarios.
Monitorización de errores	Operaciones TI	Supervisar la cantidad de errores generados para detectar fallos.
Monitorización de latencia	Operaciones TI	Asegurar que los tiempos de respuesta no se hayan incrementado significativamente.

Nota: Elaborado por el autor

3.1.4 Cierre del Pase a Producción

Una vez que ambos equipos han completado sus tareas y validado que el sistema sigue funcionando correctamente, se procede a **cerrar el pase a producción**. Este proceso incluye:

1. **Confirmación del equipo de producto:** El equipo confirma que las pruebas de regresión han sido exitosas y que todas las funcionalidades operan correctamente.
2. **Verificación del equipo de Operaciones TI:** Operaciones TI asegura que no haya degradación en el rendimiento del sistema.
3. **Notificación oficial:** Se emite una notificación formal indicando que el despliegue ha sido completado con éxito y que el sistema está operando correctamente en el entorno productivo.

3.1.5 Definición del problema

El problema principal identificado en este trabajo fue la falta de automatización en las tres fases críticas del proceso de despliegue de software en la entidad financiera. Estas fases, que incluían la preparación del paquete de liberación, la creación del manual de instalación, la obtención de múltiples aprobaciones y la organización de recursos en el calendario, dependían de procesos manuales que prolongaban considerablemente los tiempos de entrega y aumentaban la posibilidad de errores humanos.

Desde la perspectiva del autor, se observó que la ausencia de prácticas automatizadas no solo incrementaba la complejidad operativa, sino que también afectaba la eficiencia y la calidad del proceso de despliegue. A esto se sumaba la falta de colaboración efectiva entre equipos, lo cual generaba una dependencia innecesaria de actividades manuales y ralentizaba la implementación de los cambios en producción.

Por lo tanto, la necesidad de reducir riesgos, optimizar la entrega de software y mejorar la satisfacción del cliente evidenció la urgencia de implementar una plataforma de integración y despliegue continuo. Esta solución permitiría acelerar los ciclos de despliegue, minimizar errores humanos y garantizar una implementación más eficiente y segura de los componentes de las aplicaciones web.

3.1.5.1 Problema principal

El problema principal identificado en este trabajo fue la falta de automatización en las tres fases críticas del proceso de despliegue de software dentro de la entidad financiera. Estas fases incluían la preparación del paquete de liberación, la creación del manual de instalación, la obtención de múltiples aprobaciones y la organización de recursos. La alta dependencia de procesos manuales generaba tiempos prolongados en la entrega de software, aumentaba la probabilidad de errores humanos y dificultaba la eficiencia operativa, lo que impactaba negativamente la calidad del servicio y la satisfacción del cliente.

3.1.5.2 Problemas específicos

1. **Tiempos prolongados de entrega:** La ejecución manual de tareas en las fases de pre-despliegue, despliegue y post-despliegue ralentizaba significativamente los ciclos de entrega del software.
2. **Alta probabilidad de errores humanos:** La repetición manual de tareas, como la documentación y validaciones, incrementaba el riesgo de errores que podían afectar la disponibilidad y funcionamiento del software en producción.
3. **Riesgo en la calidad y seguridad:** La falta de procesos automatizados limitaba las validaciones necesarias para asegurar que los cambios cumplieran con estándares de calidad y seguridad antes de pasar a producción.

3.2 Solución

- Integración y Despliegue continuo para la gestión de cambios de componentes de las aplicaciones web en una entidad financiera

3.2.1 Objetivos

3.2.1.1 Objetivo Principal:

- Implementar una plataforma de integración y despliegue continuo para automatizar y optimizar la gestión de cambios en los componentes de las aplicaciones web, reduciendo los tiempos de entrega, minimizando errores humanos y garantizando la calidad y seguridad en una entidad financiera.

3.2.1.2 Objetivos Específicos:

- Implementar la plataforma de Integración para la gestión de cambios de componentes de las aplicaciones web en una entidad financiera para reducir los tiempos de entrega y minimizar errores manuales.
- Implementar la plataforma de despliegue para la gestión de cambios de componentes de las aplicaciones web en una entidad financiera para agilizar el proceso de implementación, reducir los pasos manuales y la complejidad en las aprobaciones.
- Optimizar las validaciones de calidad y seguridad mediante la implementación de procesos automatizados que aseguren el cumplimiento de estándares antes de pasar a producción.

3.2.2 Alcance

El alcance de este trabajo fue la habilitación de plataforma de integración y despliegue continuo para la gestión de cambios de componentes de las aplicaciones web en la entidad donde se desarrolló el proyecto.

El proyecto se desarrolló dentro de las instalaciones de la entidad financiera. Esta plataforma involucraba un conjunto de herramientas que son usadas mundialmente por distintas organizaciones y han sido validadas y probadas en organizaciones de similar tamaño. Entre las herramientas podemos destacar: Bitbucket, Jenkins, Artifactory y Sonarqube. Asimismo, se automatizó distintos procesos que están involucrados durante las fases de pre-despliegue, despliegue y post-despliegue que fueron explicadas anteriormente que están relacionadas a los flujos de integración y despliegue continuo.

Finalmente, este proyecto se concluyó con un piloto donde se puso a prueba la plataforma y los nuevos flujos creados. El piloto se ejecutó con la colaboración de otro

equipo que tomaron el rol de usuarios de la plataforma y usaron esta misma para hacer el despliegue de los cambios de su aplicación web.

3.2.3 Etapas y metodología

Para la implementación de la plataforma DevOps, se utilizó la metodología ágil SCRUM, la cual se basa en ciclos de trabajo llamados sprints. El proyecto fue dividido en 7 sprints, cuyos entregables fueron los incrementos funcionales de la plataforma.

3.2.3.1 Roles involucrados

Los roles SCRUM que participaron en el desarrollo del proyecto fueron los siguientes:

1. **Product Owner:** Responsable de traducir los requerimientos en historias de usuario. Se encargaba de mantener una comunicación constante con los líderes de los equipos de producto y de asegurar que la visión de la plataforma y las políticas internas de la entidad financiera se reflejaran correctamente.
2. **Scrum Master:** Facilitador encargado de mantener la agilidad dentro del equipo. Ayudaba a resolver problemas no técnicos relacionados con la gestión y mantenía la comunicación con otros Scrum Masters de la entidad financiera.
3. **Ingenieros DevOps:** Miembros técnicos encargados de la implementación y configuración de la plataforma DevOps.
4. **Líder Técnico:** Responsable de validar técnicamente la ejecución del proyecto, asegurando la correcta implementación de las soluciones propuestas.

3.2.3.2 Entregables finales

Los entregables principales del proyecto incluyeron:

- **Plataforma DevOps completamente funcional** con las herramientas Bitbucket, Jenkins, Artifactory y Sonarqube, instaladas y configuradas según las políticas de la entidad financiera.
- **Receta de integración continua** para aplicaciones web basadas en el framework Angular versión 10.
- **Receta de despliegue continuo** para aplicaciones web desplegadas en la plataforma Azure WebApps.

3.2.3.3 Hoja de ruta

El proyecto se dividió en 7 sprints, con el siguiente detalle:

3.2.3.3.1 Sprint 0

El Sprint 0 es clave para sentar las bases del proyecto. Su principal objetivo es diseñar la arquitectura de la plataforma DevOps, documentar los procesos actuales de integración y despliegue, y preparar al equipo de producto piloto para el uso de la nueva plataforma. Este sprint se centra en el análisis, la planificación y la creación de la infraestructura conceptual que guiará los sprints siguientes como se muestra en la Tabla 7.

Tabla 7 Tareas y actividades del sprint 0

Tarea	Descripción	Objetivo	Entregables
Diseño de la Arquitectura de la Nueva Plataforma DevOps	Se debe diseñar la arquitectura general que utilizará la plataforma DevOps, tomando en cuenta la infraestructura actual y las herramientas (Jenkins, Bitbucket, SonarQube, Artifactory, etc.).	Crear un diseño robusto que permita la escalabilidad de la plataforma y se adapte a las necesidades del equipo piloto y futuras implementaciones.	Diagrama arquitectónico, definición de conexiones y documentación de seguridad y permisos.
Creación de la Documentación Relevante sobre el Equipo de Producto Piloto	Identificar el equipo de producto piloto que usará la plataforma DevOps. Crear documentación sobre sus necesidades, metodologías y objetivos.	Tener una visión clara de las necesidades del equipo piloto y cómo la nueva plataforma mejorará sus tiempos de despliegue e integración.	Informe con descripción del equipo piloto, roles, tecnologías utilizadas y expectativas.
Creación de un Diagrama de Flujo del Desarrollo Actual	Representar gráficamente el proceso de desarrollo actual, desde la creación del código hasta el despliegue en producción.	Identificar puntos de fricción o pasos manuales que pueden ser automatizados con la nueva plataforma.	Diagrama de flujo del proceso de desarrollo, identificación de áreas para optimización.
Diseño Funcional del Estado Actual del Flujo de Integración	Crear un diseño funcional que describa cómo es el proceso de integración del código actualmente, documentando los pasos	Documentar y entender el flujo actual de integración para compararlo con el nuevo flujo automatizado.	Documento con detalles del proceso de integración y métricas de

	y herramientas utilizadas.		tiempos actuales.
Diseño Funcional del Estado Actual del Flujo de Despliegue	Documentar el proceso actual de despliegue, desde la aprobación de cambios hasta el despliegue en producción, revisando seguridad y pasos manuales.	Entender el proceso de despliegue actual y qué pasos deben ser automatizados o mejorados con la nueva plataforma DevOps.	Diagrama funcional del flujo de despliegue actual y listado de dificultades o problemas.

Nota: Elaborado por el autor

Resultados Esperados del Sprint 0

- Diseño detallado de la arquitectura de la plataforma DevOps que guiará el resto del proyecto.
- Documentación completa y actualizada sobre el equipo piloto, sus flujos de trabajo y las áreas de mejora.
- Diagramas que reflejan el estado actual de los flujos de integración y despliegue, sirviendo como referencia para futuras comparaciones.
- Identificación de las áreas críticas que deben ser automatizadas en la plataforma para mejorar la eficiencia y reducir errores humanos.

Este Sprint es fundamental para establecer una base sólida antes de pasar a la implementación de la plataforma en los siguientes sprints.

3.2.3.3.2 Sprint 1

El Sprint 1 se centro en la creación de las recetas de **Ansible** para la instalación y configuración de **Jenkins** (tanto componente maestro como esclavo) y **Bitbucket** en un ambiente no productivo. El objetivo es preparar la infraestructura base que servirá para automatizar el ciclo de vida de desarrollo y despliegue tal como se muestra en la Tabla 8.

Tabla 8 Tareas y actividades del sprint 1

Actividad	Descripción	Objetivo	Entregables
Creación de la Receta Ansible para la Instalación y Configuración de Jenkins (Componente Maestro)	Se creará una receta Ansible para automatizar la instalación y configuración del componente	Asegurar que Jenkins maestro esté completamente funcional en un entorno	Receta Ansible que automatiza la instalación y configuración de Jenkins. Jenkins maestro operativo

	maestro de Jenkins. Este será el nodo principal que gestionará los pipelines y coordinará la ejecución de trabajos entre los componentes esclavos.	controlado y listo para coordinar las tareas de CI/CD.	en el ambiente no productivo.
Configuración de Jenkins en Ambiente No Productivo (Componente Esclavo)	Instalación y configuración de los nodos esclavos de Jenkins, que ejecutarán las tareas distribuidas por el maestro, permitiendo distribuir la carga de trabajo.	Configurar un entorno distribuido que permita a Jenkins maestro delegar tareas a los nodos esclavos y ejecutar pipelines en paralelo.	Jenkins esclavo configurado y conectado al nodo maestro. Validación de la conectividad y ejecución de pipelines.
Creación de la Receta Ansible para la Instalación y Configuración de Bitbucket	Desarrollar una receta Ansible que automatice la instalación y configuración de Bitbucket para gestionar los repositorios de código fuente y conectarlo con Jenkins.	Tener Bitbucket completamente operativo, gestionando repositorios de código y conectado a Jenkins para la activación automática de pipelines mediante webhooks.	Receta Ansible para la instalación y configuración de Bitbucket. Bitbucket configurado con repositorios y conectado a Jenkins mediante webhooks.

Nota: Elaborado por el autor

Resultados Esperados del Sprint 1

- Jenkins (maestro y esclavo) estará completamente instalado y configurado en un entorno no productivo.
- Bitbucket estará listo para gestionar los repositorios de código y conectado a Jenkins para activar pipelines de integración continua.

- Las recetas Ansible permitirán la automatización de la instalación y configuración de Jenkins y Bitbucket, garantizando consistencia y repetibilidad en futuras implementaciones.
- Se probarán las conexiones entre Jenkins y Bitbucket para asegurar que los pipelines se inicien automáticamente tras cambios en el código.

Este sprint es crucial para establecer las herramientas base que facilitarán los procesos de CI/CD en la entidad financiera, asegurando que el entorno esté preparado para los siguientes sprints.

3.2.3.3.3 Sprint 2

Este sprint se centrará en la creación de las recetas Ansible para la instalación y configuración de **Artifactory** y **SonarQube** en un ambiente no productivo, así como el diseño del nuevo flujo de **Integración Continua (CI)** y **Despliegue Continuo (CD)**. Estas herramientas son críticas para la gestión de artefactos y el aseguramiento de la calidad del código, y los nuevos flujos CI/CD definirán cómo funcionará el proceso automatizado dentro de la entidad bancaria de acuerdo con la Tabla 9.

Tabla 9 Actividades del sprint 2

Tarea	Descripción	Objetivo	Entregables
Creación de la Receta Ansible para la Instalación y Configuración de Artifactory	Automatizar la instalación y configuración de Artifactory para gestionar artefactos en un entorno no productivo.	Automatizar la instalación y configuración de Artifactory para gestionar artefactos en un entorno de preproducción.	Receta Ansible para la instalación y configuración de Artifactory. Artifactory operativo en ambiente no productivo.
Creación de la Receta Ansible para la Instalación y Configuración de SonarQube	Automatizar la instalación y configuración de SonarQube para analizar la calidad del código y detectar vulnerabilidades antes de desplegar aplicaciones.	Tener SonarQube listo para analizar la calidad del código y detectar vulnerabilidades antes de desplegar las aplicaciones.	Receta Ansible para la instalación y configuración de SonarQube. SonarQube listo e integrado con los pipelines de Jenkins.
Diseño del Nuevo Flujo de Integración Continua	Diseñar un flujo de integración continua que valide automáticamente los	Definir un flujo de integración continua que asegure que cada cambio de código	Diagrama detallado del nuevo flujo de integración continua.

	cambios de código y almacene artefactos en Artifactory tras la construcción exitosa.	pase por un proceso de validación y pruebas antes de fusionarse.	Definición de los puntos de integración entre Jenkins, SonarQube, Bitbucket y Artifactory.
Diseño del Nuevo Flujo de Despliegue Continuo	Diseñar un flujo de despliegue continuo que automatice el proceso de despliegue, garantizando que los artefactos validados lleguen a producción sin intervención manual.	Automatizar el despliegue continuo asegurando que los cambios lleguen a producción de manera controlada y segura.	Diagrama detallado del nuevo flujo de despliegue continuo. Definición de los puntos de control y aprobaciones necesarias antes del despliegue.

Nota: Elaborado por el autor

Resultados Esperados del Sprint 2

- Artifactory y SonarQube completamente instalados y configurados en el ambiente no productivo.
- Recetas Ansible que permiten la instalación automática de estas herramientas.
- Flujos de integración continua y despliegue continuo diseñados, documentados e integrados en los pipelines de Jenkins.
- Validación de los procesos de integración y despliegue automático, asegurando que las herramientas se interconectan correctamente y los artefactos pasan por las validaciones requeridas.

Este sprint establece la infraestructura crítica para la gestión de artefactos y el aseguramiento de la calidad, asegurando que el código pase por todas las etapas de validación antes de ser desplegado.

3.2.3.3.4 Sprint 3

El objetivo principal de este sprint es trasladar las configuraciones de **Artifactory** y **Bitbucket** al ambiente productivo, asegurando que ambas herramientas estén plenamente operativas y alineadas con las políticas de la entidad financiera. Además, se construirá la **receta de integración continua**, que automatizará el ciclo de integración de código en el nuevo entorno productivo tal cual detalla la Tabla 9.

Tabla 9 Actividades del sprint 3

Tarea	Descripción	Objetivo	Entregables
Instalación y Configuración de Artifactory en un Ambiente Productivo	Instalación y configuración de Artifactory en producción para gestionar artefactos generados en CI/CD, cumpliendo con las políticas de seguridad y almacenamiento.	Tener Artifactory completamente operativo en producción para gestionar y almacenar artefactos de CI/CD de forma segura.	Artifactory instalado y configurado en producción, integrado con Jenkins.
Instalación y Configuración de Bitbucket en un Ambiente Productivo	Instalación y configuración de Bitbucket en producción para gestionar los repositorios de código fuente, integrando con Jenkins mediante webhooks para activar pipelines.	Tener Bitbucket operando en producción, gestionando el código fuente y activando los pipelines de CI/CD de manera automática.	Bitbucket instalado y configurado en producción, con webhooks y políticas de seguridad establecidas.
Construcción de la Receta de Integración Continua	Construcción de la receta de integración continua en Jenkins para compilar, probar, y analizar el código automáticamente tras cada commit en Bitbucket.	Tener un pipeline de integración continua funcional en Jenkins, validando automáticamente el código y almacenando artefactos en Artifactory.	Receta de integración continua configurada y validada en Jenkins, con notificaciones automáticas.

Nota: Elaborado por el autor

Resultados Esperados del Sprint 3

- **Artifactory** y **Bitbucket** completamente operativos en producción, alineados con las políticas de seguridad y gestión de la entidad financiera.
- **Pipelines de Jenkins** configurados para integrarse con Bitbucket y Artifactory, permitiendo que los cambios en el código disparen automáticamente la compilación y almacenamiento de artefactos.
- Receta de integración continua implementada y validada en producción, asegurando que cada commit en Bitbucket pase por un proceso de validación antes de ser aceptado.

Este sprint es clave para garantizar que las herramientas fundamentales de gestión de código y artefactos estén operativas en producción, y que la receta de integración continua funcione correctamente para validar los cambios de código antes de su despliegue

3.2.3.3.5 Sprint 4

El Sprint 4 se centrará en la instalación y configuración de Jenkins y sus componentes en un entorno productivo, así como en la creación de la receta de despliegue continuo que automatizará el proceso de despliegue en producción. Este sprint establecerá el flujo completo de CI/CD, permitiendo que los artefactos generados en el entorno de preproducción se desplieguen de manera controlada y segura en los entornos de producción como se muestra en la Tabla 10.

Tabla 10 Actividades del sprint 4

Tarea	Descripción	Objetivo	Entregables
Instalación y Configuración de Jenkins y sus Componentes en un Ambiente Productivo	Instalación y configuración de Jenkins en producción para coordinar y ejecutar tareas CI/CD, integrando con Bitbucket, Artifactory, y SonarQube.	Tener Jenkins operativo en producción, listo para gestionar los pipelines CI/CD de manera segura y eficiente.	Jenkins instalado y configurado en producción, conectividad validada y roles/permisos definidos.
Construcción de la Receta de Despliegue Continuo	Construcción de la receta de despliegue continuo en Jenkins para desplegar artefactos validados automáticamente en producción tras aprobaciones manuales.	Automatizar el despliegue de aplicaciones en producción, asegurando que los artefactos validados pasen por un flujo controlado y seguro.	Receta de despliegue continuo configurada y validada en Jenkins, con un pipeline operativo y aprobado.

Nota: Elaborado por el autor

Resultados Esperados del Sprint 4

- Jenkins completamente instalado y configurado en producción, gestionando los pipelines de integración y despliegue continuo.
- Los roles y permisos dentro de Jenkins están definidos, asegurando que solo los equipos autorizados puedan realizar despliegues en producción.

- La receta de despliegue continuo está configurada y validada, permitiendo que los artefactos aprobados sean desplegados automáticamente en producción.
- Las aprobaciones manuales y las políticas de rollback están implementadas para garantizar que los despliegues se realicen de manera segura y eficiente, con un proceso claro para manejar errores.

Este sprint establece el corazón del proceso de CI/CD en producción, asegurando que todos los despliegues de software estén automatizados y controlados de manera segura, con las aprobaciones y validaciones necesarias para garantizar la calidad y seguridad del software en producción.

3.2.3.3.5 Sprint 5

Este sprint se centrará en formalizar el modelo operativo de los pipelines de integración y despliegue continuos, documentar los procesos, configurar los repositorios de artefactos para el equipo piloto en Artifactory, y gestionar los usuarios, roles y permisos en todas las herramientas clave de la plataforma DevOps: Jenkins, Bitbucket, Artifactory y SonarQube como se detalla en la Tabla 11.

Tabla 11 Actividades del sprint 5

Tarea	Descripción	Objetivo	Entregables
Creación de la Documentación sobre el Modelo Operativo de los Pipelines de Integración y Despliegue Continuos	Documentación del modelo operativo de los pipelines de CI/CD, estandarizando los flujos de trabajo y sirviendo como referencia para futuros equipos.	Crear un documento formal que describa el flujo de CI/CD y las responsabilidades de cada equipo.	Documento detallado del modelo operativo de CI/CD. Diagrama de flujo del proceso completo.
Configuración de los Repositorios de Artefactos del Equipo Piloto en Artifactory	Configuración de repositorios de artefactos en Artifactory para los proyectos del equipo piloto, con políticas de seguridad y gestión adecuadas.	Tener repositorios dedicados en Artifactory para el equipo piloto, con políticas de seguridad y gestión adecuadas.	Repositorios de artefactos configurados para el equipo piloto. Políticas de acceso y limpieza configuradas.
Creación y Configuración de Usuarios, Roles y Permisos en	Creación de usuarios, roles y permisos en las herramientas clave	Asegurar que todos los usuarios tengan roles y permisos adecuados,	Configuración de usuarios, roles y permisos en Jenkins, Bitbucket,

Jenkins, Bitbucket, Artifactory y SonarQube	para asegurar el acceso adecuado y el cumplimiento de las políticas de seguridad.	respetando las políticas de seguridad de la entidad.	las de la Documentación de las políticas de acceso.	Artifactory y SonarQube.
---	---	--	---	--------------------------

Resultados Esperados del Sprint 5

- **Documentación completa** del modelo operativo de los pipelines de integración y despliegue continuo, con instrucciones claras y diagramas visuales.
- **Repositorios configurados en Artifactory** para el equipo piloto, asegurando la gestión adecuada de los artefactos generados durante los pipelines.
- **Usuarios, roles y permisos** configurados en Jenkins, Bitbucket, Artifactory y SonarQube, alineados con las políticas de seguridad y cumplimiento de la entidad financiera.

Este sprint asegura que el proceso de CI/CD esté documentado, estandarizado y operando bajo las políticas de seguridad establecidas. Además, la configuración adecuada de roles y permisos garantiza que solo los usuarios autorizados tengan acceso a los recursos críticos del sistema.

3.2.3.3.6 Sprint 6

El objetivo de este sprint es migrar el **código fuente** del equipo piloto a **Bitbucket** y configurar los **pipelines en Jenkins** para automatizar el proceso de integración y despliegue continuo. Esta es una fase crucial para poner en marcha el uso práctico de la plataforma DevOps por parte del equipo piloto y validar que los flujos automatizados funcionen sin problemas en el entorno productivo tal como se detalla en la Tabla 12.

Tabla 12 Actividades del sprint 6

Tarea	Descripción	Objetivo	Entregables
Migración del Código Fuente del Equipo Piloto a Bitbucket	Migración del código fuente del equipo piloto a Bitbucket para centralizar su gestión y aprovechar las integraciones automáticas con Jenkins.	Migrar el código a Bitbucket, asegurando que los proyectos estén bajo control de versiones y Jenkins reciba notificaciones para activar pipelines de CI/CD.	Código migrado a Bitbucket, ramas protegidas y webhooks configurados para integración con Jenkins.

Creación y Configuración de Pipelines en Jenkins	Creación y configuración de pipelines de CI/CD en Jenkins para los proyectos del equipo piloto, integrando con Bitbucket, Artifactory y SonarQube.	Automatizar el ciclo de desarrollo desde la integración continua hasta el despliegue, usando pipelines en Jenkins.	Pipelines de CI/CD configurados y operativos en Jenkins, validados y cumpliendo con las políticas de la entidad.
--	--	--	--

Nota: Elaborado por el autor

Resultados Esperados del Sprint 6

- **Código fuente** migrado con éxito a Bitbucket, con una gestión centralizada del control de versiones.
- **Pipelines de Jenkins** completamente configurados y validados para la integración y despliegue continuo.
- **Automatización del proceso de CI/CD**, permitiendo una mayor eficiencia en el desarrollo y una reducción en los tiempos de entrega.
- **Notificaciones automáticas** que informen a los equipos del estado de los pipelines en tiempo real, permitiendo una rápida respuesta ante problemas.

Este sprint representa el paso final hacia la operatividad plena del equipo piloto en la plataforma DevOps, asegurando que tanto la gestión del código como los procesos de CI/CD estén automatizados y bajo control.

3.2.3 Fundamentos teóricos utilizados

3.2.3.1 Aplicaciones web en el sector bancario

Las aplicaciones web son programas de software accesibles a través de un navegador que permiten a los usuarios realizar diversas funciones en línea, como realizar transacciones, gestionar cuentas y acceder a información financiera. En el contexto de las entidades bancarias, estas aplicaciones son fundamentales para ofrecer a sus clientes servicios digitales eficientes y seguros, lo que les permite operar de manera conveniente en cualquier lugar y en cualquier momento (González y Rodríguez, 2020). Las aplicaciones web bancarias generalmente utilizan tecnologías como HTML, CSS y JavaScript en el lado del cliente, y lenguajes de programación como Java o .NET en el lado del servidor, e integran además medidas de seguridad robustas para proteger la

información del cliente y garantizar la confidencialidad y la integridad de las transacciones (Díaz et al., 2021).

La implementación de aplicaciones web en el sector bancario no solo mejora la experiencia del cliente, sino que también optimiza los procesos internos, lo que permite a las instituciones financieras reducir costes operativos y aumentar la eficiencia en la gestión de sus servicios (Martínez y Gómez, 2019).

3.2.3.2 DevOps

DevOps es un enfoque que une las áreas de “development” (desarrollo) y “operations” (operaciones), y promueve la colaboración entre estos dos equipos a lo largo del ciclo de vida del software, desde la creación hasta la puesta en producción. Esta práctica tiene como objetivo aumentar la eficiencia, mejorar la calidad del software y reducir los tiempos de entrega. El término fue acuñado en 2008 por Patrick Debois y Yens Wasna durante la charla titulada “Agile Infrastructure”, que sentó las bases para la filosofía y las prácticas actuales de DevOps (Debois, 2008).

Desde entonces, DevOps ha evolucionado hasta convertirse en un enfoque clave para el desarrollo de software, integrando principios de agilidad y automatización con el fin de facilitar la entrega continua de software y la mejora de los servicios digitales (Kim et al., 2016; Bass et al., 2015).

3.2.3.3 Integración Continua

La integración continua es una práctica de desarrollo de software en la que los desarrolladores integran su trabajo con frecuencia, generalmente varias veces al día. Cada integración se verifica automáticamente mediante la ejecución de pruebas y una compilación del software, con el fin de detectar errores rápidamente y mejorar la calidad del software. Este enfoque permite que los equipos reduzcan el riesgo, disminuyan la duplicación de trabajo y aceleren la entrega de software al detectar problemas de integración desde las primeras etapas (Fowler, 2006; Humble & Farley, 2010). La CI fomenta un ciclo de desarrollo más eficiente y controlado, facilita la colaboración entre

equipos y garantiza que el código siempre esté en un estado desplegable (Duvall et al., 2007).

3.2.3.4 Despliegue Continuo

El despliegue continuo (CD) es una extensión de la integración continua en la que los cambios de código que pasan las pruebas automatizadas se despliegan automáticamente en producción. El objetivo es lanzar versiones de software de forma rápida, fiable y frecuente, minimizando la intervención humana y asegurando que el código siempre esté listo para desplegarse. Este enfoque reduce significativamente el tiempo entre el desarrollo y la entrega, promoviendo ciclos de realimentación más cortos y una mejora continua del producto (Humble & Farley, 2010; Chen, 2015). El despliegue continuo también ayuda a las organizaciones a reaccionar más rápidamente a las demandas del mercado y a reducir los errores al desplegar pequeños cambios con frecuencia en lugar de grandes actualizaciones (Shahin et al., 2017).

3.2.3.5 Scrum

Scrum es un marco ágil que se utiliza para la gestión de proyectos y se centra en la entrega incremental e iterativa de productos. Originado en el ámbito del desarrollo de software, Scrum permite a los equipos adaptarse rápidamente a los cambios y optimizar la colaboración a través de roles definidos, eventos y artefactos. Los roles principales de Scrum son el propietario del producto, quien define las prioridades y los requisitos del producto; el scrum máster, que facilita el proceso y elimina impedimentos; y el equipo de desarrollo, que es responsable de la entrega del trabajo. El marco se organiza en ciclos de trabajo llamados “sprints”, que generalmente tienen una duración de dos a cuatro semanas, al final de los cuales se entrega un incremento funcional del producto (Schwaber & Sutherland, 2017; Sutherland, 2014). Scrum promueve la transparencia, la inspección y la adaptación, lo que permite a los equipos mejorar continuamente sus procesos y productos (Beck et al., 2001).

3.2.4 Implementación de las áreas, procesos, sistemas y buenas practicas

3.2.4.1 *Capacitación para Desarrolladores sobre el uso de Bitbucket integrado con Jenkins*

3.2.4.1.1 Introducción a Bitbucket

Bitbucket es una plataforma de control de versiones basada en Git, utilizada para gestionar el código fuente de manera eficiente y colaborativa. En la entidad financiera, Bitbucket se implementa como herramienta clave para manejar los repositorios de código, integrándose con Jenkins para automatizar procesos de integración continua.

3.2.4.1.2 Objetivos de la capacitación

El objetivo principal de la capacitación es que los desarrolladores comprendan cómo usar Bitbucket de manera eficiente, integrando el control de versiones con Jenkins para la automatización del ciclo de desarrollo. Se busca que:

- Los desarrolladores aprendan a crear, gestionar y colaborar en repositorios de código.
- Apliquen buenas prácticas en el manejo de ramas (branches) y fusiones (merges).
- Comprendan el flujo de trabajo GitFlow y su implementación en Bitbucket.
- Sepan cómo se integran los repositorios de Bitbucket con Jenkins para ejecutar pipelines automáticos de pruebas y despliegue.

3.2.4.1.3 Contenido de la capacitación

- **Introducción a Git y Bitbucket:**
 - Conceptos fundamentales de Git: commit, push, pull, merge.
 - Diferencias entre repositorios locales y remotos.
 - Navegación por la interfaz de Bitbucket y creación de un nuevo repositorio.
- **Administración de Repositorios:**
 - Creación y configuración de repositorios en Bitbucket.
 - Asignación de permisos y roles a los colaboradores.
 - Configuración de ramas predeterminadas y protecciones de ramas.

- **Uso de Branches y GitFlow:**
 - Explicación de GitFlow: ramas de desarrollo (develop), características (feature), correcciones rápidas (hotfix) y versiones de lanzamiento (release).
 - Buenas prácticas para el manejo de ramas: cómo crear y trabajar en ramas de forma aislada.
 - Creación y uso de ramas en Bitbucket.
 - Resolución de conflictos y fusiones de código entre ramas.
- **Pull Requests:**
 - Introducción al concepto de Pull Request: qué es y cómo se usa en Bitbucket.
 - Cómo revisar y comentar cambios de código.
 - Proceso de aprobación de Pull Requests y fusión de código a la rama principal.
- **Integración de Bitbucket con Jenkins:**
 - **Configuración de Webhooks:**
 - Explicación de qué son los webhooks y cómo se configuran en Bitbucket para que notifiquen a Jenkins cuando hay cambios en el repositorio.
 - Crear un webhook en Bitbucket que active un trabajo (job) en Jenkins cada vez que se realiza un commit o merge.
 - **Conexión de Jenkins con Bitbucket:**
 - Configuración de Jenkins para conectarse a un repositorio de Bitbucket.
 - Configuración de jobs en Jenkins que se disparen automáticamente al detectar cambios en el código del repositorio.
 - Integración de Jenkins con Bitbucket para ejecutar pruebas automáticas y otros procesos de validación.
 - **Notificaciones en Bitbucket:**
 - Jenkins puede enviar el estado de cada build (éxito o fallo) de regreso a Bitbucket, para que los desarrolladores tengan visibilidad sobre el estado de la compilación o pruebas desde la interfaz de Bitbucket.

- **Buenas Prácticas:**
 - Uso adecuado de ramas: mantener un historial claro y ordenado.
 - Revisiones de código mediante Pull Requests para asegurar la calidad.
 - Aplicación de políticas de protección de ramas para evitar la fusión de código no revisado.
 - Uso de Jenkins para automatizar el proceso de integración continua, ejecutando pruebas automáticas antes de la fusión a la rama principal.

3.2.4.1.4 Metodología de la capacitación

- **Taller práctico:** Los desarrolladores participarán en ejercicios prácticos, creando y gestionando ramas, fusionando cambios y configurando webhooks en Bitbucket para activar Jenkins.
- **Revisión de código:** Simulaciones de Pull Requests donde los desarrolladores revisarán código en equipo para fomentar la colaboración y mejorar la calidad del código.
- **Integración de Jenkins:** Ejemplos prácticos de cómo configurar Jenkins para que se integre con Bitbucket y automatice procesos de validación de código.

3.2.4.1.5 Resultados esperados

- Desarrolladores capacitados en el uso eficiente de Bitbucket y en cómo integrarlo con Jenkins.
- Implementación de buenas prácticas en el manejo de repositorios y en la revisión de código.
- Optimización de la automatización del flujo de trabajo con Jenkins para garantizar la calidad del código antes de su fusión a la rama principal.
- Reducción de errores humanos gracias a la automatización de la integración continua, gestionada desde Jenkins.

3.2.4.2 *Capacitación de la herramienta Jenkins para AgileOps, seguridad TI, centro de operaciones de seguridad, gobierno de TI y desarrolladores.*

3.2.4.2.1 Introducción a Jenkins

Jenkins es una herramienta de automatización de código abierto utilizada principalmente para la integración continua (CI) y la entrega continua (CD) de

aplicaciones. Permite automatizar diferentes partes del ciclo de vida del desarrollo del software, facilitando la construcción, prueba y despliegue de aplicaciones de forma eficiente y segura.

3.2.4.2.2 Objetivos de la Capacitación

El objetivo de esta capacitación es que los diferentes equipos de la entidad financiera, como Operaciones TI, Seguridad TI, el Centro de Operaciones de Seguridad, Gobierno de TI y los desarrolladores, comprendan cómo utilizar Jenkins en sus roles específicos dentro del ciclo de integración y despliegue de software. Esto incluye:

- **Operaciones TI:** Cómo gestionar la infraestructura y automatizar despliegues.
- **Seguridad TI y Centro de Operaciones de Seguridad:** Cómo revisar y garantizar que las prácticas de seguridad se integren en el proceso de CI/CD.
- **Gobierno de TI:** Monitoreo y aprobación de procesos automatizados.
- **Desarrolladores:** Uso de Jenkins para la integración continua y pruebas automáticas.

3.2.4.2.3 Contenidos de la Capacitación para cada Área

3.2.4.2.3.1 Capacitación para Operaciones TI

- **Introducción a la Infraestructura de Jenkins:**
 - Explicación del modelo maestro-esclavo de Jenkins.
 - Configuración de nodos para ejecutar pipelines en diferentes entornos.
 - Gestión y mantenimiento de los servidores Jenkins (logs, memoria, espacio en disco, etc.).
- **Automatización de Despliegues con Jenkins:**
 - Creación de pipelines que automaticen el despliegue de aplicaciones a entornos de producción y preproducción.
 - Integración de Jenkins con herramientas como Ansible, Docker y Kubernetes para la orquestación y despliegue de infraestructura.
 - Monitoreo de los pipelines de Jenkins: cómo identificar errores y solucionarlos rápidamente.
- **Buenas Prácticas para la Infraestructura:**
 - Cómo asegurar la estabilidad de Jenkins en entornos de producción.

- Configuración de alertas y notificaciones en caso de fallos en los pipelines.
- Uso de backups y estrategias de recuperación ante desastres.

3.2.4.2.3.2 Capacitación para Seguridad TI y Centro de Operaciones de Seguridad

- **Integración de Seguridad en Jenkins:**
 - Configuración de controles de acceso y permisos para asegurar que solo usuarios autorizados puedan realizar cambios críticos.
 - Uso de plugins de seguridad como **OWASP Dependency-Check**, **Fortify** y **SonarQube** para análisis de vulnerabilidades y calidad del código.
 - Escaneo automatizado de vulnerabilidades de seguridad en el código antes de que sea desplegado.
- **Monitoreo de Prácticas de Seguridad en Jenkins:**
 - Integración de Jenkins con herramientas de seguridad y monitoreo como **ELK Stack** para la recolección de logs.
 - Configuración de alertas en caso de fallos o violaciones de seguridad durante la ejecución de pipelines.
 - Verificación automática del cumplimiento de políticas de seguridad antes de los despliegues.

3.2.4.2.3.3 Capacitación para Gobierno de TI

- **Monitoreo de Pipelines y Auditoría:**
 - Cómo utilizar Jenkins para mantener un registro detallado de todas las actividades de desarrollo y despliegue.
 - Generación de reportes y trazabilidad de cambios de código: ¿Quién realizó qué cambio y cuándo?
 - Uso de plugins de Jenkins para la creación de auditorías automáticas que garanticen el cumplimiento de normativas internas y regulatorias.
- **Aprobación Automatizada de Procesos de Despliegue:**
 - Configuración de etapas de aprobación manual dentro de los pipelines antes de realizar un despliegue.
 - Implementación de aprobaciones basadas en políticas, donde los miembros del equipo de Gobierno de TI pueden revisar y aprobar los pasos críticos del despliegue.

3.2.4.2.3.4 Capacitación para Desarrolladores

- **Integración Continua con Jenkins:**
 - Cómo configurar pipelines de Jenkins para automatizar la construcción y prueba de aplicaciones desde Bitbucket u otros sistemas de control de versiones.
 - Uso de **Maven**, **Gradle** u otras herramientas de construcción dentro de Jenkins.
 - Configuración de pruebas unitarias y pruebas de integración automatizadas en cada commit de código.
- **Pruebas Automáticas en Jenkins:**
 - Ejecución de pruebas automáticas cada vez que se realiza un commit en el código fuente.
 - Cómo integrar Jenkins con herramientas de pruebas como **JUnit**, **Selenium** o **Jest** para asegurar la calidad del software.
 - Análisis de cobertura de código y revisión de métricas con herramientas como **SonarQube**.
- **Despliegue Continuo para Desarrolladores:**
 - Cómo configurar Jenkins para que automáticamente despliegue aplicaciones en ambientes de prueba o producción al aprobar un Pull Request.
 - Ejemplos de pipelines que realizan la validación del código, pruebas automáticas y despliegue continuo en un solo flujo.

3.2.4.2.4 Metodología de la Capacitación

- **Talleres prácticos:** Cada equipo participará en simulaciones y ejercicios específicos a sus roles. Esto incluirá la creación y configuración de pipelines, la integración de herramientas de seguridad y la gestión de infraestructura automatizada.
- **Ejercicios colaborativos:** Los equipos de desarrollo y operaciones trabajarán juntos en la configuración de Jenkins, integrando sus necesidades para realizar despliegues seguros y eficientes.

- **Demostraciones en vivo:** Se mostrarán ejemplos de pipelines automatizados que integren las tareas de los distintos equipos, desde el código hasta el despliegue final.

3.2.4.2.5 Resultados Esperados

- **Operaciones TI:** Capacitados en la administración de Jenkins, capaces de gestionar la infraestructura y automatizar los procesos de despliegue.
- **Seguridad TI y Centro de Operaciones de Seguridad:** Capacitados en el uso de Jenkins para integrar y automatizar prácticas de seguridad en el proceso de desarrollo y despliegue.
- **Gobierno de TI:** Capacitado en monitoreo y auditoría automatizada, asegurando el cumplimiento de las normativas internas y externas.
- **Desarrolladores:** Capacitados para utilizar Jenkins en la automatización de la integración continua y pruebas, mejorando la calidad y velocidad de entrega de software.

Este enfoque asegura que todos los equipos entiendan su rol dentro del ciclo de vida de DevOps, utilizando Jenkins como una herramienta integral para automatizar, asegurar y auditar los procesos de desarrollo y despliegue de aplicaciones.

3.2.4.3 Plan piloto de uso de plataforma de despliegue e integración continua

3.2.4.3.1 Introducción

El **Plan Piloto** tiene como objetivo probar la efectividad de la plataforma de **Despliegue e Integración Continua** dentro de un entorno controlado, una vez que la plataforma ya esté completamente implementada. Este piloto permitirá evaluar su funcionamiento, detectar posibles problemas, y recibir retroalimentación de los equipos antes de expandir el uso de la plataforma a toda la entidad financiera.

La plataforma integra herramientas como **Jenkins** para la automatización del despliegue e integración continua, **Bitbucket** para la gestión del código, **SonarQube** para el análisis de calidad del código, y **Artifactory** para la gestión de artefactos.

3.2.4.3.2 Objetivos del Plan Piloto

- Probar el uso de la plataforma en un entorno controlado, asegurando que la integración y despliegue continuo se ejecuten de manera eficiente.
- Evaluar cómo los desarrolladores y los equipos de operaciones interactúan con la plataforma.
- Detectar problemas técnicos y realizar ajustes necesarios antes de una implementación general.
- Asegurar que los procesos automatizados de despliegue cumplan con los requisitos de calidad, seguridad, y tiempo de la entidad financiera.

3.2.4.3.3 Alcance del Piloto

El piloto se centrará en:

- **Equipo de producto:** Un equipo específico de desarrolladores que trabajará en una aplicación web de la entidad bancaria.
- **Aplicación web:** Una aplicación desarrollada en Angular, que será gestionada y desplegada a través de la plataforma DevOps.
- **Entorno de pruebas:** El piloto se llevará a cabo en el entorno de preproducción para evitar impacto en producción.
- **Duración:** El piloto se realizará en un solo **sprint** de duración aproximada de **3 semanas**.

3.2.4.3.4 Metodología del Plan Piloto

El piloto será implementado en un solo sprint, una vez que la plataforma esté completamente operativa. Las fases del piloto incluirán las siguientes actividades:

3.2.4.3.4.1 Planificación y Configuración

- **Definir las reglas de uso de la plataforma:** Se definirán las políticas que guiarán el uso de Jenkins, Bitbucket, SonarQube y Artifactory, asegurando que todas las etapas del ciclo de vida del software estén alineadas con los estándares internos.
- **Capacitación breve:** Los desarrolladores y el equipo de operaciones recibirán una capacitación rápida sobre cómo interactuar con los pipelines configurados,

cómo realizar despliegues automáticos y cómo gestionar las integraciones con Bitbucket.

- **Revisión de los Pipelines:** Verificación de que los pipelines estén correctamente configurados para construir, probar, y desplegar el código de manera automática.

Estos pipelines incluirán:

- Construcción del código desde Bitbucket.
- Ejecución de pruebas unitarias.
- Análisis de calidad con SonarQube.
- Gestión de artefactos con Artifactory.
- Despliegue automático en preproducción.

3.2.4.3.4.2. Ejecución del Piloto

- **Despliegues controlados:** Se realizará un despliegue inicial para validar que la plataforma puede manejar la construcción y despliegue de la aplicación de Angular en el entorno de preproducción.
- **Automatización de pruebas:** Durante el piloto, se validará la ejecución automática de pruebas unitarias y de integración en cada commit o merge de código, verificando que los resultados se reporten adecuadamente.
- **Monitoreo de métricas:** Se monitorearán indicadores clave como el tiempo de construcción, la tasa de fallos en despliegue, y la calidad del código con SonarQube, para asegurar que los resultados cumplan con las expectativas de los equipos involucrados.

3.2.4.3.4.3 Evaluación y Retroalimentación

- **Revisión de resultados:** Al final del sprint, se reunirán los equipos involucrados (desarrolladores, operaciones TI, seguridad TI y gobierno de TI) para evaluar los resultados del piloto.
 - Métricas clave:
 - **Tiempo de construcción y despliegue:** Reducción esperada en los tiempos de integración y despliegue.
 - **Calidad del código:** Evaluación del código mediante SonarQube para detectar problemas críticos de seguridad y calidad.

- **Tasa de éxito en despliegues:** Número de despliegues exitosos versus fallidos durante el piloto.
 - Se identificará si es necesario realizar ajustes en la configuración de los pipelines o en la infraestructura de Jenkins.
- **Retroalimentación:** Se recogerán los comentarios de los participantes para identificar áreas de mejora y posibles problemas en el flujo de trabajo.
- **Documentación:** Se elaborará un documento con las mejores prácticas recomendadas para el uso de la plataforma, basado en las lecciones aprendidas durante el piloto.

3.2.4.3.4.5 Roles y Responsabilidades

- **Desarrolladores:** Serán responsables de hacer commits, gestionar ramas y realizar Pull Requests en Bitbucket. También usarán Jenkins para validar y desplegar su código.
- **Operaciones TI:** Encargados de gestionar la infraestructura de Jenkins, incluyendo el mantenimiento de los servidores y la resolución de cualquier problema relacionado con la disponibilidad de la plataforma.
- **Seguridad TI:** Revisarán los procesos de automatización para asegurar que las políticas de seguridad se cumplan, incluyendo escaneos de vulnerabilidades y la validación del código antes del despliegue.
- **Gobierno de TI:** Asegurará que los procesos automatizados cumplan con los estándares internos de la entidad, y aprobará cualquier cambio crítico en los pipelines.

3.2.4.3.4.6 Metodología de Evaluación

La evaluación del piloto se centrará en las siguientes métricas clave:

- **Tiempo de integración y despliegue:** Medir el tiempo promedio que toma un commit de código en llegar a producción.
- **Calidad del código:** Evaluación de la cantidad de vulnerabilidades y problemas detectados por SonarQube en cada despliegue.
- **Tasa de fallos en despliegue:** Número de despliegues fallidos durante el piloto, junto con la causa raíz de dichos fallos.

- **Satisfacción de los equipos:** Recopilar feedback cualitativo de los equipos sobre el uso de la plataforma, enfocándose en la facilidad de uso, problemas encontrados y recomendaciones para la mejora.

3.2.4.3.4.7 Resultados Esperados

- Validación de que la plataforma automatizada puede gestionar de manera eficiente los despliegues e integraciones continuas dentro de un entorno controlado.
- Reducción de tiempos de despliegue y mejora en la calidad del código debido a las pruebas automatizadas y análisis de código.
- Recepción de feedback constructivo que permitirá mejorar la plataforma antes de su implementación general en otros equipos de la entidad.

3.2.4.3.4.8 Conclusión

El plan piloto de **Despliegue e Integración Continua** se llevará a cabo en un solo sprint de prueba, permitiendo a los equipos familiarizarse con la plataforma y probar sus funcionalidades clave. Esto ayudará a ajustar la plataforma para su implementación a mayor escala y garantizar que los procesos automatizados estén alineados con los objetivos de la entidad, mejorando la eficiencia y reduciendo el tiempo de entrega de software.

3.2.4.4 Generación de nuevo proceso de despliegue de software usando la nueva plataforma

3.2.4.4.1 Introducción

El nuevo proceso de **Despliegue de Software** basado en la plataforma de **Integración y Despliegue Continuo (CI/CD)** es una mejora significativa frente a los métodos anteriores de despliegue manual. Este proceso automatizado no solo reduce los tiempos de entrega, sino que también minimiza los errores humanos y asegura que el software pase por validaciones automatizadas que garantizan la calidad, la seguridad y el cumplimiento de las regulaciones internas de la entidad bancaria.

Este nuevo proceso será formalizado dentro de los **procesos operativos de TI** y estará alineado con las regulaciones establecidas por el área de **Gobierno de TI, Seguridad TI**, y las normativas regulatorias de la industria bancaria en el país.

3.2.4.4.2 Objetivo del Proceso de Despliegue

El objetivo principal del nuevo proceso es:

- **Automatizar** el flujo de despliegue de software desde el desarrollo hasta producción, asegurando que cada paso cumpla con los estándares de calidad y seguridad.
- **Reducir el tiempo** de despliegue de nuevas funcionalidades y correcciones de errores.
- **Asegurar el cumplimiento** de las políticas de seguridad y normativas regulatorias que rigen la entidad financiera.

3.2.4.4.3 Estructura del Nuevo Proceso de Despliegue

El nuevo proceso de despliegue estará estructurado en las siguientes fases:

3.2.4.4.3.1 Fase de Pre-Despliegue

1. Preparación del Paquete de Liberación:

- El equipo de desarrollo integra los cambios en un repositorio de **Bitbucket**.
- Jenkins detecta automáticamente los commits y lanza un pipeline de integración continua.
- Durante esta fase se realizan las siguientes acciones automatizadas:
 - **Compilación del código:** El código es compilado automáticamente por Jenkins.
 - **Pruebas unitarias:** Jenkins ejecuta pruebas automatizadas sobre el código.
 - **Análisis de calidad y seguridad:** Se realizan análisis de código estático usando herramientas como **SonarQube** para detectar vulnerabilidades y errores en el código.
- Si el código pasa todas las validaciones, se genera un **paquete de liberación** listo para despliegue.

2. Revisión y Aprobaciones:

- Antes de ser desplegado en producción, el paquete debe pasar por una revisión formal.

- **Gobierno de TI:** Revisa el proceso completo y aprueba el paquete de liberación desde la perspectiva del cumplimiento normativo.
- **Seguridad TI:** Realiza una revisión final de los artefactos, verificando que se han seguido las políticas de seguridad y que no existen vulnerabilidades abiertas en el código.

3. Programación de Despliegue:

- Se programa el despliegue dentro del calendario compartido de pases a producción, asegurando que no haya conflictos con otros despliegues simultáneos.

3.2.4.4.3.2 Fase de Despliegue

1. Ejecutar Pipeline de Despliegue:

- Una vez aprobado y programado, el pipeline de Jenkins inicia el despliegue automático del paquete en el ambiente productivo. El pipeline sigue los siguientes pasos:
 - **Obtener el artefacto desde Artifactory:** Jenkins obtiene el paquete de código aprobado desde Artifactory.
 - **Despliegue en Servidores de Producción:** El código es desplegado en los servidores de producción, utilizando herramientas de orquestación como **Ansible** o **Docker**, según sea necesario.
 - **Configuración de Parámetros Sensibles:** Se asegura que las variables de entorno, claves API, y demás parámetros sensibles se configuren correctamente, siguiendo las políticas de seguridad.

2. Validación de Despliegue:

- Después del despliegue, se ejecutan pruebas de validación que confirman que la aplicación está operando correctamente.
- **Pruebas de Integración:** Se realizan pruebas automatizadas de integración para verificar que el sistema funcione correctamente con otros componentes del entorno bancario.
- **Monitoreo de la Infraestructura:** El equipo de **Operaciones TI** monitorea el estado de los servidores y la infraestructura en tiempo real para asegurar que el despliegue no cause interrupciones.

3. Reporte de Resultados:

- Jenkins genera un reporte detallado del estado del despliegue y las pruebas de validación. Este reporte se envía automáticamente al equipo de desarrollo, **Seguridad TI**, y **Gobierno de TI** para su revisión.

3.2.4.4.3.3 Fase de Post-Despliegue

1. Pruebas Funcionales por el Equipo de Producto:

- El equipo de producto valida manualmente el correcto funcionamiento de las nuevas funcionalidades desde el punto de vista del usuario final.

2. Monitoreo de Producción:

- El equipo de **Operaciones TI** utiliza herramientas de monitoreo como **New Relic** para asegurarse de que el sistema funcione sin problemas en producción.
- Se monitorean métricas clave como la latencia, número de transacciones procesadas, y cualquier error de servidor que pudiera surgir.

3. Aprobación Final:

- **Gobierno de TI** realiza una revisión final del reporte de despliegue y las pruebas de producción, asegurando que el proceso se ha realizado correctamente y que todos los puntos de control establecidos han sido seguidos.
- Si todo está conforme, el despliegue se marca como exitoso y finalizado.

3.2.4.4.4 Formalización del Proceso

El proceso debe ser formalizado dentro de los **procesos operativos** de la entidad financiera. Para ello, se seguirán los siguientes pasos:

3.2.4.4.4.1 Documentación del Proceso

- Se generará una **guía detallada** que documente cada paso del proceso de despliegue, desde la integración del código hasta el monitoreo en producción.
- Esta documentación incluirá diagramas de flujo que muestren las interacciones entre Jenkins, Bitbucket, SonarQube, Artifactory y otras herramientas involucradas.

3.2.4.4.4.2 Políticas de Control y Cumplimiento

- El nuevo proceso debe alinearse con las **políticas internas de seguridad** y las regulaciones impuestas por las autoridades financieras del país, como la **Superintendencia de Banca y Seguros (SBS)**.
- **Gobierno de TI** se encargará de establecer puntos de control obligatorios, como las aprobaciones manuales por parte de Seguridad TI, antes de que se permita cualquier despliegue en producción.

3.2.4.4.4.3 Roles y Responsabilidades

- Cada equipo involucrado (Desarrollo, Operaciones TI, Seguridad TI, y Gobierno de TI) tendrá **roles claramente definidos** en el proceso.
- **Desarrolladores:** Crear y gestionar el código.
- **Operaciones TI:** Configurar y mantener la infraestructura.
- **Seguridad TI:** Revisar la seguridad de los artefactos y los despliegues.
- **Gobierno de TI:** Asegurar que el proceso cumpla con todas las normativas y regulaciones.

3.2.4.4.5 Regulaciones y Cumplimiento

La entidad financiera debe asegurarse de que el nuevo proceso cumpla con las siguientes normativas clave:

- **Normativa de la SBS:** El proceso de despliegue debe cumplir con las directrices establecidas por la Superintendencia de Banca y Seguros, incluyendo auditoría, trazabilidad, y seguridad de la información.
- **Regulaciones de Seguridad:** Se integrarán mecanismos de revisión y escaneo de seguridad automáticos para garantizar que no existan vulnerabilidades en el código que puedan comprometer los sistemas.
- **Control de Acceso y Permisos:** Solo personal autorizado (validado por Seguridad TI y Gobierno de TI) podrá ejecutar o aprobar despliegues en producción.

3.2.4.4.6 Resultados Esperados

- **Automatización y Reducción de Tiempo:** Reducción significativa de los tiempos de despliegue de software, optimizando el ciclo de vida de desarrollo.

- **Cumplimiento de Normativas:** El proceso cumple con todas las regulaciones de la entidad financiera y las normativas externas aplicables.
- **Calidad y Seguridad Aseguradas:** El código que se despliega en producción ha pasado por validaciones exhaustivas, garantizando su calidad y seguridad.

3.2.4.4.7 Conclusión

La implementación de este **nuevo proceso de despliegue de software** basado en la plataforma de CI/CD formaliza el ciclo de desarrollo dentro de la entidad bancaria, asegurando un proceso eficiente, seguro y normativamente aprobado. Este proceso automatizado permitirá a la entidad entregar nuevas funcionalidades más rápidamente, sin comprometer la calidad ni la seguridad del software.

3.3 Evaluación

La implementación de la plataforma de integración y despliegue continuo dentro de una entidad bancaria implica un ahorro en tiempo por lo que se traduce un ahorro en costos. A continuación, se explica más a detalle el ahorro de tiempo.

3.3.1 Evaluación Costo-Beneficio

3.3.1.1 Análisis de costos

Recursos de Personal:

En la Tabla 13 se muestra el costo por cada rol que participo en el proyecto.

Tabla 13 Costo por cada rol en el proyecto

Recurso	Cantidad	Costo mensual (S/)	Costo total en 3.5 meses (S/)
Ingenieros DevOps	3	6,000	63,000
Product Owner	1	10,000	35,000
Líder Técnico	1	8,000	28,000
Total			126,000

Nota: Elaborado por el autor

Recursos de Infraestructura:

Los recursos de infraestructura tienen un costo mensual los cuales se detallan en la Tabla 14.

Tabla 14 Costo por cada recurso de infraestructura usado en el proyecto

Recurso	Cantidad	Costo mensual (\$)	Costo total en 3.5 meses (\$)
Servidores Jenkins	4	800	2,800
Servidor Artifactory	1	200	700
Servidor Sonarqube	1	200	700
Servidor Bitbucket	1	200	700
Total			4,900

Nota: Elaborado por el autor

Costo total del proyecto (3.5 meses):

- **Costo de personal:** S/ 126,000
- **Costo de infraestructura:** \$ 4,900

3.3.1.2 Análisis de beneficios

El beneficio clave del proyecto es la reducción significativa en los tiempos de entrega de software. Antes de la implementación, el despliegue de una aplicación tomaba entre 3 a 5 días, pero con la plataforma DevOps, los tiempos de despliegue se reducen a 1-2 días lo cual refleja un 60% de reducción tal como se detalla en la Tabla 15.

Tiempos de despliegue anual por proyecto:

Tabla 15

Tiempos de despliegue anual por proyecto

Escenario	Despliegues mensuales	Días despliegue	por año	Días al año	Meses al año
Peor caso actual	2	5	120	4	
Mejor caso actual	2	3	72	2.4	
Peor caso nuevo	2	2	48	1.6	

Mejor caso	2	1	24	0.8
nuevo				

Nota: Elaborado por el autor

En la Tabla 16 se muestra el ahorro de tiempo anual estimado por proyecto

Tabla 16 Ahorro de tiempo anual estimado por proyecto

Escenario	Ahorro en meses por proyecto
Peor caso	2 meses
Mejor caso	0.8 meses

Nota: Elaborado por el autor

Si consideramos que la entidad financiera tiene un mínimo de 5 proyectos web que utilizan esta plataforma, el ahorro anual estimado es tal cual se muestra en la Tabla 17.

Tabla 17 Ahorro de tiempo anual por 5 proyectos

Escenario	Ahorro en meses por 5 proyectos
Peor caso	10 meses
Mejor caso	4 meses

Nota: Elaborado por el autor

Beneficios adicionales:

- **Reducción de errores manuales:** Al disminuir la cantidad de pasos manuales, se reduce el margen de error durante el despliegue.
- **Mayor seguridad:** La plataforma automatizada cumple con los requisitos de seguridad de la entidad financiera, lo que asegura que los procesos sean más seguros y repetibles.
- **Estandarización de procesos:** La automatización de Jenkins, Artifactory, Sonarqube y Bitbucket permite una ejecución de despliegues más consistente y predecible.
- **Optimización de la colaboración:** Al reducir la complejidad y mejorar la comunicación entre los equipos, se agilizan los tiempos de entrega y se mejoran los resultados generales.

3.3.2 Interpretación del VAN y del TIR

Este informe detalla el análisis económico del proyecto de implementación de una plataforma de integración y despliegue continuo en una entidad bancaria, considerando un horizonte temporal de **5 años**. El análisis incluye el cálculo del **Valor Actual Neto (VAN)** y la **Tasa Interna de Retorno (TIR)**, ajustando los ahorros en tiempo de los perfiles de **Operaciones TI, Operaciones de Seguridad y Líder Técnico**, basados en una reducción del **60%** del tiempo dedicado a estas funciones críticas, debido a la automatización.

3.3.2.1 Fase de Implementación

La fase de implementación tiene una duración de **3.5 meses**, en los cuales **3 ingenieros DevOps** son los responsables de la construcción de la plataforma. Después de esta fase, un ingeniero DevOps de soporte será empleado permanentemente para mantener la plataforma operativa.

3.3.2.1.1 Costos de Implementación:

A continuación, se detallan en la Tabla 18 los costos de los ingenieros que participaron en el proyecto.

Tabla 18 Costo de ingenieros en fase de implementación

Recurso	Cantidad	Costo mensual (PEN)	Costo total en 3.5 meses (PEN)
Ingenieros DevOps	3	6000	63,000
Ingeniero DevOps de soporte durante la fase de implementación	1	5000	17,500
Costo Total			S/ 80,500

Nota: Elaborado por el autor

En la Tabla 19 se muestran costos mensuales por cada componente de la plataforma de Integración y Despliegue continuo.

Tabla 19 Costo de servidores infraestructura en etapa de implementación

Servidores	Cantidad	Costo mensual (USD)	Costo total en 3.5 meses (USD)
------------	----------	---------------------	--------------------------------

Jenkins	4	200	2800
Artifactory	1	200	700
Sonarqube	1	200	700
Bitbucket	1	200	700
Costo Total			4,900 \$

Nota: Elaborado por el autor

Total inversion: S/80,500 + S/18,375 = S/ 98,875

3.3.2.1.2 Costos Operativos Anuales (después de la implementación)

Después de la fase de implementación, un **ingeniero DevOps de soporte** será empleado con un salario mensual de **S/ 5,000** durante los siguientes **5 años**. Además, los costos de infraestructura de servidores se mantienen anualmente tal cual se muestra en la Tabla 20.

Tabla 20 Costos operativos de personal para soporte de la plataforma

Recurso			Cantidad	Costo mensual (PEN)	Costo anual (PEN)
Ingenieros	Soporte	Plataforma	1	5000	60,000
DevOps					
Costo Total					S/ 60,000

Nota: Elaborado por el autor

En la Tabla 21 se muestra los costos operativos para el soporte de la infraestructura.

Tabla 21 Costos operativos de infraestructura para soporte de la plataforma

Servidores	Cantidad	Costo mensual (USD)	Costo total en 1 año (USD)
Jenkins	4	800	9600
Artifactory	1	200	2400
Sonarqube	1	200	2400
Bitbucket	1	200	2400
Costo Total			16,800 \$

Nota: Elaborado por el autor

Costos totales anuales (mantenimiento): S/60,000 + S/63,000 =S/123,000 por año.

3.3.3 Ahorro Anual Proyectado

El ahorro anual se deriva de la reducción del tiempo dedicado por los equipos de **Operaciones TI, Operaciones de Seguridad, y Líder Técnico** gracias a la automatización del proceso DevOps. Se considera un ahorro del **60%** en cada uno de estos perfiles tal como se detalla en la Tabla 22.

Tabla 22 Ahorro anual proyectado por cada perfil

Perfil	Costo Mensual (PEN)	Ahorro proyectado (60%) (PEN)
Operaciones TI	5000	3600
Operaciones de Seguridad	4000	2400
Lider Tecnico	8000	4800
Ahorro anual Total		S/ 10,800

Nota: Elaborado por el autor

Ahorro total anual: S/10,800×12 = S/129,600 por año.

Este ahorro se proyecta sobre **5 proyectos en el primer año**, con la adición de **2 proyectos nuevos cada año**.

3.3.4 Flujo de Caja Proyectado

En la Tabla 23 se muestra el flujo de caja anual considera los ahorros netos después de descontar los costos de soporte e infraestructura.

Tabla 23 Flujo de caja proyectado

Año	Ahorro mínimo (S/)	Ahorro máximo (S/)	Costos anuales (S/)	Flujo neto mínimo (S/)	Flujo neto máximo (S/)
0	0	0	S/ 98,875	-S/ 98,875	-S/ 98,875
1	S/ 129,600	S/ 129,600	S/ 123,000	S/ 6,600	S/ 6,600
2	S/ 194,400	S/ 194,400	S/ 123,000	S/ 71,400	S/ 71,400
3	S/ 259,200	S/ 259,200	S/ 123,000	S/ 136,200	S/ 136,200
4	S/ 324,000	S/ 324,000	S/ 123,000	S/ 201,000	S/ 201,000
5	S/ 388,800	S/ 388,800	S/ 123,000	S/ 265,800	S/ 265,800

Nota: Elaborado por el autor

3.3.5 Cálculo de VAN (Valor Actual Neto)

El **VAN** se calcula con una tasa de descuento del **10%** y los flujos netos proyectados para cada año. La fórmula del VAN es:

Resultado del VAN:

- **Escenario (Ahorro mínimo y máximo): S/ 1,704,723.15**

3.3.6. Cálculo de TIR (Tasa Interna de Retorno)

- **Escenario (Ahorro mínimo y máximo): 158.90%**

3.3.7. Conclusión

El proyecto de implementación de la plataforma DevOps es **altamente rentable**, con un **VAN positivo** de **S/ 1,704,723.15** y una **TIR** del **158.90%**. Esto demuestra que la inversión se recupera rápidamente, y la automatización ofrece un significativo ahorro de costos a largo plazo en los perfiles de Operaciones TI, Seguridad, y Liderazgo Técnico.

Este análisis confirma la viabilidad económica del proyecto, que no solo genera un retorno excelente, sino que también optimiza los procesos internos de la entidad bancaria, garantizando mayor eficiencia y menores tiempos de entrega de software.

CAPÍTULO IV. REFLEXIÓN CRÍTICA DE LA EXPERIENCIA

4.1. Introducción

En este capítulo se presenta una reflexión crítica sobre la experiencia adquirida durante la implementación de una plataforma de integración y despliegue continuo en una entidad financiera. Se abordaron los aprendizajes obtenidos, las limitaciones encontradas y se analizan las metodologías y procesos empleados, contrastándolos con la literatura especializada.

4.2. Experiencia Adquirida

Durante la realización de este trabajo, se adquirieron conocimientos significativos en las siguientes áreas:

- **Gestión de herramientas DevOps:** La implementación y administración de herramientas como Jenkins, Artifactory y SonarQube consolidaron habilidades técnicas avanzadas, particularmente en su integración para optimizar pipelines CI/CD.
- **Colaboración interdisciplinaria:** La interacción con equipos de Desarrollo, Seguridad TI y Gobierno TI fomentó habilidades de comunicación efectiva y gestión de roles en proyectos complejos.
- **Optimización de procesos:** Se comprendió la importancia de estandarizar y automatizar procesos para reducir tiempos de entrega y garantizar calidad y seguridad.
- **Cumplimiento normativo:** El trabajo fortaleció el entendimiento sobre cómo alinear implementaciones tecnológicas con marcos regulatorios y normativas como ITIL.

4.3. Limitaciones y Desafíos

Durante el desarrollo del proyecto, se presentaron las siguientes limitaciones:

- **Resistencia al cambio:** Algunos equipos mostraron resistencia a adoptar nuevas prácticas, lo que requirió mayor esfuerzo en la capacitación.

- **Burocracia interna:** Los procesos de aprobación manual ralentizaron la implementación de ciertas fases del proyecto.
- **Integración inicial de herramientas:** Configurar una conectividad fluida entre múltiples herramientas presentó desafíos técnicos, principalmente por dependencias de configuración específicas.

4.4. Análisis Crítico de las Metodologías y Procesos

El enfoque metodológico utilizado, basado en principios de **DevOps** e integración continua, demostró ser eficaz para alcanzar los objetivos del proyecto. Sin embargo, se observan puntos de mejora:

- **Comparación con otros autores:** Según Jez Humble y David Farley, en *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*, las organizaciones exitosas reducen significativamente el *Lead Time* al adoptar automatización integral en todas las fases. En comparación, aunque la implementación realizada logró una mejora del 60%, aún depende parcialmente de aprobaciones manuales, lo que limita el impacto.
- **Estandarización y madurez del proceso:** Mientras que frameworks como ITIL ofrecen un marco sólido, podrían complementarse con metodologías ágiles para mejorar la rapidez en la toma de decisiones.

4.5. Enseñanzas y Recomendaciones para Futuras Implementaciones

Este trabajo deja valiosas lecciones, tanto técnicas como operativas:

- La **automatización** no solo mejora la eficiencia, sino que reduce errores humanos.
- Es crucial alinear la tecnología con la cultura organizacional para maximizar su adopción.
- **Capacitar continuamente** a los equipos y fomentar una cultura de colaboración interdisciplinaria es clave para el éxito.

Para futuras implementaciones, se recomienda:

1. Reducir al mínimo los procesos manuales mediante herramientas de aprobación automatizada.
2. Explorar tecnologías más avanzadas para orquestar despliegues, como Kubernetes.
3. Realizar evaluaciones periódicas del proceso para asegurar su alineación con las necesidades del negocio y las tendencias del mercado.
4. Expandir el conjunto de herramientas para involucrar mas el tema de seguridad dentro del proceso de integracion y despliegue continuo. Herramientas que involucren el analisis estatico y dinamico del codigo fuente.

4.6. Conclusión

La experiencia adquirida en este proyecto ha sido enriquecedora, tanto a nivel técnico como profesional. Pese a las limitaciones enfrentadas, el impacto positivo en los tiempos de entrega y la calidad del software son evidentes, lo que refuerza la importancia de las prácticas de DevOps en el sector bancario.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

1. La implementación de una plataforma tecnológica basada en DevOps y CI/CD permitió reducir significativamente el *Lead Time* de despliegue de software.
2. La integración de herramientas como Jenkins, Bitbucket, Artifactory y SonarQube estableció un flujo de trabajo automatizado, mejorando la calidad del software y reduciendo errores humanos.
3. El proceso estandarizado de despliegue continuo demostró ser efectivo en la reducción de riesgos asociados a la liberación de nuevas versiones de software en producción.
4. La adopción de buenas prácticas y capacitación a los equipos de desarrollo y operaciones fomentó una cultura colaborativa, aumentando la eficiencia y la confianza en los procesos.
5. Aunque se lograron avances significativos, las aprobaciones manuales y algunas limitaciones de infraestructura representan áreas de oportunidad para optimizar aún más el proceso.

5.2. Recomendaciones

1. Continuar promoviendo la cultura DevOps en la organización mediante programas de capacitación continua y talleres prácticos para los equipos involucrados.
2. Explorar herramientas más avanzadas para la orquestación de despliegues (por ejemplo, Kubernetes) que permitan mayor escalabilidad y flexibilidad en los entornos de producción.
3. Automatizar completamente las aprobaciones mediante herramientas de validación de políticas, reduciendo el tiempo necesario para las liberaciones sin comprometer la seguridad.
4. Implementar métricas de desempeño más detalladas (como los *Four Key Metrics* de DORA) para identificar cuellos de botella y medir la evolución del proceso a largo plazo.

5. Realizar revisiones periódicas de la infraestructura y herramientas utilizadas, con el objetivo de adaptarlas a las necesidades cambiantes del mercado y de la organización.

5.3. Referencias Bibliográficas

- Chen, Y., Zhang, R., & Liu, T. (2021). The impact of automation on software deployment processes. *Journal of Software Engineering and Applications*, 14(4), 124-135.
- Humble, J., & Farley, D. (2010). *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley.
- Kim, G., Kang, H., & Lee, S. (2020). Improving deployment efficiency through automation: A case study in banking software development. *International Journal of Information Management*, 50, 156-167. <https://doi.org/10.1016/j.ijinfomgt.2019.05.002>
- Debois, P. (2008). Agile Infrastructure. Presentación en la conferencia Agile 2008.
- Kim, G., Humble, J., Debois, P., & Willis, J. (2016). *The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations*. IT Revolution Press.
- Bass, L., Weber, I., & Zhu, L. (2015). *DevOps: A Software Architect's Perspective*. Addison-Wesley Professional.
- Fowler, M. (2006). *Continuous Integration*. MartinFowler.com. <https://martinfowler.com/articles/continuousIntegration.html>
- Humble, J., & Farley, D. (2010). *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley Professional.
- Duvall, P. M., Matyas, S., & Glover, A. (2007). *Continuous Integration: Improving Software Quality and Reducing Risk*. Addison-Wesley Professional.
- Humble, J., & Farley, D. (2010). *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley Professional.
- Chen, L. (2015). Continuous delivery: Huge benefits, but challenges too. *IEEE Software*, 32(2), 50-54. <https://doi.org/10.1109/MS.2015.27>

- Shahin, M., Babar, M. A., & Zhu, L. (2017). Continuous integration, delivery and deployment: A systematic review on approaches, tools, challenges and practices. *IEEE Access*, 5, 3909-3943. <https://doi.org/10.1109/ACCESS.2017.2685629>
- Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Jeffries, R. (2001). *Manifesto for Agile Software Development*. <http://agilemanifesto.org/>
- Schwaber, K., & Sutherland, J. (2017). *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game*. Scrum.org. <https://scrumguides.org/scrum-guide.html>
- Sutherland, J. (2014). *Scrum: The Art of Doing Twice the Work in Half the Time*. Crown Business.
- Díaz, F., Martínez, J., & Pérez, A. (2021). *Desarrollo de aplicaciones web para el sector financiero: Retos y oportunidades*. *Journal of Financial Technology*, 5(2), 34-45.
- González, R., & Rodríguez, M. (2020). *El impacto de las aplicaciones web en la banca moderna*. *Revista de Innovación Financiera*, 10(1), 25-30.
- Martínez, L., & Gómez, S. (2019). *Transformación digital en entidades bancarias: El papel de las aplicaciones web*. *Journal of Banking and Finance*, 8(3), 15-27.

GLOSARIO

1. CI/CD (Integración Continua/Despliegue Continuo):

Prácticas de desarrollo de software que automatizan la integración de cambios de código en un repositorio compartido (CI) y el despliegue automatizado de aplicaciones en entornos de prueba o producción (CD).

2. DevOps:

Conjunto de prácticas y herramientas que integran a los equipos de desarrollo y operaciones con el objetivo de mejorar la colaboración, la entrega continua y la confiabilidad de los sistemas.

3. Lead Time for Changes:

Métrica utilizada para medir el tiempo que transcurre desde que se inicia el desarrollo de un cambio hasta que este se despliega en producción.

4. Pipeline:

Conjunto de pasos automatizados que gestionan el flujo de trabajo desde el desarrollo hasta el despliegue del software, incluyendo la construcción, las pruebas y el despliegue.

5. Jenkins:

Herramienta de automatización de código abierto utilizada para construir, probar y desplegar software. Es ampliamente usada en implementaciones de CI/CD.

6. Bitbucket:

Plataforma de control de versiones basada en Git que permite la colaboración en proyectos de desarrollo y la gestión centralizada del código fuente.

7. Artifactory:

Repositorio de artefactos utilizado para almacenar y gestionar binarios generados durante el proceso de construcción de software, como bibliotecas, ejecutables y contenedores.

8. SonarQube:

Plataforma de análisis de código que permite evaluar la calidad, seguridad y cobertura de pruebas del software desarrollado.

9. ITIL (Information Technology Infrastructure Library):

Conjunto de buenas prácticas para la gestión de servicios de TI que busca alinear los servicios tecnológicos con las necesidades del negocio.

10. Rollback:

Proceso de revertir un despliegue fallido a una versión anterior estable del software para minimizar el impacto en los usuarios.

11. Webhook:

Mecanismo que permite a las aplicaciones enviar notificaciones automáticas a otras aplicaciones cuando ocurre un evento específico, como un cambio en el código fuente.

12. Azure:

Plataforma de servicios en la nube de Microsoft que ofrece soluciones como máquinas virtuales, almacenamiento, bases de datos y servicios DevOps.

13. Ansible:

Herramienta de automatización de TI utilizada para configurar sistemas, gestionar aplicaciones y realizar despliegues de software de manera eficiente y repetible.

14. Feature Branch (Rama de Características):

Modelo de desarrollo en el cual cada nueva funcionalidad o corrección de errores se desarrolla en una rama independiente antes de integrarse en la rama principal.

15. Pull Request (Solicitud de Extracción):

Propuesta de cambios en un repositorio de control de versiones, que debe ser revisada y aprobada por otros desarrolladores antes de fusionarse con la rama principal.

16. DORA Metrics:

Conjunto de métricas propuestas por el grupo de investigación DevOps Research and Assessment (DORA) que miden el desempeño de los equipos de desarrollo y operaciones.

17. Integración Continua (CI):

Práctica en la que los desarrolladores integran su código en un repositorio compartido varias veces al día para detectar errores de manera temprana.

18. Despliegue Continuo (CD):

Proceso de automatización que asegura que el código desarrollado y validado se despliegue en producción de manera confiable.

19. Artifact:

Archivo generado como resultado de un proceso de construcción de software, como un ejecutable, librería o archivo comprimido.

20. Microservicios:

Arquitectura de software en la que una aplicación se desarrolla como un conjunto de servicios pequeños e independientes que se comunican entre sí.

21. Contenedores:

Unidades portables de software que agrupan código y todas sus dependencias para que una aplicación se ejecute de forma confiable en cualquier entorno.

22. Kubernetes:

Plataforma de orquestación de contenedores que permite la automatización de despliegues, la gestión de aplicaciones y la escalabilidad en entornos distribuidos.

23. DevSecOps:

Evolución de DevOps que integra prácticas de seguridad en todas las fases del ciclo de vida del desarrollo de software.

24. Agile:

Metodología de desarrollo de software que prioriza la colaboración, la adaptabilidad y la entrega incremental de valor a los clientes.

25. Kanban:

Herramienta visual utilizada para gestionar flujos de trabajo y mejorar la eficiencia de los procesos, basada en principios Lean.

26. Scrum:

Marco de trabajo Agile que organiza equipos y tareas en iteraciones cortas, llamadas sprints, para entregar valor de manera continua.

27. Backlog:

Lista priorizada de tareas, características o requisitos pendientes que se deben abordar durante el desarrollo de un producto.

28. Testing Automatizado:

Proceso de ejecución de pruebas de software mediante scripts o herramientas que validan automáticamente su funcionamiento.

29. Code Review:

Proceso de inspección de código por parte de otros desarrolladores para garantizar la calidad, seguridad y eficiencia del software.

30. Continuous Monitoring (Monitoreo Continuo):

Práctica de supervisión constante de sistemas y aplicaciones para detectar y resolver problemas de rendimiento o seguridad en tiempo real.