UNIVERSIDAD MAYOR, REAL Y PONTIFICIA DE SAN FRANCISCO XAVIER DE CHUQUISACA

VICERRECTORADO

CENTRO DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

FACULTAD DE CIENCIAS Y TECNOLOGÍA

"HERRAMIENTAS ESCENCIALES DEVOPS PARA LA AUTOMATIZACIÓN DEL FLUJO DE TRABAJO EN PYMES DE DESARROLLO DE SOFTWARE DE BOLIVIA"

TRABAJO EN OPCIÓN A DIPLOMADO EN DEVELOPEMENT OPERATIONS "DEVOPS" V.1.

AUTOR: ROSANA MIRIAN CORZO GUTIERREZ

SUCRE-BOLIVIA

2024

CESIÓN DE DERECHOS

Al presentar este trabajo como requisito previo para la obtención del Diploma en Developement Operations "DEVOPS" V.1. de la Universidad Mayor, Real y Pontificia de San Francisco Xavier de Chuquisaca, autorizo al Centro de Estudios de Posgrado e Investigación o a la Biblioteca de la Universidad, para que se haga de este trabajo un documento disponible para su lectura según normas de la Universidad

También cedo a la Universidad Mayor, Real y Pontifica de San Francisco Xavier de Chuquisaca, los derechos de publicación de este trabajo o parte de él, manteniendo mis derechos de autor hasta un periodo de 30 meses posterior a su aprobación.

Rosana Mirian Corzo Gutierrez

……..

FIRMA:

DEDICATORIA

Dedico este trabajo a mis padres por el amor, dedicación y apoyo que me dieron en la vida y también a mis hijas que me dieron el soporte y la inspiración para culminar esta etapa.

AGRADECIMIENTOS

Agradezco primeramente a Dios.

Agradezco a mis padres por todo el apoyo y la ayuda, que han contribuido mucho en mi realización personal y profesional.

Agradezco a toda mi familia, aunque lejos, siempre me ha apoyado, ayudado y estuvo presente durante toda mi vida.

Agradezco a cada uno de los docentes por el seguimiento y orientación a lo largo de producción es este trabajo.

RESUMEN

Actualmente las Pymes dedicadas al desarrollo de software en Bolivia, así como cualquier empresa en el área de desarrollo buscan lógicamente reducir gastos, ofrecer mejores productos al cliente, por lo cual una buena opción es la implementación de DevOps, lo cuál implica el uso de herramientas de automatización, que trae muchos beneficios para la empresa: reducción de errores humanos, despliegues más rápidas, elasticidad, mejoras en la calidad, visibilidad sobre el estado del proyecto, monitoreo continuo, reducción de costos, seguridad, flexibilidad y adaptabilidad.

Las herramientas DevOps propuestas a raíz de la investigación abordada son sugeridas tomando en cuenta que se cuenta con presupuestos reducidos, para equipos de pocos actores, pensando en la escalabilidad a futuro.

Como resultado de la investigación se presenta una lista reducida de herramientas devops que se podrían utilizar en empresas pequeñas , propiamente Pymes de Bolivia.

INDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

1. Antecedentes y Justificación

1.1. Antecedentes

Según estudios de (Microsoft, 2022) su informe señala que el 86 % de las pymes afirmaron que la pandemia aceleró los procesos de transformación digital y el 90 % remarcó que esta se vio acompañada del uso de datos para la toma de decisiones, lo cual hace remarcar la importancia de la tecnología en el ámbito cotidiano, así como en el área empresarial.

Las empresas que se dedican al desarrollo de software según la publicación realizada en el 2018 (Adriana B. Foronda B., 2020) menciona que era un total de 2952 empresas, y que la tasa de crecimiento respecto a años anteriores hasta se había duplicado, sin embargo, también menciona que esa tasa se vio estancada ese 2018.

El estudio buscaba conocer el impacto que la pandemia tuvo en la transformación digital las PYMES de Bolivia, así como el rol que jugó la tecnología para responder a este contexto. Uno de los principales hallazgos está relacionado con la aceleración en la incorporación de tecnología y el uso de datos en las pymes bolivianas. En este sentido, más del 84% de las empresas invirtió en tecnología en el último año y el 90% indicó que toman decisiones basadas en datos o están considerando hacerlo.

En el panorama empresarial actual, la agilidad, la eficiencia y la calidad son elementos cruciales para el éxito de las pequeñas y medianas empresas (Pymes). La metodología DevOps ha surgido como una respuesta a la creciente necesidad de optimizar los procesos de desarrollo, entrega y operación de software, permitiendo a las organizaciones mejorar su capacidad para adaptarse rápidamente a los cambios del mercado y satisfacer las demandas de los clientes.

La historia de DevOps puede rastrearse hasta principios de la década de 2000, cuando las empresas de tecnología comenzaron a enfrentar una creciente presión para lanzar software de manera más rápida y confiable. La necesidad de implementar cambios de manera ágil y

mantener la estabilidad del sistema condujo al surgimiento de nuevas prácticas y herramientas que finalmente evolucionaron hacia lo que hoy conocemos como DevOps.

Uno de los hitos importantes en la historia de DevOps fue la publicación en 2009 del libro "The Phoenix Project" (Kim, 2018). Este libro ficticio presenta los desafíos comunes que enfrentan las organizaciones de TI y cómo pueden superarse mediante la implementación de prácticas DevOps.

Otro evento significativo fue el surgimiento de herramientas de automatización y orquestación, como Puppet, Chef y Ansible, que permitieron a los equipos automatizar tareas repetitivas y mantener la infraestructura de manera consistente.

A medida que DevOps ganaba popularidad, se convirtió en una filosofía adoptada por una amplia gama de empresas, desde startups hasta grandes corporaciones. Las empresas que implementaron con éxito prácticas DevOps informaron beneficios significativos, como lanzamientos más rápidos, ciclos de desarrollo más cortos, mayor calidad del software y una mejor colaboración entre equipos.

Hoy en día, DevOps continúa evolucionando con la incorporación de prácticas como DevSecOps (integración de seguridad en el ciclo de vida del desarrollo de software) y la adopción de tecnologías emergentes como contenedores y microservicios. En resumen, la historia de DevOps es una historia de adaptación y evolución en respuesta a las crecientes demandas de la industria del desarrollo de software.

1.2. Justificación

1.2.1. Justificación Económica

Algunas herramientas DevOps pueden tener costos asociados, ya sea en forma de licencias de software, suscripciones a servicios en la nube o tarifas de soporte. Por lo cual la evaluación del costo total de propiedad (TCO) de las herramientas seleccionadas es crucial para evitar sorpresas financieras y asegurarse de que se ajusten al presupuesto disponible.

En el informe público de (Dynatrace, 2023) donde se ha realizado un estudio del estado de la automatización de DevOps, refleja como resultado que en cuanto a la disminución general del gasto en IT se llega hasta a un 55% con el uso de herramientas de automatización DevOps.

1.2.2. Justificación Metodológica

Es importante resaltar cómo las herramientas de automatización DevOps pueden abordar las necesidades y desafíos específicos de la organización, así como los beneficios tangibles que pueden aportar en términos de eficiencia, calidad, flexibilidad y competitividad.

Otros datos relevantes que refleja el informe de (Dynatrace, 2023) es que uno de los beneficios para las compañías que implementan el uso de herramientas de automatización DevOps es la disminución de fallos es un 57%, y la mejora de la calidad de software en un 61%.

2.- Situación problémica

El crecimiento de las PYMES en los últimos años (Microsoft, 2022) y la necesidad del desarrollo y utilización de software lleva a que a su vez se necesite desarrollar software a mayor velocidad y optimizando los recursos con los que cuentan las PYMES.

El desarrollo del software aborda una variedad de desafíos inherentes a las metodologías tradicionales (Digital ai, 2023) lo que a menudo lleva a enfrentar los siguientes problemas:

Equipos fragmentados y procesos en silos

En los entornos de desarrollo tradicionales, los equipos operan de forma aislada, lo que genera comunicaciones fragmentadas y flujos de trabajo inconexos.

Ciclos de entrega lentos e ineficientes

Los enfoques de desarrollo tradicionales se caracterizan por ciclos de entrega largos y propensos a errores, lo que resulta en retrasos en el software, releases y partes interesadas frustradas.

Falta de visibilidad y rendición de cuentas

En las configuraciones de desarrollo tradicionales, la visibilidad del proceso de entrega de software suele ser limitada, lo que dificulta realizar un seguimiento del progreso, identificar cuellos de botella y responsabilizar a los equipos de los resultados.

Procesos manuales y propensos a errores

Las intervenciones manuales y los traspasos entre desarrollo, control de calidad y operaciones introducen oportunidades de errores, inconsistencias y retrasos en el proceso de entrega de software.

Bucles de retroalimentación limitados y mejora continua

Las metodologías de desarrollo tradicionales carecen de mecanismos para recopilar comentarios oportunos, lo que obstaculiza los esfuerzos de mejora continua e inhibe la innovación.

En resumen, la falta de automatización del desarrollo de software para PyMES puede llevar a desafíos que afectan la eficiencia, la calidad, la velocidad de entrega y la competitividad en el mercado. Es importante que las PYMES consideren la implementación de herramientas y prácticas de automatización para superar estas problemáticas y maximizar su éxito en el desarrollo de software.

3- Formulación del problema de investigación

¿Qué herramientas de DevOps serían esenciales para la automatización del flujo de trabajo en el desarrollo de software para PYMES en Bolivia?

4.- Objetivo General

Proponer un conjunto de herramientas DevOps para la automatización del flujo de trabajo para el desarrollo de un software para PYMES en Bolivia.

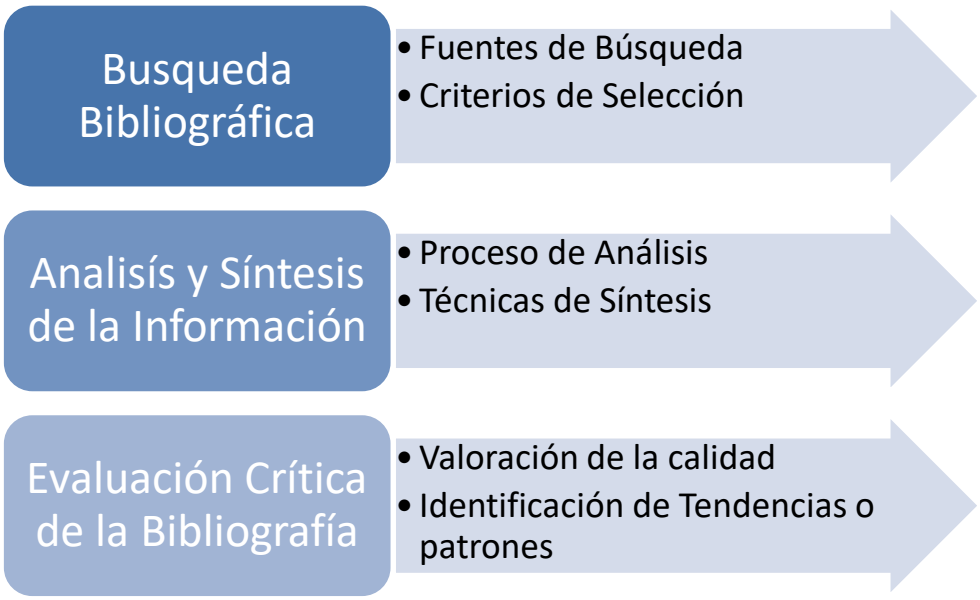
5.- Objetivos específicos

- Documentar las características y funcionalidades de las principales herramientas DevOps
- Investigar Herramientas DevOps que se adapten a las Pymes
- Comparar las funcionalidades y la capacidad de adecuar las principales herramientas DevOps dentro del flujo de trabajo en el desarrollo de Software .

6.- Diseño Metodológico

En base al conocimiento de (Gil, 2002) sobre la clasificación de los objetivos de cada investigación, esta monografía está basada en los procedimientos técnicos de investigación bibliográfica y materiales ya elaborados en libros y artículos científicos.

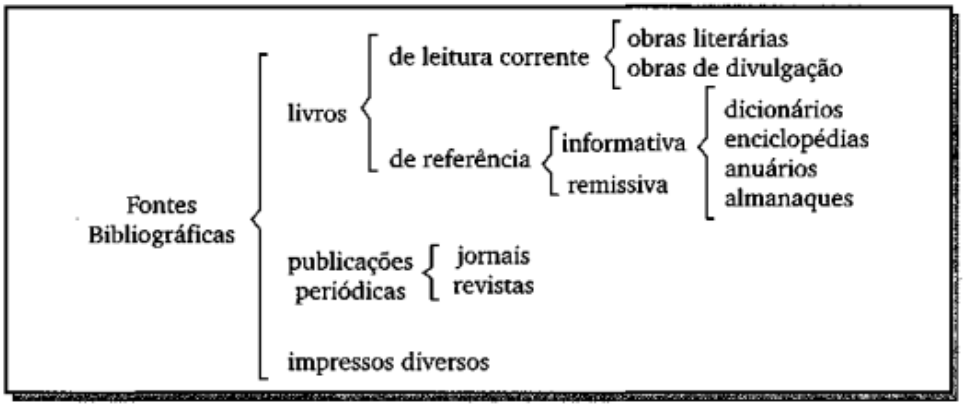
Figura 1 - *Proceso de la investigación de herramientas devops en Bolivia*



Nota: Etapas de la investigación de fuentes bibliográficas. Tomada de elaboración propia

Las fuentes bibliográficas son un gran número y pueden ser clasificadas de la siguiente manera:

Figura 2- *Fuentes Bibliográficas*



Nota: Clasificacion de Fuentes Bibliográficas. Tomada de (Gil, 2002)

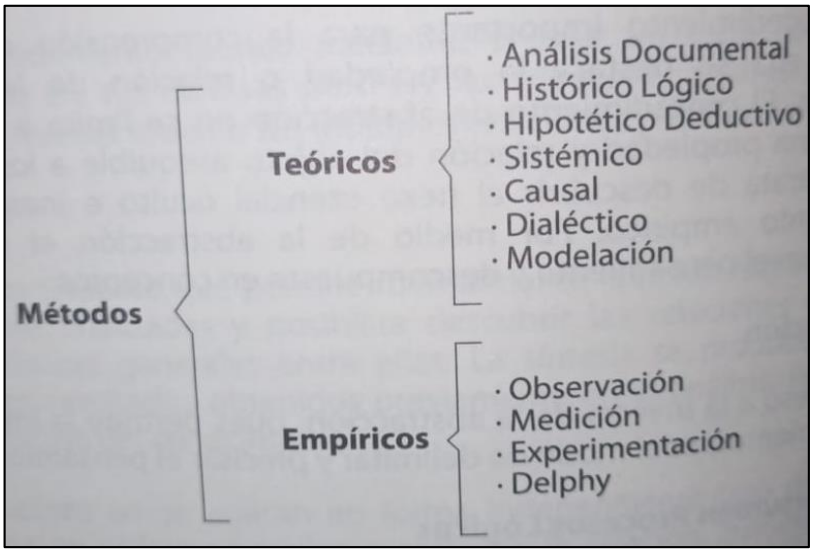
6.1- Tipo de investigación

Dada la naturaleza de esta investigación la que más se adecúa al caso es la Investigación descriptiva, ya que este se centra en describir características, fenómenos o situaciones tal como son. Este tipo de investigación es útil para establecer una comprensión inicial del tema de estudio, identificar patrones o tendencias, y proporcionar una visión general de la situación.
En este caso de investigación, recabaremos la mayor cantidad información referente a devops, herramientas de automatización de devops, pymes en Bolivia, empresas de desarrollo de software de Bolivia, devops en Bolivia, etc. Por todo lo mencionada con anterioridad es necesario tomar en cuenta leyes, noticias, boletines informativos, revistas, libros, etc.

6.2.-Métodos

En este acápite la selección de los métodos a usar para el desarrollo de esta monografía se basa en el siguiente esquema presentado por (Martínez, 2013)

Figura 3- *Métodos de Investigación*



Nota: Tipos de Métodos. Tomada de (Martínez, 2013)

6.2.1 Métodos Análisis Teórico

Atendiendo la finalidad investigativa de este trabajo desarrollado a continuación se citarán solo los métodos seleccionados.

Investigación Bibliográfica: Se recopila y analiza la información relevante de fuentes bibliográficas como libros, revistas académicas, artículos, informes técnicos, y otras publicaciones relacionadas con las Pymes, las herramientas de automatización DevOps. Este método es fundamental para fundamentar teóricamente la monografía y contextualizar el tema dentro del campo de estudio.

Investigación Documental: Se recopila y analizar documentos relacionados con el tema de estudio, como leyes, reglamentos, normativas, políticas, documentos gubernamentales, entre otros con el fin de obtener una comprensión integral del tema mediante la exploración de diversas fuentes documentales, que pueden proporcionar perspectivas únicas o datos específicos no disponibles en la literatura académica convencional.

6.2.1.1 Método Análisis –Síntesis

Este método se refiere a dos procesos intelectuales inversos que operan en unidad: el análisis y la síntesis.

El análisis es un procedimiento lógico que posibilita descomponer mentalmente un todo en sus partes y cualidades, en sus múltiples relaciones, propiedades y componentes. Permite estudiar el comportamiento de cada parte.

La síntesis es la operación inversa, que establece mentalmente la unión o combinación de las partes previamente analizadas y posibilita descubrir relaciones y características generales entre los elementos de la realidad.

6.2.1.2. Método Histórico – Lógico

A continuación, se ofrece una descripción general de lo que significa cada componente en este contexto:

Antecedentes: se refiere a comprender los orígenes, la evolución y el contexto del movimiento DevOps. Comprender cómo y por qué surgió DevOps es fundamental para aplicar sus prácticas de manera significativa. Esto implica estudiar los problemas que llevaron al surgimiento de DevOps, como los silos organizacionales, los procesos manuales que consumen mucho tiempo y la falta de colaboración entre los equipos de desarrollo y operaciones.

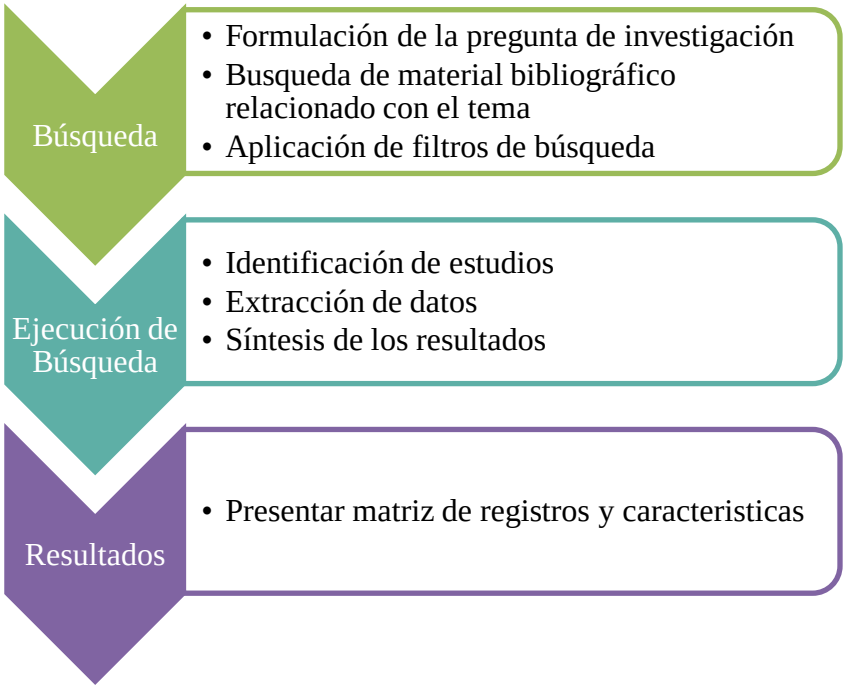
Lógico: este aspecto implica la comprensión lógica y conceptual de las prácticas y principios de DevOps. Esto incluye comprender los pilares de DevOps como la automatización, la colaboración y la retroalimentación rápida, así como prácticas específicas como la integración continua, la entrega y el monitoreo continuo.

Al combinar estos dos elementos, el método histórico-lógico DevOps busca no sólo implementar prácticas DevOps, sino también comprender el por qué detrás de ellas. Esto permite una implementación más eficaz y adaptable, ya que las empresas pueden comprender cómo se pueden ajustar las prácticas de DevOps, la selección de herramientas de automatización para satisfacer sus necesidades específicas basándose en conocimientos históricos y lógicos.

6.2.1.3. Método Sistémico

Un método sistémico para seleccionar herramientas DevOps implica tener un enfoque integral que considere no solo las características técnicas de las herramientas, sino también factores como la integración, la cultura organizacional, los requisitos comerciales y la compatibilidad. Por lo cual para la selección de herramientas será necesario un mapeo integral del contexto.

Figura 4 *- Proceso de Mapeo Integral*



Nota: Etapas del proceso de Mapeo Integral. Tomada de Elaboración Propia

6.3.- Técnicas

Revisión de Literatura: revisar investigaciones previas, libros, artículos académicos, informes técnicos y otros recursos relevantes sobre herramientas de automatización DevOps.

Análisis Documental: analizar documentos relevantes como manuales de usuario, documentación técnica, informes de casos de uso y estudios de investigación anteriores sobre herramientas de automatización DevOps. Este enfoque puede proporcionar información detallada sobre las características y funcionalidades de las herramientas.

6.4.- Procedimientos e instrumentos de investigación

Los instrumentos que se utilizaran son los siguientes:

- **Cuadro de registro para el análisis documental:** principalmente para poder organizar y catalogar la información relevante encontrada en documentos.
- **Lista de Cotejo para comparar las herramientas:** Es importante la organización, claridad; para ello se debe llevar un registro y seguimiento y así poder tomar decisiones con mayor facilidad y así poder satisfacer las necesidades específicas del usuario o del proyecto.

CAPITULO I

MARCO TEORICO Y CONTEXTUAL

1.1 Marco Conceptual

1.1.1 ¿Qué es DEVOPS?

Según (Kim, 2018) se trata de "una filosofía cultural y profesional que promueve la colaboración entre el desarrollo (Dev) y las operaciones (Ops) en toda la cadena de suministro de software, desde la planificación y el desarrollo hasta la entrega y la implementación, y tiene como objetivo acortar el ciclo de vida del desarrollo de sistemas y proporcionar entregas de software de alta calidad"

Anteriormente (Len Bass, 2015) llegó a concluir que "DevOps es un conjunto de prácticas para reducir el tiempo entre el compromiso de un cambio en un sistema y la implementación de ese cambio en producción, con una calidad aceptable"

La cultura DevOps según (cloudbees, 2020) se basa en un conjunto de principios que una organización aspira inicialmente y a los que finalmente se adhiere. Las organizaciones que han adoptado esta cultura valoran la colaboración, la experimentación y el aprendizaje. En una cultura DevOps, todos los participantes en el ciclo de vida de entrega de software (no solo el desarrollo y las operaciones) se alinean en torno a un objetivo compartido: la entrega rápida de software estable y de alta calidad desde el concepto hasta el cliente. Dado que DevOps es algo cultural, técnicamente no requiere automatización. Sin embargo, la automatización del desarrollo, las pruebas y la implementación de software a través de la entrega continua es ampliamente reconocido como un facilitador clave de DevOps. La automatización permite a las organizaciones entregar software más rápidamente y a su vez se garantiza que las operaciones puedan tener confianza en lo que se está implementando y que los clientes obtengan la calidad, la seguridad y la estabilidad ellos requieren.

1.1.2 Etapas de las aplicaciones

DevOps (Azure, s.f.) influye en el ciclo de vida de las aplicaciones a lo largo de las fases de planificación, desarrollo, entrega y uso.

Planificación

En la fase de planificación, los equipos de DevOps conciben, definen y describen las características y la funcionalidad de las aplicaciones y los sistemas que van a crear. Realizan un seguimiento del progreso tanto de forma general como de forma pormenorizada, desde tareas de un único producto hasta tareas que abarcan carteras de numerosos productos. La creación de registros de trabajo pendiente, el seguimiento de los errores, la administración del desarrollo de software ágil con Scrum, el uso de paneles Kanban y la visualización del progreso son algunas de las formas en las que los equipos de DevOps realizan la planificación con agilidad y visibilidad.

Desarrollo

La fase de desarrollo incluye todos los aspectos de la programación (escritura, pruebas, revisión e integración del código por parte de los miembros del equipo) y la compilación de ese código en artefactos de compilación que se pueden implementar en varios entornos. Los equipos de DevOps buscan innovar con rapidez sin sacrificar la calidad, la estabilidad ni la productividad. Para ello, utilizan herramientas muy productivas, automatizan los pasos cotidianos y manuales, e iteran el código en pequeños incrementos mediante pruebas automáticas e integración continua.

Entrega

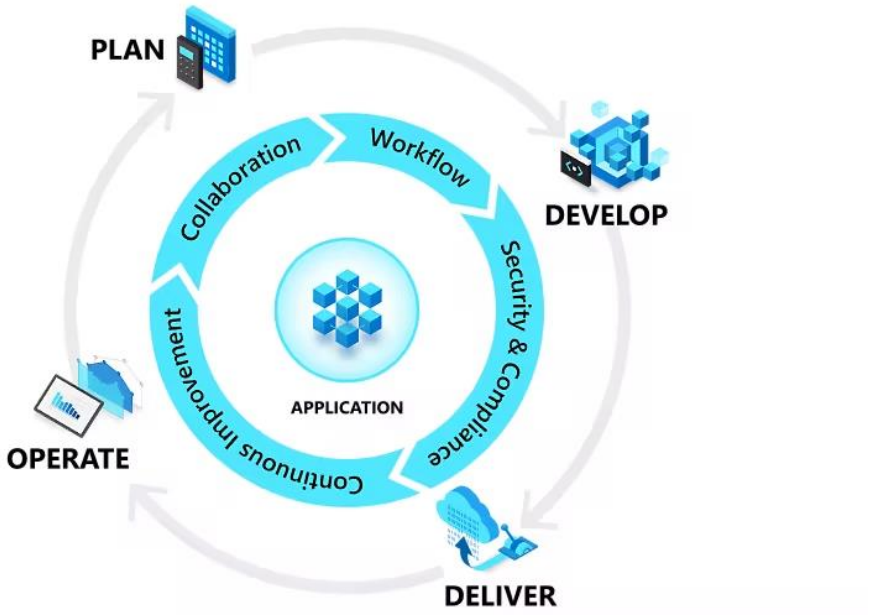
La entrega es el proceso de implementar aplicaciones en entornos de producción de un modo constante y confiable. La fase de entrega incluye también la implementación y la configuración de la infraestructura básica totalmente gobernada que constituye esos entornos.

En la fase de entrega, los equipos definen un proceso de administración de versiones con fases de aprobación manual claras. También establecen puertas automáticas que mueven las aplicaciones de una fase a otra hasta que están disponibles para los clientes. La automatización de estos procesos hace que estén controlados y sean escalables y repetibles. De este modo, los equipos que practican DevOps pueden realizar entregas con facilidad, confianza y tranquilidad.

Uso

La fase de uso implica mantener y supervisar las aplicaciones, así como solucionar los posibles problemas, en los entornos de producción. Al adoptar prácticas de DevOps, los equipos trabajan para asegurar la confiabilidad, la alta disponibilidad y el objetivo de ningún tiempo de inactividad del sistema, al tiempo que refuerzan la seguridad y el gobierno. Los equipos de DevOps buscan identificar los problemas antes de que afecten a la experiencia del cliente y mitigarlos rápidamente a medida que surgen. El mantenimiento de esta vigilancia requiere una telemetría muy completa, alertas que permitan tomar medidas y visibilidad total de las aplicaciones y del sistema subyacente.

Figura 5 - Ciclo de Vida de las aplicaciones

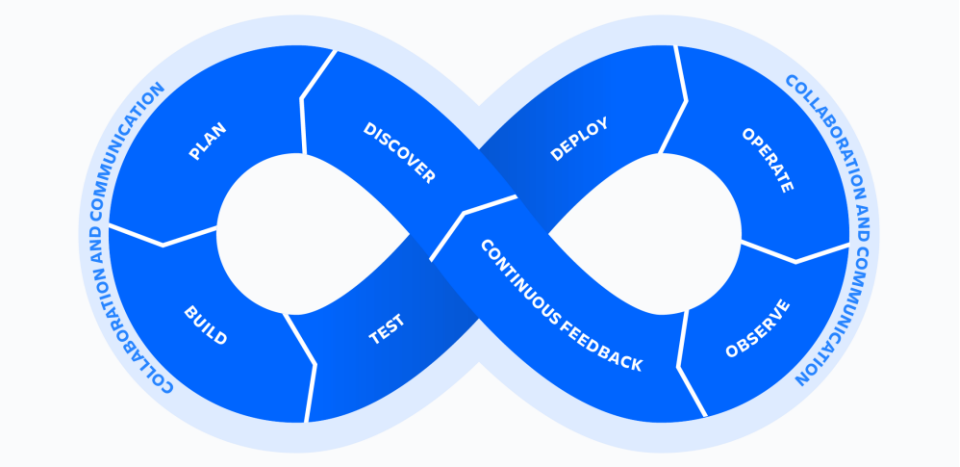


Nota: Ciclo de vida de las aplicaciones. Tomado de (Azure, s.f.)

1.1.3 Ciclo de vida DEVOPS

DevOps, una metodología que promueve la colaboración entre los equipos de desarrollo (Dev) y operaciones (Ops) para automatizar los procesos de desarrollo de software, tiene varias etapas bien definidas. Aquí se presenta un desglose de las etapas comunes:

Figura 6 - *El ciclo de vida de DevOps.*



Nota: Ocho fases de vida DevOps. Tomada de (Atlassian, s.f.)

Cada ciclo de DevOps consta de 8 fases:

Plan

Al igual que en otras metodologías ágiles, se comienza elaborando un plan para alcanzar el objetivo marcado. Todas las personas involucradas trabajarán durante la iteración para lograr el objetivo. Deberá estar suficientemente claro para abordarlo sin dudas.

Code

En esta fase se construirá el software y se definen las pruebas que deberá pasar para dar por buena la solución. En el lado de operaciones se construyen todos los automatismos necesarios para la distribución y mantenimiento del software.

Build

Una vez construida la solución, se crean los nuevos artefactos que componen el software, incluyendo las nuevas funcionalidades. Este proceso es crítico, ya que un error en la fusión del software puede provocar que deje de funcionar.

Test

Aquí es el momento de ejecutar todas las pruebas que verifiquen el correcto funcionamiento de todas las características. En esta fase es importante contar con unas buenas pruebas para verificar que todas las funcionalidades que ya existían siguen operativas. En el caso en que alguna prueba no sea satisfactoria, se debe de realizar la corrección volviendo a la fase de desarrollo.

Release

Una vez la aplicación ha pasado todas las pruebas, se puede crear una versión del software. Esta versión se marca para indicar que ha pasado todas las validaciones previas y se podría poner a disposición de los usuarios en un futuro.

Deploy

En DevOps el ciclo no acaba cuando existe una versión lista para los usuarios. Se continua realizando la integración del nuevo software en cada uno de los entornos que sean necesarios. Cada entorno tiene un propósito y en él se realizan un tipo de pruebas. Será en este momento cuando se verifiquen que las automatizaciones que se realizaron en la fase de desarrollo funcionan correctamente, permitiendo la instalación del nuevo software en los entornos. El último despliegue será en el entorno de producción, poniendo el software a disposición de los usuarios.

Operate

El sistema necesita de actualizaciones en función de su uso. Esta fase del personal de operaciones mide los recursos disponibles y los ajusta en función de las nuevas necesidades del software. Se elaboran nuevas métricas con las que vigilar el sistema.

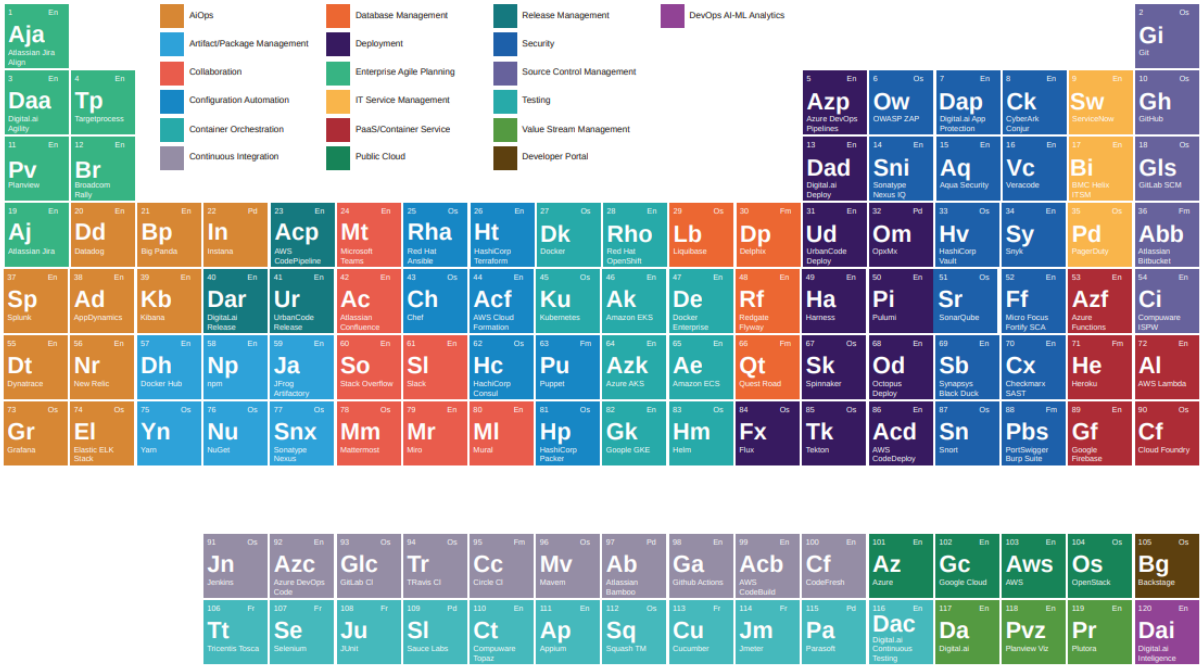
Monitor

Aquí se establecen qué parámetros de la aplicación se va a vigilar. Toda la información recogida a lo largo de un periodo de tiempo permite realizar ajustes que se reflejarán en la siguiente fase de planificación.

1.1.4 Herramientas DEVSECOPS

La tabla periódica de herramientas DevSecOps es el recurso de referencia de la industria para identificar las mejores herramientas en todo el ciclo de vida de entrega de software.

Figura 7 *- Tabla Periódica de Herramientas DevSecOps*



Nota: Tabla de herramientas DevSecOps. Tomada de *(Digital.ai, 2023)*

1.2 Marco Teórico

1.2.1. Relevancia de las PyMEs en Bolivia

Las Pequeñas y Medianas Empresas (PYMES) desempeñan un papel fundamental en la economía de Bolivia. Representan una gran parte del tejido empresarial y generan empleo en diferentes sectores. La clasificación de las empresas en Bolivia se basan en la cantidad de trabajadores que posea la misma.

Tabla 1-*Clasificación de empresas en Bolivia*

Clasificación	Número de trabajadores	Activos productivos	Ventas anuales	Exportaciones Anuales
Microempresa	Hasta 9	Hasta USD 52 500	Hasta USD 210 000	Hasta USD 26 250
Pequeña empresa	Entre 10 y 19	Entre USD 52 501 y USD 525 000	Entre USD 210 001 y USD 1 050 000	Entre USD 26 251 y USD 262 500
Mediana empresa	Entre 20 y 49	Entre USD 525 001 y USD 2 100 000	Entre USD 1 050 001 y USD 4 200 000	Entre USD 262 501 y USD 2 625 000
Empresa grande	Desde 50	Mayor a USD 2 100 001	Mayor a USD 4 200 001	Mayor a USD 2 625 001

Nota: Clasificación de empresas en Bolivia de acuerdo al número de trabajadores. Tomada de

(Banco de desarrollo de América Latina y el Caribe, 2023)

En el contexto del desarrollo de software, las PYMES tienen desafíos específicos, como recursos limitados y estructuras organizativas más simplificadas. Es por eso que la implementación de herramientas DevOps y la automatización del flujo de trabajo se vuelven aún más relevantes. Estas herramientas permiten a las PYMES optimizar sus procesos de desarrollo, ahorrar costos, mejorar la colaboración y competir de manera más efectiva en el mercado. Al aprovechar las ventajas de la automatización, las PYMES pueden acelerar la entrega de productos y servicios, adaptarse rápidamente a los cambios del entorno y brindar soluciones de software de calidad a sus clientes.

1.2.2 Beneficios principales de adoptar DevOps

Reducción de conflictos entre equipos

DevOps fomenta una cultura de colaboración y responsabilidad compartida, lo que ayuda a reducir los conflictos entre los equipos de desarrollo y operaciones que a menudo se producen en entornos tradicionales.

Las metodologías Devops son la clave para evitar echarse la culpa unos a otros cuando surgen problemas. En lugar de eso, se trabaja en equipo para identificar el error. La colaboración y transparencia entre equipos permite implementar pruebas automatizadas para detectar y corregir errores de manera rápida. Así, podemos solucionar los problemas y hacer las correcciones

necesarias en el entorno de producción sin tener que esperar largos procesos manuales. Todo fluye mucho mejor con metodologías DevOps.

Alineación de los objetivos del negocio y TI

Una de las cosas geniales que trae consigo metodologías DevOps es que todos los equipos, tanto de desarrollo como de operaciones, trabajan juntos hacia un objetivo común y alineado con el negocio. Antes, solían estar separados y eso a veces causaba problemas. La clave está en la comunicación constante y en compartir responsabilidades. Es como si todos fuéramos un gran equipo que busca entregar lo mejor al cliente y al negocio de manera rápida y efectiva. Así, nos aseguramos de que las decisiones tecnológicas que tomamos estén en línea con lo que realmente necesita el cliente y lo que planea el negocio para el futuro.

Resolución rápida de problemas

Una de las metodologías DevOps que implementa de base esta cultura es la automatización de pruebas y despliegues. Estas tareas intermedias automatizadas ayudan a detectar errores tempranamente y a resolverlos antes de que lleguen a producción. Y es una de las primeras cosas que se tienen que implementar al adoptar la cultura DevOps.

Facilita la adopción de nuevas tecnologías

Imagina que cada vez que aparece una nueva herramienta o tecnología emergente que promete mejorar nuestro trabajo, es como si fuera un emocionante juego nuevo que todos quieren probar. Con la automatización, es como si tuviéramos un equipo de organizadores expertos que rápidamente preparan el terreno para que podamos probar y utilizar la nueva tecnología sin demora. Y cuando hablamos de gestión de infraestructura como código, es como tener una lista de instrucciones clara y precisa para el juego. Esto nos asegura que todos sigamos las mismas reglas y no nos quedemos atrás. Con esta gestión eficiente de las metodologías DevOps, la adopción de tecnologías emergentes se vuelve ágil y sin interrupciones para todo el equipo.

Fomento de la innovación abierta

Imagina que el mundo de la tecnología es como una gran fiesta llena de personas creativas e inteligentes, cada una con ideas brillantes y soluciones innovadoras. Con DevOps, las empresas

pueden abrir las puertas de esa fiesta y permitir que se unan proveedores externos y comunidades de código abierto. Esta colaboración nos abre la puerta a la innovación y nos permite incorporar nuevas ideas de una manera ágil y efectiva. Así que, gracias a las metodologías DevOps y su enfoque en la innovación abierta, estamos preparados para recibir ideas frescas y soluciones creativas que nos mantienen a la vanguardia de la tecnología.

Figura 8 *- Los Beneficios de DevOps*

Nota: Los Beneficios de DevOps. Tomada de (Dynatrace, 2021)

1.2.3 Importancia de la automatización en el desarrollo de software

La automatización es un pilar fundamental en el desarrollo de software, ya que permite agilizar y optimizar los procesos, reducir errores humanos, mejorar la calidad del producto final y aumentar la eficiencia y productividad de los equipos de trabajo. La automatización de tareas repetitivas y tediosas, como la compilación, pruebas y despliegue, permite a los desarrolladores centrarse en actividades de mayor valor añadido, como la mejora de funcionalidades y la innovación. Además, la automatización proporciona una mayor consistencia y trazabilidad en el flujo de trabajo, facilitando la colaboración entre los diferentes miembros del equipo y mejorando la escalabilidad del proceso de desarrollo.

1.2.4 Herramientas Devops

Las herramientas de automatización DevOps son fundamentales para optimizar los procesos de desarrollo, pruebas, despliegue y operación de software en un entorno ágil. Estas herramientas abarcan diferentes áreas del ciclo de vida del desarrollo de software y facilitan la integración y colaboración entre equipos de desarrollo y operaciones.

La automatización es la piedra angular de la estrategia DevOps de toda empresa, así lo indica (Dynatrace, 2021). Es decir, la automatización reduce el trabajo, le ayuda a acelerar sus canales de entrega en todo el SDLC y le permite escalar su práctica de DevOps.

Tradicionalmente, procesos como las pruebas, la monitorización, el descubrimiento de errores y la corrección de los mismos comprendían un poco de automatización y mucha intervención manual. Esto funcionaba cuando los equipos pequeños trabajaban en aplicaciones monolíticas. Pero con los modernos microservicios basados en aplicaciones y con la transformación digital ejerciendo aún más presión sobre las TI, la automatización es crucial para aumentar la velocidad y la calidad impulsando procesos coherentes en cada etapa del ciclo de vida de DevOps

Desarrollo de Código:

Es importante tener el control del código fuente del software del código de un desarrollador. Las herramientas de control de código fuente ayudan a almacenar el código en diferentes cadenas para poder ver los cambios y compartirlos a fin de facilitar la colaboración. En lugar de esperar la decisión del comité de evaluación de cambios para implementarlos en producción, se puede mejorar la calidad y el rendimiento del código con revisiones por pares realizadas mediante solicitudes de incorporación de cambios.

- Git: Para el control de versiones del código fuente.
- GitHub/GitLab/Bitbucket: Plataformas de alojamiento de repositorios Git con características adicionales como seguimiento de problemas y colaboración.
- IntelliJ IDEA: Un entorno de desarrollo integrado (IDE) que ofrece características avanzadas para la escritura y depuración de código.

Integración Continua:

- Jenkins: Una herramienta de automatización de código abierto para la integración continua y la implementación continua.
- Travis CI: Un servicio de integración continua en la nube que se integra bien con repositorios de Git.
- CircleCI: Otra plataforma de integración continua que permite la ejecución rápida de pruebas y la implementación de código.

Despliegue Continuo:

- Docker: Para la creación y ejecución de contenedores, que facilita el despliegue consistente de aplicaciones.
- Kubernetes: Una plataforma de orquestación de contenedores que automatiza el despliegue, la escalabilidad y la gestión de aplicaciones en contenedores.
- Ansible: Una herramienta de automatización de infraestructura que permite la implementación y configuración consistentes de sistemas.

Monitoreo y Retroalimentación:

- Prometheus: Un sistema de monitoreo y alerta de código abierto diseñado para la recopilación y visualización de métricas.
- Grafana: Una plataforma de análisis y visualización de datos que se puede utilizar con Prometheus para crear paneles de monitoreo personalizados.
- ELK Stack (Elasticsearch, Logstash, Kibana): Una combinación de herramientas que se utiliza para la recopilación, análisis y visualización de registros y métricas.

Automatización de Infraestructura:

- Terraform: Una herramienta de infraestructura como código que permite definir y crear la infraestructura de manera declarativa.
- Chef: Un sistema de automatización de configuración que permite definir la configuración de sistemas de manera programática.
- Puppet: Otra herramienta de automatización de configuración que se utiliza para administrar la configuración de sistemas de manera consistente.

Herramientas de Automatización de Pruebas

- **Selenium:** es un conjunto de herramientas de código abierto utilizado para automatizar pruebas en aplicaciones web en diferentes navegadores y plataformas.
- **Appium: e**s un marco de automatización de pruebas de código abierto que se utiliza para probar aplicaciones móviles nativas, híbridas y web en plataformas Android e iOS.

- **Cypress:** es un marco de pruebas de código abierto diseñado específicamente para pruebas de extremo a extremo en aplicaciones web modernas.
- **Postman:** es una herramienta de desarrollo de API que se utiliza para probar, documentar y colaborar en el desarrollo de API.
- **Junit:** es una herramienta de código abierto utilizada para realizar pruebas unitarias en aplicaciones Java. Proporciona un entorno de prueba estructurado y fácil de usar que permite a los desarrolladores escribir y ejecutar pruebas de manera eficiente.

1.2.6 Estado del Arte

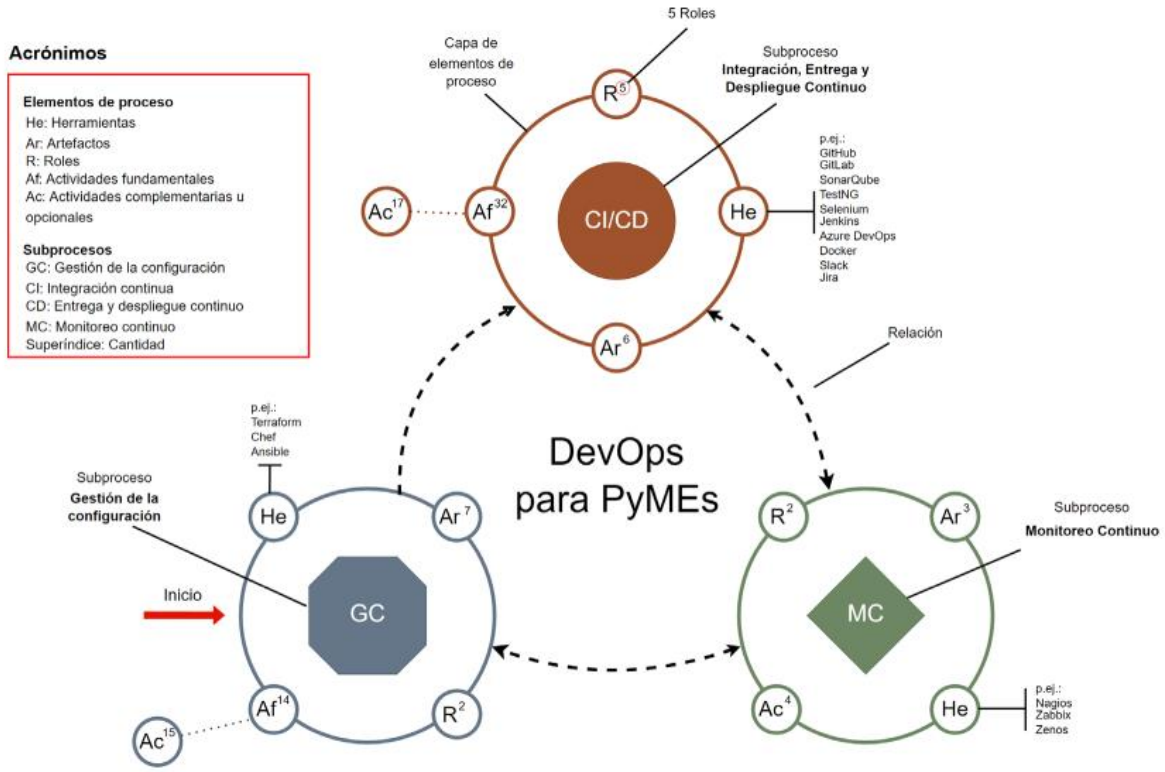
El termino DevOps en Bolivia recién está tomando importancia (Blanco, 2021). También cabe resaltar que de acuerdo a los lineamientos y estándares para el desarrollo de software de alta calidad para las empresas del sector público se va introduciendo términos de DevOps y seguridad.

Un aporte importante de (Blanco, 2021) es la realización de una encuesta que muestra que, en Bolivia, por lo menos las empresas públicas, en el área de desarrollo no cuentan con un rol de Devops.

Mientras en un contexto mundial (Transposit, 2023) las empresas están ampliando sus recursos tecnológicos e invirtiendo en soluciones de tecnología moderna los próximos 12 meses. En la actualidad, la era de la eficiencia, los encuestados también coinciden en general en que la IA y la automatización con humanos en el circuito son componentes críticos para la gestión de incidentes como una forma de mejorar la precisión y la calidad de los datos capturado durante incidentes, responder y resolver incidentes más rápido aprovechando la información basada en datos, y automatizar tareas o procesos repetitivos.

Con el propósito de colaborar con la adopción de DevOps en las Pymes de Software (Certuche S. C., 2022) han elaborado un proceso que permitirá guiar a las empresas desarrolladoras de software, conteniendo este tres subprocesos, roles y herramientas.

Figura 9 - *Procesos DevOps para Pymes*



Nota: Proceso para apoyar la implementación de DevOps. Tomada de (Certuche S. C., 2022)

1.2 Marco Contextual

El marco contextual para la selección de herramientas DevOps en Bolivia puede ser influenciado por varios factores específicos del país y su industria tecnológica; por lo cual se debe considerar los siguientes puntos:

Infraestructura tecnológica y acceso a herramientas

Es importante evaluar la disponibilidad y el acceso a herramientas de tecnología de la información en Bolivia. Esto incluye la infraestructura de red, la conectividad a internet y la disponibilidad de herramientas en el mercado local.

Cultura empresarial y adopción tecnológica

La cultura empresarial en Bolivia puede variar en términos de la adopción de tecnologías modernas como DevOps. Es importante evaluar la disposición de las organizaciones para

adoptar prácticas DevOps y la capacidad de su fuerza laboral para adaptarse a nuevas herramientas y procesos.

Presupuesto y costo de las herramientas

El presupuesto disponible puede limitar las opciones de herramientas disponibles. Es fundamental evaluar el costo de adquisición y mantenimiento de las herramientas DevOps en relación con el presupuesto disponible para la organización.

Requisitos específicos del proyecto y del negocio

Cada proyecto y empresa tiene requisitos y necesidades específicas. Es esencial identificar los requisitos funcionales y no funcionales del proyecto, así como las necesidades del negocio, para seleccionar las herramientas DevOps más adecuadas.

Soporte y comunidad

Considerar el nivel de soporte técnico disponible y la comunidad de usuarios para las herramientas DevOps seleccionadas. Contar con un sólido sistema de soporte puede ser crucial para resolver problemas y optimizar el uso de las herramientas.

Regulaciones y cumplimiento

Evaluar las regulaciones y los requisitos de cumplimiento relevantes en Bolivia que puedan afectar la selección y el uso de herramientas DevOps, especialmente en sectores altamente regulados como la banca, la salud o el gobierno.

Debido a que la implementación de DevOps es muy nueva en Bolivia (Blanco, 2021), aún están en proceso de aprobación por la Agencia de Gobierno Electrónico y Tecnologías de Información y Comunicación (AGETIC) y el Consejo para las Tecnologías de Información y Comunicación del Estado Plurinacional de Bolivia (CTIC-EPB), luego de su aprobación pasara a una segunda evaluación por las entidades públicas para finalmente ser publicado y todas las entidades públicas deberán cumplir con dichos lineamientos.

Integración y compatibilidad

Asegurarse de que las herramientas seleccionadas sean compatibles e integrables con el entorno tecnológico existente de la organización, incluyendo sistemas de gestión de bases de datos, plataformas de nube, herramientas de monitoreo, entre otros.

CAPITULO II

DIAGNOSTICO

2.1. Introducción

2.1.1. Procesamiento y Análisis de Datos

Primeramente, se ha realizado una búsqueda exhaustiva de información referente al tema, posteriormente se ha seleccionado solo bibliografía relevante y de fuentes confiables.

2.1.2 Tabulación y Codificación de datos

Para seleccionar las herramienta más adecuada para pymes, se realiza una investigación y compara varias opciones disponibles en el mercado, en la sección de anexos muestra una tabla referente a la bibliogrfía consultada para el desarrollo de esta investigación y además se detalla en varias tablas las particularidades de las herramienta DevOps.

La selección de herramientas DevOps adecuadas es crucial para el éxito de cualquier implementación de esta metodología. Las herramientas correctas facilitan la automatización de los procesos, mejoran la colaboración entre los equipos y optimizan el monitoreo y la gestión de la infraestructura. Al elegir las herramientas adecuadas, las organizaciones pueden obtener beneficios como una mayor eficiencia, una reducción de los tiempos de entrega y una mejora en la calidad del software. Por lo tanto, es esencial realizar una evaluación cuidadosa de las diferentes opciones disponibles y seleccionar las herramientas que mejor se adapten a las necesidades y objetivos específicos del proyecto.

- Se determinaron las ventajas y desventajas de las diferentes herramientas DevOps evaluadas.
- Se identificaron las características clave de cada herramienta.
- Se proporciona una visión clara y detallada de las herramientas, con el propósito de facilitar la elección de la herramienta más adecuada para cada escenario y necesidad específica.

2.1.3 Análisis y discusión de resultados

Cada una de las etapas de Devops puede ser automatizada con la ayuda de herramientas, algunas herramientas pueden ser usadas durante todo el ciclo DevOps, para ello se ha realizado una búsqueda de información de las más utilizadas. El objetivo final del uso de las herramientas es ayudar a garantizar procesos más rápidos, fiables y de mayor calidad.

Las empresas que adoptan devops demuestran ser más ágiles y capaces de adaptarse a un mercado constantes cambio he incertidumbre.

En la Tabla 12 se muestra una tabla comparativa de las herramientas devops con licencia Open Source, en base a esa tabla y al resultado de la puntuación se puede concluir que en PyMes, la selección de herramientas DevOps debe enfocarse en la simplicidad, facilidad de uso y costos efectivos.

Gestión de Código Fuente

- Git :posee un sistema de control de versiones distribuido, amplia adopción, robusto, permite colaboración eficiente.
- GitLab: Plataforma de DevOps completa que incluye gestión de código fuente, CI/CD y más. Integración CI/CD integrada, excelente para equipos pequeños y medianos.

CI/CD (Integración Continua y Entrega Continua)

- Jenkins : tien un servidor de automatización de código abierto, extensibilidad a través de plugins, gran comunidad.
- GitLab CI: Herramienta de CI/CD integrada en GitLab. Integrado con la gestión de código fuente de GitLab, fácil de usar.
- Travis CI : Herramienta de CI/CD basada en la nube. De fácil integración con GitHub, configuración simple.

Automatización y Configuración

- Ansible: es una herramienta de automatización de configuración y aprovisionamiento. Sin agentes, fácil de aprender, usa YAML para definir tareas.

- Terraform: es una herramienta de infraestructura como código (IaC). Cuenta con soporte para múltiples proveedores de nube, reutilización de configuraciones.
- Puppetipción: Herramienta de gestión de configuración. Posee una gran comunidad, biblioteca extensa de módulos.

Contenerización

- Docker: es una plataforma para desarrollar, enviar y ejecutar aplicaciones en contenedores. Tiene una amplia adopción, fácil de usar, portabilidad de aplicaciones.

Orquestación de Contenedores

- Kubernetes: de alta escalabilidad, gestión automatizada de aplicaciones en contenedores.

Monitoreo y Logging

- Prometheus: alta capacidad de recolección de métricas, integración con Grafana.
- Grafana: es una plataforma de analítica y visualización de métricas. Es ideal para integración con múltiples fuentes de datos, paneles personalizables.

Las herramientas listadas son ideales para pequeñas empresas que buscan adoptar prácticas DevOps con soluciones de automatización de código abierto. Estas herramientas no solo son robustas y flexibles, sino que también tienen el beneficio adicional de ser gratuitas, lo que es crucial para empresas con presupuestos limitados. Además, estas herramientas cuentan con una amplia comunidad de usuarios y desarrolladores, lo que facilita el soporte y la colaboración.

2.2 Conclusiones y Recomendaciones

2.2.1 Conclusiones

La selección de herramientas de DevOps para la automatización del flujo de trabajo muestra un impacto positivo en la eficiencia, la calidad, la velocidad de entrega y la colaboración en el desarrollo y despliegue de software. Estas herramientas juegan un papel crucial en la transformación digital de las PyMEs y en su capacidad para mantenerse competitivas en un entorno empresarial en constante cambio.

Mediante el desarrollo de este trabajo se logra:

- Tabular las diferentes referencias bibliográfias utilizadas para el derrollo de esta investigación.
- Documentar mediante tablas las características y funcionalidades de las principales herramientas de DevOps.
- A raiz de la comparación y evaluación de las diferentes herramientas, que del universo de herramientas Devops se puede seleccionar las herramientas que más se adaptan a las PyMes de Bolivia, de acuerdo a criterios básicos: Escalables, de fácil integración, fáciles de usar, presupuestos reducidos.

En la Tabla Nº 12 se pueden apreciar las herramientas esenciales no solo para Pymes de Bolivia, sino que, de manera general para cualquier tipo de proyecto, ya que no siempre en un entorno empresarial se van a poder utilizar herramientas para cada flujo, sin embargo, se puede utilizar estas herramientas para la automatización de las tareas más críticas.

También cabe mencionar que se ha conseguido comparar las funcionalidades y la capacidad de adecuación de las herramientas dentro del flujo de trabajo de desarrollo de software.

En la busqueda de información referente a las Pymes de Bolivia y la relación con DevOps no se ha podido obtener mucha bibliografía, prácticamente nada, debido a que no existen antecedentes referentes a la adopción de Devops en las Pymes de Bolivia; sin embargo cabe mencionar que en otros ambitos como las startups, empresas bancarias, si llegan a adoptar DevOps como cultura dentro del desarrollo de software.

2.2.2 Recomendaciones

Para futuros trabajos lo ideal es poder realizar una investigación cualitativa, donde se puedan realizar entrevistas en profundidad, encuestas transversales a profesionales que trabajen en el desarrollo de software en Bolivia, ya que de esa manera será posible obtener una gran variedad y amplitud de respuestas, lo que en consecuencia permitirá una codificación más profunda y a su vez un diagnóstico de la realidad boliviana más certero. Además, se sugiere también obtener datos más precisos y específicos sobre los equipos de TI de las empresas, permitiendo una comparación directa entre la información obtenida en pequeñas, medianas y empresas grandes

de Bolivia, ya que en la actualidad existe muy poca documentación con referencia a las Pymes que adoptan DevOps .

En Bolivia existen casos de éxito de startups y de empresas que seguramente ocupan herramientas devops, sin embargo, esto no está documentado como información al cuál el público en general pueda acceder, justo allí se considera un punto de partida para nuevas investigaciones, que podrán aportar y ser referentes en un futuro para poder diagnosticar la realidad de Bolivia y Devops.

Como en todo proyecto que necesite implementar DevOps es necesario básicamente para la selección de herramientas tomar en cuenta: las necesidades de la empresa, los costos, los recursos con los que se cuenta, considerar la compatibilidad, escalabilidad , rendimiento de las herramientas y soporte de la misma.

Bibliografía

Adriana B. Foronda B., H. A. (02 de 2020). *Internet Bolivia ORG*. Obtenido de https://internetbolivia.org/wp-content/uploads/2023/06/Situacion-Economia-digital-en-Bolivia.pdf

Atlassian. (s.f.). *Atlassian*. Recuperado el 2024 de 04 de 12, de https://www.atlassian.com/es/devops

Azure. (s.f.). *Azure*. Obtenido de https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-devops

Banco de desarrollo de América Latina y el Caribe. (2023). Las pymes en Bolivia. pág. 53. Obtenido de https://scioteca.caf.com/bitstream/handle/123456789/2132/CAF_PYMES_BOLIVIA.pdf?sequence=4&isAllowed=y

Blanco, M. P. (2021). DevSecOps, Estado del Arte en el Contexto Boliviano . *INF-FCPN-PGI Revista PGI, (7),*. Obtenido de https://ojs.umsa.bo/ojs/index.php/inf_fcpn_pgi/article/view/112

Certuche S. C., Z. K. (2022). Proceso para fomentar y apoyar la adopción de DevOps en PyMEs de Software. *Revista Científica de la Universidad Distrital Francisco José de CALDAS*, 16. Obtenido de https://revistas.udistrital.edu.co/index.php/revcie/article/view/19644/18548

cloudbees. (2020). *https://www.cloudbees.com/whitepapers/what-is-devops.* Obtenido de https://www.cloudbees.com/whitepapers/what-is-devops

CloudZero. (s.f.). *The Definitive DevOps Tools List: 55 Tools For 2024*. Recuperado el 2024, de CloudZero: https://www.cloudzero.com/blog/devops-tools/

Digital ai. (2023). *Comprehensive List of DevOps Tools You'll Need*. Recuperado el 05 de 2023, de Digital.ai: https://digital.ai/es/glossary/comprehensive-list-of-devops-tools-youll-need/

Digital.ai. (2023). *digital.ai.* Obtenido de https://digital.ai/learn/devsecops-periodic-table/

Don McVittie, A. S. (2018). *The state of DevOps Tools.* Obtenido de https://devops.com/wp-content/uploads/2018/03/StateOfDevOpsTools_v13.pdf

Dynatrace. (2021). Cómo DevOps puede transformar las TI y el negocio. Obtenido de https://assets.dynatrace.com/mx/docs/wp/15886-wp-devops-for-digital-transformation-la-sp.pdf

Dynatrace. (2023). *DevOps Automation Pulse: The current state of DevOps automation.* Madrid. Obtenido de https://www.dynatrace.com/info/automation-maturity-report/

Escuela de Administración de Negocios Institución Universitaria. (s.f.). *EAN*, 27. Obtenido de https://www.redalyc.org/pdf/206/20652069006.pdf

Gil, C. A. (2002). *Como elaborar projetos de pesquisa.* Atlas.

IBM Limited Edition. (2015). *DevOps For Dummies.* Wiley.

Kim, G. B. (2018). *The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win 3th Edition.* IT Revolution Press.

Len Bass, I. W. (2015). *DevOps: A Software Architect's Perspective.* Sebago Lake: Addison-Wesley. Obtenido de http://alecoledelavie.com/accueil/vie_uploads/Portfolio_Programs_Projects_and%20BAU/PortFolio_stuff/Courses%20resources%20stuff/DELF%20cours/DevOps/DevOps%20Delf/Outils_devops/use_case_chapitre13/DevOps_%20A%20Software%20Architect's%20Perspective.pdf

Martínez, I. F. (2013). *Apuntes de Metodología de la Investigación: Enfoque Crítico.*

Microsoft. (28 de 01 de 2022). Aceleración digital: más del 84% de las pymes bolivianas invirtió en tecnología en el último año . *News Center Microsoft Latinoamérica*. Obtenido de https://news.microsoft.com/es-xl/aceleracion-digital-mas-del-84-de-las-pymes-bolivianas-invirtio-en-tecnologia-en-el-ultimo-ano/

Pathak, A. (11 de 04 de 2022). *Kinsta*. Obtenido de https://kinsta.com/es/blog/herramientas-devops/#herramientas-devops-pipeline-cicd

RedHat. (10 de 05 de 2022). *RedHat*. Obtenido de https://www.redhat.com/es/topics/devops

Sentrio. (06 de 10 de 2021). *Sentrio*. Recuperado el 2024, de https://sentrio.io/blog/herramientas-devops/

Transposit. (2023). *The State of DevOps Automation and AI 2023.* Obtenido de www.transposit.com/resources/the-state-of-devops-automation-and-ai-2023

Wilson, B. (11 de Julio de 2023). *DevOps Cube*. Obtenido de https://devopscube.com/devops-tools-for-infrastructure-automation/

Wilson, B. (02 de 08 de 2023). Devopscube. Obtenido de https://devopscube.com/best-devops-tools/

Anexos

Tabla 2 - *Lista de las Mejores Herramientas DevOps*

Categoria de las Herramienta	Mejores Herramientas DevOps
Code Build	Maven, npm & Gradle
Code Quality and Analysis	Sonarqube
Continuous Integration	Jenkins, Github Actions & Gitlab CI
Continuous Delivery & GitOps	Jenkins, FluxCD (GitOps), ArgoCD (GitOps)
Artifact Management	Sonatype Nexus, Artifactory
Infrastructure Provisioning	Terraform & Pulumi
Configuration Management	Ansible & Chef
Logging	ELK stack & Grafana Loki
Monitoring & Observability	Prometheus, Thanos and Grafana
Secret Management	Hashicorp Vault
Container Orchestration	Kubernetes
Service Mesh	Istio
Security	Aqua Security
Multi Cloud Management	Crossplane
Cloud Cost Management	Infracost, Kubecost (k8s)
Platform Engineering	Backstage
eBPF	Cilium, Falco, Calico

Nota: Lista de las Mejores Herramientas DevOps. Tomado de (CI): (Wilson, 90+ Best DevOps Tools, 2023)

Tabla 3- *Herramientas categorizadas de acuerdo a la licencia*

Proceso	Nombre de la Herramienta	Tipo de Licencia	Sitio Web
Integración Continua (CI):	Jenkins	Open Source	jenkins.io
	GitLab CI	Open Source	about.gitlab.com
	Travis CI	Comercial / Open Source	travis-ci.com
	CircleCI	Comercial / Open Source	circleci.com
	Bamboo	Comercial	atlassian.com/software/bamboo
Entrega Continua (CD):	Ansible	Open Source	ansible.com
	Puppet	Open Source	puppet.com
	Chef	Open Source	chef.io
	Docker	Open Source	docker.com

	Kubernetes	Open Source	kubernetes.io
Despliegue Continuo:	Terraform	Open Source	terraform.io
	Spinnaker	Open Source	spinnaker.io
	Argo CD	Open Source	argoproj.github.io/argo-cd
	Harness	Comercial	harness.io
	Octopus Deploy	Comercial	octopus.com
Automatización de Infraestructura:	Ansible	Open Source	ansible.com
	Puppet	Open Source	puppet.com
	Chef	Open Source	chef.io
	Terraform	Open Source	terraform.io
	AWS CloudFormation	Comercial	aws.amazon.com/cloudformation/
Monitoreo Continuo:	Prometheus	Open Source	prometheus.io
	Grafana	Open Source	grafana.com
	ELK Stack	Open Source	elastic.co
	Nagios	Open Source	nagios.org
	Datadog	Comercial	datadoghq.com
Gestión de Configuración:	Git	Open Source	git-scm.com
	Subversion	Open Source	subversion.apache.org
	Mercurial	Open Source	mercurial-scm.org
	Bitbucket	Comercial	bitbucket.org
	GitHub	Comercial	github.com

Nota: Herramientas Devops para diferentes procesos. Tomada de Elaboración Propia

Tabla 4 - *Cuadro de Analisis Documental*

Referencia Bibliográfica	Autor(es)	Año	Título	Contenido Principal	Enfoque Principal	Conclusiones Relevantes
DevOps For Dummies. Wiley	Emily Freeman	2020	DevOps For Dummies	- Introducción a los conceptos básicos de DevOps. - Exploración de prácticas y herramientas clave. - Discusión sobre la cultura, colaboración y automatización en DevOps.	Proporciona una visión general accesible y práctica de DevOps para lectores principiantes.	Ofrece una comprensión clara de los principios fundamentales, prácticas y beneficios de DevOps, así como pautas para su implementación efectiva.
The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win	Gene Kim, Kevin Behr, George Spafford	2018	The Phoenix Project	- Presentación de los desafíos en un equipo de TI y su transformación a través de principios DevOps. - Narrativa que ilustra la aplicación práctica de DevOps en un entorno empresarial. - Exploración de temas como flujo de trabajo, colaboración, automatización y mejora continua.	Utiliza una historia ficticia para transmitir los principios y prácticas de DevOps de manera entretenida y accesible.	Destaca la importancia de la colaboración entre equipos, la automatización de procesos y el enfoque en el flujo de trabajo para lograr una entrega de software más rápida y confiable.

20 DevOps Tools for Infrastructure Automation and Monitoring	DevOps Cube	2023	20 DevOps Tools for Infrastructure Automation and Monitoring	- Enumeración de 20 herramientas populares para automatización y monitoreo de infraestructura en el contexto de DevOps. - Breve descripción de cada herramienta, incluyendo características principales, casos de uso y enlaces a recursos adicionales. - Agrupación de herramientas según su función principal, como aprovisionamiento, gestión de configuración, orquestación de contenedores y monitoreo.	Proporciona una lista exhaustiva de herramientas útiles para la automatización y monitoreo de la infraestructura en entornos DevOps.	Ofrece una visión general de las herramientas disponibles, permitiendo a los lectores explorar y evaluar opciones según sus necesidades específicas de automatización e infraestructura de monitoreo.

https://news.microsoft.com/es-xl/aceleracion-digital-mas-del-84-de-las-pymes-bolivianas-invirtio-en-tecnologia-en-el-ultimo-ano/	Microsoft	2022	Aceleración digital: más del 84% de las pymes bolivianas invirtió en tecnología en el último año	- Estadísticas sobre la adopción de tecnología por parte de las PYMES bolivianas. - Discusión sobre el impacto de la tecnología en las PYMES. - Ejemplos de casos de éxito o testimonios de empresas bolivianas que han invertido en tecnología.	Ofrece información sobre la adopción de tecnología por parte de las PYMES en Bolivia y su impacto en el desarrollo empresarial.	Destaca la importancia de la inversión en tecnología para la competitividad y el crecimiento de las PYMES bolivianas, así como las oportunidades y desafíos asociados con la aceleración digital en el país.

https://digital.ai/es/glossary/comprehensive-list-of-devops-tools-youll-need/	Digital.ai	2023	Comprehensive List of DevOps Tools You'll Need	El contenido está organizado en diferentes categorías, que incluyen herramientas para la planificación y el seguimiento, la integración continua, la entrega continua, la automatización de la infraestructura, la monitorización y el análisis, entre otras. Cada herramienta está acompañada de una breve descripción que resalta sus funciones y beneficios	Presenta una lista completa de herramientas de DevOps que son esenciales para equipos de desarrollo y operaciones.	Ofrece una visión general exhaustiva de las herramientas de DevOps disponibles, lo que puede ayudar a los equipos a seleccionar las que mejor se adapten a sus necesidades y objetivos.

https://devopscube.com/best-devops-tools/	DevOpsCube	2023	90+ Best DevOps Tools	Posee una lista de más de 90 herramientas DevOps , organizados en diferentes categorías.	Las categorias de las herramientas, de cada herramienta nos mensiona el tipo de licencia que posee . También nos indica los factores que se deben considerar para la selección de herramientas DevOps.	La mayoría de las herramientas pertenecen a la categoría de codigo abierto. La amplitud de la lista nos da a conocer más de cada una de las herramientas, por lo menos saber la categoría de las herramientas y si son gratuitos o de pago

Nota: Cuadro de Analisis Documental, para analizar la Bibliografía más relevante relacionado con el tema de estudio. Tomada de Elaboración Propia

Tabla 5 - *Herramientas de Integración Continua (CI):*

Herramienta	Licencia	Escalabilidad	Compatibilidad con la Nube	Facilidad de Uso	Soporte	Ventajas	Desventaja
Jenkins	Open Source	Alta	AWS, Azure, GCP, Otros	Media/Alta	Comunidad/Soporte Técnico	Flexibilidad, Gran Comunidad, Muchos Plugins	Requiere configuración inicial y mantenimiento continuo. Algunas caracteristicas avanzadas pueden requerir complementos adicionales
GitLab CI	Open Source	Alta	AWS, Azure, GCP	Alta	Comunidad/Soporte Técnico	Integración con GitLab, Pipeline como código	La integración con sistemas externos puede ser menos fluida. Caracteristicas avanzadas limitadas a las ediciones de pago
Travis CI	Comercial	Media	AWS, Azure, GCP	Alta	Soporte Técnico/Premium	Integración con GitHub, Configuración sencilla	Puede resultar costoso para proyectos con grandes necesidades de integración continua. La ejecución de trabajo en paralelo está restringida para los planes gratuitos.
CircleCI	Comercial	Alta	AWS, Azure, GCP	Alta	Soporte Técnico/Premium	Facilidad de Configuración, Paralelismo de Trabajos	Los planes gratuitos tienen limitaciones en términos de tiempo de ejecución de trabajos y recursos disponibles. La integración con otras herramientas y servicios es menos completa
Bamboo	Comercial	Alta	AWS, Azure, GCP	Alta	Soporte Técnico/Premium	Integración con Atlassian, Interfaz de Usuario Clara	Al ser una herramienta de Atlassian casi no se integra con herramientas de terceros. El costo es alto para equipos pequeños.

Nota: Cuadro comparativo de herramientas CI. Tomada de Elaboración Propia

Tabla 6 - *Herramientas entrega continua (CD)*

Herramienta	Licencia	Escalabilidad	Compatibilidad con la Nube	Facilidad de Uso	Soporte	Ventajas	Desventajas
Ansible	Código Abierto	Alta	Amplia	Fácil	Comunidad activa, soporte comercial disponible	Fácil de aprender y usar, no requiere agentes en nodos remotos	Velocidad de ejecución puede ser más lenta en operaciones complejas
Puppet	Mixta	Alta	Amplia	Moderada	Comunidad activa, soporte comercial disponible	Configuración declarativa, gestión centralizada de la configuración	Curva de aprendizaje empinada, configuración inicial compleja
Chef	Mixta	Alta	Amplia	Moderada	Comunidad activa, soporte comercial disponible	Configuración como código, altamente personalizable y flexible	Curva de aprendizaje empinada, configuración inicial compleja
Docker	Código Abierto	Alta	Amplia	Fácil	Comunidad activa, soporte comercial disponible	Portabilidad de aplicaciones, aislamiento de recursos, eficiencia	Gestión de múltiples contenedores puede volverse compleja
Kubernetes	Código Abierto	Alta	Amplia	Moderada	Comunidad activa, soporte comercial disponible	Orquestación de contenedores, escalado automático, gestión de recursos	Curva de aprendizaje pronunciada, configuración y administración compleja

Nota: Cuadro comparativo de herramientas CD. Tomada de Elaboración Propia

Tabla 7 - *Herramientas despliegue continuo*

Herramienta	Licencia	Escalabilidad	Compatibilidad con la Nube	Facilidad de Uso	Soporte	Ventajas	Desventajas
Terraform	Open Source	Alta	AWS, Azure, GCP, Otros	Media/Alta	Comunidad/Soporte Técnico	Infraestructura como Código, Declarativo, Multiplataforma	La gestión de estado puede ser compleja en entornos distribuidos o de gran escala
Spinnaker	Open Source	Alta	AWS, Azure, GCP, Otros	Media/Alta	Comunidad/Soporte Técnico	Despliegue Multi-cloud, Gestión Completa del Flujo de Entrega	Requiere infraestructura sólida para el despliegue, lo que puede ser costoso y complejo de mantener.
Argo CD	Open Source	Alta	Kubernetes	Alta	Comunidad/Soporte Técnico	Despliegue Continuo en Kubernetes, Declarativo	No es tan maduro como tras herramientas, lo que puede llevar a problemas de estabilidad o falta de caracteristicas avanzadas.
Harness	Comercial	Alta	AWS, Azure, GCP, Otros	Alta	Soporte Técnico/Premium	Plataforma Todo en Uno, Automatización de Despliegues	Puede ser costosa para empresas pequeñas o startups.Está más optimizada para entornos de nube, lo que limita su utilidad en entornos locales o híbridos.
Octopus Deploy	Comercial	Alta	AWS, Azure, GCP, Otros	Alta	Soporte Técnico/Premium	Despliegue Multiplataforma, Facilidad de Uso	Puede ser costosa para equipos con presupuesto limitado. Enfoque principalmente en e despliegue de aplicaciones .NET

Nota: Cuadro comparativo de herramientas de despliegue continuo. Tomada de Elaboración Propia

Tabla 8 - *Herramientas Automatización de Infraestructura*

Herramienta	Licencia	Escalabilidad	Compatibilidad	Facilidad de Uso	Soporte	Ventajas	Desventajas
Ansible	Open Source	Alta	Multiplataforma (Linux, Windows, etc.)	Alta	Comunidad/Soporte Técnico	Configuración simple y fácil de aprender. Agentless, no requiere instalar software en los nodos objetivo.	No ofrece un estado declarativo como Puppet o Chef. Puede ser menos eficiente en entornos muy grandes.
Puppet	Open Source	Alta	Multiplataforma (Linux, Windows, etc.)	Media/Alta	Comunidad/Soporte Técnico	Aborda configuraciones complejas con un modelo declarativo. Escalabilidad y gestión centralizada.	Curva de aprendizaje más pronunciada que Ansible. Requiere un agente instalado en los nodos objetivo.
Chef	Open Source	Alta	Multiplataforma (Linux, Windows, etc.)	Media/Alta	Comunidad/Soporte Técnico	Modelado de infraestructura declarativo. Gran flexibilidad y soporte para múltiples plataformas.	Requiere un agente instalado en los nodos objetivo. Configuración inicial más compleja que Ansible.
Terraform	Open Source	Alta	AWS, Azure, GCP, Otros	Media/Alta	Comunidad/Soporte Técnico	Infraestructura como código (IaC) para la gestión de recursos en la nube. Soporte multiplataforma.	La curva de aprendizaje puede ser empinada para usuarios nuevos. No gestiona la configuración de software.
AWS CloudFormation	Comercial	Alta	AWS	Media	Soporte Técnico/Premium	Totalmente integrado con el ecosistema de AWS. Infraestructura como código nativo para AWS.	Limitado a AWS, no es fácilmente portable a otras nubes. La sintaxis puede ser verbosa y compleja.
Spinnaker	Open Source	Alta	AWS, Azure, GCP, Otros	Media/Alta	Comunidad/Soporte Técnico	Orquestación de entrega continua (CI/CD) con capacidades avanzadas de despliegue. Amplia compatibilidad.	Configuración inicial compleja y curva de aprendizaje pronunciada. Requiere infraestructura adicional.

Argo CD	Open Source	Alta	Kubernetes	Alta	Comunidad/Soporte Técnico	Automatización de despliegues de aplicaciones en Kubernetes. Integración directa con GitOps.	Limitado a entornos Kubernetes. La curva de aprendizaje puede ser pronunciada para usuarios nuevos.
Harness	Comercial	Alta	AWS, Azure, GCP, Otros	Alta	Soporte Técnico/Premium	Automatización de CI/CD con soporte para múltiples plataformas y servicios en la nube. Visualizaciones y análisis avanzados.	Requiere una curva de aprendizaje inicial. Puede ser costoso para grandes implementaciones.
Octopus Deploy	Comercial	Alta	AWS, Azure, GCP, Otros	Alta	Soporte Técnico/Premium	Despliegue automatizado de aplicaciones multiplataforma con una interfaz fácil de usar. Gestión centralizada.	Centrado en aplicaciones, no maneja infraestructura. Limitado a entornos Windows y Linux.

Nota: Cuadro comparativo de herramientas de infraestructura. Tomada de Elaboración Propia

Tabla 9 - *Herramientas Monitoreo Continuo*

Herramienta	Licencia	Escalabilidad	Compatibilidad con la Nube	Facilidad de Uso	Soporte	Ventajas	Desventajas
Prometheus	Código Abierto	Alta	Amplia	Moderada	Comunidad activa, soporte comercial disponible	Recolección y almacenamiento de métricas de forma eficiente	Configuración inicial puede ser compleja, falta de interfaces gráficas avanzadas
Grafana	Código Abierto	Alta	Amplia	Fácil	Comunidad activa, soporte comercial disponible	Visualización de datos con gran flexibilidad y personalización	Puede requerir integración con otras herramientas para monitoreo activo

ELK Stack	Mixta	Alta	Amplia	Moderada	Comunidad activa, soporte comercial disponible	Análisis de logs en tiempo real, almacenamiento y visualización	Configuración y mantenimiento puede ser complejo, requerimientos de hardware
Nagios	Código Abierto	Moderada	Amplia	Moderada	Comunidad activa, soporte comercial disponible	Monitoreo de infraestructura y servicios en tiempo real	Configuración y personalización pueden requerir experiencia técnica
Datadog	Comercial	Alta	Amplia	Fácil	Soporte comercial	Monitoreo y análisis de infraestructura, aplicaciones y servicios	Costo, puede ser prohibitivo para pequeñas empresas

Nota: Cuadro comparativo de herramientas de monitoreo continuo. Tomada de Elaboración Propia

Tabla 10 - *Herramientas Gestión de Configuración*

Herramienta	Licencia	Escalabilidad	Compatibilidad	Facilidad de Uso	Soporte	Ventajas	Desventaja
Git	Open Source	Alta	Multiplataforma (Linux, Windows, macOS)	Alta	Comunidad/Soporte Técnico	Distribuido, Ramificación y Fusión Fácil, Rápido	La gestión de grandes repositorios puede ser compleja y lenta. La complejidad de las ramificaciones y fusiones puede causar confusiones
Subversion	Open Source	Alta	Multiplataforma (Linux, Windows, macOS)	Media/Alta	Comunidad/Soporte Técnico	Centralizado, Manejo de Archivos Binarios, Buen Soporte de Ramificación	Al ser centralizado solo depende de un único repositorio, lo cual puede resultar en problemas de rendimiento y disponibilidad si el servidor central falla.

Mercurial	Open Source	Alta	Multiplataforma (Linux, Windows, macOS)	Media/Alta	Comunidad/Soporte Técnico	Distribuido, Fácil de Aprender, Buen Soporte de Ramificación	Menor cantidad de herramientas y servicios en comparación a Git.
Bitbucket	Comercial	Alta	Git	Alta	Soporte Técnico/Premium	Integración con Jira, Pipelines de CI/CD, Repositorios Privados Gratuitos	La integración con otras herramientas no es tán extensa como en otras plataformar. Documentación y soporte no es tan amplio.
GitHub	Comercial	Alta	Git	Alta	Soporte Técnico/Premium	Amplia Comunidad, Integración con Herramientas de Desarrollo, GitHub Actions	Costoso para pequeños equipos Dependencia de un servicio en la nube.

Nota: Cuadro comparativo de herramientas gestión de configuración. Tomada de Elaboración Propia

A continuación se muestra una tabla comparativa elaborada con algunas herramientas DevOps populares basadas en varios criterios, con puntajes del 1 al 5 (donde 1 es el puntaje más bajo y 5 el más alto):

Tabla 11 – *Tabla comparativa de herramientas DevOps*

Etapa	Herramienta	Facilidad de uso	Funcionalidad	Costo	Soporte técnico	Compatibilidad	Escalabilidad	Seguridad	Integraciones	Puntaje Total
Gestión del Código Fuente	GitHub	5	5	4	5	5	4	5	5	38
	GitLab	4	5	4	4	5	4	5	5	36
	Bitbucket	4	4	4	4	5	4	4	4	33
	Azure DevOps	4	4	4	4	5	4	4	5	34

	AWS CodeCommit	3	4	5	5	5	4	5	4	35
CI/CD	Jenkins	3	5	4	4	5	5	4	5	35
	GitLab CI	4	5	4	4	5	4	5	5	36
	CircleCI	4	5	4	4	5	4	4	5	35
	Travis CI	3	4	4	4	4	4	4	4	31
	GitHub Actions	5	5	4	5	5	4	5	5	38
Automatización y Configuración	Ansible	4	5	4	4	5	5	5	5	37
	Puppet	3	5	4	4	5	5	5	5	36
	Chef	3	5	4	4	5	5	5	5	36
	SaltStack	3	4	4	4	5	4	5	4	33
	Terraform	4	5	4	4	5	5	5	5	37
Contenerización	Docker	5	5	4	4	5	5	5	5	38
	Podman	4	4	4	4	4	4	5	4	33
	rkt	3	4	4	4	4	4	5	4	32
	LXC	3	3	4	3	4	4	4	3	28
	Buildah	4	4	4	4	4	4	4	4	32
Orquestación de Contenedores	Kubernetes	4	5	4	4	5	5	5	5	37
	Docker Swarm	4	4	4	4	4	4	4	4	32
	Amazon ECS	4	5	4	5	5	5	5	5	38
	Google Kubernetes Engine (GKE)	4	5	4	5	5	5	5	5	38
	Azure Kubernetes Service (AKS)	4	5	4	5	5	5	5	5	38

Monitoreo y Logging	Prometheus	4	4	5	5	5	5	4	5	37
	Grafana	4	4	5	5	5	4	4	5	36
	ELK Stack	3	5	4	4	5	5	5	5	36
	Datadog	4	5	3	4	5	4	5	5	35
	New Relic	4	5	3	4	5	4	5	5	35

Nota: Tabla comparativa basada en la facilidad de uso, funcionalidad, costo, soporte técnico, compatibilidad, escalabilidad, seguridad, capacidad de integrarse. Tomada de elaboración propia.

A continuación se muestra una tabla comparativa elaborada con herramientas DevOps Open Source basadas en varios criterios, con puntajes del 1 al 10 (donde 1 es el puntaje más bajo y 10 el más alto):

Tabla 12– *Tabla comparativa de herramientas DevOps Open Source*

Categoría	Herramienta	Facilidad de Uso	Funcionalidad	Costo	Soporte Técnico	Compatibilidad	Escalabilidad	Seguridad	Integraciones
Gestión de Código Fuente	GitLab	9	9	10	8	9	9	9	9
	Git	8	8	10	7	9	8	9	8
CI/CD	Jenkins	7	10	10	8	9	9	8	10
	GitLab CI	9	9	10	8	9	9	9	9
	Travis CI	9	8	10	7	9	8	8	8

Automatización y Configuración	Ansible	9	9	10	8	9	9	9	9
	Terraform	8	9	10	8	9	9	9	9
	Puppet	7	9	10	8	9	9	9	9
Contenerización	Docker	9	9	10	9	9	9	9	9
Orquestación de Contenedores	Kubernetes	7	10	10	8	9	10	9	10
Monitoreo y Logging	Prometheus	8	9	10	8	9	9	9	9
	Grafana	8	9	10	8	9	9	9	9

Nota: Lista de son ideales para PyMes que buscan adoptar prácticas DevOps con soluciones de automatización de código abierto.Tomada de elaboración propia.