



Universidad Internacional de La Rioja
Escuela Superior de Ingeniería y Tecnología

Máster Universitario en Desarrollo y Operaciones (DevOps)
**Catálogo de herramientas de gestión de la
configuración que implementan la
metodología GitOps**

Trabajo fin de estudio presentado por:	Johan Sebastian Giraldo Hurtado
Tipo de trabajo:	Desarrollo práctico
Director/a:	Juan Manuel González Calleros
Codirector/a:	Luis Eduardo Sepúlveda Rodríguez
Fecha:	20 de septiembre del 2023

Dedicatoria

Este trabajo está dedicado a mis compañeros de trabajo del área de *Release Management* del Banco Scotiabank Colpatria, quienes durante este largo tiempo me proporcionaron todo su apoyo y paciencia para cumplir este objetivo; además la influencia del equipo me ha ayudado a crecer profesionalmente en mi carrera laboral.

Johan Sebastian Giraldo Hurtado

Agradecimientos

Agradezco a Dios, por brindarme la oportunidad en cuanto a vida, recursos y esfuerzos para estudiar esta maestría.

Agradezco a mi padre, *Jose Oscar Giraldo Herrera (q.e.p.d)* y mi madre *Fabiola Hurtado Buitrago, Otan (q.e.p.d)*, mi familia, compañeros de estudio y de trabajo, les agradezco haber creído en mí, cuando yo en ocasiones puede haber dudado de mis capacidades, gracias por creer en mí y que me convertiría en un gran profesional.

Agradezco a la UNIR - Universidad Internacional de La Rioja, a sus directivos, docentes y demás personas que hacen parte de la Maestría en Desarrollo y Operaciones (DevOps), por todo el apoyo prestado durante este proceso académico.

Agradezco a Juan Manuel González Calleros director del presente trabajo de final de Máster, por las asesorías permanentes y tiempo dedicado en todo este proceso.

Agradezco a Luis Eduardo Sepúlveda Rodríguez codirector del presente trabajo de final de Máster, por toda la ayuda brindada en este proceso, por guiarme en este camino para poder culminar esta etapa, también quiero agradecerle por haberme inspirado a seguir su ejemplo y pasión por los sistemas operativos tipo GNU/Linux.

Por último, le agradezco a Manuel Morejon Espinosa por su constante ayuda en mi carrera profesional y en mi rol como ingeniero DevOps, gracias por contagiarme todo el espíritu de la magia de los contenedores de Docker.

Resumen

Poner en marcha el funcionamiento de una característica de software es todo un reto para cualquier organización, desde su planificación, desarrollo y puesta en escena.

La infraestructura en la que es desplegada esta nueva característica también es un elemento clave para su funcionamiento. Como se puede contemplar existen numerosos elementos (personas, procesos, artefactos, herramientas, instrumentos, etc.) que influyen en todo este proceso. La administración de todos los elementos considerados anteriormente resulta ser una actividad tediosa, por eso es necesario unir ciertas prácticas y metodologías que permitan obtener el panorama general desde el inicio del proceso de construcción de software, hasta las fases finales de implementación y despliegue.

La empresa *Puppet* menciona en su reporte del estado DevOps (Kersten et al, 2021) que las organizaciones que han implementado la filosofía DevOps hasta el momento, han incrementado el rendimiento de sus equipos de TI.

En base a las prácticas que promueve la filosofía DevOps, se pretende construir un catálogo de herramientas que permita abarcar todos los elementos que son implicados en la puesta en marcha de un proyecto de desarrollo de software, sin dejar de lado todos los factores que pueden verse involucrados como por ejemplo uno de ellos es la gestión de la configuración, la cual busca mantener los sistemas informáticos, los servidores y el software en un estado deseado y uniforme.

No obstante son muchos los elementos que intervienen en la construcción y puesta marcha de un proyecto de software; tomando en consideración que la gestión de la configuración es un elemento importante en el desarrollo de software y teniendo en cuenta las nuevas tendencias tecnológicas que han surgido en los últimos años. Se ha determinado probar una herramienta que implemente la metodología GitOps y construir una guía de despliegue. La cual evaluará las características y aspectos más importantes, para determinar las ventajas y desventajas en cada etapa del ciclo de vida DevOps.

Posteriormente se desplegará un prototipo funcional el cual permita hacer la demostración de todo el proceso de integración y despliegue continuo de una aplicación para observar su

funcionamiento y poder concluir sobre el comportamiento de la herramienta que implementa la metodología GitOps.

Este catálogo de herramientas ayudará tanto a las organizaciones, empresas y usuarios a seleccionar la herramienta GitOps que sea pertinente para la puesta en marcha de un proyecto de software.

Palabras clave: GitOps, DevOps, Continuous Integration, Continuous Deployment, Continuous Delivery.

Abstract

Getting a software feature up and running is a challenge for any organization, from its planning, development, and staging.

The infrastructure in which this new feature is deployed is also a key element for its operation. As you can see, there are numerous elements (people, processes, artifacts, tools, instruments, etc.) that influence this entire process. The administration of all the elements considered above turns out to be a tedious activity, which is why it is necessary to combine certain practices and methodologies that allow obtaining the general picture from the beginning of the software construction process, until the final phases of implementation and deployment.

The Puppet Company mentions in its DevOps state report (Kersten et al., 2021) that organizations that have implemented the DevOps philosophy so far have increased the performance of their IT teams.

Based on the practices promoted by the DevOps philosophy, the aim is to build a catalog of tools that covers all the elements that are involved in the implementation of a software development project, without leaving aside all the factors that can be seen. involved, for example, one of them is configuration management, which seeks to maintain computer systems, servers, and software in a desired and uniform state.

However, there are many elements involved in the construction and implementation of a software project; taking into consideration that configuration management is an important element in software development and taking into account the new technological trends that have emerged in recent years. It has been determined to test a tool that implements the GitOps methodology and build a deployment guide. Which will evaluate the most important features and aspects, to determine the advantages and disadvantages at each stage of the DevOps life cycle.

Subsequently, a functional prototype will be deployed which allows the demonstration of the entire process of integration and continuous deployment of an application to observe its operation and be able to conclude on the behavior of the tool that implements the GitOps methodology.

This catalog of tools will help organizations, companies, and users select the GitOps tool that is relevant for the implementation of a software project.

Keywords: GitOps, DevOps, Continuous Integration, Continuous Deployment, Continuous Delivery.

Índice de contenidos

1. Introducción	15
1.1. Justificación del trabajo	16
1.2. Planteamiento del problema	16
1.3. Estructura de la memoria	18
2. Contexto y estado del arte	19
2.1. Contextualización y antecedentes	19
2.1.1. Equipo de desarrollo	19
2.1.2. Equipo de Operaciones	20
2.1.3. Ciclo de vida del desarrollo de software	20
2.1.4. DevOps	21
2.1.5. Release management	21
2.1.6. Control de versiones con Git	21
2.1.7. Integración continua	23
2.1.8. Entrega continua	24
2.1.9. Despliegue continuo	24
2.1.10. Entrega continua vs despliegue continuo	24
2.1.11. Computación en la nube	25
2.1.12. Contenedores	26
2.1.13. Kubernetes	26
2.1.14. GitOps	27
2.2. Antecedentes	27
2.3. Trabajos relacionados	33
2.4. Conclusiones del estado del arte	38
3. Objetivos y metodología de trabajo	41
3.1. Objetivo general	41
3.2. Objetivos específicos	41
3.3. Metodología del trabajo	41
4. Desarrollo específico de la contribución	43

4.1. Planificación / Análisis / Requisitos	43
4.1.1. Buscar e identificar herramientas (técnicas)	43
4.1.2. Determinar aspectos a evaluar de las herramientas	45
4.1.3. Definir/Determinar Características de las herramientas	48
4.1.4. Construir el catálogo de herramientas GitOps caracterizadas	49
4.1.5. Evaluar las alternativas	51
4.2. Descripción del sistema desarrollado / Implementación	58
4.2.1. Construir la guía de despliegue	58
4.2.1.1. Describir prerrequisitos para la instalación	58
4.2.1.2. Planificar la instalación	60
4.2.1.3. Describir el proceso de instalación	61
4.2.1.4. Describir aspectos de afinamiento en la configuración del ambiente	64
4.2.1.5. Describir el proceso de pruebas de la herramienta seleccionada	70
4.2.1.6. Describir aspectos de mantenibilidad de la herramienta	71
4.3. Evaluación	72
4.3.1. Implementar la herramienta seleccionada	72
4.3.1.1. Cumplir prerrequisitos indicados en la guía de despliegue	72
4.3.1.2. Realizar proceso de instalación de la herramienta seleccionada	74
4.3.1.3. Cumplir aspectos de configuración indicados en la guía de despliegue	76
4.3.1.4. Verificar el despliegue y el cumplimiento funcional de la herramienta seleccionada	79
4.3.2. Ejecutar la guía de despliegue a través de la implementación de un prototipo funcional de la herramienta GitOps seleccionada	83
4.3.2.1. Desplegar el prototipo funcional en el ambiente previamente creado	83
4.3.2.2. Cumplir las pruebas indicadas en la guía de despliegue	94
4.3.2.3. Realizar el informe de los resultados obtenidos en las pruebas	102
5. Conclusiones y trabajo futuro	103
5.1. Conclusiones	103
5.2. Líneas de trabajo futuro	105
5.3. Cumplimiento de Objetivos	105

Referencias bibliográficas	107
Anexos	112
Apéndice	112
A. Abreviaciones y Acrónimos	112
B. Estructura de trabajo desglosa	113
C. Actividades del proyecto trasladadas a un diagrama de Gantt	114
C.1. Herramienta OpenProject	114
C.2. Diagrama de Gantt - GitOps	114
C.3. Diagrama de Gantt - GitOps - Fases	115
C.4. Diagrama de Gantt - GitOps - Fase 0	115
C.5. Diagrama de Gantt - GitOps - Fase 1	116
C.6. Diagrama de Gantt - GitOps - Fase 2	116
C.7. Diagrama de Gantt - GitOps - Fase 3	117
C.8. Diagrama de Gantt - GitOps - Fase 4	117
C.9. Diagrama de Gantt - GitOps	118
D.1. Un SMS de herramientas que implementan la metodología GitOps - página 1	119
D.2. Un SMS de herramientas que implementan la metodología GitOps - página 2	120
D.3. Un SMS de herramientas que implementan la metodología GitOps - página 3	121
D.4. Un SMS de herramientas que implementan la metodología GitOps - página 4	122
D.5. Un SMS de herramientas que implementan la metodología GitOps - página 5	123
D.6. Un SMS de herramientas que implementan la metodología GitOps - página 6	124
D.7. Un SMS de herramientas que implementan la metodología GitOps - página 7	125
D.8. Un SMS de herramientas que implementan la metodología GitOps - página 8	126
D.9. Un SMS de herramientas que implementan la metodología GitOps - página 9	127
D.10. Un SMS de herramientas que implementan la metodología GitOps - página 10	128
D.11. Un SMS de herramientas que implementan la metodología GitOps - página 11	129
D.12. Un SMS de herramientas que implementan la metodología GitOps - página 12	130
D.13. Un SMS de herramientas que implementan la metodología GitOps - página 13	131
D.14. Un SMS de herramientas que implementan la metodología GitOps - página 14	132

D.15. Un SMS de herramientas que implementan la metodología GitOps - página 15	133
D.16. Un SMS de herramientas que implementan la metodología GitOps - página 16	134
D.17. Un SMS de herramientas que implementan la metodología GitOps - página 17	135
D.18. Un SMS de herramientas que implementan la metodología GitOps - página 18	136
D.19. Un SMS de herramientas que implementan la metodología GitOps - página 19	137
D.20. Un SMS de herramientas que implementan la metodología GitOps - página 20	138
D.21. Un SMS de herramientas que implementan la metodología GitOps - página 21	139
D.22. Un SMS de herramientas que implementan la metodología GitOps - página 22	140
D.23. Un SMS de herramientas que implementan la metodología GitOps - página 23	141

Índice de figuras

Figura 1. Flujo de trabajo con GitOps	17
Figura 2. Fases definidas	42
Figura 3. Modelo del proceso de instalación - paso 1	60
Figura 4. Modelo del proceso de instalación - paso 2	61
Figura 5. Modelo del proceso de instalación - paso 3	63
Figura 6. Interfaz de usuario de Argo CD en la sección Settings/Repositories	65
Figura 7. Interfaz de usuario de Argo CD en la sección Settings/Repositories + Connect Repo	65
Figura 8. Interfaz de usuario de Argo CD - Test - Connect Repo	66
Figura 9. Flujo de trabajo con GitOps + Argo CD Image Updater	67
Figura 10. Modelo del proceso de instalación - paso 4	69
Figura 11. Modelo del proceso de instalación - paso 5	71
Figura 12. Modelo del proceso de instalación - paso 6	72
Figura 13. Ejecución del comando: kubectl version --short	73
Figura 14. Ejecución del comando: helm version --short	73
Figura 15. Imagen de la aplicación Docker Desktop	74
Figura 16. Ejecución del comando: helm repo add argo https://argoproj.github.io/argo-helm	75
Figura 17. Ejecución del comando: helm install argo-cd argo/argo-cd -n argocd --create-namespace	75
Figura 18. Configuración de credenciales del repositorio de Git	76
Figura 19. Instalación del Argo CD Image Updater por medio de manifiestos YAML	77
Figura 20. Verificación de la Instalación del Argo CD Image Updater por consola	77
Figura 21. Configuración del objeto ConfigMap que permite modificar el repositorio de las imágenes de Docker	78
Figura 22. Verificación de los valores aplicados por el manifiesto YAML en el objeto ConfigMap	79
Figura 23. Ejecución del comando: helm ls -n argocd	79
Figura 24. Ejecución del comando: helm status argo-cd -n argocd	80
Figura 25. Ejecución del comando: kubectl get ns	81
Figura 26. Ejecución del comando: kubectl -n argocd get pods	81
Figura 27. Imagen de la aplicación Lens - Workloads - Overview	82
	12

Figura 28. Imagen de la aplicación Lens - Workloads - Pods	83
Figura 29. Ejecución del comando que permite hacer forwarding del puerto	84
Figura 30. Interfaz web de la herramienta Argo CD	85
Figura 31. Ejecución del comando <code>kubectl -n argocd get secret argocd-initial-admin-secret</code>	85
Figura 32. Inicio de sesión en la herramienta Argo CD	86
Figura 33. Creación del primer despliegue en Argo CD - parte 1	87
Figura 34. Creación del primer despliegue en Argo CD - parte 2	88
Figura 35. Despliegue del prototipo funcional - parte 1	88
Figura 36. Despliegue del prototipo funcional - parte 2	89
Figura 37. Ciclo de reconciliación ejecutado manualmente por el usuario	89
Figura 38. Verificación del despliegue de la aplicación por medio del ciclo de reconciliación	90
Figura 39. Verificación del despliegue de la aplicación por la CLI	90
Figura 40. Verificación del despliegue de la aplicación por medio de lens	91
Figura 41. Selección del pod desplegado por medio de lens	92
Figura 42. Selección del puerto para realizar el forwarding	92
Figura 43. Mensaje de verificación del forwarding	93
Figura 44. Servicio web desplegado a través de un deployment de kubernetes por medio de Argo CD	
Figura 45. Desplegar el prototipo funcional por medio de un archivo YAML	95
Figura 46. Desplegar el prototipo funcional por medio de un archivo YAML	95
Figura 47. Desplegar el prototipo funcional por medio de un archivo YAML	96
Figura 48. Desplegar el prototipo funcional por medio de un archivo YAML	96
Figura 49. Sincronización del repositorio de Git y el cluster de Kubernetes	97
Figura 50. Revisión de la versión del despliegue	98
Figura 51. Logs del controlador Argo CD Image Updater	99
Figura 52. Diferencia entre la versión de la imagen de Docker	100
Figura 53. Despliegue con la nueva versión de la imagen de Docker	101
Figura 54. Proceso de destrucción y construcción del otro Pod con la nueva versión	101
Figura 55. Revisión de la nueva versión del despliegue	102

Índice de tablas

Tabla 1. Resultado de las consultas en las bases de datos	45
Tabla 2. Cuadro comparativo de las etapas del ciclo DevOps versus las características de la herramienta	46
Tabla 3. Cuadro comparativo de las características versus la herramienta	49
Tabla 4. Catálogo de las herramientas	51
Tabla 5. Pesos criterios de selección	54
Tabla 6. Matriz DAR - Decision, Analysis and Resolution	57
Tabla 7. Resumen de cumplimiento de objetivos	106

1. Introducción

La conceptualización del término GitOps entendida como: “GitOps es lo mejor desde la configuración como código. Git cambió la forma en que colaboramos, pero la configuración declarativa es la clave para manejar la infraestructura a escala y sienta las bases para la próxima generación de herramientas de administración” fue emitida por Hightower (2019) y aunque carece de formalismo, su impacto ha sido relevante en la industria del software. En este contexto, el presente trabajo de fin de Máster se enfoca en examinar la forma de trabajo, las características de la metodología GitOps y las herramientas tecnológicas que permiten su implementación.

Es relevante reconocer que, por medio de la metodología GitOps es posible declarar en un repositorio la aplicación en desarrollo como la infraestructura necesaria para su despliegue. Además, proporciona la capacidad de definir el comportamiento de este despliegue y los pasos necesarios para llevarlo a cabo. Esta metodología es especialmente valiosa ya que permite declarar en un repositorio todos los componentes de la infraestructura necesaria, lo que garantiza coherencia y evita inconsistencias entre los distintos entornos, ya sean de prueba o producción. En resumen, GitOps otorga un control preciso sobre las versiones de software y los recursos computacionales, optimizando la gestión de todo el ciclo de vida de las aplicaciones y la infraestructura asociada.

Otros aspectos a considerar con la forma de trabajo GitOps, es la seguridad, lo anterior representado en la gestión de acceso que puede darse a los usuarios del repositorio. Por una parte, los desarrolladores solo se centraran en enviar peticiones de *Pull Request* para modificar su aplicación y del mismo modo, el equipo de operaciones, se encargaran de mantener los manifiestos de la infraestructura que es desplegada a través de *Kubernetes*. Además, una ventaja de limitar y dividir las responsabilidades beneficia a los profesionales de IT, al enfocar sus funciones y habilidades. Se ha vuelto normal en las organizaciones que todos los profesionales deben de tener un mar de conocimientos con un centímetro de profundidad, lo que significa que el desarrollador en muchos casos realiza el despliegue sin

tener conocimientos profundos de la infraestructura y precisamente este tipo de escenarios son los que se buscan evitar en la actualidad.

1.1. Justificación del trabajo

El gobierno del ciclo de vida del desarrollo de software (también conocido como SDLC o *Systems Development Life Cycle*) es un modelo complejo donde existe interacción humana, esta puede ser propensa a errores como por ejemplo; desarrollar mal una funcionalidad o característica, versionar incorrectamente, no realizar las pruebas correctamente, omitir las pruebas requeridas (unitarias, integración, performance, seguridad) y esto conlleva a que el despliegue no sea el adecuado.

En consecuencia existirá o se generará un inconveniente entre el equipo de desarrollo y operaciones. Esto a su vez ocasionará un reproceso para ambos equipos, en donde deberán ejecutar tareas manuales aumentando la probabilidad de incurrir en el mismo error.

Las demoras o retrasos en las entregas de los productos o funcionalidades que se acuerdan con un cliente, generan una percepción de desagrado causando a su vez desconfianza con el equipo o empresa que este haya contratado. La confianza que se establece con los clientes es fundamental para el crecimiento y desarrollo de las organizaciones, permitiéndoles participar en otros proyectos o postularse a nuevos.

1.2. Planteamiento del problema

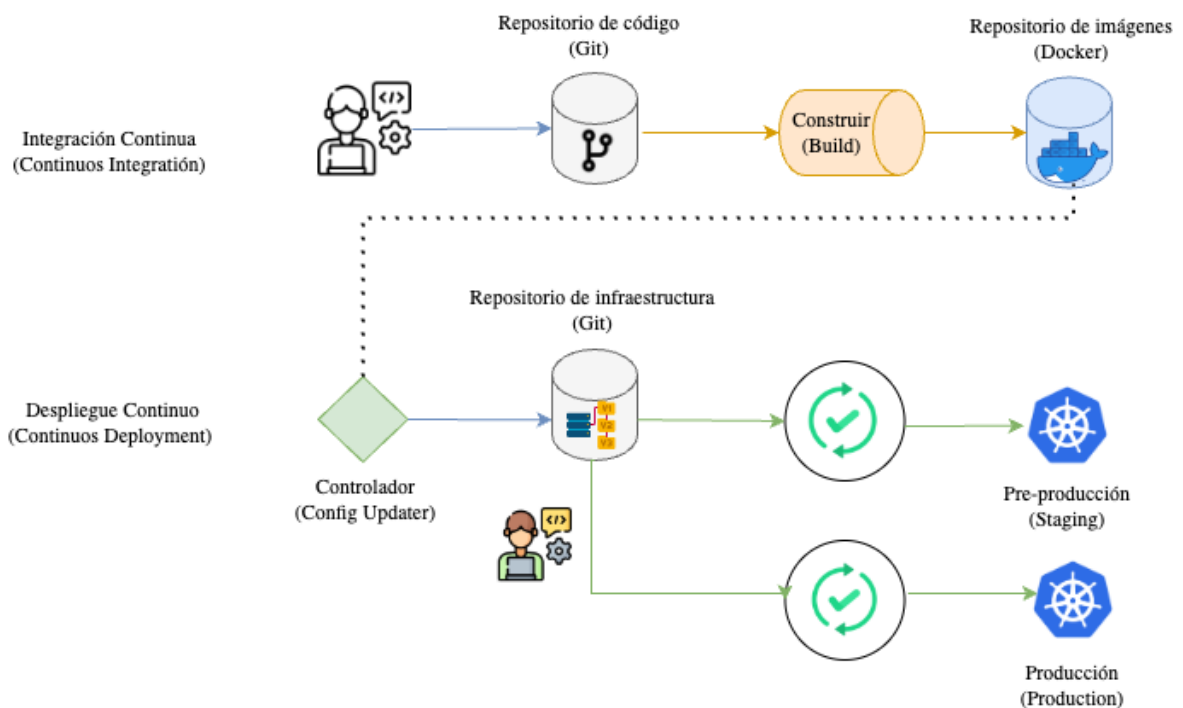
Una opción para abordar esta problemática sería, seleccionar una serie de prácticas, procesos, estándares o metodologías que permitan la automatización del ciclo de vida del desarrollo de software, sin embargo esto no sería suficiente, también se pretende garantizar la fiabilidad de la infraestructura que aloja el software que es desarrollado.

A inicio del año 2017 (Works, 2017), la empresa *Weaveworks* comparte las experiencias que ha obtenido usando la herramienta *Kubernetes* (esta herramienta es construida por *Google*, pensada en la época actual para suplir las necesidades de la nube, pero también se puede

usar *On premise*). Esta herramienta de *Kubernetes* se une con los flujos de integración y despliegue continuo (CI/CD) y como resultado se origina una nueva metodología llamada **GitOps**.

Como su nombre lo dice **GitOps** utiliza la herramienta de *Git* como única fuente de la verdad, lo que indica que todos los elementos que sean almacenados en un repositorio, corresponderá a los que se vean reflejados en la infraestructura (Kapelonis et al., 2022).

Figura 1. Flujo de trabajo con GitOps. Según (Weaveworks, 2019).



Fuente: elaboración propia.

Se plantea realizar la búsqueda e identificación de herramientas que implementen la metodología GitOps, de tal manera que se puedan caracterizar teniendo en cuenta los elementos, aspectos y componentes más relevantes.

Luego, se efectuará la evaluación de estas herramientas teniendo en cuenta el área y/o disciplina de **CMMI** denominada **Decision Analysis and Resolution (DAR)** de tal forma que, se eliminen los juicios de valor y se puedan tomar decisiones basadas en indicadores, para caracterizar las herramientas que implementan la metodología GitOps.

Posteriormente, se selecciona la herramienta que mejor resultados tenga de la evaluación para construir una guía de despliegue que contenga elementos como por ejemplo requerimientos, pasos para la instalación, configuración y mantenimiento.

Por último se evaluará la guía de despliegue de la herramienta que implementa la metodología GitOps por medio de un prototipo funcional para observar cómo se comporta este, en todo el ciclo de vida de desarrollo de software.

1.3. Estructura de la memoria

A continuación se describen brevemente las fases o etapas posteriores que corresponden a la construcción del documento y desarrollo del trabajo final de Máster.

En primer lugar se encuentra el **contexto y estado del arte** en donde se hace la búsqueda, identificación, selección de los trabajos académicos relacionados con la temática a tratar en el presente trabajo, también se menciona a los principales referentes en el desarrollo del tema de investigación.

En segundo lugar se encuentran los **objetivos y metodología de trabajo**, en este apartado del trabajo se especifica la motivación de todo el trabajo final de maestría, el cual se plasma se representa en el objetivo principal y se desglosa en objetivos alcanzables para poder lograr esta meta.

En la metodología de trabajo se mencionan los modelos utilizados o las guías de referencia para establecer un proceso metodológico que permita alcanzar los objetivos.

En tercer lugar se encuentra el **desarrollo específico de la contribución** y en este punto se ejecutan todas las actividades para desarrollar los objetivos específicos y poder alcanzar esas pequeñas metas, las cuales componen todo el desarrollo del trabajo final de Máster.

Y en último lugar se encuentran las **conclusiones y trabajo a futuro** en donde se realiza la reflexión de todo el trabajo realizado, las lecciones aprendidas y las oportunidades que surgen de este trabajo.

2. Contexto y estado del arte

En la presentación de este trabajo, se introduce al lector en un panorama esencial: el contexto y estado del arte en torno a la metodología GitOps y su aplicación a través de diversas herramientas. La creciente complejidad de los entornos tecnológicos empresariales ha generado la necesidad de abordar los desafíos relacionados con la gestión, la implementación y el monitoreo de aplicaciones de manera más eficaz y ágil. En este contexto, la metodología GitOps ha emergido como un enfoque prometedor, basado en la sincronización continua entre las definiciones de infraestructura y la implementación real, todo ello respaldado por el control de versiones que ofrece Git. El análisis del estado del arte revela cómo diversas herramientas han abrazado y operacionalizado esta metodología, cada una con su propia perspectiva y capacidades únicas. Comprender este contexto resulta vital para apreciar el marco en el cual se desarrolla este trabajo, así como para trazar una línea clara entre los enfoques previos y las contribuciones innovadoras que se explorarán en las secciones venideras.

2.1. Contextualización y antecedentes

2.1.1. Equipo de desarrollo

El equipo de desarrollo suele estar formado por entre 3 a 9 profesionales que se encargan de desarrollar el producto, auto-organizándose y auto-gestionándose para conseguir entregar un incremento de software al final del ciclo de desarrollo. (Roche, s. f.).

2.1.2. Equipo de Operaciones

El departamento informático es un pilar fundamental en cualquier empresa, pues es la única manera de garantizar una infraestructura informática de alto rendimiento que mantenga en marcha todos los procesos necesarios.

¿Qué es el departamento de TI?

El departamento de informática de una empresa está conformado por un grupo de profesionales de TI que desarrollan, implementan y mantienen los sistemas a gran escala que hacen posibles las operaciones comerciales de las empresas modernas.

¿Cuál es la función del departamento de TI?

La tarea principal del departamento de TI es gestionar la infraestructura informática de una empresa y mantenerla en funcionamiento. Sin embargo, es importante tener en cuenta que este departamento puede tener un aspecto diferente en cada empresa. Además, los tipos de sistemas en los que trabajan pueden variar dependiendo de la cultura y las necesidades de la empresa, así como de la experiencia y los conocimientos de los miembros del equipo. Algunos equipos de TI son generalistas y trabajan con una amplia gama de sistemas y servicios. Otros, en cambio, se centran en tecnologías específicas (como redes o servicios web) o en un tipo concreto de sistema (apoyo a ventas, producción, logística, etc.). También hay equipos de TI que se centran por completo en la tecnología, mientras que otros están formados por especialistas en datos o analistas con profundos conocimientos de los procesos empresariales. En definitiva, estos equipos tienen éxito porque están formados por el grupo de personas adecuado para afrontar cualquier reto. (Freshworks, s. f.).

2.1.3. Ciclo de vida del desarrollo de software

El ciclo de vida de desarrollo de software (SDLC) es un proceso utilizado por la industria del software para diseñar, desarrollar y probar software de alta calidad. El SDLC tiene como objetivo producir un software de alta calidad que cumpla o supere las expectativas del cliente, llegue a su finalización dentro de los tiempos y costos estimados. (TutorialsPoint, s. f.)

2.1.4. DevOps

DevOps permite que los roles que antes estaban aislados (desarrollo, operaciones de TI, ingeniería de la calidad y seguridad) se coordinen y colaboren para producir productos mejores y más confiables. Al adoptar una cultura de DevOps junto con prácticas y herramientas de DevOps, los equipos adquieren la capacidad de responder mejor a las necesidades de los clientes, aumentar la confianza en las aplicaciones que crean y alcanzar los objetivos empresariales en menos tiempo. (DevOpsDays, s. f.).

2.1.5. Release management

La gestión de versiones es el proceso de gestión, planificación, programación y control de una compilación de software a través de diferentes etapas y entornos; incluye pruebas y despliegue de versiones de software. (Wikipedia, s. f.).

2.1.6. Control de versiones con Git

El control de versiones, también conocido como "control de código fuente", es la práctica de rastrear y gestionar los cambios en el código de software. Los sistemas de control de versiones son herramientas de software que ayudan a los equipos de software a gestionar los cambios en el código fuente a lo largo del tiempo. A medida que los entornos de desarrollo se aceleran, los sistemas de control de versiones ayudan a los equipos de software a trabajar de forma más rápida e inteligente. Son especialmente útiles para los equipos de DevOps, ya que les ayudan a reducir el tiempo de desarrollo y a aumentar las implementaciones exitosas.

El software de control de versiones realiza un seguimiento de todas las modificaciones en el código en un tipo especial de base de datos. Si se comete un error, los desarrolladores pueden ir hacia atrás en el tiempo y comparar las versiones anteriores del código para ayudar a resolver el error, al tiempo que se minimizan las interrupciones para todos los miembros del equipo.

Para casi todos los proyectos de software, el código fuente es como las joyas de la corona, un activo valioso cuyo valor debe protegerse. Para la mayoría de equipos de software, el código fuente es un repositorio del conocimiento de valor incalculable y de la comprensión sobre el dominio del problema que los desarrolladores han recopilado y perfeccionado con un esfuerzo cuidadoso. El control de versiones protege el código fuente tanto de las catástrofes como del deterioro casual de los errores humanos y las consecuencias accidentales.

Los desarrolladores de software que trabajan en equipos están escribiendo continuamente nuevo código fuente y cambiando el que ya existe. El código de un proyecto, una aplicación o un componente de software normalmente se organiza en una estructura de carpetas o "árbol de archivos". Un desarrollador del equipo podría estar trabajando en una nueva función mientras otro desarrollador soluciona un error no relacionado cambiando código. Cada desarrollador podría hacer sus cambios en varias partes del árbol de archivos.

El control de versiones ayuda a los equipos a resolver este tipo de problemas al realizar un seguimiento de todos los cambios individuales de cada colaborador y al contribuir a evitar que el trabajo concurrente entre en conflicto. Los cambios realizados en una parte del software pueden ser incompatibles con los que ha hecho otro desarrollador que está trabajando al mismo tiempo. Este problema debería detectarse y solucionarse de manera ordenada sin bloquear el trabajo del resto del equipo. Además, en todo el desarrollo de software, cualquier cambio puede introducir nuevos errores por sí mismo y el nuevo software no es fiable hasta que se prueba. De este modo, las pruebas y el desarrollo van de la mano hasta que está lista una nueva versión.

Un buen software de control de versiones soporta el flujo de trabajo preferido de un desarrollador sin imponer una forma determinada de trabajar. Idealmente, también funciona en cualquier plataforma, en vez de ordenar qué sistema operativo o cadena de herramientas deben utilizar los desarrolladores. Los sistemas de control de versiones excepcionales facilitan un flujo sencillo y continuo de cambios en el código en vez del mecanismo frustrante y burdo del bloqueo de archivos, que da luz verde a un desarrollador a expensas de bloquear el progreso de los demás.

Los equipos de software que no utilizan ninguna forma de control de versiones a menudo se encuentran con problemas como no saber qué cambios que se han hecho están disponibles

para los usuarios o la creación de cambios incompatibles entre dos partes no relacionadas que tienen que desvincularse y revisarse exhaustivamente. Si eres un desarrollador que nunca ha utilizado el control de versiones, puede que hayas añadido versiones a tus archivos, quizás con sufijos como "final" o "más reciente", y que después hayas tenido que enfrentarte con una nueva versión final. Quizás has convertido en comentarios bloques de código, porque quieres desactivar una determinada función sin eliminar el código, con el miedo de que pueda utilizarse más adelante. El control de versiones es una forma de solucionar estos problemas.

El software de control de versiones es una parte esencial del día a día de las prácticas profesionales del equipo de software moderno. Los desarrolladores de software individuales que están acostumbrados a trabajar con un sistema de control de versiones potente en sus equipos suelen reconocer el increíble valor que el control de versiones también les da incluso en los proyectos pequeños en los que trabajan solos. Una vez acostumbrados a las potentes ventajas de los sistemas de control de versiones, muchos desarrolladores no se plantearían trabajar sin ellos incluso para los proyectos que no son de software. (Atlassian, s. f.).

2.1.7. Integración continua

La integración continua (CI) es la práctica de automatizar la integración de los cambios de código de varios contribuidores en un único proyecto de software. Es una de las principales prácticas recomendadas de DevOps, que permite a los desarrolladores fusionar con frecuencia los cambios de código en un repositorio central donde luego se ejecutan las compilaciones y pruebas. Las herramientas automatizadas sirven para verificar que el nuevo código es correcto antes de la integración.

Un sistema de control de versiones del código fuente es el punto clave del proceso de CI. El sistema de control de versiones también se complementa con otras comprobaciones como las pruebas automatizadas de calidad del código, las herramientas de revisión de estilo de sintaxis y mucho más. (Rehkopf, s. f.).

2.1.8. Entrega continua

La entrega continua (CD) es una colección de muchas prácticas recomendadas de organización y metodología ágil. Con la CD, una organización se centra en la creación de un proceso de publicación de software sencillo y automatizado. La pieza central de este proceso de publicación es un ciclo de *feedback* iterativo. El ciclo de *feedback* gira en torno a la entrega de software al usuario final lo más rápido posible, aprendiendo de la experiencia práctica e incorporando ese *feedback* en la siguiente publicación. (Rehkopf, s. f.).

2.1.9. Despliegue continuo

El despliegue continuo constituye un enfoque en el que los equipos publican productos de calidad con frecuencia y de forma predecible desde repositorios de código fuente hasta la producción de forma automatizada. (Atlassian, s. f.).

2.1.10. Entrega continua vs despliegue continuo

Diferenciar la implementación de la entrega continua puede ser complicado, por sus nombres. Ambas se abrevian CD, por sus siglas en inglés, y tienen responsabilidades muy similares. La entrega es la precursora de la implementación. En la entrega, hay un paso final de aprobación manual antes de la publicación en producción.

Se describe el siguiente ejercicio de mnemotecnica para poder recordar la diferencia entre los dos. Imaginemos que va a llegar un paquete de la tienda online favorita. Mientras se espera, se coordina con un servicio de mensajería. Esta es la fase de entrega. Una vez que el paquete ha llegado, se abre y comprueba que todo está como se esperaba. Si no es así, es posible rechazarlo y devolverlo. Si el paquete es correcto, se puede "implementar" y usar el nuevo artículo.

En la fase de entrega, los desarrolladores revisan y fusionan los cambios de código que luego se empaquetan en un artefacto. A continuación, este paquete se mueve a un entorno de producción donde espera la aprobación para abrirse e implementarse. En la fase de

implementación, el paquete se abre y se revisa con un sistema de comprobaciones automatizadas. Si las comprobaciones fallan, el paquete se rechaza.

Una vez pasadas las comprobaciones, el paquete se implementa automáticamente en producción. La implementación continua es la canalización de implementación de software automatizada e integral, de extremo a extremo. (Pittet, s. f.).

2.1.11. Computación en la nube

El concepto descrito por Mell y Grance, (2011) del Instituto Nacional de Estándares y Tecnología (NIST por sus siglas en inglés, National Institute of Standards and Technology), se ajusta a la presente investigación, quienes indican que la computación en la nube (*Cloud Computing*) es un modelo para la habilitación ubicua, conveniente, de acceso a la red bajo demanda, a un conjunto compartido de recursos informáticos configurables (por ejemplo: redes, servidores, almacenamiento, aplicaciones y servicios) que se pueden aprovisionar y lanzar rápidamente con un mínimo esfuerzo de gestión o interacción del proveedor de servicios. Asimismo, dicho instituto define los tres modelos de servicio de la computación en la nube, a saber:

Software as a Service - SaaS: capacidad de proporcionar aplicaciones que se ejecutan en infraestructura en la nube, accesibles desde varios dispositivos a través de una interfaz ligera (por ejemplo, navegador web). El cliente no administra o controla la infraestructura en la nube subyacente (redes, servidores, sistemas operativos, almacenamiento o capacidades de la aplicación individual) (A. Q. Ali et al., 2019; Mell y Grance, 2011).

Platform as a Service - PaaS: capacidad de proporcionar infraestructura en la nube para desarrollar aplicaciones o adquirirlas. El cliente no administra o controla la infraestructura en la nube subyacente (redes, servidores, sistemas operativos o almacenamiento), pero tiene control sobre las aplicaciones desarrolladas y los posibles ajustes de configuración para el ambiente o entorno de alojamiento de aplicaciones (Mell y Grance, 2011; van Rossem et al., 2018).

Infrastructure as a Service - IaaS: capacidad de proporcionar aprovisionamiento de procesamiento, almacenamiento, redes y otros recursos de computación fundamentales

donde el cliente puede desarrollar y ejecutar software arbitrario (sistemas operativos y aplicaciones). El cliente no administra o controla la infraestructura en la nube subyacente, pero tiene control sobre sistemas operativos, almacenamiento y aplicaciones desarrolladas, y posiblemente tiene control limitado en componentes de red seleccionados (por ejemplo, firewalls) (Mell y Grance, 2011).

2.1.12. Contenedores

Un contenedor es una unidad estándar de software que empaqueta el código y todas sus dependencias para que la aplicación se ejecute de manera rápida y confiable de un entorno informático a otro. Una imagen de contenedor de Docker es un paquete de software ligero, independiente y ejecutable que incluye todo lo necesario para ejecutar una aplicación: código, tiempo de ejecución, herramientas del sistema, bibliotecas del sistema y configuraciones.

Las imágenes de contenedores se convierten en contenedores en tiempo de ejecución y, en el caso de los contenedores de Docker, las imágenes se convierten en contenedores cuando se ejecutan en Docker Engine. Disponible para aplicaciones basadas en Linux y Windows, el software en contenedores siempre se ejecutará de la misma manera, independientemente de la infraestructura. Los contenedores aíslan el software de su entorno y garantizan que funcione de manera uniforme a pesar de las diferencias, por ejemplo, entre el desarrollo y la puesta en escena. Docker. (s. f.).

2.1.13. Kubernetes

Kubernetes es una plataforma portable y extensible de código abierto para administrar cargas de trabajo y servicios. Kubernetes facilita la automatización y la configuración declarativa. Tiene un ecosistema grande y en rápido crecimiento. El soporte, las herramientas y los servicios para Kubernetes están ampliamente disponibles.

Google liberó el proyecto Kubernetes en el año 2014. Kubernetes se basa en la experiencia de Google corriendo aplicaciones en producción a gran escala por década y media, junto a las mejores ideas y prácticas de la comunidad. (Kubernetes, 2022).

2.1.14. GitOps

GitOps es un marco centrado en Git para la implementación de aplicaciones que simplifica enormemente el desarrollo de software en entornos nativos de la nube. Utiliza las mejores prácticas de DevOps, como el control de versiones y CI/CD, y las aplica a la automatización de la infraestructura. (Weaveworks, s. f.).

2.2. Antecedentes

En el examen profundo de la evolución organizativa y sus vivencias, surge una trama de antecedentes que ha esculpido la trayectoria del uso de GitOps en el contexto empresarial. Comprender estos antecedentes emerge como una etapa esencial para adentrarse en cómo las organizaciones han afrontado los cambios tecnológicos y las demandas del entorno empresarial en evolución. A través de esta exploración, se resalta la necesidad de analizar estas experiencias pasadas para extraer aprendizajes valiosos que puedan guiar una implementación informada y exitosa de GitOps en el panorama actual, en constante cambio y repleto de desafíos.

En el año 2016, uno de los ingenieros del equipo de *Weaveworks* declaró que estaba a punto de introducir un cambio en el sistema que, si no funcionaba según lo previsto, podría acabar con todo el sistema. No funcionó y el sistema se eliminó, pero el equipo de *Weaveworks* pudo restablecer el sistema en aproximadamente cuarenta minutos porque se describió completamente en varios archivos de configuración de Git. El sistema incluía no solo el clúster, sino también la aplicación, el monitoreo y otras piezas. Cada vez que se realiza un cambio en el sistema, primero se confirma y luego se permite que se propague automáticamente a producción.

El equipo de *Weaveworks* estaba practicando algo así como una versión 0.1 de GitOps incluso antes de que existiera el término. Tenían reglas simples como no permitir que sucediera un cambio, si era un cambio permanente. Cada cambio tenía que confirmarse en el sistema de registro para el estado deseado que se almacenaba en una instancia de GitHub resistente, distribuida, autenticada, no repudiable y segura.

Muchas de estas prácticas fueron presentadas al equipo de *Weaveworks* por empleados que trajeron consigo estos aprendizajes sobre cómo construir sistemas resilientes usando principios declarativos.

En el año 2017, llegó el nacimiento de GitOps, después del incidente, complacidos con su progreso, el equipo se reunió e hizo una lista de estos principios por los que operaban su sistema *Kubernetes*. Resumieron los principios en los pocos más importantes. Fue durante una de estas discusiones que Alexis Richardson, CEO de *Weaveworks*, se dio cuenta de que una palabra que describe todo esto era 'GitOps'. La idea fundamental detrás de GitOps era hacer que las operaciones fueran automáticas para todo el sistema en función de un modelo del sistema que vivía fuera del sistema, y Git fue donde habían elegido colocar ese modelo.

Una tarde, mientras discutía este patrón de administrar todo el sistema usando Git, a Alexis se le ocurrió la palabra 'GitOps'. Richardson reprodujo la frase de un amigo, James Governor, fundador de *RedMonk*, quien dijo que posiblemente era la palabra más fea que había escuchado en mucho tiempo y que ahora no podía dejar de escucharla. Al querer un término que fuera fácil de pronunciar y que se quedara grabado en la memoria de alguien una vez que lo escuchara, Richardson supo que había encontrado lo que estaba buscando.

Richardson pronto presentó los cuatro principios de GitOps en charlas. Estos son los cuatro principios:

1. Todo el sistema se describe declarativamente
2. El estado canónico del sistema deseado se versiona en Git
3. Cambios aprobados que se pueden aplicar automáticamente al sistema
4. Agentes de software para garantizar la corrección y alertar sobre divergencias

Puede leer más sobre estos principios en la Guía de GitOps.
<https://www.weave.works/technologies/gitops/>

En el año 2018, llega la adopción en toda la industria, GitOps pronto fue adoptado por los principales proveedores de nube AWS, Azure y Google Cloud.

Y en el año 2019, *Weaveworks* lanzó *Flux* y *Flagger*, ambas herramientas de código abierto que permiten la entrega progresiva utilizando el modelo GitOps. Estas herramientas hicieron que GitOps fuera más operativo para las organizaciones. Tanto *Flux* como *Flagger* se convirtieron en proyectos *sandbox* de *CNCF*. Este fue el comienzo del movimiento GitOps que pasó de ser un pequeño grupo de primeros usuarios a una comunidad de colaboradores de todos los rincones del ecosistema nativo de la nube.

La revisión de diversos artículos en línea revela que un número creciente de empresas ha adoptado GitOps como enfoque fundamental para gestionar sus operaciones de TI. A continuación se mencionan algunos de ellos:

En el artículo titulado ***Intential to Showcase Infrastructure as Code for Cloud Network Automation at ONUG Fall 2019*** (Intential, s. f.), presenta una visión sólida y clara de la participación de Intential en el evento ONUG Fall 2019. El enfoque en la automatización de redes inteligentes y la implementación de infraestructura como código se destaca como un avance significativo en la gestión de la configuración de redes en entornos en la nube. La mención de utilizar la estrategia de GitOps para administrar la configuración de red es especialmente relevante en un contexto tecnológico en constante cambio. Además, la cita del copresidente y cofundador de ONUG, Nick Lippis, refuerza la importancia y la pertinencia de las soluciones de automatización de red en entornos complejos y multinube. El artículo proporciona una visión prometedora sobre cómo *Intential* y su enfoque multidominio están en línea con las necesidades actuales de orquestación y automatización en este entorno tecnológico en evolución.

Schalm (2020), en el artículo titulado ***Instana Becomes First APM Solution with GitOps Enabled Agent Management***, presenta un avance significativo en el ámbito de la gestión del rendimiento de aplicaciones (APM) con un enfoque innovador. La disponibilidad de la primera solución de APM con un agente habilitado para GitOps ofrece a las organizaciones una forma más eficiente y automatizada de administrar la configuración de los agentes en entornos de producción. La cita del director técnico de *Instana*, Chris Farrell, destaca la importancia de esta capacidad única para brindar un mayor control a los usuarios y reducir la carga manual, lo que optimiza la administración de la configuración del agente. Esta solución aborda una necesidad crítica en el ámbito de las aplicaciones nativas de la nube y ofrece una manera efectiva de mantener la consistencia y el control en despliegues a gran escala.

El autor (Yates, 2021), en el artículo titulado ***GitOps has been ‘controversial’ — but the truth is your organization needs it***, el artículo ofrece una analogía interesante entre el mito de utilizar solo el 10% del cerebro y la forma en que a menudo se utiliza Git en el desarrollo de software. El artículo pone de relieve la importancia de ir más allá de ver Git únicamente como una herramienta para alojar y administrar código. Al presentar GitOps como un enfoque que aprovecha al máximo el potencial de Git, se hace hincapié en su papel central en la gestión e implementación de la infraestructura. Esta comparación capta la esencia del artículo, resaltando cómo GitOps puede liberar un nivel significativamente mayor de eficiencia y coherencia en el despliegue de infraestructura para las organizaciones. La metáfora y la explicación proporcionan una perspectiva perspicaz sobre el valor más amplio que GitOps puede aportar, convirtiendo Git en una herramienta más poderosa en la gestión de la tecnología.

En el sitio oficial de la empresa *Red Hat*, en el apartado titulado ***Red Hat Makes DevOps a Reality with OpenShift GitOps and OpenShift Pipelines*** (Raleigh, 2021), el artículo destaca una importante evolución en las capacidades de *Red Hat OpenShift* y su enfoque en mejorar la integración entre los equipos de desarrollo y operaciones. La introducción de *OpenShift GitOps* y *OpenShift Pipelines* representa un paso hacia adelante en la alineación de los equipos, en línea con la filosofía de DevOps. El artículo enfatiza acertadamente cómo estas

nuevas características ayudan a resolver los desafíos que a menudo impiden la adopción completa de DevOps, al proporcionar herramientas que permiten una colaboración temprana y continua entre los equipos de desarrollo y operaciones. La mención de la mejora en seguridad, previsibilidad y visibilidad a lo largo del ciclo de vida de la aplicación resalta cómo estas adiciones pueden ofrecer beneficios significativos en la implementación y administración de aplicaciones en la nube híbrida. En general, el artículo destaca cómo *Red Hat* está respondiendo a las necesidades cambiantes de la industria al proporcionar soluciones que abordan los desafíos comunes en la colaboración entre equipos.

En el sitio oficial de la empresa *Platform9*, en el apartado titulado ***Platform9 Announces SaaS GitOps Engine to Simplify Multicloud Operations and Governance With Latest Kubernetes Release*** (Platform9, 2021), artículo refleja una evolución significativa en el ámbito de la administración de clústeres *Kubernetes* y la gestión multinube. El anuncio de nuevas características por parte de *Platform9* se destaca por su enfoque en simplificar la complejidad operativa en torno a la administración de múltiples clústeres y despliegues de *Kubernetes*. Las características mencionadas, como el GitOps Profile Engine para la gobernanza basada en plantillas, mejoras de IDE y estrategias de actualización especificadas por el usuario, abordan cuestiones fundamentales en términos de gobernanza, seguridad y facilidad de uso. La cita de Madhura Maskasky, co-fundadora y VP de Producto de *Platform9*, resalta cómo estas nuevas funciones están diseñadas para empoderar a los desarrolladores al brindar acceso instantáneo y simplificar la administración desde un único panel. Además, la capacidad de crear y gestionar clústeres de *Kubernetes* en una variedad de entornos, desde la capa bare metal hasta AWS y Azure, subraya la versatilidad y la amplitud de la plataforma. En general, el artículo presenta un avance significativo en la gestión de *Kubernetes* y la administración multinube, con un enfoque claro en la eficiencia y la experiencia del usuario.

El autor (Kornstädt, 2021), en el artículo titulado ***Deutsche Telekom rolls out 5G on Kubernetes with Weaveworks***, aborda de manera efectiva la importancia del enfoque GitOps en el contexto de la implementación de servicios 5G nativos de la nube por parte de

Deutsche Telekom. El artículo destaca cómo, aunque la infraestructura de *Kubernetes* es un componente fundamental de los planes de la empresa de telecomunicaciones, GitOps desempeña un papel igualmente crucial en la automatización de la infraestructura de TI. La explicación sobre GitOps y su uso en la sincronización constante del estado de los recursos de producción con un estado deseado expresado en código proporciona claridad sobre este enfoque. La distinción entre ejecutar una aplicación basada en *Kubernetes* y ser verdaderamente nativa de la nube, subrayada por la cita de Michal Sewera, ofrece una visión profunda sobre los estándares y objetivos que *Deutsche Telekom* persigue en su proyecto 5G nativo de la nube. El artículo presenta una interconexión inteligente entre las capacidades de *Kubernetes*, GitOps y la visión de la tecnología 5G, demostrando cómo estas tecnologías convergen para permitir aplicaciones de próxima generación que sean ágiles, resistentes y adaptables a cambios constantes.

Teniendo en cuenta el panel de discusión con *Weaveworks*, *HashiCorp*, *Red Hat* y *GitLab*, titulado ***GitOps: The Future of Infrastructure Automation*** (Davis et al., 2021), en donde se proporciona una perspectiva clara y coherente sobre la importancia de la automatización de la infraestructura moderna y cómo GitOps se alinea con las prácticas de DevOps para abordar esta necesidad. El panel de discusión comunica eficazmente cómo GitOps toma las buenas prácticas de DevOps y las aplica específicamente a la automatización de la infraestructura. La descripción de GitOps como un conjunto de prácticas para manipular configuraciones de infraestructura y aplicaciones mediante Git es concisa y precisa, proporcionando una comprensión sólida de su enfoque. La mención de Git como una única fuente de verdad para la infraestructura y las aplicaciones declarativas refuerza la importancia de la consistencia y la trazabilidad en el proceso de automatización. En general, el panel ofrece una introducción esclarecedora a GitOps y cómo aborda los desafíos de la automatización de la infraestructura moderna de manera coherente con los principios de DevOps.

El autor (Ennis, 2022), en el artículo titulado ***Now Generally Available: Harness GitOps-as-a-Service Delivers Scalability, Control, and Security to Enterprises***, resalta de manera efectiva el valor añadido que *Harness GitOps* aporta a los flujos de trabajo de

entrega continua (CD) empresarial. La descripción de cómo *Harness GitOps* brinda una solución declarativa que incorpora seguridad empresarial, gobernanza y escalabilidad, alineándose con las robustas canalizaciones visuales de CD de *Harness*, resalta su enfoque en la calidad y el cumplimiento. La mención de la rápida implementación y la ligereza operativa de GitOps, así como su base en el proyecto Argo CD, contribuye a la credibilidad y eficacia de la solución propuesta. El artículo explica cómo aprovechar GitOps no solo amplía las capacidades de entrega de software, sino que también capacita a los desarrolladores con herramientas poderosas para una variedad de funciones, desde integración continua hasta control de costos y seguridad. La mención del gobierno basado en *Open Policy Agent (OPA)* enfatiza la robustez y el compromiso con los requisitos de seguridad en un contexto empresarial. En general, el artículo proporciona una visión clara y convincente sobre cómo *Harness GitOps* contribuye a la mejora de los flujos de trabajo de CD empresarial y a la adopción efectiva de GitOps.

2.3. Trabajos relacionados

La búsqueda de los trabajos relacionados, ha permitido seleccionar algunos que tratan la metodología GitOps, después de la contextualización de cada trabajo se clasifican los trabajos por ciertos temas, a continuación se detalla en profundidad. a) microservicios, b) CI/CD y Plataformas IaaS, c) Cluster, d) Orquestación

Microservicios

Con relación a los microservicios se da a conocer el resultado del trabajo de grado de maestría titulado ***Metodología GitOps para despliegue de aplicaciones basadas en microservicios*** del autor (Maderuelo, 2021), dando a conocer una descripción clara de la evolución desde la arquitectura monolítica hacia el modelo de microservicios y cómo GitOps surge como un enfoque contemporáneo para integrar estas tecnologías emergentes en el proceso de desarrollo y despliegue de software. La propuesta de desplegar un clúster de *Kubernetes* en un entorno virtual para demostrar esta metodología y servir como laboratorio experimental es un enfoque práctico y relevante. El trabajo de grado presenta una

comprensión profunda del uso de técnicas DevOps, como la gestión de infraestructura como código y la automatización, para mejorar la eficiencia y la resiliencia en el despliegue de aplicaciones e infraestructuras. Además, la mención de implementar la solución en un entorno productivo de una nube pública (*Amazon Web Services - EKS*) agrega un componente realista y aplicable al proyecto. En general, destaca la relevancia de las tecnologías modernas, como GitOps y Kubernetes, y su aplicación en un contexto práctico, proporcionando una visión prometedora sobre cómo estas metodologías pueden mejorar los procesos de desarrollo y despliegue en la actualidad.

CI/CD y Plataformas IaaS

Con relación a CI/CD y Plataformas IaaS se da a conocer el resultado del trabajo de grado titulado ***Diseño, desarrollo e implementación de una arquitectura GitOps (Git y Kubernetes) que integre infraestructura basada en código desplegadas en plataformas IaaS (Infraestructura como Servicio)*** de la autora Chacón Cabrera (2022), dando a conocer un proyecto integral y con un enfoque práctico que aborda desafíos relevantes en el desarrollo y despliegue de aplicaciones e infraestructuras. Identifica claramente los problemas que enfrenta la empresa en términos de complejidad y tiempo en el proceso de comercialización de productos, y cómo la metodología GitOps podría abordar estas limitaciones al mejorar la entrega, la resolución de problemas, la recuperación y la eficiencia.

Los autores Beetz y Harrer (2022), en el artículo titulado ***GitOps: The Evolution of DevOps?***, presentan un enfoque valioso al abordar la relación entre DevOps y GitOps, y su evolución como conceptos en el ámbito de la ingeniería de software. La introducción del artículo establece claramente el propósito de analizar si GitOps realmente representa la evolución de DevOps, lo que indica una intención crítica y esclarecedora. El hecho de reconocer la falta de definiciones estándar para ambos conceptos demuestra una comprensión de la ambigüedad inherente en este campo. El enfoque de sintetizar definiciones para DevOps y GitOps y luego comparar sus elementos clave es un método sólido para evaluar y contrastar sus características fundamentales. La comparación de elementos clave permitirá a los lectores

comprender las similitudes, diferencias y posibles áreas de mejora que GitOps podría ofrecer en comparación con DevOps.

Los autores Gupta, Bhatia, Memoria y Manani (2022), en el artículo titulado ***Prevalence of GitOps, DevOps in Fast CI/CD Cycles***, presentan un enfoque contemporáneo y relevante al abordar los desafíos y las necesidades en la entrega continua y la integración continua (CI/CD) en el desarrollo de software. La descripción de cómo las aplicaciones basadas en contenedores abordan problemas complejos en CI/CD, como la portabilidad y el control de versiones, demuestra una comprensión sólida de las implicaciones prácticas de esta tecnología. La introducción del concepto GitOps como un enfoque ágil, confiable, rápido y eficiente para mejorar los niveles de rendimiento en entornos cloud-native es relevante en el contexto actual de desarrollo de software. La inclusión de un nuevo punto de interés como GitOps en el proceso de desarrollo agrega valor a la discusión y sugiere una exploración más profunda. La identificación del objetivo principal del artículo de comprender el proceso de Kubernetes GitOps en las operaciones, acceder a los beneficios de implementar GitOps en el entorno de *Kubernetes* e implementar Kubernetes GitOps en AWS es clara y concisa, lo que ayuda a los lectores a entender la dirección de la investigación.

Los autores López-Viana, Díaz y Pérez (2022), en el artículo titulado ***Continuous Deployment in IoT Edge Computing : A GitOps implementation***, abordan de manera efectiva el paradigma emergente del Edge Computing y su relación con la metodología GitOps en el contexto de las soluciones IoT EC. La introducción del concepto de Edge Computing y su comparación con la computación en la nube proporciona un contexto claro para el lector. El artículo presenta de manera coherente los objetivos de los profesionales de Edge Computing al buscar replicar los beneficios de la computación en la nube, así como las mejores prácticas de desarrollo y operación nativas de la nube utilizando GitOps. Esta conexión entre Edge Computing, GitOps y las aplicaciones distribuidas es pertinente y muestra una comprensión sólida de las tendencias actuales en el ámbito tecnológico. La propuesta de una prueba de concepto basada en herramientas de la *Cloud Native Computing Foundation* (CNCF) para evaluar la viabilidad de aplicar GitOps en soluciones IoT EC agrega un componente práctico al

artículo. La identificación de los inconvenientes y desafíos encontrados en la implementación de GitOps en el contexto de IoT EC, como el aprovisionamiento automático de la infraestructura física y las limitaciones en los dispositivos de borde, también contribuye a la comprensión de la aplicación práctica de estos conceptos.

Los autores Garg et al. (2021), en el artículo titulado ***On Continuous Integration / Continuous Delivery for Automated Deployment of Machine Learning Models using MLOps***, abordan un tema relevante y contemporáneo en el ámbito del aprendizaje automático y la implementación de modelos, destacando la conexión entre DevOps y MLOps. La analogía establecida entre el proceso de desarrollo de software tradicional y la implementación de modelos en el aprendizaje automático es una introducción efectiva para los lectores. La identificación de los desafíos al usar canalizaciones de CI/CD en aplicaciones que incorporan componentes de MLOps es relevante y muestra una comprensión clara de las complejidades de esta área. La discusión sobre cómo los pioneros en el campo resuelven estos problemas utilizando herramientas exclusivas y las proporcionadas por proveedores de la nube es esclarecedora y agrega valor al contenido. El enfoque en brindar una perspectiva más amplia sobre el ciclo de vida del aprendizaje automático y las diferencias entre DevOps y MLOps es apropiado y útil para los lectores que buscan comprender las particularidades de la implementación de modelos. La discusión sobre herramientas y técnicas para ejecutar canalizaciones de CI/CD de marcos de aprendizaje automático en el enfoque MLOps, así como las implementaciones basadas en GitOps, agrega un componente práctico al artículo.

Cluster

Los autores Ramadoni, Utami y Fatta (2021), en el artículo titulado ***Analysis on the Use of Declarative and Pull-based Deployment Models on GitOps Using Argo CD***, plantean una investigación interesante que busca abordar problemas comunes en la implementación de metodologías DevOps mediante la aplicación del método GitOps. La identificación de problemas de seguridad y la ineficacia del proceso de reversión en la implementación de aplicaciones en una plataforma de aplicaciones subraya la relevancia de abordar estas limitaciones.

La elección de utilizar el enfoque declarativo y basado en extracción en lugar de los modelos basados en inserción es un enfoque metodológico clave que podría proporcionar una visión nueva y efectiva para la implementación de CI/CD utilizando GitOps. La selección de la herramienta Argo CD como operador y *Kubernetes* como plataforma para implementar aplicaciones basadas en contenedores es una decisión coherente con los objetivos de la investigación y contribuye a la validez y aplicabilidad de los resultados.

El objetivo de proporcionar recomendaciones a empresas, instituciones y personas que deseen adoptar o mejorar la implementación de DevOps utilizando GitOps es una motivación pragmática y aplicada para la investigación. Esto sugiere que los resultados pueden ser de interés práctico y proporcionar orientación valiosa para aquellos que buscan mejorar sus prácticas de desarrollo y despliegue.

El autor Báez (2021), en el artículo de revista titulado ***Repositorios git y orquestadores de contenedores kubernetes en la optimización de la gestión empresarial***, presenta un enfoque claro y aplicado para analizar la eficiencia de los repositorios Git y los orquestadores de contenedores *Kubernetes* en empresas. La elección de la metodología de revisión sistemática y descriptiva es apropiada para el objetivo de la investigación, ya que permite un análisis exhaustivo de la literatura existente en el período de tiempo especificado. La descripción de la búsqueda en bases de datos secundarios y el rango de años seleccionado (2017-2021) es una estrategia metodológica que proporciona un marco temporal relevante para la investigación. La mención de que los resultados constatan que la implementación de plataformas y entrada de información de parámetros del clúster de *Kubernetes* y Git se conectan con un centro de ejecución y emiten actividades asincrónicas es una observación interesante que sugiere la integración y la interacción de estos elementos en la práctica empresarial. Las conclusiones del estudio, que enfatizan la validación completa mediante el despliegue de procesos organizacionales y la optimización de servicios y aplicaciones en la nube, refuerzan la aplicabilidad de los resultados en un contexto real. Además, la mención de que estas mejoras pueden lograrse sin grandes inversiones financieras agrega un aspecto pragmático y atractivo al estudio. En general, el resumen presenta una investigación sólida y relevante que aborda la eficiencia de los repositorios Git y los orquestadores de

contenedores *Kubernetes* en empresas. La elección de la metodología y la aplicabilidad de las conclusiones sugieren que el artículo puede proporcionar información valiosa para profesionales y organizaciones que buscan mejorar sus procesos de gestión y despliegue en entornos empresariales.

Orquestación

Los autores Leiter et al. (2023) en el artículo titulado ***GitOps and Kubernetes Operator-based Network Function Configuration***, plantean una investigación interesante que explora cómo el enfoque nativo de la nube impacta en la gestión y automatización de la red, particularmente en el contexto de *Kubernetes* como estándar para ejecutar funciones de red en contenedores. El artículo presenta un enfoque específico al utilizar los operadores de *Kubernetes* para la administración de configuración basada en *Netconf* y cómo esto se alinea con los principios de GitOps. La descripción de cómo los operadores de *Kubernetes* y el patrón de operador pueden automatizar la administración de redes y el trabajo manual es relevante y demuestra una comprensión sólida de cómo aprovechar las capacidades de *Kubernetes* en el contexto de la gestión de la red.

2.4. Conclusiones del estado del arte

Dentro de la experiencia obtenida como ingeniero DevOps, se han encontrado varios retos en el momento de desplegar infraestructura, uno de los más grandes es la integridad, luego de que se crea o se despliega un contenedor, las personas del equipo pueden ingresar a este e instalarle el software que requieran, luego de que la funcionalidad vaya cambiando y no sea suficiente el software o las librerías que ya se encuentran instaladas. Entonces la infraestructura desplegada inicialmente, no concuerda con la que se encuentra en este momento en la organización una vez haya transcurrido el tiempo; además puede ocurrir un fallo si la persona que está instalando los requisitos necesarios para continuar la operación no tiene el conocimiento suficiente para manipular el contenedor.

Otra situación como ingeniero DevOps, es encontrar contenedores que han sido desplegados a través de comandos y no de manifiestos YAML; que significa esto que si el contenedor se

llega a detener o fallar por algún motivo ese aplicativo queda sin funcionamiento y en muchas ocasiones ese contenedor lleva mucho tiempo prestando la funcionalidad a la organización, esto es una mala práctica, lo ideal sería desplegar contenedores a través de manifiestos YAML que permitan recrear su estado. Esto es debido a la inexperiencia con las nuevas tecnologías que pueda tener la persona encargada de la infraestructura.

Teniendo en cuenta las situaciones anteriormente mencionadas, se analizan diferentes retos y oportunidades de mejora. Uno de los retos es garantizar el inventario de la infraestructura, a que hace referencia esto, a que toda la infraestructura que se tenga contemplada para una solución concuerde con la infraestructura desplegada, puede ocurrir el caso de que por solventar una falla se desplieguen más aplicaciones que no se tengan planeadas y en este caso ya se empieza a generar discrepancias entre lo que se encuentra en la arquitectura y lo que realmente existe en los servidores.

Otro desafío importante en la administración de las aplicaciones e infraestructura, es el acceso, cómo se puede garantizar que cierto grupo de usuarios tengan acceso a los recursos que necesitan, sin tener acceso a otros que no están relacionados con estos; por ejemplo si un desarrollador modifica o despliega un contenedor que no le corresponde, esto cómo puede afectar a la organización, se pueden aumentar los costos, ya que modificó una funcionalidad que se encontraba bien, y ahora no funciona correctamente.

Al parecer, las organizaciones necesitan un departamento adicional o un grupo de individuos diferentes a los desarrolladores y administradores de infraestructura que puedan garantizar el acceso adecuado a los recursos, el uso adecuado de los recursos y la integridad de estos. Es posible que las organizaciones con mayor madurez puedan implementar este nuevo equipo de control de accesos o gestión de la configuración; pero las empresas pequeñas y medianas no podrán llevar a cabo las buenas prácticas para poder dar una respuesta a estos desafíos.

Es por este motivo que se realizó la búsqueda de los trabajos relacionados para poder comparar si estos trataban estos desafíos y cuál era la mejor forma de solucionar estos, dando como respuesta la metodología GitOps.

Dentro de la revisión sistemática o SMS de la literatura que se realizó se obtuvo como resultado la falta de proliferación de estudios formales alrededor de la metodología GitOps

pocos trabajos académicos tratan la problemática descrita anteriormente y esto condujo a considerar la documentación/literatura gris de los fabricantes de las herramientas para responder las preguntas planteadas.

La metodología GitOps permite abarcar muchas de las problemáticas o situaciones anteriormente expuestas y esta es una gran motivación para desarrollar este trabajo. Exponer o mostrar cómo el comportamiento de la metodología GitOps permite establecer una serie de prácticas que permiten darle un mejor manejo a las situaciones o experiencias anteriormente mencionadas.

Si es cierto, que se encontraron trabajos que mostraban el uso de la herramienta Argo CD como implementación para la metodología GitOps, no se encontraron trabajos que mencionan las distintas herramientas que usan GitOps y tampoco se encontró alguna referencia en donde se mencionan las ventajas y desventajas de estas otras implementaciones, esto ya es una gran motivación para investigar y mostrar las diferentes implementaciones que se encuentran en el mercado de la metodología GitOps.

Además de mostrar otras herramientas que implementen GitOps, se desarrollara una guía, un manual o un texto, que permita a los lectores instalar y configurar la herramienta de GitOps, esto con el fin de disminuir la curva de aprendizaje que puede tener un usuario en el momento de empezar con una nueva tecnología. Por otra parte, a la falta de conocimiento de los profesionales y partiendo del supuesto de que no todas las empresas cumplen con los prerequisites para implementar ciertas tecnologías, es necesario ilustrar las diferentes alternativas tecnológicas para poder solucionar algunos de los retos a los que se enfrentan al administrar sus aplicaciones e infraestructura.

Esta guía permitirá a las empresas alinearse de acuerdo a sus necesidades y escoger la herramienta que mejor se adapte a ellas.

3. Objetivos y metodología de trabajo

3.1. Objetivo general

Construir y validar una guía de despliegue para una herramienta tecnológica que se ajuste a la metodología GitOps.

3.2. Objetivos específicos

- Caracterizar herramientas que implementan la metodología GitOps tomando como referencia las etapas del ciclo de vida DevOps.
- Seleccionar la herramienta más relevante para la metodología GitOps, teniendo en cuenta aspectos como el desarrollo, documentación y tiempo de vida.
- Construir una guía de despliegue de acuerdo a la metodología GitOps para realizar la implementación tecnológica seleccionada.
- Evaluar la efectividad de la guía de despliegue a través de la implementación de un prototipo funcional de la herramienta GitOps seleccionada.

3.3. Metodología del trabajo

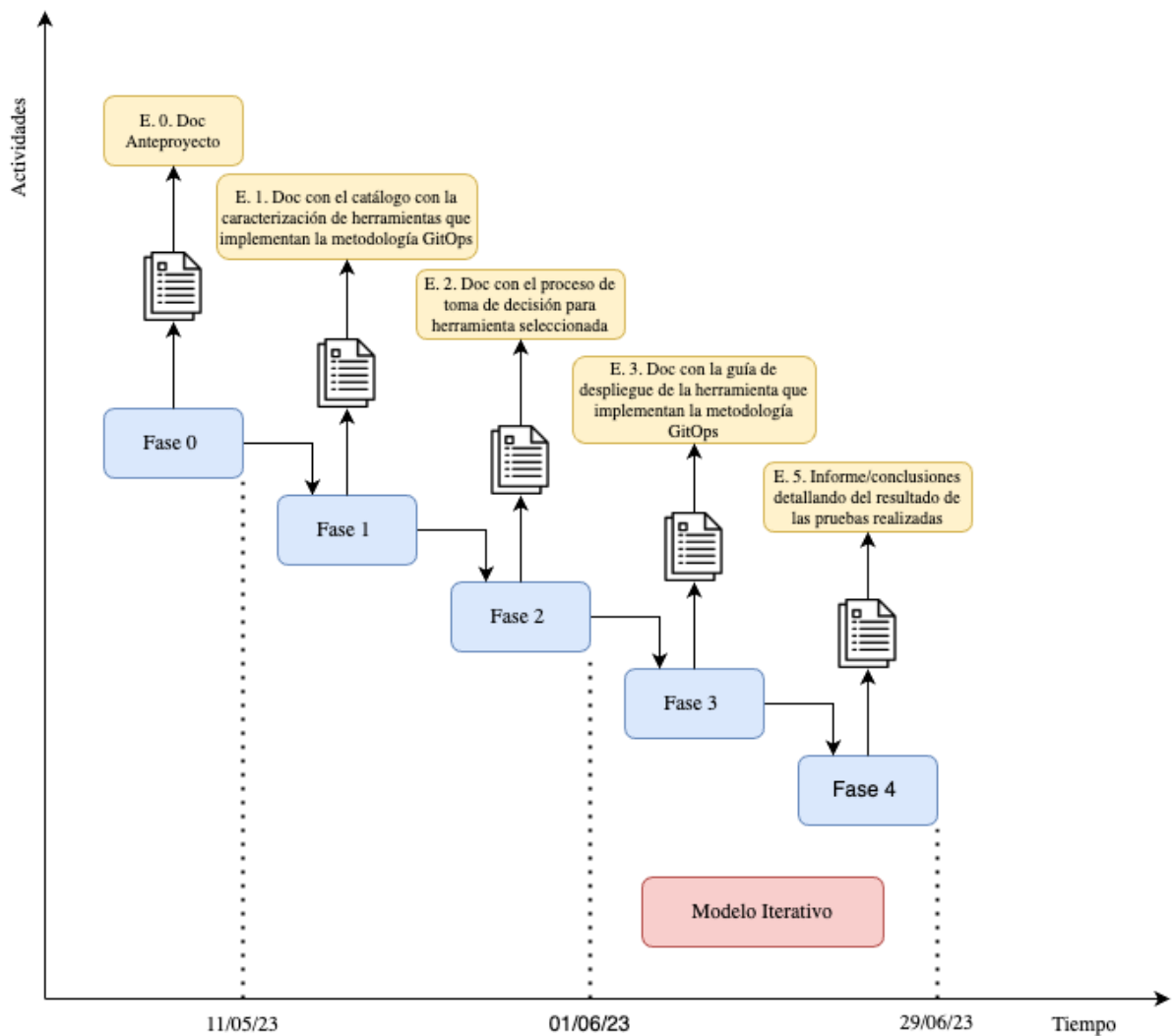
En el presente trabajo de fin de Máster, se va a ejecutar bajo un **modelo híbrido** el cual contempla aspectos de dos modelos del ciclo de vida del desarrollo de software ó *software development life cycle* (SDLC); el primero es el **modelo de ciclo de vida secuencial lineal** del cual se toma la ejecución del proyecto el cual debe desarrollarse en fases, estas se deben completar antes de que pueda comenzar la siguiente fase y no puede ocurrir una superposición en las fases.

La fase inicial corresponde a la construcción del documento del anteproyecto esta fase se ha denominado la fase 0, las siguientes fases se enfocan en cumplir los objetivos establecidos por medio de las actividades que se han planificado. Cada fase tiene como objetivo entregar un artefacto o entregable necesario para continuar con la fase posterior y poder llegar al

objetivo que es construir “construir y validar una guía de despliegue para una herramienta tecnológica que se ajuste a la metodología GitOps”.

El segundo modelo utilizado en el proyecto es el **SDLC - Modelo iterativo**, el cual nos va a permitir evaluar en varias iteraciones, las pruebas que se establecen en las fases posteriores (3 y 4).

Figura 2. Fases definidas.



Fuente: elaboración propia.

A continuación se describe cada una de las fases:

- **Fase 0:** En esta fase se lleva a cabo la planificación de todo el proyecto, en la cual se identificaron todas las actividades necesarias para construir el trabajo final de Máster.
- **Fase 1:** Es la fase en donde se realiza la búsqueda de las herramientas que implementan la metodología GitOps, se determinan las características y aspectos a evaluar de estas y se construye un catálogo con estas herramientas.
- **Fase 2:** En esta fase se emplea el *Decision Analysis and Resolution (DAR)* área/disciplina de *CMMI*. De tal manera que, se elimine la suposición, subjetividad y se puedan tomar decisiones basadas en indicadores y métricas, para seleccionar la herramienta que implementa la metodología GitOps.
- **Fase 3:** Es la fase en la cual se expresa una solución construyendo una guía de despliegue de acuerdo a la metodología GitOps para realizar la implementación tecnológica seleccionada.
- **Fase 4:** En esta fase se pone en marcha la implementación, se instala la herramienta GitOps seleccionada y se evalúa la guía de despliegue a través de la implementación de un prototipo funcional.

Esta metodología está inspirada en el tesis doctoral, *Modelo de referencia para la gestión de recursos y servicios informáticos en ecosistemas tecnológicos de apoyo a la investigación* (Sepúlveda, 2023)

4. Desarrollo específico de la contribución

4.1. Planificación / Análisis / Requisitos

4.1.1. Buscar e identificar herramientas (técnicas)

En esta sección se realizó una revisión sistemática de literatura para determinar qué estudios formales que puedan aportar información valiosa para el desarrollo de los objetivos específicos que se plantearon en el Trabajo de Final de Master. Esta revisión sistemática se realizó en base a lo planteado por (Candela et al., 2022) en donde se construye un SMS

adaptado al campo de la informática. Adjunto a este trabajo se relaciona la bitácora usada para la revisión bibliográfica y en la cual se obtuvieron los siguientes resultados.

Tabla 1. Resultado de las consultas en las bases de datos.

Base de datos	ACM	IEEE	WOS	Springer	Scopus	Science Direct	Total
Estudios sin aplicación de criterios de exclusión	676	823	2966	7391	3870	3952	19678
Aplicación de los criterios de exclusión	37	812	71	334	25	64	1343

Fuente: elaboración propia.

Se obtuvieron como resultado **1343** publicaciones, en relación a las búsquedas en las bases de datos académicas, de términos asociados a la palabra “GitOps”. Posteriormente a esto se realizó un proceso de búsqueda por el método bola nieve el cual permitió filtrar estos estudios y reducirlos a 6 publicaciones; las cuales ayudaran a soportar la información, argumentos o afirmaciones presentadas en este Trabajo de Final de Master. Teniendo en cuenta ese número tan reducido de estudios relacionados, se evidencia la falta de proliferación de estudios formales alrededor de las herramientas y la necesidad de recurrir a la literatura gris (artículos no formales, publicaciones en sitios web, documentación oficial de industria, etc.) para encontrar información con respecto de las herramientas que puedan implementar la metodología GitOps.

Después de realizar una revisión sistemática sobre la metodología GitOps, se identificaron tres herramientas relevantes. Estas herramientas son:

1. **Argo CD:** Es una herramienta de código abierto utilizada para la implementación continua (*Continuous Deployment*) y la administración de aplicaciones en *Kubernetes*. Es una parte esencial del ecosistema de herramientas de Argo Project, que tiene como objetivo simplificar y automatizar el ciclo de vida de las aplicaciones basadas en contenedores en un entorno *Kubernetes*.
2. **Flux CD:** Es otra herramienta de código abierto que se utiliza para la implementación continua (*Continuous Deployment*) y la administración de aplicaciones en *Kubernetes*. Al igual que Argo CD, Flux CD también se basa en la metodología GitOps para

administrar la infraestructura y las aplicaciones de *Kubernetes* a través de repositorios Git.

- 3. **Jenkins X:** Es una plataforma de código abierto diseñada para automatizar el flujo de trabajo de desarrollo, pruebas y despliegue de aplicaciones en *Kubernetes* y entornos basados en la nube. Está construido sobre *Jenkins*, una herramienta popular para la integración y entrega continua (CI/CD), y proporciona una capa adicional de abstracción y automatización específicamente para aplicaciones nativas de la nube que se ejecutan en *Kubernetes*.

4.1.2. Determinar aspectos a evaluar de las herramientas

Para determinar qué aspectos se deben evaluar se ha desarrollado un cuadro comparativo, que permitirá valorar el aporte de cada una de las herramientas en cada una de las etapas del ciclo DevOps.

Tabla 2. Cuadro comparativo de las etapas del ciclo DevOps versus las características de la herramienta.

Etapas del Ciclo DevOps	Argo CD	Flux CD	Jenkins X
Planificación			
Desarrollo	Compatibilidad con múltiples herramientas de gestión/plantillas de configuración (Kustomize, Helm, Jsonnet, plain-YAML)	Soporte personalizado y Helm	Soporte Helm

Catálogo de herramientas de gestión de la configuración que implementan la metodología GitOps

<p>Construir</p>	<p>Implementación automatizada de aplicaciones en entornos de destino específicos</p>	<p>Interoperabilidad con proveedores de API de clúster (CAPI)</p>	<p>Jenkins X se envía con Tekton para una forma nativa de nube declarativa limpia para describir las canalizaciones</p> <p>Jenkins X brinda la capacidad de interactuar con canalizaciones a través de comentarios en solicitudes de incorporación de cambios(chatops)</p>
<p>Pruebas</p>		<p>Actualizaciones automatizadas de imágenes de contenedores a Git (escaneo de imágenes)</p>	
<p>Versionamiento</p>	<p>Retroceder/Volver a cualquier lugar a cualquier configuración de aplicación confirmada en el repositorio de Git</p>	<p>Integración perfecta con proveedores de Git (GitHub, GitLab, Bitbucket)</p> <p>Configuración de fuente de repositorios de Git y Helm, y cubos compatibles con S3 (p. ej., Minio)</p>	<p>Integración con Git</p>
<p>Despliegue</p>	<p>Sincronización automática o manual de aplicaciones a su estado deseado</p> <p>Integración de webhook (GitHub, BitBucket, GitLab)</p> <p>Enlaces PreSync, Sync, PostSync para admitir implementaciones de aplicaciones complejas (por ejemplo, actualizaciones blue/green y canary)</p>	<p>Manejo de eventos externos (receptores webhook)</p> <p>Interoperabilidad con proveedores de flujo de trabajo (GitHub Actions, Tekton, Argo)</p> <p>Actualizaciones automatizadas de imágenes de contenedores a Git (aplicación de parches)</p> <p>Conciliación periódica y desencadenada por eventos</p>	<p>Sincronización automática o manual de aplicaciones a su estado deseado</p>

<p>Operar</p>	<p>Capacidad para administrar e implementar en múltiples clústeres</p> <p>CLI para automatización e integración de CI</p> <p>Anulaciones de parámetros para anular parámetros de helm en Git</p>	<p>Gestión de dependencias (infraestructura y cargas de trabajo)</p>	<p>Jenkins X se remite a Terraform para configurar y administrar la infraestructura de la nube que necesita Jenkins X</p> <p>Jenkins X también puede funcionar bien con Jenkins para usuarios que tienen cargas de trabajo tradicionales.</p>
<p>Monitoreo</p>	<p>Interfaz de usuario web que proporciona una vista en tiempo real de la actividad de la aplicación</p> <p>Detección y visualización automáticas de cambios de configuración</p> <p>Análisis del estado de salud de los recursos de la aplicación</p> <p>Métricas de prometheus</p> <p>Pistas de auditoría para eventos de aplicaciones y llamadas API</p>	<p>Evaluación del estado (clústeres y cargas de trabajo)</p> <p>Alertas a sistemas externos (remitentes de webhook)</p>	
<p>Seguridad</p>	<p>Políticas de multiusuario y RBAC para autorización</p> <p>Integración SSO (OIDC, OAuth2, LDAP, SAML 2.0, GitHub, GitLab, Microsoft, LinkedIn)</p> <p>Tokens de acceso para la automatización</p>	<p>Integración con Kubernetes RBAC</p> <p>Actualizaciones automatizadas de imágenes de contenedores a Git (escaneo de imágenes)</p> <p>Validación basada en políticas (OPA, Kyverno, controladores de admisión)</p>	<p>Jenkins X prefiere trabajar con soluciones de proveedores de secretos reales como Vault o, cuando sea posible, con soluciones alojadas en la nube como administradores de Google , Azure o Amazon Secrets.</p> <p>Jenkins X GitOps funciona con External Secrets para brindar una experiencia integrada, de modo que la fuente de confianza de sus secretos sea un administrador de secretos y los valores</p>

			se repliquen en el clúster cuando sea necesario.
--	--	--	--

Fuente: elaboración propia.

Es importante señalar que el soporte para cada etapa del ciclo DevOps puede variar según las capacidades y extensiones de cada herramienta. Además, en algunos casos, se pueden requerir complementos o configuraciones adicionales para lograr la integración completa con las herramientas específicas utilizadas en cada etapa del ciclo DevOps. Cada herramienta tiene su enfoque particular y características únicas, por lo que se recomienda investigar y evaluar cuidadosamente qué herramienta se adapta mejor a los requisitos de tu equipo y proyecto.

4.1.3. Definir/Determinar Características de las herramientas

Para Definir/Determinar qué características se deben evaluar se ha desarrollado un cuadro comparativo, el cual permitirá apreciar las características generales de las herramientas.

Tabla 3. Cuadro comparativo de las características versus la herramienta.

Características	Argo CD	Flux CD	Jenkins X
Tipo de herramienta	Despliegue y gestión de aplicaciones en Kubernetes con GitOps	Despliegue y gestión de aplicaciones en Kubernetes con GitOps	Plataforma de CI/CD para aplicaciones nativas de la nube en Kubernetes
Metodología GitOps	Sí	Sí	Sí
Automatización	Implementación continua y sincronización	Implementación continua y sincronización	Automatización completa del flujo de trabajo de CI/CD

Entornos de previsualización	Sí	Sí	Sí (Mediante Previews)
Integración CI/CD	Integración con herramientas CI/CD mediante plugins	Integración con herramientas CI/CD	Integración con herramientas CI/CD
Extensibilidad	Sí, mediante plugins	Sí, mediante plugins	Sí, mediante plugins
Enfoque	Enfocado en la administración de aplicaciones en Kubernetes	Enfocado en la administración de aplicaciones en Kubernetes	Enfocado en aplicaciones nativas de la nube en Kubernetes
Facilidad de uso	Mayor enfoque en operaciones de Kubernetes	Mayor enfoque en operaciones de Kubernetes	Mayor enfoque en desarrollo y entrega continua
Comunidad y soporte	Activa comunidad de usuarios y soporte	Activa comunidad de usuarios y soporte	Activa comunidad de usuarios y soporte
Hosting: On premise/Cloud	On premise/Cloud	On premise/Cloud	On premise/Cloud

Fuente: elaboración propia.

Es importante tener en cuenta que este cuadro comparativo es solo una vista general y que cada herramienta tiene sus ventajas y desventajas según los requisitos y preferencias específicas del proyecto o equipo. Es recomendable investigar más a fondo cada herramienta y realizar pruebas para determinar cuál se ajusta mejor a tus necesidades.

4.1.4. Construir el catálogo de herramientas GitOps caracterizadas

A Continuación se presenta el diseño del catálogo para las herramientas que implementan la metodología GitOps con la información más relevante con respecto a ella.

Tabla 4. Catálogo de las herramientas

Catálogo de herramientas	Argo CD	Flux CD	Jenkins X
Sitio web oficial	https://argoproj.github.io/argo-cd/	https://fluxcd.io/	https://www.jenkins-x.io/
Repositorio GitHub	https://github.com/argoproj/argo-cd	https://github.com/fluxcd/flux	https://github.com/jenkins-x/jx
Crunchbase	https://www.crunchbase.com/organization/cloud-native-computing-foundation	https://www.crunchbase.com/organization/cloud-native-computing-foundation	https://www.crunchbase.com/organization/continuous-delivery-foundation-cdf
Primer commit	14 de Febrero de 2018	07 de Julio de 2016	5 de enero de 2018
Último commit	18 de septiembre de 2023	01 de Noviembre de 2022	15 de septiembre de 2023
Contribuidores	[1069 es el número de contribuidores actualizado hasta 2023]	[275 es el número de contribuidores actualizado hasta 2023]	[36 es el número de contribuidores actualizado hasta 2023]
Dev Stats	https://argo.devstats.com/	https://flux.devstats.com/	
Stack Overflow	https://stackoverflow.com/questions/tagged/argoproj+or+argo	https://stackoverflow.com/questions/tagged/flux	https://stackoverflow.com/questions/tagged/jenkins-x
Blog	https://blog.argoproj.io/	https://fluxcd.io/blog/	https://jenkins-x.io/blog/
Slack	https://argoproj.github.io/community/join-slack	https://cloud-native.slack.com/messages/flux	https://kubernetes.slack.com/messages/C9MBGQJRH https://kubernetes.slack.com/messages/flux

			.com/messages/C9LTHT2BB
--	--	--	---

Fuente: elaboración propia.

4.1.5. Evaluar las alternativas

Considerando los elementos e información extraídos de la **Tabla 1 y Tabla 2**, los cuales serán utilizados como insumo para aplicar la metodología conocida como "Análisis de Decisiones y Resolución" (DAR por sus siglas en inglés), tal como se describe en los modelos de Integración de Modelos de Madurez de Capacidades (CMMI). Estos modelos CMMI, definidos como mejores prácticas orientadoras para las organizaciones, facilitan la mejora continua de sus procesos (CMMI, 2010). Por lo tanto, la metodología DAR es un enfoque analítico que involucra un proceso de evaluación formal para sopesar las opciones o cursos de acción disponibles en una decisión de importancia para la organización. Esta metodología emplea criterios predefinidos para minimizar la subjetividad y conferir mayor confiabilidad en la elección de una solución que cumpla con los requerimientos y limitaciones inherentes al problema en cuestión. (Valencia, 2022).

4.1.5.1. Definir los criterios para evaluar las soluciones alternativas

Restricciones DAR. Si alguna herramienta no cumple al menos una de las siguientes condiciones, no se considerará en la evaluación de los criterios de selección y se descartará como opción de elección.

- Organización: la herramienta a seleccionar debe ser un proyecto perteneciente a la "Landscape - Cloud Native Computing Foundation" de la CNCF (s. f.).
- Objeto de kubernetes: la herramienta a seleccionar debe ser un elemento que se pueda integrar fácilmente con un cluster de *kubernetes*, por ejemplo ser un controlador.

Criterios de selección DAR. Las herramientas serán calificadas aplicando los siguientes criterios.

- **Hosting:** la herramienta a seleccionar debe contar con la capacidad de ejecutarse en varios entornos de computación, pueden ser On premise o Cloud.
- **Entornos de previsualización:** la herramienta debe contar con tableros de control que permitan observar las operaciones realizadas.
- **Extensibilidad:** las funciones de la herramienta pueden ser extendidas mediante plugins o software que se pueda acoplar.
- **Facilidad de uso:** se calificará cada tecnología según su facilidad de instalación, configuración, despliegue, mantenibilidad y uso para implementar integración y despliegue continuo.
- **Comunidad y soporte:** las calificaciones (estrellas y bifurcaciones) de los repositorios oficiales de cada tecnología en GitHub, se tendrán en cuenta para estimar la popularidad de las mismas, lo que puede traducirse en mayor disponibilidad de soporte, herramientas y evolución de la tecnología.
- **Documentación:** se calificará la disponibilidad de documentación oficial, página web, foros, comunidad y el esquema de soporte para la solución de problemas de implementación o errores en el código fuente.

4.1.5.2. Definir el rango y la escala para clasificar los criterios de evaluación

Con base en lo anterior y para efectos del presente trabajo final de Master, en la **Tabla 5** se definieron los pesos, que representan la importancia de cada criterio de selección frente a las herramientas, en una escala de valor de criterio de tres valores: 1 a 3, donde 0 equivale “no cumple”, 1 equivale “parcialmente lo cumple” y 3 “cumple plenamente”. Para definir los pesos se empleó un procedimiento de votación entre 3 personas vinculadas al trabajo final de Master, quienes de manera aislada establecieron el nivel de importancia de cada criterio, para posteriormente promediar los resultados.

Tabla 5. Pesos criterios de selección

Criterio	1	2	3
Hosting			
Entornos de previsualización			
Extensibilidad			
Facilidad de uso			
Comunidad y soporte			
Documentación			

4.1.5.3. Clasificar los criterios

Continuando con el proceso de selección mediante la aplicación de la metodología DAR, se efectuó la calificación de cada criterio de selección, de la siguiente manera:

- **Hosting:** la herramienta a seleccionar debe contar con la capacidad de ejecutarse en varios entornos de computación, pueden ser On premise o Cloud.
 - Calificación en 1: Se ejecuta en un entorno On premise.
 - Calificación en 2: Se ejecuta en un entorno Cloud.
 - Calificación en 3: Se ejecuta en un entorno On premise/Cloud.
- **Entornos de previsualización:** este criterio se calificó con base en la siguiente escala.
 - Calificación en 1: la previsualización de la herramienta es a través de la consola o CLI.
 - Calificación en 2: Algunas de las funcionalidades de la herramienta se pueden previsualizar en la interfaz gráfica.
 - Calificación en 3: La mayoría de las funcionalidades de la herramienta se pueden previsualizar en la interfaz gráfica.

- Extensibilidad: este criterio se calificó con base en la siguiente escala.
 - Calificación en 1: No posee extensiones para aumentar las capacidades de la herramienta.
 - Calificación en 2: Posee extensiones creadas solo por la misma organización o proyecto.
 - Calificación en 3: Posee extensiones que se pueden adaptar a las APIs de los diferentes proveedores de herramientas.
- Facilidad de uso: se calificará cada tecnología según su facilidad de instalación, configuración, despliegue, mantenibilidad y uso para implementar integración y despliegue continuo.
 - Calificación en 1: la instalación es por medio de una serie de comandos.
 - Calificación en 2: la instalación es por medio archivos YAMLS.
 - Calificación en 3: la instalación es por medio de un Helm Chart.
- Comunidad y soporte: la calificación de este criterio se obtuvo mediante la suma de la cantidad de estrellas del proyecto según los repositorios oficiales de cada herramienta.
 - Calificación en 1: la suma de la cantidad de estrellas del proyecto según los repositorios oficiales se encuentra entre 0 y 4.500.
 - Calificación en 2: la suma de la cantidad de estrellas del proyecto según los repositorios oficiales se encuentra entre 4.500 y 9000.
 - Calificación en 3: la suma de la cantidad de estrellas del proyecto según los repositorios oficiales se encuentra entre 9000 o más.
- Documentación: este criterio se calificó con base en la siguiente escala.
 - Calificación en 1: no registra fuentes oficiales de consulta y poca información en sitios no oficiales.

- Calificación en 2: no registra fuentes oficiales de consulta, aunque se registra información en sitios no oficiales.
- Calificación en 3: registra información oficial de consulta a nivel básica y avanzada, de fácil acceso

4.1.5.4. Evaluar los criterios y su importancia relativa

Se lleva a cabo la verificación de cada una de las herramientas con el propósito de comprobar si se ajustan o no a las restricciones definidas.

- Organización: teniendo en cuenta que la tecnología a seleccionar debe ser un proyecto perteneciente a la *Cloud Native Computing Foundation*, se procede a revisar el Interactive LandScape en el siguiente link: <https://landscape.cncf.io/>, y se verifica que la herramienta Argo CD, Flux CD y Jenkins X, hacen parte de los proyectos soportados por la Cloud Native Computing Foundation.
- Objeto de kubernetes: Por otra parte se debe considerar que la herramienta a seleccionar debe ser un elemento que se pueda integrar fácilmente con un cluster de kubernetes; se encuentra dentro de la documentación que las herramientas Argo CD y Flux CD son objetos que se pueden instalar dentro del cluster de kubernetes. Otro punto es que la herramienta de Jenkins X también se puede instalar en kubernetes por medio de archivos YAML.

Teniendo en cuenta la verificación de las restricciones establecidas para evaluar las herramientas, no se descarta ninguna herramienta y se procede a evaluar las 3 alternativas.

A continuación se construye la **tabla 6** para realizar la evaluación de las alternativas, teniendo en cuenta el valor de cada criterio y la calificación que se le asigna a cada uno.

Tabla 6. Matriz DAR - Decision, Analysis and Resolution

Declaración del problema:		Seleccionar una herramienta que implemente la metodología GitOps.						
Peso de cada criterio	2	1	2	3	3	3	14	
% de aporte de cada criterio	14%	7%	14%	21%	21%	21%	100%	
Opciones	H	EP	E	FU	CS	D	Puntaje	Representación
Argo CD	3	3	3	3	3	3	3	21%
Flux CD	3	2	3	3	2	3	2.7	19%
Jenkins X	3	2	3	2	1	2	2.1	15%
Restricciones								
<u>Organización:</u> la herramienta a seleccionar debe ser un proyecto perteneciente a la Cloud Native Computing Foundation.								
<u>Objeto de kubernetes:</u> la herramienta a seleccionar debe ser un elemento que se pueda integrar fácilmente con un cluster de kubernetes, por ejemplo ser un controlador.								
Riesgos								
Fin del soporte a la herramienta por parte de sus creadores.								
Convenciones								
<ul style="list-style-type: none"> ● H: Hosting ● EP: Entornos de previsualización ● E: Extensibilidad ● FU: Facilidad de uso ● CS: Comunidad y soporte ● D: Documentación 								

Fuente: elaboración propia.

Análisis de resultados metodología DAR

Luego de aplicar la metodología DAR, la herramienta que implementa la metodología GitOps con mayor puntaje es **Argo CD** con 3 puntos equivalentes al 21% de representación respecto a las demás herramientas. Asimismo, **Flux CD** obtuvo el segundo lugar con 2.7 puntos equivalente al 19% de representación y **Jenkins X** obtuvo el tercer lugar con 2.1 puntos equivalente al 15%. De este modo, se decide seleccionar Argo CD como la herramienta en la cual se desplegará un prototipo que permita visibilizar las ventajas y oportunidades que ofrece la metodología GitOps.

Entre las características más relevantes que ofrece Argo CD, se destacan las siguientes:

- **Interfaz de usuario amigable:** Argo CD ofrece una interfaz de usuario basada en web que facilita la gestión y visualización de aplicaciones y recursos en clústeres de Kubernetes. Esta interfaz intuitiva ayuda a los usuarios a comprender y controlar el estado de sus aplicaciones.
- **Implementación Declarativa:** Argo CD adopta una filosofía de implementación declarativa, lo que significa que definen el estado deseado de tu las aplicaciones y recursos en archivos YAML. Luego, Argo CD se encarga de comparar y sincronizar continuamente el estado real del clúster con el estado declarado en los archivos YAML.
- **Automatización de Sincronización:** Argo CD automatiza la comparación y sincronización de recursos en el clúster con los archivos de configuración YAML. Esto garantiza que el estado actual del clúster coincida con el estado deseado, lo que reduce la posibilidad de desviaciones no deseadas.
- **Auditoría e Historial:** Argo CD proporciona un registro detallado de las actividades y cambios realizados en las aplicaciones a lo largo del tiempo. Esto facilita la auditoría y permite a los usuarios rastrear quién realizó qué cambios y cuándo.
- **Soporte GitOps:** Argo CD es compatible con la metodología GitOps, que implica almacenar y gestionar el estado deseado de las aplicaciones en repositorios Git. Esto permite una gestión centralizada de la configuración y un seguimiento de versiones efectivo.
- **Automatización de Rollbacks:** En caso de fallos durante la implementación, Argo CD permite revertir a versiones anteriores de las aplicaciones de manera sencilla. Esto ayuda a mantener la estabilidad del sistema y acelerar la recuperación de problemas.
- **Helm Support:** Argo CD es compatible con Helm, lo que significa que puede gestionar la implementación y actualización de aplicaciones empaquetadas en gráficos Helm. Esto facilita la gestión de aplicaciones complejas con múltiples componentes y dependencias.
- **Extensiones personalizadas:** Argo CD permite la creación de extensiones personalizadas para integrar con sistemas y flujos de trabajo específicos. Esto brinda

flexibilidad para adaptar la herramienta a las necesidades específicas de la organización.

- Notificaciones y Alertas: Puedes configurar notificaciones y alertas para estar informado sobre los cambios y eventos relacionados con las aplicaciones y recursos.

Por otra parte, Flux CD obtuvo una muy buena calificación, es la herramienta precursora en el uso de la metodología GitOps y posee una colección de controladores como por ejemplo el operador flagger que permite extender las funcionalidades de Flux CD, para desarrollar todo un entorno de integración y despliegue continuo completo. Para ampliar la información sobre flagger visite el siguiente enlace: <https://flagger.app/>

Considerando los beneficios de la herramienta Flux CD, la empresa *Weaveworks* desarrolló un subsistema de Flux CD para Argo CD llamado **flamingo** el cual permite expandir las funcionalidades de Argo CD con las ventajas de la colección de controladores que maneja Flux CD. Además, **Flamingo** disminuye la preocupación y la distancia experimentada por los usuarios al adoptar una nueva herramienta, ya que se integra de manera fluida con Argo CD para aprovechar al máximo las ventajas de ambos enfoques. Para ampliar la información sobre flamingo visite los siguientes enlaces:

- <https://www.weave.works/blog/flamingo-expand-argo-cd-with-flux>
- <https://github.com/flux-subsystem-argo/flamingo>

4.2. Descripción del sistema desarrollado / Implementación

4.2.1. Construir la guía de despliegue

4.2.1.1. Describir prerequisites para la instalación

Considere el cumplimiento previo de los siguientes elementos para una instalación correcta de la herramienta:

- Permiso de usuario dentro del sistema, debe ser un usuario con privilegios elevados o administrador.
- Permiso o acceso a internet, si se encuentra en un ambiente On premise o Cloud.

- Tener libre los siguientes puertos de la máquina: 80, 443, 6443, 30000-40000.
- Tener una memoria RAM con una capacidad de 4 - 8 GB.
- Tener instalado Kubernetes > = v1.27.2
- Tener instalado Kubectl > = v1.27.4
- Tener instalado Helm > = v3.12.3
- Acceso a los manifiestos Yaml o Helm Charts de la versión más reciente de Argo CD (En este caso la versión v2.8.0+804d4b8).

Una vez adquirida la comprensión de la necesidad de implementar la herramienta Kubernetes, resulta fundamental considerar los siguientes aspectos: (Morejón, 2023).

- **Kubernetes como Servicio:** La adecuada implementación o actualización de los nodos control-plane en el entorno de *Kubernetes* puede presentar un desafío considerable y complejo. Las plataformas empleadas en la rutina diaria ofrecen la opción de configurar los nodos control-plane como un servicio. Esto implica que la instalación, el mantenimiento y las actualizaciones serán gestionados por dichas plataformas. La disponibilidad de este servicio y su eventual costo adicional dependerán de la plataforma específica en uso. Entre las opciones que proporcionan este servicio se encuentran las siguientes:
 - <https://aws.amazon.com/es/eks/>
 - <https://cloud.google.com/kubernetes-engine>
 - <https://azure.microsoft.com/en-us/products/kubernetes-service/>
 - <https://www.alibabacloud.com/product/kubernetes>
 - <https://www.digitalocean.com/products/kubernetes>
 - <https://www.civo.com>
- **Entornos locales:** Se presentan diversas alternativas que posibilitan la implementación de *Kubernetes* en entornos locales. La ventaja inherente a estas opciones radica en la posibilidad de establecer un clúster sin incurrir en gastos, lo cual resulta idóneo para llevar a cabo las pertinentes pruebas conceptuales. Dentro de las alternativas más prevalentes, se destacan las siguientes:
 - <https://www.docker.com/products/docker-desktop/>
 - <https://minikube.sigs.k8s.io/docs/>

- <https://kind.sigs.k8s.io>
- <https://microk8s.io>
- <https://k3s.io>
- <https://k3d.io>
- **Operadores:** Los operadores de Kubernetes posibilitarán la ejecución integral de la instalación del clúster en las máquinas virtuales de la infraestructura seleccionada. Las tareas relacionadas con la instalación, el mantenimiento y las actualizaciones quedan a cargo del equipo que emplee estos mecanismos. Entre las soluciones más prevalentes, se incluyen las siguientes:
 - <https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>
 - <https://github.com/kubernetes/kops>

Figura 3. Modelo del proceso de instalación - paso 1.



Fuente: elaboración propia.

4.2.1.2. Planificar la instalación

Después de considerar el cumplimiento de los elementos y aspectos mencionados en el punto **4.2.1.1**; se utilizará un entorno fácil y rápido de instalar, el cual puede destruirse en cualquier momento.

Nota: por lo tanto es relevante señalar que la instalación en cuestión tiene la capacidad de ser reproducida en un número variable de ocasiones (n veces), lo que contribuye a la posibilidad de llevar a cabo un experimento completamente replicable.

En última instancia, la opción elegida para esta implementación es **Docker Desktop**. Esta elección se respalda en sus características y capacidades que se alinean de manera eficaz con los objetivos y requerimientos del proyecto. Con *Docker Desktop* como herramienta

seleccionada, se establece una base sólida para llevar a cabo las acciones propuestas con eficiencia y confiabilidad. (Morejón, 2023).

Para realizar la instalación de los prerrequisitos se recomienda hacerlo en el siguiente orden:

1. Instalar y configurar la herramienta **kubectl** en su sistema, para esto se recomienda que siga las instrucciones de la siguiente guía: <https://kubernetes.io/es/docs/tasks/tools/included/install-kubectl-linux/>
2. Instalar y configurar la herramienta **helm** en su sistema, para esto se recomienda que siga las instrucciones de la siguiente guía: <https://helm.sh/es/docs/intro/install/>
3. Instalar y configurar la herramienta **kubernetes** en su sistema usando **Docker Desktop**, para esto se recomienda que siga las instrucciones de la siguiente guía: <https://docs.docker.com/desktop/kubernetes/>

Figura 4. Modelo del proceso de instalación - paso 2.



Fuente: elaboración propia.

4.2.1.3. Describir el proceso de instalación

A continuación se describe detalladamente el procedimiento para la instalación de la herramienta seleccionada Argo CD que implementa la metodología GitOps.

En este punto es importante que tenga en cuenta si es el administrador de la plataforma, que podría delegar el mantenimiento de las herramientas a los mantenedores, por lo cual se recomienda usar gestores de paquetes de los sistemas operativos, los cuales le permiten gestionar las actualizaciones de las versiones y realizar una correcta instalación de los prerrequisitos del punto **4.2.1.1** y de la misma herramienta.

Teniendo en cuenta que se utilizara un entorno local, la instalación de todo el software necesario se realizará sobre una máquina MacBook Air M1, 2020 con 8GB de memoria RAM

y un sistema operativo Ventura 13.4.1. Por consiguiente estos son los pasos que serán necesarios para instalar los prerequisites.

Primero se debe instalar el software o el binario como lo conocen los administradores de plataformas **kubect1**, el cual es una interfaz de comandos que permite hacer consultas y administrar los elementos del cluster de kubernetes:

```
$ brew install kubect1
```

Segundo, instalar el software o el binario como lo conocen los administradores de plataformas **helm**, el cual es el administrador de paquetes de Kubernetes y es el software que permitirá descargar los paquetes de la herramienta Argo CD de los repositorios oficiales y mantenidos por la comunidad del proyecto.

```
$ brew install helm
```

Tercero, instalar la herramienta kubernetes con Docker Desktop sobre la máquina, para esto realice el siguiente procedimiento:

- En el panel de control de Docker, seleccione **Configuración**.
- Seleccione **Kubernetes** en la barra lateral izquierda.
- Junto a **Habilitar Kubernetes**, seleccione la casilla de verificación.
- Seleccione **Aplicar y reiniciar** para guardar la configuración y luego seleccione **Instalar** para confirmar. Esto instancia las imágenes necesarias para ejecutar el servidor de *Kubernetes* como contenedores e instala el comando **/usr/local/bin/kubect1** en su máquina.

Finalmente es el momento de instalar la herramienta que implementa la metodología GitOps, seleccionada.

A continuación se comparte la guía oficial de la herramienta:
<https://argo-cd.readthedocs.io/en/stable/operator-manual/installation/>

En esta guía podrá observar los diferentes métodos de instalación como por ejemplo Multi-Tenant, Core, Kustomize y **Helm**.

En este caso se instalará la herramienta por medio de **helm**; para este procedimiento se ha usado la documentación oficial de referencia para implementar la instalación: <https://github.com/argoproj/argo-helm> y <https://argoproj.github.io/argo-helm/>

Por medio de la CLI se deben ejecutar las siguientes instrucciones:

```
$ helm repo add argo https://argoproj.github.io/argo-helm
```

Este comando usa el gestor de paquetes de **helm** para acceder al repositorio de artefactos <https://artifacthub.io/> y agregar el paquete de la herramienta Argo CD al repositorio local de este.

Luego por medio de **helm** se realiza la instalación de este paquete y se le define el **namespace** de *kubernetes* en donde va estar alojada la herramienta Argo CD.

```
$ helm install argo-cd argo/argo-cd -n argocd --create-namespace
```

Por otra parte también es posible realizar la instalación de la herramienta Argo CD por medio de manifiestos YAML.

```
$ kubectl apply -f
https://raw.githubusercontent.com/argoproj/argo-cd/master/manifests
/install.yaml -n argocd
```

La desventaja es que la gestión de la actualización se debe realizar de forma manual y no obtendremos todas las ventajas de instalarlo por medio de un Helm Chart.

se sugiere realizar una revisión exhaustiva del contenido disponible en el siguiente enlace: <https://argo-cd.readthedocs.io/en/stable/operator-manual/installation/>

Figura 5. Modelo del proceso de instalación - paso 3.



Fuente: elaboración propia.

4.2.1.4. Describir aspectos de afinamiento en la configuración del ambiente

Es posible hacer modificaciones para el despliegue en la configuración de la herramienta por medio de **helm**, se sugiere realizar una revisión exhaustiva del contenido disponible en el siguiente enlace: https://helm.sh/es/docs/intro/using_helm/

Por ejemplo se puede observar las opciones que se pueden configurar en el chat, con el comando:

```
$ helm show values argo/argo-cd
```

Posteriormente, es posible modificar cualquiera de estas configuraciones en un archivo en formato YAML y, luego, suministrar dicho archivo durante el proceso de instalación.

```
$ helm install -f values.yaml argo/argo-cd argo-cd
```

Configuración de repositorios privados

Si los manifiestos de la aplicación se encuentran en un repositorio privado, se deben configurar las credenciales del repositorio. Argo CD admite credenciales HTTPS y SSH Git.

Credenciales

Los repositorios privados que requieren un nombre de usuario y contraseña normalmente tienen una URL que comienza `https://con git@` o `ssh://`.

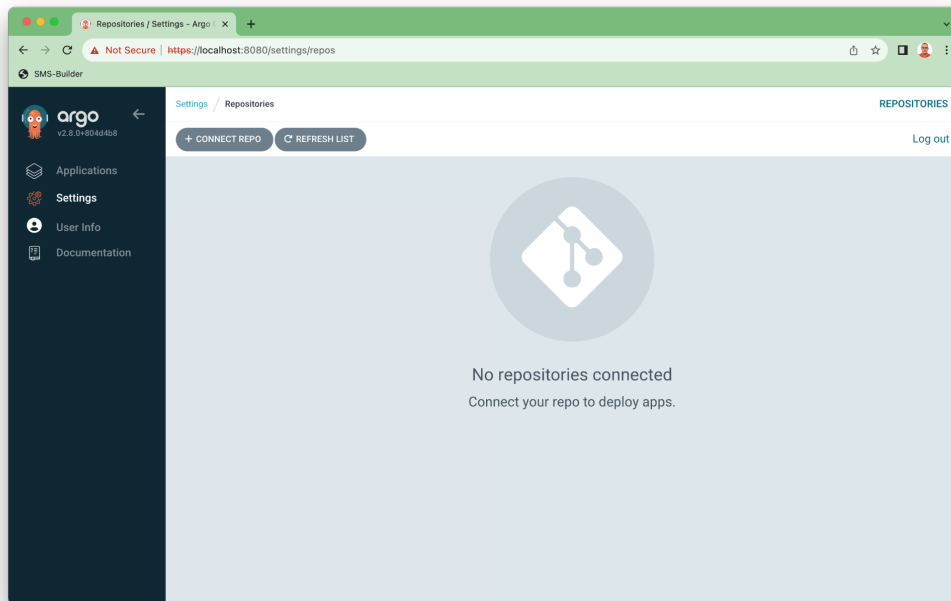
Las credenciales se pueden configurar usando Argo CD CLI:

```
$ argocd repo add https://github.com/argoproj/argocd-example-apps  
--username <username> --password <password>
```

o por medio de la interfaz de usuario:

1. Navegar a Settings/Repositories

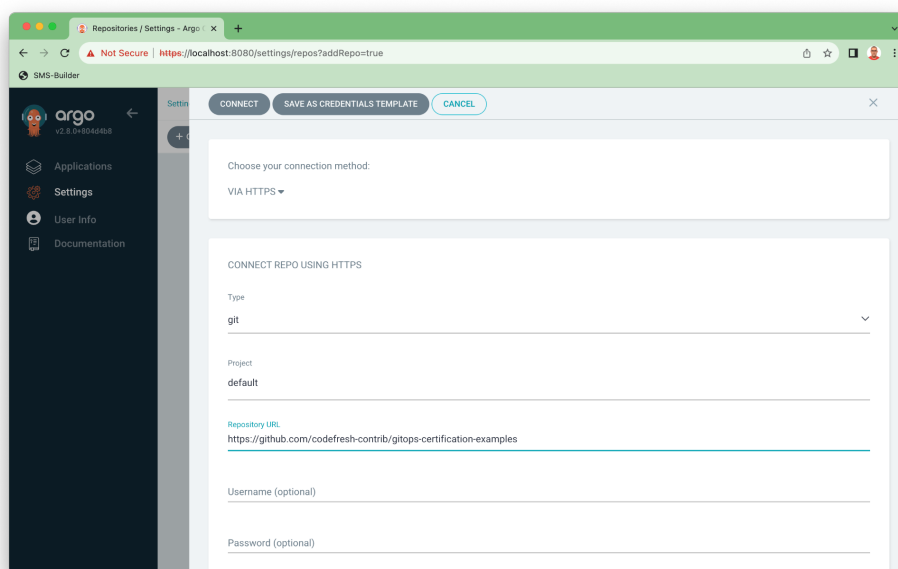
Figura 6. Interfaz de usuario de Argo CD en la sección Settings/Repositories.



Fuente: elaboración propia.

2. Haga clic en el botón Connect Repo using HTTPS e ingrese las credenciales

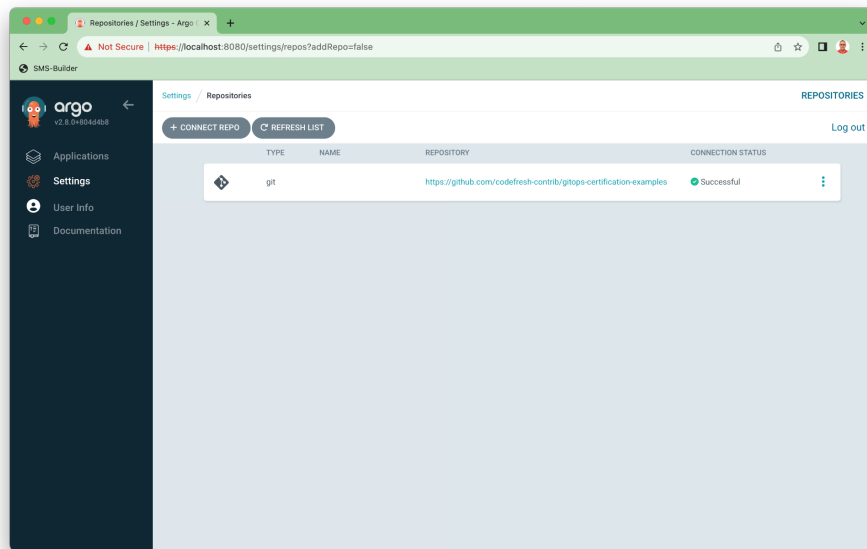
Figura 7. Interfaz de usuario de Argo CD en la sección Settings/Repositories + Connect Repo.



Fuente: elaboración propia.

3. Haga clic **Connect** para probar la conexión y agregar el repositorio.

Figura 8. Interfaz de usuario de Argo CD - Test - Connect Repo.



Fuente: elaboración propia.

Se sugiere realizar una revisión exhaustiva del contenido disponible en el siguiente enlace:

<https://argo-cd.readthedocs.io/en/stable/user-guide/private-repositories/>

Argo CD Image Updater

Es importante destacar que las herramientas que implementan la metodología GitOps se centran en el flujo de trabajo de despliegue continuo, lo quiere decir que se centran en el despliegue de las aplicaciones y no en su construcción.

La herramienta Argo CD, tiene una herramienta para solucionar esta problemática y es **Argo CD Image Updater**. Esta herramienta es la encargada de actualizar automáticamente las imágenes de contenedor de las cargas de trabajo de *Kubernetes* administradas por Argo CD. Esto quiere decir que el componente **Image Updater** es un controlador que se encarga de verificar la versión de la imagen dependiendo de la estrategia de actualización que se elija.

A continuación se presentan las diferentes estrategias de actualización:

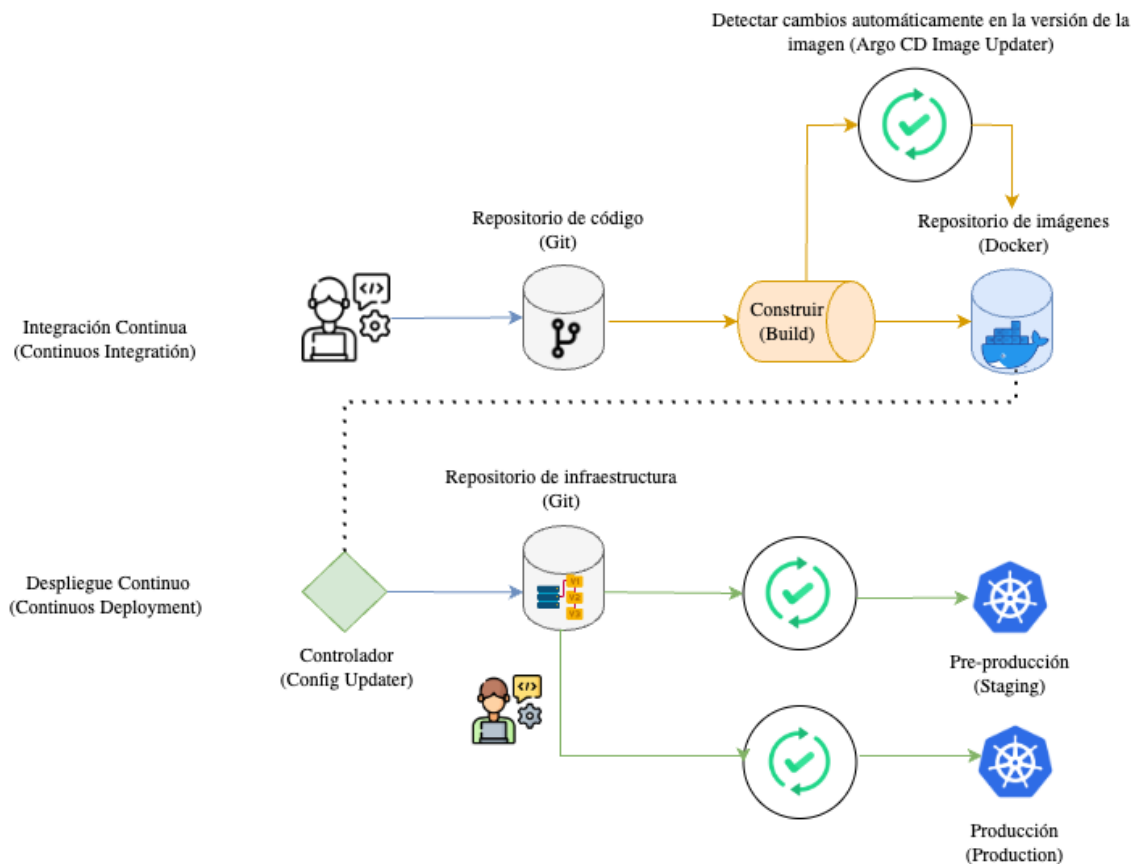
- **semver**: actualice a la versión más alta permitida según la restricción de imagen dada.
- **latest**: actualizar a la etiqueta de imagen creada más recientemente.
- **name**: actualiza a la última etiqueta en una lista ordenada alfabéticamente.
- **digest**: actualización a la versión enviada más reciente de una etiqueta mutable

Se sugiere realizar una revisión exhaustiva del contenido disponible en el siguiente enlace:

<https://argocd-image-updater.readthedocs.io/en/stable/>

En la **figura 9**, en la que se muestra nuevamente el flujo de trabajo de GitOps, y se añade el elemento **Argo CD Image Updater**, para demostrar gráficamente en donde interviene esta herramienta en todo el flujo de trabajo GitOps.

Figura 9. Flujo de trabajo con GitOps + Argo CD Image Updater.



Fuente: elaboración propia.

Métodos de instalación

Se recomienda ejecutar Argo CD Image Updater en el mismo clúster de espacio de nombres de *Kubernetes* en el que se ejecuta Argo CD.

Instalación como carga de trabajo de Kubernetes en el espacio de nombres de Argo CD

La forma más sencilla de ejecutar el actualizador de imágenes es instalarlo como una carga de trabajo de *Kubernetes* en el espacio de nombres donde se ejecuta Argo CD.

Aplicar los manifiestos de instalación.

```
$ kubectl apply -n argocd -f
https://raw.githubusercontent.com/argoproj-labs/argocd-image-updater/stable/manifests/install.yaml
```

Se sugiere realizar una revisión exhaustiva del contenido disponible en el siguiente enlace:

<https://argocd-image-updater.readthedocs.io/en/stable/install/installation/>

Configuración de Registries de Contenedores

Para que la herramienta Argo CD Image Updater pueda detectar la nueva versión de la imagen de Docker; la pueda descargar y correr dentro del cluster es necesario configurar un objeto ConfigMap dentro del cluster de la siguiente manera:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: argocd-image-updater-config
data:
  registries.conf: |
    registries:
    - name: Docker Hub
      api_url: https://registry-1.docker.io
      prefix: docker.io
      defaultns: library
      insecure: yes

$ kubectl apply -n argocd -f argocd-image-updater-config-cm.yaml
```

Y posteriormente se debe crear un objeto **tipo aplicación** con la definición de las estrategias para actualizar las imágenes de Docker dentro del clúster.

```

apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  annotations:
    argocd-image-updater.argoproj.io/write-back-method: argocd
    argocd-image-updater.argoproj.io/image-list:
myalias=docker.io/kostiscodefresh/gitops-simple-app
    argocd-image-updater.argoproj.io/myalias.update-strategy: semver
    argocd-image-updater.argoproj.io/myalias.force-update: "true"
  name: 'image-updater'
  namespace: argocd
spec:
  destination:
    server: 'https://kubernetes.default.svc'
    namespace: default
  source:
    repoURL:
'https://github.com/codefresh-contrib/gitops-certification-examples'
    path: ./helm-app
    targetRevision: HEAD
    sources: []
    project: 'default'
    
```

Para ampliar esta información, se sugiere realizar una revisión exhaustiva del contenido disponible en el siguiente enlace:

<https://argocd-image-updater.readthedocs.io/en/stable/configuration/registries/>

Figura 10. Modelo del proceso de instalación - paso 4.



Fuente: elaboración propia.

4.2.1.5. Describir el proceso de pruebas de la herramienta seleccionada

Por medio de helm podemos verificar los lanzamientos que se han realizado en el sistema, con el comando:

```
$ helm ls -n argocd
```

Por otra parte, se puede para verificar el estado de una liberación (release) que se ha instalado en un clúster de *Kubernetes* utilizando Helm, con el comando:

```
$ helm status argo-cd -n argocd
```

En el momento que se ejecute el comando **helm status**, la consola debe mostrar la información de la herramienta Argo CD.

A través del software **kubectl** se realizará la verificación de la creación del namespace que corresponde a la herramienta Argo CD.

```
$ kubectl get ns
```

Posteriormente, se verificará los Pods que se encuentran dentro del namespace de Argo CD y el estado en que se encuentran.

```
$ kubectl -n argocd get pods
```

La prueba anterior también se puede realizar por medio de la GUI de la herramienta *Lens*, la cual proporciona una interfaz gráfica y un conjunto de características avanzadas para gestionar y trabajar con clústeres de *Kubernetes* de manera más eficiente. *Lens* es una herramienta de código abierto que está diseñada para simplificar las tareas de administración, monitoreo y desarrollo en entornos de *Kubernetes*. Consulte el siguiente enlace para mayor información: <https://k8slens.dev/>

Si en este punto, se han logrado realizar las anteriores verificaciones mencionadas, es momento de verificar el funcionamiento del servicio de la herramienta Argo CD por medio del navegador, para realizar esto es necesario que ejecute el siguiente comando:

```
$ kubectl port-forward service/argo-cd-argocd-server -n argocd
8080:443
```

Finalmente abra el navegador e ingrese a la siguiente dirección <http://localhost:8080> y acepte el certificado, como resultado debe observar la interfaz gráfica de la herramienta Argo CD.

Figura 11. Modelo del proceso de instalación - paso 5.



Fuente: elaboración propia.

4.2.1.6. Describir aspectos de mantenibilidad de la herramienta

A través de **helm** es posible actualizar la versión del chat de Argo CD, lo que realiza el gestor de paquetes es observar los cambios realizados en los manifiestos de cada versión y aplicar dichos cambios a la infraestructura desplegada en el cluster de *kubernetes* para actualizar los componentes.

Para realizar lo anterior mencionado se utiliza el comando helm con la siguiente estructura:

```
$ helm upgrade NOMBRE_DE_LA_LIBERACION \
NOMBRE_DEL_REPOSITORIO/NOMBRE_DEL_CHART --version
NUMERO_DE_LA_VERSION
```

Donde **NOMBRE_DE_LA_LIBERACION** es el nombre de la liberación existente y **NOMBRE_DEL_REPOSITORIO/NOMBRE_DEL_CHART** es la ruta del repositorio y el nombre del chart.

Siguiendo estos pasos, es posible aumentar la versión del Helm Chart y permitir que los usuarios actualicen sus instalaciones para aprovechar las nuevas características y correcciones de errores.

Figura 12. Modelo del proceso de instalación - paso 6.



Fuente: elaboración propia.

4.3. Evaluación

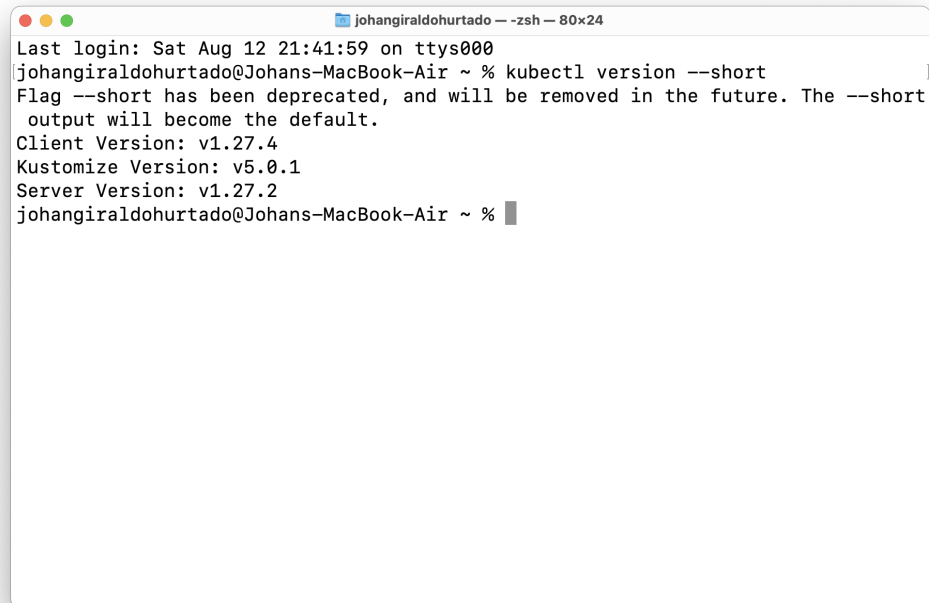
4.3.1. Implementar la herramienta seleccionada

En el ámbito de la gestión de aplicaciones y tecnologías emergentes, la metodología GitOps ha cobrado una creciente relevancia al ofrecer un enfoque innovador y eficiente para la implementación y operación de aplicaciones. Este enfoque, que utiliza Git como fuente de verdad para la infraestructura y la configuración, ha demostrado su capacidad para agilizar el proceso de despliegue y proporcionar mayor control y seguridad en el ciclo de vida de las aplicaciones. En este contexto, este trabajo se centra en demostrar cómo la implementación de una aplicación se puede llevar a cabo exitosamente utilizando GitOps como marco metodológico. A lo largo de esta investigación, exploraremos las etapas clave de este proceso, desde la definición de la infraestructura como código hasta la sincronización continua con el repositorio Git, ilustrando cómo GitOps transforma la manera en que las organizaciones gestionan sus aplicaciones, brindando agilidad, trazabilidad y una mayor capacidad de respuesta a los desafíos cambiantes del entorno tecnológico.

4.3.1.1. Cumplir prerequisites indicados en la guía de despliegue

A continuación se ejecutará el comando necesario para verificar la instalación de kubectl.

```
$ kubectl version --short
```

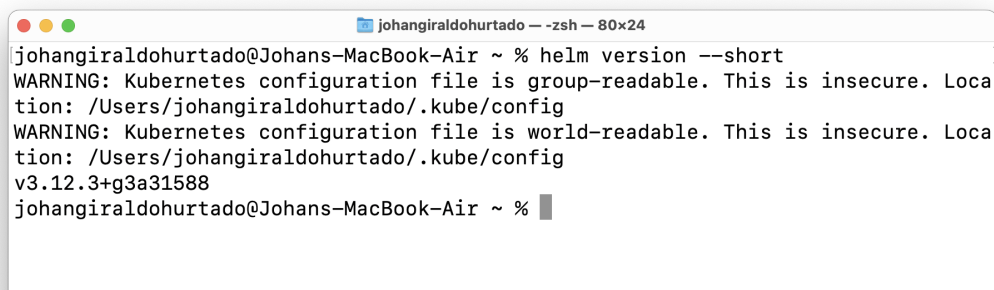

Figura 13. Ejecución del comando: `kubectl version --short`.A terminal window titled 'johangiraldohurtado --zsh -- 80x24' showing the execution of the command 'kubectl version --short'. The output displays the client, kustomize, and server versions, along with a deprecation warning for the --short flag.

```
johangiraldohurtado@Johans-MacBook-Air ~ % kubectl version --short
Flag --short has been deprecated, and will be removed in the future. The --short
output will become the default.
Client Version: v1.27.4
Kustomize Version: v5.0.1
Server Version: v1.27.2
johangiraldohurtado@Johans-MacBook-Air ~ %
```

Fuente: elaboración propia.

A continuación se ejecutará el comando necesario para verificar la instalación de helm.

```
$ helm version --short
```

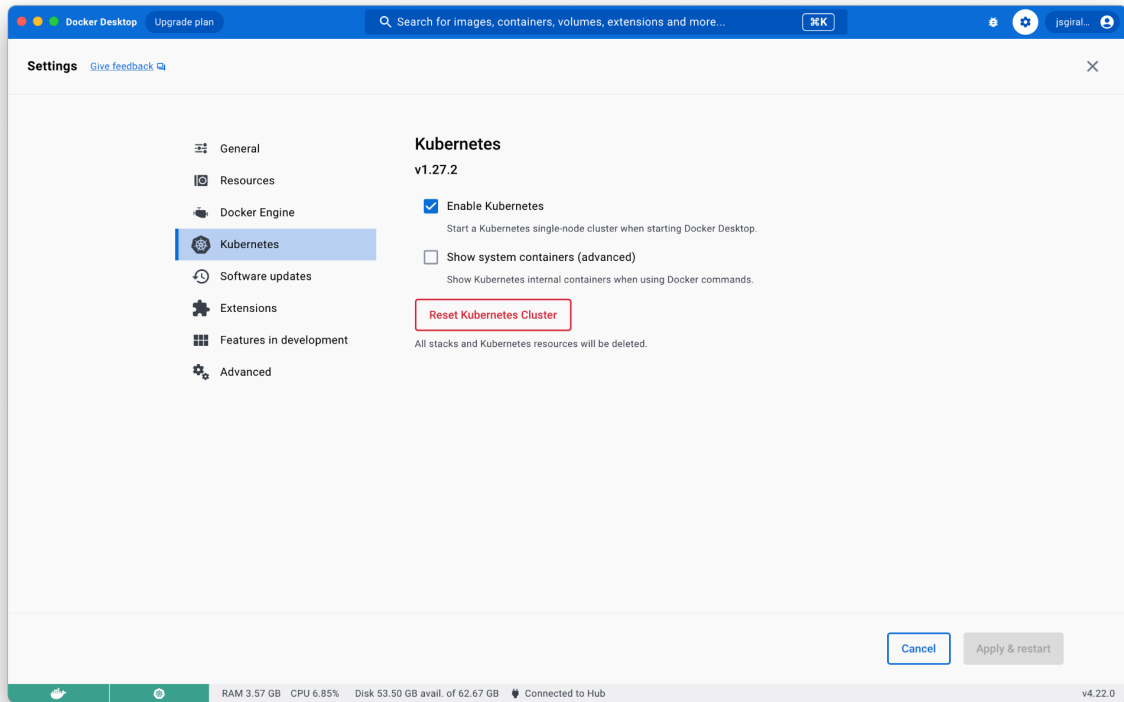
Figura 14. Ejecución del comando: `helm version --short`.A terminal window titled 'johangiraldohurtado --zsh -- 80x24' showing the execution of the command 'helm version --short'. The output displays the helm version and two warnings about the permissions of the Kubernetes configuration file.

```
johangiraldohurtado@Johans-MacBook-Air ~ % helm version --short
WARNING: Kubernetes configuration file is group-readable. This is insecure. Loca
tion: /Users/johangiraldohurtado/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Loca
tion: /Users/johangiraldohurtado/.kube/config
v3.12.3+g3a31588
johangiraldohurtado@Johans-MacBook-Air ~ %
```

Fuente: elaboración propia.

Imagen de la comprobación de la instalación de Kubernetes con la herramienta Docker Desktop.

Figura 15. Imagen de la aplicación Docker Desktop.



Fuente: elaboración propia.

4.3.1.2. Realizar proceso de instalación de la herramienta seleccionada

En esta sección se deben ejecutar los comandos mencionados para realizar la instalación de la herramienta que implementa la metodología GitOps seleccionada.

Primeramente, se debe agregar el repositorio de Argo CD a la máquina local con el comando:

```
$ helm repo add argo https://argoproj.github.io/argo-helm
```

Figura 16. Ejecución del comando: `helm repo add argo https://argoproj.github.io/argo-helm`.

```
johangiraldohurtado@Johans-MacBook-Air ~ % helm repo add argo https://argoproj.github.io/argo-helm
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /Users/johangiraldohurtado/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /Users/johangiraldohurtado/.kube/config
"argo" has been added to your repositories
johangiraldohurtado@Johans-MacBook-Air ~ %
```

Fuente: elaboración propia.

Luego de agregar el repositorio, se procede a instalar el paquete con el siguiente comando:

```
$ helm install argo-cd argo/argo-cd -n argocd --create-namespace
```

Figura 17. Ejecución del comando: `helm install argo-cd argo/argo-cd -n argocd --create-namespace`.

```
johangiraldohurtado@Johans-MacBook-Air ~ % helm install argo-cd argo/argo-cd -n argocd --create-namespace
Last login: Sun Aug 13 17:44:31 on ttys000
johangiraldohurtado@Johans-MacBook-Air ~ % helm install argo-cd argo/argo-cd -n argocd --create-namespace
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /Users/johangiraldohurtado/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /Users/johangiraldohurtado/.kube/config
NAME: argo-cd
LAST DEPLOYED: Sun Aug 13 18:18:29 2023
NAMESPACE: argocd
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
In order to access the server UI you have the following options:

1. kubectl port-forward service/argo-cd-argocd-server -n argocd 8080:443
   and then open the browser on http://localhost:8080 and accept the certificate

2. enable ingress in the values file `server.ingress.enabled` and either
   - Add the annotation for ssl passthrough: https://argo-cd.readthedocs.io/en/stable/operator-manual/ingress/#option-1-ssl-passthrough
   - Set the `configs.params."server.insecure"` in the values file and terminate SSL at your ingress: https://argo-cd.readthedocs.io/en/stable/operator-manual/ingress/#option-2-multiple-ingress-objects-and-hosts

After reaching the UI the first time you can login with username: admin and the random password generated during the installation. You can find the password by running:

kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}" | base64 -d

(You should delete the initial secret afterwards as suggested by the Getting Started Guide: https://argo-cd.readthedocs.io/en/stable/getting_started/#4-login-using-the-cli)
johangiraldohurtado@Johans-MacBook-Air ~ %
```

Fuente: elaboración propia.

4.3.1.3. Cumplir aspectos de configuración indicados en la guía de despliegue

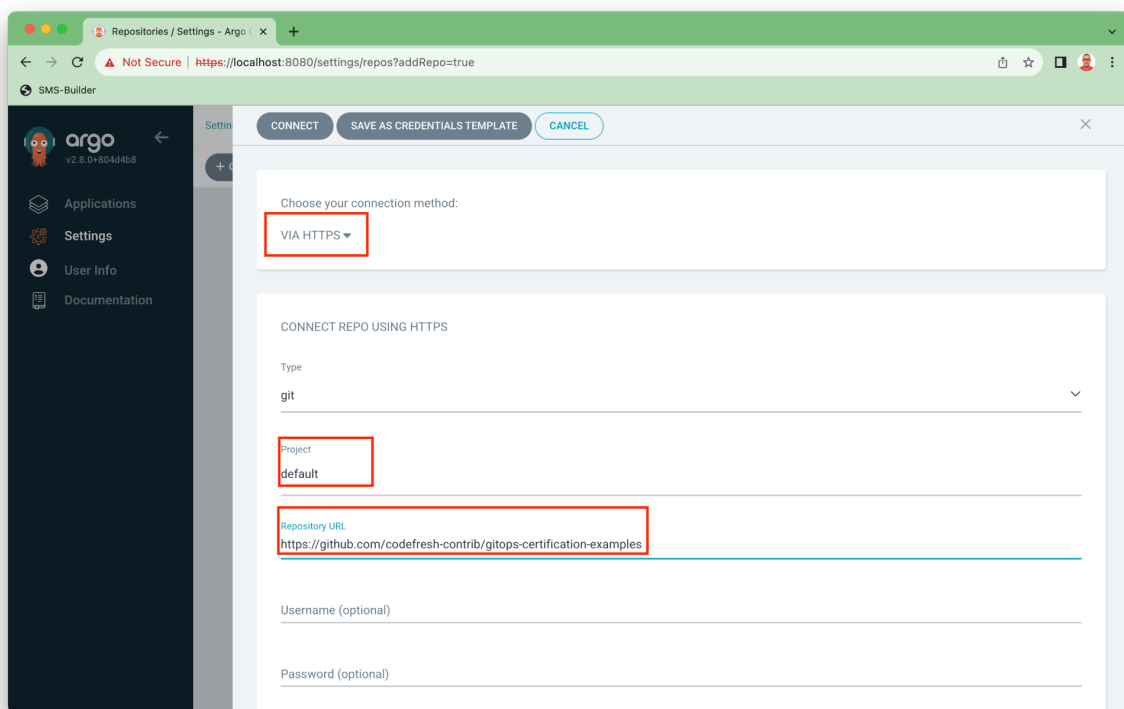
Uno de los elementos que se deben cumplir es la conexión al repositorio de Git de donde se extrae los manifiestos YAML para el despliegue de la aplicación.

Para esto se ingresa a la interfaz de Argo CD por medio del siguiente enlace: <https://localhost:8080/>

Luego se dirige al menú de Settings/Repositories, se presiona en el botón Connect Repo using HTTPS.

Posteriormente se selecciona el método de conexión en este caso es **HTTPS**, se indica el nombre del **proyecto** y la **url del repositorio de Git**.

Figura 18. Configuración de credenciales del repositorio de Git.



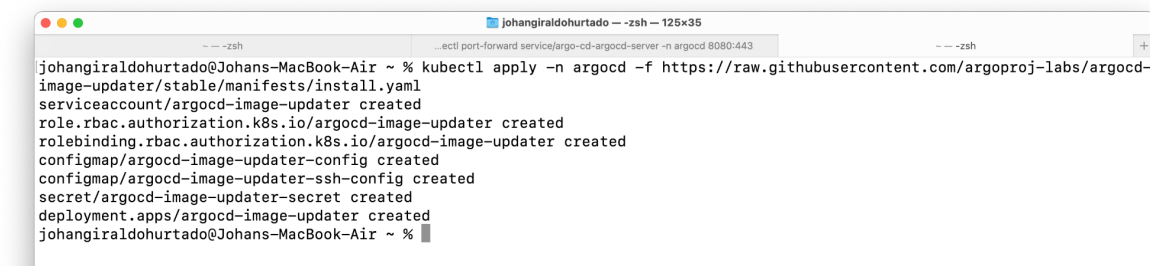
Fuente: elaboración propia.

Otro punto importante de esta sección es realizar el despliegue del objeto Argo CD Image Updater.

Para realizarlo es necesario ejecutar el siguiente comando:

```
$ kubectl apply -n argocd -f
https://raw.githubusercontent.com/argoproj-labs/argocd-image-updater
r/stable/manifests/install.yaml
```

Figura 19. Instalación del Argo CD Image Updater por medio de manifiestos YAML.



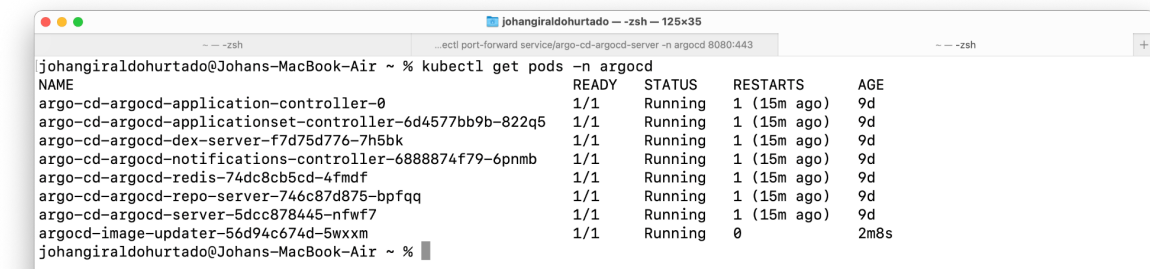
```
johangiraldohurtado@Johans-MacBook-Air ~ % kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj-labs/argocd-image-updater/stable/manifests/install.yaml
serviceaccount/argocd-image-updater created
role.rbac.authorization.k8s.io/argocd-image-updater created
rolebinding.rbac.authorization.k8s.io/argocd-image-updater created
configmap/argocd-image-updater-config created
configmap/argocd-image-updater-ssh-config created
secret/argocd-image-updater-secret created
deployment.apps/argocd-image-updater created
johangiraldohurtado@Johans-MacBook-Air ~ %
```

Fuente: elaboración propia.

Luego de crear el objeto Argo CD Image Updater por medio de manifiestos YAML, es necesario verificar la creación del Pod de este objeto en el cluster de *kubernetes* en el namespace de *argocd*. Esta operación se realiza a través de siguiente comando:

```
$ kubectl get pods -n argocd
```

Figura 20. Verificación de la Instalación del Argo CD Image Updater por consola.

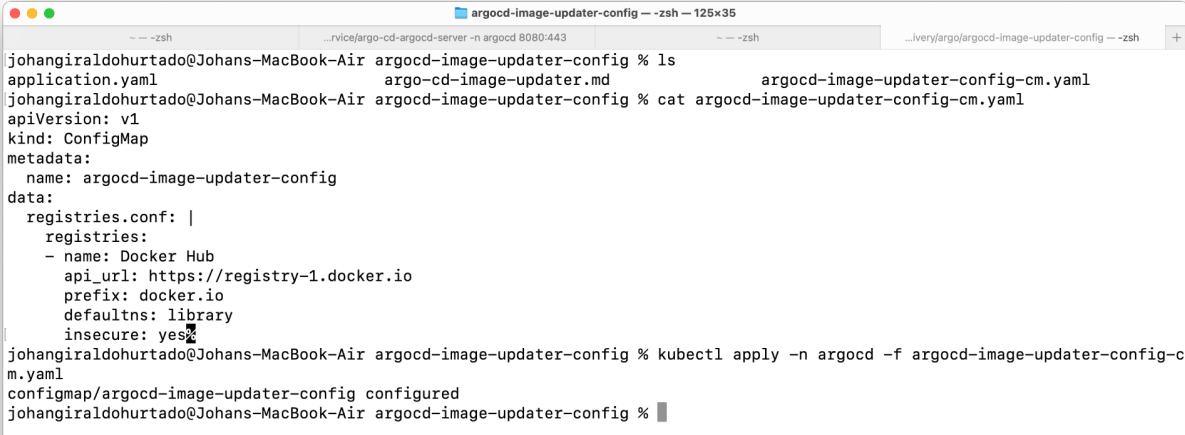


```
johangiraldohurtado@Johans-MacBook-Air ~ % kubectl get pods -n argocd
NAME                                READY   STATUS    RESTARTS   AGE
argo-cd-argocd-application-controller-0  1/1    Running   1 (15m ago)  9d
argo-cd-argocd-applicationset-controller-6d4577bb9b-822q5  1/1    Running   1 (15m ago)  9d
argo-cd-argocd-dex-server-f7d75d776-7h5bk  1/1    Running   1 (15m ago)  9d
argo-cd-argocd-notifications-controller-6888874f79-6pnmb  1/1    Running   1 (15m ago)  9d
argo-cd-argocd-redis-74dc8cb5cd-4fmdf  1/1    Running   1 (15m ago)  9d
argo-cd-argocd-repo-server-746c87d875-bpfqq  1/1    Running   1 (15m ago)  9d
argo-cd-argocd-server-5dcc878445-nfwf7  1/1    Running   1 (15m ago)  9d
argocd-image-updater-56d94c674d-5wxxm  1/1    Running   0          2m8s
johangiraldohurtado@Johans-MacBook-Air ~ %
```

Fuente: elaboración propia.

Por último se debe crear el manifiesto YAML que permite realizar la configuración del registro de Docker para la actualización de las imágenes. Luego de crear este archivo indicado en la sección **4.2.1.4**.

Figura 21. Configuración del objeto ConfigMap que permite modificar el repositorio de las imágenes de Docker.

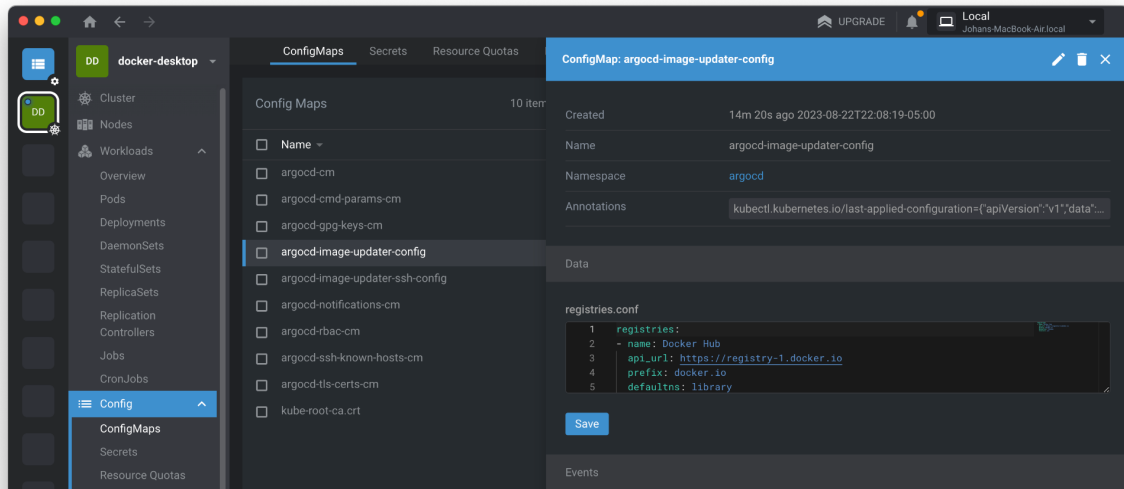


```
argocd-image-updater-config -- zsh -- 125x35
-- zsh
~/rvice/argo-cd-argocd-server -n argocd 8080:443
-- zsh
~/lvery/argo/argocd-image-updater-config -- zsh
johangiraldohurtado@Johans-MacBook-Air argocd-image-updater-config % ls
application.yaml          argo-cd-image-updater.md          argocd-image-updater-config-cm.yaml
johangiraldohurtado@Johans-MacBook-Air argocd-image-updater-config % cat argocd-image-updater-config-cm.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: argocd-image-updater-config
data:
  registries.conf: |
    registries:
    - name: Docker Hub
      api_url: https://registry-1.docker.io
      prefix: docker.io
      defaultns: library
      insecure: yes
johangiraldohurtado@Johans-MacBook-Air argocd-image-updater-config % kubectl apply -n argocd -f argocd-image-updater-config-cm.yaml
configmap/argocd-image-updater-config configured
johangiraldohurtado@Johans-MacBook-Air argocd-image-updater-config %
```

Fuente: elaboración propia.

Por medio de la herramienta *lens*, es posible verificar la configuración del objeto ConfigMap por interfaz gráfica.

Figura 22. Verificación de los valores aplicados por el manifiesto YAML en el objeto ConfigMap.



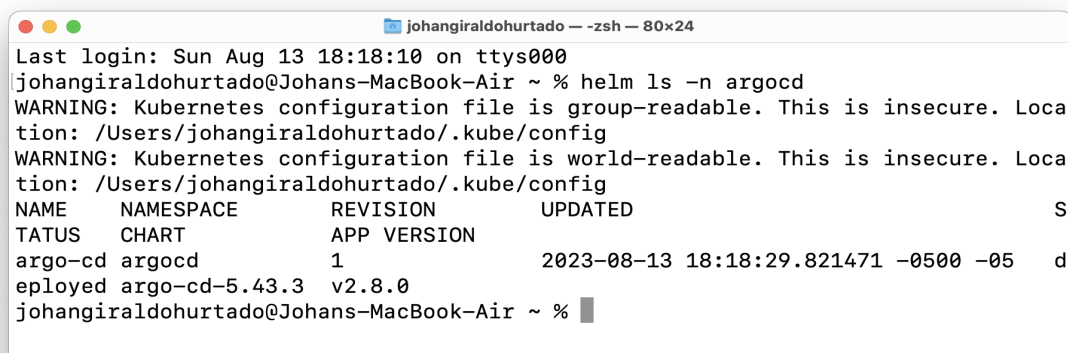
Fuente: elaboración propia.

4.3.1.4. Verificar el despliegue y el cumplimiento funcional de la herramienta seleccionada

Para verificar que el helm chart de Argo CD fue correctamente instalado es necesario ejecutar el siguiente comando:

```
$ helm ls -n argocd
```

Figura 23. Ejecución del comando: `helm ls -n argocd`.

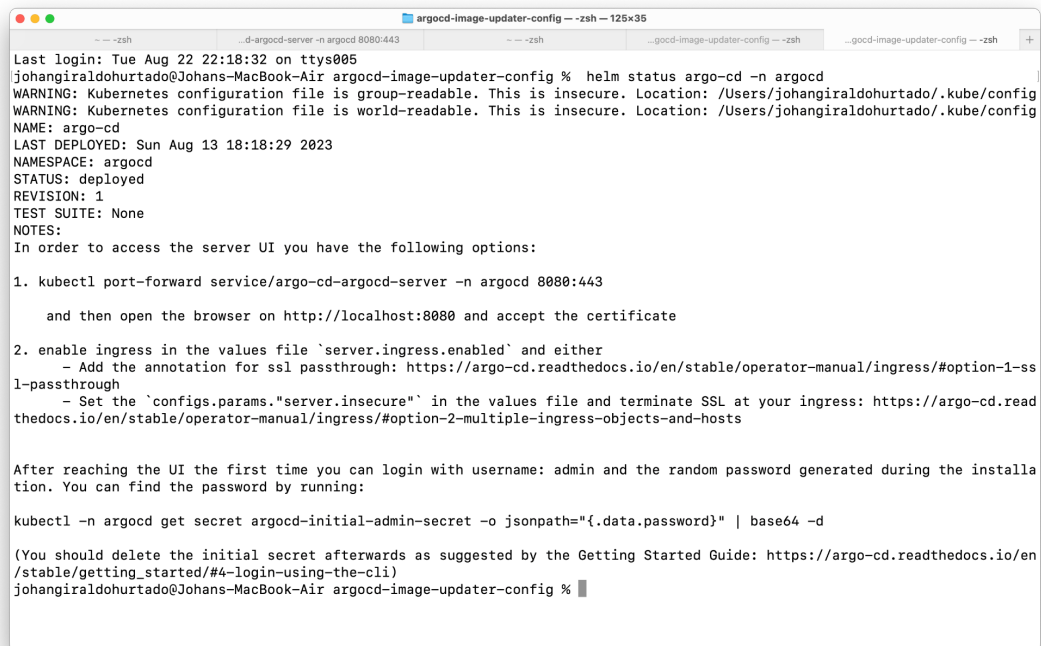


Fuente: elaboración propia.

Para verificar el estado en que se encuentra el helm chart de Argo CD es necesario ejecutar el siguiente comando:

```
$ helm status argo-cd -n argocd
```

Figura 24. Ejecución del comando: `helm status argo-cd -n argocd`.



```
argocd-image-updater-config -- zsh -- 125x35
Last login: Tue Aug 22 22:18:32 on ttys005
johangiraldohurtado@Johans-MacBook-Air argocd-image-updater-config % helm status argo-cd -n argocd
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /Users/johangiraldohurtado/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /Users/johangiraldohurtado/.kube/config
NAME: argo-cd
LAST DEPLOYED: Sun Aug 13 18:18:29 2023
NAMESPACE: argocd
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
In order to access the server UI you have the following options:

1. kubectl port-forward service/argo-cd-argocd-server -n argocd 8080:443

   and then open the browser on http://localhost:8080 and accept the certificate

2. enable ingress in the values file `server.ingress.enabled` and either
   - Add the annotation for ssl passthrough: https://argo-cd.readthedocs.io/en/stable/operator-manual/ingress/#option-1-ssl-passthrough
   - Set the `configs.params."server.insecure"` in the values file and terminate SSL at your ingress: https://argo-cd.readthedocs.io/en/stable/operator-manual/ingress/#option-2-multiple-ingress-objects-and-hosts

After reaching the UI the first time you can login with username: admin and the random password generated during the installation. You can find the password by running:

kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}" | base64 -d

(You should delete the initial secret afterwards as suggested by the Getting Started Guide: https://argo-cd.readthedocs.io/en/stable/getting_started/#4-login-using-the-cli)
johangiraldohurtado@Johans-MacBook-Air argocd-image-updater-config %
```

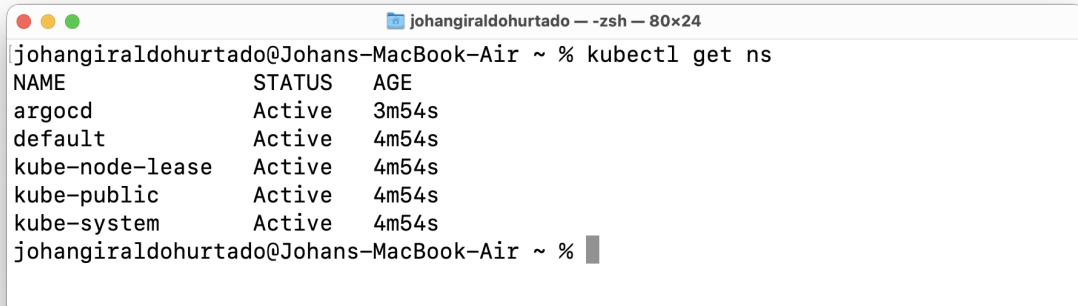
Fuente: elaboración propia.

Como se puede observar en la **Figura 24**, el Helm Chart de la herramienta Argo CD fue desplegado correctamente y proporciona la información necesaria para verificar su despliegue. Es necesario realizar un forwarding de puertos para exponer el puerto de la herramienta en la máquina y por otra parte también es necesario obtener el secreto de la credencial del administrador que está codificado en base 64.

Para verificar la creación del namespace de Argo CD es necesario ejecutar el siguiente comando:


```
$ kubectl get ns
```

Figura 25. Ejecución del comando: `kubectl get ns`.

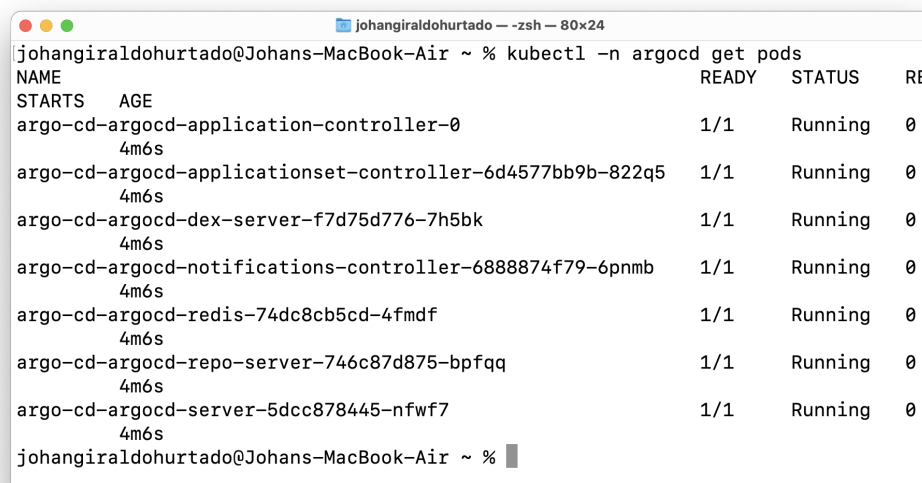


Fuente: elaboración propia.

Para verificar el estado de los pods en el namespace de Argo CD dentro del cluster de *kubernetes* es necesario ejecutar el siguiente comando:

```
$ kubectl -n argocd get pods
```

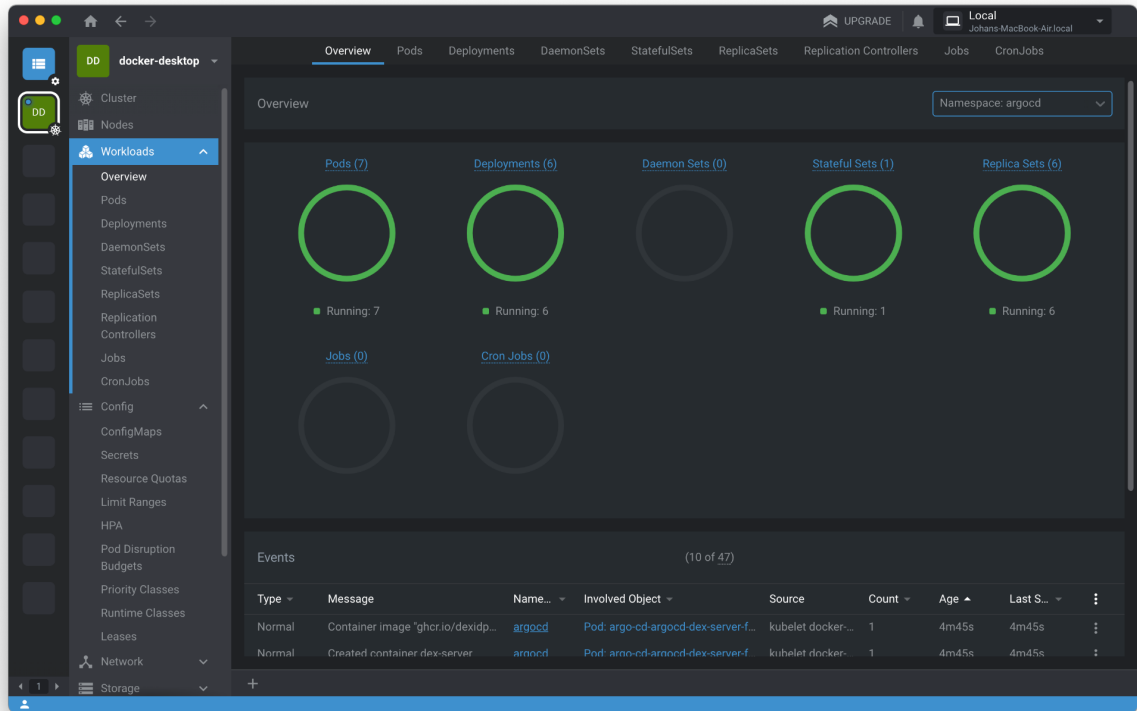
Figura 26. Ejecución del comando: `kubectl -n argocd get pods`.



Fuente: elaboración propia.

Como se puede observar en la **Figura 26**, los Pods se crearon correctamente, todos los Pods se encuentran en el estado de ejecución y no tienen ningún reinicio.

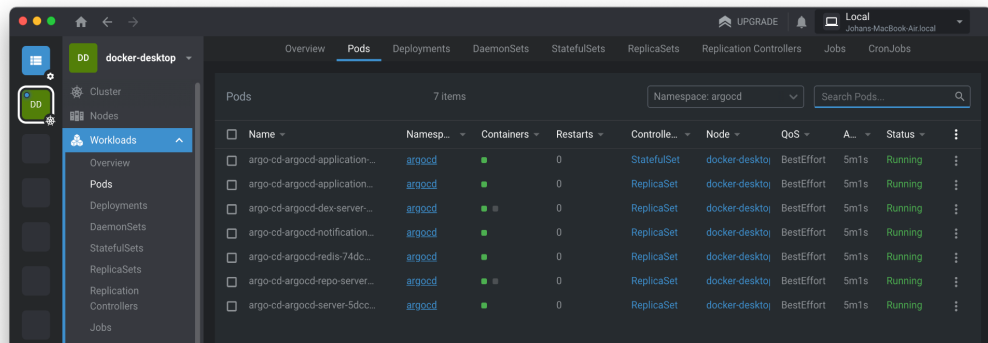
Figura 27. Imagen de la aplicación Lens - Workloads - Overview.



Fuente: elaboración propia.

Otra forma de verificar que todos los elementos del Helm Chart de la herramienta Argo CD fueron desplegados es por medio de la aplicación lens, como se muestra en la **Figura 27**. En donde se selecciona el namespace de argocd y se puede observar la vista general de los elementos creados.

Figura 28. Imagen de la aplicación Lens - Workloads - Pods.



Fuente: elaboración propia.

La aplicación de escritorio de *lens*, permite seleccionar cualquier objeto de kubernetes para realizar su respectiva inspección y comprobar su funcionamiento, en la **Figura 28** se puede observar el estado de los Pods que fueron creados en el momento de instalar el helm chart de la herramienta Argo CD.

4.3.2. Ejecutar la guía de despliegue a través de la implementación de un prototipo funcional de la herramienta GitOps seleccionada

En un panorama tecnológico en constante evolución, la implementación de aplicaciones se ha convertido en un proceso crítico para el éxito de las organizaciones. En este contexto, GitOps emerge como una metodología disruptiva que redefine la forma en que las aplicaciones se despliegan y gestionan. A través de la integración estrecha entre las capacidades de Git y la infraestructura como código, GitOps ofrece una nueva perspectiva que promete agilidad, seguridad y control en todo el ciclo de vida de las aplicaciones. En este análisis, exploraremos cómo la implementación de GitOps no solo optimiza la administración de aplicaciones, sino que también allana el camino para una colaboración más fluida entre equipos de desarrollo y operaciones, estableciendo una base sólida para la innovación y la excelencia en la entrega de software.

4.3.2.1. Desplegar el prototipo funcional en el ambiente previamente creado

Con el fin de comprobar el funcionamiento de la herramienta Argo CD, se desplegará un prototipo funcional que permite verificar y observar las funcionalidades de esta.

Se tomará un ejemplo de la certificación *GitOps Fundamentals* ofrecida por la compañía *CodeFresh*, la cual busca marcar el comienzo de la revolución GitOps con la primera plataforma empresarial de entrega continua impulsada por Argo. Para ampliar esta información, se sugiere realizar una revisión exhaustiva del contenido disponible en los siguientes enlaces:

- <https://codefresh.io/about/>
- <https://learning.codefresh.io/course/gitops-fundamentals>

El ejemplo que se presenta a continuación, es extraído de la certificación *GitOps Fundamentals*:

Primer despliegue

Después de efectuar la instalación de Argo CD, debe iniciar sesión desde la interfaz gráfica. Por medio del siguiente enlace:

- <https://localhost:8080/>

Como se menciona anteriormente en la **Figura 24**, es necesario realizar un forwarding del puerto a través del siguiente comando:

```
$ kubectl port-forward service/argo-cd-argocd-server -n argocd 8080:443
```

Figura 29. Ejecución del comando que permite hacer forwarding del puerto.

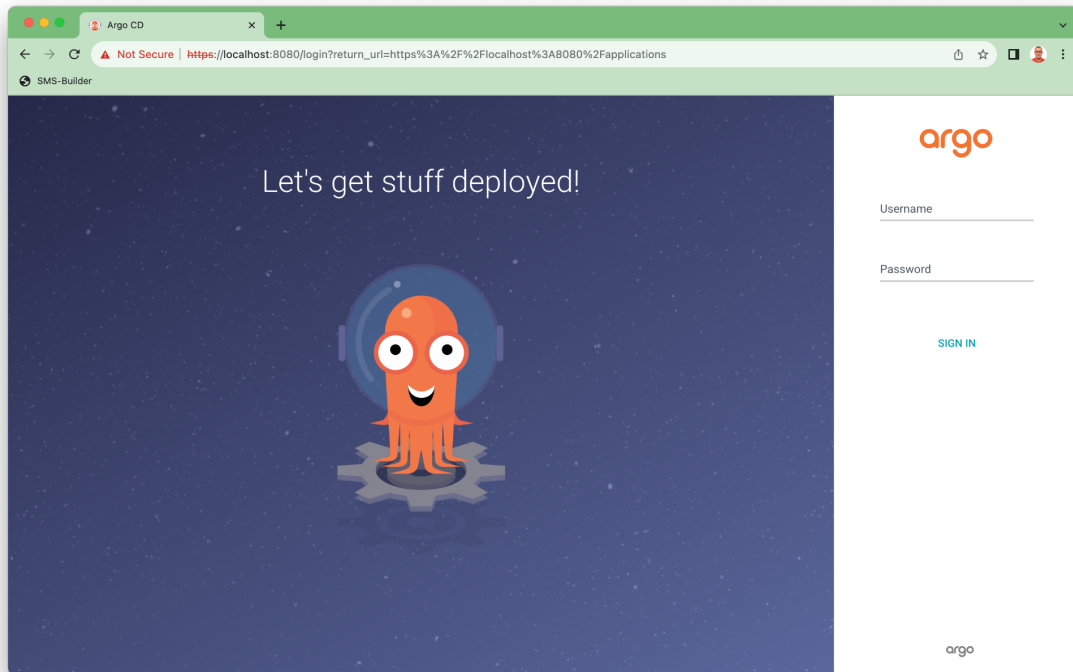


```
johangiraldohurtado@Johans-MacBook-Air ~ % kubectl port-forward service/argo-cd-argocd-server -n argocd 8080:443
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
```

Fuente: elaboración propia.

Luego de la ejecución del comando de la **Figura 29**, es posible acceder por el navegador web.

Figura 30. Interfaz web de la herramienta Argo CD.

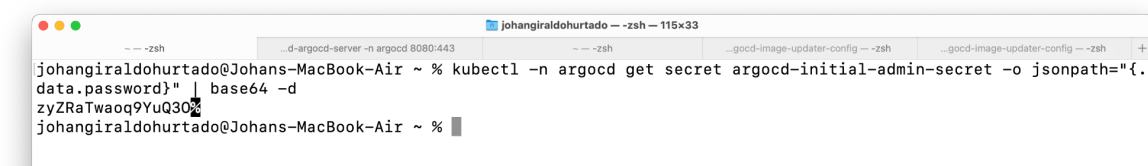


Fuente: elaboración propia.

El usuario para acceder a la herramienta es **admin** y la contraseña se debe obtener de un baúl de secretos y decodificarlo por medio del siguiente comando:

```
$ kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}" | base64 -d
```

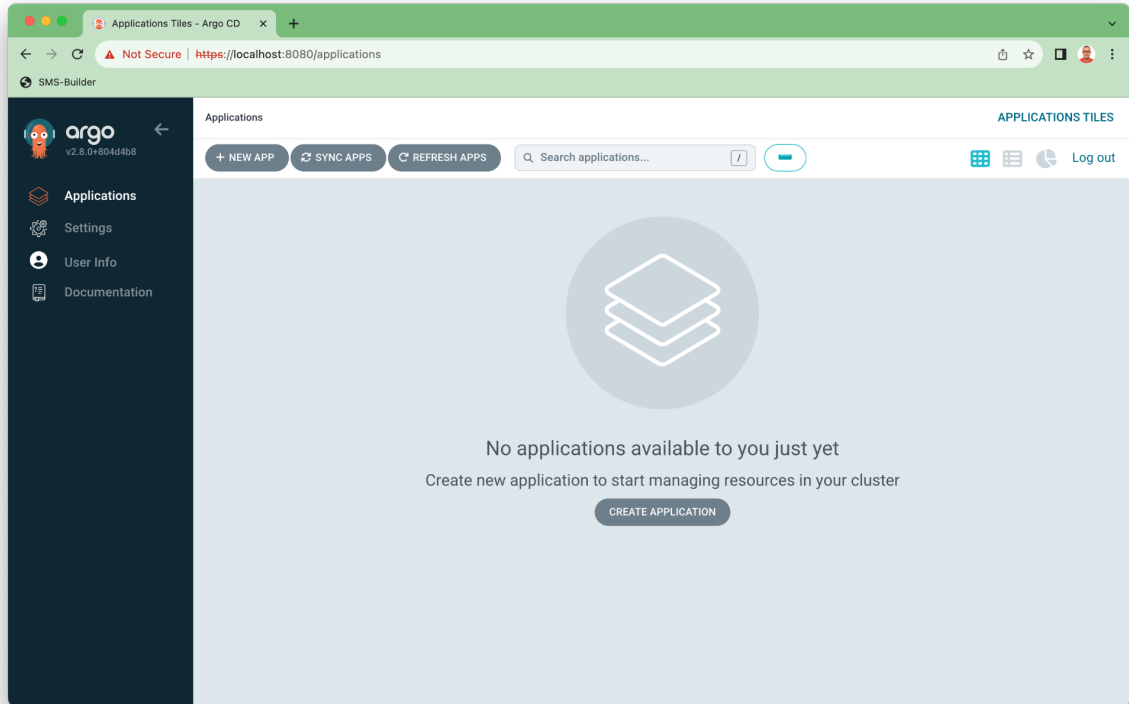
Figura 31. Ejecución del comando `kubectl -n argocd get secret argocd-initial-admin-secret`.



Fuente: elaboración propia.

En este momento, se inicia sesión con el usuario **admin** y la contraseña **zyZRaTwa0q9YuQ30**.

Figura 32. Inicio de sesión en la herramienta Argo CD.



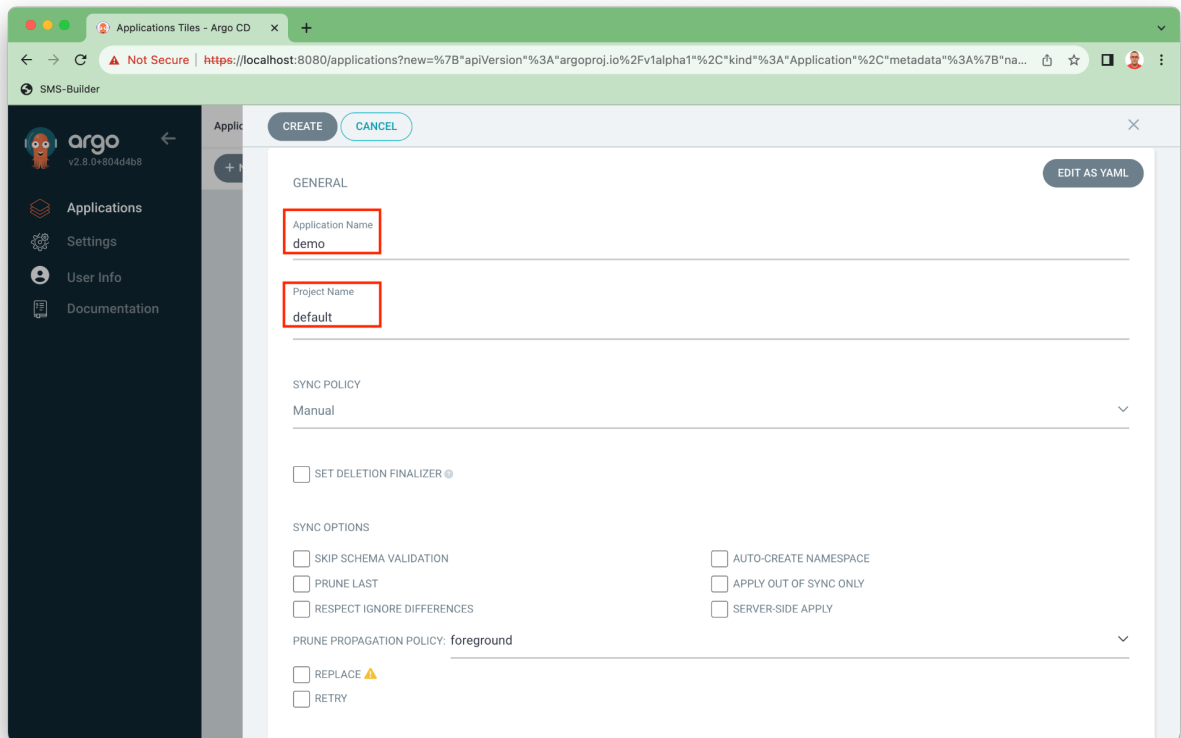
Fuente: elaboración propia.

La interfaz de usuario comienza vacía porque no se implementa nada en el clúster. Haga clic en el botón "Nueva aplicación" en la parte superior izquierda y complete los siguientes detalles:

- nombre de la aplicación: **demo**
- proyecto: **default**
- URL del repositorio: **https://github.com/codefresh-contrib/gitops-certification-examples**
- ruta: **./simple-app**
- Clúster: **https://kubernetes.default.svc** (este es el mismo clúster donde está instalado ArgoCD)
- Espacio de nombres: **default**

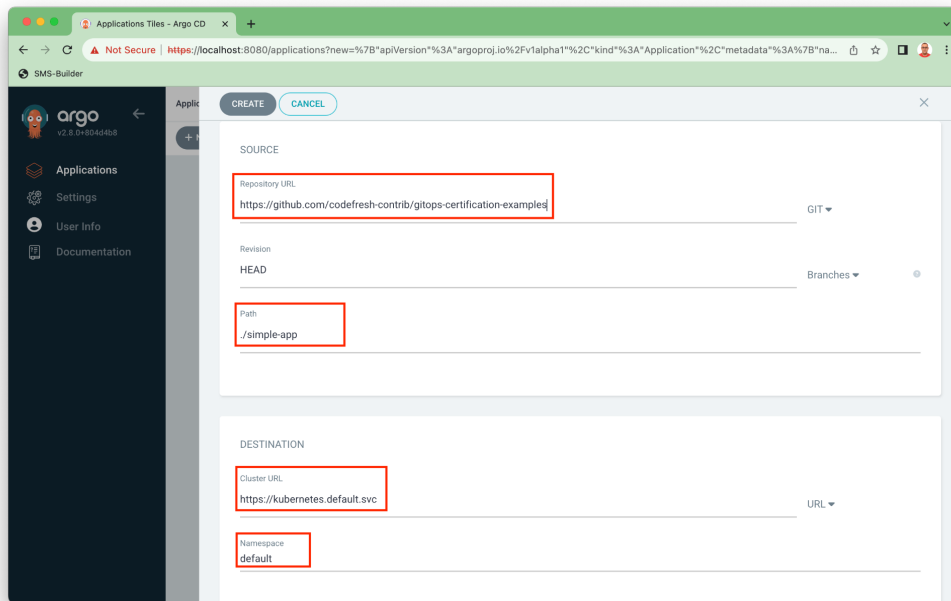
Deje todos los demás valores vacíos o con selecciones predeterminadas. Finalmente haga clic en el botón Crear. La entrada de la aplicación aparecerá en el panel principal.

Figura 33. Creación del primer despliegue en Argo CD - parte 1.



Fuente: elaboración propia.

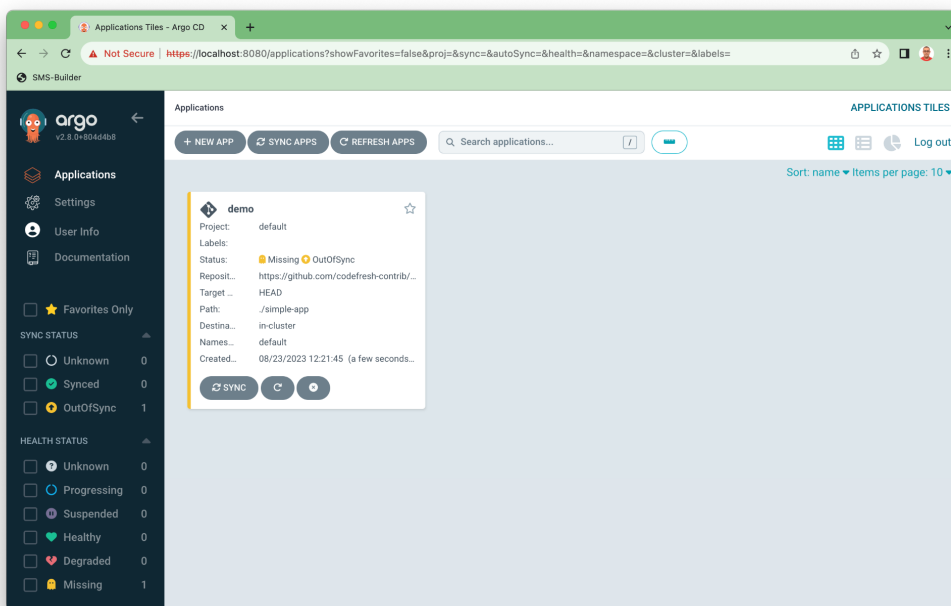
Figura 34. Creación del primer despliegue en Argo CD - parte 2.



Fuente: elaboración propia.

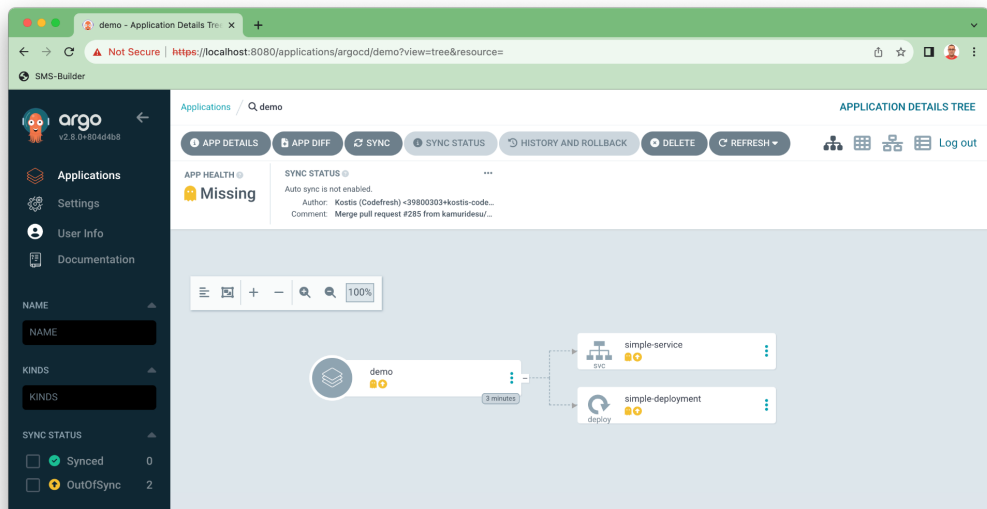
Luego de ingresar los datos se debe ver lo siguiente:

Figura 35. Despliegue del prototipo funcional - parte 1.



Fuente: elaboración propia.

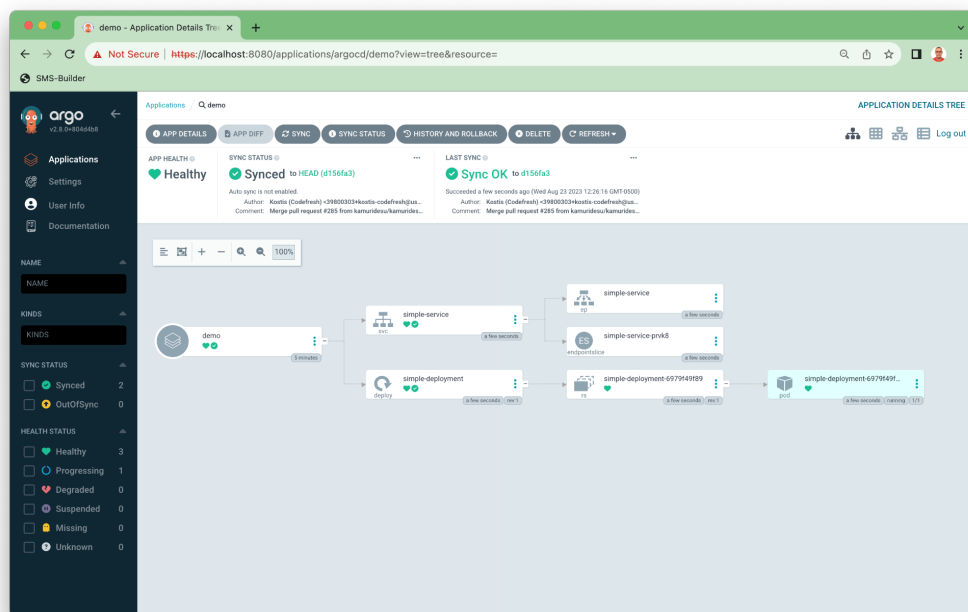
Figura 36. Despliegue del prototipo funcional - parte 2.



Fuente: elaboración propia.

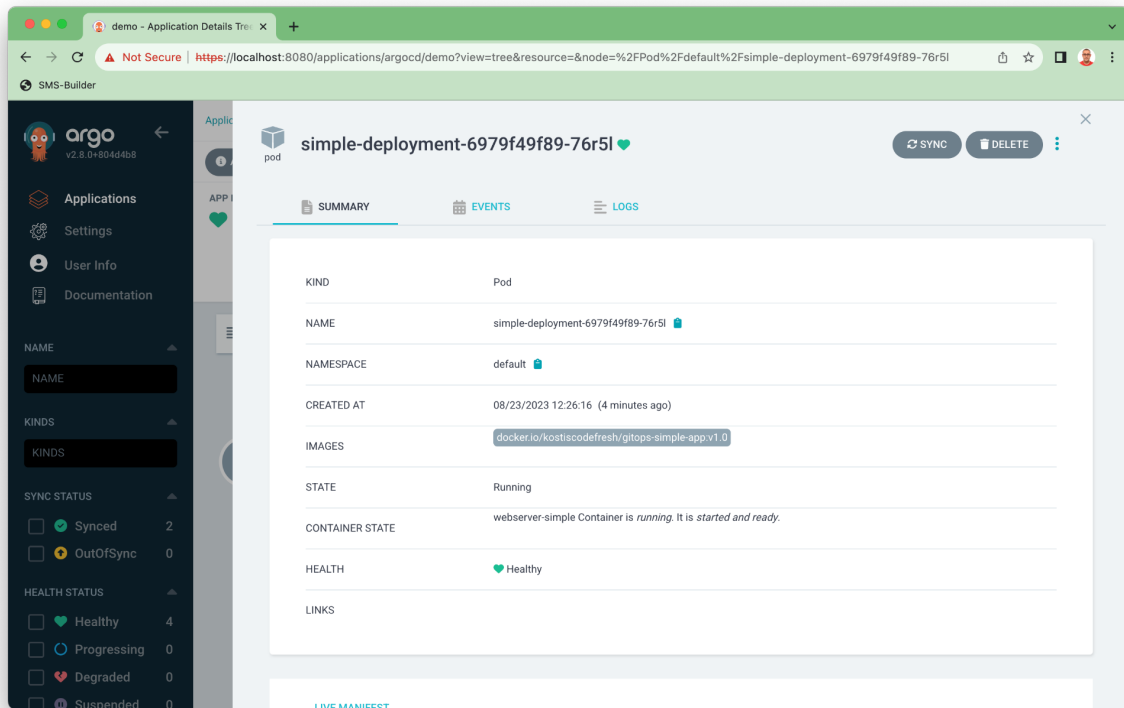
Ahora sincronice la aplicación para asegurarse de que esté implementada. Debería observar lo siguiente:

Figura 37. Ciclo de reconciliación ejecutado manualmente por el usuario.



Fuente: elaboración propia.

Figura 38. Verificación del despliegue de la aplicación por medio del ciclo de reconciliación.

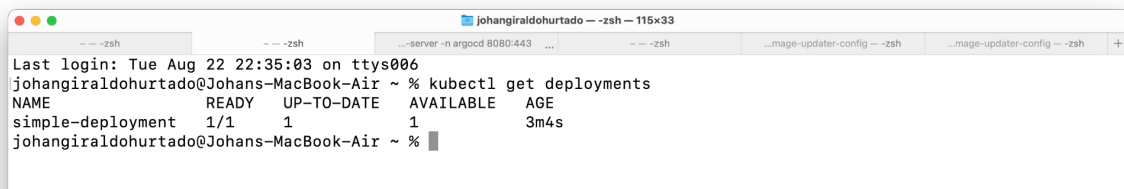


Fuente: elaboración propia.

Es posible verificar la aplicación desde la CLI, ejecutando el siguiente comando:

```
$ kubectl get deployments
```

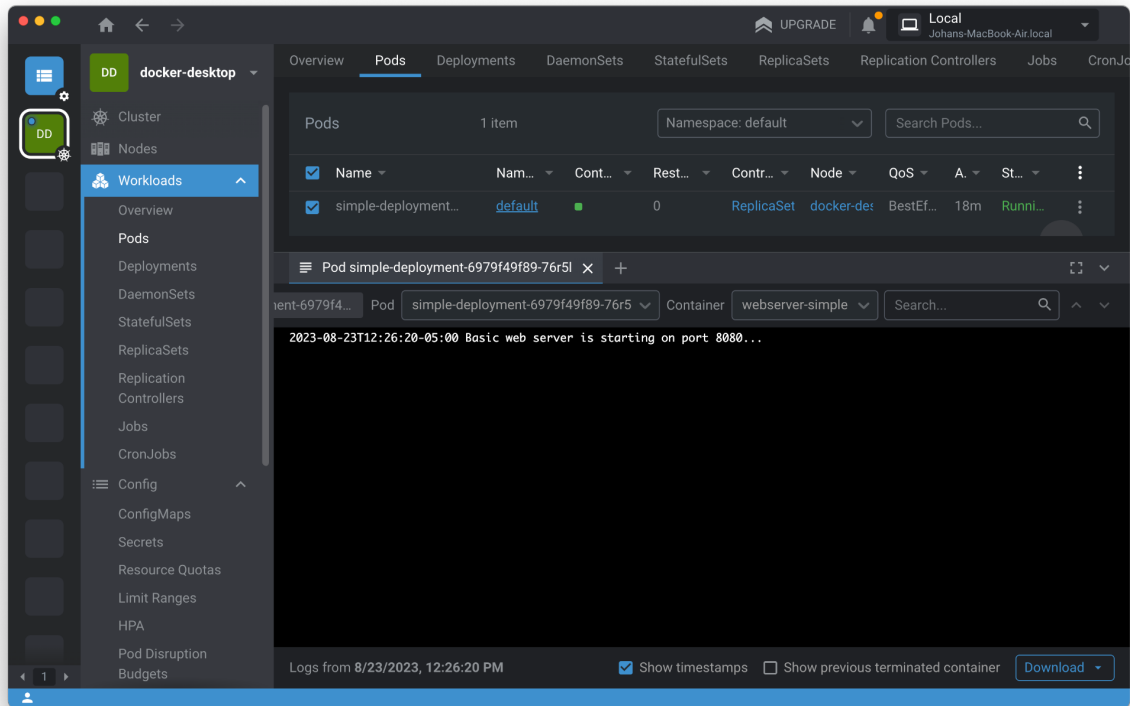
Figura 39. Verificación del despliegue de la aplicación por la CLI.



Fuente: elaboración propia.

Por otro lado también es posible verificar el estado del despliegue de la aplicación por medio de la aplicación de escritorio *lens*.

Figura 40. Verificación del despliegue de la aplicación por medio de *lens*.

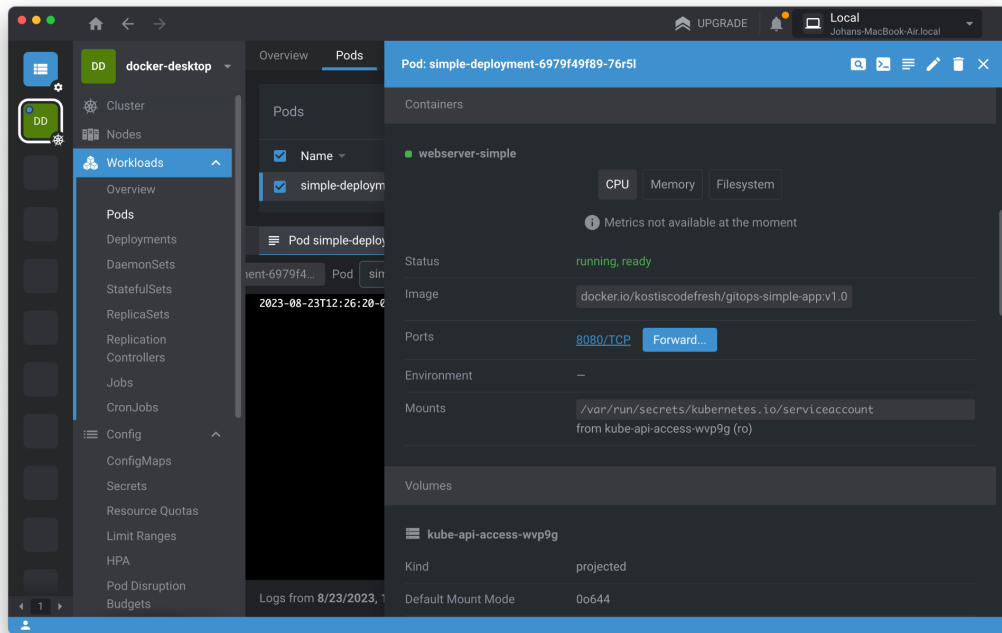


Fuente: elaboración propia.

La aplicación de escritorio *lens* también permite realizar operaciones a través de la GUI en el cluster de *kubernetes*, que normalmente cuestan un poco de trabajo realizarlas por medio de la consola. En este caso se realizará el forwarding del puerto para poder observar el servicio web que se desplegó.

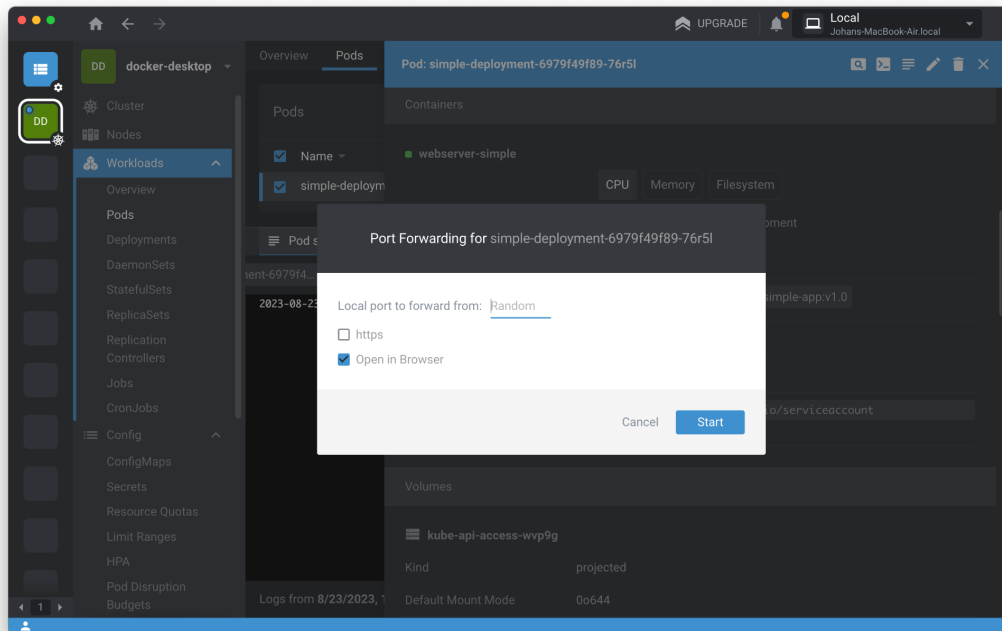
Para realizar lo anteriormente mencionado es necesario seleccionar el **Pod** del **Deployment**, buscar en el menú desplegable la sección **containers**, la opción **ports** y presionar el botón **forward**. Este procedimiento se ilustra en las Figuras **41**, **42** y **43**:

Figura 41. Selección del pod desplegado por medio de lens.



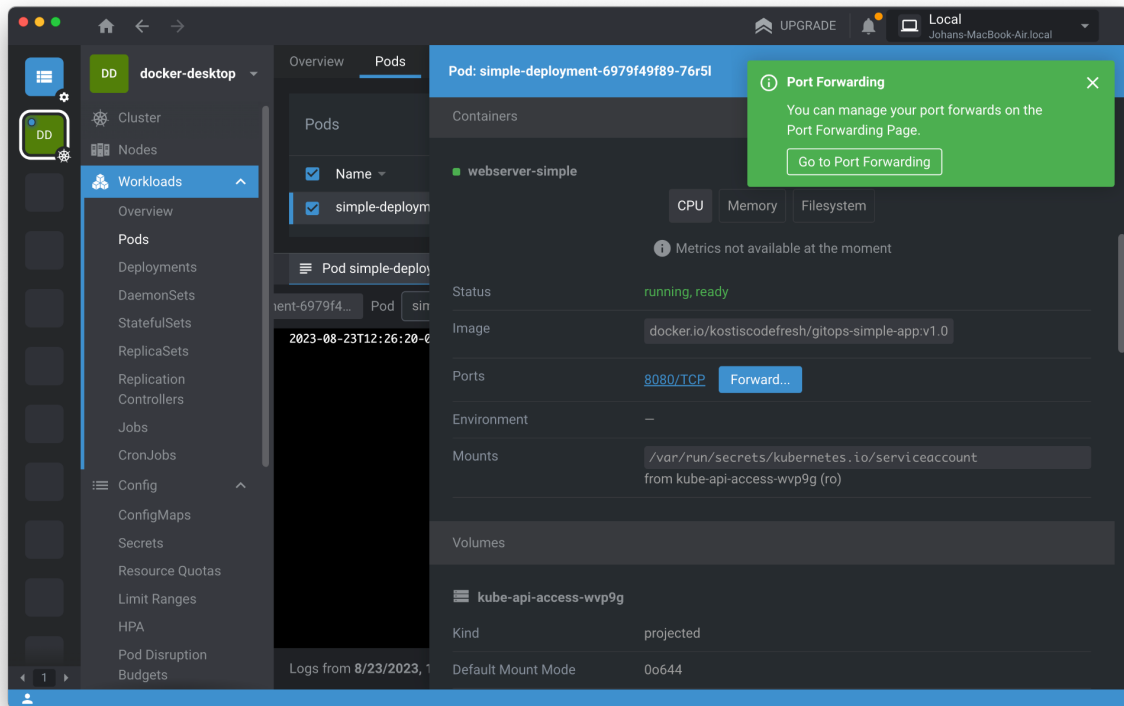
Fuente: elaboración propia.

Figura 42. Selección del puerto para realizar el forwarding.



Fuente: elaboración propia.

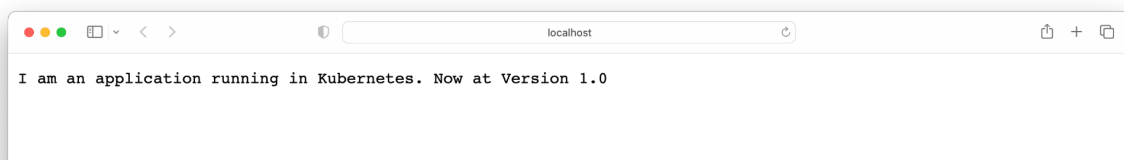
Figura 43. Mensaje de verificación del forwarding.



Fuente: elaboración propia.

Por último, luego de presionar el botón **forward** y confirmar el puerto, se abre automáticamente el navegador web y se puede observar el servicio web desplegado. Como se muestra en la **Figura 44**.

Figura 44. Servicio web desplegado a través de un deployment de kubernetes por medio de Argo CD.



Fuente: elaboración propia.

4.3.2.2. Cumplir las pruebas indicadas en la guía de despliegue

Hasta el momento se ha realizado la instalación y configuración de la herramienta que implementa la metodología GitOps. Además se realizó el despliegue del prototipo funcional que permite demostrar las funcionalidades de la herramienta. Es importante destacar que hasta este punto se ha realizado el proceso de despliegue continuo; es hora de cumplir con la prueba más importante la cual fue mencionada en la sección **4.2.1.4**, la cual corresponde a modificar el despliegue del prototipo funcional para que pueda incluir la fase de integración continua por medio del objeto Argo CD Image Updater.

Para realizar este proceso es necesario recordar el manifiesto YAML, que se indico en la sección de configuración **4.2.1.4**.

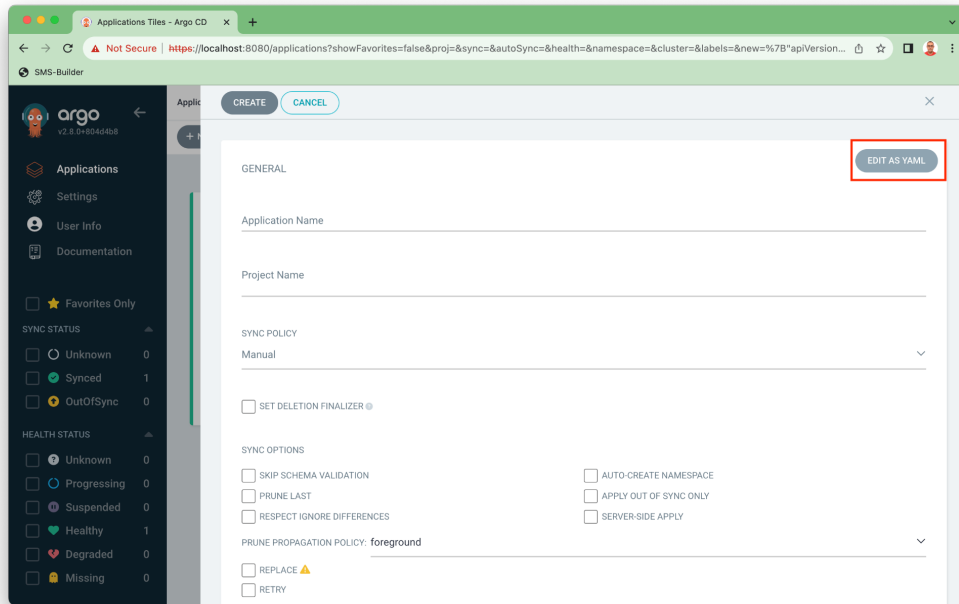
```

apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  annotations:
    argocd-image-updater.argoproj.io/write-back-method: argocd
    argocd-image-updater.argoproj.io/image-list:
myalias=docker.io/kostiscodefresh/gitops-simple-app
    argocd-image-updater.argoproj.io/myalias.update-strategy: semver
    argocd-image-updater.argoproj.io/myalias.force-update: "true"
  name: 'image-updater'
  namespace: argocd
spec:
  destination:
    server: 'https://kubernetes.default.svc'
    namespace: default
  source:
    repoURL:
'https://github.com/codefresh-contrib/gitops-certification-examples'
    path: ./helm-app
    targetRevision: HEAD
    sources: []
    project: 'default'

```

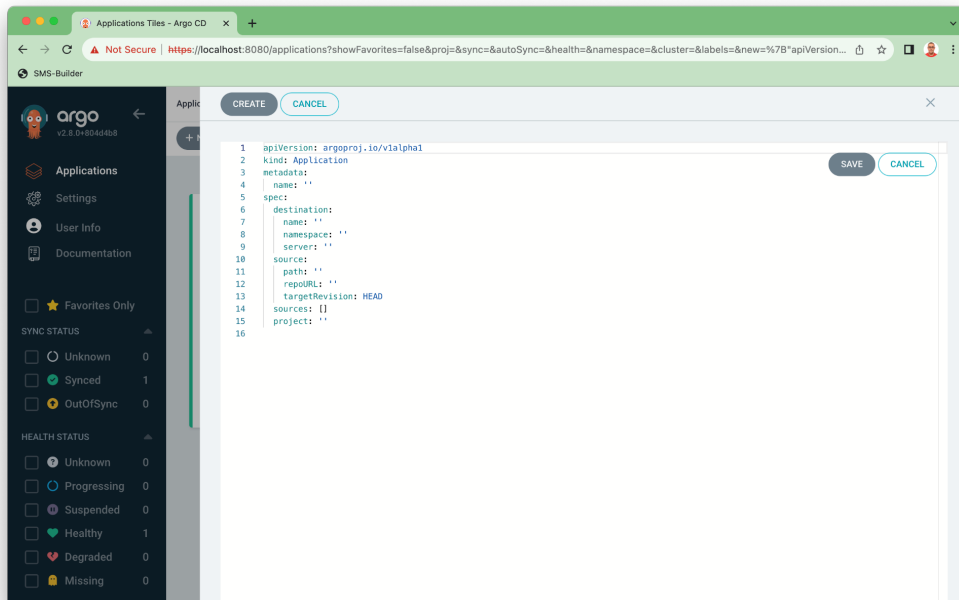
Por medio de la interfaz gráfica de la herramienta Argo CD, es posible crear el despliegue de la aplicación, como se muestra en la sección **4.3.2.1**. Pero en este caso no se ingresarán los datos por medio del formulario web, lo que se pretende es ingresar todo el manifiesto YAML, por la sección **EDIT AS YAML** de la sección **NEW APP**. Este procedimiento se ilustra en las **Figuras 45, 46, 47 y 48**:

Figura 45. Desplegar el prototipo funcional por medio de un archivo YAML.



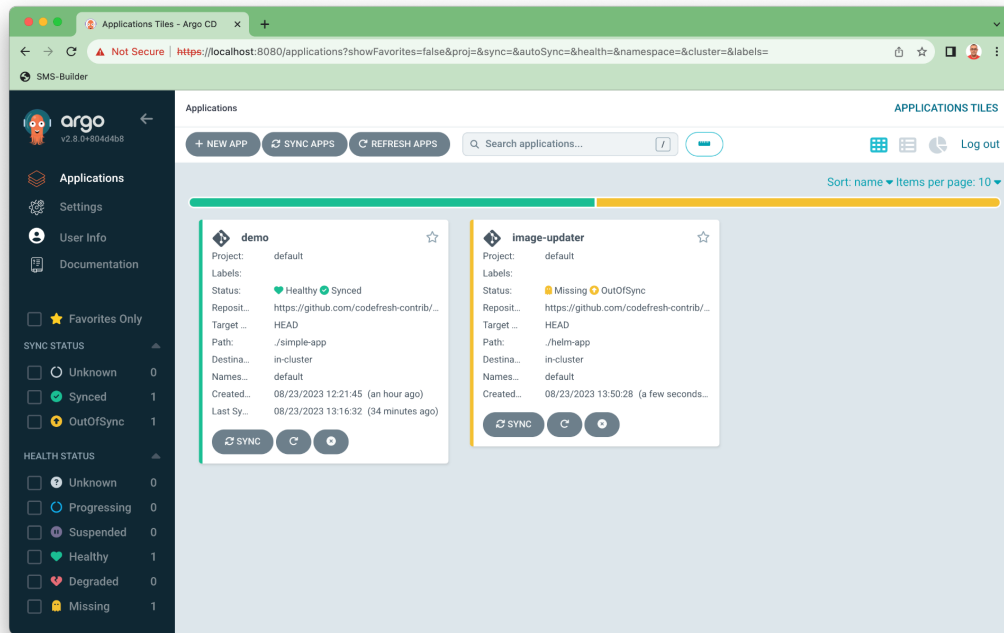
Fuente: elaboración propia.

Figura 46. Desplegar el prototipo funcional por medio de un archivo YAML.



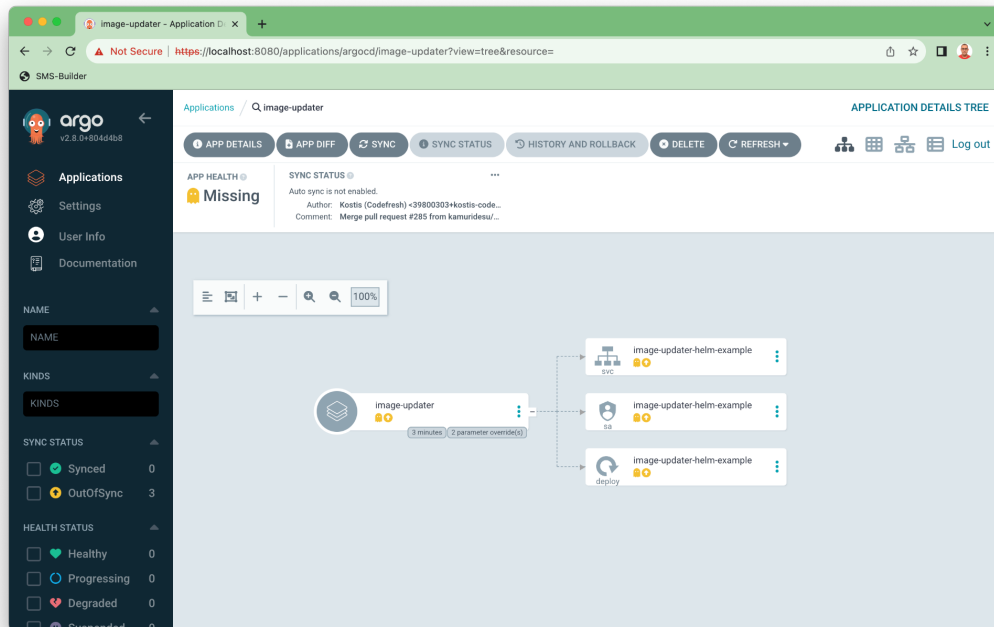
Fuente: elaboración propia.

Figura 47. Desplegar el prototipo funcional por medio de un archivo YAML.



Fuente: elaboración propia.

Figura 48. Desplegar el prototipo funcional por medio de un archivo YAML.

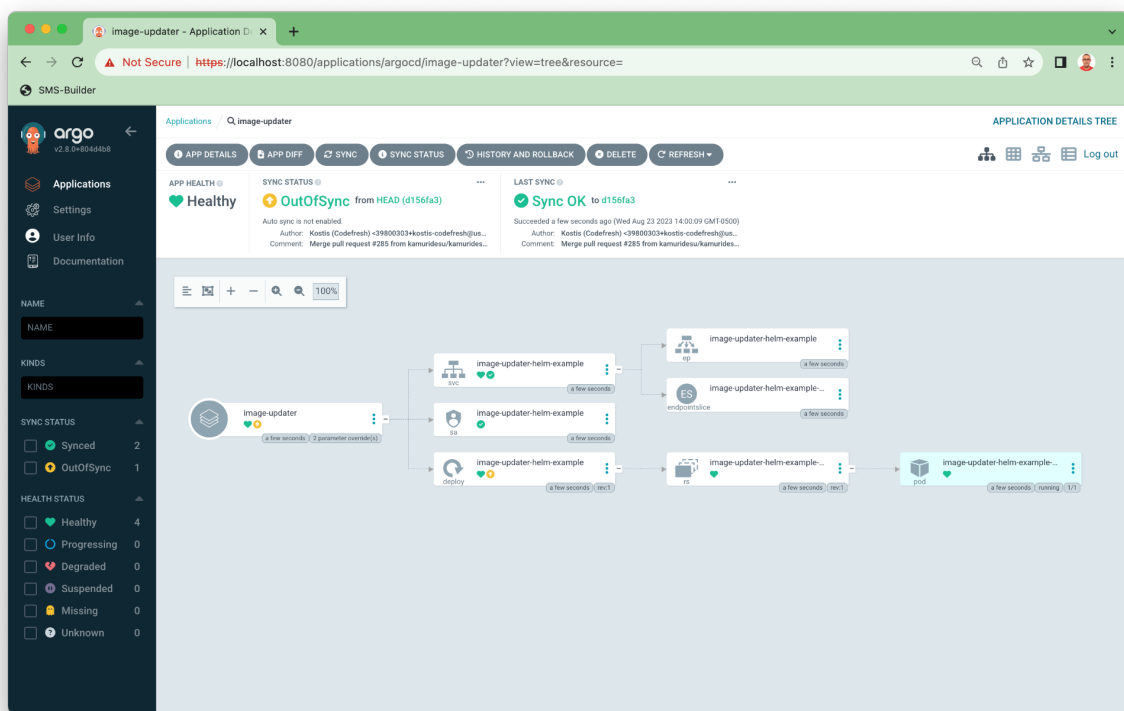


Fuente: elaboración propia.

El despliegue de la aplicación se encuentra en color amarillo, porque la herramienta Argo CD, no ha realizado el ciclo de reconciliación, este se puede ejecutar automáticamente, determinando un periodo de tiempo en minutos, pero en este caso practico es necesario ejecutar el proceso manualmente para ver los cambios realizados y como la herramienta los presenta a traves de la interfaz gráfica.

Se procede a realizar el proceso de sincronización entre el repositorio de Git y el estado deseado del cluster en *kubernetes*.

Figura 49. Sincronización del repositorio de Git y el cluster de Kubernetes.



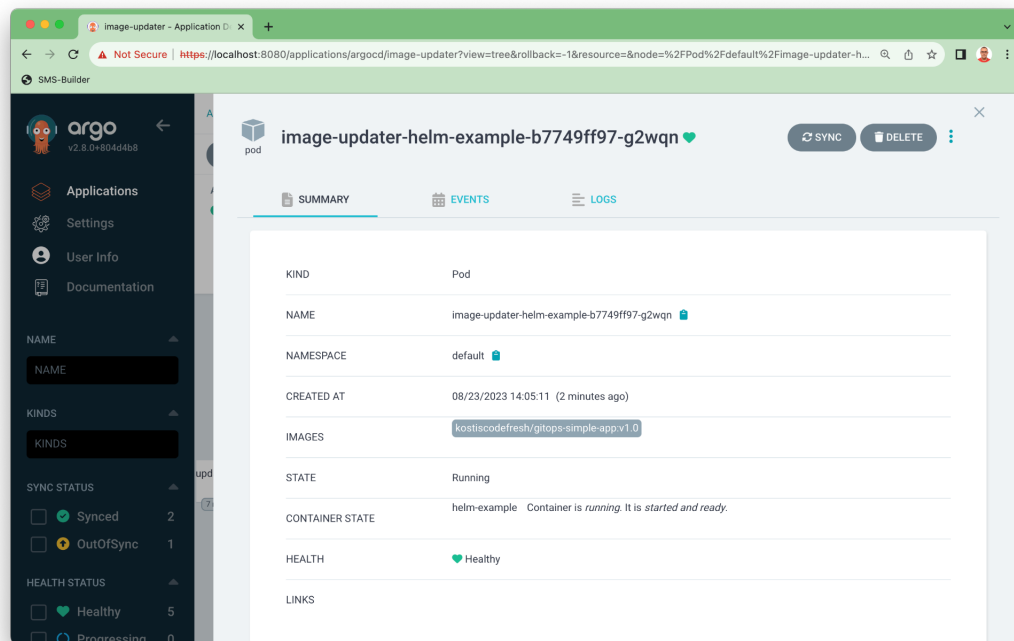
Fuente: elaboración propia.

Luego de realizar el proceso de sincronización, se puede observar en la interfaz gráfica de Argo CD, que el objeto *Deployment* de *kubernetes*, presenta una alerta amarilla, en este punto, se intuye que el objeto **Argo CD Image Updater** se encuentra en funcionamiento,

puesto que esa advertencia es para informar que ha encontrado una nueva versión de la imagen que se encuentra en el cluster de *kubernetes*.

Antes de proseguir con cualquier operación se debe realizar la verificación de la versión de la imagen de Docker que se encuentra desplegada en el cluster, para esto podemos ingresar al **Pod** por medio de la interfaz. En este caso la versión que se debe presentar es la **v1.0**.

Figura 50. Revisión de la versión del despliegue.

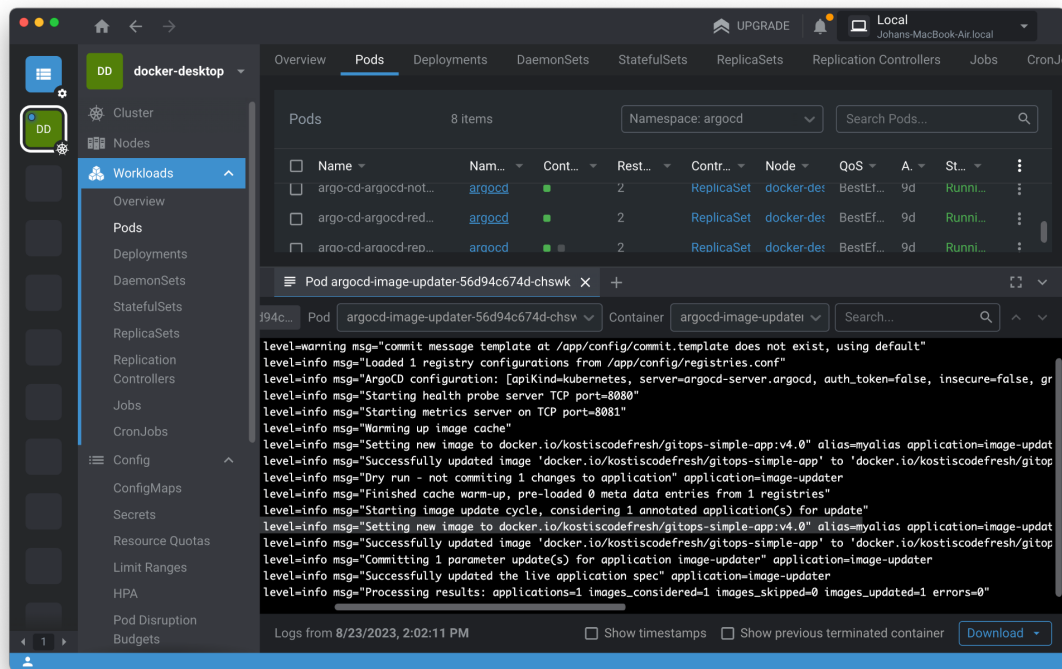


Fuente: elaboración propia.

En este punto es necesario realizar una revisión del controlador **Argo CD Image Updater**, por medio de los instrumentos de *Kubernetes*. Es por esto que se utilizara la herramienta *lens*, para observar los logs de este objeto.

Si el controlador se instaló correctamente, debe presentar mensajes informativos los cuales indican que es posible actualizar la imagen de docker, mostrar cual es la versión de la imagen que se puede alcanzar con la actualización y el repositorio al que ingresa para obtener esta imagen. Este repositorio debe ser el que se configuró en la sección **4.2.14**.

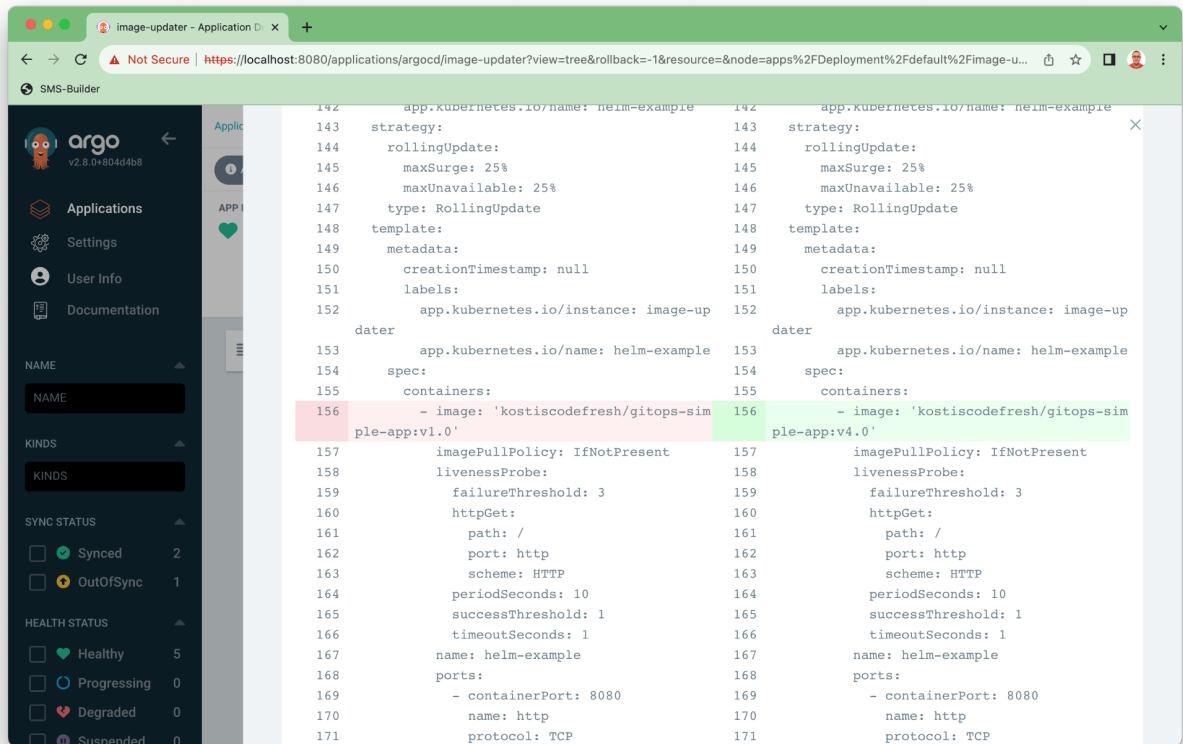
Figura 51. Logs del controlador Argo CD Image Updater.



Fuente: elaboración propia.

También es posible encontrar la diferencia de la versión de la imagen de Docker, el controlador verifica la versión desplegada en el cluster versus la última versión que se encuentra en el repositorio de Git. Para realizar esta validación se ingresa a través de la interfaz gráfica, se presiona el objeto **Deployment** y se dirige hacia la sección Diff, la herramienta se encargará de resaltar las diferencias para revisarlas.

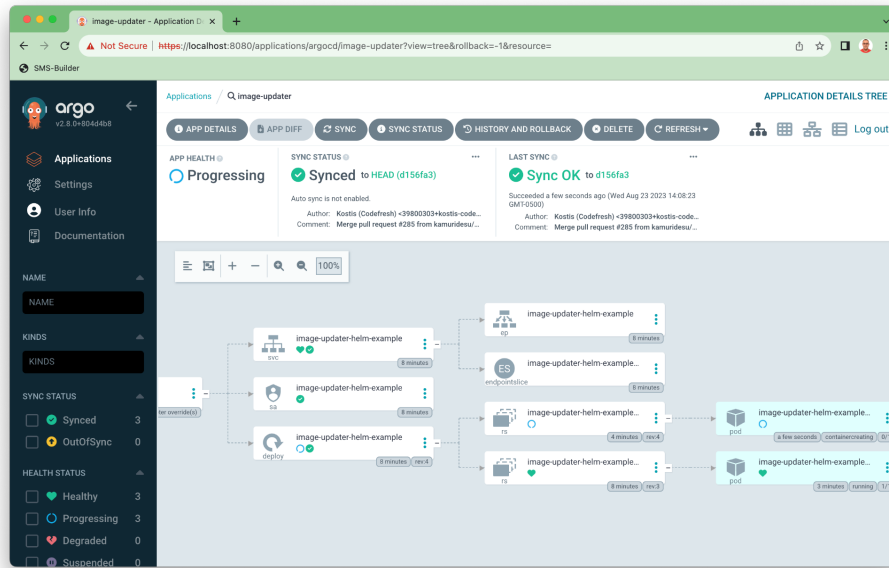
Figura 52. Diferencia entre la versión de la imagen de Docker.



Fuente: elaboración propia.

Luego de verificar la versión deseada del despliegue, se procede a realizar el proceso de sincronización. La interfaz de Argo CD, muestra la construcción del **Pod** con la nueva imagen de Docker.

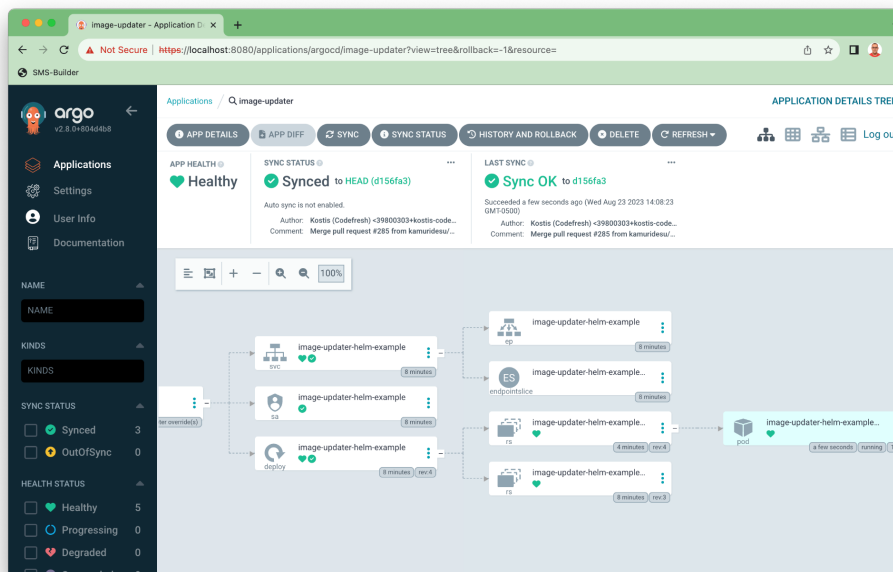
Figura 53. Despliegue con la nueva versión de la imagen de Docker.



Fuente: elaboración propia.

Finalmente es posible percibir cómo se realizó la actualización de la imagen de Docker creando un **Pod** con la nueva versión y destruyendo el de la versión anterior.

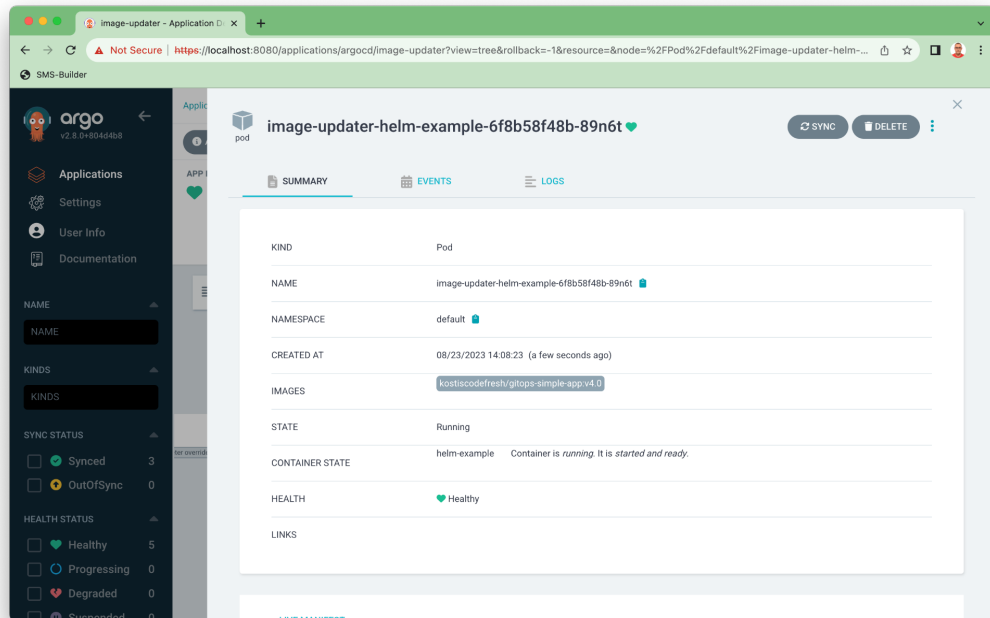
Figura 54. Proceso de destrucción y construcción del otro Pod con la nueva versión.



Fuente: elaboración propia.

Para terminar la demostración es necesario verificar la nueva versión de la imagen de Docker, en este caso la nueva versión es la versión **v4.0**.

Figura 55. Revisión de la nueva versión del despliegue.



Fuente: elaboración propia.

4.3.2.3. Realizar el informe de los resultados obtenidos en las pruebas

Las pruebas propuestas, permitieron verificar la instalación de la herramienta Argo CD. La primera validación que se realizó fue a través de Helm para verificar el despliegue del Chart; en la **Figura 23** podemos observar el Helm Chart de Argo CD desplegado en el sistema.

Posteriormente se validó el estado del despliegue se encontraba correcto, nuevamente se utilizó la herramienta del helm para rectificar el estado del Helm Chart como se muestra en la **Figura 24**; por medio del comando utilizado para esta validación, se obtiene información relevante para acceder a la herramienta, como por ejemplo el port forwarding que se debe realizar en la máquina y como obtener las credenciales de acceso.

Por otra parte, a nivel de *kubernetes* se realizó la validación de la creación del namespace de Argo CD, por medio de la herramienta Kubectl como se muestra en la **Figura 25** y seguido de esto se validó la ejecución de los Pods que conforman todo el despliegue de la herramienta de Argo CD por medio de Helm Chart como se puede observar en la **Figura 26**.

La creación de todos los objetos a nivel de *Kubernetes* fue verificada, para garantizar primero que los elementos se encontraban y posteriormente que el conjunto de todos estos objetos, representan el despliegue de la herramienta Argo CD, la cual fue usada en el trabajo de fin de Master para desplegar prototipos funcionales; Los cuales permitieron comprobar la facilidad de uso de los manifiestos de *kubernetes* y como la git permite versionar toda la infraestructura de un servicio para ser representada en un cluster de *kubernetes*.

5. Conclusiones y trabajo futuro

5.1. Conclusiones

Al término de este trabajo de fin de Máster, se concluye que existe una falta de proliferación de estudios formales alrededor de las herramientas que implementan la metodología GitOps, existe mucha documentación gris (artículos no formales, publicaciones en sitios web, documentación oficial de industria, etc.) de las herramientas, pero no una base científica que examine detalladamente las características de estas por medio de procesos formales o métodos científicos.

Se elaboró un mapeo sistemático de las herramientas que implementa la metodología GitOps para poder determinar cuáles herramientas eran las más destacadas en el mercado, esto permitió realizar la construcción de un cuadro comparativo que evaluaba las características de las herramientas versus las etapas de la filosofía DevOps en las cuales se pudieran ver involucradas.

Se llevó a cabo una evaluación formal a través de la metodología *DAR*, dicha evaluación permitió eliminar la subjetividad y/o preferencia por cualquier herramienta, por lo tanto se establecieron criterios y restricciones con diferentes valores numéricos, los cuales fueron claves para calificar las herramientas y seleccionar la herramienta GitOps más destacada; a la

fecha la herramienta que logró cumplir los aspectos que se tuvieron en cuenta (desarrollo, documentación y tiempo de vida) fue Argo CD.

Se construyó una guía de despliegue teniendo en cuenta las recomendaciones de la literatura, es decir, se tomaron en cuenta ejemplos de libros, certificaciones, la misma documentación de la herramienta para construir una guía que se pudiera implementar con facilidad, a sí mismo que el lector pueda reproducir con el tiempo la ejecución de esta y los resultados obtenidos sean iguales o muy parecidos.

Se tuvo en cuenta, la revolución que generó la era de los contenedores y como impacto en el desarrollo de software y la gestión de la configuración, es por eso que la implementación del prototipo funcional fue por medio de un repositorio público de GitHub (<https://github.com/codefresh-contrib/gitops-certification-examples>), el cual posee los materiales necesarios para reproducir todos los elementos que se desplegaron en el cluster de *Kubernetes*; dichos manifiestos se encuentran versionados y hasta que la empresa *Codefresh* mantenga este repositorio, sera posible reproducir todas las pruebas desplegadas por medio de GitOps en la herramienta Argo CD.

GitOps es un tema del cual no se habla mucho en la comunidad hispana, este catálogo de herramientas que implementan la metodología GitOps busca dar el primer paso a profesionales de TI para que se acerquen a estas nuevas herramientas.

Para concluir este trabajo de final de Máster, se destacó la facilidad de implementar una canalización de integración y despliegue continuo mediante una de las herramientas que adopta la metodología GitOps. Es crucial tener en cuenta esto, ya que muchas veces los profesionales están familiarizados con herramientas tradicionales como Jenkins, Ansible y Terraform, y pueden no estar al tanto de las nuevas opciones que ofrecen una implementación más fluida y simplificada, reduciendo la necesidad de configuraciones complicadas.

5.2. Líneas de trabajo futuro

Como trabajo futuro de este trabajo de fin de Máster se contempla lo siguiente:

- Realizar la implementación de las otras herramientas que implementan la metodología GitOps siguiendo la guía de despliegue propuesta.
- Propagar la metodología GitOps como el proceso de construcción y entrega de software en las empresas de TI.
- Invitar a los profesionales de TI a generar conocimiento científico, evaluar las metodologías y procesos que se implementan en las compañías de manera formal, usando artefactos e instrumentos que permitan eliminar la subjetividad.
- Invitar a los profesionales de TI a compartir las experiencias en el uso de las nuevas tecnologías, para difundir sus características y las ventajas de implementar herramientas como estas.

5.3. Cumplimiento de Objetivos

Esta sección del documento presenta el resumen del cumplimiento de objetivos alcanzados en el presente Trabajo Final de Máster. Ver Tabla 7.

Tabla 7. Resumen de cumplimiento de objetivos.

Objetivo específico 1 : Caracterizar herramientas que implementan la metodología GitOps tomando como referencia las etapas del ciclo de vida DevOps.	
Detalle de cumplimiento	Observaciones
Ver Sección 4. Desarrollo específico de la contribución	<p>Paso 1: Mapeo sistemático de herramientas que implementan la metodología GitOps (sección 4.1.1).</p> <p>Paso 2: El insumo primordial para el cumplimiento de este objetivo se realizó en el numeral 4.1.2 - Tabla 2. Cuadro comparativo de las etapas del ciclo DevOps versus las características de la herramienta.</p>

Objetivo específico 2 : Seleccionar la herramienta más relevante para la metodología GitOps, teniendo en cuenta aspectos como el desarrollo, documentación y tiempo de vida.	
Detalle de cumplimiento	Observaciones

Ver Sección 4. Desarrollo específico de la contribución	<p>Se aplicó la metodología DAR para la selección de la herramienta que implementa la metodología (sección 4.1.5, 4.1.5.1, 4.1.5.2, 4.1.5.3)</p> <p>Finalmente, la selección de la herramienta se realiza en la sección 4.1.5.4 Análisis de resultados metodología DAR.</p>
---	---

Objetivo específico 3 : Construir una guía de despliegue de acuerdo a la metodología GitOps para realizar la implementación tecnológica seleccionada.	
Detalle de cumplimiento	Observaciones
Ver Sección 4.2 Descripción del sistema desarrollado / Implementación	Se realizó la construcción de una guía de despliegue en un ambiente que permite volverse a reproducir para obtener los mismos resultados (sección 4.1.2, 4.1.2.1, 4.1.2.2, 4.1.2.3, 4.1.2.4, 4.1.2.5, 4.1.2.6).

Objetivo específico 4 : Evaluar la efectividad de la guía de despliegue a través de la implementación de un prototipo funcional de la herramienta GitOps seleccionada.	
Detalle de cumplimiento	Observaciones
Ver Sección 4.3 Evaluación	Se desplegó un prototipo funcional dentro de la herramienta que permitiera realizar la validación de las pruebas descritas (sección 4.3.2, 4.3.2.1, 4.3.2.2, 4.3.2.3).

Fuente: elaboración propia.

Referencias bibliográficas

- Ali, A. Q., Sultan, A. B. M., Ghani, A. A. A., y Zulzalil, H. (2019). *A Systematic Mapping Study on the Customization Solutions of Software as a Service Applications*. *IEEE Access*, 7, 88196–88217. <https://doi.org/10.1109/ACCESS.2019.2925499>
- Atlassian. (s. f.). *¿Qué es el control de versiones?*. <https://www.atlassian.com/es/git/tutorials/what-is-version-control>
- Atlassian. (s. f.). *¿Qué es la entrega continua?*. <https://www.atlassian.com/es/continuous-delivery>
- Báez, N. (2021). Repositorios Git Y Orquestadores De Contenedores Kubernetes En La Optimización De La Gestión Empresarial. *Business Innova Sciences*, 2(3). <https://doi.org/10.58720/bis.v2i3.55>
- Beetz, F., & Harrer, S. (2022). GitOps: The Evolution of DevOps? *IEEE Software*, 39(4), 70–75. <https://doi.org/10.1109/MS.2021.3119106>
- Beetz, F., Kammer, A., & Harrer, S. (s. f.). *GitOps Tech*. <https://www.gitops.tech/>
- Candela-Uribe, C. A., Sepúlveda-Rodríguez, L. E., Chavarro-Porras, J. C., Sanabria-Ordoñez, J. A., Garrido, J. L., Rodríguez-Domínguez, C., & Guerrero-Contreras, G. (2022). SMS-Builder: An adaptive software tool for building systematic mapping studies. *SoftwareX*, 17. <https://doi.org/10.1016/j.softx.2021.100935>
- Chacón Cabrera, L. (2022). *Diseño, desarrollo e implementación de una arquitectura GitOps (Git y Kubernetes) que integre infraestructura basada en código desplegadas en plataformas IaaS (Infraestructura como Servicio)*. <https://dspace.ups.edu.ec/bitstream/123456789/22397/1/UPS-CT009712.pdf>
- Davis, C., Appnel, T., Ploski, R., O’Leary, B., & Chia, W. (2021). *GitOps: Infrastructure Automation*. GitLab. <https://about.gitlab.com/why/gitops-infrastructure-automation/>

DevOpsDays. (s. f.). *Página principal*. <https://devopsdays.io/>

Docker. (s. f.). *¿Qué es un contenedor?*. <https://www.docker.com/resources/what-container/>

Ennis, P. (2022, septiembre 15). *Generally Available: Harness GitOps as a Service*. Harness. <https://www.harness.io/blog/generally-available-harness-gitops-as-a-service>

Freshworks. (s. f.). *Equipo de TI: estructura, funciones y procesos*. <https://www.freshworks.com/latam/freshservice/equipo-ti/>

Fowler, M. (2006). *martinfowler.com*. <https://martinfowler.com/articles/continuousIntegration.html>

Garg, S., Pundir, P., Rathee, G., Gupta, P. K., Garg, S., & Ahlawat, S. (2021). On Continuous Integration / Continuous Delivery for Automated Deployment of Machine Learning Models using MLOps. In *Proceedings - 2021 IEEE 4th International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2021* (Issue 4th IEEE International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), pp. 25–28). <https://doi.org/10.1109/AIKE52691.2021.00010>

Gupta, S., Bhatia, M., Memoria, M., & Manani, P. (2022). Prevalence of GitOps, DevOps in Fast CI/CD Cycles. *2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing, COM-IT-CON 2022*, 1, 589–596. <https://doi.org/10.1109/COM-IT-CON54601.2022.9850786>

Hightower, K. [@kelseyhightower]. (2019, 21 de agosto). *GitOps is the best thing since configuration as code* [Tweet]. Twitter. <https://twitter.com/kelseyhightower/status/1164192321891528704?lang=es>

ITential. (s. f.). *ITential to Showcase Infrastructure as Code for Cloud Network Automation at ONUG Fall 2019*.

<https://www.itential.com/news/itential-to-showcase-infrastructure-as-code-for-cloud-network-automation-at-onug-fall-2019/>

Kapelonis, K., & Seligson, H. (s. f.). *code fresh*.
<https://learning.codefresh.io/course/gitops-with-argo>

Kersten, N., Stahnke, M., McCarthy, K., & O'Connell, C. (2021). *State of DevOps Report 2021*.
<https://media.webteam.puppet.com/uploads/2021/07/Puppet-State-of-DevOps-Report-2021.pdf>

Kornstädt, P. (2021, noviembre 30). *Deutsche Telekom rolls out 5G on Kubernetes with Weaveworks*. Deutsche Telekom.
<https://www.telekom.com/en/media/media-information/archive/deutsche-telekom-rolls-out-5g-on-kubernetes-with-weaveworks-641936>

Kubernetes. (2022, Julio 17). *¿Qué es Kubernetes?*.
<https://kubernetes.io/es/docs/concepts/overview/what-is-kubernetes/>

Leiter, Á., Hegyi, A., Kispál, I., Böösy, P., Galambosi, N., & Tar, G. Z. (2023). GitOps and Kubernetes Operator-based Network Function Configuration. *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*, 1–5.
<https://doi.org/10.1109/noms56928.2023.10154212>

López-Viana, R., Díaz, J., & Pérez, J. E. (2022). Continuous Deployment in IoT Edge Computing A GitOps implementation. *Iberian Conference on Information Systems and Technologies, CISTI, 2022-June*, 1–6.
<https://doi.org/10.23919/CISTI54924.2022.9820108>

Maderuelo, F. (2021). Metodología GitOps para despliegue de aplicaciones basadas en microservicios [universidad politecnica de madrid]. In *Archivo Digital UPM*.
<https://oa.upm.es/67264/>

Mell, P., y Grance, T. (2011). *The NIST-National Institute of Standards and Technology-Definition of Cloud Computing*. En NIST Special Publication 800-145 (p. 7).

Morejón, M. (2023, Abril). *Érase una vez Kubernetes*. Leanpub.
<https://leanpub.com/erases-una-vez-kubernetes>

Morejón, M. (2023, Mayo). *Érase una vez Docker*. Leanpub.
<https://leanpub.com/erases-una-vez-docker>

Landscape - Cloud Native Computing Foundation. (s. f.). CNCF. <https://landscape.cncf.io/>

Pittet, S. (s. f.). *Comparación entre implementación continua y entrega continua*. Atlassian.
<https://www.atlassian.com/es/continuous-delivery/software-testing/continuous-deployment>

Platform9. (2021, Octubre 21). *Platform9 announces SaaS GitOps engine to simplify multicloud operations and governance with latest Kubernetes release*.
<https://platform9.com/press/platform9-announces-saas-gitops-engine-to-simplify-multicloud-operations-and-governance-with-latest-kubernetes-release/>

Raleigh, N.C. (2021, Mayo 3). *Red Hat makes DevOps a reality with OpenShift GitOps and OpenShift Pipelines*. Red Hat.
<https://www.redhat.com/en/about/press-releases/red-hat-makes-devops-reality-openshift-gitops-and-openshift-pipelines>

Ramadoni, Utami, E., & Fatta, H. Al. (2021). Analysis on the Use of Declarative and Pull-based Deployment Models on GitOps Using Argo CD. *ICOI ACT 2021 - 4th International Conference on Information and Communications Technology: The Role of AI in Health and Social Revolution in Turbulence Era*, 186–191.
<https://doi.org/10.1109/ICOI ACT53268.2021.9563984>

Rehkopf, M. (s. f.). *¿En qué consiste la integración continua?*. Atlassian.
<https://www.atlassian.com/es/continuous-delivery/continuous-integration>

Rehkopf, M. (s. f.). *Principios de la entrega continua*. Atlassian.
<https://www.atlassian.com/es/continuous-delivery/principles>

- Roche, J. (s. f.). *Scrum: roles y responsabilidades*. Deloitte. <https://www2.deloitte.com/es/es/pages/technology/articles/roles-y-responsabilidades-scrum.html>
- Schalm, D. (2020, julio 22). *Instana becomes first APM solution with GitOps-enabled agent management*. DevOps.com. <https://devops.com/instana-becomes-first-apm-solution-with-gitops-enabled-agent-management/>
- Sepúlveda, L. (2023). *Modelo de referencia para la gestión de recursos y servicios informáticos en ecosistemas tecnológicos de apoyo a la investigación* [Universidad Tecnológica de Pereira]. <https://hdl.handle.net/11059/14573>
- TutorialsPoint. (s. f.). *SDLC Overview*. https://www.tutorialspoint.com/sdlc/sdlc_overview.htm
- Valencia, A. F. (2022). *Tecnología Serverless para la Ejecución de Flujos de Trabajo Científico en Ecosistemas Tecnológicos de Apoyo a la Investigación (ETAI)* [Tesis de maestría, Universidad Tecnológica de Pereira]. <https://acofipapers.org/index.php/eiei/article/view/3207/2267>
- van Rossem, S., Sayadi, B., Rouillet, L., Mimidis, A., Paolino, M., Veitch, P., Berde, B., Labrador, I., Ramos, A., Tavernier, W., Ollora, E., y Soler, J. (2018). *A Vision for the Next Generation Platform-as-a-Service*. 2018 IEEE 5G World Forum (5GWF), 14–19. <https://doi.org/10.1109 / 5GWF.2018.851697>
- Yates, C. (2021, Enero 21). *GitOps has been controversial, but the truth is your organization needs it*. The Next Web. <https://thenextweb.com/news/gitops-has-been-controversial-but-the-truth-is-your-organization-needs-it>
- Weaveworks. (2019, Junio 11). *Automate Kubernetes with GitOps*. Weaveworks. <https://www.weave.works/blog/automate-kubernetes-with-gitops>

Weaveworks. (S. f.). *Guide To GitOps*. Weaveworks.

<https://www.weave.works/technologies/gitops/>

Wikipedia. (s. f.). *Release management*. Recuperado 23 de agosto de 2023 de

https://en.wikipedia.org/wiki/Release_management

Anexos

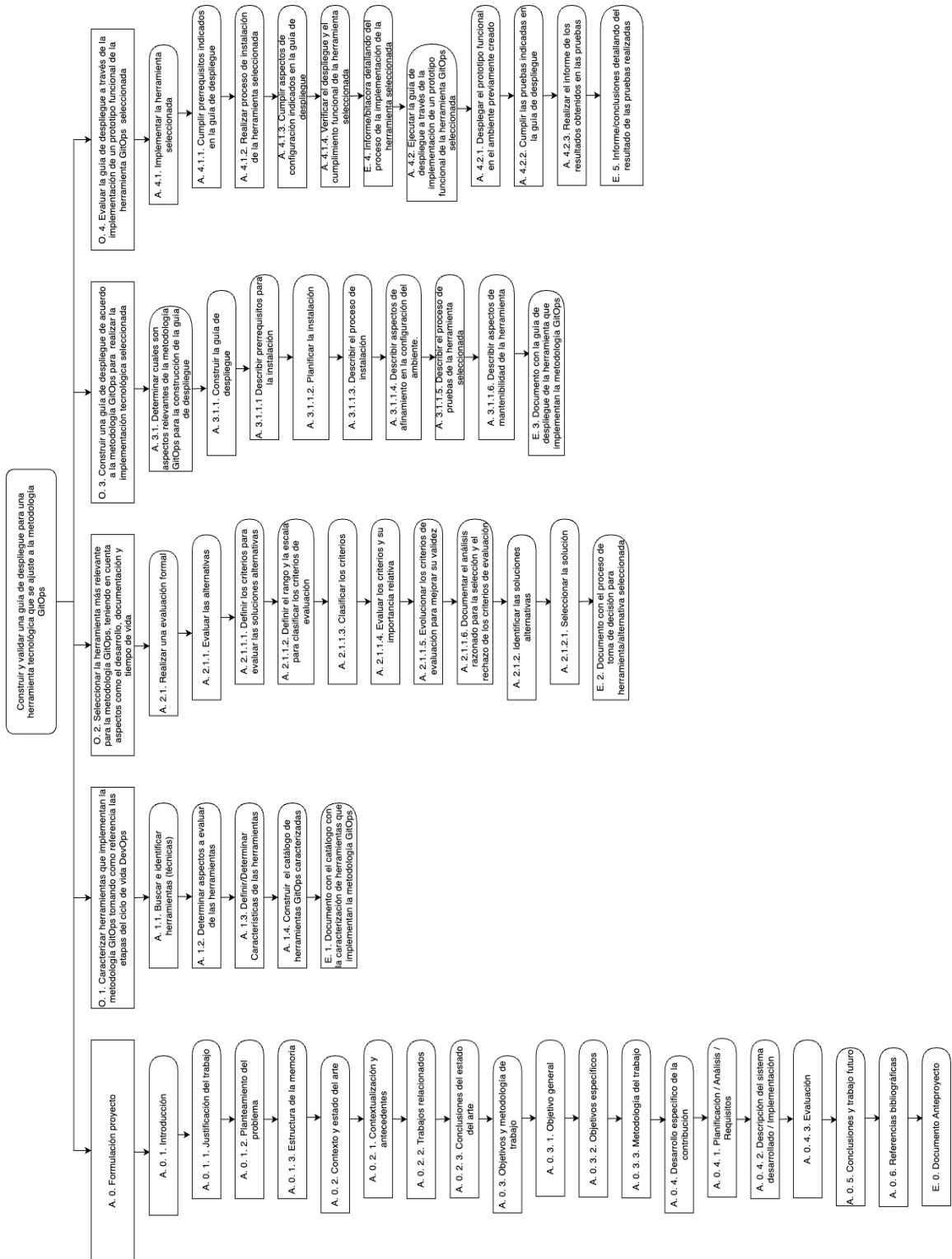
Apéndice

A. Abreviaciones y Acrónimos

CI/CD Integración y despliegue continuo

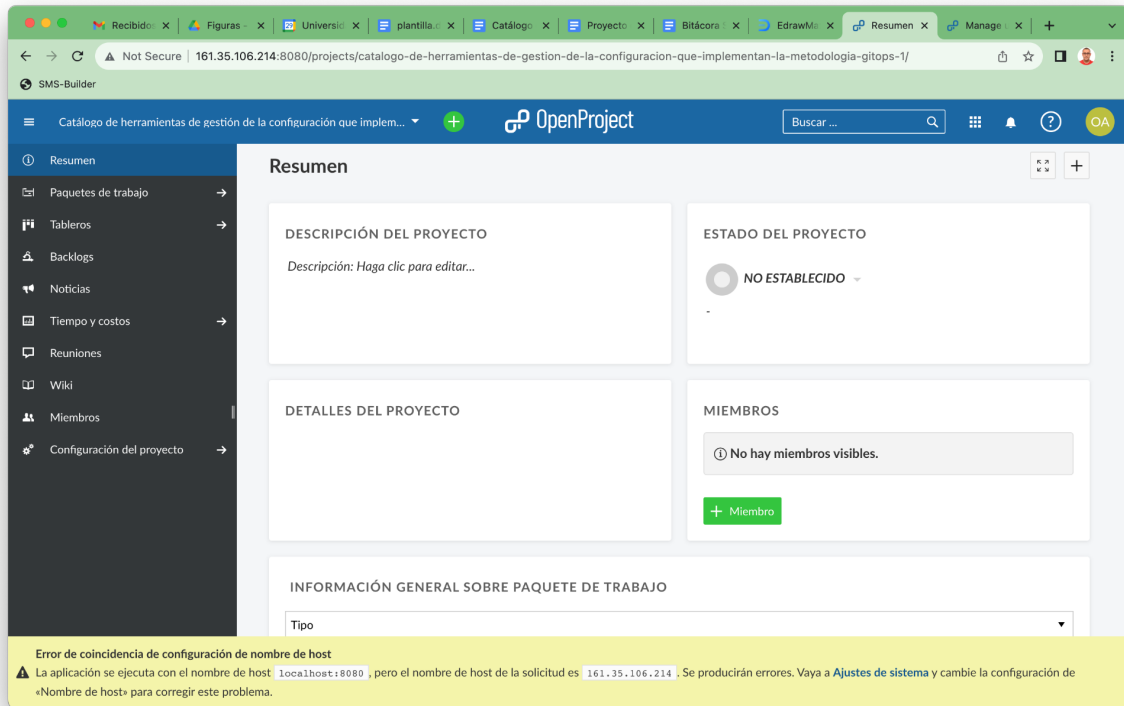
SMS Estudio de mapeo sistemático

B. Estructura de trabajo desglosa



C. Actividades del proyecto trasladadas a un diagrama de Gantt

C.1. Herramienta OpenProject



C.2. Diagrama de Gantt - GitOps



C.3. Diagrama de Gantt - GitOps - Fases

Diagrama de gantt - GitOps

+ Crear
Incluir proyectos 1
Filtro 1
Tabla
Info
Expandir
More

ID ↑	ASUNTO	TIPO	PROGRESO (%)	*
99	Construir y validar una guía de despliegue para una herramienta tecnológica que se ajuste a la metodo...	PHASE	<div style="width: 11%;"></div> 11%	
100	Fase 0 - Documentación	PHASE	<div style="width: 33%;"></div> 33%	
121	Fase 1 - Caracterizar herramientas que implementan la metodología GitOps tomando como refere...	PHASE	<div style="width: 0%;"></div> 0%	
127	Fase 2 - Seleccionar la herramienta más relevante para la metodología GitOps, teniendo en cuenta ...	PHASE	<div style="width: 0%;"></div> 0%	
139	Fase 3 - Construir una guía de despliegue de acuerdo a la metodología GitOps para realizar la impl...	PHASE	<div style="width: 0%;"></div> 0%	
149	Fase 4 - Evaluar la guía de despliegue a través de la implementación de un prototipo funcional de l...	PHASE	<div style="width: 0%;"></div> 0%	

+ Crear un nuevo paquete de trabajo

C.4. Diagrama de Gantt - GitOps - Fase 0

Diagrama de gantt - GitOps - Work

+ Crear
Incluir proyectos 1
Filtro 1
Tabla
Info
Expandir
More

ASUNTO	TIPO	PROGRESO (%)	*
Construir y validar una guía de despliegue para una herramienta tecnológica que se ajuste a la metodología GitOps	PHASE	<div style="width: 11%;"></div> 11%	
Fase 0 - Documentación	PHASE	<div style="width: 33%;"></div> 33%	Info More
A. 0. Formulación proyecto	TASK	<div style="width: 35%;"></div> 35%	
> A. 0. 1. Introducción	TASK	<div style="width: 66%;"></div> 66%	
> A. 0. 2. Contexto y estado del arte	TASK	<div style="width: 0%;"></div> 0%	
> A. 0. 3. Objetivos y metodología de trabajo	TASK	<div style="width: 100%;"></div> 100%	
> A. 0. 4. Desarrollo específico de la contribución	TASK	<div style="width: 0%;"></div> 0%	
A. 0. 5. Conclusiones y trabajo futuro	TASK	<div style="width: 0%;"></div> 0%	
A. 0. 6. Identificar y utilizar referencias bibliográficas	TASK	<div style="width: 0%;"></div> 0%	
E. 0. Documento Anteproyecto	MILESTONE	<div style="width: 0%;"></div> 0%	

C.5. Diagrama de Gantt - GitOps - Fase 1

Diagrama de gantt - GitOps - Work + Crear Incluir proyectos 1 Filtro 1 Tabla Info Expandir More

ASUNTO	TIPO	PROGRESO (%)	*
<ul style="list-style-type: none"> Construir y validar una guía de despliegue para una herramienta tecnológica que se ajuste a la metodología GitOps 	PHASE	<div style="width: 43%;"></div> 43%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Fase 0 - Documentación 	PHASE	<div style="width: 93%;"></div> 93%	Info More
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Fase 1 - Caracterizar herramientas que implementan la metodología GitOps tomando como referencia las etapas del ... 	PHASE	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 1.1. Buscar e identificar herramientas (técnicas) 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 1.2. Determinar aspectos a evaluar de las herramientas 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 1.3. Definir/Determinar Características de las herramientas 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 1.4. Construir el catálogo de herramientas GitOps caracterizadas 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> E. 1. Documento con el catálogo con la caracterización de herramientas que implementan la metodología GitOps 	MILESTONE	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Fase 2 - Seleccionar la herramienta más relevante para la metodología GitOps, teniendo en cuenta aspectos como el ... 	PHASE	<div style="width: 0%;"></div> 0%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Fase 3 - Construir una guía de despliegue de acuerdo a la metodología GitOps para realizar la implementación tecnol... 	PHASE	<div style="width: 0%;"></div> 0%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Fase 4 - Evaluar la guía de despliegue a través de la implementación de un prototipo funcional de la herramienta Git... 	PHASE	<div style="width: 0%;"></div> 0%	

C.6. Diagrama de Gantt - GitOps - Fase 2

Diagrama de gantt - GitOps - Work + Crear Incluir proyectos 1 Filtro 1 Tabla Info Expandir More

ASUNTO	TIPO	PROGRESO (%)	*
<ul style="list-style-type: none"> Fase 2 - Seleccionar la herramienta más relevante para la metodología GitOps, teniendo en cuenta aspectos como ... 	PHASE	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1. Realizar una evaluación formal 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.1. Evaluar las alternativas 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.1.1. Definir los criterios para evaluar las soluciones alternativas. 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.1.2. Definir el rango y la escala para clasificar los criterios de evaluación. 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.1.3. Clasificar los criterios 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.1.4. Evaluar los criterios y su importancia relativa 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.1.5. Evolucionar los criterios de evaluación para mejorar su validez. 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.1.6. Documentar el análisis razonado para la selección y el rechazo de los criterios de evaluación. 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.2. Identificar las soluciones alternativas 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> A. 2.1.2.1. Seleccionar la solución 	TASK	<div style="width: 100%;"></div> 100%	
<ul style="list-style-type: none"> <ul style="list-style-type: none"> E. 2. Documento con el proceso de toma de decisión para herramienta/alternativa seleccionada 	MILESTONE	<div style="width: 100%;"></div> 100%	

C.7. Diagrama de Gantt - GitOps - Fase 3

Diagrama de gantt - GitOps - Work + Crear Incluir proyectos 1 Filtro 1 Tabla Info Expandir More

ASUNTO	TIPO	PROGRESO (%)	*
> Fase 2 - Seleccionar la herramienta más relevante para la metodología GitOps, teniendo en cuenta aspectos como ...	PHASE	<div style="width: 100%;"></div> 100%	
▼ Fase 3 - Construir una guía de despliegue de acuerdo a la metodología GitOps para realizar la implementación tecn...	PHASE	<div style="width: 100%;"></div> 100%	
▼ A. 3.1. Determinar cuales son aspectos relevantes de la metodología GitOps para la construcción de la guía de ...	TASK	<div style="width: 100%;"></div> 100%	
▼ A. 3.1.1. Construir la guía de despliegue	TASK	<div style="width: 100%;"></div> 100%	
A. 3.1.1.1. Describir prerequisites para la instalación	TASK	<div style="width: 100%;"></div> 100%	
A. 3.1.1.2. Planificar la instalación	TASK	<div style="width: 100%;"></div> 100%	
A. 3.1.1.3. Describir el proceso de instalación	TASK	<div style="width: 100%;"></div> 100%	
A. 3.1.1.4. Describir aspectos de afinamiento en la configuración del ambiente	TASK	<div style="width: 100%;"></div> 100%	
A. 3.1.1.5. Describir el proceso de pruebas de la herramienta seleccionada	TASK	<div style="width: 100%;"></div> 100%	
A. 3.1.1.6. Describir aspectos de mantenibilidad de la herramienta	TASK	<div style="width: 100%;"></div> 100%	
E. 3. Documento con la guía de despliegue de la herramienta que implementan la metodología GitOps	MILESTONE	<div style="width: 100%;"></div> 100%	
> Fase 4 - Evaluar la guía de despliegue a través de la implementación de un prototipo funcional de la herramienta Gi...	PHASE	<div style="width: 0%;"></div> 0%	

C.8. Diagrama de Gantt - GitOps - Fase 4

Diagrama de gantt - GitOps - Work + Crear Incluir proyectos 1 Filtro 1 Tabla Info Expandir More

ASUNTO	TIPO	PROGRESO (%)	*
☰ ▼ Fase 4 - Evaluar la guía de despliegue a través de la implementación de un prototipo funcional de la herramienta Gi...	PHASE	<div style="width: 100%;"></div> 100%	Info ...
▼ A. 4.1. Implementar la herramienta seleccionada	TASK	<div style="width: 100%;"></div> 100%	
A. 4.1.1. Cumplir prerequisites indicados en la guía de despliegue	TASK	<div style="width: 100%;"></div> 100%	
A. 4.1.2. Realizar proceso de instalación de la herramienta seleccionada	TASK	<div style="width: 100%;"></div> 100%	
A. 4.1.3. Cumplir aspectos de configuración indicados en la guía de despliegue	TASK	<div style="width: 100%;"></div> 100%	
A. 4.1.4. Verificar el despliegue y el cumplimiento funcional de la herramienta seleccionada	TASK	<div style="width: 100%;"></div> 100%	
E. 4. Informe/bitácora detallado del proceso de la implementación de la herramienta seleccionada	MILESTONE	<div style="width: 100%;"></div> 100%	
▼ A. 4.2. Ejecutar la guía de despliegue a través de la implementación de un prototipo funcional de la herramienta...	TASK	<div style="width: 100%;"></div> 100%	
A. 4.2.1. Desplegar el prototipo funcional en el ambiente previamente creado	TASK	<div style="width: 100%;"></div> 100%	
A. 4.2.2. Cumplir las pruebas indicadas en la guía de despliegue	TASK	<div style="width: 100%;"></div> 100%	
A. 4.2.3. Realizar el informe de los resultados obtenidos en las pruebas	TASK	<div style="width: 100%;"></div> 100%	
E. 5. Informe/conclusiones detallando del resultado de las pruebas realizadas	MILESTONE	<div style="width: 100%;"></div> 100%	

C.9. Diagrama de Gantt - GitOps



D.1. Un SMS de herramientas que implementan la metodología GitOps -
página 1

Informe sobre la implementación del Estudio de Mapeo Sistemático

Título sugerido:

Un mapeo sistemático de herramientas que implementan la metodología GitOps

Acrónimos y definiciones

- GQM: Goal-Question-Metric
- SMS: Systematic Mapping Study
- PICOC: Population, Intervention, Outcomes, and Context

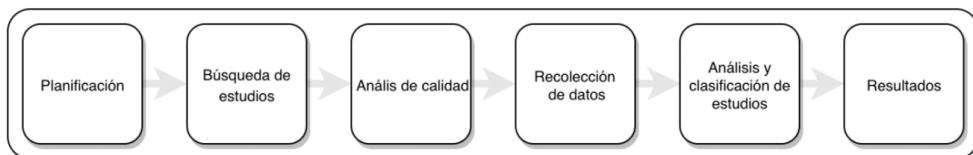


Figura 1: Etapas del proceso para la construcción de un SMS.

En la figura 1 se describen la etapas del mapeo sistemático realizado para las herramientas que implementan la metodología GitOps, estas etapas son: 1) planificación, 2) búsqueda de estudios, 3) Análisis de calidad, 4) Recolección de datos, 5) Análisis y clasificación de estudios y 6) Resultados

D.2. Un SMS de herramientas que implementan la metodología GitOps -

página 2

Tabla de contenidos

1. ETAPA 1: PLANEACIÓN	3
1.1. DEFINICIÓN DE OBJETIVOS PARA EL SMS	3
1.2. DEFINICIÓN DE PREGUNTAS DE INVESTIGACIÓN	3
1.3. DEFINICIÓN DE MÉTRICAS	3
2. ETAPA 2: BÚSQUEDA DE ESTUDIOS	4
2.1. ESTRATEGIA DE BÚSQUEDA	4
2.2. BÚSQUEDA DE BASE DE DATOS	4
2.2.1. IDENTIFICACIÓN DE ESTUDIOS POR BÚSQUEDA EN BASES DE DATOS:	4
2.2.1.1. CADENAS DE BÚSQUEDA EN LA BASE DE DATOS DE ACM	5
2.2.1.2. CADENAS DE BÚSQUEDA PARA LA BASE DE DATOS IEEE Xplore	6
2.2.1.3. CADENAS DE BÚSQUEDA PARA LA BASE DE DATOS Web Of Science	6
2.2.1.4. CADENAS DE BÚSQUEDA PARA LA BASE DE DATOS Springer	7
2.2.1.5. CADENAS DE BÚSQUEDA PARA LA BASE DE DATOS Scopus	7
2.2.1.6. CADENAS DE BÚSQUEDA PARA LA BASE DE DATOS Science Direct	8
2.2.1.7. RESUMEN	9
2.2.2. APLICACIÓN DE CRITERIOS DE EXCLUSIÓN EN BASES DE DATOS	9
2.2.2.1. APLICACIÓN DE CRITERIOS DE EXCLUSIÓN EN LA BASE DE DATOS DE ACM	9
2.2.2.2. APLICACIÓN DE CRITERIOS DE EXCLUSIÓN EN LA BASE DE DATOS IEEE Xplore	10
2.2.2.3. APLICACIÓN DE CRITERIOS DE EXCLUSIÓN EN LA BASE DE DATOS DE Web Of Science	10
2.2.2.4. APLICACIÓN DE CRITERIOS DE EXCLUSIÓN EN LA BASE DE DATOS Springer	11
2.2.2.5. APLICACIÓN DE CRITERIOS DE EXCLUSIÓN EN LA BASE DE DATOS Scopus	12
2.2.2.6. APLICACIÓN DE CRITERIOS DE EXCLUSIÓN EN LA BASE DE DATOS Science Direct	12
2.2.2.7. RESUMEN	13
2.2.3. APLICAR EXCLUSIÓN DUPLICADA	13
2.2.3.1. PANTALLA EN TÍTULO, PALABRAS CLAVE Y RESUMEN	15
2.3. PROCESO DE BÚSQUEDA DE BOLA DE NIEVE	16
2.3.1. APLICAR EL PROCESO DE BÚSQUEDA DE BOLA DE NIEVE	16
GitOps: THE EVOLUTION OF DEVOPS?	17
ANALYSIS ON THE USE OF DECLARATIVE AND PULL-BASED DEPLOYMENT MODELS ON GitOps USING ARGO CD	18
PREVALENCE OF GitOps, DEVOPS IN FAST CI/CD CYCLES	19
CONTINUOUS DEPLOYMENT IN IoT EDGE COMPUTING A GitOps IMPLEMENTATION	20
ON CONTINUOUS INTEGRATION / CONTINUOUS DELIVERY FOR AUTOMATED DEPLOYMENT OF MACHINE LEARNING MODELS USING MLOps	21

D.3. Un SMS de herramientas que implementan la metodología GitOps -

página 3

1. Etapa 1: Planeación

Esta etapa busca definir el propósito general a lograr con el SMS. Al mismo tiempo esta etapa incluye el establecimiento de aspectos como objetivos, preguntas de investigación y métricas. Esto se hace siguiendo el modelo Goal-Question-Metric (GQM). A continuación, la definición de metas para el SMS aplicado a las herramientas que implementan la metodología GitOps.

1.1. Definición de objetivos para el SMS

Tabla 1: Metas para el SMS

Goal	Description
G1	Caracterizar tecnologías que implementan la metodología GitOps.
G2	Identificar las tendencias de uso y estándares de las herramientas que implementan la metodología GitOps.

1.2. Definición de preguntas de investigación

Tabla 2: preguntas de investigación para el SMS

Goal	Question	Description	Motivation
G1	Q1	¿Cuáles son las herramientas que implementan la metodología GitOps?	Sobre estas herramientas: 1- Para saber cuales son 2- Reconocer sus funcionalidades
G2	Q2	¿Cuáles son los escenarios en los que las empresas tecnológicas toman como alternativa el uso de las herramientas que implementan la metodología GitOps?	Sobre estas herramientas: 1- Reconocer características y tendencias en el uso de herramientas que implementan la metodología GitOps 2- Reconocer ventajas y desventajas de las herramientas que implementan la metodología GitOps

1.3. Definición de métricas

Tabla 3: Métricas definidas para SMS

Metric	Description
M1	Número de herramientas que implementan la metodología GitOps.
M2	Número de casos de éxito de empresas que implementan la metodología GitOps.
M3	Número de empresas que alguna vez han implementado la metodología GitOps.

D.4. Un SMS de herramientas que implementan la metodología GitOps -

página 4

2. Etapa 2: Búsqueda de estudios

Esta etapa consta de las siguientes secciones: 1) Estrategia de búsqueda ya sea de forma independiente o combinada; 2) Identificación general de los estudios; 3) Filtrar y finalmente 4) Selección de estudios a incluir en el SMS.

2.1. Estrategia de búsqueda

Este trabajo combina las estrategias "búsqueda en base de datos" y "búsqueda de bola de nieve". Para la estrategia de "búsqueda en base de datos" se determinaron las siguientes bases de datos: ACM, IEEE Explorer, Springer, Scopus y Web Of Science.

2.2. Búsqueda de base de datos

Para ello se seleccionaron las siguientes bases de datos: ACM, IEEE Explorer, Springer, Scopus y Web of Science.

2.2.1. Identificación de estudios por búsqueda en bases de datos:

Esta etapa del proceso requiere establecer las palabras clave establecidas que se utilizarán en las cadenas de búsqueda para cada una de las bases de datos elegidas. Los plazos deben tomar en consideración los elementos identificados en la etapa de Planificación, para lo cual también se utiliza como guía metodológica el modelo PICOC.

Tabla 4: Aplicación del modelo PICOC

Componente	Descripción
Población	Estudios relacionados con el software que implementan la metodología GitOps.
Intervención	Identificación y caracterización del software que implementa la metodología GitOps.
Comparación	Identificar casos de éxito en el uso de la metodología GitOps, al igual que las tendencias de uso. (ambitos)
Resultados	- Número de herramientas que implementan la metodología GitOps - Cantidad de ámbitos encontrados en el uso de la metodología GitOps
Contexto	Ámbito de las prácticas DevOps.

Tabla 5: Palabras identified using the PICO model

Población	Intervención	Comparación	Resultados	Contexto
Marcos de gestión de GitOps Buenas prácticas de Ti Procesos de TI Tecnologías de TI	Identificación (Conjunto, Grupo, Partes o Elementos)	Casos de aceptación y éxito. Casos de estudio.	Conjunto de herramientas que implementan la metodología GitOps.	Ecosistema de tecnología, ecosistema digital y ecosistema empresarial

D.5. Un SMS de herramientas que implementan la metodología GitOps -

página 5

Table 6: Palabras clave de búsqueda en la base de datos

Keywords	Synonyms
Implementation	Deployment, Delivery, Release, Building, Making, Software, Tools
GitOps	DevOps, Continuous Integration, Continuous Deployment, Continuous Delivery

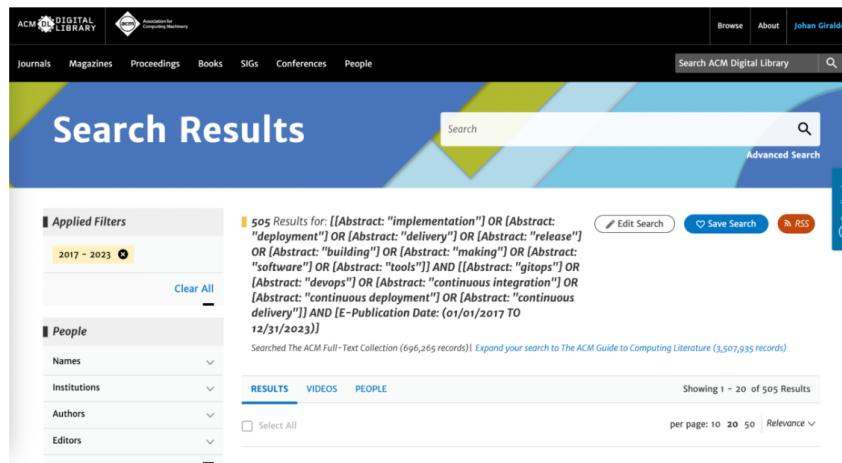
Table 7: Criterio Inclusion/Exclusion

Category	Inclusion	Exclusion
Fields	Abstract, title, keywords	- Full document
Publication type	Journal articles, Conferences proceedings	Thesis and books chapters.
Area/Discipline	Technological Infrastructure, Cloud Computing, Release management	Areas not related to Technological Infrastructure, Cloud Computing, Release management
Period	Between 2019 to 2023	Less than 2019
Language	English, Spanish	Spanish

Esta búsqueda utiliza solo palabras clave en el título, resumen y palabras clave. Los criterios de exclusión aún no se consideran.

2.2.1.1. Cadenas de búsqueda en la base de datos de ACM

Abstract:("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND Abstract:("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")

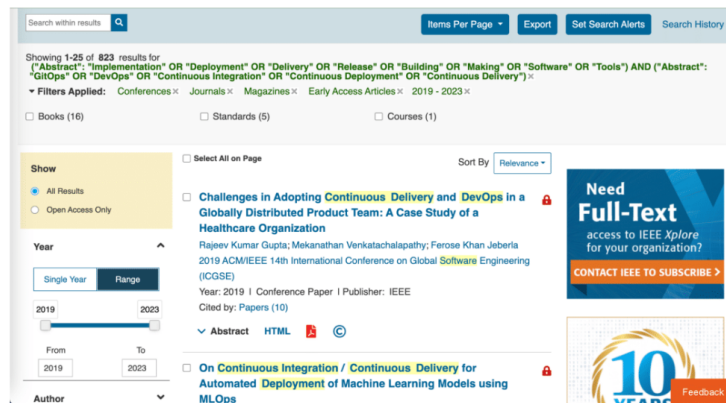


D.6. Un SMS de herramientas que implementan la metodología GitOps -

página 6

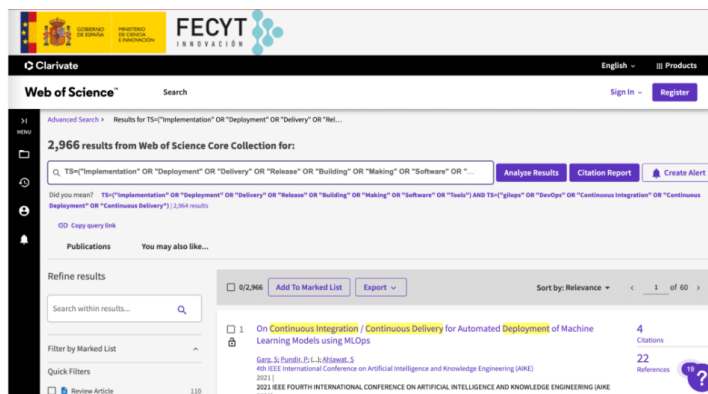
2.2.1.2. Cadenas de búsqueda para la base de datos IEEE Xplore

("Abstract": "Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND ("Abstract": "GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")



2.2.1.3. Cadenas de búsqueda para la base de datos de Web Of Science

TS=("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND TS=("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")



D.7. Un SMS de herramientas que implementan la metodología GitOps -
 página 7

2.2.1.4. Cadenas de búsqueda para la base de datos Springer

(Implementation OR Deployment OR Delivery OR Release OR Building OR Making OR Software OR Tools) AND ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")

2.2.1.4. Cadenas de búsqueda para la base de datos Springer

(Implementation OR Deployment OR Delivery OR Release OR Building OR Making OR Software OR Tools) AND ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")

Springer Link

(Implementation OR Deployment OR Delivery OR Release OR Building OR Making OR Software OR Tools) AND ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")

7,391 Result(s) for '(Implementation OR Deployment OR Delivery OR Release OR Building OR Making OR Software OR Tools) AND ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")' within 2019 - 2023

Refine Your Search

Content Type

Chapter	4,268
Book	1,560
Conference Paper	1,550
Article	1,425
Conference Proceedings	638
Reference Work Entry	108
Reference Work	26
Video Segment	21
Protocol	9

Discipline

Computer Science	4,069
Engineering	1,398
Business and Management	703
Medicine & Public Health	236
Biomedicine	141

Sort By: Relevance, Newest First, Oldest First, Date Published

Show documents published 2019 - 2023 (Available 1955 - 2023)

Start year: 2019, End year: 2023

Book

Continuous Integration (CI) and Continuous Delivery (CD)
 A Practical Guide to Designing and Developing Pipelines
 Henry van Merode (2023)

Book

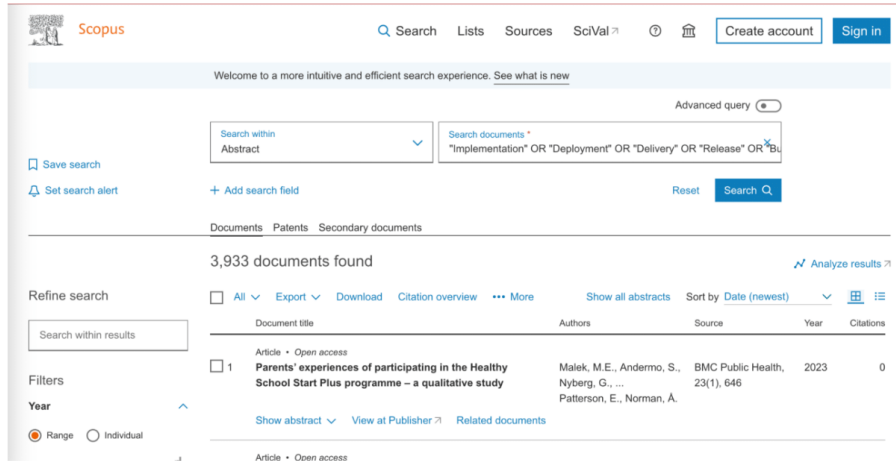
A Complete Guide to DevOps with AWS
 Deploy, Build, and Scale Services with AWS Tools and Techniques
 Osama Mustafa (2023)

2.2.1.5. Cadenas de búsqueda para la base de datos SCOPUS

ABS ("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND ABS ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")

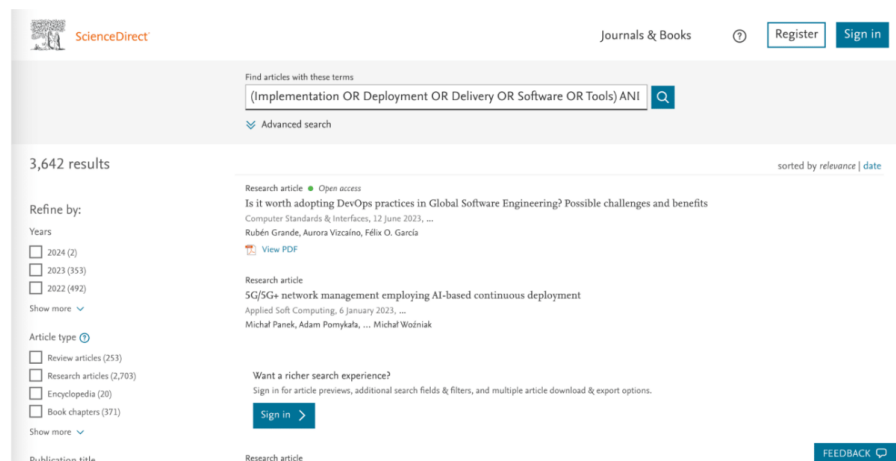
D.8. Un SMS de herramientas que implementan la metodología GitOps -

página 8



2.2.1.6. Cadenas de búsqueda para la base de datos SCIENCE DIRECT

(Implementation OR Deployment OR Delivery OR Software OR Tools) AND ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment")



D.9. Un SMS de herramientas que implementan la metodología GitOps -

página 9

2.2.1.7. Resumen

Este es el resultado antes de aplicar los criterios de exclusión.

Database	ACM	IEEE	WOS	Springer	Scopus	Science Direct	Total
Studies without application of exclusion criteria	676	823	2966	7391	3933	3642	19599

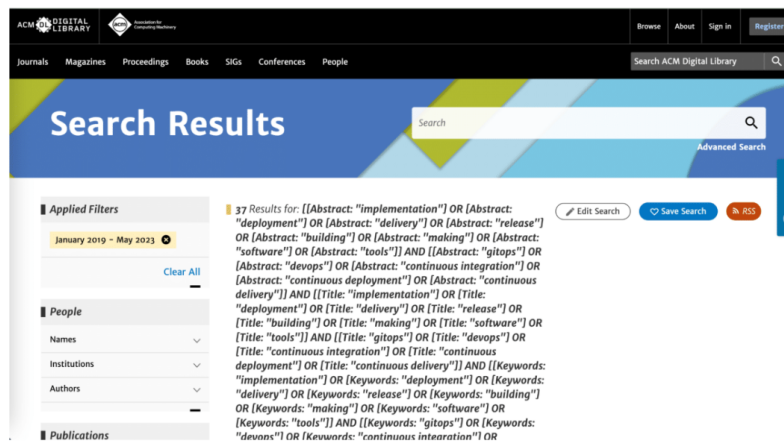
2.2.2. Aplicación de criterios de exclusión en bases de datos

Esta búsqueda se realiza sobre el título, el resumen y las palabras clave, además de considerar los criterios y la exclusión.

También se aplicó un rango de búsqueda definiendo la fecha de inicio 2019/01 y de fin 2023/05

2.2.2.1. Aplicación de criterios de exclusión en la base de datos de ACM

Abstract:("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND Abstract:("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery") AND Title:("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND Title:("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery") AND Keyword:("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND Keyword:("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")

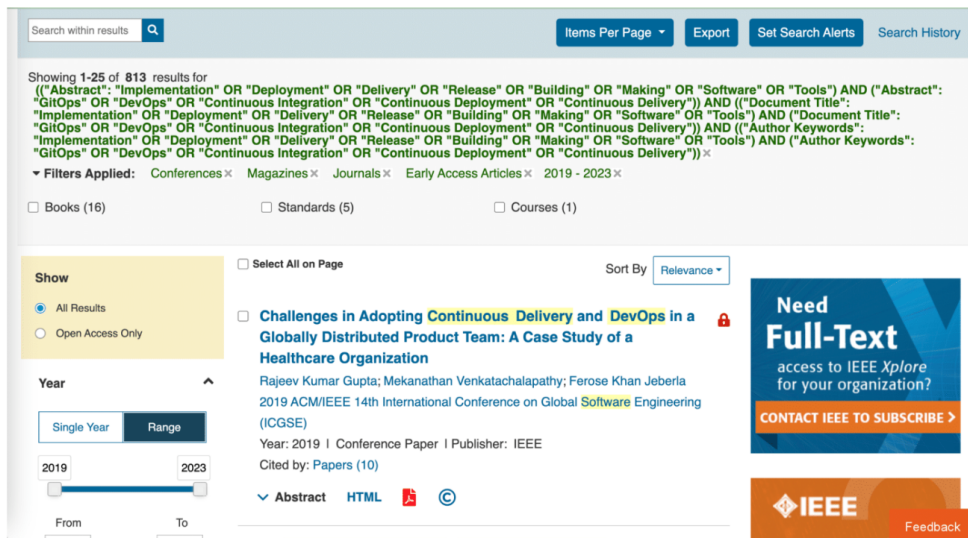


D.10. Un SMS de herramientas que implementan la metodología GitOps -

página 10

2.2.2.2. Aplicación de criterios de exclusión en la base de datos IEEE Xplore

(("Abstract": "Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND ("Abstract": "GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")) AND (("Document Title": "Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND ("Document Title": "GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")) AND (("Author Keywords": "Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND ("Author Keywords": "GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery"))

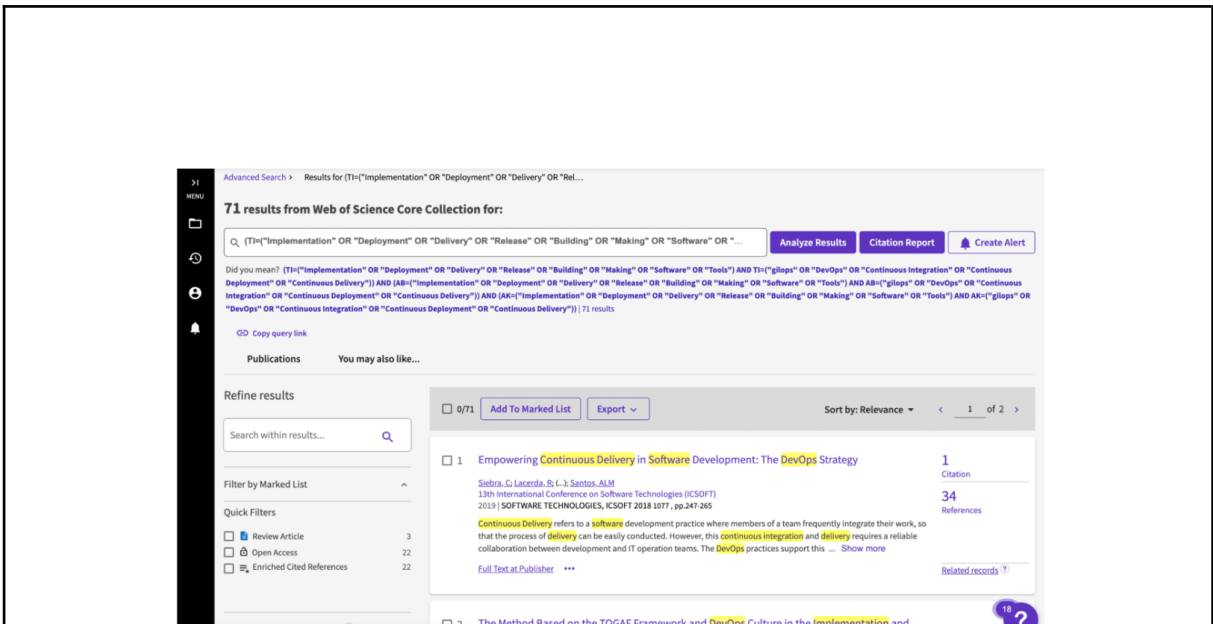


2.2.2.3. Aplicación de criterios de exclusión en la base de datos de Web Of Science

(TI=("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND TI=("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")) AND (AB=("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND AB=("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")) AND (AK=("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND AK=("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery"))

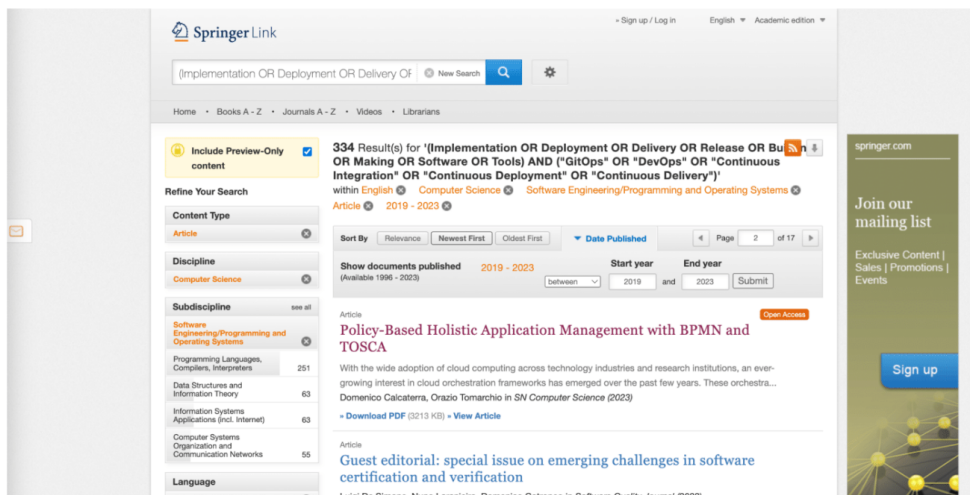
D.11. Un SMS de herramientas que implementan la metodología GitOps -

página 11



2.2.2.4. Aplicación de criterios de exclusión en la base de datos Springer

(Implementation OR Deployment OR Delivery OR Release OR Building OR Making OR Software OR Tools) AND ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")



D.12. Un SMS de herramientas que implementan la metodología GitOps -

página 12

2.2.2.5. Aplicación de criterios de exclusión en la base de datos Scopus

(ABS("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND ABS ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")) AND (TITLE("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND TITLE("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery")) AND (AUTHKEY("Implementation" OR "Deployment" OR "Delivery" OR "Release" OR "Building" OR "Making" OR "Software" OR "Tools") AND AUTHKEY("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment" OR "Continuous Delivery"))

25 documents found

Refine search

Search within results

Filters Clear all

Year Clear

Range Individual

2019 — 2023

Subject area Clear (2)

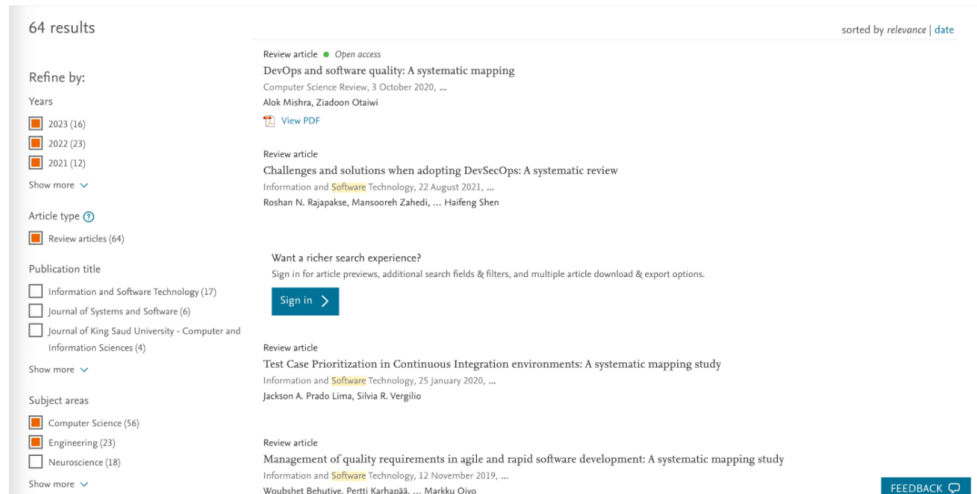
- Limited to Computer Science 25
- Limited to Engineering 8
- Mathematics 4
- Physics and Astronomy 3

Document title	Authors	Source	Year	Citations
1 Continuous deployment in software-intensive system-of-systems	Dakkak, A., Bosch, J., Olsson, H.H., Issa Maltos, D.	Information and Software Technology, 159, 107200	2023	0
2 5G/5G+ network management employing AI-based continuous deployment	Panek, M., Pomykala, A., Jablonski, I., Wozniak, M.	Applied Soft Computing, 134, 109984	2023	0
3 Continuous delivery: A PARS online course development cycle	Borgman, J., McArdle, C.	Computers and Composition, 66, 102741	2022	0
4 IMPLEMENTING CONTINUOUS DELIVERY IN A FINTECH	Lewis, Jayadi, R.	Journal of Theoretical	2022	0

2.2.2.6. Aplicación de criterios de exclusión en la base de datos Science Direct

(Implementation OR Deployment OR Delivery OR Software OR Tools) AND ("GitOps" OR "DevOps" OR "Continuous Integration" OR "Continuous Deployment")

D.13. Un SMS de herramientas que implementan la metodología GitOps -
página 13



2.2.2.7. Resumen

Este es el resultado después de aplicar los criterios de exclusión.

Database	ACM	IEEE	WOS	Springer	Scopus	Science Direct	Total
Studies without application of exclusion criteria	676	823	2966	7391	3870	3952	19678
Applying the exclusion criteria	37	812	71	334	25	64	1343

2.2.3. Aplicar exclusión duplicada

En las siguientes imágenes puede ver algunas partes de la interfaz del software SMS-GIA, que fue construido para ayudar en el proceso de construcción de un estudio de mapeo sistemático.

D.14. Un SMS de herramientas que implementan la metodología GitOps -
página 14

The screenshot shows a web browser window with the URL '143.110.233.68:9080/index.xhtml'. The page title is 'SMS-Builder'. The main content area is titled 'SMS - Builder :: sms-gitops'. It contains a form with three fields: 'Nombre SMS:' with the value 'sms-gitops', 'Descripción:' with the text 'SMS para Catálogo de herramientas de gestión de la configuración que implementan la metodología GitOps', and 'Paso de Evaluación:' with a dropdown menu set to '7-Ver Seleccionadas'. A blue 'Aceptar' button is at the bottom.

Después de cargar los estudios en el programa SMS-GIA, fue posible eliminar algunos registros duplicados. Vea la figura a continuación.

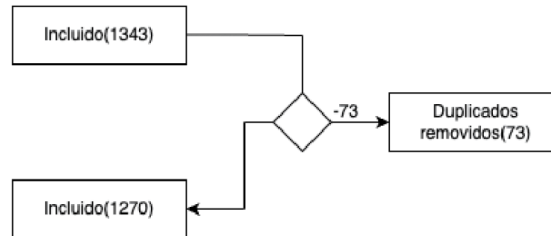
The screenshot shows the 'SMS - Builder :: sms-gitops' interface. At the top right is a 'Salir' button. Below is a search bar labeled 'Sugerir'. A pagination bar shows '1-20 de 1343' items. The main table has three columns: 'Fuente', 'Estudios', and 'Autores'. The table contains several rows of study entries. Two rows are highlighted in blue, indicating they are selected. The selected rows are:

Fuente	Estudios	Autores
<input type="checkbox"/>	IEEE 'Under-reported' Security Defects in Kubernetes Manifests	Shamim, Shazbul Islam ; Bose, Dibyendu Brinto ; Rahman, Akond
<input type="checkbox"/>	SPRINGER 2PC*: a distributed transaction concurrency control protocol of multi-microservice based on cloud computing platform	Yin, Wei ; Wang, Hui ; Sun, Haiying ; Chen, Xiaohong ; Liu, Jing ; Fan, Pan
<input type="checkbox"/>	IEEE 2nd International Workshop on the Foundations of Infrastructure Specification and Testing: FIST 2023	Quattrocchi, G ; Tamburri, D A ; Baresi, L
<input type="checkbox"/>	IEEE 5G-Kube: Complex Telco Core Infrastructure Deployment Made Low-Cost	Noor, Asma ; Scotece, Domenico ; Foschini, Luca ; Corradi, Antonio
<input type="checkbox"/>	SCOPUS 5G/5G+ network management employing AI-based continuous deployment	Pomykała, Adam ; Woźniak, Michał ; Jabłoński, Ireneusz ; Panek, Michał
<input checked="" type="checkbox"/>	WOS 5G/5G+ network management employing AI-based continuous deployment	Woźniak, Michał ; Pomykała, Adam ; Jabłoński, Ireneusz ; Panek, Michał
<input type="checkbox"/>	IEEE A Brief Survey of Current Software Engineering Practices in Continuous Integration and Automated Accessibility Testing	Sane, Parth
<input checked="" type="checkbox"/>	WOS A Brief Survey of Current Software Engineering Practices in Continuous Integration and Automated Accessibility Testing	Sane, Parth
<input type="checkbox"/>	SPRINGER A CEP-driven framework for real-time news impact prediction on financial markets	Al-Qudah, Islam ; Rabhi, Fethi A. ; Chen, Weisi ; El Majzoub, Ahmad
<input type="checkbox"/>	IEEE A Cloud-Based Framework for Machine Learning Workloads and Applications	Duma, D C ; Costantini, A ; Gomes, J ; Castell, W Zu ; Lucas, J M De ; Kozlov, V Y ; Ustr, Z ; Wolniewicz, P ; David, M ; Cacha, I Heredia ; Caballer, M ; Iglesias, L Lloret ; Tran, V ; Hardt, M ; Ito, K ; Dlugolinsky, S ; Nguyen, G ; Plociennik, M ; Donvito, G ; Antonacci, M ; Garçon, L ; Lopez, Ferrnández, P Orviz ; Plasencia, I C ; Alic, A S ; Molt, G

Se eliminaron 73 elementos duplicados, ahora se incluyen 1270 estudios.

D.15. Un SMS de herramientas que implementan la metodología GitOps -

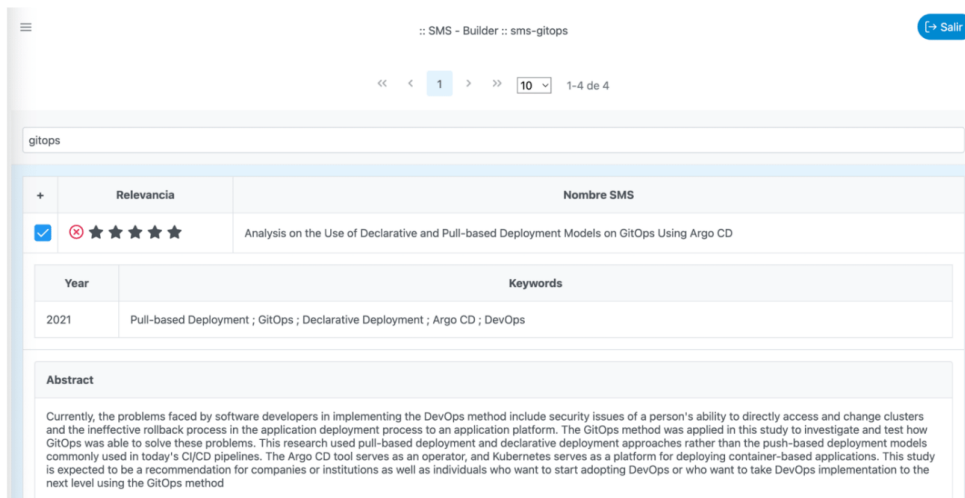
página 15



2.2.3.1. Pantalla en título, palabras clave y resumen

Esta parte del proceso consiste en evaluar 1270 estudios mediante la lectura parcial de los documentos para determinar cuáles deben descartarse según los criterios de exclusión determinados en el mapeo sistemático.

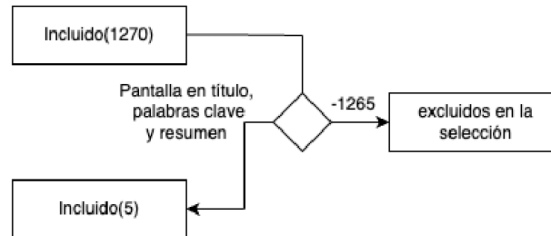
La siguiente es la interfaz del software SMS-GIA configurada para este propósito.



Luego de este proceso de exclusión se incluyen 5 estudios para continuar con el SMS, como se muestra a continuación.

D.16. Un SMS de herramientas que implementan la metodología GitOps -

página 16

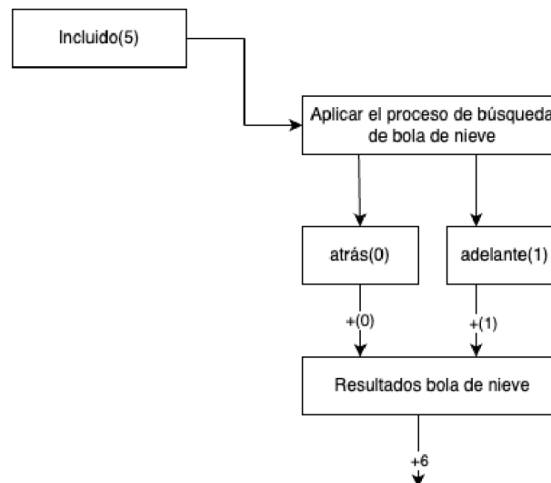


El resultado de la estrategia de búsqueda en la base de datos terminó con 5 estudios incluidos.

2.3. Proceso de búsqueda de bola de nieve

2.3.1. Aplicar el proceso de búsqueda de bola de nieve

El proceso de búsqueda por bola de nieve consistió en analizar los 5 estudios de base. Con cada documento comenzamos con la búsqueda hacia atrás revisando la lista de referencias utilizadas. La búsqueda directa también se llevó a cabo utilizando Google Scholar para identificar qué artículos han citado cada uno de los estudios analizados. En todos los casos, siempre se tuvieron en cuenta los criterios de inclusión y exclusión determinados al inicio del SMS. Después de realizar el proceso de búsqueda de bola de nieve, se obtuvieron 0 estudios hacia atrás y 1 hacia adelante para un total de 1 estudios nuevos. En total, el proceso de búsqueda de bola de nieve da como resultado 6 estudios.



D.17. Un SMS de herramientas que implementan la metodología GitOps -

página 17

Lista de los nuevos estudios arrojados por el proceso de búsqueda de bola de nieve.

GitOps: The Evolution of DevOps?

Atrás (21)

No	Study
1	G. Kim, J. Humble, P. Debois, and J. Willis, <i>The DevOps Handbook: How to Create World-Class Agility, Reliability and Security in Technology Organizations</i> . Portland, OR: IT Revolution, 2016.
2	D. Safar, "DevOps: The next evolution-GitOps," <i>Weaveworks</i> , 2017. https://www.weave.works/blog/devops-the-next-evolution-gitops (accessed June 27, 2019).
3	R. Jabbari, N. Bin Ali, K. Petersen, and B. Tanveer, "What is DevOps?: A systematic mapping study on definitions and practices," in <i>Proc. Scientific Workshop XP2016</i> , Edinburgh, U.K.: Association for Computing Machinery, May 24, 2016, p. 12. doi: 10.1145/2962695.2962707.
4	"Guide to GitOps," <i>Weaveworks</i> . https://www.weave.works/technologies/gitops/ (accessed May 02, 2019).
5	A. Richardson, "GitOps-Operations by pull request," <i>Weaveworks</i> . https://www.weave.works/blog/gitops-operations-by-pull-request (accessed May 02, 2019).
6	A. Richardson, "What is GitOps?" <i>Weaveworks</i> . https://www.weave.works/blog/what-is-gitops-really (accessed May 02, 2019).
7	R. Jabbari, N. Bin Ali, K. Petersen, and B. Tanveer, "Towards a benefits dependency network for DevOps based on a systematic literature review," <i>J. Softw., Evol. Process</i> , vol. 30, no. 6, p. e1957, July 2018. doi: 10.1002/smr.1957.
8	J. Davis and R. Daniels, <i>Effective DevOps: Building a Culture of Collaboration, Affinity, and Tooling at Scale</i> . Sebastopol, CA: O'Reilly Media, 2016.
9	D. Sato, <i>DevOps in Practice: Reliable and Automated Software Delivery</i> . São Paulo, Brazil: Casa do Código, 2014.
10	C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, "DevOps," <i>IEEE Softw.</i> , vol. 33, no. 3, pp. 94-100, 2016. doi: 10.1109/MS.2016.68.
11	L. E. Lwakatara, P. Kuvaja, and M. Oivo, "Dimensions of DevOps" in <i>Proc. 2015 16th Int. Conf. Agile Processes Softw. Eng., Extreme Programming, C. Lassenius, T. Dingsøyr, and M. Paasivaara, Eds.</i> vol. 212. Cham: Springer International Publishing, May 25-29, 2015, pp. 212-217. doi: 10.1007/978-3-319-18612-2_19.
12	C. Haddad, "DevOps = DevOps principles + DevOps practices," <i>DZone</i> , 2014. https://dzone.com/articles/devops-devops-principles (accessed June 08, 2019).
13	"What is DevOps?" <i>Amazon Web Services</i> , 2019. https://aws.amazon.com/devops/what-is-devops (accessed June 08, 2019).
14	M. Fowler, "Continuous integration," <i>Martin Fowler</i> , 2006. https://martinfowler.com/articles/continuousIntegration.html (accessed June 06, 2019).
15	L. Chen, "Continuous delivery: Huge benefits, but challenges too," <i>IEEE Softw.</i> , vol. 32, no. 2, pp. 50-54, 2015. doi: 10.1109/MS.2015.27.
16	A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices architecture enables DevOps: Migration to a cloud-native architecture," <i>IEEE Softw.</i> , vol. 33, no. 3, pp. 42-52, 2016. doi: 10.1109/MS.2016.64.
17	N. Paez, "Versioning strategy for DevOps implementations," in <i>Proc. 2018 Congreso Argentino de Ciencias de la Informática y Desarrollos de Investigación (CACIDI)</i> , pp. 1-6. doi: 10.1109/CACIDI.2018.8584362.
18	V. Farcic, <i>The DevOps 2.4 Toolkit: Continuous Deployment to Kubernetes: Continuously Deploying Applications With Jenkins to a Kubernetes Cluster</i> . Birmingham, U.K.: Packt, 2019.
19	B. Burns, E. Villalba, D. Strelbel, and L. Evenson, <i>Kubernetes Best Practices: Blueprints for Building Successful Applications</i> . Sebastopol, CA: O'Reilly Media, 2020.
20	"GitOps-style continuous delivery with cloud build," <i>Google Cloud</i> , 2019. https://cloud.google.com/kubernetes-engine/docs/tutorials/gitops-cloud-build (accessed June 25, 2019).
21	T. A. Limoncelli, "GitOps: A path to more self-service IT," <i>ACM Queue</i> , vol. 16, no. 3, pp. 13-26, 2018. doi: 10.1145/3236386.3237207.

D.18. Un SMS de herramientas que implementan la metodología GitOps -

página 18

Adelante (25)

No	Study
1	Donca, I. C., Stan, O. P., Misaros, M., Gota, D., & Miclea, L. (2022). Method for continuous integration and deployment using a pipeline generator for agile software projects. <i>Sensors</i> , 22(12), 4637.
2	Rauh, L., Gärtner, S., Brandt, D., Oberle, M., Stock, D., & Bauernhansl, T. (2022). Towards ai lifecycle management in manufacturing using the asset administration shell (aas). <i>Procedia CIRP</i> , 107, 576-581.
3	Theunissen, T., Hoppenbrouwers, S., & Overbeek, S. (2022). Approaches for Documentation in Continuous Software Development. <i>Complex Systems Informatics and Modeling Quarterly</i> , (32), 1-27.
4	Scotece, D., Noor, A., Foschini, L., & Corradi, A. (2023). 5G-Kube: Complex Telco Core Infrastructure Deployment Made Low-Cost. <i>IEEE Communications Magazine</i> .
5	Bhimij, W., Carder, D., Dart, E., Duarte, J., Fisk, I., Gardner, R., ... & Würthwein, F. (2023). Snowmass 2021 Computational Frontier CompF4 Topical Group Report Storage and Processing Resource Access. <i>Computing and Software for Big Science</i> , 7(1), 5.
6	Coleman, C., Griswold, W. G., & Mitchell, N. (2022). Do Cloud Developers Prefer CLIs or Web Consoles? CLIs Mostly, Though It Varies by Task. <i>arXiv preprint arXiv:2209.07365</i> .
7	Flechas, M. A., Attebury, G., Bloom, K., Bockelman, B., Gray, L., Holzman, B., ... & Thiltges, J. (2022). Collaborative computing support for analysis facilities exploiting software as infrastructure techniques. <i>arXiv preprint arXiv:2203.10161</i> .
8	Pourmajidi, W., Zhang, L., Steinbacher, J., Erwin, T., & Miransky, A. (2023). A Reference Architecture for Observability and Compliance of Cloud Native Applications. <i>arXiv preprint arXiv:2302.11617</i> .
9	Kintzel, F., Calizzano, R., & Rehm, G. (2022). Cloud Infrastructure of the European Language Grid. In <i>European Language Grid: A Language Technology Platform for Multilingual Europe</i> (pp. 95-106). Cham: Springer International Publishing.
10	Talasila, P., Gomes, C., Mikkelsen, P. H., Arboleda, S. G., Kamburjan, E., & Larsen, P. G. (2023). Digital Twin as a Service (DTaaS): A Platform for Digital Twin Developers and Users. <i>arXiv preprint arXiv:2305.07244</i> .
11	Leiter, Á., Hegyi, A., Kispál, I., Böösy, P., Galambosi, N., & Tar, G. Z. (2023, May). GitOps and Kubernetes Operator-based Network Function Configuration. In <i>NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium</i> (pp. 1-5). IEEE
12	Bhimij, W., Guok, C., Würthwein, F., Neubauer, M. S., van Gemmeren, P., Weaver, B., ... & Lehman, T. (2022). arXiv: Snowmass 2021 Computational Frontier CompF4 Topical Group Report: Storage and Processing Resource Access (No. FERMILAB-PUB-22-715-SCD).
13	Deutsch, D. (2023). Machine learning operations-domain analysis, reference architecture, and example implementation/Author Daniel Deutsch, LL. B.(WU). LL. M.(WU).
14	Henning, S. (2023). Scalability Benchmarking of Cloud-Native Applications Applied to Event-Driven Microservices (Doctoral dissertation).
15	Kwan, A. (2023). Piniverse: A Versatile AI-Powered Framework for Streamlining Digital Content Interaction and Empowering User Creativity (Doctoral dissertation, WORCESTER POLYTECHNIC INSTITUTE).
16	Quillen, N. C. (2022). Tools Engineers Need to Minimize Risk around CI/CD Pipelines in the Cloud (Doctoral dissertation, Capella University)
17	Silveira, D. M. D. (2022). LEAN DATA ENGINEERING. COMBINING STATE OF THE ART PRINCIPLES TO PROCESS DATA EFFICIENTLY (Doctoral dissertation).
18	Rughinis, P. E. R. V., & Stan, E. I. M. Privacy in Critical Infrastructure.
19	Loureiro, J., & de Oliveira, D. (2022, September). ORBITER: um Arcabouço para Implantação Automática de Aplicações Big Data em Arquiteturas Serverless. In <i>Anais do XXXVII Simpósio Brasileiro de Bancos de Dados</i> (pp. 379-384). SBC.
20	Ops, K. U. G. (2022). IMPLEMENTANDO UM SISTEMA DE coNTEINERzAção coM (Doctoral dissertation, UNIVERSIDADE FEDERAL DE PERNAMBUCO).
21	Wagner, D. (2022). Entwicklung eines Multitagentensystems für das dezentrale Deployment von Softwarekomponenten (Master's thesis).
22	Rodrigues, P. G., & Garcia, V. C. Implementando um sistema de containerização com Kubernetes usando GitOps.
23	DE NARRATIVA, U. F. P. C. HIAGO ALMEIDA DIAS.
24	Rughinis, R. V., & Stan, I. M. TEZĂ DE DOCTORAT Confidentialitatea datelor în infrastructuri critice.
25	이장원, 응웬, 탄응웬, & 김영한. (2023). 6G 네트워크를 위한 분산 클러스터 구성 시스템 설계. 한국통신학회 학술대회논문집, 200-201.

Analysis on the Use of Declarative and Pull-based Deployment Models on GitOps Using Argo CD

Atrás (14)

No	Study
1	R. Bolscher and M. Daneva, "Designing software architecture to support continuous delivery and DevOps: A systematic literature review," <i>ICSOFT 2019 - Proc. 14th Int. Conf. Softw. Technol.</i> , pp. 27-39, 2019, doi: 10.5220/0007837000270039.
2	T. Phoenix, P. A. Novel, H. Your, and B. Winpdf, "The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business WinpdfThe Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Winpdf."

D.19. Un SMS de herramientas que implementan la metodología GitOps -

3	M. K. A. Abbass, R. I. E. Osman, A. M. H. Mohammed, and M. W. A. Alshaikh, "Adopting continuous integration and continuous delivery for small teams," Proc. Int. Conf. Comput. Control. Electr. Electron. Eng. 2019, ICCCEEE 2019, 2019, doi: 10.1109/ICCCEEE46830.2019.9070849.
4	A. Proulx, F. Raymond, B. Roy, and F. Petrillo, "Problems and Solutions of Continuous Deployment: A Systematic Review," no. December 2018, 2018, [Online]. Available: http://arxiv.org/abs/1812.08939 .
5	L. Leite, C. Rocha, F. Kon, D. Milojicic, and P. Meirelles, "A survey of DevOps concepts and challenges," ACM Computing Surveys. 2019, doi: 10.1145/3359981.
6	M. Shahin, M. Ali Babar, and L. Zhu, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices," IEEE Access, vol. 5, no. Ci, pp. 3909–3943, 2017, doi: 10.1109/ACCESS.2017.2685629.
7	A. Agarwal, S. Gupta, and T. Choudhury, "Proceedings of the 2019 9th International Conference on Advances in Computing and Communication, ICACC 2019," Proc. 2019 9th Int. Conf. Adv. Comput. Commun. ICACC 2019, no. June, pp. 290–293, 2019.
8	G. Attardi, A. Barchiesi, A. Colla, R. Di Lallo, and F. Galeazzi, "Declarative modeling for deploying a container platform," Proc. - 32nd IEEE Int. Conf. Adv. Inf. Netw. Appl. Work. WAINA 2018, vol. 2018-Janua, pp. 386–389, 2018, doi: 10.1109/WAINA.2018.00116.
9	B. G. Kim, J. Humble, P. Debois, and J. Willis, The DevOps handbook, vol. First Edit, no. 7, 2016.
10	M. Wurster, U. Breitenbucher, O. Kopp, and F. Leymann, "Modeling and automated execution of application deployment tests," Proc. - 2018 IEEE 22nd Int. Enterp. Distrib. Object Comput. Conf. EDOC 2018, pp. 171– 180, 2018, doi: 10.1109/EDOC.2018.00030.
11	M. Wurster et al., "The essential deployment metamodel: a systematic review of deployment automation technologies," Software-Intensive Cyber-Physical Syst., vol. 35, no. 1–2, pp. 63–75, 2020, doi: 10.1007/s00450-019-00412-x.
12	N. Asthana, T. Chetalas, A. Karve, A. Segal, M. Dubey, and S. Zeng, "A declarative approach for service enablement on hybrid cloud orchestration engines," IEEE/IFIP Netw. Oper. Manag. Symp. Cogn. Manag. a Cyber World, NOMS 2018, pp. 1–7, 2018, doi: 10.1109/NOMS.2018.8406175.
13	X. Tao and X. Etchevers, "Poster : A Declarative Approach for Updating Distributed," 2018 IEEE/ACM 40th Int. Conf. Softw. Eng. Companion, no. May, pp. 392–393, 2018.
14	M. Shahin, M. A. Babar, M. Zahedi, and L. Zhu, "Beyond Continuous Delivery: An Empirical Investigation of Continuous Deployment Challenges," Int. Symp. Empir. Softw. Eng. Meas., vol. 2017-Novem, no. November, pp. 111–120, 2017, doi: 10.11

Adelante (3)

No	Study
1	Donca, I. C., Stan, O. P., Misaros, M., Gota, D., & Miclea, L. (2022). Method for continuous integration and deployment using a pipeline generator for agile software projects. Sensors, 22(12), 4637.
2	Yang, S., Moon, J., Kim, J., Lee, K., & Lee, K. (2023). FLScalizer: Federated Learning Lifecycle Management Platform. IEEE Access.
3	Leiter, A., Hegyi, A., Kispál, I., Böösy, P., Galambosi, N., & Tar, G. Z. (2023, May). GitOps and Kubernetes Operator-based Network Function Configuration. In NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium (pp. 1-5). IEEE.

Prevalence of GitOps, DevOps in Fast CI/CD Cycles

Atrás (17)

No	Study
1	A. Srivatsav Amaradri and S. Bindu Nutalapati, "Continuous Integration, Deployment and Testing in DevOps Environment", 2016. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:1044691/FULLTEXT02.pdf . [Accessed: 02- Apr- 2022].
2	S. Sharma and B. Coyne, DevOps for Dummies, IBM Limited Edition, 3rd ed. John Wiley & Sons, Inc, 2017.
3	"What DevOps is to the Cloud, GitOps is to Cloud Native", Weave.works, 2019. [Online]. Available: https://www.weave.works/blog/gitops-is-cloud-native . [Accessed: 02- Apr- 2022].
4	F. Beetz, A. Kammer and D. Harrer, "GitOps", GitOps. [Online]. Available: https://www.gitops.tech/ . [Accessed: 02- Apr- 2022].
5	Á. Sándor, "11 Reasons for Adopting GitOps", Blog.container- solutions.com, 2019. [Online]. Available: https://blog.container-solutions.com/why-adopt-gitops . [Accessed: 02- Apr- 2022].
6	"The GitOps FAQ", Weave.works. [Online]. Available: https://www.weave.works/technologies/gitops-frequently-asked- questions/ . [Accessed: 02- Apr- 2022].
7	S. Thorpe, "30+ Tools List for GitOps - DZone DevOps", dzone.com, 2020. [Online]. Available: https://dzone.com/articles/30-tools-list-for-gitops . [Accessed: 02- Apr- 2022].
8	"Argo CD - Declarative GitOps CD for Kubernetes", Argo- cd.readthedocs.io. [Online]. Available: https://argo-cd.readthedocs.io/en/stable/ . [Accessed: 02- Apr- 2022].
9	"GitHub - fluxcd/flux: Successor: https://github.com/fluxcd/flux2 — The GitOps Kubernetes operator", GitHub. [Online]. Available: https://github.com/fluxcd/flux . [Accessed: 02- Apr- 2022].

D.20. Un SMS de herramientas que implementan la metodología GitOps -

página 20

10	"GitHub - hasura/gitkube: Build and deploy docker images to Kubernetes using git push", GitHub. [Online]. Available: https://github.com/hasura/gitkube#~:text=Gitkube%20is%20a%20tool%20for,and%20deploy%20to%20Kubernetes%20automaticall y. [Accessed: 02- Apr- 2022].
11	"Jenkins X - Cloud Native CI/CD Built On Kubernetes", Jenkins X - Cloud Native CI/CD Built On Kubernetes, 2022. [Online]. Available: https://jenkins-x.io/ . [Accessed: 02- Apr- 2022].
12	"GitHub - fluxcd/helm-operator: Successor: https://github.com/fluxcd/helm-controller — The Flux Helm Operator, once upon a time a solution for declarative Helming.", GitHub. [Online]. Available: https://github.com/fluxcd/helm-operator . [Accessed: 02- Apr- 2022].
13	"What is JenkinsX? Building Cloud Native Apps Painlessly", CloudBees. [Online]. Available: https://www.cloudbees.com/whitepapers/building-cloud-native-apps-painlessly . [Accessed: 11- Dec- 2020].
14	A. Cloud, "How GitOps Is Solving Customer Pain Points", Medium, 2019. [Online]. Available: https://medium.datadriveninvestor.com/how-gitops-is-solving-customer-pain-points-877b6b8ba6cc . [Accessed: 11- Dec- 2020].
15	"Is GitOps the next big thing in DevOps? Atlassian Git Tutorial", Atlassian. [Online]. Available: https://www.atlassian.com/git/tutorials/gitops . [Accessed: 11- Dec- 2020].
16	A. Gillis, "What is GitOps and why is It Important?", SearchITOperations. [Online]. Available: https://www.techtarget.com/searchitoperations/definition/GitOps . [Accessed: 11- Dec- 2020].
17	P. Rai, Madhurima, S. Dhir, Madhulika and A. Garg, "A prologue of JENKINS with comparative scrutiny of various software integration tools," 2015 2nd International Conference on Computing for Sustainable Global

Adelante (1)

No	Study
1	Krishna, M. Y. S., & Gawre, S. K. (2023). MLOps for Enhancing the Accuracy of Machine Learning Models using DevOps, Continuous Integration, and Continuous Deployment. Research Reports on Computer Science, 97-103.

Continuous Deployment in IoT Edge Computing A GitOps implementation

Atrás (15)

No	Study
1	Yousefpour, A., Fung, C., Nguyen, T., Kadiyala, K., Jalali, F., Niakanlahiji, A., Kong, J., Jue, J.P.: All one needs to know about fog computing and related edge computing paradigms: A complete survey. Journal of Systems Architecture 98, 289–330 (2019). DOI 10.1016/j.sysarc.2019.02.009
2	C. Ebert, G. Gallardo, J. Hernantes and N. Serrano, "DevOps," in IEEE Software, vol. 33, no. 3, pp. 94-100, May-June 2016, doi: 10.1109/MS.2016.668.
3	L. Georgeta Gușeală, D. -V. Bratu and S. -A. Moraru, "DevOps Transformation for Multi-Cloud IoT Applications," 2019 International Conference on Sensing and Instrumentation in IoT Era (ISSI), 2019, pp. 1-6, doi: 10.1109/ISSI47111.2019.9043730.
4	2021 State of DevOps Report, Puppet 2021. Online available https://puppet.com/resources/report/2021-state-of-devops-report
5	L. E. Lwakatare et al., "Towards DevOps in the Embedded Systems Domain: Why is It So Hard?," 2016 49th Hawaii International Conference on System Sciences (HICSS), 2016, pp. 5437-5446, doi: 10.1109/HICSS.2016.671. https://ieeexplore.ieee.org/document/7427859
6	P. Nguyen et al., "Advances in Deployment and Orchestration Approaches for IoT - A Systematic Review," 2019 IEEE International Congress on Internet of Things (ICIOT), 2019, pp. 53-60, doi: 10.1109/ICIOT.2019.00021.
7	Jungha Hong, Yong-Geun Hong, Xavier de Foy, Matthias Kovatsch, Eve Schooler, Dirk Kutscher, "IoT Edge Challenges and Functions", Internet Engineering Task Force, 2022, [Online]. Available: https://datatracker.ietf.org/doc/html/draft-irtf-t2trg-iot-edge-04
8	From DevOps to EdgeOps: A Vision for Edge Computing Eclipse Foundation, Eclipse Foundation 2021. Available online: https://outreach.eclipse.foundation/edge-computing-edgeops-white-paper
9	ISO/IEC: Tr 30164:2020 - internet of things (iot) -edge computing. Tech. rep., ISO/IEC (2020)
10	Group, O.C.A.W., et al.: Openfog reference architecture for fog computing. OPFRA001 20817, 162 (2017)
11	Jez Humble and David Farley. 2010. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (1st. ed.). Addison-Wesley Professional.
12	Gene Kim, Patrick Debois, John Willis, and Jez Humble. 2016. The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations. IT Revolution Press.
13	Kief Morris. 2016. Infrastructure as Code: Managing Servers in the Cloud (1st. ed.). O'Reilly Media, Inc.
14	Thomas A. Limoncelli. 2018. GitOps: a path to more self-service IT. Commun. ACM 61, 9 (September 2018), 38–42. DOI: 10.1145/3233241
15	M. Fogli et al., "Performance Evaluation of Kubernetes Distributions (K8s, K3s, KubeEdge) in an Adaptive and Federated Cloud Infrastructure for Disadvantaged Tactical Networks," 2021 International Conference on Military Communication and Information Systems (ICMCIS), (ICMCIS), 2021, pp. 1-7, doi: 10.1109/ICMCIS52405.2021.9486396.

D.21. Un SMS de herramientas que implementan la metodología GitOps -

página 21

Adelante (3)

No	Study
1	Bijwe, A., & Shankar, P. (2022, October). Challenges of Adopting DevOps Culture on the Internet of Things Applications-A Solution Model. In 2022 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS) (pp. 638-645). IEEE.
2	Kwan, A. (2023). Piverse: A Versatile AI-Powered Framework for Streamlining Digital Content Interaction and Empowering User Creativity (Doctoral dissertation, WORCESTER POLYTECHNIC INSTITUTE).
3	Bijwe, A., & Shankar, P. DEVOPS CULTURE AND PRACTICES FOR IOT APPLICATIONS.

On Continuous Integration / Continuous Delivery for Automated Deployment of Machine Learning Models using MLOps

Atrás (29)

No	Study
1	P. Langley and H. A. Simon, "Applications of machine learning and rule induction," Communications of the ACM, vol. 38, no. 11, pp. 54– 64, 1995.
2	L. E. Lwakatare, A. Raj, I. Crnkovic, J. Bosch, and H. H. Olsson, "Large- scale machine learning systems in real-world industrial settings: A review of challenges and solutions," Information and Software Technology, vol. 127, p. 106368, 2020.
3	S. Garg and S. K. Garg, "Automated cloud infrastructure, continuous integration and continuous delivery using docker with robust container security," in 2019 IEEE Conference on Multimedia Information Process- ing and Retrieval (MIPR). IEEE, 2019, pp. 467–470.
4	Y. Wu, E. Dobriban, and S. Davidson, "Deltagrad: Rapid retraining of machine learning models," in International Conference on Machine Learning. PMLR, 2020, pp. 10 355–10 366.
5	D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden tech- nical debt in machine learning systems," Advances in neural information processing systems, vol. 28, pp. 2503–2511, 2015.
6	S. Alla and S. K. Adari, "What is mlops?" in Beginning MLOps with MLFlow. Springer, 2021, pp. 79–124.
7	Z. Wan, X. Xia, D. Lo, and G. C. Murphy, "How does machine learning change software development practices?" IEEE Transactions on Software Engineering, 2019.
8	C. T. Wolf and D. Paine, "Sensemaking practices in the everyday work of ai/ml software engineering," in Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops, 2020, pp. 86–92.
9	"Empowering app development for developers — docker," https://www.docker.com/ , (Accessed on 09/13/2021).
10	G. Sayfan, Mastering kubernetes. Packt Publishing Ltd, 2017. [11] "Amazon sagemaker – machine learning – amazon web services," https://aws.amazon.com/sagemaker/ , (Accessed on 09/13/2021).
11	"Introduction to kubeflow — kubeflow," https://www.kubeflow.org/docs/about/kubeflow/ , (Accessed on 09/02/2021).
12	M. Zaharia, A. Chen, A. Davidson, A. Ghodsi, S. A. Hong, A. Konwin- ski, S. Murching, T. Nykodym, P. Ogilvie, M. Parkhe et al., "Accelerat- ing the machine learning lifecycle with mlflow," IEEE Data Eng. Bull., vol. 41, no. 4, pp. 39–45, 2018.
13	"Gitops — gitops is continuous deployment for cloud native applica- tions," https://www.gitops.tech/ , (Accessed on 09/02/2021).
14	C. Singh, N. S. Gaba, M. Kaur, and B. Kaur, "Comparison of different ci/cd tools integrated with cloud platform," in 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Conflu- ence). IEEE, 2019, pp. 7–12.
15	J. Turnbull, The Docker Book: Containerization is the new virtualization. James Turnbull, 2014.
16	J. Rufino, M. Alam, J. Ferreira, A. Rehman, and K. F. Tsang, "Orchestra- tion of containerized microservices for iiot using docker," in 2017 IEEE International Conference on Industrial Technology (ICIT). IEEE, 2017, pp. 1532–1536.
17	"Mlops: Continuous delivery and automation pipelines in machine learning," https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning , (Accessed on 08/13/2021).
18	I. Karamitsos, S. Albarhami, and C. Apostolopoulos, "Applying devops practices of continuous automation for machine learning," Information, vol. 11, no. 7, p. 363, 2020.
19	"Detecting data drift with mlops — 10clouds," https://10clouds.com/blog/detecting-data-drift-mlops/ , (Accessed on 09/7/2021).
20	T. A. Limoncelli, "Gitops: a path to more self-service it," Communica- tions of the ACM, vol. 61, no. 9, pp. 38–42, 2018.
21	"Gitops — gitops is continuous deployment for cloud native applica- tions," https://www.gitops.tech/ , (Accessed on 09/15/2021).
22	V. Lenarduzzi, F. Lomio, S. Moreschini, D. Taibi, and D. A. Tamburri, "Software quality for ai: Where are we now?" in International Confer- ence on Software Quality. Springer, 2021, pp. 43–53.
23	C. Renggli, B. Karla's, B. Ding, F. Liu, K. Schawinski, W. Wu, and C. Zhang, "Continuous integration of machine learning models with ease. ml/ci: Towards a rigorous yet practical treatment," arXiv preprint arXiv:1903.00278, 2019.
24	J. Lim, H. Lee, Y. Won, and H. Yeon, "Mlop lifecycle scheme for vision-based inspection process in manufacturing," in 2019 {USENIX} Conference on Operational Machine Learning (OpML, 19), 2019, pp. 9–11.
25	C. Murphy, G. E. Kaiser, and M. Arias, "A framework for quality assurance of machine learning applications," 2006.
26	J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning test- ing: Survey, landscapes and horizons," IEEE Transactions on Software Engineering, 2020.

D.22. Un SMS de herramientas que implementan la metodología GitOps -

página 22

27	T. Y. Chen, "Metamorphic testing: A simple method for alleviating the test oracle problem," in 2015 IEEE/ACM 10th International Workshop on Automation of Software Test. IEEE, 2015, pp. 53-54.
28	J. Wang, L. Li, and A. Zeller, "Better code, better sharing: on the need of analyzing jupyter notebooks," in Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results, 2020, pp. 53-56.
29	"Quality assurance for machine learning models - part 1," https://blog.sasken.com/quality-assurance-for-machine-learning-models-part-1-why-quality-assurance-is-critical-for-machine-learning-models , (Accessed on 08/25/2021).

Adelante (22)

No	Study
1	Tercan, H., & Meisen, T. (2022). Machine learning and deep learning based predictive quality in manufacturing: a systematic review. <i>Journal of Intelligent Manufacturing</i> , 33(7), 1879-1905.
2	Shankar, S., Garcia, R., Hellerstein, J. M., & Parameswaran, A. G. (2022). Operationalizing machine learning: An interview study. <i>arXiv preprint arXiv:2209.09125</i> .
3	Zhang, S., Jia, F., Wang, C., & Wu, Q. (2022, September). Targeted hyperparameter optimization with lexicographic preferences over multiple objectives. In <i>The Eleventh International Conference on Learning Representations</i> .
4	Sundberg, L., & Holmström, J. (2023). Democratizing artificial intelligence: How no-code AI can leverage machine learning operations. <i>Business Horizons</i> .
5	Robertson, S., Wang, Z. J., Moritz, D., Kery, M. B., & Hohman, F. (2023, April). Angler: Helping Machine Translation Practitioners Prioritize Model Improvements. In <i>Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems</i> (pp. 1-20).
6	Filippou, K., Aifantis, G., Papakostas, G. A., & Tsekouras, G. E. (2023). Structure Learning and Hyperparameter Optimization Using an Automated Machine Learning (AutoML) Pipeline. <i>Information</i> , 14(4), 232.
7	Pineda-Jaramillo, J., & Viti, F. (2023). MLOps in freight rail operations. <i>Engineering Applications of Artificial Intelligence</i> , 123, 106222.
8	Naydenov, N., & Ruseva, S. (2022, March). Combining container orchestration and machine learning in the cloud: A systematic mapping study. In <i>2022 21st International Symposium INFOTEH-IAHORINA (INFOTEH)</i> (pp. 1-6). IEEE.
9	Barry, M., Bifet, A., & Billy, J. L. (2023, May). StreamAI: Dealing with Challenges of Continual Learning Systems for Serving AI in Production. In <i>2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)</i> (pp. 134-137). IEEE.
10	Das, S. D., & Bala, P. K. (2023). What drives MLOps adoption? An analysis using the TOE framework. <i>Journal of Decision Systems</i> , 1-37.
11	Khattak, F. K., Subasri, V., Krishnan, A., Dolatabadi, E., Pandya, D., Seyyed-Kalantari, L., & Rudzicz, F. (2023). MLHops: Machine Learning for Healthcare Operations. <i>arXiv preprint arXiv:2305.02474</i> .
12	Chen, R., Pu, Y., Shi, B., & Wu, W. (2023). An automatic model management system and its implementation for AIOps on microservice platforms. <i>The Journal of Supercomputing</i> , 79(10), 11410-11426.
13	Yulianto, Y., & Utami, E. (2022, December). Automatic Deployment Pipeline for Containerized Application of IoT Device. In <i>2022 6th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)</i> (pp. 81-86). IEEE.
14	Díaz Gil, F. (2022). Metodologías de control de calidad y pruebas para soluciones tecnológicas basadas en AI/ML y Big Data.
15	Krishna, M. Y. S., & Gawre, S. K. (2023). MLOps for Enhancing the Accuracy of Machine Learning Models using DevOps, Continuous Integration, and Continuous Deployment. <i>Research Reports on Computer Science</i> , 97-103.
16	Vega Arellano, J. P. (2022). Stroke Detection with Deep Learning (Doctoral dissertation, SRH Hochschule Heidelberg).
17	Yau, T. C. (2023). Investigate the challenges and opportunities of MLOps
18	Fathurahman, F., Minarno, A. E., & Faiqurahman, M. (2023). AUTOMATE DEPLOYMENT APLIKASI WEB MENGGUNAKAN METODE GITOPS PADA KUBERNETES CLUSTER (STUDI KASUS: WEB RISET INFORMATIKA UNIVERSITAS MUHAMMADIYAH MALANG). <i>Elektrika Borneo</i> , 9(1).
19	Malgund, S. J., & Sowmyarani, C. N. AUTOMATING DEPLOYMENTS OF THE LATEST APPLICATION VERSION USING CI-CD WORKFLOW.
20	Fathurahman, F. (2022). Automate Deployment Aplikasi Web Riset Menggunakan Metode Gitops Pada Kubernetes Cluster (Doctoral dissertation, Universitas Muhammadiyah Malang).
21	Bechu, G., Beugnard, A., Cao, C. G., Perez, Q., Urtado, C., & Vauttier, S. (2022, November). Déploiement dirigé par les modèles de jumeaux numériques dans les environnements intelligents. In <i>HUT LaConf 2022-L'interdisciplinarité au service des environnements intelligents</i> .
22	Colliander, J. (2022). Radiosignaalien tunnistaminen neuroverkon avulla (Master's thesis).

D.23. Un SMS de herramientas que implementan la metodología GitOps -

página 23

La siguiente tabla muestra los resultados de la bola de nieve

GitOps: The Evolution of DevOps? & Analysis on the Use of Declarative and Pull-based Deployment Models on GitOps Using Argo CD

No	Study	B	F
1	Leiter, Á., Hegyi, A., Kispál, I., Böösy, P., Galambosi, N., & Tar, G. Z. (2023, May). GitOps and Kubernetes Operator-based Network Function Configuration. In NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium (pp. 1-5). IEEE.	0	1
Subtotal		0	1
Total resultados bola de nieve		1	1