GitOps: A review of current state, challenges and future potential in cloud-native deployments

Craig Dillon
School of Enterprise Computing and Digital Transformation
Technological University Dublin
Dublin, Ireland
x00205790@mytudublin.ie

***Abstract*—Modern DevOps lifecycles, often hosted using microservices and cloud-based container platforms, encounter challenges in maintaining ephemeral infrastructure. GitOps has emerged as a framework that builds on DevOps principles to provide a platform for deployment and management of cloud-native DevOps lifecycles. This approach aims to address the intricacies of cloud-native DevOps practices, relying on version-controlled repositories as the single source of truth for applications and infrastructure.**

This paper seeks to provide a balanced examination of the use of GitOps principles in cloud-native deployments and exploring how this relationship can impact scalability, reliability and operational efficiency in modern DevOps lifecycles. In performing this exploration, this paper will identify gaps in existing literature, providing potential avenues for further research which will contribute valuable insights into the rapidly evolving DevOps ecosystem.

Keywords— DevOps, GitOps, Cloud-native, Containerisation, CI/CD, Continuous Integration, Continuous Deployment

I. Introduction

A. Background

Organisations are increasingly adopting cloud-native offerings on public cloud platforms, which causes a shift in their operational practices. GitOps is a framework and rapidly evolving paradigm that offers a strategic approach based on DevOps methodologies and principles to manage the automation of cloud-native infrastructure. The relationship between GitOps and DevOps is particularly effective in the context of cloud-native computing where there is an emphasis on scalability, rapid deployment and cost efficiency.

One of the fundamental strengths of GitOps is the ability to extend traditional DevOps workflows to effectively manage cloud-native deployments by employing version-controlled repositories as the single source of truth for both application and infrastructure configurations.

This paper aims to review the differences between DevOps and GitOps, looking closely the strategic benefits their relationship has. Despite limited research in a rapidly evolving field, this paper endeavours to contribute insights for teams or organisations considering integrating GitOps into their DevOps operations.

B. Motivation

The adoption of cloud-native computing by organisations is rapidly increasing which results in the growing relationship between DevOps and GitOps. This literature review will explore how GitOps incorporates into Continuous Integration/Continuous Deployment (CI/CD) pipelines for cloud-native deployments and the potential advantages this brings, providing insight and knowledge for teams or organisations to make informed decisions when considering adopting GitOps. Some key questions that will be answered are;

- What are the main challenges of adopting GitOps?
- What are the benefits of adopting GitOps?
- What tools and skills are required?

C. Contribution

This review consolidates and synthesises existing literature on DevOps and GitOps, drawing from a number of journals and conference proceedings to provide an overview of the similarities, differences and relationship between the two paradigms, ultimately identifying gaps in current literature and offering potential future research areas.

II. Theoretical background

A. Overview of DevOps

DevOps is a mature and well understood methodology that has seen widespread adoption in the IT industry. Key principles that form DevOps include, automation, collaboration and communication, Continuous Integration, Continuous Delivery or Deployment (CI/CD) and continuous improvement, to name a few. DevOps is not only a framework for software development and delivery but also brings organisational cultural changes, usually in the form of Lean and Agile methodologies. To sum it up, *"DevOps is the culture, process, and technology to foster close collaboration of software development and IT operations"*, Ebert & Hochstein (2022).

B. Overview of GitOps

GitOps builds on DevOps practices with clearly defined axioms, Vinto & Bueno (2022) states *"The three main pillars of GitOps are:*

- *Git is the single source of truth*
- *Treat everything as code*

- *Operations are performed through Git workflows*

And the GitOps Principles are defined as;

- *Declaritive - A system managed by GitOps must have its desired state expressed declaratively*
- *Versioned and immutable - The desired state is stored in a way that enforces immutability and versioning retains a complete version history*
- *Pulled automatically - Software agents automatically pull the desired state declarations from the source*

It is important to note that the last item in this reference is not the only deployment method in GitOps, an alternative approach will be discussed in a later section.

C. Challenges of DevOps in a cloud-native world

Current DevOps solutions in cloud-native computing are often *"ad hoc, non-repeatable and unpredictable"* while being tightly bound to proprietary DevOps tools that hinder adaptability, Dalla Palma et al. (2022). In one study, DevOps deployments to a cloud based microservices platform showed noticeable latency variations in responsiveness of applications when scaling horizontally, Jindal & Gerndt (2021). In relation to cloud-native computing, the current DevOps technology stack is found to be lacking in comprehensive tooling that allows users to improve code quality, visualise code metrics or perform code smell detection, Dalla Palma et al. (2022).

D. Key concepts of GitOps

GitOps can operate in either a push-based method or a pull-based method, with both having benefits and disadvantages that must be considered when deciding to employ GitOps as part of a CI/CD workflow. With push-based GitOps changes are initiated when developers actively push updates to the repository which then triggers automated workflows that immediately apply the desired configuration to the deployment environment, Beetz & Harrer (2021).

Pull-based GitOps makes use of an *"operator"* and is often used in tandem with *"container orchestration platforms, such as Kubernetes"* Beetz & Harrer (2021). The key difference in this approach is that the operator actively monitors for changes in both the repository and deployment environment, applying changes when they are pushed to the repository and preventing deviation of the deployment environment by automatically restoring the desired state.

Infrastructure as Code (IaC) plays a key role in GitOps as it can then perform a wide range of infrastructure updates through the version control system. GitOps performs the function of IaC and provides *"Orchestration, observability, declarative IaC, containers and immutable infrastructures, DevOps best practices"*, Gupta et al. (2022).

E. GitOps in CI/CD pipelines

GitOps builds on traditional CI/CD pipelines by providing a *"self-service"* platform for developers to deploy infrastructure continuously, Beetz & Harrer (2021). An example workflow would be;

- Work is committed to the repository
- CI tools are then triggered which builds the application, runs automated tests, code analysis and other processes associated with CI.
- If the CI process is successful then changes are automatically committed to the relevant repository, branch or in the case of containers, registry.
- At this point the GitOps operator is actively monitoring for changes and applies them to the relevant deployment environment

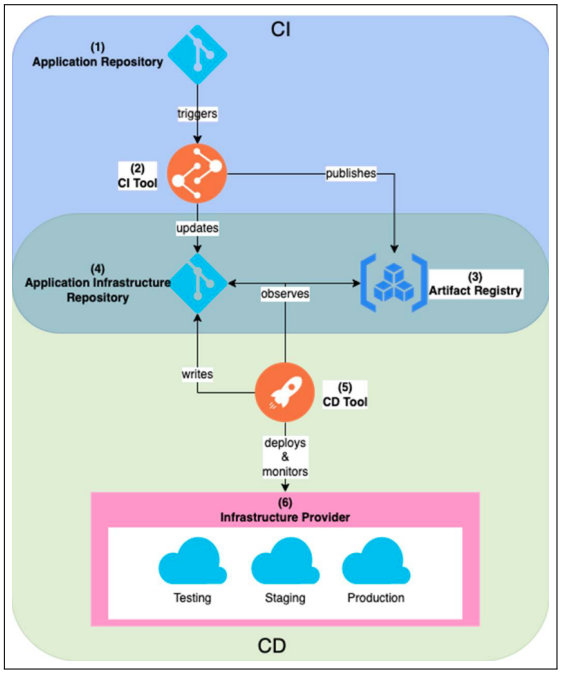

Fig. 1. Example GitOps workflow(Source: López-Viana et al. (2022))

F. Tooling

The GitOps ecosystem has an active and supportive community with many tools and projects that are part of the Cloud Native Computing Foundation (CNCF) support program, Kormaník & Porubän (2023). An example toolset for a GitOps framework would be;

- Gitlab for CI tasks and version control
- Docker to build containers
- Container registry
- ArgoCD as the GitOps operator, also providing a visual dashboard
- Kubernetes as the orchestration platform
- Helm and Helm Operator, Helm is a Kubernetes package manager simplifying the deployment and management of applications and Helm Operator automates these deployments, enabling lifecycle management of Kubernetes applications
- Terraform for IaC
- Prometheus and Grafana for performance and service health monitoring

López-Viana et al. (2022), Kormaník & Porubän (2023), Utami et al. (2021), Gupta et al. (2022)

III. Benefits and challenges of GitOps in CI/CD

A. Efficiency gains

GitOps has been shown to have code efficiencies over similar DevOps deployments with a reduction in overall lines of code, lines of code per files and a logical file structure that is more readable, Kormaník & Porubän (2023). YAML (Yet Another Markup Language) is the predominant configuration language in GitOps, is human-readable and while there are strict syntax rules it is generally well understood by developers.

In a survey into GitHub Actions, Saroar & Nayebi (2023) found that 39% of developers had no challenges writing in YAML, with syntax being the largest challenge and learning curve the lowest rated challenge.

```
apiVersion: azure.upbound.io/v1beta1
kind: ResourceGroup
metadata:
  name: k8s-testing
  annotations:
    crossplane.io/external-name:
        k8s_testing
spec:
  forProvider:
    location: northeurope
  providerConfigRef:
    name: k8s-testing
```

Listing 1. Example YAML configuration for creating an Azure Resource Group using Crossplane, (Source: Author, 2023)

Other benefits are improved auditability and observability via metrics and events provided by the GitOps operator, disaster recovery and business continuity scenarios due to automatic change promotion and drift detection, López-Viana et al. (2022). GitOps can also provide faster and more secure deployments, tool independent architecture, while backups are a feature of GitOps due to everything being stored as code in a repository along with the ability to compare and track environments easily, Gupta et al. (2022).

B. Challenges and considerations

Adopting GitOps into a DevOps CI/CD pipeline does carry some potential challenges. Kubernetes is the most common orchestration platform for managing and deploying containerised applications which means that working knowledge of Kubernetes is a requirement, López-Viana et al. (2022). Possible issues would arise around the use of YAML where syntax or object references may be broken, Gupta et al. (2022), leading to possible outages and other issues if not maintained properly.

IV. Limitations

GitOps is a relatively new paradigm in the IT operations space, with Gartner considering GitOps an innovation trigger in 2023, Perri (2023). Taking this into consideration, there are not a great amount academic papers covering the topic in detail compared to other well established methodologies like DevOps, for example. GitOps is a rapidly developing field which means that some reference materials may be missing information that is at the bleeding edge, the author noticed a lack of information regarding Crossplane and Pulumi, both IaC tools, and Werf, a full CI/CD tool designed for creating, testing and deploying containers to Kubernetes.

V. Conclusion

GitOps is a dynamic, quickly evolving ecosystem built on the principals and methodologies of DevOps. While there are some differences between the two frameworks, they are not mutually exclusive. As the IT industry continues to adopt serverless technologies, GitOps provides a framework that builds on the core tenets of DevOps that is suited to the cloud-native environment. At its core, GitOps is an extension of traditional DevOps CI/CD pipelines, simplifying the process, in particular Continuous Deployment in relation to ephemeral, serverless architectures. GitOps utilises version-controlled repositories with declaritive code to provide streamlined workflows that are scalable and can effectively manage event-driven workflows and transient resources.

GitOps, however, is not a one-size-fits-all solution and organisations need to assess their specific needs with careful consideration in relation to knowledge gaps in the technology stack, in particular Kubernetes. If organisations are not heavily reliant on cloud-native computing then GitOps as a framework may not be a worthwhile endeavour, although some concepts such as Git being the single source of information, treating everything as code and declaritive statefulness may be worth considering in isolation.

In conclusion, GitOps is a paradigm set to play a key role in shaping the future of IT operations by removing complexities of infrastructure management, allowing developers to direct their focus towards innovation and application development. As the IT industry continues its rapid adoption of cloud-native technologies, GitOps stands out as an effective means of deploying, managing and maintaining the statefulness of inherently transient services. GitOps is a strategic enabler for organisations adopting cloud-native computing, allowing teams to strike a balance between agility, scalability and simplicity in their CI/CD pipelines.

VI. Future work

As this is a rapidly changing ecosystem, there are several avenues of further research. For example, other papers focus on using Kubernetes as the orchestration platform and this can bring significant overhead and cost to inexperienced teams or those with a restricted budget.

A. GitOps on a lightweight Kubernetes platform

Kubernetes has matured rapidly in recent years and there are several options relating to small scale implementations like k3s, Minikube and MicroK8s that would be worth exploring, especially in relation to hosting GitOps related tooling. These small scale distributions may be beneficial to teams who have resource or budget limitations, topics worth exploring are;

- Scalability - can these lightweight platforms scale effectively?

- Tooling - do GitOps tools work on these platforms? Are there key dependencies missing?
- Resource utilisation and performance - will the lightweight platforms handle increasingly large GitOps workloads?

B. GitOps in multi-cloud and hybrid environments

Crossplane is a GitOps IaC tool that has matured quickly and can deploy applications, services and infrastructure across a range of platforms, research into its effectiveness across multi-cloud or even hybrid on premise/cloud environments. Crossplane relies on providers to deploy to non Kubernetes platforms, which includes a Terraform provider that can make use of Terraform configurations. Further research into the challenges, effectiveness and ease of use would be of great benefits to teams who are considering or find themselves straddling a multi-cloud platform that makes use of a range of infrastructure like microservices, VMs and on premise environments.

C. Exploring cutting edge GitOps tools

New tools and products are regularly released within the GitOps ecosystem, often with the goal of simplifying the GitOps approach for teams with little Kubernetes experience. Werf is a tool that is worth consideration for further research, it aims to simplify the CI/CD process when creating containerised services by building, testing, pushing to a container registry and deploying with one command using Helm. This tool also benefits from other features such as instant rollback, multi-environment support and efficient container lifecycle management. GitLab also provide GitLab for GitOps where Terraform and Kubernetes are integrated as the IaC and orchestration tools. A comparison, especially in relation to teams who are already invested in the GitLab ecosystem is worth deeper analysis.

D. GitOps in industry specific cases

Examining the adoption or possible benefits of GitOps in certain industries would provide valuable insight into business needs and ability to gain a competitive advantage. Some industries and sectors that could be investigated are;

- Finance - in particular FinTech and investment banking
- Energy - renewables and smart grid/agile pricing
- E-Commerce - supply chain management and online retail platforms
- Technology - software development, AI and cybersecurity

E. GitOps in multi-cloud and hybrid environments

While undertaking this study, two other methodologies came to light and present a possible avenue for comparison to DevOps and GitOps in a cloud-native setting.

- RADON - an EU funded research project that provides a framework for FaaS (Function as a Service) within a single IDE and built on top of DevOps principles
- NoOps - a strategy that aims to minimise the involvement of operations by utilising high levels of automation.

REFERENCES

Beetz, F. & Harrer, S. (2021), 'Gitops: The evolution of devops?', *IEEE Software* **39**(4), 70–75.

Dalla Palma, S., Catolino, G., Di Nucci, D., Tamburri, D. A. & van den Heuvel, W.-J. (2022), 'Go serverless with radon! a practical devops experience report', *IEEE Software* **40**(2), 80–89.

Ebert, C. & Hochstein, L. (2022), 'Devops in practice', *IEEE Software* **40**(1), 29–36.

Gupta, S., Bhatia, M., Memoria, M. & Manani, P. (2022), Prevalence of gitops, devops in fast ci/cd cycles, *in* '2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON)', Vol. 1, IEEE, pp. 589–596.

Jindal, A. & Gerndt, M. (2021), From devops to noops: Is it worth it?, *in* 'Cloud Computing and Services Science: 10th International Conference, CLOSER 2020, Prague, Czech Republic, May 7–9, 2020, Revised Selected Papers 10', Springer, pp. 178–202.

Kormaník, T. & Porubän, J. (2023), Exploring gitops: An approach to cloud cluster system deployment, *in* '2023 21st International Conference on Emerging eLearning Technologies and Applications (ICETA)', IEEE, pp. 318–323.

López-Viana, R., Díaz, J. & Pérez, J. E. (2022), Continuous deployment in iot edge computing: A gitops implementation, *in* '2022 17th Iberian Conference on Information Systems and Technologies (CISTI)', IEEE, pp. 1–6.

Perri, L. (2023), '4 exciting new trends in the gartner emerging technologies hype cycle'. accessed 28/12/2023.
URL: *https://www.gartner.com/en/articles/what-s-new-in-the-2023-gartner-hype-cycle-for-emerging-technologies*

Saroar, S. G. & Nayebi, M. (2023), 'Developers' perception of github actions: A survey analysis', *arXiv preprint arXiv:2303.04084* .

Utami, E., Al Fatta, H. et al. (2021), Analysis on the use of declarative and pull-based deployment models on gitops using argo cd, *in* '2021 4th International Conference on Information and Communications Technology (ICOIACT)', IEEE, pp. 186–191.

Vinto, N. & Bueno, A. (2022), *GitOps Cookbook*, O'Reilly Media.