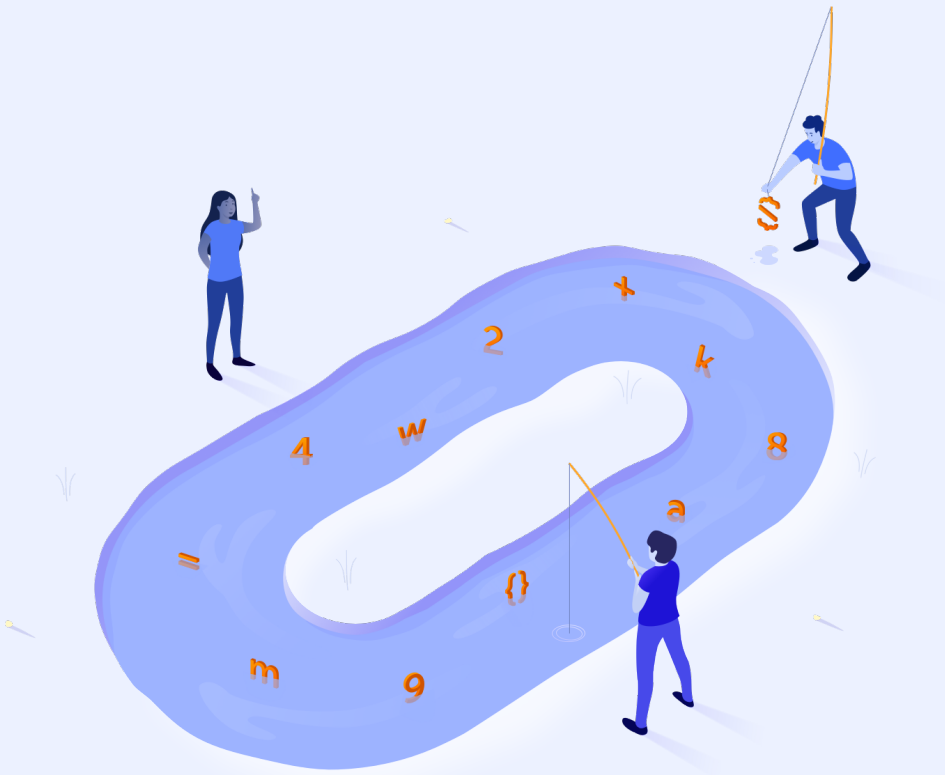


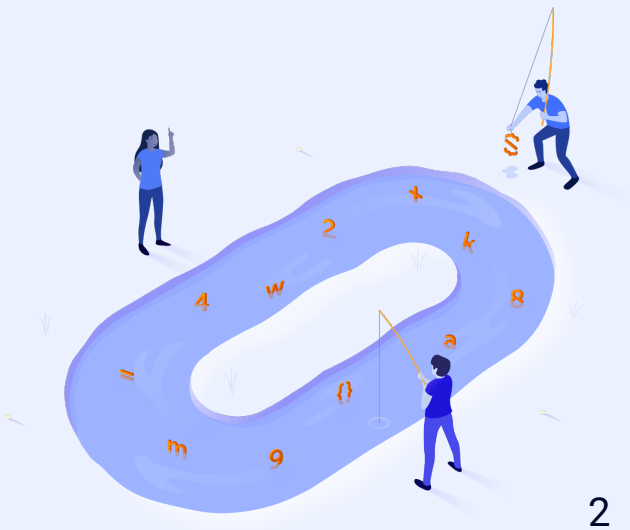
SPRING 2022

Get up to speed on GitOps

Adaptavist

Get up to speed on GitOps

Contents



Introduction

DevOps transformed the software development life cycle (SDLC), automating processes and bringing teams together to speed up and scale development. Organisations that have embraced DevOps can deploy code to production hundreds of times a day, and customers are reaping the benefits. With mature DevOps processes in place, including infrastructure automation and IaC (infrastructure as code), infrastructure management becomes efficient and repeatable. GitOps is a natural next step.

In this ebook, we explain what GitOps is and get under the skin of this operational framework. We'll also cover:

- How you 'do' GitOps
- GitOps core components
- The four principles of GitOps
- The difference between DevOps and GitOps
- Benefits GitOps brings to your organisation

Let's get started...

Part one

What is GitOps?

GitOps is an operational framework applied to infrastructure and configuration automation. It incorporates software development lifecycle best practices, such as collaboration, version control, continuous delivery/deployment (CD), and compliance, and applies them to infrastructure automation. GitOps is a selection of developer-centric practices which are applied to other aspects of the application lifecycle. With GitOps, teams can easily manage infrastructure using the same software development tools and processes they're already familiar with.

Unlike IaC, which is version-control-system agnostic, GitOps leverages Git and its fundamental patterns. It follows a branch, code, check, merge (or pull request) workflow to develop the codebase. Then you have a GitOps operator that sits between the pipeline and whichever orchestration system you use. Its job is to pick up the code changes by commit(s) and pull in the new state declaration from Git. In part two, we take a closer look at what a GitOps workflow looks like.

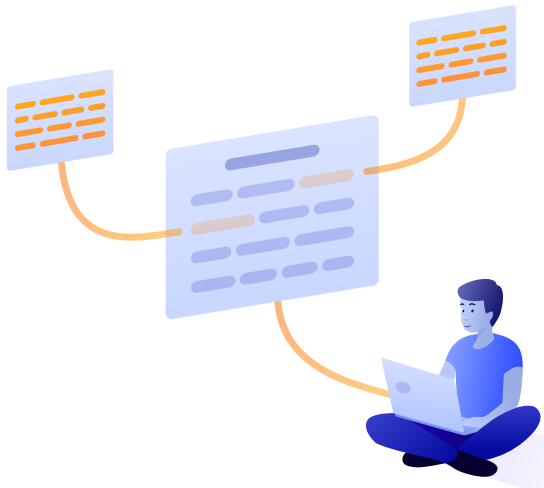
Where does GitOps come from?

To understand the principles that underpin GitOps, it's time for a little history lesson. Back before the internet ruled the world, software development was a physical, segmented process. Dedicated computers with real wires and networking devices hosted applications. And real humans kept them running – from the developers who wrote

the code and the system administrators who looked after the servers to the network teams who ensured all those computers were connected. It was neat and tidy, but it was slow.

Then, with virtualisation, things changed almost overnight. Instead of calling the systems department to get a computer to do the work, developers would use software and scripts to define infrastructure components, either in a web browser or from a command line to construct a virtual machine and virtual networks. Physical infrastructure had been replaced by ephemeral infrastructure that used software definitions to achieve the same result. The code for that software was stored in a repository, alongside code for all of an organisation's virtual assets – a single source of truth from infrastructure to application.

This infrastructure-as-code (IaC) approach is at the heart of GitOps. In GitOps, everything from applications and networking to computing resources and storage is represented as code. Therefore, the review, merge, and execution can be controlled through automation.



Part two

How does GitOps work?

GitOps helps you effectively manage infrastructure and deployment pipelines. As we mentioned earlier, Git is at the heart of GitOps.
Git is an open source distributed version control system that helps developers coordinate when they're working on the same code.
Git is the most widely used version control system in the world.
It's an essential tool in many organisations' DevOps practice and sits at the heart of GitOps.

GitOps started as a way to manage Kubernetes – an open-source container orchestrating system for automating software deployment, scaling, and management – but it's evolved. While it's still an effective way to manage Kubernetes, and Kubernetes remains the popular choice to implement GitOps, it can also be applied to all sorts of infrastructure automation options, including virtual machines (VMs) and containers.

With Git as the single source of truth, GitOps automated processes alert your teams if there's a change, triggering automatic infrastructure deployments or rollbacks to converge on the state stored in Git. This saves developers time, creates stability, and can result in huge cost savings for the business.

How do you ‘do’ GitOps?

Now you have a bit more understanding of what GitOps is, you might be wondering how you make it happen. First, you’ll need to determine if your organisation would benefit from GitOps – asking questions around whether you’re prepared to embrace GitOps’ software delivery culture, and taking a close look at your application delivery pipeline. Engage with DevOps and IT teams to find out if it’s the right fit.

Then, if GitOps sounds like the framework for you, the next step is to design and implement best practice processes for your infrastructure team to stick to and bring new solutions on board. It might make sense to involve external experts, like Adaptavist, while leaning on existing knowledge within your organisation. If that knowledge is lacking, then you should hire a specialist or consider a training programme to bring your people up to speed.

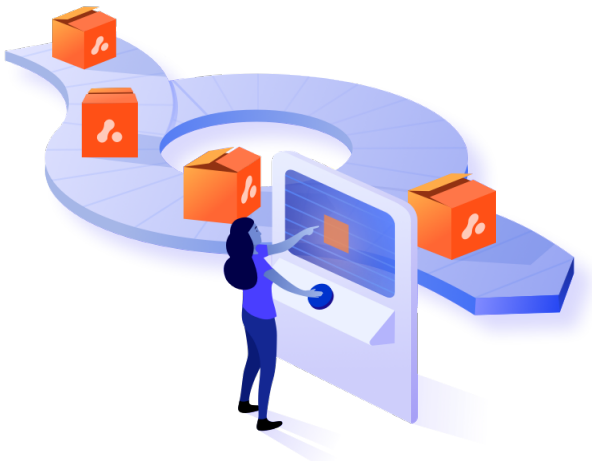
Other than using Git, to do GitOps, you don’t need a specific product or platform. But you will need to incorporate three core components into your workflow: IaC, pull requests, and CD.

- IaC, as mentioned earlier, refers to the practice of keeping all your infrastructure and/or configurations stored as code. You’ll need a Git repository to track any changes made to a project.

- Pull requests are the change mechanism used for infrastructure updates in GitOps. It’s where teams come together to collaborate and how approvals happen. Merging an approved pull request promotes the updated infrastructure definition,

and a GitOps operator will pull those changes to update the environments as necessary.

- CD refers to your pipeline, which continuously applies the state changes to your environment. If there are errors, GitOps automation can roll back the changes to the previous working version, ensuring environmental stability. If there are manual changes made, GitOps automation can enforce the desired state of the environment based on the declared configuration in source control.



Part three

GitOps principles

There are four key principles that apply when implementing your GitOps initiative.

1. A declarative system

First up, is the fact that GitOps is a declarative system, rather than an imperative system. A declarative system is one which is focused on the result or 'end-state', rather than the steps you need to take to achieve it – you 'declare' the desired state of your application or cluster. Users can specify the desired state, making it easy to store these states in Git.

2. Versioned and immutable

With Git as your single source of truth, your desired state is stored in a way that enforces immutability and versioning. With a complete version history, you can easily revert to a previous application state and ensure any commits must meet strict security measures.
And because all the changes you make to system infrastructure are stored chronologically, you can easily identify and analyse changes over time, making audits a breeze.

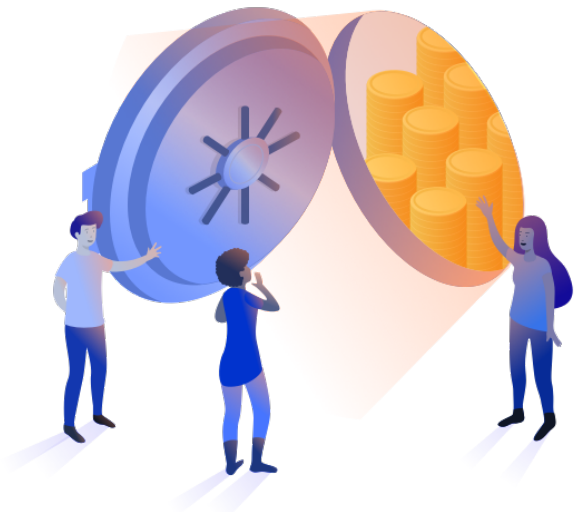
3. Pulled automatically

With your declared state held in Git, software agents automatically pull desired state declarations from the source. That way, you can relax knowing any changes made meet your demands. Once you've updated and approved it, automation means that state will be consistently applied with zero effort from your teams.

4. Continuously reconciled

Human error is inevitable. Fortunately, GitOps workflows are self-healing, so you can have peace of mind. Software agents are at work in the background, continuously observing the actual system state and attempting to apply the desired state. And, if anything changes unexpectedly or out of sync with that state, you'll be alerted immediately.

So there you have it: declarative, versioned, automatic, and continuously reconciled. The four core principles that GitOps is based on. With that in mind, let's look at how GitOps relate to DevOps, and how they're different.



Part four

GitOps vs DevOps: what's the difference?

If you're interested in GitOps, chances are your organisation is already practising DevOps. But how are the two related and what sets them apart?

The main thing to remember is that DevOps involves a cultural change, bringing development, operations, and tools together to help organisations build and improve products faster than ever before. It's about encouraging and enabling teams to work in a more cohesive and collaborative way, and removing siloed thinking from the business.

Organisations with a mature DevOps culture depend on continuous integration/continuous deployment, and are focused on automation and frequent deployments. The tools you might use as part of your DevOps approach can vary. By doing DevOps, organisations can deploy more often, develop new features, release updates more quickly, and fix any issues in an instant.

What's special about GitOps?

GitOps, on the other hand, is a specific operation process that relates to a specific tool – Git. It's an advanced application of DevOps principles, a best-practice approach that optimises Git as your organisation's single source of truth. It provides an operational model for managing infrastructure provisioning and software deployments,

enabling developers to manage and trigger deployments using Git. GitOps principles are relevant to different types of infrastructure automation, including Kubernetes, VMs, and containers.

DevOps and GitOps share a lot of the same principles and practices. In short, if you already have a DevOps culture, you might choose to use GitOps to manage your infrastructure. If you do, your DevOps experience will make it a lot easier to incorporate GitOps into your workflow.

Key benefits of GitOps

For leadership and key decision-makers not dealing with infrastructure on a day-to-day basis, the benefits of GitOps might seem vague and niche. But they would be wrong to dismiss it. For those already in the know, having a framework to develop, deploy, and scale your infrastructure, is integral to business success. GitOps can have a huge impact, driving efficiency, improving security, and helping you achieve commercial goals. Here, we explore those benefits in more detail.

Power productivity and collaboration

At the heart of GitOps are best practice approaches for IaC, CD pipelines, and Git workflows. For your DevOps teams, adopting GitOps won't be a huge leap – they'll already have a lot of the skills and knowledge required to make it happen. But the pay-off will be significant. With simplified processes and a streamlined review and approval process, they'll have enhanced visibility and be able to collaborate more easily with their colleagues.

Speed up development and deployment

Time is money, so speed is essential to stay competitive. GitOps ensures faster and more frequent deployments by helping eliminate environmental configuration differences that cause lost developer time in troubleshooting and correcting issues. This lets your team ship more changes per day and increase their output. As a result, you can deliver more business and customer value, optimising customer feedback by responding quickly and meeting their needs. And if there's a problem, it's easy to roll back to a previous state.

Make your developers happy

'It worked on my machine' is something we've all heard before. Making application environment configurations predictable and repeatable lets developers do what they like doing – coding. They're no longer spending hours of time troubleshooting potential issues or manually deploying or configuring servers and application components. Employee satisfaction and retention is vital for business success. And GitOps can help. Automation means developers can focus on developing rather than time-consuming, manual tasks.

Reduce costs

With infrastructure automation, you'll see productivity soar and downtime take a dive, thanks to built-in deployment and roll-back functionality, saving the business significant toil. GitOps also makes it easier to manage your cloud resources, which alone can have a knock-on effect to your bottom line.

Be compliant, be stable

With GitOps, your teams can iterate faster without living in fear of an unstable, unreliable environment. That single source of truth means it's easy and quick to recover if an iteration fails, simply rollback to a previous state. What's more, with Git by your side, you get an audit trail of all cluster changes, so you can keep track of who did what for code reviews and compliance.

Stay secure without compromise

Strict compliance often means decreased collaboration and limits on who can make changes in production. With GitOps, pull requests mean anyone can propose changes without impacting the integrity of the production branch. With automated changes and a running history to refer to, everyone can make a contribution without compromising the security of your applications or clusters. Rather than needing additional credentials or access from the outside, changes are applied from the inside by the GitOps operator when it sees changes in Git.

Conclusion

Speed up and evolve with GitOps

We hope this ebook has given you a good grounding in GitOps, getting you up to speed on the principles and practices behind it, and explaining some of the key benefits your business stands to gain. It's clear that GitOps is a powerful addition to your DevOps culture to speed and scale infrastructure automation. To make it part of your approach, the next step is designing best-practice processes for your team to follow.

At Adaptavist, it's our mission to deliver end-to-end DevOps services and solutions to help you build and implement CI/CD systems.
We use the latest industry-proven techniques to maximise the benefits of your IT investment. From DevOps implementation and strategic guidance to coaching, training, and DevOps as a service.

We also partner with some of the world's leading GitOps solution providers to help you drive infrastructure automation and collaboration. From migration and integration to training your teams, we're here to help.

GitOps takes your infrastructure automation to the next level.
Ready to find out more?

Get in touch

Adaptavist

We help organisations transform to continuous change being their business as usual. We do this by supplying technology, providing advice, and delivering change through modern, iterative approaches to development, deployment, and application lifecycle management.

Adaptavist is Atlassian's largest platinum partner, supporting more than half of the Fortune 500. We are uniquely placed to provide our experience, expertise, and insight to help your business.

Whether you want training for your team, to build a software platform for your company, or to automate your existing tooling, we can help you. If you want to unlock the full power of Atlassian and transform your business at scale, get in touch with our team today.

Learn more or get in touch:

adaptavist.com

ATLASSIAN Training Partner

ATLASSIAN Platinum Solution Partner ENTERPRISE

ATLASSIAN Platinum Top Vendor