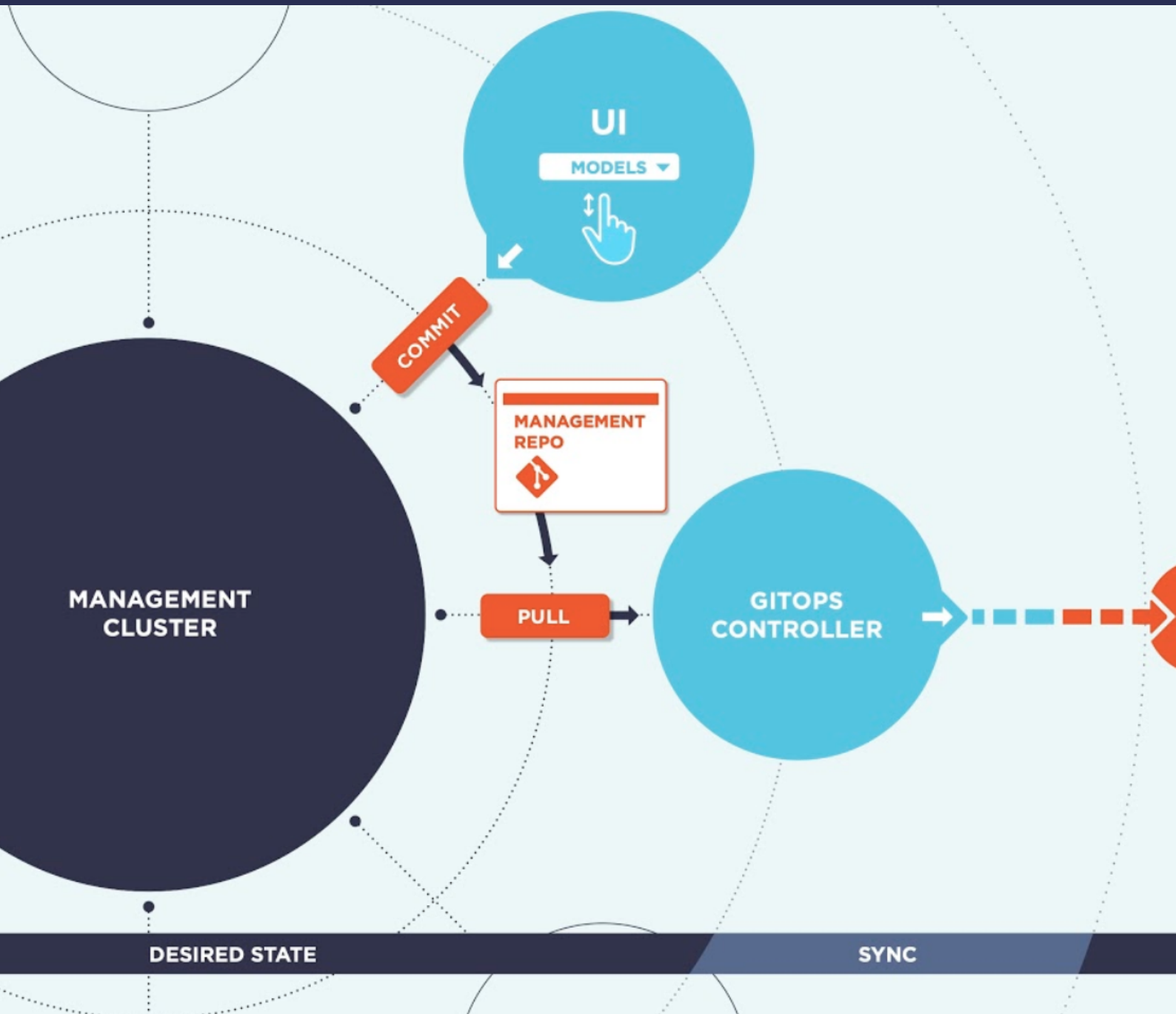


Solving Cloud Native App Delivery with the Weave Kubernetes Platform



Solving Cloud Native App Delivery with the Weave Kubernetes Platform

Congratulations on making the transition to cloud native and Kubernetes. You may now find yourself navigating through a vast array of options when it comes to piecing together the correct applications for a development and production ready Kubernetes platform. To complicate matters even further, there are specific use cases and business requirements you need to meet like machine learning and natural language processing. Configuring and managing these bespoke platforms, and ensuring that applications and add-ons work together correctly so they can be replicated confidently across environments without decreasing velocity is a challenge.

The key is a flexible and reproducible cloud native platform that can be adopted quickly and implemented with the correct technologies for your workloads yet operated wherever you want; whether that's on premise, across public clouds or in a hybrid-cloud configuration. Solving these kinds of problems takes more than a cluster installer. Not only do you need a framework for managing clusters throughout their entire lifecycle, but preferably, you need to do it with familiar tooling that engineering teams use on a daily basis.

WEAVERWORKS - THE GITOPS PIONEERS

By now you've heard about **GitOps**, what it is and how it can speed up development and deliver value for you and your customers without compromising on security and reliability. You may have first learned about it in the context of deploying and reconciling your application code to Kubernetes from a central source of truth kept in Git. But what you may or may not be aware of is that Kubernetes and its ecosystem of cloud native tooling are also declarative and can also be managed from Git. This means that GitOps can effectively manage all of your infrastructure as well as the complete cluster lifecycle, providing the exact same benefits that Git brings to cluster application deployments.

As the originators of GitOps, the fundamental workflows and underlying technologies of the Weave Kubernetes Platform (WKP) are based entirely on GitOps best practices. An automated Git-backed operations tooling approach reduces the complexity of configuration management for complete Kubernetes platforms. With the configuration kept in Git, WKP enables developers and operators to easily define, administer and install consistent, repeatable platforms with all of their add-ons and components with a simple 'git clone'.

4 Key benefits of GitOps in WKP

Git-based configuration management and monitoring brings the following to your organization:

- 1. Consistent environments** - Create, update and manage application clusters with all of the add-ons needed for an agile cloud native platform.
- 2. Reduce day 2 operations cost** - Minimize operations overhead with automated cluster lifecycle management: upgrades, security patches and cluster extension updates.
- 3. Active monitoring** - Validate applications and cluster changes and get alerts for inconsistencies with instant dashboards that require no configuration and are displayed straight from Git after launching a cluster.
- 4. Stronger security and compliance** - Increase security and meet regulatory requirements with auditable and transparent GitOps managed policies for real-time visibility and access control across your clusters and applications.

Common challenges for teams implementing Kubernetes

Delivering, managing and developing on Kubernetes presents challenges on many different levels before its benefits are realized. You need to ensure consistent cluster configuration across environments. Gaining optimal velocity requires that teams across your organization can work together and often on the same clusters. Below is a summary of the most common problems experienced on a day to day basis for those who work with Kubernetes in an enterprise.

Kubernetes persona	Challenge
<p>Platform Architect/Operator</p> <p>Responsible for creating a definition for clusters that will fill corporate and team requirements.</p>	<p>Defining standardized production ready clusters. Managing Kubernetes and cluster component upgrades.</p> <p>Needs to plan for security and compliance regulations.</p>
<p>Cluster Operator</p> <p>Responsible for managing clusters, taking the standardized platform definition and managing an individual cluster.</p>	<p>Cluster and cluster component upgrades. Scale up/scale down, managing customized configuration choices over time.</p> <p>Needs to implement security and compliance regulations.</p>
<p>Application Developer</p> <p>Responsible for creating an application that will run on the production platform. They will develop the application, test it in a variety of set-ups and release new versions.</p> <p>SRE/Application Developer</p> <p>Responsible for creating an application that will run on the production platform. They will develop the application, test it in a variety of set-ups and release new versions.</p>	<p>How to deploy new features and updates. How to set up identical environments in order to test, observe and debug issues.</p> <p>Needs to manage application uptime: observability, logging and all other related services.</p>

ENTERPRISE READY GITOPS, DEVELOPER APPROVED - WEAVE KUBERNETES PLATFORM

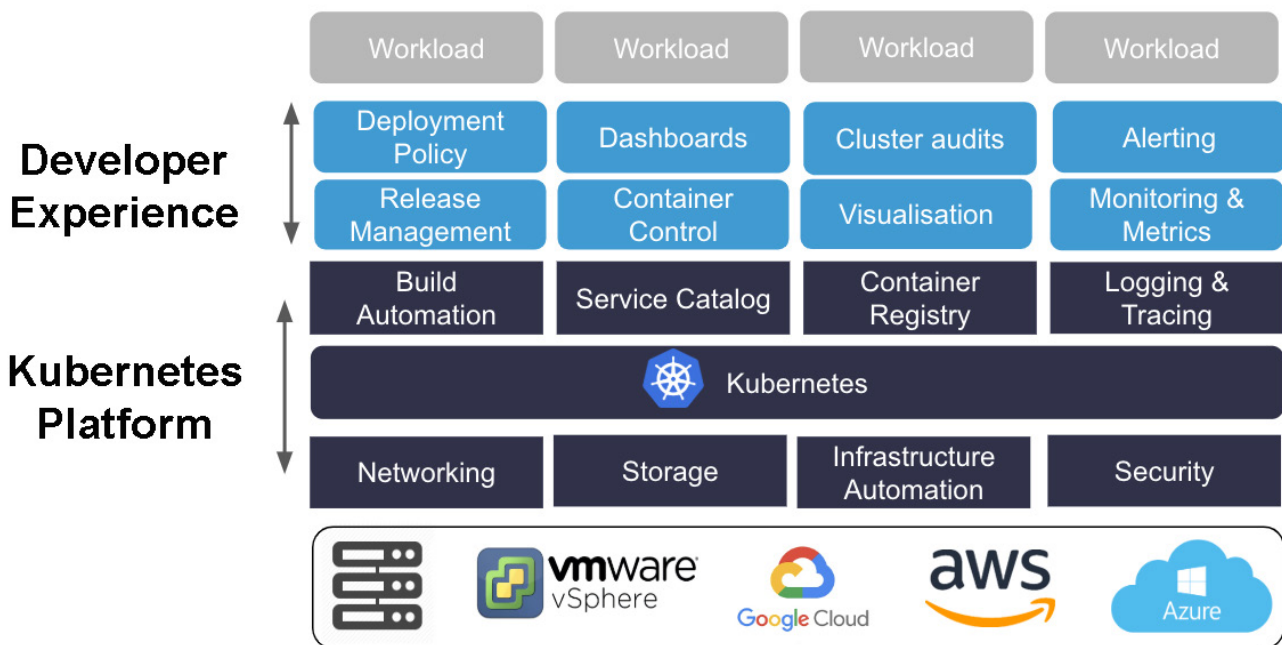
The Weave Kubernetes Platform is a production ready platform out of the box with absolutely no lock-in. GitOps is both the underlying architecture and developer experience of WKP which simplifies the configuration and management of Kubernetes platforms across your organization.

Operations teams build Kubernetes platforms by choosing from a catalogue of curated and up to date add-ons and other tools that solve common use cases such as monitoring, CI/CD deployment pipelines or specific cases such as machine learning. Once the tools have been selected and configured to work together, the configuration is checked into Git as a model definition that can be attached to a base version of pure upstream Kubernetes or installed to a variety of hosted Kubernetes versions such as EKS, AKS and GKE.

This provides developers with a self-service method to spin up correctly configured clusters based on their use case requirements, whenever and wherever they need to.

A stable, fully integrated stack managed from Git avoids one off clusters and other difficult to replicate cluster configurations and allows teams to deliver and manage reliable and predictable platforms across different environments whether on-premise, in the cloud or cluster fleets across multiple clouds.

Since declarative platform models are kept in Git, the entire cluster lifecycle including upgrades, security patches, rollbacks and other maintenance tasks are easily managed via pull requests. Most importantly, WKP provides policy and Git-based rules to specify, audit and control who can change what in the cluster configuration.



HOW THE WEAWE KUBERNETES PLATFORM WORKS

With Cluster creation in WKP is a three step process:

1. Automate cluster configuration with models

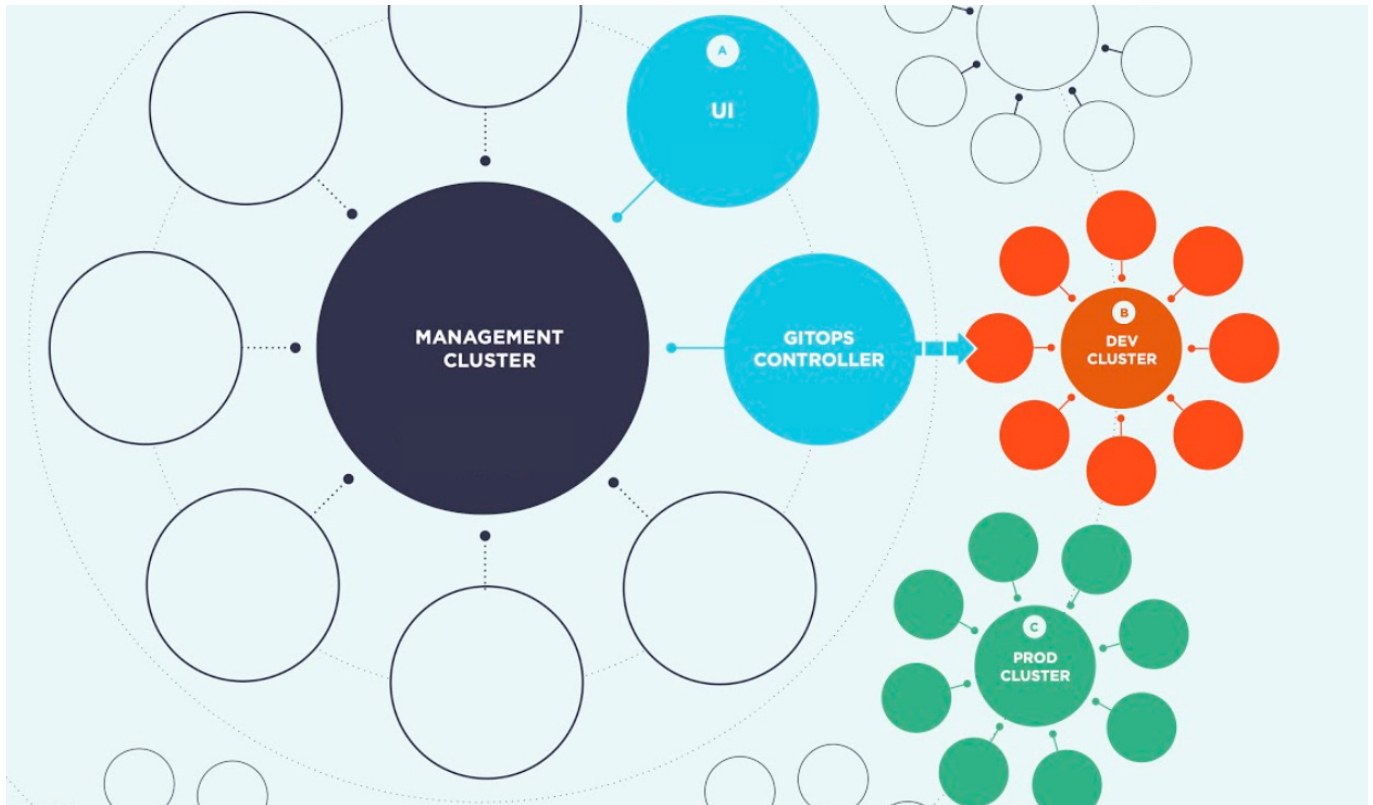
At the core of reproducible, centralized cluster configuration are WKP models. Build a model with a stack of add-ons for the type of clusters needed across your organization, for example you may have specific cluster stack configuration for machine learning or a specific backend for different cloud providers.

2. Control everything from Git

Store the cluster model in Git and make it available as a configured template for different teams across your organization. Attach a model and manage an installation of pure upstream Kubernetes with its add-ons, and control it all from Git. Easily manage the entire cluster lifecycle including security patches, upgrades and patches with GitOps.

3. Enable Developer self service

With policies and cluster configuration all in one place, developers and platform teams can simply `git fork` and `git clone` from the management clusters to create consistent and repeatable clusters at scale across environments, clouds and on premise.



Repeatable, consistent clusters anywhere

Step 1: Define a model

One of the basic ideas behind WKP is the ability to keep the configuration of a stack of integrated applications, add-ons and components in Git. Configuring add-ons and applications once provides a way for clusters to be easily reproduced across environments and clouds in a predictable way. WKP refers to these pre-configured stacks of applications and add-ons as a model. After a model is configured and checked into Git, teams can access it and attach it to a cluster during its creation for a fully configured out of the box Kubernetes platform.

The use of models avoids snowflake clusters everytime a specific app or configuration is required. Instead, teams can spin up clusters with well-defined patterns, saving time and avoiding the complexity of Kubernetes configuration management.

Custom models can be created by selecting from a carefully curated and up to date set of pre-configured apps, components and add-ons available from the WKP component and add-on repository that includes applications and add-ons such as: Prometheus, Grafana, Flagger, Helm, Ingress, NGINX and Flux.



Models 3 + Add Models

Name	K8s version	Node groups	Components & apps	Last change	
Production cluster Standardised production cluster for base application workloads.	1.14	10x m5.large 3x m5.xlarge	Prometheus, metrics server, fluentd, Elasticsearch, Istio, Jaeger, Helm, Tiller, wkp-external-dns, Vault	2019-11-12 16:41 dimitri@weave.works	Create a new cluster
Machine learning cluster Standardised production cluster for machine learning workloads.	1.14	5x m5.large 4x m5.8xlarge	Kubeflow, Knative, Prometheus, metrics server, fluentd, Elasticsearch, Istio, Jaeger, Helm, Tiller, wkp-external-dns	2019-11-15 12:29 filip@weave.works	Create a new cluster
Search components All components needed to work on the search functionality of the website.	1.14	5x m5.large	Elasticsearch-cluster-search, search-ui, search-api, redis	2019-11-14 12:36 simon@weave.works	Create a new cluster

Step 2: Create a cluster

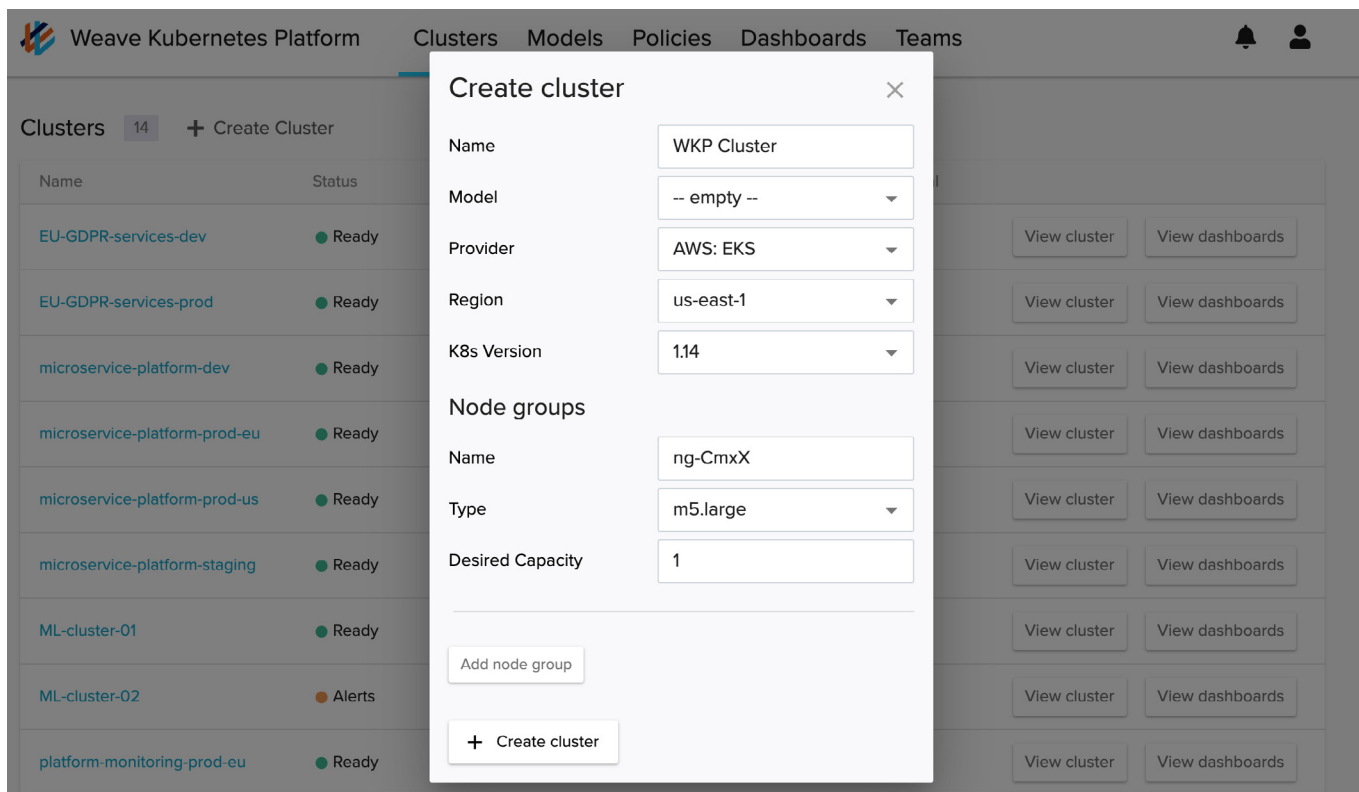
The Cluster tab provides a bird’s eye view onto all of the clusters or even fleets created with WKP throughout your organization. All versions of the configuration and declarative definitions of these clusters and their components and add-ons are stored in Git and can be audited or managed with GitOps.



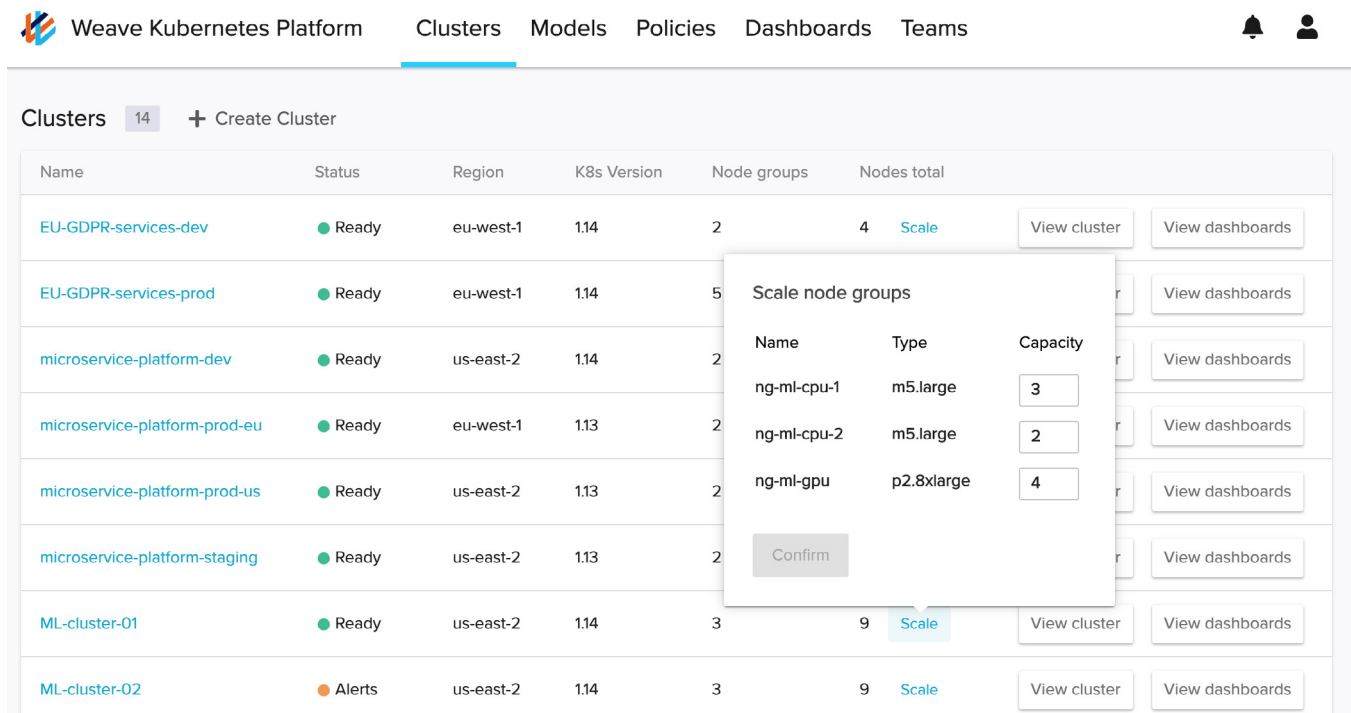
Clusters 14 + Create Cluster

Name	Status	Region	K8s Version	Node groups	Nodes total		
EU-GDPR-services-dev	● Ready	eu-west-1	1.14	2	4	Scale	View cluster View dashboards
EU-GDPR-services-prod	● Ready	eu-west-1	1.14	5	10	Scale	View cluster View dashboards
microservice-platform-dev	● Ready	us-east-2	1.14	2	4	Scale	View cluster View dashboards
microservice-platform-prod-eu	● Ready	eu-west-1	1.13	2	8	Scale	View cluster View dashboards
microservice-platform-prod-us	● Ready	us-east-2	1.13	2	8	Scale	View cluster View dashboards
microservice-platform-staging	● Ready	us-east-2	1.13	2	8	Scale	View cluster View dashboards
ML-cluster-01	● Ready	us-east-2	1.14	3	9	Scale	View cluster View dashboards
ML-cluster-02	● Alerts	us-east-2	1.14	3	9	Scale	View cluster View dashboards
platform-monitoring-prod-eu	● Ready	eu-west-1	1.13	1	15	Scale	View cluster View dashboards

You can also create a cluster from this view and attach the model you'd like to add to create a specialized platform. Note that you can nest models to create different backends for a multicloud configuration. Next choose where you want to launch the cluster, public cloud, and on-premise and finally select the version of Kubernetes you want to launch on.



With a configured cluster, it's simple to scale straight from the WKP management console, and then version control the change in Git, also providing a full audit trail of who did what.



Step 3: Manage and monitor the cluster

With the cluster created and launched, clicking on its definition takes you to the cluster's instant dashboard. The dashboard displays the cluster's added components with a snapshot of its health. The dashboard also provides an easy way to drill down onto an individual configuration to view the history of what's been changed to the cluster in Git.

With Git-backed control and management, clusters are fully compliant with auditable logs of everything that's ever been changed or added. Developers also don't have to learn a new workflow to create new clusters across environments. A single, standard workflow for both application development and cluster operations can also be applied, simplifying Kubernetes management.

The screenshot displays the Weave Kubernetes Platform dashboard. On the left, under 'Cluster Components' (12), there is a list of components with icons and 'Open' buttons: wkp-alb-ingress-controller, wkp-external-dns, wkp-flux-bootstrap, wkp-flux-helm-operator, wkp-github-service, wkp-gitops-repo-broker, wkp-grafana (with 'Open Grafana' button), wkp-prometheus (with 'Open Alertmanager' and 'Open Prometheus' buttons), wkp-scope (with 'Open Scope' button), wkp-tiller, wkp-ui (with 'Open Ui' button), and wkp-workspaces. On the right, under 'Workspaces' (3), there is a list of namespaces with 'Github repo' buttons: namespace team-alpha-ns, namespace team-alpha-ns, and namespace team-charlie-ns. Below that, under 'Active Git branches' (2), there is a list of branches with commit hashes and update information: master (Default Branch) updated a day ago by Weaveworks Kubernetes Platform (hash 57bf4137) and bump-wkp-ui updated 2 months ago by GitHub (hash 03de6b1e).

UNLOCK CLOUD NATIVE AGILITY WITH GITOPS AND WKP

Reap the rewards of increased velocity and reduced operational complexity backed by our experts. Accelerate cluster delivery and security while introducing proven developer best practices and policy management with GitOps workflows. Benefit from repeatable, flexible Kubernetes cluster management across all infrastructure - in the cloud and on-premise.

Weaveworks makes it easy to accelerate Kubernetes adoption.

[Register for a demo](#) or [contact sales](#) for more information.