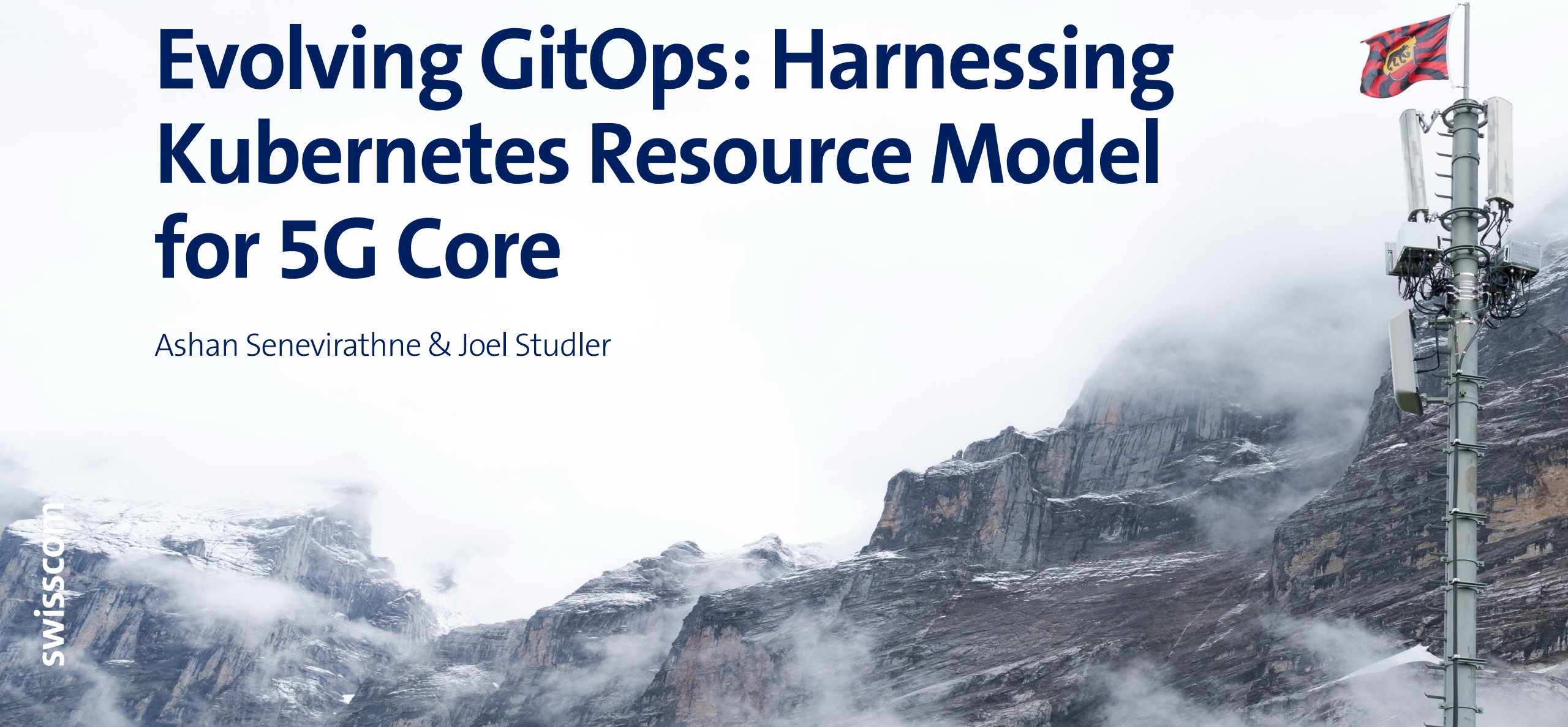
Evolving GitOps: Harnessing Kubernetes Resource Model for 5G Core

Ashan Senevirathne & Joel Studler

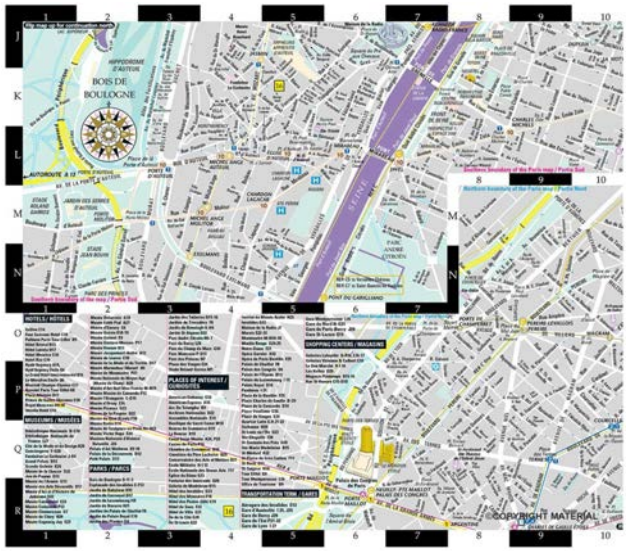


An Analogy...

Static paper map

- Fixed
- Static
- Unchanging
- Overwhelming

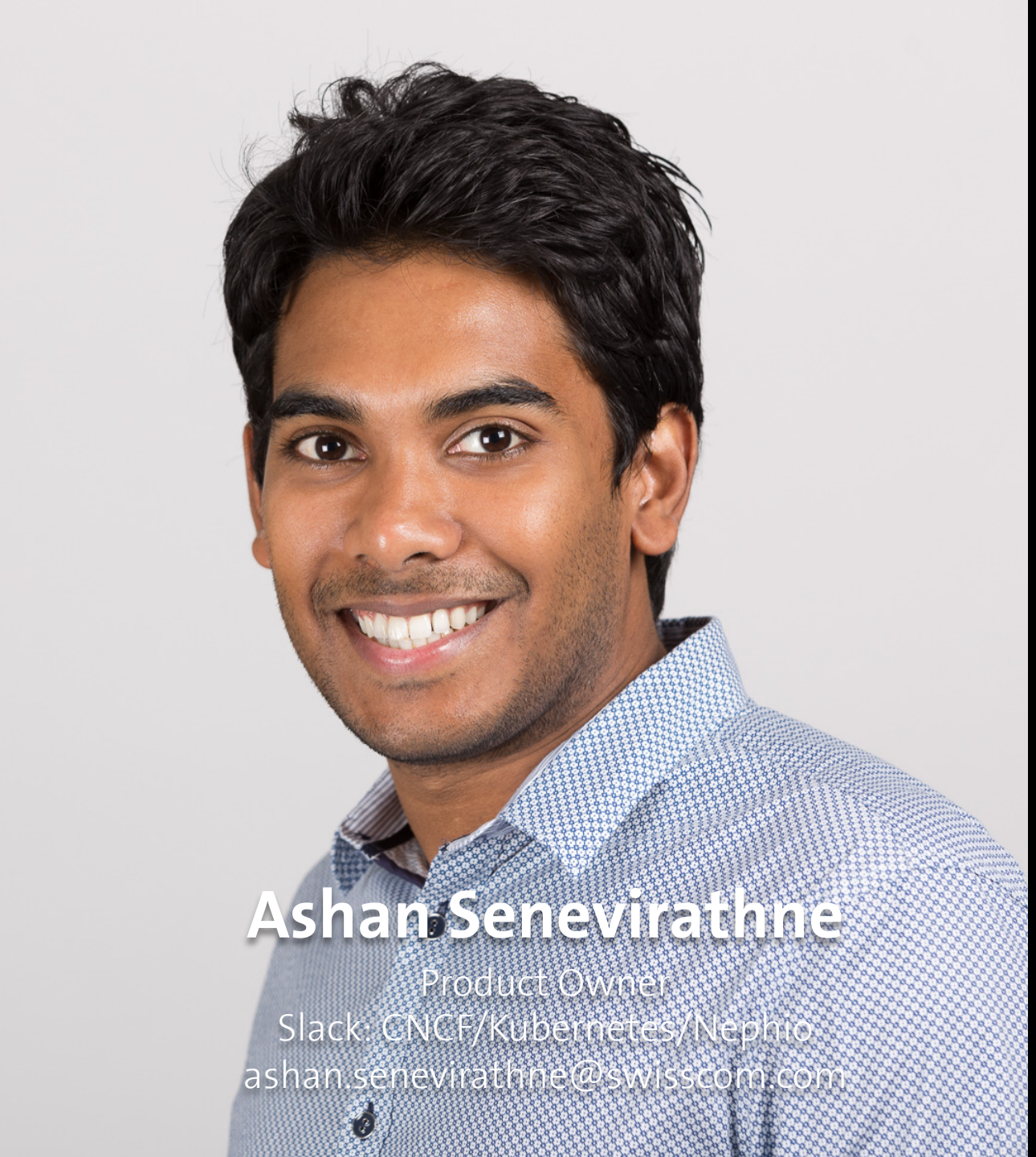
This is GitOps today



Apple maps

- Dynamic
- Changes based on external conditions
- More focused
- Simple to navigate

This is GitOps w/ KRM

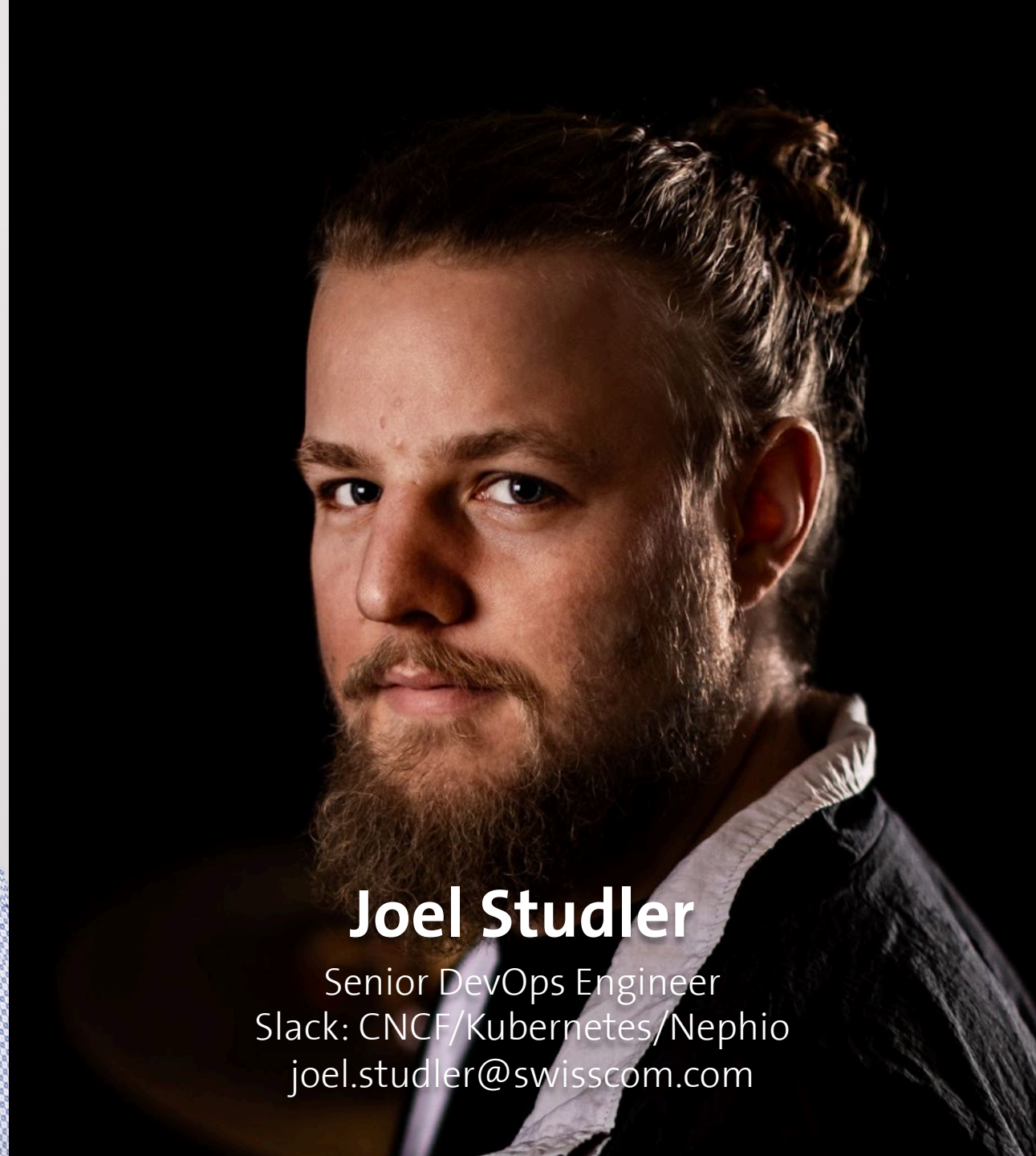


Ashan Senevirathne

Product Owner

Slack: CNCF/Kubernetes/Nephio

ashan.senevirathne@swisscom.com



Joel Studler

Senior DevOps Engineer

Slack: CNCF/Kubernetes/Nephio

joel.studler@swisscom.com


swisscom

From Telco to TechCo: What Does It Mean for Us?

5G - driving our journey from Telco to TechCo
by Swisscom CTIO Mark Düsener at Connect Conference 2022
https://www.youtube.com/watch?v=hND7TiXJED8

Cloud Nativeness

we need to introduce Reconciliation and leverage Kubernetes Patterns

Simplicity

we need to introduce abstraction in our configuration and break it into smaller manageable parts

Automation

we need to move from an imperative workflow-driven model to a declarative intent-based model

What Is a 5G Core?

Each blue object

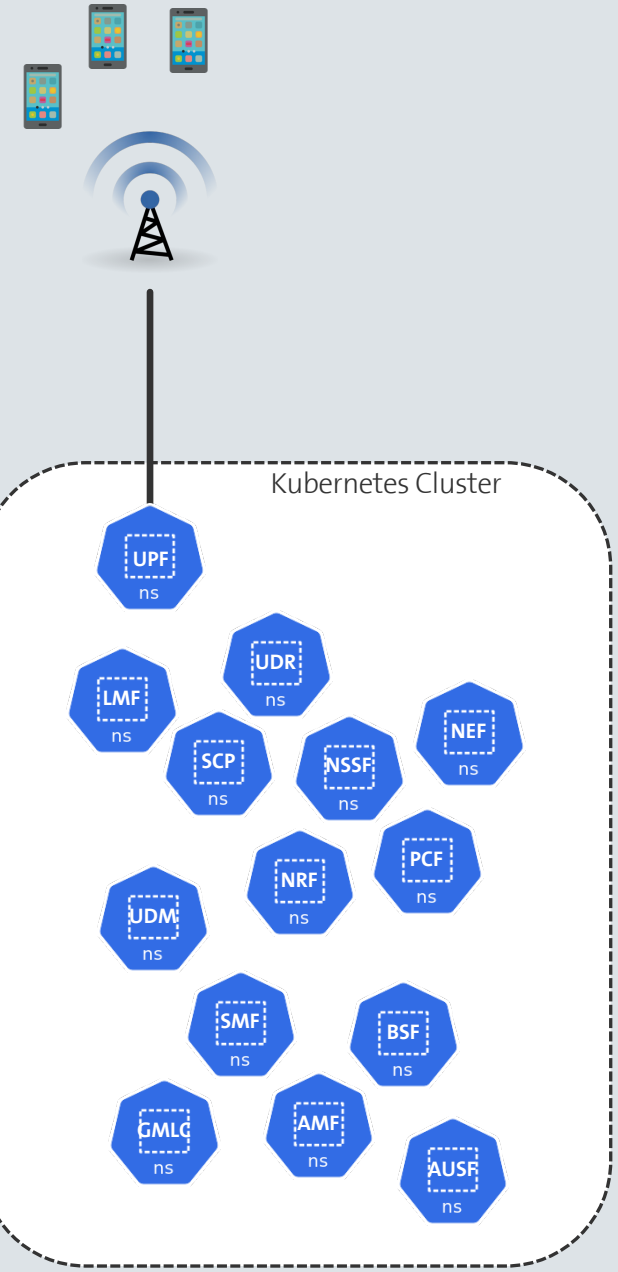
- is an App, in Mobile Networks this is called «CNF» aka «Containerized Network Function» – e.g. Router (UPF), Authentication Service (AUSF)
- Deployed using Helm

Configuration is done via

- Helm Values
- Other interfaces such as NETCONF

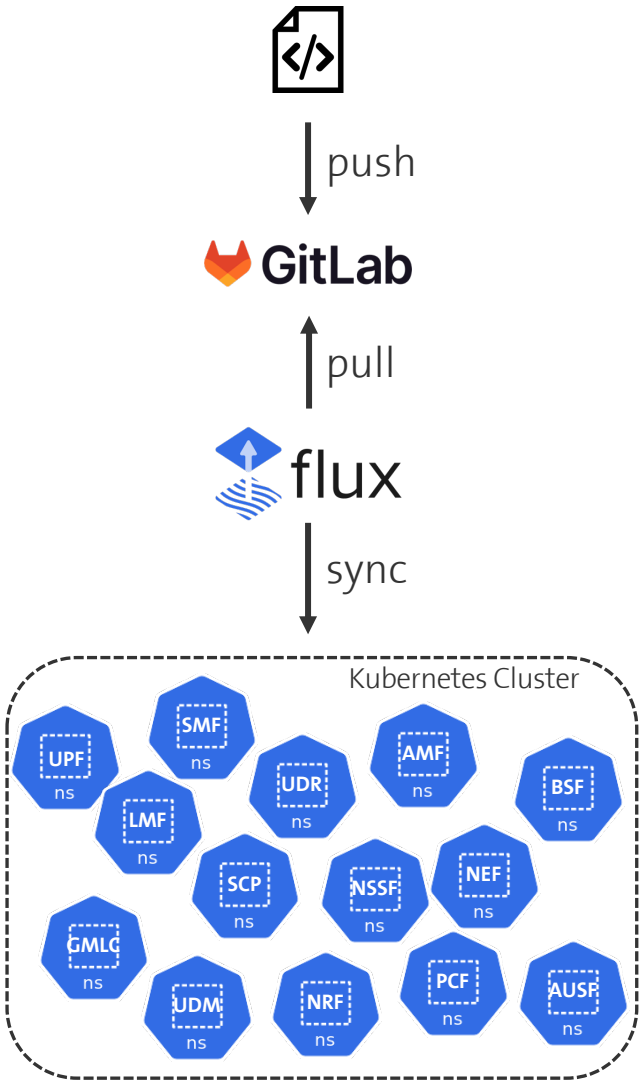
Scale

- A development environment contains ~2000 pods
- A total of 5000 interdependent configuration parameters



Current State of 5G Core Configuration using GitOps

- Git as source of truth
- Declarative configuration
- Continuous reconciliation
- Versioned and immutable



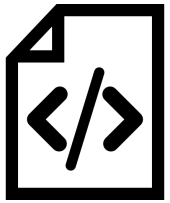
How We Are Moving from GitOps to Kubernetes Resource Model in 5G Core

By Ashan Senevirathne & Joel Studler

KubeCon Europe 2024

https://sched.co/1YeN2

Example <config>



Example <config>

IP addresses

Subnets

VLANs

DNS Records

Network function variables

Infrastructure variables

Network function-Network function mapping

Secret references

Certificate references

LACK OF ABSTRACTION

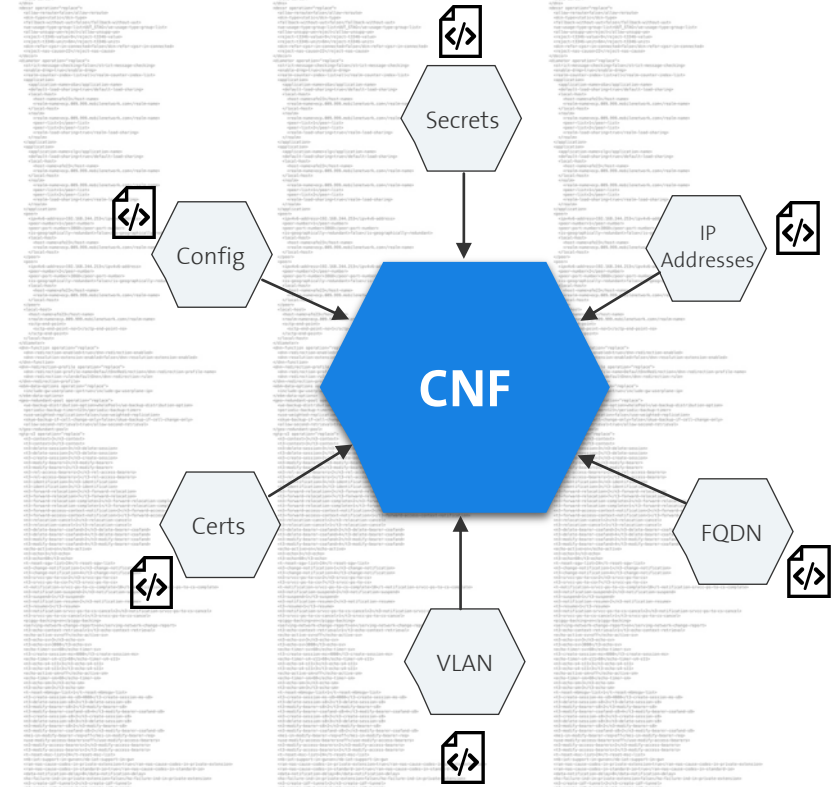
```
      <application-name>s6a</application-name>
      <default-load-sharing>true</default-load-sharing>
      <local-host>
        <host-name>afe23</host-name>
        <realm-name>ecp.009.999.mobilenetwork.com</realm-name>
      </local-host>
      <realm>
        <realm-name>ecp.009.999.mobilenetwork.com</realm-name>
        <peer-list>1</peer-list>
        <peer-list>2</peer-list>
        <realm-load-sharing>true</realm-load-sharing>
      </realm>
    </application>
    <application>
      <application-name>slg</application-name>
      <default-load-sharing>true</default-load-sharing>
      <local-host>
        <host-name>afe23</host-name>
        <realm-name>ecp.009.999.mobilenetwork.com</realm-name>
      </local-host>
      <realm>
        <realm-name>ecp.009.999.mobilenetwork.com</realm-name>
        <peer-list>1</peer-list>
        <peer-list>2</peer-list>
        <realm-load-sharing>true</realm-load-sharing>
      </realm>
    </application>
    <peer>
      <ipv4v6-address>192.168.244.253</ipv4v6-address>
      <peer-number>1</peer-number>
      <peer-port-number>3868</peer-port-number>
      <is-geographically-redundant>false</is-geographically-redundant>
      <local-host>
        <host-name>afe23</host-name>
        <realm-name>ecp.009.999.mobilenetwork.com</realm-name>
      </local-host>
    </peer>
    <peer>
      <ipv4v6-address>192.168.244.213</ipv4v6-address>
      <peer-number>2</peer-number>
      <peer-port-number>3868</peer-port-number>
      <is-geographically-redundant>false</is-geographically-redundant>
      <local-host>
        <host-name>afe23</host-name>
        <realm-name>ecp.009.999.mobilenetwork.com</realm-name>
      </local-host>
    </peer>
    <local-host>
      <host-name>afe23</host-name>
      <realm-name>ecp.009.999.mobilenetwork.com</realm-name>
      <sctp-end-point>
        <sctp-end-point-no>1</sctp-end-point-no>
      </sctp-end-point>
    </local-host>
  </diameter>
  <dnn-function operation="replace">
    <dnn-redirection-enabled>true</dnn-redirection-enabled>
    <dnn-resolution-extension-enabled>false</dnn-resolution-extension-enabled>
  </dnn-function>
  <dnn-redirection-profile operation="replace">
    <dnn-redirection-profile-name>DefaultDnnRedirection</dnn-redirection-profile-name>
    <dnn-redirection-rule>defaultDnn</dnn-redirection-rule>
</dnn-redirection-profile>
```

Configuration Philosophy

Several **unsolved problems** introduce toil into a **GitOps based configuration management system**.

- **Complexity in configurations**
  5G core configurations typically involve thousands of parameters and are thus not user friendly
- **Redundancies across configurations**
  Repeated parameters across application config and helm charts
- **Limited configuration discovery**
  Automatic configuration based on the environment still weak
- **"Need to know" all details up-front**
  IP addresses, VLANs
- **Outside –> In**
  Instructions are pushed to the receiving system

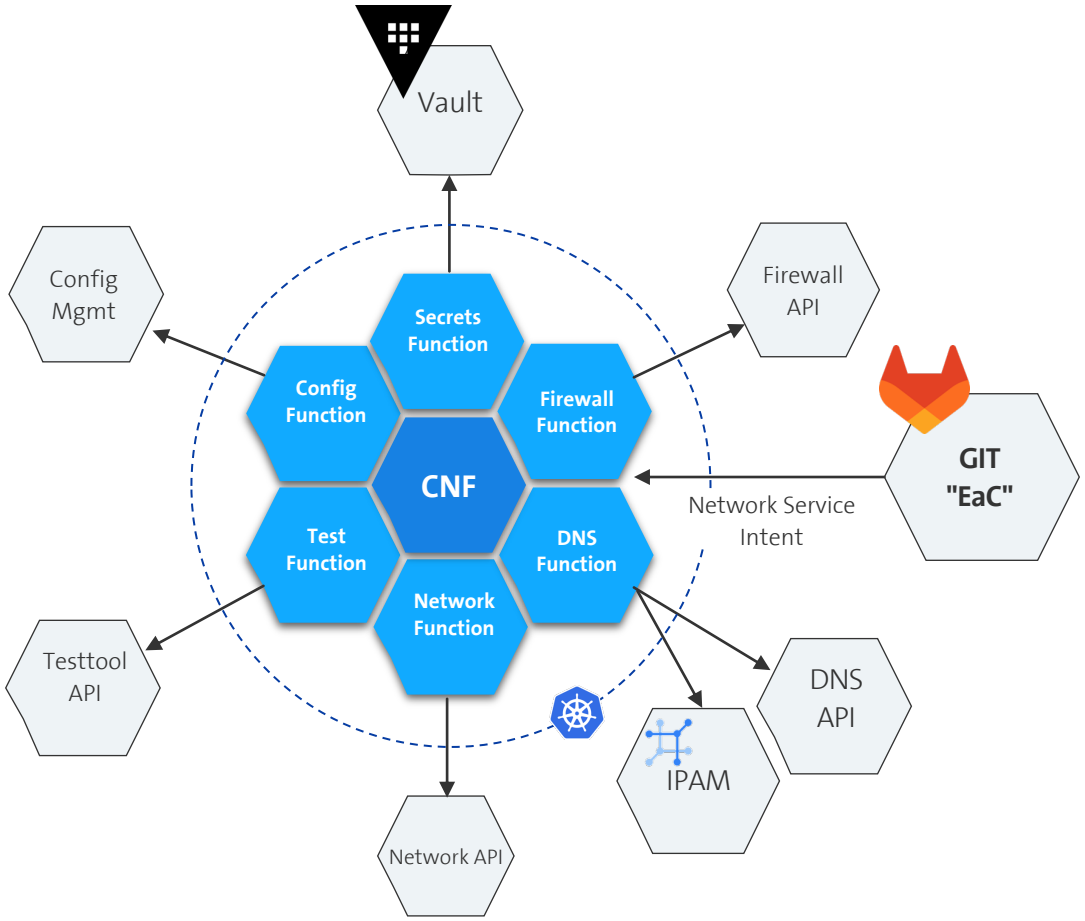
https://sre.google/workbook/configuration-design



What If?

*Instead of telling the system **how to do everything**, we tell system **what we want**, and **system figures out the best possible way** to make it happen.*

- CNF receives the **intent** to move into an operational state **without step-by-step instructions**
- **Dynamic config assembly** based on **required resources** during runtime
- **Inside → out**
  System requests relevant information automatically from external sources
- **Extend Kubernetes API** to bring Kubernetes-native capabilities



Kubernetes Resource Model (KRM)

API extensions

Custom Resource Definitions extend the Kubernetes API.

e.g. API definition for NetBox PrefixClaim

CRs as Instances

Custom Resources instantiate a CRD.

e.g. NetBox PrefixClaim resource

Business Logic

Use of Operators or templates to run custom logic

e.g. Assemble a config, Self healing

What Are Our Requirements?

KRM Based

Input and Output can be KRM resources

Garbage Collection

Resources belonging together are lifecycled together

Output Flexibility

Generated resources stored with various backends

Reconciliation

Output is updated as Input is changed

Functions

Ability to define helper functions (e.g. get status field from K8s CR)

Gradual Adoption

Migration from static to dynamic taken at own pace.

But What About Helm, Kustomize, Argo and Flux?

Limited Dynamic Assembly

We cannot use live Kubernetes resources as source of information

No Custom Functions

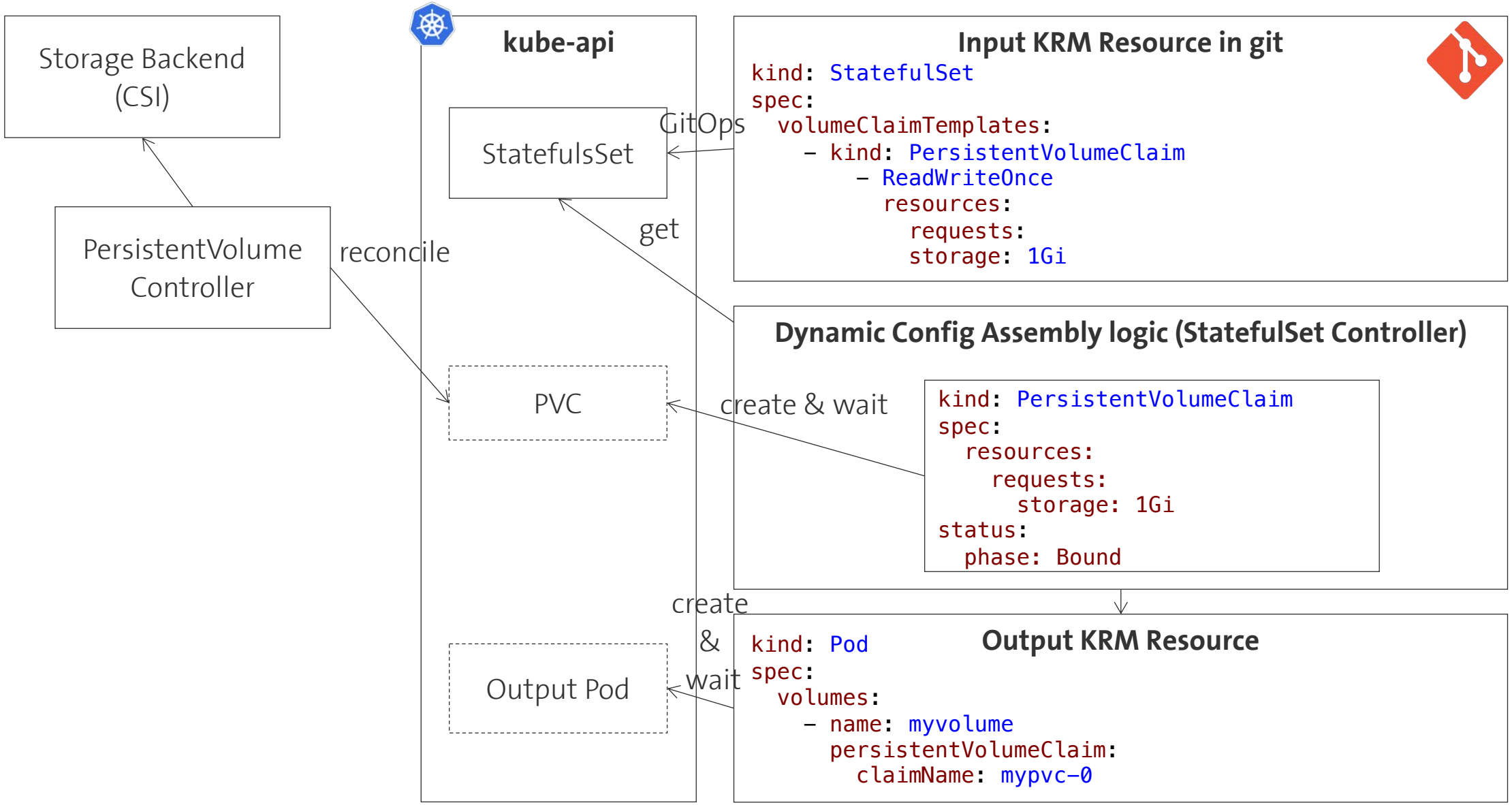
We cannot invoke arbitrary code into Kustomize/Helm templating

Only Partially KRM

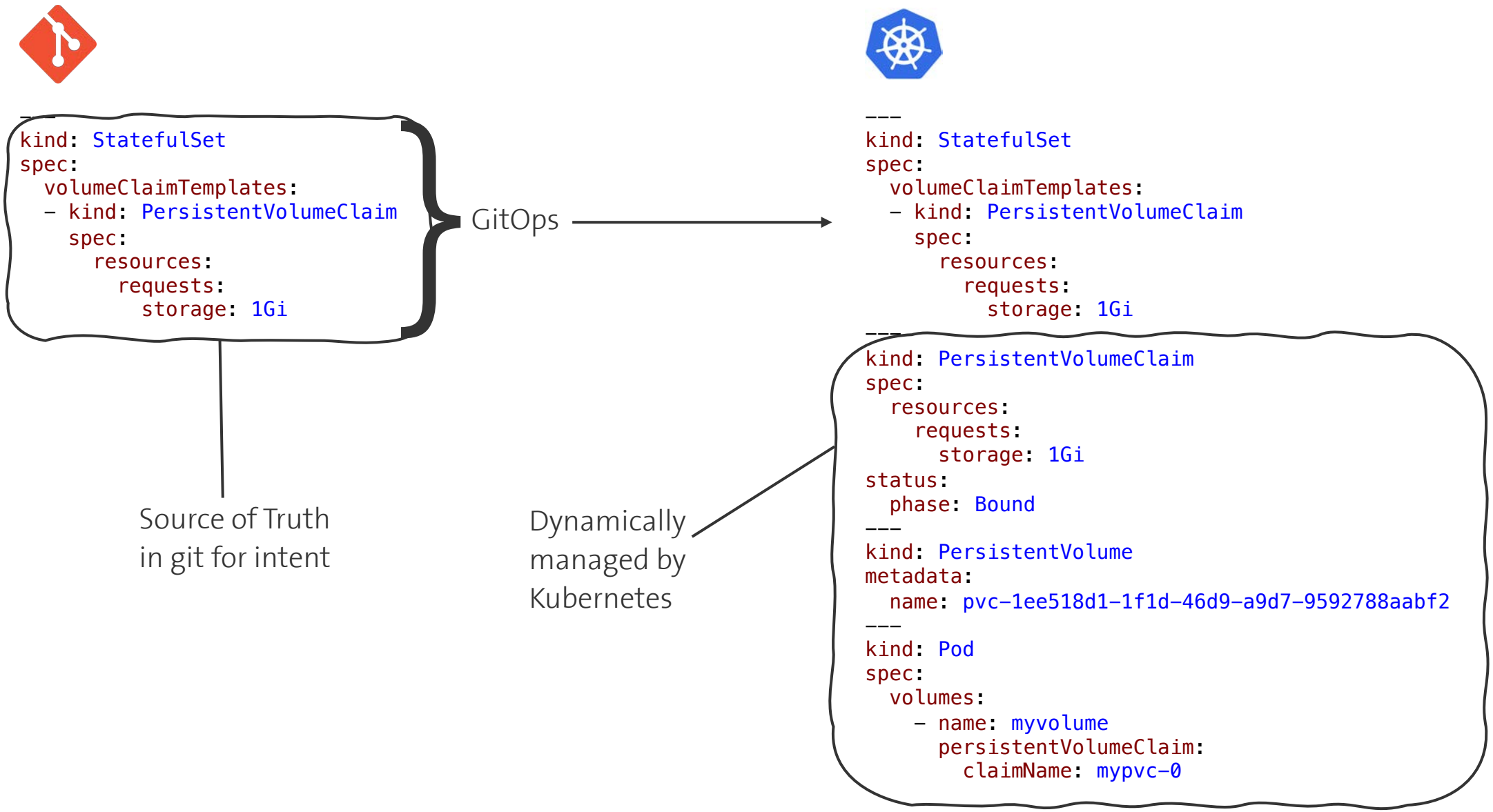
Not all Resources involved follow KRM


GitOps 😍 KRM
swisscom

Config Abstraction Example in Kubernetes: PVC for StatefulSet



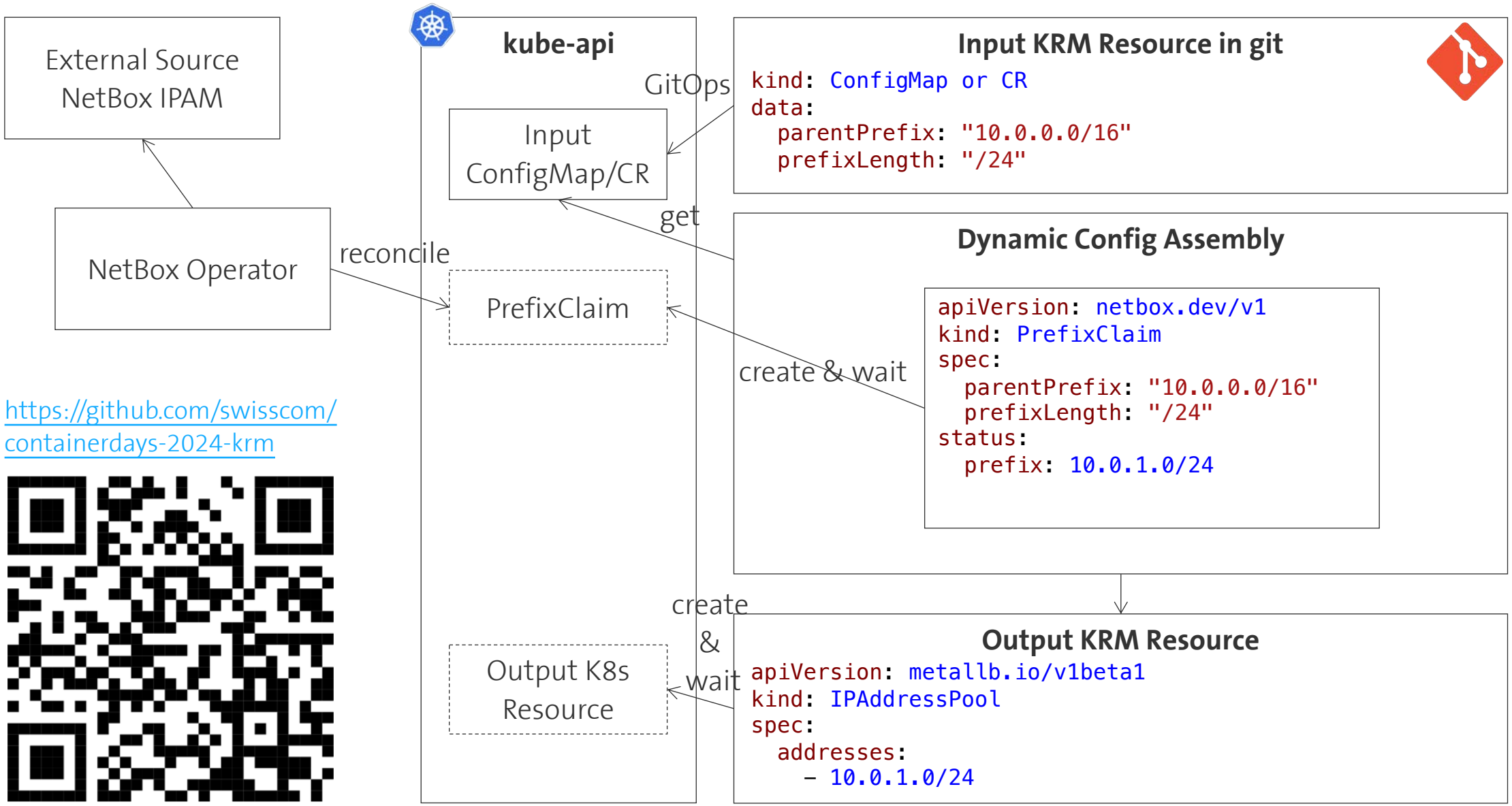
Shared Source of Truth in Git and Kubernetes



GitOps vs. GitOps with KRM

	GitOps	GitOps with KRM
Source of Truth	Git	Git (intent) and Kubernetes (dynamic state)
Config Management / Abstraction	All details are in Git, no abstraction	Git holds intent; Kubernetes manages details
Flexibility	Less flexible, changes require Git updates	More flexible, dynamic assembly in Kubernetes
Scalability	More cumbersome and complex to scale	Easier to scale
Operational Complexity	High; all details tracked in Git	Medium; Kubernetes automates dynamic management

Config Abstraction Demo Example: Dynamically Assign IPs for MetalLB



https://github.com/swisscom/containerdays-2024-krm

Ansible + Jinja2

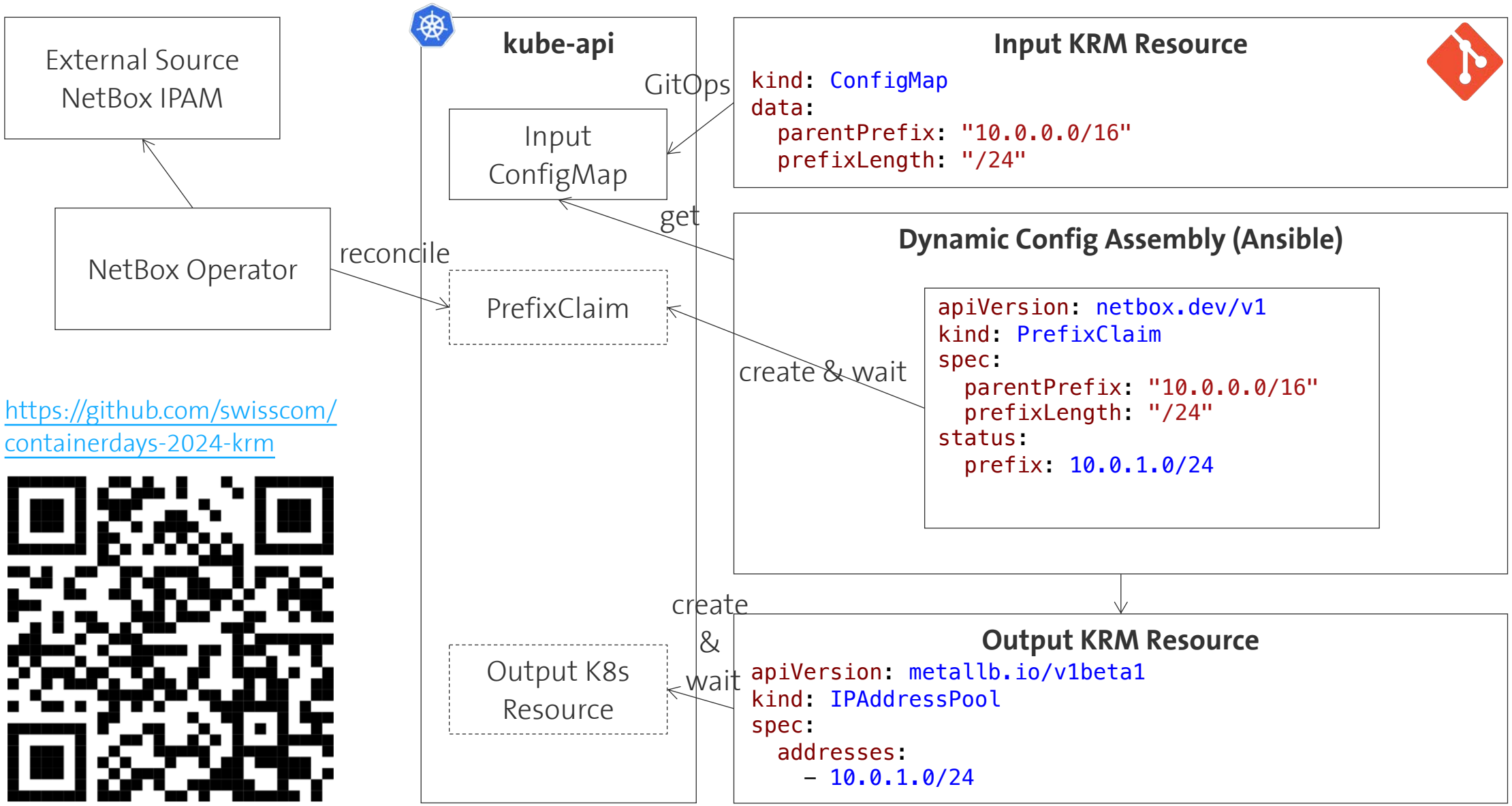
Open-Source automation framework
https://github.com/ansible/ansible (62k Stars, 23k Forks)

Business Logic in Templates and Playbooks
Jinja2 for templating, Playbooks for imperative logic

State Management
Stateless, can use Kubernetes Garbage Collection

No Reconciliation
Just a binary, triggering/execution not solved

Config Abstraction With Ansible + Jinja2



https://github.com/swisscom/containerdays-2024-krm

Kubernetes Operator

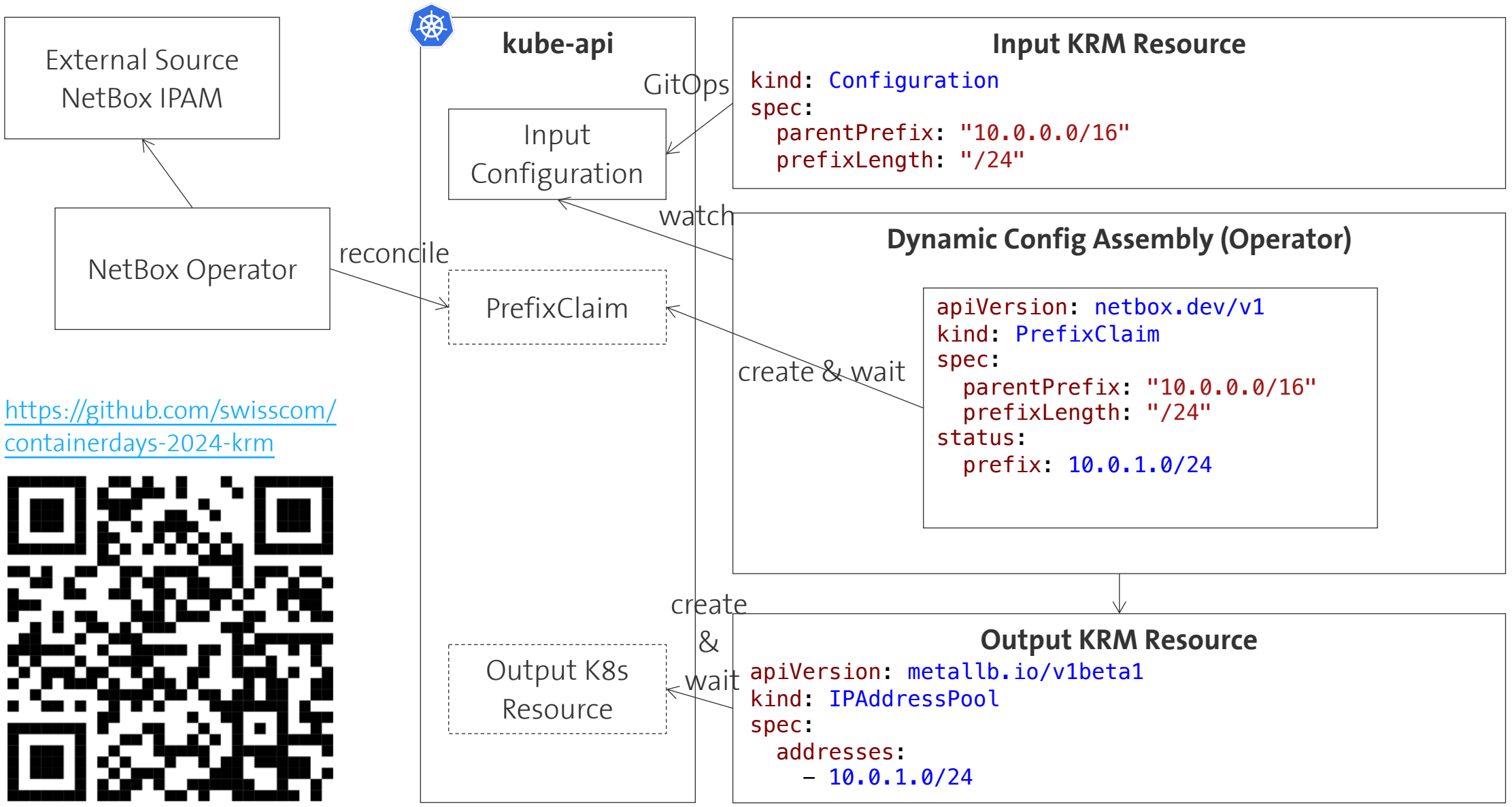
Golang Based Operator scaffolded using Kubebuilder
https://github.com/kubernetes-sigs/kubebuilder

Business Logic in Go Code
Maximum flexibility for imperative logic

State Management
Uses Kubernetes Garbage Collection and Finalizers

Reconciliation
Event driven triggering, constant reconciliation

Config Abstraction With Kubernetes Operator



https://github.com/swisscom/containerdays-2024-krm

Ansible vs. Kubernetes Operator

	Ansible+Jinja2	Kubernetes Operator
KRM Based	●●○○○	●●●○○
Garbage Collection	●●●●○	●●●●○
Output Flexibility	●●●●●	●●●●●
Reconciliation	●○○○○	●●●●●
Functions	●●●●○	●●●●●
Gradual Adoption	●●●●●	●○○○○

No mature Kubernetes native tool available to dynamically assemble configurations

Our Learnings & Suggestions

... on configuration abstraction

Avoid

Checking in low level configurations in Git
Things like IP Addresses, VLAN IDs

Complicated configurations
Avoid unnecessary layers of abstraction

Ignore tooling gaps
Avoid using tools like Ansible or Jinja that aren't designed with Kubernetes in mind

Neglect Industry Relevance
Don't assume KRM challenges and solutions are unique to your sector

Aim to

Leverage abstraction
Simplify complex configurations by focusing on essential controls

Reuse cloud native tools to be in-band with K8s
E.g. Flux, Argo, Testkube, cert-manager

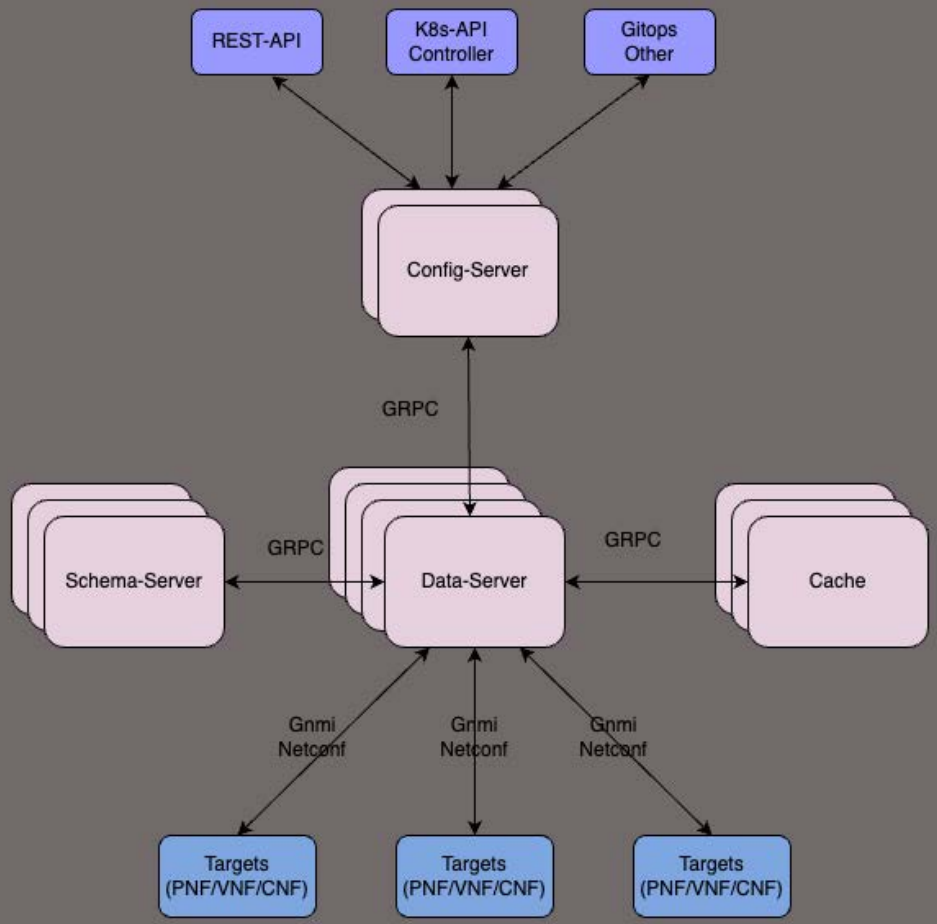
Prioritize scalability
Design KRM-based tools with future growth and scaling

Contribute to the ecosystem
Share your code


Community
Contributions
We build on community-driven tools and
proudly give back, driving innovation together!
swisscom

Config Sync Operator SDC

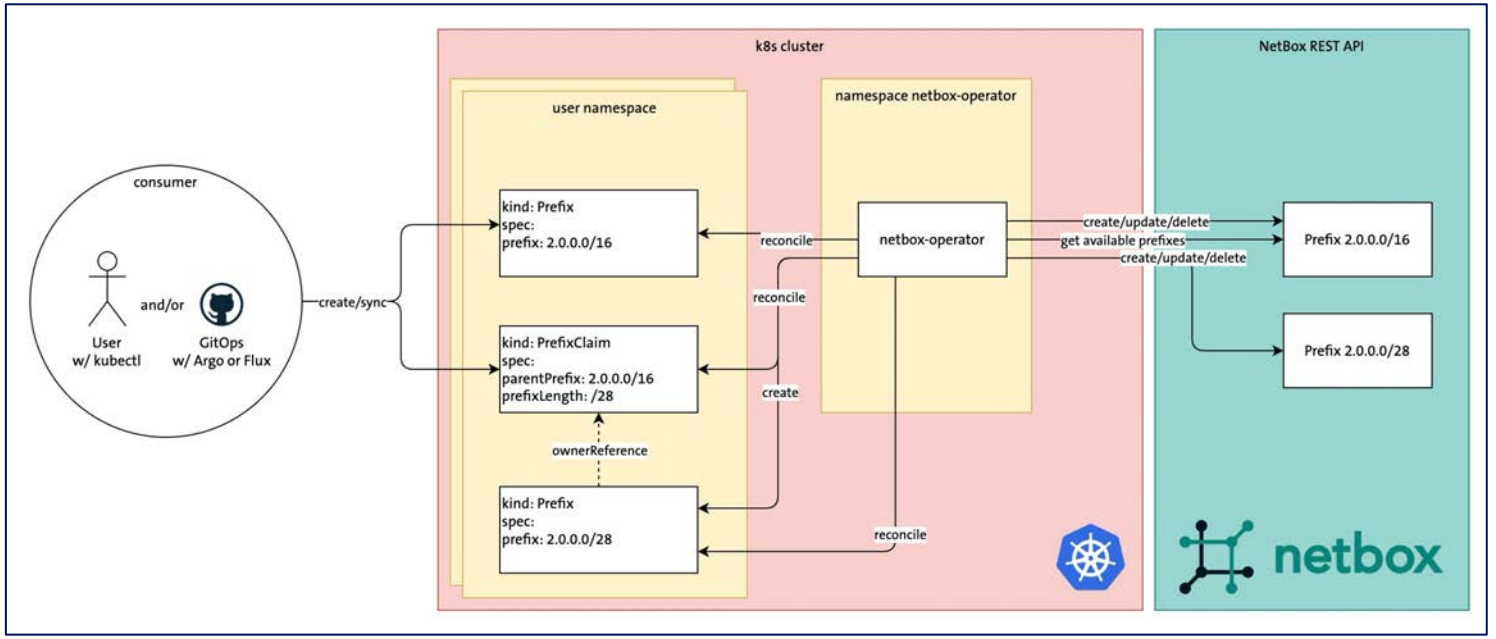
- “Schema Driven Configuration” (SDC)
- Use case: Config apply tool for 5G Core
- Part of KubeNet community https://discord.gg/64WpUaxt
- Protocols: gNMI, NETCONF (Telco Industry Standard)
- Supported Targets: Physical (PNF), Virtual Machines (VNF) Cloud Native (CNF)
- Vendor agnostic
- Declarative (Kubernetes CRs, Operators, API extension)



https://docs.sdcio.dev / https://github.com/sdcio

NetBox Operator

NetBox Operator, a tool designed to integrate NetBox resource management directly into your Kubernetes environment.



https://github.com/netbox-community/netbox-operator
CNCF Blog Post

Related Talks

How We Are Moving from GitOps to Kubernetes Resource Model in 5G Core
by Ashan Senevirathne and Joel Studler at KubeCon Europe 2024
https://www.youtube.com/watch?v=crmTnB6Zwt8

From Spreadsheets to "Everything as Code"
by Joel Studler and Jéremy Weber at ContainerDays 2023
https://www.youtube.com/watch?v=eMAMqW2aLTI

Thanks!

Q&A