Automation for the deployment of 5G Core and Open RAN

Shujaur Mufti
Senior Manager, Global Partners Solution Architecture
TME
smufti@redhat.com

Nov 9, 2022
“Anything not automated is slowing you down”
Mobile Network Deployment Scale varies

- **Cell Sites**: Thousands of sites
- **Edge and Far Edge DCs**: Dozens of sites
- **Regional and Central DCs**: Handful Sites

Applications and Cluster size:
- Small clusters
- Compact cluster(s)
- Large Cluster(s) for many/heavier application
Cloud Native Realization is benefiting

Top Considerations: Speed, Security, and Scale
Automation and GitOps
Traditional Provisioning
✓ slow
✓ error prone
✓ hard to scale

Infrastructure as Code (IaC)
✓ Manage infrastructure with declarative model
✓ Follows DevOps practices
✓ ZTP for infrastructure

Automation
✓ repeatable outcome for specialized task
✓ no human intervention

Orchestration
✓ organize and coordinate the execution of multiple automation tasks to define a workflow

Self-Organizing Networks (SON)
✓ Automation for configuration, optimization and healing of RAN
✓ ZTP for RU

GitOps
✓ continuous reconciliation (self-healing)
✓ declarative operations
✓ end-to-end zero-touch provisioning (ZTP)
Existing practices and GitOps
What is GitOps?

An developer-centric approach to Continuous Delivery and infrastructure operation

- Treat everything as code
- Git is the single source of truth
- Auto-reconcile ops through Git workflows
Why GitOps?

**Standard Workflow**
Familiar tools and Git workflows from application development teams

**Enhanced Security**
Review changes beforehand, detect configuration drifts, and take action

**Visibility and Audit**
Capturing and tracing any change to clusters through Git history

**Multi-cluster consistency**
Reliably and consistently configure multiple Kubernetes clusters and deployment
GitOps Principles

The system is described declaratively
The desired state is versioned in Git
Approved changes can be applied automatically
A controller exists to detect and act on drift
GitOps Controllers

1. Git Repo
2. GitOps Controllers
3. Reconciliation Loop
4. Events, Logs & Metrics
5.

Red Hat
GitOps Pipelines

Declarative State
(Blueprints & Manifests)

Versioned Declarations
(e.g. Git Repo)

Continuous Deployment
(Pipelines)

Closed-Loop Automation
(GitOps Controllers)
GitOps Workflow

A declarative approach to application delivery
Example: The GitOps Application Delivery Model

Source Git Repository → CI → Image Registry
Example: The GitOps Application Delivery Model
Example: The GitOps Application Delivery Model

1. Source Git Repository
2. CI
3. Image Registry
4. Pull Request
5. CD
6. Kubernetes
7. Monitor
8. Deploy
9. Detect drift
10. Take action

Example: The GitOps Application Delivery Model

Source Git Repository → CI → Image Registry → Pull Request → CD → Kubernetes → Monitor → Deploy → Detect drift → Take action
GitOps Application Delivery Model with RedHat OpenShift
GitOps for Core and Open RAN
5GC and Open RAN Deployment Scenarios

- Public Macro sites & Small Cells
- In building solutions
- Private Networks

CU - CU-CP/RIC/5GC/APP
DU - UP/RIC/UPF/APP
RU - CU-UP
Baseline Requirements

Scale
- Targeting environments where scale could potentially be:
  - 100's of regions, each managed and supported individually
  - 10,000's of sites per region
  - 1-100s of nodes per site

Cookie Cut Approach
- Design Once and Deploy many

Single Source of Truth
- Deployment Templates (Cookie Stamps)
- Deployment Metadata details (Ingredients)

Controlled Versioning
- Has been deployed
- To be deployed
- To be updated
GitOps Model for 5GC/vRAN Deployment Automation

High Level Provisioning Flow

1. 3rd-party Site Modeling Tools
2. Manifests in GIT
3. Deploy with Zero Touch
Operational Flow with GitOps

- K8s Multi Cluster Manager
- Helm
- Kustomize
- K8s Manifests
- Orchestration

Site Designs & Inventory

Serverless w/ CI/CD Pipelines
(Transform design to GitOps structures)
Example of GitOps repo structure

Git repo with the base or common configuration

Git Repo for 5GC & RAN

Git repo with clusters definitions using blueprints and base configurations

Git repo with blueprints using base definitions
ZTP with GitOps

- **GIT REPO**
  - Define Site at GitOps repo
- **Orchestrator**
  - Webhook notification
  - Follow Progress
- **GitOps Controllers**
  - Register to Message Bus
  - Configure Infrastructure
  - OCP RAN Configuration
  - Deploy CNF
- **Message Bus**
  - Logs, Metrics, Events & Alarms
  - Deploy Cluster
Example: Upgrading non-RAN Cluster

- **Set Cluster Version in GitOps repo**
- **Webhook notification**
- **Follow Progress**
- **Register to Message Bus**
- **If in maintenance window execute upgrade**
- **Rolling Upgrade**
- **Message Bus**
  
  - Logs, Metrics, Events & Alarms
Example: Upgrading O-RAN Cluster (general view)

- **GIT REPO**
  - Set non-master MCPs to Pause in Git Repo
  - Set Cluster Version in GitOps repo
- **Orchestrator**
  - Webhook notification
  - Follow Progress
- **GitOps Controllers**
  - Register to Message Bus
  - If in maintenance window execute upgrade of control-plane
- **Message Bus**
- **Control-Plane**
  - If in maintenance window execute upgrade of MCP
  - Upgrade MCP #n
- **Upgrade MCP #n**
- **Logs, Metrics, Events & Alarms**

Loop Cycle for all MCPs

UnPause next MCP in Git Repo

Webhook notification

Follow Progress
Example: Move FrontHaul Configuration from DU1 to DU2
Example: Configuration Changes in GitOps

Example of GitHub flow for GitOps:

1) Operations team **creates a branch or fork** of the configuration. Changes on a branch or fork do not affect the main branch.
2) Add and **commit changes to branch** or fork
3) **Open a Pull Request (PR)**
   a) On PR **automate pipeline for initial validation** (eg. artifact validation and dry run deployment to test cluster)
4) Discussion, review and **approve changes by approvers**
5) Pipeline to **deploy to test environment** for end-to-end testing and validation
6) After all validations and approvals, changes are accepted and **merged to the main branch**

Diagrams from https://guides.github.com/introduction/flow/
End-to-End Zero-Touch Provisioning (ZTP)

GitOps makes possible End-to-End ZTP

- ZTP for ancillary configurations
  - E.g. DNS, DHCP, IPAM, SSO, Keys & Certificates, etc.
- ZTP for network elements:
  - E.g. CIDR allocations, Switches, Routers & Load Balancers configurations
- ZTP for infrastructure:
  - E.g. BIOS and firmware configurations, RAID configurations, etc
- ZTP for platform:
  - Deployment & configuration of Kubernetes clusters
- ZTP for applications:
  - Deployment and configuration of applications
  - Bootstrapping of CNF
The Journey to Zero-Touch Operations (ZTO)

GitOps paves the way for Zero-Touch Operations

- GitOps for Zero-Touch Operations (ZTO) combines:
  - GitOps
  - + End-to-End Zero-Touch Provisioning (ZTP)
  - + Observability
  - + Data Analytics
  - + Machine Learning (ML)
  - + Artificial Intelligence (AI)
Red Hat Telco RAN Automation with GitOps

PLANNING & INTEGRATION
Declarative State

Continuous Deployment
Transform Blueprints & Manifests to GitOps Structures using OpenShift Pipelines

MANAGEMENT CLUSTERS
Versioned Repository
Write Design Structures to Central Git Repository which becomes the single source of truth

Closed-Loop Automation
Synchronizes your RAN Cluster Deployments with GitOps Controllers using the Git Repository

Customer Site Design
Inventory Management
Orchestrator
iPaaS and other Tools (eg: Fuse & 3Scale)

K8s Manifests
RHACM App
ArgoCD App
Helm App
Kustomize

MANAGEMENT CLUSTERS

Closed-Loop Automation

OpenShift Serverless

Ansible

Custom Scripts

Continuous Deployment

K8s

MANAGEMENT CLUSTERS

RHACM

BYO Controller

Domain Specific Orchestrator

Infrastructure As Code

Versioned Repository

ArgoCD

OpenShift

K8s

Ansible

Continuous Deployment

K8s

Ansible

Continuous Deployment

K8s

Ansible

Continuous Deployment

K8s

Ansible

Continuous Deployment

K8s

Ansible

Continuous Deployment

K8s

Ansible

Continuous Deployment

K8s

Ansible
Innovate your future and Optimize what you have.
redhat.com/telco

linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHat