OpenSearchCon EUROPE

Operating OpenSearch at Scale with Kubernetes Operators: Lessons Learned and Community Insights

Prudhvi Godithi, Amazon and Christian Dinse, SAP SE

Agenda

- Operating OpenSearch in Kubernetes: Challenges & Learnings
  - Designing a Kubernetes Operator for OpenSearch
  - Updates & Upgrades
  - Monitoring & Status Reporting

- Community Insights: OpenSearch Operator
  - Key Features & What's New in 3.0
  - Community Insights
  - Roadmap & Getting Involved

OpenSearch @ SAP Cloud Logging

- Internal & customer-facing observability service
  - Pre-defined **configurations and contents** tailored to multiple supported runtimes
  - **Instance-based** with a high degree of administrative freedom for users
  - Service plans with **different resource specs** and OpenSearch **node topologies**
  - 2 to 30 data nodes, 10GB to 240TB instance capacity

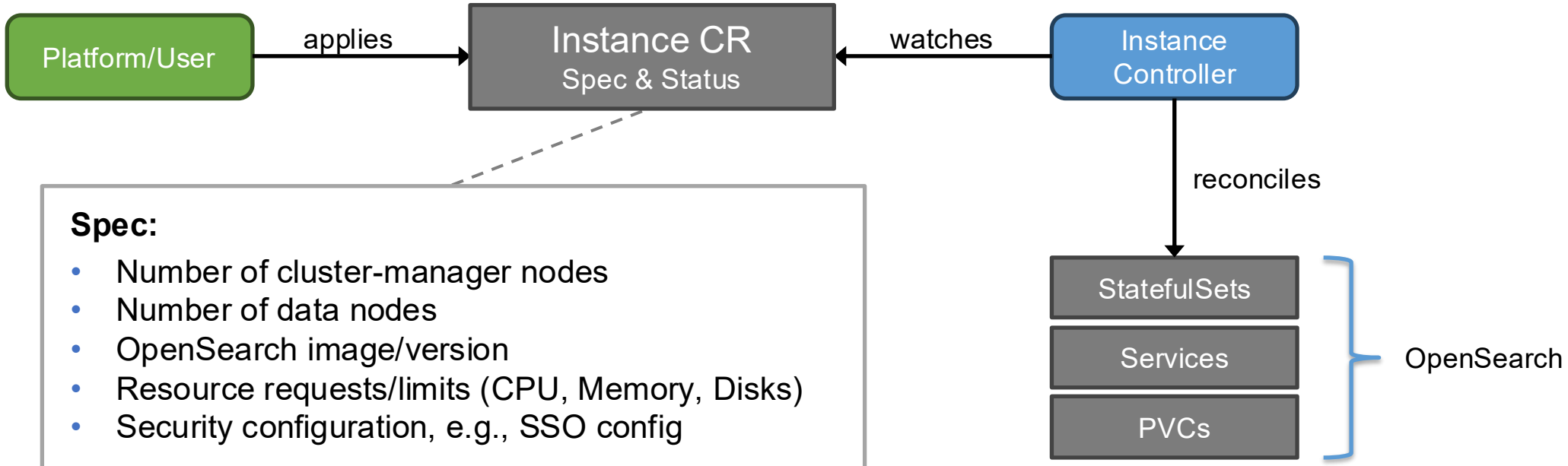
- Scale
  - 16,000+ service instances in production, each with its own OpenSearch cluster
  - 300 to 400 instances per **Kubernetes** cluster
  - Zero-downtime and high availability (HA) requirements

Designing a Kubernetes Operator for OpenSearch

What is a Kubernetes Operator?

"Operators are software extensions to Kubernetes that make use of custom resources to manage applications and their components."

Introducing a Custom Resource (CR)



Spec:

- Number of cluster-manager nodes
- Number of data nodes
- OpenSearch image/version
- Resource requests/limits (CPU, Memory, Disks)
- Security configuration, e.g., SSO config

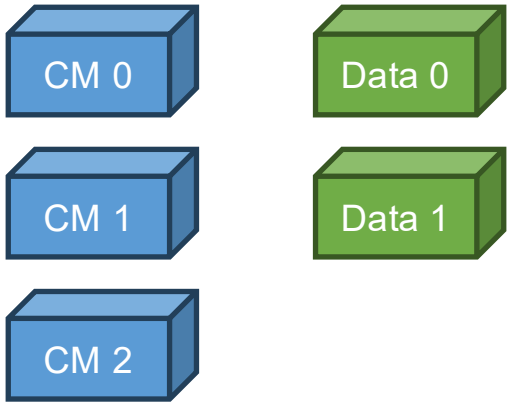
…

Status:

- OpenSearch health

…

Cluster Topology: Specification




```
Spec:
  cluster_manager_nodes: 3
  data_nodes: 2

  data_node_disk_size: 100GB
  data_node_memory_limit: 5GB
  …
```

CM: Cluster Manager

Cluster Topology: Combined Nodes?

single node w/
cluster-manager and data role

```
Spec:
  cluster_manager_nodes: 1
  data_nodes: 1

or
  combined_nodes: 1

or
  cluster_manager_data_nodes: 1
```

?

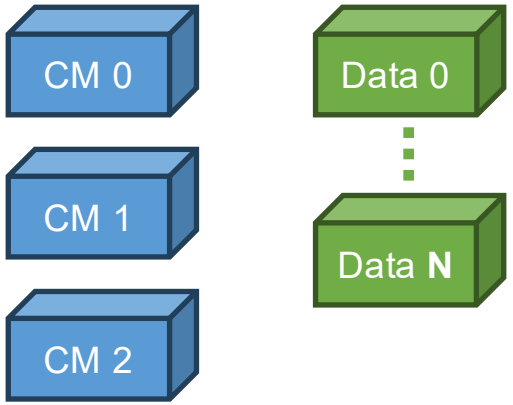
→ Extend custom-resource definition for each new combination of OpenSearch roles :/

Cluster Topology: Node Groups I

single node w/
cluster-manager and data role

```
Spec:
  groups:
  - name: single-node
    roles: cluster-manager, data
    nodes: 1
```

Cluster Topology: Node Groups II



- Enable node group specific configuration
- Allow “similar” node groups, e.g., hot/warm groups for **storage tiering**

```
Spec:
  groups:
  - name: cm-nodes
    roles: cluster-manager
    min_nodes: 3
    max_nodes: 3

  - name: data-nodes
    roles: data
    min_nodes: 2
    max_nodes: 8
```

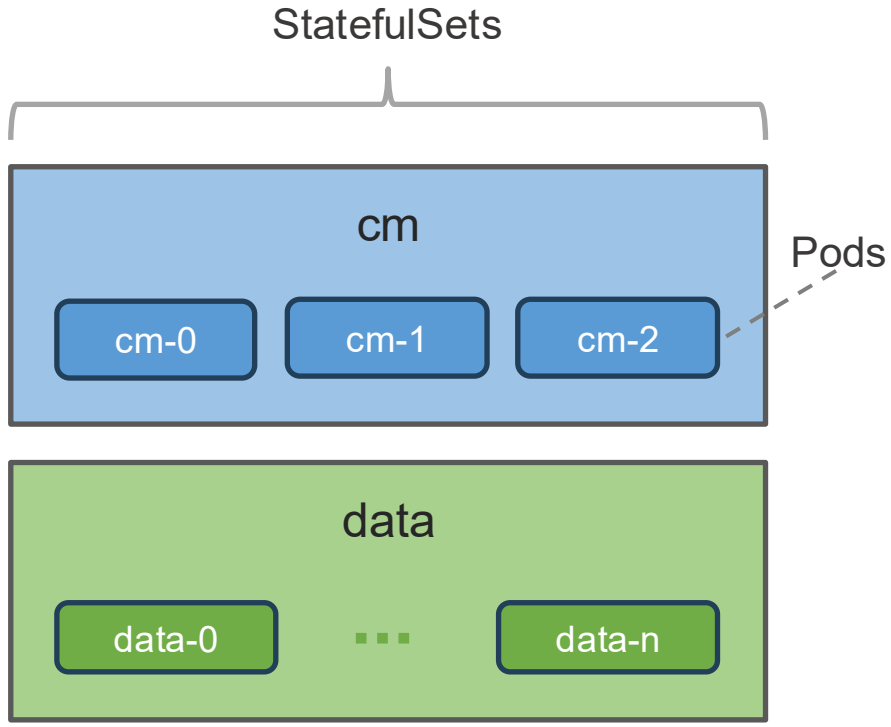
CM: Cluster Manager

Cluster Topology: Reconciliation


```
Spec:
  groups:
  - name: cm-nodes
    roles: cluster-manager
    min_nodes: 3
    max_nodes: 3

  - name: data-nodes
    roles: data
    min_nodes: 2
    max_nodes: 8
```

reconcile



+ Services, Secrets, ConfigMaps, …

+ static configuration

Updates & Upgrades

Updates & Upgrades

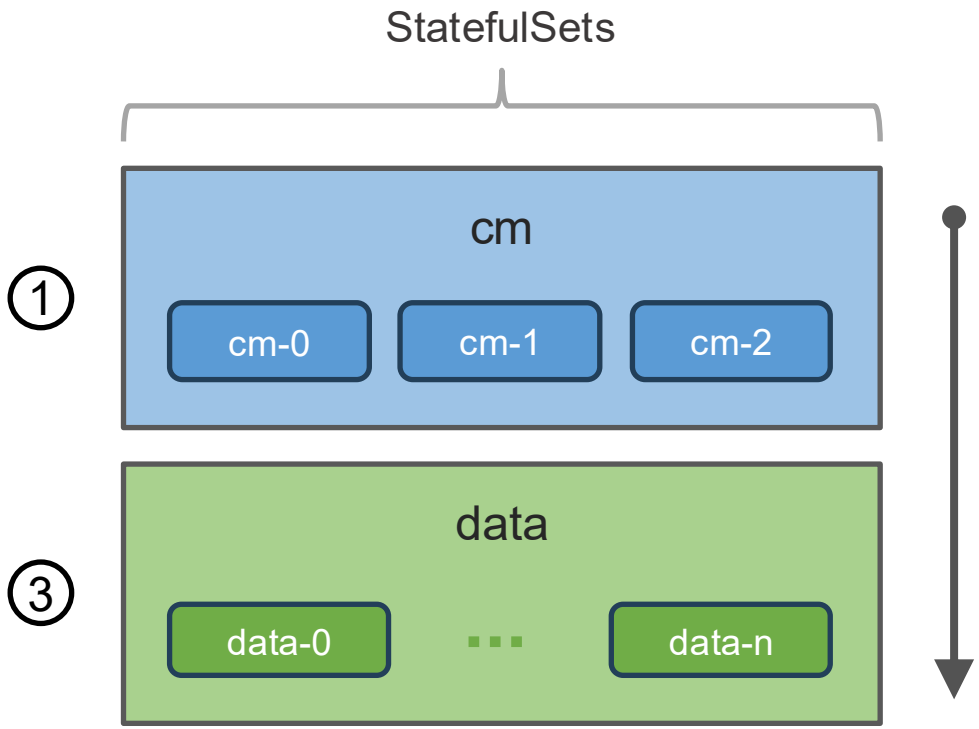
- “Update”
  - Any change that requires an OpenSearch node restart
- “Upgrade”
  - Rolling out a newer OpenSearch version to StatefulSets
- Requirements
  - No data loss
  - Zero downtime
  - High reliability
  - Speed

Updates & Upgrades: Order I

Config change only

Group	OpenSearch Roles
1	**cm** but not data
2	**cm** and **data**
3	**data** but not cm
4	not cm and not data

→ *cluster settings up-to-date AEAP*

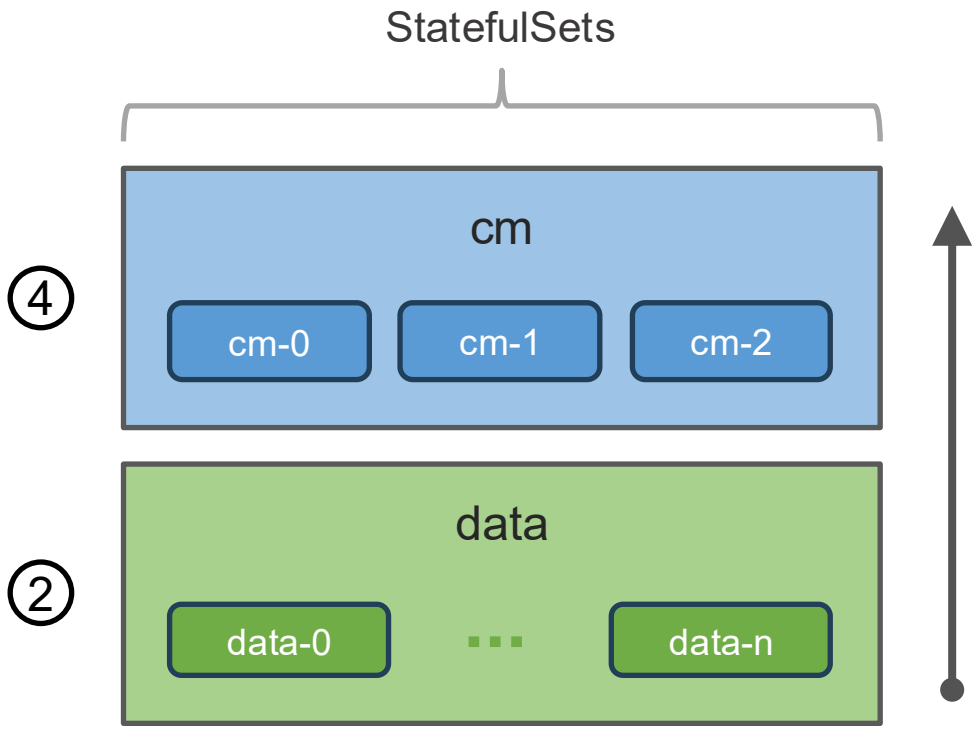


Updates & Upgrades: Order II

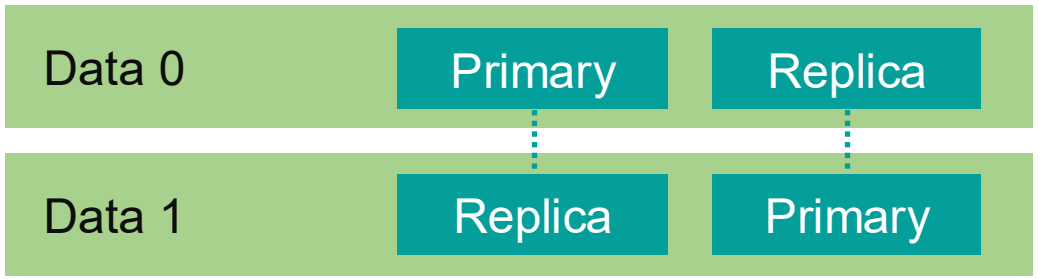
Version change (upgrade)

Group	OpenSearch Roles
1	not cm and not data
2	**data** but not cm
3	**cm** and **data**
4	**cm** but not data

→ *cluster state compatibility*



Updates & Upgrades: Shard Recovery I



X

Example Index:
two primary shards, one replica shard per primary shard

health: green

Preconditions for node restart

- All shards are assigned
- All nodes are online

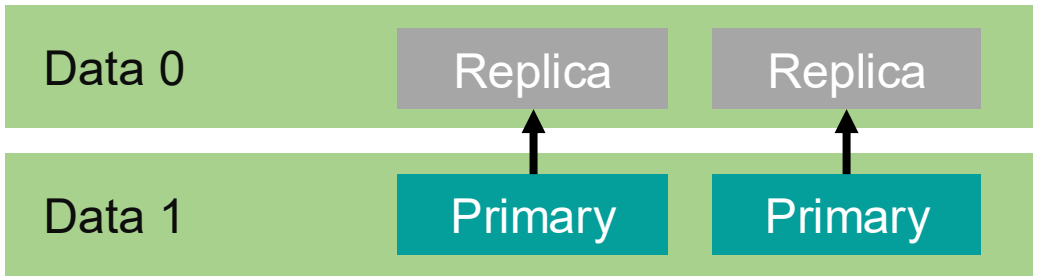
Updates & Upgrades: Shard Recovery II

Promote in-sync replica shard to primary*

- Ingestion & search continue to work
- Replication is temporarily blocked

health: **yellow**

Updates & Upgrades: Shard Recovery III



health: yellow → green

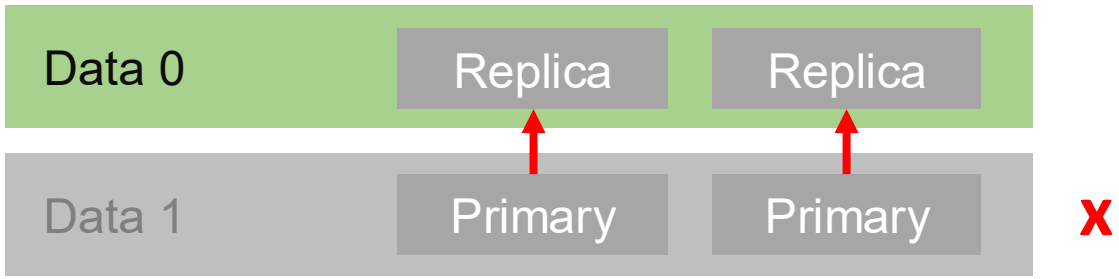
Delayed unassigned shards

- Catch up by replaying transaction log

Full replica sync

- Full rebuilt by copying index segments and replaying transaction log
- Replicas may also be assigned to another node

Updates & Upgrades: Shard Recovery IV



x

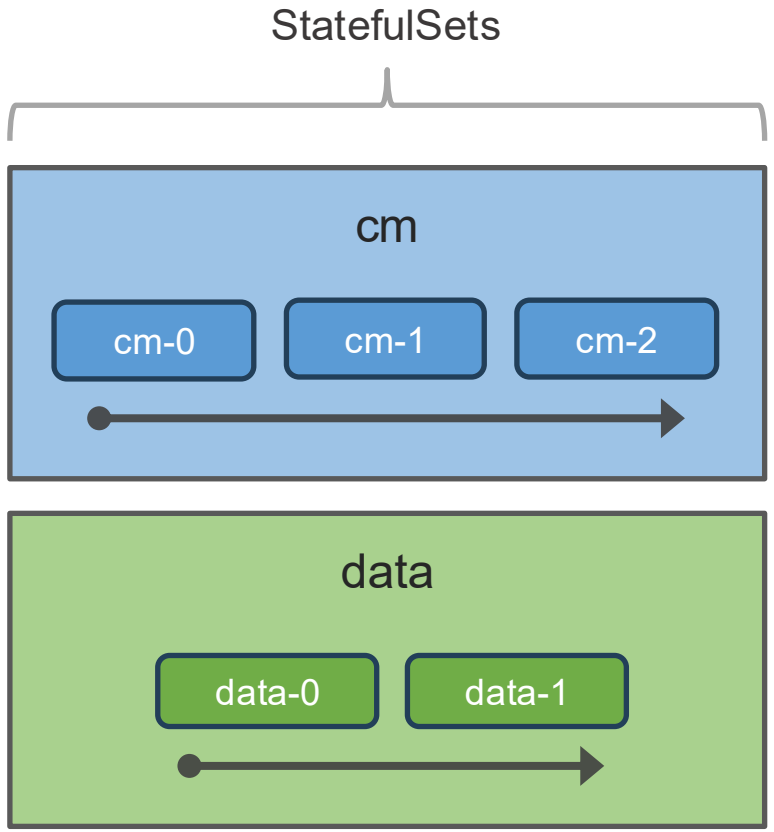
Important: Wait for Recovery!

- only restart next node when nodes and shards are recovered

health: yellow → **red**

Updates & Upgrades: Rolling Updates I


```
Spec:
   image: opensearch:1.3.20 2.19.4
   groups:
      …
```



UpdateStrategy: **RollingUpdate**

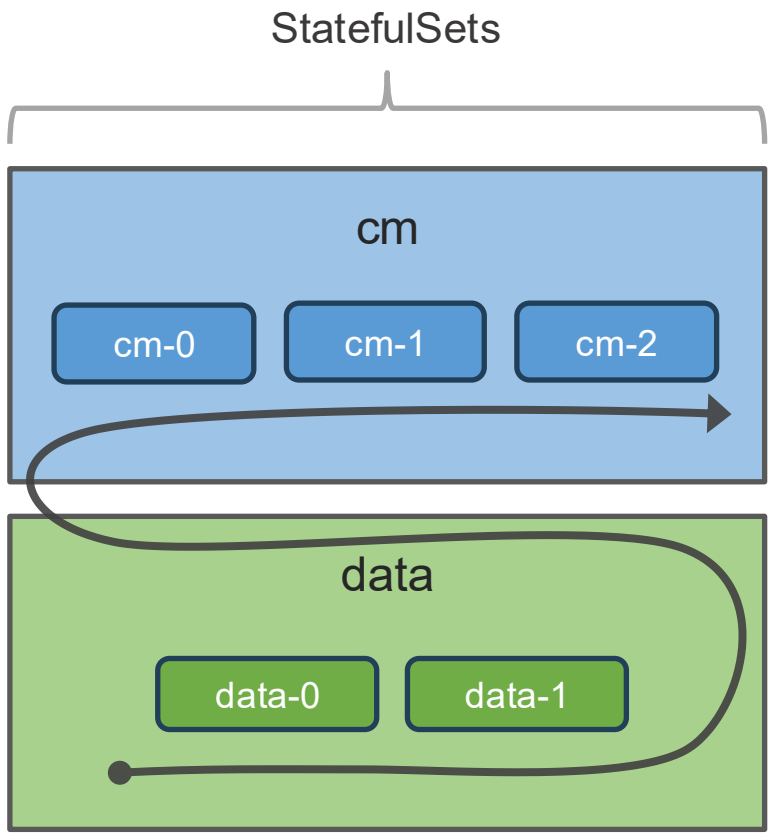
- Restart Pods one by one
- Continue when the restarted Pod is ready again

Problems:

- StatefulSets are independent → Parallel updates of cm and data nodes
- Ready != Shards recovered → Early restarts can cause downtime

Updates & Upgrades: Rolling Updates II


```
Spec:
   image: opensearch:~~1.3.20~~ 2.19.4
   groups:
      …
```



UpdateStrategy: `OnDelete`

- Restart Pods by deleting them manually (controller)

Benefits:

- Follow correct update order → Keep cluster state compatible
- Wait for shard recoveries → Zero downtime

Monitoring & Status Reporting

Monitoring & Status: Goals

- Self Observability
  - Expose observability signals to a monitoring stack for alerting and debugging
  - Cover OpenSearch, Kubernetes Operator(s), and other components
- Operator Decisions
  - Keep track of OpenSearch state for taking actions (updates, autoscaling, and more)
  - Represent OpenSearch state on Kubernetes level

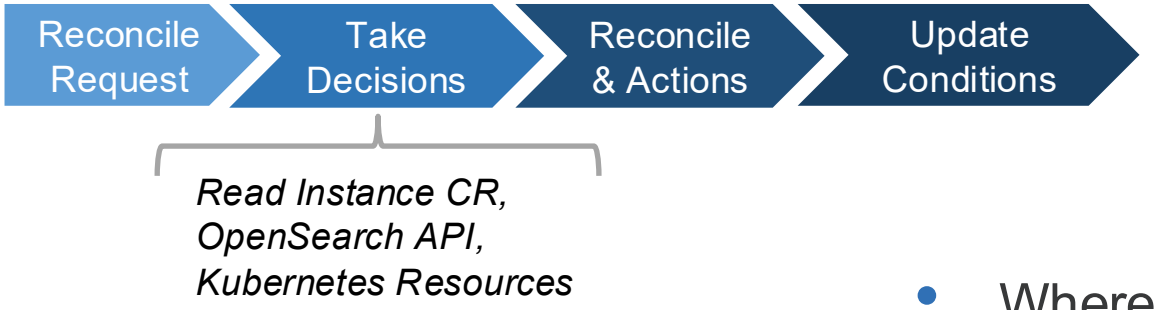
Monitoring & Status: CR Conditions


```
Spec:
  …
Status:
  conditions:
    - type: opensearch/reconciled
      status: True
      reason: succeeded
      message: …
      last_transition: …
    - type: opensearch/healthy
      status: True
      reason: green
      message: …
      last_transition: …
```

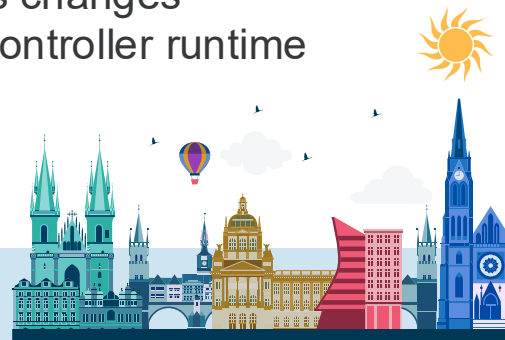
- Status list entry with
  - Type
  - Tri-State status (True, False, Unknown)
  - Machine readable reason
  - Human readable message
  - Timestamp of last transition
- Managed by controllers/reconcilers

Monitoring & Status: Watching for Changes

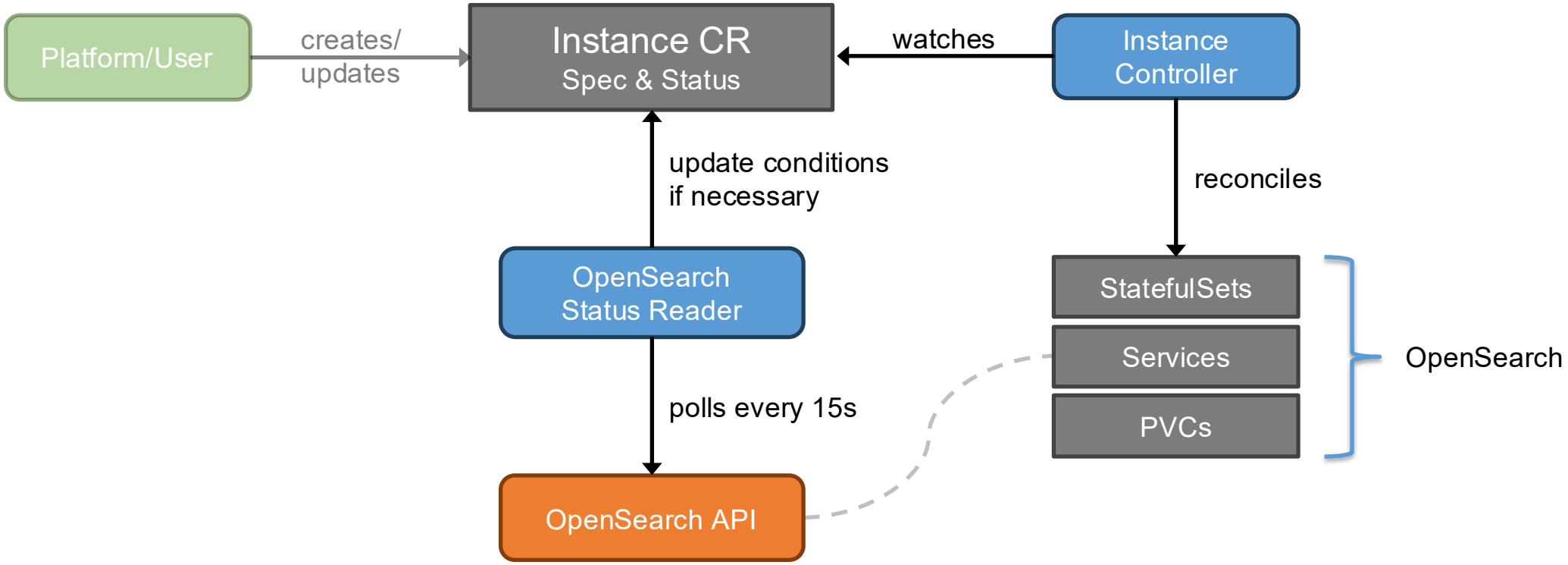
Instance Controller



- Where do reconcile requests come from?
  - Resource watches (incl. Instance CR)
  - Regular requeues at fixed interval
- Problem: latency vs reconciliation overhead
  - Fast requeues
    → Quick reaction to OpenSearch status changes
    → High load on Kubernetes APIs and controller runtime



Monitoring & Status: High-Speed Status Loop



→ fast polling triggers slow reconciliation on demand

Introducing the open-source OpenSearch Operator

Community contributed open-source Kubernetes operator for deploying, managing, and orchestrating OpenSearch and Dashboards

Repo Stats

OpenSearchCon EUROPE

Numbers from GitHub and LFX insights

530+ GitHub Stars	300+ Forks	600+ Contributors (including collaborators)	190+ Organizations Contributed
20+ Releases	V3.0.0-alpha Latest Release	600+ Issues Closed	590+ Total Pull Requests

Key Features

Cluster Management

OpenSearchCon EUROPE

Multi-Node Pool Architecture

Separate pools for cluster manager, data, ingest, coordinating, ML, each with independent resources

Dashboards Deployment

Deploy & configure OpenSearch Dashboards alongside your clusters automatically

Scaling

Scale per node pool by editing replicas, operator handles the rest

Rolling Upgrades

Operator performs quorum-safe rolling restart with health checks

Cluster Recovery

Automatic failure detection when multiple pods are down. Switches to parallel pod restart mode for faster recovery, then reverts to sequential mode once healthy

Storage & Scaling

OpenSearchCon EUROPE

Online Volume Expansion

Resize PVCs without downtime with disk scaling on the fly

Multiple Persistence Options

PVC (default), emptyDir, hostPath. Choose per node pool

Custom Storage Classes

Define storage class, access modes, PVC labels & annotations

NFS Volume Support

Direct mount NFS shares, no external provisioner or CSI driver needed

SmartScaler

Safe node drain before removal, data transferred before node goes offline

Security & TLS

OpenSearchCon EUROPE

Auto-Generated Certificates

CA + per-node certs generated automatically, configurable duration

TLS Hot Reloading

Automatic certificate rotation before expiry with hot reload — OpenSearch picks up renewed certs without pod restarts

Independent TLS Config

Transport, HTTP, and Dashboards TLS each configured separately

Declarative Security CRDs

OpensearchUser, OpensearchRole, UserRoleBinding, ActionGroup, Tenant

Custom Admin User

Auto-generated random password stored as K8s secret, bcrypt-hashed for OpenSearch security config. Supports user-provided credentials secret as override.

Configuration & Extensibility

OpenSearchCon EUROPE

Plugin Installation

Install plugins at bootstrap for OpenSearch and Dashboards

Keystore Secrets

Manage sensitive config like S3 credentials via Kubernetes secrets

Cluster Additional Configs

Custom opensearch.yml settings, general with per-nodepool overrides

Snapshot Repositories

Configure supported snapshot backends declaratively

ISM Policies via CRD

Index State Management with apply-to-existing-indices flag

Kubernetes-Native Operations

OpenSearchCon EUROPE

Init & Sidecar Containers

Add init-containers and sidecars for log shipping, monitoring, service mesh

Security Context

Pod & container security context, readOnlyRootFilesystem support

Scheduling Controls

Affinity, anti-affinity, topology spread, PDB, priority classes

Custom Probes

Configurable liveness, readiness, startup probes with custom commands

Multi-Namespace Watch

Operator watches across namespaces, custom domain name support

V3.0.0-alpha release

OpenSearch Operator 3.0 (Alpha)

3.0 is a major operator rewrite focused on correctness and resilience

- Eliminated deadlocks during cluster changes
- Fixed split-brain risks in rolling restarts
- Improved cluster stability under failure/error conditions
- Stabilized behavior across multi-AZ and complex deployments
- Over 100+ critical bug fixes and quality improvements

New Features:

- SmartScaler enabled by default
- Multi-namespace support
- Namespace-scoped RBAC
- TLS certificate hot reloading
- gRPC API support
- Improved support for multi-cluster / multi-team environments

Getting Involved

Getting Involved

OpenSearchCon EUROPE

Open an Issue

Feature requests, bug reports, questions — all welcome. When in doubt, open an issue!

Fork → Branch → PR

Standard GitHub workflow with DCO sign-off on commits

Review & Iterate

Provide constructive feedback and iterate together

Good First Issues

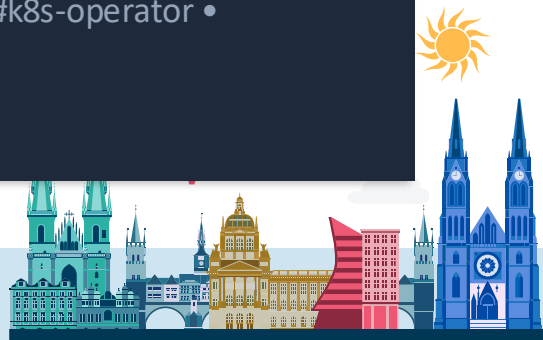
Labeled issues for newcomers, a great way to start contributing

Roadmap Development

What's next?

Join the Conversation

GitHub Issues • OpenSearch Slack #k8s-operator • Community Forum



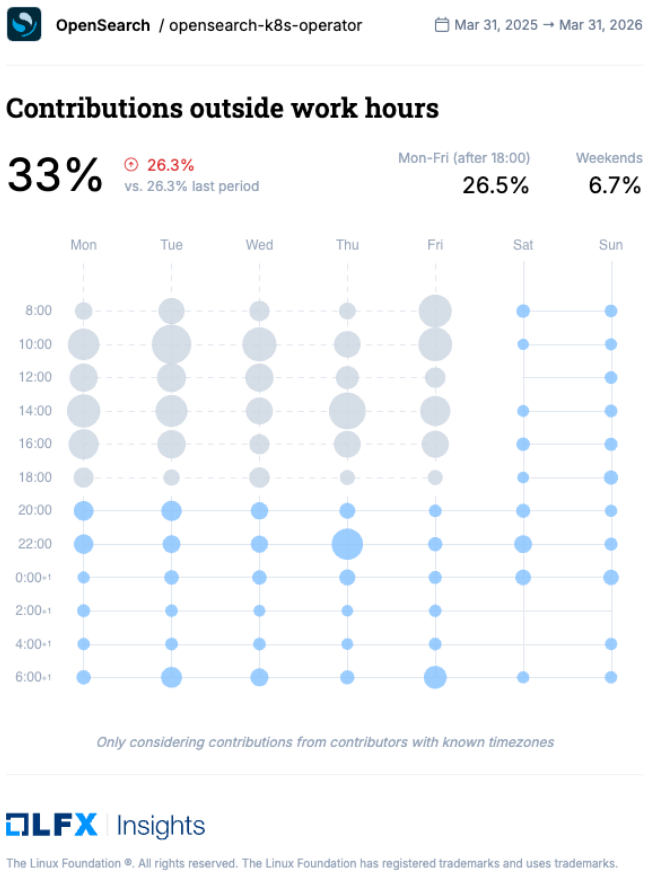
Who's building the operator?

Evenings, Weekends, and Open Source

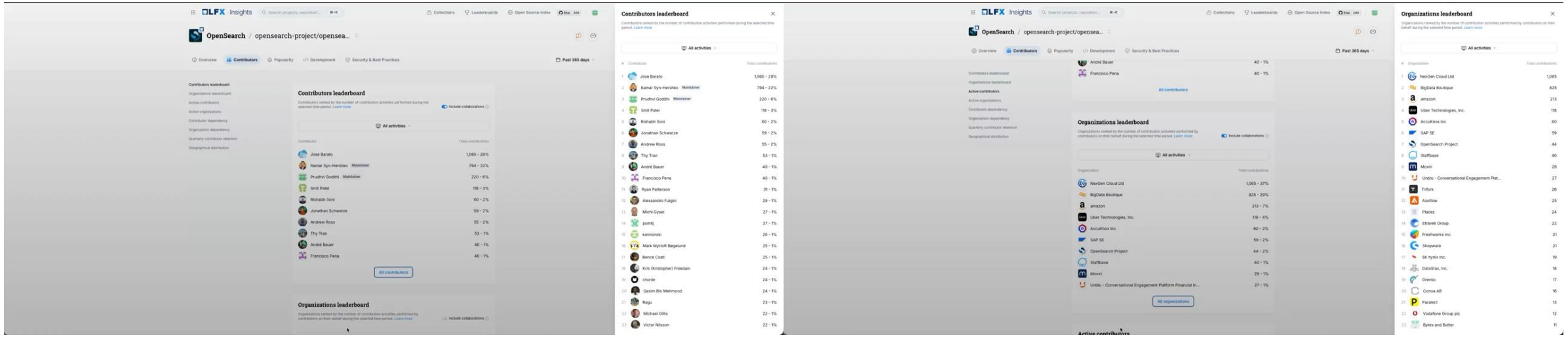
OpenSearchCon EUROPE

In last year timeline, 33% of contributions to the opensearch-k8s-operator happen outside work hours, evenings, nights, and weekends.

To every contributor who opened a PR after dinner or reviewed code on a Sunday thank you for building this operator.



Contributors & Organizations Involved



BigData Boutique - Project Maintainers

BIGDATA BOUTIQUE

BigData Boutique is an OpenSearch Foundation Member company and Accredited Support Provider.

Assumed lead maintainer role late 2025 and are the driving force behind the 3.0 release.

BDB provide enterprise support for the Operator and OpenSearch.


OpenSearchCon
EUROPE

Links

- Operating OpenSearch at Scale: Lessons From Managing 10,000+ Clusters Across Hyperscalers - Hariharan Gandhi, SAP SE
  https://www.youtube.com/watch?v=nqhfKkbj-Lw
- Kubernetes Operator Pattern
  https://kubernetes.io/docs/concepts/extend-kubernetes/operator/
- OpenSearch Kubernetes Operator
  https://github.com/opensearch-project/opensearch-k8s-operator