Mastering Kafka workload balancing with Strimzi's Cruise Control integration

Paolo Patierno

@ppatierno

- Senior Principal Software Engineer @ Red Hat
  - Messaging & data streaming
- CNCF Ambassador
- Strimzi maintainer
- Running, swimming, Formula 1 & MotoGP addicted

STRIMZI

Apache Kafka & Strimzi

STRIMZI

In a nutshell ...

- Apache Kafka
  - Distributed event streaming platform
  - A cluster is made by brokers
  - Clients exchange data/events/messages through topics split in replicated partitions
- Strimzi
  - Running Apache Kafka on Kubernetes
  - From day-1 to deploy your cluster to day-2 for managing the cluster
  - Operator pattern and Kubernetes-oriented approach

STRIMZI

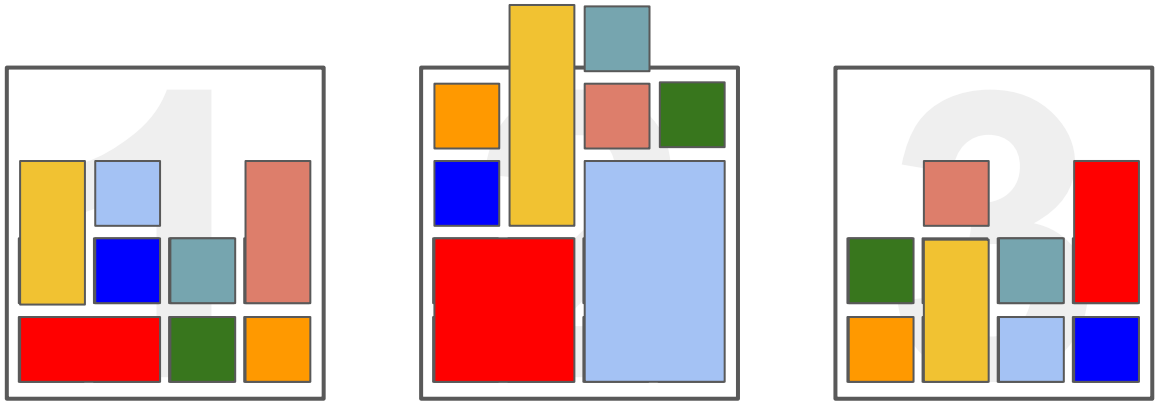
What if your load is unbalanced?

STRIMZI

The cluster is balanced at the start

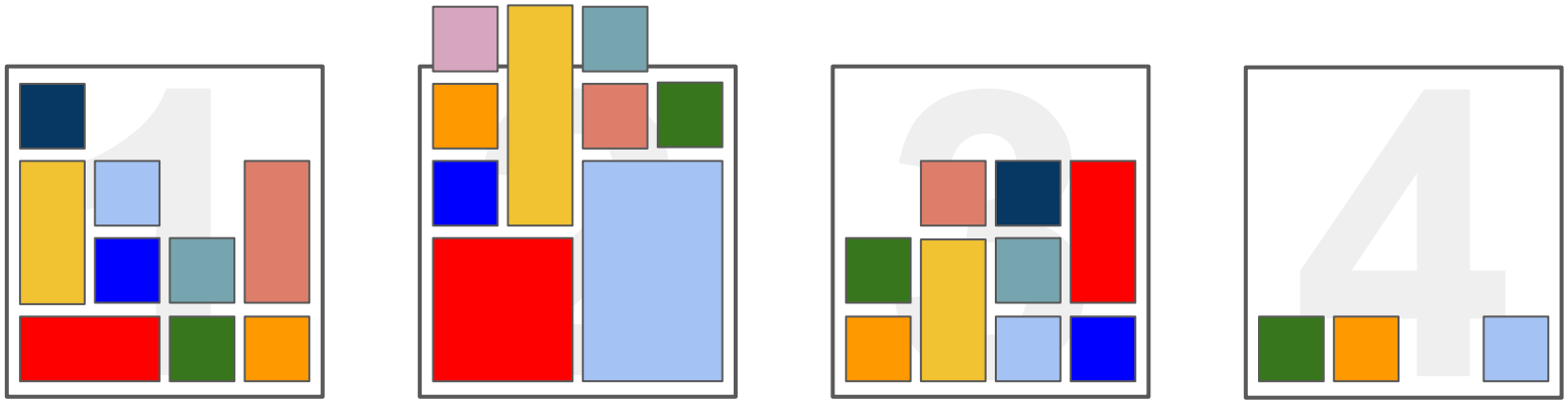
STRIMZI

Brokers get different load overtime



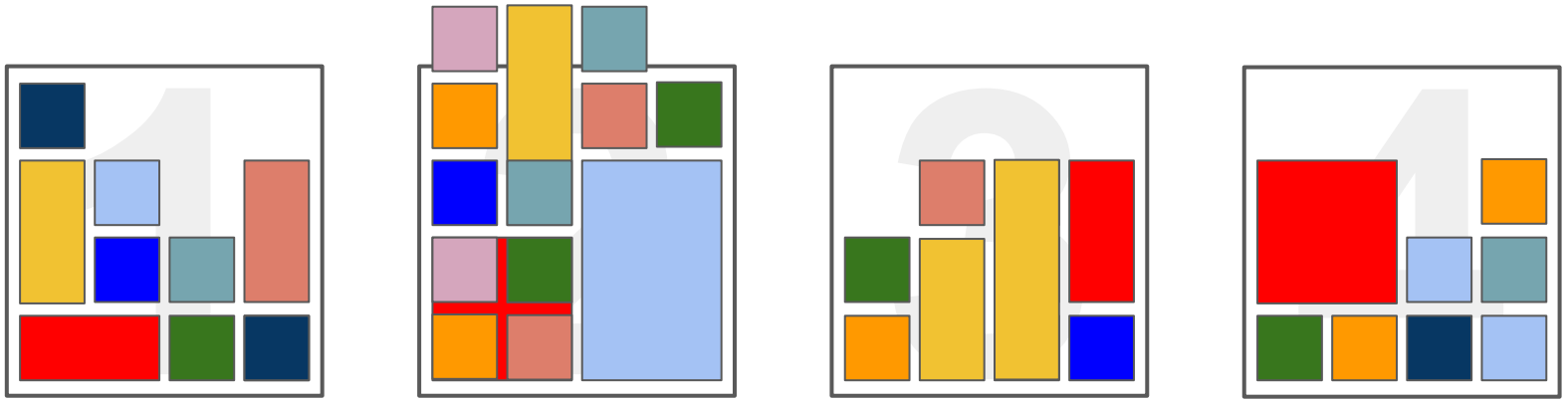
STRIMZI

Scaling doesn't help



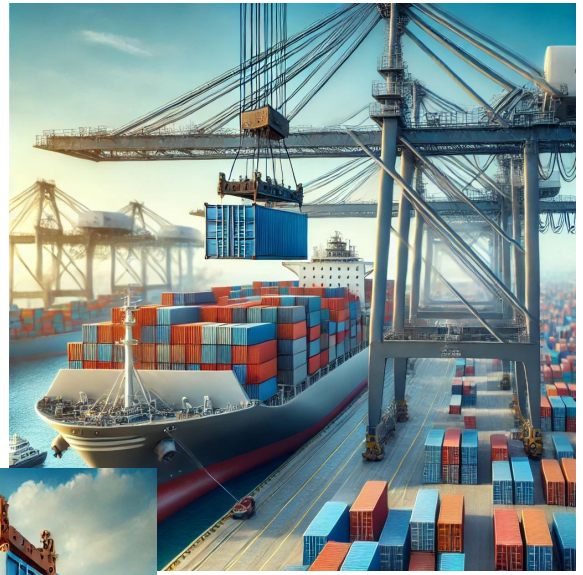
STRIMZI

Rebalance



STRIMZI

How to rebalance your load?



STRIMZI

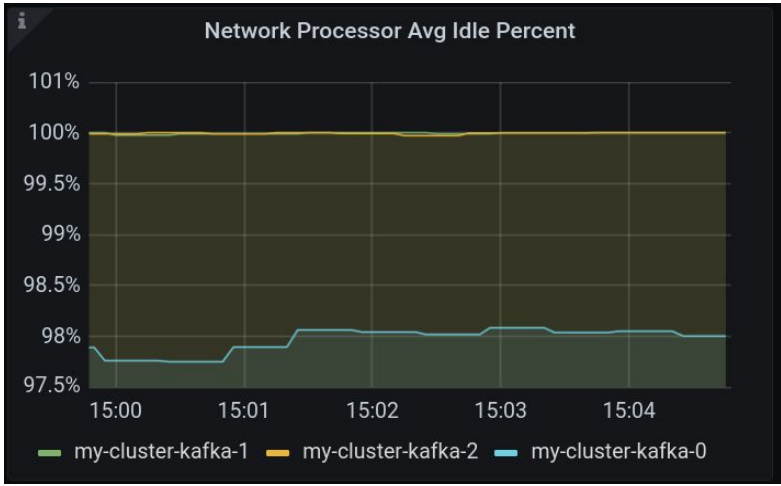
Cruise Control

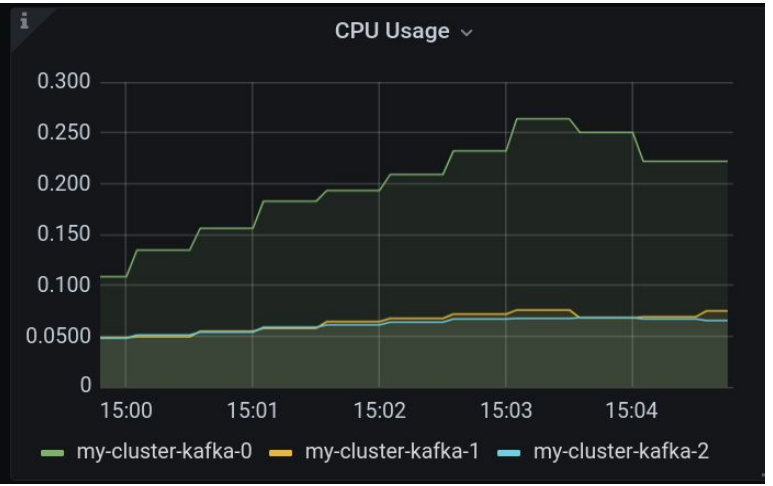
- Open Source project developed by LinkedIn
- Apache Kafka cluster re-balancing by ...
  - Monitoring CPU, disk, network throughput, and partition leadership
  - Processing an optimization proposal
  - Moving topics' partitions across brokers

Cruise Control

- Goal-based approach
  - Hard (must be satisfied) & Soft (best effort)
- Users can define constraints:
  - Disk usage
  - Optimal network load, CPU and memory utilization
  - Distribution of leader replicas
  - Rack awareness

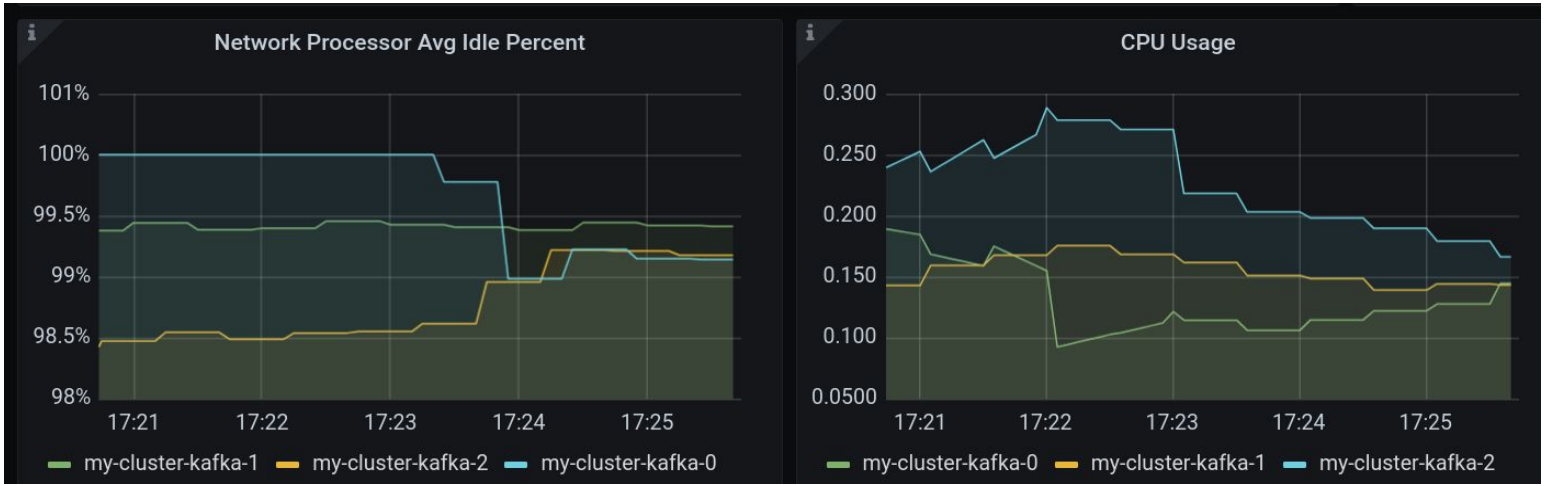
Unbalanced workload



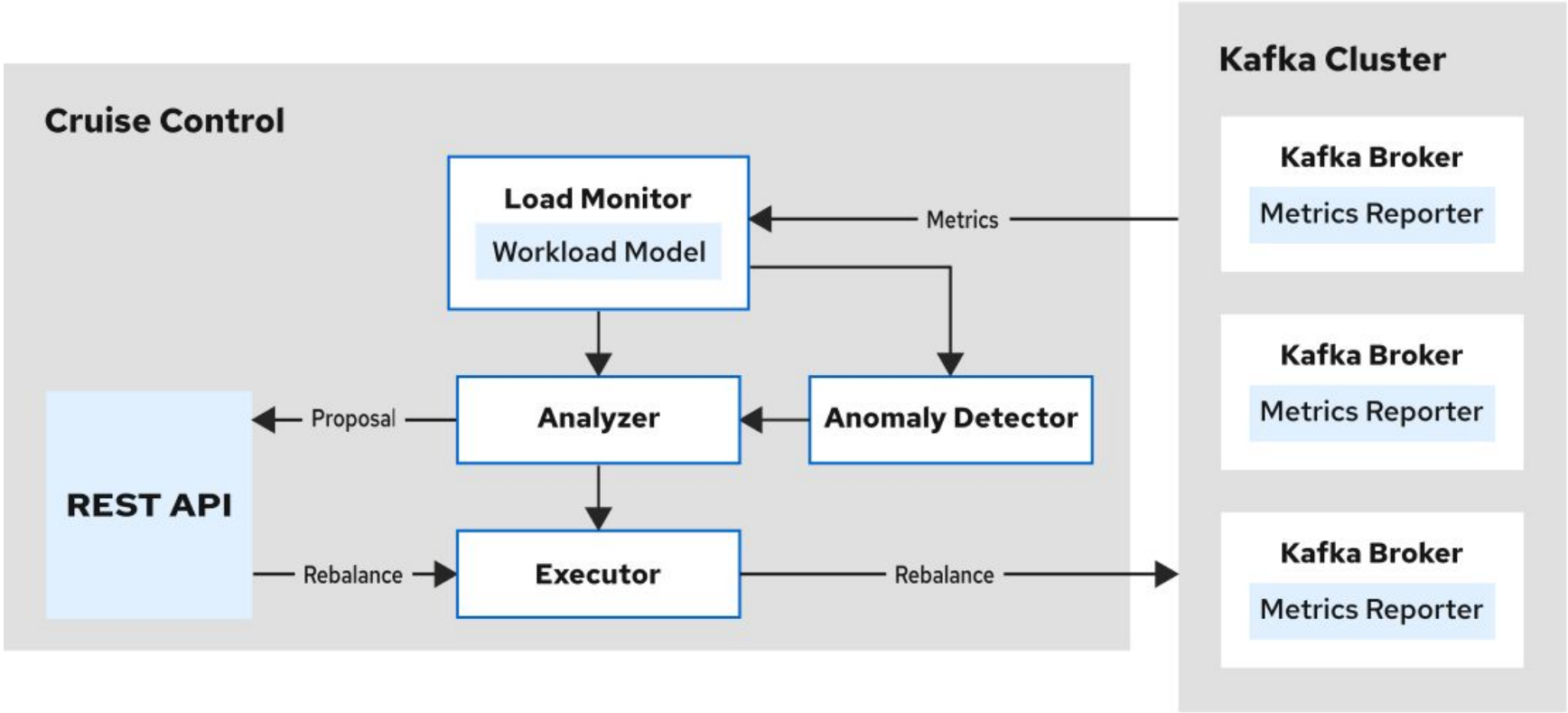


STRIMZI

Balanced workload



STRIMZI


Cruise Control
Load Monitor
Workload Model
Metrics
Analyzer
Anomaly Detector
Proposal
REST API
Rebalance
Executor
Rebalance
Kafka Cluster
Kafka Broker
Metrics Reporter
Kafka Broker
Metrics Reporter
Kafka Broker
Metrics Reporter

STRIMZI

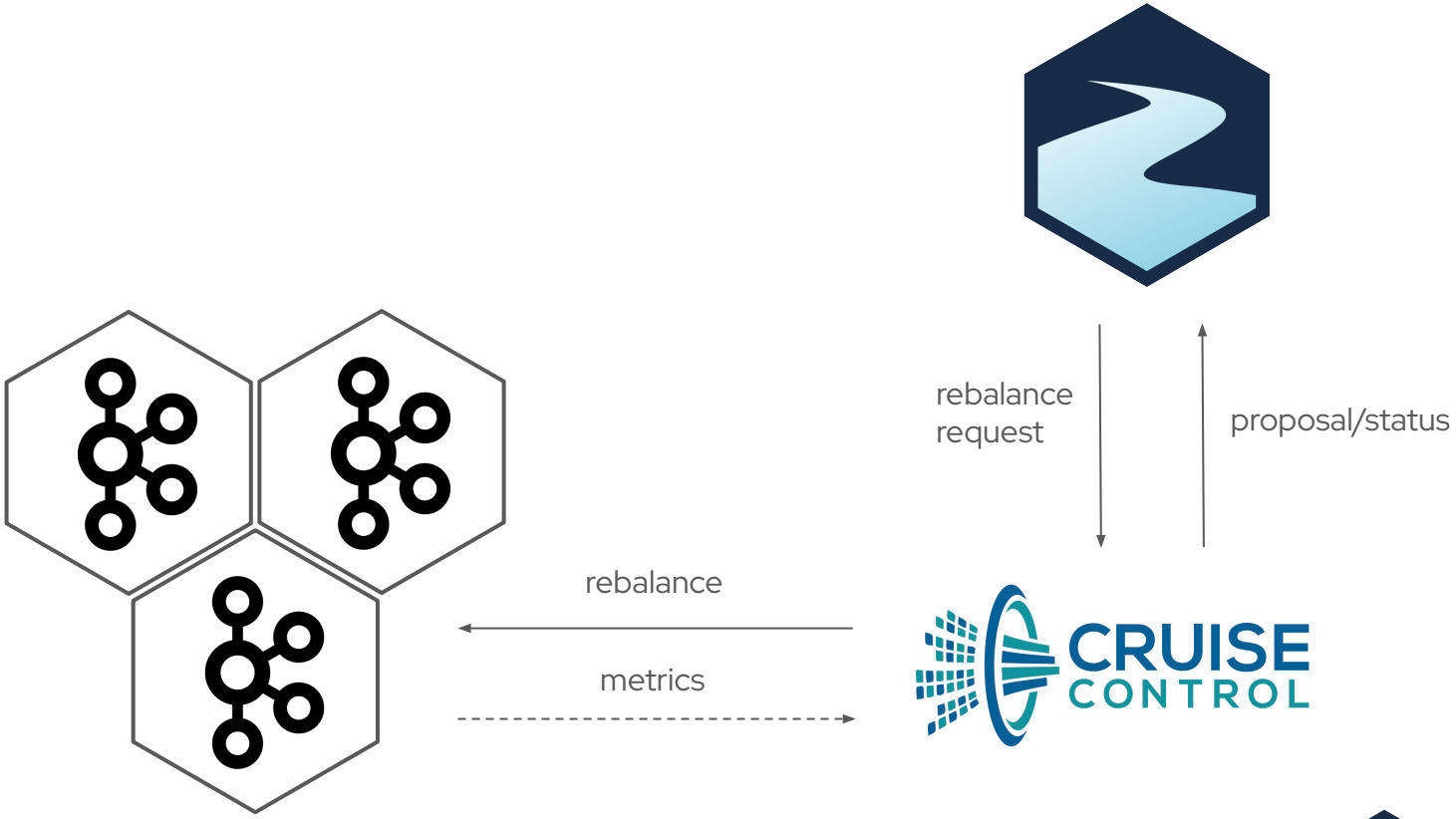
Strimzi's Cruise Control Integration

STRIMZI

Configuration & Deployment

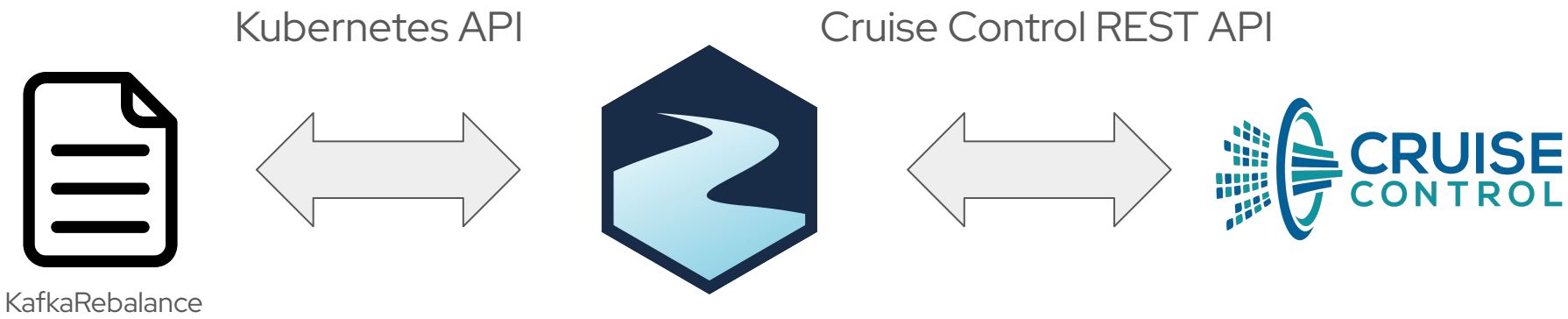
- Cruise Control enabled within the `Kafka` custom resource
- Configured via the `spec.cruiseControl` field
  - Default goals configuration
  - Broker capacity (e.g. network, CPU, storage)
  - Logging, resources, ...
- Run as a separated Deployment
- Brokers configured to use metrics reporter

STRIMZI


rebalance
request
proposal/status
rebalance
metrics
CRUISE
CONTROL

STRIMZI

Kube-native rebalancing experience

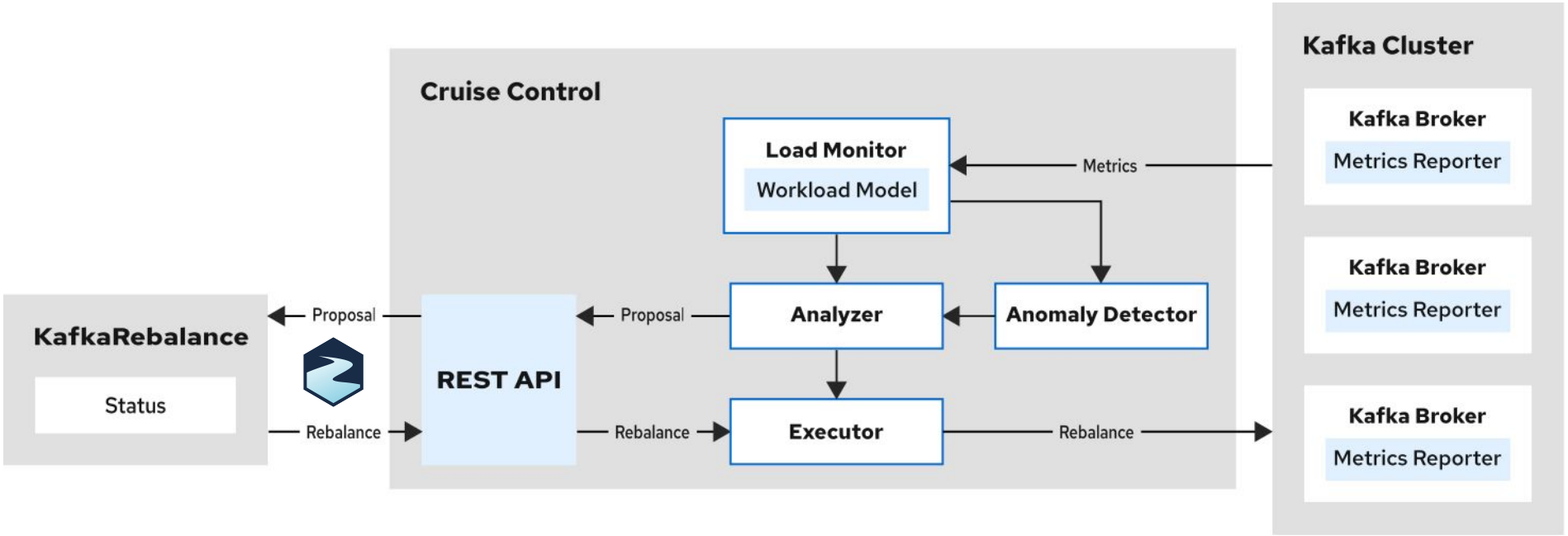


STRIMZI

User creates a `KafkaRebalance`

```yaml
apiVersion: kafka.strimzi.io/v1beta2
Kind: KafkaRebalance
metadata:
 Name: my-rebalance
 labels:
   strimzi.io/cluster: my-cluster
spec:
 goals:
   - CpuCapacityGoal
   - NetworkInboundCapacityGoal
   - DiskCapacityGoal
   - RackAwareGoal
 mode: full
 replicationThrottle: 100
 # ... and more
```


STRIMZI


Kafka Cluster
Kafka Broker
Metrics Reporter
Kafka Broker
Metrics Reporter
Kafka Broker
Metrics Reporter
Cruise Control
Load Monitor
Workload Model
Metrics
Analyzer
Anomaly Detector
REST API
Executor
Proposal
Proposal
Rebalance
Rebalance
Rebalance
KafkaRebalance
Status

STRIMZI

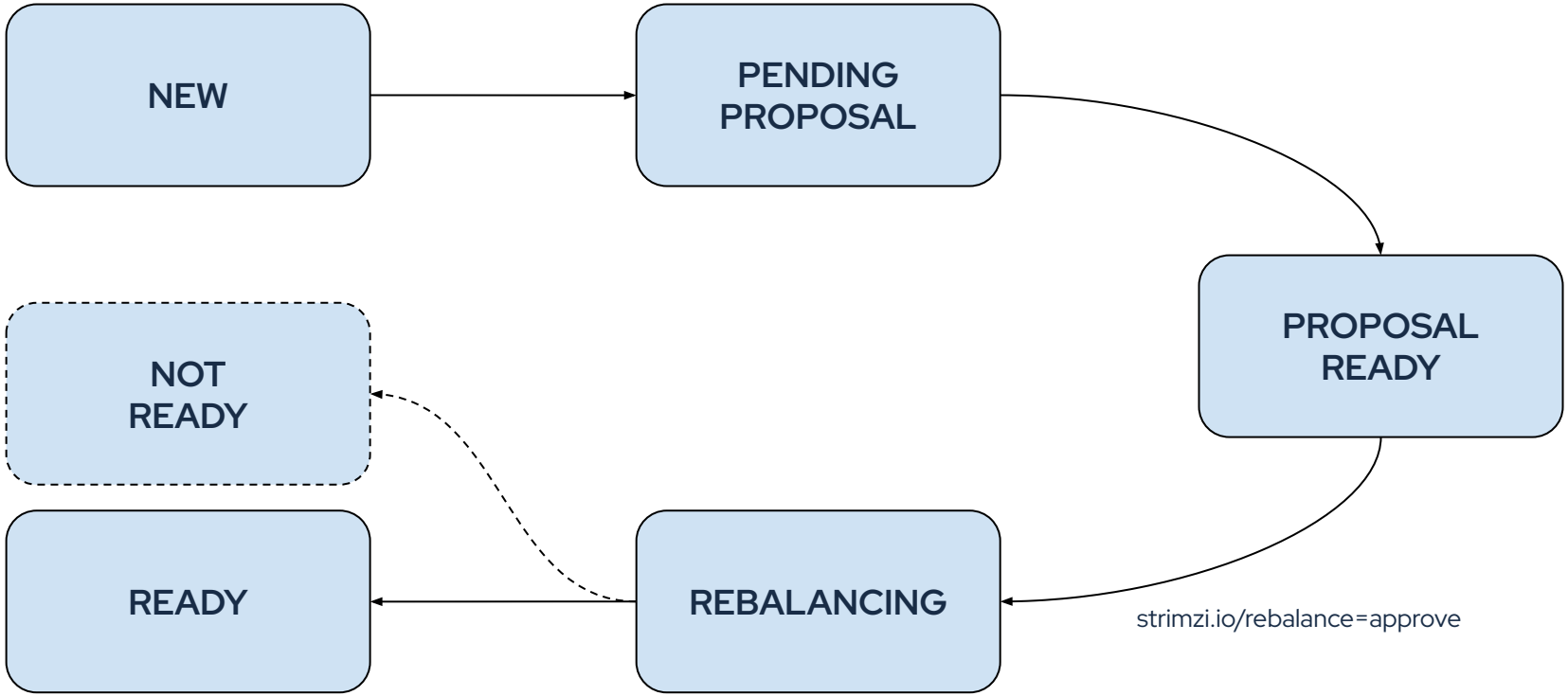
Cruise Control returns a proposal

```
status:
  conditions:
  - lastTransitionTime: "2025-03-24T03:45:16.824326Z"
    status: "True"
    type: ProposalReady
  optimizationResult:
    dataToMoveMB: 5643
    excludedBrokersForLeadership: []
    excludedBrokersForReplicaMove: []
    excludedTopics: []
    intraBrokerDataToMoveMB: 0
    monitoredPartitionsPercentage: 100
    numIntraBrokerReplicaMovements: 0
    numLeaderMovements: 4
    numReplicaMovements: 67
    onDemandBalancednessScoreAfter: 98
    onDemandBalancednessScoreBefore: 85
    # …
```

Annotation-driven approach

- Use `strimzi.io/rebalance` annotation to:
  - `approve` a proposal
  - `stop` the current proposal generation or ongoing rebalancing
  - `refresh` a proposal
- Use auto-approval mechanism via `strimzi.io/rebalance-auto-approval=true`

STRIMZI


NEW
PENDING PROPOSAL
PROPOSAL READY
NOT READY
READY
REBALANCING
strimzi.io/rebalance=approve

STRIMZI

Rebalancing `mode`(s)

- **`full`** (default)
  - To run a full rebalance across all the brokers in the cluster
- **`add-brokers`**
  - To move replicas across newly created `brokers` after a scaling up
- **`remove-brokers`**
  - To move replicas out of the `brokers` to be removed before doing a scaling down
- **`remove-disks`**
  - To move replicas between JBOD disks used for storage on the same broker

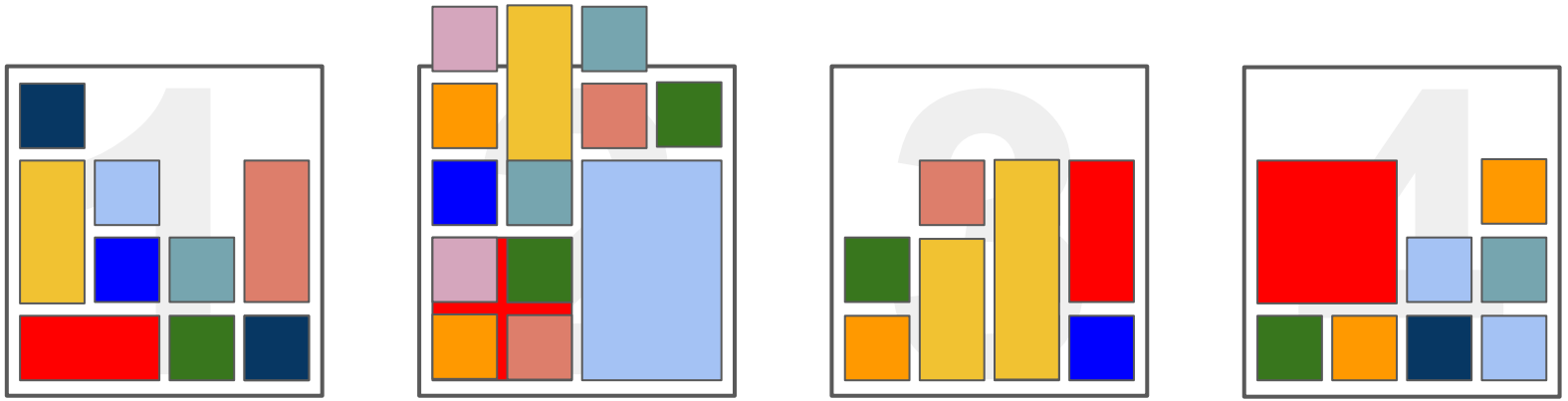
Auto-rebalancing on cluster scaling

STRIMZI

Auto-rebalancing

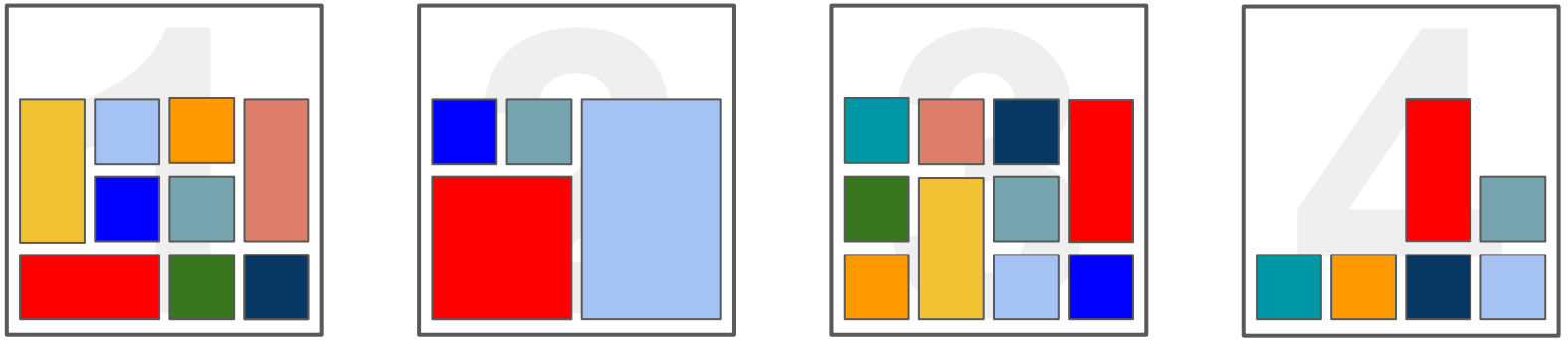
- Automatically triggers a rebalance when brokers are added or removed
  - Makes it easier to scale the Kafka cluster up or down
  - Moves partition replicas to new brokers after they were added
  - Moves partition replicas from brokers before they are removed
- Rebalancing via a `KafkaRebalance` template

Scale-up



STRIMZI

Scale-down



STRIMZI

What's next?

STRIMZI

Kafka cluster self-healing

- Strimzi leverages Cruise Control for cluster rebalancing
  - Manual rebalancing involves the usage of a **`KafkaRebalance`** custom resource
  - Auto-rebalancing helps only on scaling the cluster up or down
- Plan to integrate the Cruise Control self-healing feature
  - Anomaly detectors to detect broker or disk failures, goal violation, topic anomalies and more
  - Anomalies being notified and fixed where possible

STRIMZI

Join us

https://strimzi.io/join-us/

STRIMZI

Do you want more about Strimzi?

STRIMZI

Other talks

- Strimzi: What's New and What's Next
  - Project lightning talks, Wed 1st April 15:46 BST
- Simplifying Apache Kafka on Kubernetes with Strimzi
  - Maintainer Track, Wed 2nd April 15:15 BST

Thank you

 Website: https://strimzi.io

 GitHub: https://github.com/strimzi

 Twitter: @strimziio

 YouTube: https://youtube.com/c/Strimzi

 LinkedIn: https://www.linkedin.com/company/strimzi

STRIMZI