KubeCon | CloudNativeCon

Europe 2025

KubeCon | CloudNativeCon
Europe 2025

Beyond the Limits: Scaling Kubernetes Controllers Horizontally

Tim Ebert, STACKIT

Introduction

Introduction – Context

- Managing thousands of clusters at STACKIT Kubernetes Engine
- Based on open source project Gardener
- Running controllers at scale
- Master's thesis: "Horizontally Scalable Kubernetes Controllers"

STACKIT
A Brand of Schwarz Digits

Gardener

Introduction – Controller Basics

Controllers facilitate declarative state management:

1. Watch objects for changes
2. Cache objects in memory
3. Enqueue object key on relevant changes
4. Read current state (from cache)
5. Make changes
6. Report observed status

Demo

Problem Statement

Problem Statement

- Controllers must prevent conflicts
- Perform leader election
- Only a single active instance
- Controllers are not horizontally scalable
- Limits large-scale use cases
- No standard solution exists

Demo

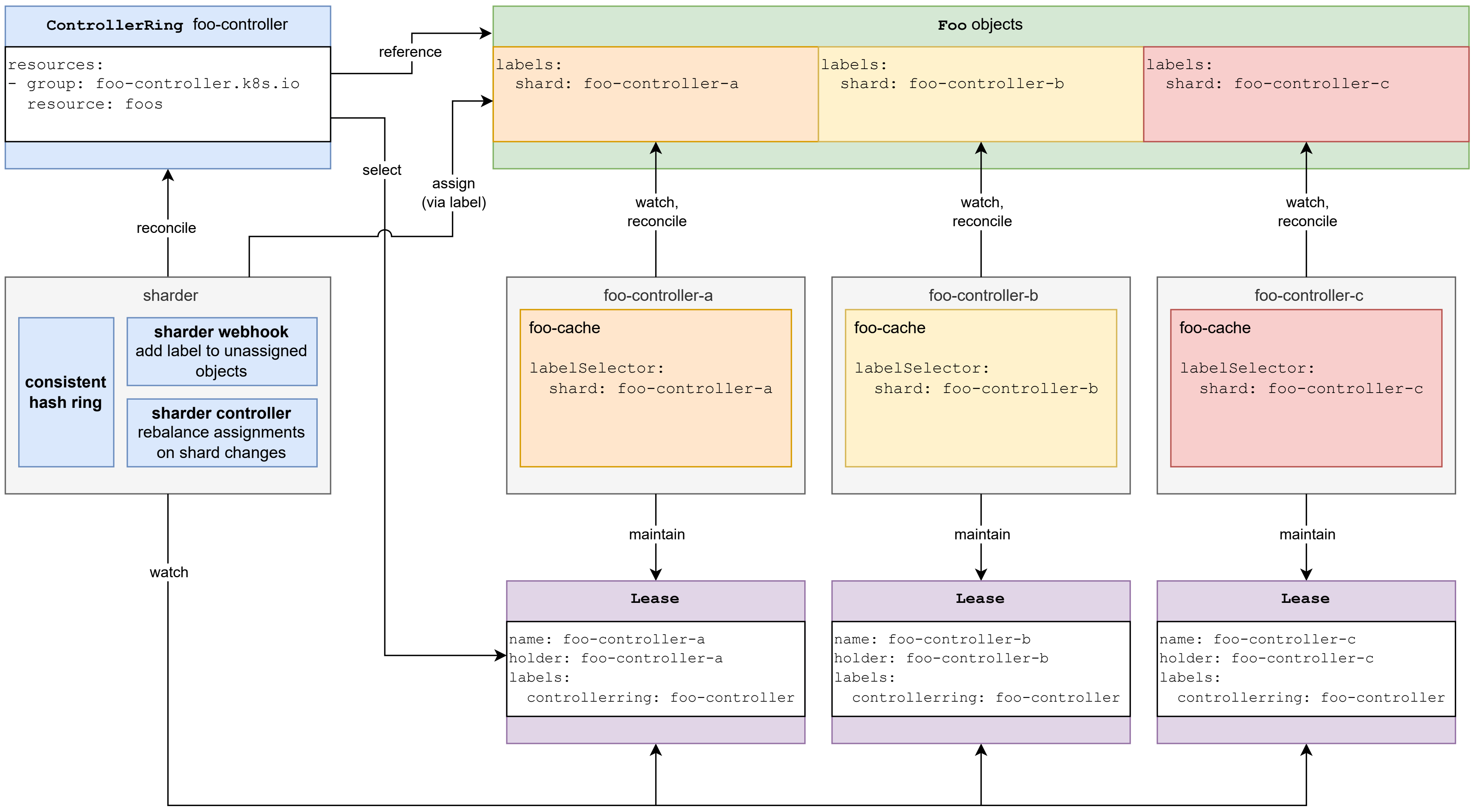
Design

Design – Key Features

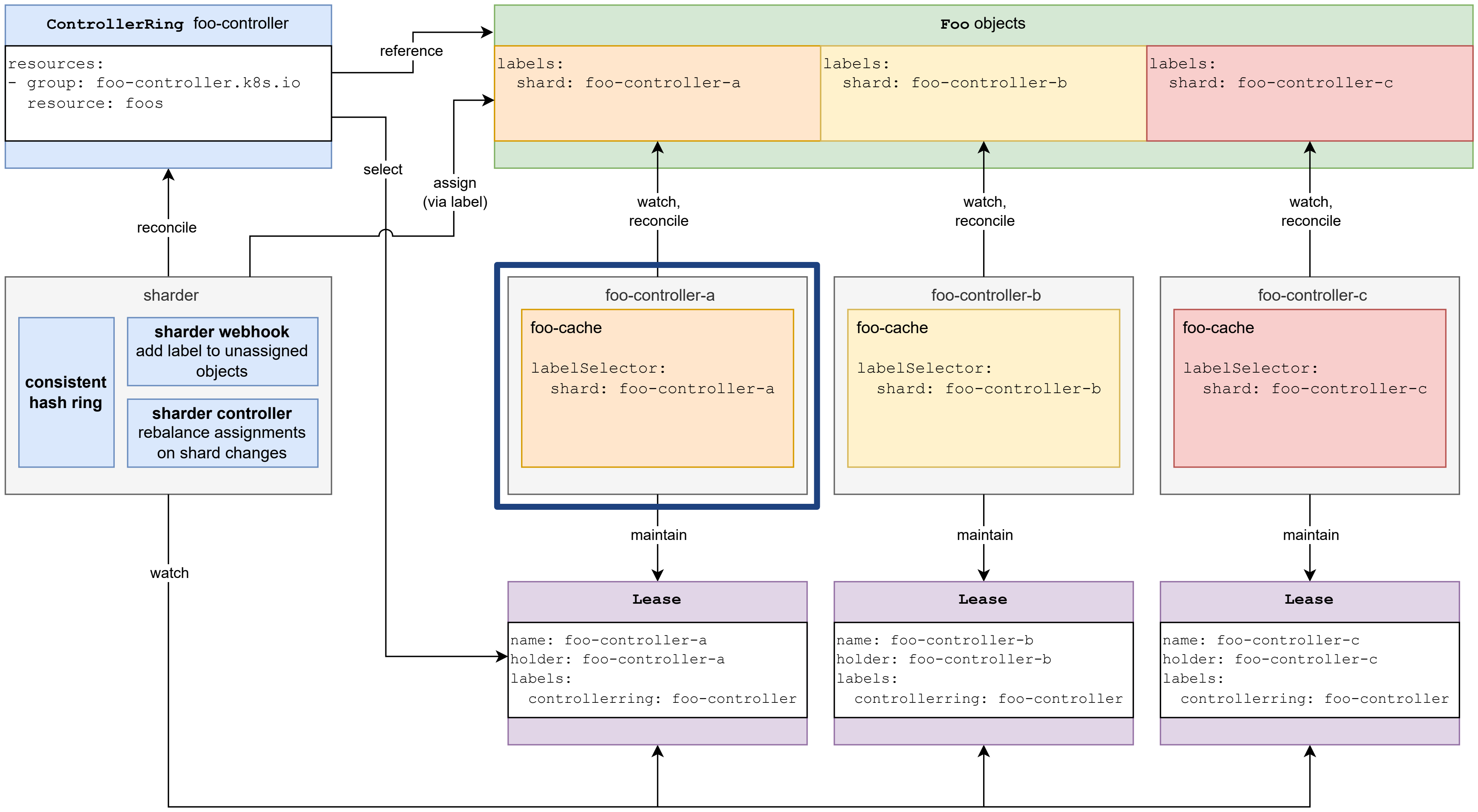
- Sharding mechanisms inspired by distributed databases
- Dynamic membership and failure detection
- Automatic failover and rebalancing
- Transparent label-based coordination
- Prevents concurrent reconciliations
- Reusable implementation

Design

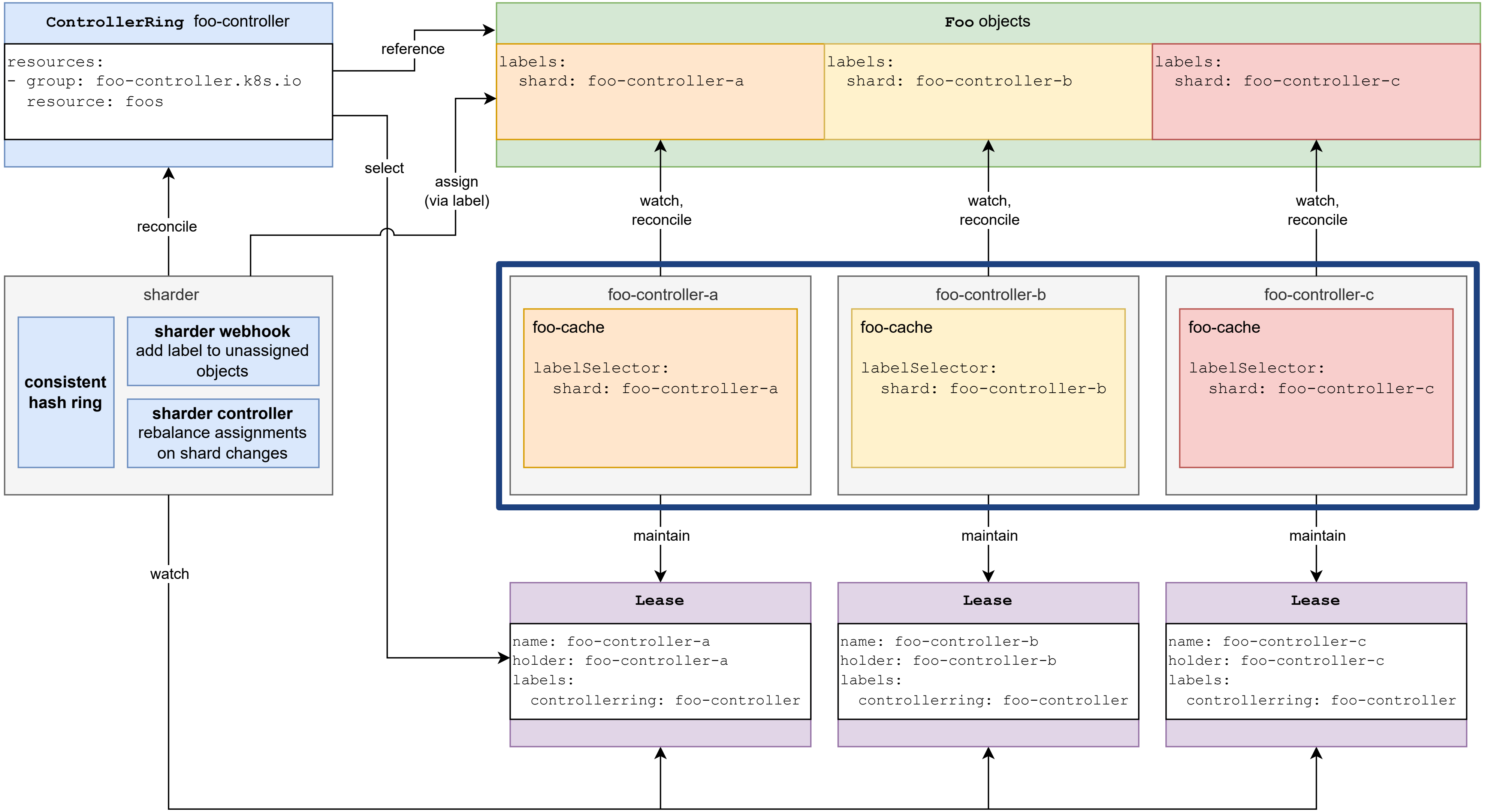
KubeCon CloudNativeCon Europe 2025



Design


ControllerRing foo-controller
resources:
- group: foo-controller.k8s.io
  resource: foos
reference
select
assign (via label)
Foo objects
labels:
  shard: foo-controller-a
labels:
  shard: foo-controller-b
labels:
  shard: foo-controller-c
reconcile
watch, reconcile
watch, reconcile
watch, reconcile
sharder
consistent hash ring
sharder webhook
add label to unassigned objects
sharder controller
rebalance assignments on shard changes
foo-controller-a
foo-cache
labelSelector:
  shard: foo-controller-a
foo-controller-b
foo-cache
labelSelector:
  shard: foo-controller-b
foo-controller-c
foo-cache
labelSelector:
  shard: foo-controller-c
maintain
maintain
maintain
watch
Lease
name: foo-controller-a
holder: foo-controller-a
labels:
  controllerring: foo-controller
Lease
name: foo-controller-b
holder: foo-controller-b
labels:
  controllerring: foo-controller
Lease
name: foo-controller-c
holder: foo-controller-c
labels:
  controllerring: foo-controller

Design



ControllerRing foo-controller

```
resources:
- group: foo-controller.k8s.io
  resource: foos
```

Foo objects

```
labels:
  shard: foo-controller-a
```

```
labels:
  shard: foo-controller-b
```

```
labels:
  shard: foo-controller-c
```

sharder

consistent hash ring

sharder webhook add label to unassigned objects

sharder controller rebalance assignments on shard changes

foo-controller-a

foo-cache

```
labelSelector:
  shard: foo-controller-a
```

foo-controller-b

foo-cache

```
labelSelector:
  shard: foo-controller-b
```

foo-controller-c

foo-cache

```
labelSelector:
  shard: foo-controller-c
```

Lease

```
name: foo-controller-a
holder: foo-controller-a
labels:
  controllerring: foo-controller
```

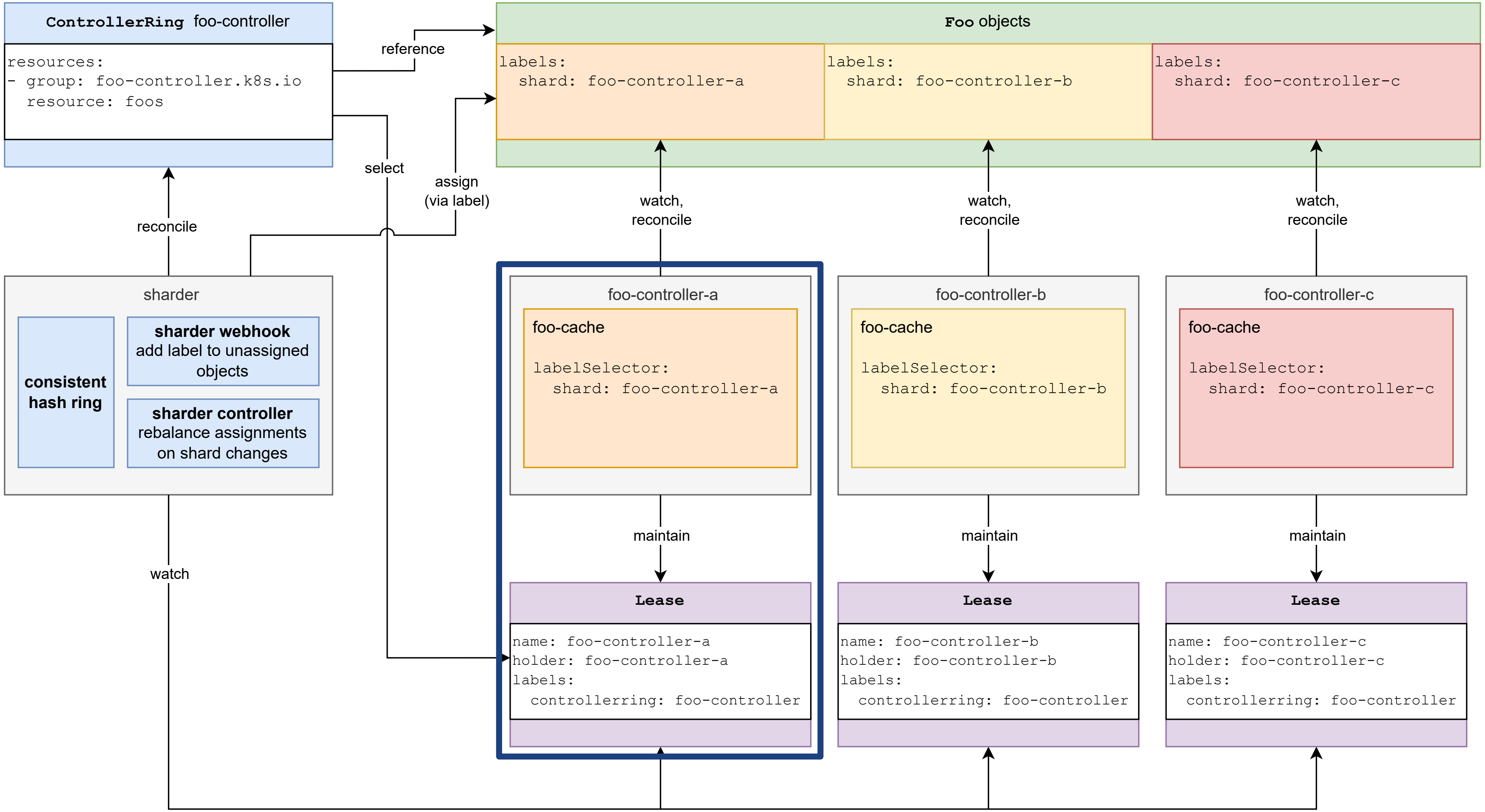
Lease

```
name: foo-controller-b
holder: foo-controller-b
labels:
  controllerring: foo-controller
```

Lease

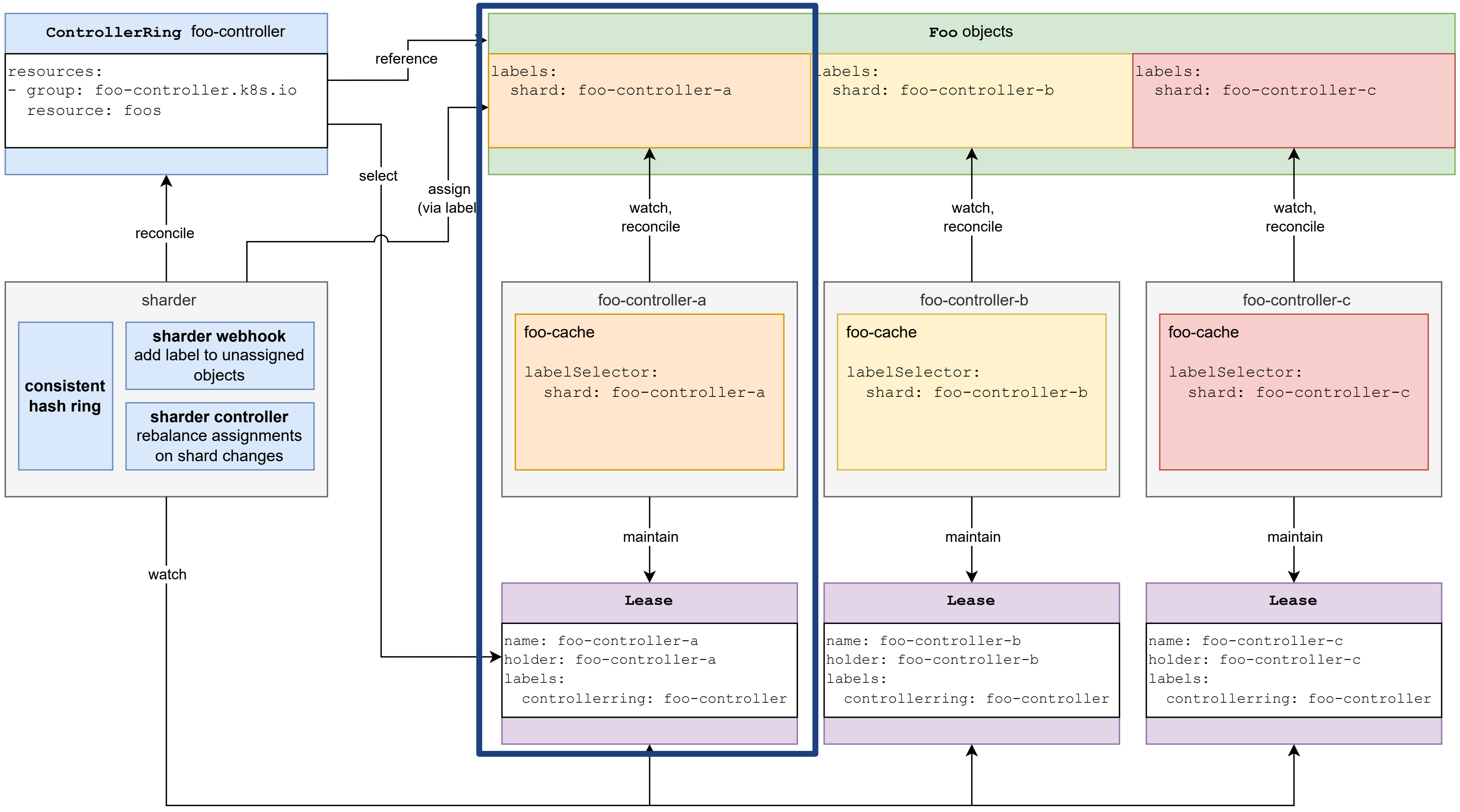
```
name: foo-controller-c
holder: foo-controller-c
labels:
  controllerring: foo-controller
```

Design

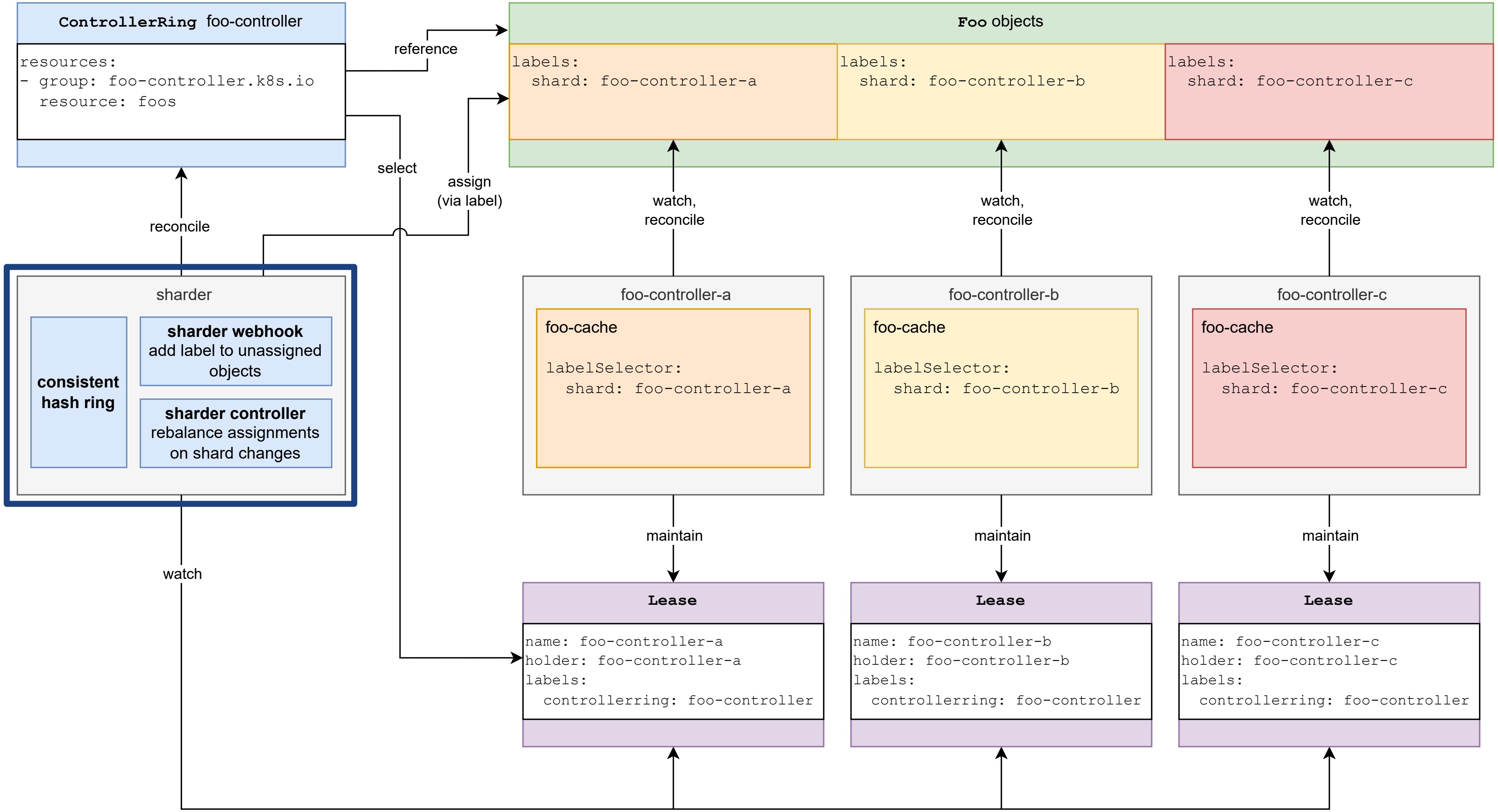


ControllerRing foo-controller
resources:
- group: foo-controller.k8s.io
  resource: foos
reference
select
assign (via label)
reconcile
Foo objects
labels:
  shard: foo-controller-a
labels:
  shard: foo-controller-b
labels:
  shard: foo-controller-c
watch, reconcile
watch, reconcile
watch, reconcile
sharder
consistent hash ring
sharder webhook
add label to unassigned objects
sharder controller
rebalance assignments on shard changes
foo-controller-a
foo-cache
labelSelector:
  shard: foo-controller-a
foo-controller-b
foo-cache
labelSelector:
  shard: foo-controller-b
foo-controller-c
foo-cache
labelSelector:
  shard: foo-controller-c
maintain
maintain
maintain
watch
Lease
name: foo-controller-a
holder: foo-controller-a
labels:
  controllerring: foo-controller
Lease
name: foo-controller-b
holder: foo-controller-b
labels:
  controllerring: foo-controller
Lease
name: foo-controller-c
holder: foo-controller-c
labels:
  controllerring: foo-controller

Design


ControllerRing foo-controller
resources:
- group: foo-controller.k8s.io
  resource: foos
reference
select
assign
(via label
reconcile
sharder
consistent hash ring
sharder webhook
add label to unassigned objects
sharder controller
rebalance assignments on shard changes
watch
Foo objects
labels:
  shard: foo-controller-a
labels:
  shard: foo-controller-b
labels:
  shard: foo-controller-c
watch,
reconcile
watch,
reconcile
watch,
reconcile
foo-controller-a
foo-cache
labelSelector:
  shard: foo-controller-a
foo-controller-b
foo-cache
labelSelector:
  shard: foo-controller-b
foo-controller-c
foo-cache
labelSelector:
  shard: foo-controller-c
maintain
maintain
maintain
Lease
name: foo-controller-a
holder: foo-controller-a
labels:
  controllerring: foo-controller
Lease
name: foo-controller-b
holder: foo-controller-b
labels:
  controllerring: foo-controller
Lease
name: foo-controller-c
holder: foo-controller-c
labels:
  controllerring: foo-controller

Design



ControllerRing foo-controller

```
resources:
- group: foo-controller.k8s.io
  resource: foos
```

Foo objects

```
labels:
  shard: foo-controller-a
```

```
labels:
  shard: foo-controller-b
```

```
labels:
  shard: foo-controller-c
```

sharder

consistent hash ring

sharder webhook
add label to unassigned objects

sharder controller
rebalance assignments on shard changes

foo-controller-a

foo-cache

```
labelSelector:
  shard: foo-controller-a
```

foo-controller-b

foo-cache

```
labelSelector:
  shard: foo-controller-b
```

foo-controller-c

foo-cache

```
labelSelector:
  shard: foo-controller-c
```

Lease

```
name: foo-controller-a
holder: foo-controller-a
labels:
  controllerring: foo-controller
```

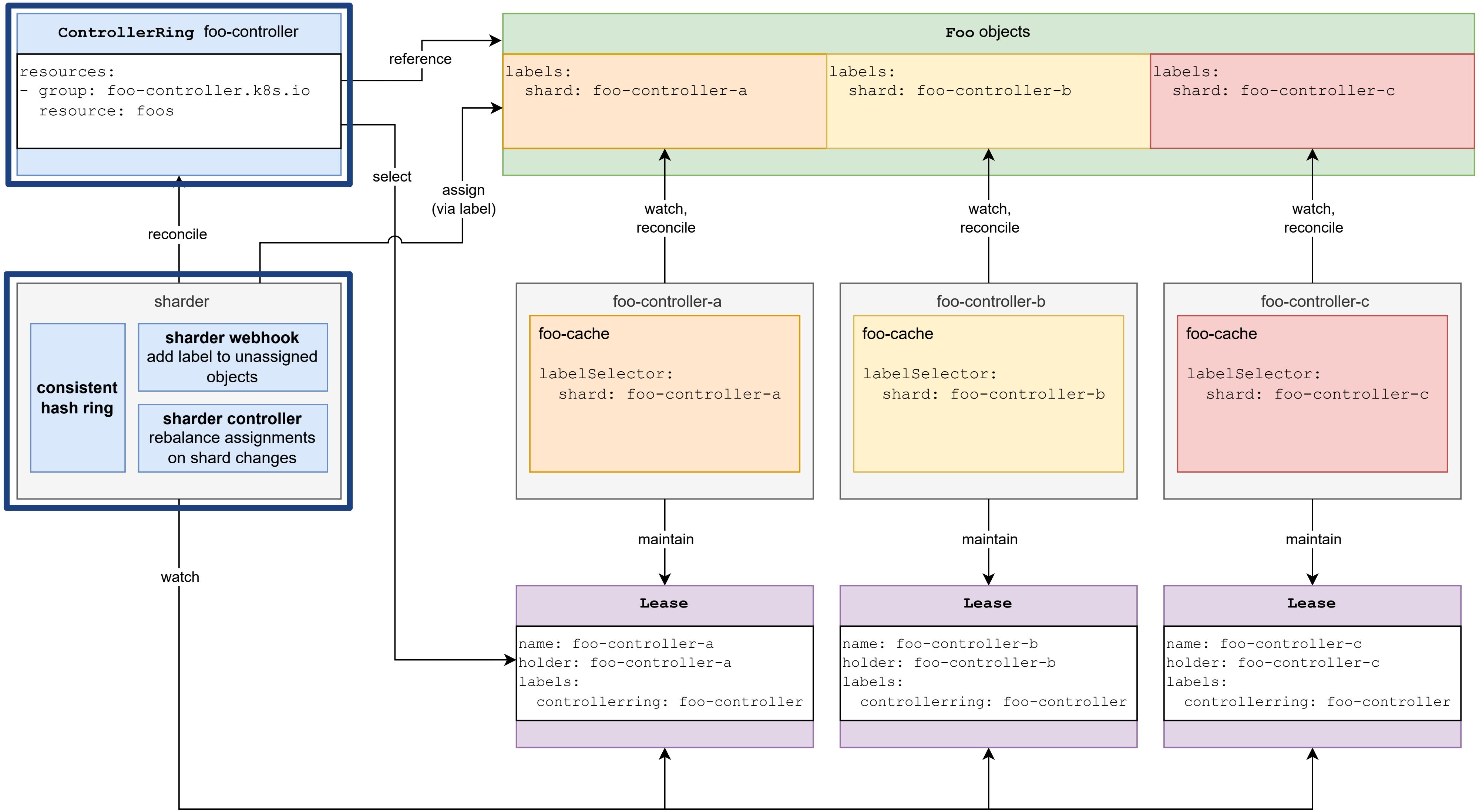
Lease

```
name: foo-controller-b
holder: foo-controller-b
labels:
  controllerring: foo-controller
```

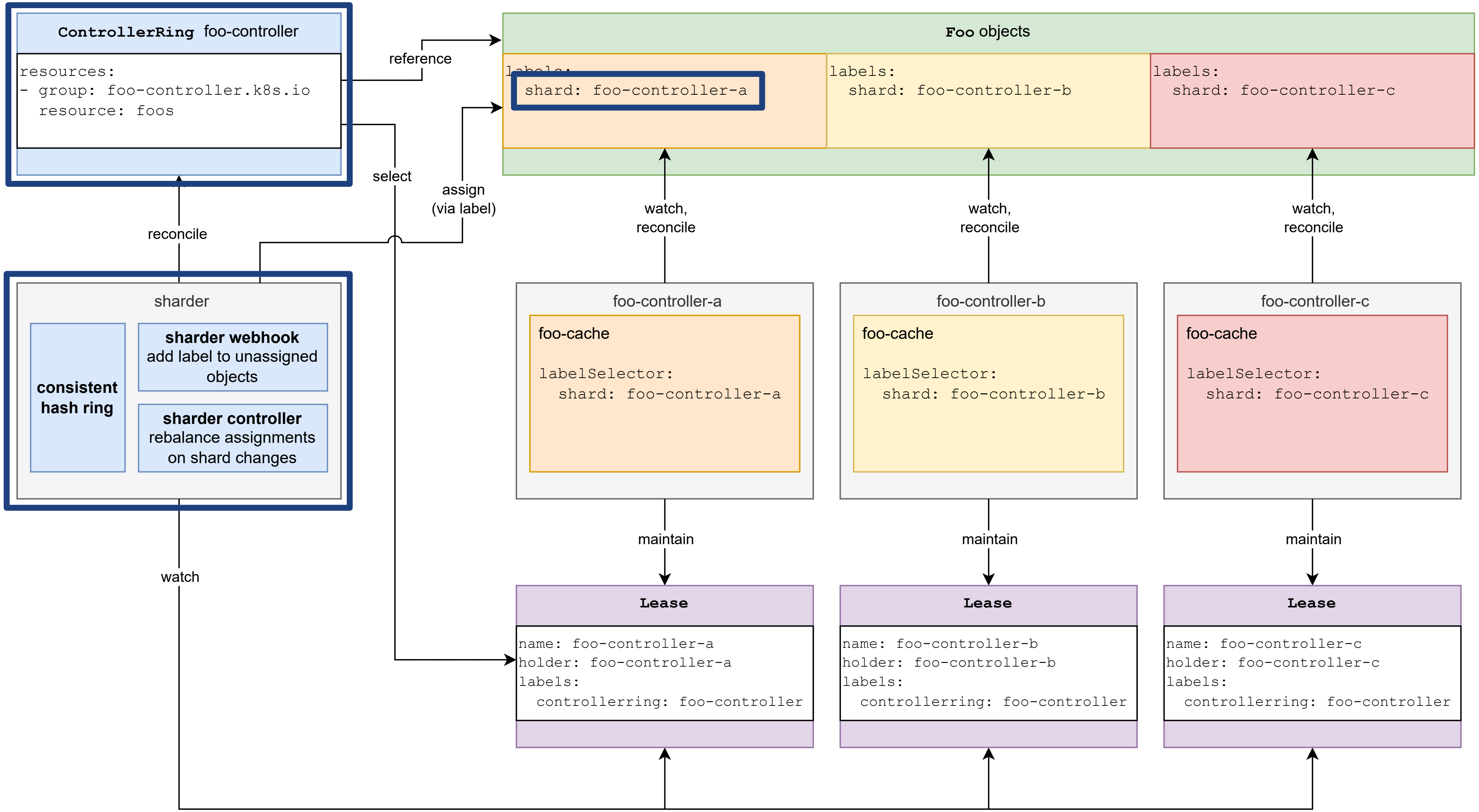
Lease

```
name: foo-controller-c
holder: foo-controller-c
labels:
  controllerring: foo-controller
```

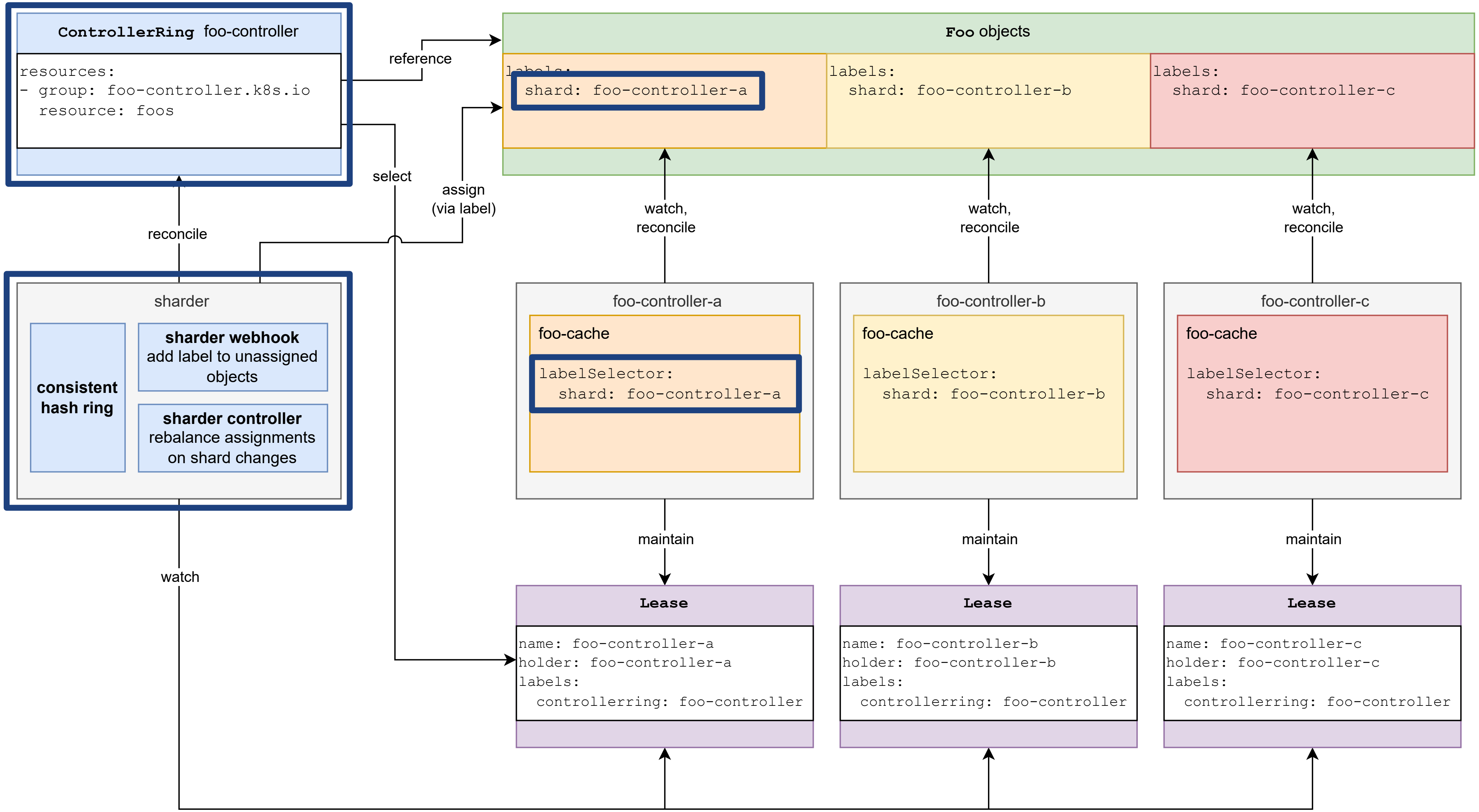
Design


ControllerRing foo-controller
resources:
- group: foo-controller.k8s.io
  resource: foos
reference
select
assign (via label)
reconcile
Foo objects
labels:
  shard: foo-controller-a
labels:
  shard: foo-controller-b
labels:
  shard: foo-controller-c
watch, reconcile
watch, reconcile
watch, reconcile
sharder
consistent hash ring
sharder webhook
add label to unassigned objects
sharder controller
rebalance assignments on shard changes
foo-controller-a
foo-cache
labelSelector:
  shard: foo-controller-a
foo-controller-b
foo-cache
labelSelector:
  shard: foo-controller-b
foo-controller-c
foo-cache
labelSelector:
  shard: foo-controller-c
maintain
maintain
maintain
watch
Lease
name: foo-controller-a
holder: foo-controller-a
labels:
  controllerring: foo-controller
Lease
name: foo-controller-b
holder: foo-controller-b
labels:
  controllerring: foo-controller
Lease
name: foo-controller-c
holder: foo-controller-c
labels:
  controllerring: foo-controller

Design


ControllerRing foo-controller
resources:
- group: foo-controller.k8s.io
  resource: foos
reference
select
assign (via label)
reconcile
Foo objects
labels:
shard: foo-controller-a
labels:
shard: foo-controller-b
labels:
shard: foo-controller-c
watch, reconcile
watch, reconcile
watch, reconcile
sharder
consistent hash ring
sharder webhook
add label to unassigned objects
sharder controller
rebalance assignments on shard changes
foo-controller-a
foo-cache
labelSelector:
shard: foo-controller-a
foo-controller-b
foo-cache
labelSelector:
shard: foo-controller-b
foo-controller-c
foo-cache
labelSelector:
shard: foo-controller-c
maintain
maintain
maintain
watch
Lease
name: foo-controller-a
holder: foo-controller-a
labels:
controllerring: foo-controller
Lease
name: foo-controller-b
holder: foo-controller-b
labels:
controllerring: foo-controller
Lease
name: foo-controller-c
holder: foo-controller-c
labels:
controllerring: foo-controller

Design


ControllerRing foo-controller
resources:
- group: foo-controller.k8s.io
  resource: foos
reference
select
assign
(via label)
reconcile
Foo objects
labels:
  shard: foo-controller-a
labels:
  shard: foo-controller-b
labels:
  shard: foo-controller-c
watch,
reconcile
watch,
reconcile
watch,
reconcile
sharder
consistent
hash ring
sharder webhook
add label to unassigned objects
sharder controller
rebalance assignments on shard changes
foo-controller-a
foo-cache
labelSelector:
  shard: foo-controller-a
foo-controller-b
foo-cache
labelSelector:
  shard: foo-controller-b
foo-controller-c
foo-cache
labelSelector:
  shard: foo-controller-c
maintain
maintain
maintain
watch
Lease
name: foo-controller-a
holder: foo-controller-a
labels:
  controllerring: foo-controller
Lease
name: foo-controller-b
holder: foo-controller-b
labels:
  controllerring: foo-controller
Lease
name: foo-controller-c
holder: foo-controller-c
labels:
  controllerring: foo-controller

Demo

Implementation

Implementation – 1) Install Components

```
kubectl apply --server-side –k
 "https://github.com/timebertt/kubernetes-controller-
  sharding//config/default?ref=main"
```

Implementation – 2) ControllerRing

```yaml
apiVersion: sharding.timebertt.dev/v1alpha1
kind: ControllerRing
metadata:
  name: webhosting-operator
spec:
  resources:
  - group: webhosting.timebertt.dev
    resource: websites
    controlledResources:
    - group: apps
      resource: deployments
    - group: networking.k8s.io
      resource: ingresses
    # ...
```

Implementation – 3) Shard Lease

```yaml
apiVersion: coordination.k8s.io/v1
kind: Lease
metadata:
  labels:
    alpha.sharding.timebertt.dev/controllerring: webhosting-operator
  name: webhosting-operator-65ffcdb674-8sst9
  namespace: webhosting-system
spec:
  holderIdentity: webhosting-operator-65ffcdb674-8sst9
  acquireTime: "2025-04-03T10:45:51.992779Z"
  renewTime: "2025-04-03T11:02:26.817751Z”
  leaseDurationSeconds: 15
```

Implementation – 3) Shard Lease

```
apiVersion: coordination.k8s.io/v1
kind: Lease
metadata:
  labels:
    alpha.sharding.timebertt.dev/controllerring: webhosting-operator
  name: webhosting-operator-65ffcdb674-8sst9
  namespace: webhosting-system
spec:
  holderIdentity: webhosting-operator-65ffcdb674-8sst9
  acquireTime: "2025-04-03T10:45:51.992779Z"
  renewTime: "2025-04-03T11:02:26.817751Z”
  leaseDurationSeconds: 15
```

Implementation – 3) Shard Lease

```yaml
apiVersion: coordination.k8s.io/v1
kind: Lease
metadata:
  labels:
    alpha.sharding.timebertt.dev/controllerring: webhosting-operator
  name: webhosting-operator-65ffcdb674-8sst9
  namespace: webhosting-system
spec:
  holderIdentity: webhosting-operator-65ffcdb674-8sst9
  acquireTime: "2025-04-03T10:45:51.992779Z"
  renewTime: "2025-04-03T11:02:26.817751Z”
  leaseDurationSeconds: 15
```

Implementation – 3) Shard Lease

```
shardLease, err := shardlease.NewResourceLock(restConfig, shardlease.Options{
  ControllerRingName: "webhosting-operator",
})
if err != nil {
  return err
}

mgr, err := manager.New(restConfig, manager.Options{
  LeaderElection:                      true,
  LeaderElectionResourceLockInterface: shardLease,
  LeaderElectionReleaseOnCancel:       true,
  // ...
})
if err != nil {
  return err
}
```

Implementation – 3) Shard Lease

```go
shardLease, err := shardlease.NewResourceLock(restConfig, shardlease.Options{
  ControllerRingName: "webhosting-operator",
})
if err != nil {
  return err
}

mgr, err := manager.New(restConfig, manager.Options{
  LeaderElection:                      true,
  LeaderElectionResourceLockInterface: shardLease,
  LeaderElectionReleaseOnCancel:       true,
  // ...
})
if err != nil {
  return err
}
```

Implementation – 3) Shard Lease

```go
shardLease, err := shardlease.NewResourceLock(restConfig, shardlease.Options{
  ControllerRingName: "webhosting-operator",
})
if err != nil {
  return err
}

mgr, err := manager.New(restConfig, manager.Options{
  LeaderElection:                      true,
  LeaderElectionResourceLockInterface: shardLease,
  LeaderElectionReleaseOnCancel:       true,
  // ...
})
if err != nil {
  return err
}
```

Implementation – 4) Filter Watch Cache

Only watch and reconcile objects with this label:

```
shard.alpha.sharding.timebertt.dev/webhosting-operator=<shard-name>
```

Implementation – 4) Filter Watch Cache

```go
labelSelector := labels.SelectorFromSet(labels.Set{
  // shard.alpha.sharding.timebertt.dev/webhosting-operator=<shard-name>
  shardingv1alpha1.LabelShard("webhosting-operator"): shardLease.Identity(),
})

mgr, err := manager.New(restConfig, manager.Options{
  Cache: cache.Options{
    DefaultLabelSelector: labelSelector,
  },
})
if err != nil {
  return err
}
```

Implementation – 4) Filter Watch Cache

```go
labelSelector := labels.SelectorFromSet(labels.Set{
  // shard.alpha.sharding.timebertt.dev/webhosting-operator=<shard-name>
  shardingv1alpha1.LabelShard("webhosting-operator"): shardLease.Identity(),
})

mgr, err := manager.New(restConfig, manager.Options{
  Cache: cache.Options{
    DefaultLabelSelector: labelSelector,
  },
})
if err != nil {
  return err
}
```

Implementation – 5) Drain Operations

Stop reconciling objects with this label and remove the labels:

```
drain.alpha.sharding.timebertt.dev/webhosting-operator=true
```

Implementation – 5) Drain Operations

```
err := builder.ControllerManagedBy(mgr).
    For(&webhostingv1alpha1.Website{}, builder.WithPredicates(
        WebsitePredicate(),
    )).
    Owns(&appsv1.Deployment{}, builder.WithPredicates(DeploymentPredicate())).
    Complete(
        reconciler,
    )
```

Implementation – 5) Drain Operations

```
err := builder.ControllerManagedBy(mgr).
  For(&webhostingv1alpha1.Website{}, builder.WithPredicates(
    WebsitePredicate(),
  )).
  Owns(&appsv1.Deployment{}, builder.WithPredicates(DeploymentPredicate())).
  Complete(
    reconciler,
  )
```

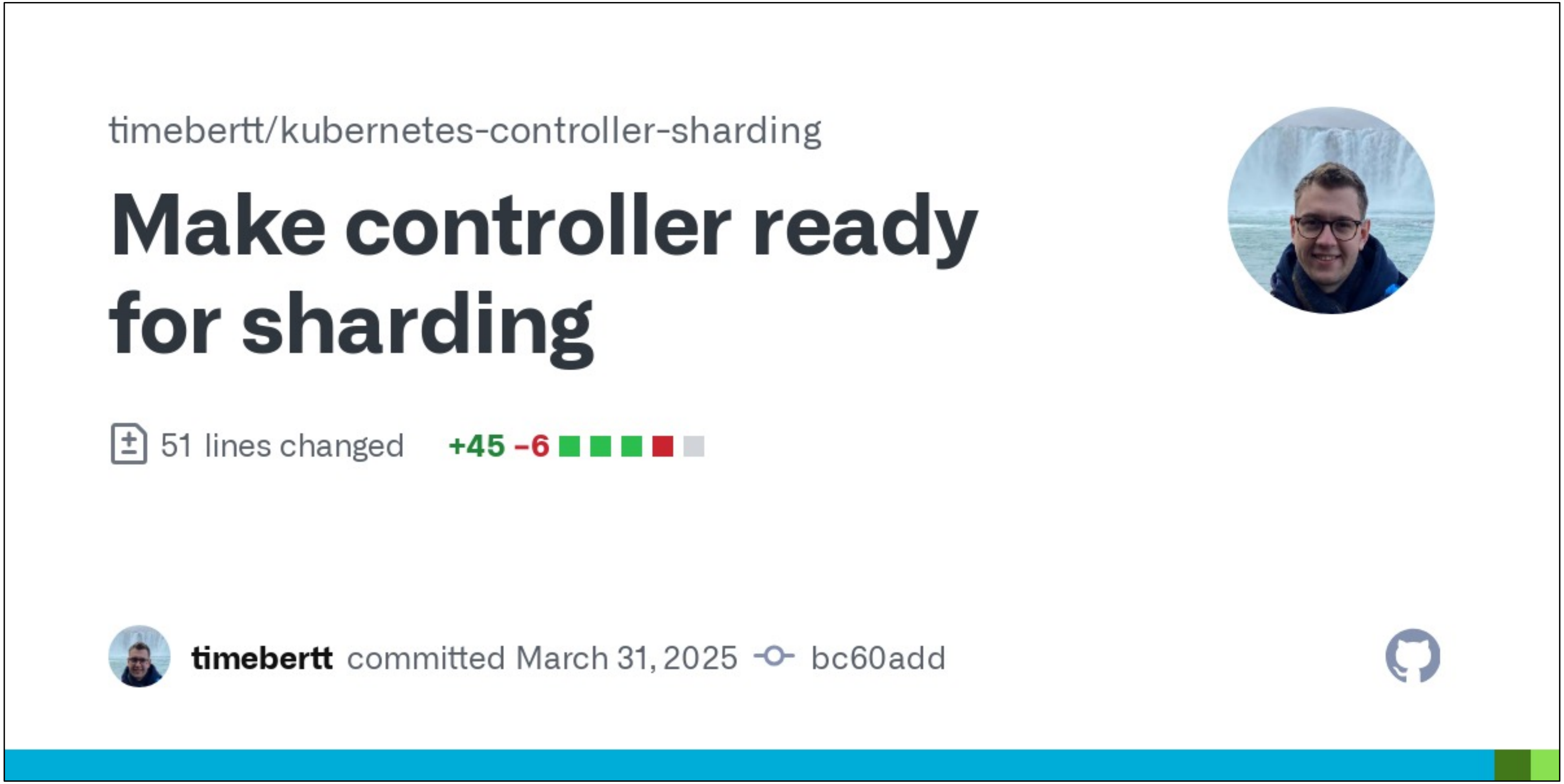
Implementation – 5) Drain Operations

```go
err := builder.ControllerManagedBy(mgr).
  For(&webhostingv1alpha1.Website{}, builder.WithPredicates(
    shardcontroller.Predicate(controllerRing, shardName, WebsitePredicate()),
  )).
  Owns(&appsv1.Deployment{}, builder.WithPredicates(DeploymentPredicate())).
  Complete(
    reconciler,
  )
```

Implementation – 5) Drain Operations

```
err := builder.ControllerManagedBy(mgr).
  For(&webhostingv1alpha1.Website{}, builder.WithPredicates(
    shardcontroller.Predicate(controllerRing, shardName, WebsitePredicate()),
  )).
  Owns(&appsv1.Deployment{}, builder.WithPredicates(DeploymentPredicate())).
  Complete(
    shardcontroller.NewShardedReconciler(mgr).
      For(&webhostingv1alpha1.Website{}).
      InControllerRing(controllerRing).
      WithShardName(shardName).
      MustBuild(reconciler),
  )
```

Implementation – Summary



timebertt/kubernetes-controller-sharding

Make controller ready for sharding

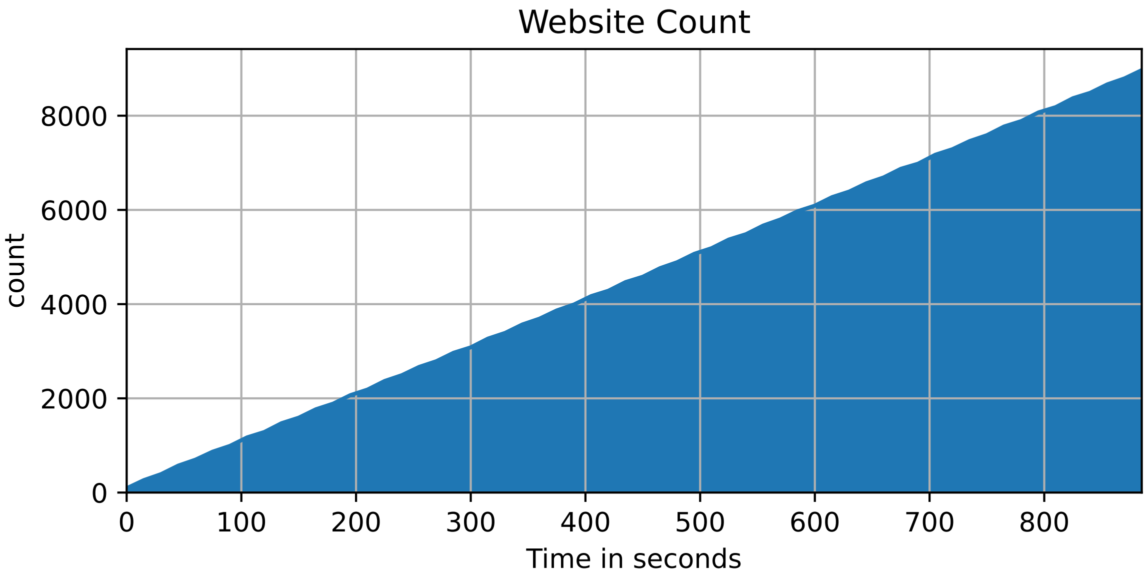
51 lines changed +45 −6 🟩🟩🟩🟥⬜

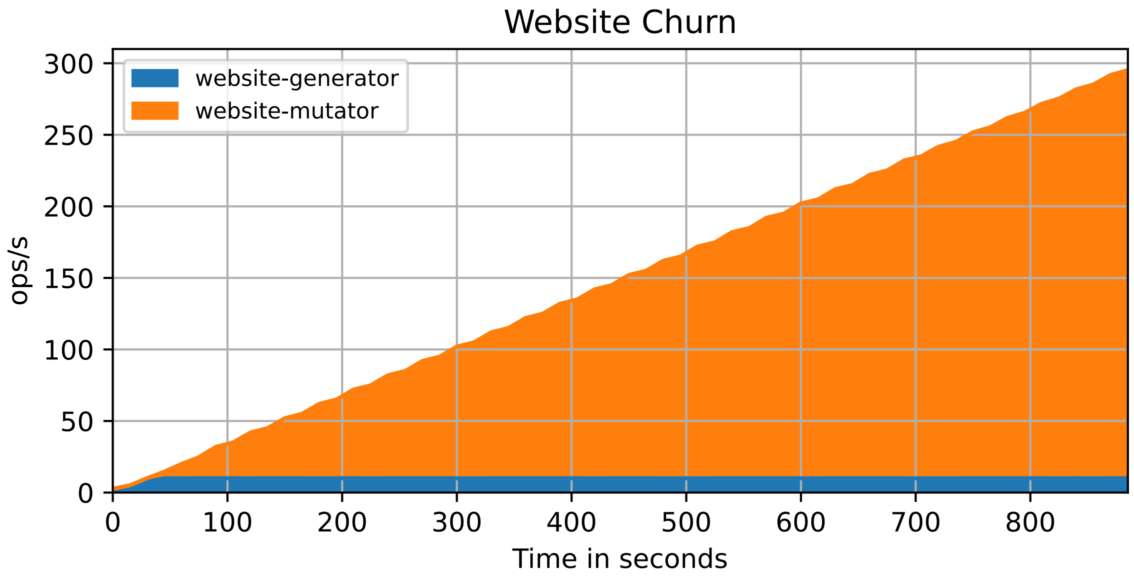
timebertt committed March 31, 2025 bc60add

Evaluation

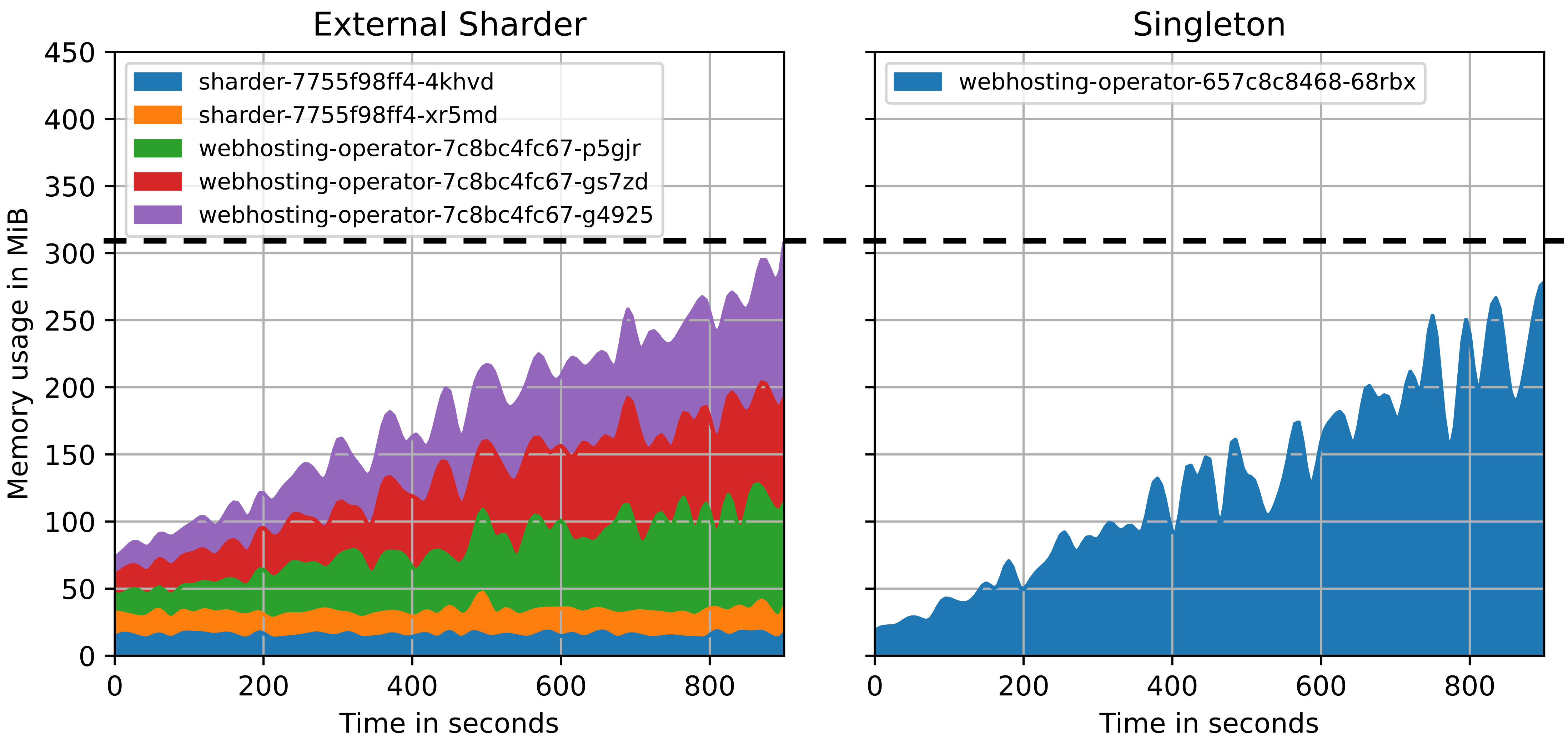
Evaluation – Load Tests

- Dimension 1: object count – up to 9.000 objects
- Dimension 2: object churn – up to 300 changes per second





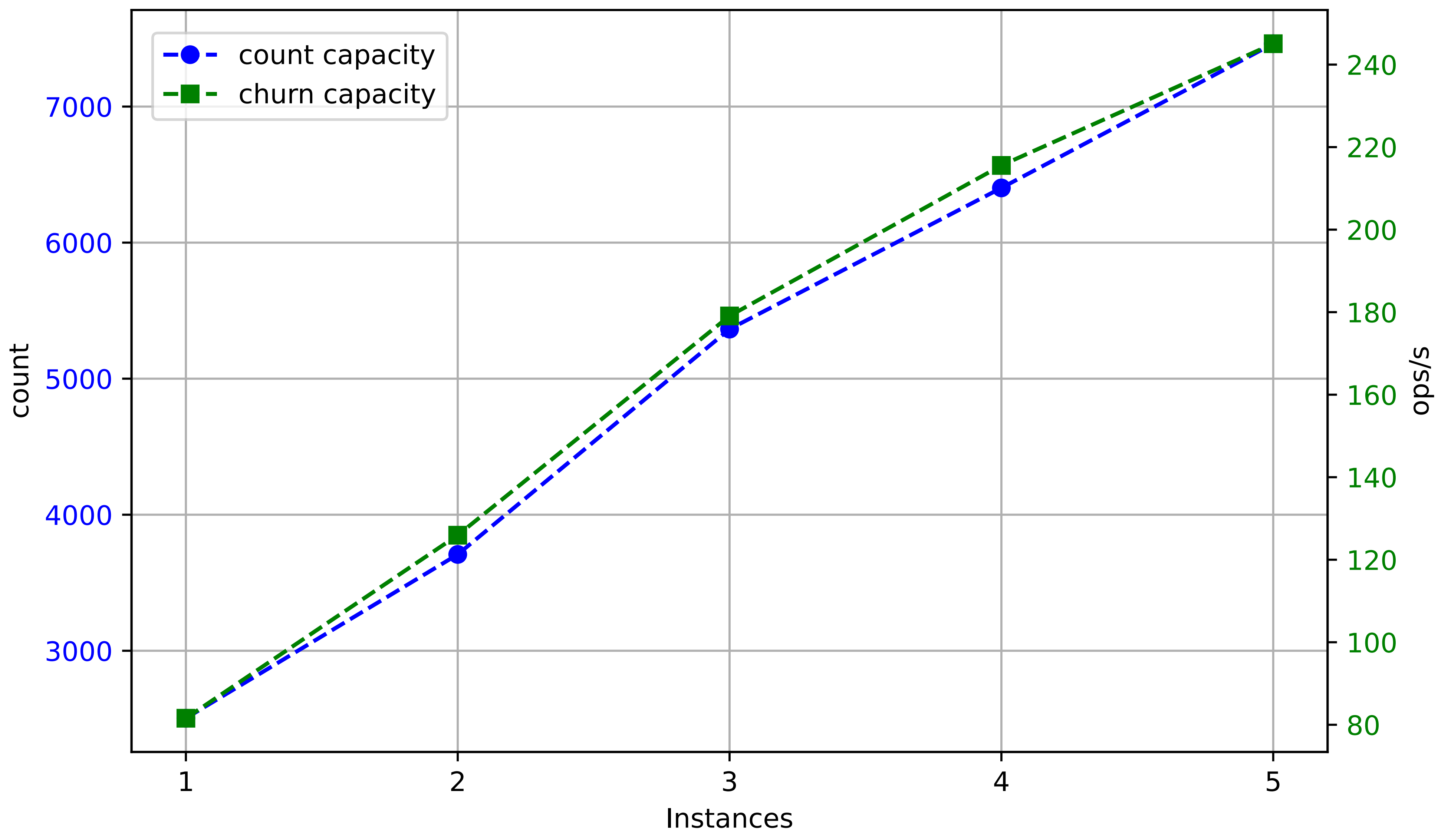
Evaluation – Resource Usage



Evaluation – Performance Measurements

```yaml
queries:
# SLO 1: Queue Latency: p99 < 1s
- name: latency-queue
  query: |
    histogram_quantile(0.99, sum by (le) (
      workqueue_queue_duration_seconds_bucket{
        job="webhosting-operator", name="website"
      }
    ))
# SLO 2: Reconciliation Latency: p99 < 5s
- name: latency-reconciliation
  query: |
    histogram_quantile(0.99, sum by (le) (
      experiment_website_reconciliation_duration_seconds_bucket{
        job="experiment"
      }
    ))
```

Evaluation – Load Capacity



Conclusion

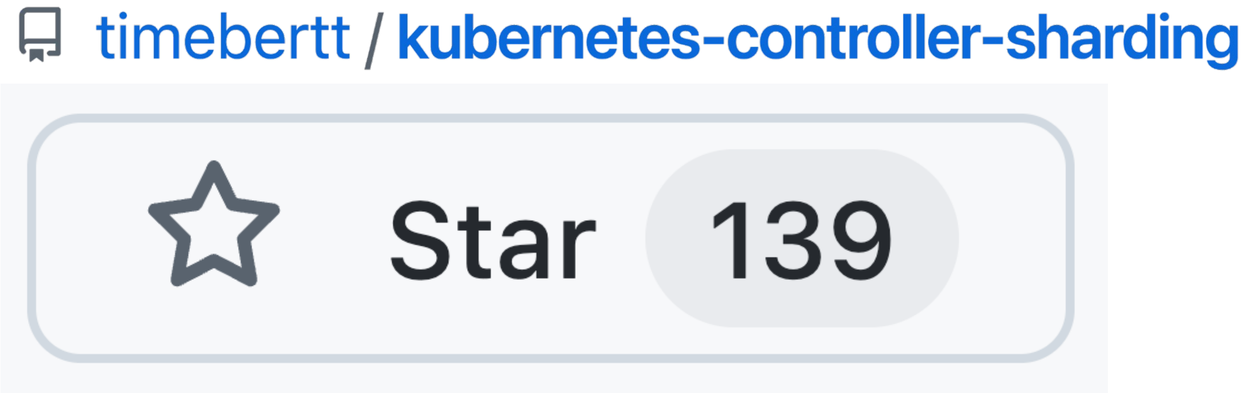
Conclusion

- Makes Kubernetes controllers horizontally scalable
- Capacity increases almost linearly with every added instance
- Design applicable to arbitrary controllers
- Implementation simple to reuse
- Ready for usage and collaboration in the open source community
- Gather experience in real-world scenarios

Questions?

Questions?

Find my project + Master’s thesis on GitHub!



timebertt / **kubernetes-controller-sharding**

Tim Ebert, STACKIT

timebertt@gmail.com

Feedback welcome!

https://sched.co/1txFG

KubeCon | CloudNativeCon

Europe 2025