Cloud Native Solution Development With the Cloud Pak for Applications

Chris Bailey
Senior Technical Staff Member
Cross-Cloud Pak Applications and Accelerators

Twitter: @Chris__Bailey
Email: baileyc@uk.ibm.com

Code ⇄ think

IBM

Cloud Native Development

Building Apps for Cloud-Native Deployments

- ✓ Follow 12 factor principles
- ✓ Provide Liveness and Readiness Checks
- ✓ Provide Metrics and Request Tracking for Observability
- ✓ Package as a Container Image
- ✓ Configure for Kubernetes
- ✓ Deploy, Scale and Manage on OpenShift

Cloud Solution Development

Going Beyond a Single Cloud-Native Microservice

- ✓ Multi-microservice solutions
- ✓ Service discovery and binding
- ✓ Application level configuration
- ✓ Change control and multi-environment support
- ✓ Application level metrics, monitoring and day-2 operations

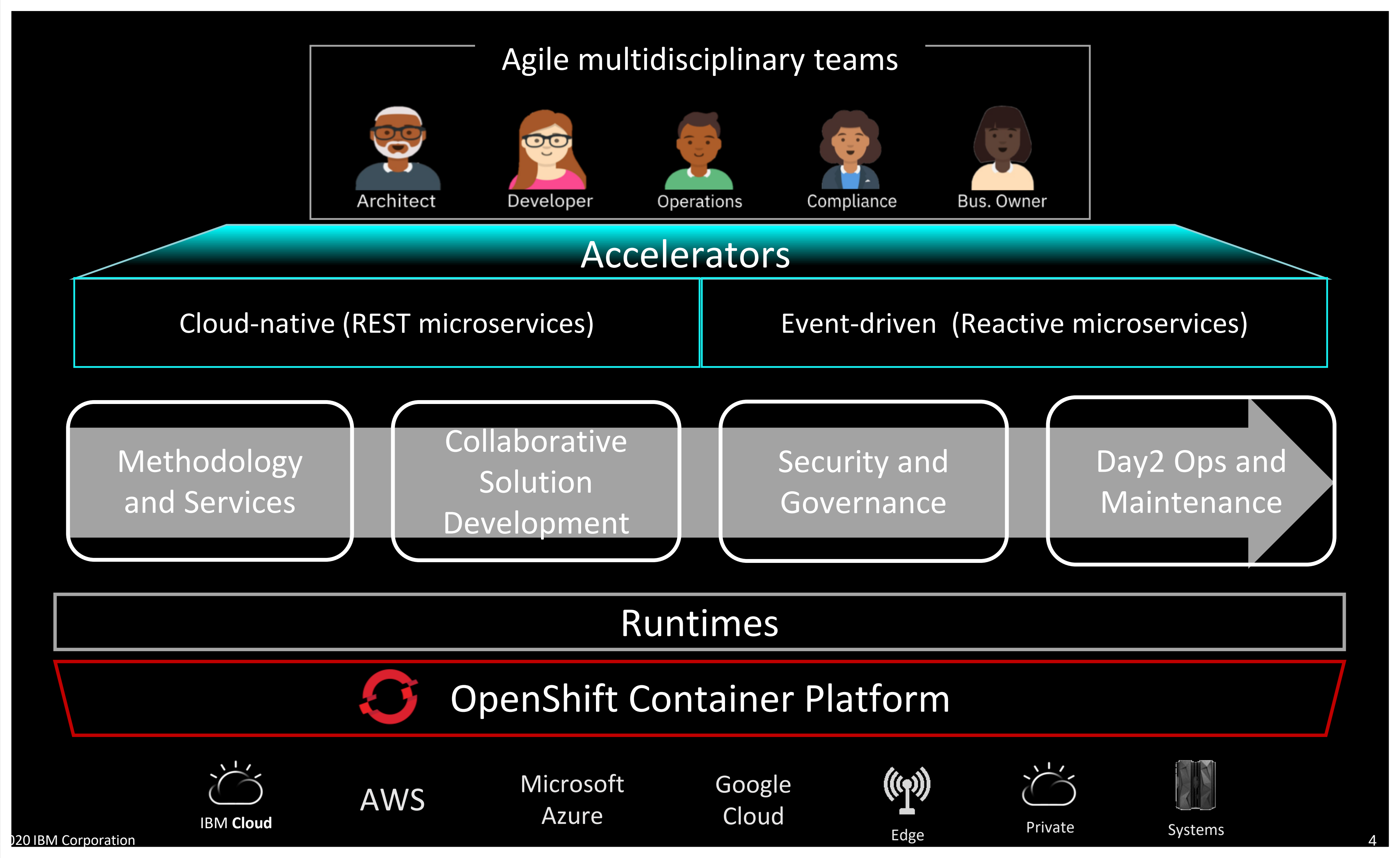
Accelerators for Cloud-Native and Event-Driven Solutions *(Tech Preview)*

Accelerating Cloud-native **solutions** faster from idea to production

Unleash

Unlock

Optimize



Agile multidisciplinary teams

Accelerators

Cloud-native (REST microservices)

Event-driven (Reactive microservices)

Methodology and Services

Collaborative Solution Development

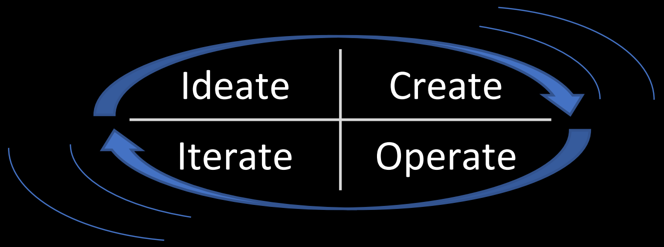
Security and Governance

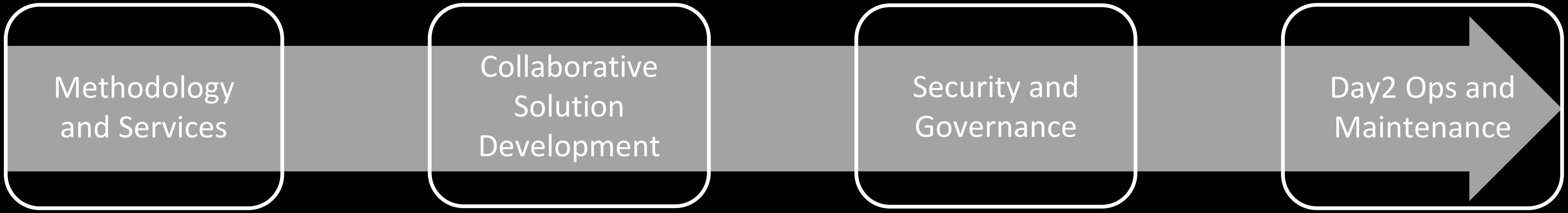
Day2 Ops and Maintenance

Runtimes

OpenShift Container Platform

How are teams are accelerated?


Ideate
Create
Iterate
Operate


Methodology and Services
Collaborative Solution Development
Security and Governance
Day2 Ops and Maintenance

Approach business problems differently

Methodology and Services

Learn using reference architectures

Apply agile principles

Co-create with Garage expertise & services

Teams productive in hours not weeks

Collaborative Solution Development

Define using reference architectures

Auto-provision projects

Develop iteratively with unified tools

Faster time to value, integrated end-to-end

Security and Governance

Build and deploy with Secure DevOps

Promote applications and services using GitOps

Govern your enterprise choices

Lower cost to operate total solution

Day2 Ops and Maintenance

Observe with unique telemetry

Automate updates to production workload

Operate with Day2 Actions

Value of Accelerators

Accelerating cloud-native solutions empowers, enables, and meets enterprise needs

Empower teams to rapidly innovate

IT Ops and Architects are now able to provide curated, pre-configured stacks and pipelines to empower teams to rapidly innovate

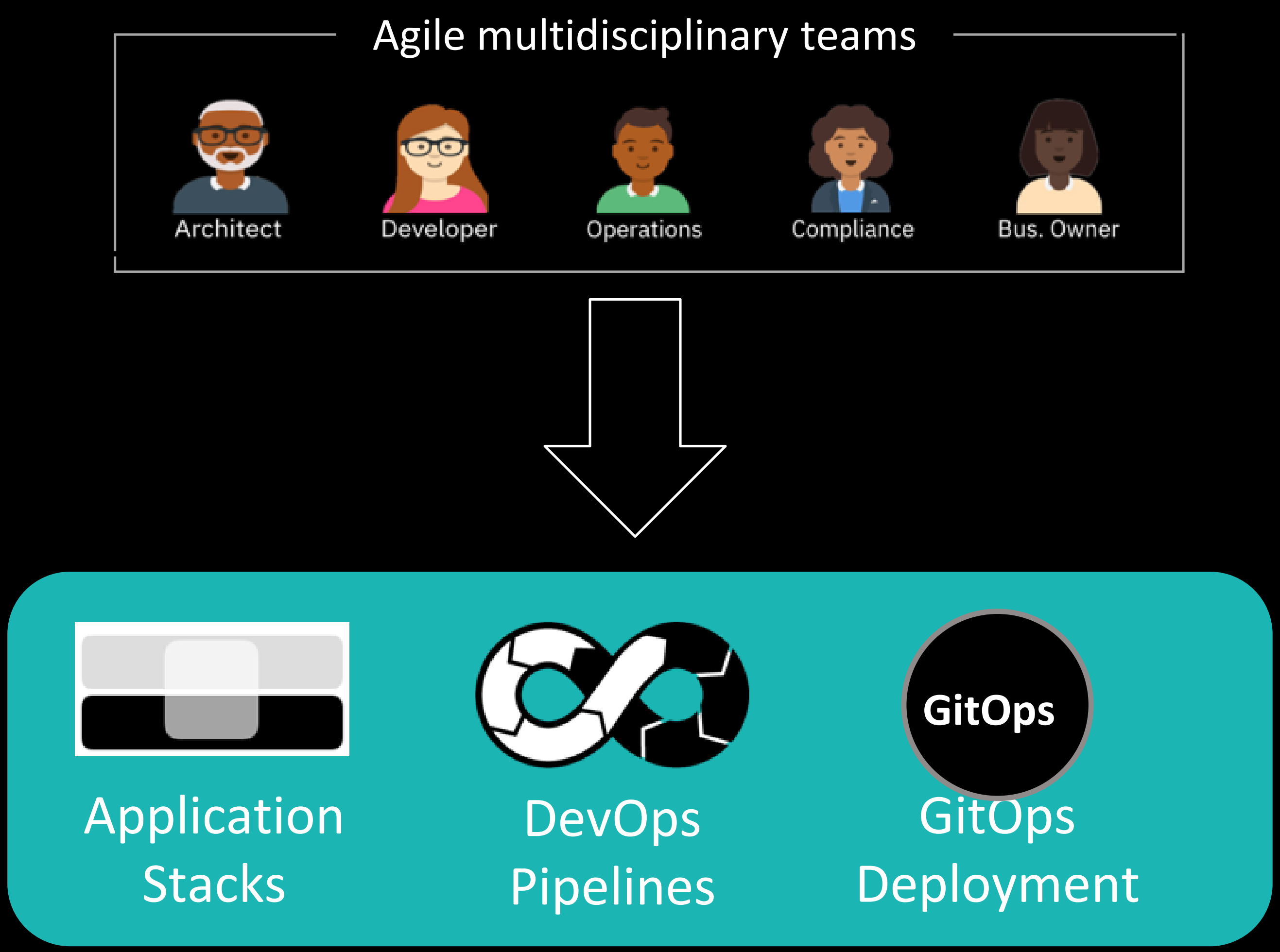
Enable governance to your standards

Simplify how teams comply with your unique operational, security and technology choices

Manage solutions at scale

Manage your solutions from idea to production when massively scaled in containers using integrated DevOps, GitOps and Operational Insights

Cloud Pak for Applications delivers flexibility and consistency at scale



Agile multidisciplinary teams

Architect

Developer

Operations

Compliance

Bus. Owner

GitOps

Application Stacks

DevOps Pipelines

GitOps Deployment

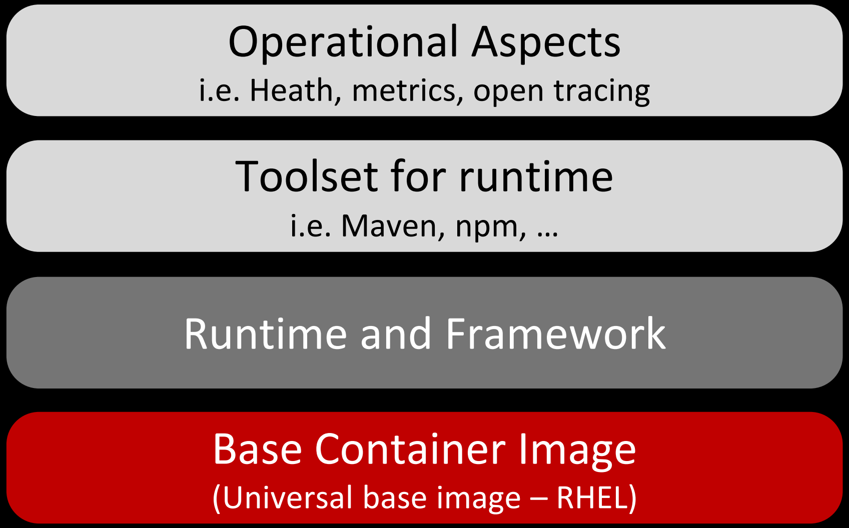
Unique decisions for developing and deploying cloud-native apps rely on collaboration from multidisciplinary teams

Cloud Pak for Applications delivers:

- A flexible open approach to codify and centrally manage your decisions
- Consistency at scale and greater productivity for developers

Application Stacks

- ✓ Container configuration, build framework and deployment manifest
- ✓ Language, Runtime, Frameworks, Libraries included
- ✓ Runtime build and debug tools for iterative development
- ✓ Common operational capabilities (e.g. health, monitoring endpts, tracing)
- ✓ Semantic versioned
- ✓ Labels provide traceability (owner, git origin, versions, timestamps)
- ✓ Customizable: Providing enterprise governance and consistency at scale



Operational Aspects
i.e. Heath, metrics, open tracing

Toolset for runtime
i.e. Maven, npm, …

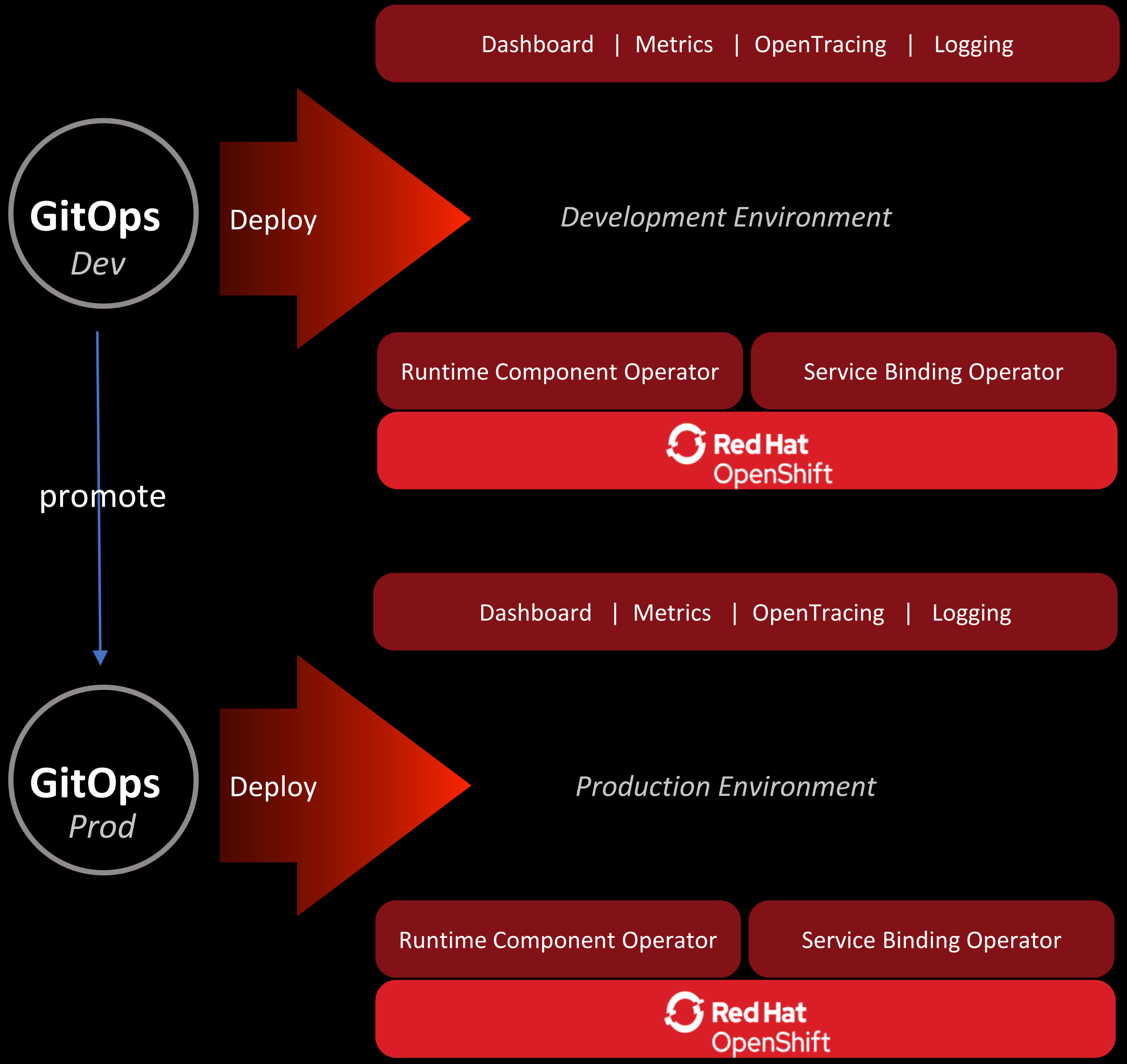
Runtime and Framework

Base Container Image
(Universal base image – RHEL)

GitOps with Kustomize

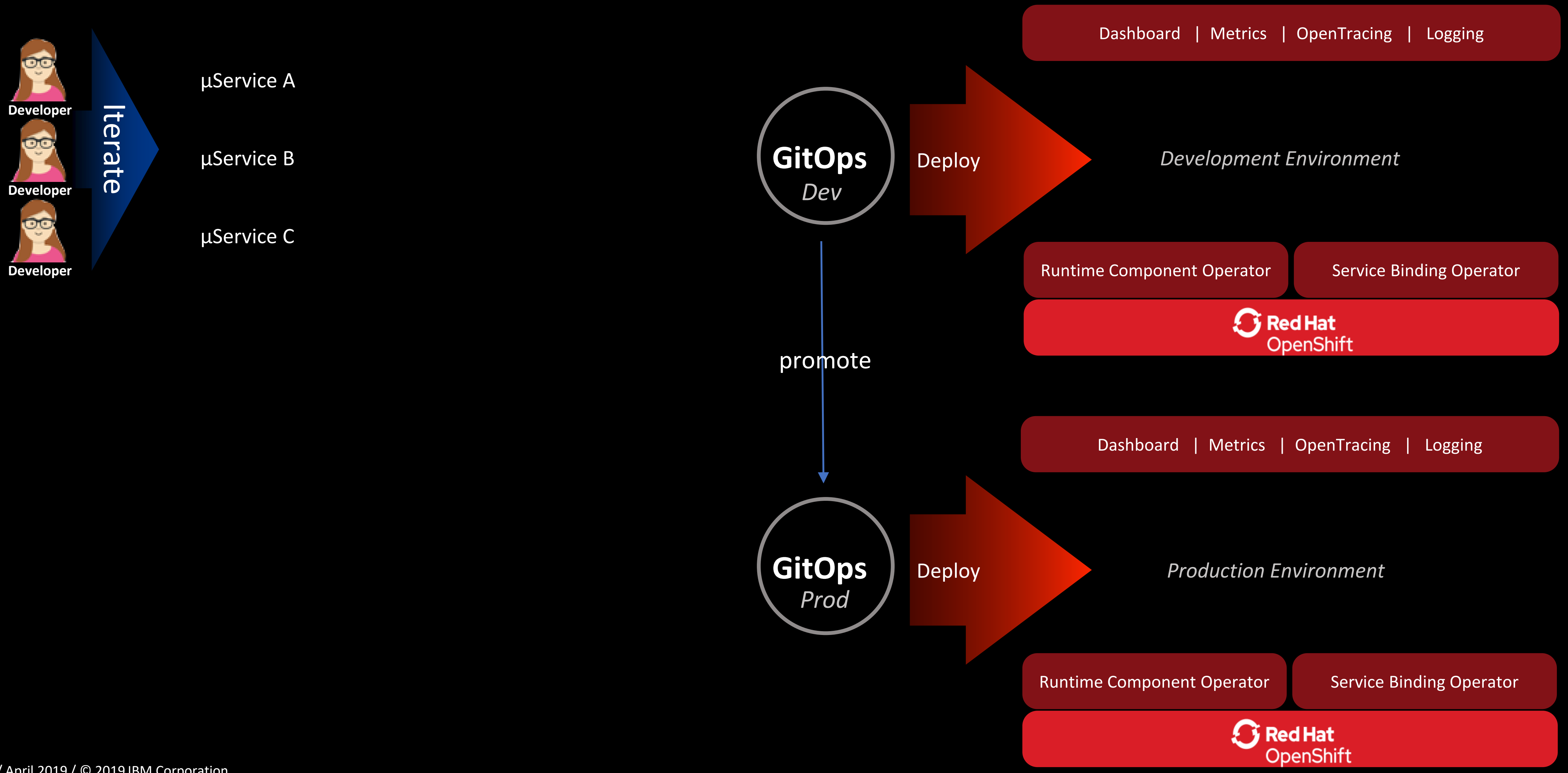
Kubernetes Native Configuration Control and Management

- ✓ Single “Source of Truth” for solution deployments
- ✓ Separation on concerns between Development and Operations
- ✓ Solution-level and per-environment configuration
- ✓ Change control and roll-back mechanisms Deploy
- ✓ Promotion of changes through Dev, Staging and Production



GitOps with Kustomize

Kubernetes Native Configuration Control and Management



Developer

Developer

Developer

Iterate

µService A

µService B

µService C

GitOps *Dev*

Deploy

Dashboard | Metrics | OpenTracing | Logging

Development Environment

Runtime Component Operator

Service Binding Operator

Red Hat OpenShift

promote

GitOps *Prod*

Deploy

Dashboard | Metrics | OpenTracing | Logging

Production Environment

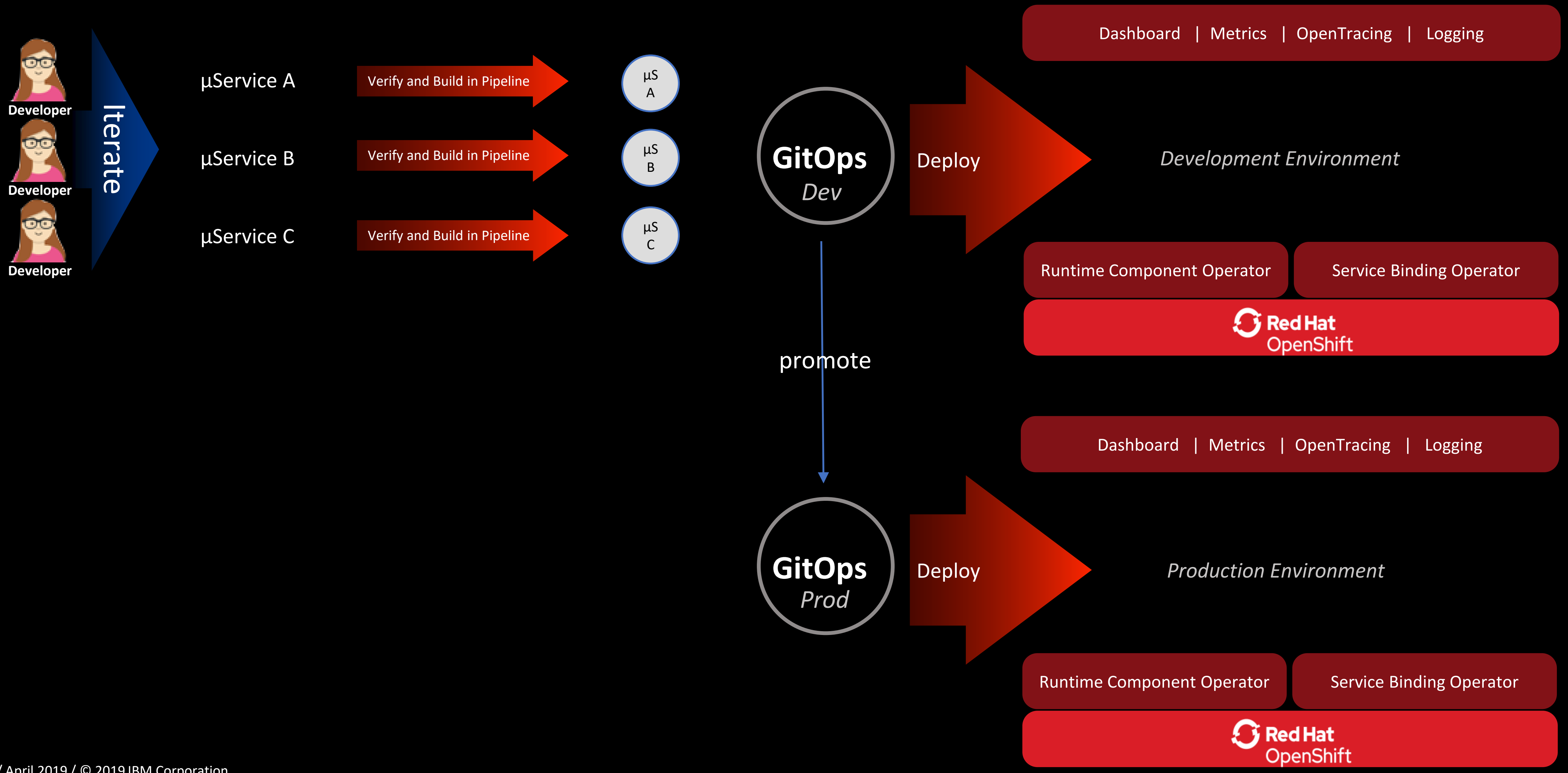
Runtime Component Operator

Service Binding Operator

Red Hat OpenShift

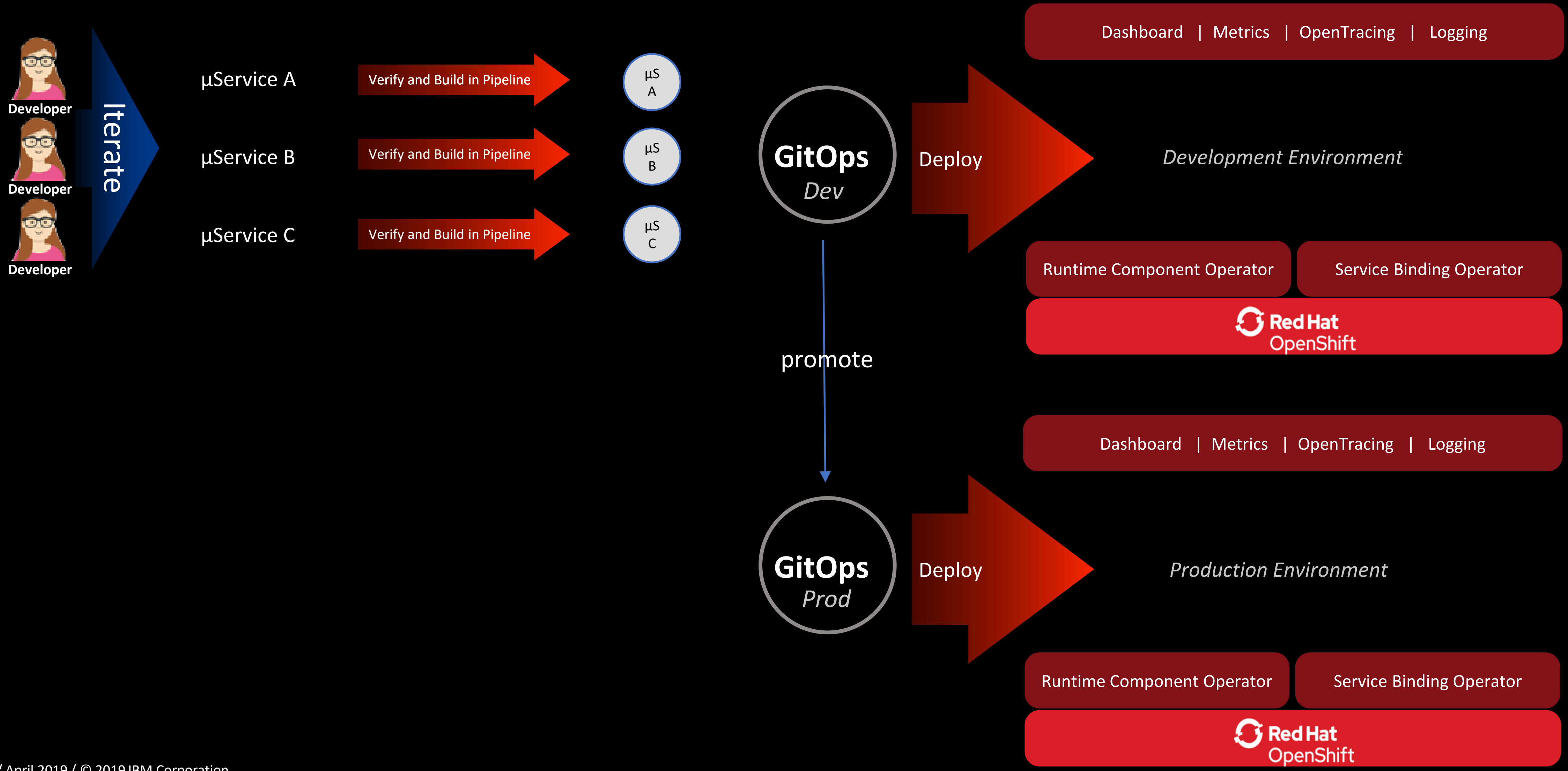
GitOps with Kustomize

Kubernetes Native Configuration Control and Management



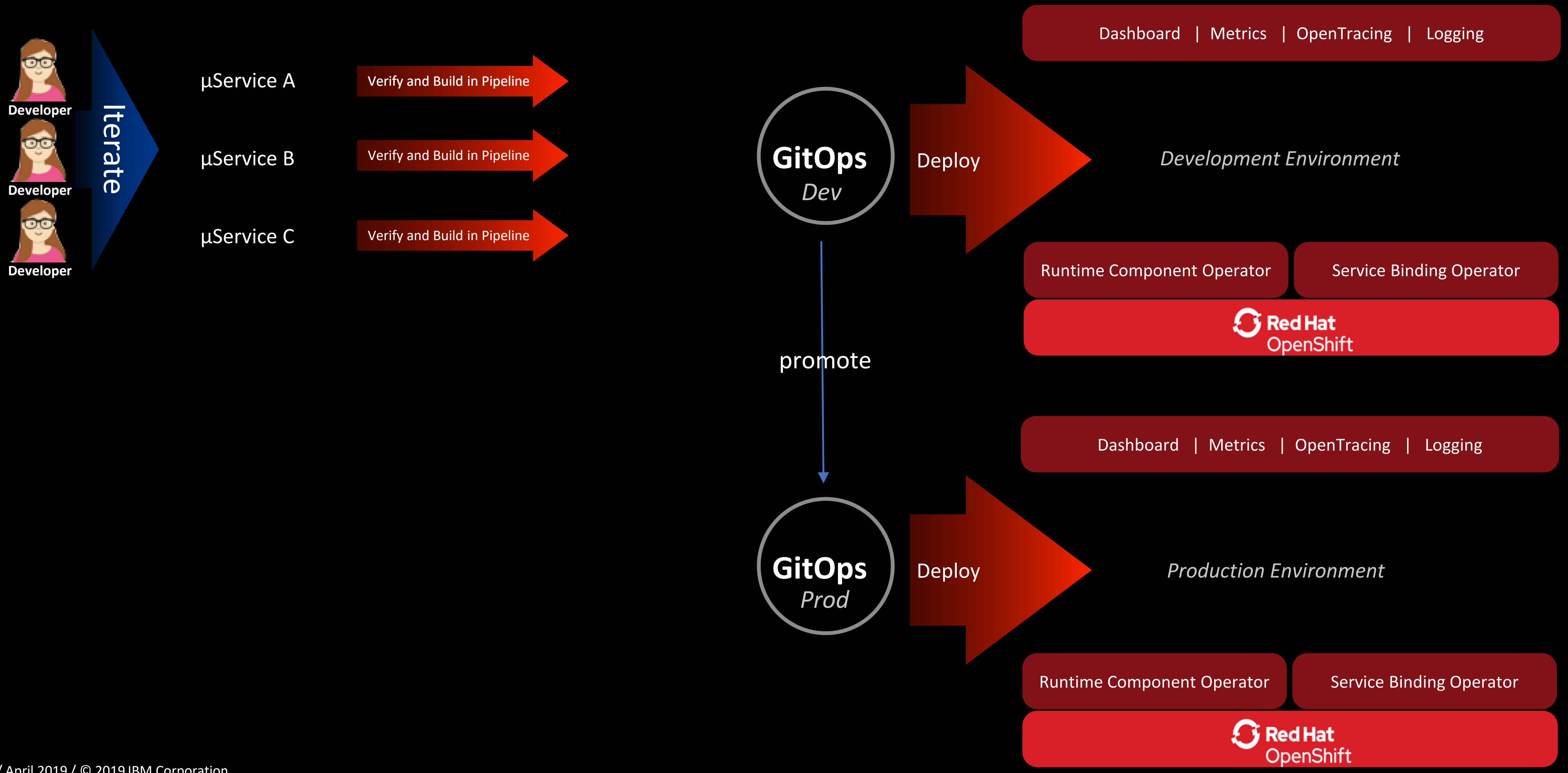
GitOps with Kustomize

Kubernetes Native Configuration Control and Management



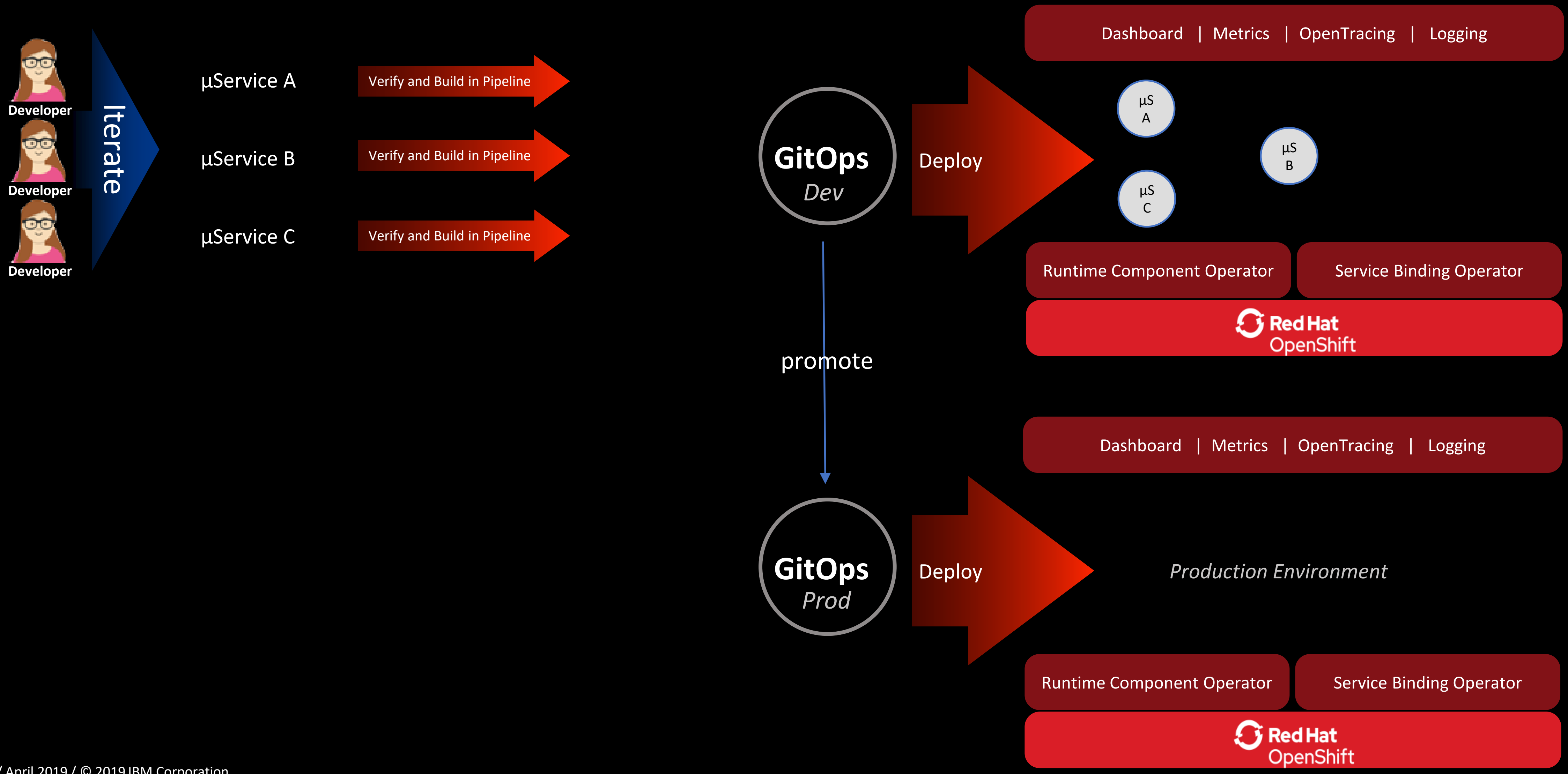
GitOps with Kustomize

Kubernetes Native Configuration Control and Management



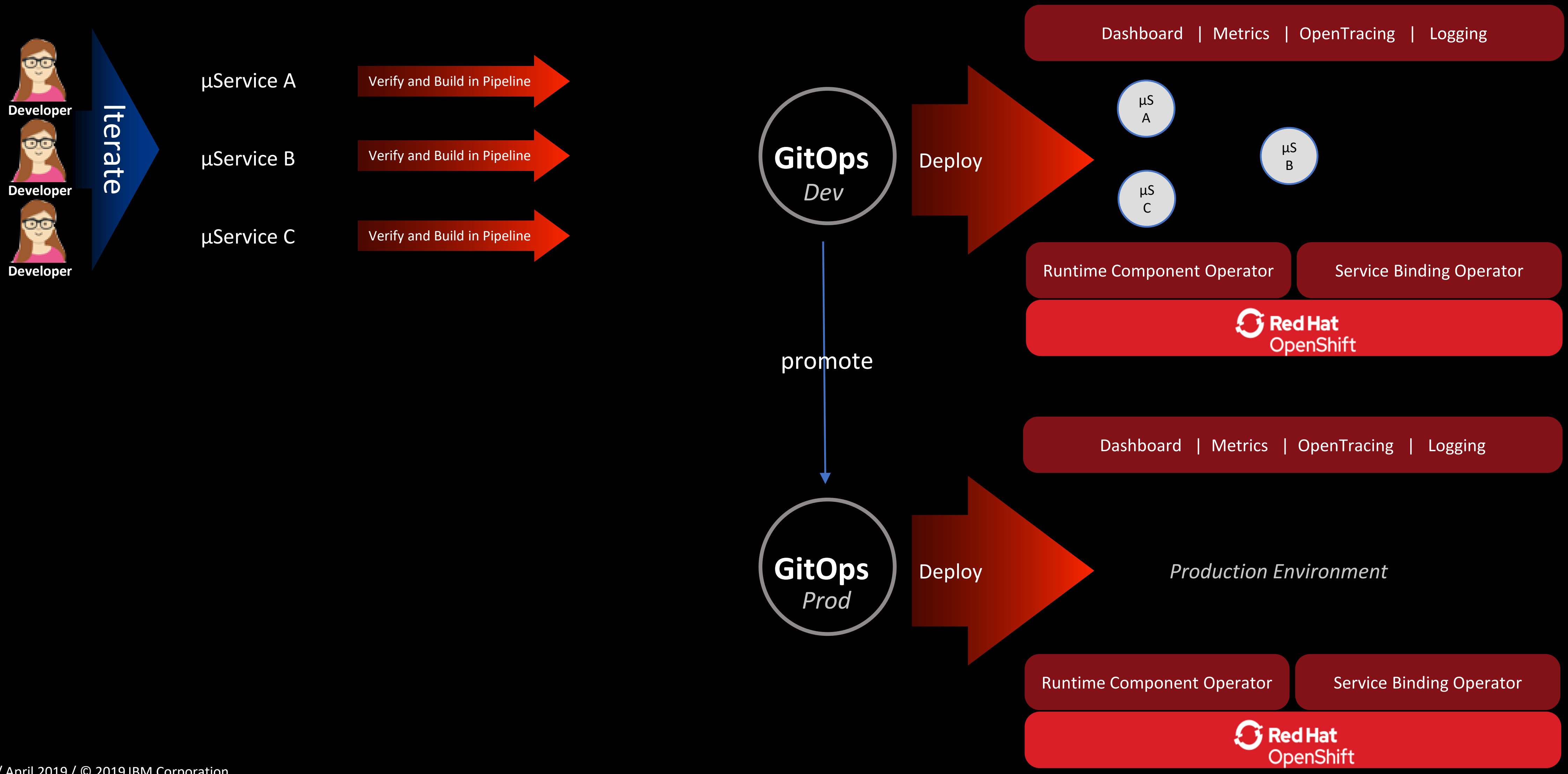
GitOps with Kustomize

Kubernetes Native Configuration Control and Management


Developer
Developer
Developer
Iterate
µService A
Verify and Build in Pipeline
µService B
Verify and Build in Pipeline
µService C
Verify and Build in Pipeline
GitOps
Dev
Deploy
Dashboard | Metrics | OpenTracing | Logging
µS
A
µS
B
µS
C
Runtime Component Operator
Service Binding Operator
Red Hat
OpenShift
promote
GitOps
Prod
Deploy
Dashboard | Metrics | OpenTracing | Logging
Production Environment
Runtime Component Operator
Service Binding Operator
Red Hat
OpenShift

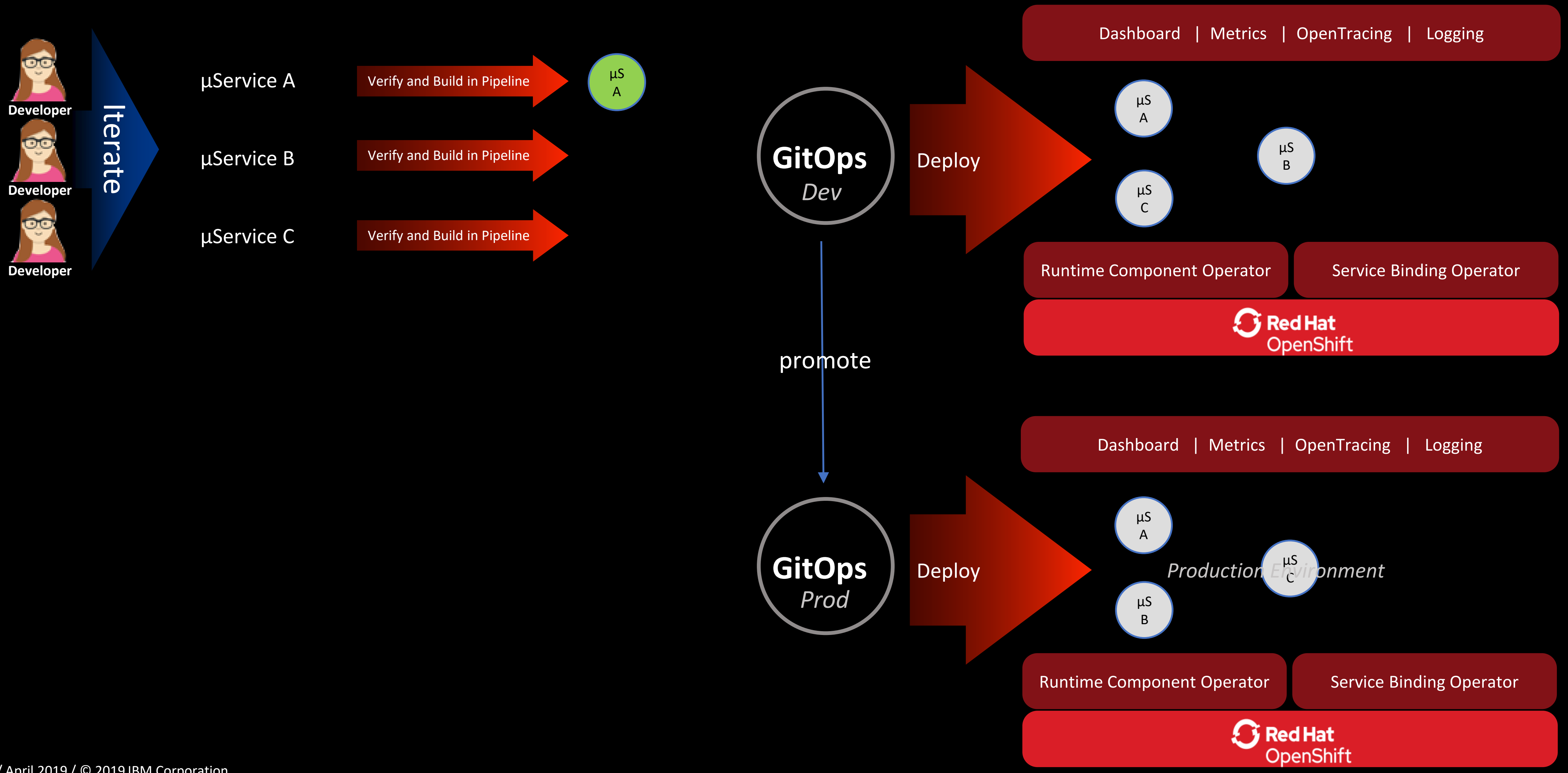
GitOps with Kustomize

Kubernetes Native Configuration Control and Management


Developer
Developer
Developer
Iterate
µService A
µService B
µService C
Verify and Build in Pipeline
Verify and Build in Pipeline
Verify and Build in Pipeline
GitOps
Dev
Deploy
Dashboard | Metrics | OpenTracing | Logging
µS A
µS B
µS C
Runtime Component Operator
Service Binding Operator
Red Hat OpenShift
promote
GitOps
Prod
Deploy
Dashboard | Metrics | OpenTracing | Logging
Production Environment
Runtime Component Operator
Service Binding Operator
Red Hat OpenShift

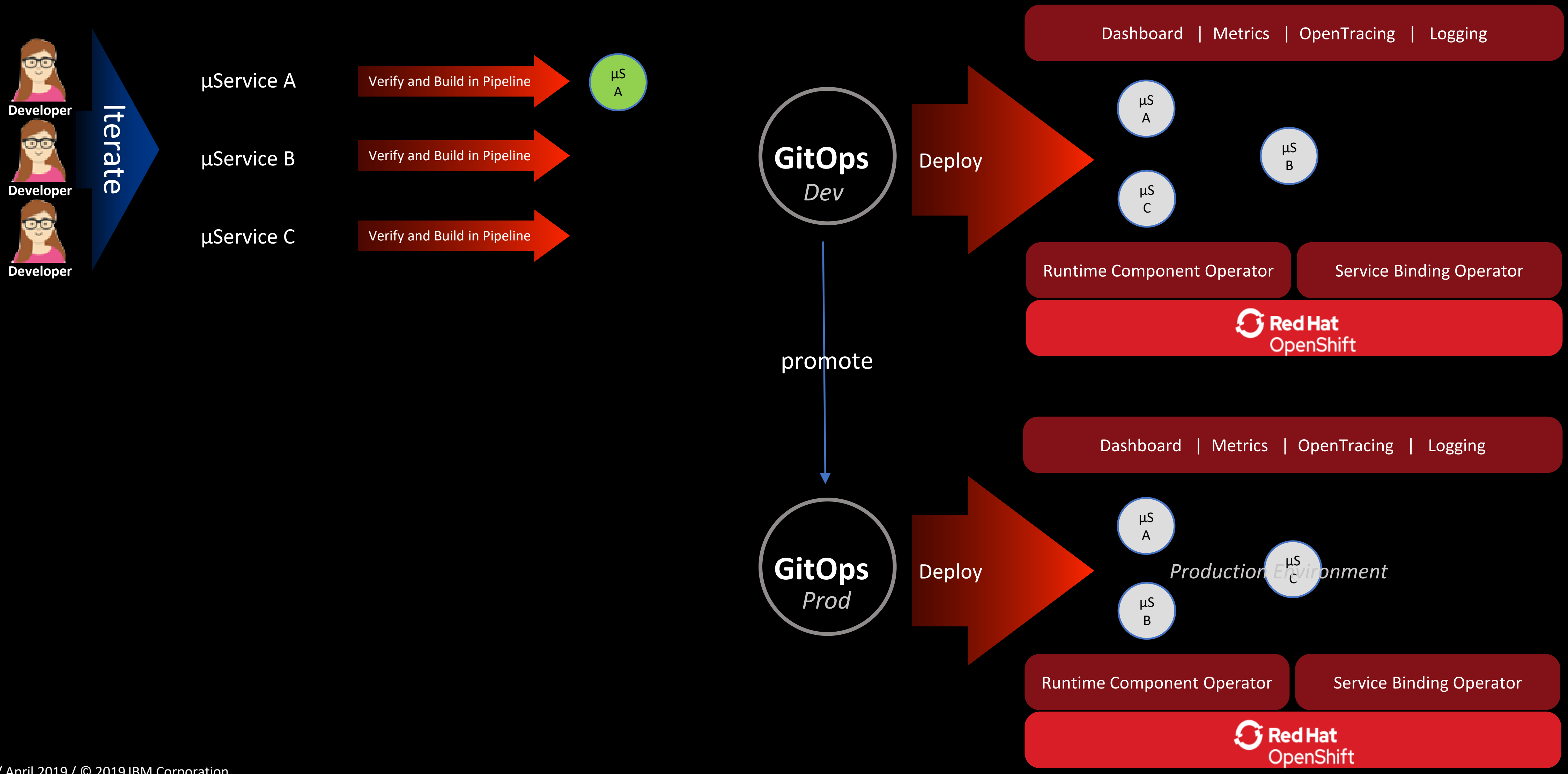
GitOps with Kustomize

Kubernetes Native Configuration Control and Management



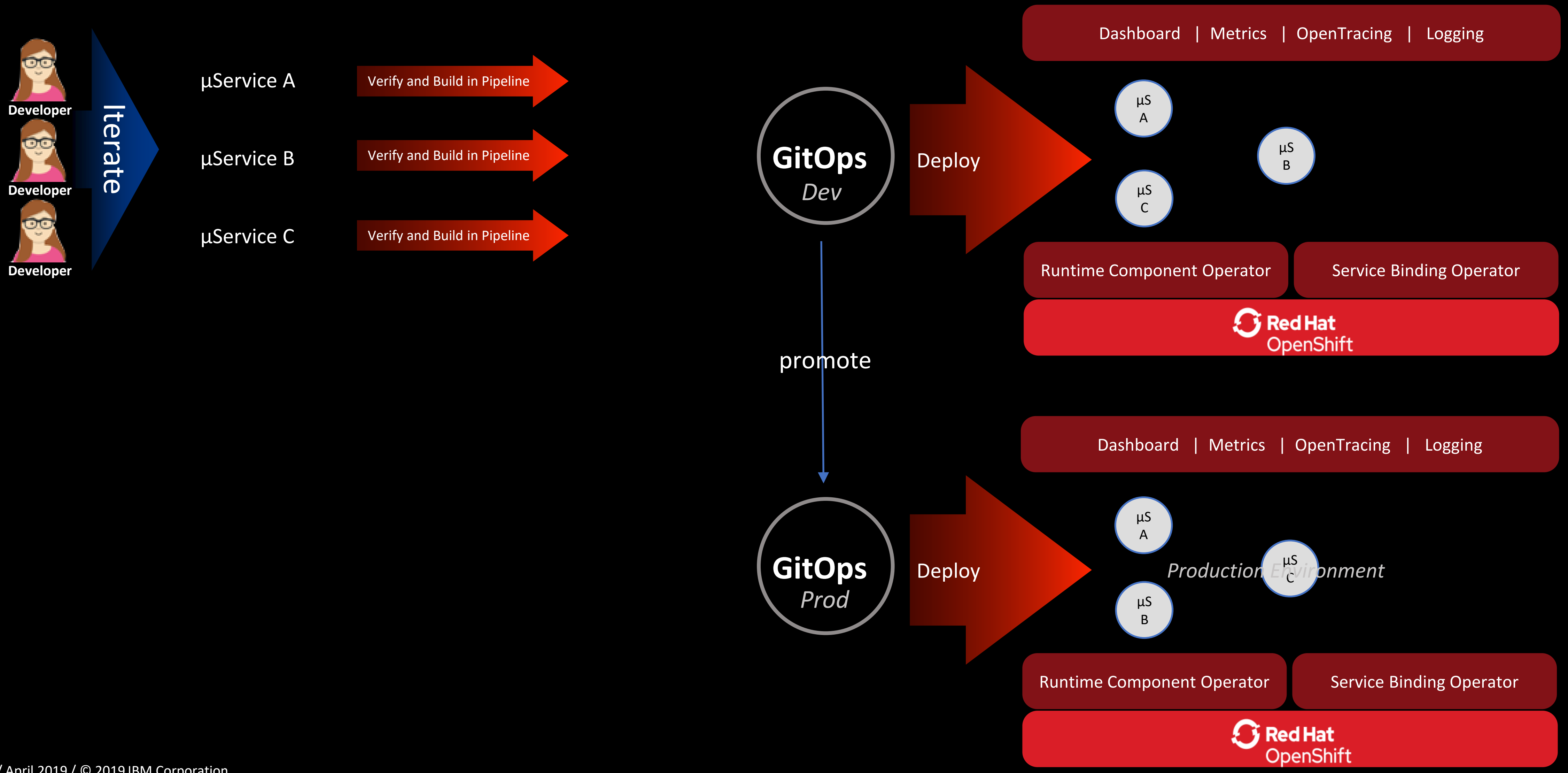
GitOps with Kustomize

Kubernetes Native Configuration Control and Management



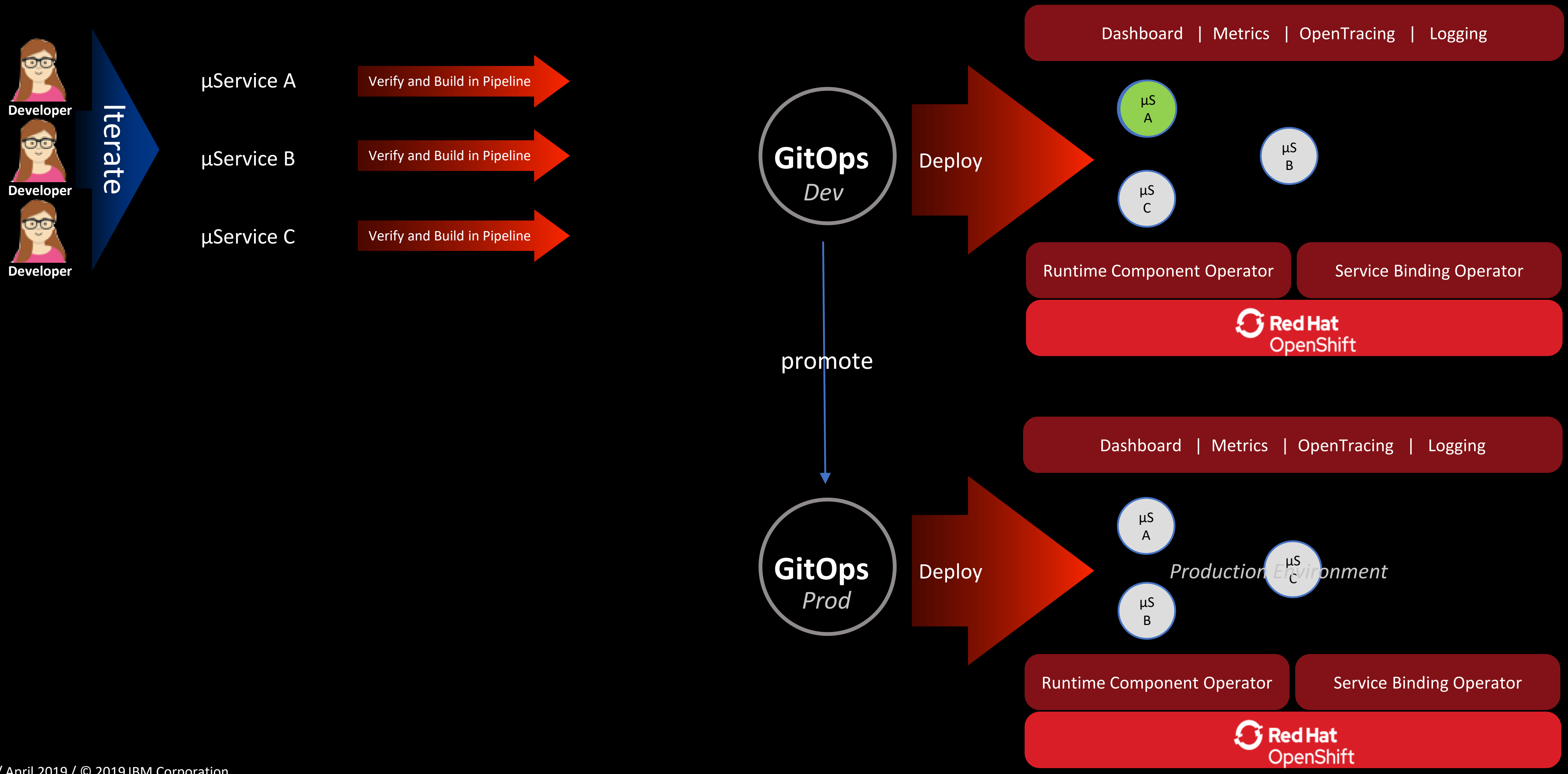
GitOps with Kustomize

Kubernetes Native Configuration Control and Management



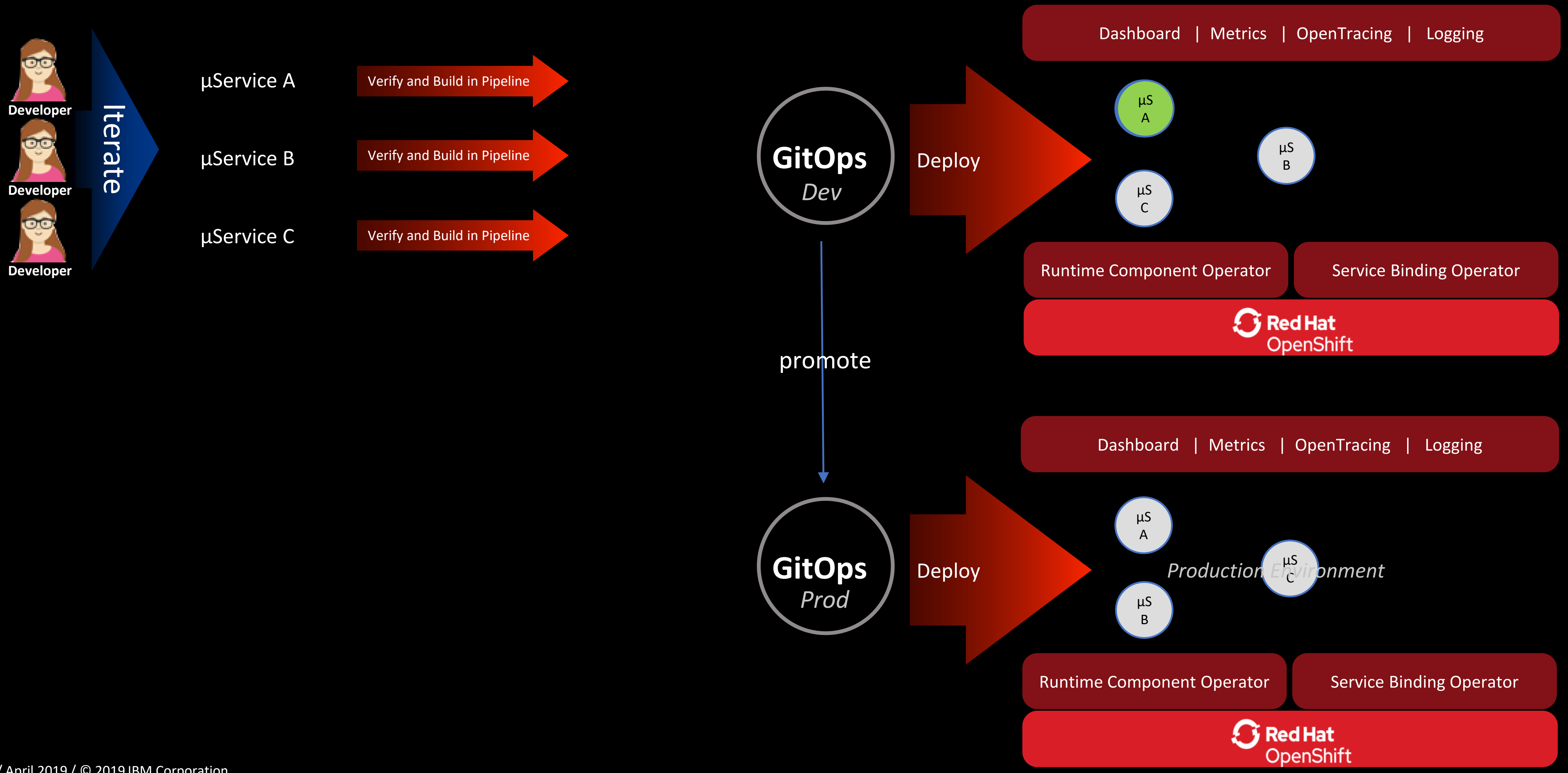
GitOps with Kustomize

Kubernetes Native Configuration Control and Management



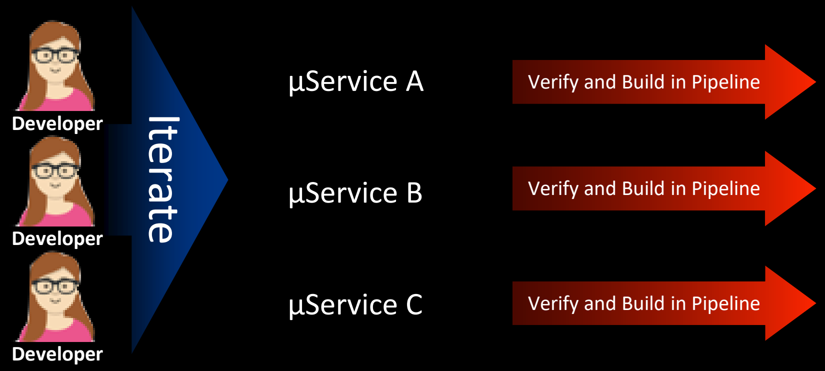
GitOps with Kustomize

Kubernetes Native Configuration Control and Management

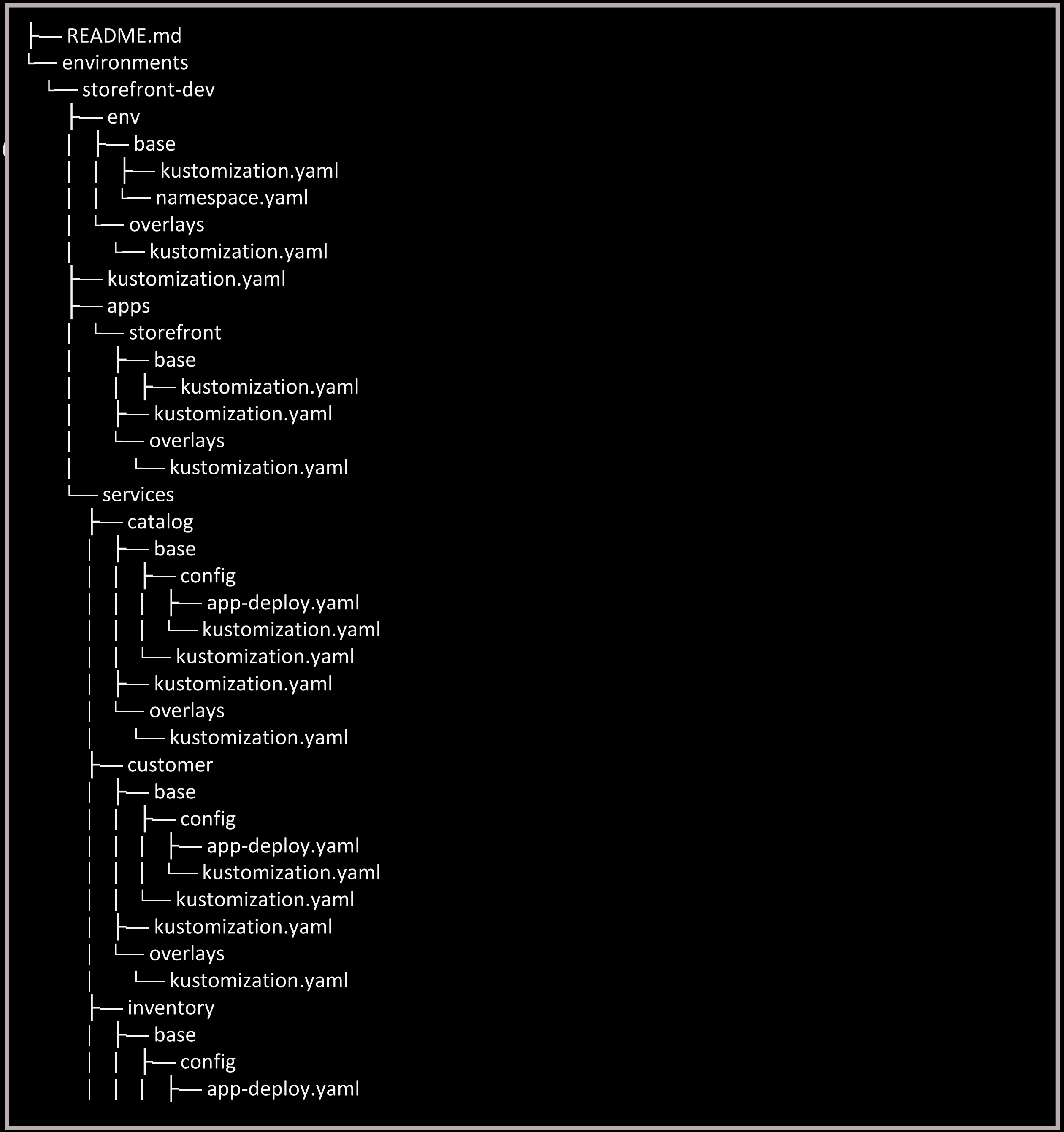


GitOps with Kustomize

Kubernetes Native Configuration Control and Man



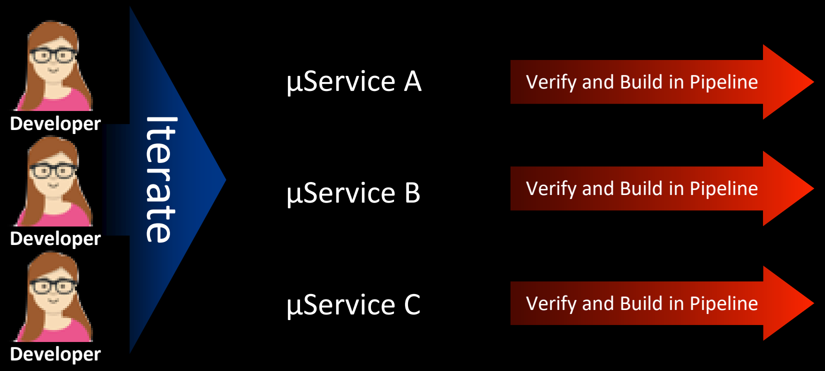
Developer
Developer
Developer
Iterate
µService A
Verify and Build in Pipeline
µService B
Verify and Build in Pipeline
µService C
Verify and Build in Pipeline



```
├── README.md
└── environments
    └── storefront-dev
        ├── env
        │   ├── base
        │   │   ├── kustomization.yaml
        │   │   └── namespace.yaml
        │   └── overlays
        │       └── kustomization.yaml
        ├── kustomization.yaml
        ├── apps
        │   └── storefront
        │       ├── base
        │       │   ├── kustomization.yaml
        │       ├── kustomization.yaml
        │       └── overlays
        │           └── kustomization.yaml
        └── services
            ├── catalog
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── customer
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── inventory
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
```

GitOps with Kustomize

Kubernetes Native Configuration Control and Man



Developer

Developer

Developer

Iterate

µService A — Verify and Build in Pipeline

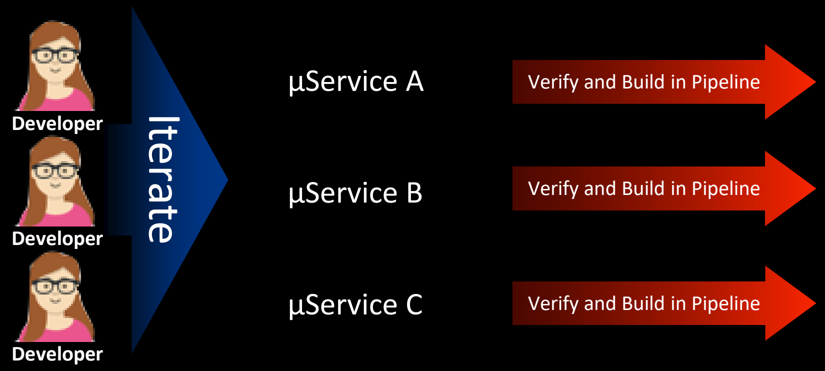
µService B — Verify and Build in Pipeline

µService C — Verify and Build in Pipeline

```
├── README.md
└── environments
    └── storefront-dev
        ├── env
        │   ├── base
        │   │   ├── kustomization.yaml
        │   │   └── namespace.yaml
        │   └── overlays
        │       └── kustomization.yaml
        ├── kustomization.yaml
        ├── apps
        │   └── storefront
        │       ├── base
        │       │   ├── kustomization.yaml
        │       ├── kustomization.yaml
        │       └── overlays
        │           └── kustomization.yaml
        └── services
            ├── catalog
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── customer
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── inventory
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
```

GitOps with Kustomize

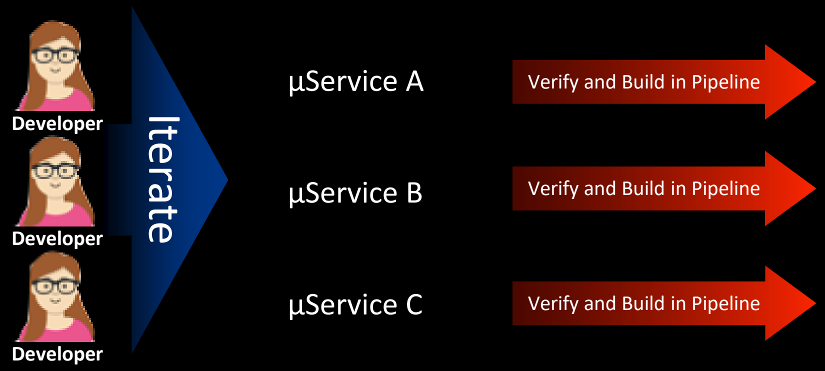
Kubernetes Native Configuration Control and Man

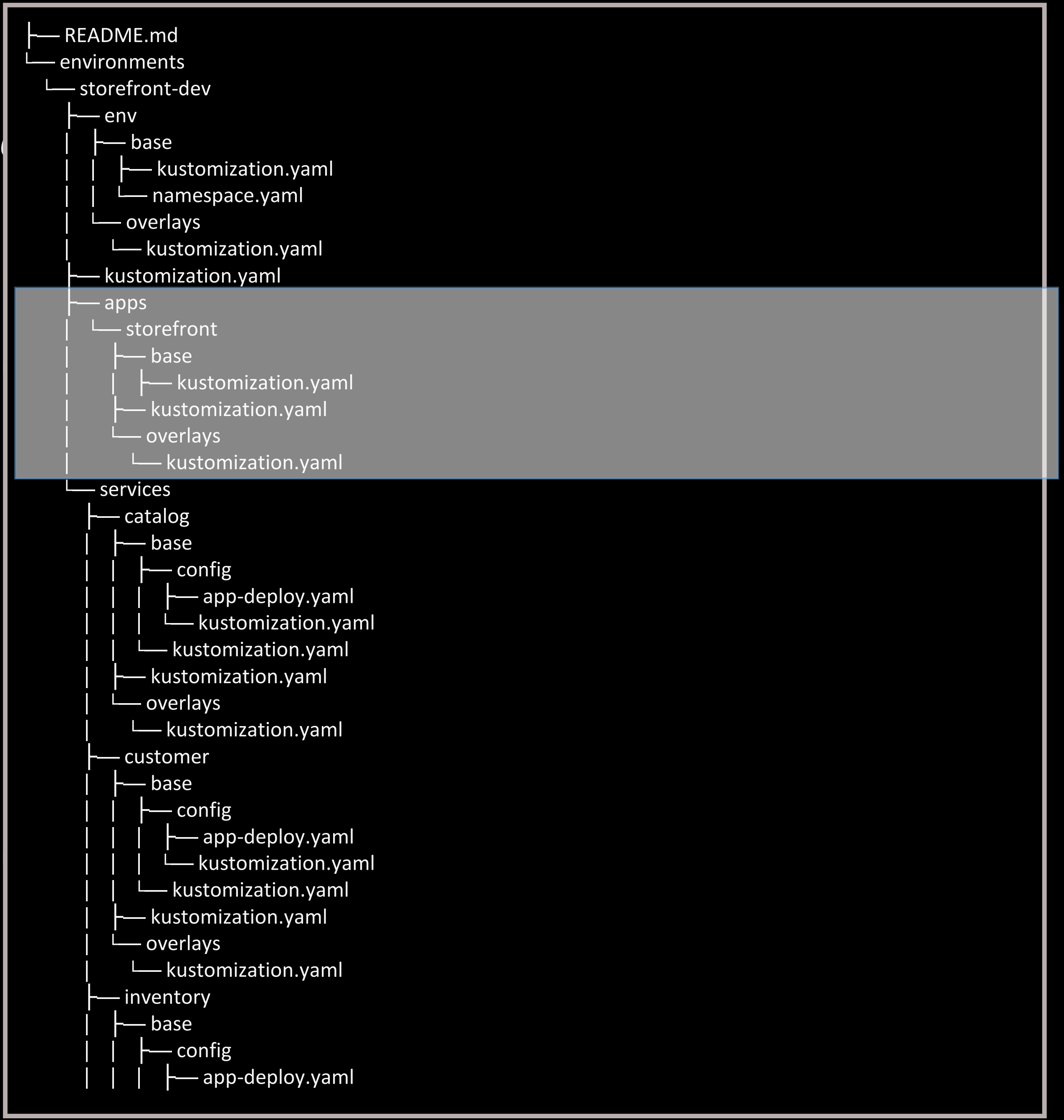

Developer
Developer
Developer
Iterate
µService A
Verify and Build in Pipeline
µService B
Verify and Build in Pipeline
µService C
Verify and Build in Pipeline

```
├── README.md
└── environments
    └── storefront-dev
        ├── env
        │   ├── base
        │   │   ├── kustomization.yaml
        │   │   └── namespace.yaml
        │   └── overlays
        │       └── kustomization.yaml
        ├── kustomization.yaml
        ├── apps
        │   └── storefront
        │       ├── base
        │       │   ├── kustomization.yaml
        │       ├── kustomization.yaml
        │       └── overlays
        │           └── kustomization.yaml
        └── services
            ├── catalog
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── customer
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── inventory
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
```

GitOps with Kustomize

Kubernetes Native Configuration Control and Man

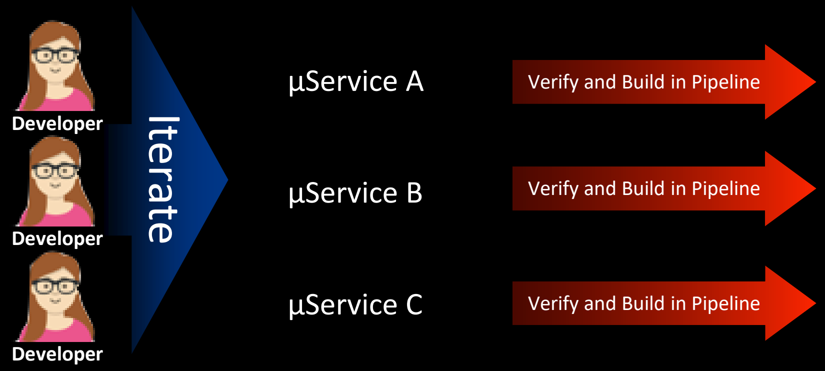

Developer
Developer
Developer
Iterate
µService A
Verify and Build in Pipeline
µService B
Verify and Build in Pipeline
µService C
Verify and Build in Pipeline

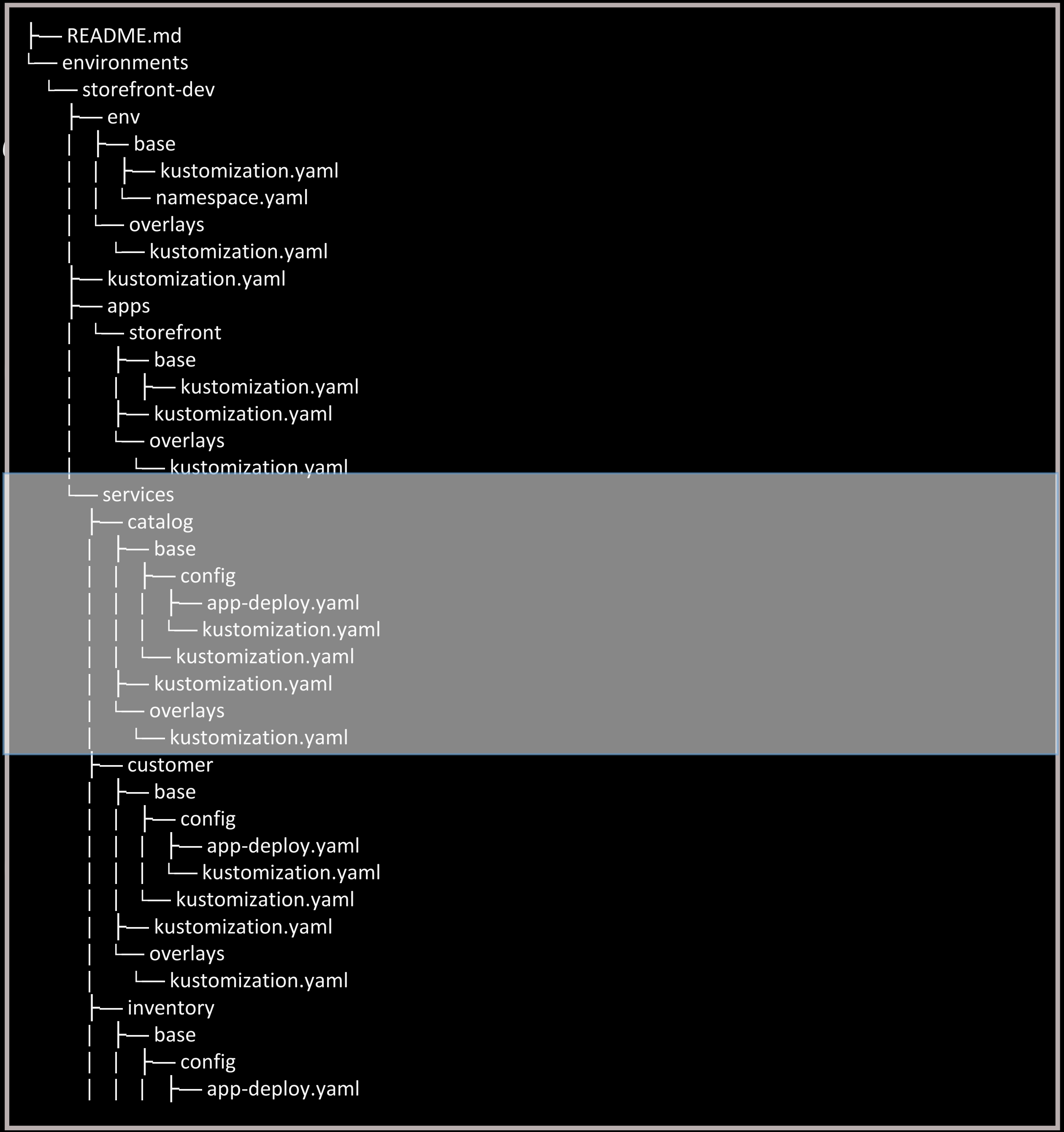


```
├── README.md
└── environments
    └── storefront-dev
        ├── env
        │   ├── base
        │   │   ├── kustomization.yaml
        │   │   └── namespace.yaml
        │   └── overlays
        │       └── kustomization.yaml
        ├── kustomization.yaml
        ├── apps
        │   └── storefront
        │       ├── base
        │       │   ├── kustomization.yaml
        │       ├── kustomization.yaml
        │       └── overlays
        │           └── kustomization.yaml
        └── services
            ├── catalog
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── customer
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── inventory
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
```

GitOps with Kustomize

Kubernetes Native Configuration Control and Man

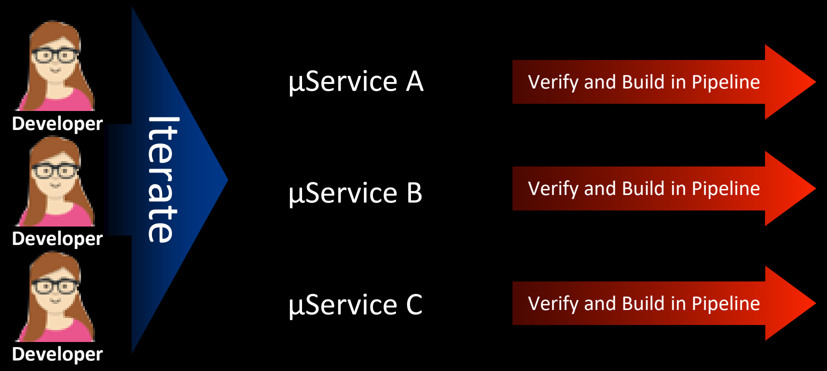

Developer
Developer
Developer
Iterate
µService A
Verify and Build in Pipeline
µService B
Verify and Build in Pipeline
µService C
Verify and Build in Pipeline



```
├── README.md
└── environments
    └── storefront-dev
        ├── env
        │   ├── base
        │   │   ├── kustomization.yaml
        │   │   └── namespace.yaml
        │   └── overlays
        │       └── kustomization.yaml
        ├── kustomization.yaml
        ├── apps
        │   └── storefront
        │       ├── base
        │       │   ├── kustomization.yaml
        │       ├── kustomization.yaml
        │       └── overlays
        │           └── kustomization.yaml
        └── services
            ├── catalog
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── customer
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── inventory
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
```

GitOps with Kustomize

Kubernetes Native Configuration Control and Man



Developer

Developer

Developer

Iterate

µService A — Verify and Build in Pipeline

µService B — Verify and Build in Pipeline

µService C — Verify and Build in Pipeline

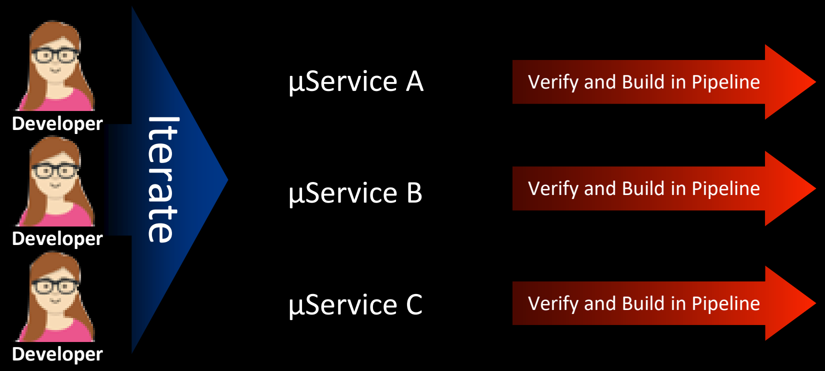
Create PR to update
services config directory

Overlays used to
provide operational config

```
├── README.md
└── environments
    └── storefront-dev
        ├── env
        │   ├── base
        │   │   ├── kustomization.yaml
        │   │   └── namespace.yaml
        │   └── overlays
        │       └── kustomization.yaml
        ├── kustomization.yaml
        ├── apps
        │   └── storefront
        │       ├── base
        │       │   ├── kustomization.yaml
        │       ├── kustomization.yaml
        │       └── overlays
        │           └── kustomization.yaml
        └── services
            ├── catalog
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── customer
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── inventory
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
```

GitOps with Kustomize

Kubernetes Native Configuration Control and Man


Developer
Developer
Developer
Iterate
µService A
Verify and Build in Pipeline
µService B
Verify and Build in Pipeline
µService C
Verify and Build in Pipeline

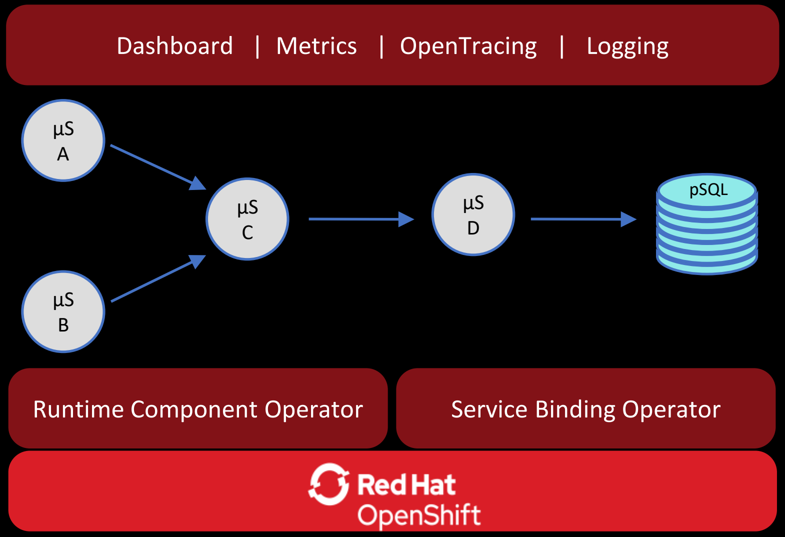
```
$ ./odo pipelines --help
Pipeline operations (bootstrap, build, environment, init, service, webhook)

Usage:
  odo pipelines [flags]
  odo pipelines [command]

Examples:
odo pipelines
init

  See sub-commands individually for more examples

Available Commands:
  bootstrap   Initialize pipelines
  build       Build pipelines files
  environment Manage an environment in GitOps (add)
  init        Initialize pipelines
  service     Manage services in an environment (add)
  webhook     Manage Git repository webhooks (create, delete, list)
```

```
├── README.md
└── environments
    └── storefront-dev
        ├── env
        │   ├── base
        │   │   ├── kustomization.yaml
        │   │   └── namespace.yaml
        │   └── overlays
        │       └── kustomization.yaml
        ├── kustomization.yaml
        ├── apps
        │   └── storefront
        │       ├── base
        │       │   ├── kustomization.yaml
        │       ├── kustomization.yaml
        │       └── overlays
        │           └── kustomization.yaml
        └── services
            ├── catalog
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── customer
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
            │   │   │   └── kustomization.yaml
            │   │   └── kustomization.yaml
            │   ├── kustomization.yaml
            │   └── overlays
            │       └── kustomization.yaml
            ├── inventory
            │   ├── base
            │   │   ├── config
            │   │   │   ├── app-deploy.yaml
```

Service Binding

Service Discovery and Dynamic Configuration

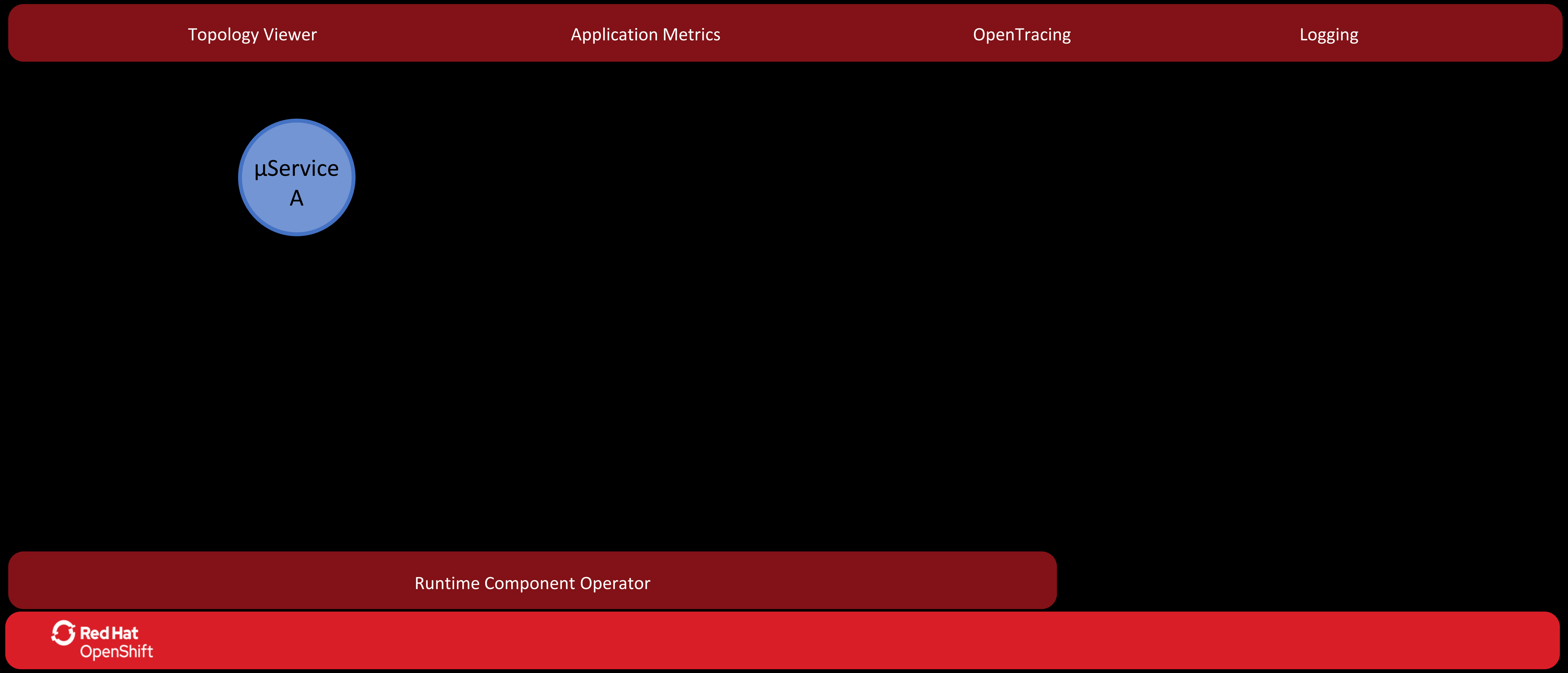
- ✓ Enables dynamic discovery and configuration between microservices
- ✓ Enables dynamic discovery and configuration with services
- ✓ Removes hard coded configuration from microservices
- ✓ Makes microservices portable between environments


Dashboard | Metrics | OpenTracing | Logging
µS A
µS B
µS C
µS D
pSQL
Runtime Component Operator
Service Binding Operator
Red Hat OpenShift

Development, Staging and Production Environments

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService A

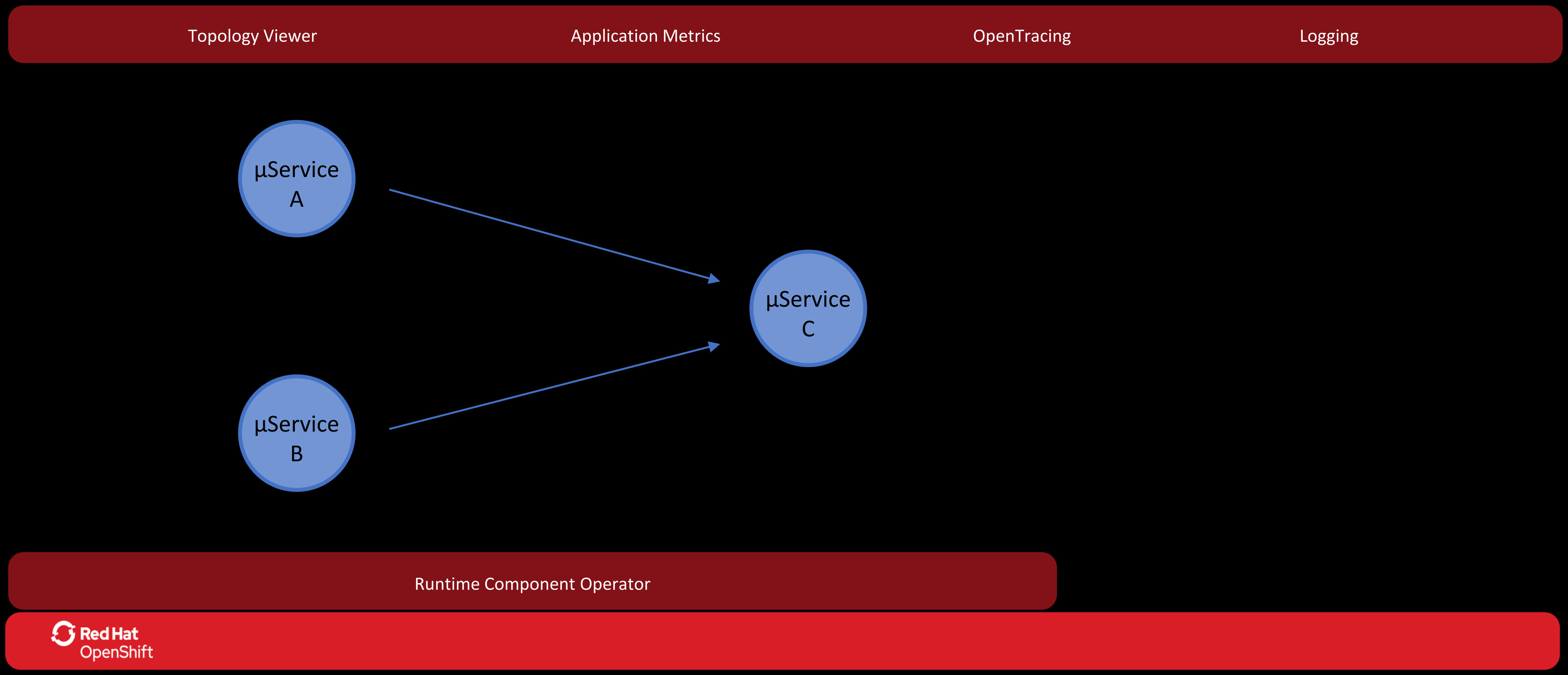
Runtime Component Operator

Red Hat OpenShift

Development Environment: “storefront-dev”

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService A

µService B

µService C

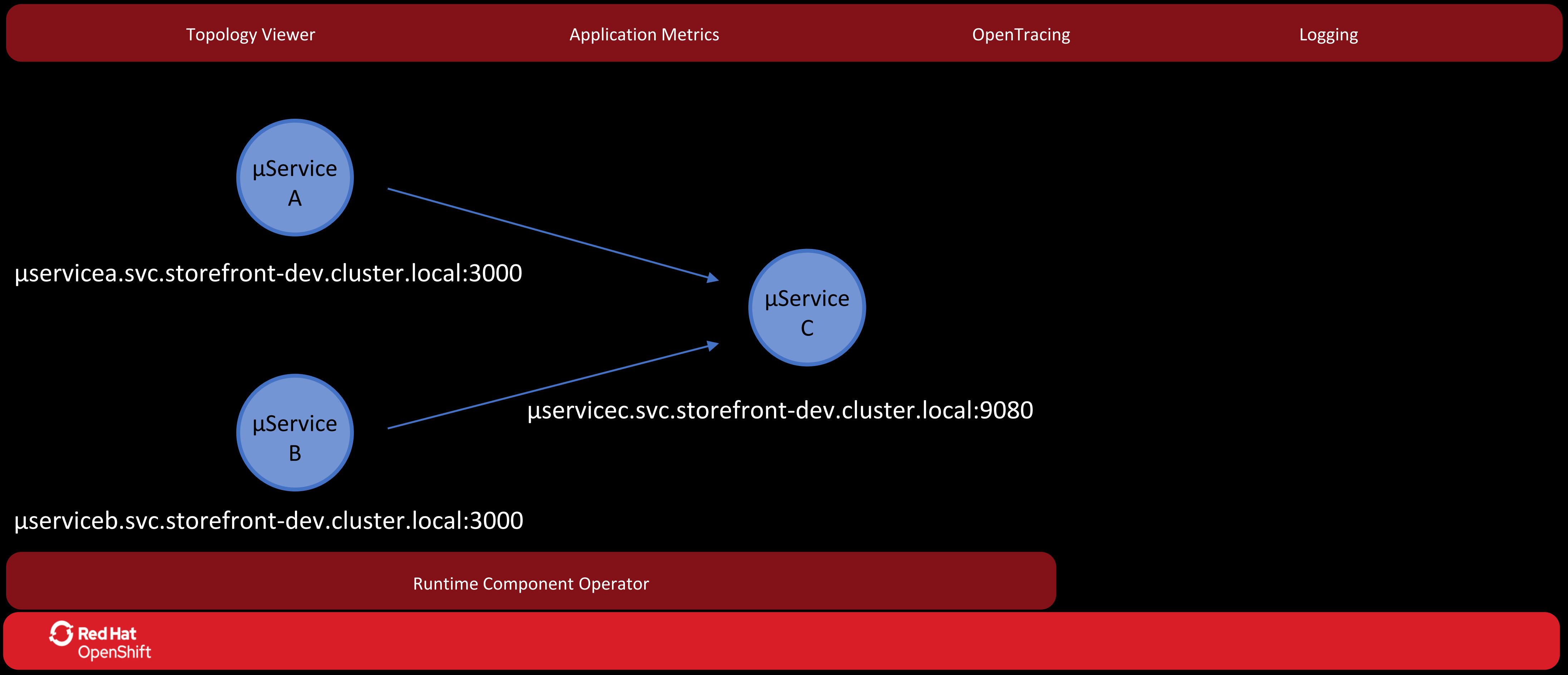
Runtime Component Operator

Red Hat OpenShift

Development Environment: “storefront-dev”

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer
Application Metrics
OpenTracing
Logging
µService A
µservicea.svc.storefront-dev.cluster.local:3000
µService C
µservicec.svc.storefront-dev.cluster.local:9080
µService B
µserviceb.svc.storefront-dev.cluster.local:3000
Runtime Component Operator
Red Hat OpenShift

Development Environment: "storefront-dev"

Service Binding

Service Discovery and Dynamic Configuration

app-deploy.yaml for µService C

```
apiVersion: app.stacks/v1beta1
kind: RuntimeComponent
metadata:
 name: µServiceC
spec:
 applicationImage: quay.io/µservices/µservicec
 version: 1.0.0
 createKnativeService: false
 expose: true
 livenessProbe:
   …

 readinessProbe:
   …

 monitoring:
   labels:
     k8s-app: storefront
 service:
   port: 9080
   type: NodePort
```

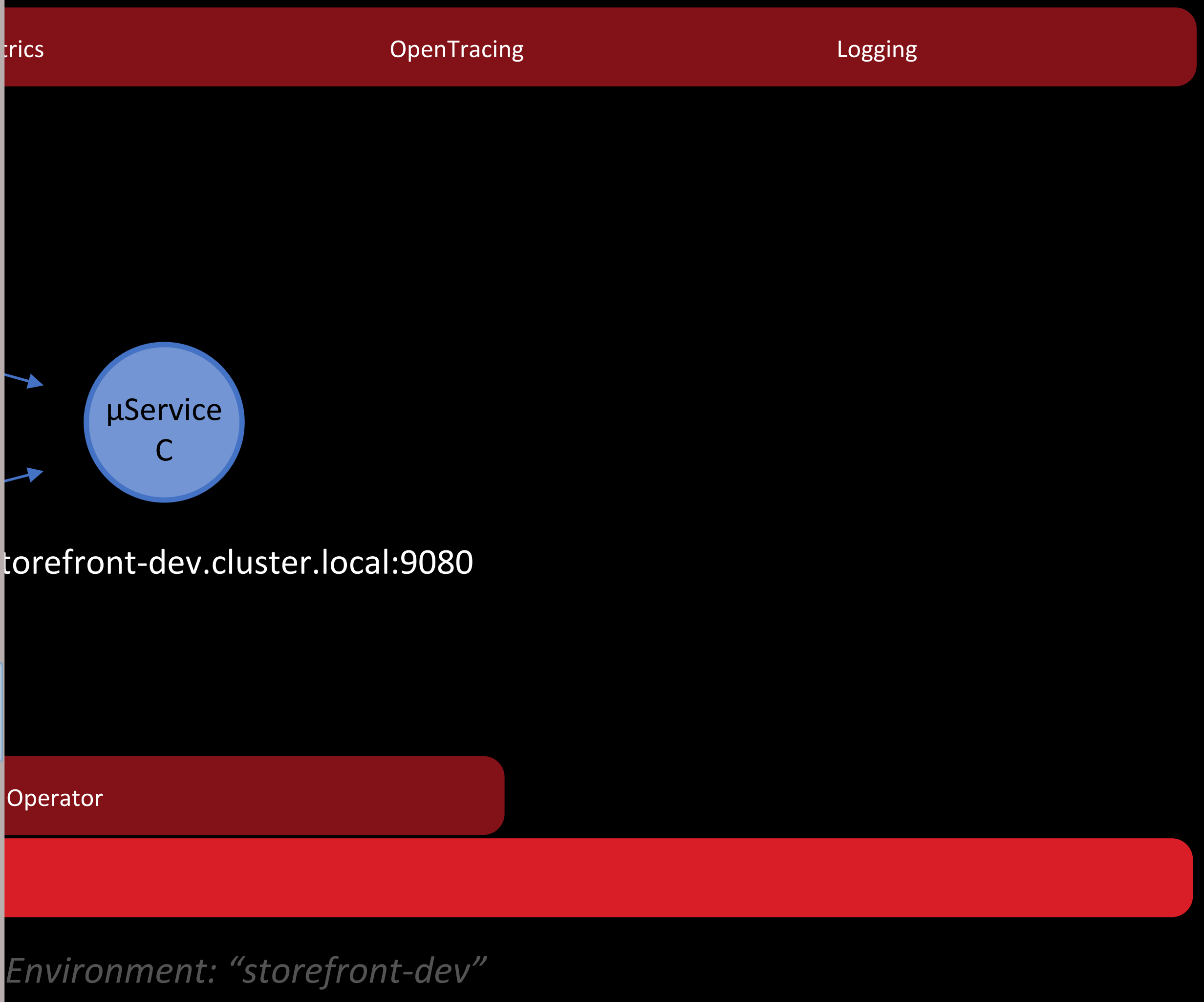
rics
OpenTracing
Logging
µService C
torefront-dev.cluster.local:9080
Operator
Environment: “storefront-dev”

Service Binding

Service Discovery and Dynamic Configuration

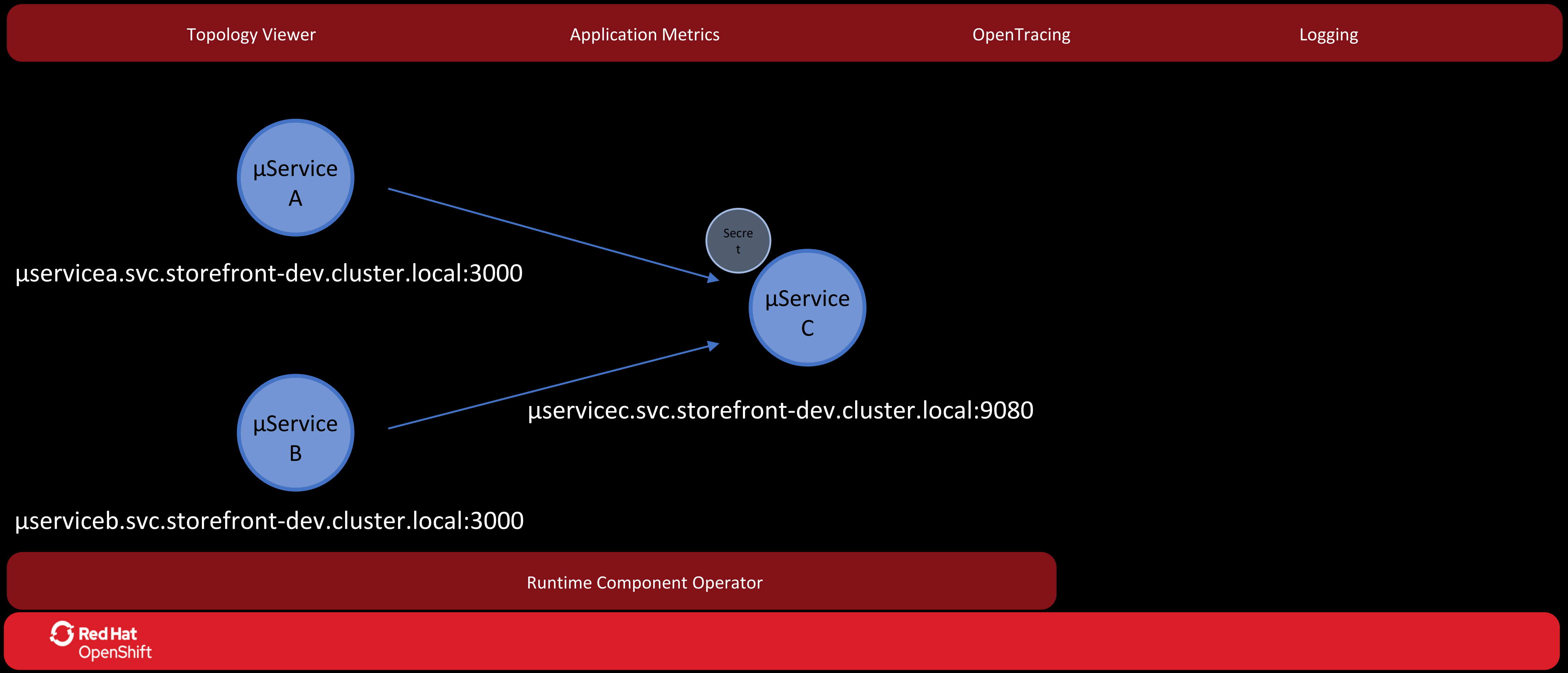
app-deploy.yaml for µService C

```
apiVersion: app.stacks/v1beta1
kind: RuntimeComponent
metadata:
  name: µServiceC
spec:
  applicationImage: quay.io/µservices/µservicec
  version: 1.0.0
  createKnativeService: false
  expose: true
  livenessProbe:
    …

  readinessProbe:
    …

  monitoring:
    labels:
      k8s-app: storefront
  service:
    port: 9080
    type: NodePort
    provides:
      category: openapi
      context: /
```


rics
OpenTracing
Logging
µService C
torefront-dev.cluster.local:9080
Operator
Environment: “storefront-dev”

Service Binding

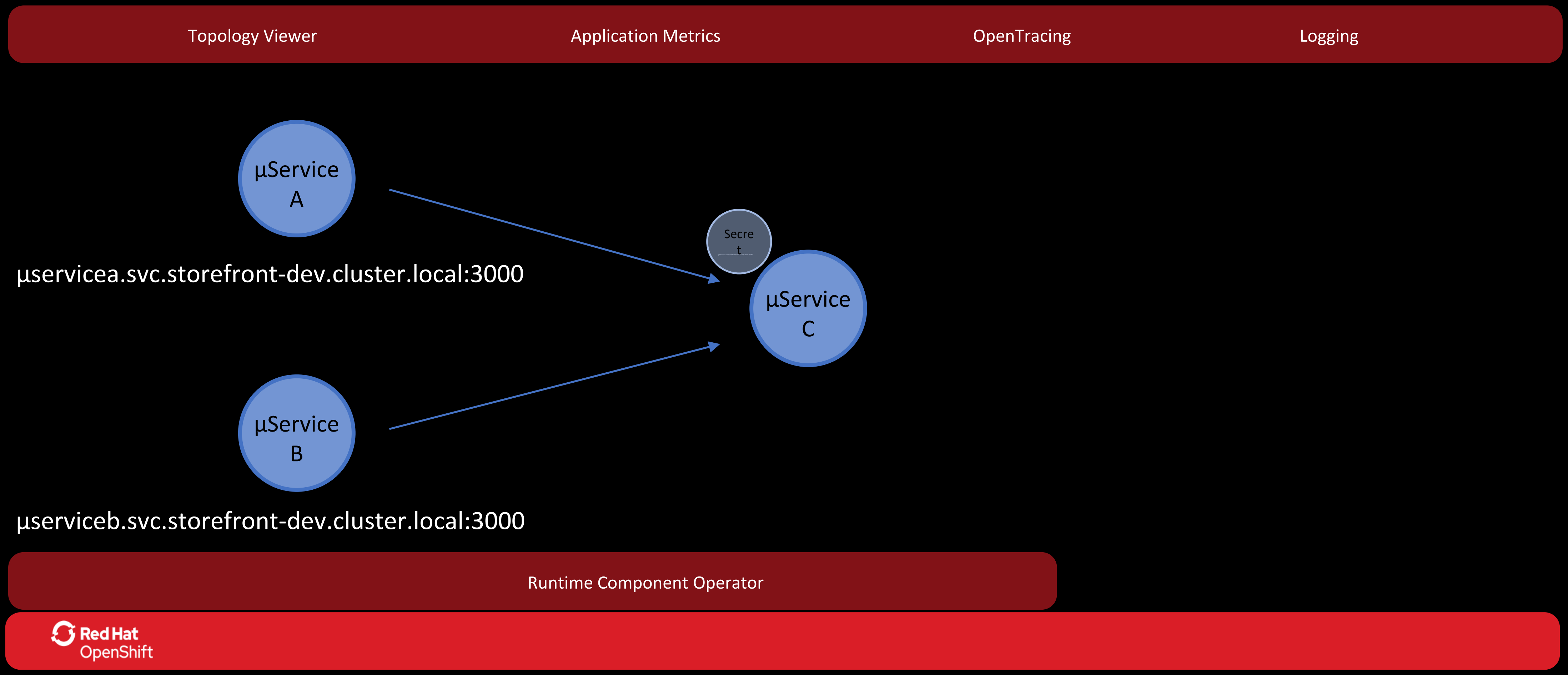
Service Discovery and Dynamic Configuration


Topology Viewer
Application Metrics
OpenTracing
Logging
µService A
µservicea.svc.storefront-dev.cluster.local:3000
Secre t
µService C
µservicec.svc.storefront-dev.cluster.local:9080
µService B
µserviceb.svc.storefront-dev.cluster.local:3000
Runtime Component Operator
Red Hat OpenShift

Development Environment: "storefront-dev"

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µservicea.svc.storefront-dev.cluster.local:3000

µserviceb.svc.storefront-dev.cluster.local:3000

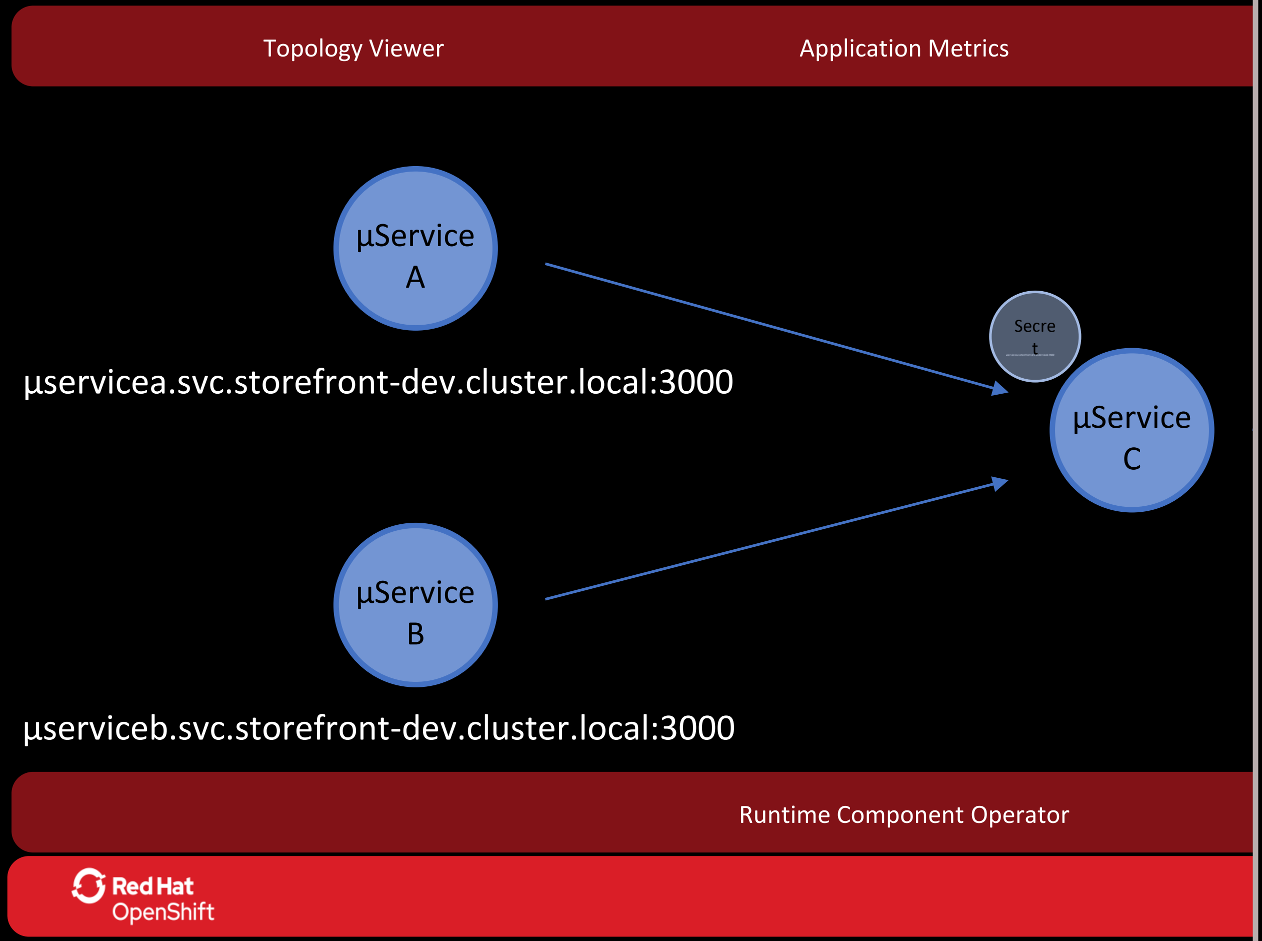
Runtime Component Operator

Red Hat OpenShift

Development Environment: "storefront-dev"

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

µService A

µservicea.svc.storefront-dev.cluster.local:3000

Secre
t

µService C

µService B

µserviceb.svc.storefront-dev.cluster.local:3000

Runtime Component Operator

Red Hat OpenShift

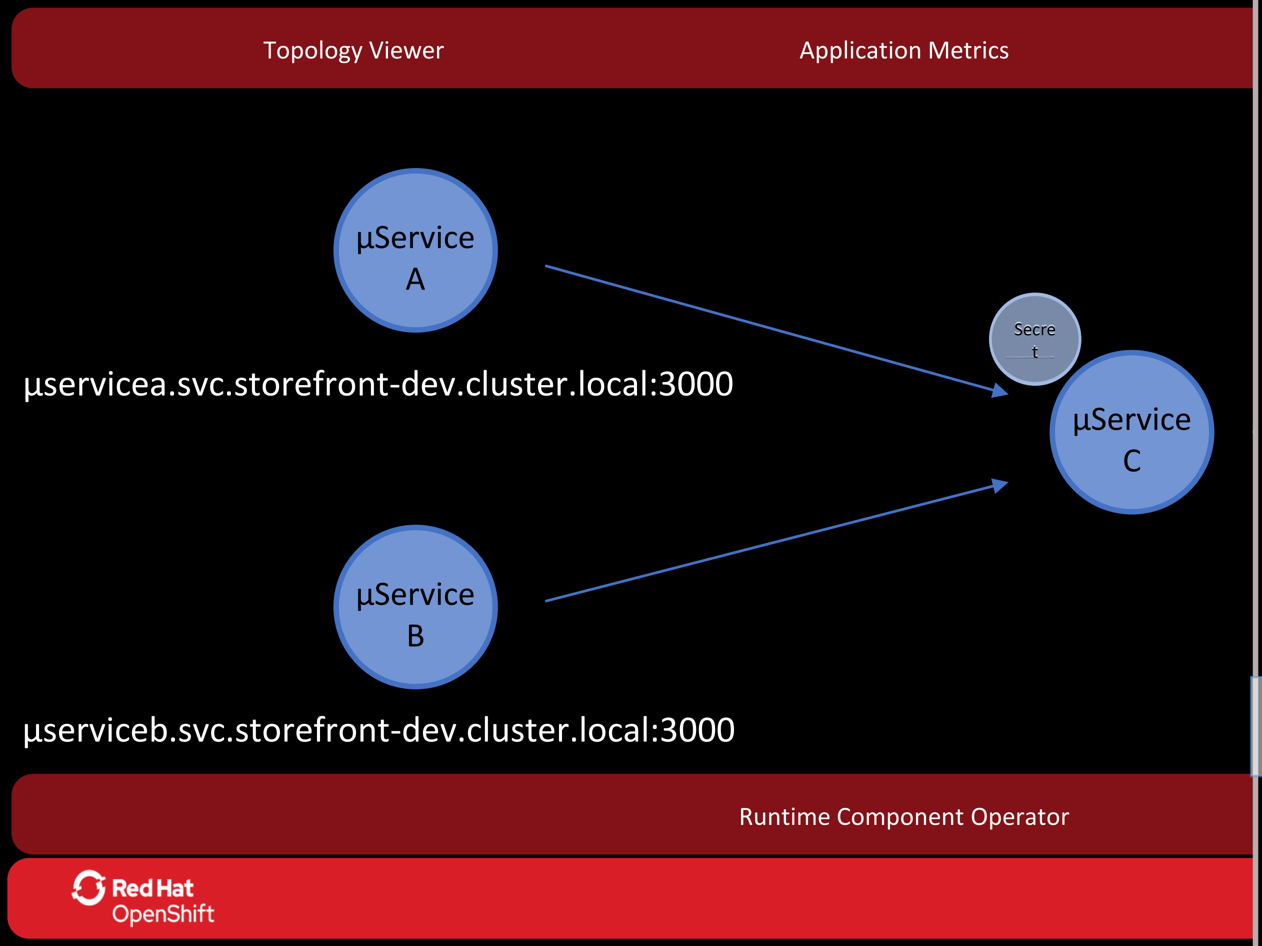
Development Environment: “sto

app-deploy.yaml for µService A

```
apiVersion: app.stacks/v1beta1
kind: RuntimeComponent
metadata:
 name: µServiceA
spec:
 applicationImage: quay.io/µservices/µservicea
 version: 1.0.0
 createKnativeService: false
 expose: true
 livenessProbe:
   …

 readinessProbe:
   …

 monitoring:
  labels:
   k8s-app: storefront
 service:
  port: 3000
  type: NodePort
```

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

µService A

µservicea.svc.storefront-dev.cluster.local:3000

Secret

µService C

µService B

µserviceb.svc.storefront-dev.cluster.local:3000

Runtime Component Operator

Red Hat OpenShift

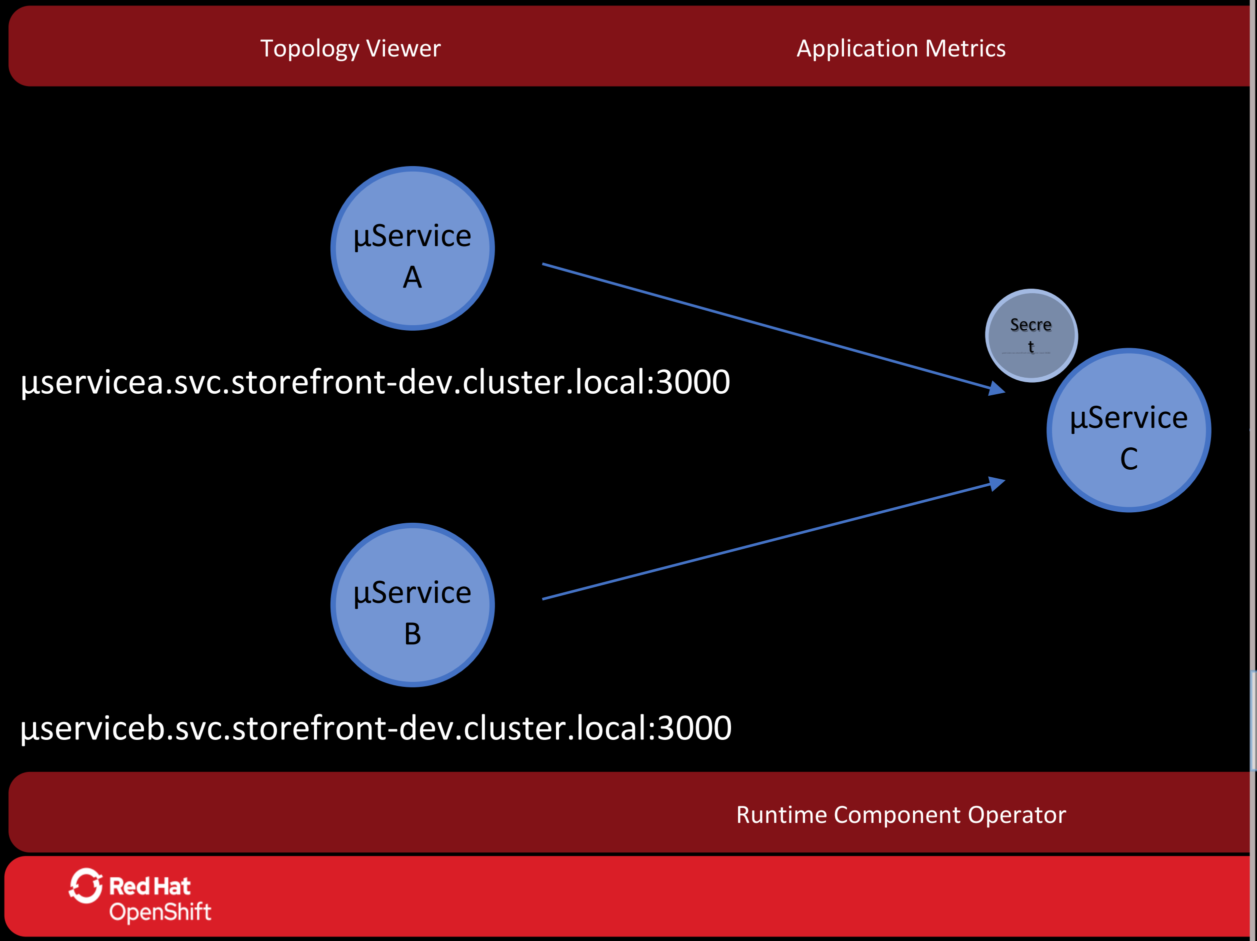
Development Environment: "sto

app-deploy.yaml for µService A

```
apiVersion: app.stacks/v1beta1
kind: RuntimeComponent
metadata:
  name: µServiceA
spec:
  applicationImage: quay.io/µservices/µservicea
  version: 1.0.0
  createKnativeService: false
  expose: true
  livenessProbe:
    …

  readinessProbe:
    …

  monitoring:
    labels:
      k8s-app: storefront
  service:
    port: 3000
    type: NodePort
    consumes:
    - category: openapi
      name: µservicec
```

Service Binding

Service Discovery and Dynamic Configuration


Topology Viewer
Application Metrics
µService A
µservicea.svc.storefront-dev.cluster.local:3000
Secre
t
µService C
µService B
µserviceb.svc.storefront-dev.cluster.local:3000
Runtime Component Operator
Red Hat OpenShift

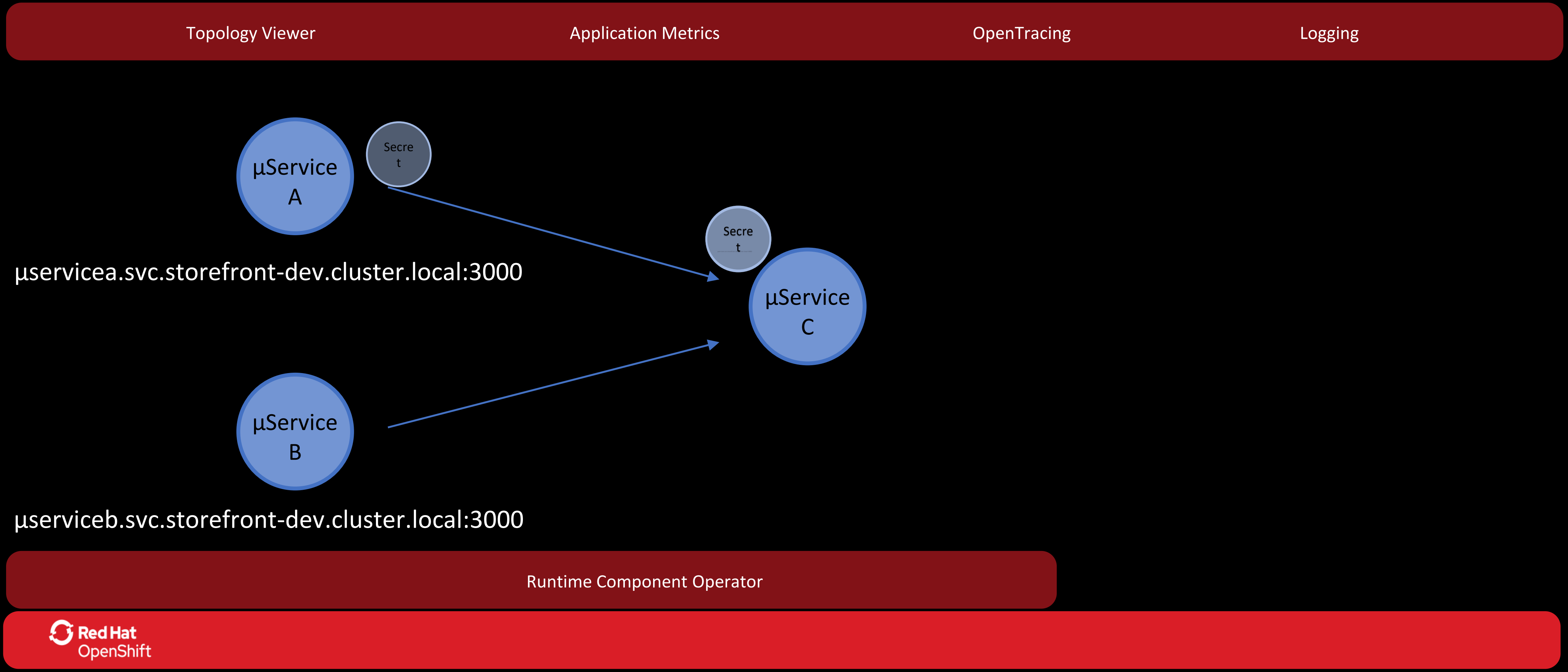
Development Environment: "st

app-deploy.yaml for µService A

```
apiVersion: app.stacks/v1beta1
kind: RuntimeComponent
metadata:
 name: µServiceA
spec:
 applicationImage: quay.io/µservices/µservicea
 version: 1.0.0
 createKnativeService: false
 expose: true
 livenessProbe:
  …

 readinessProbe:
  …

 monitoring:
  labels:
   k8s-app: storefront
 service:
  port: 3000
  type: NodePort
  consumes:
  - category: openapi
   name: µservicec
```

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService A

Secre
t

µservicea.svc.storefront-dev.cluster.local:3000

Secre
t

µService C

µService B

µserviceb.svc.storefront-dev.cluster.local:3000

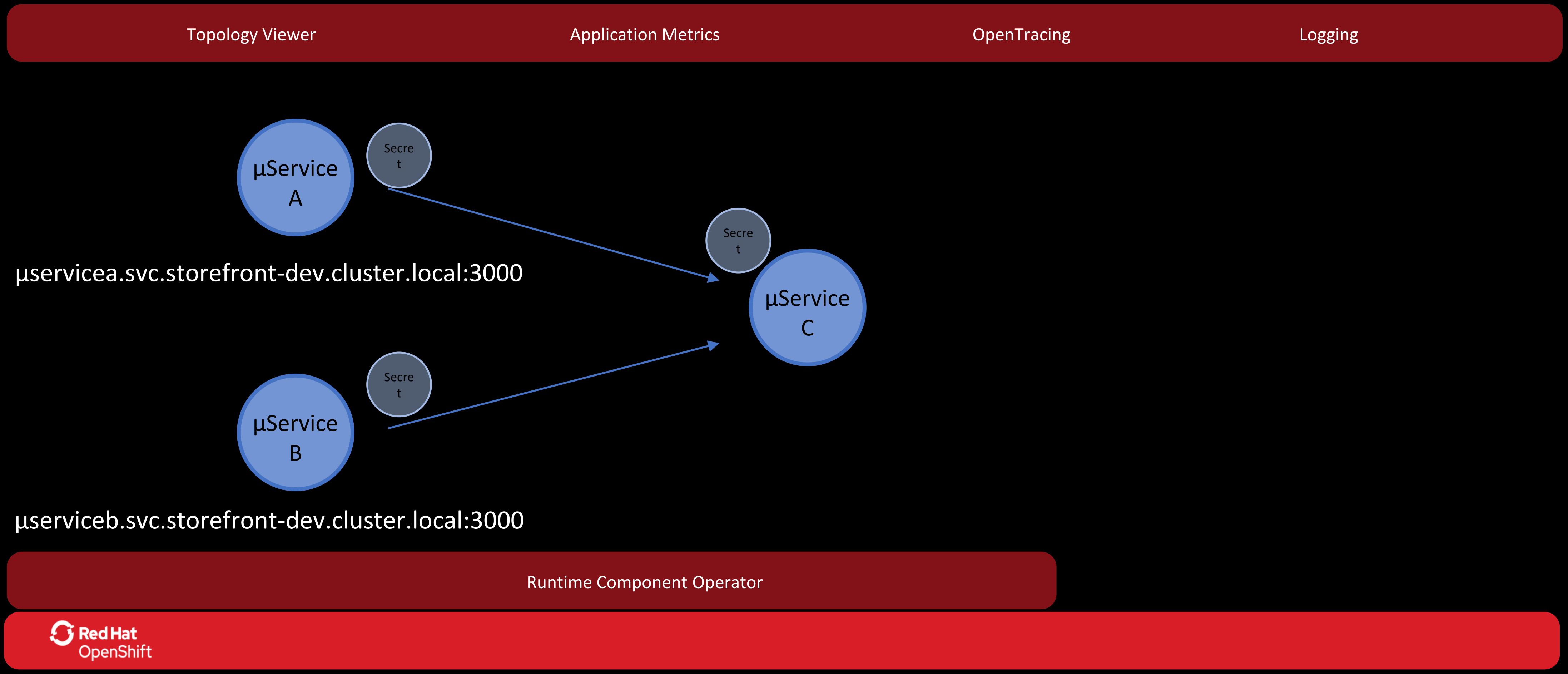
Runtime Component Operator

Red Hat
OpenShift

Development Environment: "storefront-dev"

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService A

Secret

µservicea.svc.storefront-dev.cluster.local:3000

Secret

µService C

Secret

µService B

µserviceb.svc.storefront-dev.cluster.local:3000

Runtime Component Operator

Red Hat OpenShift

Development Environment: “storefront-dev”

Service Binding

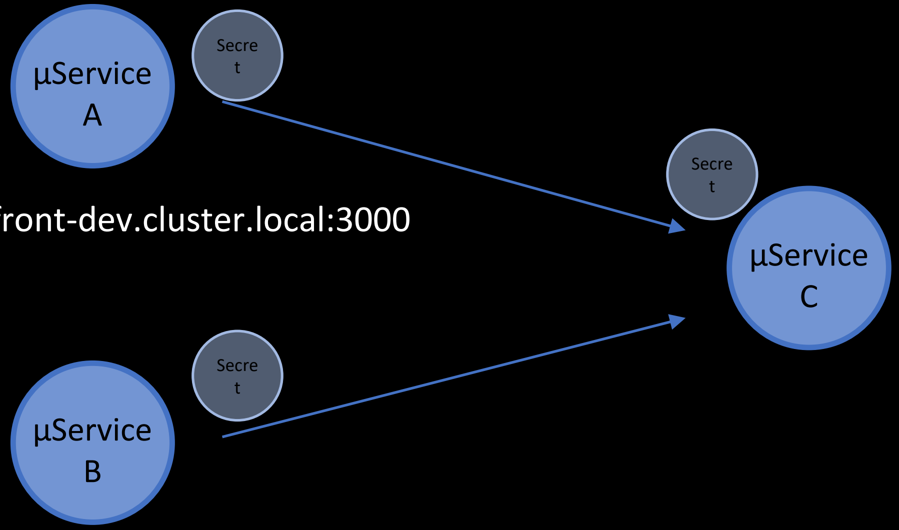
Service Discovery and Dynamic Configuration

Topology Viewer

Application Metrics

OpenTracing

Logging



µservicea.svc.storefront-dev.cluster.local:3000

µserviceb.svc.storefront-dev.cluster.local:3000

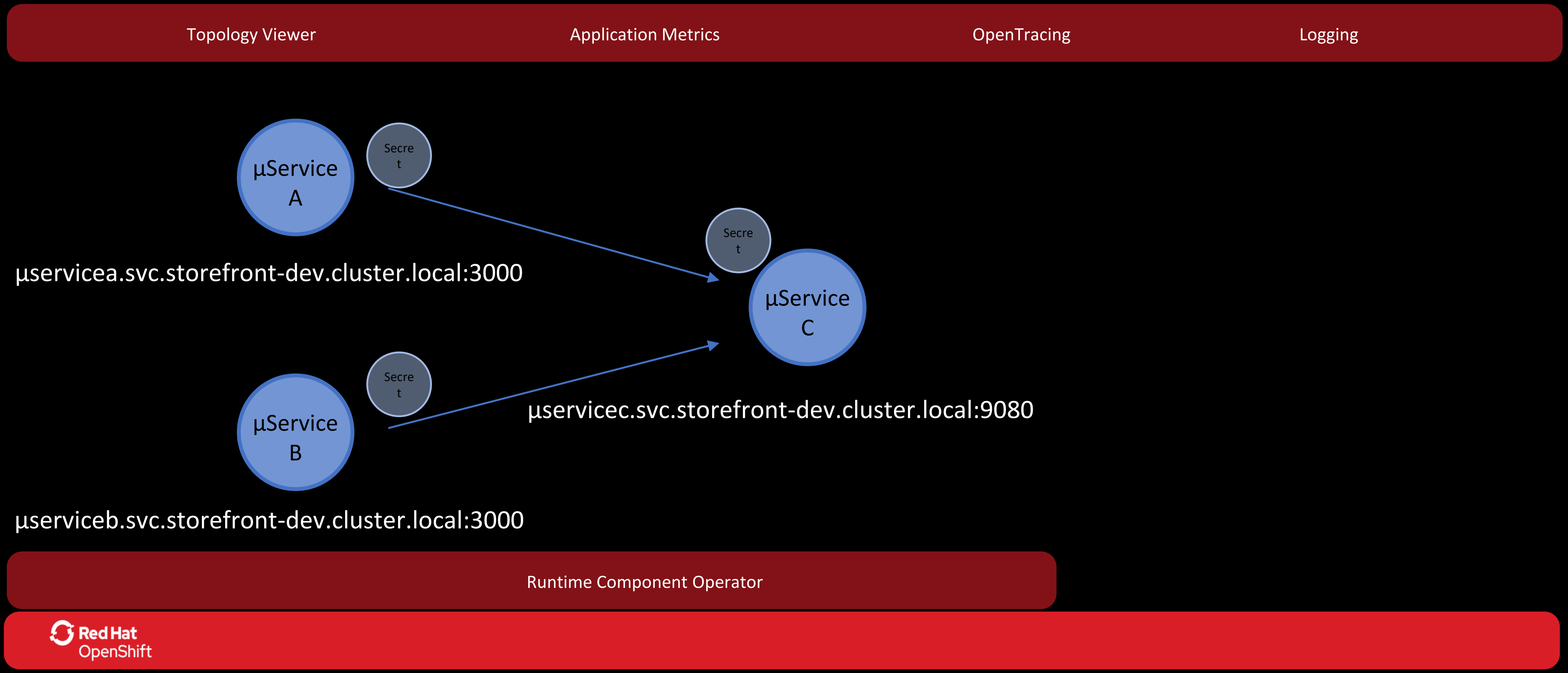
Runtime Component Operator

Red Hat OpenShift

Development Environment: “storefront-staging”

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService
A

Secre
t

µservicea.svc.storefront-dev.cluster.local:3000

Secre
t

µService
C

Secre
t

µService
B

µservicec.svc.storefront-dev.cluster.local:9080

µserviceb.svc.storefront-dev.cluster.local:3000

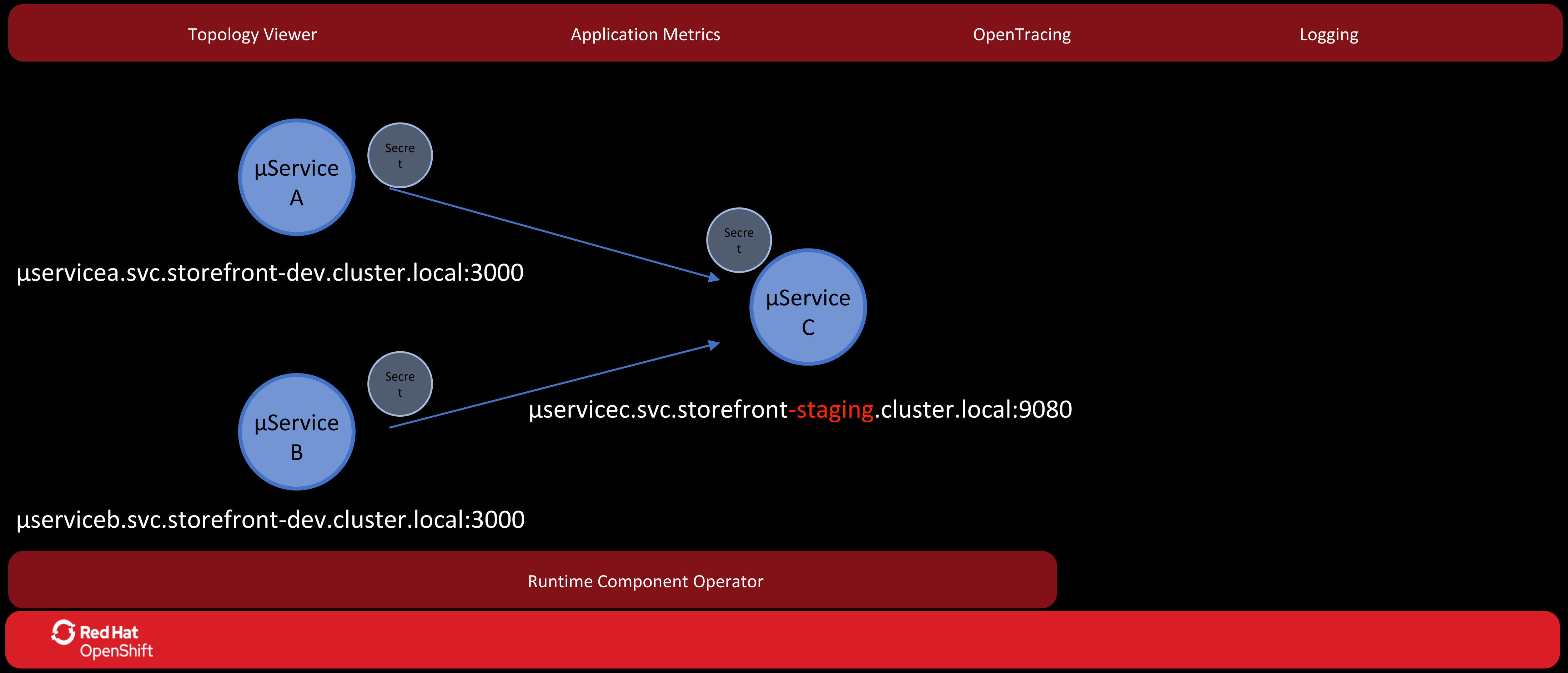
Runtime Component Operator

Red Hat
OpenShift

Development Environment: "storefront-staging"

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService A

Secret

µservicea.svc.storefront-dev.cluster.local:3000

Secret

µService C

µservicec.svc.storefront-staging.cluster.local:9080

Secret

µService B

µserviceb.svc.storefront-dev.cluster.local:3000

Runtime Component Operator

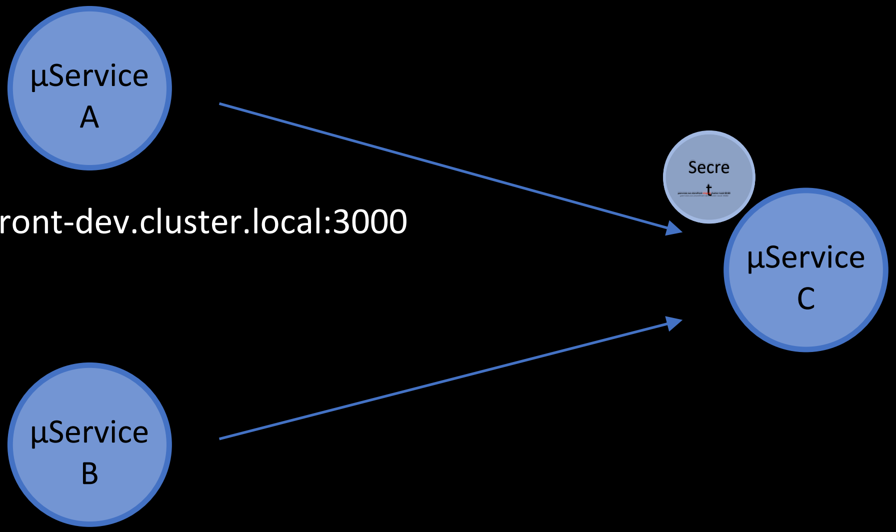
Red Hat OpenShift

Development Environment: “storefront-staging”

Service Binding

Service Discovery and Dynamic Configuration

Topology Viewer | Application Metrics | OpenTracing | Logging



µservicea.svc.storefront-dev.cluster.local:3000

µserviceb.svc.storefront-dev.cluster.local:3000

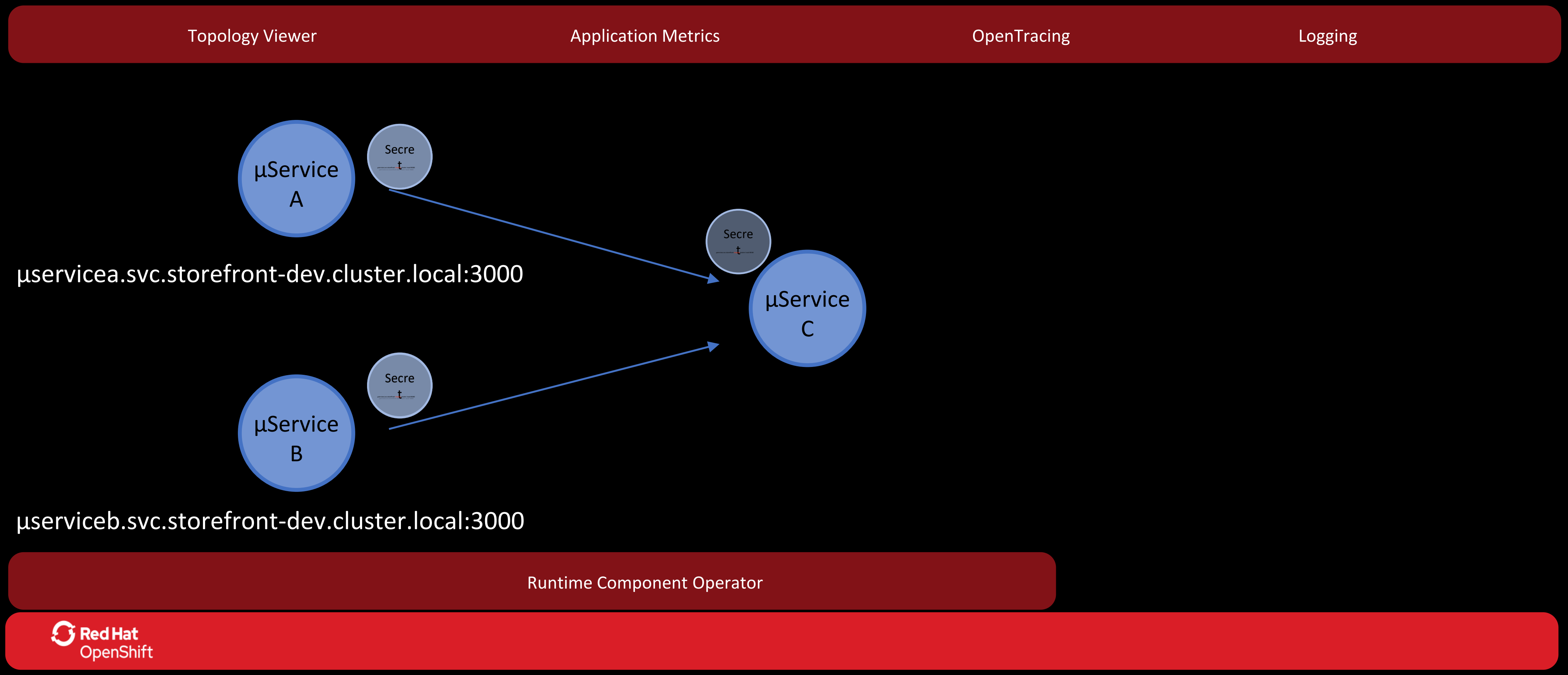
Runtime Component Operator

Red Hat OpenShift

Development Environment: “storefront-staging”

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService A

Secret

µservicea.svc.storefront-dev.cluster.local:3000

Secret

µService C

Secret

µService B

µserviceb.svc.storefront-dev.cluster.local:3000

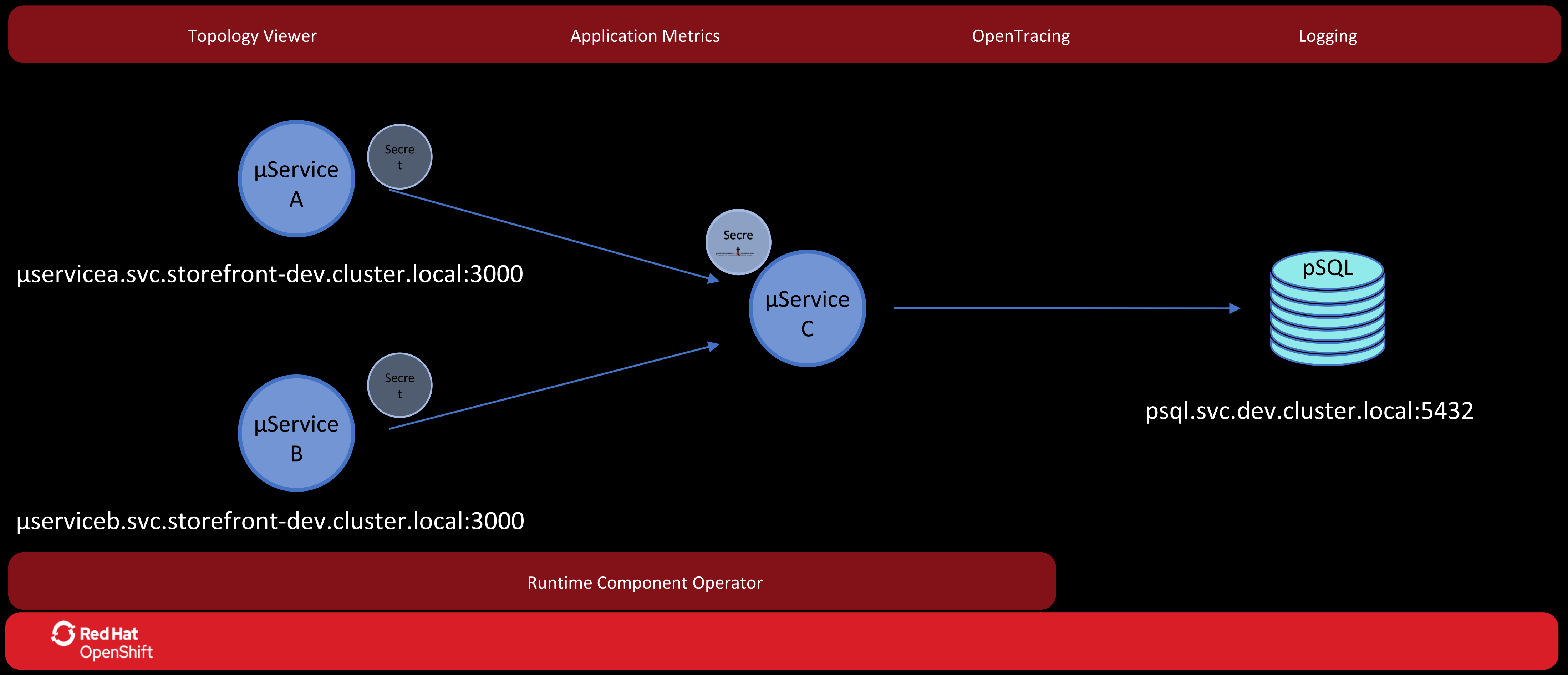
Runtime Component Operator

Red Hat OpenShift

Development Environment: "storefront-staging"

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService A

Secret

µservicea.svc.storefront-dev.cluster.local:3000

µService B

Secret

µserviceb.svc.storefront-dev.cluster.local:3000

Secret

µService C

pSQL

psql.svc.dev.cluster.local:5432

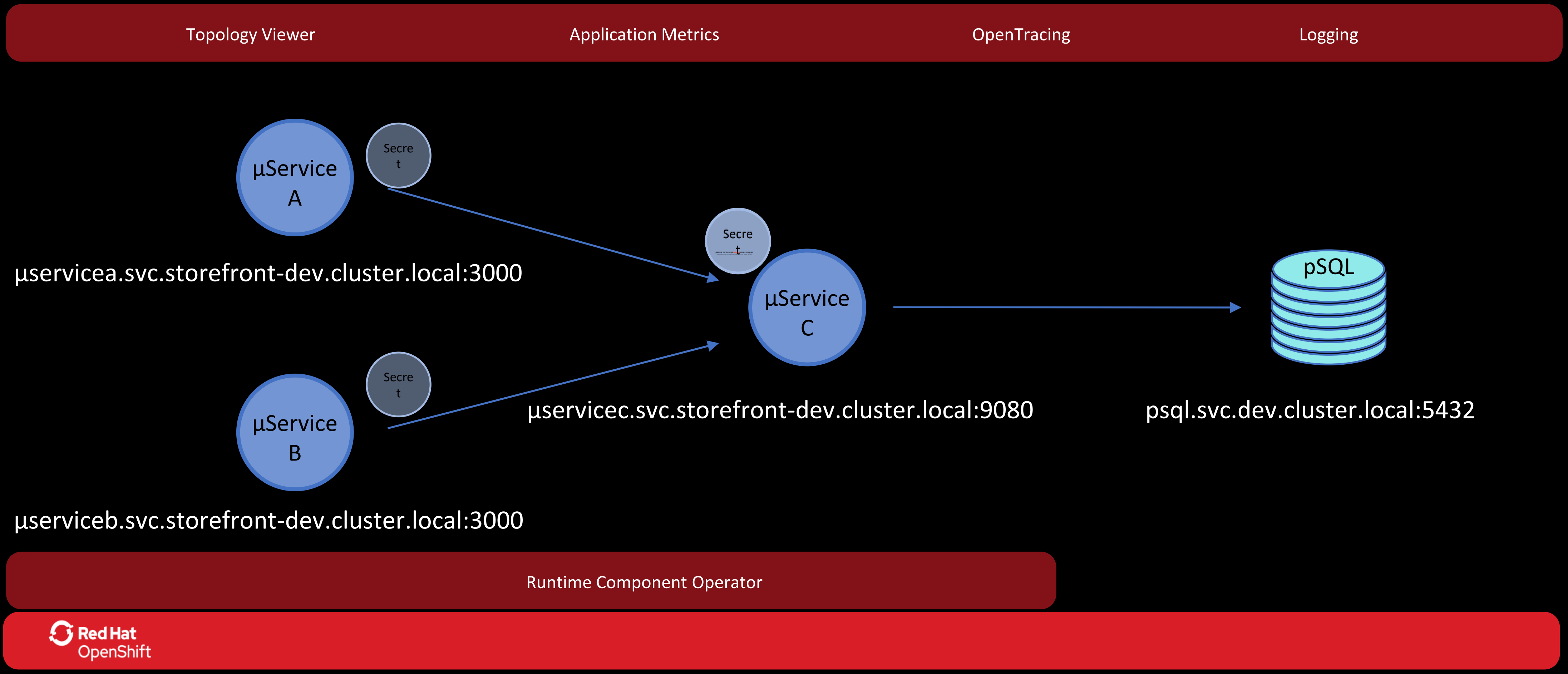
Runtime Component Operator

Red Hat OpenShift

Development Environment: “storefront-staging”

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µService A

Secret

µservicea.svc.storefront-dev.cluster.local:3000

Secret

µService C

µservicec.svc.storefront-dev.cluster.local:9080

pSQL

psql.svc.dev.cluster.local:5432

Secret

µService B

µserviceb.svc.storefront-dev.cluster.local:3000

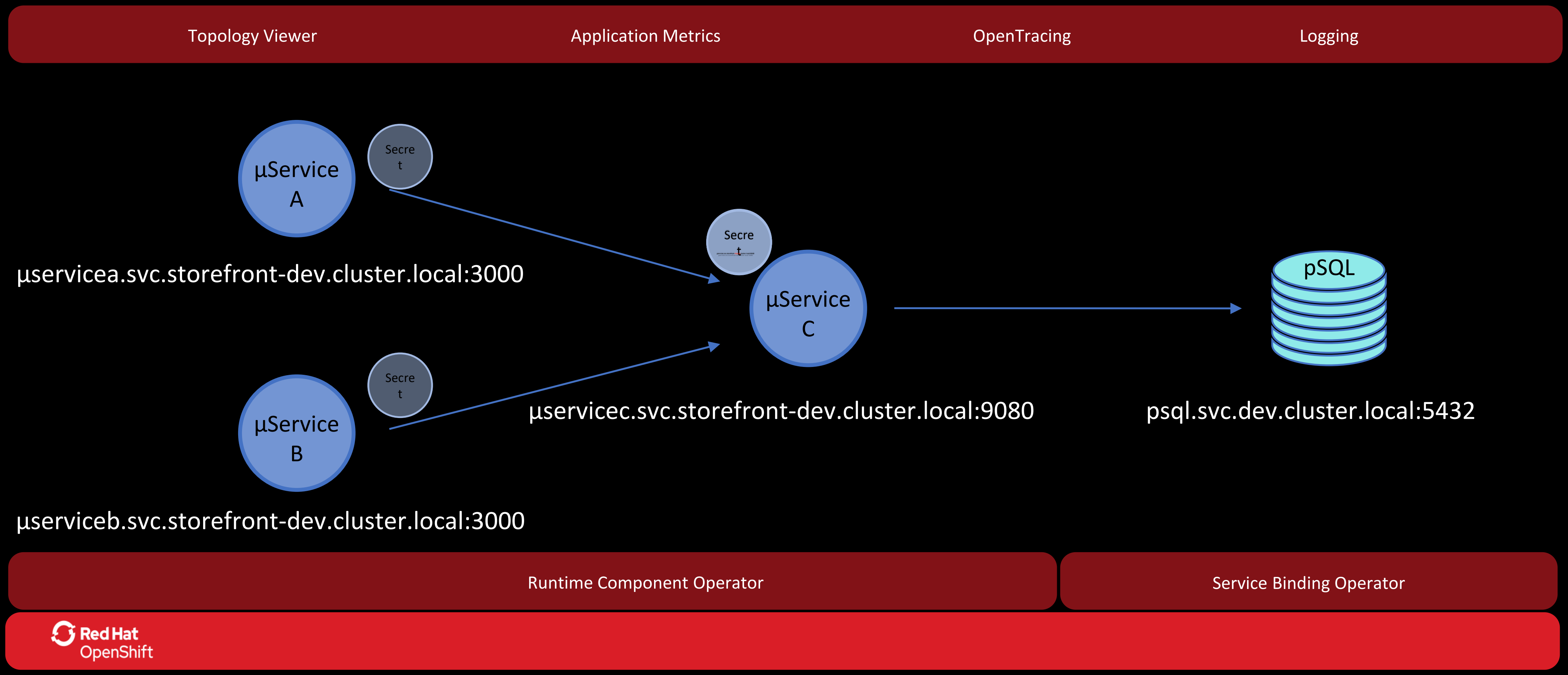
Runtime Component Operator

Red Hat OpenShift

Development Environment: “storefront-dev”

Service Binding

Service Discovery and Dynamic Configuration



Topology Viewer

Application Metrics

OpenTracing

Logging

µservicea.svc.storefront-dev.cluster.local:3000

µservicec.svc.storefront-dev.cluster.local:9080

psql.svc.dev.cluster.local:5432

µserviceb.svc.storefront-dev.cluster.local:3000

Runtime Component Operator

Service Binding Operator

Red Hat OpenShift

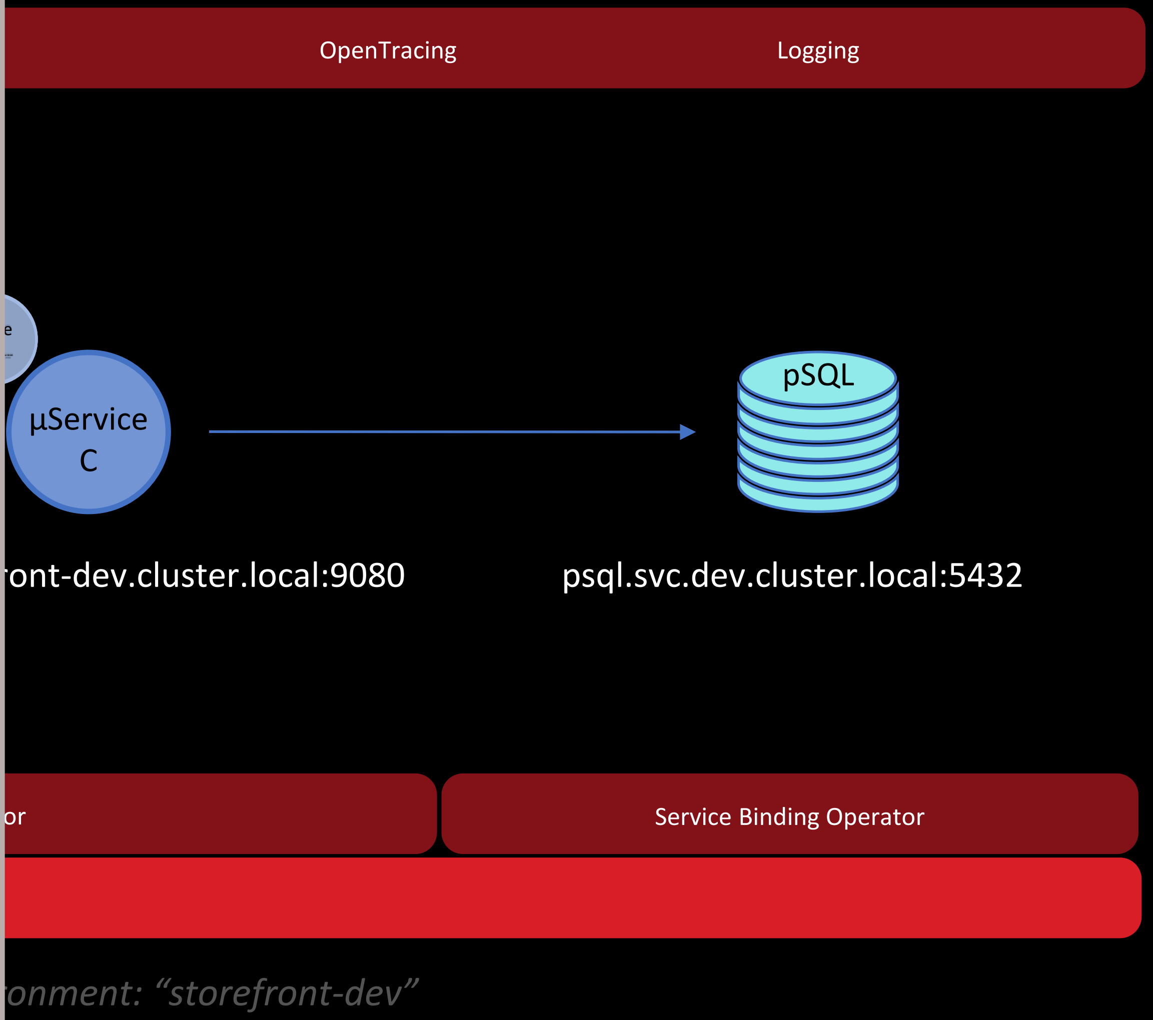
Development Environment: "storefront-dev"

Service Binding

Service Discovery and Dynamic Configuration

ServiceBindingRequest for µService C

```
apiVersion: apps.openshift.io/v1alpha1
kind: ServiceBindingRequest
metadata:
  name: µservicec-psql
spec:
  applicationSelector:
    group: apps
    resource: deployments
    resourceRef: µservicec
    version: v1
  backingServiceSelector:
    group: postgresql.baiju.dev
    version: v1alpha1
    kind: Database
    resourceRef: psql
```

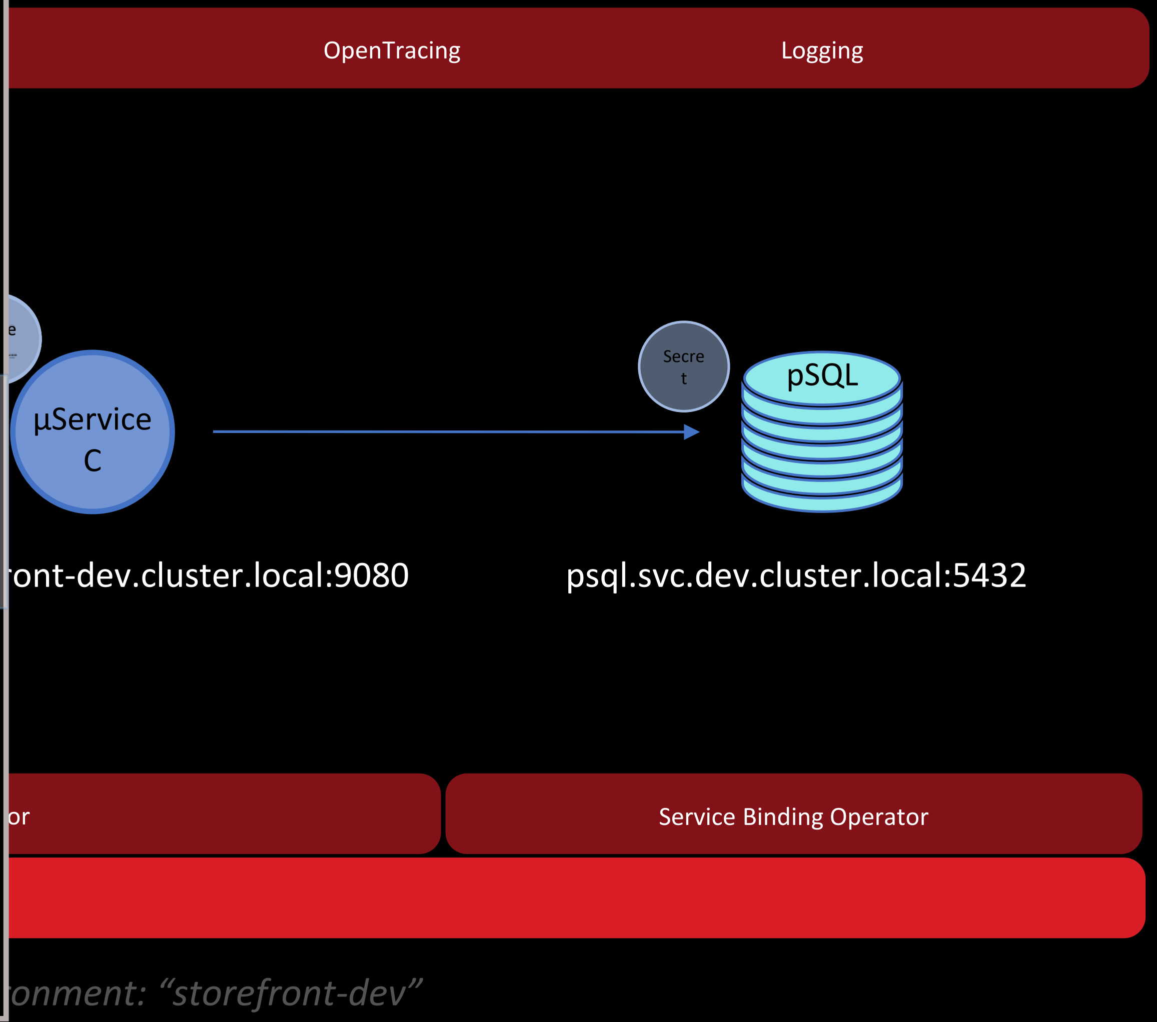


Service Binding

Service Discovery and Dynamic Configuration

ServiceBindingRequest for µService C

```
apiVersion: apps.openshift.io/v1alpha1
kind: ServiceBindingRequest
metadata:
 name: µservicec-psql
spec:
 applicationSelector:
  group: apps
  resource: deployments
  resourceRef: µservicec
  version: v1
 backingServiceSelector:
  group: postgresql.baiju.dev
  version: v1alpha1
  kind: Database
  resourceRef: psql
```

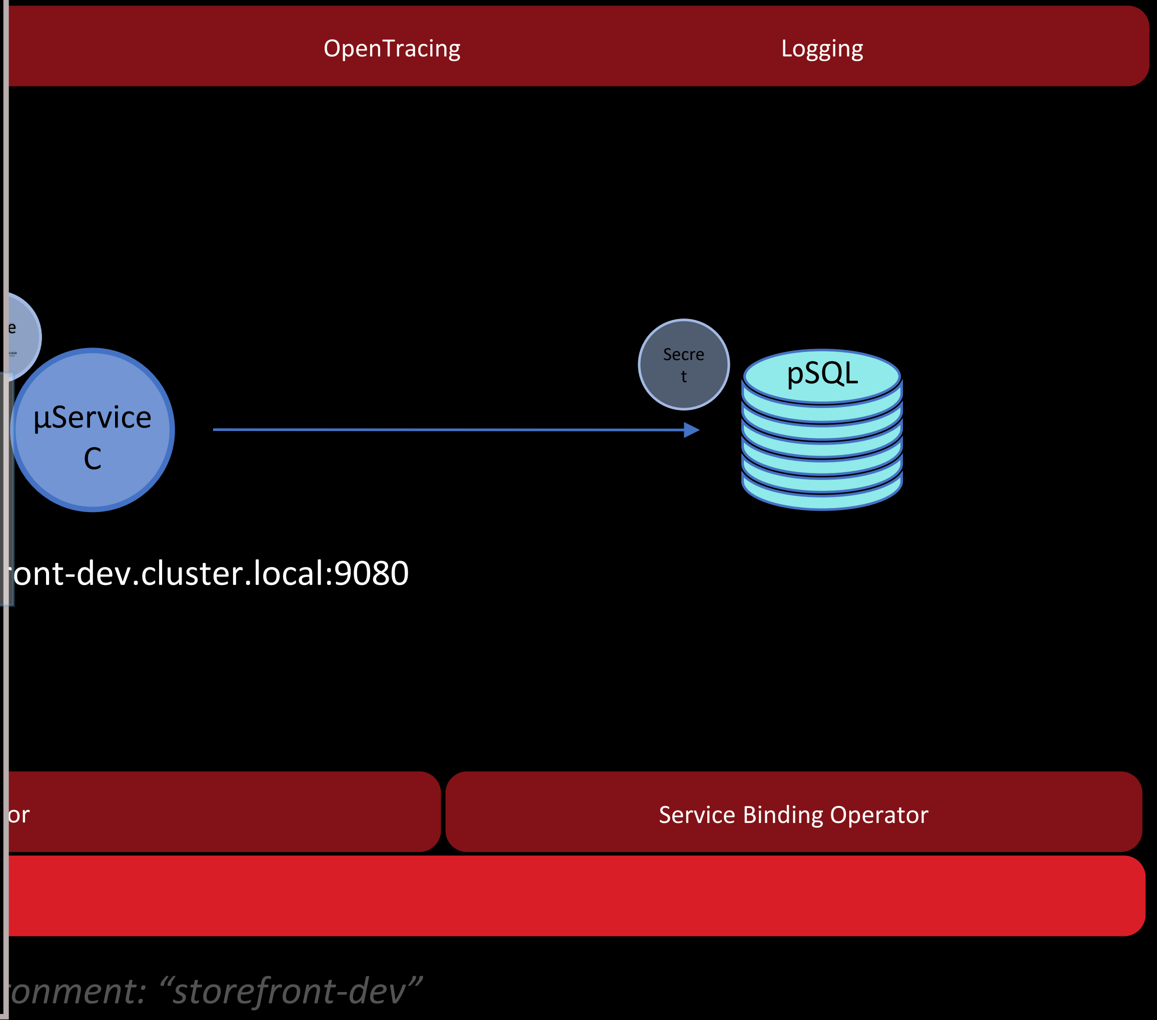


Service Binding

Service Discovery and Dynamic Configuration

ServiceBindingRequest for µService C

```
apiVersion: apps.openshift.io/v1alpha1
kind: ServiceBindingRequest
metadata:
 name: µservicec-psql
spec:
 applicationSelector:
   group: apps
   resource: deployments
   resourceRef: µservicec
   version: v1
 backingServiceSelector:
   group: postgresql.baiju.dev
   version: v1alpha1
   kind: Database
   resourceRef: psql
```

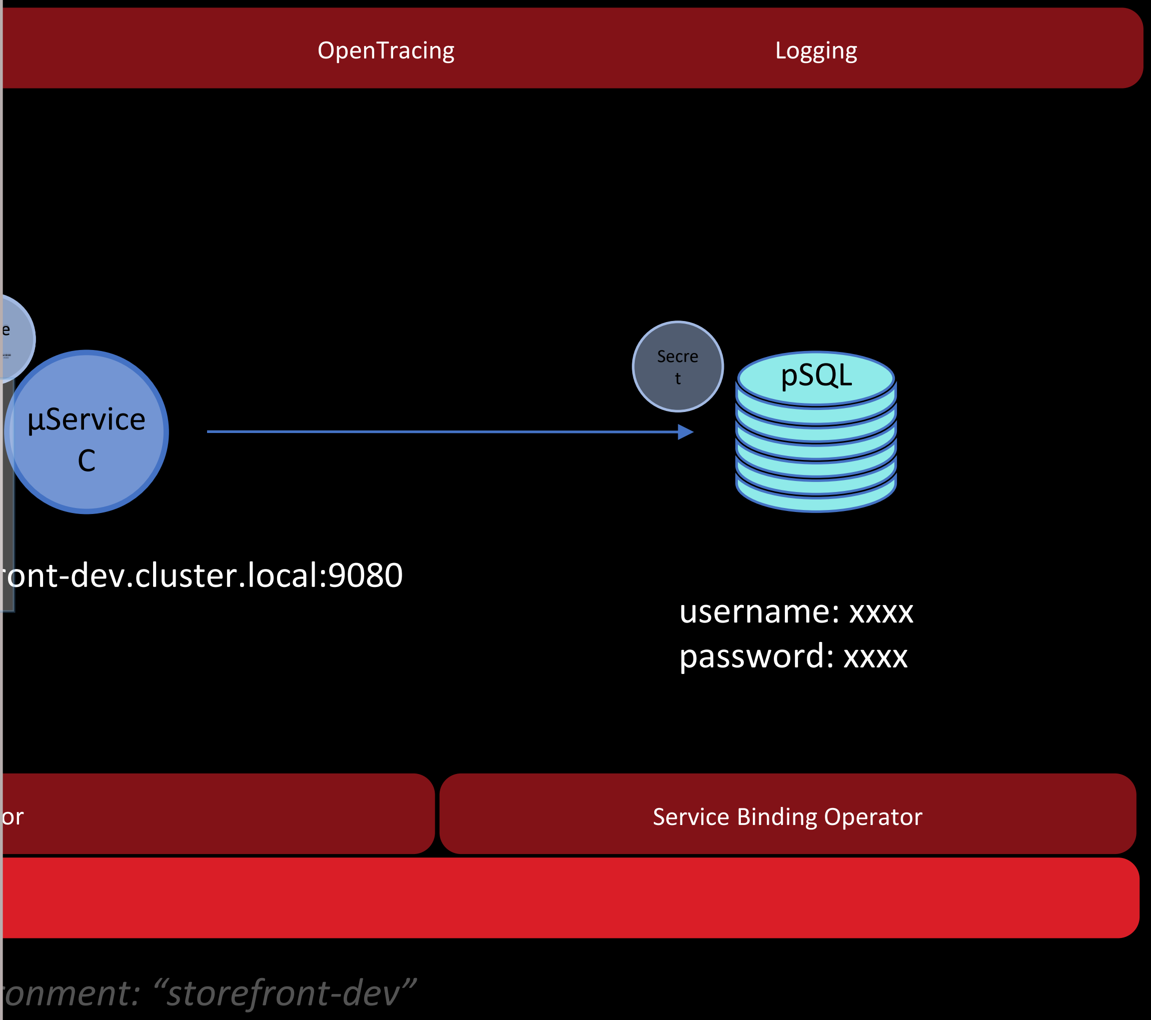


Service Binding

Service Discovery and Dynamic Configuration

ServiceBindingRequest for µService C

```
apiVersion: apps.openshift.io/v1alpha1
kind: ServiceBindingRequest
metadata:
 name: µservicec-psql
spec:
 applicationSelector:
   group: apps
   resource: deployments
   resourceRef: µservicec
   version: v1
 backingServiceSelector:
   group: postgresql.baiju.dev
   version: v1alpha1
   kind: Database
   resourceRef: catalog-db
```

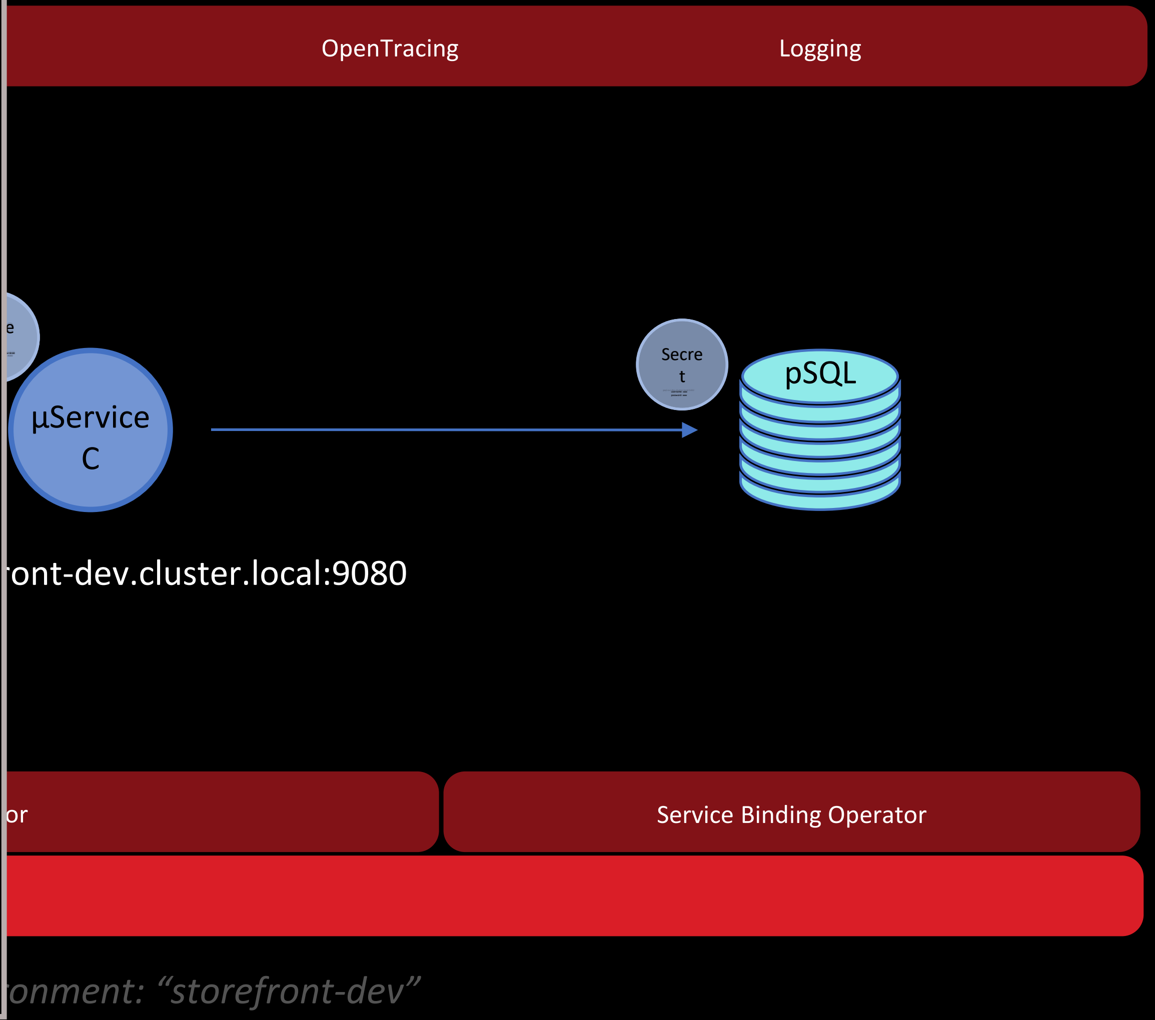


Service Binding

Service Discovery and Dynamic Configuration

ServiceBindingRequest for µService C

```
apiVersion: apps.openshift.io/v1alpha1
kind: ServiceBindingRequest
metadata:
  name: µservicec-psql
spec:
  applicationSelector:
    group: apps
    resource: deployments
    resourceRef: µservicec
    version: v1
  backingServiceSelector:
    group: postgresql.baiju.dev
    version: v1alpha1
    kind: Database
    resourceRef: catalog-db
```

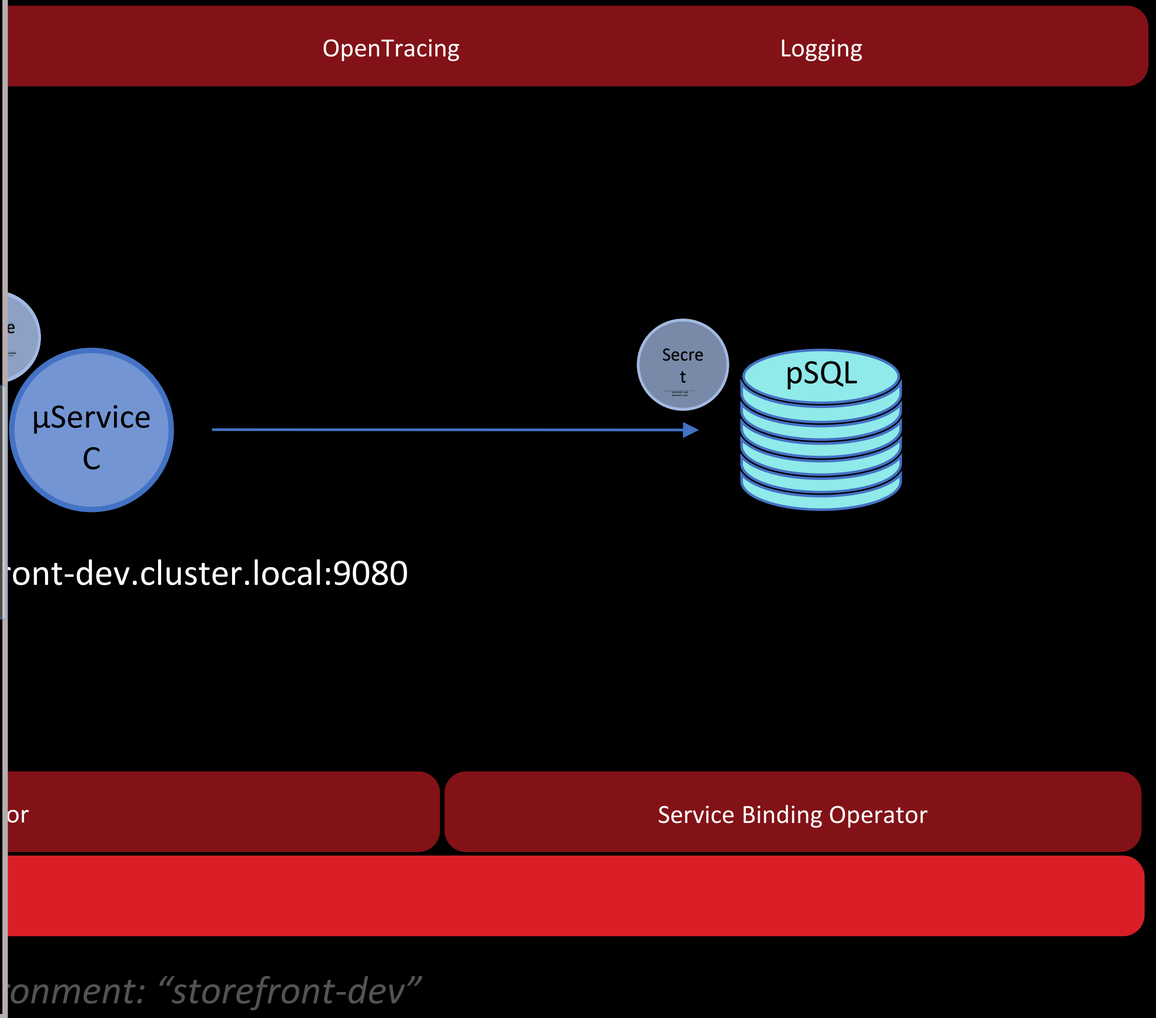


Service Binding

Service Discovery and Dynamic Configuration

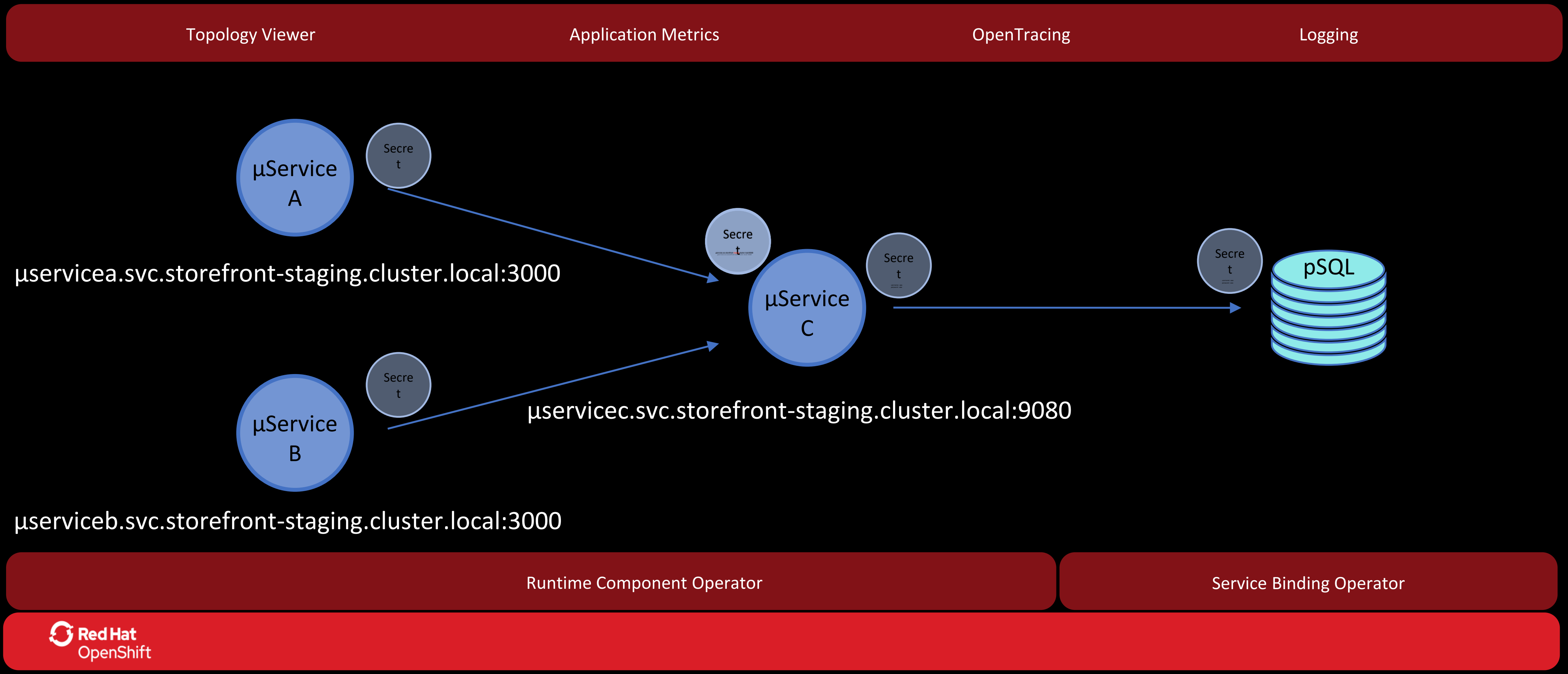
ServiceBindingRequest for µService C

```
apiVersion: apps.openshift.io/v1alpha1
kind: ServiceBindingRequest
metadata:
  name: µservicec-psql
spec:
  applicationSelector:
    group: apps
    resource: deployments
    resourceRef: µservicec
    version: v1
  backingServiceSelector:
    group: postgresql.baiju.dev
    version: v1alpha1
    kind: Database
    resourceRef: catalog-db
```



Service Binding

Service Discovery and Dynamic Configuration


Topology Viewer
Application Metrics
OpenTracing
Logging
µService A
Secre t
µservicea.svc.storefront-staging.cluster.local:3000
Secre t
µService C
Secre t
Secre t
pSQL
Secre t
µService B
µservicec.svc.storefront-staging.cluster.local:9080
µserviceb.svc.storefront-staging.cluster.local:3000
Runtime Component Operator
Service Binding Operator
Red Hat OpenShift

Development Environment: “storefront-staging”

Design Led Acceleration Workflow

IBM Garage Services
IBM Expert Labs

Ideate

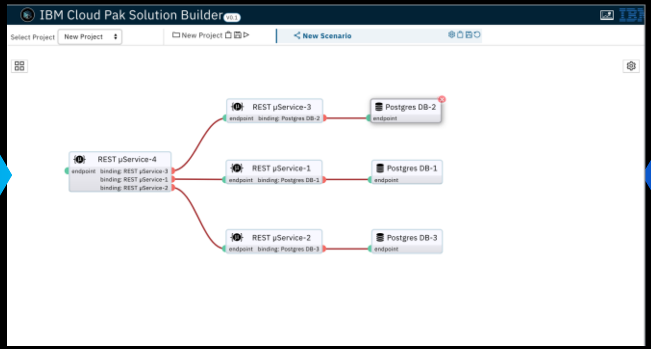
Solution Architect

Business Analyst

Business Objectives & Solution Concept

Ideate

Solution Builder



IBM Cloud Pak Solution Builder v0.1

Select Project: New Project

New Project

New Scenario

Solution Design & Architecture

Accelerator Content

Solution Patterns

Runtime Stacks

Operator Backed Services

- Reference Architecture Patterns
  - Start from pattern and edit
  - Start from blank canvas
- Pre-built templates and stacks
  - Domain specific code content
- OSS, Red Hat and IBM Services
  - Operator enabled
  - Service Binding support

Expert and Best-Practise Defined Content

Create

µService A — Verify and Build Pipeline

µService B — Verify and Build Pipeline

µService C — Verify and Build Pipeline

µService D — Verify and Build Pipeline

git
Source Repos

Built on OpenShift Pipelines

GitOps

git
Source Repos

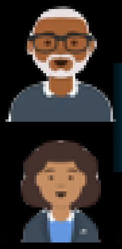
Design Led Acceleration Workflow

IBM Garage Services
IBM Expert Labs

Ideate

Solution Architect

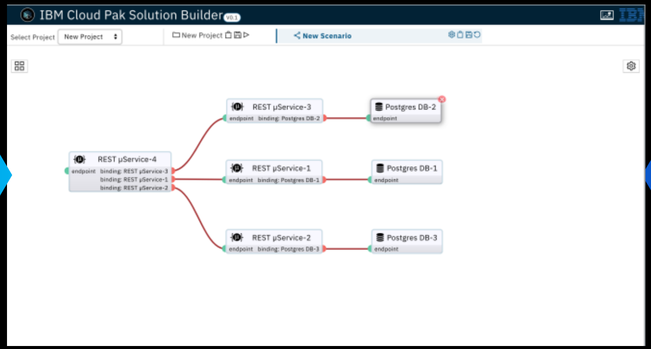
Business Analyst



Business Objectives
& Solution Concept

Ideate

Solution Builder



Solution Design & Architecture

Accelerator Content

Solution Patterns

Runtime Stacks

Operator Backed Services

- Reference Architecture Patterns
  - Start from pattern and edit
  - Start from blank canvas
- Pre-built templates and stacks
  - Domain specific code content
- OSS, Red Hat and IBM Services
  - Operator enabled
  - Service Binding support

Expert and Best-Practise Defined Content

Create

µService A — Verify and Build Pipeline — µS A

µService B — Verify and Build Pipeline — µS B

µService C — Verify and Build Pipeline — µS C

µService D — Verify and Build Pipeline — µS D

git
Source Repos

Built on OpenShift Pipelines

GitOps

git
Source Repos

Design Led Acceleration Workflow

IBM Garage Services
IBM Expert Labs

Ideate

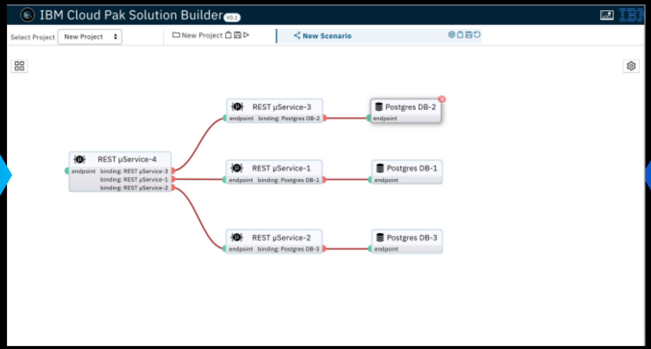
Solution Architect

Business Analyst

Business Objectives & Solution Concept

Ideate

Solution Builder



Solution Design & Architecture

Accelerator Content

Solution Patterns

Runtime Stacks

Operator Backed Services

- Reference Architecture Patterns
  - Start from pattern and edit
  - Start from blank canvas
- Pre-built templates and stacks
  - Domain specific code content
- OSS, Red Hat and IBM Services
  - Operator enabled
  - Service Binding support

Expert and Best-Practise Defined Content

Create

µService A — Verify and Build Pipeline

µService B — Verify and Build Pipeline

µService C — Verify and Build Pipeline

µService D — Verify and Build Pipeline

git
Source Repos

Built on OpenShift Pipelines

GitOps

git
Source Repos

Design Led Acceleration Workflow

IBM Garage Services
IBM Expert Labs

Ideate

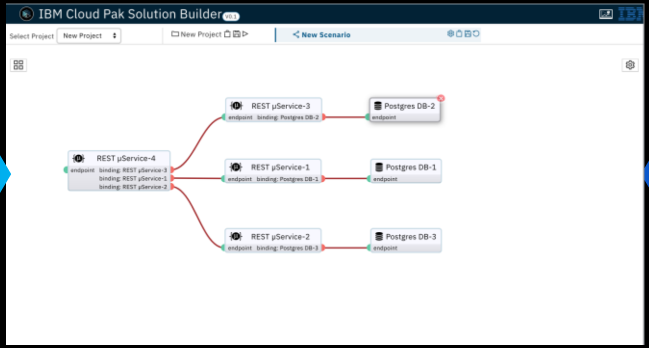
Solution
Architect

Business
Analyst

Business Objectives
& Solution Concept

Ideate

Solution Builder



Solution Design & Architecture

Accelerator Content

Solution
Patterns

Runtime
Stacks

Operator
Backed Services

- Reference Architecture Patterns
  - Start from pattern and edit
  - Start from blank canvas
- Pre-built templates and stacks
  - Domain specific code content
- OSS, Red Hat and IBM Services
  - Operator enabled
  - Service Binding support

Expert and Best-Practise Defined Content

Create

µService A — Verify and Build Pipeline

µService B — Verify and Build Pipeline

µService C — Verify and Build Pipeline

µService D — Verify and Build Pipeline

git
Source Repos

Built on OpenShift
Pipelines

GitOps

git
Source Repos

Deploy

Dashboard | Metrics | OpenTracing | Logging

Runtime Component Operator

Service Binding
Operator

Red Hat
OpenShift

Development, Staging and Production Environments

Design Led Acceleration Workflow

IBM Garage Services
IBM Expert Labs

Ideate

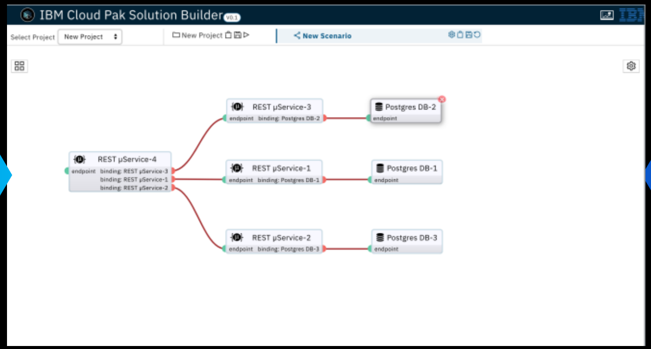
Solution Architect

Business Analyst

Business Objectives
& Solution Concept

Ideate

Solution Builder



Solution Design & Architecture

Accelerator Content

Solution Patterns

Runtime Stacks

Operator Backed Services

- Reference Architecture Patterns
  - Start from pattern and edit
  - Start from blank canvas
- Pre-built templates and stacks
  - Domain specific code content
- OSS, Red Hat and IBM Services
  - Operator enabled
  - Service Binding support

Expert and Best-Practise Defined Content

Create

Developer

Developer

Developer

Developer

Iterate

µService A

Verify and Build Pipeline

µService B

Verify and Build Pipeline

µService C

Verify and Build Pipeline

µService D

Verify and Build Pipeline

git
Source Repos

Built on OpenShift Pipelines

GitOps

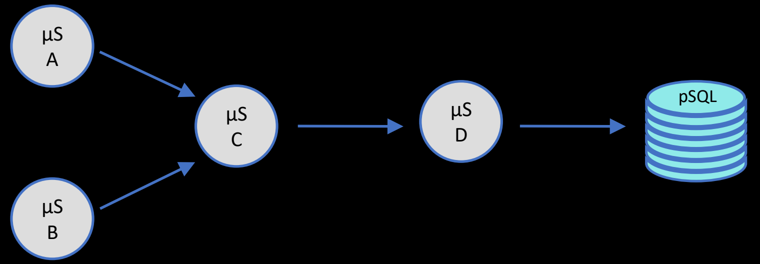
git
Source Repos

Operate

Admin &
Operations

Deploy

Dashboard | Metrics | OpenTracing | Logging



Runtime Component Operator

Service Binding Operator

Red Hat
OpenShift

Development, Staging and Production Environments

It’s time for real code in action!

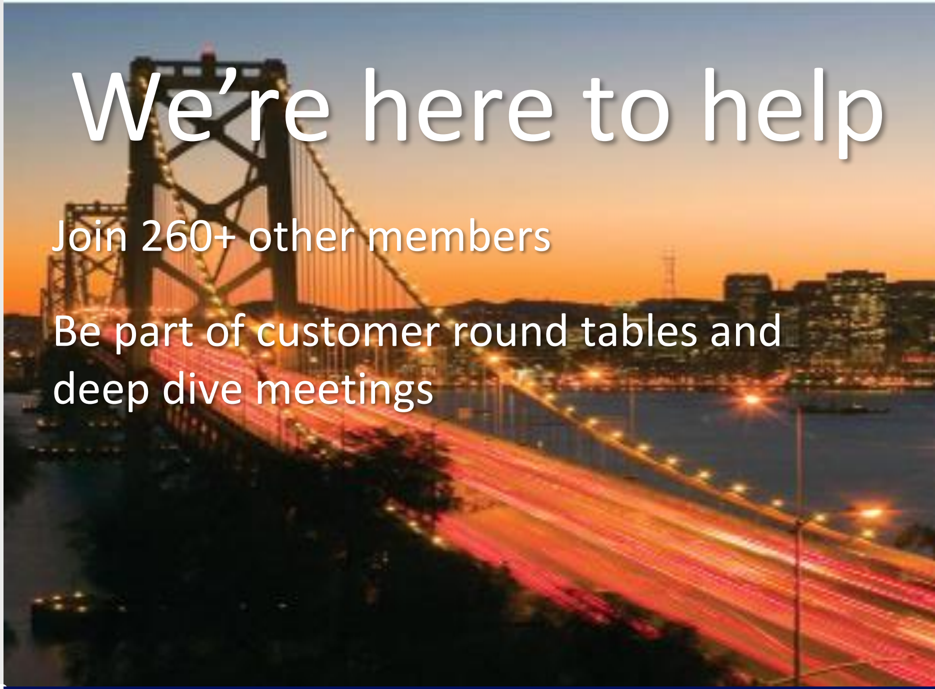
Demo

Join the CAB

WebSphere and Cloud Pak for Applications

Customer Advisory Board

claudiab@us.ibm.com
https://ibm.webex.com/meet/claudiab
http://ibm.biz/WASCABCommunityResources
http://ibm.biz/WebSphereAdvisoryBoard



We're here to help

Join 260+ other members

Be part of customer round tables and deep dive meetings

Weekly meetings	Monthly meetings	Special
Thursday and Friday 9:15 am EST JOIN	Business Partner track Other timezones JOIN	Cloud Pak Week Previews Demos Labs, workshops 1-on-1

Engage when you have time:

- ✓ Stay in the loop at meetings
- ✓ Share solutions and pain points
- ✓ Connect with other customers
- ✓ Access to resources and experts
- ✓ Customized meetings
- ✓ Special offers

Don’t miss out …

Join and follow our *User Group* community to stay informed:
http://ibm.biz/WUG-community

- Learn from Experts
- Join free webinars
- Access a library full of content
- Ask questions

Join our weekly Developer Series -
View upcoming webinars and enroll:
http://ibm.biz/WUG-dev-series

IBM