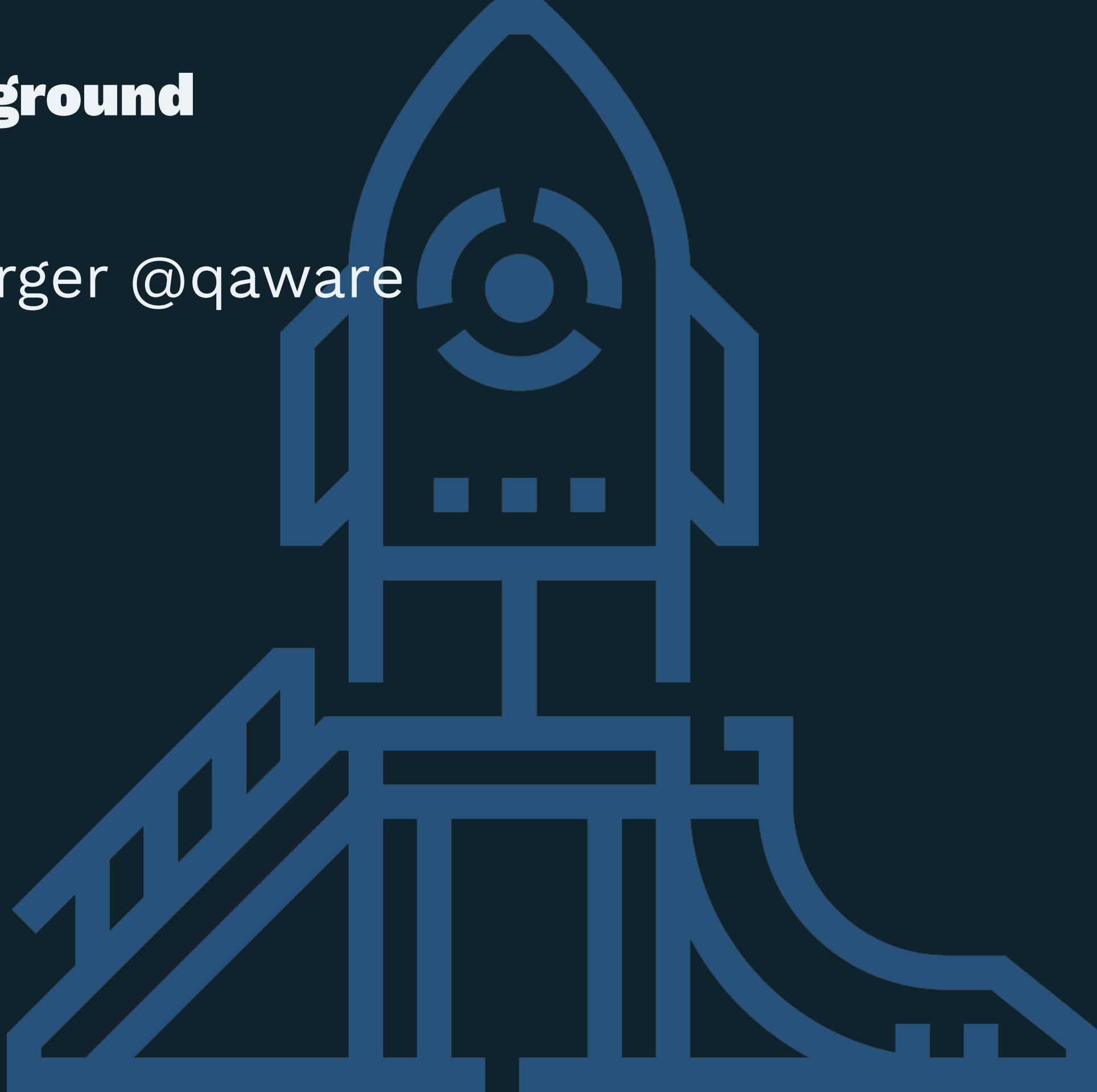


Istio Playground

@adersberger @qaware



Our network today



- Optimize first: Switch network off and on again and use 5GHz networking
- Plan A: Local installation
- Plan B: Use GKE clusters
- Plan C: Use Katacoda
- Plan D: Steamworks

Workshop Prerequisites

- Bash
- git Client
- Text editor (like VS.Code)

Baby Step: Grab the Code

```
git clone https://github.com/adersberger/istio-playground
```

```
cd istio-playground/code
```

Baby Step: Install a (local) Kubernetes Cluster

<https://www.docker.com/community-edition>

Docker
Community Edition

Version 18.04.0-ce-mac62 (23965)

- Preferences: enable Kubernetes
- Preferences: increase resource usage to 3 cores and 8 GB memory



Engine: 18.04.0-ce



Compose: 1.21.0



Machine: 0.14.0



Notary Notary: 0.6.0

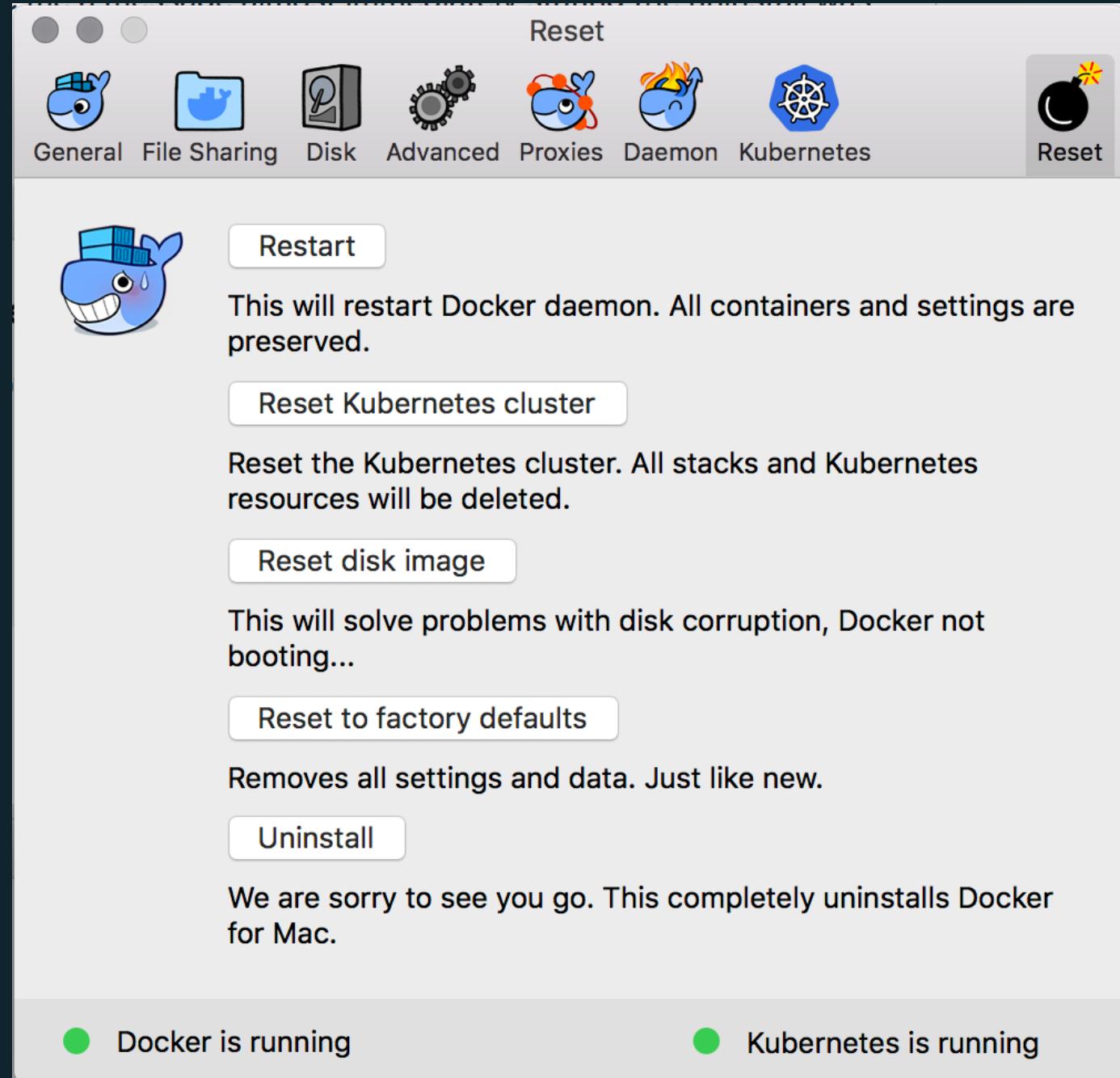


Credential Helper: 0.6.0



Kubernetes: v1.9.6

The Ultimate Guide to Fix Strange Kubernetes Behavior



Setup Kubernetes Environment

```
# Switch k8s context
kubectl config use-context docker-for-desktop
# Deploy k8s dashboard
kubectl create -f https://raw.githubusercontent.com/kubernetes/dashboard/master/src/deploy/recommended/kubernetes-dashboard.yaml
# Extract id of default service account token (referred as TOKENID)
kubectl describe serviceaccount default
# Grab token and insert it into k8s Dashboard UI auth dialog
kubectl describe secret TOKENID
# Start local proxy
kubectl proxy --port=8001 &
# Open k8s Dashboard
open http://localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/#!/login
```

Deploy Istio

```
curl -L https://git.io/getLatestIstio | sh -  
cd istio-1.0.1  
export PATH=$PWD/bin:$PATH  
istioctl version
```

```
# deploy Istio  
# (demo setting, default deployment is via Helm)  
kubectl apply -f install/kubernetes/istio-demo.yaml  
kubectl get pods -n istio-system  
  
# label default namespace to be auto-sidecarred  
kubectl label namespace default istio-injection=enabled  
kubectl get namespace -L istio-injection
```

Deploy Sample Application (BookInfo)

```
kubectl apply -f samples/bookinfo/platform/kube/bookinfo.yaml  
kubectl get pods  
istioctl create -f samples/bookinfo/networking/bookinfo-gateway.yaml  
istioctl get gateways  
open http://localhost/productpage
```

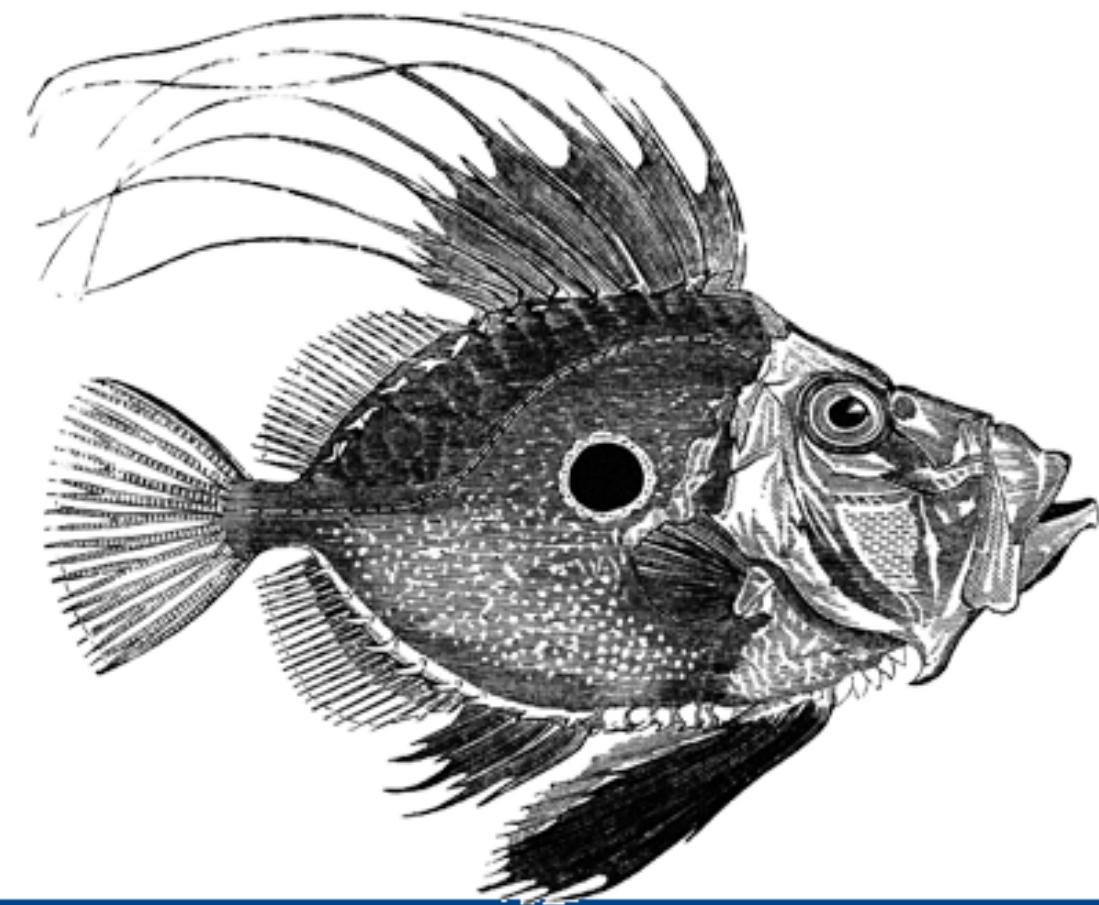
Hands-on



Why?



Fight Microservice Obesity



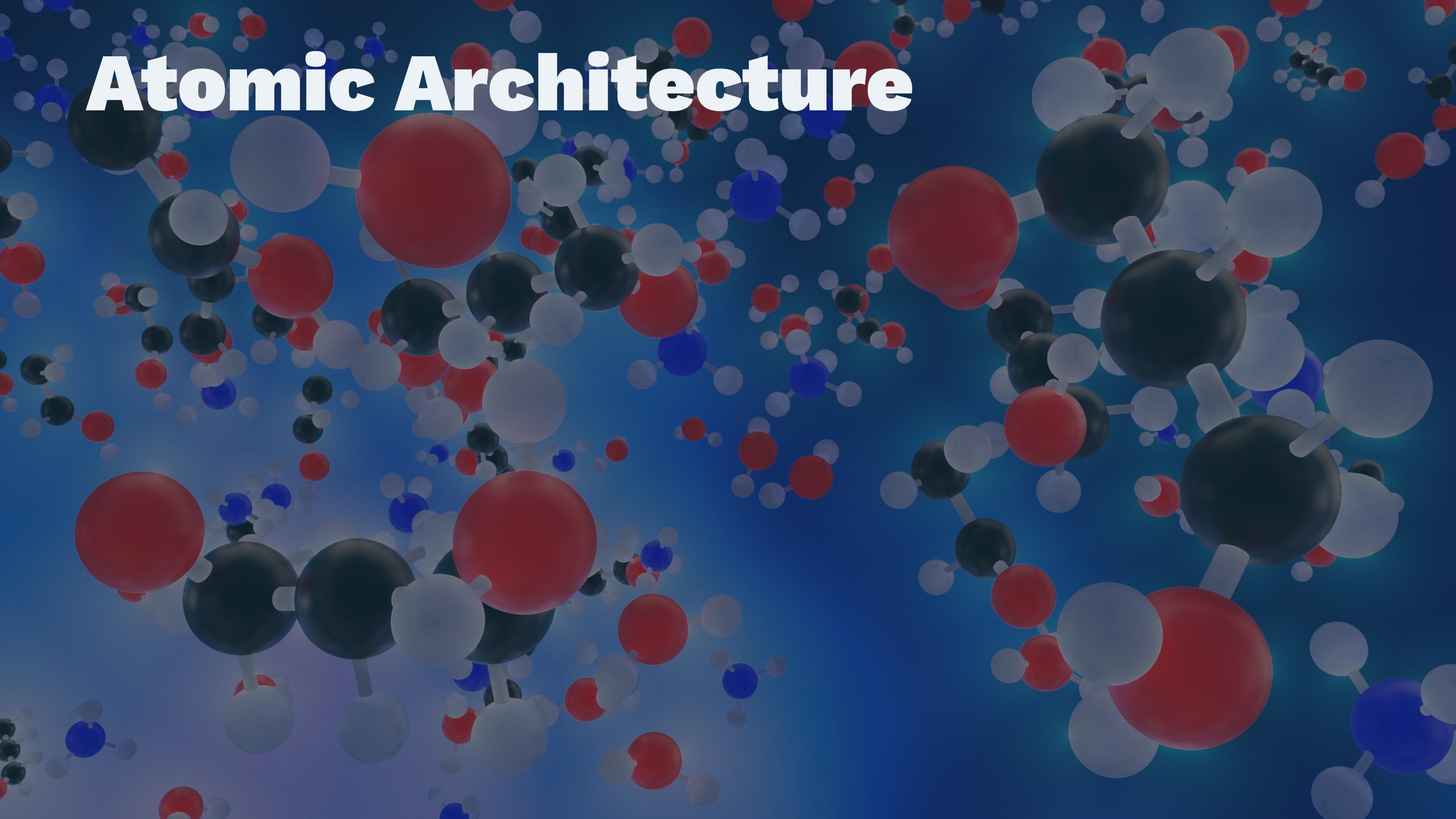
Hype-Driven Software Development

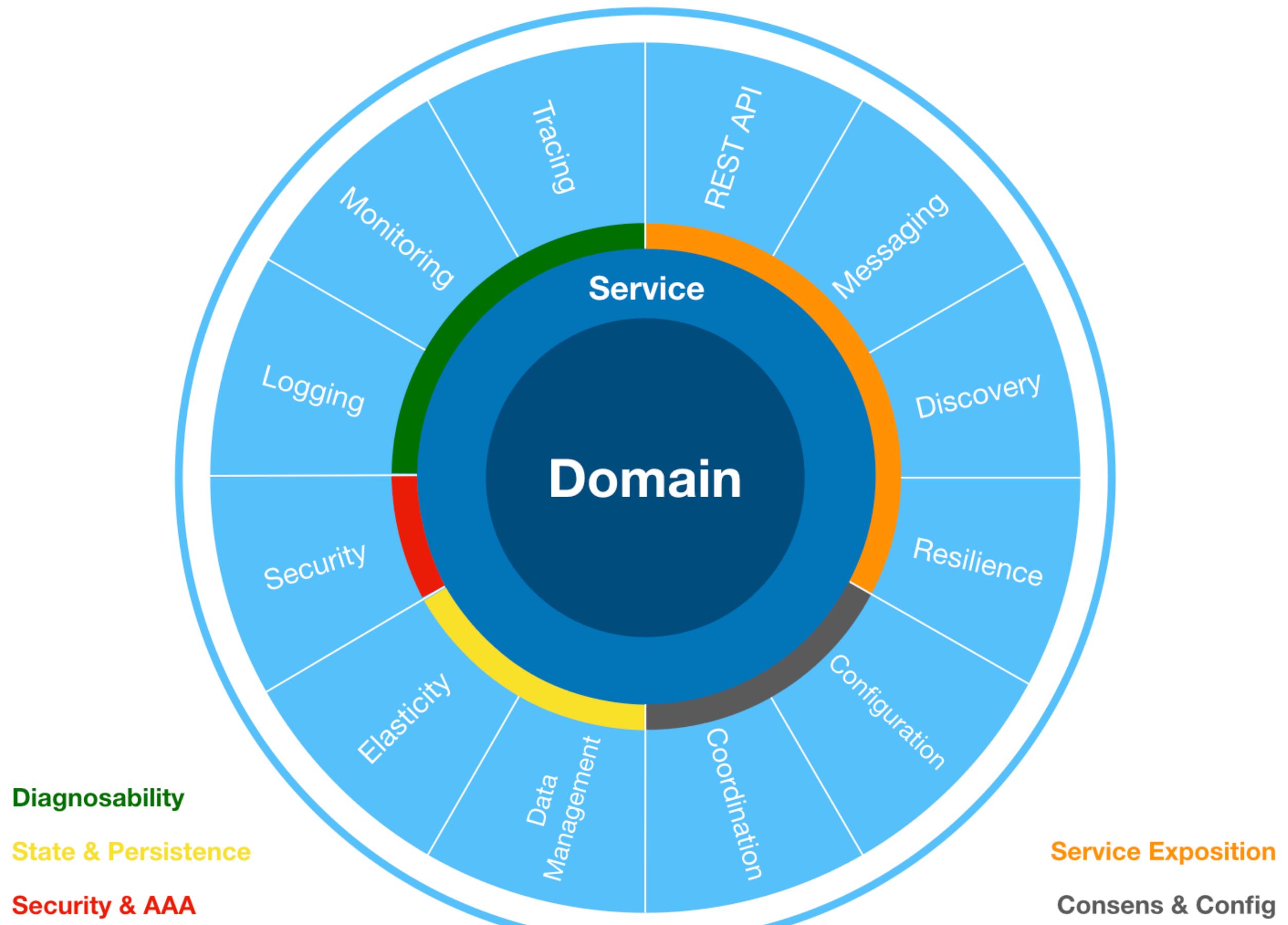
Istio Edition

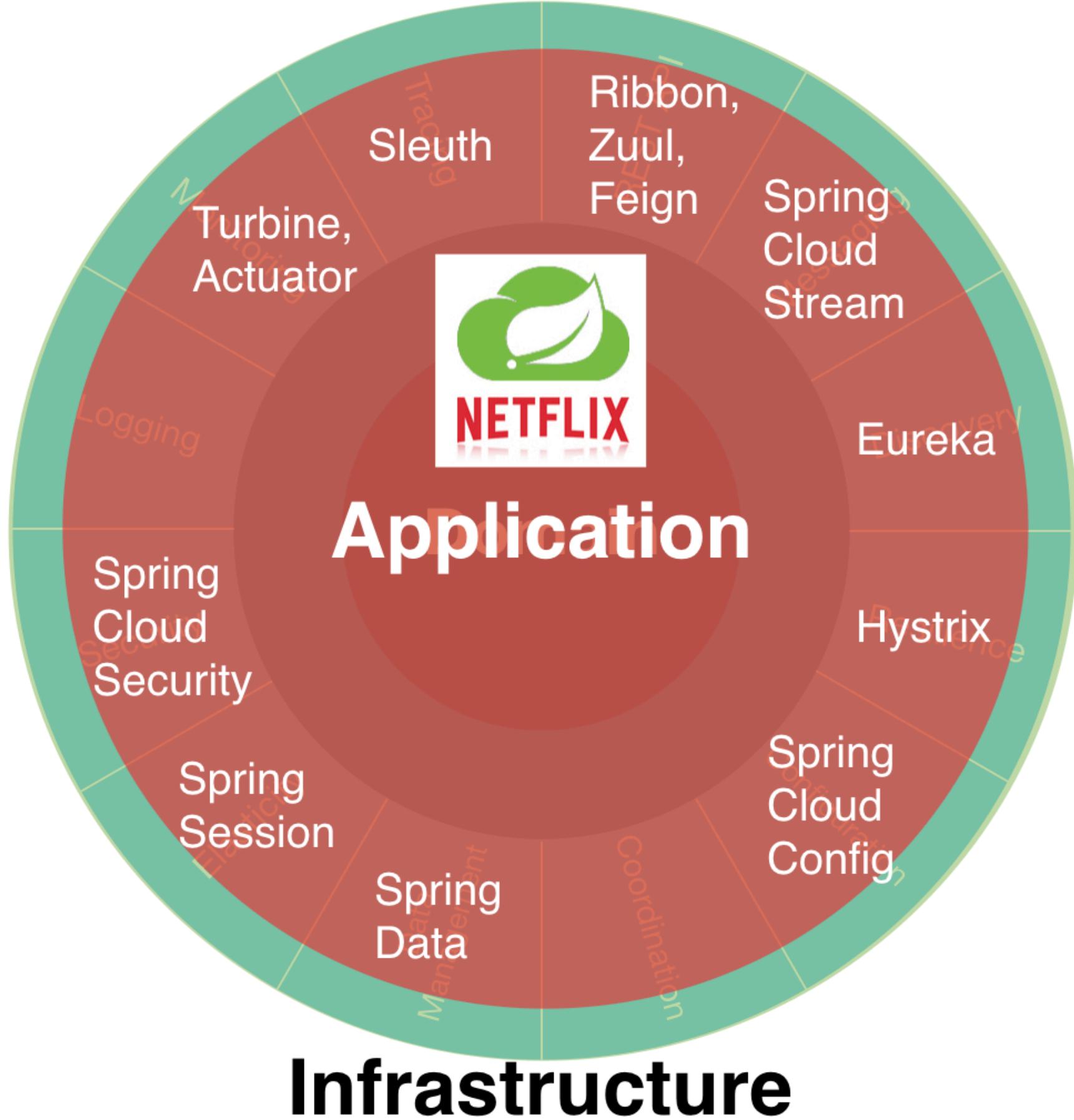
O RLY[?]

Josef Adersberger

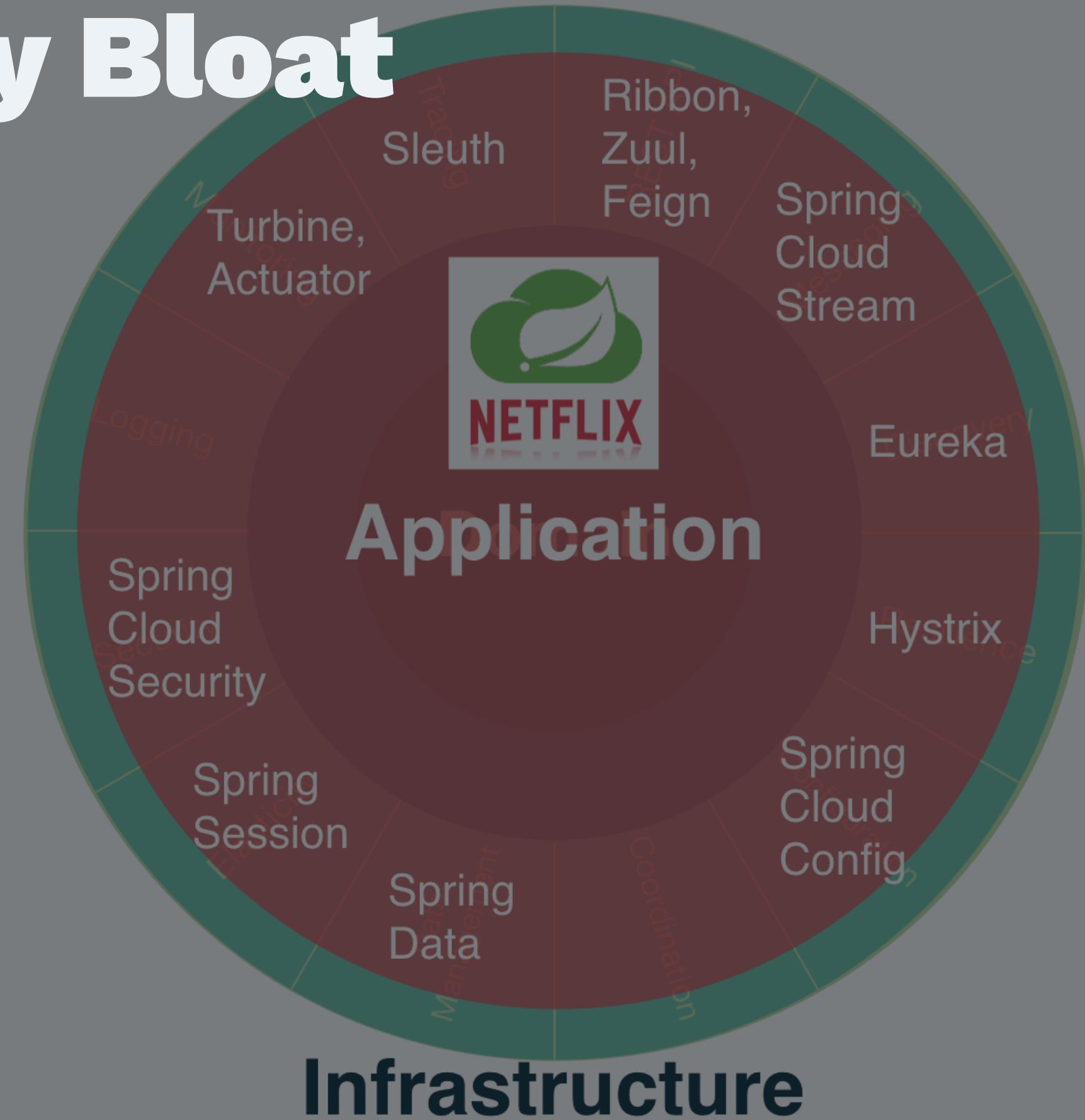
Atomic Architecture







Library Bloat



Can we evolve existing enterprise applications into the cloud with reasonable effort?



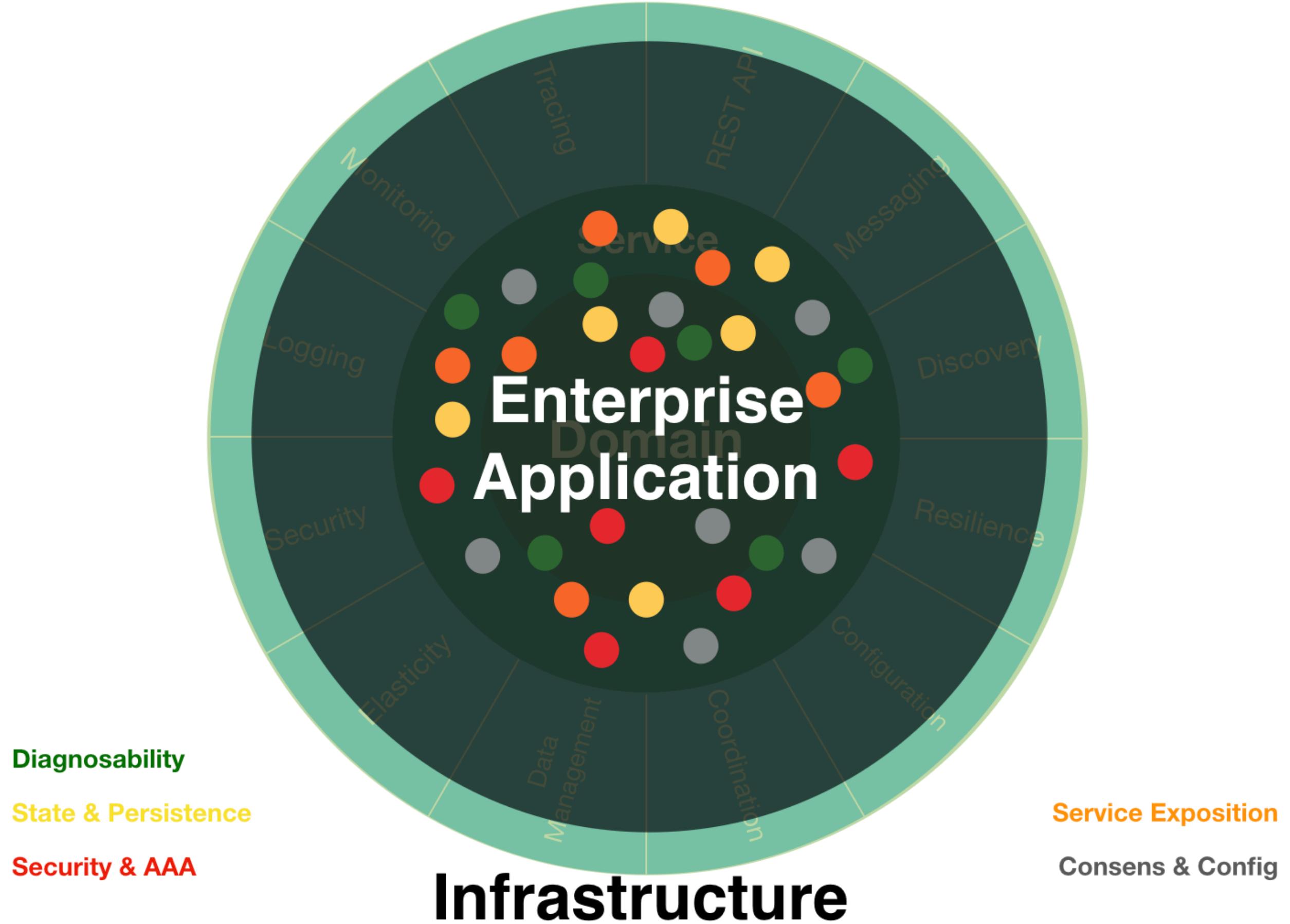
- Monolithic Deployment
- Traditional Infrastructure

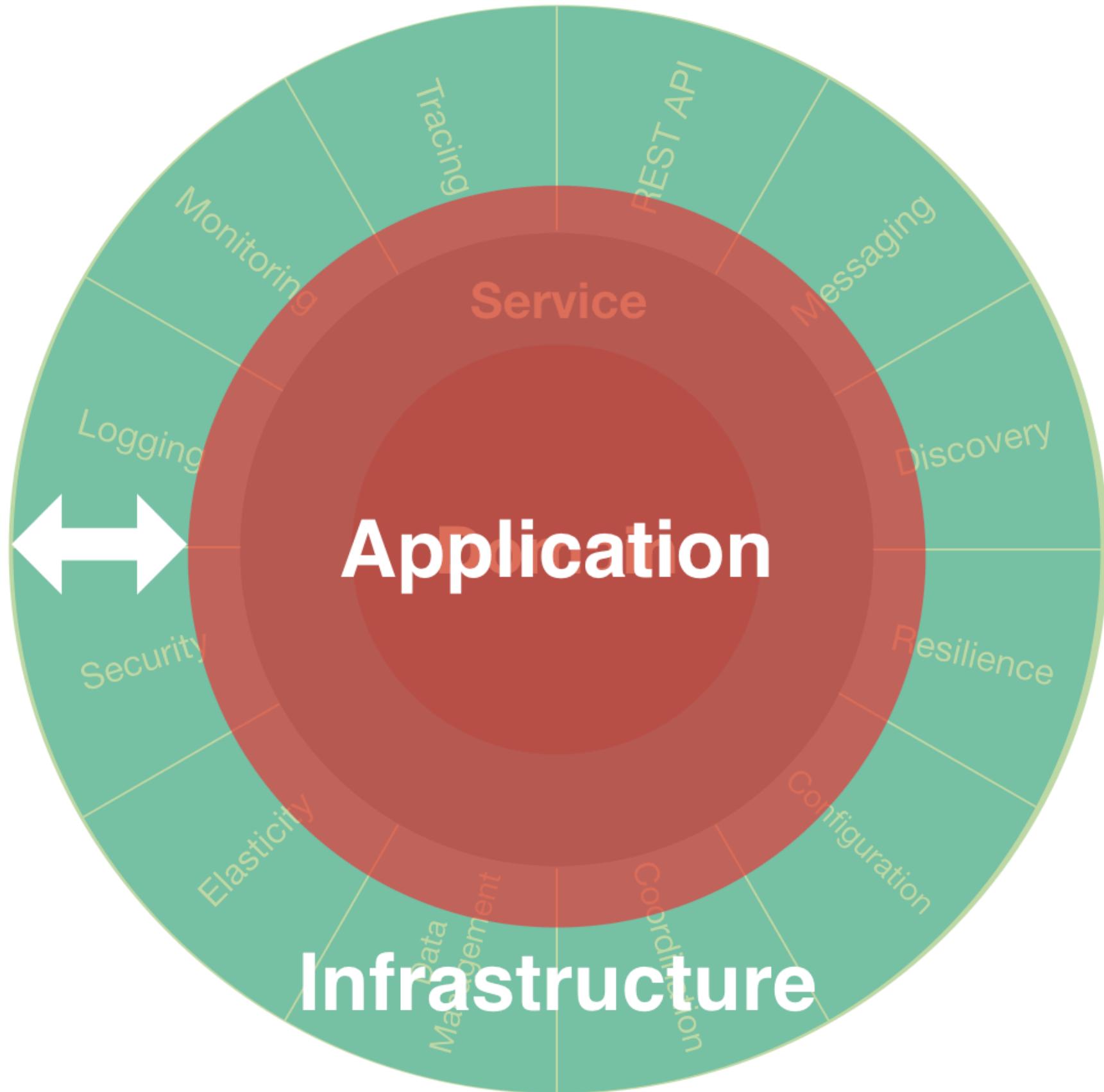


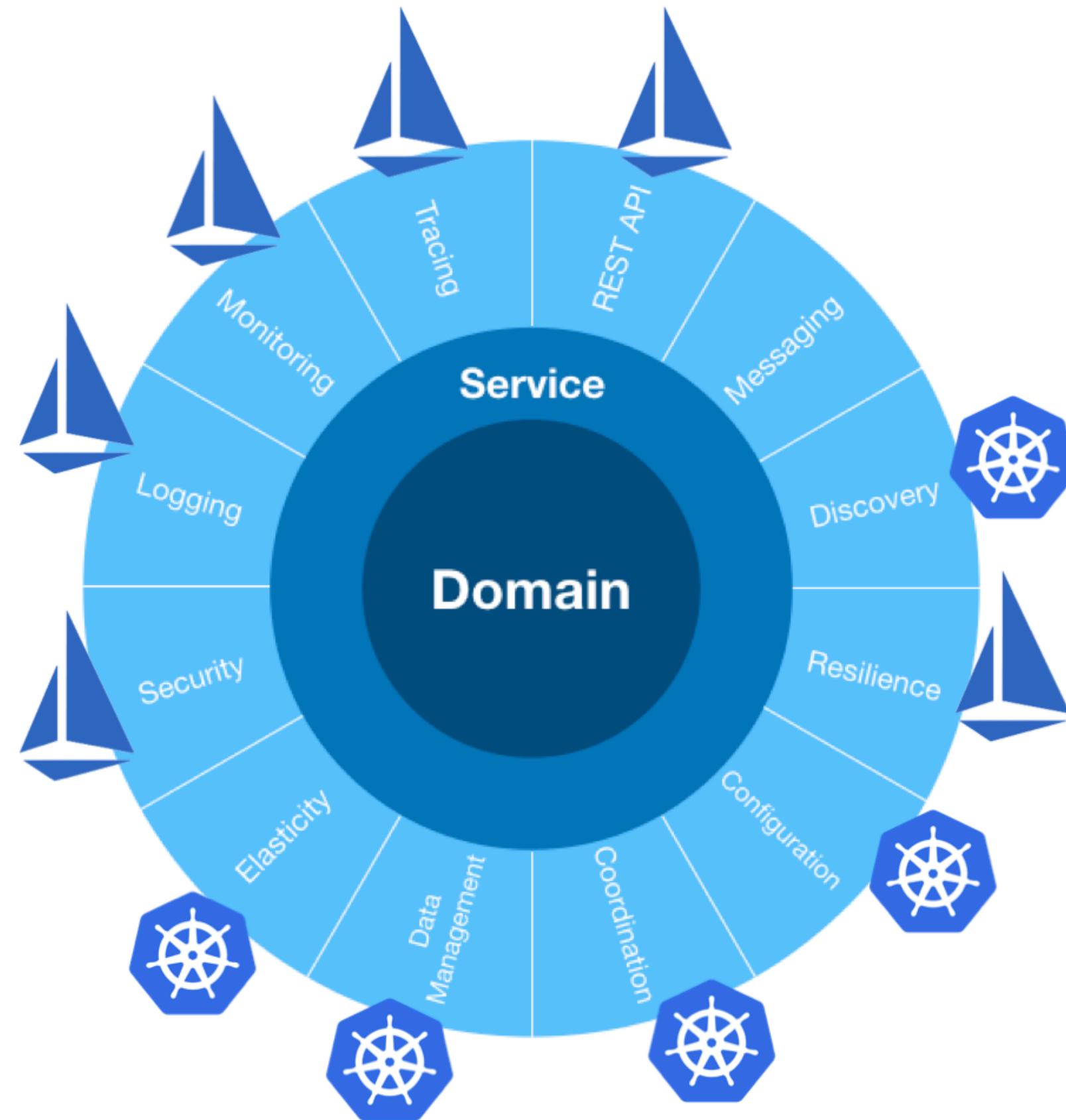
- Containerization
- 12-Factor App Principles



- Microservices
- Cloud-native Apps

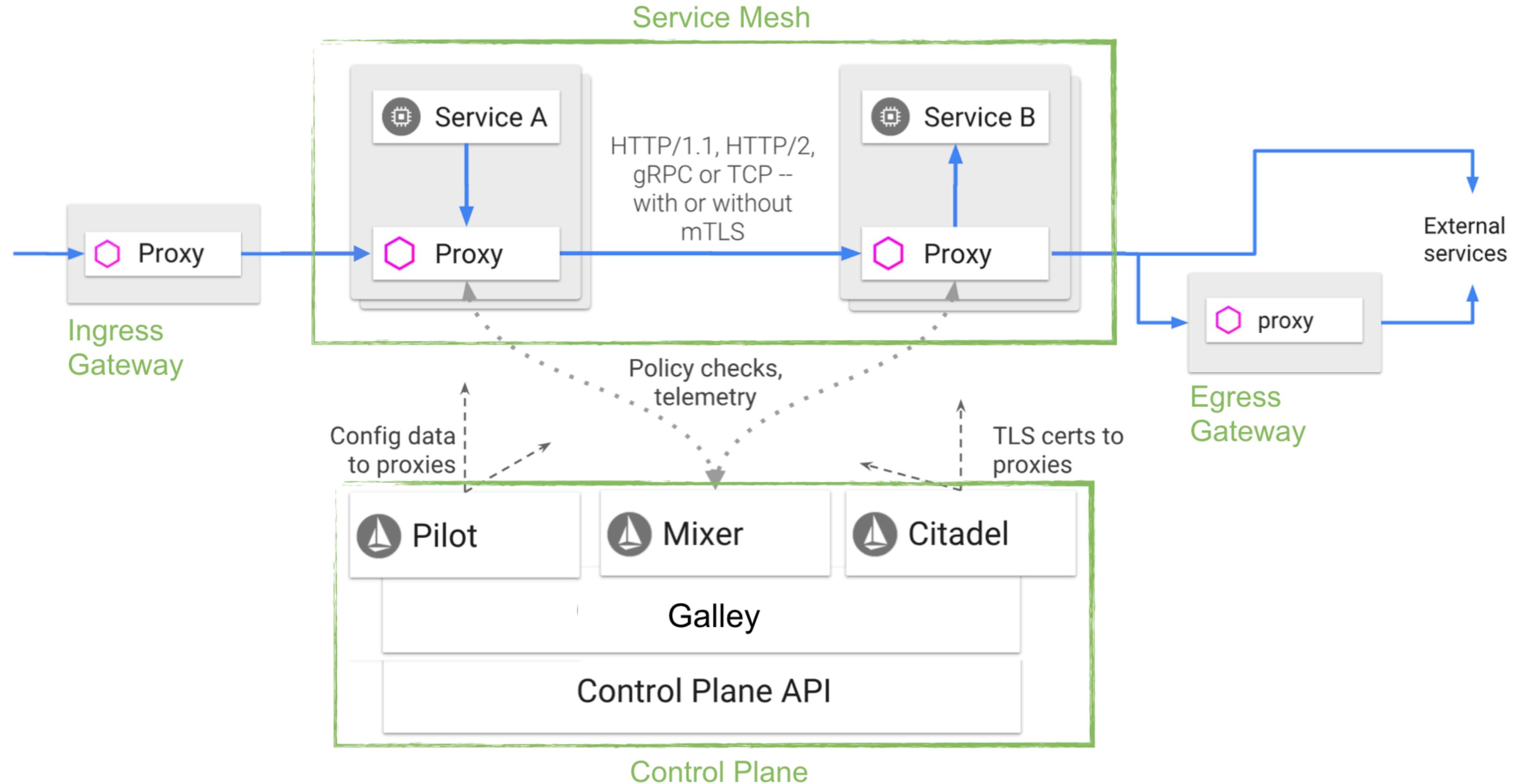




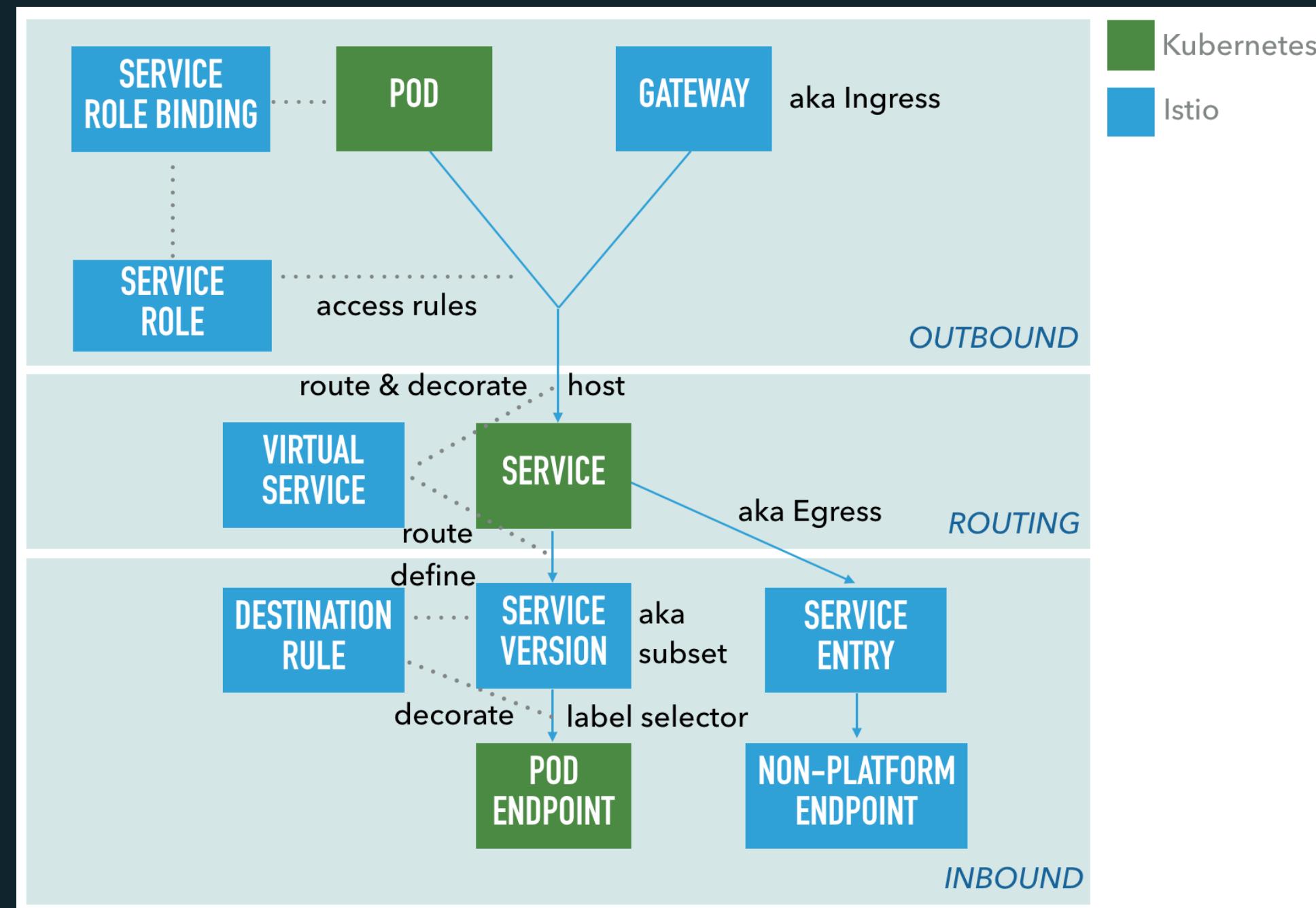


Setting the Sails with Istio 1.0.1

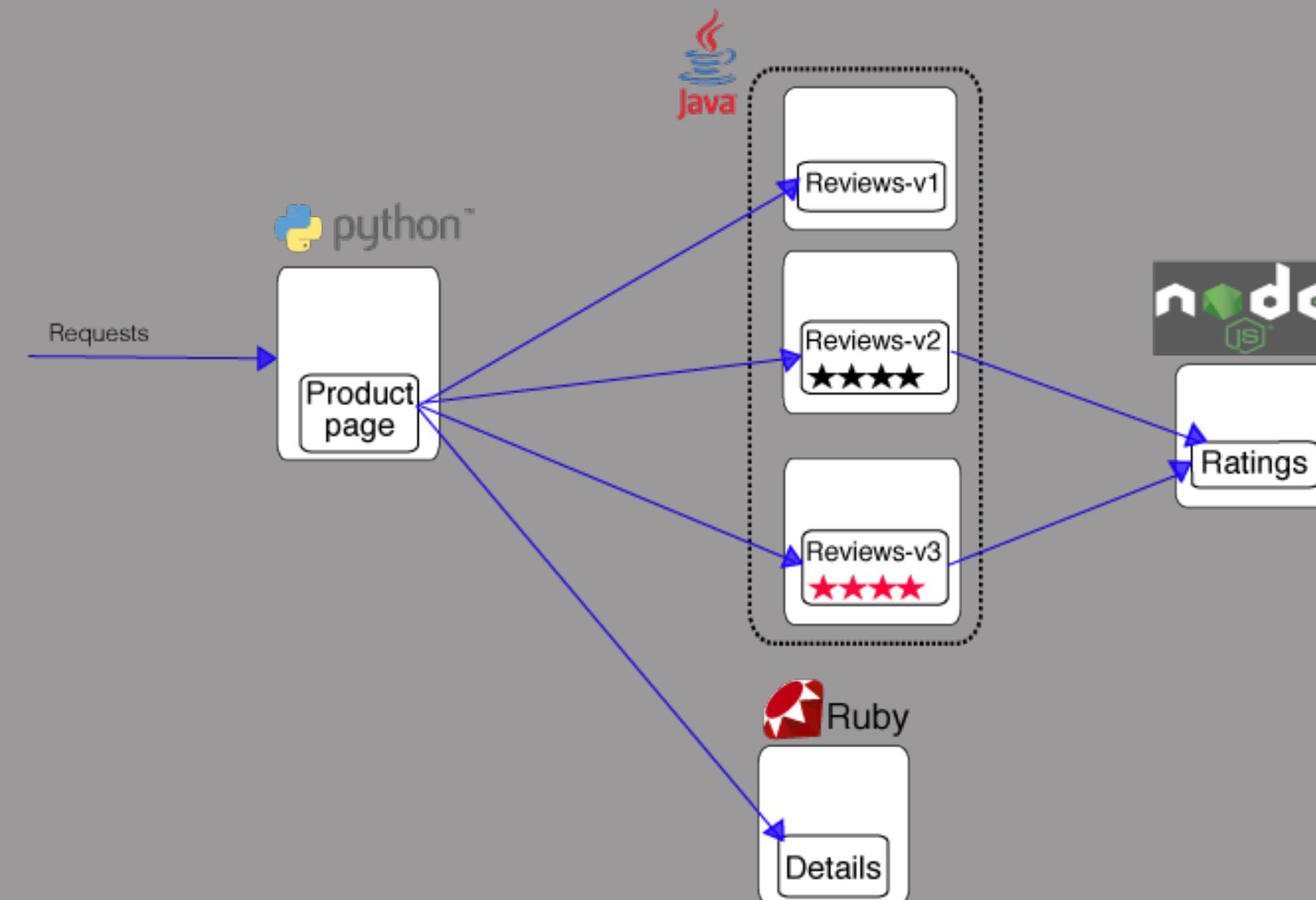




Istio Abstractions



Sample Application: BookInfo¹



¹ Istio BookInfo Sample (<https://istio.io/docs/examples/bookinfo>)

Cluster

Namespaces

Nodes

Persistent Volumes

Roles

Storage Classes

Namespace

default ▾

Overview

Workloads

Cron Jobs

Daemon Sets

Deployments

Jobs

Pods

Replica Sets

Replication Controllers

Stateful Sets

Discovery and Load Balancing

Ingresses

Services

Config and Storage

Config Maps

Persistent Volume Claims

Secrets

Details

Name: productpage-v1-7bbdd59459-5bb28

Namespace: default

Labels: app: productpage pod-template-hash: 3668815015 version: v1

Annotations: sidecar.istio.io/status: {"version":"55c9e544b52e1d4e45d18a58d0b34ba4b72531e45fb6d1572c77191422556ffc","initContainers":["istio-init"],"containers":["istio-proxy"],"volumes":["istio-envoy","istio-c..."]}

Creation Time: 2018-06-11T14:26 UTC

Status: Running

QoS Class: Burstable

Network

Node: docker-for-desktop

IP: 10.1.1.178

Containers

productpage

Image: istio/examples-bookinfo-productpage-v1:1.5.0

Environment variables: -

Commands: -

Args: -

istio-proxy

Image: docker.io/istio/proxyv2:0.8.0

Environment variables: POD_NAME: productpage-v1-7bbdd59459-5bb28

POD_NAMESPACE: default

INSTANCE_IP:

ISTIO_META_POD_NAME: productpage-v1-7bbdd59459-5bb28

ISTIO_META_INTERCEPTION_MODE: REDIRECT

Commands: -

Args: -

Init Containers

istio-init

Image: docker.io/istio/proxy_init:0.8.0

Environment variables: -

Commands: -

Args: -

Bookinfo: Gateway

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: bookinfo-gateway
spec:
  selector:
    istio: ingressgateway # use istio default controller
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "*"
```

Bookinfo: VirtualService

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: bookinfo
spec:
  hosts:
  - "*"
  gateways:
  - bookinfo-gateway
  http:
  - match:
    - uri:
        exact: /productpage
    - uri:
        exact: /login
    - uri:
        exact: /logout
    - uri:
        prefix: /api/v1/products
  route:
  - destination:
      host: productpage
      port:
        number: 9080
```

Bookinfo: DestinationRule

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: productpage
spec:
  host: productpage
  subsets:
  - name: v1
    labels:
      version: v1
```

Hands-on: Have a look around the YAMLs and Dashboard



Expose Istio Observability Tools

#Metrics: Prometheus

```
kubectl expose deployment prometheus --name=prometheus-expose \
--port=9090 --target-port=9090 --type=LoadBalancer -n=istio-system
```

#Metrics: Grafana

```
kubectl expose deployment grafana --name=grafana-expose \
--port=3000 --target-port=3000 --type=LoadBalancer -n=istio-system
open http://localhost:3000/d/1/istio-dashboard
```

#Tracing: Jaeger

```
kubectl expose deployment istio-tracing --name=tracing-expose \
--port=16686 --target-port=16686 --type=LoadBalancer -n=istio-system
open http://localhost:16686
```

#Tracing: ServiceGraph

```
kubectl expose service servicegraph --name=servicegraph-expose \
--port=8088 --target-port=8088 --type=LoadBalancer -n=istio-system
open http://localhost:8088/force/forcegraph.html
open http://localhost:8088/dotviz
```

Deploy Missing Observability Feature: Log Analysis (EFK)

```
cd .. #go to istio-playground/code
kubectl apply -f logging-stack.yaml
kubectl get pods -n=logging
kubectl expose deployment kibana --name=kibana-expose \
--port=5601 --target-port=5601 --type=LoadBalancer -n=logging
istioctl create -f fluentd-istio.yaml
```

Deploy Missing Observability Feature: Log Analysis (EFK)

open <http://localhost:5601/app/kibana>

- Perform some requests to the BookInfo application
- Use * as the index pattern
- Select @timestamp as the time filter field name

fluentd-istio.yaml (1/3)

```
# Configuration for logentry instances
apiVersion: "config.istio.io/v1alpha2"
kind: logentry
metadata:
  name: newlog
  namespace: istio-system
spec:
  severity: '"info"'
  timestamp: request.time
  variables:
    source: source.labels["app"] | source.service | "unknown"
    user: source.user | "unknown"
    destination: destination.labels["app"] | destination.service | "unknown"
    responseCode: response.code | 0
    responseSize: response.size | 0
    latency: response.duration | "0ms"
    monitored_resource_type: '"UNSPECIFIED"'
```

fluentd-istio.yaml (2/3)

```
# Configuration for a fluentd handler
apiVersion: "config.istio.io/v1alpha2"
kind: fluentd
metadata:
  name: handler
  namespace: istio-system
spec:
  address: "fluentd-es.logging:24224"
```

fluentd-istio.yaml (3/3)

```
# Rule to send logentry instances to the fluentd handler
apiVersion: "config.istio.io/v1alpha2"
kind: rule
metadata:
  name: newlogtofluentd
  namespace: istio-system
spec:
  match: "true" # match for all requests
  actions:
    - handler: handler.fluentd
      instances:
        - newlog.logentry
```

Stimulate!

```
slapper -rate 4 -targets ./target -workers 2 -maxY 15s
```

Download from: <https://github.com/adersberger/slapper/releases/tag/0.1>

Slapper² in action

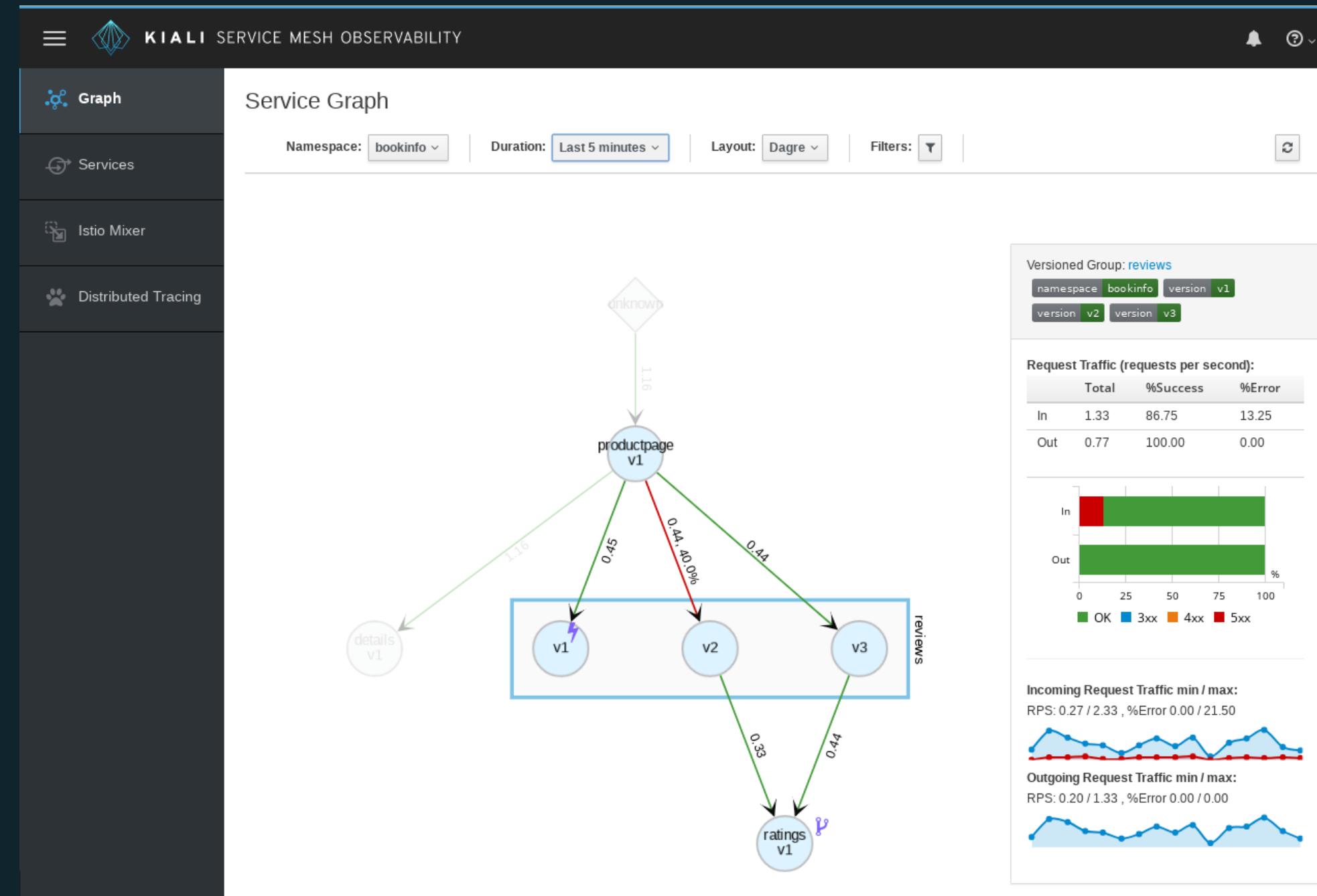
```
sent: 773    in-flight: 0    rate:    4/4 RPS responses: [200]: 7
72  [503]: 1    0/    0]
<1.0 ms: [    0/    0]
<1.0 ms: [    0/    0]
1.0-1.3 ms: [    0/    0]
1.3-1.7 ms: [    0/    0]
1.7-2.1 ms: [    0/    0]
2.1-2.8 ms: [    0/    0]
2.8-3.5 ms: [    0/    0]
3.5-4.6 ms: [    0/    0]
4.6-5.9 ms: [    0/    0]
5.9-7.6 ms: [    0/    0]
7.6-9.8 ms: [    0/    0]
10- 13 ms: [    0/    0]
13- 16 ms: [    0/    0]
16- 21 ms: [    0/    0]
21- 27 ms: [    2/    0] ****
27- 35 ms: [    11/    0] ****
35- 45 ms: [    10/    0] ****
45- 57 ms: [    9/    0] ****
57- 74 ms: [    7/    0] ****
74- 95 ms: [    0/    0]
95-122 ms: [    0/    0]
```

- ² Key bindings:
- q, ctrl-c - quit
- r - reset stats
- k - increase rate by 100 RPS
- j - decrease rate by 100 RPS

Hands-on



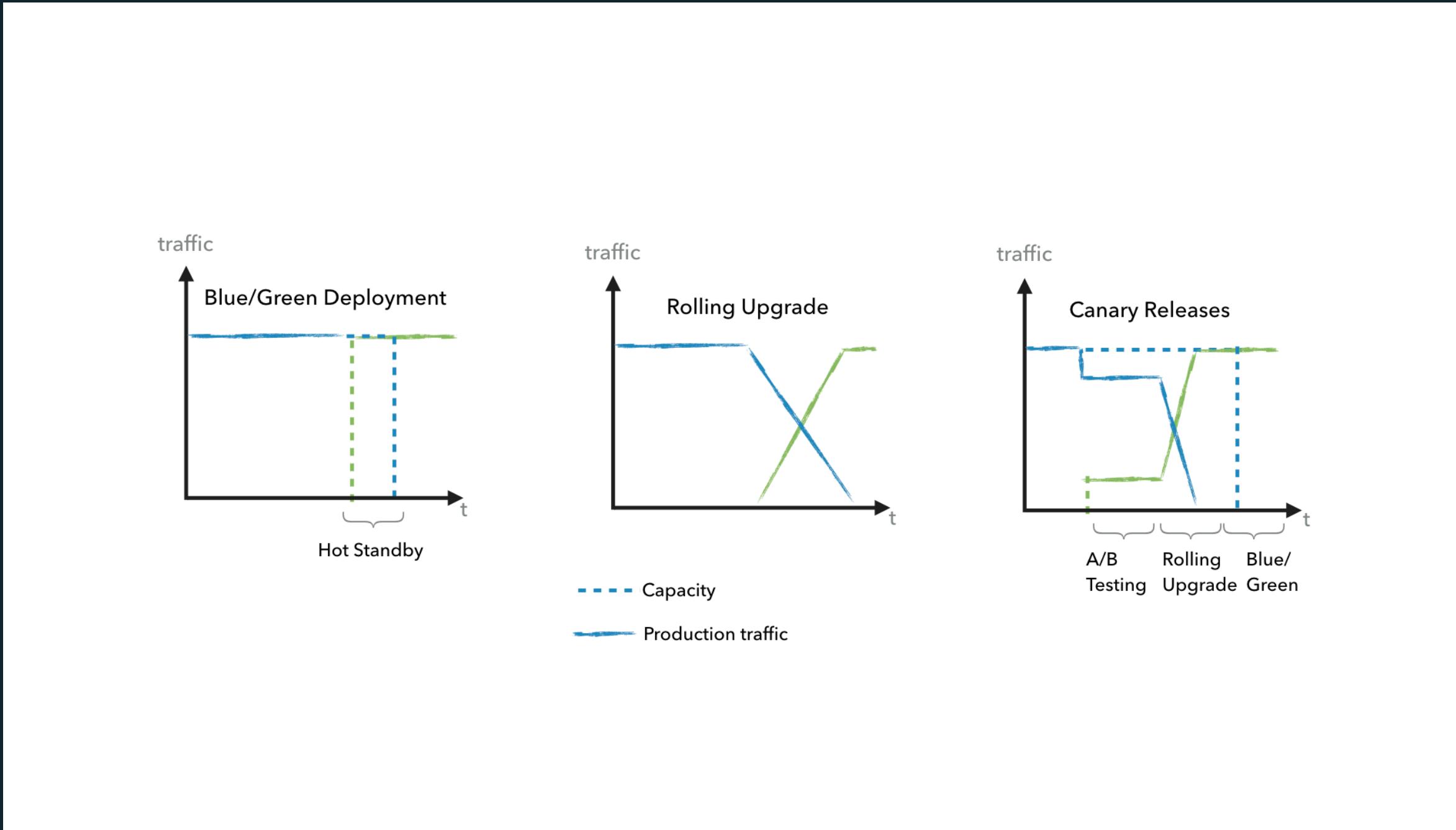
Observability Outlook: Kiali



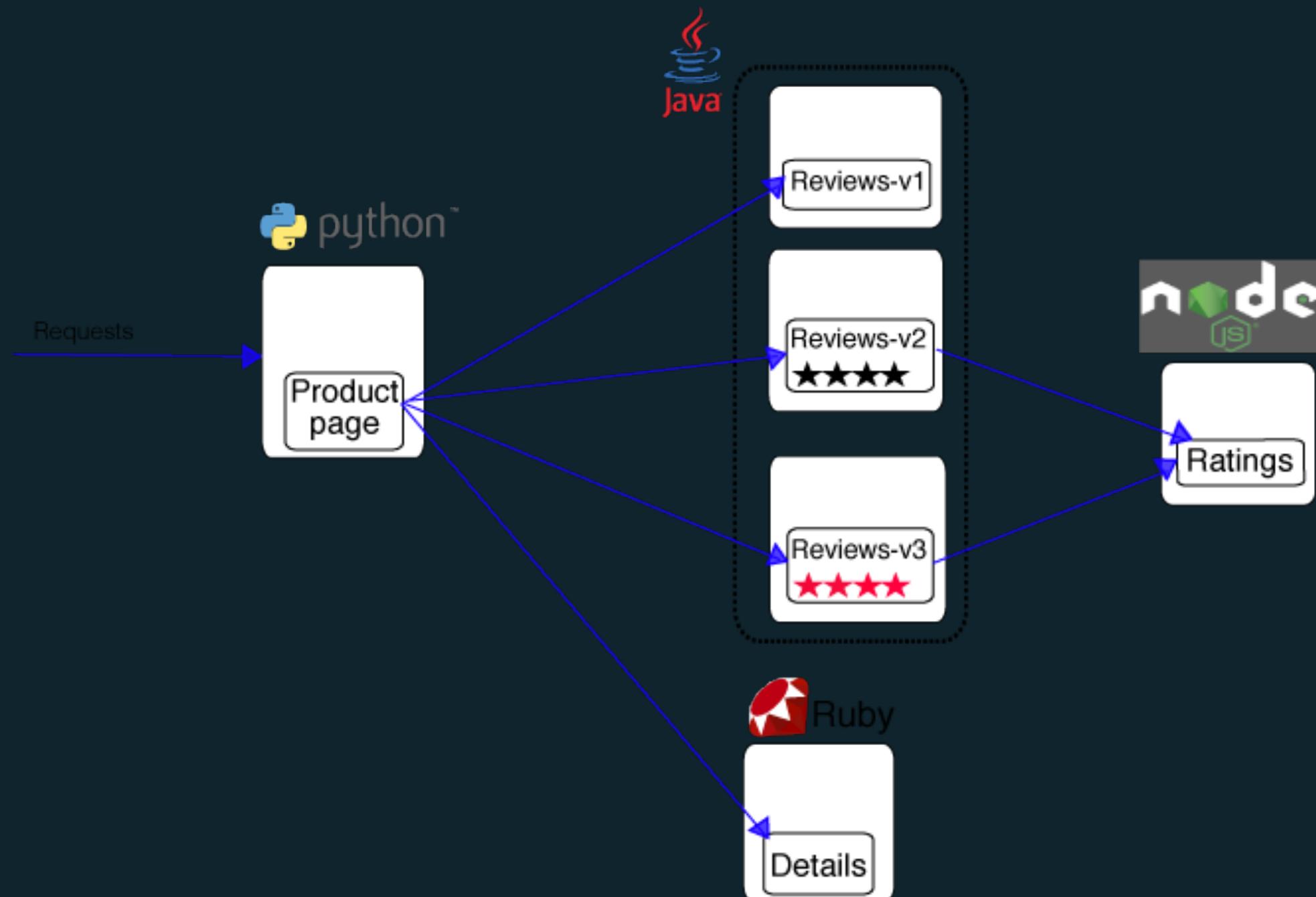
Observability Outlook: Kiali (macOS setup)

```
brew install gettext
brew link --force gettext
# follow k8s setup guide: https://www.kiali.io/gettingstarted
kubectl expose deployment kiali --name=kiali-expose \
    --port=20001 --target-port=20001 --type=LoadBalancer -n=istio-system
open http://localhost:20001
# login with admin/admin
```

Release Patterns



Sample Application Recap



Sample Desination Rule

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: reviews
spec:
  host: reviews
  subsets:
  - name: v1
    labels:
      version: v1
  - name: v2
    labels:
      version: v2
  - name: v3
    labels:
      version: v3
```

Canary Releases: A/B Testing

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: reviews
spec:
  hosts:
    - reviews
  http:
    - match:
        - headers:
            end-user:
              exact: jason
      route:
        - destination:
            host: reviews
            subset: v2
    - route:
        - destination:
            host: reviews
            subset: v1
```

Canary Releases: A/B Testing

```
cd istio-1.0.1
```

```
istioctl create -f samples/bookinfo/networking/virtual-service-all-v1.yaml
```

```
istioctl create -f samples/bookinfo/networking/destination-rule-all.yaml
```

```
istioctl replace -f samples/bookinfo/networking/virtual-service-reviews-test-v2.yaml
```

```
#open BookInfo application and login as user jason (password jason)  
open http://localhost/productpage
```

- login as "jason" / "jason" leads to v2 (black stars)
- anonymous user leads to v1 (no stars)

Canary Releases: Rolling Upgrade

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: reviews
spec:
  hosts:
    - reviews
  http:
    - route:
        - destination:
            host: reviews
            subset: v1
            weight: 50
        - destination:
            host: reviews
            subset: v3
            weight: 50
```

```
istioctl replace -f samples/bookinfo/networking/virtual-service-reviews-50-v3.yaml
```

Canary Releases: Blue/Green

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: reviews
spec:
  hosts:
    - reviews
  http:
    - route:
        - destination:
            host: reviews
            subset: v3
```

```
istioctl replace -f samples/bookinfo/networking/virtual-service-reviews-v3.yaml
istioctl get routerules
```

Hands-on



Time to Play!

Traffic Management	Resiliency	Security	Observability
Request Routing	Timeouts	mTLS	Metrics
Load Balancing	Circuit Breaker	Role-Based Access Control	Logs
Traffic Shifting	Health Checks (active, passive)	Workload Identity	Traces
Traffic Mirroring	Retries	Authentication Policies	
Service Discovery	Rate Limiting	CORS Handling	
Ingress, Egress	Delay & Fault Injection	TLS Termination, SNI	
API Specification	Connection Pooling		
Multicloud Mesh			

<https://istio.io/docs/tasks>

<https://istio.io/about/feature-stages>

Hands-on



Thank you!



josef.adersberger@qaware.de



@adersberger

TWITTER.COM/QAWARE - SLIDEShare.NET/QAWARE

FAQ

Q: How does the Envoy proxy intercept requests?

A: With IPtable rules (alls rules pointing to envoy)

Q: How does the auto-sidecar magic work?

A: With an Istio admission controller enhancing the deployments

Q: How can I list all Istio custom resource definitions and commands?

A: kubectl api-resources

Q: I can't see any metrics, logs, traces. What should I do?

A: Restart istio-telemetry Deployment or kubectl replace -f fluentd-istio.yaml