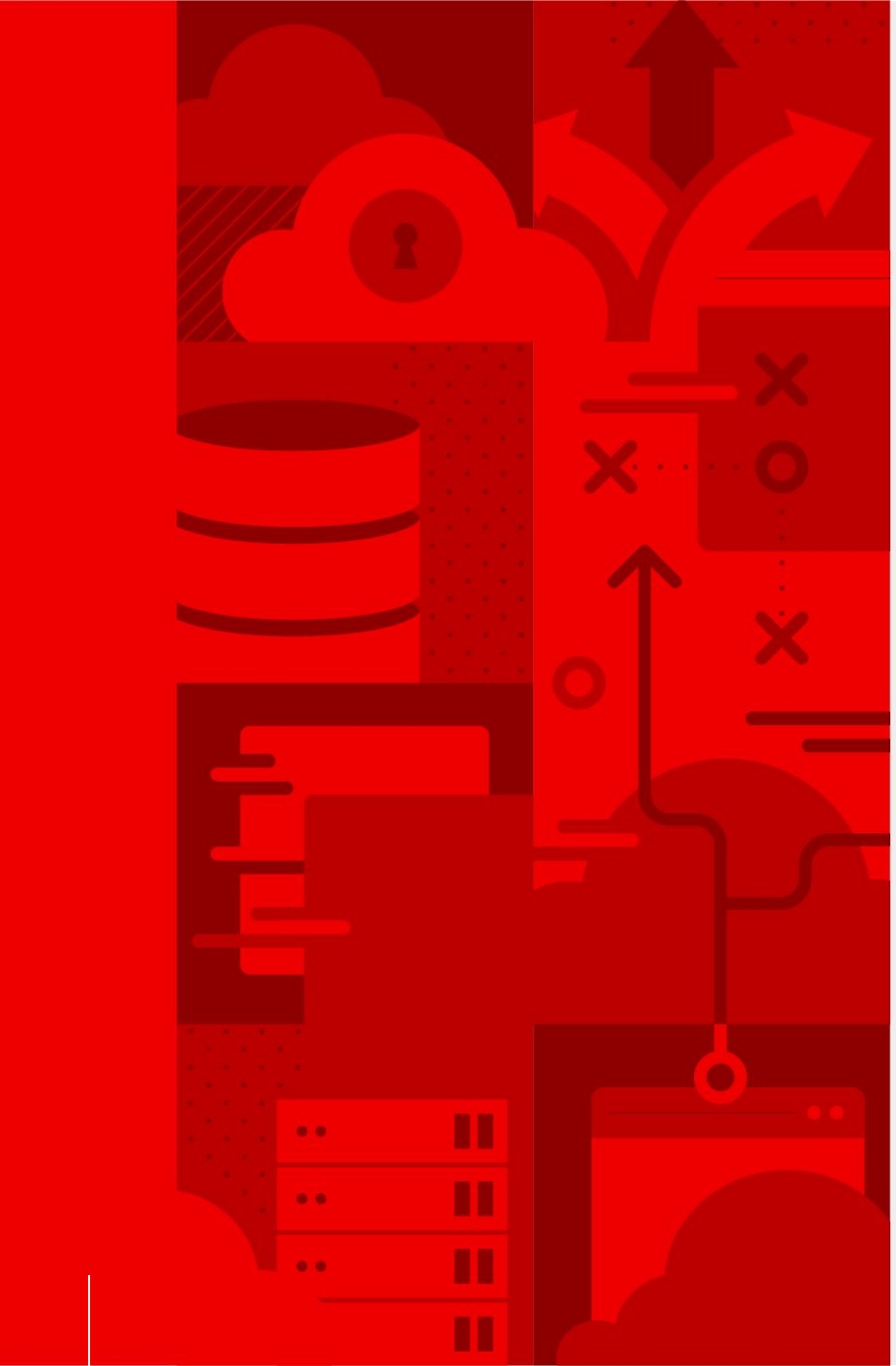
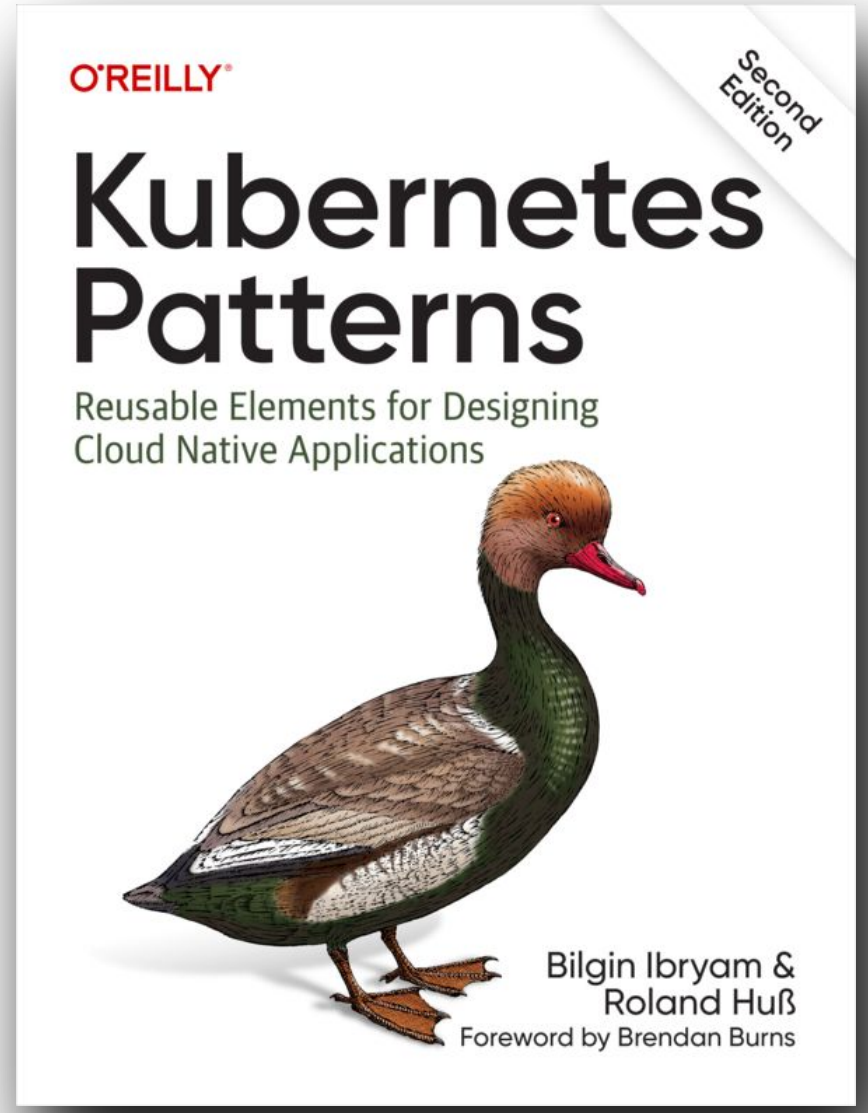


Kubernetes Patterns

Reusable Elements for Designing Cloud-Native Applications

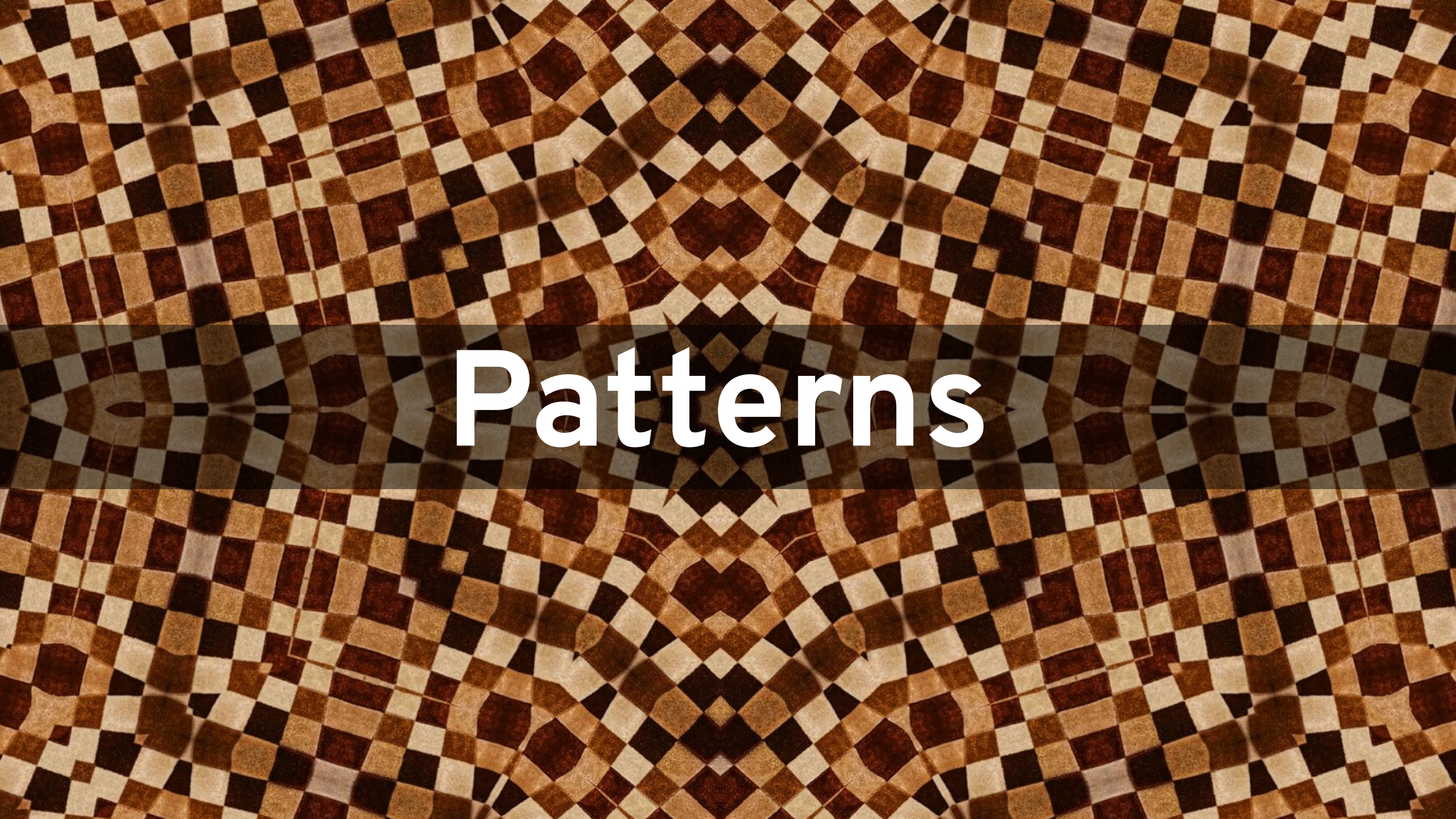
Dr. Roland Huß
Senior Principal Software Engineer
@ro14nd@hachyderm.io
https://k8spatterns.io

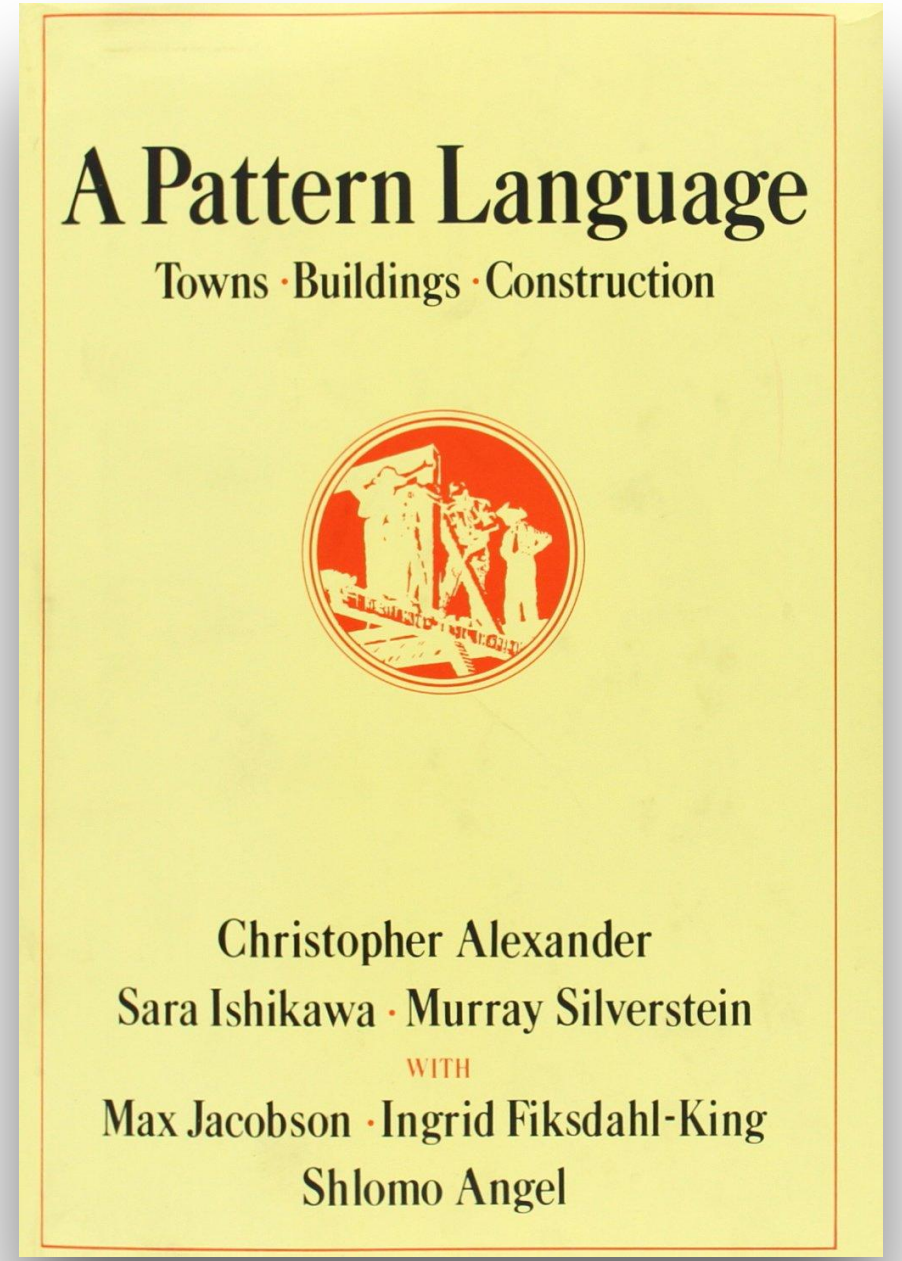
Red Hat



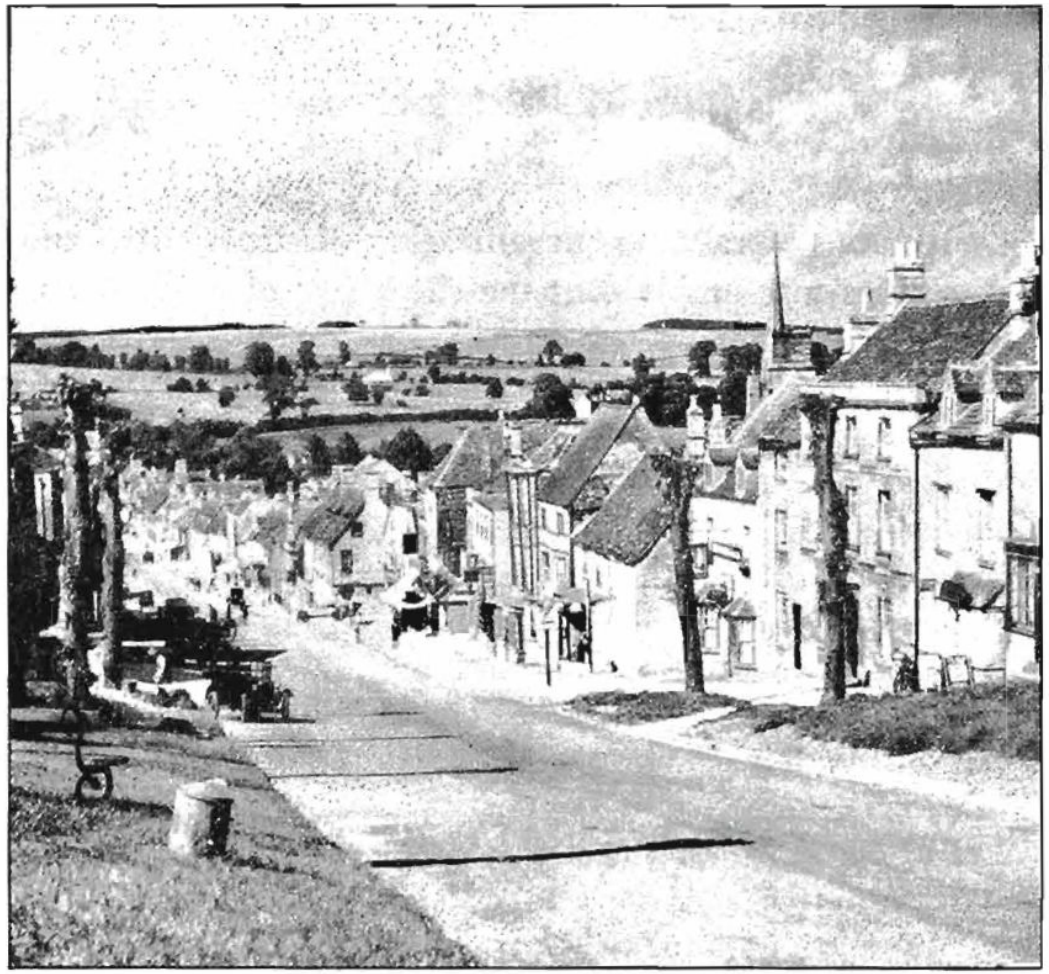
https://k8spatterns.io

Red Hat

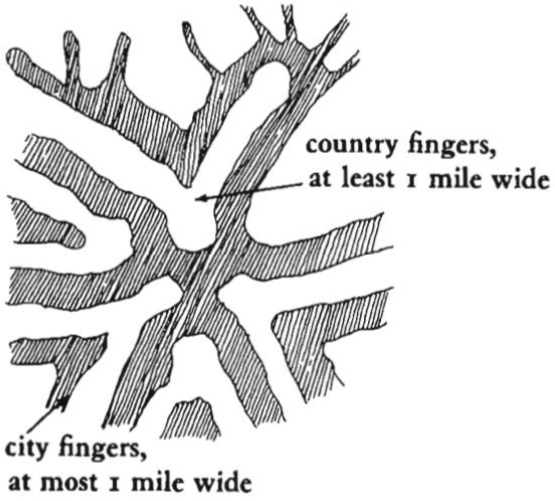

Patterns


A Pattern Language
Towns · Buildings · Construction
Christopher Alexander
Sara Ishikawa · Murray Silverstein
WITH
Max Jacobson · Ingrid Fiksdahl-King
Shlomo Angel

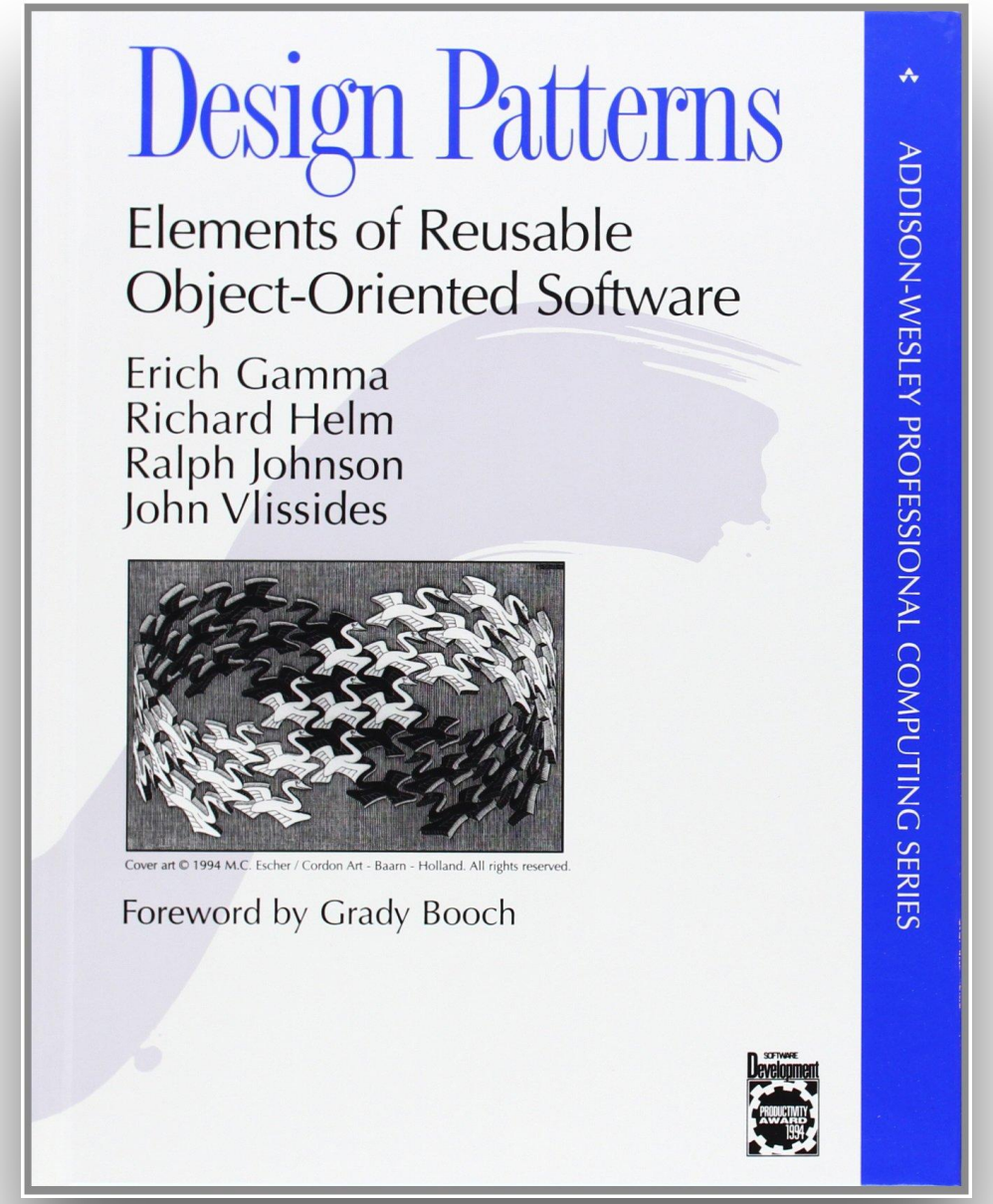
3 CITY COUNTRY FINGERS**



When the countryside is far away the city becomes a prison.



A Pattern Language, Christopher Alexander et. al, 1977, pp. 21


Design Patterns
Elements of Reusable
Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Cover art © 1994 M.C. Escher / Cordon Art - Baarn - Holland. All rights reserved.
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

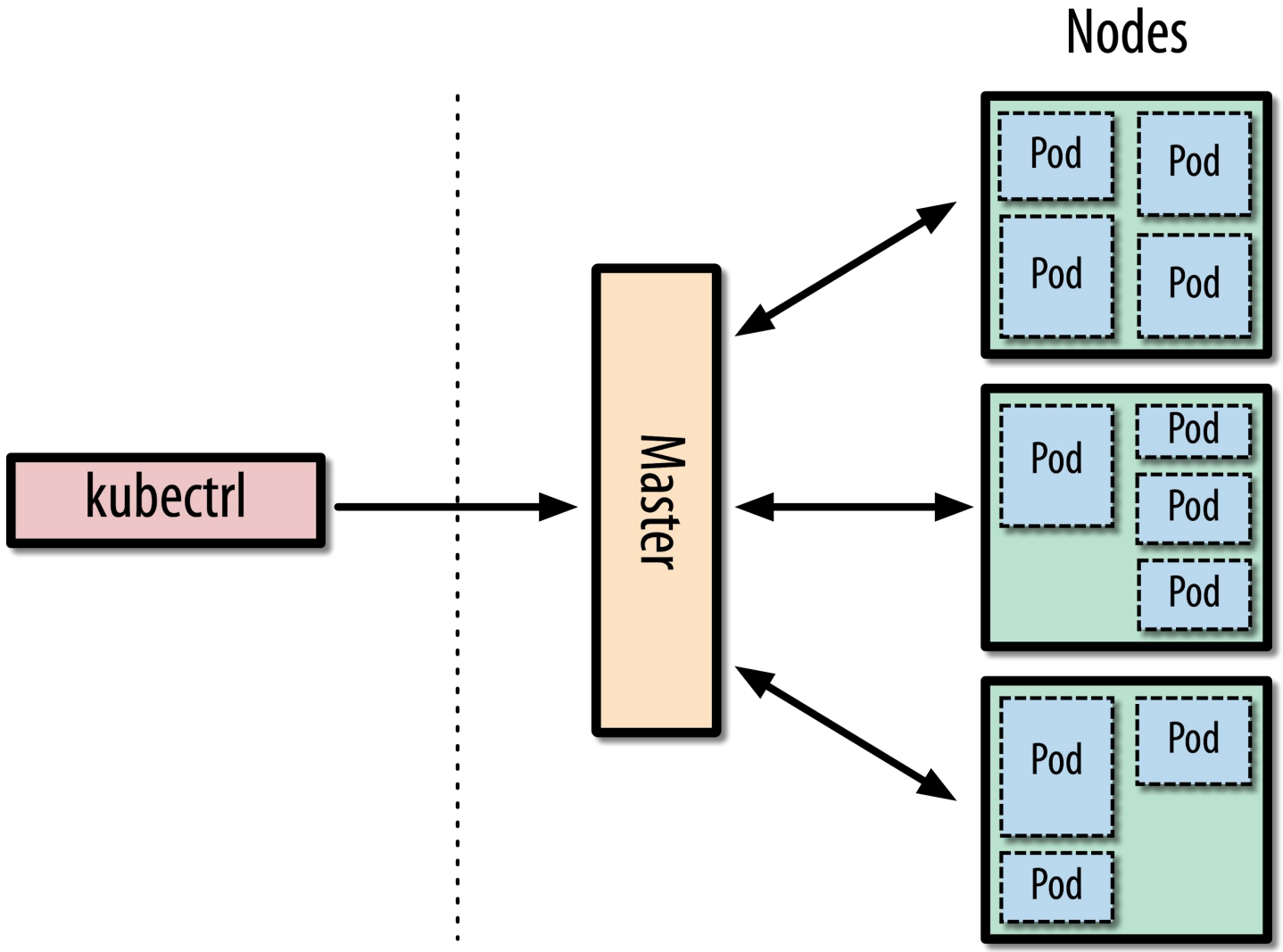
Red Hat


Kubernetes

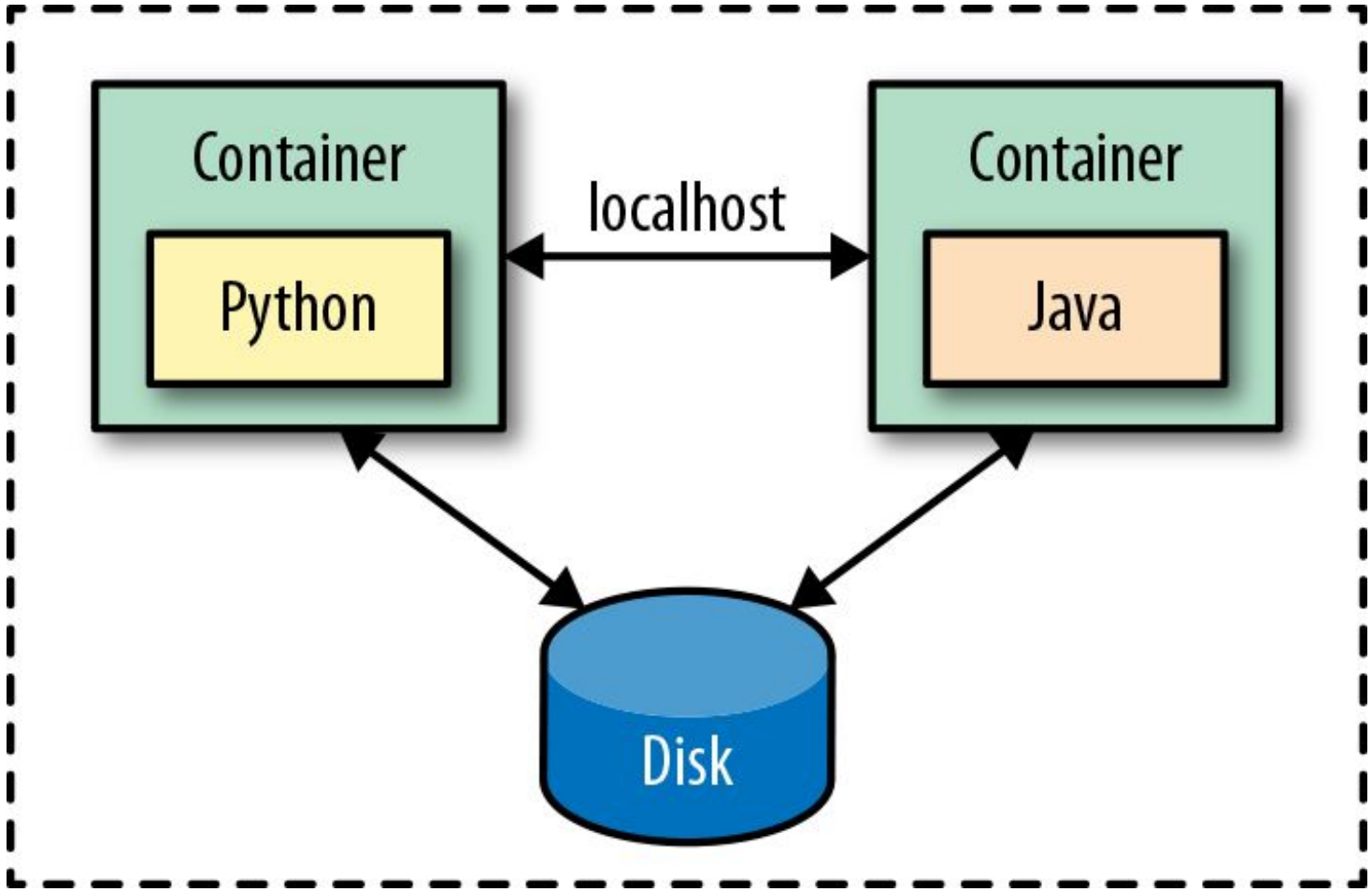
Kubernetes

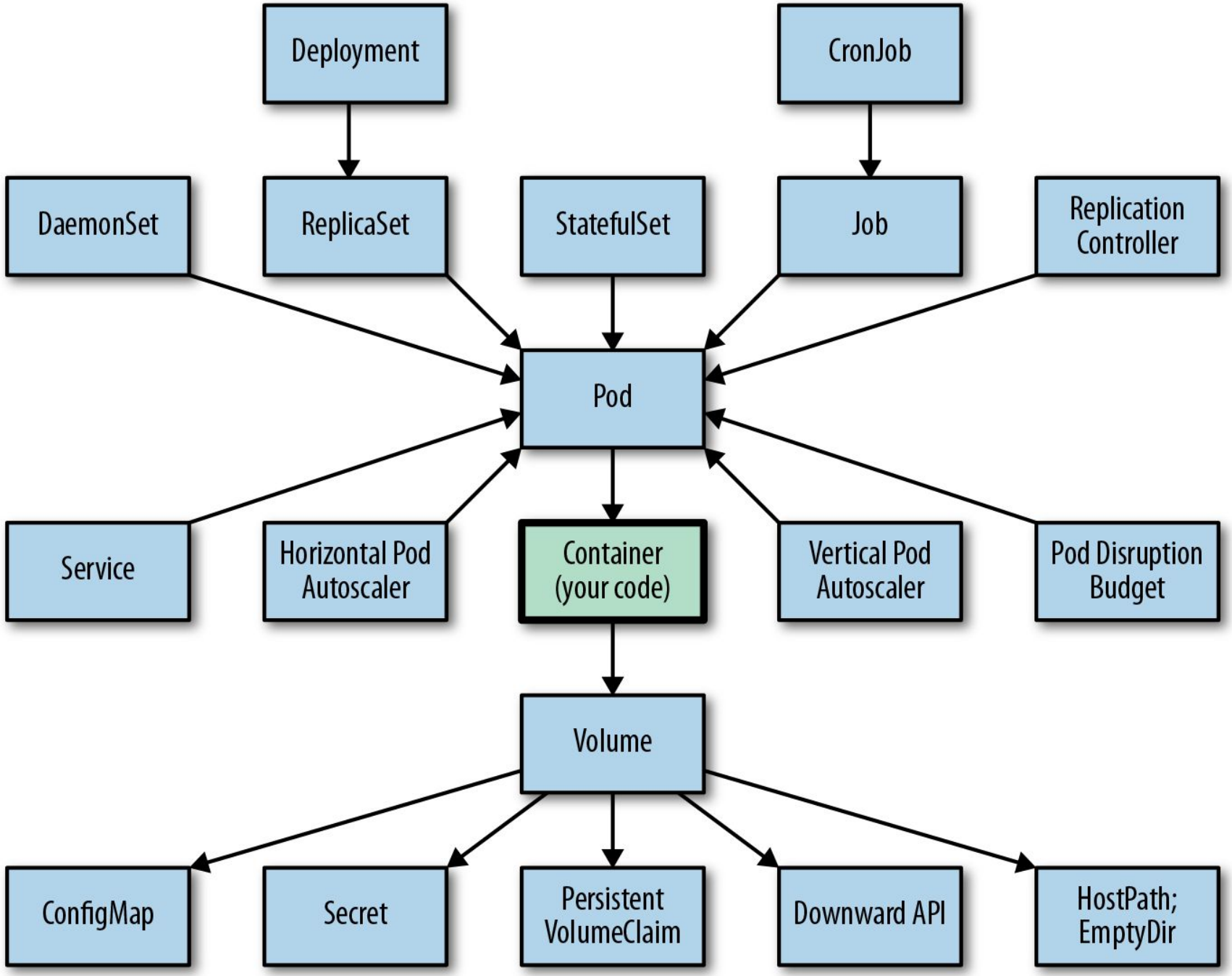
- Open Source container orchestration system started by Google in 2014
  - Scheduling
  - Self-healing
  - Horizontal and vertical scaling
  - Service discovery
  - Rollout and Rollbacks
- Declarative resource-centric REST API

Architecture

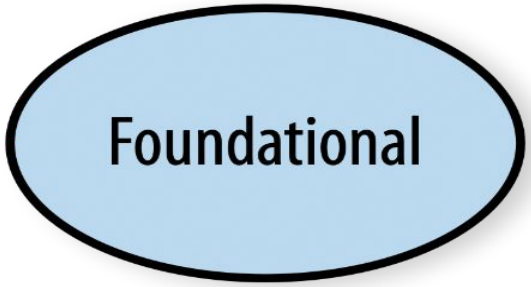


Pod



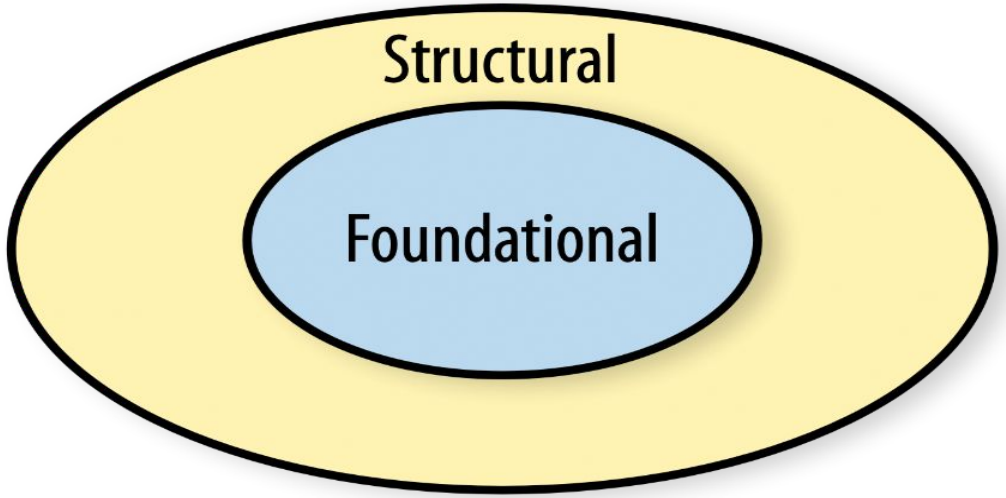

Deployment
CronJob
DaemonSet
ReplicaSet
StatefulSet
Job
Replication Controller
Pod
Service
Horizontal Pod Autoscaler
Container (your code)
Vertical Pod Autoscaler
Pod Disruption Budget
Volume
ConfigMap
Secret
Persistent VolumeClaim
Downward API
HostPath; EmptyDir

Pattern Categories



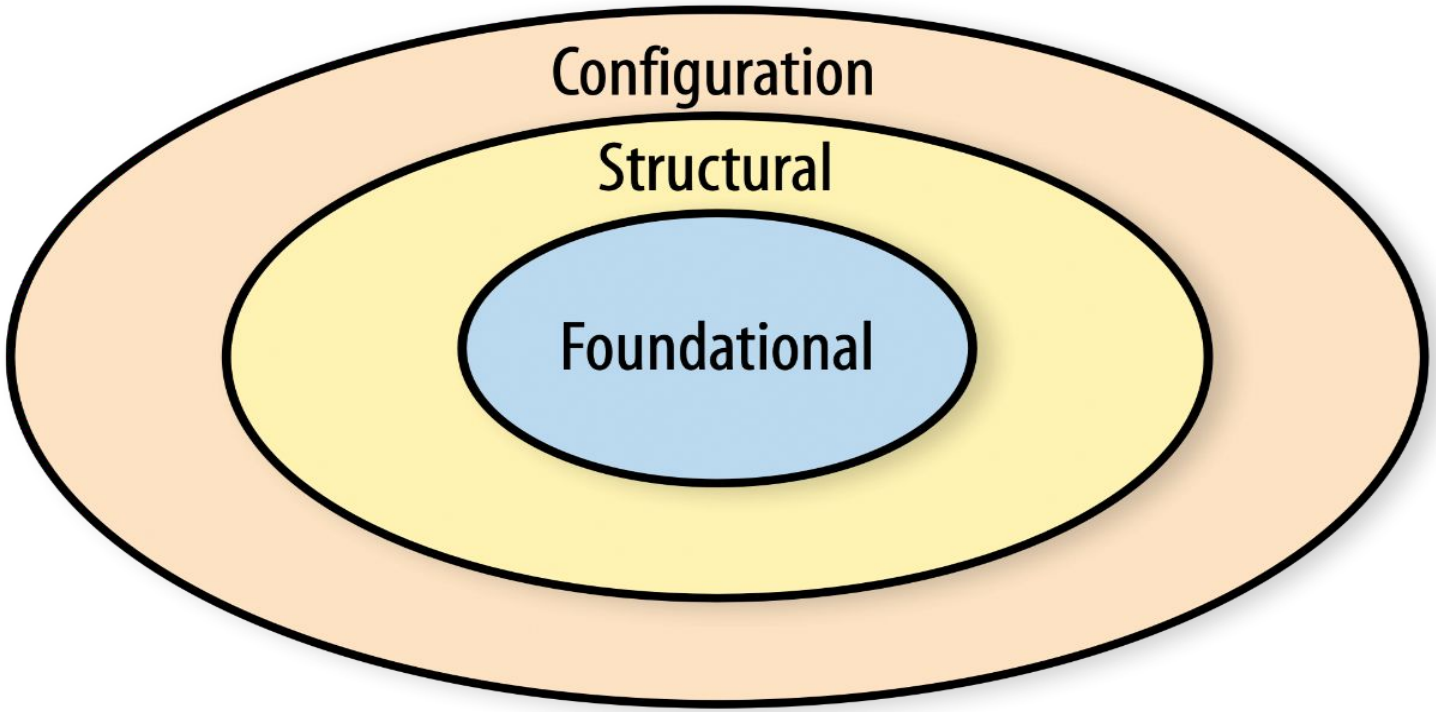
Red Hat

Pattern Categories

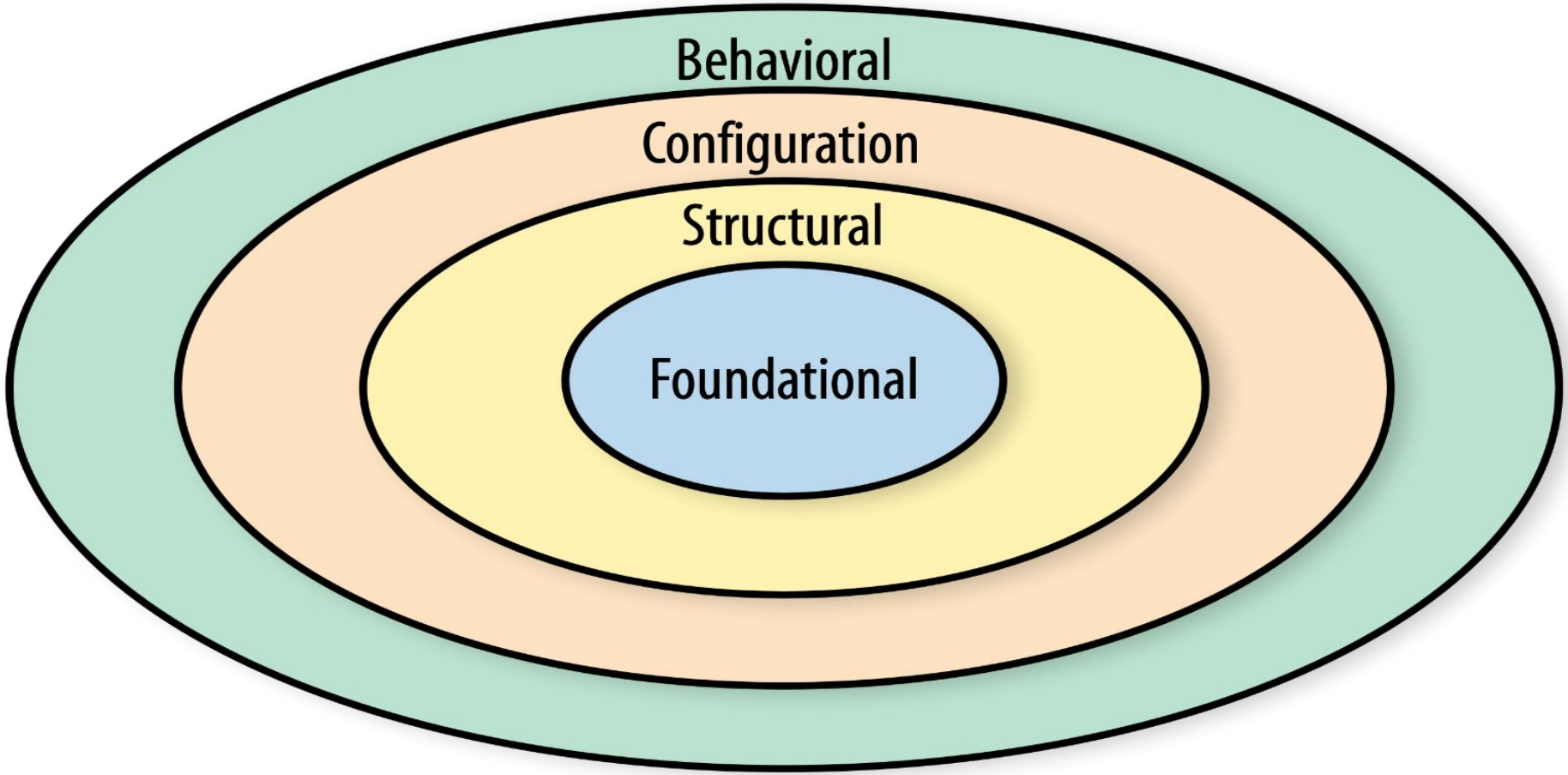


Red Hat

Pattern Categories

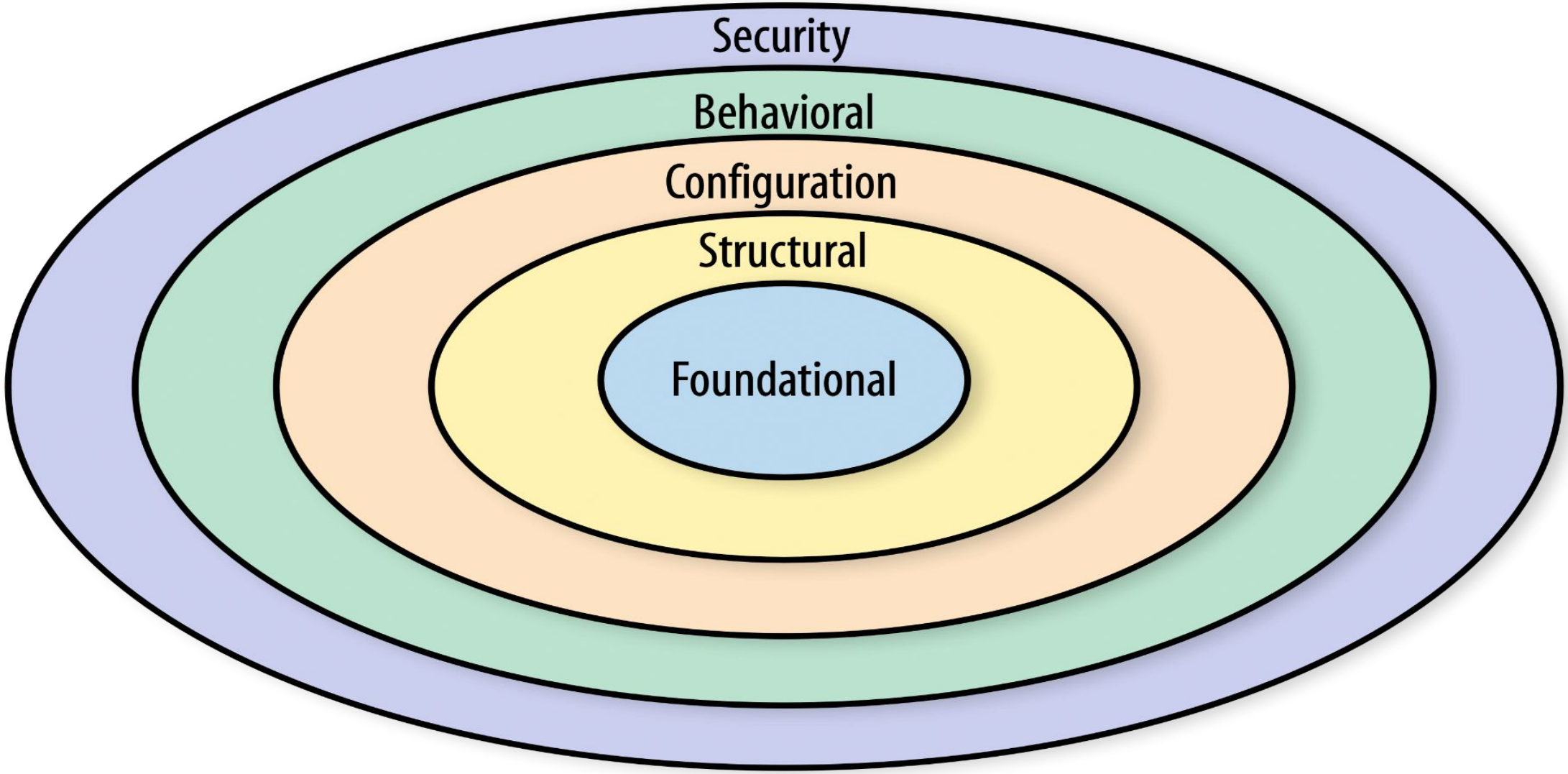


Pattern Categories



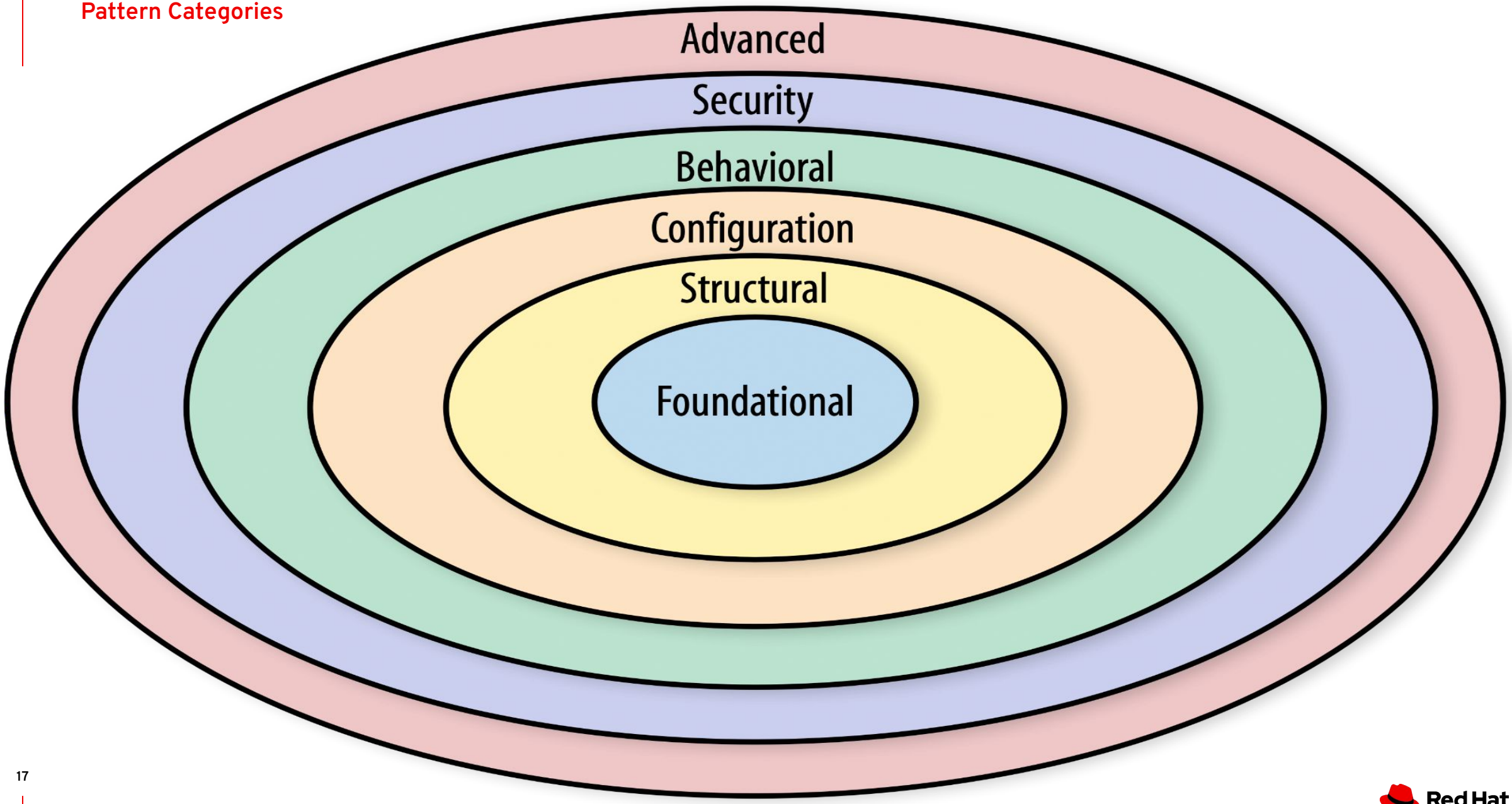
Red Hat

Pattern Categories

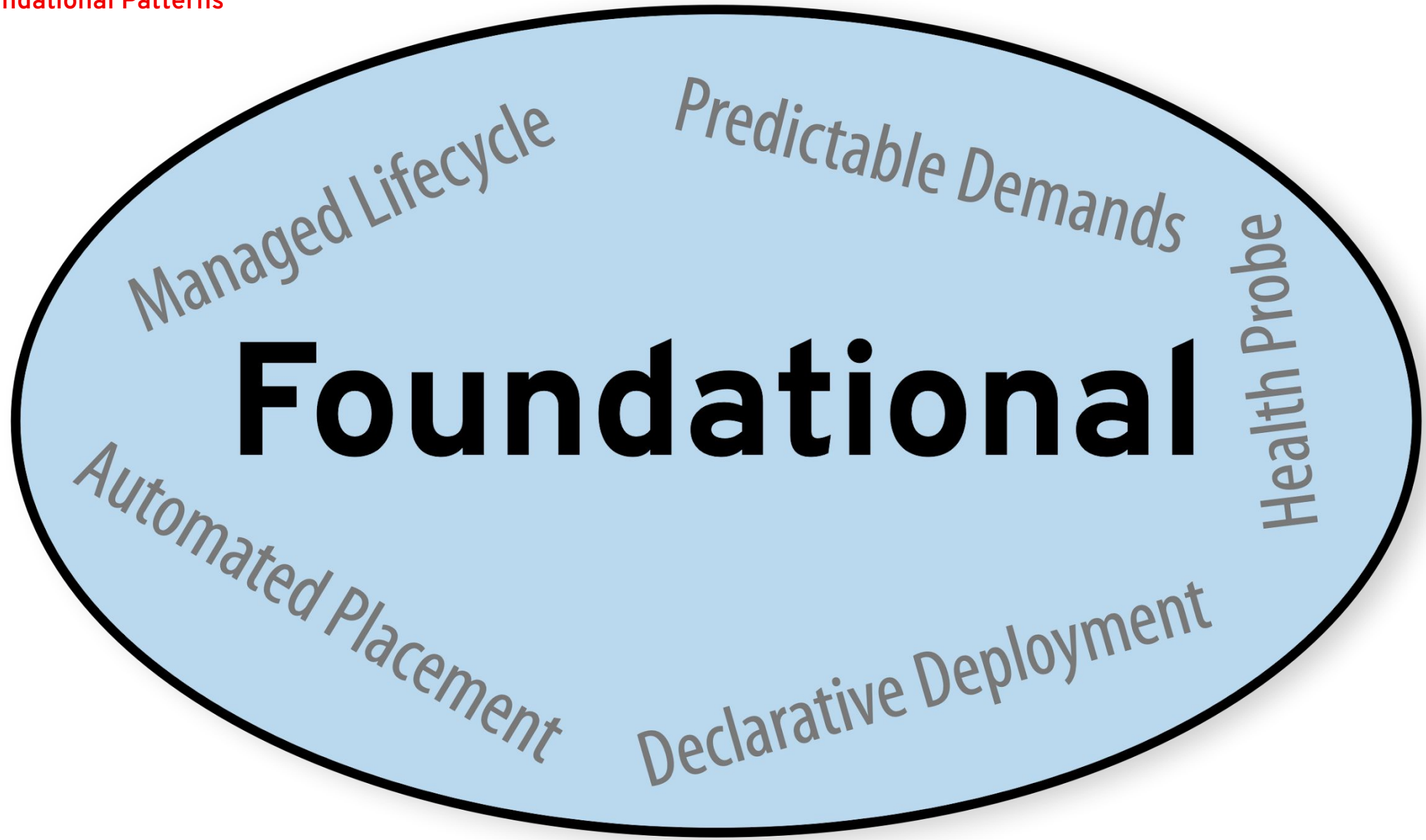


Red Hat

Pattern Categories



Red Hat


Managed Lifecycle
Predictable Demands
Health Probe
Foundational
Automated Placement
Declarative Deployment


Predictable
Demands

How to declare application requirements

Application Requirements

- Declared requirements help in
  - Scheduling decisions
  - Capacity planning
  - Matching infrastructure services
- Hard runtime dependencies
  - Persistent Volumes
  - Host ports
  - Dependencies on ConfigMaps and Secrets

Resource Profile

```
apiVersion: v1
kind: Pod
metadata:
  name: http-server
spec:
  containers:
  - image: nginx
    name: nginx
    resources:
      requests:
        cpu: 200m
        memory: 100Mi
      limits:
        cpu: 300m
        memory: 200Mi
```

Quality-of-Service Classes

- **Best Effort**
  - No requests or limits
- **Burstable**
  - requests < limits
- **Guaranteed**
  - requests == limits

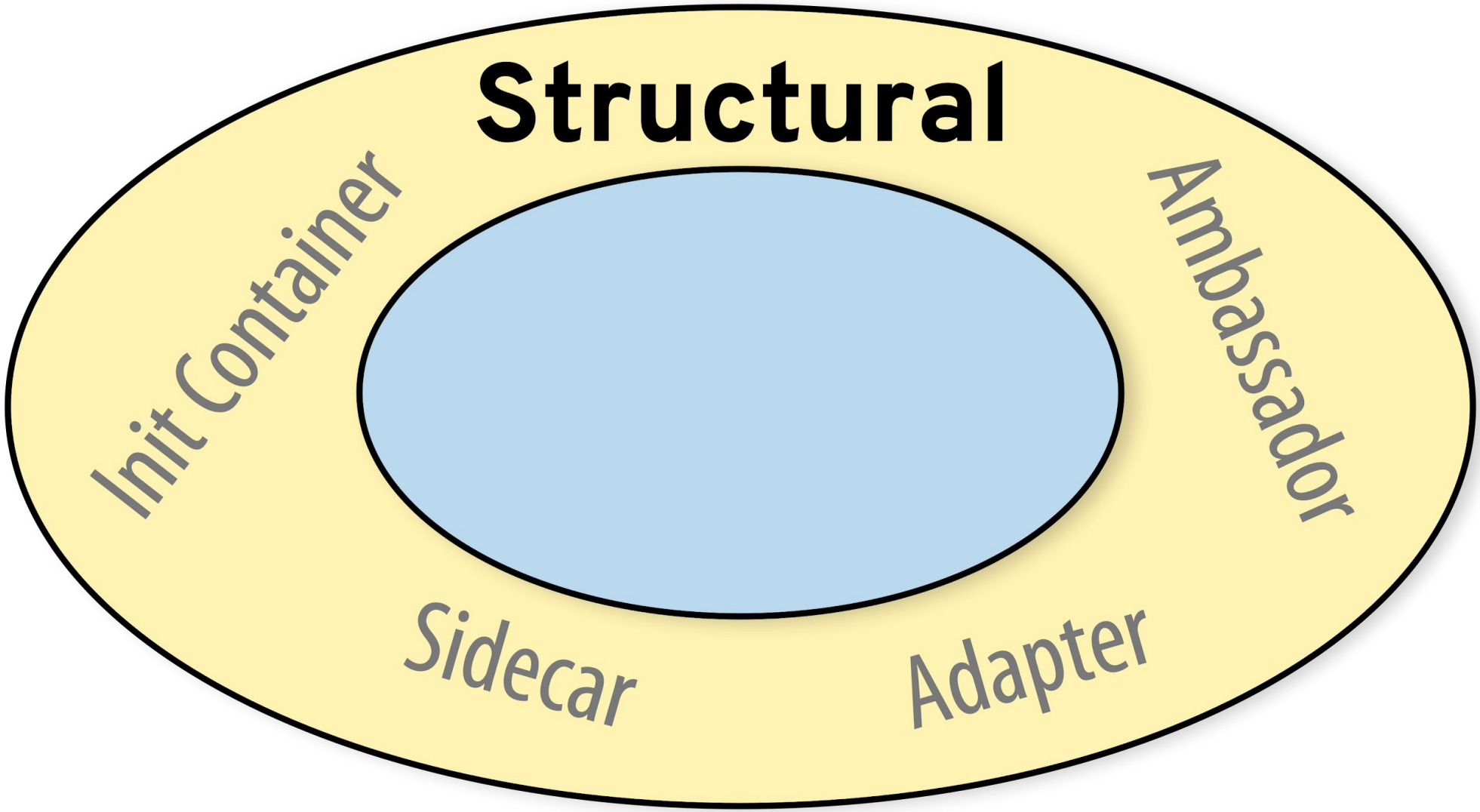
Recommendations

For **memory**, set requests **equal** to limits

For **cpu**, set requests, but **no limits**

References:

- For the love of god, stop using CPU limits on Kubernetes
- What everyone should know about Kubernetes memory limits, OOMKilled pods, and pizza parties



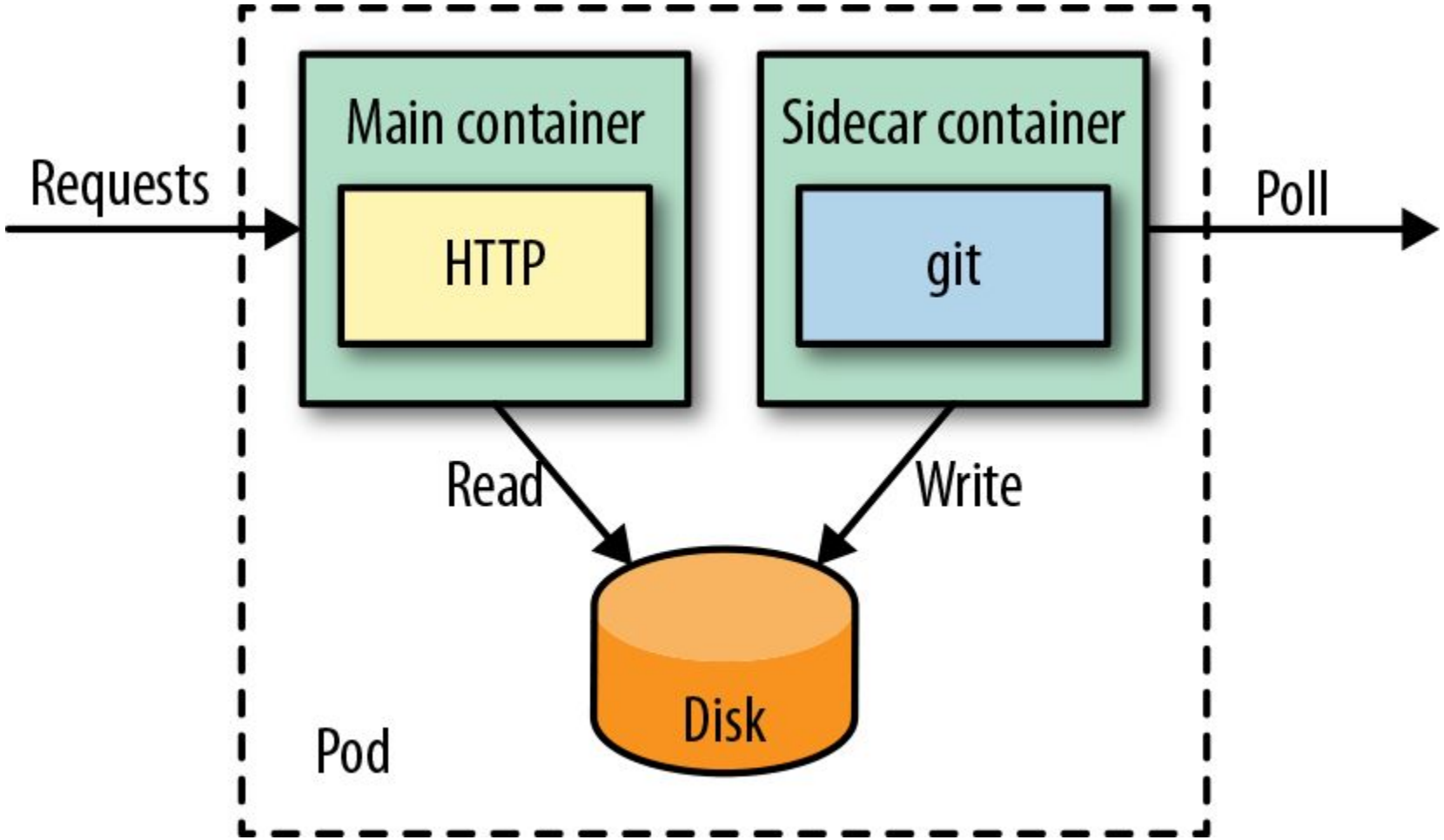
Sidecar

How to enhance the functionality of an application without changing it

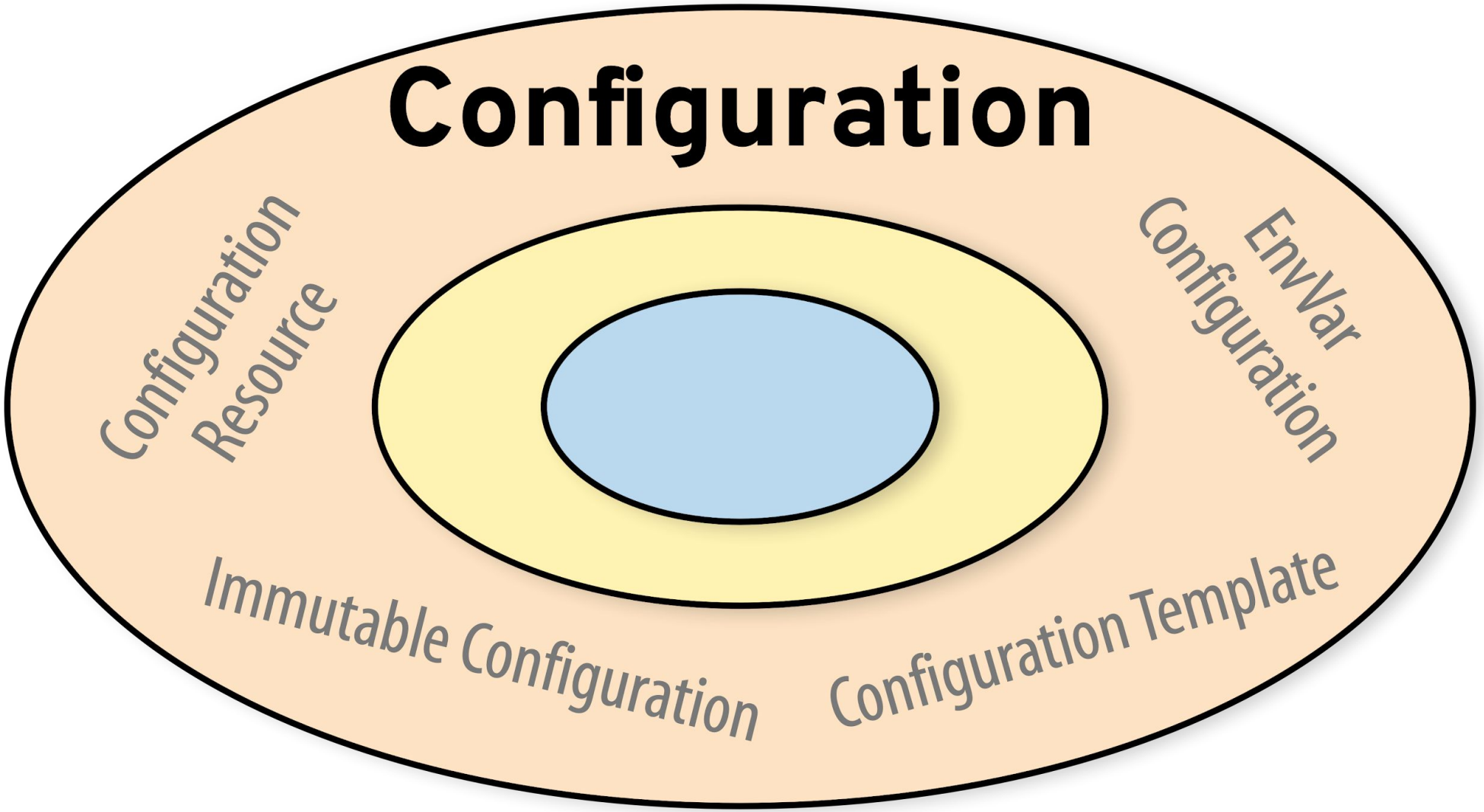
Sidecar

- Runtime collaboration of containers
- Connected via shared resources:
  - Network
  - Volumes
- Similar what AOP is for programming
- Separation of concerns

Sidecar



Demo


Configuration
Configuration Resource
EnvVar Configuration
Immutable Configuration
Configuration Template

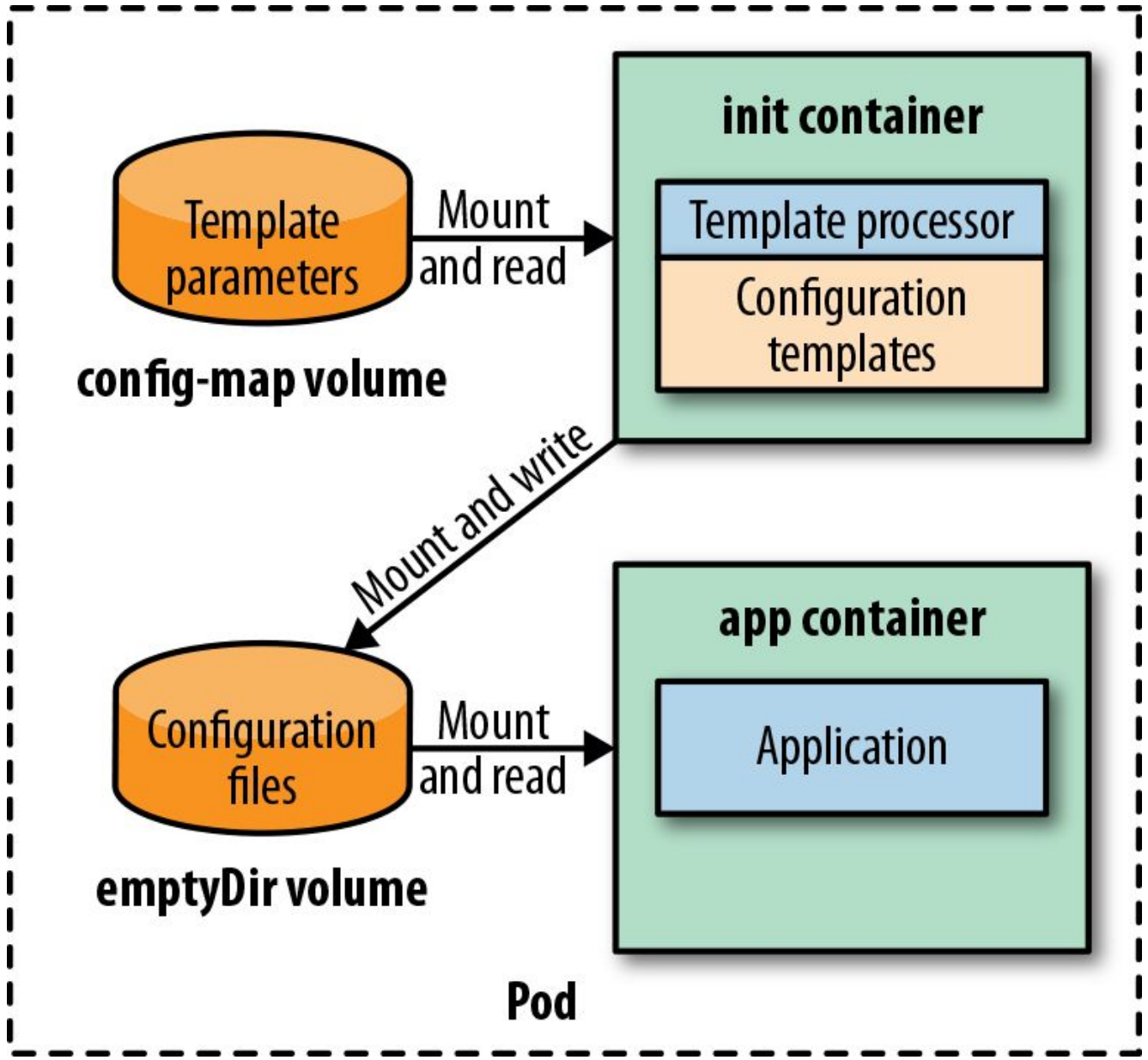
Configuration Template

How to manage large and complex similar configuration data

Preparing Configuration during Startup

- Init Container ...
  - ... contains a template processor
  - ... holds the configuration template
  - ... picks up template parameter from a ConfigMap
  - ... stores final configuration on a shared volume
- Main Container ....
  - ... accesses created configuration from shared volume

Configuration Template

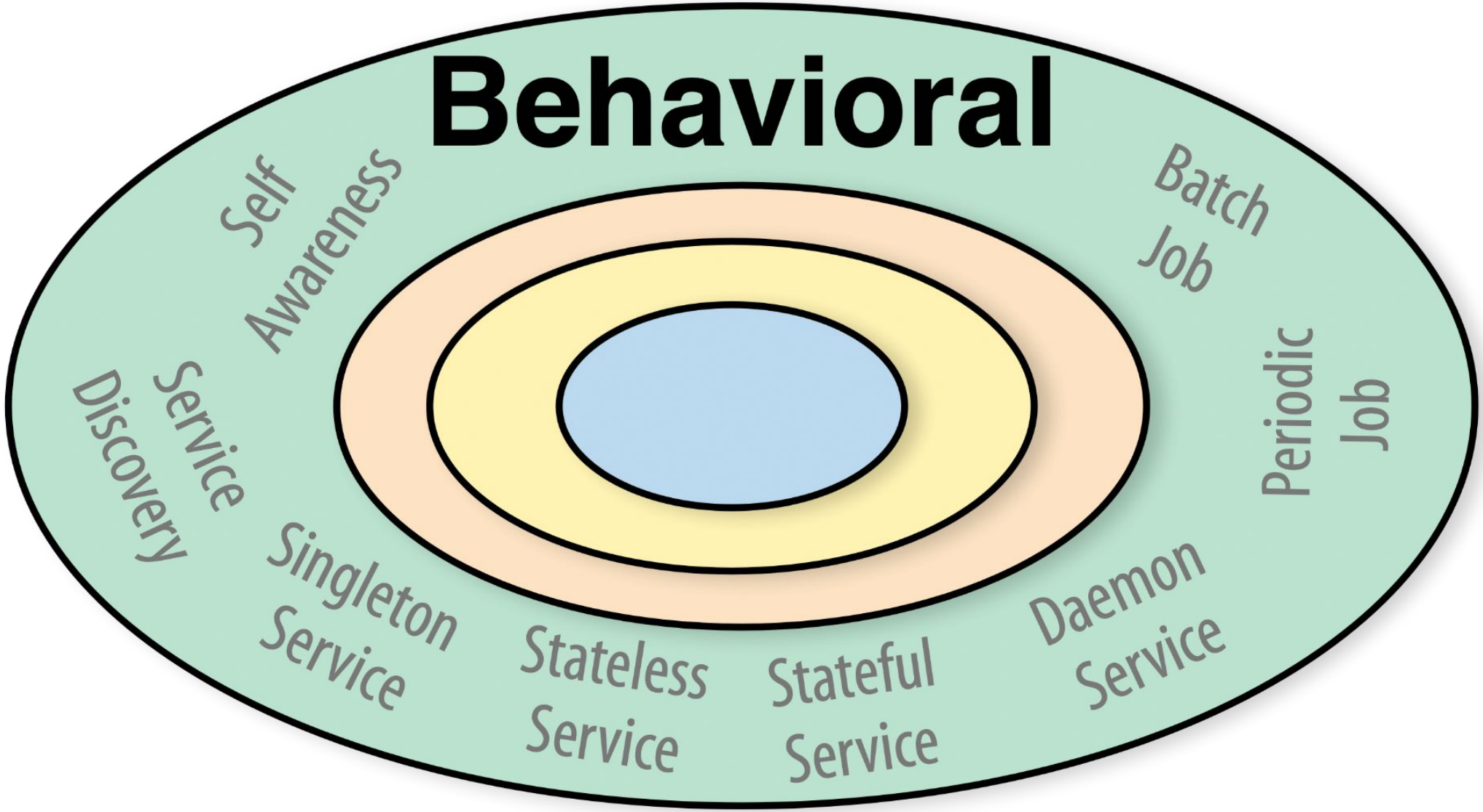


Configuration Template

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: wildfly-cm-template
spec:
  replicas: 1
  template:
    spec:
      initContainers:
      - image: k8spatterns/config-init
        name: init
        volumeMounts:
        - mountPath: "/params"
          name: wildfly-parameters
        - mountPath: "/out"
          name: wildfly-config
```

```
containers:
- image: jboss/wildfly:10.1.0.Final
  name: server
  volumeMounts:
  - mountPath: "/config"
    name: wildfly-config
volumes:
- name: wildfly-parameters
  configMap:
    name: wildfly-params-cm
- name: wildfly-config
  emptyDir: {}
```

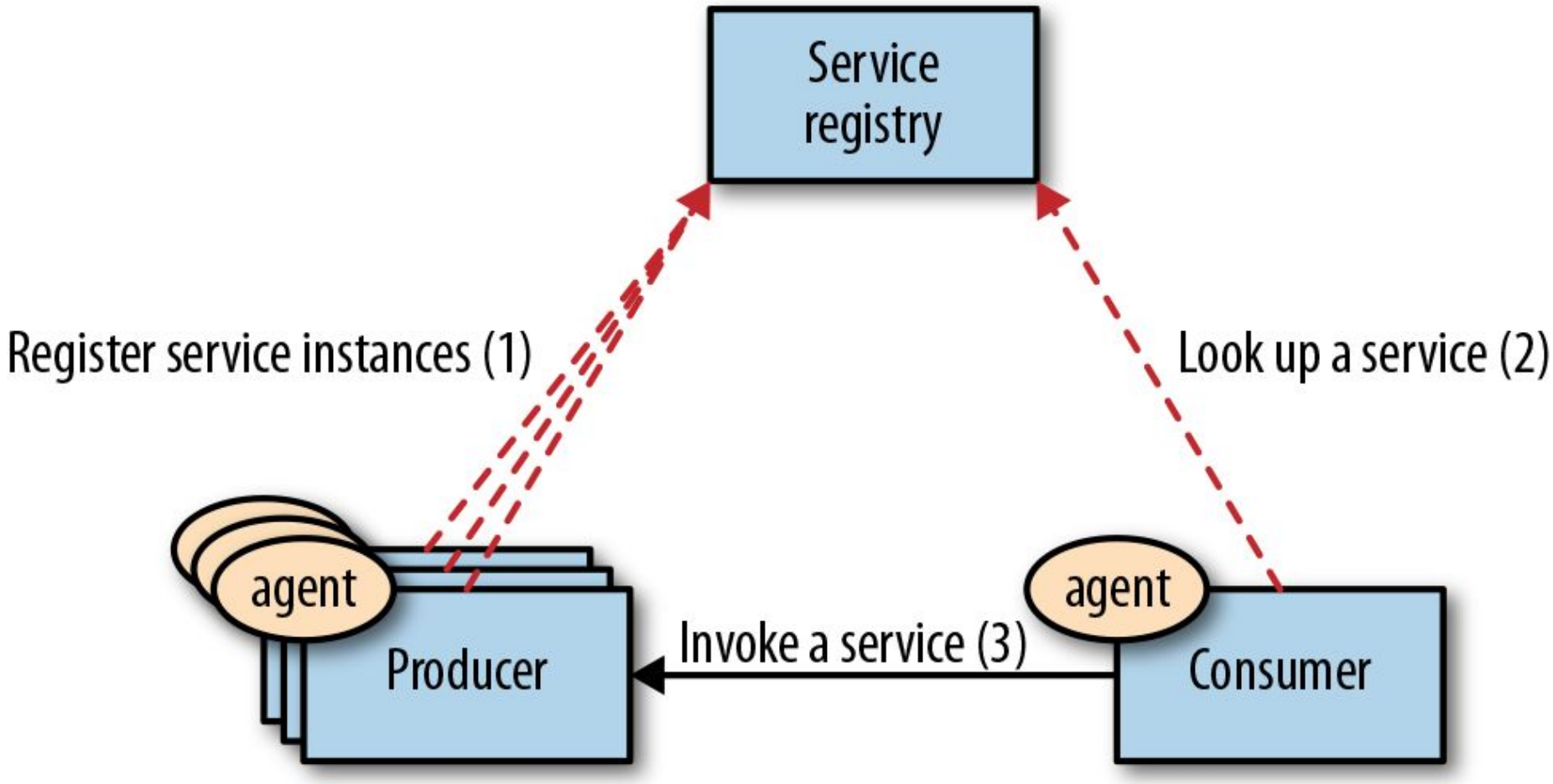
Behavioral Patterns



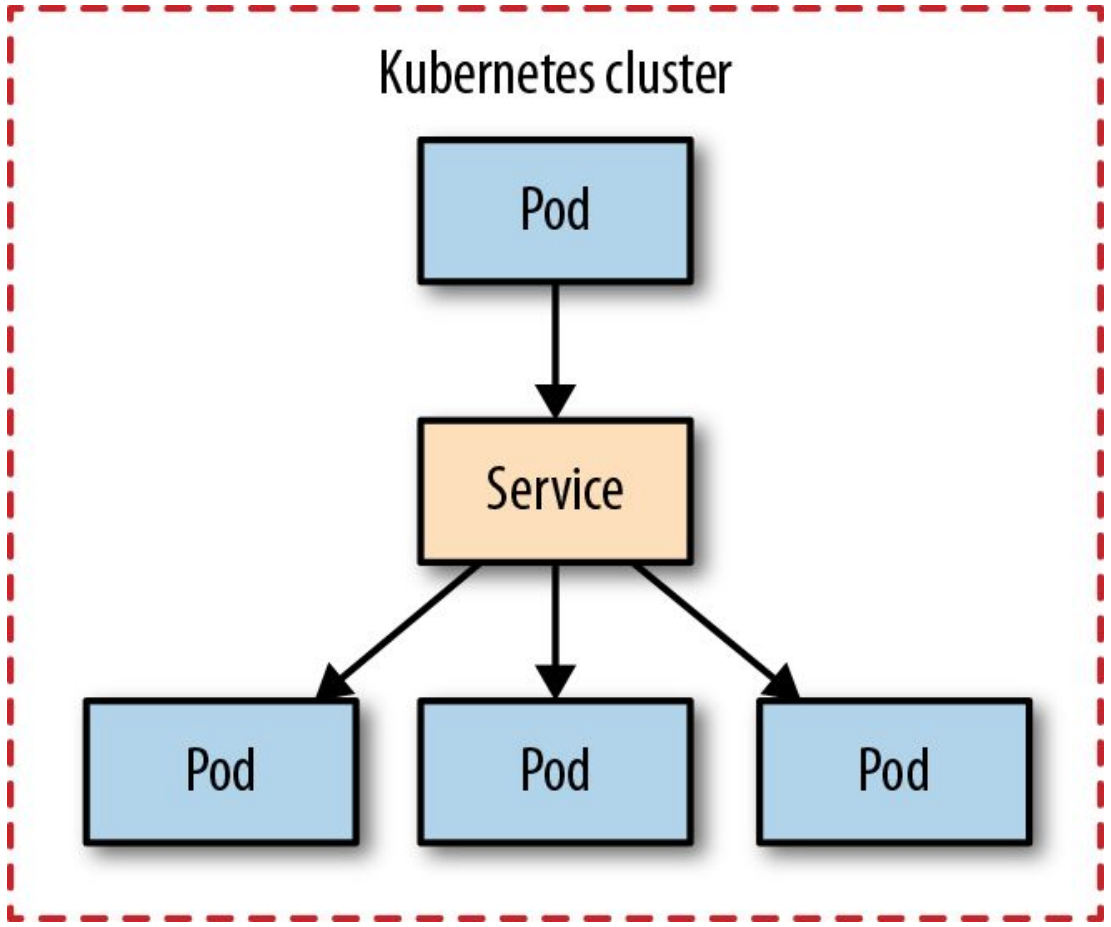

Service
Discovery

How to discover and use services

Client-side Service Discovery (non Kubernetes)



Internal Service Discovery

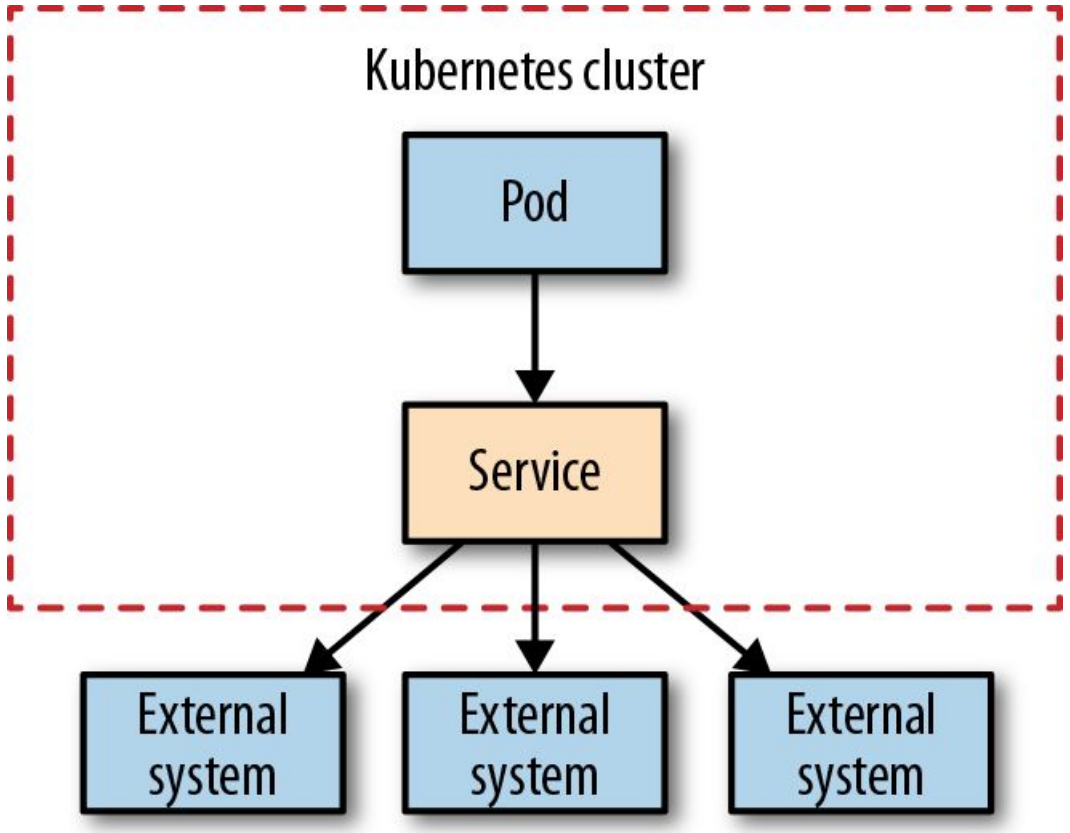


- Discovery through DNS lookups
- Pods picked by label selector
- Multiple ports per Service
- Session affinity on IP address possible
- Successful readiness probes required for routing
- Virtual IP address for each Service

Service

```
apiVersion: v1
kind: Service
metadata:
  name: random-generator
spec:
  selector:
    app: random-generator
  ports:
  - port: 8080
    protocol: TCP
    targetPort: 8080
```

Manual Service Discovery

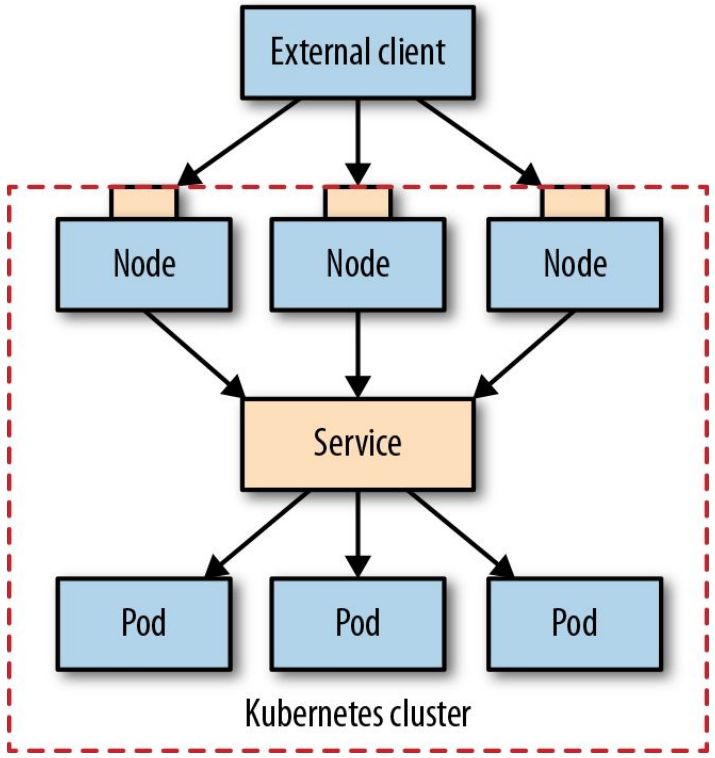


- Service without selector
- Manually creating Endpoint resource with the same name as the Service
- Service of type **`ExternalName`** map are registered as DNS CNAMEs

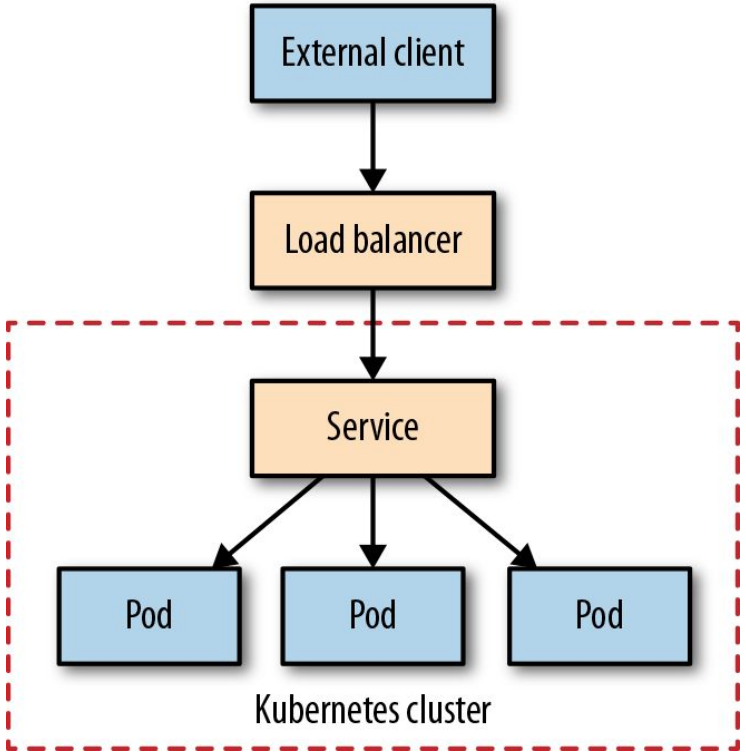
Endpoints

```
apiVersion: v1
kind: Endpoints
metadata:
  name: external-service
subsets:
- addresses:
  - ip: 1.1.1.1
  - ip: 2.2.2.2
  ports:
  - port: 8080
```

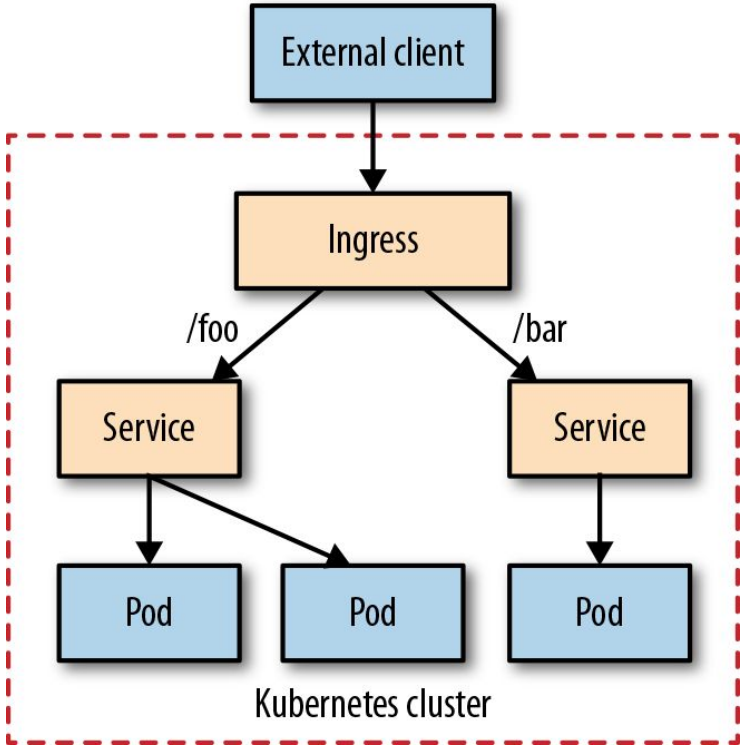
Node Port



Load Balancer



Ingress

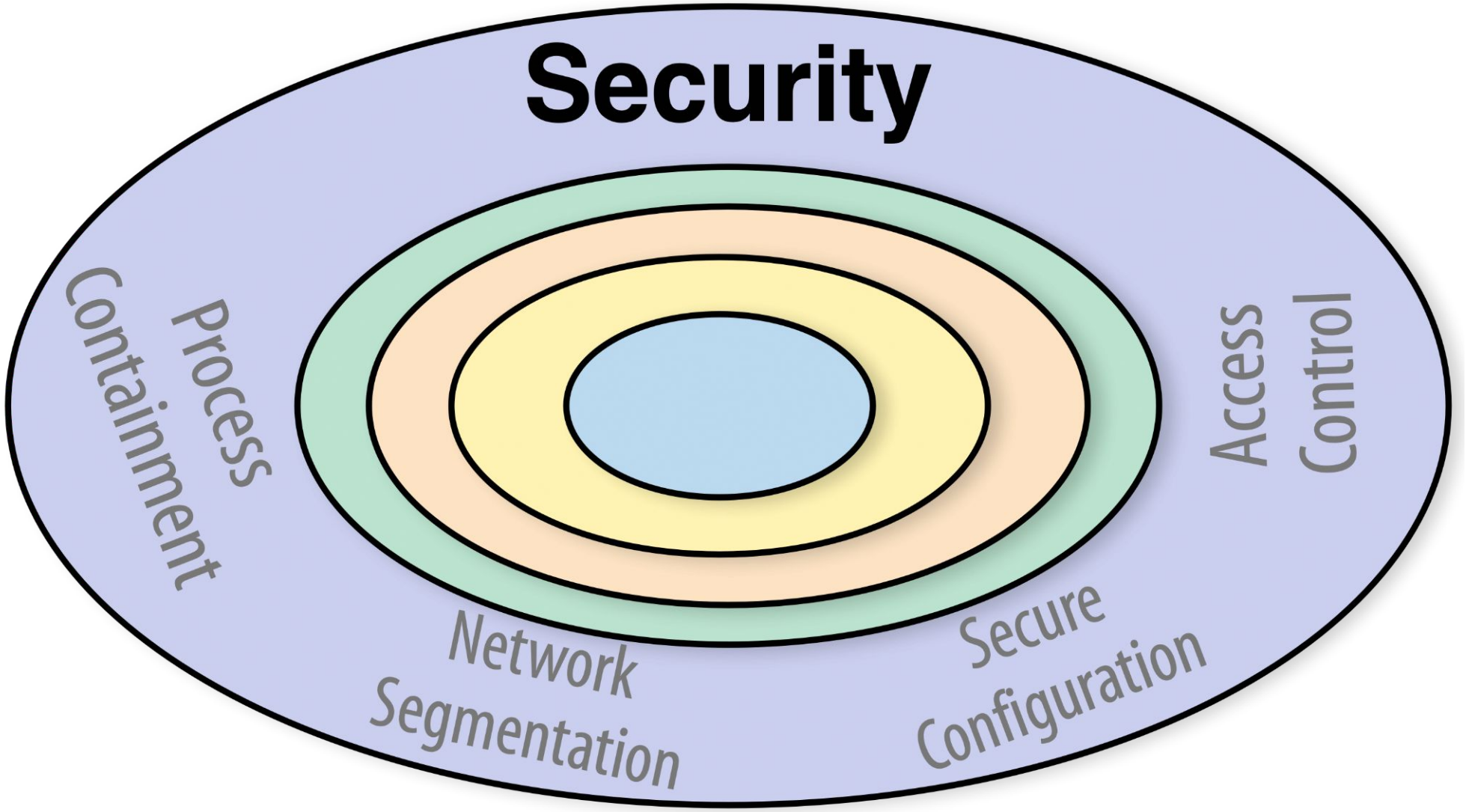


Ingress

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: random-generator
spec:
  rules:
  - http:
      paths:
      - path: /
        backend:
          serviceName: random-generator
          servicePort: 8080
      - path: /cluster-status
        backend:
          serviceName: cluster-status
          servicePort: 80
```

Service Discovery

Name	Configuration	Client type	Summary
ClusterIP	`type: ClusterIP` `.spec.selector`	Internal	The most common internal discovery mechanism
Manual IP	`type: ClusterIP` `kind: Endpoints`	Internal	External IP discovery
Manual FQDN	`type: ExternalName` `.spec.externalName`	Internal	External FQDN discovery
Headless Service	`type: ClusterIP` `.spec.clusterIP: None`	Internal	DNS-based discovery without a virtual IP
NodePort	`type: NodePort`	External	Preferred for non-HTTP traffic
LoadBalancer	`type: LoadBalancer`	External	Requires supporting cloud infrastructure
Ingress	`kind: Ingress`	External	L7/HTTP-based smart routing mechanism


Security
Process Containment
Network Segmentation
Secure Configuration
Access Control

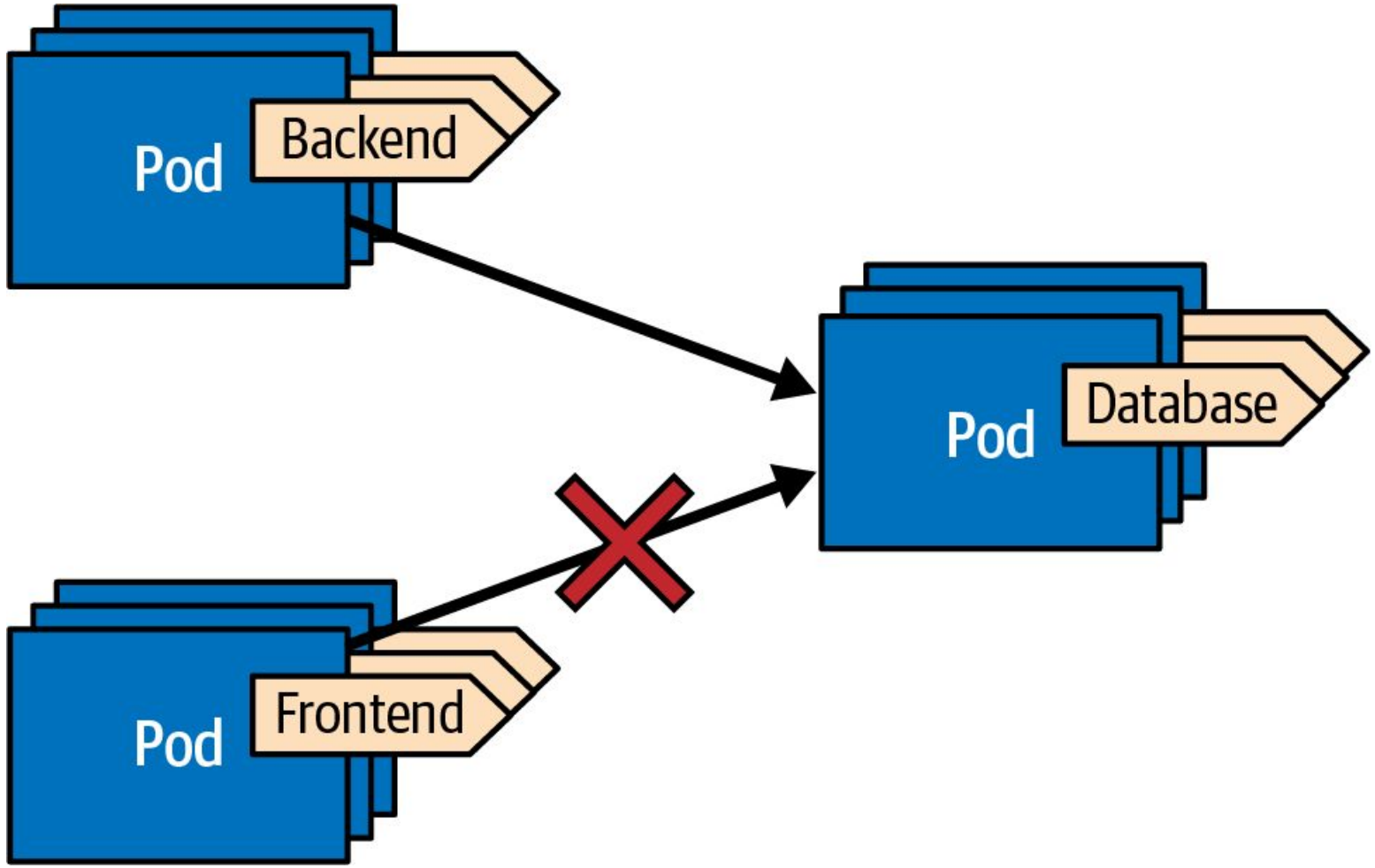

Network Segmentation

How to isolate network traffic for multiple tenants

Network Policy

- Essential Kubernetes resource for network segmentation
- Defines custom inbound/outbound rules for Pods
  - **ingress**: Rules for regulating innbound traffic
  - **egress**: Rules for regulating outgoing traffic
- Uses Pod selectors to apply policies via labels
  - **direct**: Directly configure individual Pods by their id
  - **role-based:** Use roles for flexibly reconfiguring policies
- Requires a **CNI plugin** that supports NetworkPolicies
  - NetworkPolicies operate on OSI Level 3/4
- NetworkPolicy objects are **namespace-scoped**

Network Policy



Network Policy

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: allow-database
spec:
  podSelector:
    matchLabels:
      app: chili-shop
      id: database
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: chili-shop
          id: backend
```

Role-based Network Policy

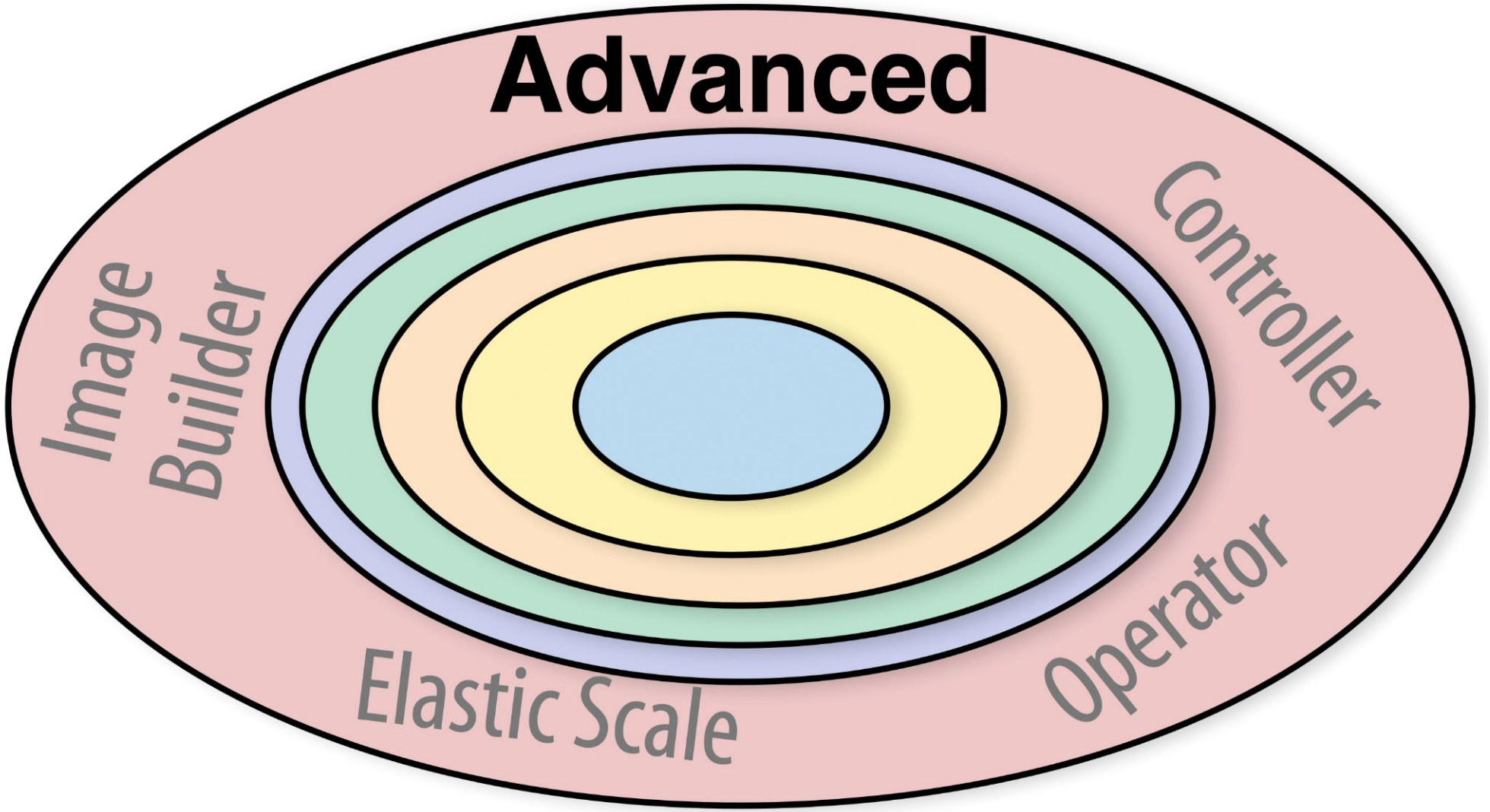
```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: allow-database-client
spec:
  podSelector:
    matchLabels:
      app: chili-shop
      id: database
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: chili-shop
          role-database-client: 'true'
```

Authorization Policy

- OSI Level 7 network control (HTTP)
  - provided by Kubernetes Add-Ons, e.g. ServiceMesh like Istio
- Policy components:
  - **Selector**: Pod targeting
  - **Action**: Traffic handling
  - **Rule List**: Action conditions
- Typically used with default deny-all policy
- Allows application-level authorization
  - “Access Control” for securing Kubernetes API server

Authorization Policy

```
apiVersion: security.istio.io/v1beta1
kind: AuthorizationPolicy
metadata:
  name: prometheus-scraper
  namespace: istio-system
spec:
  selector:
    matchLabels:
      has-metrics: "true"
  action: ALLOW
  rules:
  - from:
    - source:
        namespaces: ["prometheus"]
    to:
    - operation:
        methods: [ "GET" ]
        paths: ["/metrics/*"]
```


Advanced
Image Builder
Controller
Elastic Scale
Operator

Controller

How to get from the current state to a declared target state

State Reconciliation

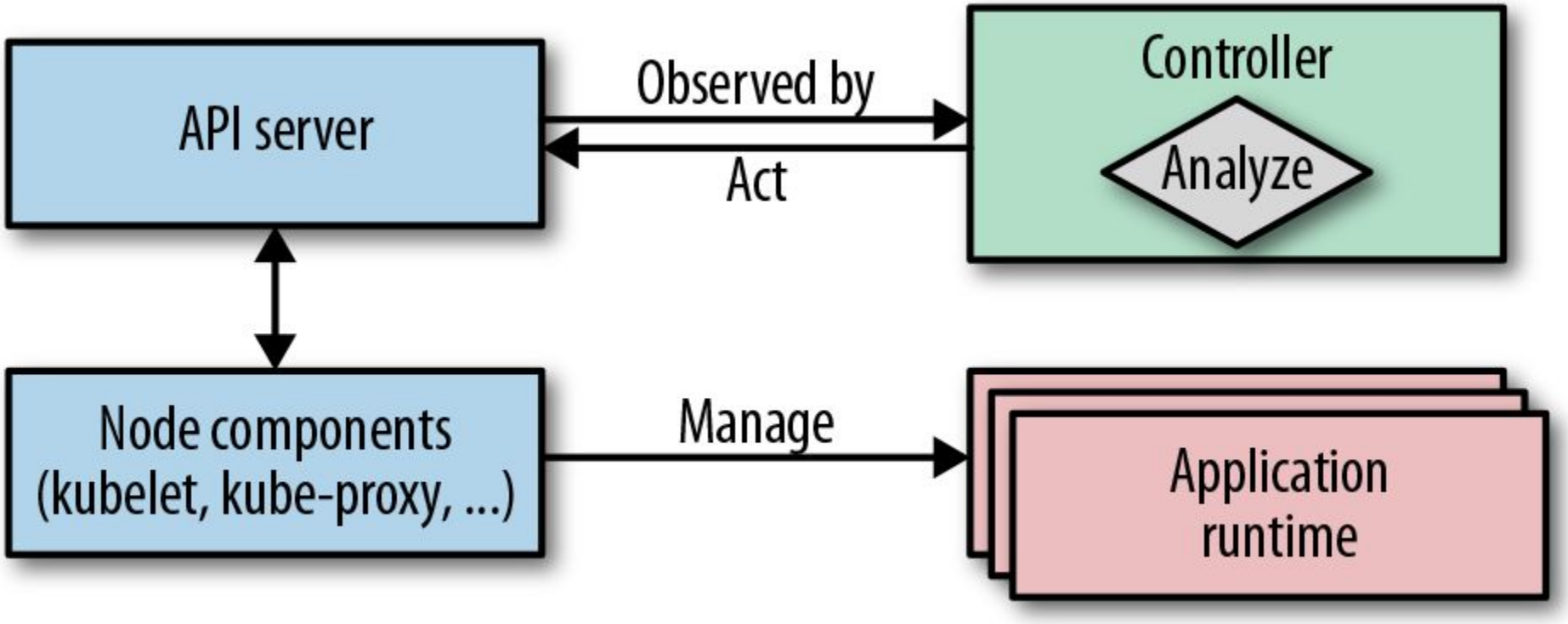
- Kubernetes as distributed state manager
- Make the **actual** state more like the declared **target** state.

🔎 ***Observe*** - Discover the actual state

🤔 ***Analyze*** - Determine difference to target state

🔨 ***Act-*** Perform actions to drive the actual to the desired state

Observe - Analyze - Act



Common Triggers

- Labels
  - Indexed by backend
  - Suitable for selector-like functionality
  - Limitation on characterset for names and values
- Annotations
  - No syntax restrictions
  - Not indexed
- ConfigMaps
  - Good for complex structured state declarations
  - Simple alternative to CustomResourceDefinitions

ConfigMap Watch Controller

```
namespace=${WATCH_NAMESPACE:-default}
base=http://localhost:8001
ns=namespaces/$namespace

curl -N -s $base/api/v1/${ns}/configmaps?watch=true | \
while read -r event
do
  type=$(echo "$event" | jq -r '.type')
  config_map=$(echo "$event" | jq -r '.object.metadata.name')
  annotations=$(echo "$event" | jq -r '.object.metadata.annotations')

  if [ $type = "MODIFIED" ]; then
    # Restart Pods using this ConfigMap
    # ...
  fi
done
```


Operator

How to encapsulate operational knowledge into executable software

Definition

> An **operator** is a Kubernetes **controller** that understands two domains: Kubernetes and *something else*. By combining knowledge of both areas, it can **automate** tasks that usually require a human operator that understands both domains.
>
> **Jimmy Zelinskie**
> http://bit.ly/2Fjlx1h

Technical:

Operator = Controller + CustomResourceDefinition

CustomResourceDefinition

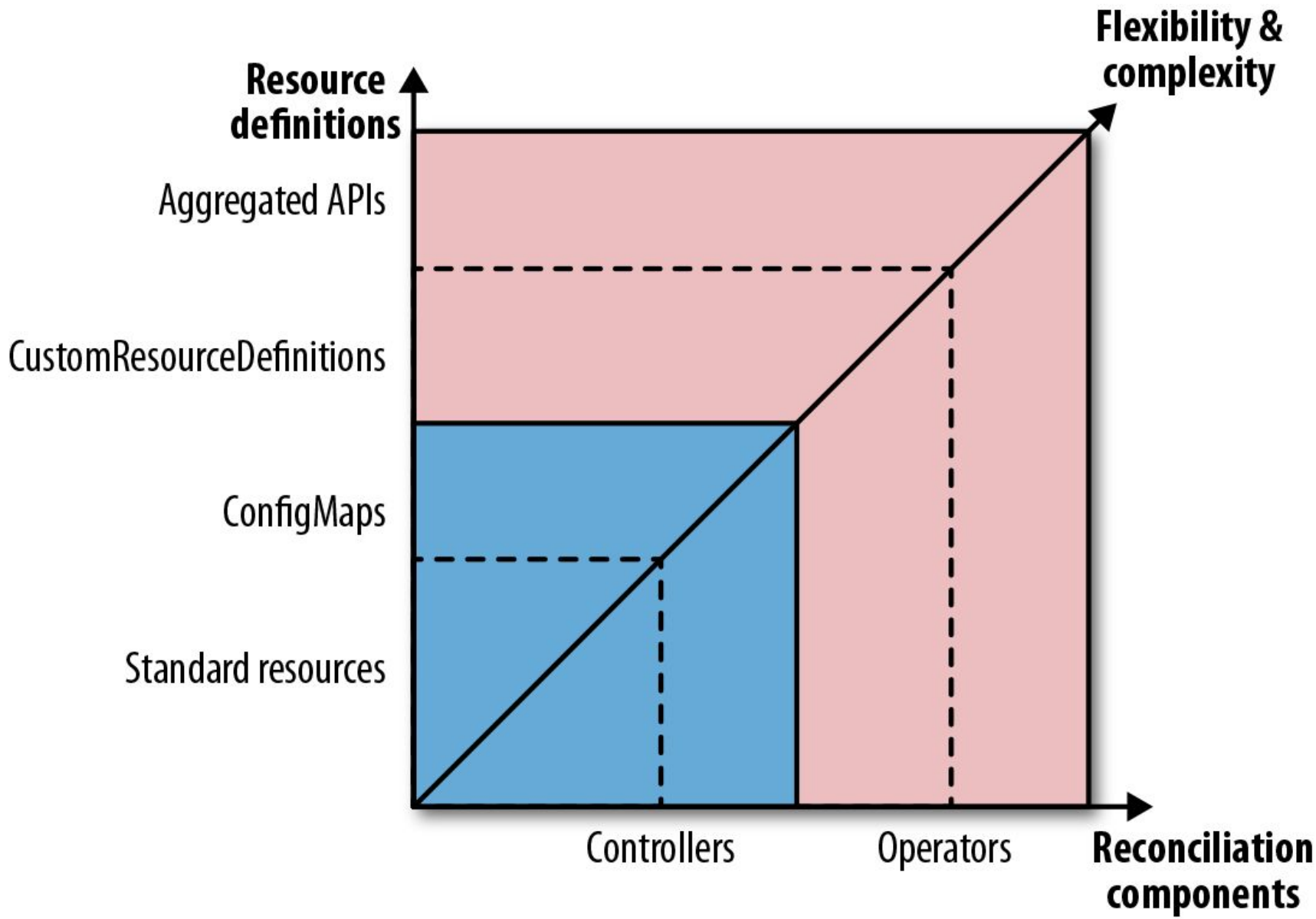
Custom resource is modelling a custom domain and managed through the Kubernetes API

```
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: configwatchers.k8spatterns.io
spec:
  scope: Namespaced
  group: k8spatterns.io
  version: v1
  names:
    kind: ConfigWatcher
    plural: configwatchers
  validation:
    openAPIV3Schema:
    ...
```

Custom Resource

Operator

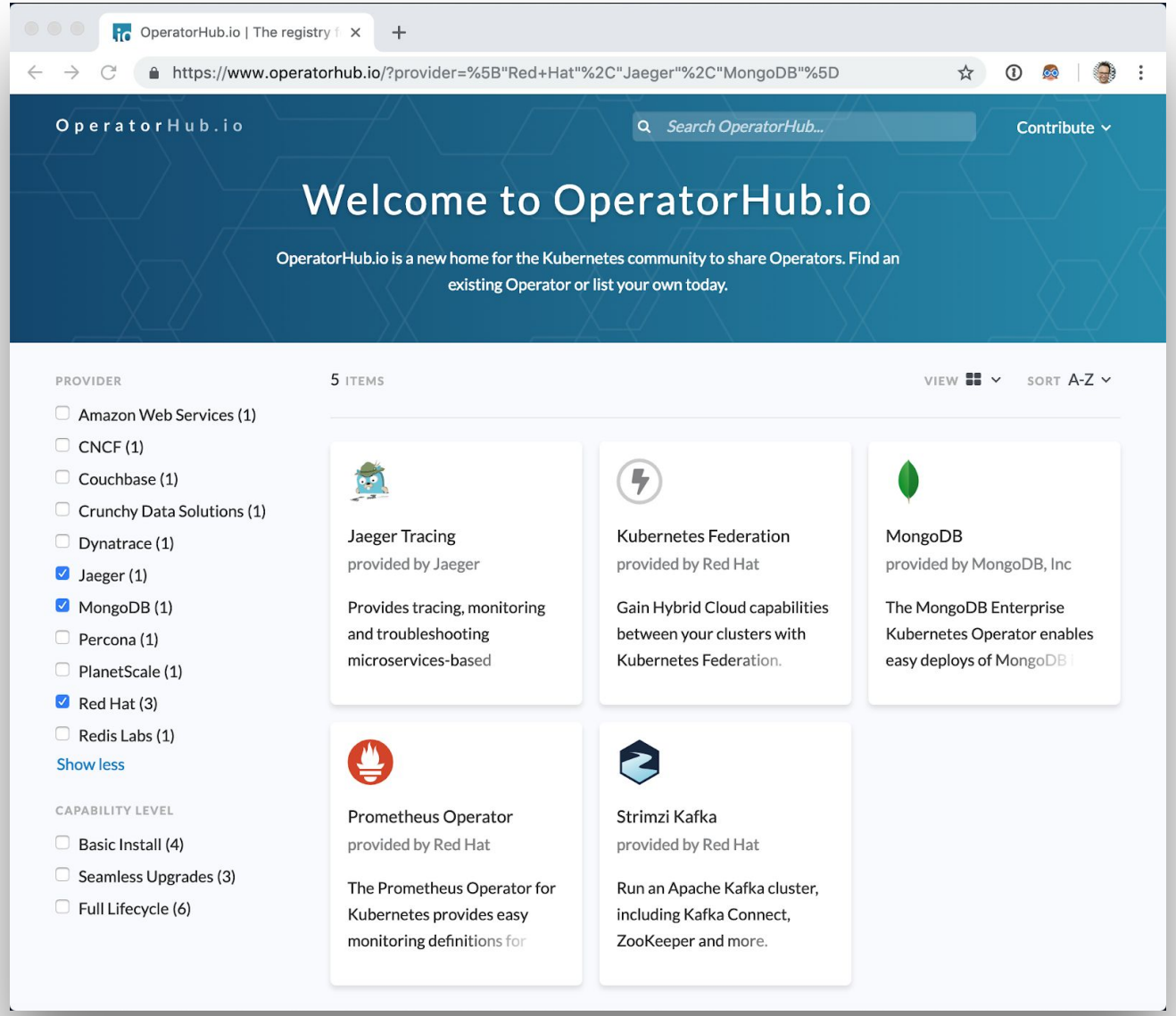
```
kind: ConfigWatcher
apiVersion: k8spatterns.io/v1
metadata:
  name: webapp-config-watcher
spec:
  configMap: webapp-config
  podSelector:
    app: webapp
```


Flexibility & complexity
Resource definitions
Aggregated APIs
CustomResourceDefinitions
ConfigMaps
Standard resources
Controllers
Operators
Reconciliation components

CRD Classification

- Installation CRDs
  - Installing and operating applications
  - Backup and Restore
  - Monitoring and self-healing
  - Example: Prometheus for installing Prometheus & components
- Application CRDs
  - Application specific domain concepts
  - Example: ServiceMonitor for registering Kubernetes service to be scraped by Prometheus

Operator Hub



Red Hat

Operator Development

- Operator can be implemented in any language
- Frameworks:
  - Operator Framework (Golang, Helm, Ansible, Java) https://github.com/operator-framework
  - Kubebuilder (Golang) https://github.com/kubernetes-sigs/kubebuilder
  - Metacontroller (Language agnostic) https://metacontroller.app/

Demo

Thank you

https://k8spatterns.io

@ro14nd@hachyderm.io

@bibryam

Red Hat

Picture Credits

https://www.pexels.com/photo/brown-and-black-pattern-2158386/
https://pixabay.com/photos/ship-helm-sunset-cutter-coast-guard-759954/
https://unsplash.com/photos/yo01Z-9HQAw
https://www.pexels.com/photo/turned-on-light-crane-1117211/
https://www.pexels.com/photo/shallow-focus-photography-of-black-ship-1095814/
https://pixabay.com/photos/containers-storage-rusted-rusty-1209079/
https://pixabay.com/photos/motocross-sidecar-race-motorsport-1045661/
https://www.pexels.com/photo/golden-gate-bridge-san-francisco-california-1141853/
https://unsplash.com/photos/M_I-crkDO-k
https://unsplash.com/photos/FqnkBeq1wGg
https://www.pexels.com/de-de/foto/brown-brick-roof-auf-luftaufnahme-3637890/
https://unsplash.com/de/fotos/yfq6c6j-5WI
https://unsplash.com/de/fotos/3wPJxh-piRw
https://unsplash.com/de/fotos/0ji5tjZQ2I4
https://www.pexels.com/de-de/foto/weisse-tasse-auf-dem-tisch-53405/
https://www.pexels.com/photo/reflection-playstation-pad-gaming-18174/
https://unsplash.com/photos/UJP_QpCKj7M
https://www.pexels.com/photo/grayscale-photo-of-person-holding-chess-piece-1498958/
https://pixabay.com/photos/cans-manufacturing-business-2888650/
https://unsplash.com/photos/IRoX0shwjUQ
https://www.freeimages.com/photo/poppy-in-wheat-1344010
https://pixabay.com/photos/telescope-field-glass-spyglass-1966366/
https://unsplash.com/photos/UHDx3BHIFvY
https://pixabay.com/photos/bake-advent-christmas-cookie-1786926/
https://pixabay.com/photos/balloons-colors-party-celebration-1869790/
https://unsplash.com/photos/ymf4_9Y9S_A
https://unsplash.com/photos/ZUabNmumOcA
https://pixabay.com/photos/lost-places-machines-old-factory-3991951/
https://pixabay.com/images/id-407101/
https://www.pexels.com/photo/backlit-beach-dawn-dusk-588561/