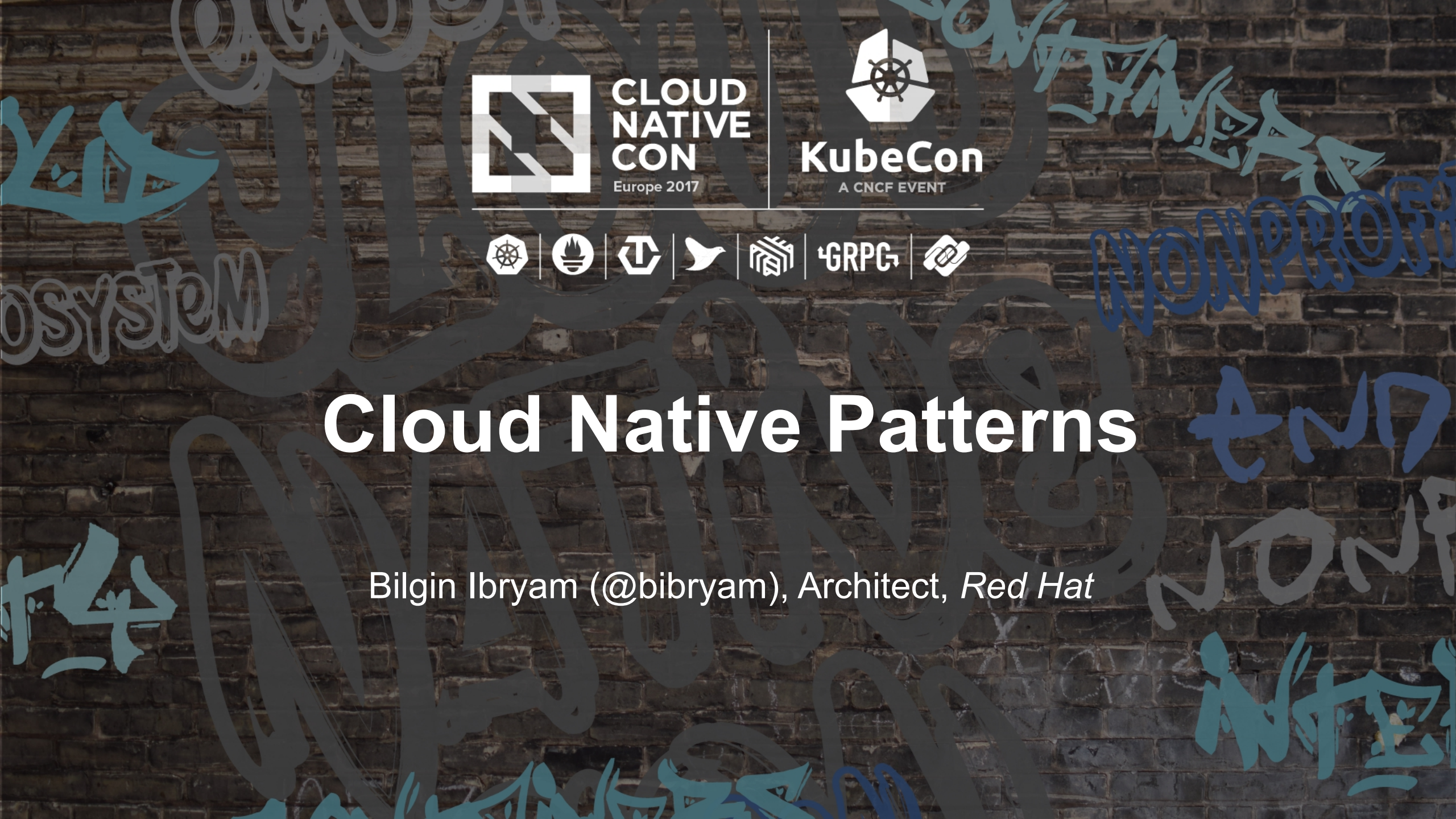


Cloud Native Patterns

Bilgin Ibryam (@bibryam), Architect, *Red Hat*

Cloud Native in Practice

Applications adopting the principles of
Microservices packaged as
Containers orchestrated by
Platforms running on top of
Cloud infrastructure

Cloud Native Platforms

OPENSHIFT

AWS ECS

Nomad

NETFLIX | OSS

TECTONIC by CoreOS

MESOS

RANCHER

spring by Pivotal

kubernetes

docker SWARM

KONTENA

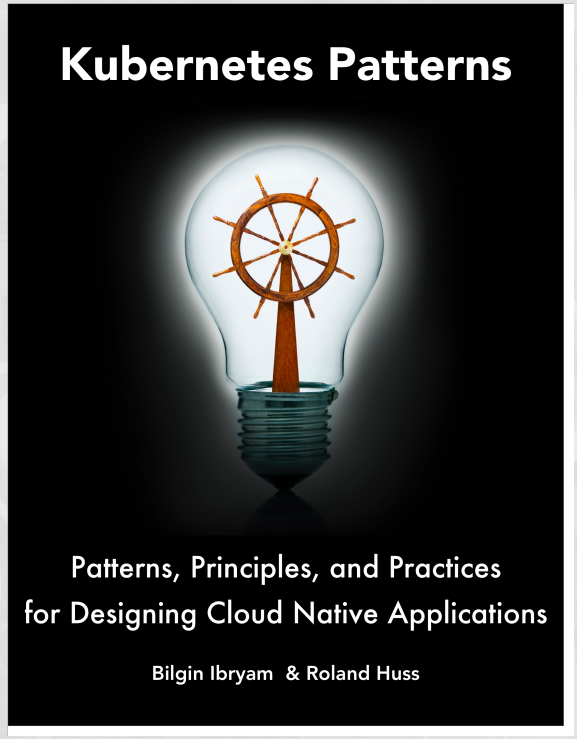
CLOUDFOUNDRY

Common Abstractions and Primitives

Application packaging (**Container**)
Deployment unit (**Pod**)
Declarative update/rollback (**Deployment**)
Application placement (**Scheduler**)
Artifact grouping (**Label**)
Resources isolation (**Container/Namespace**)
Service discovery & load balancing (**Service**)

10 Cloud Native...

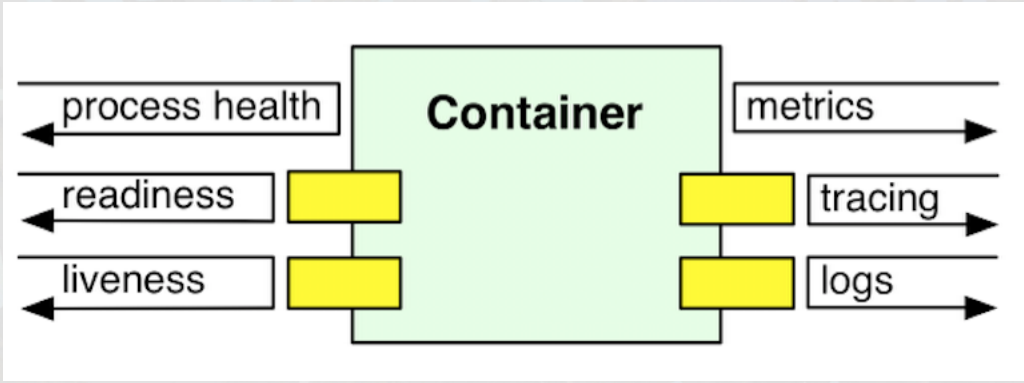
Principles
Patterns
Practices
Traits



http://leanpub.com/k8spatterns/

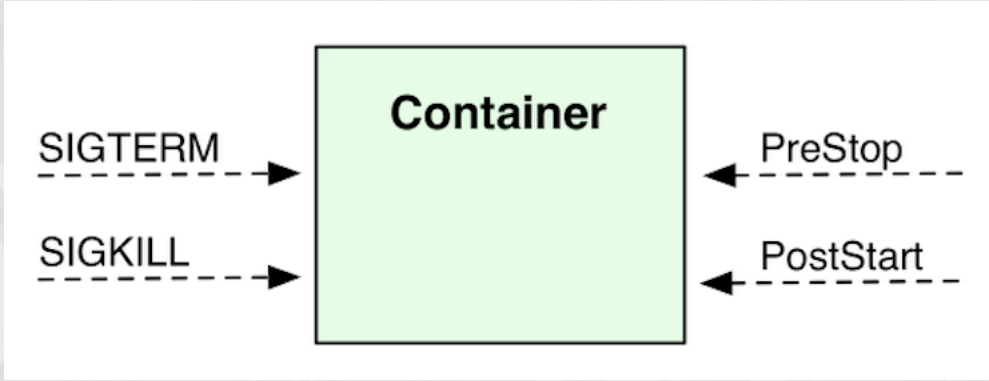
1. Observable Interior

Process Health Check
Application Readiness Health Check
Application Liveness Health Check
Metric collection
Log aggregation
Termination message

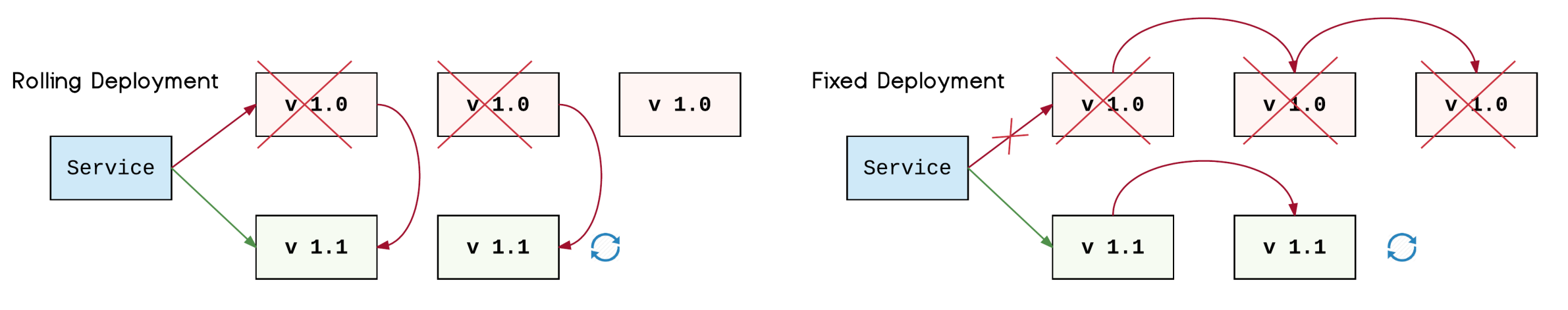


2. Life Cycle Conformance

SIGTERM
SIGKILL
PreStop
PostStart
ReleaseMemory (may be in the future)

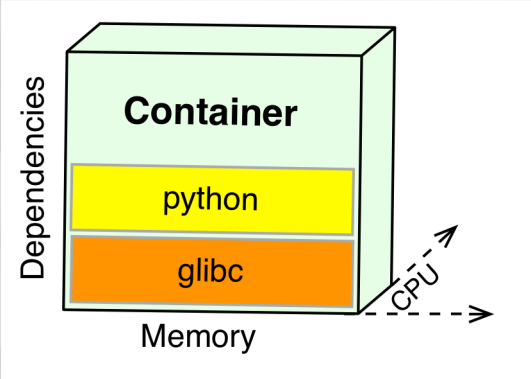


3. Declarative Updates

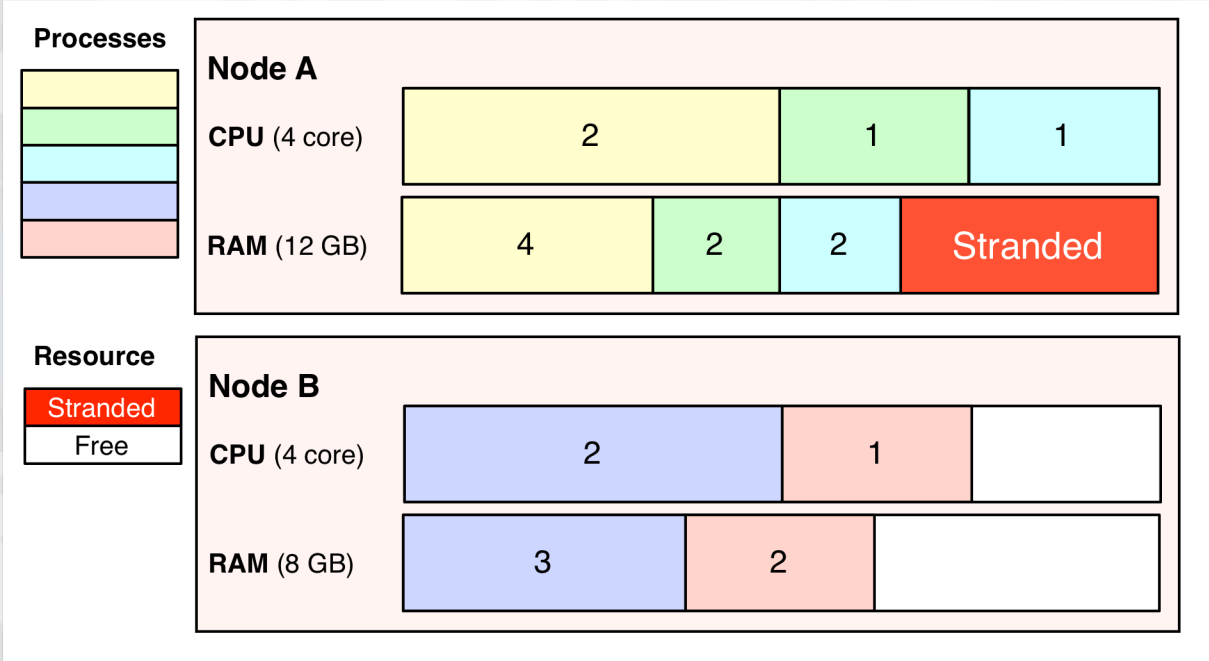


4. Predictable Resource Profile

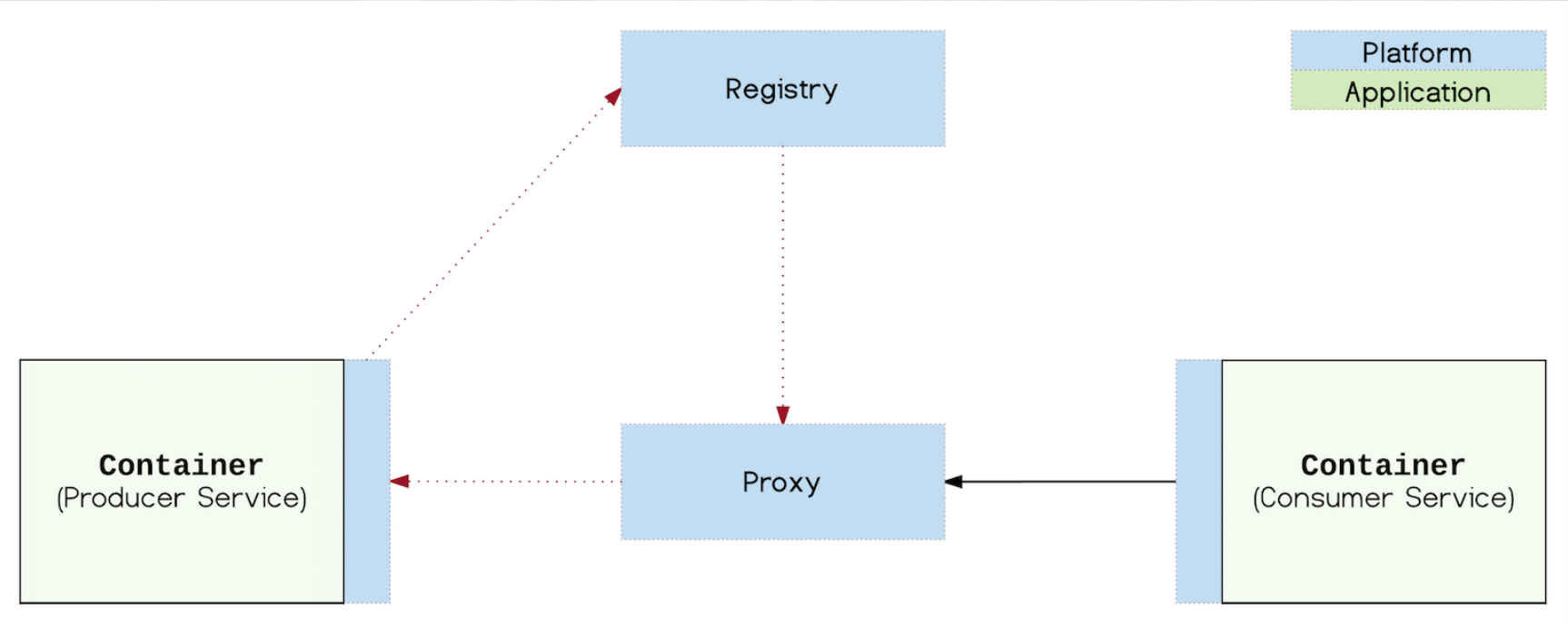
resources.limits.cpu
resources.limits.memory
resources.requests.cpu
resources.requests.memory
PersistentVolumeClaim



5. Dynamic Placement



6. Service Discovery & Load Balancing



7. (Scheduled) Batch Jobs

Example batch jobs in Java

JDK Timer
JDK ScheduledExecutorService
Quartz Scheduler
Spring Batch

Batch jobs in Kubernetes

8. Clustered Services

How to run a single HA instance of a service? → **Pod with 1 replica**
How to initialize an application with custom steps? → **Init Containers**
How to run a process on every node? → **DaemonSet**
How to manage a stateful service? → **StatefulSet**

9. Executable Application Manifest

Application binaries → Container
Deployment unit → Pod
Artifact grouping → Labels
Resource demands → request/limit/PVC
Configurations → ConfigMap/Secret
Resource management → Namespaces
Update/rollback mechanism → Deployment

AppManifest.yml

10. In Summary

1. Let the platform automate your routine tasks.
(placement, updates, healthchecks, self-healing, auto-scaling)

2. Move XFR/NFR from your application to the platform.
(service discovery, job mgmt, config mgmt, log aggregation, metric collection, etc)

3. Allow Developers to focus on the business domain.
(show creativity and talent to create great domain designs, hidden behind beautiful APIs)