

# STATEFUL APPLICATIONS IN KUBERNETES: READY FOR PRODUCTION!



**Niraj Tolia, Co-Founder**  
@nirajtolia / ntolia@kasten.io

**Julio Lopez, Member of Technical Staff**  
@julio5524 / julio@kasten.io






A grayscale photograph of a ship's bridge, showing a large steering wheel in the foreground, multiple electronic display screens in the background, and various control panels. The image is used as a background for the text.

# Kubernetes

**Container Orchestration:  
Automated Deployment, Scaling, & Management**



A loaf of rustic bread, possibly ciabatta, is the central focus. It is cut into two thick slices that are stacked on a folded, light-colored cloth with thin blue stripes. The bread sits on two thick, rectangular wooden boards. The top board is covered with a layer of small, golden-brown wheat seeds. The entire scene is set against a dark, textured background, with more wheat seeds scattered across the surface. The lighting is dramatic, highlighting the texture of the bread's crust and the softness of its interior.

**Kubernetes,  
the greatest  
thing since  
sliced bread?**



# kubernetes philosophy



## Developer and Application Focused

Puts the needs of the application and developer first and optimizes for agility

## Enforces Good DevOps Hygiene

Immutability, config as code, automation makes it easy to repave all infrastructure

## Declarative Approach

A robust systems approach where the state of the world is reconciled with the expectation





# key kubernetes features

## Self-Healing

Auto restart of unhealthy containers to match service levels



## Resource Utilization

Better bin packing for higher resource utilization



## Deployment Options

Variety of upgrade deployment strategies w/ rollback options



## Auto Scaling

Scale applications up and down in response to load



## Portability

Isolates developers and applications from infrastructure



## Service Discovery

Familiar IP and DNS-based service discovery and load balancing





App Definition and Development

Database and Data Warehouse

Vitess

ArangoDB

BIGCHAIN

CarbonData

cassandra

Cockroach LABS

Couchbase

CRATE.IO

druid

FoundationDB

FreeDB

IBM DB2

Iguazio

Infinispan

MariaDB

MEMSQL

Microsoft SQL Server

mongoDB

MySQL

neo4j

Qoms

ORACLE DATABASE

OrientDB

Pachyderm

pilosa

PostgreSQL

presto

Quobale

redis

RethinkDB

SCYLLA

snowflake

software

SQL Data Warehouse

STARBURST DATA

TIDB

VERTICA

YugaByte

Streaming

NATS

cloudevents

Amazon Kinesis

Bitbucket

GitHub

GitLab

OpenMessaging

PULSAR

RabbitMQ

StreamSets

Source Code Management

Bitbucket

GitHub

GitLab

OpenMessaging

PULSAR

RabbitMQ

StreamSets

Application Definition and Image Build

TELEPRESENCE

apache brooklyn

bitnami

DRAFT

habitat

HELM

Kaniko

ksnnnet

KubeVirt

minikube

On-Prem インフラ

OPENAPI INITIATIVE

Packer

Google Cloud Skaffold

YIPEE.IO

Continuous Integration / Continuous Delivery (CI/CD)

AppVeyor

argo

Bamboo

BRIGADE

buddybuild

Buildkite

circleci

Skycap

cloud bees

codefresh

CODESHIP

Concourse

ContainerOps

Drone

GITKUBE

GitLab RUNNER

go

Habitus

Jenkins

JenkinsX

Octopus Deploy

openstack. Zuul

Semaphore

shippable

Spinnaker

TeamCity

Travis CI

weave flux

wercker

DEPLOY

Orchestration & Management

Scheduling & Orchestration

kubernetes

Amazon ECS

SWARM

MESOS

Microsoft Azure Service Fabric

Nomad

Coordination & Service Discovery

CoreDNS

Apache Zookeeper

Consul

ContainerPilot

etcd

NETFLIX EUREKA

airbnb SmartStack

Service Management

GRPC

envoy

LINKERD

Open Policy Agent

SCALE

Ambassador

Apache Thrift

AVI Networks

backplane

CONDUIT

IS

HAPROXY

heptio Contour

HYSTRIX

Istio

Kong

NETFLIX RIBBON

NETFLIX ZUUL

NGINX

OPEN SERVICE BROKER API

traefik

TURBINE LABS

vamp

Runtime

Cloud-Native Storage

ROOK

ceph

CSI

DATERA

DELLEMC

Diamanti

GLUSTER

HATCHWAY

HEDVIG

portworx

PURE STORAGE

Quobyte

REX-Ray

ROBIN

Sheepdog

STORAGEOS

SWIFT

TRITON Object Storage

Container Runtime

containerd

rkt

cri-o

Google Cloud gVisor

Intel Clear Containers

kata

lxd

Pouch

runV

Singularity

SmartOS

Cloud-Native Network

CNI

alcide

aporeto

aviatrix

big switch

cilium

Contiv

CUMULUS

flannel

GuardCore

LIGATO

midonet

NSX

nuagenetworks

Open vSwitch

OPENCONTRAIL

PLUMgrid

ROMANA

weave net

Provisioning

Host Management / Tooling

ANSIBLE

CFEngine

CHEF

kube-bench

LINUXKIT

puppet

RUN DECK

SALTSTACK

StackStorm

Infrastructure Automation

BOSH

CLOUDIFY

InfraKit

juju

Kubernetes

ManageIQ

Terraform

Container Registries

Amazon ECR

Anno Registry

codefresh REGISTRY

REGISTRY

Google Container Registry

HARBOR

IBM Cloud

Jfrog Artifactory

Portus

ATOMIC

QUAY

Secure Images

anchore

aqua

BLACKBUCK

clair

Grafeas

NeuVector

OpenSCAP

Sonatype Nexus

Twistlock

Key Management

spiffe

SPIRE

CYBERARK CONJUR

KeyWhiz

KNOX

CONFIDENT

ORACLE POLICY AUTOMATION

Teleport

Vault

Cloud

Public

Alibaba Cloud

aws

Azure

Baidu Cloud

DigitalOcean

FUJITSU

Google Cloud

HUAWEI

IBM Cloud

Joyent

ORACLE

packet

Tencent Cloud

Private

DIGITAL REBAR

FOREMAN

MAAS

openstack

vmware

l.cncf.io

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

CLOUD NATIVE Landscape

CLOUD NATIVE COMPUTING FOUNDATION

Redpoint

Amplify

Platforms

Certified Kubernetes - Distribution

ASILESTACKS

AppsCode

caicloud

CANONICAL

CONTAINER RUNTIME

containerShip

DaoCloud

Diamanti

docker

Ghostcloud

Giant Swarm

IBM Cloud

inwinstack

JD.COM

Joyent

PHAROS

kube-spawn

Kubermatic

kublir

MESOSPHERE

MIRANTIS

navops

NetEase Cloud

OPENSIFT

ORACLE

Pivotal

PLATFORM

QFusion

RANCHER

STACKPOINT

SUSE

TECTONIC

Telekube

Typhoon

vmware

ise2c

Certified Kubernetes - Hosted

灵犀云

Alibaba Cloud

AmazonEKS

Amazon Container Service

Azure

Baidu Cloud

博云

CISCO

EasyStack

eking Technology

Google Kubernetes Engine

通云科技

HUAWEI

IBM Cloud

nirmata

ORACLE

SAP

Tencent Cloud

时道云

ZTE

Certified Kubernetes - Installer

APPENDIA

CoreOS

Gardener

Google Cloud

heptio

ORACLE

SAMSUNG SDS

Scaleway

Scaleworks

STRATOCLUSTER

SUPERGIANT

Monitoring

Prometheus

Amazon CloudWatch

APDYNAMICS

AppNeta

CloudHealth

COSCALE

DATADOG

dynatrace

Google Stackdriver

Grafana

graphite

HEATSHIELD

icinga

influxdb

INSTANA

IRONdb

Librato

LIGHTSTEP

Log Analytics

Nagios

NETSIL

NEW RELIC

NODESOURCE

OPENTECNO

Outlyer

sensu

SENTRY

SignalFx

StackRox

StackState

Sysdig

Thanos

weave cloud

weave scope

Logging

fluentd

elastic

graylog

Humio

LOGGLY

logz.io

loom

sematext

splunk

sumologic

Tracing

skywalking

Spring Cloud Sleuth

zipkin

Serverless

platform.sh

platformer.io

Scalingo

Tsuru

Cloud Native Landscape

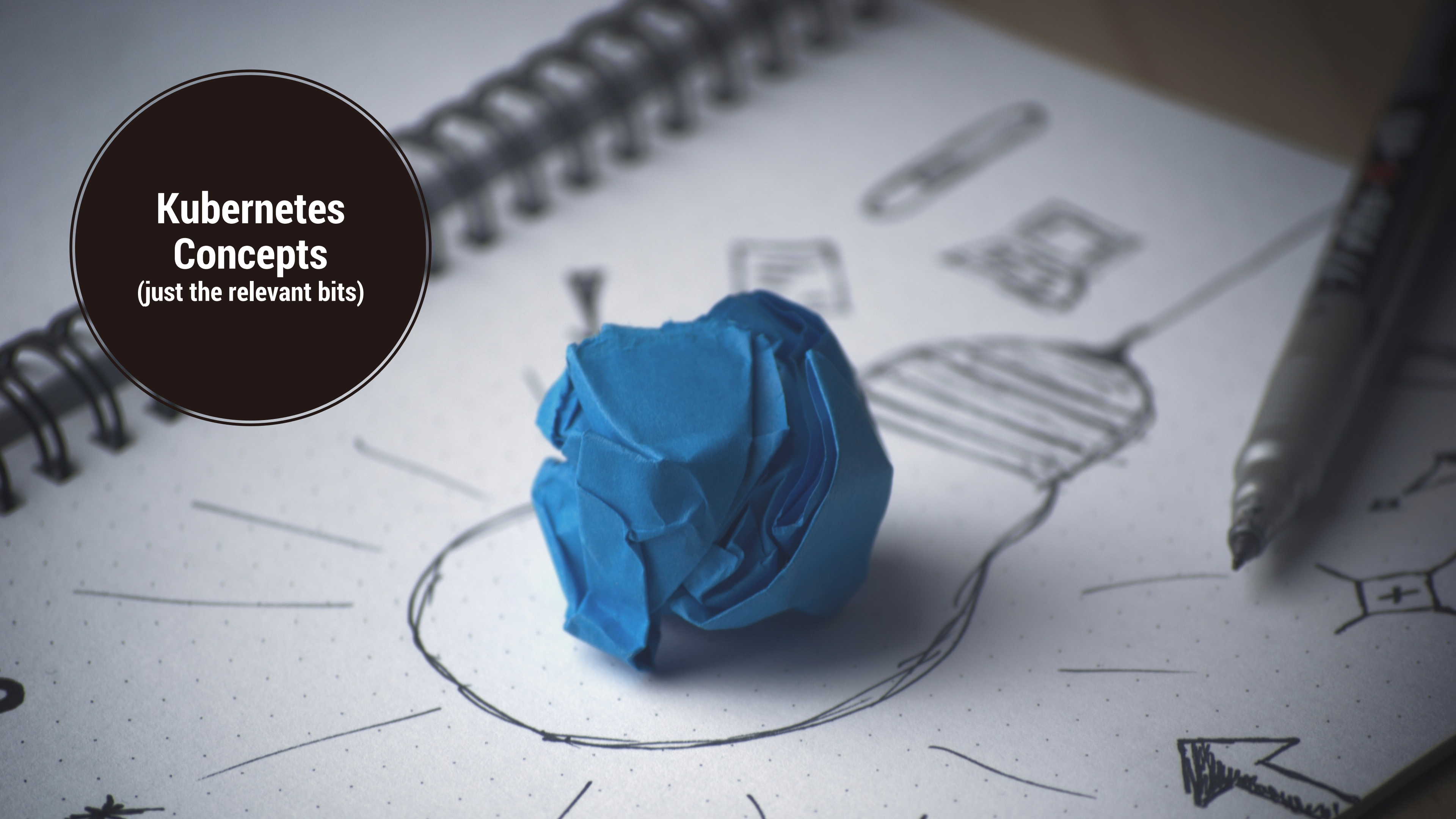
Cloud Native Computing Foundation

Redpoint

Amplify

# The Power of Community!



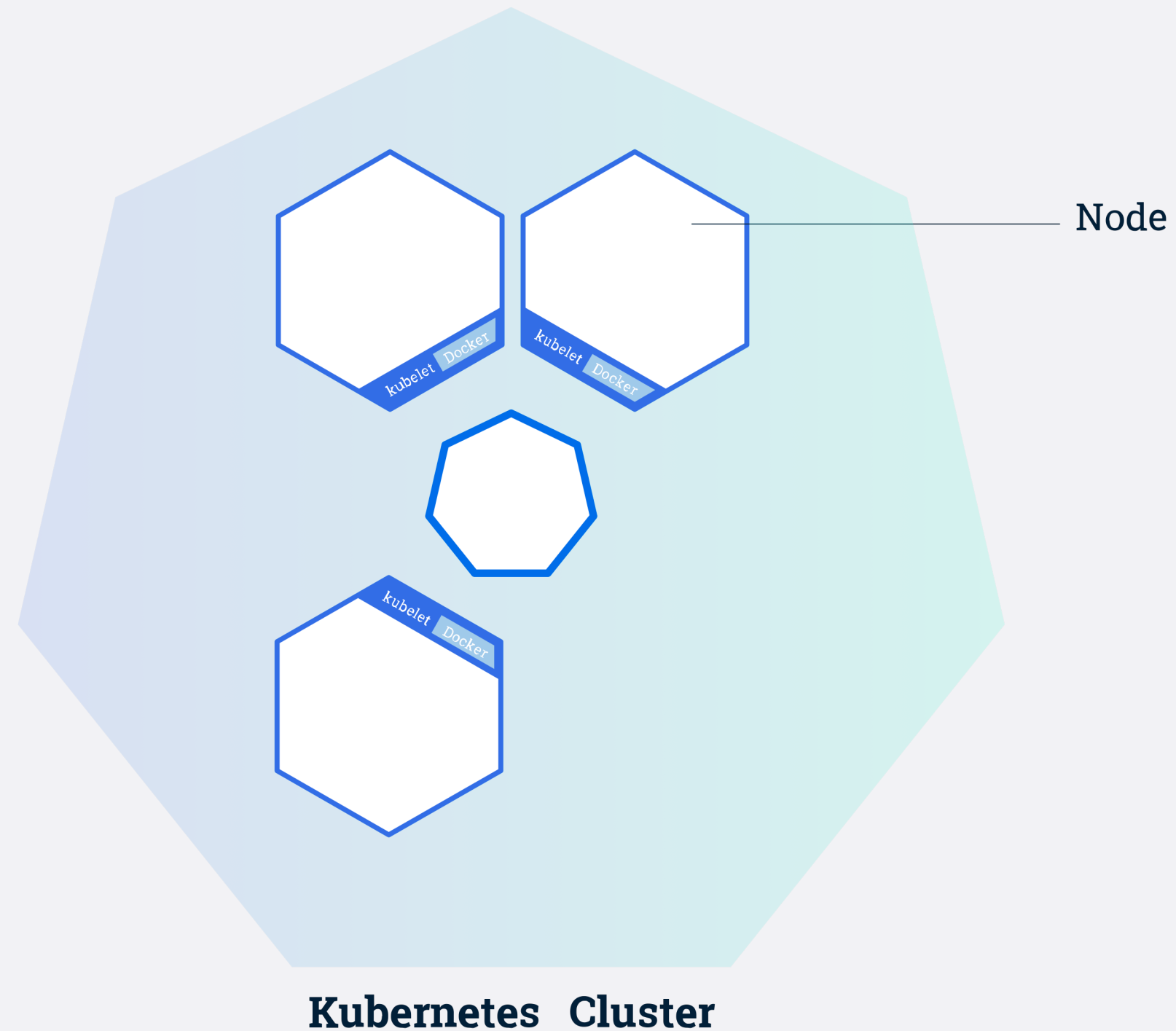
A crumpled blue paper ball sits in the center of a spiral-bound notebook page. The page is covered in faint, hand-drawn sketches, including a large circle around the paper ball, a speech bubble, and various geometric shapes. A pen lies on the right side of the page. The background is slightly blurred, showing the spiral binding of the notebook.

# Kubernetes Concepts

(just the relevant bits)

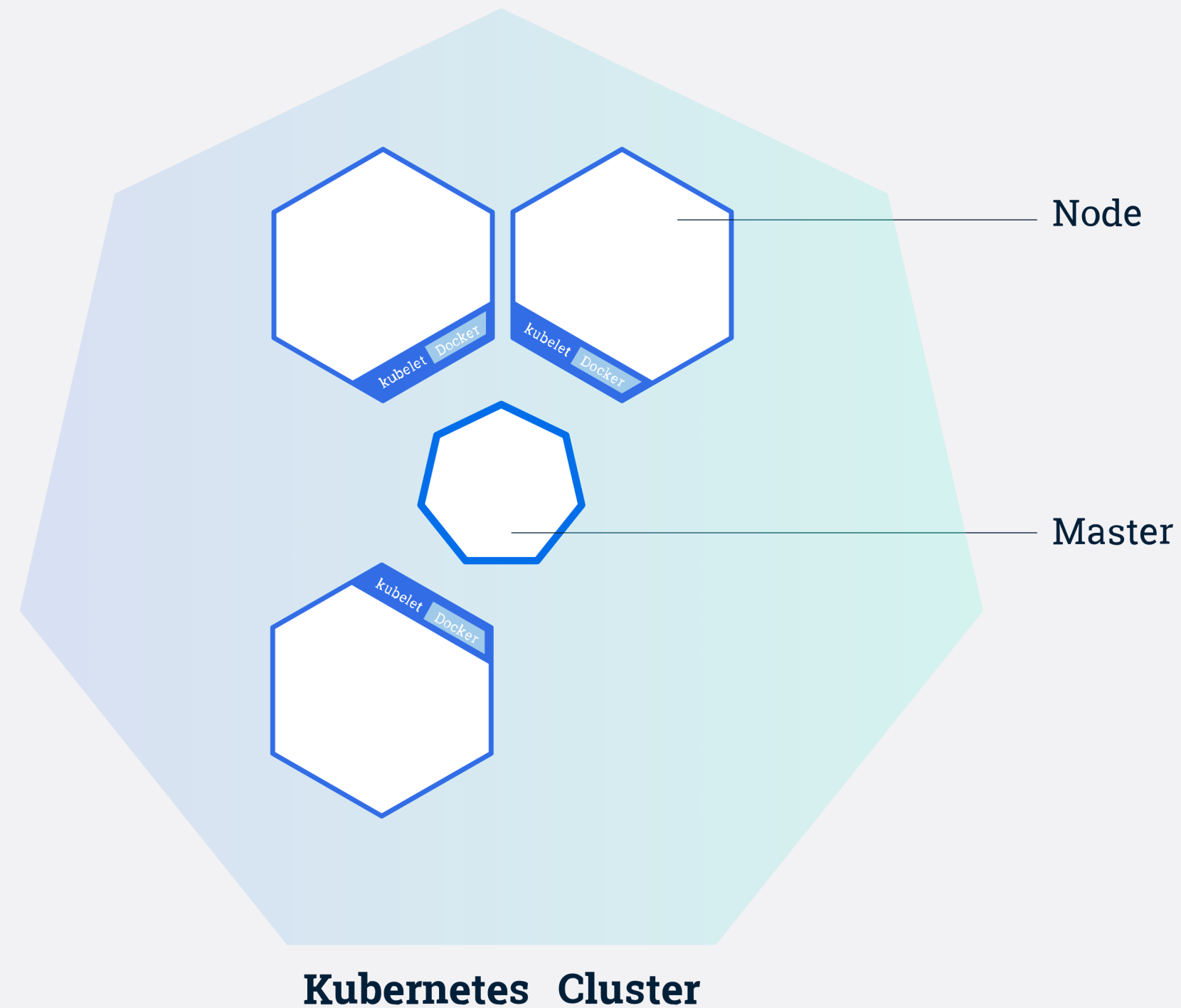


# (selected) kubernetes concepts – cluster + nodes



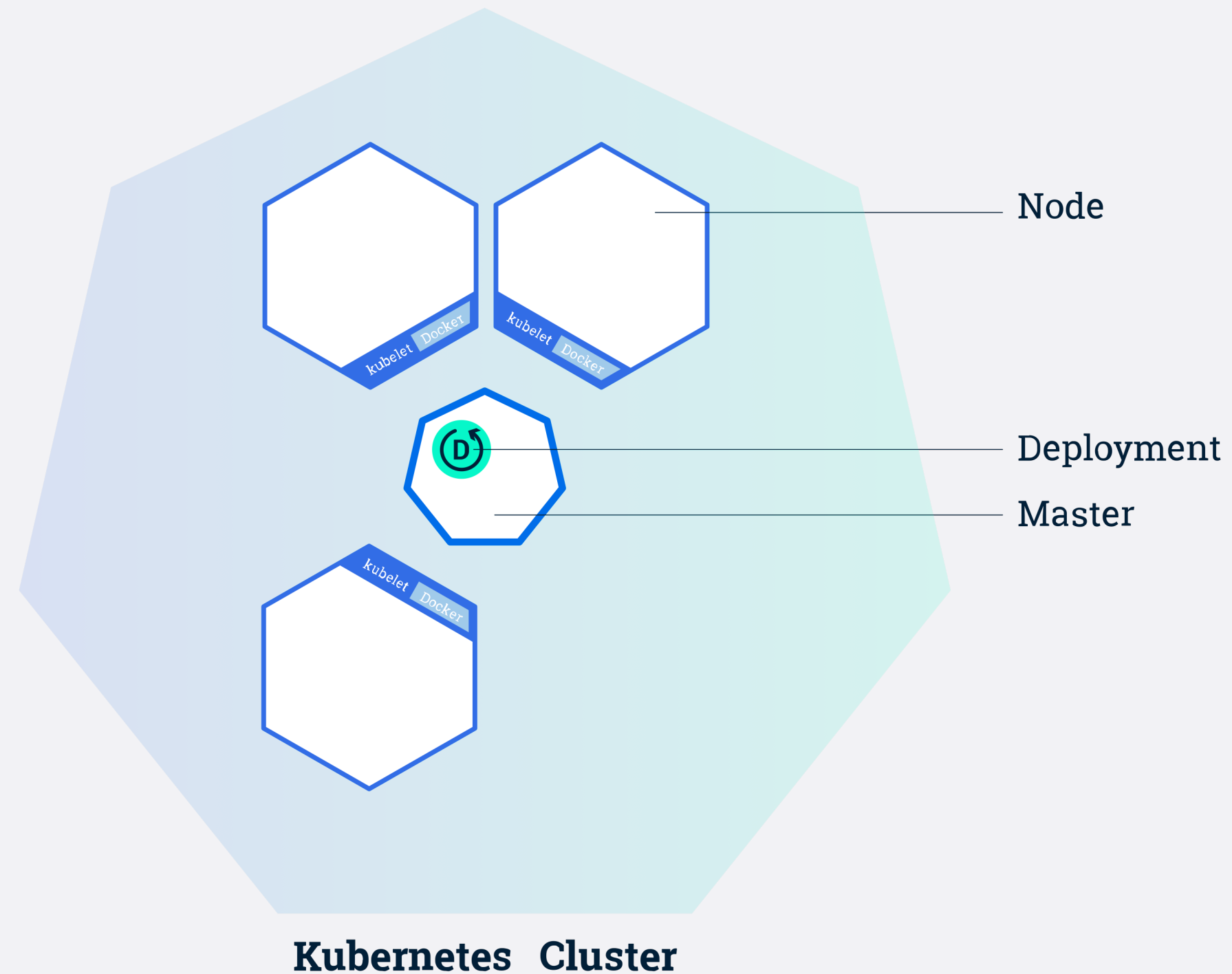


# (selected) kubernetes concepts – master node



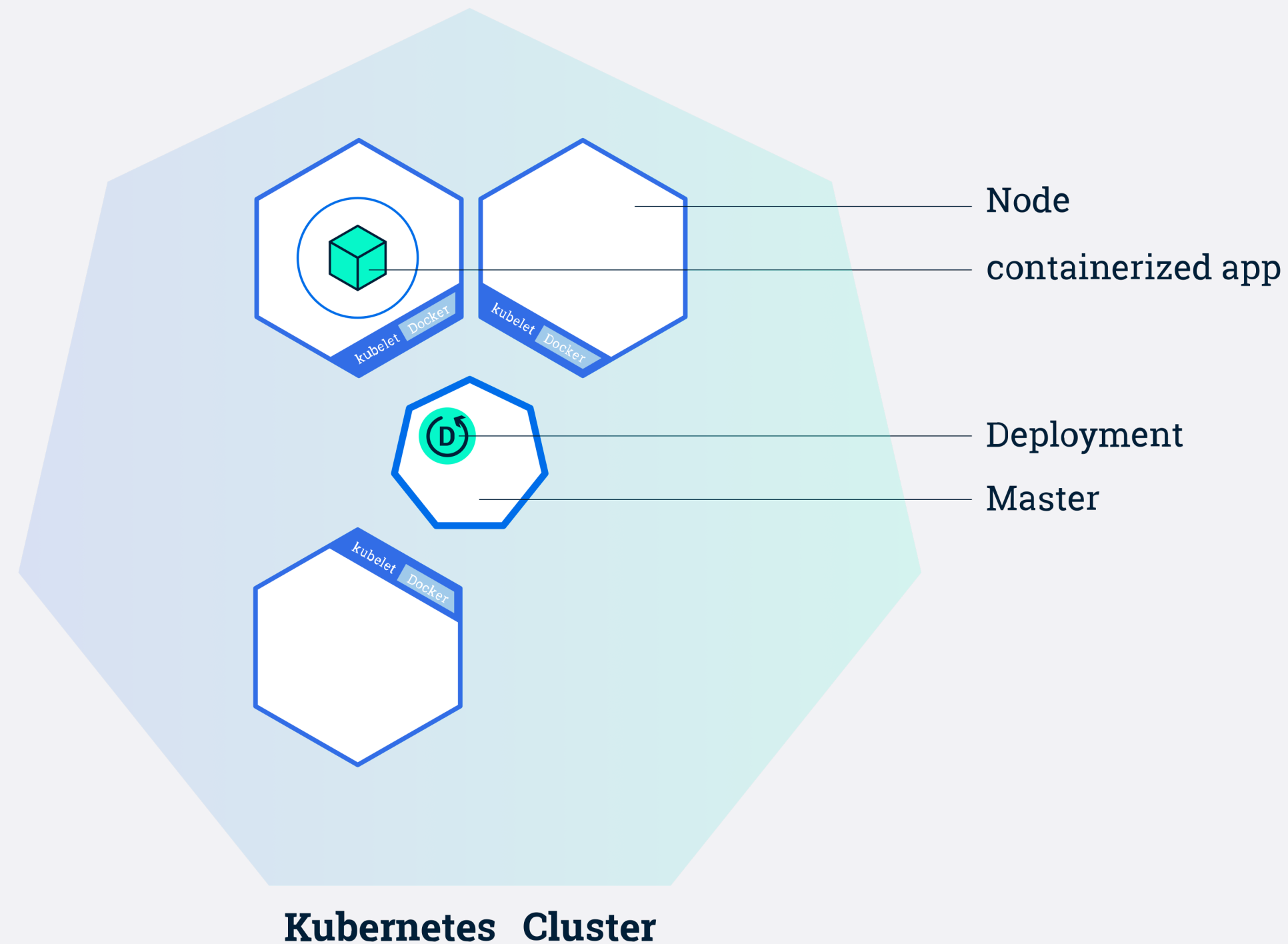


# (selected) kubernetes concepts – deployments





# (selected) kubernetes concepts – deployed app







# Storage Options for Kubernetes

**HOUSTON EXPRESS**  
**HAMBURG**  
IMO 9294891



# kubernetes portable storage abstractions

## file and block focus



*and more...*





# | dynamic storage provisioning for persistent storage



## 01 Self Service

Allow high developer velocity, no admin in the loop

## 02 Portable

No references to underlying storage provider. Allows application portability

## 03 On-Demand

Provisioned at time of use. Lifecycle can be tied to the application.





# | dynamic storage provisioning persistent volume (pv)



*A Persistent Volume (PV) represents provisioned storage in the cluster (e.g., NFS, iSCSI, other block, etc.). A PV's lifecycle is independent of the container/pod that uses it.*



# | dynamic storage provisioning

## | persistent volume claim (pvc)

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my-claim
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 8Gi
  storageClassName: ssd
```





# dynamic storage provisioning persistent volume claim (pvc)

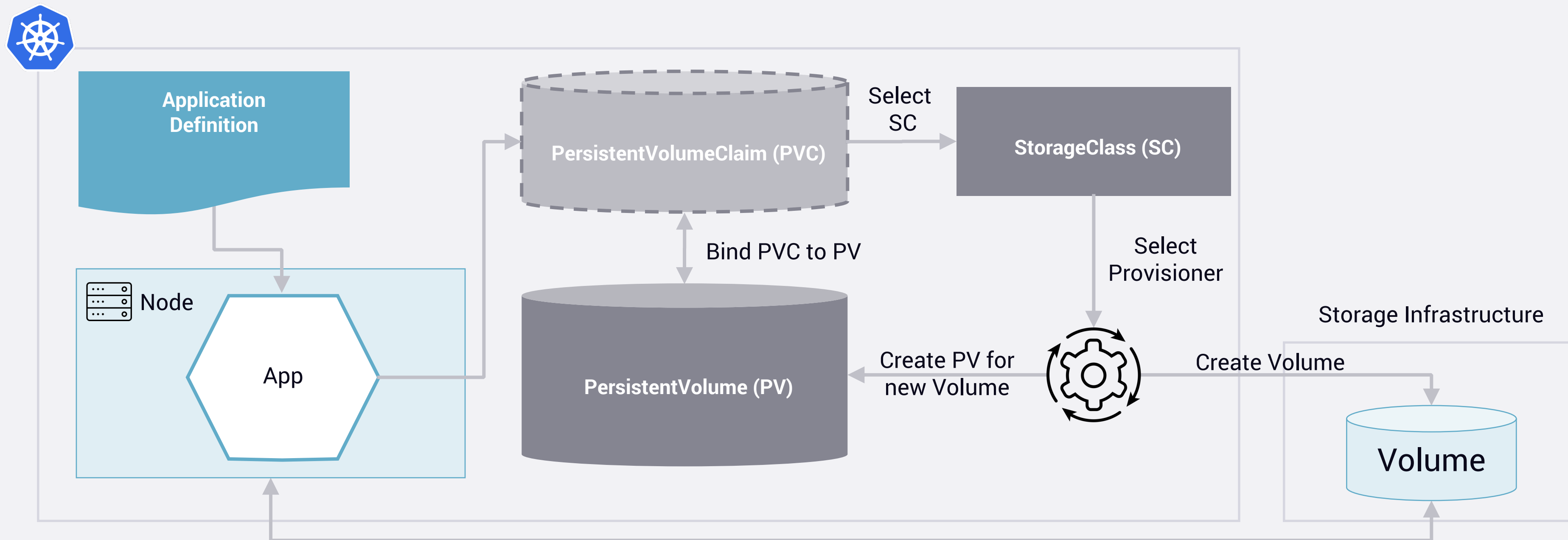
```
kind: Deployment
apiVersion: v1
metadata:
  name: my-app
spec:
  template:
    spec:
      containers:
      - name: app-container
        image: alpine:3.7
        command: ["my-app.sh"]
        args: ["--datadir", "/data/my-app"]
        volumeMounts:
        - name: data-volume
          mountPath: /data
      volumes:
      - name: data-volume
        persistentVolumeClaim:
          claimName: my-claim
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my-claim
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 8Gi
  storageClassName: ssd
```



# dynamic storage provisioning

## putting it all together



Volume mounted on node where Pod is scheduled  
(based on Pod -> PVC -> PV mapping)



# container storage interface the path forward



CONTAINER  
STORAGE  
INTERFACE

See [Managing Disk Volumes in Kubernetes](#)  
SDC 2018 talk by Saad and Nikhil for more info!

## Out of Tree

Independent Development and  
Release Cycles, Easier to Maintain

## Standard Deployment

Common deployment interface using  
native Kubernetes primitives

## File & Block

Standardized implementation  
APIs for using file and block

## Cross-Orchestrator

Vendor friendly. Kubernetes,  
Mesos, CloudFoundry,



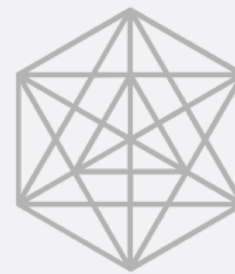
# other operational concerns

## scheduling, backup, restore, migration



### state is meaningful

- Instances are unique and are not interchangeable
- Access to persistent data is needed across restarts



### resiliency is complex

- High-availability depends on instance coordination
- Frequent restarts/pre-empts destabilize service



### data is important

- How does backup, recovery, and migrate work? See Kasten's K10 as an example!
- Resource contention concerns





# Developer and Operator Support





# StatefulSets

## support for stateful applications



### Stable Identifiers

Stable network identifiers for applications that depend on this

### Stable Persistence

Includes persistent mapping across pod restarts and reschedules

### Ordered Operations

Ordered and graceful deployment, scaling, termination

### Update Operations

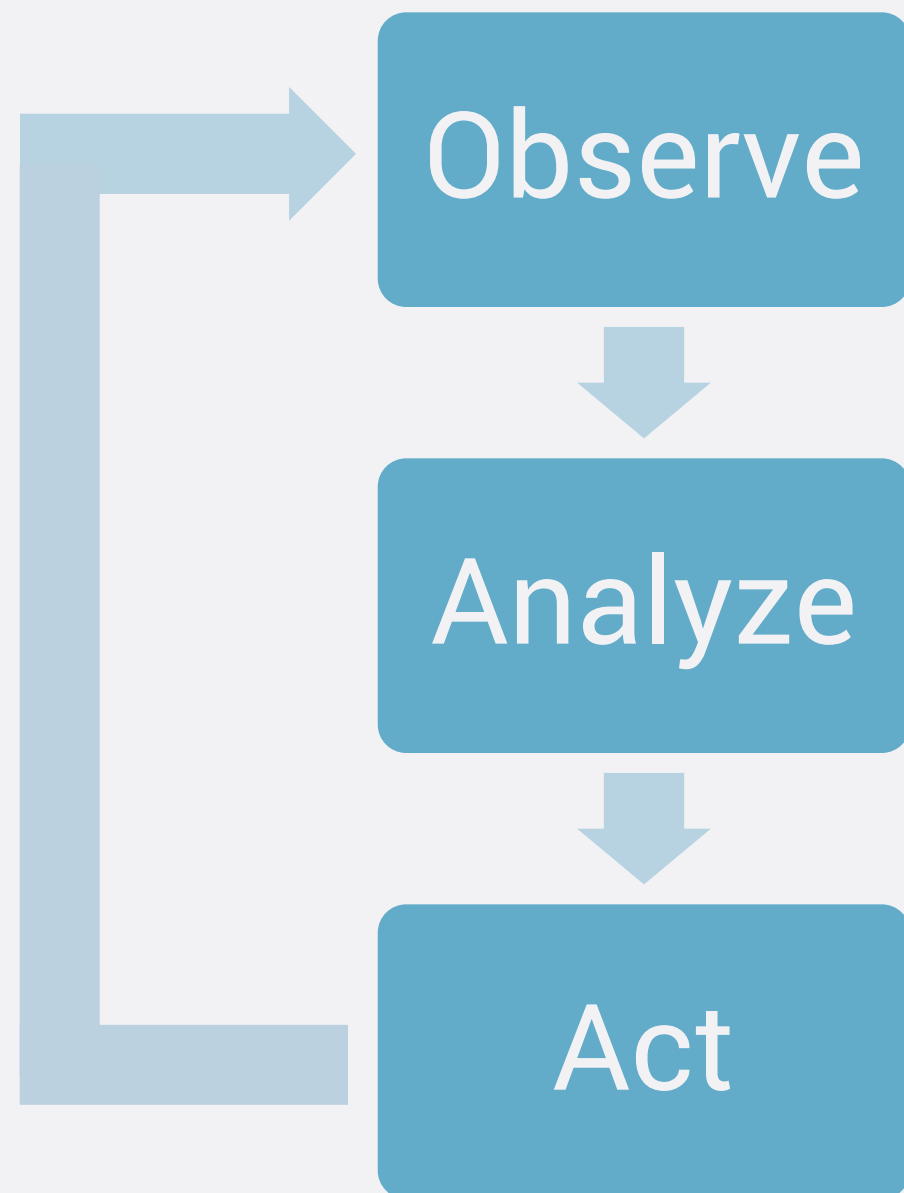
Rolling updates with restrictions





# the operator design pattern to deploy and manage apps

human ops knowledge → software



## Support Complex Ops

Backups, Recovery, Scaling,  
Upgrades

## Active Reconciliation

Reconcile desired vs. actual state

## SDK-based

Easy to get started with multiple SDKs.  
Still a few sharp edges though.

## Extensible

Developer-extensible via  
CustomResourceDefinitions







## kanister: A framework for application-level data management

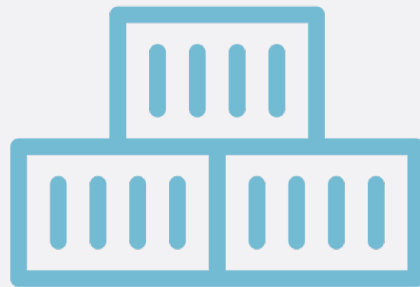
- Supports complex distributed applications
- Separates mechanism from policy/orchestration
- Allows for unified schedulers and monitoring
- Clean API allows for developer extensions



<https://github.com/kanisterio>



# operator high-level overview



Application



Action Request  
(Custom Resource)

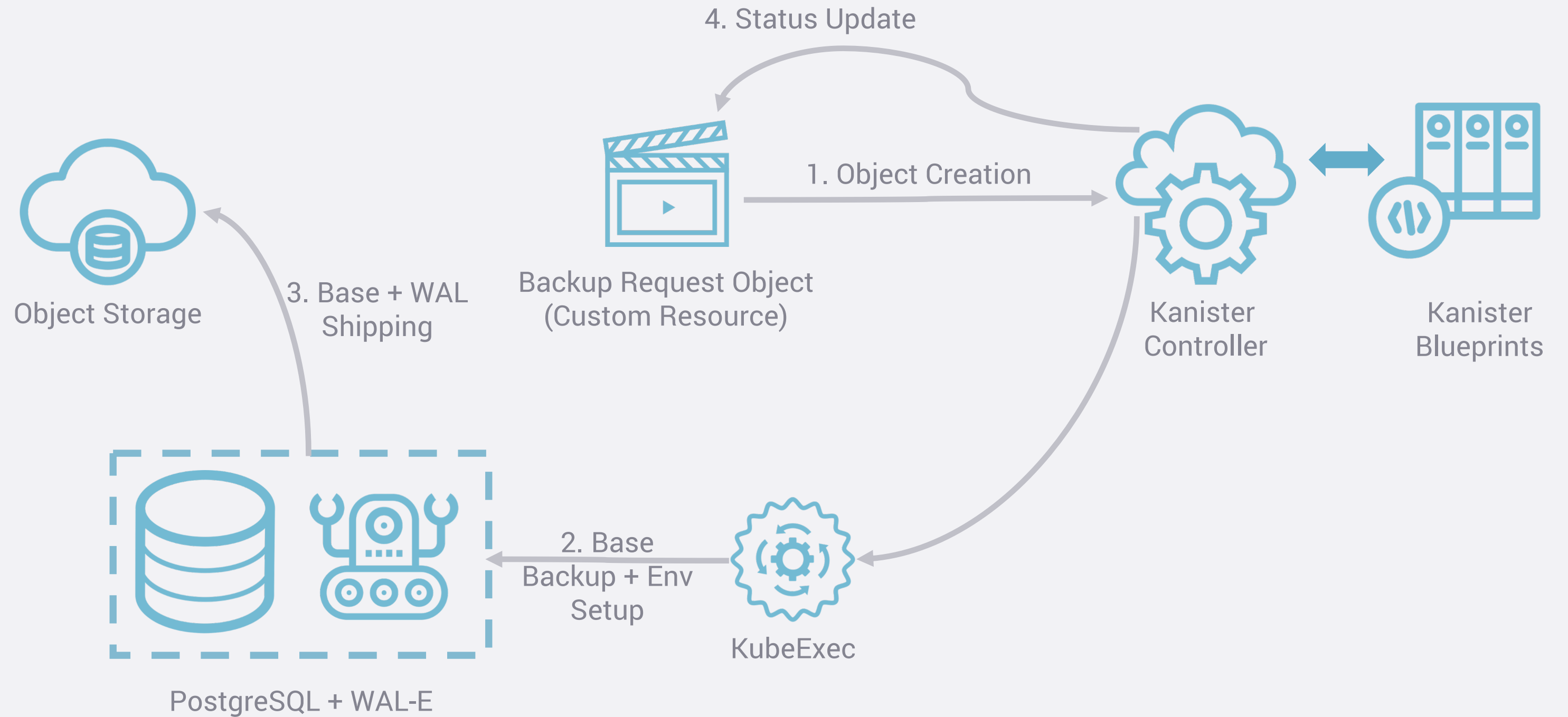


Controller



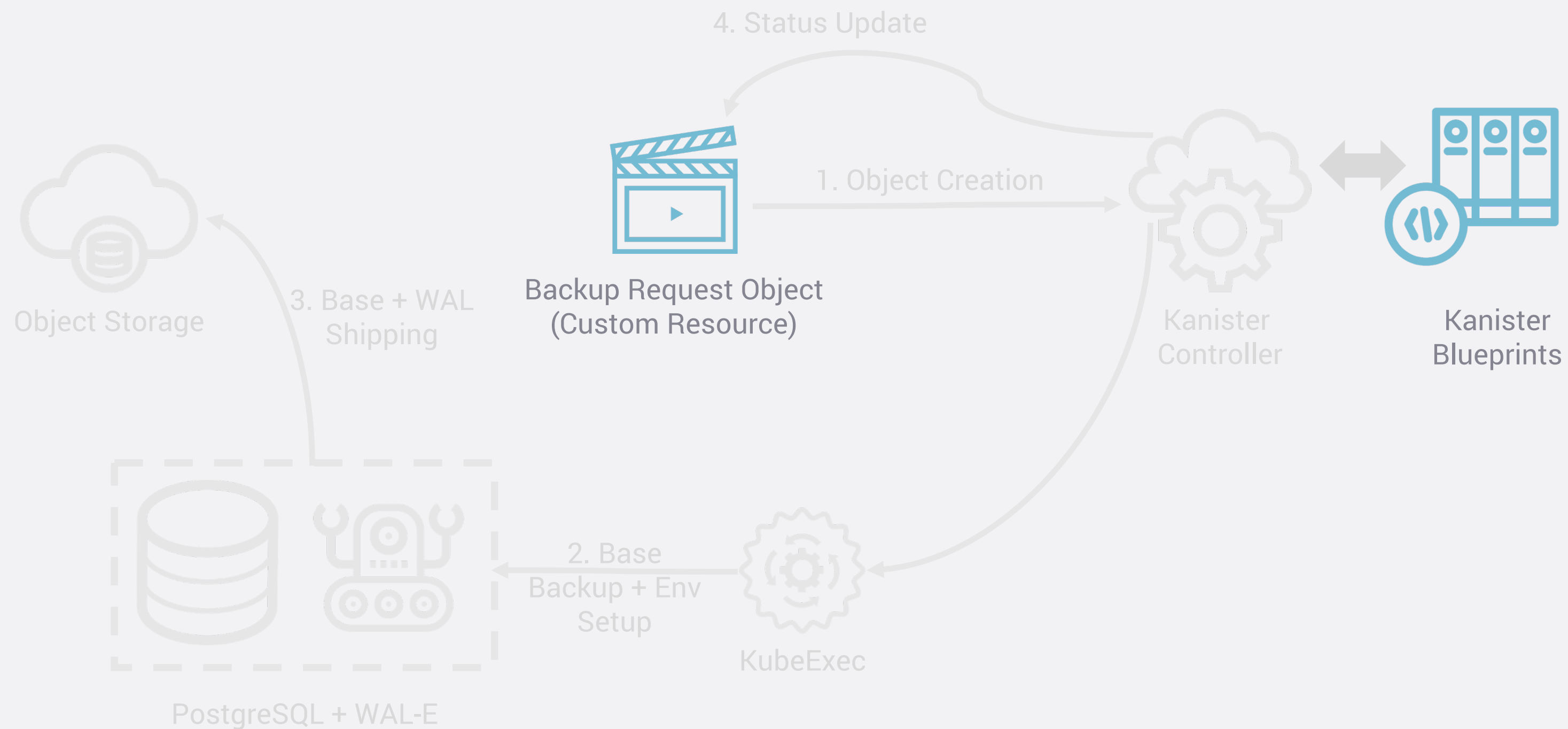
# kanister operator example

## postgresql backup



# kanister operator example

## postgresql backup





# kanister

## actionset (abridged)

```
apiVersion: cr.kanister.io/v1alpha1
kind: ActionSet
spec:
  actions:
    - name: backup
      blueprint: postgresql
      object:
        kind: StatefulSet
        name: postgresql-cluster
        namespace: default
      configMaps:
        ...
```



# kanister blueprint (abridged)

```
apiVersion: cr.kanister.io/v1alpha1
kind: Blueprint
actions:
  backup:
    type: StatefulSet
    phases:
      - func: KubeExec
        args:
          - '{{ .StatefulSet.Namespace }}'
          - '{{ index .StatefulSet.Pods 0 }}'
          - postgresql-tools-sidecar
          - bash
          - -c
          - wal-e ...
      - func: ...
  restore:
    ...
```





# other awesome stateful operators

Look at the extensive list at

<https://github.com/operator-framework/awesome-operators>



*and more...*



# packaging your applications

## helm: the kubernetes package manager

### off-the-shelf stateful “charts”

Multiple community charts available for databases, NoSQL systems, and more.



### organize settings

Easy-to-use mechanisms and a single place to codify your application’s configuration options.

```
$ helm install stable/postgresql  
--set persistence.size=40Gi  
--set persistence.storageClass=ssd
```

### supports composability

Enhance or restrict based on your goals.  
Compose stateful services within your apps.

```
<your-app>/requirements.yaml  
dependencies:  
-name: postgresql
```





# Upcoming Developments





# cloud-native databases

**cockroachdb, vitess, yugabyte, and more...**



## scalable

Auto-scaling built to respond to load and deliver predictable performance



## resilient

Fault-tolerance built in to support transparent self-healing infra



## self-managing

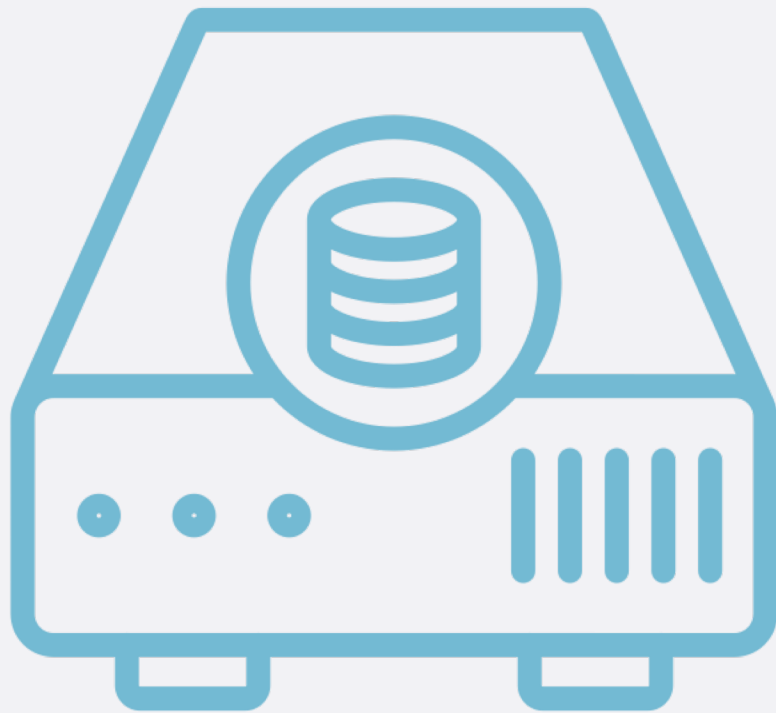
Reduces ops overhead by automatically handling system management tasks





# | local persistent volumes (beta)

## | local disks “done right”



### Leverage Local Disks

For systems (Ceph, Cassandra, etc.) that work best on local storage

### Common Primitives

Uses well-know PersistentVolume, PersistentVolumeClaim, StorageClass

### Smarter Scheduling

Smarter pod scheduling and volume binding compared to hostPath

### Expose as Block

Not just file system access anymore



# kubernetes and state wrapping up



## Stateful is Ready for Production!



### 01

#### Platform Support

Equivalent features and concepts that made stateless successful

### 02

#### Storage Vendor Choices

Large number of storage provider choices, CSI, Portability Abstractions

### 03

#### Relational / NoSQL Systems

Support from traditional relational and NoSQL systems. First-class operators. Cloud-Native DBs.

### 04

#### Increased Production Usage

50%+ users using stateful applications - SIG-APPs Survey, Apr'18



「thank you」