Knative Workshop

Jaffa Sztejnbok
Developer Advocate
IBM

jaffas@il.ibm.com

Code ⇄ **think**

IBM

Knative

Kubernetes-based platform to deploy and manage modern serverless workloads.

https://github.com/knative

https://knative.dev/

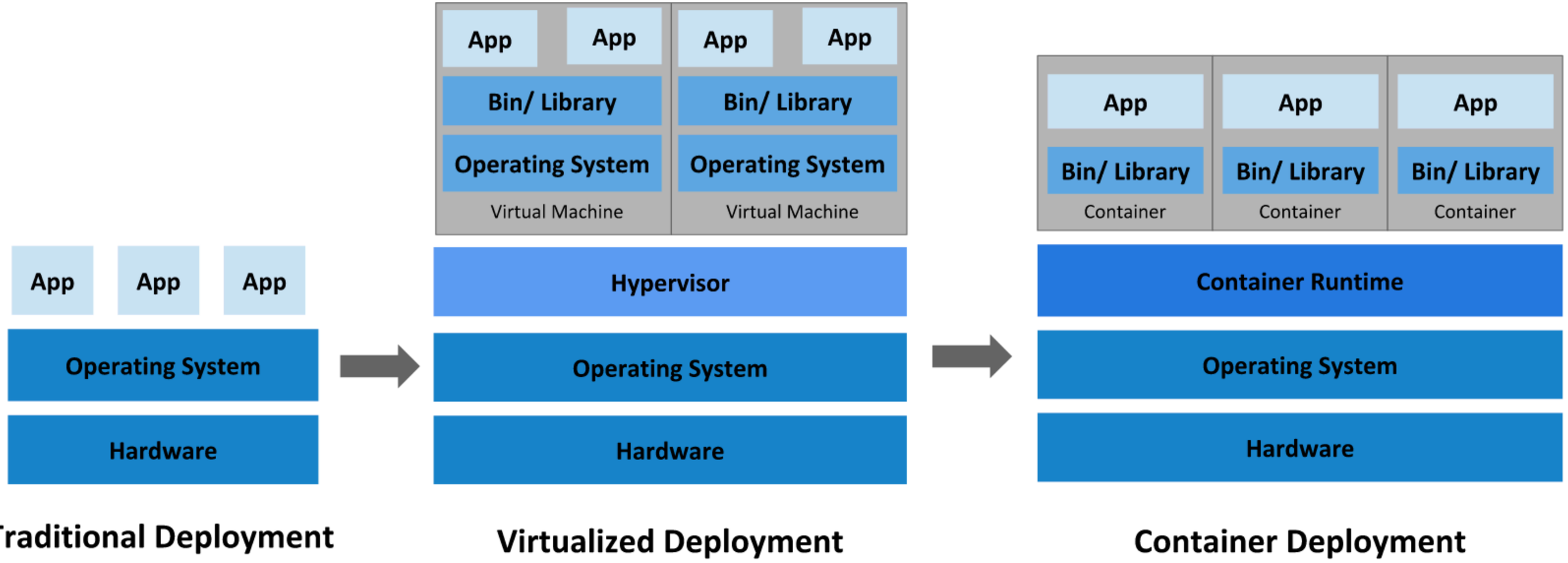
Presentation and Lab

https://jaffas.gitbook.io/knative101/

Register IBM Cloud

1. Go to https://jaffas.gitbook.io/docker101/
2. Run **Lab 1 - Accessing IBM Cloud**
3. Run **Lab 2 - Claim Your Kubernetes Cluster and install Knative on your cluster**

Quick history



Containers

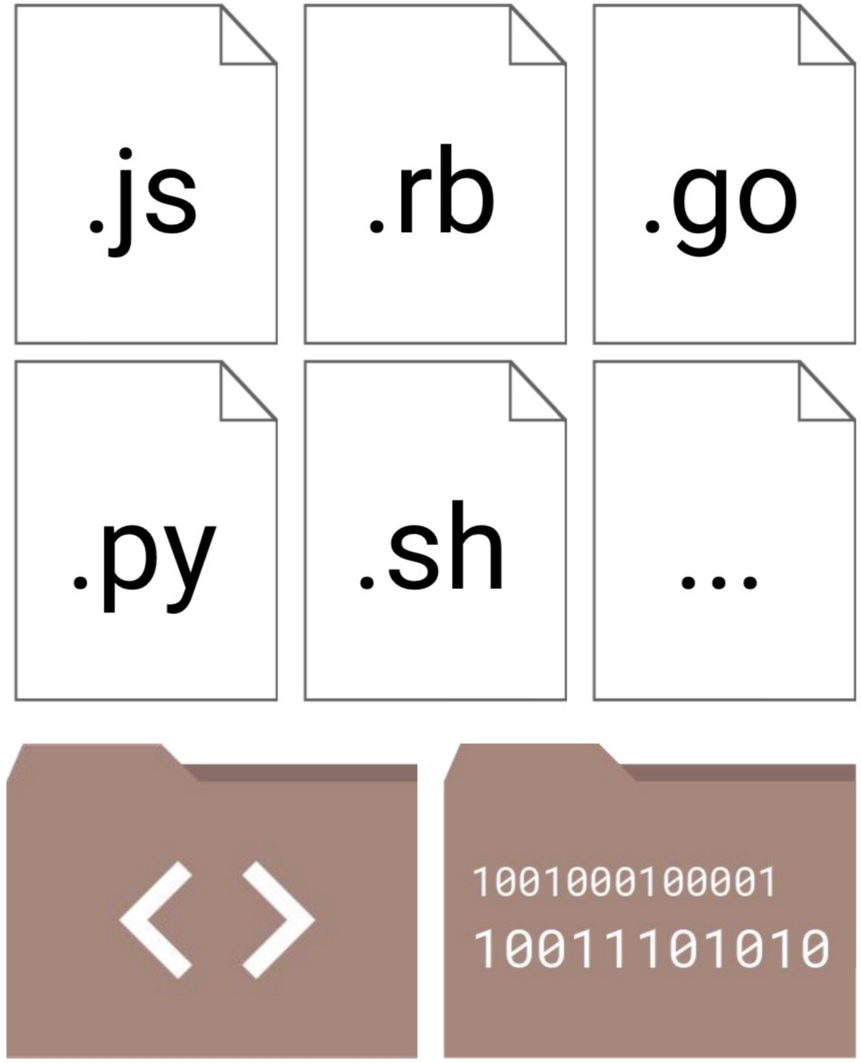
Any language

Any library

Any binary

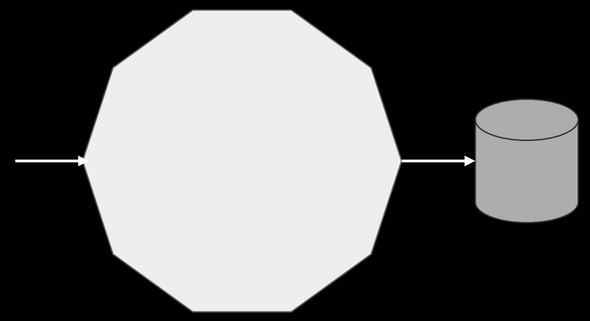
Ecosystem of base images

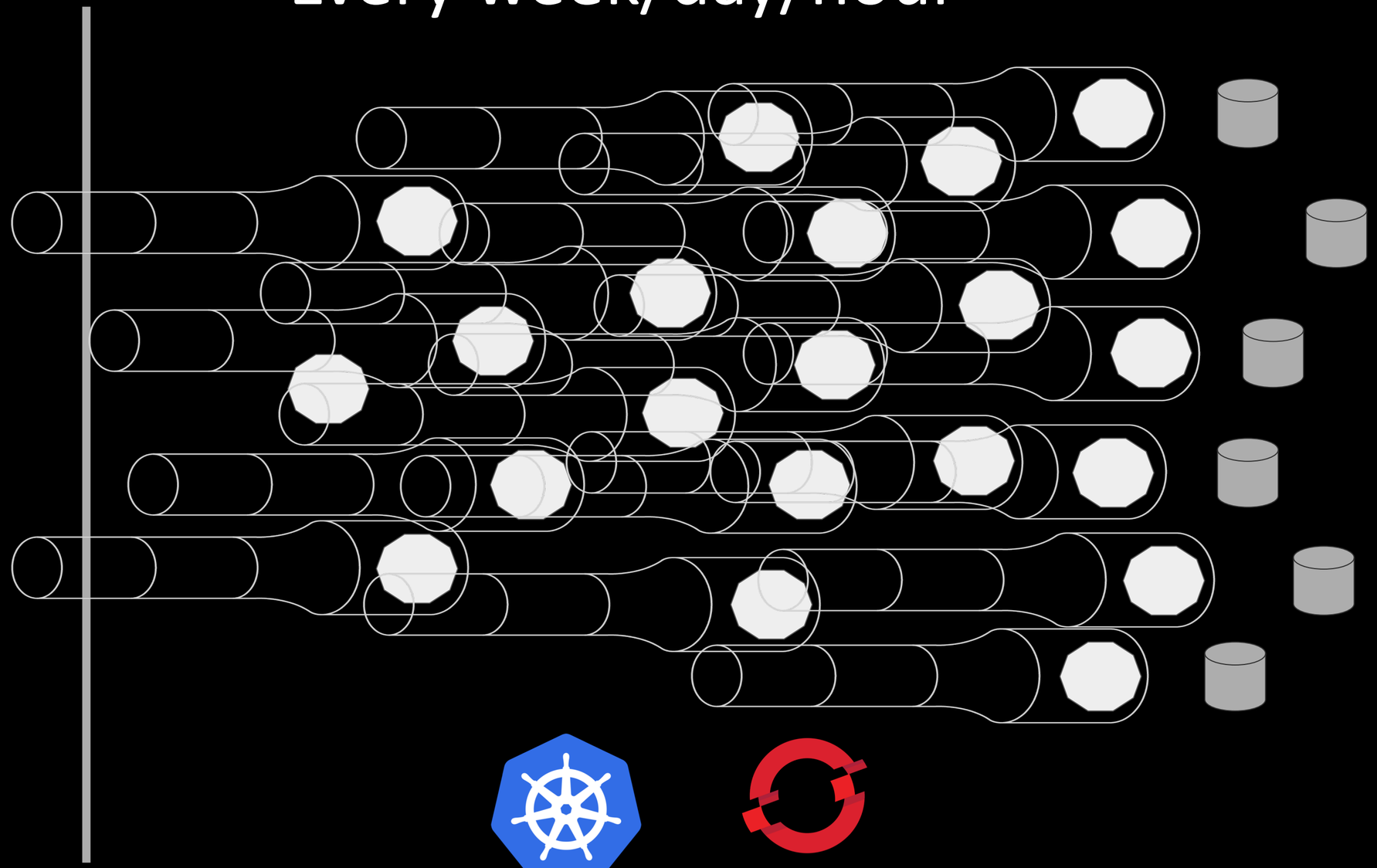
Industry standard



Every 4 months

Every week/day/hour





Love Thy Mono

OPENSHIFT

The need for something different...

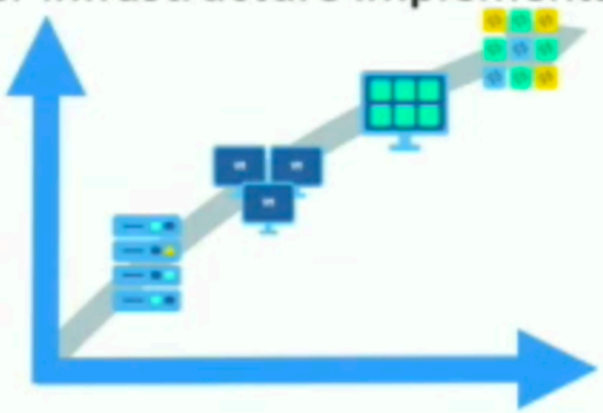
Kubernetes: Most popular Container Management project today.
Deploy and manage your containerized applications.

Resources
Containers
Pods
Replica Sets
Deployments
Services
Endpoints
Secrets
Networks
Volumes/PV/PVC
Ingress/LBs

Tooling & Other "stuff"
yaml
Spec vs Status
helm
kubectl
Istio
...

Blue/green deployments

The promise of Cloud Computing:
- Higher level abstractions
- Management of infrastructure for me
  - "Decreasing concern (and control) over infrastructure implementation"



I just want to deploy my code!

Serverless usage models

No servers

Pay only for usage

Event-driven

Portable

Cloud Native Computing Foundation

"Serverless computing refers to the concept of building and running applications that do not require server management. It describes a finer-grained deployment model where applications, bundled as one or more ***functions***, are uploaded to a platform and then executed, scaled, and billed in response to the exact demand needed at the moment."

https://www.cncf.io/blog/2018/02/14/cncf-takes-first-step-towards-serverless-computing/

Introducing Knative

- An opinionated and simplified view of application/container management
- Allowing developers to focus on coding
- Built as an extension to Kubernetes
  - Still Kubernetes under the covers
  - Access to full Kubernetes if needed
  - Integrates with the non-Knative workloads
- Building blocks on which Cloud Providers can build a platform

Knative for developers

Want to

Write code

Don't have to

Build docker image
Upload image to registry
Deploy service
Expose to the internet
Setup logging & monitoring
Scale workload...

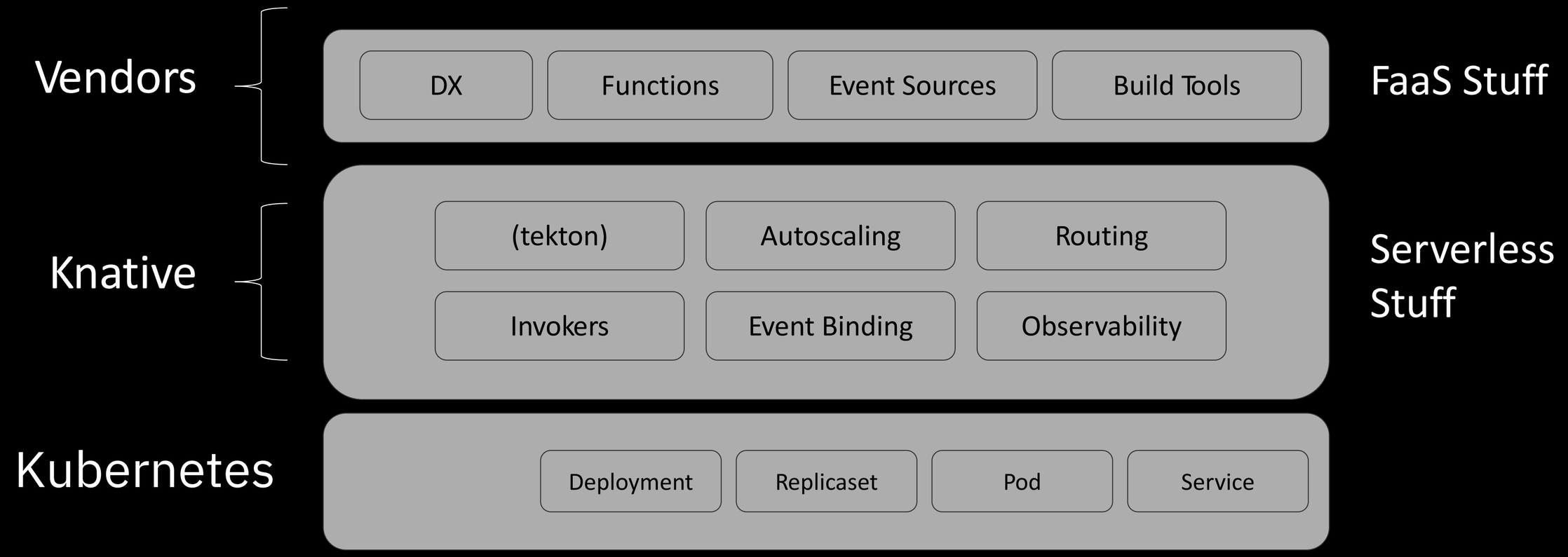
Knative

- Covers, ~~**build**~~, **serving, eventing ...**
- Common API and runtimes environments for serving workloads
- Focus on **modern** development patterns
- Implements **learnings** from over 50 companies contributing
- **Portability** of experiences, tooling, and workloads between Knative environments on any Kubernetes, Public or **On-Prem**

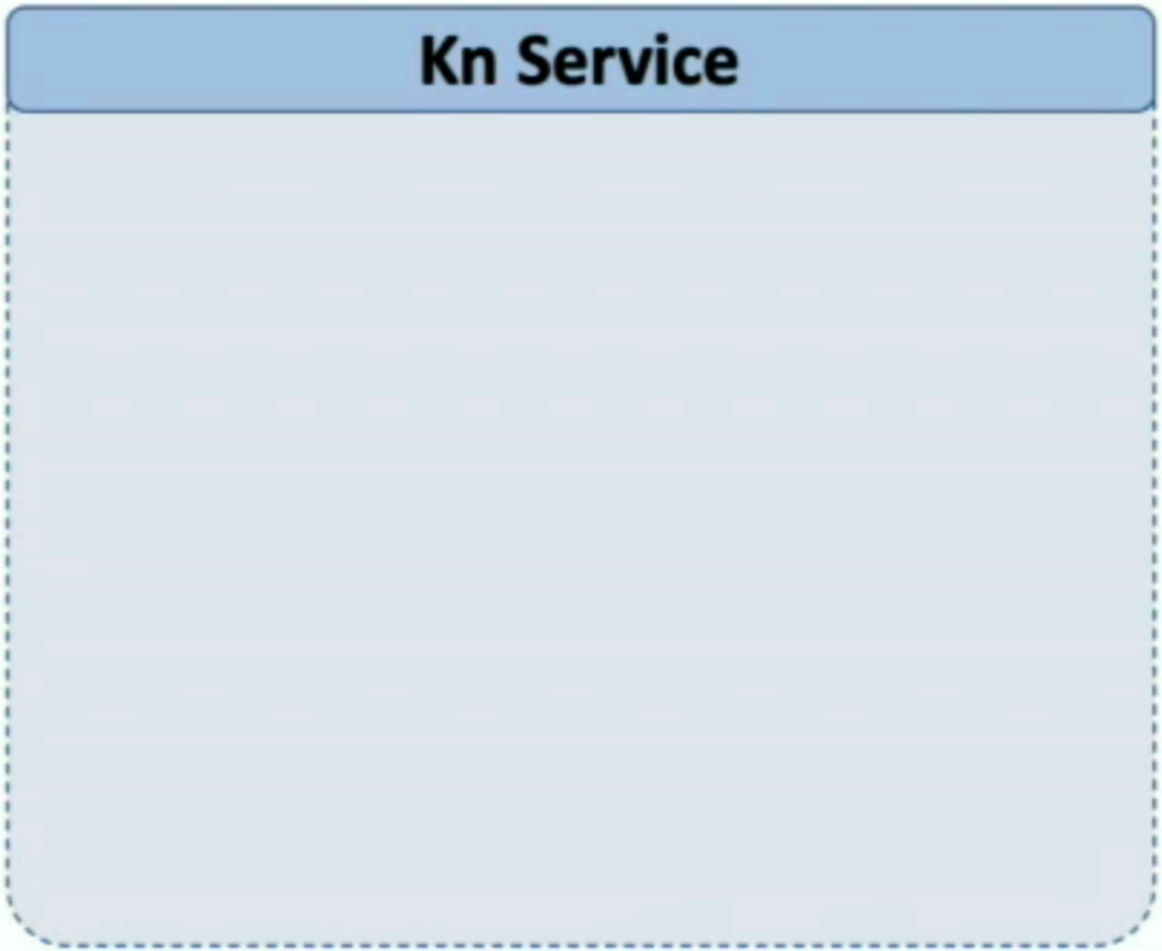
OSS: Apache License 2.0

https://github.com/knative
https://knative.dev

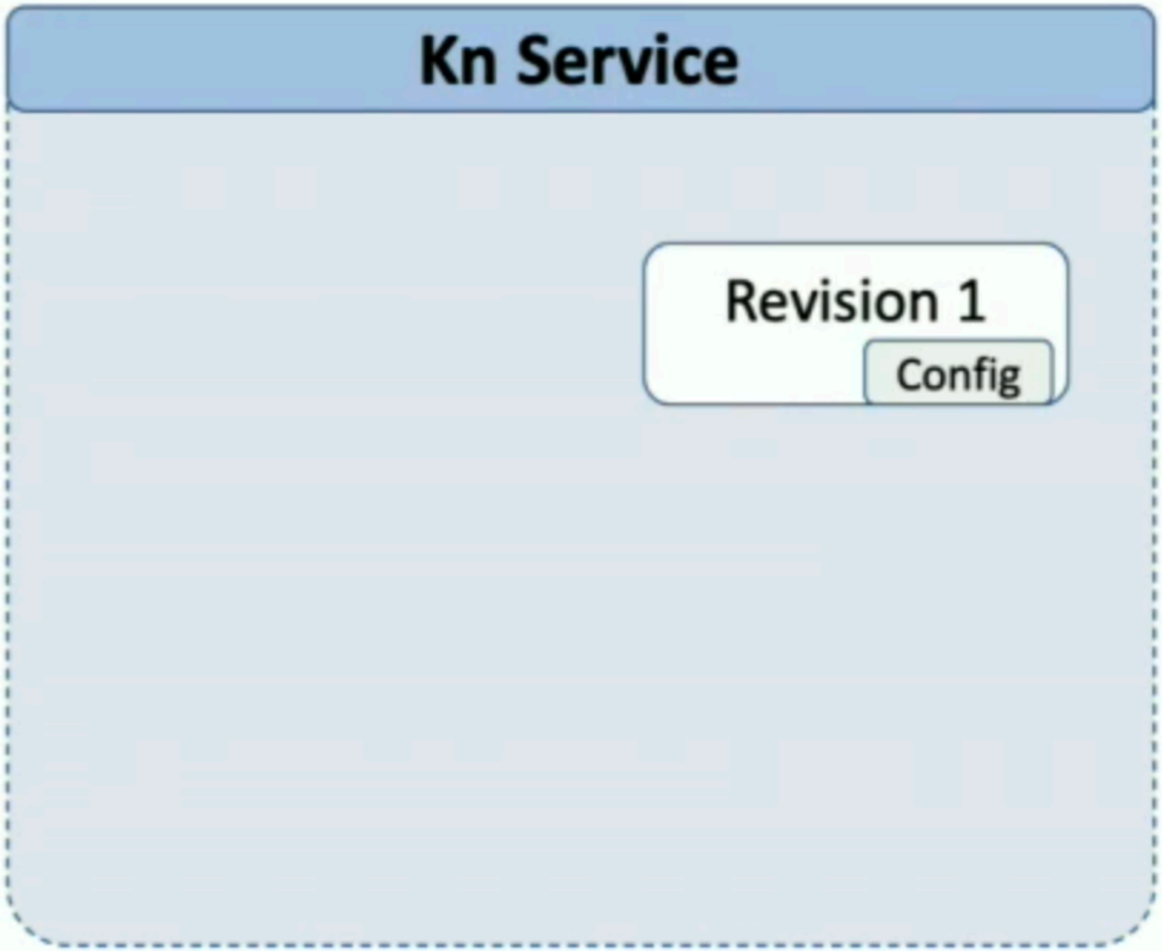
Primitives


Vendors
DX
Functions
Event Sources
Build Tools
FaaS Stuff
Knative
(tekton)
Autoscaling
Routing
Invokers
Event Binding
Observability
Serverless Stuff
Kubernetes
Deployment
Replicaset
Pod
Service

Knative Serving - The Model

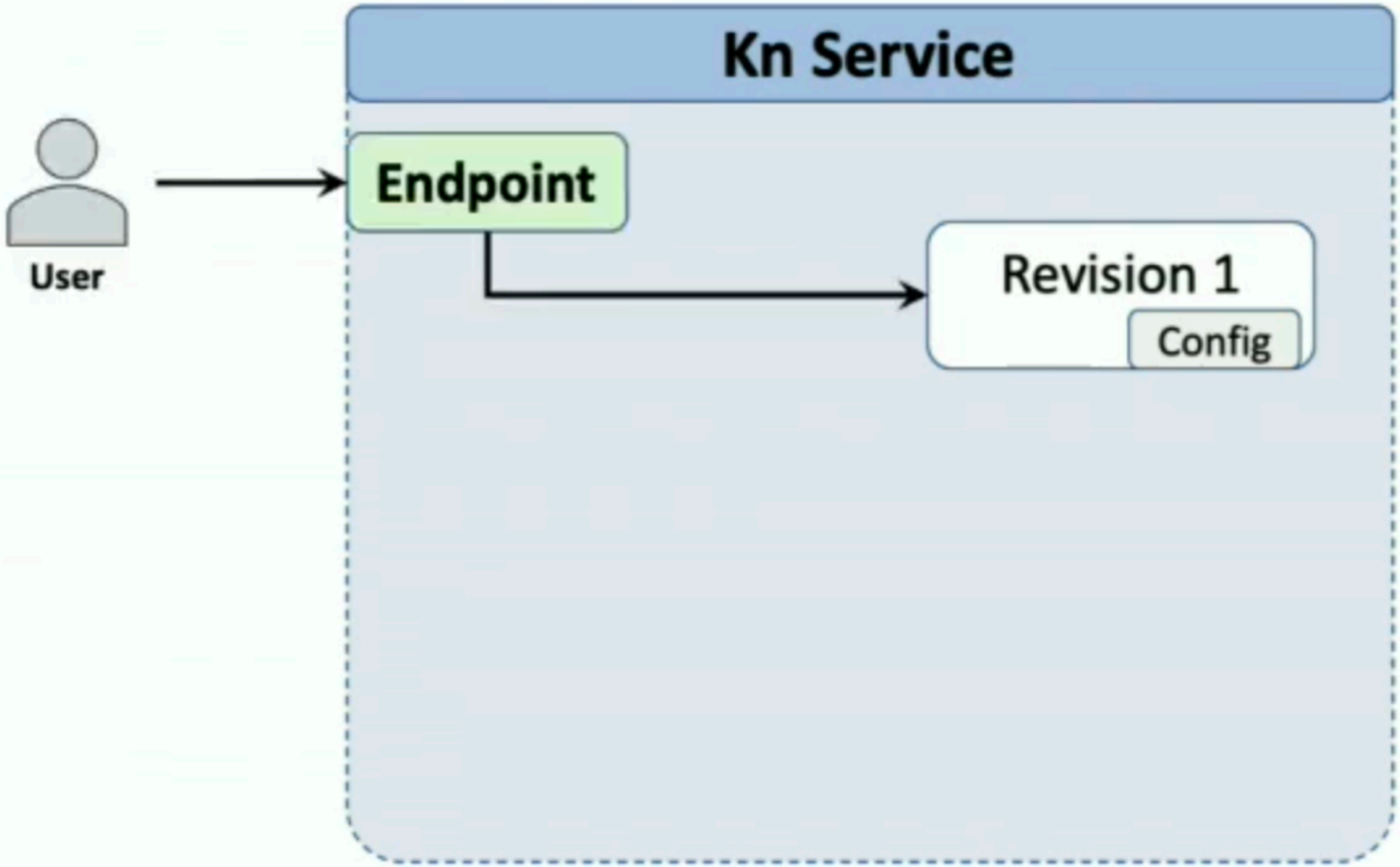


Knative Serving - The Model



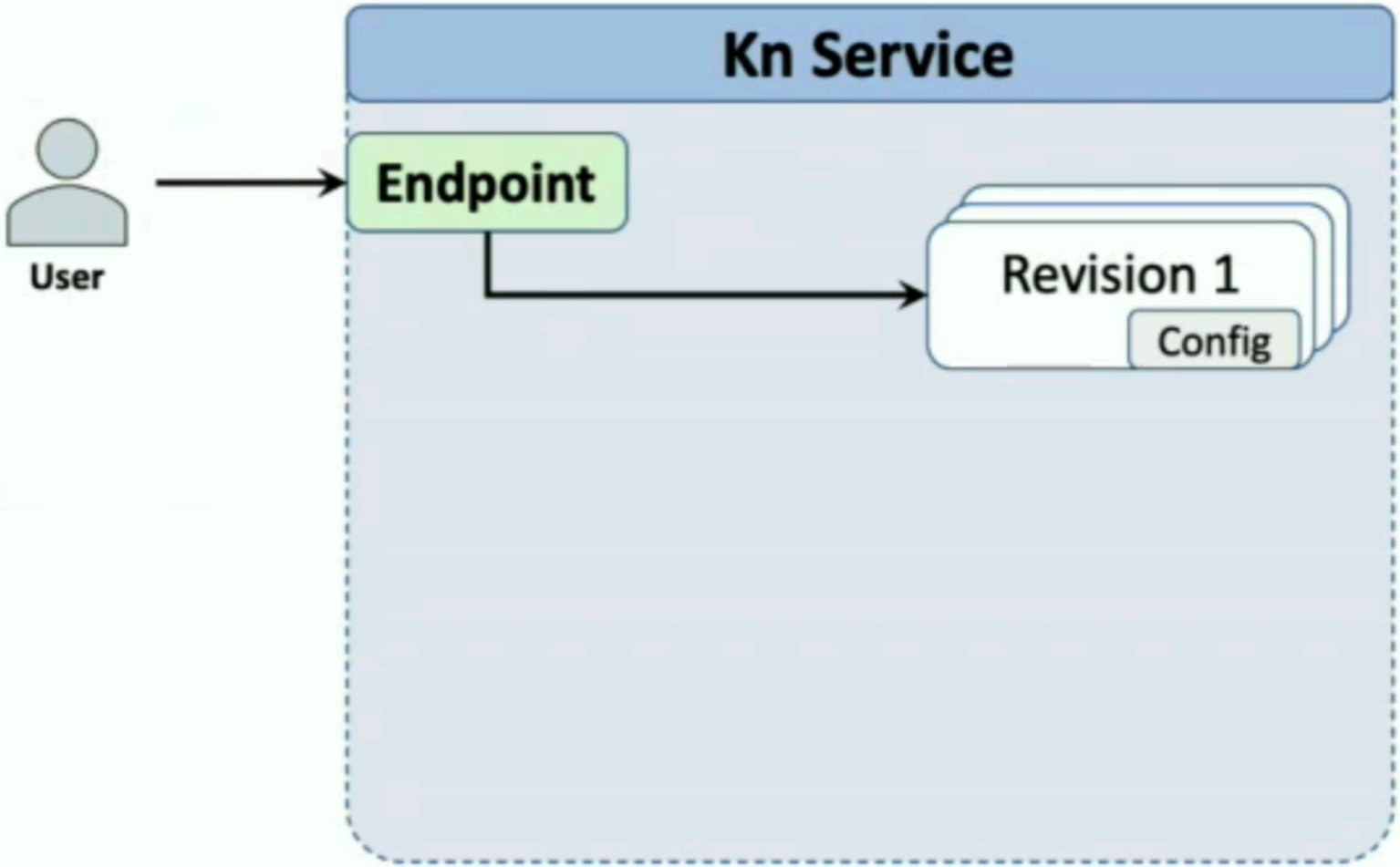
- Deploy app as pod/revision
  - Revision specific config
  - E.g. image, env vars, scale

Knative Serving - The Model



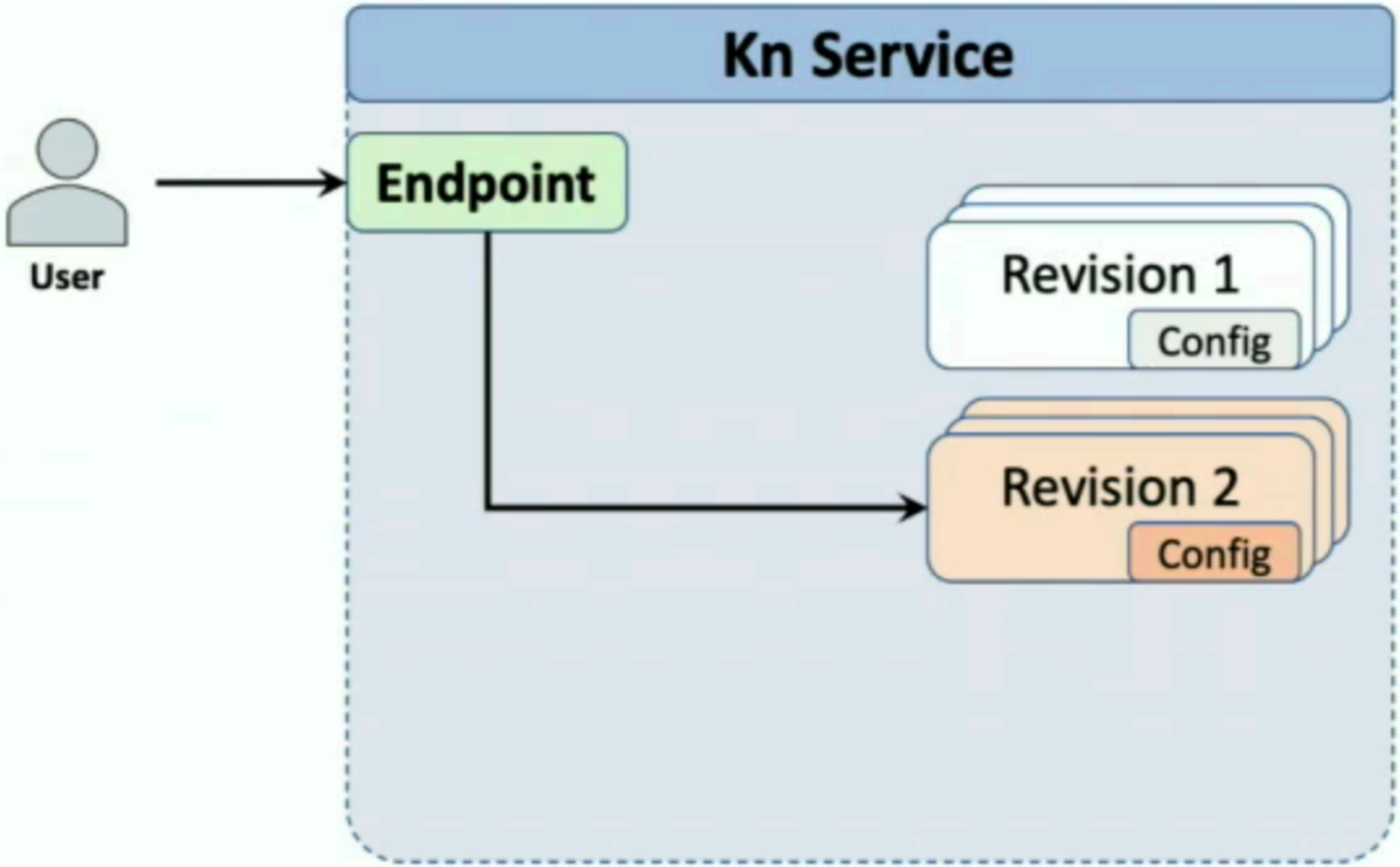
- Deploy app as pod/revision
  - Revision specific config
  - E.g. image, env vars, scale
- Networking auto-setup

Knative Serving - The Model



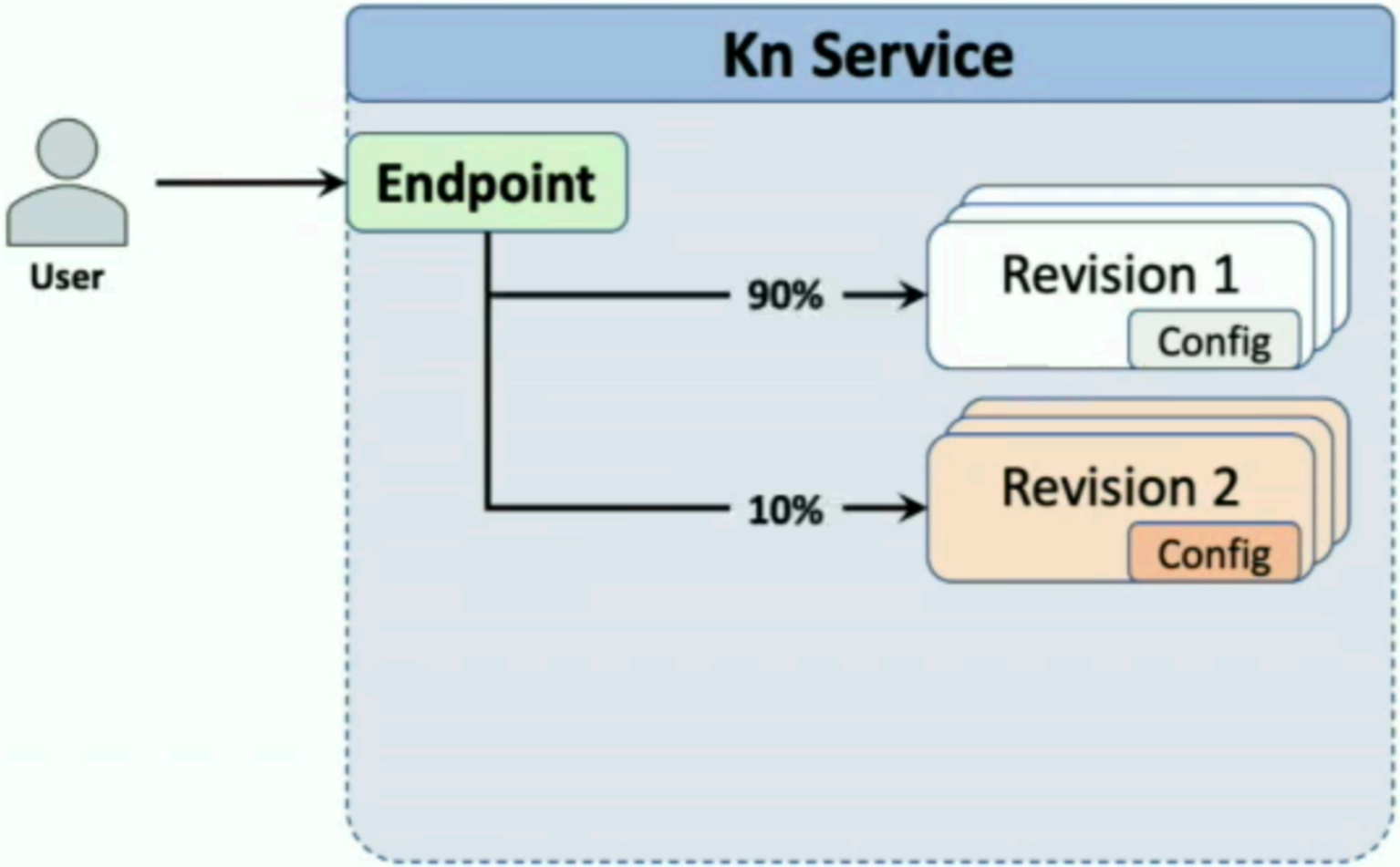
- Deploy app as pod/revision
  - Revision specific config
  - E.g. image, env vars, scale
- Networking auto-setup
- Revisions are scaled up/down
  - Based on load
  - Even down to zero

Knative Serving - The Model



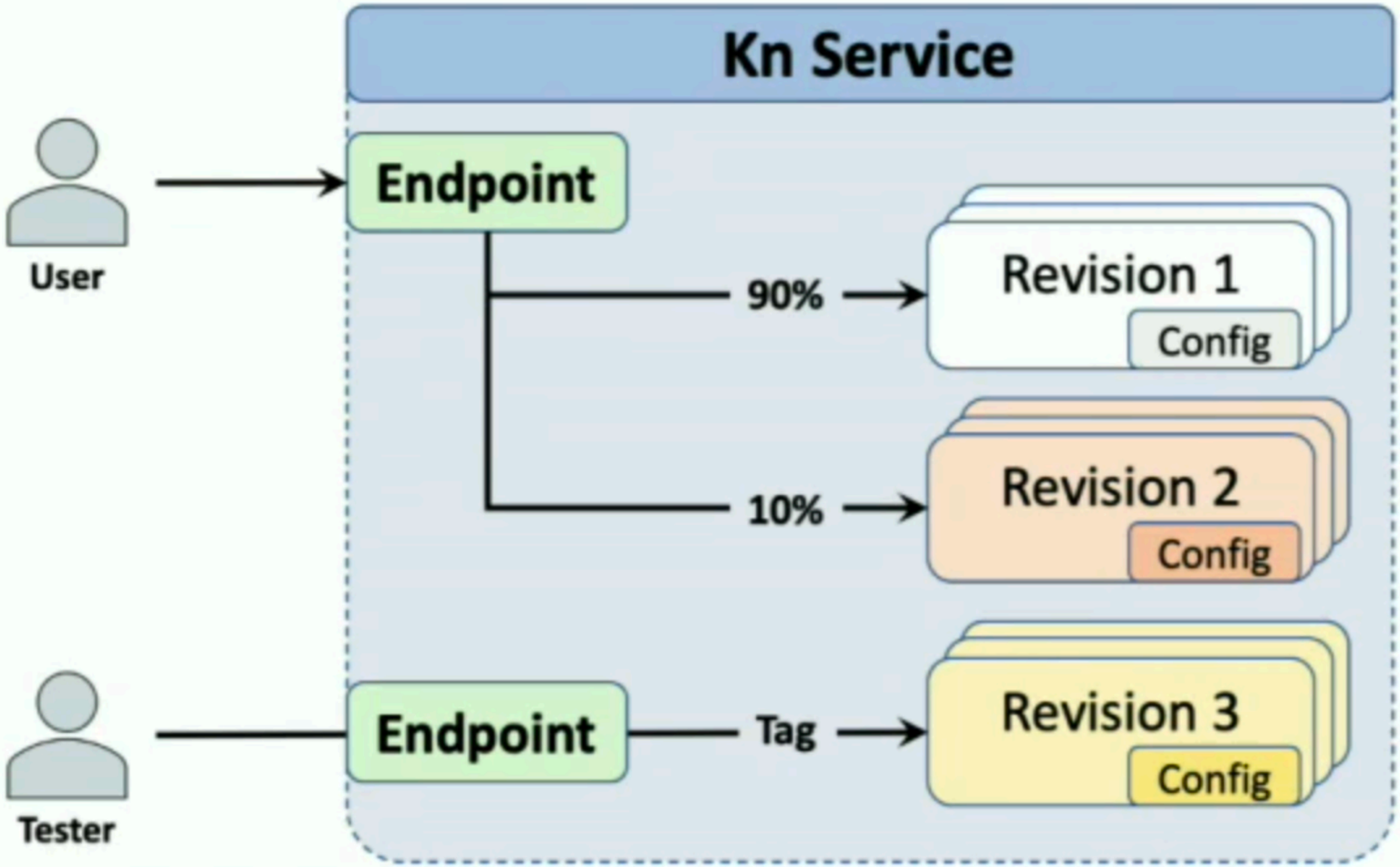
- Deploy app as pod/revision
  - Revision specific config
  - E.g. image, env vars, scale
- Networking auto-setup
- Revisions are scaled up/down
  - Based on load
  - Even down to zero
- Updates create Revisions
  - Auto-migration to new

Knative Serving - The Model



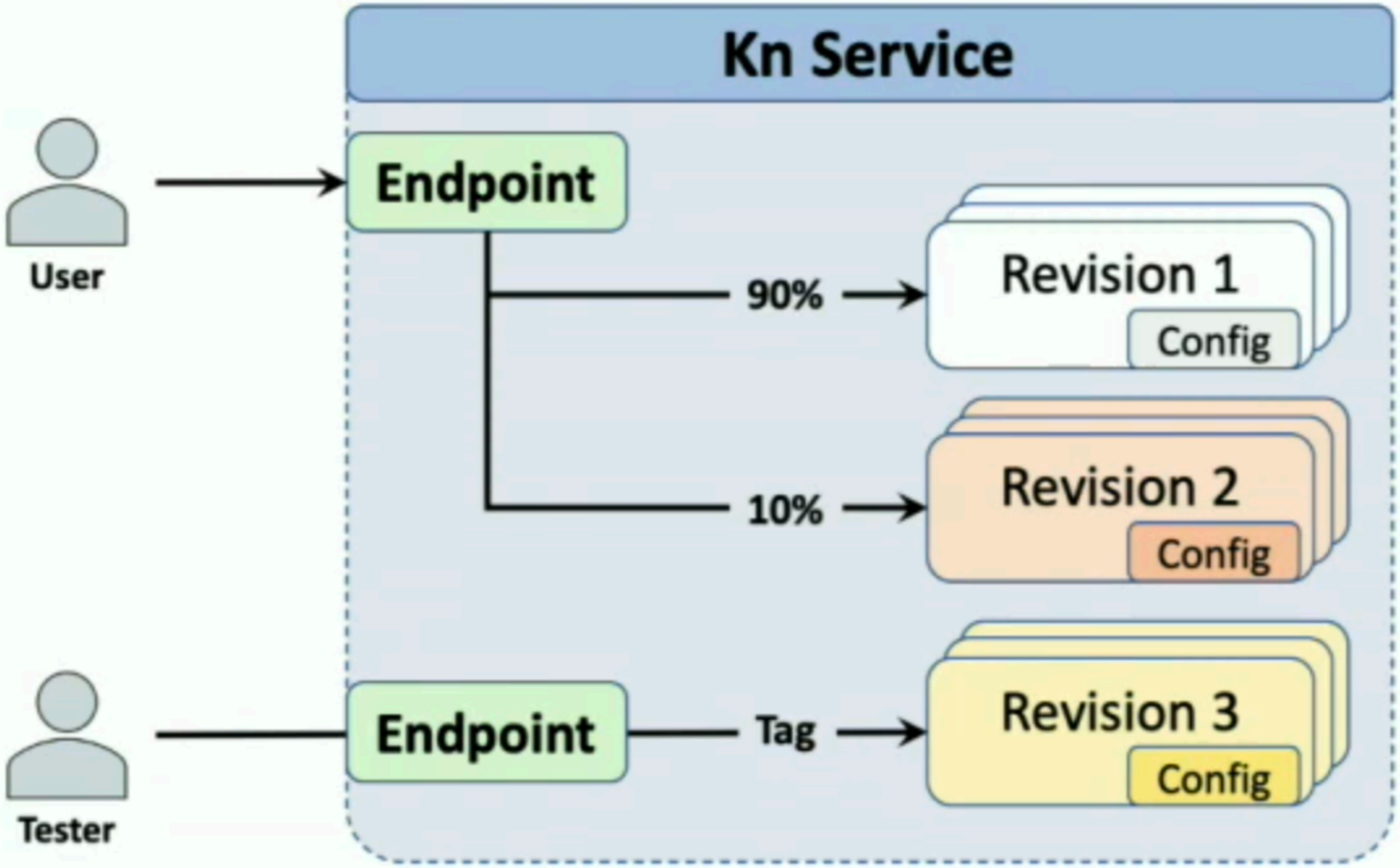
- Deploy app as pod/revision
  - Revision specific config
  - E.g. image, env vars, scale
- Networking auto-setup
- Revisions are scaled up/down
  - Based on load
  - Even down to zero
- Updates create Revisions
  - Auto-migration to new
- Traffic splitting based on %

Knative Serving - The Model



- Deploy app as pod/revision
  - Revision specific config
  - E.g. image, env vars, scale
- Networking auto-setup
- Revisions are scaled up/down
  - Based on load
  - Even down to zero
- Updates create Revisions
  - Auto-migration to new
- Traffic splitting based on %
- Dedicated URLs to Revisions

Knative Serving - The Model



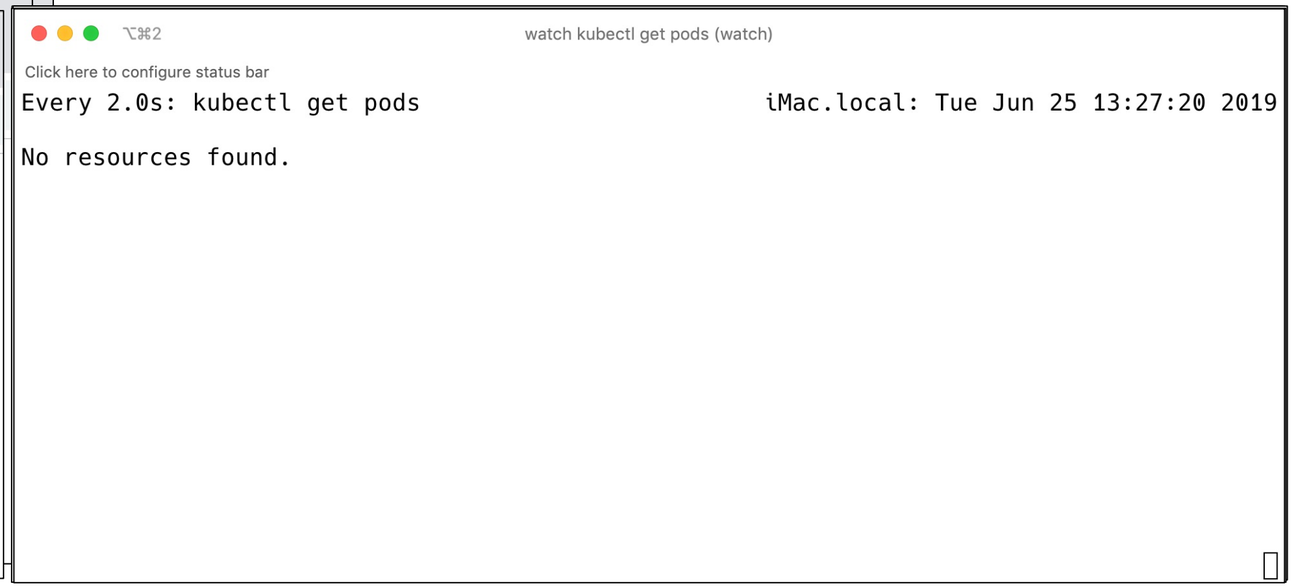
- Deploy app as pod/revision
  - Revision specific config
  - E.g. image, env vars, scale
- Networking auto-setup
- Revisions are scaled up/down
  - Based on load
  - Even down to zero
- Updates create Revisions
  - Auto-migration to new
- Traffic splitting based on %
- Dedicated URLs to Revisions

And Knative manages all of these resources for you!

Hello World

```
▸ kn service create helloworld --image csantanapr/helloworld
Service 'helloworld' successfully created in namespace 'default'.

▸ kn service list
NAME        DOMAIN                                                    GENERATION   AGE   CONDITIONS   READY   REASON
helloworld  helloworld-default.csantana-demos.us-south.containers.appdomain.cloud   1            45s   3 OK / 3     True
```



© 2019 IBM Corporation : Confidential

Automatically scales based on requests

Req

Scales out fast

Scales down to zero

Higher Utilization

Code

Container runtime contract

HTTP

~~STATE~~

app.js

```
const app = express();

app.get('/', (req, res) => {
  console.log('Hello world received a request.');

  const target = process.env.TARGET || 'World';
  res.send(`Hello ${target}!`);
});
```

main.go

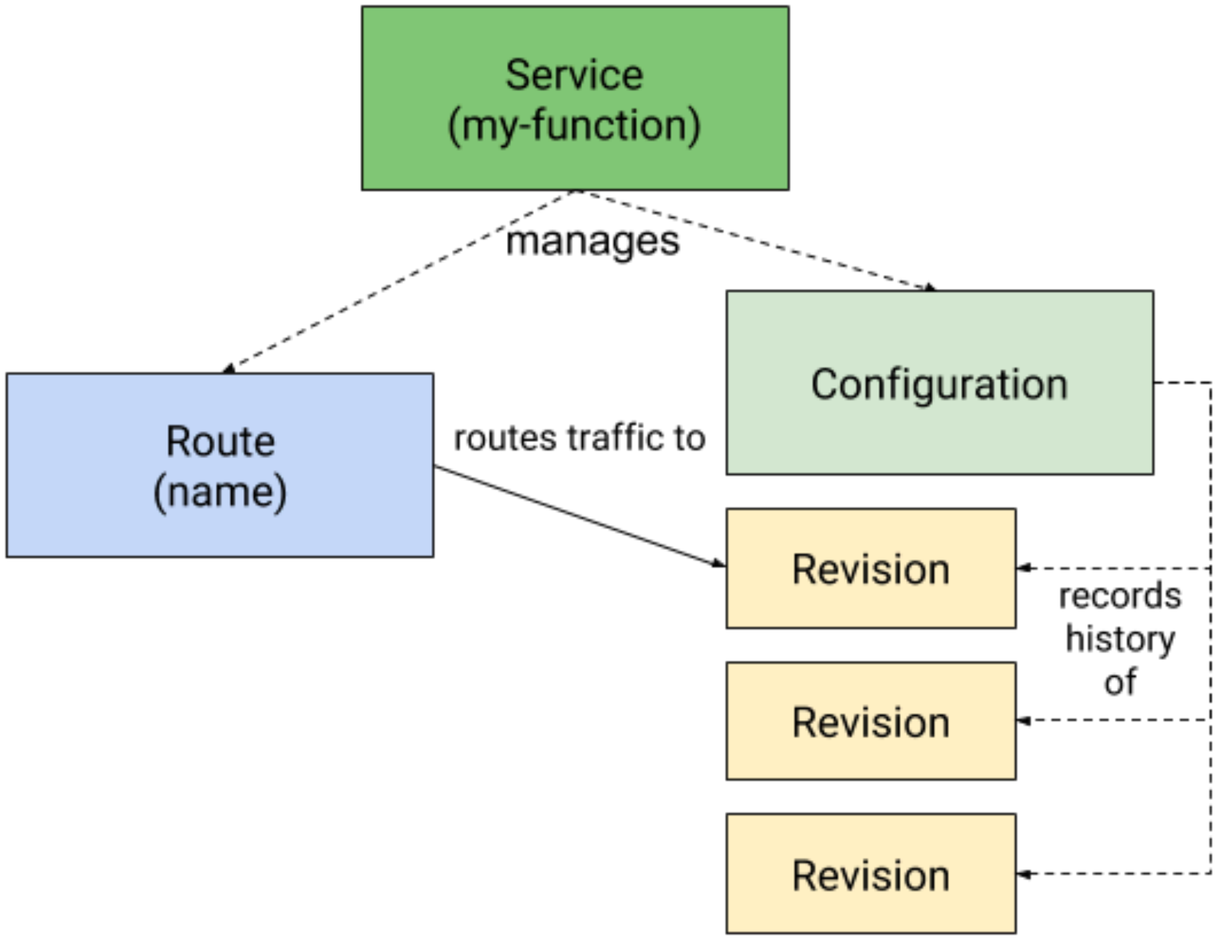
```
func handler(w http.ResponseWriter, r *http.Request) {
        log.Print("Hello world received a request.")
        target := os.Getenv("TARGET")
        if target == "" {
                target = "World"
        }
        fmt.Fprintf(w, "Hello %s!\n", target)
}
```

Interesting Capabilities

- Scale-to-zero: No pod == no memory
- Scale-from-zero: Traffic spike starts N pods
- Configurations & Revisions - built-in Blue/Green
- In-Cluster Image Building (Tekton)
- Traffic splitting
- Eventing System

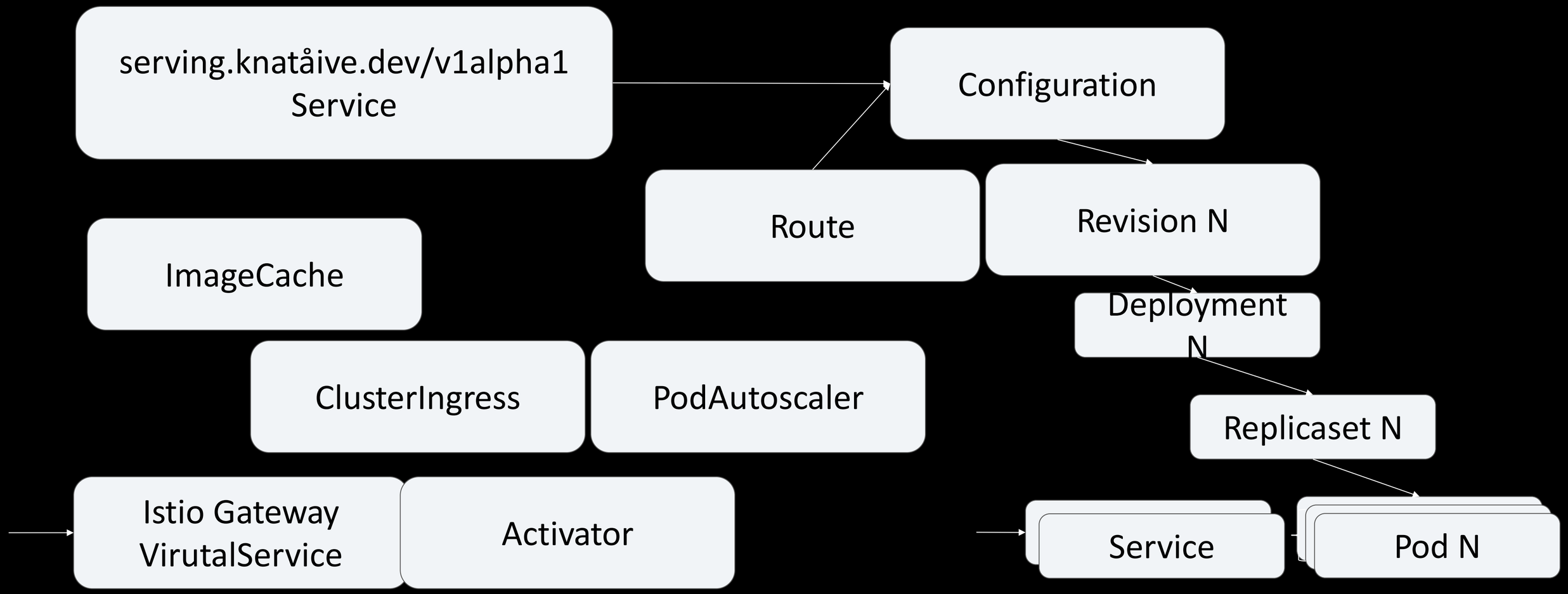
Knative Service Resource Types

- A **Route** provides a named endpoint that routes traffic to one or more revisions.
- **Revisions**, which are immutable snapshots of code + config, created by a Configuration
- **Configuration**, which acts as a stream of environment Revisions.
- **Service** is a top level abstraction for managing Routes and Configurations



© 2019 IBM Corporation : Confidential

kubectl apply -f ksvc.yaml



Knative Service yaml (minimum)

```yaml
apiVersion: serving.knative.dev/v1beta1
kind: Service
metadata:
  name: helloworld
spec:
  template:
    spec:
      containers:
      - image: csantanapr/helloworld
```

```
▸ kubectl apply -f service.yaml
service.serving.knative.dev/helloworld configured
```

https://github.com/knative/serving/blob/master/docs/runtime-contract.md

https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.10/#containerport-v1-core

* Default port is 8080, expose as environment variable PORT to the container, can override with containerPort

© 2019 IBM Corporation : Confidential

Knative custom resources (CR)

```
▸ kubectl apply -f service.yaml
service.serving.knative.dev/helloworld configured

▸ kubectl get ksvc
NAME         URL                                                                    LATESTCREATED      LATESTREADY        READY
helloworld   http://helloworld-default.csantana-demos.us-south.containers.appdomain.cloud   helloworld-xg59c   helloworld-xg59c   True

▸ kubectl get configuration
NAME         LATESTCREATED      LATESTREADY        READY   REASON
helloworld   helloworld-xg59c   helloworld-xg59c   True

▸ kubectl get revision
NAME               SERVICE NAME       GENERATION   READY   REASON
helloworld-bs225   helloworld-bs225   1            True
helloworld-v76f8   helloworld-v76f8   2            True
helloworld-xg59c   helloworld-xg59c   3            True

▸ kubectl get kpa
NAME               READY   REASON
helloworld-bs225   False   NoTraffic
helloworld-v76f8   False   NoTraffic
helloworld-xg59c   True

▸ kubectl get route
NAME         URL                                                                    READY   REASON
helloworld   http://helloworld-default.csantana-demos.us-south.containers.appdomain.cloud   True

▸ kubectl get virtualservice -n knative-serving
NAME                                          GATEWAYS                        HOSTS
route-43c53ba4-976d-11e9-870d-feddda1c832c   [knative-ingress-gateway mesh]   [helloworld-default.csantana-demos.us-south.containers.app
```

© 2019 IBM Corporation : Confidential

Kubernetes resources

```
▸ kubectl get deployments
NAME                          READY   UP-TO-DATE   AVAILABLE   AGE
helloworld-bs225-deployment   0/0     0            0           132m
helloworld-v76f8-deployment   0/0     0            0           129m
helloworld-xg59c-deployment   1/1     1            1           38m

▸ kubectl get replicaset
NAME                                    DESIRED   CURRENT   READY   AGE
helloworld-bs225-deployment-55cf49588d   0         0         0       132m
helloworld-v76f8-deployment-846bd476d5   0         0         0       130m
helloworld-xg59c-deployment-7ffcb44896   1         1         1       38m

▸ kubectl get pods
NAME                                            READY   STATUS    RESTARTS   AGE
helloworld-xg59c-deployment-7ffcb44896-tcbgk   2/2     Running   0          3m24s

▸ kubectl get service
NAME                     TYPE           CLUSTER-IP       EXTERNAL-IP                                                PORT(S)    AGE
helloworld               ExternalName   <none>           istio-ingressgateway.istio-system.svc.cluster.local   <none>     133m
helloworld-bs225         ClusterIP      172.21.235.60    <none>                                                     80/TCP     133m
helloworld-bs225-metrics ClusterIP      172.21.148.62    <none>                                                     9090/TCP   133m
helloworld-bs225-priv    ClusterIP      172.21.94.11     <none>                                                     80/TCP     133m
helloworld-v76f8         ClusterIP      172.21.61.212    <none>                                                     80/TCP     130m
helloworld-v76f8-metrics ClusterIP      172.21.204.145   <none>                                                     9090/TCP   130m
helloworld-v76f8-priv    ClusterIP      172.21.228.11    <none>                                                     80/TCP     130m
helloworld-xg59c         ClusterIP      172.21.97.209    <none>                                                     80/TCP     39m
helloworld-xg59c-metrics ClusterIP      172.21.71.241    <none>                                                     9090/TCP   39m
helloworld-xg59c-priv    ClusterIP      172.21.228.157   <none>                                                     80/TCP     39m
```

© 2019 IBM Corporation : Confidential

Knative Autoscaling

KPA (Knative Pod Autoscaler)

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: helloworld
spec:
  template:
    metadata:
      annotations:
        autoscaling.knative.dev/target: "50"
        autoscaling.knative.dev/minScale: "2"
        autoscaling.knative.dev/maxScale: "100"
    spec:
      containers:
      - image: csantanapr/helloworld
```

HPA (Horizontal Pod Autoscaler)

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: helloworld
spec:
  template:
    metadata:
      annotations:
        autoscaling.knative.dev/class: hpa.autoscaling.knative.dev
        autoscaling.knative.dev/metric: cpu
        autoscaling.knative.dev/target: "85"
    spec:
      containers:
      - image: csantanapr/helloworld
        resources:
          limits:
            cpu: "1"
          requests:
            cpu: 100m
```

watch kubectl get pods (watch)

Click here to configure status bar

```
Every 2.0s: kubectl get pods                    iMac.local: Tue Jun 25 15:57:39 2019

NAME                                          READY   STATUS    RESTARTS   AGE
helloworld-57mxc-deployment-7c98f94b85-9crwh  2/2     Running   0          7m11s
helloworld-57mxc-deployment-7c98f94b85-p9mr5  2/2     Running   0          7m3s
```

```
kubectl get hpa
NAME              REFERENCE                                TARGETS        MINPODS   MAXPODS      REPLICAS   AGE
helloworld-hxlhc  Deployment/helloworld-hxlhc-deployment   167%/85%       1         2147483647   28         4m30s
```

https://github.com/knative/docs/blob/master/docs/serving/configuring-the-autoscaler.md

https://github.com/knative/docs/blob/master/docs/serving/samples/autoscale-go/README.md

© 2019 IBM Corporation : Confidential

Knative Traffic Splitting

Configurations

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Configuration
metadata:
  name: hello-blue
spec:
  revisionTemplate:
    metadata:
      labels:
        knative.dev/type: container
    spec:
      container:
        image: gcr.io/knative-samples/knative-route-demo:blue
        imagePullPolicy: Always
        env:
          - name: T_VERSION
            value: "blue"
---
apiVersion: serving.knative.dev/v1alpha1
kind: Configuration
metadata:
  name: hello-green
spec:
  revisionTemplate:
    metadata:
      labels:
        knative.dev/type: container
    spec:
      container:
        image: gcr.io/knative-samples/knative-route-demo:green
        env:
          - name: T_VERSION
            value: "green"
```

Route (blue 90%, green 10%)

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Route
metadata:
  name: helloworld
spec:
  traffic:
    - configurationName: hello-blue
      name: v1
      percent: 90
    - configurationName: hello-green
      name: v2
      percent: 10
```

* Route.traffic can split using revisionName instead of configurationName

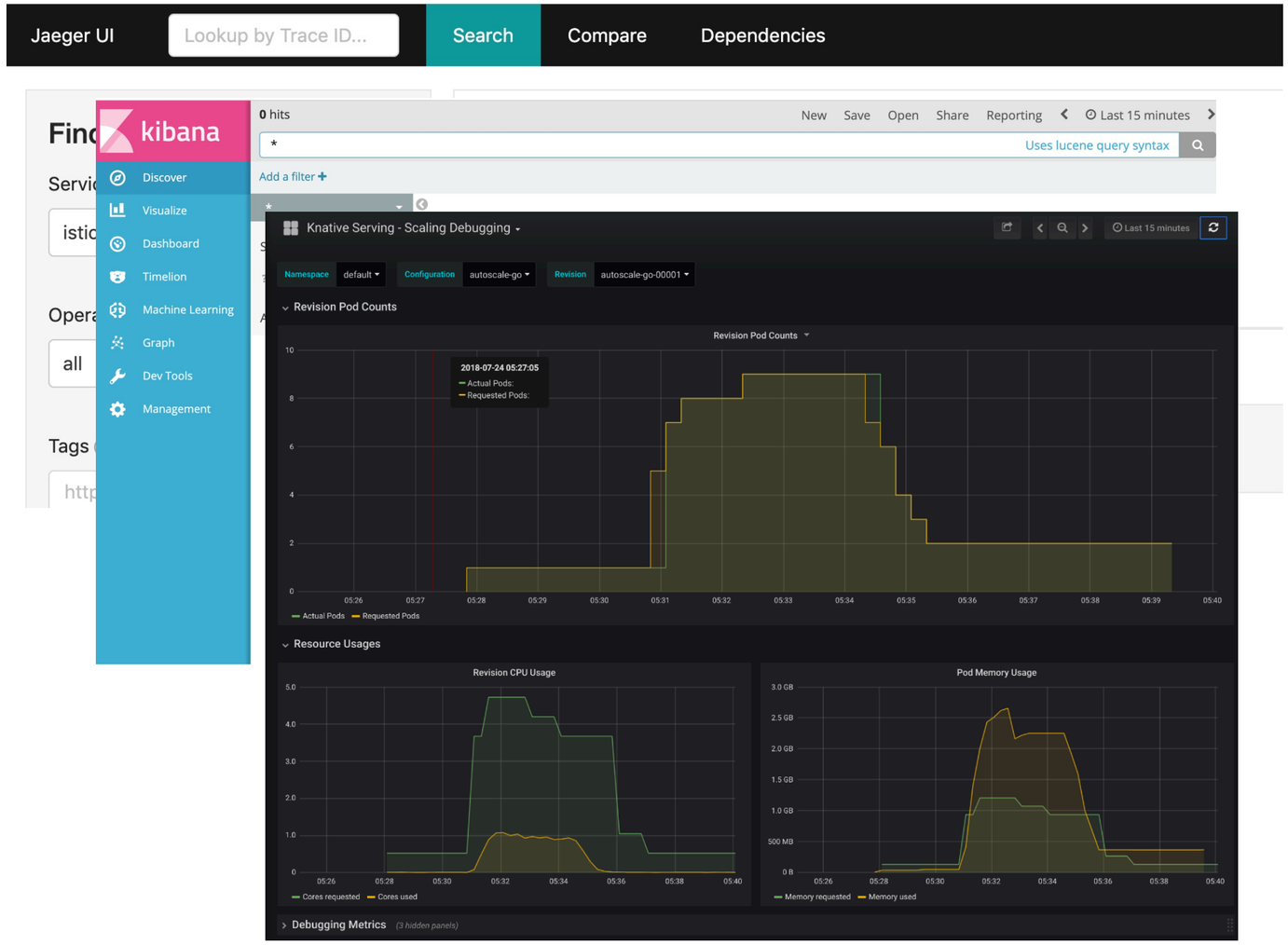
```
kubectl get route -o yaml | grep url
    url: http://helloworld.default.svc.cluster.local
    url: http://helloworld-v1-default.csantana-demos.us-south.containers.appdomain.cloud
    url: http://helloworld-v2-default.csantana-demos.us-south.containers.appdomain.cloud
  url: http://helloworld-default.csantana-demos.us-south.containers.appdomain.cloud
```

© 2019 IBM Corporation : Confidential

https://github.com/knative/docs/blob/master/docs/serving/samples/traffic-splitting/README.md

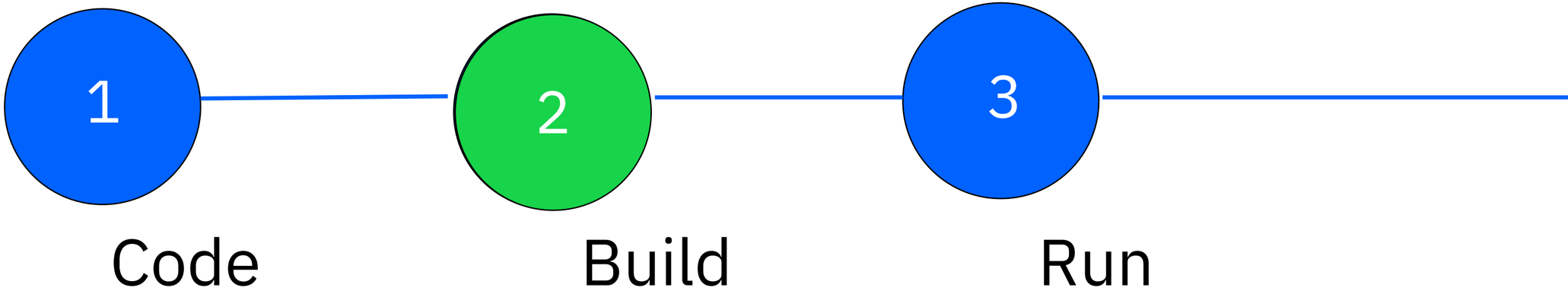
Metrics, Logs, Tracing

- Knative control plane instrumented
  - Namespace knative-monitoring
- Metrics
  - Prometheus, Grafana
- Logs
  - ELK and plugin options via fluentd
- Tracing
  - Zipkin
  - Jaeger collector
- Istio observability



© 2019 IBM Corporation : Confidential

Build



Knative Build

Moving to Tekton

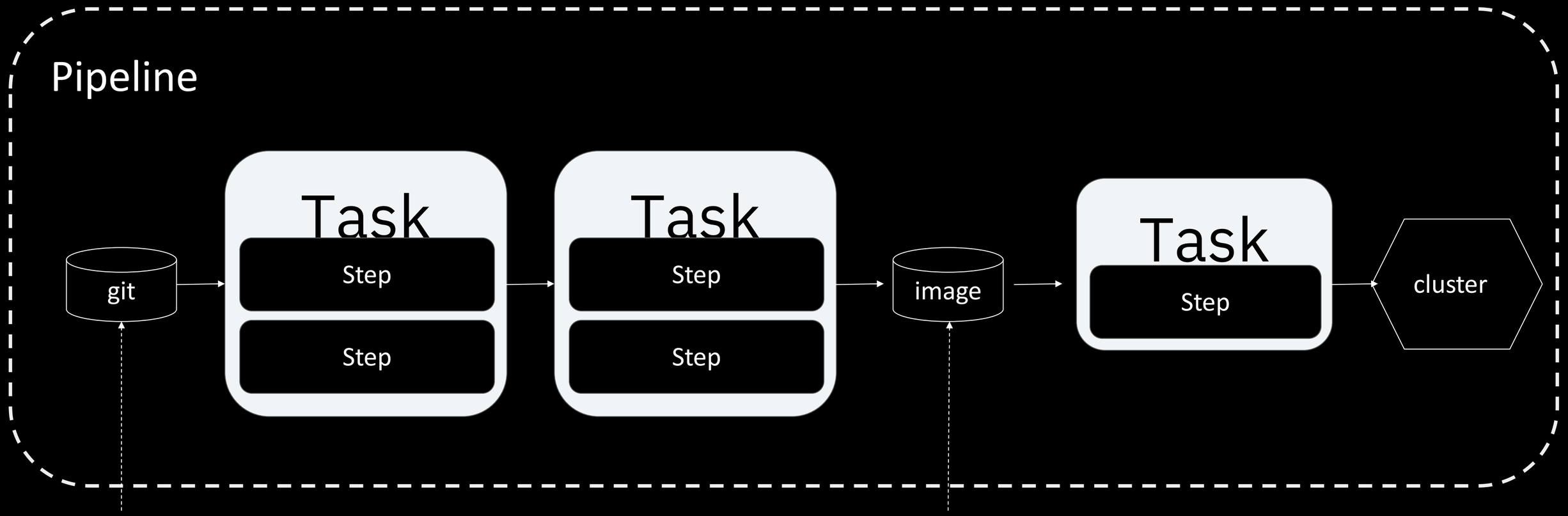
TEKTON

https://tekton.dev/

Pipelines - Tekton

- Governed by the Continuous Delivery Foundation (cd.foundation)
- Contributions from Google, Cloudbees, IBM, Pivotal, Red Hat and more
- Originated from the Knative Build subproject
- Build your linux container images in-cluster
- Automate deployments
- Defines new Kinds via CRDs: Pipeline, Task
- Reusable Tasks (https://github.com/tektoncd/catalog)
  - git clone
  - mvn, bazel, s2i (python, ruby, etc)
  - "docker build" (buildah, kaniko, makisu)

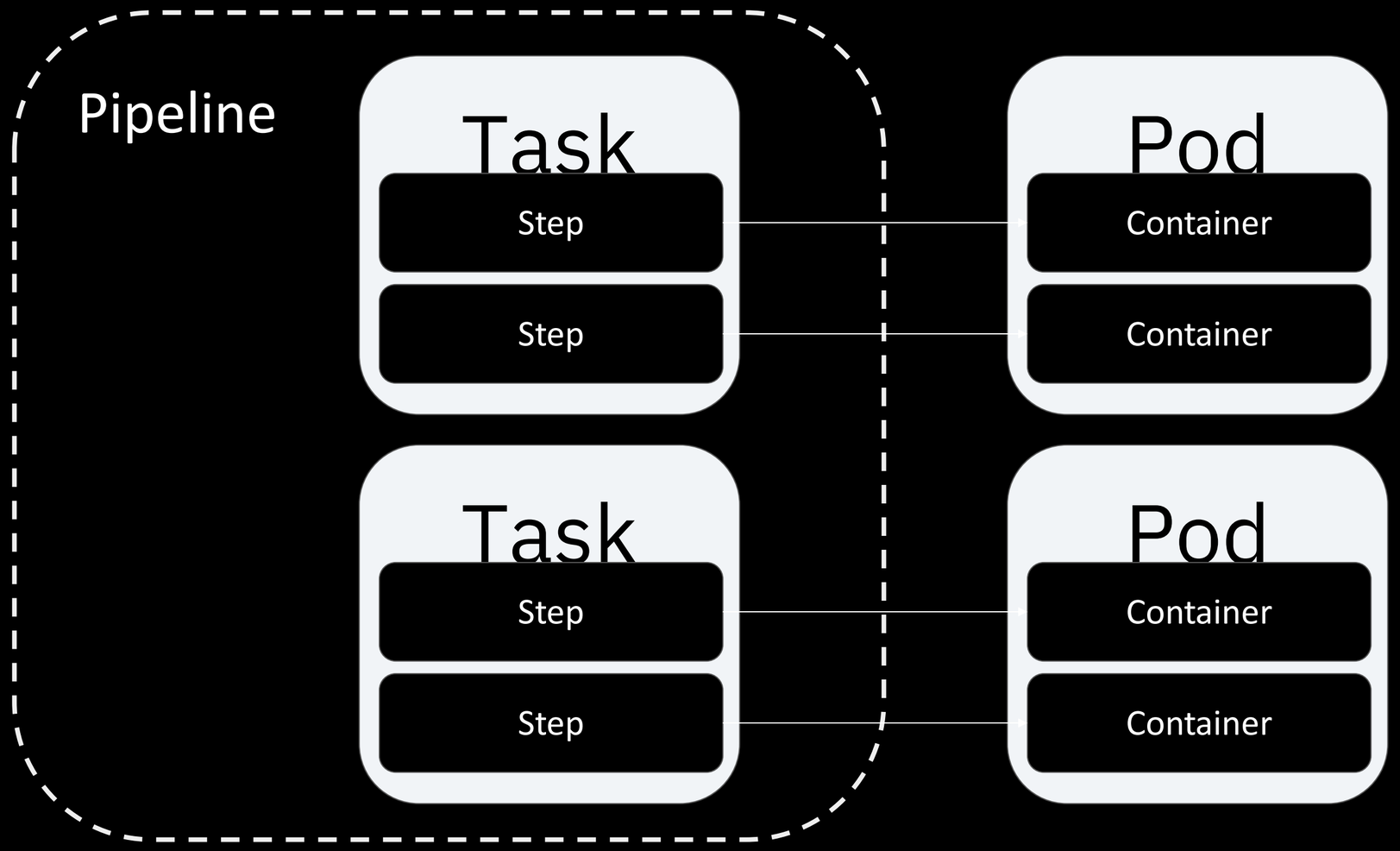
Pipelines, Tasks, Steps, Resources



Pipeline Resourc

Pipeline Resourc

Pipeline: Tasks & Steps -> Pods & Containers



Knative Eventing

Benefits

- Orchestrate in/out event cluster event sources
- Binds declaratively event sources, triggers, and services
- Scales from just few events to real time streaming message pipelines
- Use standard based specification CNCF CloudEvents (cloudevents.io)

Extendable

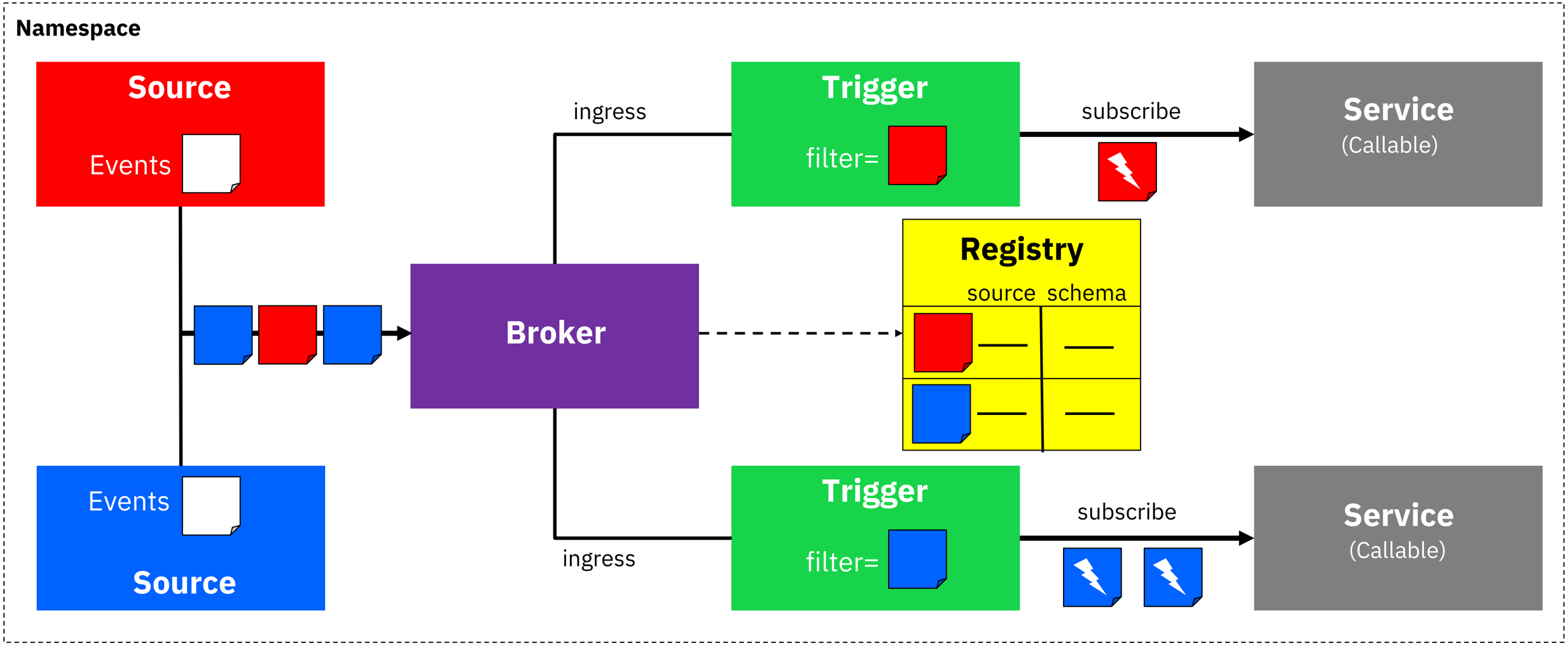
- Use existing event sources or build your own
- Pluggable channel implementation (e.g. PubSub. Kafka, In-Memory, Cron)
- Abstractions to route events from a central broker to filter based triggers

Knative Event Trigger

- Delivers events to a target as Cloud Events
- Provides filtering of events
  - Only PR events for specific GitHub Project
  - Only Kafka messages from topic stocks
- Trigger can produce new events
  - Output goes back to the Broker

```yaml
apiVersion: eventing.knative.dev/v1alpha1
kind: Trigger
metadata:
  name: t
spec:
  filter:
    sourceAndType:
      type: com.example.someevent
  subscriber:
    ref:
      apiVersion: serving.knative.dev/v1alpha1
      kind: Service
      name: message-dumper
```

Knative Eventing



Knative Event Sources available

Name	Description
Apache Camel	Allows to use Apache Camel components for pushing events into Knative
Apache Kafka	Brings Apache Kafka messages into Knative
AWS SQS	Brings AWS Simple Queue Service messages into Knative
Cron Job	Uses an in-memory timer to produce events on the specified Cron schedule.
GCP PubSub	Brings GCP PubSub messages into Knative
GitHub	Brings GitHub organization/repository events into Knative
GitLab	Brings GitLab repository events into Knative.
Google Cloud Scheduler	Google Cloud Scheduler events in Knative when jobs are triggered
Google Cloud Storage	Brings Google Cloud Storage bucket/object events into Knative
Kubernetes	Brings Kubernetes cluster/infrastructure events into Knative

https://github.com/knative/docs/tree/master/docs/eventing/sources#sources

Knative Eventing Sources

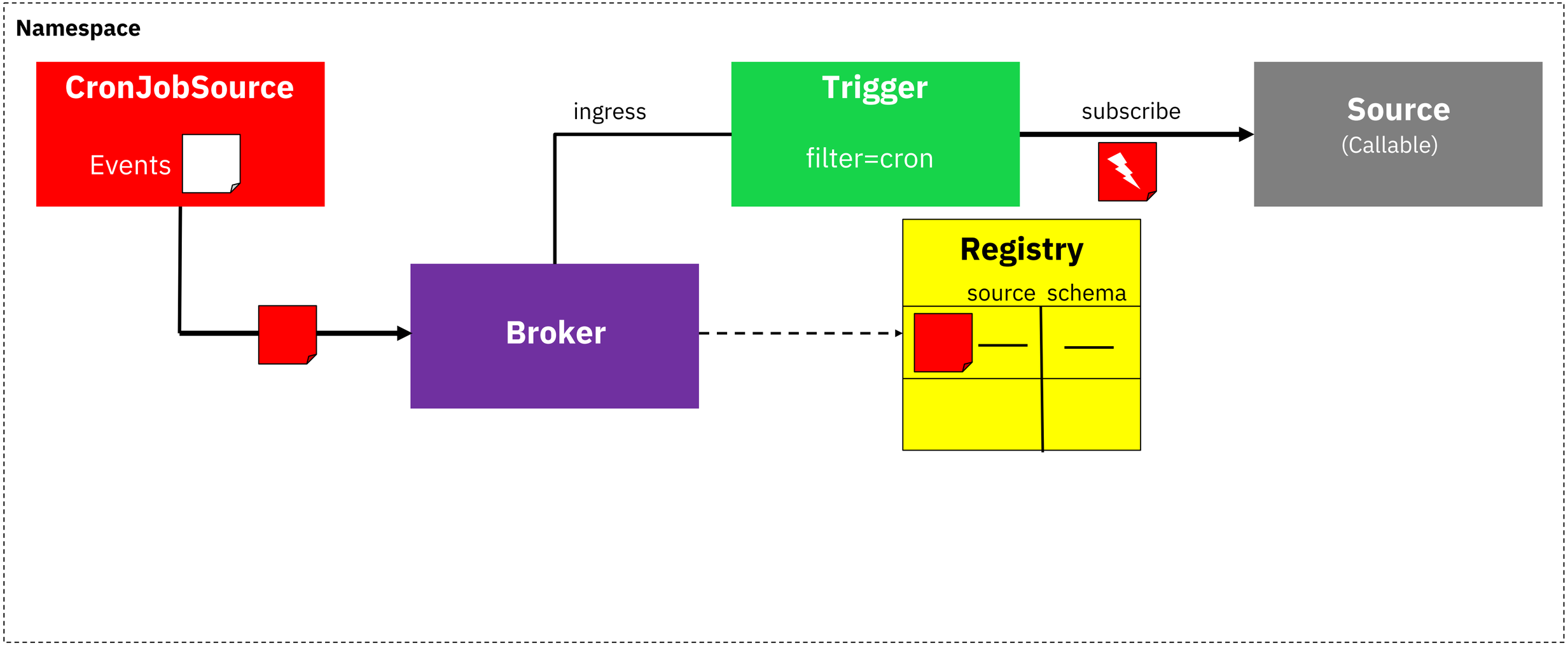
User Perspective

- Discover new event sources
- Configures external event sources
- Write new ones

Benefits

- Declarative create subscriptions
- List subscriptions
- Modify – changes reconciled
- Repeatable
- Clear separation of concerns
  - External / Broker

Knative Eventing



1 Source

```
$ kubectl apply -f source.yaml
```

```yaml
apiVersion: sources.eventing.knative.dev/v1alpha1
kind: CronJobSource
metadata:
  name: test-cronjob-source-broker
spec:
  schedule: "*/2 * * * *"
  data: '{"message": "Hello world!"}'
  sink:
    apiVersion: eventing.knative.dev/v1alpha1
    kind: Broker
    name: default
```

2 Registry

```
$ kubectl get eventtypes
```

NAME	TYPE	SOURCE	SCHEMA	BROKER
dev.knative.cronjob.event-z7th8	dev.knative.cronjob.event	/apis/v1/namespaces/default/cronjobsources/test-cronjob-source-broker		default

TYPE

dev.knative.cronjob.event

3 Trigger

```
$ kubectl apply -f trigger.yaml
```

```yaml
apiVersion: eventing.knative.dev/v1alpha1
kind: Trigger
metadata:
  name: trigger-cron
spec:
  filter:
    sourceAndType:
      type: dev.knative.cronjob.event
  subscriber:
    ref:
      apiVersion: serving.knative.dev/v1alpha1
      kind: Service
      name: event-display
```

Resources

https://knative.dev
https://github.com/IBM/knative101
https://cloud.ibm.com/docs/containers?topic=containers-serverless-apps-knative
https://developer.ibm.com/articles/knative-what-is-it-why-you-should-care
https://developer.ibm.com/videos/build-your-first-app-on-knative-serverless-tech-talk/

https://learning.oreilly.com/live-training/courses/introduction-to-knative/0636920297840
https://learning.oreilly.com/live-training/courses/kubernetes-serverless-with-knative/0636920271055

https://learning.oreilly.com/library/view/getting-started-with/9781492047025/

https://redhat-developer-demos.github.io/knative-tutorial/knative-tutorial/0.5.0/index.html

https://serverless-architecture.io/blog/what-exactly-is-knative-an-interview-with-evan-anderson-from-google
https://thenewstack.io/how-knative-can-unite-kubernetes-and-serverless
https://containerjournal.com/2019/02/13/what-is-knative-and-what-can-it-do-for-you

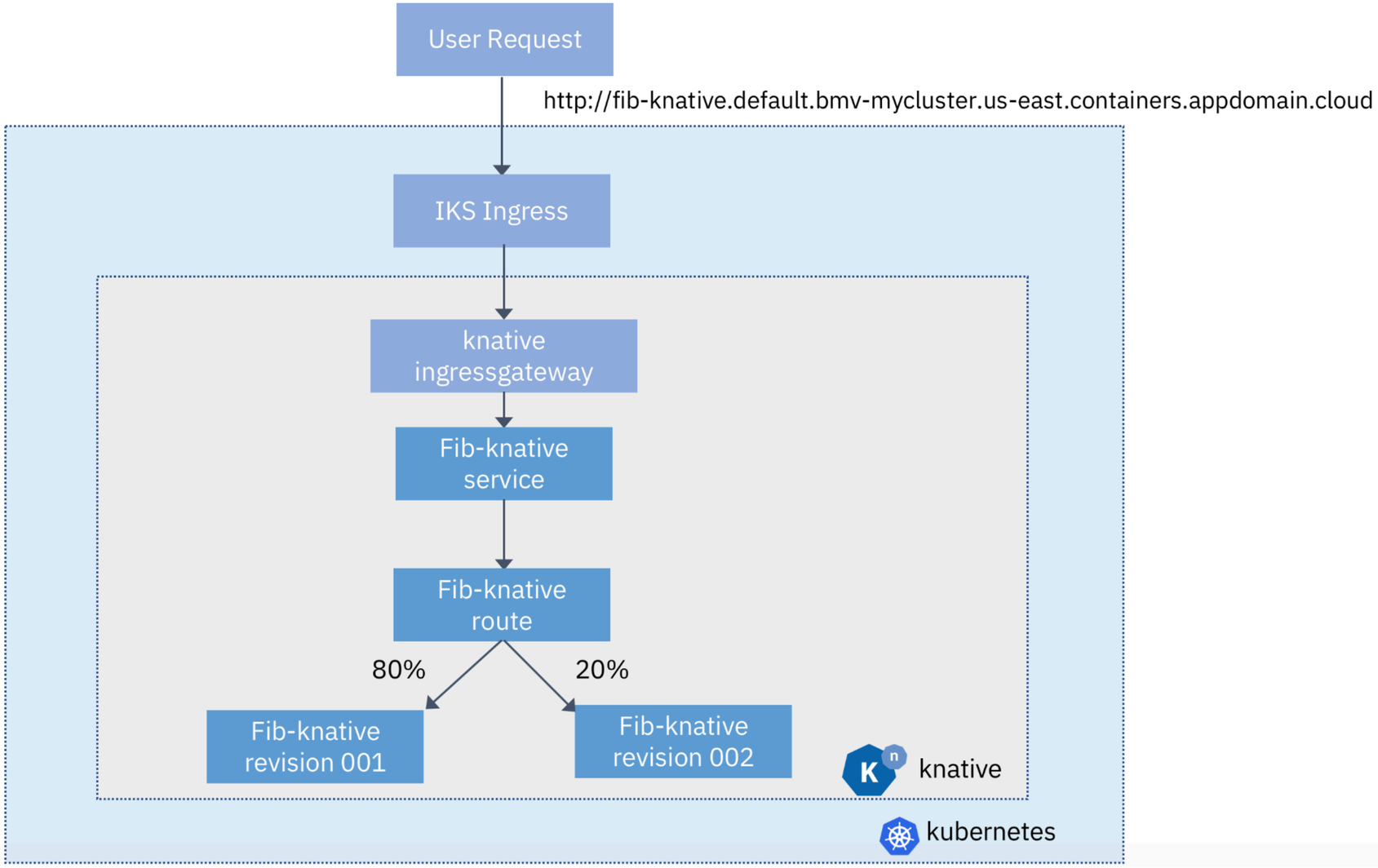
https://developer.ibm.com/tutorials/knative-build-app-development-with-tekton/

https://github.com/csantanapr/knative-ibm-developer
https://github.com/csantanapr/knative-kubecon19eu
https://github.com/csantanapr/knative-eventing-basics
https://github.com/csantanapr/knative-think2019
https://github.com/knative/serving-operator

IBM Cloud © 2019 IBM Corporation

Presentation and Lab

https://jaffas.gitbook.io/knative101/



Thank you

 / © 2019 IBM Corporation