

Modern event-driven Workloads with Knative

Roland Huß [@ro14nd@mastodon.green](https://ro14nd.mastodon.green)
Senior Principal Software Engineer, Red Hat
OpenShift Serverless Architect
Ex-Knative TOC member

Matthias Weßendorf [@matzew@hachyderm.io](mailto:matzew@hachyderm.io)
Principal Software Engineer, Red Hat
OpenShift Serverless Engineer
Knative contributor



Wait ... wat ?



Serverless

“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”

-- CNCF Definition, <https://www.cncf.io/blog/2018/02/14/cncf-takes-first-step-towards-serverless-computing/>

Serverless vs. FaaS

Serverless is a **Deployment Model** that abstracts away the driving machine infrastructure.

- No server management required
- Executed, scaled and billed according to demand
- Defines a deployment packaging, but otherwise agnostic to the application

FaaS (Function-as-a-Service) is a **Programming Model** that mandates developing your application with fine grained function that match a given signature.

- Deployed as Serverless application
- Typically used as *glue code* to connect services

A photograph of a traditional wooden longtail boat sailing on the ocean. The boat's hull is dark wood, and its deck is made of light-colored planks. A large wooden pole (steering oar) is visible at the stern. The boat is moving towards a group of green, craggy limestone islands under a blue sky with white clouds.

Knative

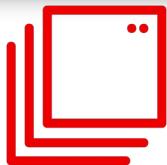
Kubernetes-based platform to **deploy and manage modern serverless workloads.**

<https://knative.dev>

Bringing Serverless applications to Kubernetes

Serving

A request-driven model that serves the container with your application and can "scale to zero".



Eventing

Common infrastructure for consuming and producing events that will stimulate applications.



Functions

A programming model that lets you focus on just your code for faster iterations.



Background Information

- Started as an **Open Source** Project mid-2018 by Google
- CNCF incubating project since 03/2022
- Community driven with a lot of vendor backing
 - <https://github.com/knative>
 - <https://knative.dev>
 - Support by Google, Red Hat, IBM, VMware, Triggermesh, SAP and more
 - Organized in multiple Working Groups with weekly meetings
 - <https://knative.dev/docs/contributing/about/>
- Releases
 - Current: **v1.8.0**
 - 6 week release cadence, switching to a 4 releases / year cadence

Try Knative !

- Install from resource descriptors on Kubernetes Cluster
 - <https://knative.dev/docs/install/>
- Google **Cloud Run** (managed and on GKE)
 - <https://cloud.google.com/run/>
 - Not all Knative features implemented
 - see <https://ahmet.im/blog/cloud-run-is-a-knative> ?
- IBM Cloud **Code Engine**
 - <https://www.ibm.com/cloud/code-engine>
 - Vanilla Knative, additional workloads (like batch)
- Red Hat **OpenShift Serverless**
 - <https://www.openshift.com/learn/topics/serverless>
 - Supports all Knative features

Serving



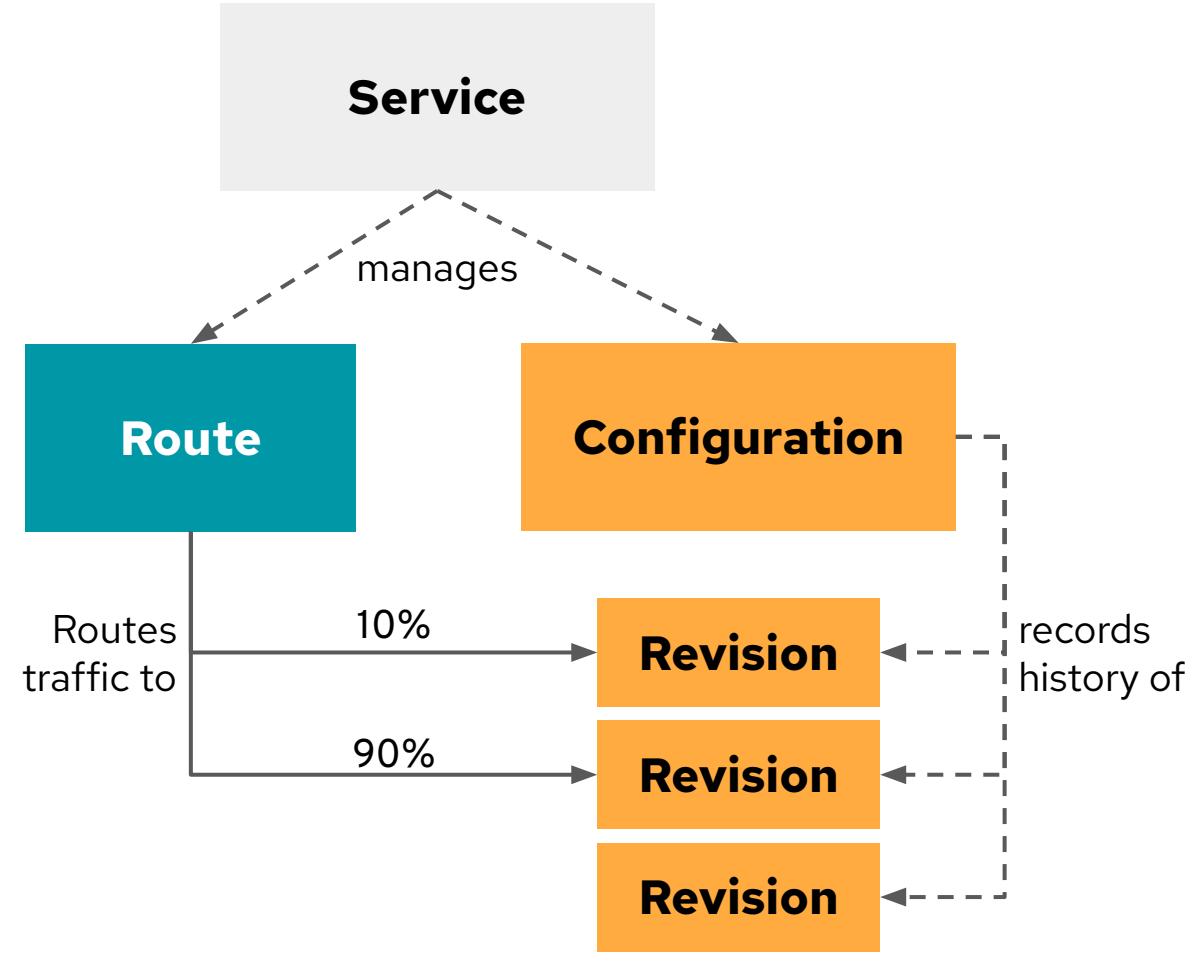
Concepts

- **Demand-based autoscaling**, including scale-to-zero
- Separation of code and configuration
- Opinionated and simplified deployment model catered for **stateless applications**
- Rich **traffic split capabilities** to enable custom rollout strategies of new versions

**Route, scale-to-zero and track
application revisions with ease.**

Resources

- **Configuration** represent the *floating HEAD* of a history of **Revisions**
- **Revision** represents an immutable snapshot of code and configuration
- **Route** configure ingress over a collection of Revisions
- **Service** (not K8s services !) is a top-level entity that manage a set of Routes and Configurations



From Deployment to KService

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: random
spec:
  replicas: 1
  selector:
    matchLabels:
      app: random
  template:
    metadata:
      labels:
        app: random
    spec:
      containers:
        - image: rhuss/random
          name: random
        ports:
          - containerPort: 8080
```

```
apiVersion: serving.knative.dev/v1
kind: Service
metadata:
  name: random
spec:
  replicas: 1
  selector:
    matchLabels:
      app: random
  template:
    metadata:
      labels:
        app: random
    spec:
      containers:
        - image: rhuss/random
          name: random
        ports:
          - containerPort: 8080
```

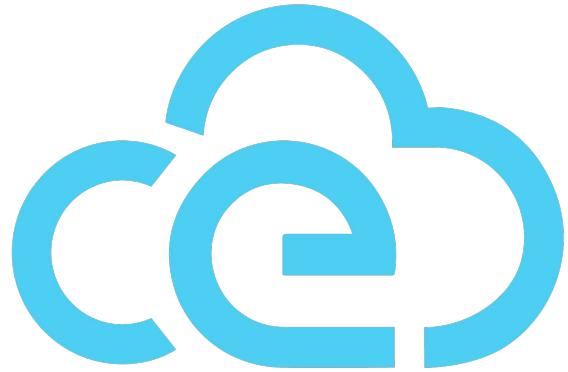
No more K8s
Service or
Ingress
required!

Demo

A panoramic view of the Hamburg Speicherstadt (Warehouse District) during sunset. The scene is framed by two long, red-brick buildings with many small arched windows. A narrow canal runs through the center, with a metal arch bridge spanning it. In the background, a modern skyscraper rises against a sky filled with warm, orange and yellow clouds. The water of the canal reflects the vibrant colors of the sunset.

Eventing

CNCF CloudEvents



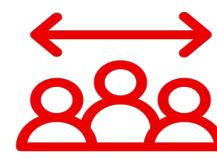
Consistency



Accessibility



Portability



<https://cloudevents.io/>

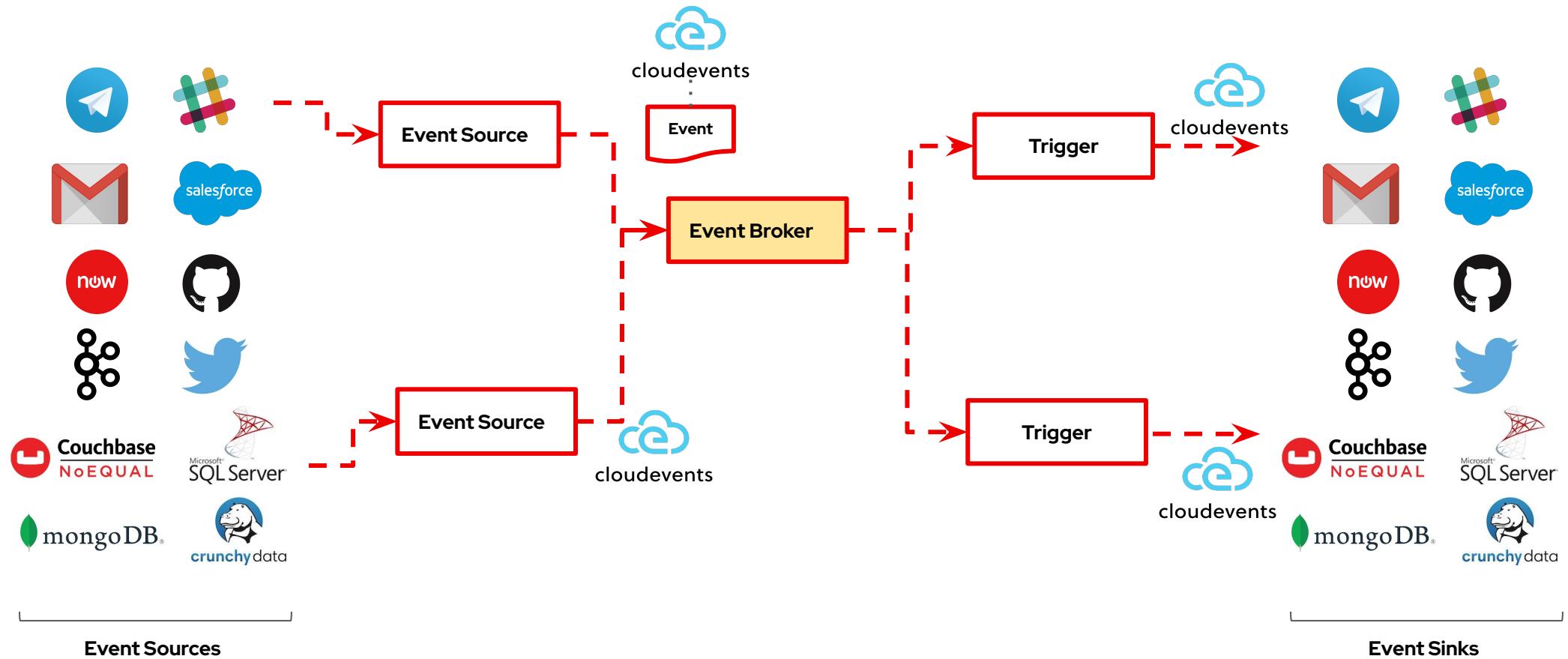
- Backbone for Messaging in Knative Eventing
- Common format for message and event interchange across disparate and heterogeneous services

¹⁷

ro14nd@mastodon.green
matzew@hachyderm.io

 Red Hat

Event Driven Applications



Event Sources

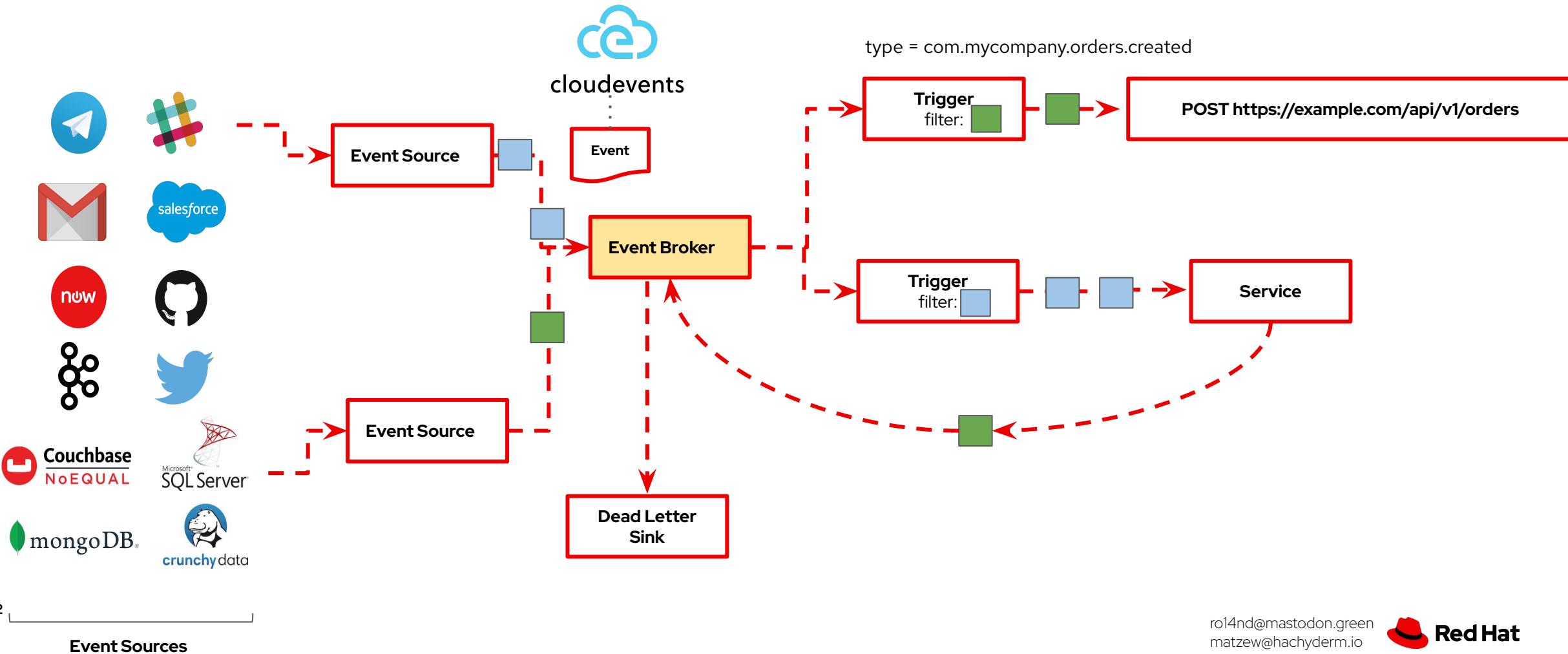
- Integrating 3rd party systems with Knative
- More often “**Adapter**” than an original event source
- Declared with a **Custom Resource**
- Evaluated by an Operator
- Push or Pull based
- Converting custom event formats to **CloudEvents**

Event Sources for integration

Builtin Sources	
PingSource	Emitting static CloudEvents periodically
ApiServerSource	Kubernetes API Server events as CloudEvents
SinkBinding	Binds an arbitrary Pod specification to a Sink
ContainerSource	Meta-Source combining SinkBinding & Deployment
Contributed Sources	
GitHubSource	Converts GitHub webhooks events to CloudEvents
KafkaSource	Import Apache Kafka records as CloudEvents
RabbitMQSource	Import RabbitMQ messages as CloudEvents

Broker API	
Broker	Brokers provide a well-known endpoint for event delivery that senders can use with minimal knowledge of the event routing strategy.
Implementation (via class annotation)	
Channel Broker	Reference implementation, based on Channel
Kafka Broker	Ingress/Egress of Apache Kafka records as CloudEvents (binary)
Rabbit Broker	Uses RabbitMQ for persistence

Knative Event Mesh



Event Processing

- Use **filter**(s) on Triggers for CloudEvent attributes
 - *Do not subscribe to all events (especially type)*
- Return **application specific** CloudEvents in response to the delivery
 - Success: return a new event (avoid filter loops on type)
 - Error: return new event for application level errors
 - provide context of failing payload, if possible
 - Use different types for return, to avoid filter loops
- Use **Dead-Letter-Sinks** to catch not-delivered events
 - there can be lot's of things going wrong

Event Processing

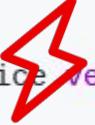
```
@Funq
public CloudEvent<ValidOrder> function(CloudEvent<Order> newOrder) {

    // call to external system
    final ValidOrder vo = orderService.verify(newOrder.data());
    final LocalDate eta = vo.getDeliveryDate();

    // return application specific events:
    if (eta.isAfter(LocalDate.now().plusDays(30))) {
        return CloudEventBuilder.create()
            .specVersion("1.0")
            .type("delayed.order.created")
            .source("http://my-corp.com/orders")
            .subject("delayed")
            .id(UUID.randomUUID().toString())
            .build(vo);
    } else {
        return CloudEventBuilder.create()
            .specVersion("1.0")
            .type("order.created")
            .source("http://my-corp.com/orders")
            .subject("ontime")
            .id(UUID.randomUUID().toString())
            .build(vo);
    }
}
```

Event Processing

```
@Funq
public CloudEvent<ValidOrder> function(CloudEvent<Order> newOrder) {

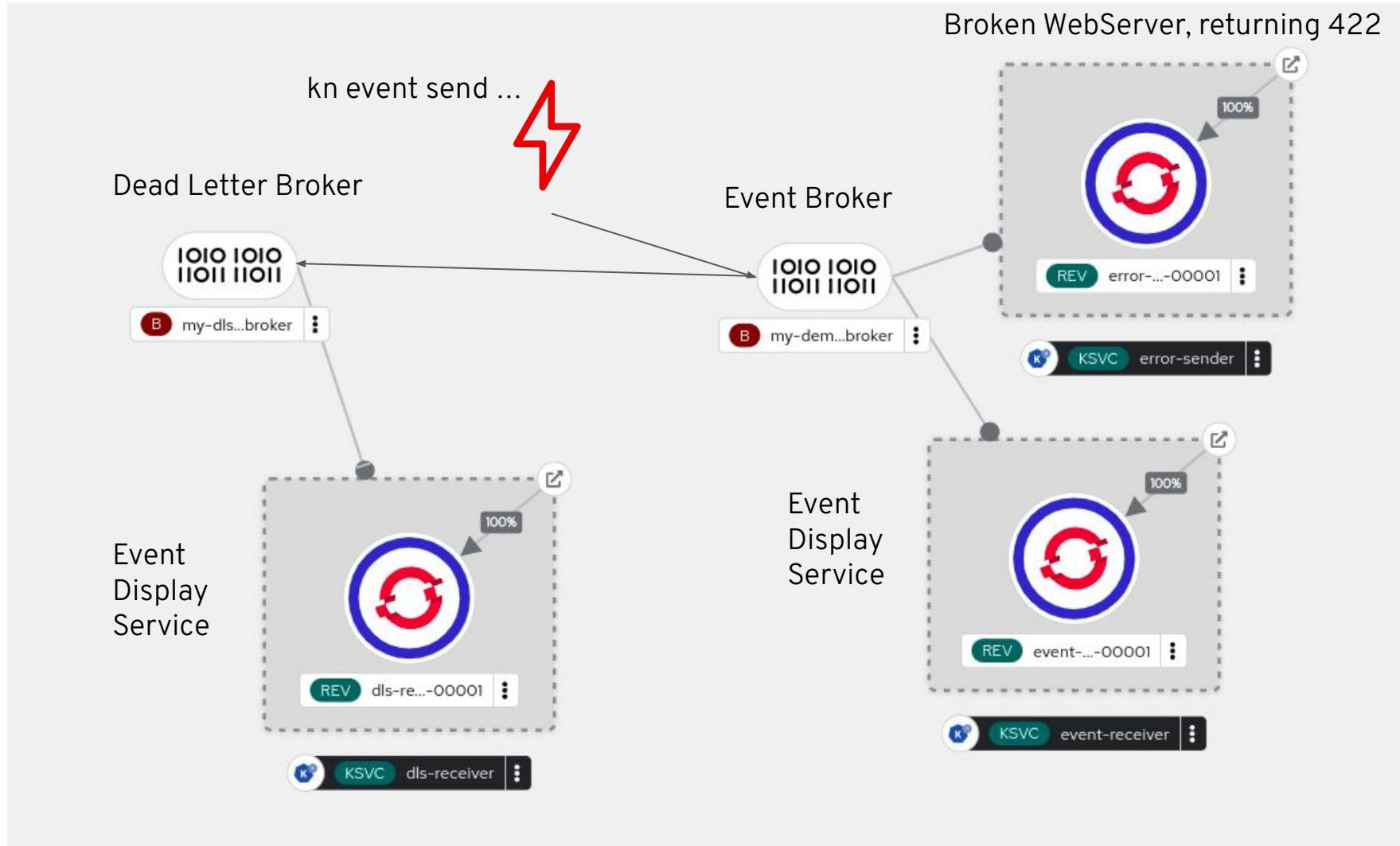
    // call to external system
    final ValidOrder vo = orderService.verify(newOrder.data());

    // return a CloudEvent with the checked order
    return CloudEventBuilder.create().type(.....)....build(vo);
}
```

Event Delivery Spec

- Concept of dead letter sink
- Retries
- Backoff
- Well defined retry behavior,
based on HTTP status codes:

Response code	Meaning	Retry	Delivery completed	Error
1xx	(Unspecified)	No*	No*	Yes*
200	Accepted, event in reply	No	Yes	No
202	Event accepted	No	Yes	No
other 2xx	(Unspecified)	No	Yes	No
3xx	(Unspecified)	No*	No*	Yes*
400	Unparseable event	No	No	Yes
404	Endpoint does not exist	Yes	No	Yes
408	Request Timeout	Yes	No	Yes
409	Conflict / Processing in progress	Yes	No	Yes
429	Too Many Requests / Overloaded	Yes	No	Yes
other 4xx	Error	No	No	Yes
5xx	Error	Yes	No	Yes

Demo



(Kafka) Broker config

- Every broker has its own ConfigMap
- Setup a dedicated **broker** as the dead letter sink of the **broker**
 - Use Triggers to filter based on error code
- Override (delivery) defaults only when needed
- Kafka: 10 partitions and 3 replication nodes

New Trigger filters

- **CNCF Subscriptions API**

- Experimental feature in Knative Eventing
 - Needs to be enabled via ConfigMap
- Based on CloudEvent metadata
- Supported dialects
 - Exact, prefix, suffix, all, any, not, cesql

```
apiVersion: eventing.knative.dev/v1
kind: Trigger
metadata:
  name: my-suffix-trigger
spec:
  broker: default
  filters:
    - suffix:
        type: .created:
```

New Trigger filters

- **CNCF Subscriptions API**

- Experimental feature in Knative Eventing
 - Needs to be enabled via ConfigMap
- Based on CloudEvent metadata
- Supported dialects
 - Exact, prefix, suffix, all, any, not, **cesql**

```
apiVersion: eventing.knative.dev/v1
kind: Trigger
metadata:
  name: little-bobby-tables
spec:
  broker: default
  filters:
    - cesql: "source LIKE '%commerce%' AND type IN ('order.created', 'order.updated', 'order.canceled')"
```

Kamelets

Kamelets

- Part of **Camel-K**, a runtime platform on top of Kubernetes for Apache Camel routes
- Snippets that contain a **route template**
- Types of Kamelets
 - **Sources** for incoming cloud events
 - **Sinks** for outgoing cloud events
 - **Actions** for transformations
- Kamelets are instantiated via **KameletBindings**
 - Filling in Kamelet parameters, like authentication information
 - Connecting to sink (for sources)
- kn-source-kamelet : Kn **plugin** for managing Kamelets

Functions

kn-func

- Opinionated **programming model** for Knative
 - Scaffolding of project templates for multiple runtimes
 - Quarkus, Node.js, Spring, Python, Rust, Go...
 - Building and pushing container images
 - Deploying as Knative services
- Available as plugin **kn func** or standalone CLI **func**
 - <https://github.com/knative/func>
- **Local development** mode
- Technologies
 - Cloud-native Buildpacks
 - Local Docker or Podman
 - On-cluster builds

Demo

Summary



Summary

Knative Serving

- Simplified Deployment for stateless workloads
- Traffic based autoscaling including Scale-to-Zero
- Traffic splitting for custom rollout / rollback scenarios

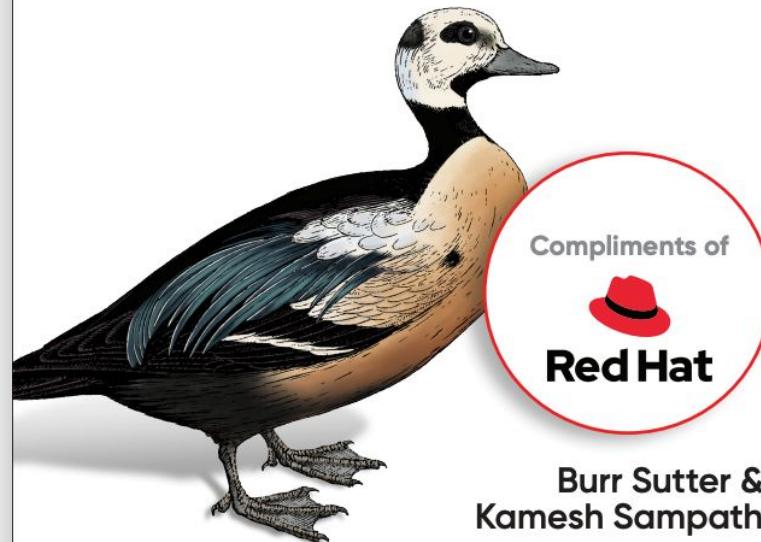
Knative Eventing

- External Triggers for feeding Knative Services
- Based on CNCF CloudEvents
- Backed by proven messaging systems
- Flexible messaging setup

O'REILLY®

Knative Cookbook

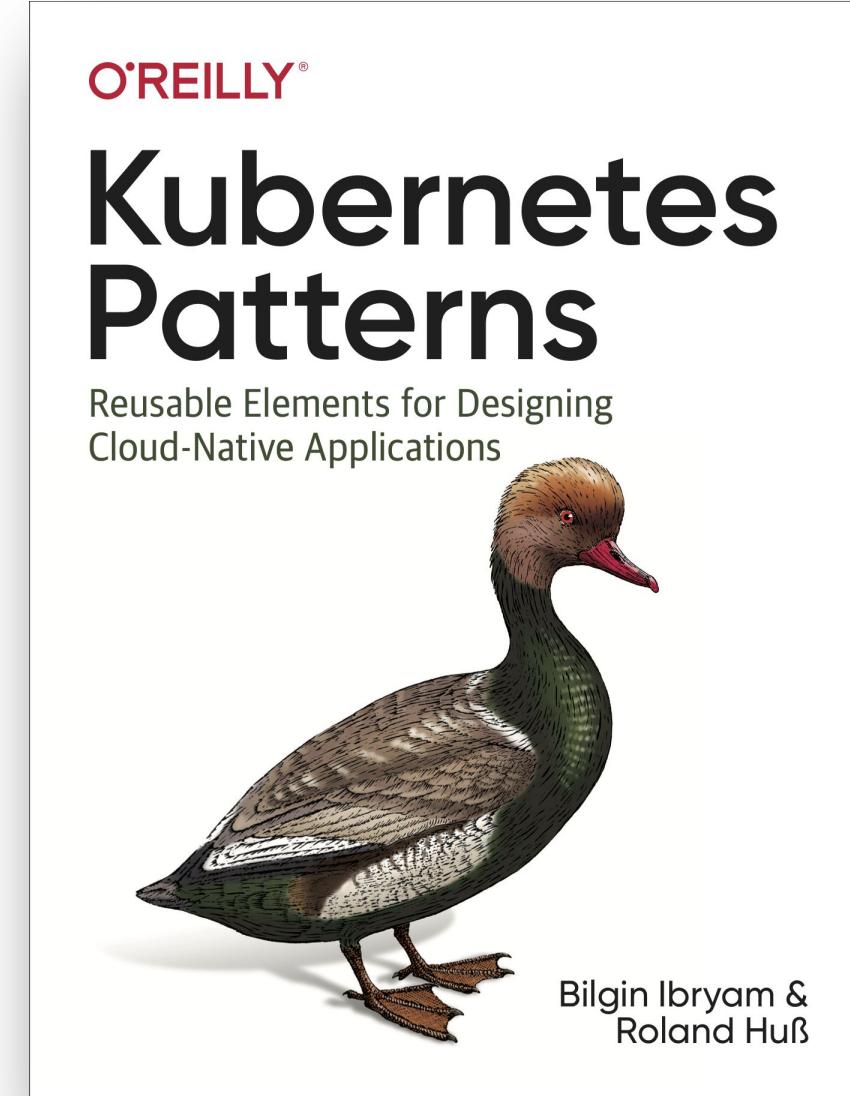
Building Effective Serverless Applications
with Kubernetes and OpenShift



<http://dn.dev/knative-cookbook>

ro14nd@mastodon.green
matzew@hachyderm.io





<https://k8spatterns.io>

ro14nd@mastodon.green
matzew@hachyderm.io



Thank you



ro14nd@mastodon.green
matzew@hachyderm.io

<https://github.com/matzew/wjax2022-knative>

Picture Credits

<https://www.pexels.com/photo/boat-island-ocean-sea-218999/>

<https://unsplash.com/photos/t6t2-gXKxXM>

<https://unsplash.com/photos/UGMf30W28qc>

<https://pixabay.com/photos/hamburg-speicherstadt-channel-2976711/>

<https://pixabay.com/photos/beer-machine-alcohol-brewery-1513436/>

<https://pixabay.com/photos/camel-sunset-landscape-tourism-2500618/>

<https://unsplash.com/photos/9SWHIgu8A8k>

<https://me.me/i/aws-lambda-is-just-glorified-cgi-bin-imgflip-com-change-my-mind-d0b715592ba34b08b79452ad02783ca2>

https://unsplash.com/photos/dodn_0TESNO

<https://pixabay.com/photos/annoy-cells-stars-dendrites-sepia-2926087/>