Going Serverless with Knative

The future is already here...

Nicola Ferraro

Principal Software Engineer

CNCF Napoli

09/10/2019

Agenda

1. Serverless

2. Knative Basics

3. Event Mesh

4. Demo

5. Wrap up

About me

2009   2019

Nicola Ferraro
Principal Software Engineer at Red Hat
Twitter: **@ni_ferraro**

I work on:
- Apache Camel and Camel K (+ Knative Sources)
- Syndesis
- Red Hat Fuse
- Fabric8 Dev Tools

SERVERLESS

Serverless: what does it mean?

No! It does **not** mean: *"No servers"*

And no! It does **not** mean: *"The servers of someone else"*

However…….

Serverless: what does it mean?

Serverless ≠ AWS Lambda (FaaS)

Started in 2014

AWS Lambda lets you **run code without provisioning or managing servers**.

You **pay only for the compute time** you consume -

there is **no charge when** your code is **not running**.

And the term "serverless" was coined two years before Lambda (2012):
http://readwrite.com/2012/10/15/why-the-future-of-software-and-apps-is-serverless/

Red Hat

Serverless: new FaaS

There are many (other) FaaS platform under the sun.

Serverless: CNCF

*"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 <u>functions</u>, 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/
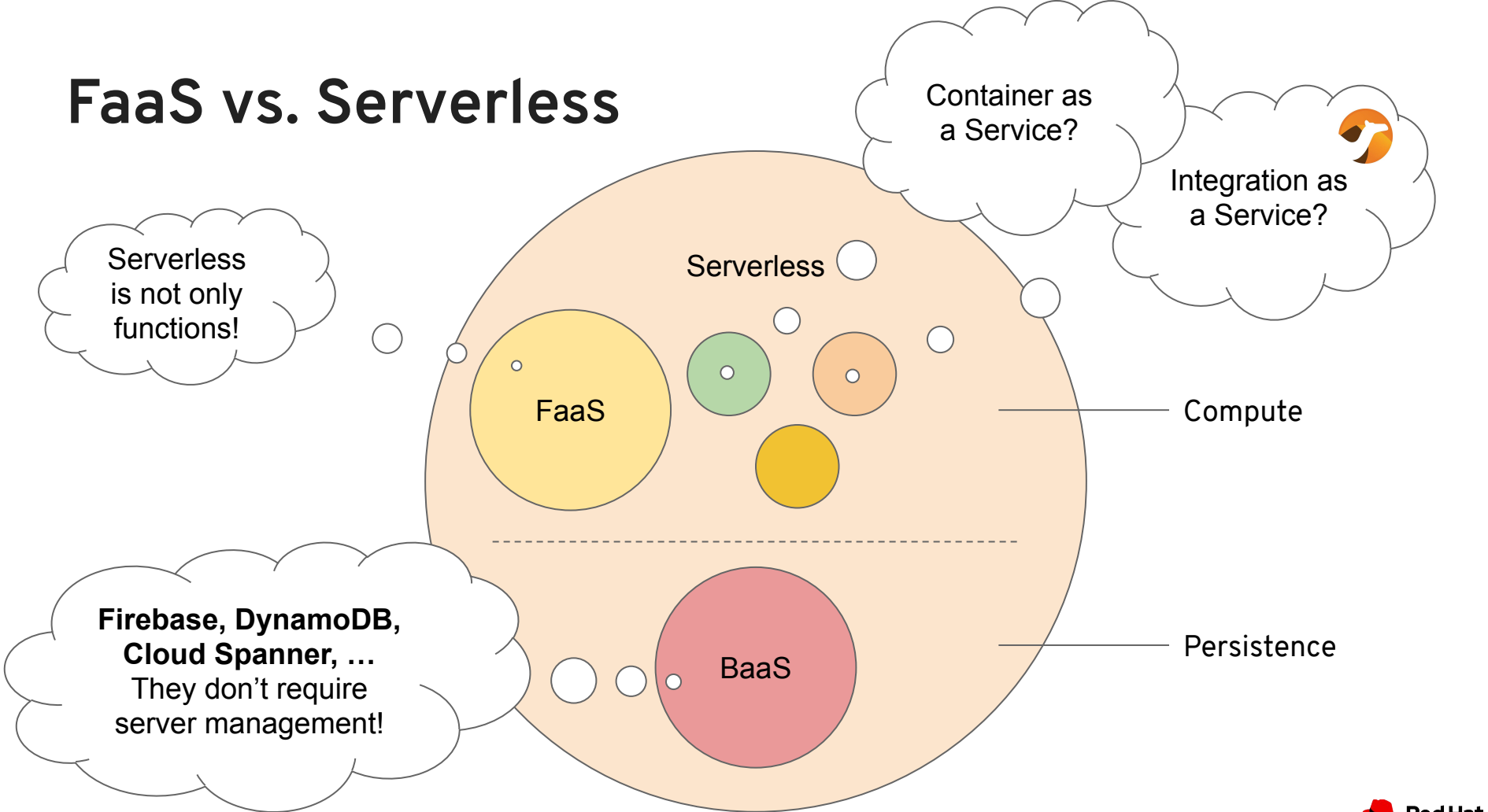
But...

Kelsey Hightower ✓
@kelseyhightower

Serverless has evolved beyond FaaS and functions are
an implementation detail.

FaaS vs. Serverless

Serverless
is not only
functions!

Container as
a Service?

Integration as
a Service?

Serverless

FaaS

Compute

**Firebase, DynamoDB,
Cloud Spanner, …**
They don't require
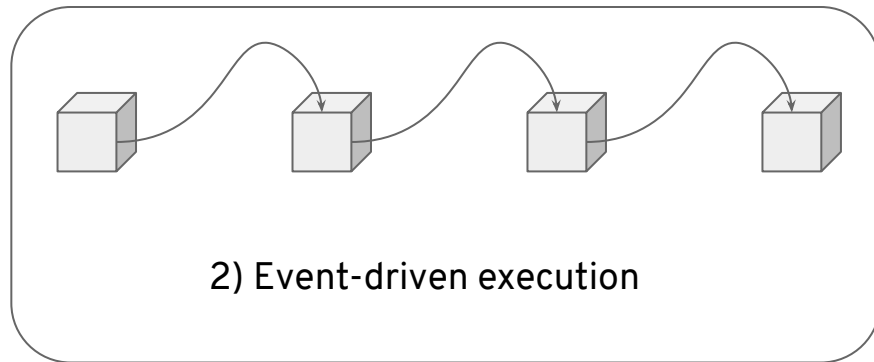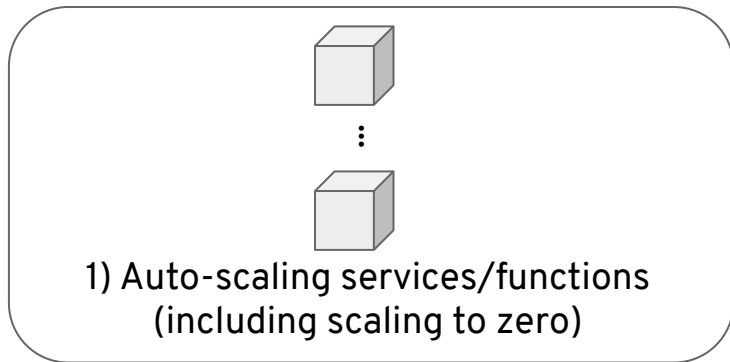server management!

BaaS

Persistence

Red Hat

Serverless (Compute)

Here's my **personal definition** of "serverless".

Before it becomes a billing model, we need to define it "technically".

Serverless is a <u>programming model</u> based on:

1) Auto-scaling services/functions
(including scaling to zero)

2) Event-driven execution
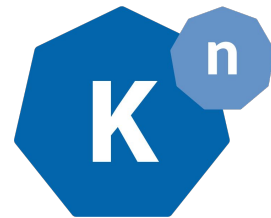
On demand execution...

KNATIVE BASICS

Knative

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

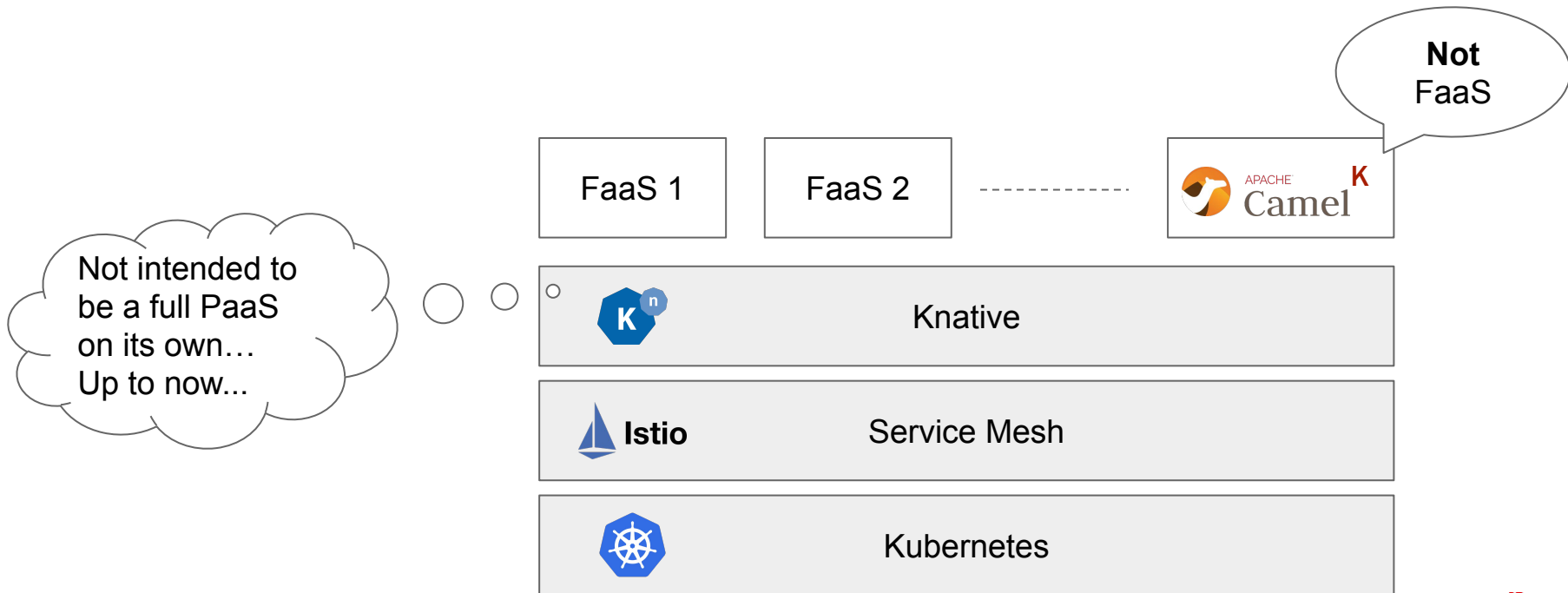Open Source project started in July 2018.

https://knative.dev/

Knative building Blocks

Focuses on "building blocks".

Not
FaaS

| FaaS 1 | FaaS 2 | APACHE Camel K |

Not intended to be a full PaaS on its own… Up to now...

Knative

Istio   Service Mesh

Kubernetes

Knative

Building blocks (a.k.a CRD) for running serverless applications.

Knative Build

<<custom-resource>>
Build

<<custom-resource>>

<<custom-resource>>
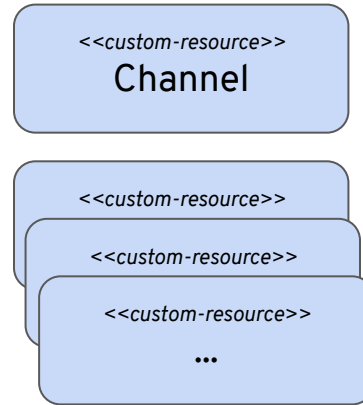
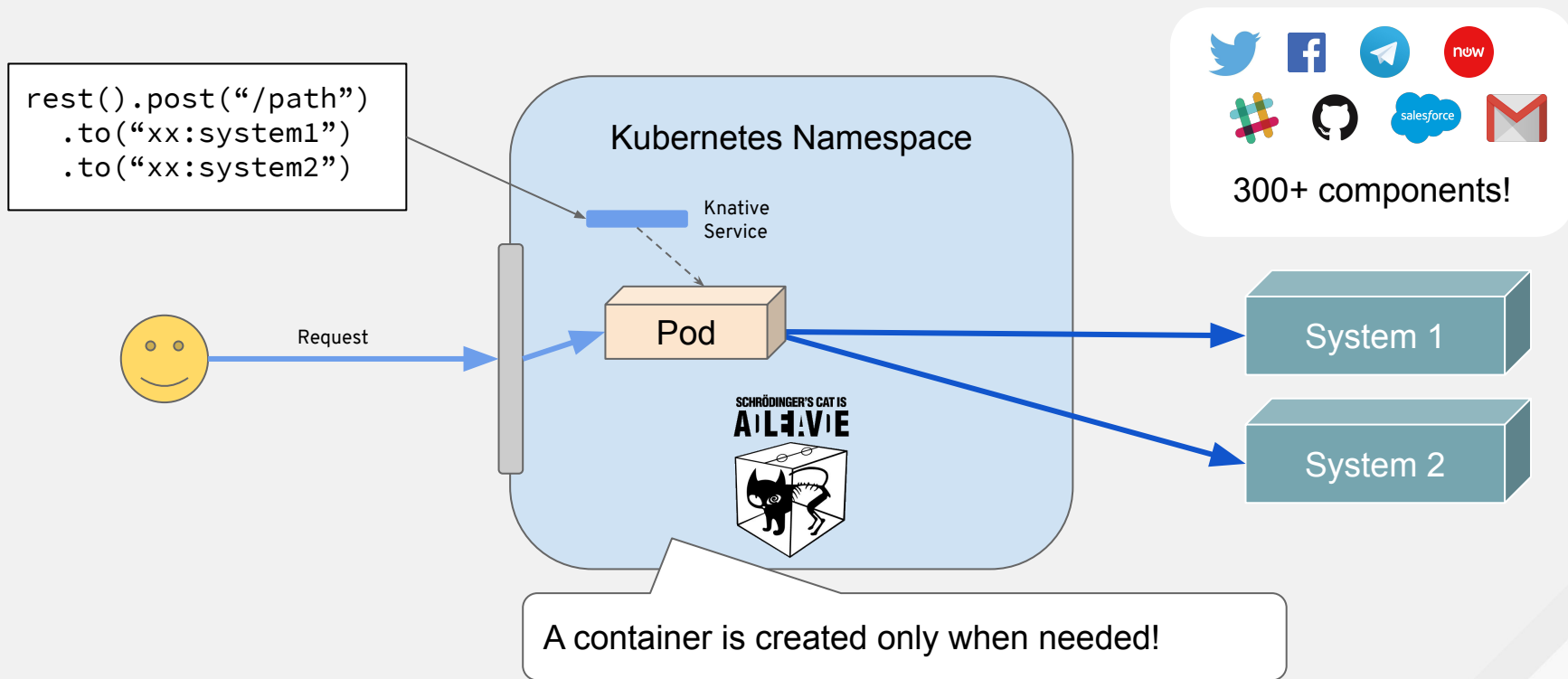<<custom-resource>>

No longer released in 0.7.0.

Can plug external build systems, e.g.
https://github.com/tektoncd/pipeline

Knative Serving

<<custom-resource>>
Service

<<custom-resource>>

<<custom-resource>>

<<custom-resource>>
...
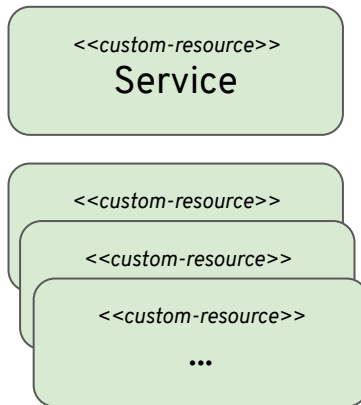
Auto-scaling and scale-to-zero

Knative Eventing

<<custom-resource>>
Channel

<<custom-resource>>

<<custom-resource>>

<<custom-resource>>
...

Messaging for **event-based applications**

Red Hat

What does it mean?

```
rest().post("/path")
  .to("xx:system1")
  .to("xx:system2")
```

Kubernetes Namespace

Knative Service

Pod

SCHRÖDINGER'S CAT IS ALIVE

Request

300+ components!

System 1

System 2

A container is created only when needed!

What does it mean?

```
rest().post("/path")
   .to("xx:system1")
   .to("xx:system2")
```

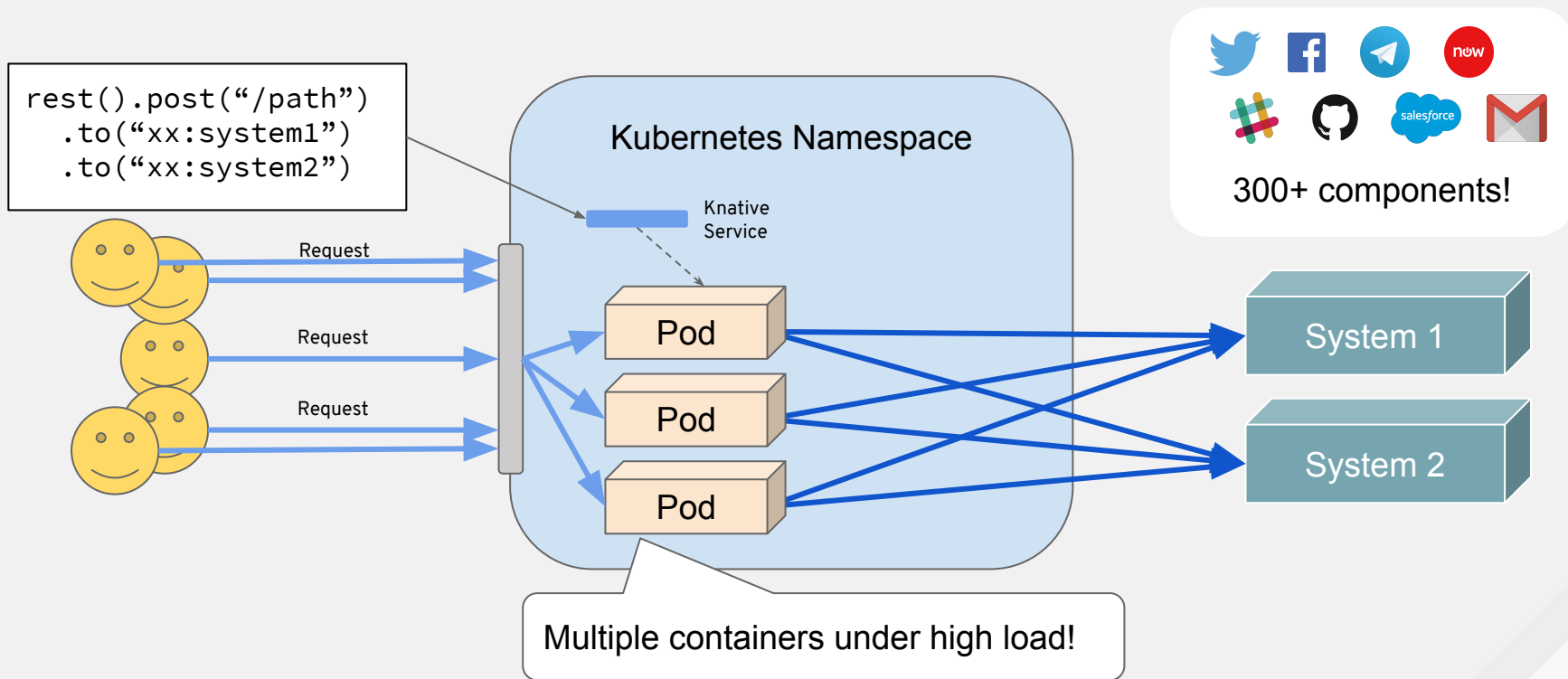Kubernetes Namespace

Knative
Service

SCHRÖDINGER'S CAT IS
A L I V E
D E A D

There's no container if no one needs it!

300+ components!

System 1

System 2

What does it mean?

```
rest().post("/path")
  .to("xx:system1")
  .to("xx:system2")
```

Kubernetes Namespace

Knative Service

Request

Request

Request

Pod

Pod

Pod

Multiple containers under high load!

300+ components!

System 1

System 2

Knative Eventing

```
from("knative:channel/a")
  .to("xx:system1")
  .to("xx:system2")
```

https://cloudevents.io/

Kubernetes Namespace

Knative Service

Knative Subscription

Cloud Event

Channel a

Pod

Backed by an in-memory provisioner, or even a Kafka topic!
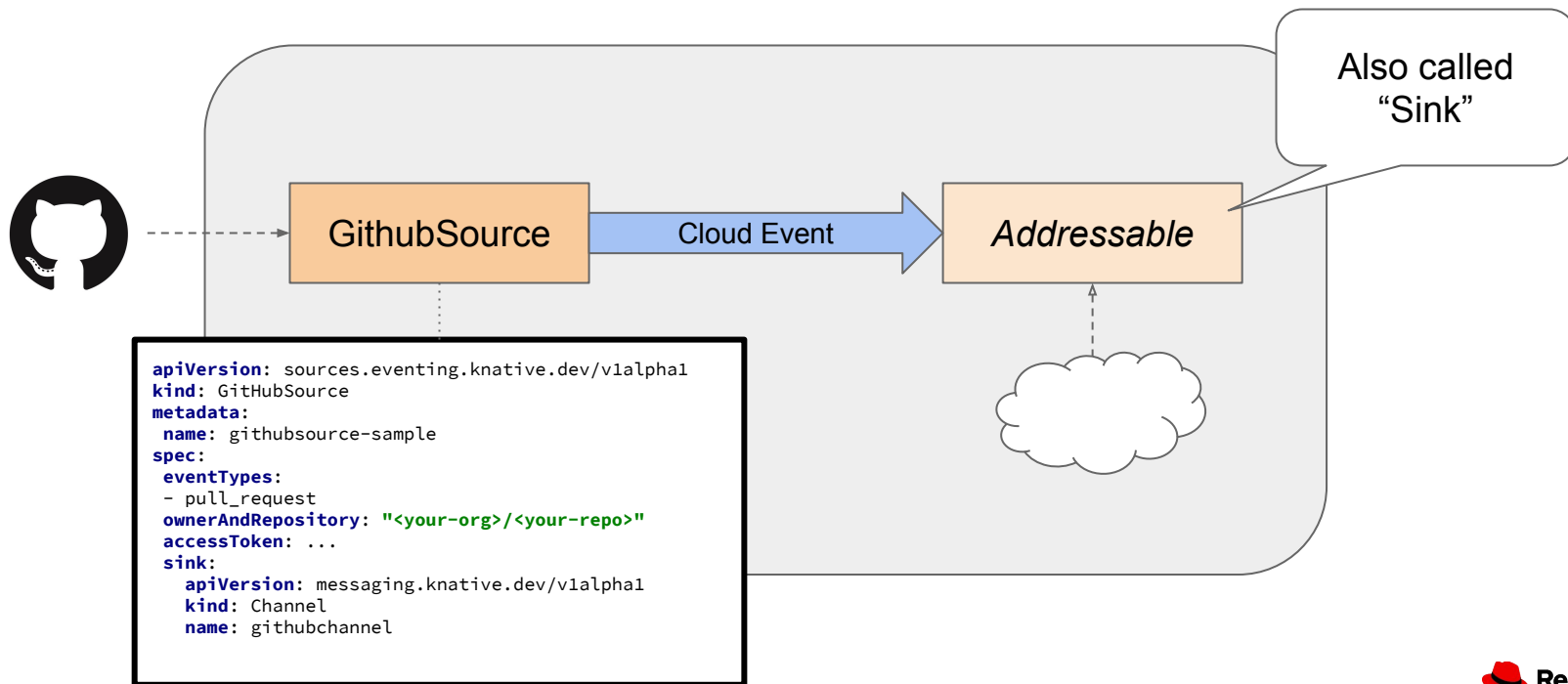
300+ components!

System 1

System 2

KNATIVE
EVENT MESH

Red Hat

Knative Eventing Sources

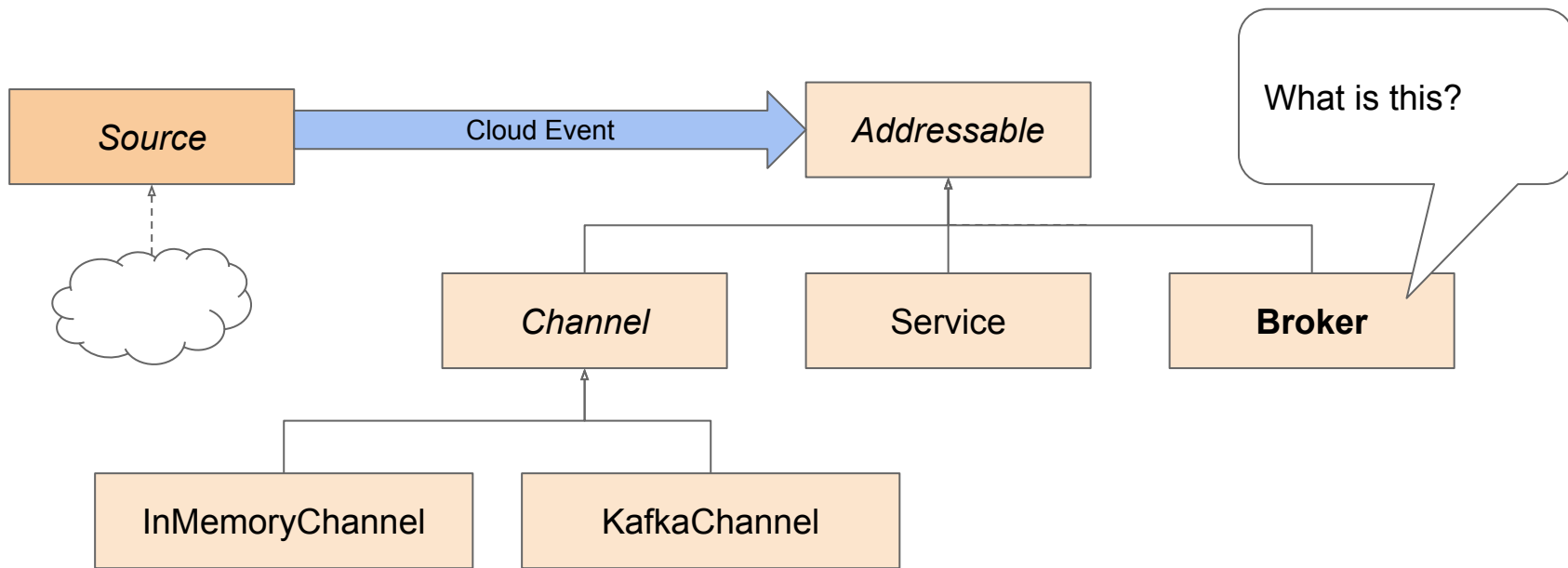A source generates or imports events and relays them as CloudEvents.

E.g.

Also called "Sink"

GithubSource → Cloud Event → *Addressable*

```
apiVersion: sources.eventing.knative.dev/v1alpha1
kind: GitHubSource
metadata:
 name: githubsource-sample
spec:
 eventTypes:
 - pull_request
 ownerAndRepository: "<your-org>/<your-repo>"
 accessToken: ...
 sink:
   apiVersion: messaging.knative.dev/v1alpha1
   kind: Channel
   name: githubchannel
```

Knative Eventing Sources

Multiple kind of sources. Extensible mechanism.

Definitely the best one ;)

Provide access to all Camel components.

Source

GithubSource   NatsSource   KafkaSource   **CamelSource**

300+ components!

Red Hat
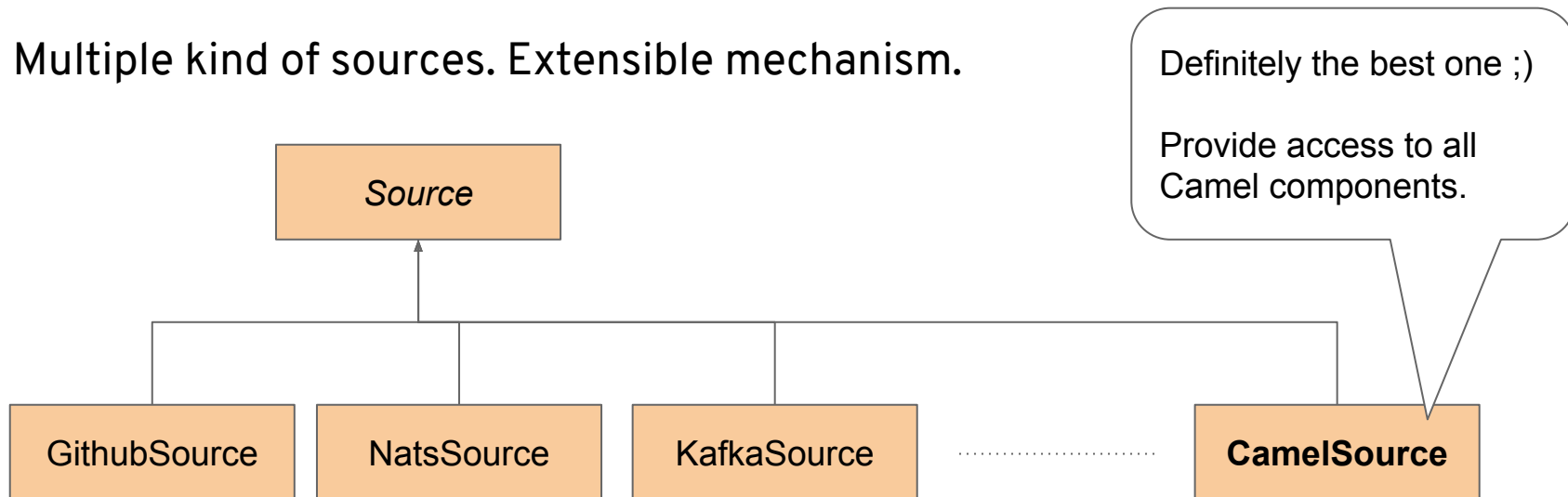
Knative: Addressable

Many objects are enabled to receive CloudEvents.

Broker centric model

Center of the event mesh

Source 1

Source 2

Source 3

Source 4

Broker

Trigger 1

Trigger 2

Trigger 3

Trigger 4

Trigger 5

Service 1

Service 2

Service 3

Still based on channels under the hood...
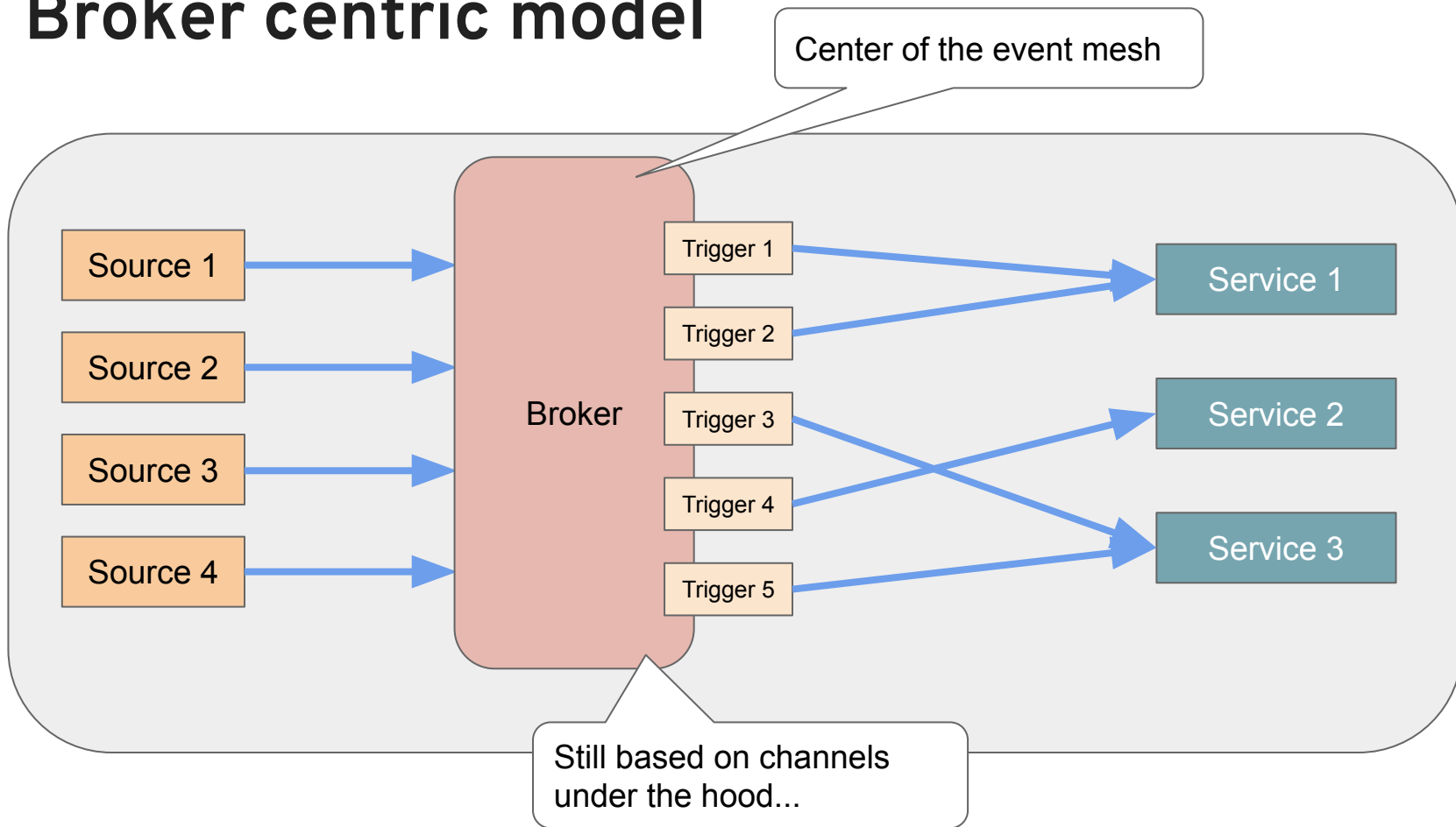
DEMO

Knative Demo

Showing Knative Serving, Eventing (Channels + Broker/Trigger model)

Code at: https://github.com/nicolaferraro/knative-demo

Topics covered:

- Creating multiple sources
- Creating services
- Injecting the broker
- Connecting services to events
- Export events to external services (Camel K)
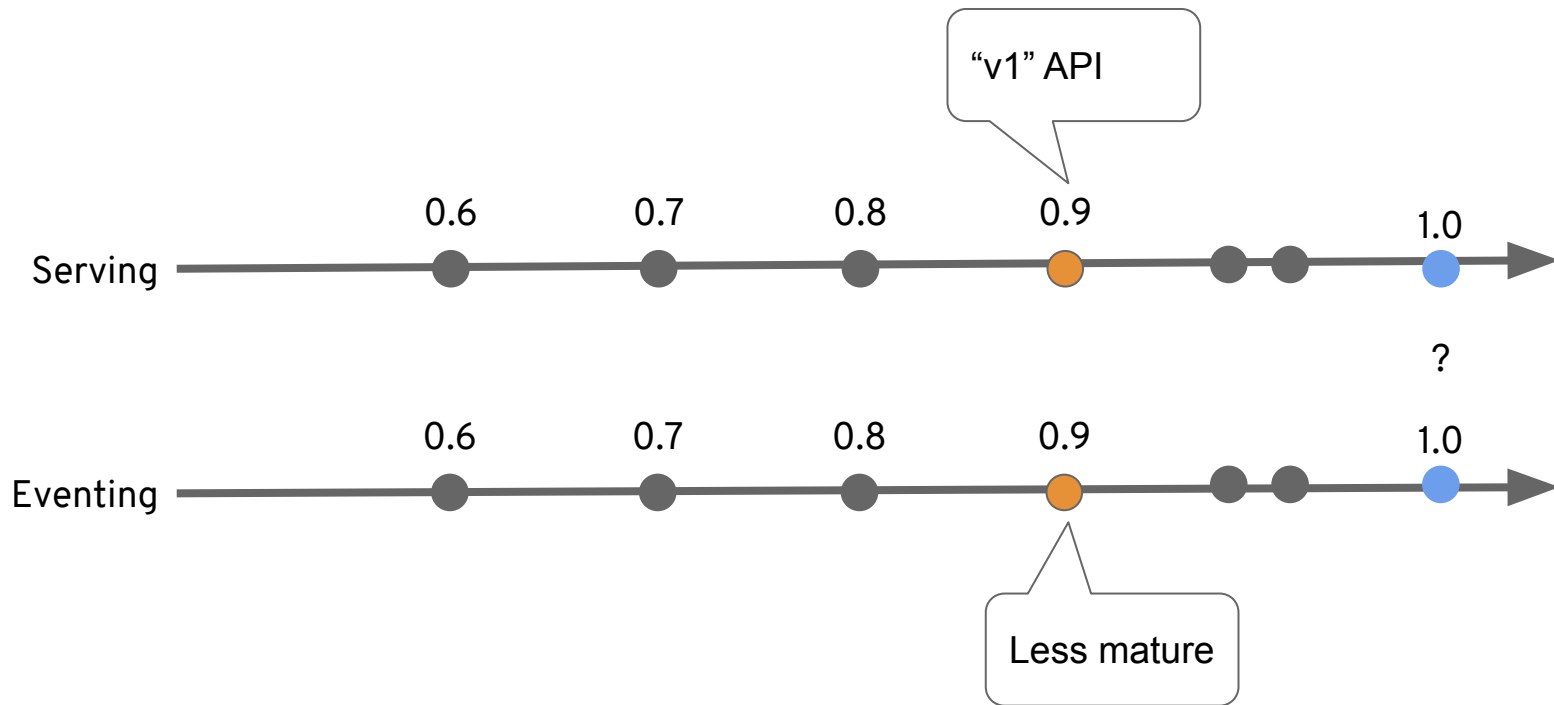
Red Hat

What is Apache Camel K?
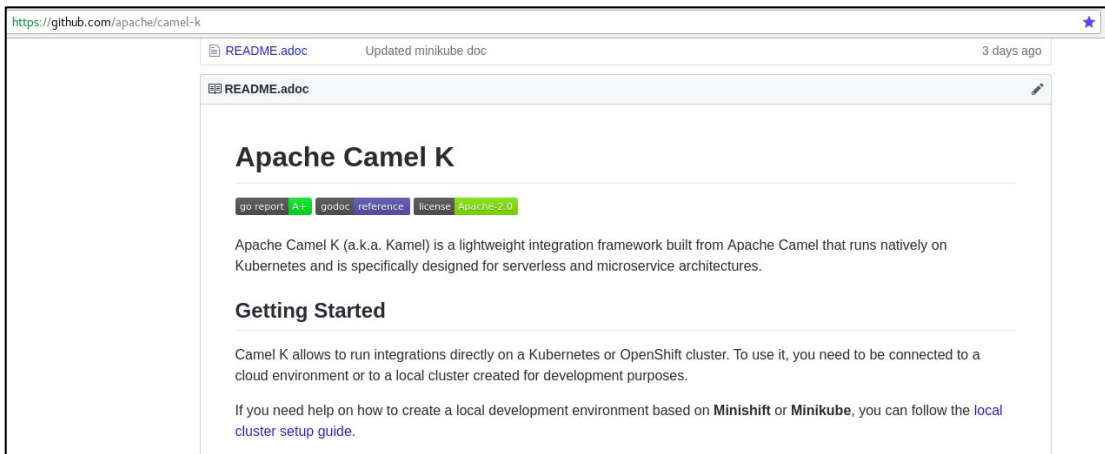
A lightweight integration platform, born on Kubernetes, with serverless superpowers

WRAP UP

Knative Status

An update on Knative foundation status

October 1, 2019

Knative community members,

Since the start of the Knative project, there have been questions about **whether Knative would be donated to a foundation, such as CNCF. Google leadership has considered this, and has decided not to donate Knative to any foundation for the foreseeable future.**

Knative continues to be an open source project with an open and diverse community, with several vendors represented on the Steering Committee. Within the next few weeks, we plan to clarify how project members can attain leadership positions in Knative. As always, our goal is to ensure that Knative serves the needs of our users, the community, and everyone who benefits from using Knative.

Donna Malayeri, on behalf of Google

Community is waiting to know "why"...

Red Hat

Thank you!

If you like Camel K:

1) Star it on github! ★ Star

2) Use it!

3) Contribute!

We love contributions!

https://github.com/apache/camel-k

Follow me on Twitter:

@ni_ferraro