APACHE CAMEL K

Bringing serverless workloads to the enterprise

Luca Burgazzoli
Principal Software Engineer

Nicola Ferraro
Senior Software Engineer

07/05/2019
Luca Burgazzoli  
Principal Software Engineer at Red Hat  
Twitter: @lburgazzoli

Nicola Ferraro  
Senior Software Engineer at Red Hat  
Twitter: @ni_ferraro

We work on:  
- Apache Camel and Camel K  
- Syndesis  
- Red Hat Fuse
Agenda

1. Brief Introduction about Apache Camel
2. Camel K
   a. Demo
3. Camel K and Knative
   a. Demo
4. What’s next
APACHE CAMEL
What is Apache Camel?

- The swiss knife of integration
- >10 years of development - still one of the most active Apache projects
- Based on Enterprise Integration Patterns (EIP)
- Uses a powerful Domain Specific Language (DSL)
- Can integrate anything
- Supports 300+ components
- The cornerstone of Red Hat Fuse

http://camel.apache.org/
System Integration

System A → ? → System B
System Integration

- Different transports
- Different Data Model
- Asynchrony
- Failures
Enterprise Integration Patterns
Enterprise Integration Patterns
Integration Framework
System Integration
Camel Routes

from("file:data/inbox")
  .to("jms:queue:order");

Java DSL

_XML DSL_

```xml
<route>
  <from uri="file:data/inbox"/>
  <to uri="jms:queue:order"/>
</route>
```
Camel Routes with Splitter
Camel Routes with Splitter

```java
from("file:inbox")
```
from("file:inbox")
  .split(body().tokenize("\n"))
Camel Routes with Splitter

from("file:inbox")
    .split(body().tokenize("\n"))
    .marshal(customToXml)
Camel Routes with Splitter

```java
from("file:inbox")
    .split(body()._tokenize("\n"))
    .marshal(customToXml)
    .to("activemq:line");
```
Runs on popular Java Runtimes
APACHE CAMEL K
What is Apache Camel K?

- A lightweight integration platform based on Apache Camel, born on Kubernetes, with serverless superpowers.
- Based on operator-sdk
- A community-driven project
- A subproject of Apache Camel started on August 31st, 2018

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

1. Create a integration file (Java, Groovy, Kotlin, JS, XML...)

```java
// Lookup every second the 'www.google.com' domain name and log the output
from('timer:dns?period=1s')
    .routeId('dns')
    .setHeader('dns.domain')
        .constant('www.google.com')
    .to('dns:ip')
    .log('log:dns');
```

2. Run it

```
$ kamel run integration.groovy
```

3. It runs on Kubernetes

Camel DSL, based on EIPs...
Architecture

Dev Environment

Cloud

“Integration” Custom Resource
Camel K Operator
Running Pod

Live updates!

Tailored for cloud-native development experience

Fast redeploy!
Less than 1 second!
CAMEL K
PERFORMANCE
Performance

The operator understands the code so it can:

- can choose the lightest runtime for the integration (e.g. Quarkus)
- can package a minimal set of components that are required by the code
- can fine tune Camel and JVM parameters based on user the code
- can optimize creation of container images

Compared to a traditional spring-boot stack

- it requires less memory and cpu
- does not require to generate a uber jar thus improve deployment speed
A web-based integration platform (Syndesis).

Target:
- Citizen Integrators

Features:
- Multiple connectors built from Camel components
- Few clicks to define a integration
- Graphical data mapping capabilities
- Design, expose or consume REST API
- Integrated with Apicur.io for API design
- Integrated with 3-scale for API management
- Now works with Camel K as runtime engine for integrations!
Time to run an integration in Syndesis

... compared to the traditional spring-boot backend

Lower is better :)}
DEMO
Let’s build a Telegram Chat Bot in few minutes.

You’ll need **shortly on your phone:**
- Telegram App
- A QR Code reader
Camel K Demo

https://youtu.be/-O-6dwqxH1E
Let’s build a Telegram Chat Bot in few minutes.

With circuit breakers and retries!

Code: [https://github.com/nicolaferraro/camel-k-chuck-bot](https://github.com/nicolaferraro/camel-k-chuck-bot)

@camelkbot
CAMEL K & KNATIVE
Knative defines **building blocks** for “Serverless” applications ([https://github.com/knative/](https://github.com/knative/)). A **building block** is a CRD with a controller (or “operator”) that manages its lifecycle.

Knative Build

- <<custom-resource>>
  - Build
  - ...  

**Standardize building container images**

Knative Serving

- <<custom-resource>>
  - Service
  - ...  

**Auto-scaling and scale-to-zero**

Knative Eventing

- <<custom-resource>>
  - Channel
  - ...  

**Messaging for event-based applications**
from("knative:channel/a")
  .to("http:my-host/api/path");
What does it mean?

There’s no container if no one needs it!

```
rest().post("/path")
 .to("xx:system1")
 .to("xx:system2")
```
What does it mean?

A container is created only when needed!

rest().post("/path")
  .to("xx:system1")
  .to("xx:system2")
What does it mean?

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

300+ components!

Kubernetes Namespace

Knative Service

Multiple containers under high load!
Knative Eventing

Building blocks for event-based serverless applications.

Producers

Kafka, In-memory, ...

Channel

https://cloudevents.io/

Consumers

Subscription

Mmh? Isn’t that like JMS?
Knative Eventing

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

300+ components!

https://cloudevents.io/
Camel K in Knative Eventing

Same model for different purposes

300+ components!
Knative Demo

https://youtu.be/btf_e2GniXM
KNATIVE CAMEL SOURCES
Knative Camel Sources

Standard Knative Sources that can be used without any knowledge of Camel K!

... but Camel K is under the hood ;)

https://github.com/knative/eventing-sources/releases
Knative Camel Sources

Coming soon...

Camel components.. “Cloud Edition”

CamelTwitterSource

CamelFacebookSource

CamelTelegramSource

CamelServiceNowSource

300+ components!
A Dev writes Camel K code.

```
from("timer:tick?period={{period}}")
 .to("xx:my-system")
 .unmarshal().json()
 .transform("custom")
 .to("knative:endpoint/sink")
```

Users can instantiate the source to do their job.
RECAP AND WHAT’S NEXT...
What is Camel K?

A lightweight integration platform, born on Kubernetes, with serverless superpowers.

1. Runs on “vanilla” Kubernetes (1),
2. Openshift (2) and gives its best on a Knative-powered cluster (3)!
What’s next?

Knative features:
● Camel Components as Knative Sources
● Use Camel K to build your own Knative Source

Most notable features:
● Support for Camel 3 (Now M2, GA in September 2019)
  ○ Lightweight modules to reduce footprint
  ○ Webhook meta-component (more components that can scale down to 0)
● Integration with Operator Hub
● Full integrated with Syndesis
● Better IDE support
● Better integration with CI tools
● Testing
● Camel K UI
What can you do next?

If you like Camel K:

1) Star it on github!
2) Use it!
3) Contribute!

We love contributions!

Follow us on Twitter:
@ni_ferraro
@lburgazzoli

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

plus.google.com/+RedHat
linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/redhat