Camel riders in the cloud

Red Hat DevNation Live
March 2018

About me

- Senior Principal Software Engineer at Red Hat
- 9 years as full time Apache Camel committer
- Author of Camel in Action books
- Based in Denmark, Europe

Blog: http://www.davsclaus.com
Twitter: @davsclaus
Linkedin: davsclaus
E-Mail: cibsen@redhat.com

System Integration

Figure 1.1 Camel is the glue between disparate systems.

Integration Framework

PATTERN BASED INTEGRATION

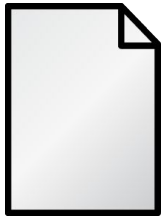
Apache Camel, a powerful pattern-based integration engine with a comprehensive set of connectors and data formats to tackle any integration problem.

ENTERPRISE INTEGRATION PATTERNS

Build integrations using enterprise best practices.

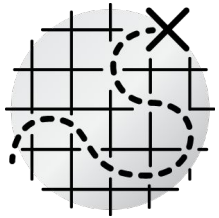
200+ COMPONENTS

Batch, messaging, web services, cloud, APIs, and more ...



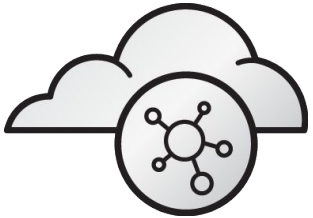
BUILT-IN DATA TRANSFORMATION

JSON, XML, HL7, YAML, SOAP, Java, CSV, and more ...



INTUITIVE ROUTING

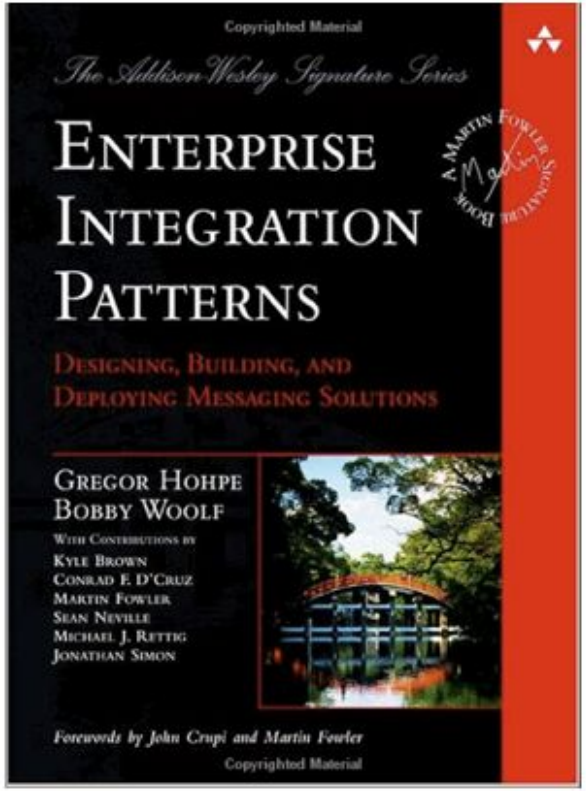
Develop integrations quickly in Java or XML.



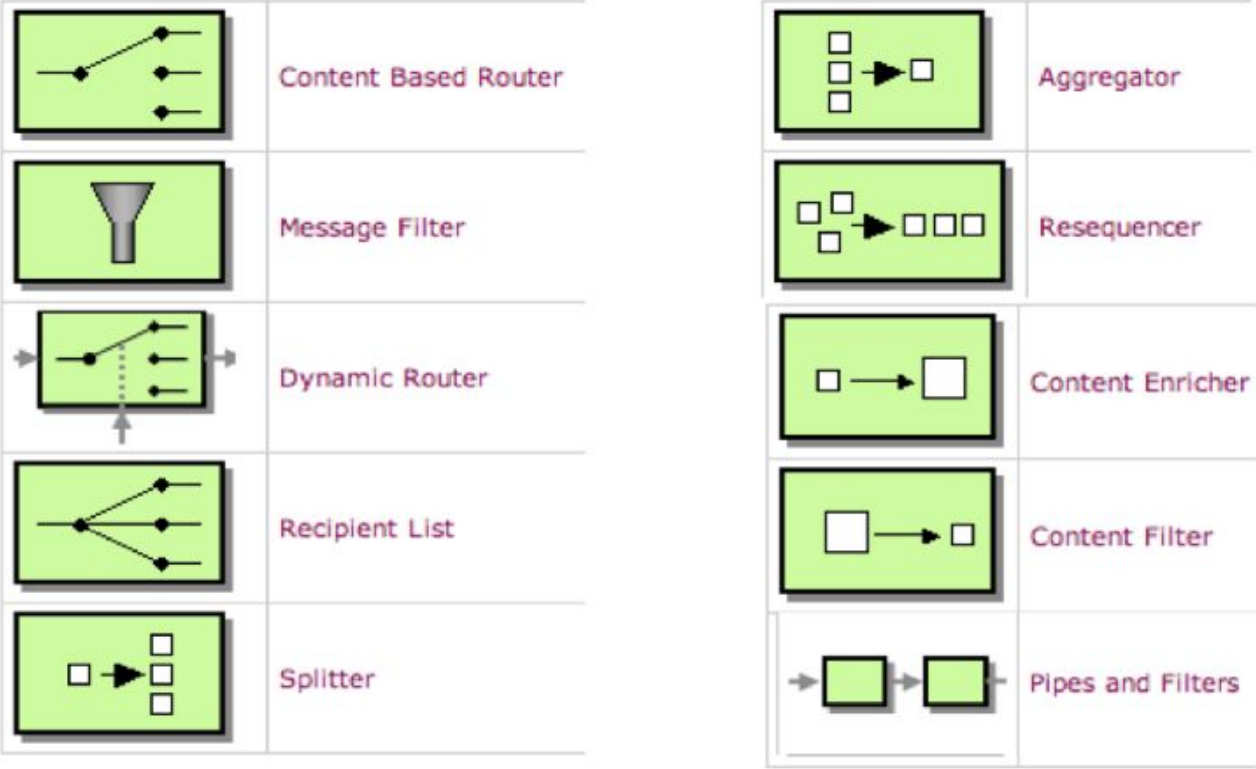
NATIVE REST SUPPORT

Create, connect, and compose APIs with ease.

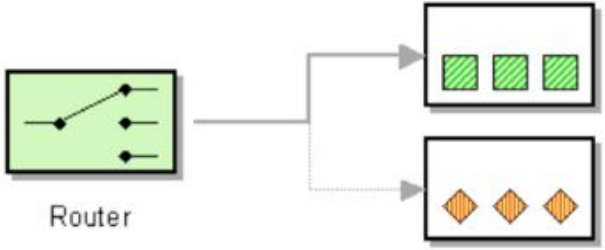
Enterprise Integration Patterns



Enterprise Integration Patterns



Camel Routes




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

Java DSL

XML DSL

```
<route>
  <from uri="file:data/inbox"/>
  <to uri="jms:queue:order"/>
</route>
```

Camel Architecture

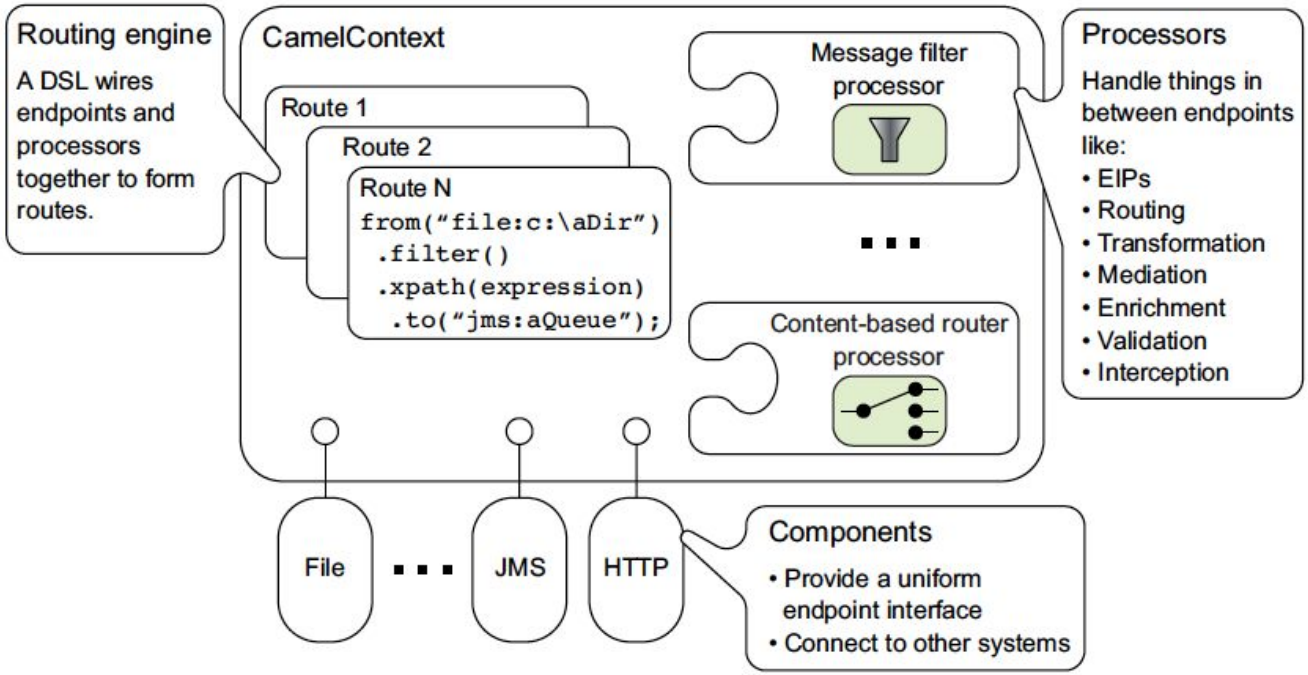
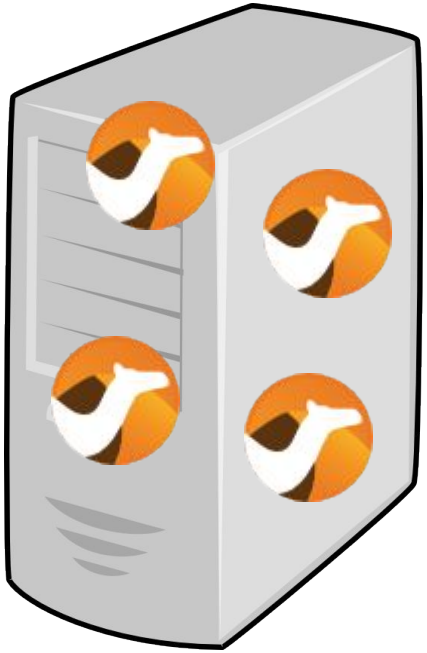


Figure 1.8 At a high level, Camel is composed of routes, processors, and components. All of these are contained within `CamelContext`.

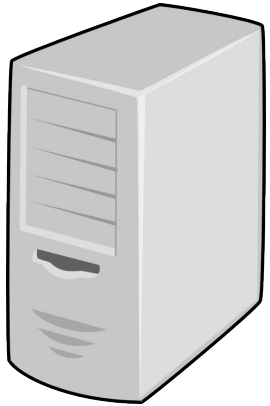
Camel runs everywhere



Application Servers

Linux Containers

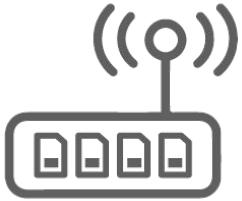
Camel connects everything



Legacy Systems

- File
- FTP
- JMS
- JDBC
- SQL
- TCP/UDP
- Mail
- HDFS
- JPA
- MongoDB
- ...

- AWS
  - S3
  - SQS
  - Kinesis
  - ...
- Google
  - Bigquery
  - Pubsub
- Azure
  - Blob
  - Queue

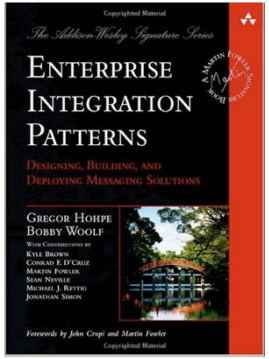


IoT

- CoAP
- MQTT
- PubNub

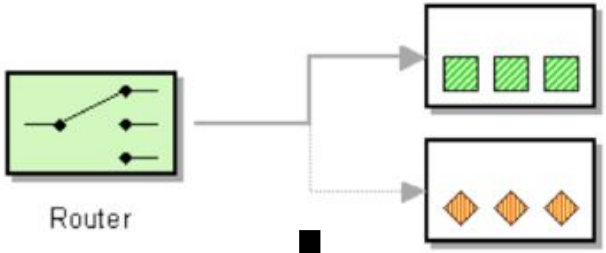
- Box / Dropbox
- Facebook
- Linkedin
- Salesforce
- SAP
- ServiceNow


TOOL BOX

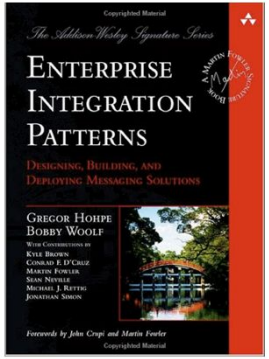

Copyrighted Material
The Addison-Wesley Signature Series
Enterprise Integration Patterns
Designing, Building, and Deploying Messaging Solutions
Gregor Hohpe
Bobby Woolf
With Contributions by
Kyle Brown
Conrad F. D'Cruz
Martin Fowler
Sean Neville
Michael J. Rettig
Jonathan Simon
Forewords by John Crupi and Martin Fowler
Copyrighted Material

+


TOOL BOX

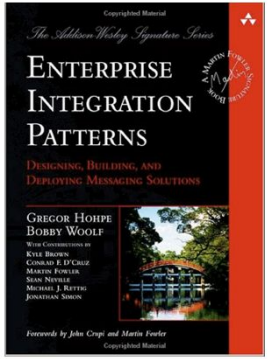

Router

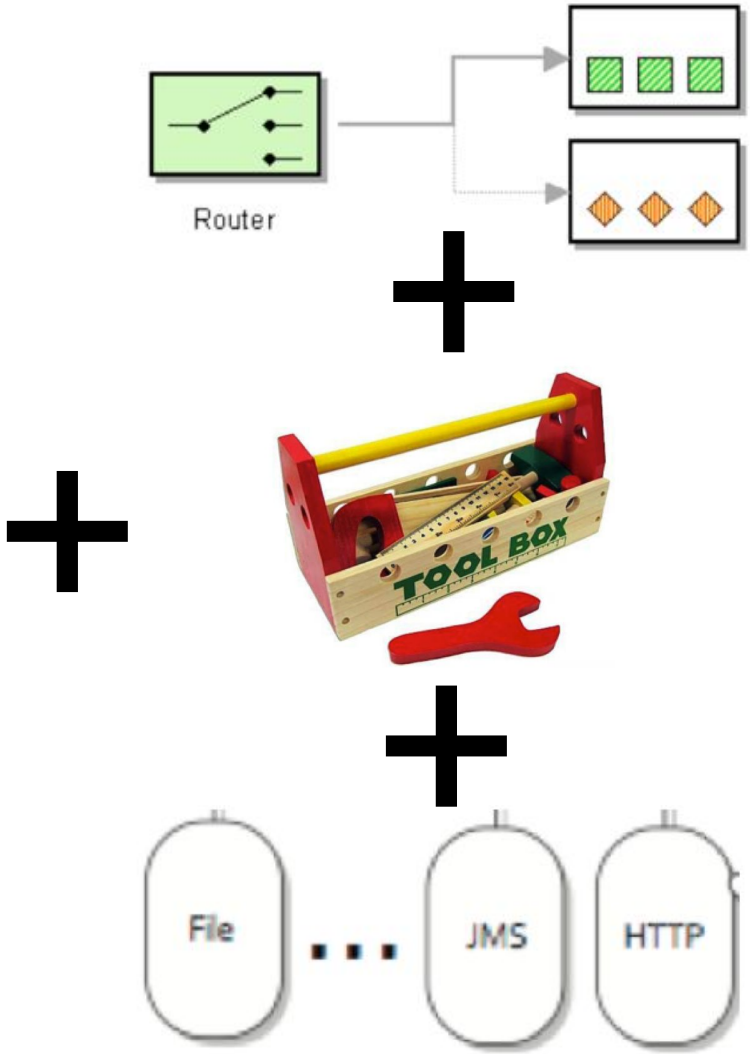
+

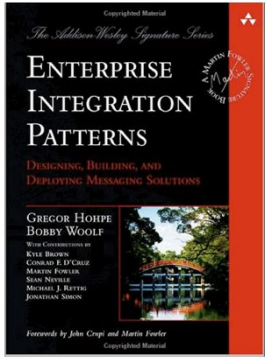

Copyrighted Material
The Addison-Wesley Signature Series
Enterprise Integration Patterns
Designing, Building, and Deploying Messaging Solutions
Gregor Hohpe
Bobby Woolf
With Contributions by
Kyle Brown
Conrad F. D'Cruz
Martin Fowler
Sean Neville
Michael J. Rettig
Jonathan Simon
Forewords by John Crupi and Martin Fowler
Copyrighted Material

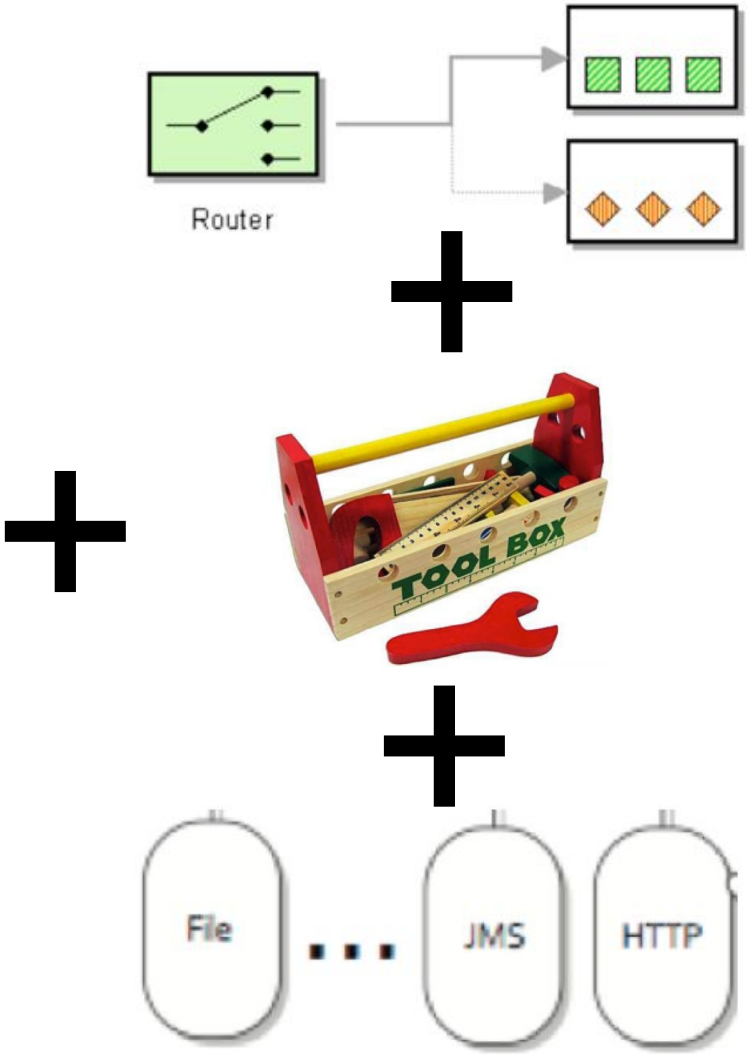
+


TOOL BOX


Copyrighted Material
The Addison-Wesley Signature Series
Enterprise Integration Patterns
Designing, Building, and Deploying Messaging Solutions
Gregor Hohpe
Bobby Woolf
With Contributions by
Kyle Brown
Conrad F. D'Cruz
Martin Fowler
Sean Neville
Michael J. Rettig
Jonathan Simon
Forewords by John Crupi and Martin Fowler
Copyrighted Material

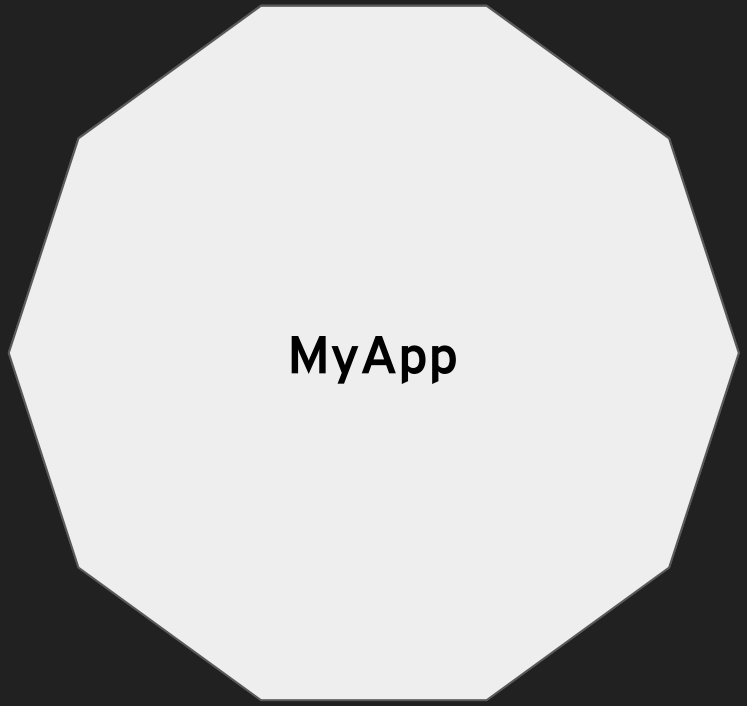

Router
TOOL BOX
File
JMS
HTTP


Copyrighted Material
The Addison-Wesley Signature Series
Enterprise Integration Patterns
Designing, Building, and Deploying Messaging Solutions
Gregor Hohpe
Bobby Woolf
With Contributions by
Kyle Brown
Conrad F. D'Cruz
Martin Fowler
Sean Neville
Michael J. Rettig
Jonathan Simon
Forewords by John Crupi and Martin Fowler
Copyrighted Material

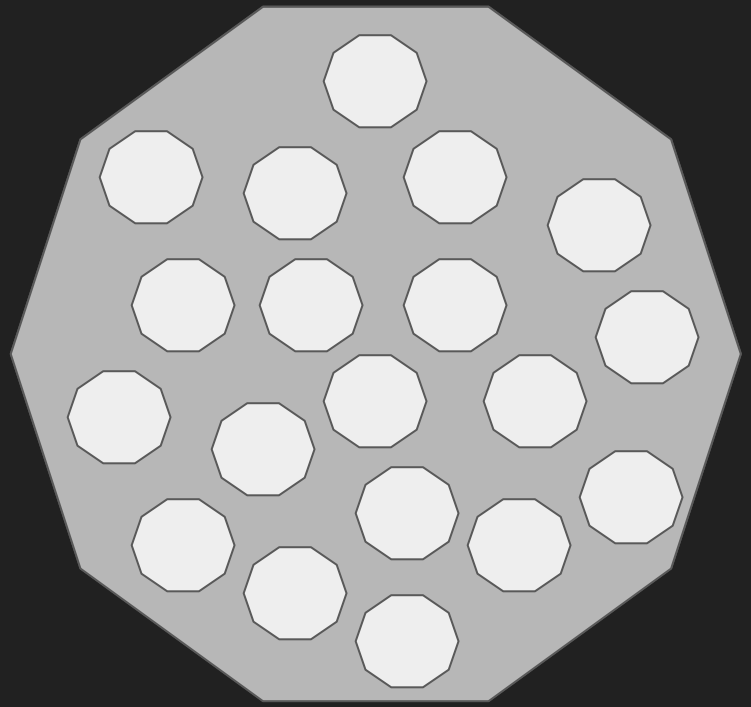

Router
TOOL BOX
File
JMS
HTTP

What about Camel in the Cloud?

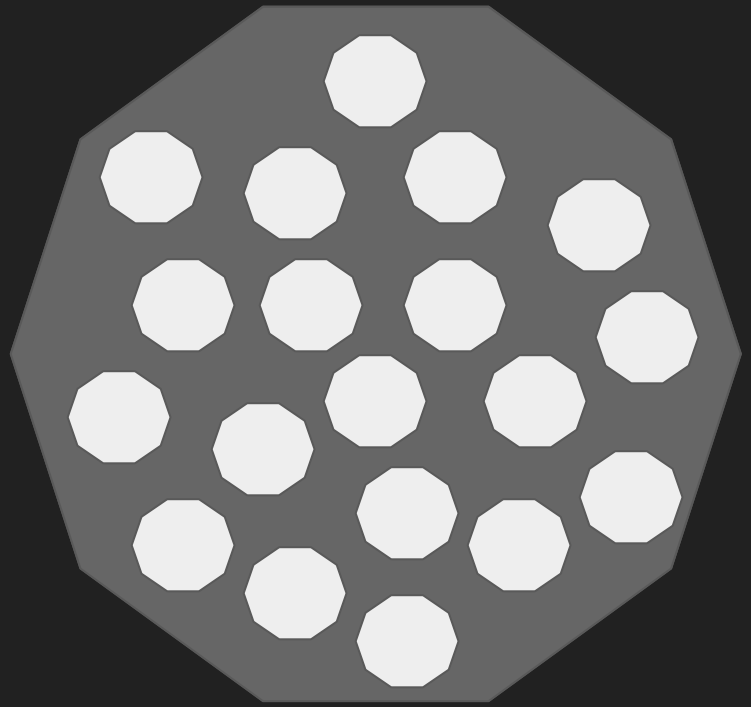
Monolith



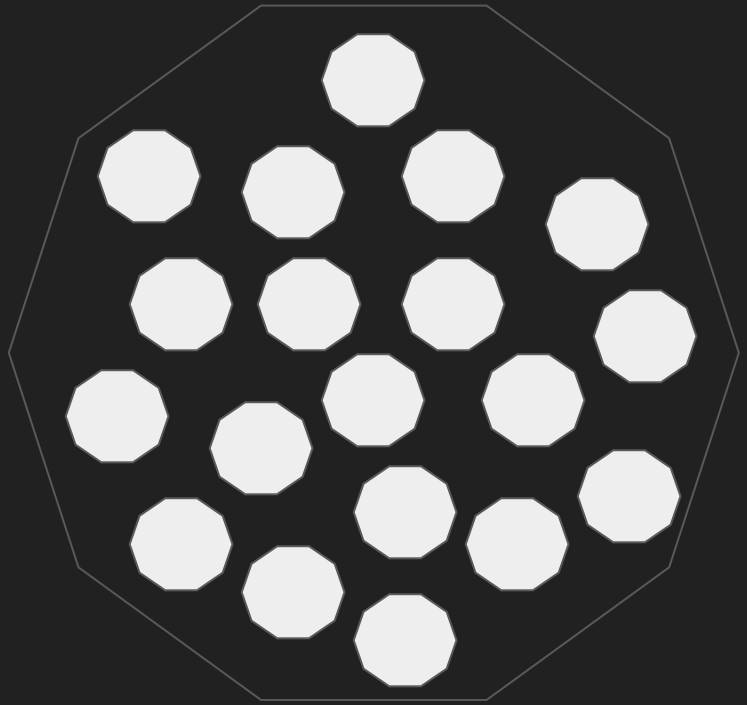
Microservices



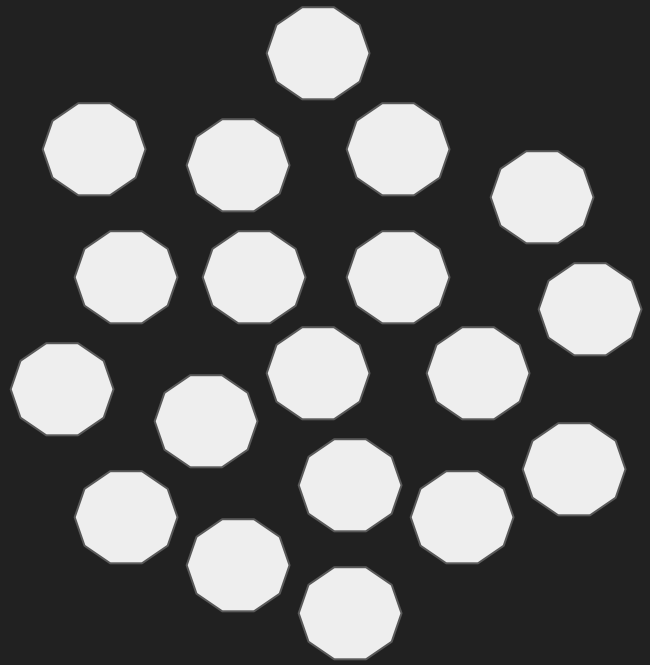
Microservices



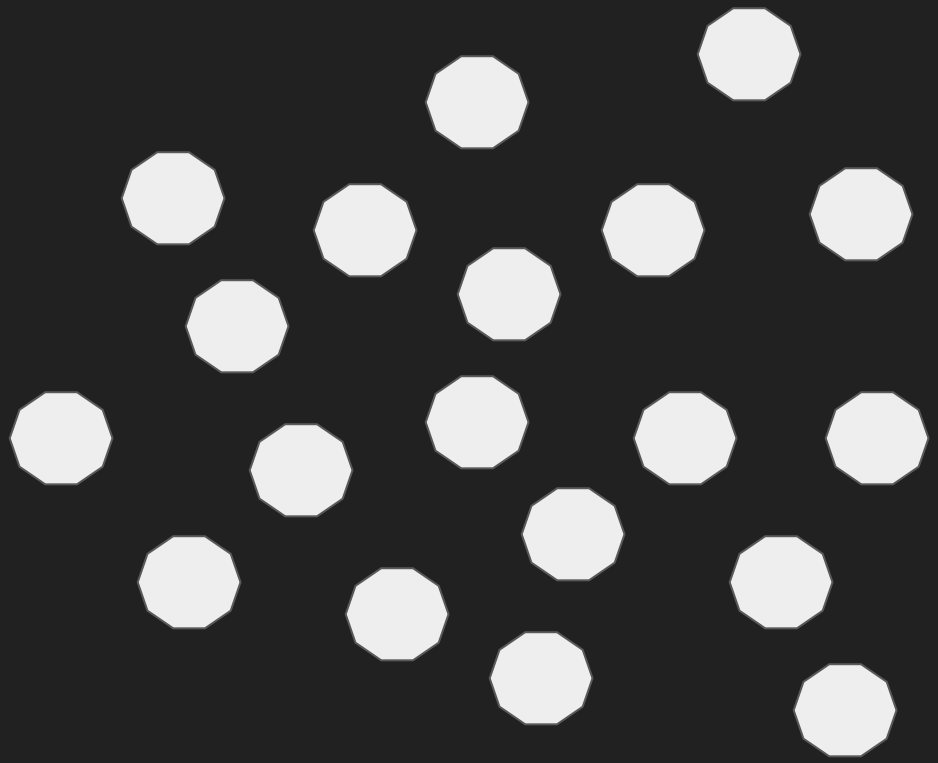
Microservices



Microservices

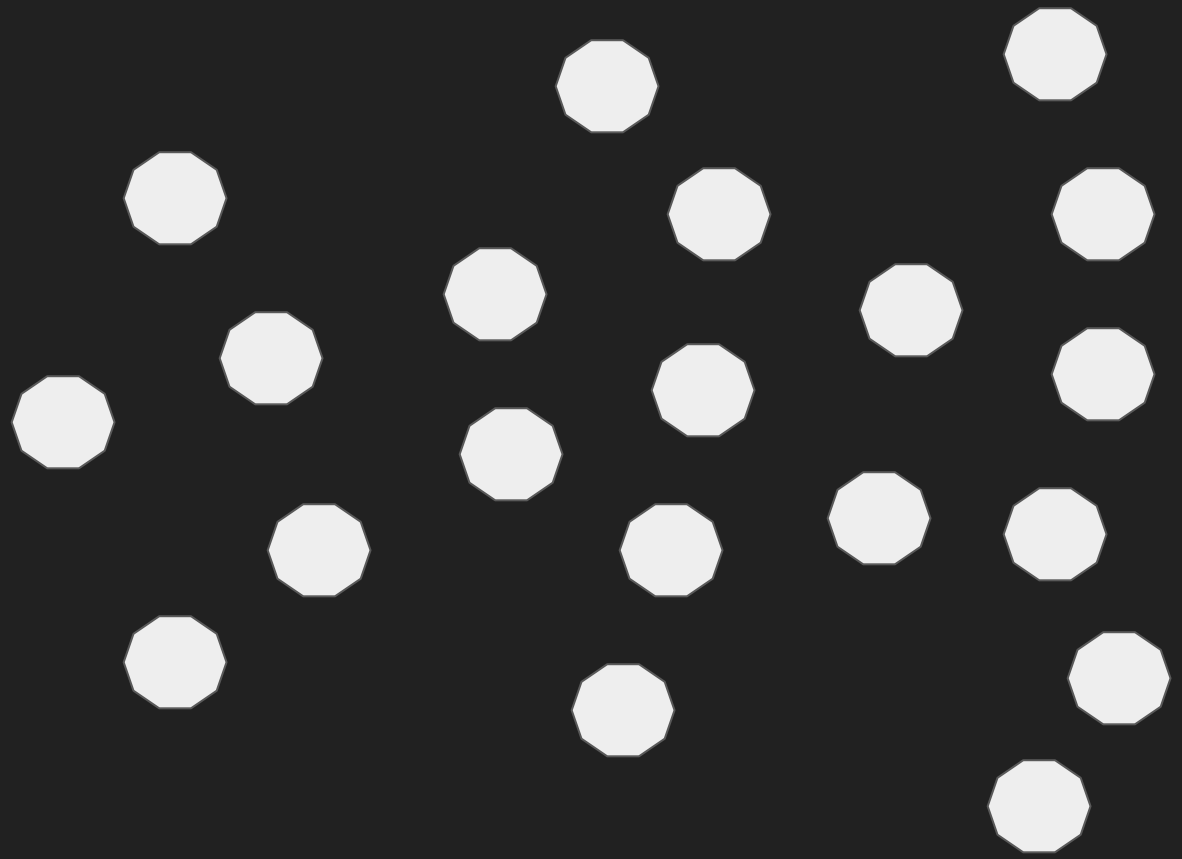


Microservices



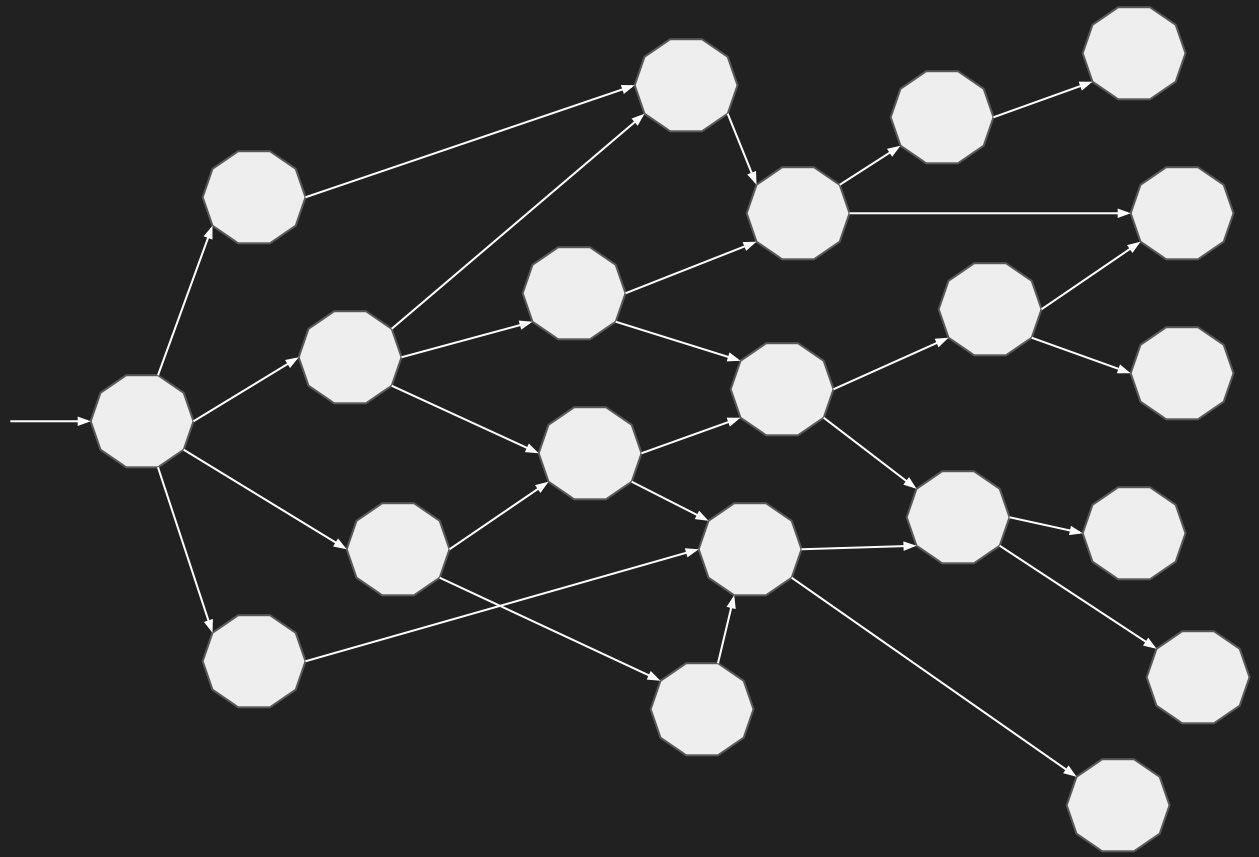
@burrsutter

Microservices



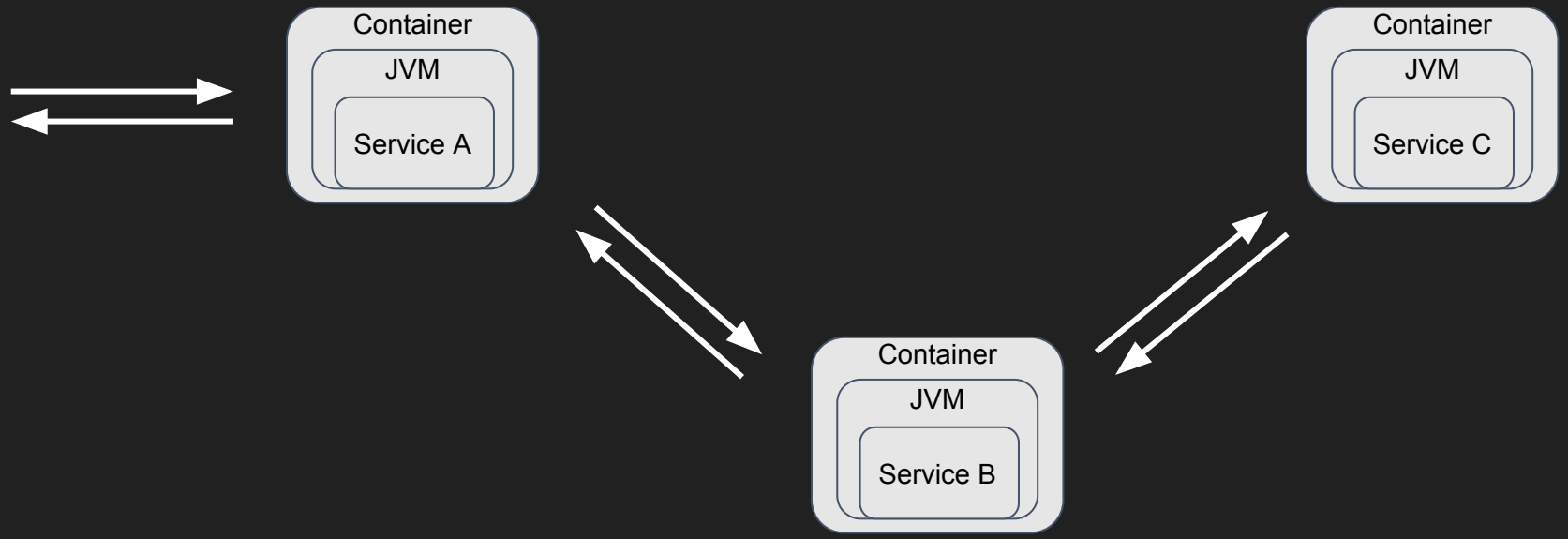
@burrsutter

Network of Services



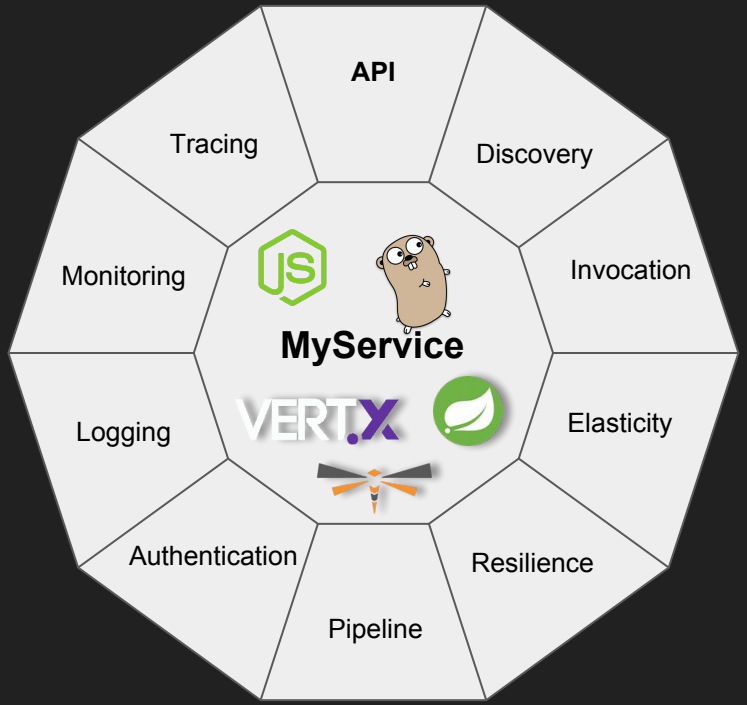
@burrsutter

Microservices == Distributed Computing



@burrsutter

Microservices'ilities

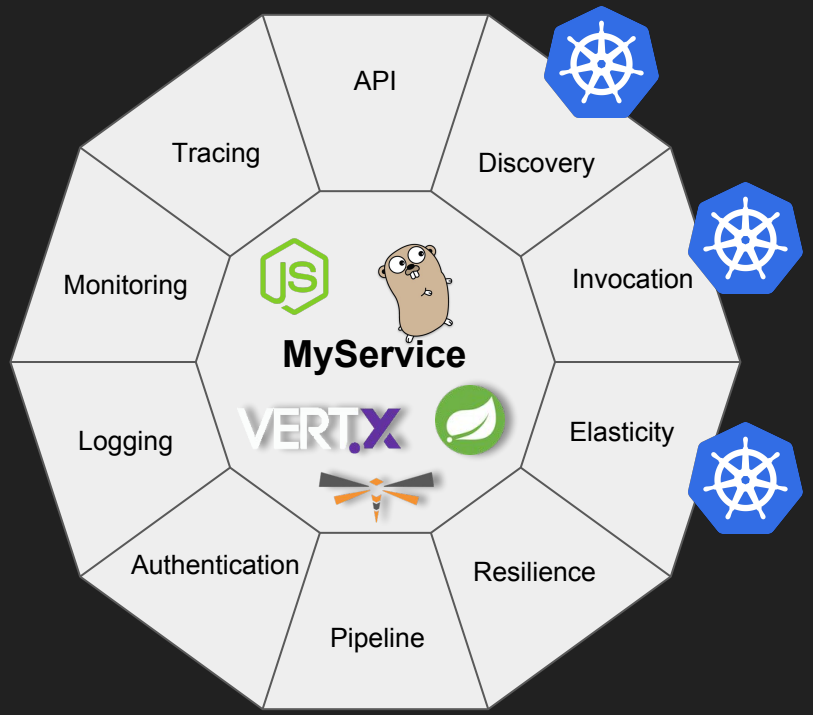


@burrsutter

OPENSHIFT

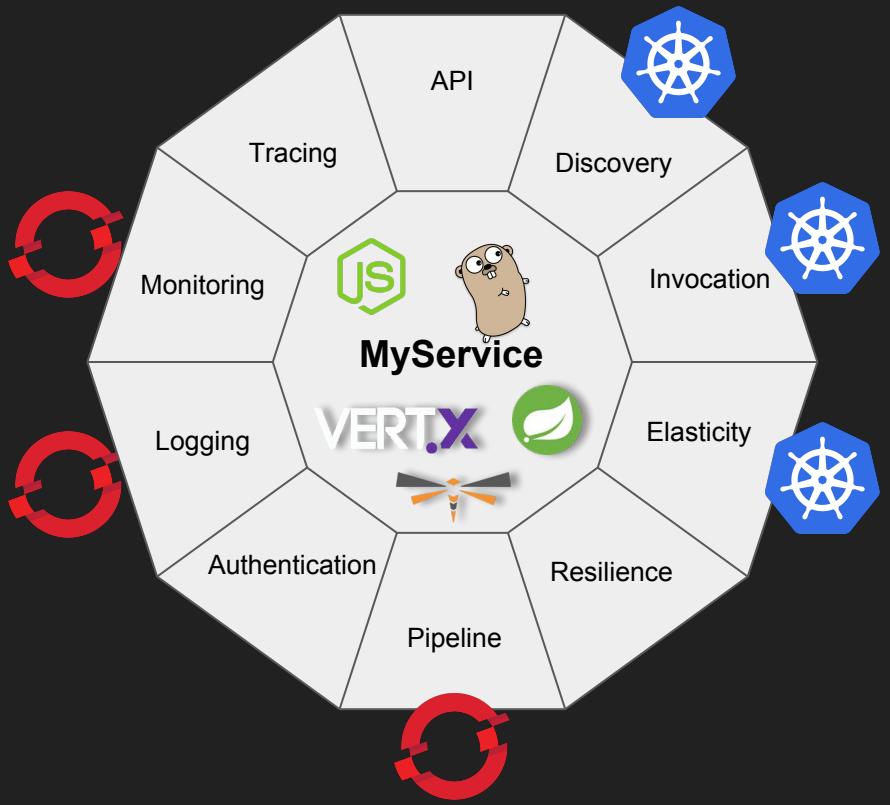
@burrsutter

Microservices'ilities + Kubernetes



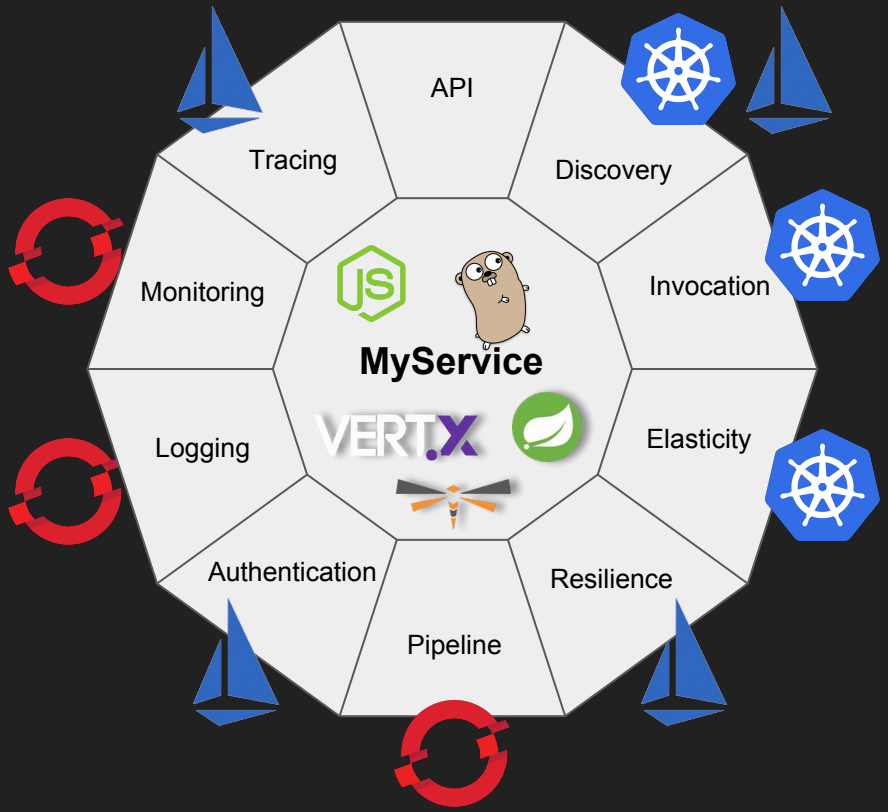
@burrsutter

Microservices'ilities + OpenShift



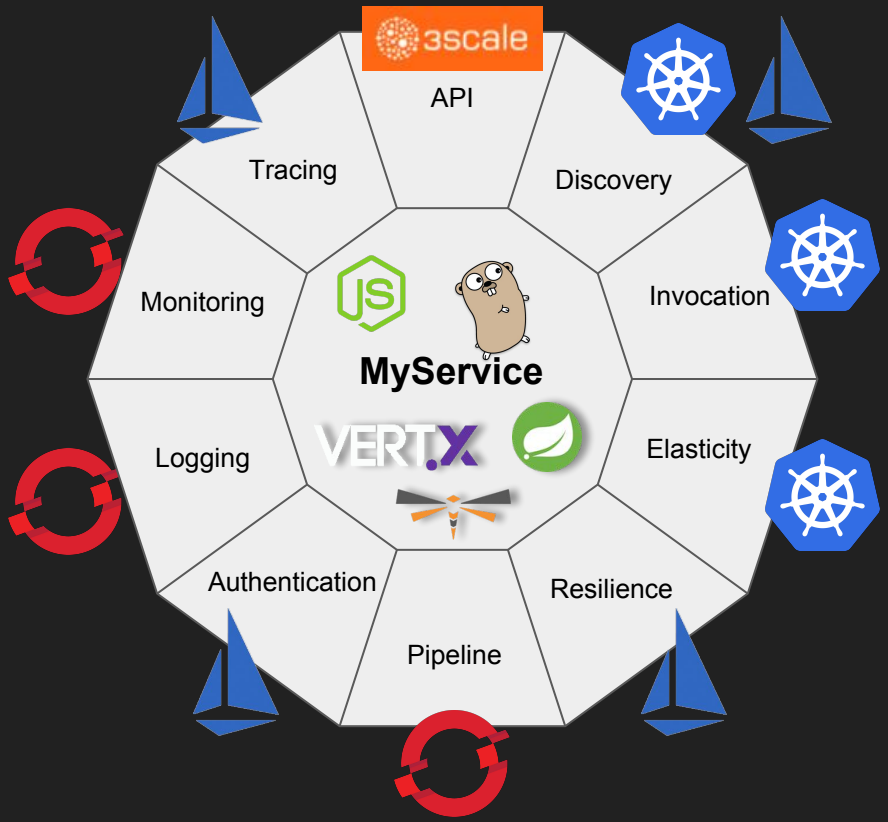
@burrsutter

Microservices'ilities + Istio



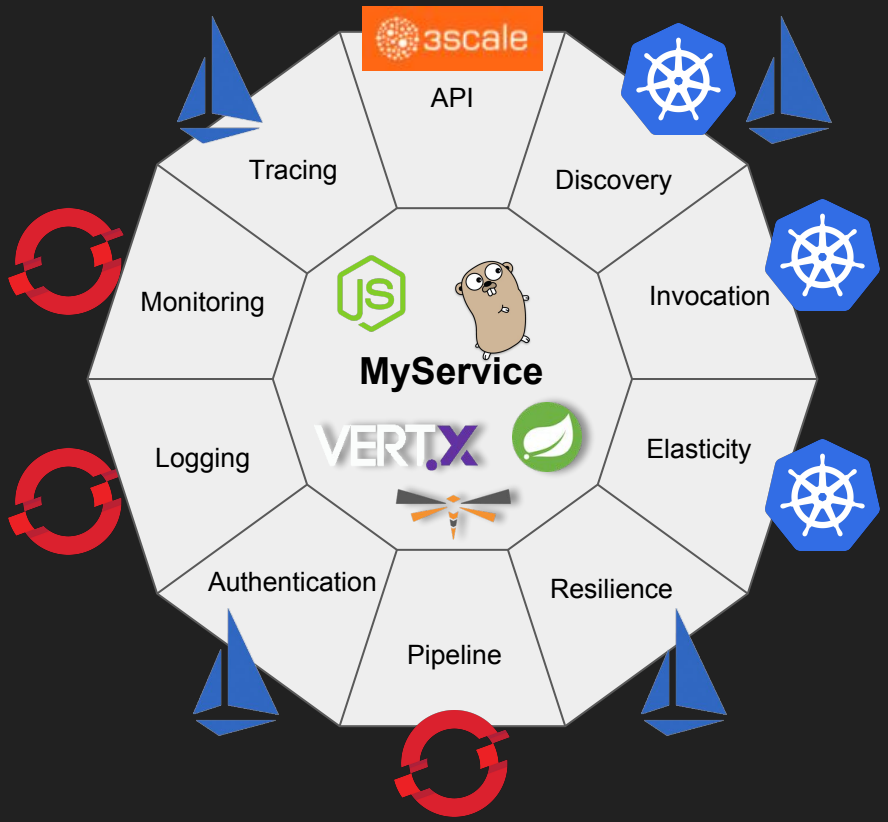
@burrsutter

Microservices'ilities + 3scale



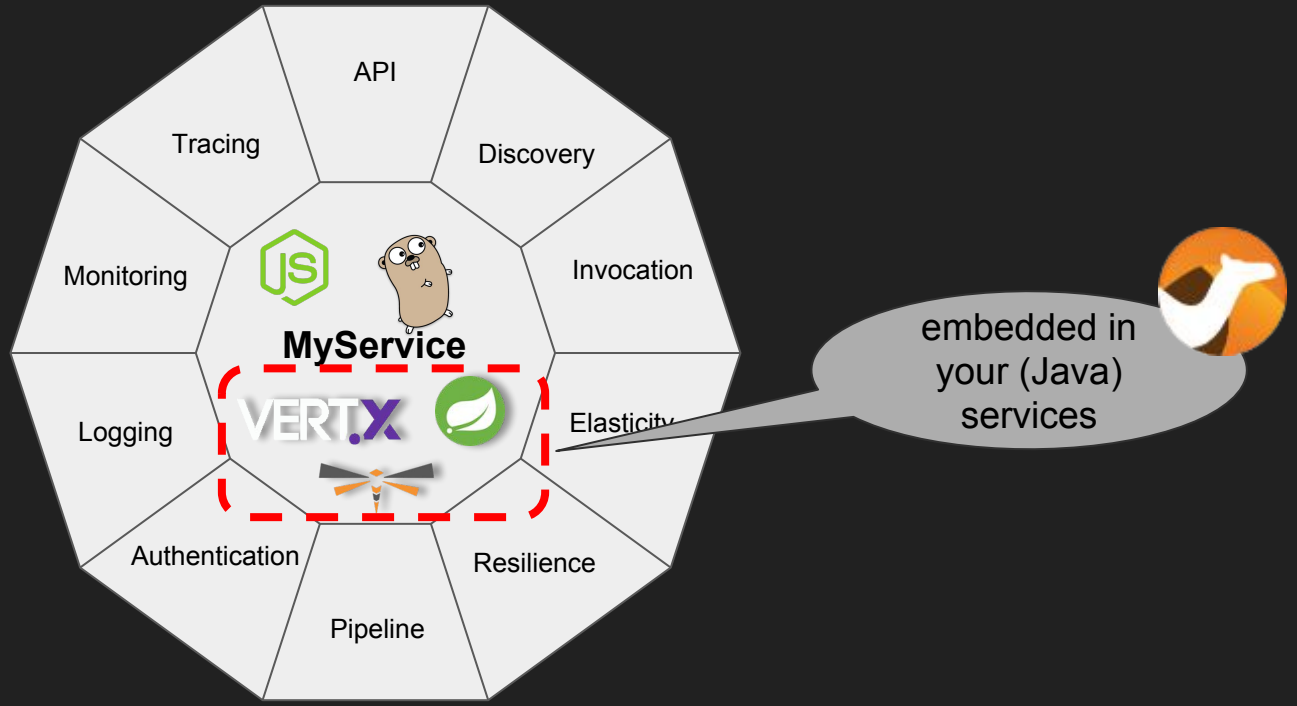
@burrsutter

But where is Camel?

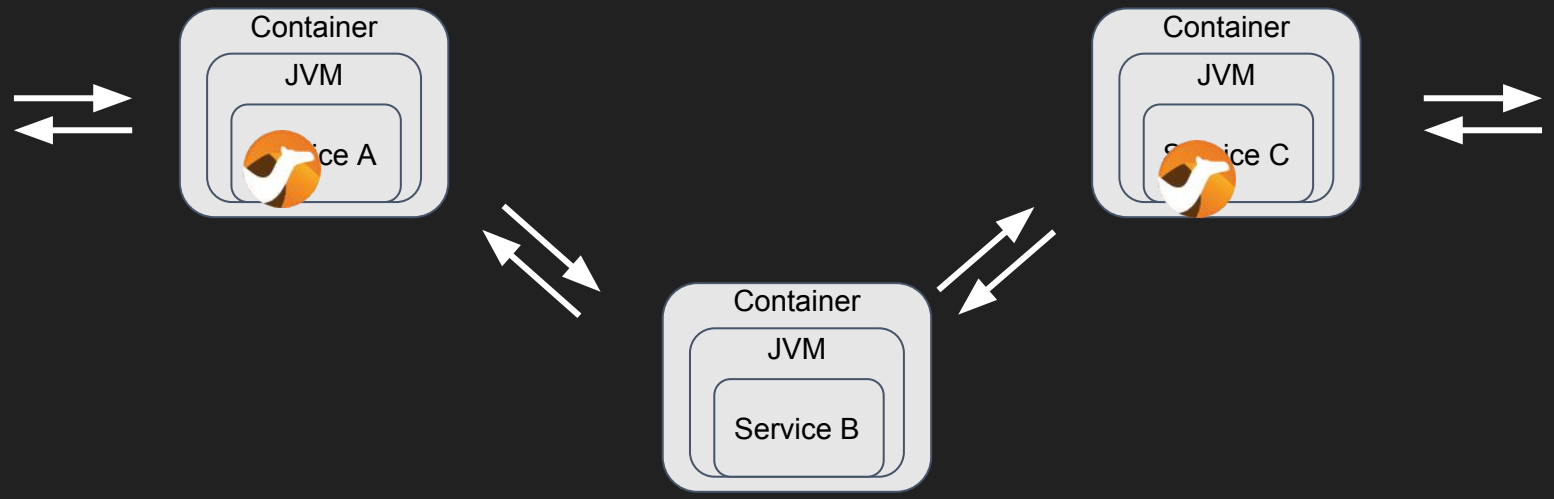


@burrsutter

But where is Camel?

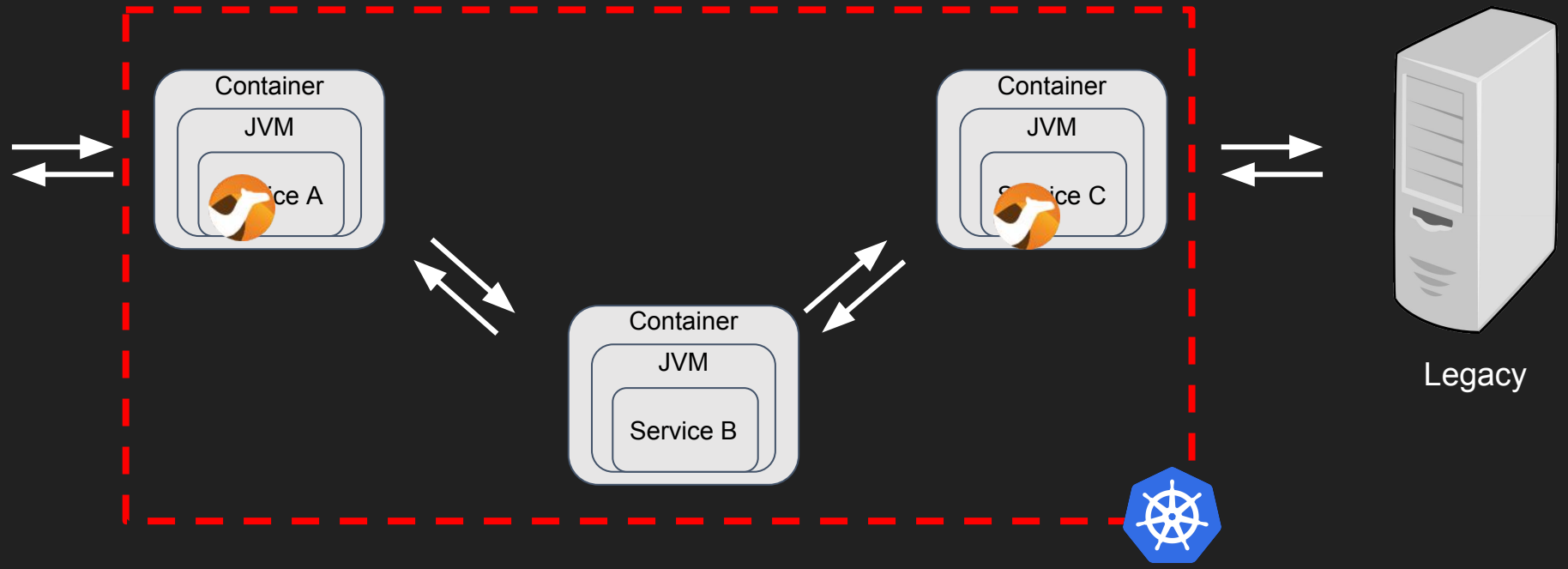


Microservices == Distributed Integration



@burrsutter

Microservices == Distributed Integration



@burrsutter

THE THREE PILLARS OF AGILE INTEGRATION

Key foundational capabilities needed by today's enterprises

DISTRIBUTED INTEGRATION

-LIGHTWEIGHT
-PATTERN BASED
-EVENT ORIENTED
-COMMUNITY SOURCED

FLEXIBILITY

CONTAINERS

- CLOUD NATIVE SOLUTIONS
-LEARN ARTIFACTS, INDIVIDUALLY DEPLOYABLE
- CONTAINER BASED SCALING AND HIGH AVAILABILITY

SCALABILITY

kubernetes

APIs

-WELL DEFINED, REUSABLE, AND WELL MANAGED END-POINTS
-ECOSYSTEM LEVERAGE

RE-USABILITY

3scale

Camel in the cloud

Best Practice - Small in Size

- Camel is light-weight
  - (camel-core 4mb)
- Add only Camel component you need
- Single fat-jar via
  - Spring Boot
  - WildFly-Swarm
  - Vert.X
  - etc.

Best Practice - Stateless

- Favour stateless applications

- If state is needed:
  - Data-grid
    - camel-infinispan
    - camel-hazelcast
    - camel-ignite
    - ...
  - Database
    - camel-sql
    - camel-jpa
    - ...
  - Kubernetes Stateful-set

Best Practice - Configuration Management

- Kubernetes ConfigMap
  - Inject via ENV
  - Inject via files
- Kubernetes Secrets
  - Inject via ENV
  - Inject via files
  - Inject via files on classpath

```
// inject configuration via spring-style @Value
@Value("${fallback}")
private String fallback;
```

```
.simple( text: "${sysenv.FALLBACK}")
```

```
$ oc get cm -o yaml my-configmap
apiVersion: v1
data:
  fallback: I still got no response
kind: ConfigMap
```

Best Practice - Fault Tolerant

- Camel Retry
  - onException
  - errorHandler

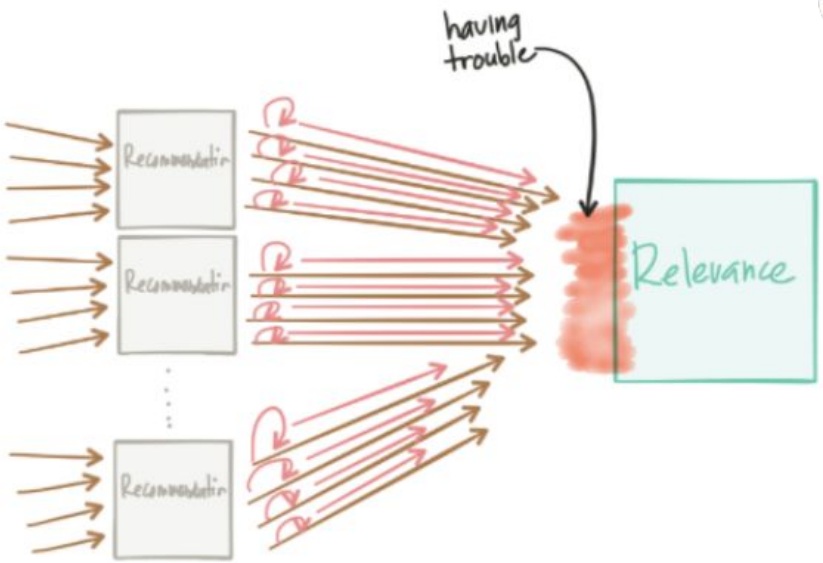
```
onException(Exception.class)
    .maximumRedeliveries(10)
    .redeliveryDelay(1000);
```

- Camel Hystrix
  - circuit breaker

Best Practice - Fault Tolerant

- Camel Retry
  - onException
  - errorHandler

```
onException(Exception.class)
    .maximumRedeliveries(10)
    .redeliveryDelay(1000);
```



Thundering Herd Problem

Figure by Christian Posta

Best Practice - Health Checks

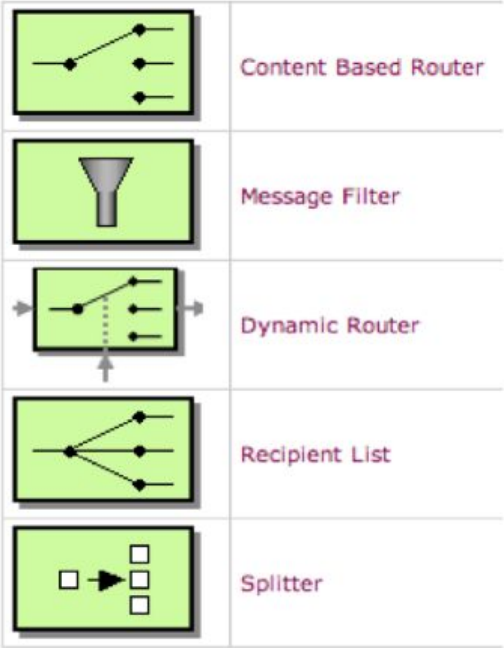
- Health Checks
  - camel-spring-boot actuator
  - wildfly-swarm monitor
- Readiness Probe
  - Kubernetes
- Liveness Probe
  - Kubernetes

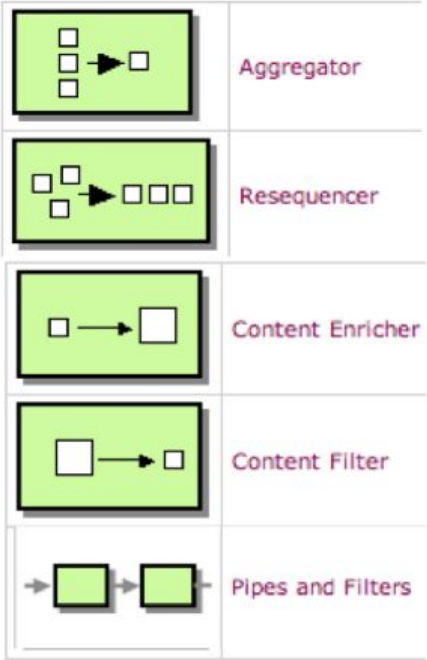
client-hystrix-myproject.192.168.64.4.nip.io/health


```
{
    status: "UP",
  - camel: {
        status: "UP",
        name: "camel-1",
        version: "2.20.2",
        contextStatus: "Started",
    },
  - camel-health-checks: {
        status: "UP",
        route:route1: "UP",
    },
  - diskSpace: {
        status: "UP",
        total: 19195224064,
        free: 5747757056,
        threshold: 10485760,
    },
}
```

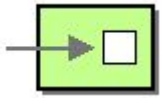
Best Practice - EIP Patterns

- Works anywhere





EIP Cloud Patterns



Service Call

Plugins

- Kubernetes
- Ribbon
- Consul
- Etcd
- Zookeeper

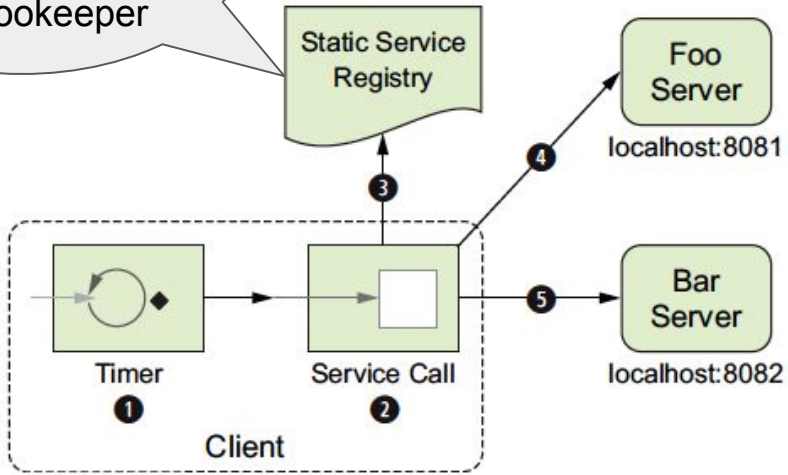
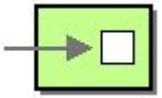


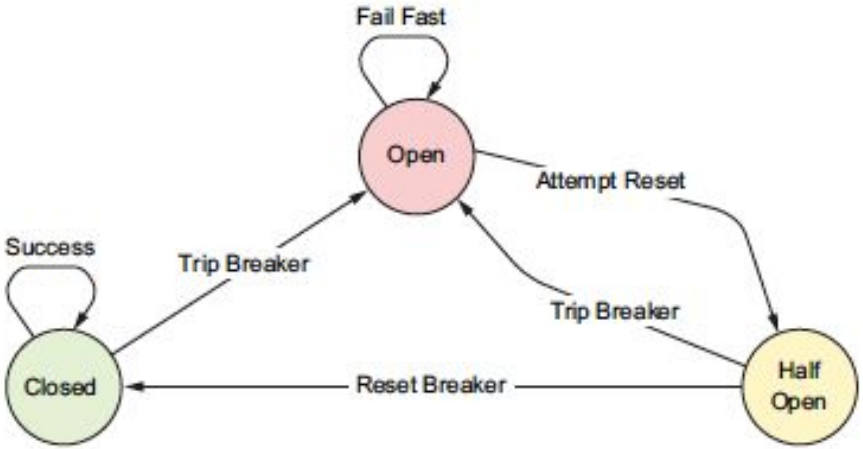
Figure 17.9 A timer (1) triggers the Service Call EIP (2) to call a clustered service. The physical locations of the service are looked up in the service registry (3). The service is then called in a round-robin fashion by calling either Foo server (4) or Bar server (5).

```
from("timer")
  .serviceCall("hello-service");
```

EIP Cloud Patterns



Hystrix EIP

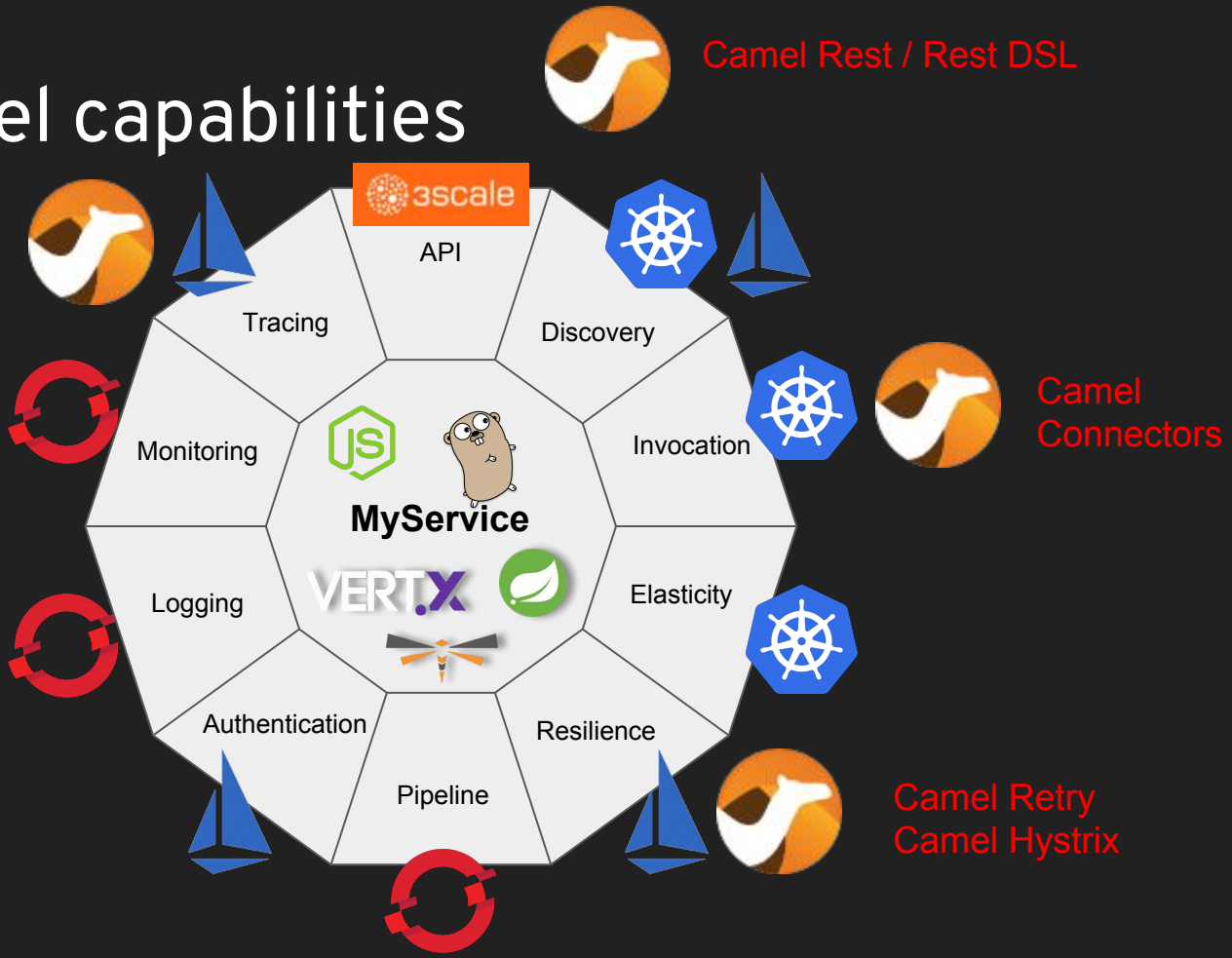



```
from("timer:foo")
  .hystrix()
    .to("http:myservice")
  .onFallback()
    .to("bean:fallback")
  .end()
```

EIP Cloud Patterns

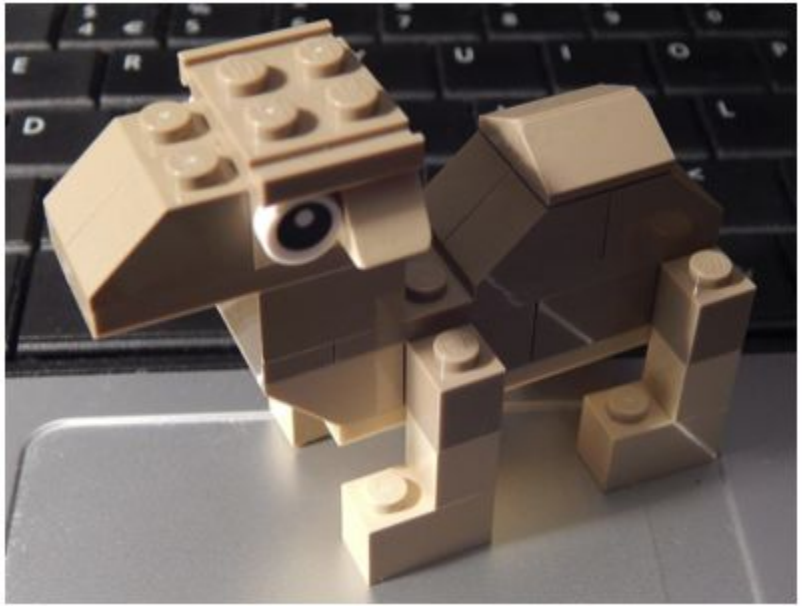
Distributed Tracing

Usable Camel capabilities

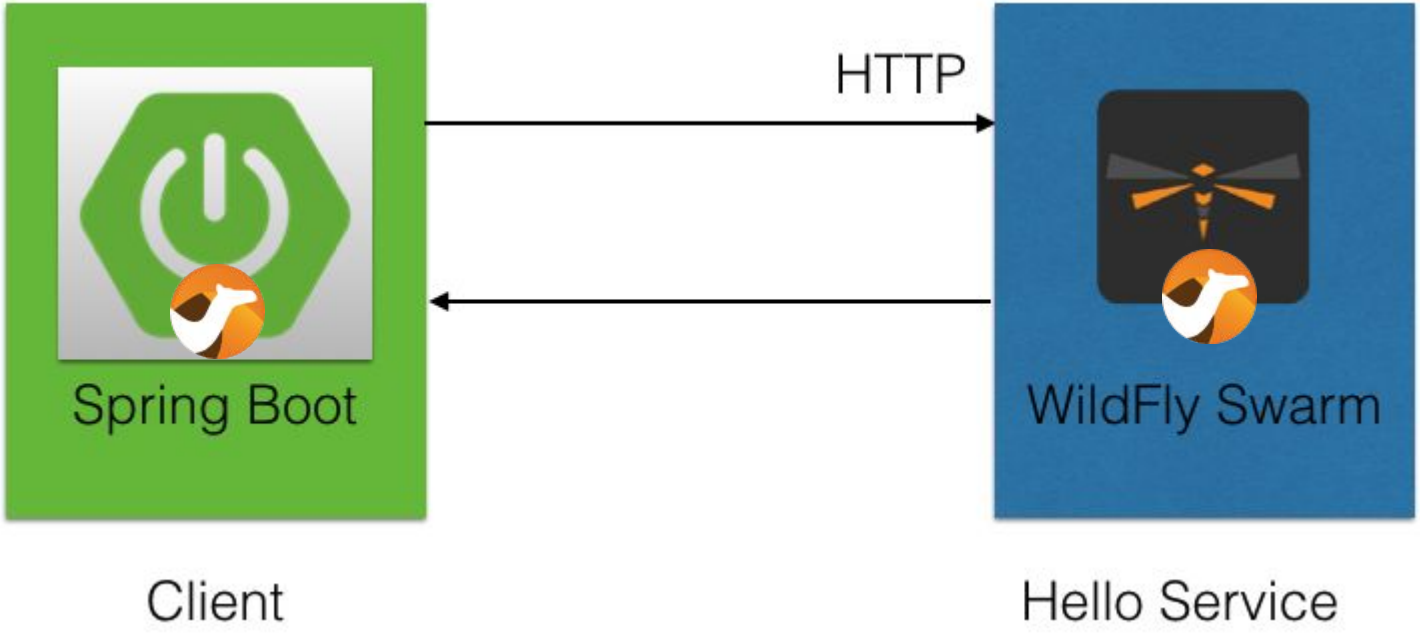


@burrsutter

Demo Time



Basic Demo



More Information

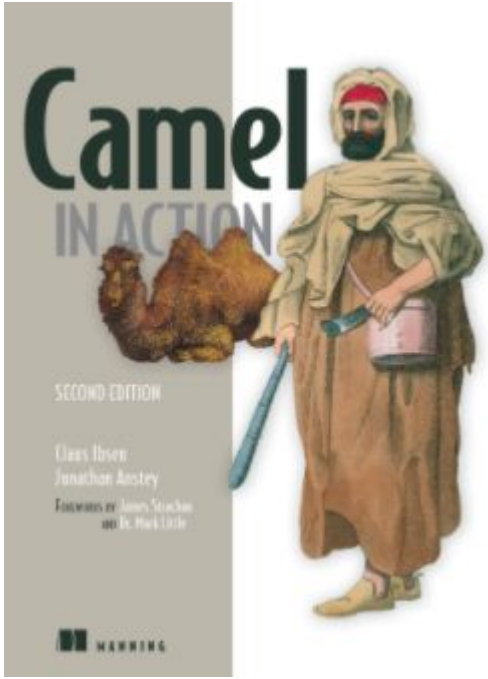
- Slides and Demo source code:
  https://github.com/davsclaus/camel-riders-in-the-cloud
- Apache Camel website:
  http://camel.apache.org
- Best "What is Apache Camel" article:
  https://dzone.com/articles/open-source-integration-apache
- My blog:
  http://www.davsclaus.com
- DevNation Live
  https://developers.redhat.com/devnationlive

Camel in Action 2nd edition

- Discount code (40%):

  `ctwdevnatlive18`

  (ordering from Manning)



https://www.manning.com/books/camel-in-action-second-edition

Q & A