



Message Driven Microservices & Monoliths with Micronaut

Todd Sharp

Developer Advocate - Cloud & Cloud DB

todd.sharp@oracle.com

@recursivecodes

Q1 - 2021

Safe Harbor

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

Statements in this presentation relating to Oracle's future plans, expectations, beliefs, intentions and prospects are "forward-looking statements" and are subject to material risks and uncertainties. A detailed discussion of these factors and other risks that affect our business is contained in Oracle's Securities and Exchange Commission (SEC) filings, including our most recent reports on Form 10-K and Form 10-Q under the heading "Risk Factors." These filings are available on the SEC's website or on Oracle's website at <http://www.oracle.com/investor>. All information in this presentation is current as of September 2019 and Oracle undertakes no duty to update any statement in light of new information or future events.





About Me

- Currently
 - Developer Advocate @ Oracle
- Previously
 - AT&T
 - Booz, Allen & Hamilton
- 17 Years Full-Stack
 - Java, Groovy, Grails, ColdFusion
 - JavaScript, Angular, Node

In the next 60 minutes, you'll learn how to use messaging to reliably communicate between distributed services using popular tools and services and the Micronaut framework.

E-commerce Workflow

1. Order Placed
2. Order Shipped
3. Order Updated

Services Need To Talk!

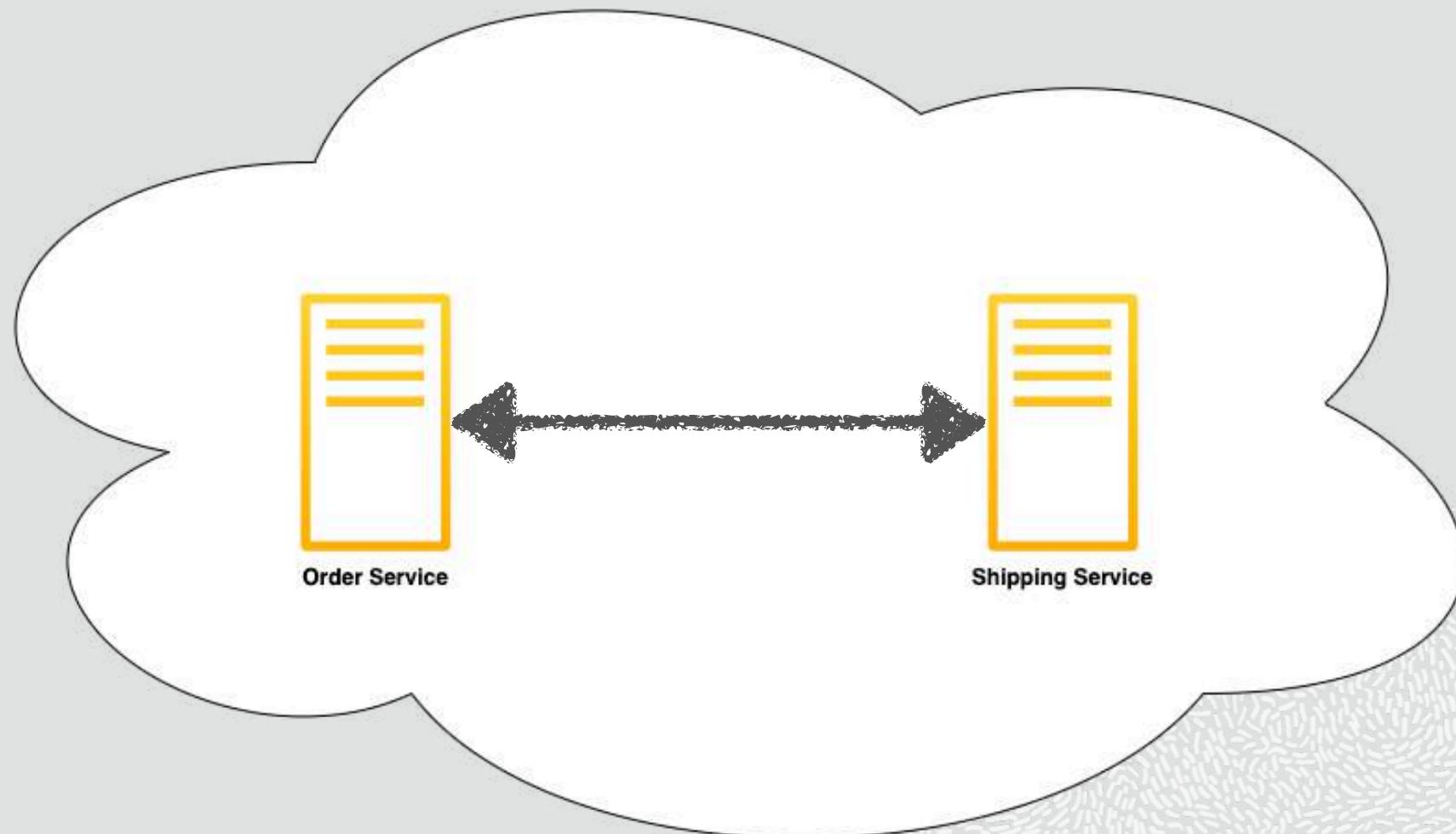




Image 7 [Ryan McGuire](#) from Pixabay



8 Image by [bluebudgie](#) from [Pixabay](#)

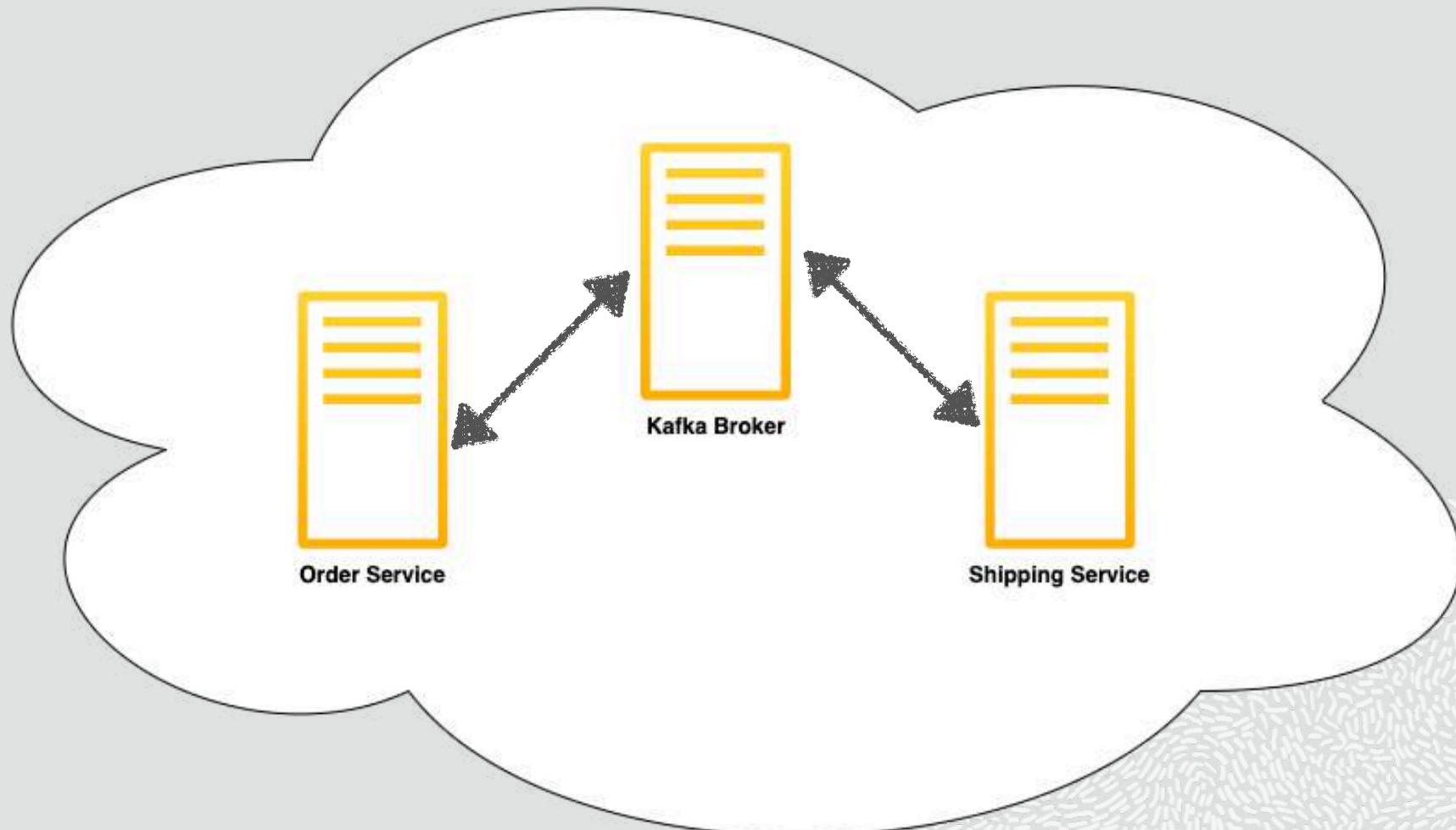


GAMER DREAM



10 Image by [Nebraska Department of Education](#) from [Pixabay](#)

Kafka



Getting Started with Kafka

```
1 # download kafka
2 $ curl -O https://ftp.wayne.edu/apache/kafka/2.7.0/kafka_2.13-2.7.0.tgz
3 # unzip & switch directory
4 $ tar xvf kafka_2.13-2.7.0.tgz && cd kafka_2.13-2.7.0/
5 # start zookeeper
6 $ bin/zookeeper-server-start.sh config/zookeeper.properties
7 # start broker
8 $ bin/kafka-server-start.sh config/server.properties
```

Getting Started with Kafka

```
1 # create topic  
2 $ bin/kafka-topics.sh --create --topic order-topic --bootstrap-server localhost:9092  
3 # test producer  
4 $ bin/kafka-console-producer.sh --topic order-topic --bootstrap-server localhost:9092  
5 # test consumer  
6 $ bin/kafka-console-consumer.sh --topic order-topic --bootstrap-server localhost:9092
```



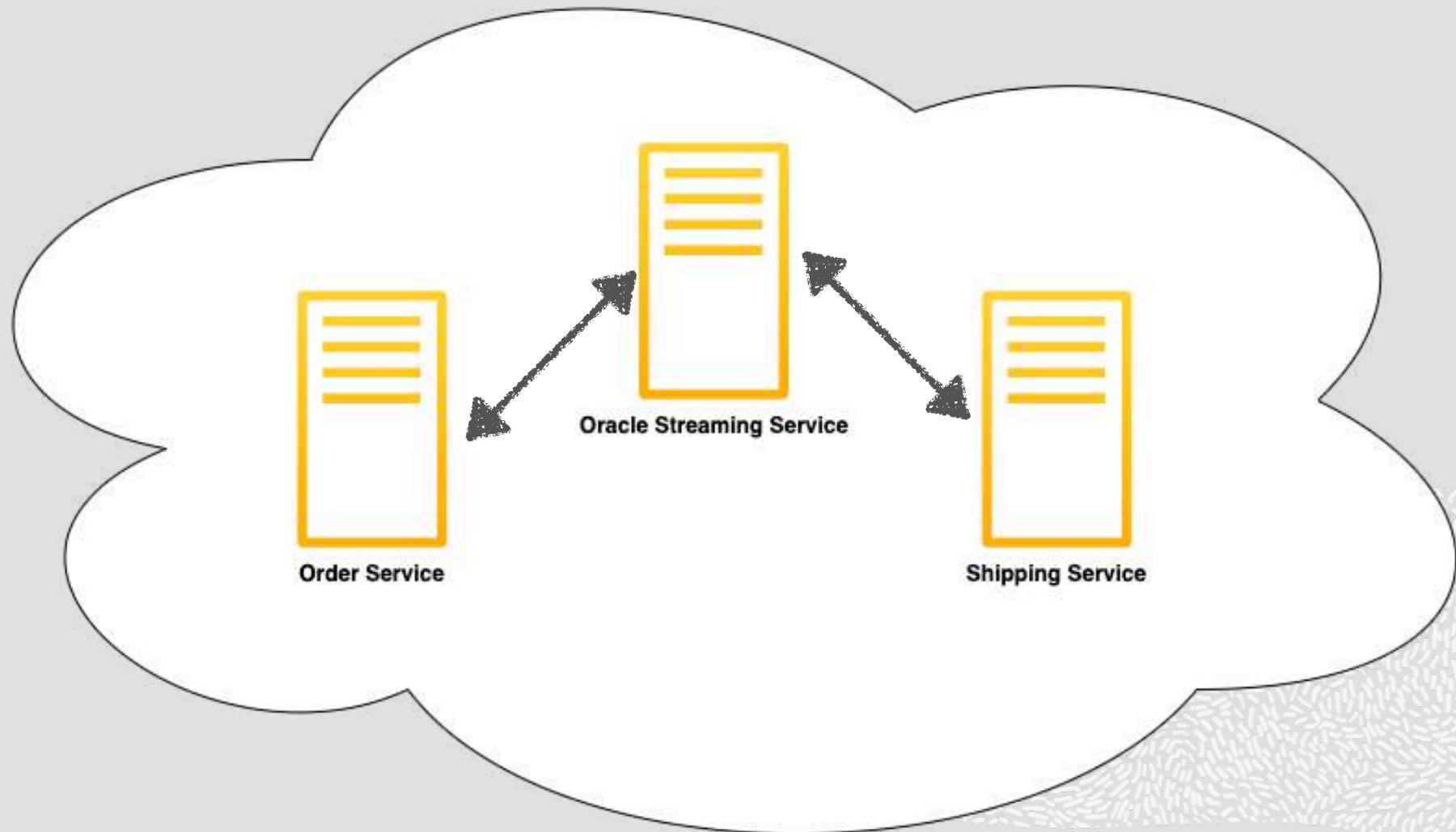
Demo: Orders & Shipments



Messaging in the Cloud



Oracle Streaming Service



Oracle Streaming Service

The screenshot shows the Oracle Streaming Service interface. The left sidebar includes 'Solutions and Platform', 'Analytics' (which is selected), 'Resource Manager', and 'Email Delivery'. The main content area has a title 'Streaming' with a red star icon. On the left, under 'Analytics', there's a 'Streaming' section with a 'Create Stream' button and a red star icon. Below it are fields for 'Scope' and 'COMPARTMENT', with 'demo-compartment' selected. To the right, a table lists streams with columns 'Name', 'Status', and 'Created'. The table contains three entries, all marked as 'Active':

Name	Status	Created
[redacted]	● Active	Tue, 23 Jul 2019 13:22:45
[redacted]	● Active	Tue, 30 Apr 2019 15:45:12
[redacted]	● Active	Fri, 05 Apr 2019 15:45:12

Oracle Streaming Service

- Create Stream (stream == topic)
- Create User, Group, Policy & Auth Token
- Configure KAFKA_SASL_JAAS_CONFIG

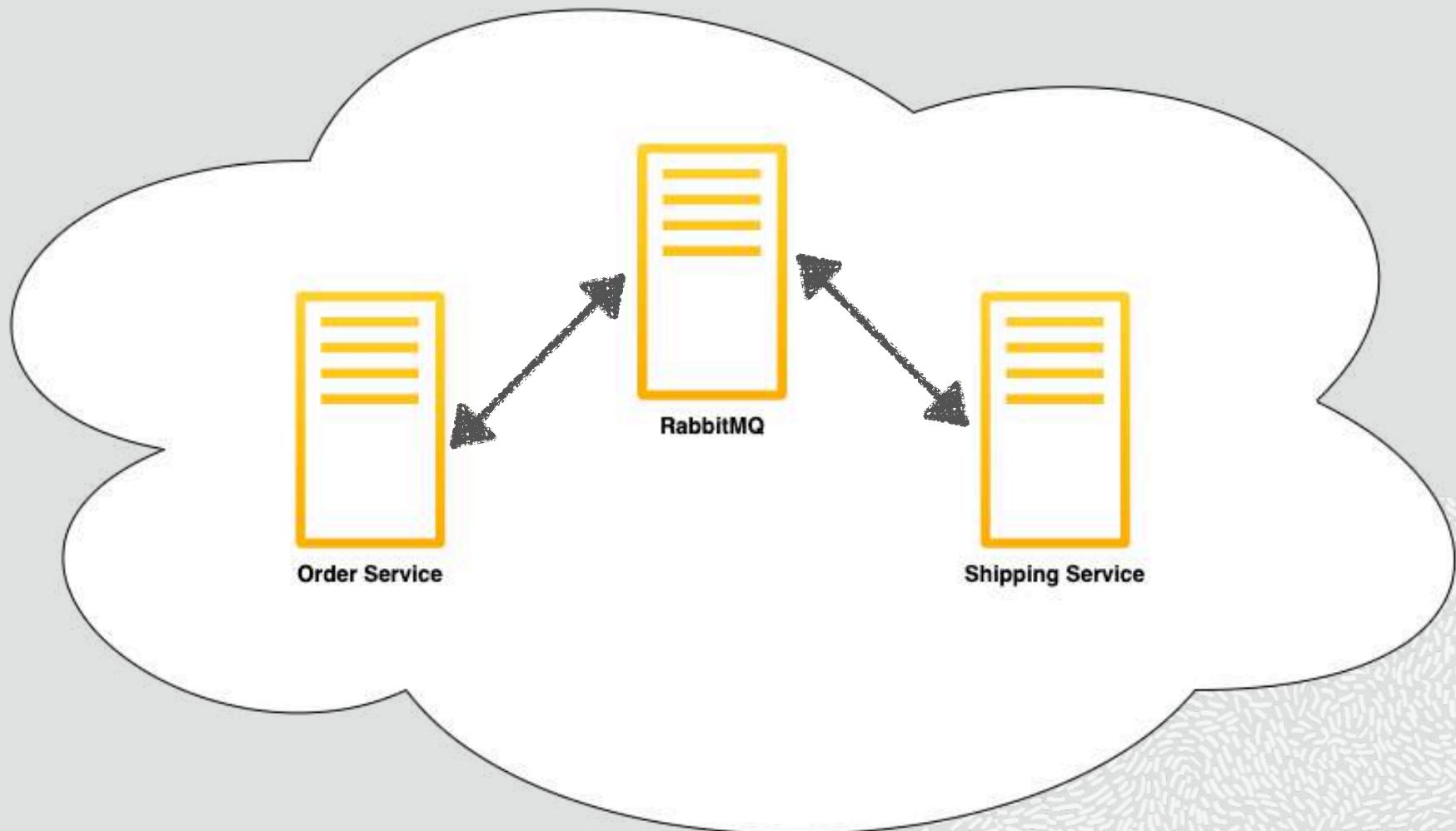
Oracle Streaming Service

KAFKA_SASL_JAAS_CONFIG

```
org.apache.kafka.common.security.plain.PlainLoginModule required username="[tenancyName] /  
[username]/[stream pool OCID]"  
password="[auth token]";
```

Demo: Switch Config to use OSS

RabbitMQ



RabbitMQ - Create Queue(s)

▼ Add a new queue

Type: Classic

Name: order-queue *

Durability: Durable

Auto delete: No

Arguments: [] = [] String

Add Message TTL ? | Auto expire ? | Max length ? | Max length bytes
Dead letter exchange ? | Dead letter routing key ? | Single active co.
Lazy mode ? Master locator ?

Add queue

RabbitMQ - Create Exchange

▼ Add a new exchange

Name: *

Type:

Durability:

Auto delete:

Internal:

Arguments: = String

Add Alternate exchange ?

Add exchange

RabbitMQ - Bind Exchange to Queue(s)

Exchange: micronaut.demo

▼ Overview

Message rates last minute ?

Currently idle

Details

Type	direct
Features	durable: true
Policy	

▼ Bindings

This exchange

↓

... no bindings ...

Add binding from this exchange

To queue: order-queue *

Routing key: order

Arguments: | = String

Bind

RabbitMQ - Create Listener(s) & Producer(s)

```
1 # order-svc
2 $ mn create-rabbitmq-producer codes.recursive.messaging.OrderProducer
3 $ mn create-rabbitmq-listener codes.recursive.messaging.ShipmentConsumer
4
5 # shipment-svc
6 $ mn create-rabbitmq-listener codes.recursive.messaging.OrderConsumer
7 $ mn create-rabbitmq-producer codes.recursive.messaging.ShipmentProducer
```

Demo: RabbitMQ for Messaging

Regarding Threading

- You can control the number of threads used for consumers via config
- RabbitMQ will not use multi-threaded consumers unless you specify the executor



```
1 @RabbitListener(executor = "consumer")
```

```
micronaut:  
  executors:  
    consumer:  
      type: fixed  
      nThreads: 25
```

A dark rectangular box containing three small, semi-transparent colored circles: red, yellow, and green, arranged horizontally. Below them is a block of configuration code for Micronaut executors.

Blog Posts

- <https://blogs.oracle.com/developers/easy-messaging-with-micronauts-kafka-support-and-oracle-streaming-service>
- <https://blogs.oracle.com/developers/message-driven-microservices-monoliths-with-micronaut-part-1:-installing-kafka-sending-your-first-message>
- <https://blogs.oracle.com/developers/message-driven-microservices-monoliths-with-micronaut-part-2:-consuming-messages>
- <https://blogs.oracle.com/developers/message-driven-microservices-monoliths-with-micronaut-part-3:-switching-to-oracle-streaming-service>
- <https://blogs.oracle.com/developers/message-driven-microservices-monoliths-with-micronaut-part-4:-using-rabbitmq-for-messaging>

Code Repos

- <https://github.com/recursivecodes/order-svc-kafka>
- <https://github.com/recursivecodes/shipping-svc-kafka>
- <https://github.com/recursivecodes/order-svc-rabbitmq>
- <https://github.com/recursivecodes/shipping-svc-rabbitmq>

Socials & Contact

- <https://blogs.oracle.com/author/todd-sharp>
- <https://recursive.codes>
- <https://twitter.com/recursivecodes>
- <https://www.linkedin.com/in/toddrsharp/>
- <https://github.com/recursivecodes>
- todd.sharp@oracle.com