



FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

Universidad de las Américas

INTEGRACION DE SISTEMAS

**Taller – Configuración de RabbitMQ e Integración con
Apache Camel**

Camila Cabrera

Kristiany Cerón

Tito Jaramillo

Camila Vega

17 de diciembre del 2025

1. Repositorio Github:

<https://github.com/krissceron/EntregablesDeTaller.git>

2. Capturas de pantalla

Consola de RabbitMQ

RabbitMQ™ RabbitMQ 3.13.7 Erlang 26.2.5.16 Refreshed

Overview Connections Channels Exchanges Queues and Streams Admin

Queue test.camel.queue

Overview

Queued messages [last minute] ?

Ready: 14
Unacked: 0
Total: 14

Message rates [last minute] ?

Publish: 0.20/s

Details

Features	durable: true queue storage version: 1	State	running	Consumers	0	Consumer capacity	0%	Messages	Total 14 Ready 14 Unacked 0 In memory 1 Persistent 0 Transient 0 Paged Out 13	Message body bytes 546 B Process memory 34 KB
Policy										
Operator policy										
Effective policy definition										

Consumers (0)

Bindings (2)

From	Routing key	Arguments
(Default exchange binding) test.camel.exchange	test.camel.queue	Unbind

II

RabbitMQ™ RabbitMQ 3.13.7 Erlang 26.2.5.16

Overview Connections Channels Exchanges Queues and Streams Admin

From exchange:

Routing key:

Arguments: = String

Bind

Publish message

Get messages

Warning: getting messages from a queue is a destructive action. ?

Ack Mode: Nack message requeue true

Encoding: Auto string / base64

Messages: 1

Get Message(s)

Message 1

The server reported 26 messages remaining.

Exchange	test.camel.exchange
Routing Key	test.camel.queue
Redelivered	0
Properties	headers: CamelMessageTimestamp: 1766022880840 fireTime: 1766022880
Payload	39 bytes
Encoding	string

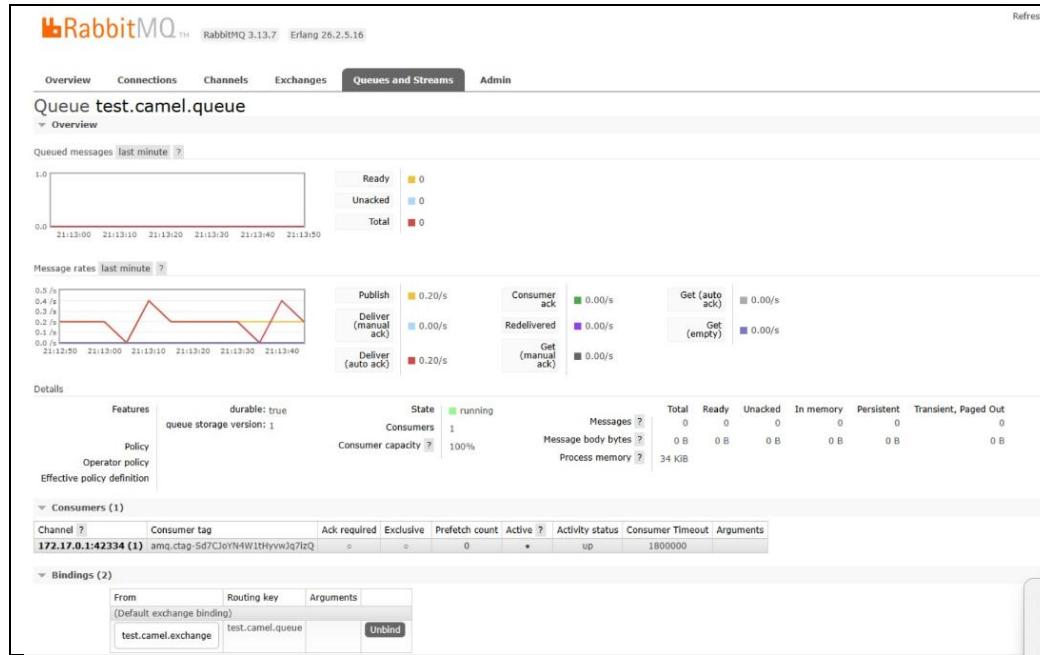
Mensaje generado en 2025-12-17 20:54:40

Move messages

To move messages, the shovel plugin must be enabled, try:

```
$ rabbitmq-plugins enable rabbitmq_shovel rabbitmq_shovel_management
```

Delete



The screenshot shows the Eclipse IDE interface with the Camel application open. The code editor displays `ProducerRoute.java`:

```

import org.apache.camel.builder.RouteBuilder;
import org.springframework.stereotype.Component;

@Component
public class ProducerRoute extends RouteBuilder {
    @Override
    public void configure() throws Exception {
        from("timer::generate?period=5000")
            .routeId("route2")
            .to("rabbitmq://test.camel.queue");
    }
}

```

The terminal window shows the log output:

```

PS C:\Users\kriss\camel-rabbit-lab> mvn spring-boot:run
[...]
2025-12-17 21:12:22.716 INFO 4828 --- [.camel.exchange] route2
erado en 2025-12-17 21:11:38 : RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:22.717 INFO 4828 --- [.camel.exchange] route2
1:43 : RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:22.717 INFO 4828 --- [.camel.exchange] route2
1:48 : RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:22.717 INFO 4828 --- [.camel.exchange] route2
1:53 : RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:22.717 INFO 4828 --- [.camel.exchange] route2
1:58 : RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:22.717 INFO 4828 --- [.camel.exchange] route2
2:03 : RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:23.659 INFO 4828 --- [imer:://generate] route1
2025-12-17 21:12:23.667 INFO 4828 --- [.camel.exchange] route2
2:23 : Envio: Mensaje generado en 2025-1
: RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:28.657 INFO 4828 --- [imer:://generate] route1
2025-12-17 21:12:28.666 INFO 4828 --- [.camel.exchange] route2
2:28 : Envio: Mensaje generado en 2025-1
: RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:33.661 INFO 4828 --- [imer:://generate] route1
2025-12-17 21:12:33.671 INFO 4828 --- [.camel.exchange] route2
2:33 : Envio: Mensaje generado en 2025-1
: RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:38.665 INFO 4828 --- [imer:://generate] route1
2025-12-17 21:12:38.671 INFO 4828 --- [.camel.exchange] route2
2:38 : Envio: Mensaje generado en 2025-1
: RECIBIDO DESDE RABBITMQ: Mensaje gen
2025-12-17 21:12:43.666 INFO 4828 --- [imer:://generate] route1
2025-12-17 21:12:43.669 INFO 4828 --- [.camel.exchange] route2

```

1. Qué patrón de integración se aplicó.

En el taller se aplicó el patrón de mensajería asíncrona (Asynchronous Messaging), específicamente el patrón Productor–Consumidor mediante un Message Broker.

Este patrón permite que un productor envíe mensajes a una cola gestionada por un broker (RabbitMQ) sin necesidad de conocer directamente al consumidor. Apache Camel se encargó de orquestar el envío y la recepción de mensajes utilizando rutas de integración.

2. Cómo se logró el desacoplamiento productor consumidor.

El desacoplamiento se logró mediante el uso de RabbitMQ como intermediario entre el productor y el consumidor.

El productor Camel únicamente publica mensajes en una cola (test.camel.queue) sin saber quién los consumirá ni cuándo. Por su parte, el consumidor escucha esa cola y procesa los mensajes de forma independiente. Gracias a esto, ambos componentes pueden ejecutarse, modificarse o reiniciarse sin afectar directamente al otro, siempre que la cola esté disponible.

3. Ventajas que observaron durante la práctica.

- Desacoplamiento de sistemas, ya que productor y consumidor no dependen entre sí.
- Mayor escalabilidad, permitiendo agregar más consumidores sin cambiar el productor.
- Tolerancia a fallos, porque los mensajes permanecen en la cola si el consumidor no está disponible.
- Procesamiento asincrónico, lo que mejora el rendimiento y evita bloqueos.
- Facilidad de monitoreo, gracias a la consola de administración de RabbitMQ.