Sincrono e Assíncrono Camel:

Há uma alternativa ao direct e multicast. Na rota e sub-rotas podemos aplicar algo chamado de **Staged event-driven architecture** ou simplesmente SEDA.

A ideia do SEDA é que cada rota (e sub-rota) possua uma fila dedicada de entrada e as rotas enviam mensagens para essas filas para se comunicar. Dentro dessa arquitetura, as mensagens são chamadas de eventos. A rota fica então consumindo as mensagens/eventos da fila, tudo funcionando em paralelo.

Para usar SEDA basta substituir a palavra direct por seda, com isso, o multicast se tornará desnecessário:

from("file:pedidos?delay=5s&noop=true").

routeId("rota-pedidos").

to("seda:soap").

to("seda:http");

from("seda:soap").

routeId("rota-soap").

log("chamando servico soap ${body}").

to("mock:soap");

from("seda:http").

routeId("rota-http").

setProperty("pedidoId", xpath("/pedido/id/text()")).

setProperty("email", xpath("/pedido/pagamento/email-titular/text()")).

split().

xpath("/pedido/itens/item").

filter().

xpath("/item/formato[text()='EBOOK']").

setProperty("ebookId", xpath("/item/livro/codigo/text()")).

setHeader(Exchange.HTTP\_QUERY,

simple("clienteId=${property.email}&pedidoId=${property.pedidoId}&ebookId=${property.ebookId}")).

to("http4://localhost:8080/webservices/ebook/item");

Enquanto o direct usa o processamento síncrono, o seda usa assíncrono.

É importante mencionar que seda não implementa qualquer tipo de persistência ou a recuperação, sendo tudo processado dentro da JVM. Se você precisa de persistência, confiabilidade ou processamento distribuído, JMS é a melhor escolha.

Fique tranquilo, nós usaremos o JMS com ActiveMQ nesse treinamento.