Artigo
Julio Esquerdo · Out. 10, 2022 15min de leitura

Produzindo e Consumindo mensagens no Kafka com o Caché/IRIS via REST

Olá,

De acordo com a Wikipédia, o kafka é:

"uma plataforma open-source de processamento de streams desenvolvida pela Apache Software Foundationescrita em Scalae JavaO projeto tem como objetivo fornecer uma plataforma unificada, de alta capacidade e baixa latência para tratamento de dados em tempo real. Sua camada de armazenamento é, essencialmente, uma "fila de mensagens de publishers/subscribers maciçamente escalável projetada como um log de transações distribuído", tornando-o altamente valioso para infra-estruturas corporativas que processam transmissão de dados."

Neste pequeno exemplo, vamos consumir e publicar mensagens no Kafka utilizando sua interface mais básica, apenas para vermos o funcionamento. E sempre lembrando que no IRIS temos um adaptador nativo para comunicação com o Kafka.

Vamos utilizar o Linux como sistema operacional para o Kafka e o IRIS.

Após instalar o Kafka (existem vários tutoriais na web) vamos ter disponíveis o kafka-console-consumer.sh e o kafka-console-producer.sh. Eles nos permitem consumir e publicar mensagens em um tópico do Kafka. Um breve exemplo é executar o kafka-console-consumer.sh em uma tela e em outra executar o kafka-console-producer.sh, e neste último ir mandando mensagens para o Kafka. A medida que as mensagens são enviadas elas são recebidas e apresentadas pelo processo de consumer:

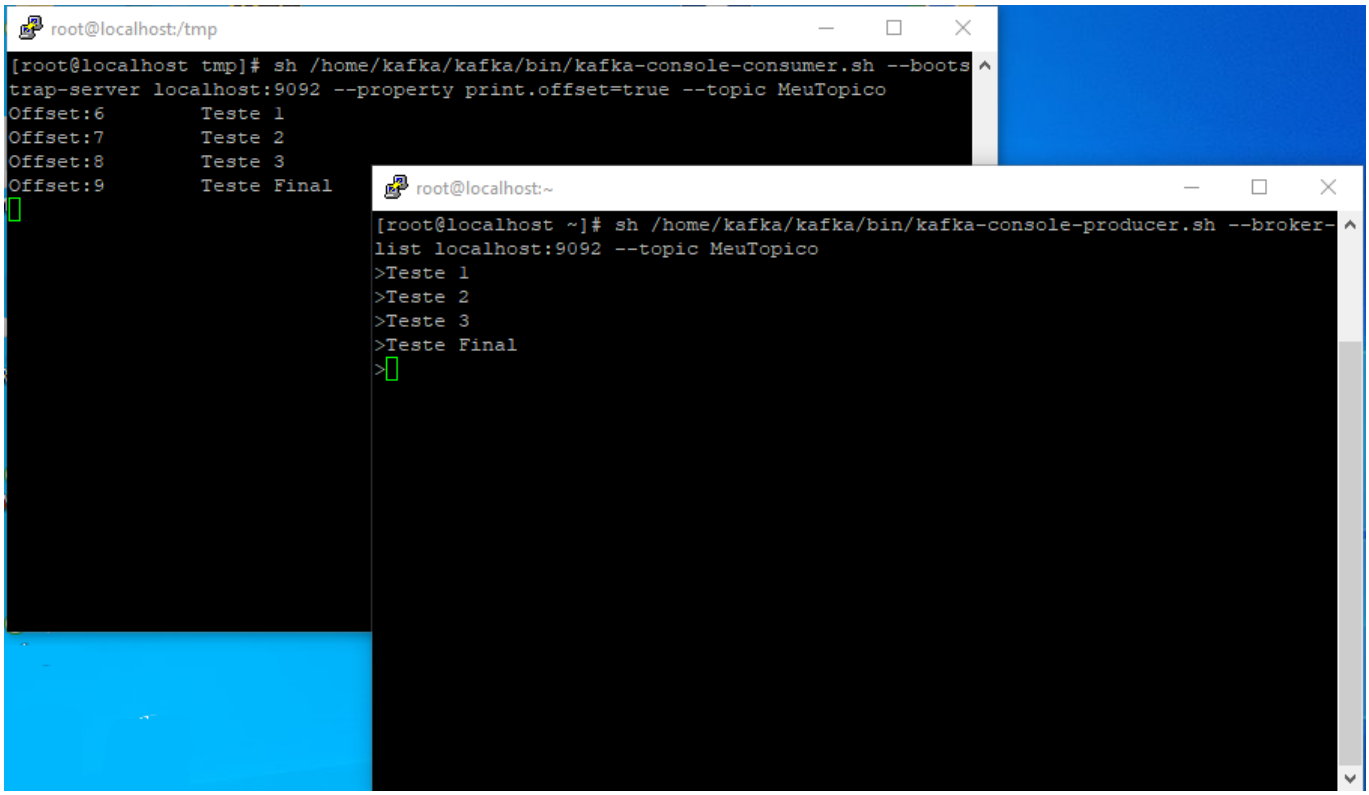

Fig 1. Exemplo do Kafka Consumer e Producer

Para acessar as rotinas de consumer e producer do Kafka montamos uma rotina que utiliza a função $ZF(-1), chamando via shell do Linux estas rotinas:

```
kafka	;
producer(topico,mensagem)
	Set resp=$ZF(-1,"echo '"_mensagem_"'¦sudo sh /home/kafka/kafka/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic "_topico_" 1> /tmp/kafkaProducer.txt 2>/tmp/kafkaProducer.err")
	Quit resp
consumer(topico,offset,max,timeout)
	Kill response
	Set resp=$ZF(-1,"rm -rf /tmp/kafkaConsumer.txt")
	If resp=0
	{
		Set resp=$ZF(-1,"sudo sh /home/kafka/kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --property print.offset=true --topic "_topico_" --partition 0 --offset "_offset_" --max-messages "_max_" --timeout-ms "_timeout_" 1> /tmp/kafkaConsumer.txt 2>/tmp/kafkaConsumer.err")
		If resp=0
		{
			Set pos=0
			Set file = ##class(%File).%New("/tmp/kafkaConsumer.txt")
		Set sc =file.Open("R")
			Set str=file.ReadLine()
		While 'file.AtEnd
		{
		Set pos=pos+1
		Set response(pos,0)=$Piece($Piece(str,$Char(9),1),":",2)
		Set response(pos,1)=$Piece(str,$Char(9),2,99999)
		Set offset=$Piece($Piece(str,$Char(9),1),":",2)
		Set str=file.ReadLine()
		}
	Do file.Close()
			Set response=offset_";"_pos
			Kill pos,sc,str
		}
	}
	Quit resp
```

A rotina está bem simples, o rótulo producer recebe a mensagem e o tópico onde se deseja publicar a mensagem e chama o kafka-console-producer. Já o rótulo consumer recebe o tópico, o offset da mensagem inicial desejada, a quantidade de mensagens e um valor de timeout em milissegundos.

Desta forma podemos fazer a chamada as rotinas do Kafka por dentro do Caché ou do Iris:

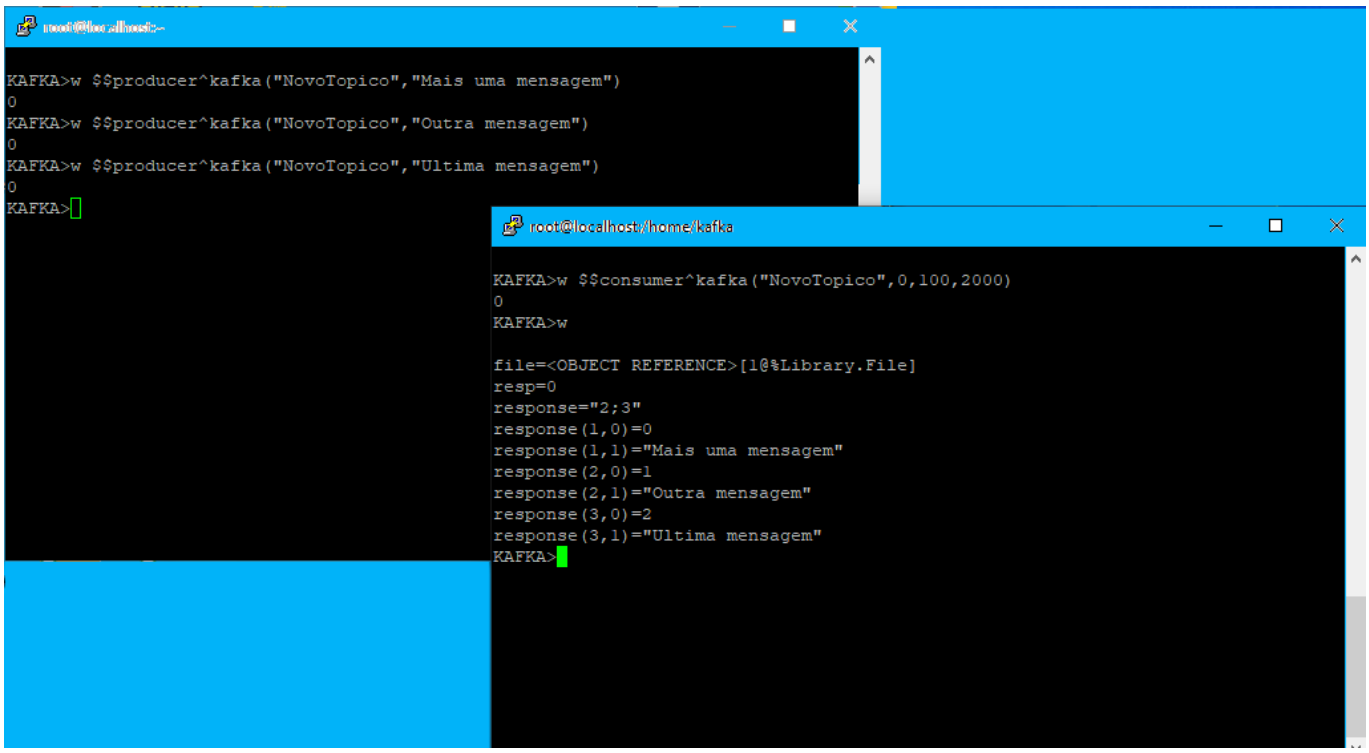

Fig. 2 – Utilizando a rotina dentro do Caché/IRIS para produzir/consumir mensagens

Uma vez que nossa rotina está funcionando, podemos agora montar uma camada REST para podermos acessa-la mais facilmente. Para isso criamos a classe kafka.rest:

```
Class kafka.rest Extends %CSP.REST
{

Parameter CONTENTTYPE = "application/json";

Parameter CHARSET = "utf-8";

XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
{
<Routes>
<Route Url="/test" Method="GET" Call="Test" />
<Route Url="/producer" Method="POST" Call="Producer" />
<Route Url="/consumer" Method="GET" Call="Consumer" />
</Routes>
}

ClassMethod Test() As %Status
{
	Set obj={
		"produto": "Api Kafka Demo",
		"versao": "1.00",
		"data": ($zdt($h,3,1))
	}

	Write obj.%ToJSON()

	Quit 1
}

ClassMethod Consumer() As %Status [ ProcedureBlock = 0 ]
{
```

```
		Set topico=%request.Get("topico")
		Set offset=%request.Get("offset")
		Set max=%request.Get("max")
		Set timeout=%request.Get("timeout")
		Set resp=$$consumer^kafka(topico,offset,max,timeout)
		If resp=0
		{
			If $Data(response)
			{
				Set tot=$Piece(response,";",2)
				Set ultimoOffset=$Piece(response,";",1)

				Set saida={
					"ultimoOffset": (ultimoOffset),
					"numMsgRetornado": (tot),
					"mensagens": []
				}
				For i=1:1:tot
				{
					Set obj = {
			"offset": (response(i,0)),
						"conteudo": (response(i,1))
					}
					Do saida.mensagens.%Push(obj)
				}
				Write saida.%ToJSON()

			} Else {
				Set %response.Status="500 Internal Server Error"
			}

		} Else {
			Set %response.Status="500 Internal Server Error"
		}
		Quit 1
}

ClassMethod Producer() As %Status
{
		Set topico=%request.Get("topico")
		Set mensagem=%request.Content.Read()
		Set resp=$$producer^kafka(topico,mensagem)
		Kill msgKafka
		If resp'=0
		{
			Set %response.Status="500 Internal Server Error"
		}
		Quit 1
}

}
```

Lembre de configurar a classe REST para que ela possa ser acessada em Administração do Sistema->Segurança->Aplicações->Aplicações Web.

Uma vez que sua classe esteja disponível podemos consumir o serviço com o Postman:

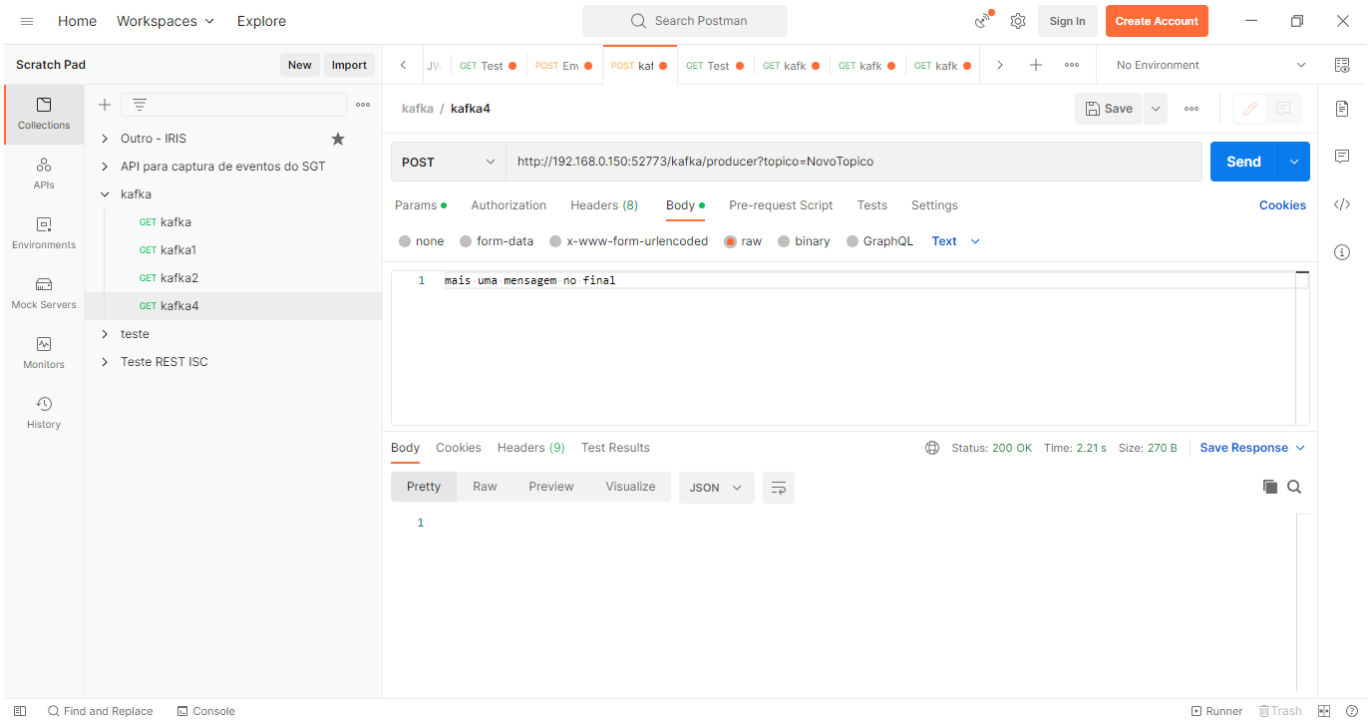

Fig 3. Postman enviando uma nova mensagem para o Kafka via API REST

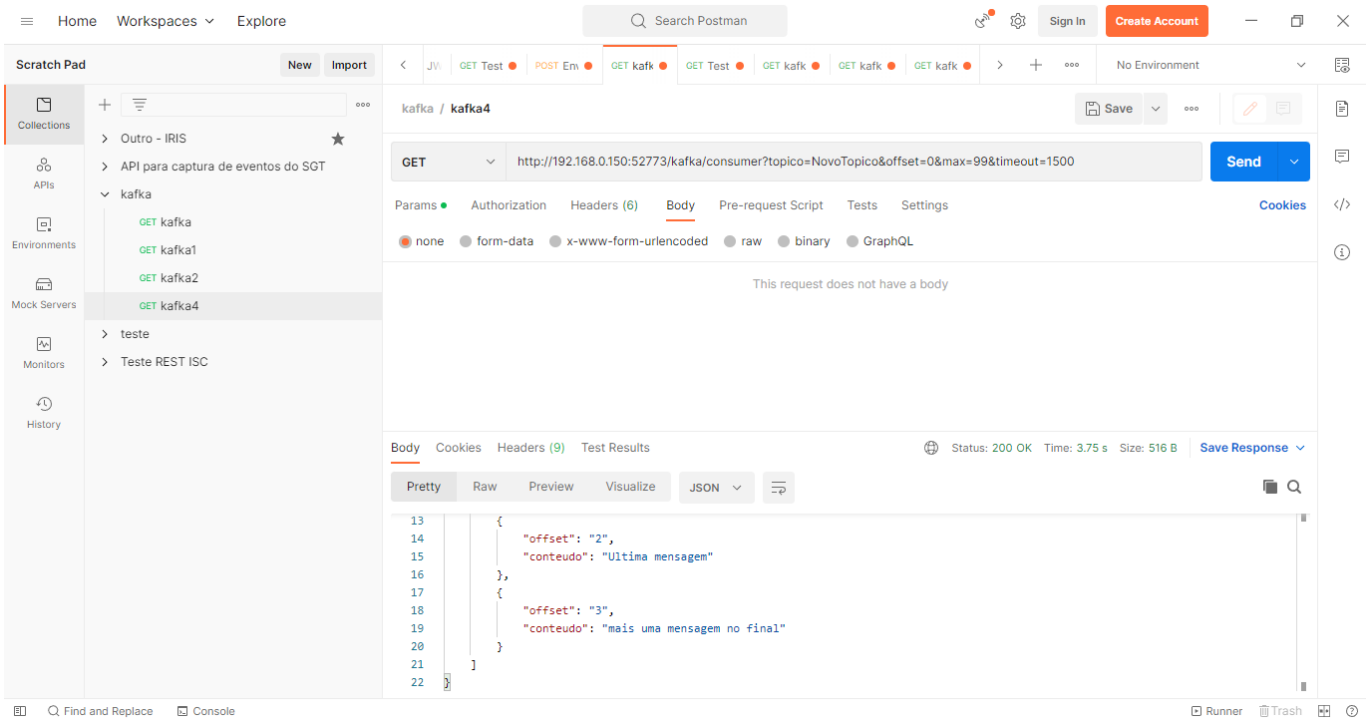

Fig 4: Postman consumindo mensagens do Kafka via API REST

Este é apenas um exemplo do que podemos fazer com o IRIS. Através dele podemos não somente criar interfaces que vão facilitar a comunicação entre diversos sistemas, mas também coletar, filtrar e armazenar informações relevantes de diversos sistemas, e depois disponibiliza-las para consumo via REST, JBDC ou outra tecnologia desejada. E, novamente lembrando, o IRIS tem um adaptador nativo para comunicação com o Kafka.

Bons códigos!

#Caché #InterSystems IRIS

URL de origem:https://pt.community.intersystems.com/post/produzindo-e-consumindo-mensagens-no-kafka-com-o-cach%C3%A9iris-rest