Universidade Federal de Pernambuco
Centro de Informática

Graduação em Ciência da Computação

Controle dinâmico de filas em Rabbitmq

Proposta de Trabalho de Graduação

Aluno: Jefferson Elder da Mota Nascimento (jemn@cin.ufpe.br)
Orientador: Nelson Souto Rosa (nsr@cin.ufpe.br)
Área: Sistemas Distribuídos

Recife
2020.2

Resumo

Sistemas horizontalmente escalonáveis deixaram de ser uma possibilidade desejável e passaram a ser uma necessidade real no planejamento e desenvolvimento de softwares. Levando em consideração esse cenário, os serviços de *Message Queuing* vem se destacando entre as possibilidades oferecidas pelo mercado, graças ao poder, simplicidade e velocidade que oferecem. RabbitMq é um software *open source* capaz de receber, organizar em buffer e disponibilizar mensagens de forma confiável entre aplicações distintas. Para que a comunicação ocorra de maneira mais eficiente possível, o tamanho do *buffer* onde as mensagens são enfileiradas deve ser diretamente proporcional a capacidade do consumidor de consumi-las. As estratégias existentes para definir esse tamanho máximo, tem se mostrado ineficazes e muitas vezes reativas, exigindo recompilações e paradas totais ou parciais de sistemas em produção. Tendo em vista essa problemática, este trabalho tem por objetivo oferecer uma estratégia para definição de um valor ótimo para o tamanho dessa fila em tempo de execução, evitando intervenções corretivas e permitindo assim uma melhor performance do sistema como um todo.

Introdução

O padrão de microsserviços é um paradigma de arquitetura de software que prega a construção de sistemas complexos a partir de um conjunto de serviços menores independentes. Por apresentar uma composição descentralizada, esses sistemas permitem uma maior escalabilidade e manutenibilidade dessas estruturas que se coordenam por meio de troca mensagens entre as partes interessadas.

Message Queueing é um modelo de implementação desse padrão que oferece comunicação confiável entre diferentes componentes de um mesmo sistema sem a necessidade de acoplamento entre as partes. Essa arquitetura define protocolos assíncronos onde um serviço produtor envia mensagens que precisam ser processadas por um outro, chamado consumidor. Essas mensagens são armazenadas em uma fila por um gerenciador que se encarrega de entregar ao destinatário, uma vez que este esteja disponível. Um dos gerenciadores de mensagens de código aberto mais populares é o RabbitMQ [1].

Em um contexto onde confirmações de recebimento (*Acks*) são requeridas de forma explícita, o RabbitMQ possui um mecanismo definido por um valor que estabelece quantas mensagens devem ser enviadas ao consumidor e armazenadas em cache pelo cliente antes de esperar pelo envio de *Acks*, o *prefetch count*.

Todas as mensagens com recebimento confirmado são removidas da fila e se tornam invisíveis para outros consumidores. Portanto, um valor de *prefetch count* muito pequeno pode ser bastante prejudicial. Nesse cenário, o gerenciador de mensagens pode passar por longos períodos de espera até que finalmente receba permissão para enviar mais mensagens (*Ack*). Por outro lado, um valor grande poderia tirar muitas mensagens da fila de uma vez e entregá-las ao primeiro consumidor disponível, mantendo os outros em estado de ócio [2].

Para que a comunicação entre as partes se dê de maneira ótima, a estratégia corrente tem sido a aplicação de boas práticas para definição do *prefetch count* em tempo de compilação. Essas boas práticas se baseiam na previsão da capacidade de processamento das aplicações consumidoras, no número de processos produtores ou ainda no número de consumidores. Essas soluções mostram-se ineficazes em cenários críticos onde essas variáveis são imprevisíveis ou quando uma possível correção em tempo de execução seja necessária[1].

Esse trabalho, portanto, se propõe a oferecer uma estratégia no uso do RabbitMQ para definição de valor do *prefetch count* em tempo de execução, estabelecendo parâmetros e métodos que poderão ser aplicados na implementação de sistemas críticos de arquitetura baseada em microsserviços.

Objetivos

Este trabalho tem como objetivo fundamental definir uma estratégia para determinar um valor ótimo do parâmetro *prefetch count* em tempo de execução, para sistemas distribuídos implementados com uso do gerenciador de mensagens RabbitMQ. Para tanto, será necessário:

- Definir métricas para avaliação de desempenho;
- Implementar algoritmo que ofereça uma solução dinâmica para a definição desse valor;
- Comparar resultados obtidos com as soluções existentes no mercado.

Cronograma

Atividade	Maio	Junho				Julho				Agosto				
Revisão bibliográfica	X	X	X	X										
Implementação				X	X	X	X	X						
Estudo do algoritmo				X	X	X								
Experimentos								X	X	X				
Avaliação dos resultados										X	X			
Escrita do TG											X	X	X	
Preparação da apresentação													X	X

Referências

[1] RabbitMQ. RabbitMQ, 2021. Disponível em: <https://www.rabbitmq.com/> Acesso em: 06 mai. 2021.

[2] SACKMAN, Matthew. Some queuing theory: throughput, latency and bandwidth. blog.rabbitmq. 2012.
Disponível em: <https://blog.rabbitmq.com/posts/2012/05/some-queuing-theory-throughput-latency-and-bandwidth/> Acesso em: 06 mai. 2021.

Possíveis Avaliadores

- Prof. Carlos Ferraz;
- Prof. Kiev Gama.

Assinaturas

Recife, __ de _____ de ____

_____

Jefferson Elder da Mota Nascimento

Aluno

_____

Nelson Souto Rosa

Orientador