Circuit Breaker Pattern

Quando passamos da construção de sistemas monolíticos para microsserviços, uma das primeiras coisas que notamos é que cada um dos componentes encontrados em nosso ecossistema de microsserviços faz parte de uma rede. E comunicar através de uma rede pode nos trazer problemas que antes desconhecíamos, já que passamos a ter uma única aplicação rodando para vários componentes independentes que precisam trabalhar juntos para garantir o correto funcionamento do sistema. Quando falamos de tolerância a falhas em sistemas distribuídos, nos referimos à capacidade do sistema de continuar operando se um de seus componentes falhar.

Padrão Circuit Breaker

Em seguida, veremos mais de perto o padrão do disjuntor que vimos na classe 2.

Problema

Um sistema de microsserviço que comunica de forma síncrona pode ser exposto a uma cadeia de falhas. Se um microsserviço ficar sem resposta, seus clientes (outros microsserviços, que de alguma forma, dependem deste primeiro microsserviço para completar uma solicitação) também podem ter problemas e deixar de responder às solicitações dos clientes. O problema se propaga recursivamente através do sistema, causando falhas em cascata.

Solução

Evite enviar novos pedidos a um serviço se detectarmos um problema. Para verificar o status dos microsserviços, usamos um Circuit Breaker que atua como um intermediário na comunicação.

Verificação do status com o Actuator

Resilience4j integra-se com **Actuator** (uma série de serviços padrão que fornecem informações de status de serviço) para exibir informações de status do Circuit Breaker em tempo real de várias maneiras:

- O estado atual do Circuit Breaker pode ser monitorado usando o endpoint /**actuator/health**.

- O Circuit Breaker também publica eventos para um endpoint do Actuator, por exemplo, transições de estado em /**actuator/circuitbreakerevents**.

Resilience4j

Resilience4j é uma biblioteca usada para implementarmos aos nossos projetos tolerância a falhas. Essa biblioteca foi inspirada no Netflix Hystrix, que é uma outra biblioteca de tolerância a falhas, mas projetada para Java 8 e programação funcional

Conclusão!

Para encerrar, podemos mencionar que o padrão Circuit Breaker nos ajuda a gerenciar a cadeia de eventos ou chamadas para microsserviços ao longo do ciclo de vida de uma solicitação. Essa gestão se materializa quando ocorre um erro em um microsserviço, dando ao sistema a possibilidade de se recuperar por meio de uma série de tentativas (retry) e, caso não funcionem, facilita a implementação de um fluxo alternativo através do mecanismo de fallback .