Biblioteca de sistema distrubuído

Pedro Correa
11718563 - pedro.figueiredo563@al.faj.br

6 de setembro de 2020

Decidi escolher não uma biblioteca mas um método de comunicação que esta auxilia bastante em trabalhar com sistemas distribuídos pois eles facilitam o trabalho do desenvolvedor ao 'padroniza' a comunicação entre dois sistemas. Como os sistemas distribuídos estão em máquinas diferentes, é necessário um método de comunicação em comum para que eles consigam trocar informações, muitos utilizam o HTTP, porém há um trabalho imenso entre os desenvolvedores sobre os *payloads* que devem ser enviados, códigos de erros e o que cada um significa, conforme o crescimento do sistema e da empresa, tratar de tudo isso se torna um inferno. Além de que cada sistema pode ser escrito em uma linguagem diferente por conta de sua proposta.

1 O que é gRPC?

gRPC é um método de comunicação que foi desenvolvido pela Google para facilitar e deixar eficiente a comunção entre aplicações de diferentes linguagens e distribuídas se comunicarem mais rapidamente.

Essa comunicação utiliza o RPC, que significa *Remote Procedure Calls* (Chamada de Procedimento Remotas em tradução livre). Este protocolo é a uma das peça central do por que esse método de comunicação venho para facilitar a vida dos desenvolvedores, ele basicamente deixa o desenvolvedor chamar um procedimento de outro programa que esteja em outra máquina, com isso o desenvolvedor não precisa deixar explícito a requisição, a própria biblioteca se encarrega de fazer a chamada.

Mas como essa parte é realizada? Como o gRPC sabe qual serviço deve ser chamado para executar o procedimento? Cada serviço deverá possuir um *stub* que irá providenciar os mesmos métodos do servidor que devem ser expostos para os outros serviços. Com isso, a instância do gRPC do seu serviço saberá qual serviço deverá ser chamado, toda essa comunicação e mapeamento é gerenciado pela própria biblioteca.

2 HTTP/2

Outra peça central do gRPC é o HTTP/2, este protocolo é a versão mais otimizada do protocolo que tanto utilizamos, o HTTP, que esta na versão HTTP 1.1, porém popularmente chamamos de apenas HTTP.

E por que o gRPC utiliza ele ao invés do puro HTTP? Bom, um dos fatores que venho para a criação do gRPC é que a comunicação seja extremamente rápida, e como o HTTP transfere texto através da rede, para comunicação de muita informação acaba sendo muito lenta essa comunicação.

Com isso em mente, foi-se criado o HTTP/2 para que ao invés de ser transmitido texto pela rede, seja transmitido bits, assim a comunicação de muita informação deverá ser mais rápida pois estamos trabalhando com a camada mais baixa da rede.

3 Requisitos não funcionais

A seguir segue a lista dos requisitos não funcionais aplicados pela biblioteca gRPC.

- **Desempenho**: Como comentado anteriormente, a ideia da criação do gRPC é ser uma comunicação seja rápida por se tratar de muita informação que é necessária ser transmitida, ele se beneficia bastante do HTTP/2 para este requisito.
- **Eficiência**: Como comentado anteriormente, outra ideia do gRPC é ter uma comunicação eficiente, sem que os desenvolvedores precisem se preocupar em saber como deve ser feito a chamada para o serviço, podem simplesmente fazer a chamada como se fosse uma chamada de um método na linguagem de programação que ele preferir.
- **Manutenabilidade**: Como o responsável por expor e cuidar dos endpoints é a própria biblioteca através dos *stubs*, acaba facilitando ainda mais a manitenção e versionamento de um endpoint, pois nem o desenvolvedor que irá utilizar ou a pessoa que irá fazer a manutenção, para acrescentar ou retirar um campo, precisam saber sobre como esta feita a implementação.
- **Portabilidade**: Atualmente esta bilbioteca esta disponível, com suporte oficial, para 11 linguagens de programação (Android Java, C#/.NET, C++, Dart, Go, Java, Kotlin, Node.js, Objective-C, PHP, Python e Ruby), então independente do sistema operacional ou da capacidade técnica de seu time, a biblioteca poderá ser utilizada por eles.
- **Reusabilidade**: Possui facilidade para ser utilizada e reutilizada em várias sistemas, sem precisar deixar explícito o canal de comunicação.