**Alunos: Henrique Almeida Macedo Matrículas: UC20100461**

**Vinicius Mortari da Costa UC19201624**

**Comunicação performática entre microsserviços**

Com a ampla adoção do uso de microsserviços e implantação de sistemas na nuvem, fica claro que muitas das tecnologias e protocolos de comunicação utilizados não aproveitam realmente a infraestrutura existente dos grandes provedores de nuvem. Não só isso, mas esta comunicação é muitas vezes garantida ou atendida várias vezes.

Não é à toa que em 2015, o Google abriu seu projeto de estrutura interna de RPC como uma solução alternativa para integrar microsserviços, aplicativos móveis e em tempo real, chamado gRPC. Uma estrutura que se apoia nos ombros de gigantes como HTTP/2 e Protobuf v3, oferecendo alto desempenho, baixa latência, uso eficiente da CPU e interoperabilidade entre plataformas. Além de ser mais rápido e eficiente que uma API REST, com o gRPC, adotamos uma maneira mais idiomática e expressiva de descrever uma determinada API na linguagem de sua escolha.

**Alguns cenários de uso do gRPC:**

* **Sistemas multilíngues:**com seu suporte de geração de código nativo para uma ampla gama de linguagens de desenvolvimento, o gRPC é excelente para gerenciar conexões em um ambiente poliglota.
* **Streaming em tempo real:** quando a comunicação em tempo real é um requisito, a capacidade do gRPC de gerenciar o streaming bidirecional permite que seu sistema envie e receba mensagens em tempo real sem esperar pela comunicação de resposta do cliente.
* **Redes de baixa largura de banda e de baixa potência:** o uso de mensagens Protobuf serializadas pelo gRPC permite mensagens leves, o que contribui para maior eficiência e velocidade em redes de baixa potência e de largura de banda restritas (especialmente quando comparadas a JSON). A IoT seria um exemplo desse tipo de rede que poderia se beneficiar das APIs gRPC.

**Diferenças entre as APIs REST e gRPC:**

**Protobuf em vez de JSON / XML**

Tanto as APIs REST quanto as APIs RPC enviam e recebem mensagens usando os formatos de mensagem JSON ou XML.

O JSON se tornou um formato popular, pois é flexível, eficiente, neutro em relação à plataforma, independente de idioma, baseado em texto e legível por humanos. Porém, em determinadas situações de uso, o JSON não é rápido ou leve o suficiente ao transmitir dados entre sistemas.

Em contraste com REST, gRPC não tem problemas com velocidade e peso, oferecendo maior eficiência ao transmitir mensagens usando o formato Protobuf (buffers de protocolo).

**Construído em HTTP 2 em vez de HTTP 1.1**

As APIs REST geralmente são construídas em HTTP 1.1, que usa um modelo de comunicação de *request-response*. Isso significa que, quando um microsserviço recebe várias solicitações de mais de um cliente, ele precisa atendê-las uma de cada vez, de forma sequencial, o que torna todo o sistema mais lento.

Já o gRPC usa HTTP 2 e aproveita seu suporte para comunicação bidirecional. Assim, quando um microsserviço recebe várias solicitações de mais de um cliente, ele alcança a multiplexação atendendo a muitas solicitações e respostas simultaneamente.

**Geração de código nativo em vez de usar ferramentas de terceiros**

Os recursos de geração de código são nativos do gRPC. Com APIs REST, é necessário usar uma ferramenta de terceiros, como Swagger, para gerar automaticamente o código para chamadas de API em várias linguagens.

O compilador protoc que acompanha o gRPC é compatível com uma ampla variedade de linguagens de programação. E isso o torna excelente para ambientes onde se está conectando vários microsserviços diferentes que são codificados em diferentes idiomas e executados em diferentes plataformas.

**Transmissão de mensagens 7 a 10 vezes mais rápida**

De acordo com[testes](https://medium.com/@EmperorRXF/evaluating-performance-of-rest-vs-grpc-1b8bdf0b22da#:~:text=gRPC%20is%20roughly%207%20times,of%20HTTP%2F2%20by%20gRPC.) ,as conexões da API gRPC são mais rápidas do que as conexões da API REST.

Cerca de 7 vezes mais rápido do que REST ao receber dados e cerca de 10 vezes mais rápido do que REST ao enviar dados para esta carga específica.

**Implementação mais lenta do que REST**

Apesar dos benefícios na velocidade de transmissão de mensagens, a implementação da API gRPC é muito mais lenta do que a implementação da API REST.

Leva-se aproximadamente 45 minutos para implementar um serviço gRPC simples contra  cerca de apenas 10 minutos para implementar uma API REST.

Isso ocorre, principalmente, porque o gRPC ainda não é amplamente adotado, especialmente em comparação com a onipresença das APIs REST.