gRPC
&
Protocol buffer

Adrián Braojos
Marc Duch
Haopeng Lin Ye
Miguel Moreno

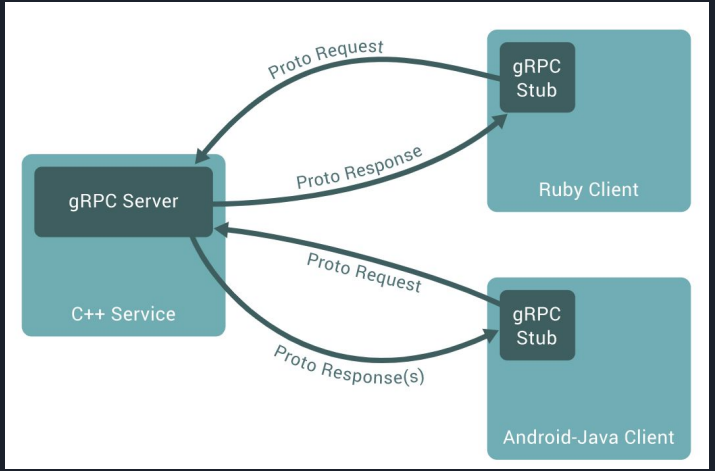

gRPC

¿Qué es y qué hace?

- Implementación de RPC diseñado por Google

  g(Google) + RPC

- Uso en comunicaciones cliente-servidor.
  Permite una comunicación tan sencilla como en máquina local.



Versiones más relevantes

-Creación: Agosto 2016

-Nueva versión cada 6 semanas

-Actual (4-10-2022): 1.49.0, 14 de Septiembre

Características

-Open Source

-Multiplataforma

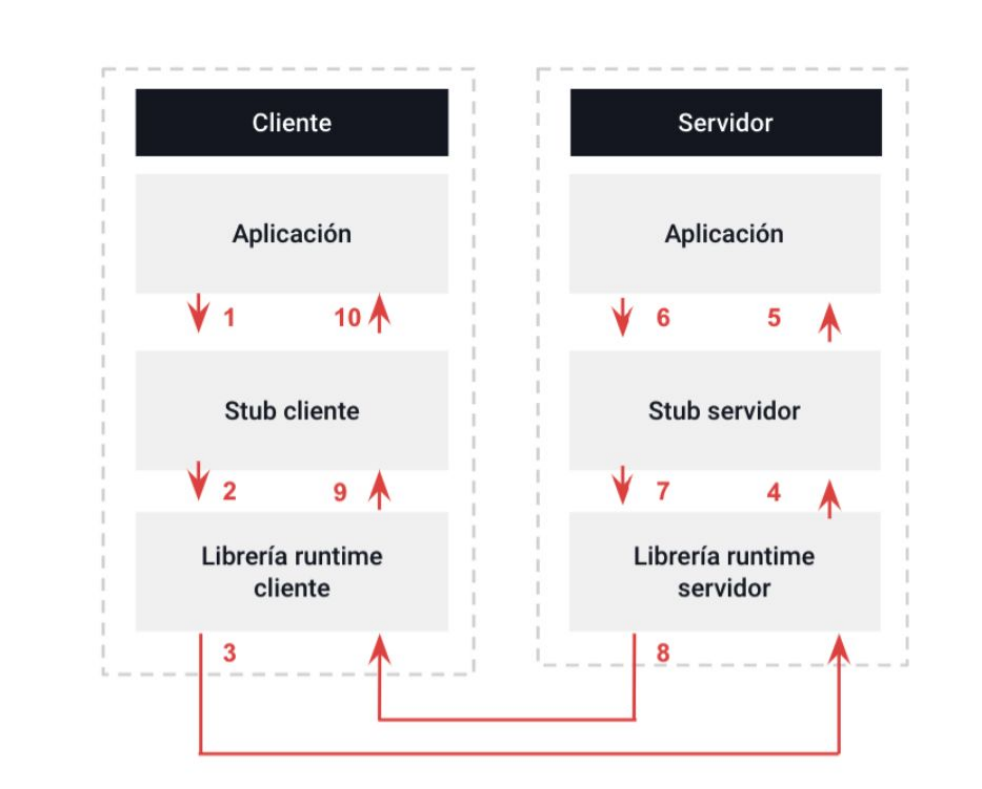
-Streaming bidireccional con transporte basado en HTTP/2

-Protocol Buffers para codificado de data

-Consistente a través de plataformas

-Bibliotecas para 12 lenguajes de programación

¿Cómo funciona?



Puntos fuertes

-Sencilla: llamada a una función.

-Flexible: soporte para 12 lenguajes.

-Alto rendimiento: más rápido que alternativas.

-Accesible: open source.

Puntos débiles

-Falta de información: comunidad pequeña.

-No llamadas a navegador: necesita proxy.

-Formato no leíble: Protocol Buffer pasa a binario.

-Difícil aprendizaje: Mayor complejidad que HTTP.

Conclusión

gRPC es una tecnología nueva que renueva RPC para solucionar problemas de REST.

Su rendimiento es más alto.

Por ser nueva su uso y comunidad no es tan amplio.

En un futuro podría ser la opción predominante en comunicaciones cliente-servidor.

Protocol buffer

¿Qué es y qué hace?

- Formato multiplataforma para serializar e intercambiar datos independientemente del lenguaje.
- También llamado Protobuf
- Desarrollado por Google
- Se basa en un **IDL**

¿Qué es un IDL?

- IDL: Interface Description Language = Lenguaje de Descripción de Interfaz

- Describen una interfaz que permite la comunicación entre programas desarrollados en lenguajes de programación diferentes.

Versiones más relevantes

-Creación: Inicios de 2001

-En 2008 se hace *open source*

-Protobuf 2.0: Contiene un generador de código para C++, Java, c# y Python

-Protobuf 3.0: Contiene un generador de código para JavaNano, Go, Objective-C y JavaScript (3.0.0-beta-2)

-También existen generadores implementados por terceros para los lenguajes Ballerina, C, C++, Dart, Elixir, Erlang, Haskell, JavaScript, Perl, PHP, Prolog, R, Rust, Scala, Swift, Julia y Nim.

-Actual (4-10-2022): 3.20.1, 22 de abril

Características

-Open Source

-Multiplataforma

-Diseño enfocado al rendimiento y la simplicidad

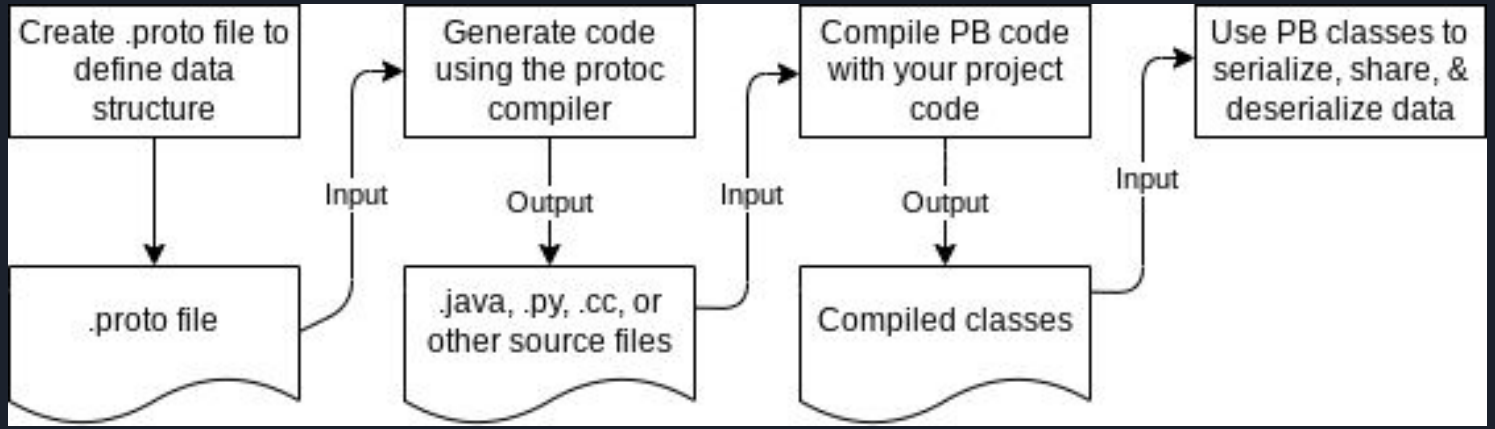
-Ofrece su propio gRCP

-El mensaje se serializa en formato *binary wire*

-La implementación oficial incluye un formato de serialización ASCII

-No está destinado a la lectura humana

¿Cómo funciona?



Puntos fuertes

-Multiplataforma y multilenguaje

-Diseñado con el rendimiento y la simplicidad en mente

-El formato *binary wire* es compatible en las dos direcciones

-La codificación es unas 2,5 veces más rápida que con Json y Json stream. Y la decodificación unas 5 veces más rápida

-Accesible: open source.

Puntos débiles

-No hay una forma de incluir o referirse a una especificación en particular

-Los bytes serializados no están encriptados por defecto, la encriptación debe realizarse después de serializar y, en el caso de enviar la información, los dos puntos deben tener el mismo protocolo/algoritmo de encriptación

-El formato *binary wire* no es autodescriptivo: no hay forma de interpretar la estructura sin una especificación externa.

-Los archivos .proto a veces generan un código muy poco práctico y difícil de debugar. Además resulta difícil de leer, puesto que no está diseñado para la lectura humana.

Conclusión

Aunque más rápido que sus competidores, resulta ser poco práctico en muchos casos

En consecuencia es poco usado

Es recomendable usarlo cuando necesitemos interoperabilidad entre lenguajes o necesitemos una serialización/deserialización rápida, y el código no necesite ser leído por alguien

Referencias

- https://www.altexsoft.com/blog/what-is-grpc/
- https://en.wikipedia.org/wiki/GRPC
- https://grpc.io/
- https://en.wikipedia.org/wiki/Protocol_Buffers
- https://developers.google.com/protocol-buffers/docs/overview
- https://www.adaptiv.co.nz/protobuf-what-is-it-why-you-should-care-and-when-should-you-use-it/
- https://mnwa.medium.com/what-the-hell-is-protobuf-4aff084c5db4

Reparto de tareas

- Adrián Braojos: Ha hecho las secciones de “¿Cómo funciona? “y “Conclusión” del Protocol Buffer y ha añadido algo de información. Ha pasado la parte de Protocol Buffer a la presentación.

- Marc Duch: Ha reunido la mayor parte de información sobre Protocol Buffer.

- Haopeng Lin Ye: Ha reunido una parte significativa de información sobre gRPC y comenzado la parte de gRPC de la presentación.

- Miguel Moreno: Ha añadido información extra que faltaba sobre gRPC y revisado y completado la parte de gRPC de la presentación.