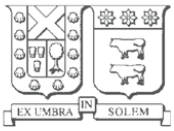
Departamento de Informática
Universidad Técnica Federico Santa María



Informe Comparación: GRPC vs RabbitMQ

Profesor: Mauricio Solar
Ayudante: Humberto Farías

Camilo Núñez F.
Hugo Sepúlveda A.

4 de enero de 2020

1. Protocolos

1.a. RPC: GRPC

Un sistema RPC (Remote Procedure Call) consiste en procedimientos/funciones que pueden ser llamados tanto de una máquina externa como de la misma donde corre el programa, permitiendo una gran transparencia como sistema distribuido. Los elementos más importantes son la programación con interfaces de los servicios, lo que permite la interacción distribuida de las máquinas; una semántica asociada a las llamadas de las funciones, estableciéndose los parámetros correctos de input/output y estandarizando la comunicación; y la transparencia de esta implementación, que oculta el paso de mensajes en las funciones. Las principales características de este tipo de sistemas son la heterogeneidad de los servidores y clientes, la no diferenciación de problemas en el lado del cliente o servidor e incertidumbre en la localización de los recursos. Estos aspectos pueden verse como una ventaja o problema, dependiendo del contexto en el que se ocupe.

La librería GRPC es un gran ejemplo de framework para construir este tipo de sistemas, pues tiene una fácil creación de los servicios y funciones con alta transparencia para las llamadas a objetos y métodos. Para crear la conexión y permitir el consumo de servicios del lado del cliente se utiliza un objeto *stub* (canales), principal vía de comunicación cliente-servidor.

También abarca otras herramientas que facilita el manejo y llamadas de funciones tales como:

- Llamadas unarias (un objeto) o streams (conjuntos de objetos).
- Sincronismo y asincronismo.
- Deadlines y timeouts.
- Interceptores y manejadorses (interceptors y handlers).

1.b. AMQP: RabbitMQ

El Advanced Message Queuing Protocol (AMQP) es un estándar para envío de mensajes entre aplicaciones. Este estándar funciona por medio de tres actores y un objeto: un mensaje, un productor que crea un mensaje y lo envía, un brker de mensajería que distribuye el mensaje de acuerdo con reglas definidas en diferentes colas, y un consumidor (consumer) que recupera el mensaje de la cola y lo edita.

Bajo este contexto, RabbitMQ es un bróker que utiliza el estándar AMQP 0-9-1, el cual permite que las aplicaciones cliente se comuniquen con los middleware brókers de mensajería de formas más eficiente en las colas.

1.c. Comparación

Los RPC han sido diseñados para otras necesidades no enfocadas en los mensajes, sino en el compartir recursos y mejor coordinación entre las máquinas pertenecientes a una red, en cambio AMQP sí tiene un enfoque hacia los mensajes, pues el estándar define todos los elementos del sistema que están en una comunicación constante a través de mensajes y canales definidos. Es por ello, que para microservicios de mensajería se debe optar por RabbitMQ (AMQP), debido a mejores métodos y estructuras que están hechas especialmente para lo pedido, al contrario de la llamadas a métodos u objetos externos.

Bajo este contexto, RabbitMQ tiene distintas implementaciones orientadas al de desarrollo aplicaciones de mensajería, como la utilización de multicanales (no existentes en GRPC), dados por la función `queue_bind`, la cual brinda la opción de realizar un envío de broadcasting a los canales que se indiquen; y las funciones asociadas al uso de `exchanges` como `exchange_declare` y `queue_declare`, las cuales fueron cruciales para el ruteo de mensajes entre los canales de los clientes y el servidor.

Finalmente, RabbitMQ tiene múltiples wrappers y adapters, a diferencia GRPC, que permiten usar librerías como `asyncio` de Python:3 y `Tornado`, las cuales se integran mejor con características de RabbitMQ como loops internos y envío de mensajes asíncronos.