Informe de Sistemas Distribuidos

Tarea 2

Sebastián Jara 201573523-9

Gabriel Astorga 201573591-3

Resumen

¿que es gRPC? es un sistema de llamada a procedimiento remoto (RPC) de código abierto desarrollado inicialmente en Google. Utiliza Como transporte HTTP/2 y Protocol Buffers como lenguaje de descripción de interfaz. Proporciona características como autenticación, transmisión bidireccional y control de flujo, enlaces bloqueantes o no bloqueantes, cancelaciones y tiempos de espera. ¿y RabbitMQ? es un software de negociación de mensajes de código abierto, y entra dentro de la categoría de middleware de mensajería.

1) Comparativa

Rabbit implementa un sistema de colas, y según nuestra configuración para lograr el objetivo, cada cliente y el servidor tienen su propia cola, donde los mensajes que llegan al servidor se procesan asíncronamente y se le reenvía el mensaje correspondiente al cliente correspondiente (llegando a su cola), donde su ventaja es que los mensajes no se pierden por no estar conectado al momento de que les haya llegado el mensaje, quedando guardados en un buffer hasta su entrega, lo que también entrega una desventaja ya que el colapso del buffer (200 mb aprox) significa la pérdida de todos los mensajes. gRPC se basa en llamar funciones remotas, dejando las operaciones computacionales al servidor, el fuerte tipado de los datos es su ventaja ya que se declara un objeto con ciertos tipos de características (ejem. string perro, int32 patas) y luego puede ser accedido fácilmente por varios lenguajes de programación, facilitando la comunicación de microservicios.

2) Conclusiones

La recomendación final es de usar gRPC, a pesar de las ventajas ya mencionadas de Rabbit para una mensajería real-time con un respaldo de datos, el tipado de gRPC lo hace vencedor, ya que elimina la redundancia y facilita la conección en de microservicios híbridos, además de lograr manejar de mejor manera la recepción de mensajes, ya que se sabe siempre qué es lo que va a llegar o más bien lo que debería llegar, el tema de manejar la permanencia de mensajes, se puede controlar con el log y la asincronicidad también es controlado con un threadpool por lo que sus flancos débiles pueden ser solventados con algo de programación, además de ser más fácil de dockerizar para cuándo se necesita hacer algo de Extrem Programing.