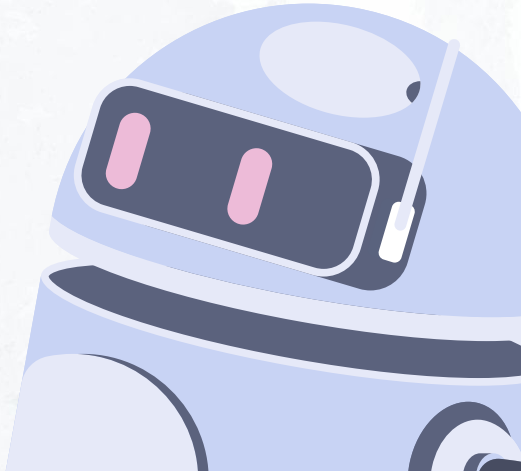
GraphQL y gRPC

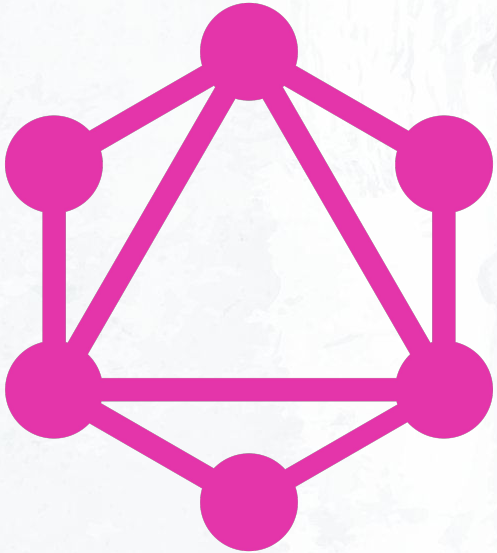


Análisis y diseño de aplicaciones I.
Cabrera, Ferreira, Glass, Lorenzo, Maidana, Rama

Índice

01 →

¿Qué es GraphQL?



GraphQL

GraphQL

Es un lenguaje de consulta y manipulación de datos desarrollado por Facebook y lanzado en 2015. Diseñado para que clientes HTTP puedan realizar llamadas a la API y obtener los datos que necesitan del backend.

También permite la mutación de datos, lo que permite modificar o crear nuevos datos en el servidor.

¿Cómo surge GraphQL?

Limitaciones de las API Rest

Se desarrolló internamente en Facebook para solucionar los problemas de:

(-) **UnderFetching:** Se necesitan muchas llamadas a la API para poder obtener los datos necesarios

(+) **OverFetching**: Se obtienen muchos datos que no son necesarios.

Diferencias entre GraphQL y API REST

(+) Sintaxis y estructura ⟶

GraphQL utiliza un solo endpoint y sintaxis de consulta para solicitar y recibir datos

(+) OverFetching y UnderFetching ⟶

Resuelve estos problemas haciendo consultas dinámicas.

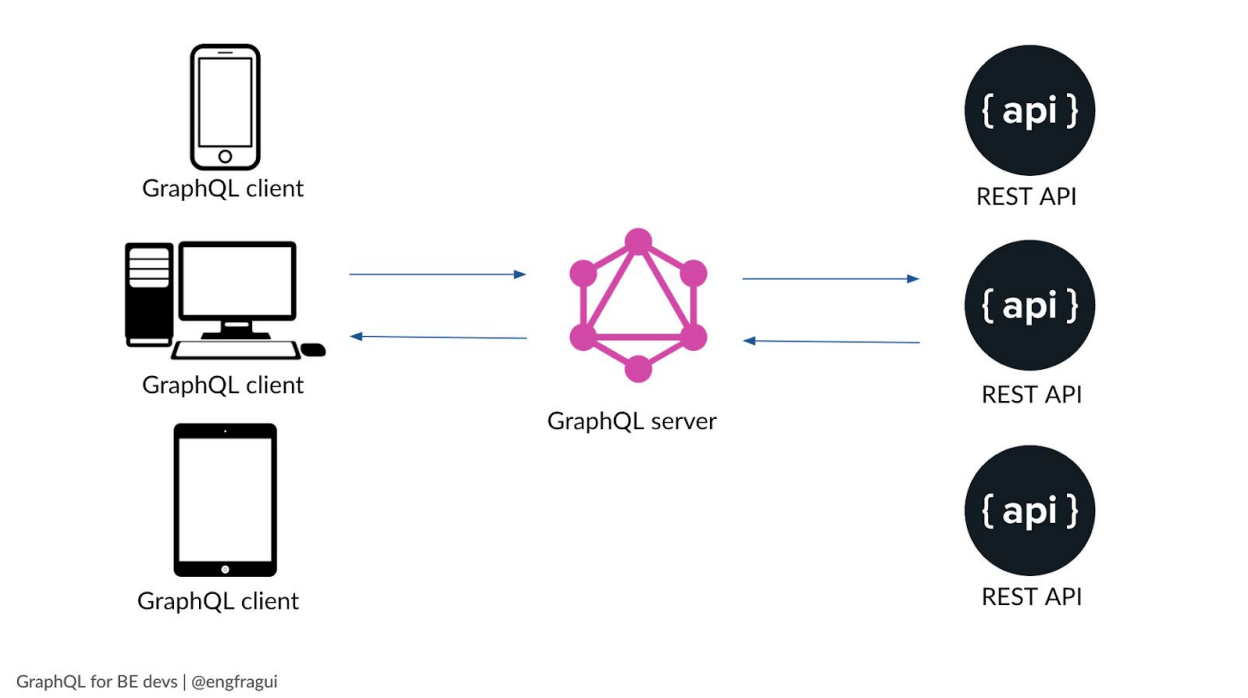
(+) Compatibilidad ⟶

Permite realizar cambios sin afectar a los clientes existentes.

(+) Rendimiento ⟶

GraphQL resulta más eficiente en escenarios complejos. REST cuando son específicos y estáticos.

GraphQL puede integrarse con API REST



Flujo de GraphQL

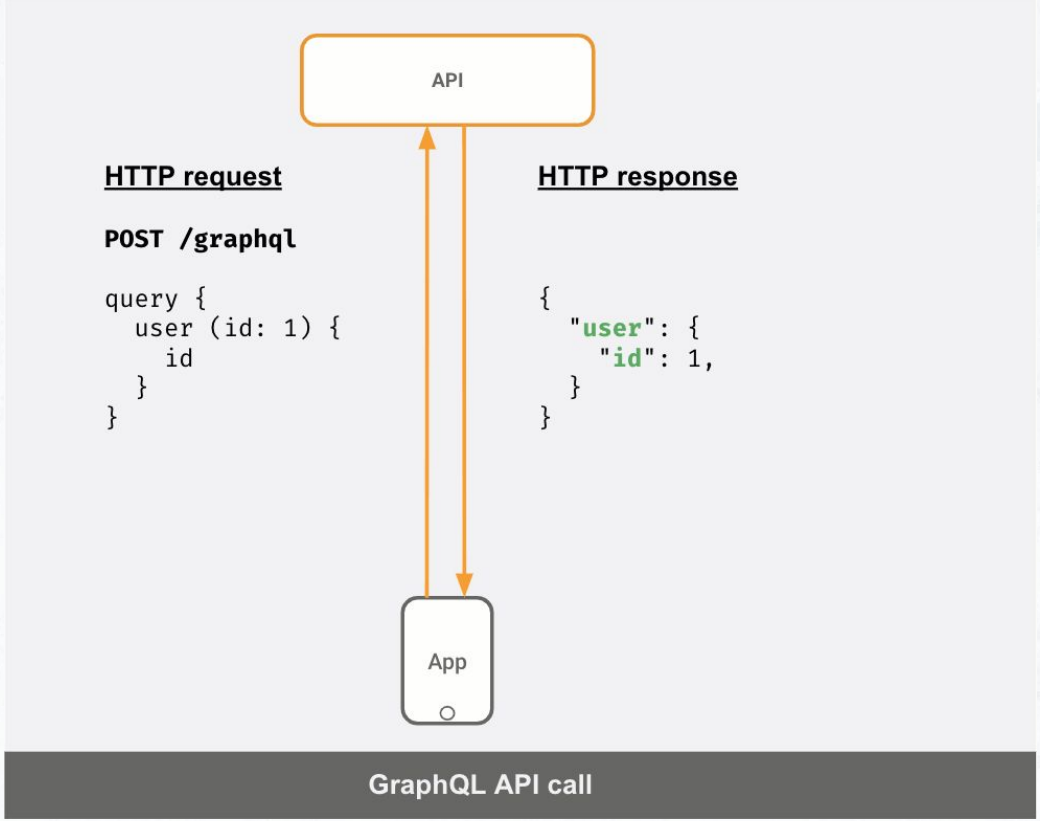
1. El cliente realiza la consulta al servidor con un JSON (al igual que API REST) indicando los datos que solicita.

2. El servidor obtiene el objeto JSON y extrae la cadena de consulta. Según la sintaxis GraphQL y el esquema, el servidor procesa y valida la consulta.

3. El servidor solicita las llamadas a la base de datos y a los servicios necesarios para obtener los datos.

4. Finalmente el servidor devuelve el objeto JSON con los datos solicitados.

Flujo de GraphQL



02 →

¿Qué es gRPC?

(gRPC)

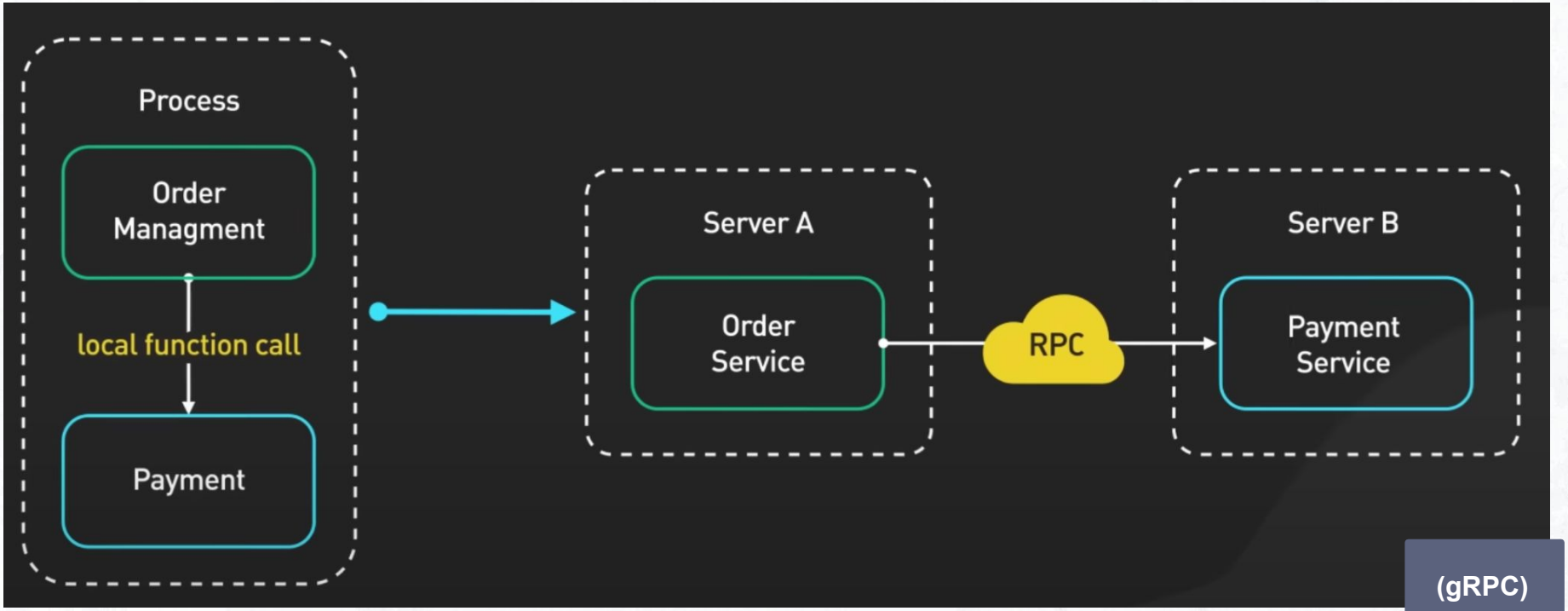
google

Remote

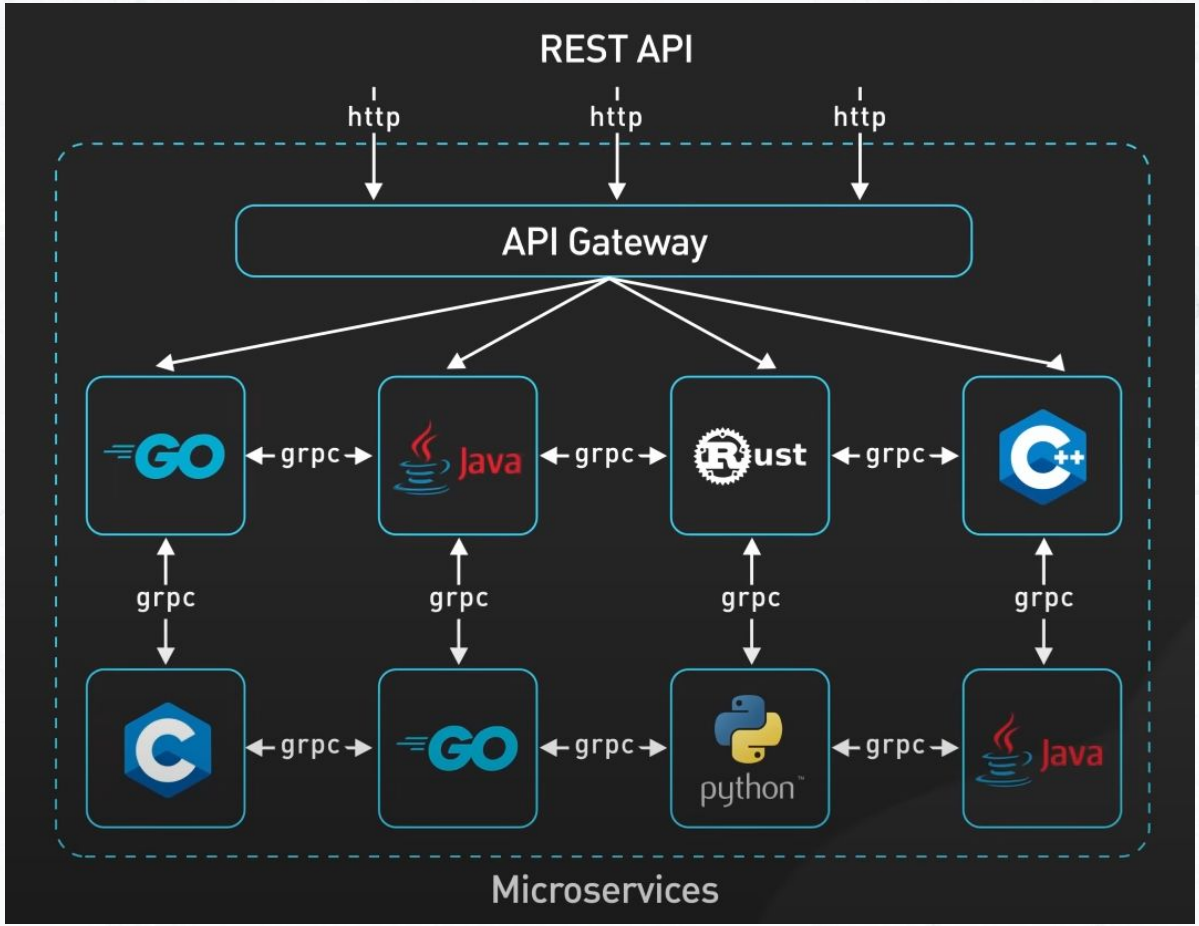
Procedure

Call

Local Procedure Call vs Remote Procedure Call

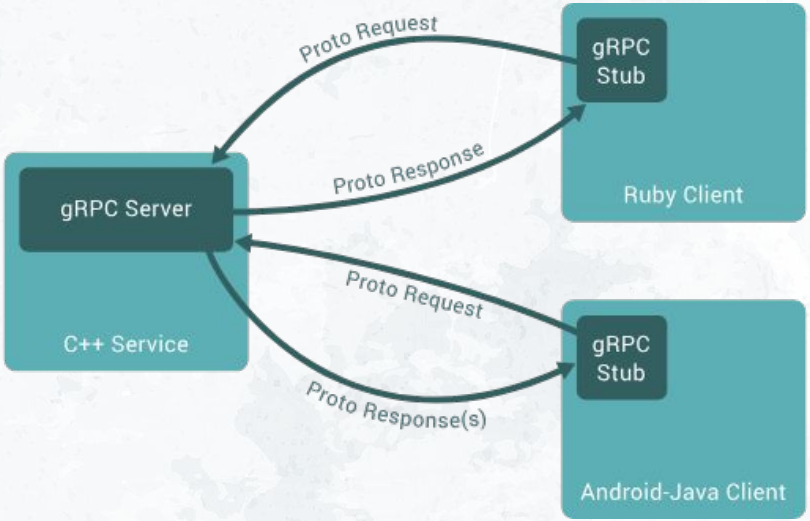


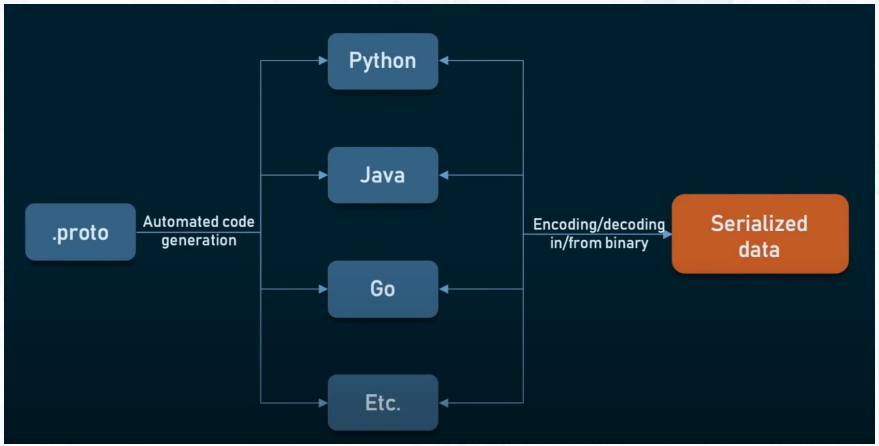
Ecosistema de desarrollo fácil


REST API
http
http
http
API Gateway
GO
grpc
Java
grpc
Rust
grpc
C++
grpc
grpc
grpc
grpc
C
grpc
GO
grpc
python
grpc
Java
Microservices

Protocol Buffers

- Serialización Eficiente
- Definición de interfaces
- Generación de código
- Versionado y evolución
- Compatibilidad con múltiples lenguajes






```
message Person {
  optional string name = 1;
  optional int32 id = 2;
  optional string email = 3;
}
```

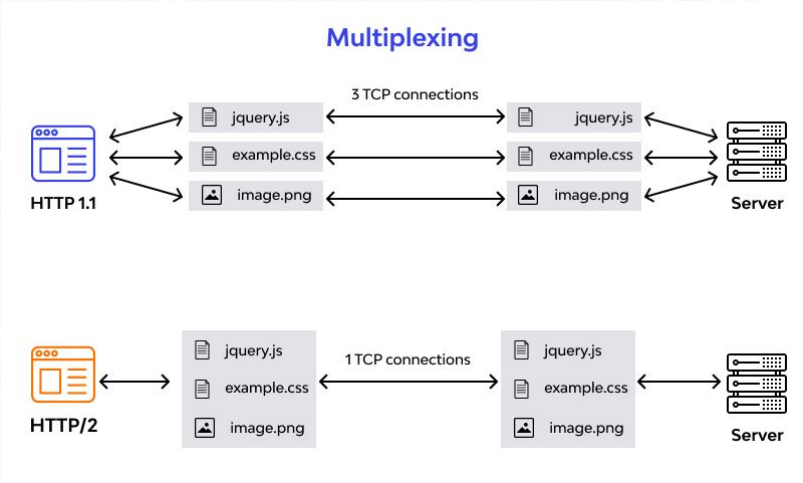
Binary Encoding

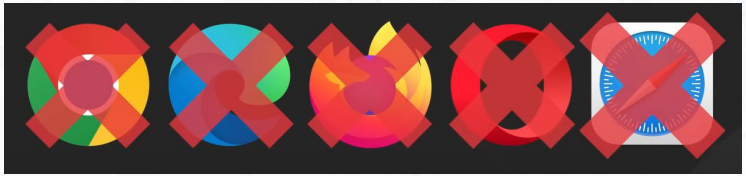
Binary Encoding - Protobuf

type	field tag	length	value
0b string	00 01	00 08	6B 65 79 62 6F 61 72 64 k e y b o a r d
0a i64	00 02	1	
0a i64	00 03	100	

```
JSON
{
“productName”: “keyboard”,
“quantity”: “1”,
“price”: “100”
}
```

HTTP/2





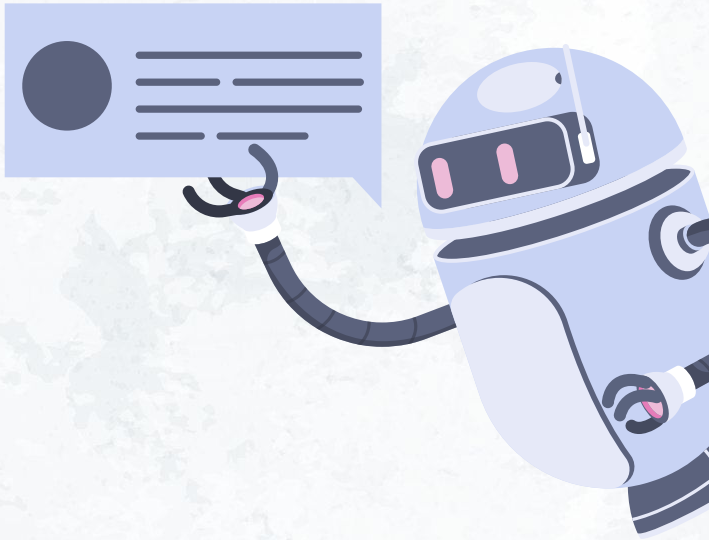
03 →

Demo gRPC

https://github.com/Cglassm/gRPC_demo

(gRPC)

¿Preguntas?



Muchas garcias!