gRPC

Programación Distribuida y Tiempo Real

¿Qué es gRPC?

gRPC

Conceptos generales

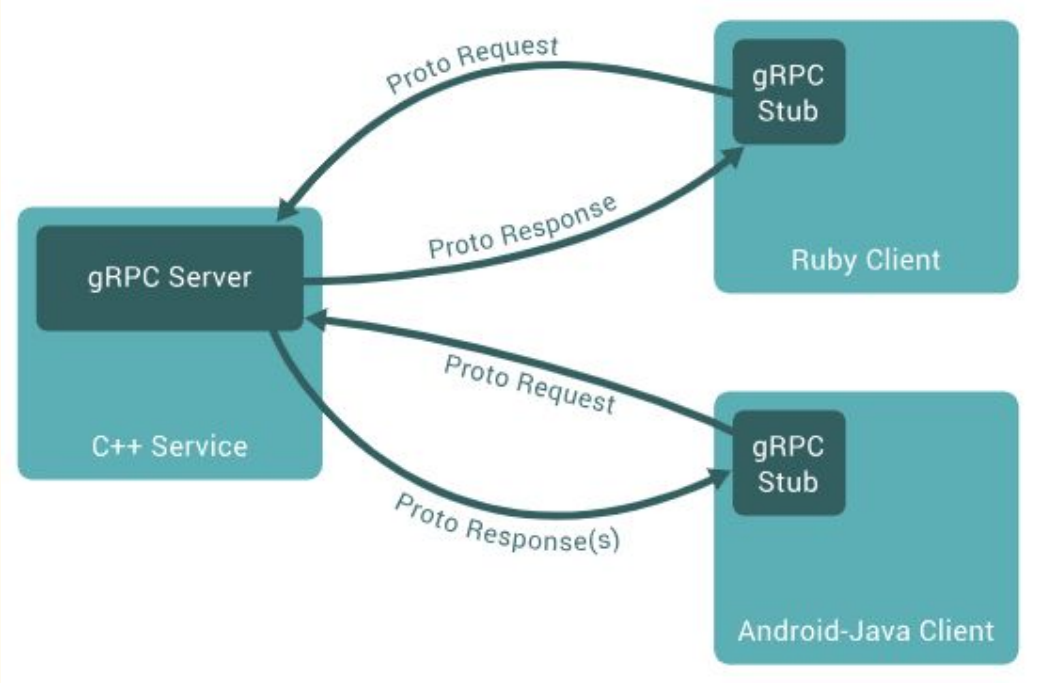
A language-neutral, platform-neutral remote procedure call (RPC) framework and toolset developed at Google

gRPC

Conceptos generales

- Permite definir un servicio utilizando Protocol Buffers (protobuf)
- Permite generar clientes idiomáticos y servidores stubs desde la definición del servicio en varios lenguajes (Android Java, C#/.Net, Kotlin/JVM, Go, Node.js, PHP, Python, etc)

Esquema cliente/servidor



Workflow gRPC

gRPC Workflow

en PDyTR

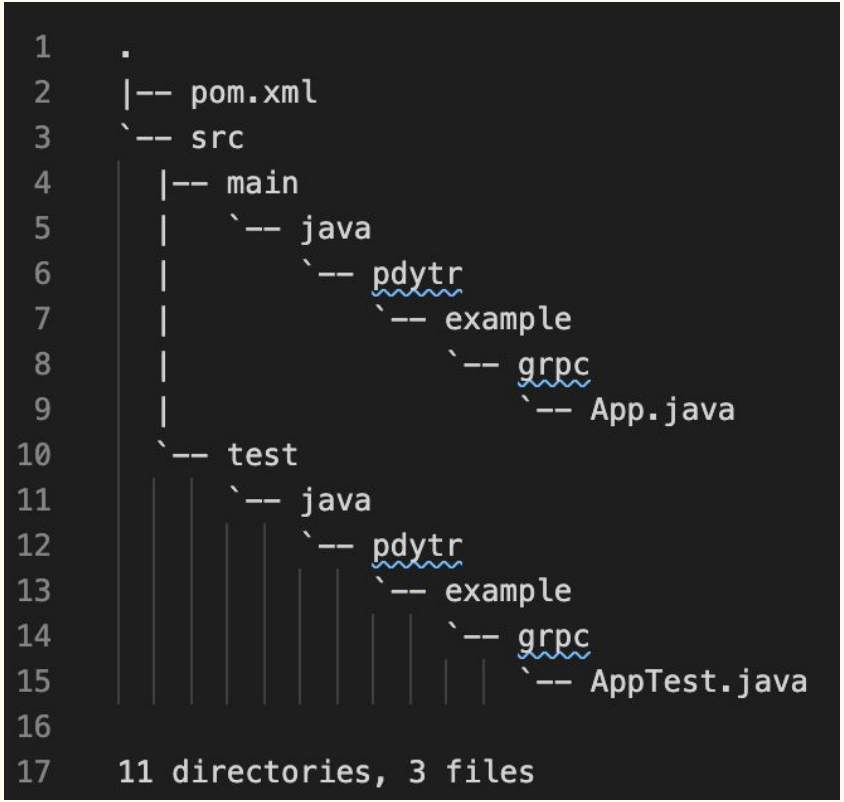
- Utilizar Maven
  - Generador de los proyectos
  - Instalador de paquetes
  - Compilador
  - Ejecutor
- Con el proyecto generado, se crea el `.proto`

gRPC Workflow

Crear proyecto

```
# Crear un nuevo proyecto
mvn archetype:generate \
-DgroupId=pdytr.example.grpc \
-DartifactId=grpc-hello-server \
-DarchetypeArtifactId=maven-archetype-quickstart \
-DinteractiveMode=false
```

Árbol de archivos generados



Dependencias maven

```xml
<dependency>
  <groupId>io.grpc</groupId>
  <artifactId>grpc-netty</artifactId>
  <version>1.7.0</version>
</dependency>
<dependency>
  <groupId>io.grpc</groupId>
  <artifactId>grpc-protobuf</artifactId>
  <version>1.7.0</version>
</dependency>
<dependency>
  <groupId>io.grpc</groupId>
  <artifactId>grpc-stub</artifactId>
  <version>1.7.0</version>
</dependency>
<dependency>
  <groupId>javax.annotation</groupId>
  <artifactId>javax.annotation-api</artifactId>
  <version>1.2</version>
</dependency>
```

Build

```
<build>
  <extensions>
    <extens
      <grou
      <arti
      <vers
    </exten
  </extensi
  <plugins>
    <plugin
      <grou
      <arti
      <vers
      <conf
        <so
        <ta
      </con
    </plugi
```

```
<plugin>
  <groupId>org.xolstice.maven.plugins</groupId>
  <artifactId>protobuf-maven-plugin</artifactId>
  <version>0.5.0</version>
  <configuration>
    <protocArtifact>com.google.protobuf:protoc:3.4.0:exe:${os.detected.classifier}</protocArti
    <pluginId>grpc-java</pluginId>
    <pluginArtifact>io.grpc:protoc-gen-grpc-java:1.7.0:exe:${os.detected.classifier}</pluginAr
  </configuration>
  <executions>
    <execution>
      <goals>
        <goal>compile</goal>
        <goal>compile-custom</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

gRPC Workflow (Generar *.proto)

```
# creamos el directorio para el .proto
root@31945f6e4199:/pdytr# mkdir -p grpc-hello-server/src/main/proto
# creamos el .proto
root@31945f6e4199:/pdytr# touch grpc-hello-server/src/main/proto/GreetingService.proto
```

Protocol Buffer (GreetingService.proto)

```
syntax = "proto3";
package pdytr.example.grpc;

message HelloRequest {
   string name = 1;
}
message HelloResponse {
   string greeting = 1;
}
service GreetingService {
   rpc greeting(HelloRequest) returns (HelloResponse);
}
```

Protocol Buffer (GreetingService.proto)

```
# Estos son los archivos generados cuando se ejecuta después de haber generado el .proto
“mvn -DskipTests package”
root@31945f6e4199:/pdytr/grpc-hello-server# mvn -DskipTests package
root@31945f6e4199:/pdytr/grpc-hello-server# find target/generated-sources -name '*.java'
target/generated-sources/protobuf/grpc-java/pdytr/example/grpc/GreetingServiceGrpc.java
target/generated-sources/protobuf/java/pdytr/example/grpc/GreetingServiceOuterClass.java
```

gRPC Implementación

Implementación

Con los archivos anteriores

- Crear el Server (en este caso se llamará App.java)
- Crear el Client

Ver ejemplo

Codegen and Run w/ Maven

```
# Correr el servidor en Terminal 1
root@31945f6e4199:/pdytr/grpc-hello-server# mvn -DskipTests package exec:java
-Dexec.mainClass=pdytr.example.grpc.App
Server started
```

```
# Correr el cliente en Terminal 2
root@31945f6e4199:/pdytr/grpc-hello-server# mvn -DskipTests exec:java
-Dexec.mainClass=pdytr.example.grpc.Client
```

Resultados

```
[INFO]
[INFO] --- exec-maven-plugin:3.0.0:java (default-cli) @ grpc-hello-server ---
Server started
name: "Ray"
```

docker (com.docker.cli container exec -it --user root pdytr bash)

```
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- exec-maven-plugin:3.0.0:java (default-cli) @ grpc-hello-server ---
greeting: "Hello there, Ray"
```

¡Muchas gracias!

¿Preguntas?

Links útiles

Google mini lab in GCP with gRPC and Java

- Building a gRPC service with Java

gRPC Concepts

- https://grpc.io/docs/what-is-grpc/core-concepts/

Gist para instalar Maven en Docker

- https://gist.github.com/gmaron/0b32b2b1e078998ef54a481d84817db9#file-pdytr-exec-update-docker-container