gRPC

Comment communiquer en gRPC avec Quarkus

Niusha Maxime PAKSERESHT - 24 Janvier 2024 - Montréal JUG

gRPC ?

Remote **P**rocedure **C**all développé par Google

D’autres protocoles existent

- RMI (Java **R**emote **M**ethod **I**nvocation) => TCP/IP
- SOAP (**S**imple **O**bject **A**ccess **P**rotocol) => HTTP XML
- REST (**R**epresentational **S**tate **T**ransfert) => HTTP JSON

gRPC => HTTP 2 / **Données binaires**

HTTP 2

- Multiplexing
- Streaming
- Gain en performance

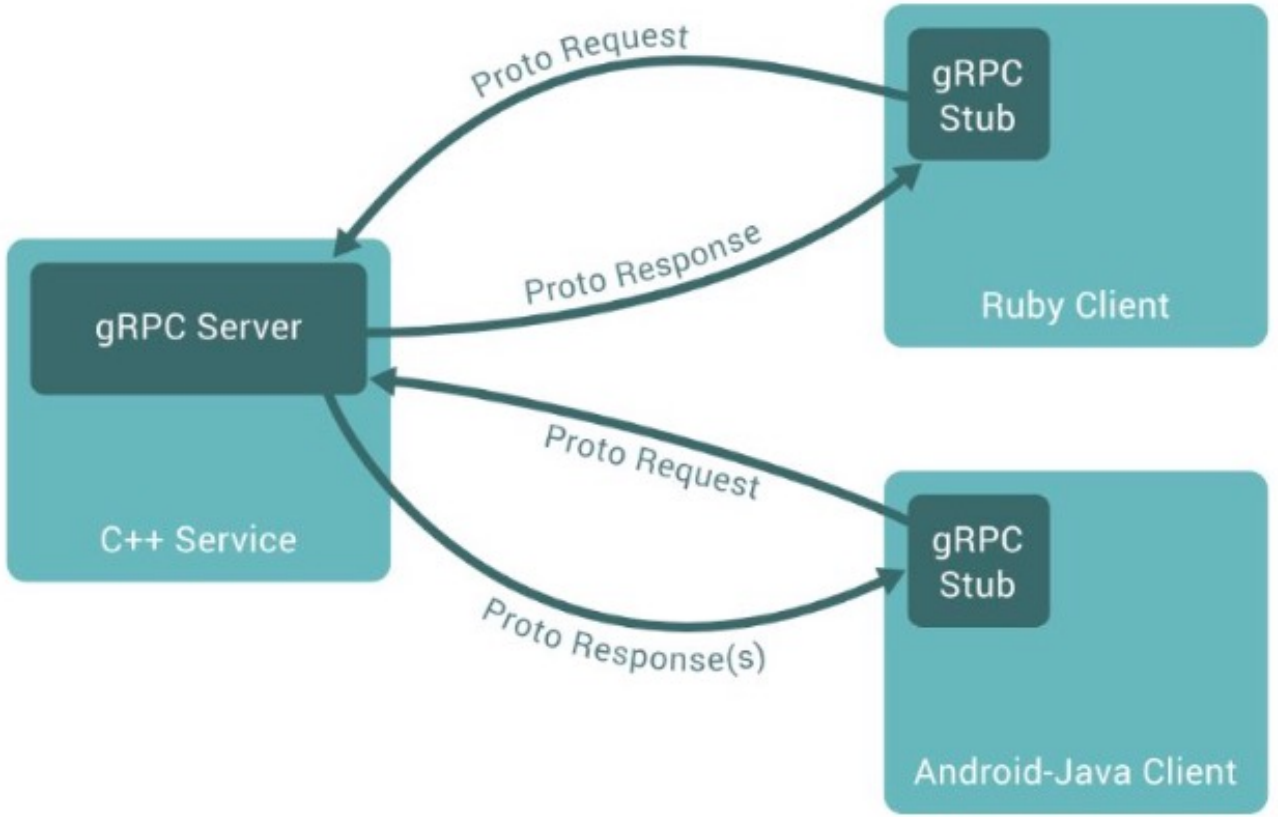
Comment ça marche

- protobuf : **I**nterface **D**efinition **L**anguage

```
service TrainPositionService {
  rpc GetPosition(GetPositionRequest) returns (TrainPosition);
  rpc ReportPosition (PositionRequest) returns (PositionResponse);
}

message GetPositionRequest {
  int64 trainId = 1;
}

message TrainPosition {
  int64 trainId = 1;
  int64 trackId = 2;
  float kilometer = 3;
}

message PositionRequest {
  int64 trainId = 1;
  int64 trackId = 2;
  float kilometer = 3;
}

message PositionResponse {
  TrainPosition actualTrain = 1;
  TrainPosition previousTrain = 2;
  TrainPosition nextTrain = 3;
}
```

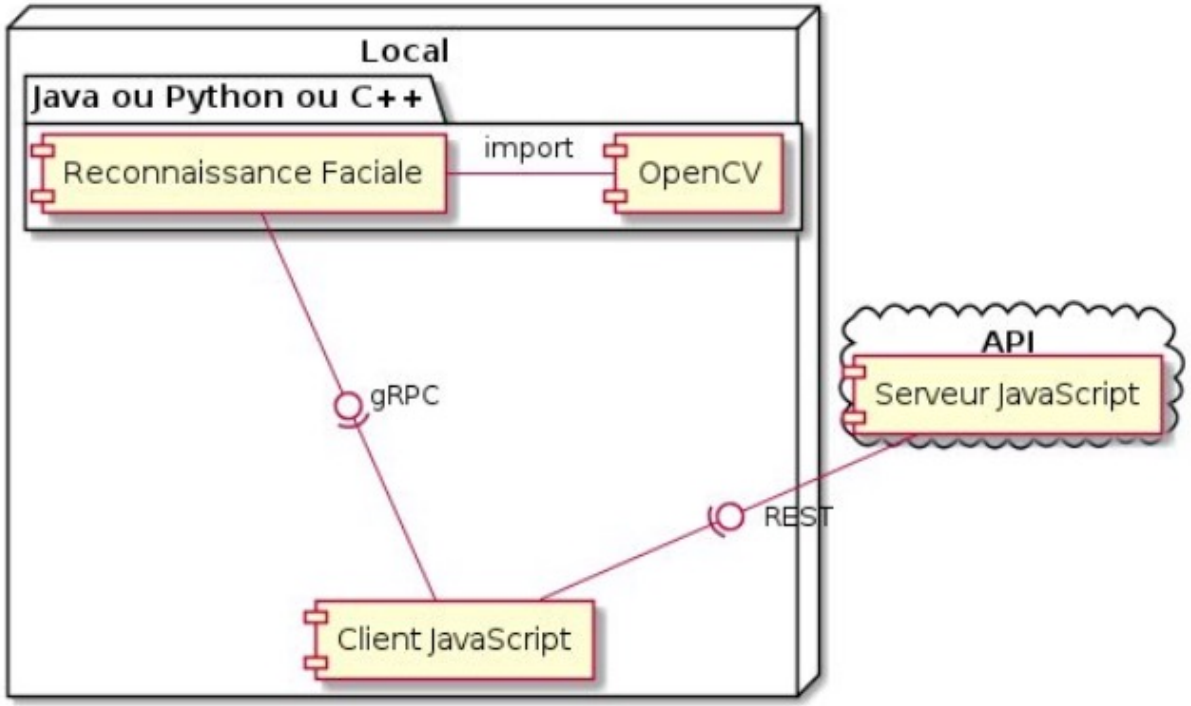
Comment ça marche

- Permet de générer un client / serveur pour le langage ciblé



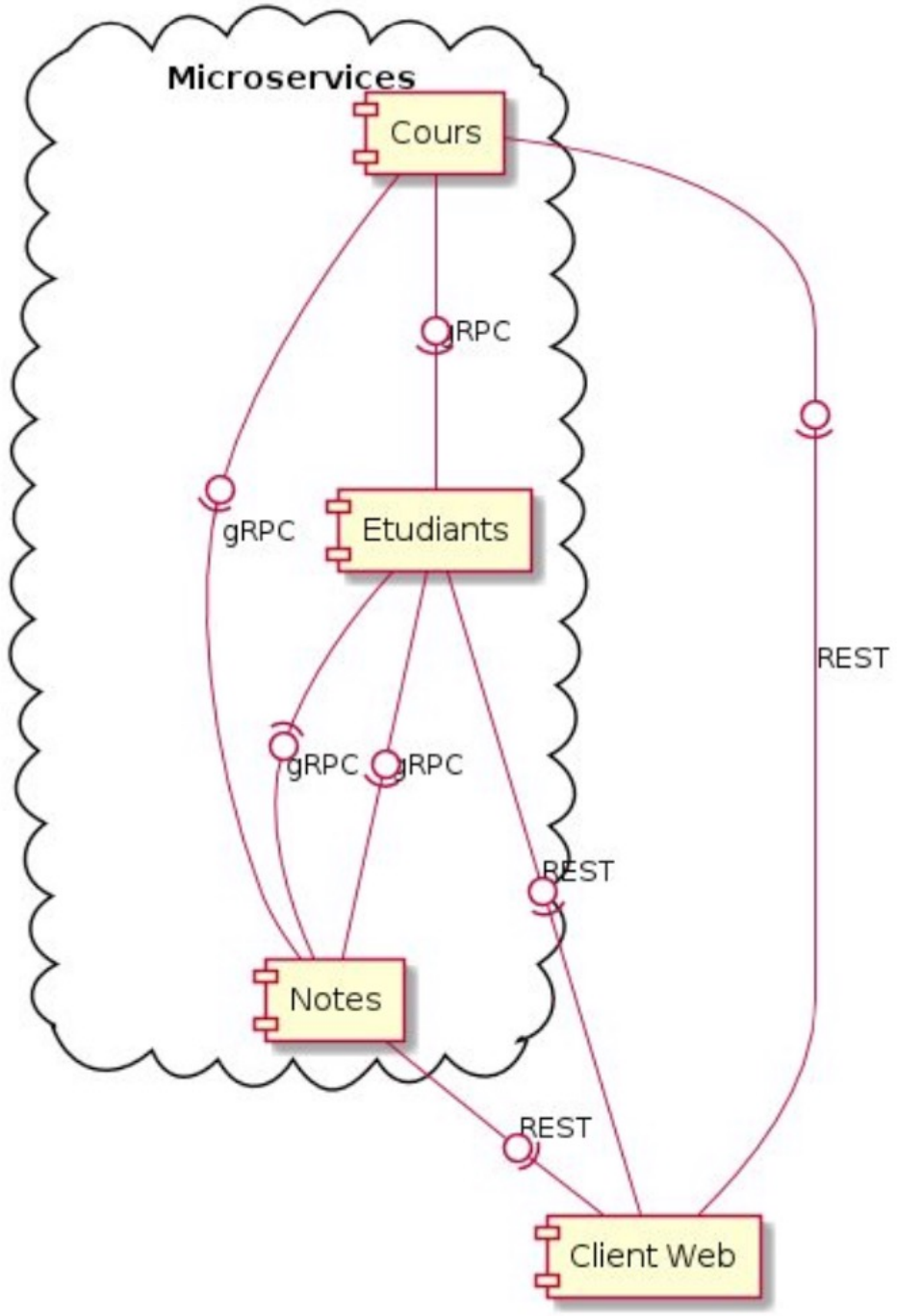
Exemple d’utilisation

- Intégration d’une bibliothèque de reconnaissance d’images dans un langage tiers



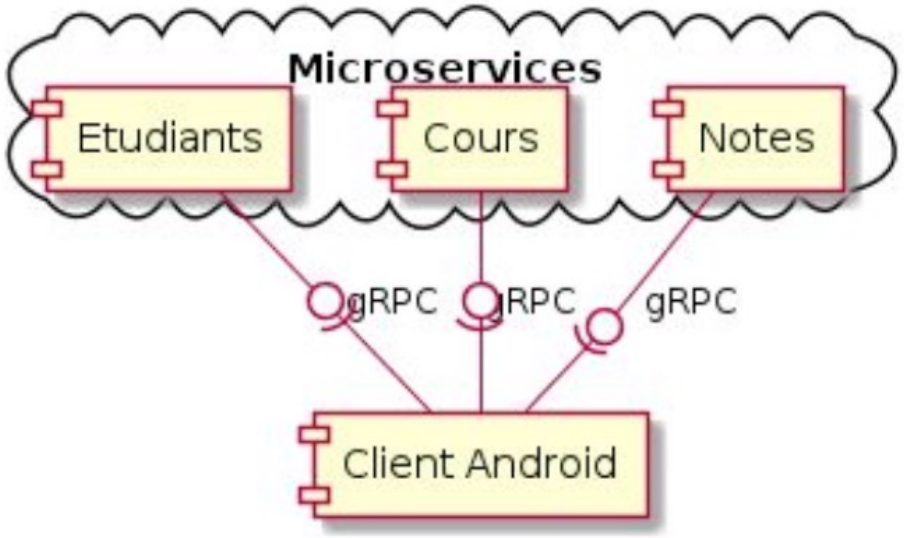
Exemple d’utilisation

- Communication interne dans une architecture micro-services
- Exposition API externes en REST
- Parfait pour du service à service



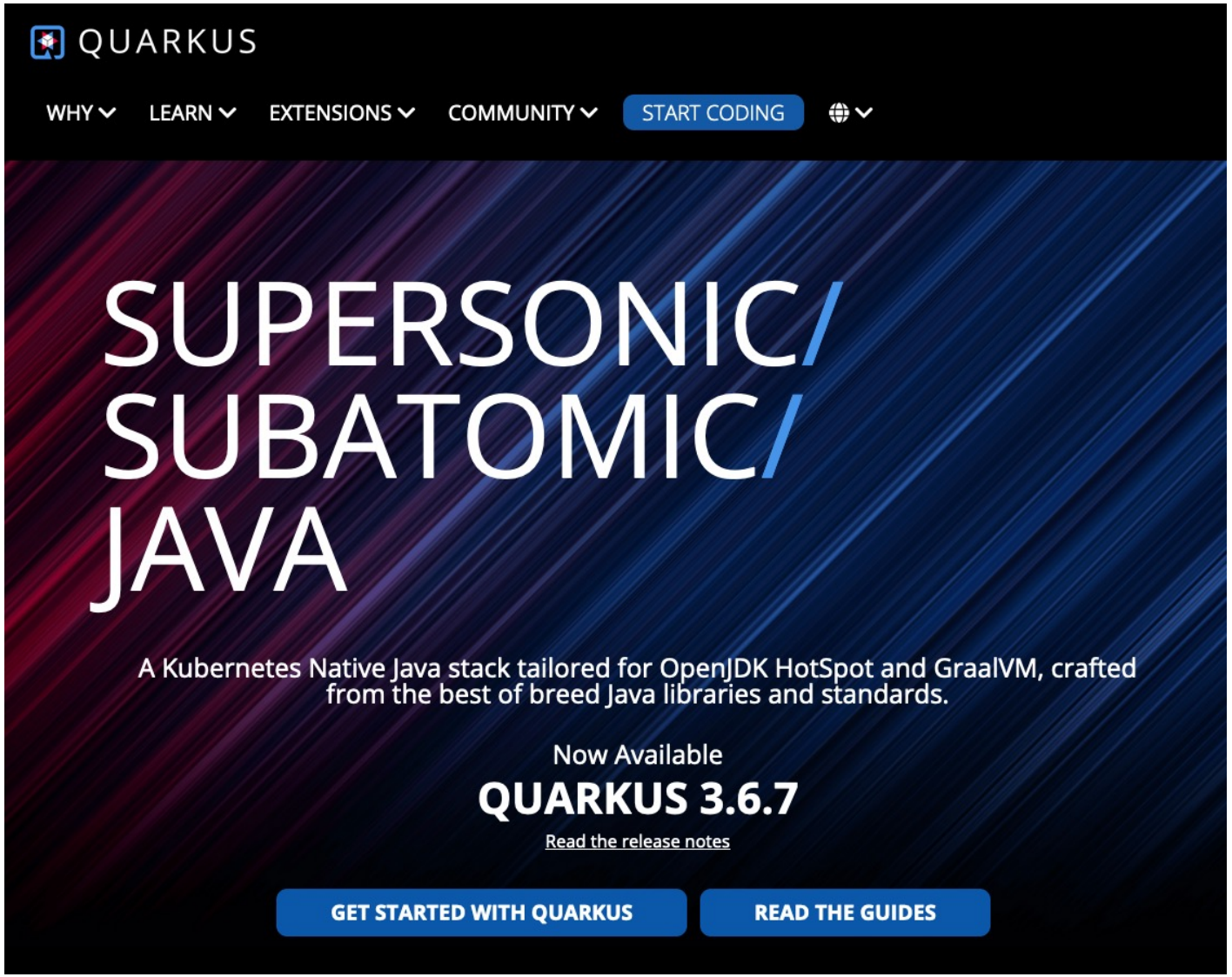
Exemple d’utilisation

- Communication avec application mobile Android / iOS

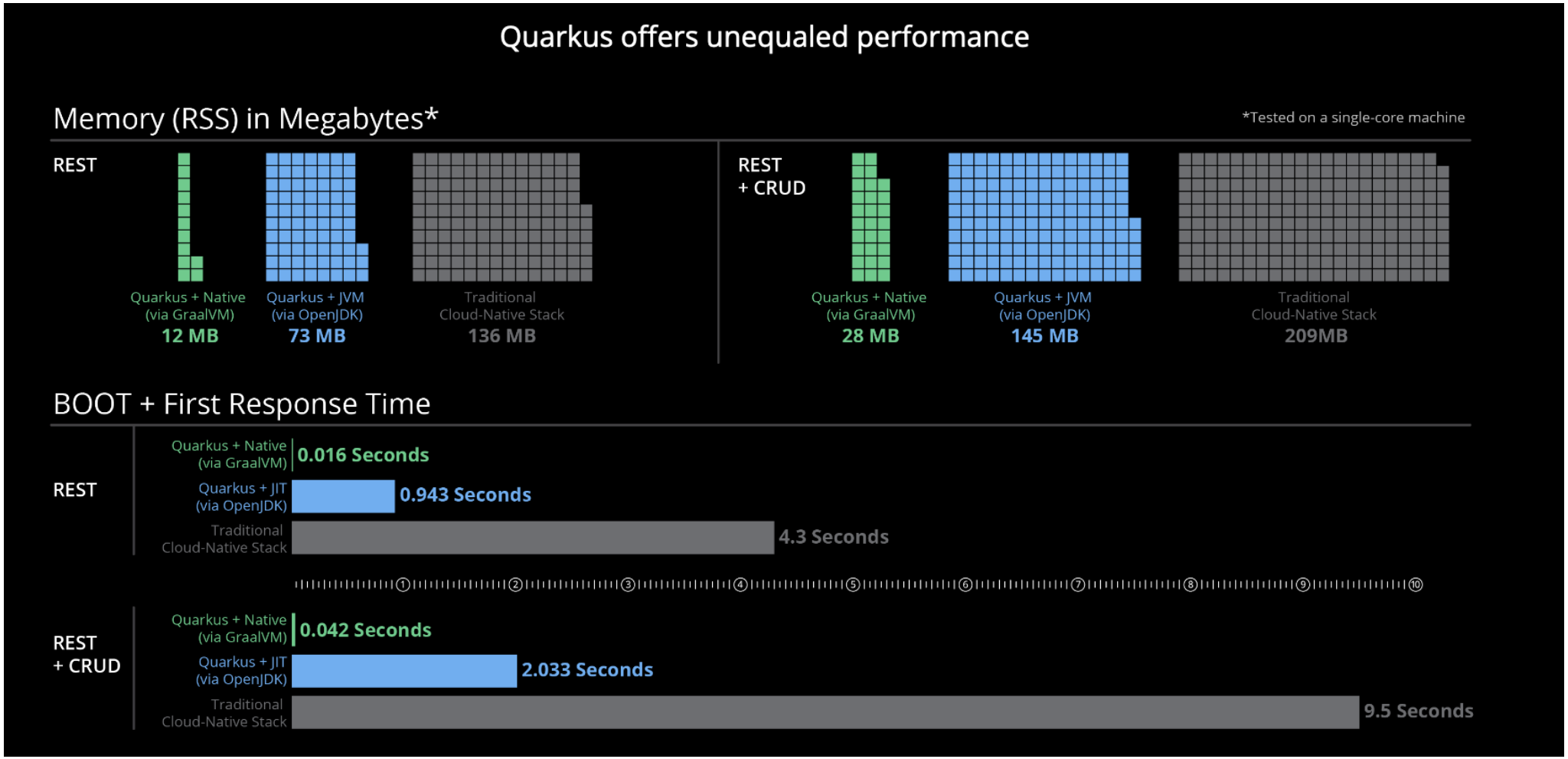


Quarkus

- Alternative à Spring
  - Plus légère
- Temps de démarrage plus rapide
  - Parfait pour des micro applications sur le cloud
- Compilation native
- Système d’extensions
- Standards Java

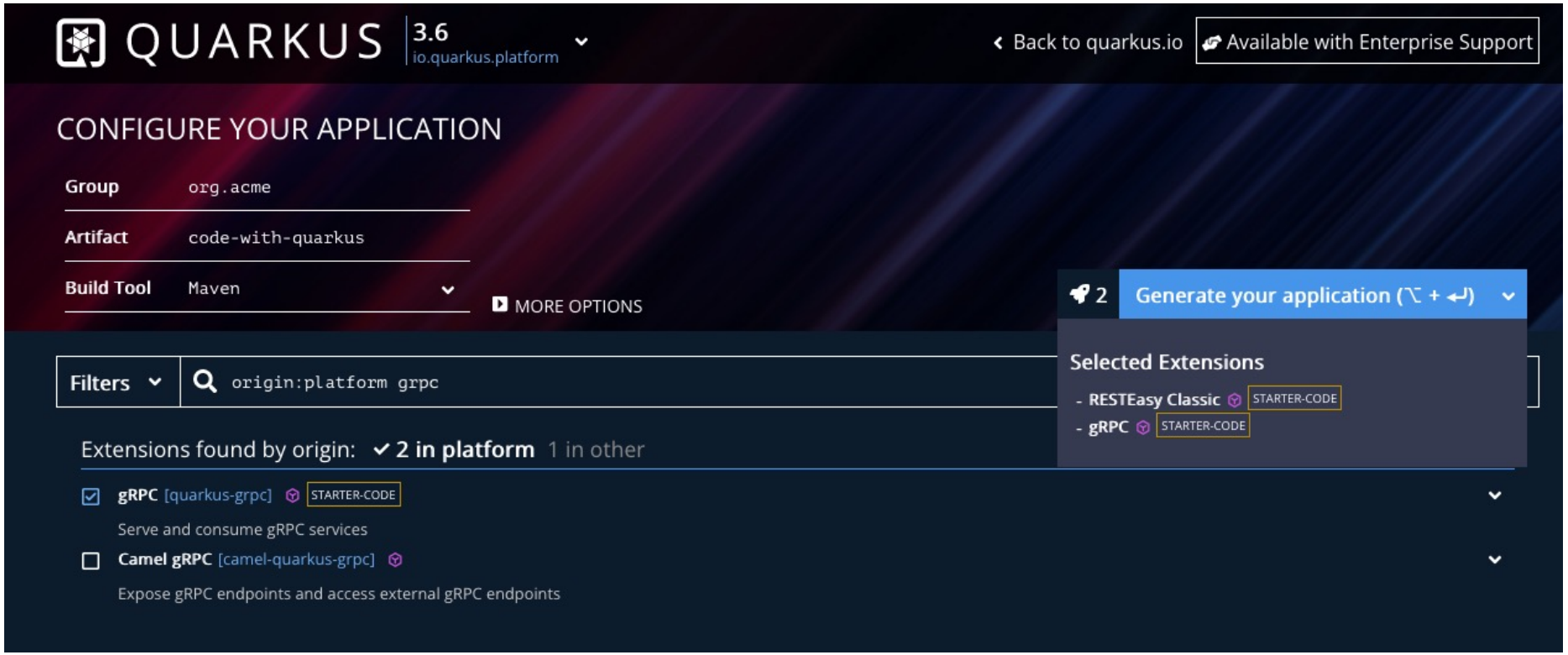


Quarkus


Quarkus offers unequaled performance
Memory (RSS) in Megabytes*
*Tested on a single-core machine
REST
Quarkus + Native (via GraalVM)
12 MB
Quarkus + JVM (via OpenJDK)
73 MB
Traditional Cloud-Native Stack
136 MB
REST + CRUD
Quarkus + Native (via GraalVM)
28 MB
Quarkus + JVM (via OpenJDK)
145 MB
Traditional Cloud-Native Stack
209MB
BOOT + First Response Time
REST
Quarkus + Native (via GraalVM)
0.016 Seconds
Quarkus + JIT (via OpenJDK)
0.943 Seconds
Traditional Cloud-Native Stack
4.3 Seconds
REST + CRUD
Quarkus + Native (via GraalVM)
0.042 Seconds
Quarkus + JIT (via OpenJDK)
2.033 Seconds
Traditional Cloud-Native Stack
9.5 Seconds

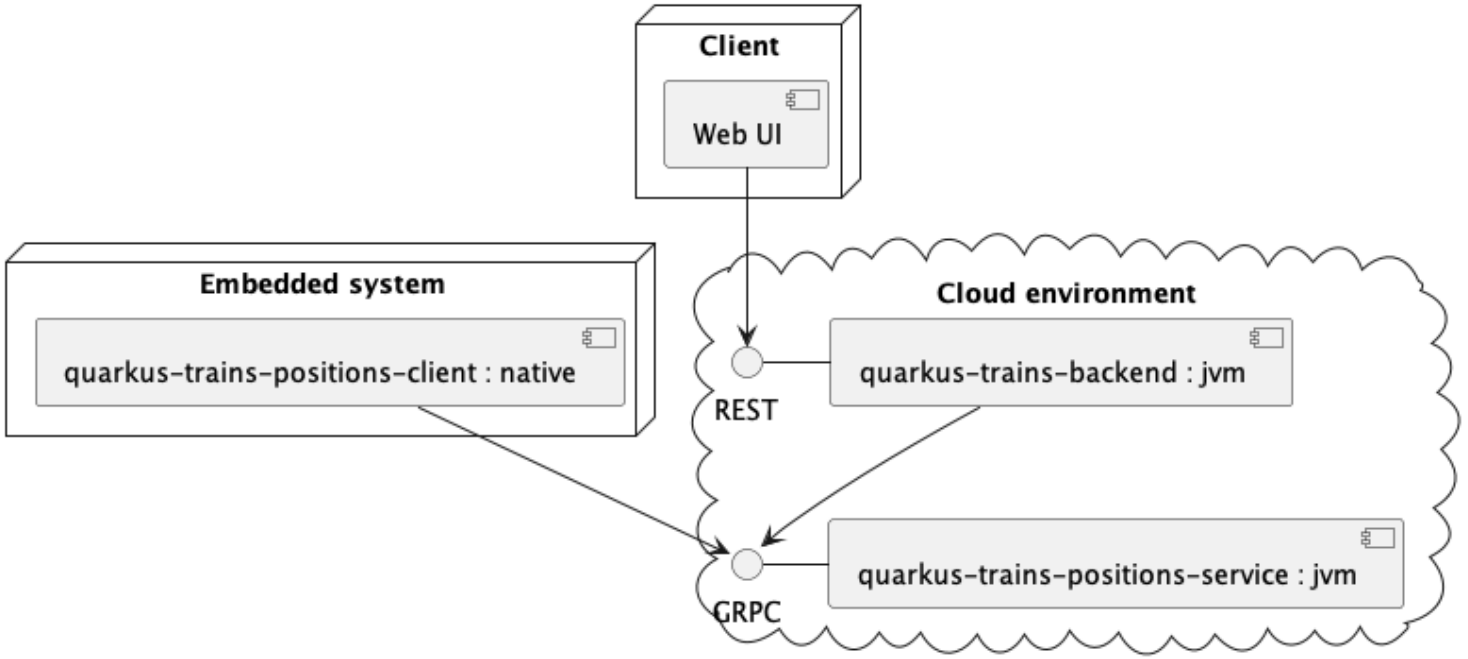
Quarkus

- https://code.quarkus.io/
- Intègre une extension gRPC
- Starter code disponible



Example codé avec Quarkus

- API REST : Liste de trains
- Service gRPC : Positions des trains
- Client gRPC natif : Position de chaque train



https://github.com/niushapaks/quarkus-trains/