gRPC Tutorial

Assignment-2

Assignment-2 has been released!

The allowed languages for Assignment-2 are **Go/Python**.

We will be having a doubt session on the assignment soon!

Installation

For protobuf installation, https://grpc.io/docs/protoc-installation/.

For gRPC installation, refer: https://grpc.io/blog/installation/.


gRPC

(g) + (RPC) - Remote Procedure Calls?

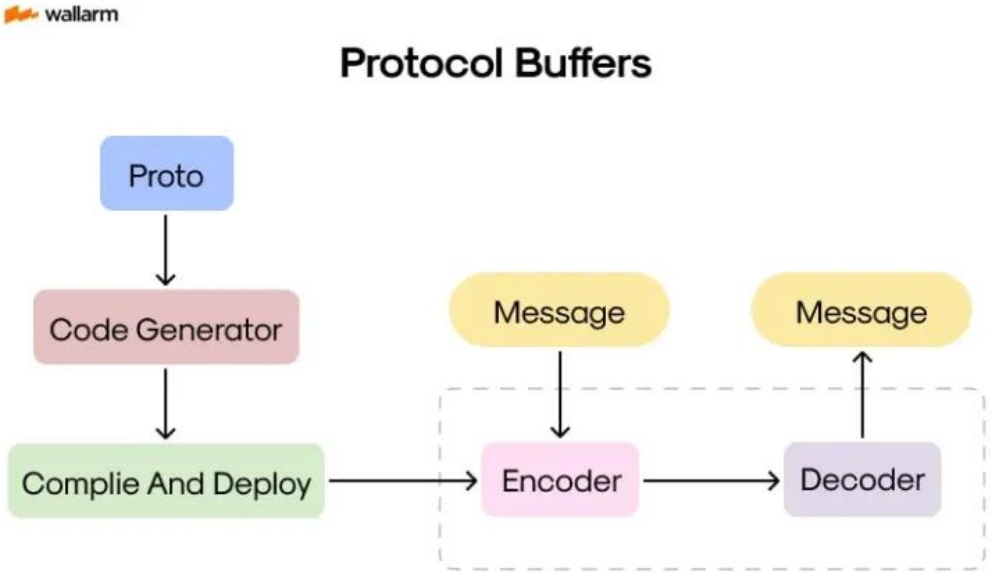
- Provides abstractions to allow client programs to get executed on remote server machine.
- Internally uses UDP and TCP.
- Uses HTTP/2 (???)
- Flow of execution:
  - Prepares function call
  - Contacts the server for executing the problem
  - Report the results back to the client program
- gRPC uses Protobuf as IDL (???)

Interface Definition Languages (IDL)

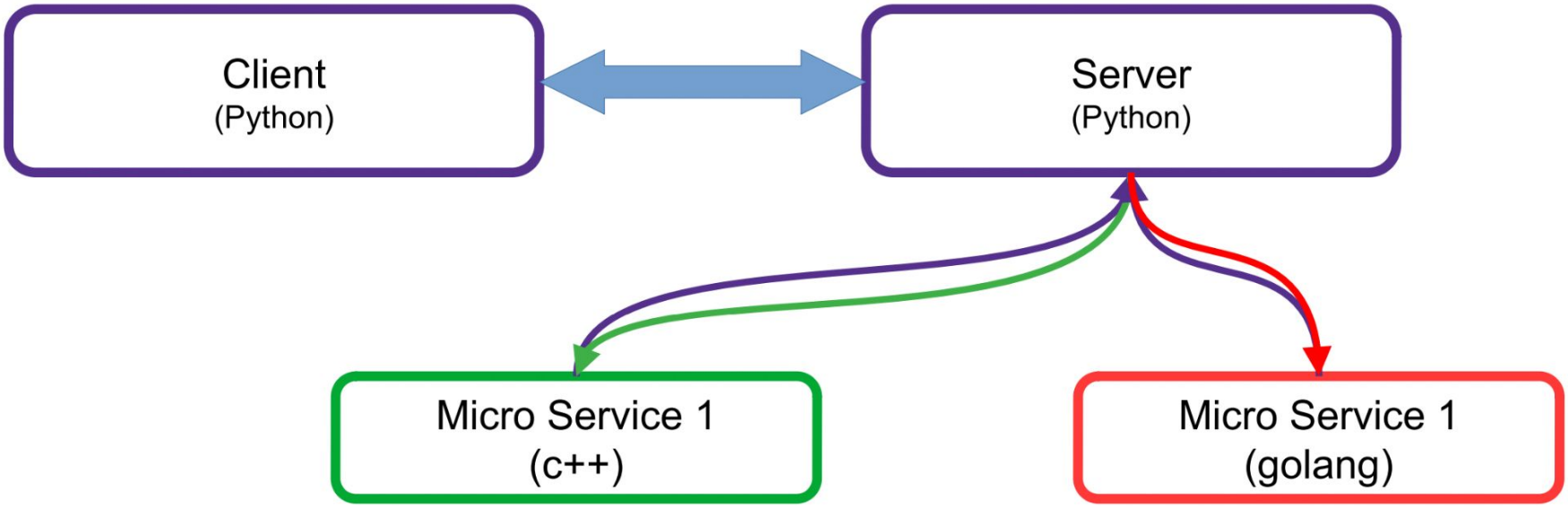
- Language which is used to define the interface between a client and server process in a distributed system.
- Describes both the service interface and structure of the payload messages.
- Few atomic data types are present.
- Achieves language neutrality
- Non procedural (Only interfaces, no implementation)
- Very useful in distributed systems (Why???)

Protocol Buffers (.proto)

Protocol buffers are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler.



gRPC


Client
(Python)
Server
(Python)
Micro Service 1
(c++)
Micro Service 1
(golang)

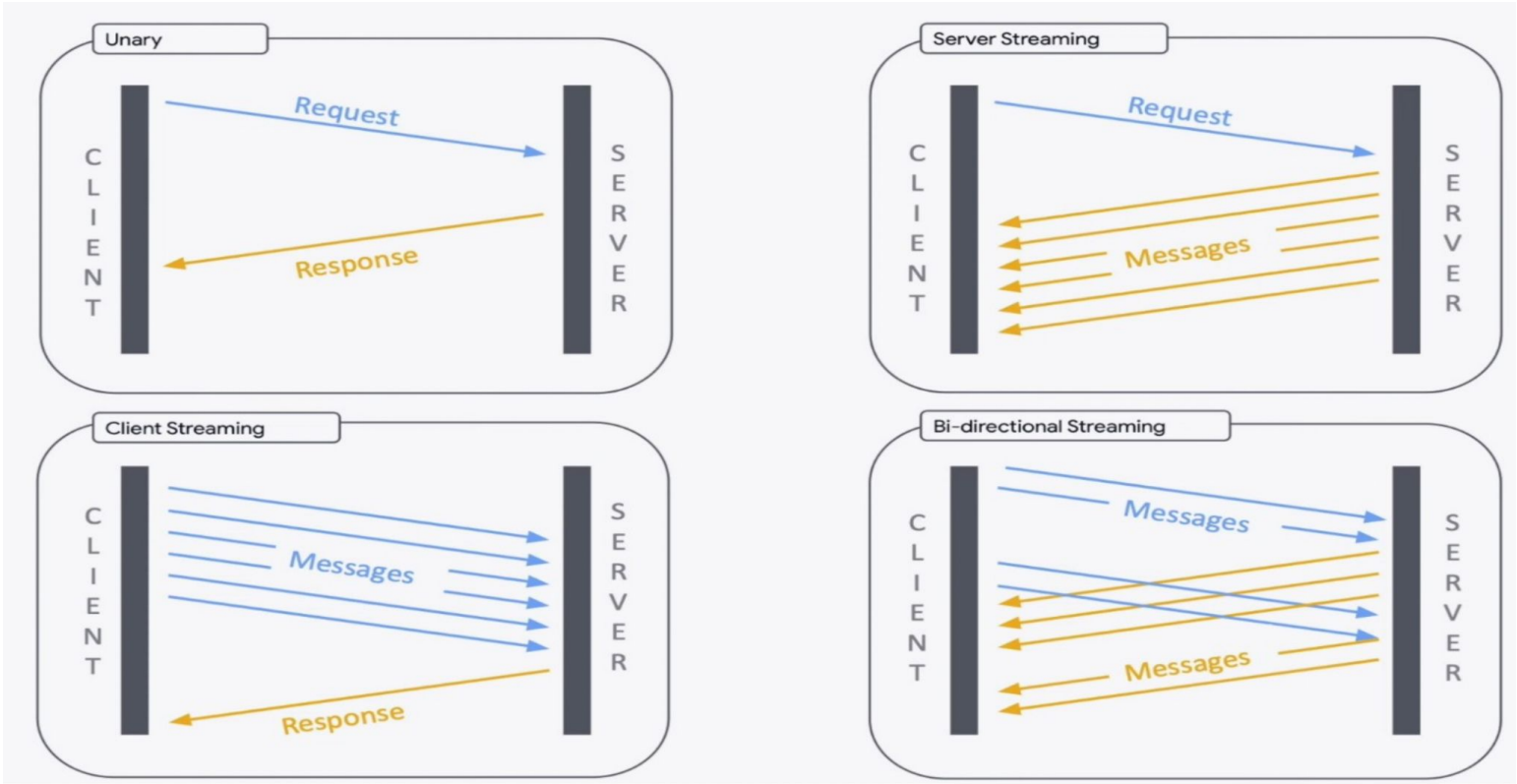
gRPC vs REST

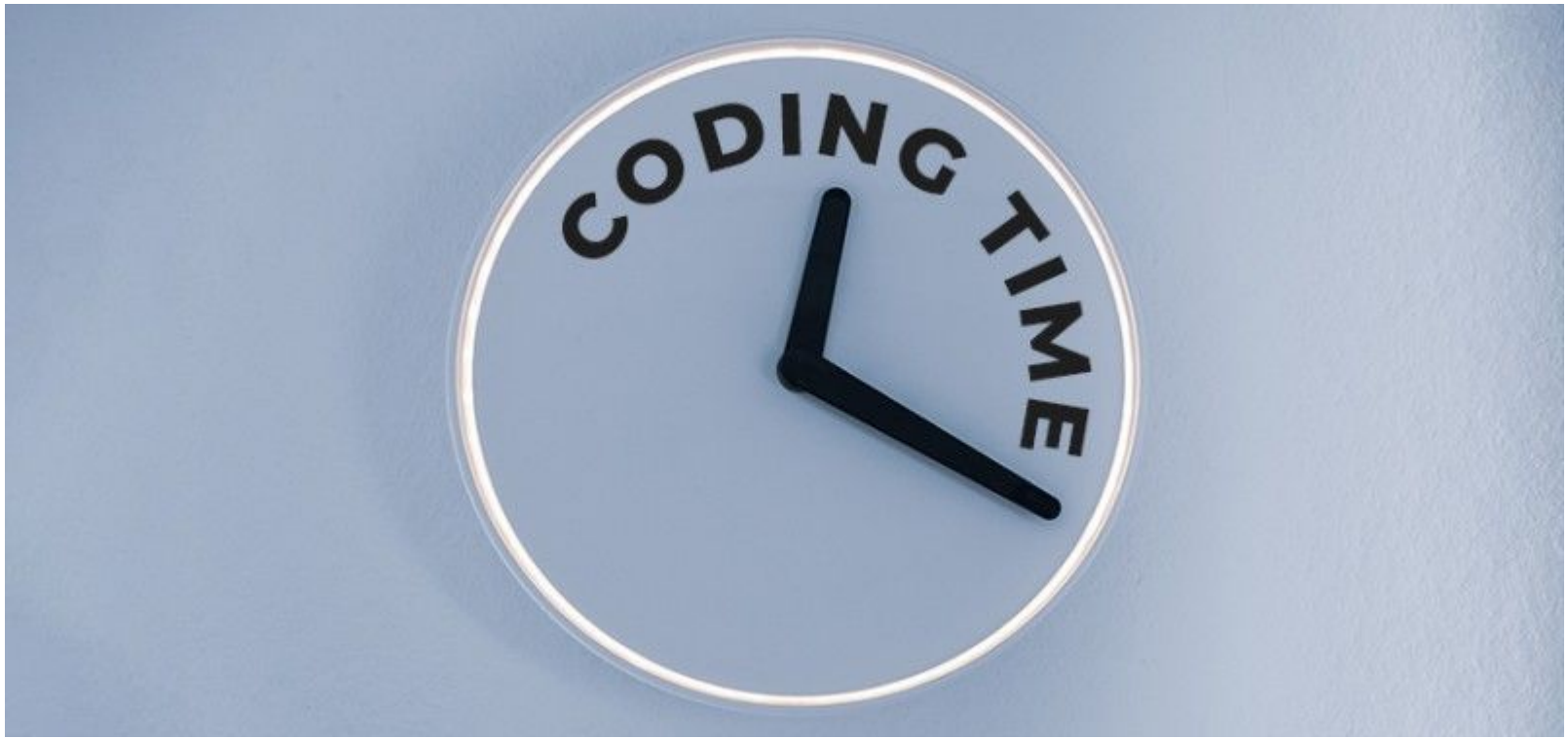
- REST uses HTTP/1.1 while gRPC uses HTTP/2
- What are the key differences?

gRPC vs REST

- REST uses HTTP/1.1 while gRPC uses HTTP/2
- What are the key differences?
  - HTTP/2 is much **faster and efficient**.
  - HTTP/2 offers weighted prioritization.
  - HTTP/1.1 loads resources one after the other.
- https://www.cloudflare.com/en-gb/learning/performance/http2-vs-http1.1/
- Apart from this, a key difference is that gRPC allows for bi-directional streaming.

Types of Streaming in gRPC




CODING TIME

Compiling proto files

The below commands generates the code needed for communication in the servers.

Go

```
protoc -I=. --go_out=. --go-grpc_out=. student_info.proto
```

Python

```
python -m grpc_tools.protoc -I. --python_out=. --grpc_python_out=. course_registration.proto
python -m grpc_tools.protoc -I. --python_out=. --grpc_python_out=. course_info.proto
python -m grpc_tools.protoc -I. --python_out=. --grpc_python_out=. student_info.proto
```

Other features of gRPC

- Allows streaming support.
- Streaming allows for the server (client) to send multiple replies (requests) to one request from (to) a client (server).
- These responses need not all be sent at once and can be sent as the server obtains more data that matches the response.
- The server marks the end of the stream by sending the status details of the server and trailing metadata to the client.

Other features of gRPC

- Authentication via Transport Layer Security (TLS) is another feature of gRPC.
- Interceptors is another nice feature of gRPC
  - Invoked prior to every RPC call.
  - These act as useful mechanisms to perform any pre- or post-processing as part of the RPC call.
  - Useful for logging, metric collection, etc. (Read more on gRPC website, used in A-2 as well)
- Deadlocks, timeouts, cancellations, and error handling, are other additional features supported.

Documentation

For official gRPC documentation, refer: https://grpc.io/docs/.

You can read more about Protocol Buffers at: https://protobuf.dev/

Questions?

Thank You :)