gRPC

- DS_Examples/grpc
- Protobuf definition
- Similar to SOAP but without XML
  - More space efficient
  - Easier to read and write
- https://developers.google.com/protocol-buffers
- Current version: 3 (proto3)
- Automatically generates classes with getters/setters
  - Read/write binary data (very efficient)
- Supports versioning (backwards compatibility)

.proto files

- Namespace definition
  - package example
- Messages → data structures
  - optional/repeated (removed in proto3)
  - type
  - name
  - tag value

```
package customers;

message Person {
        optional string name = 1;
        optional string surname = 2;
        optional int32 ssn = 3;
}

message CustomerBase {
        repeated Person = 1;
}
```

.proto files

- Service → service definition
- Types
  - Synchronous single call
  - Response-streaming
  - Request-streaming
  - Bidirectional streaming

```
service CustomerService {

  rpc GetCustomer(Request) returns (Person){}

  rpc GetAllCustomers(Request) returns (stream
Person) {}

  rpc SendCustomers(stream Person) returns
(Response) {}

  rpc ProcessCustomerRequests(stream Person)
returns (ResultSet) {}

}
```

gRPC

- Generate Python classes (pip install grpcio-tools)
  - `python -m grpc_tools.protoc --proto_path=$SRC_DIR --python_out=$OUT_DIR --grpc_python_out=$GRPC_OUT $PATH_TO_PROTO_FILE`
- Write implementation code (client/server)

Exercises: gRPC

- DS_Examples/grpc/example.proto
- Generate client/server stubs.

1. Change the return type of the SendPurchases method so that the client receives for each Purchase an object indicating if the purchase was successfully added to an existing customer or, when not, an error message.

Homework: gRPC

2. Add a new service method that uses a response stream from the server to receive all Customers with purchases that, in sum, exceed a given amount passed as parameter.
   - “Give me all customers that purchased more than 200€ worth of articles”.