**Advanced Extensions**

Exercise 9 - Events

**Extension 1 -** pub/sub (fanout)

Look at the following tutorial

<https://www.rabbitmq.com/tutorials/tutorial-three-java.html>

Modify the src/main/java/org/freo/purchase/Publish.java file to Publish to a fanout exchange instead of a queue.

Now modify the Python client to subscribe to the fanout exchange:

<https://www.rabbitmq.com/tutorials/tutorial-three-python.html>

Run multiple copies of the Python client to subscribe to the events.

Exercise 10 - gRPC

**Extension 2**

Create a Python server for the same proto definition.

Hint: You already created part of the required code when you generated the client stubs, because it also generates the Python code you need to handle the server.

Hint 2: You need something like this as a PurchaseServer.py

This is modified from <https://github.com/grpc/grpc/blob/master/examples/python/helloworld/greeter_server.py>

from concurrent import futures

import time

import logging

import grpc

import purchase\_pb2

import purchase\_pb2\_grpc

class Purchase(purchase\_pb2\_grpc.PurchaseServicer):

def purchase(self, request, context):

return purchase\_pb2.PurchaseReply()

def serve():

server = grpc.server(futures.ThreadPoolExecutor(max\_workers=10))

purchase\_pb2\_grpc.add\_PurchaseServicer\_to\_server(Purchase(), server)

server.add\_insecure\_port('[::]:50051')

server.start()

try:

while True:

time.sleep(\_ONE\_DAY\_IN\_SECONDS)

except KeyboardInterrupt:

server.stop(0)

if \_\_name\_\_ == '\_\_main\_\_':

logging.basicConfig()

serve()

Exercise 15 - Ballerina

**Extension**

Extend the service to offer both gRPC and RESTful mediation into the SOAP message.

This guide might help you:

<https://ballerina.io/learn/by-guide/grpc-service/>