Rome is the city where I live since I wos born. It counts about 3.5 official inhabitants to witch we have to add about 1 million of non resident people (students, tourists, workers). To these figures we have also to add about 700 vehicles each 1000 people, a rate that makes mobility a challenge.

Everyone could easily realize that with there figures besides mobility there are also pollution issues.

A partial solution of these issues could be addressed reducing the vehicles that daily deliver goods.

For a more sustainable for the environment and efficient mobility in such a context the daily delivery of goods could be done by means of (through) “smart” electrical vehicles.

To overcome the limited life of the batteries these vehicles has been enginnered with special battery pack that could be easily and rapidly changed in dedicated service station.

In this way these vehicle will not need to stop for charging their batteries and so could guarantee much more short breaks

Could be much more efficient in comparison to the other electric vehicle (that do non support this kind of batteries – not engineered in such a way)

I do not know if this is a realizable scenario but if so it is necessary a software solution by witch manage this scenario.

Each driver of this ecological delivery company according to his delivery plan each morning program the expected battery changes that will reasonably occur during a day. By means of a mobile application (web application) he will book one or more fresh batteries supplied by the stations distributed in the city.

In case of emergency it will be the vehicle itself that will notify to the driver the necessity to come to the nearest station with available fresh batteries, find by a software running in the vehicle.

In this chart I have tried to summarize the technology stacks that realize the design patterns chosed for the architecture of this system

In this chart I have arranged that will accomplish the lifecycle steps imagined for this project and the corresponding realizing services both in a local and in a cloud environment.

## Demo

## Integration test - Docker Local

## Quality assurance - Docker Hub - Jenkins@openshift