POLYTECH
NICE-SOPHIA

UNIVERSITÉ
CÔTE D'AZUR

DRONE DELIVERY SOLUTION

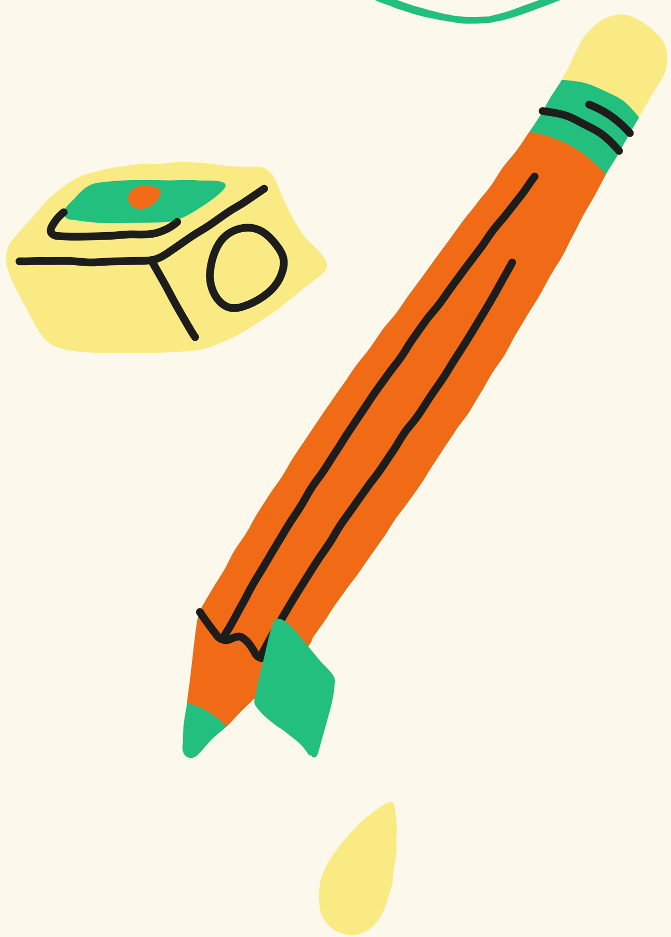
Deliver packages at great distance by drone

25.02.2022

ANAGONOU Patrick
ANIGLO Jonas
FRANCIS Anas
ZABOURDINE Soulaiman

Summary

- The subject
- Evolution
- Handling of the multiple regions
- Resiliency of the FlightMonitor
- Comparison between Kafka and RabbitMQ
- Team organisation
- What’s left to do



The subject

V10bis: battery management, charging dock networks with optimization (docking station are on the path of heavy drones for long-distance deliveries, “à la TESLA“)

spring®

PostgreSQL

Java™

RabbitMQ™

Evolution

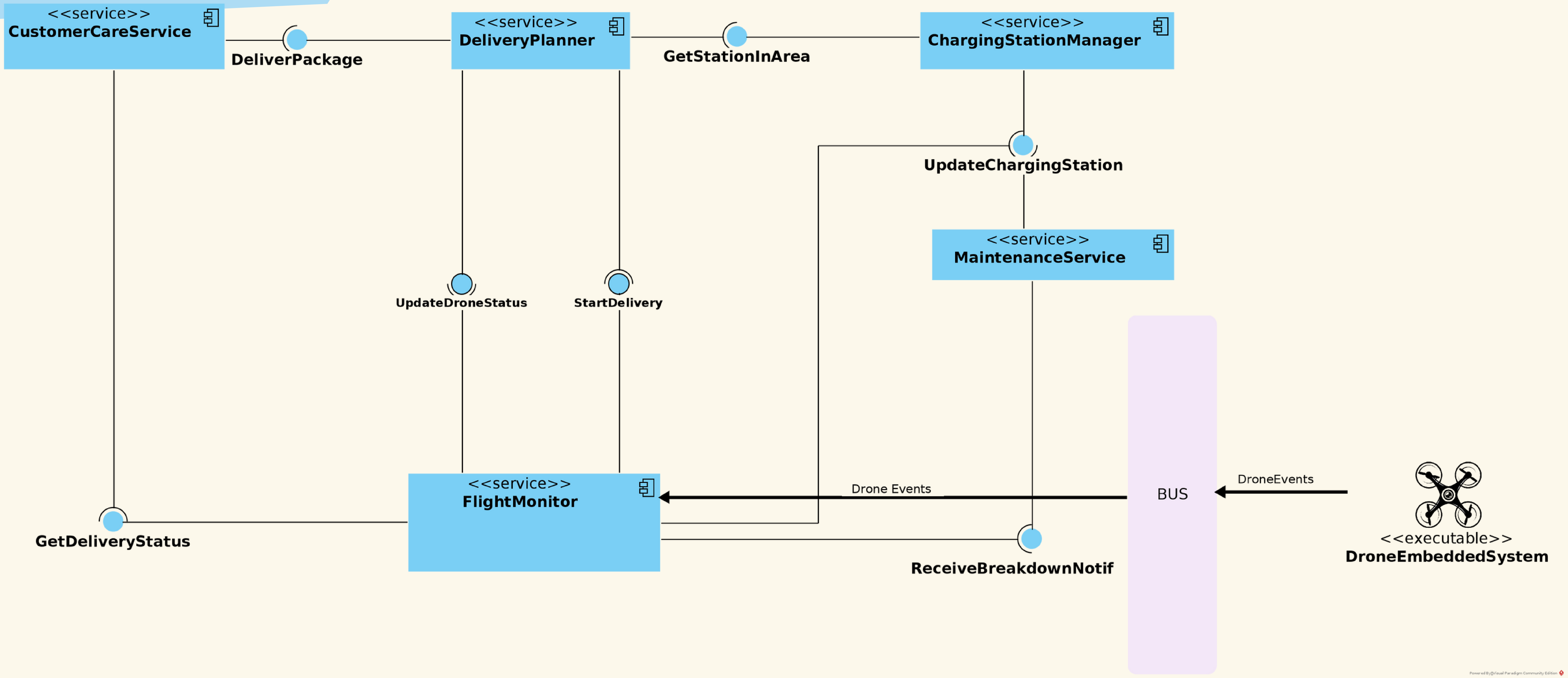
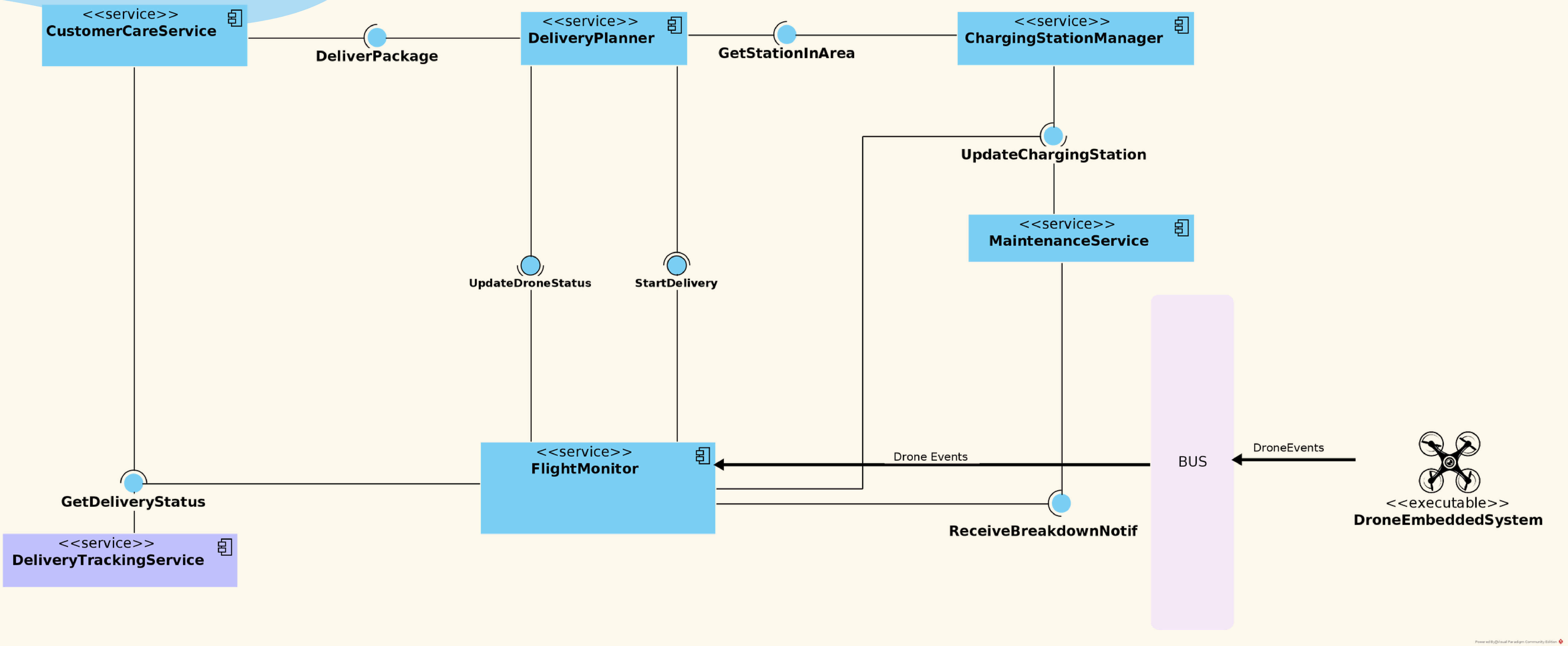
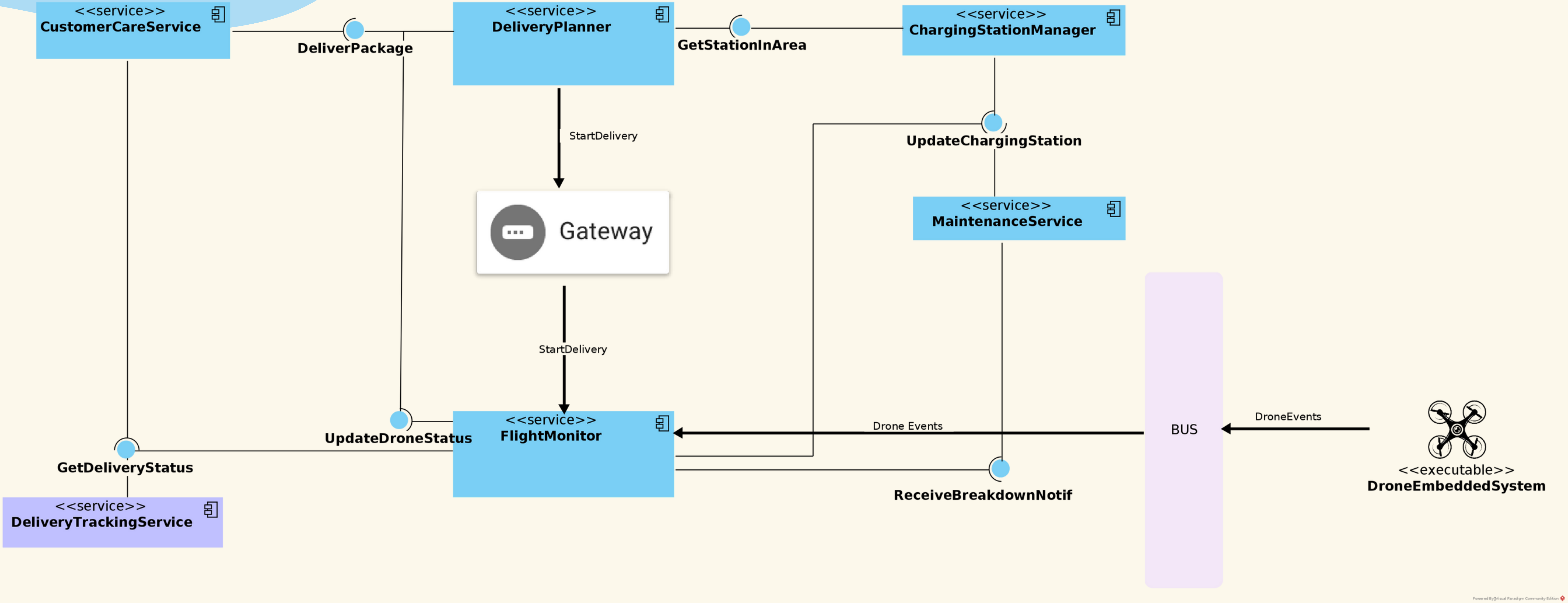


Diagram of our previous architecture

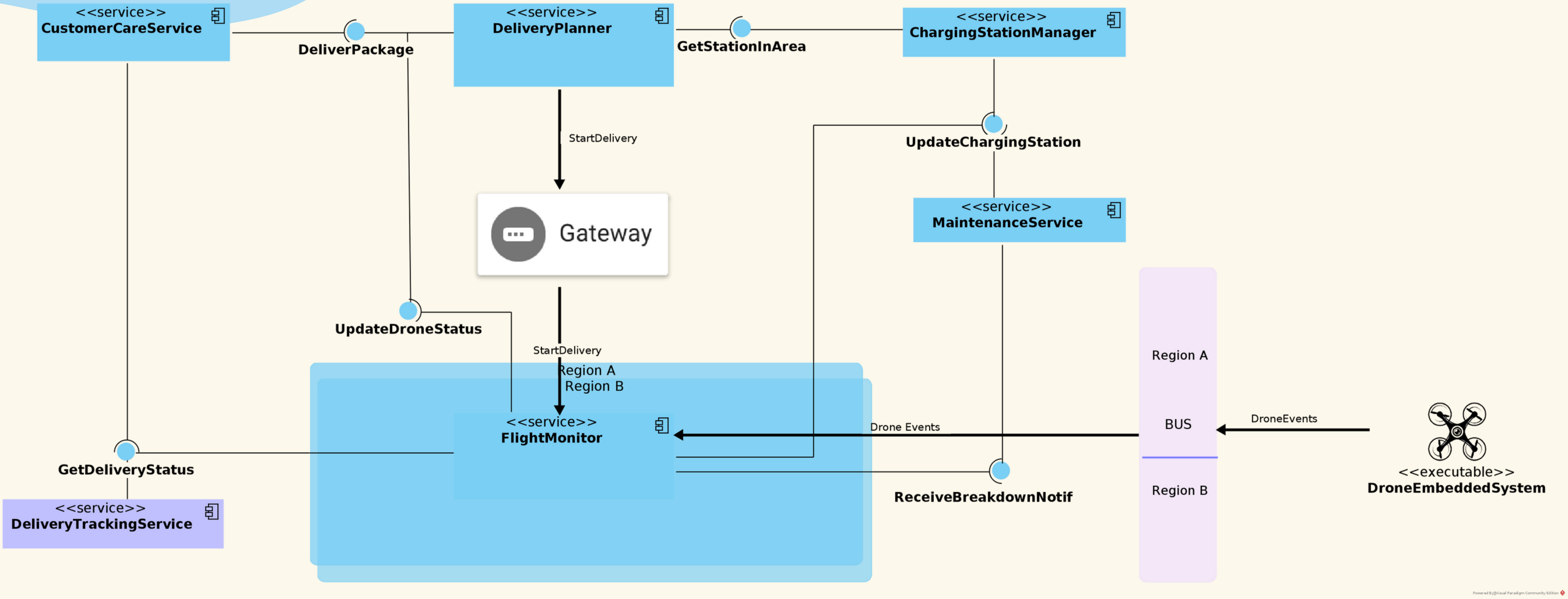
Evolution


<<service>>
CustomerCareService
DeliverPackage
<<service>>
DeliveryPlanner
GetStationInArea
<<service>>
ChargingStationManager
UpdateChargingStation
<<service>>
MaintenanceService
UpdateDroneStatus
StartDelivery
<<service>>
FlightMonitor
Drone Events
BUS
DroneEvents
<<executable>>
DroneEmbeddedSystem
ReceiveBreakdownNotif
GetDeliveryStatus
<<service>>
DeliveryTrackingService
Powered By Visual Paradigm Community Edition

Evolution

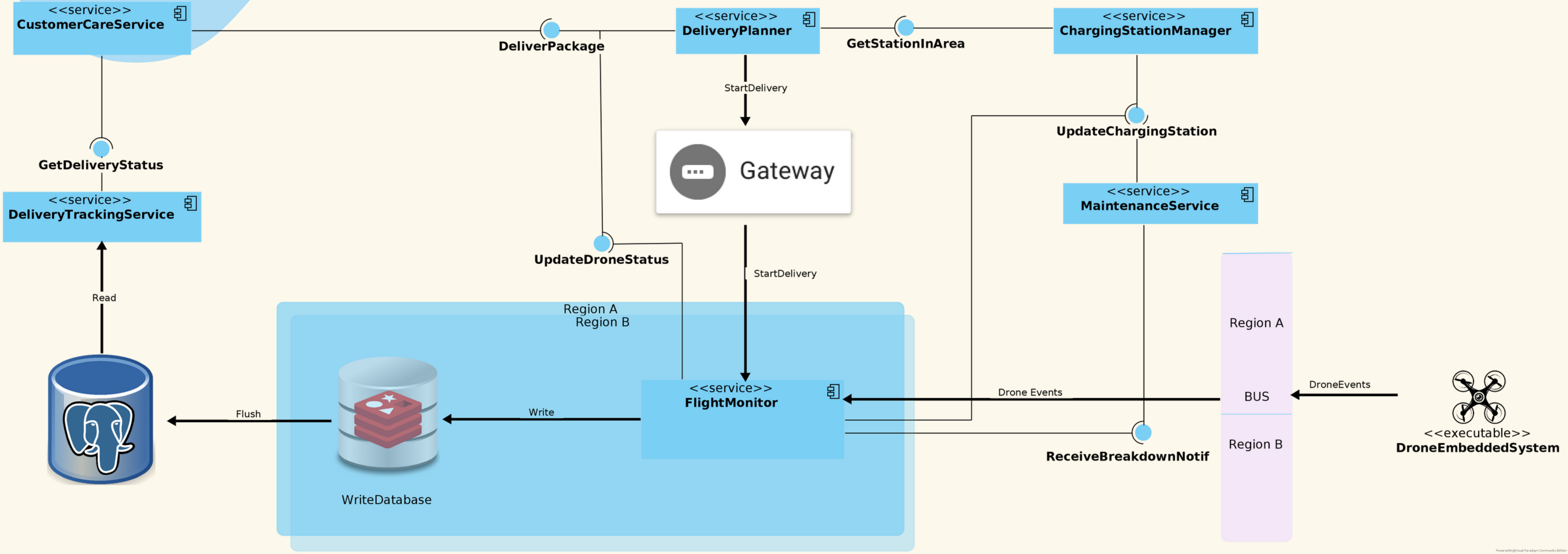

<<service>>
CustomerCareService
DeliverPackage
<<service>>
DeliveryPlanner
GetStationInArea
<<service>>
ChargingStationManager
StartDelivery
UpdateChargingStation
Gateway
<<service>>
MaintenanceService
StartDelivery
BUS
DroneEvents
Drone Events
<<service>>
FlightMonitor
UpdateDroneStatus
GetDeliveryStatus
ReceiveBreakdownNotif
<<service>>
DeliveryTrackingService
<<executable>>
DroneEmbeddedSystem
Powered By Visual Paradigm Community Edition

Evolution


<<service>>
CustomerCareService
DeliverPackage
<<service>>
DeliveryPlanner
GetStationInArea
<<service>>
ChargingStationManager
StartDelivery
UpdateChargingStation
Gateway
<<service>>
MaintenanceService
UpdateDroneStatus
StartDelivery
Region A
Region B
Region A
BUS
Region B
DroneEvents
<<service>>
FlightMonitor
Drone Events
GetDeliveryStatus
ReceiveBreakdownNotif
<<executable>>
DroneEmbeddedSystem
<<service>>
DeliveryTrackingService
Powered By Visual Paradigm Community Edition

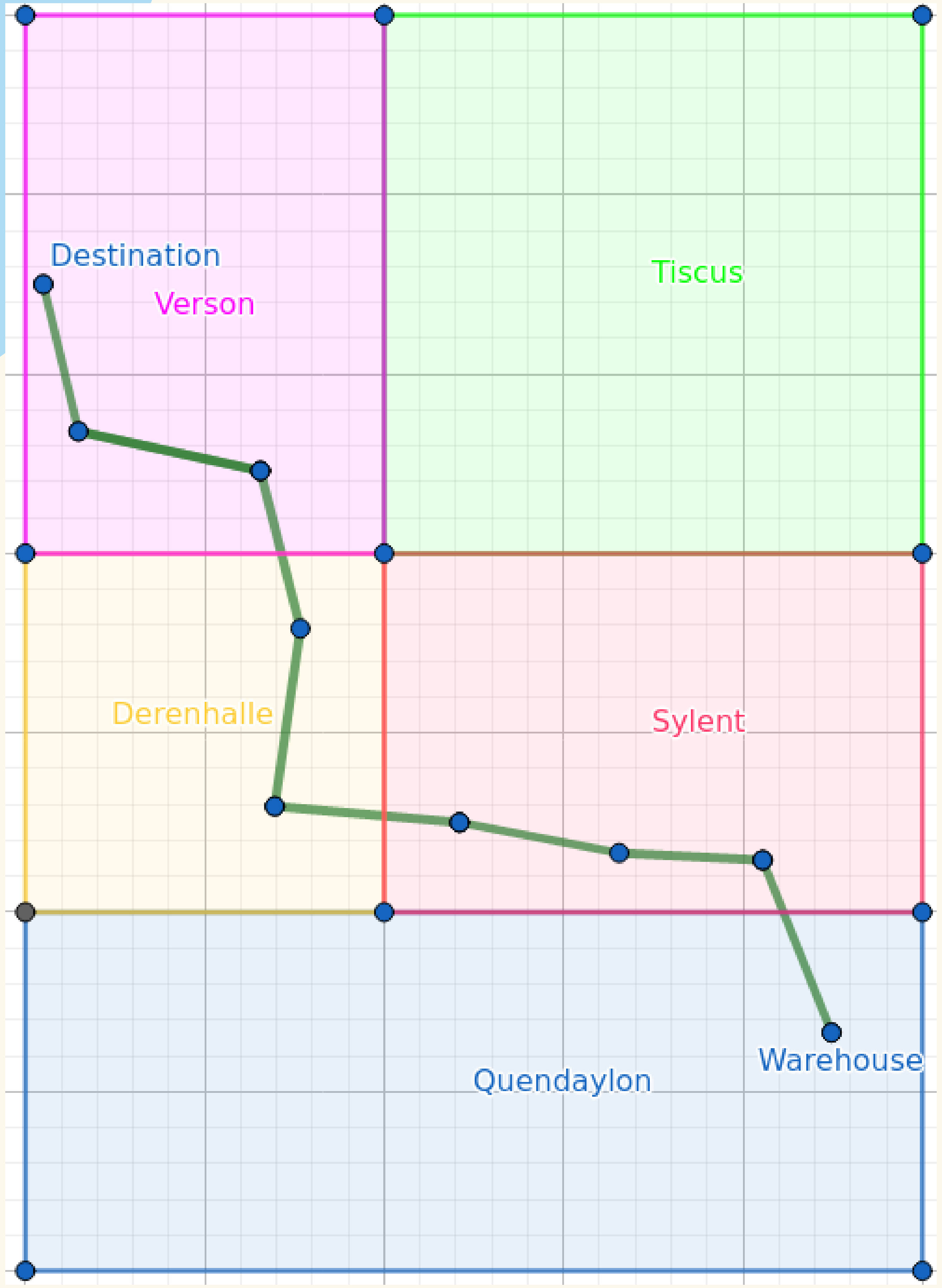
Multiple regions

Architecture



Multiple regions

Demo



FlightMonitor's Resiliency

Communication between the deliveryPlanner and the flightMonitor

Idempotence

- FlightMonitors are capable of getting the same message multiple time and handle it in a idempotent way

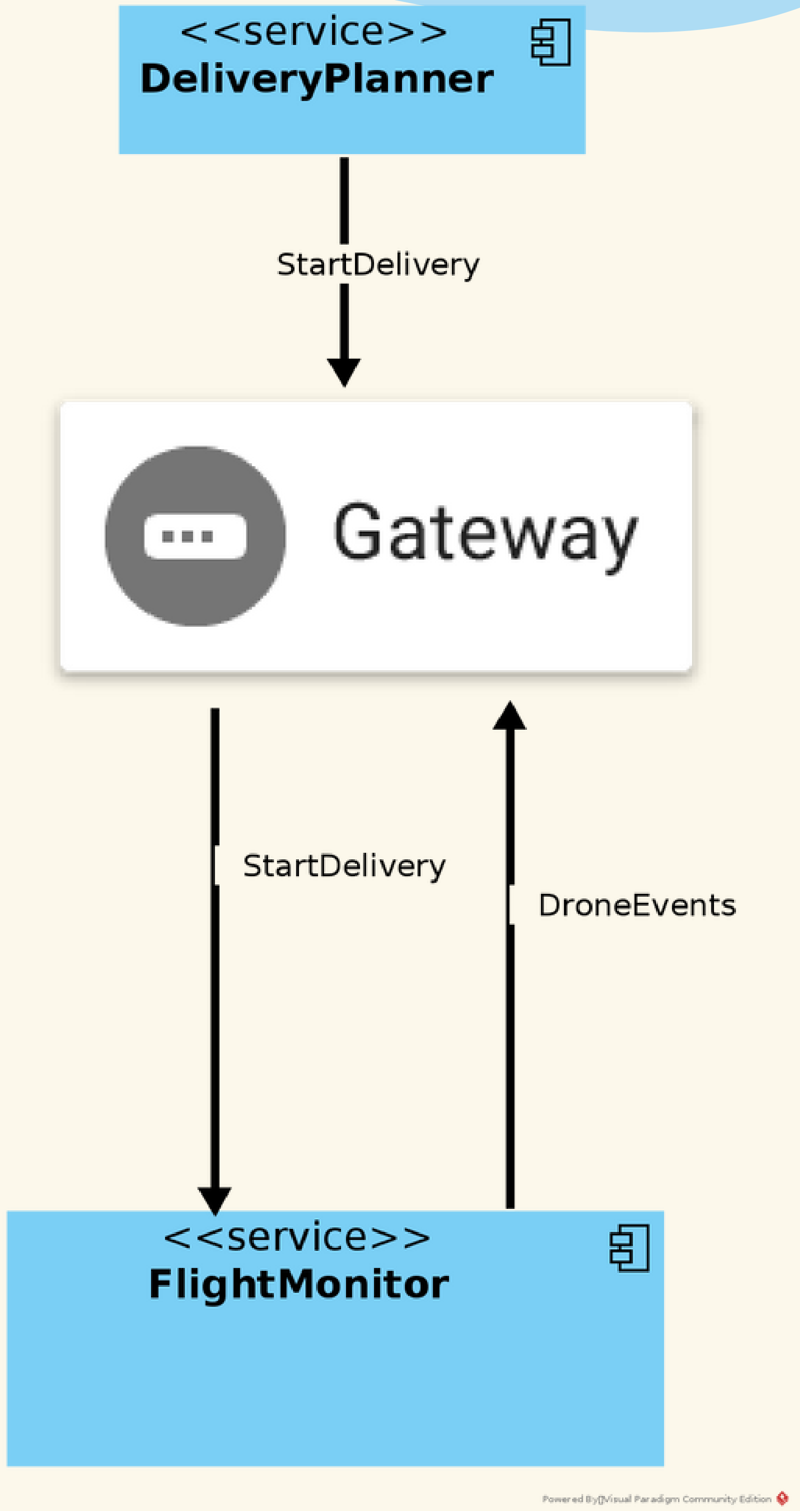
Consumer contract

- We reinforce validation of the messages the flightMonitor get from other services through defensive coding

Fallbacks & Retries

- The gateway is responsible for retrying requests if the FlightMonitor is not available

  The FlightMonitor also retries requests when it can’t reach the other services



FlightMonitor's Resiliency

Communication between the drone and the flightMonitor

Types of communication

- REGULAR
- SPECIAL (stops, finished delivery, start)
- MAYDAY

We implemented manual confirmation for important messages and left less critical messages with the default auto-acknowledgment

FlightMonitor's Resiliency

What if the ... fails ?

FlightMonitor

Within a region we can have multiple instances of flightMonitor so if a single instance of flightMonitor fails we can keep registering the progress of the flying drones
<u>Possible Mitigations:</u> use a deployment solution with healthcheck, quick restart and multiple instances

Gateway

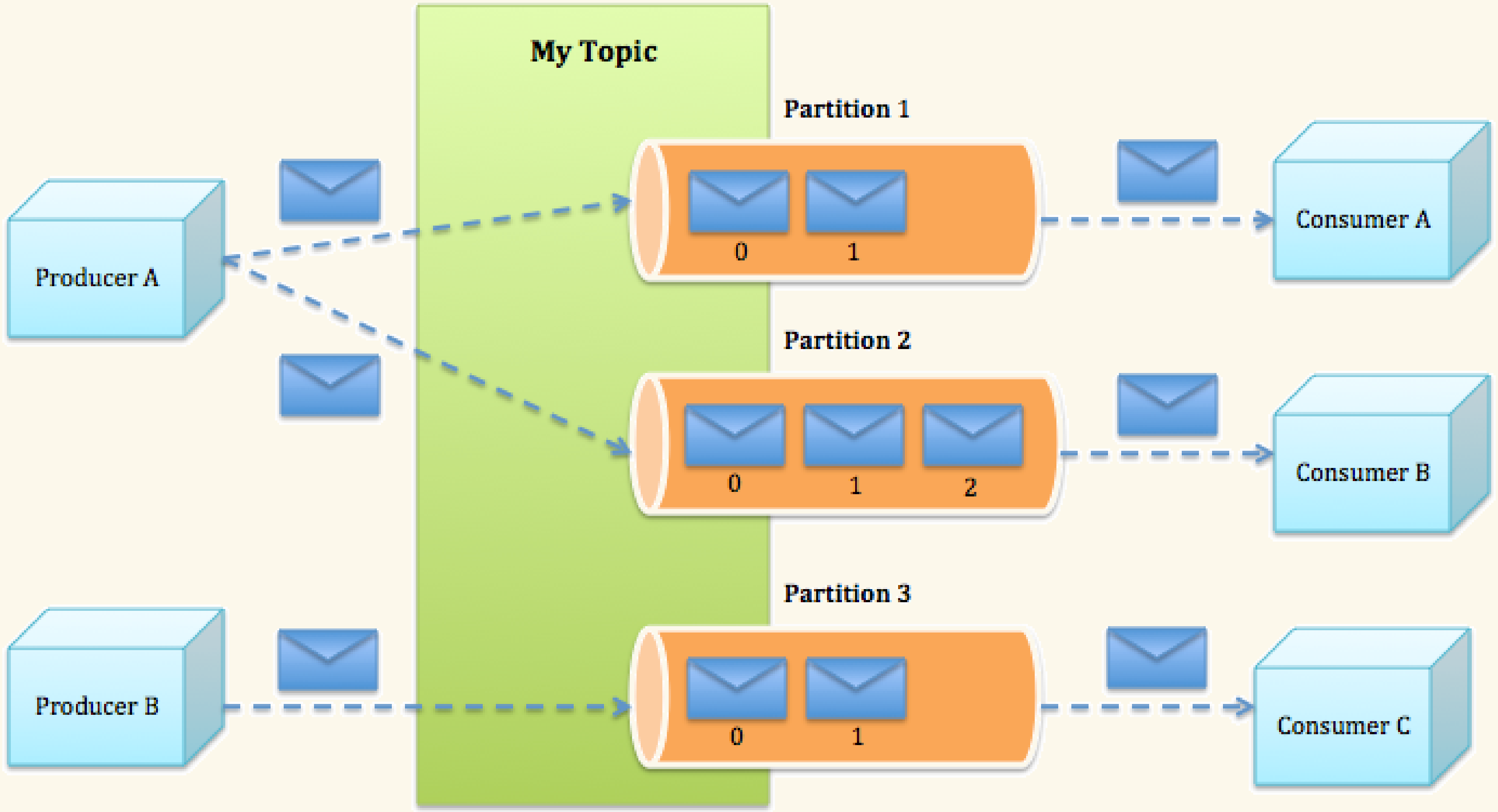
We can't start new drones but we can still handle flying drones and accept new deliveries
<u>Possible Mitigations</u>: retry later, multiple gateways, and load balancer

Message bus

We can't follow the drones anymore or launch new drones
<u>Possible Mitigations</u>: multi-region cluster of message brokers, reliable deployment

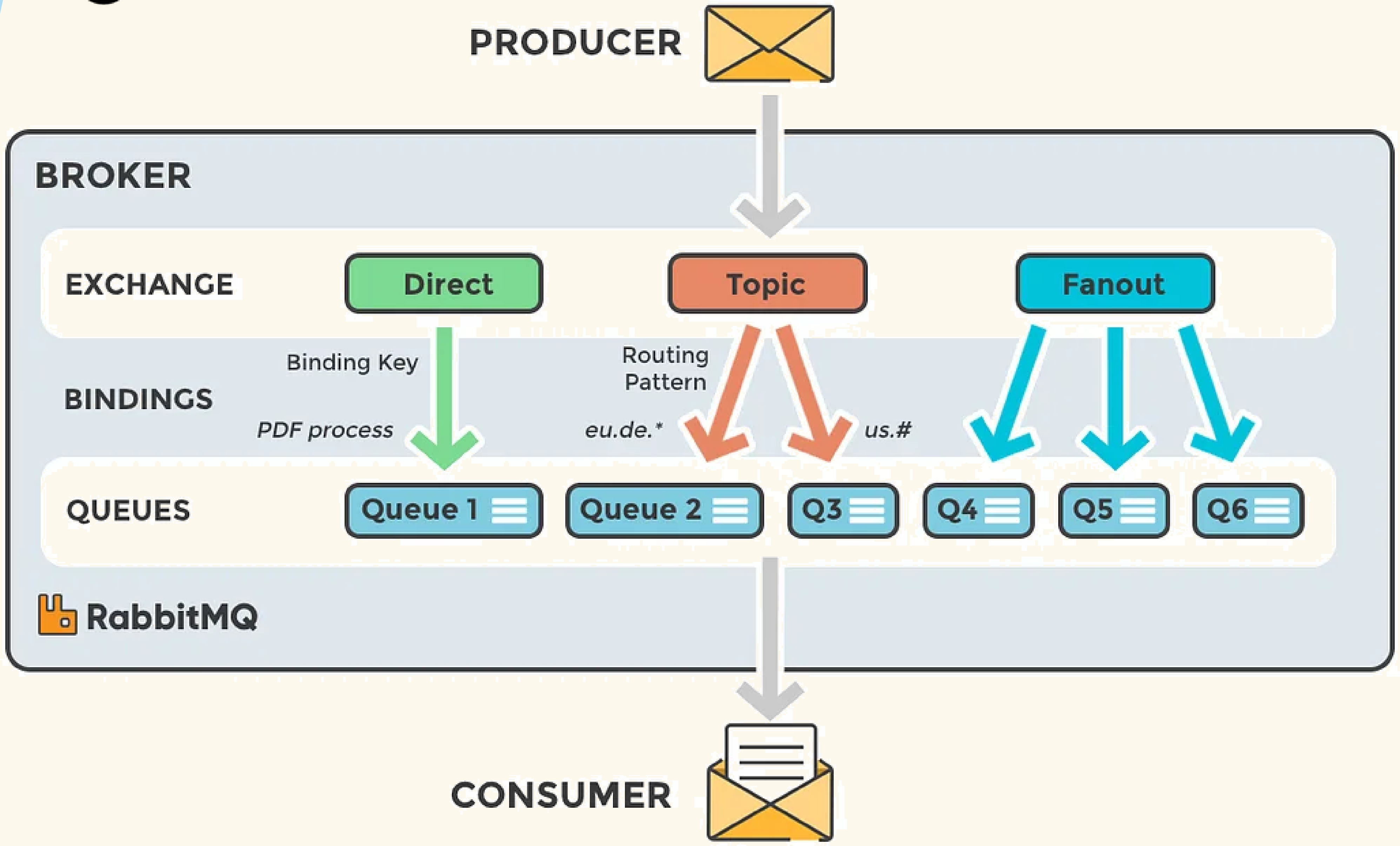
Kafka and RabbitMQ

Two different paradigms



Kafka architecture

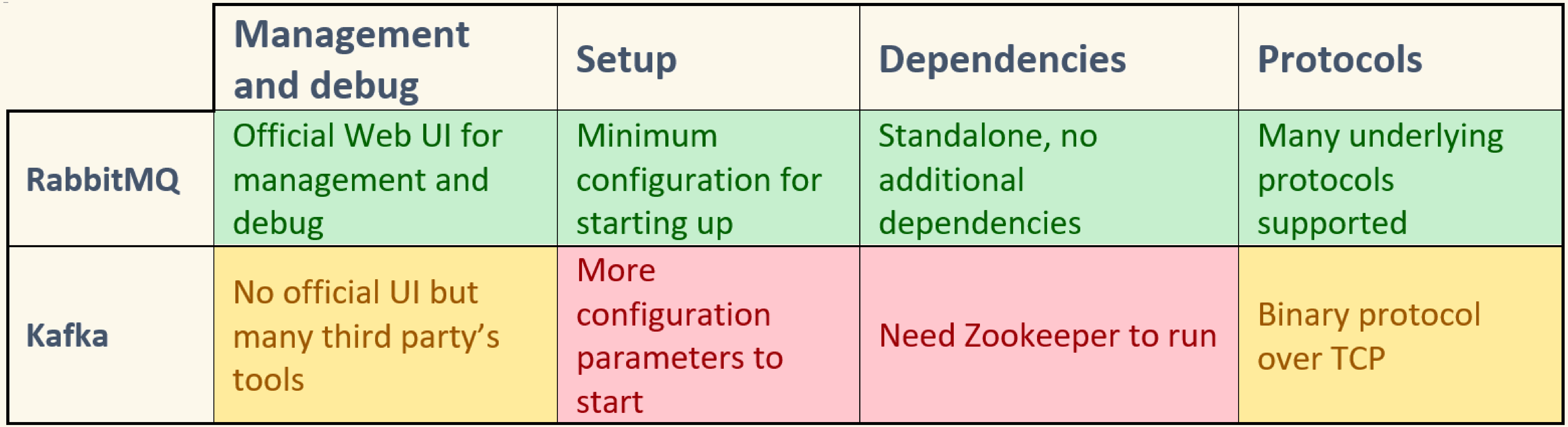
kafka RabbitMQ™



RabbitMQ architecture

Kafka and RabbitMQ

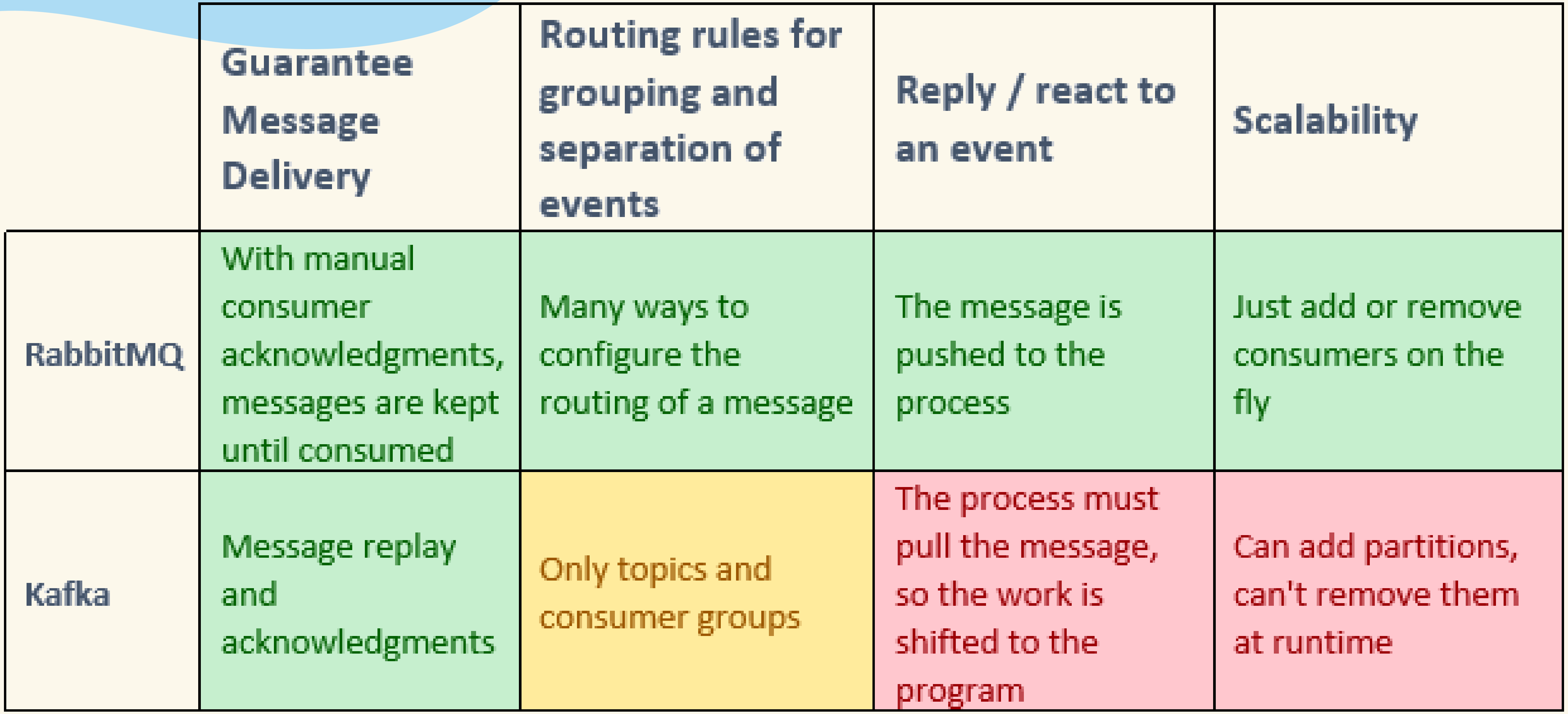
Non - functional comparison



	Management and debug	Setup	Dependencies	Protocols
RabbitMQ	Official Web UI for management and debug	Minimum configuration for starting up	Standalone, no additional dependencies	Many underlying protocols supported
Kafka	No official UI but many third party’s tools	More configuration parameters to start	Need Zookeeper to run	Binary protocol over TCP

Kafka and RabbitMQ

Functional



	Guarantee Message Delivery	Routing rules for grouping and separation of events	Reply / react to an event	Scalability
RabbitMQ	With manual consumer acknowledgments, messages are kept until consumed	Many ways to configure the routing of a message	The message is pushed to the process	Just add or remove consumers on the fly
Kafka	Message replay and acknowledgments	Only topics and consumer groups	The process must pull the message, so the work is shifted to the program	Can add partitions, can't remove them at runtime

Team organization

- Patrick : Changing Region management+ Kafka vs AMQP
- Jonas : Synchronisation of REDIS
- Anas : Gateway + CI maintainability
- Soulaiman : Resiliency on services

What's left to do

- Improve path of flightplan
- More realistic regions area
- Improve resiliency on all services

Thank you for your attention !