ALPEGA

A JOURNEY TOWARDS AN INTEGRATED LOGISTICS PLATFORM

REDHAT OPEN TOUR,
VIENNA 2022

MAG. STEFAN HEIL

alpega
Shaping Transport Collaboration

Who is Alpega?

Credit CC Pixabay

Alpega group: A history of logistics excellence

alpega
Shaping Transport Collaboration

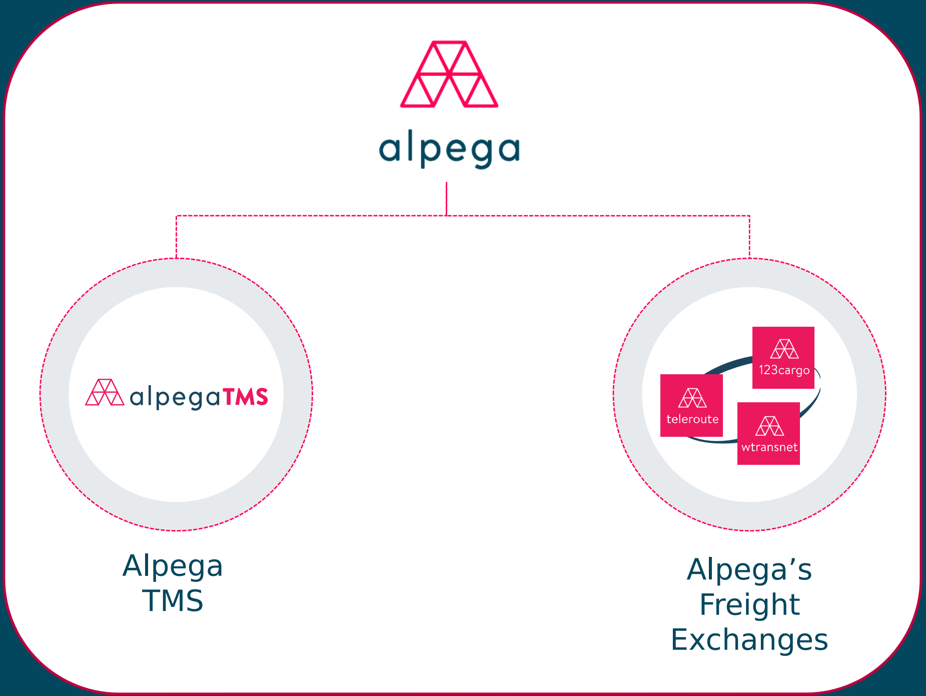
- Alpega TMS is the combination of two of the market's most innovative, established and trusted TMS solutions

transwide
part of alpega

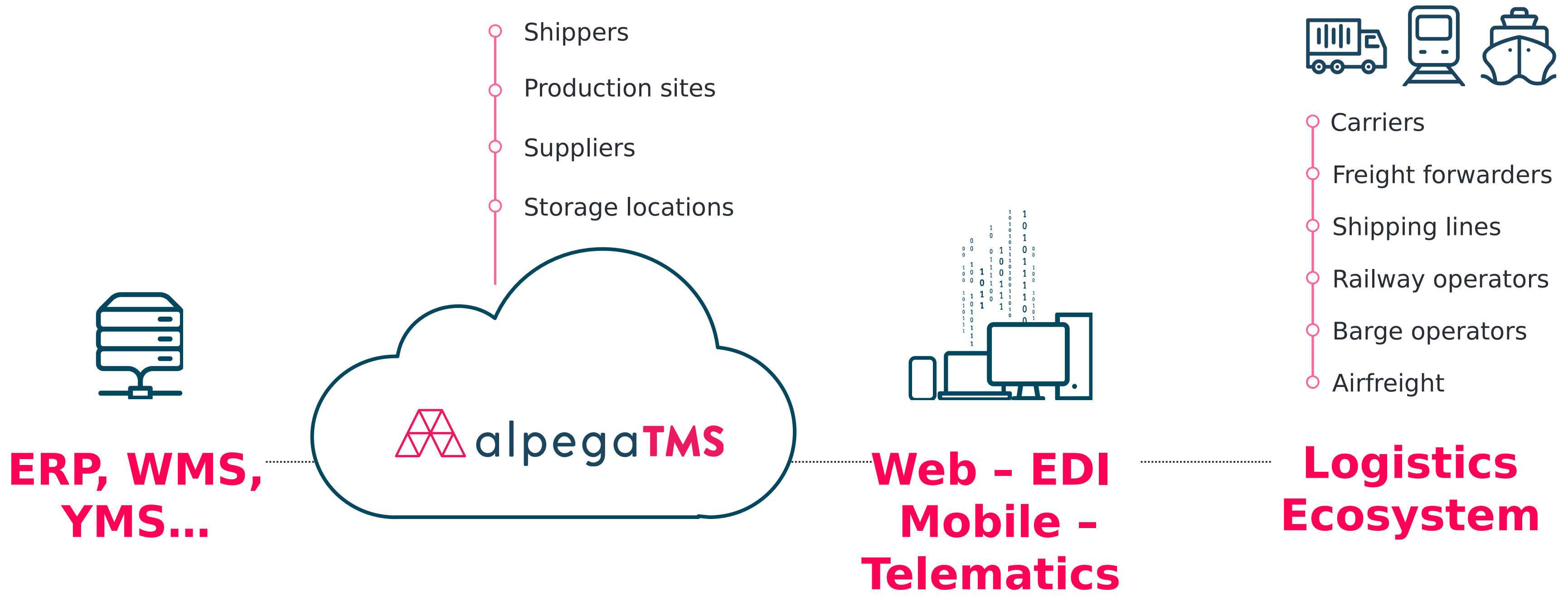
+

inet
part of alpega

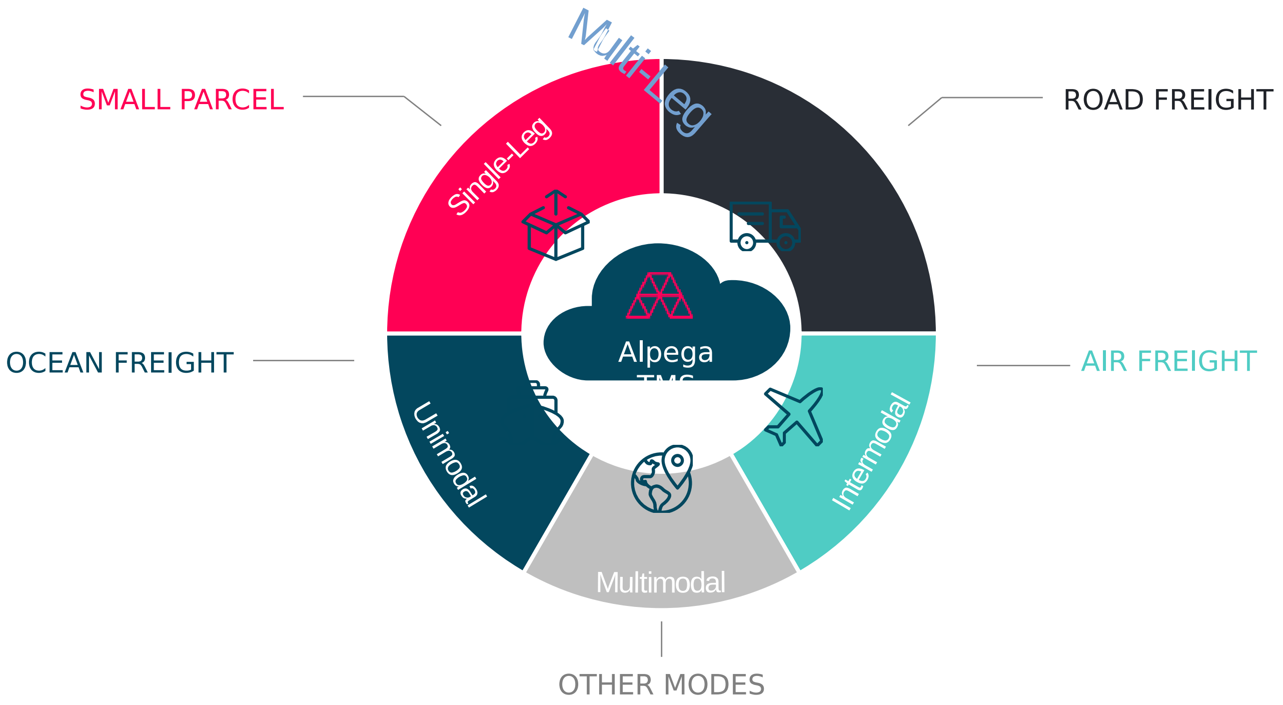
alpegaTMS



Alpega TMS seamlessly connects logistics ecosystems



Global and multimodal



“

Alpega TMS provided us with the efficiencies and insight that we needed to support our growth

“

The ability to quickly connect with all of our suppliers is my personal favorite

“

Major advantages for our company and supply networks

“

An agile and collaborative tool

SIEMENS

BMW GROUP

adidas

Cargill

INEOS

Unilever

DHL

Alpega’s strengths and core expertise

1

DEEP-ROOTED LOGISTICS DOMAIN EXPERTISE

600+ LOGISTICS EXPERTS
30+ years of experience

2

10 YEARS IN A ROW IN THE GARTNER MAGIC QUADRANT

Shows recognition of our unique offering to the market, especially when combined with the international Freight Exchanges that are part of Alpega Group

3

SCALABLE

Our future-proof solution lets you start with exactly what you need and scale up as your complexity increases

4

80 COUNTRIES

Alpega is a globally operating logistics software company

5

BORN IN THE CLOUD

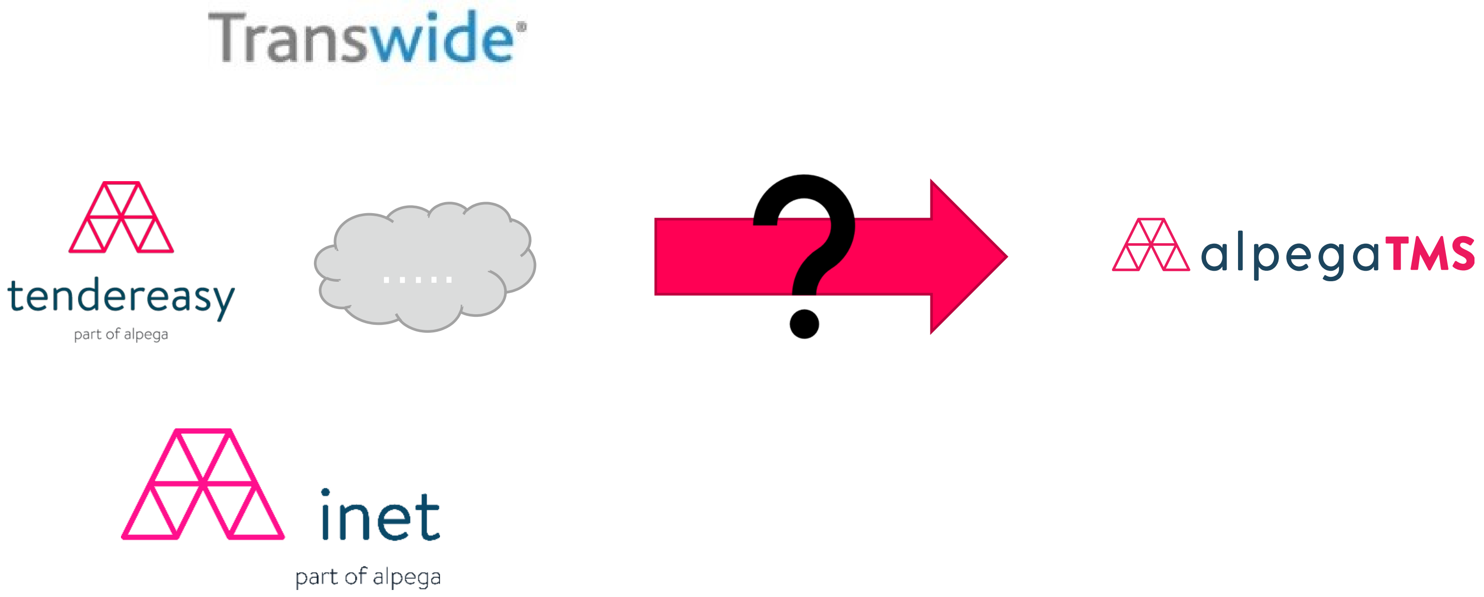
All our systems are cloud-based and available 24/7

6

OVER 147,000 USERS

We have a proven track record- our TMS can be easily integrated into any given IT environment, any ERP system

FROM SINGLE PRODUCTS TO ONE PLATFORM



THE MAIN CHALLENGES ON OUR JOURNEY

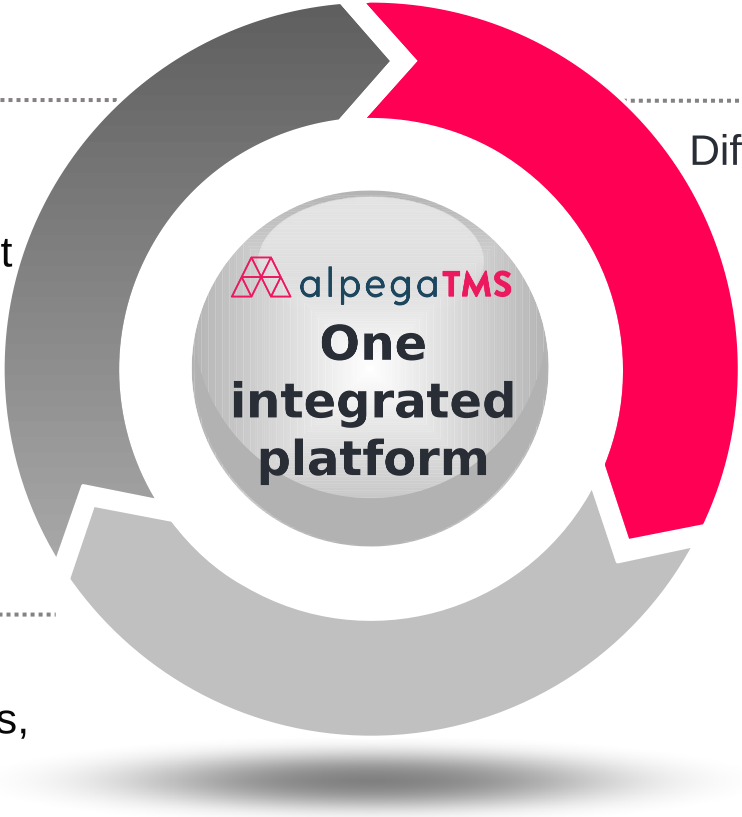
alpega
Shaping Transport Collaboration

Architecture

Autonomous products built with different technologies, different clouds and different design patterns must work seamlessly together

User experience

Different products designed by different teams, using different technologies and different patterns need to appear as „one solution“ to the customer



Testing

Different technology stacks, different automation patterns, integration testing across multiple platforms

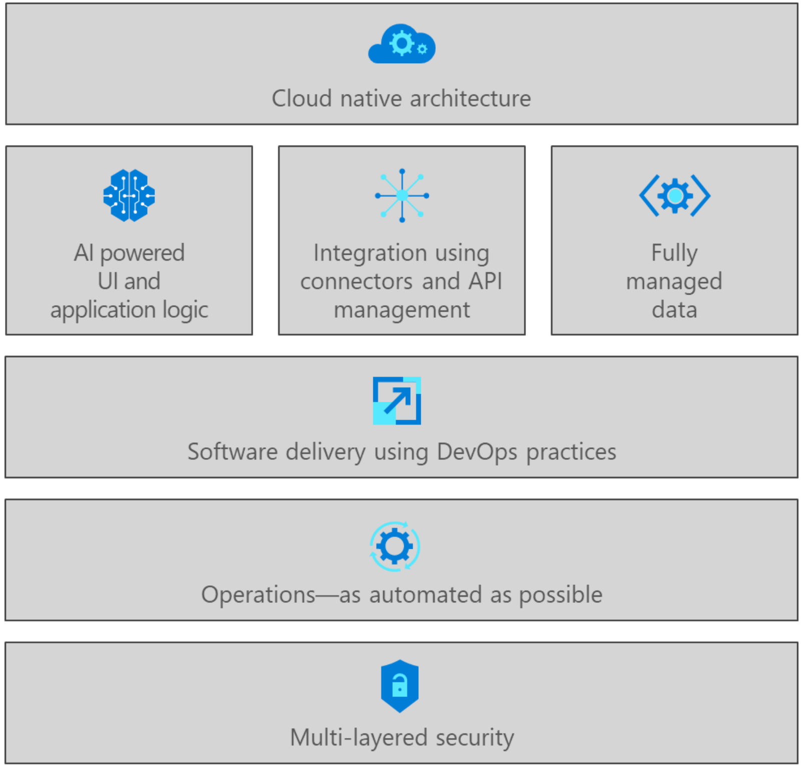
/ Cloud native – Concepts

CLOUD NATIVE APPLICATION DEVELOPMENT

alpega
Shaping Transport Collaboration

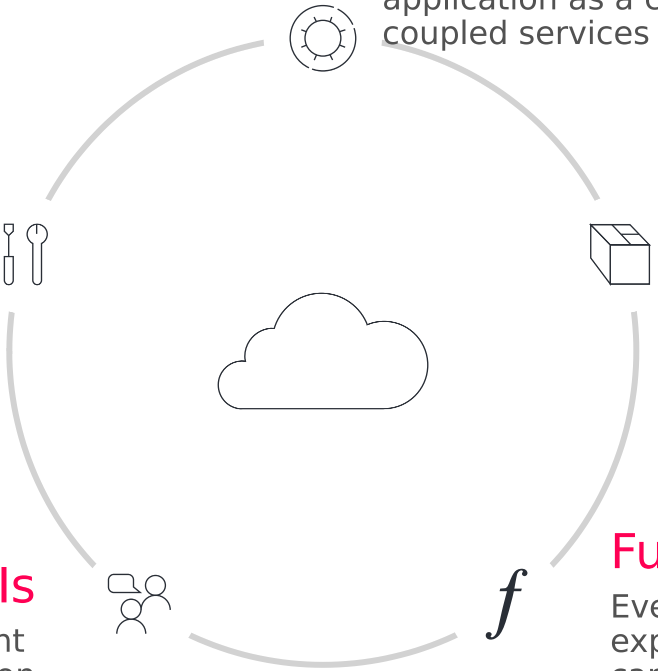
Cloud native application development

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds.


Cloud native architecture
AI powered UI and application logic
Integration using connectors and API management
Fully managed data
Software delivery using DevOps practices
Operations—as automated as possible
Multi-layered security

https://azure.microsoft.com/en-us/overview/cloudnative/
CNCF Cloud Native definition: https://github.com/cncf/toc/blob/main/DEFINITION.md
CNCF Cloud Native glossary: https://github.com/cncf/glossary/blob/main/cloudnative-glossary.pdf

CLOUD NATIVE ARCHITECTURE ELEMENTS



Microservices

Architectural approach to developing an application as a collection of modular, loosely coupled services

Containers

Standard deployment format to abstract code from underlying infrastructure differences

Functions

Event driven compute-on-demand experience that extends with capabilities
to implement code triggered by events

APIs

Expose services as light-weight APIs for easier integration

DevOps

People, processes, and technology that promote collaborative building and delivery

alpega
Shaping Transport Collaboration

BUSINESS INITIATIVES AND TECHNOLOGY

Innovation

App Modernization

Speed & Efficiency

MICROSERVICES The new architecture for modern apps

DEVOPS The new operating model

CONTAINER The new way to delivery apps

CLOUD Driving down the cost of infrastructure

spring BOOT

OPENSHIFT docker redhat

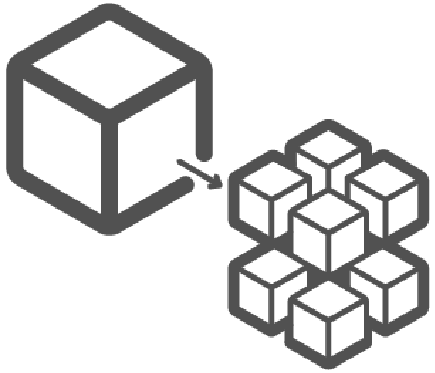
kubernetes

AWS CloudFormation

API Gateway

Amazon Elasticsearch Service

MICRO SERVICE ARCHITECTURE



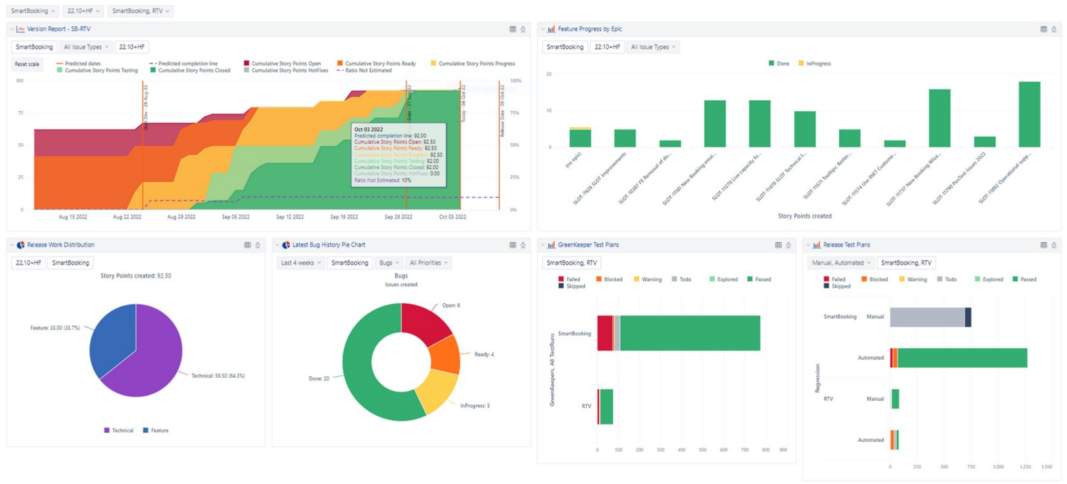
- The benefit of decomposing an application into different smaller services is that it improves modularity
- Easier to understand, develop, test, and become more resilient to architecture erosion
- Parallelizes development by enabling small autonomous teams to develop, deploy and scale their respective services independently
- Isolated for measurements - helps to quickly see where are the bottlenecks and where improvements should be made
- Sounds easier than it actually is – distributed monolith

DECOUPLE DEVELOPMENT AND RELEASE

Part of being Cloud-native is the abilitiy to ship as often as possible

We ask our teams everday:
Can we ship today?

Build health
Feature readiness
Bug status
Test status



/ Architecture blueprints

Credit CC Pixabay

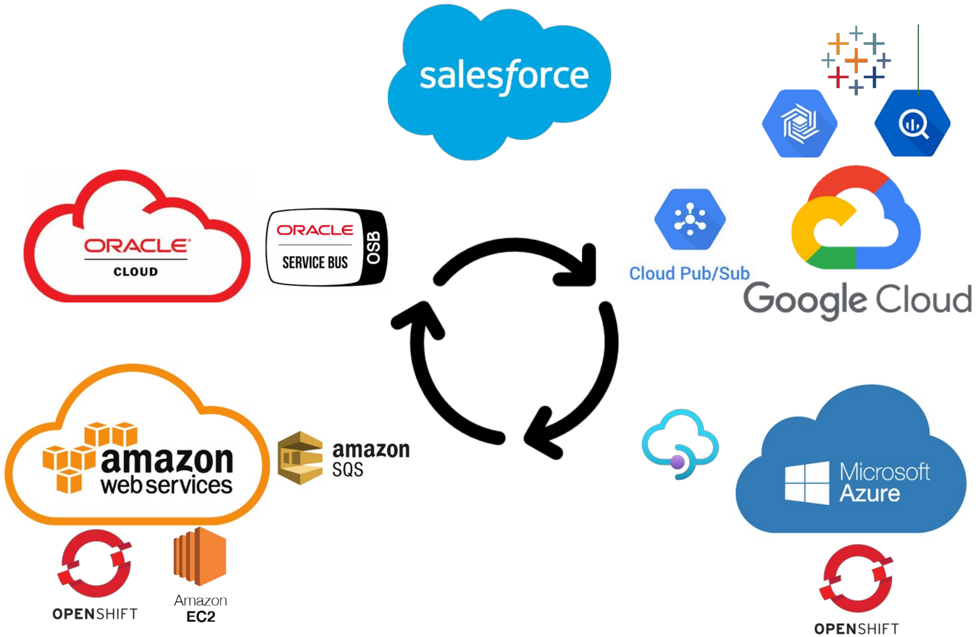
A TRUE MULTI-CLOUD ENVIRONMENT

Oracle Kubernetes Engine

Operating OKE clusters as we want to stick to managed services in each cloud; OCI Azure Interconnnect might allow us ARO usage in future

Google Kubernetes Engine

GKE for testing purposes only, as we currently use Google Cloud solely for our business intelligence and data science landscape



RedHat Open Shift on Amazon

Operating 3 ROSA clusters plus currently migrating from self-managed Openshift clusters and pure EC2 installations

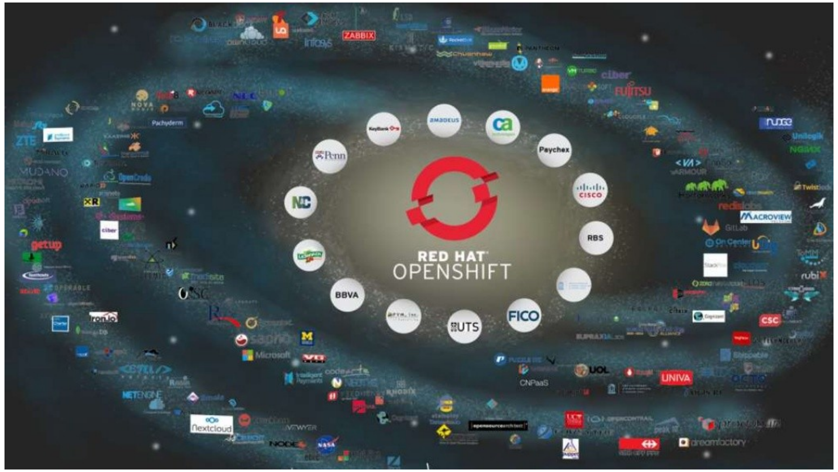
Azure RedHat OpenShift

Operating 3 ARO clusters that we originally migrated from AWS self-managed clusters to Azure

WHY OPENSHIFT CONTAINER PLATFORM

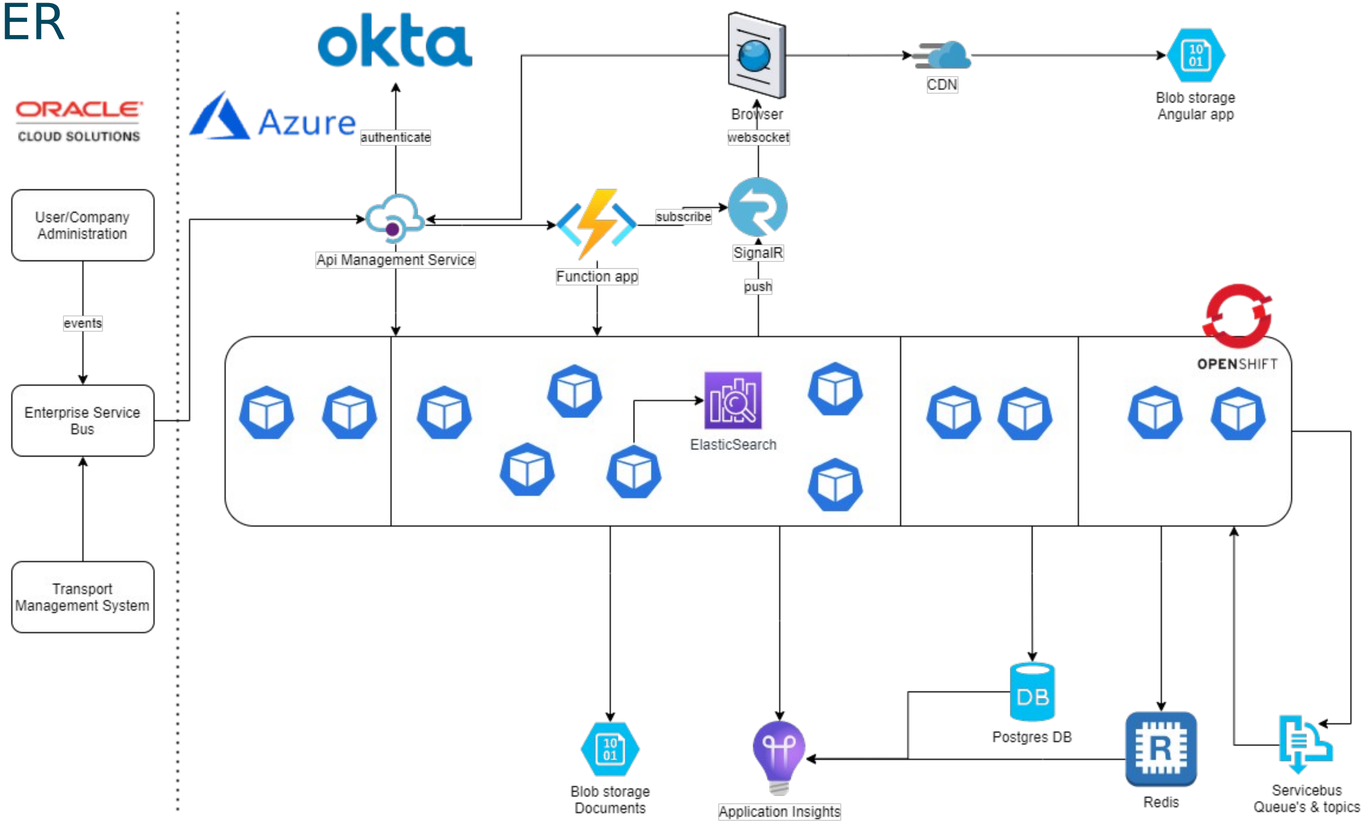
OPENSHIFT - Platform as a Service

- Allow best cooperation between development and infrastructure
- Produces faster complex platforms
- Low running costs
- Active and broad community
- Allowed us seamless migration from AWS to Azure

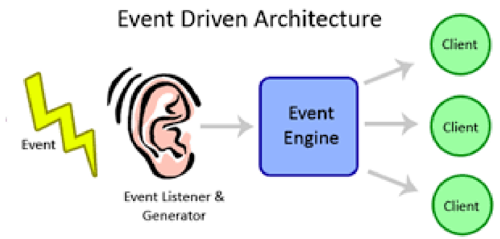


ARCHITECTURE BLUEPRINT

STANDARDIZING SERVICE ARCHITECTURE PER CLOUD PROVIDER



EVENT DRIVEN APPLICATION



Events

- **BookingAccepted**
- **BookingRejected**
- **BookingUpdateAccepted**
- **BookingUpdateRejected** (an update of an existing booking failed due to the restrictions applied to the new context)
- **SlotUnavailabilityDetected**

- **BookingCancelled** (when user cancels the booking, or a template update cancels a pre-booking)

Event Driven

very intuitive and naturally well-suited to applications where a
are based on events
(slot bookingAccepted/bookingUpdateAccepted/SlotUnavailabi

> concurrency is easily handled without any locking
> track history of booking actions

Our system has therefore an « Eventual consistency »

Concept to view and update data in « high concurency » environment without locking/slowness

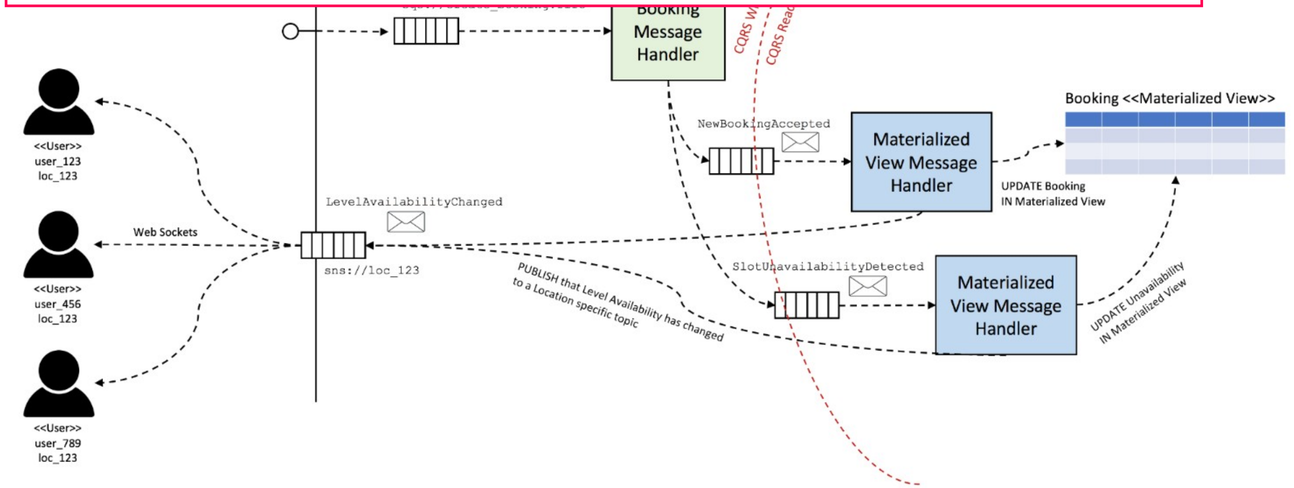
Consistency = All clients see the same data, even with concurrent updates.
High Availablity = All clients can access some version of the data
Tolerance = a liveness guarantee ~ (informally guarantees that, if no new updates are made to a given data item, eventually all accesses to that item will return the last updated value).

BOOKING DESIGN –CQRS PATTERN

CQRS*: separate writing from reading – different technologies applied

- Often reading data is much more frequent than writing.
- Reading data we typically retrieve a larger amount of data or lists of data compared to writing that should affect one aggregate only.
- Reads from a user perspective has to be more performant than writes. User tends to find it easier to accept a slower response when data is changed.

*(*CQRS=Command Query Responsibility Segregation)*



Booking Message Handler
CQRS W
CQRS Read
Booking <<Materialized View>>
NewBookingAccepted
Materialized View Message Handler
UPDATE Booking IN Materialized View
<<User>> user_123 loc_123
Web Sockets
LevelAvailabilityChanged
sns://loc_123
PUBLISH that Level Availability has changed to a Location specific topic
SlotUnavailabilityDetected
Materialized View Message Handler
UPDATE Unavailability IN Materialized View
<<User>> user_456 loc_123
<<User>> user_789 loc_123

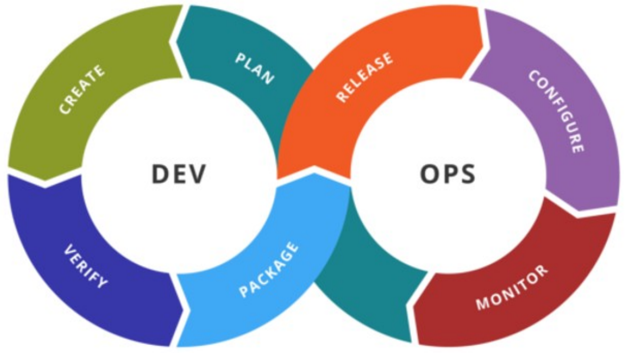
/ A word on CI/CD

©credit CC Pixabay

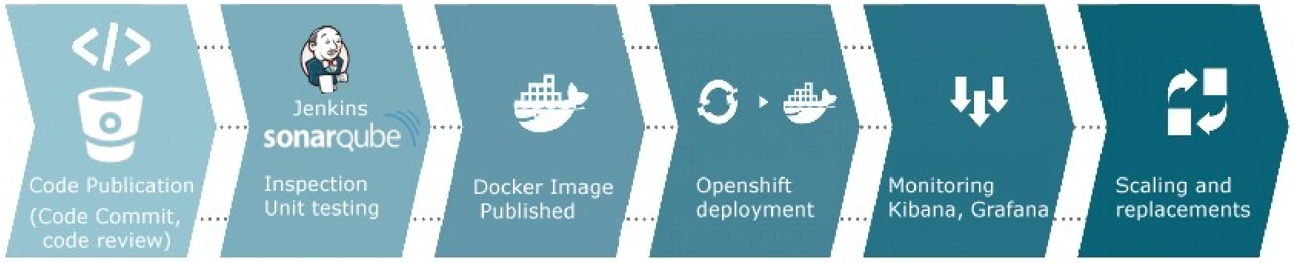
DEVOPS & CONTINUOUS DELIVERY

DevOps = the union of people, process, and technology to continually provide value to customers.

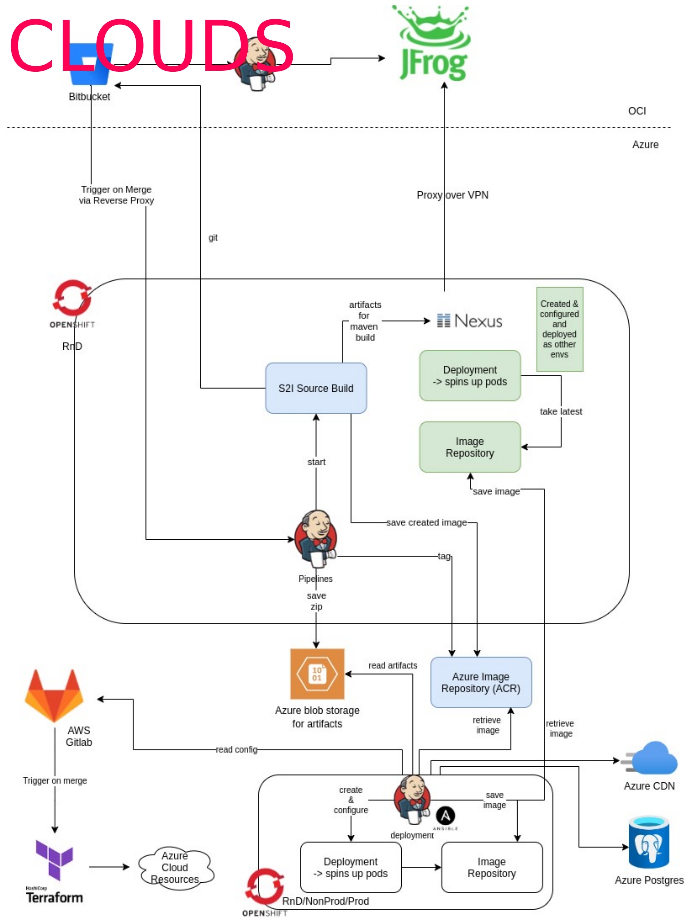
Continuous Delivery = every code change is built, tested, and then pushed to a non-production testing or staging environment. The difference between continuous delivery and continuous deployment is the presence of a manual approval to update to production.



Continuous delivery via Jenkins as orchestrator for OpenShift

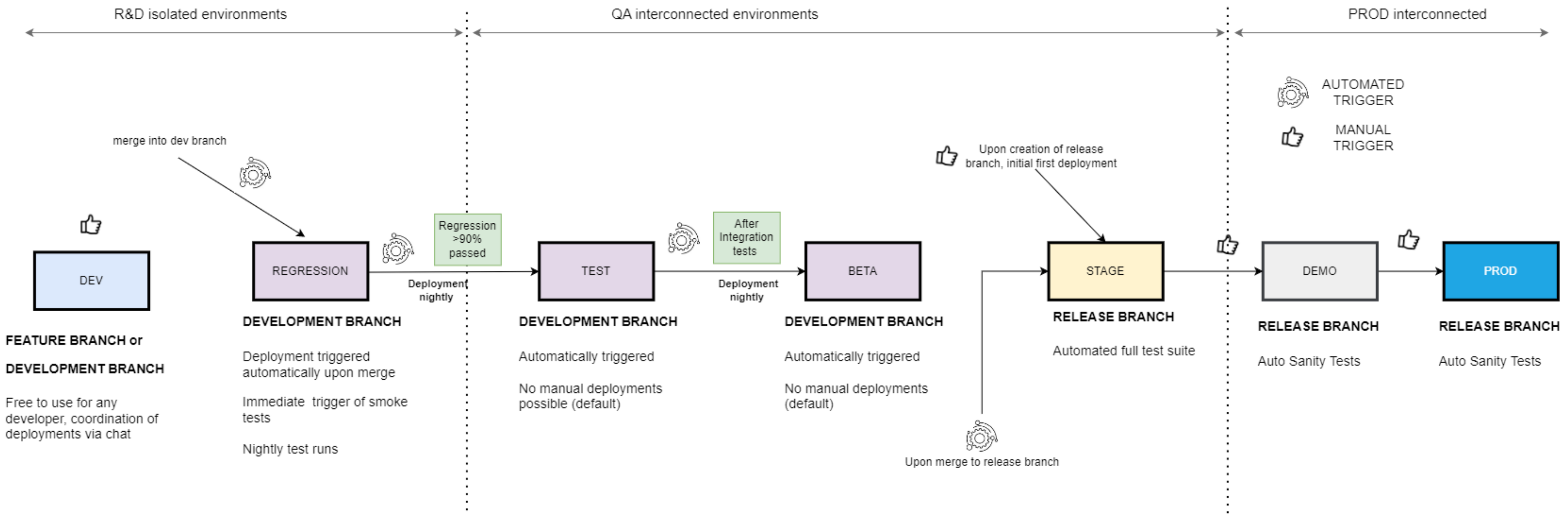


PIPELINES SPREAD ACROSS THE CLOUDS



- **Jenkins** as orchestrator of the whole process (feature complete, huge set of plugins)
- **Seperated build** and deploy pipelines
- **Seperated Openshift clusters** (build in RnD, deploy in rest of the clusters)
- **Terraform** for all cloud ressource provisioning

AUTOMATED PIPELINES UPON MERGE



FEATURE BRANCH or

DEVELOPMENT BRANCH

Free to use for any developer, coordination of deployments via chat

DEVELOPMENT BRANCH

Deployment triggered automatically upon merge

Immediate trigger of smoke tests

Nightly test runs

DEVELOPMENT BRANCH

Automatically triggered

No manual deployments possible (default)

DEVELOPMENT BRANCH

Automatically triggered

No manual deployments (default)

RELEASE BRANCH

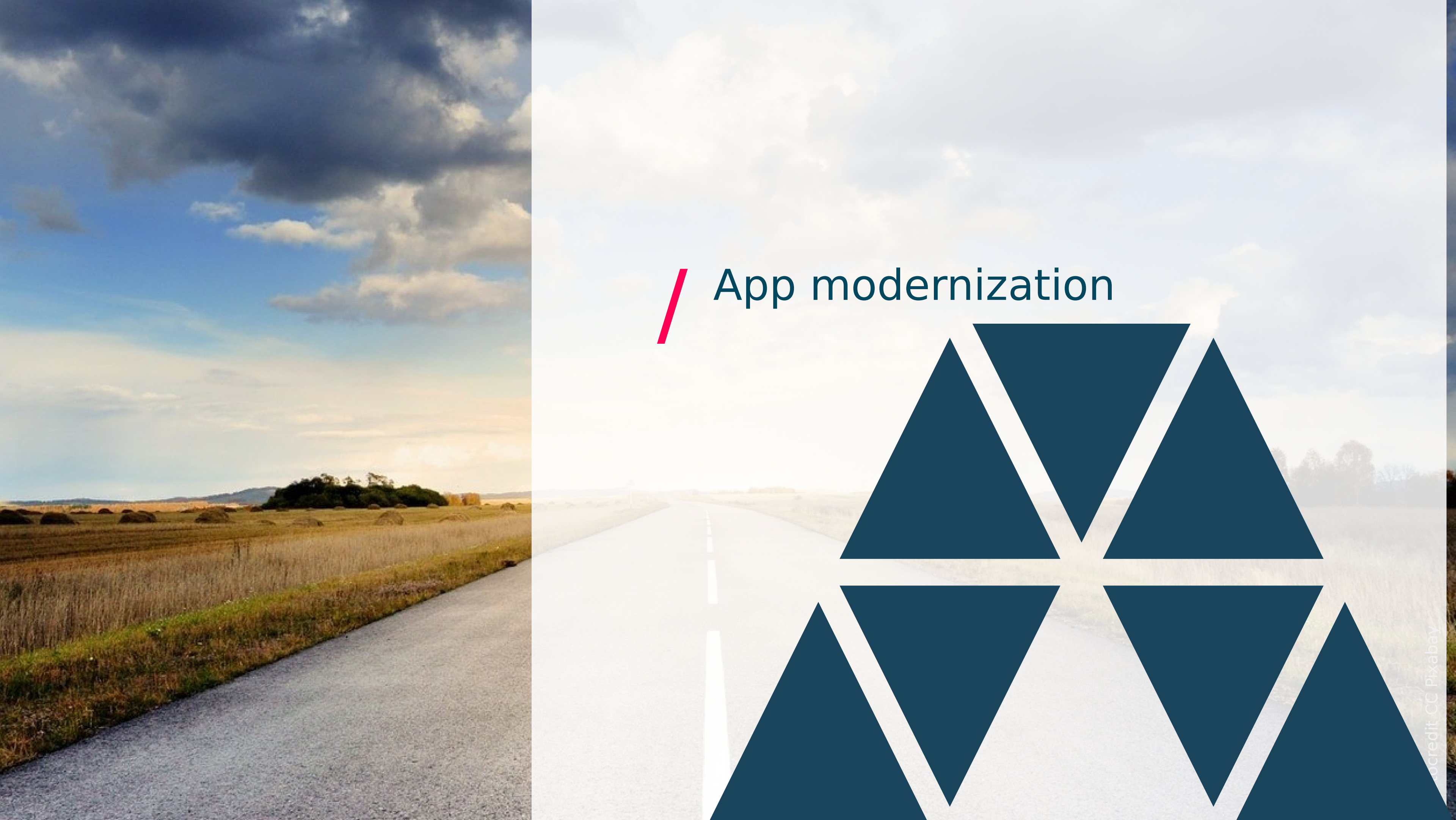
Automated full test suite

RELEASE BRANCH

Auto Sanity Tests

RELEASE BRANCH

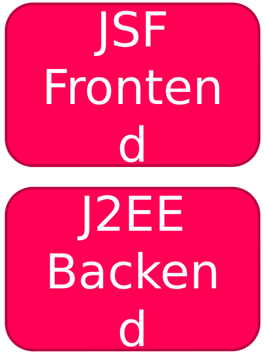
Auto Sanity Tests



/ App modernization

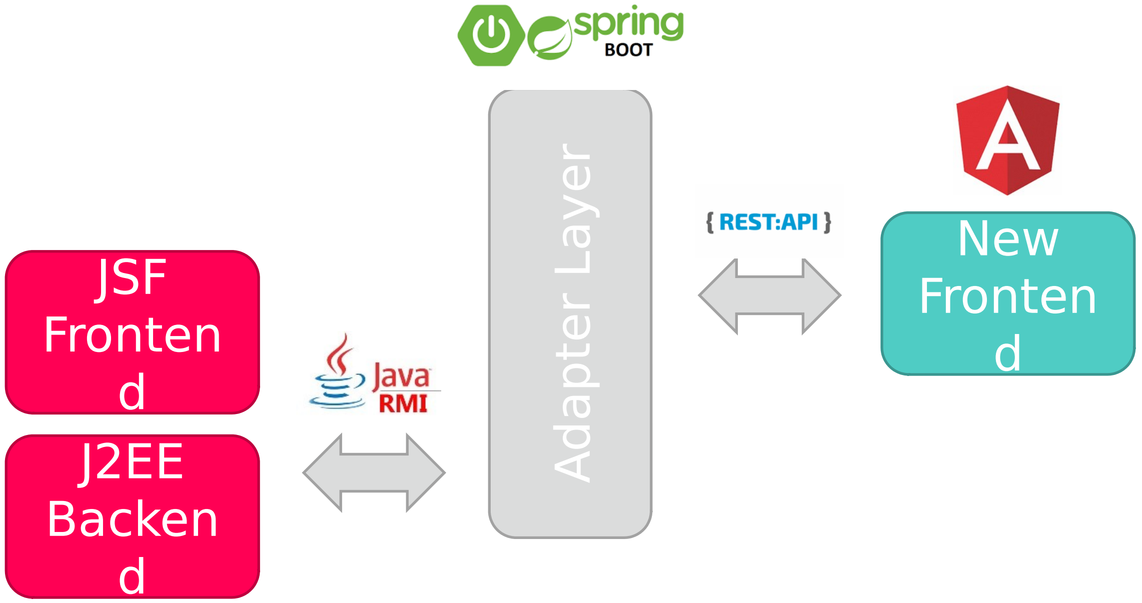
HOW TO MOVE LEGACY TO NEW ARCHITECTURE?

GOAL IS TO MIGRATE STEP BY STEP TO NEW ARCHITECTURE



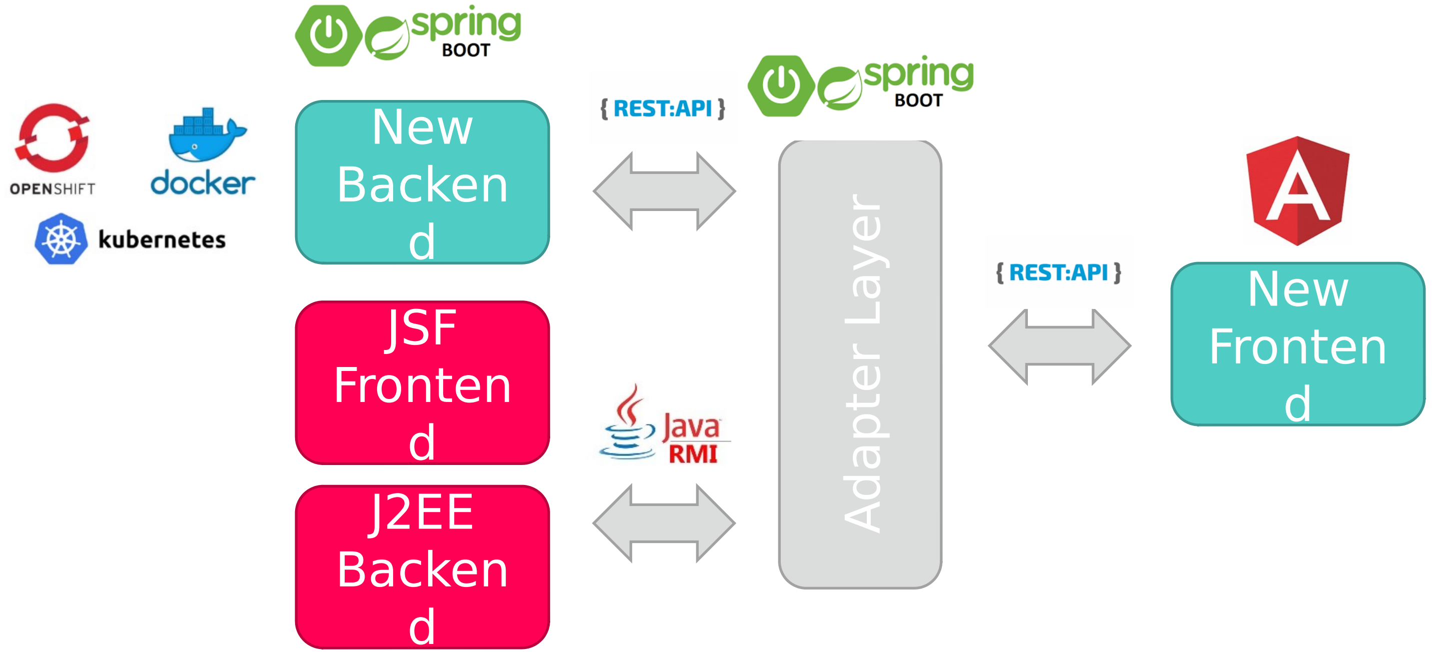
HOW TO MOVE LEGACY TO NEW ARCHITECTURE?

INTRODUCE AN ADAPTER LAYER



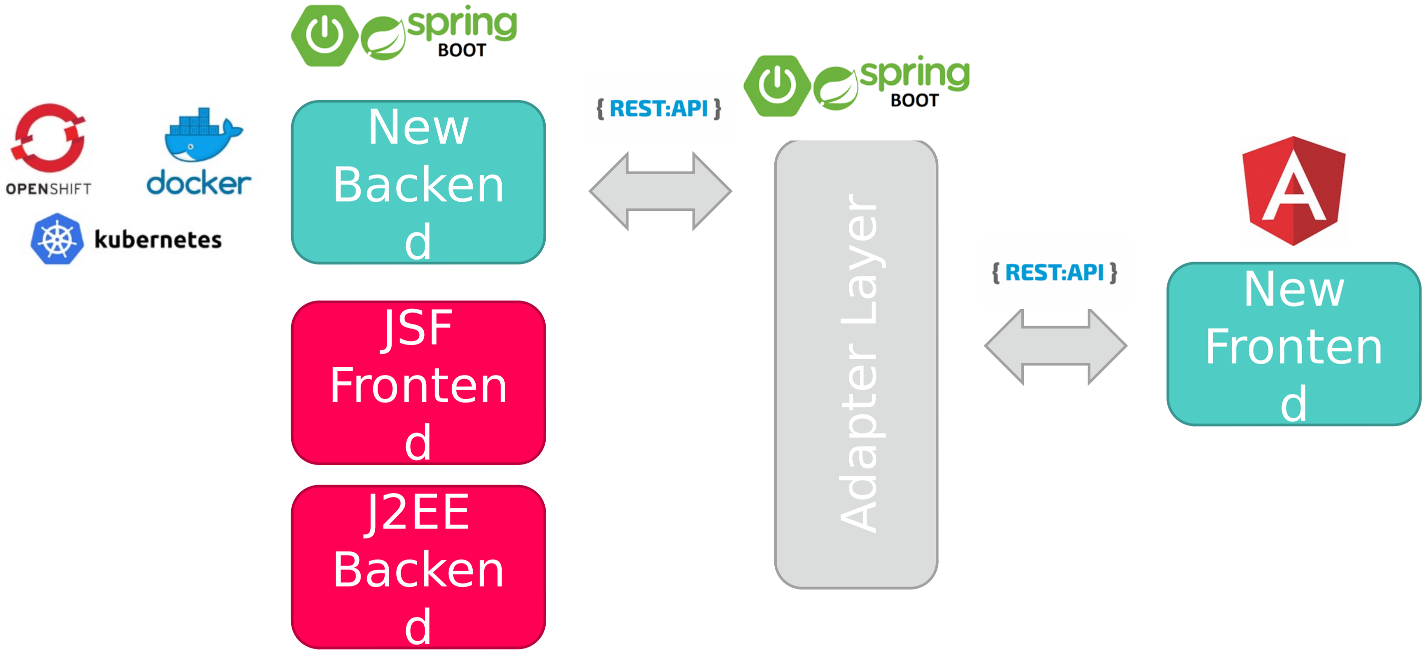
HOW TO MOVE LEGACY TO NEW ARCHITECTURE?

BUILD UP THE NEW BACKEND



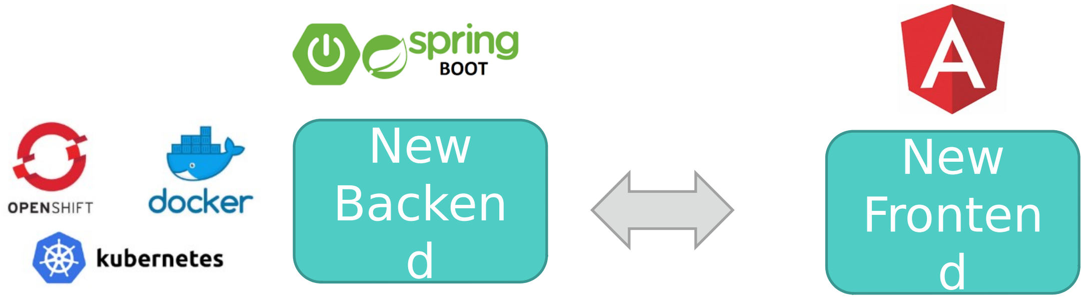
HOW TO MOVE LEGACY TO NEW ARCHITECTURE?

CUT OLD BACKEND ONCE NEW IS READY, KEEP IT FOR MIGRATION



HOW TO MOVE LEGACY TO NEW ARCHITECTURE?

FINAL PICTURE AFTER MIGRATION



/ Summary and outlook

IT‘S A JOURNEY…

alpega
Shaping Transport Collaboration

SUMMARY AND NEXT STEPS

UI/UX

Testing

Architectur
e

- Defined a blueprint for modern architecture
- Defined a blueprint for legacy migration
- Continuously work in simplifying our cloud integration and infrastructure
- Automate as much as possible

Thank you!

Mag. Stefan Heil

Email

stefan.heil@alpegagroup.com

Phone

+43 664 162 40 70

LinkedIn

https://www.linkedin.com/in/stefanheil/

Director of Engineering