OPENSHIFT @Amadeus

Switching to a containerized application platform
Amadeus is a technology company dedicated to the global travel industry.

We are present in 195 countries with a worldwide team of more than 12,000 people.

Our solutions help improve the business performance of travel agencies, corporations, airlines, airports, hotels, railways and more.
Connecting the entire travel ecosystem

- Cruiselines
- Hotels
- Car rental
- Ground handlers
- Ferry operators
- Airports
- Ground transportation
- Insurance companies
- Travel agencies
- Airlines
Supporting the entire traveller life cycle

- Pre-trip
- On trip
- Post-trip
- Inspire
- Search
- Buy/Purchase
Robust global operations

- **1.6+ billion** data requests processed per day
- **525+ million** travel agency bookings processed in 2014
- **695+ million** Passengers Boarded (PBs) in 2014
- **95%** of the world’s scheduled network airline seats

No single point of failure

'Follow The Sun’ operational concept: global support and local knowledge

Aim: 100% service availability

Security drives everything we do
Amadeus Constraints

High consistency & low response times

Reservation Record / Content Provider Inventory

Updated:
- From all around the world
- Concurrent updates
- By several systems (airlines, hotels, travel agency, etc...)

Need for:
- Very high-consistency: transaction based
- Low response times
Amadeus Constraints

High volume

December 2014 (customer + internal traffic):
- At peak: \(~210\,000\) queries per second
- Average: \(~145\,000\) queries per second

Thousands (and thousands) of application servers

\textbf{100+ TB} of compressed data logged every day
Amadeus System

Where we are

- Large distributed system (SOA)
  - 5000+ “micro” services

- One data-center + disaster-recovery sites

- Thousands of servers
  - x86-64 Linux servers
  - Pre-configured upfront for specific tasks
  - N+x model as servers can’t be replaced quickly
  - Roles cannot be changed
Amadeus System

Where we were

- Large distributed system (SOA)
  - 5000+ “micro” services
- One data center + disaster recovery sites
- Thousands of servers
  - x86-64 Linux servers
  - Pre-configured upfront for specific tasks
  - N+x model as servers can’t be replaced quickly
  - Roles cannot be changed

Despite the number of servers largely managed as pets
Amadeus System

Where we are going

- Closer to our customers
  - Reduce the latency

- Remote operations
  - Install appliances within customer premises when appropriate

- Multiple data-centers / cloud capability

- Evolve our data-center model
  - Better use of resources
  - Flexibility/Agility
  - Higher availability
Amadeus System

Where we are going

- Closer to our customers
  - Reduce latency
- Remote operations
  - Install appliances within customer premises
- Multiple data centers
- Evolve our data center model
  - Better use of resources
  - Flexibility/Agility
  - Higher availability

Paradigm Shift
Deploy the application **as a whole**
With **all** its dependencies

- Reproducibility
- Homogeneity
- Technology agnostic
- Simplify operations
Automated Scheduling

- Manage a cluster **as a whole**
  - Any task on any server
  - Start tasks in the cluster

Decouple **what** should run from **where** it should run
Technological Stack

OpenShift

Linux

Docker

Kubernetes
Technological Stack: **OPENSHIFT**

Docker + Kubernetes

- Linux Container Technology
- Container image format
- Easy deployment

**Technological Stack:**

- **docker**
  - Linux Container Technology
  - Container image format
  - Easy deployment

- **kubernetes**
  - Orchestration of Linux containers
  - “Manage a cluster of Linux containers as a single system”
  - Automatic placement, self-healing
Partnership with Redhat

_ Started a year ago

_ Software engineers
  • From Amadeus embedded in Redhat teams
  • From Redhat embedded in Amadeus teams

_ Open Source contributions
OPENSHIFT v3

Nodes (aka minions)
Pods: scheduling and scaling unit
OPENSshift v3

5 instances of my POD
OPENSHIFT v3
OPENSHIFT v3
About pods...

Always same content

Always listening on same ports
About pods...

Always same content

Always listening on same ports

Each pod instance gets its own IP address

https://github.com/openshift/openshift-sdn
Pods labeling

Key/value pairs attached to a pod

Labels allow for whatever organizations or conventions you want

Example:
- "environment": "CustomerFacingTest",
  "environment": "Production", etc...
- "customer": "X", "customer": "Y"
How to target a service that exists on multiple hosts and can change location?
**OPENSHIFT Services**

- Service represents a group of pods

```
"ports": [
  {
    "protocol": "TCP",
    "port": 80
  }
]
```

- List of Ports

- A way to select the endpoints: **label selectors**
  - Endpoints are loosely coupled with the service

```
"selector": [
  {
    "environment": "production",
    "app": "myGreatApp"
  }
]
```
OPENSHIFT Services

Endpoints list maintained by API Server
• Endpoint = IP address for a pod

REST API

Watchable for addition/removal of endpoints

Add-on: DNS server watching API server
• Name resolution of services possible
What OPENSHIFT does for us
Operational Model

Homogenization: everything looks the same!
Designed to Accommodate Constraints

- Not limited to HTTP

- We use our own protocol over TCP
  - Kind of HTTP/2 but before it was invented

- What we needed to keep:
  - Low latency
  - Keeping connection open
  - Multiplexing
Easy deployment
Easy deployment

Pod v1

libvirt

Laptop
Easy deployment

Pod v1

Variety of IaaS

Private/public clouds
Easy deployment
Easy deployment

Scale what ran on developers laptops

Pod v1
Openstack
Production
Easy deployment

Unified Process

Pod v1
libvirt
Laptop

Pod v1
Variety of IaaS
Private/public clouds

Pod v1
Openstack
Production
To conclude...
Key takeways

Building a cloud-ready system with OPENSHIFT
- Macro-management
- Self-healing / resiliency
- Dynamic system
- Multi-cloud/data-center capability

Uniform operations
- Simplification
- Reduced operational burden
- Enable transition to DevOps model

Great collaboration with Red Hat
Thank you