SBB CFF FFS

Sind Container die Virtualisierung der Zukunft?

Baltisar Oswald
Szabolcs Szádeczky-Kardoss

BAT Nr. 36, 03.03.2017

SBB CFF FFS

Was bewegt die SBB?

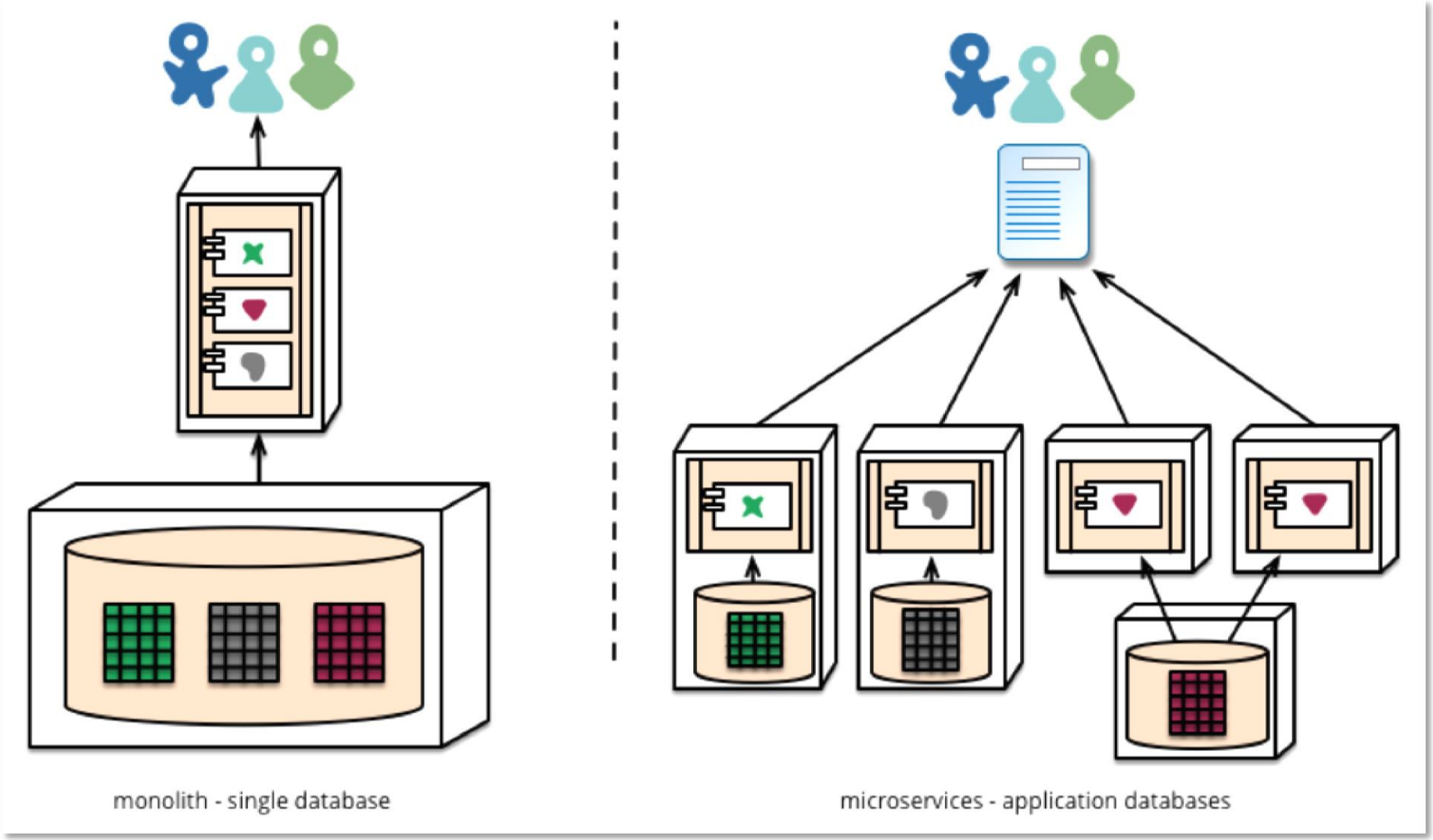
- Konzern
  - SBB Positionierung als Mobilitätsintegrator von Tür-zu-Tür
  - Intermodaler Wettbewerb
    - Fernbusse, Uber, Selbstfahrende Fahrzeuge, BlaBlaCar, …
  - Digitalisierung (2015-)
- SBB-IT (-2015)
  - Shared Plattformen (WAS, Oracle, WMQ, …)
  - Monolithen-Anwendungen
  - Starre ITIL-Prozesse ⇔ Time-to-Market
  - Betriebsstabilität

SBB CFF FFS

Was bewegt die SBB-IT?

- → Bottom-Up
  - Agile Entwicklungsteams (~2014)
  - Kleinere PoCs in der Public Cloud (2015)
  - Neue Trends in der Community:
    - Microservices
    - Containerisierung
- → Top-Down
  - IT-Teilstrategie Cloud-Computing (2014)
  - Beschaffung VIAS (2014), VIPS (2016), Public Cloud
  - Technologiestrategie (2016)
  - Agile Transformation (2016-)

Microservices



Quelle: http://martinfowler.com/articles/microservices/images/decentralised-data.png

Die Herausforderung

Multiplicity of Stacks

User DB

Queue

Static website

Analytics DB

Web frontend

API endpoint

Background workers

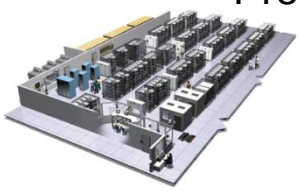
Do services and apps interact appropriately?

Multiplicity of deployment environments

Development VM

Public Cloud

Production Cluster



QA server

Disaster recovery

Customer Data Center

Contributor's laptop

Production Servers

Can I migrate smoothly and quickly?

Quelle: https://www.slideshare.net/dotCloud/why-docker2bisv4

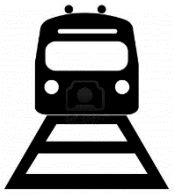
SBB CFF FFS

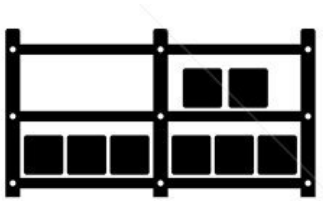
Ein ähnliches Problem: Güterverkehr vor 1960

Multiplicity of Goods

Do I worry about how goods interact (e.g. coffee beans next to spices)

Multiplicity of transportation / storing methods





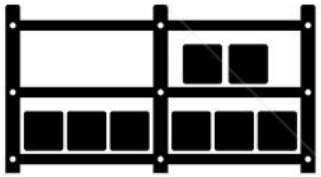
Can I transport quickly and smoothly (e.g. from boat to train to truck)

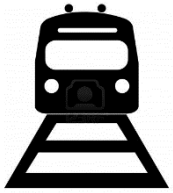
Quelle: https://www.slideshare.net/dotCloud/why-docker2bisv4

Die Lösung: ISO-Container (Intermodal shipping container)

A standard container that is loaded with virtually any goods, and stays sealed until it reaches final delivery.

…in between, can be loaded and unloaded, stacked, transported efficiently over long distances, and transferred from one mode of transport to another



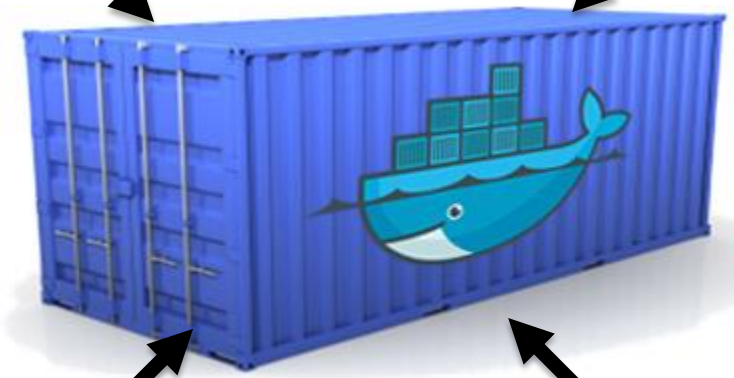


Quelle: https://www.slideshare.net/dotCloud/why-docker2b1sv4

Containerisierung

Static website User DB Web frontend Queue Analytics DB

An engine that enables any payload to be encapsulated as a lightweight, portable, self-sufficient container…

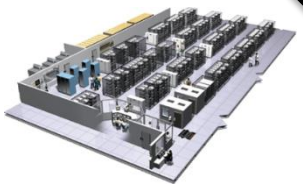


…that can be manipulated using standard operations and run consistently on virtually any hardware platform

Development VM

QA server

Customer Data Center



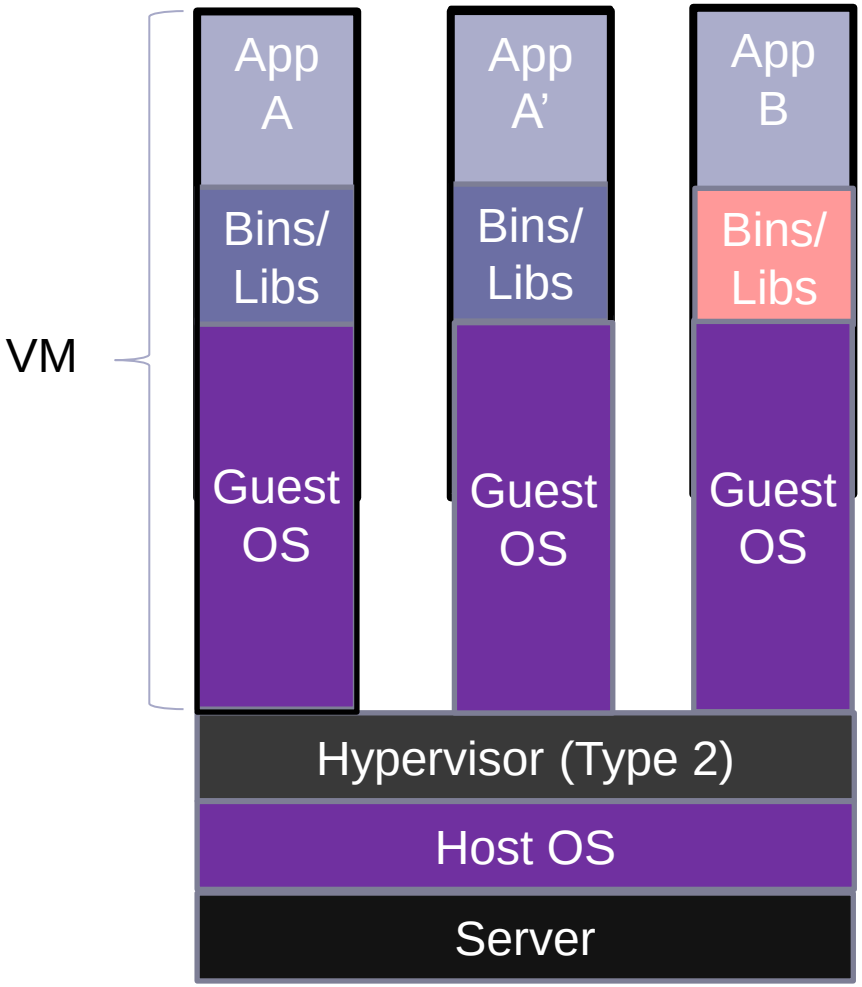
Public Cloud

Production Cluster

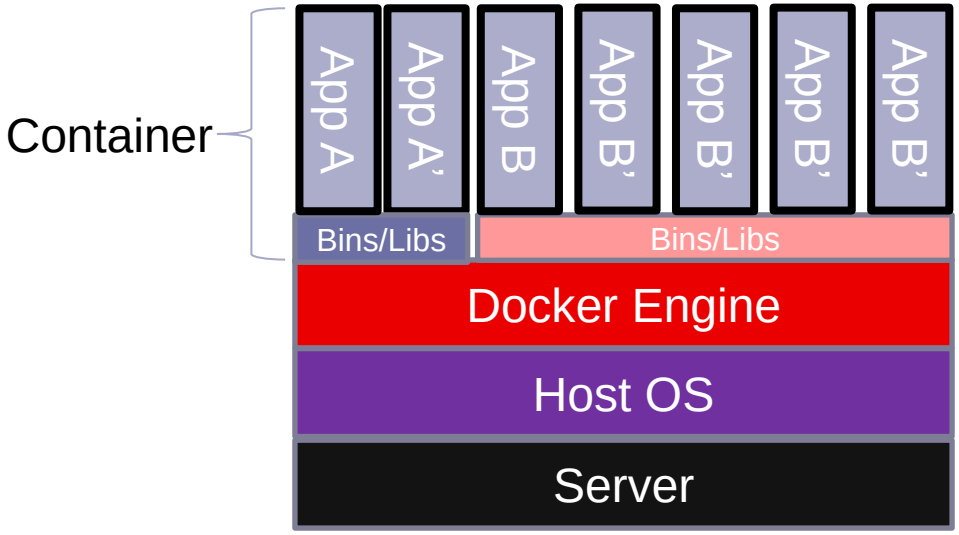
Contributor's laptop

Quelle: https://www.slideshare.net/dotCloud/why-docker2bisv4

VMs vs. Containers



Containers are isolated, but share OS and, where appropriate, bins/libraries



Quelle: https://www.slideshare.net/dotCloud/why-docker2bisv4

Agile Transformation

- In der Software-Entwicklung präsent seit mehreren Jahren
  - Speerspitze: u.a. das Team WZU
  - Tools: Git, Jenkins, Nexus, JIRA, Confluence, Ansible, …
    … und **Docker**
- Es ist aber weniger eine technologische Frage
  - Denkweise
  - Organisation und Prozesse (auch ausserhalb der IT)
- Plattform kann auch viel zu kürzeren Durchlaufzeiten beitragen
  - Continuous Deployment
  - Self-Service (bis in die Produktion)

SBB CFF FFS

Cloud Stack als Technologie Stack

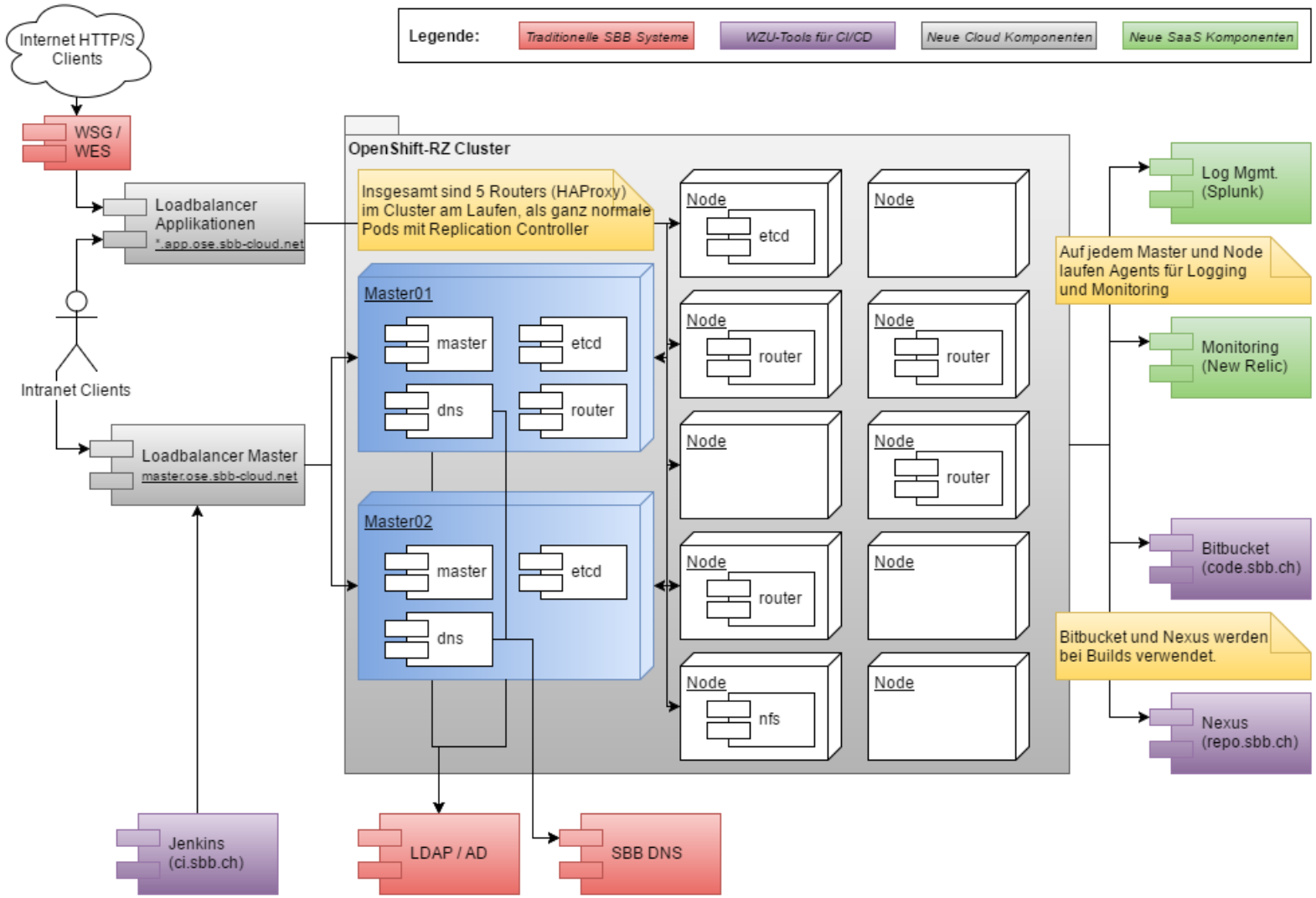
→ Vision

- Ich brauche eine Entwicklungs- und Betriebsumgebung für eine neue Anwendung.
- Ich gebe folgende Parameter ins Bestellportal ein: Name der Anwendung, gewünschter Technologie Stack.
- Innert Sekunden erhalte ich durchgängige Umgebungen von Entwicklung bis Produktion mit automatisierten Transport- und Testprozessen sowie den entsprechenden virtuellen Entwicklungsarbeitsplatz.
- Minuten später arbeite ich am Code und starte meine ersten Deployments. Das Team und externe Partner können mit mir virtuell zusammenarbeiten.

Docker + Kubernetes + OpenShift

- Q2 2015: Docker
  - Docker ist cool, aber es braucht eine Orchestrierung
- Q3 2015: Docker + Kubernetes 1.0
  - Kubernetes ist auch schön, aber manche Enterprise-Features fehlen uns
- Q4 2015: Docker + Kubernetes + OpenShift 3.0
  - Builds and Deployments für Entwickler
  - Enterprise-fähige Verwaltung von Images
  - Enterprise-fähiges Applikationsmanagement durch Admin-GUI, Namespaces, Quotas, Authentisierung und Authorisierung, Netzwerkzonierung, Port-Management
- Heute: OpenShift 3.2 produktiv (bald 3.3)

OpenShift 3.2



OpenShift Plattform in Zahlen

Typ	Anzahl
Projekte (Namespaces)	400
Laufende Container (Pods)	1400
Services	1100
Routen	900
Deployments pro Tag	210
User (Entwickler und Applikations Manager)	300

SBB CFF FFS

OpenShift - Demo

Lessons learned

- Container per Design «Infrastructure as Code»
- 100%ige Automatisierung
- Software Entwicklungs-Methoden für Plattform Engineers
- Keine SLA auf laufende Prozesse, sondern «nur» auf die Self-Healing Mechanismen
  - Restarts von Prozessen sind die Normalität (7*24 «Wartung»)
  - «Time to repair» und automatisches «Repair» als Schlüssel für Betriebsstabilität
  - Skalierung ausschliesslich auf horizontaler Ebene.
  - Viele Technologien und Applikationen unterstützen diese Art von SLA noch nicht

Ausblick

Container Plattform als Ablösung der klassischen Virtualisierung

- → Fokus fürs 2017
  - Windows Container
  - COTS Anwendungen in Container
  - BigData Integration
  - GPU Anwendungen
  - RDBMS in Container (HA)

- → Container und Containerorchestrierung ist die Zukunft
  - «Wo/wie die Workloads laufen, wird egal sein»