

Kapitel 8: Cloud Native Softwarearchitektur



Vorlesung
**CLOUD
COMPUTING**

Walmart

Supercentre



Enter

Exit

Low Prices
\$2



2002

WAL-MART®

Help | Your Account | Cart & Checkout

Electronics Photo Movies Music Books Toys Home Garden & Hardware Sporting Goods Gifts Jewelry & Watches See All Departments

Search Entire Site for: Find

Roses on Rollback! One Dozen Fresh-Cut, Long-Stem Roses Only \$24.88

Home Page

Electronics

- Computers & Printers
- Internet Service
- TV, DVD & Video

More...

Photo

- Digital Cameras
- Camcorders
- Photo Center

More...

Movies 12,000 Movies!

Music 80,000 CDs!

Books 500,000 Titles!

Video Games

- PS2
- Xbox

More...

Toys

- Outdoor Play
- Bikes & Scooters

More...

Baby & Baby Registry

- Shop by Age or Stage

Home

- Furniture
- Appliances
- Bed & Bath

More...

Garden & Hardware

- Patio & Grilling
- Lawn Equipment
- Home Improvement

More...

Tires

Sporting Goods

- Exercise & Hot Tubs
- Camping & Hiking

More...

Gifts

New customer?
Click here to get started.

Find 300,000 Textbooks
Every Day Low Prices

Find it by ISBN or category.

- Business
- Education
- Test Prep
- Medical
- Engineering
- Computer

See all textbooks.

Furnish the Dorm

Find the desk that fits your living space and study habits.
See all desks.

Sleep well with our low prices on bedding.
See all college bedding.

Make dorm life easier with refrigerators, dishwashers and more.
See all small appliances.

Stay in Touch

Low prices on notebook PCs and printers.

Get online for only \$9.94 a month.

Call anytime for as low as 4 cents per minute.

Back To College

Portable Electronics
Cameras
Shopping Cards
See More

K-12

Clothing
School Supplies
See More

WAL-MART Services

- Upload Digital Photos
- Find a Gift Registry
- Find and Order Tires
- Refill Prescriptions

Kategorien

Warenkorb

Registrierung

Artikel

Dienste

2008

Free Shipping with  site to store

Store Finder ▾ Local Ad ▾ Top Products ▾ Sign In/New ▾ Track Orders ▾

See All Departments

- Apparel & Jewelry ▾
- Baby ▾
- Books, DVDs & MP3s ▾
- Electronics ▾
- For the Home ▾
- Gifts & Registries ▾
- Grocery ▾
- Health & Beauty ▾
- Outdoor Living ▾
- Pharmacy ▾
- Photo ▾
- Sports & Fitness ▾
- Toys & Video Games ▾
- In Stores Now ▾
- See All Departments

Savings Add Up

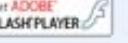
- Clearance
- Rollbacks
- \$4 Prescriptions
- Video Games Under \$20
- Clearance on Baby Items

What's New

- Classifieds
- Halloween
- Elevenmoms
- Better Homes

GREAT VALUE, NEW PRODUCTS

Download Flash Player version 7 or later for the ultimate Walmart.com shopping experience.



Blu-ray Price Drop

★★★★★ Magnavox from \$258
Sony Samsung

See all: [Blu-ray Players](#)

50% Off Top 50 Books

The 50 top sellers are 50% off — get great reads for the whole family, from \$4.
Shop: [50% Off Top 50 Books](#)

Save on Top-Brand Electronics



http://ad.yieldmanager.com/st?ad_type

Zusätzliche Dienste

2011

Shop for College 2011

New customer? Sign In | Help

Value of the Day | Local Ad | Store Finder | Registry | Gift Cards

Track My Orders | My Account | My Lists

Search All Departments ▾ Search Go My Cart ▾ Choose My Store ▾

Must-Have College Essentials at Can't-Miss Prices COLLEGE 2011

Bedding from \$35	Futons & Sofas from \$99	Storage from \$19	Desks from \$39
			
Laptops & Netbooks from \$278	Appliances from \$57	TVs from \$149	iPods from \$49
			

Shop for College 2011

ADVERTISEMENT

ADVERTISEMENT


Back to School Essentials
From uniforms to supplies, find everything for school at Every Day Low Prices.


Have Your Fun in the Sun
Enjoy hot summer savings on everything you'll need for some outdoor family fun.


School Supplies
Uniforms


Pools & Waterslides
Swing Sets
Trampolines


Watch New Releases Instantly



Zusätzliche Dienste (z.T. personalisiert)

Bad News.

CHALLENGE

Four years ago, the Walmart Global eCommerce system was a monolithic application, deployed once every 2 months. This represented an unsustainable rate of innovation given the competitive climate. Walmart recognized the need to fundamentally transform technology delivery to compete in the Digital Economy.

Walmart auf <http://www.oneops.com>

- "[...] it was unable to scale for 6 million pageviews per minute and was down for most of the day during peak events."
- "This is the multi-million dollar question which the IT Department of Walmart Canada had to address after they were failing to provide to their users on Black Fridays for two years in a row."

<https://blog.risingstack.com/how-enterprises-benefit-from-microservices-architectures>

Lange Zyklen von Dev-to-Prod.

Mangelnde Skalierbarkeit.

Mangelnde Elastizität.

2016

Zusätzliche Dienste.

The screenshot shows the Walmart website homepage. At the top, there's a navigation bar with links for 'GIFT Cards', 'Registry', 'List', 'Wish List', 'New Products', 'Track Order', 'Credit Card', and 'Help'. Below the navigation is a search bar with a magnifying glass icon and a 'My Account' link. A large orange banner at the top left says 'FREE Shipping plus more! 40,000+ items today!' with a house icon. To its right is a red arrow pointing down with the text 'Rollbacks on iPad Air \$100 off 16GB & 32GB WiFi'. Below this, there's a 'Shop Now' button and a note about 10% off select items. In the center, there's a large image of two iPads. To the right of the iPads is a product detail for a 'George Foreman 12-Serving Indoor/Outdoor Grill', showing a price of \$49.95, a list price of \$79.99, and a save of \$30.04. It includes options for 'Core plan (optional)', quantity selection, and buttons for 'Add to Cart', 'Add to Registry', and 'Add to List'. Below the grill image is a section titled 'Customers also viewed these products' with several other items like a George Foreman 14-Serving Grill and a Better Homes and Gardens 13.3" Touch Screen TV.

Good News.

1000 Deployments pro Tag ...

... durch die Entwickler.

~ 100% Verfügbarkeit

Neue Geschäftsmodelle, Anwendungen und Geräte (IoT, mobile, APIs)

Ressourcen Sparsamkeit.

Elastische Skalierbarkeit.

RESULTS

Today the Walmart eCommerce platform is hosted in some of the largest OpenStack clouds and is managed exclusively via OneOps. On a typical day there are now over 1,000 deployments, executed on-demand by development teams, each taking only minutes on average to complete.

Walmart auf <http://www.oneops.com>

"They wanted to prepare for the world by 2020, with 4 billion people connected, 25+ million apps available, and 5.200 GB of data for each person on Earth. Walmart replatformed [...] with the intention of achieving close to 100% availability with reasonable costs."

<https://blog.risingstack.com/how-enterprises-benefit-from-microservices-architectures>

- "In fact, the organization reports that some 3,000 engineers [...] drive 30,000 changes per month to Walmart software."
- "Those new applications, which span everything from mobile devices to the Internet of things (IoT), are crucial weapons in a global e-commerce contest that pits Walmart against the likes of Amazon and Alibaba, as well as a host of other rivals that are emerging as the cost of entry into the online retail sector continues to decline in the age of the API economy."

<http://www.baselinemag.com/enterprise-apps/walmart-embraces-microservices-to-get-more-agile.html>

- "The Walmart [...] servers [...] were able to handle all mobile Black Friday traffic with about 10 CPU cores and 28Gb RAM."
- "On Thanksgiving weekend, Walmart servers processed 1.5 billion requests per day. 70 percent of which were delivered through mobile."

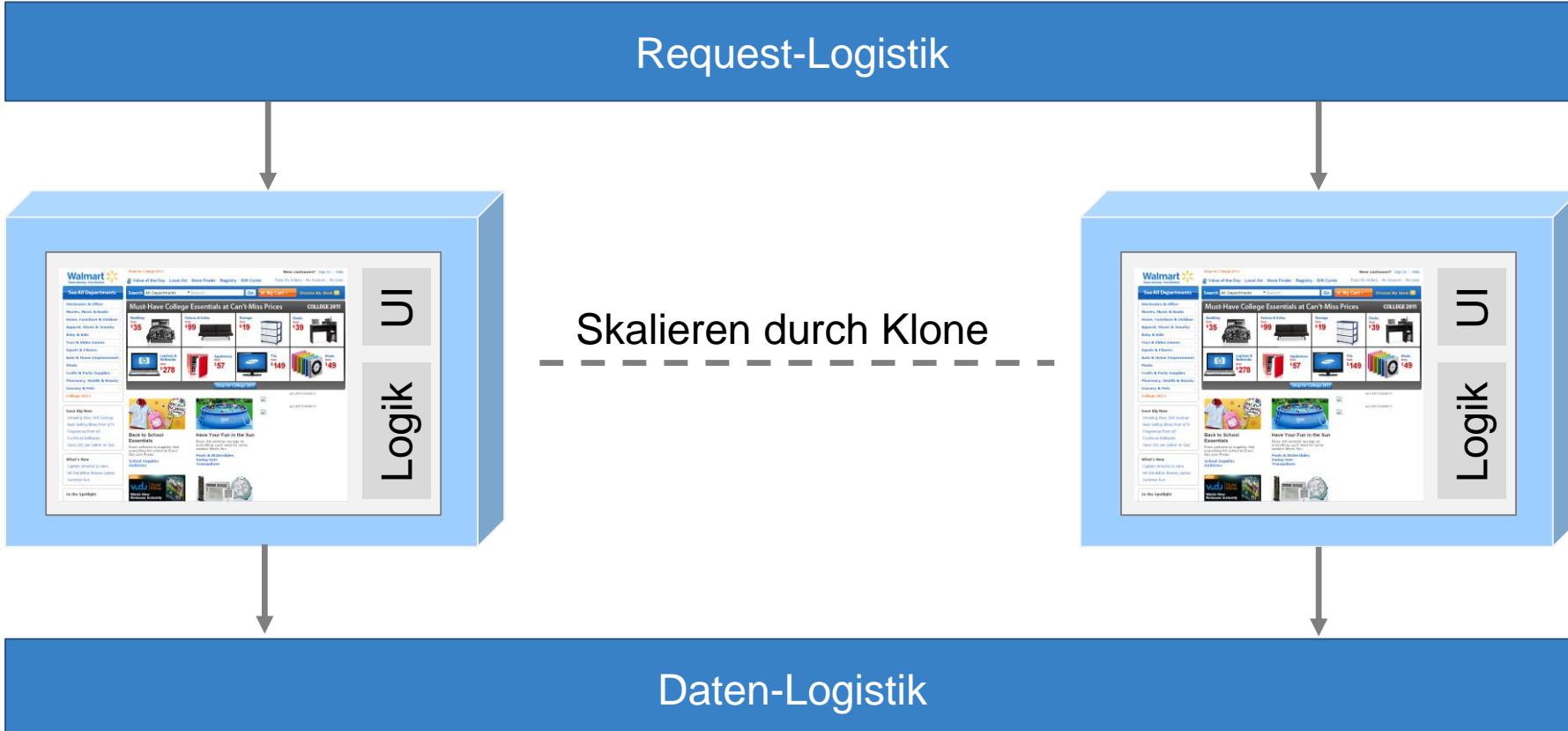
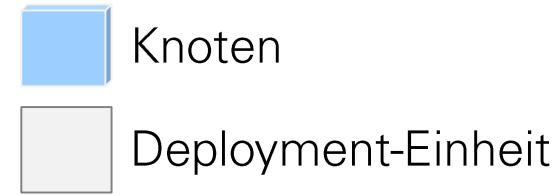
<http://techcrunch.com/2014/12/02/walmart-com-reports-biggest-cyber-monday-in-history-mobile-traffic-at-70-over-the-holidays>



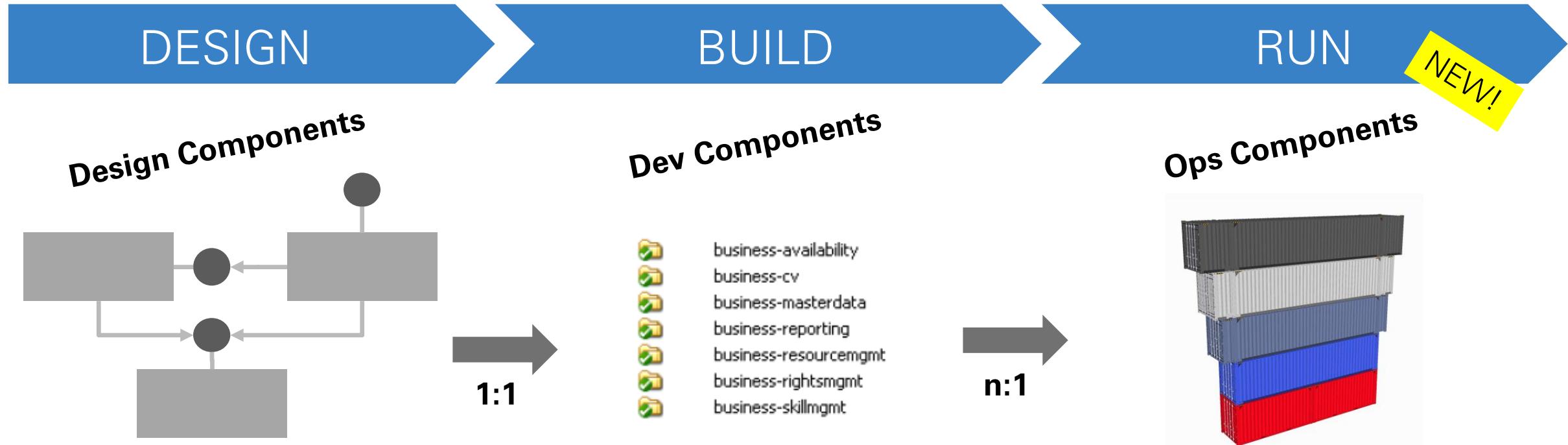
WHAT DID THEY DO?



Betriebsmonolithen



Cloud Native Application Development: Components All Along the Software Lifecycle.

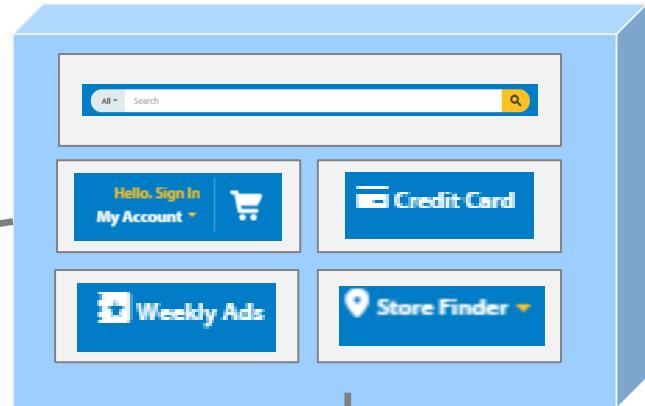
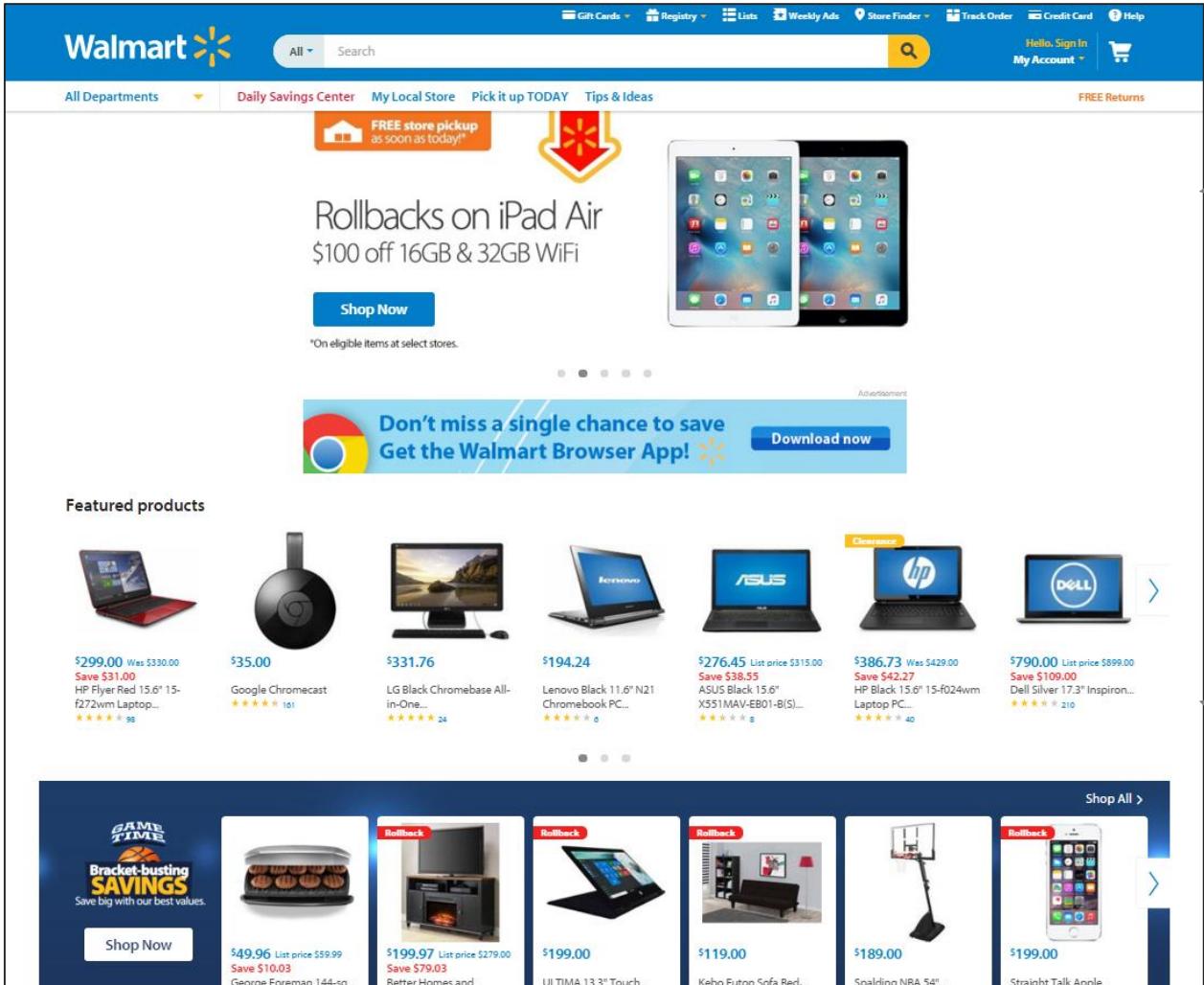


- Complexity unit
- Data integrity unit
- Coherent and cohesive features unit
- Decoupled unit

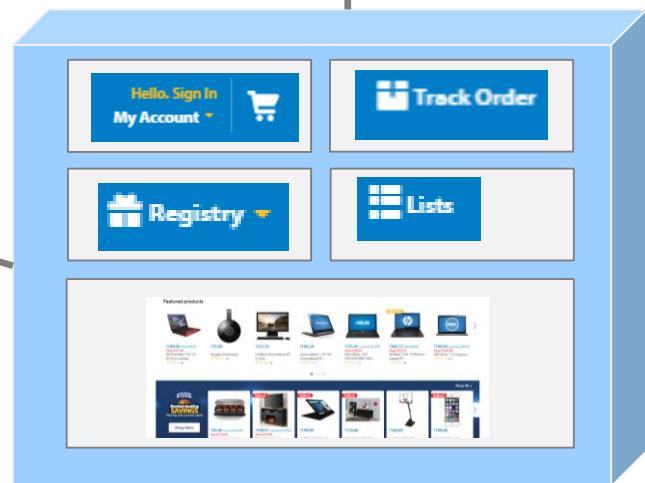
- Planning unit
- Team assignment unit
- Knowledge unit
- Development unit
- Integration unit

- Release unit
- Deployment unit
- Runtime unit (crash, slow-down, access)
- Scaling unit

Betriebskomponenten



Skalieren durch
Verteilung und Klone



Regel 1 für den Betrieb in der Cloud.

“Everything fails all the time.”

— Werner Vogels, CTO of Amazon



Regel 2 für den Betrieb in der Cloud.

Soll nur der Himmel die Grenze sein, dann funktioniert nur horizontale Skalierung.



Regel 3 für den Betrieb in der Cloud.

Wer in die Cloud will, der sollte Cloud sprechen.

TCP

HTTP

DHCP
DNS

Cloud-Architektur aus Sicht der Softwarearchitektur: Design for Failure.

1. Jede Komponente läuft eigenständig und isoliert → *Betriebskomponenten*
2. Die Betriebskomponenten kommunizieren untereinander über Internet-Protokolle → *HTTP, UDP, ...*
3. Jede Betriebskomponente kann in mehreren Instanzen laufen und bietet damit Redundanz. Es gibt keinen „Common Point of Failure“ → *Cluster Orchestrator*
4. Jede Betriebskomponente besitzt Diagnoseschnittstellen um ein fehlerhaftes Verhalten erkennen zu können
5. Jeder Micro Service kann zu jeder Zeit neu gestartet und auf einem anderen Knoten in Betrieb genommen werden. Er besitzt keinen eigenen Zustand.
6. Die Implementierung hinter einem jeden Micro Service kann ausgetauscht werden, ohne dass die Nutzer davon etwas bemerken.

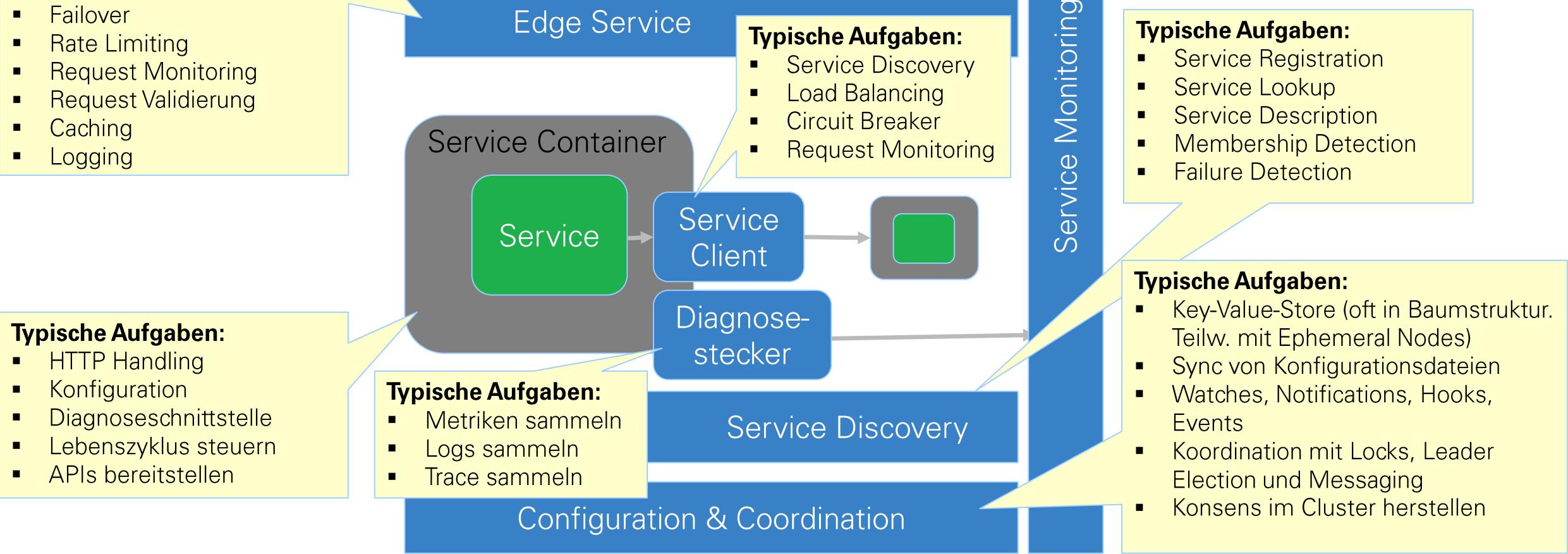
Betriebskomponenten benötigen eine Infrastruktur um sie herum: Eine Micro-Service-Plattform.

Typische Aufgaben:

- Authentifizierung
- Load Shedding
- Load Balancing
- Failover
- Rate Limiting
- Request Monitoring
- Request Validierung
- Caching
- Logging

Typische Aufgaben:

- HTTP Handling
- Konfiguration
- Diagnoseschnittstelle
- Lebenszyklus steuern
- APIs bereitstellen



Typische Aufgaben:

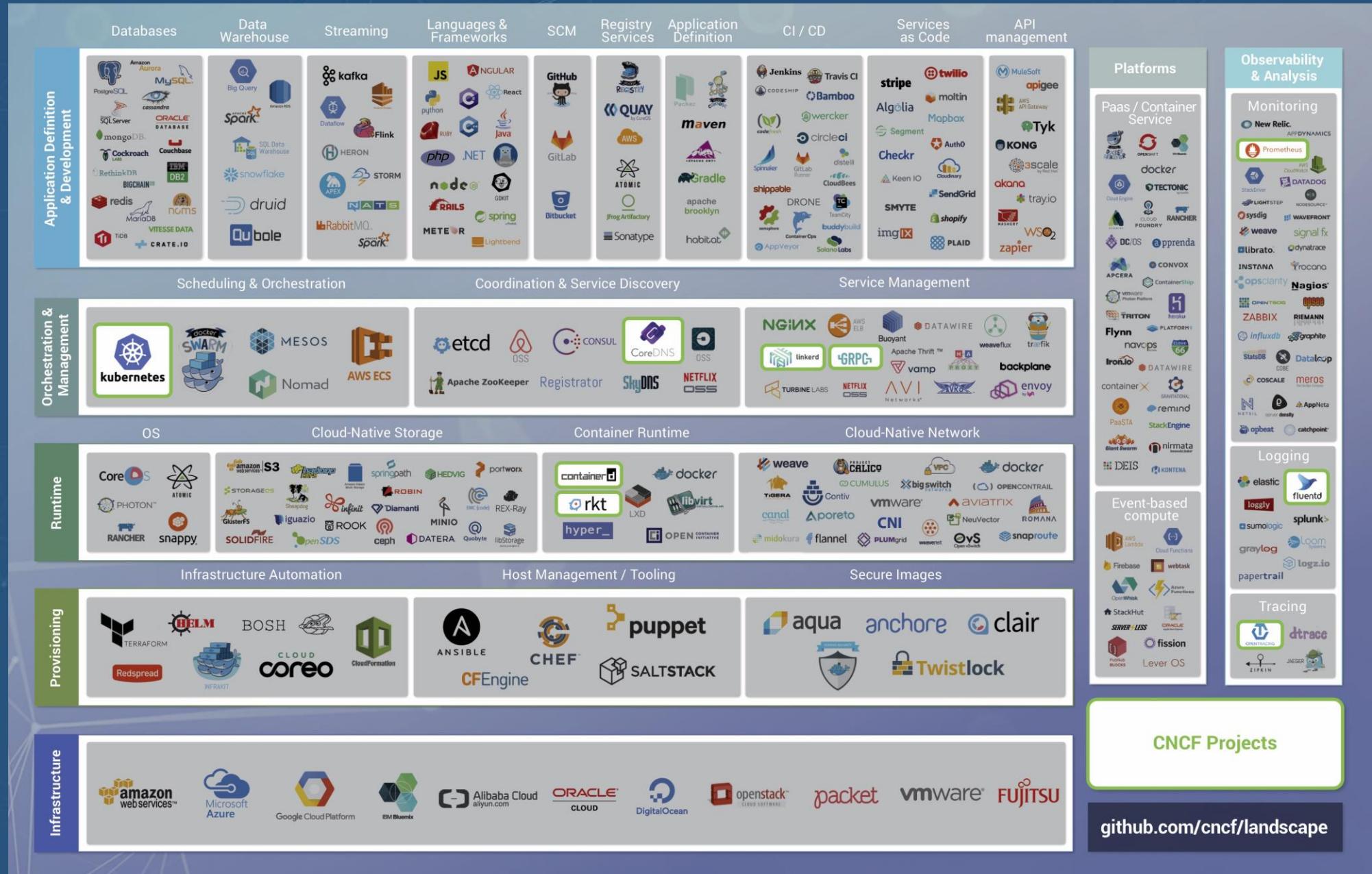
- Aggregation von Metriken
- Sammlung von Logs
- Sammlung von Traces
- Analyse / Visualisierung
- Alerting

Typische Aufgaben:

- Service Registration
- Service Lookup
- Service Description
- Membership Detection
- Failure Detection

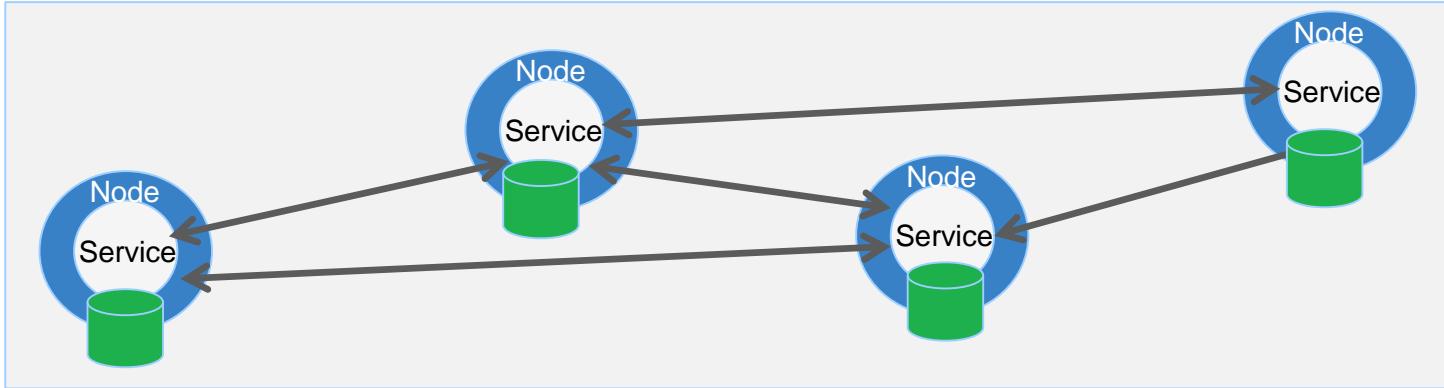
Typische Aufgaben:

- Key-Value-Store (oft in Baumstruktur, Teilw. mit Ephemeral Nodes)
- Sync von Konfigurationsdateien
- Watches, Notifications, Hooks, Events
- Koordination mit Locks, Leader Election und Messaging
- Konsens im Cluster herstellen



Configuration & Coordination: Verteilter Zustand & Konsens

Ein verteilter Konfigurationsspeicher.



Wie wird der Zustand des Konfigurationsspeichers im Cluster synchronisiert?

Gossip Protokolle: Inspiriert von der Verbreitung von Tratsch in sozialen Netzwerken.

Grundlage: Ein Netzwerk an Agenten mit eigenem Zustand

Agenten verteilen einen Gossip-Strom

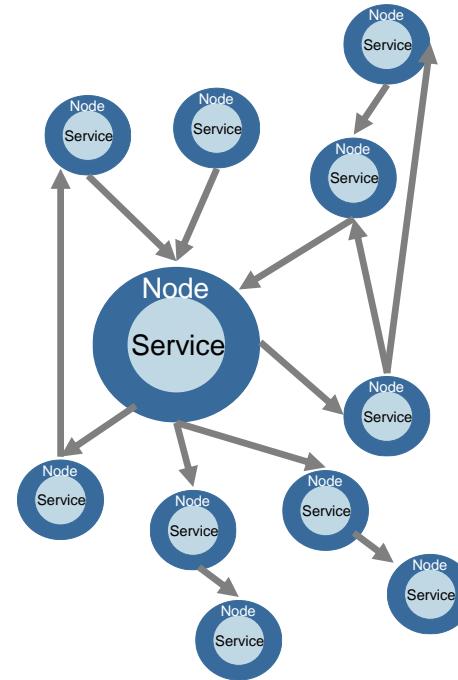
- Nachricht: Quelle, Inhalt / Zustand, Zeitstempel
- Nachrichten werden in einem festen Takt periodisch versendet an eine bestimmte Anzahl anderer Knoten (Fanout)

Virale Verbreitung des Gossip-Stroms

- Knoten, die mit mir in einer Gruppe sind, bekommen auf jeden Fall eine Nachricht
- Die Top x% an Knoten, die mir Nachrichten schicken bekommen eine Nachricht

Nachrichten, denen vertraut wird, werden in den lokalen Zustand übernommen

- Die gleiche Nachricht wurde von mehreren Seiten gehört
- Die Nachricht stammt von Knoten, denen der Agent vertraut
- Es ist keine aktuellere Nachricht vorhanden



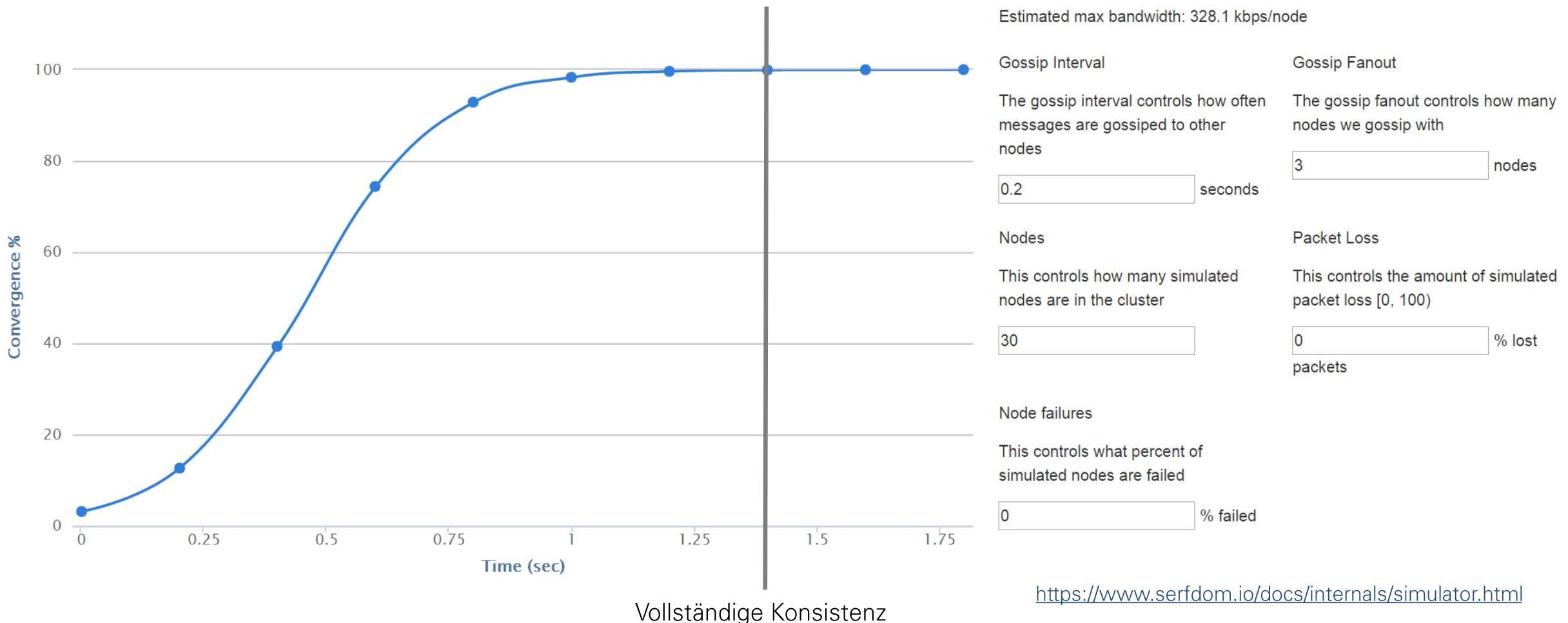
Vorteile:

- Keine zentralen Einheiten notwendig.
- Fehlerhafte Partitionen im Netzwerk werden umschifft. Die Kommunikation muss nicht verlässlich sein.

Nachteile:

- Der Zustand ist potenziell inkonsistent verteilt (konvergiert aber mit der Zeit)
- Overhead durch redundante Nachrichten.

Die Konvergenz der Daten und damit der Zeitpunkt der vollständigen Konsistenz ist berechenbar.



Die Lamport-Uhr als Basis für logische Zeitstempel um zeitliche Ordnung von Nachrichten zu erkennen.

- Gegeben sei die Menge an Nachrichten N
- Dann sei die Funktion ts definiert für einen logischen Zeitstempel auf Basis einer logischen Uhr:
 $ts(n \in N) \Rightarrow$ numerischer Wert
- Mit der Konsistenzbedingung:
 $\square n_1, n_2 \in N : ts(n_1) < ts(n_2) \Rightarrow n_1 \rightarrow n_2$
- Wobei \rightarrow als die „hat sich vorher ereignet“ Relation definiert ist.

Beim Senden:

$Uhr = Uhr + 1;$

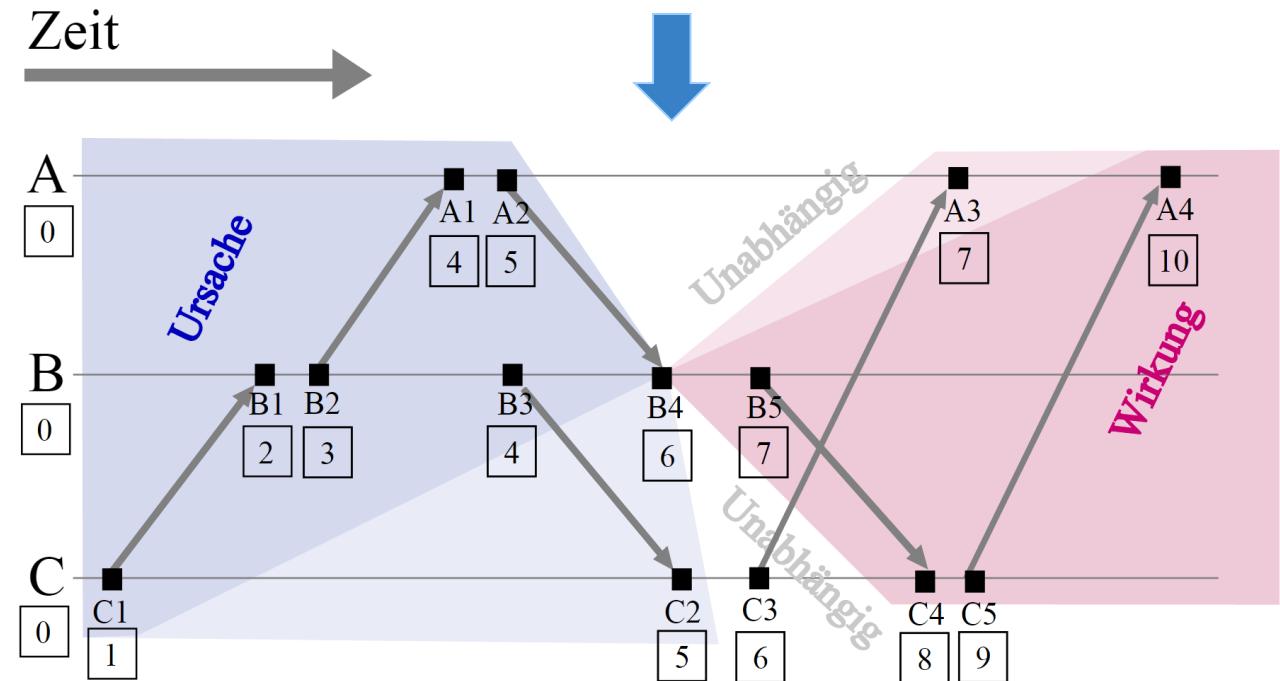
$Zeitstempel = Uhr;$

$sende(Nachricht, Zeitstempel);$

Beim Empfangen:

$(Nachricht, Zeitstempel) = empfange();$

$Uhr = max(Zeitstempel, Uhr) + 1;$



Protokolle für verteilten Konsens sind im Gegensatz zu Gossip-Protokollen konsistent aber nicht hoch-verfügbar.

Grundlage: Netzwerk an Agenten

Prinzip: Es reicht, wenn der Zustand auf einer einfachen Mehrheit der Knoten konsistent ist und die restlichen Knoten ihre Inkonsistenz erkennen.

Verfahren:

- Das Netzwerk einigt sich per einfacher Mehrheit auf einen Leader-Agenten – initial und falls der Leader-Agent nicht erreichbar ist. Eine Partition in der Minderheit kann keinen Leader-Agenten wählen.
- Alle Änderungen laufen über den Leader-Agenten. Dieser verteilt per Multicast Änderungsnachrichten periodisch im festen Takt an alle weiteren Agenten.
- Quittiert die einfache Mehrheit an Agenten die Änderungsnachricht, so wird die Änderung im Leader und (per Nachricht) auch in den Agenten aktiv, die quittiert haben. Ansonsten wird der Zustand als inkonsistent angenommen.

Konkrete Konsens-Protokolle: Raft, Paxos

Vorteile:

- Fehlerhafte Partitionen im Netzwerk werden toleriert und nach Behebung des Fehlers wieder automatisch konsistent.
- Streng konsistente Daten.

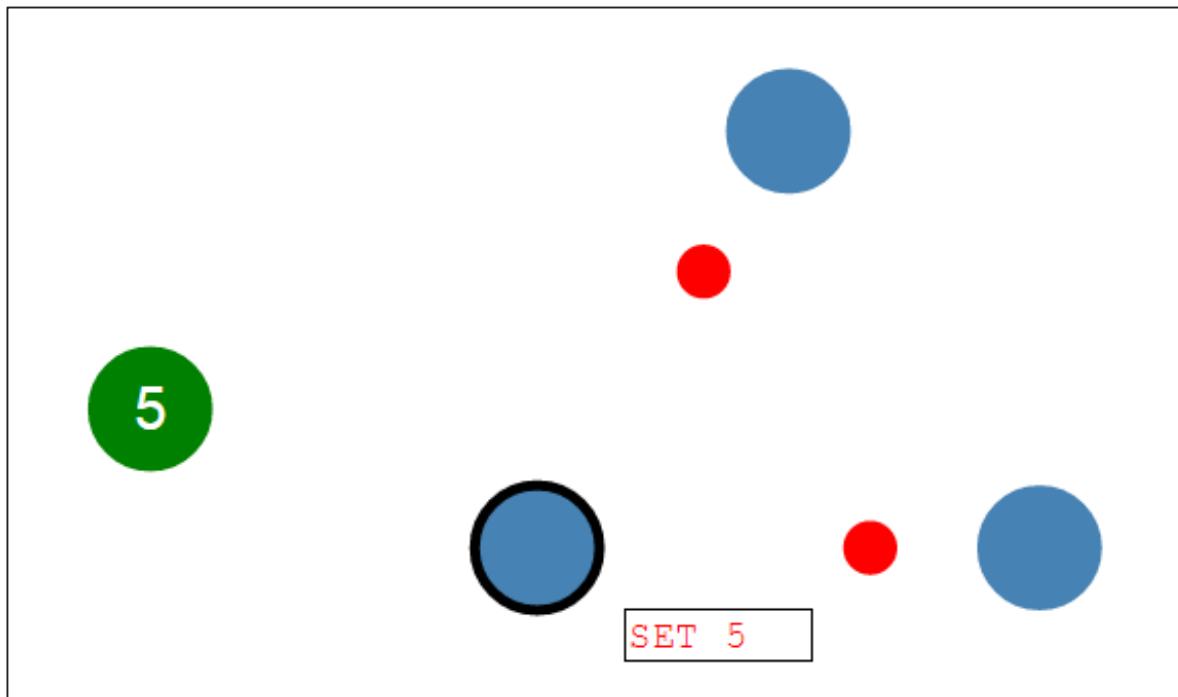
Nachteile:

- Der zentrale Leader-Agent limitiert den Durchsatz an Änderungen.
- Nicht hoch-verfügbar: Bei einer Netzwerk-Partition kann die kleinere Partition nicht weiterarbeiten. Ist die Mehrheit in keiner Partition, so kann insgesamt nicht weiter gearbeitet werden.

Das Raft Konsens-Protokoll

Ongaro, Diego; Ousterhout, John (2013).

"In Search of an Understandable Consensus Algorithm".



<http://thesecretlivesofdata.com/raft>

Paxos als Alternative zu Raft

- Erstmals 1989 beschrieben von Leslie Lamport zur Replikation von Zustandsautomaten in verteilten Systemen auf Basis (s)einer logischen Uhr.
- Unterschied zu Raft: Komplexer.
Aber funktional gleichwertig zu Raft.

Quellen:

- <http://the-paper-trail.org/blog/consensus-protocols-paxos/>
- <https://www.youtube.com/watch?v=JEpsBg0AO6o>
- http://www.uio.no/studier/emner/matnat/ifi/INF5040/h08/group/The_Paxos_Algorithm.pdf
- <http://research.microsoft.com/en-us/um/people/lamport/pubs/paxos-simple.pdf>

Implementing Replicated Logs with Paxos

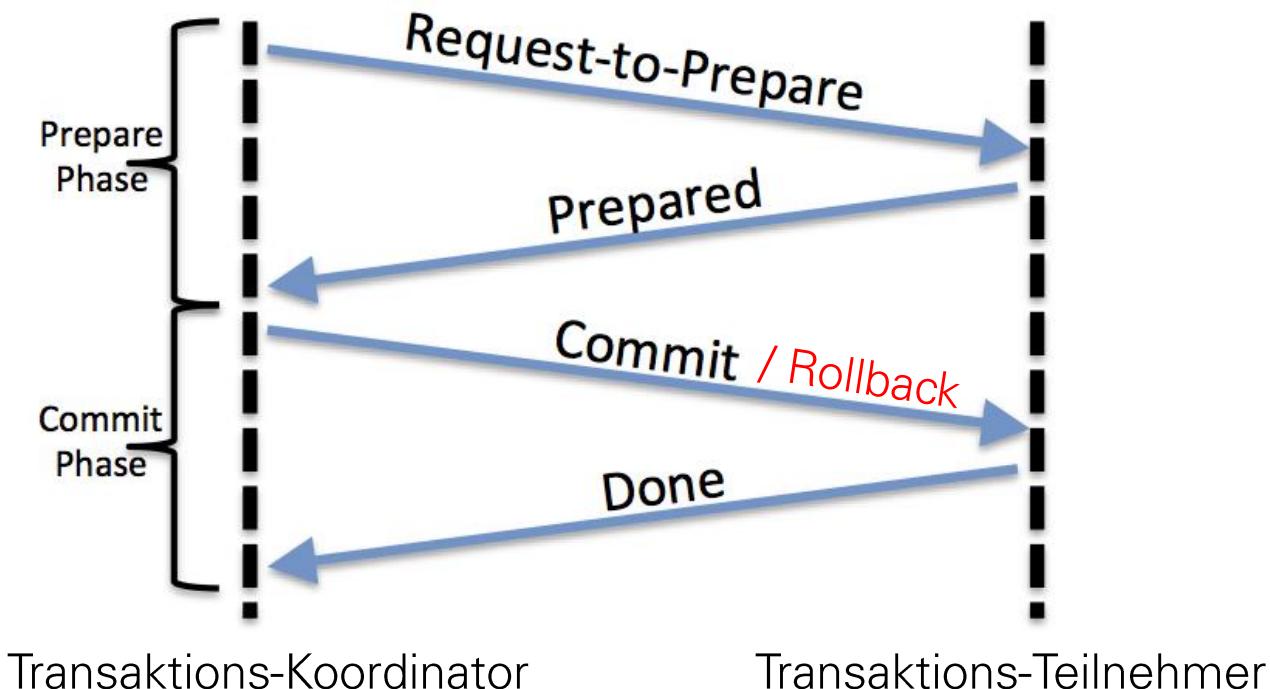
John Ousterhout and Diego Ongaro
Stanford University



Note: this material borrows heavily from slides by Lorenzo Alvisi, Ali Ghodsi, and David Mazières

Ist strenge Konsistenz über alle Knoten notwendig, so verbleibt das 2-Phase-Commit Protokoll (2PC)

Ein Transaktionskoordinator verteilt die Änderungen und aktiviert diese erst bei Zustimmung aller. Ansonsten werden die Änderungen rückgängig gemacht.



Vorteil:

- Alle Knoten sind konsistent zueinander.

Nachteile:

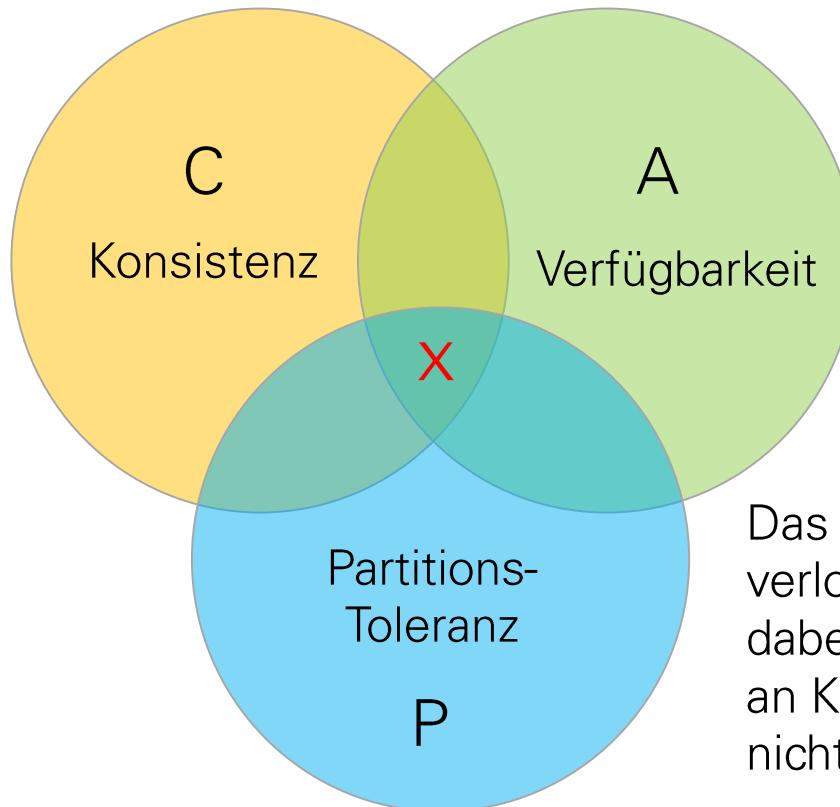
- Zeitintensiv, da stets alle Knoten zustimmen müssen.
- Das System funktioniert nicht mehr, sobald das Netzwerk partitioniert ist.

Das CAP Theorem.

Theorem von Brewer für Eigenschaften von zustandsbehafteten verteilten Systemen – mittlerweile auch formal bewiesen.

Brewer, Eric A. "Towards robust distributed systems." *PODC*. 2000.

Es gibt drei wesentliche Eigenschaften, von denen ein verteiltes System nur zwei gleichzeitig haben kann:

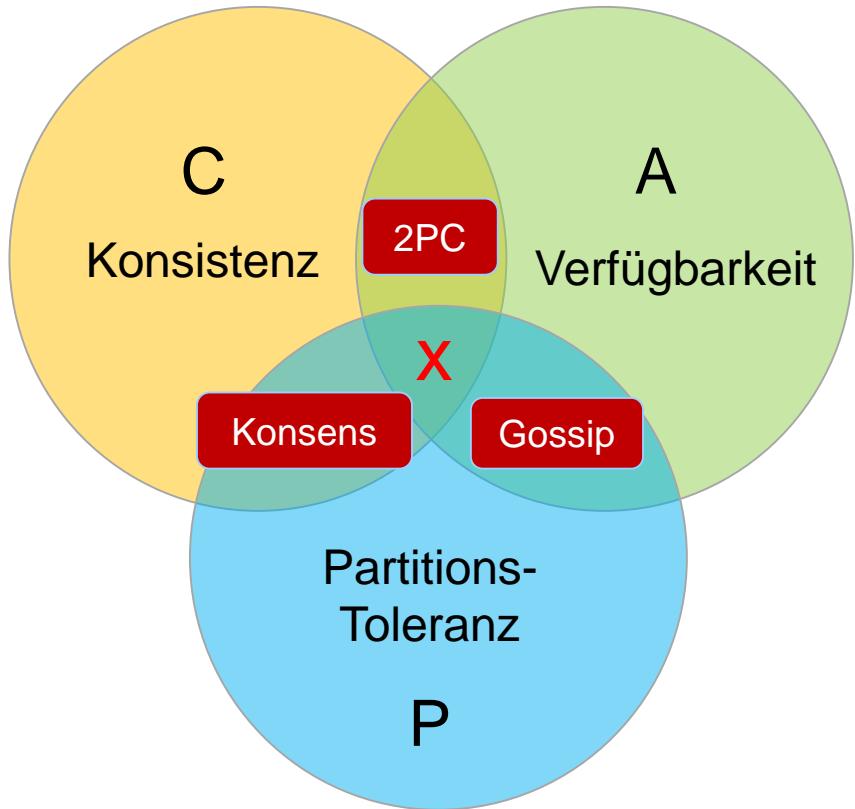


Alle Knoten sehen die selben Daten zur selben Zeit. Alle Kopien sind stets gleich.

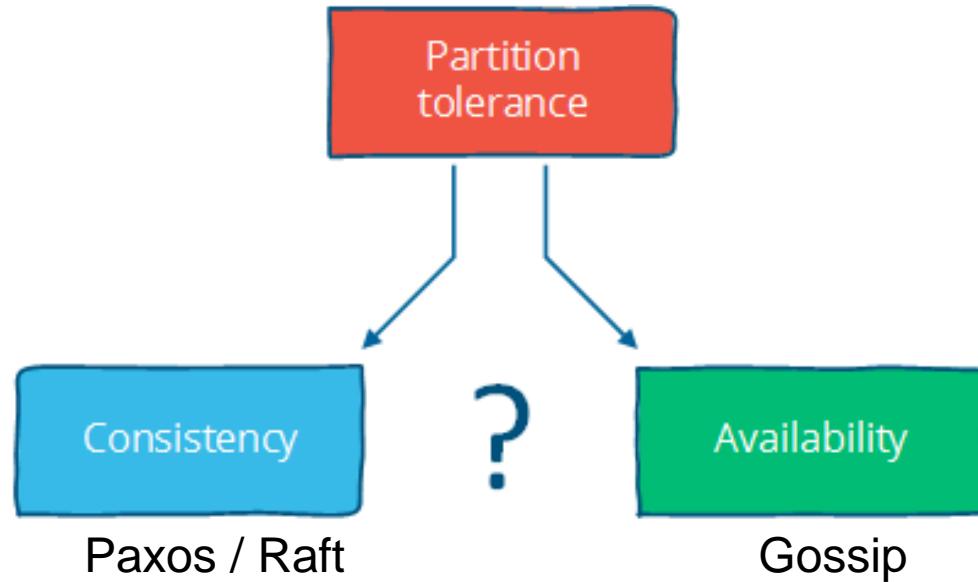
Das System läuft auch, wenn einzelne Knoten ausfallen. Ausfälle von Knoten und Kanälen halten die überlebenden Knoten nicht von ihrer Funktion ab.

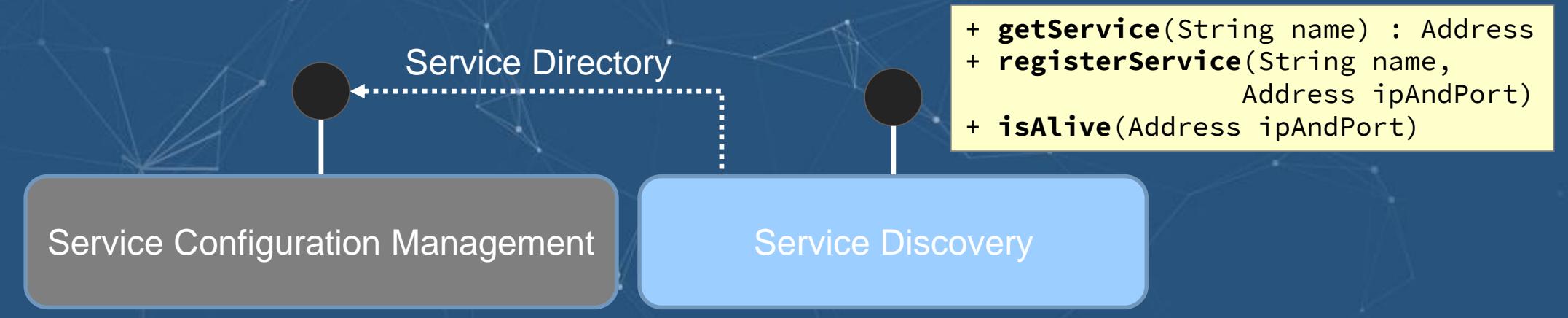
Das System funktioniert auch im Fall von verlorenen Nachrichten. Das System kann dabei damit umgehen, dass sich das Netzwerk an Knoten in mehrere Partitionen aufteilt, die nicht miteinander kommunizieren.

Die vorgestellten Protokolle und das CAP Theorem.



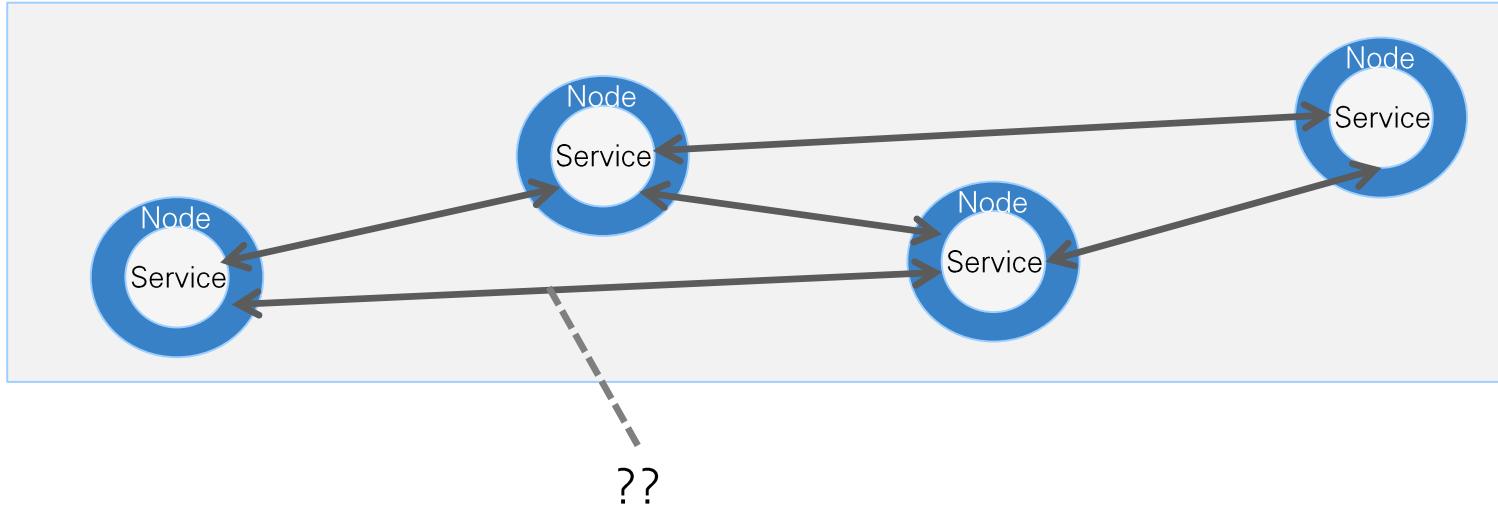
In der Cloud müssen Partitionen angenommen werden.
Damit ist die Entscheidung binär zwischen Konsistenz und
Verfügbarkeit.





Service Discovery

Die Probleme einer klassischen Verknüpfung von Services in der Cloud.



Probleme:

- Mangelnde Redundanz: Jeder Service wird direkt genutzt. Er kann nicht unmittelbar in mehreren Instanzen laufen, die Redundanz schaffen.
- Mangelnde Flexibilität: Die Services können nicht ohne Seiteneffekt neu gestartet oder auf einem anderen Knoten in Betrieb genommen werden – oder sogar durch eine andere Service-Implementierung ausgetauscht werden.

Lösungen:

- Dynamischer DNS
- Ambassador
- Dynamischer Konfigurationsdateien und Umgebungsvariablen

Das Ambassador Pattern

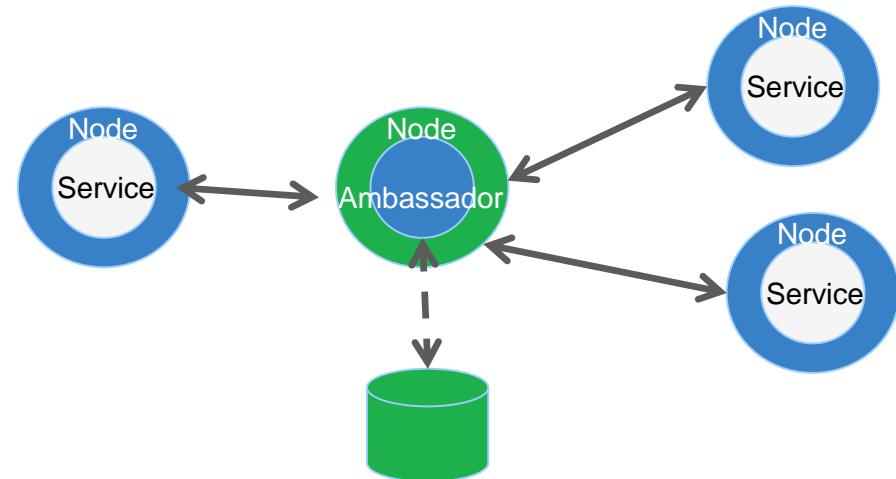
Ein Ambassador-Knoten für jede Knoten-Art (z.B. Webserver)

- **Service Registration:**
 - Beobachtet das Cluster und erkennt neue und kranke/tote Knoten in seiner Gruppe.
 - Hinterlegt die aktuell aktiven Knoten im Konfigurationsspeicher.
- **Service Discovery:** Der Client kommuniziert mit dem Ambassador-Knoten, der die Anfragen aber möglichst effizient an einen Knoten der Gruppe weiterreicht.

Der Ambassador-Knoten kann dabei eine Reihe an Zusatzdienste erweisen bei der Verbindung zum Service

(Service Binding):

- Load Balancing inklusive Failover
- Service Monitoring
- Circuit Breaker Pattern
- Throttling



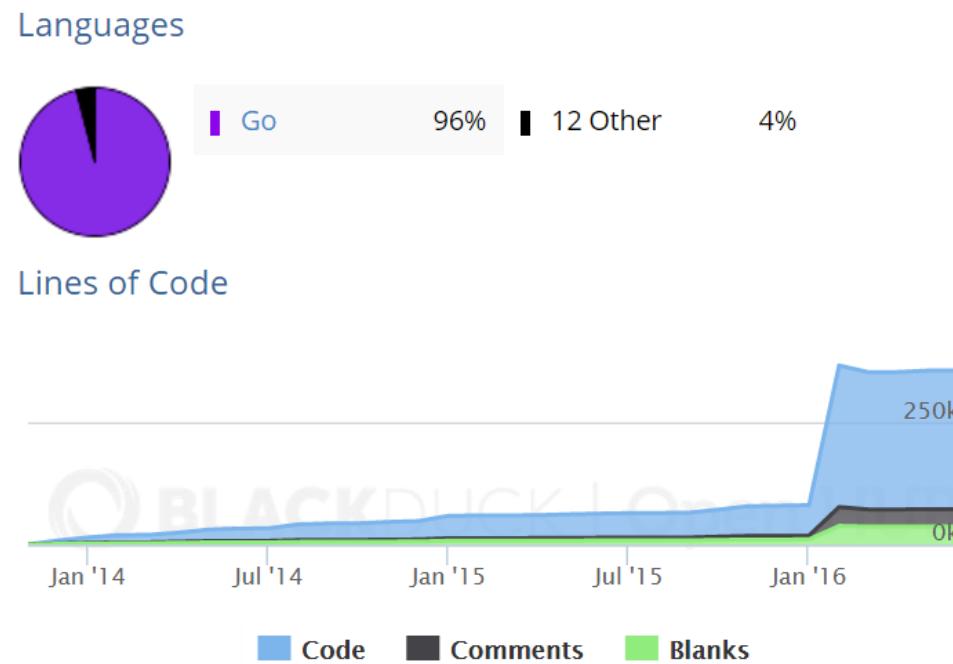
Beispiel: Consul

<https://www.consul.io>

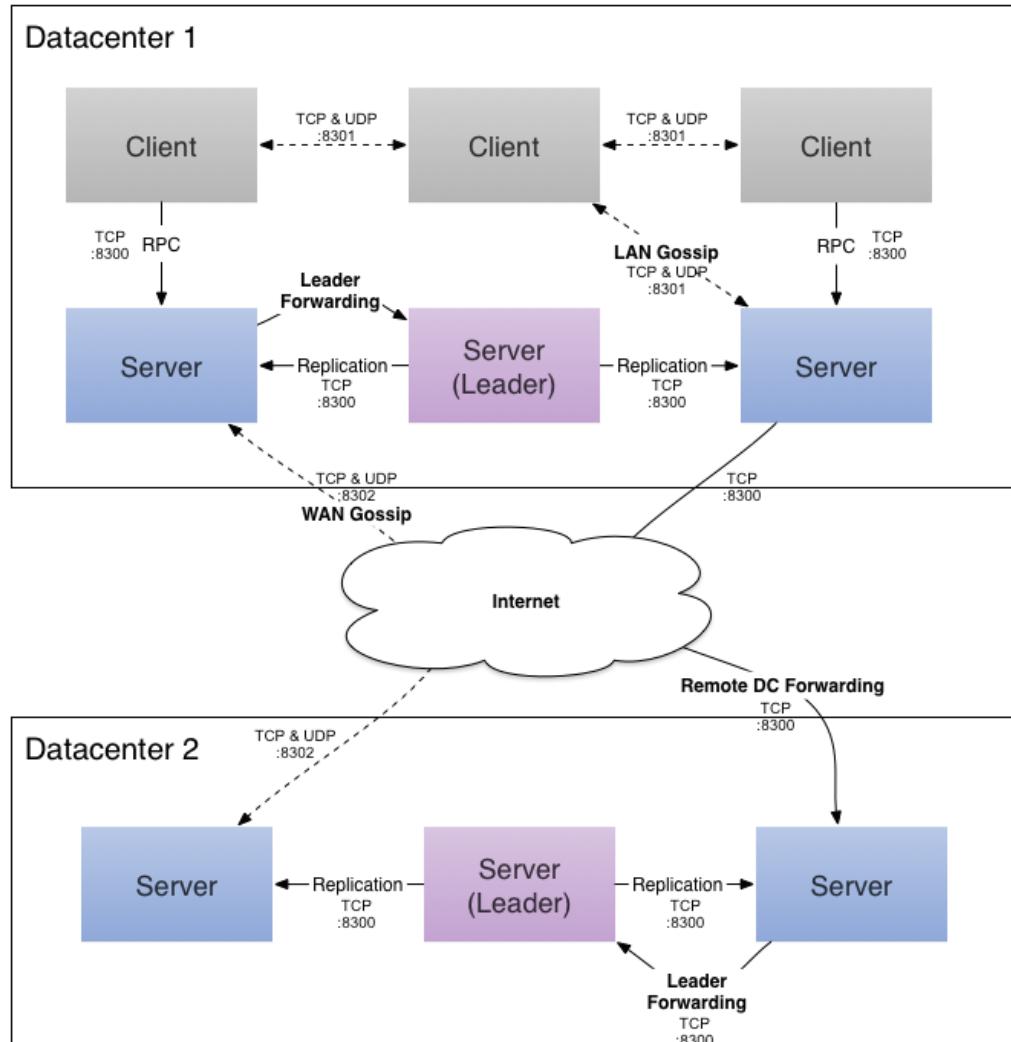


Consul

- Infrastruktur für Configuration & Coordination (per Raft-Protokoll) und Service Discovery (per DNS oder HTTP-Lookups) inkl. Health Checking der Services.
- Open Source Projekt der Firma HashiCorp (<https://www.consul.io>), von dem auch eine kommerzielle Variante verfügbar ist.



Consul Architektur.



- Consul besteht aus einer einzigen Binary (*consul*), dem *Consul Agent*.
- Der Consul Agent kann sowohl als *Server* als auch als *Client* gestartet werden.
- Alle Agenten finden sich gegenseitig über ein Gossip-Protokoll und identifizieren so auch Fehler.
- Die Server synchronisieren ihren Zustand über ein Konsens-Protokoll (Raft). Empfohlen werden 3-5 Consul-Server pro Datacenter auf dedizierten Maschinen.
- Die Clients sind i.W. zustandslos und laufen auf allen Maschinen, die mit Consul interagieren. Die Kommunikation mit Consul läuft stets über den lokalen Agenten, der dann weiter vermittelt.
- Consul Server können sich über mehrere Datacenter hinweg synchronisieren.

siehe <https://www.consul.io/docs/internals/architecture.html>

Architektur von Cloud Native Systemen

#GIFEE



Google's
(and Facebook's, Twitter's, Airbnb's, ...)

Infrastructure
For
Everyone
Else

DISRUPT

INDUSTRIALIZE

ZE

HYPERSCALE

(traffic, data, features)

ANTIFRAGILITY

(resiliency & autonomy)

CLOUD NATIVE APPLICATION

PRODUCTIVITY

(continuous delivery & devops)

OPEX SAVINGS

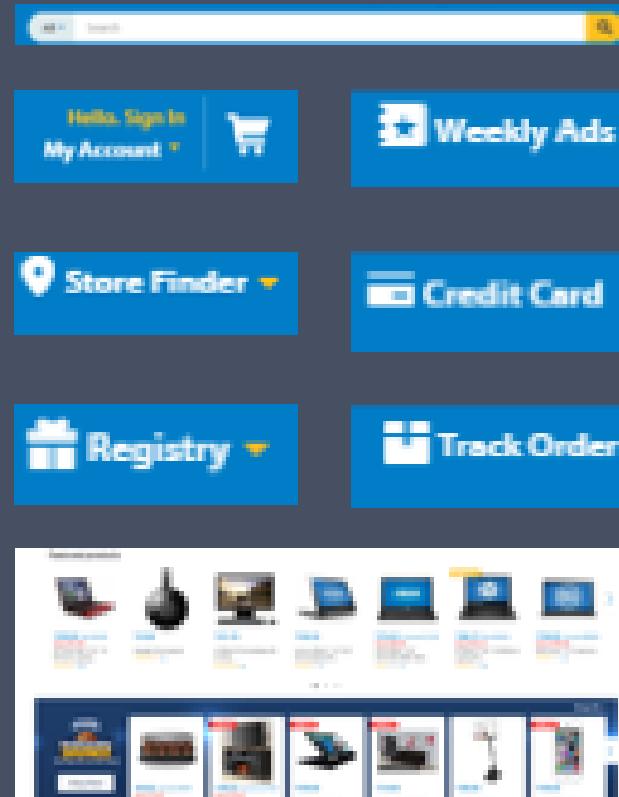
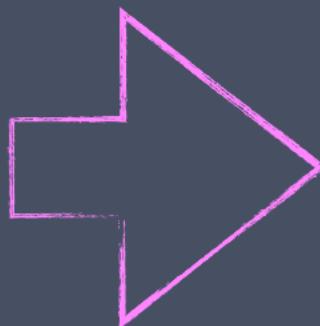
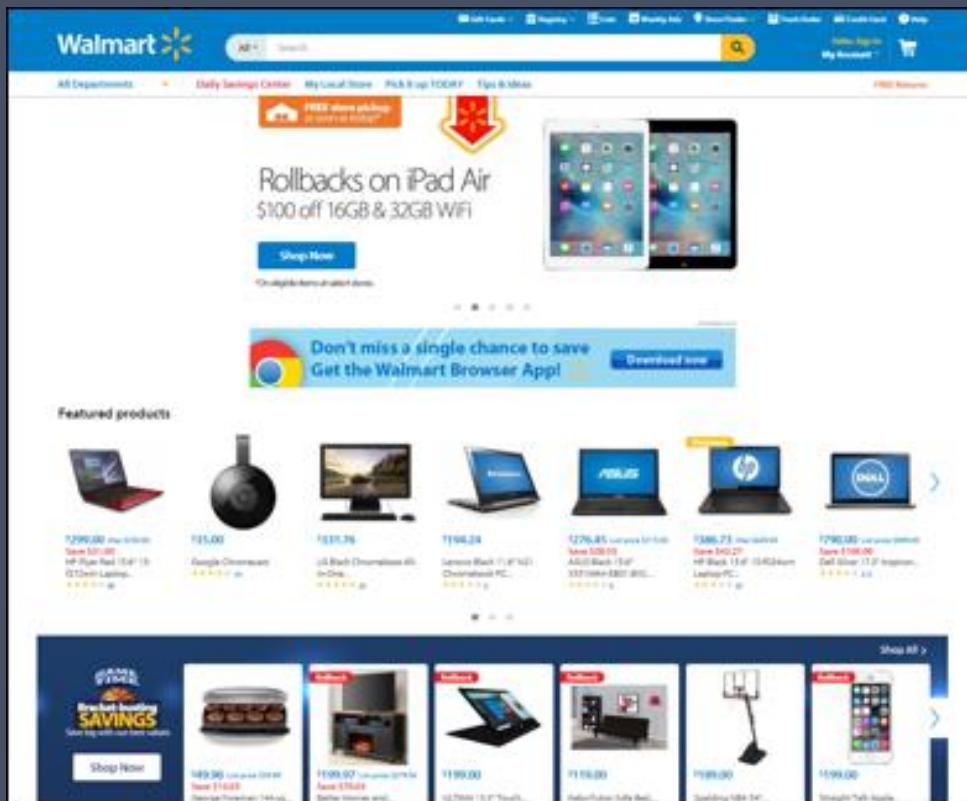
(utilization & automation)

BUILD AND COMPOSED
AS MICROSERVICES

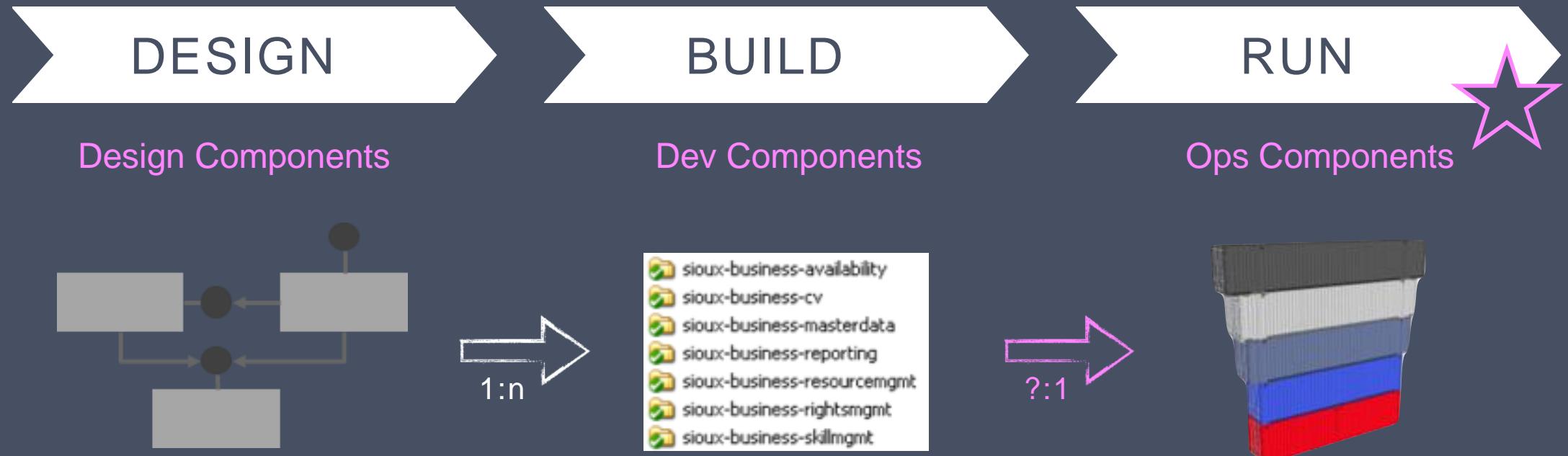
CLOUD NATIVE APPLICATION
PACKAGED AND
DISTRIBUTED AS CONTAINERS
EXECUTED IN THE CLOUD
DYNAMICALLY

BUILD AND COMPOSED
AS MICROSERVICES

Teile und herrsche



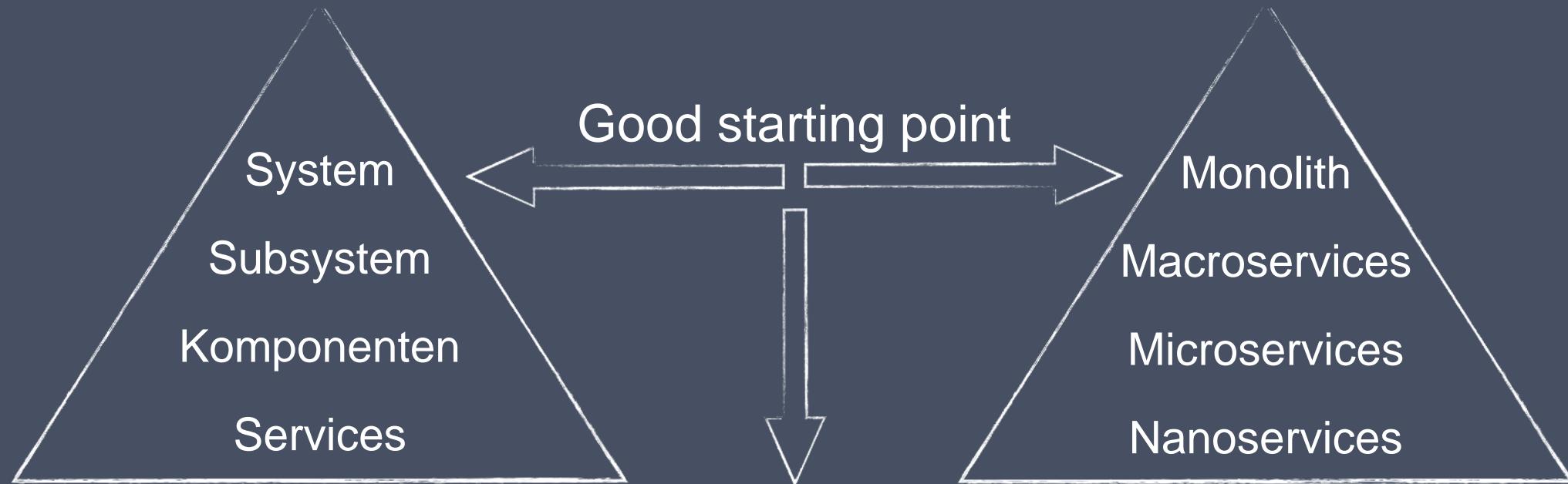
Komponenten entlang des Software-Lebenszyklus



Dev Components



Ops Components



Decomposition Trade-Offs

- | | |
|--|--|
| + Independent releases, deployments, teams | - Distribution debt |
| + Short roundtrips | - Increased infrastructure complexity |
| + Runtime isolation (crash, slow-down) | - Increased troubleshooting complexity |
| + More flexible to scale | - Increased integration complexity |
| + Higher resources utilisation | - Higher resource overhead (potentially) |

PACKAGED AND
DISTRIBUTED AS CONTAINERS

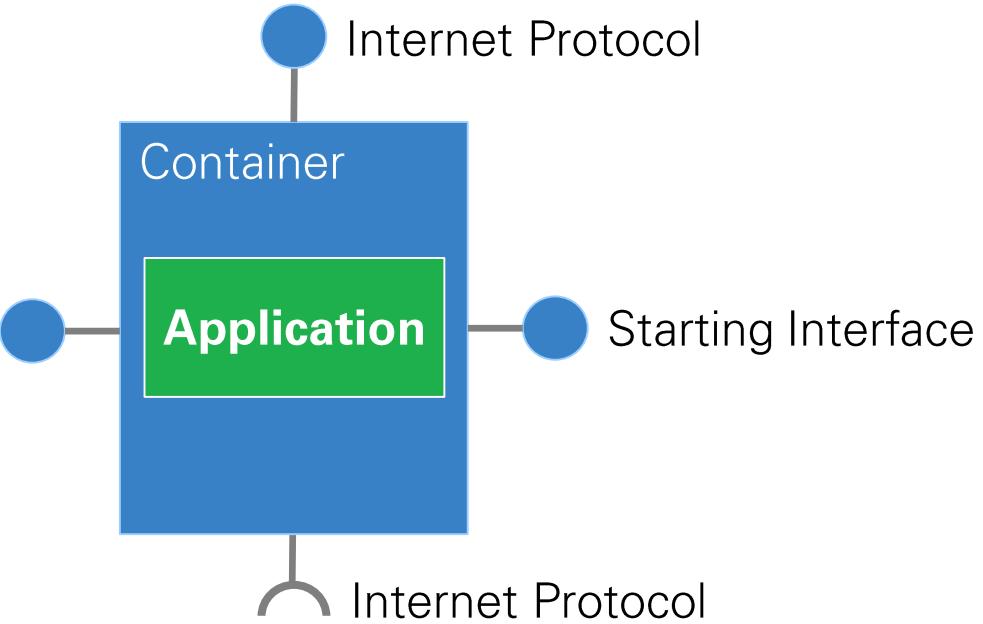


- “Verpackung” für Ops Components
- Standard-Schnittstellen für Standard-Betriebsprozeduren
- Einfach zu transportierende, schnell zu startende und mit wenig Overhead ausführbare Software-Einheiten

Die Anatomie einer Betriebs-Komponente.



Diagnose Interface



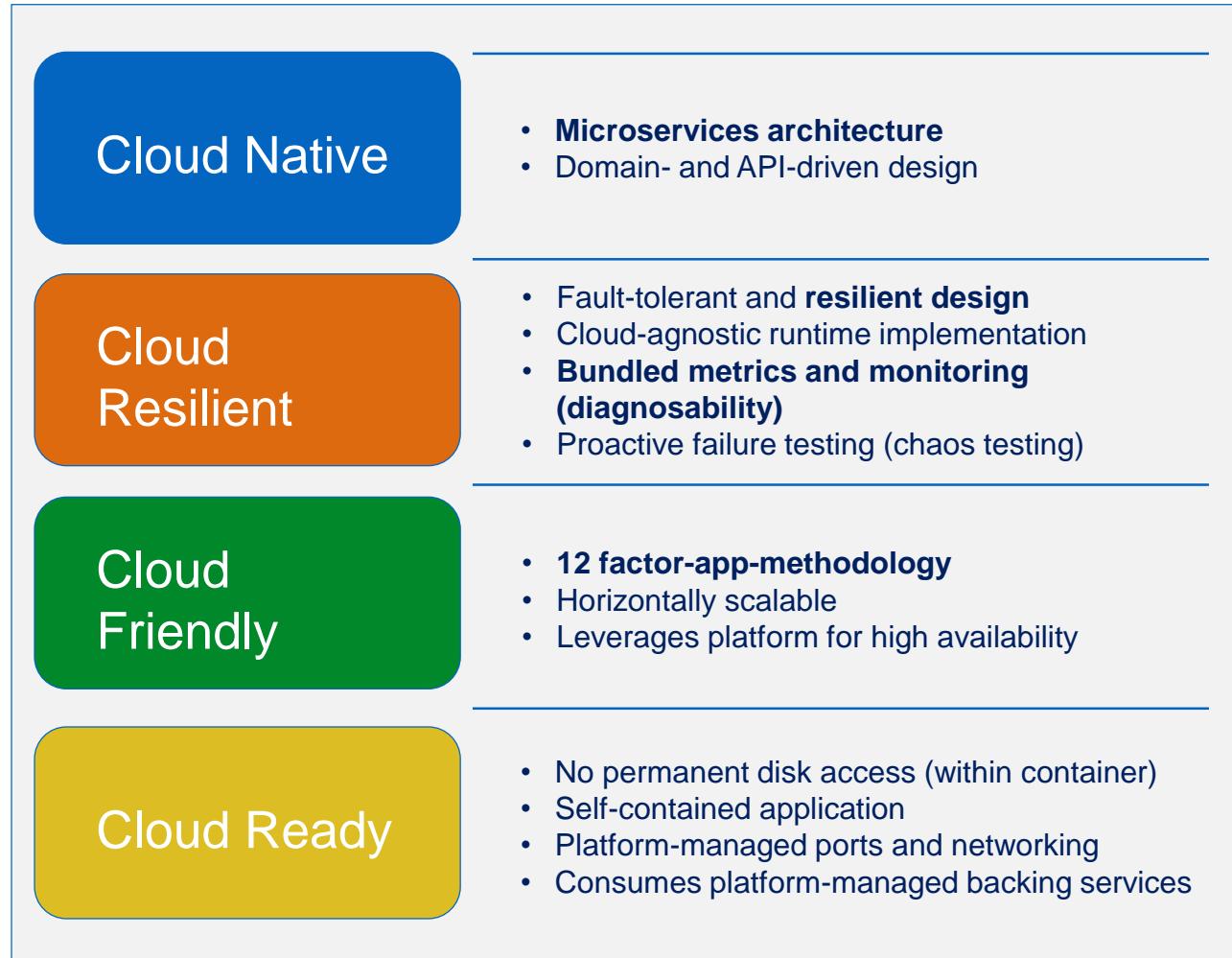
DYNAMICALLY
EXECUTED IN THE CLOUD

ARCHITEKTUR VON CLOUD NATIVE APPLIKATIONEN

Die 5 Gebote der Cloud.

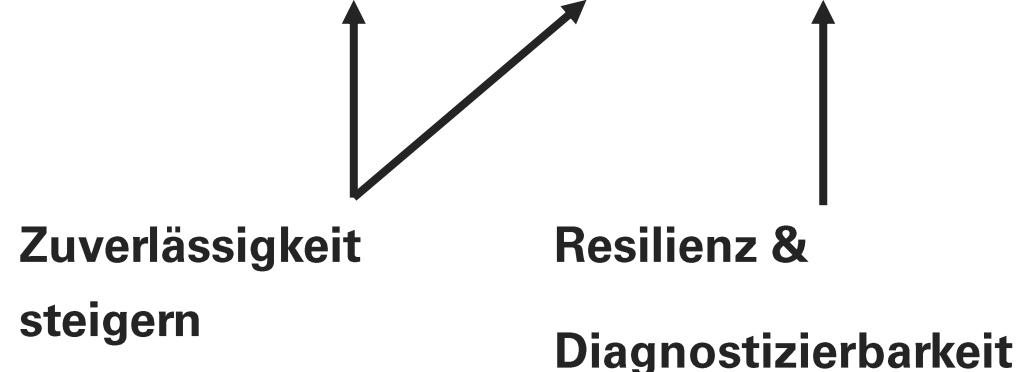
1. Everything Fails All The Time.
2. Focus on MTTR and not on MTTF.
3. Respect the Eight Fallacies of Distributed Computing.
4. Scale out, not up.
5. Treat resources as cattle, not pets.

Das Cloud Native Application Reifegradmodell



Resilienz

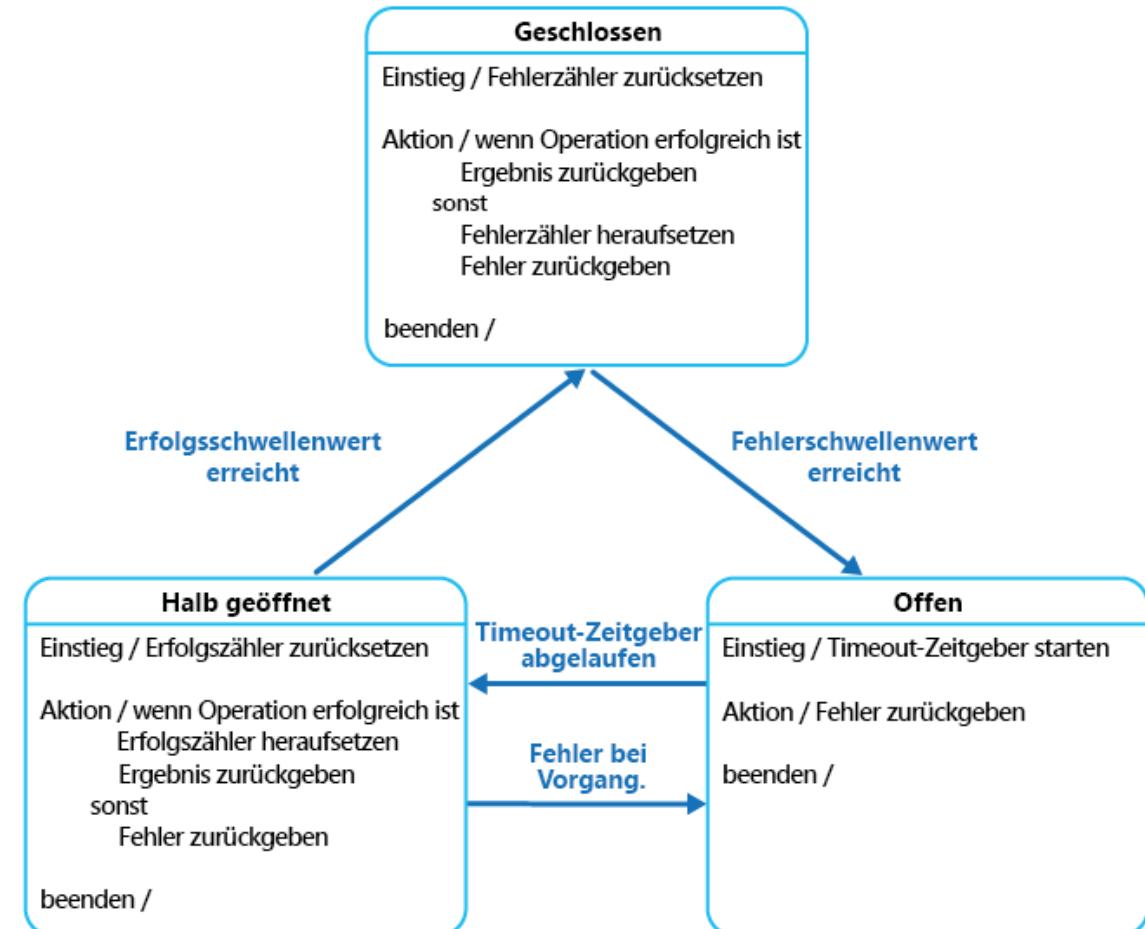
$$\text{Verfügbarkeit} = \text{MTTF} / (\text{MTTF} + \text{MTTR})$$



Resilienz: Die Fähigkeit eines Systems mit unerwarteten und fehlerhaften Situationen umzugehen

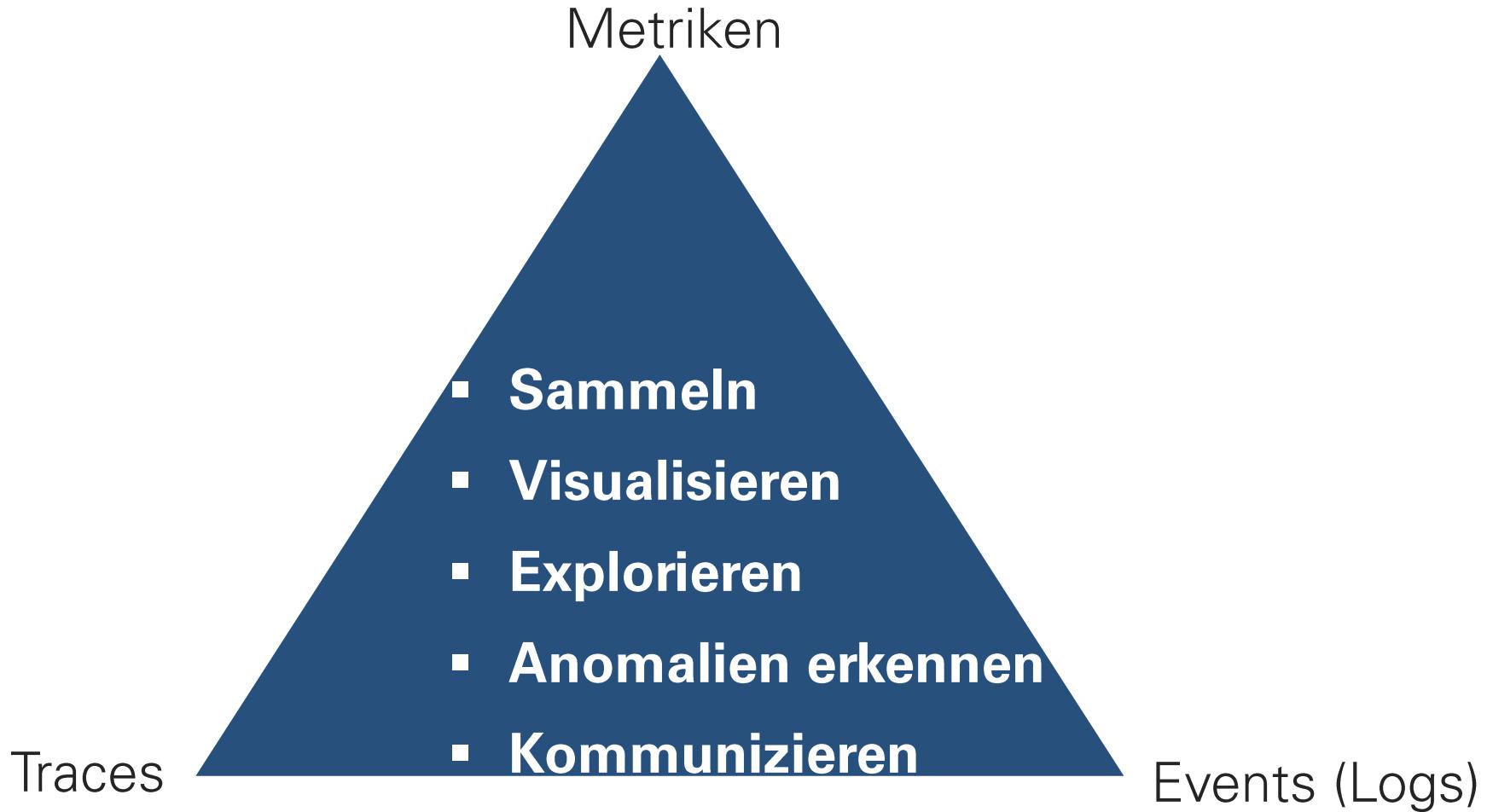
- Ohne dass es der Nutzer merkt (Bestfall)
- Mit einer „graceful degradation“ des Services (schlechtester Fall)

Resilienz-Pattern: Circuit Breaker



Weitere Patterns: <https://docs.microsoft.com/de-de/azure/architecture/patterns/category/resiliency>

Dignostizierbarkeit



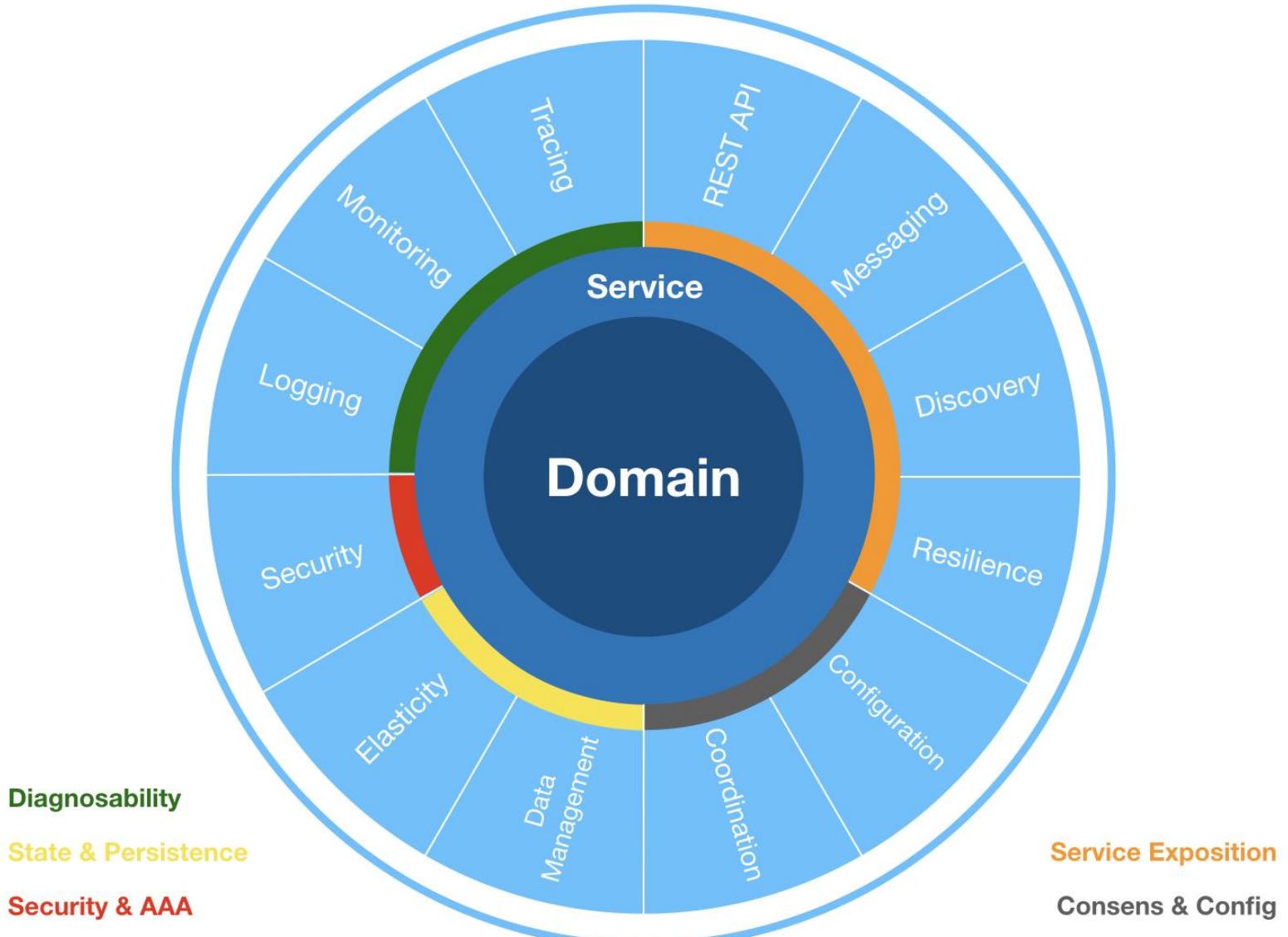
12 Factor App

1	Codebase One codebase tracked in revision control, many deploys.	7	Port binding Export services via port binding.
2	Dependencies Explicitly declare and isolate dependencies.	8	Concurrency Scale out via the process model.
3	Configuration Store config in the environment.	9	Disposability Maximize robustness with fast startup and graceful shutdown.
4	Backing Services Treat backing services as attached resources.	10	Dev/Prod Parity Keep development, staging, and production as similar as possible
5	Build, release, run Strictly separate build and run stages.	11	Logs Treat logs as event streams.
6	Processes Execute the app as one or more stateless processes.	12	Admin processes Run admin/management tasks as one-off processes.

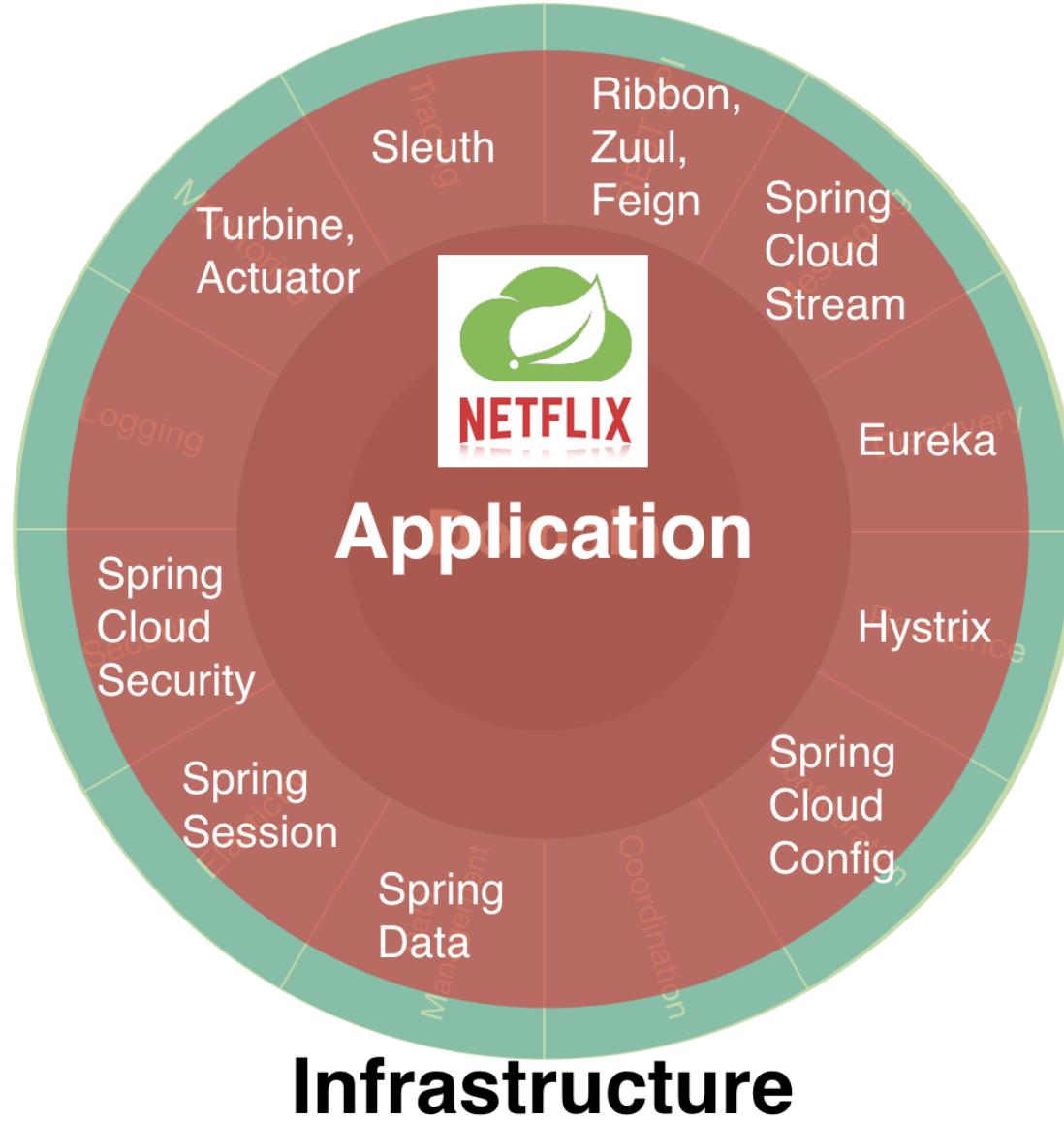
<https://12factor.net/de>

<https://www.slideshare.net/Alicanakku1/12-factor-apps>

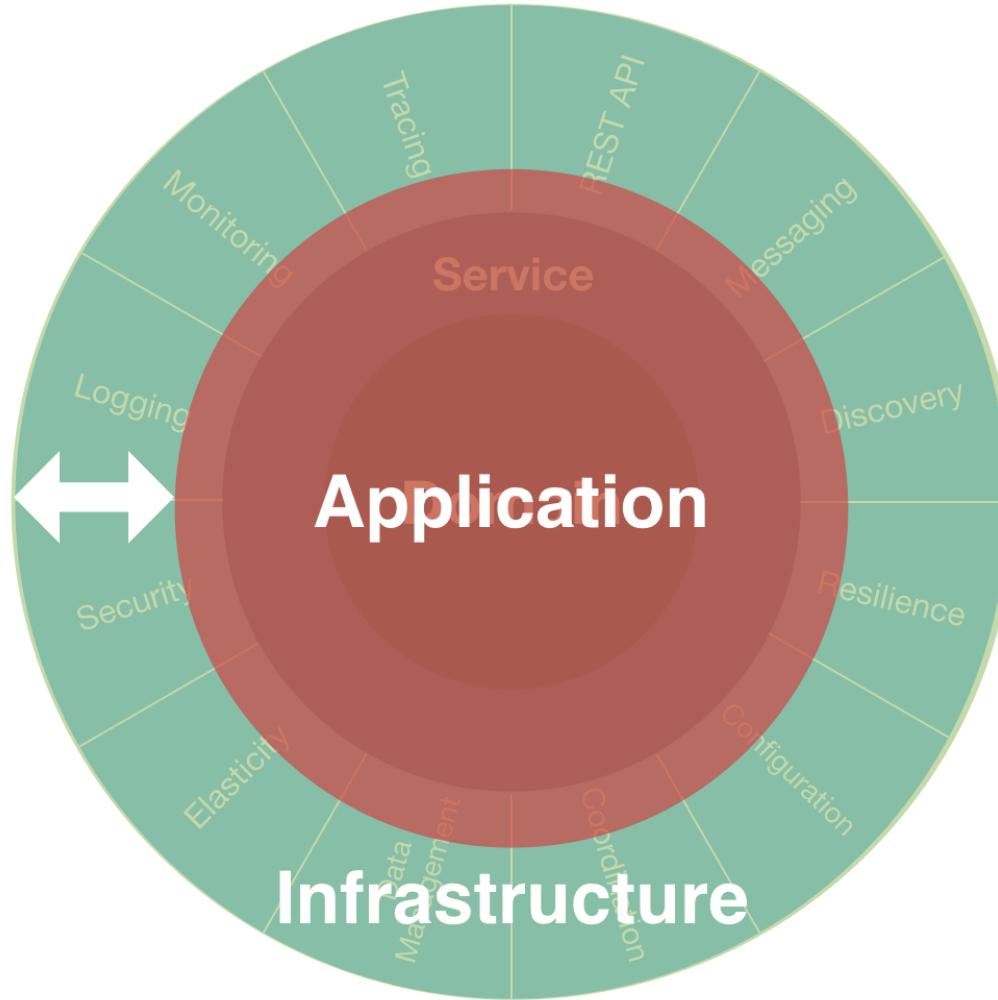
Technische Aspekte von Microservices



Technische Aspekte von Microservices: Library Lösung

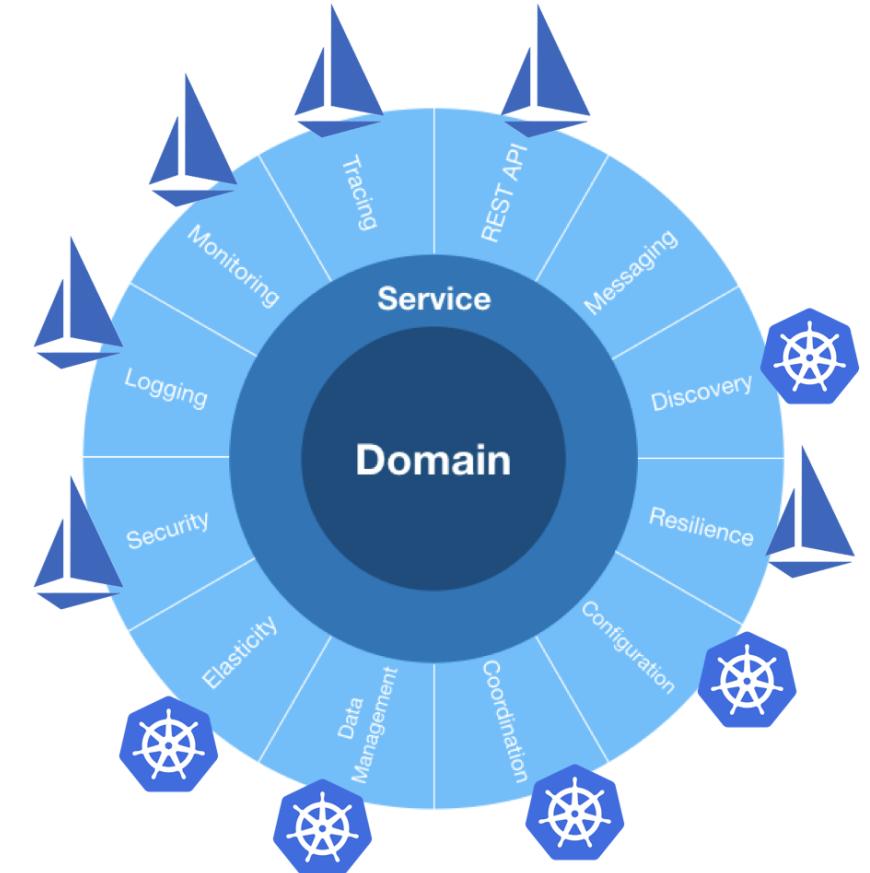
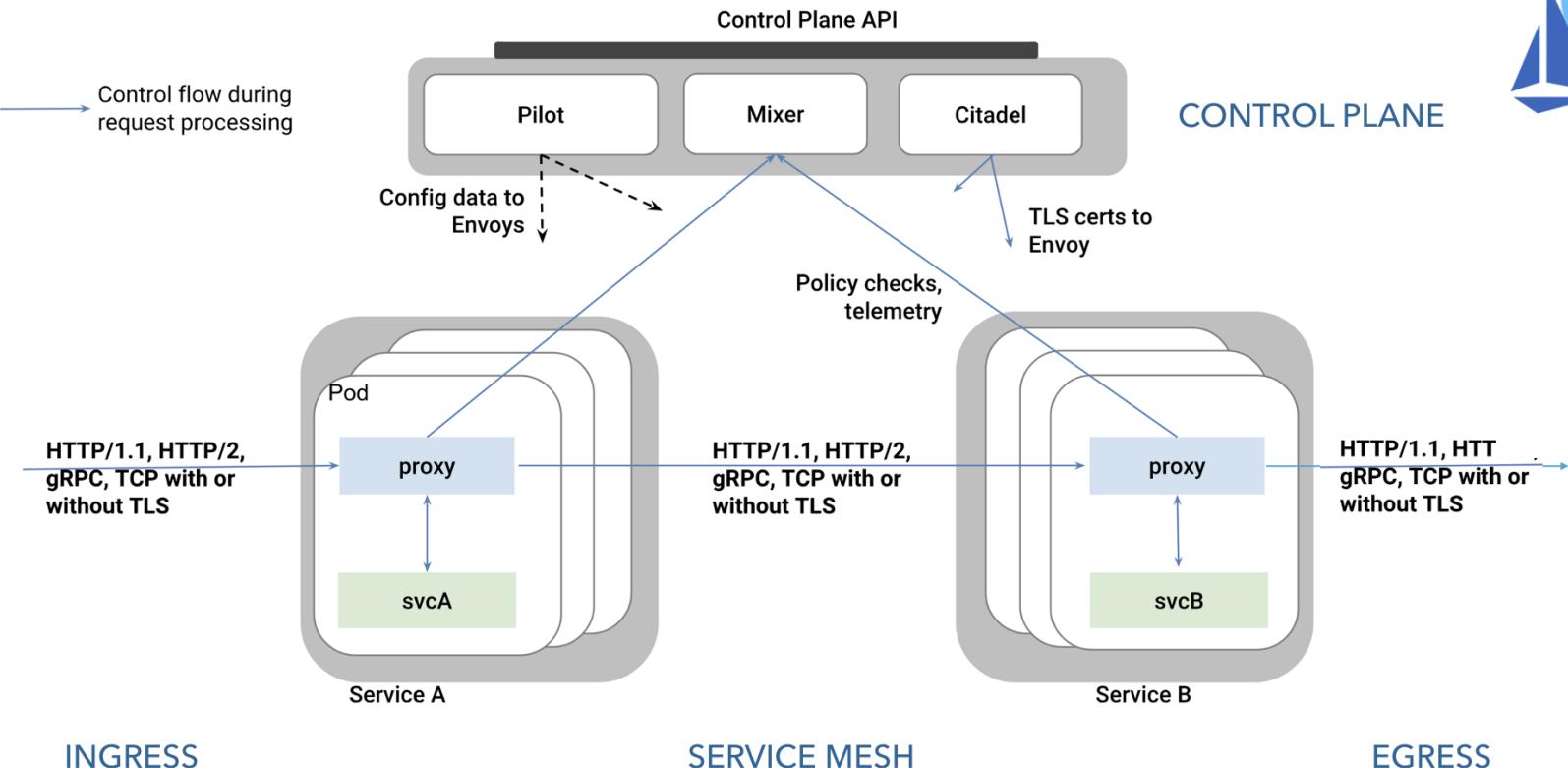


Technische Aspekte von Microservices: Infrastruktur Lösung

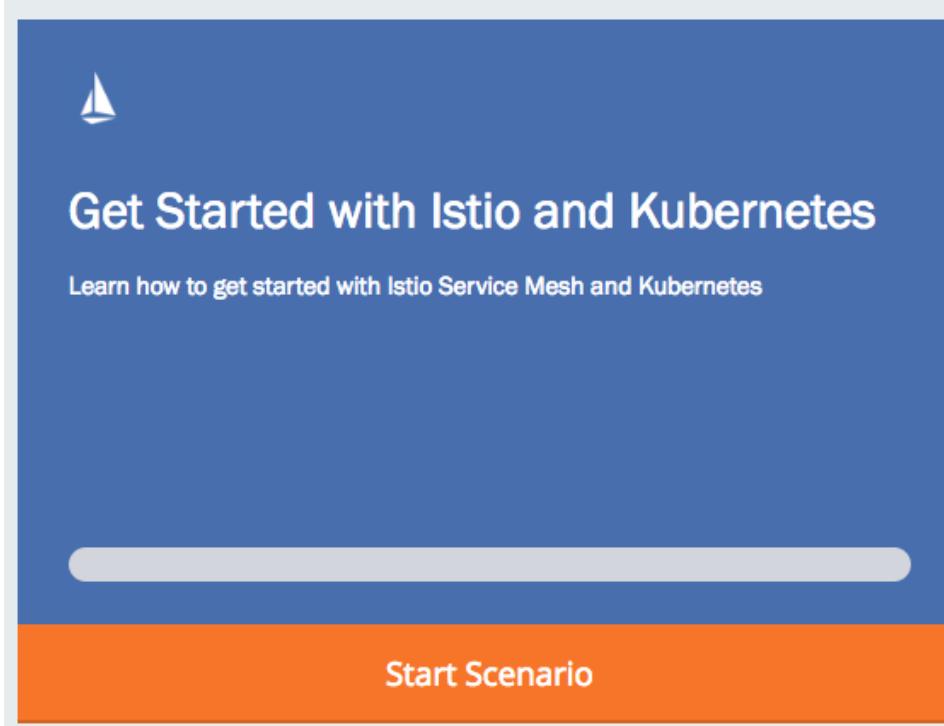


Istio Service Mesh

- Open-Source-Projekt von Google, IBM, Lyft, RedHat u.A.
- Aufsatz auf Kubernetes, der wichtige technische Aspekte auf Infrastruktur-Seite ergänzt



Die heutige Übung



<https://katacoda.com/courses/istio>