

Cloud Computing Cloud-Architektur

Walmart

Supercentre



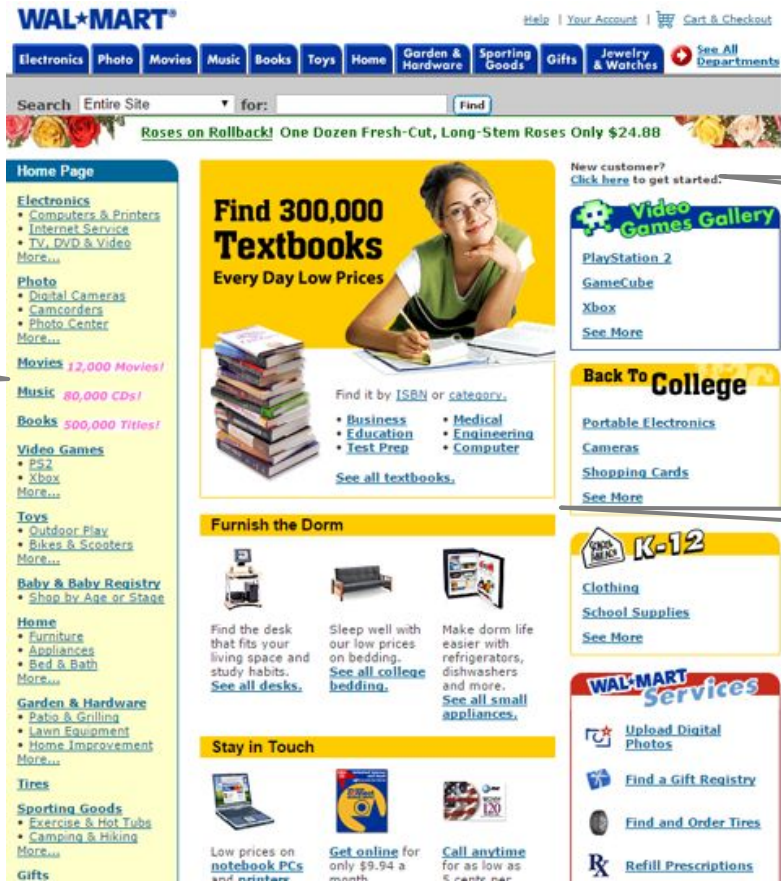
Enter

Exit

Low Prices
\$2



2002



Kategorien


Warenkorb

Registrierung

Artikel

Dienste

2008


Walmart  Free Shipping with **site to store**
Save money. Live better. Store Finder Local Ad Top Products Sign In/New Track Orders

See All Departments Search Entire Site FIND Help Cart


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

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](#)

Internet Archive
WayBack Machine
http://ad.yieldmanager.com/st?ad_type

Save on Top-Brand Electronics


Savings Add Up
[Clearance](#)
[Rollbacks](#)
[\\$4 Prescriptions](#)
[Video Games Under \\$20](#)
[Clearance on Baby Items](#)

What's New
[Classifieds](#)
[Halloween](#)
[Elevenmoms](#)
[Better Homes](#)

Zusätzliche
Dienste

2011

Walmart
Save money. Live better.

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](#)

See All Departments

- Electronics & Office
- Movies, Music & Books
- Home, Furniture & Outdoor
- Apparel, Shoes & Jewelry
- Baby & Kids
- Toys & Video Games
- Sports & Fitness
- Auto & Home Improvement
- Photo
- Crafts & Party Supplies
- Pharmacy, Health & Beauty
- Grocery & Pets
- College 2011

Save Big Now

- Amazing Xbox 360 Savings
- Best-Selling Bikes from \$79
- Fragrances from \$5
- Furniture Rollbacks
- Save 10¢ per Gallon on Gas









What's New

- Captain America Is Here
- HP Dandelion Breeze Laptop
- Summer Fun

In the Spotlight

Search

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.

[School Supplies](#)
[Uniforms](#)


Have Your Fun in the Sun

Enjoy hot summer savings on everything you'll need for some outdoor family fun.

[Pools & Waterslides](#)
[Swing Sets](#)
[Trampolines](#)

NEW
Watch New Releases Instantly

WU Digital Movies
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

Walmart  [Hello, Sign In](#) [My Account](#)

[All Departments](#) [Daily Savings Center](#) [My Local Store](#) [Pick it up TODAY](#) [Tips & Ideas](#) [FREE Returns](#)

Zusätzliche
Dienste.

Don't miss a single chance to save
Get the Walmart Browser App! [Download now](#)

Featured products

 <p>\$299.00 (was \$329.00) Save \$30.00 HP Pavilion 15" i5-4210U 15.6" Diagonal, 1366 x 768</p>	 <p>\$35.00 Google Chromecast</p>	 <p>\$331.76 LG Basic Chromebook K10-400</p>	 <p>\$194.24 Lenovo PC 11.6" i21 Chromebook</p>	 <p>\$276.45 (was \$315.00) Save \$38.55 ASUS K551MA 15.6" 15.6" Diagonal, 1366 x 768</p>
--	---	--	---	---



Customers also viewed these products

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>

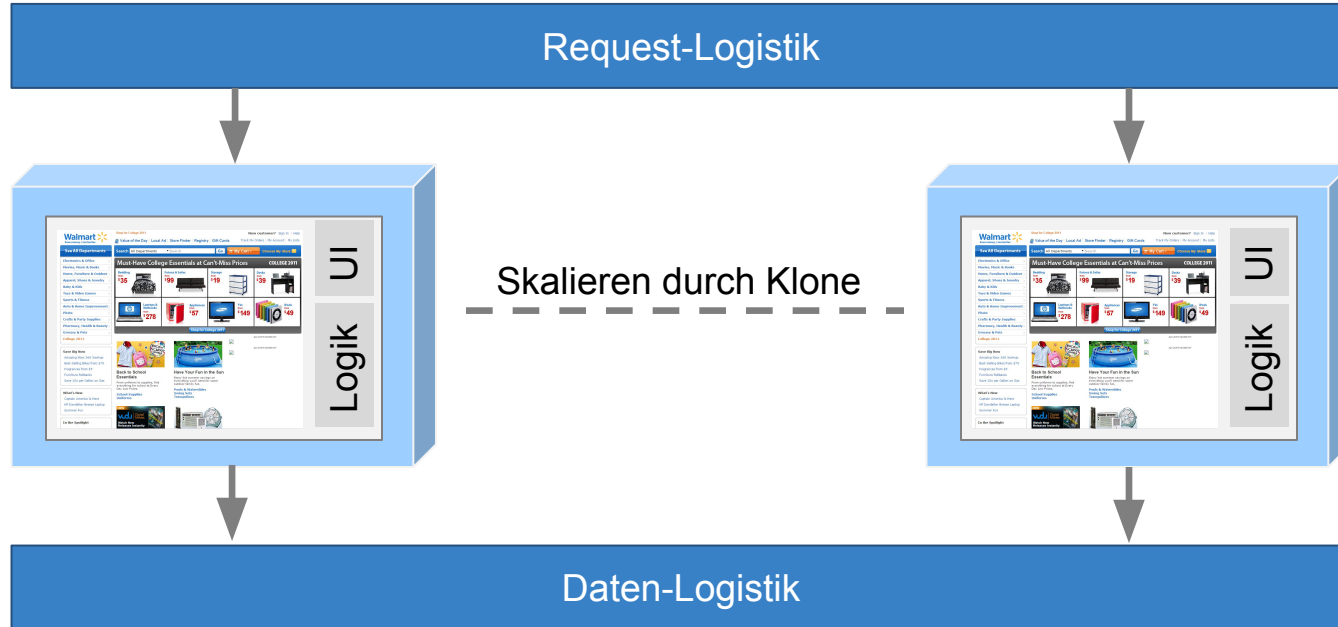
- "The Walmart [...] servers [...] were able to handle all mobile Black Friday traffic with about 10 CPU cores and 28Gb RAM."
- "On Thanksgiving weekend, Walmart servers processed 1.5 billion requests per day. 70 percent of which were delivered through mobile."

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

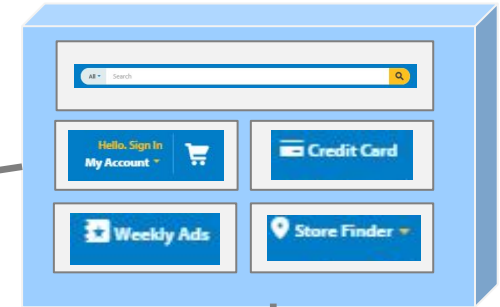
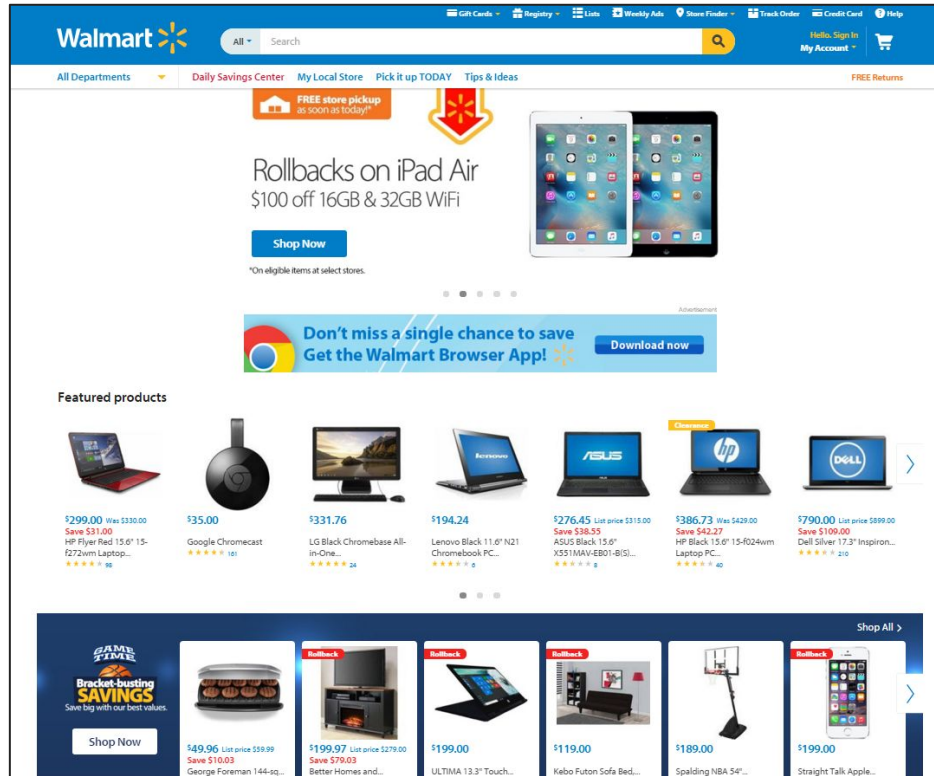


WHAT DID THEY DO?

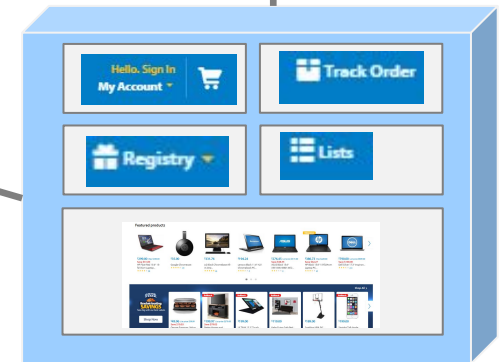
Von Betriebsmonolithen ...



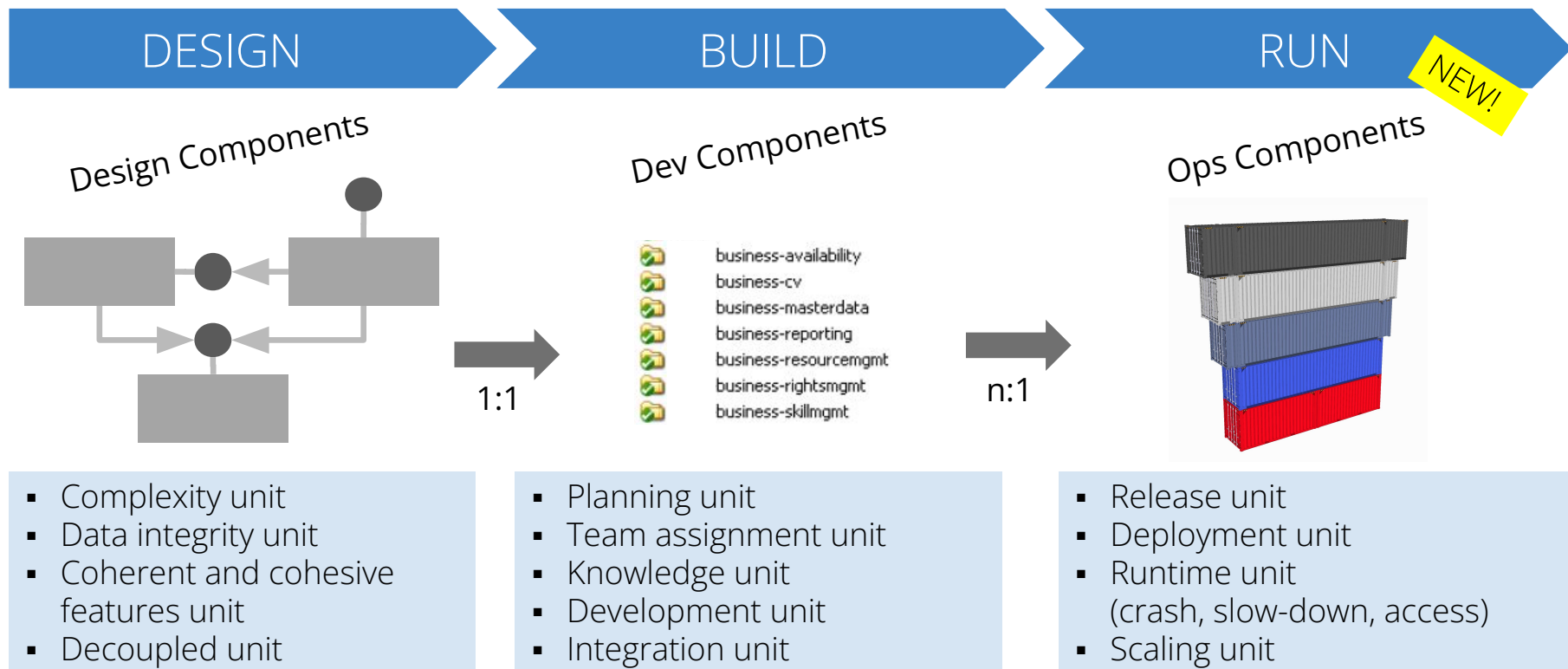
... zu Betriebskomponenten



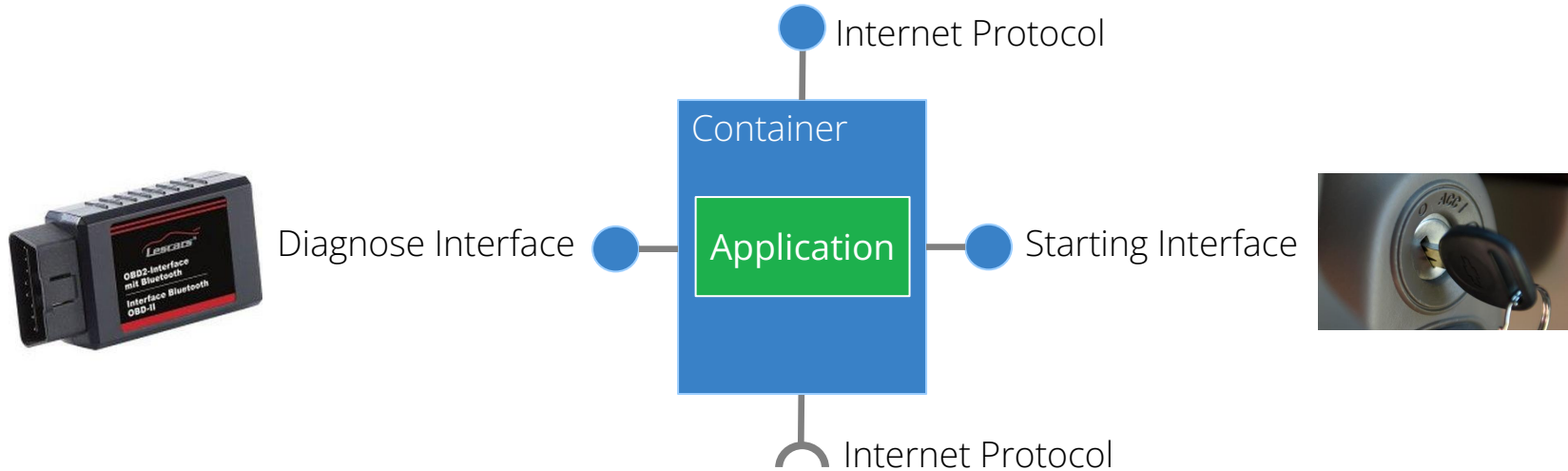
Skalieren durch
Verteilung und Klonen



Cloud Native Application Development: Components All Along the Software Lifecycle



Die Anatomie einer Betriebs-Komponente



Dev Components



Ops Components

Guter Ausgangspunkt

System

Subsystems

Components

Services

Monolith

Macroservices

Microservices

Nanoservices

Decomposition Trade-Offs

- + More flexible to scale
- + Runtime isolation (crash, slow-down, ...)
- + Independent releases, deployments, teams
- + Higher utilization possible

- Distribution debt: Latency
- Increasing infrastructure complexity
- Increasing troubleshooting complexity
- Increasing integration complexity



Regeln, Gebote, Trugschlüsse

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

Die 5 Gebote der Cloud

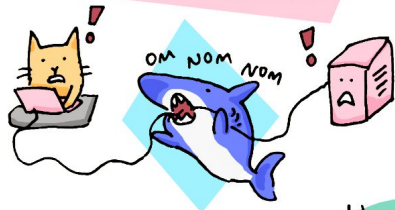
1. Everything Fails All The Time.
2. Focus on MTTR and not on MTTF.
3. Respect the Eight Fallacies of Distributed Computing.
4. Scale out, not up.
5. Treat resources as cattle, not pets.



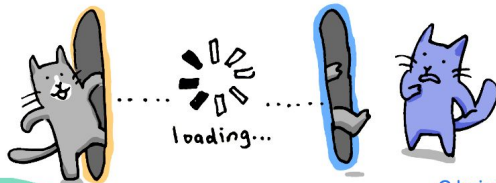
Quelle: https://de.wikipedia.org/wiki/Zehn_Gebote

Eight fallacies of distributed computing

① The network is reliable



② Latency is ZERO



③ Bandwidth is infinite



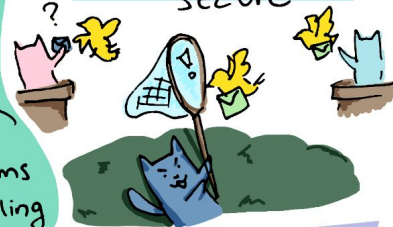
⑧ The network is homogeneous



the 8 Fallacies of Distributed Computing

Originally formulated by L. Peter Deutsch & colleagues at Sun Microsystems in 1994; #8 added in 1997 by James Gosling

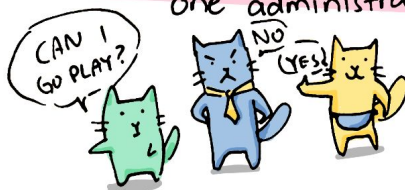
④ The network is secure



⑦ Transport costs \$0



⑥ There is only one administrator



⑤ Topology doesn't change



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 Microservice 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 Microservice kann ausgetauscht werden, ohne dass die Nutzer davon etwas bemerken.



Plattform und Reifegrad

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:

- Metriken sammeln
- Logs sammeln
- Trace sammeln

Typische Aufgaben:

- Service Discovery
- Load Balancing
- Circuit Breaker
- Request Monitoring

Typische Aufgaben:

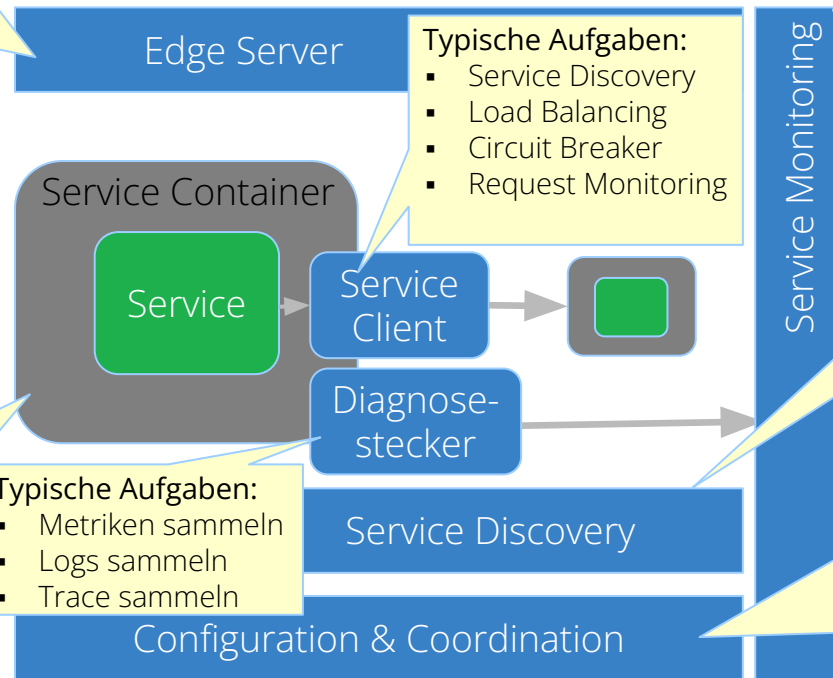
- Aggregation von Metriken
- Sammlung von Logs
- Sammlung von Traces
- Analyse / Visualisierung
- Alerting

Typische Aufgaben:

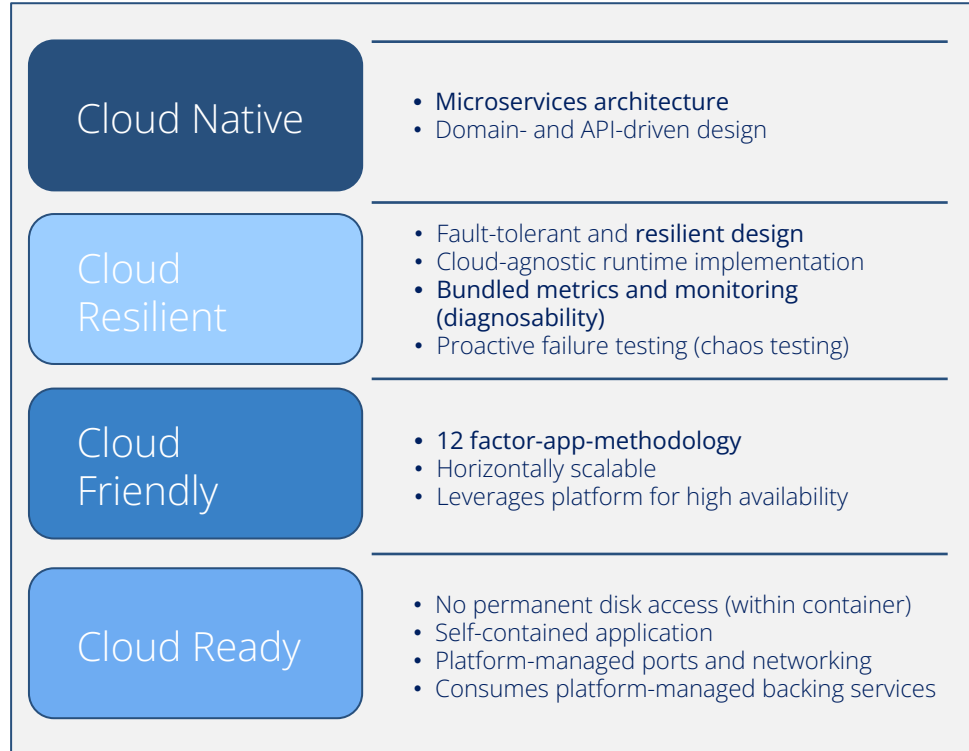
- Service Registration
- Service Lookup
- Service Description
- Membership Detection
- Failure Detection

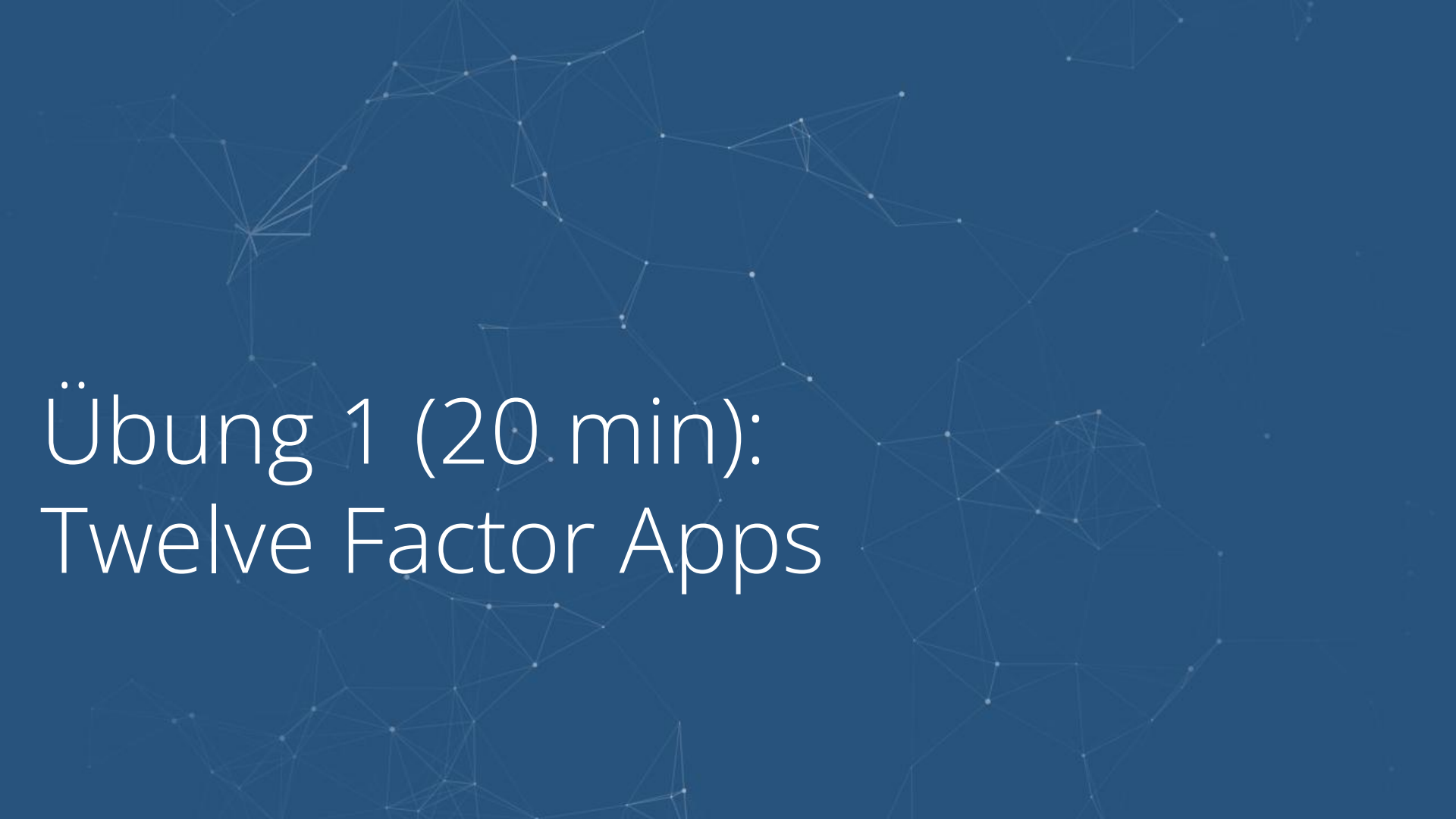
Typische Aufgaben:

- 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



Das Cloud Native Application Reifegradmodell





Übung 1 (20 min): Twelve Factor Apps

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>



Resilienz

Resilienz

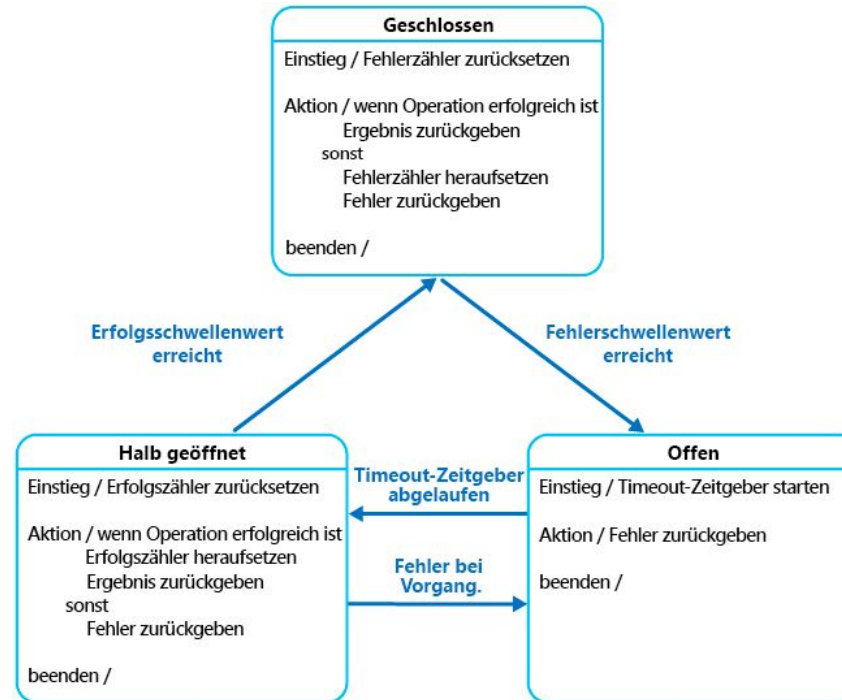
$$\text{Verfügbarkeit} = \text{MTTF} / (\text{MTTF} + \text{MTTR})$$



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

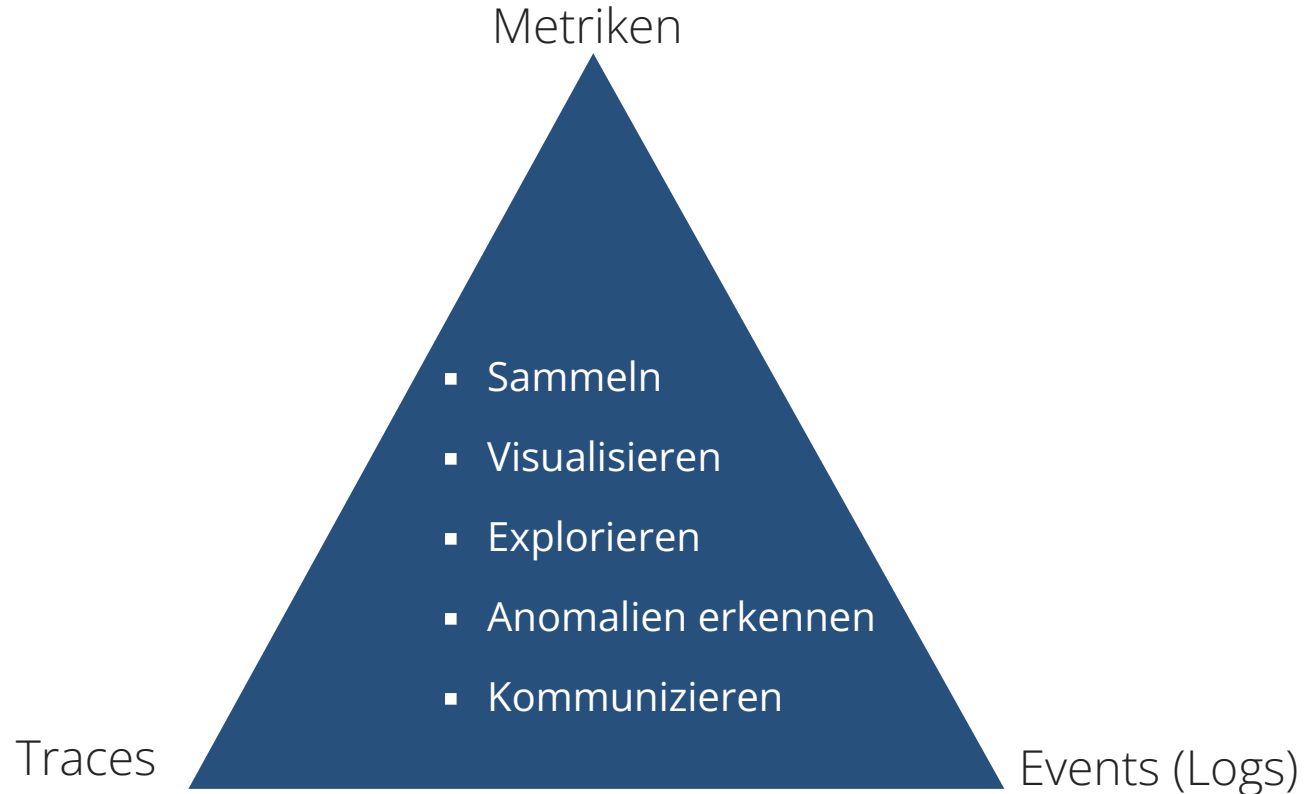
- Ohne dass es der Nutzer merkt (Bestfall)
- Mit ein einer „graceful degradation“ des Services (schlechtester Fall)

Resilienz-Pattern: Circuit Breaker



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

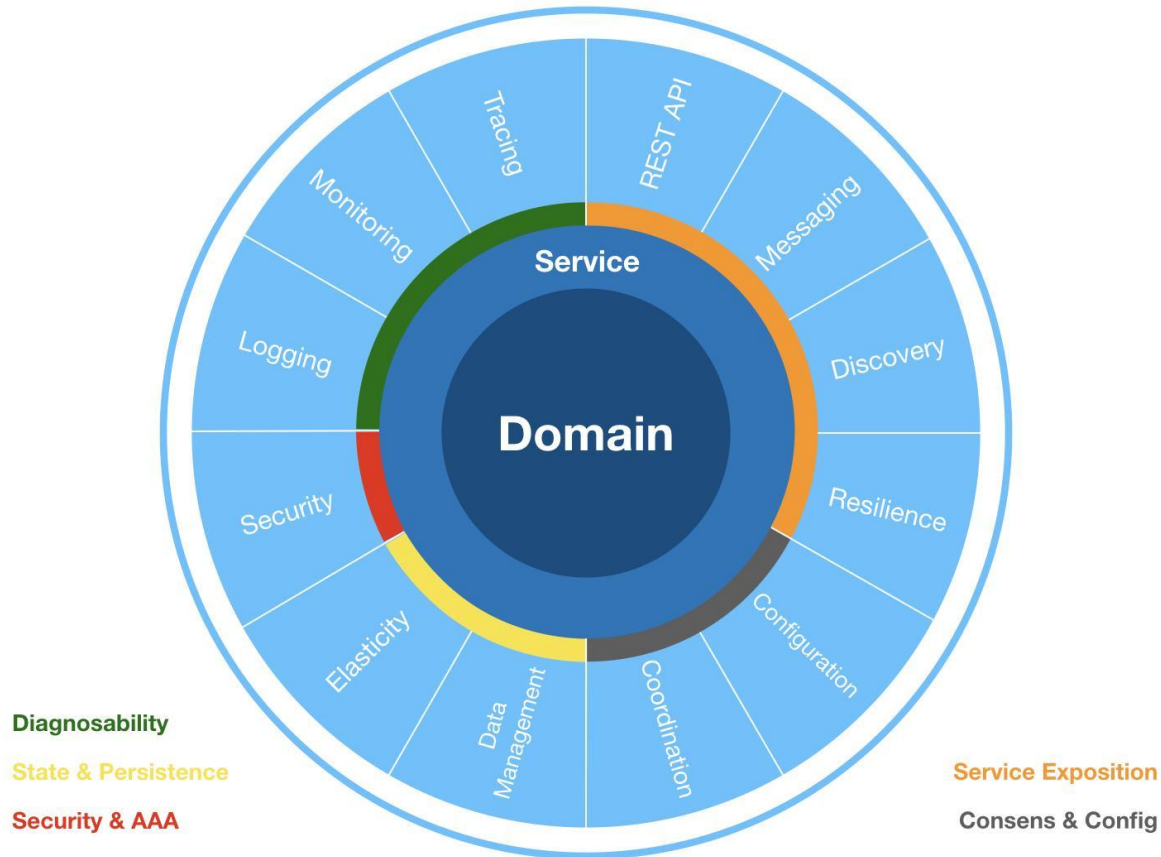
Diagnostizierbarkeit



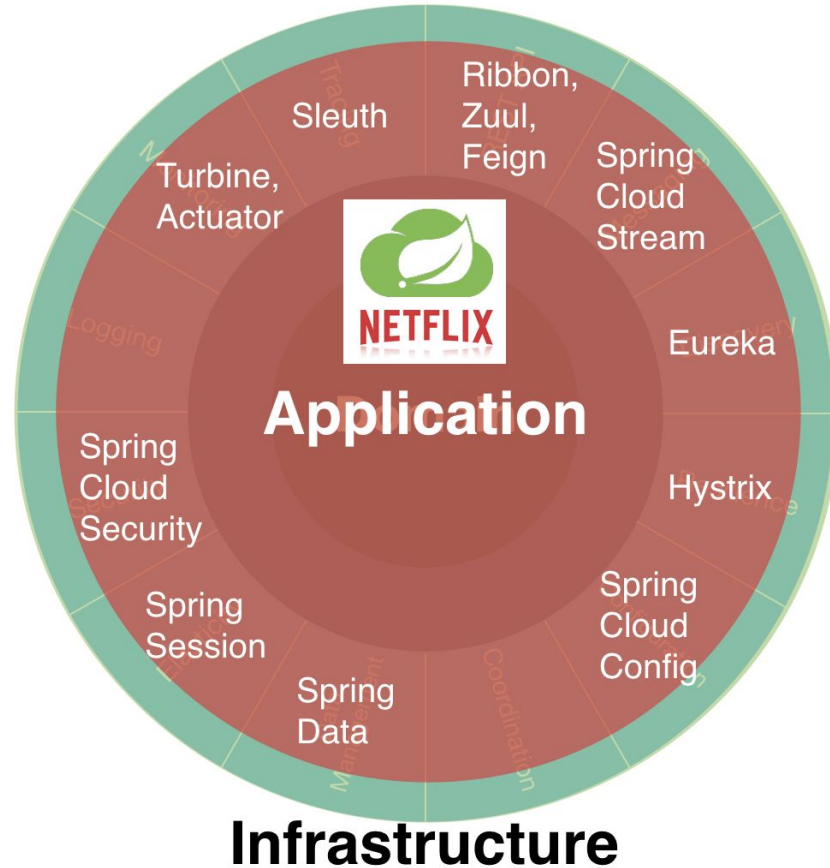


Technische Aspekte von Microservices

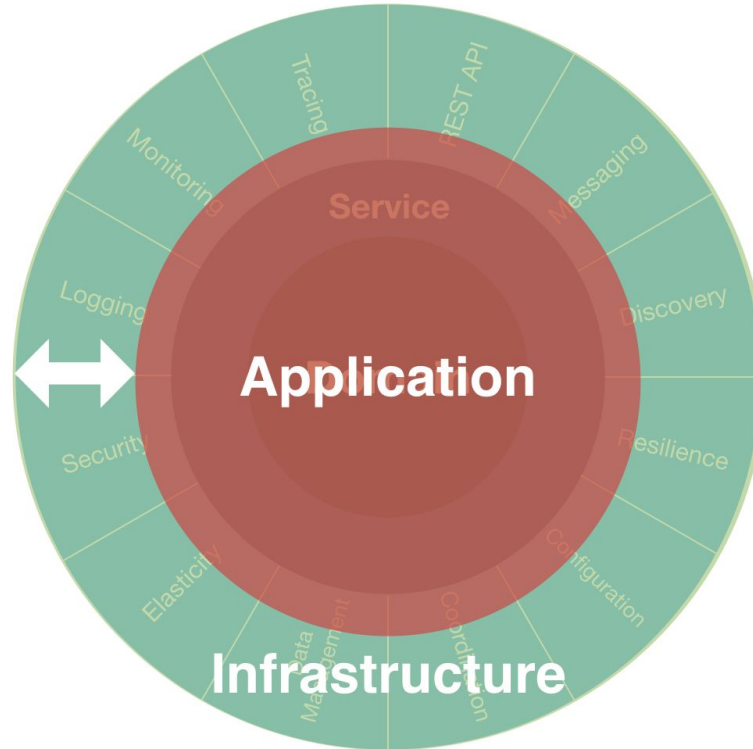
Technische Aspekte von Microservices

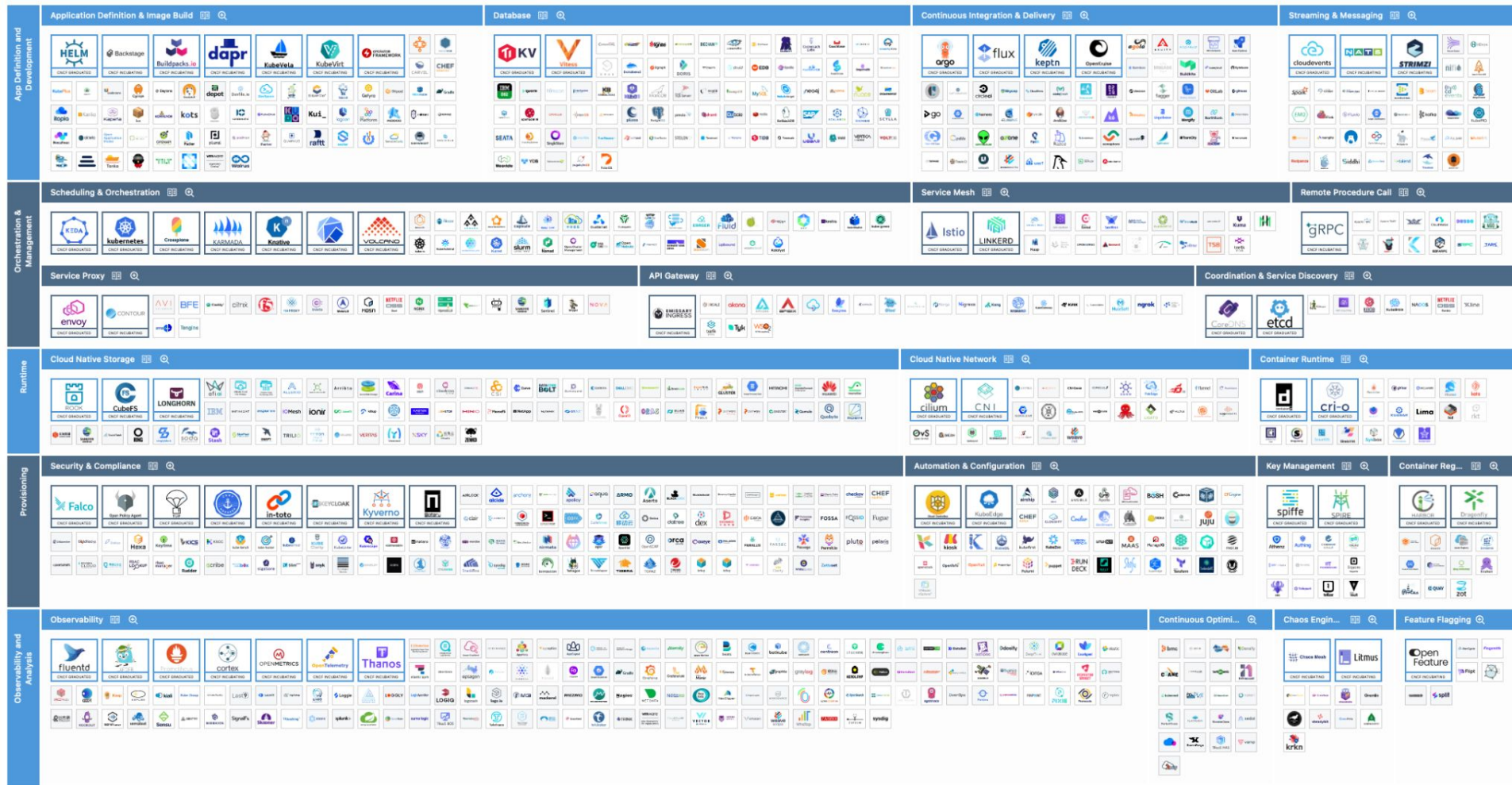


Technische Aspekte von Microservices: Library Lösung



Technische Aspekte von Microservices: Infrastruktur Lösung





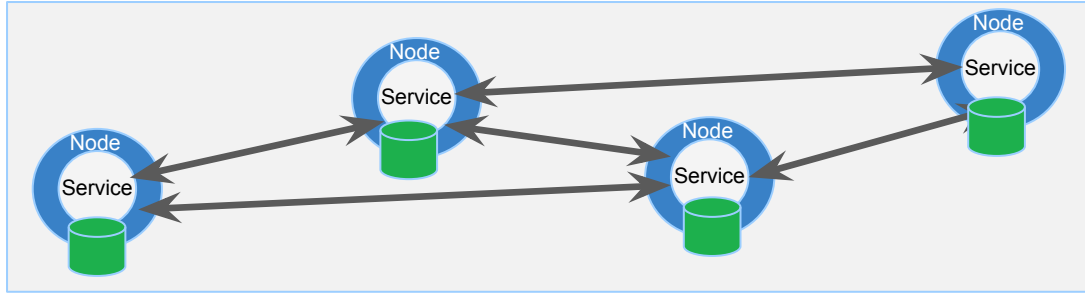
<https://landscape.cncf.io/>

<https://github.com/cncf/landscape>



Konfiguration & Koordination: Verteilter Zustand und Konsens

Ein verteilter Konfigurationsspeicher



Wie wird der Zustand des Konfigurationsspeichers im Cluster synchronisiert?

Das CAP-Theorem

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

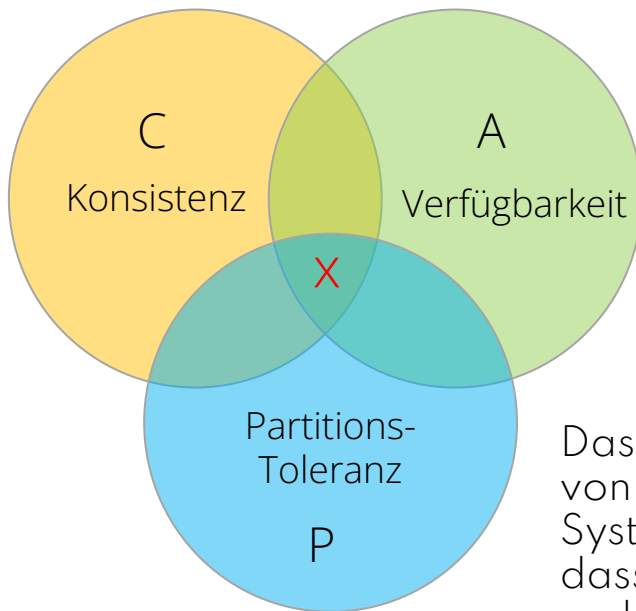
Es gibt drei wesentliche Eigenschaften, von denen ein verteiltes System nur zwei gleichzeitig haben kann:

- Konsistenz (C)
- Verfügbarkeit (A)
- Partitionstoleranz (P)

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

Das CAP-Theorem

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



Jede Anfrage erhält eine Antwort (keinen Fehler) in akzeptabler Zeit. 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.

Im Falle einer Netzwerk-Partition muss man sich zwischen Konsistenz und Verfügbarkeit entscheiden, beides geht nicht.

Gossip Protokolle für Hoch-Verfügbarkeit

Grundlage: Ein Netzwerk an Agenten mit eigenem Zustand Agenten verteilen einen Gossip-Strom

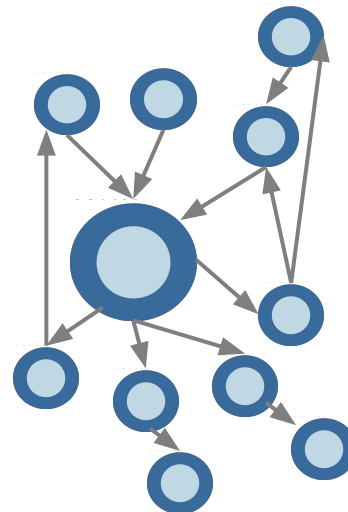
- Nachricht: Quelle, Inhalt / Zustand, Zeitstempel
- Nachrichten werden in einem festen Takt periodisch versendet an eine bestimmte Anzahl anderer Knoten (Fanout)

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, wenn

- die gleiche Nachricht von mehreren Seiten gehört wurde
- die Nachricht von Knoten stammt, denen der Agent vertraut
- keine aktuellere Nachricht vorhanden sind



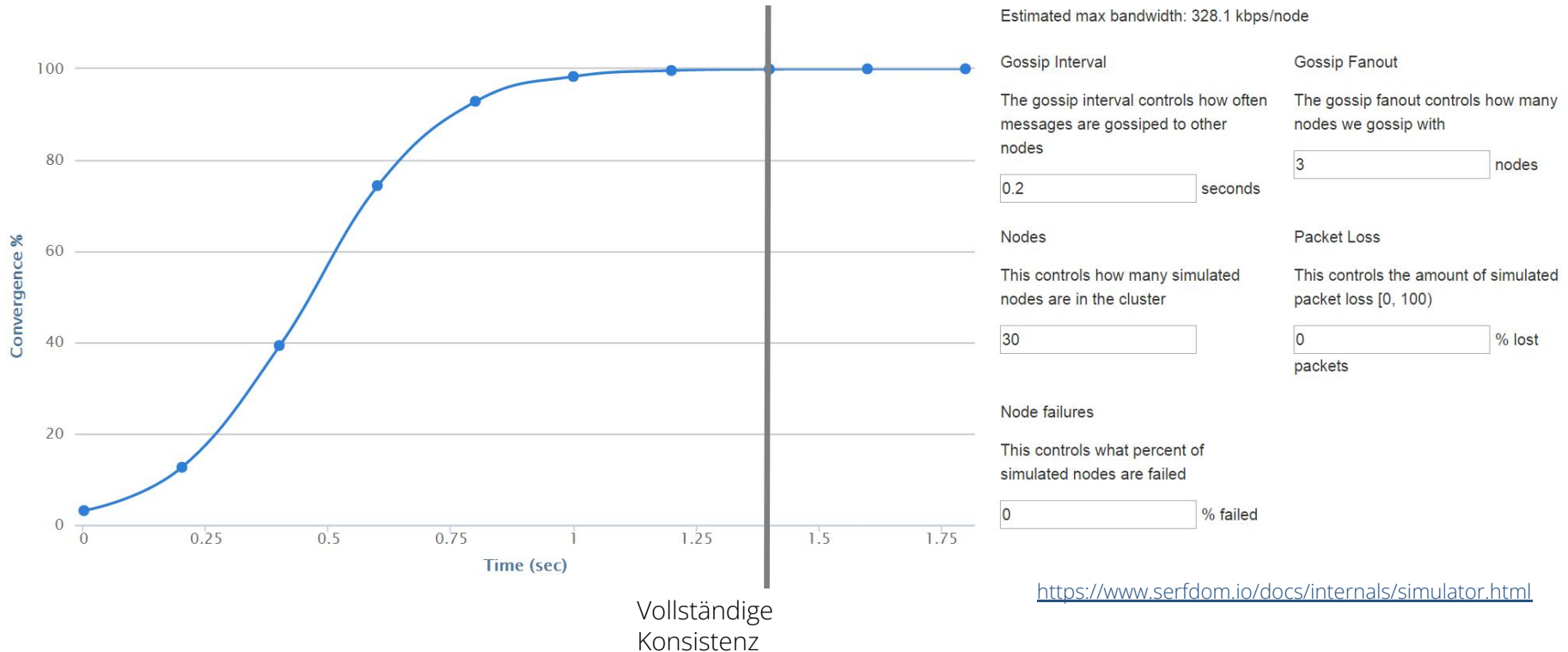
Vorteile:

- 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



Protokolle für verteilten Konsens: 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.

Vorteile:

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

Nachteile:

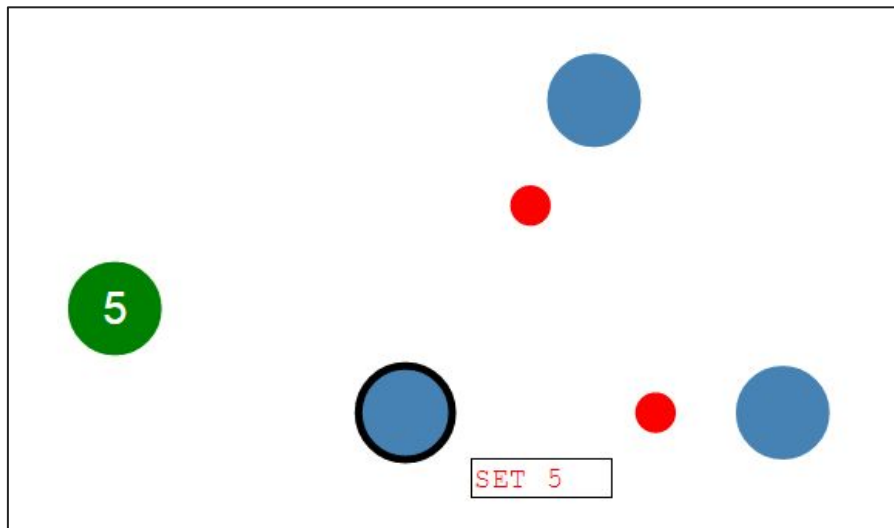
- Der zentrale Leader-Agent limitiert den Durchsatz an Änderungen.
- Nicht hoch-verfügbar: Bei einer Netzwerk-Partition kann die kleinere Partition nicht weiterarbeiten. Ist die Mehrheit in keiner Partition, so kann insgesamt nicht weiter gearbeitet werden.

Konkrete Konsens-Protokolle: Raft, Paxos

Das Raft Konsens-Protokoll

Ongaro, Diego; Ousterhout, John (2013).

["In Search of an Understandable Consensus Algorithm".](#)



<http://thesecretlivesofdata.com/raft>

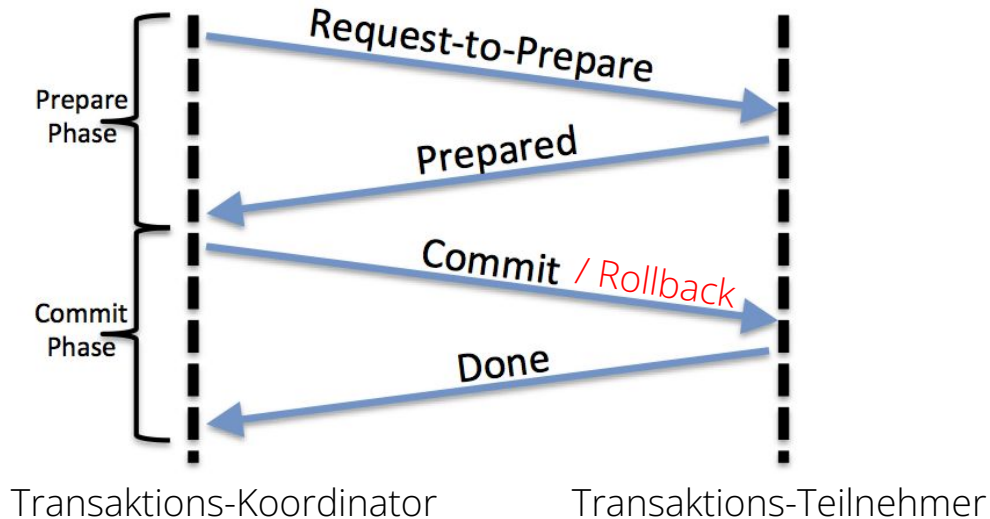
<https://raft.github.io/>



Übung 2 & 3 (30 min): Raft Konsens Protokoll etcd

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.



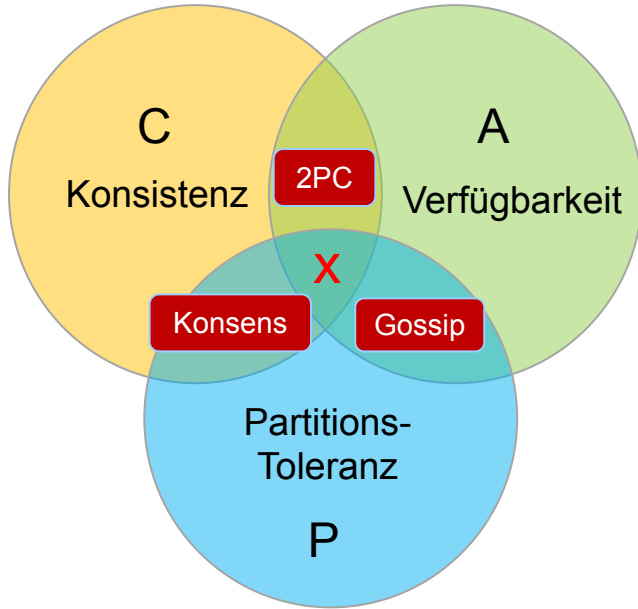
Vorteil:

- Alle Knoten sind konsistent zueinander.

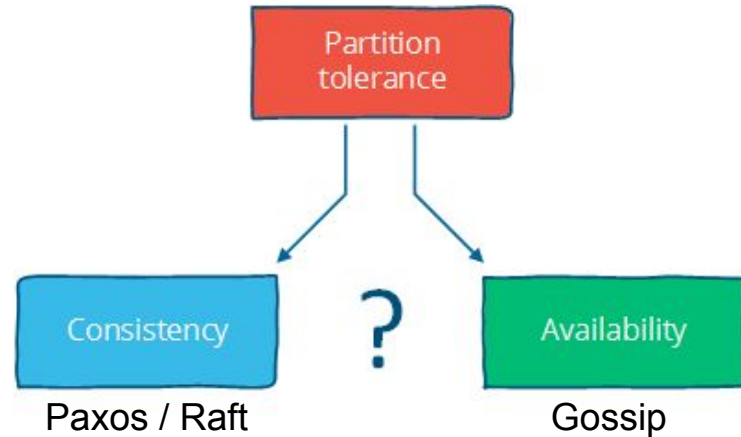
Nachteile:

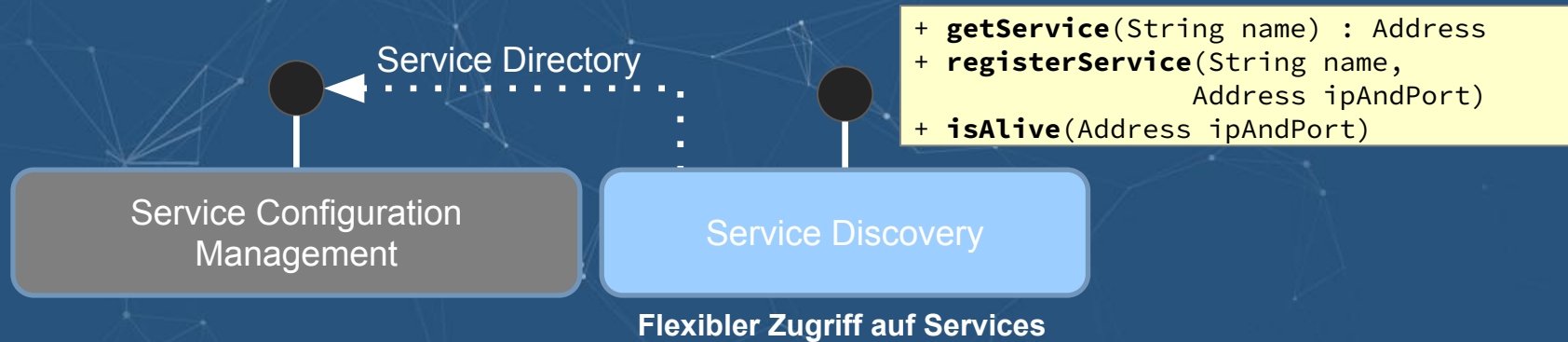
- Zeitintensiv, da stets alle Knoten zustimmen müssen.
- Das System funktioniert nicht mehr, sobald das Netzwerk partitioniert ist.

Die vorgestellten Protokolle und das CAP Theorem



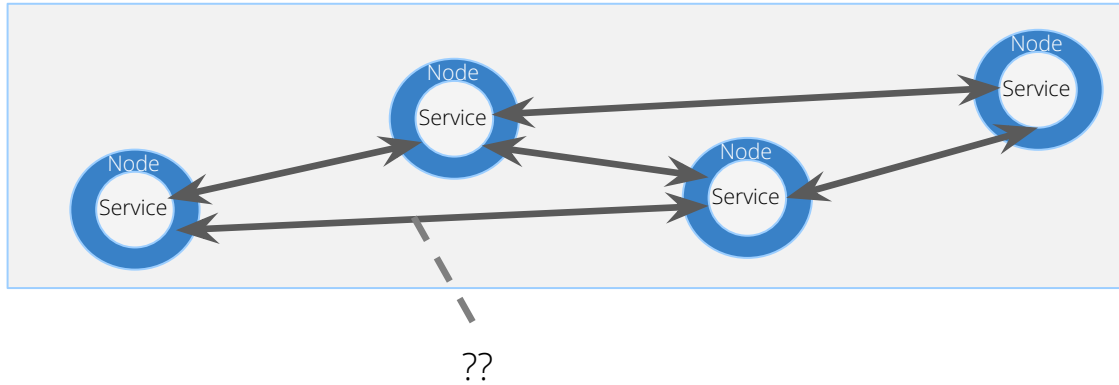
In der Cloud müssen Partitionen angenommen werden. Damit ist die Entscheidung binär zwischen Konsistenz und Verfügbarkeit.





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:**

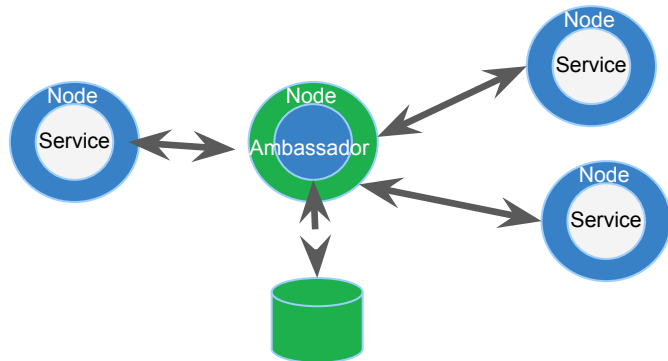
- Beobachtet das Cluster und erkennt neue und kranke/tote Knoten in seiner Gruppe.
- Hinterlegt die aktuell aktiven Knoten im Konfigurationsspeicher.


- **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





Übung 4: Consul & Traefik