

NetfliX - xilfteN

Die umgekehrte Architekturbewertung eines Internet-Giganten

STEFAN TOTH, EMBARC

JUG Kaiserslautern, 27. Mai 2015

Stefan Toth



stefan.toth@embarc.de



@st_toth



xing.to/sto



www.embarc.de
www.swamuster.de



Agenda

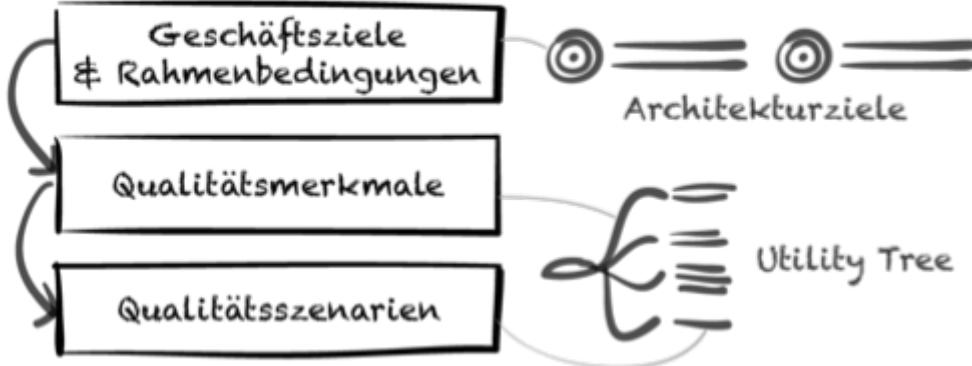
1

- 1 Einstieg**
- 2 Was ist Netflix?
- 3 Lösung im Detail
- 4 Ergebnisse zusammengefasst
- 5 Weitere Informationen



Architekturbewertung

Anforderungen konkretisieren:



Bewertung:

Passt meine Architektur zu diesem Kontext?

- Stärken
- Schwächen
- Risiken
- Probleme
- Kompromisse

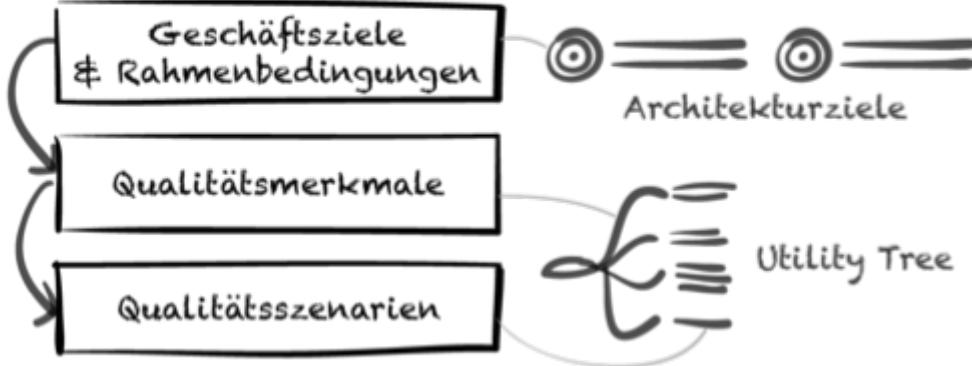
Methoden:

- **ATAM** – Architecture Tradeoff Analysis Method
- **CBAM** – Cost-Benefit Analysis Method
- **SACAM** – Software Architecture Comparison Analysis Method
- **LAAAM** – Lightweight Architecture Alternative Assessment Method
- ...



Umgekehrte Bewertung?

Anforderungen konkretisieren:



Bewertung:

Passt meine Architektur zu diesem Kontext?

- Stärken
- Schwächen
- Risiken
- Probleme
- Kompromisse

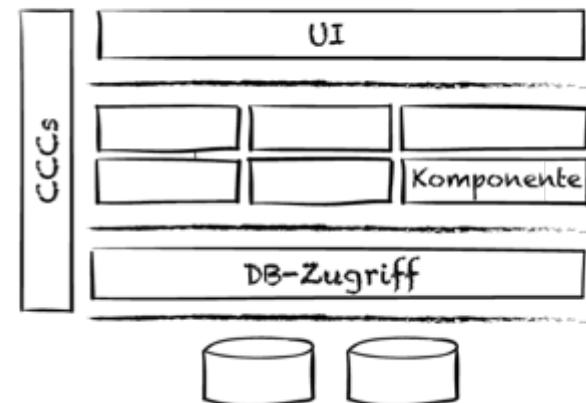
Umgekehrte Bewertung:

Welcher Kontext macht diese Architektur passend?

- Ziele
- Qualitätsmerkmale
- Qualitätsszenarien
- Rahmenbedingungen
- Kompromissaussagen

Architektur analysieren:

Ansätze
Entscheidungen
Lösungen
Probleme
Prinzipien
...



Warum umgekehrt bewerten?



- Weil die Netflix Architekturkonzepte „trendig“ sind
 - Cloud
 - Microservices
 - Reactive Extensions
 - NoSQL Datenbanken
 - ...
- Weil sich viele fragen: Passt das auch zu uns?
- Weil ich „kommt drauf an“ zu allgemein finde ...

Agenda

2

- 1 Einstieg**
- 2 Was ist Netflix?**
- 3 Lösung im Detail
- 4 Ergebnisse zusammengefasst
- 5 Weitere Informationen





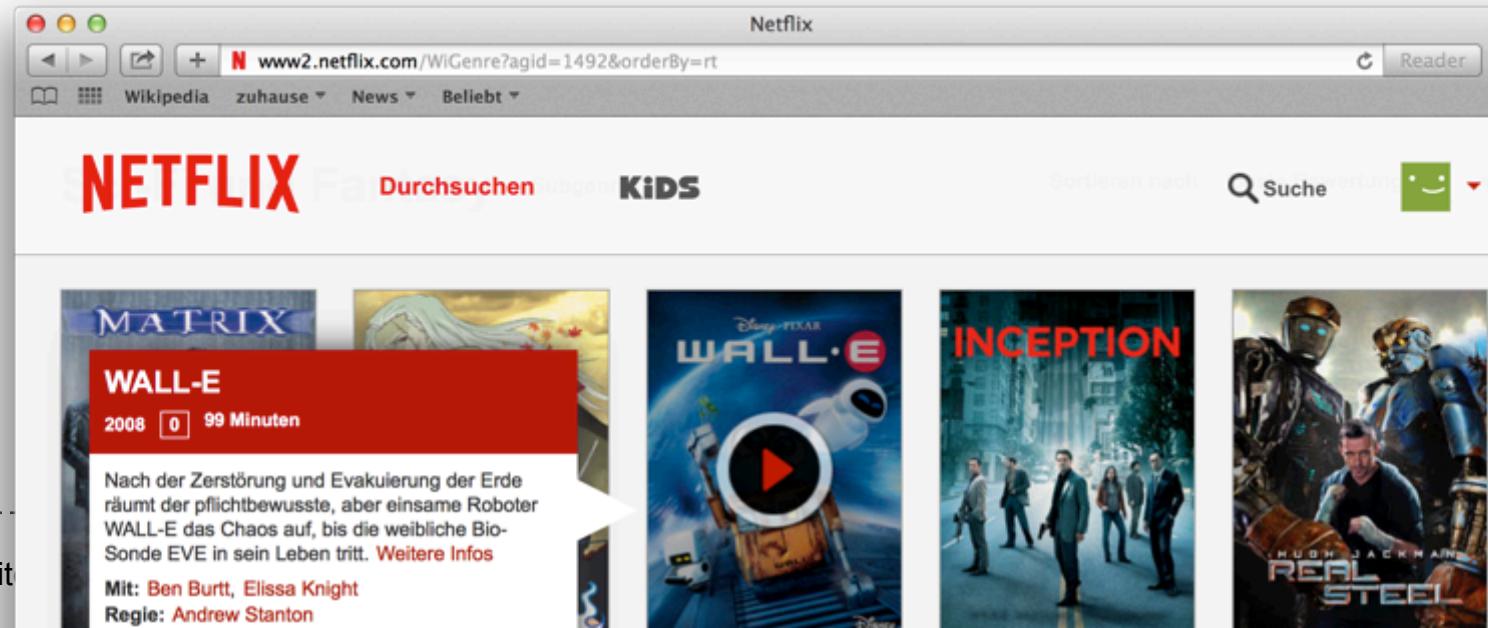
NETFLIX



Was ist Netflix?

"Netflix is the world's leading Internet television network with over 57 million members in nearly 50 countries enjoying more than two billion hours of TV shows and movies per month, including original series. For one low monthly price, Netflix members can watch as much as they want, anytime, anywhere, on nearly any Internet-connected screen. Members can play, pause and resume watching, all without commercials or commitments."

→ <http://ir.netflix.com>



‘Netflix is the king of online streaming, using more global bandwidth than cat videos and piracy combined.’

House of Cards

★★★★★ 2013 TV-MA 1 Season HD 51

Sharks gliding ominously beneath the surface of the water? They're a lot less menacing than this Congressman.



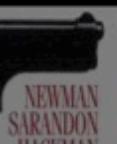
This winner of three Emmys, including Outstanding Directing for David Fincher, stars Kevin Spacey and Robin Wright.



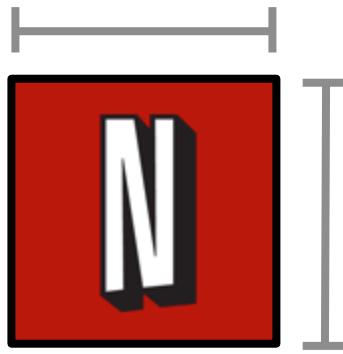
Because you watched Orange Is the New Black



Because you watched Red Lights



Netflix – Wie groß ist ‘groß’?



- **600+** Services und Applikationen
- **Milliarden** Requests am Tag
- **> 2 Milliarden** Stunden Filme und Serien
- **10.000e** Ec2 Instanzen in mehreren AWS Regionen/Zonen
- Cassandra NoSQL DB in einem Regionen-übergreifenden, globalen Ring mit **Terabytes** an Daten
- Zu Spitzenzeiten **1/3** der Internet-Bandbreite in den USA (Downstream)

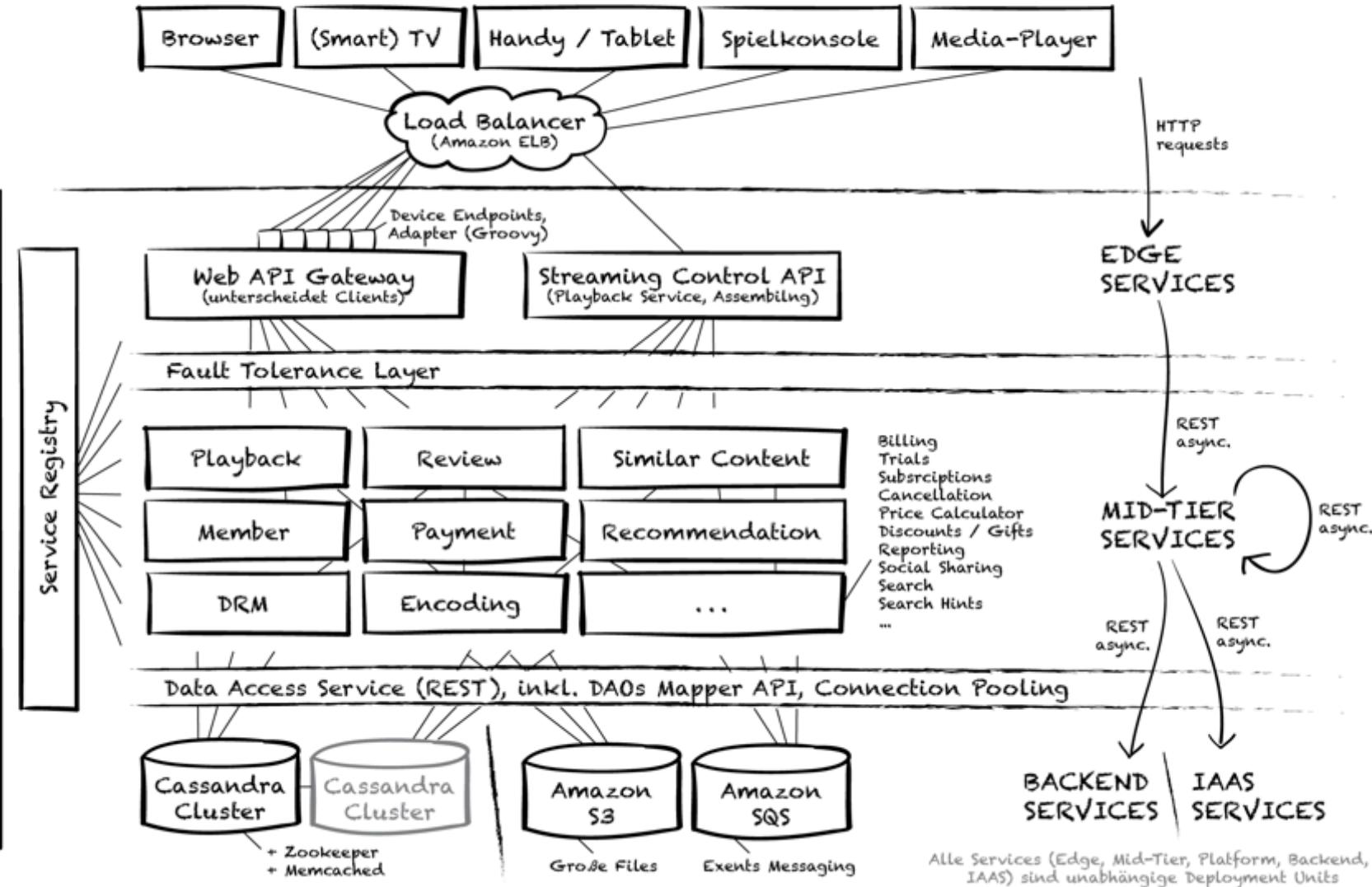
Netflix Architekturüberblick I



Netflix Architekturüberblick II

PLATFORM SERVICES

i18n, Security, Monitoring, Configuration, Logging, Rx für Java, dyn. Routing, Caching, DI-Container



Agenda

3

- 1 Einstieg**
- 2 Was ist Netflix?**
- 3 Lösung im Detail**
- 4 Ergebnisse zusammengefasst**
- 5 Weitere Informationen**



Lösung im Detail (1)



Microservices



Was sind Microservices?

“In short, the microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API.”



(James Lewis, Martin Fowler)

Charakteristische Eigenschaften

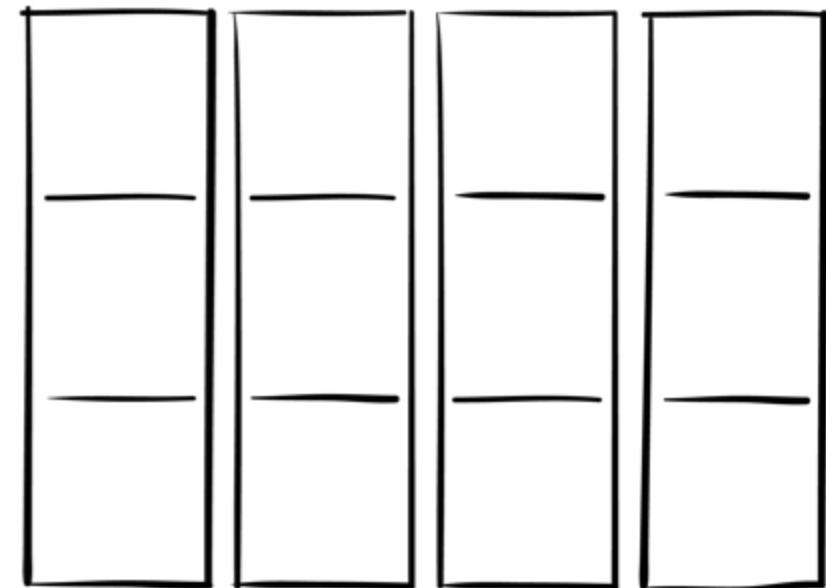
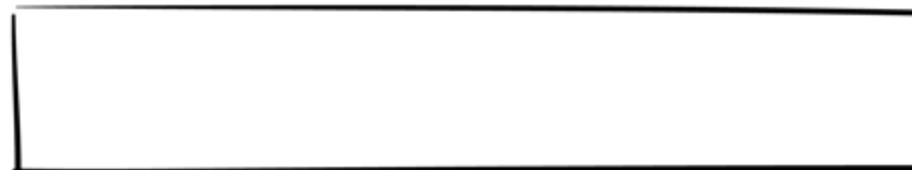
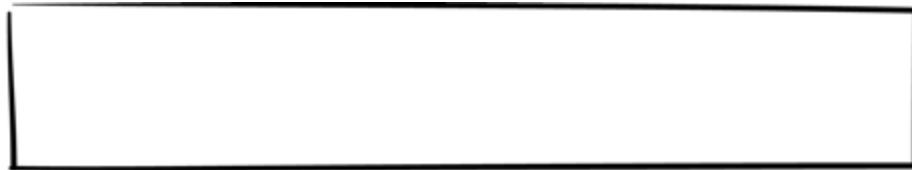
- Zerlegung in relativ kleine (fachliche) Services
- Services sehr lose gekoppelt
- Services einzeln installierbar und upgradebar
- Dezentrale Datenhaltung
- Hoher Freiheitsgrad bei Technologieauswahl

Im Prinzip...

Schichten



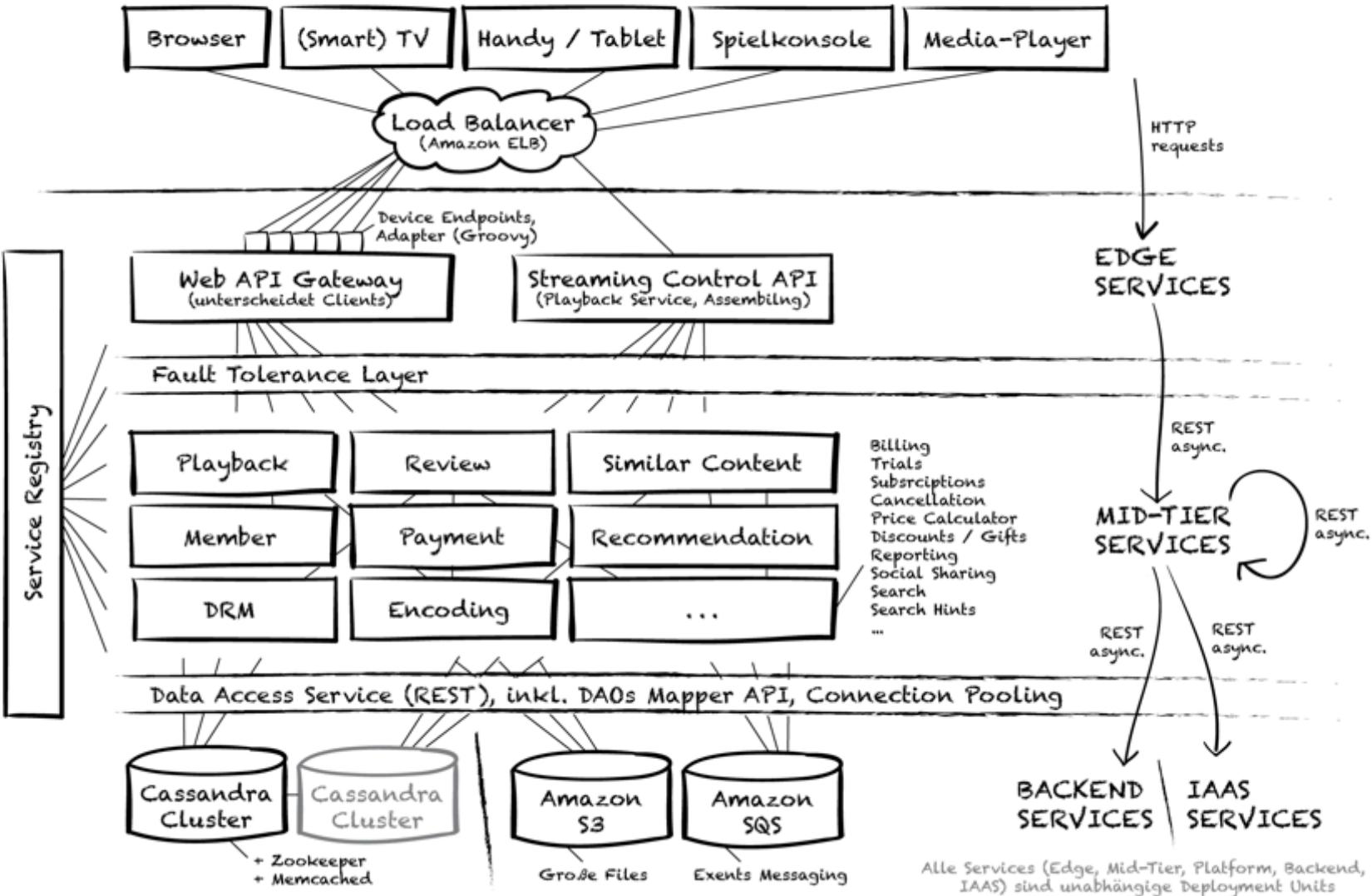
Vertikalen



Microservices eingeordnet ...

PLATFORM SERVICES

i18n, Security, Monitoring, Configuration, Logging, Rx für Java, dyn. Routing, Caching, DI-Container

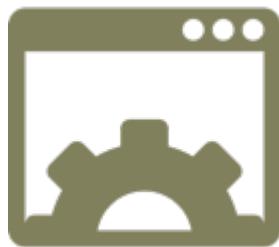


Microservices bei Netflix



Funktionalität

- Netflix-System besteht aus 600+ Services
- Beispiele für Services („Applications“)
 - Registrierung (Sign-up)
 - Suche (Search)
 - Empfehlungen (Recommendation)
 - Bewertungen (Ratings)
 - Leihhistorie (Rental History)
 - ...



Beispiel für eine Service-Trennung

Umgekehrt

www.netflix.com/search/Sc

Wikipedia zuhause News Beliebt

Reader

NETFLIX Durchsuchen KIDS

Sc

Personen

- Scarlett Johansson
- Arnold Schwarzenegger
- Til Schweiger
- Matthias Schweighöfer
- Michael Schweighöfer
- Taylor Schilling
- Marie-Luise Schramm
- David Schwimmer
- Scott McNeil
- Scott Lilly

The screenshot shows a web browser window with the Netflix logo at the top. The address bar displays 'www.netflix.com/search/Sc'. Below the address bar, there are links for 'Wikipedia', 'zuhause', 'News', and 'Beliebt'. On the right side of the header, there is a 'Reader' link. The main content area has a red border around the left sidebar. The sidebar contains a heading 'Personen' followed by a list of names. To the right of the sidebar, there are two rows of movie posters. The top row includes 'Charlie und die Schokoladenfabrik', 'Schmetterling und Taucherglocke', 'Babe', and 'schau mich an!'. The bottom row includes 'Scooby-Doo - Die Monster sind los', 'SCIENCE OF SLEEP Anleitung zum Träumen', 'Schwedisch für Fortgeschrittene', and 'Korblau wie wir Juchs'.

Microservices bei Netflix (org.)

Teams **volumfänglich** für ihre Services **verantwortlich**

- Entwicklung
- Release und Deployment
- Betrieb

Keine klassische **Management-Steuerung**

Keine Beeinflussung durch andere Teams oder
eine zentrale Instanz

- Wenige technologische Vorgaben
- Release nach Belieben



“Freedom & Responsibility”

Verwendete Technologien

Plattformen

Apache HTTP Server

Apache Tomcat

Bottle (Python)

...

Persistenz

Cassandra

RDBMS (MySQL)

in-memory caches

Amazon S3

CDN

...

Programmiersprachen

Java

Groovy

Scala

Python

JavaScript

Clojure

Dart

Ruby

C++

...



Konsequenzen für Netflix

- Neue Technologien sind leicht ausprobiert
- Technologie-Stack kann nach und nach modernisiert werden (keine langfristige Binding)
- Fehler in einem Service haben wenig Einfluss auf andere Services (Resilience-Grundlage)
- Gute Time-to-Market für neue Funktionalität

Herausforderungen



- Die Gesamtmenge der verwendeten Technologien ist sehr heterogen
- Abstimmung / Koordination schwierig
- Jedes Service-Team muss Know-How für eigentlich querschnittliche Themen haben

Lösung im Detail (2)



Netflix OSS Komponenten



Netflix Cloud Stack

Individuelle Software

Applikationen, Services, ...

PaaS

Laufzeitumgebung, Web-/Application-Server, Frameworks
für querschnittliche Aspekte, Management Tools, ...



IaaS

Virtuelle Maschinen, Netzwerkkommunikation,
Load Balancing, Datenspeicher, ...



The Netflix Open Source Platform Components fill gaps in Amazon Web Services. The goal is to make cloud infrastructure more robust, flexible and glitch free.

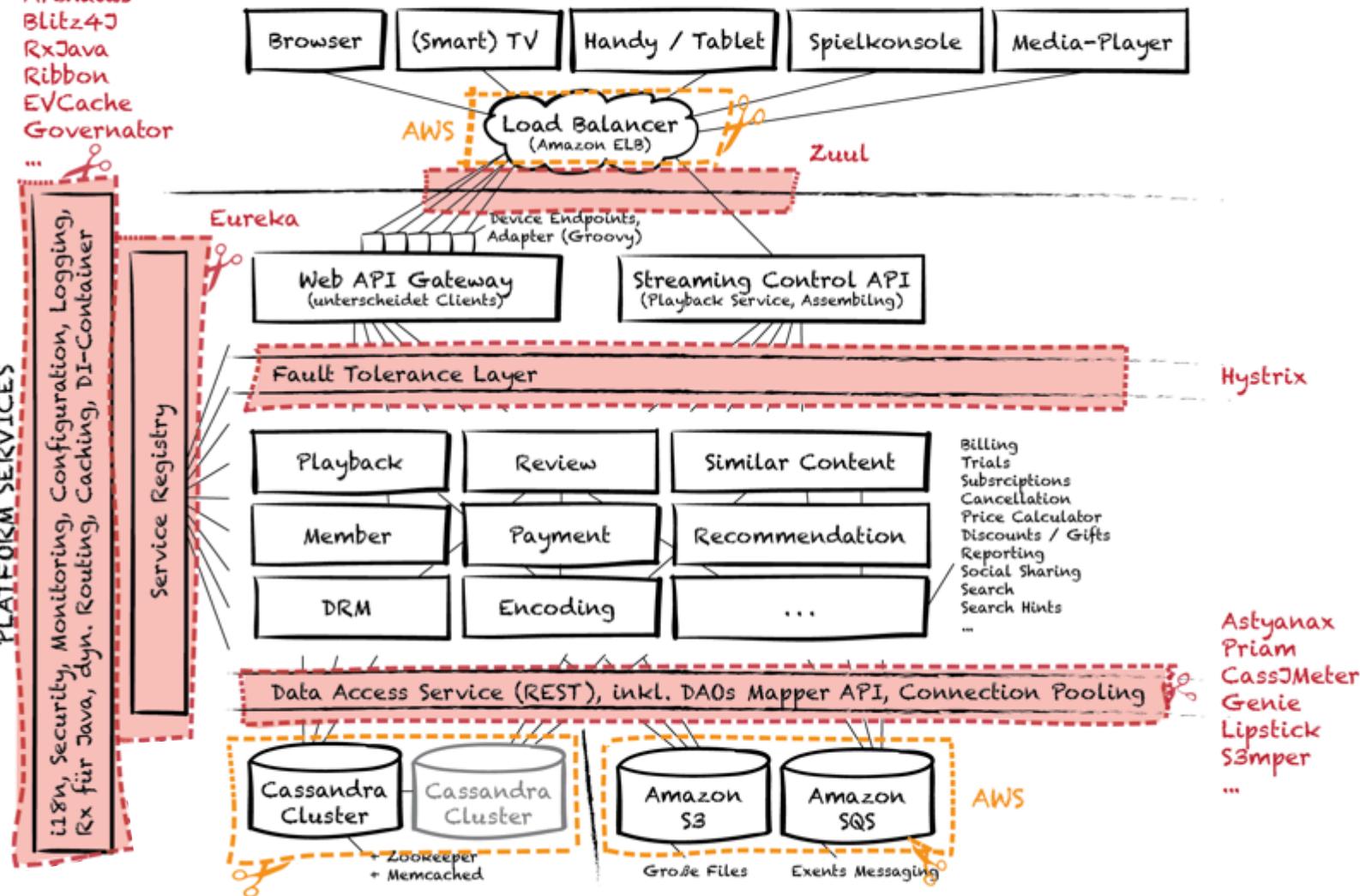


Netflix Open Source Services

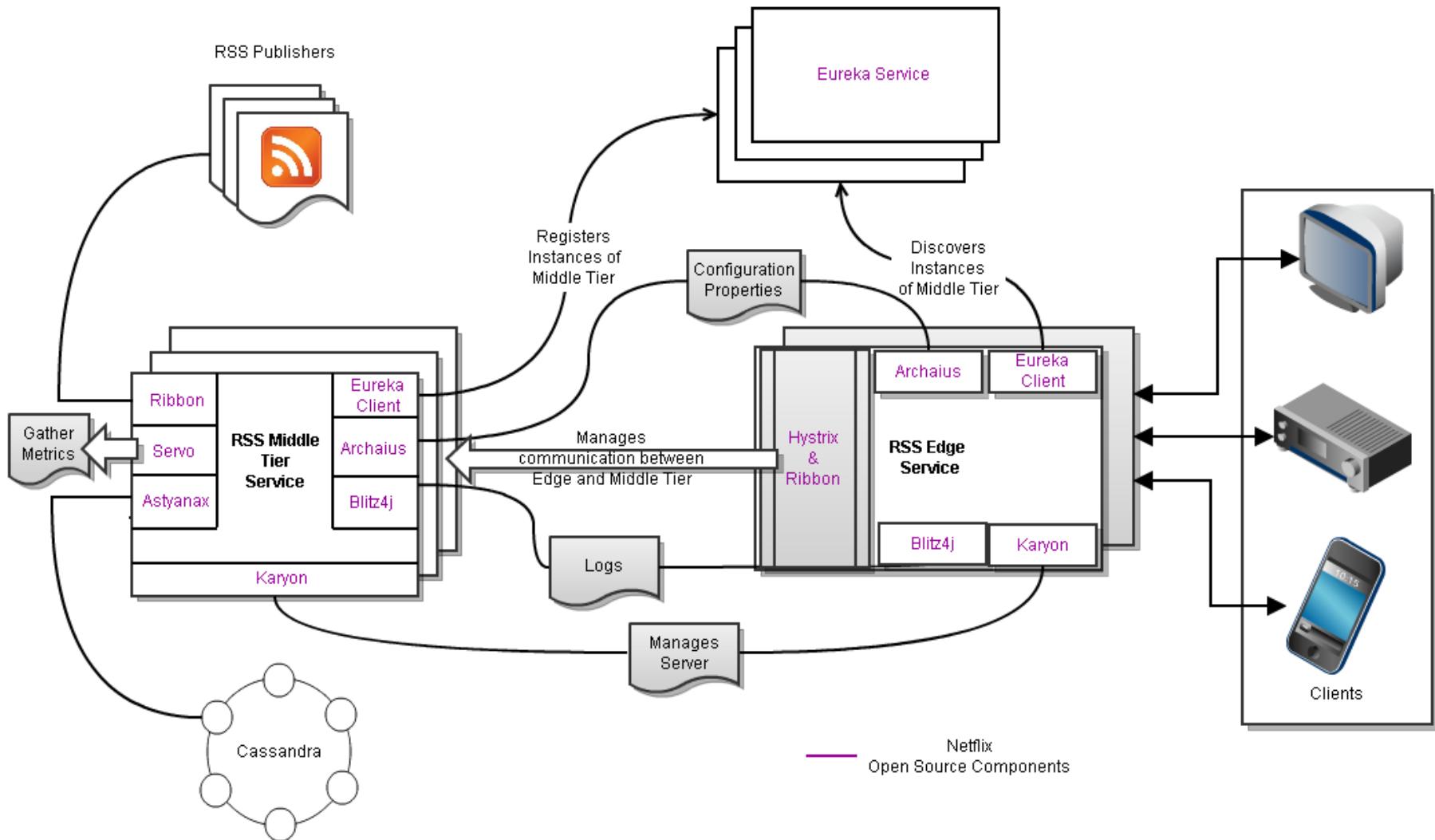


Netflix OSS eingeordnet ...

Zuul
Servo
Archaius
Blitz4J
RxJava
Ribbon
EVCache
Governator
...



Beispielanwendung (2 Services)



Konsequenzen für Netflix

- Anforderungen an Entwickler sinken
- Schnelle Time-to-Market für neue Features
- Architektur-Prinzipien werden unterstützt (Serviceaufbau, Kommunikation, Verwendung von 3rd Party etc.)
- „Open-sourcing“ von Projekten fördert:
 - Ein Mindestmaß an Dokumentation (Wiki, Techblog)
 - Sauberes Design
 - Bild als interessanter Arbeitgeber



Herausforderungen

- Netflix-Spezifika in Entwicklung recht prominent
- Neues Projekt auf dieser Basis hat Overhead

Lösung im Detail (3)



Netflix Deployment



Deployment bei Netflix

Antwort auf **Koordinationsproblem** bei Deployments?

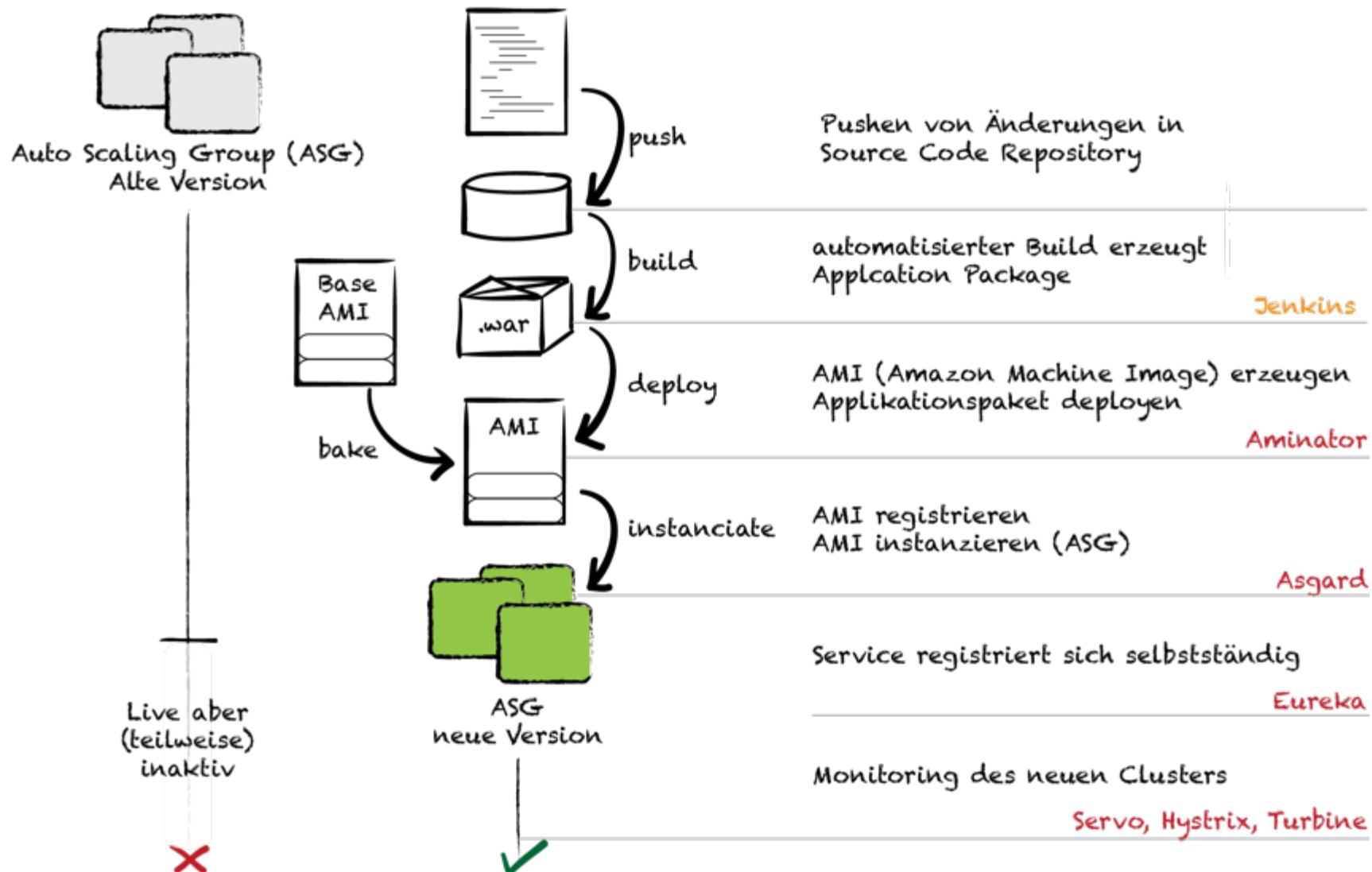
Antwort auf **Komplexität** bei Teamabhängigkeiten?

Unterstützte Anarchie

- **ca. 100** Deployments pro Tag
- Teams arbeiten selbstverantwortlich und unabhängig
- **Keine** Qualitätssicherungsabteilung
- **Keine** Release Engineers
- **Keine** Gesamtkoordination von Releases / Deployments



Komplette Automatisierung ...



Konsequenzen für Netflix

- Schnelles Rollback (bzw. Fallback)
- Billige Tests in Produktionsumgebung statt teurer Testumgebung (die schwer realistisch zu gestalten ist)
- Toolkette abstrahiert und entkoppelt Entwickler von zentralen Einstellungen und Konfigurationen

Herausforderungen



- Mehrfache Hardware erforderlich
- Koordinationsprobleme auf First-come-first-serve heruntergebrochen
- Hoher Grad an Werkzeugunterstützung und Automatisierung notwendig
- Hohe Anforderungen an Logging und Monitoring

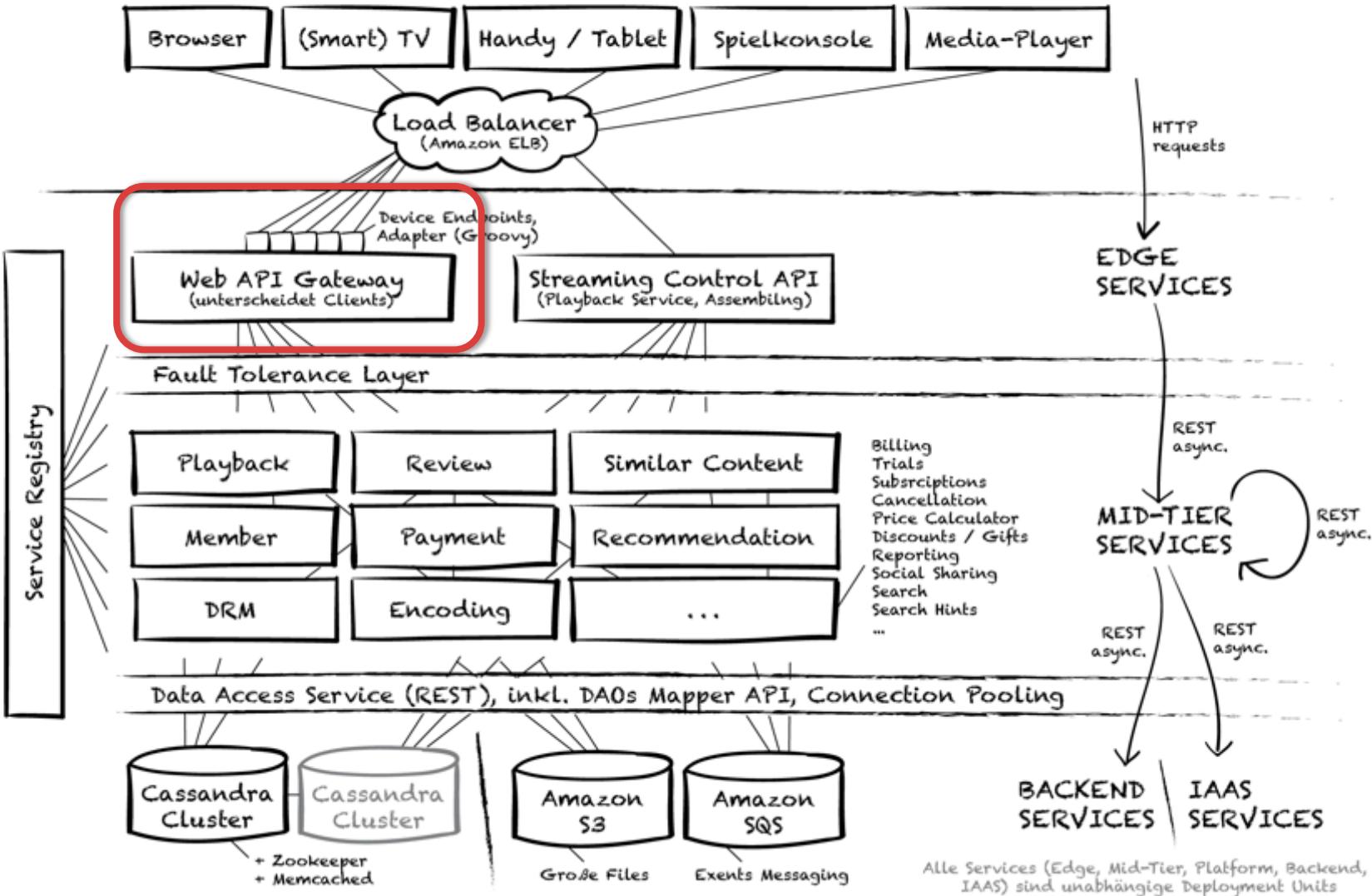
Lösung im Detail (4)



Netflix API eingeordnet ...

PLATFORM SERVICES

i18n, Security, Monitoring, Configuration, Logging, Rx für Java, dyn. Routing, Caching, DI-Container



Wie nutzen / kombinieren Clients die Services?

Client



...



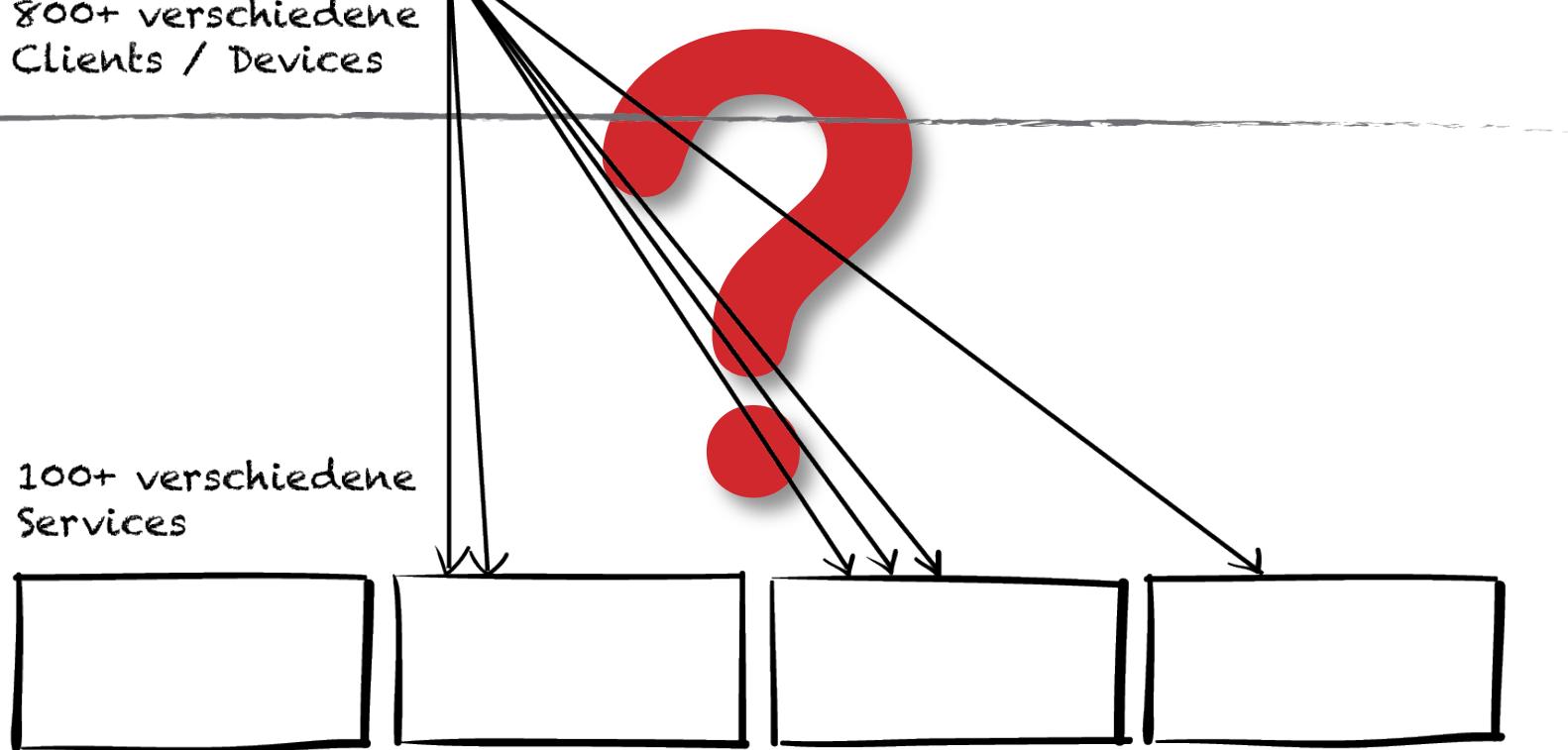
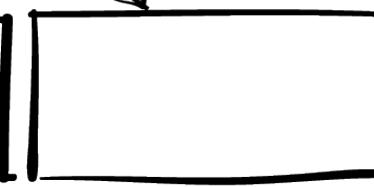
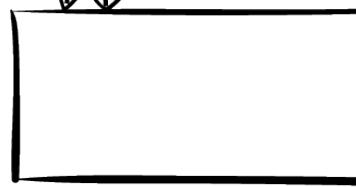
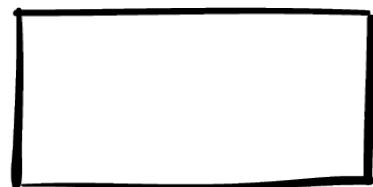
...

800+ verschiedene
Clients / Devices

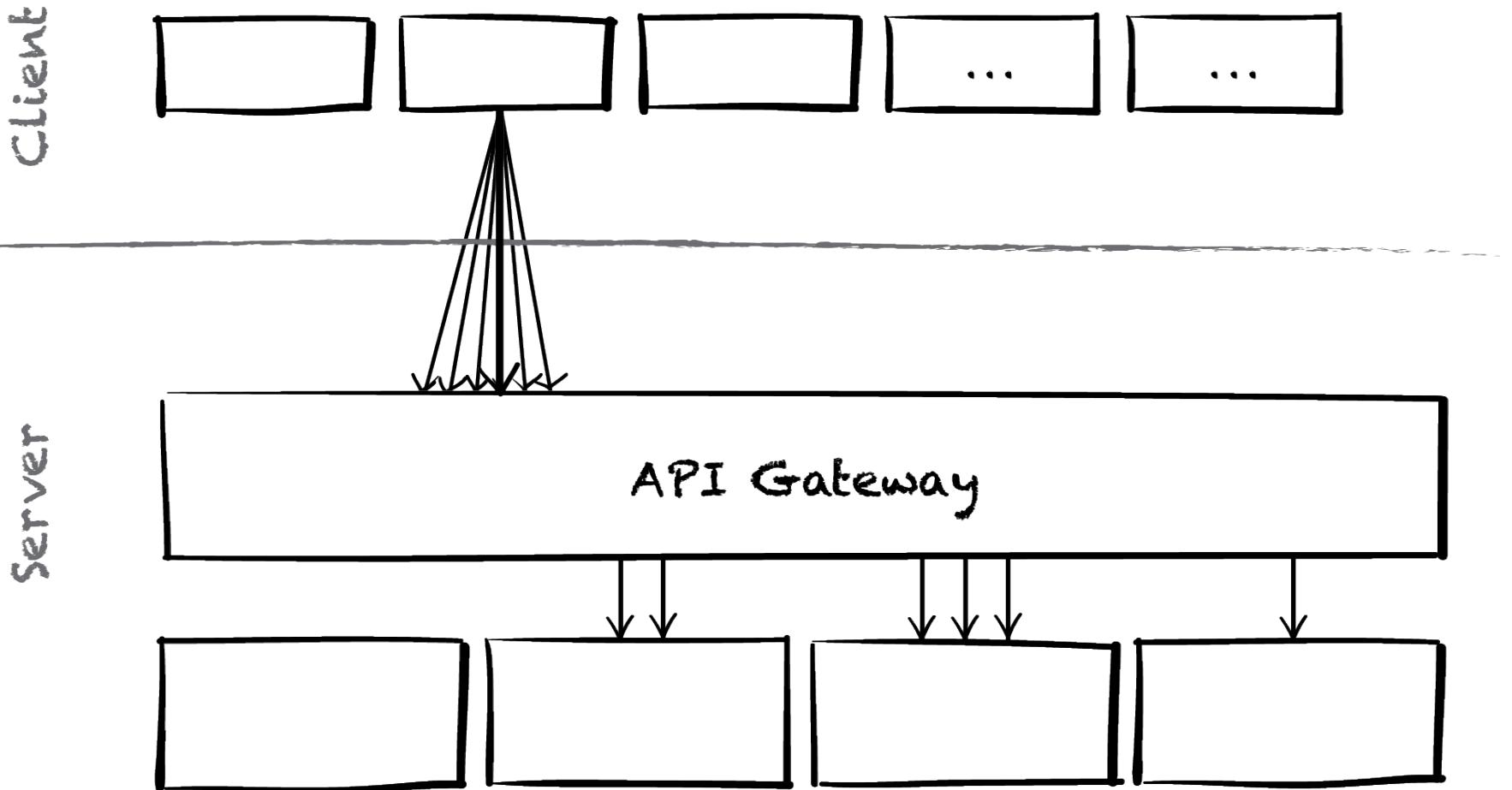
Server



100+ verschiedene
Services

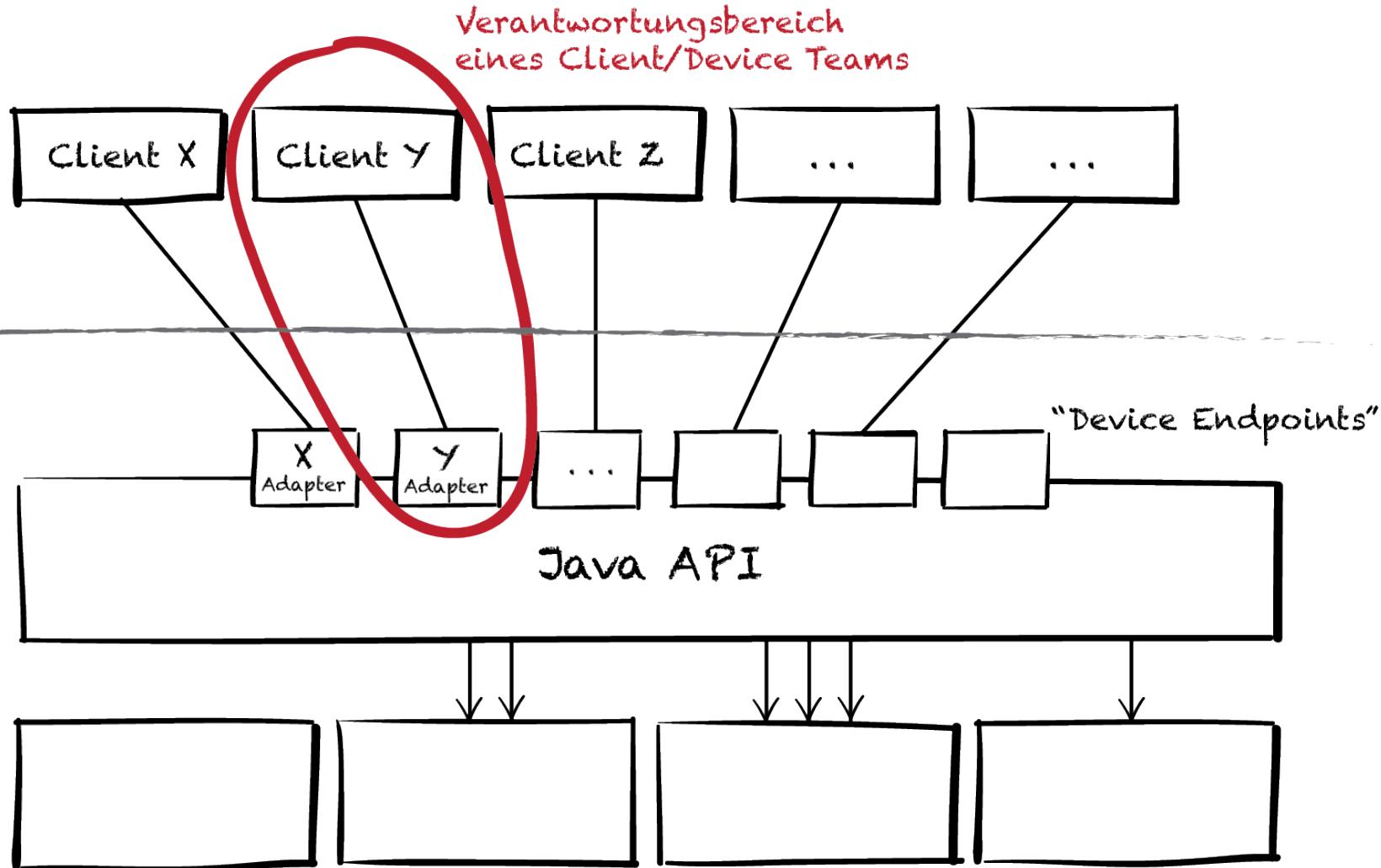


API Gateway („Remote Facade“)



Netflix API

Client



Konsequenzen für Netflix

- Exzellente UI Experience / Performance pro Device
- Kurze Abstimmungswege zwischen Client- und Serverseitigen Entwicklern
- Framework unterstützt schnelle Entwicklung von Client-spezifischen APIs



Herausforderungen

- Die UI-Teams müssen auch Server-code schreiben und verantworten
- Mit vielen Geräten entstehen immer mehr Client-spezifische APIs

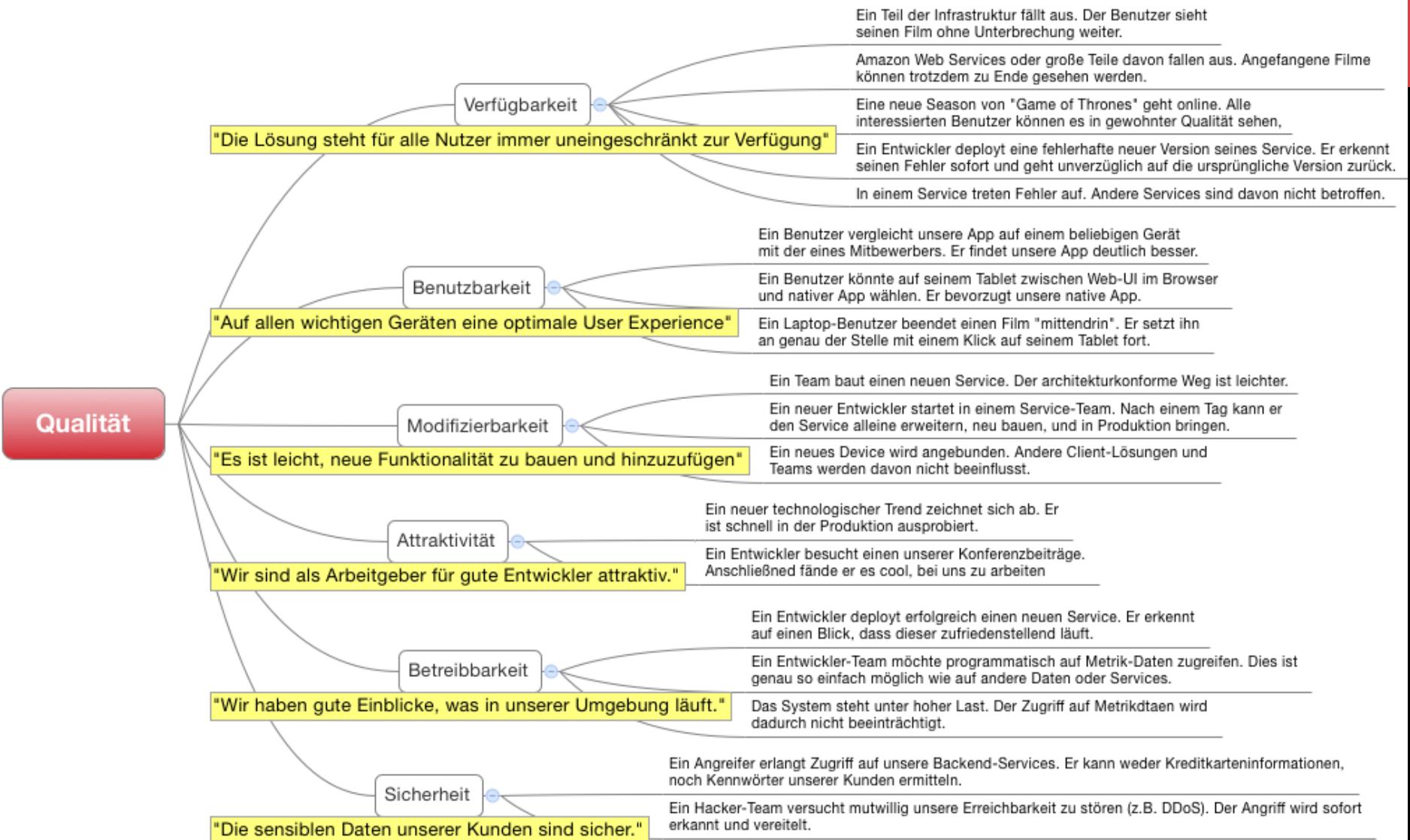
Agenda

4

- 1 Einstieg**
- 2 Was ist Netflix?**
- 3 Lösung im Detail**
- 4 Ergebnisse zusammengefasst**
- 5 Weitere Informationen**



Qualitätsanforderungen



Zentrale Randbedingungen

Welche Rahmenbedingungen bräuchte man aus unserer Sicht für die Netflix-Lösung?



1. (Weiter-)Entwicklung eines **langlebigen Produktes**



2. Größe des Produktes rechtfertigt **mehrere Teams**



3. Teams können **selbstbestimmt** arbeiten



4. Deployment in **Cloud** machbar

5. Scheitern im Deployment möglich

6. Einsatz von **Open Source**-Lösungen OK

Das hier!

Zusammenfassende Kompromissaussagen

ist wichtiger als das hier...

NETFLIX

Technologieentscheidungen auf
Teamebene und **Experimente** zur besseren
Erreichung qualitativer Ziele...

...sind einer **homogenen**
Systemlandschaft mit hoher Integrität
vorzuziehen.

NETFLIX



Innovationskraft und **Wachstum** sind sehr wichtige Aspekte der Softwareentwicklung.

Kontrolle, Planungsfähigkeit und Statusaussagen gegenüber dem Management sind diesen Aspekten klar untergeordnet.



NETFLIX



Die **schnelle** Erstellung und **Auslieferung**
von Funktionalität...



... ist wichtiger als gut getestete Lösungen
und „**Fehlerfreiheit**“ in Produktion.



NETFLIX

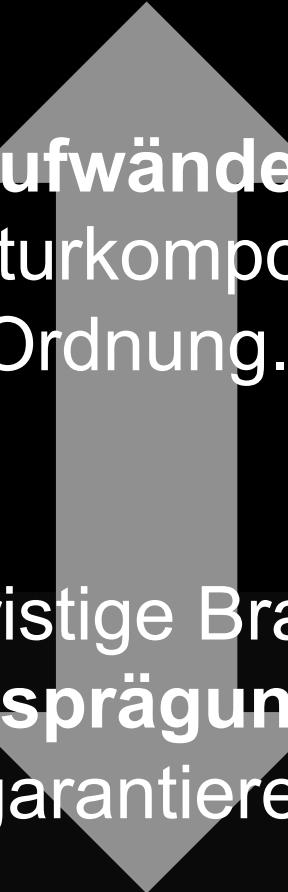


Um **herausragende Qualität** und
entsprechende Marktvorteile zu erzielen...

...sind redundante Entwicklungstätigkeiten
und **geringere Wiederverwendungs-**
möglichkeiten jederzeit in Kauf zu nehmen.



NETFLIX



Höhere initiale **Aufwände** für Framework- und Infrastrukturkomponenten sind in Ordnung...

...um die langfristige Brauchbarkeit und **zeitgemäße Ausprägung** der Lösung zu garantieren



NETFLIX

Agenda

5

- 1 Einstieg**
- 2 Was ist Netflix?**
- 3 Lösung im Detail**
- 4 Ergebnisse zusammengefasst**
- 5 Weitere Informationen**



Netflix Tech Blog

→ <http://techblog.netflix.com>

The screenshot shows a web browser window for techblog.netflix.com. The page features the Netflix logo at the top left and the title "The Netflix Tech Blog" at the top right. A sidebar on the right contains a "Links" section with various Netflix regional blogs and other links. The main content area displays a blog post from Wednesday, April 8, 2015, titled "Introducing Vector: Netflix's On-Host Performance Monitoring Tool" by Martin Spier, Amer Ather, and Brendan Gregg. The post includes a large "Vector" logo with an owl icon and a detailed description of the tool.

Wednesday, April 8, 2015

Introducing Vector: Netflix's On-Host Performance Monitoring Tool

by Martin Spier, Amer Ather and Brendan Gregg



Vector is an open source host-level performance monitoring framework, which exposes hand-picked, high-resolution system and application metrics to every engineer's browser. Having the right metrics available on demand and at a high resolution is key to understanding how a system behaves and correctly troubleshooting performance issues. Previously, we'd login to instances as needed, run a variety of commands, and sift through the output for the metrics that matter. Vector cuts down the time to get to those metrics, helping us respond to incidents more quickly.

Vector provides a simple way for users to visualize and analyze system and application-level metrics in near real-time. It leverages the battle tested open source system monitoring framework, Performance Co-Pilot (PCP), layering on top a flexible and user-friendly UI. The UI polls metrics at up to 1 second resolution, rendering the data in completely configurable dashboards that simplify cross-metric correlation and analysis.

Links

- [Netflix US & Canada Blog](#)
- [Netflix America Latina Blog](#)
- [Netflix Brasil Blog](#)
- [Netflix Benelux Blog](#)
- [Netflix DACH Blog](#)
- [Netflix France Blog](#)
- [Netflix Nordics Blog](#)
- [Netflix UK & Ireland Blog](#)
- [Netflix ISP Speed Index](#)
- [Open positions at Netflix](#)
- [Netflix Website](#)
- [Facebook Netflix Page](#)
- [Netflix UI Engineering](#)

[RSS Feed](#)



Software Engineering Radio #216

→ <http://www.se-radio.net/2014/12/episode-216-adrian-cockcroft-on-the-modern-cloud-based-platform/>

The screenshot shows a web browser displaying the Software Engineering Radio website at se-radio.net. The header features the SE-Radio logo, the text "Software Engineering Radio", and the subtitle "The Podcast for Professional Software Developers". Below the header is a navigation bar with links for Home, About SE-Radio, Be A Host, Praise, Team, Contact, and Legal. The main content area displays the title "Episode 216: Adrian Cockcroft on the Modern Cloud-based Platform". Below the title is a subtext indicating it was filed in "Episodes" by "SE-Radio" on December 10, 2014, with 7 comments. To the right, there's a "Connect" section with social media icons for Facebook, Twitter, Google+, LinkedIn, and RSS. At the bottom right of the main content area, there's a box for "IEEE Computer Society" featuring their logo and a snippet about "IEEE Software" magazine.

Episode 216: Adrian Cockcroft on the Modern Cloud-based Platform

Filed in [Episodes](#) by [SE-Radio](#) on December 10, 2014 • 7 Comments



Adrian Cockcroft discusses the challenges in creating a dynamic, flexible, cloud-based platform with SE Radio host [Stefan Tilkov](#). After briefly discussing the definition of "cloud computing," Adrian explains the history behind Netflix's move to the cloud (which he led). After highlighting some of the differences that have developers and architects must face, Adrian talks about lessons for other kinds of companies and industries. Microservices and microservices architecture are a hot topic

Connect



IEEE computer society



IEEE Software offers pioneering ideas, expert analyses, and thoughtful insights for software professionals who need to keep up with rapid technology change. It's the

Artikel im Javamagazin 5.15

→ House of Guts –
Keine Angst vor Chaos

Netflix: Resilience
konsequent
zu Ende gedacht

Stefan Toth
Javamagazin 5.15



Netflix Open Source Software

→ <http://netflix.github.io>

The screenshot shows a web browser window for netflix.github.io. The page features the Netflix OSS logo and navigation tabs for 'Repositories' and 'Powered By NetflixOSS'. A sidebar on the left lists 'Availability' for projects like Hystrix, SimianArmy, Turbine, ICE, Asgard, Frigga, and Glisten, each with a small icon. The main content area displays detailed information for the SimianArmy project, including its description, statistics (Stars: 2445, Forks: 298, Language: Java, Open Issues: 26), and last update (02/11/15 @18:32:30). To the right, there's a sidebar with links to mailing lists and a 'Stay in Touch' section with links to the Netflix Tech Blog, @NetflixOSS, Slideshare, Netflix Meetup, and Jobs at Netflix.

NETFLIX | OSS Netflix Open Source Software Center

Repositories Powered By NetflixOSS Availability

HYSTRIX

SIMIANARMY

TURBINE

SimianArmy

Tools for keeping your cloud operating in top form. Chaos Monkey is a resiliency tool that helps applications tolerate random instance failures. [More Info](#)

Stars: 2445
Forks: 298
Language: Java
Open Issues: 26
Updated: 02/11/15 @18:32:30

Cloud Management

ICE

ASGARD

FRIGGA

GLISTEN

of our open source projects. Click on the following links to subscribe and/or maintain your subscriptions.

- Asgard Mailing List
- Astyanax Mailing List
- Curator Mailing List
- Exhibitor Mailing List
- SimianArmy Mailing List

Stay in Touch

- Netflix Tech Blog
- @NetflixOSS
- Slideshare
- Netflix Meetup
- Jobs at Netflix

Vielen Dank.

Ich freue mich auf Eure Fragen!

 stefan.toth@embarc.de

 @st_toth

 xing.to/sto

DOWNLOAD FOLIEN:

<http://www.embarc.de/blog/>