Microservices at Netflix Scale

First Principles, Tradeoffs, Lessons Learned

Ruslan Meshenberg
@rusmeshenberg

NETFLIX

Microservices:
all benefits, no costs?

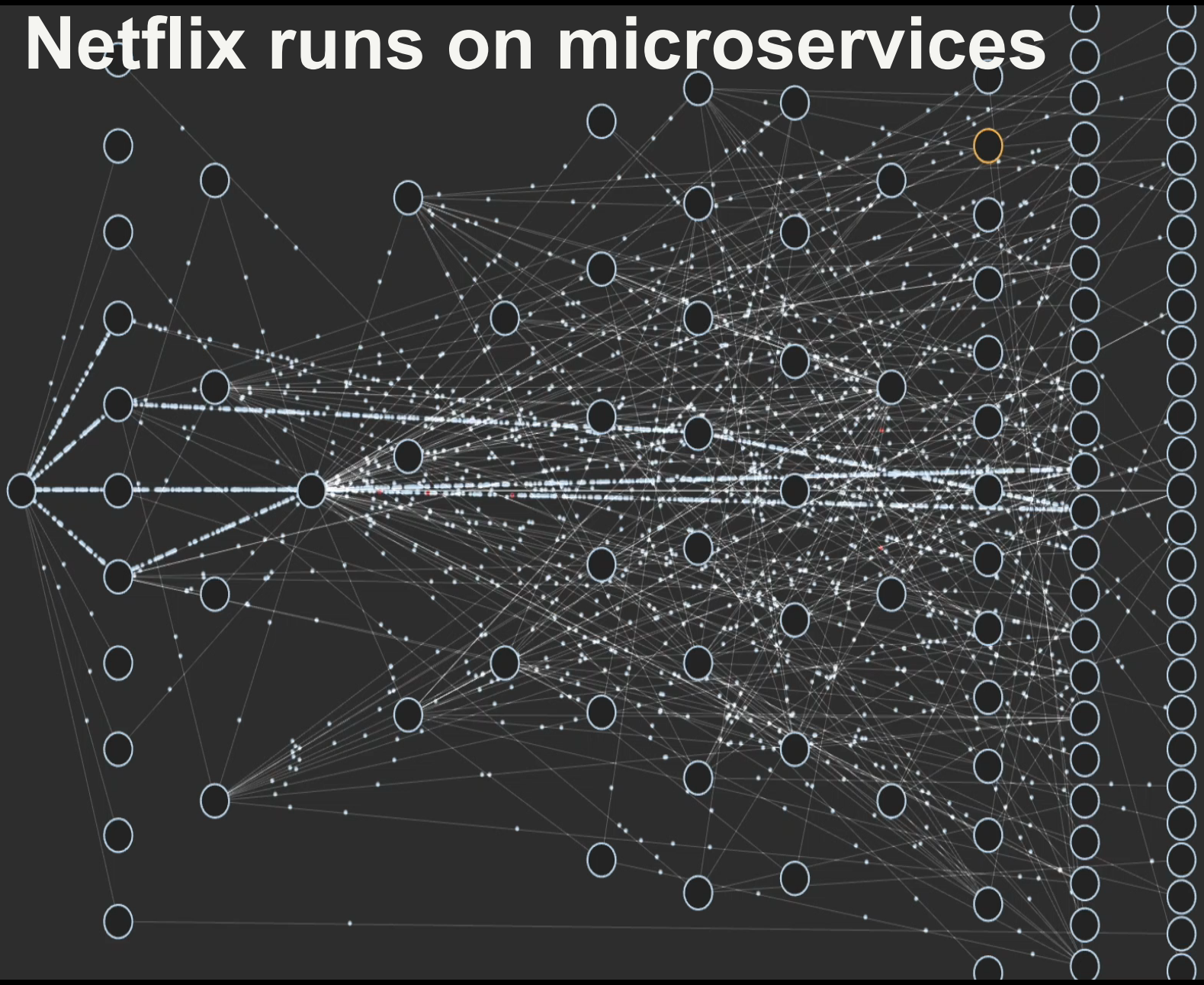
NETFLIX

Netflix is the world's leading Internet television network with over 81 million members in over 190 countries enjoying more than 125 million hours of TV shows and movies per day, including original series, documentaries and feature films.

Ruslan Meshenberg
Director, Platform Engineering

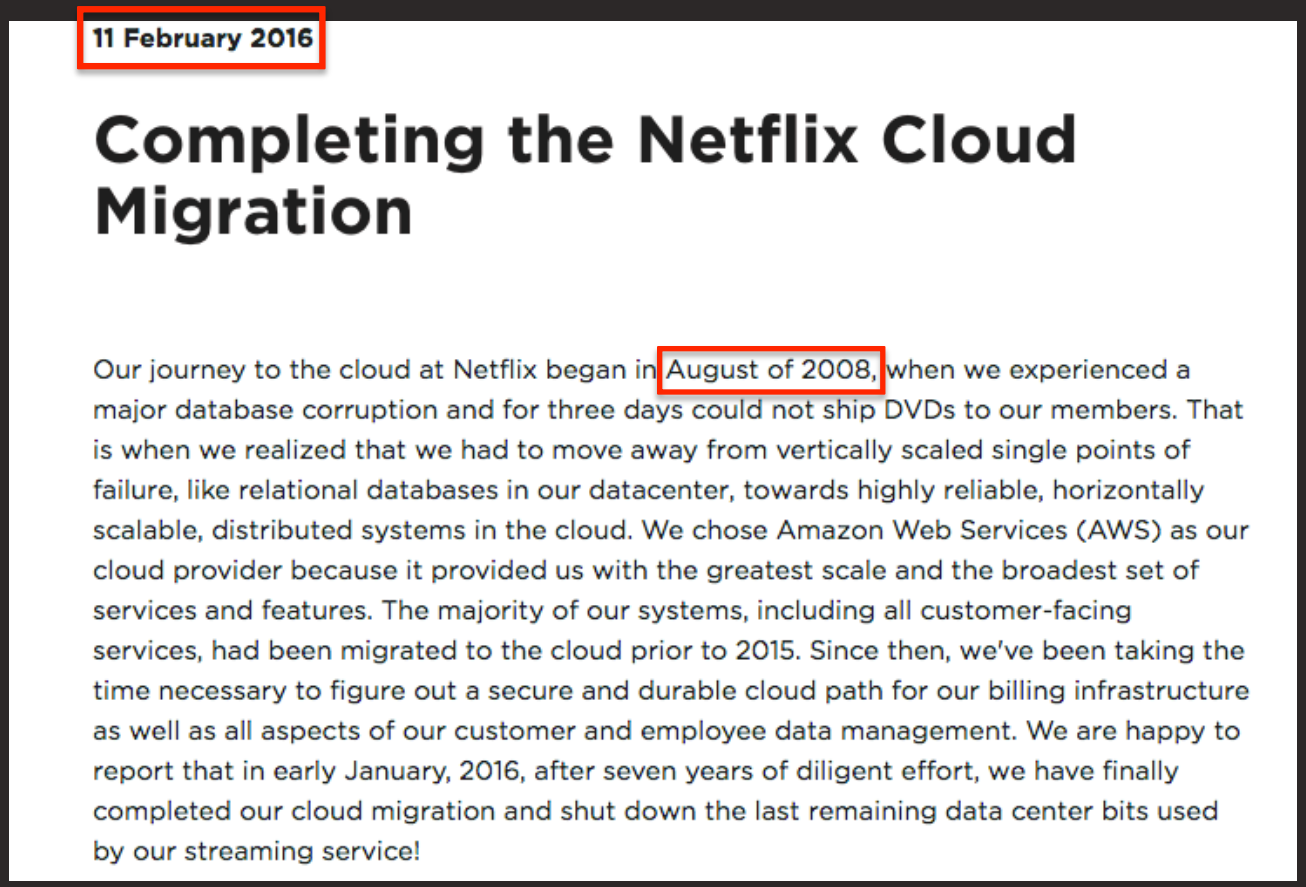
- Runtime Systems
- Container Runtime
- Persistence and Databases
- Real Time Data Infrastructure

Netflix runs on microservices



Netflix journey to microservices

Our journey took 7 years



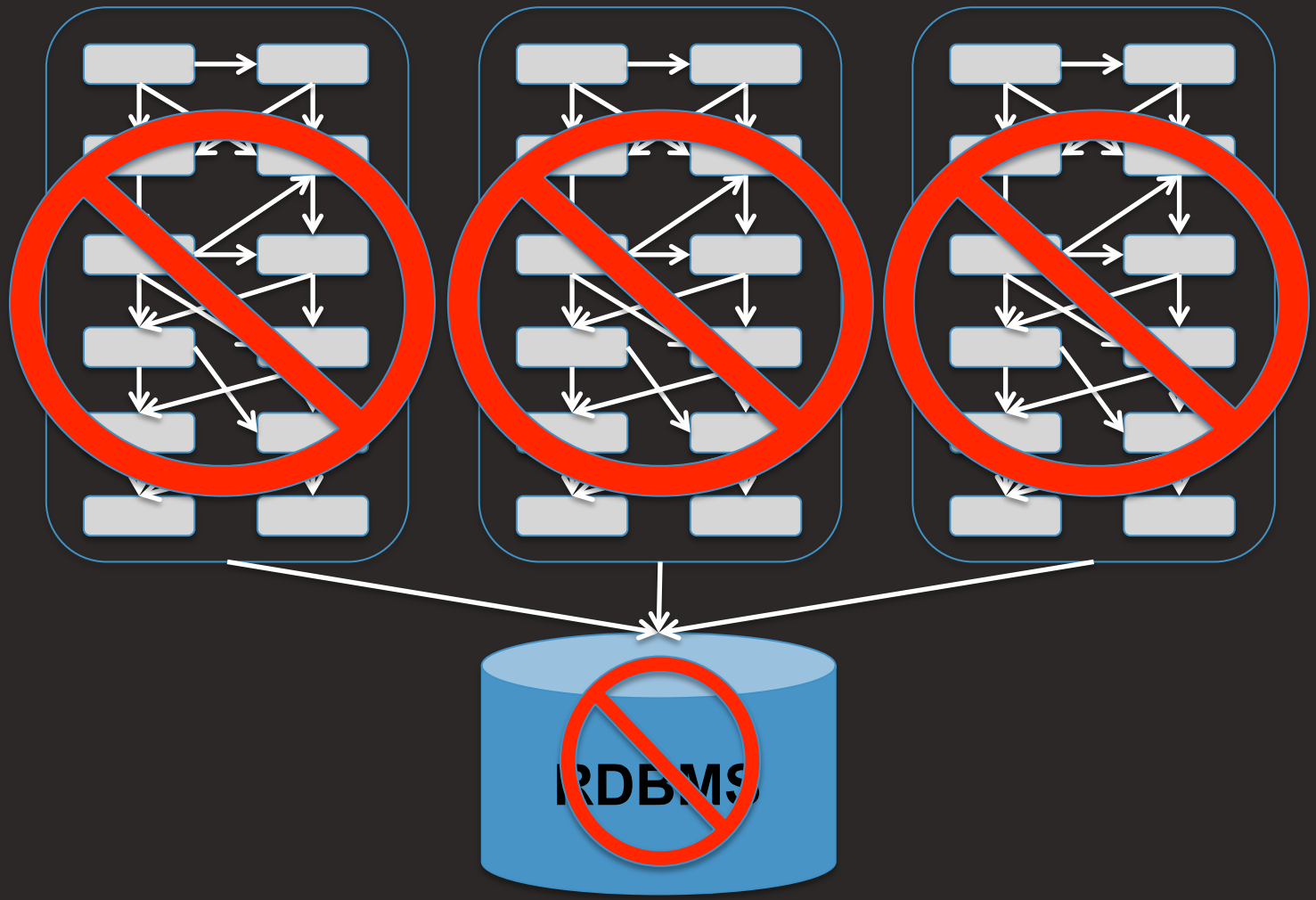
11 February 2016

Completing the Netflix Cloud Migration

Our journey to the cloud at Netflix began in August of 2008, when we experienced a major database corruption and for three days could not ship DVDs to our members. That is when we realized that we had to move away from vertically scaled single points of failure, like relational databases in our datacenter, towards highly reliable, horizontally scalable, distributed systems in the cloud. We chose Amazon Web Services (AWS) as our cloud provider because it provided us with the greatest scale and the broadest set of services and features. The majority of our systems, including all customer-facing services, had been migrated to the cloud prior to 2015. Since then, we've been taking the time necessary to figure out a secure and durable cloud path for our billing infrastructure as well as all aspects of our customer and employee data management. We are happy to report that in early January, 2016, after seven years of diligent effort, we have finally completed our cloud migration and shut down the last remaining data center bits used by our streaming service!

https://media.netflix.com/en/company-blog/completing-the-netflix-cloud-migration

Data Center - Monolith



August 2008

NETFLIX

Your Account | Queue | Help

We're Sorry DVD Shipments Are Delayed

Dear Betsy,

Our shipping system is unexpectedly down. We received a DVD back from you and should have shipped you a DVD, but we likely have not. Our goal is to ship DVDs as soon as possible, and we will keep you posted on the status of your DVD shipments.

We are sorry for any inconvenience this has caused. If your DVD shipment is delayed, we will be issuing a credit to your account in the next few days. You don't need to do anything. The credit will be automatically applied to your next billing statement.

Again, we apologize for the delay and thank you for your understanding. If you need further assistance, please call us at 1-888-638-3549.

-The Netflix Team

First Principles

Buy vs. Build

- Use or contribute to OSS technologies first
- Only build what you have to

Services should be stateless*

- Must not rely on sticky sessions
- Prove by Chaos testing

*Except the Persistence / Caching layers

Scale out vs. scale up

- If you keep scaling up, you'll hit a limit
- Horizontal scaling gives you a longer runway

Redundancy and Isolation For Resiliency

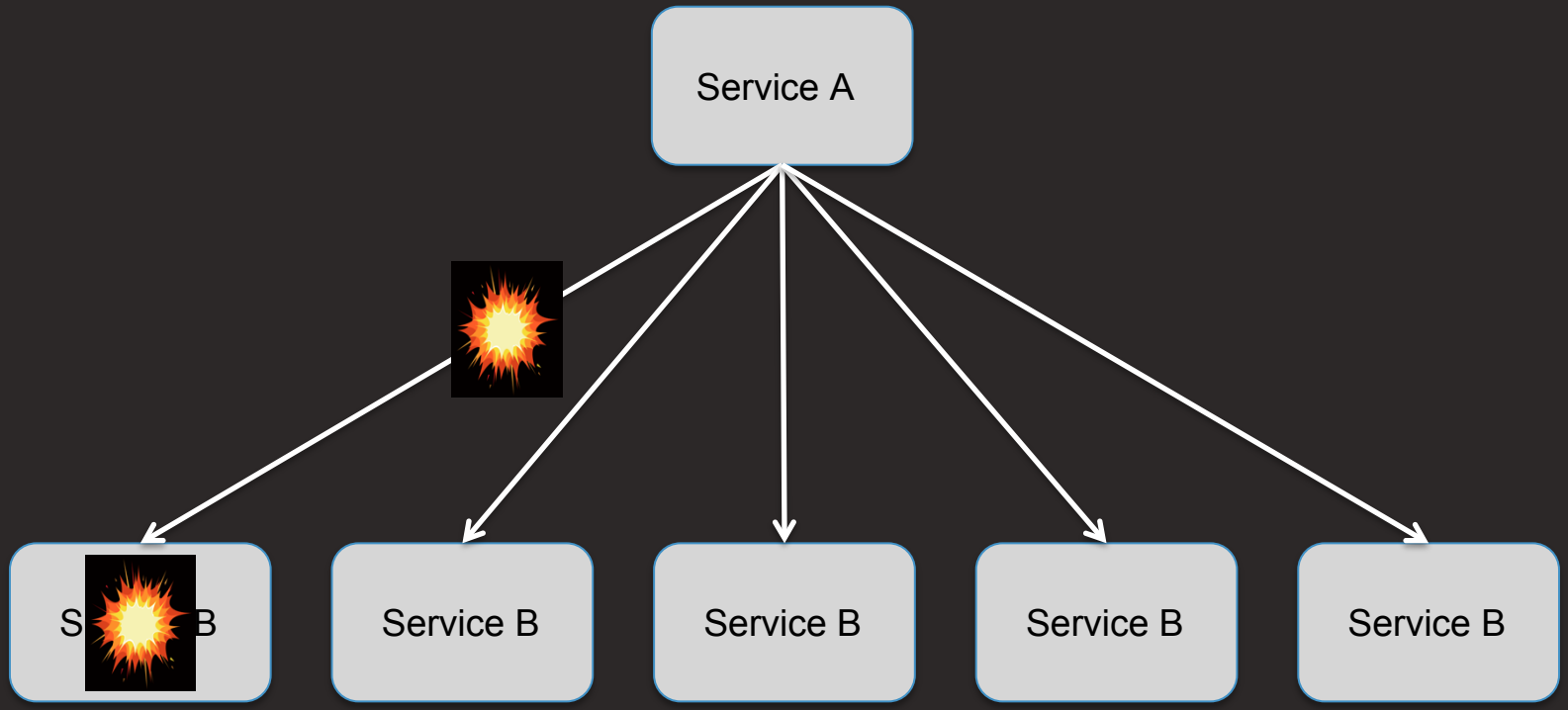
- Make more than one of anything
- Isolate the blast radius for any given failure

Automate destructive testing

- Simian Army
- Started with Chaos Monkey

First Principles In Action

Stateless services



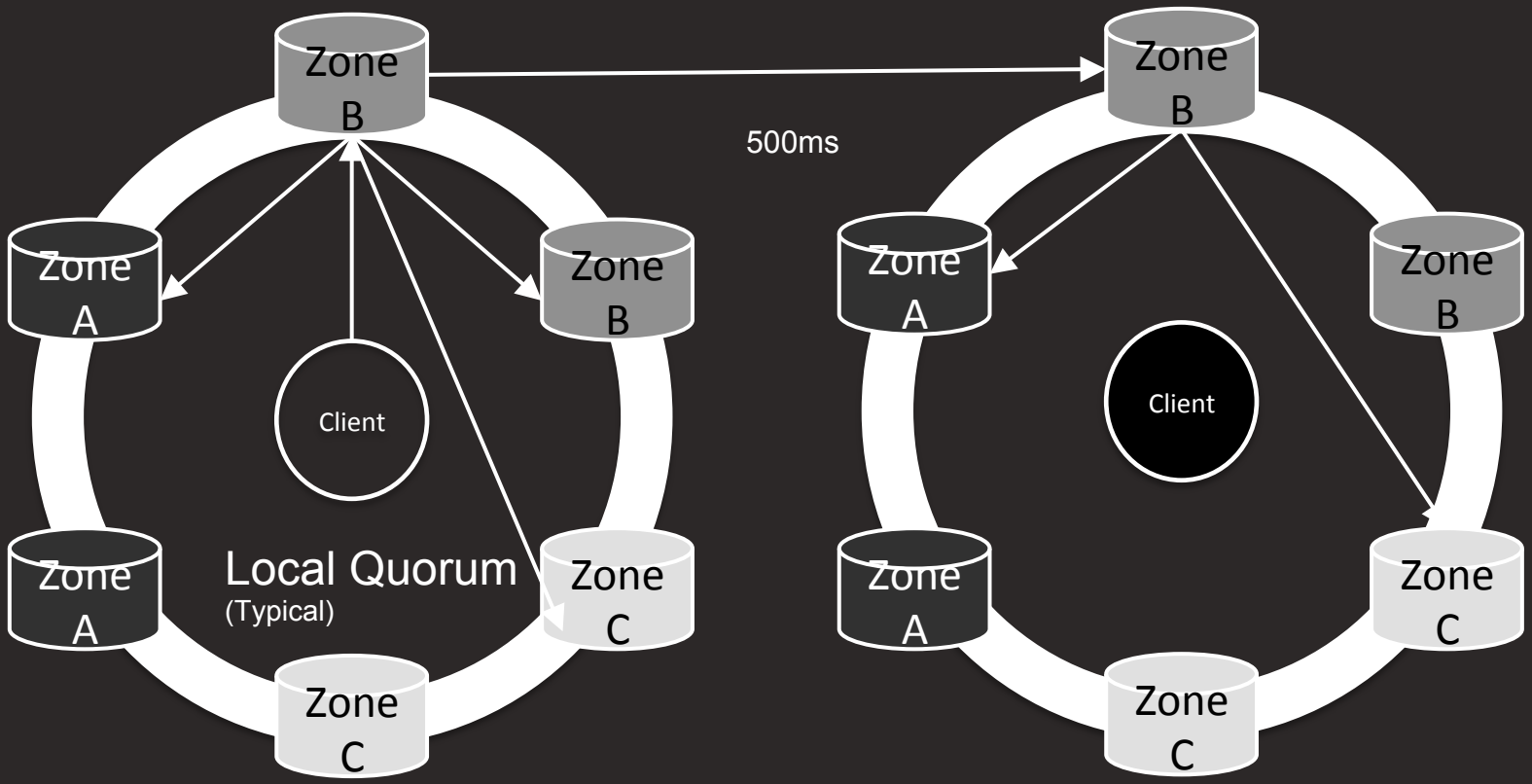
Verify stateless

Data – from RDBMS to Cassandra

- NoSQL at scale
- Open Source
- Multi-Regional
- Multi-directional

- Available
- Partition Tolerance
- Tunable Consistency*

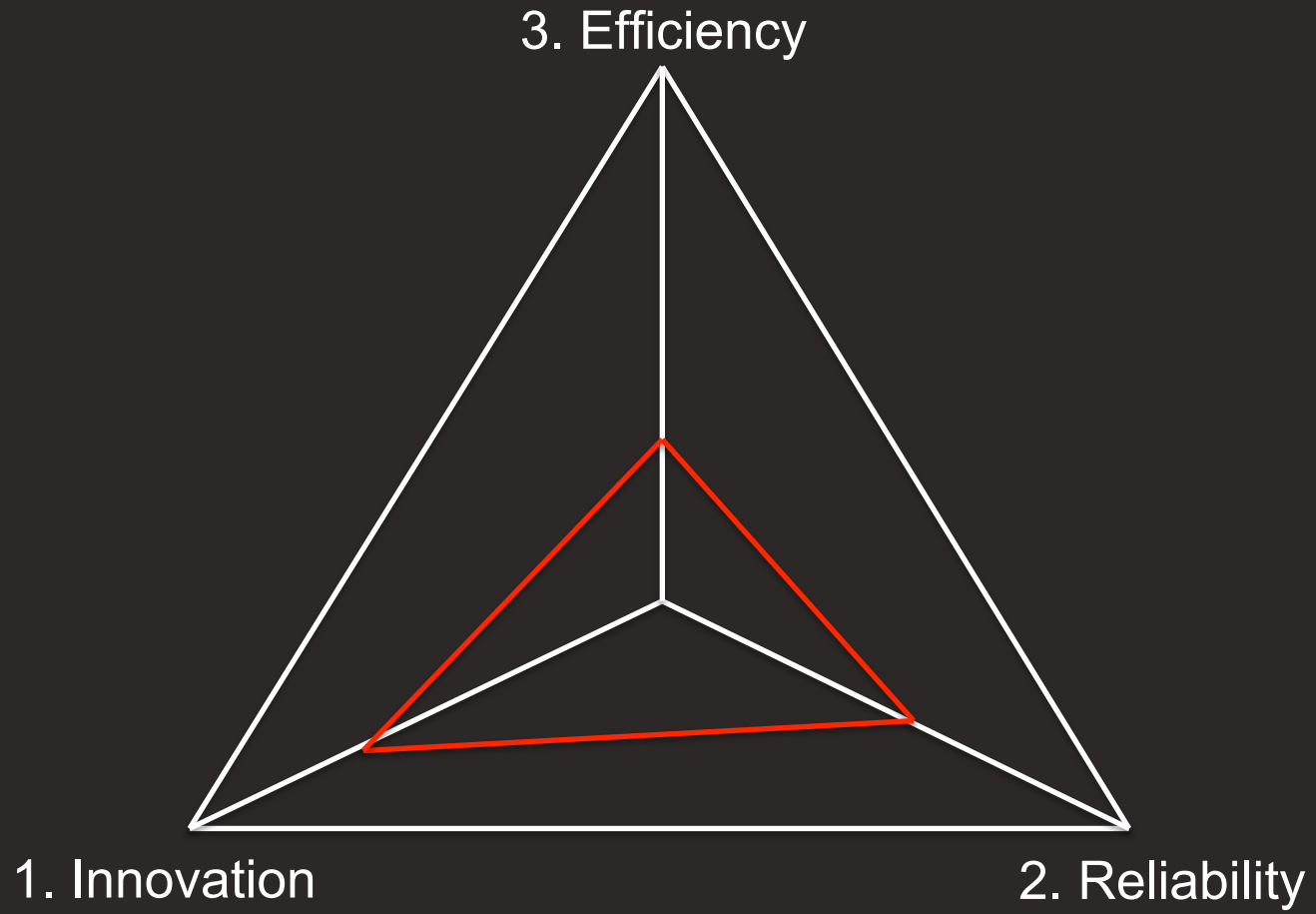
Multi-Regional Replication



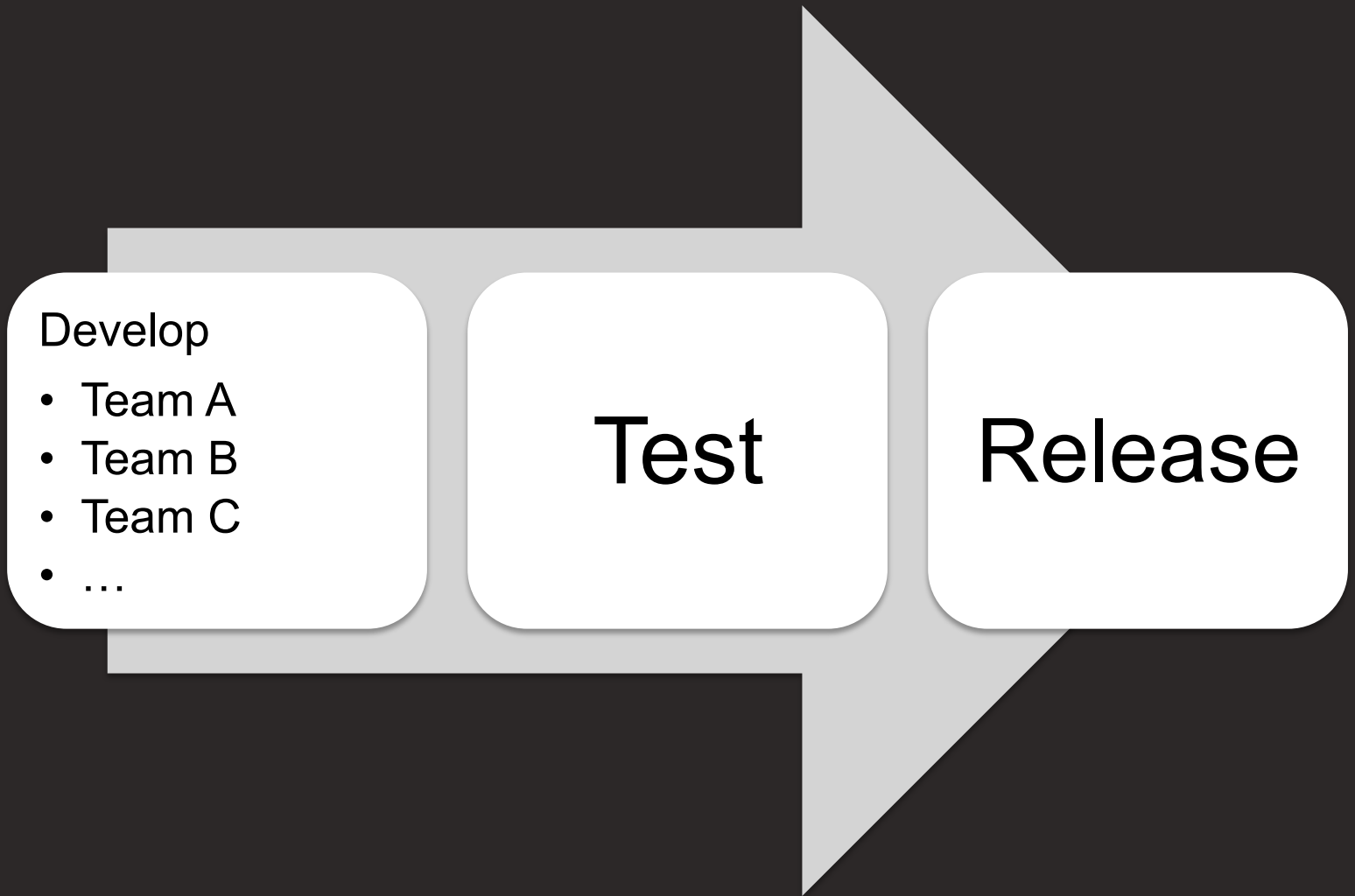
Last, but not least - Billing

Microservices – Benefits

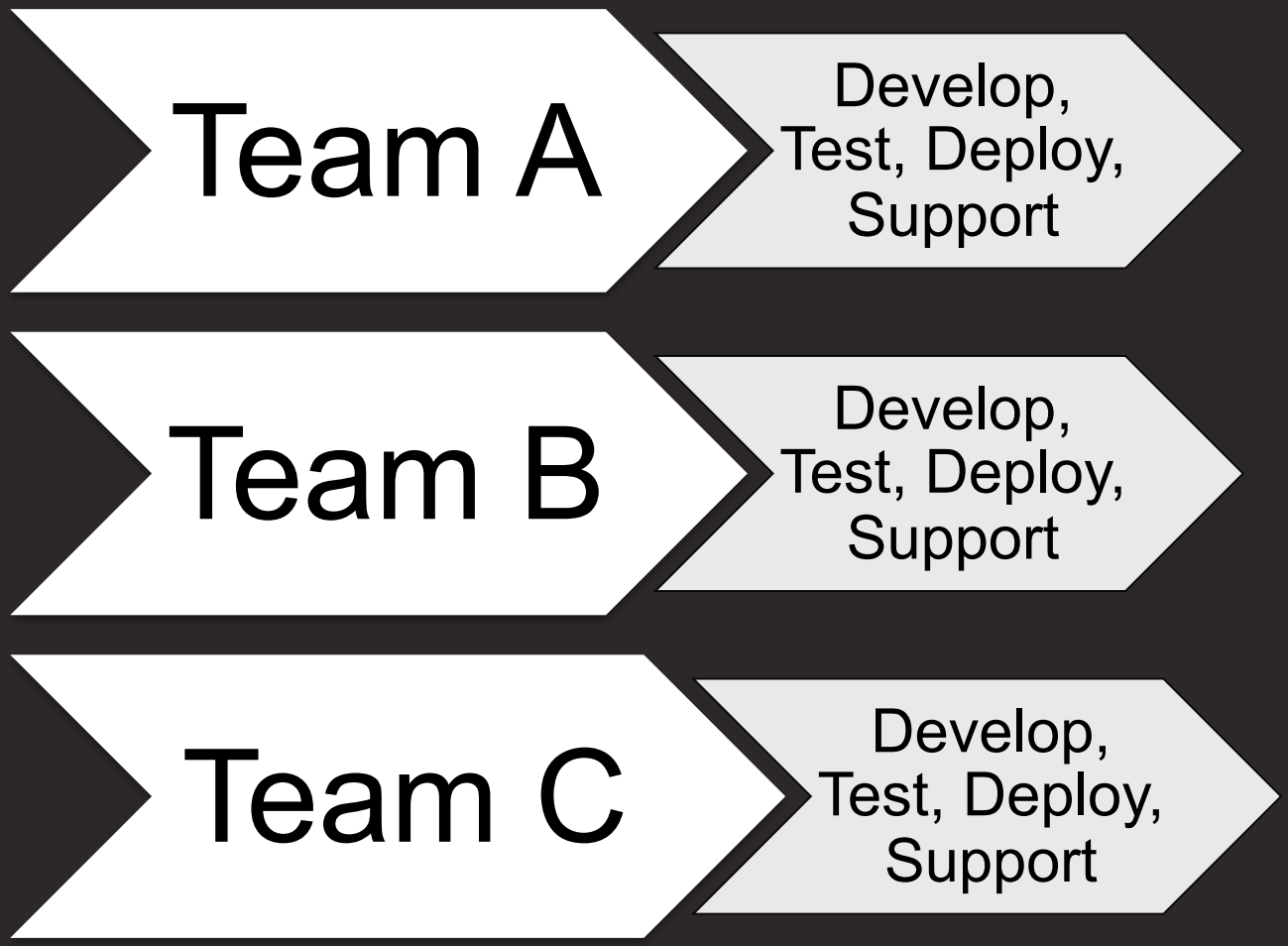
Our Priorities

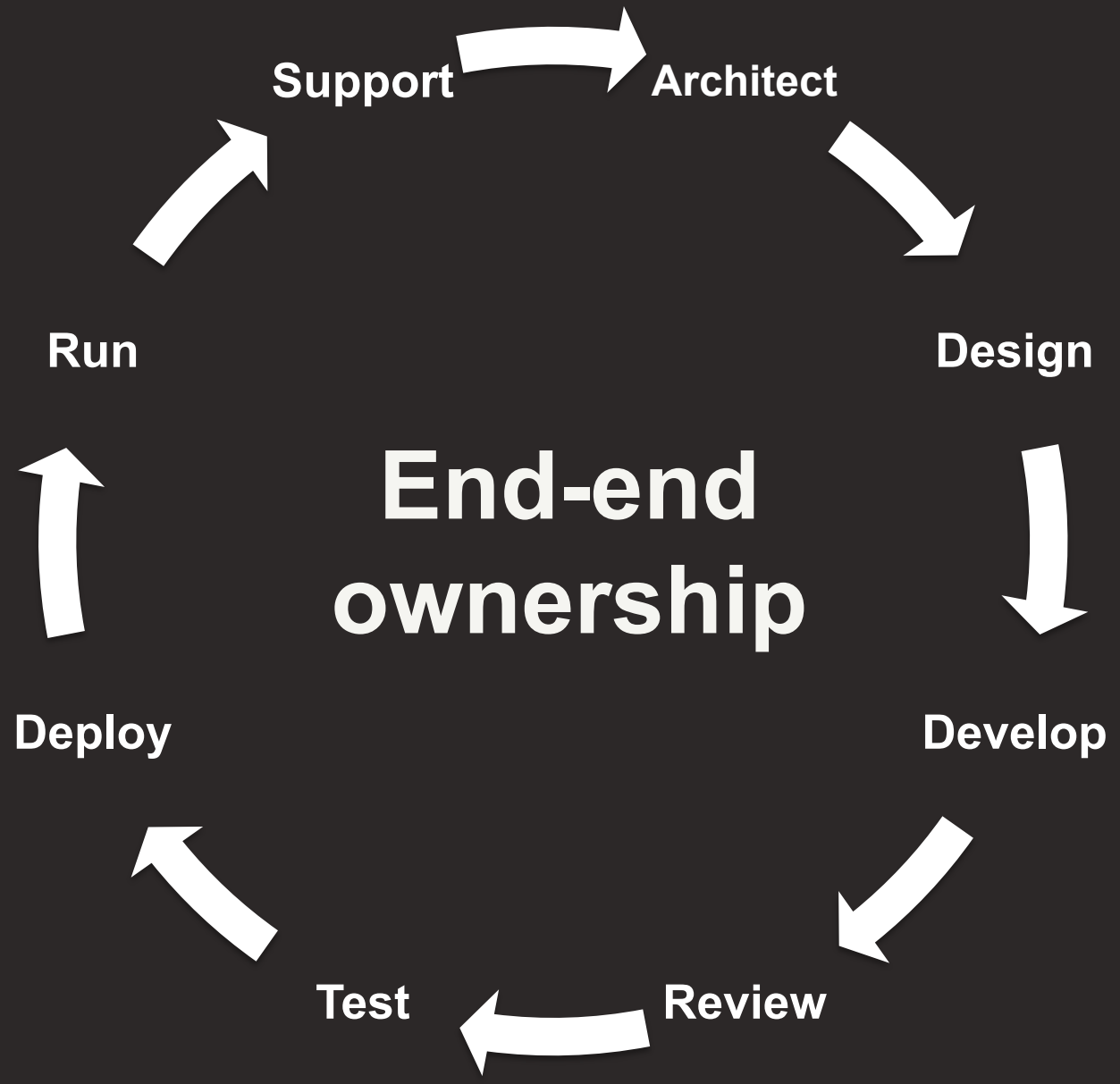


Innovation:
tight coupling doesn't work

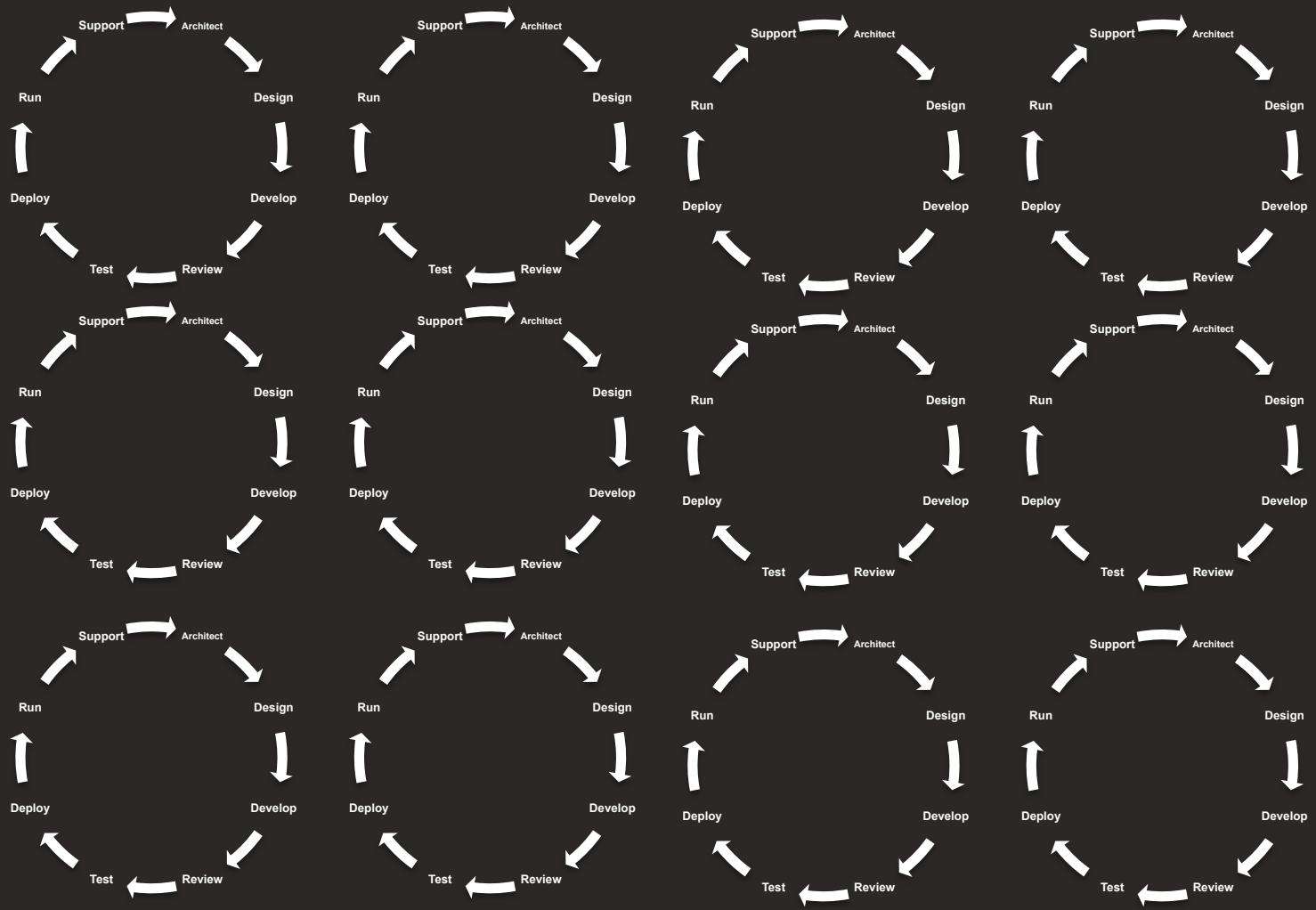


Innovation: Loose coupling

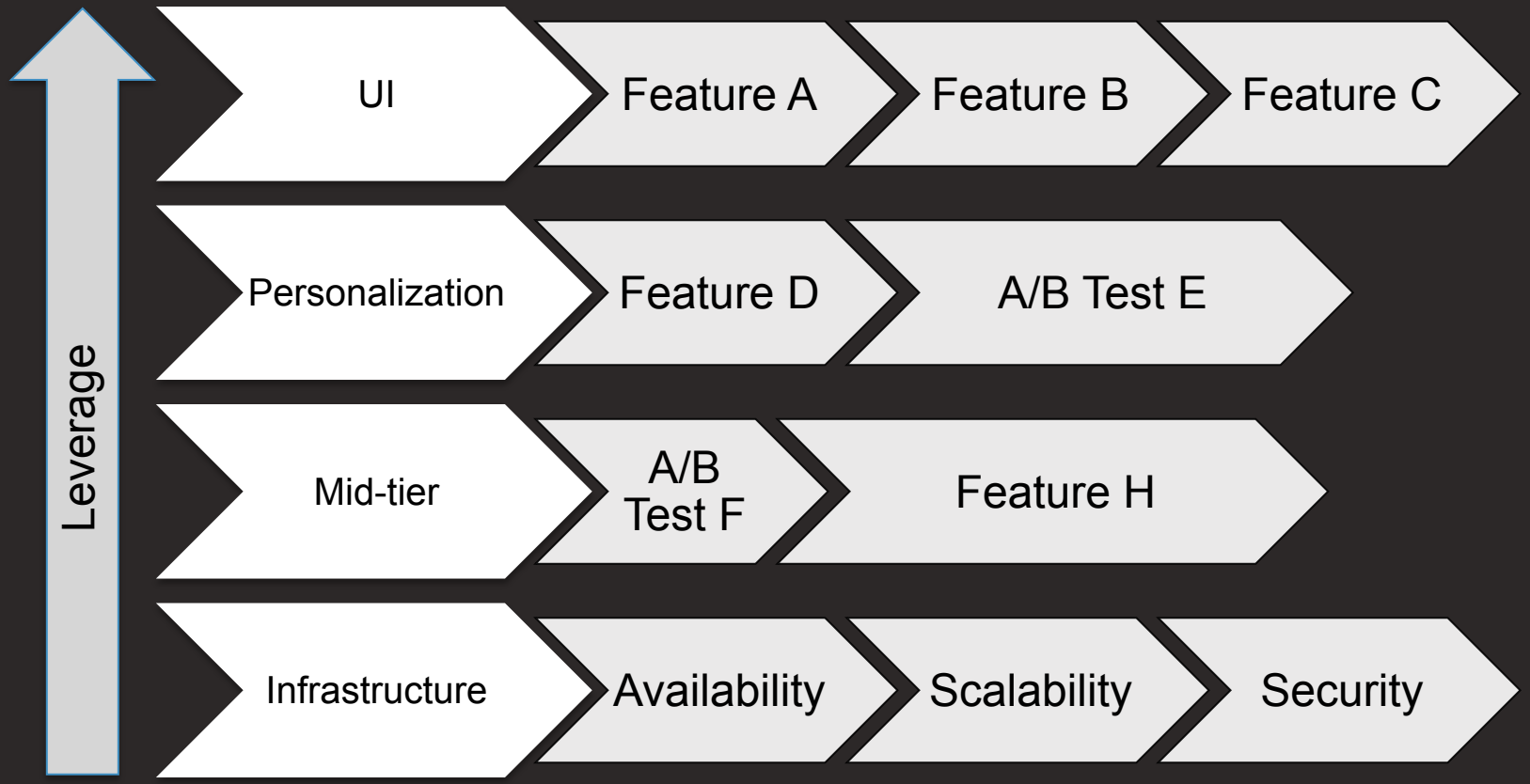



Support
Architect
Design
Develop
Review
Test
Deploy
Run
End-end ownership

End-end ownership + velocity



Separation of concerns



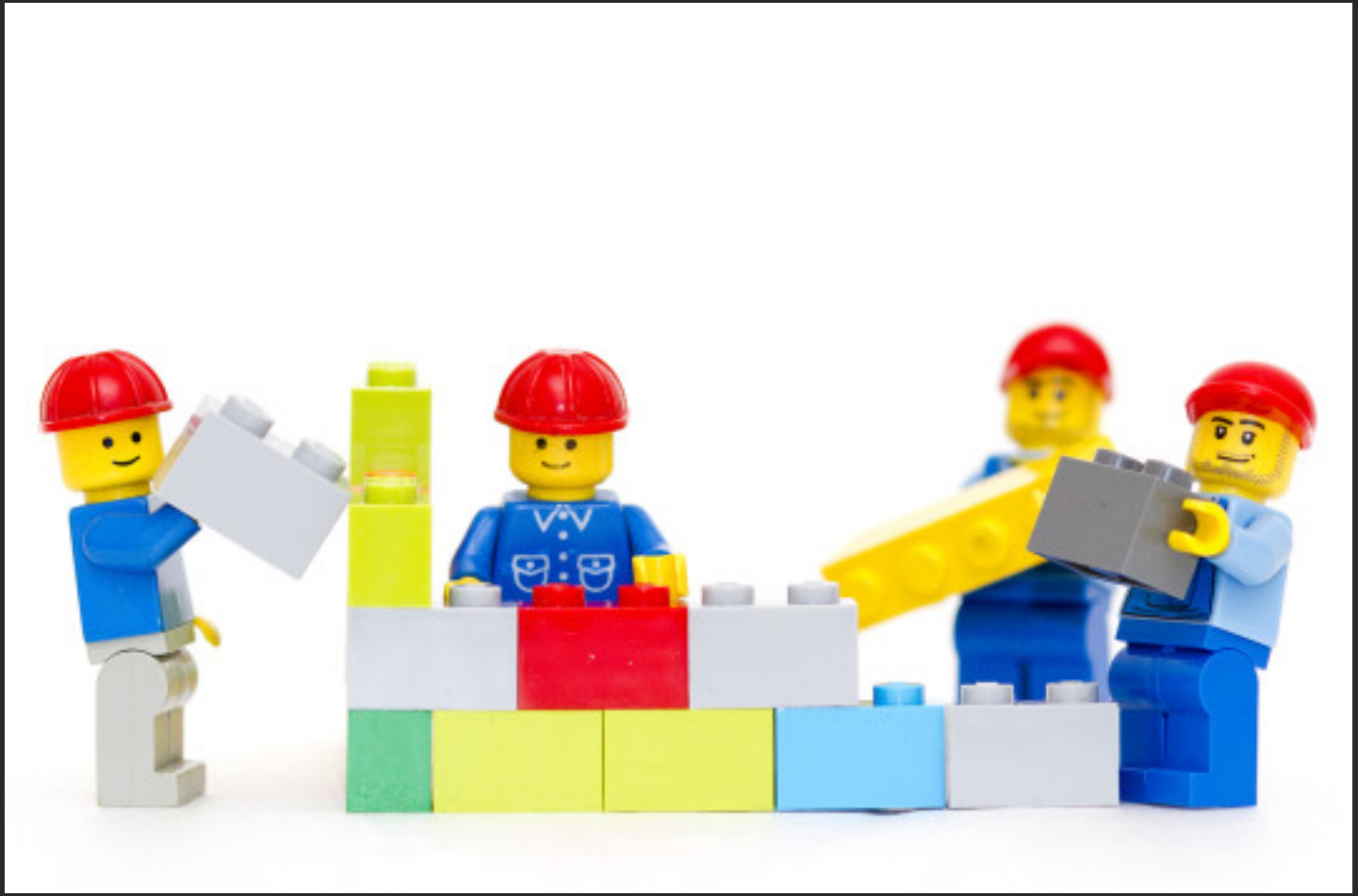
Microservices – Costs

Microservices
Is an org change!

Org changes are **hard!**

Evolving the organization

Central infrastructure investment



Migration doesn't happen overnight

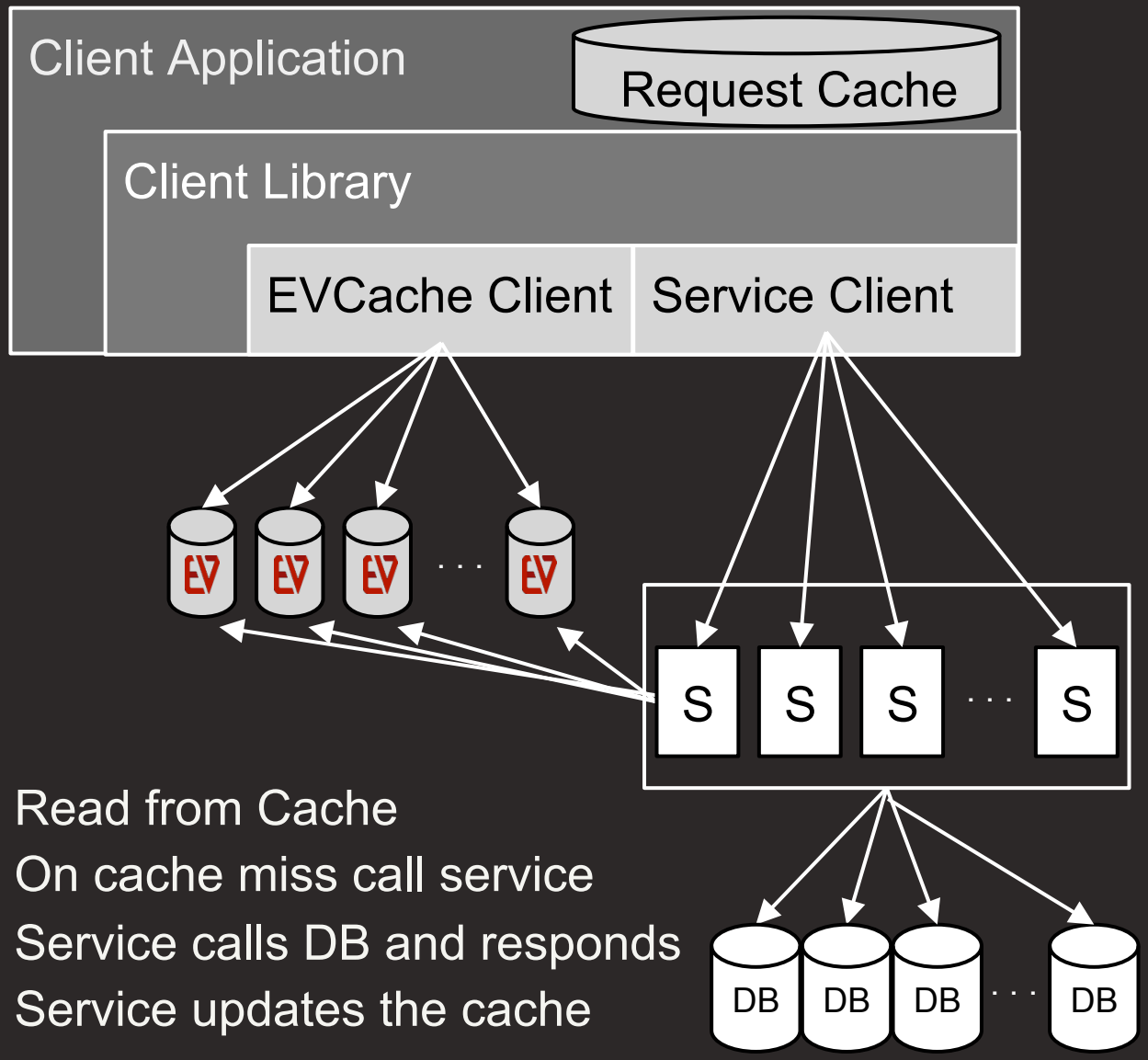
- Living in the hybrid world
- Supporting 2 tech stacks
- Double the maintenance
- Multi-master data replication

Microservices - Lessons Learned

IPC is crucial for loose coupling

- Common language between the services
- Establishes the contract of interaction

Caching to protect DBs

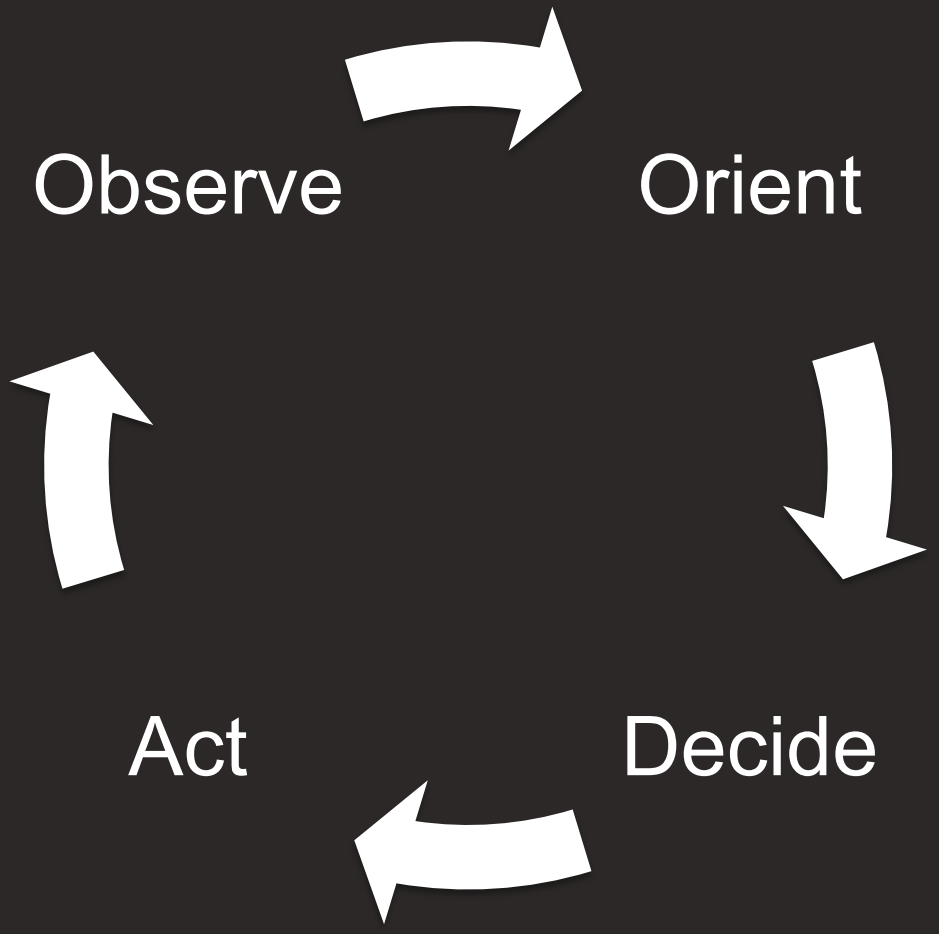


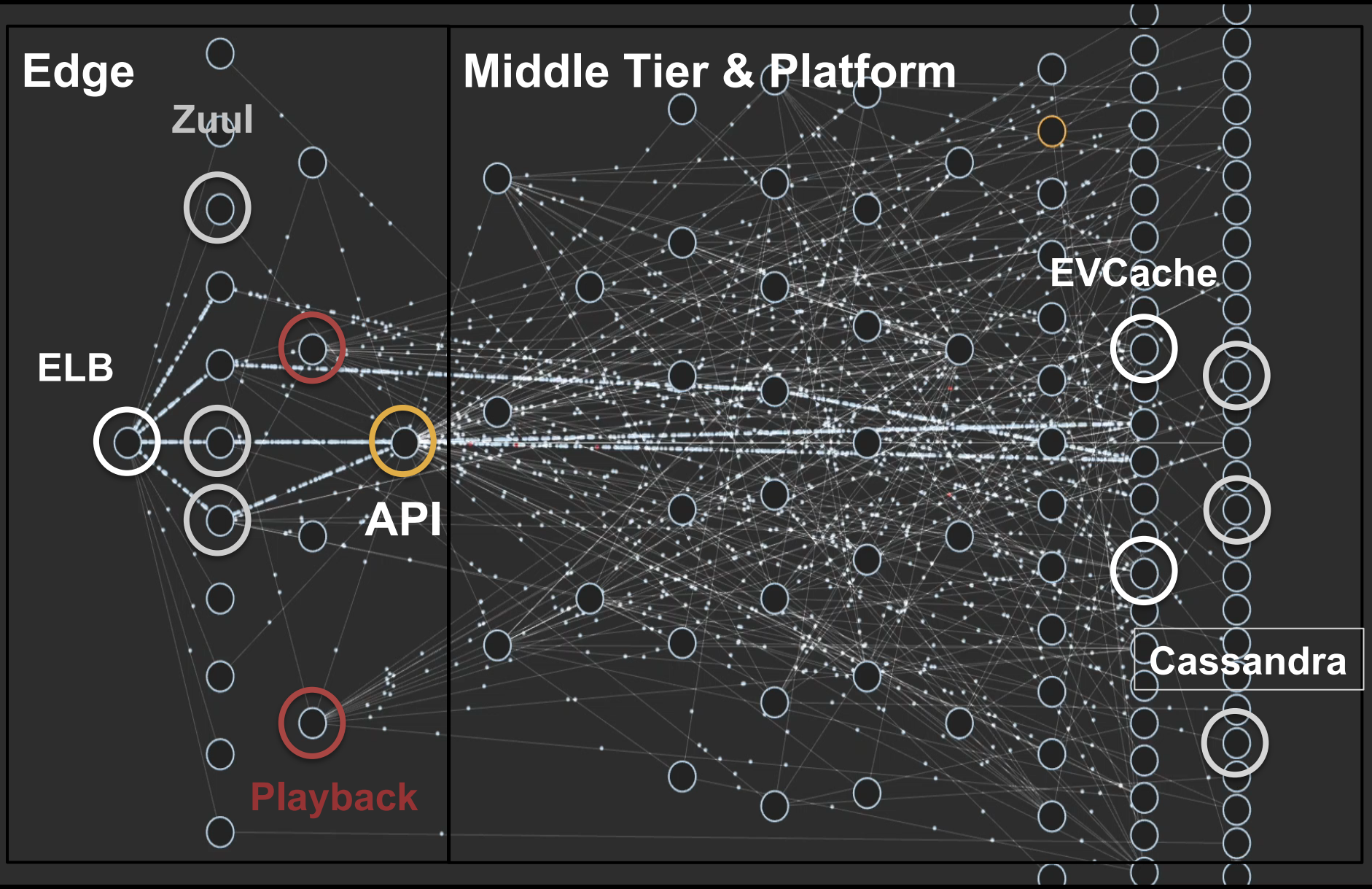
1. Read from Cache
2. On cache miss call service
3. Service calls DB and responds
4. Service updates the cache

Operational visibility matters

If you can't see it, you can't improve it

Will your Telemetry scale?




Edge
Zuul
ELB
API
Playback
Middle Tier & Platform
EVCache
Cassandra

Reliability Matters

- We strive for 4 9’s of availability
- That leaves only 52 minutes of downtime per **YEAR**
- Netflix outages lead to…

Disappointment

Outrage


Amy
@FrantaFtMaynard
Follow
My Netflix isn't working, call the police!!!
3:57 PM - 3 Feb 2015
6
6

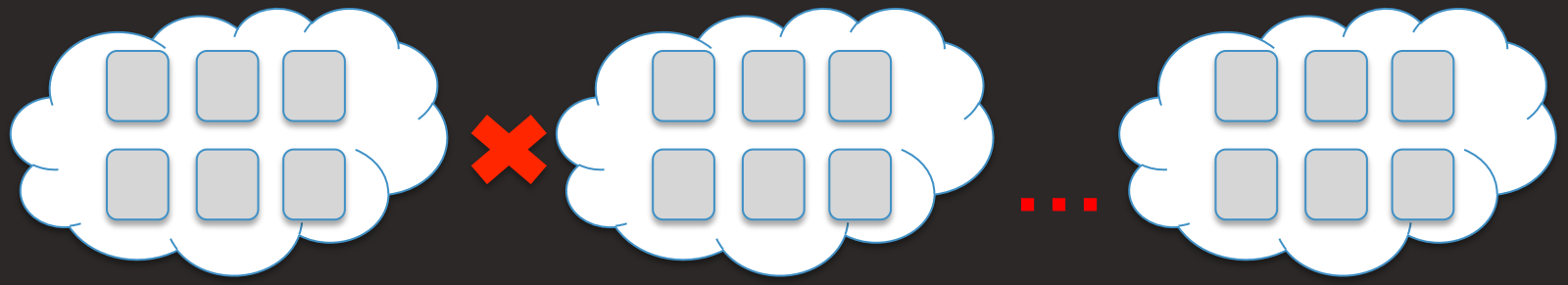
Withdrawal


The Strumbellas
@thestrumbellas
Follow
I didn't realize my dependency on netflix until it went down and I'm now shaking in the bathtub in a hulk costume quoting 'friends' lines.
4:29 PM - 3 Feb 2015 · Kawartha Lakes, Ontario, Canada
19
47

Humor


Jeff Schneider Retweeted
Danny Mullane @PlatinumMuller · 57m
Netflix is down. Talked to my dad there for a few minutes, seems like a nice guy..
1
1

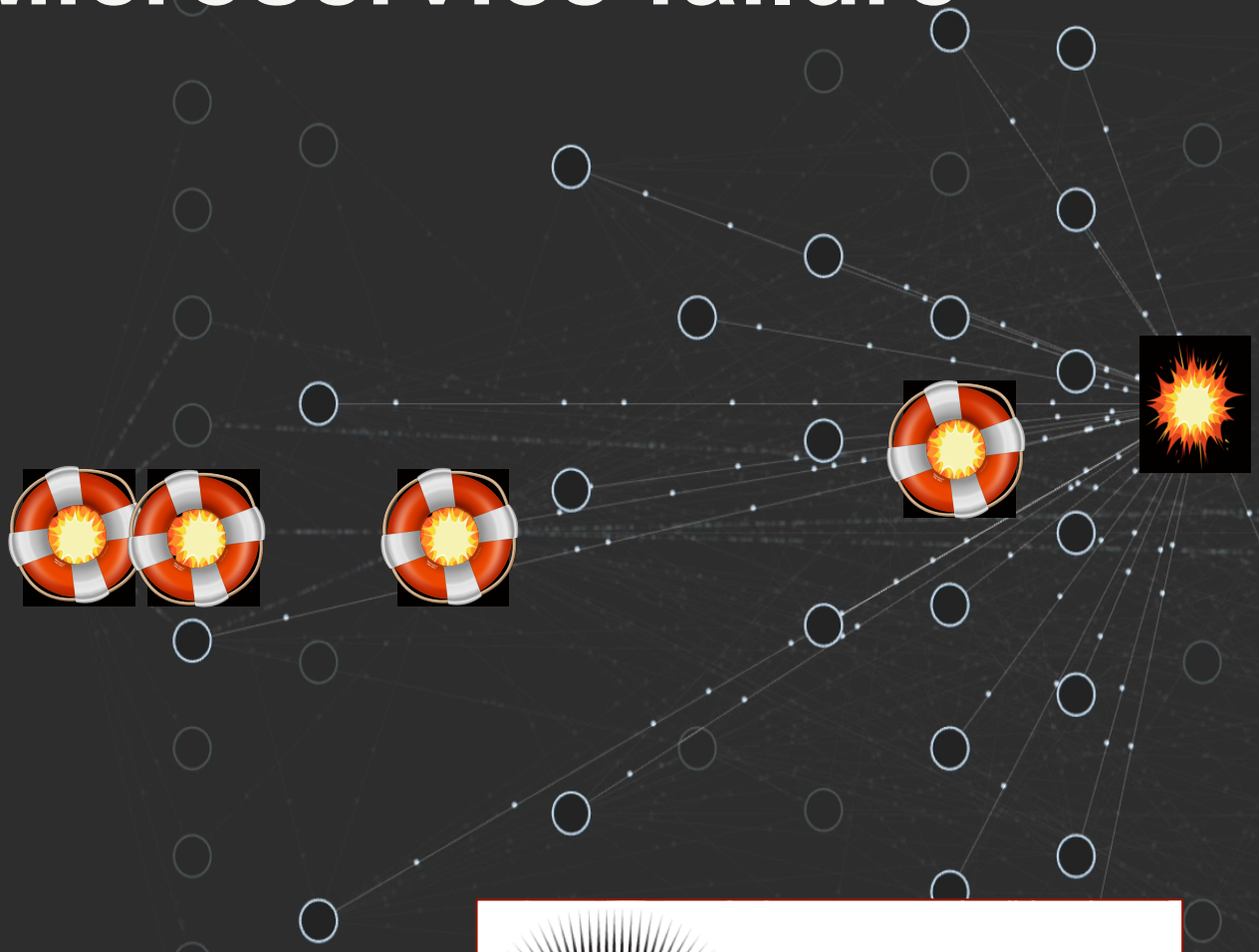
Cascading failures



99% availability 99% availability 99% availability

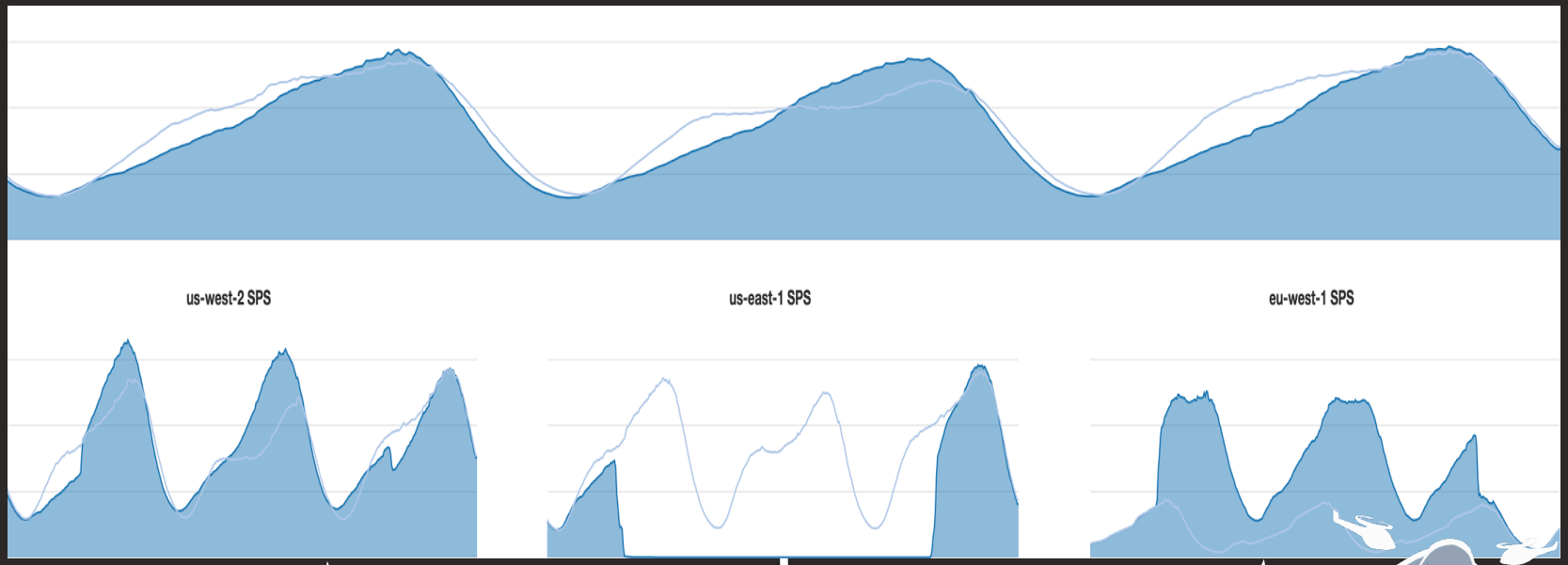
$99\%^{500} = 0.0657\%$

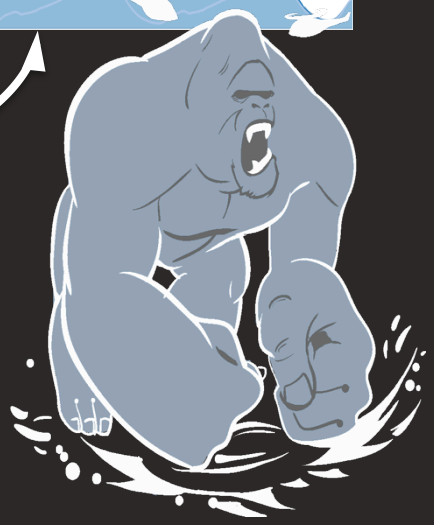
Microservice failure



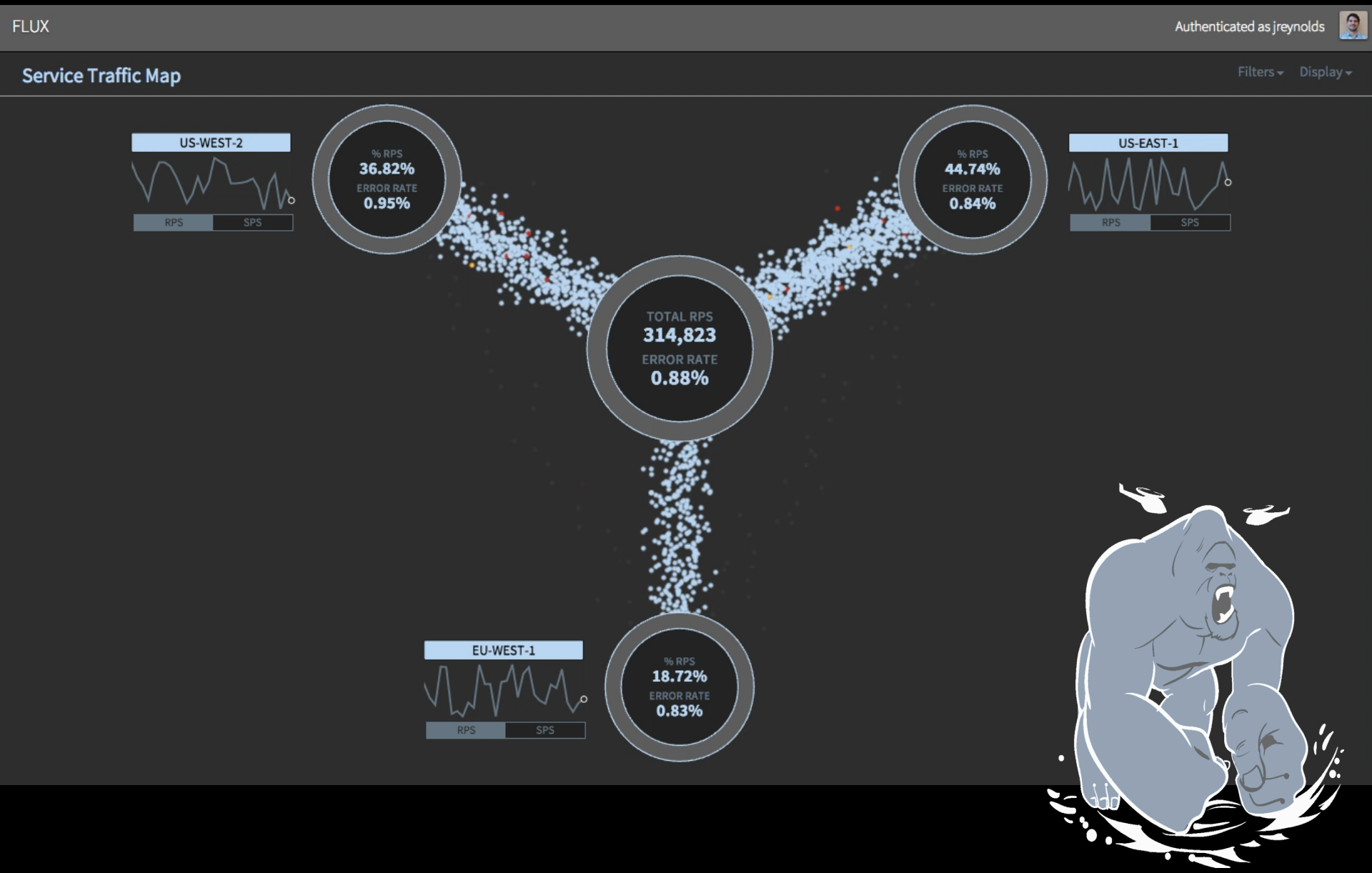
FIT
Fault-Injection
Test Framework

Regional fail-over





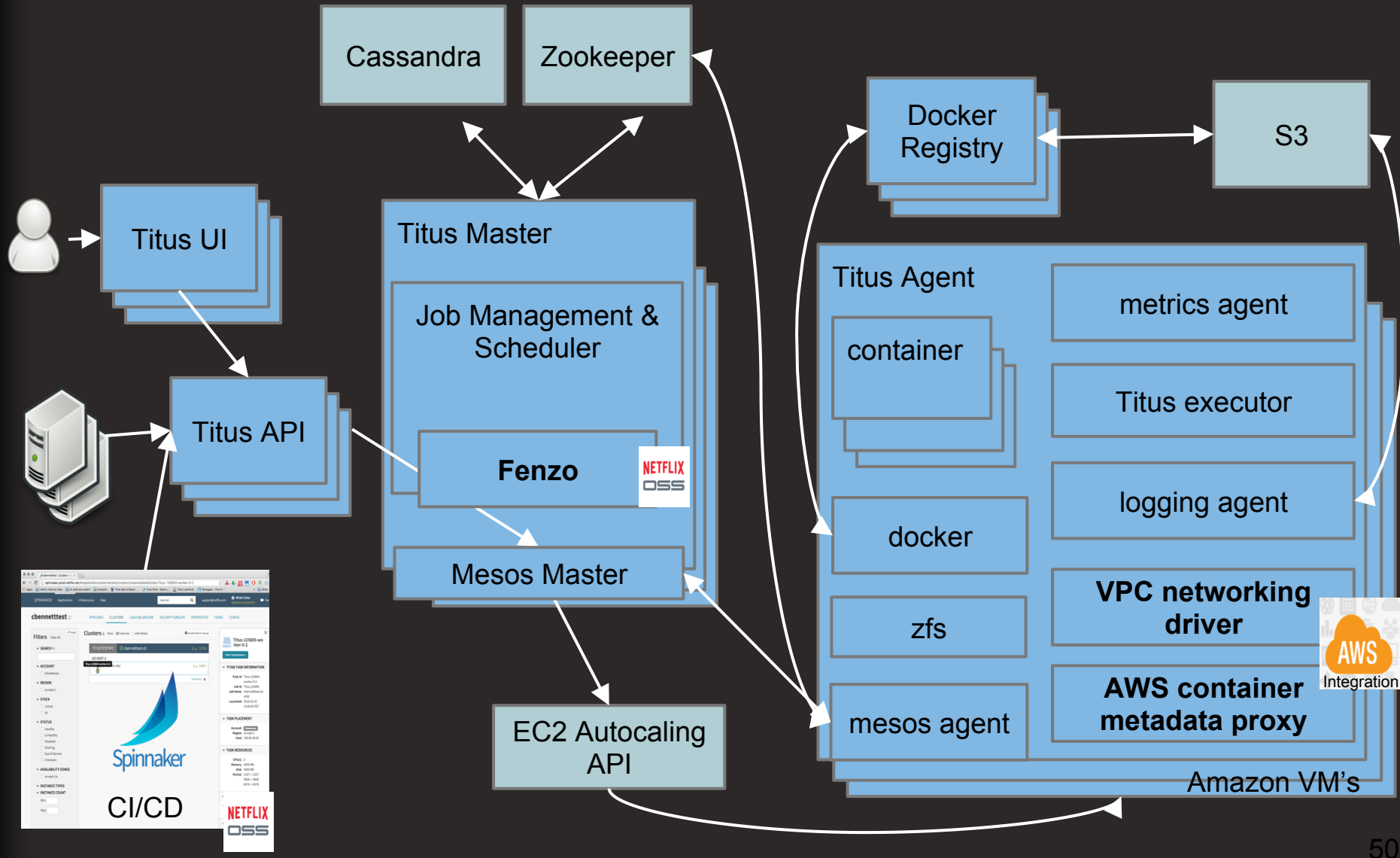
Regional fail-over



A word on containers

- Containers change the level of encapsulation from VM to process
- Containers can help deliver great developer experience
- To run containers in production at scale…

Requires something like this:



Microservices - Resources

Netflix Open Source Software Center

Netflix is committed to open source. Netflix both leverages and provides open source technology focused on providing the leading Internet television network. Our technology focuses on providing immersive experiences across all internet-connected screens. Netflix's deployment technology allows for continuous build and integration into our worldwide deployments serving members in over 50 countries. Our focus on reliability defined the bar for cloud based elastic deployments with several layers of failover. Netflix also provides the technology to operate services responsibility with operational insight, peak performance, and security. We provide technologies for data (persistent & semi-persistent) that serve the real-time load to our 62 million members, as well as power the big data analytics that allow us to make informed decisions on how to improve our service. If you want to learn more, jump into any of the functional areas below to learn more.

http://netflix.github.com

Build and Delivery Tools

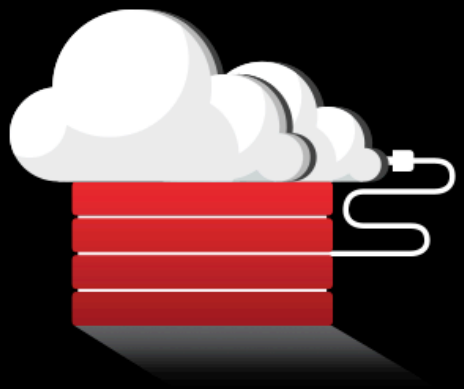
Taking code from desktop to the cloud

Netflix has open sourced many of our Gradle plugins under the name Nebula. Nebula started off as a set of strong opinions to make Gradle simple to use for our developers. But we quickly learned that we could use the same assumptions on our open source projects and on other Gradle plugins to make them easy to build, test and deploy. By standardizing plugin development, we've lowered the barrier to generating them, allowing us to keep our build modular and composable.

We require additional tools to take these builds from the developers' desks to AWS. There are tens of thousands of instances running Netflix. Every one of these runs on top of an image created by our open source tool Aminator. Once packaged, these AMIs are deployed to AWS using our cloud deployment and management tool, Spinnaker.

http://netflix.github.com

NETFLIX | OSS



Common Runtime Services & Libraries

Runtime containers, libraries and services that power microservices

The cloud platform is the foundation and technology stack for the majority of the services within Netflix. The cloud platform consists of cloud services, application libraries and application containers. Specifically, the platform provides service discovery through Eureka, distributed configuration through Archaius, resilient and intelligent inter-process and service communication through Ribbon. To provide reliability beyond single service calls, Hystrix is provided to isolate latency and fault tolerance at runtime. The previous libraries and services can be used with any JVM based container.

The platform provides JVM container services through Karyon and Governator and support for non-JVM runtimes via the Prana sidecar. While Prana provides proxy capabilities within an instance, Zuul (which integrates Hystrix, Eureka, and Ribbon as part of its IPC capabilities) provides dyamically scriptable proxying at the edge of the cloud deployment.

The platform works well within the EC2 cloud utilizing the Amazon autoscaler. For container applications and batch jobs running on Apache Mesos, Fenzo is a scheduler that provides advanced scheduling and resource management for cloud native frameworks. Fenzo provides plugin implementations for bin packing, cluster autoscaling, and custom scheduling optimizations can be implemented through user-defined plugins.

http://netflix.github.com

NETFLIX | OSS

Data Persistence

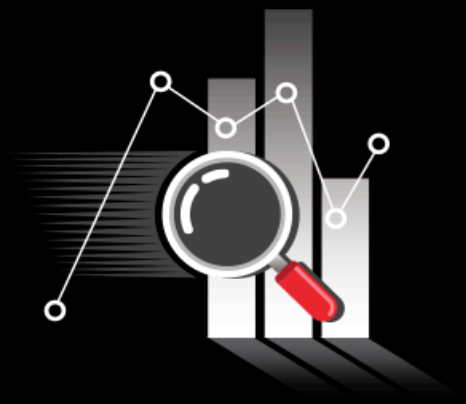
Storing and Serving data in the Cloud.

Handling over a trillion data operations per day requires an interesting mix of "off the shelf OSS" and in house projects. No single data technology can meet every use case or satisfy every latency requirement. Our needs range from non-durable in-memory stores like Memcached and Redis, to searchable datastores such as Elastic and durable must-never-go-down datastores like Cassandra and MySQL.

Our Cloud usage and the scale at which we consume these technologies, has required us to build tools and services that enhance the datastores we use. We've created the sidecars Raigad and Priam to help with the deployment, management and backup/recovery of our hundreds of Elastic and Cassandra clusters. We've created EVCache and Dynomite to use Memcached and Redis at scale. We've even developed the Dyno client library to better consume Dynomite in the Cloud.

http://netflix.github.com

NETFLIX | OSS



Insight, Reliability and Performance

Providing Actionable Insight at Massive Scale

Telemetry and metrics play a critical role in the operations of any company, and at more than a billion metrics per minute flowing into Atlas, our time-series telemetry platform, they play a critical role at Netflix. However, Operational Insight is considered a higher-order family of products at Netflix, including the ability to understand the current components of our cloud ecosystem via Edda, and the easy integration of Java application code with Atlas via the Spectator library.

Effective performance instrumentation allows engineers to drill quickly on a massive volume of metrics, making critical decisions quickly and efficiently. Vector exposes high-resolution host-level metrics with minimal overhead.

Outside of the operational domain, cost management and visibility into where our resources are used in the cloud is a multi-million question to be answered; we've built Ice as a way to expose ongoing cost and and cloud utilization trends to engineers so they can have a better understanding of the footprint of their applications in our environment.

Finally to validate reliability, we have the Simian Army which tests our instances for random failures.

http://netflix.github.com

NETFLIX | OSS

Security

Defending at Scale

Security is an increasingly important area for organizations of all types and sizes, and Netflix is happy to contribute a variety of security tools and solutions to the open source community. Our security-related open source efforts focus primarily on operational tools and systems to make security teams more efficient and effective when securing large and dynamic environments.

Security Monkey helps monitor and secure large AWS-based environments, allowing security teams to identify potential security weaknesses. Scumblr is an intelligence gathering tool that leverages Internet-wide targeted searches to surface specific security issues for investigation. FIDO (not a part of or service of the FIDO Alliance) is a security orchestration framework for analyzing events and automating incident response.

http://netflix.github.com

NETFLIX | OSS

Wrap up

Microservices bring great value to development velocity, availability and other dimensions

Microservices at scale require organizational change and centralized infrastructure investment

Be aware of your situation and what works for you

Questions?

Ruslan Meshenberg
@rusmeshenberg