Microservices

From Winning the War …To Keeping the Peace

Andrew McVeigh

QConSF October 2018

Background

A bit about Andrew's work...

- Many different domains
- Trading & risk systems

- PhD on software components

- Riot Games
- Hulu

Microservices Work!

But make sure you prepare for the challenges...

1. Microservices in Gaming
2. Microservices for Video
3. Believe the Hype
4. Believe the Challenges

1.

Microservices in Gaming

A microservice architecture
@ scale

League of Legends

@ Scale


LEAGUE OF LEGENDS STATS
MORE THAN
67MILLION
MONTHLY ACTIVE PLAYERS
MORE THAN
27MILLION
DAILY ACTIVE PLAYERS
MORE THAN
7.5MILLION
PEAK CONCURRENT PLAYERS
STATS RELEASED JANUARY 2014

Gaming Particulars

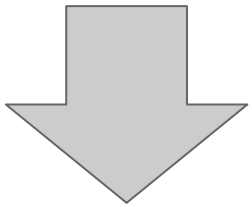
- Low latency
- Need to match up players = shared state
- Rapid development cycles
- Lots of engineers working on 1 game

1A.
Winning the War

Evolving from a monolith

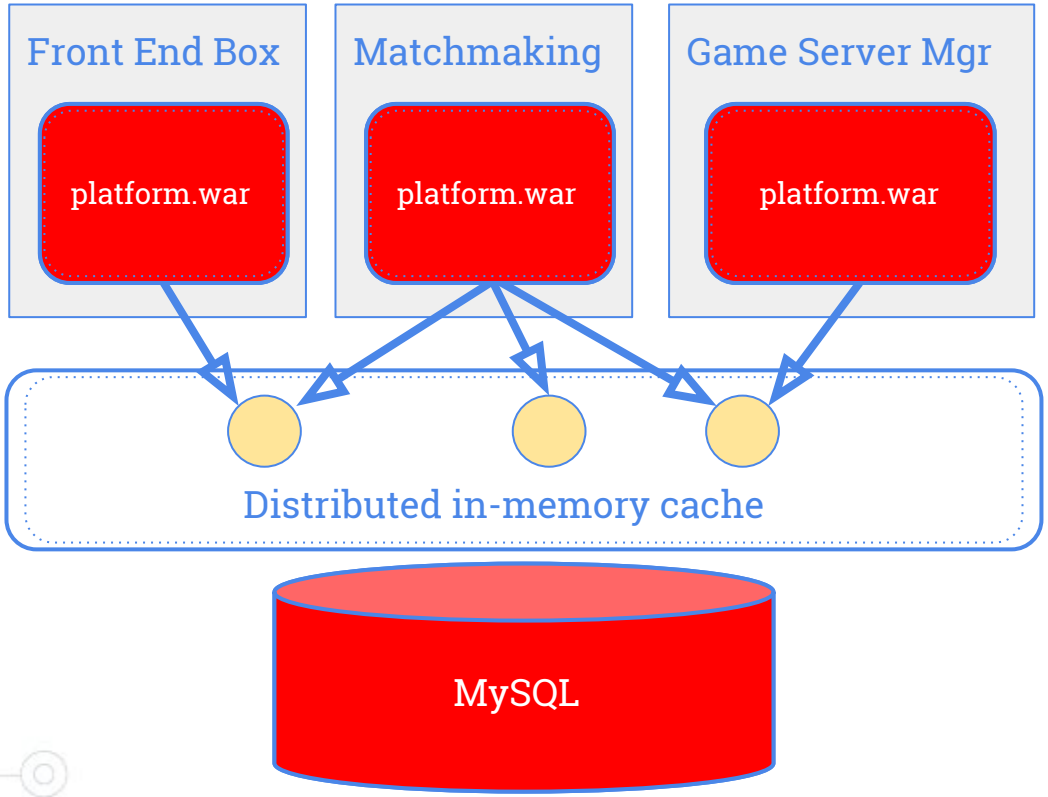
Evolving the Architecture

- 2009: Large service monolith
  - Matchmaking, game selection, inventory etc

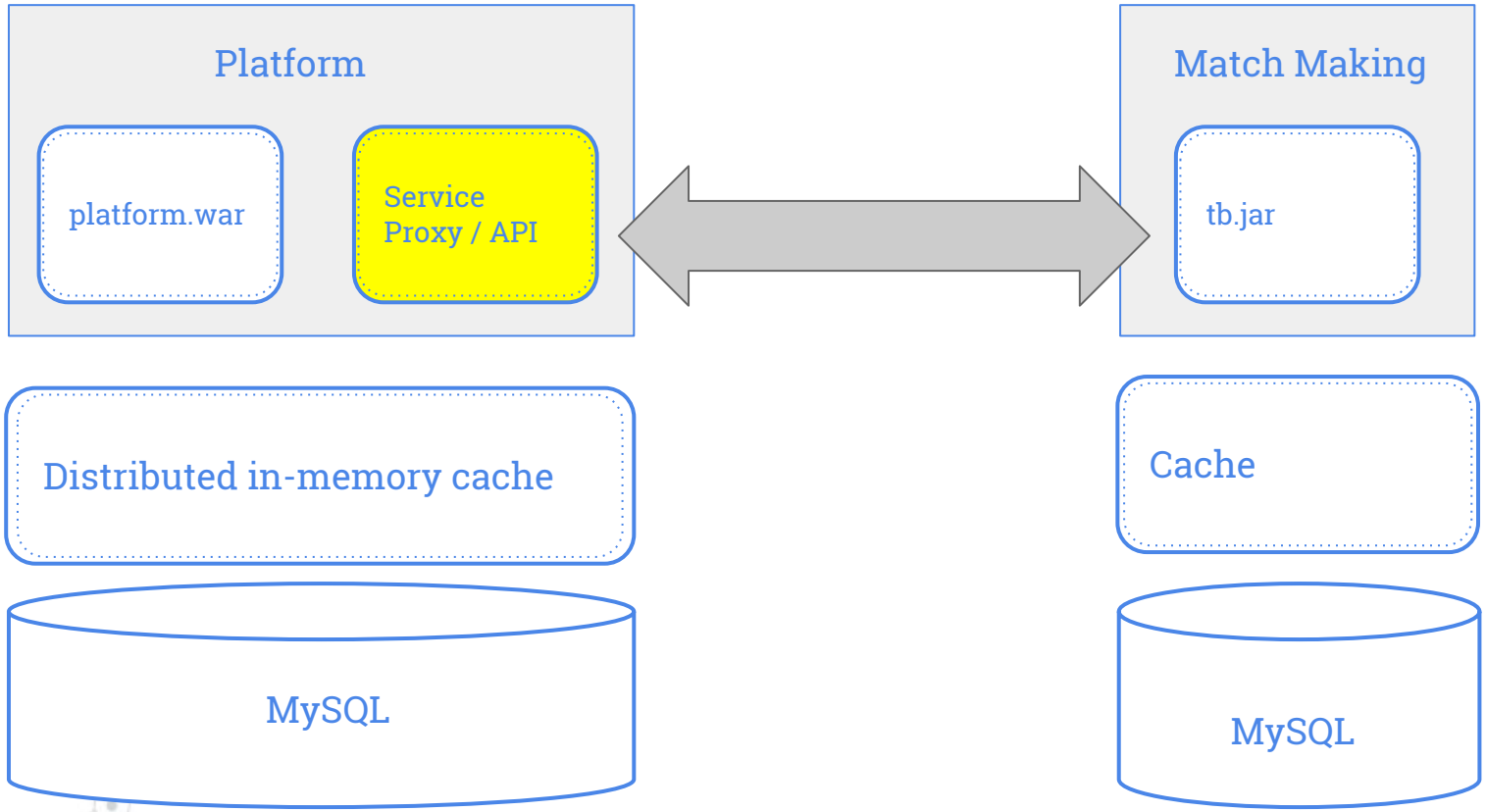


- 2012: Started evolving to microservices
  - http://bit.ly/evolving-league

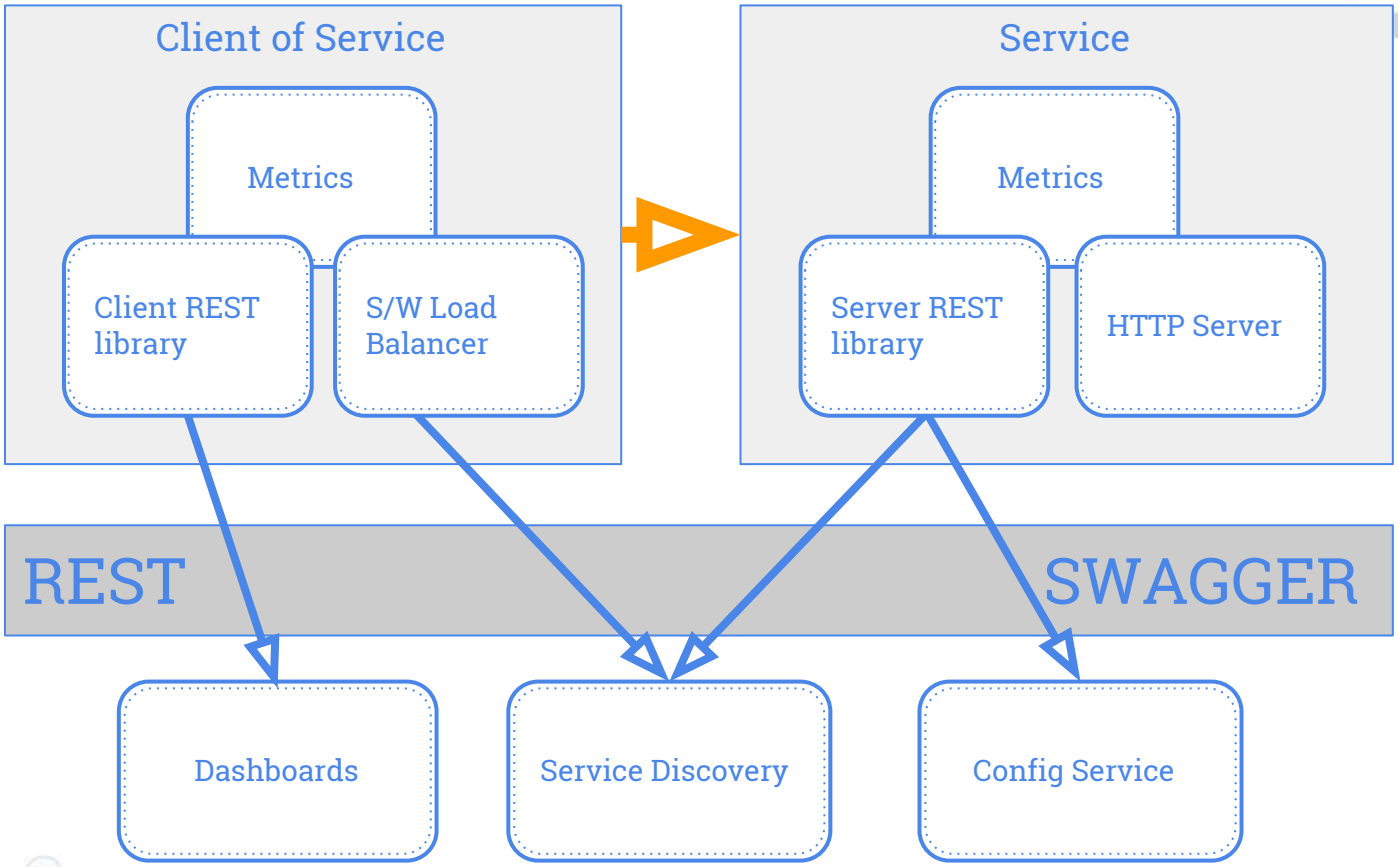
League of Legends



Microservices for New Features



Standard Infrastructure

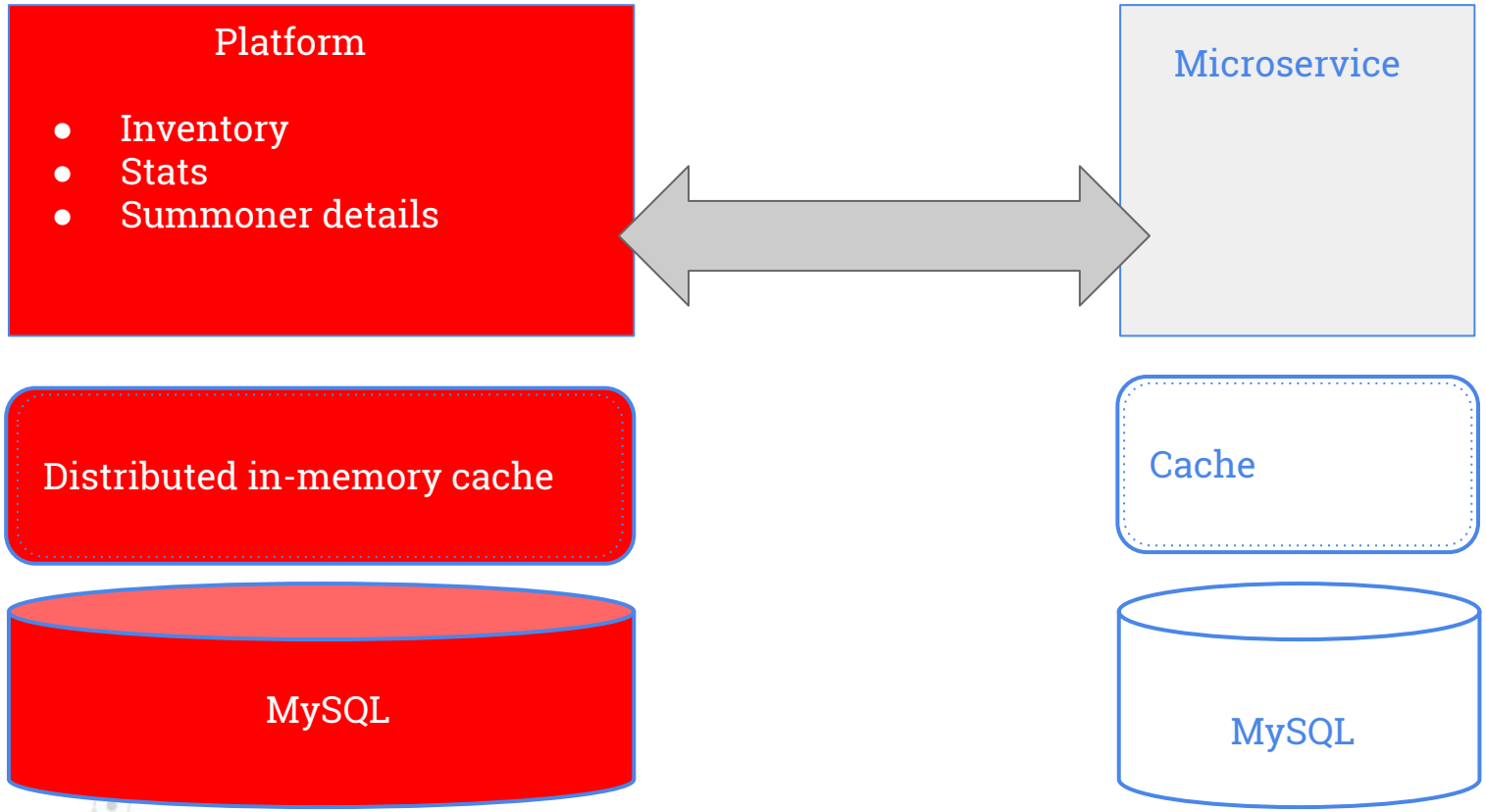


1B.

Keeping the Peace

Held back by remains of the monolith

Not Quite Free of the Monolith...



Ouch - Hitting the Wall

League of Legends's Clash tourneys delayed indefinitely

Dominic Tarason
Contributor
3rd July 2018 / 7:14PM

Technically, the platform was written as a monolithic service, which means that when things go wrong, it's difficult to debug. This monolithic platform handles things like: starting games, telling your client what skins and champions you own, what level you are, etc. Over time, we've been refactoring aspects of the platform opportunistically, but this is a long-term process that is akin to rebuilding parts of the airplane mid-flight.

If We Could Redo?

- Decouple state completely
  - Inventory service
  - Catalog service
  - Runes service
- Socialize to get prioritization
- Simplify infrastructure
  - Config System too “clever”
  - Too much “smarts” in fat libraries

2.

Microservices for Internet Video

Hundreds of tiny pieces...

Hulu

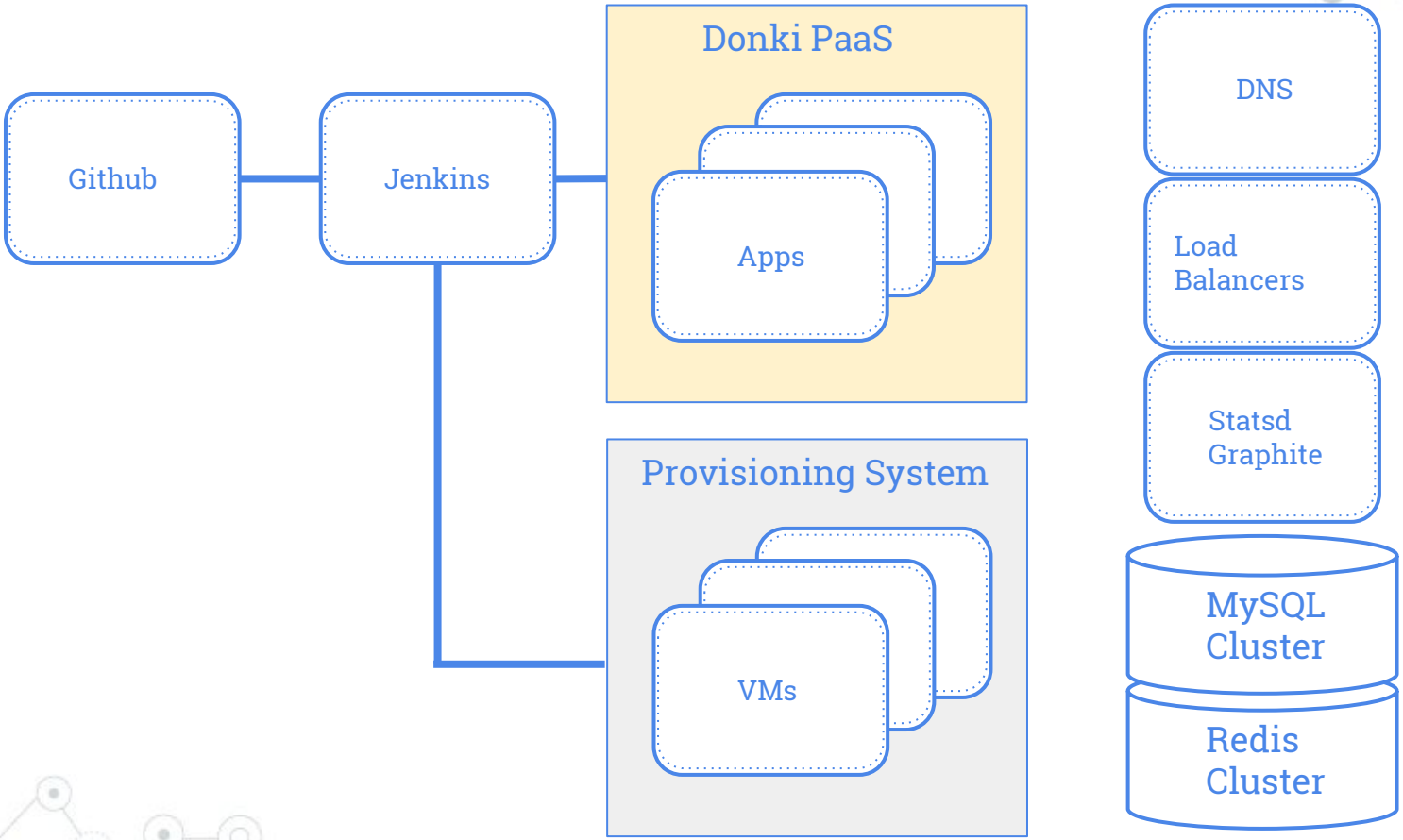
- 2016: Full microservices architecture
  - Evolved / replaced existing VOD architecture
  - Live TV
  - 20m+ total subs
  - 1m+ live subs
- 15 month development!
  - 800+ microservices

http://bit.ly/hulu-landscape

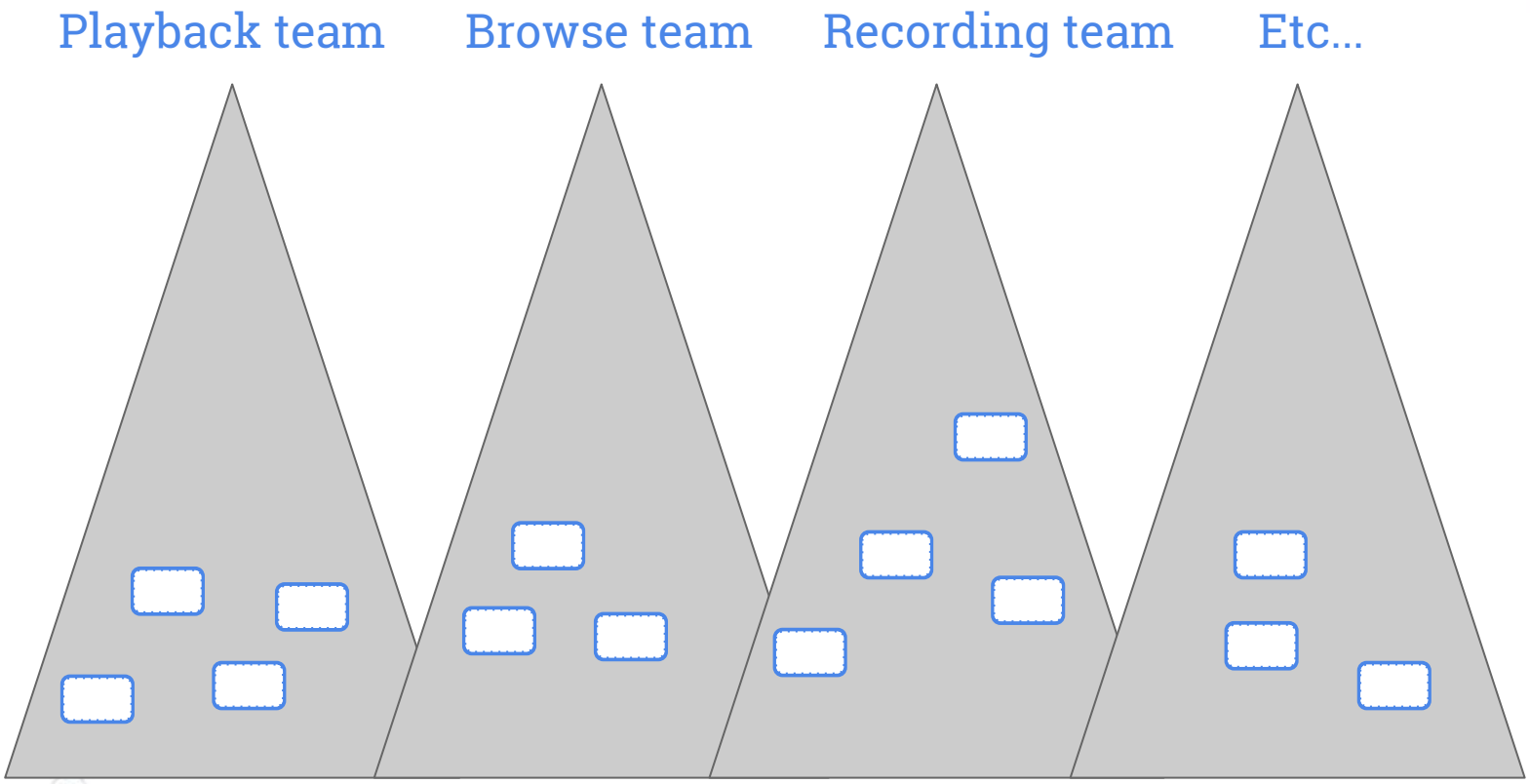
Video System Particulars

- Lots of caching to support browsing
- TV show metadata needed everywhere
- Real-time playback to support live TV
- Lots of integration (billing, ads etc)

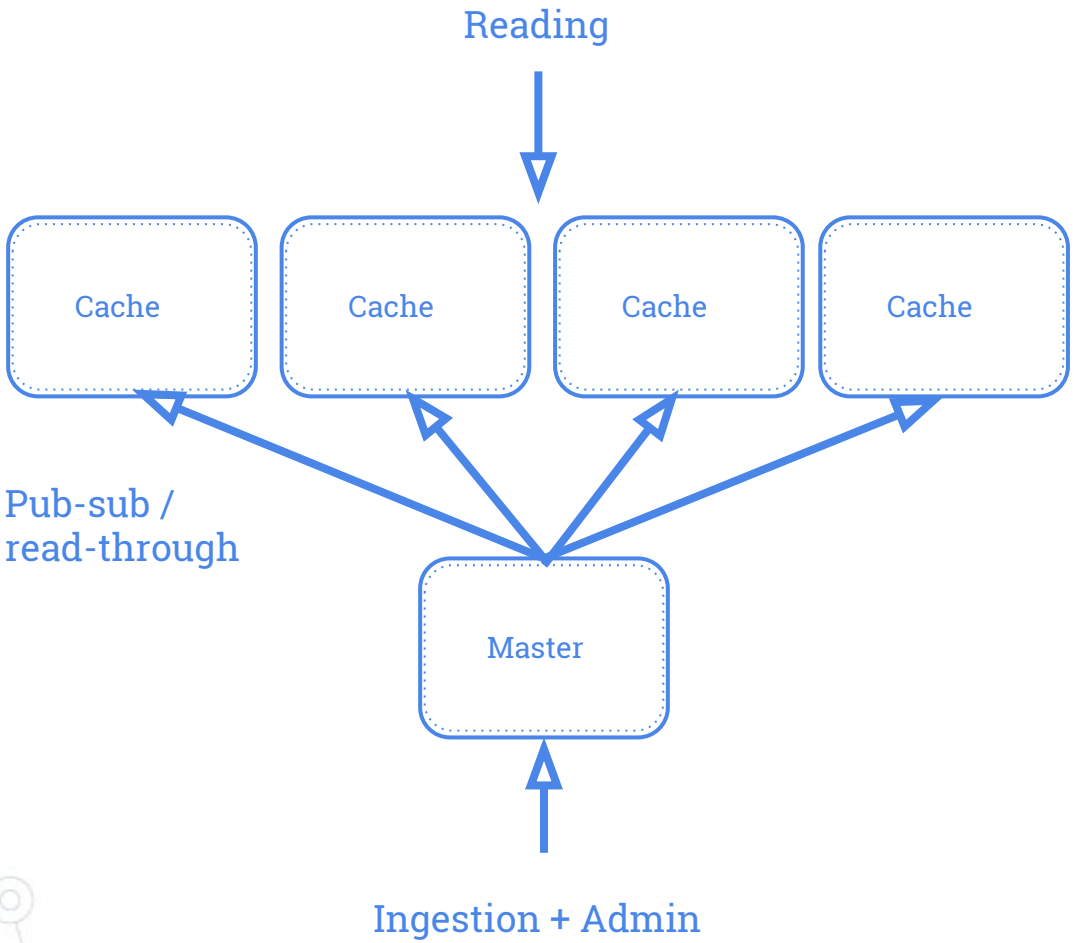
Infrastructure for Microservices



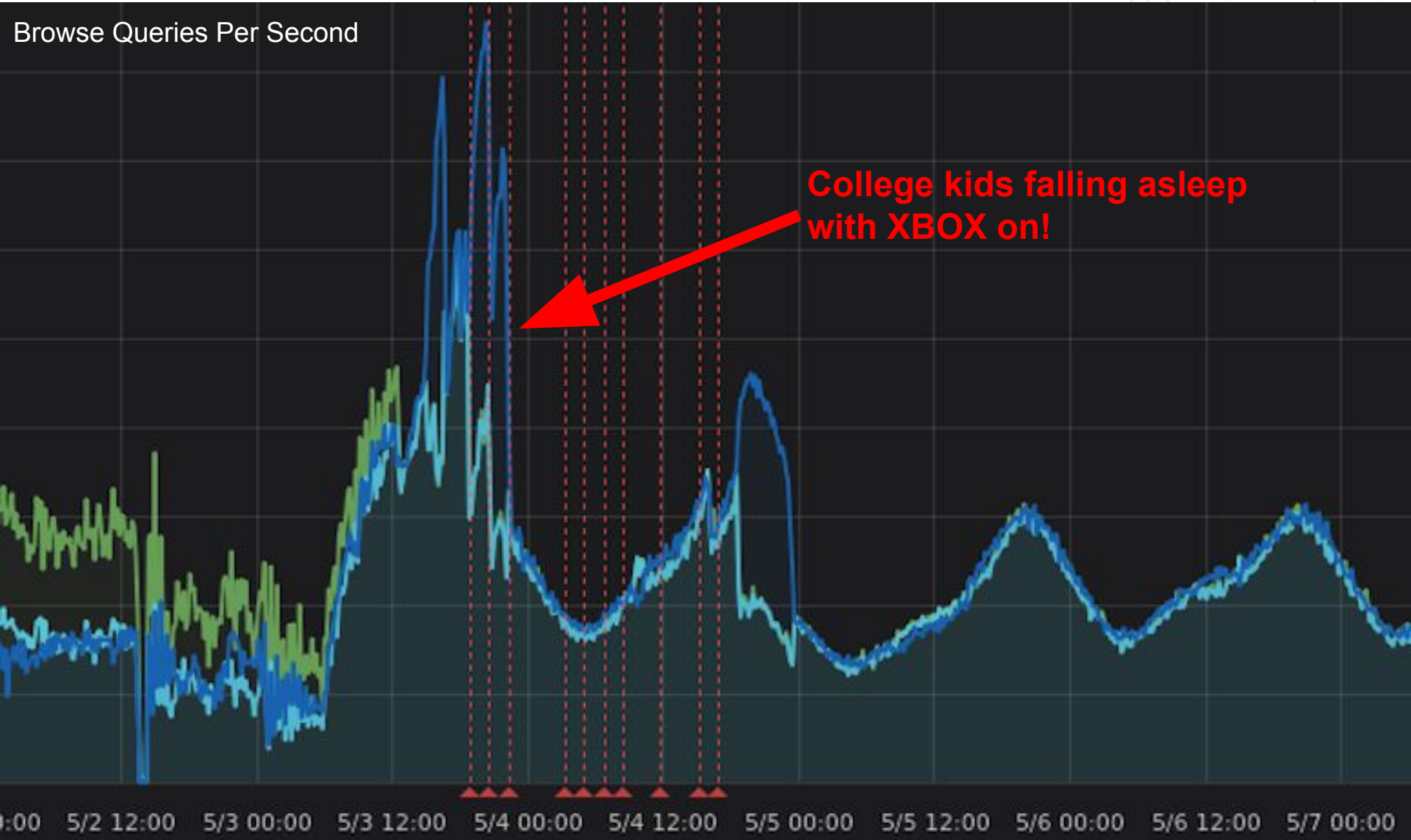
Microservice Ownership



Data Distribution Pattern



Scaling for Launch Day



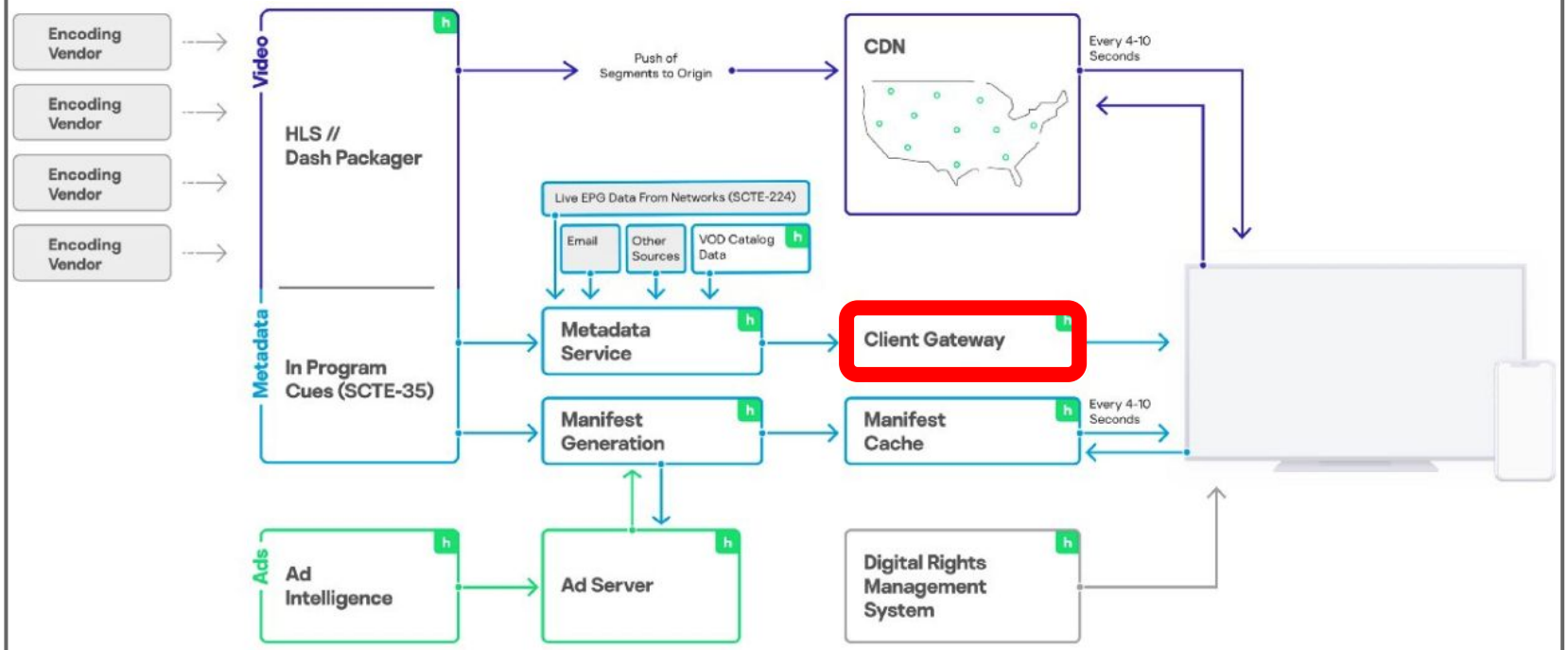
2B.

Keeping the Peace

Every issue gets magnified...

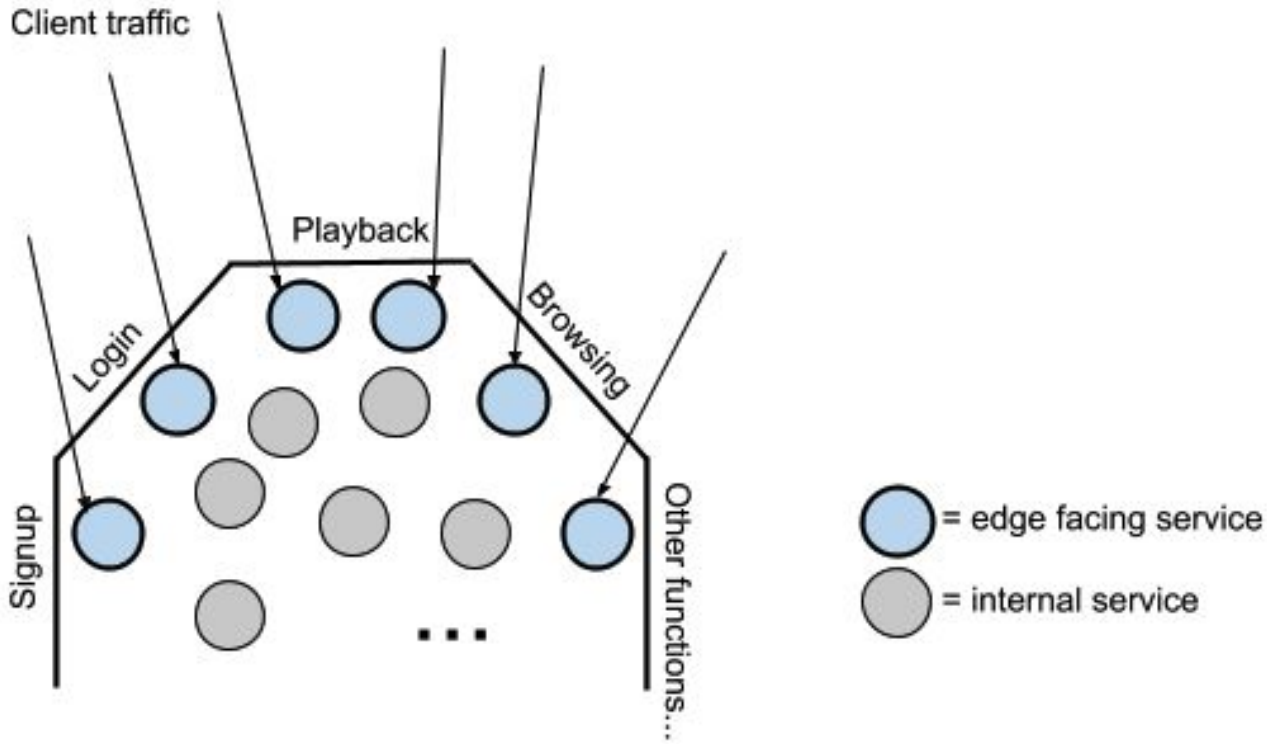
Arch / Ops Dashboard

Login | Browse & Search | Playback | Ads | DVR … etc

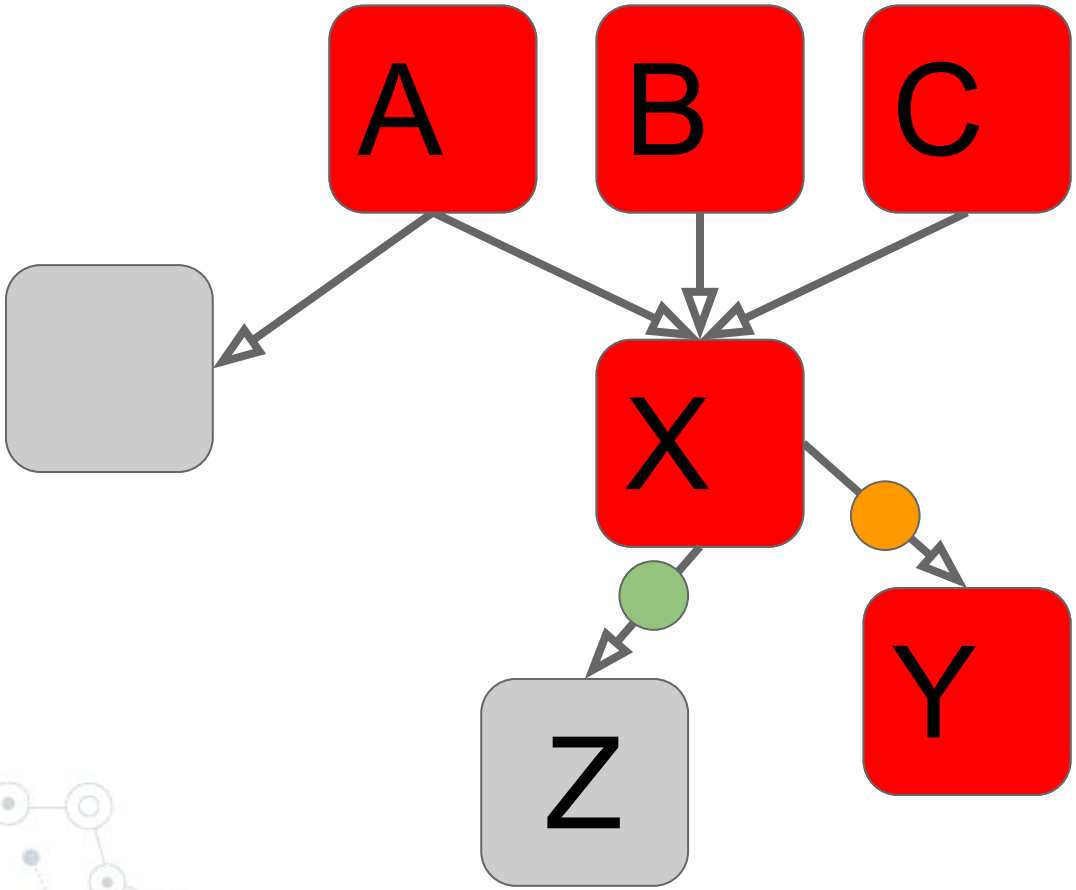


Scaling For Growth

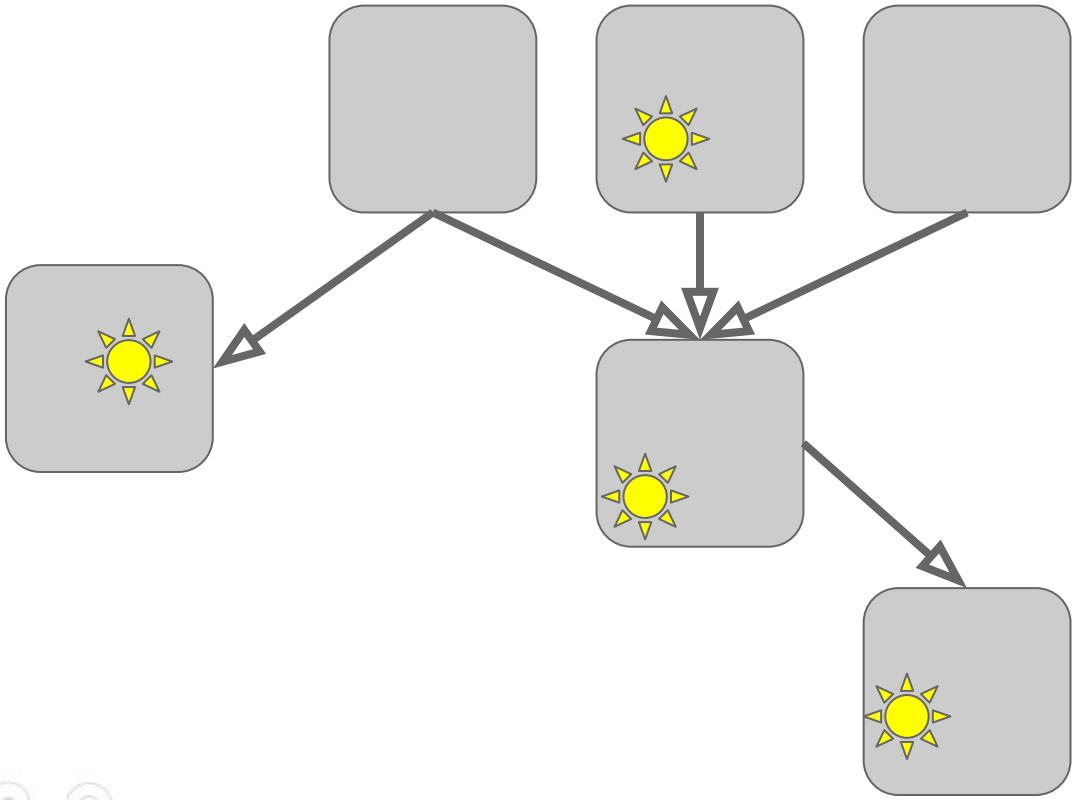
http://bit.ly/hulu-scaling



Circuit Breakers avoid Firestorms



Cross-Cutting Requirements



Cloud versus DC

- Could target cloud or DC
  - But no elasticity, must overprovision
- Abstracting cloud & DC

  == Lowest common denominator
- Hard to do proper blue-green in DC

3.
Believe the Hype

The many benefits of microservices

They Actually Work!

- Ownership & Independence
- Development velocity
- Operational & Development scaling

They Actually Work!

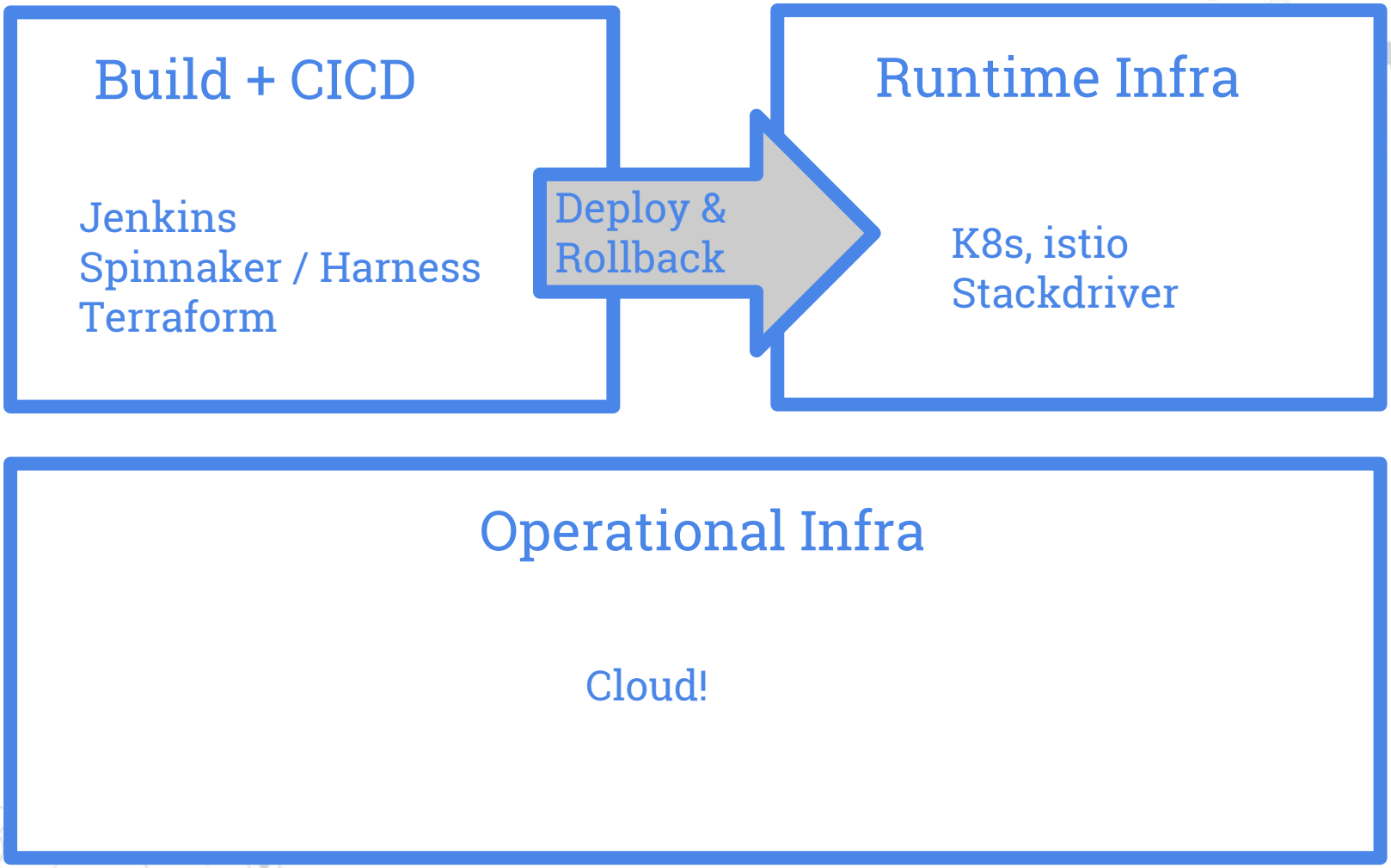
- Granular deployment
- Evolution
- Organizational alignment

4.

Believe the Challenges

Standing on the shoulders of
constantly improving infrastructure...

Common CICD + Operating Env



Cloud Approach

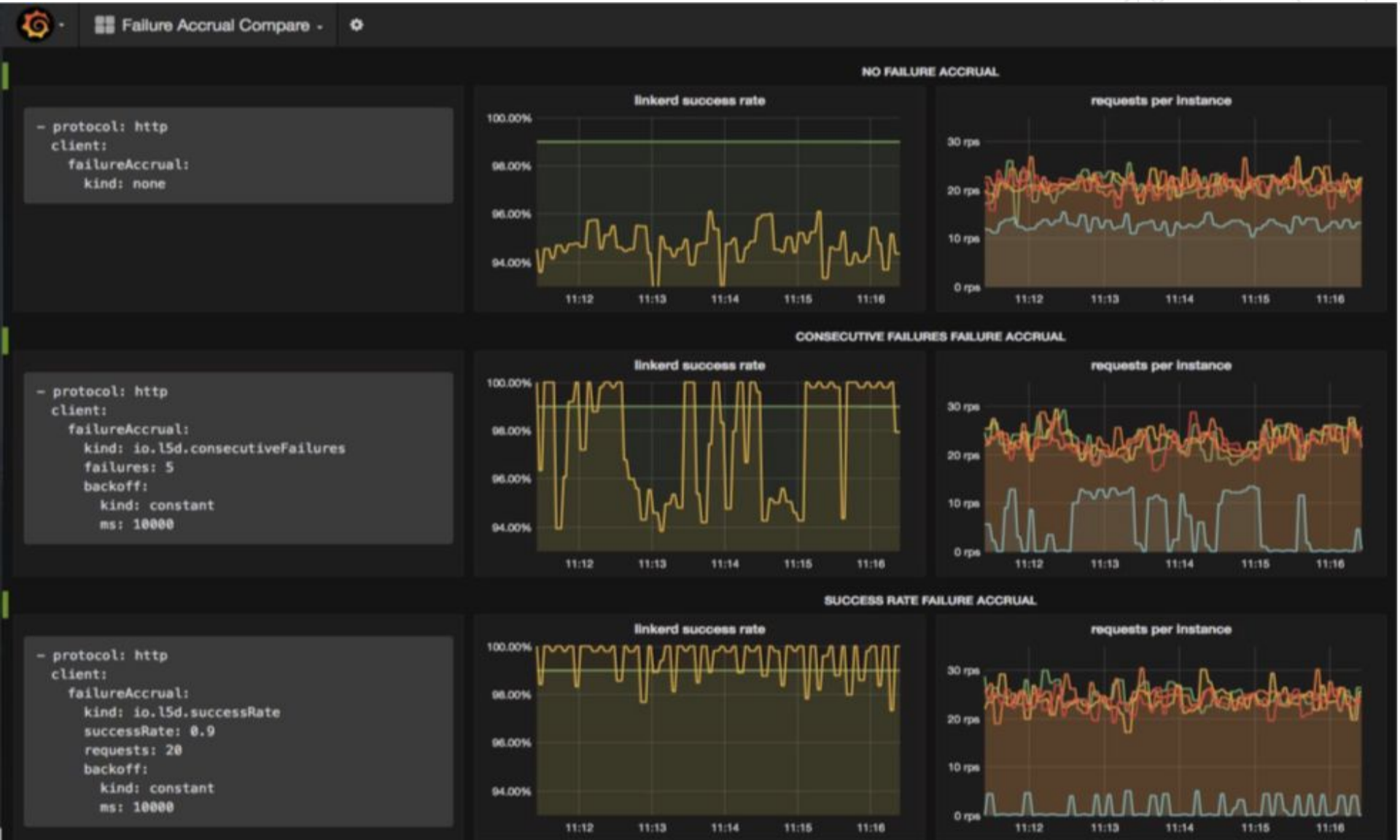
Preferred approach

- Pick one cloud provider (per workflow?)
- Consider costs early
- Multi-region, multi-account on day 1

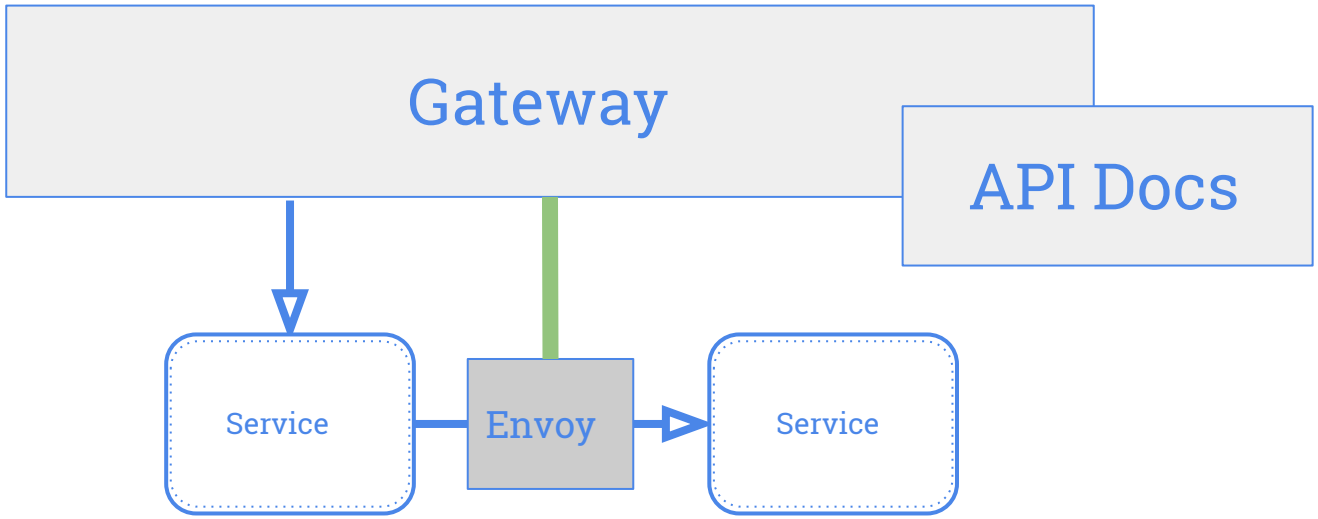
Cloud has so many advantages over DC

- Elasticity
- Easy environments (blue-green)
- Better shared services (db, queues etc)

Circuit Breakers by Default!



API Gateways



Platforms

For things that need holistic treatment

… but still allow self-service

- Load testing
- Billing
- Browse caching
- A / B testing
- UI layout

Takeaways

Microservices offer many benefits

- Isolation & independence
- Granular deployment, scaling & evolution

Use infra to protect against common issues

- Full CICD
- Infrastructure as code
- Circuit breakers to prevent firestorms
- Istio to help with monitoring + more
- Cloud elasticity FTW

Thanks for Listening!

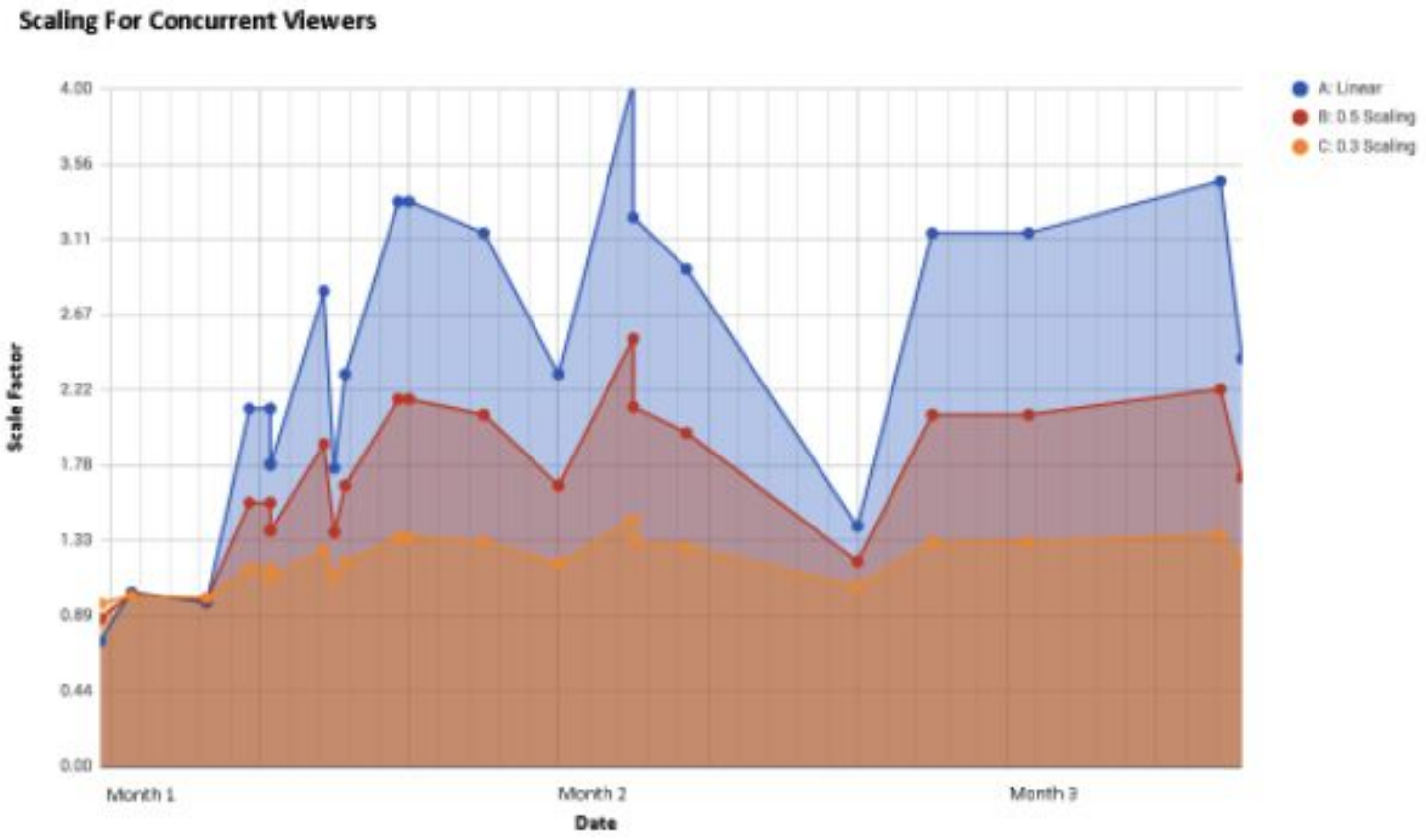
Any questions?

andrew.mcveigh@gmail.com

AMA

Decomposing the Monolith
Tues 2:55pm Boardroom C

Scaling For Growth



But look out!!!