Creating Business Value Through Unified Observability:
Equinox's Journey to Brand and Tooling Consolidation

FEBRUARY 7, 2023

A Presentation by Joel Miller

EQUINOX

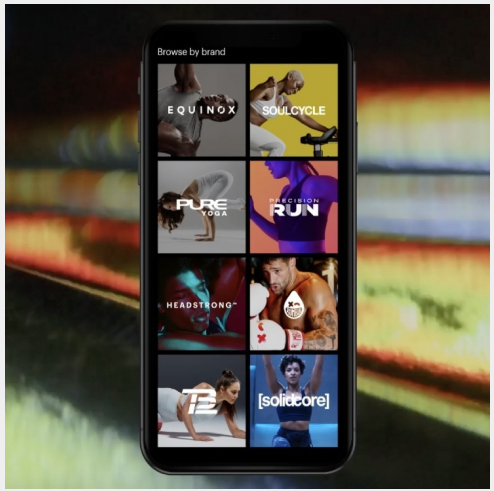
A BIT ABOUT ME

- Software Engineering background with focus on building scalable, secure and performant systems
- OSS Enthusiast – started tinkering with Linux 20 years ago
- ES user since version 1.4 (ca. 2015)
- Currently Director of Platform Engineering at Equinox

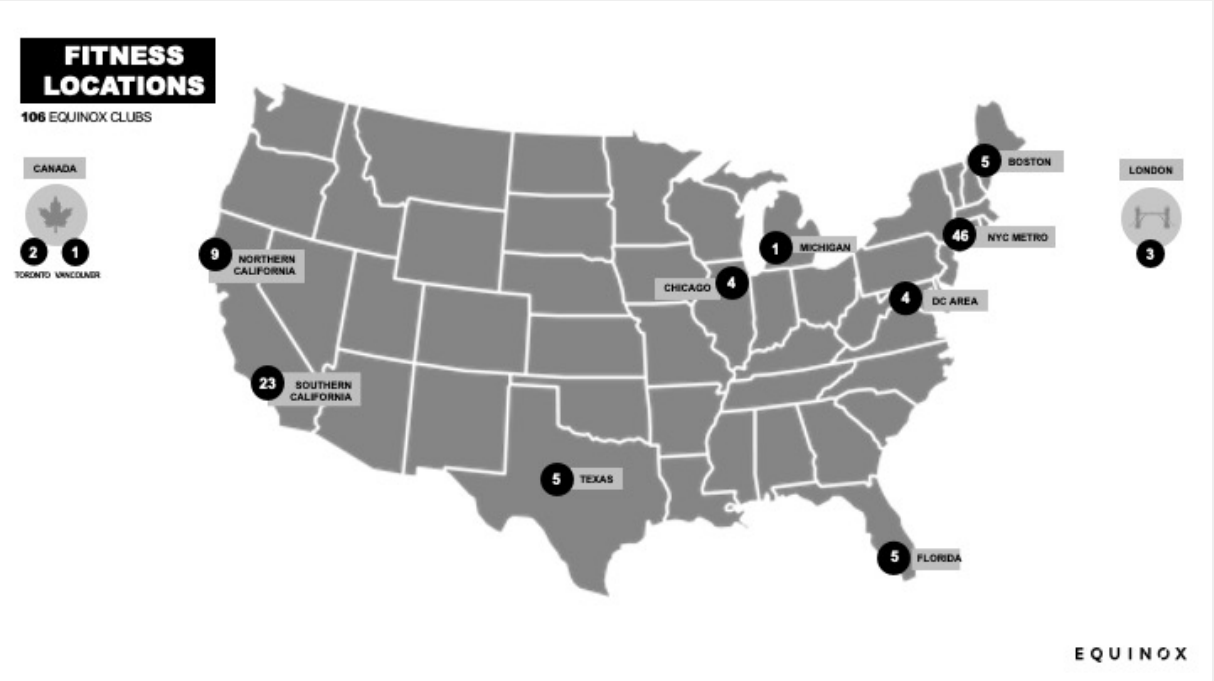
EQUINOX BRAND PORTFOLIO

MEDIA (DIGITAL EXPERIENCE)

EQX+



FITNESS (PHYSICAL CLUBS)



BACKGROUND

DEFINITIONS

- APM = Application Performance Monitoring
- ECS = Elastic Container Service
- EC2 = Elastic Compute Cloud (Amazon-hosted VM)
- LDAP = Lightweight Directory Access Protocol (central authorization system)
- ILM = Index Lifecycle Management
- PR = Pull Request (proposed code changes)

THE PROBLEM

- Separate but similar application architecture and tooling across brands utilizing different standards
- Multiple tools used for observability use cases (logging and APM), each managed separately and living in their own silos
- Prohibitively expensive licensing and ingest quotas for existing tooling which made broader adoption and usage a non-starter

THE GOAL(S)

1. Reduce cost and cognitive load through removing three legacy logging systems and centralizing architecture into one system
2. Correlate logs and traces through global context across all Equinox services
3. Dramatically improve the speed and ability to view and understand application as well as user behavior and proactively solve issues
4. Leverage company-wide LDAP system to manage tool access – everyone gets an account by virtue of onboarding
5. Maintain and improve existing observability functionality

THE VISION

Utilize modular, best-of-breed components to create a “single pane of glass” for anyone in the company to consume the data, metrics and logs they need in a low-friction and seamless way

PROJECT EVALUATION

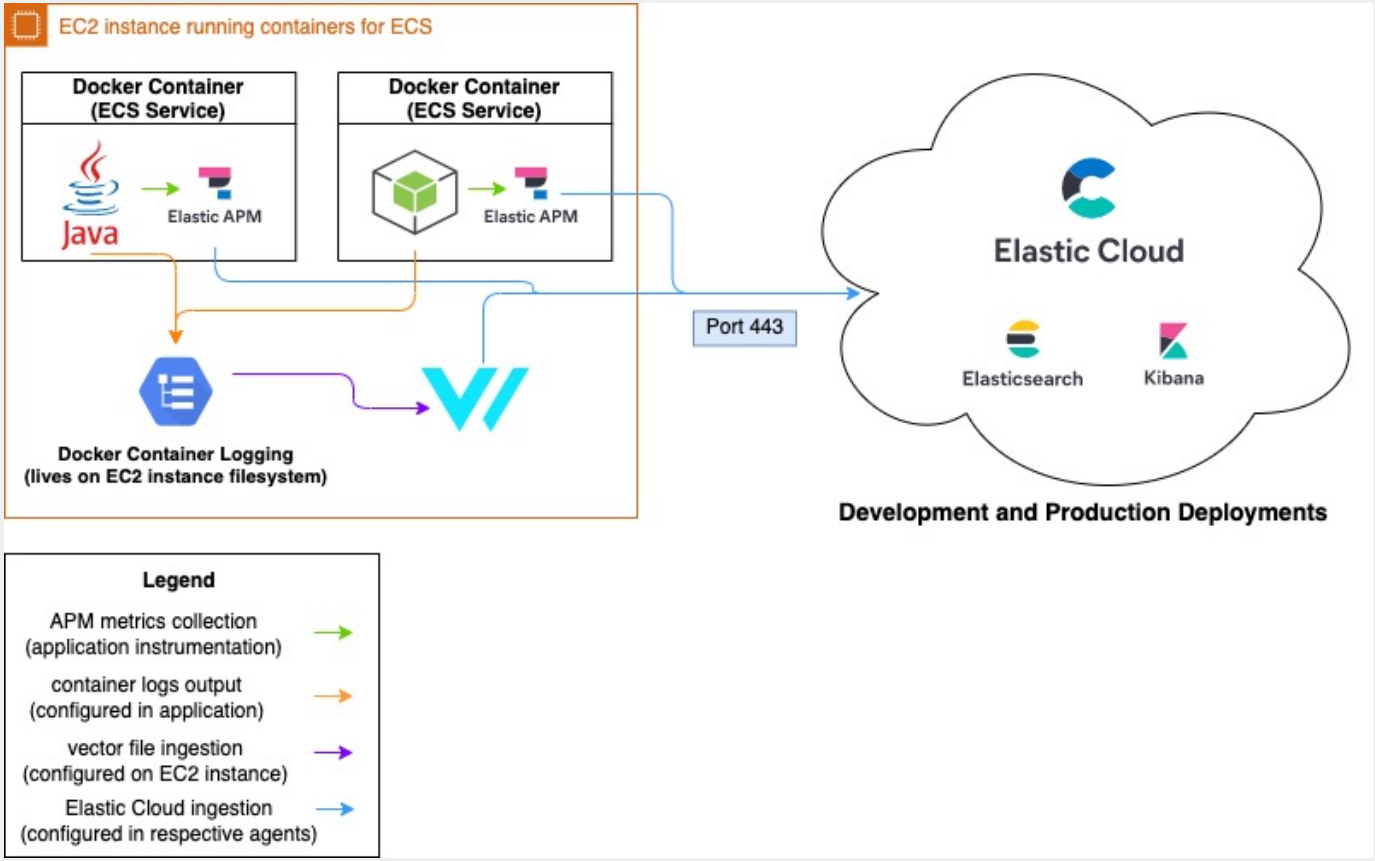
REQUEST FOR COMMENT (RFC) PROCESS

- Designed to be a collaborative process to solicit input for building something
- Comprehensive write-up designed to answer questions around goals, development effort, dependencies, performance, security, cost, etc.
- Socialize with early adopters (volunteers and leads) first and after iterating on initial feedback share with entire tech org
- Create detailed Proof of Concept (POC) instrumenting a sample service as part of the process

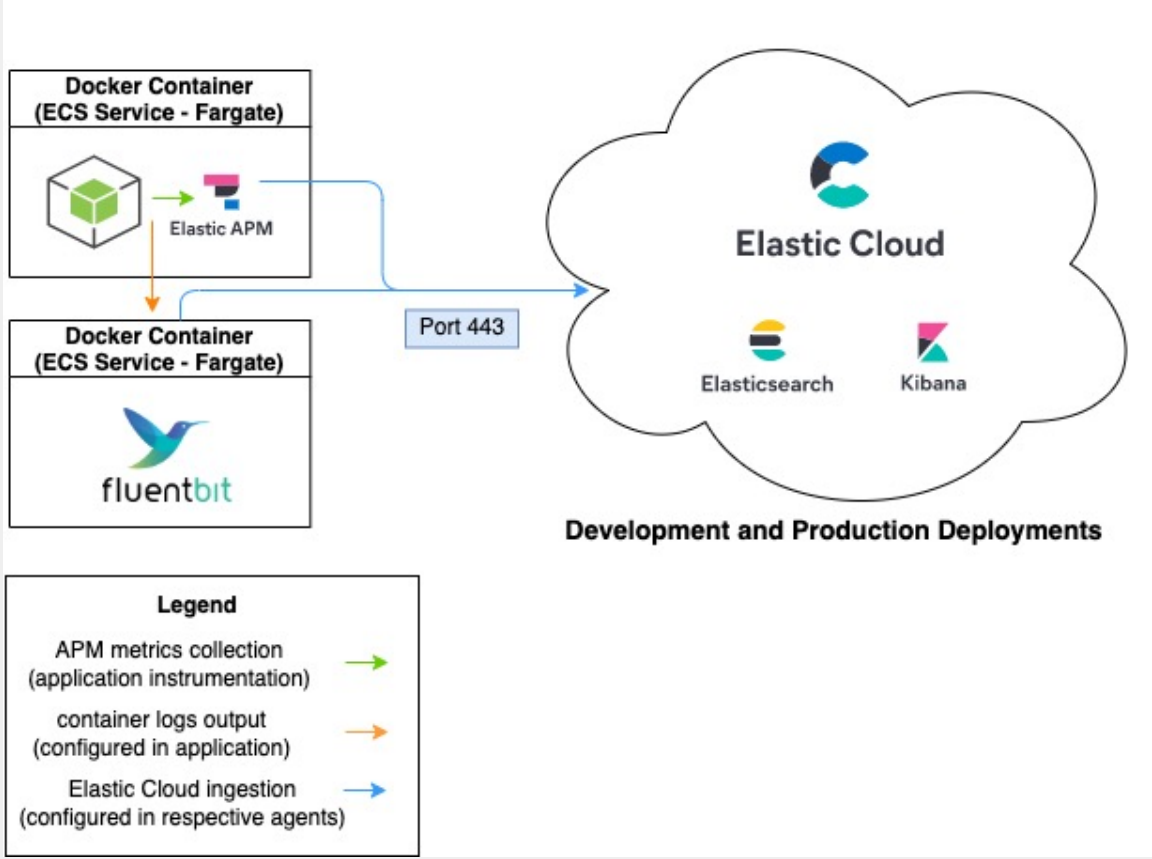
PROOF OF CONCEPT (POC) LEARNINGS

- Evaluated multiple ingest clients for logs (beats, logstash, vector, firelens/fluentbit)
- Developed a template which service owners could copy/paste for instrumenting the most heavily used frameworks (Node.js and SpringBoot Java) with Elastic APM
- Initial iteration on Elastic server-side configurations (ILM, data tiering, cluster sizing)

FITNESS ARCHITECTURE

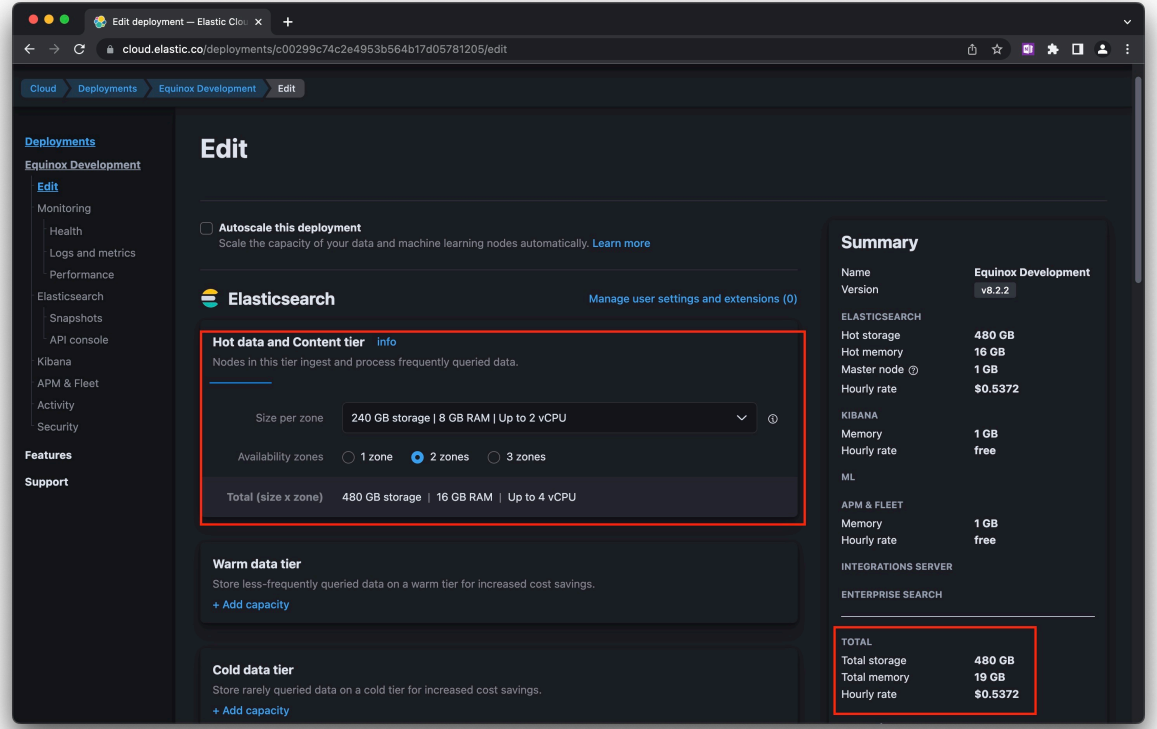


MEDIA ARCHITECTURE



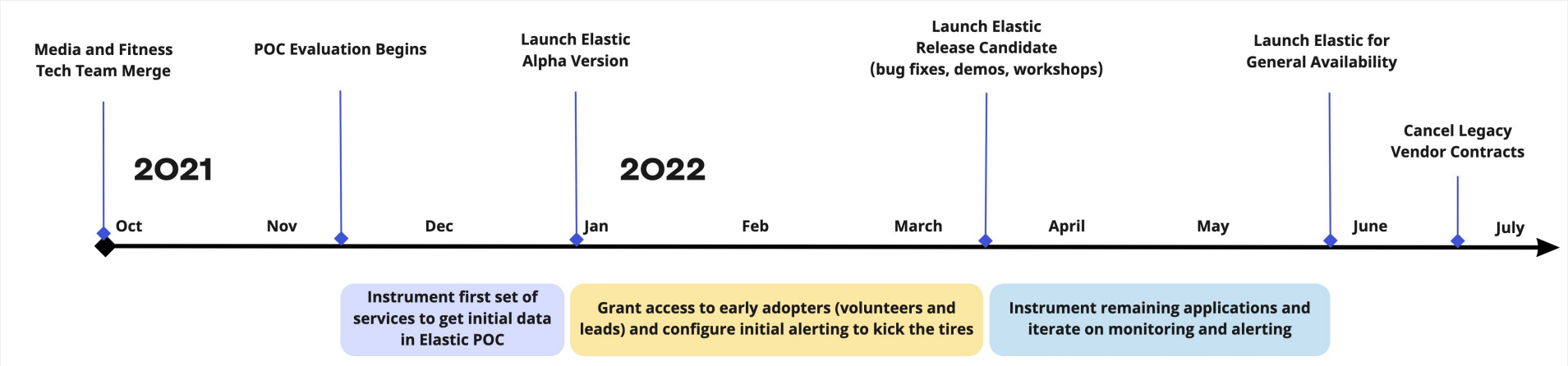
ELASTIC CLOUD ARCHITECTURE

- Cluster Sizing – based on ingest volume and retention
- Created index naming conventions to segregate data by producer, environment and date (<brand>-<environment>-<content>.YYYY.DD.MM)
- ILM – index size, retention policies and data tier movement
- Upgrade and Rollback Agility



PROJECT EXECUTION

TIMELINE AND ROLLOUT



CHALLENGES AND SOLUTIONS

CHALLENGES	SOLUTIONS
Noisy services / abusive loggers (logging every cache call, defaulting to DEBUG logging, etc.)	• Created dashboarding to categorize highest ingest offenders • Cut PRs and worked with service owners to reduce logging to sane levels and dramatically improve signal:noise ratio • Audited core logging and instrumentation – discovered and fixed many naïve defaults
Educating consumers about the structure of the logging schema	• Worked directly with consumers to guide them • Encouraged questions in public forums to help everyone look at the same data and most effectively utilize the tooling • Created Equinox-specific documentation, demos and workshops
APM alert thresholds	• Established high thresholds as a starting point • Worked with service owners to tune for specific threshold requirements (latency, throughput, etc.)

LESSONS LEARNED

- Ensure your index rotation policy is tuned properly before you start sending larger amounts of data (rotate after one day or if index reaches 100 GB)
- Data streams are very useful for the workload characteristics of observability use-cases (append-only time-series)
- Tune your workloads to be very predictable and consistent, then disable auto-scaling; this will avoid unpleasant surprises and config thrash
- Utilize tagging heavily, invest the effort into making your tooling support tags as first-class citizens and educate your users on how to effectively use them
- Ensure your Hot / Warm / Cold node cluster setup is consistent between all data tiers – if it is not, you will get silent failures and inconsistent behavior

OUTCOMES

- 80% reduction in annual observability OpEx spend (order of magnitude smaller)
- 50% reduction in log ingest due to improved logging hygiene
- 400% increase in users with access to the observability tooling and a reduction of the onboarding time from *days to minutes* due to LDAP integration
- Global ability to catalog and map service dependencies based upon request flow

Q + A