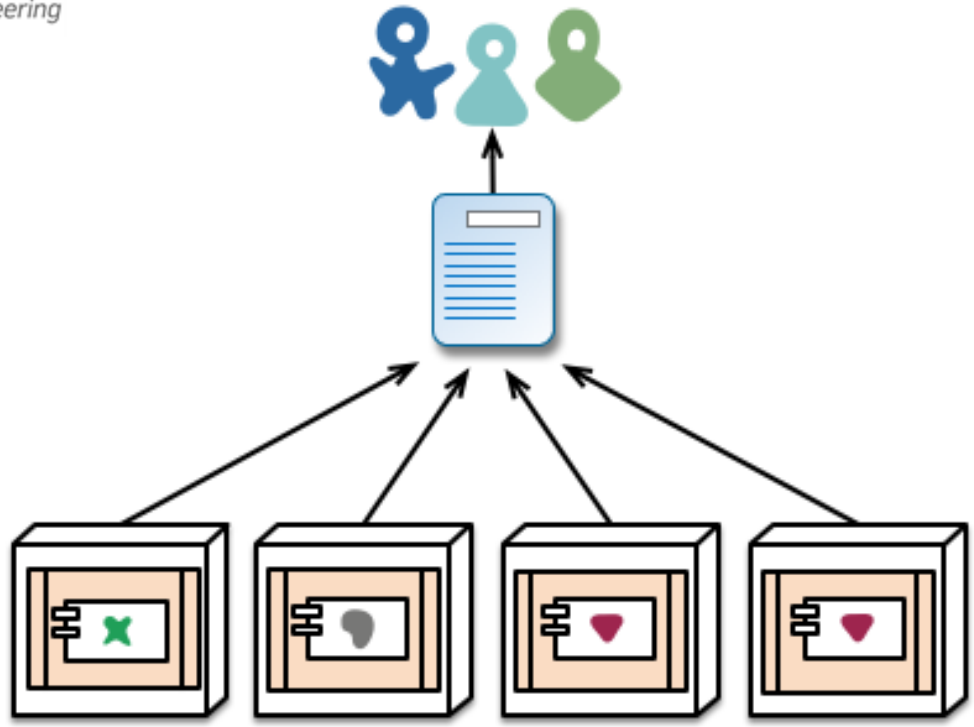


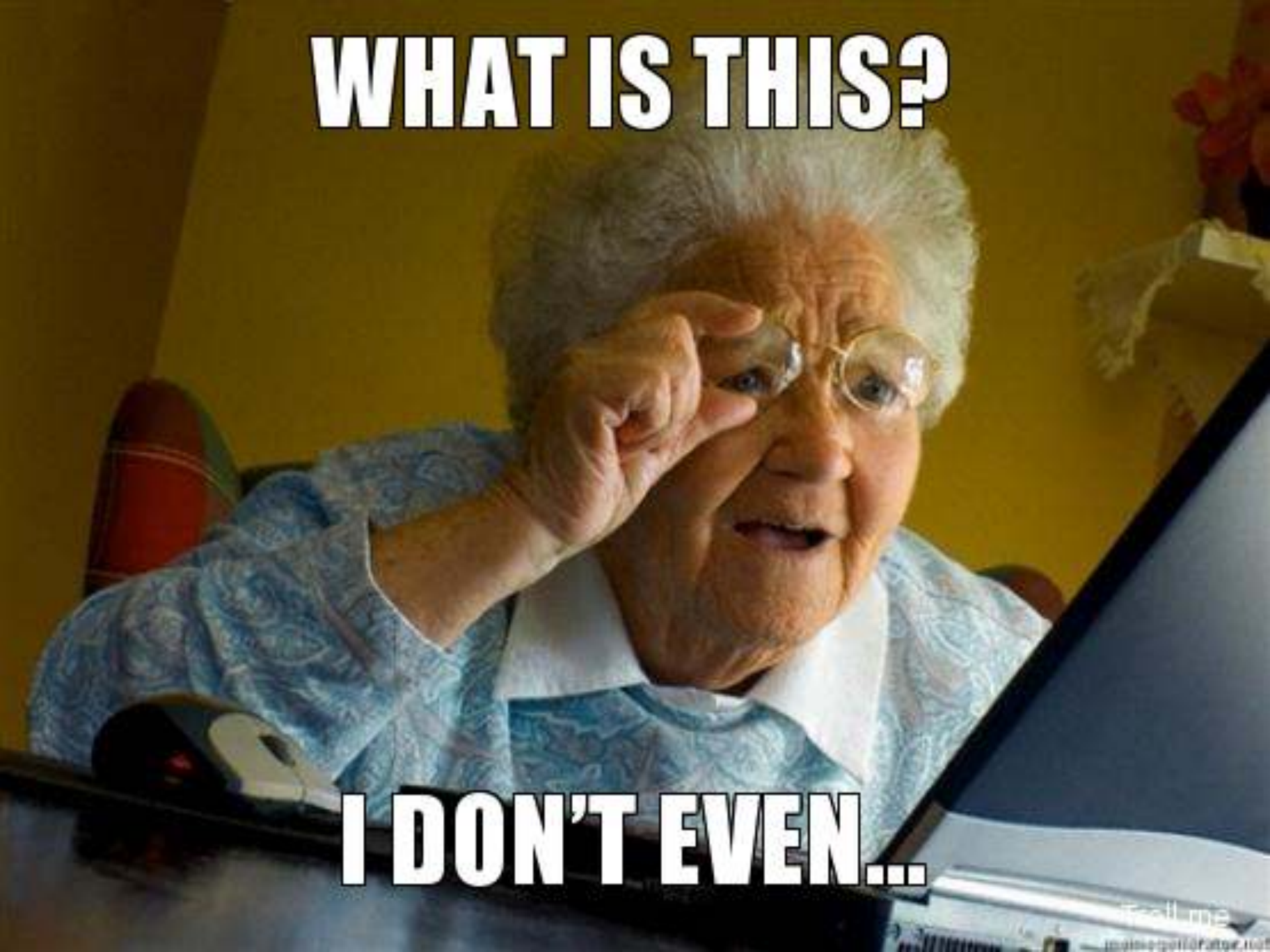
µServices Architecture

- SA at EPAM Systems
- primary skill is **Java**
- hands-on-coding with **Groovy**, **Ruby**
- trying to learn some **Scala** and **Erlang**
- passionate about **agile**, **clean code** and **devops**

Izzet Mustafayev@EPAM Systems
@webdizz webdizz izzetmustafaiev
http://webdizz.name

Agenda

- What is this?
- Architecture
- Case study
- Existing tools
- Summary
- How to get there?
- Q&A


WHAT IS THIS?
I DON'T EVEN...

Microservices

The term "Microservice Architecture" has sprung up over the last few years to describe a particular way of designing software applications as suites of independently deployable services. While there is no precise definition of this architectural style, there are certain common characteristics around organization around business capability, automated deployment, intelligence in the endpoints, and decentralized control of languages and data.

---

25 March 2014

James Lewis

James Lewis is a Principal Consultant at ThoughtWorks and member of the Technology Advisory Board. James' interest in building applications out of small collaborating services stems from a background in integrating enterprise systems at scale. He's built a number of systems using microservices and has been an active participant in the growing community for a couple of years.

Martin Fowler

Martin Fowler is an author, speaker, and general loud-mouth on software development. He's long been puzzled by the problem of how componentize software systems, having heard more vague claims than he's happy with. He hopes that microservices will live up to the early promise its advocates have found.

Overview

- **Small** (10-100 LOC)

Overview

- **Small** (10-100 LOC)
- **Lightweight** (run several per box)

Overview

- **Small** (10-100 LOC)
- **Lightweight** (run several per box)
- Takes **strength** of platform/language (**polyglot**)

Overview

- **Small** (10-100 LOC)
- **Lightweight** (run several per box)
- Takes **strength** of platform/language (**polyglot**)
- **Independent** (development/deployment)

Overview

- **Small** (10-100 LOC)
- **Lightweight** (run several per box)
- Takes **strength** of platform/language (**polyglot**)
- **Independent** (development/deployment)
- **Stateless** (everything persisted in DB)

Overview

- **Small** (10-100 LOC)
- **Lightweight** (run several per box)
- Takes **strength** of platform/language (**polyglot**)
- **Independent** (development/deployment)
- **Stateless** (everything persisted in DB)
- **Monitored** (health and business value)

Architecture

- Design
- Implementation
- Delivery
- Monitoring

Design

- What ab SOA?
- Externalize Configuration
- Versioning
- Communication Contract
- Asynchronicity
- Reusability
- Scalability
- Availability

What ab SOA?

- Sensible idea
- Combat challenges of large monolithic applications
- Lack of guidance

Externalize Configuration

- Support for X > 1 environments
- Run-time changes

Versioning

- Leave multiple old microservice versions running
- Fast introduction vs. slow retirement asymmetry

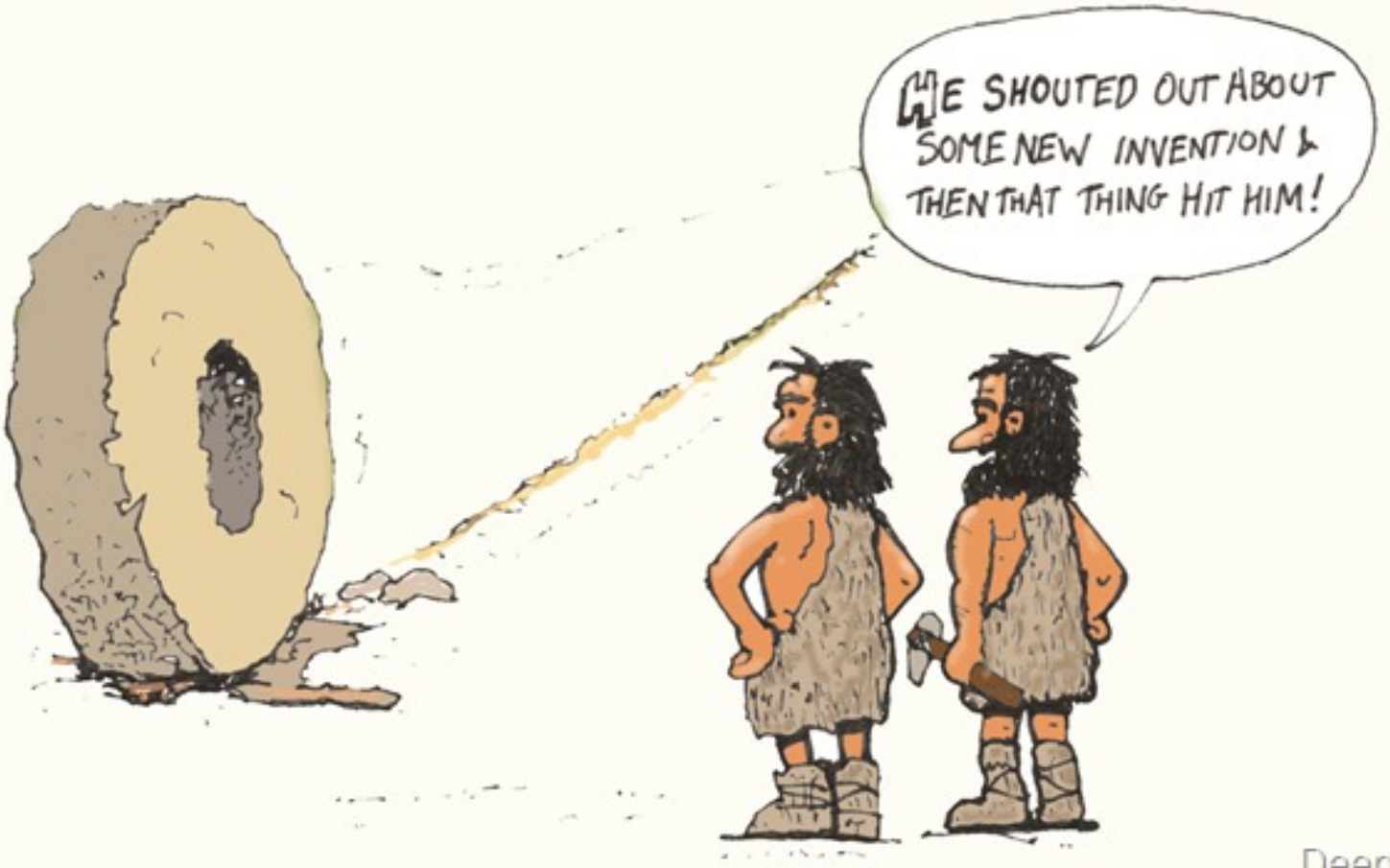
Communication Contract

- Own service - own client
- Thin transport

Asynchronicity

- Non-blocking
- Reactive model
- Messaging to send and forget

Reusability



- Each service does one dedicated thing
- Utilize suitable components/frameworks

Scalability

- Cohesive
- Stateless
- Independent

Availability

- Fault-tolerance
- Introduce failures

Implementation

- Separate Concerns

Separate Concerns

- Inverse Conway's law – teams own service groups

Separate Concerns

- Inverse Conway's law – teams own service groups
- One "verb" per single function microservice

Separate Concerns

- Inverse Conway's law – teams own service groups
- One "verb" per single function microservice
- One developer independently produces a microservice

Separate Concerns

- Inverse Conway's law – teams own service groups
- One "verb" per single function microservice
- One developer independently produces a microservice
- Each microservice is it's own build

Separate Concerns

- Inverse Conway's law – teams own service groups
- One "verb" per single function microservice
- One developer independently produces a microservice
- Each microservice is it's own build
- Stateful cached data access layer

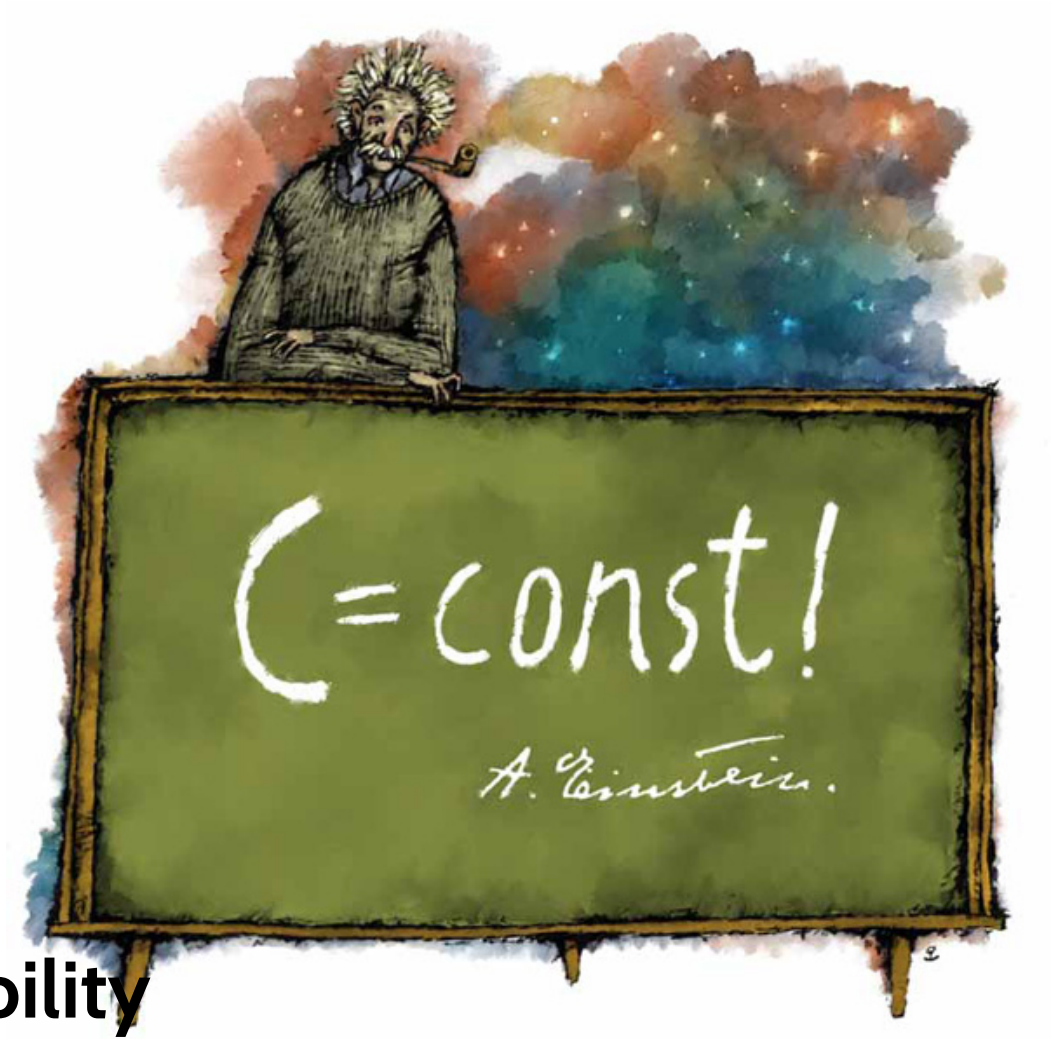
Separate Concerns

- Inverse Conway's law – teams own service groups
- One "verb" per single function microservice
- One developer independently produces a microservice
- Each microservice is it's own build
- Stateful cached data access layer
- Lightweight mocked dependencies

Separate Concerns

- Inverse Conway's law – teams own service groups
- One "verb" per single function microservice
- One developer independently produces a microservice
- Each microservice is it's own build
- Stateful cached data access layer
- Lightweight mocked dependencies
- Easy to run locally

Delivery



- Disposability
- Containerisation

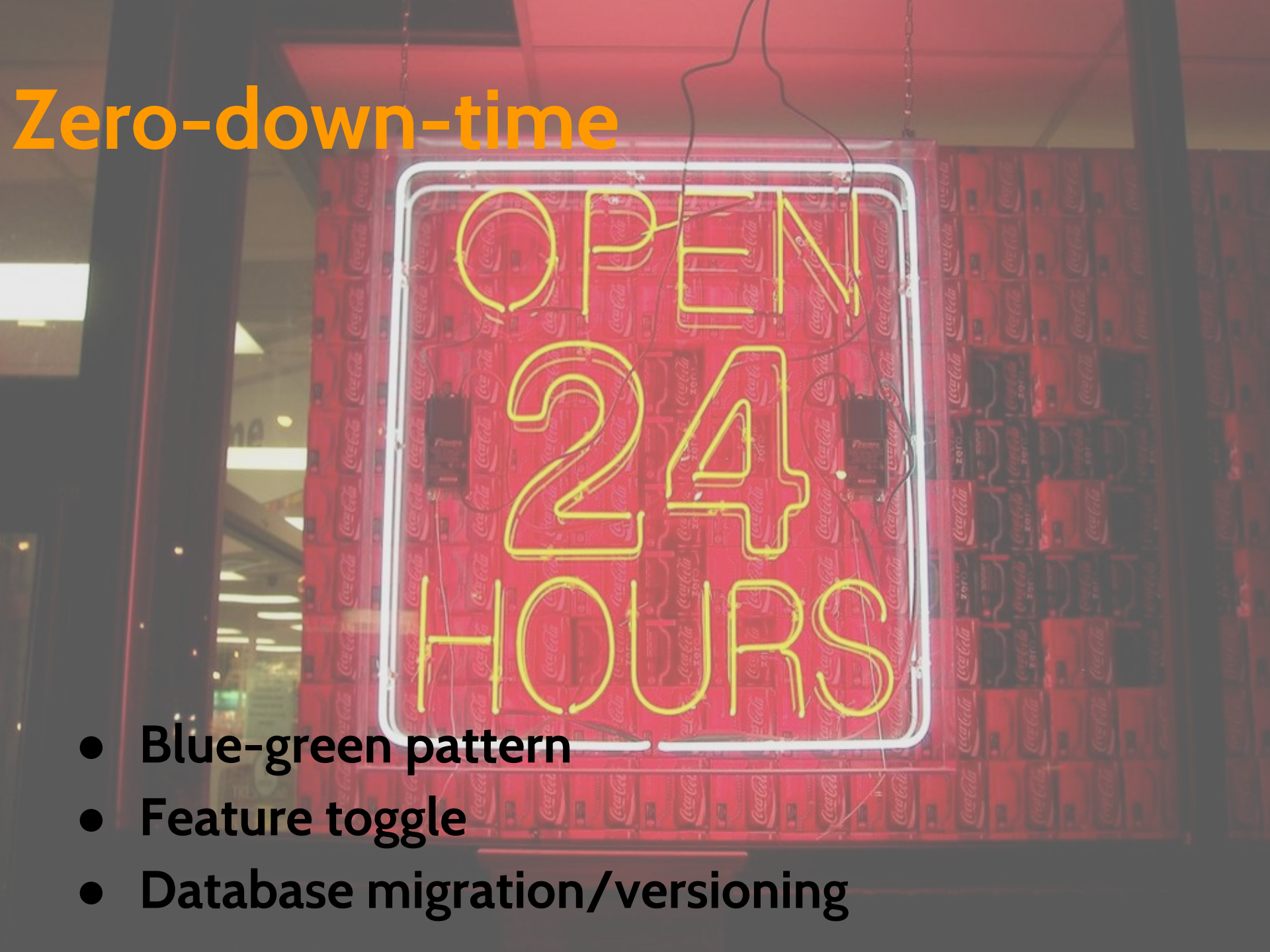
Disposability

- Build
- Run
- Destroy

Containerisation

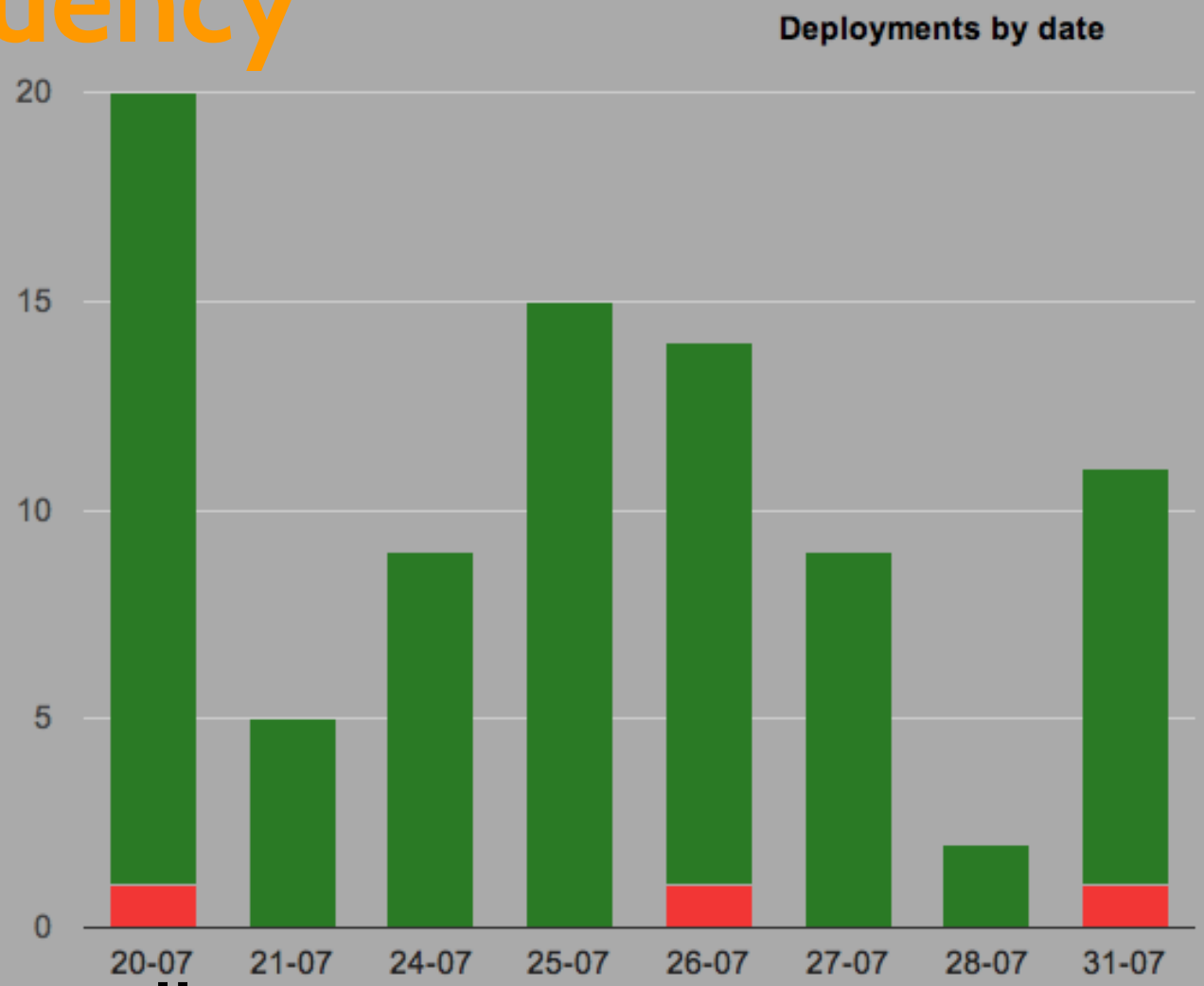
- Container as deployment artifact
- Environment agnostic
- New version - new container
- All dependencies built in

Zero-down-time



- Blue-green pattern
- Feature toggle
- Database migration/versioning

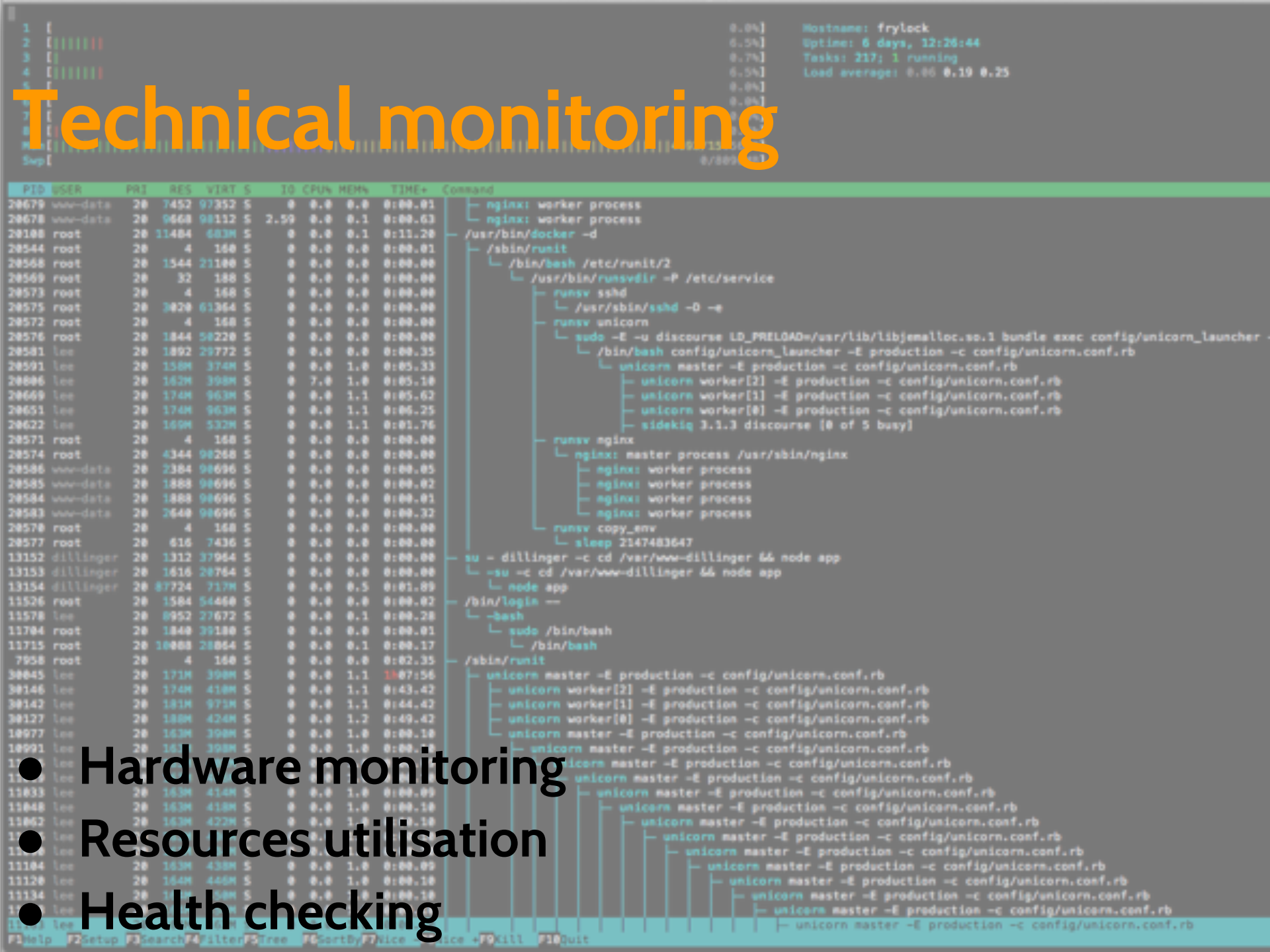
Frequency



- Fast rollouts
- Fast rollbacks

Monitoring

- Technical
- Business
- Stats


Technical monitoring
Hostname: frylock
Uptime: 6 days, 12:26:44
Tasks: 217; 1 running
Load average: 0.06 0.19 0.25
Hardware monitoring
Resources utilisation
Health checking

Business monitoring

- Business aware metrics
- A/B testing metrics

Stats matter on scale

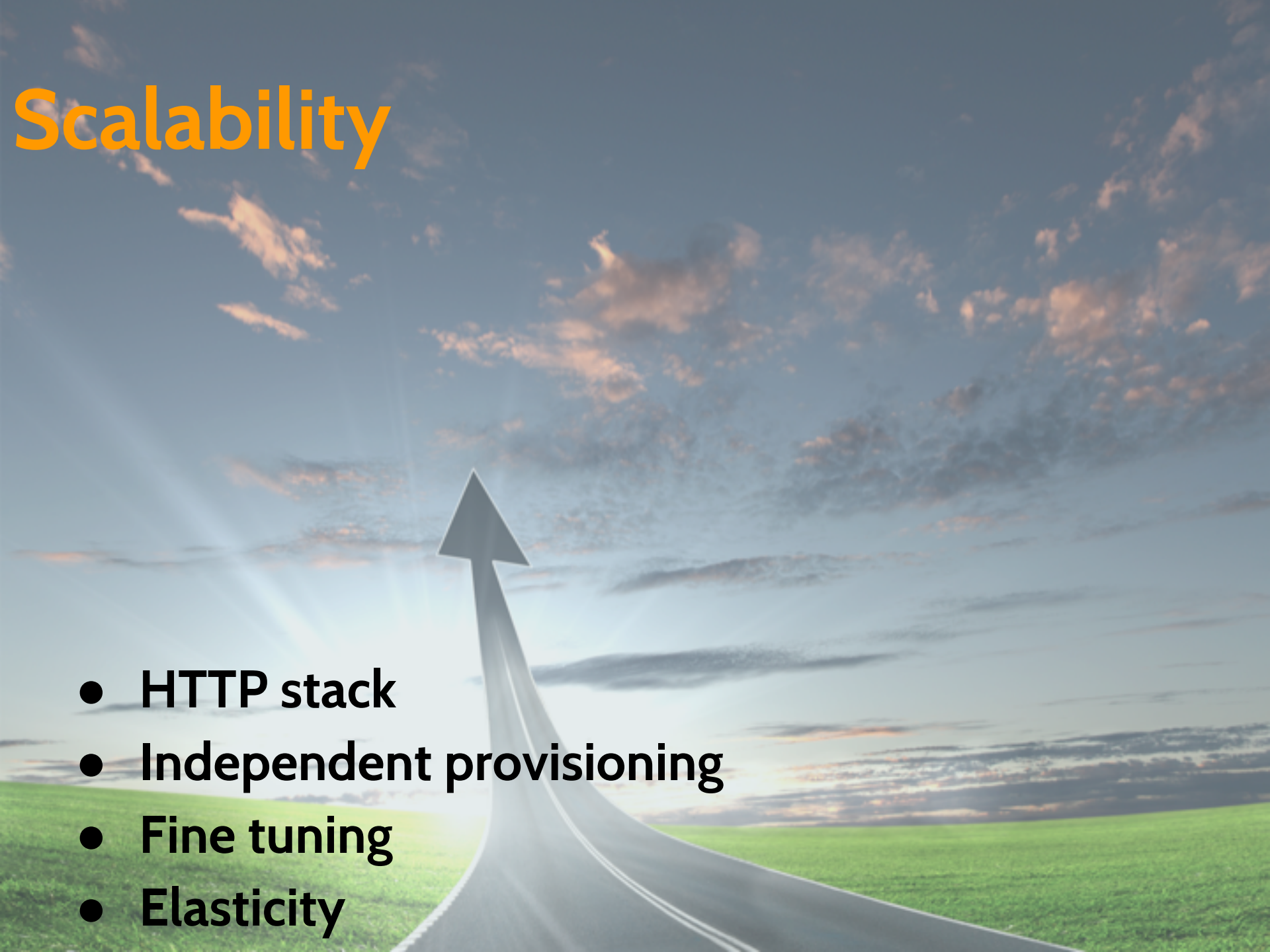
- Increased traffic velocity makes individual events less important
- More vertical components requires more measurement
- Failure Rate is more informative than individual failures

Benefits

Toolset unchained

- Polyglot technology stack
- Polyglot persistence
- Frameworks
- Thin transport

Scalability



- HTTP stack
- Independent provisioning
- Fine tuning
- Elasticity

Independence

- Development
- Testing
- Deployment
- Reliability

Shortcomings

Shortcomings

- **More responsibility** from **Devs** to support **Ops**

Shortcomings

- **More responsibility** from **Devs** to support **Ops**
- **Polyglot infrastructure** (if any)

Shortcomings

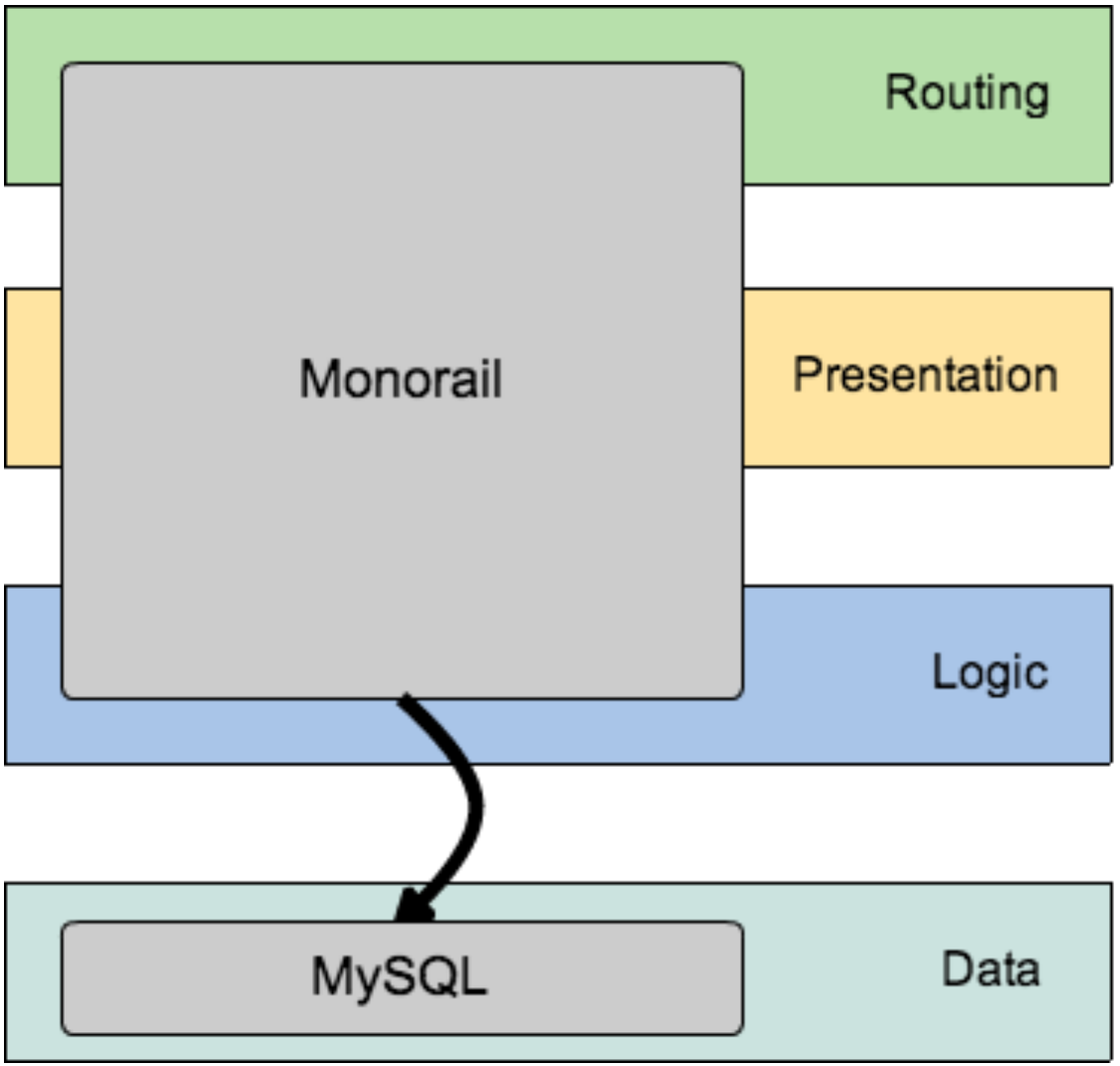
- **More responsibility** from **Devs** to support **Ops**
- **Polyglot infrastructure** (if any)
- **Orchestration**

Shortcomings

- **More responsibility** from **Devs** to support **Ops**
- **Polyglot infrastructure** (if any)
- **Orchestration**
- **Costs**


case
study

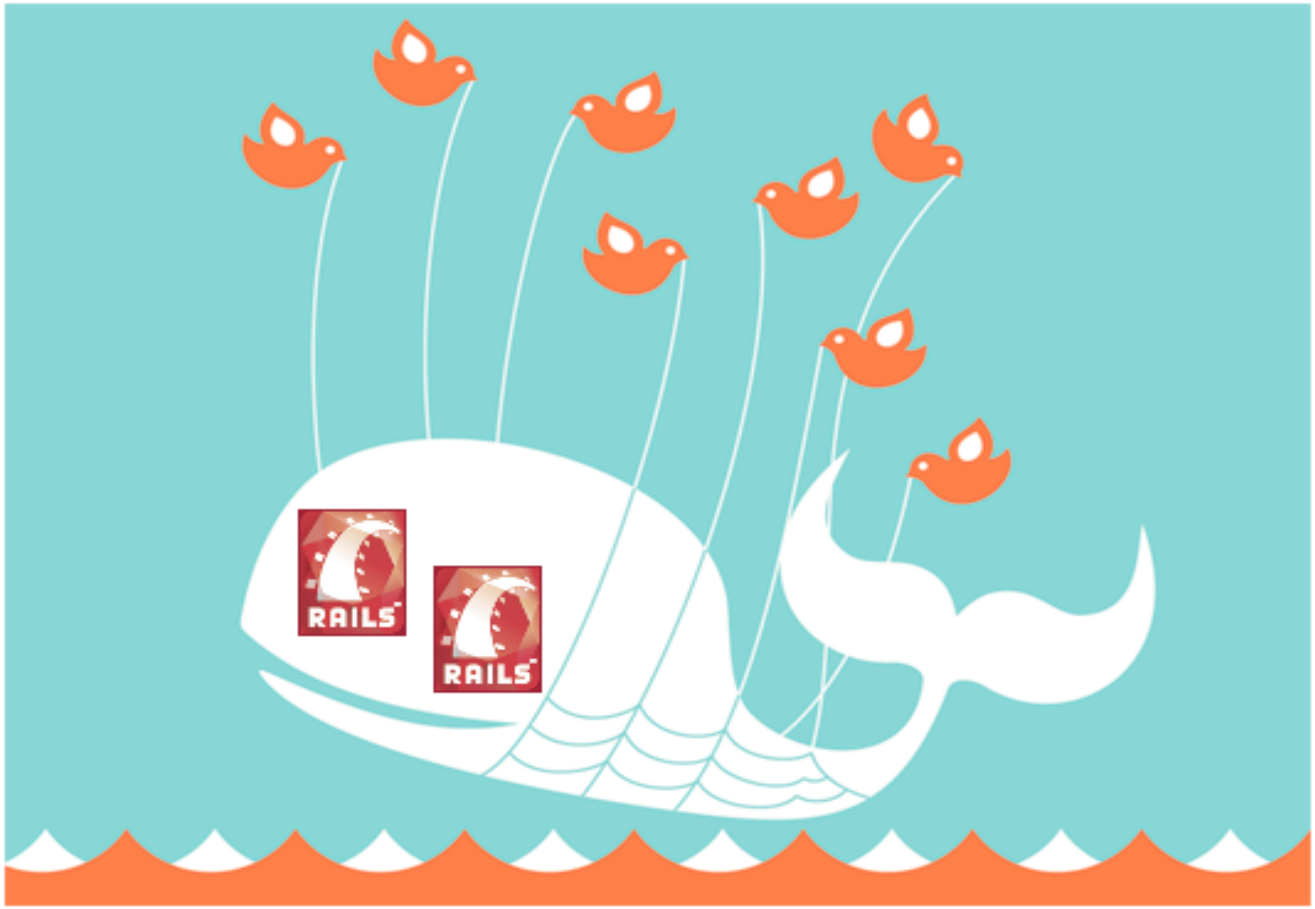
Twitter before



twitter

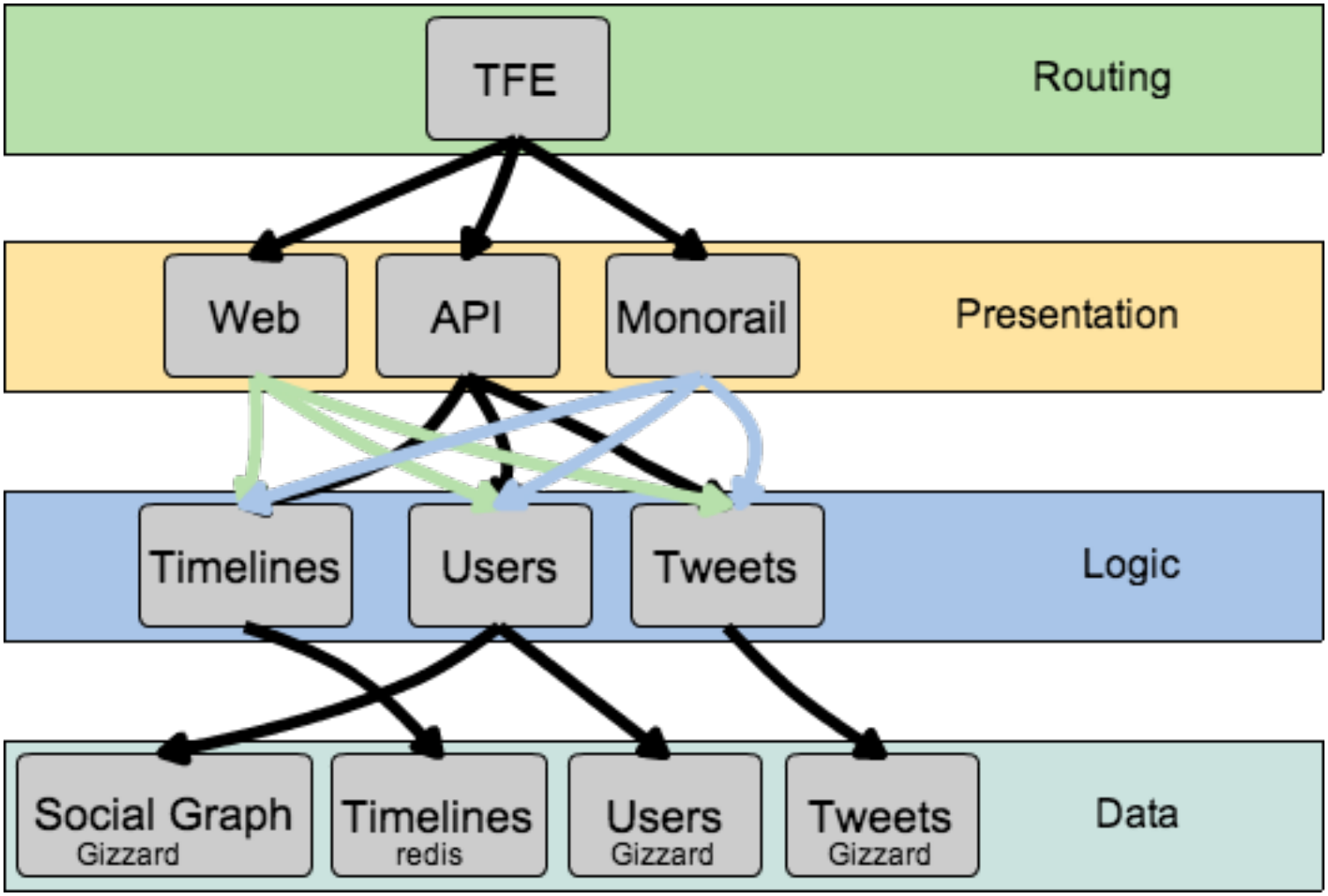
Twitter is currently down for temporary maintenance.

We expect to be back in 30 minutes. Thanks for your patience.

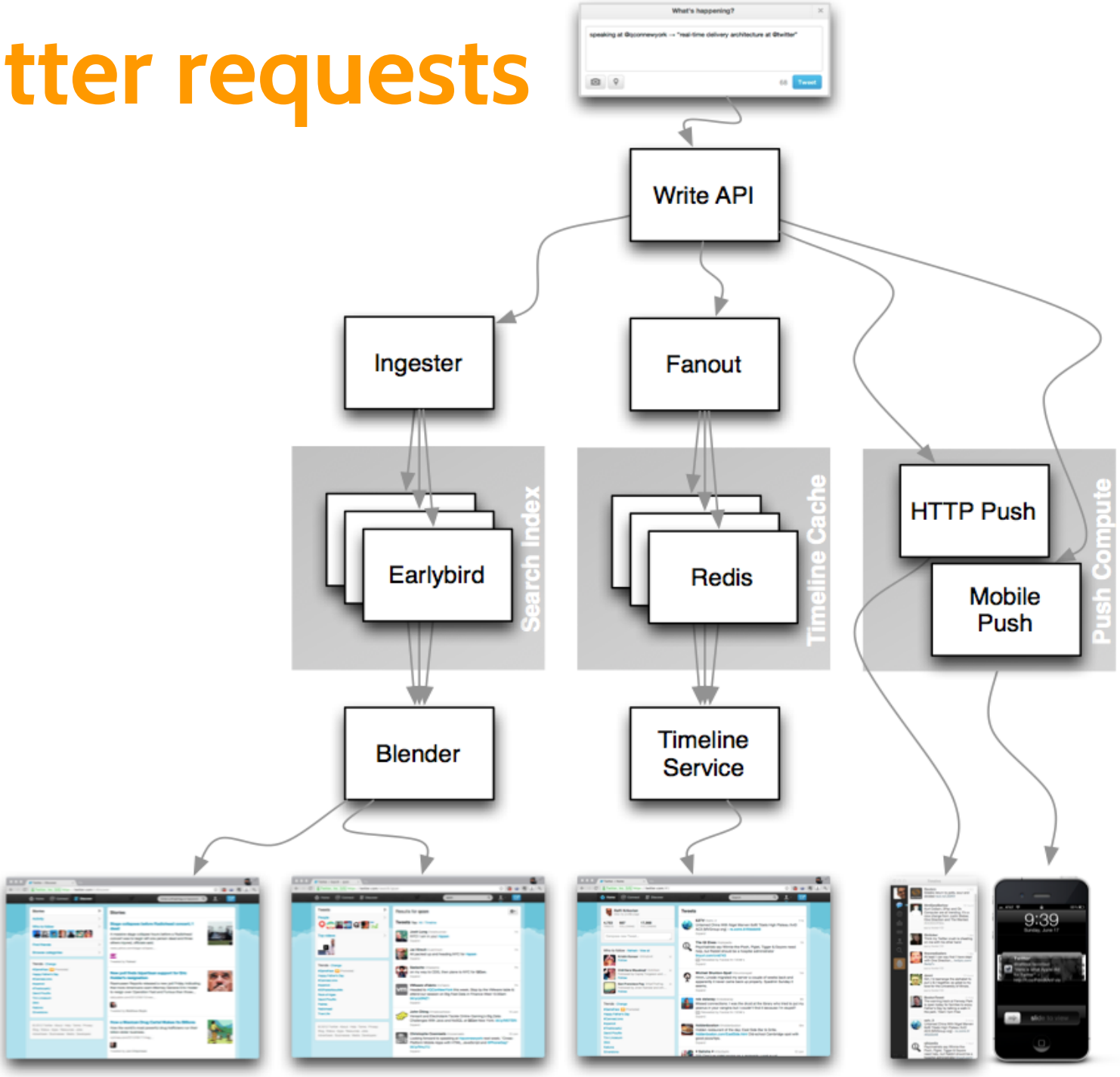


 Blog Help

Twitter now*


TFE
Routing
Web
API
Monorail
Presentation
Timelines
Users
Tweets
Logic
Social Graph
Gizzard
Timelines
redis
Users
Gizzard
Tweets
Gizzard
Data

Twitter requests



Lessons learned

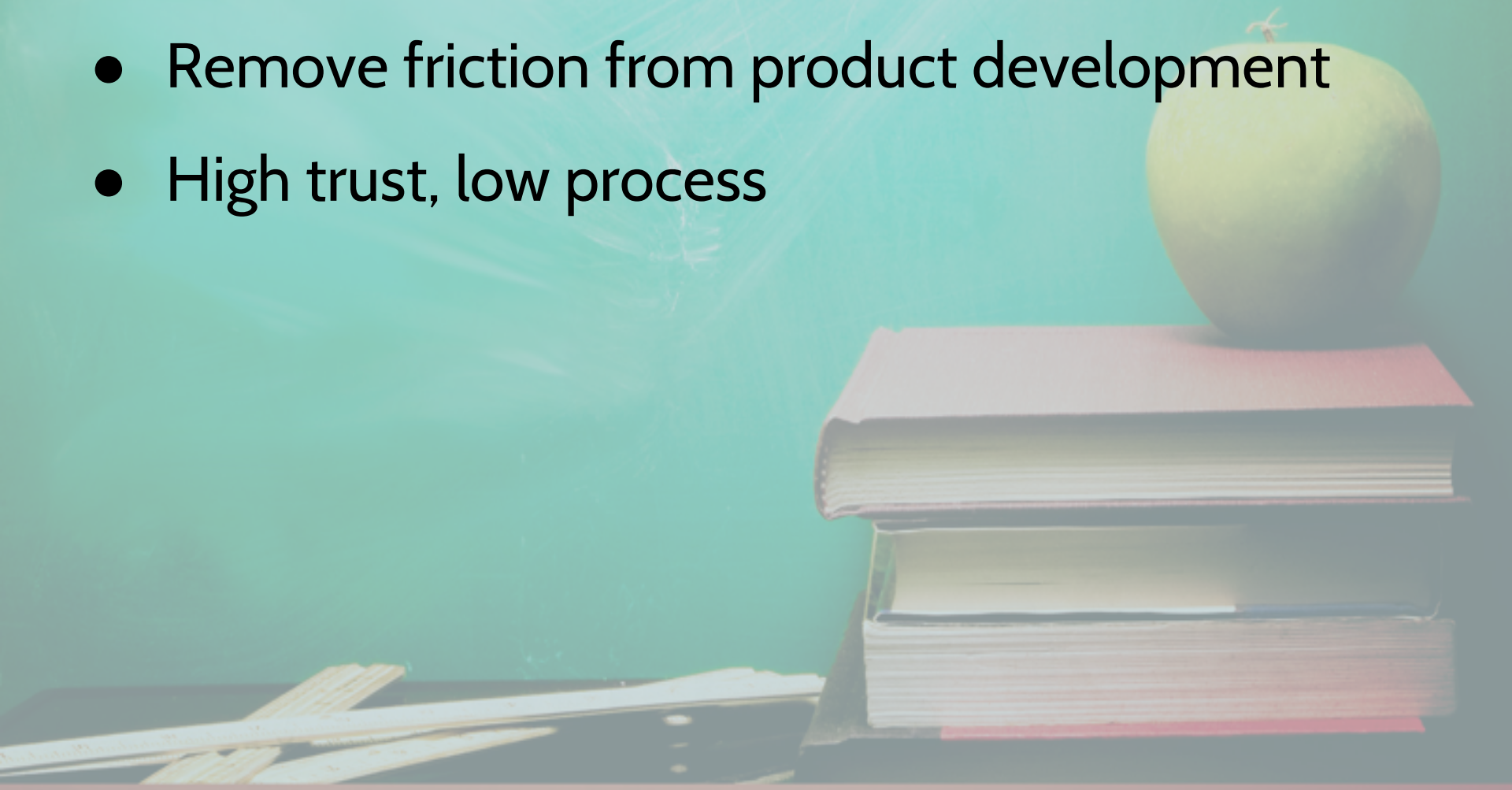
- Speed wins in the marketplace

Lessons learned

- Speed wins in the marketplace
- Remove friction from product development

Lessons learned

- Speed wins in the marketplace
- Remove friction from product development
- High trust, low process

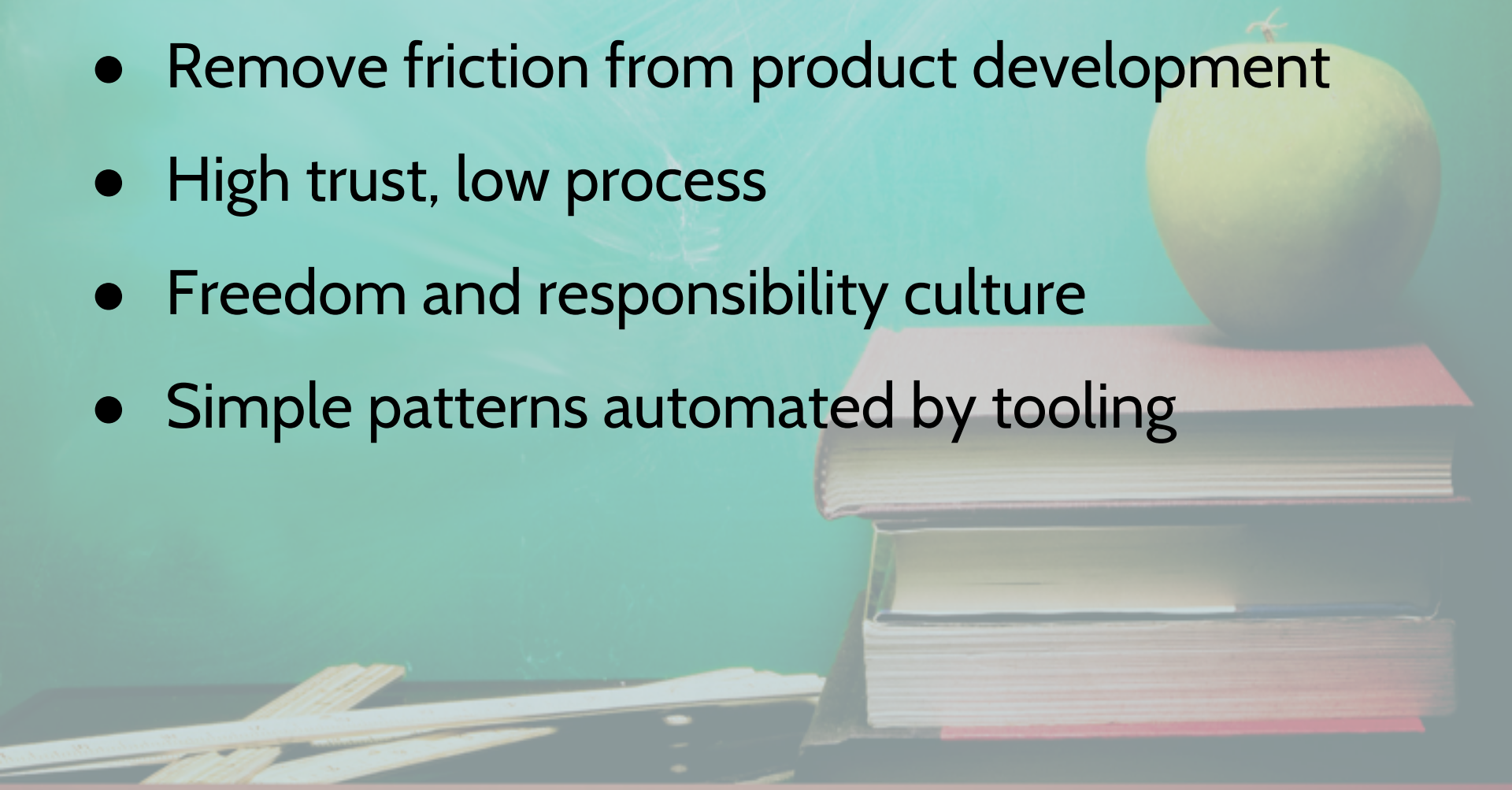


Lessons learned

- Speed wins in the marketplace
- Remove friction from product development
- High trust, low process
- Freedom and responsibility culture

Lessons learned

- Speed wins in the marketplace
- Remove friction from product development
- High trust, low process
- Freedom and responsibility culture
- Simple patterns automated by tooling



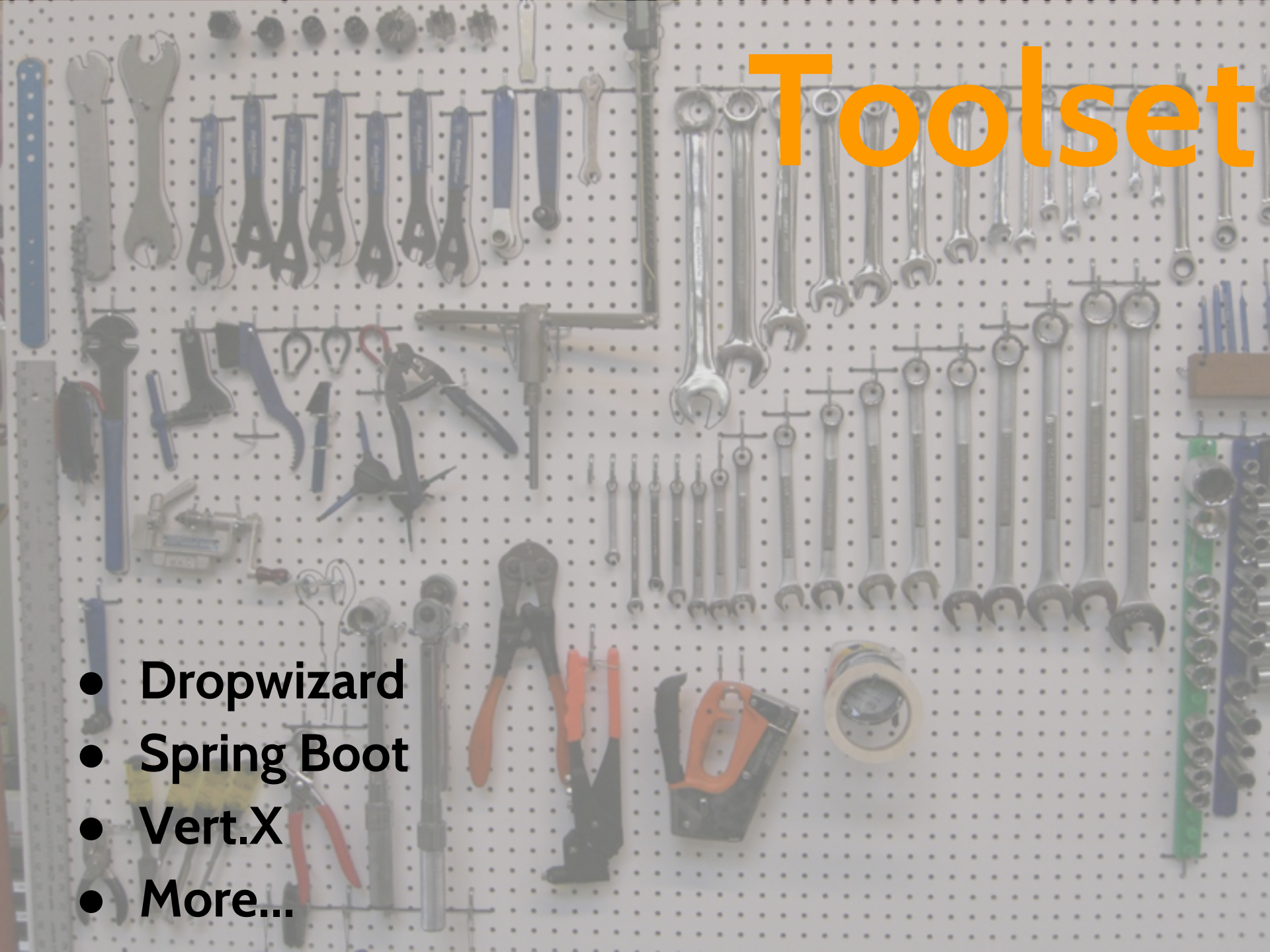
Lessons learned

- Speed wins in the marketplace
- Remove friction from product development
- High trust, low process
- Freedom and responsibility culture
- Simple patterns automated by tooling
- Microservices for speed and availability

Lessons learned

- Speed wins in the marketplace
- Remove friction from product development
- High trust, low process
- Freedom and responsibility culture
- Simple patterns automated by tooling
- Microservices for speed and availability
- Use statistics to monitor behavior

Toolset



- Dropwizard
- Spring Boot
- Vert.X
- More...

Dropwizard http://dropwizard.github.io/dropwizard/

Java framework for developing ops-friendly, high-performance, RESTful web services.

- pulls together **stable**, mature **libraries** from the Java ecosystem into a simple, **light-weight** package
- has out-of-the-box support for sophisticated **configuration**, application **metrics**, **logging**, **operational tools**, and much more

Spring Boot http://projects.spring.io/spring-boot/

Takes an opinionated view of building production-ready Spring applications.

- favors convention over configuration and is designed to get you up and running as quickly as possible.

- production-ready features such as metrics, health checks and externalized configuration

Vert.X http://vertx.io/

- Lightweight, reactive, application platform
- Superficially similar to Node.js - but not a clone!
- Inspired also from Erlang/OTP
- Polyglot
- High performance
- Simple but not simplistic

More...

- Ratpack http://www.ratpack.io/
- Sinatra http://www.sinatrarb.com/
- Webbit https://github.com/webbit/webbit
- Finagle http://twitter.github.io/finagle/
- Connect http://www.senchalabs.org/connect/
- XDropWizard https://github.com/timmolter/XDropWizard

Summary

Summary

- Speed wins in the marketplace

Summary

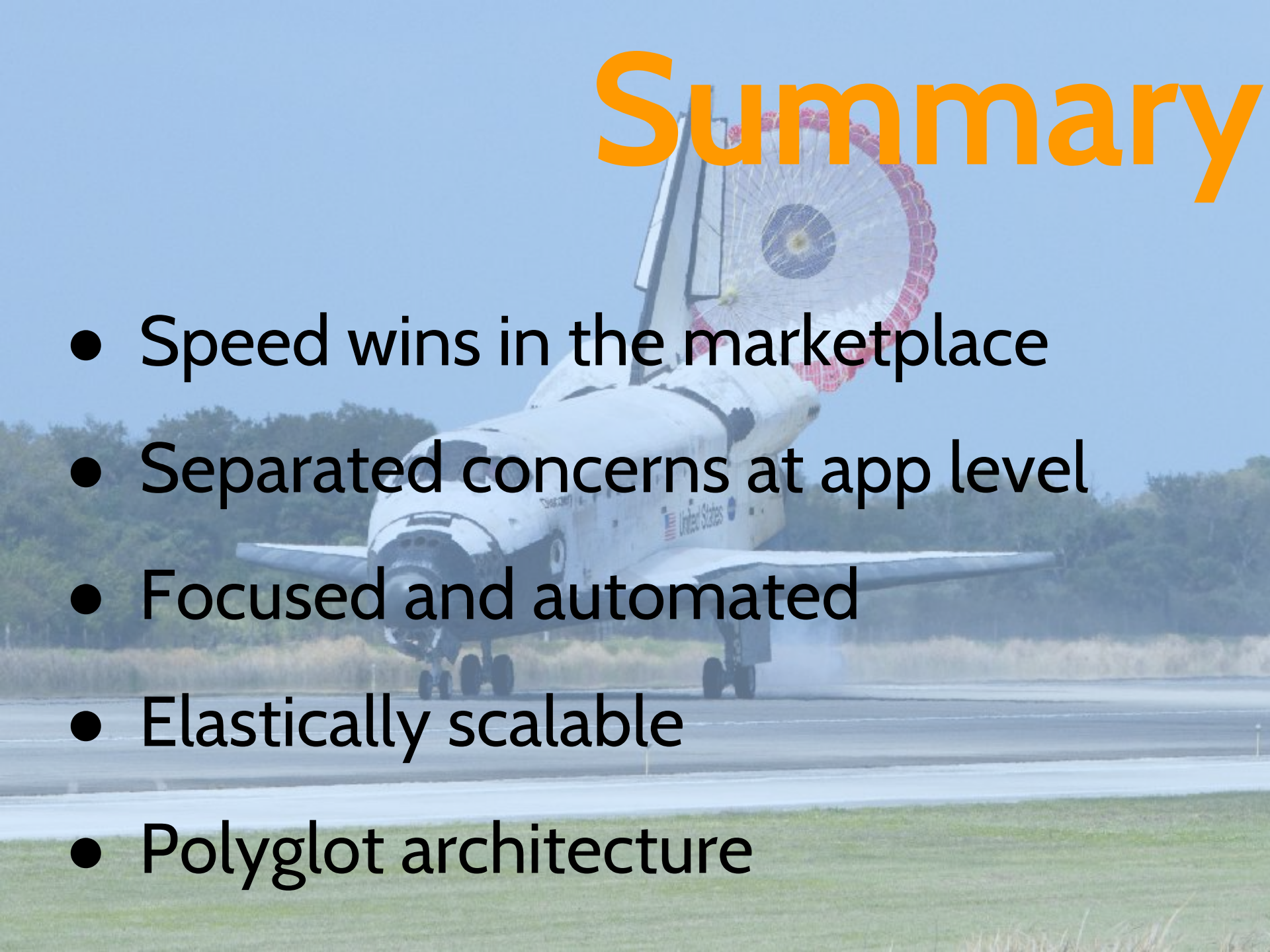
- Speed wins in the marketplace
- Separated concerns at app level

Summary

- Speed wins in the marketplace
- Separated concerns at app level
- Focused and automated

Summary

- Speed wins in the marketplace
- Separated concerns at app level
- Focused and automated
- Elastically scalable

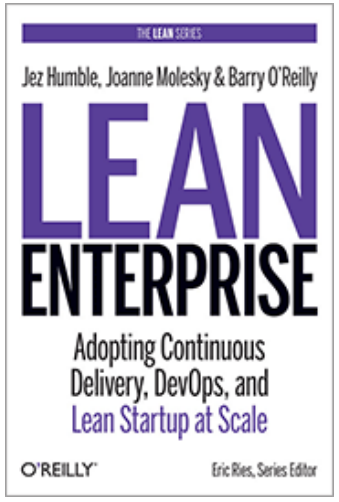


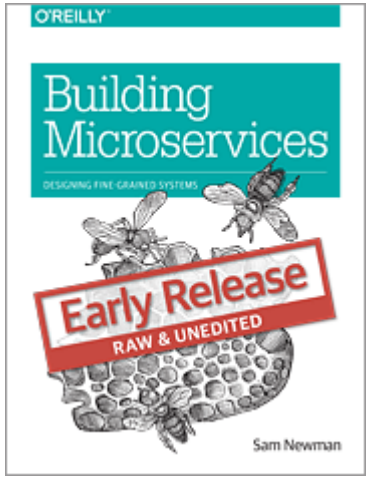
Summary

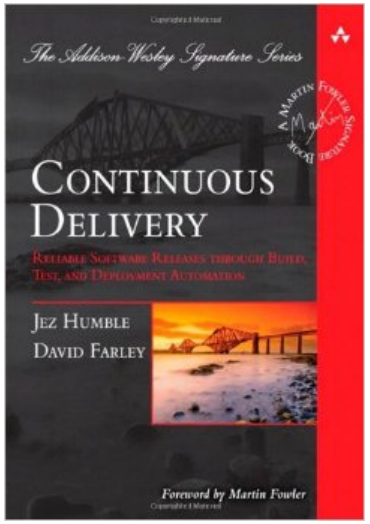
- Speed wins in the marketplace
- Separated concerns at app level
- Focused and automated
- Elastically scalable
- Polyglot architecture

How to get there?

- Automate repeating activities
- "Run what you wrote" – root access and duty
- Freedom and responsibility for developers







References

- http://martinfowler.com/articles/microservices.html
- http://yobriefca.se/blog/2013/04/28/micro-service-architecture/
- http://goo.gl/4dOVjj
- http://goo.gl/RGgnd3
- http://goo.gl/TmZ4Ee
- http://12factor.net/
- http://goo.gl/tnPxLP

Q&A


FEEDBACK

Izzet Mustafayev@EPAM Systems
@webdizz webdizz izzetmustafaiev
http://webdizz.name