

MicroService Challenges

Fred George

fredgeorge@acm.org



@fgeorge52

Fred George

- **Programmer**

- Since 1968 (Basic)
- 65,000 hours experience
- 70+ languages
- Computer Science Degree 1973

- **Manager**

- 17 years IBM
- Business degree, MIT Sloan School 1986
- Product Owner, IBM
- VP, ThoughtWorks
- Co-founder, Outpace (Silicon Valley)
- Senior Advisor to 3 tech companies



- **Technologist**

- Computer networks – 70's
- Token Ring LAN – 80's
- GUI's – late 80's
- OO – late 80's
- Agile – late 90's
- MicroServices – mid-2000's

Fred George

Consulting Roles

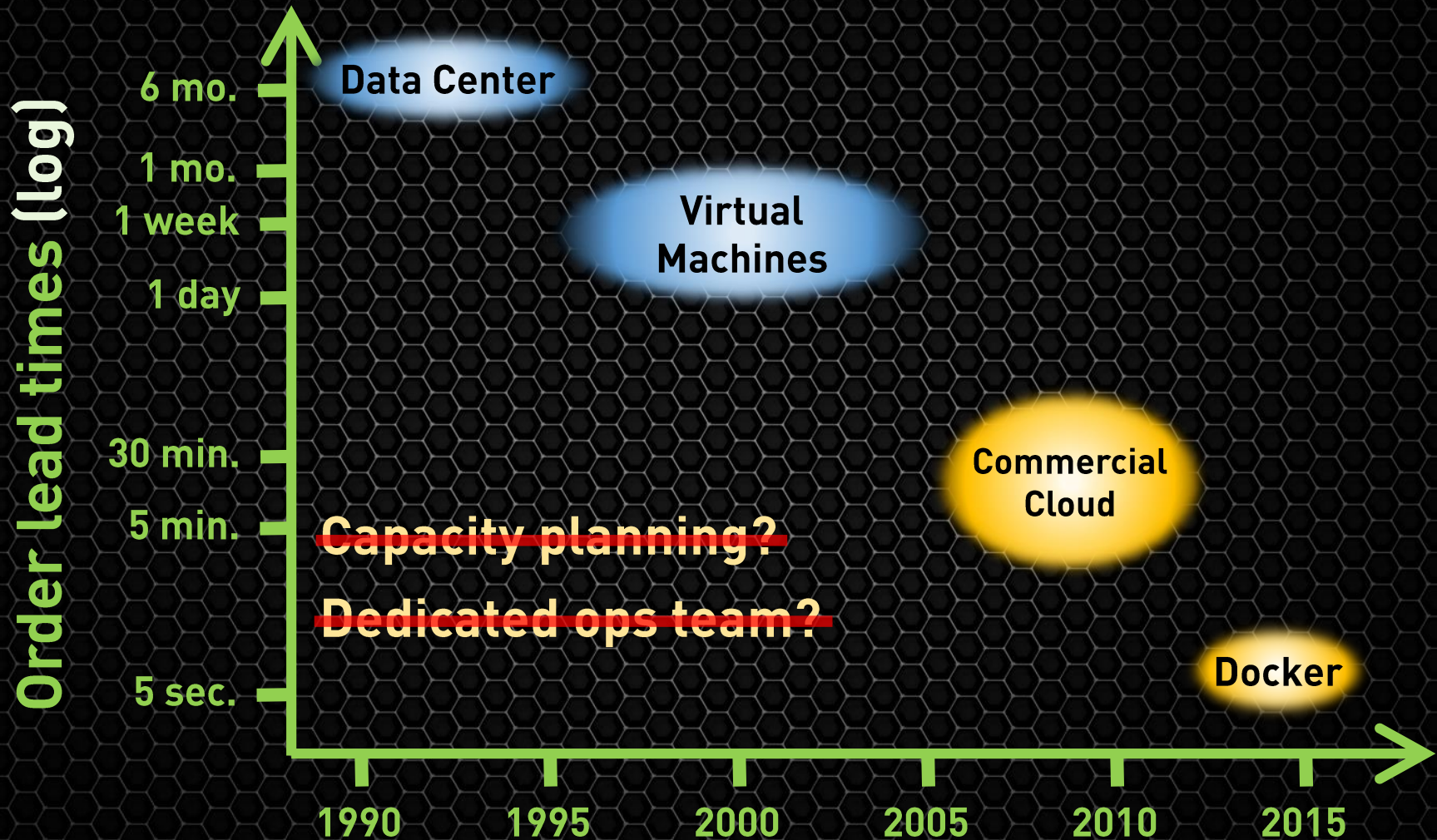
- Change Agent
- Disruptor
- “Hand grenade I am throwing into development”
 - CTO describing Fred to his Vice President



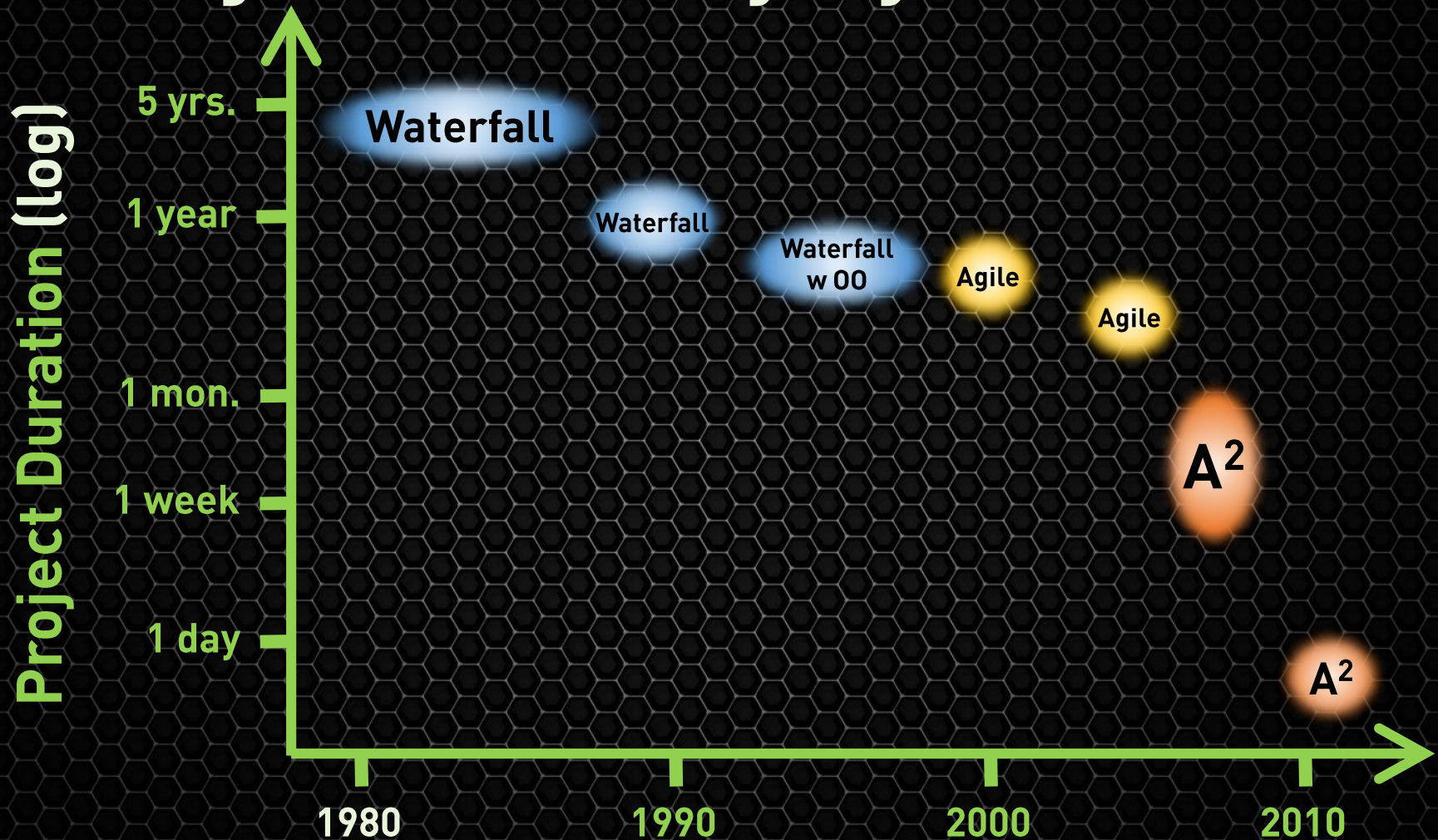
Why Now?



Hardware Lead Times



Project Delivery Cycles



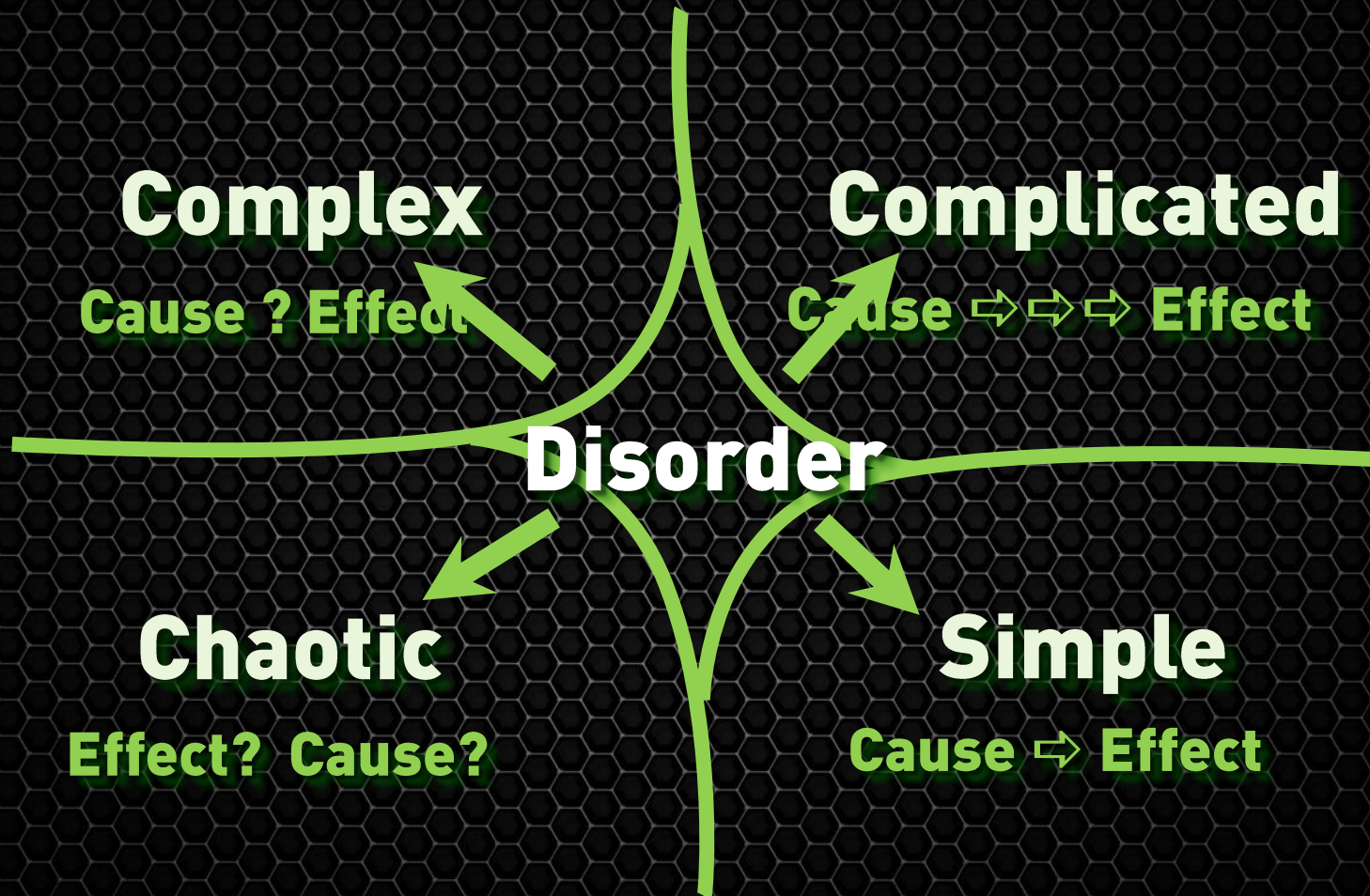
How Fast Can You Go?

We've Been Busy

In the last 7 days **53** developers have made **882** commits resulting in **652** deploys

Deployment to Production Every 3.5 minutes

Why: New Problem Domains: The **Cynefin** Framework



Why: Competition is coming

- **Enablement through technologies**
 - Cloud computing
 - Ubiquitous, high bandwidth
 - Languages (and supporting frameworks)
- **Recognition of business opportunities**
 - Silicon Valley innovators as role models
 - Accelerating business needs
 - Few inhibitors for global competitors
 - Reduction of entry barriers for niche competitors

Summary Principles of MicroServices

- Very, very small
- Team size of one to develop/maintain
- Loosely coupled (including flow)
- Multiple versions acceptable (encouraged?)
- Self-monitoring of each service
- Publish interesting “stuff” (w/o explicit requirements)
- “Application” seems to be a poor conceptualization

Common Theme of All Challenges:

Inexperience!

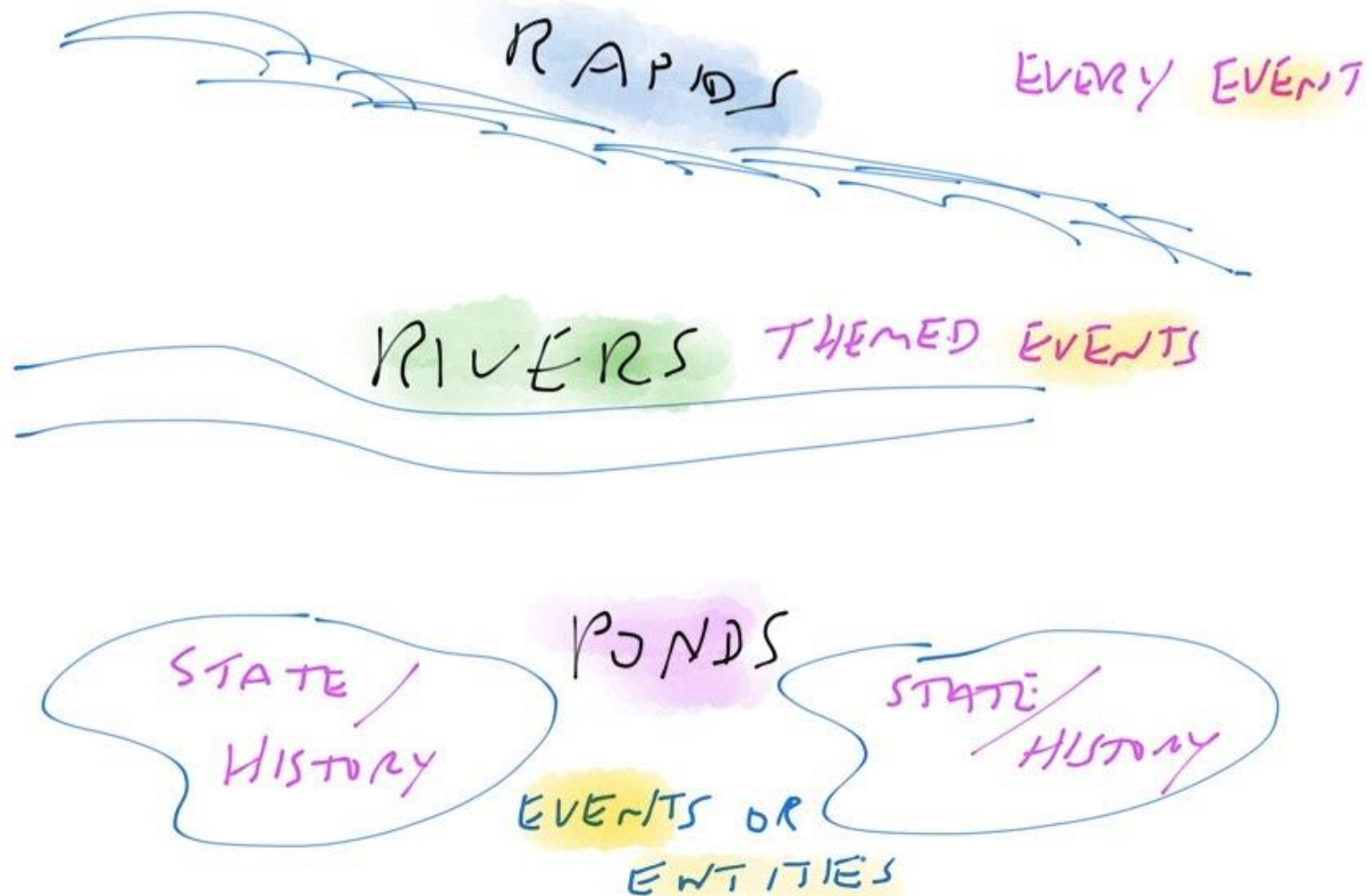
Challenge 1

Deconstructing the Databases

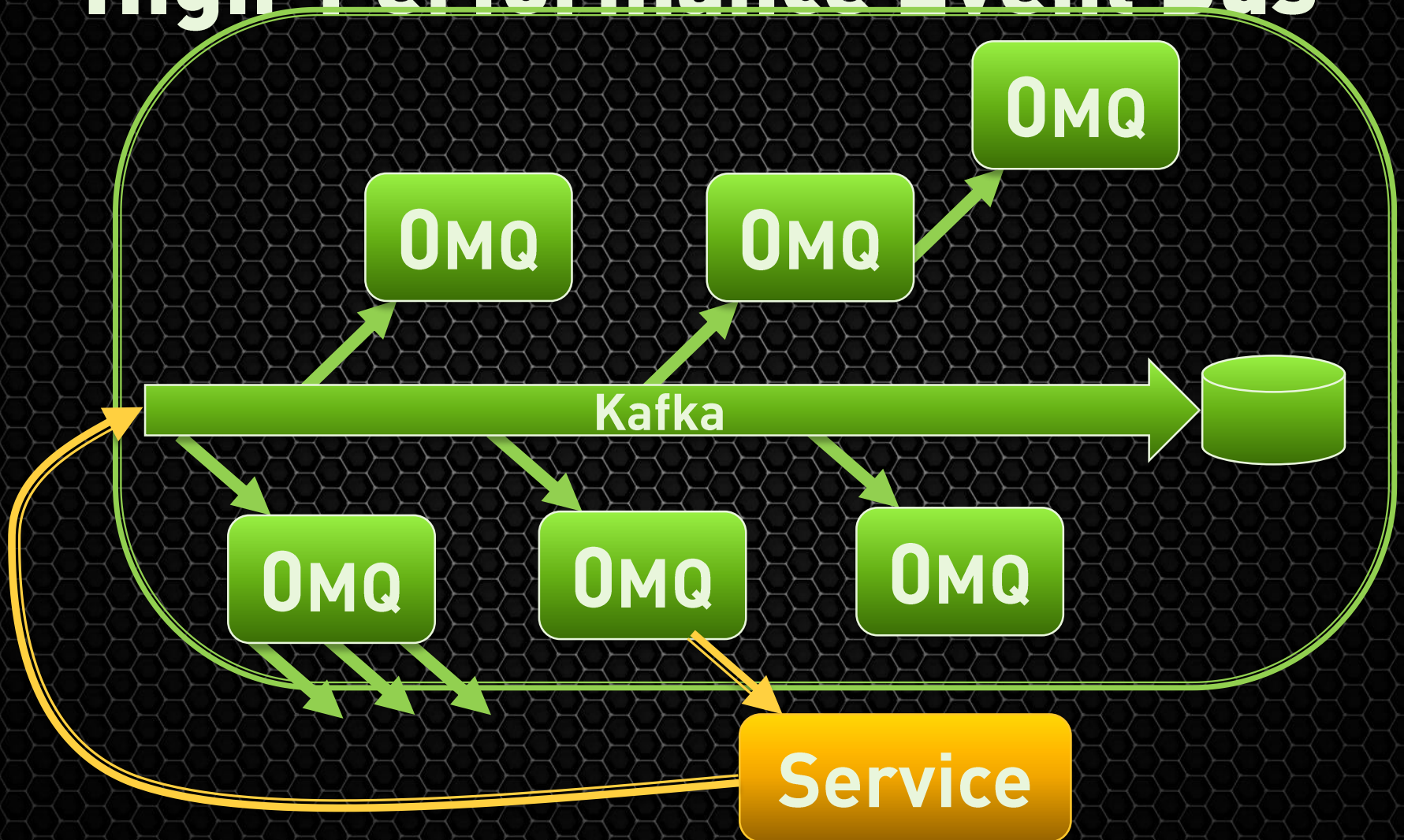
How Many Databases?

- **Fortune 100 view:**
 - Entity-oriented; consistency essential
 - As few as possible
- **MicroService view:**
 - Event bus replaces operational database
 - DB per MicroService (if persistence needed)
 - Poly-glot (various NoSQL, SQL)
 - Few (10%) writable; even fewer transactional

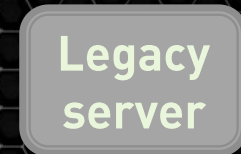
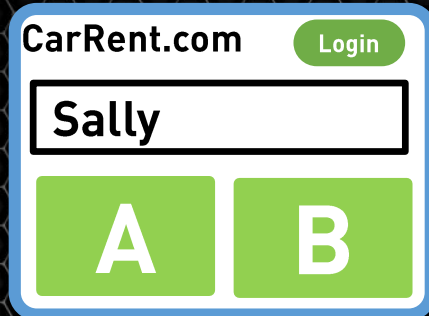
Rapids, Rivers, and Ponds



High-Performance Event Bus



Incremental Applications



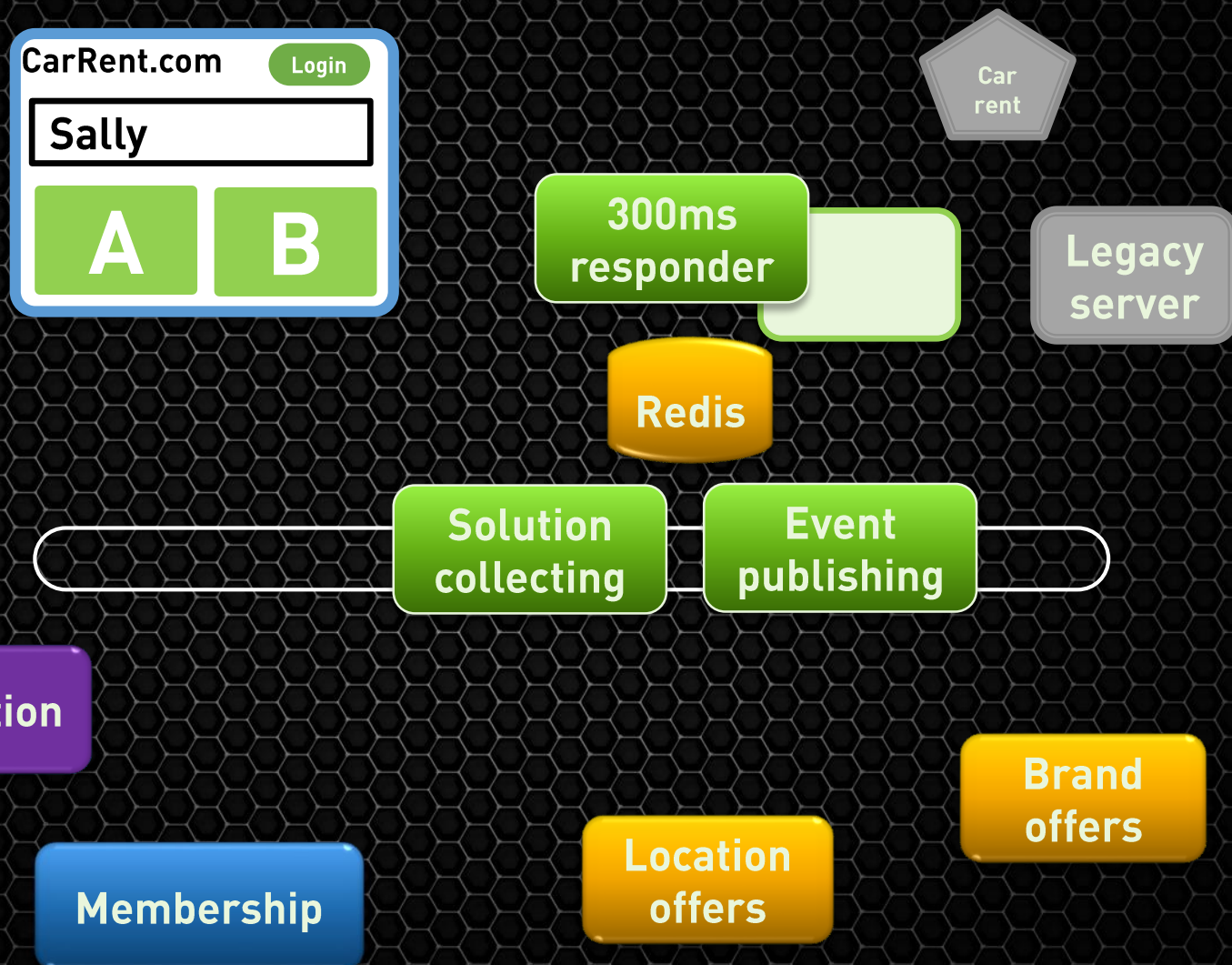
Event Bus

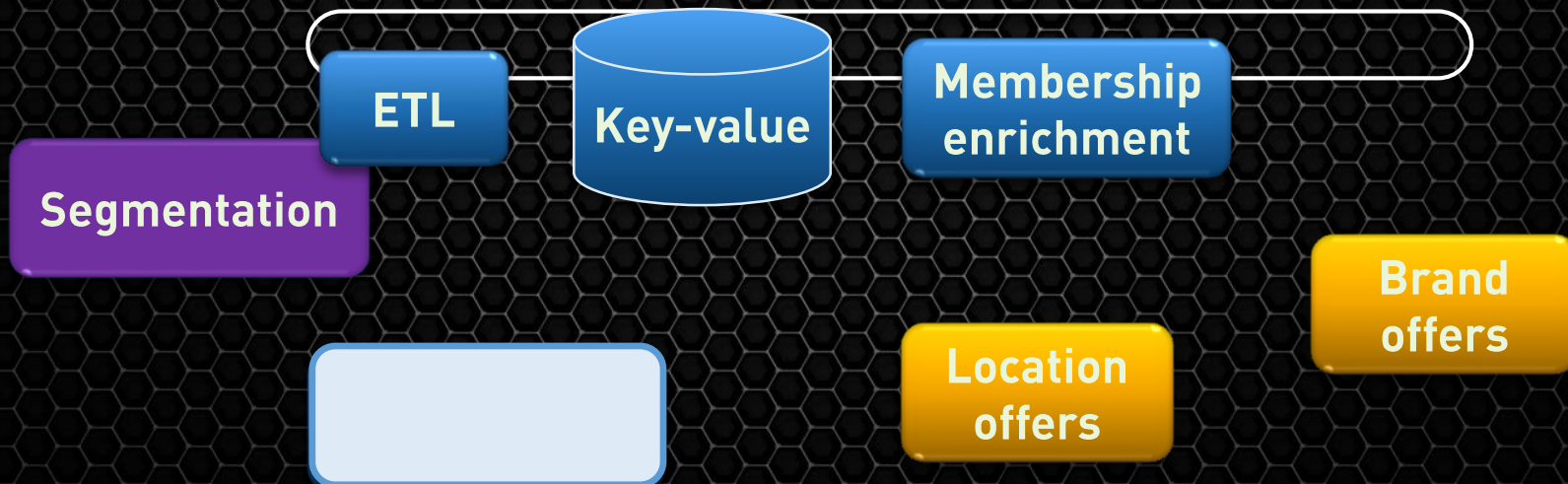
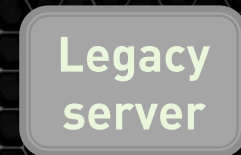
Segmentation

Membership

**Location
offers**

**Brand
offers**





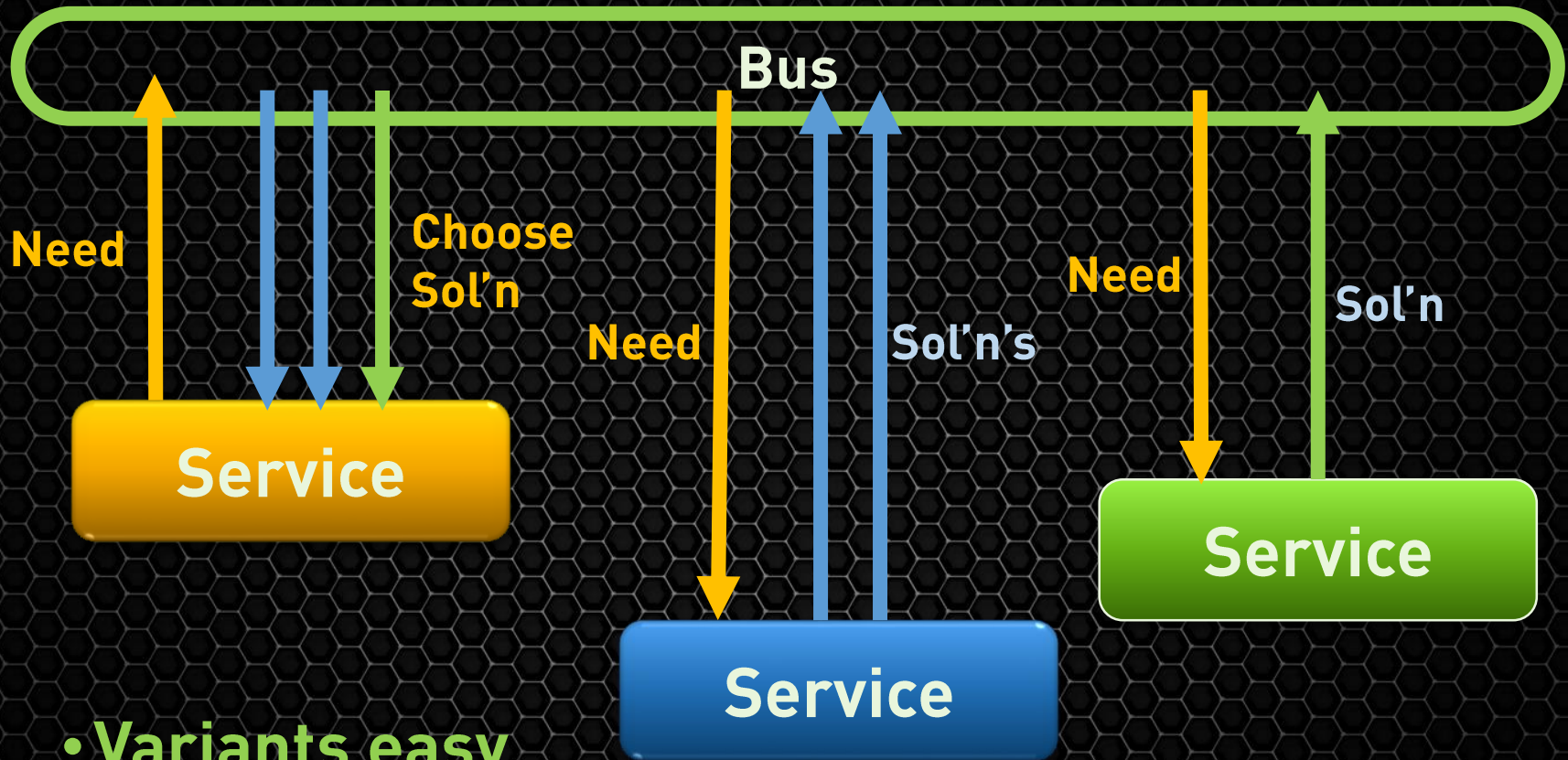
Challenge 2

**Synchronous or
Asynchronous**

Chad Fowler **vs.** Fred George

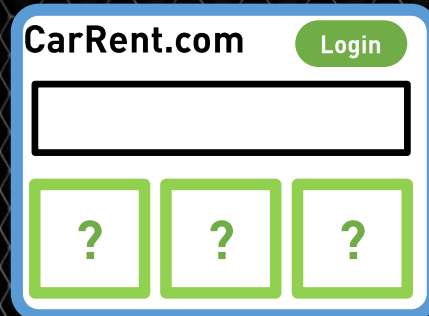
- **Chad: Use Synchronous as default**
 - Algorithms typically described serially
 - Programmer understanding / productivity
- **Fred: Use Asynchronous as default**
 - Robustness should be primary goal
 - Supports better de-coupling
 - Teach the programmers!

Need Asynchronous Pattern



- Variants easy
- Graceful degradation

Asynchronous Services Example



CarRent.com Login



Rental
offers

Legacy
server

Bus

Segmentation

Membership

Location
offers

Brand
offers



Segmentation

Membership

Location
offers

Brand
offers

CarRent.com Login

?

?

?



Rental
offers

Legacy
server



Segmentation

Membership

Location
offers

Brand
offers

CarRent.com Login

?

?

?



Rental offers

Legacy server



Segmentation

Membership

B
Location offers

A
Brand offers

CarRent.com Login

?

?

?



Rental
offers

Legacy
server

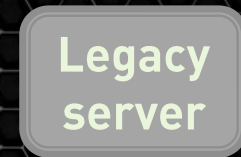
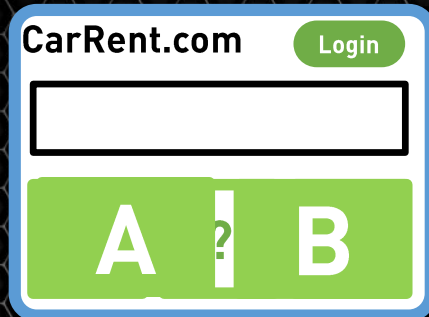


Segmentation

Membership

Location
offers

Brand
offers



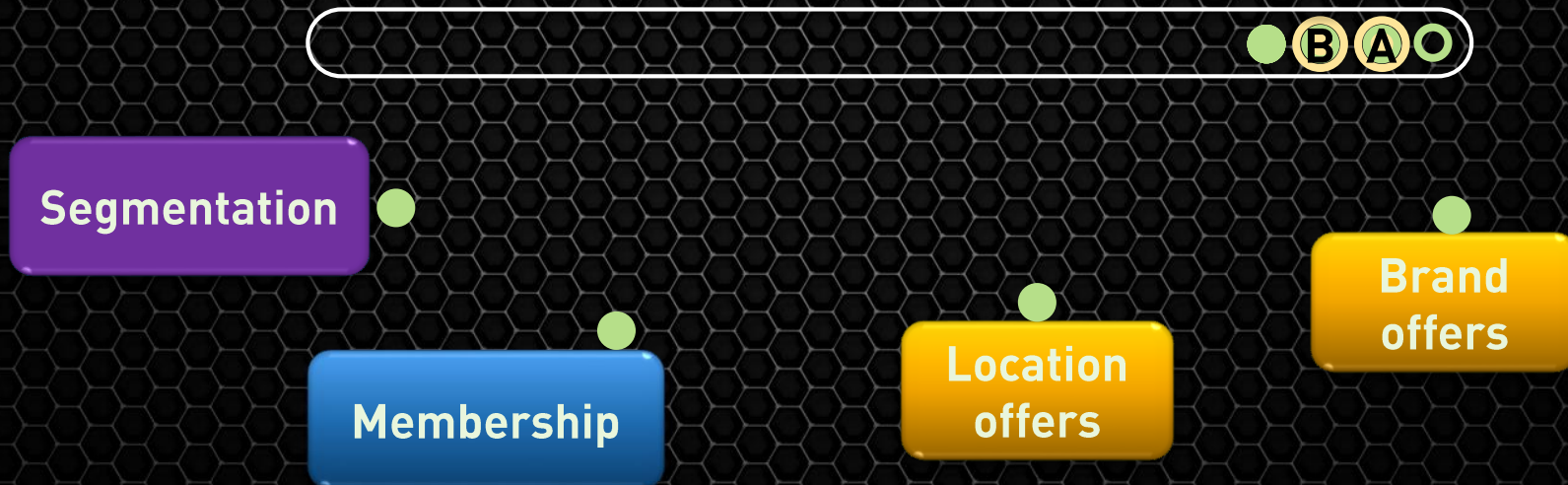


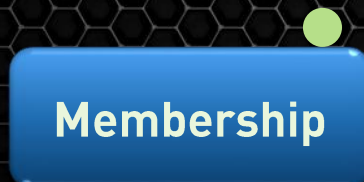
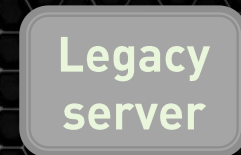
Segmentation

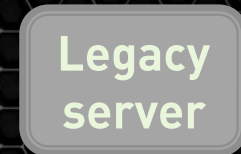
Membership

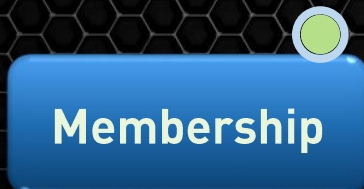
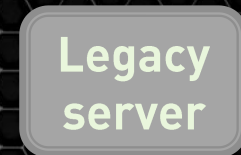
Location
offers

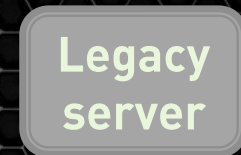
Brand
offers











CarRent.com Login

A B



Rental offers

Legacy server



Segmentation

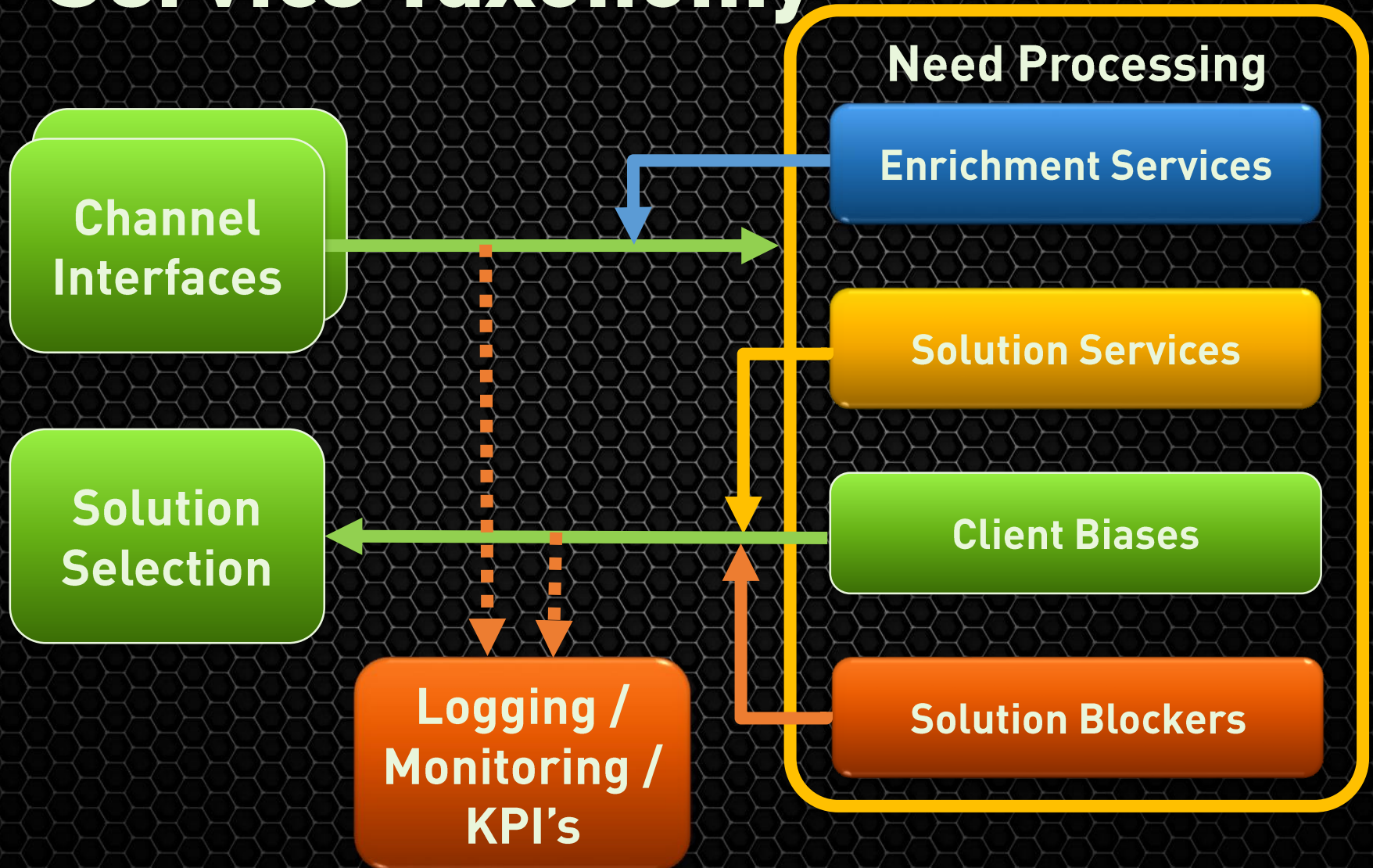


Membership

F
Location offers

E
Brand offers

Service Taxonomy



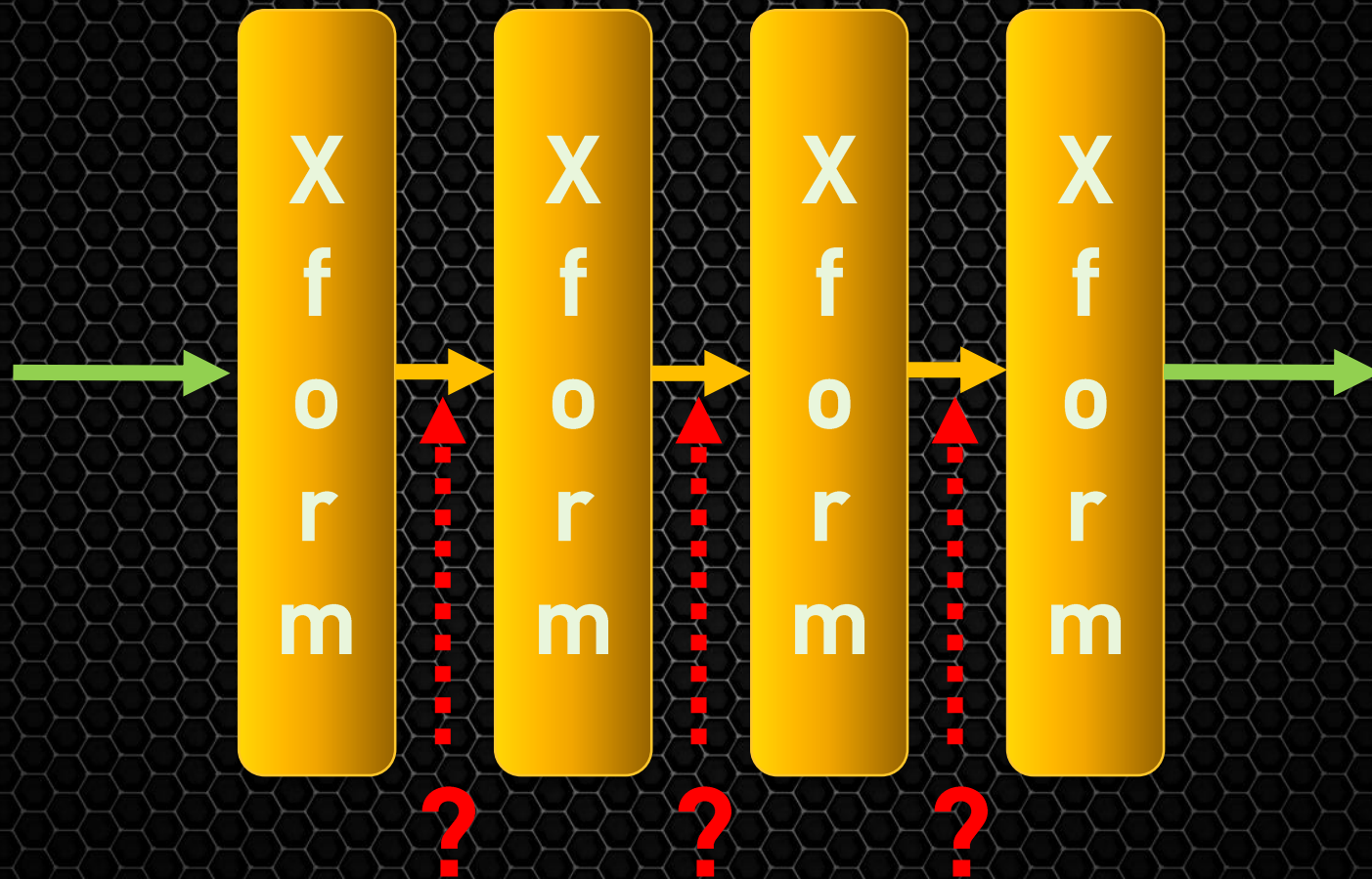
Challenge 3

MicroServices or Clojure

MicroServices Like OO

- **Conceptualization (Job)**
 - Every service has one job
 - If two jobs, make two services
- **Communication**
 - Minimize messages (whether RESTful or Events)
- **Encapsulate**
 - Service has its own persistence
 - If sharing persistence, just one logical service

Clojure **Loves** Shared Data



3 Companies – 3 Variants

Number	
Languages	
Coupling	

3 Companies – 3 Variants

	Forward 2008-now
Number	300+
Languages	Ruby, Node.js, Clojure, R, ...
Coupling	DB w cron, RESTful, Kafka bus

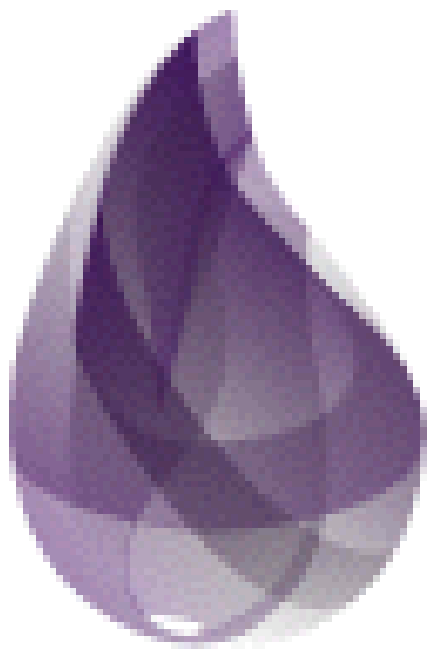
3 Companies – 3 Variants

	Forward 2008-now	Daily Mail 2012-now
Number	300+	3 + dozens
Languages	Ruby, Node.js, Clojure, R, ...	Clojure, Node.js
Coupling	DB w cron, RESTful, Kafka bus	RabbitMQ

3 Companies – 3 Variants

	Forward 2008-now	Daily Mail 2012-now	Outpace 2013-now
Number	300+	3 + dozens	25
Languages	Ruby, Node.js, Clojure, R, ...	Clojure, Node.js	Clojure
Coupling	DB w cron, RESTful, Kafka bus	RabbitMQ	Orchestration, RabbitMQ

A New Hope...



elixir

Challenge 4

Choosing Architectures and Frameworks

Open Source Frameworks

The Netflix logo is displayed in a large, bold, white, sans-serif font. The letters have a slight 3D effect with a dark shadow. The logo is centered within a solid red rectangular background.

Node.js Very Active

PIGATO



PIGATO - an high-performance microservices framework based on ZeroMQ

Challenge 5

No Design Patterns
Book Yet

Conferences / Meetups Starting...

Microservices and Cloud Native Apps - SF Bay Area

[Home](#) [Members](#) [Sponsors](#) [Photos](#) [Discussions](#) [More](#)

 [My profile](#)

London μ Services (Microservices) User Group

[Home](#) [Members](#) [Sponsors](#) [Photos](#) [Discussions](#) [More](#)

[Join us!](#)



[Home](#) [Members](#) [Sponsors](#) [Photos](#) [Discussions](#) [More](#)

[Join us!](#)

Challenge 6

Corequisite Technology, Processes, and Organization

A word cloud on a dark background with a light hexagonal pattern. The central text is "Go Faster" in a large, bold, yellow font with a diagonal hatching pattern. Surrounding it are various terms in white and green fonts of different sizes. The terms include: "Managerless process" (top center), "Cassandra" (top right), "Lean startup" (top left), "Programmer anarchy" (middle left), "MicroServices" (center, in green), "Docker" (middle right), "Dev/Ops" (bottom left), "Cloud" (bottom center-left), "Full-stack developer" (bottom center-right), "Agile" (bottom right), "Event bus" (bottom left-center), "MVP (minimum viable product)" (bottom center), and "No-SQL" (bottom right).

Managerless
process

Cassandra

Lean startup

Programmer
anarchy

MicroServices

Docker

Go Faster

Dev/Ops

Cloud

Full-stack
developer

Agile

Event bus

MVP (minimum
viable product)

No-SQL

Go Faster

Managerless process

MicroServices

Docker

Full-stack developer

Agile

No-SQL

MVP (minimum viable product)

Event bus

Dev/Ops

Cloud

Lean startup

Programmer anarchy

Cassandra

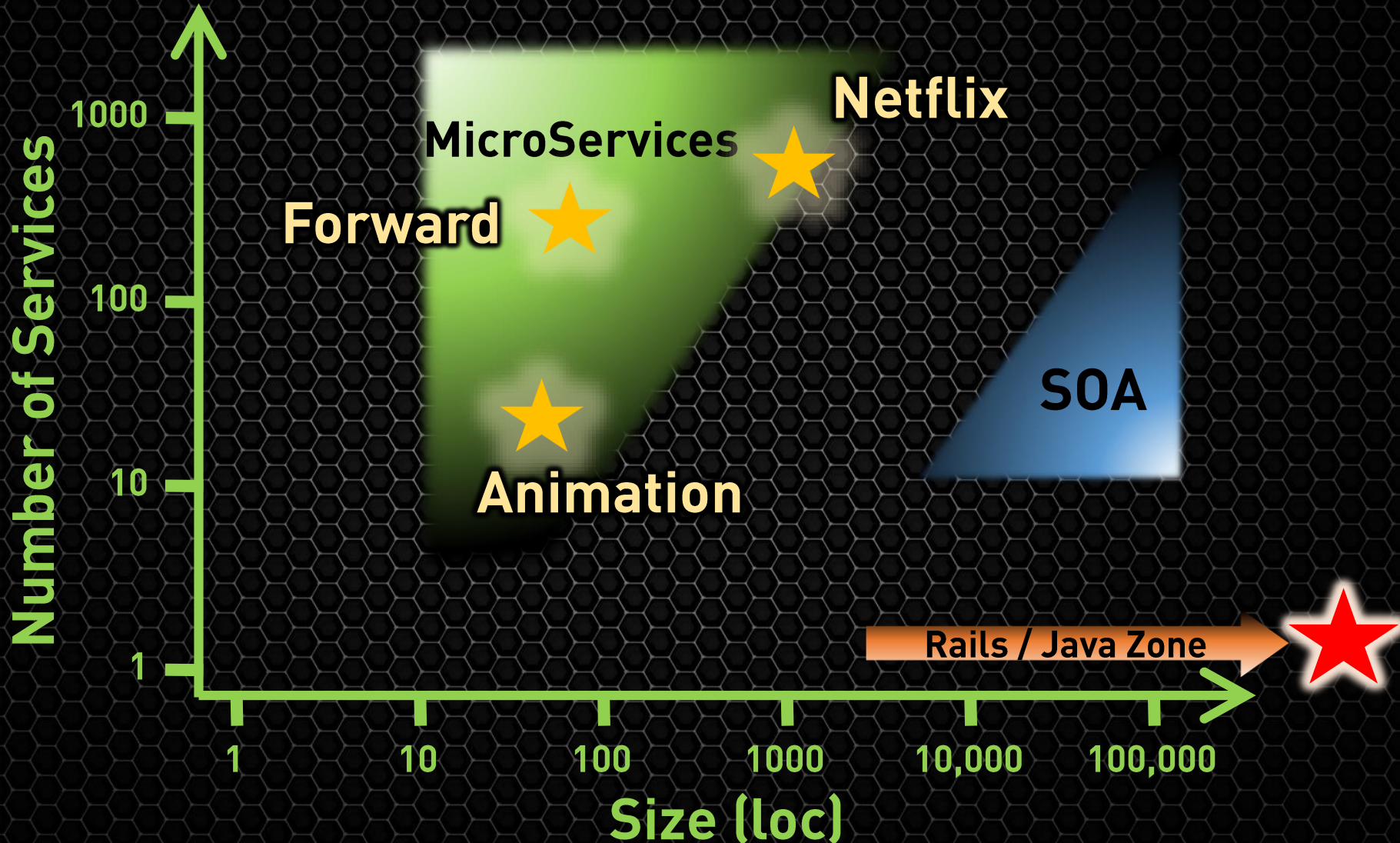
Challenge 7

What is a MicroService?

Taxonomy May be Useful (before it's too late)

- Synchronicity degree
 - Primary API access to services
- Ratio of number of services and average size
 - Zones for clarity
- DB / service ratio
 - Expose potential DB hindrances to rapid deployment

Count / Size Ratio



More Information...

- **Google:**
 - **MicroService Architecture for videos**
 - **MicroXchg 2015 conference**
- **Complementary topic:**
 - **Google for Programmer Anarchy**

MicroService Challenges

Fred George

fredgeorge@acm.org



@fgeorge52