

Microservices

@Steve_Upton

Who am I?

- Steve Upton
- English / Welsh / British / Irish
- BSc. Computer Science (Cardiff University)
- Physics & Astronomy (Open University)
- IBM (6 years)
 - Messaging
 - OASIS MQTT TC member
 - Working with clients on Microservice systems
 - London μ Service user group
- HERE Berlin (2 years)
 - Microservices and robots
- ThoughtWorks
 - Lead QA Consultant



Who are you?

Microservices

The histories of Microservices

Why you shouldn't do Microservices

Microservices in the Real World

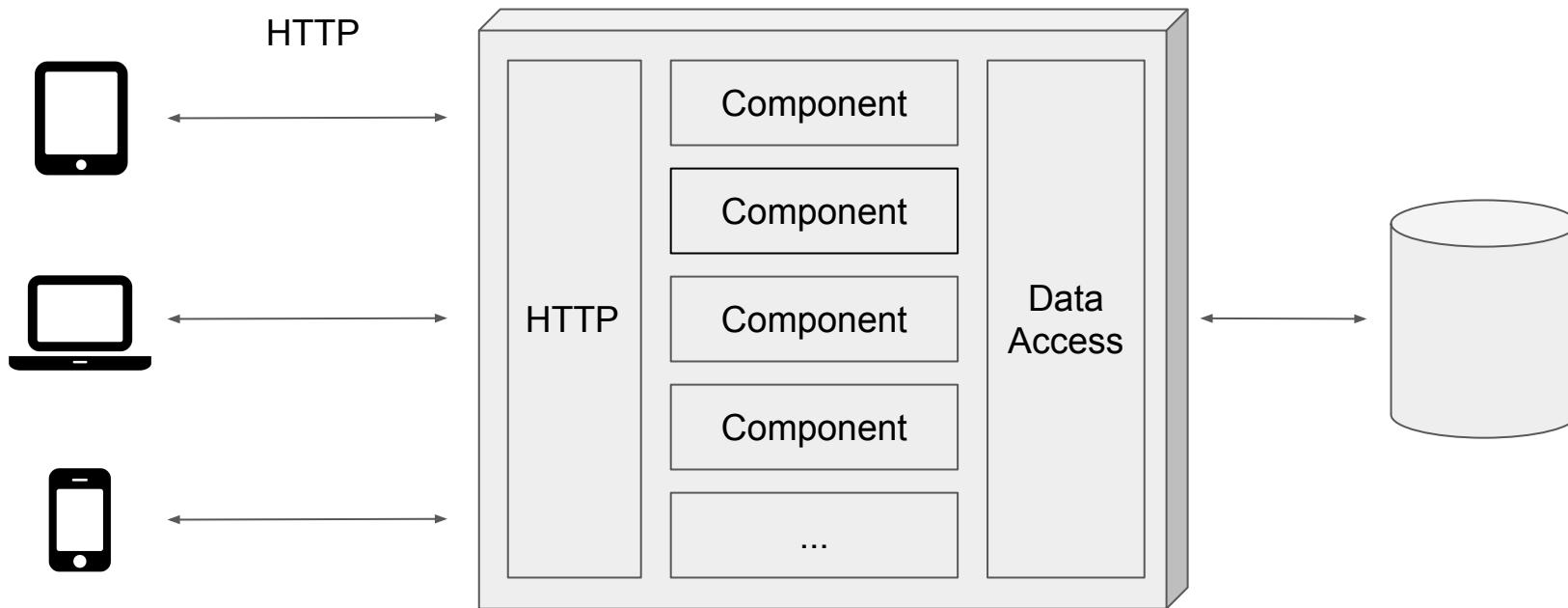
The future of Microservices

The histories of Microservices

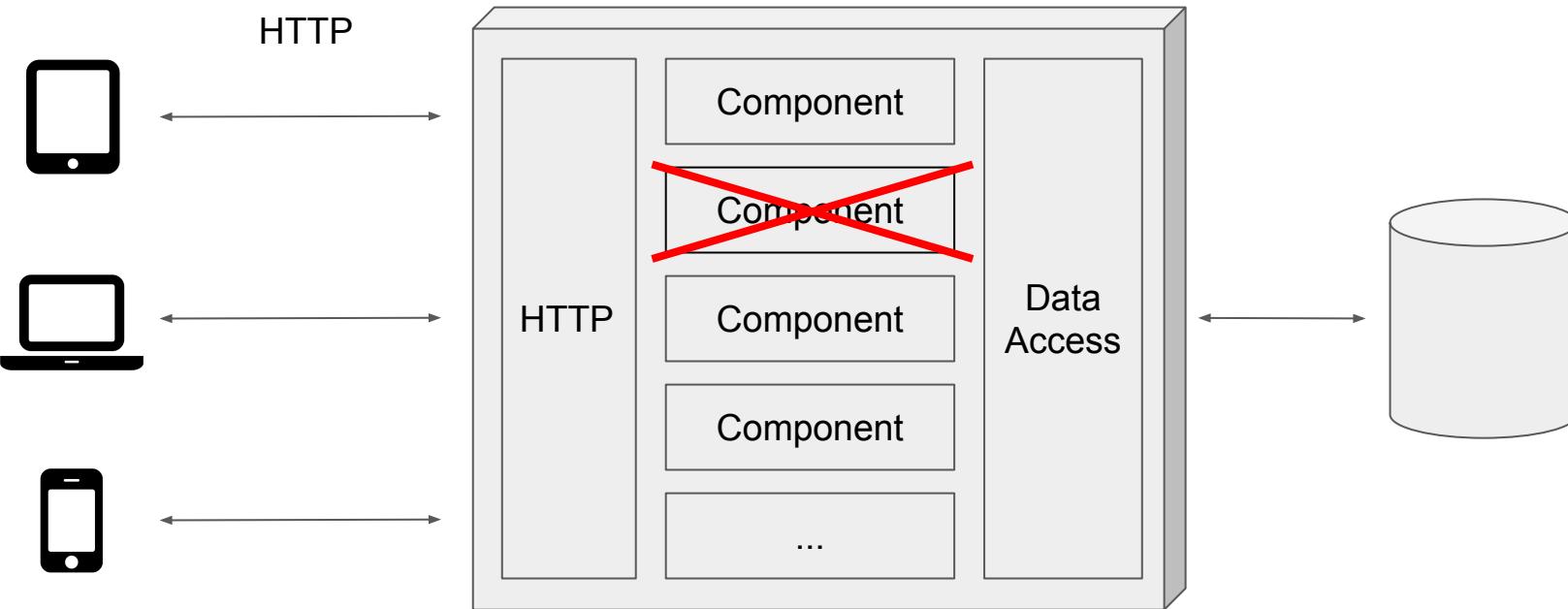
The standard Microservices creation story

NETFLIX

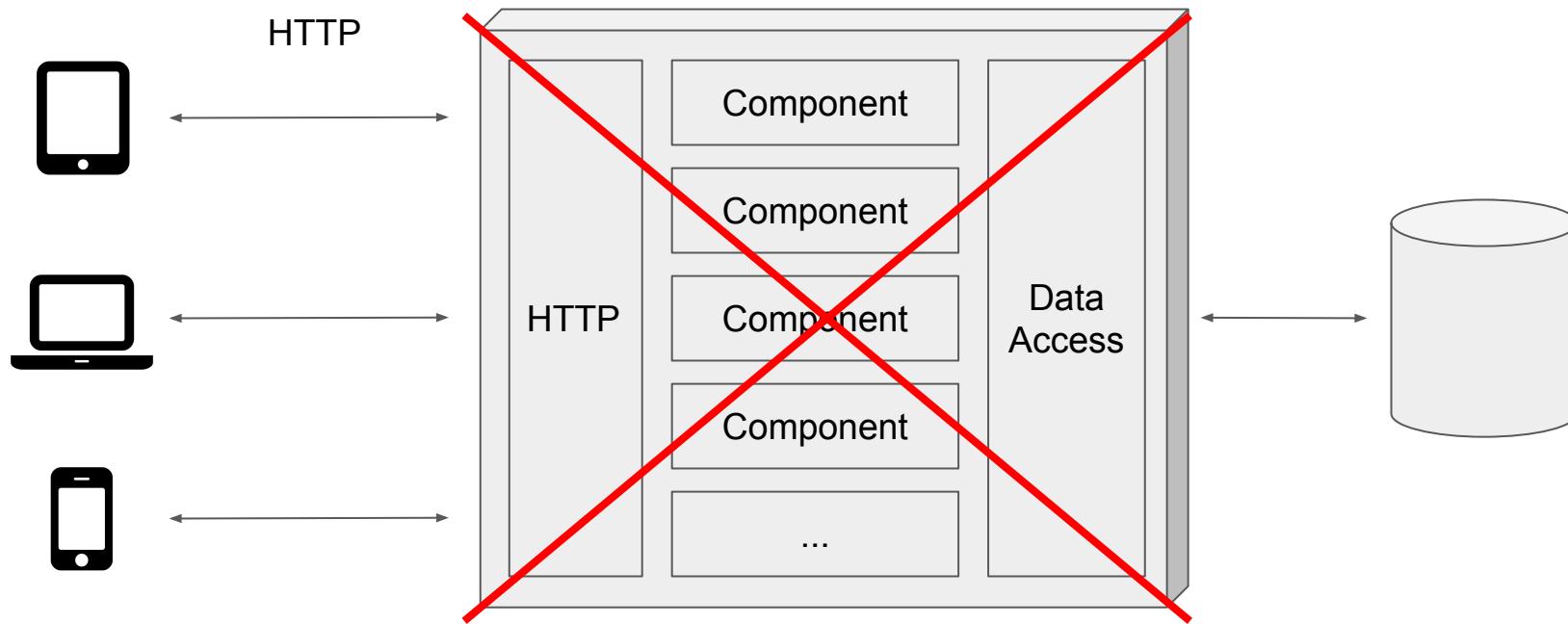
Monolith



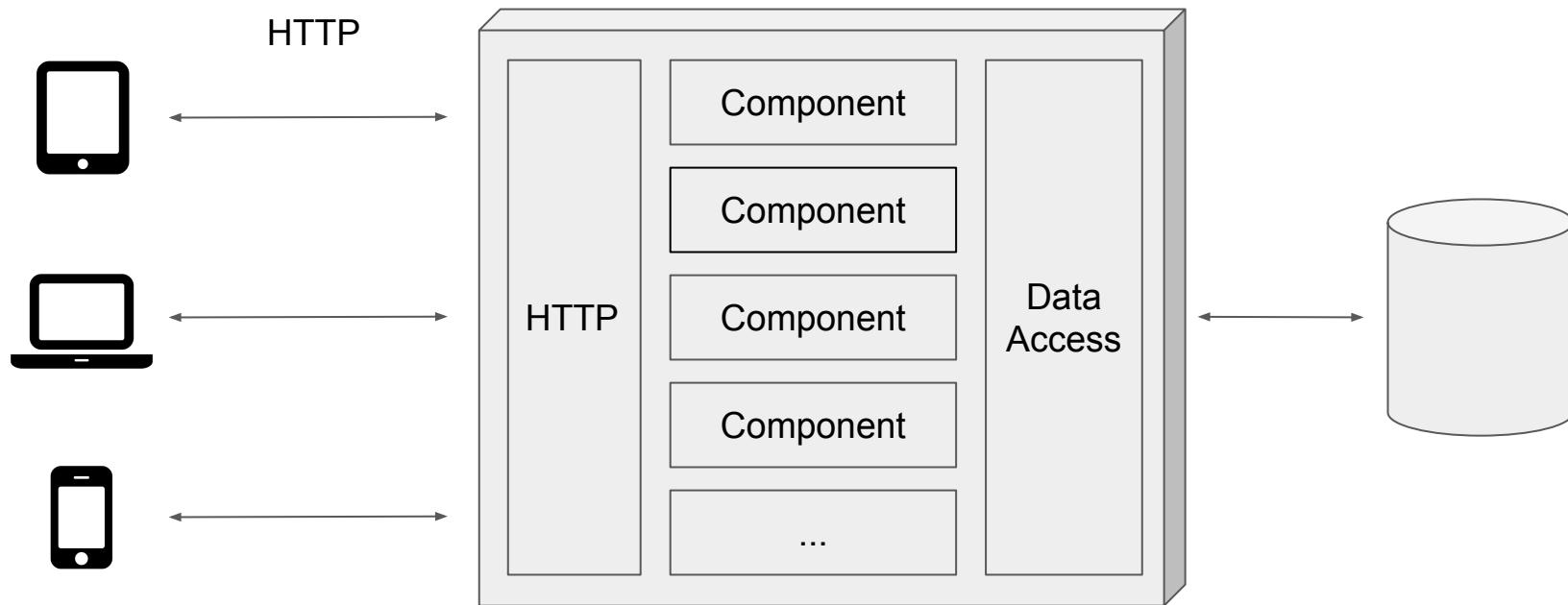
Monolith



Monolith



Monolith



Hard to scale

Hard to scale

RDBMS was also a **SPOF**

Hard to scale

RDBMS was also a **SPOF** and **bottleneck**

Hard to scale

RDBMS was also a **SPOF** and **bottleneck**

Vertical scaling easy, but **limited**

Hard to scale

RDBMS was also a **SPOF** and **bottleneck**

Vertical scaling easy, but **limited**

Horizontal scaling helpful, but **slow and expensive**

Hard to scale

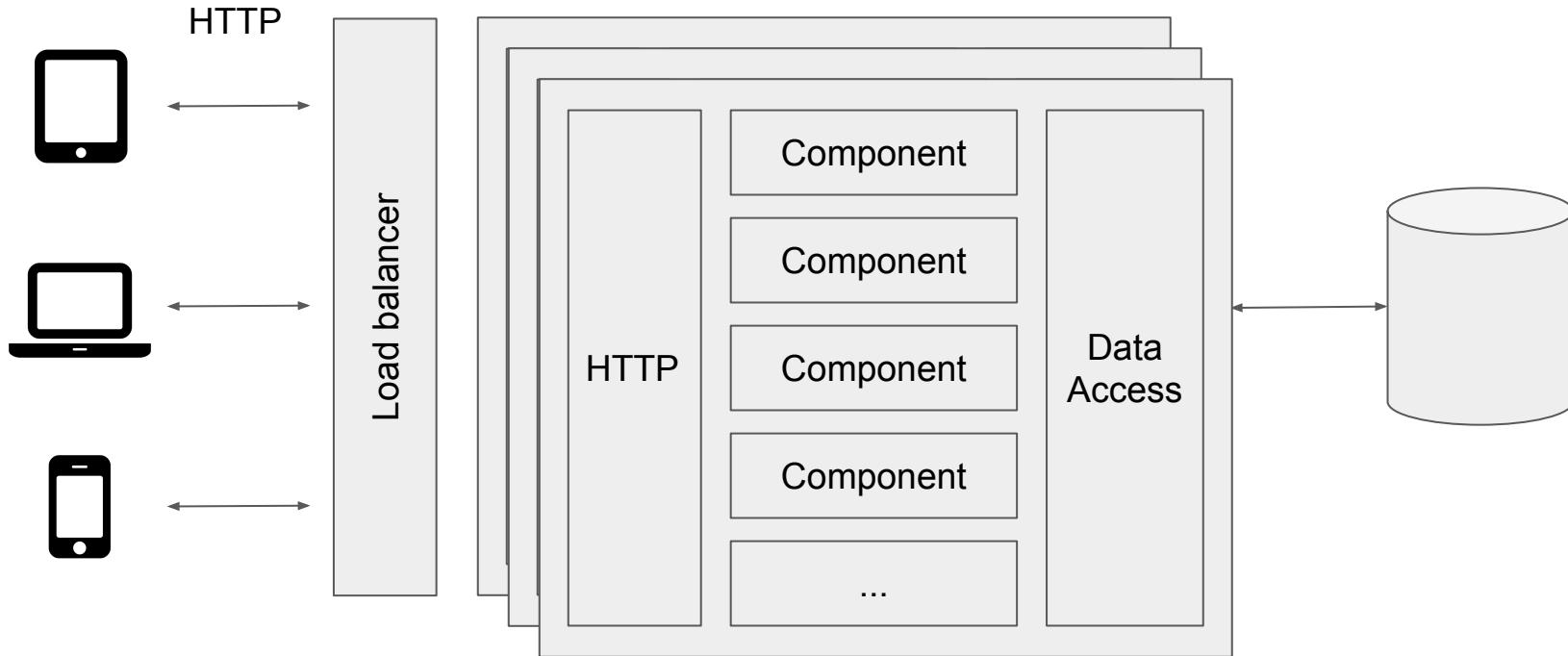
RDBMS was also a **SPOF** and **bottleneck**

Vertical scaling easy, but **limited**

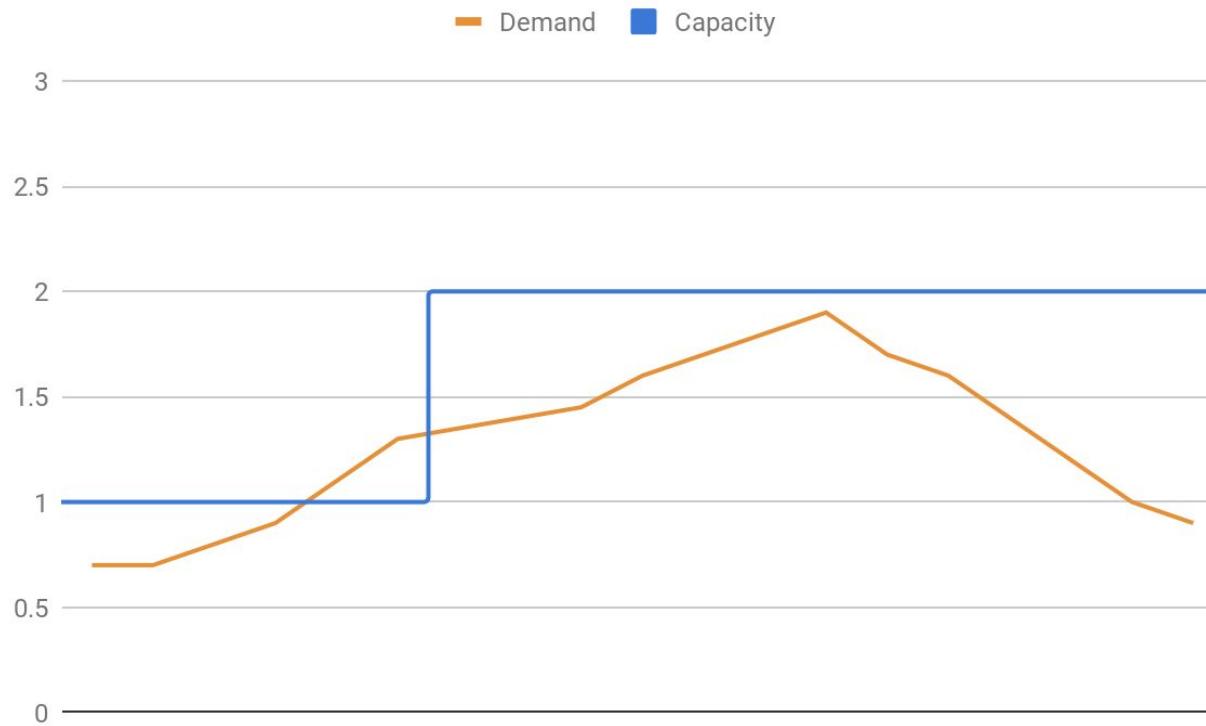
Horizontal scaling helpful, but **slow and expensive**

Scaling the whole monolith was **wasteful**

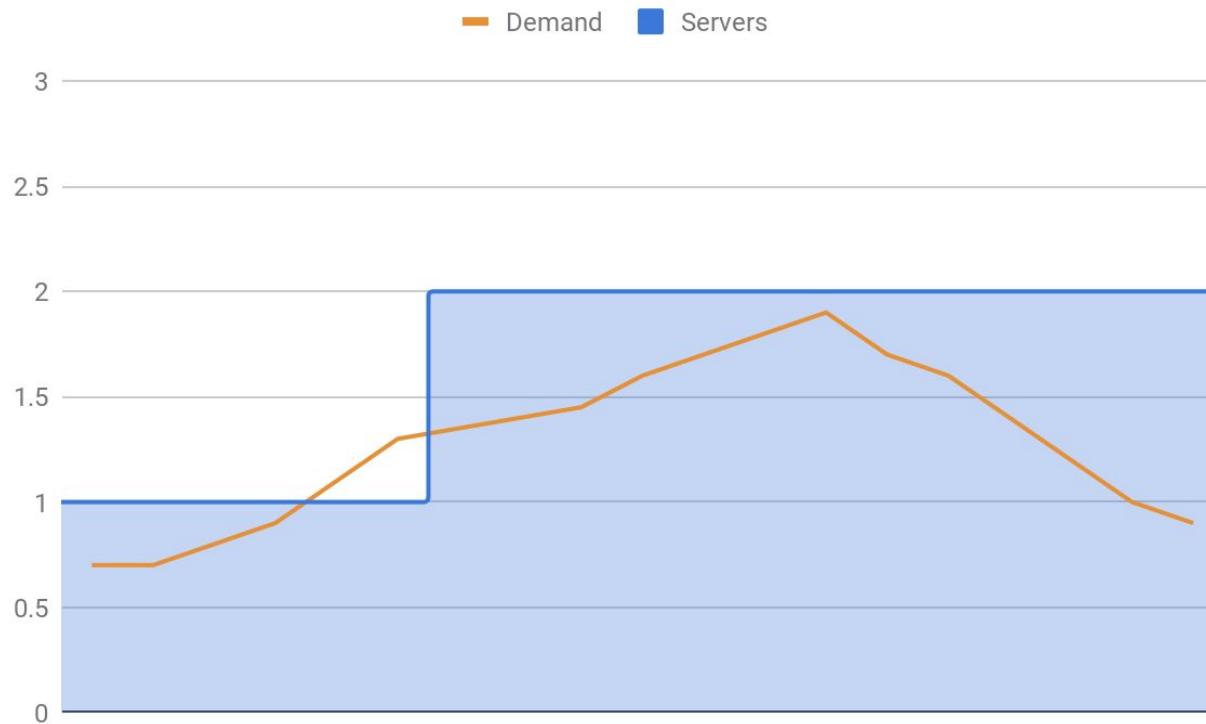
Scaling the monolith



Scaling the monolith



Scaling the monolith



“...the average server operates at no more than **12 to 18 percent** of its capacity”

Natural Resources Defense Council

Tightly coupled, slow to change

Tightly coupled, slow to change

Sharing knowledge of components

Difficult to upgrade without affecting other services

Tightly coupled, slow to change

Sharing knowledge of components

Difficult to upgrade without affecting other services

Sharing libraries, platform, OS etc.

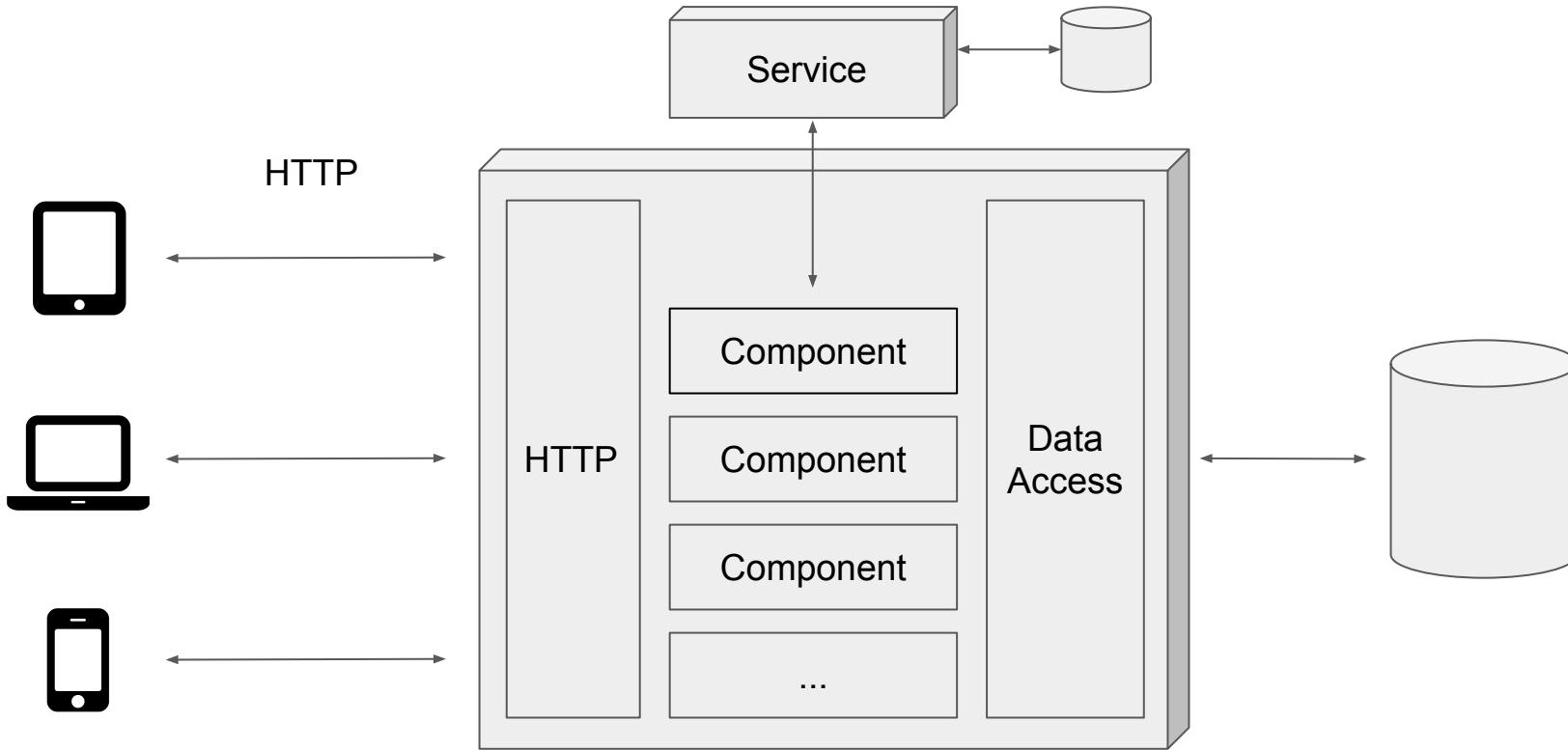
Teams have **little choice** in setup



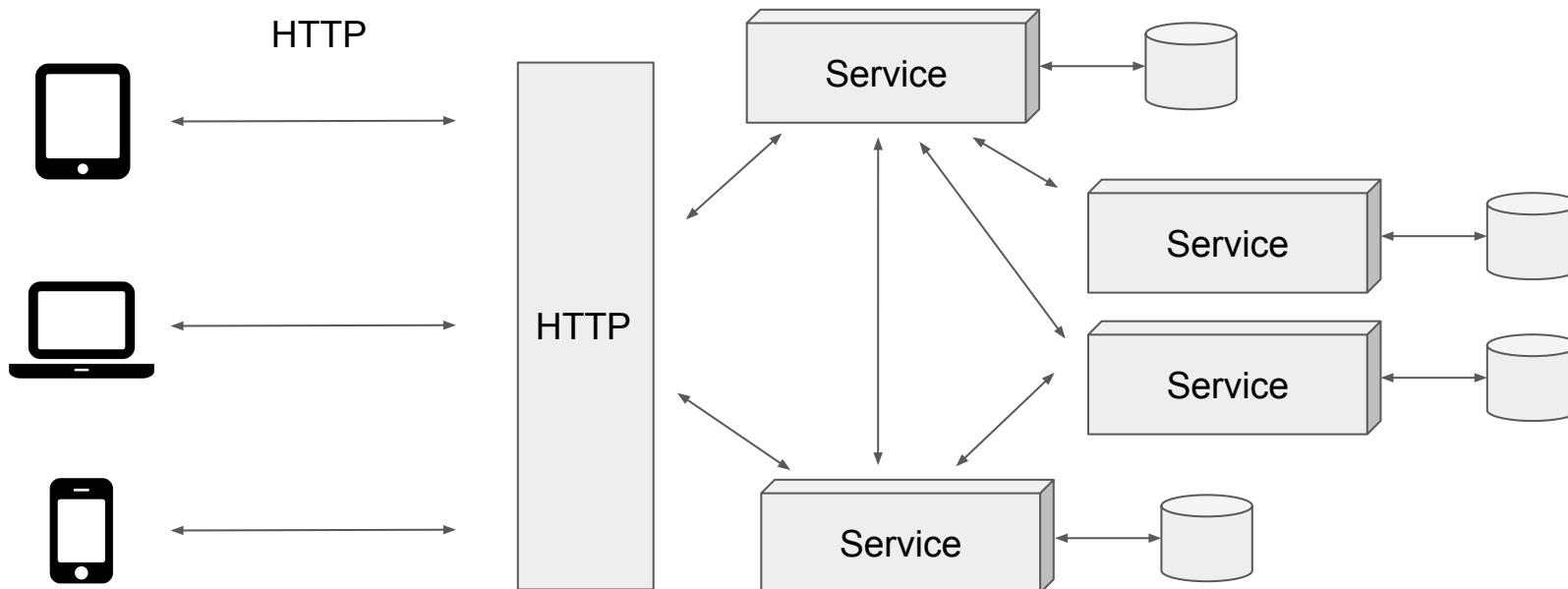




Moving to the cloud



Microservices



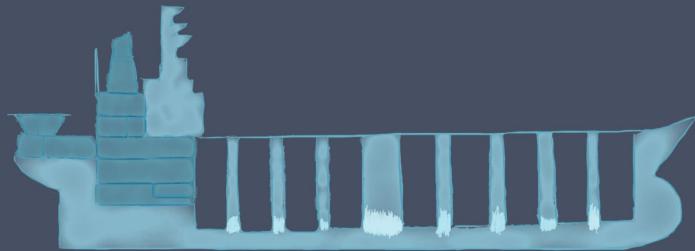
Resilience to failure

Able to **isolate failures**

Resilience to failure

Able to **isolate** failures

BULKHEADING

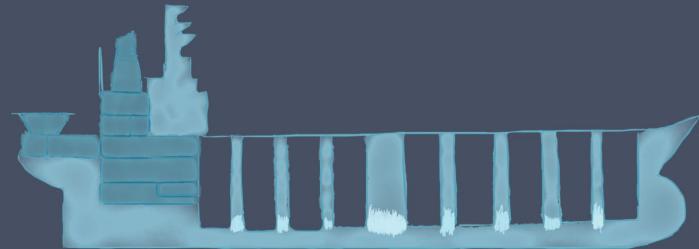


Resilience to failure

Able to **isolate** failures

Gracefully adapt to failures

BULKHEADING

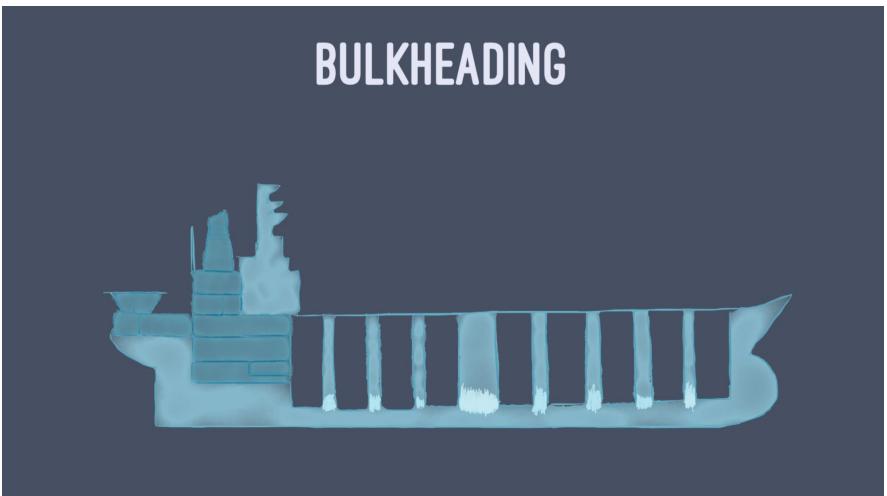


Resilience to failure

Able to **isolate** failures

Gracefully adapt to failures

Quick **recovery**



Loosely coupled, easy to change

Share **contracts**, not internals

Little to no knowledge of other services needed

Loosely coupled, easy to change

Share **contracts**, not internals

Little to no knowledge of other services needed

Teams **free to choose** OS, language, libraries etc.

Free to **upgrade** and **experiment independently**

Easy to scale

Vertical scaling easy and **more effective**

Easy to scale

Vertical scaling easy and **more effective**

Horizontal scaling also easy!

Easy to scale

Vertical scaling easy and **more effective**

Horizontal scaling also easy!

More **efficient** use of resources

Easy to scale

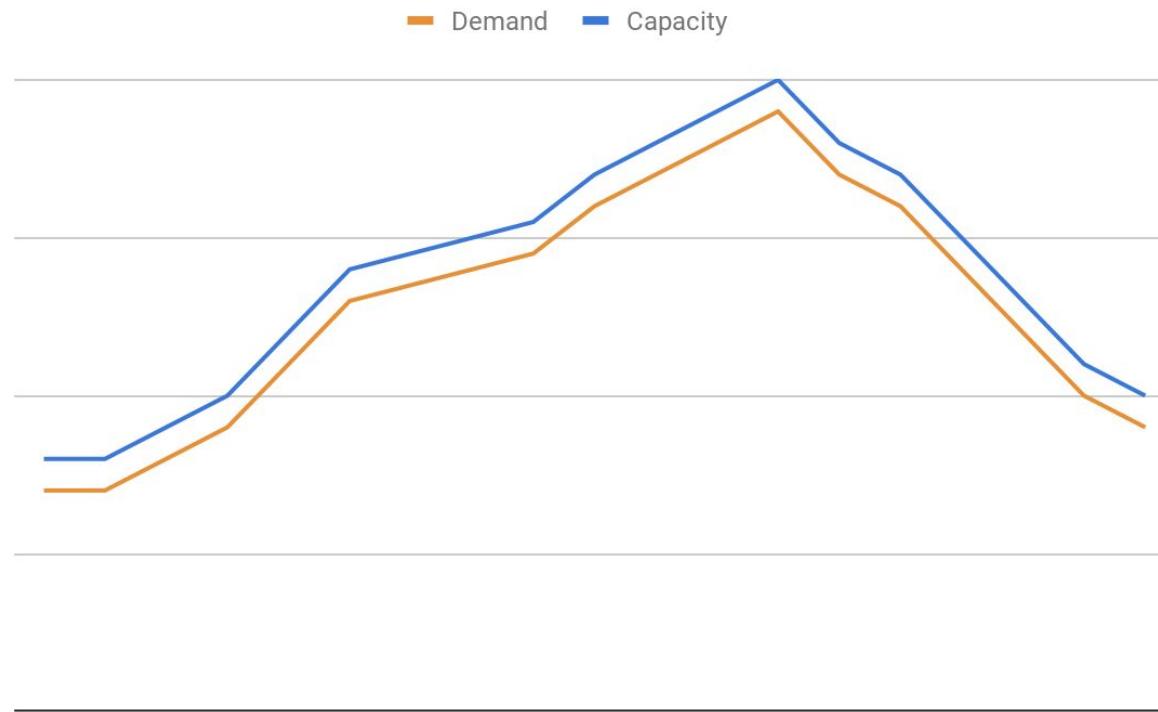
Vertical scaling easy and **more effective**

Horizontal scaling also easy!

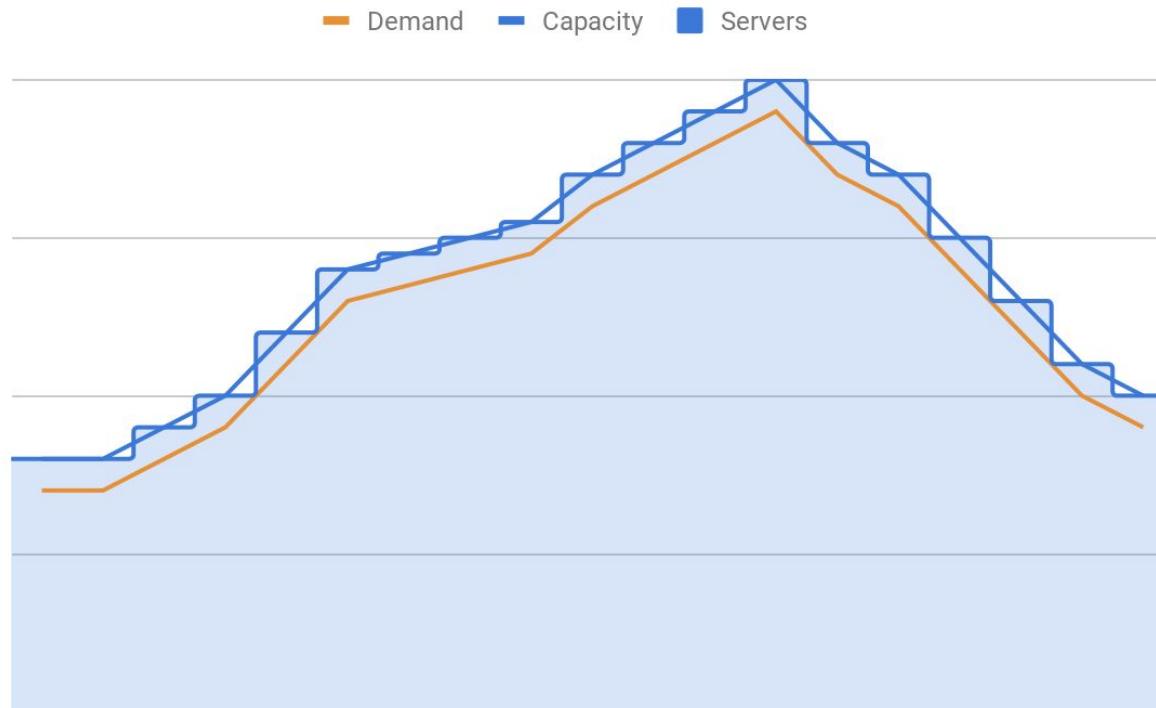
More **efficient** use of resources

Auto scaling of individual services possible

Scaling microservices

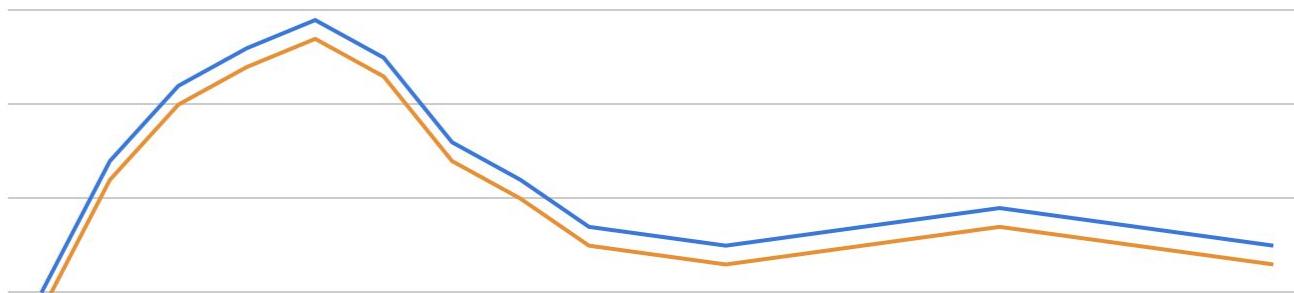


Scaling microservices



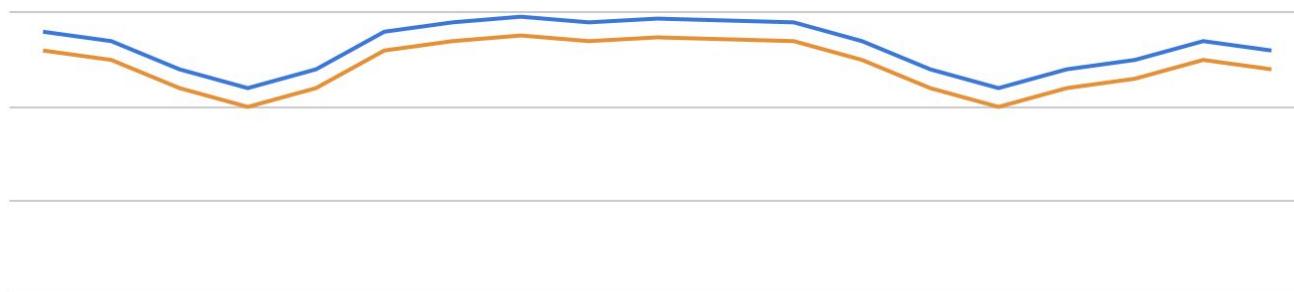
Login service

Demand Capacity



Search service

Demand Capacity



Auto scaling



Amazon EC2

The screenshot shows the configuration of an Auto Scaling rule. A yellow box highlights the 'Rule name' field containing 'MyScalingRule'. Another yellow box highlights the 'Scaling adjustment' field showing '+1 Instances'. A third yellow box highlights the 'CloudWatch metric' dropdown set to 'YARNMemoryAvailablePercentage'. A fourth yellow box highlights the 'Evaluation period' dropdown set to '1 five-minute periods'. A fifth yellow box highlights the 'Comparison operator' dropdown set to 'less than or equal to'. A sixth yellow box highlights the 'Threshold' input field set to '15 percent'. A seventh yellow box highlights the 'Cooldown' input field set to '1 five-minute periods'. A eighth yellow box highlights the 'for' condition set to '1 five-minute periods'.

Rule name: MyScalingRule

Add 1 Instances

Evaluation period

CloudWatch metric: YARNMemoryAvailablePercentage

is less than or equal to 15 percent

for 1 five-minute periods

Scaling adjustment

Cooldown period: 1 five-minute periods

Comparison operator

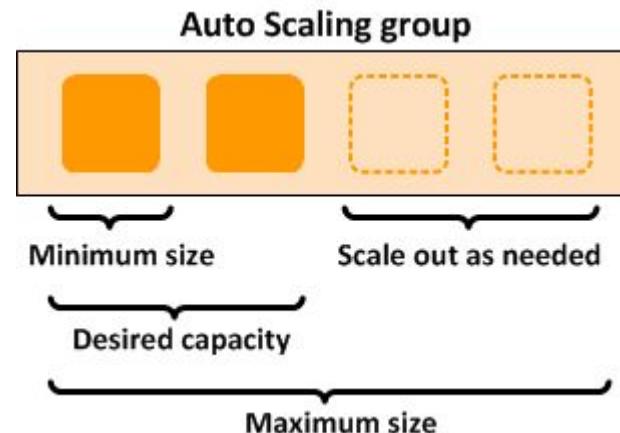
Threshold: 15 percent

Cooldown

Auto scaling



Amazon EC2



NIST Definition of Cloud Computing

- On-demand, self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service



“Cattle not pets”

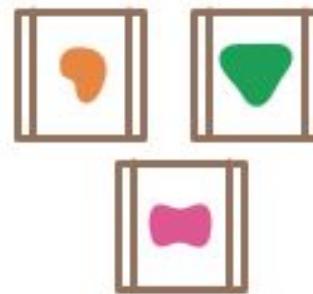
Gavin McCance, CERN

“In short, the microservice architectural style is an approach to developing a single application as a **suite of small services**, each **running in its own process** and communicating with lightweight mechanisms, often an HTTP resource API. These services are **built around business capabilities** and **independently deployable** by fully automated deployment machinery. There is a **bare minimum of centralized management** of these services, which may be written in different programming languages and use different data storage technologies.”

James Lewis and Martin Fowler, ThoughtWorks

**“Loosely coupled service oriented
architecture with bounded contexts”**

Adrian Cockcroft, Architect @ Netflix



EFFICIENCY

(of delivery)

~~EFFICIENCY~~

(of delivery)

SPEED

(of delivery)

~~RESOURCE
EFFICIENCY~~

FLOW
EFFICIENCY



<https://netflix.github.io/>

<http://techblog.netflix.com/>



nebula



restify



Atlas



NETFLIX

NETFLIX



ebay



CLOUD FOUNDRY

amazon



zalando

Groupon®

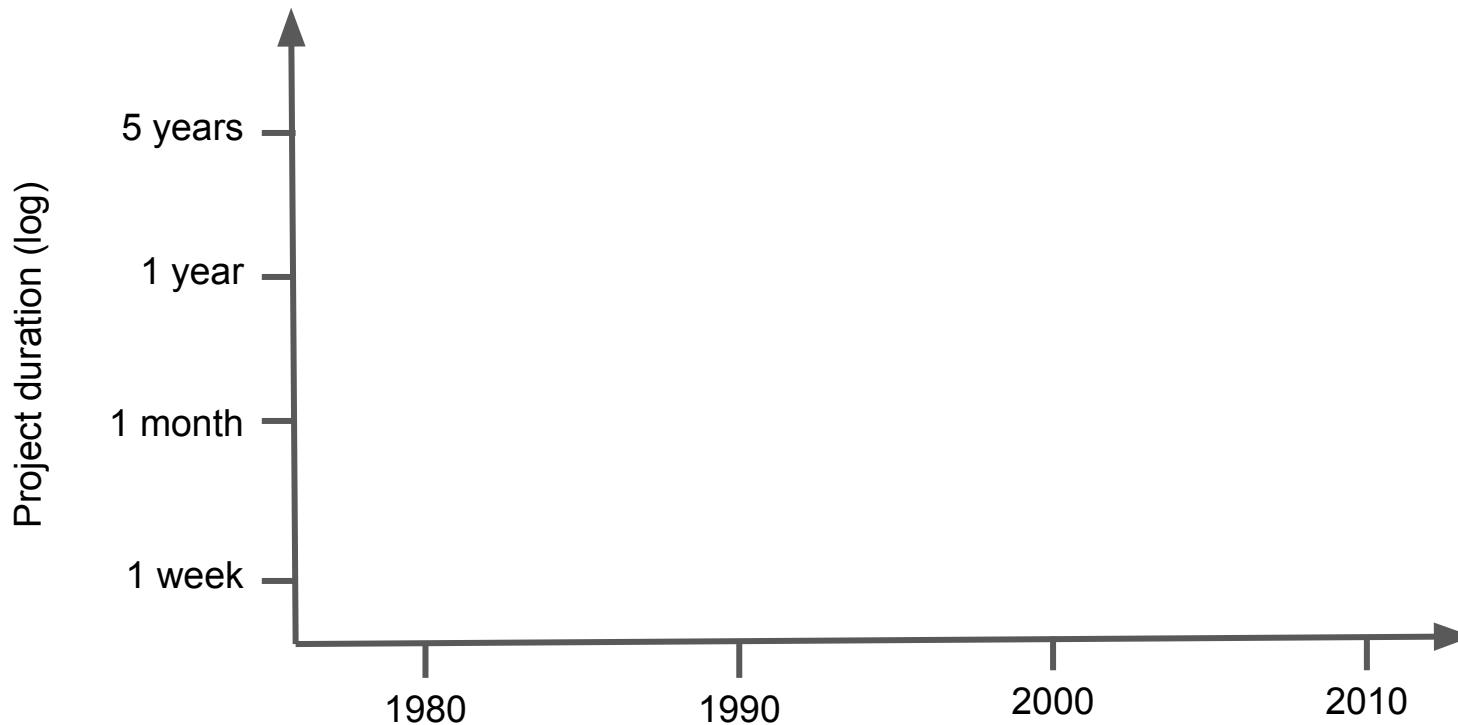
the guardian

The other story

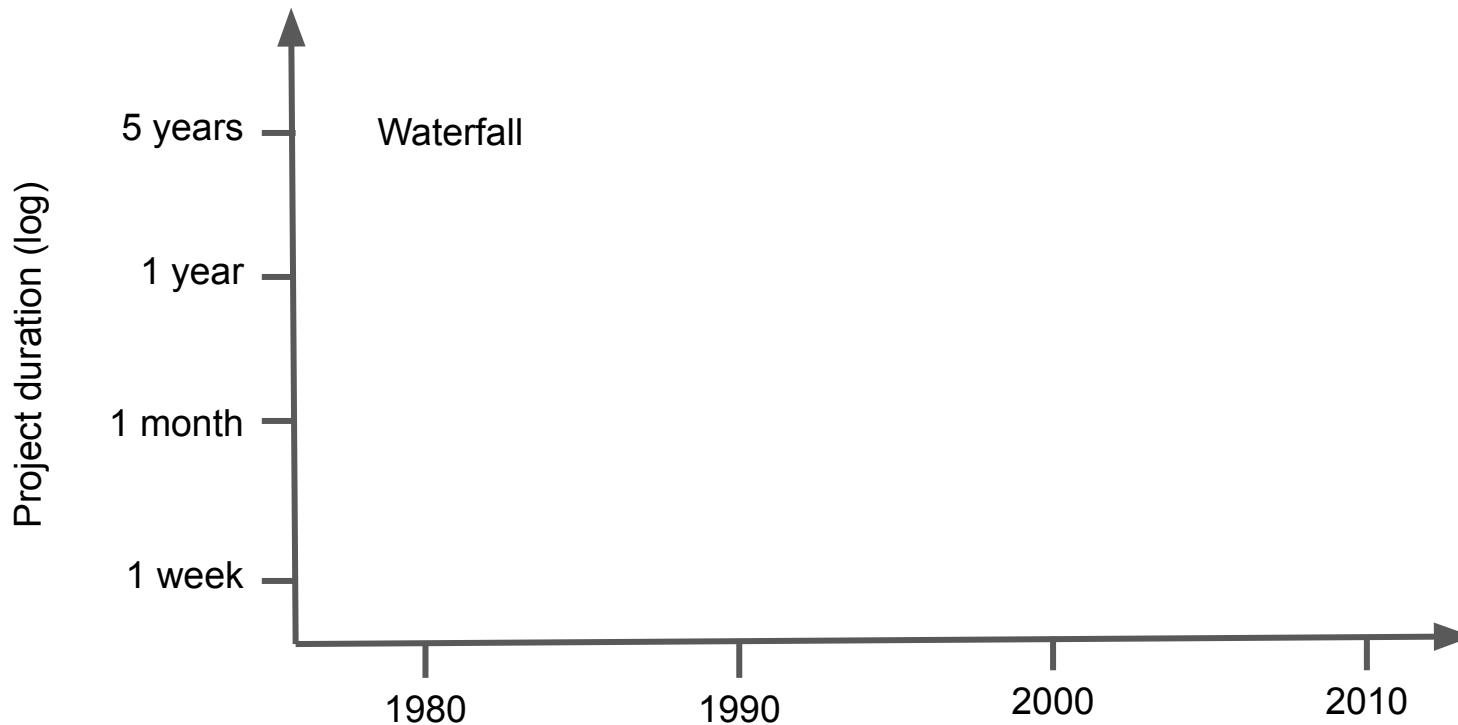
The other story credit to Fred George



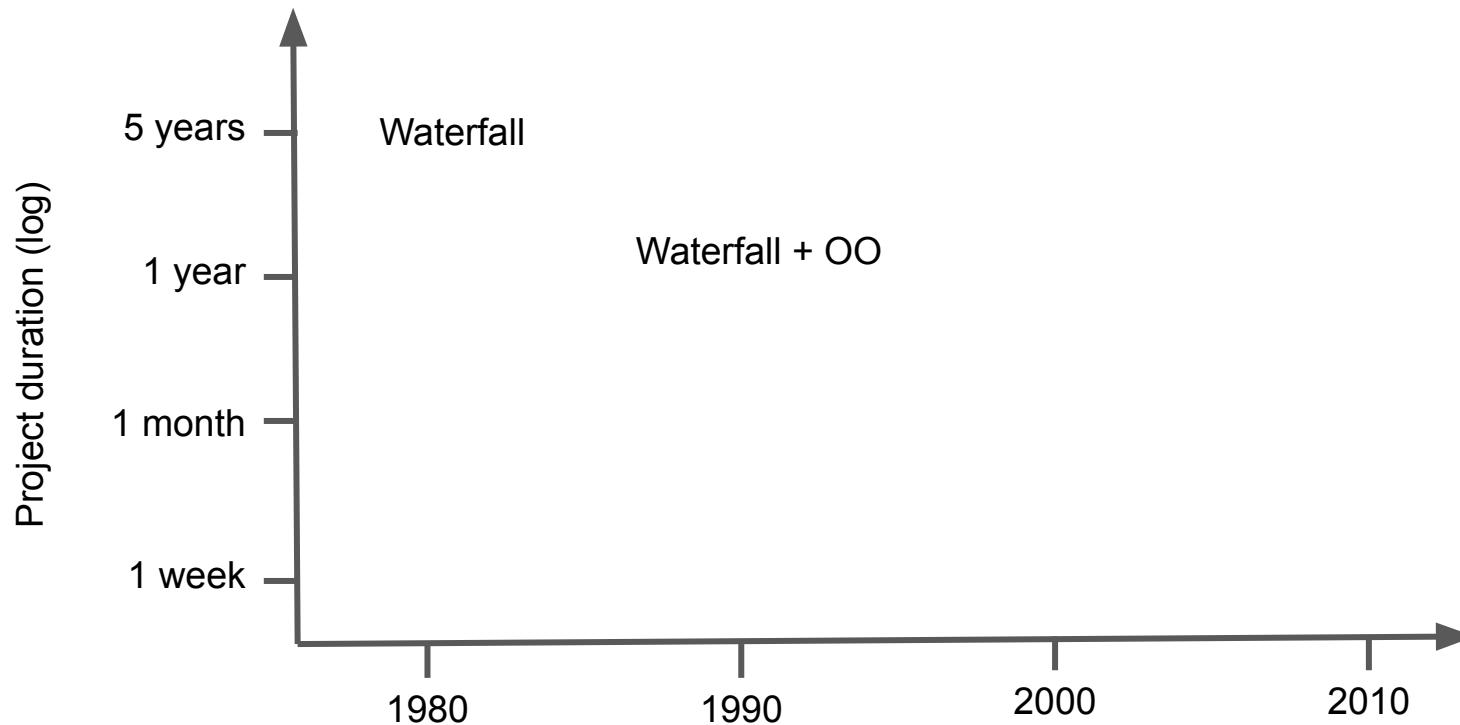
Project delivery cycles



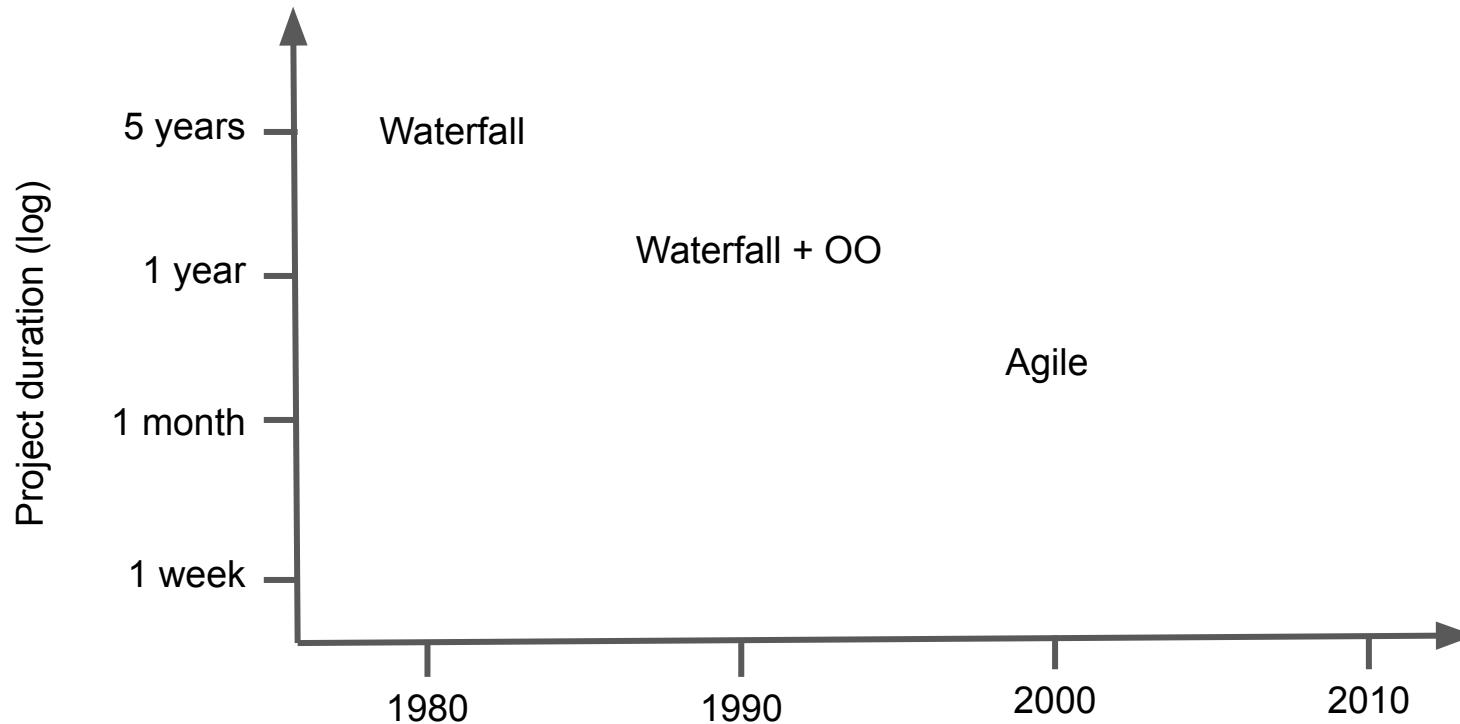
Project delivery cycles



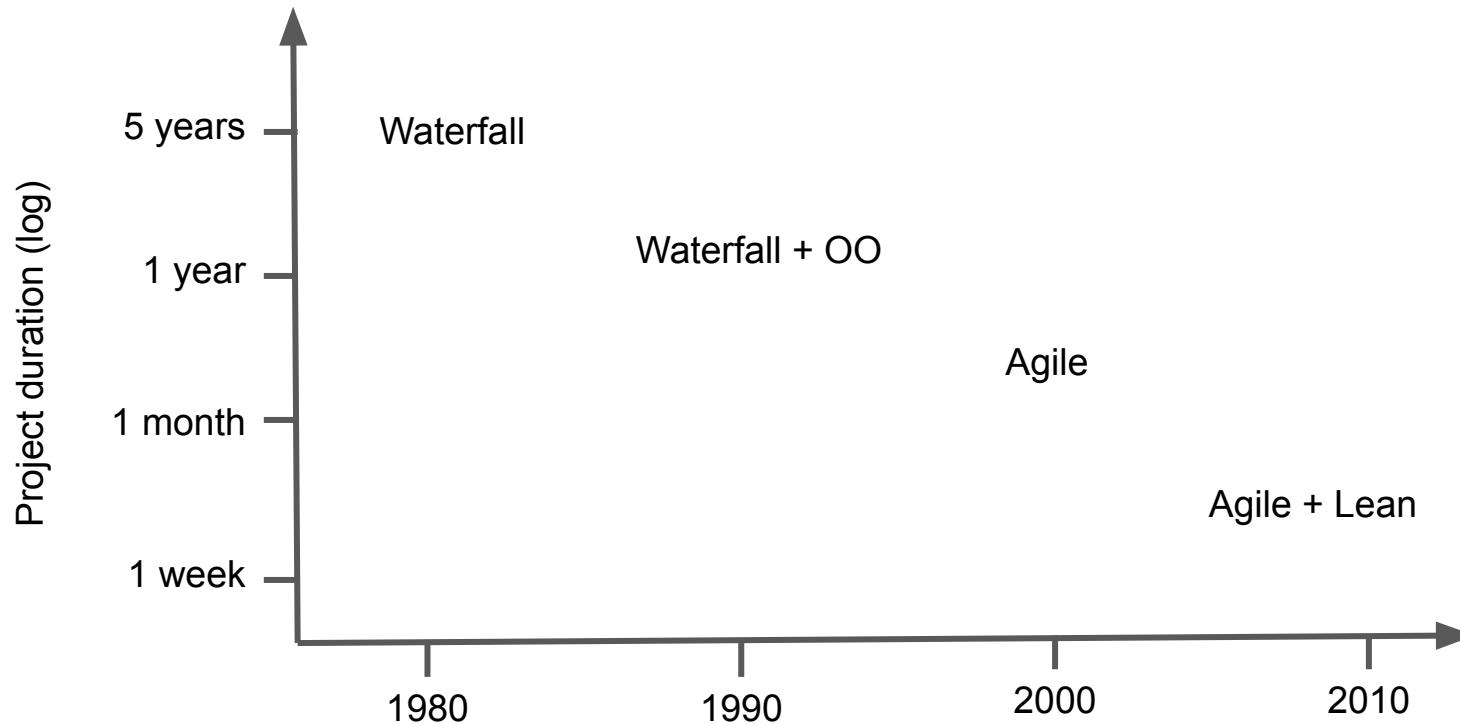
Project delivery cycles



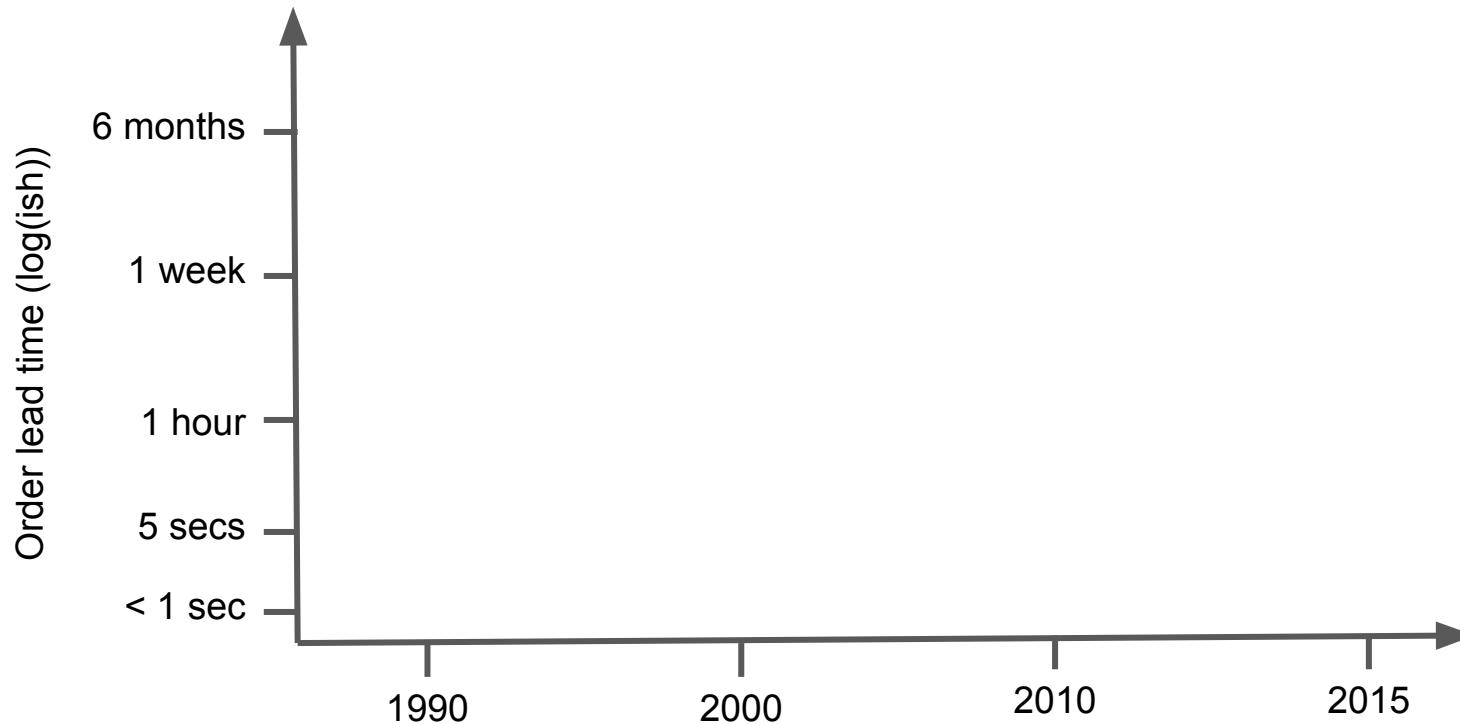
Project delivery cycles



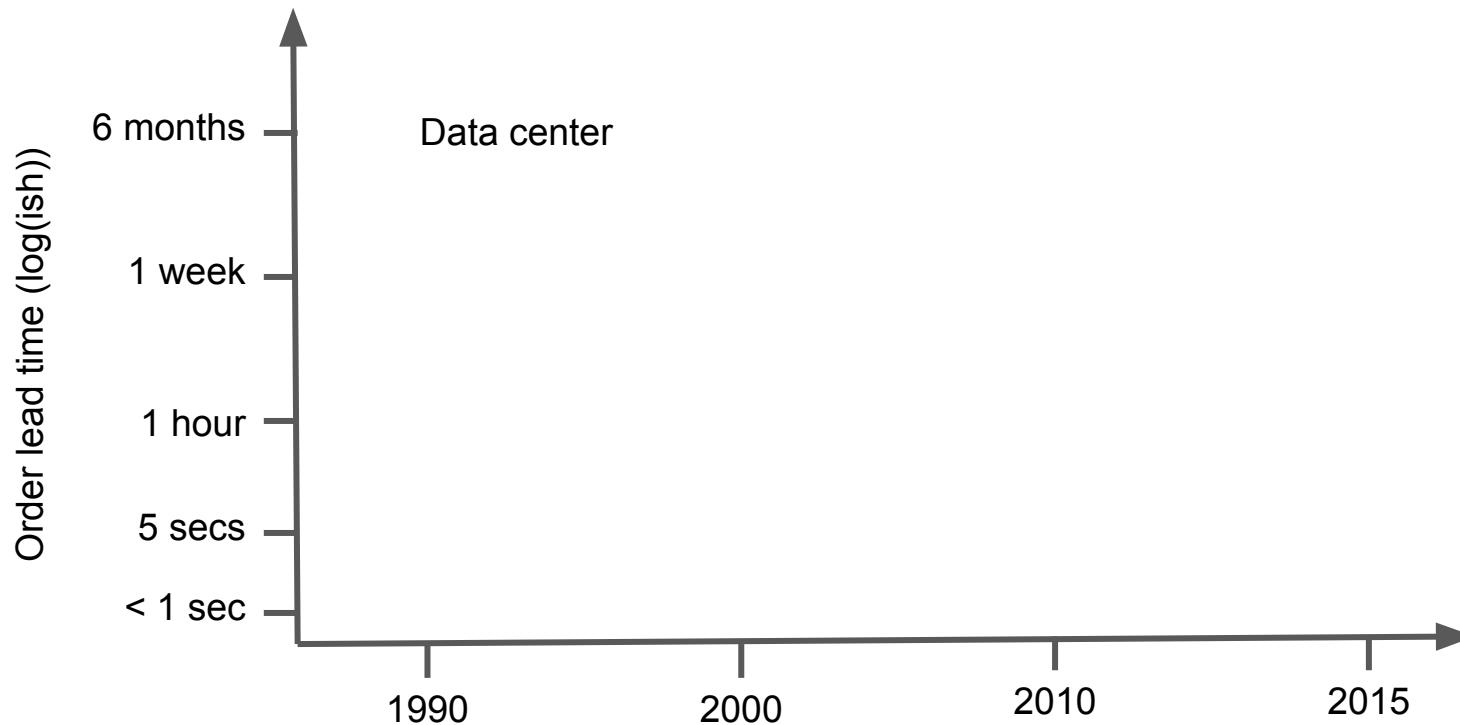
Project delivery cycles



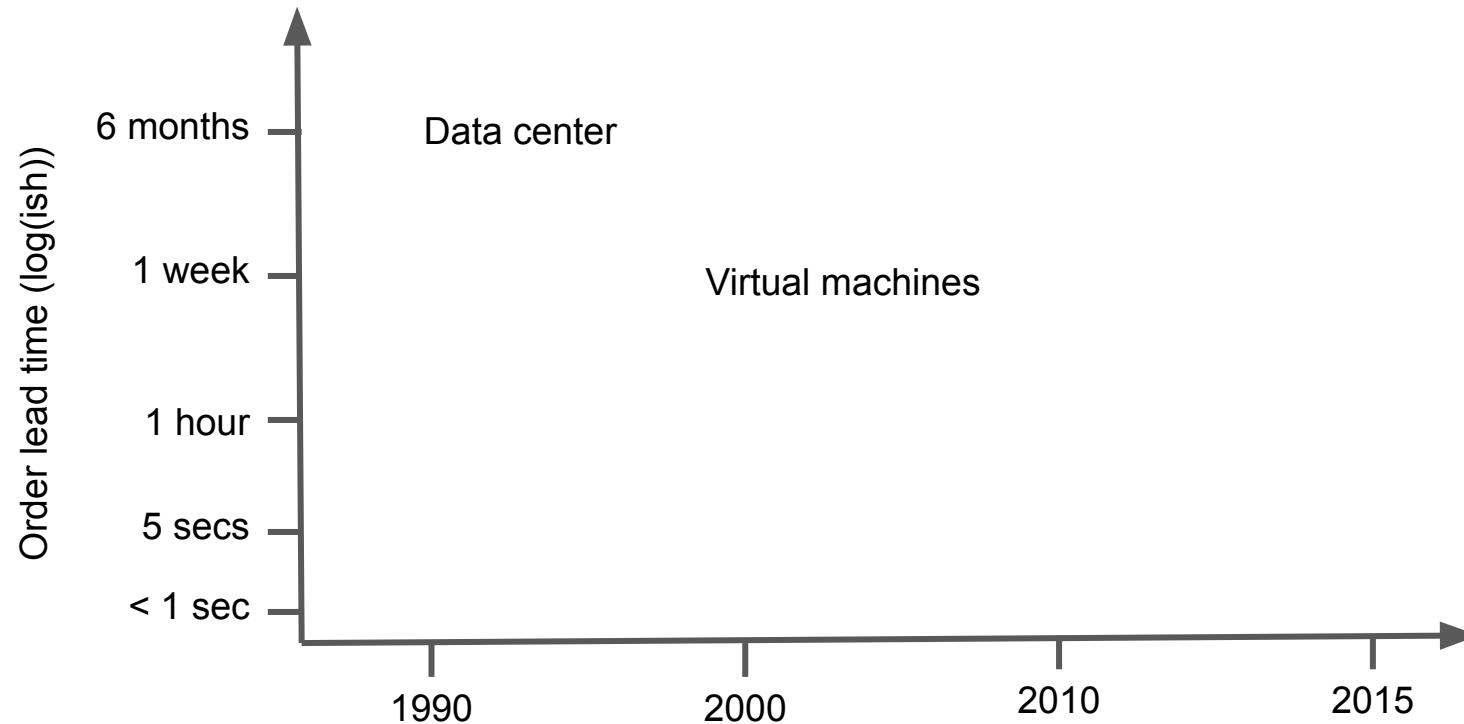
Hardware lead times



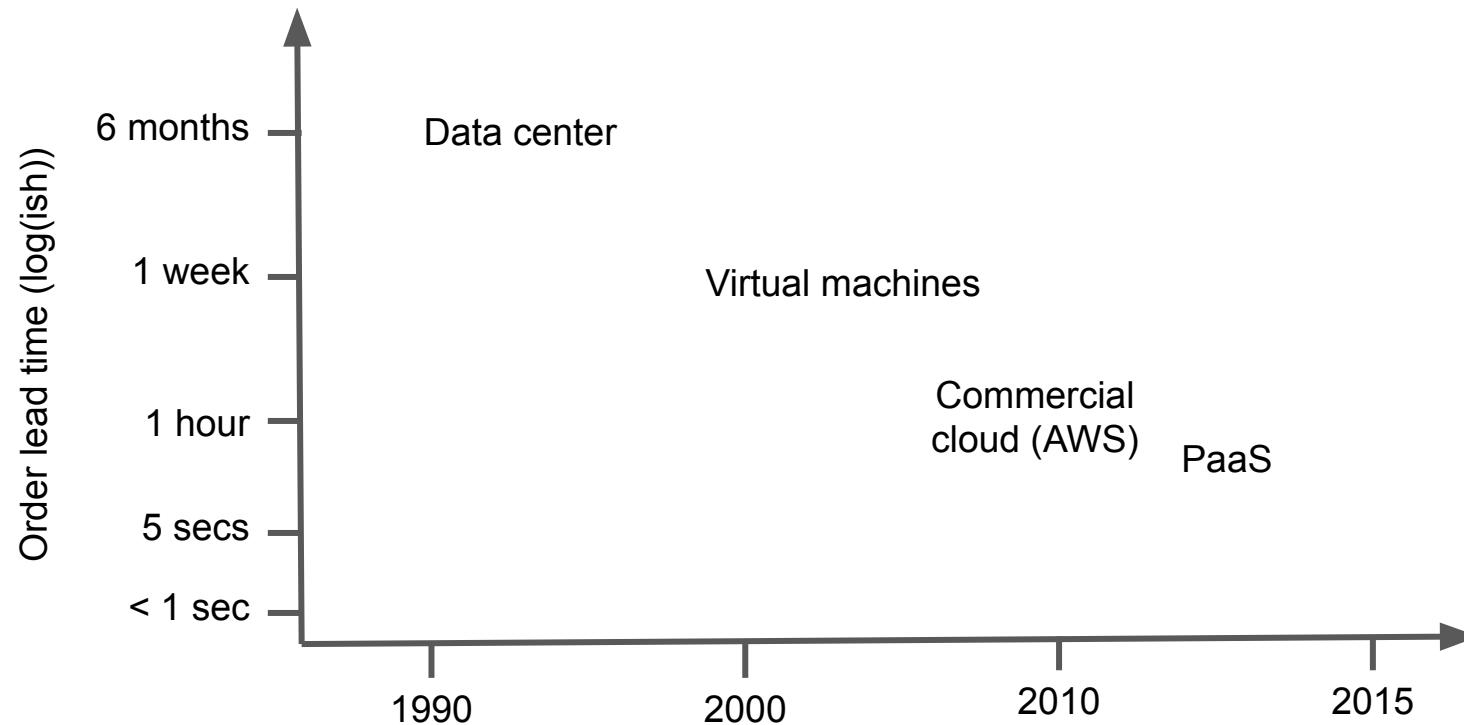
Hardware lead times



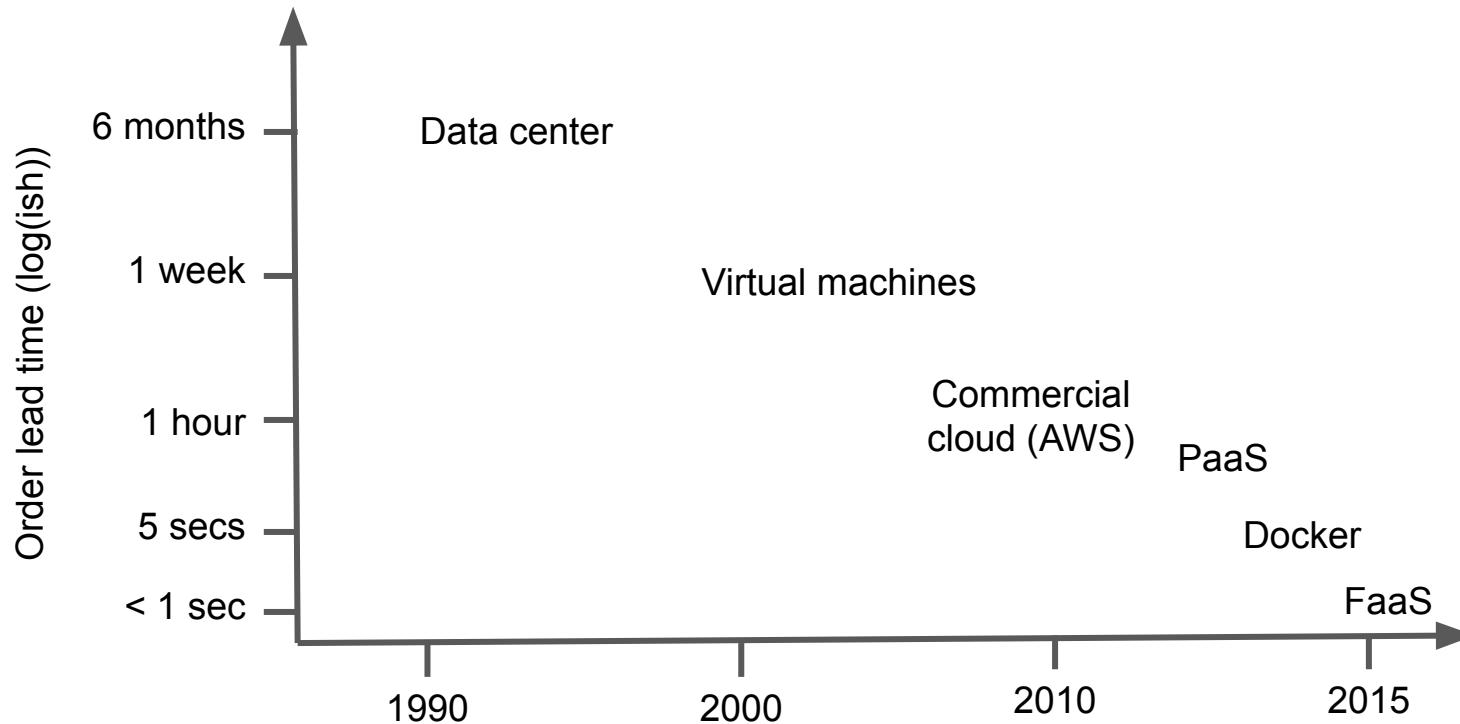
Hardware lead times



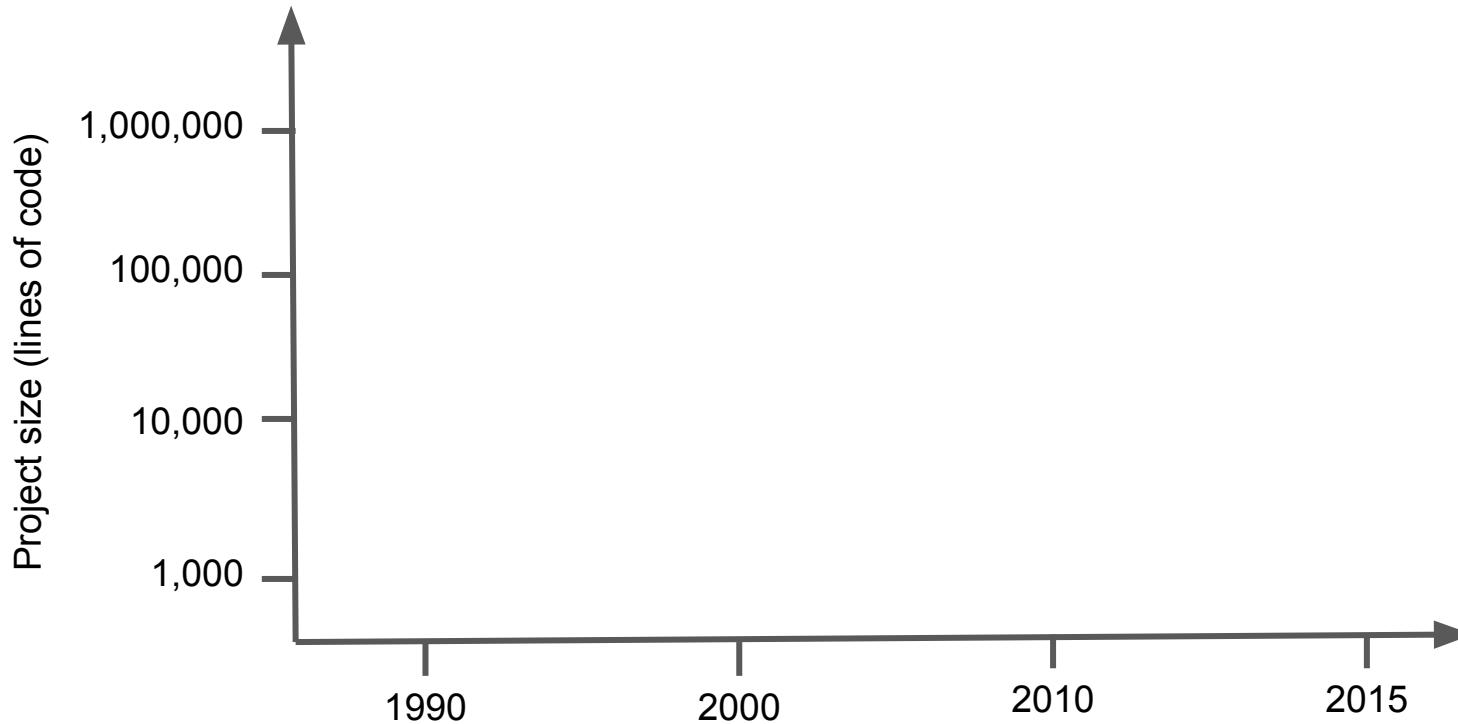
Hardware lead times



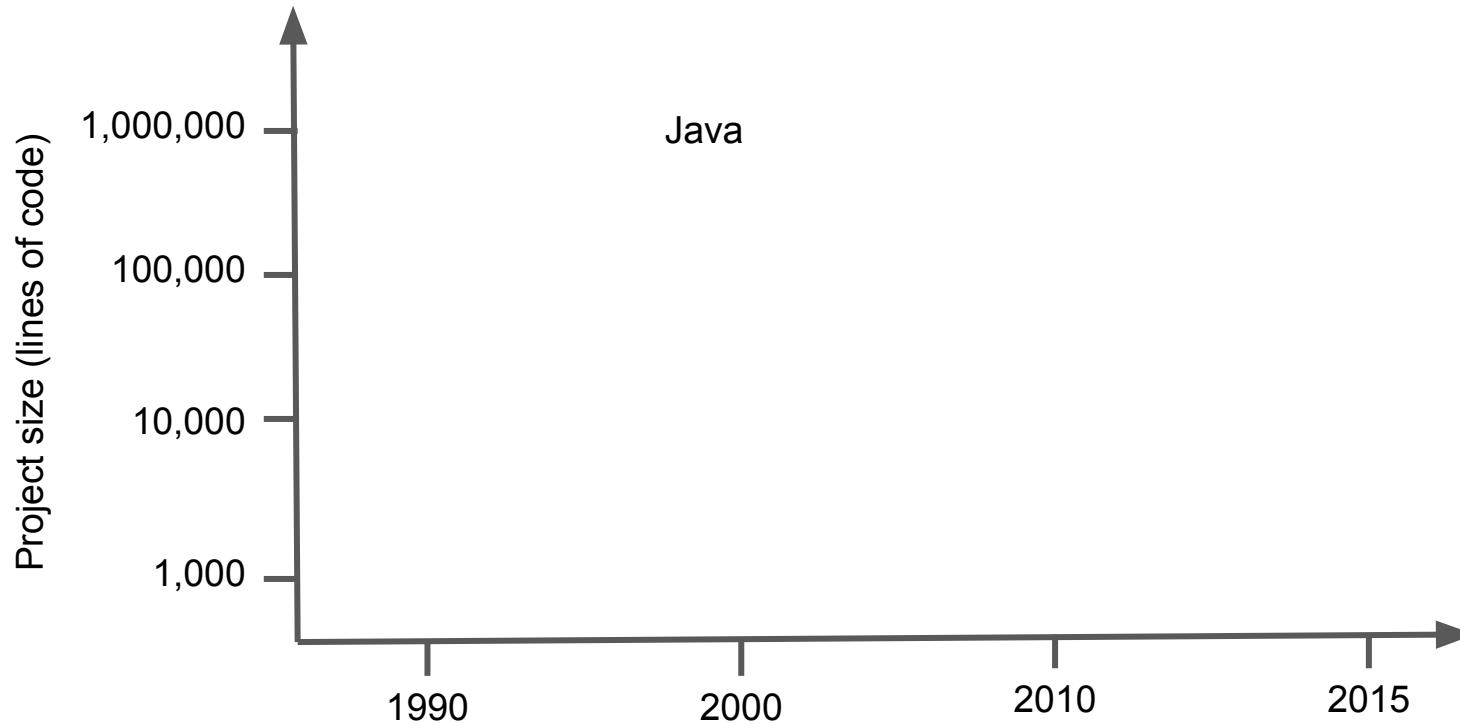
Hardware lead times



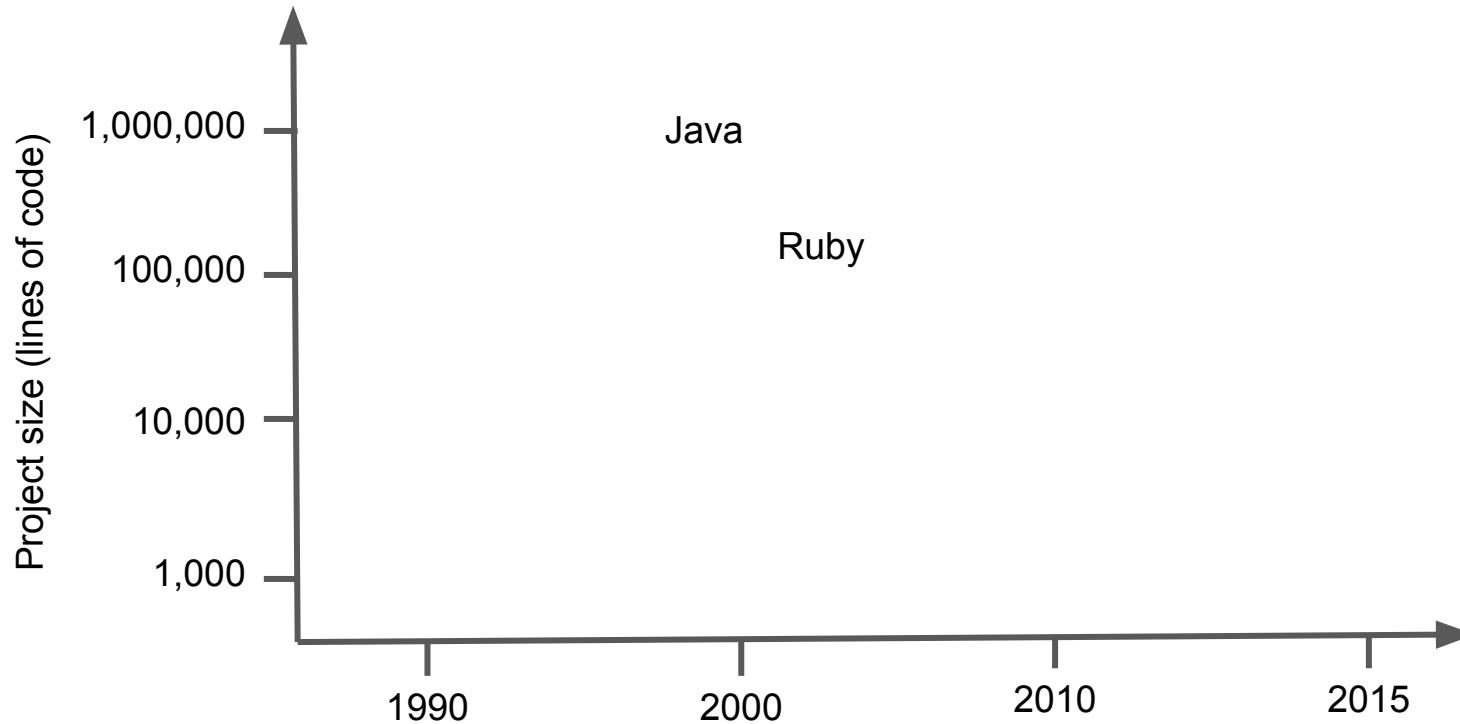
Project size



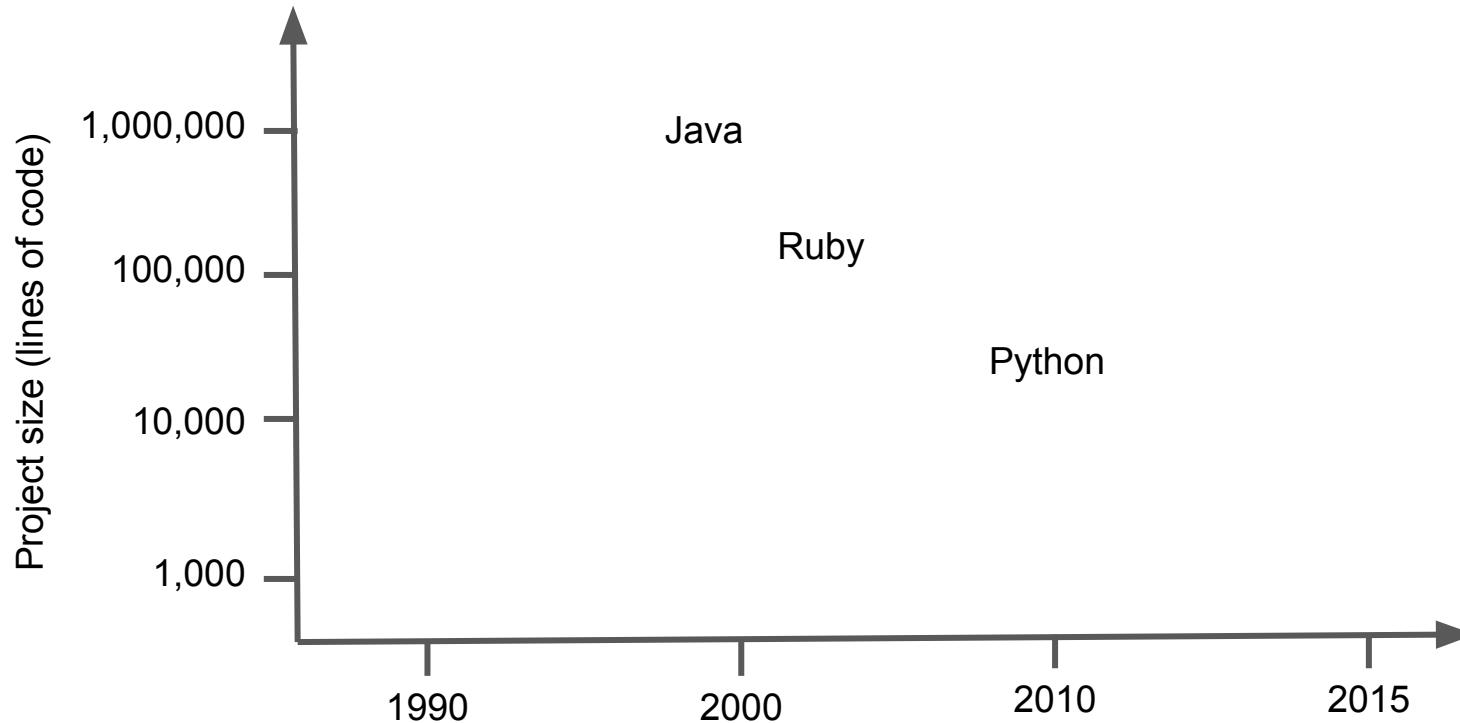
Project size



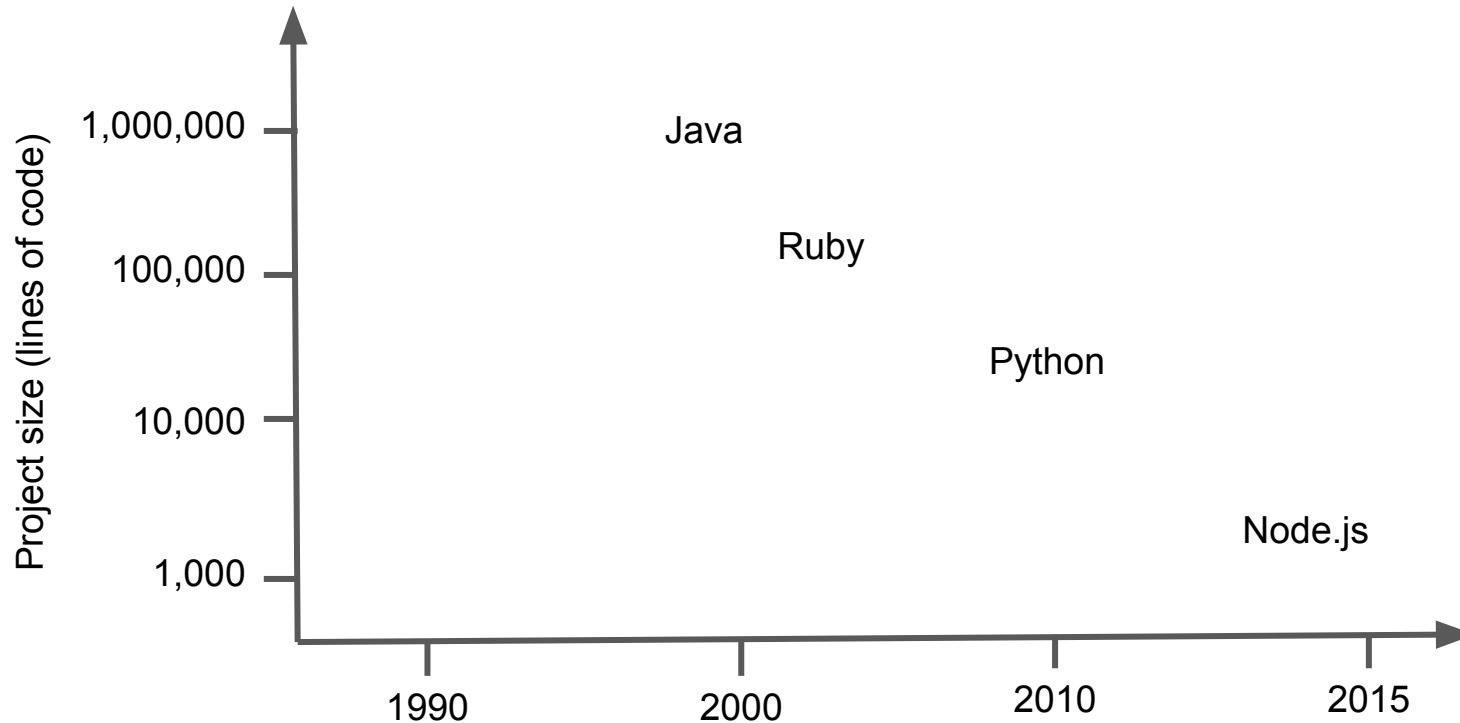
Project size



Project size



Project size



Projects are
being
delivered
faster

Projects are
being
delivered
faster

Hardware is
quicker to
provision

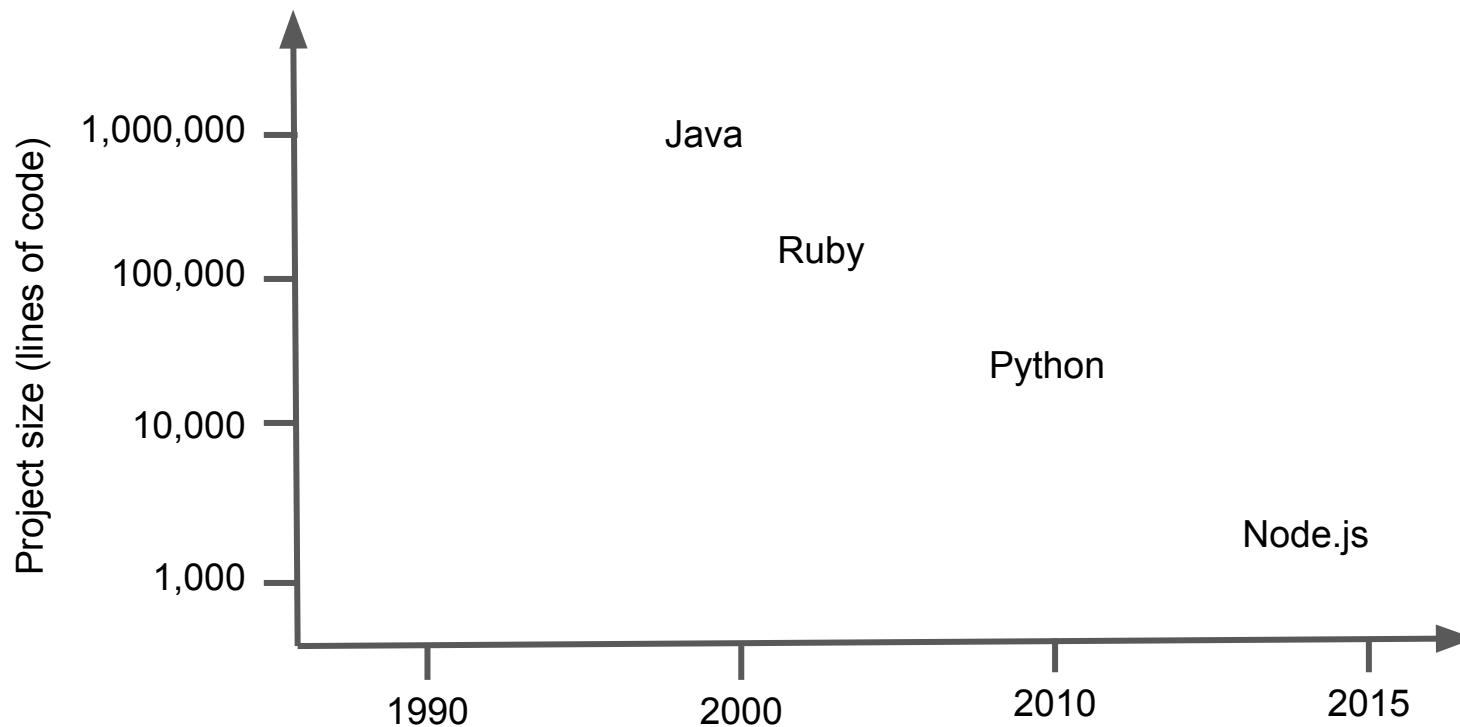
Projects are
being
delivered
faster

Hardware is
quicker to
provision

Projects are
getting
smaller

Microservices are an **emergent**
architecture

Project size

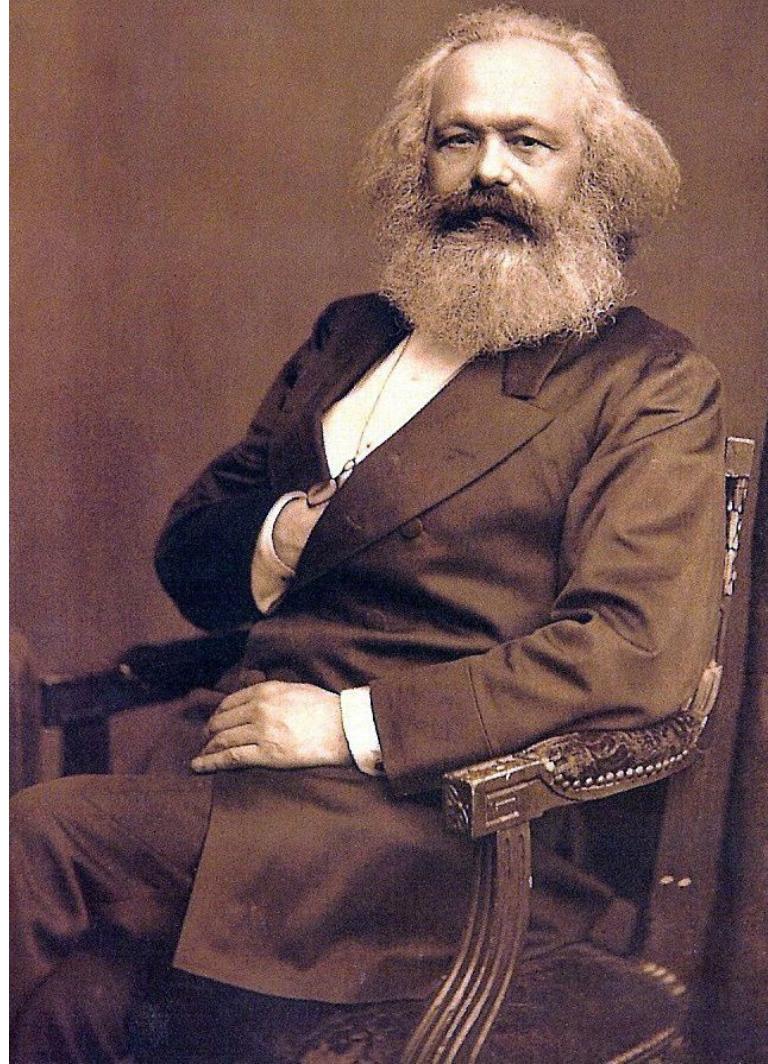


Onboarding the new dev...



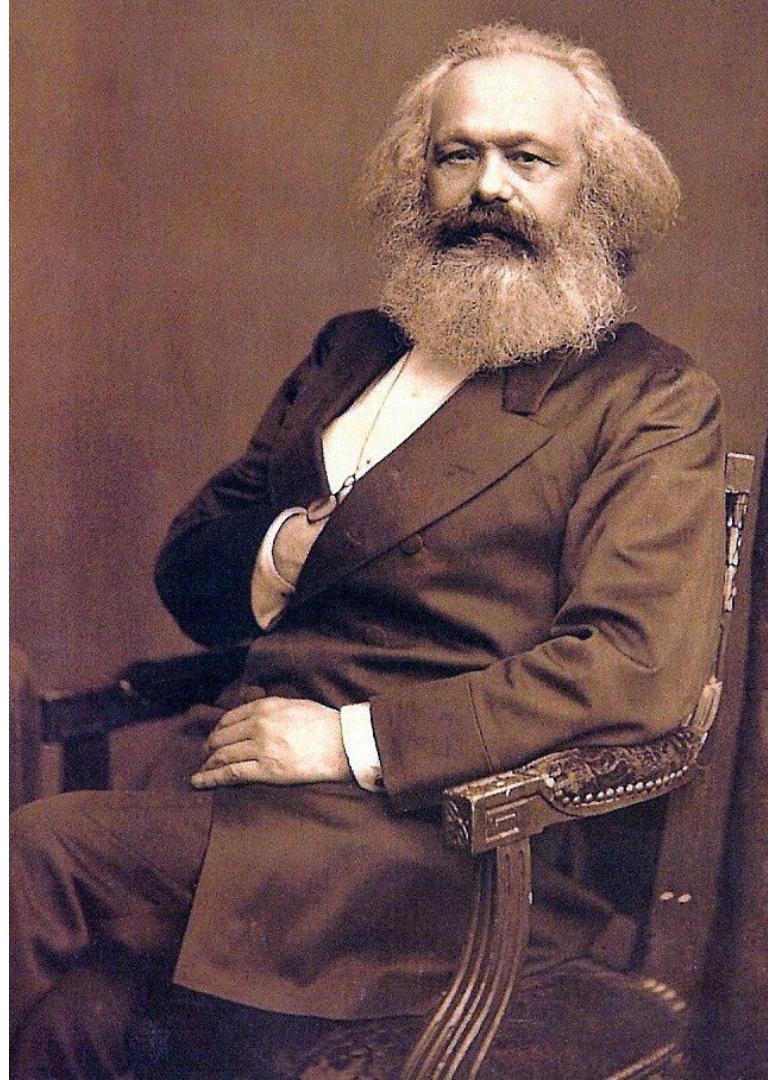
credit: <https://twitter.com/moonpolysoft/status/781868129269919744>

Marx on Microservices



Marx on Microservices

Economic and Philosophic Manuscripts of 1844

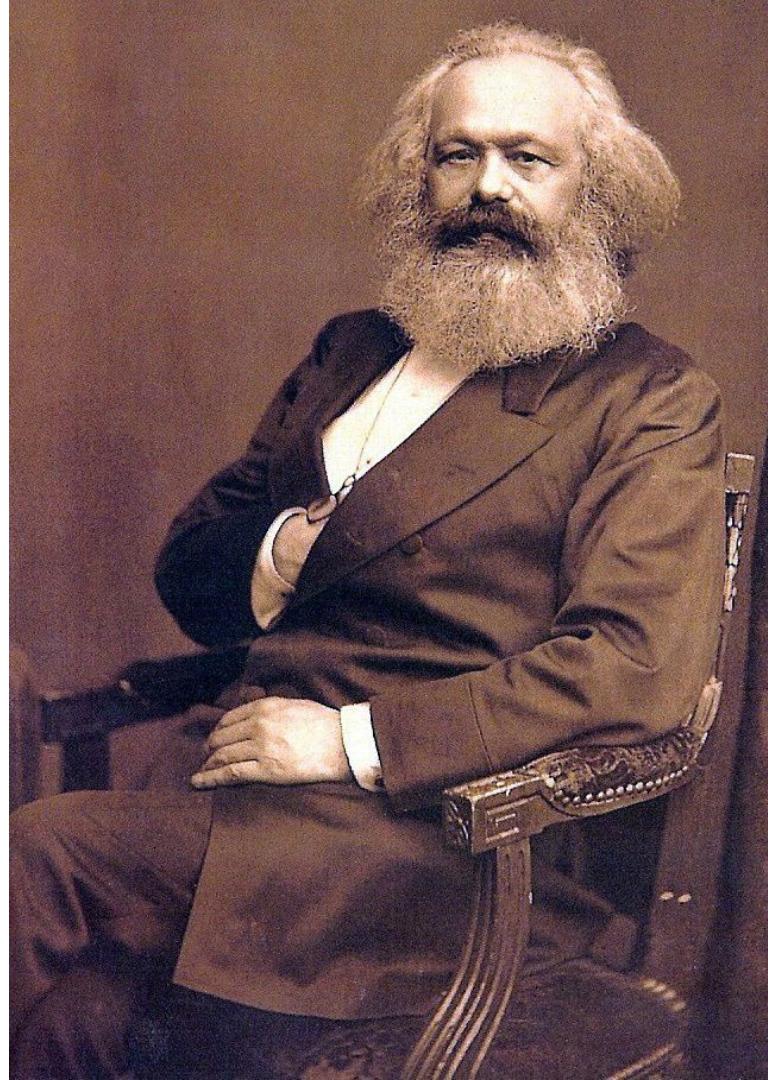


Marx on Microservices

Economic and Philosophic Manuscripts of 1844

Gattungswesen (species-being)

Workers feeling a connection to their work



Marx on Microservices

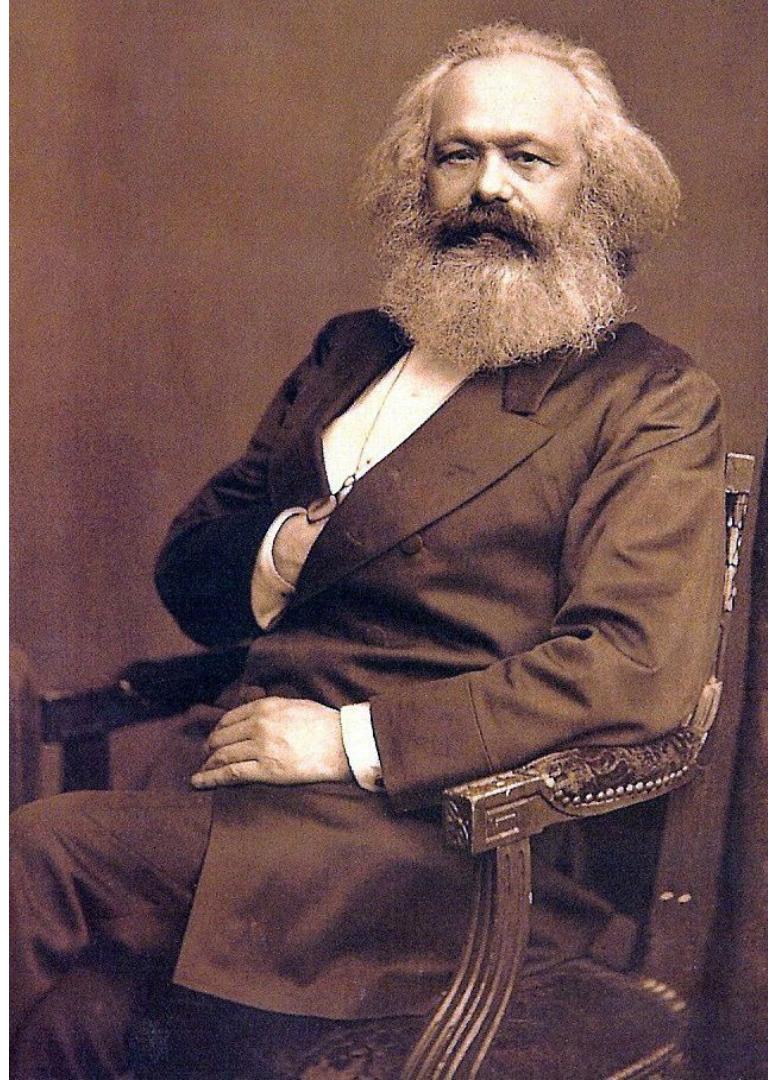
Economic and Philosophic Manuscripts of 1844

Gattungswesen (species-being)

Workers feeling a connection to their work

Entfremdung (alienation)

Workers feel estranged from their work



Products not Projects

“Any piece of software reflects the organizational structure that produced it”

Conway's Law

Inverse Conway Maneuver?

What sort of organisation do you want to work in?

The other, other story

SOA

Service Oriented Architecture

Service Oriented Architecture

Boundaries are explicit

Service Oriented Architecture

Boundaries are explicit

Services are autonomous

Service Oriented Architecture

Boundaries are explicit

Services are autonomous

Services share schema and contract, not class

Service Oriented Architecture

Boundaries are explicit

Services are autonomous

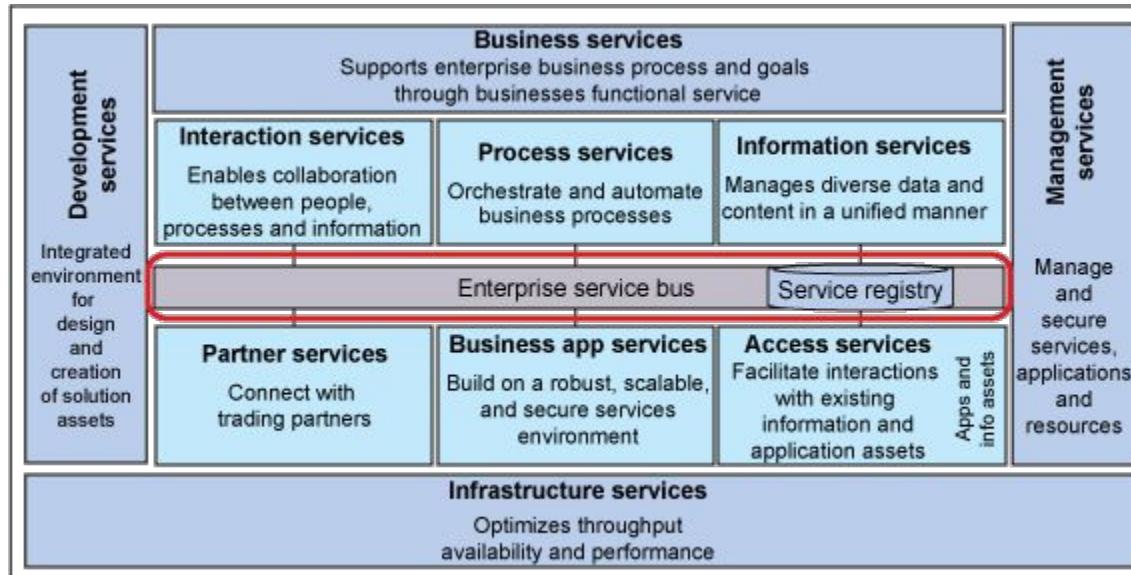
Services share schema and contract, not class

Service compatibility is based on policy

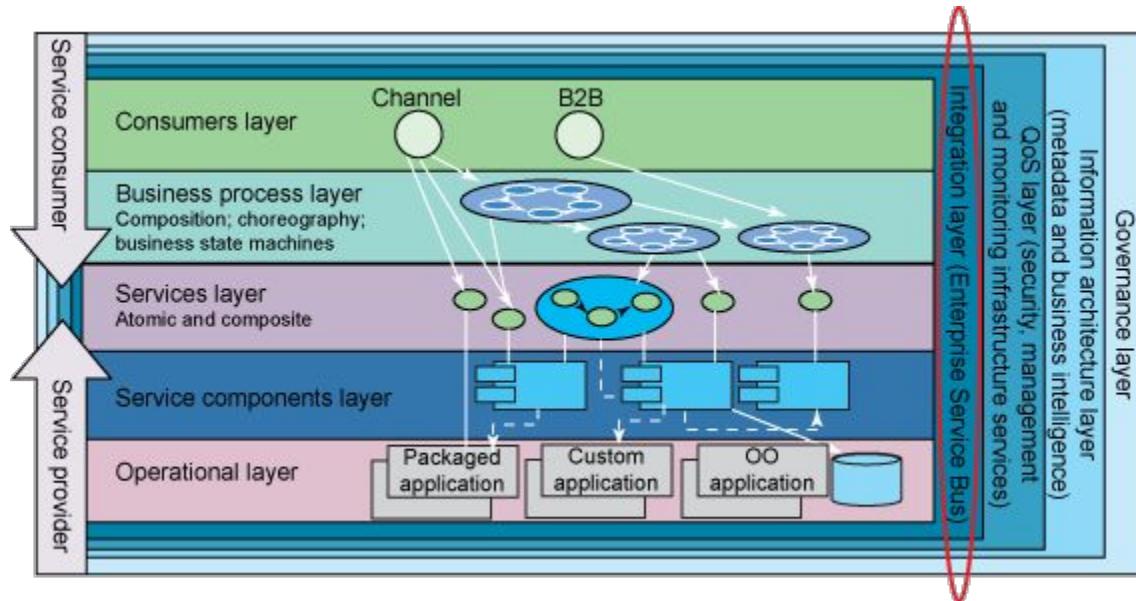
IBM SOA Foundation

IBM SOA Foundation Reference Architecture

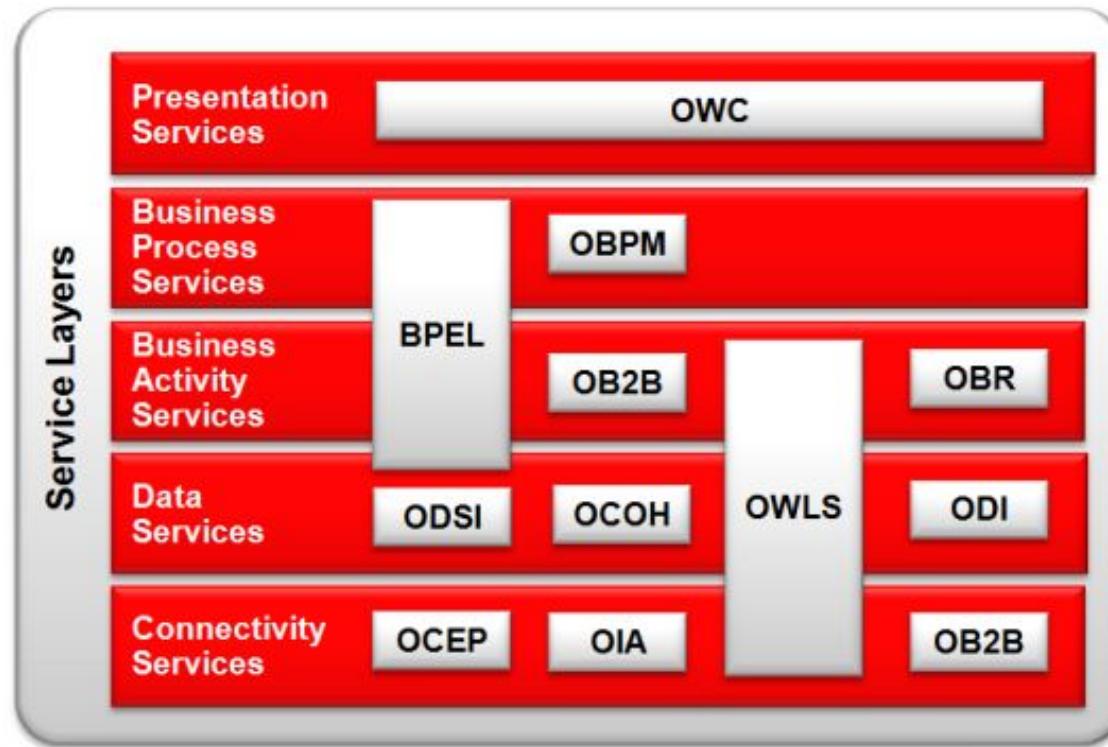
IBM SOA Foundation Reference Architecture logical model



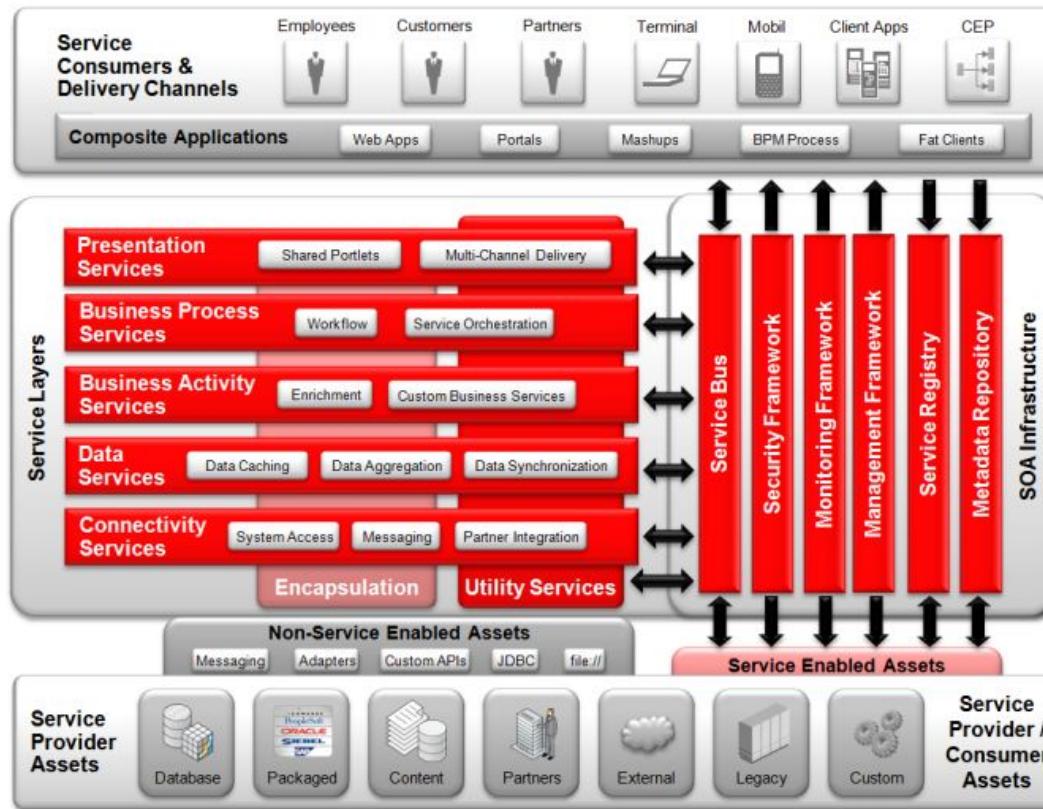
IBM SOA Foundation Reference Architecture solution



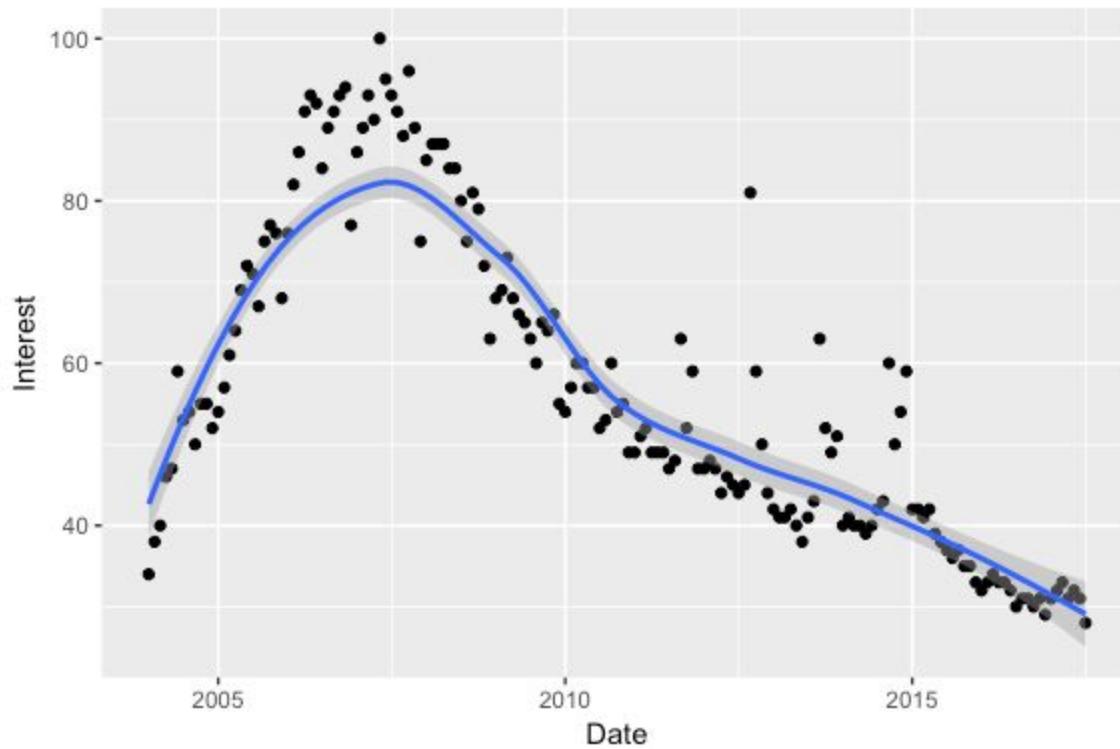
Oracle SOA Foundation Reference Architecture (Product Mapping)



Oracle SOA Foundation Reference Architecture



Google Trends: SOA



“The latest shiny new technology will not make things better... If you want **spectacular gains**, then you need to make a **spectacular commitment to change.**”

Anne Thomas Manes, SOA is Dead; Long Live Services

“... and that's where we need to concentrate
from this point forward: **Services**”

Anne Thomas Manes, SOA is Dead; Long Live Services

Why you shouldn't do Microservices

CAP Theorem

Dr. Eric Brewer (2000)

CAP Theorem

Consistency

Availability

Partition tolerance

CAP Theorem

Consistency

(all requests return the correct results)

Availability

Partition tolerance

CAP Theorem

Consistency

(all requests return the correct results)

Availability

(all requests complete)

Partition tolerance

CAP Theorem

Consistency

(all requests return the correct results)

Availability

(all requests complete)

Partition tolerance

(the network may fail)

CAP Theorem

Consistency

(all requests return the correct results)

Availability

(all requests complete)

Partition tolerance

(the network may fail)

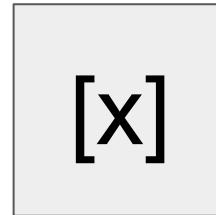
Pick 2

CAP Theorem - Example 1

User 1



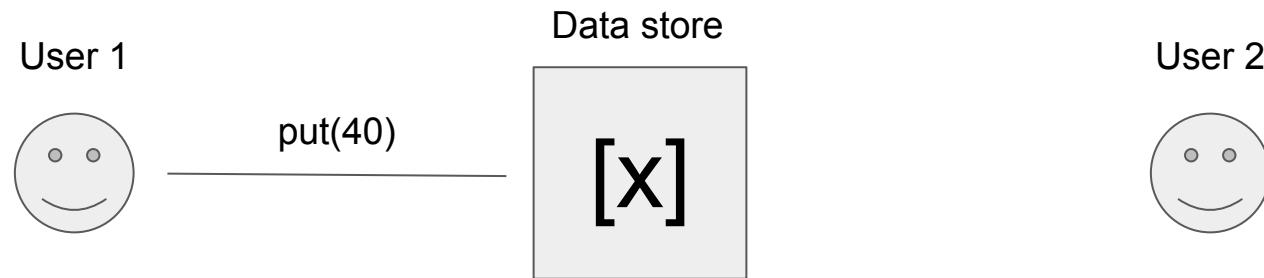
Data store



User 2



CAP Theorem - Example 1



CAP Theorem - Example 1

User 1



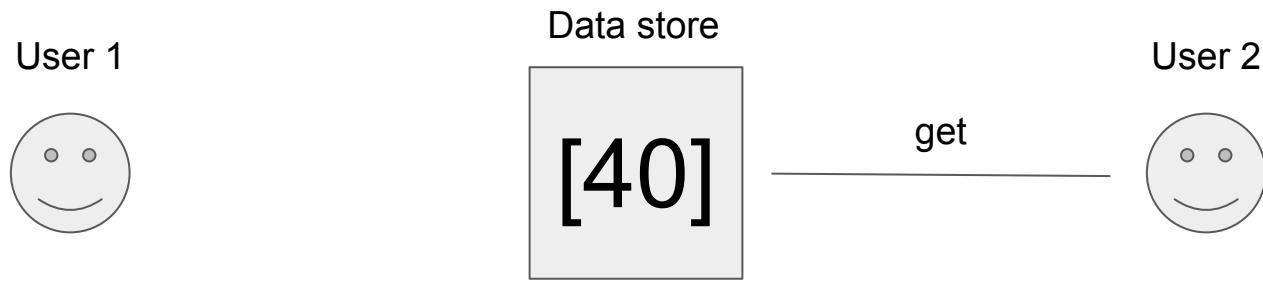
Data store

[40]

User 2



CAP Theorem - Example 1

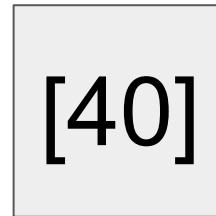


CAP Theorem - Example 1

User 1



Data store



User 2



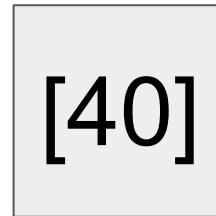
40

CAP Theorem - Example 1

User 1



Data store



User 2



40

- ✓ Consistent (correct)
- ✓ Available (answered)
- ✗ Partition tolerant (no network partitions)

CAP Theorem - Example 1



- ✓ Consistent (correct)
- ✓ Available (answered)
- ✗ Partition tolerant (no network partitions)

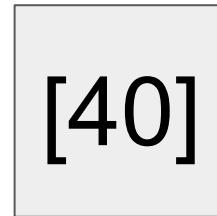
[PROBLEM] No real world system looks like this

CAP Theorem - Example 2

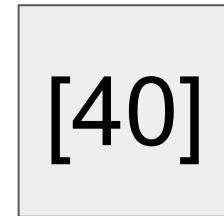
User 1



Data store (US)



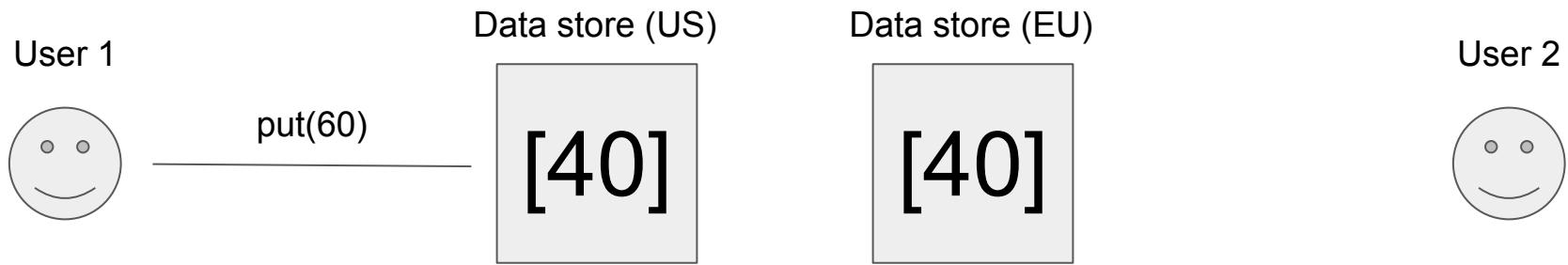
Data store (EU)



User 2



CAP Theorem - Example 2

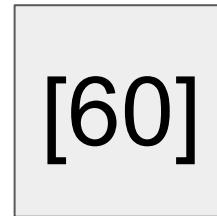


CAP Theorem - Example 2

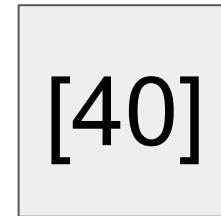
User 1



Data store (US)



Data store (EU)



User 2

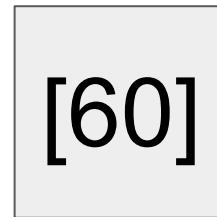


CAP Theorem - Example 2

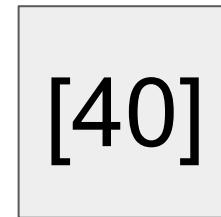
User 1



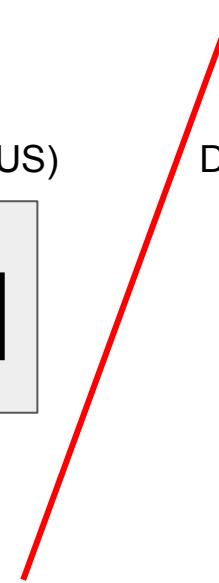
Data store (US)



Data store (EU)



User 2

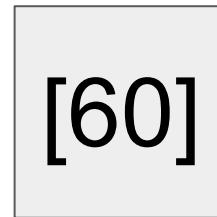


CAP Theorem - Example 2

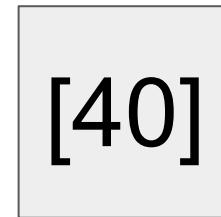
User 1



Data store (US)



Data store (EU)



get

User 2



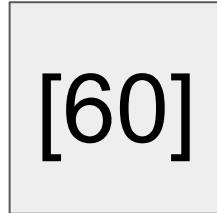
What now?

CAP Theorem - Example 2

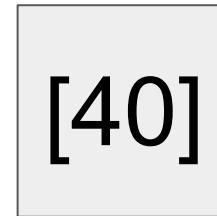
User 1



Data store (US)

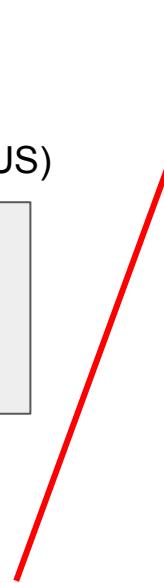


Data store (EU)



get → 40

User 2

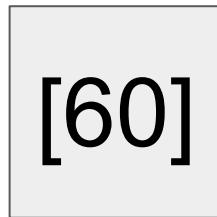


CAP Theorem - Example 2

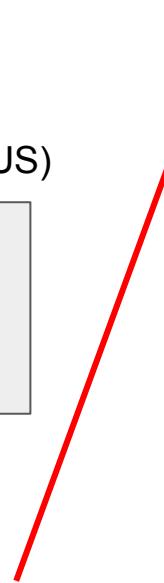
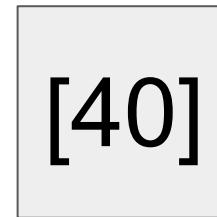
User 1



Data store (US)



Data store (EU)



get → 40

User 2



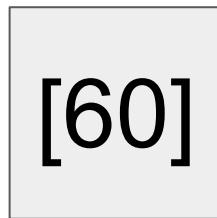
Consistent (incorrect)
Available (answered)
Partition tolerant

CAP Theorem - Example 2

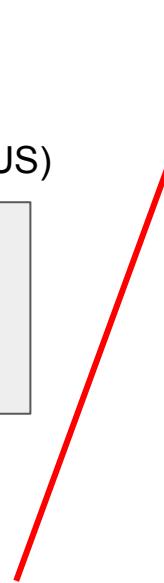
User 1



Data store (US)



Data store (EU)



get → 40

User 2



Consistent (incorrect)
Available (answered)
Partition tolerant



User 2

wait

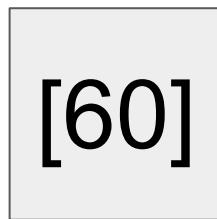


CAP Theorem - Example 2

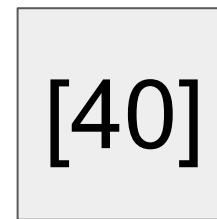
User 1



Data store (US)



Data store (EU)



get → 40

User 2



- Consistent (incorrect)
- Available (answered)
- Partition tolerant

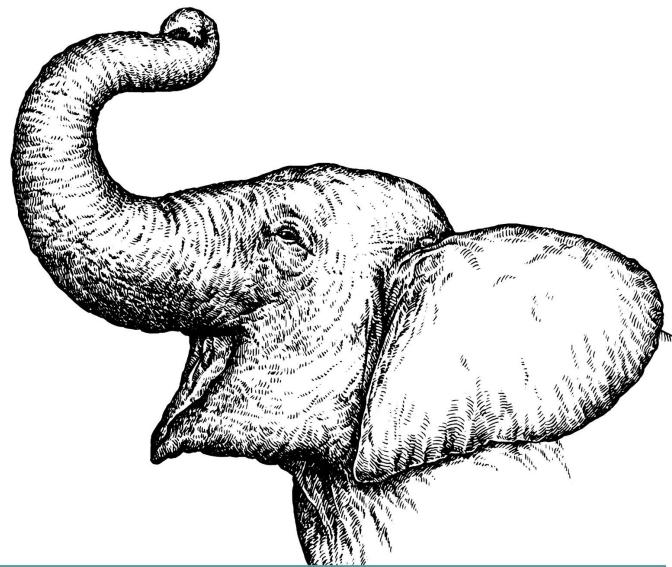
User 2

wait



- Consistent (correct)
- Available (not answered)
- Partition tolerant

The answer to every programming question ever conceived



It Depends

The Definitive Guide

O RLY?

@ThePracticalDev

Profile picture?

Profile picture?

Bank balance?

Profile picture?

ATM withdrawal?

Bank balance?

Profile picture?

Football scores?

ATM withdrawal?

Bank balance?

Profile picture?

Football scores?

ATM withdrawal?

Bank balance?

Parcel tracker?

Pick 2

Pick 2

Understand the tradeoffs

Pick 2

All distributed systems have to deal with CAP

All distributed systems have to deal with CAP

Microservices add CAP **where it wasn't before**

“Microservices are great for turning method calls
in to distributed computing problems”

Aaron Patterson

“A distributed system is one in which the failure of
a computer you didn't even know existed can
render your own computer unusable”

Leslie Lamport

The fallacies of distributed computing

The network is reliable

Latency is zero

Bandwidth is infinite

The network is secure

Topology doesn't change

There is one administrator

Transport cost is zero

The network is homogeneous

The network is reliable

~~The network is reliable~~

~~The network is reliable~~

Infrastructure is attacked

~~The network is reliable~~

Infrastructure is attacked



~~The network is reliable~~

Infrastructure is attacked

Networks are attacked



~~The network is reliable~~

Infrastructure is attacked

Networks are attacked

Machines fail



~~The network is reliable~~

Infrastructure is attacked

Networks are attacked

Machines fail

```
$traceroute wikipedia.org
traceroute to wikipedia.org (66.230.200.100), 64 hops max, 44 byte packets
 1 124.ae0.xr1.3d12.xs4all.net (194.109.21.1)  0.305 ms  0.360 ms  0.405 ms
 2 0.so-6-0-0.xr1.tc2.xs4all.net (194.109.5.10)  0.634 ms  0.716 ms  0.673 ms
 3 ams-ix-c00.wvfiber.net (195.69.145.58)  0.638 ms  0.601 ms  0.551 ms
 4 lon-c00-pos-4-0.OC48-ams-pos11-0.wvfiber.net (63.223.28.201)  7.512 ms  7.427 ms  7.494 ms
 5 nyc60-pos-1-0.OC48-lon-c00-pos-3-0.wvfiber.net (63.223.28.145)  84.108 ms  83.804 ms  83.995 ms
 6 66.216.1.181 (66.216.1.181)  83.435 ms  83.278 ms  83.348 ms
 7 ash-c01-tge-3-3.TG-nyc-c01-1-1.wvfiber.net (66.216.1.161)  89.563 ms  89.554 ms  89.551 ms
 8 atl-c01-tge-3-1.TG-ash-c01-3-1.wvfiber.net (66.216.1.157)  103.701 ms  103.606 ms  103.596 ms
 9 cpp-hostway.wvfiber.net (63.223.8.26)  103.678 ms  103.609 ms  103.630 ms
10 e1-12.co2.as30217.net (64.156.25.105)  113.014 ms  113.044 ms  113.084 ms
11 10ge5-1.csv5-pmptpa.wikimedia.org (84.40.25.102)  113.153 ms  113.251 ms  113.180 ms
12 rr.pmptpa.wikimedia.org (66.230.200.100)  113.069 ms  113.172 ms  113.003 ms
```



The latency is zero

~~The latency is zero~~

~~The latency is zero~~

US East to west (+~65ms)

~~The latency is zero~~

US East to west (+~65ms)

Atlantic crossing (+~90ms)

London to Auckland (+~270ms)

~~The latency is zero~~

US East to west (+~65ms)

Atlantic crossing (+~90ms)

London to Auckland (+~270ms)



~~The latency is zero~~

US East to west (+~65ms)

Atlantic crossing (+~90ms)

London to Auckland (+~270ms)

The case of the 500-mile email



Bandwidth is infinite

~~Bandwidth is infinite~~

~~Bandwidth is finite~~



~~Bandwidth is finite~~

VOIP, streaming video/audio...



~~Bandwidth is finite~~

VOIP, streaming video/audio...

1000s of services sharing a line?



made on imgur

~~Bandwidth is infinite~~

VOIP, streaming video/audio...

1000s of services sharing a line?

Even T-1 lines get saturated

Bottlenecks?

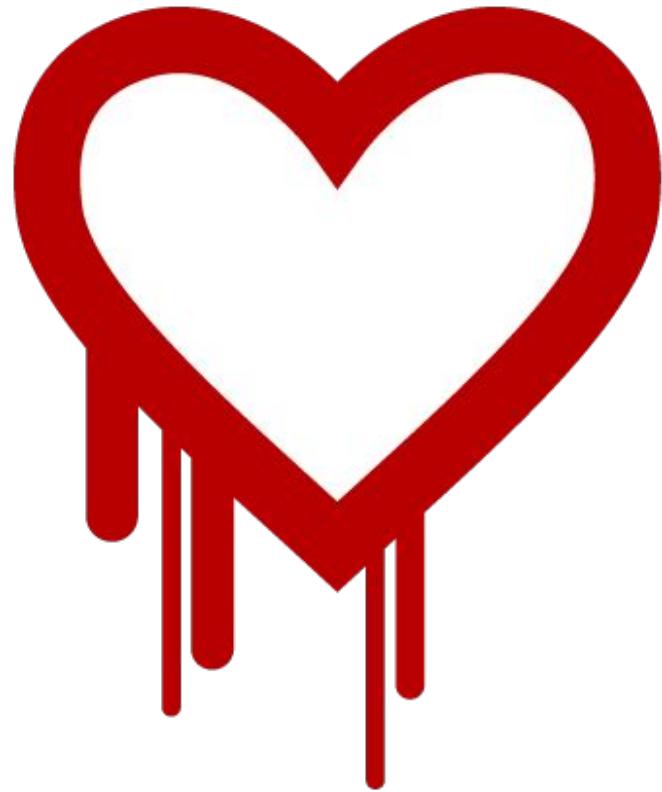


The network is secure

~~The network is secure~~

~~The network is secure~~

Heartbleed?



~~The network is secure~~

Heartbleed?

DynDNS?



“In a relatively short time we've taken a system built to resist destruction by nuclear weapons and made it vulnerable to toasters.”

Jeff Jarmoc

~~The network is secure~~

Heartbleed?

DynDNS?

Network border isn't good enough



"In a relatively short time we've taken a system built to resist destruction by nuclear weapons and made it vulnerable to toasters."

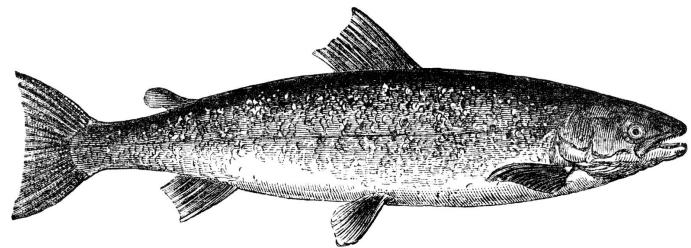
Jeff Jarmoc

~~The network is secure~~

Heartbleed?

DynDNS?

Network border isn't good enough



Expert

Hoping Nobody
Hacks You

Topology doesn't change

~~Topology doesn't change~~

~~Topology doesn't change~~

Topology is dynamic!

~~Topology doesn't change~~

Topology is dynamic! (Cattle not pets)

~~Topology doesn't change~~

Topology is dynamic! (Cattle not pets)

Can't rely on specific endpoints...

~~Topology doesn't change~~

Topology is dynamic! (Cattle not pets)

Can't rely on specific endpoints...

...or performance, routing etc.

~~Topology doesn't change~~

Topology is dynamic! (Cattle not pets)

Can't rely on specific endpoints...

...or performance, routing etc.

Bits a network can be unreachable

There is one administrator

~~There is one administrator~~

~~There is one administrator~~

Who grants access to database?

~~There is one administrator~~

Who grants access to database?

...and to another service?

~~There is one administrator~~

Who grants access to database?

...and to another service?



~~There is one administrator~~

Who grants access to database?

...and to another service?

Who do you complain to?



**KEEP
CALM
AND
CALL A
SYSADMIN**

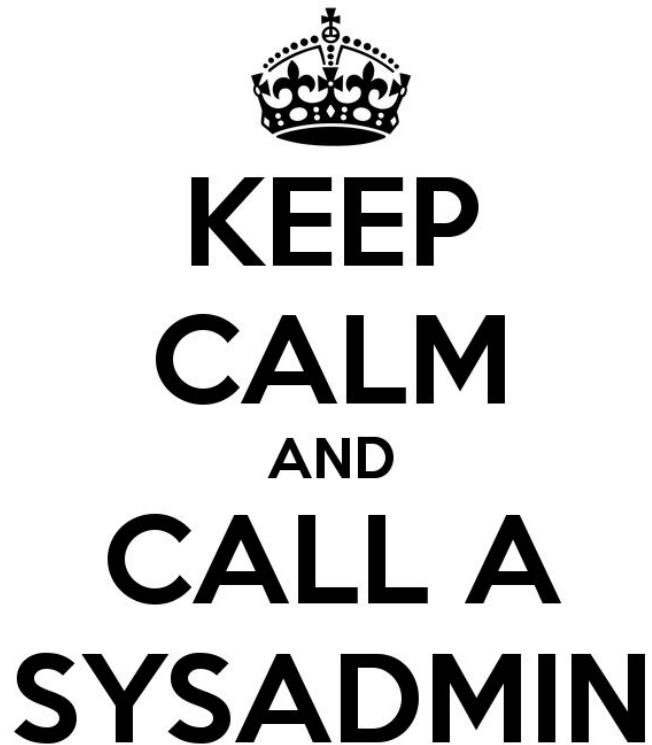
~~There is one administrator~~

Who grants access to database?

...and to another service?

Who do you complain to?

Coordinated updates?



Transport cost is zero

~~Transport cost is zero~~

~~Transport cost is zero~~

Literally not free

Data Transfer OUT From Amazon S3 To Internet

First 1 GB / month	\$0.000 per GB
Up to 10 TB / month	\$0.090 per GB
Next 40 TB / month	\$0.085 per GB
Next 100 TB / month	\$0.070 per GB
Next 150 TB / month	\$0.050 per GB

~~Transport cost is zero~~

Literally not free

Who's paying?

Data Transfer OUT From Amazon S3 To Internet

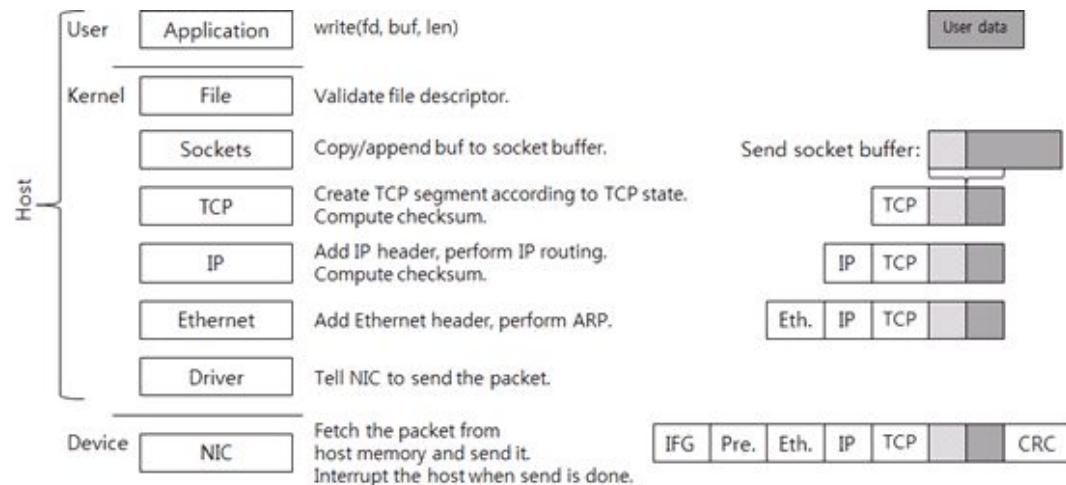
First 1 GB / month	\$0.000 per GB
Up to 10 TB / month	\$0.090 per GB
Next 40 TB / month	\$0.085 per GB
Next 100 TB / month	\$0.070 per GB
Next 150 TB / month	\$0.050 per GB

~~Transport cost is zero~~

Literally not free

Who's paying?

Marshalling isn't free either



The network is homogeneous

~~The network is homogeneous~~

~~The network is homogeneous~~

My pockets aren't homogeneous!



~~The network is homogeneous~~

My pockets aren't homogeneous!

Can't assume you're talking to UNIX etc.

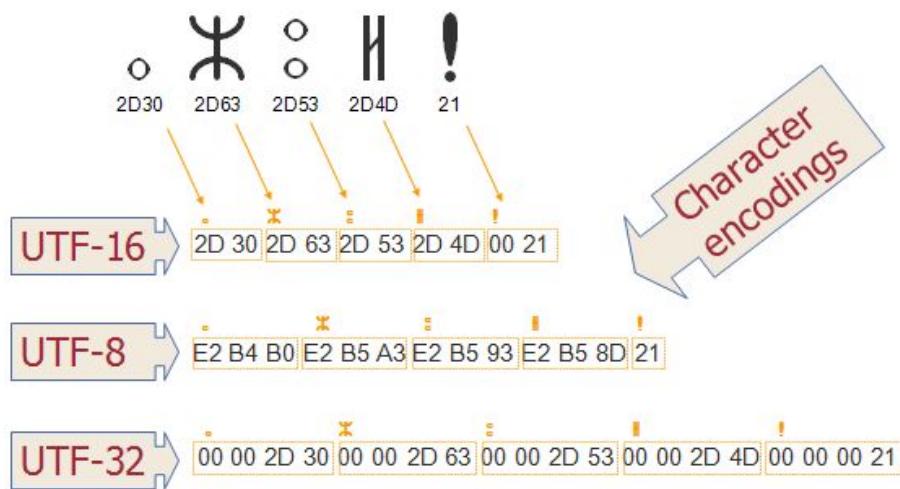


~~The network is homogeneous~~

My pockets aren't homogeneous!

Can't assume you're talking to UNIX etc.

Can't assume character encodings



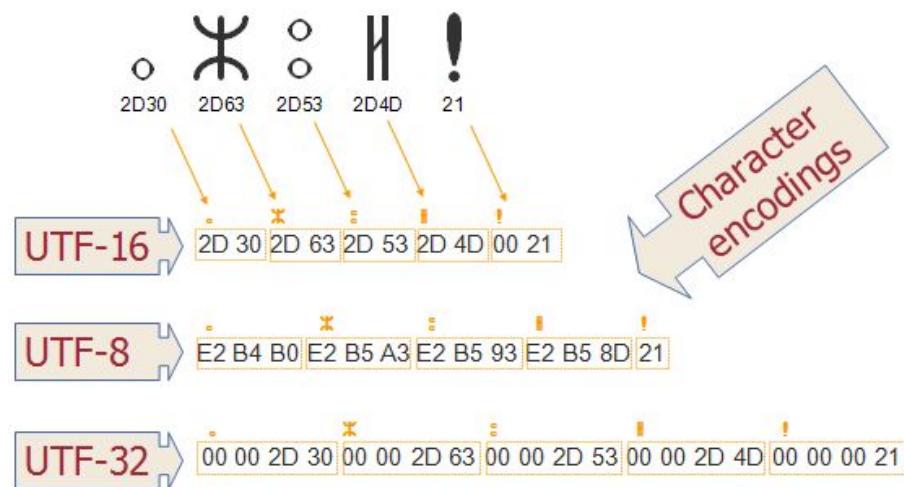
~~The network is homogeneous~~

My pockets aren't homogeneous!

Can't assume you're talking to UNIX etc.

Can't assume character encodings

Have to agree on exchange formats



The fallacies of distributed computing

The network is reliable

Latency is zero

Bandwidth is infinite

The network is secure

Topology doesn't change

There is one administrator

Transport cost is zero

The network is homogeneous

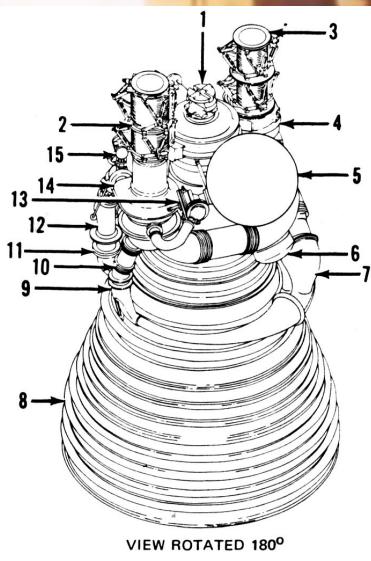
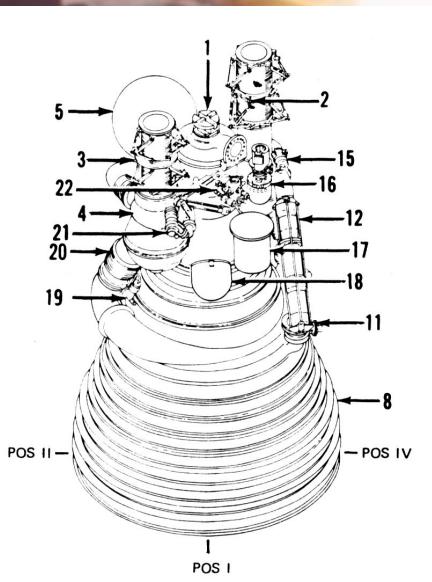
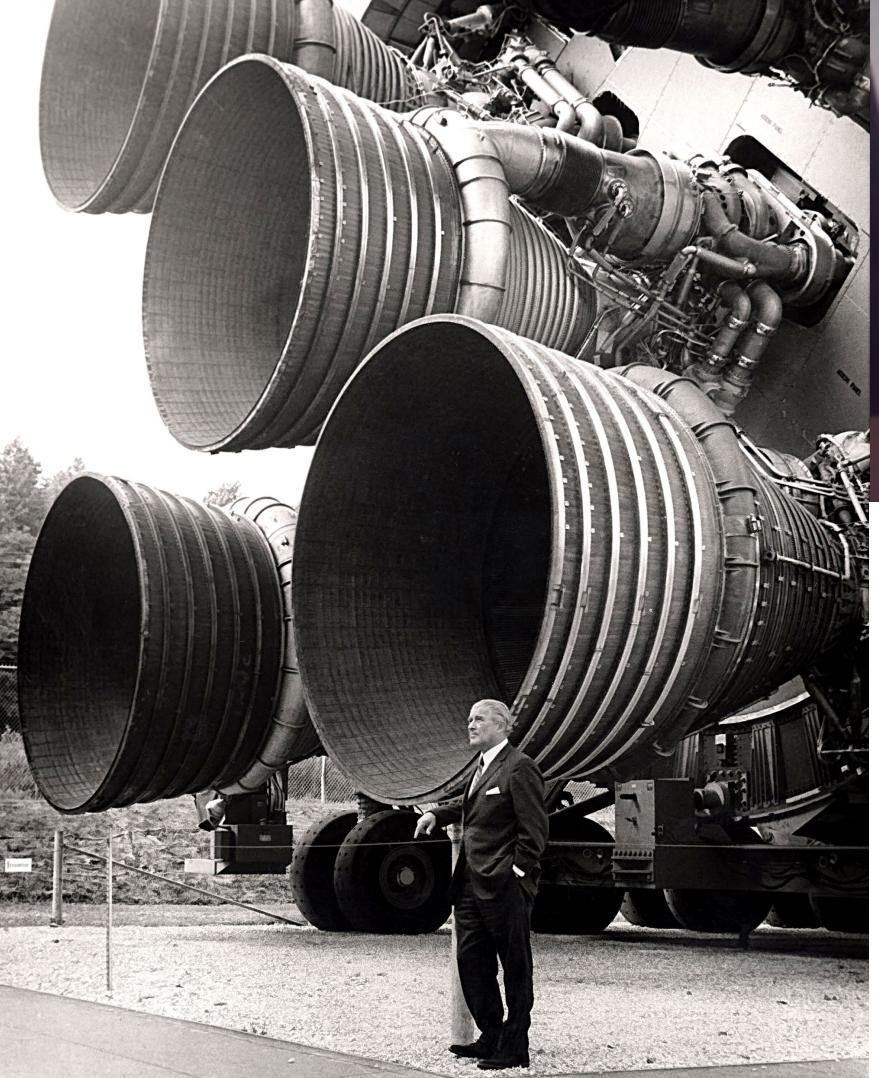


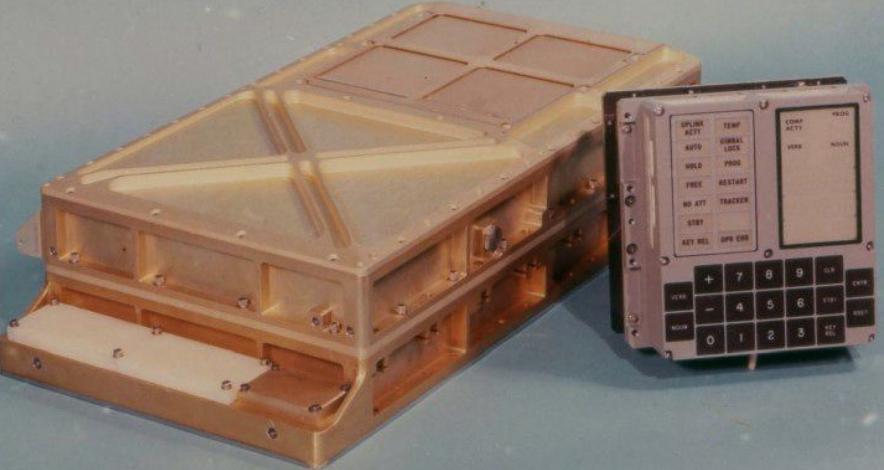
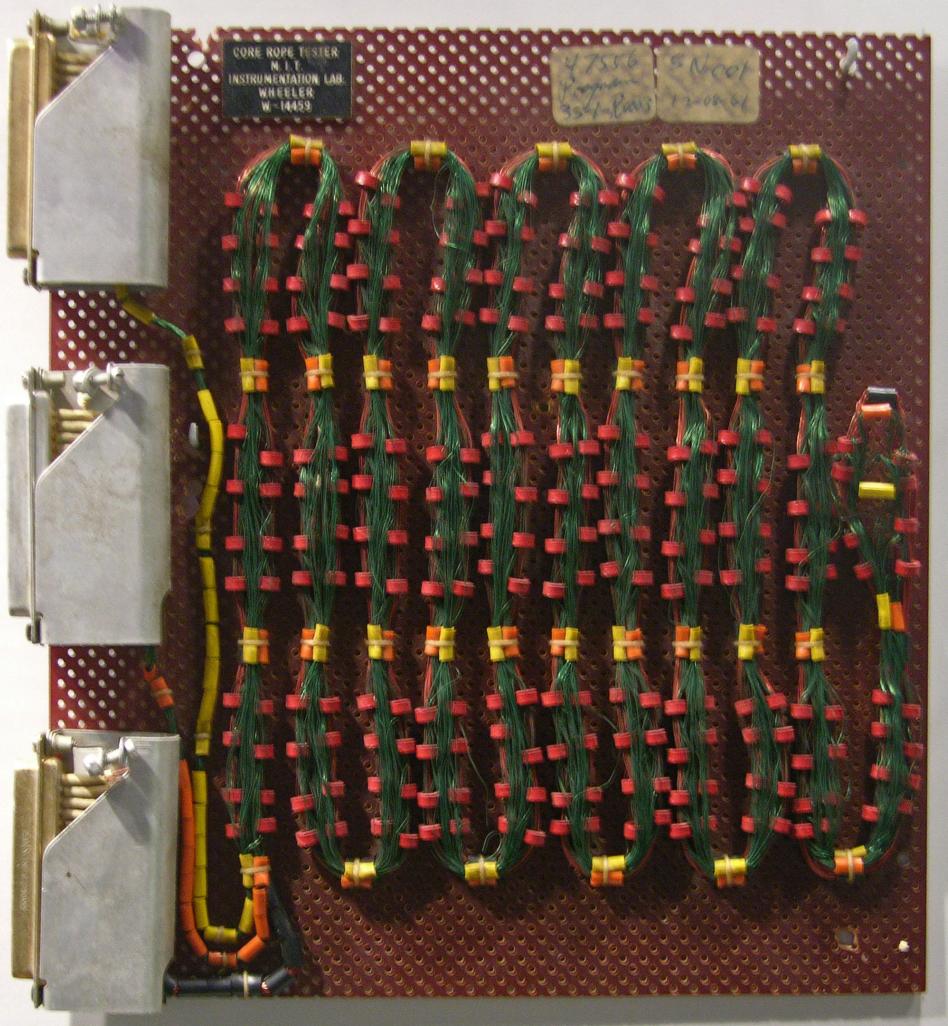
Aaron Cohen



Aaron Cohen
NASA









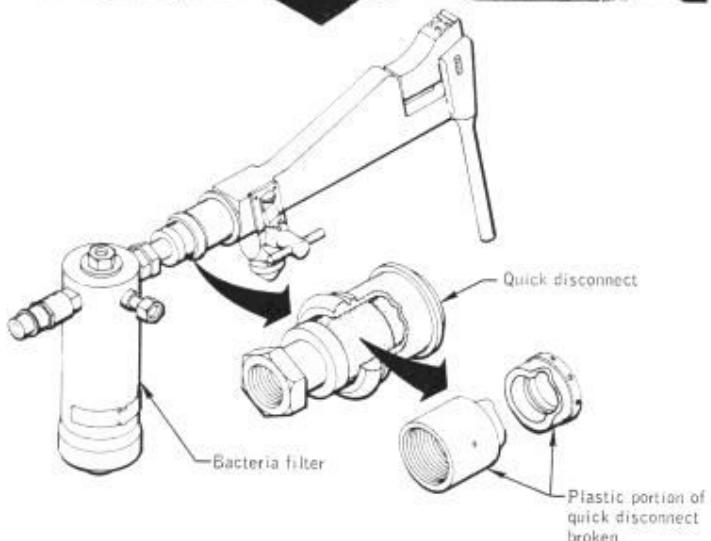
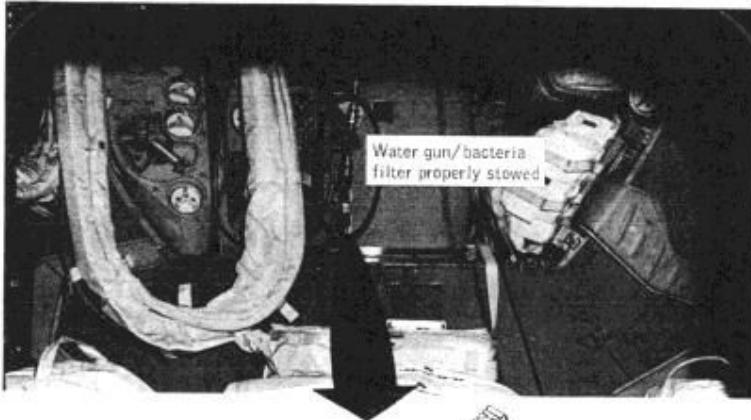
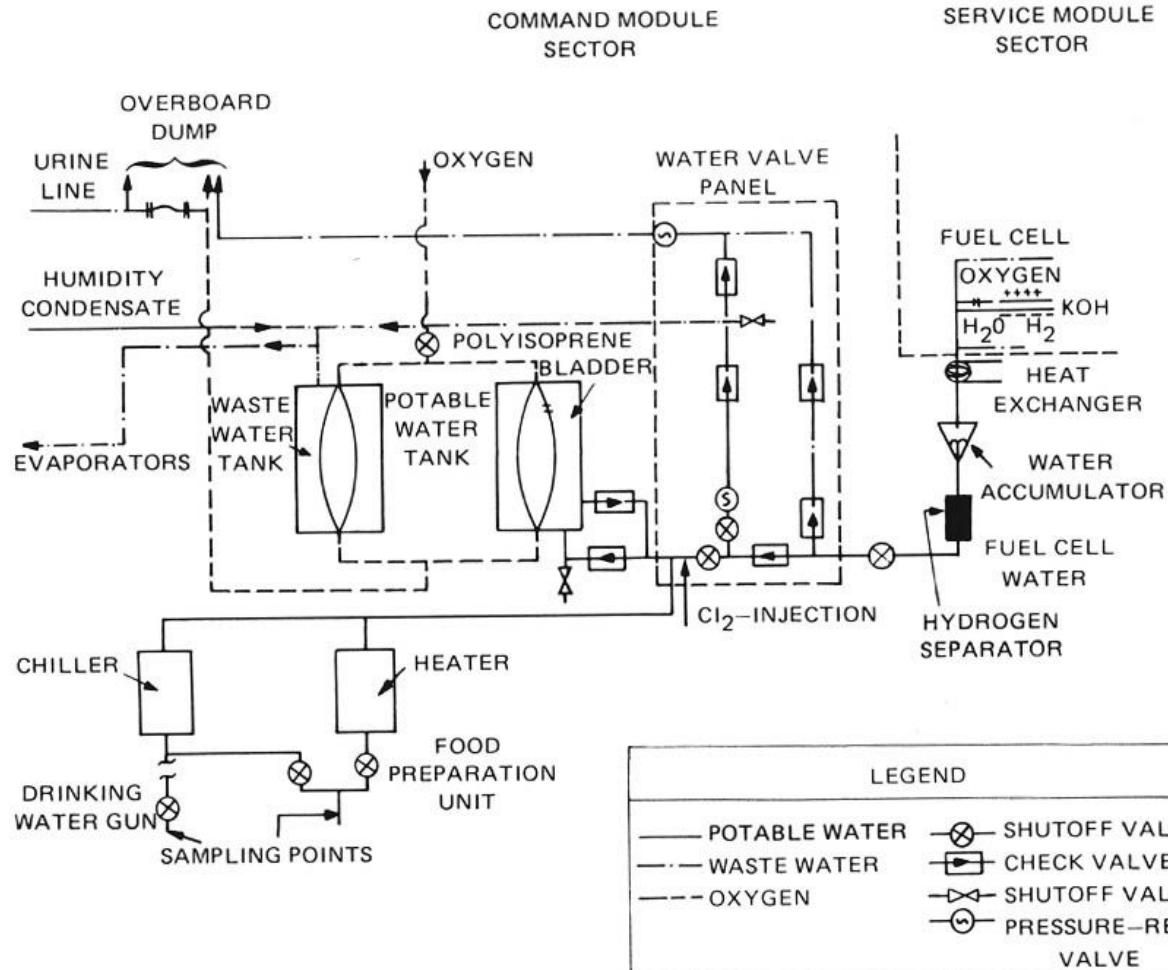
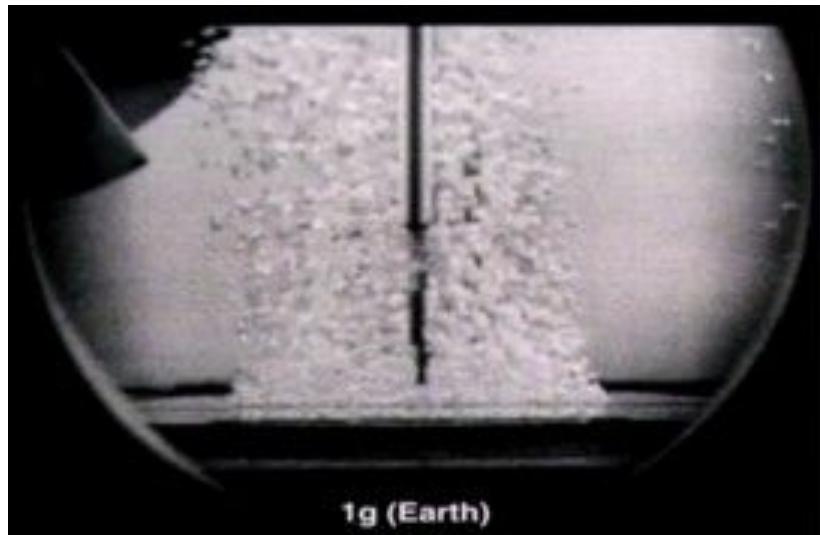
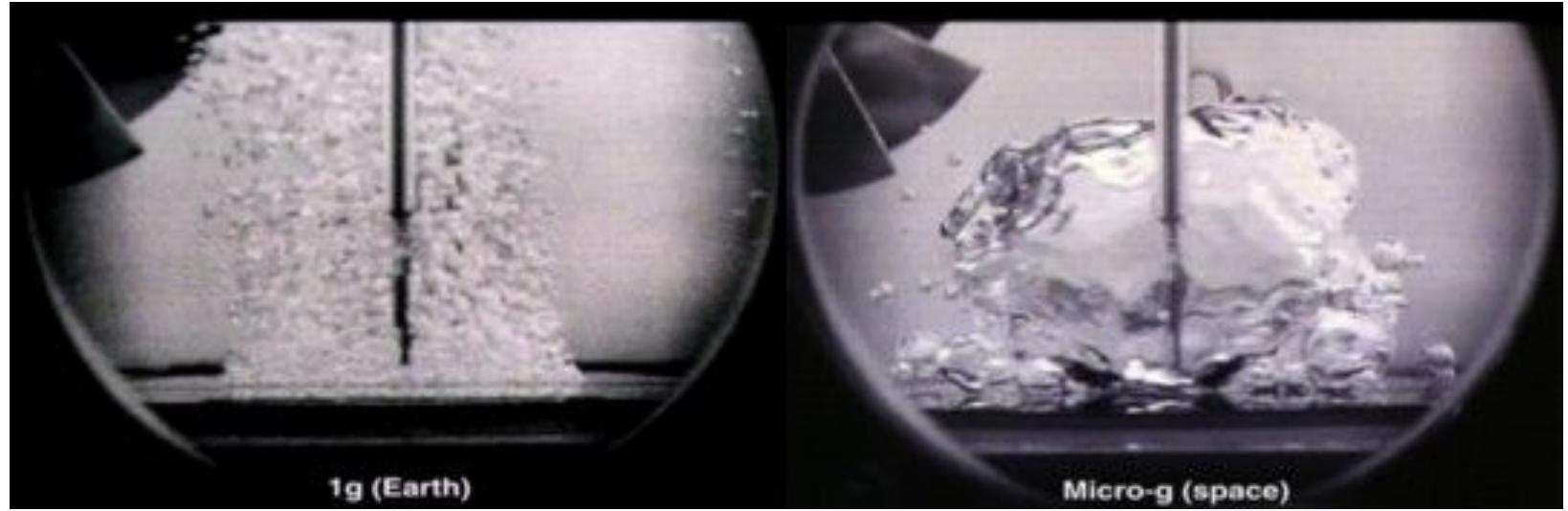


Figure 14-28.- Water gun/bacteria filter.













ZERO G
EXPERIMENT

Microservices in the Real World

Continuous Integration

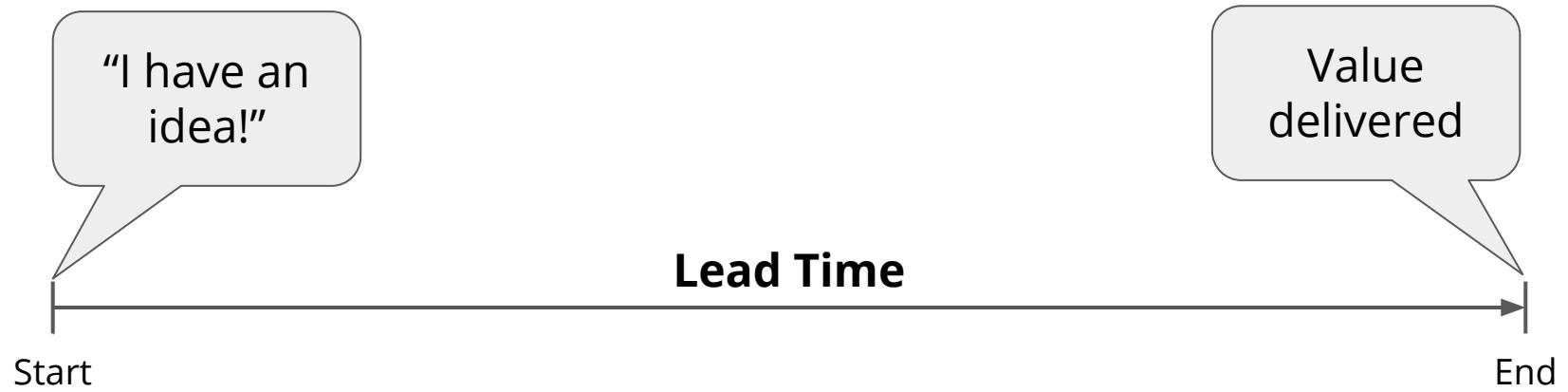
Continuous Delivery

Continuous Deployment

DevOps





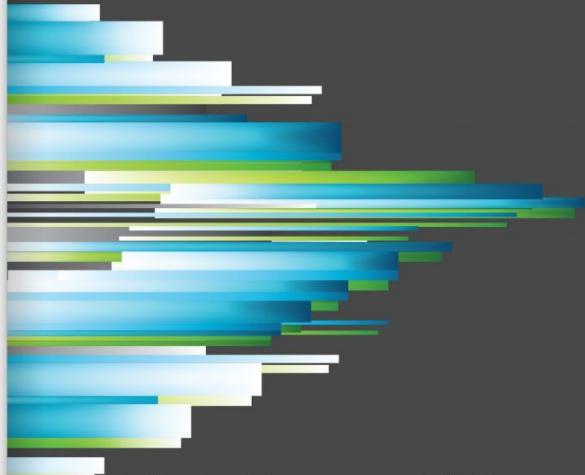


Goal:
Reduce Lead Time

THE SCIENCE OF LEAN SOFTWARE AND DEVOPS

ACCELERATE

Building and Scaling High Performing
Technology Organizations



Nicole Forsgren, PhD
Jez Humble, and Gene Kim

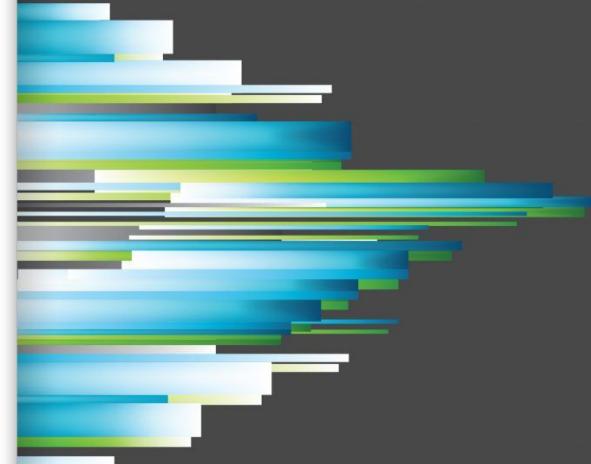
with forewords by Martin Fowler and Courtney Kissler
and a case study contributed by Steve Bell and Karen Whitley Bell

4 Key Metrics

THE SCIENCE OF LEAN SOFTWARE AND DEVOPS

ACCELERATE

Building and Scaling High Performing
Technology Organizations

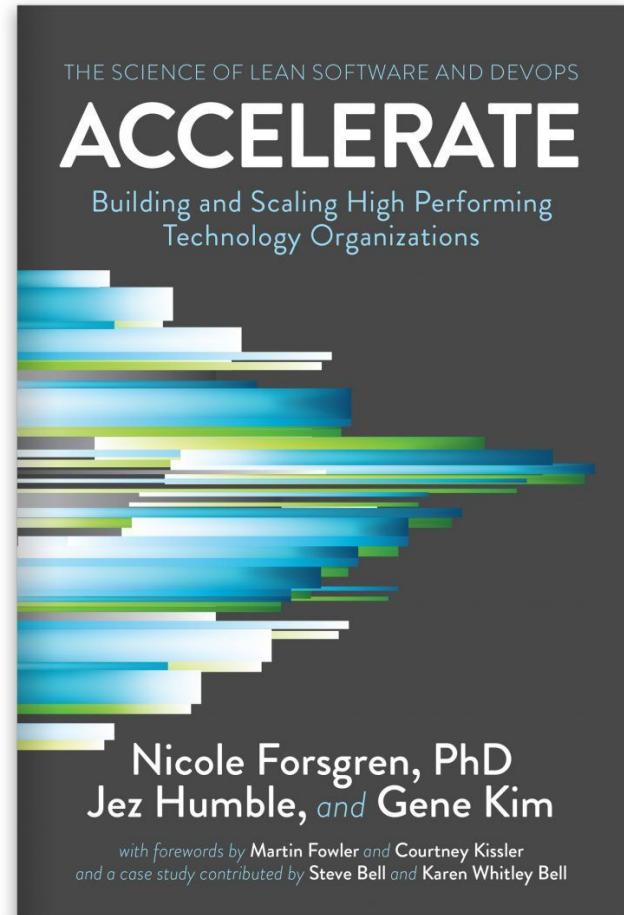


Nicole Forsgren, PhD
Jez Humble, and Gene Kim

with forewords by Martin Fowler and Courtney Kissler
and a case study contributed by Steve Bell and Karen Whitley Bell

4 Key Metrics

1. Lead time
2. Deployment frequency
3. Mean time to restore (MTTR)
4. Change fail percentage



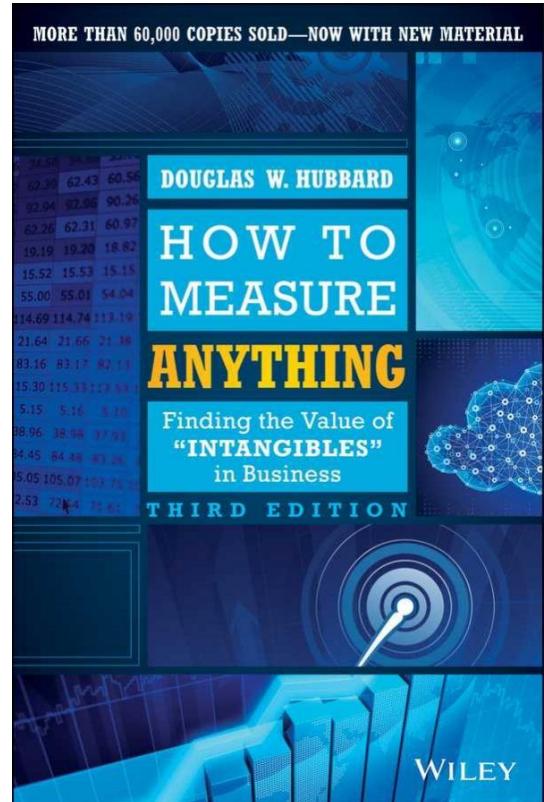
MORE THAN 60,000 COPIES SOLD—NOW WITH NEW MATERIAL

DOUGLAS W. HUBBARD
**HOW TO
MEASURE
ANYTHING**
Finding the Value of
“INTANGIBLES”
in Business

THIRD EDITION

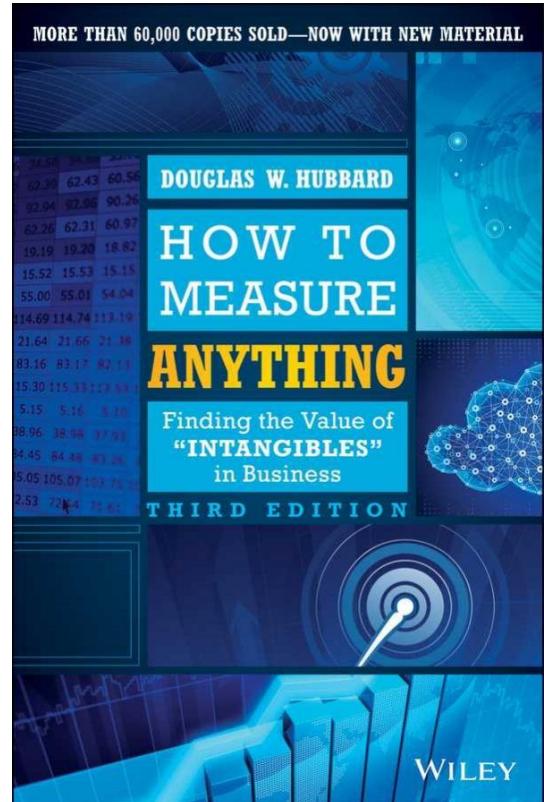


Information Value of Variables



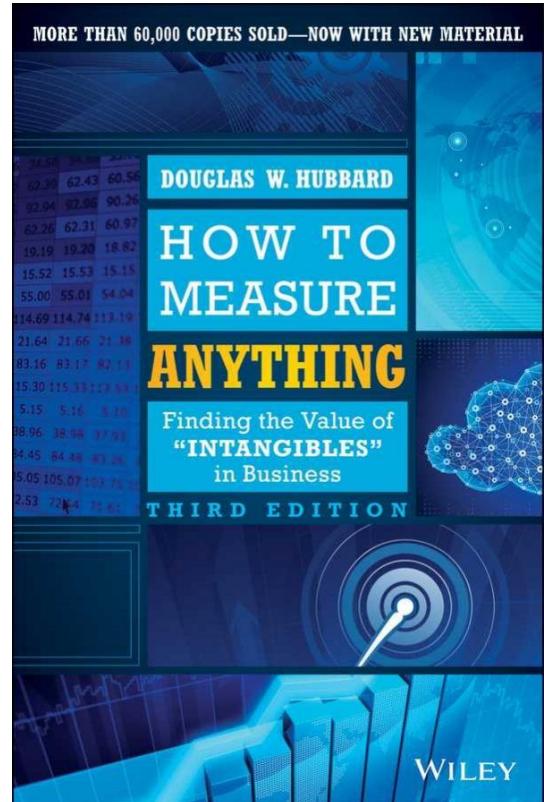
Information Value of Variables

1. Project cancelled?



Information Value of Variables

1. Project cancelled?
2. Are people using it?



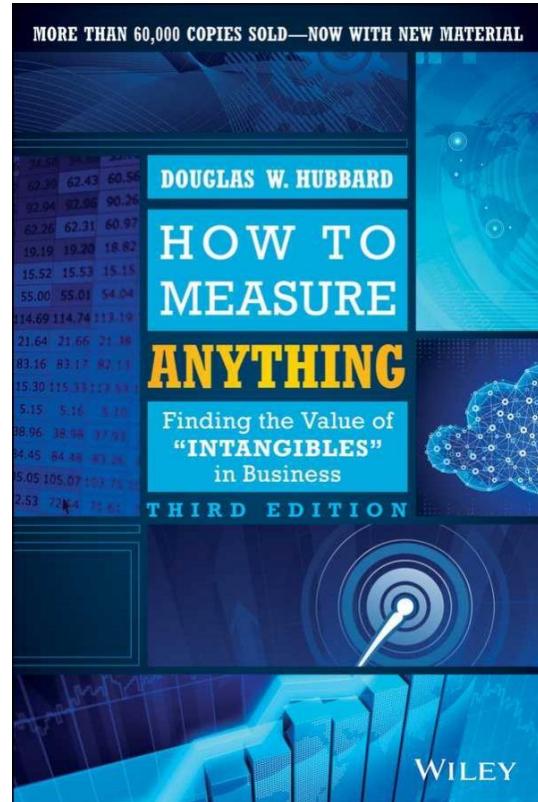
Information Value of Variables

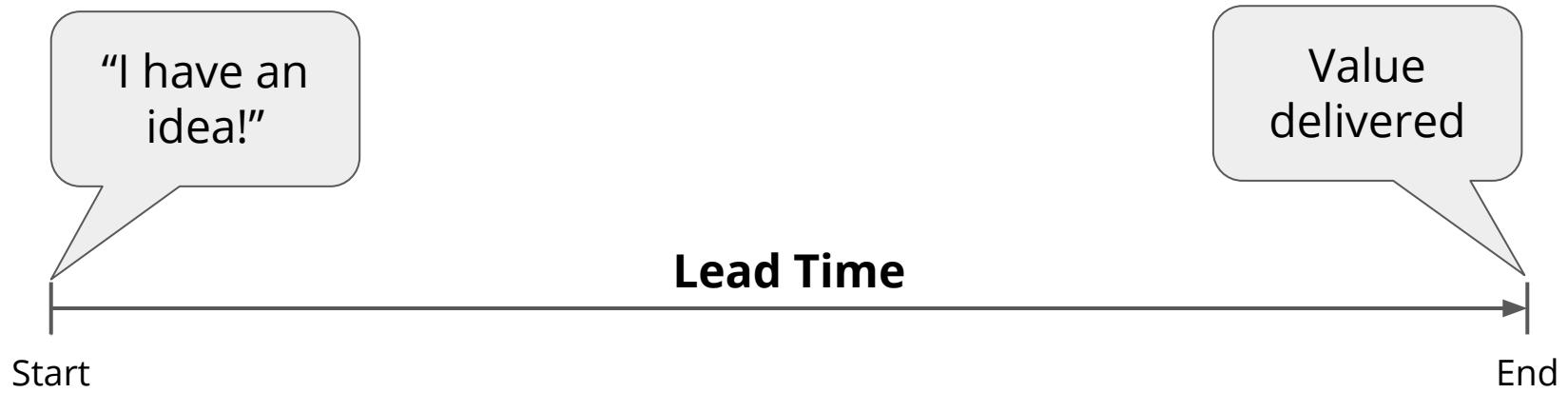
1. Project cancelled?

2. Are people using it?

...

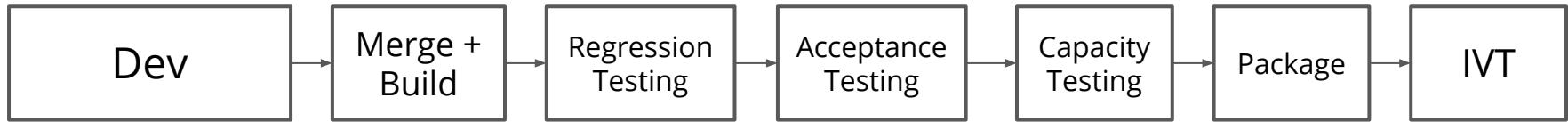
n. Cost

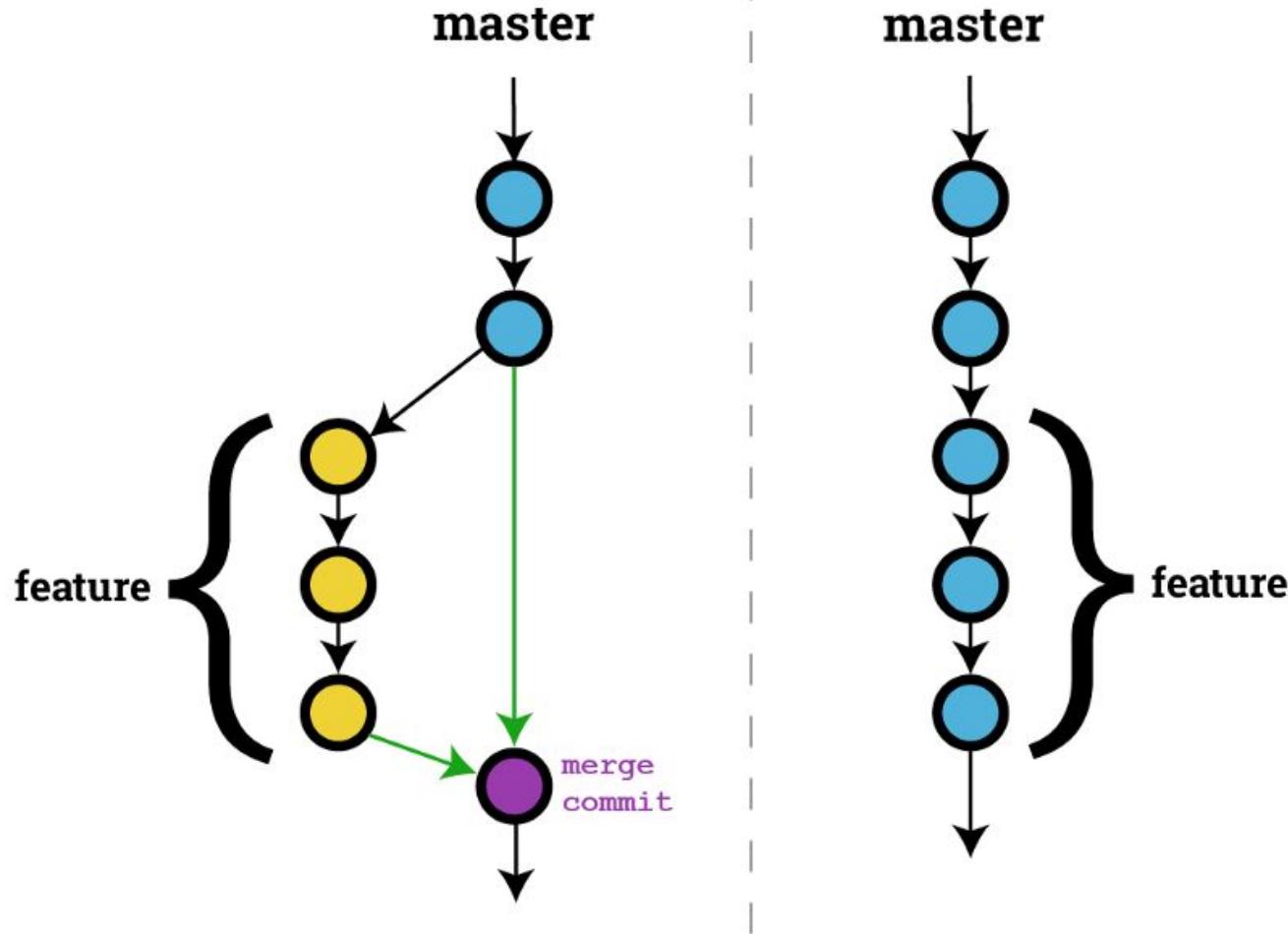


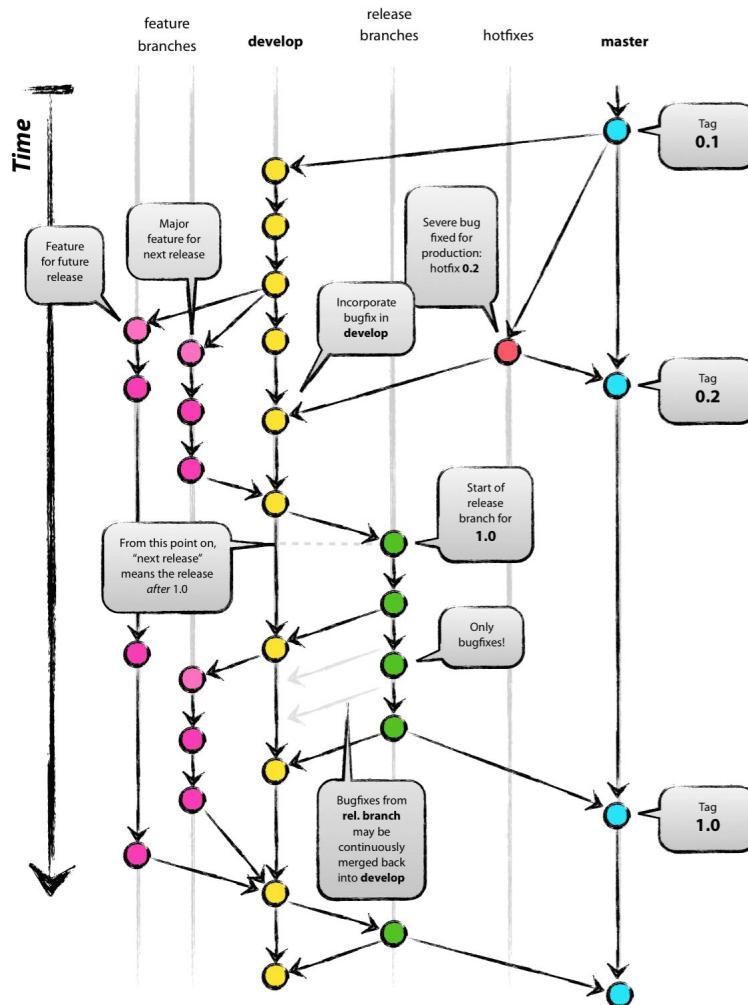














6 changes, 2 conflicts

Local Changes (Read-only)

```
public int getMin() {
    int min = data[0];
    for (int value : data) {
        if (value < min) {
            min = value;
        }
    }
    return min;
}

public int getAverage() {
    if (data.length == 0) {
        return 0;
    }

    int sum = 0;
    int i = 0;
    while (i < data.length) {
        sum += data[i];
        i++;
    }
    return sum / data.length;
}

public void printData() {
    // Print out the data
    for (int value : data) {
        System.out.print(value + " ");
    }
    System.out.println();
}

// Print out the stats:
int avg = getAverage();
int min = getMin();
```

LF

Result

```
* NumberFun class manages some data and generates
* My Changes and Teammate's changes applied!
*/
public class NumberFun {
    private int[] data;
    public NumberFun(int[] data) {
        this.data = data;
    }
    public int getAverage() {
        int sum = 0;
        int i = 0;
        while (i < data.length) {
            sum += data[i];
            i++;
        }
        return sum / data.length;
    }
    public void printData() {
        // Print out the data
        int i = 0;
        while (i < data.length) {
            int value = data[i];
            System.out.print(value + ", ");
            i++;
        }
        System.out.println();
    }
    // Print out the stats:
    int avg = getAverage();
    System.out.println("Stats: avg = " + avg);
}
```

CRLF

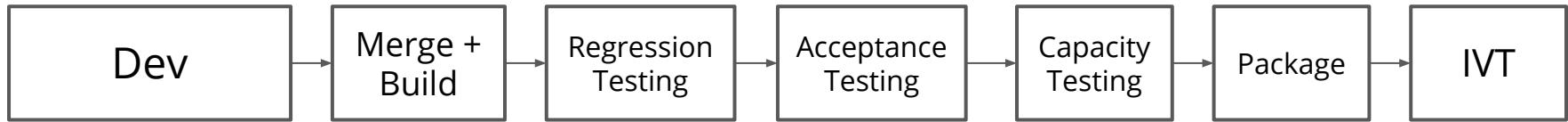
```
* NumberFun class manages some data and generates
* Teammate's changes applied!
*/
public class NumberFun {
    private int[] data;
    public NumberFun(int[] data) {
        this.data = data;
    }
    public int getAverage() {
        int sum = 0;
        for (int i = 0; i < data.length; i++) {
            sum += data[i];
        }
        return sum / data.length;
    }
    public int getMax() {
        int max = data[0];
        for (int value : data) {
            if (value > max) {
                max = value;
            }
        }
        return max;
    }
    public void printData() {
        // Print out the data
        for (int i = 0; i < data.length; i++) {
            int value = data[i];
            System.out.print(value + ", ");
        }
        System.out.println();
    }
}
```

Accept Left

Accept Right

Apply

Abort



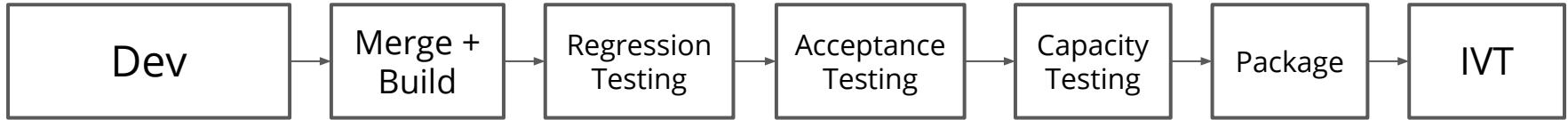
Problems

Large deltas between changes (merge hell)

New builds may break previous features

Long feedback cycles

Only testable/shippable at the end



Value Adding:

Time spent adding value for the user

Non Value Adding:

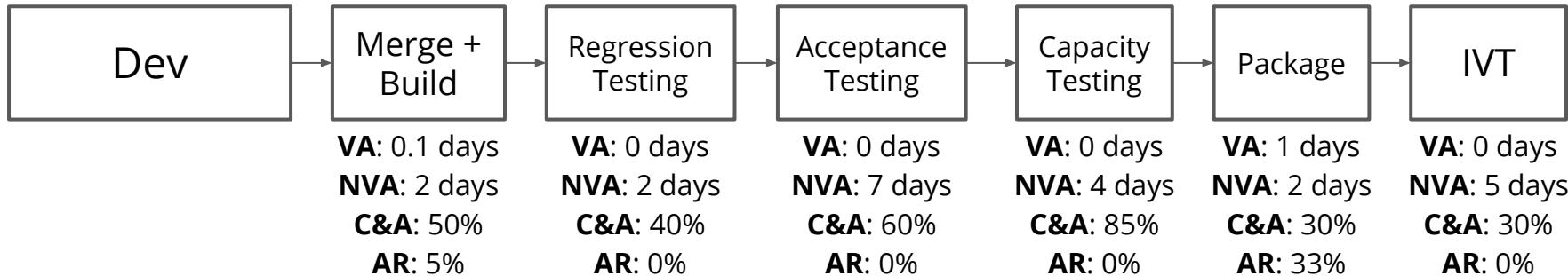
Time spent **not** adding value for the user

Complete & Accurate:

Percentage of times the process completes without problems

Activity Ratio:

Ratio of Value Adding and Non Value Adding



Value Adding:

Time spent adding value for the user

Non Value Adding:

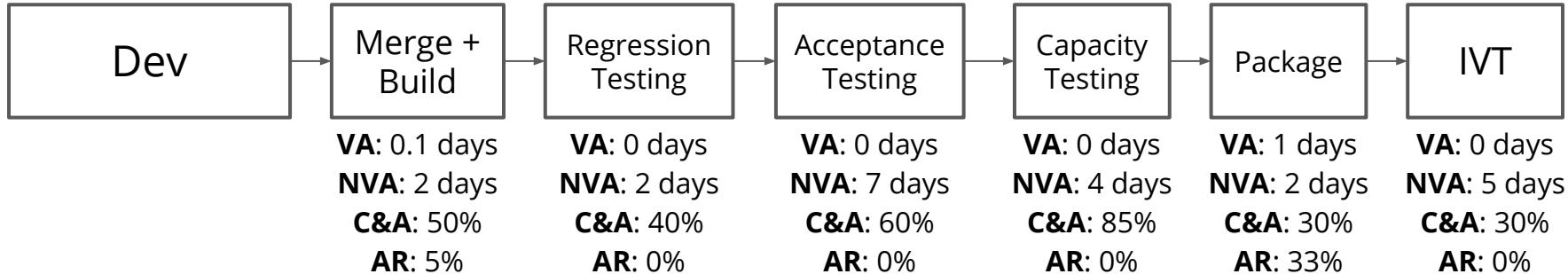
Time spent **not** adding value for the user

Complete & Accurate:

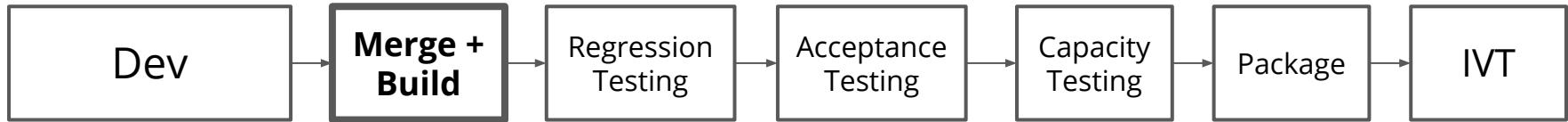
Percentage of times the process completes without problems

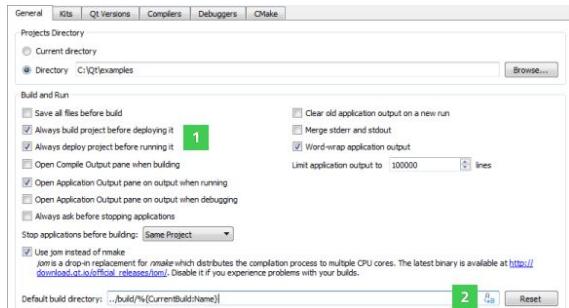
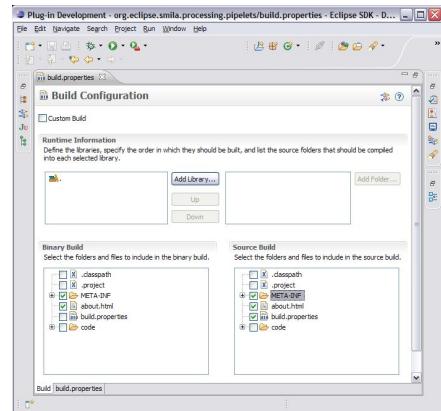
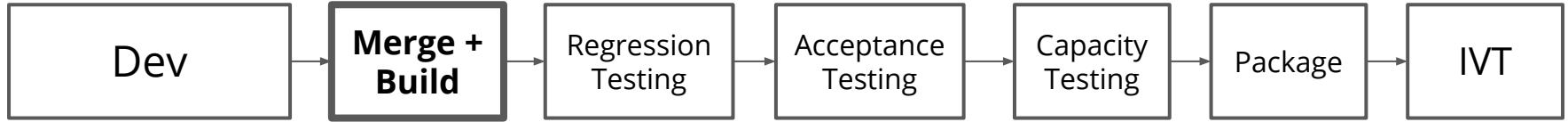
Activity Ratio:

Ratio of Value Adding and Non Value Adding



Value Adding: 1.1 days
Non Value Adding: 22 days
Complete & Accurate: 9%
Activity Ratio: 5%





```

> Task :logging:compileJava          Grouped Logs
Note: /Users/eric/src/gradle/gradle/subprojects/logging/
src/main/java/org/gradle/internal/logging/progress/ProgressLogger.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Note: Some input files use unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
===== 48% EXECUTING [6s] Status Bar
> :toolingApi:compileJava
> :logging:compileTestFixturesGroovy
> :dependencyManagement:compileJava
> :reporting:classpathManifest      Work In-Progress

```

Configuring Build Options

This document details how to configure Firefox builds. Most of the time a `mozconfig` file is not required. The default options are the most well-supported, so it is preferable to add as few options as possible. Please read the following directions carefully before building, and follow them in order. Skipping any step may cause the build to fail, or the built software to be unusable. Build options, including options not usable from the command-line, may appear in `configvars.sh` files in the source tree.

Using a `mozconfig` configuration file

The choice of which Mozilla application to build and other configuration options can be configured in a `mozconfig` file. (It is possible to manually call `configure` with command-line options, but this is not recommended). The `mozconfig` file should be in your source directory (that is, `/mozilla-central/mozconfig` or `/comm-central/mozconfig`).

Create a blank `mozconfig` file:

```
1 | echo "# My first mozilla config" > mozconfig
```

If your `mozconfig` isn't in your source directory, you can also use the `MOZCONFIG` environment variable to specify the path to your `mozconfig`. The path you specify must be an absolute path or else `client.mk` will not find it. This is useful if you choose to have multiple `mozconfig` files for different applications or configurations (see below for a full example). Note that in the `export` example below the filename was not `mozconfig`. Regardless of the name of the actual file you use, we refer to this file as the `mozconfig` file in the examples below.

Setting the `mozconfig` path:

```
1 | export MOZCONFIG=~/mozilla/mozconfig-firefox
```

Calling the file `.mozconfig` (with a leading dot) is also supported, but this is not recommended because it may make the file harder to find. This will also help when troubleshooting because people will want to know which build options you have selected and will assume that you have put them in your `mozconfig` file.

`mozconfig` contains two types of options

- Options prefixed with `mk_add_options` are passed to `client.mk`. The most important of these is `MOZ_OBJDIR`, which controls where your application gets built (also known as the object directory).
- Options prefixed with `ac_add_options` are passed to `configure`, and affect the build process.

Building with an objdir

This means that the source code and object files are not intermingled in your directory system

Build Instructions

Ben Fry edited this page on 11 Jun - 79 revisions

If you have questions about the contents of this document, ask them in the [forum](#).

The quick version

Let's try the easy route first:

- Install Java 8
 - On Windows and Linux, install the latest [JRE 8 JRE 8u202](#) from Oracle.
 - On macOS, download and install [JDK 8 JDK 8u202](#).
 - Other versions of Java are not supported. It must be 8u202. Later versions of Java (9, 10, 13, whatever) don't work either.
 - OpenJDK is also not supported. Really. Please don't use it (and then complain when things break).
- Install Apache Ant
 - On Windows and Linux, use the [installation instructions](#)
 - Or on Ubuntu 16.04, it's just a matter of `sudo apt-get install ant`
 - On macOS, it's much easier to via [Homebrew](#) or MacPorts.
 - Ant 1.8 or later is required.
- Clone the [source repository](#) from GitHub.
 - On the command line, enter:
`git clone https://github.com/processing/processing.git`
 - You can probably use GitHub for Windows or GitHub for Mac instead of the command line, however these aren't tested/supporteded and we only use the command line for development. Use [this link](#) to download the command line version.
- Open a terminal/console/command prompt, change to the directory where you cloned Processing, and type:
`cd build
ant run`

With any luck, there will be all sorts of console spew for a few moments, followed by a Processing window showing up. Hooray!

To get the latest updates, just run `git pull` from the command line (or GUI client) and do an `ant run` from inside the `build` folder as shown above.

That didn't work

Oh well, here are more details for what may have gone wrong:

- No internet connection? – When building for the first time on Windows and Linux, you'll need to have an internet connection, because additional files need to be downloaded.
- Remove the `work` folder – If it was working in the past, try removing the `work` folder. On OS X, that's `build/macosx/work`, on Windows, it's `build/windows/work`, and so on. When larger changes occur, sometimes an old work directory can cause problems.

Deployment Pipeline

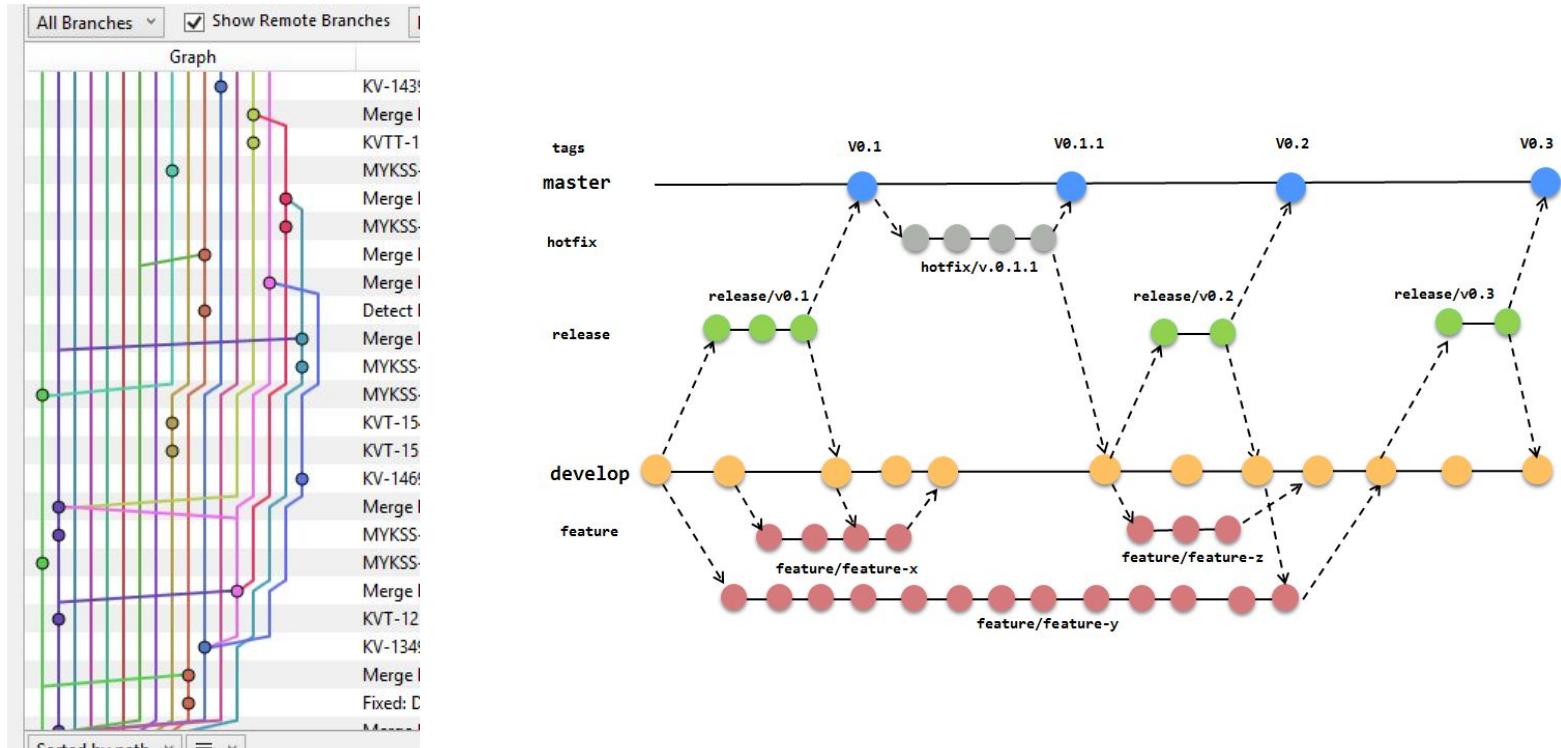
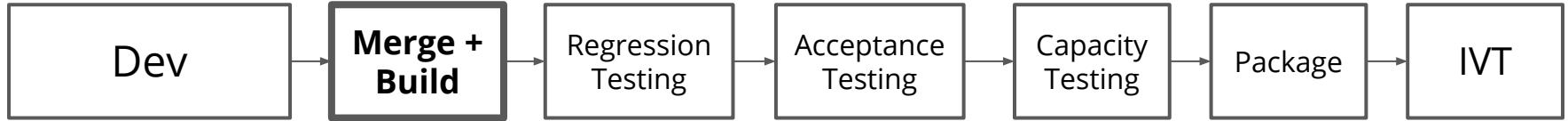
- Fully automated build (no human interaction required)
- Tests each build
- Runs on dedicated infrastructure (not local machine)
- Automatically triggered by commits

“...a deployment pipeline is an **automated manifestation** of your **process** for getting software from version control into the hands of your users.”

Jez Humble, Continuous Delivery

"The way the deployment pipeline works, is as follows. **Every change** that is made to an application's **configuration, source code, environment, or data, triggers the creation of a new instance of the pipeline.** One of the first steps in the pipeline is to create binaries and installers. The rest of the pipeline runs a series of tests on the binaries to prove that they can be released. **Each test that the release candidate passes gives us more confidence** that this particular combination of binary code, configuration information, environment, and data will work. **If the release candidate passes all the tests, it can be released."**

Jez Humble, Continuous Delivery



"Continuous Integration (CI) is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early. By **integrating regularly**, you can **detect errors quickly**, and **locate them more easily.**"

Jez Humble, Continuous Delivery

“Un[released/merged] code makes me nervous”

Clifton Cunningham, CTO @ TES Global



Jenkins



circleci



Task - Stage View

	Checkout Source	Build war	Unit Tests	Code Analysis	Publish to Nexus	Build and Tag OpenShift Image	Deploy to Dev	Integration Tests	Copy Image to Nexus Docker Registry	Blue/Green Production Deployment	Switch over to new Version
Average stage times: (Average full run time: ~29min)											
9s)	19s	1min 9s	1min 1s	2min 10s	1min 55s	22s	19s	16s	41s	23s	800ms
#8 Apr 20 09:20 No Changes	22s	3min 36s	55s	1min 37s	40s	18s	23s	16s	37s	23s	800ms (paused for 19min 11s)
#7 Apr 19 19:35 No Changes	15s	1min 0s	50s	1min 45s	1min 30s	14s	23s	16s	2min 7s failed		

	Validation	Checkout	Build	Unit test	Static analysis	Collect code coverage reports	Post-build actions
Average stage times: (Average full run time: ~3min 20s)							
20s)	97ms	7s	1min 10s	38s	39s	438ms	269ms
#6 May 29 13:22 99 commits	119ms	8s	1min 48s	38s	39s	438ms	269ms
#5 May 29 13:09 1 commit	76ms	5s	35s failed				
#4 May 29 12:28 16 commits	96ms	8s	1min 7s failed				



Run tests #2 started 10m 21s ago finished 9m 6s ago duration 1m 15s

2 1

```

Status: Downloaded newer image for node@sha256:2a3fe9a32d0de074bb2dbd46273ec452f3610cd055a2ea8bb94b020c9ff61f20.

Successfully pulled node@sha256:2a3fe9a32d0de074bb2dbd46273ec452f3610cd055a2ea8bb94b020c9ff61f20.

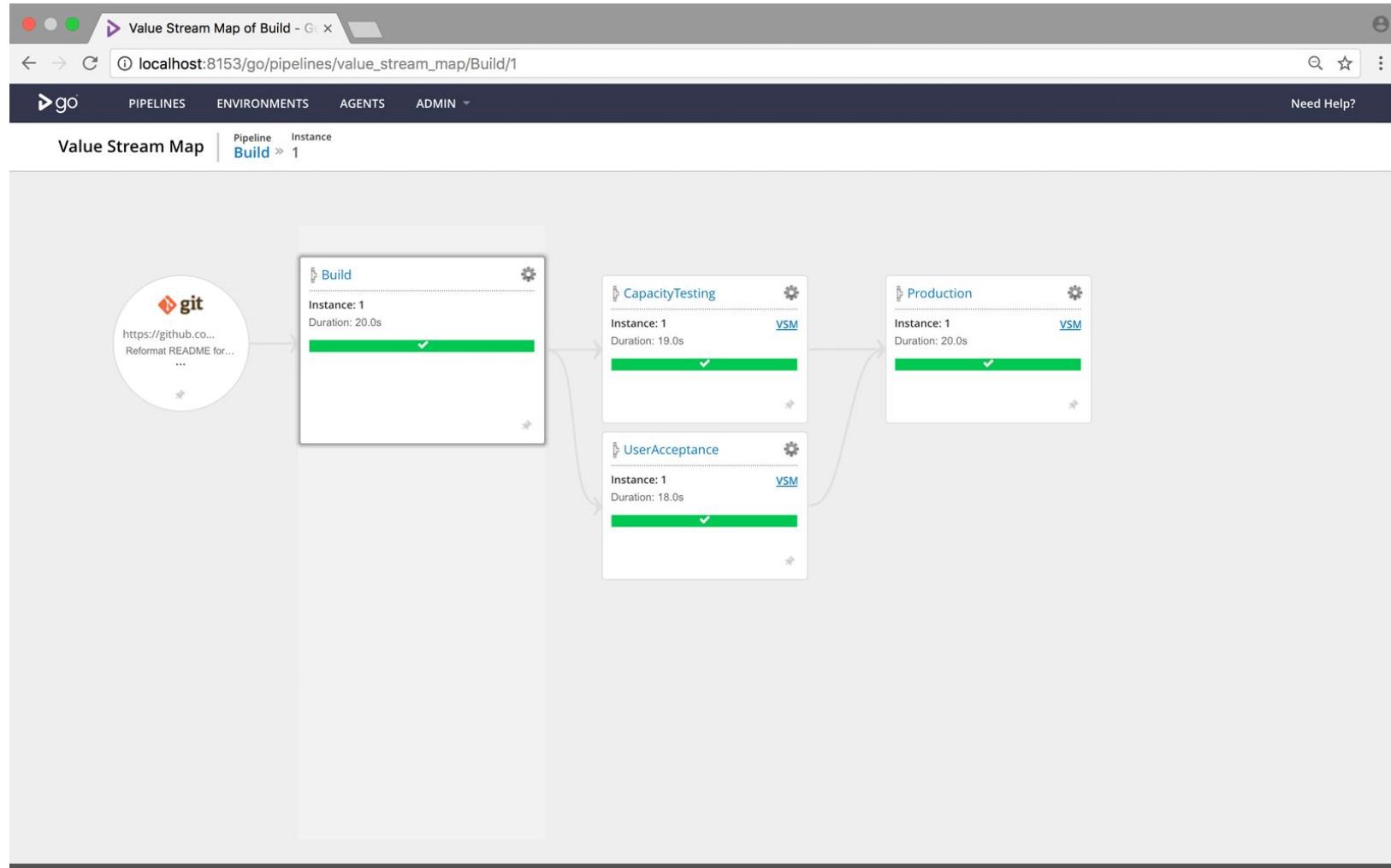
+ mv dependency-cache/node_modules hello_hapi
+ cd hello_hapi
+ npm test
npm info it worked if it ends with ok
npm info using npm@5.0.3
npm info using node@v8.1.3
25npm info lifecycle hello@1.0.0-pretest: hello@1.0.0
25npm info lifecycle hello@1.0.0-test: hello@1.0.0

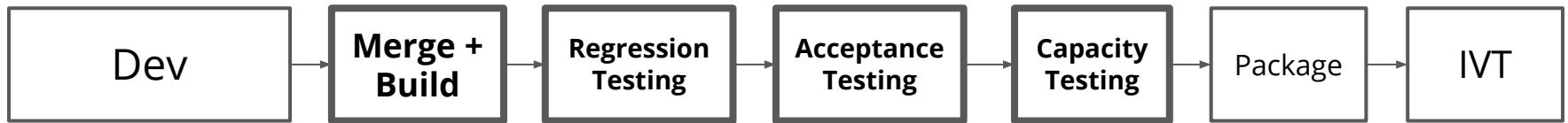
> hello@1.0.0 test /tmp/build/e17d3cfcc/hello_hapi
> NODE_ENV=test node node_modules/lab/bin/lab -v -L -C -D

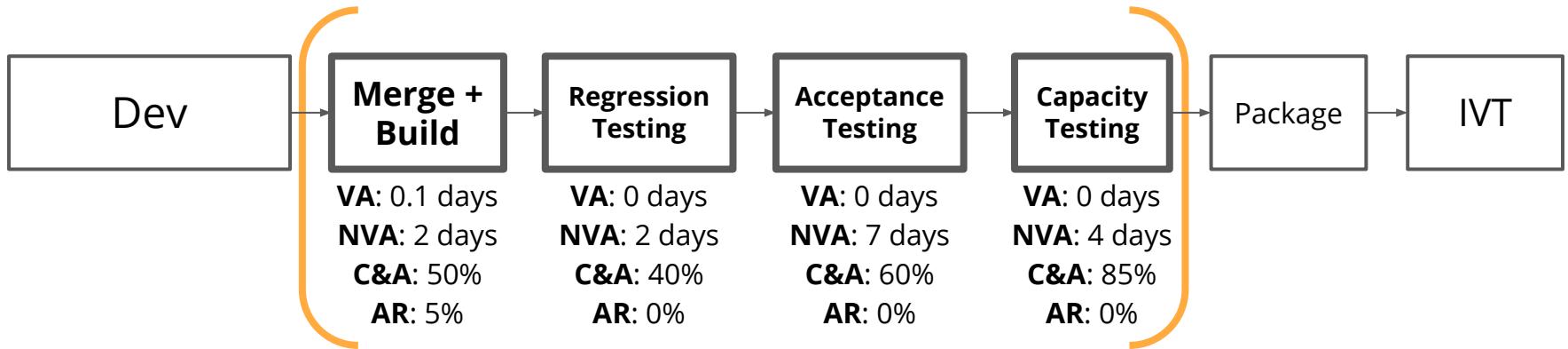
Basic HTTP Tests
  ✓ 1) Greets /hello/homer} (120 ms)
  ✓ 2) Greets /hello/Homer%20Simpson} (16 ms)
test greetings
  ✓ 3) greets with name (2 ms)

3 tests complete
Test duration: 181 ms
No global variable leaks detected
Linting results: No issues

npm info lifecycle hello@1.0.0-posttest: hello@1.0.0
npm info ok
  
```

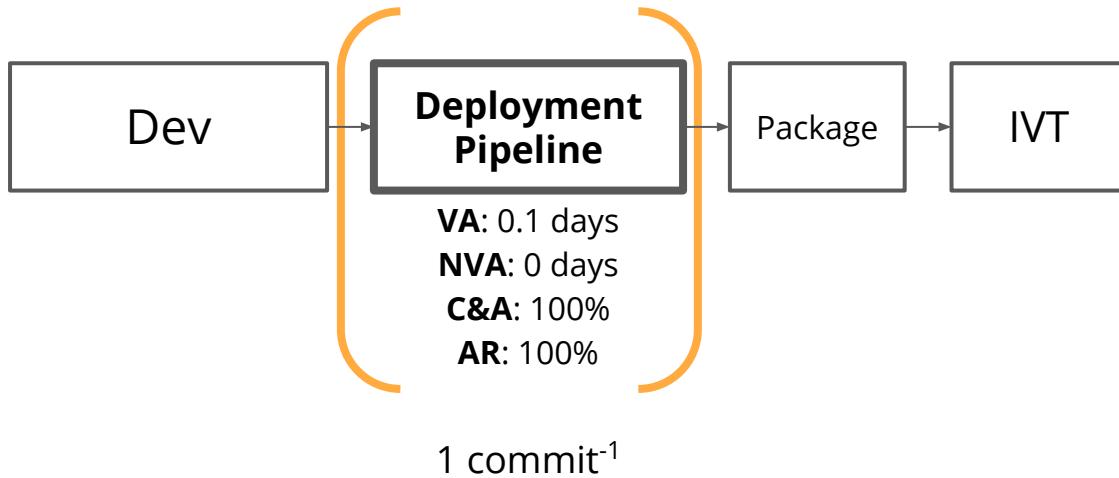






1 month⁻¹

VA: 0.1 days
NVA: 15 days
C&A: 10%
AR: 0.6%



1 commit⁻¹

VA: 0.1 days
NVA: 0 days
C&A: 100%
AR: 100%

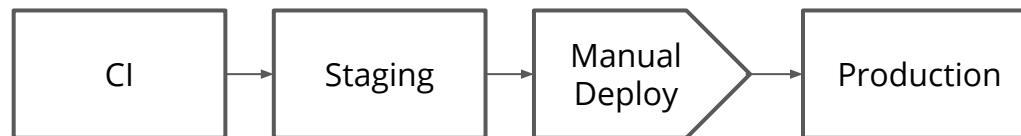
Continuous Integration

- No long lived branches! (Trunk Based Development)
- Integrate regularly (at least daily)
- Every change triggers feedback (Deployment Pipeline)
- Small deltas between changes (easy to find problems)
- Every commit should be merged to master*
- More than a process/tool!

Continuous Delivery

Continuous Delivery

- Always be able to put a product into production (PSP)
- For applications: something that can be shipped to a user
- For services: deployed to a pre-production environment
- Click to deploy to production



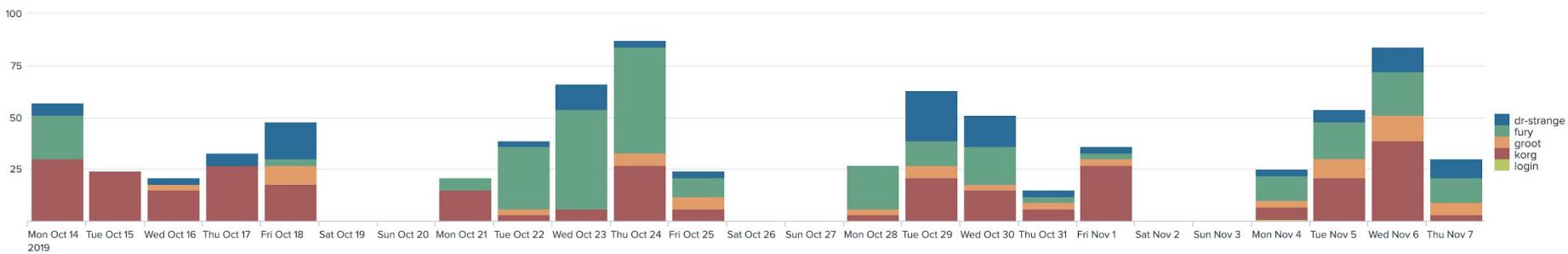
“Our highest priority is to satisfy the customer through early and continuous delivery of valuable software”

agilemanifesto.org

Continuous Deployment

“If you have more meetings than releases,
you’re an enterprise company”

Clifton Cunningham, CTO @ TES Global



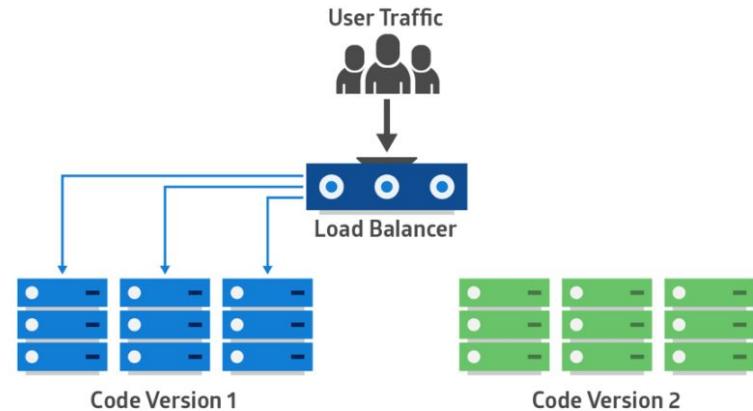
Continuous Deployment

- Every change goes to production! (if it passes the tests)
- Requires more confidence in pipeline
- May require tooling and organisational changes



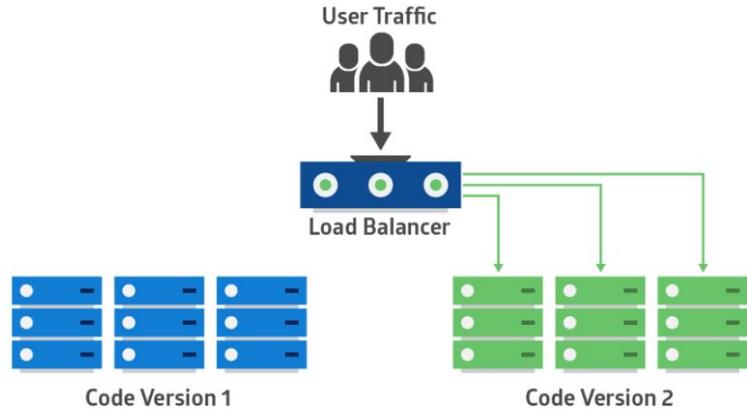
Blue Green Deployment

- Don't change, replace
- Zero-downtime deployment
- Easy rollback
- Immutable infrastructure



Blue Green Deployment

- Don't change, replace
- Zero-downtime deployment
- Easy rollback
- Immutable infrastructure



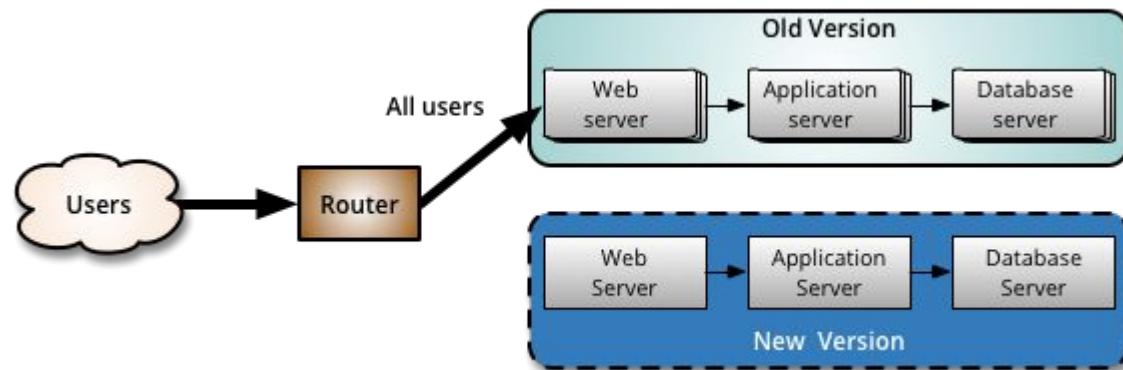
Blue Green Deployment

- Don't change, replace
- Zero-downtime deployment
- Easy rollback
- Immutable infrastructure



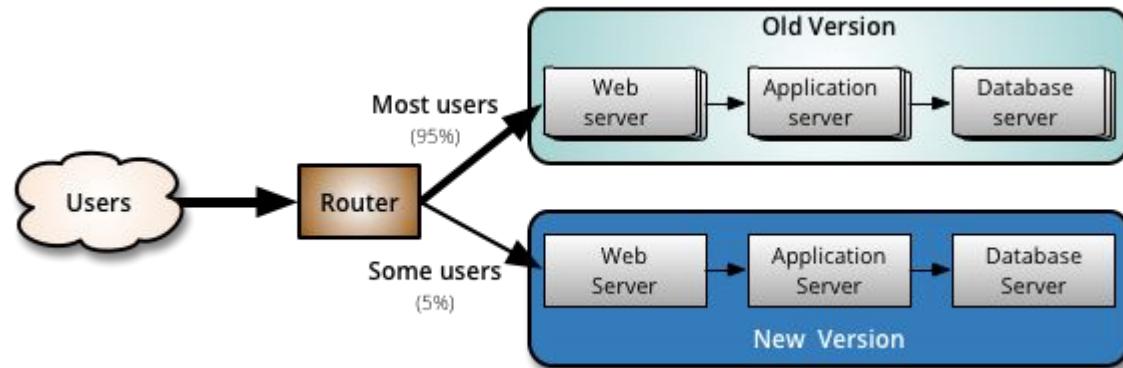
Canary Release

- Release to a small cohort first
- Useful for A/B testing
- Limits blast radius



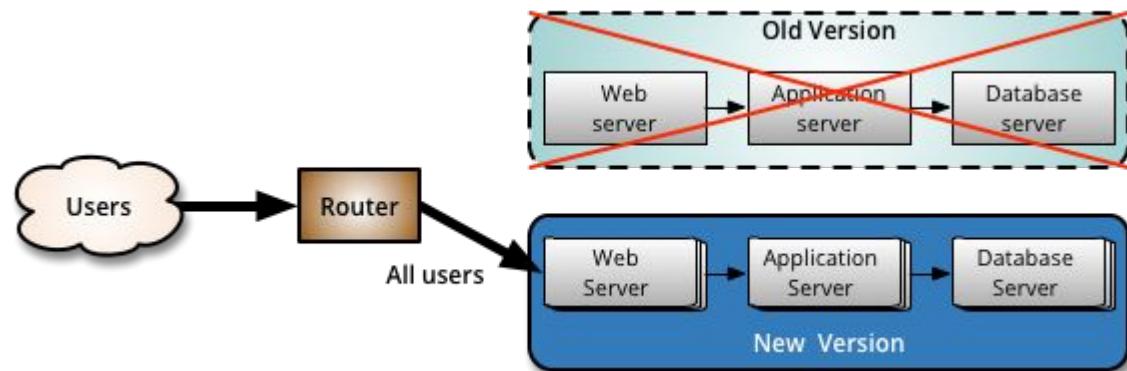
Canary Release

- Release to a small cohort first
- Useful for A/B testing
- Limits blast radius



Canary Release

- Release to a small cohort first
- Useful for A/B testing
- Limits blast radius



Feature Flag

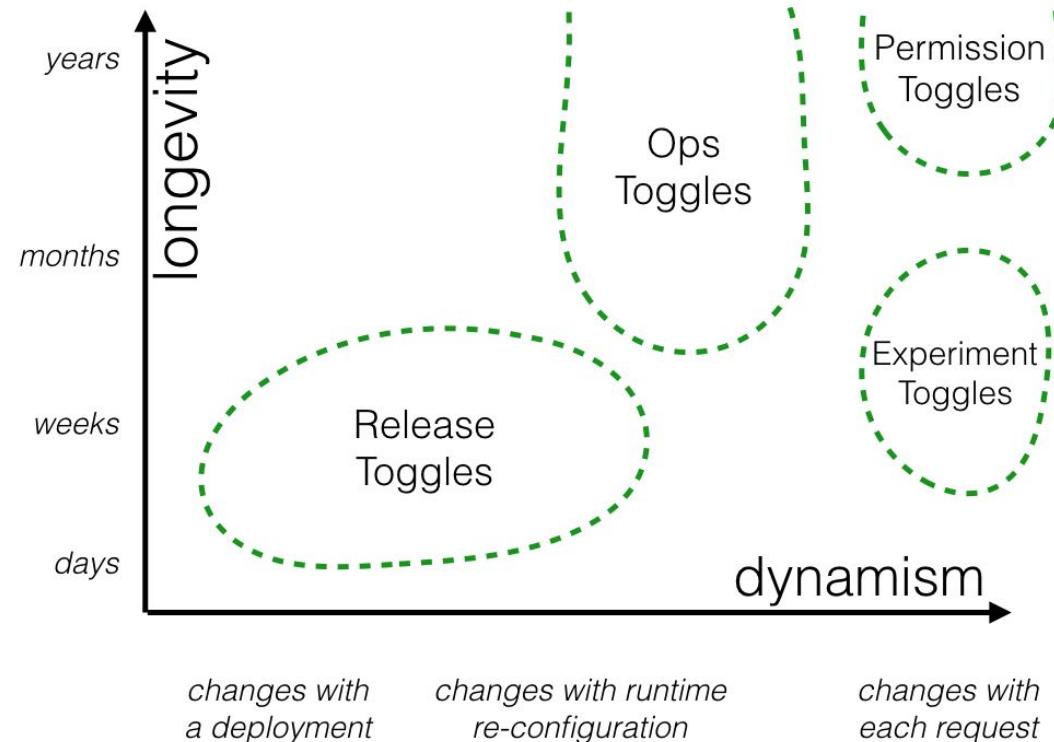
- Deploy continuously
- Deploy != Release
- Release by toggle
- Extra code complexity

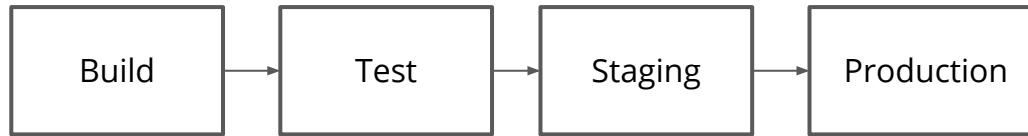
The screenshot shows a dashboard interface for managing feature flags. On the left is a sidebar with navigation links: Production (selected), Feature flags (highlighted), Users, Segments, Goals, Debugger, Audit log, Quickstart, Documentation, and Intergrations. The main area is titled "Dashboard" and contains a section for "Your feature flags". A search bar at the top right says "Find flags by name, key or description". A green button labeled "NEW FLAG" is also present. Below the search bar is a table listing five feature flags:

Flag	Last Added	Last Requested	Status
Site Maintenance Mode	2 days ago	a few seconds ago	ON
Enable Debugger	19 days ago	17 minutes ago	OFF
New UI	7 months ago	3 hours ago	ON
RabbitMQ Kill Switch	9 months ago	1 day ago	ON
New Recommendations Engine	12 months ago	2 days ago	OFF

Feature Flag

- Deploy continuously
- Deploy != Release
- Release by toggle
- Extra code complexity
- Long lived vs short lived





Continuous **Integration**

Continuous **Delivery**

Continuous **Deployment**

DevOps

DevOps

From Wikipedia, the free encyclopedia



Some of this article's [listed sources](#) **may not be reliable**. Please help this article by looking for better, more reliable sources. Unreliable citations may be challenged or deleted. (December 2018)
(Learn how and when to remove this template message)

DevOps (a clipped compound of "development" and "operations") is a set of software development practices^[failed verification] that combines **software development** (*Dev*) with information technology operations (*Ops*) to shorten the **systems development life cycle** while delivering features, fixes, and updates frequently in close alignment with business objectives.^[1]

Software development

Core activities

Processes · Requirements · Design
Engineering · Construction · Testing
Debugging · Deployment · Maintenance

Paradigms and models

Definition [edit]

Academics and practitioners have not developed a unique definition for the term "DevOps."^{[a][b][c][d]}



Cindy Sridharan @copyconstruct · Apr 13, 2017



I ask one of my own meetup attendees - what do you do?

His answer ---

DevOps

Happens every single time.

DevOps is not a profession. 😩



Cindy Sridharan

@copyconstruct

Almost want to yell - Stop saying DevOps - it doesn't mean what you think it means.



Cindy Sridharan

@copyconstruct



Microservices == SOA + DevOps

Where devops means "automation & infrastructure"

every single day I hear a different definition of "devops" 😩



There is No Such Thing as a "Devops Team"



Jez Humble

Principal Consultant

[Continuous Delivery »](#)

[Technology »](#)

“10+ Deploys Per Day: Dev and Ops Cooperation at Flickr”

John Allspaw, Velocity 2009

“...a cross-disciplinary community of practice dedicated to the study of building, evolving and operating rapidly-changing resilient systems at scale.”

Jez Humble

Principles of Software Delivery

- Create a Repeatable, Reliable Process for Releasing Software
- Keep Everything in Version Control
- Continuous Improvement
- Automate Almost Everything
- Build Quality In
- Done Means Released
- If It Hurts, Do It More Frequently, and Bring the Pain Forward
- Everybody Is Responsible for the Delivery Process

Automate Almost Everything

HOW LONG CAN YOU WORK ON MAKING A ROUTINE TASK MORE
EFFICIENT BEFORE YOU'RE SPENDING MORE TIME THAN YOU SAVE?
(ACROSS FIVE YEARS)

		HOW OFTEN YOU DO THE TASK					
		50/DAY	5/DAY	DAILY	WEEKLY	MONTHLY	YEARLY
1 SECOND		1 DAY	2 HOURS	30 MINUTES	4 MINUTES	1 MINUTE	5 SECONDS
5 SECONDS		5 DAYS	12 HOURS	2 HOURS	21 MINUTES	5 MINUTES	25 SECONDS
30 SECONDS		4 WEEKS	3 DAYS	12 HOURS	2 HOURS	30 MINUTES	2 MINUTES
HOW MUCH TIME YOU SHAVE OFF	1 MINUTE	8 WEEKS	6 DAYS	1 DAY	4 HOURS	1 HOUR	5 MINUTES
	5 MINUTES	9 MONTHS	4 WEEKS	6 DAYS	21 HOURS	5 HOURS	25 MINUTES
	30 MINUTES		6 MONTHS	5 WEEKS	5 DAYS	1 DAY	2 HOURS
	1 HOUR		10 MONTHS	2 MONTHS	10 DAYS	2 DAYS	5 HOURS
	6 HOURS				2 MONTHS	2 WEEKS	1 DAY
	1 DAY					8 WEEKS	5 DAYS

Build Quality In
and
Done Means Released

DOD (per AC / per task)

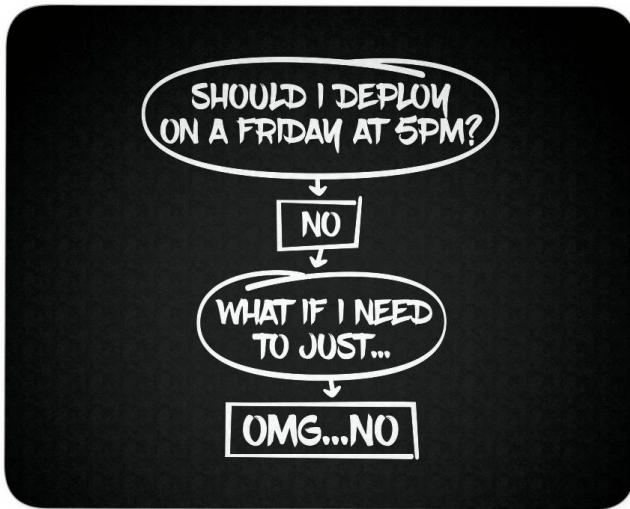
ThoughtWorks

- Test written?
- manual walkthrough (IE II, ...)
done?
- is ADR entry necessary?
- deployed to PROD?
- re-read story?
- documented? storybook?
- talked with a BA/XD during
the process?

LOAD
TEST
ADDED?

• MIGRATION
NEEDED?
• THREAT
MODEL UPDATE

If It Hurts, Do It More Frequently



DEPLOYING ON A FRIDAY?



ONE DOES NOT SIMPLY

DEPLOY ON FRIDAY

DOWN HERE WE DON'T DEPLOY ON
FRIDAY



YOU DEPLOYED ON A FRIDAY



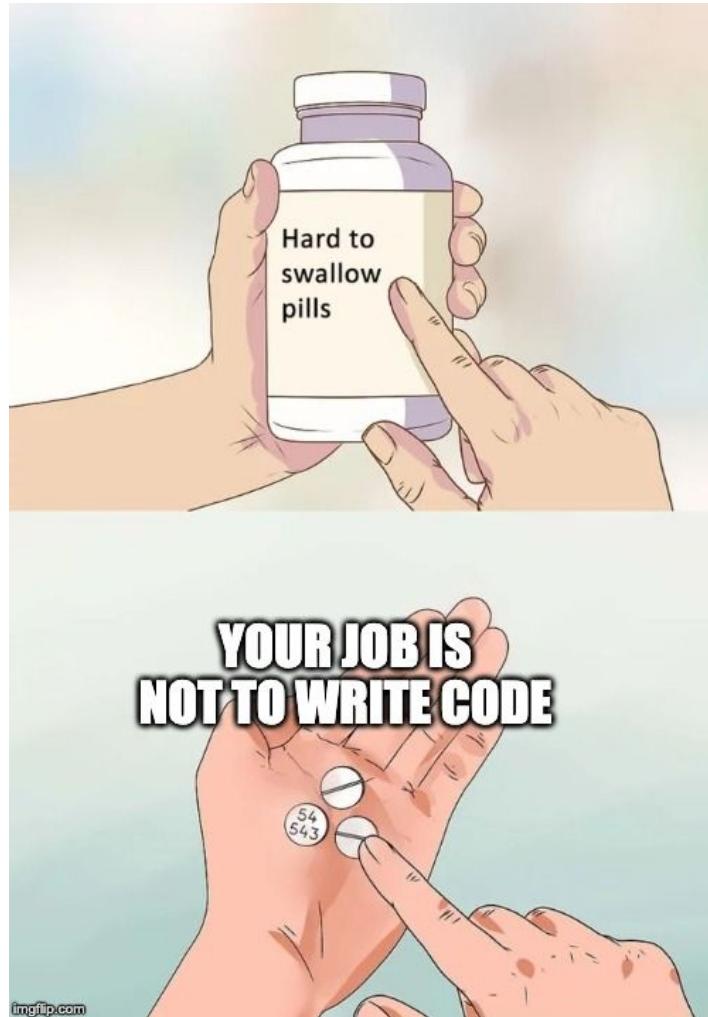
LOOK GUYS!

ITS FRIDAY, TIME TO DEPLOY!



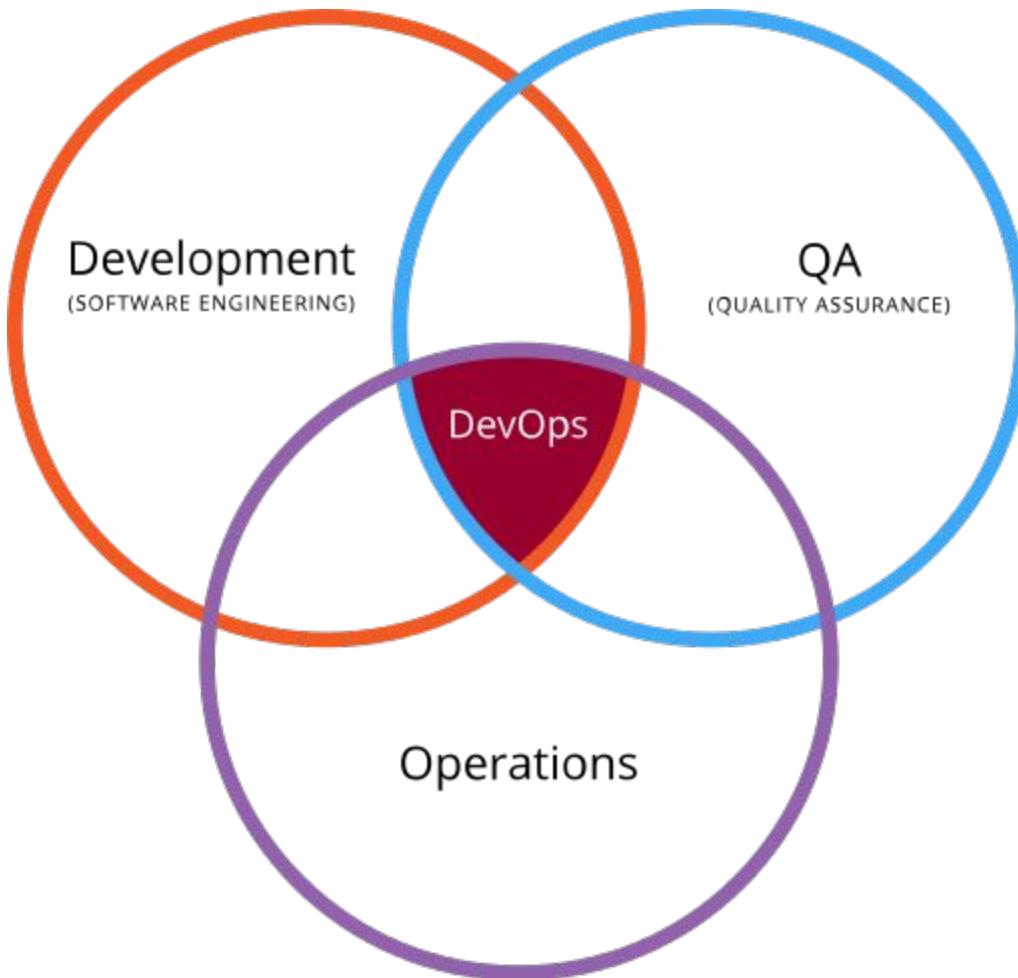
IF YOU DEPLOY MONTH OLD CHANGES ON A
FRIDAY





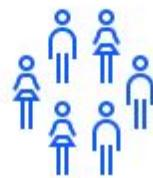
Everybody Is Responsible for the
Delivery Process



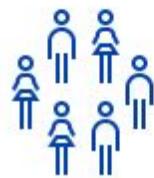


FUNCTIONAL

Common functional expertise



System Analysts



Testers



Developers

CROSS-FUNCTIONAL

Representatives from the various functions



Development Team

**WORKED FINE IN
DEV**

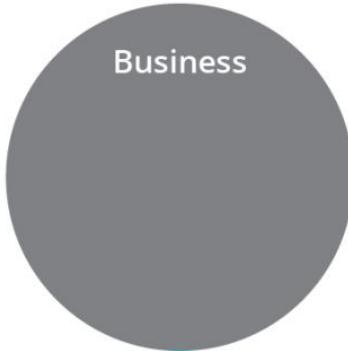
OPS PROBLEM NOW

memegenerator.net

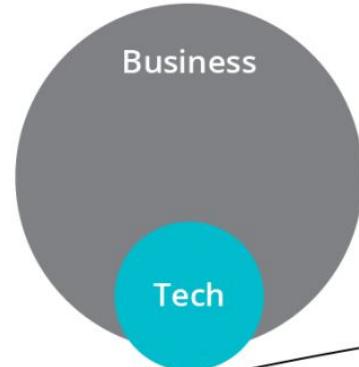
Supporting role
(1960/70)



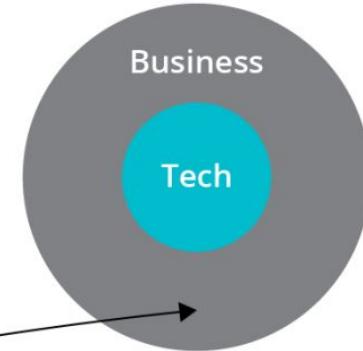
Collaboration
(1980)



Technology-driven differentiation
(1990)



Technology
is the business



Projects

On time, on budget

vs

Products

Stability

Speed

vs

Stability

Speed

Lead Time

Deployment Frequency

Stability

Failure Rate

Mean Time To Recover

VS

DevOps

DevOps for Data Science

CD in Regulated Environments

DevOps for Data Science

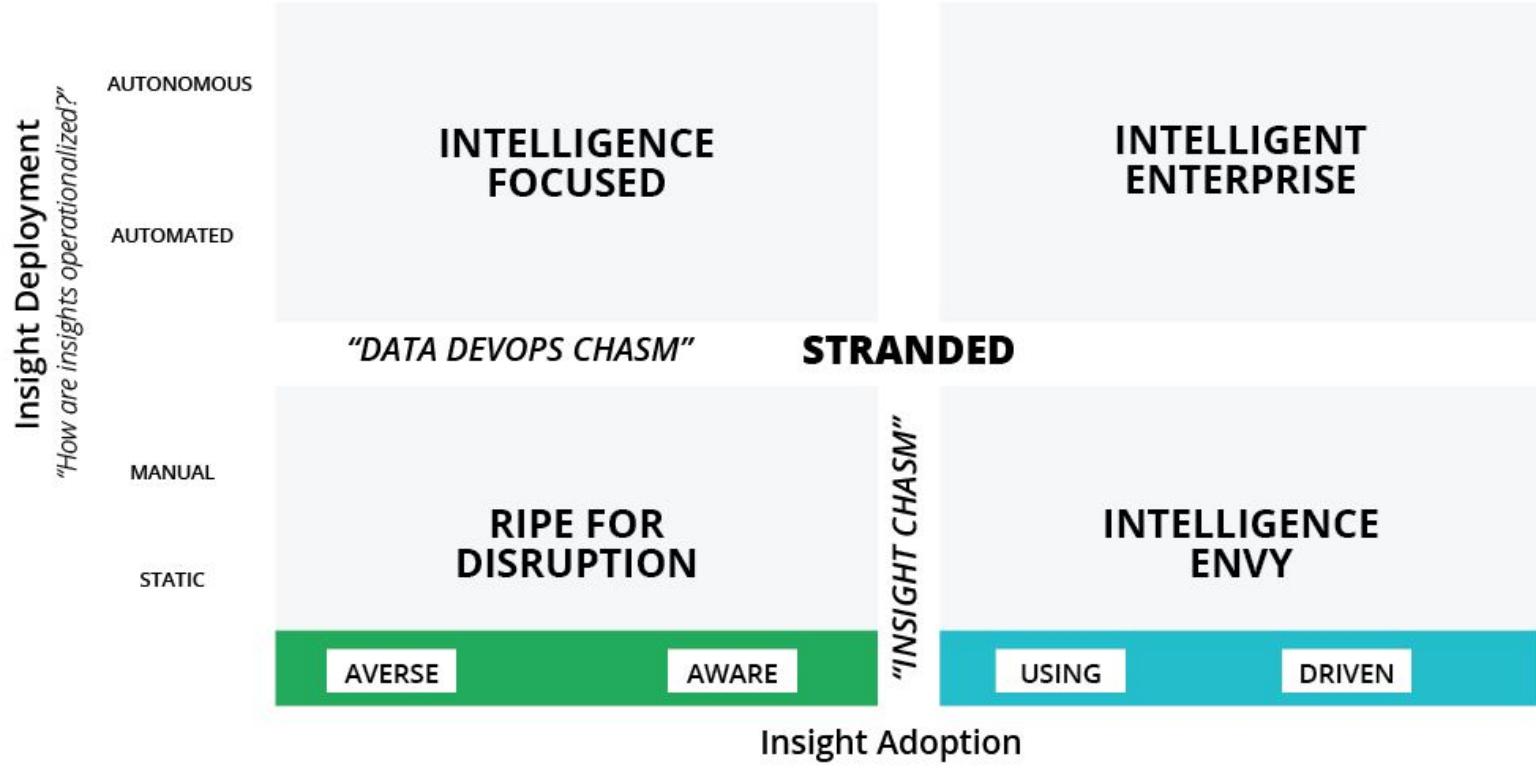
“...Every change that is made to an application’s configuration, source code, environment, or data, triggers the creation of a new instance of the pipeline...”

Jez Humble, Continuous Delivery

Triggerer	Commit	Stages
	 master → 7fba6d04  Un-fuck CSP headers.	  
	 master → 74266f82  Add CSP headers.	  

How do you define failure of a model?

- Human in the loop?
- What metrics can you use?
- What is the lag time?
- Competitive models?



Intelligent Enterprise Series

References

<https://www.thoughtworks.com/insights/articles/intelligent-enterprise-series-models-enterprise-intelligence>

<https://www.thoughtworks.com/insights/articles/intelligent-enterprise-series-cd4ml>

<https://martinfowler.com/articles/cd4ml.html>

<https://www.youtube.com/watch?v=OjhG-0RUIk>





“Concurrent Production”

“The development contracts are being performed concurrent with the production contracts”

2015 Lockheed Martin Annual Report

Release It!

Design and Deploy
Production-Ready Software



Michael T. Nygard

Ship It!

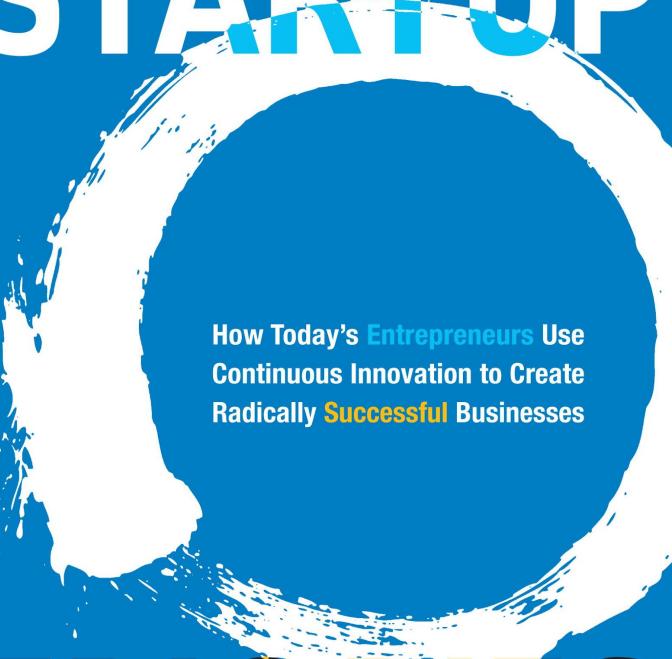
*A Practical Guide to
Successful Software Projects*



Jared Richardson William Gwaltney, Jr.

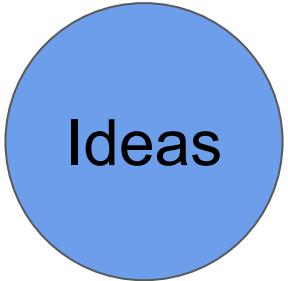
THE NEW YORK TIMES BESTSELLER

THE LEAN STARTUP

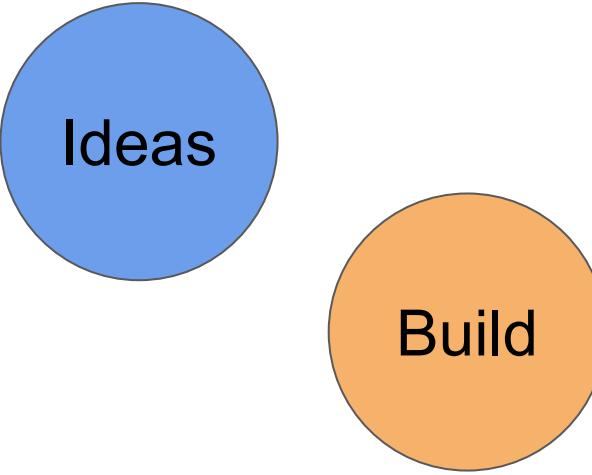


How Today's Entrepreneurs Use
Continuous Innovation to Create
Radically **Successful** Businesses

ERIC RIES

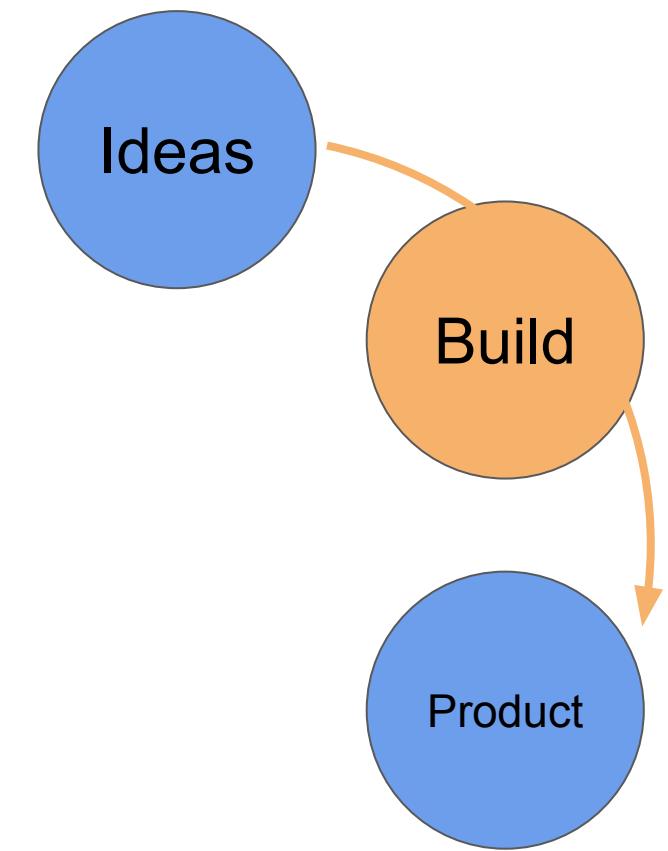


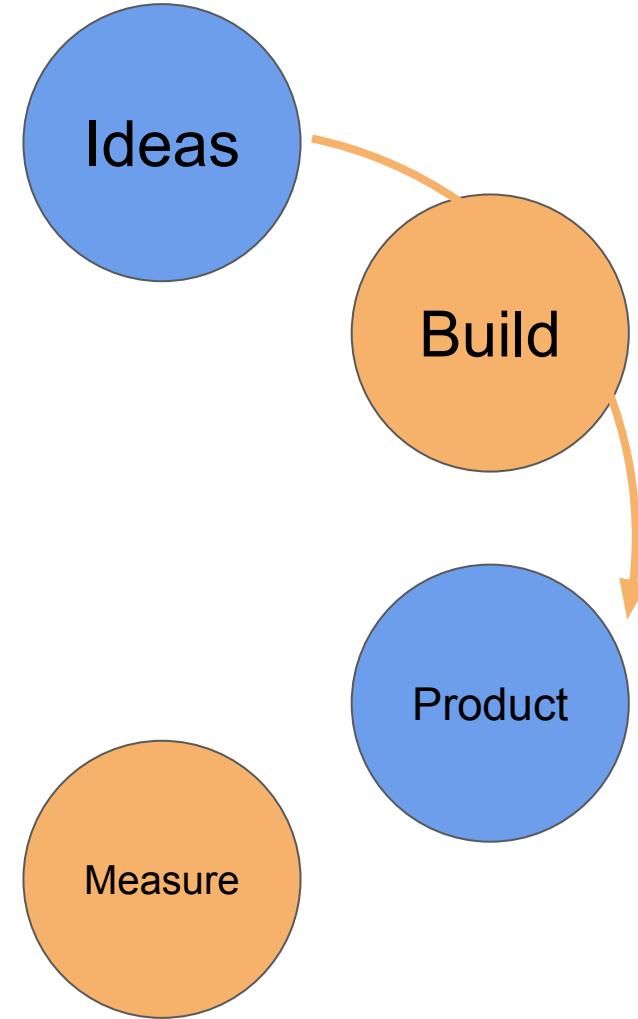
Ideas

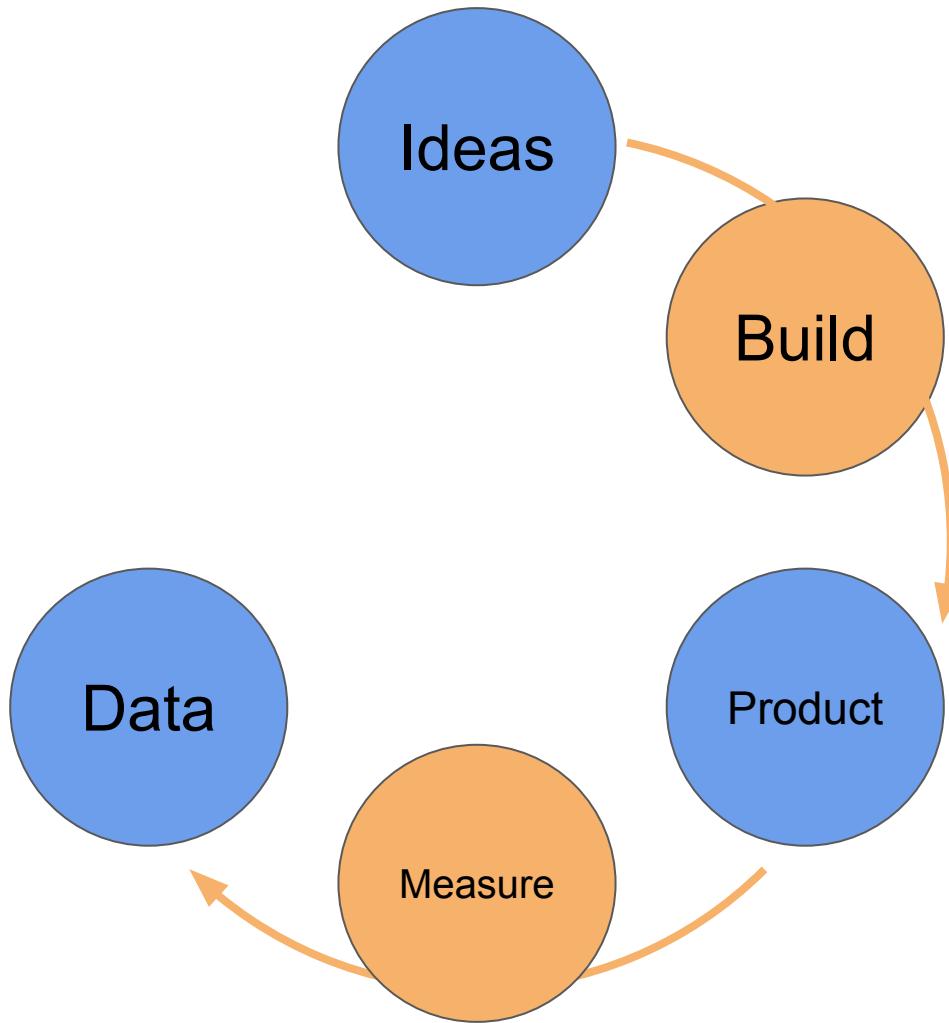


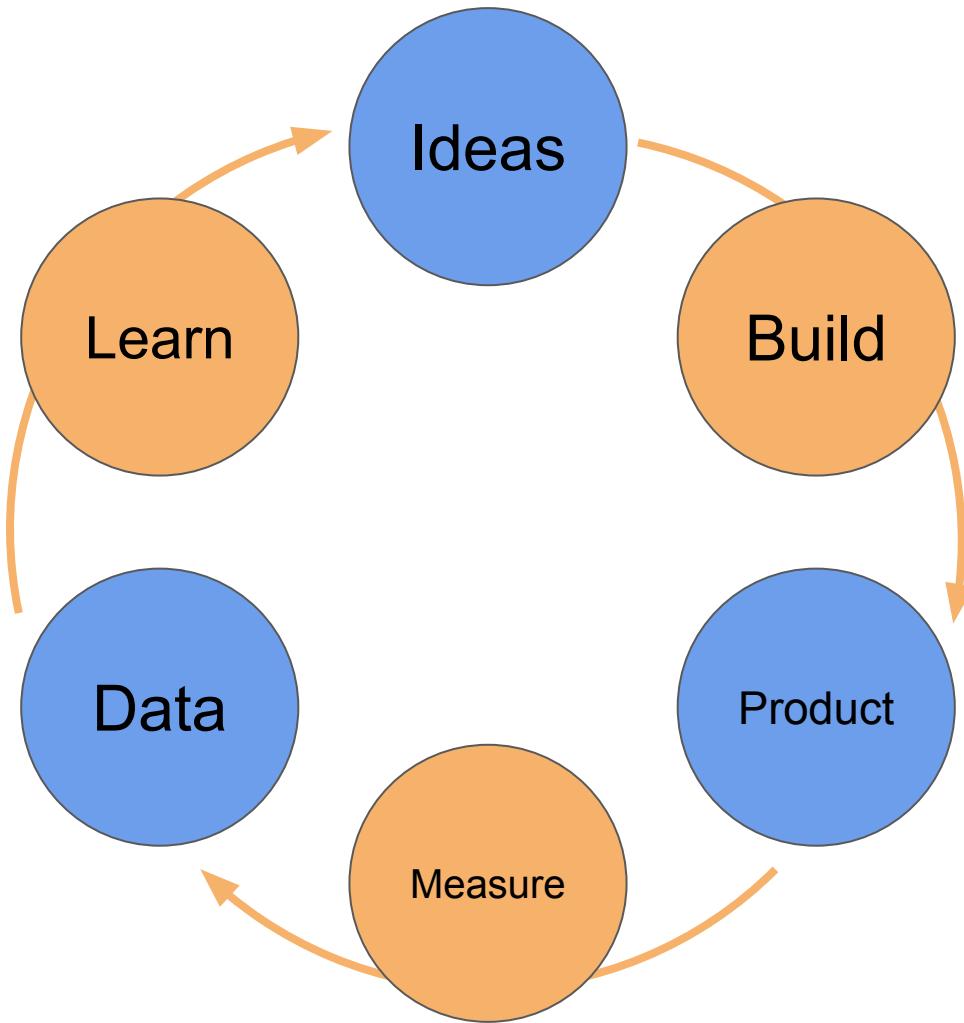
Ideas

Build











Some of the oldest F-35s on the globe are forced to sit and wait for upgrades
trib.al/CR0omrP



© Sightline Media Group

10:02 AM - 1 Jun 2018

The F-35 Could Reportedly Break Itself if it Goes Too Fast, And Other F-35 Problems



Kyle Mizokami

6/18/19 9:00pm • Filed to: F-35 SAGA ▾



68.1K



219

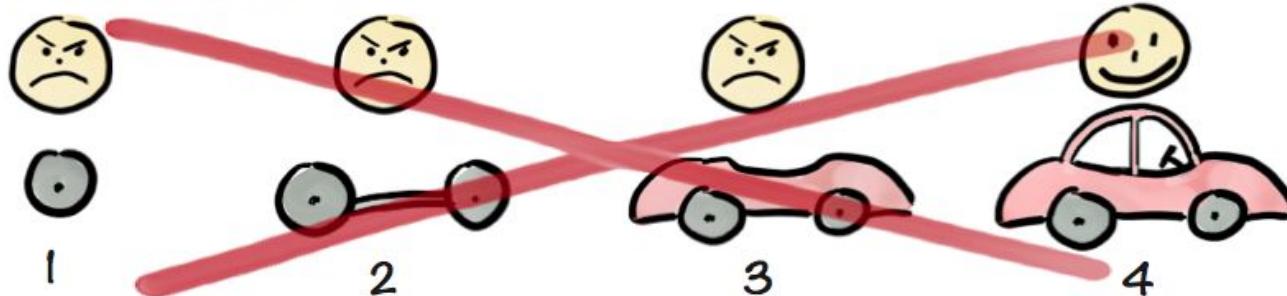


3

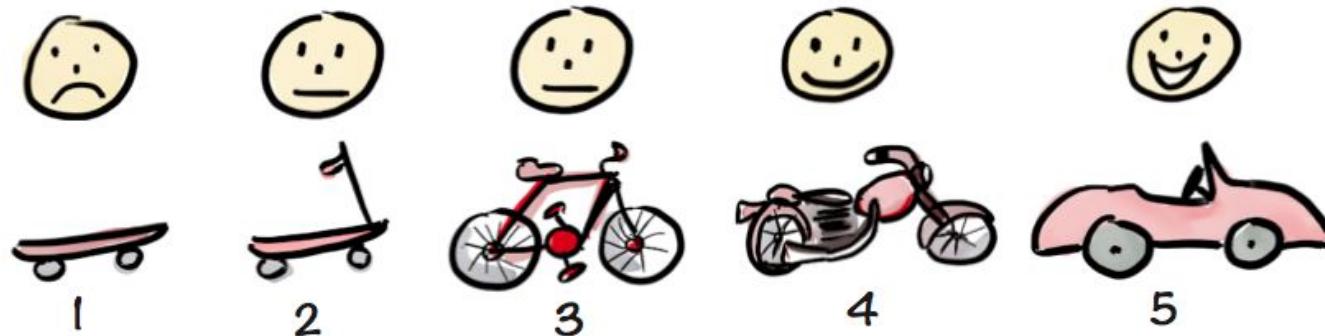


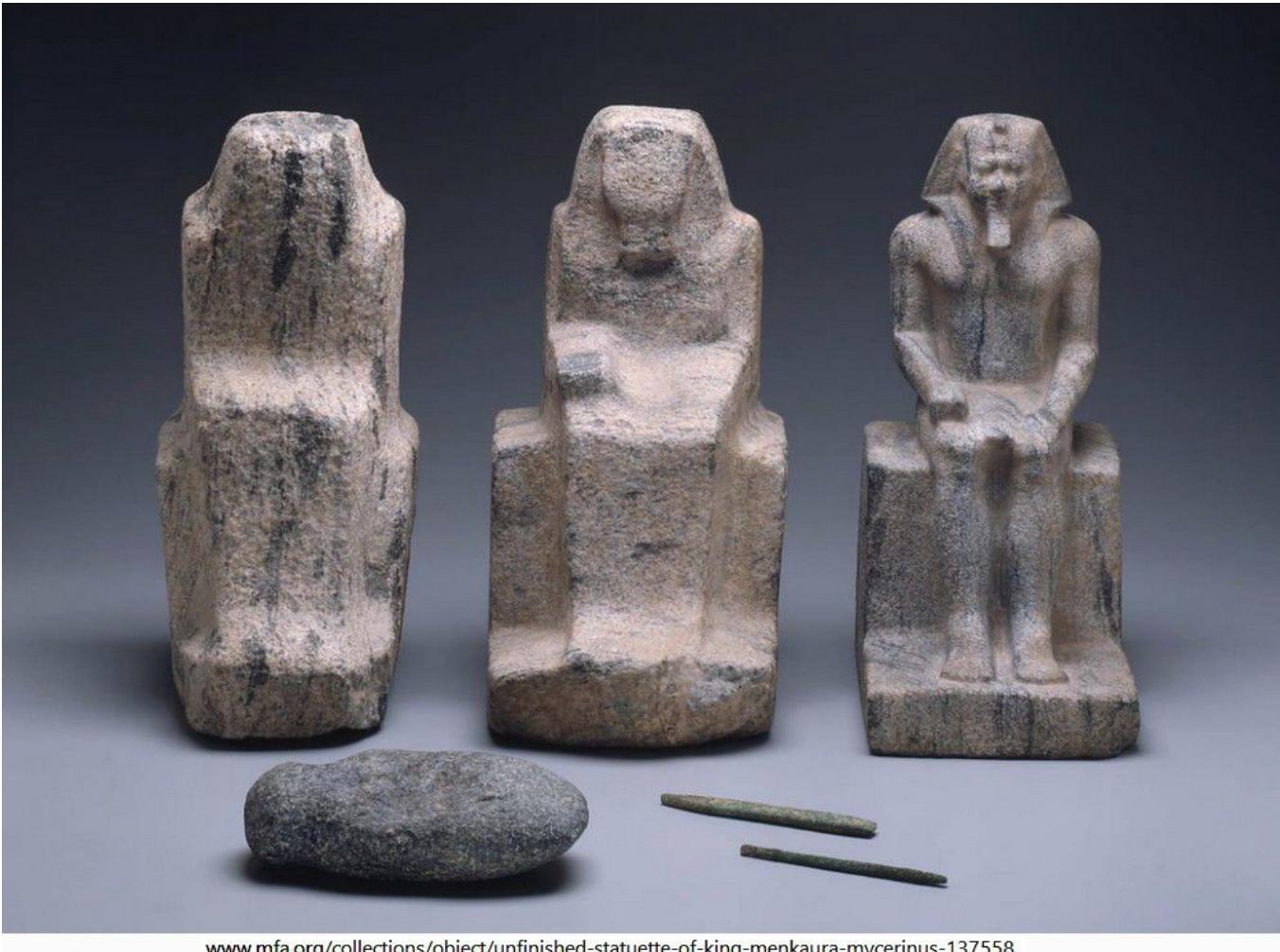
Photo: U.S. Department of Defense

Not like this....



Like this!





www.mfa.org/collections/object/unfinished-statuette-of-king-menkaura-mycerinus-137558

Containers

Messaging

Streaming Data

Containers

Containers

credit to Anne Currie



Bare Metal

Bare Metal

✓ Powerful

✓ Simple

Bare Metal

✓ Powerful

✓ Simple

✗ Brittle

✗ Inflexible

Bare Metal

✓ Powerful

✓ Simple

✗ Brittle

✗ Inflexible



Virtual Machines

Virtual Machines

✓ Flexible

✓ Networking

✓ Security

Virtual Machines

✓ Flexible

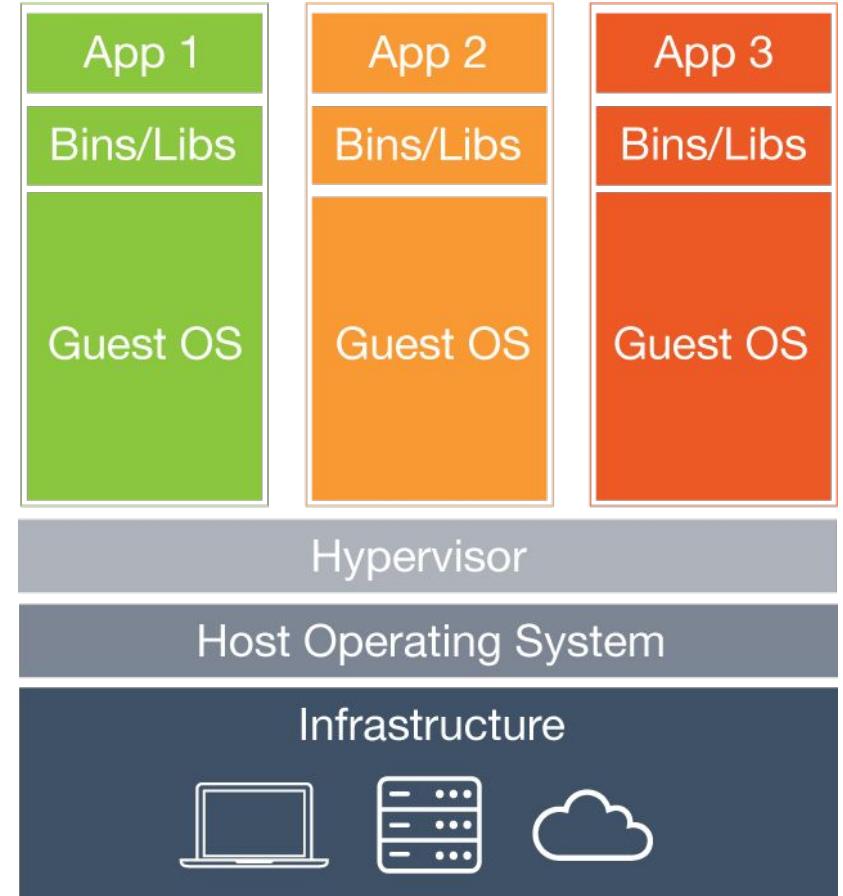
✓ Networking

✓ Security

✗ Overweight

Virtual Machines

- ✓ Flexible
 - ✓ Networking
 - ✓ Security
- ✗ Overweight



Virtual Machines

- ✓ Flexible
- ✓ Networking
- ✓ Security
- ✗ Overweight



Containers

Containers

✓ Lightweight

✓ Agile

Containers

✓ Lightweight

✓ Agile

✗ Untested

✗ Networking

✗ Security

Containers

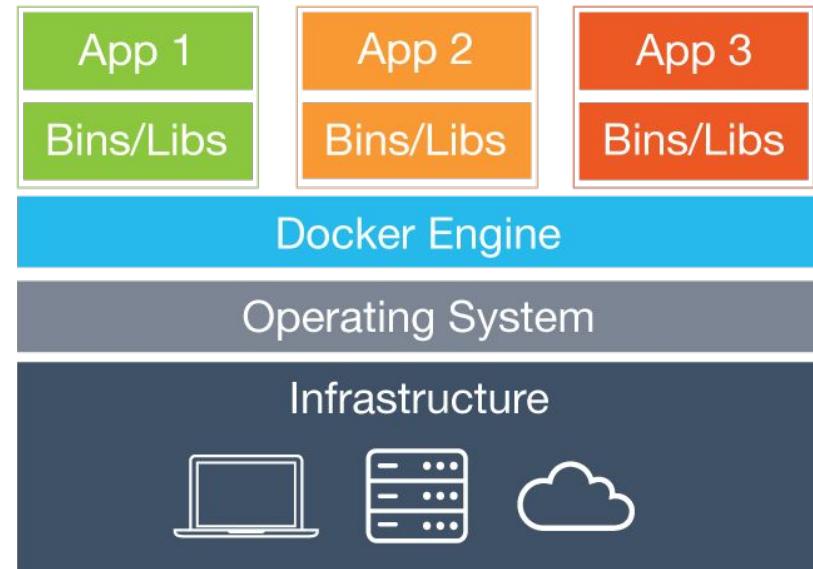
✓ Lightweight

✓ Agile

✗ Untested

✗ Networking

✗ Security



Containers

✓ Lightweight

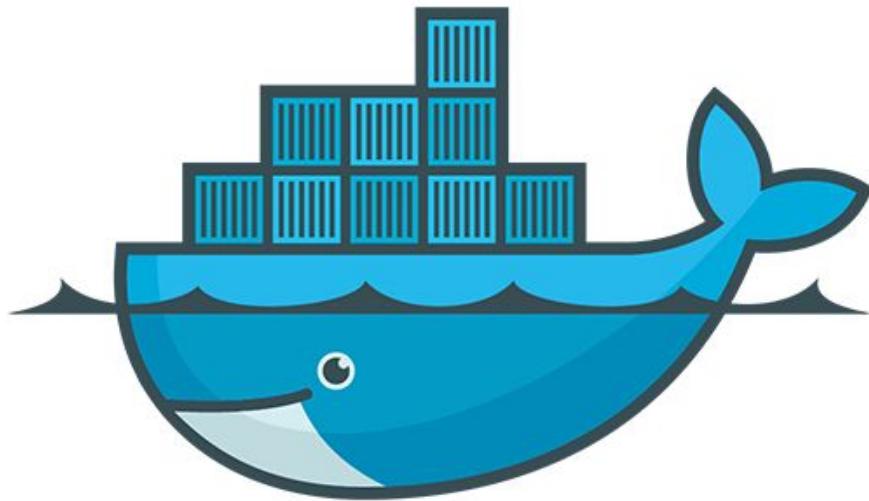
✓ Agile

✗ Untested

✗ Networking

✗ Security





docker

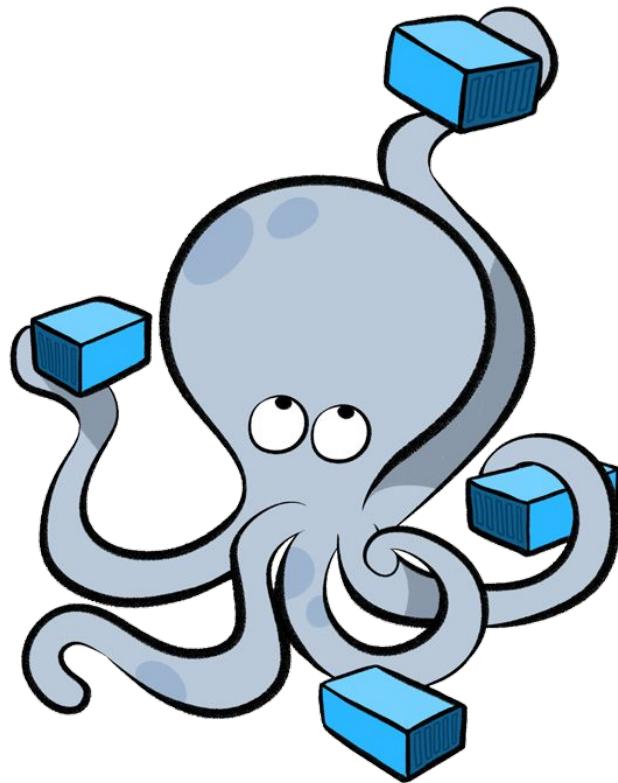
Exercise

Configuration Management



Docker Compose

Manage multi-container apps



Docker Compose

Manage multi-container apps

Collection of dockerfiles → services

```
version: '2'  
services:  
  web:  
    build: .  
    ports:  
      - "5000:5000"  
    volumes:  
      - ./code  
      - logvolume01:/var/log  
    links:  
      - redis  
  redis:  
    image: redis  
volumes:  
  logvolume01: {}
```

Is Docker production ready?

Is Docker production ready?

is docker production ready? Microphone Search

All Videos Images News Shopping More Settings Tools

About 168.000 results (0,61 seconds)

Docker, not production-ready? Not so, says Docker | InfoWorld
www.infoworld.com/article/.../docker-not-production-ready-not-so-says-docker.html ▾
Mar 22, 2016 - Docker has taken dev-and-test by storm, but production deployments have awaited more mature security and management. Some customers ...

Containers reality check: They're still not production-ready - TechBeacon
<https://techbeacon.com/containers-reality-check-why-theyre-still-not-production-ready> ▾
Just because containers are increasingly popular, however, doesn't necessarily mean they're ready for prime time. Docker and other container platform vendors ...

Docker in Production: A History of Failure – The HFT Guy
<https://thehftguy.com/2016/11/01/docker-in-production-an-history-of-failure/> ▾
Nov 1, 2016 - The issue with Docker is that it doesn't do any of that. It's just a dumb container system. It has the drawbacks of containers without the benefits. There are currently no good, battle tested, production ready orchestration system in existence.

Docker not ready for primetime | Hacker News
<https://news.ycombinator.com/item?id=12377457> ▾
Aug 29, 2016 - I have run docker in production at past employers, and am getting ready to do so again at my current employer. However I don't run it as a ...

Is Docker production ready?

is docker production ready?

All Videos Images News

About 168.000 results (0,61 seconds)

companies using docker

All News Videos Images

About 597.000 results (0,68 seconds)

Is Docker production ready?

Not great at backwards compatibility...

Requires extra tooling



kubernetes

is docker production ready?

All Videos Images News

About 168.000 results (0,61 seconds)

companies using docker

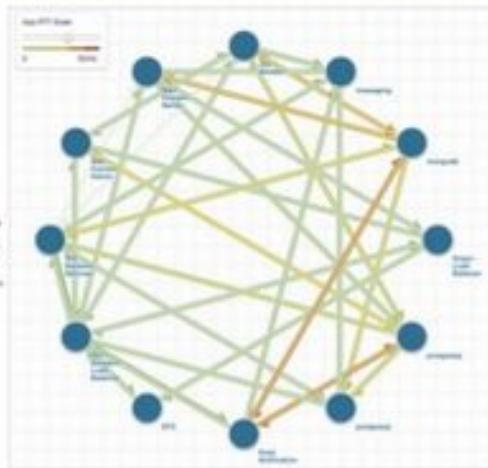
All News Videos Images

About 597.000 results (0,68 seconds)

“Death Star” Architecture Diagrams



Netflix



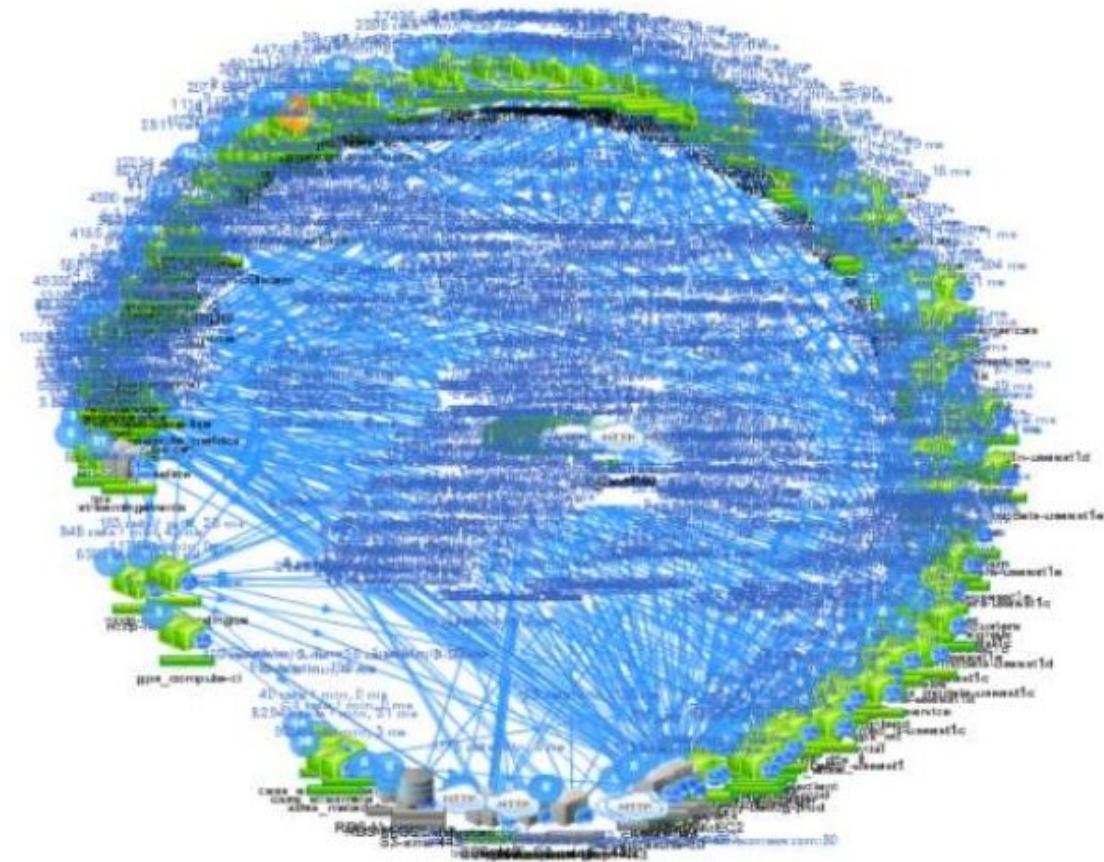
Gilt Groupe (12 of 450)



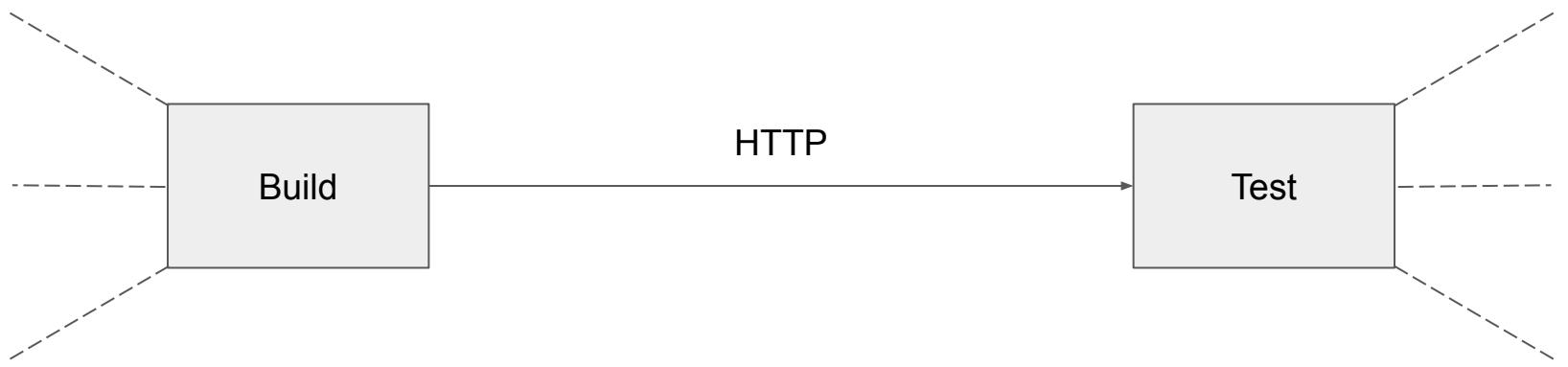
Twitter

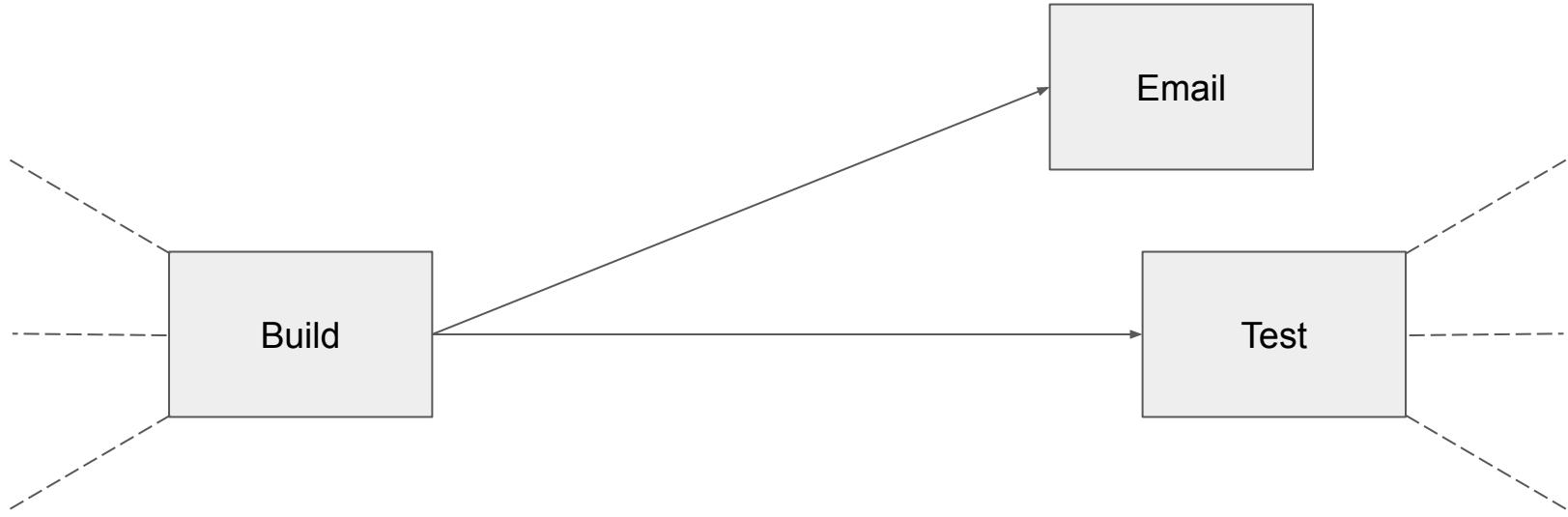
As visualized by Appdynamics, Boundary.com and Twitter internal tools

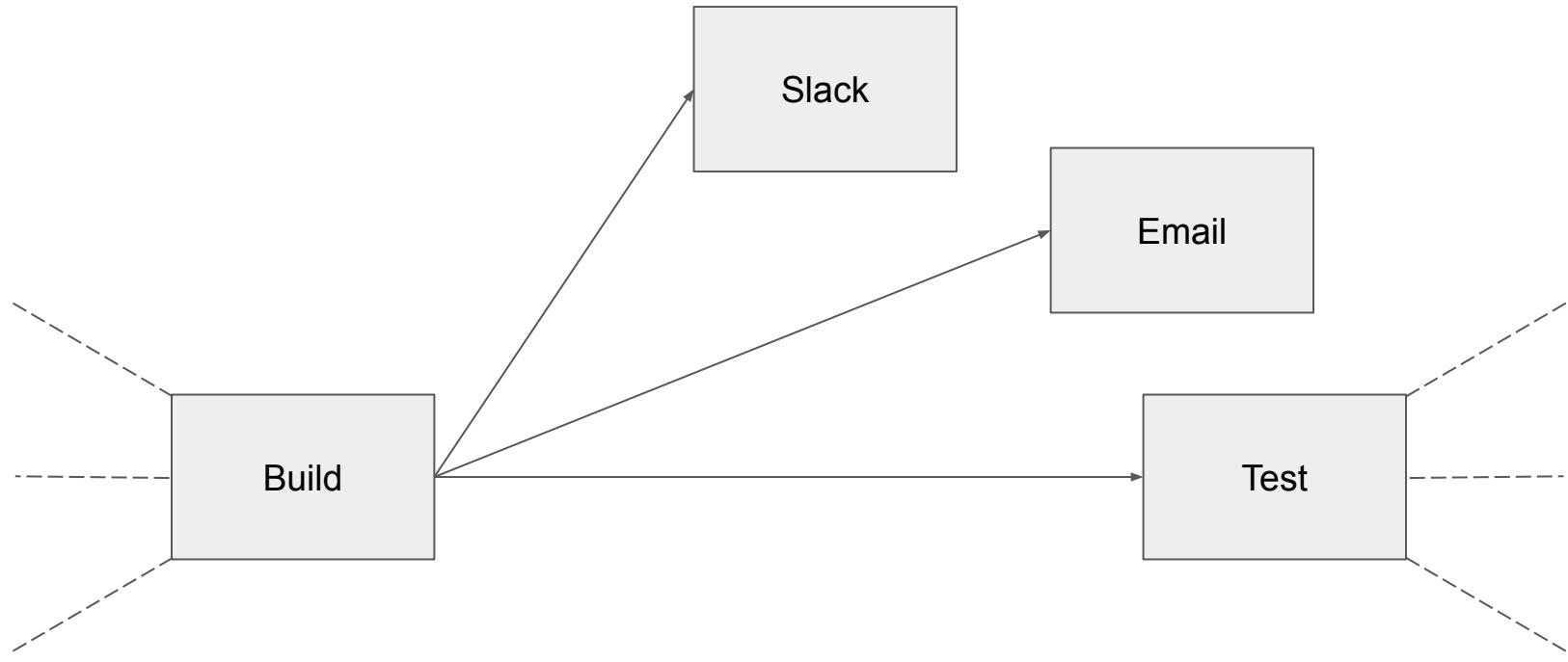
Netflix & Micro-Services

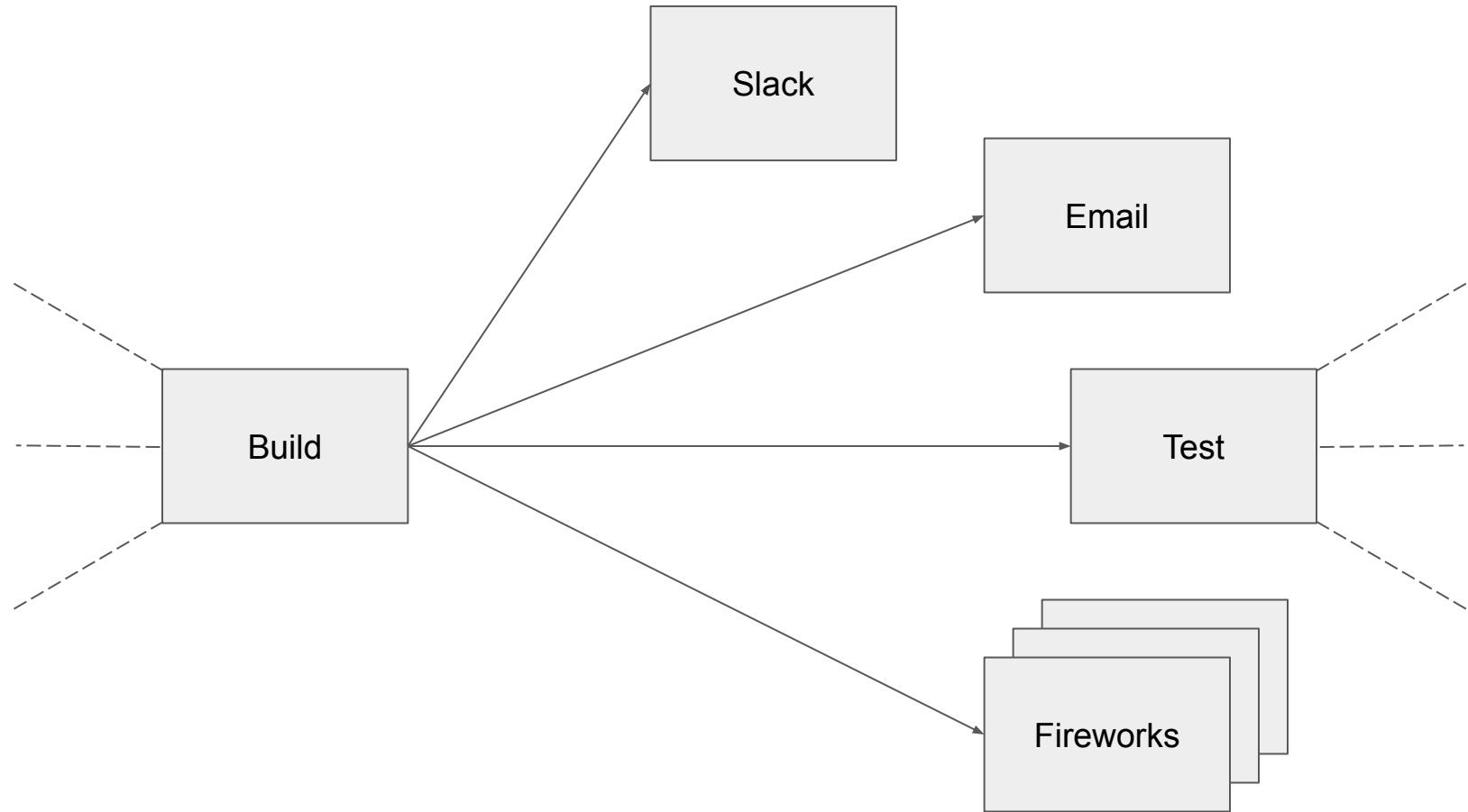


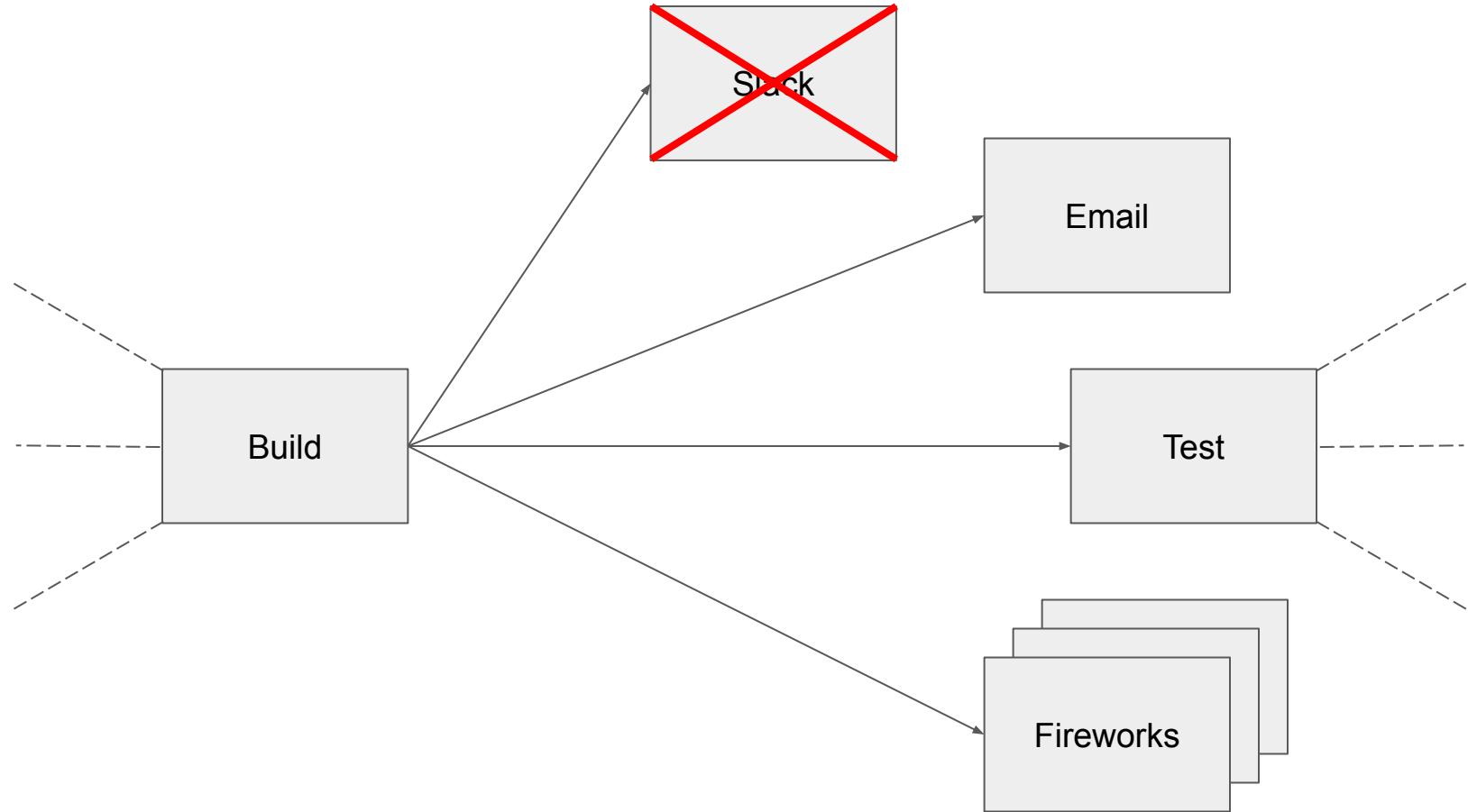
Inter-service communication

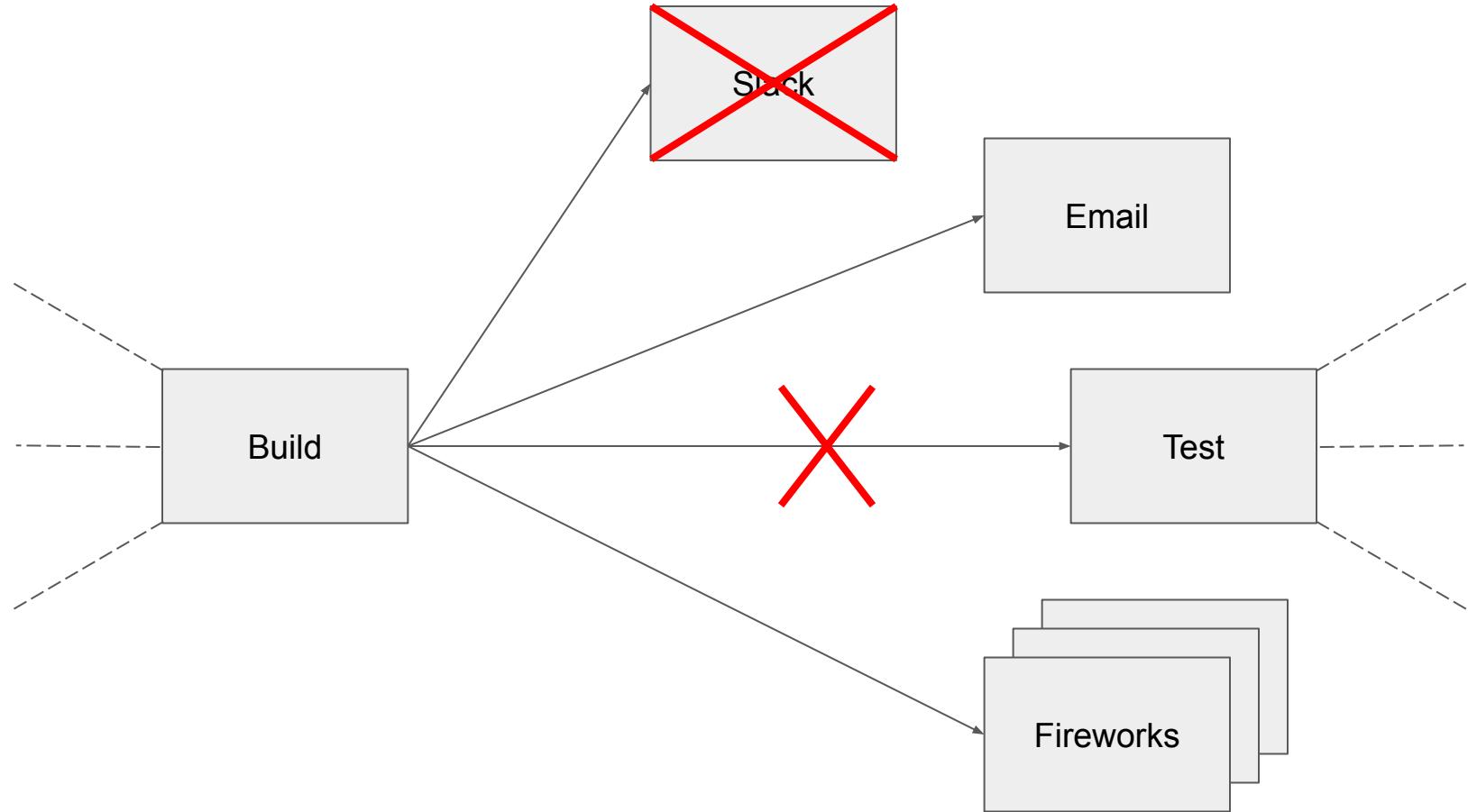












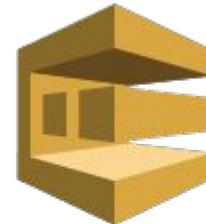


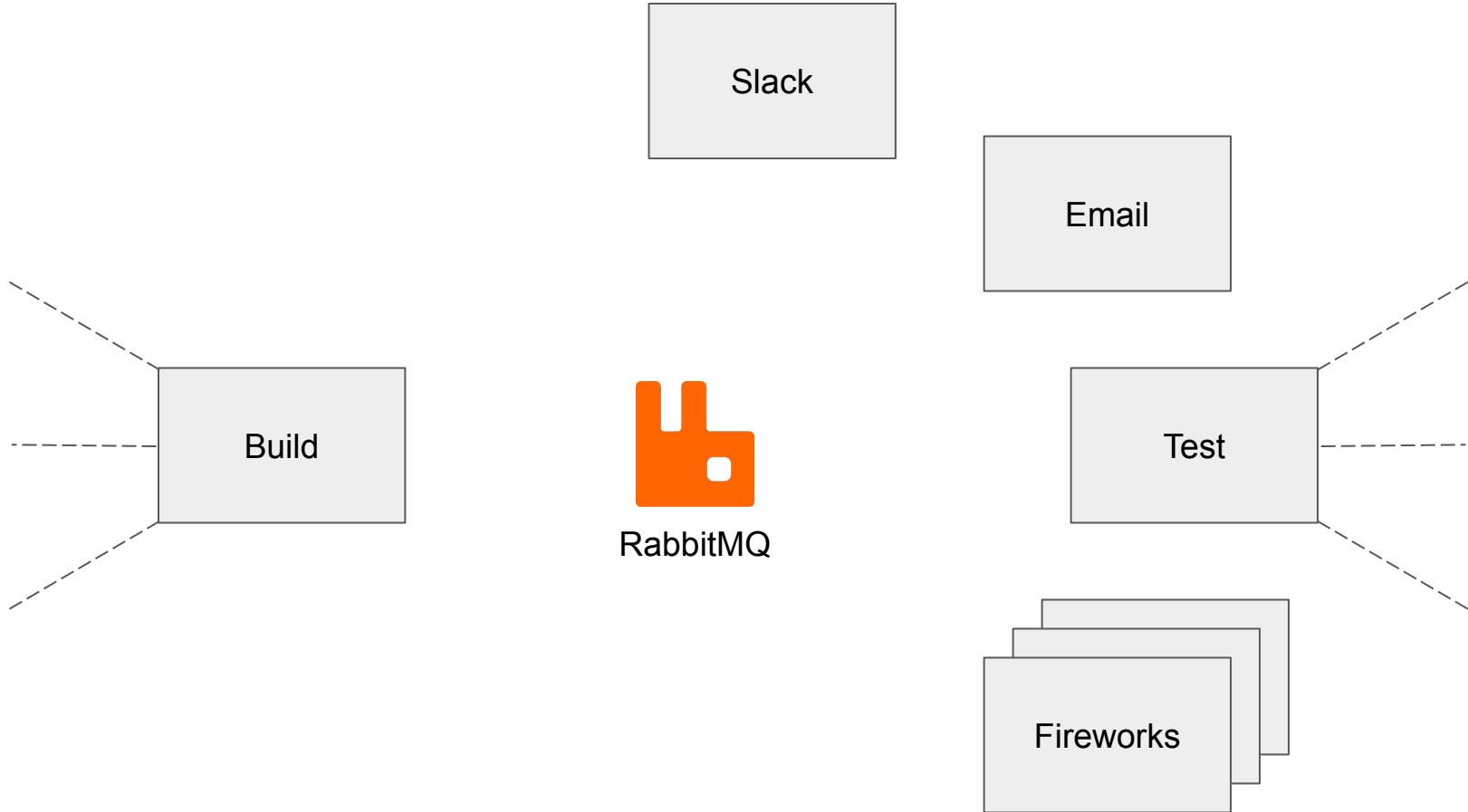
Service Discovery

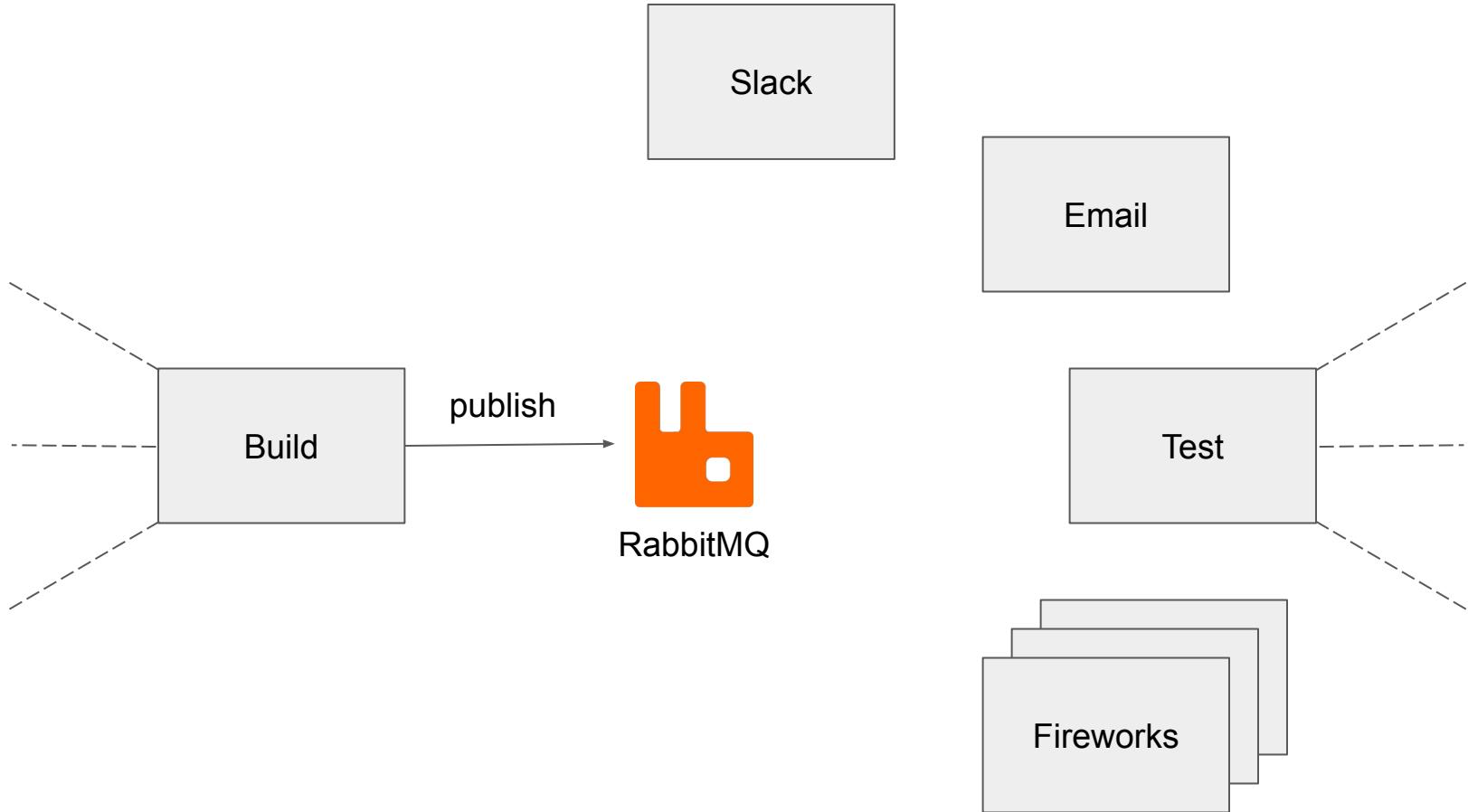
“Service Discovery
is an anti-pattern”

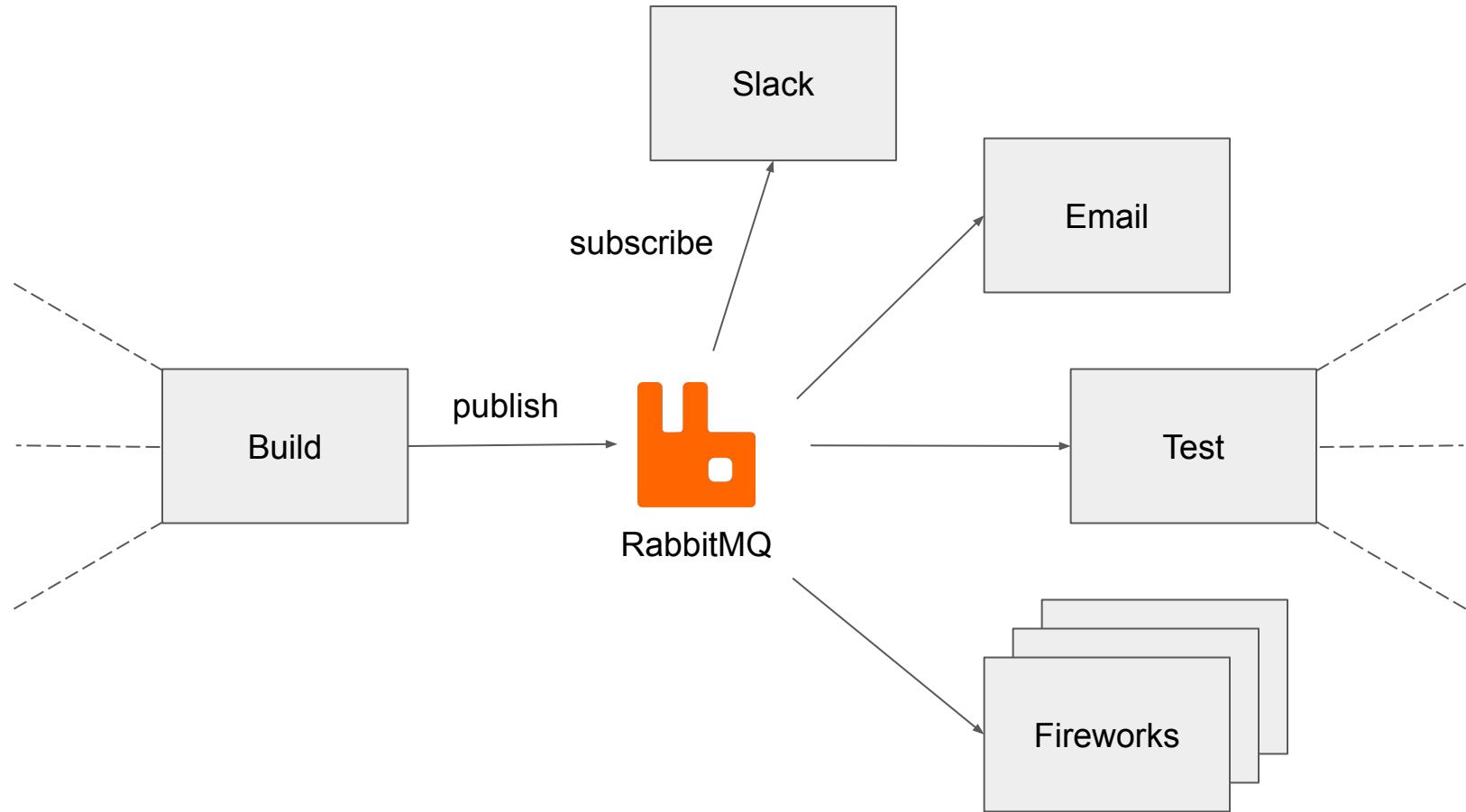
Richard Rodger, CTO @ nearForm

Messaging!









Time To Live

Time To Live

```
mq.publish(msg, {ttl: 10});
```

Time To Live

```
function put(msg, startTime) {  
    response = http.put(msg);  
    if (response.code != 200) {  
        if (getTime() - startTime < timeout) {  
            put(msg, startTime);  
        }  
    }  
  
    mq.publish(msg, {ttl: 10});  
}  
  
startTime = getTime();  
put(msg, startTime);
```

Time To Live

```
function put(msg, startTime) {  
    response = http.put(msg);  
    if (response.code == 404) {  
        // ???  
    } else if (response.code != 200) {  
        if (getTime() - startTime < timeout) {  
            put(msg, startTime);  
        }  
    }  
}  
  
mq.publish(msg, {ttl: 10});  
  
startTime = getTime();  
put(msg, startTime);
```

Time To Live

```
function put(msg, startTime) {  
    response = http.put(msg);  
    if (response.code == 404) {  
        // ???  
    } else if (response.code == 429) {  
        // ???  
    } else if (response.code == 500) {  
        // ???  
    } else if (response.code != 200) {  
        if (getTime() - startTime < timeout) {  
            put(msg, startTime);  
        }  
    }  
}  
  
startTime = getTime();  
put(msg, startTime);
```

Time To Live

```
function put(msg, startTime) {  
    response = http.put(msg);  
    if (response.code == 404) {  
        // ???  
    } else if (response.code == 429) {  
        // ???  
    } else if (response.code == 500) {  
        // ???  
    } else if (response.code != 200) {  
        if (getTime() - startTime < timeout) {  
            timeSpent = getTime() - startTime;  
            delay = Math.pow(timeSpent, 2);  
            setTimeout(put(msg, startTime), delay);  
        }  
    }  
}  
  
startTime = getTime();  
put(msg, startTime);
```

Time To Live

```
function put(msg, startTime) {  
    response = http.put(msg);  
    if (response.code == 404) {  
        // ???  
    } else if (response.code == 429) {  
        // ???  
    } else if (response.code == 500) {  
        // ???  
    } else if (response.code != 200) {  
        if (getTime() - startTime < timeout) {  
            timeSpent = getTime() - startTime;  
            delay = Math.pow(timeSpent, 2);  
            remainingTime = timeout - timeSpent;  
            delay = Math.min(delay, remainingTime);  
            setTimeout(put(msg, startTime), delay);  
        }  
    }  
}  
  
startTime = getTime();  
put(msg, startTime);
```

Time To Live

```
function put(msg, startTime) {
    response = http.put(msg);
    if (response.code == 404) {
        // ???
    } else if (response.code == 429) {
        // ???
    } else if (response.code == 500) {
        // ???
    } else if (response.code != 200) {
        if (getTime() - startTime < timeout) {
            timeSpent = getTime() - startTime;
            fuzz = Math.floor(Math.random() * 10);
            delay = Math.pow(timeSpent, 2) + fuzz;
            remainingTime = timeout - timeSpent;
            delay = Math.min(delay, remainingTime);
            setTimeout(put(msg, startTime), delay);
        }
    }
}

startTime = getTime();
put(msg, startTime);
```

Quality of Service

Quality of Service

```
// Best effort, fire and forget
mq.publish(unimportantMessage, {qos: 0});
```

Quality of Service

```
// Best effort, fire and forget
mq.publish(unimportantMessage, {qos: 0});
```

```
// At least once delivery
mq.publish(idempotentMessage, {qos: 1});
```

Quality of Service

```
// Best effort, fire and forget
mq.publish(unimportantMessage, {qos: 0});
```

```
// At least once delivery
mq.publish(idempotentMessage, {qos: 1});
```

```
// Exactly once delivery
mq.publish(importantMessage, {qos: 2});
```

Concurrency

Concurrency

Upload text → Generate video → Post video

Concurrency

Fast

Fast

Upload text → Generate video → Post video

Concurrency

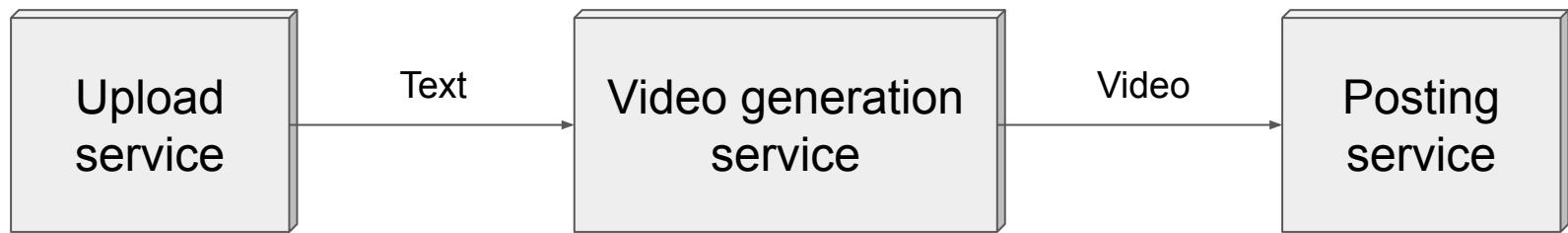
Fast

Slow

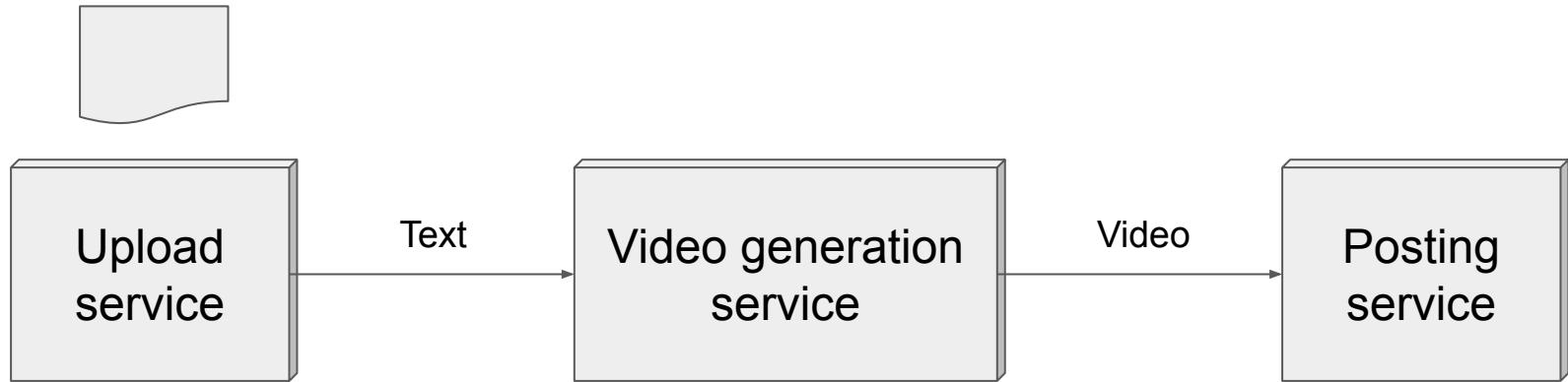
Fast

Upload text → Generate video → Post video

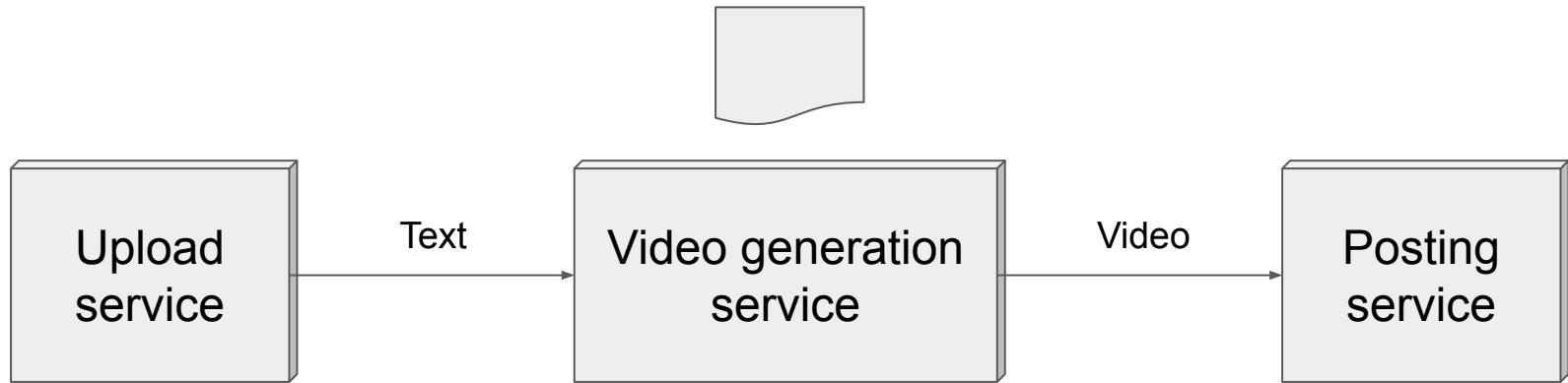
Concurrency



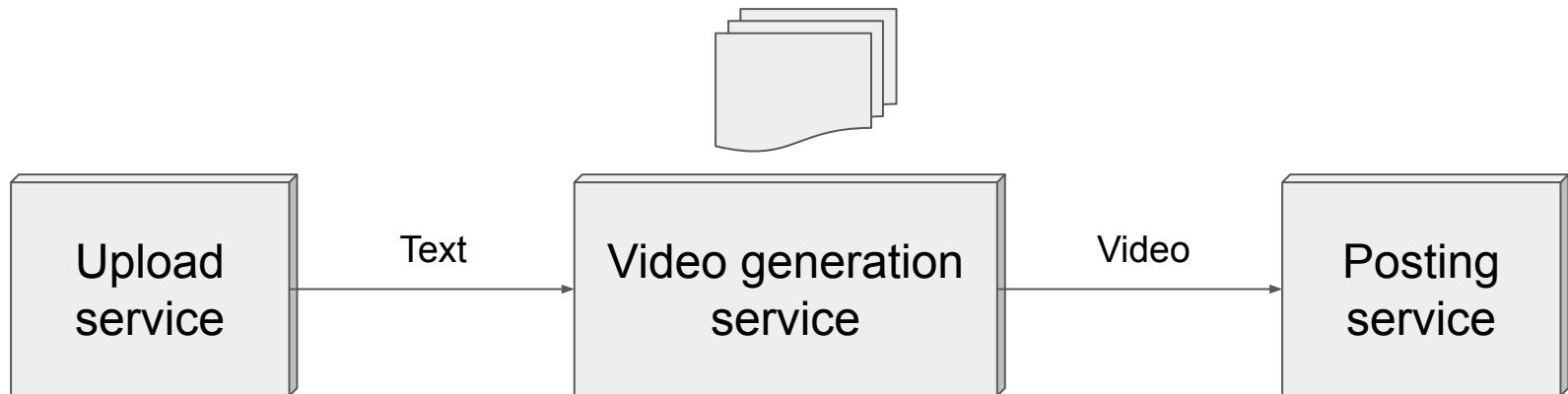
Concurrency



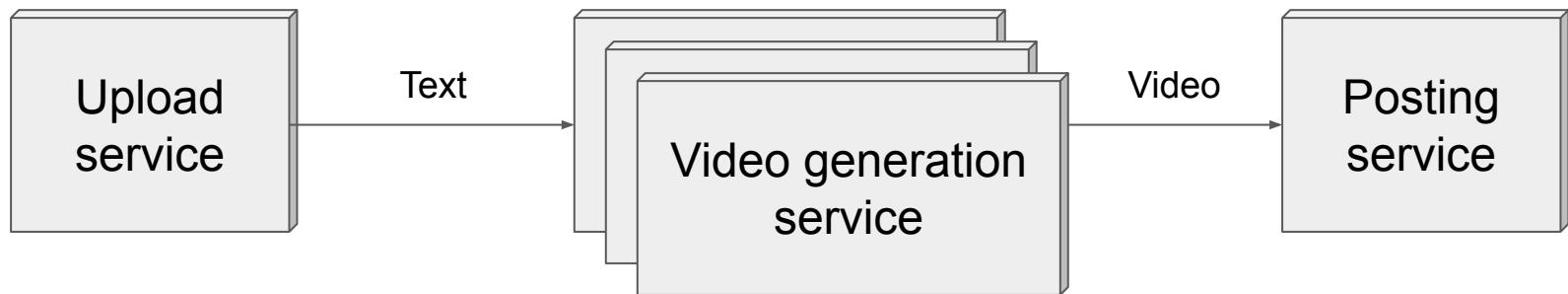
Concurrency



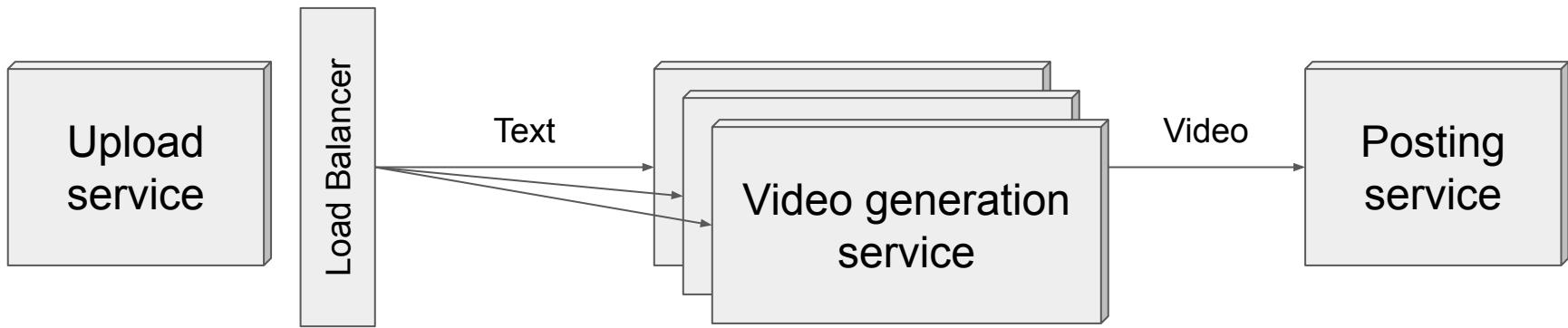
Concurrency



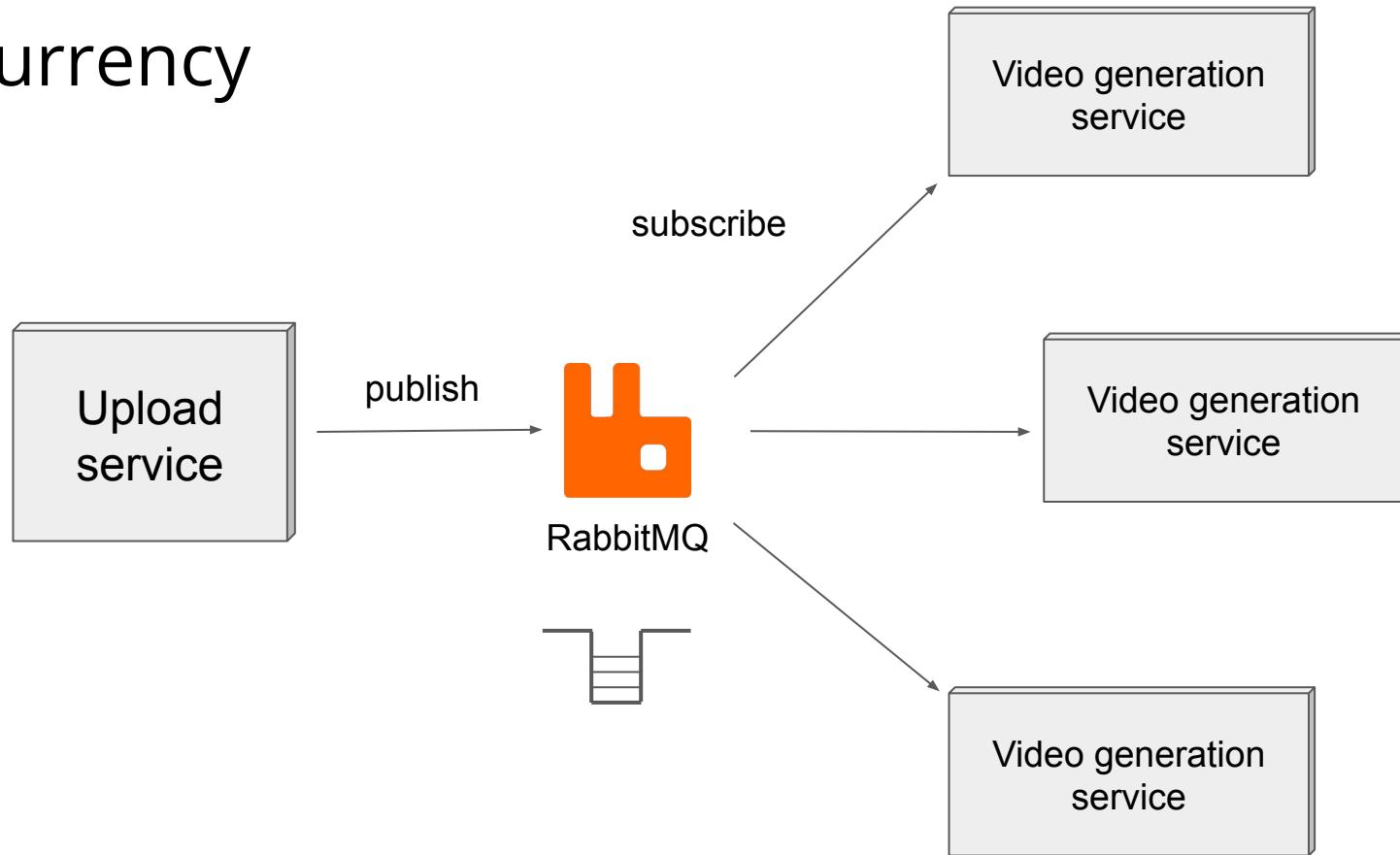
Concurrency



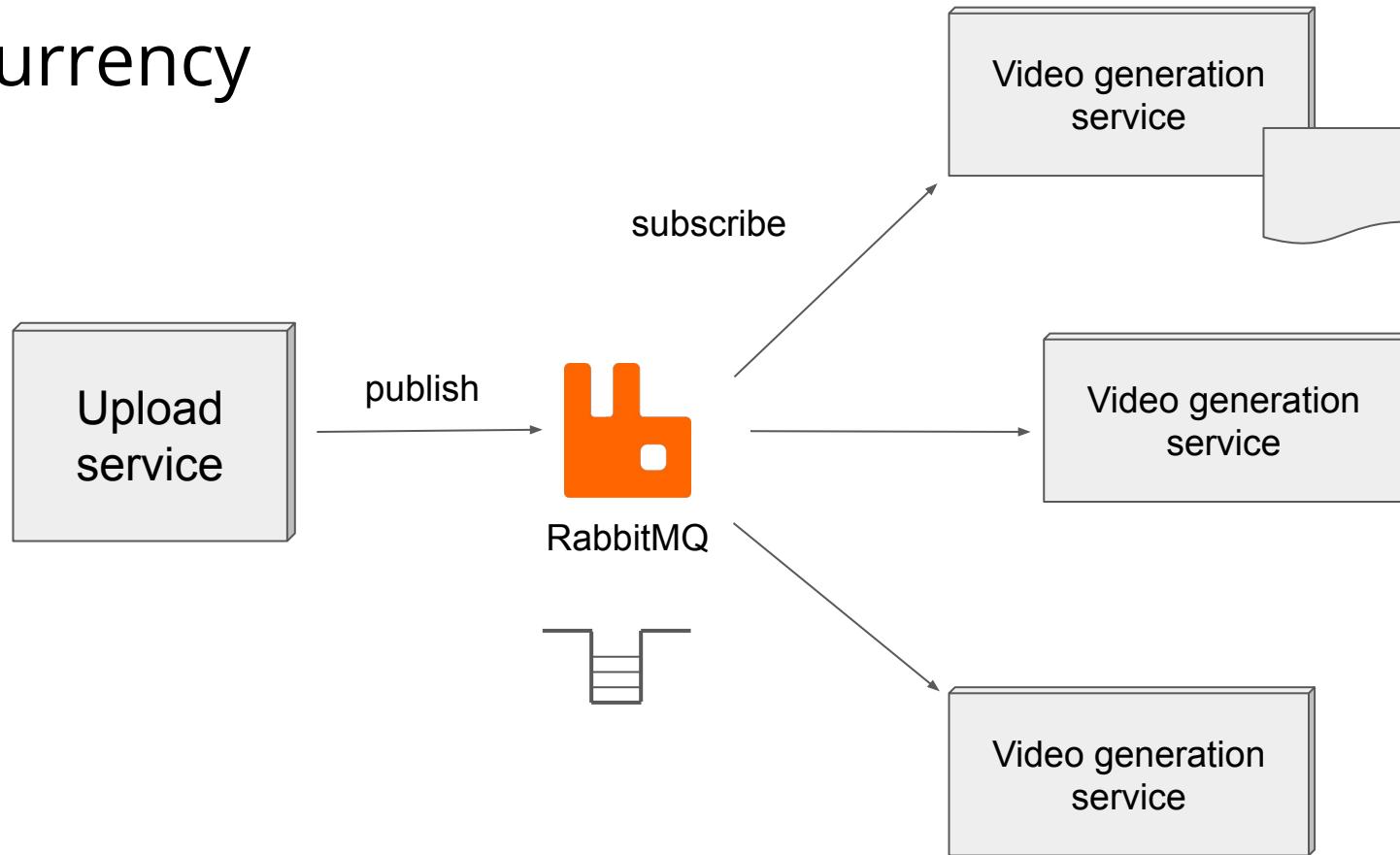
Concurrency



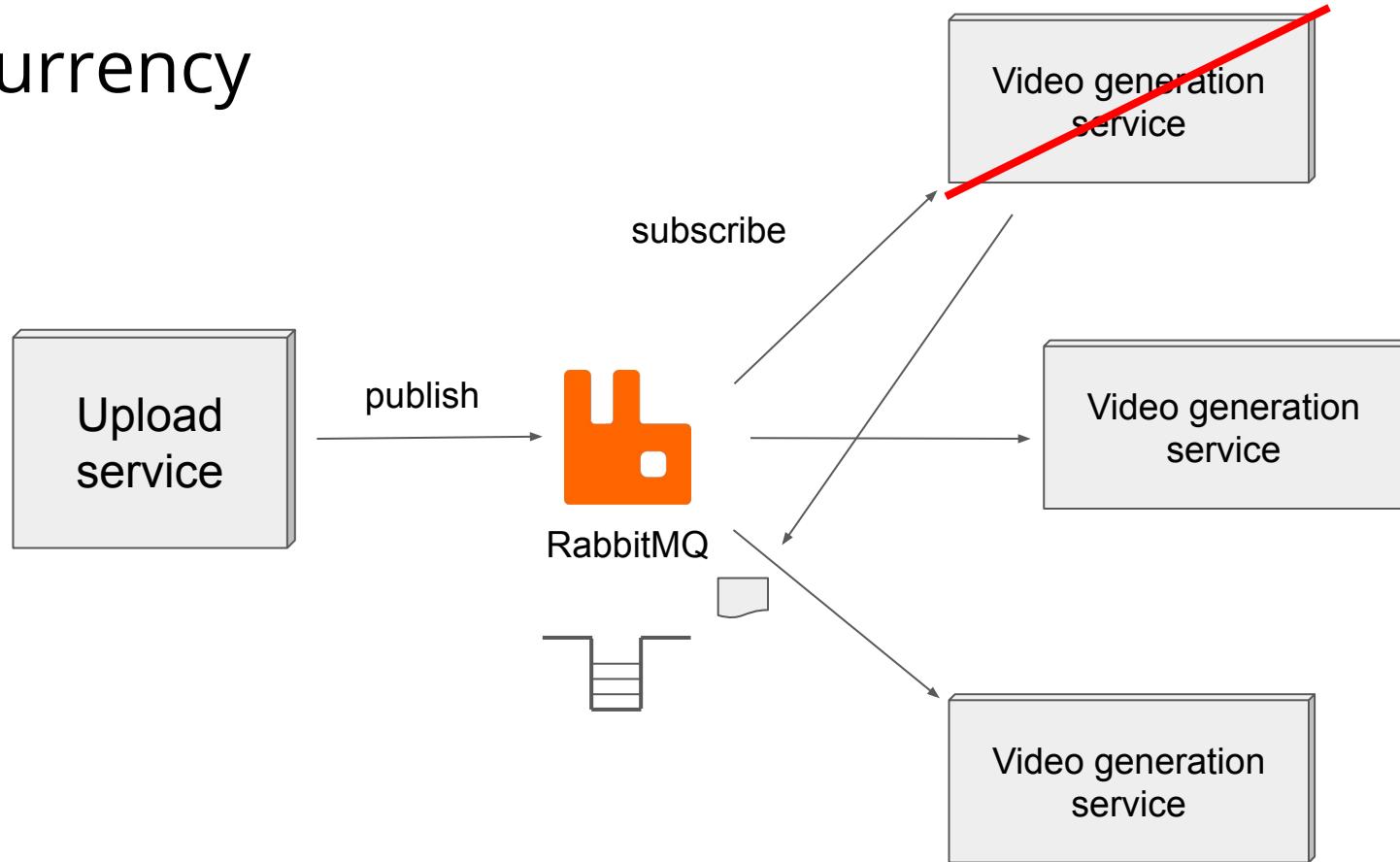
Concurrency



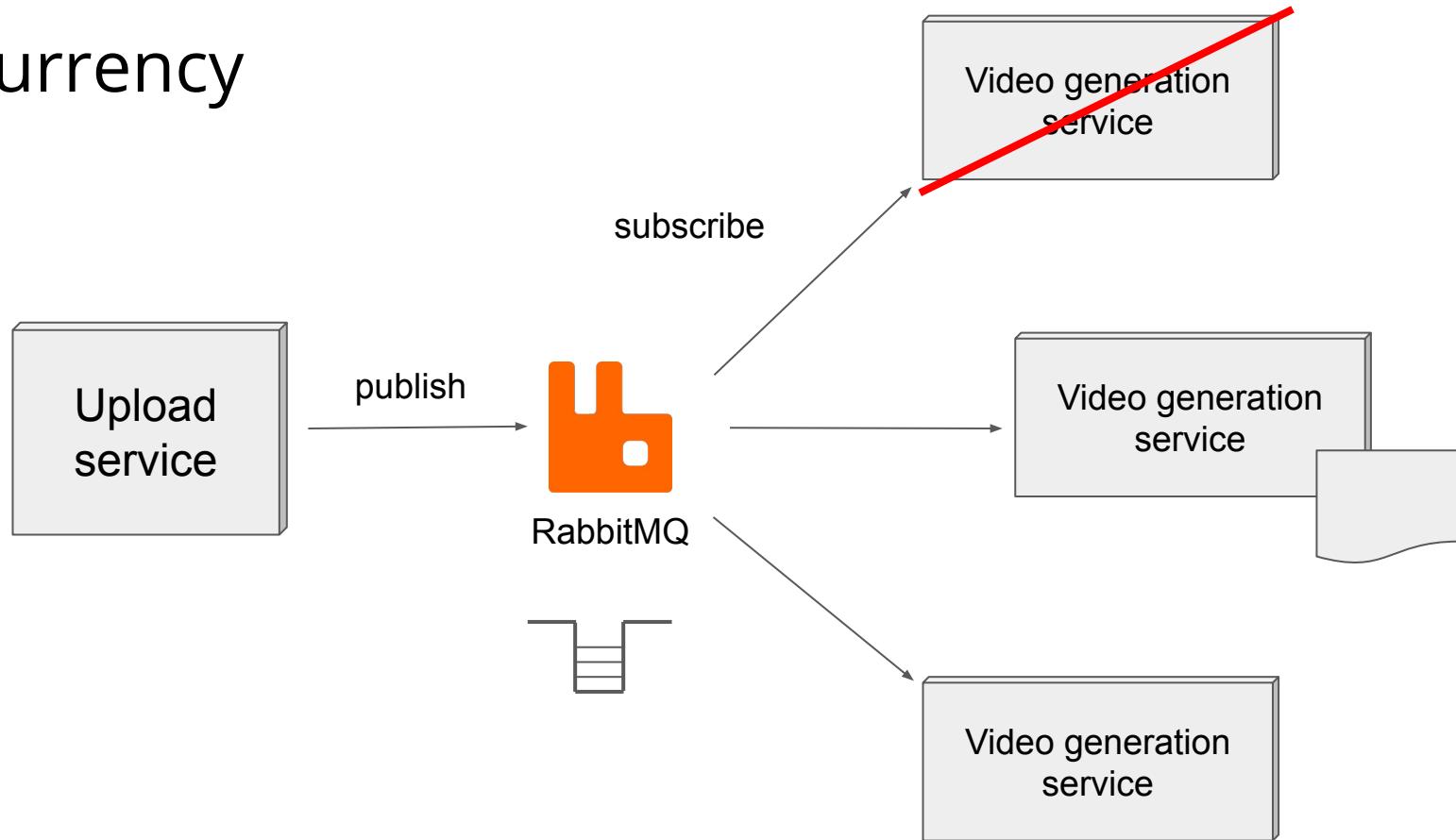
Concurrency



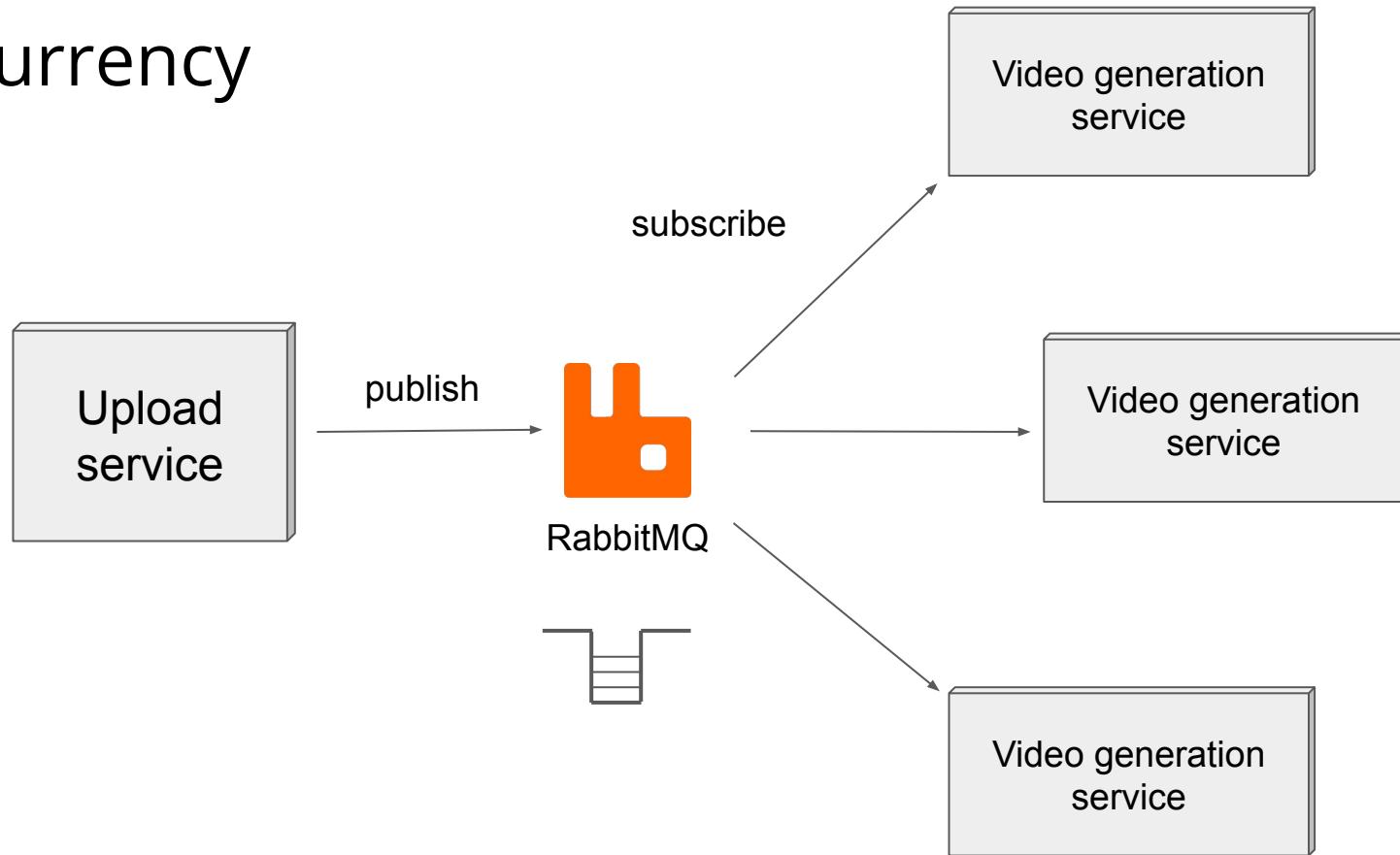
Concurrency



Concurrency



Concurrency



“Publish everything!”
Tom Livesey, Droplet

Build FAILED
Buildosaurus ANGRY

MQ Buildosaurus

Exercise



Lots of data sources



Lots of data sources

Lots of data consumers



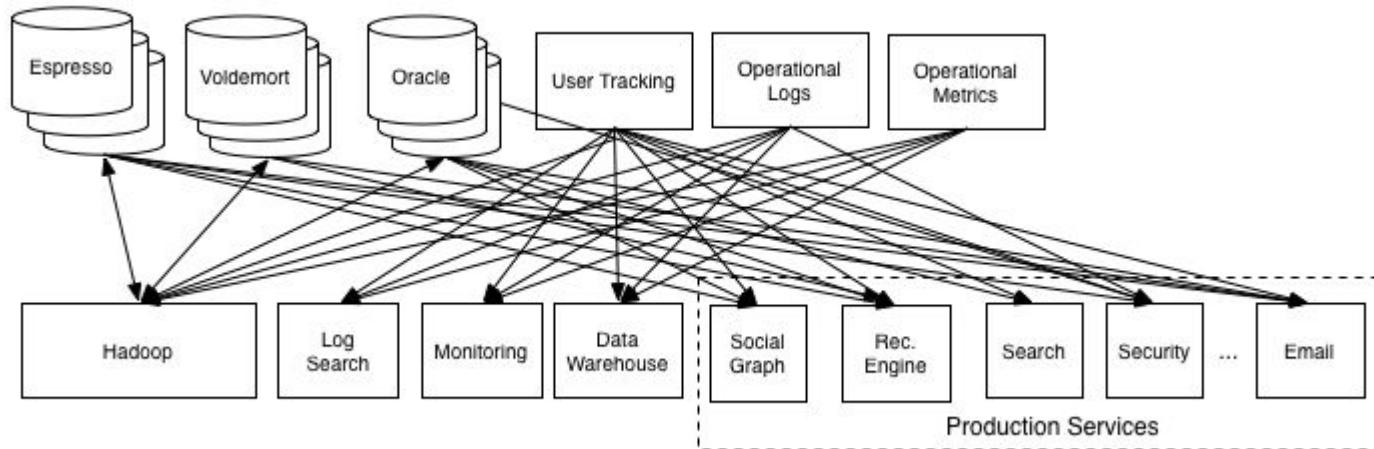
Lots of data sources

Lots of data consumers

Lots of integrations

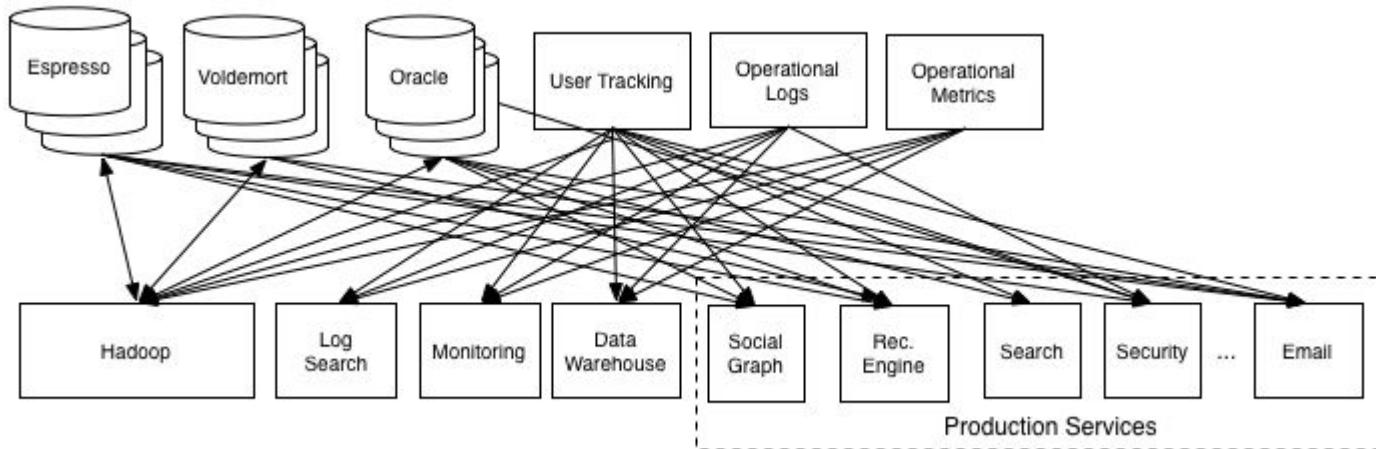


LinkedIn before

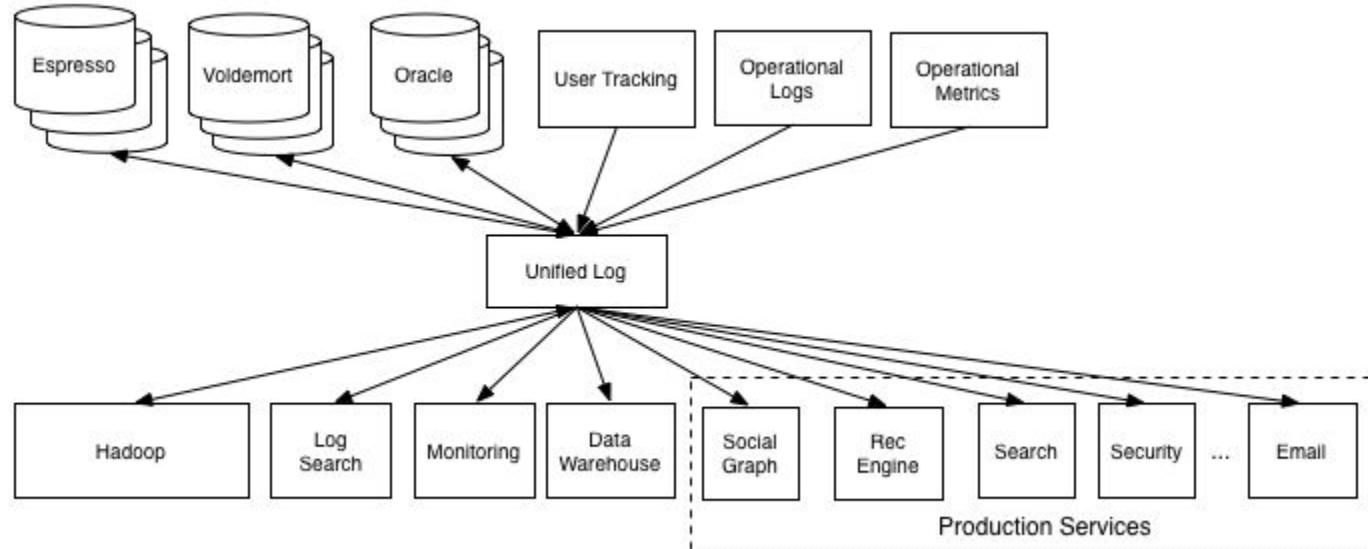


LinkedIn before

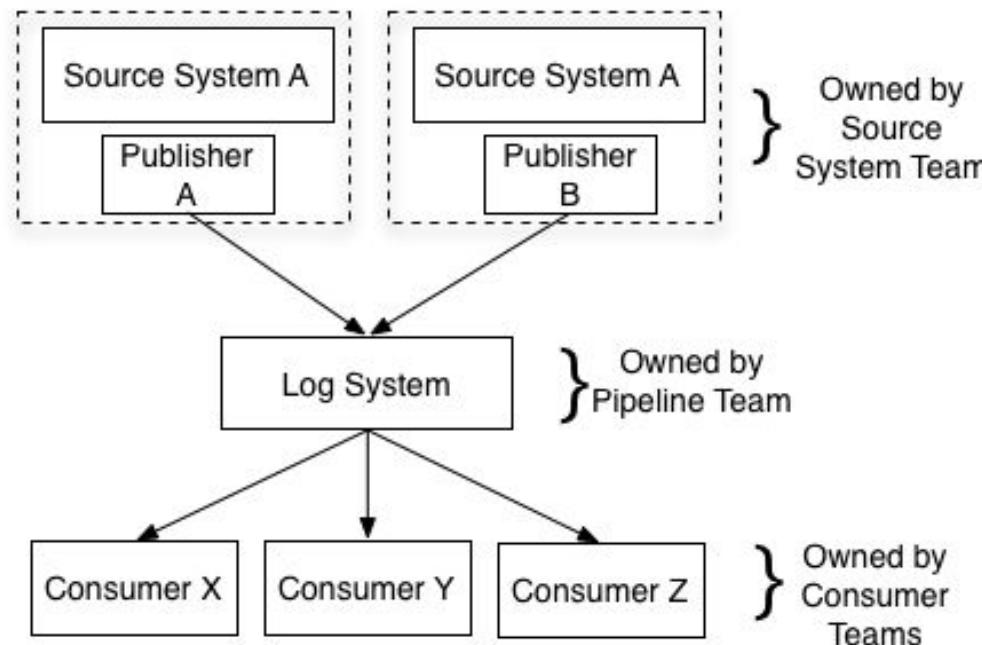
$O(n^2)$



LinkedIn after



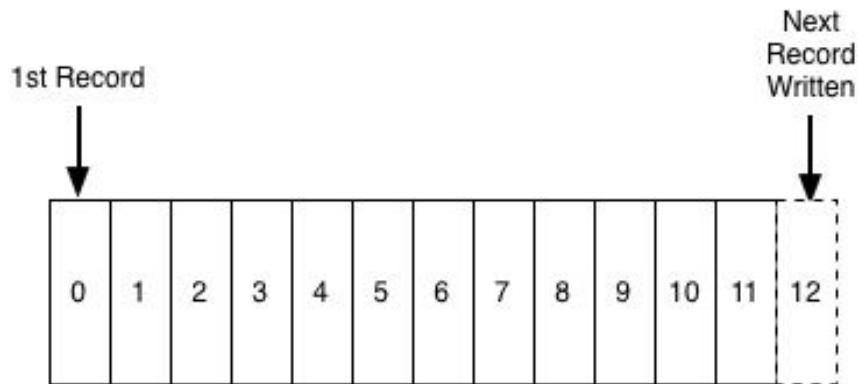
Up close



The log

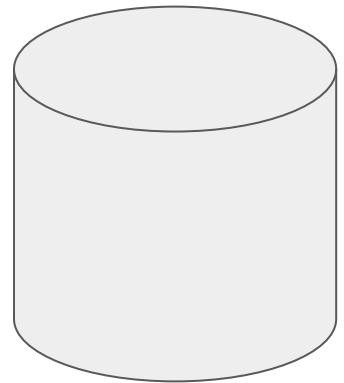
Append only

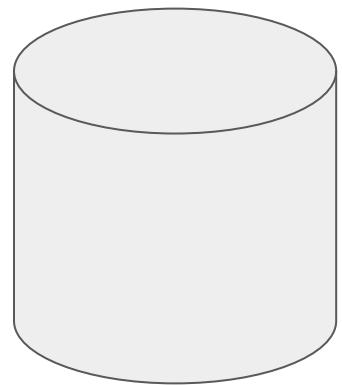
Time ordered



Distributed logging





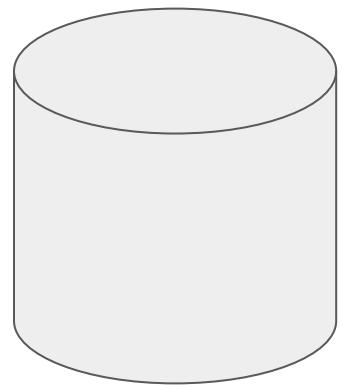


add(20)

add(5)

subtract(10)

add(15)



add(20)



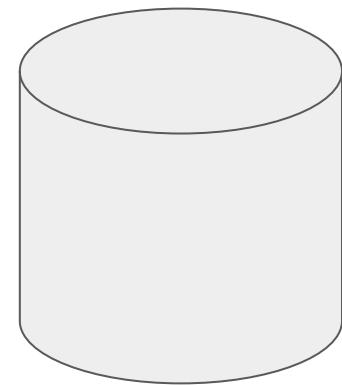
add(5)

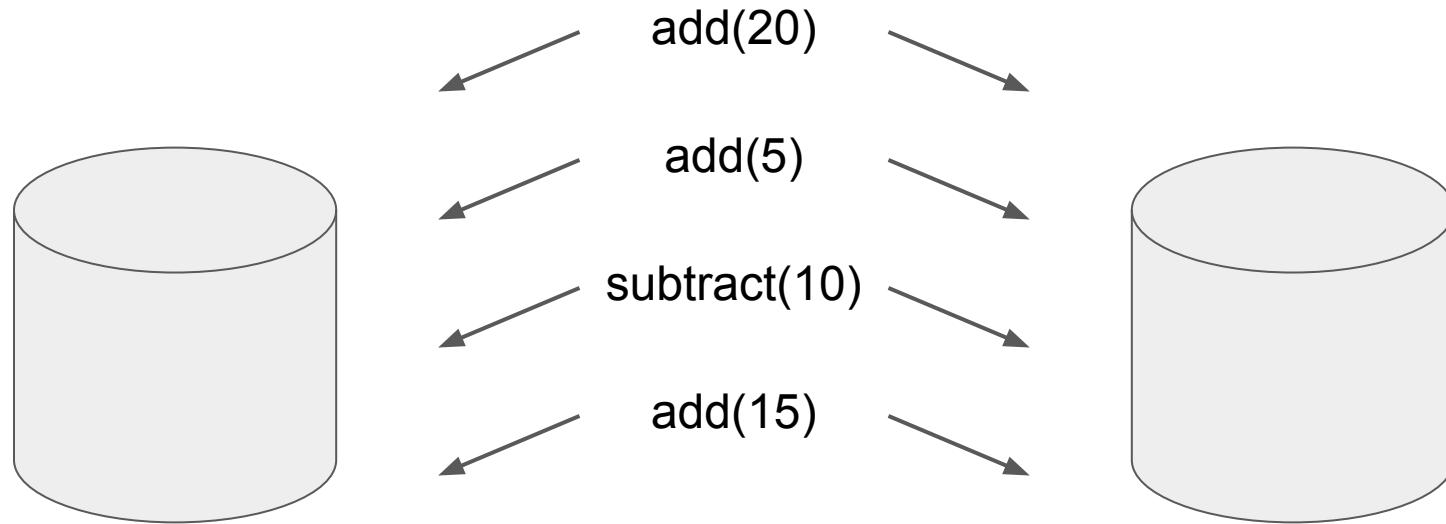


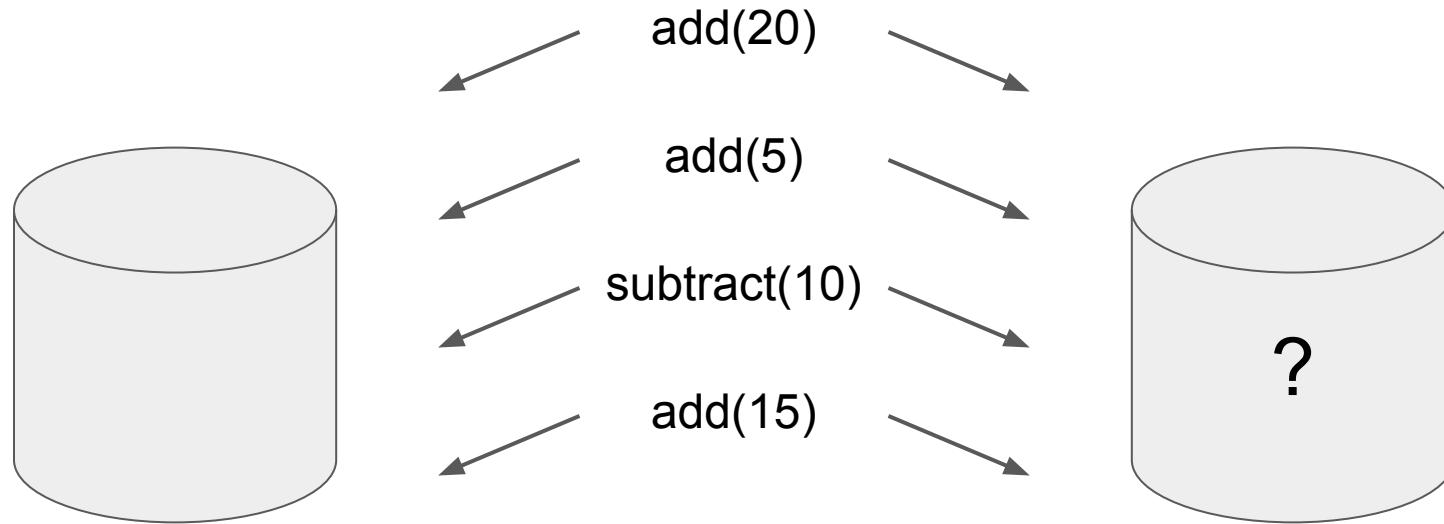
subtract(10)



add(15)





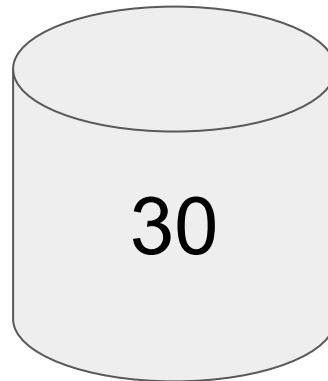


“If two identical, deterministic processes begin in the same state and get the same inputs in the same order, they will produce the same output and end in the same state.”

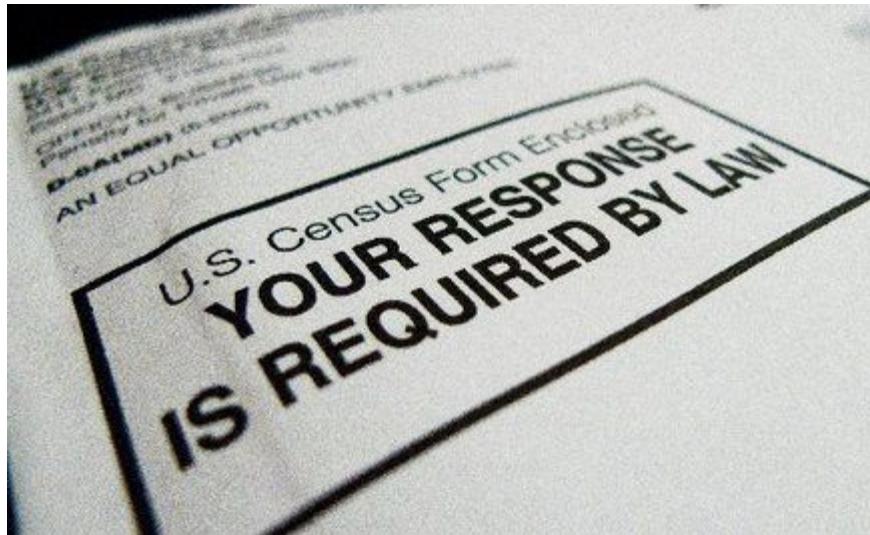
State Machine Replication Principle, Jay Kreps, LinkedIn

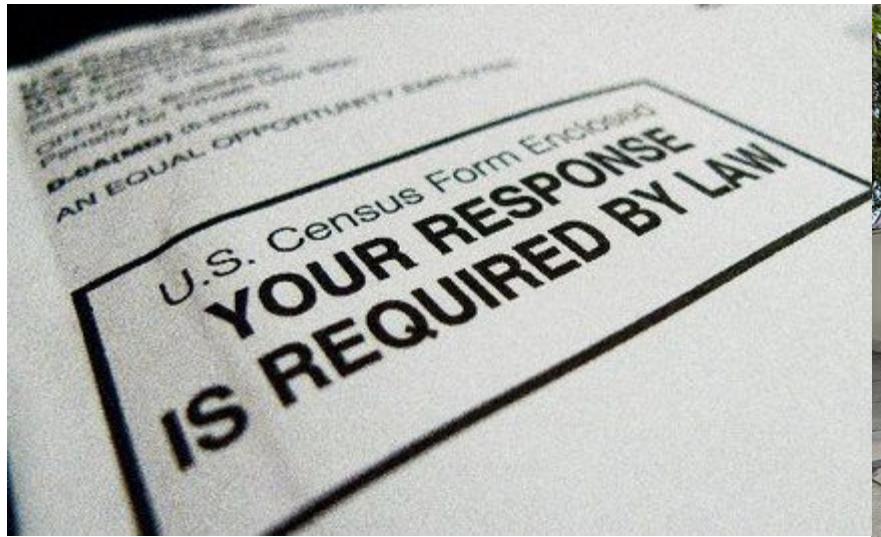
add(20)
add(5)
subtract(10)
add(15)

?



Log as source of Truth



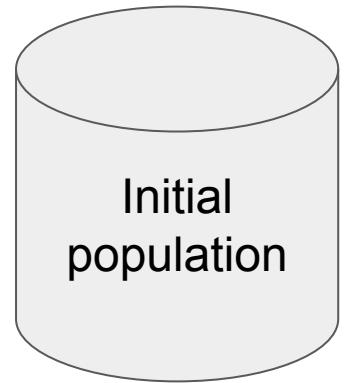


2010 RESIDENT POPULATION

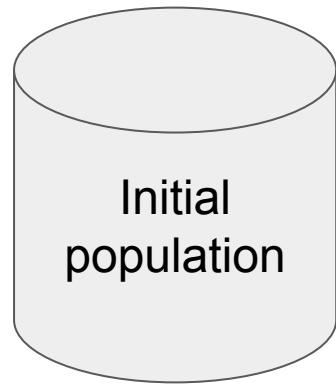


308,745,538

United States
Census
2010



Initial
population

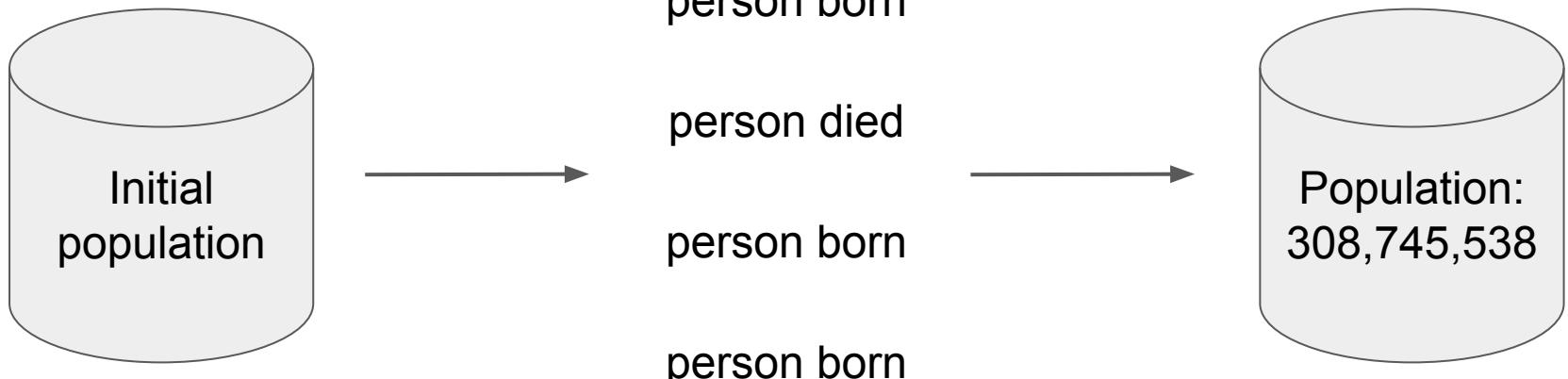


person born

person died

person born

person born





Summa de Arithmetica geo

metria. Proportioni: et proportionalita:
Monumentale impressa In Lofcolano su la riva vlt Benacense et
vnuco carpionito Laco: Amensimmo Sitos de li antique: &
evidenti ruine ol la nobil cit Benaco vita illustra:
to: Cum numerosta de Empatoris epiphaphis
di antique e perfette littere scalpiti vo:
tato: e cui finissimi e mirabil co:
lone marmozzi numeri
fragimenti di alaba:
stro porphyri e serpentini. Loco certo
lettori uno sifero occulto:
de mirata digne fote. Fil. Dr. OISTAR BENESTRO:
terra se ritro: BIBLIOTEK
HARO. STOCKHOLMS HOGSKOLA.

Continuentia de tutta lopera:

De numeri e misure in tutti modi
occurenti.
Proportioni e proportionalita notissima
vol 5: de Euclide e de tutti li altri
sui libri.
Quia unico euidente numero. 13. per
le quantita continue proportionali del
6: e 7: de Euclide extracte.
Tutte le parti de l'algoismo: cioè relesa:
re partem multiplicare: summare: e so:
trarre con tutte sue pag in santi e rotti
e radici e progreffioni.
De le regole mercantefca vita del. 3: e
sui fondamenti co' casi creplari p' m:
8: e guadagni per difteri transpotatio:
ni: e inuestite.
Partiri: multiplicare: summare: e sottrarre
le proportioni: e de tutte sotti radici.
De le tre regole del Catay ditta posi:
zione: e sua origine.
Quidientie generali: ouer conclusioni nu:
mero. 66. ab soluere ogni caso che per
regole ordinarie non si podesse.
Tutte sorte binomii: e recisi: e altre linee
irrationali del decimo de Euclide.
Tutte regole de Algebra ditte de la colla

e lo fabrichi e fondamenti.
E spoglie in tutti modi: e le parti.
Socde de bestiam: e le parti.
Fatti: pescioni: e cottimi: iuello: logazioni:
e godimenti.
Varatti in tutti modi semplici: compo:
site col tempo.
Lambi: real: sechy: fittiti: e dimpuniti:
ouer communi.
Termini
Adritti semplici: e a capo: vanno: e altri
Rifitati: adritti: contide: tempo: e venars: e de:
recare a via di piu partire.
Quargentis: ellos: affirmare: e corattare.
Abolti: caff: e ragioni: straordinarie: va:
rice: diuersitate: a tutto occurrerete: come
nella sequente taula appare: ordinamente
de tutte.
Ordine a faser tener ogni cotoe scriptu:
ree de le quaderno in vinegia.
Tariffa de tutte vianze: e costumi mer:
cantefci in tutto il mondo.
Paventica e theorica de geometria: e de li
cinque corpi regulari: e altri dependenti
E molte altre cose de grandissimi piace:
rize frutto: comino costitualmente per
la sequente taula appare.



BLOCKCHAIN





2 million writes s⁻¹
(on 3 cheap machines)

10s of millions of writes s⁻¹
(LinkedIn production)



2 million writes s⁻¹
(on 3 cheap machines)

10s of millions of writes s⁻¹
(LinkedIn production)

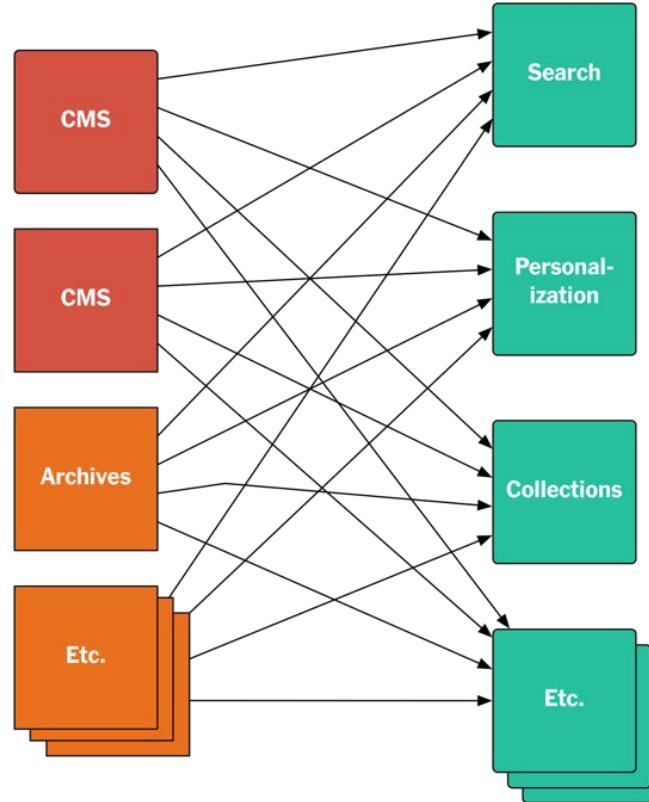
7 transactions s⁻¹
(Bitcoin network)

Log as source of Truth

Event Sourcing

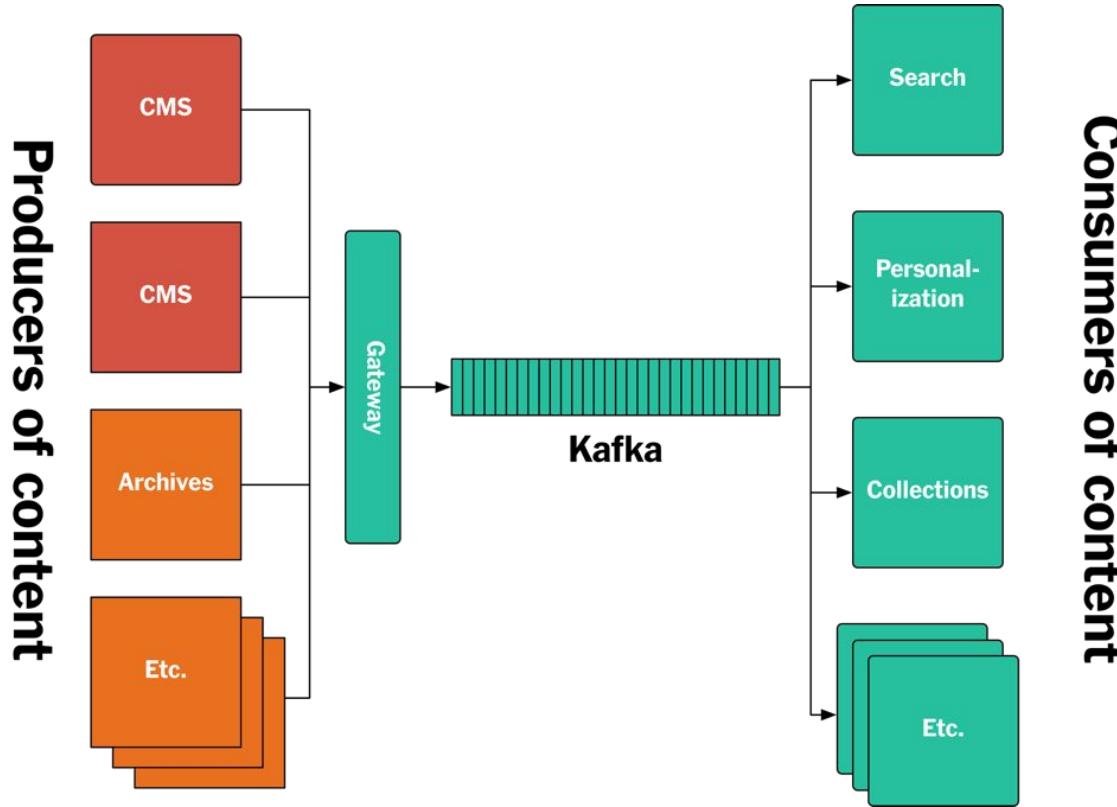
New York Times

Producers of content



Consumers of content

New York Times



New York Times Monolog

Section 1

Byline 1

Tag 1

Tag 2

Image 1

Image 2

Image 3

Section 2

Article 2

Article 1

Image 2, version 2

Tag 2, version 2

"We don't care if our production system crashes."

Valerii Vasylkov, William Hill



Full-fledged stream
processing framework

Build an app that
consumes from Kafka
directly

Full-fledged stream
processing framework



Build an app that
consumes from Kafka
directly



Data science. **Analytics**
about the business

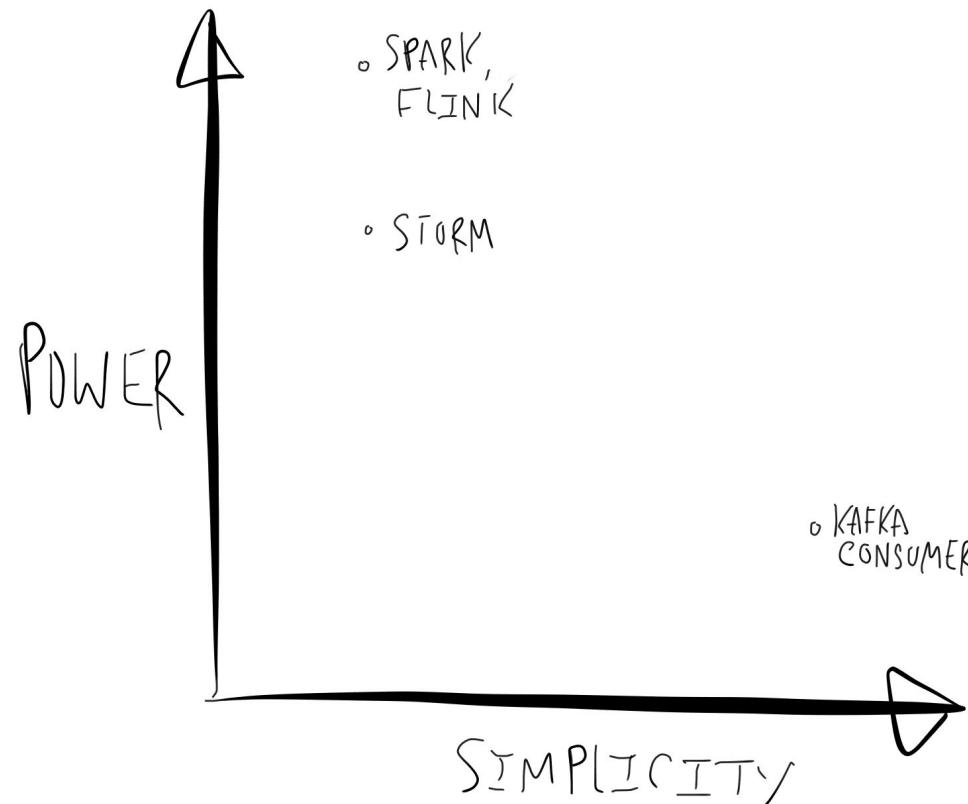
Full-fledged stream
processing framework

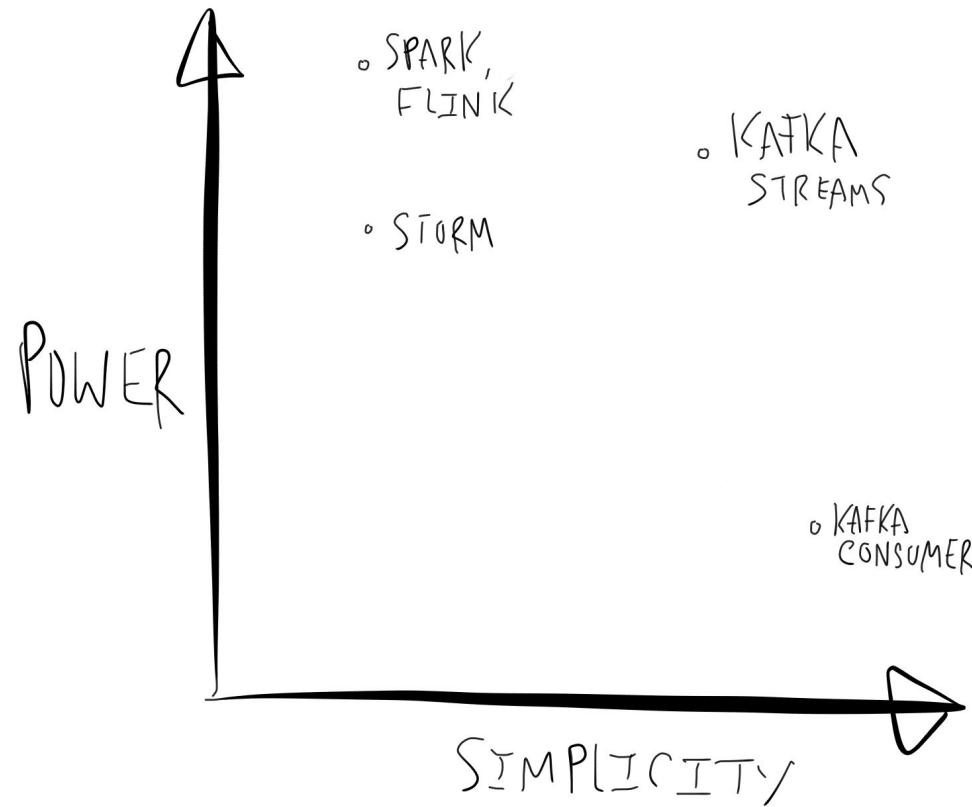


Microservice like. **Core**
business functions

Build an app that
consumes from Kafka
directly



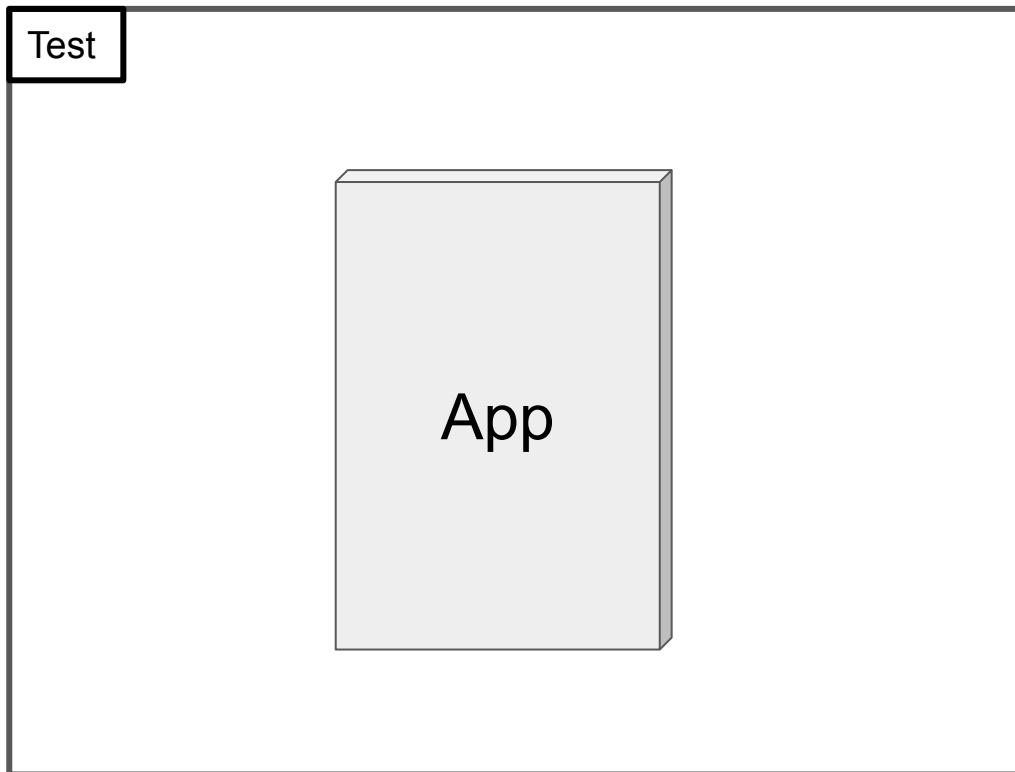




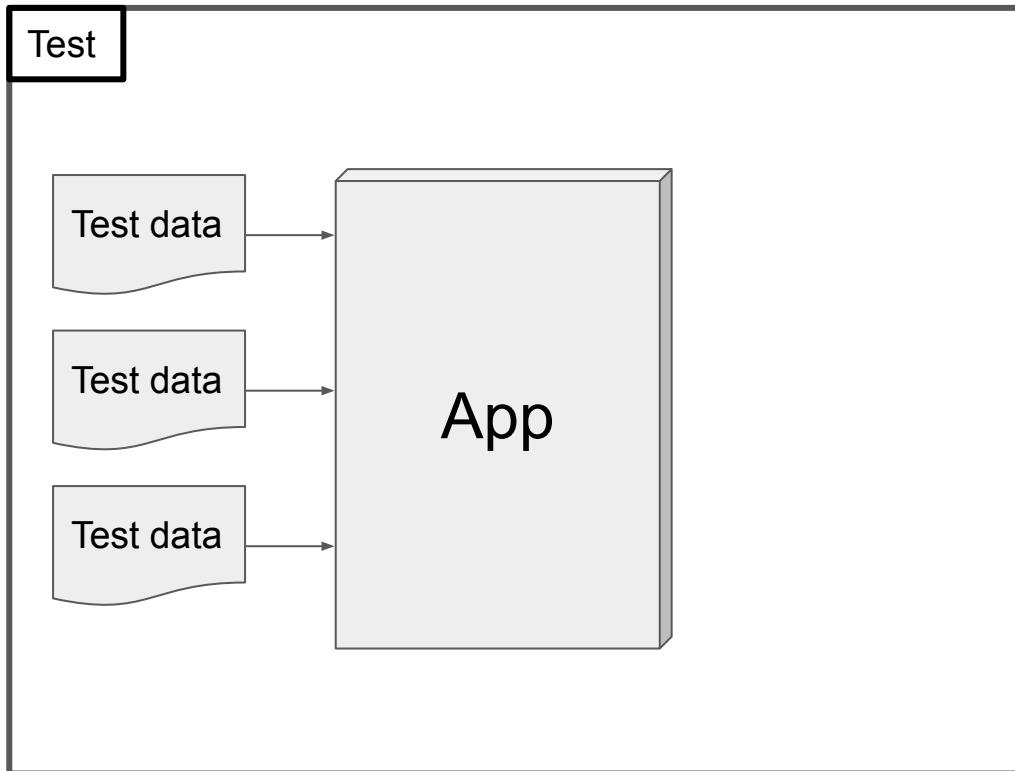
Testing Microservices

Monolithic testing

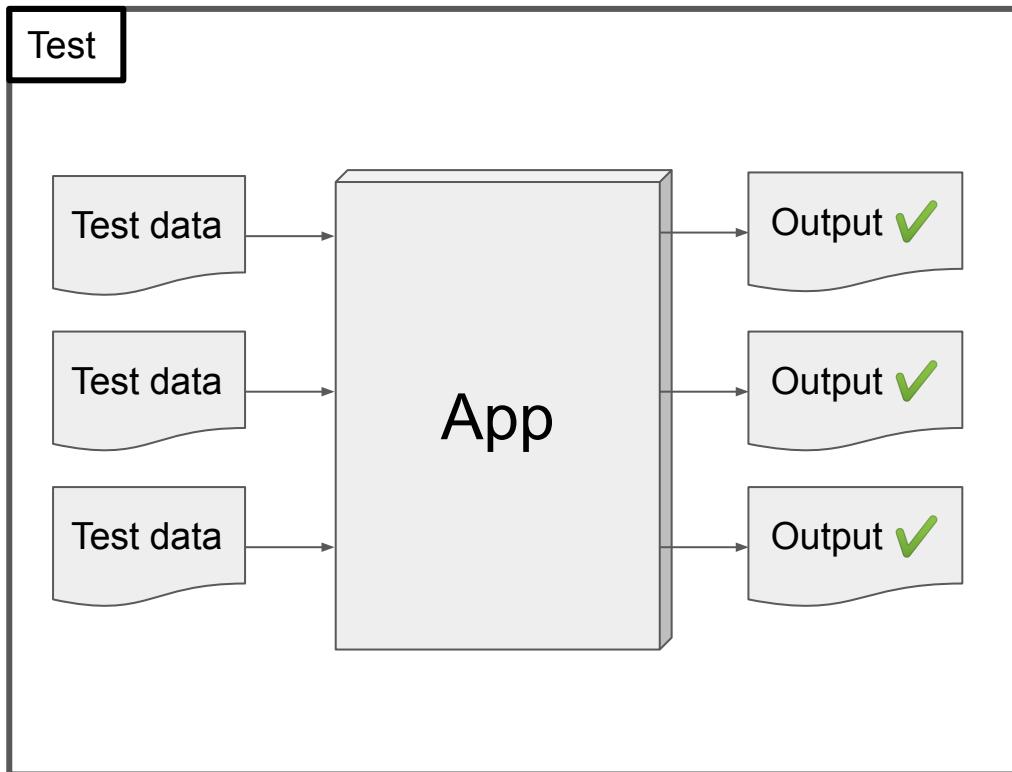
Monolithic testing



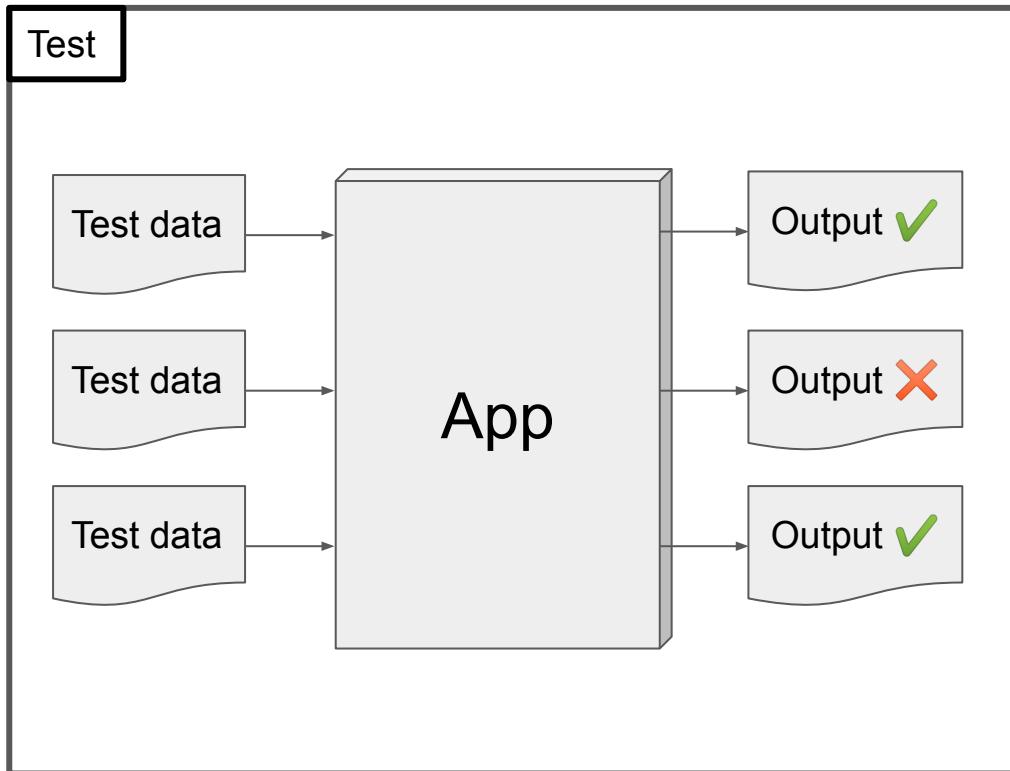
Monolithic testing



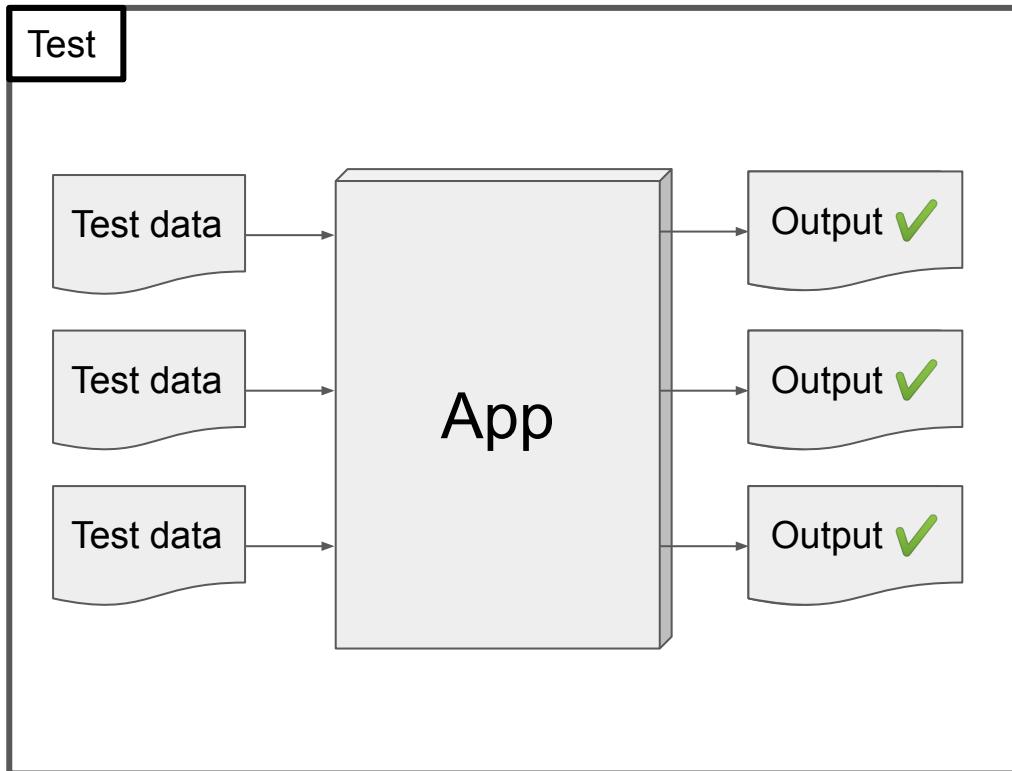
Monolithic testing



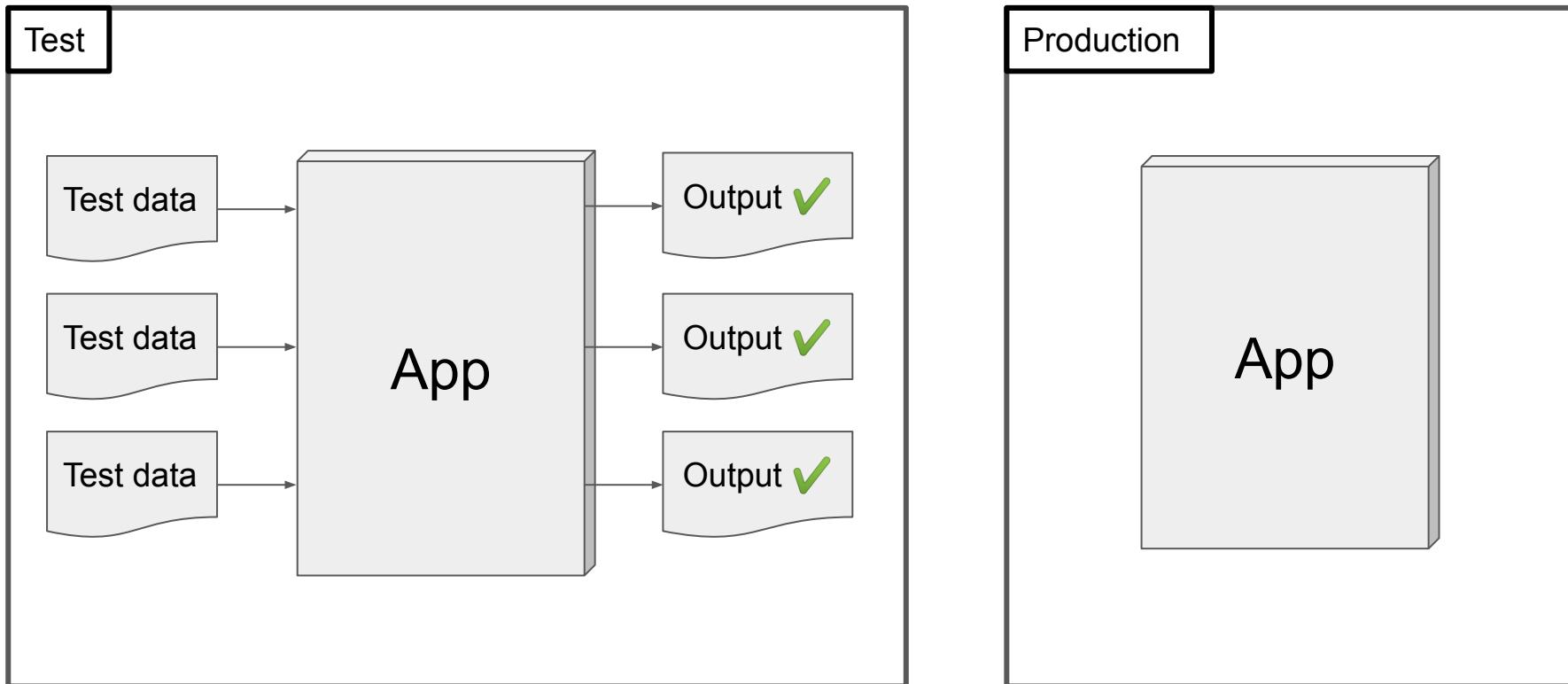
Monolithic testing



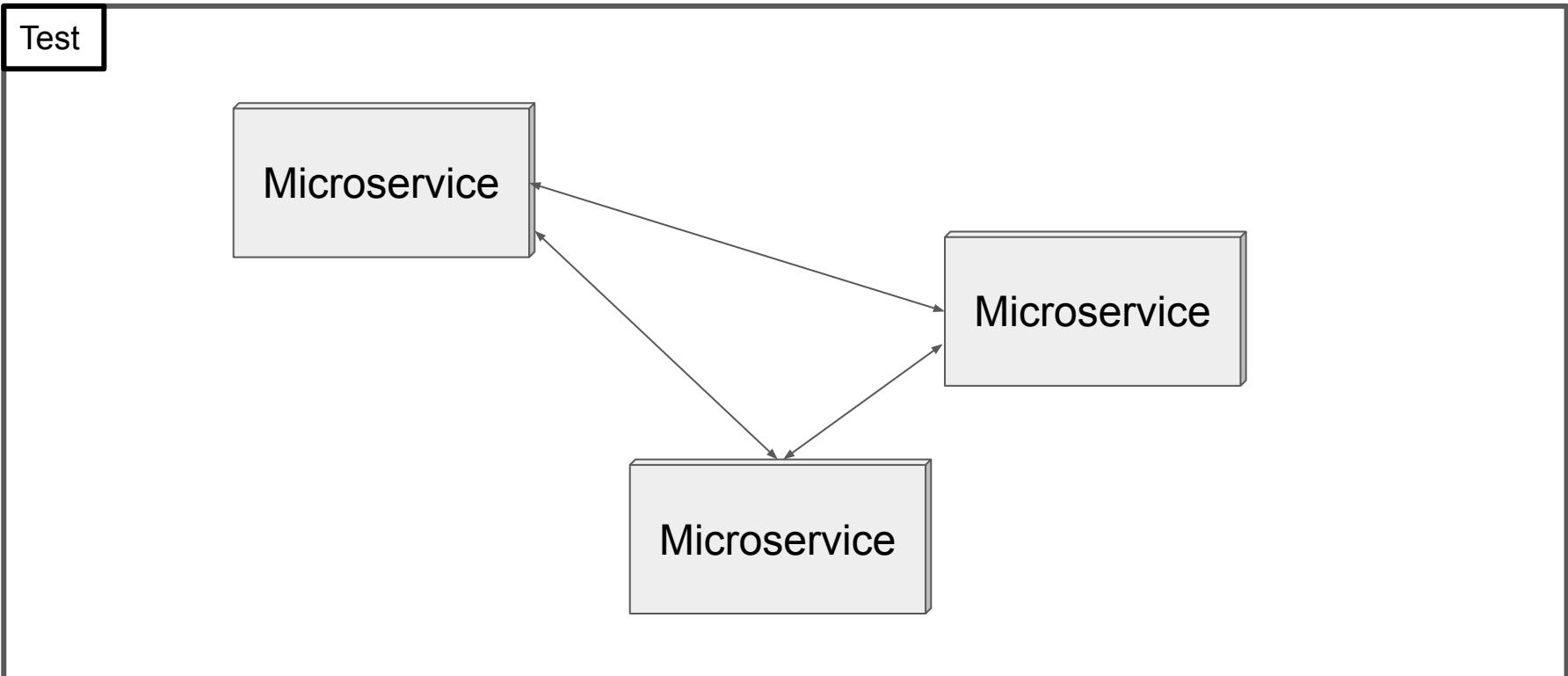
Monolithic testing



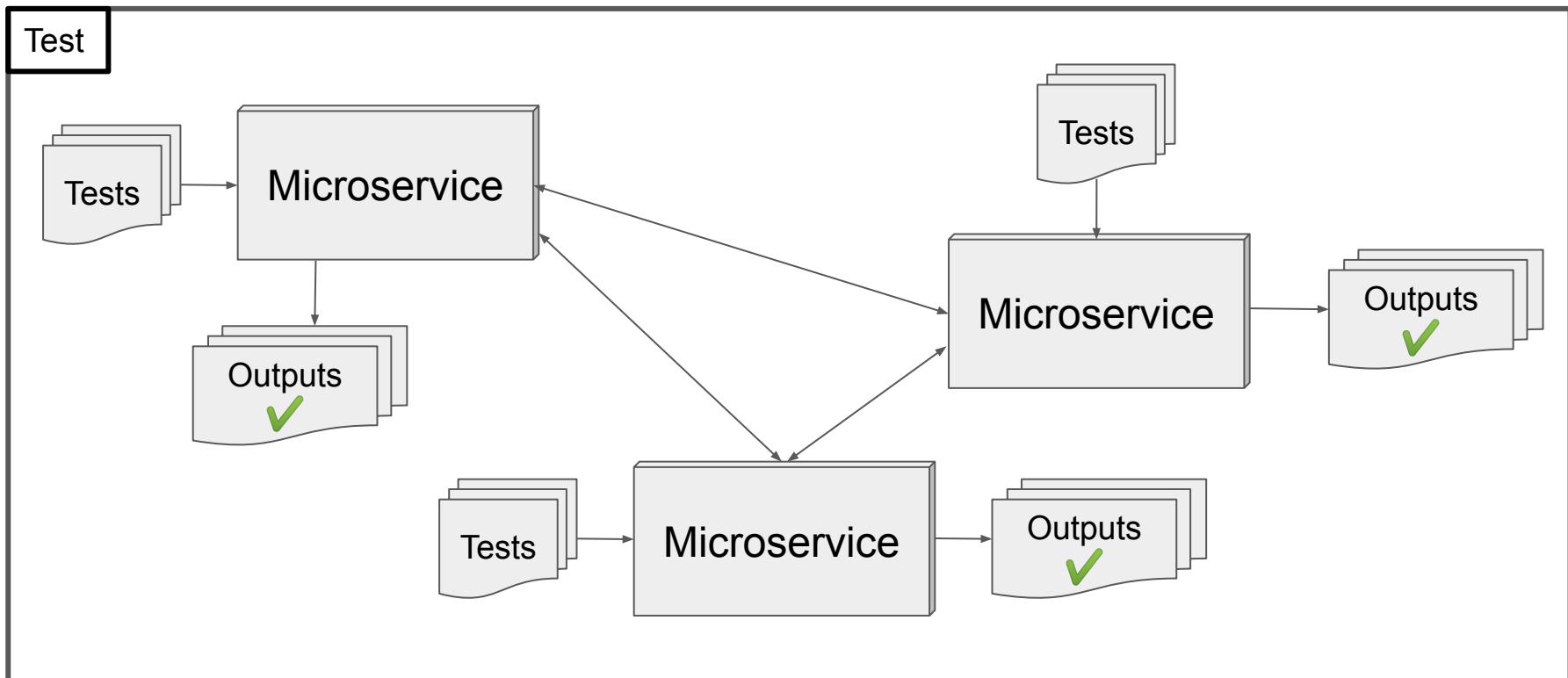
Monolithic testing



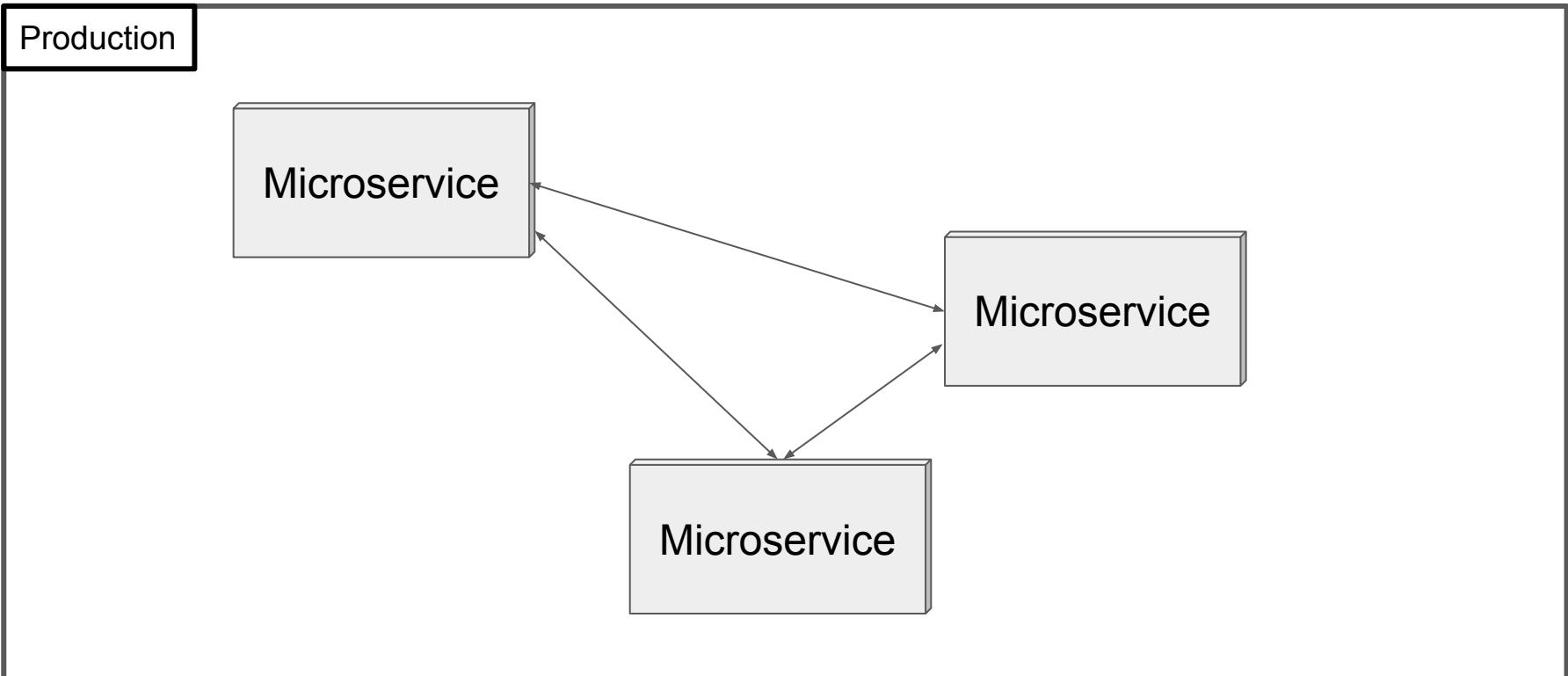
Microservice testing



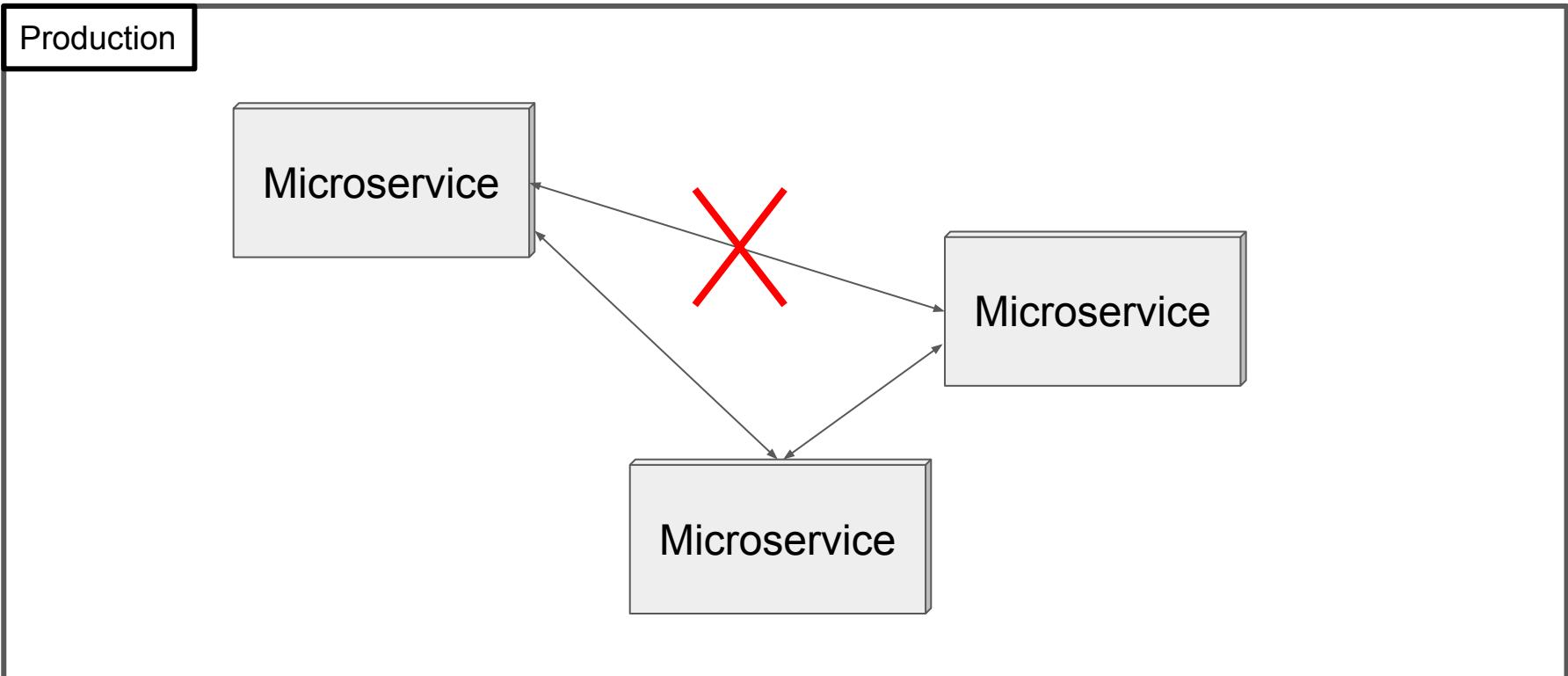
Microservice testing



Microservice testing



Microservice testing



Monitoring = Testing

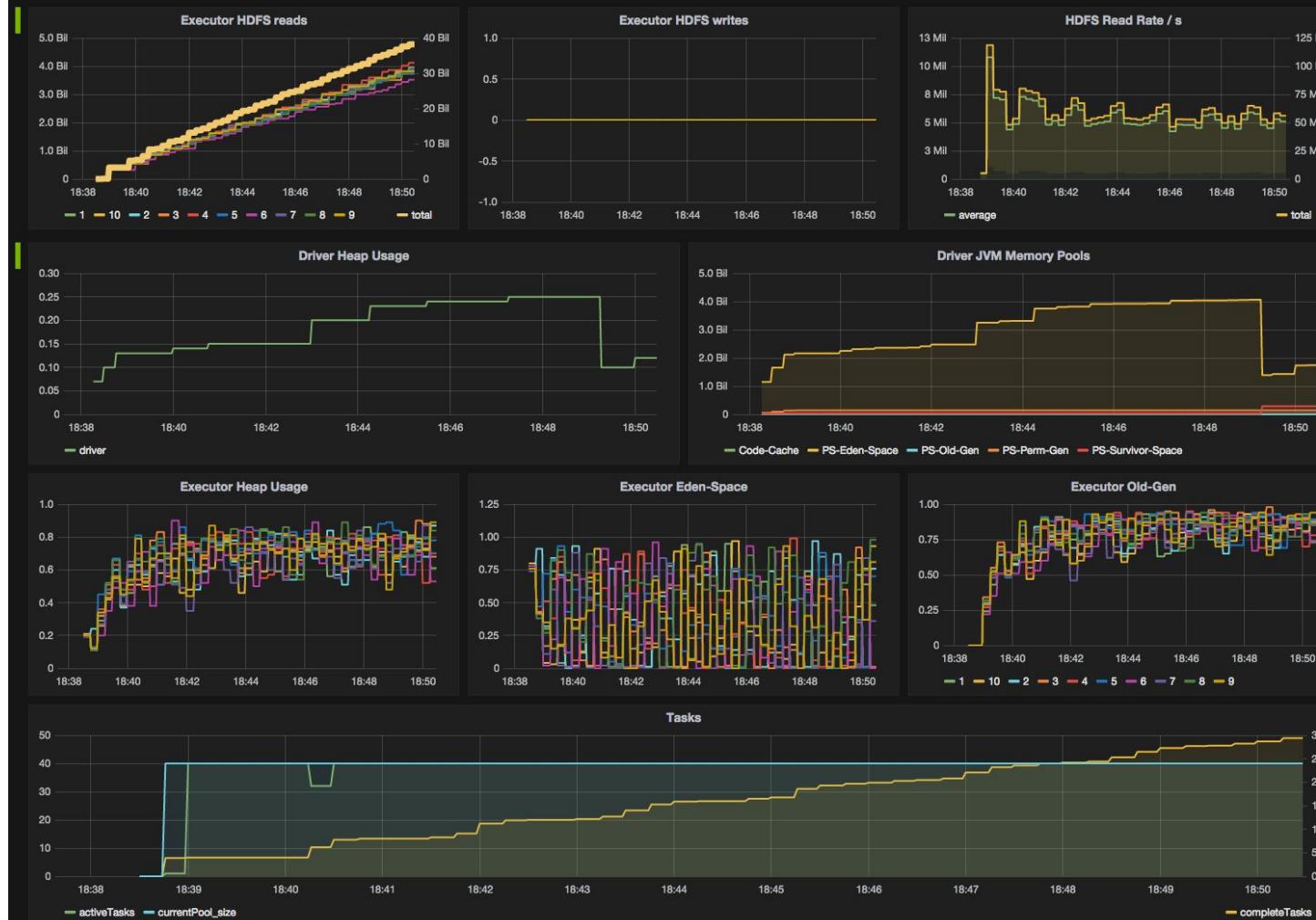


DATADOG



SENTRY

application_ID: application_1443599818521_3111 -



Observability (O11y)

“[Observability is] about being able to ask arbitrary questions about your environment without knowing ahead of time what you want it to ask”

Charity Majors

I DON'T ALWAYS TEST MY
CODE

A photograph of the character "The Most Interesting Man in the World" from the TV show "Borat". He is an older man with a full grey beard and mustache, wearing a dark pinstripe suit jacket over a white button-down shirt. He is leaning forward, resting his chin on his hand, with a bottle of Dos Equis beer on a wooden table in front of him.

BUT WHEN I DO I DO IT IN
PRODUCTION

NETFLIX



SIMIAN ARMY

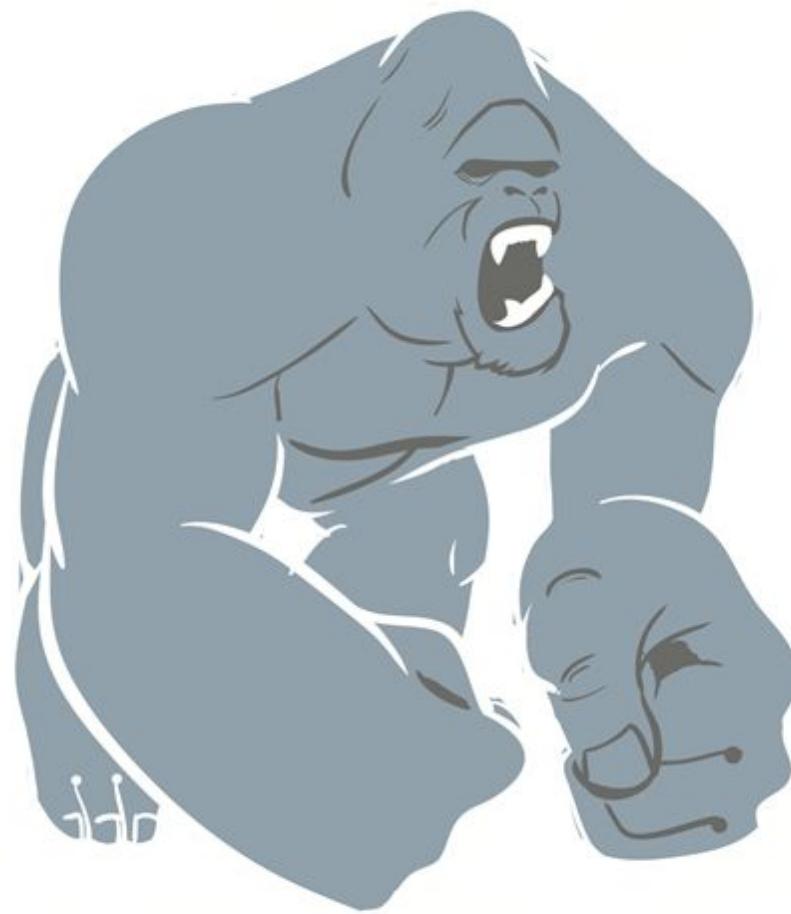


amazon
web services™

Chaos Gorilla



Chaos Kong



Conformity Monkey



Security Monkey



Janitor Monkey



Latency Monkey



“Everything fails all the time”

Werner Vogels, VP + CTO @ Amazon

Design for Failure

“It is far better to be in a constant state
of minor failure”

Richard Rodger, CEO @ nearForm

Self healing systems

Debugging Microservices

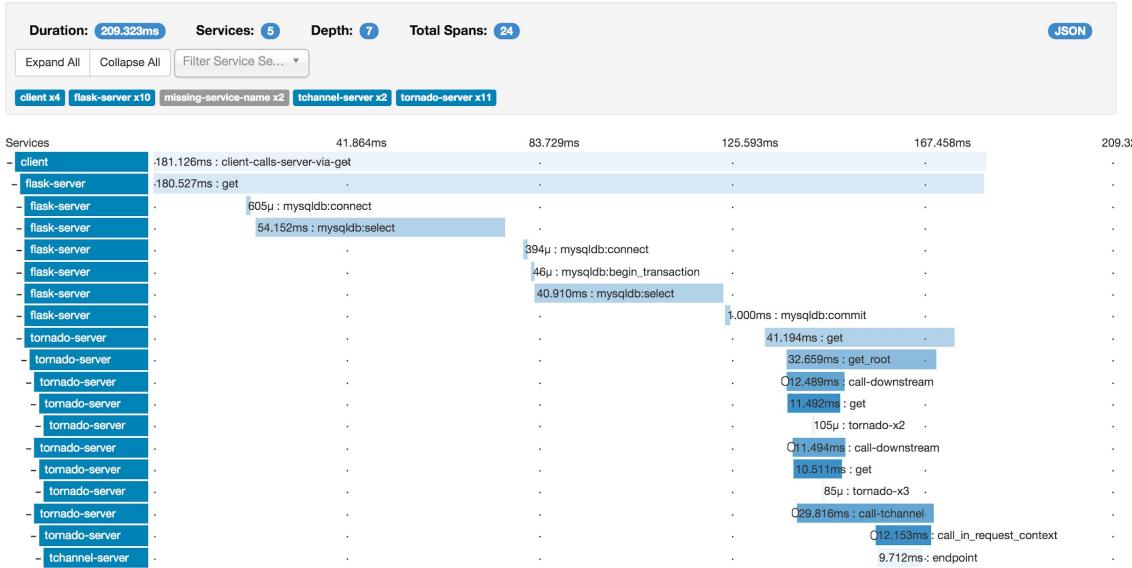
splunk®>



elastic



logstash



“How big should my Microservice be?”

Everyone

“Not all of a large system will
be well designed”

Eric Evans

Domain-Driven DESIGN

Tackling Complexity in the Heart of Software

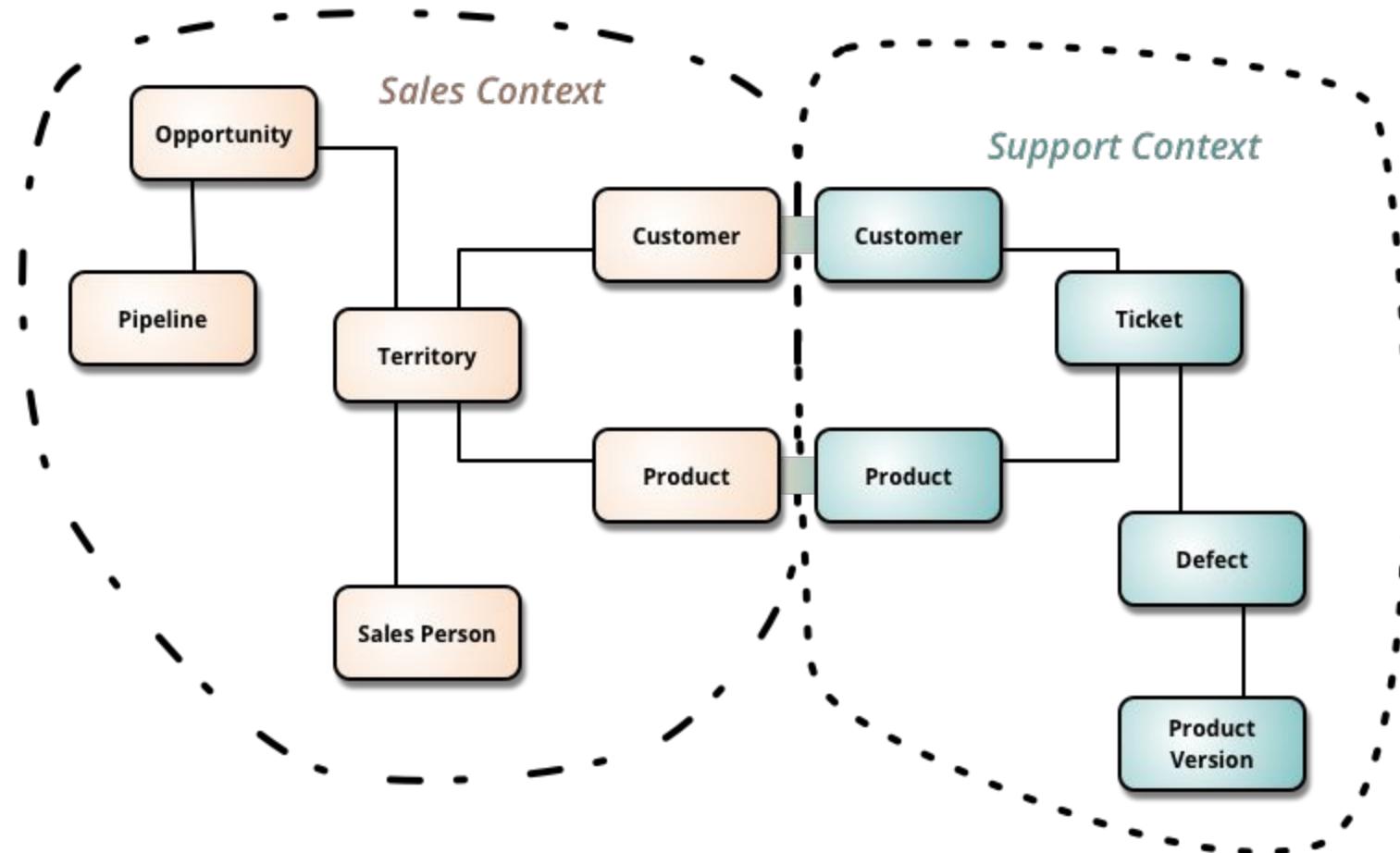


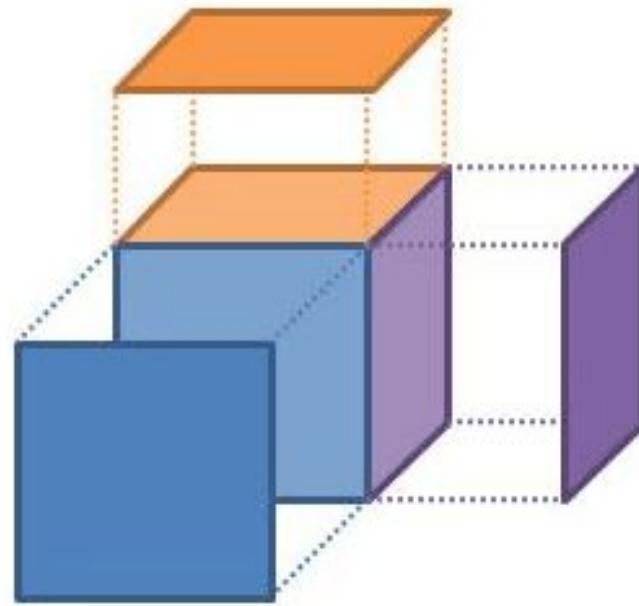
Eric Evans

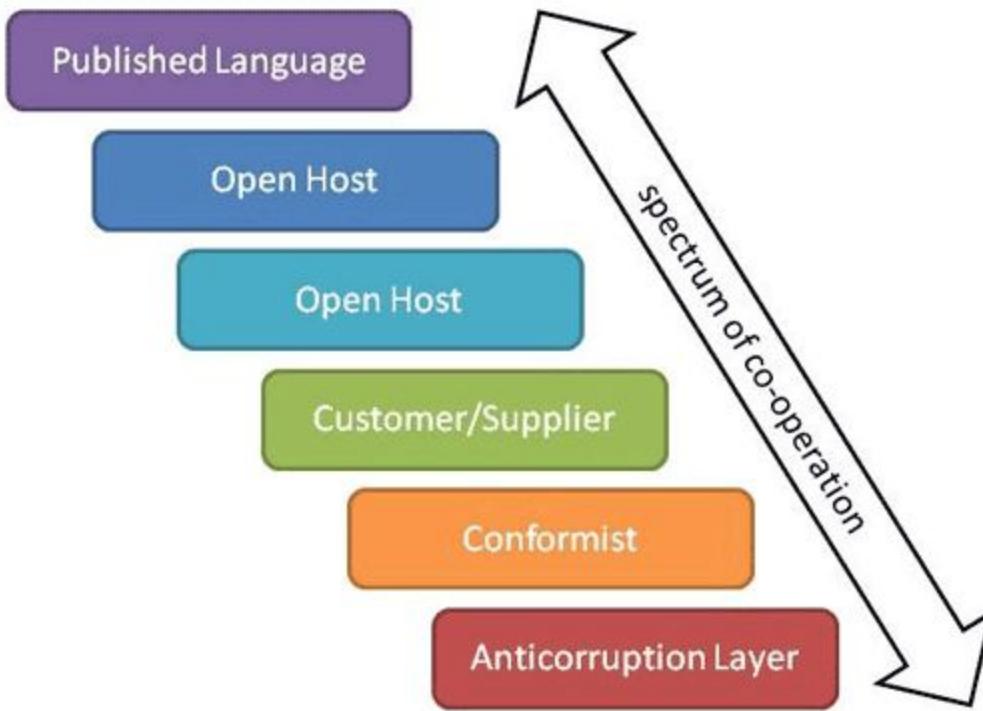
Foreword by Martin Fowler

1. Focus on the **core domain**.
2. Explore models in a creative collaboration of domain practitioners and software practitioners.
3. Speak a **ubiquitous language** within an explicitly **bounded context**.







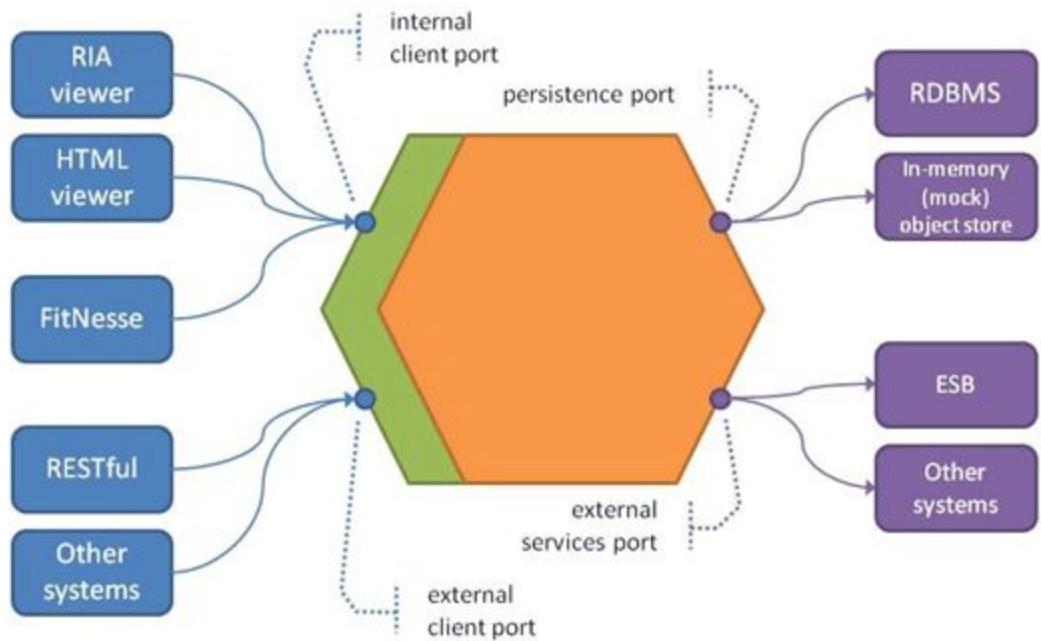


Presentation
Layer

Application
Layer

Domain
Layer

Infrastructure
Layer







“It isn't the easiest read in the software literature, but it's one of those books that amply repays a substantial investment.”

Martin Fowler

“... big enough to fit in your head”

Dan North



“Replaceable Component Architecture”

Dan North



“The first draft of anything is shit”
Ernest Hemingway

Monolith first?

Your Code as a Crime Scene

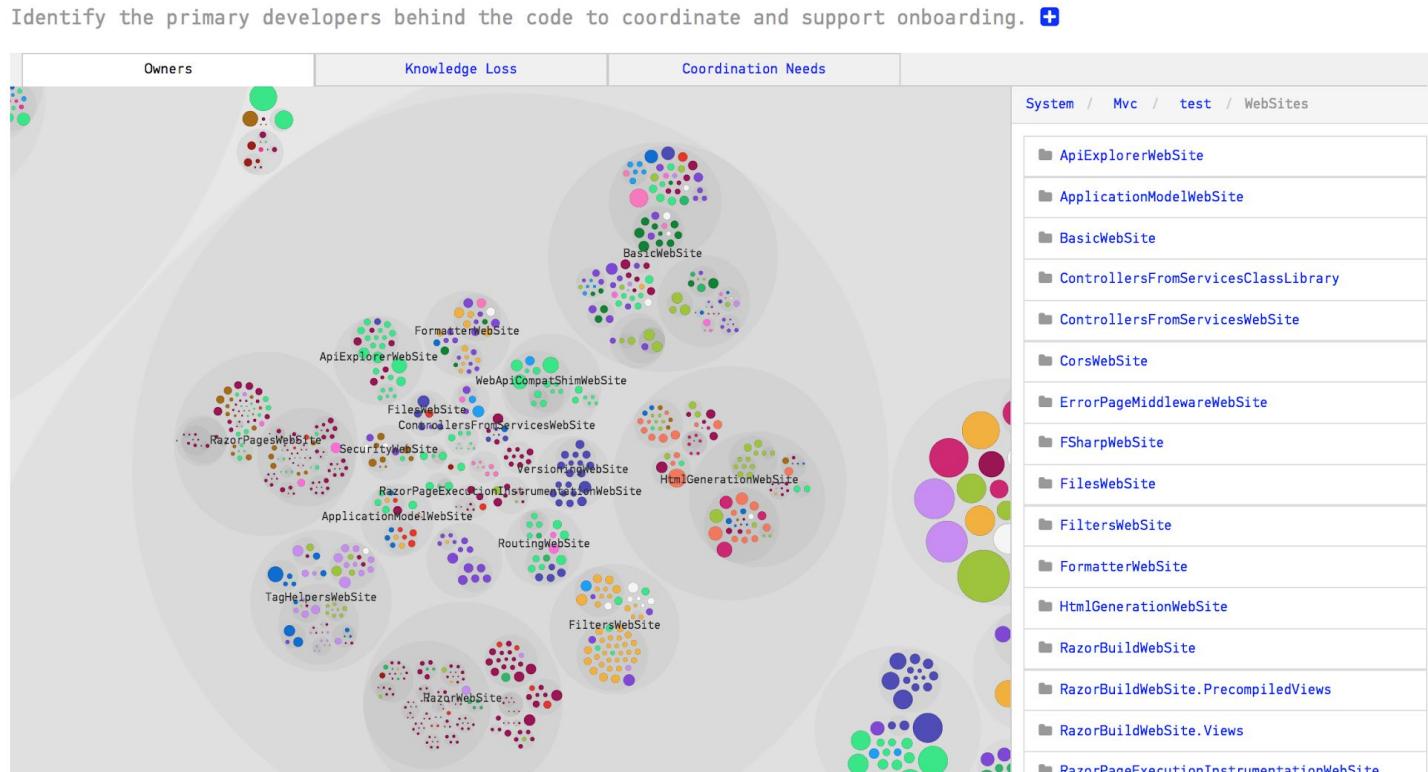
Use Forensic Techniques
to Arrest Defects, Bottlenecks, and
Bad Design in Your Programs



CØDESCENE

ANALYSIS RESULTS
ASP.NET MVC

- Dashboard
- Scope
- Technical Debt
- Architecture
- Social Analyses
 - Social Networks
 - Individuals
 - Teams
 - Modus Operandi
 - Authors
- Project Management

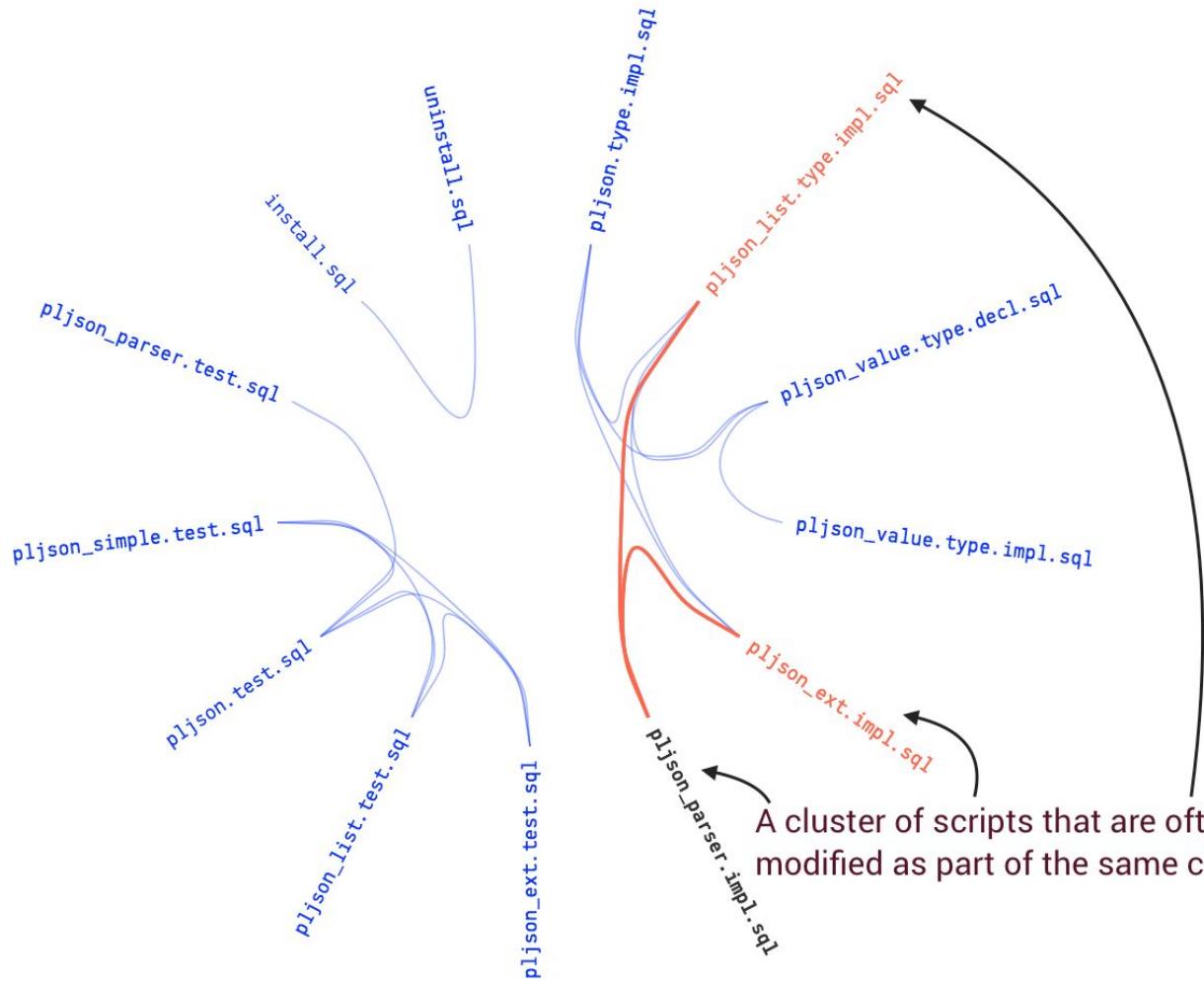


System / Mvc / test / WebSites

- ApiExplorerWebSite
- ApplicationModelWebSite
- BasicWebSite
- ControllersFromServicesClassLibrary
- ControllersFromServicesWebSite
- CorsWebSite
- ErrorPageMiddlewareWebSite
- FSharpWebSite
- FilesWebSite
- FiltersWebSite
- FormatterWebSite
- HtmlGenerationWebSite
- RazorBuildWebSite
- RazorBuildWebSite.PrecompiledViews
- RazorBuildWebSite.Views
- RazorPageExecutionInstrumentationWebSite
- RoutingWebSite
- TagHelpersWebSite
- VersioningWebSite

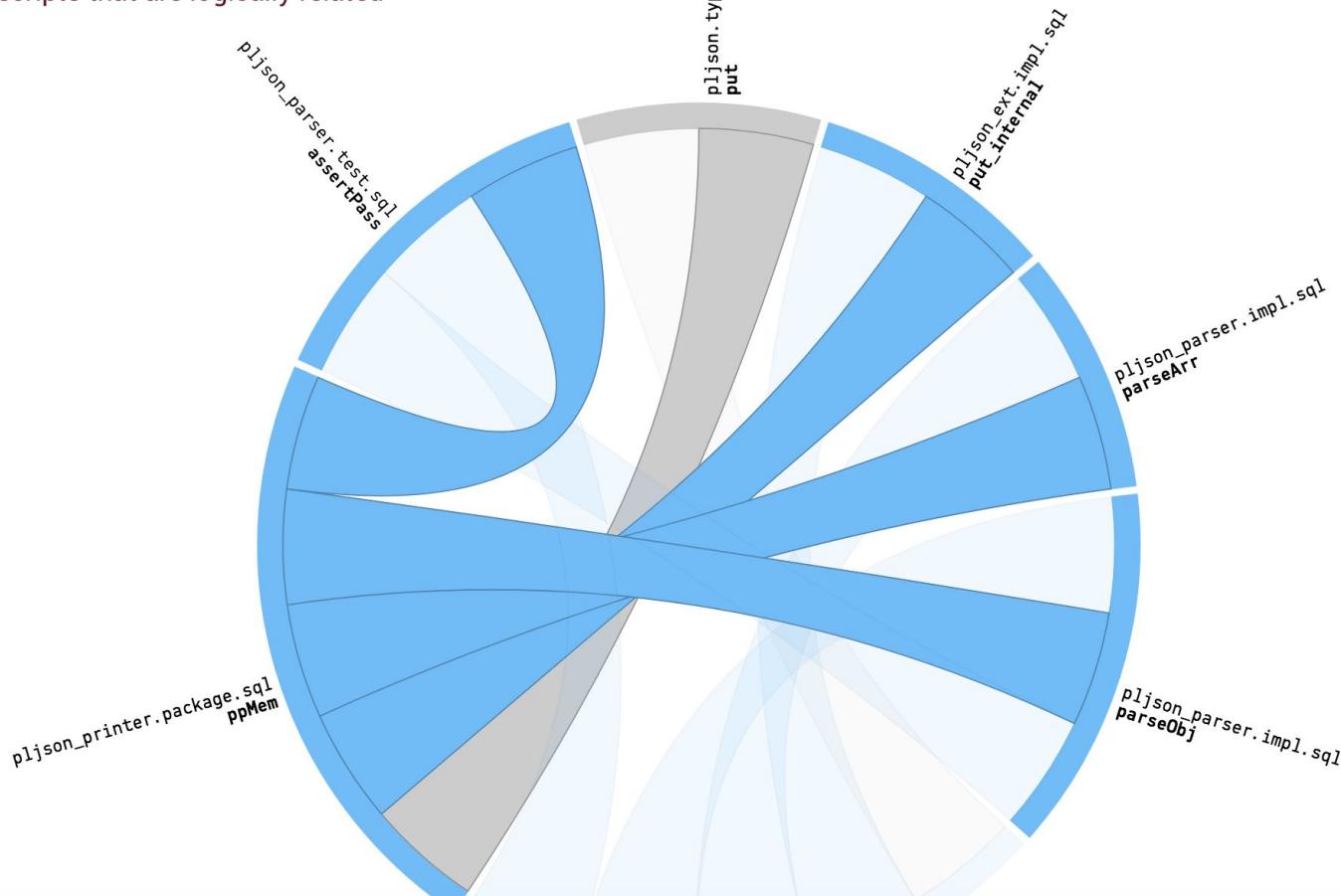
Legend mapping developer names to colors:

- Pranav Krishnamoorthy
- Ryan Nowak
- Kiran Challa
- Doug Bunting
- jacalvar
- Sornakumar Sundararajan
- Ajay Bhargav Baaskaran
- Jass Bagga
- Harsh Gupta
- Ryan Brandenburg
- N. Taylor Mullen
- Kirthi Krishnamraju
- Andrew Peters
- harshgMSFT
- Javier Calvarro Nelson
- dougbu
- damianedwards
- Youngjune Hong
- Steve Sanderson
- ryanbrandenburg



A cluster of scripts that are often modified as part of the same changes.

Identify procedures in different SQL scripts that are logically related





THE TWELVE-FACTOR APP

12factor.net

I. Codebase

One codebase tracked in revision control, many deploys

I. Codebase

One codebase tracked in revision control, many deploys

Always use version control!

I. Codebase

One codebase tracked in revision control, many deploys

Always use version control!

One app (microservice) per repo

Codebase



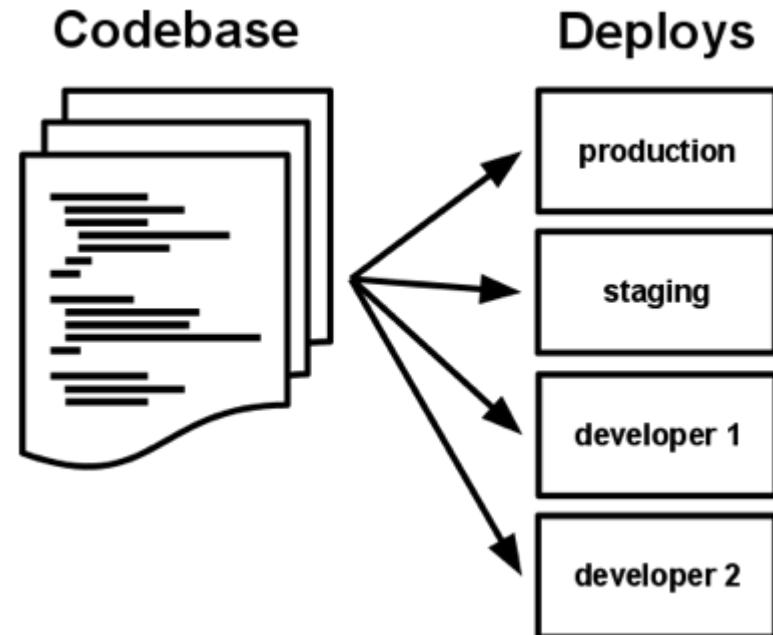
I. Codebase

One codebase tracked in revision control, many deploys

Always use version control!

One app (microservice) per repo

Multiple deploys per app (local, dev, prod, etc.)



II. Dependencies

Explicitly declare and isolate dependencies

II. Dependencies

Explicitly declare and isolate dependencies

Never assume something is pre-installed

II. Dependencies

Explicitly declare and isolate dependencies

Never assume something is pre-installed

Package managers/repos make this easy!

```
$ pip install -r requirements.txt
```

```
$ npm install
```

```
$ bundle install
```

II. Dependencies

Explicitly declare and isolate dependencies

Never assume something is pre-installed

Package managers/repos make this easy!

Isolate dependencies to prevent 'leaking'

```
$ pip install -r requirements.txt
```

```
$ npm install
```

```
$ bundle install
```

II. Dependencies

Explicitly declare and isolate dependencies

Never assume something is pre-installed

Package managers/repos make this easy!

Isolate dependencies to prevent 'leaking'

```
"dependencies": {  
  "back": "^0.1.5",  
  "ringbufferjs": "0.0.1",  
  "xtend": "^4.0.0"  
},
```

```
$ npm install
```



node_modules

II. Dependencies

Explicitly declare and isolate dependencies

Never assume something is pre-installed

Package managers/repos make this easy!

Isolate dependencies to prevent 'leaking'

(Aims to) prevent "it works on my machine"

```
"dependencies": {  
  "back": "^0.1.5",  
  "ringbufferjs": "0.0.1",  
  "xtend": "^4.0.0"  
},
```

```
$ npm install
```



node_modules

III. Config

Store config in the environment

III. Config

Store config in the environment

Config changes between deploys (dev, prod,
etc.)

III. Config

Store config in the environment

Config changes between deploys (dev, prod,
etc.)

Store config in environment not codebase

III. Config

Store config in the environment

Config changes between deploys (dev, prod, etc.)

Store config in environment not codebase



III. Config

Store config in the environment

Config changes between deploys (dev, prod, etc.)

Store config in environment not codebase

Suggest environment vars, but other options...



III. Config

Store config in the environment

Config changes between deploys (dev, prod, etc.)

Store config in environment not codebase

Suggest environment vars, but other options...

Supports multiple deploys from one codebase



III. Config

Store config in the environment

Config changes between deploys (dev, prod, etc.)

Store config in environment not codebase

Suggest environment vars, but other options...

Supports multiple deploys from one codebase

Makes Open Sourcing easier



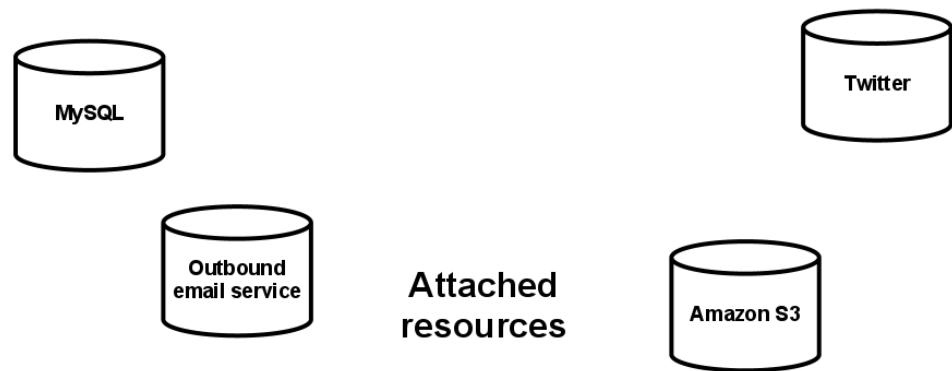
IV. Backing services

Treat backing services as attached resources

IV. Backing services

Treat backing services as attached resources

Backing services are any remote services

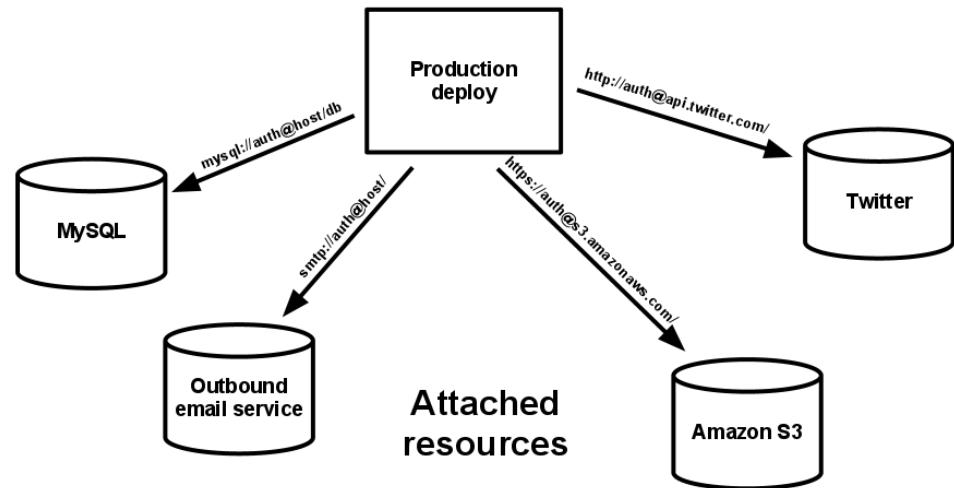


IV. Backing services

Treat backing services as attached resources

Backing services are any remote services

Loosely coupled to a deploy



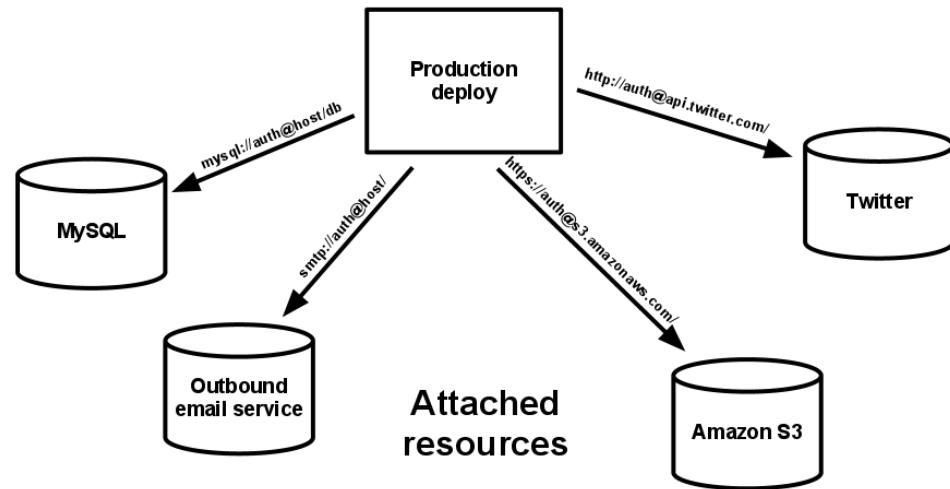
IV. Backing services

Treat backing services as attached resources

Backing services are any remote services

Loosely coupled to a deploy

Treated as local resources



IV. Backing services

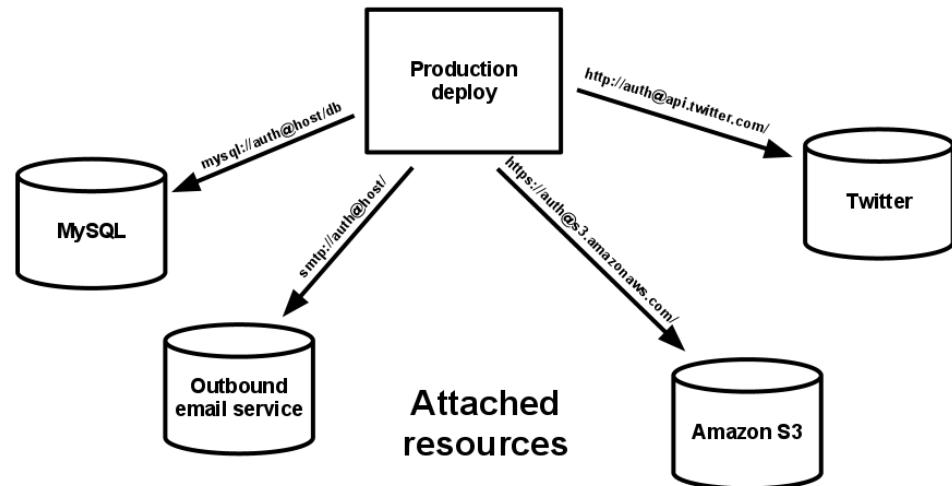
Treat backing services as attached resources

Backing services are any remote services

Loosely coupled to a deploy

Treated as local resources

Allows easy swapping out of services



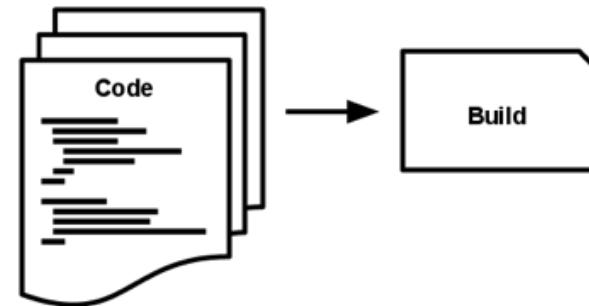
V. Build, release, run

Strictly separate build and run stages

V. Build, release, run

Strictly separate build and run stages

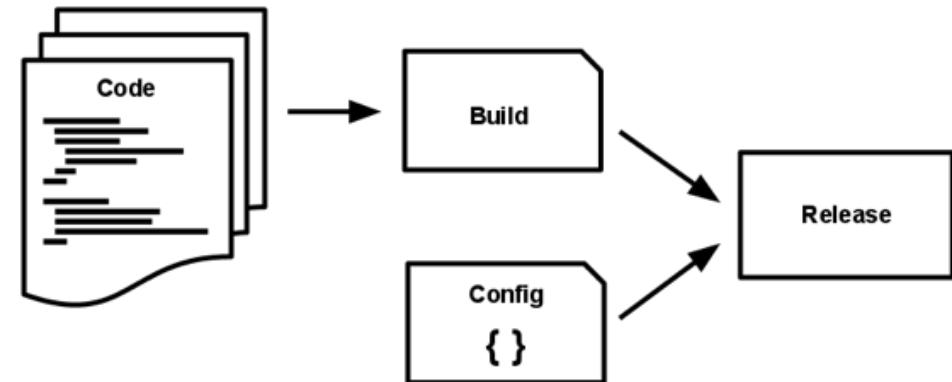
Build



V. Build, release, run

Strictly separate build and run stages

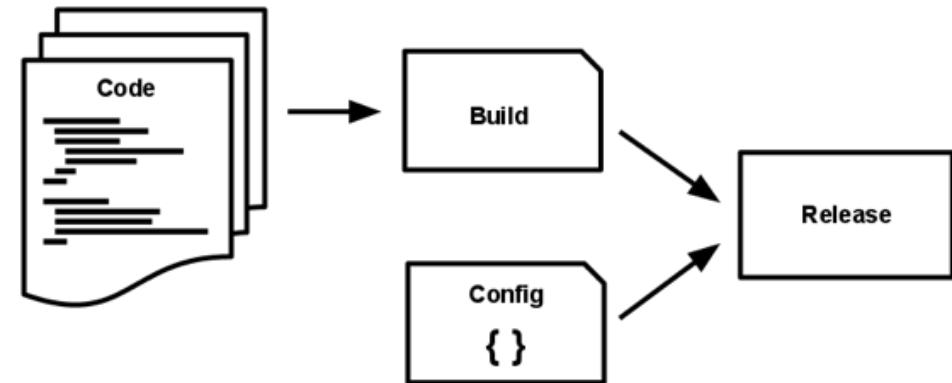
Build → Release



V. Build, release, run

Strictly separate build and run stages

Build → Release → Run

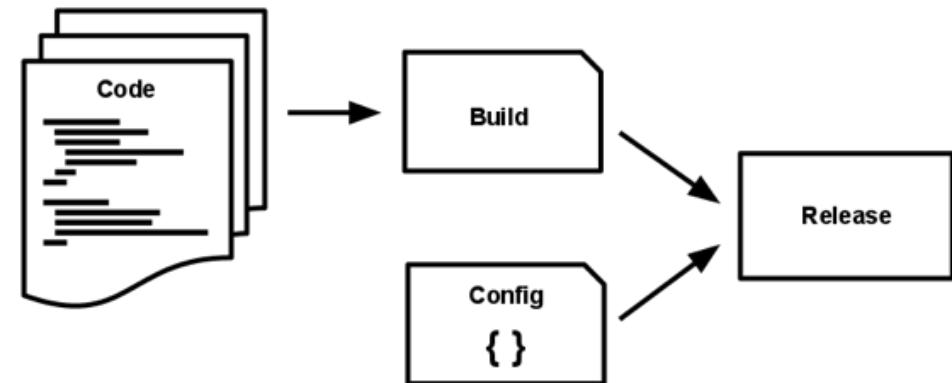


V. Build, release, run

Strictly separate build and run stages

Build → Release → Run

Strict separation between the stages



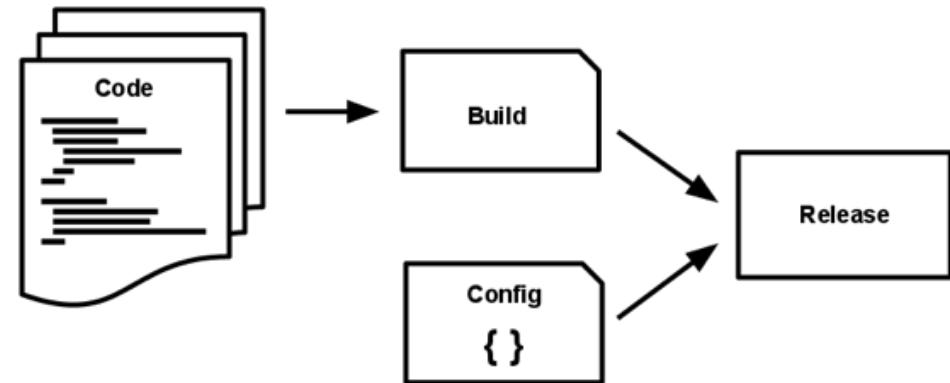
V. Build, release, run

Strictly separate build and run stages

Build → Release → Run

Strict separation between the stages

One way (no tweaking the live server!)



VI. Processes

Execute the app as one or more stateless processes

VI. Processes

Execute the app as one or more stateless processes

Apps run as processes (duh)

VI. Processes

Execute the app as one or more stateless processes

Apps run as processes (duh)

No state: every node is independent

VI. Processes

Execute the app as one or more stateless processes

Apps run as processes (duh)

No state: every node is independent

Caching is fine, but not for future use

VI. Processes

Execute the app as one or more stateless processes

Apps run as processes (duh)

No state: every node is independent

Caching is fine, but not for future use

State must be stored in backing services

VII. Port binding

Export services via port binding

VII. Port binding

Export services via port binding

Expose endpoints

`http://localhost:5000/`

`test.mosquitto.org:1883`

`redis://redisbox:6379/0`

VII. Port binding

Export services via port binding

Expose endpoints

No runtime injection or native APIs

`http://localhost:5000/`

`test.mosquitto.org:1883`

`redis://redisbox:6379/0`

VII. Port binding

Export services via port binding

Expose endpoints

No runtime injection or native APIs

`http://localhost:5000/`

HTTP or other protocols make things easy

`test.mosquitto.org:1883`

`redis://redisbox:6379/0`

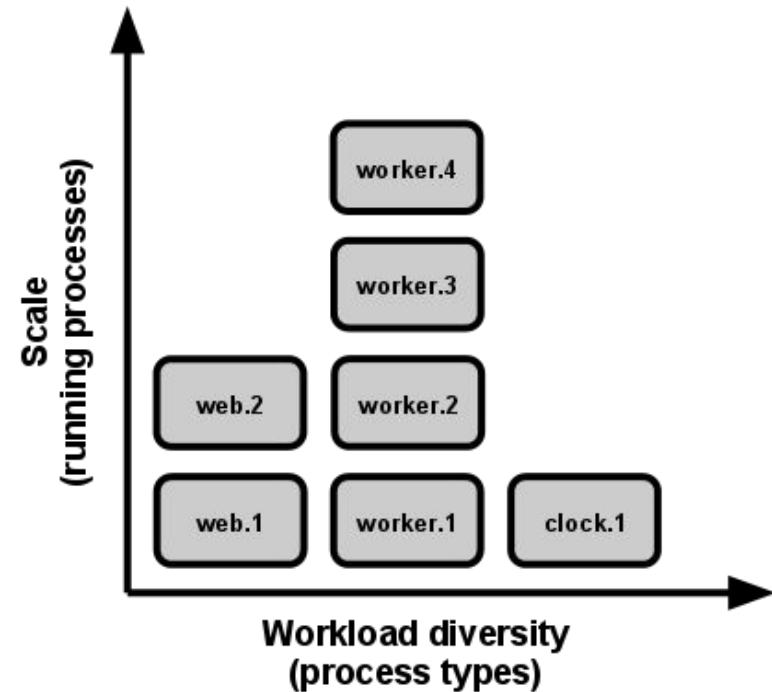
VIII. Concurrency

Scale out via the process model

VIII. Concurrency

Scale out via the process model

Processes are the unit of scaling

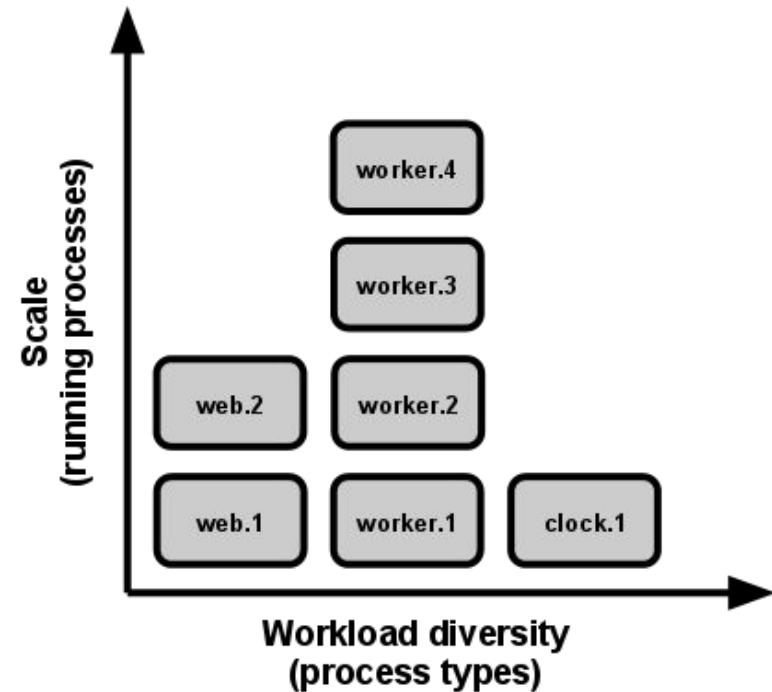


VIII. Concurrency

Scale out via the process model

Processes are the unit of scaling

Don't daemonize!



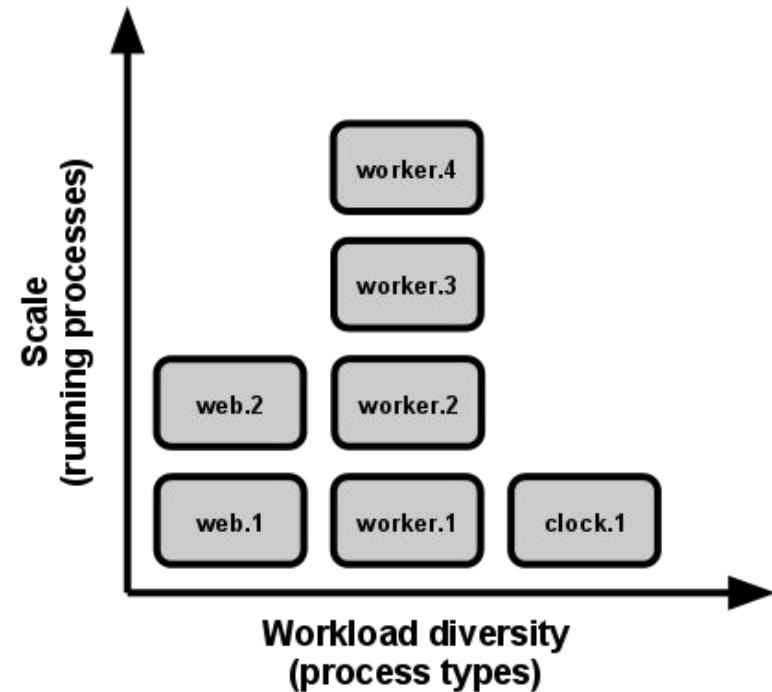
VIII. Concurrency

Scale out via the process model

Processes are the unit of scaling

Don't daemonize!

Take advantage of OS resource scaling



IX. Disposability

Maximize robustness with fast startup and graceful shutdown

IX. Disposability

Maximize robustness with fast startup and graceful shutdown

Can be started or stopped quickly (Cattle not pets)

IX. Disposability

Maximize robustness with fast startup and graceful shutdown

Can be started or stopped quickly (Cattle not pets)

Quick startup allows easy scaling up

Graceful shutdown allows easy scaling down

IX. Disposability

Maximize robustness with fast startup and graceful shutdown

Can be started or stopped quickly (Cattle not pets)

Quick startup allows easy scaling up

Graceful shutdown allows easy scaling down

Easier if you scale with independent processes...

X. Dev/prod parity

Keep development, staging, and production as similar as possible

X. Dev/prod parity

Keep development, staging, and production as similar as possible

Keep time between environment deploys low

X. Dev/prod parity

Keep development, staging, and production as similar as possible

Keep time between environment deploys low

Devs also handle deployment and maintenance
(DevOps)

Keep tooling between environments as similar as possible

X. Dev/prod parity

Keep development, staging, and production as similar as possible

Keep time between environment deploys low

Devs also handle deployment and maintenance
(DevOps)

Keep tooling between environments as similar as possible

If you can, try one environment...

XI. Logs

Treat logs as event streams

XI. Logs

Treat logs as event streams

Don't mess around writing to files, just use `stdout`

XI. Logs

Treat logs as event streams

Don't mess around writing to files, just use `stdout`

In dev, you can watch `stdout`

In production, it can be routed to other stores (Splunk
etc.)

XI. Logs

Treat logs as event streams

Don't mess around writing to files, just use `stdout`

In dev, you can watch `stdout`

In production, it can be routed to other stores (Splunk etc.)

Keeps aggregated logs time ordered (like Kafka)



XII. Admin processes

Run admin/management tasks as one-off processes

XII. Admin processes

Run admin/management tasks as one-off processes

One-off admin tasks (DB migrations etc.) should be run from production environment

XII. Admin processes

Run admin/management tasks as one-off processes

One-off admin tasks (DB migrations etc.) should be run from production environment

Don't run scripts from local machine or directly on the DB

XII. Admin processes

Run admin/management tasks as one-off processes

One-off admin tasks (DB migrations etc.) should be run from production environment

Don't run scripts from local machine or directly on the DB

Check in scripts to app repos to keep things in sync

XII. Admin processes

Run admin/management tasks as one-off processes

One-off admin tasks (DB migrations etc.) should be run from production environment

Don't run scripts from local machine or directly on the DB

Check in scripts to app repos to keep things in sync

ssh into production box and run the task

XII. Admin processes

Run admin/management tasks as one-off processes

One-off admin tasks (DB migrations etc.) should be run from production environment

Don't run scripts from local machine or directly on the DB

Check in scripts to app repos to keep things in sync

ssh into production box and run the task

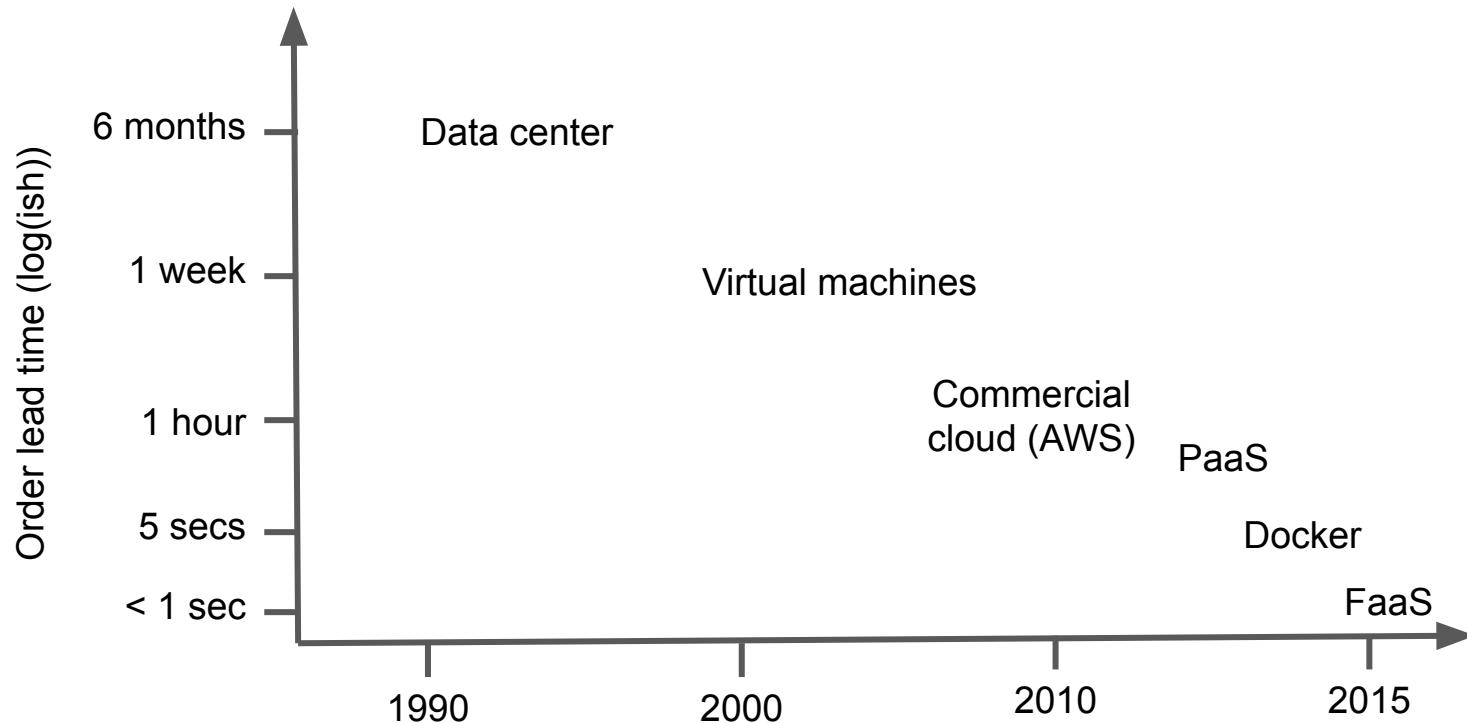
...or don't. Immutable Infrastructure!

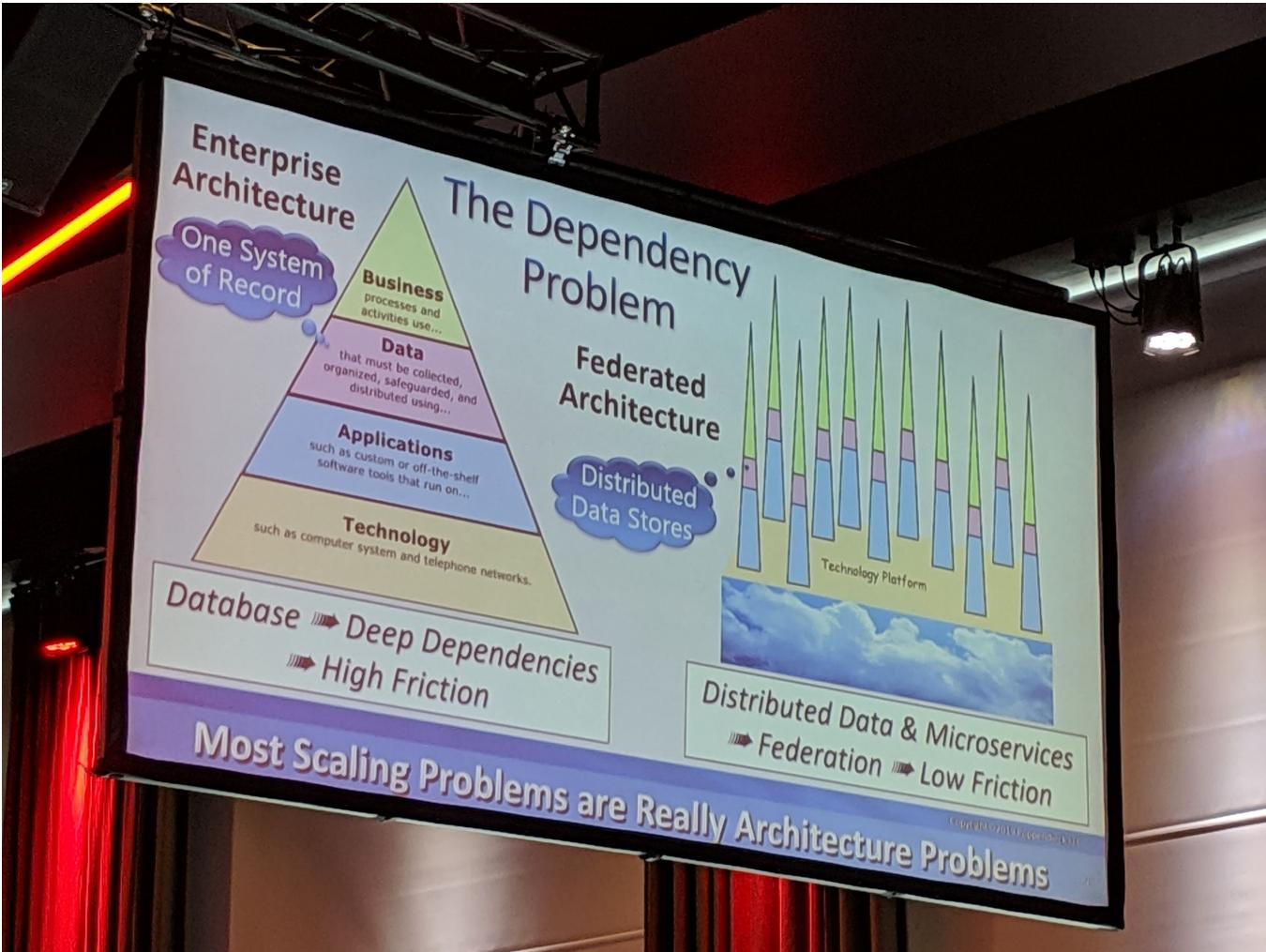


CHEF™



ANSIBLE





Microservices or distributed systems?

Microservices or **modern** systems?

The future of Microservices

Saying something so people feel better about giving you money

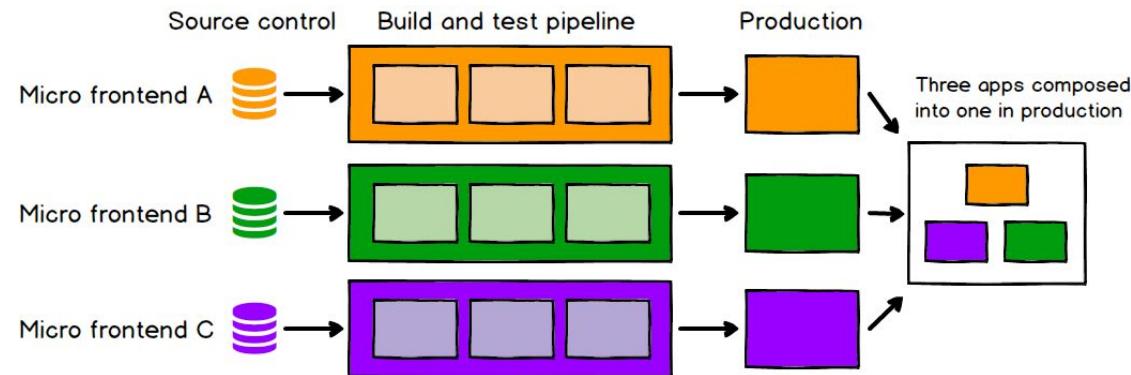


Making

Arbitrary Forecasts

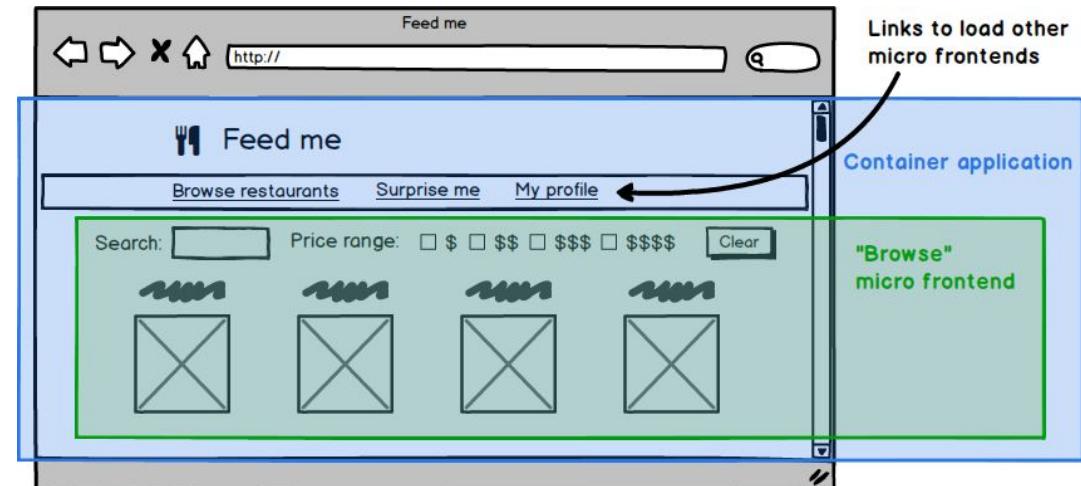
Micro Frontends

- Microservices, but for the frontend
- That's it



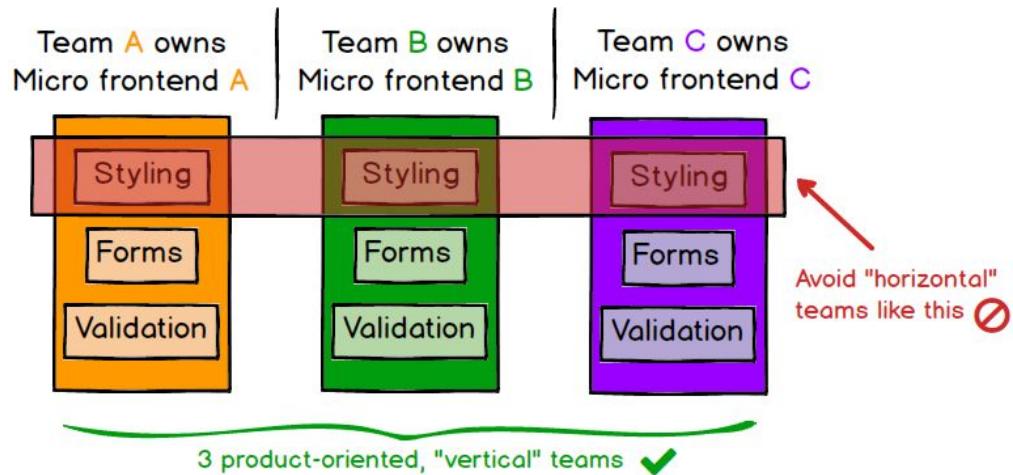
Micro Frontends

- Microservices, but for the frontend
- That's it
- Integration patterns

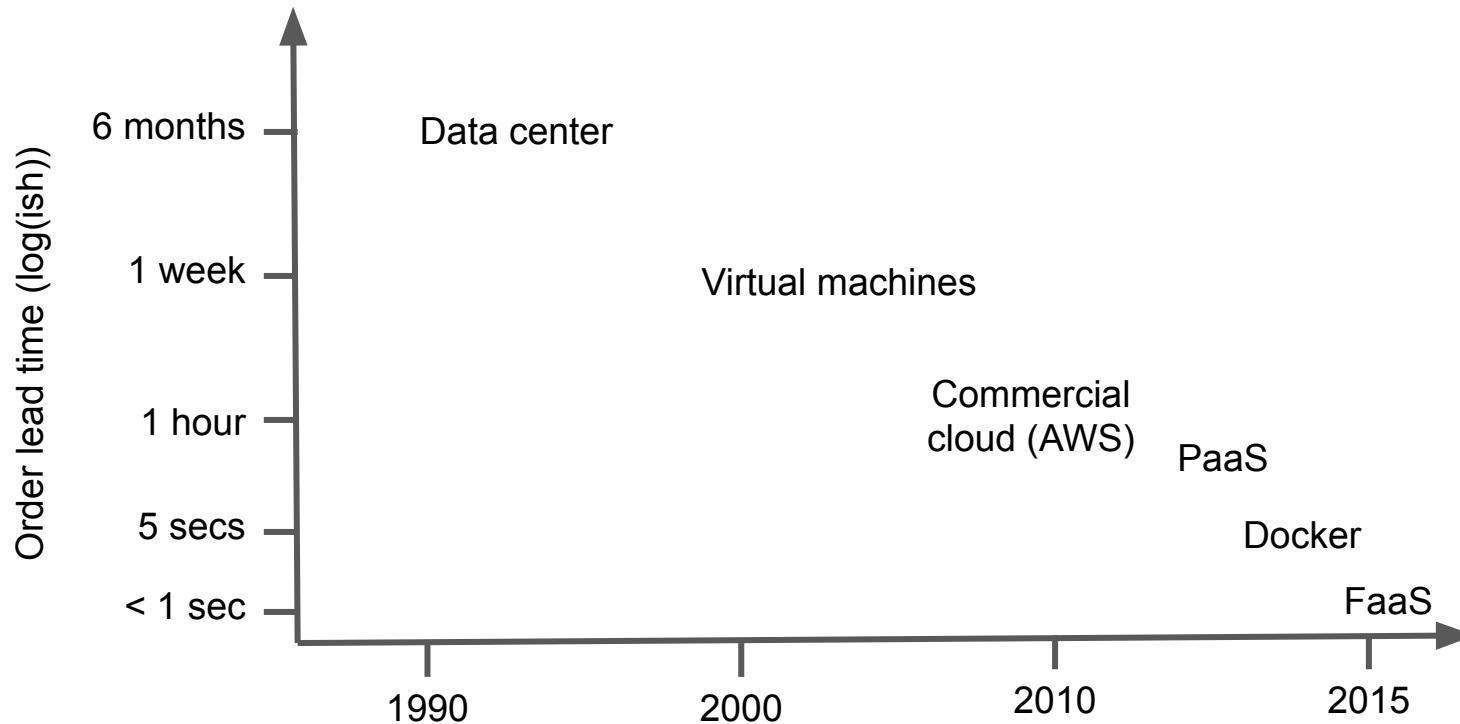


Micro Frontends

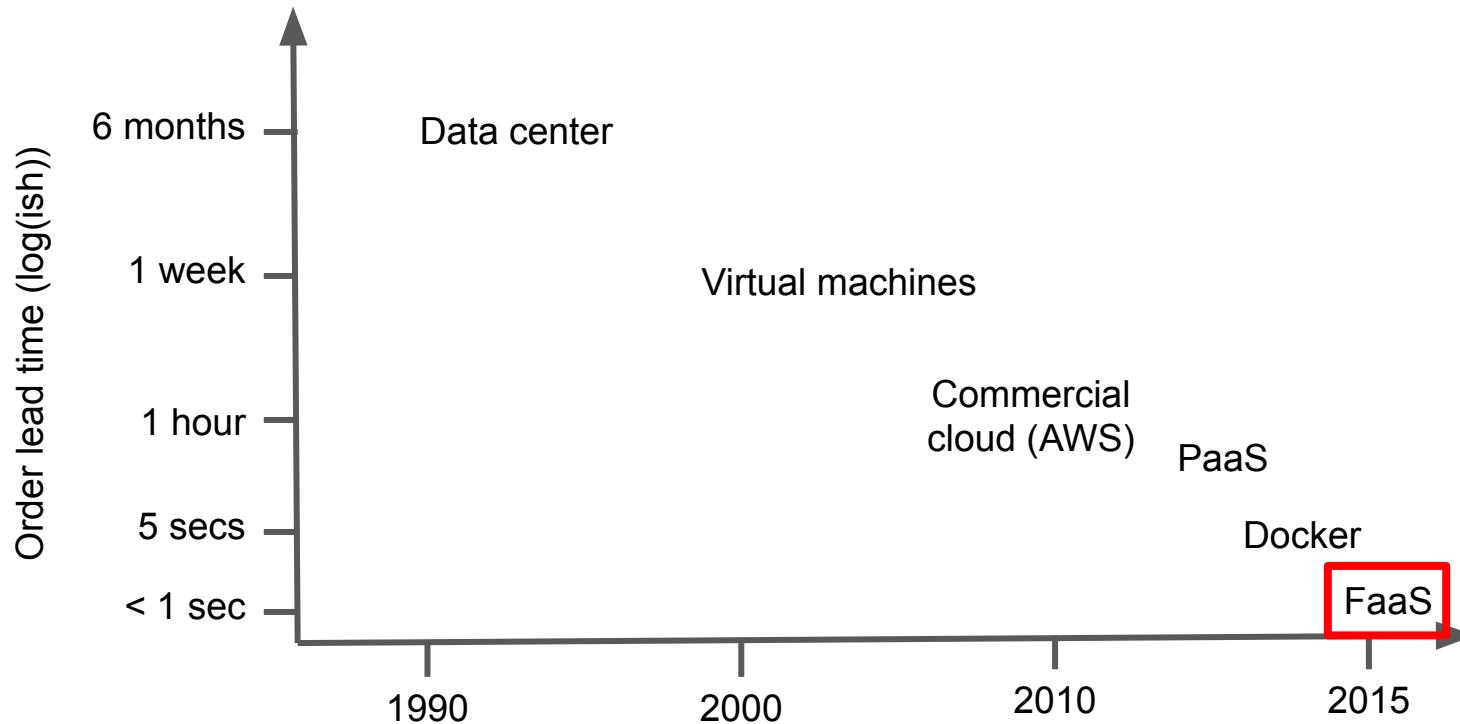
- Microservices, but for the frontend
- That's it
- Integration patterns



Hardware lead times



Hardware lead times



Serverless

'Serverless' (not really)

Function as a Service (FaaS)

No persistence, no upkeep costs

200 ms startup time (Node.js)



Amazon Lambda

What is the role of a manager?

Managers

Managers

Provide guidance

Managers

Provide guidance

Keep things 'safe'

Managers

Provide guidance

Keep things 'safe'

Organise the team

Managers

Provide guidance

Keep things 'safe'

Organise the team

Enable team to work

Managers

Provide guidance

Keep things 'safe'

Organise the team

Enable team to work

Microservices

Managers

Provide guidance

Keep things 'safe'

Organise the team

Enable team to work

Microservices

Explore new tech

Managers

Provide guidance

Keep things 'safe'

Organise the team

Enable team to work

Microservices

Explore new tech

Experiment

Managers

Provide guidance

Keep things 'safe'

Organise the team

Enable team to work

Microservices

Explore new tech

Experiment

Enables self-organisation

Managers

Provide guidance

Keep things 'safe'

Organise the team

Enable team to work

Microservices

Explore new tech

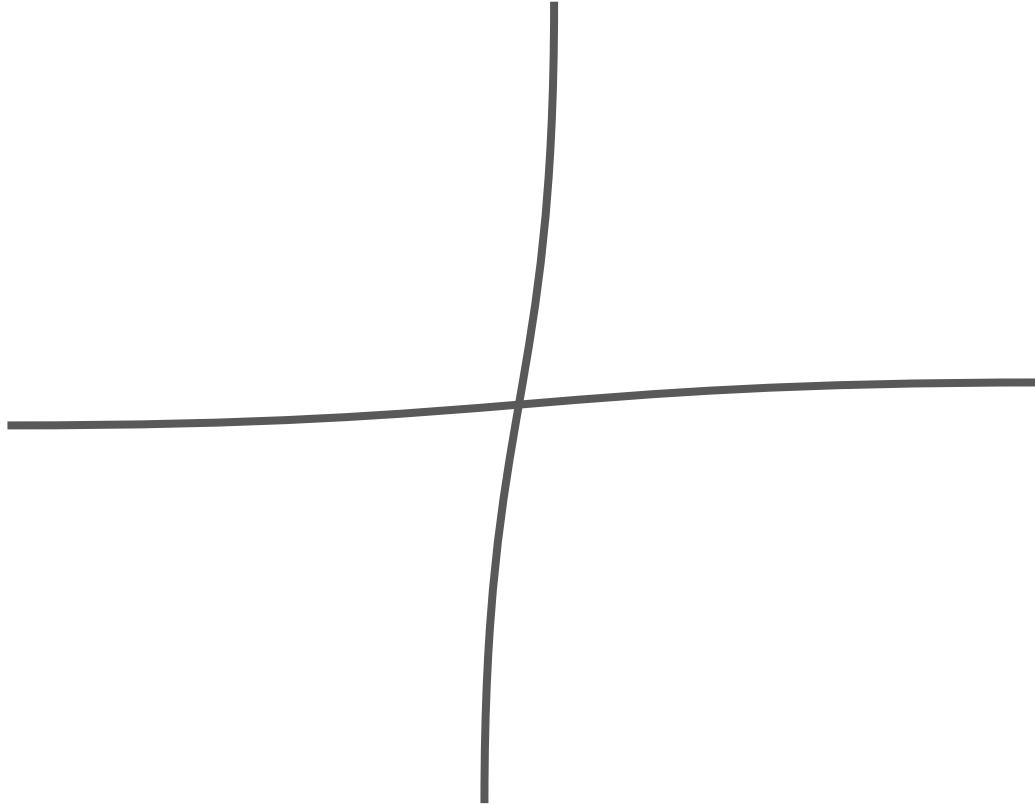
Experiment

Enables self-organisation

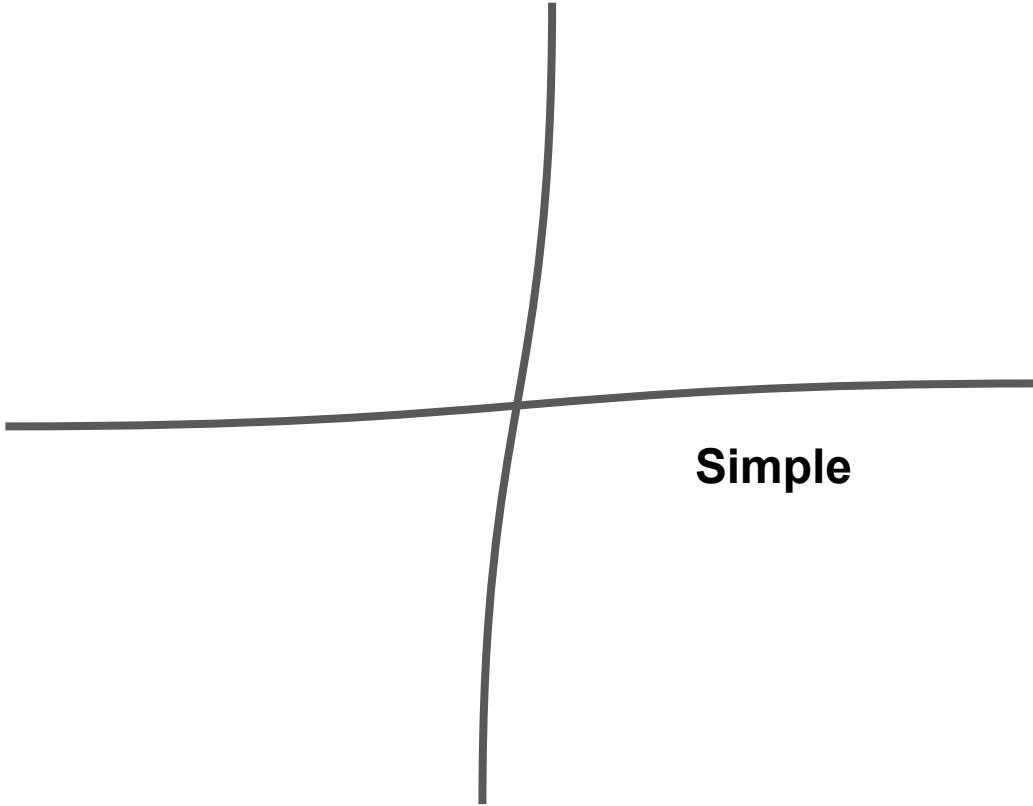
Tech empowers team

Cynefin Framework

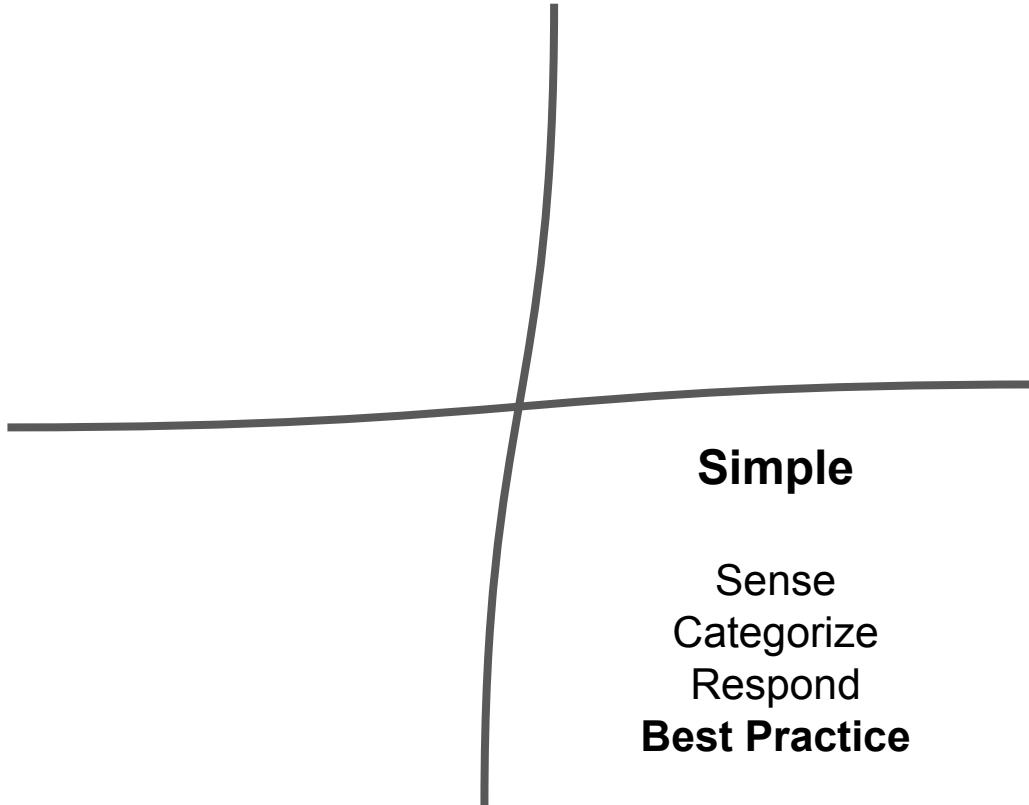
Cynefin Framework



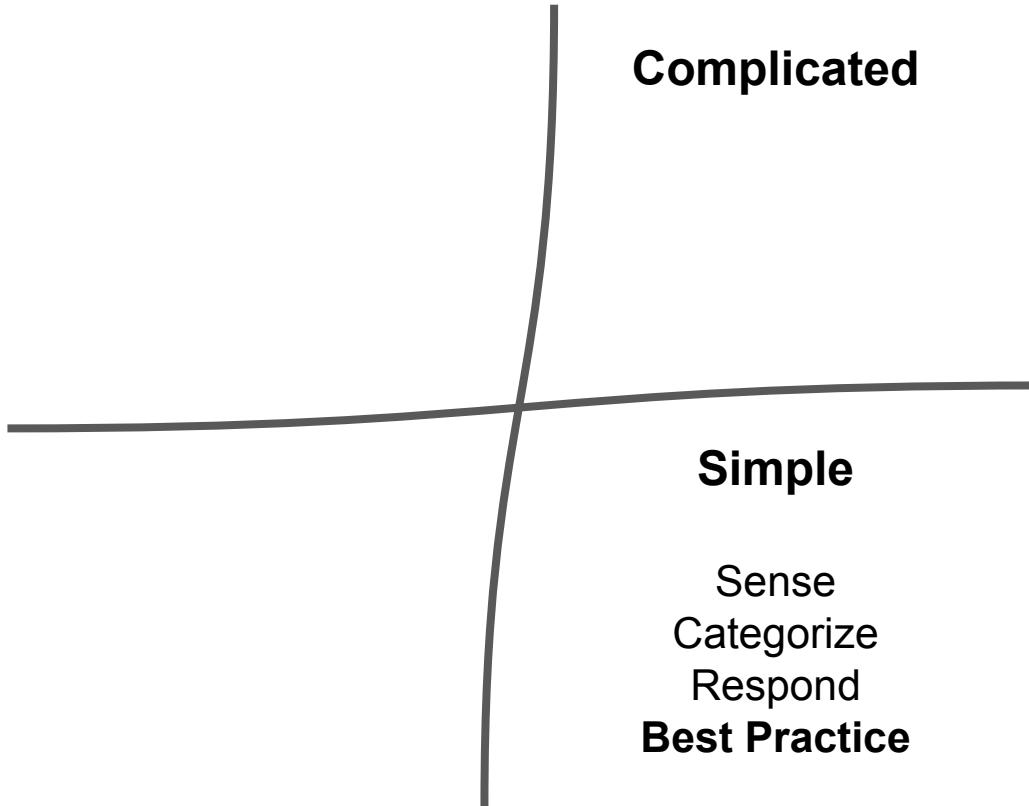
Cynefin Framework



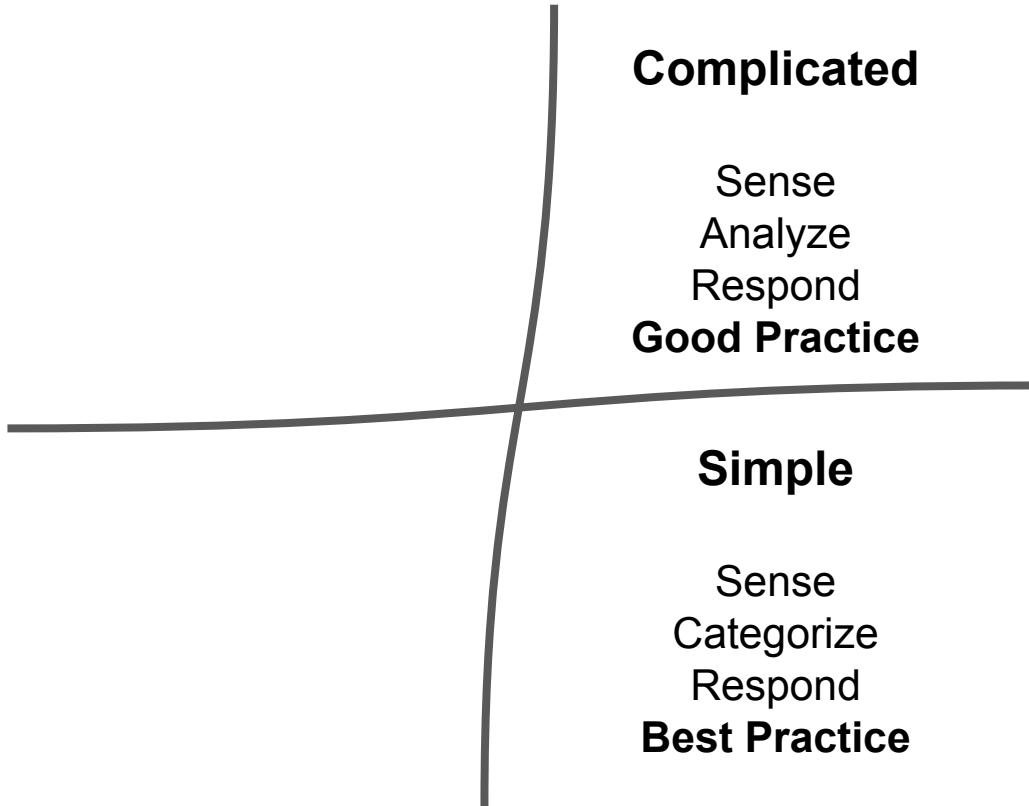
Cynefin Framework



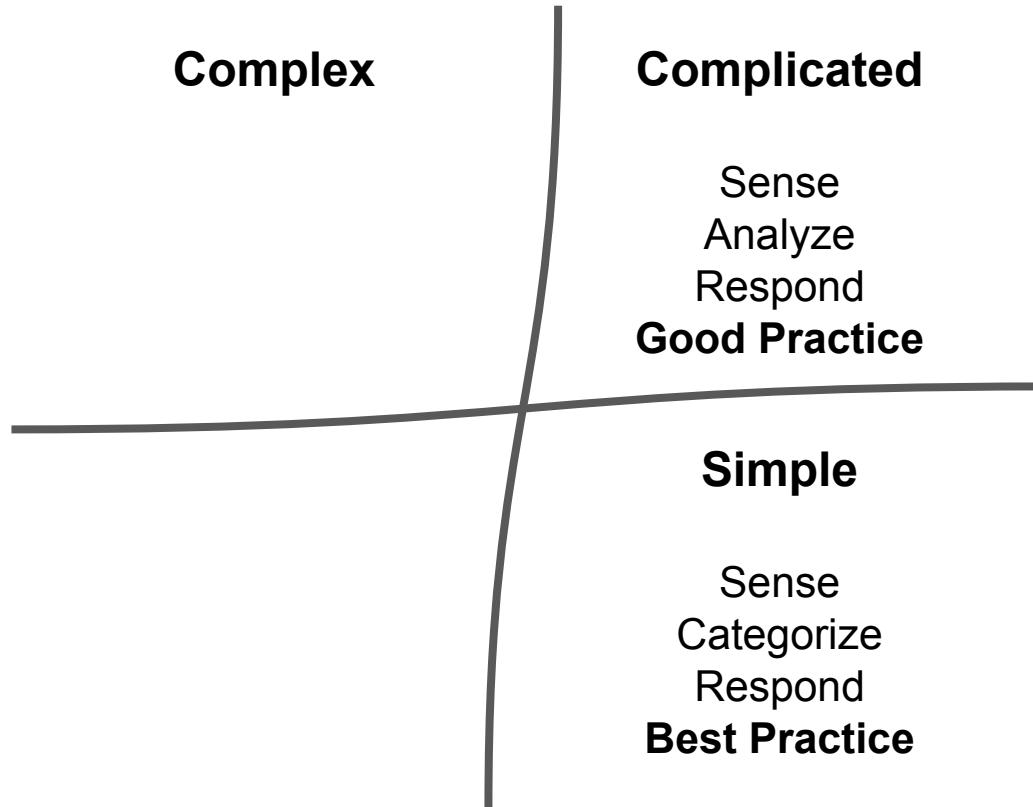
Cynefin Framework



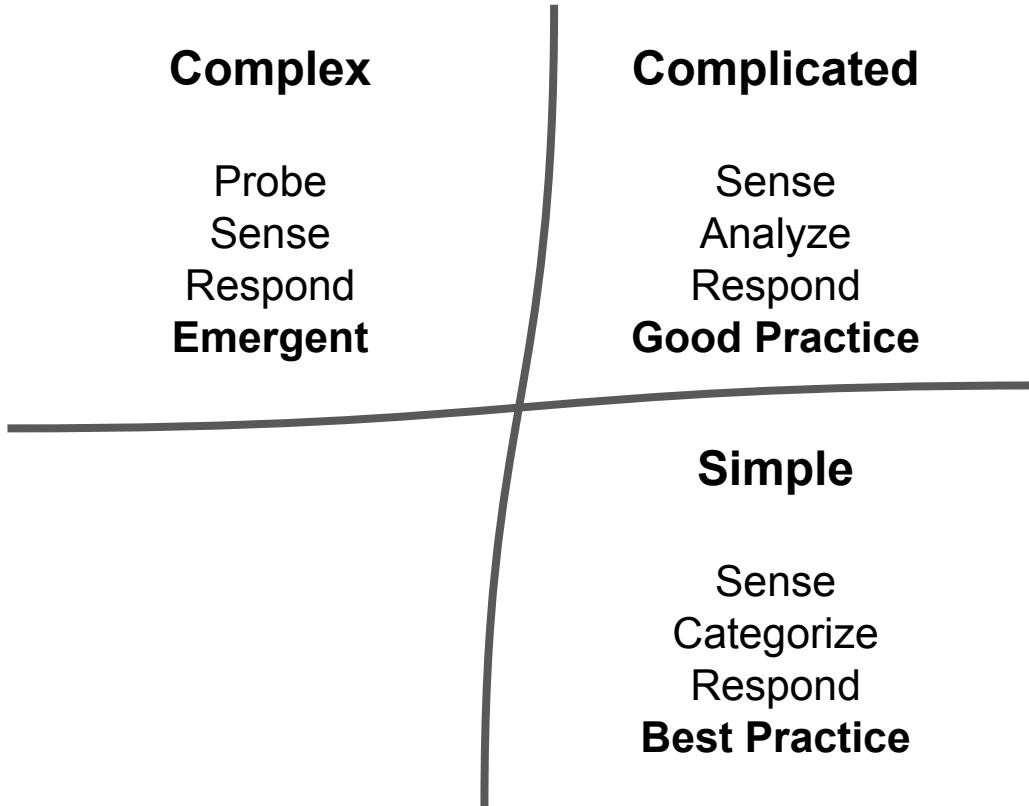
Cynefin Framework



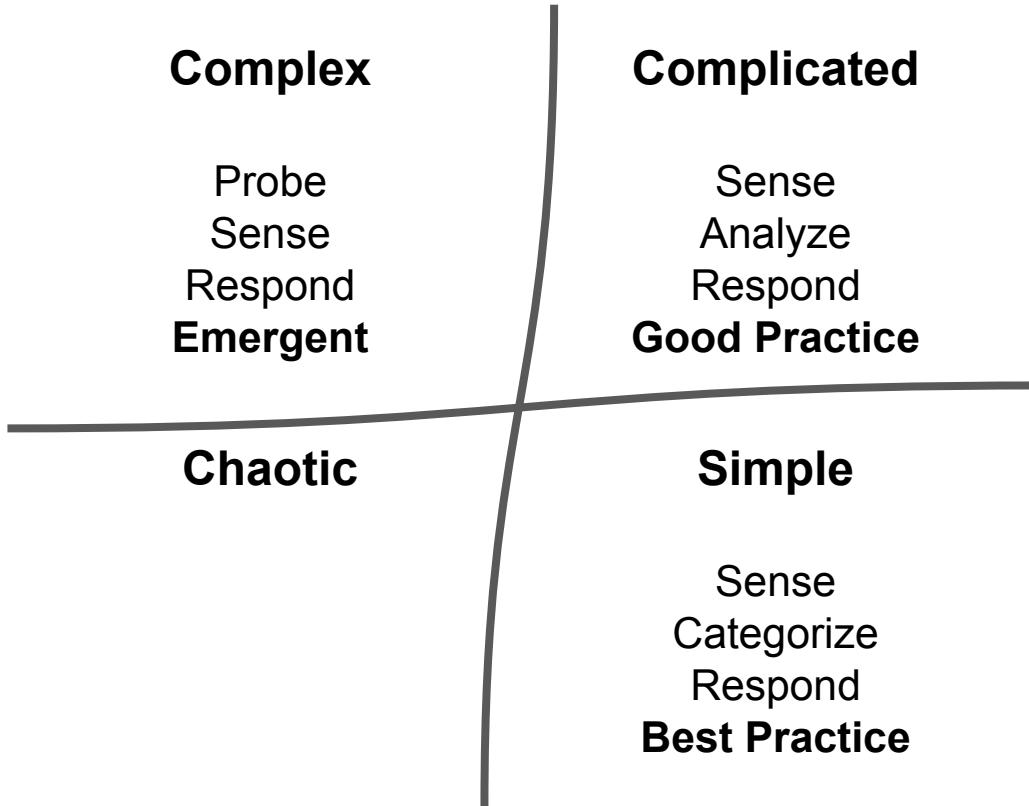
Cynefin Framework



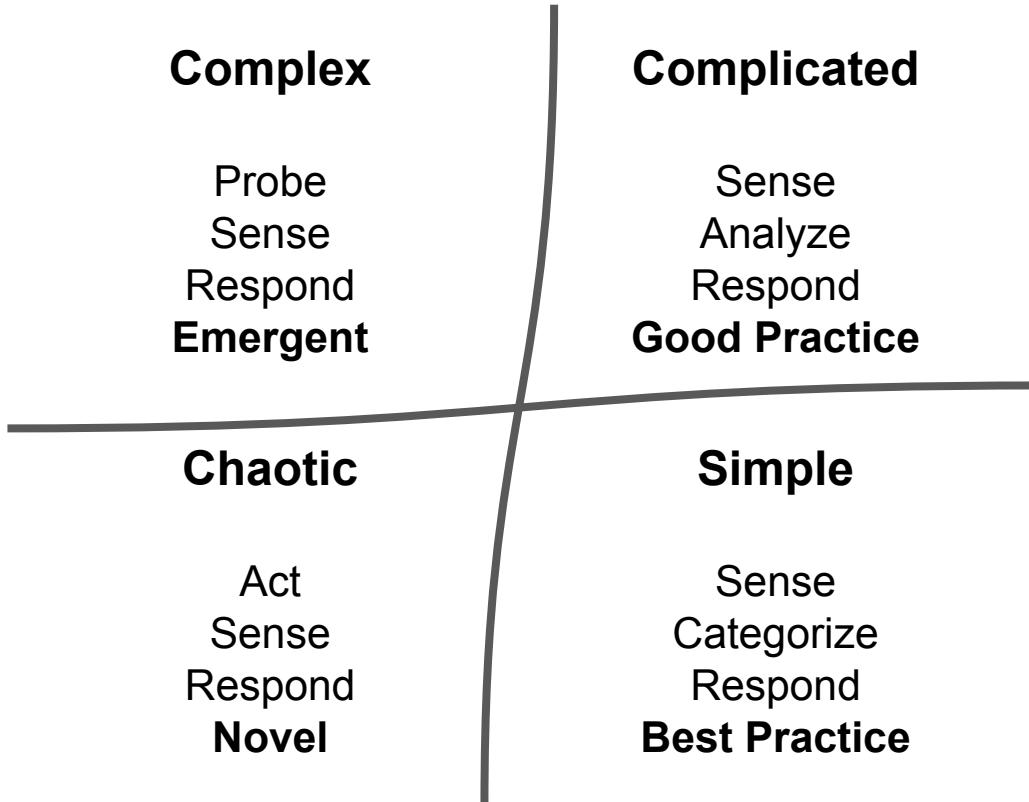
Cynefin Framework



Cynefin Framework



Cynefin Framework

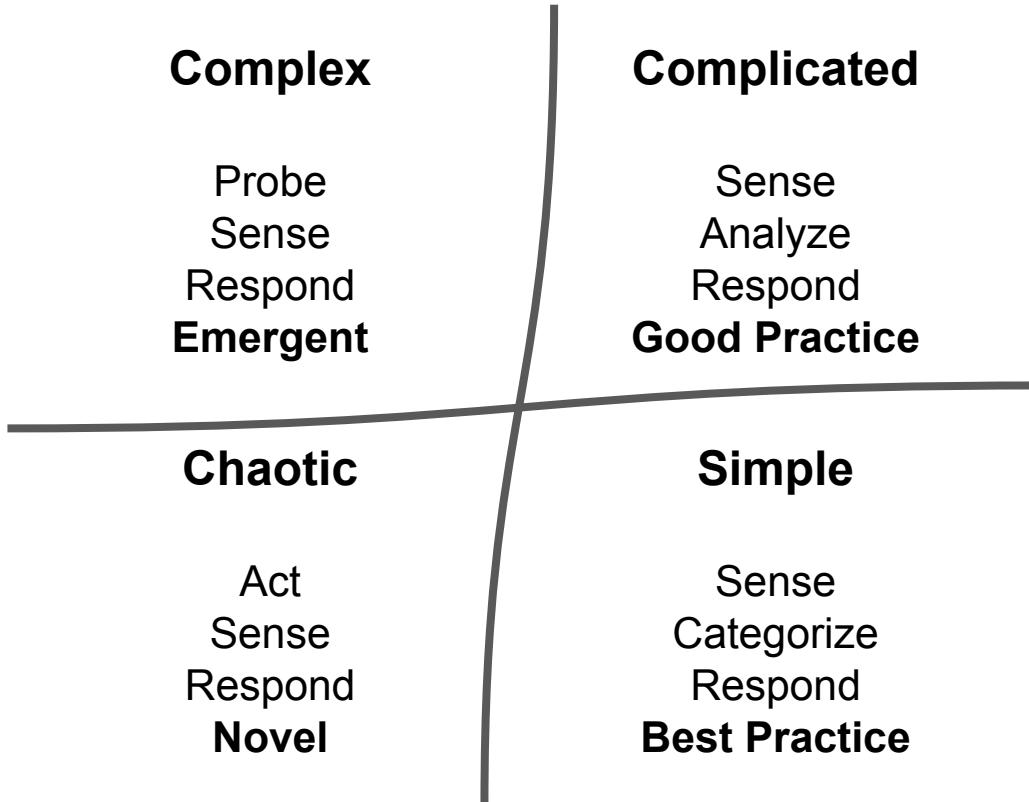


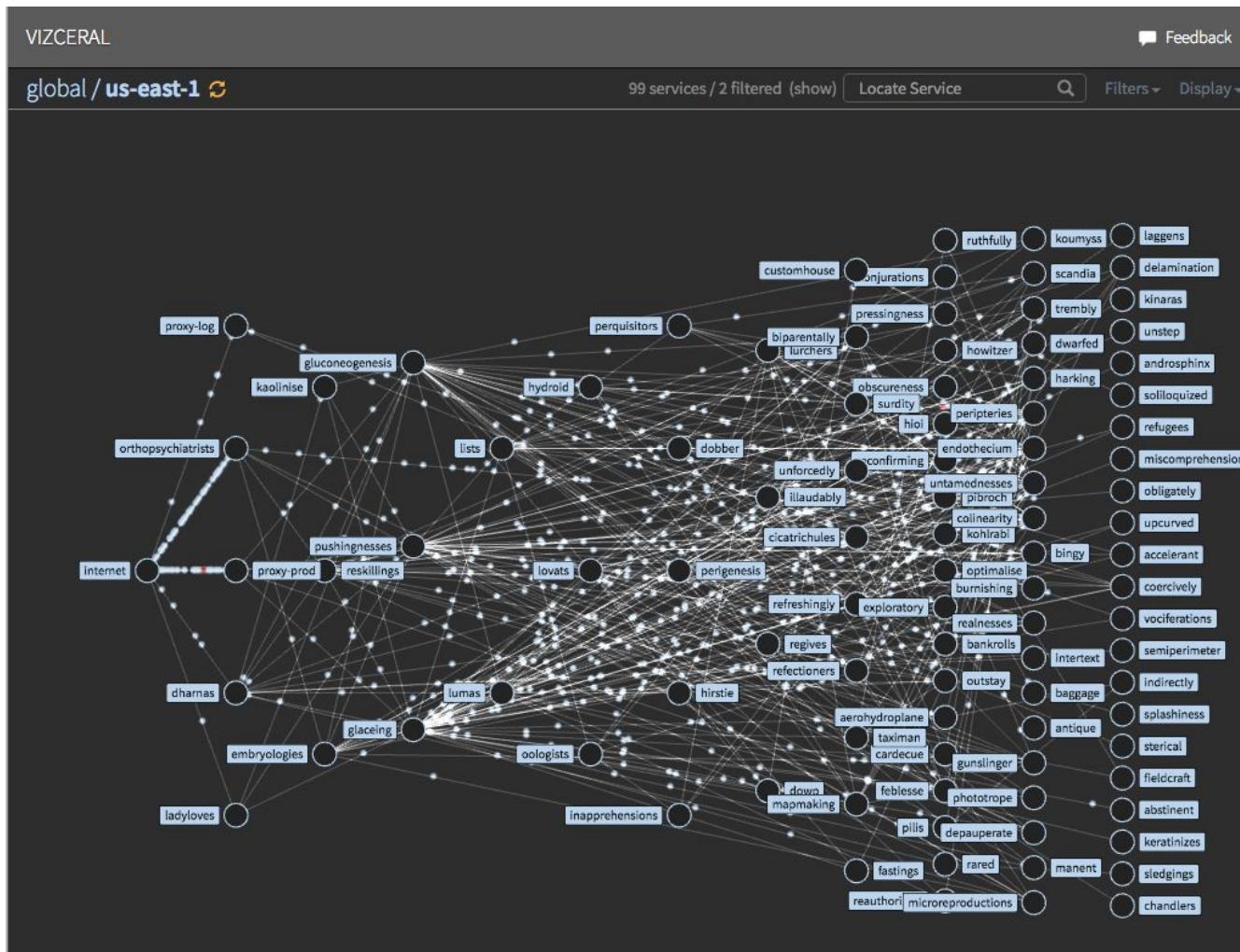
Catfin Framework



credit: @johnCutlefish

Cynefin Framework





VIZCERAL

Service Traffic Map / us-east-1

200 services / 116 filtered (show)

Locate Service



Filter +

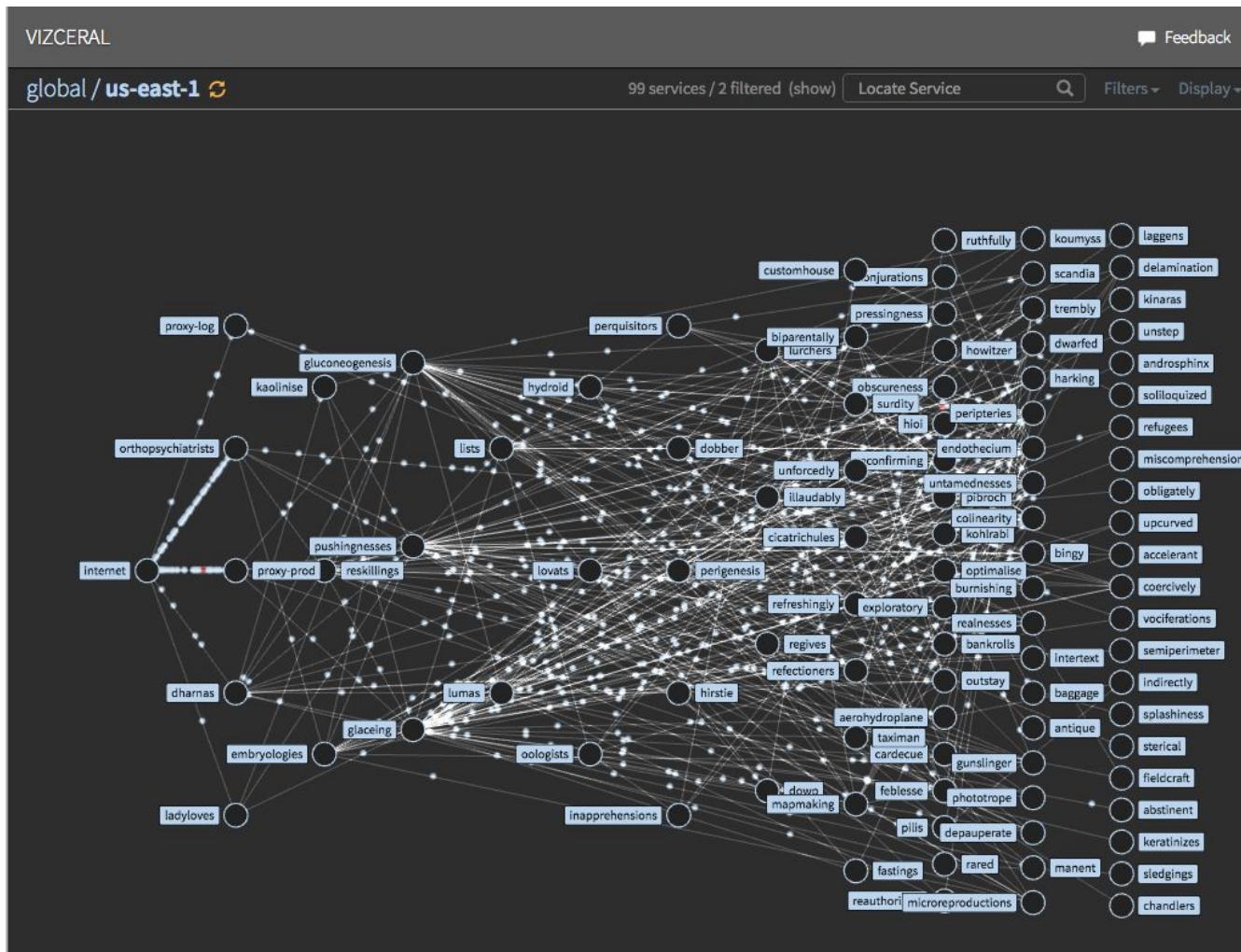
Display +



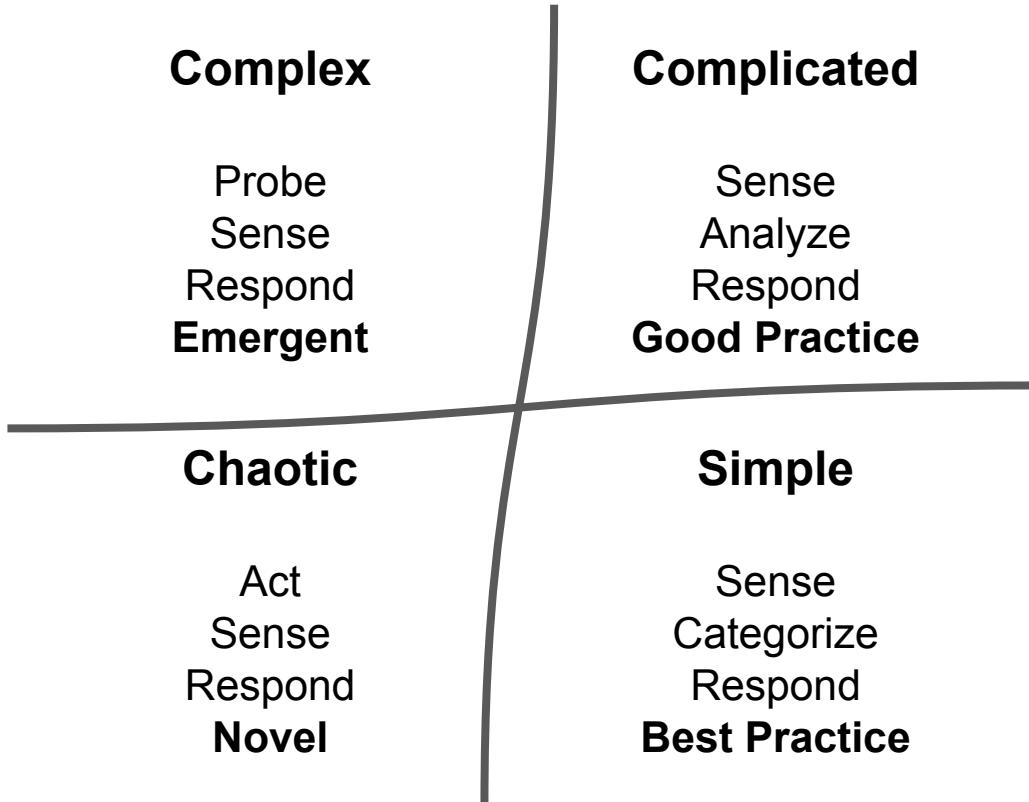
global / us-east-1 

99 services / 2 filtered (show)

Locate Service



Cynefin Framework



PROGRAMMER ANARCHY

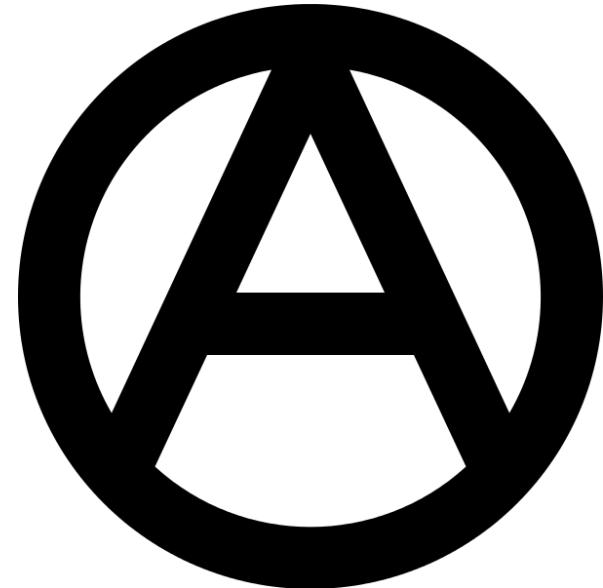
Programmer Anarchy

No managers (or other titles)

Programmers take responsibility

Expect/allow failure (remove fear)

Work directly with customers



Conclusion

Speed > efficiency

Speed > efficiency

DevOps

Speed > efficiency

DevOps

Empower teams

Speed > efficiency

DevOps

Empower teams

Embrace the distributed world

Microservices are **hard**.

Microservices are **hard**. Some things you get
for **free**

Microservices are **hard**. Some things you get for **free**, but you have to **work for the good stuff**.

Microservices are **hard**. Some things you get for **free**, but you have to **work for the good stuff**. If you won't put in the work, you shouldn't be doing Microservices.

“Microservices are **hard**. Some things you get for **free**, but you have to **work for the good stuff**. If you won’t put in the work, you shouldn’t be doing Microservices. (**You should be doing that stuff anyway!**)”

“Microservices are **hard**. Some things you get for **free**, but you have to **work for the good stuff**. If you won’t put in the work, you shouldn’t be doing Microservices. (**You should be doing that stuff anyway!**)”

Steve Upton

Questions?

@Steve_Upton
steveupton.io

This work is licensed under a [Creative Commons Attribution 4.0 International License](#).

Books!

A Novel About IT,
DevOps, and Helping
Your Business Win

The Phoenix Project

Gene Kim, Kevin Behr,
and George Spafford



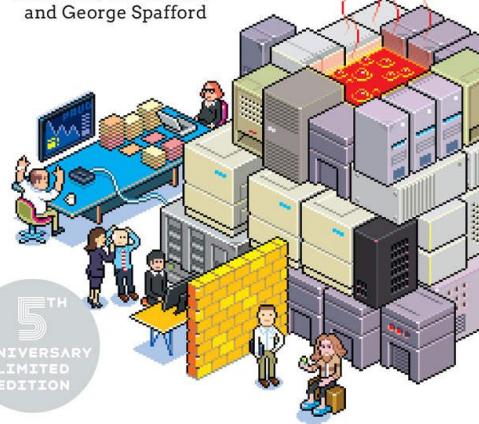
5TH
ANNIVERSARY
LIMITED
EDITION

Books!

A Novel About IT,
DevOps, and Helping
Your Business Win

The Phoenix Project

Gene Kim, Kevin Behr,
and George Spafford



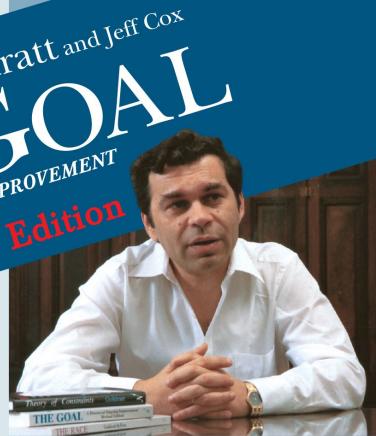
Includes
bonus
presentation
by the
author

Eliyahu M. Goldratt and Jeff Cox
THE GOAL
A PROCESS OF ONGOING IMPROVEMENT
30th Anniversary Edition

Eli Goldratt has been described by *Fortune* as a “guru to industry” and by *Business Week* as a “genius”. His book, *The Goal*, is a gripping fast-paced business novel.

“*Goal* readers are now doing the best work of their lives.”
Success Magazine

“A factory may be an unlikely setting for a novel, but the book has been wildly effective...”
Tom Peters



THE BEST-SELLING BUSINESS
NOVEL THAT INTRODUCED THE
**THEORY OF
CONSTRAINTS**
AND CHANGED HOW
AMERICA DOES BUSINESS

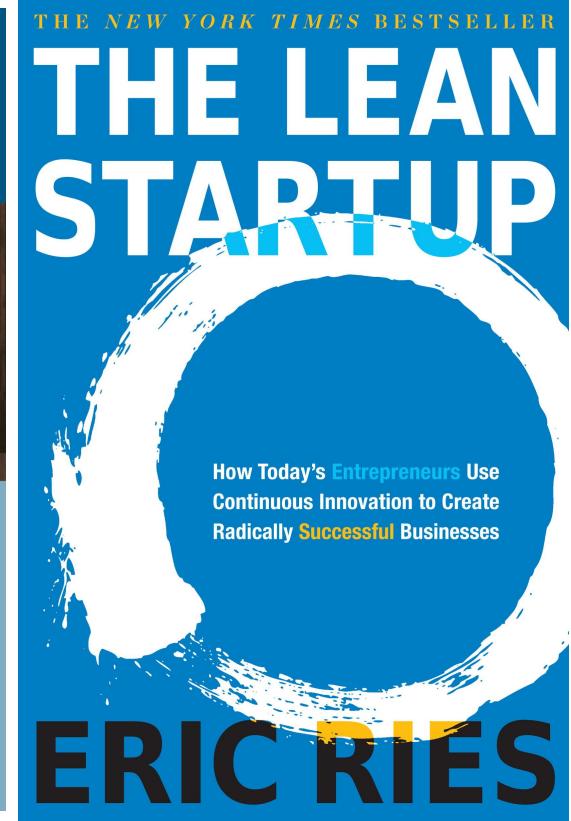
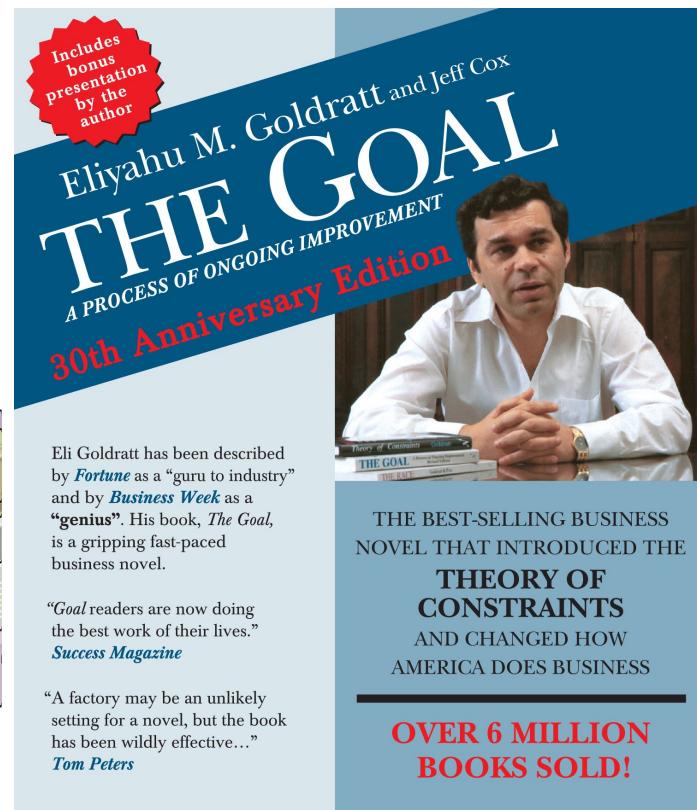
**OVER 6 MILLION
BOOKS SOLD!**

Books!

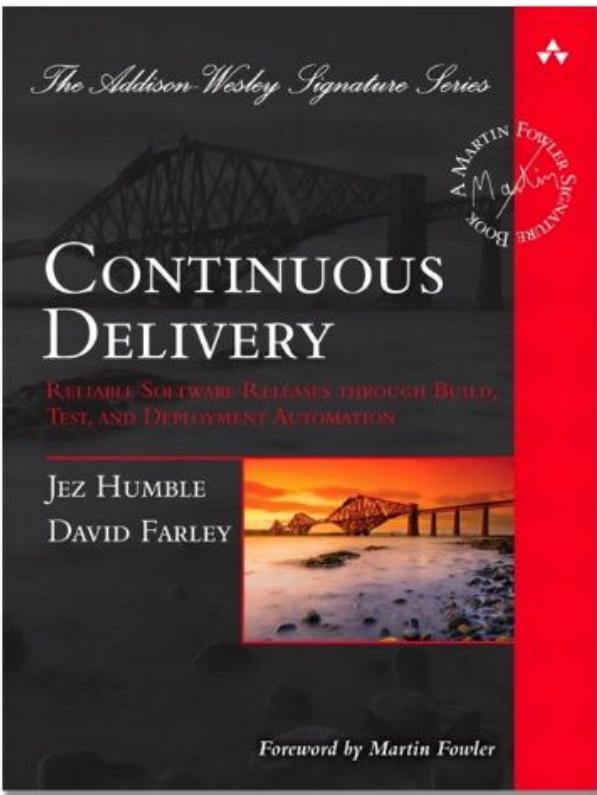
A Novel About IT,
DevOps, and Helping
Your Business Win

The Phoenix Project

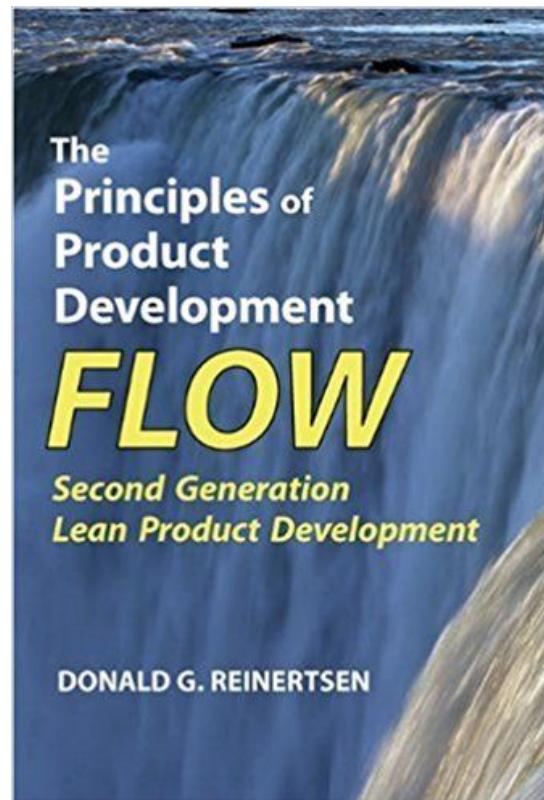
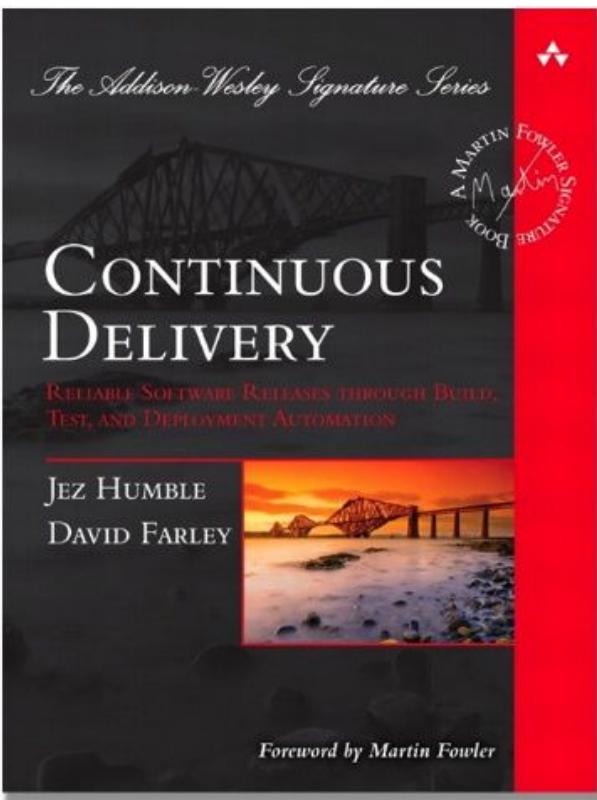
Gene Kim, Kevin Behr,
and George Spafford



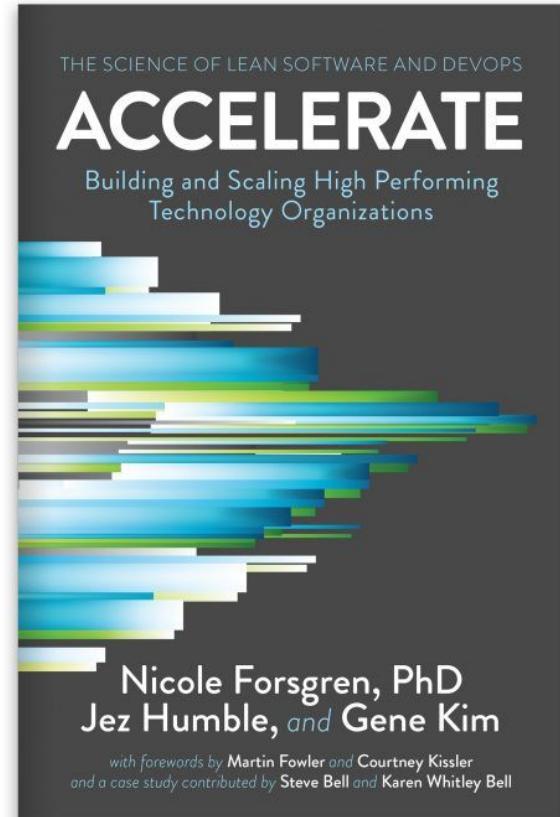
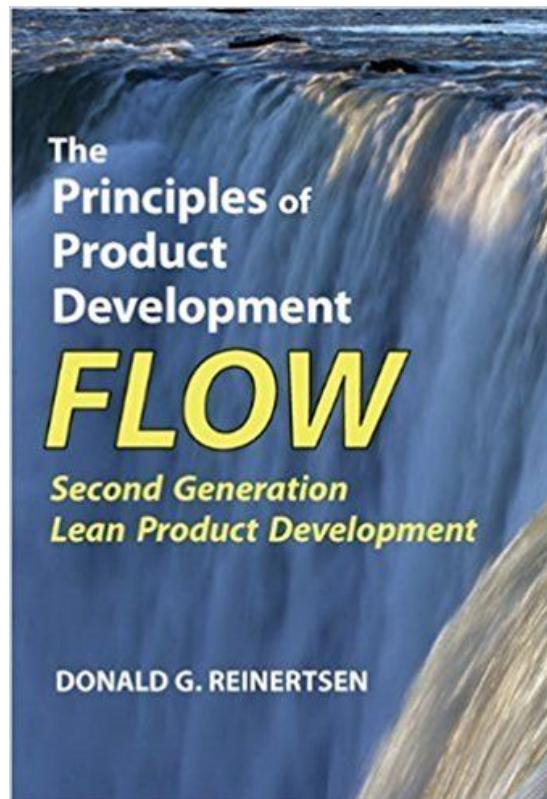
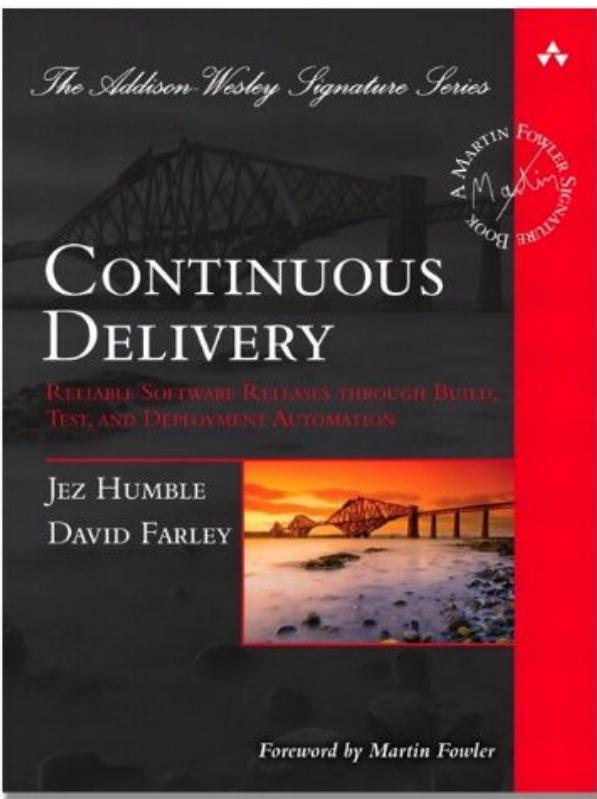
Books!



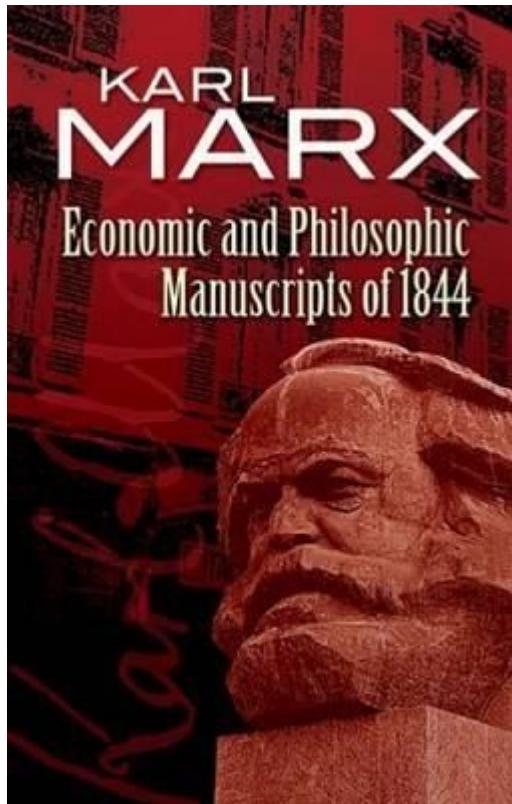
Books!



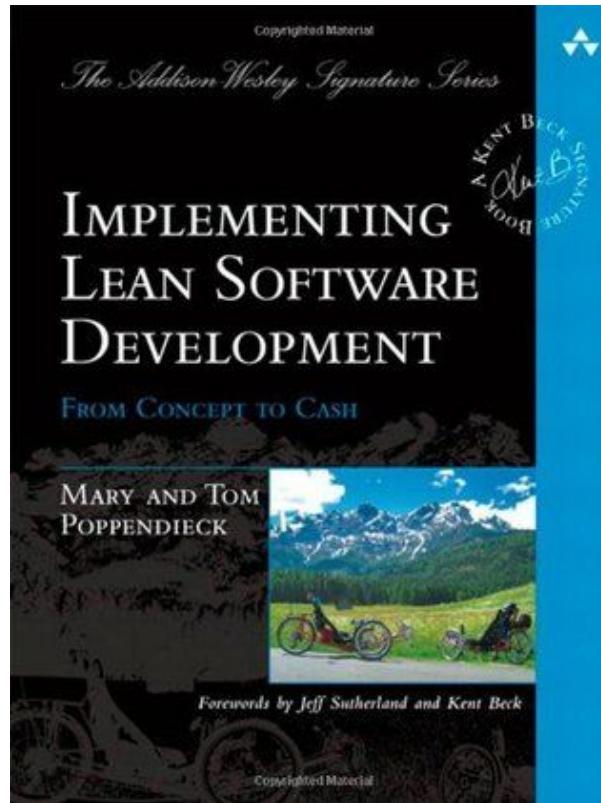
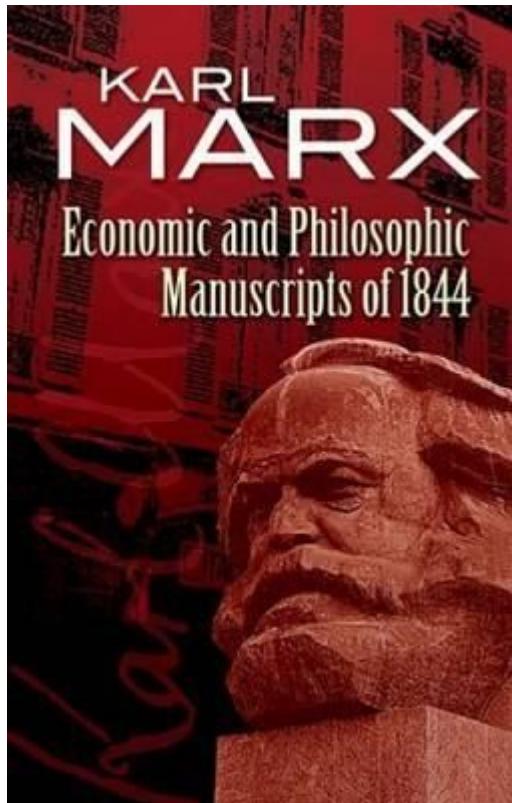
Books!



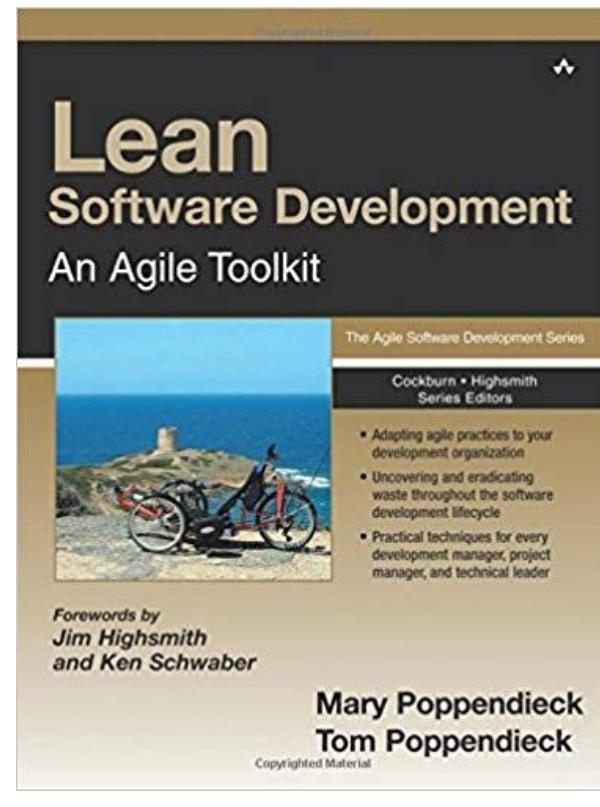
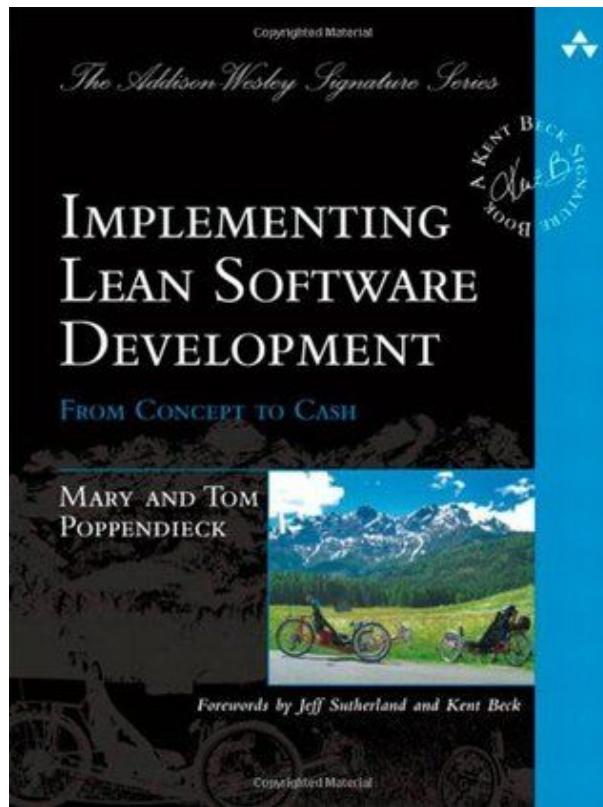
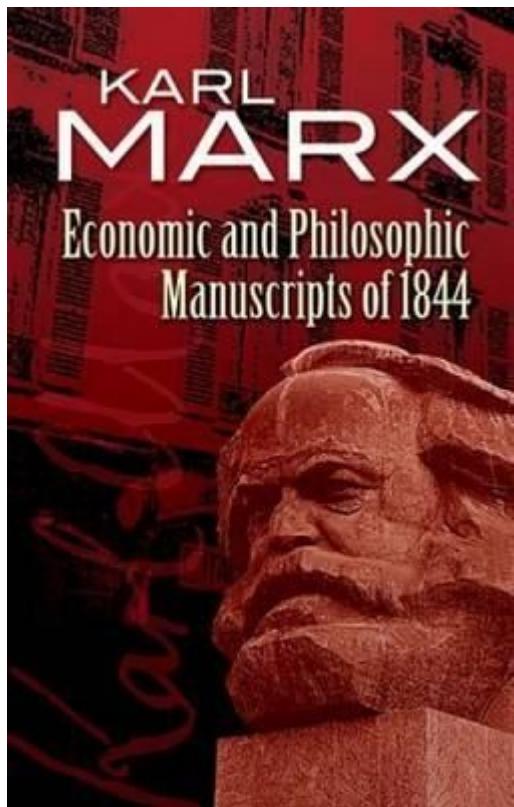
Books!



Books!



Books!



Essential reading

<https://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed>

<https://gist.github.com/chitchcock/1281611>

<https://engineering.linkedin.com/distributed-systems/log-what-every-software-engineer-should-know-about-real-time-datas-unifying>

<http://martinfowler.com/articles/microservices.html>

<http://jonasboner.com/bla-bla-microservices-bla-bla/>

<https://www.youtube.com/watch?v=EWUeZoyB0k>

Good reading

<http://highscalability.com/blog/2014/10/27/microservices-in-production-the-good-the-bad-the-it-works.html>

<http://highscalability.com/blog/2014/4/8/microservices-not-a-free-lunch.html>

<http://www.nearform.com/nodecrunch/microservices-series-4-beware-monolith/>

<https://hbr.org/2007/11/a-leaders-framework-for-decision-making>

<https://www.confluent.io/blog/publishing-apache-kafka-new-york-times/>

<http://milinda.pathirage.org/kappa-architecture.com/>

Good reading

<https://www.cio.com/article/2438748/the-it-measurement-inversion.html>

References

<http://www.slideshare.net/BruceWong3/the-case-for-chaos>

<http://www.slideshare.net/fredgeorge>

<http://www.slideshare.net/dataloop/anne-currie-force12io-game-of-hosts-containers-vs-vms>

<https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IB.pdf>

<https://martinjeeblog.com/2012/11/20/what-is-programmer-anarchy-and-does-it-have-a-future/>

<http://static.googleusercontent.com/media/research.google.com/en//people/jeff/Berkeley-Latency-Mar2012.pdf>

<http://apsblog.burtongroup.com/2009/01/soa-is-dead-long-live-services.html>

References cont.

<http://www.ibm.com/developerworks/webservices/library/ar-esbpat1/>

<https://ocw.mit.edu/courses/aeronautics-and-astronautics/16-885j-aircraft-systems-engineering-fall-2005/>

<https://www.fogcreek.com/blog/eight-fallacies-of-distributed-computing-tech-talk/>

<http://www.ibiblio.org/harris/500milemail.html>

<http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/2015-Annual-Report.pdf>

<http://www.counterpunch.org/2013/03/04/when-money-is-no-object-the-strange-saga-of-the-f-35/>

References cont.

<https://www.youtube.com/watch?v=dxk8b9rSKOo>

<http://assets.en.oreilly.com/1/event/60/Velocity%20Culture%20Presentation.pdf>

<http://www.cubrid.org/blog/dev-platform/understanding-tcp-ip-network-stack/>

<https://www.oreilly.com/ideas/an-introduction-to-immutable-infrastructure>

<http://www.methodsandtools.com/archive/archive.php?id=97>

<http://redmonk.com/sogrady/2017/07/20/soa-microservices/>

https://medium.com/@haydar_ai/learning-how-to-git-merging-branches-and-resolving-conflict-61652834d4b0

<https://www.thoughtworks.com/de/continuous-integration>

References cont.

<https://www.thoughtworks.com/de/insights/blog/path-devops>

<https://martinfowler.com/articles/micro-frontends.html>

<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>

<https://martinfowler.com/bliki/BlueGreenDeployment.html>

<https://martinfowler.com/articles/feature-toggles.html>

<https://dev.to/mostlyjason/intro-to-deployment-strategies-blue-green-canary-and-more-3a3>

Bad things

<http://www.ibm.com/developerworks/webservices/library/ar-esbpat1/>

<http://www.networkworld.com/article/3114195/system-management/the-8-fallacies-of-distributed-computing-are-becoming-irrelevant.html>

Image credits

<http://fontawesome.io/>

<https://github.com/thepracticaldev/orly-full-res>

<http://dimaip.github.io/slides/docker101.html>

<http://www.clipartlord.com/category/military-clip-art/bomb-clip-art/explosion-clip-art/>

https://en.wikipedia.org/wiki/File:S-IC_engines_and_Von_Braun.jpg

http://www.nutsvolts.com/magazine/article/the_computer_that_took_man_to_the_moon

<https://spaceflightnow.com/2014/11/05/engineers-recommend-changes-to-orion-heat-shield/>

<https://www.hq.nasa.gov/alsj/alsj-DrinkFood.html>

Image credits cont.

<https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19720013196.pdf>

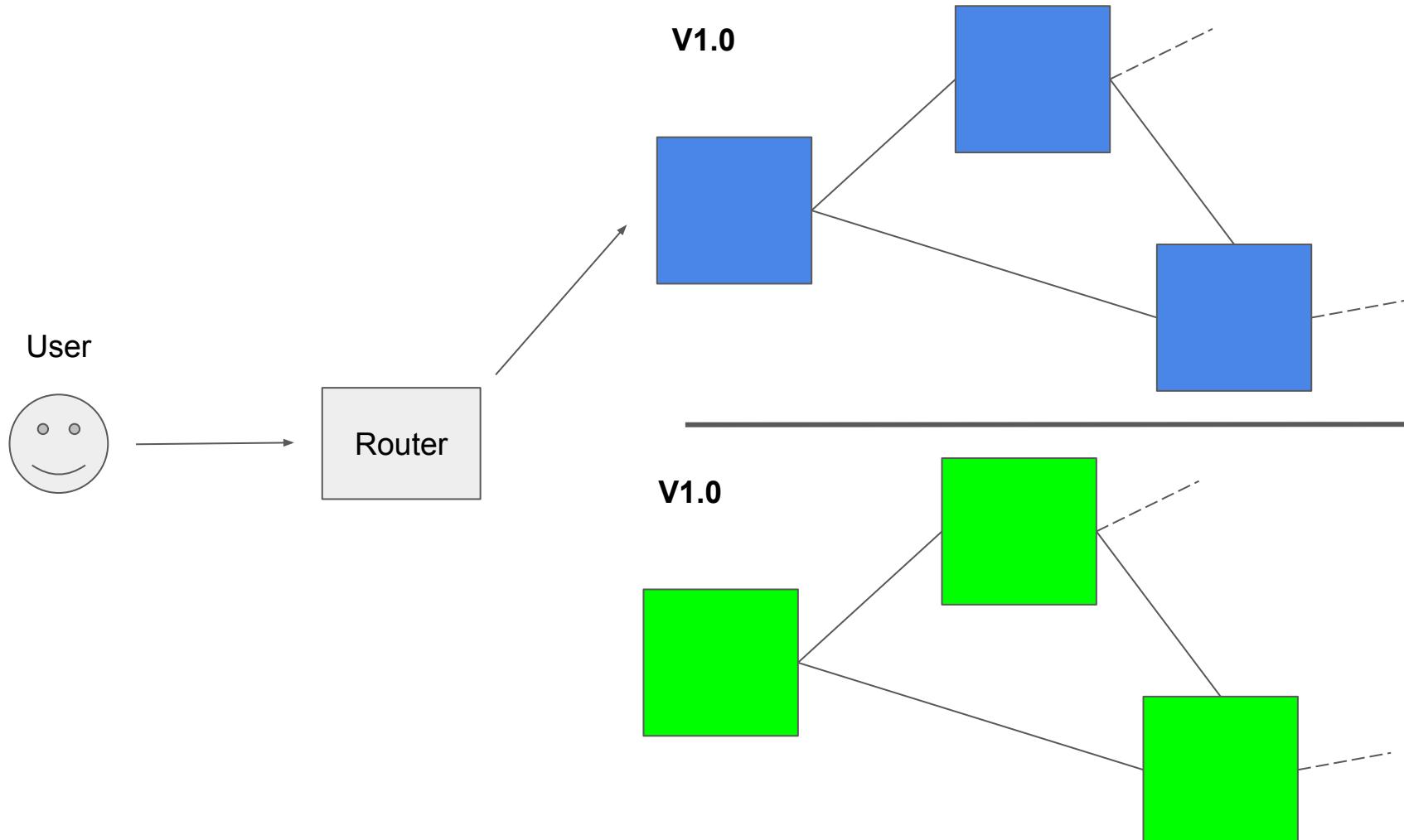
Extra material

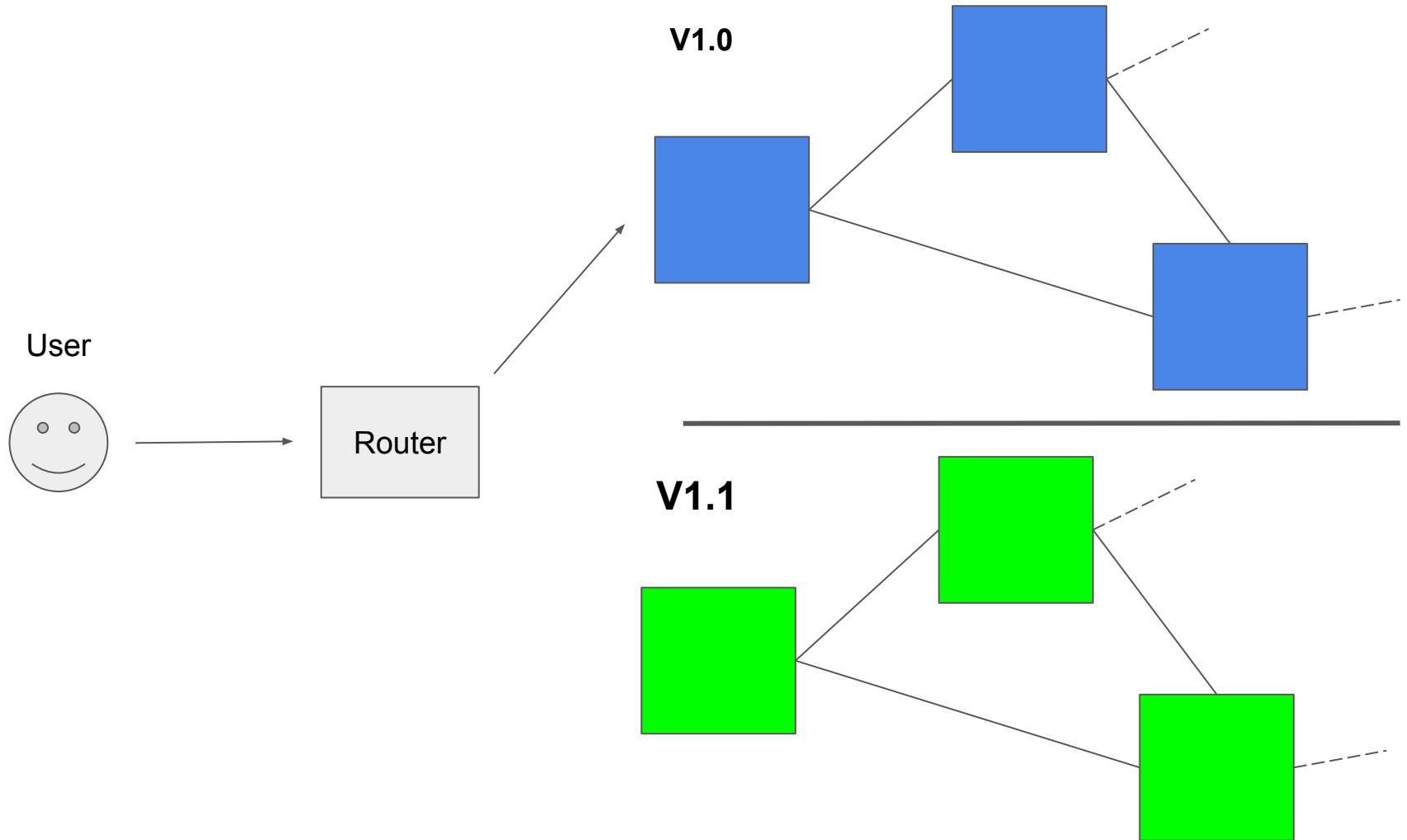
Blue-Green Deployment

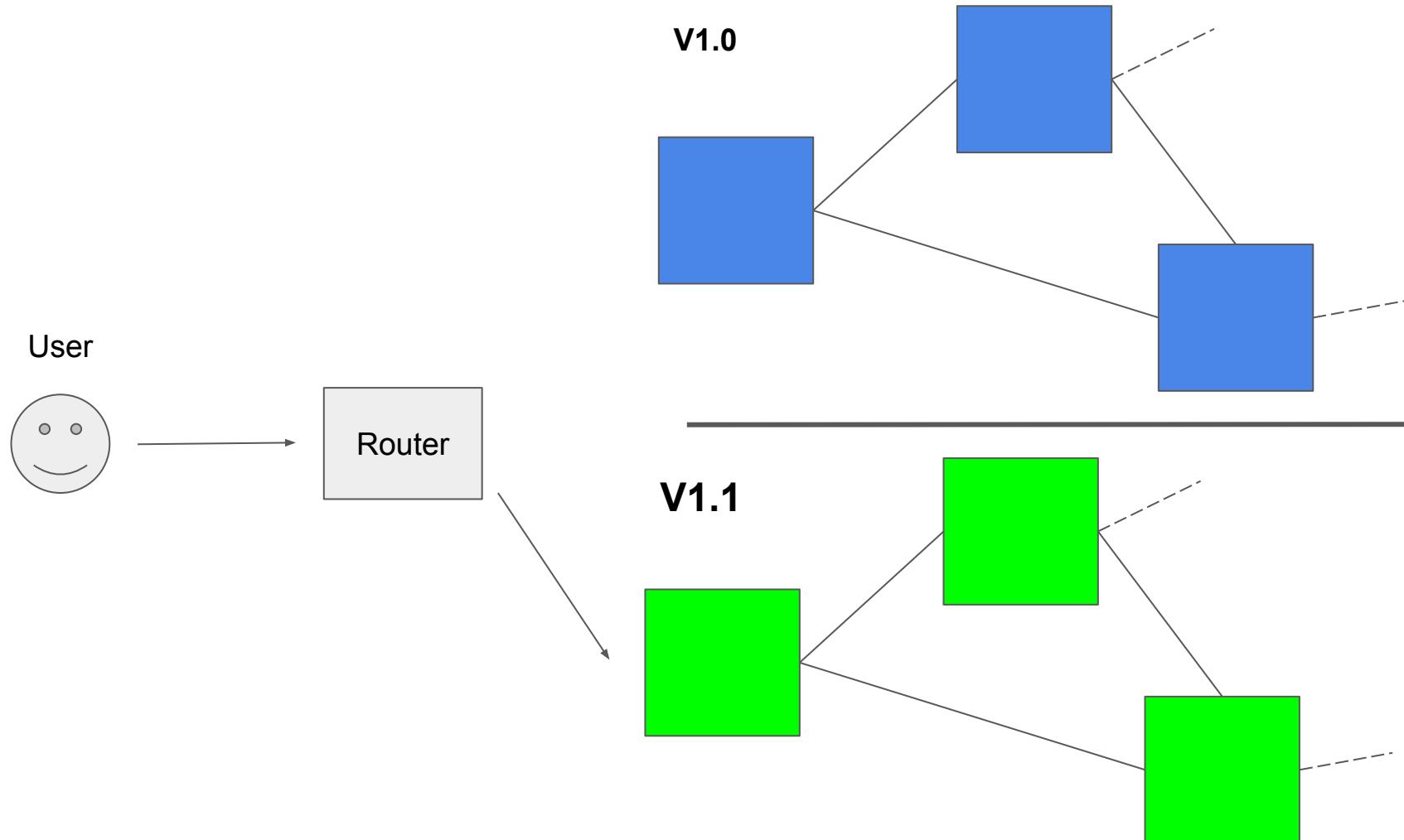
ESBs are shit

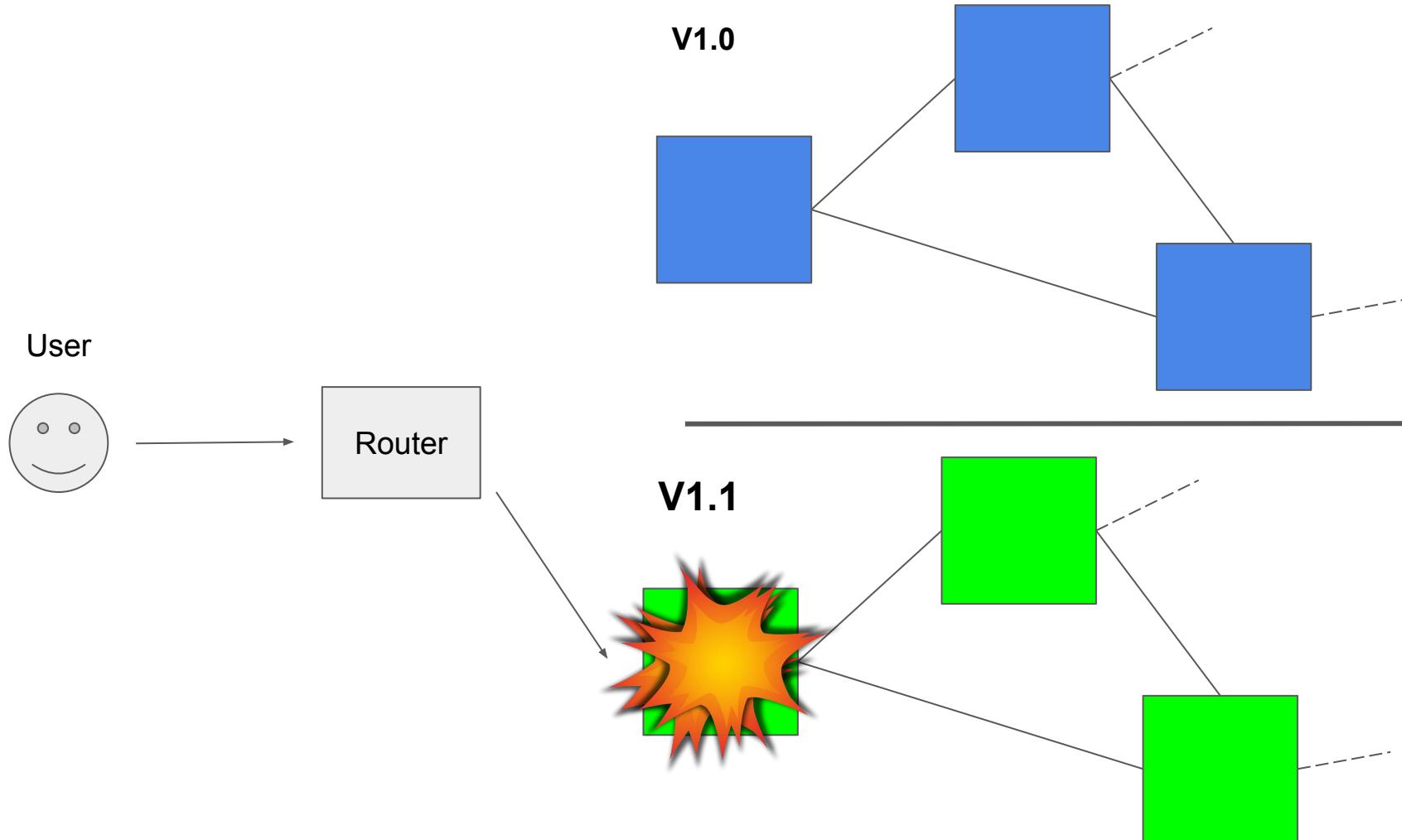
Language choice for Microservices

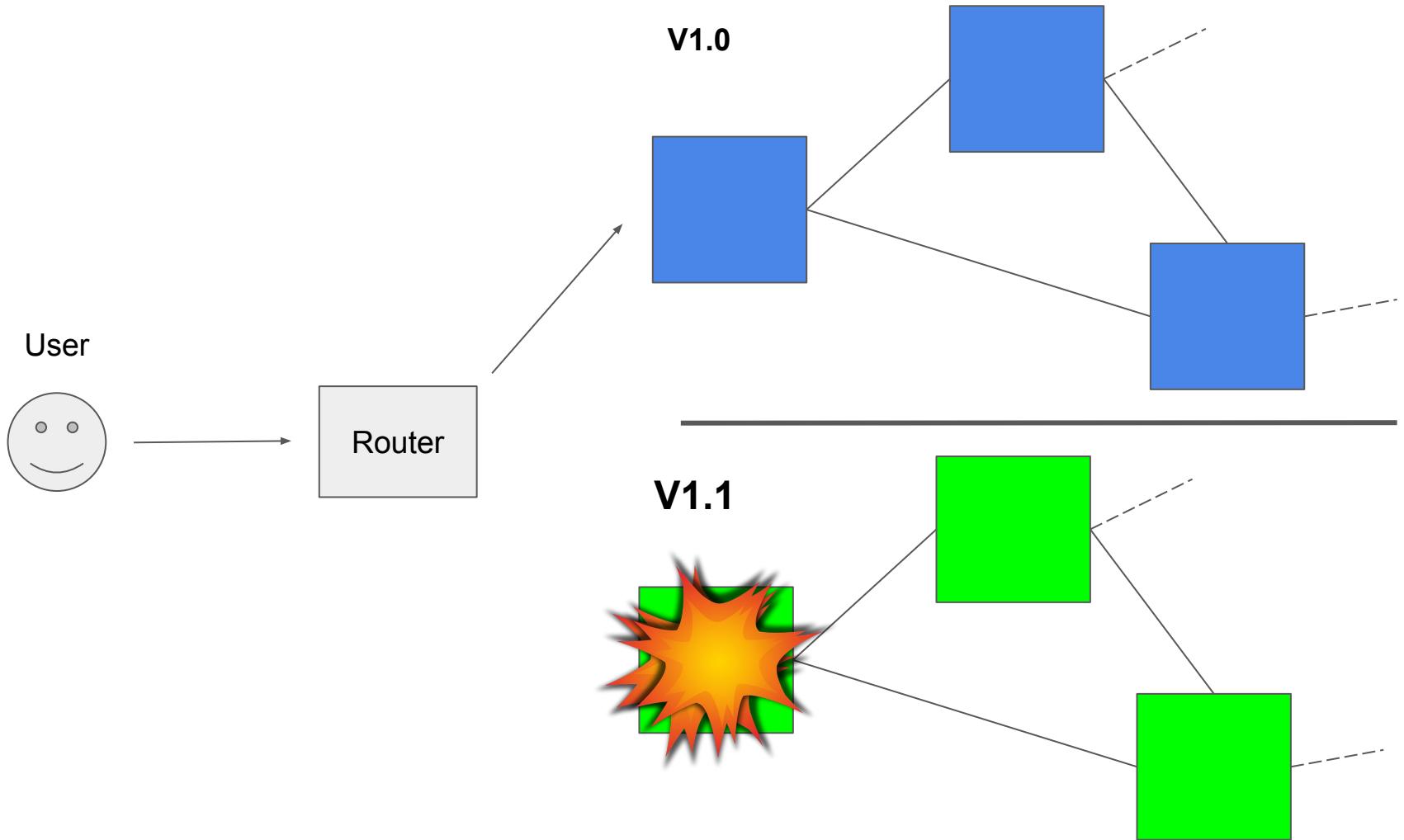
Blue-Green deployment









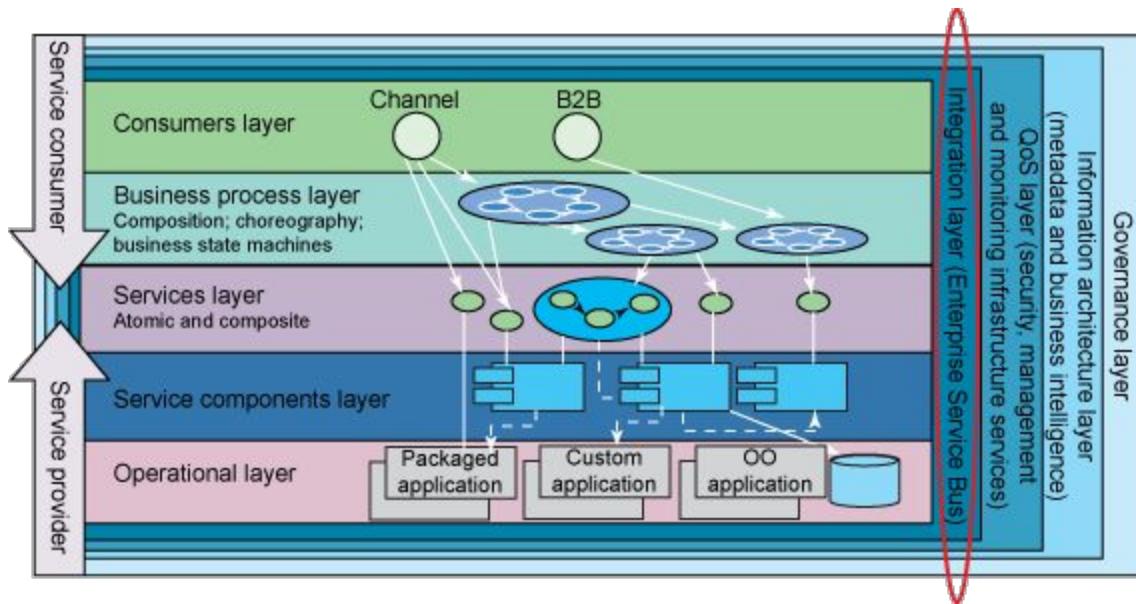


Possible with monoliths, just **harder!**

ESBs are shit

ESBs are shit
credit to Russ Miles





Service

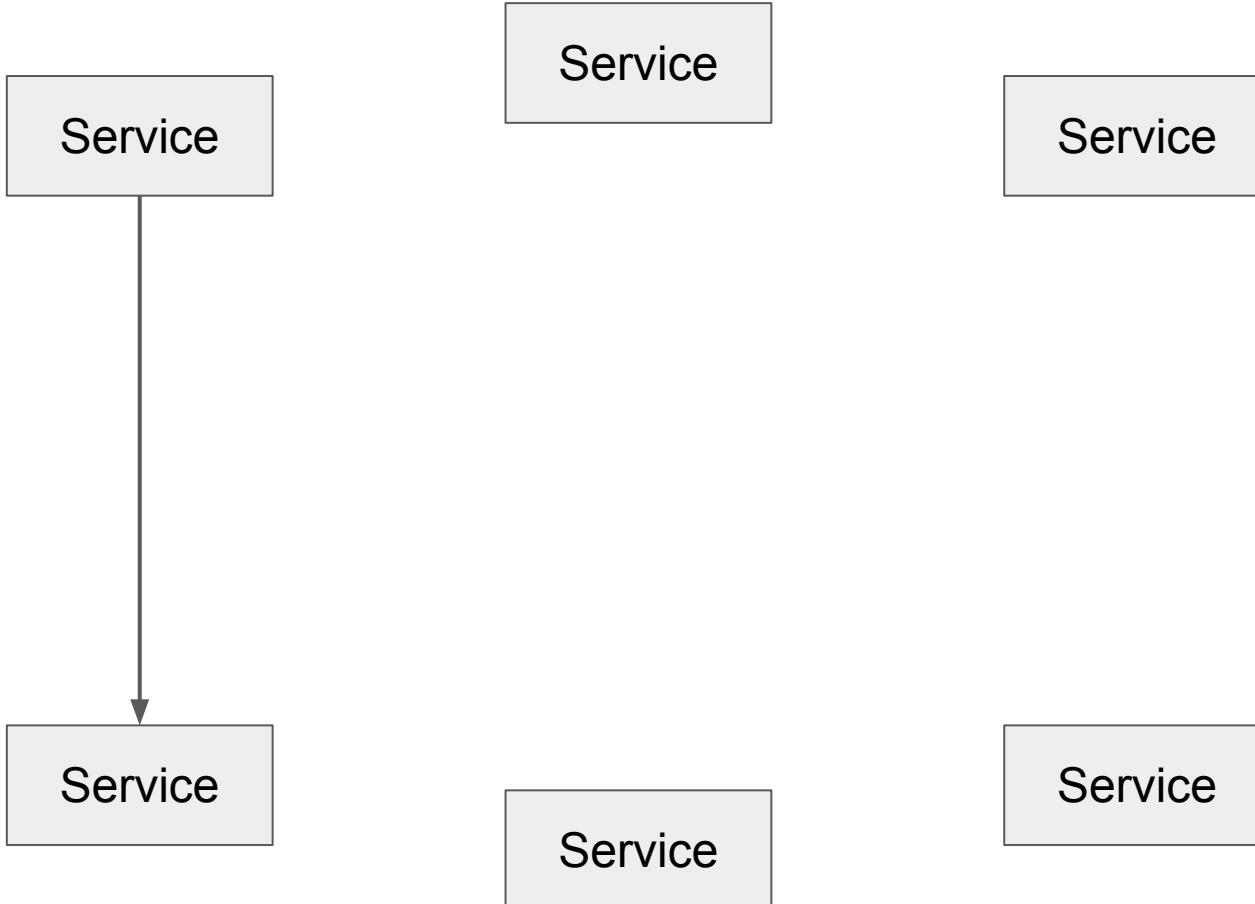
Service

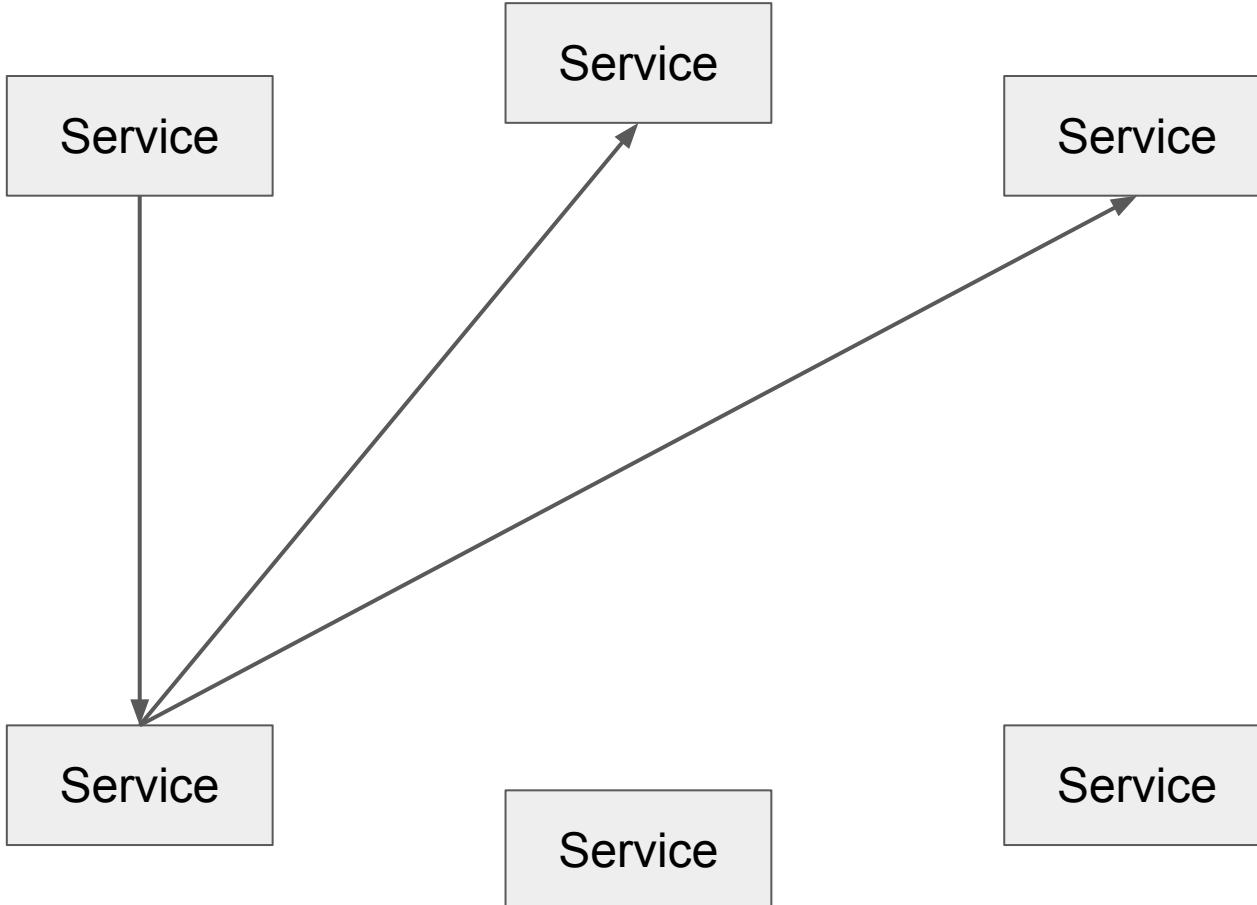
Service

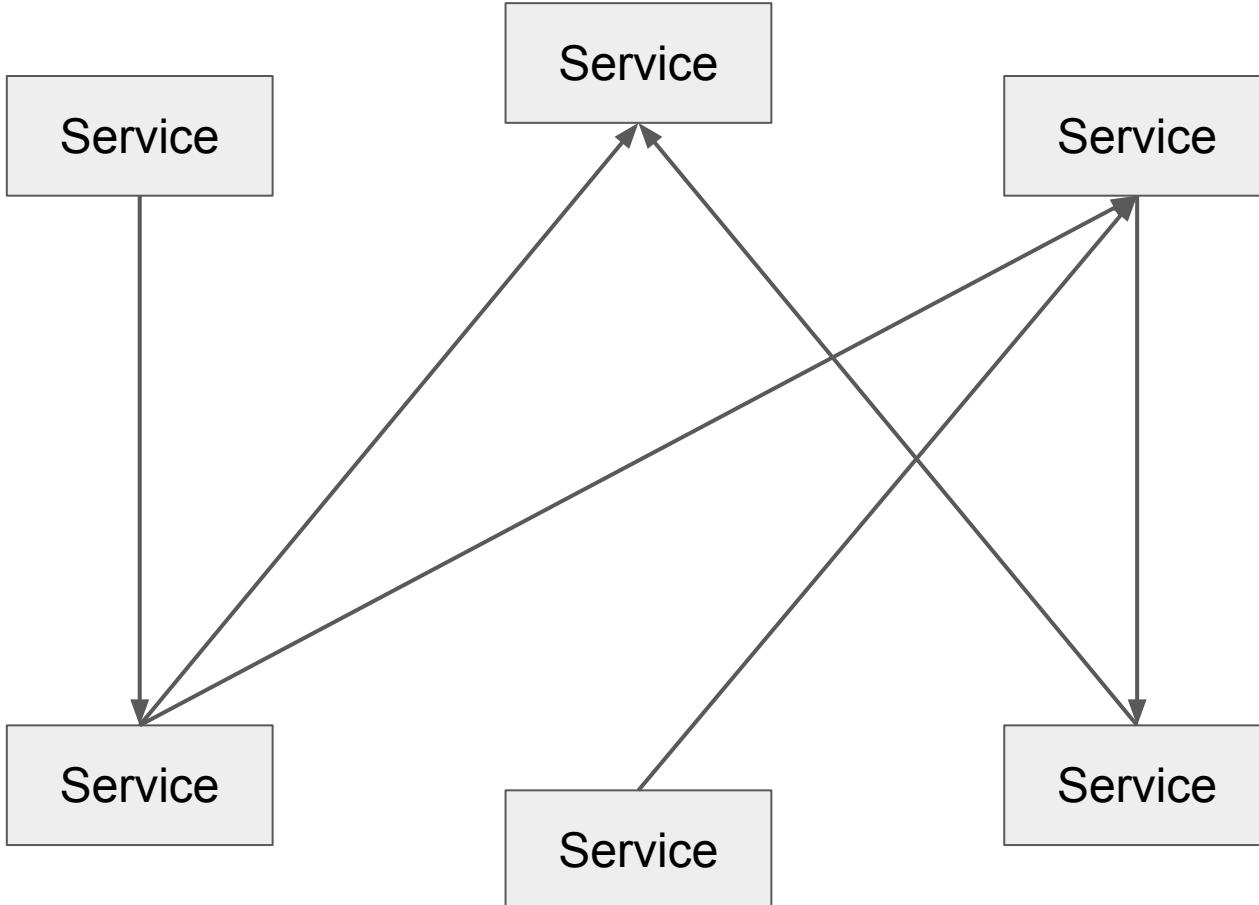
Service

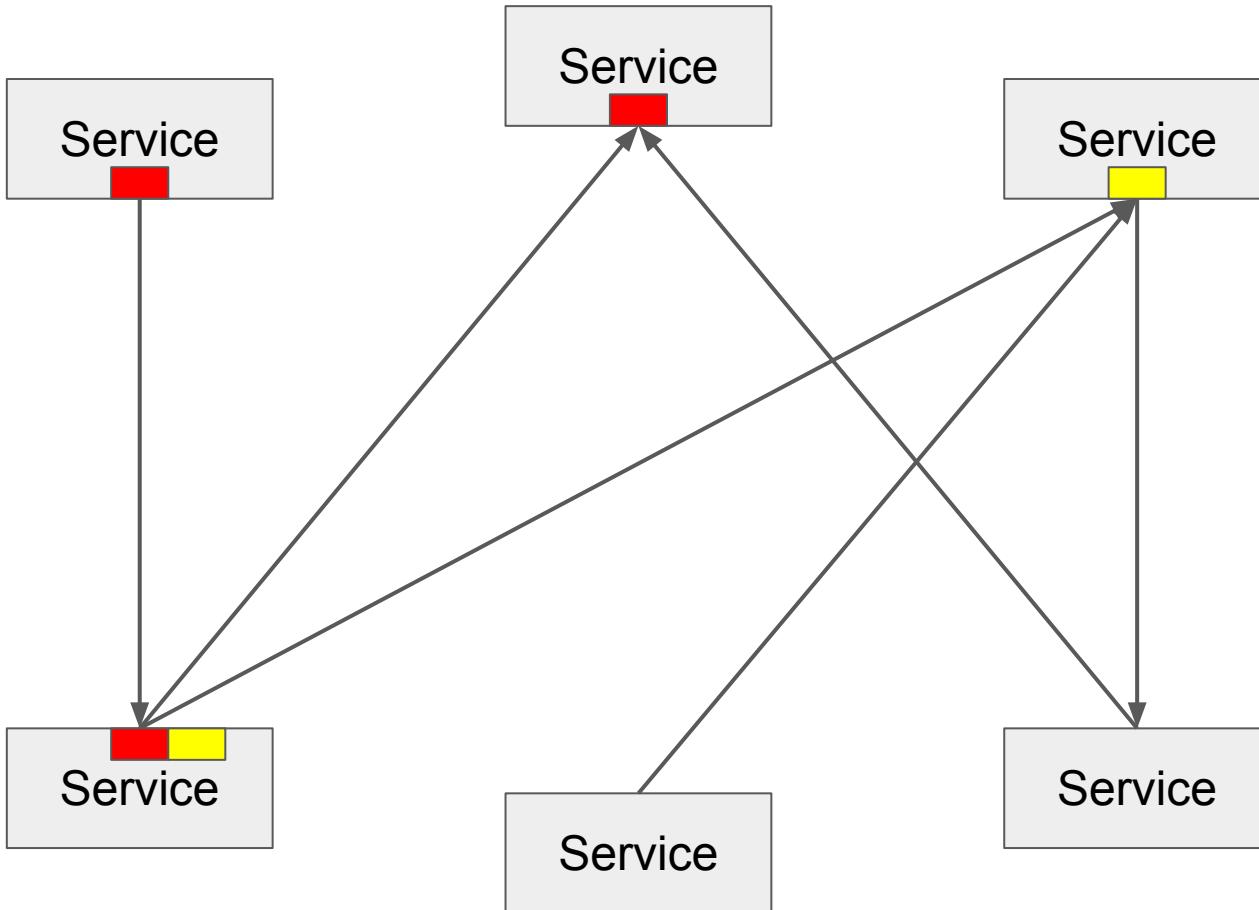
Service

Service









Smart endpoints and dumb pipes

Dumb Diagrams

