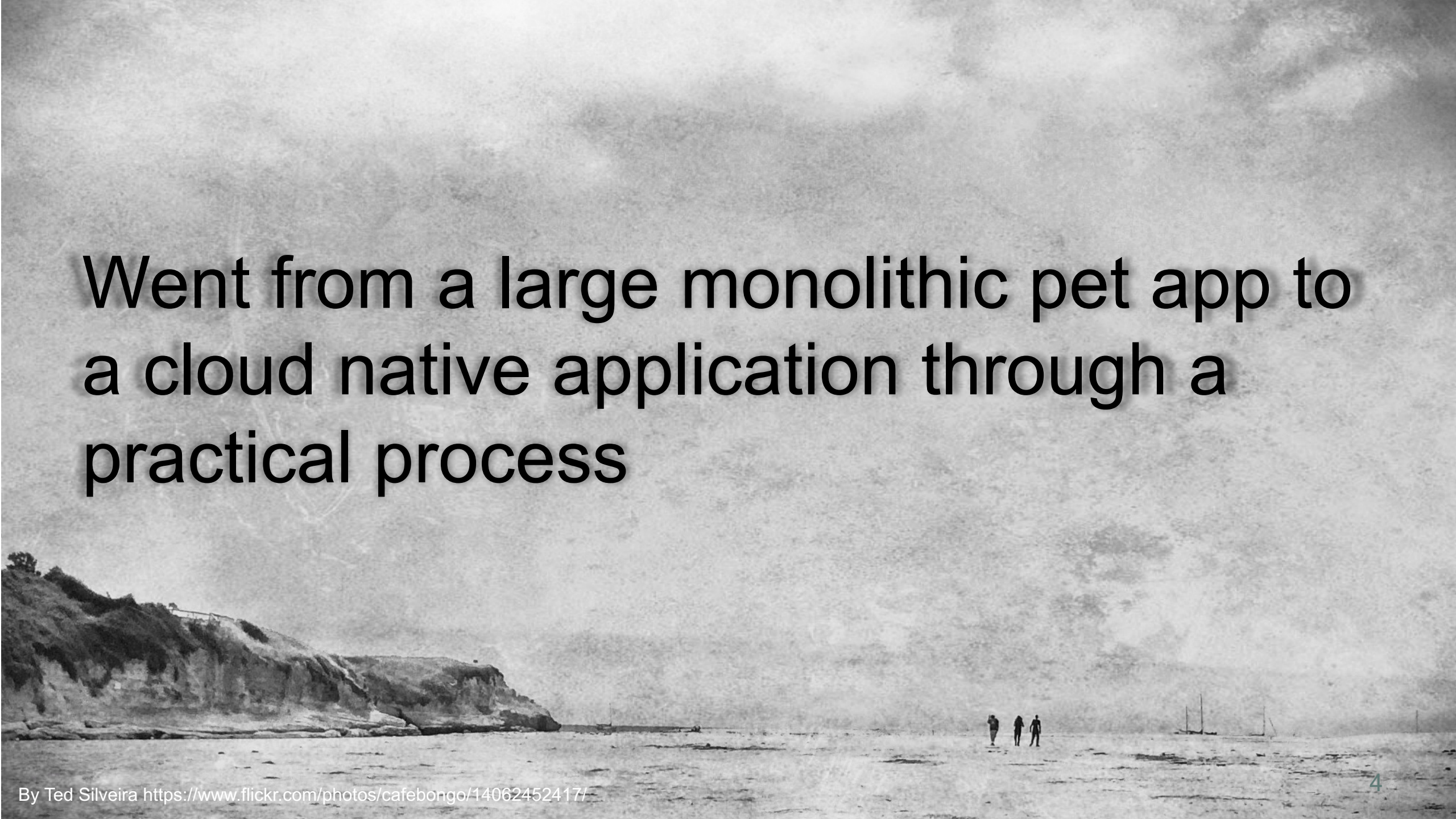

Hewlett Packard
Enterprise
Dipping Your Toes Into Cloud Native Application Development

Who is this guy?

- mattfarina.com, @mattfarina
- Principal Engineer, Advanced Technology Group, Hewlett Packard Enterprise
- Go, Node.js, PHP, Ruby, Python, .NET… in just past 5 years
- Recently wrote book, “Go in Practice”
- Co-lead Kubernetes SIG Apps
- Likes to build things in the cloud
- Doesn’t like bullet lists

By Swaminathan https://www.flickr.com/photos/araswami/2284451064/



By Ted Silveira https://www.flickr.com/photos/cafebongo/14062452417/

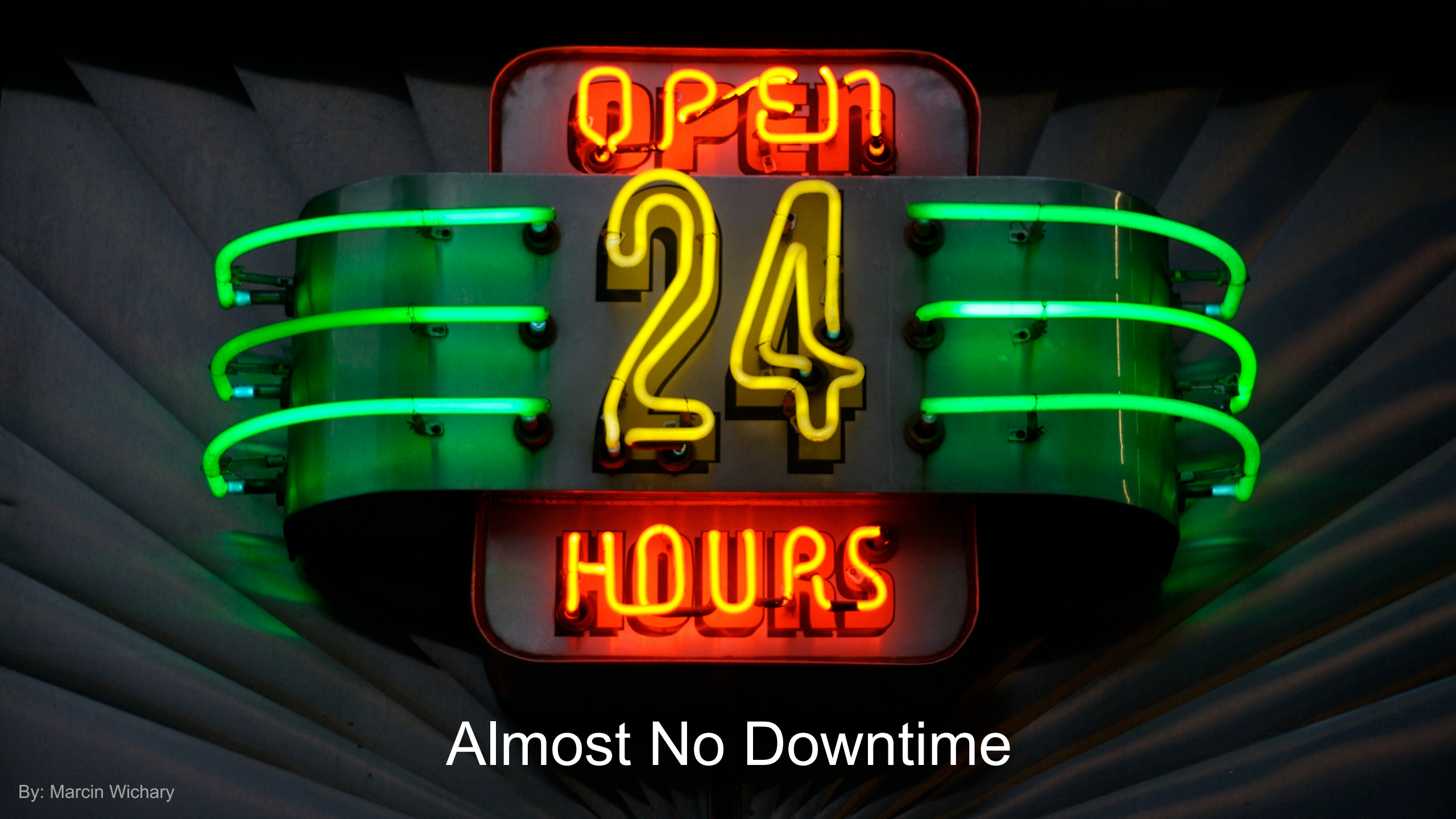
By Dubwise Version https://www.flickr.com/photos/dubwise_version/4718137117

1. Why care about cloud native development?
2. What is cloud native?
3. What tools can be used?
4. How can you get from here to there?

Why care about cloud native?

Speed and Flexibility

By: Fernando Frazão/Agência Brasil



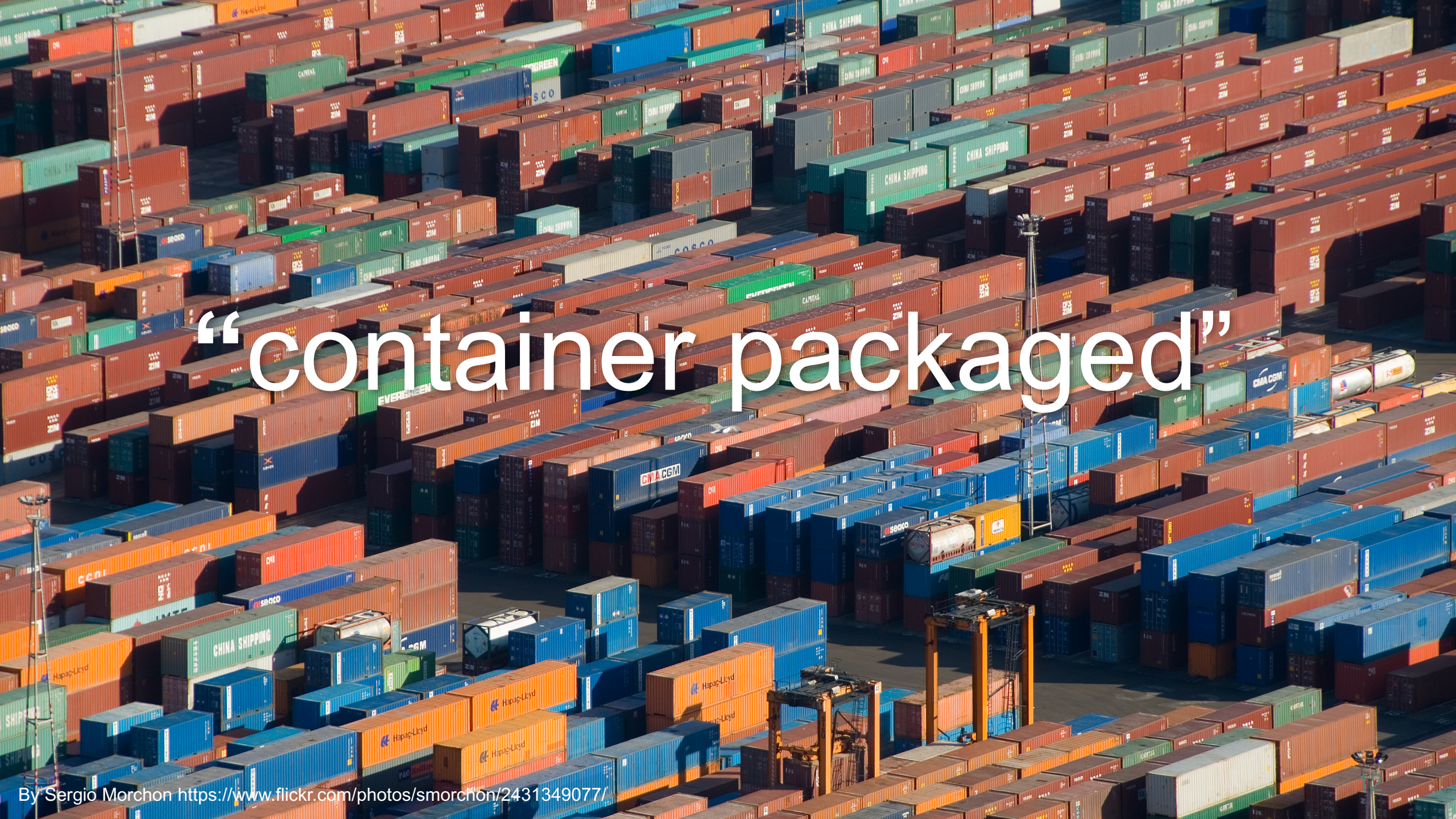
Almost No Downtime

By: Marcin Wichary

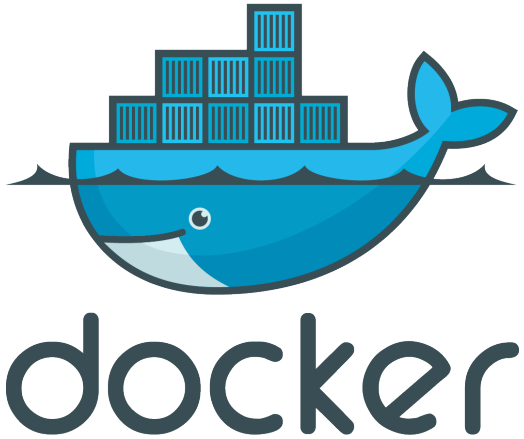
What does cloud native really mean?

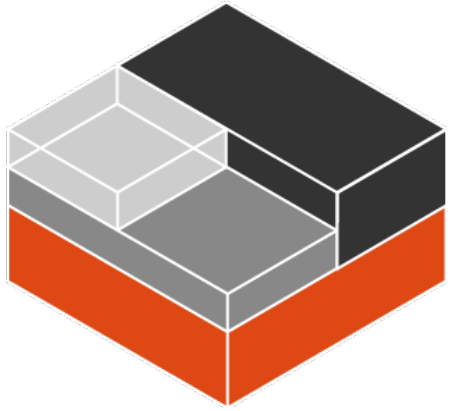
“container packaged, dynamically scheduled, and microservices based application development and operations.”

– Cloud Native Computing Foundation (CNCF)

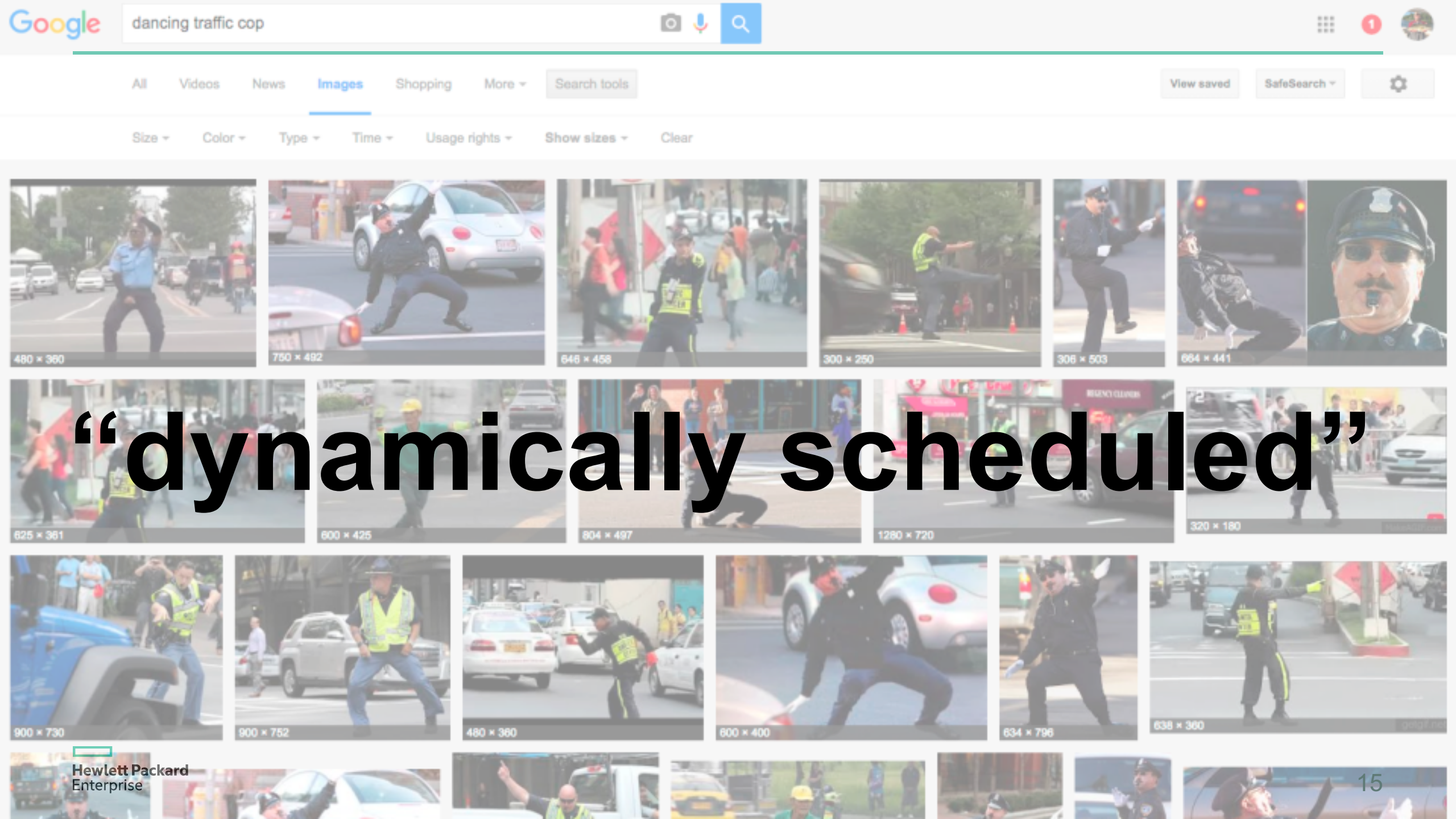

“container packaged”
By Sergio Morchon https://www.flickr.com/photos/smorchon/2431349077/

Containers isn’t just Docker…



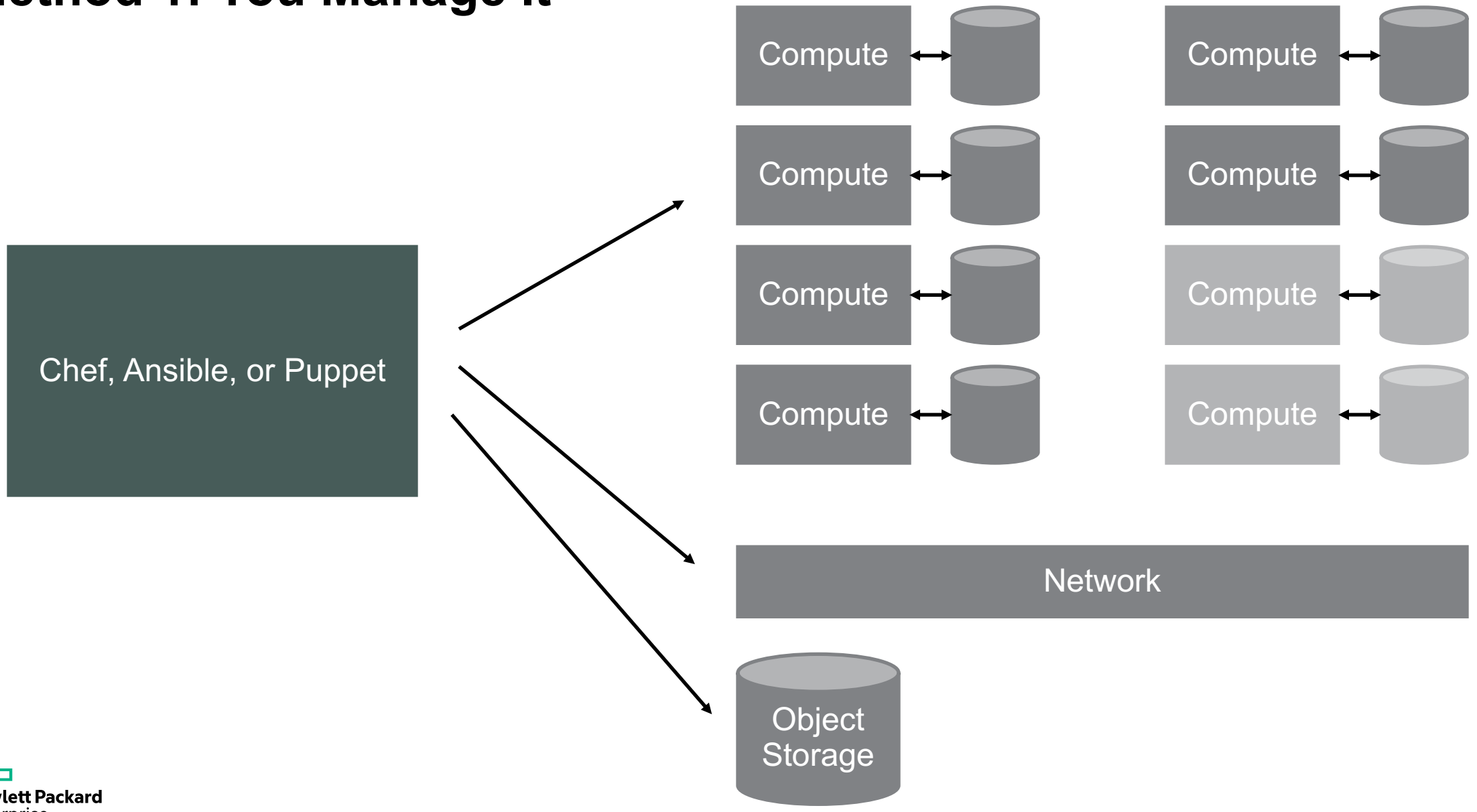


Does this mean I can’t do cloud native with VMs?



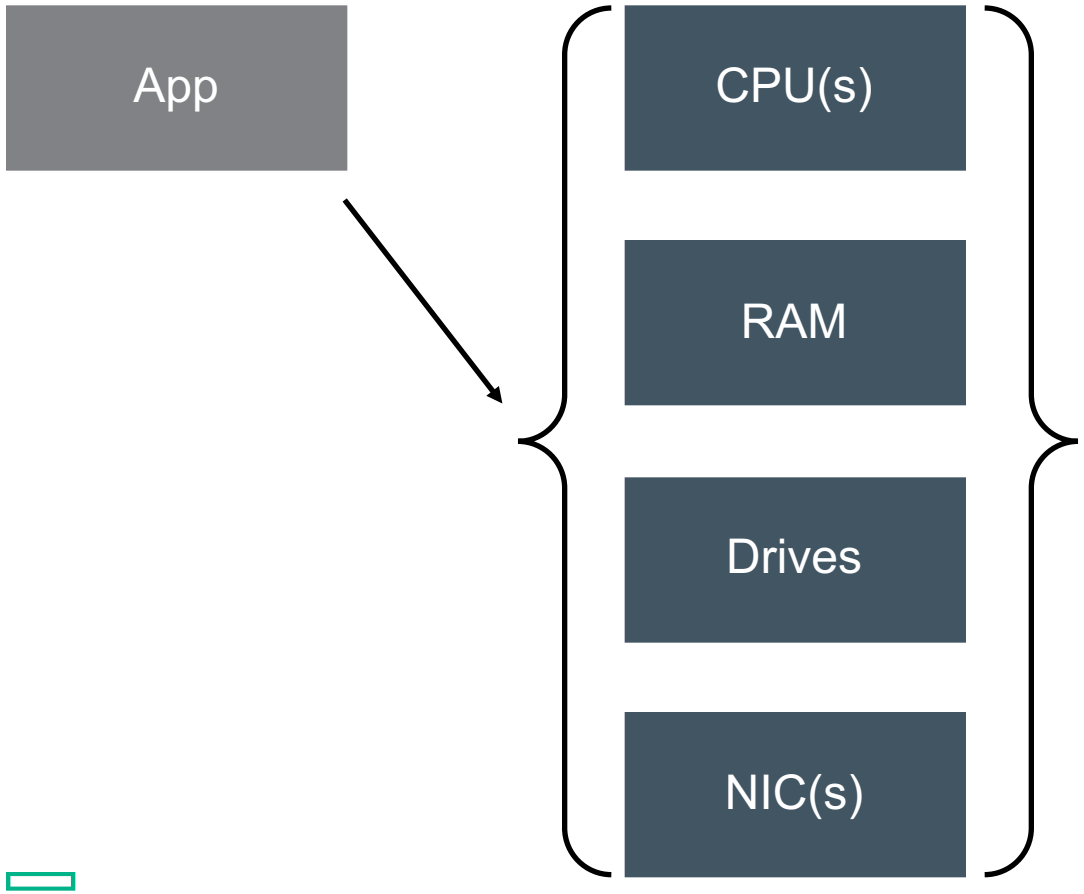
“dynamically scheduled”

Method 1: You Manage It

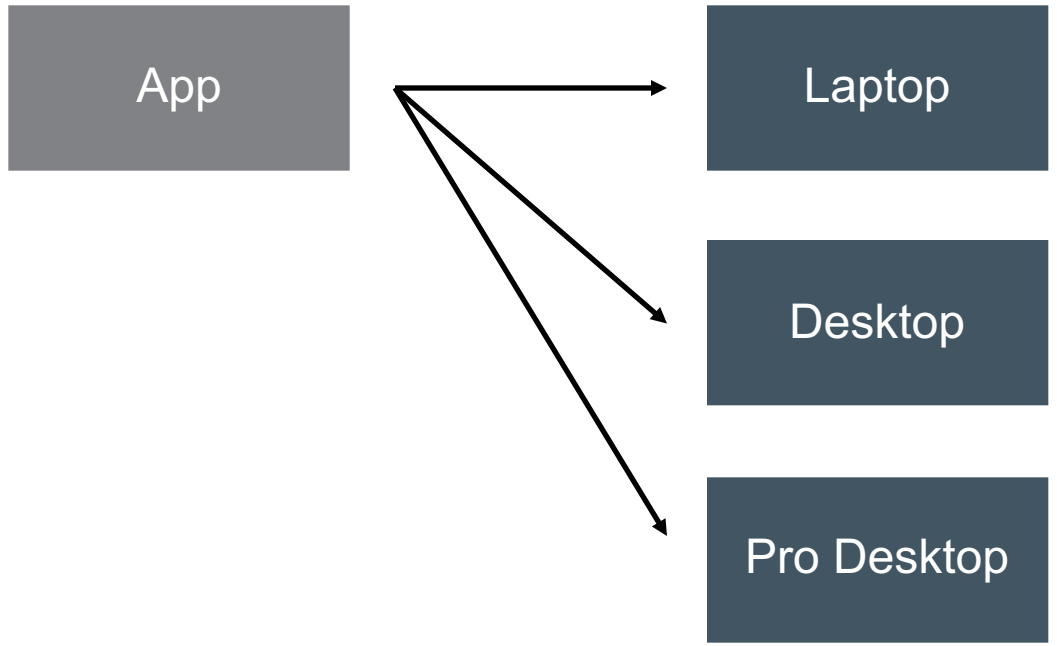


But, what way do most desktop applications work?

Choice 1

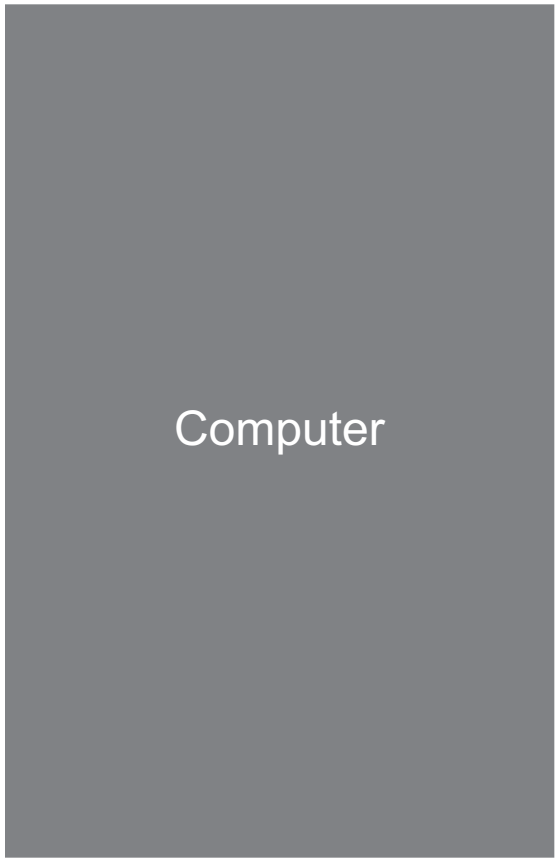


Choice B



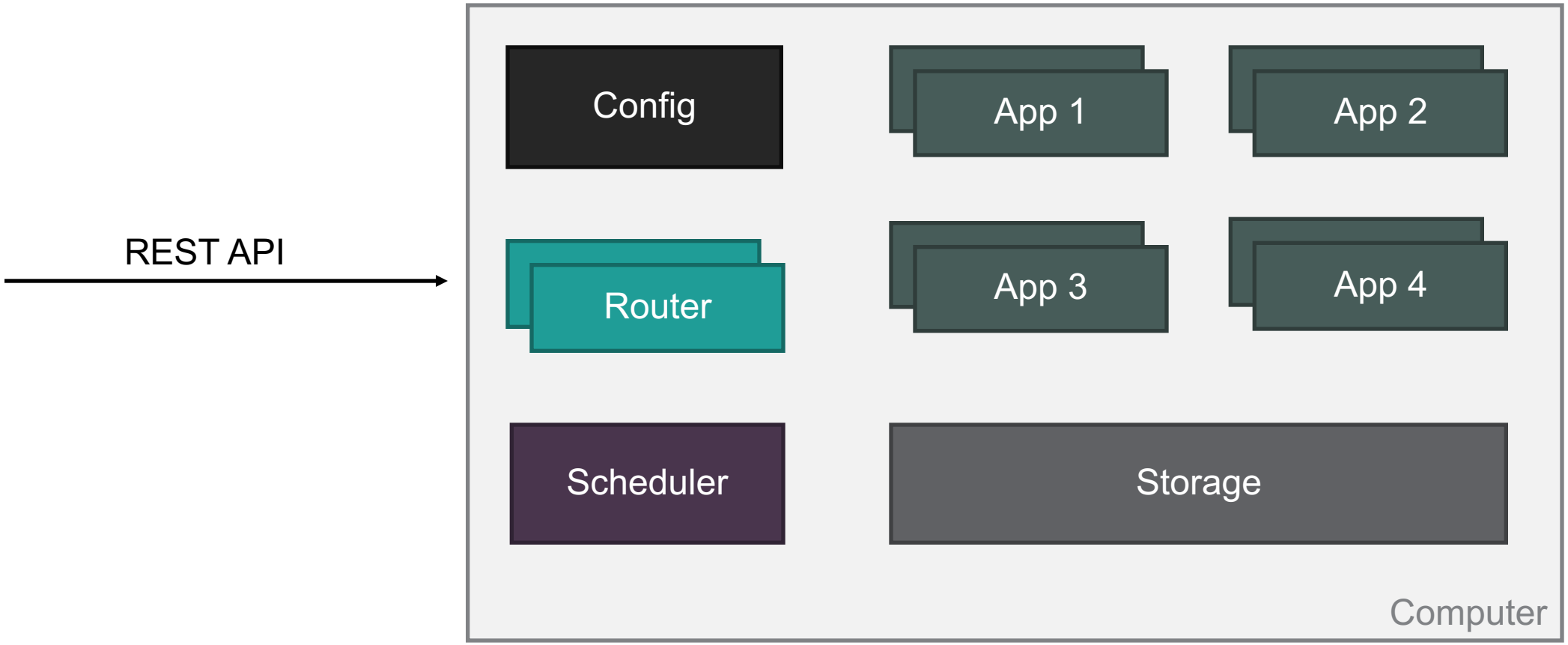
Hewlett Packard Enterprise

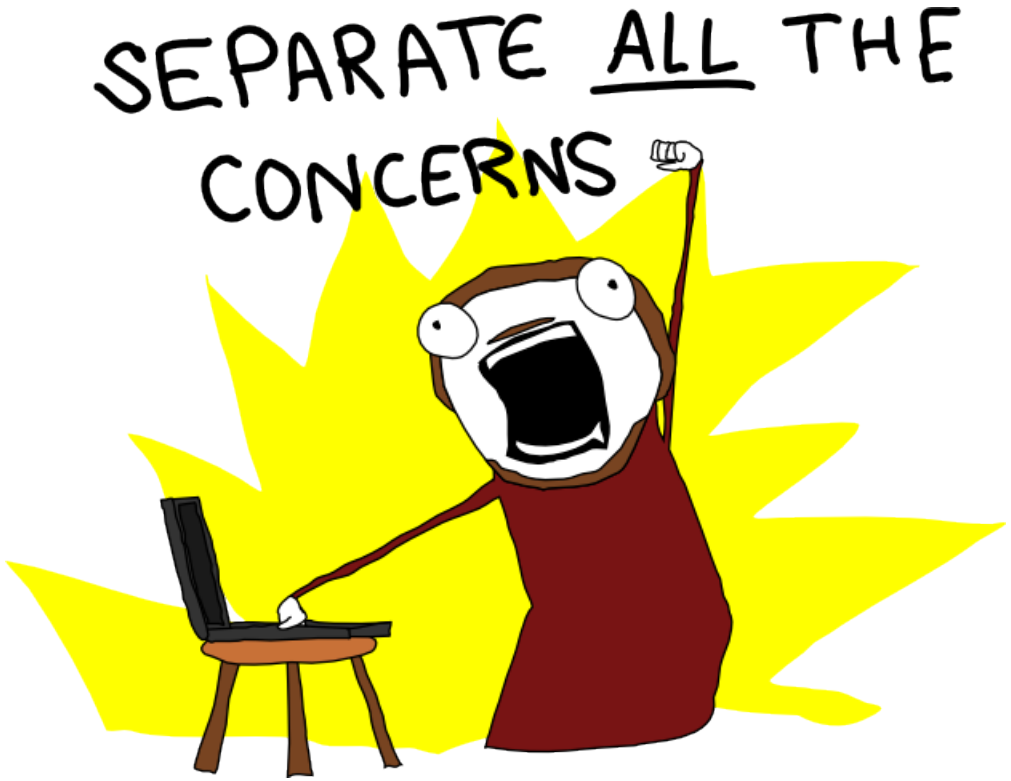
Method 2: Datacenter/Cluster as a Computer



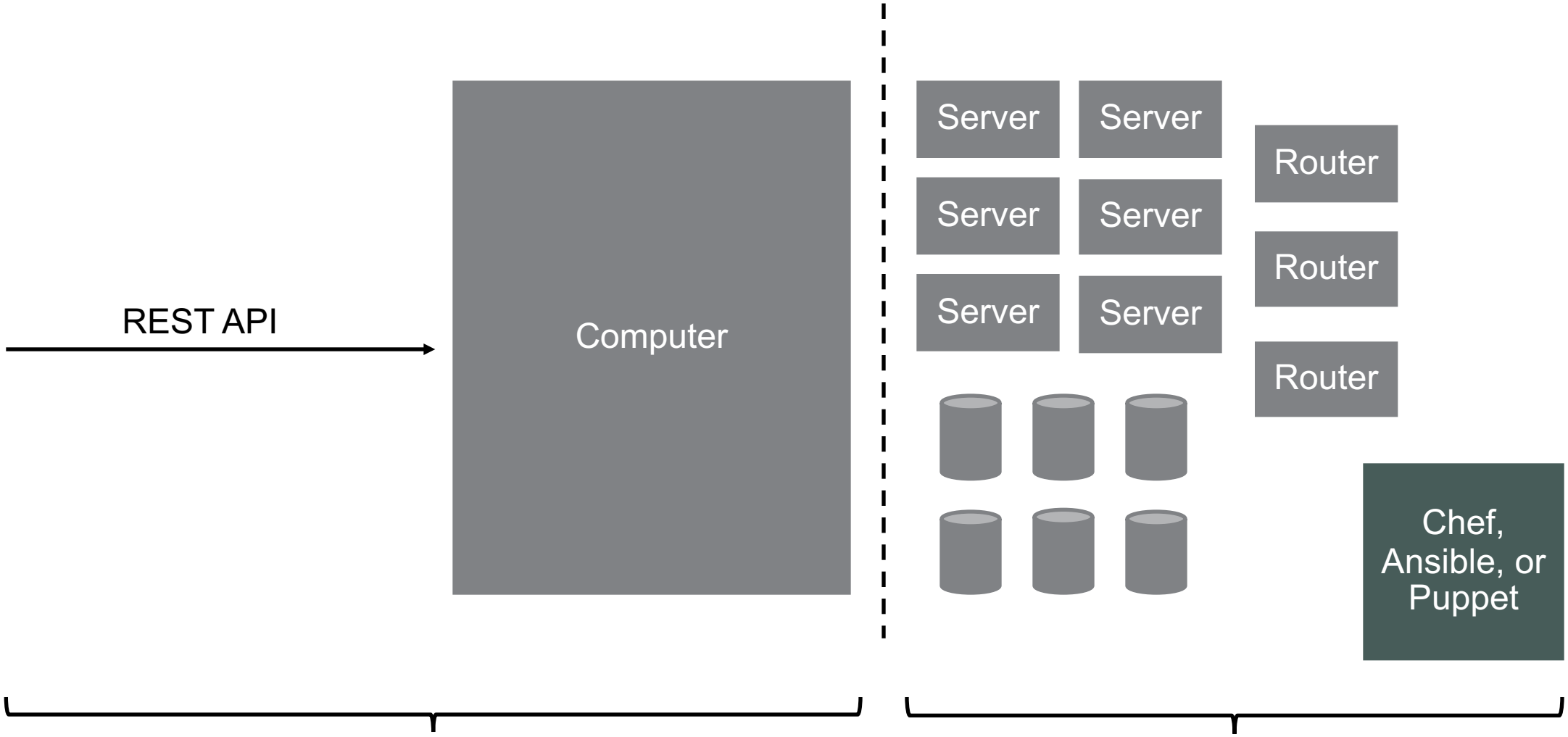
- Could be a datacenter, rack, or single computer
- Manages the needed infra and changes for your apps for you
- Think of it as one computer rather than managed collection

What's Happening Inside The Computer?

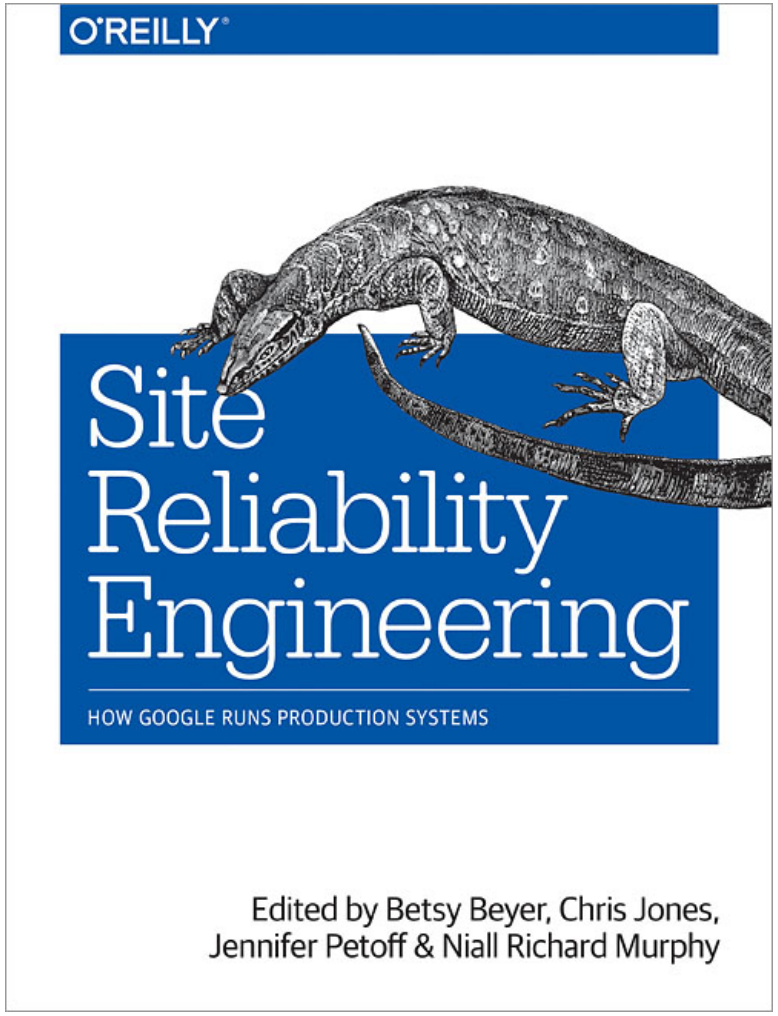


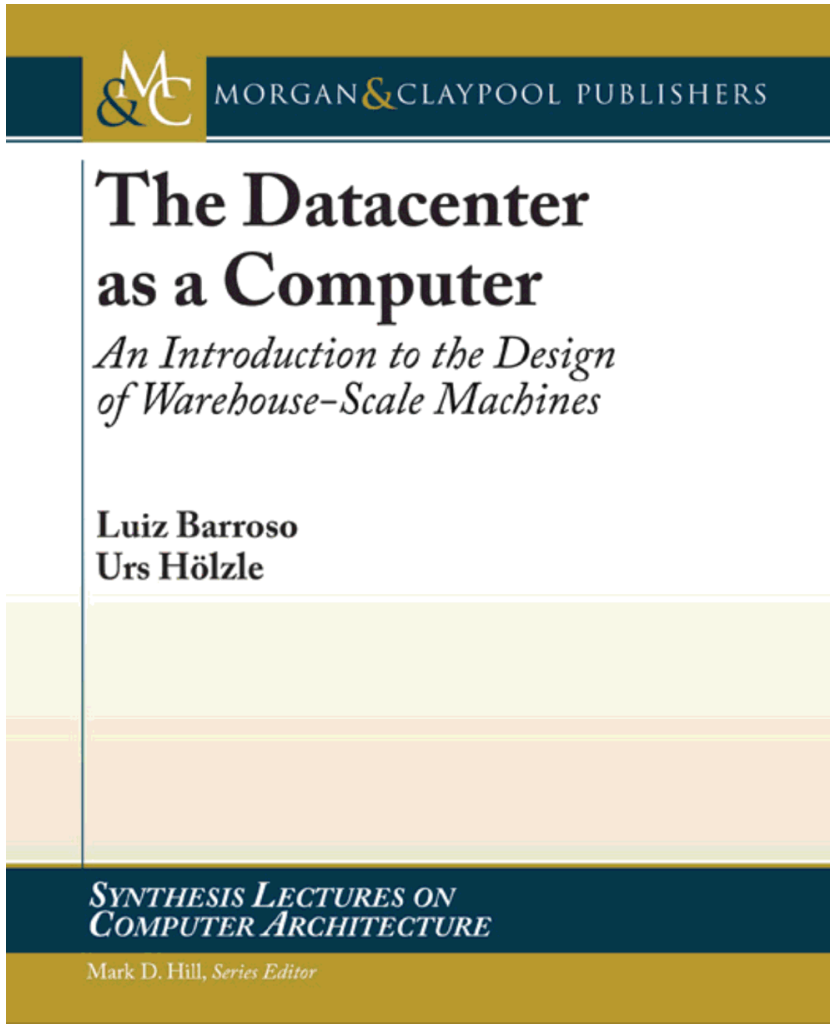

SEPARATE ALL THE
CONCERNS

Combined Truth

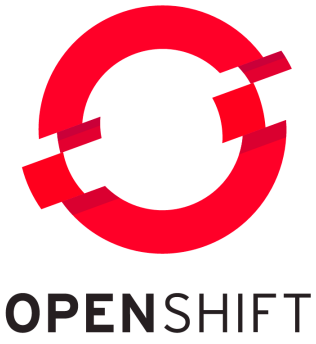


Managing Apps and a Datacenter as a Computer





What You Can Use Today



HPE Helion Stackato

heroku

“microservices based”

What is a microservice?

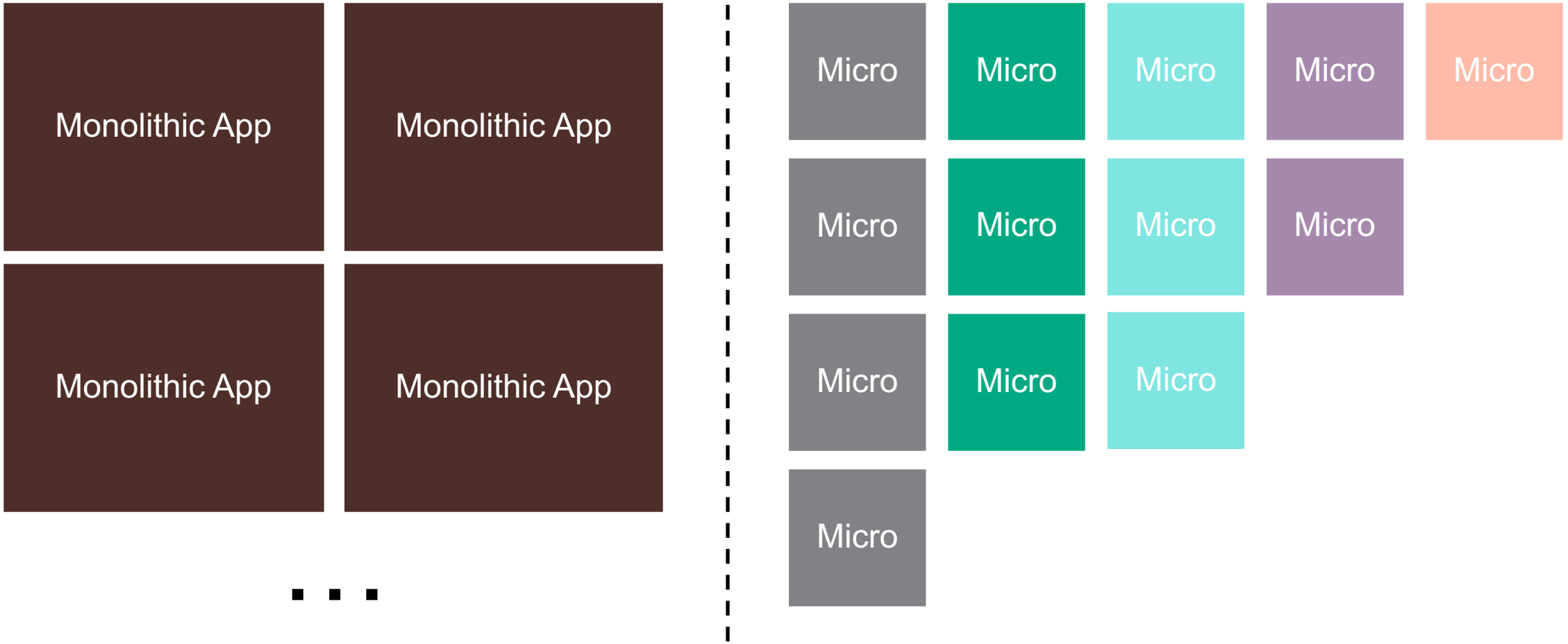
–Small

–Focused on doing one thing well (single responsibility principle)

–Independently deployable

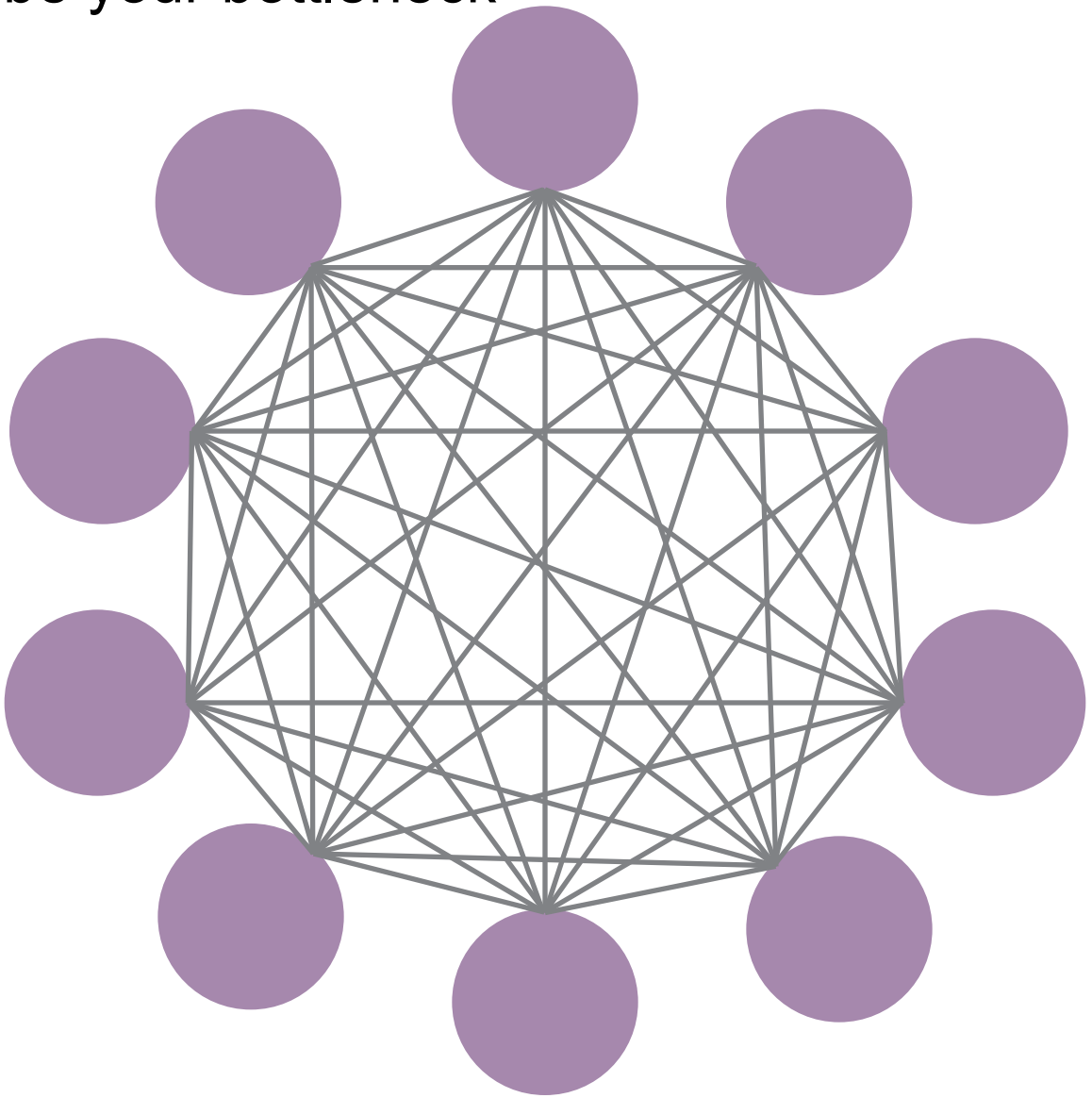
–Clearly defined API for interaction

–Potentially reusable in other systems


Monolithic App
Monolithic App
Monolithic App
Monolithic App
Micro
Micro
Micro
Micro
Micro
Micro
Micro
Micro
Micro
Micro
Micro
Micro
Micro

Potential Network Problem

Don’t let the network be your bottleneck



What makes a good REST API?

– Versioned, typically in the URL

– Use proper HTTP methods

– Behind Authentication and Authorization

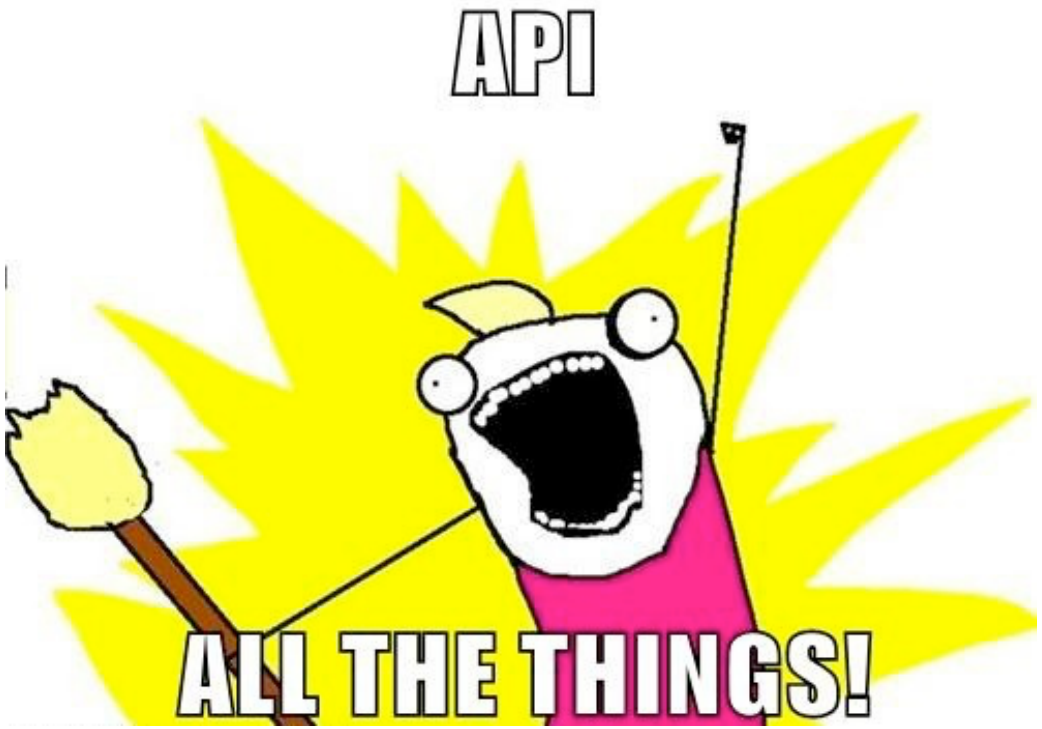
– HTTP/2 if possible (for pipelines and connection reuse)

– TLS/HTTPS (encrypted transport)

– Proper HTTP response codes

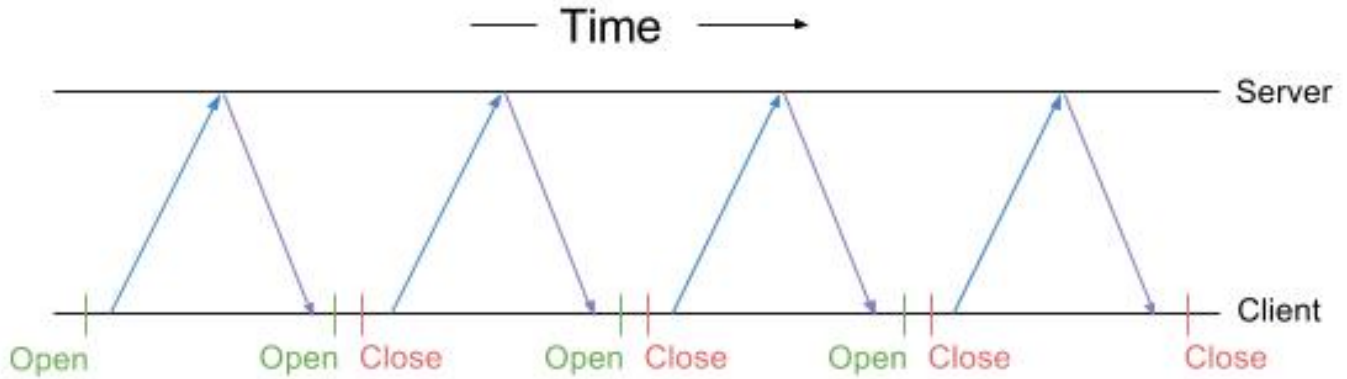
– JSON

– Open API Initiative (Swagger)

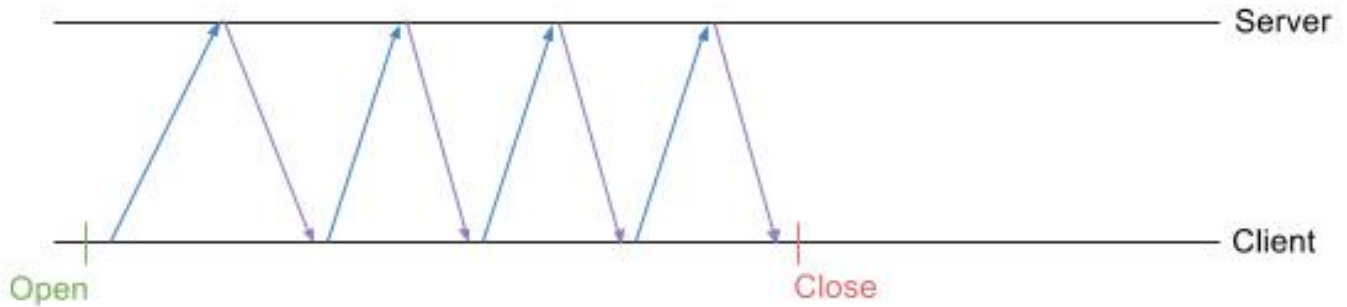


Reuse connections

Each request using a different connection

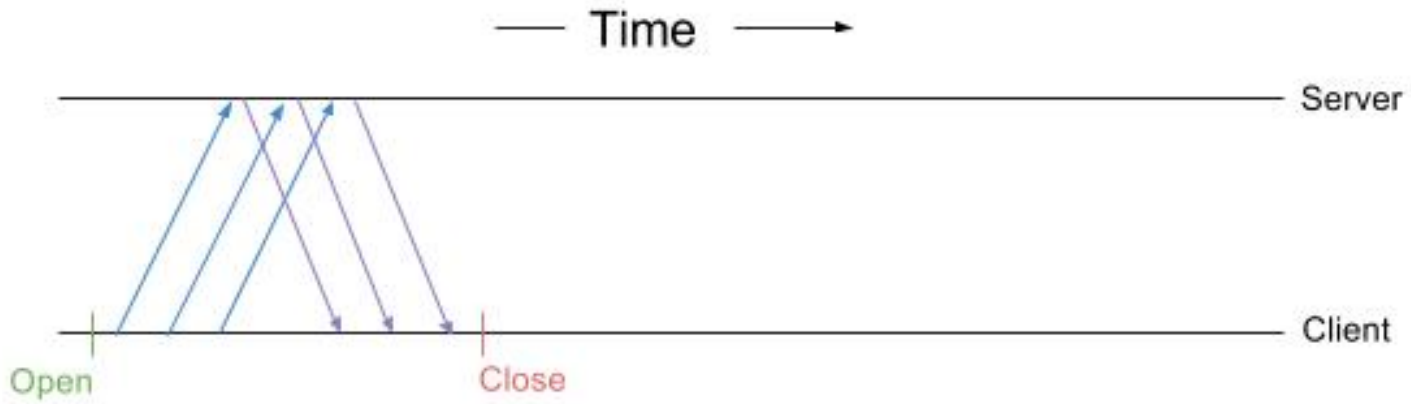


All requests sharing the same connection

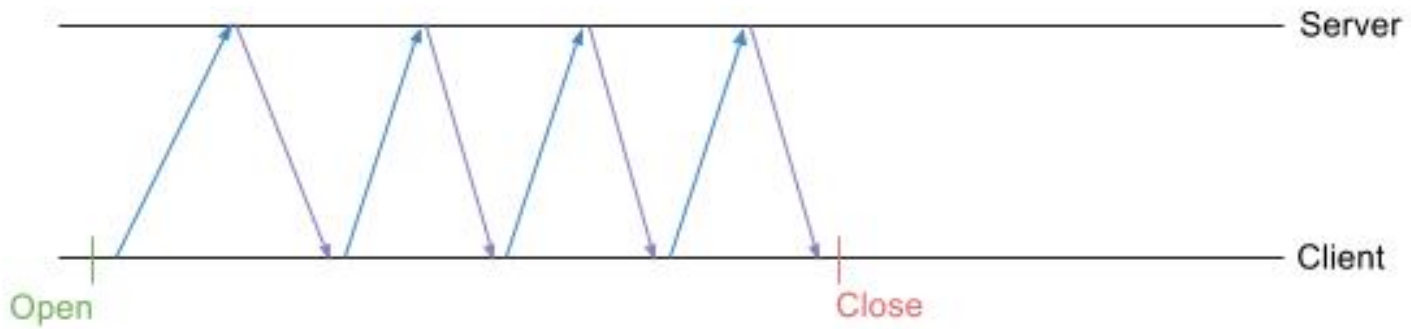


HTTP/2 and Pipelining

Pipelining, where requests are concurrent



Requests in serial, one after another

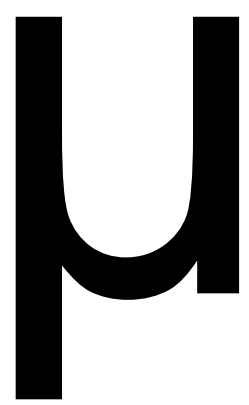


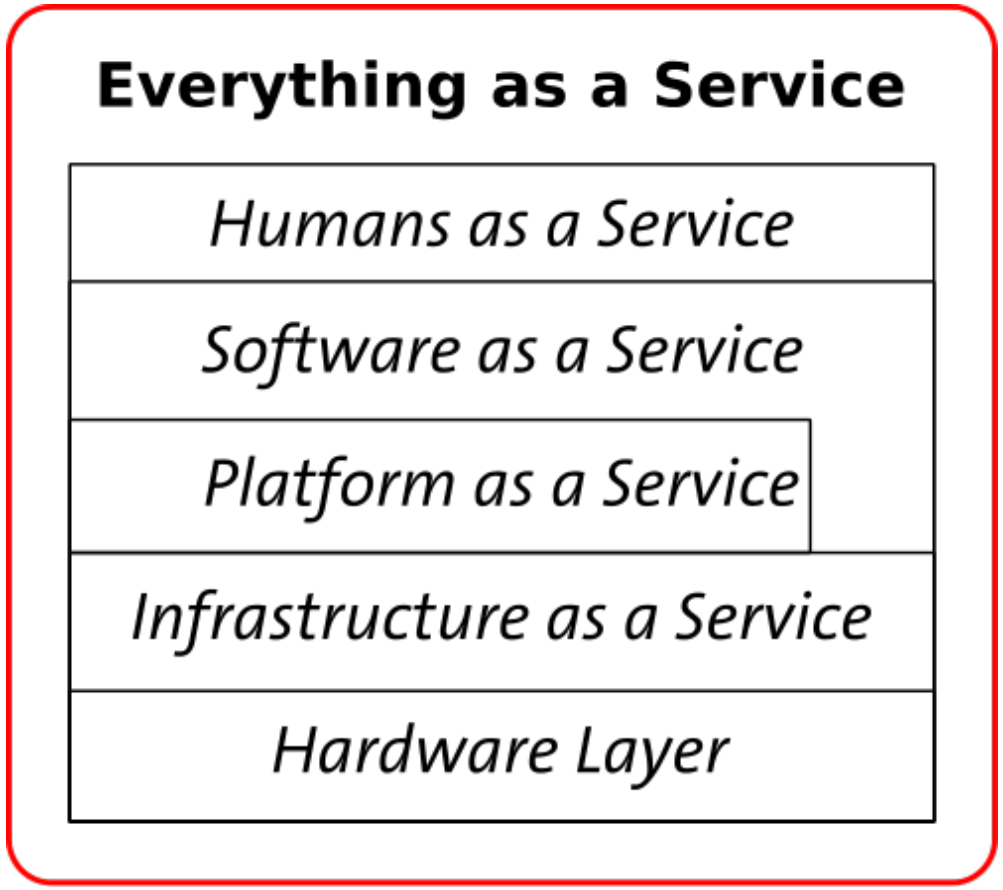
12 Factor++ (12factor.net)

1. One codebase tracked in revision control, many deploys
2. Explicitly declare and isolate dependencies
3. Store config in the environment
4. Treat backing services as attached resources
5. Strictly separate build and run stages
6. Execute the app as one or more typically stateless ~~processes~~ containers
7. Export services via port binding
8. Scale out via ~~the process model~~ containers
9. Maximize robustness with fast startup and graceful shutdown
10. Keep development, staging, and production as similar as possible
11. Treat logs as event streams
12. Run admin/management tasks as one-off ~~processes~~ containers

Store config in the environment

Treat backing services as attached resources



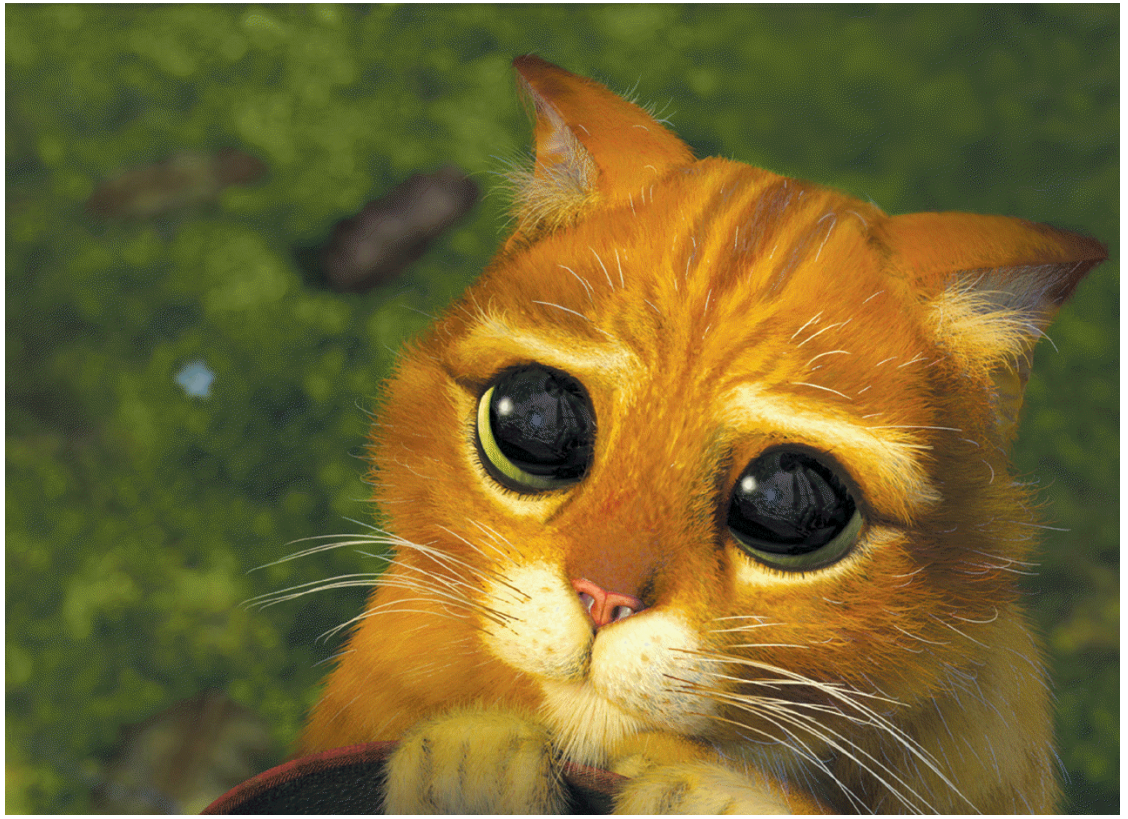


Execute the app as one or more ~~stateless~~ processes

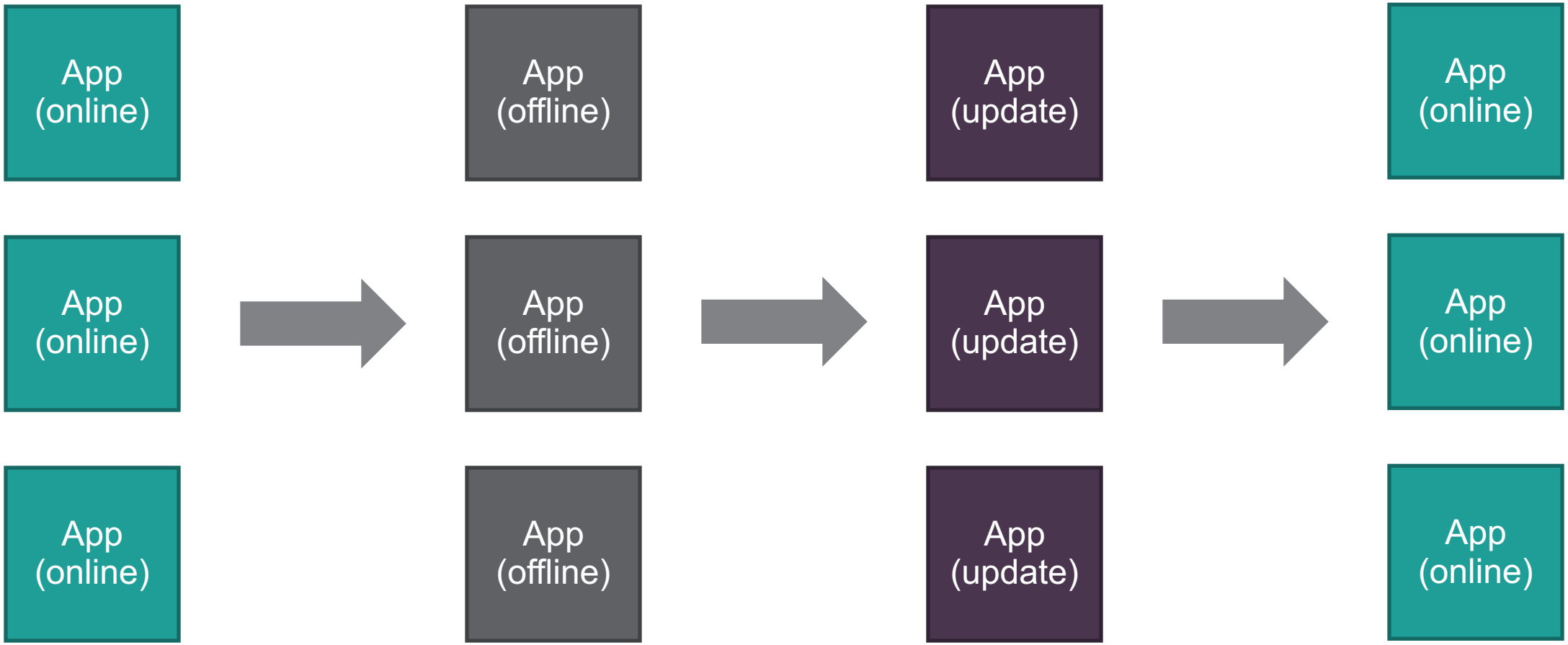
Hewlett Packard
Enterprise

Treat logs as event streams (~~stdout~~)

Pets vs cattle

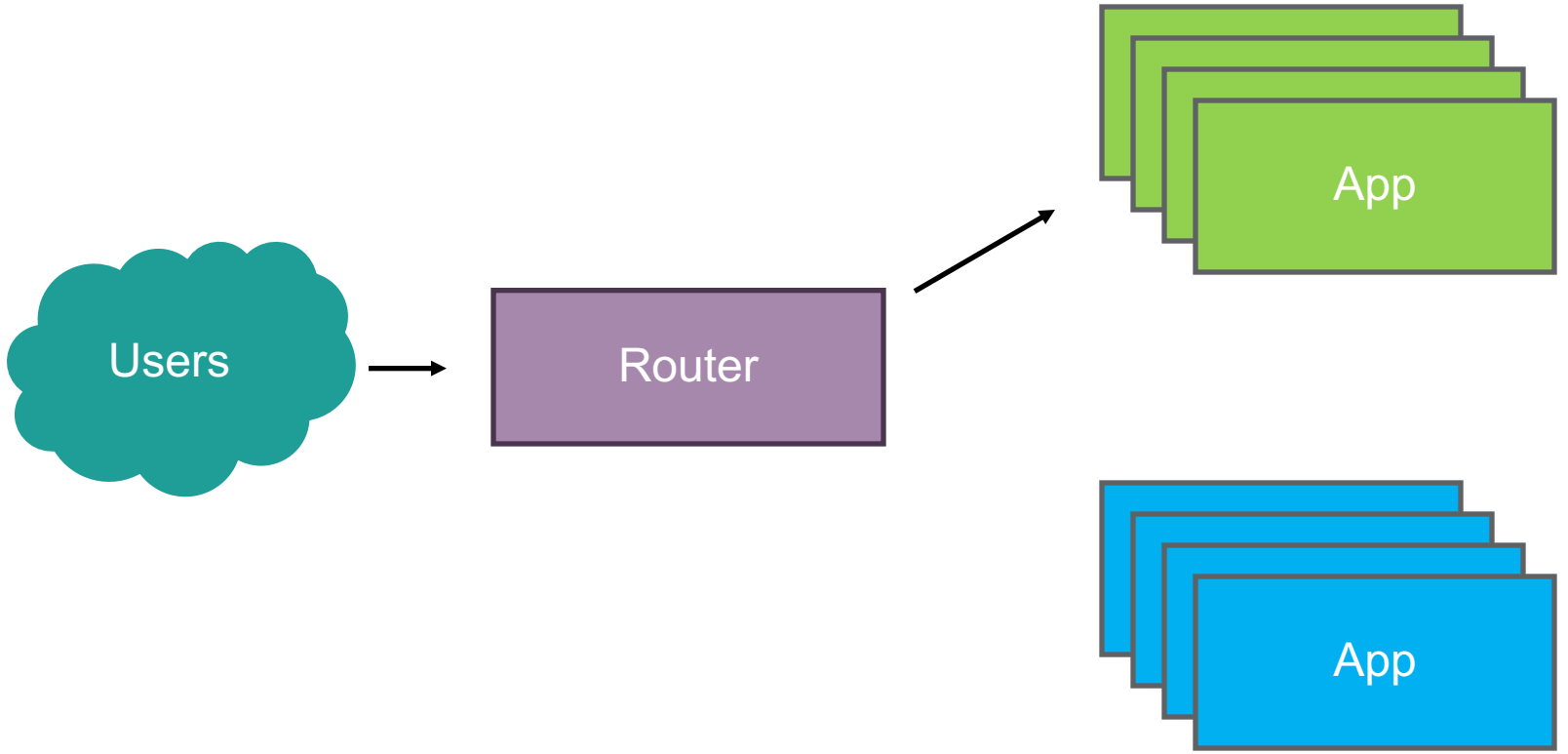


Updating Applications


App (online)
App (online)
App (online)
App (offline)
App (offline)
App (offline)
App (update)
App (update)
App (update)
App (online)
App (online)
App (online)

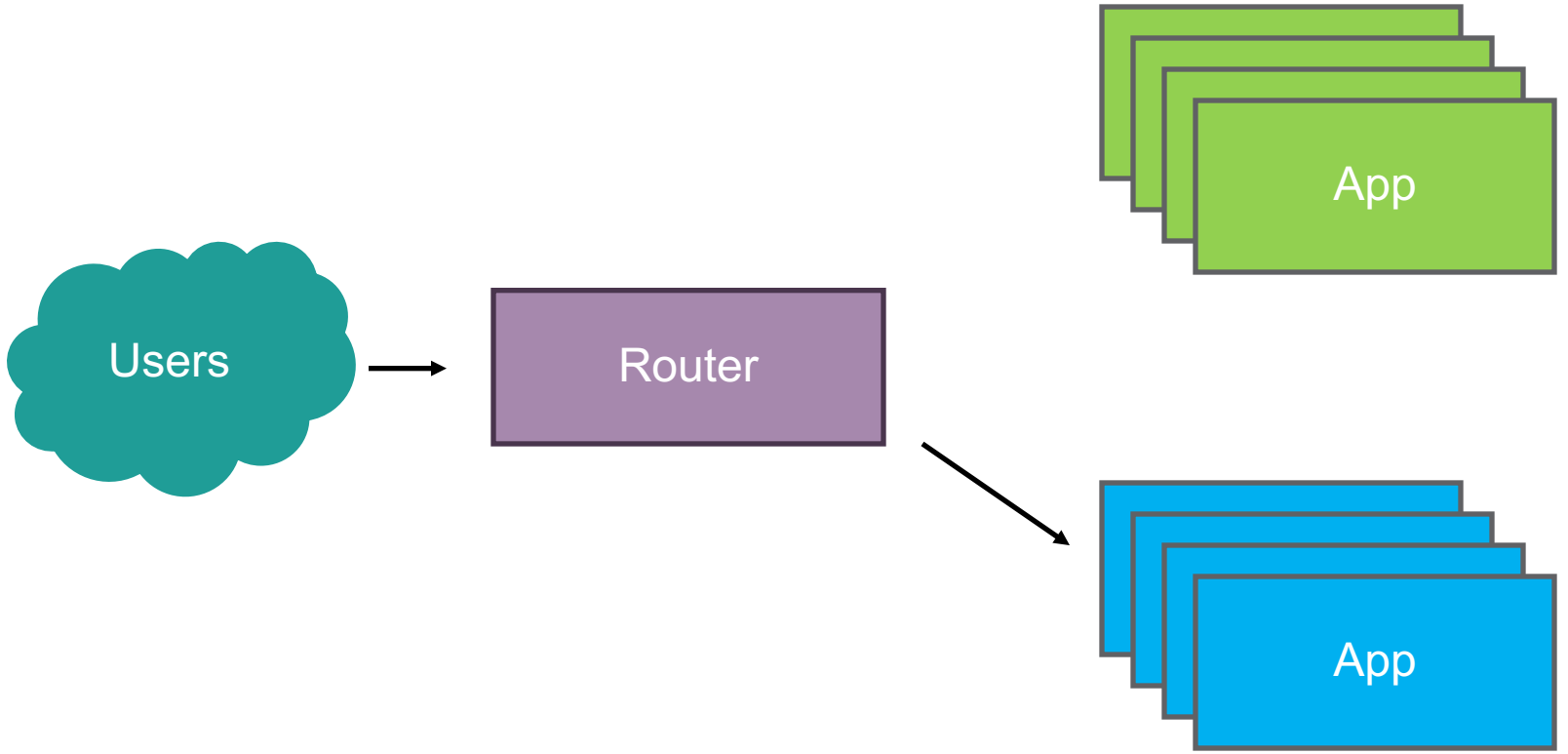
Blue / Green Deployments

Start with Green

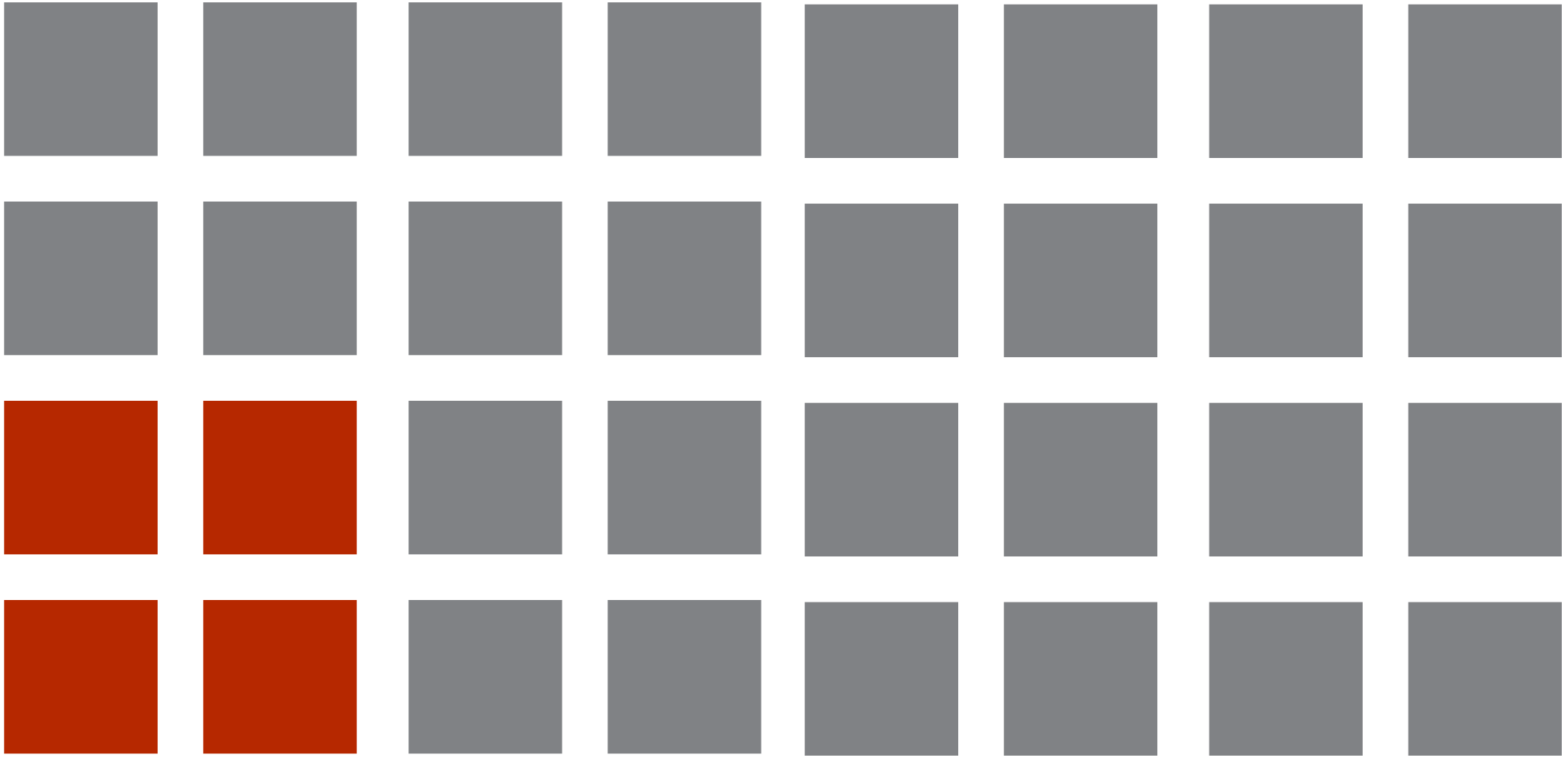


Blue / Green Deployments

Switch to Blue

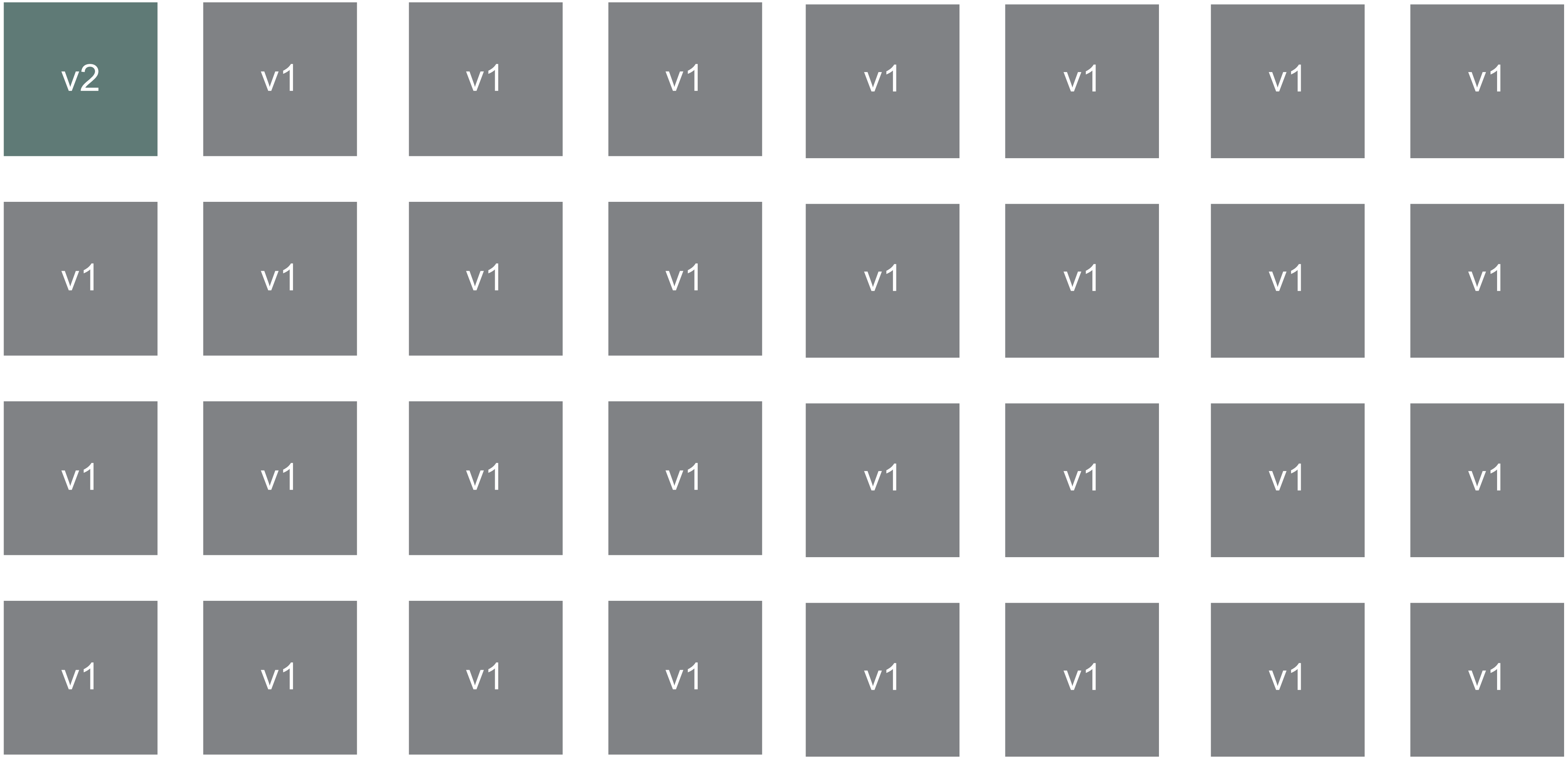


Canary Release



Rolling Updates

Rolling Updates



Rolling Updates

Rolling Updates

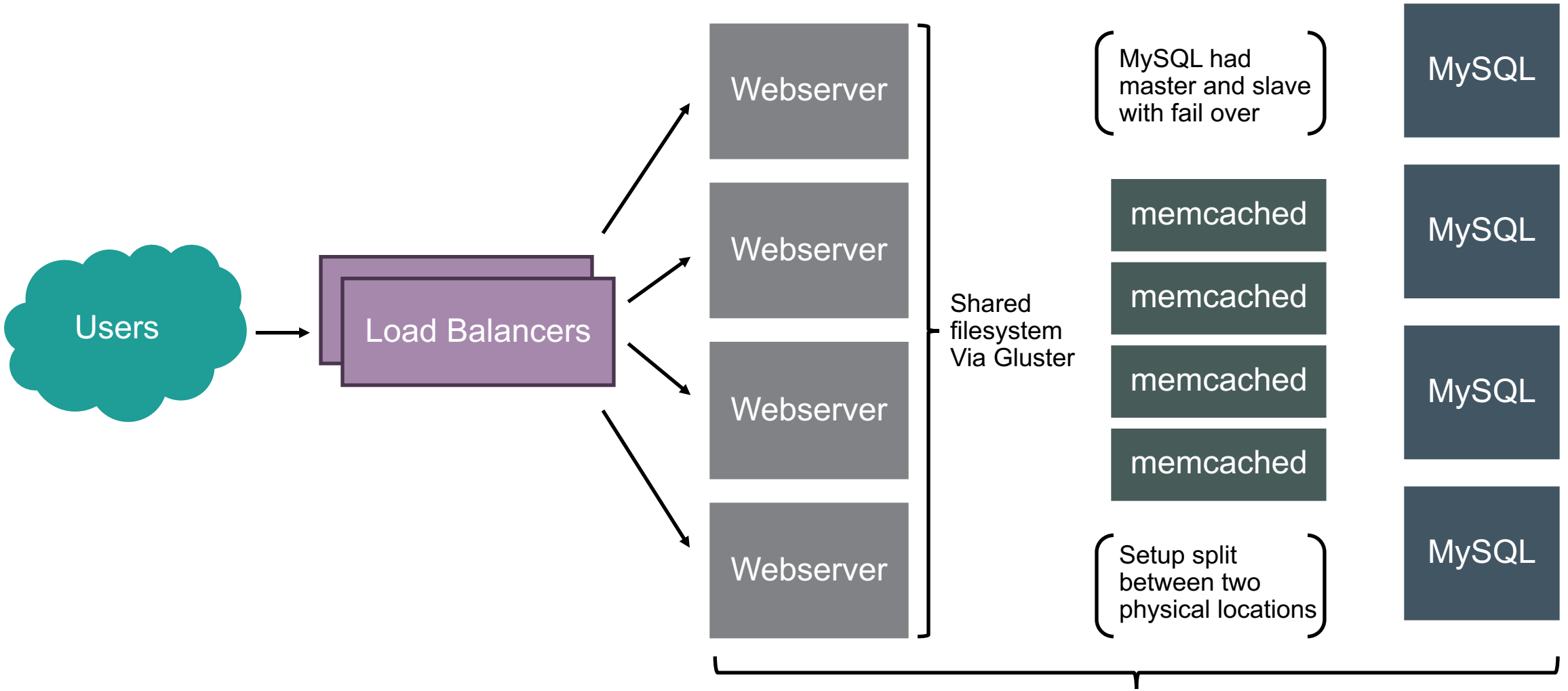
3 Paths To Cloud Native:

1. Migration
2. Greenfield
3. Additive

Path 1: Migrating

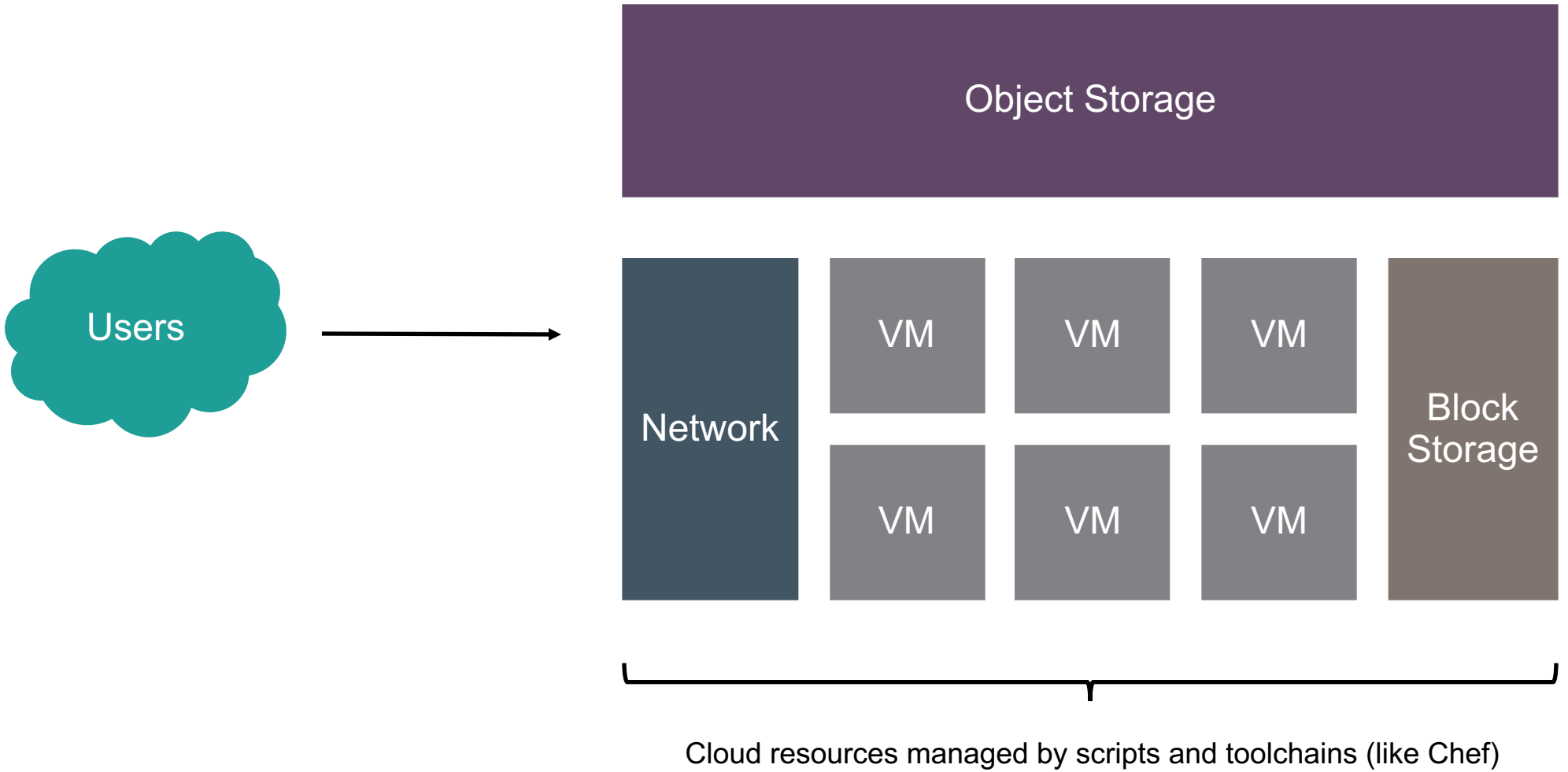
My Web App Before The Cloud

This may look familiar



3x Web Apps
+ more coming

… at the same time... Experiments in the cloud




CLOUD
FOUNDRY™


stackato®
by ActiveState®

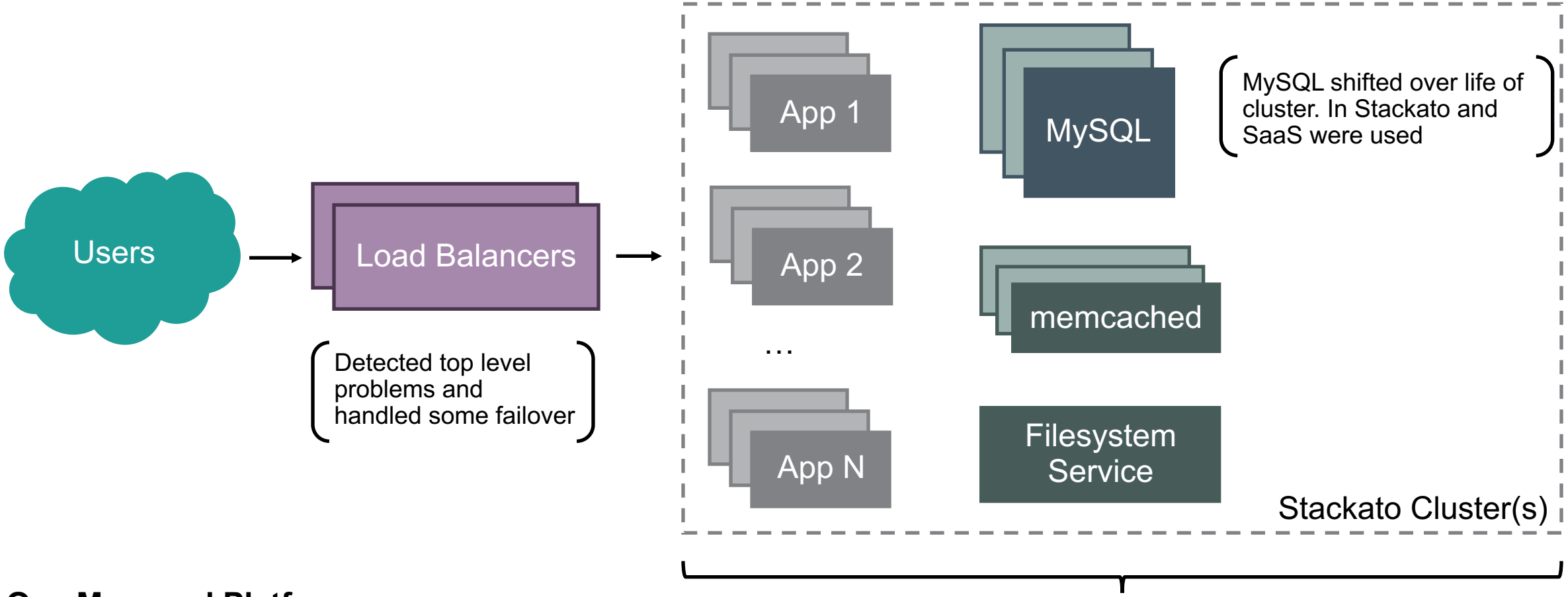
aPaaS

Development Platform

HPE Helion Stackato

Stackato Clusters

Production, QA, and Development Clusters

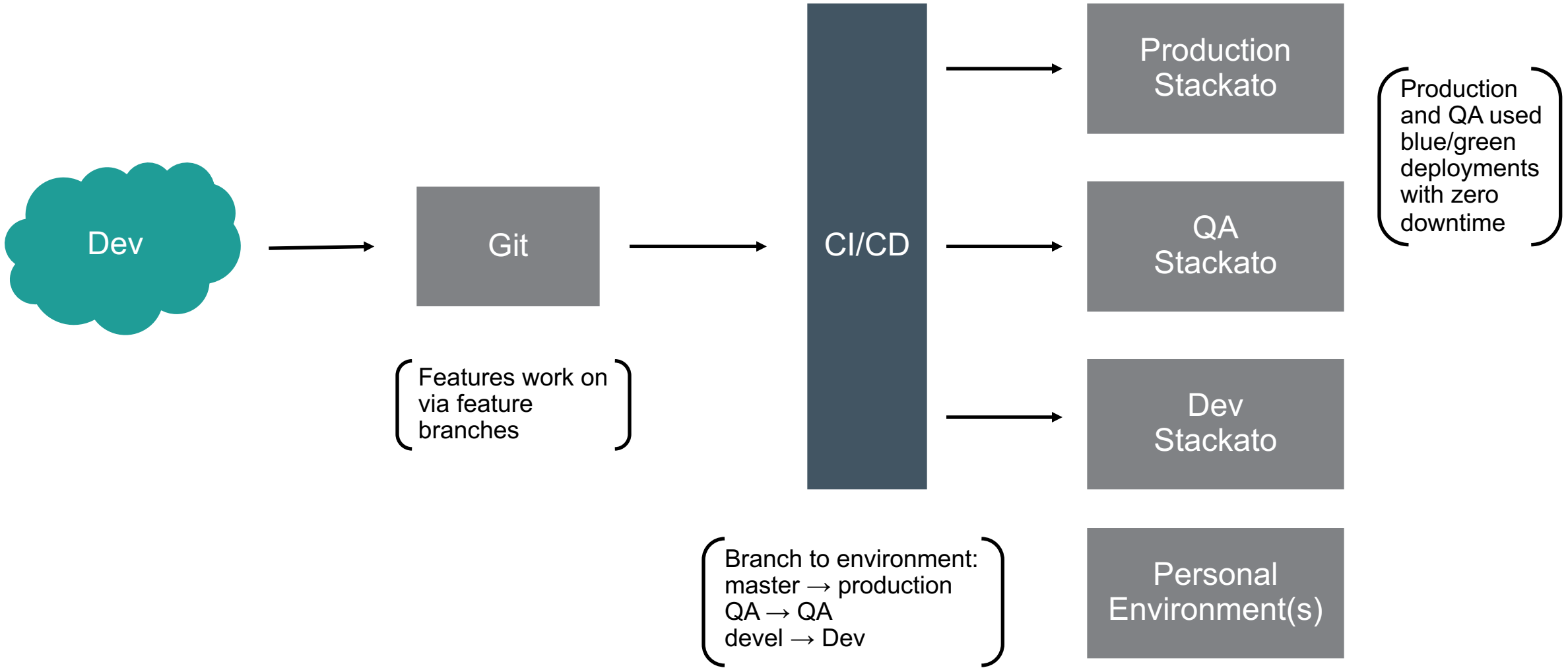


Ops Managed Platform

Automation and scripts managed the clusters running on a IaaS (backed by VMs, Networking, Block Storage, and Object Storage).

Stackato Clusters

Dev Process



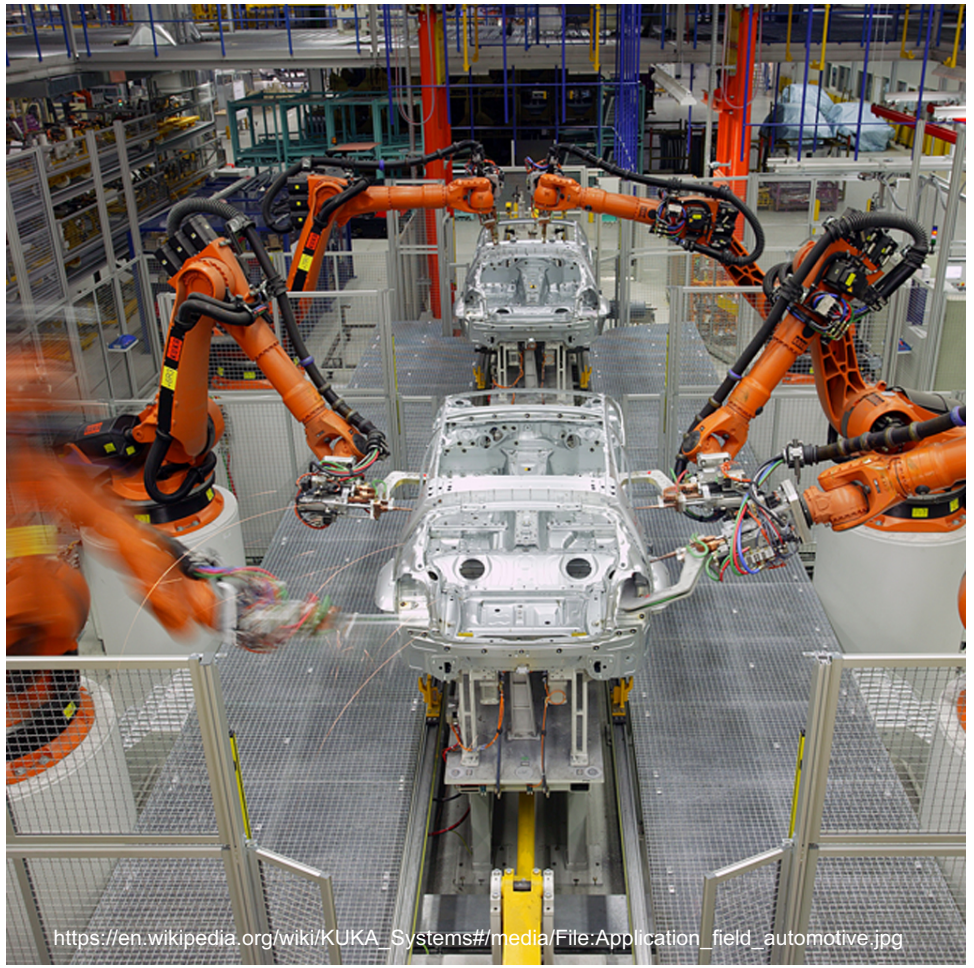
Hewlett Packard Enterprise

Continuous Integration is about trust

https://commons.wikimedia.org/wiki/File:CISV_trust_game.JPG

Continuous Integration brings reproducibility

https://en.wikipedia.org/wiki/Assembly_line#/media/File:Ford_assembly_line_-_1913.jpg



https://en.wikipedia.org/wiki/KUKA_Systems#/media/File:Application_field_automotive.jpg

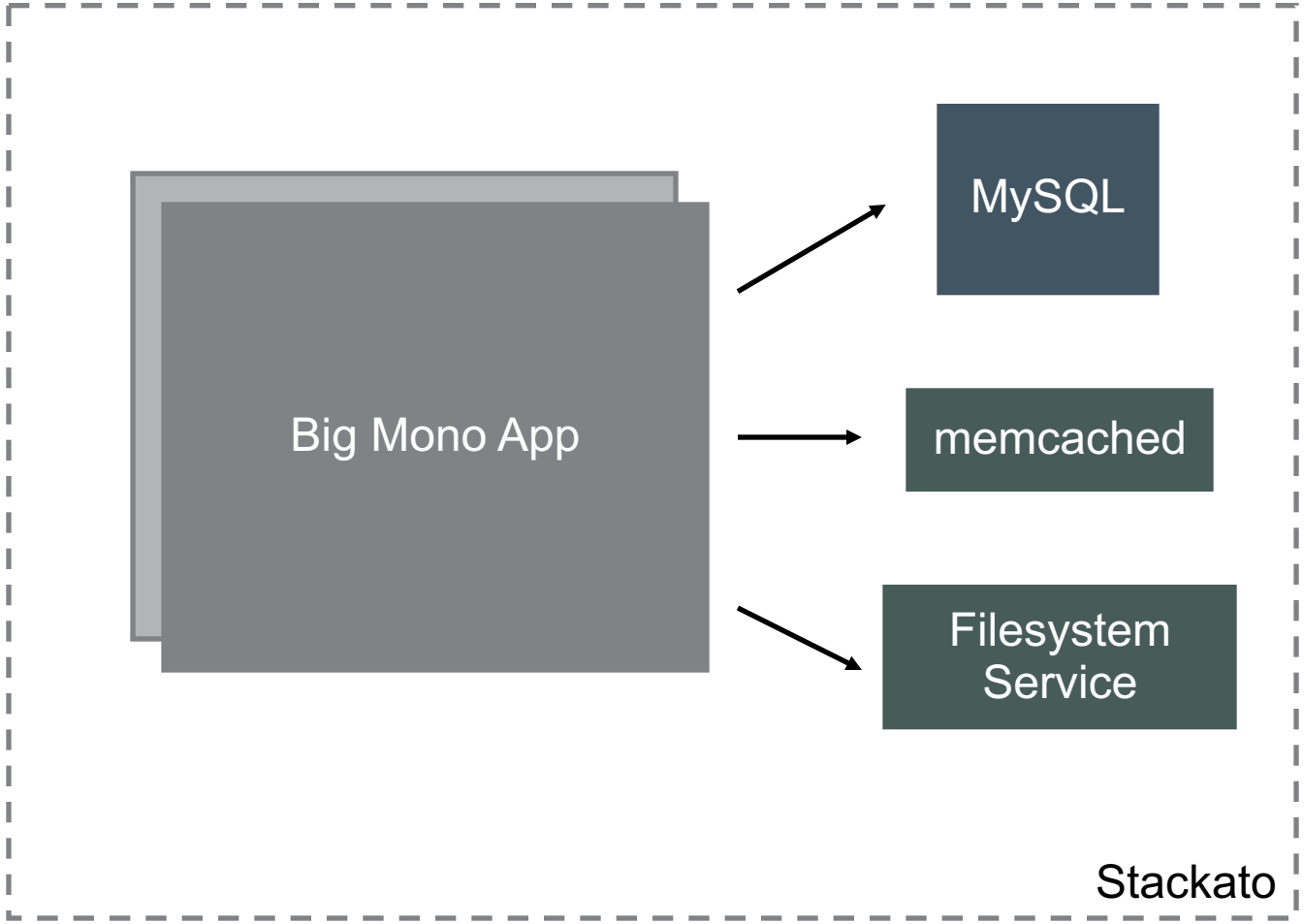
Hewlett Packard Enterprise

What is CD in CI/CD?

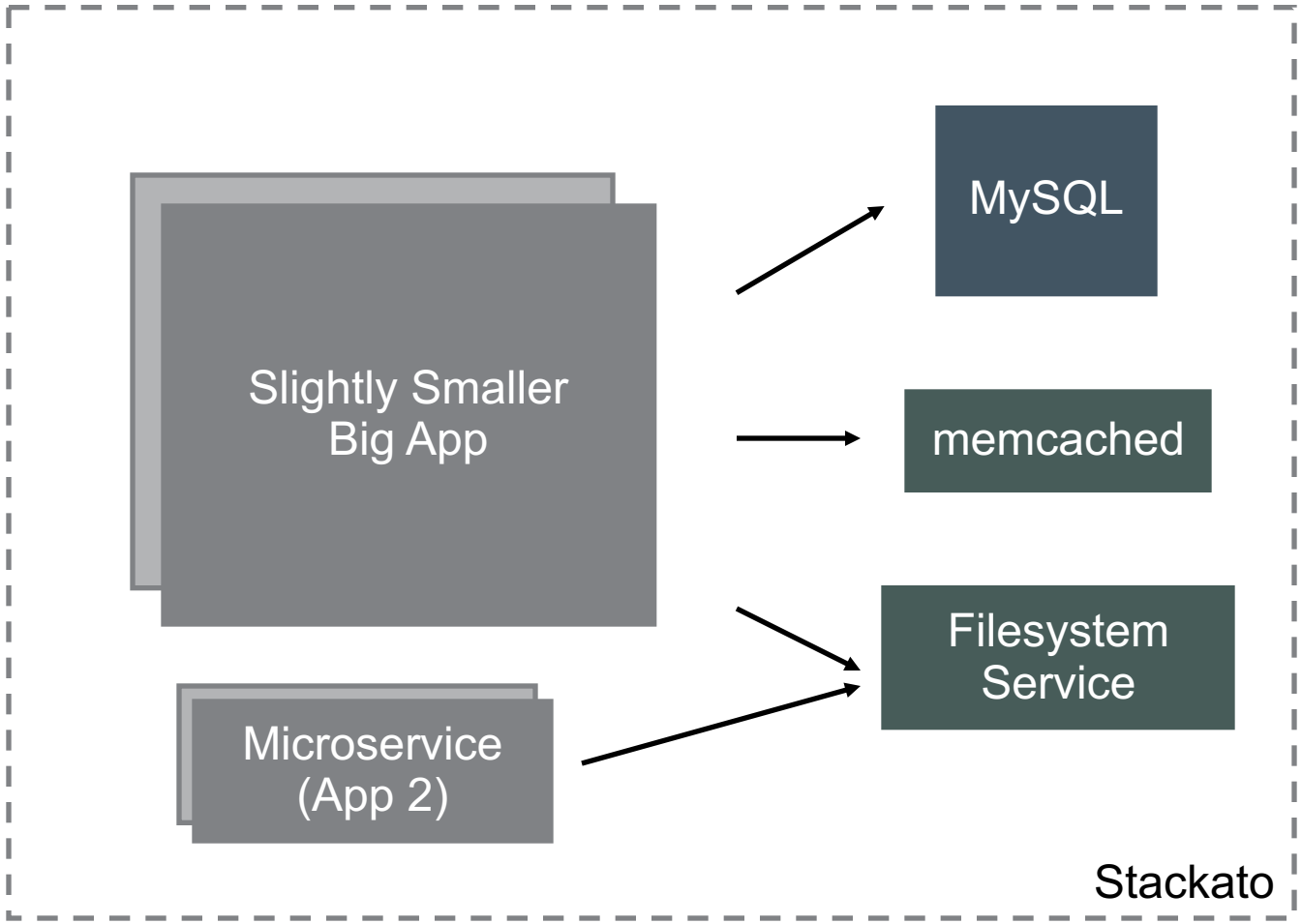
Continuous Delivery

Continuous Deployment

Started With A Mono App



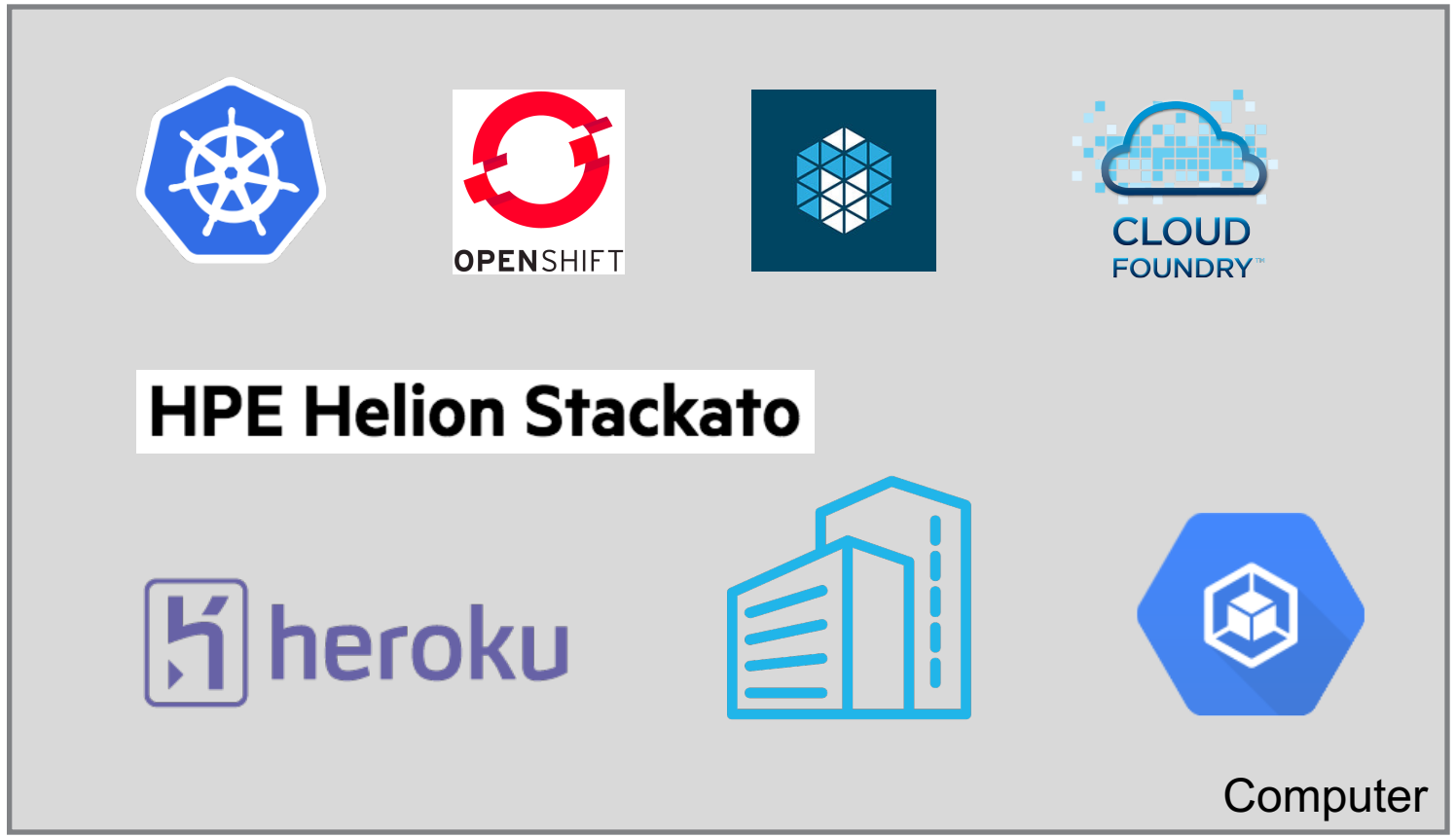
Started With A Mono App



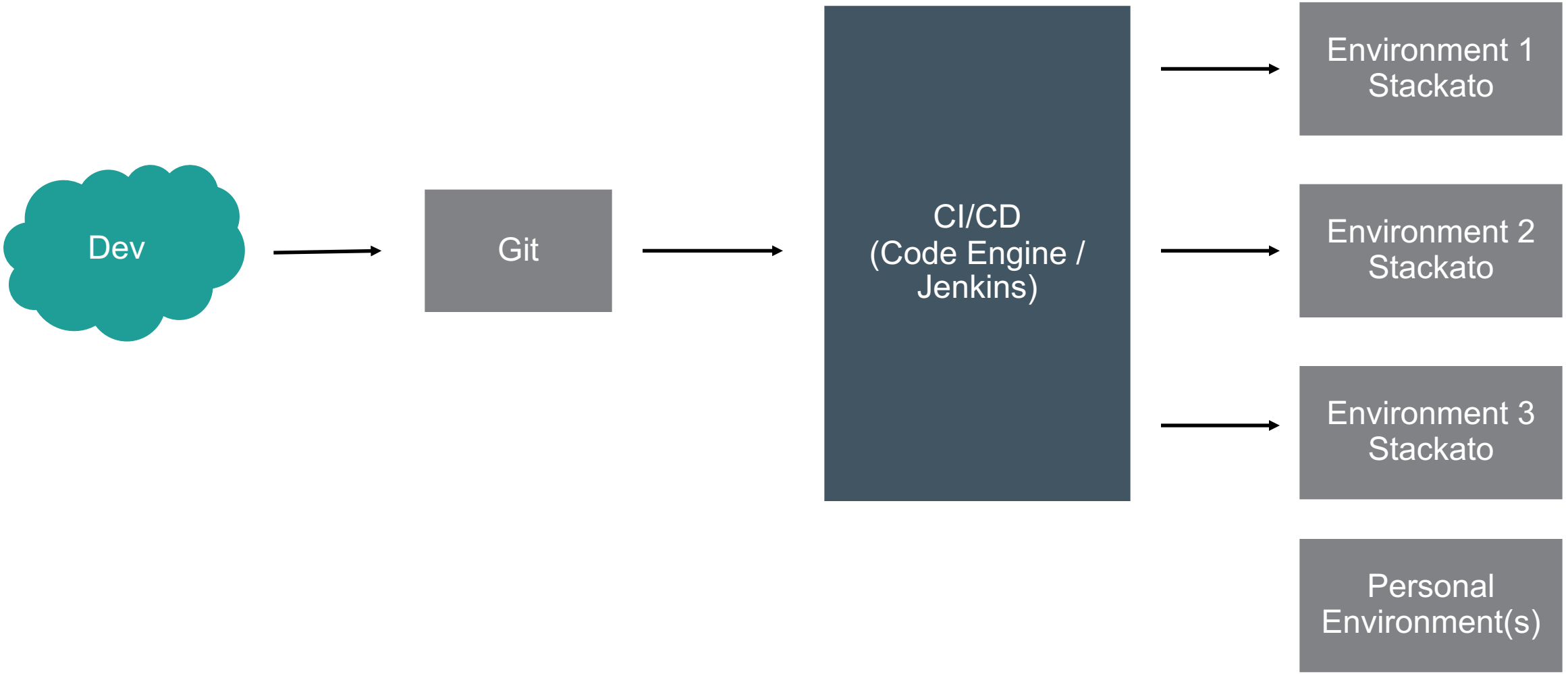
Path 2: Greenfield

Start With Your Setup

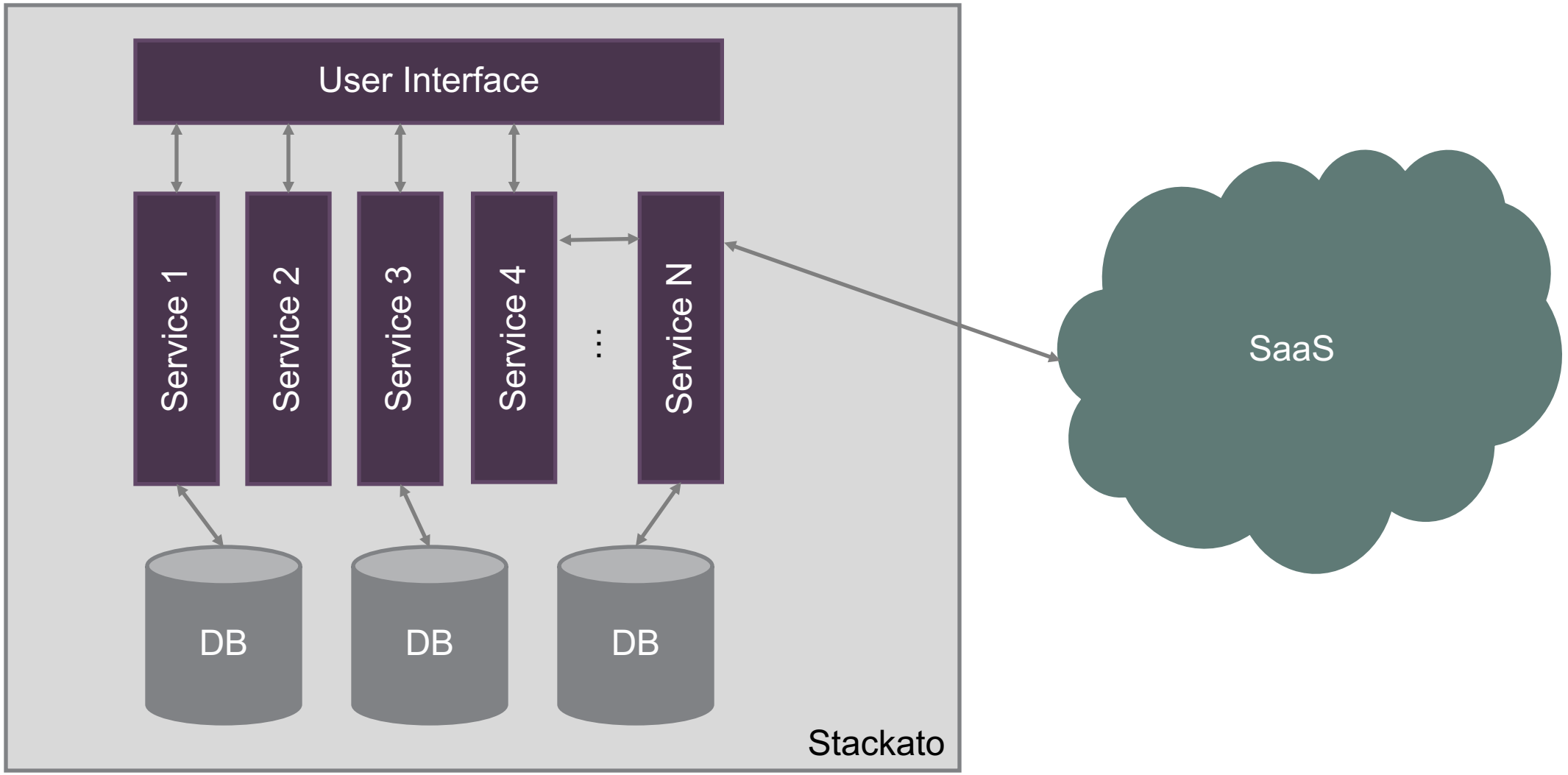
REST API



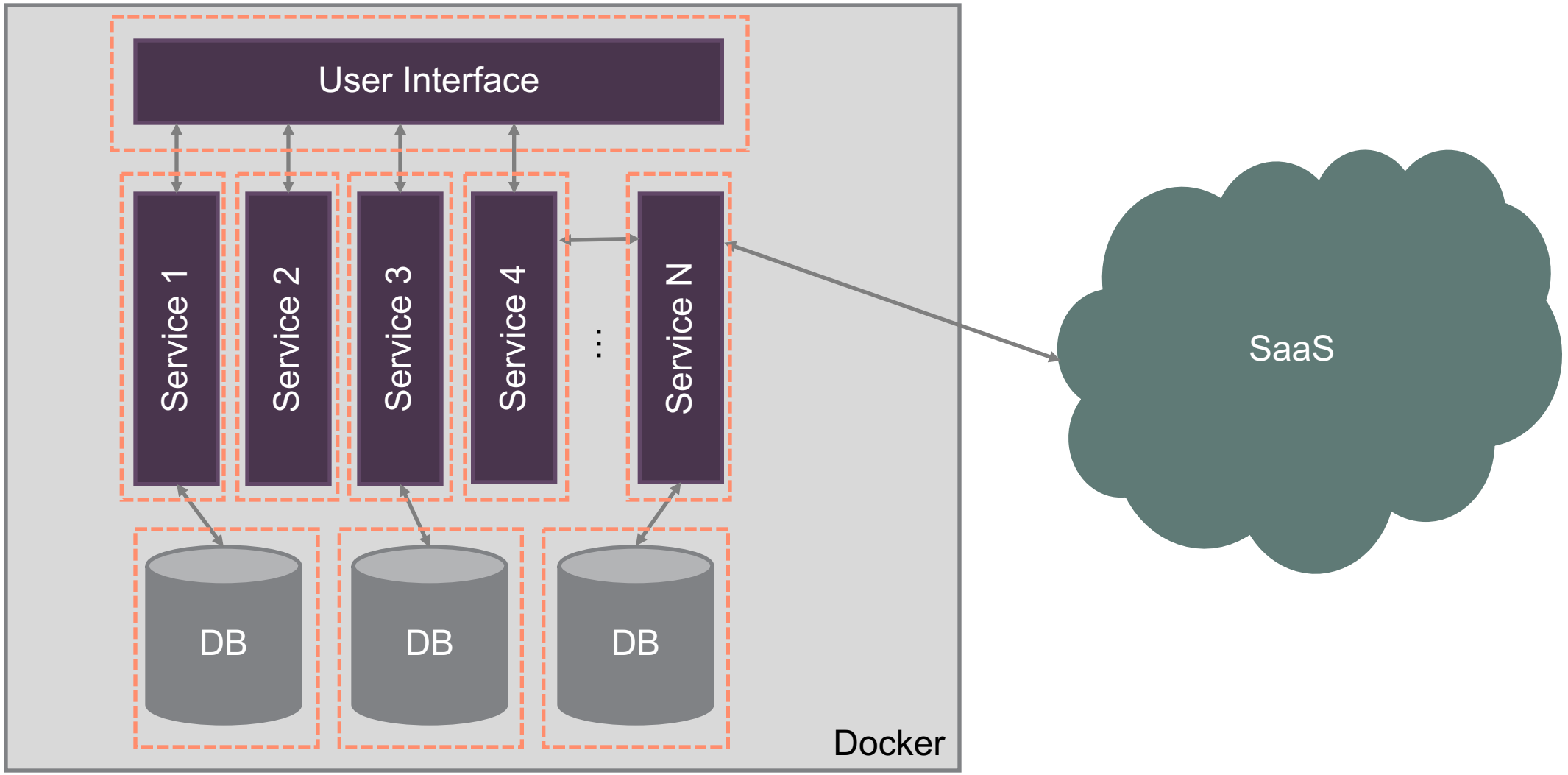
Setup CI


Dev
Git
CI/CD
(Code Engine /
Jenkins)
Environment 1
Stackato
Environment 2
Stackato
Environment 3
Stackato
Personal
Environment(s)

Built a Cloud Native Application

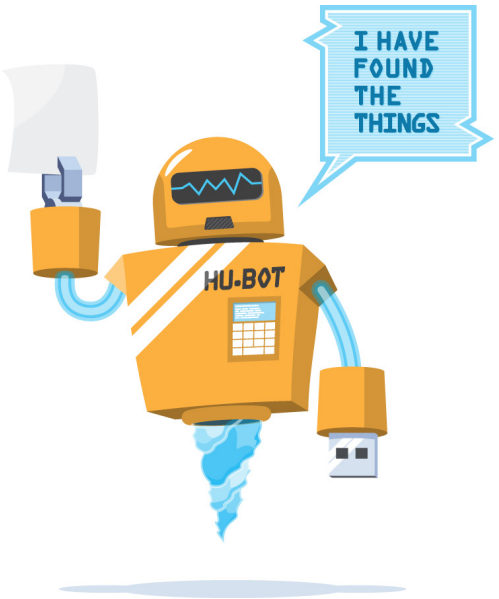


Built a Cloud Native Application



Automation and ChatOps FTW

Jenkins

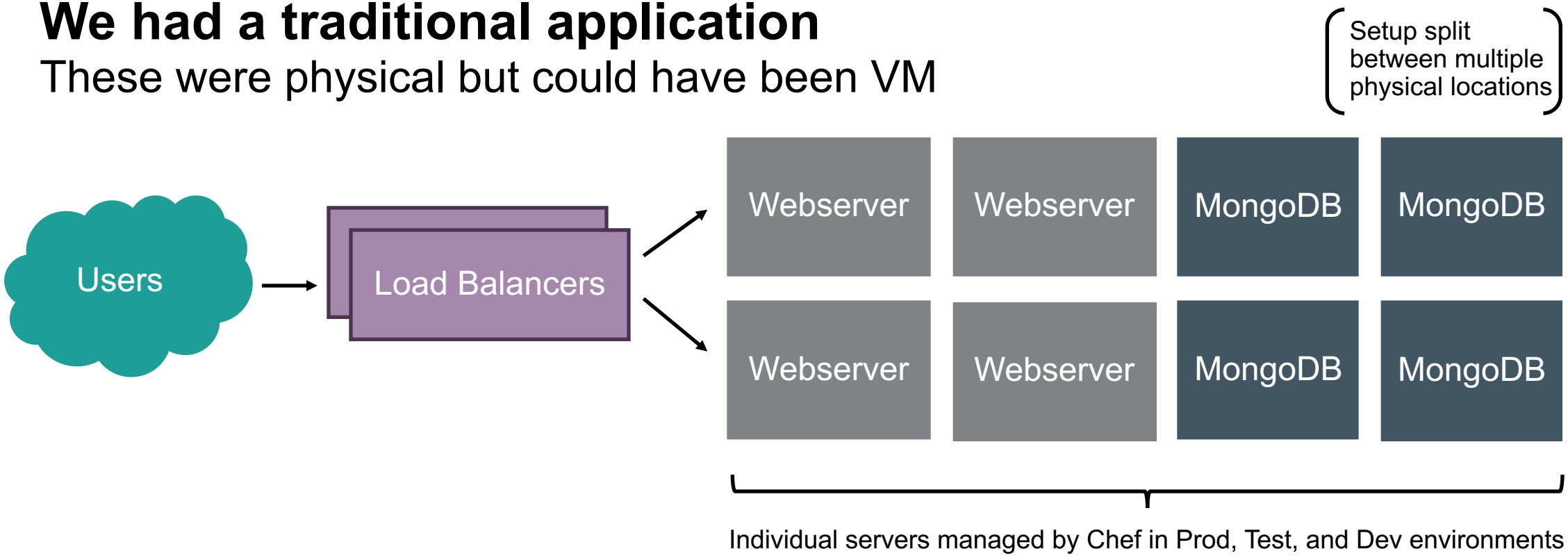


Atlassian HipChat

Path 3: Additive

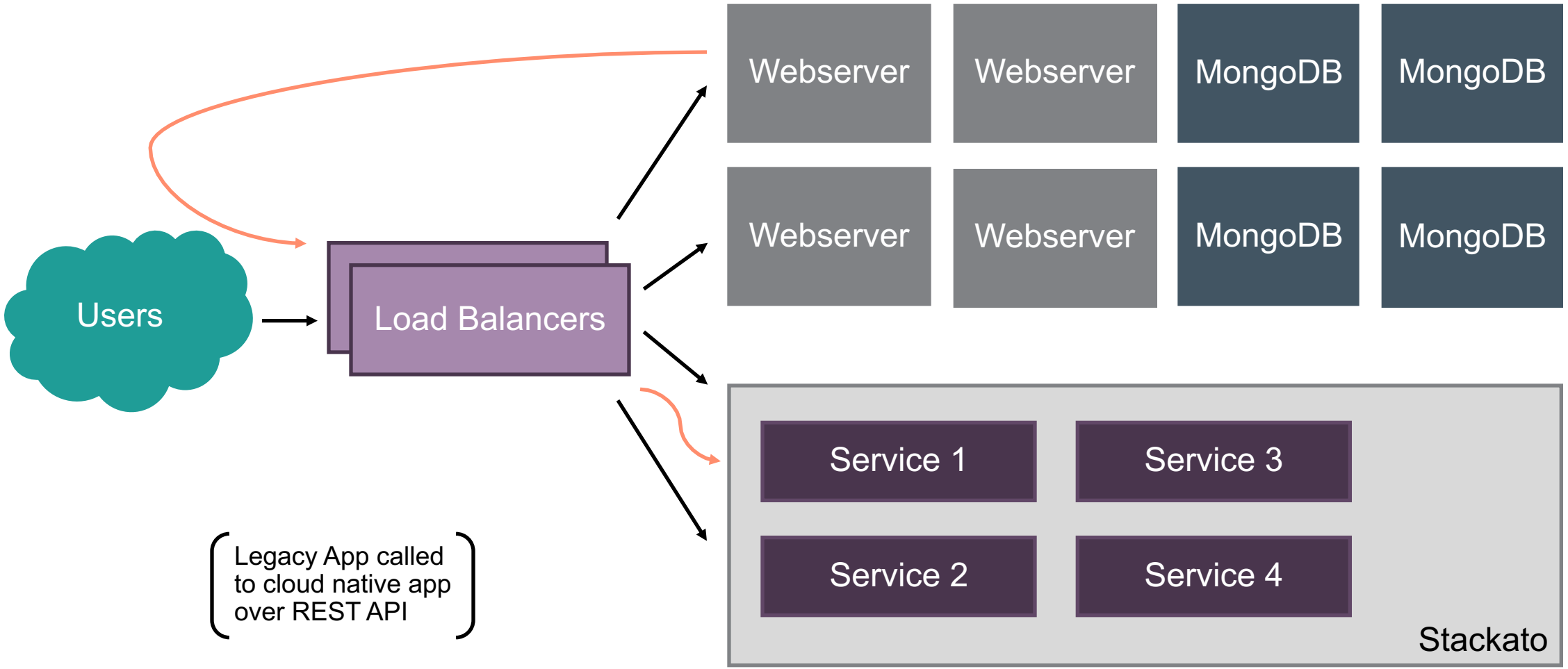
We had a traditional application

These were physical but could have been VM

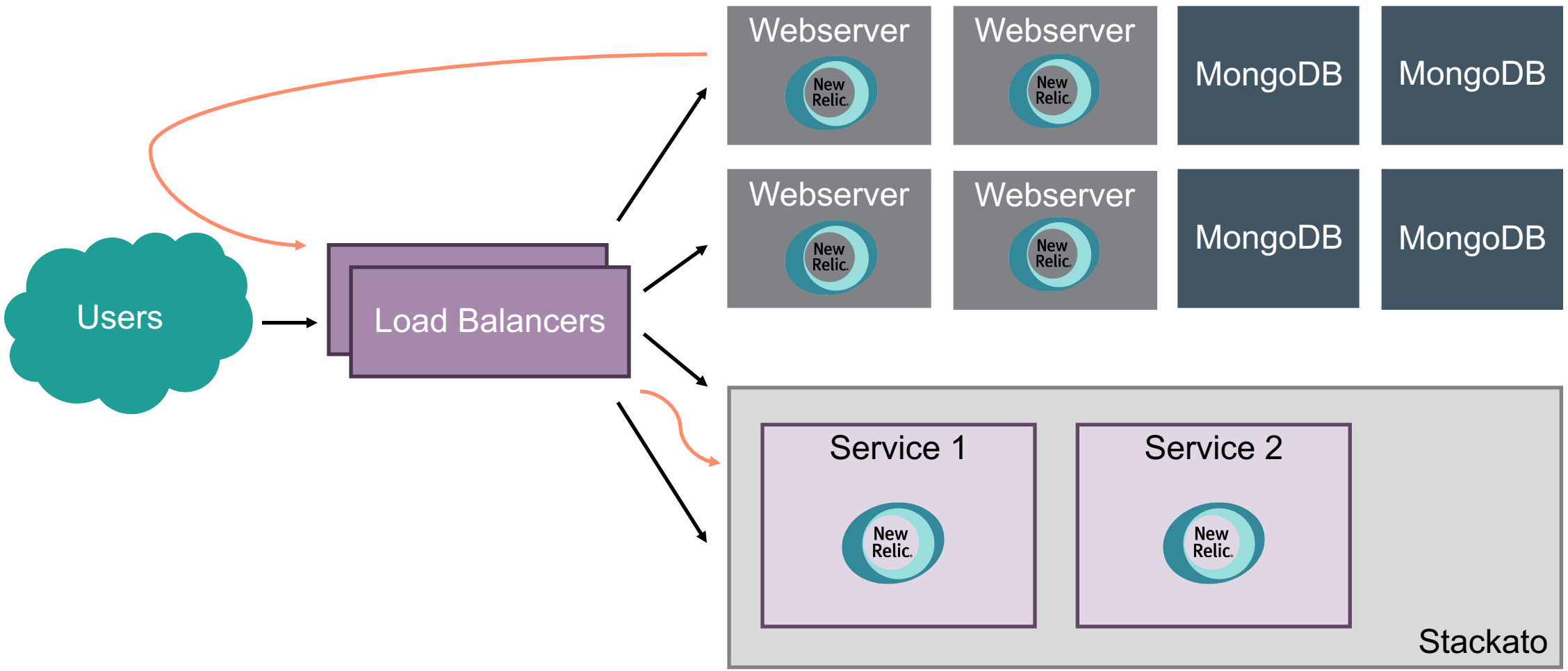


Added Cloud Native Environment

The legacy environment was not retired



Monitor Everything



Monitor Intelligently

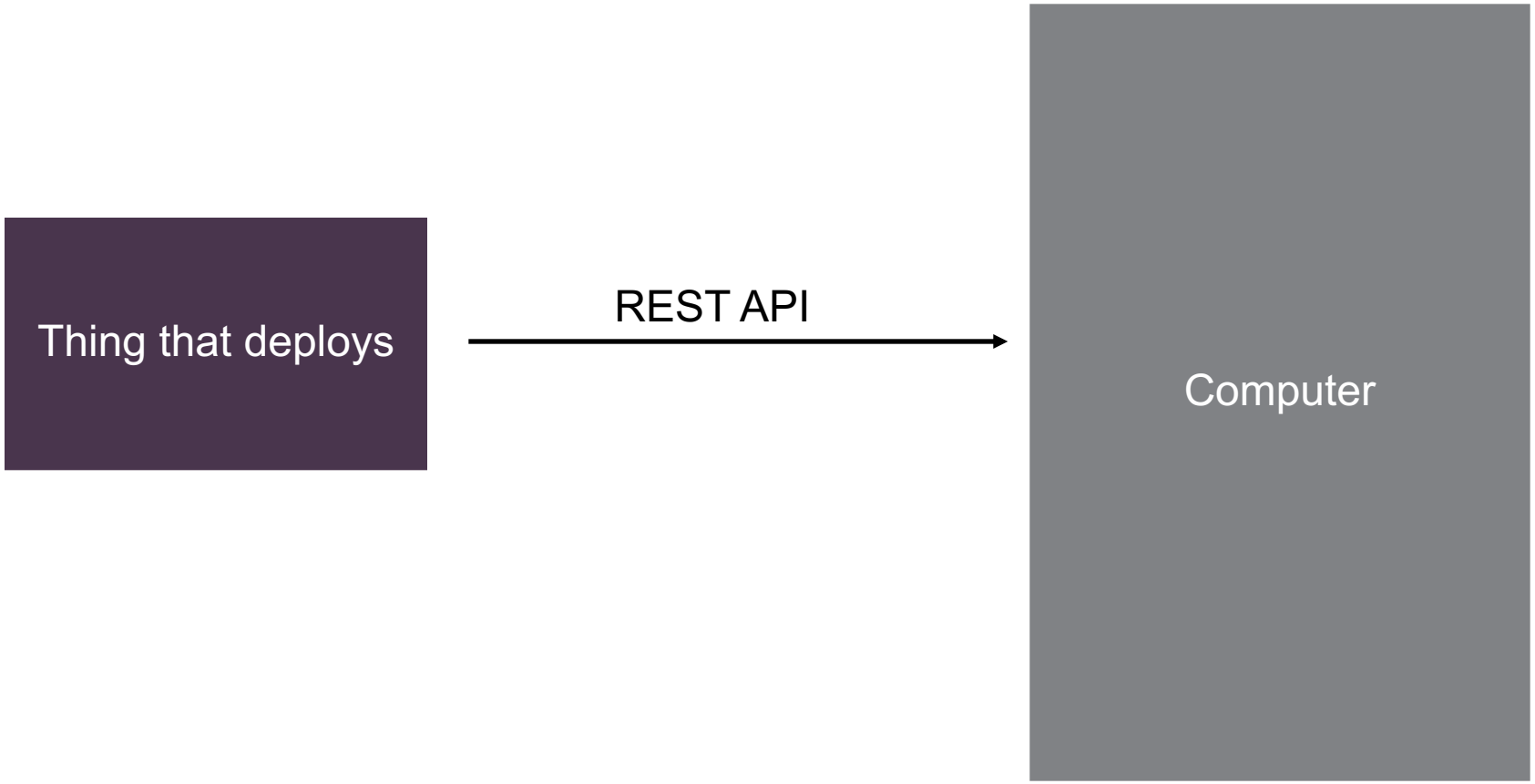
If you didn’t monitor it did it happen?

Tools You Can Use


gRPC

Datacenter/Cluster as a Computer

Make it easy for developers



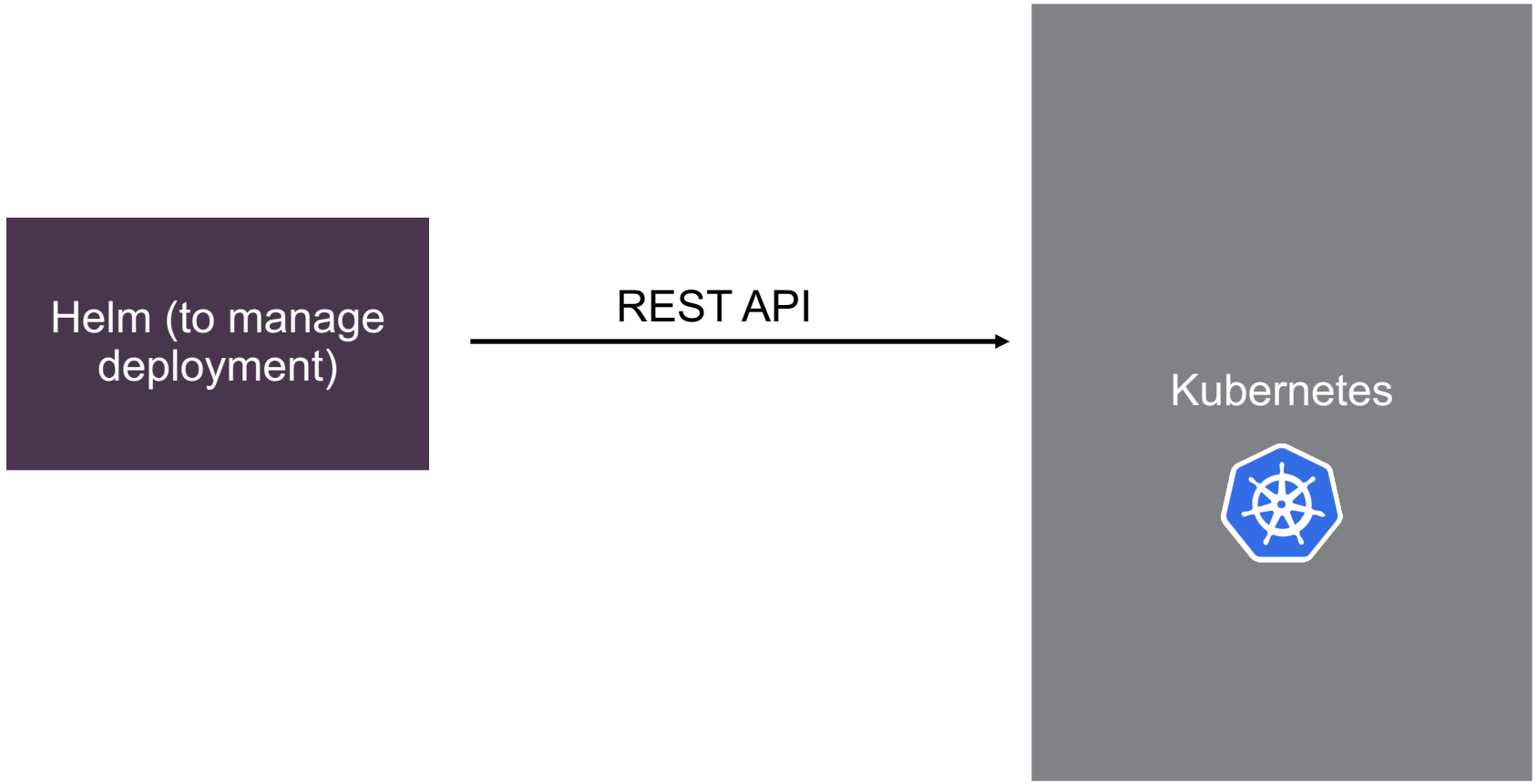
Start With A Platform

Think Datacenter as a Computer

HPE Helion Stackato

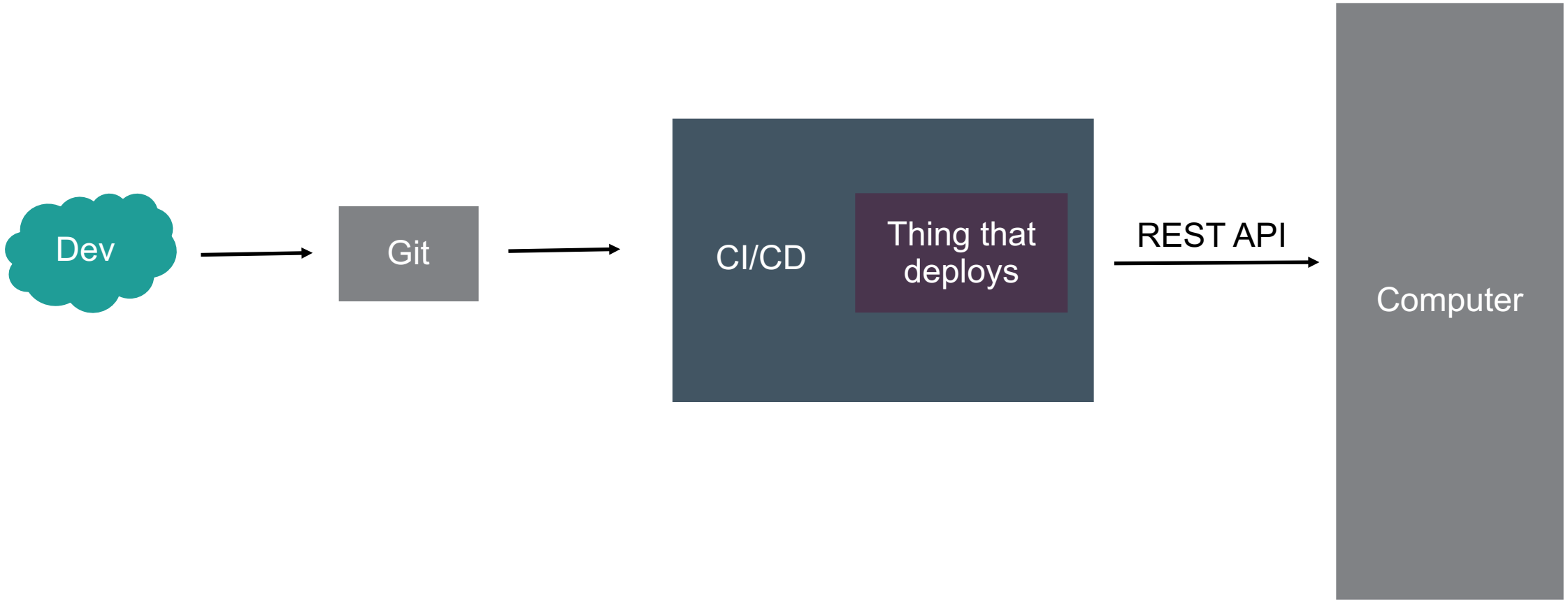
Lifecycle Management

Consider Helm when using Kubernetes



The CI/CD System

Make it easy for developers



CI/CD Systems

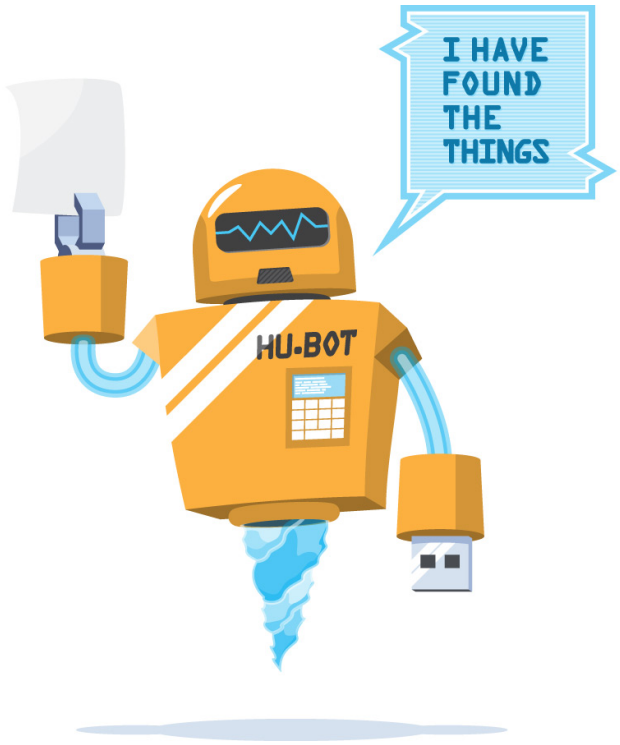
There are just so many

Hewlett Packard Enterprise

Some Are Container Based By Default

Hewlett Packard Enterprise

ChatOps



slack

Atlassian HipChat

Config service

Monitor Everything

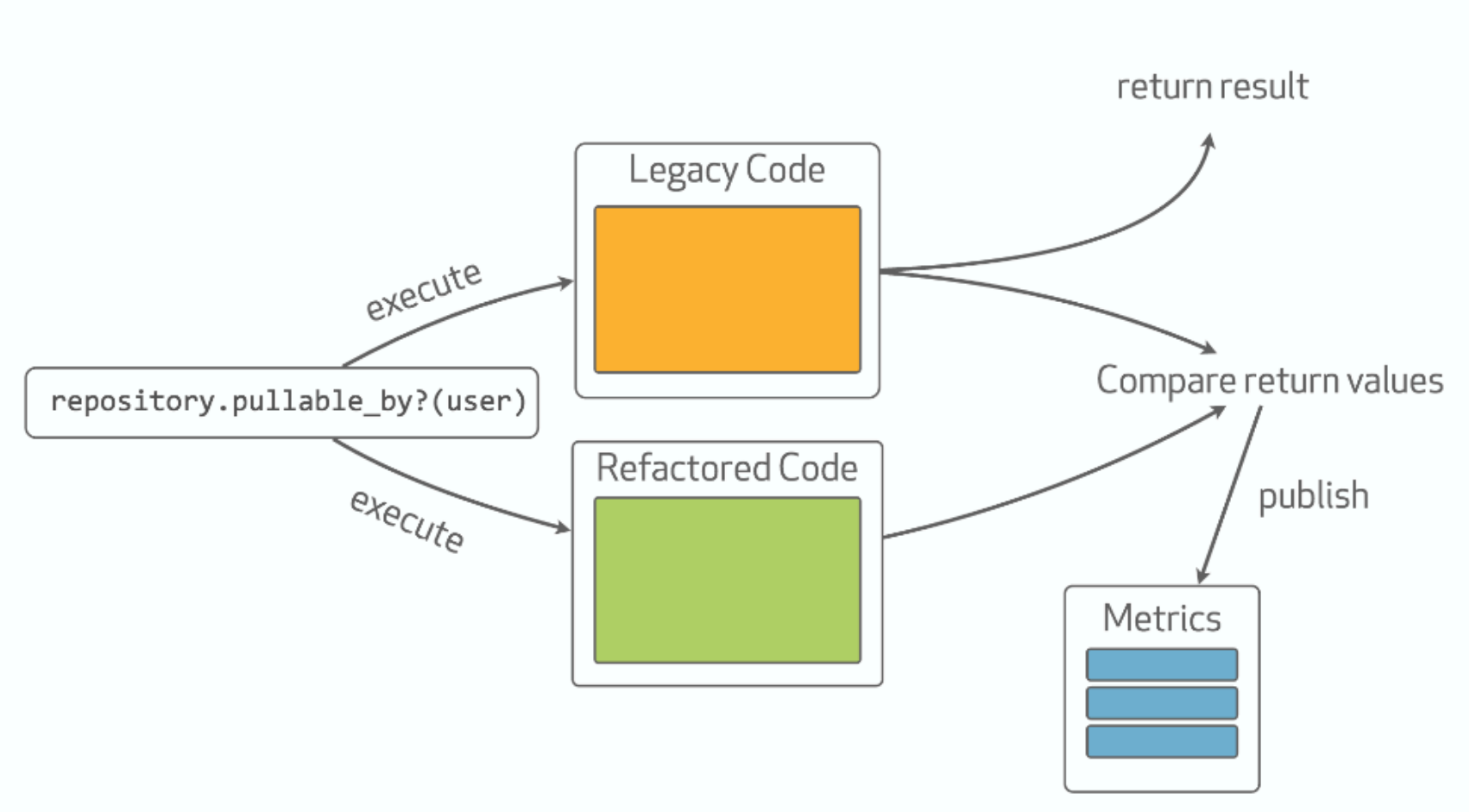
Prometheus

OPENTSDB

ICINGA

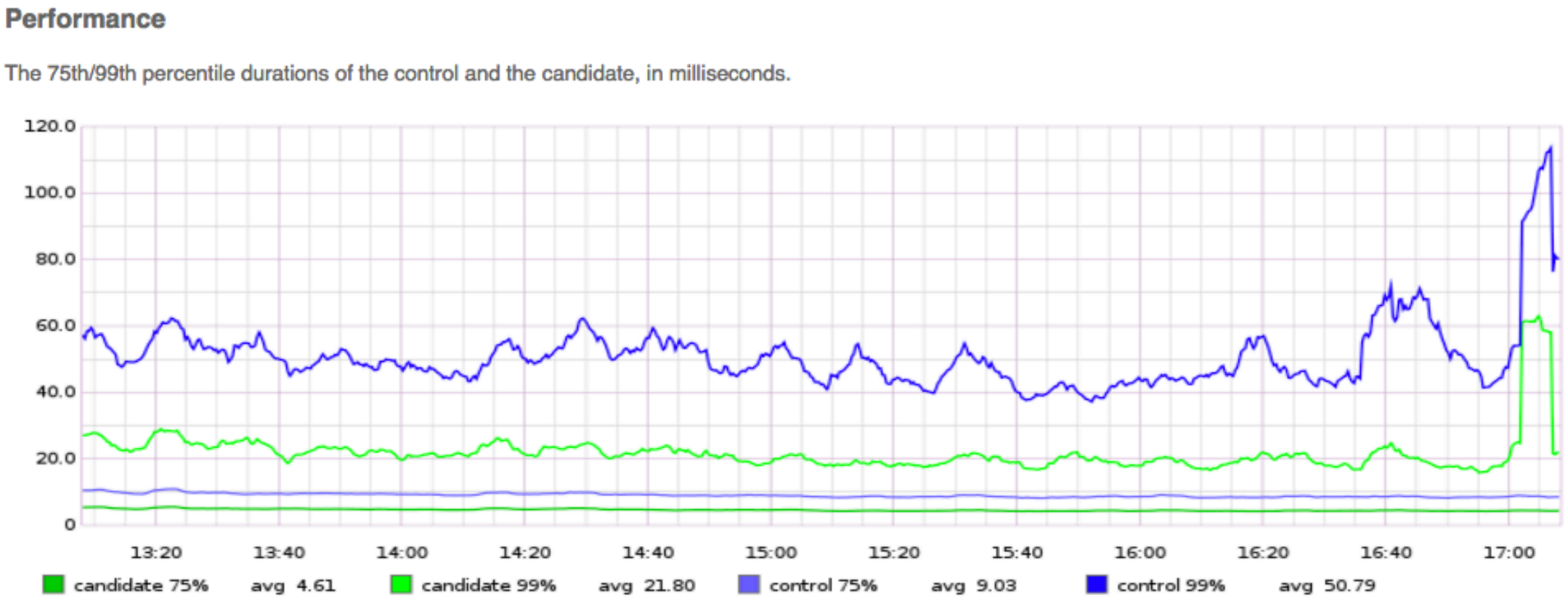
GitHub Scientist

Plus others following suit with more language support



GitHub Scientist

http://githubengineering.com/scientist/



Hewlett Packard
Enterprise

Questions?

Matt Farina
@mattfarina
mattfarina.com / hpe.com