Cloud Native & Microservices at Lunar Way

IDA Presentation - Aarhus School of Engineering
Kasper Nissen - @phennex
Martin Jensen - @mrjensens


kubernetes
Removing failing pod
23
slave2
192.168.1.202
slave3
192.168.1.203
slave4
192.168.1.204
foto: Lars Kruse, Aarhus Universitet
Pervasive Systems group, Section of Electrical and Computer Engineering, Department of Engineering, Aarhus University

Who?

Kasper Nissen (@phennex)

- Cloud Architect / SRE @lunarway
- Previous; LEGO Systems, IT Minds, Drivelogger
- Organiser & Co-Founder of Cloud Native Aarhus
- MSc. Computer Engineering
- Founder Cloud Native DK Slack Community
- Occasional speaker at meet ups and conferences
- Blogger at kubecloud.io

Who?

Martin Jensen (@mrjensens)

- Web Architect @lunarway
- Previous; IT Minds, Drivelogger
- MSc. Computer Engineering
- Blogger at kubecloud.io

lunar way®

AGENDA

- Cloud Native
- Container Orchestration
- Observability
- Microservices
- Service Communication
- Deployment

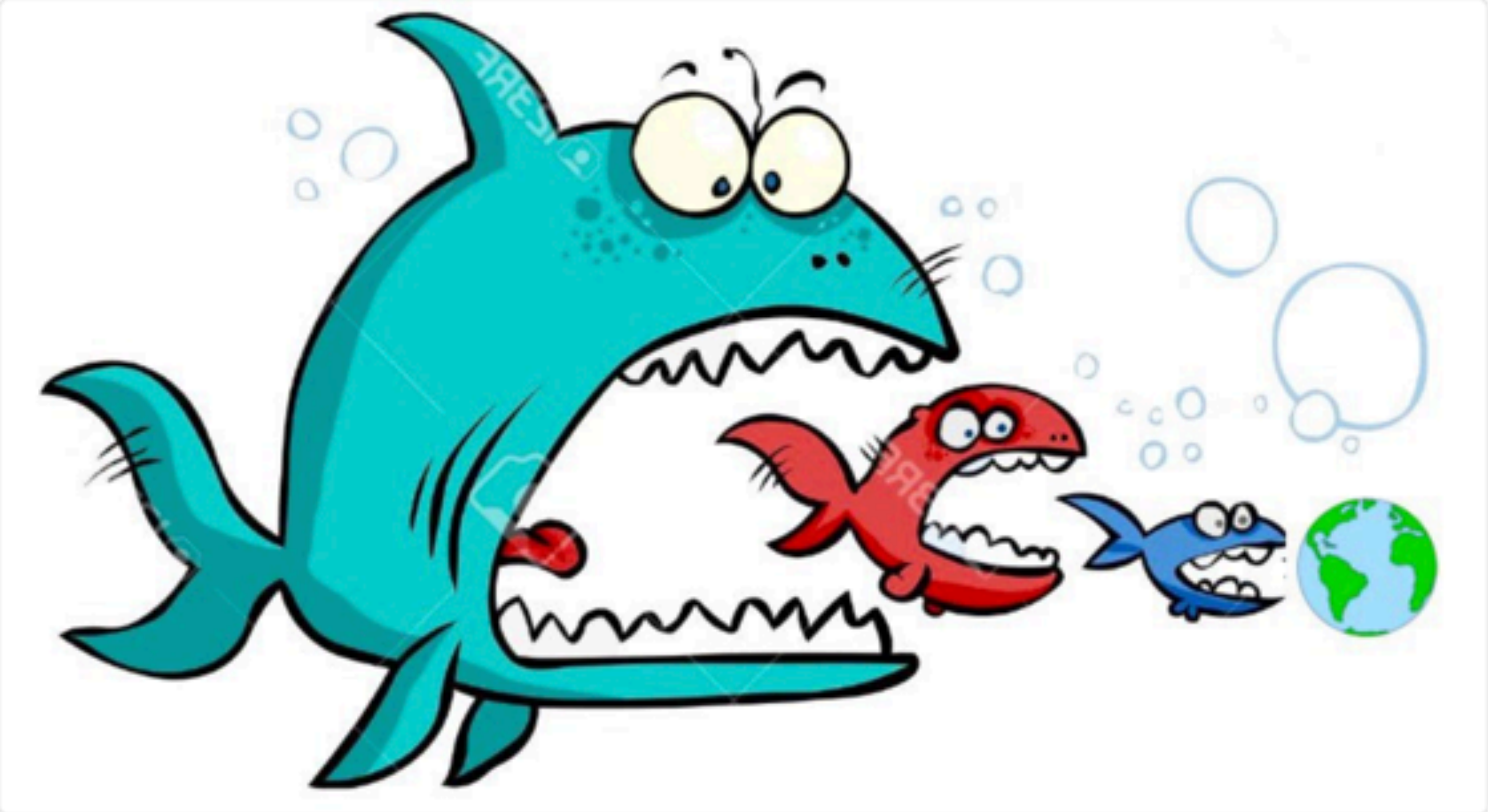
Software is eating the world...

Marc Andreessen, 2011

Joseph Jacks
@asynchio
Følger

"Software is eating the world; OSS is eating software; Cloud is eating OSS!" .. *spot* on. // @monadic

Oversæt tweet



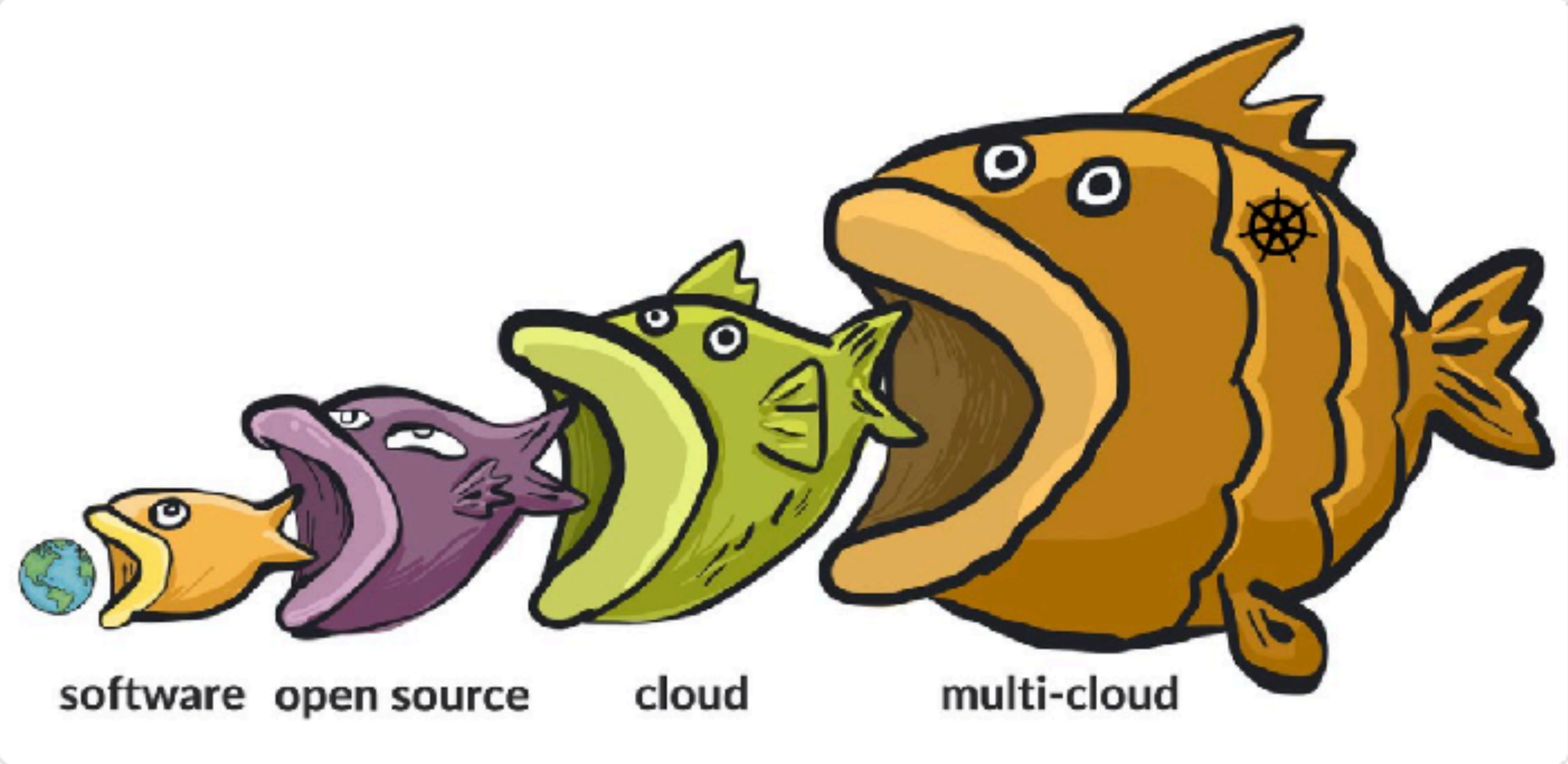
09.05 - 10. apr. 2016

61 Retweets **57** Likes

5 61 57

bassam
@bassamtabbara
Follow

And multi-cloud is going to eat cloud 💥
twitter.com/asynchio/statu …



4:45 PM - 16 May 2018

35 Retweets **78** Likes

5 35 78

Explain, please!

Software is eating the world: all companies are moving towards a digital presence, on way or the other

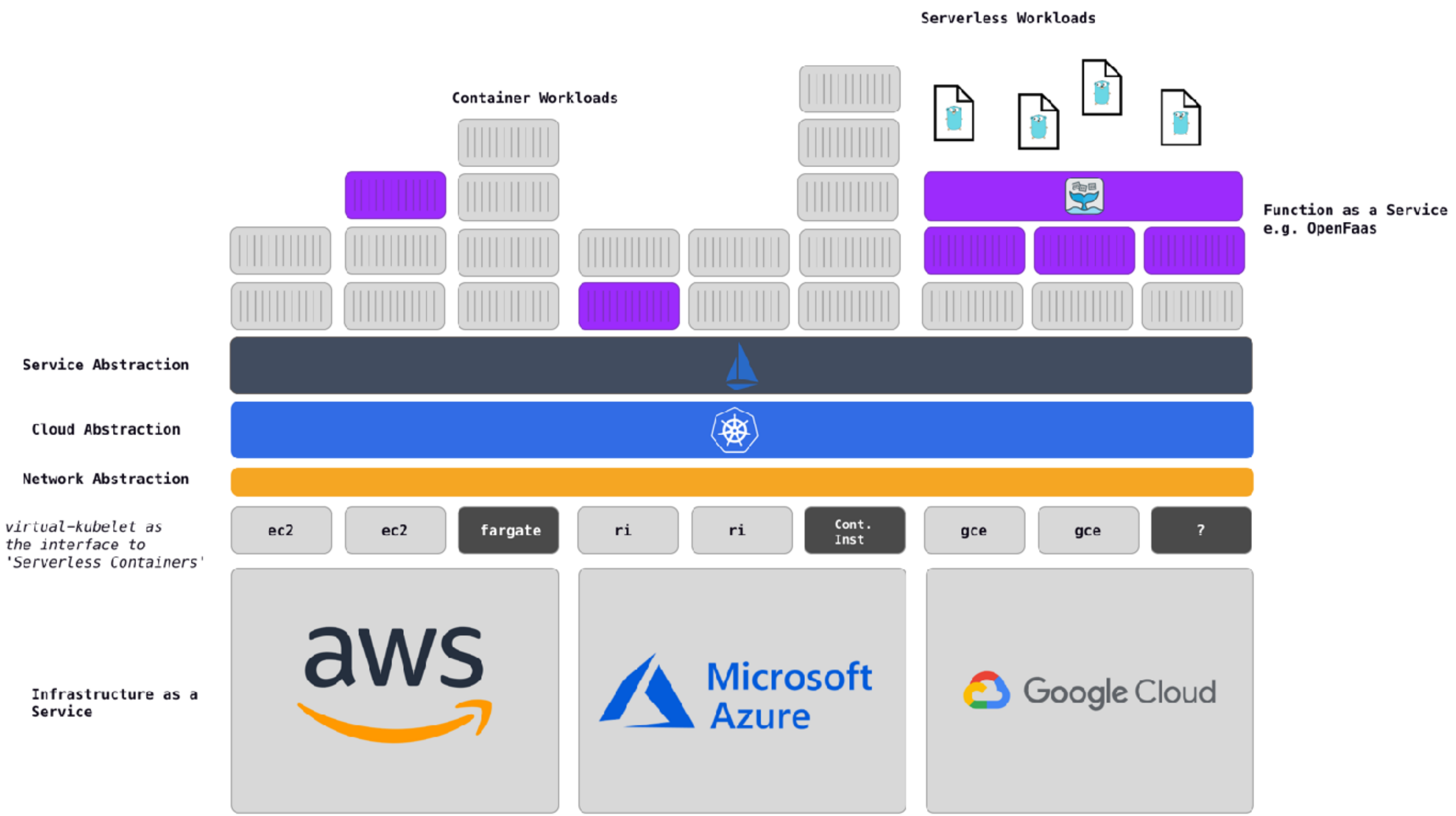
Cloud is eating software: many are moving their systems to the cloud to utilize the elastic and dynamic scalability

Multi-Cloud will eat cloud: in order to not be locked-in to a specific vendor, an increased focus will be on spreading workloads across clouds

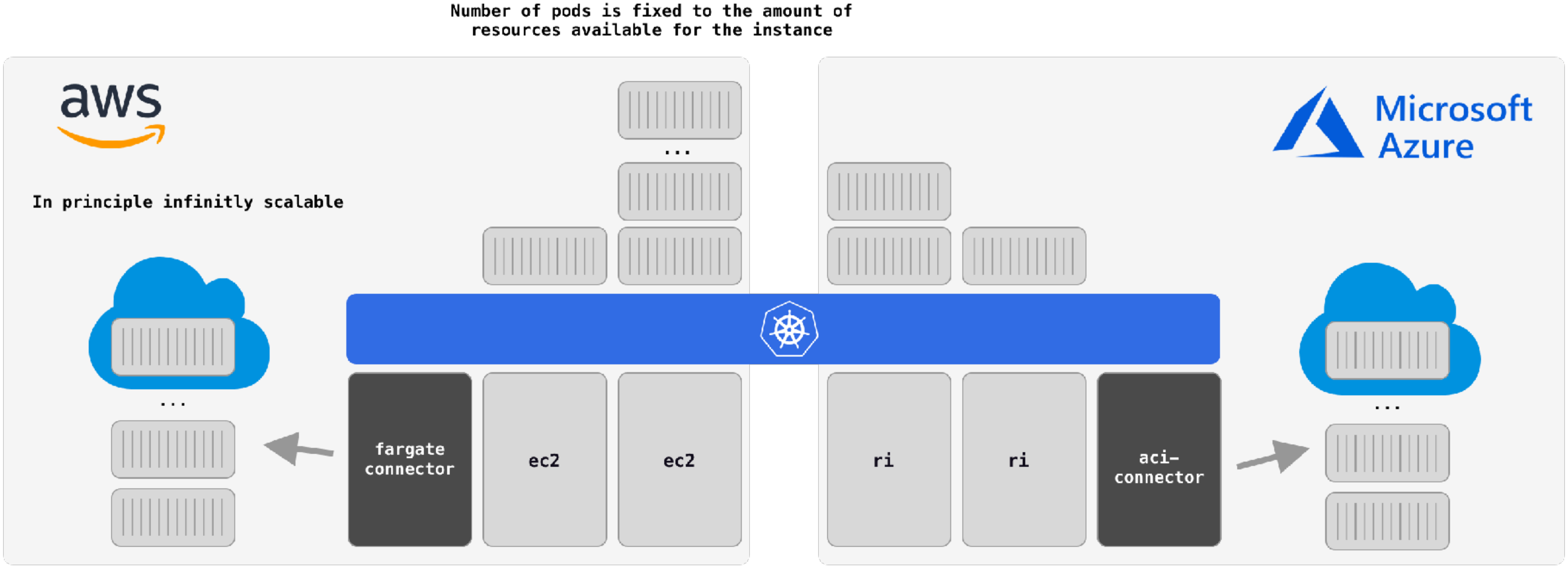
OSS is an enabler for multi-cloud…

We want to move workloads between providers and choose the one who offers the best Prices? Security? Features?

Multi-cloud, the utopian dream?



Virtual-kubelet - infinitly scalable



Cloud Native

https://pixabay.com/en/skyline-new-york-600001/

Cloud Native, the CNCF definition

Cloud native technologies empower organizations to build and run ***scalable applications*** *in modern,* ***dynamic environments*** *such as public, private, and hybrid clouds.* ***Containers****,* ***service meshes****,* ***microservices****,* ***immutable infrastructure****, and* ***declarative APIs*** *exemplify this approach.*

These techniques enable ***loosely coupled*** *systems that are* ***resilient****,* ***manageable****, and* ***observable****. Combined with robust automation, they allow engineers to make* ***high-impact changes frequently*** *and* ***predictably*** *with* ***minimal toil****.*

The Cloud Native Computing Foundation seeks to drive adoption of this ***paradigm*** *by fostering and sustaining an ecosystem of* ***open source, vendor-neutral projects****. We democratize state-of-the-art patterns to make these innovations accessible for everyone.*

https://www.cncf.io/about/faq/

CLOUD NATIVE
COMPUTING FOUNDATION

Key characteristics

- Scalable systems
- Microservices
- Dynamic environments
- Containers
- Immutable Infrastructure
- Observability and manageability
- Open source

Why are we adapting this paradigm at Lunar Way?


https://www.pexels.com/photo/flight-sky-earth-space-2159/
USA
Speed!


Cloud Native Landscape
v20180525
See the interactive landscape at l.cncf.io
Greyed logos are not open source
App Definition and Development
Database and Data Warehouse
Streaming
Source Code Management
Application Definition and Image Build
Continuous Integration / Continuous Delivery (CI/CD)
Orchestration & Management
Scheduling & Orchestration
Coordination & Service Discovery
Service Management
Runtime
Cloud-Native Storage
Container Runtime
Cloud-Native Network
Provisioning
Host Management / Tooling
Infrastructure Automation
Container Registries
Secure Images
Key Management
Cloud
Public
Private
Platforms
Certified Kubernetes - Distribution
Certified Kubernetes - Hosted
Certified Kubernetes - Installer
Non-Certified Kubernetes
PaaS/Container Service
Observability & Analysis
Monitoring
Logging
Tracing
Serverless
Kubernetes Certified Service Provider
Kubernetes Training Partner
Special
Vitess
CNCF Incubating
NATS
cloudevents
CNCF Sandbox
kubernetes
CNCF Graduated
CoreDNS
GRPC
envoy
LINKERD
Open Policy Agent
ROOK
containerd
rkt
CNI
Notary
TUF
spiffe
SPIRE
Prometheus
fluentd
JAEGER
OPENTRACING
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.
l.cncf.io
CLOUD NATIVE Landscape
CLOUD NATIVE COMPUTING FOUNDATION
Redpoint
Amplify

Serverless Cloud Native Landscape
v20180525

See the serverless interactive landscape at s.cncf.io

Greyed logos are not open source

Tools
dashbird, Epsagon, event gateway, Glee, IO|pipe, Iron.io, Node Lambda, OVERCLOCK, python-λ, SIGMA, STACKERY, THUNDRA

Security
intrinsic, Protego, PURESEC, snyk

Framework
APEX, .arc Architect, aws Chalice, AWS SAM, Claudia.js, FLOGO, gun.io, serverless, Shep, SPARTA, Spring Cloud Function

Platform

Hosted
ALGORITHMIA, AWS Lambda, Azure Functions, BINARIS, Clay, cloudboost, CLOUDFLARE, Google Cloud Functions, HUAWEI, HYPER.SH, IBM Cloud, NANO-LAMBDA, PubNub FUNCTIONS, spotinst, stdlib, syncano, twilio, WEBLAB, ZEIT

Installable
Apache OpenWhisk, AppScale, DISPATCH, fission, fn, GALACTIC FOG, Kubeless, LunchBadger, nuclio, OPENFAAS, OpenLambda, riff

s.cncf.io

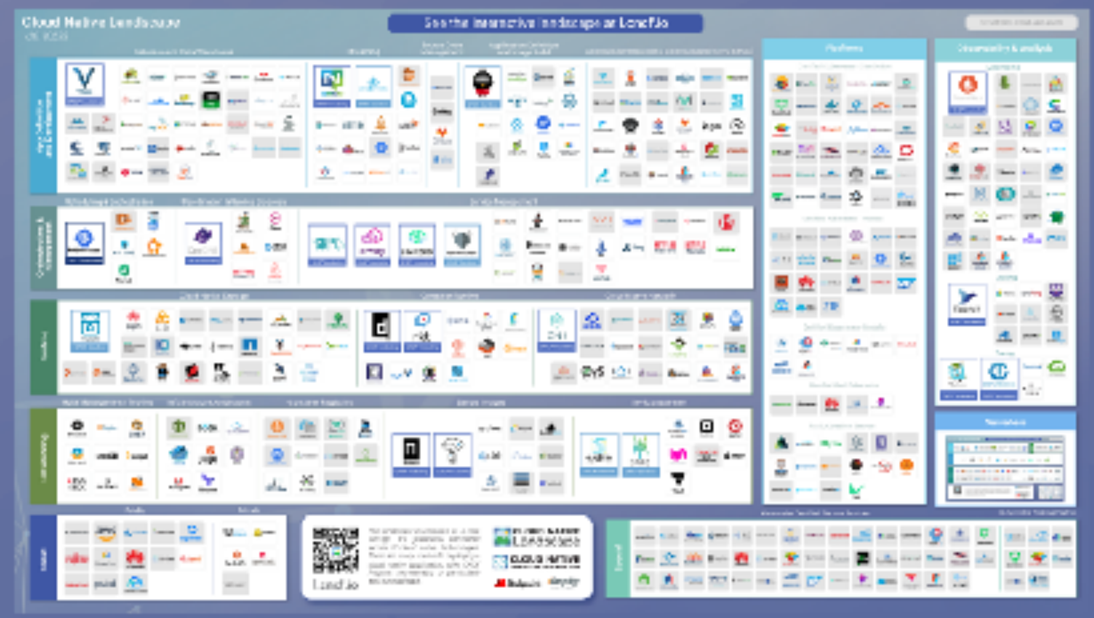
Serverless computing refers to a new model of cloud native computing, enabled by architectures that do not require server management to build and run applications. This landscape illustrates a finer-grained deployment model where applications, bundled as one or more functions, are uploaded to a platform and then executed, scaled, and billed in response to the exact demand needed at the moment.

CLOUD NATIVE Landscape

CLOUD NATIVE COMPUTING FOUNDATION

Redpoint

Cloud Native Landscape



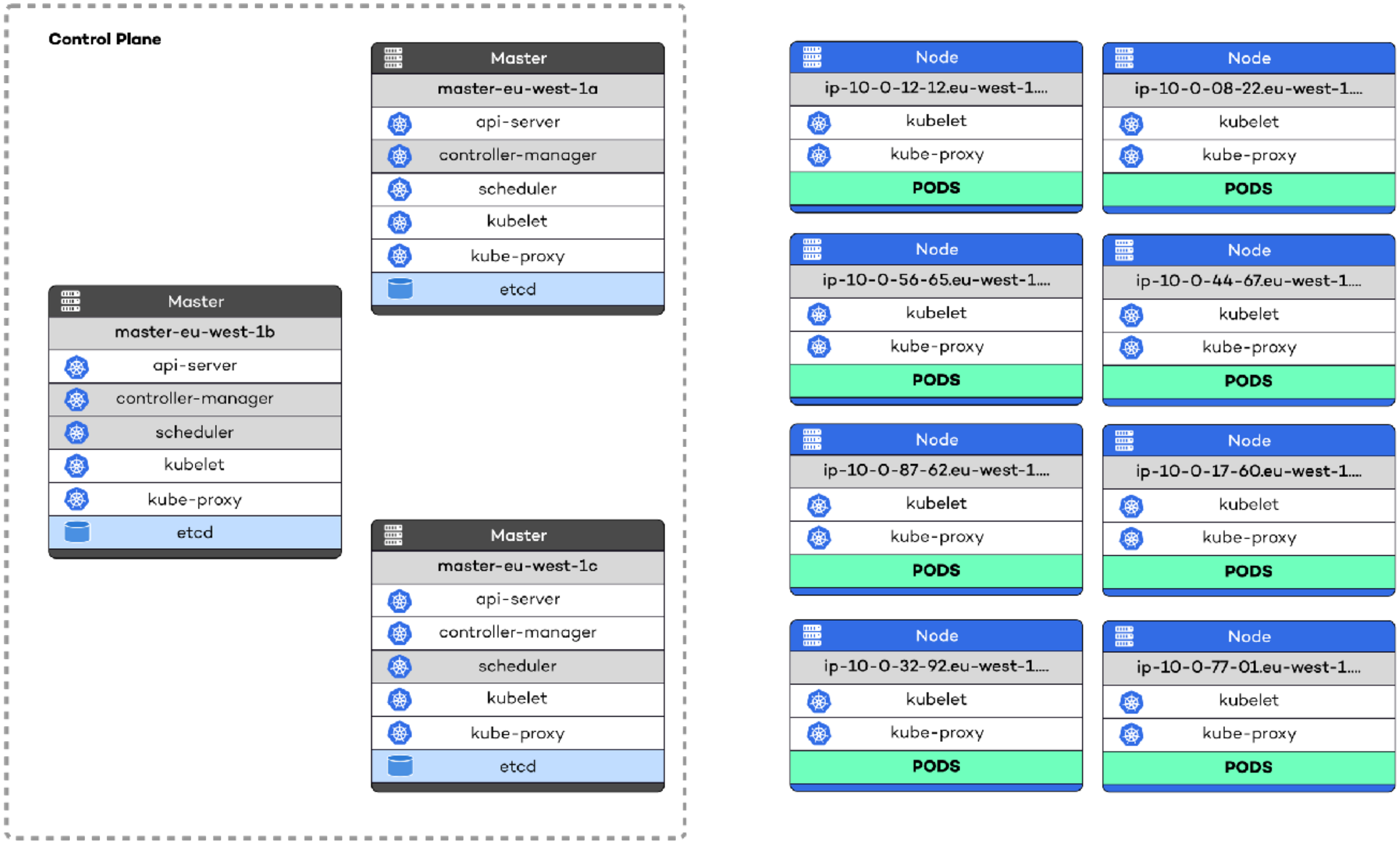
Container Orchestration with Kubernetes

https://www.pexels.com/photo/shallow-focus-photography-of-black-ship-1095814/

Kubernetes at Lunar Way

- Kubernetes in production since March 2017
- Three clusters at the moment (dev, staging, prod)
- Kubernetes Operations (Kops) with quite a lot of configuration
- Production environment is a multi-master highly available cluster
- Started at Kubernetes 1.5 and are now at 1.9.6

Highly Available Kubernetes



Kops

```
kops create cluster \
  --name dev.example.com \
  --state s3://some-s3-bucket \
  --node-count 3 \
  --zones eu-west-1a,eu-west-1b,eu-west-1c \
  --master-zones eu-west-1a,eu-west-1b,eu-west-1c \
  --dns private \
  --node-size m4.large \
  --master-size m4.large \
  --topology private \
  --networking weave \
  --yes
```

Declarative Cluster Spec

```yaml
apiVersion: kops/v1alpha2
kind: Cluster
metadata:
  name: k8s.test.lunarway.com
spec:
  api:
    loadBalancer:
      type: Public
  authorization:
    rbac: {}
  channel: stable
  cloudProvider: aws
  configBase: s3://somebucket/k8s.test.lunarway.com
  dnsZone: DNSZONE
  etcdClusters:
  - etcdMembers:
    - instanceGroup: master-eu-west-1a
      name: a
    name: main
  - etcdMembers:
    - instanceGroup: master-eu-west-1a
      name: a
    name: events
```

Kops - Pros/Cons

Pros

- Very easy to spin up clusters
- Highly configurable
- Declarative cluster specifications
- Possible to output to terraform if needed

Cons

- Previously pretty bad defaults
- Release cadence is lacking a couple of months behind upstream Kubernetes

Kubernetes - Pros/Cons

Pros

- Allow us to easily deploy services independently
- Rolling back is fast
- Provides us with resilience
- Makes management of services easy and immutable

Cons

- Very complex system
- Sometimes to high velocity

Other interesting tales

- First upgrades of the productions clusters caused a 30 min outage, because of network congestion fetching pods
- Some services where just moved to this dynamic environment, not handling termination very well
- Problems with kubelet increasing resource consumption

Revisiting the fallacies of ~~distributed~~ Cloud Native 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.

Observability

https://www.pexels.com/photo/men-working-at-night-256219/

Observability

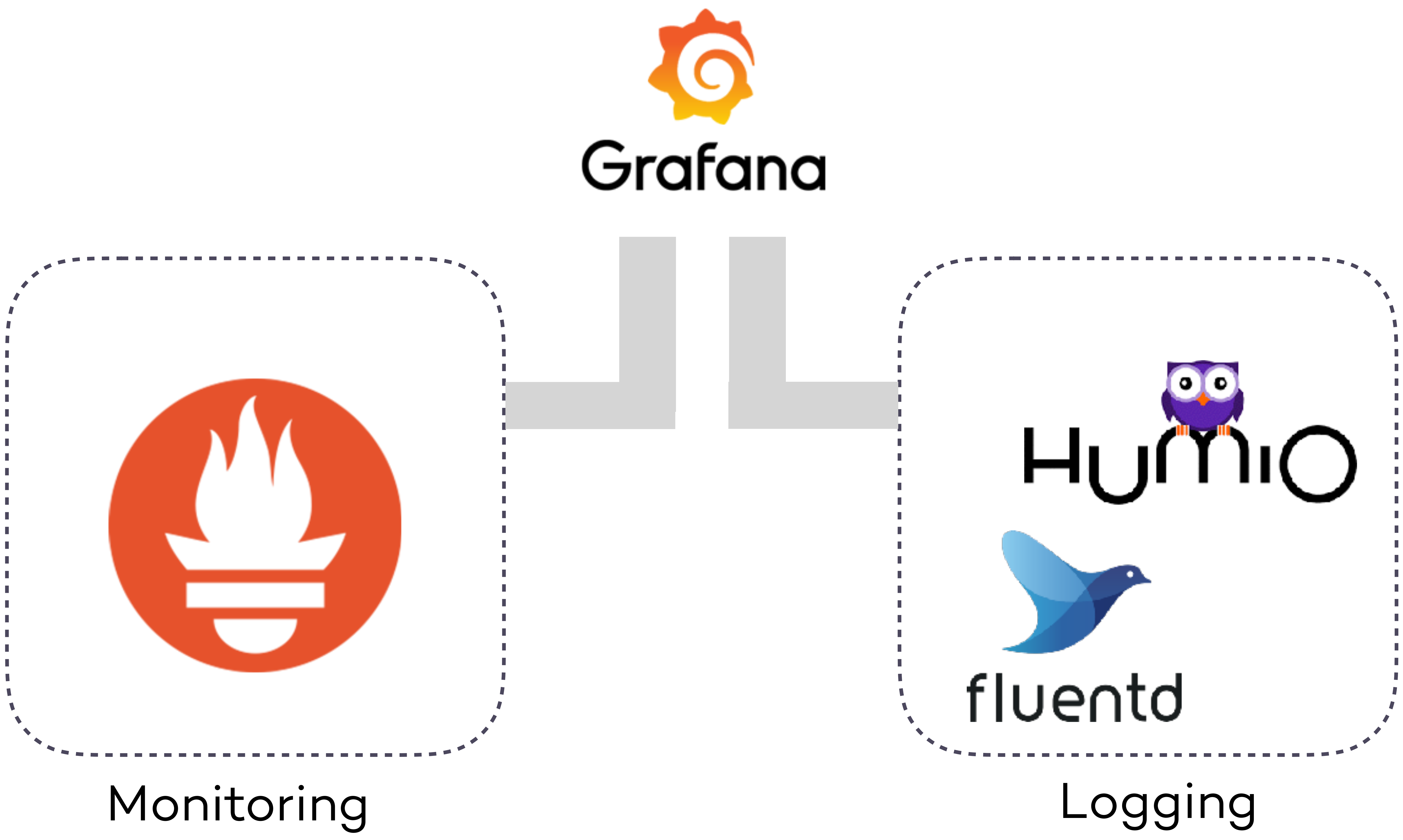
- The overarching goal of various schools of thought on observability, is bringing better visibility into systems.

OBSERVABILITY IS NOT JUST ABOUT LOGS, METRICS, AND TRACES

Logs, metrics, and traces are useful tools that help with testing, understanding, and debugging systems. However, it's important to note that plainly having logs, metrics, and traces does not result in observable systems.

Cindy Sridharan, Book "Distributed Systems Observability", 2018

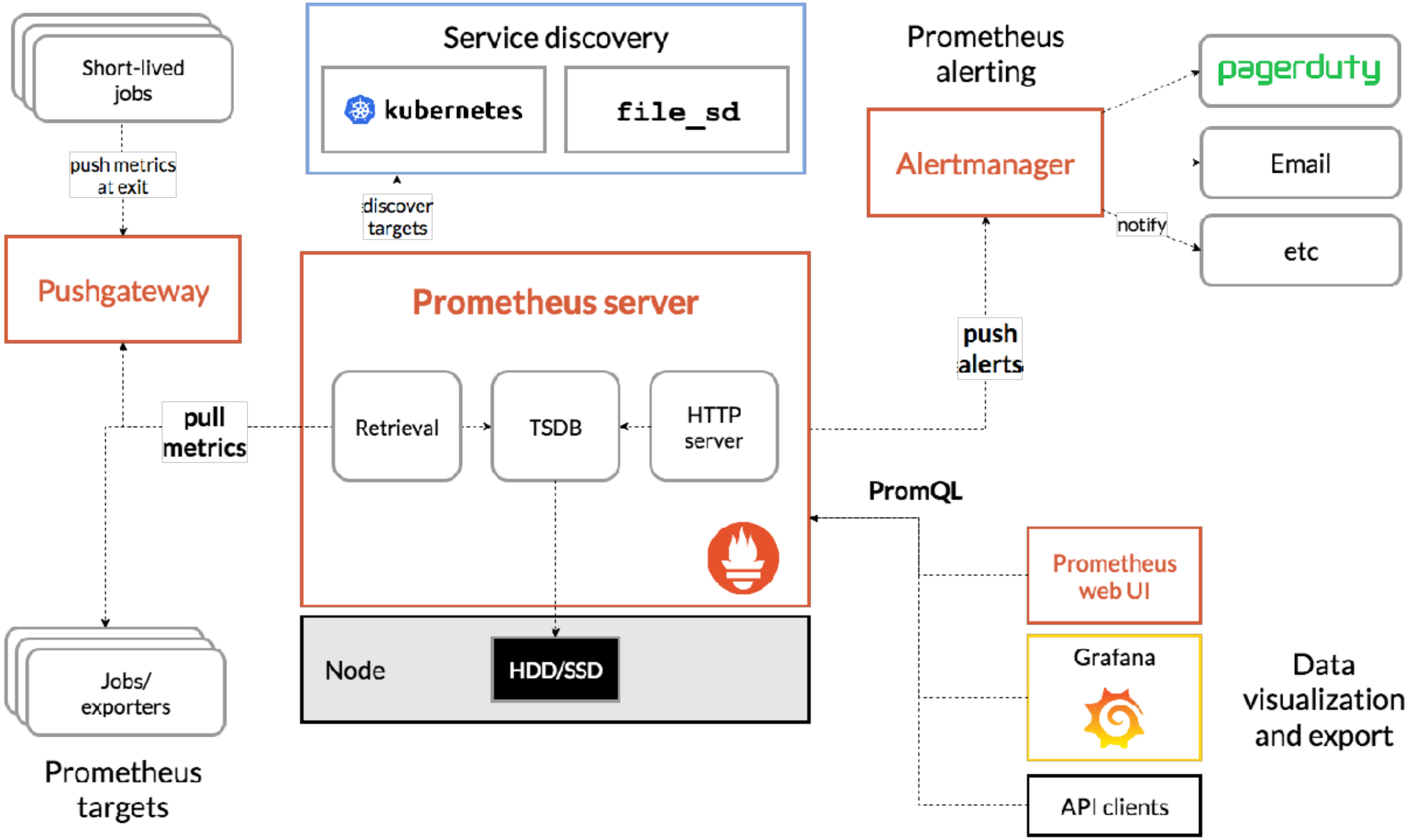
Observability at Lunar Way



What is Prometheus?

- Monitoring system and Timeseries Database
- Instrumentation
- Metrics collection and storage
- Querying
- Alerting
- Dashboard / Graphing / Trending

Prometheus, an overview



Why Monitor?

- Analysing long-term trends
- Comparing over time or experiment groups
- Alerting
- Building dashboards to gain insights
- Conducting ad hoc retrospective analysis

Basically, being able to find out what is broken and why...

and ... even better... know it before it impacts customers..

Prometheus - Pros/Cons

Pros

- Provides great insights to all of our services
- Makes it easy for developers to instrument their services
- Integrates well with many different services

Cons

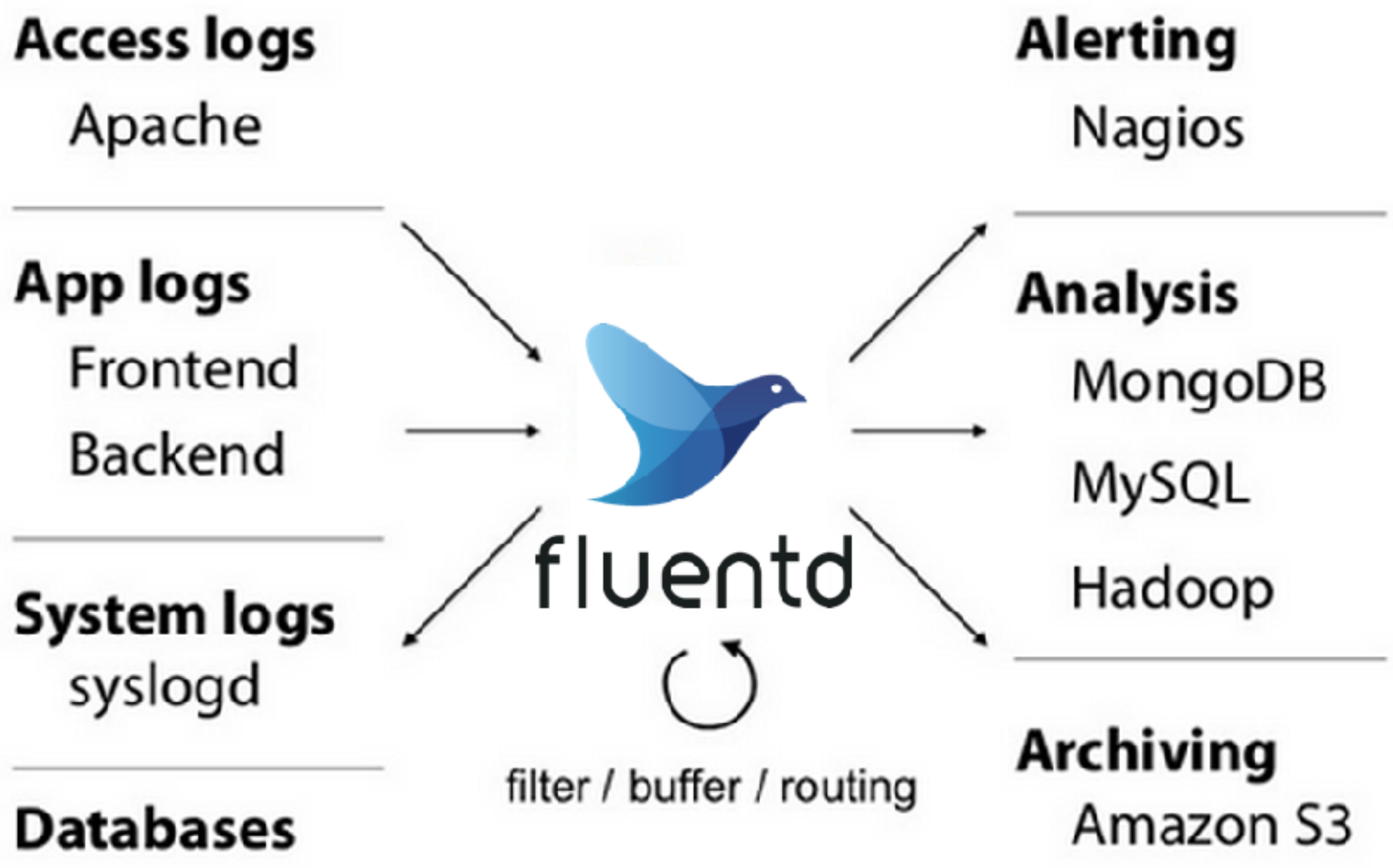
- Prometheus do not support clustered setup

Other interesting tales

- We've had many internal discussions on when to use logs and when to use metrics
- Before Prometheus 2.0 we had a lot of difficulties with high memory consumption
- We write exporters internally for monitoring external partners

fluentd, what is it?

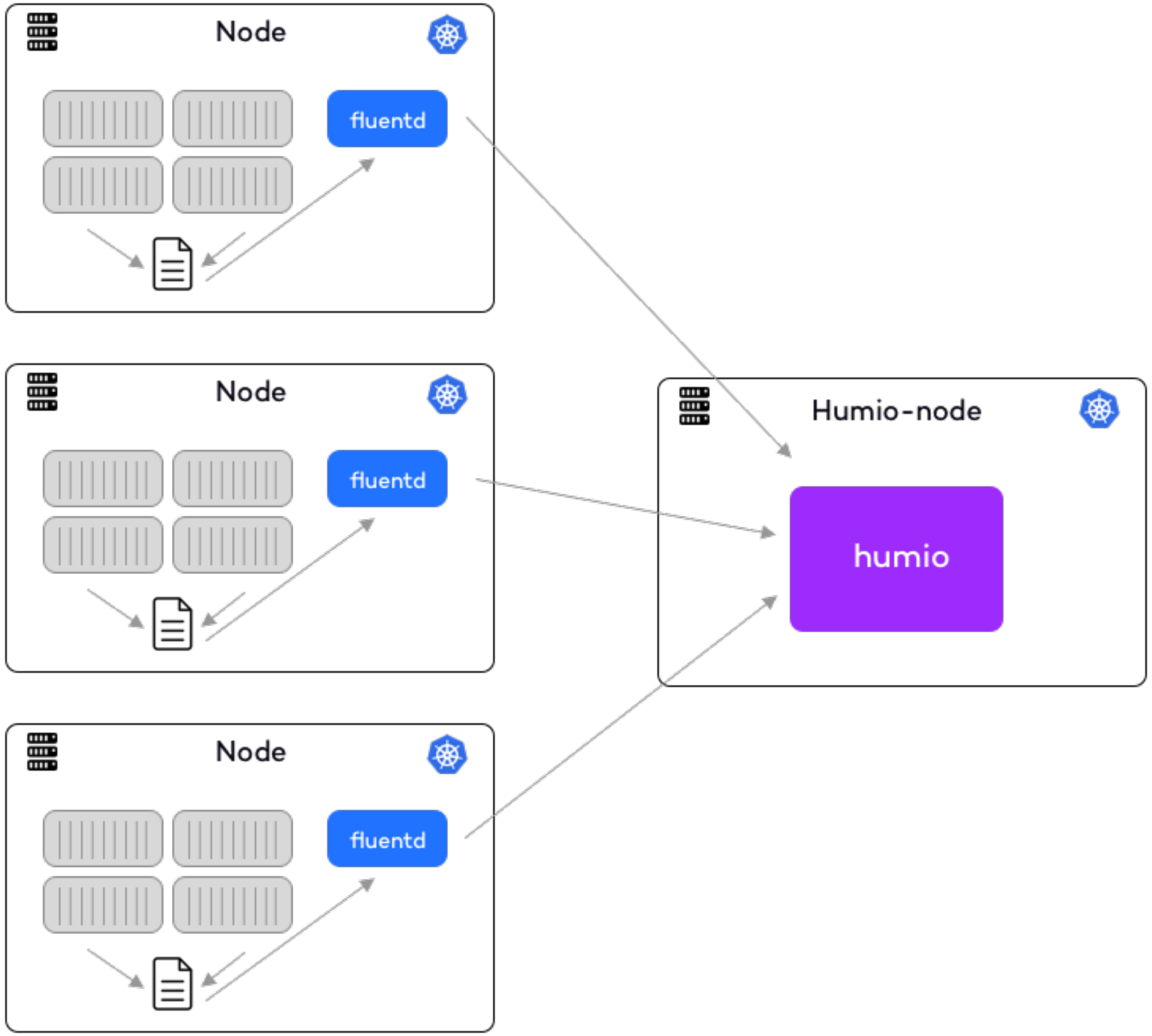
- Fluentd is a log collector
- Hosted by the CNCF



humio, what is it?

- Humio is a log management solution (unfortunately not open-source)
- Humio provides a simple and developer friendly query language for getting insights into your logs

Architecture overview



Microservices

https://pixabay.com/en/beehive-bees-honeycomb-honey-bee-337695/

What is a microservice?

*Microservices are **small**, **autonomous** services that **work together***
Sam Newman

***UNIX philosophy**: Do One Thing and Do It Well*

```
curl https://microservices.io                | grep -i 'communication'
cat what_is_a_microservices.txt              | grep -i 'communication'
kubectl logs -f api-64fdb4bcd5-9gflk | grep -i 'communication'
```

What is a microservice?

*…a single application as a suite of small services, each running in its **own process** and **communicating with lightweight mechanisms**…*

*…services are built around business capabilities and **independently deployable** by fully **automated deployment** machinery*

Martin Fowler & James Lewis

Microservices overall

Benefits

Velocity

Autonomy

Coherence and low coupling

Resilience

Independent deployment

Challenges

Debugging and tracing

Increased overall complexity

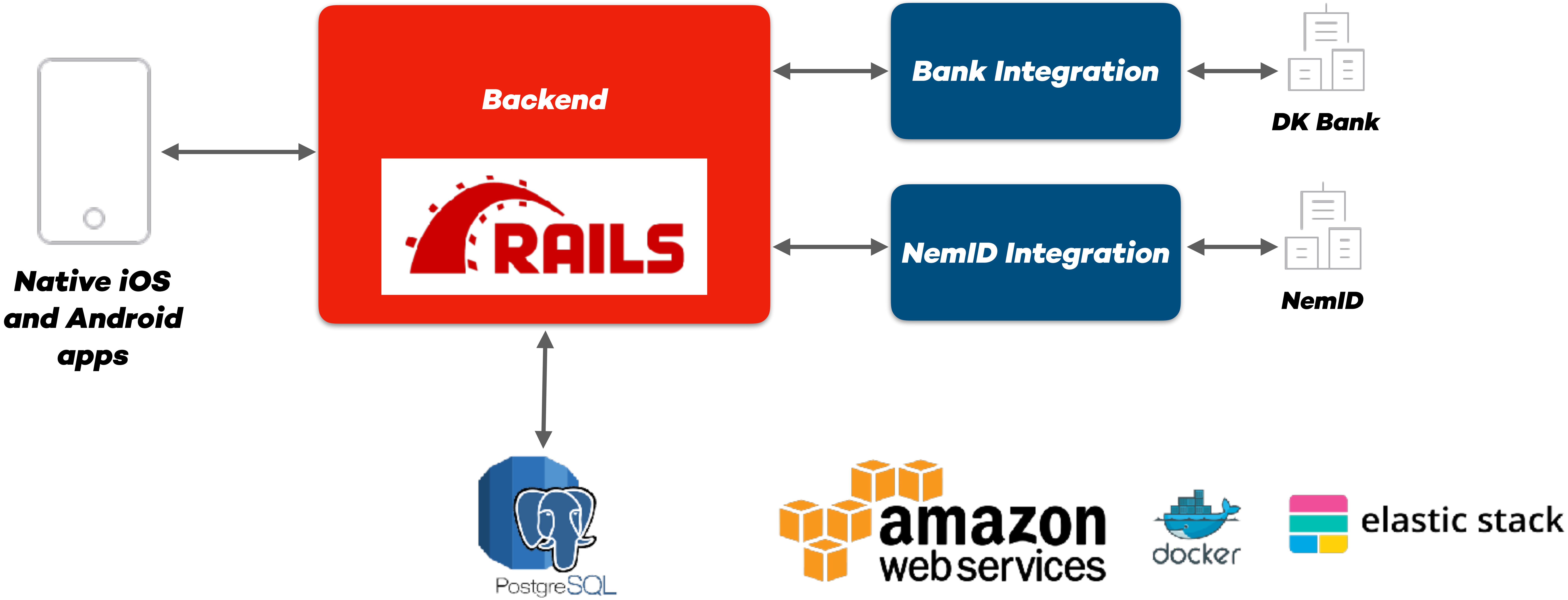
Communication patterns

Insight across service boundaries

Orchestration of business processes across services

Sharing data

Monolith in the cloud



Why split into microservices?

Goals

Scalability

Decoupling

Fast experiments

Autonomy

Velocity - Small, independent and fast deployments

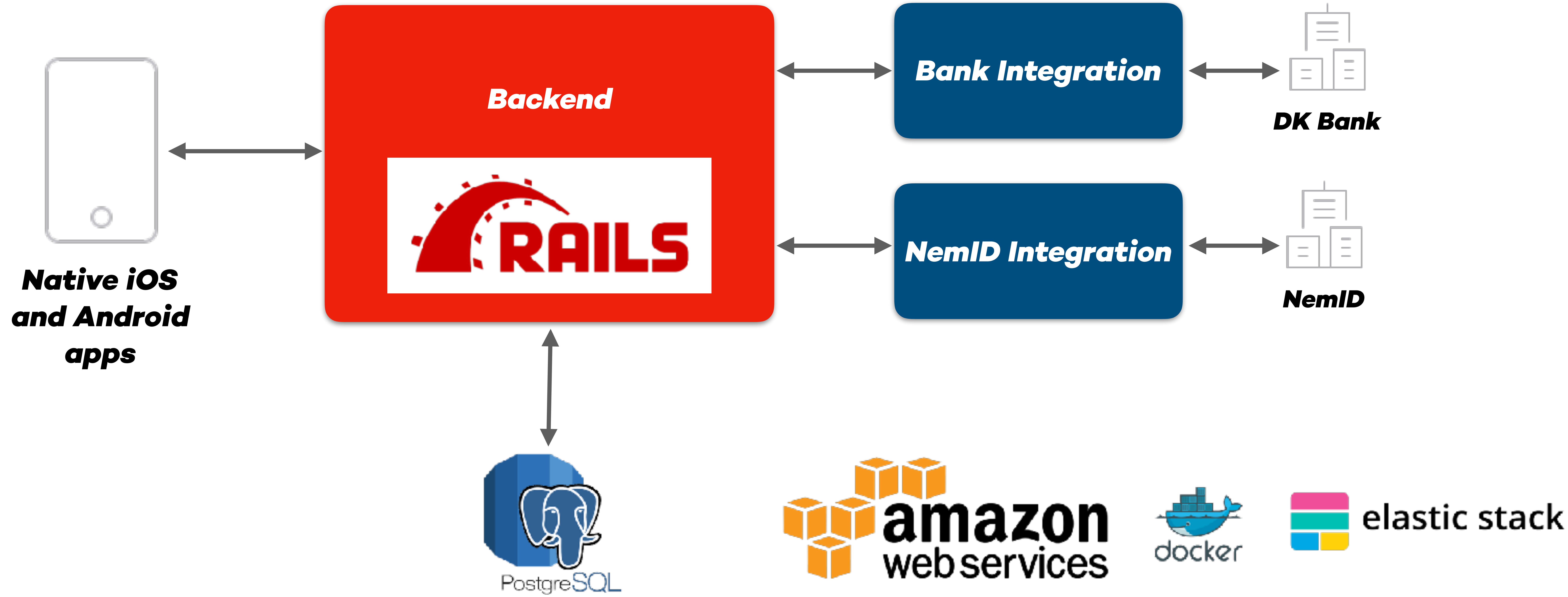
Resilience

How? Strangler application

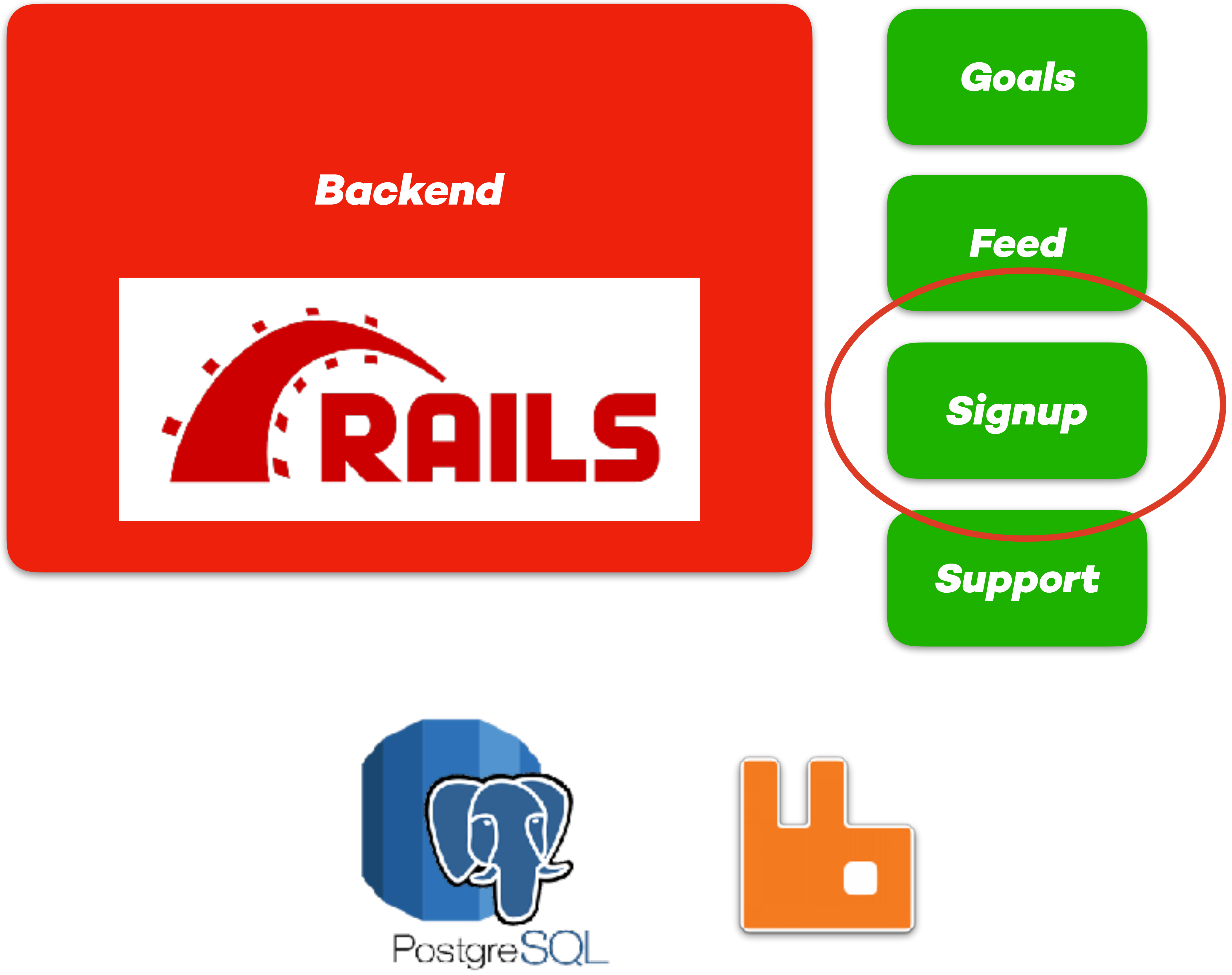
https://www.martinfowler.com/bliki/StranglerApplication.html

https://www.pexels.com/photo/gray-trunk-green-leaf-tree-beside-body-of-water-762679

Monolith in the cloud



Breaking the monolith

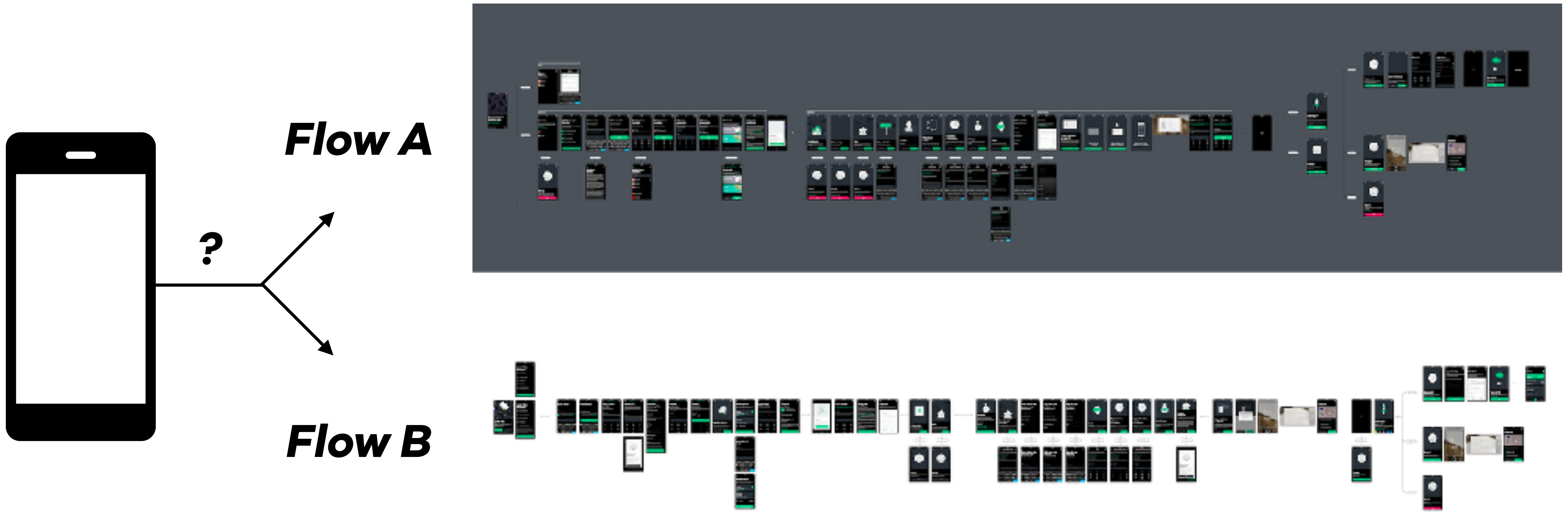


Signup

Dynamic flow with redo

Experiments

A/B testing

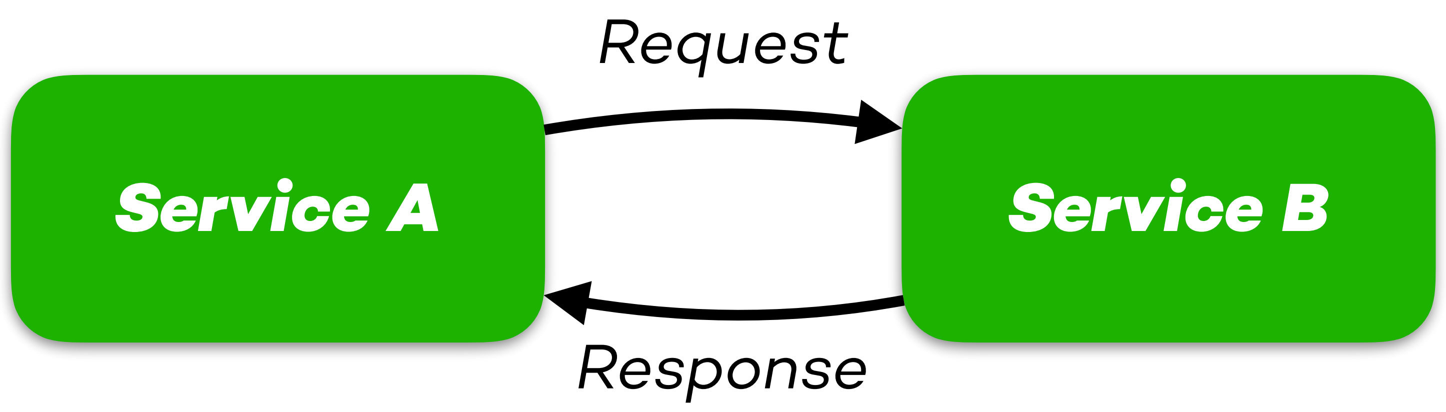


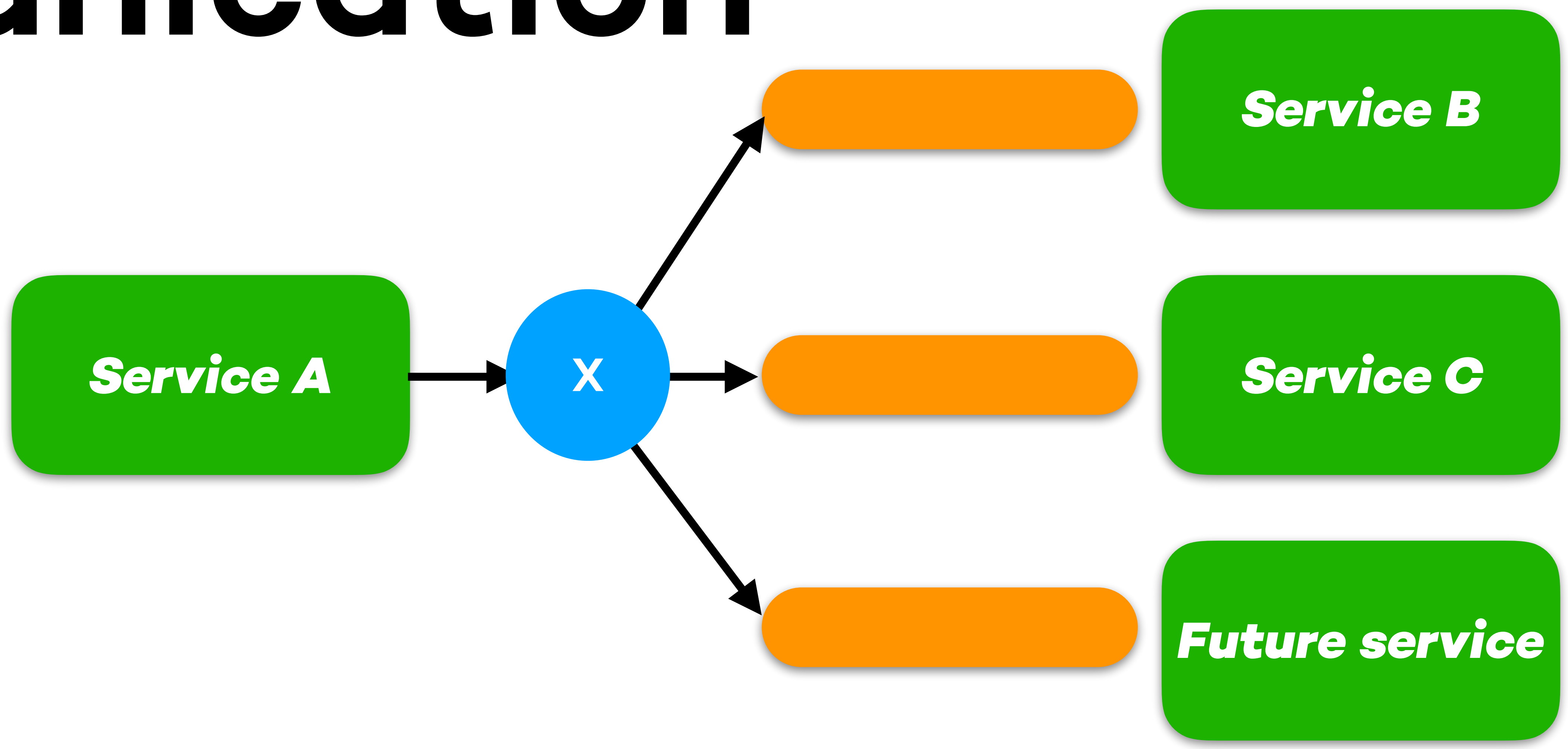
Communication

https://www.pexels.com/photo/antique-communication-phone-museum-35886

Introducing async communication

Inter service communication





Synchronous req/resp

Closed communication

High coupling

Works well with synchronous request from app

Async messages

Open ended communication (pub/sub)

Low coupling

Works bad with synchronous request from app

Might complicate flows

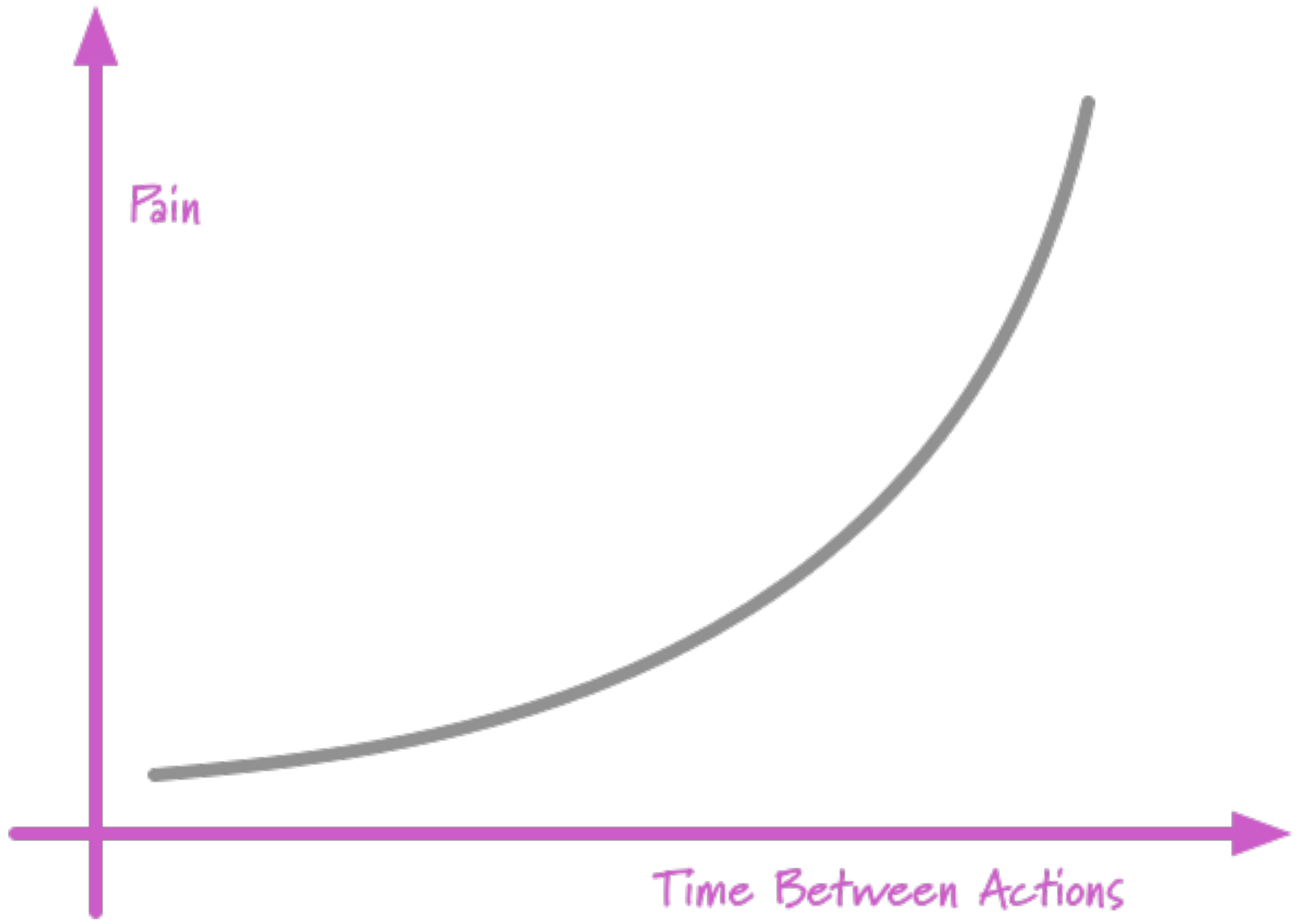
Independent deployment

https://pixabay.com/en/parachutist-skydiver-skydiving-sky-333879

Independent deployment

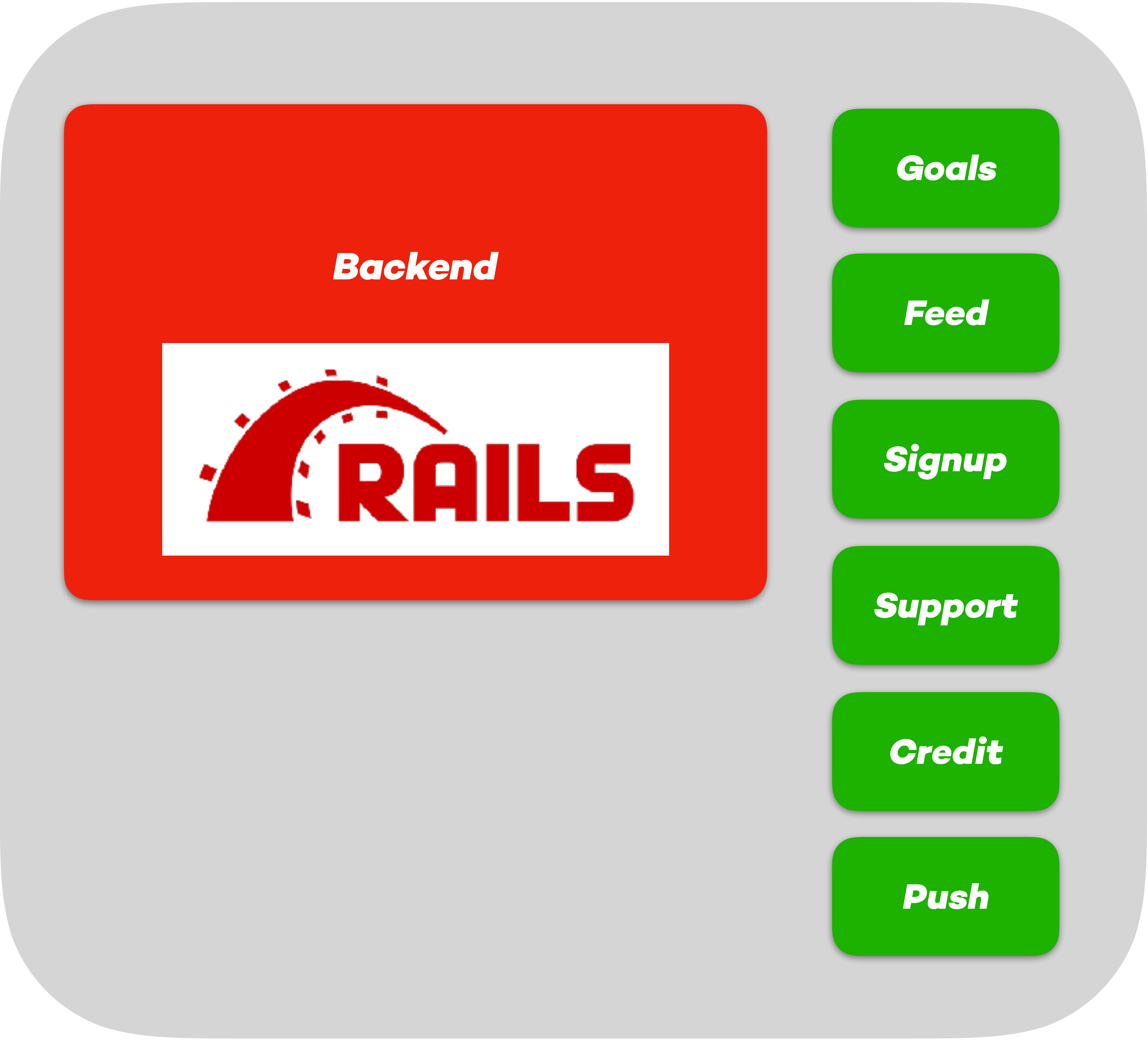
- Service increase lead to increased infrastructure needs
- Splitting the monolith without decoupling deployment was a pain
- Big Bang deployments

"If it hurts, do it more often"



https://martinfowler.com/bliki/FrequencyReducesDifficulty.html

Deployable unit old setup

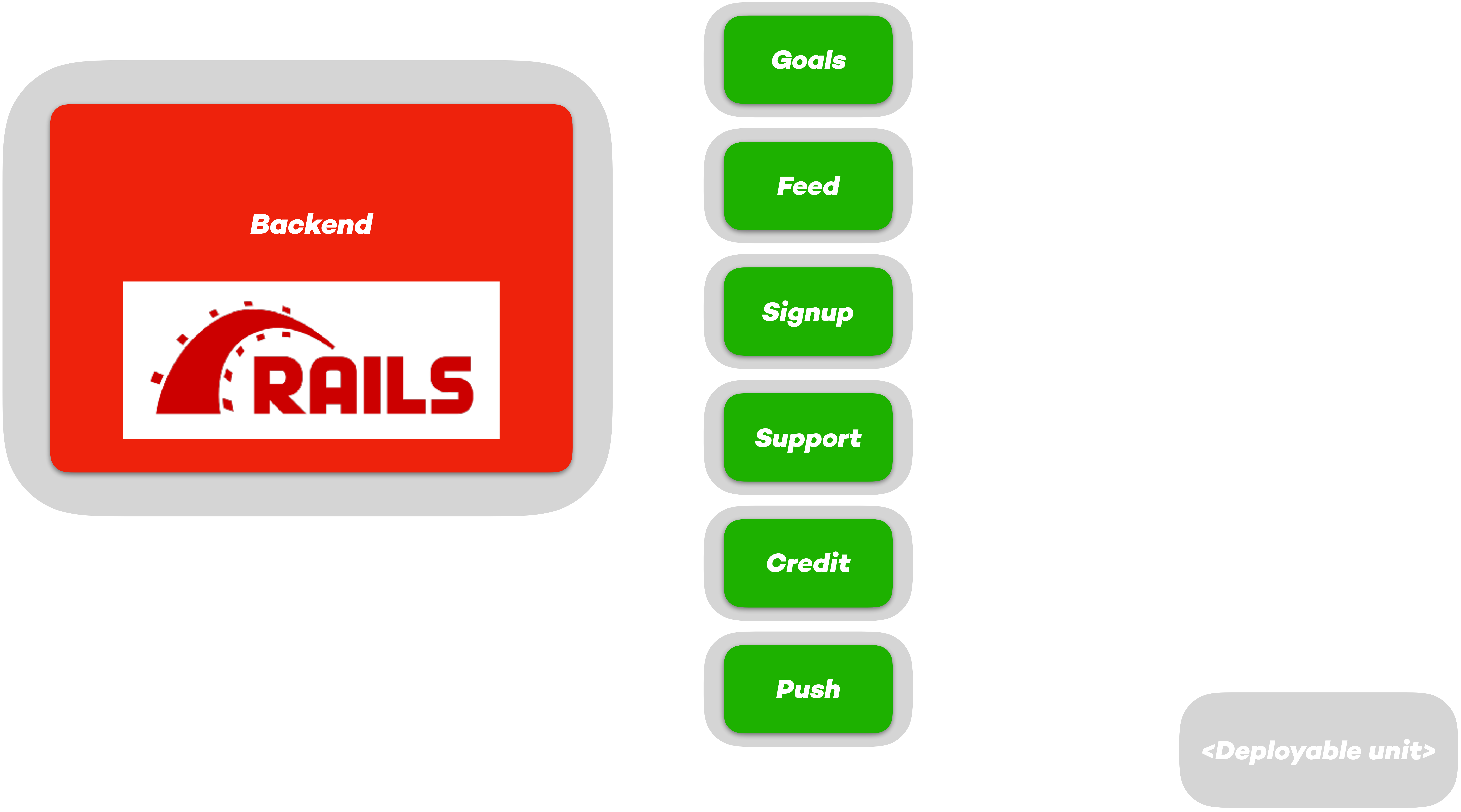


<Deployable unit>

Deploying to app-server pets

- Three VMs running Docker Compose
- All services running on all nodes (not scalable)
- Jenkins job as orchestrator (Terraform, Ansible)
- Slack synchronise which versions to deploy (bottleneck)
- Deploying was "too exciting"
- Adding a new service caused toil and risk for other services wellbeing
- SSH as an emergency handle

Deployable units with Kubernetes



Deploying to Kubernetes

- Fleet of VMs packing workload
- Deploying one service at a time
- Jenkins step per pipeline using kubectl
- Deploy whenever you want without fear
- Liveness/readiness probes catch the errors without influencing prod
- Adding a new service is easy

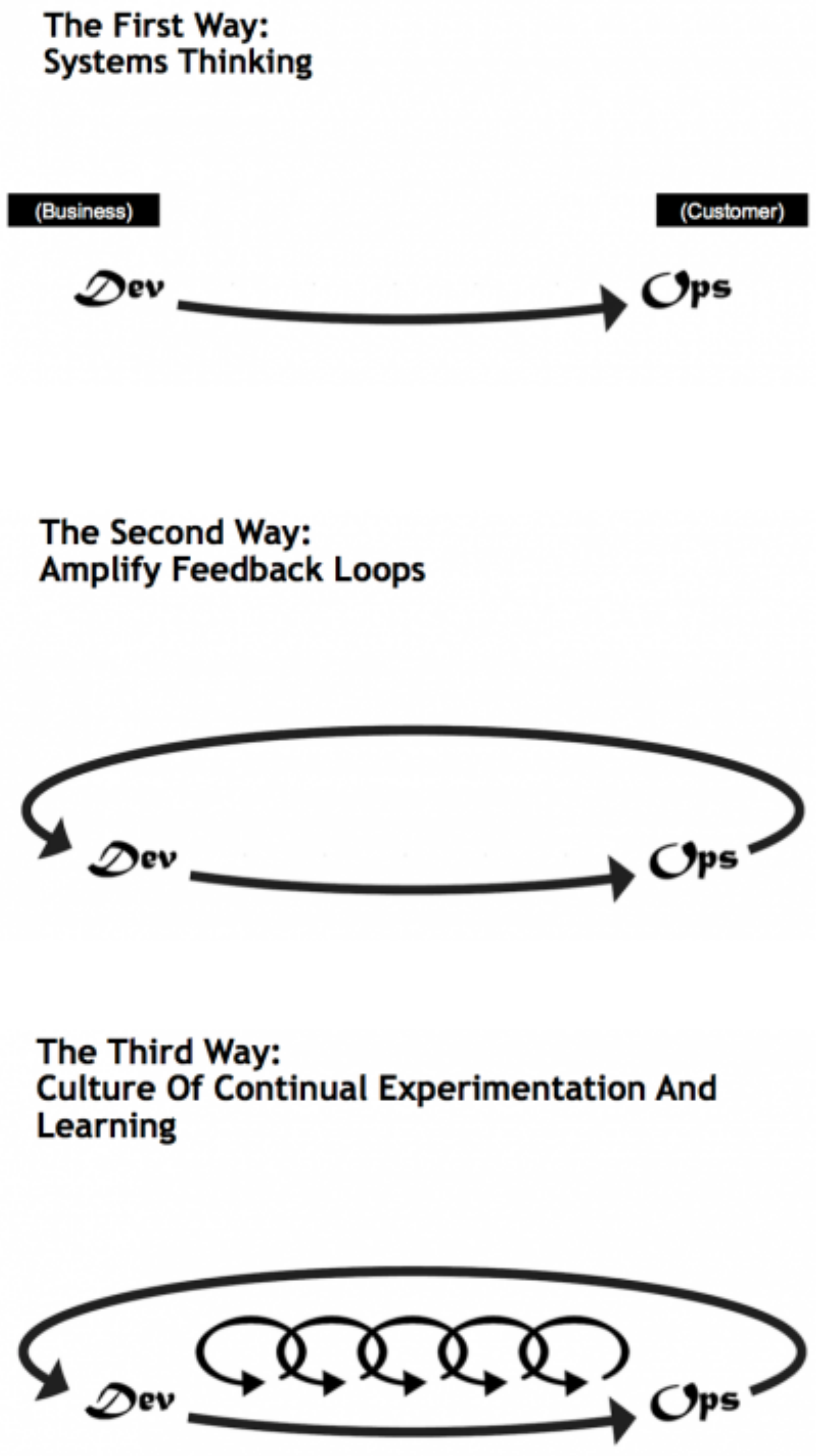
Feedback loops

https://www.pexels.com/photo/black-and-white-roller-coaster-106155

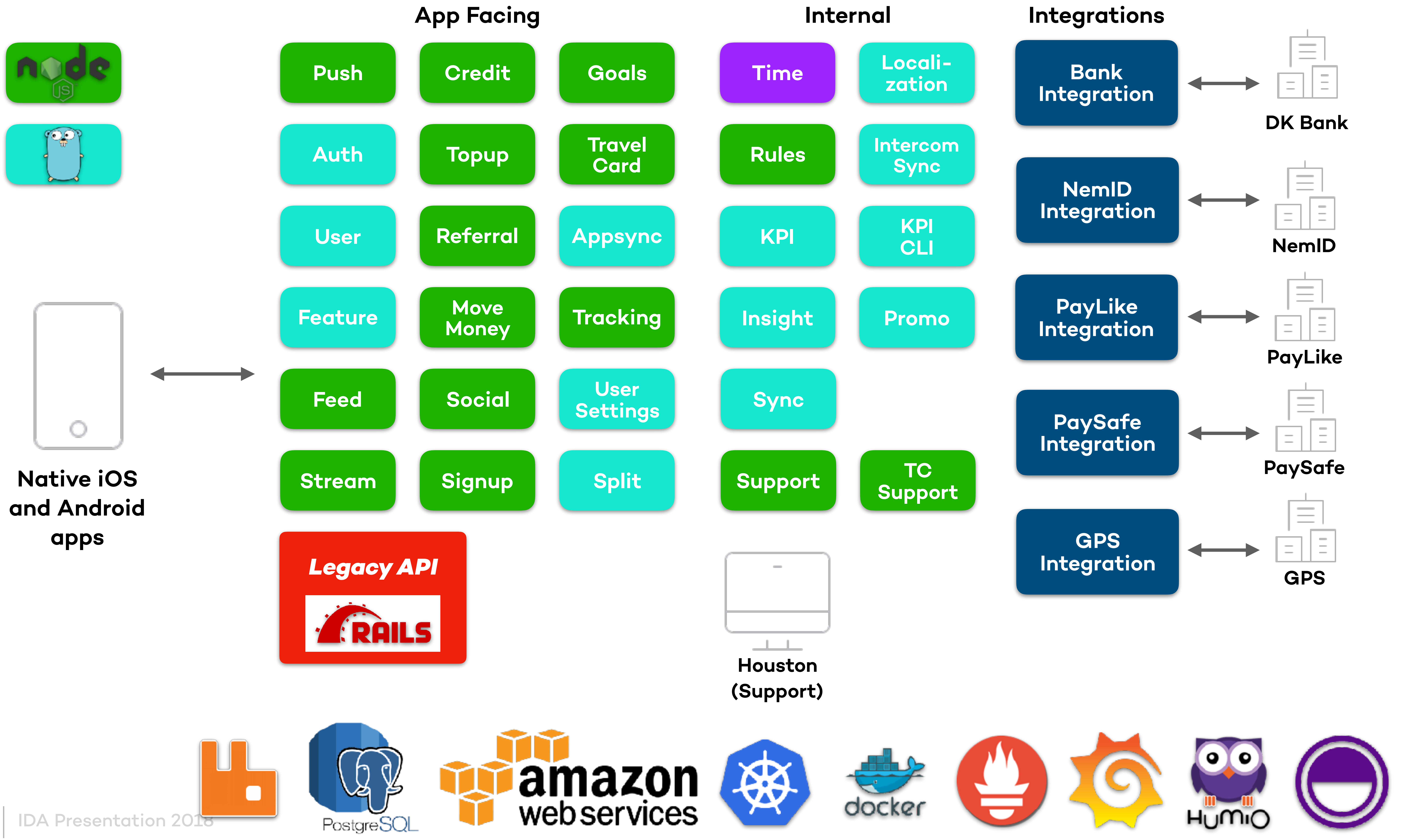
Feedback loops

- The three ways

- Reducing time from commit to production
  - Reduce risk
  - Fast feedback
  - Reduce work in progress
  - Ease debugging of bugs



http://itrevolution.com/the-three-ways-principles-underpinning-devops


App Facing
Internal
Integrations
Push
Credit
Goals
Auth
Topup
Travel Card
User
Referral
Appsync
Feature
Move Money
Tracking
Feed
Social
User Settings
Stream
Signup
Split
Legacy API
RAILS
Time
Locali-zation
Rules
Intercom Sync
KPI
KPI CLI
Insight
Promo
Sync
Support
TC Support
Houston (Support)
Bank Integration
DK Bank
NemID Integration
NemID
PayLike Integration
PayLike
PaySafe Integration
PaySafe
GPS Integration
GPS
Native iOS and Android apps
PostgreSQL
amazon web services
docker
HUMIO

What's next?

- Utilising Custom Resource Definitions and a controller to do Release Management - Moving towards **GitOps**
- Services Mesh, Istio is now 1.0
- Adopting more Operators to ease operations of e.g. Prometheus.
- Provide a FaaS on top of Kubernetes
- Extend Kubernetes with virtual-kubelet for additional serverless resources
- Solve the big pain of local development?

Wrapping up

Key takeaways if entering Cloud Native and Microservices

Kubernetes is complex and has a steep learning curve, but it enables so many possibilities

Prioritise your infrastructure to unlock the potential of Microservices

Monitoring and alerts are very important in such a dynamic environment, but be aware of alert fatigue

Read your logs and make them easily accessible for all your developers

If it hurts, do it often

lunar way®

Questions?

Thank you!

@phennex
@mrjensens