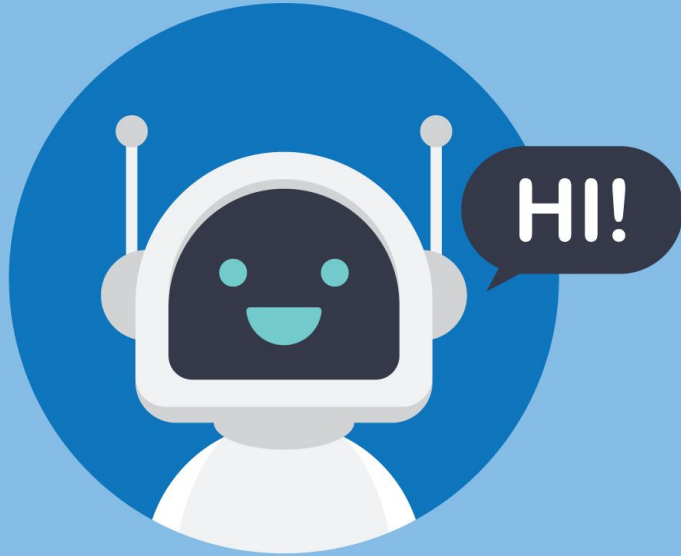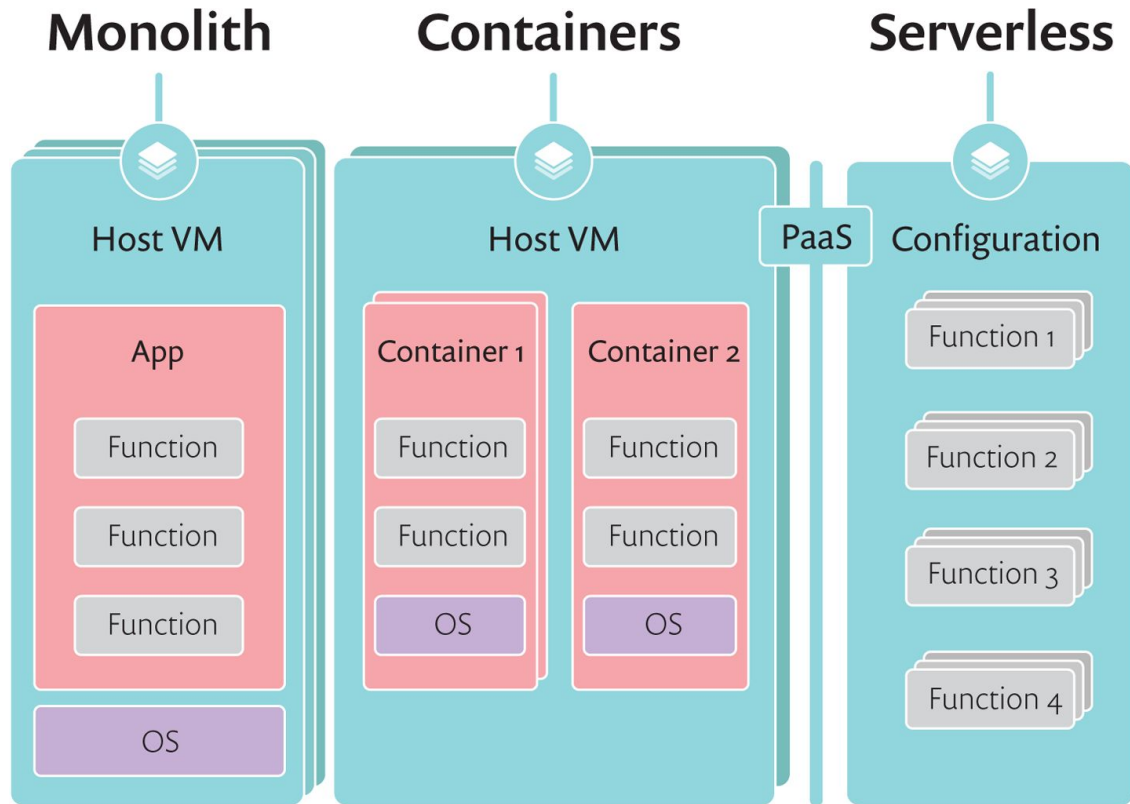# About me

- Operating Systems Professor
- Entrepreneur(cloudsociety.com)
- Cloud Native enthusiast
- SRE at Wizeline
- (Cloud-Native)+GT Meetup Organizer

Sergio Méndez

IT Infrastructure Evolution

Monolith

Host VM

App

- Function
- Function
- Function

OS

Containers

Host VM

Container 1

- Function
- Function
- OS

Container 2

- Function
- Function
- OS

PaaS

Serverless

Configuration

- Function 1
- Function 2
- Function 3
- Function 4

Ref: Tal, L. (2019). Serverless Security. O'Reilly Media.

Basic concepts

#1 Cloud Native

Cloud-native is an approach to building and running applications that exploits the advantages of the cloud computing model.

#2 Cloud Native

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.

What is serverless?

"An architectural approach to software solutions that relies on small independent functions running on transient servers in an elastic runtime environment."

Ref: Amundsen M. (2020). What is serverless?. O'Reilly Media

Serverless means a high level server abstraction, so you don't need to manage your servers anymore.

Ref: Amundsen M. (2020). What is serverless?. O'Reilly Media

Serverless Platform features

- Event-Driven
- Streaming Data
- Auto-Scaling
- Fault Tolerance

Ref: Amundsen M. (2020). What is serverless?. O'Reilly Media

The Goods and the Bads

- Cheaper than the traditional cloud
- Scalable
- Lower Ops costs
- Focus on user experience/Devs

- Vendor lock-in
- Learning curve
- Unsuitable for long-term tasks
- Doesnt support many languages

Ref: https://dzone.com/articles/the-good-and-the-bad-of-serverless-architecture
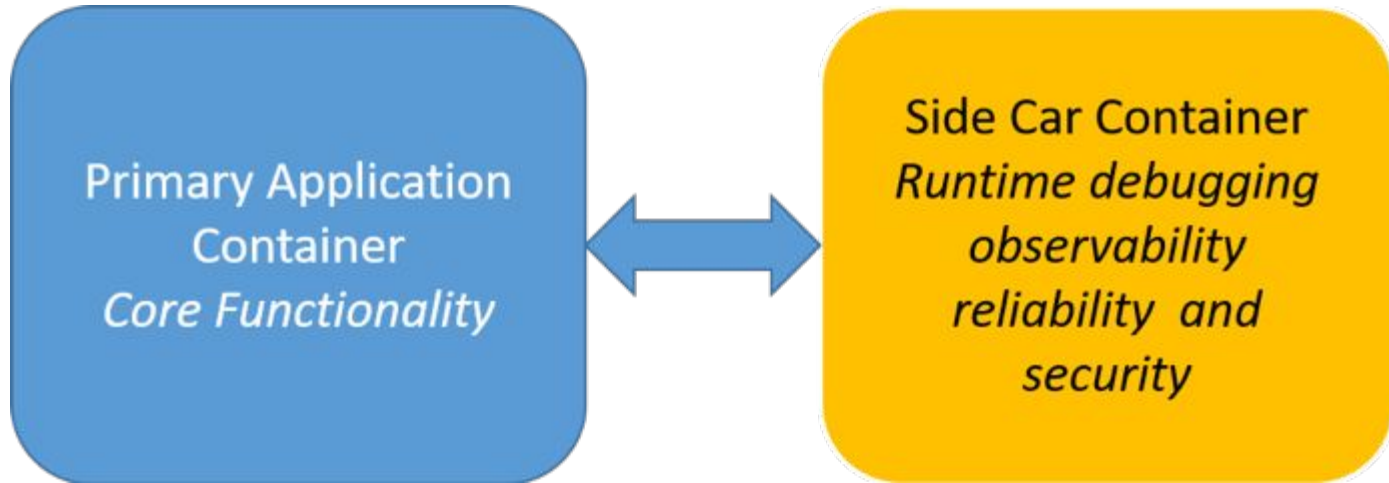
About controlling the system

"they have a great deal of control over the design and definition of the solution but very little control over the system once it is deployed to the platform."

Ref: Amundsen M. (2020). What is serverless?. O'Reilly Media
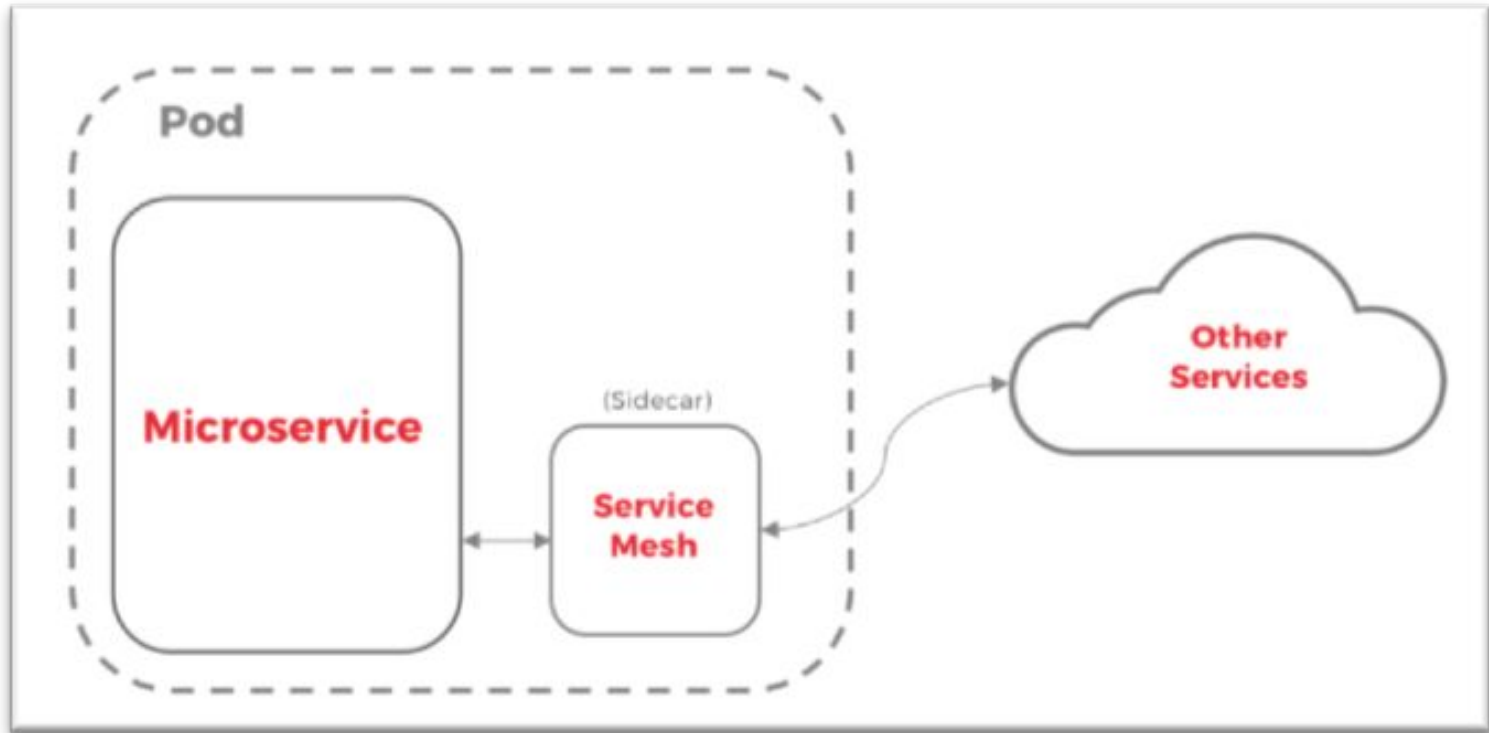
Use Cases

- Internet of Things applications
- Virtual assistants and chatbots
- Auto-scaling Websites and APIs
- Event streaming(pub/sub,logs)
- Image and Video Manipulation
- Processing Events and SaaS
- Multi-language Applications
- CI/CD

Ref: https://serverless.com/learn/use-cases/

What is Service Mesh?

Is a dedicated infrastructure layer for handling service-to-service communication in order to make it visible, manageable, and controlled. Generally every service mesh is implemented as a series (or a "mesh") of interconnected network proxies designed to better manage service traffic.
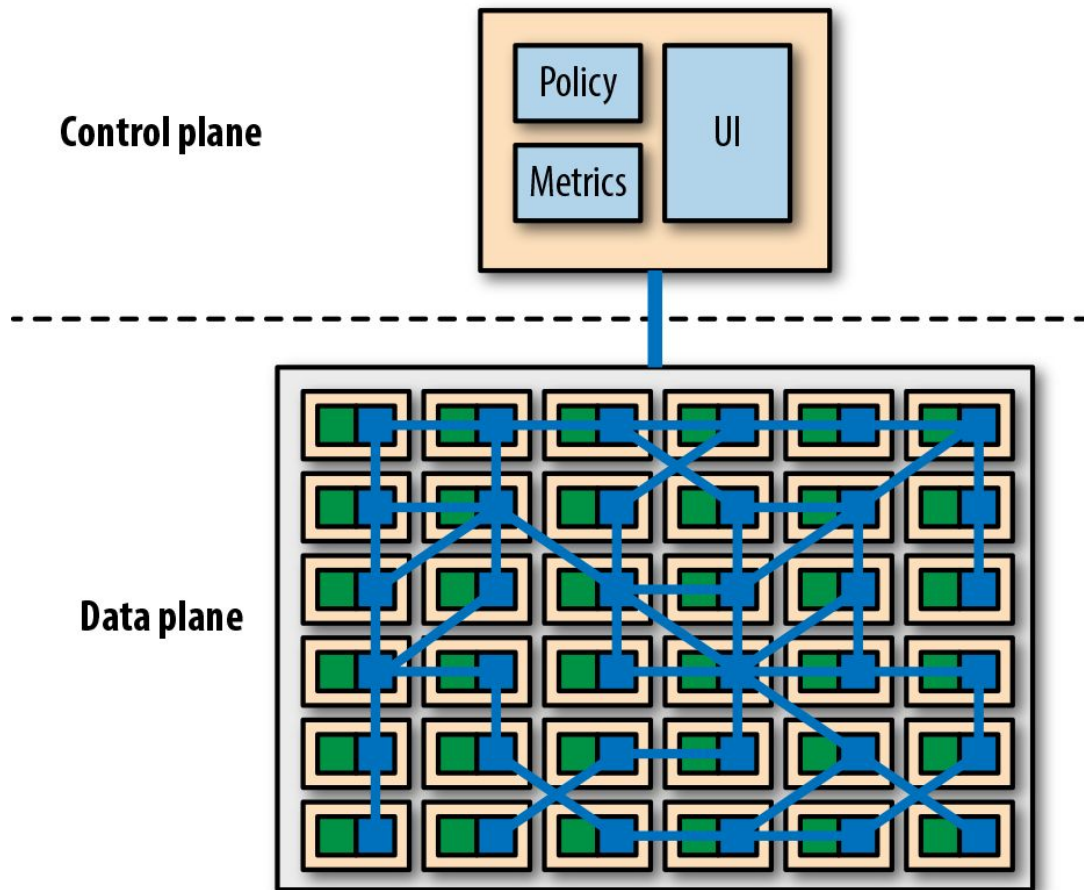
Ref: Miranda G. (2018). The Service Mesh. O'Reilly Media.

Ref: https://containerjournal.com/topics/container-ecosystems/what-is-service-mesh-and-why-do-we-need-it/

Service Mesh has 2 components:

- data plane
- control plane

Control plane

Policy

Metrics

UI

Data plane

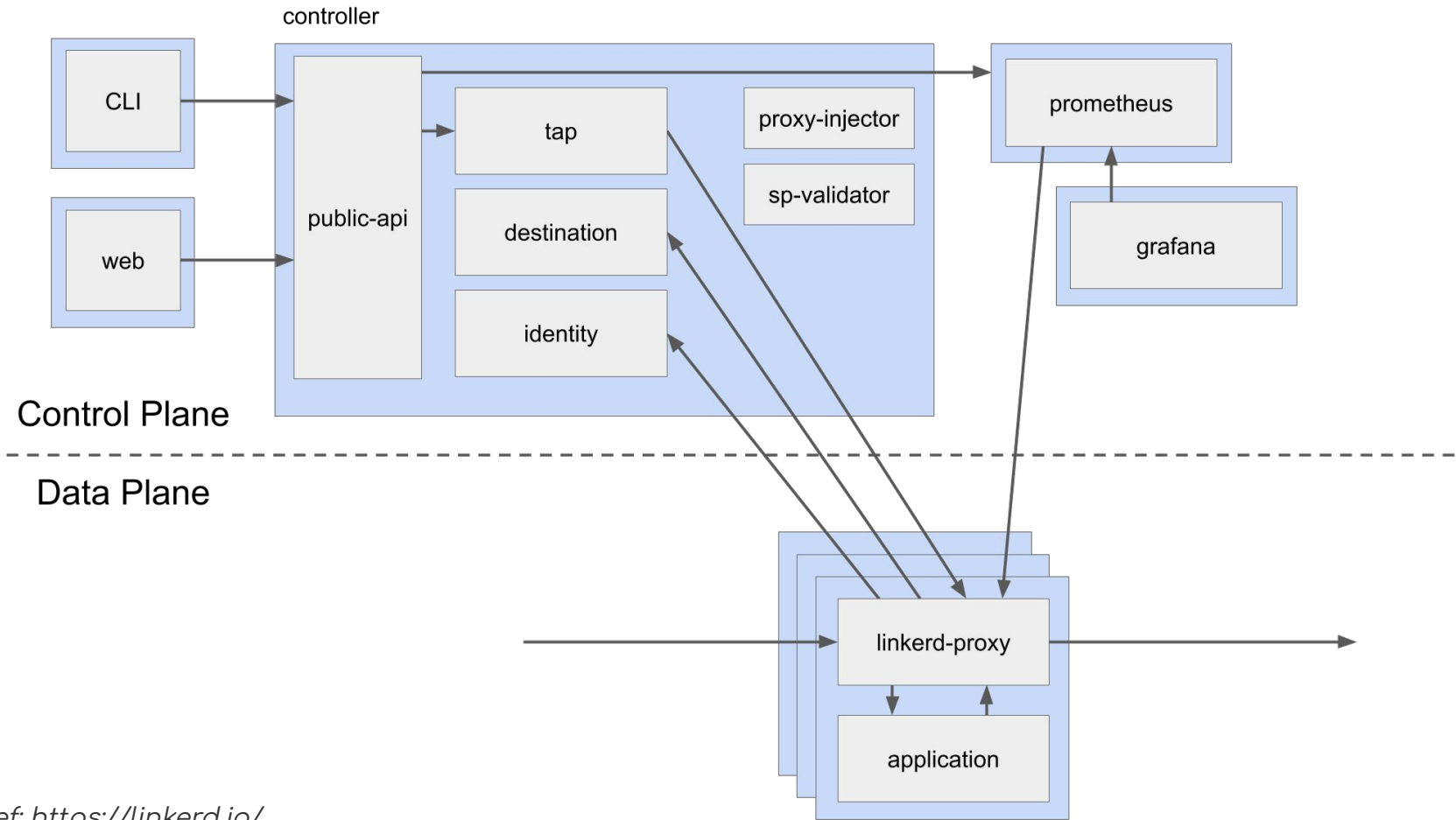*Ref: Miranda G. (2018). The Service Mesh. O'Reilly Media.*

LINKERD

Linkerd is an ultralight service mesh for Kubernetes. It gives you observability, reliability, and security without requiring any code changes.

controller

CLI

web

public-api

tap

destination

identity

proxy-injector

sp-validator

prometheus

grafana

Control Plane

Data Plane

linkerd-proxy

application

Ref: https://linkerd.io/

What does Linkerd do?

- **Observability**: Service-level *golden metrics*: success rates, latencies, throughput. Service topologies.
- **Reliability**: Retries, timeouts, load balancing, circuit breaking
- **Security**: Transparent mTLS, cert management and rotation, policy

In an ultralight package focused on **operational simplicity** first and foremost.

Ref: https://linkerd.io/
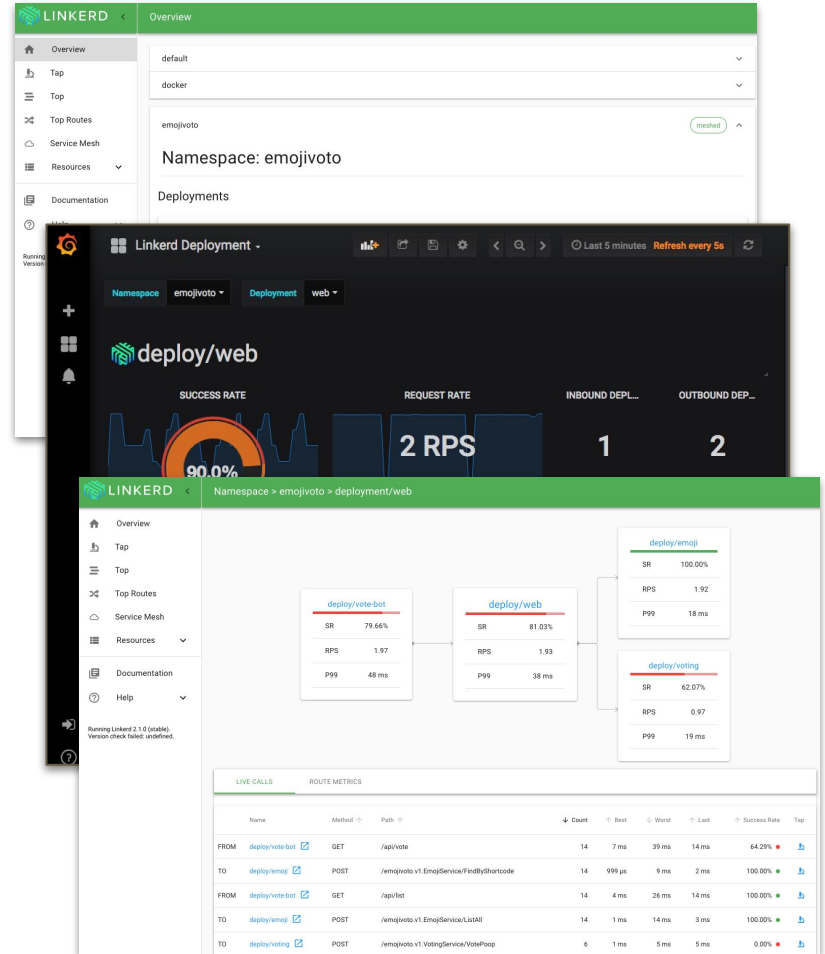
Why should I care?

Linkerd gives ...

... **platform owners** (SREs, architects)

... the **observability**, **reliability**, and **security** primitives

... that are **critical** for cloud native architectures

... with **no developer involvement!**

👉 Who?

👉 What?

👉 Why?

👉 The magic

Linkerd doesn't just solve technical problems, it solves **socio-technical problems**: by decoupling them developers, it gives platform owners control over their destiny.

Ref: https://linkerd.io/

LINKERD

OpenFaaS

OpenFaaS is an open source serverless platform to provide an easy for developers to **deploy event-driven functions and microservices to Kubernetes** without repetitive, boiler-plate coding. **Package** your code or an existing binary in a **Docker** image to get a highly scalable endpoint with auto-scaling and metrics.

Ref: https://www.openfaas.com/

Features

- Ease UI portal and one-click install functions
- Package functions in any language for Linux/Windows using Docker/OCI image
- Portable, runs on existing hardware, public/private cloud, Kubernetes and Docker Swarm native
- CLI, YAML format for templating/defining functions
- Auto-scales as demand increases

Ref: https://www.openfaas.com/

Reference Architecture

Ref: https://docs.openfaas.com/architecture/stack/#conceptual-workflow
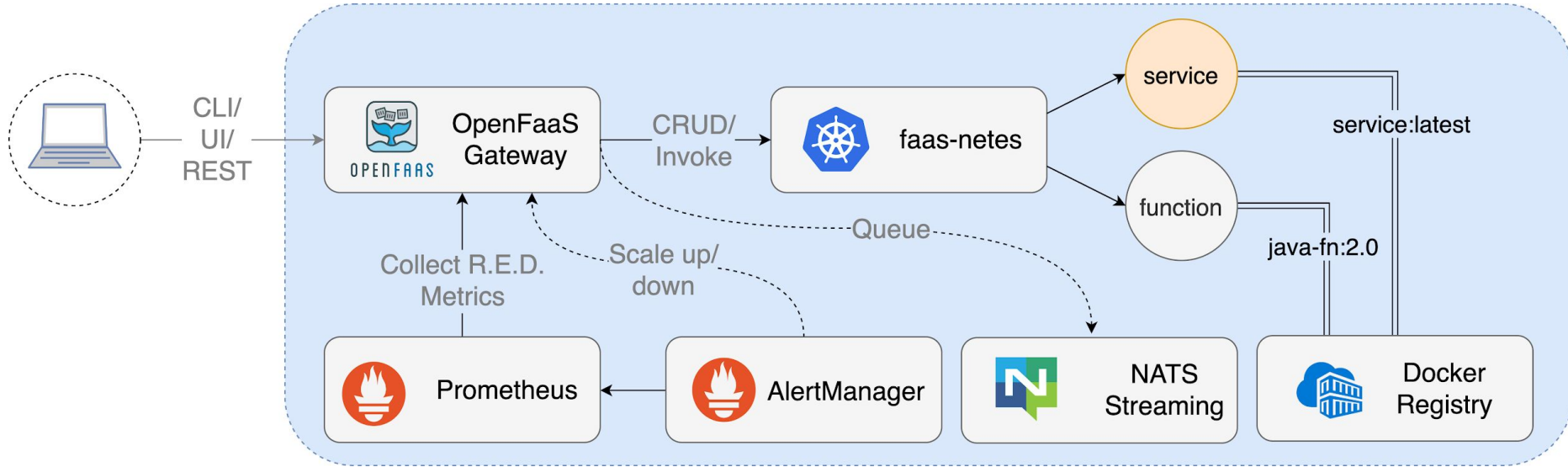
Serverless Real Use Case

Telecom needs

Build a tool to create assisted chatbots that automate sales, support, monitoring, etc., using IM APIs from Slack, Telegram, Whatsapp, Facebook, etc.

Telecom Application requirements

- Prevent vendor lock-in
- Fast integration with technologies
- Short development time
- Implement DevOps & CI/CD
- Spend less money on cloud infrastructure
- Performance
- Resilience system
- Observability

Solution

- OpenFaaS(Serverless Platform)
- Linkerd(Canary Deployments, Observability)
- Kubernetes(Scaling & Orchestration)

Chatbot interactions

Instant Messengers

Platform

percepts

sensors

environment

?

agent

actions

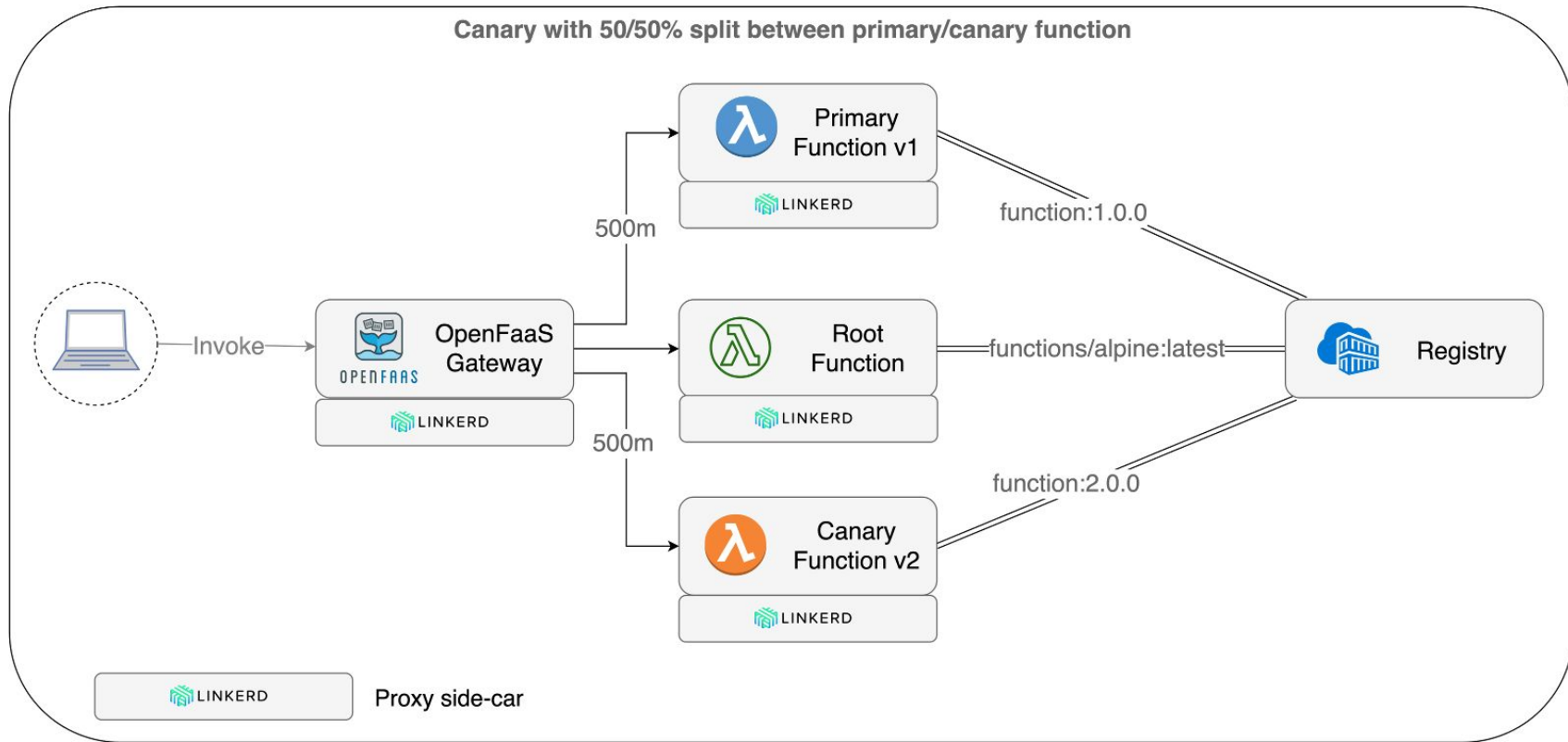effectors
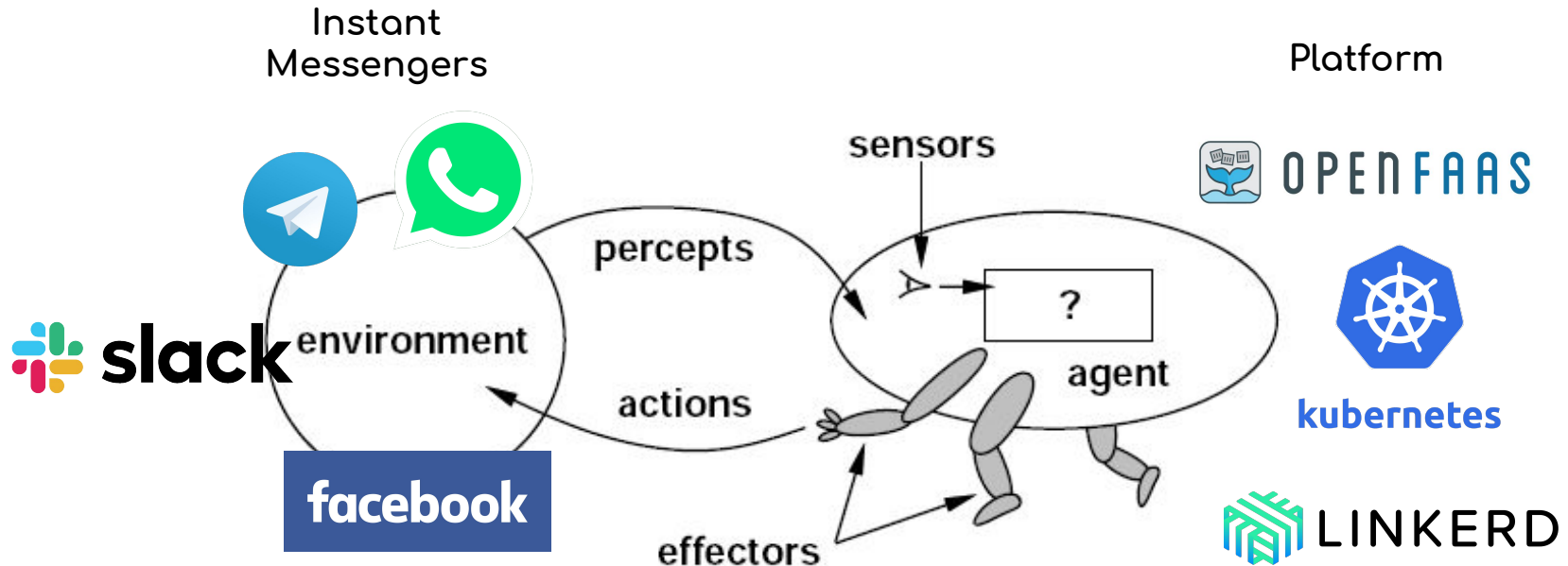
Canary, Blue/Green Deployments

Blue/Green

"The blue/green deployment strategy differs from a ramped deployment, version B (green) is deployed alongside version A (blue) with exactly the same amount of instances. After testing that the new version meets all the requirements the traffic is switched from version A to version B at the load balancer level."

Canary

"A canary deployment consists of gradually shifting production traffic from version A to version B. Usually the traffic is split based on weight. For example, 90 percent of the requests go to version A, 10 percent go to version B."

Ref: https://thenewstack.io/deployment-strategies/

Serverless Chatbot Architecture

Canary with 50/50% split between primary/canary function

Invoke

OpenFaaS Gateway
LINKERD

500m

Primary Function v1
LINKERD

Root Function
LINKERD

Canary Function v2
LINKERD

500m

function:1.0.0

functions/alpine:latest

function:2.0.0

Registry

LINKERD   Proxy side-car

Ref: https://github.com/openfaas-incubator/openfaas-linkerd2

Benefits for this solution

- Zero vendor lock-in
- Fast integration with technologies
- Short development time
- CI/CD ready platform
- Spend less money in a smart way
- Autoscalable application
- Resilience system
- Observability without modifications

Demo

This demo shows:

- Claudia Slack Chatbot
- OpenFaaS functions templates
- Linkerd integrations
- Canary deployments
- Observability

Repository

https://github.com/sergioarmgpl/
linkerd-openfaas-kubeconeu-2020

References

- George Miranda, The Service Mesh, O'Reilly Media
- Liran Tal. Serverless Security. O'Reilly Media
- Linkerd, https://linkerd.io
- OpenFaas, https://www.openfaas.com
- Container Journal, https://bit.ly/38qR3I9
- Imperial College, Agents. https://bit.ly/2UmXFRV
- Serverless Use Cases, https://bit.ly/33uDO7F

Follow me

@sergioarmgpl

Thanks