ORACLE

Cloud Native Labs

Serverless Patterns

Design and Use Patterns in Serverless

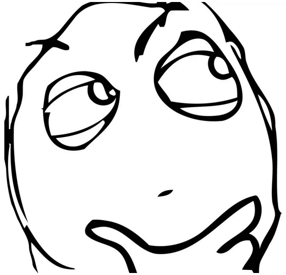
Jesse Butler Cloud Developer Advocate, Oracle Cloud Infrastructure @jlb13

#OracleCloudNative

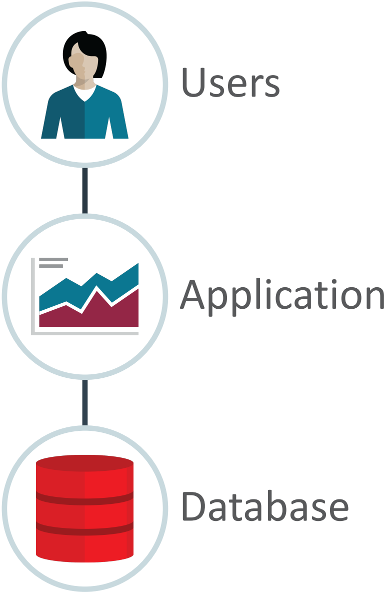
cloudnative.oracle.com

Level Set

- Roles in the room?
- Serverless users?
- In production?
- Lambda? Azure? Something Else?



Monolithic Applications

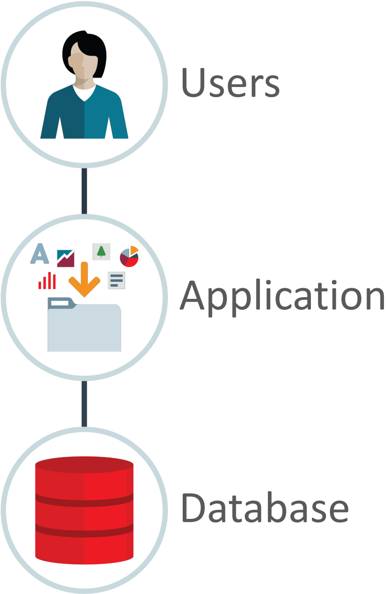


Users

Application

Database

Monolithic Applications

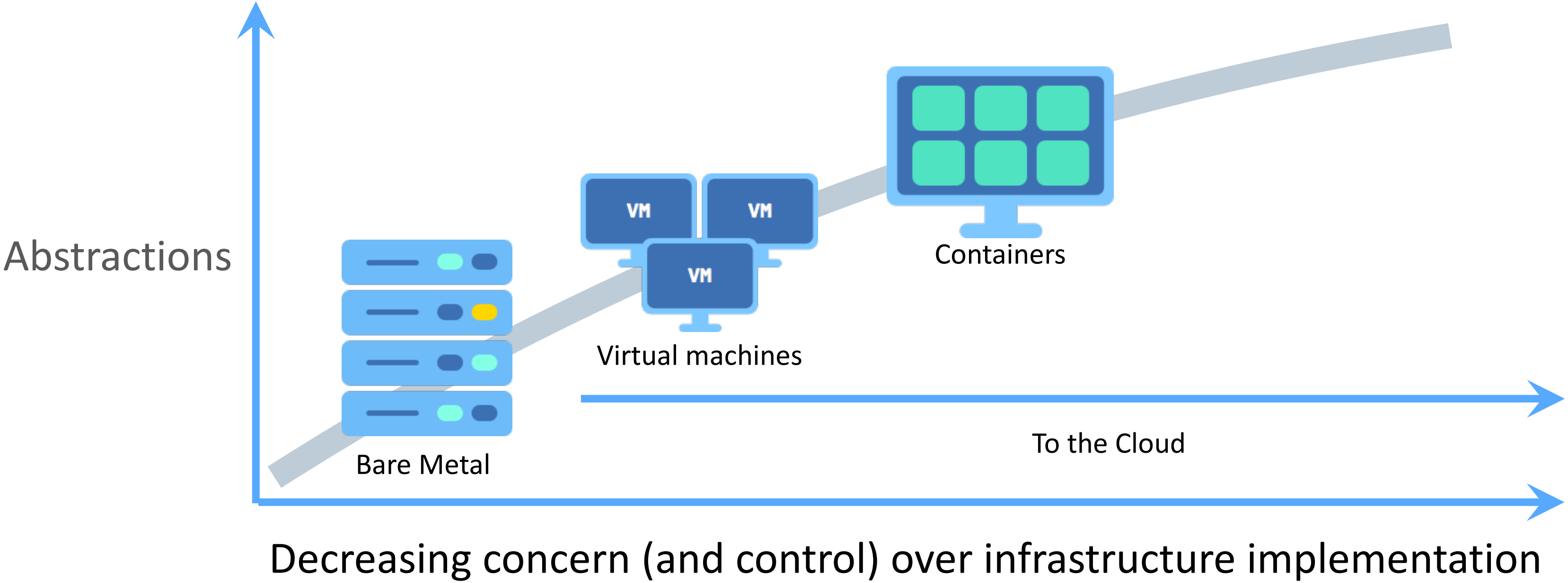


Users

Application

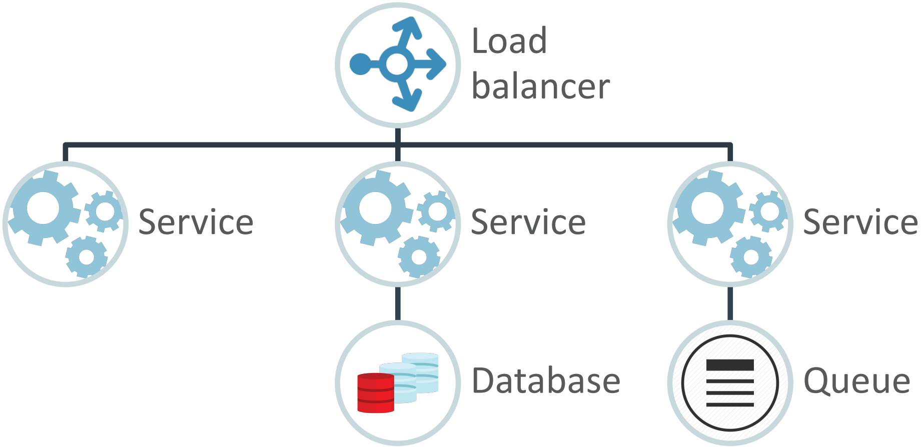
Database

Virtualization and Consolidation


Abstractions
VM
VM
VM
Containers
Virtual machines
Bare Metal
To the Cloud
Decreasing concern (and control) over infrastructure implementation

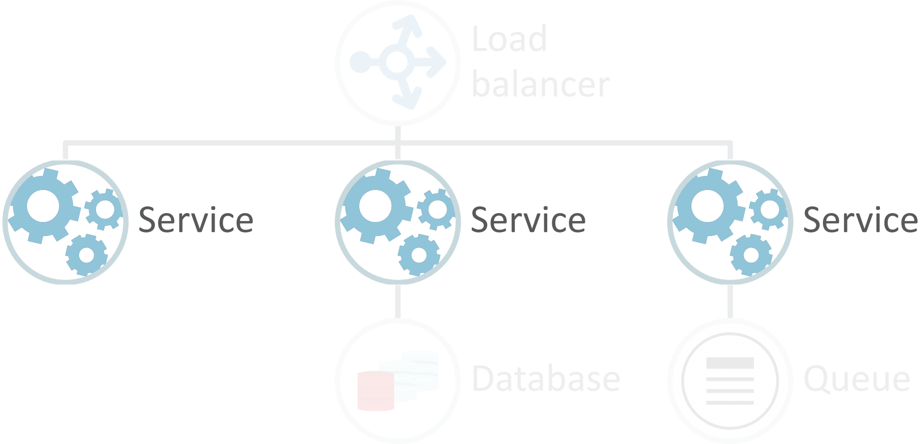
Microservices

Deploying Code to Systems We Build in the Cloud with Containers and Kubernetes

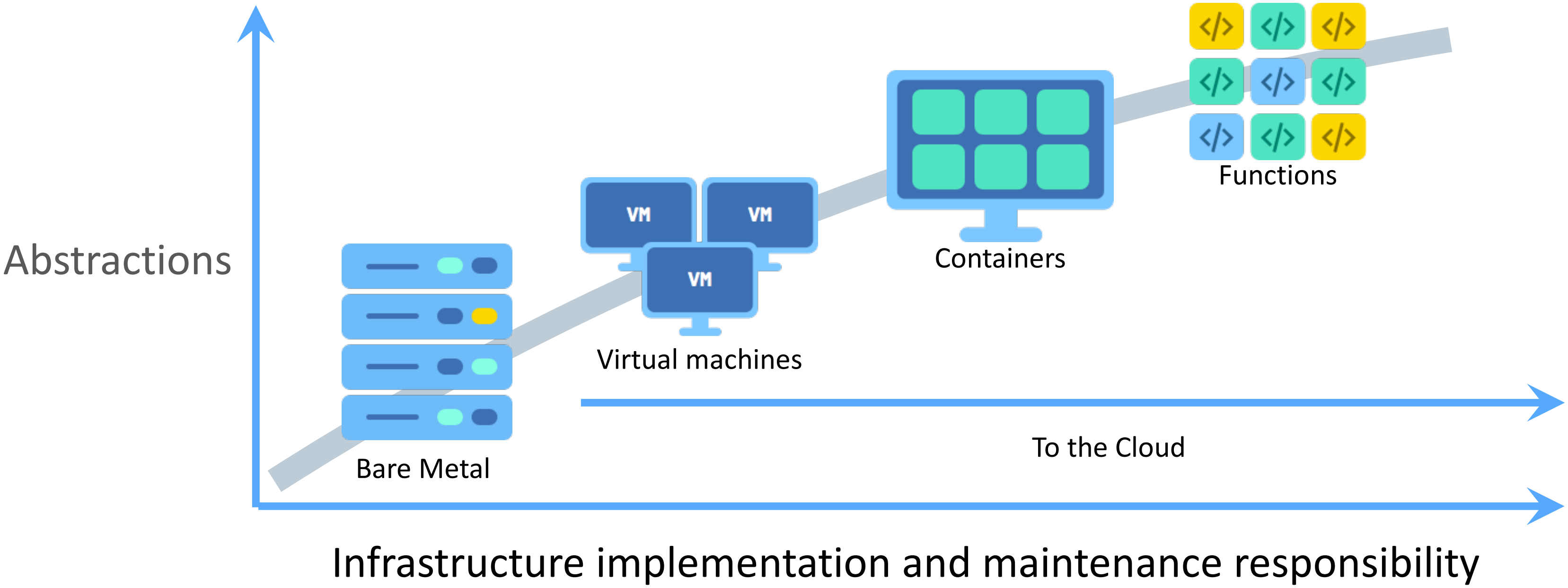


Serverless

Deploying Code to Systems We Build in the Cloud with Containers and Kubernetes



Trend towards Serverless



Serverless is a Spectrum

Container Orchestration
Kubernetes
Nomad
Docker Swarm

Ideally managed, but still infrastructure you care about

Container Services
ECS, Fargate
Azure CS
GCP Cloud Run

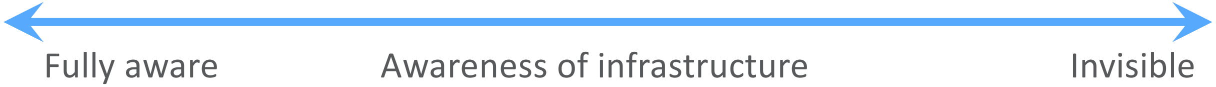
Managed service provision containers, abstracts infrastructure you care about

DIY FaaS
OpenFaaS
Fn Project

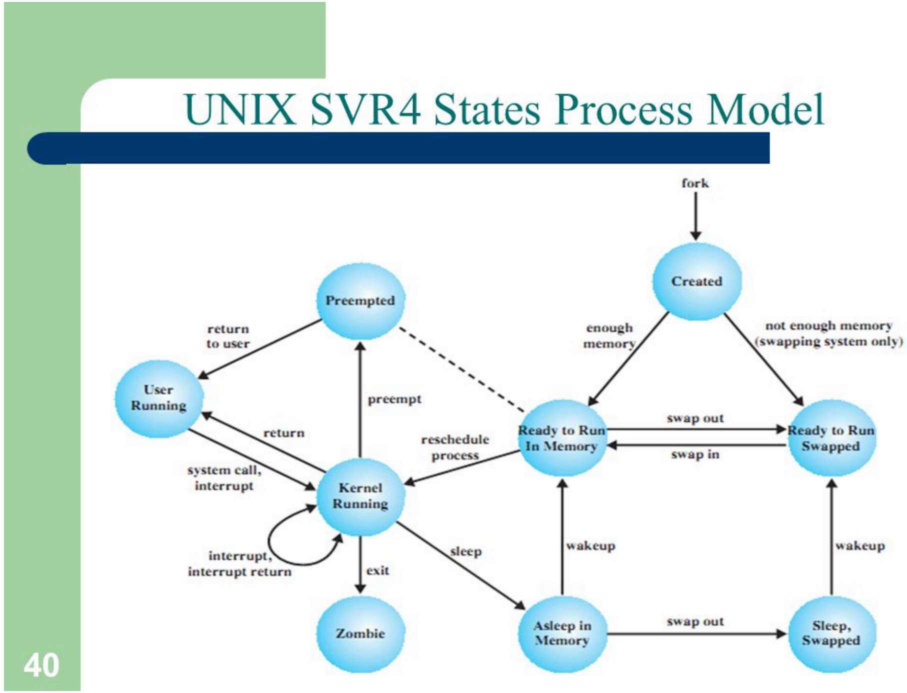
Leverage container management system under the covers, introduce Functions architecture

Managed Serverless
AWS Lambda
Azure Functions
Google Functions
Oracle Functions

Fully managed platform for hosting and executing code



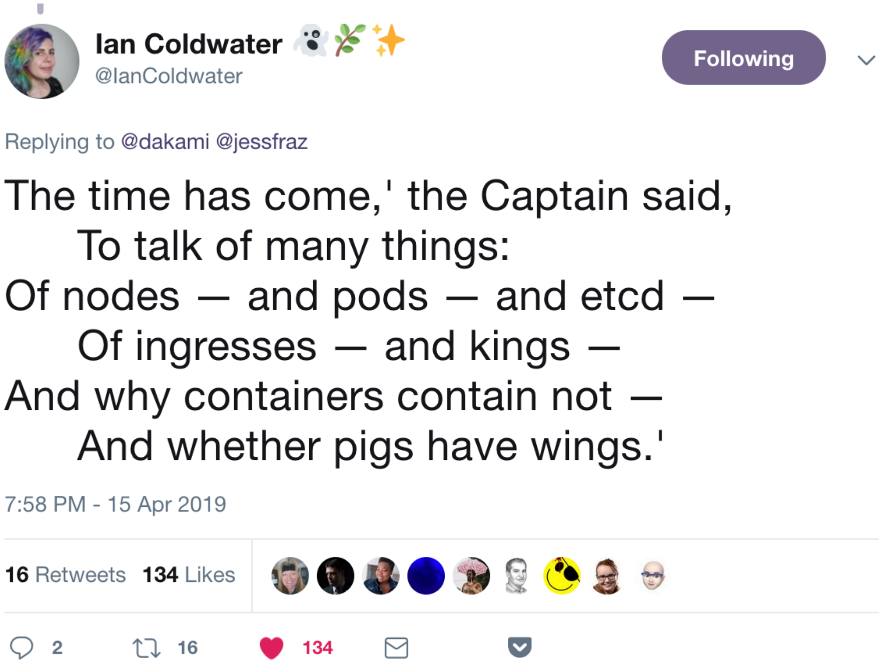
Containers are the new Process Model, Right?



UNIX SVR4 States Process Model

40

Containers are the new Process Model, Right?



Ian Coldwater 👻🌿✨
@IanColdwater

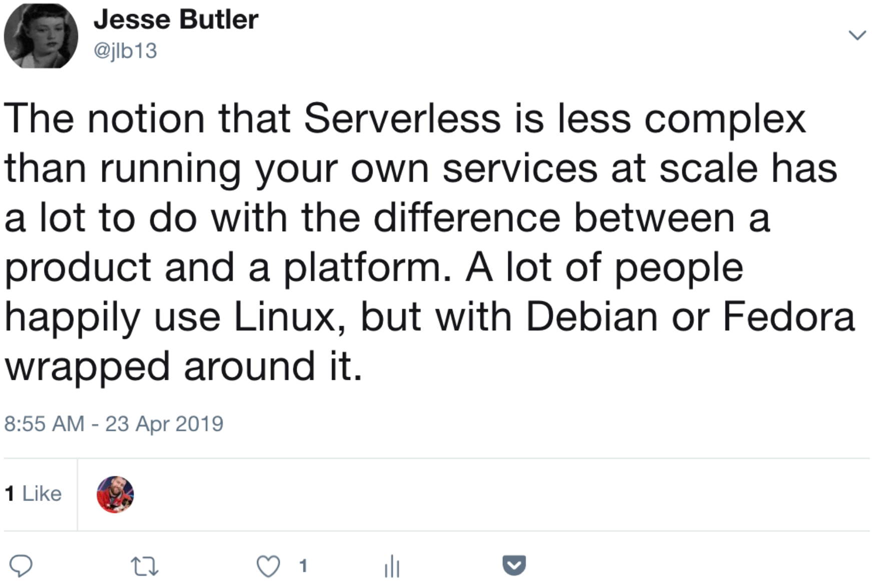
Replying to @dakami @jessfraz

The time has come,' the Captain said,
To talk of many things:
Of nodes — and pods — and etcd —
Of ingresses — and kings —
And why containers contain not —
And whether pigs have wings.'

7:58 PM - 15 Apr 2019

16 Retweets **134** Likes

Don’t Conflate Requirements with Complexity Aversion



Jesse Butler
@jlb13

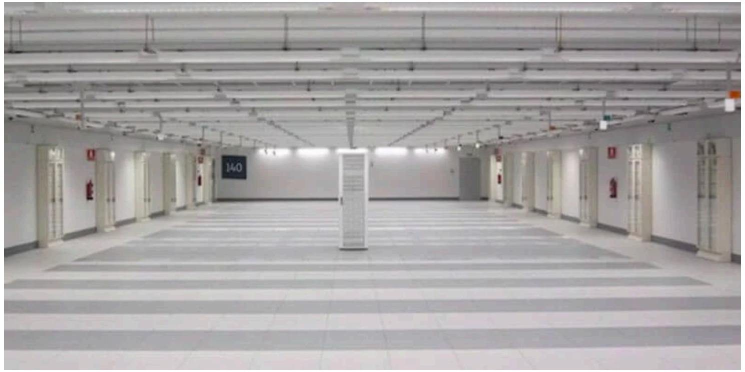
The notion that Serverless is less complex than running your own services at scale has a lot to do with the difference between a product and a platform. A lot of people happily use Linux, but with Debian or Fedora wrapped around it.

8:55 AM - 23 Apr 2019

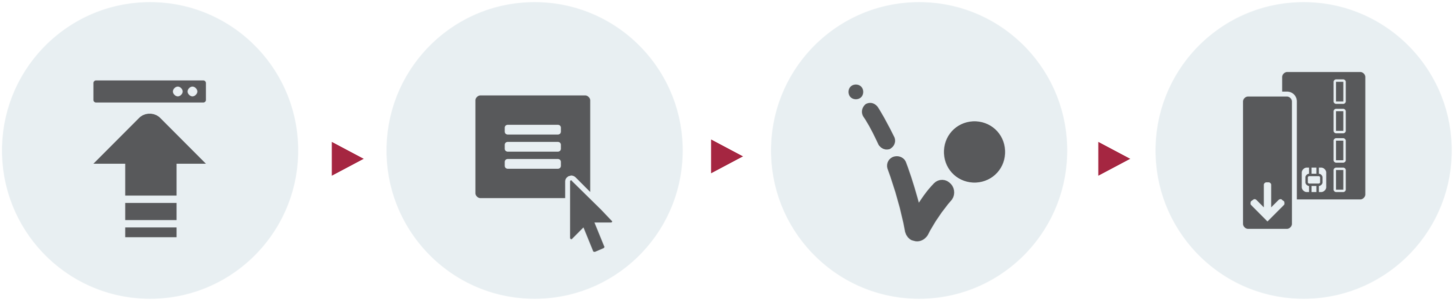
1 Like

What Is Serverless

- Event-driven architecture
- Invisible infrastructure
- Automatic scaling on demand
- Granular billing for execution time
- Fault tolerant and highly available



Serverless Deployment, the Duck Test



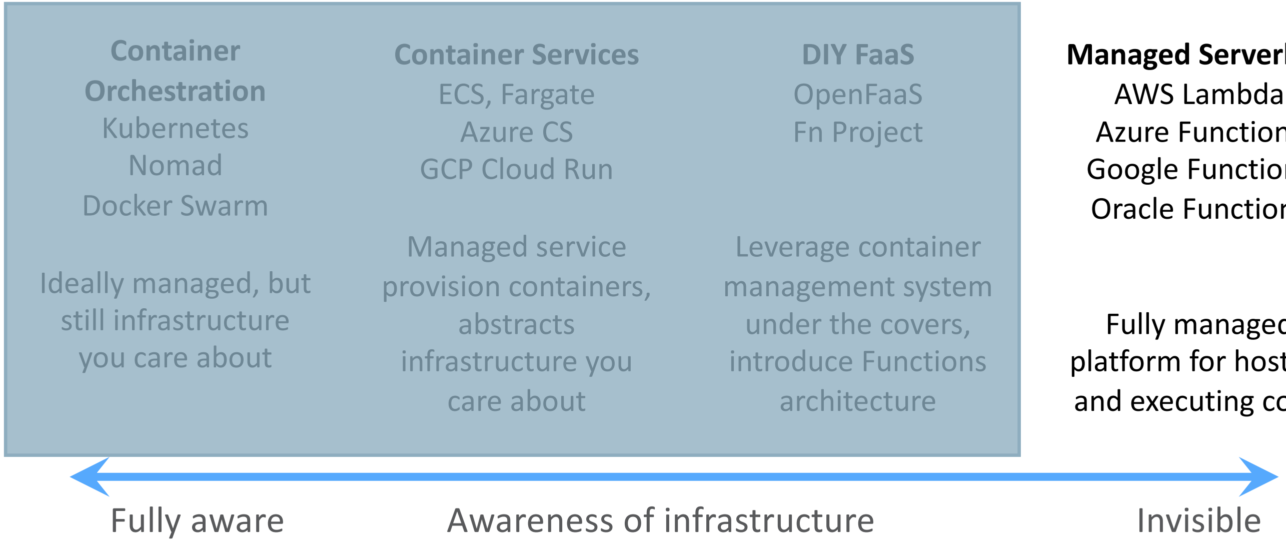
Upload Function Source Code

Configure Function Trigger

Function is invoked when triggered

Pay for execution time, not idle time

Serverless is Not Really a Spectrum



Container Orchestration
Kubernetes
Nomad
Docker Swarm

Ideally managed, but still infrastructure you care about

Container Services
ECS, Fargate
Azure CS
GCP Cloud Run

Managed service provision containers, abstracts infrastructure you care about

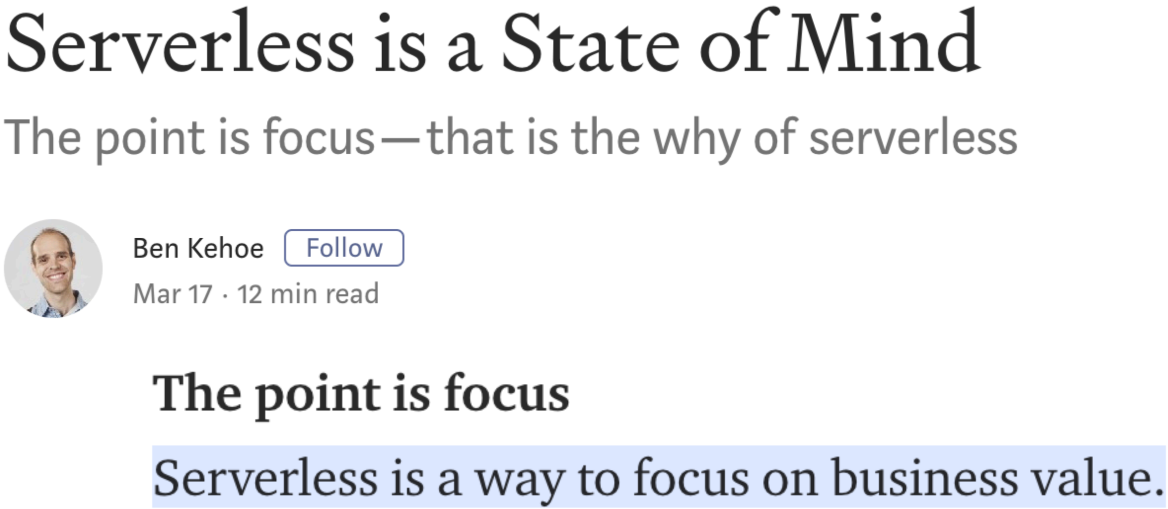
DIY FaaS
OpenFaaS
Fn Project

Leverage container management system under the covers, introduce Functions architecture

Managed Serverless
AWS Lambda
Azure Functions
Google Functions
Oracle Functions

Fully managed platform for hosting and executing code

What Is Serverless, Distilled



Serverless is a State of Mind

The point is focus—that is the why of serverless

Ben Kehoe

Mar 17 · 12 min read

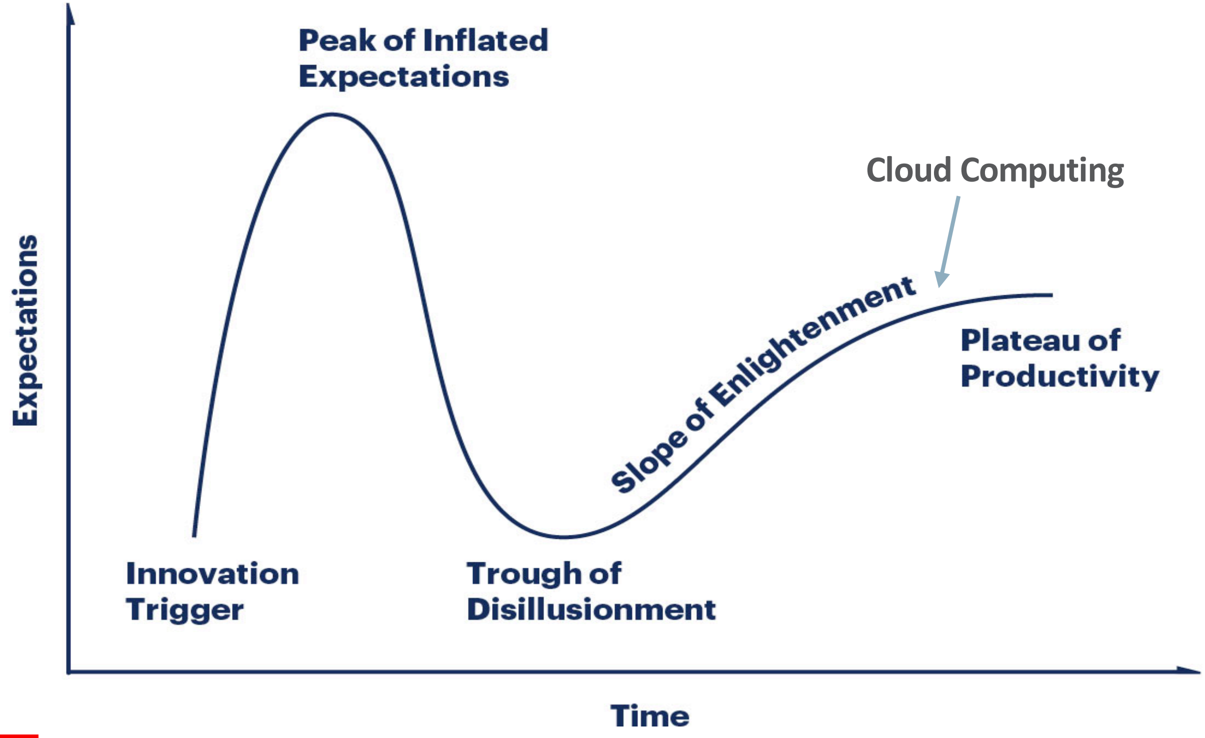
The point is focus

Serverless is a way to focus on business value.

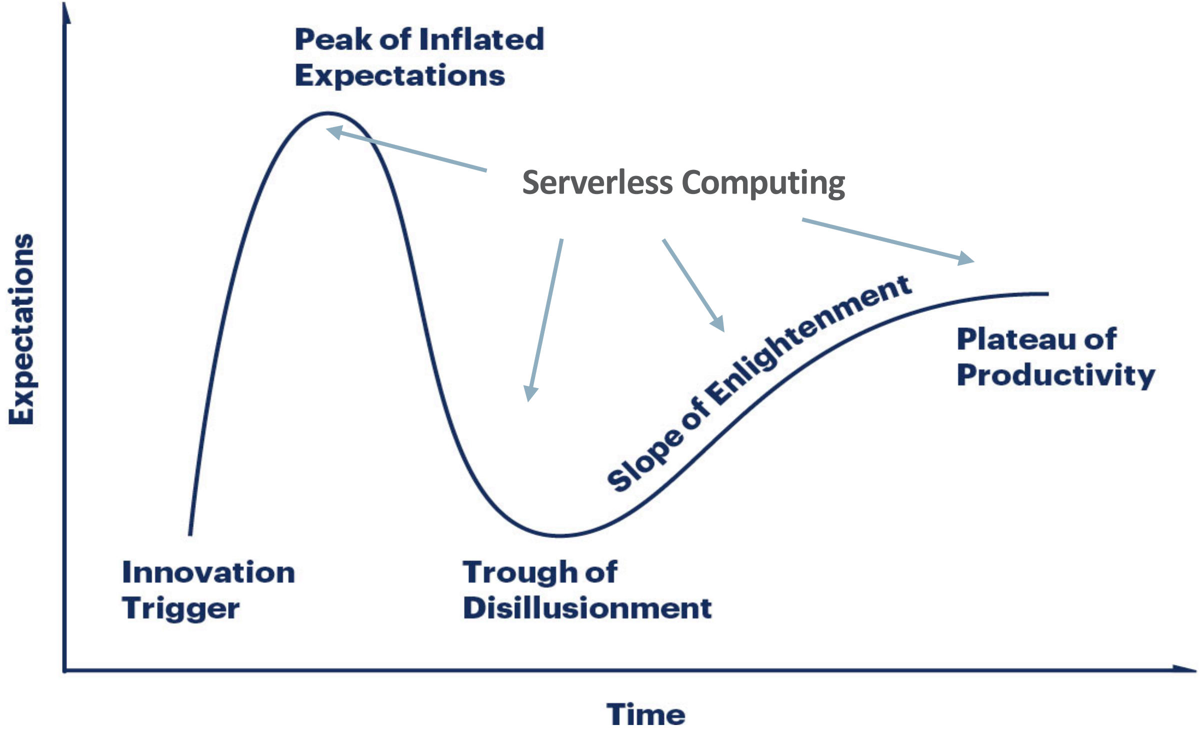
What Is Serverless Not

- It’s not magic, it’s a choice
- Brownfield: You need to break the monolith apart regardless
- Greenfield: You need a solid design
- Nothing is free

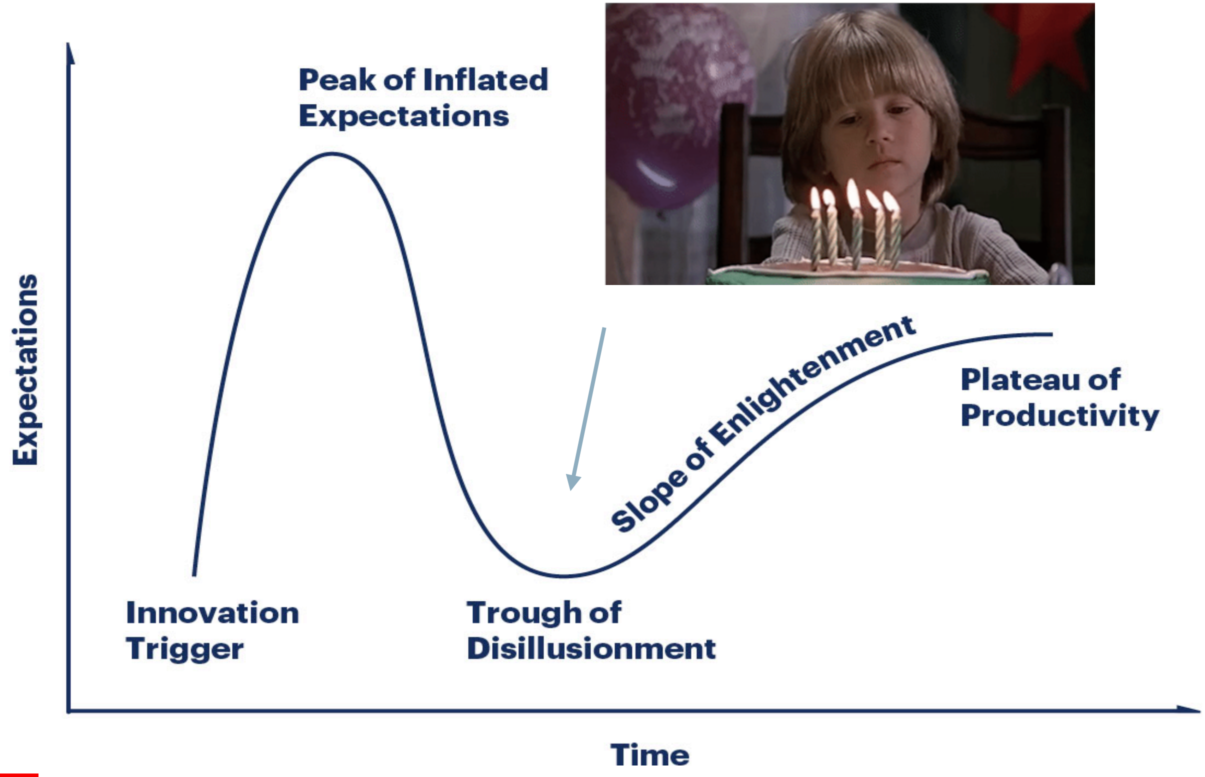
Hype Cycle – Productive Adoption



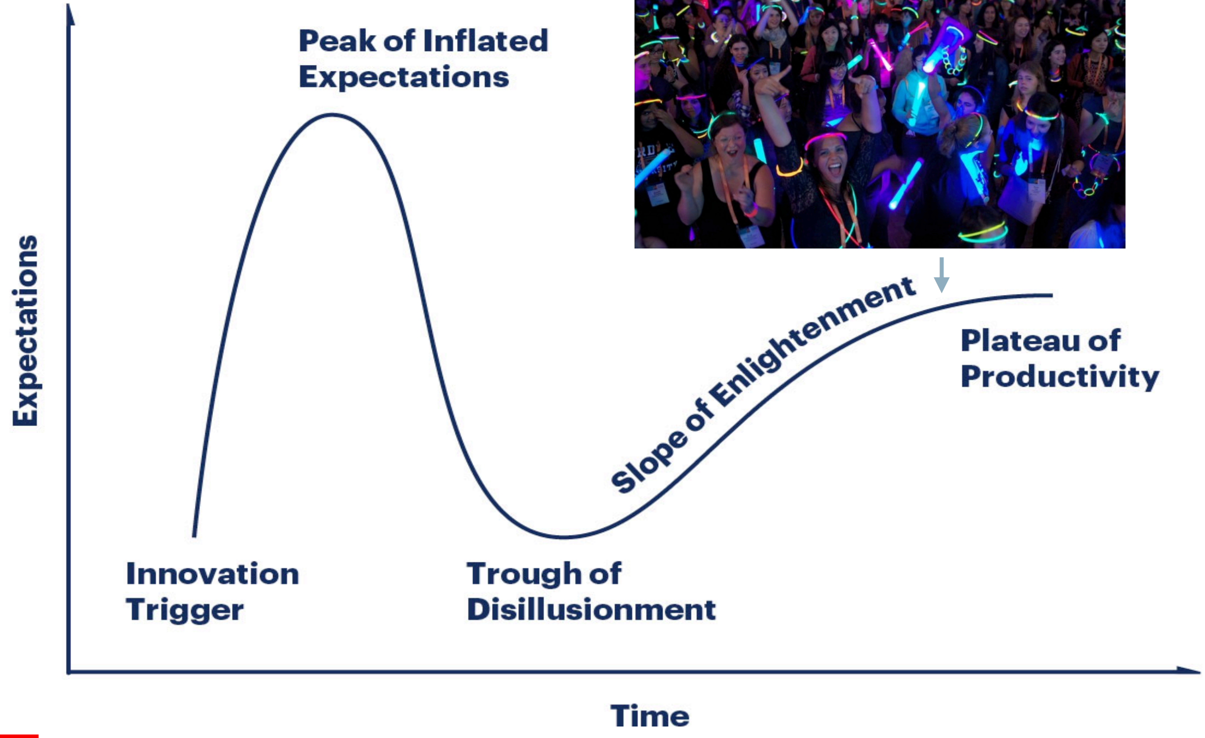
Hype Cycle – Serverless in Waves



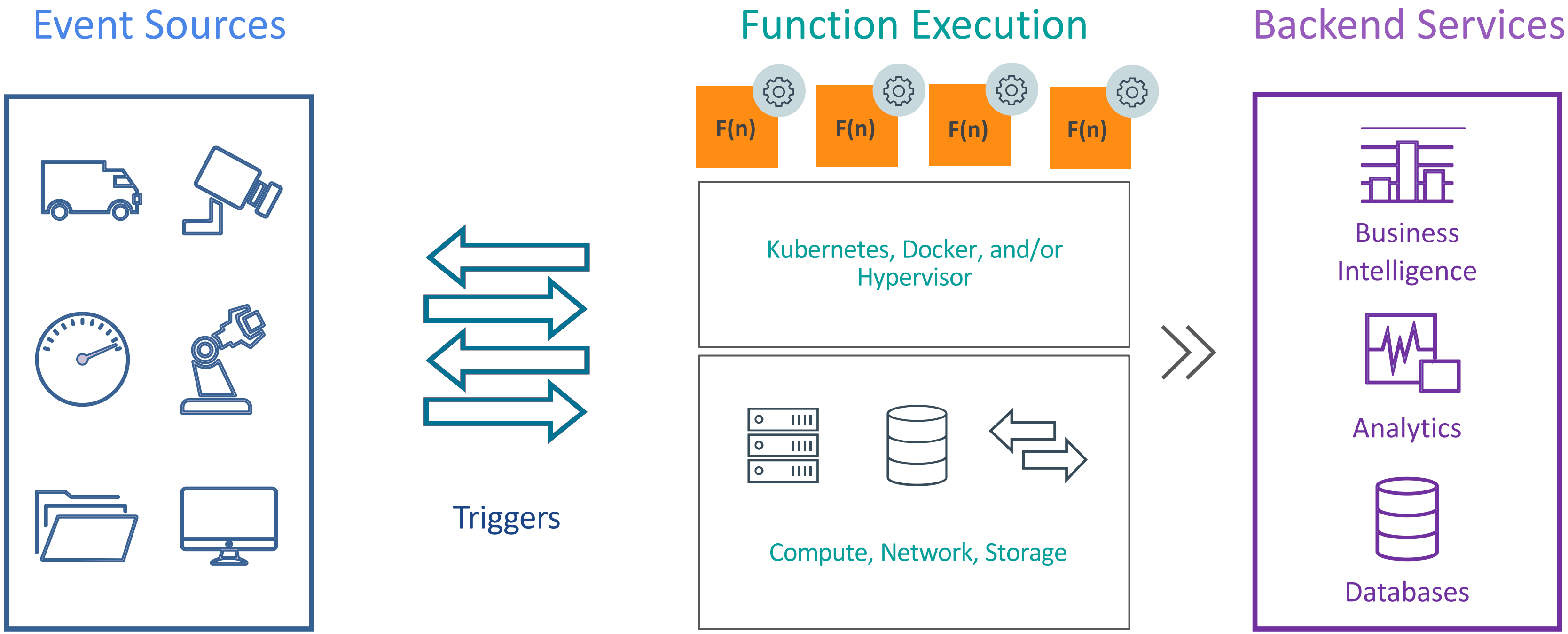
Get Through This Quickly…



…And Get Here

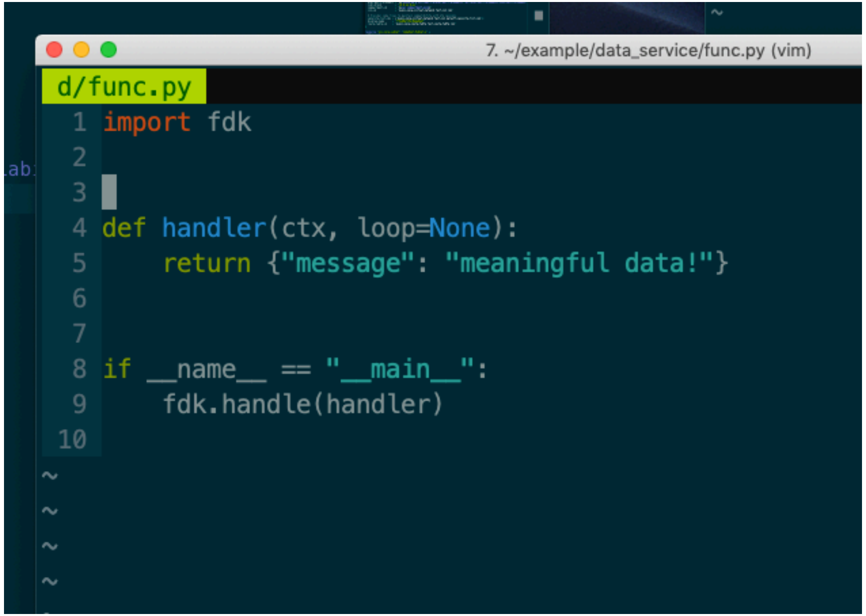

Peak of Inflated Expectations
Slope of Enlightenment
Plateau of Productivity
Expectations
Innovation Trigger
Trough of Disillusionment
Time

Serverless From 30k Feet



Function Example

- Different projects and products differ in use and workflow
- Just the code, configured against any number of event triggers
- Basically follow microservices rules of engagement
- As with microservices, applications are composed of many functions



```python
import fdk


def handler(ctx, loop=None):
    return {"message": "meaningful data!"}


if __name__ == "__main__":
    fdk.handle(handler)

```

Events and Execution Models

- Events are driven by context
- Execution model is your choice

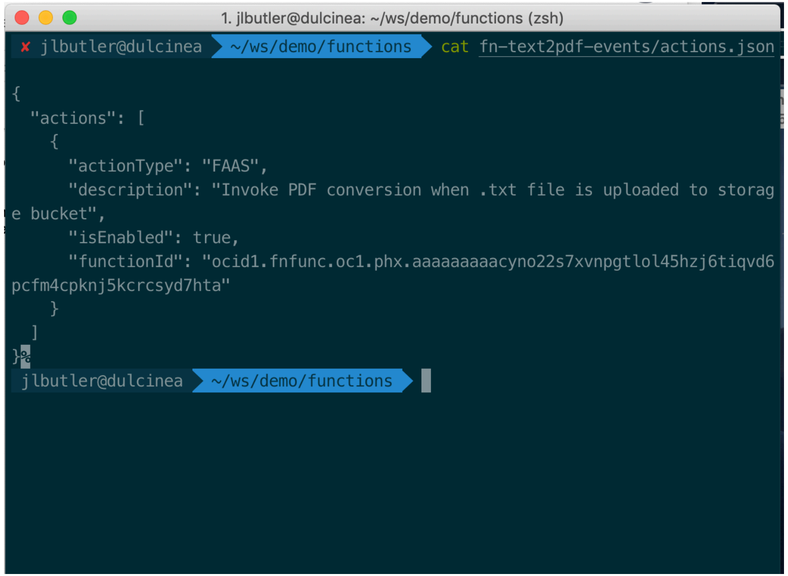
ORACLE

Events, Determined by Context

- Changes in data
- API Invocations
- Requests on endpoints
- Changes in resources
- Timers, Alarms, Direct Invocation…

Using Events

- Events are configured differently per platform
- Inform the platform as to what event(s) should invoke this function
- Function can consume the event and do the things
- CNCF Serverless WG has drafted a CloudEvents specification



```
✘ jlbutler@dulcinea  ~/ws/demo/functions  cat fn-text2pdf-events/actions.json

{
  "actions": [
    {
      "actionType": "FAAS",
      "description": "Invoke PDF conversion when .txt file is uploaded to storag
e bucket",
      "isEnabled": true,
      "functionId": "ocid1.fnfunc.oc1.phx.aaaaaaaacyno22s7xvnpgtlol45hzj6tiqvd6
pcfm4cpknj5kcrcsyd7hta"
    }
  ]
}%
 jlbutler@dulcinea  ~/ws/demo/functions 
```

Execution Models

- Synchronous
- Asynchronous
- Streaming

ORACLE

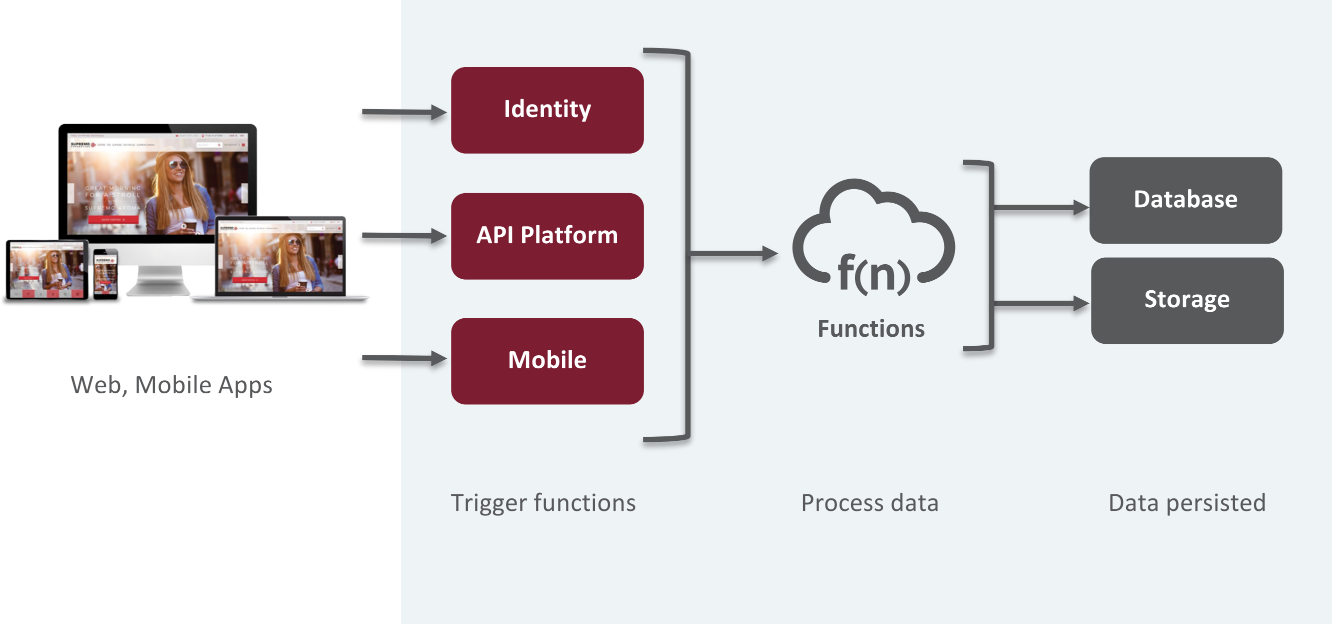
Error Handling by Model

- Synchronous
  - Calling code handles the error
- Asynchronous
  - System retries based upon timeline
- Streaming
  - System retries based upon data efficacy

What Can We Build With Serverless?

- Web and Mobile Backends
- Any other backend API implementations
- Real-time Processing of Files, Streams
- Batch Processing
- Glueing up SaaS things
- Kind of anything

Web and Mobile Backends



Web, Mobile Apps

Identity

API Platform

Mobile

f(n)

Functions

Database

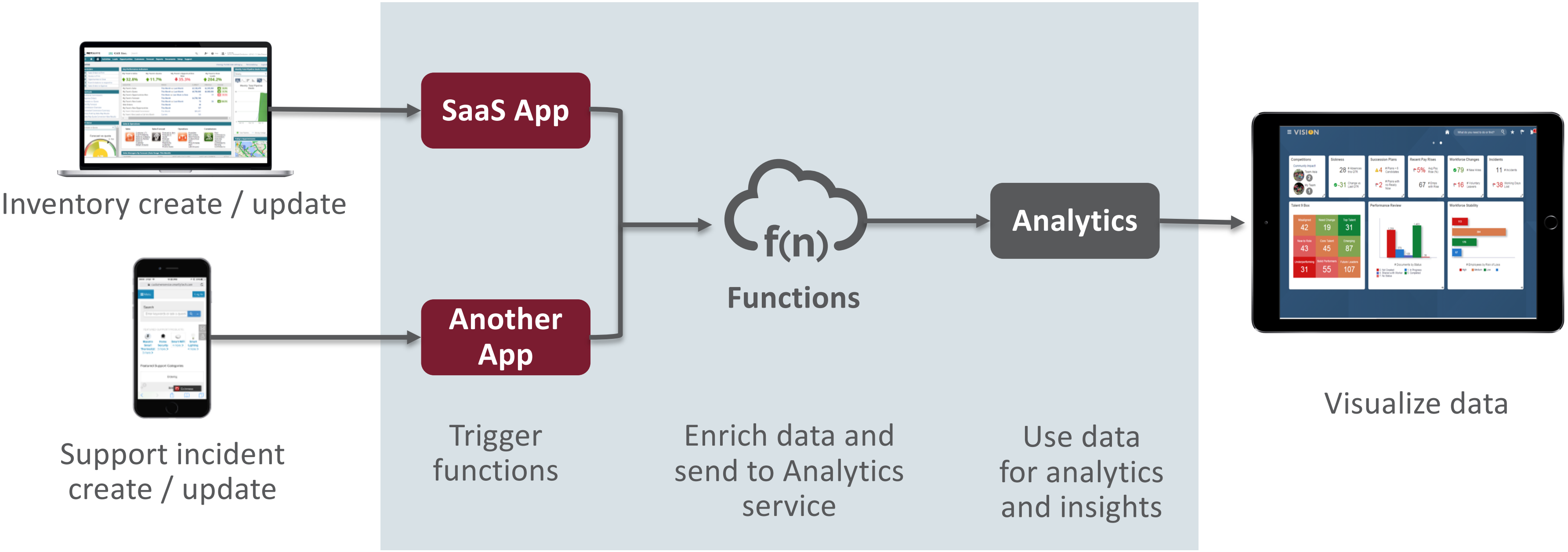
Storage

Trigger functions

Process data

Data persisted

Extend and Enhance Existing Applications



Inventory create / update

Support incident create / update

SaaS App

Another App

Functions

Analytics

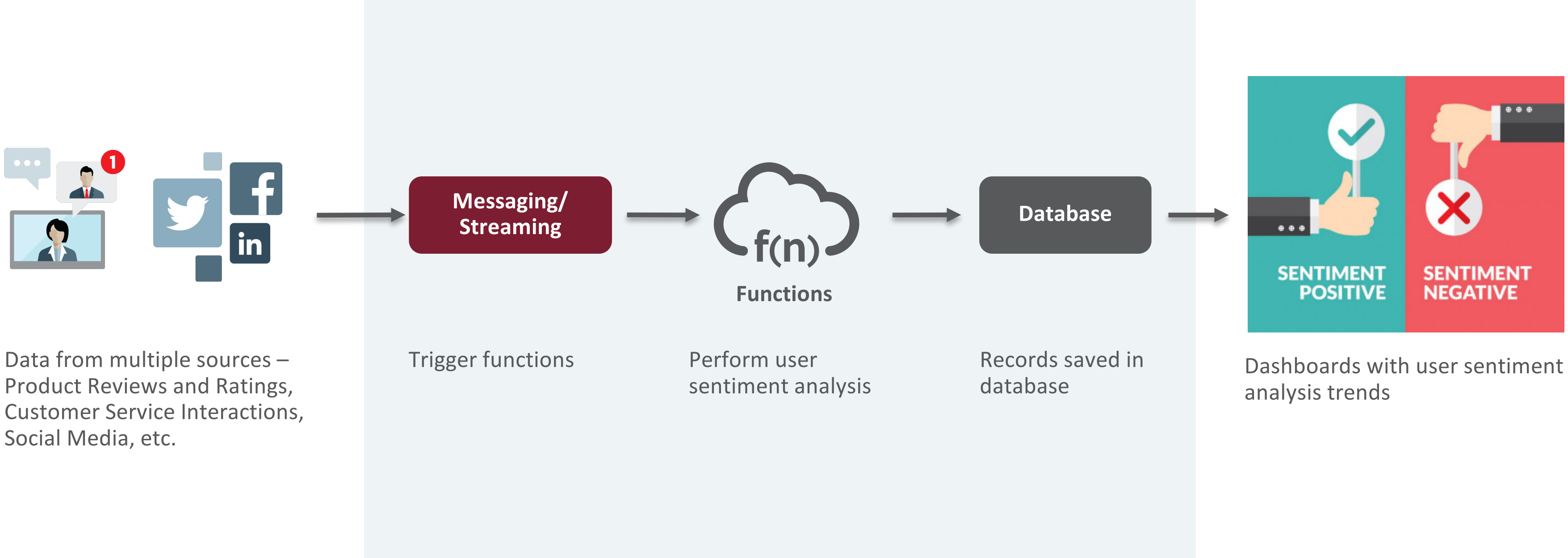
Trigger functions

Enrich data and send to Analytics service

Use data for analytics and insights

Visualize data

Real-Time Stream Processing



Messaging/ Streaming → Functions → Database

Data from multiple sources – Product Reviews and Ratings, Customer Service Interactions, Social Media, etc.

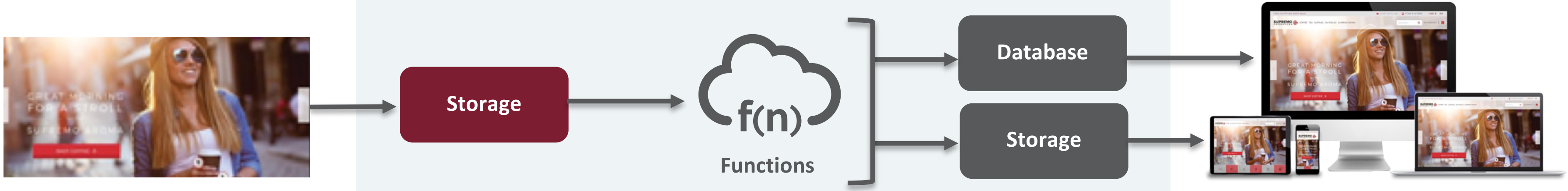
Trigger functions

Perform user sentiment analysis

Records saved in database

Dashboards with user sentiment analysis trends

Real-Time File Processing



High resolution product image uploaded to storage

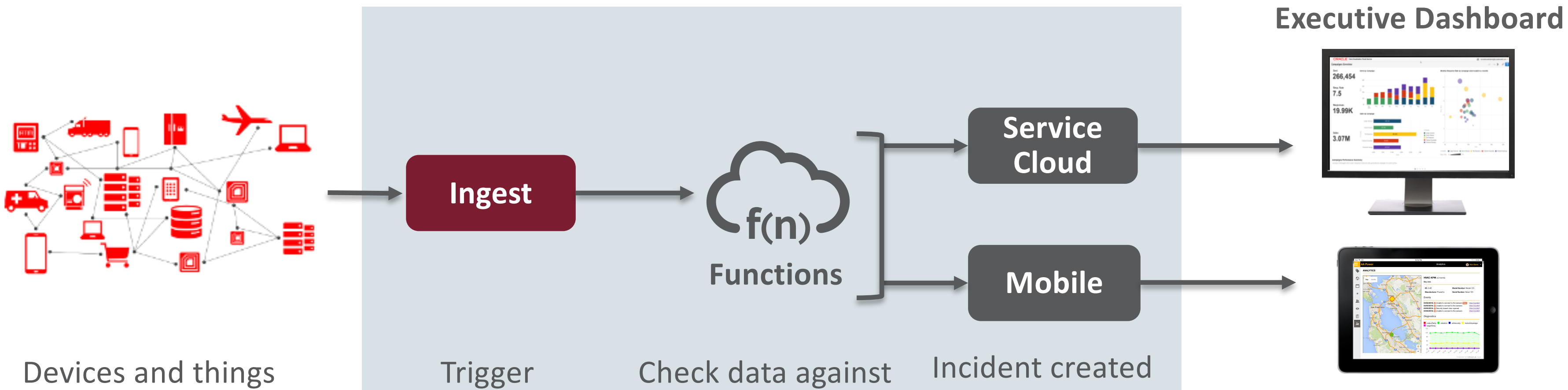
Trigger functions

Generate images of different resolutions and sizes

Images saved in Storage, metadata in Database

Generated images displayed on various pages and devices

Internet of Things



Executive Dashboard

Ingest

Functions

Service Cloud

Mobile

Technician App

Devices and things streaming sensor data

Trigger functions

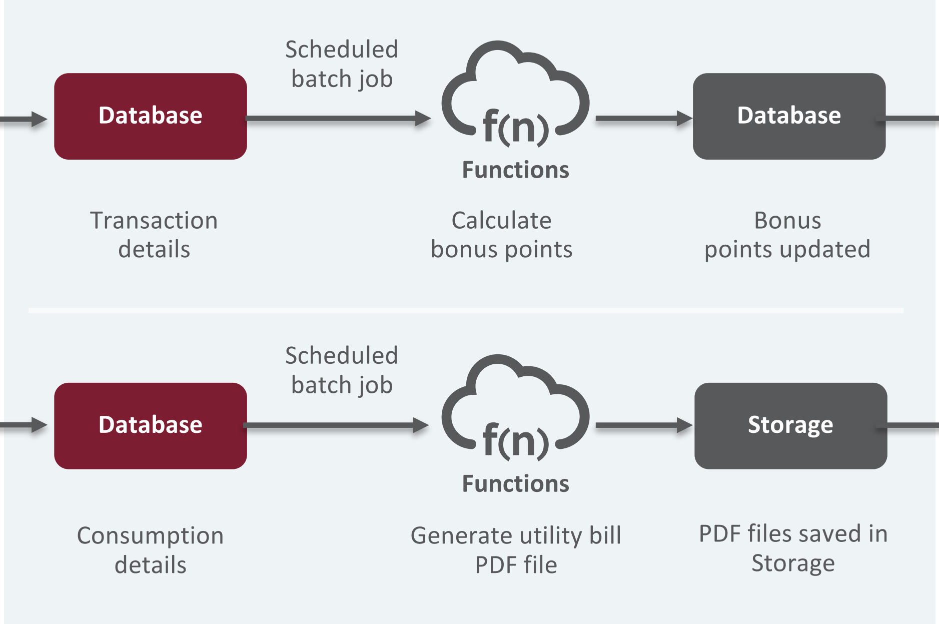
Check data against thresholds. If exceeded, raise support incidents, send notifications

Incident created in Service Cloud, notification sent to the technician

Batch Processing

Credit card transactions

Utility consumption



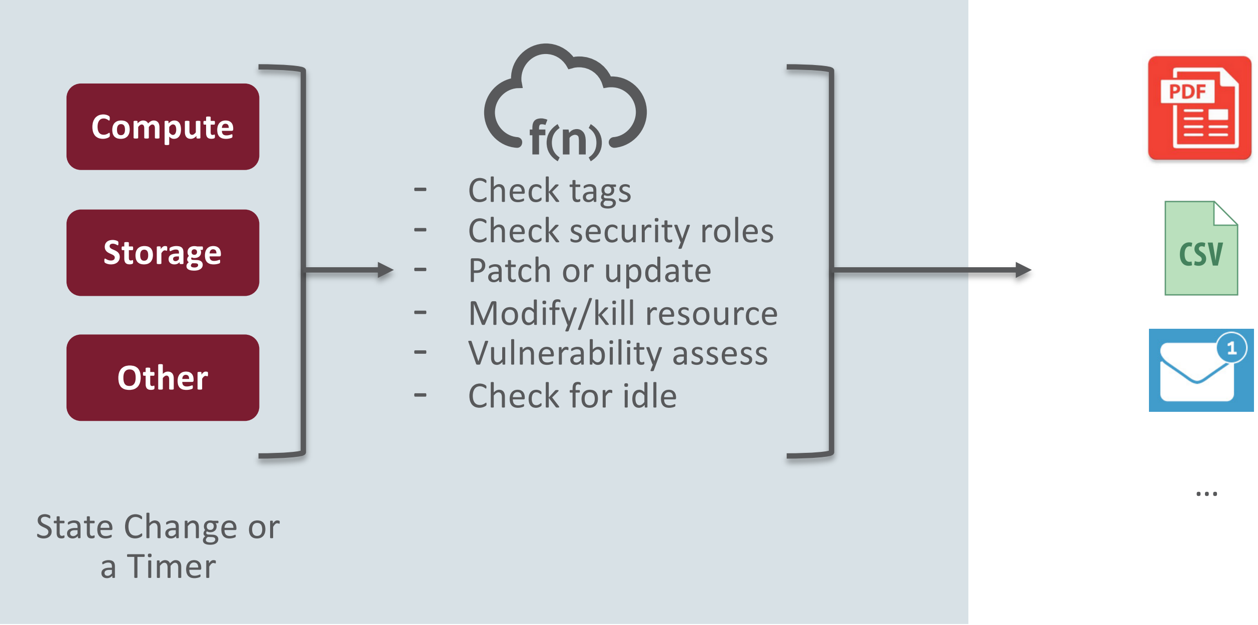
Congratulations!

You've earned 40,000 bonus points.

Loyalty bonus received

Utility bill PDF file

DevOps Automation



Compute

Storage

Other

State Change or
a Timer

f(n)

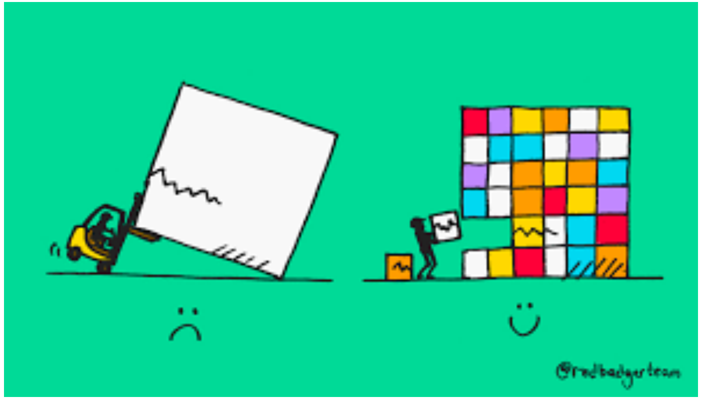
- Check tags
- Check security roles
- Patch or update
- Modify/kill resource
- Vulnerability assess
- Check for idle

Big Ideas Around the Code

- Don’t own what you don’t have to
- Serverless is all about APIs
- When you do implement, simplify
- Plan ahead for observability

Choose a Pattern that Helps you Minimize

- Less is more: shoot for a single handler per module
- If one function does more than one discrete thing, break it up
- Better to proliferate than to decompensate
- Observability and triage become infinitely easier at the boundaries



Separate and Simplify

- Simplify application estate as well
- Events can and probably should define application boundaries
- Share libraries between functions and applications, not execution context

ORACLE

Code Reuse

- Not long ago, Serverless function deployments were zip files
- Now, many Serverless platforms expose a container image to you
- Others use layering which mimics the container image stacking

ORACLE

Observability

- Metrics
  - Aggregate data regarding the behavior of a thing over time
- Tracing
  - Instrumentation which provides an instance of an action, traversing the entire stack
- Logging
  - Developer breadcrumbs we leave to give context for a certain code path

ORACLE

Triaging Issues

- Monolith in a VM – log into the host, look at logs, run a debugger
- Containers in Kubernetes – Istio, Jaeger, Prometheus, Grafana, et al
- Serverless… is complicated. Logging is there, but that's not very useful at scale
- OpenTracing & Jaeger is a possibility

Observability Solutions From Your Provider

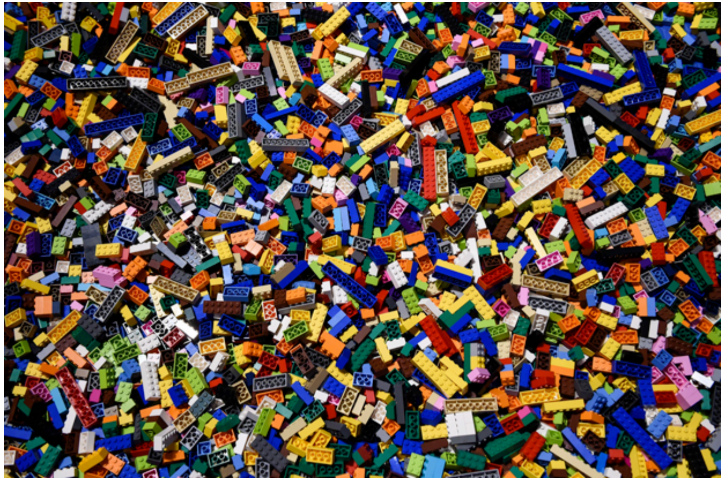
Cloudwatch, X-ray, Stackdriver, OCI Monitoring and Logging, Azure Insights, IBM Monitoring, and others

Observability Solutions From Other Experts

Honeycomb, IO|Pipe, DataDog, Dashbird, Thundra, Epsagon, Splunk, Lightstep, Solo, and others (sorry if I missed yours!)

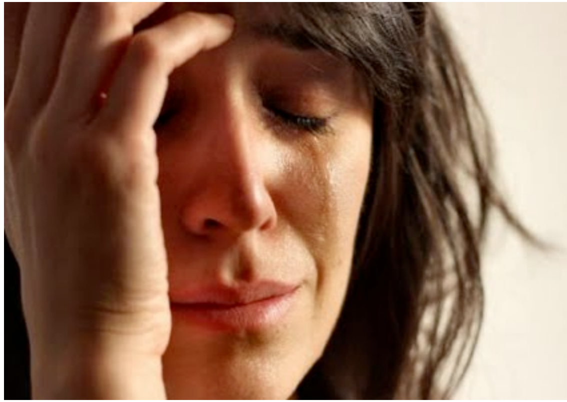
Use Services and SaaS When Possible

- DBaaS
- Identity, Auth, Forms
- Storage Services
- Email, SMS
- Maps, GPS
- Media Streaming
- Chat and Chatbots



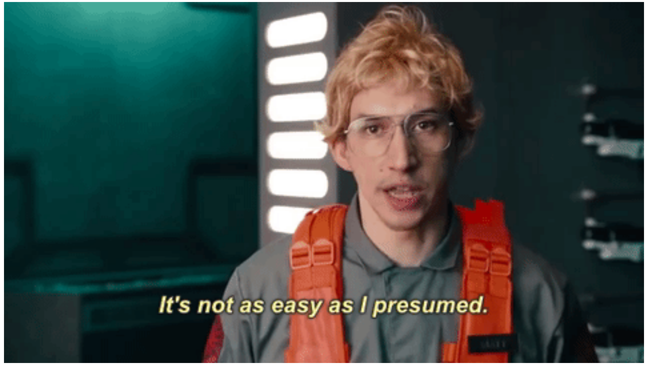
Worries

- Don’t worry too much about cold starts
- Do worry about data egress and migration
- Pay attention to the system you are integrating with, keep it open if possible



Is Serverless Simpler?

- In a word, no
- But that doesn’t mean it’s not better
- Serverless doesn’t really mean less complexity
- Resolving complexity is generally directly related to your core business



ORACLE

@jlb13

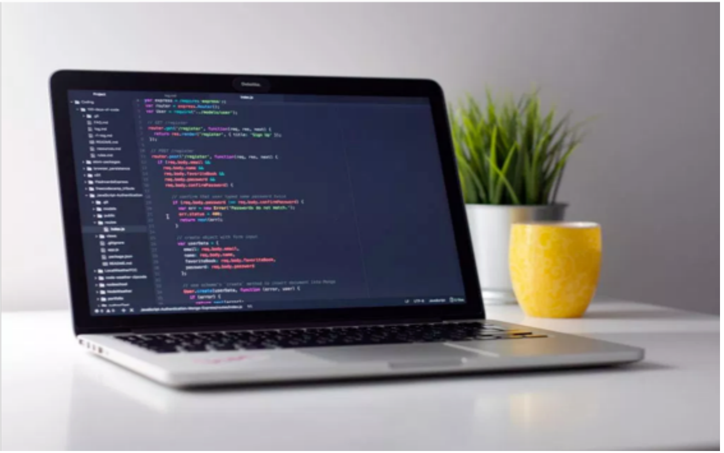
Is Serverless Better?

- In a word, yes
- Less toil in deployment and maintenance of systems is beneficial to focus
- OpEx reductions can be profound
- Tradeoff: we depend upon third parties to address issues as they arise

ORACLE

Final Thoughts

- Put Serverless on your radar
  - Greenfield
  - Brownfield migration
- Often a POC becomes production
- Resist the urge to compare DevOps and Serverless. Apples to Apple Pie.
- Build stuff!



ORACLE

@jlb13

ORACLE

Cloud Native Labs

Thanks!

@jlb13

cloud.oracle.com/trial

cloudnative.oracle.com