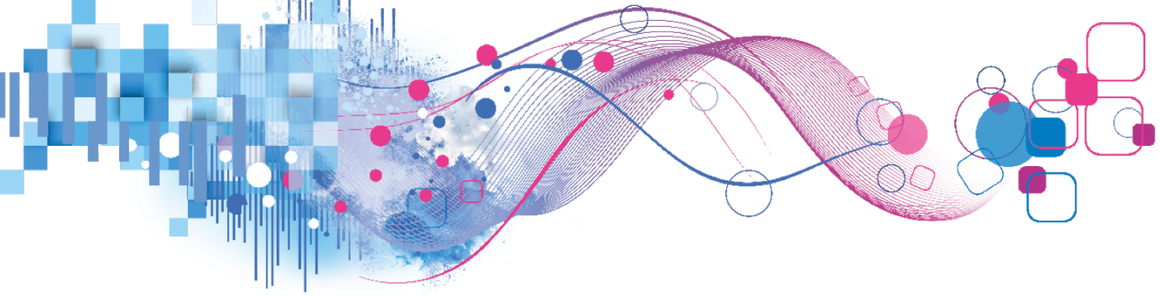


PaaS Vision & Demonstration

Author: J. Deeming

Version: 4.2

Date: December 2017

Team Objectives & Structure

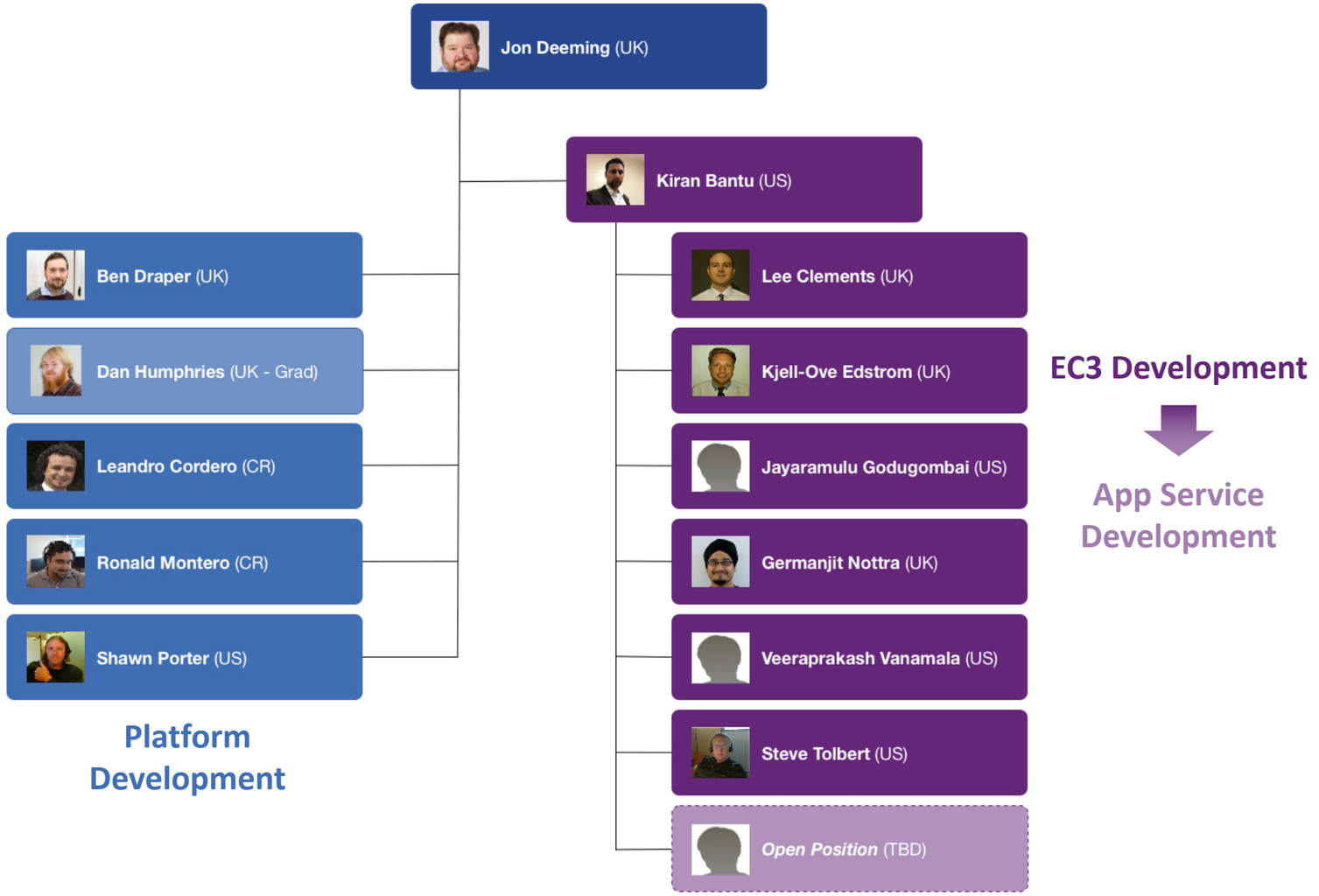
Who are we and what are we here to do?

Create the reference architecture and implementation for EITS's Platform-as-a-Service offering.

The PaaS will provide a standard target platform for our application developers to build upon.

The platform will run elastically using on-site, co-lo and/or cloud resources as needed.

This will enable new agile and flexible deployment models and open up options to fund infrastructure using capex, opex or a combination, dependant on the business need.



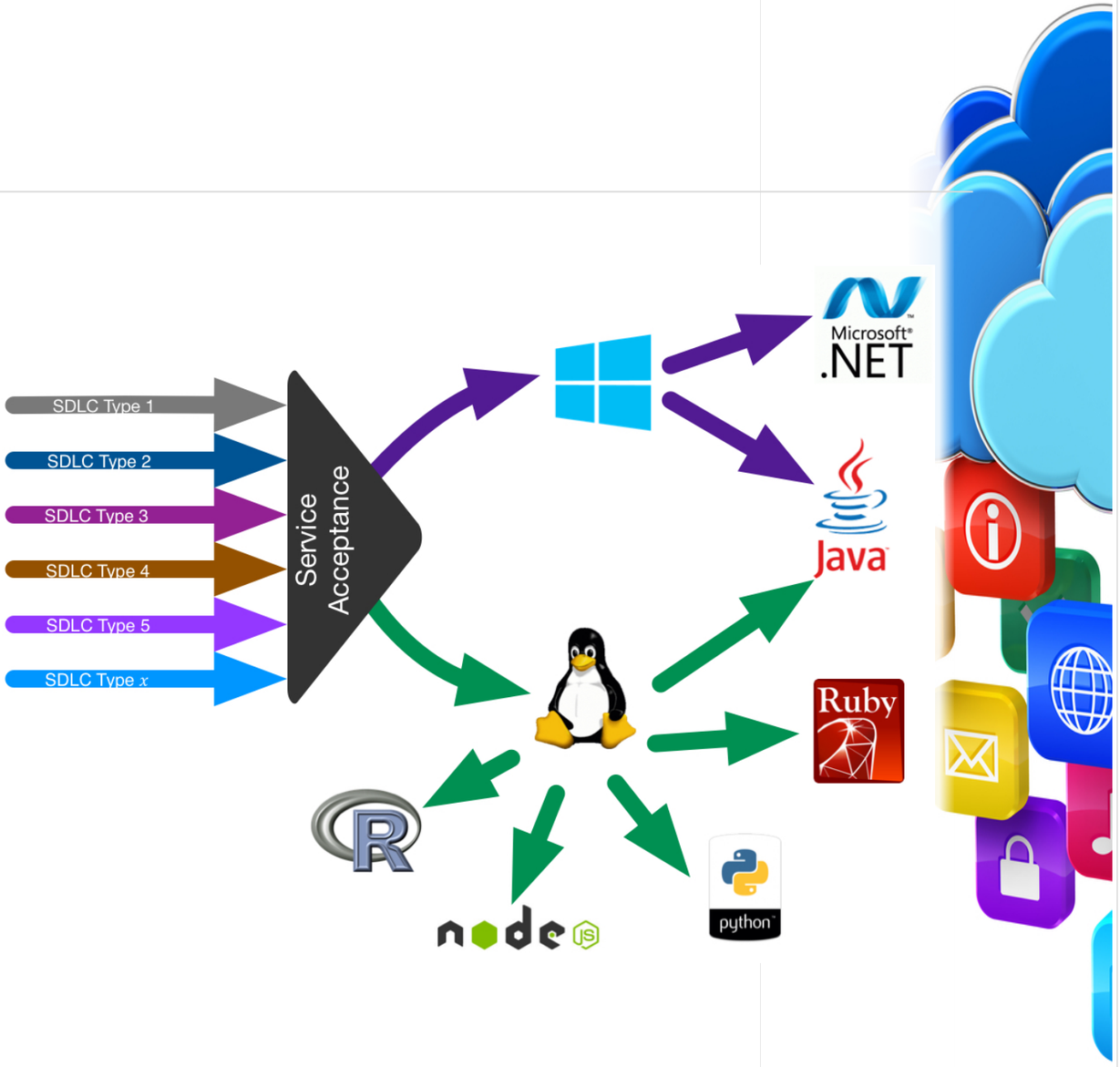
Current State

Multiple development processes and deployment platforms

- 40+ development teams globally with locally optimized processes
- Disparate tools pipelines and deployment platforms
- Multiple programming languages in use
- Significant manual effort and risk of error
- Lack of standards hinder automation
- EITS Service Acceptance is necessary but often perceived as a blocker to innovation and speed of delivery
- Difficult to develop standard ITSM changes with so much variance

Local team optimizations slow down the overall process execution, increasing delays, cost and risk.

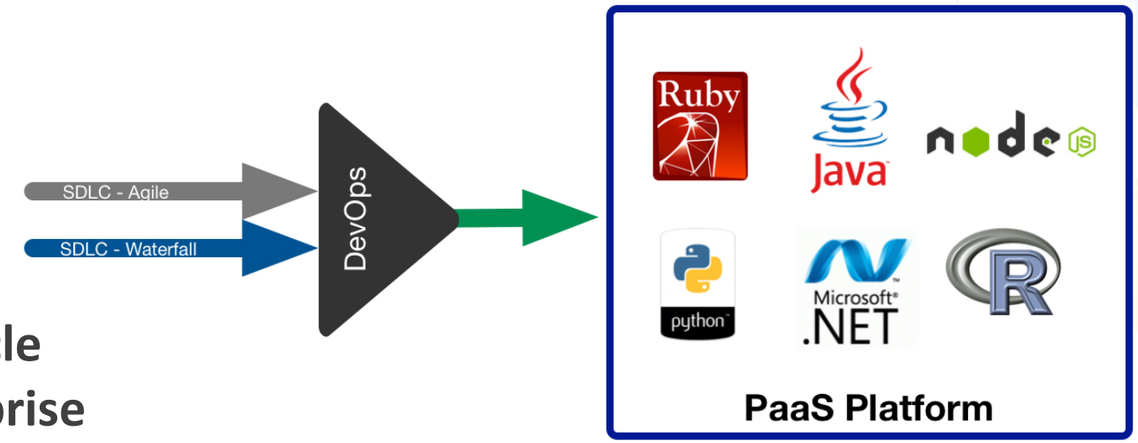
We need to standardize and automate as much as possible, whilst continuing to allow the developers choice in their language usage to tackle problems in the optimal fashion.



Future State

Standard Platform-As-A-Service (PaaS) for developers to deploy on.

- Unified development processes – potentially one agile and one waterfall (for financial client demands)
- Standard tools pipeline and PaaS platform
- Continue to support multiple programming languages
- Evolve Service Acceptance into a collaborative and contemporary DevOps function to work with Developers to transition code into production
- Focus on standardized ITSM changes to allow automation with minimal human intervention

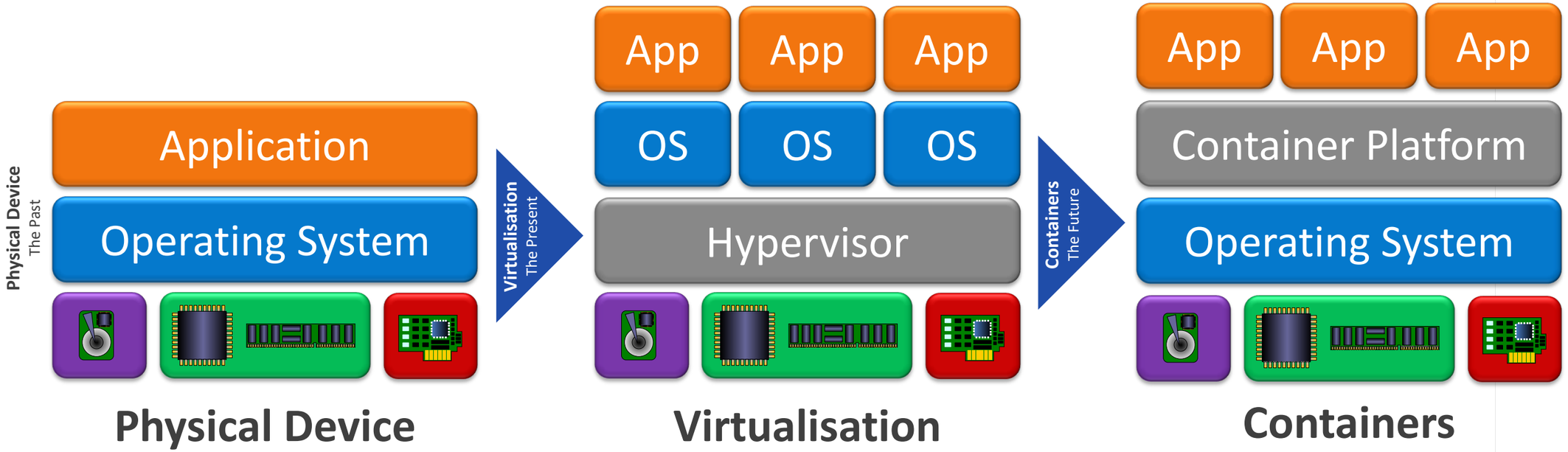


Moving to a unified global development lifecycle allows optimization of the process at an Enterprise level – reducing risk/delays and increasing throughput.

We continue to allow developers choice in their language usage, whilst standardizing the manner in which they deliver code into production.

Why Containers?

Differences from Virtual Machines

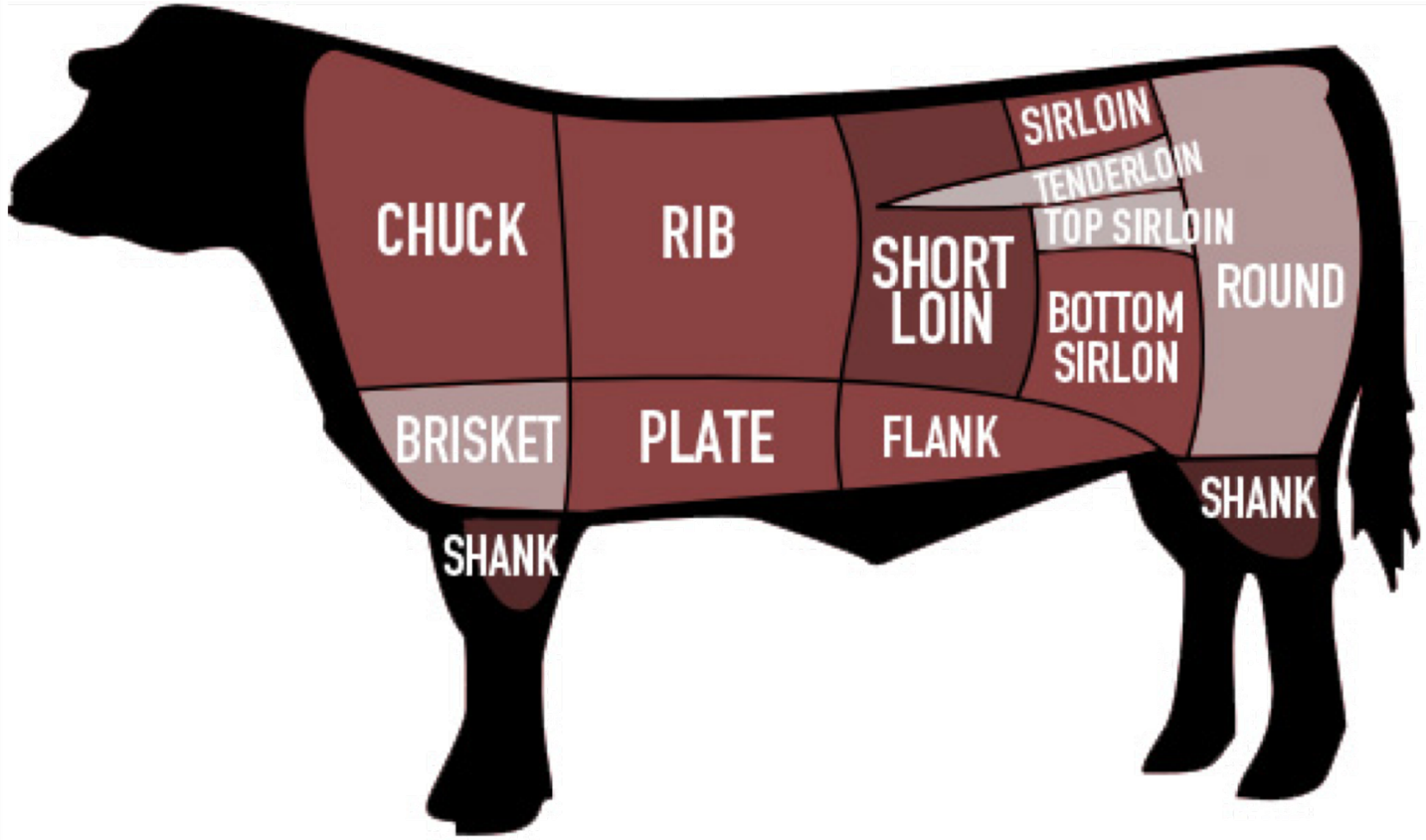


- Easier deployment and roll-back as App is loosely coupled to the Operating System
- Allows for a hybrid private:public architecture more easily due to standard platform on both infrastructures
- Greater consolidation - from c.40:1 in VMs to 100:1 in containers. Some implementations seeing 1000+:1*
- Less patching/licensing through elimination of hypervisor and OS instances.
- Opportunities to leverage 'white box' commodity servers and reduce hardware spend

* = Dependent on memory usage of applications and decomposition strategy

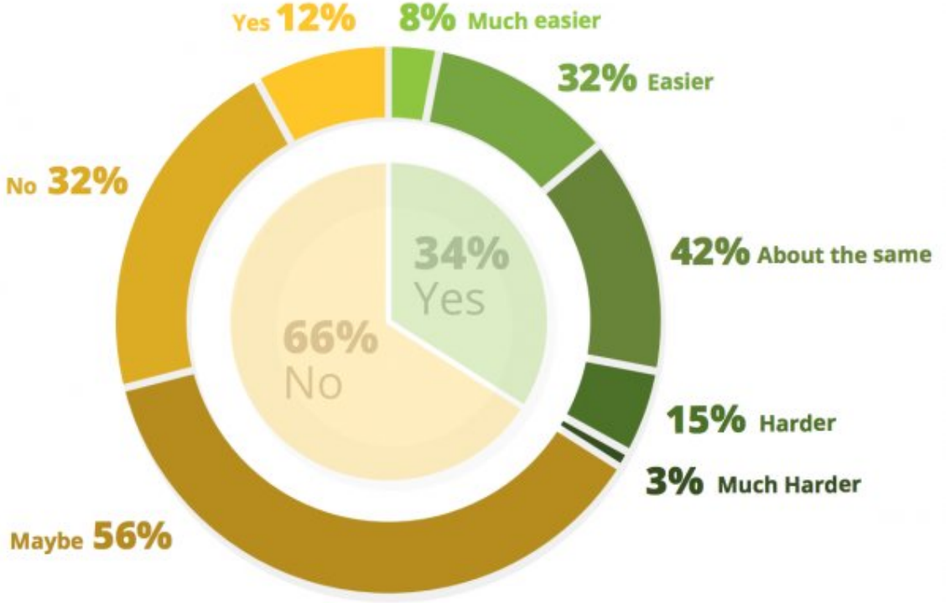
MicroServices vs Monoliths

Supporting multiple architectural styles



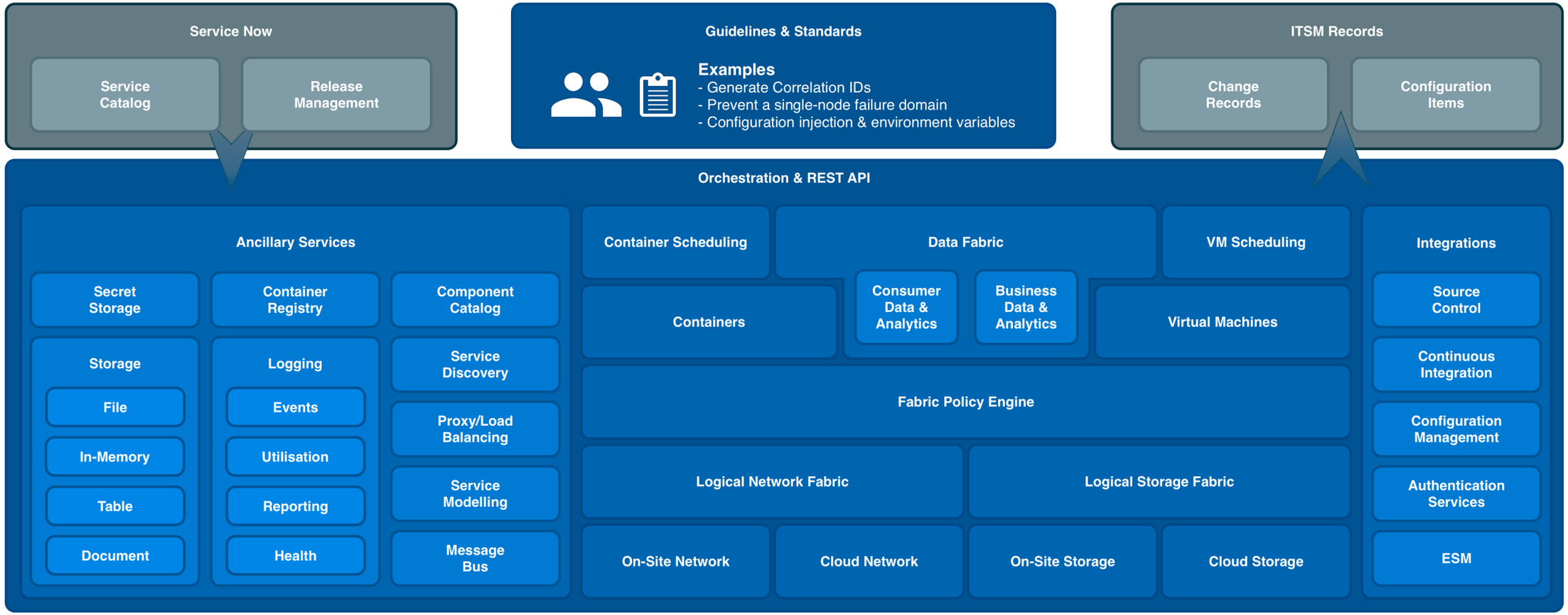
[Non-Microservice Users]
Are you planning to move to a microservices architecture?

[Microservice Users]
Has moving to a microservice architecture made your job easier or harder?



High-Level PaaS Reference Architecture

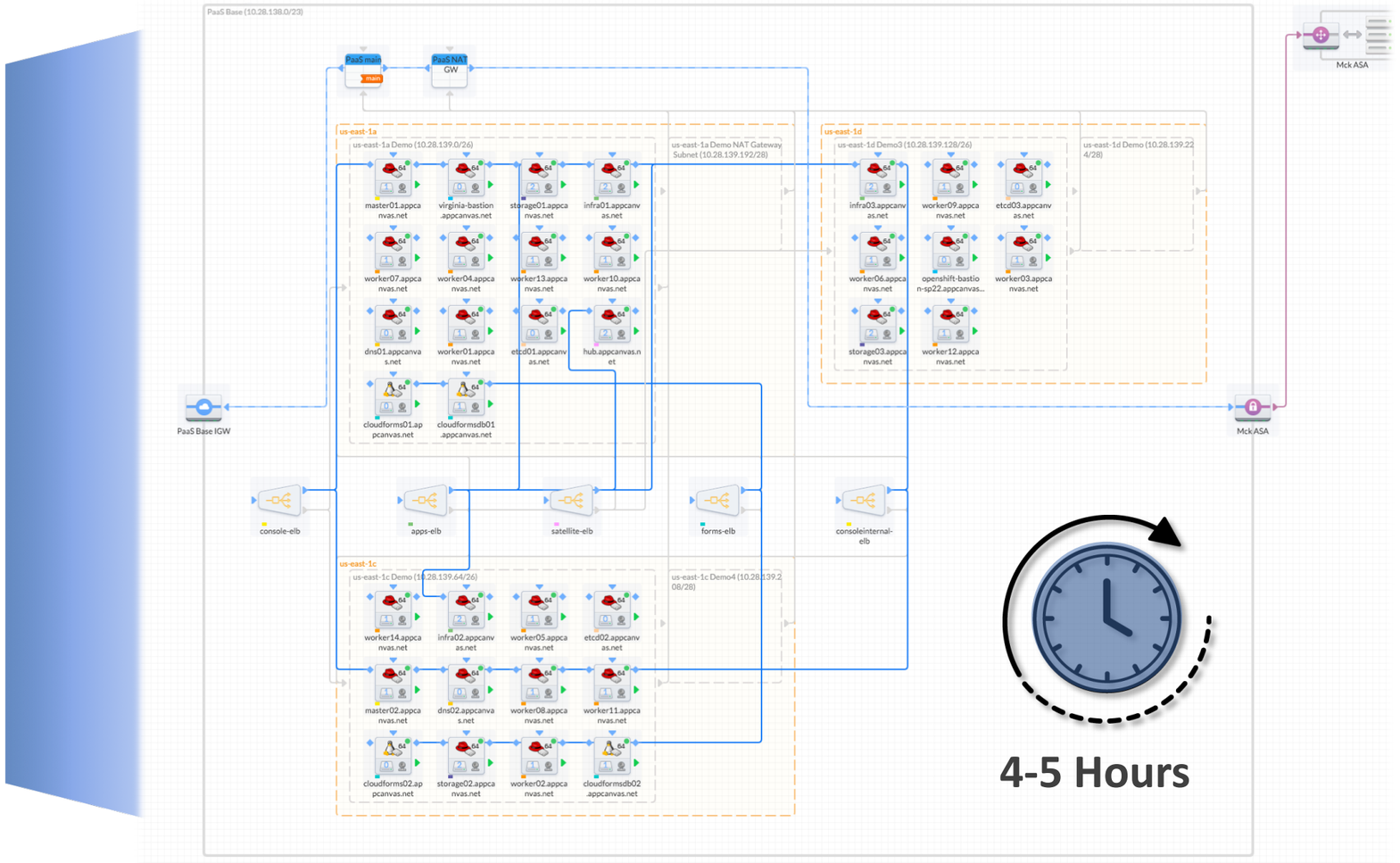
Current architectural vision



Where We Are

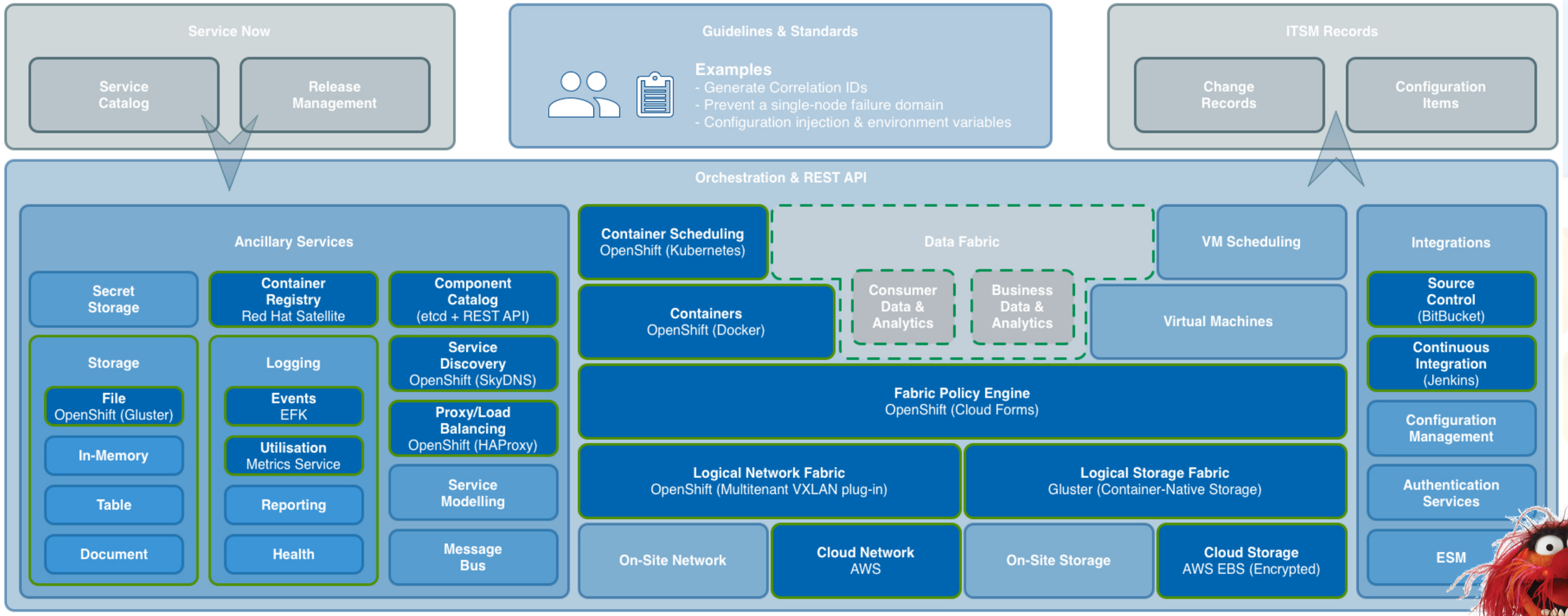
A fully functional, highly-resilient AWS environment for internal use...

- Cluster Configuration
- OpenShift Installation
- Linux Installation
- AWS Bastion Host
- AWS Setup



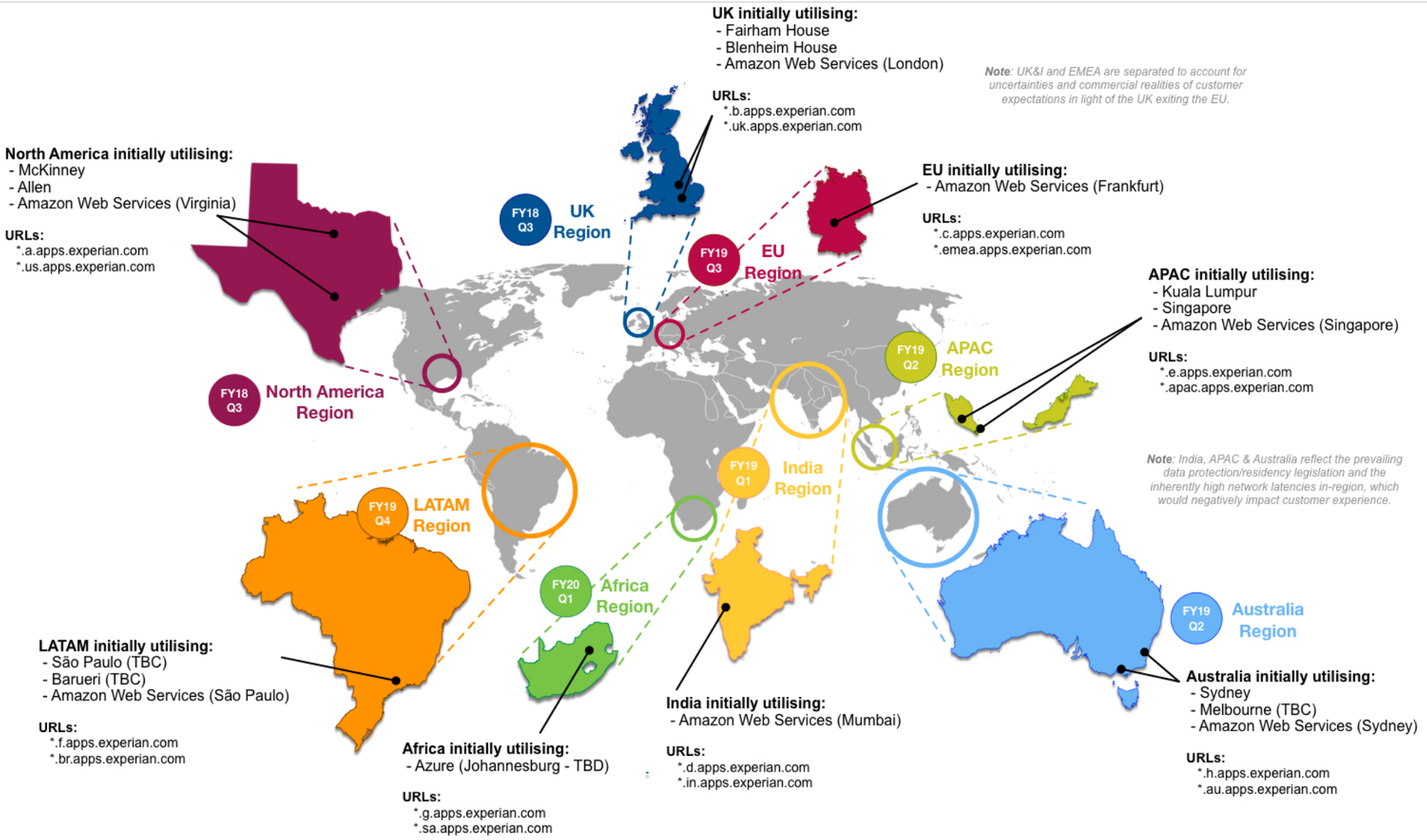
High-Level PaaS – Phase 1 'Animal'

Initial deliverables



High-Level Availability Zones Concept & Public Cloud Alignment

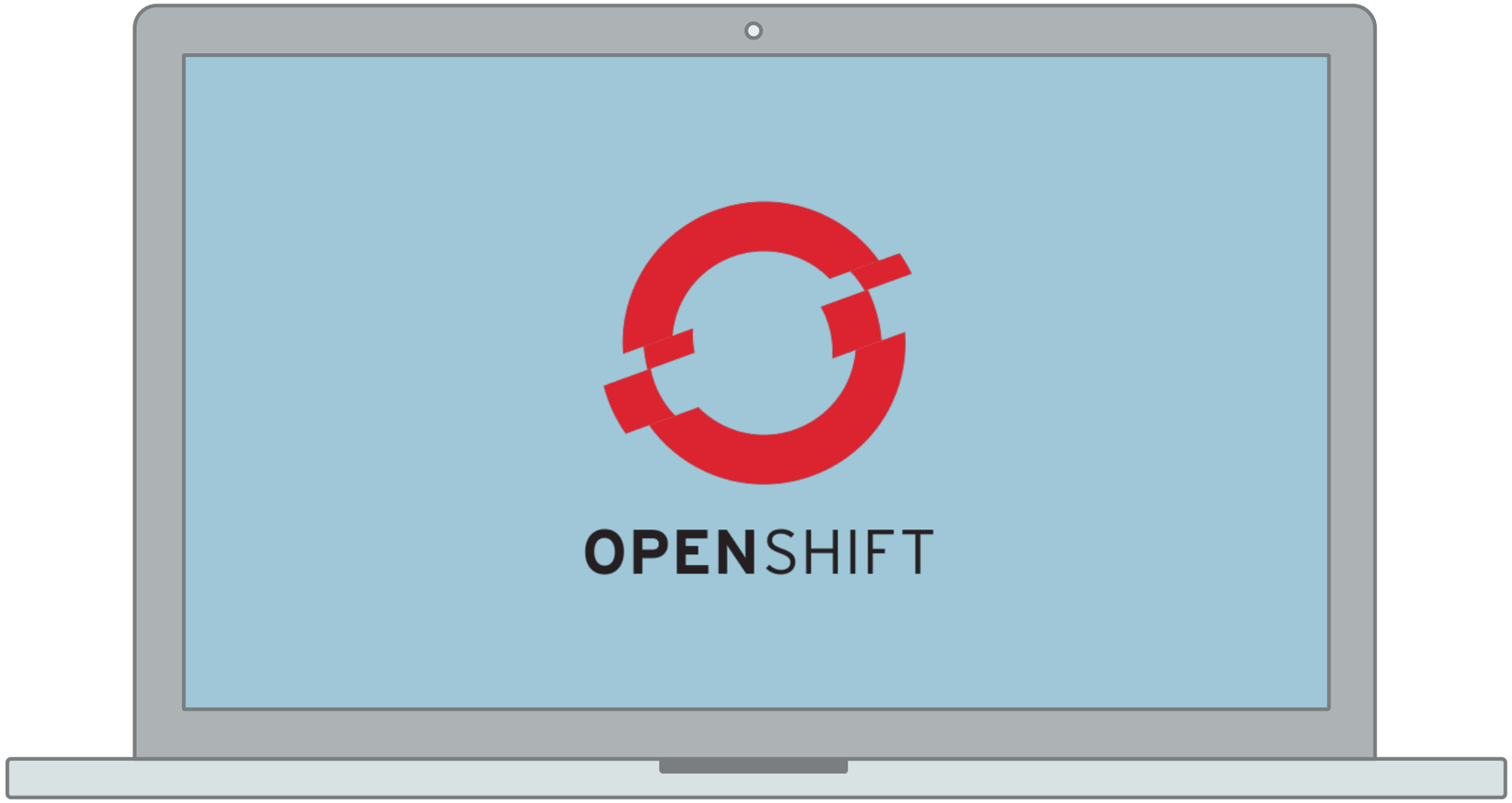
Dual zones per region – PaaS spans zones, but is resilient within region



Note: Region/Zone terminology similar to AWS, so architecturally familiar to developers

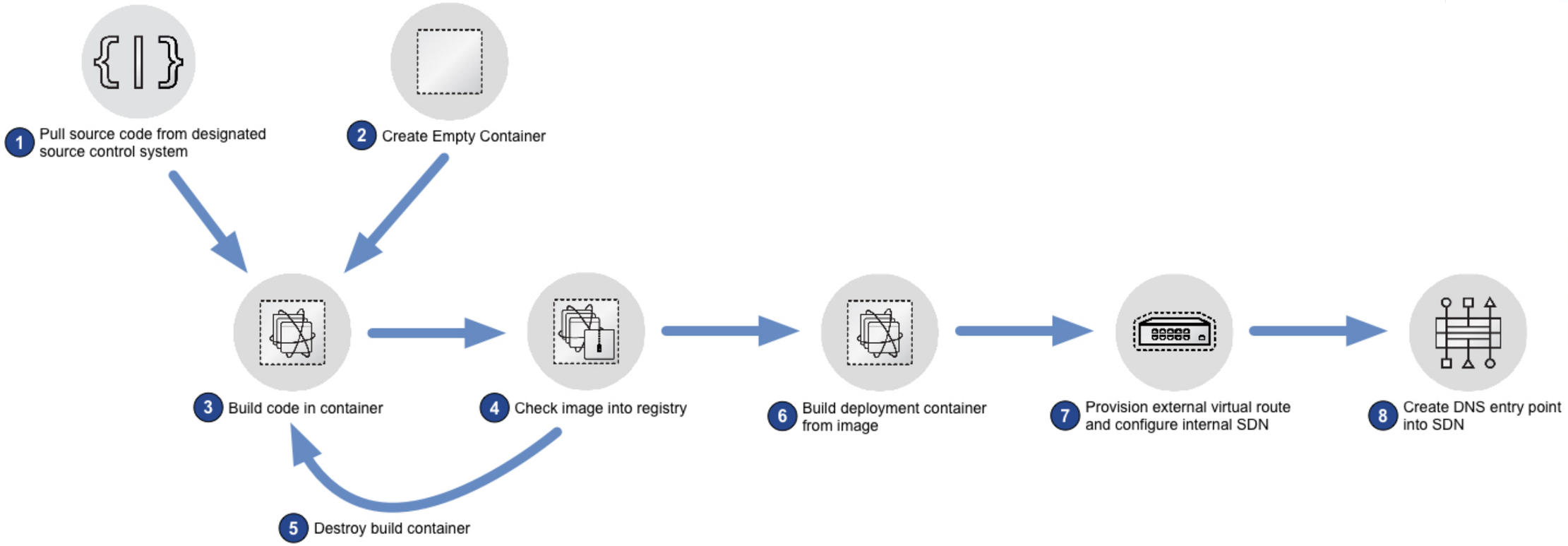
Developer Experience – Application Deployment

A simple Python chat-room deployment



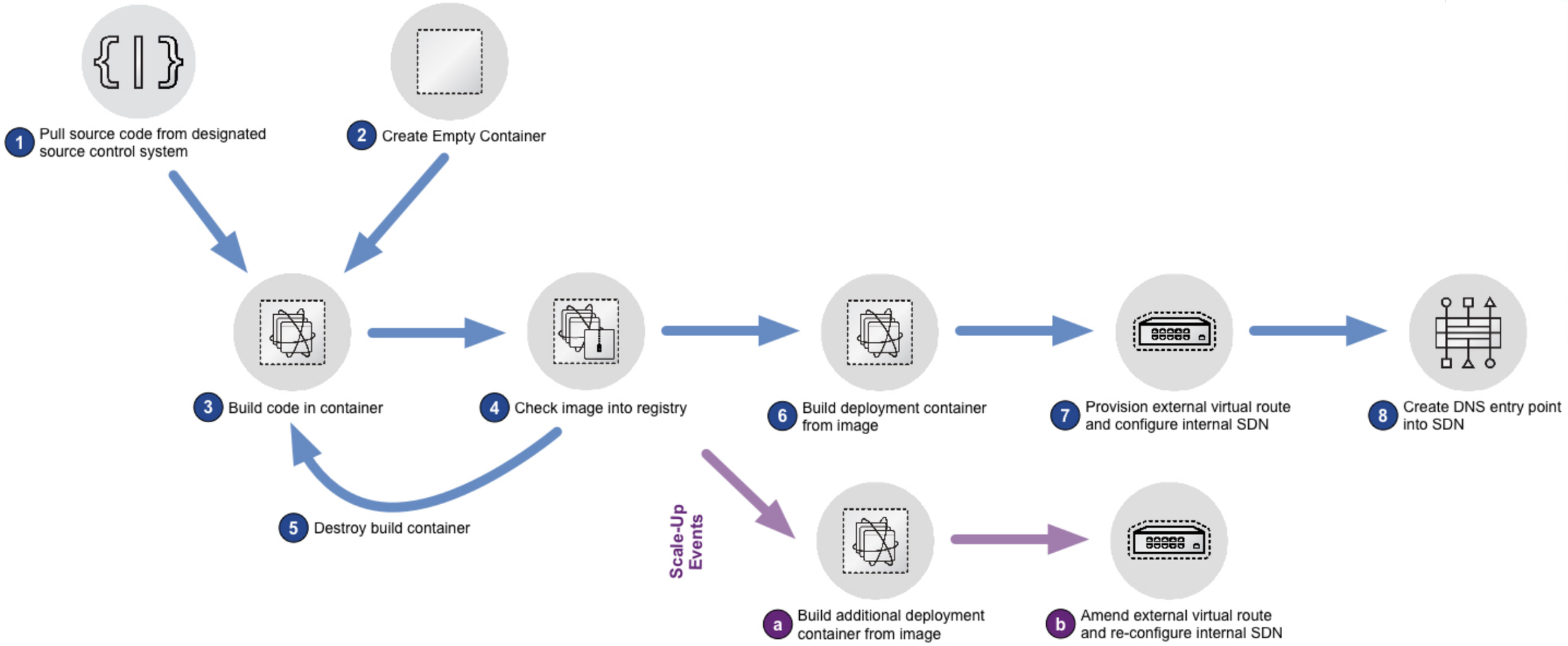
Behind The Scenes

What's actually happening here?



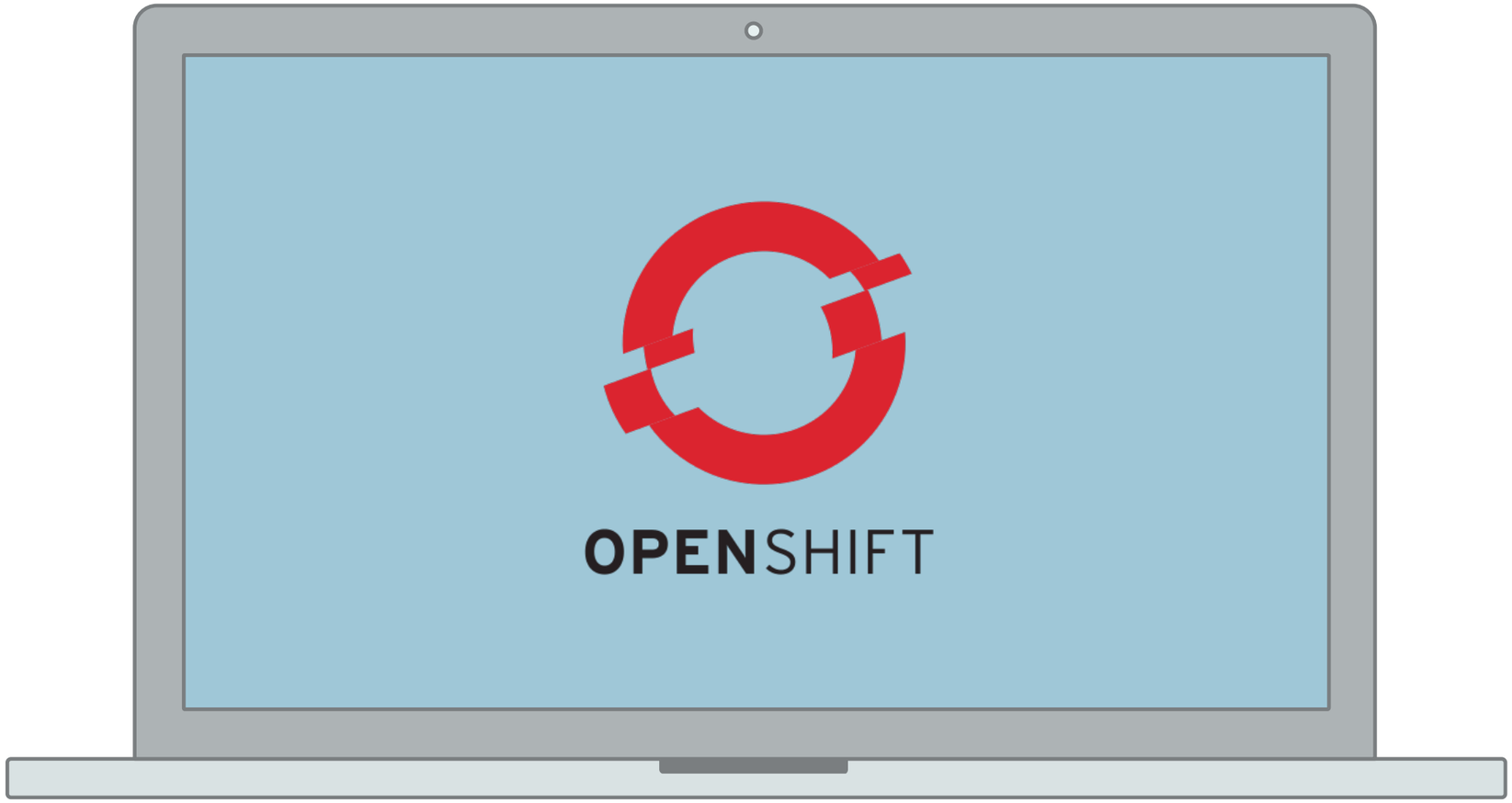
Behind The Scenes

What's actually happening here?

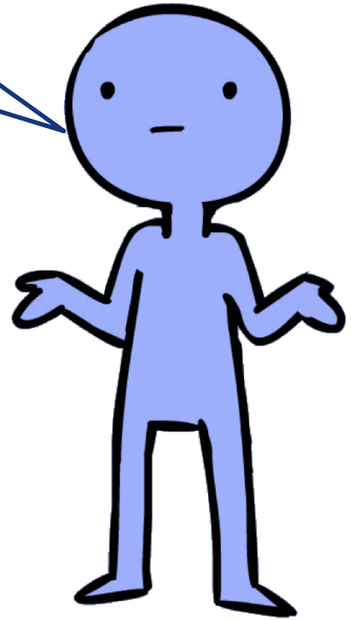


Developer Experience – Service Changes

Scaling up, adding components and enabling SSL



That’s nice, but it’s a bit simplistic?!?

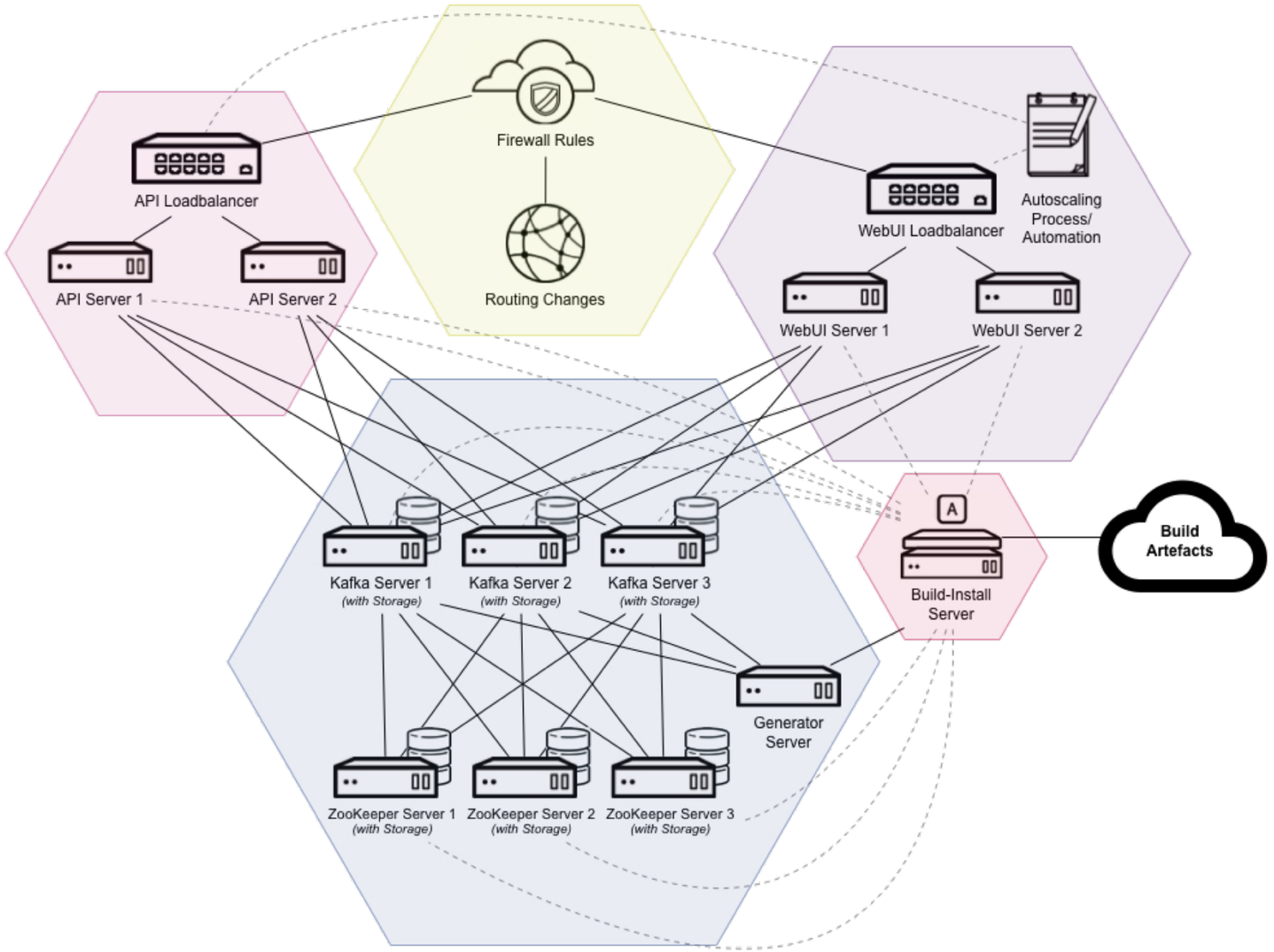


Demo - Complex Setup

What we're about to deploy...

Demo – Complex Setup

Current equivalent...



Bill of Materials

- Firewall Rules
- Routing Changes
- 2 x Loadbalancer VIPs
- 12 x Virtual Machines
- 6 x Storage Volumes
- Custom elastic scaling
  (typically over-provision)

Avg. Duration: 2+ weeks

Demo – Complex Setup

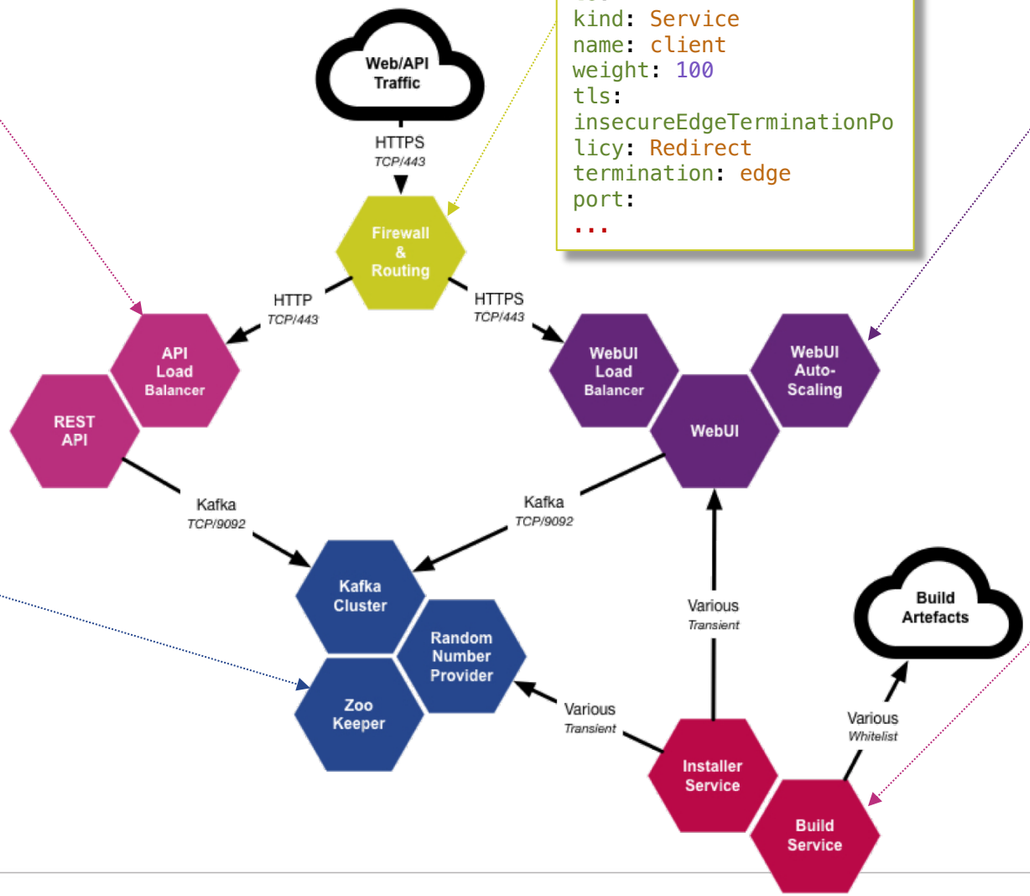
A picture is worth ~~1000 words~~ c.500 lines of code...

```
apiVersion: v1
kind: Service
metadata:
name: rest
labels:
app: rest-proxy
spec:
ports:
- name: rest
protocol: TCP
port: 80
targetPort: 8082
selector:
name: rest-proxy
---
apiVersion: v1
kind: ReplicationController
namespace: default
...
```

```
apiVersion: v1
kind: Route
metadata:
name: client
spec:
host: client-
demo.apps.appcanvas.net
to:
kind: Service
name: client
weight: 100
tls:
insecureEdgeTerminationPo
licy: Redirect
termination: edge
port:
...
```

```
apiVersion: v1
kind: DeploymentConfig
metadata:
generation: 1
labels:
app: client
name: client
spec:
replicas: 2
selector:
deploymentconfig: client
strategy:
activeDeadlineSeconds:
21600
resources: {}
rollingParams:
intervalSeconds: 1
maxSurge: 25%
maxUnavailable: 25%
timeoutSeconds: 600
...
```

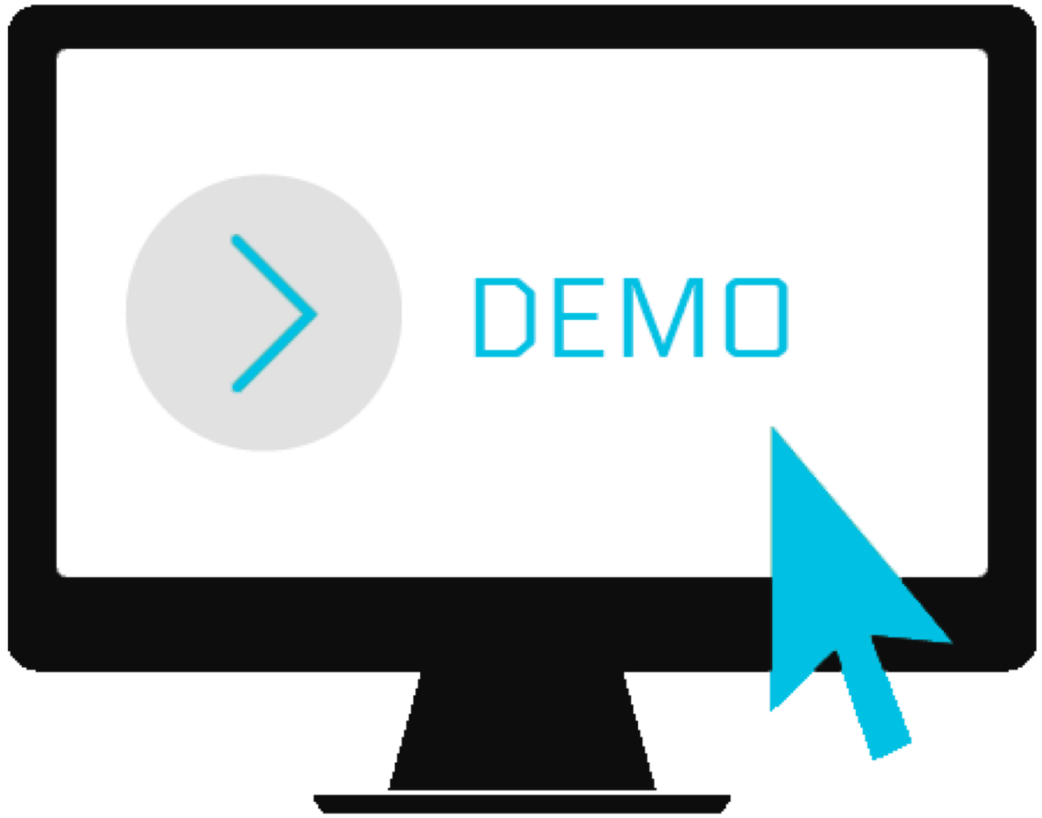
```
apiVersion: v1
kind: Service
metadata:
name: kafka
spec:
ports:
- name: 2181-tcp
protocol: TCP
port: 2181
targetPort: 2181
- name: 2888-tcp
protocol: TCP
port: 2888
targetPort: 2888
- name: 3888-tcp
protocol: TCP
port: 3888
targetPort: 3888
-  name: 9092-tcp
...
```




```
apiVersion: v1
kind: BuildConfig
metadata:
labels:
app: client
name: client
spec:
nodeSelector: null
output:
to:
kind: ImageStreamTag
name: client:latest
postCommit: {}
resources: {}
runPolicy: Serial
...
```

Demo – Complex Setup

Deploying highly-available MicroServices alongside Monoliths...



Beaker – Phase 2 Capabilities & FY18 Milestones

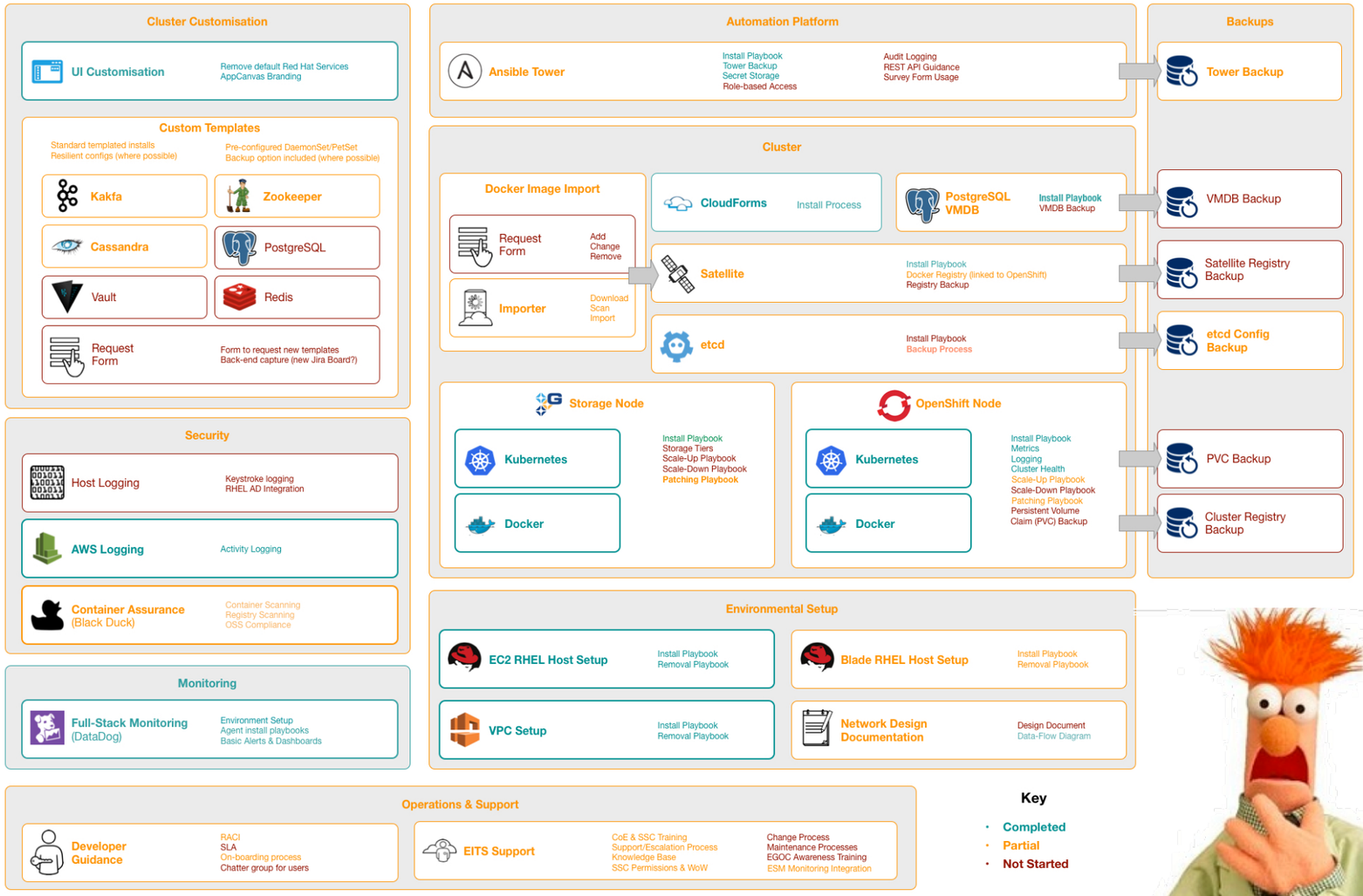
What we're working on...

Q1
- Reference Cluster
- Internal Cluster
- HP Synergy On-Site

Q2
- Begin UK External Cluster
- Begin US External Cluster

Q3
- Complete UK External Cluster
- Complete US External Cluster

Q4
- Iteration and Clean-Up
- Begin planning for addition of 'VMs'



Take-Aways…
Things to think about.

PaaS is a huge opportunity for Experian and EITS

- By demonstrating infrastructure innovation and adapting to modern cloud technologies, we stay relevant – both commercially and technologically
- We begin future proofing our infrastructure.
  Are we ever going to expend capex on a data-center build ever again?
  If not, shouldn't we be cloud-ready?
- We're already seeing that using these sorts of technologies attracts talent.
  It challenges an external stereotype that Experian are staid and don't innovate

How can I use PaaS and where is best to start?