


Agenda

Two sharpened pencils, one slightly above the other, pointing towards the top right. They are dark in color with visible wood at the tips.

- What is cloud native?
- Where are we today in the cloud native world?
- The Cloud Native ecosystem.
- Cloud Native adoption.
- Challenges.

A dark, atmospheric photograph of the Temple of Apollo in Side, Turkey. The temple features four prominent classical columns supporting a pediment. The structure is set against a dark, rocky background, and its reflection is visible in a pool of water in the foreground. The text "What is Cloud Native?" is overlaid in white, centered across the middle of the image.

What is Cloud Native?

What is Cloud Native?

“A new computing paradigm that is optimized for modern distributed systems environments capable of scaling to tens of thousands of self healing multi-tenant nodes”

-Cloud Native Computing Foundation

Pillars of Cloud Native

A photograph of a classical temple with four prominent columns, set against a dark, rocky background. The temple is illuminated from above, casting shadows on the ground. The columns are made of light-colored stone and have a fluted design. The temple's pediment is also visible, featuring some carvings. The overall scene is dark and atmospheric, with the temple standing as a central, well-lit structure.

Continuous
Delivery

Containers

Devops

Microservices

Pillars of Cloud Native: Devops

“DevOps is the practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support.”

-The Agile Admin blog


<https://theagileadmin.com/what-is-devops/>

Pillars of Cloud Native: Devops



- Generally based on principles of CALMS
- Based on the ideas of Automation, Measurement, Sharing
- Emphasis on a Collaborative culture in organizations
- Shifting operations more to the left.
 - Operations teams does more than just “server management”
 - Uses same techniques as developers for systems work

Pillars of Cloud Native: CD



- Origins in the automation segment of Devops
- Similar to Continuous Integration
 - Frequent code commits into source control
 - Run automated builds against each code commit
 - Result= Detecting errors quicker
- Continuous Delivery
 - Step 1: Continuous Integration
 - Release code builds to end users
 - Result= ship software quicker to end users

Pillars of Cloud Native: Microservices

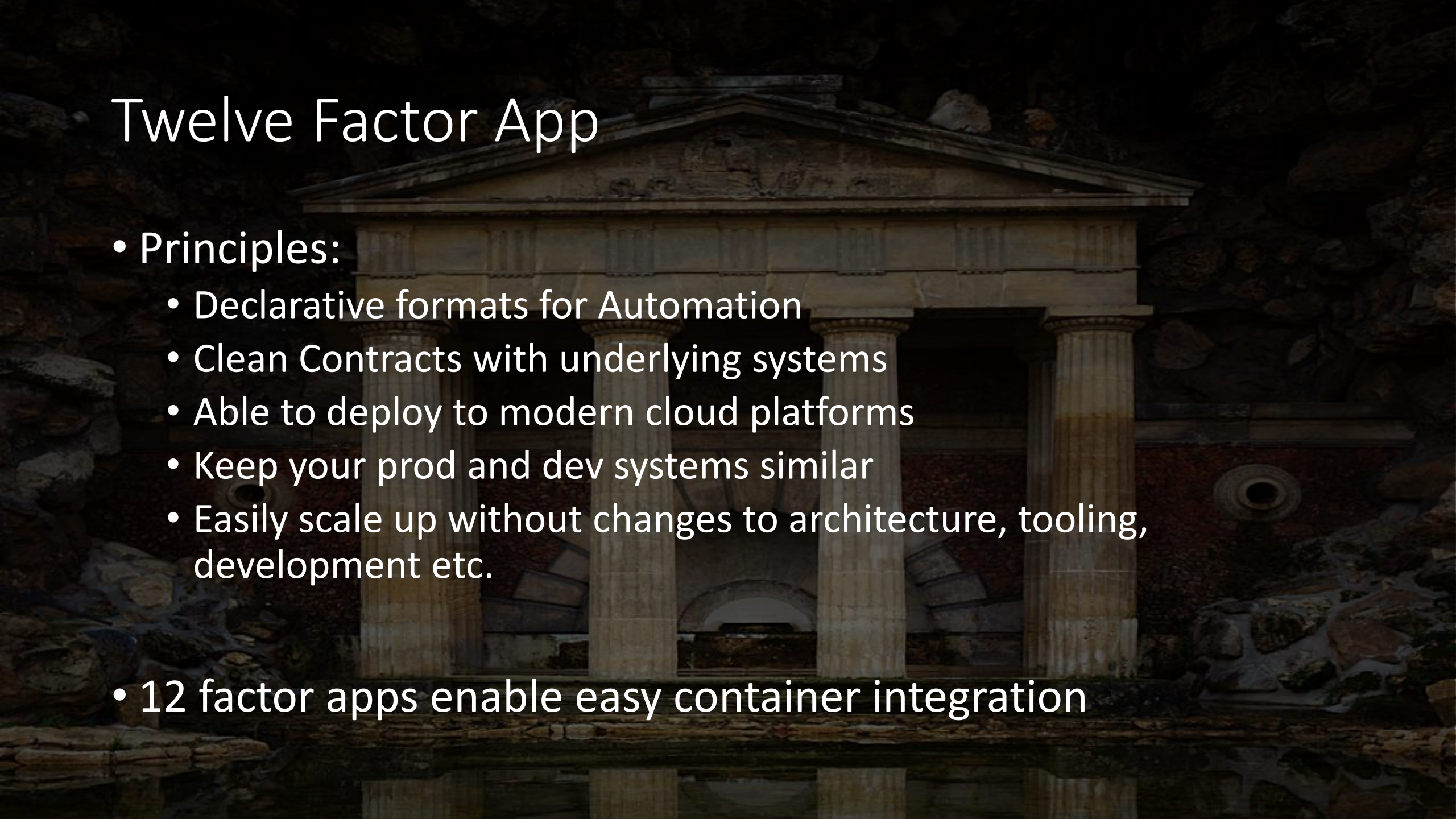


Microservices Design



- Start with Twelve-Factor App design
- <https://12factor.net>
- Based on the principals of software design and deployment at Heroku
- Development best practice that synergizes with devops engineers

Twelve Factor App



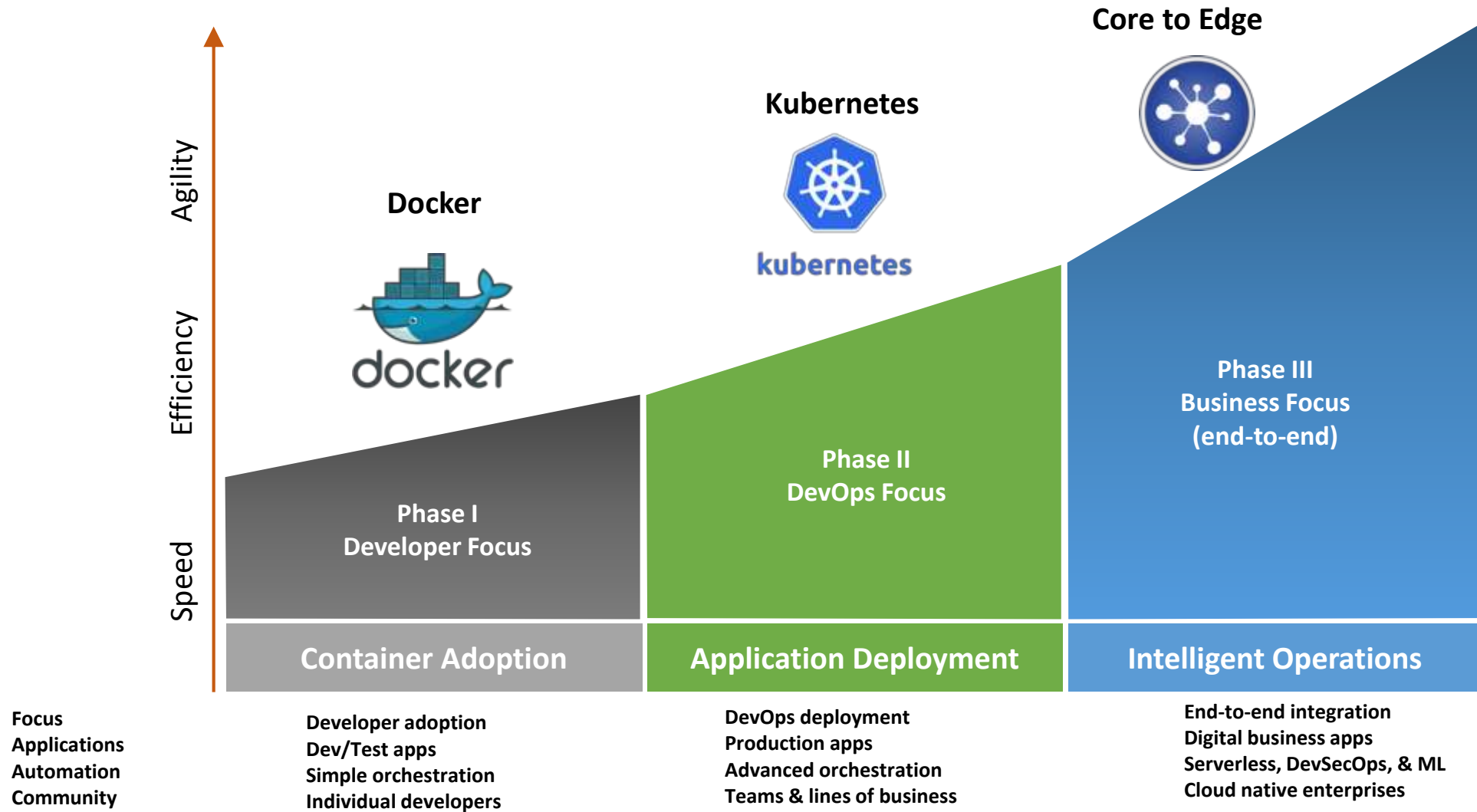
- Principles:
 - Declarative formats for Automation
 - Clean Contracts with underlying systems
 - Able to deploy to modern cloud platforms
 - Keep your prod and dev systems similar
 - Easily scale up without changes to architecture, tooling, development etc.
- 12 factor apps enable easy container integration


Pillars of Cloud Native: Containers

- Way to package applications
- Fits really well as a packaging strategy for microservices
- Not a new concept
- Popularized by the growth of Docker, and Kubernetes



Pillars of Cloud Native: Containers



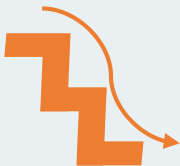



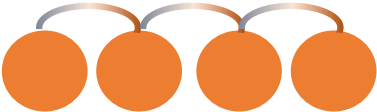

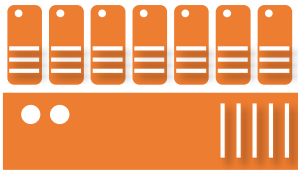

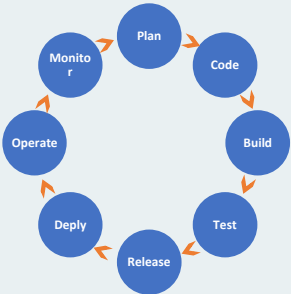
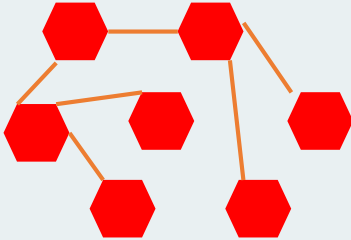
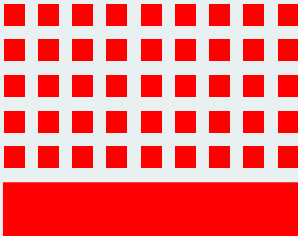
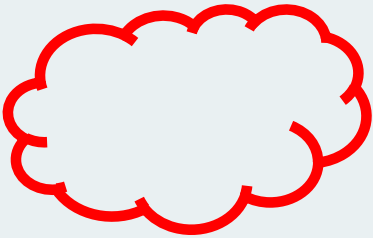
1		React.js 468 mentions
2		Kubernetes 335
3		Docker 252
4		Linux 240
5		Tensorflow 226
6		Vue.js 192
7		Laravel 170
8		Angular.js 123
9		Python 95
10	 	Ruby on Rails / Android 84 (tied)

Developer Trends in the Cloud: Open source Digital Ocean Survey, October 2018

Respondents=4300

<https://www.digitalocean.com/currents/october-2018/>

Evolution of Development and Deployment

	Development Process	Application Architecture	Deployment and Packaging	Application Infrastructure
~ 1980	Waterfall 	Monolithic 	Physical Server 	Datacenter 
~ 1990				
~ 2000	Agile 	N-Tier 	Virtual Servers 	Hosted 
~ 2010	DevOps 	Microservices 	Containers 	Cloud 
Now				

Cloud Native Usecases

Key Container Use Cases

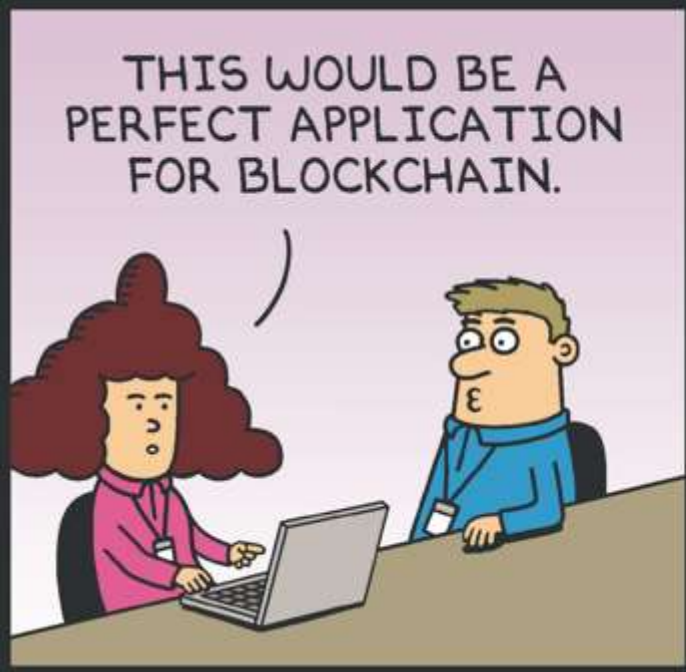
	Share	Container Use Cases	Orchestration Use Cases
Development	65%	Developer productivity; Consistent appstacks in Dev, Test & Production	Automated deploys to accelerate application release cadence
CI/CD/DevOps	48%	Containerized dependencies; Container registries;	Rolling updates and reversals
Operations	41%	Standardized environments for dev, testing and operations	Resilient, self-healing systems; High Availability; Elastic Scalability
Refactor Legacy Apps	34%	Refactor from N-tier to portable containerized applications	Run distributed, stateful apps on scale-out infrastructure
Migrate to Cloud	33%	Move entire appstacks and see them run identically in the cloud	Cloud bursting; Reduce infrastructure costs by avoiding over-provisioning
New Microservice Apps	32%	Create small purpose-built services that can be assembled to scalable custom applications	Dynamically manage large-scale microservices infrastructure

SOURCE: THE EVOLUTION OF THE MODERN SOFTWARE SUPPLY CHAIN, DOCKER SURVEY 2016

A close-up photograph of a small tabby kitten with green eyes, perched on a tree trunk and looking upwards. The kitten has brown and black stripes and white whiskers. The background is a soft-focus green forest.

New Technology??

Must Adopt!!



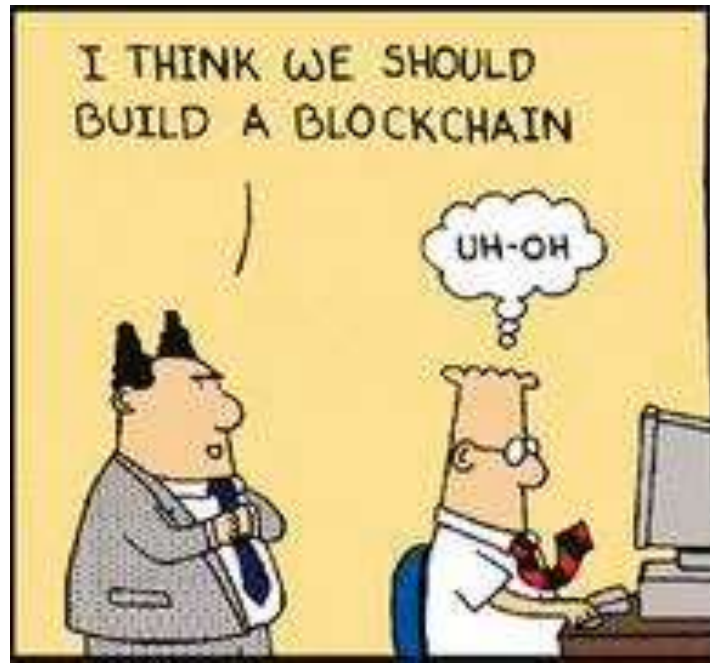
Dilbert.com @ScottAdamsSays



10-17-17 © 2017 Scott Adams, Inc./Dist. by Andrews McMeel



Business and Engineering



5 Adams E-mail: SCOTTADAMS@AOL.COM



11/17 © 1995 United Feature Syndicate, Inc.(NYC)



The Business Case for Cloud Native

- Quicker Time to Deliver
- Modernizing present day applications
- Develop new applications quickly
- Improve speed of innovation



Quicker Time To Deliver

- Containers + Microservices allows for a common language between your development and operations teams
 - Shared Understanding...
 - Allows for IT in general to practice a devops culture
 - Less friction between various teams in the organization
- Practicing Continuous Delivery allows you to ship faster
 - Process of making changes becomes easily
 - Reduces perceived risk of making changes

Modernizing present day applications

- Shipping applications in containers reduces dependencies on underlying infrastructure
- As a result, previous on premise applications can be exported to the cloud.
- Kubernetes provides a single unified platform to deploy containers across all your infrastructure

Develop new applications quickly

- Rich technical ecosystem.
- Large community
 - Kubernetes and CNCF slack has over 35k people
 - Plenty of meetups in many different cities
- Based on opensource
 - Developers can read the source code of platforms they are using
- Easier to find developers who want to work on newer technologies

Improve speed of innovation

- Cloud Native brings a new culture, technology and processes to accelerate innovation in organizations.
- Devops, CI/CD, Containerization modernizes your existing development teams
- Allows them to go much faster than before.



CLOUD NATIVE TRAIL MAP

The Cloud Native Landscape *landscape.cncf.io* has a large number of options. This Cloud Native Trail Map is a recommended process for leveraging open source, cloud native technologies. At each step, you can choose a vendor-supported offering or do it yourself, and everything after step #3 is optional based on your circumstances.

HELP ALONG THE WAY

A. Training and Certification

Consider training offerings from CNCF and then take the exam to become a Certified Kubernetes Administrator or a Certified Kubernetes Application Developer.
cncf.io/training

B. Consulting Help

If you want assistance with Kubernetes and the surrounding ecosystem, consider leveraging a Kubernetes Certified Service Provider.
cncf.io/csp

C. Join CNCF's End User Community

For companies that don't offer cloud native services externally.
cncf.io/industry

WHAT IS 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.

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.



1. CONTAINERIZATION

- Commonly done with Docker containers
- Any size application and dependencies (even PDP-11 code running on an emulator) can be containerized
- Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices

3. ORCHESTRATION & APPLICATION DEFINITION

- Kubernetes is the market-leading orchestration solution
- You should select a Certified Kubernetes Distribution, Hosted Platform, or Installer: cncf.io/csp
- Helm Charts help you define, install, and upgrade even the most complex Kubernetes applications



5. SERVICE MESH AND DISCOVERY

- CoreDNS is a fast and flexible tool that is useful for service discovery
- Envoy and Linkerd each enable service mesh architectures
- They offer health checking, routing, and load balancing



7. DISTRIBUTED DATABASE AND STORAGE

When you need more resiliency and scalability than you can get from a single database, Vitess is a good option for running MySQL at scale through sharding. Rook is a storage orchestrator that integrates a diverse set of storage solutions into Kubernetes.



9. CONTAINER RUNTIME

You can use alternative container runtimes. The most common, all of which are OCI-compliant, are containerd, rkt, and CRI-O.



2. CI/CD

- Setup Continuous Integration/Continuous Delivery (CI/CD) so that changes to your source code automatically result in a new container being built, tested, and deployed to staging and eventually, perhaps, to production
- Setup automated rollouts, roll backs and testing

4. OBSERVABILITY & ANALYSIS

- Pick solutions for monitoring, logging and tracing
- Consider CNCF projects Prometheus for monitoring, Fluentd for logging and Jaeger for Tracing
- For tracing, look for an OpenTracing-compatible implementation like Jaeger



6. NETWORKING

To enable more flexible networking, use a CNI-compliant network project like Calico, Flannel, or Weave Net.



8. STREAMING AND MESSAGING

When you need higher performance than JSON-RPC, consider using gRPC. NATS is publish/subscribe message-oriented middleware.



10. SOFTWARE DISTRIBUTION

If you need to do secure software distribution, evaluate Notary, an implementation of The Update Framework.



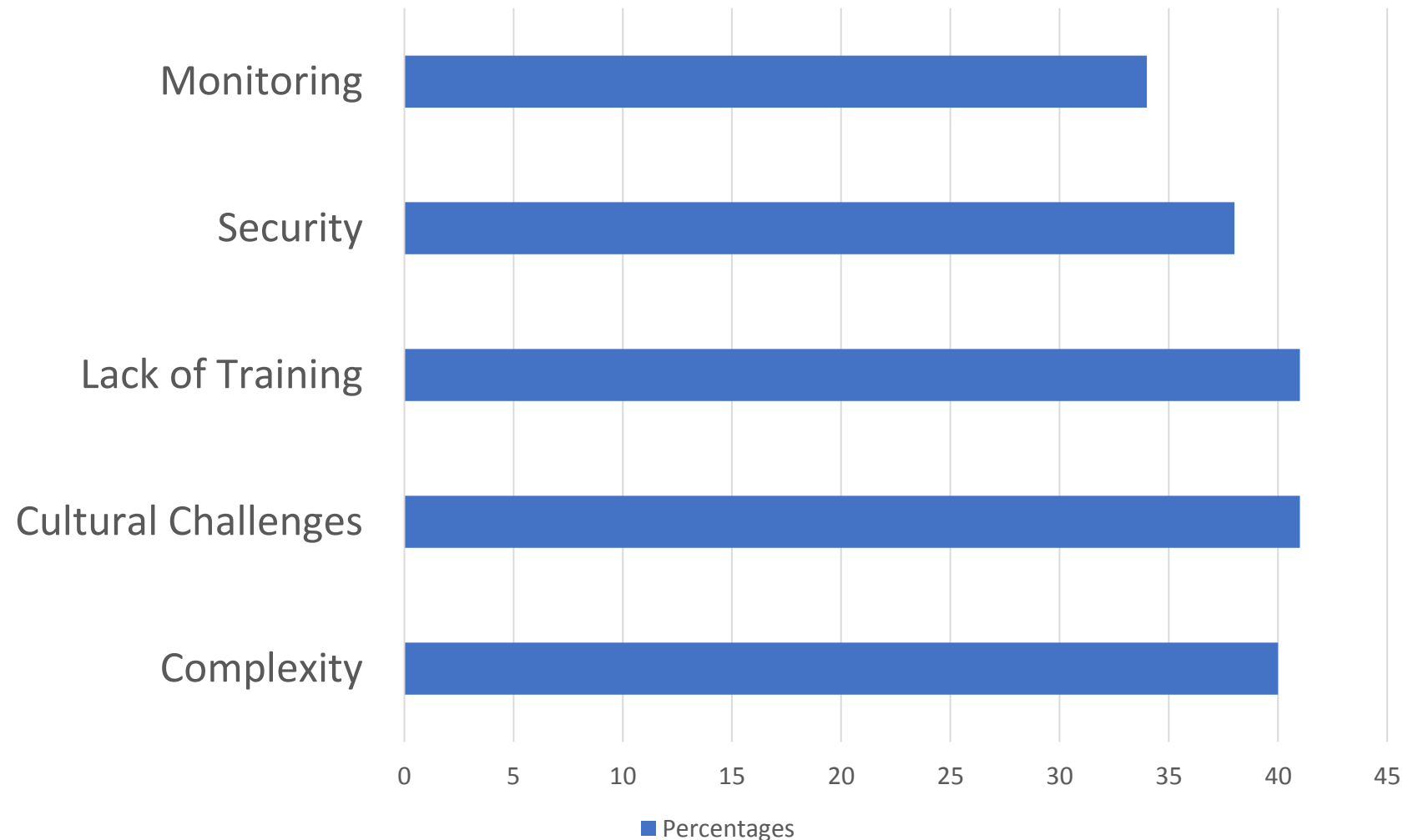
CNCF Trail Map

<https://landscape.cncf.io/images/landscape.pdf>

Challenges



Top 5 challenges to cloud native adoption...



Other Challenges

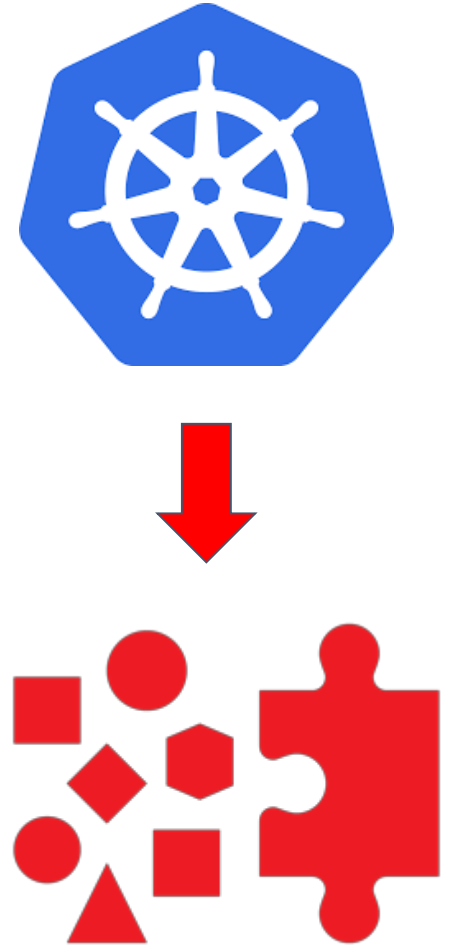
- Storage (30% down from 41%)
- Networking (30% down from 38%)
- Reliability (17% down from 20%)
- Logging (25% down from 32%)
- Scaling (20% down from 24%)



Kubernetes & Cloud Native Challenges

- Managing, maintaining, upgrading Kubernetes Control Plane
 - API Server, etcd, scheduler etc....
- Managing, maintaining, upgrading Kubernetes Data Plane
 - In place upgrades, deploy parallel cluster etc....
- Figuring out container networking & storage
 - Overlays, persistent storage etc... - it should just work
- Managing Teams
 - How do I manage & control team access to my clusters?
- Security, security, security

Source: Oracle Customer Survey 2018



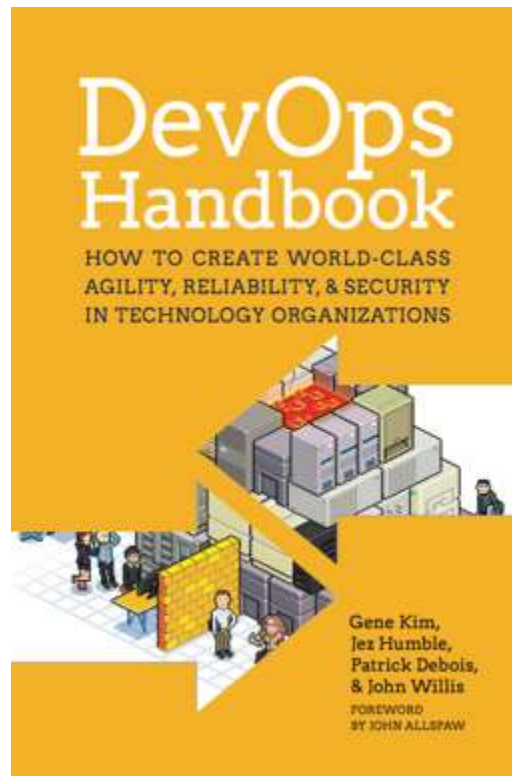
A photograph of a child standing on a wide set of stone steps. The steps are made of large, rectangular stone blocks and lead up to a wall of similar stone blocks. The child is wearing a cap and overalls. The scene is dimly lit, with a strong light source from the right creating a bright diagonal beam across the wall and steps. The overall mood is contemplative and quiet.

Where do I start?

Some strategies...

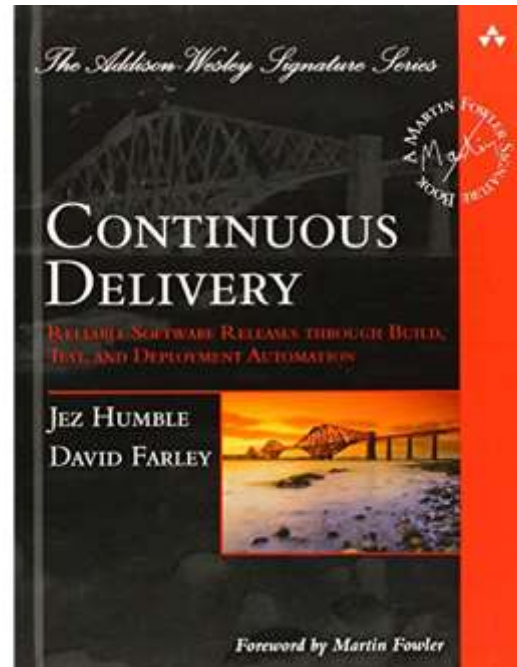
Silos

- Heavily siloed organizations can benefit from a devops mindset
- Use containers as a way to break down silos in your engineering orgs

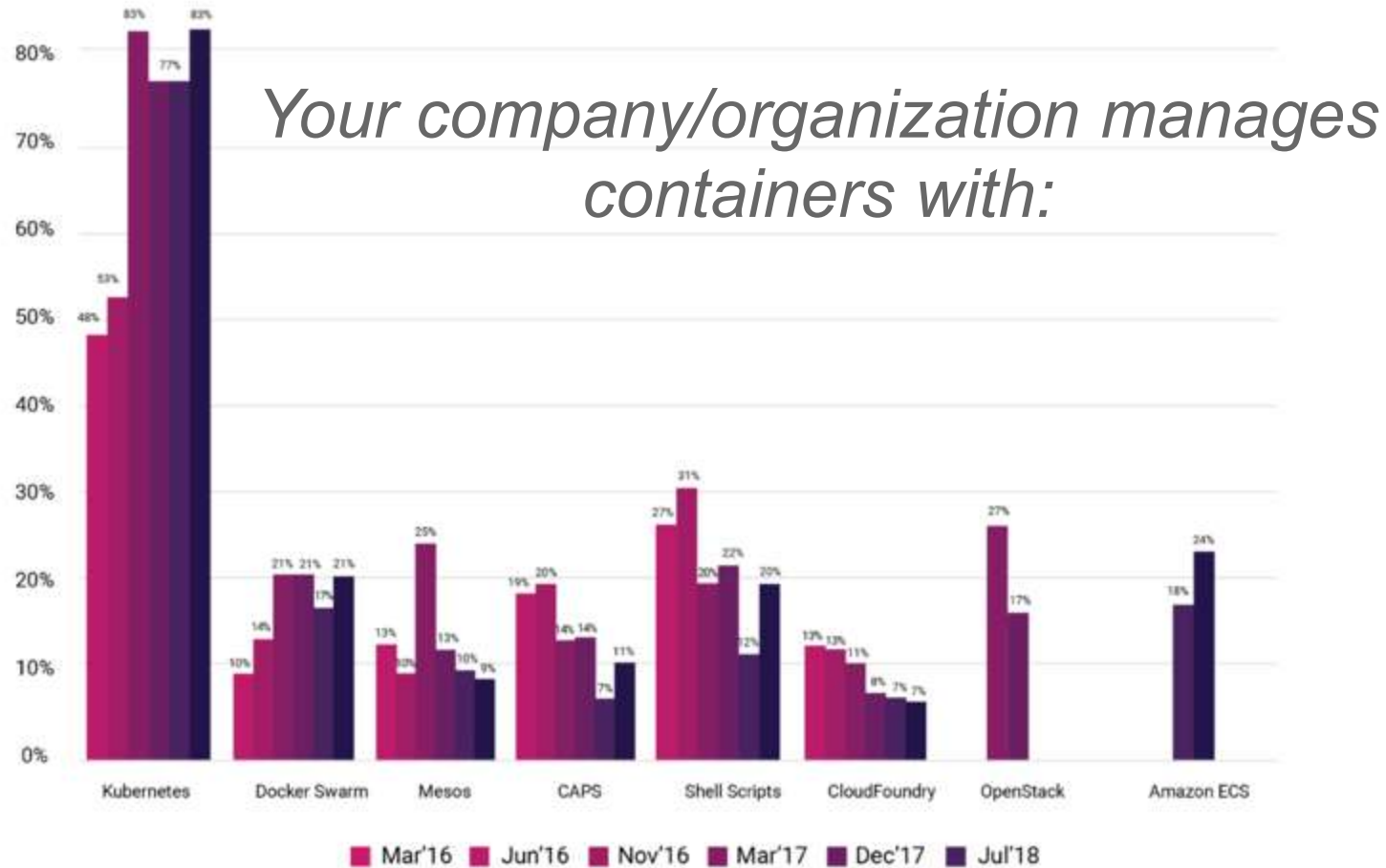


Releasing Code

- Step 1: Invest in Continuous Integration
- Step 2: Continuous Delivery



Orchestration?



A photograph of a vintage IBM mainframe computer room. In the foreground, a man in a suit is seated at a console, looking at a document. To his right, a large, multi-colored (red, blue, and black) mainframe cabinet stands. In the background, another person is visible near a similar cabinet. The room is filled with various computer components, including tape drives and control panels. The text "Nobody ever got fired for buying IBM. using Kubernetes." is overlaid on the image in white, with a red underline under the word "buying".

Nobody ever got
fired for buying IBM.
using Kubernetes.

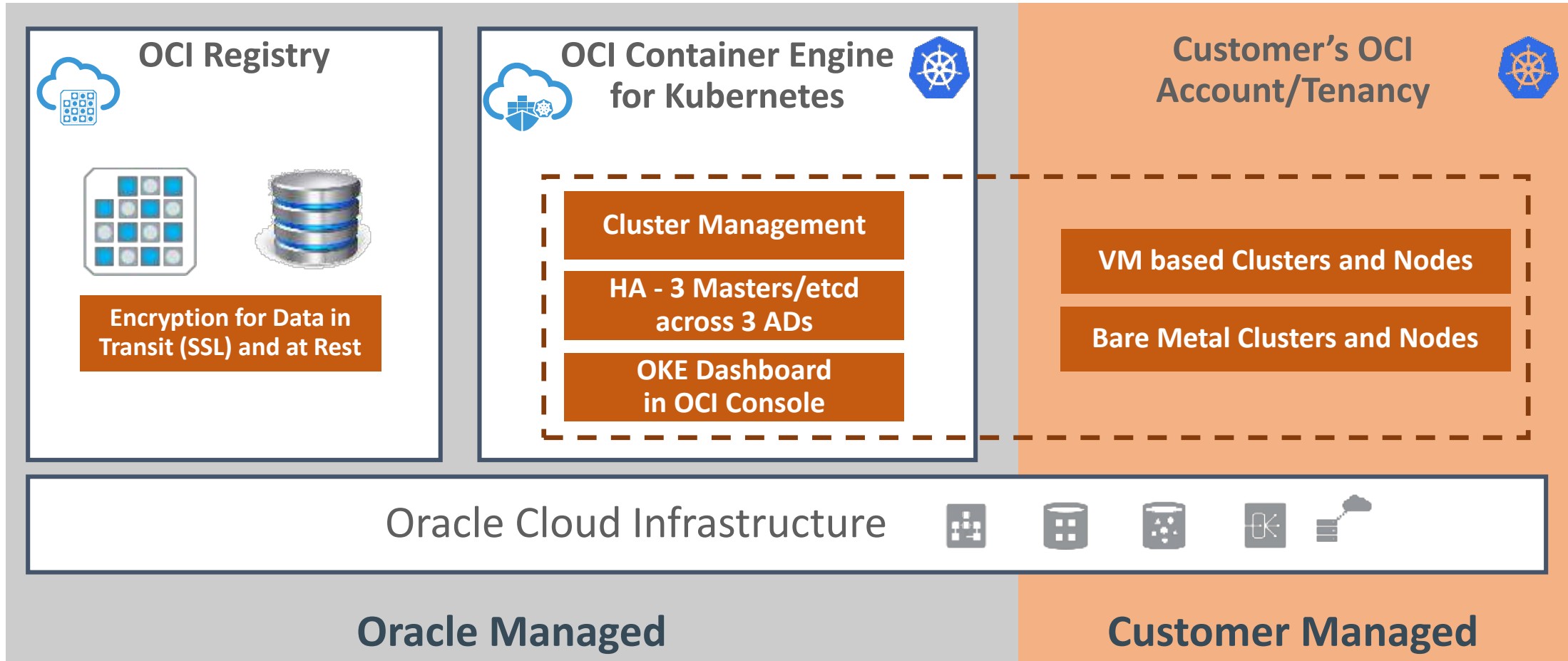
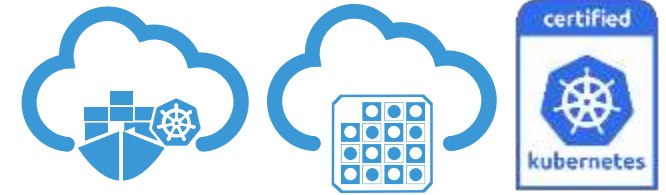
A man with glasses and a light-colored sweater is sitting in a server room, looking down at a document or a small screen he is holding. He is surrounded by a massive, chaotic tangle of white and orange network cables that fill the background and foreground, creating a dense, almost overwhelming visual. The lighting is dim, with some highlights on the cables and the man's face.

Biggest Lie: “Kubernetes is easy”

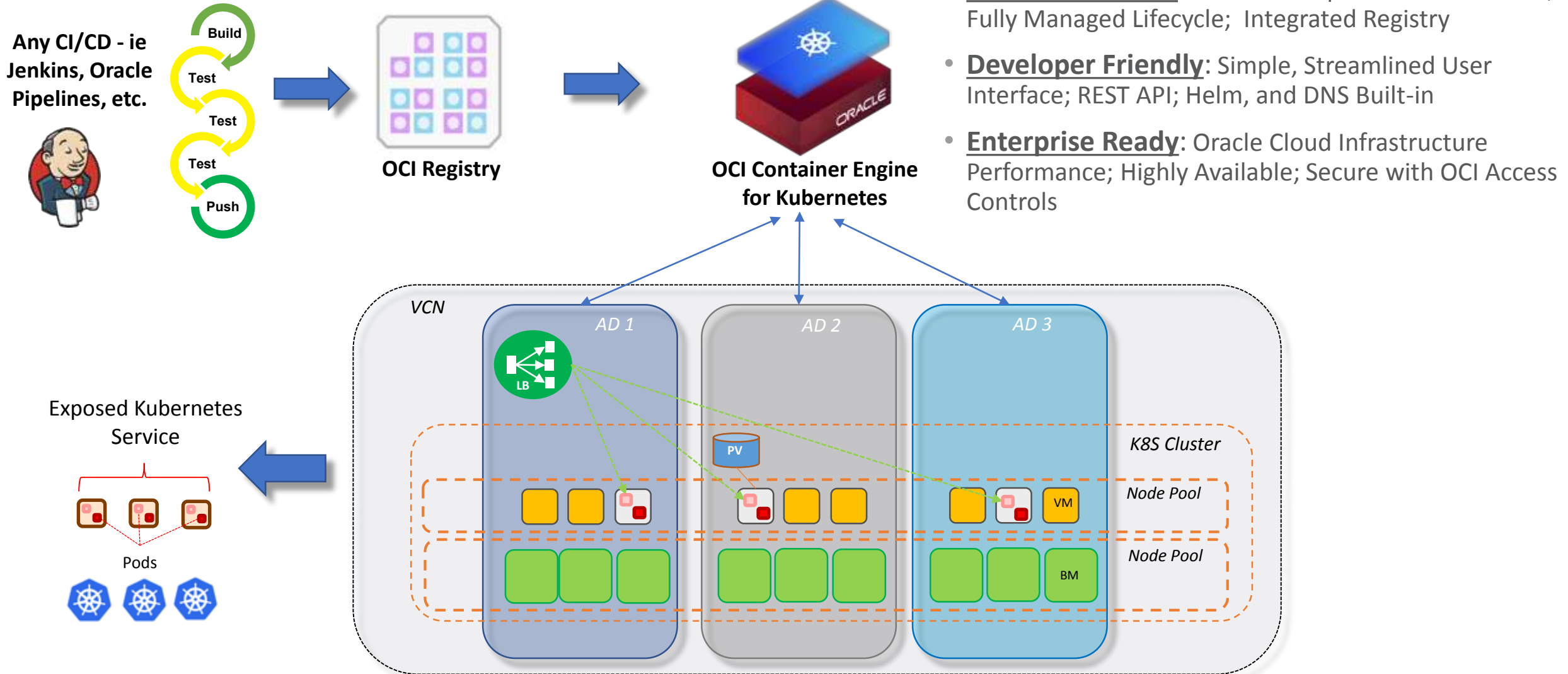
Kubernetes is complex

- Use a Kubernetes Managed Service
 - Like Oracle Container Engine for Kubernetes, Google Kubernetes Engine etc
- Benefits:
- Enables developers to get started and deploy containers quickly.
- Gives DevOps teams visibility and control for Kubernetes management.
- Combines production grade container orchestration of open Kubernetes, with control, security, IAM, and high predictable performance of cloud infrastructure
- Manage what you really need to manage

Kubernetes is Complex



End to End Workflow...



Take It slow...

Greenfield OR Strangler application patterns

