Beyond Virtualization

Derek Collison - Apcera, Inc.
@derekcollison

June 12, 2014 - QCon New York

About

Derek Collison

- Architected and built TIBCO Rendezvous and EMS Messaging Systems
- Co-founded AJAX APIs group at Google
- Designed and built Cloud Foundry
- Founder and CEO at Apcera
- Inspiration: Fast Distributed Systems

The future of enterprise IT lies beyond virtualization

Virtualization ==

EVERYTHING is a distributed system these days

So orchestration and composing systems will define the future

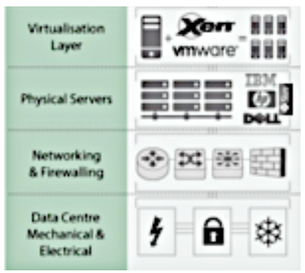
To look into the future

Let’s see where we are

IT Today

Old school

Virtualization



IaaS

IaaS, SaaS, PaaS

Cloud

?

We care about what's next —?

Automate undifferentiated heavy lifting, speed up the mundane

Orchestrate Secure and Compliant Composeable Systems

Align the value to you with the value to your organization

Build what you need..

Assemble the rest

PaaS helps

PaaS Helps

- Tries to speed up deployment
- Preset, biased approach
- Only a small piece of the puzzle
  - Enterprises need lifecycle management, security, compliance, governance, etc.

PaaS is Not Enough

http://apcera.com/blog/paas-is-not-enough/

Docker helps

Docker Helps

- The dawn of the composeable enterprise
- More control over the pieces
- Great Ecosystem!

DockerCon Initiatives

- libSwarm
- libContainer
- libChan

Docker The Future

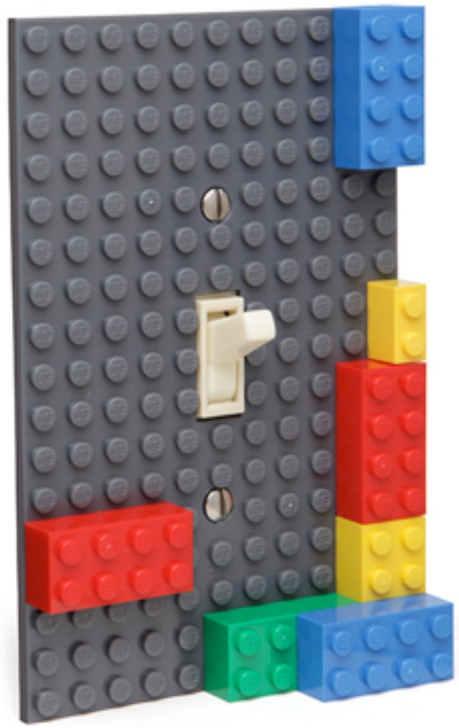
- Identity
- Authorization
- Trust

Docker TBDs

- How to compose and orchestrate the system?
  - etcd? confd?
  - Make it transparent
  - Don’t make me rewrite
  - libSwarm, libChan?
- What about compliance?
  - Heartbleed?
  - Linux zero-day exploit?
  - Tell me if I am compliant
  - Tell me what is at risk

We Want Things to Just Work

- Self Service
- Composeable Systems (legos)
- Faster Iterative Development
- Faster Deployments
- Fault Tolerance
- High Availability
- Guaranteed SLAs



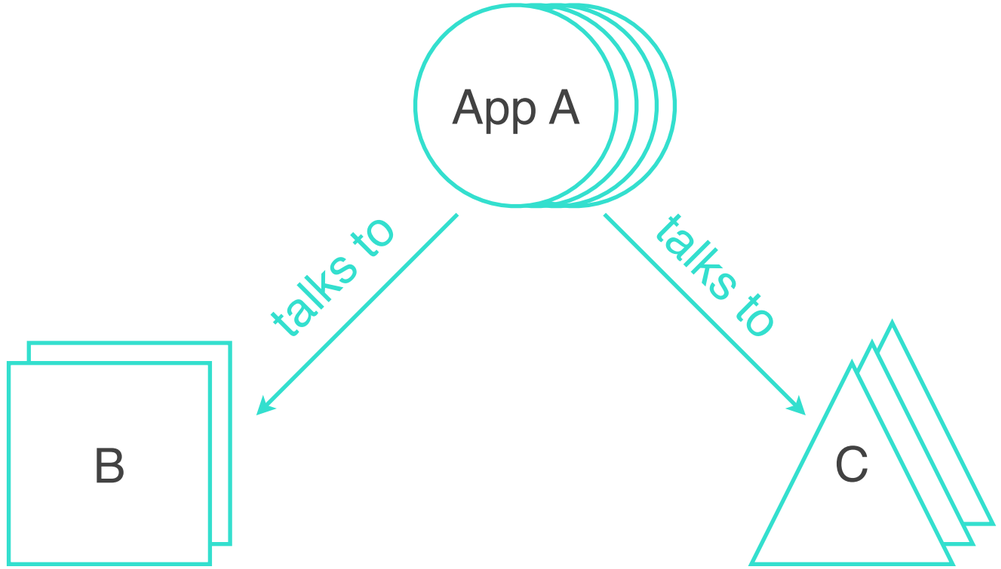
We’re getting there

The Future of IT

- Declarative
- Composeable
- Extreme Agility
- Security and Compliance - Transparently
- Fluid and Abstracted Infrastructure and Services
- Multiple delivery models in one system

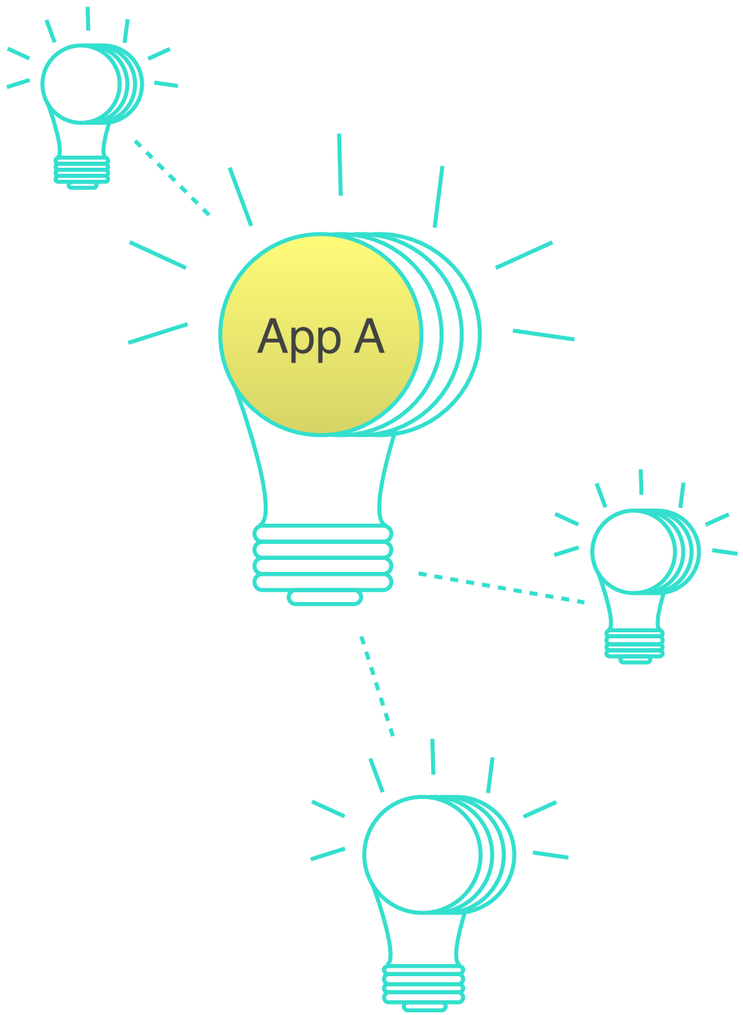
Declarative

- App A needs:
  - X memory and Y CPU
  - N storage
  - I/O SLAs for talking to B and C
  - available URL for trusted identities
  - run on premise, co-located near B



Intelligent workloads

Intelligent systems



Where do we start?

Required Functionality

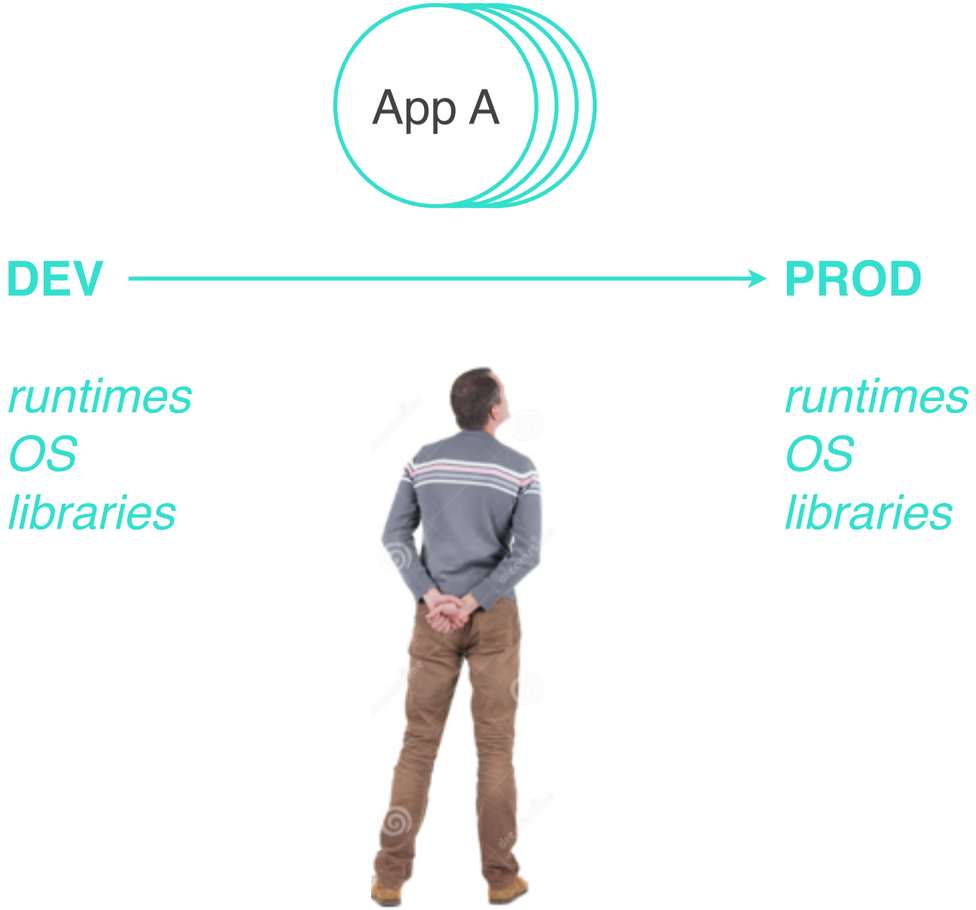
- What App A needs
- Where App A runs
- How App A finds B and C
- How others find App A
- What happens on failures

Required Functionality

- What App A needs
  Packaging & Dependencies
- Where App A runs
  Provisioning & Scheduling
- How App A finds B and C
  Addressing & Discovery
- How others find App A
  External Mapping
- What happens on failures
  Monitoring & Management

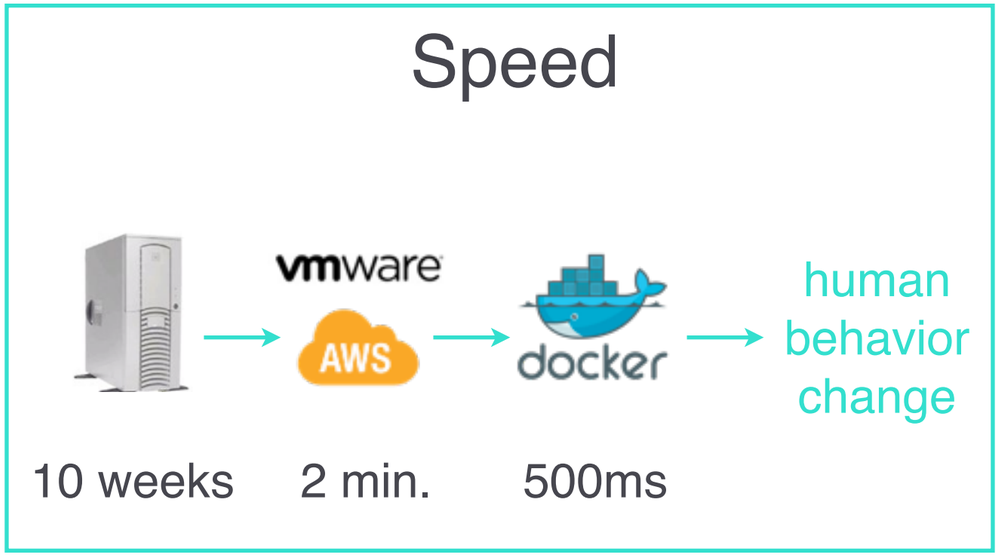
Packaging & Dependencies

- What the job needs to run
- Changes from Dev to Prod
- Runtimes, OS, libraries
- Who defines what these are
- Whether existing tools are sufficient for consistency, compliance, auditing
  - SCCS and Chef / Puppet
  - AMIs or VMDKs
  - Docker Images



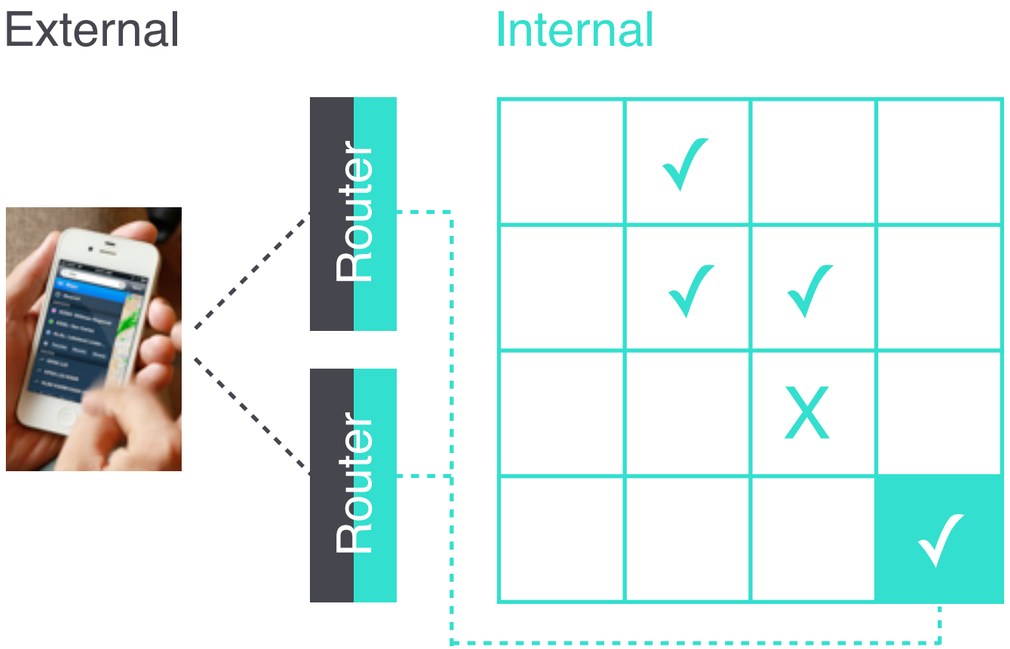
Provisioning & Scheduling

- Where workloads run
- Network perimeter security models
- Unit of work: VM, App, Image
- Automatic, instantaneous and transparent policy compliance
- Compliance and deployment handled independently
- New tools: Mesos, Fleet, Diego



Addressing & Discovery

- DNS is insufficient - inside
- Needs to fit what we have, without changing apps
- System reacts as things move
- Load balancing
- Scaling up and down



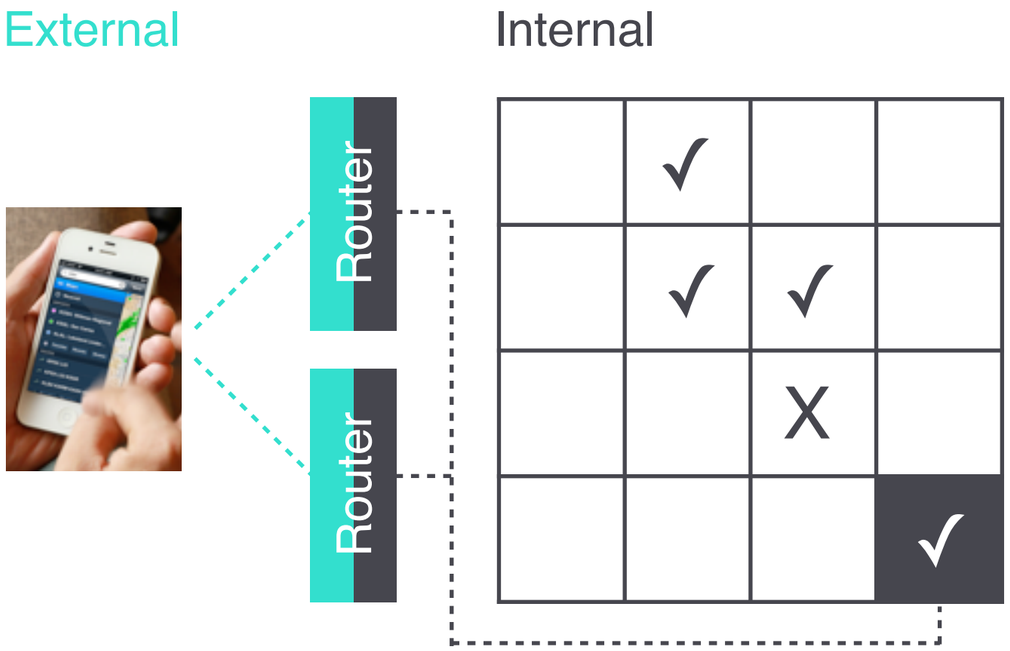
CoreOS

CONSUL

ETCD / CONFD

External Mapping

- HTTP/TCP connectivity
- How do you find something?
- Load balancing
- Rapid scaling
- Health monitoring and repair
- DNS sufficient for external, but not internal



Monitoring & Management

- What happens when something fails?
- Manual or Automatic?
- Who determines failure and whether we trust the system
- Its sick, not dead
  - Latency vs. Chaos monkey
- Measure the effect of change beforehand?
- Extensible & Pluggable

Google

BORG / Omega

Chaos

Latency

Bolt-on is not the way to get there

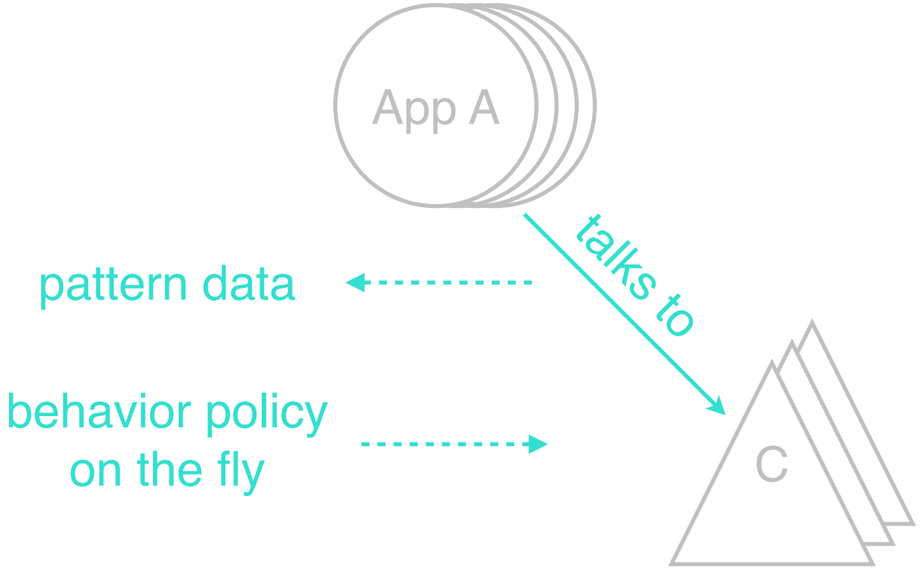
What we need is a platform OS

Programmable, pluggable, and composeable from the inside out

The secure, hybrid, trusted platform OS for multi-datacenter

A Platform OS

- All resources in a common pool
- Real-time networking, addressing, and discovery
- Awareness of ontologies **AND** communication semantics
- Contextual security and policy just work
- Built for rapid change - all change
- Policy-compliant resource isolation, connectivity, and SLAs



We Have the Right Pieces

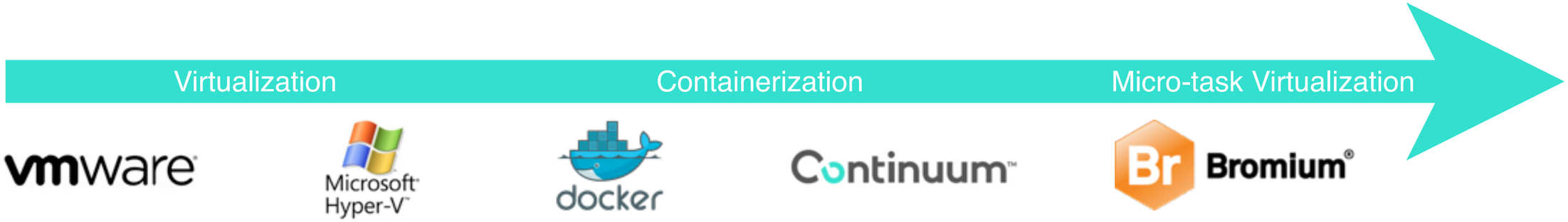
- Isolation Contexts - Docker
- SDN - Software-Defined Networking
- Management and Resource Pooling (CMPs)
- Intelligent and Compliant Job Scheduling
- Intelligent Canarying, A/B rollouts and testing

Just not in one place

Isolation Context

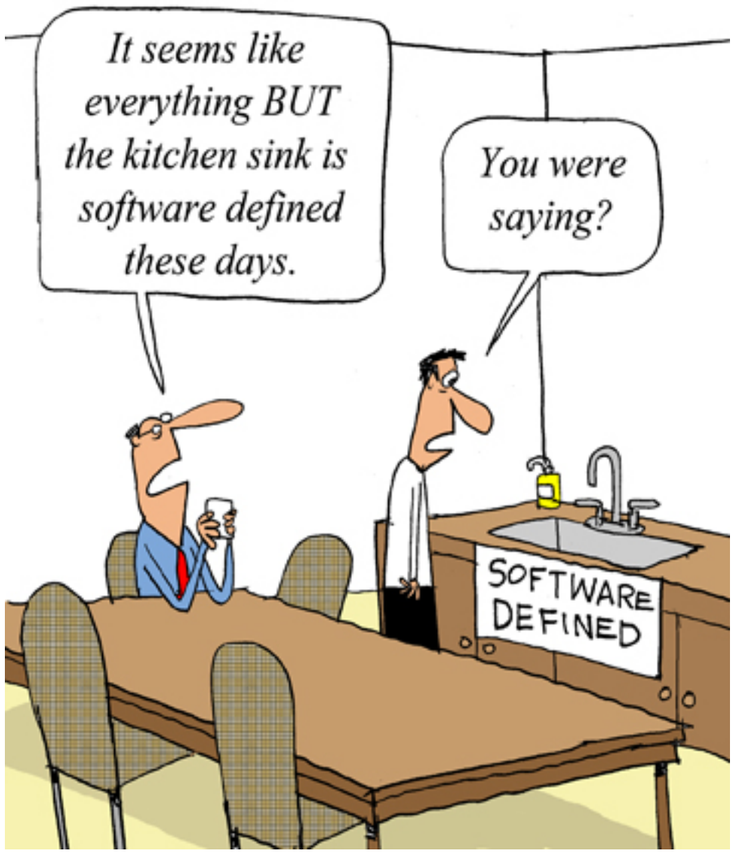
- Isolation Context: isolated, insulated, autonomous
- Speed and weight
  - Hypervisors for virtualization
  - LXC, libContainer (containers) - Docker
  - Micro-task virtualization
- Google chargeback diversion

Faster, more lightweight and purpose-built



SDN – Software-Defined Networking

- Network perimeter security
- Application-level changes
- Layer 7 semantics
  - How many INSERTS per second from all of App A?
  - Can I disallow DROP and DELETE calls between 1-3AM?
- Compliant and transparent network
  - It just works, e.g. mobile

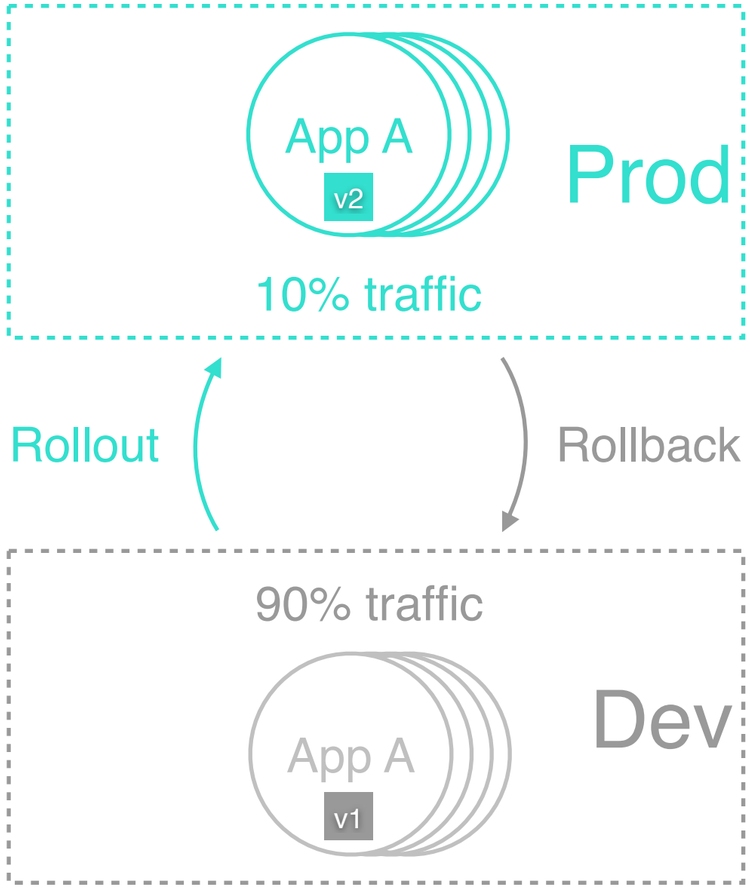


Intelligent, Compliant Job Scheduling

- Pick the best place to run for a given job and policy
- How the system rebalances and utilizes new resources
- Centralized or Distributed algorithms
- How policy affects decision-making (e.g., geography)
- New tools: Mesos, Fleet, Diego

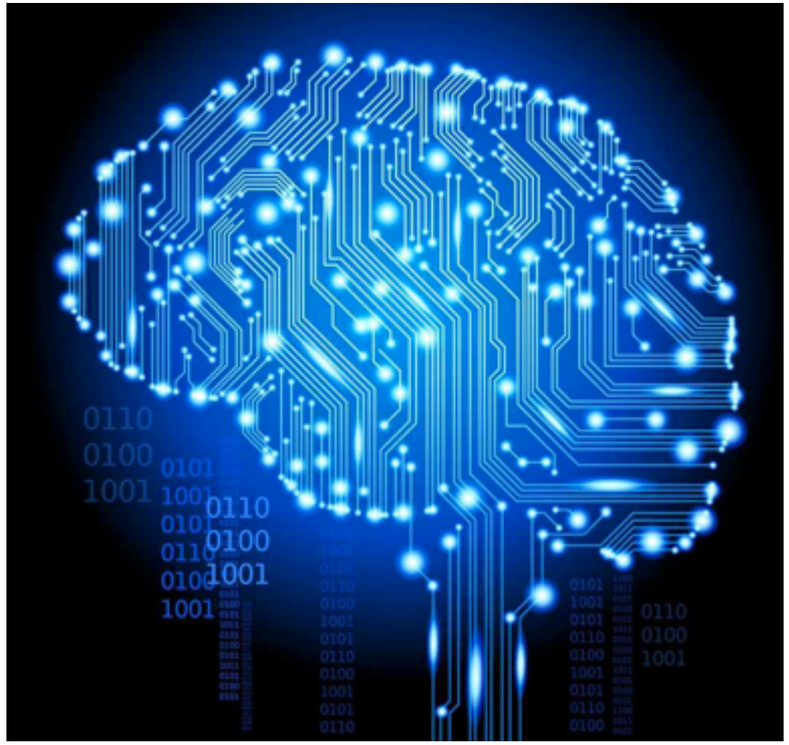
Intelligent Canarying

- Measured rollout success
- A/B testing
- Blue-green deployments
- Automated rollout and rollback

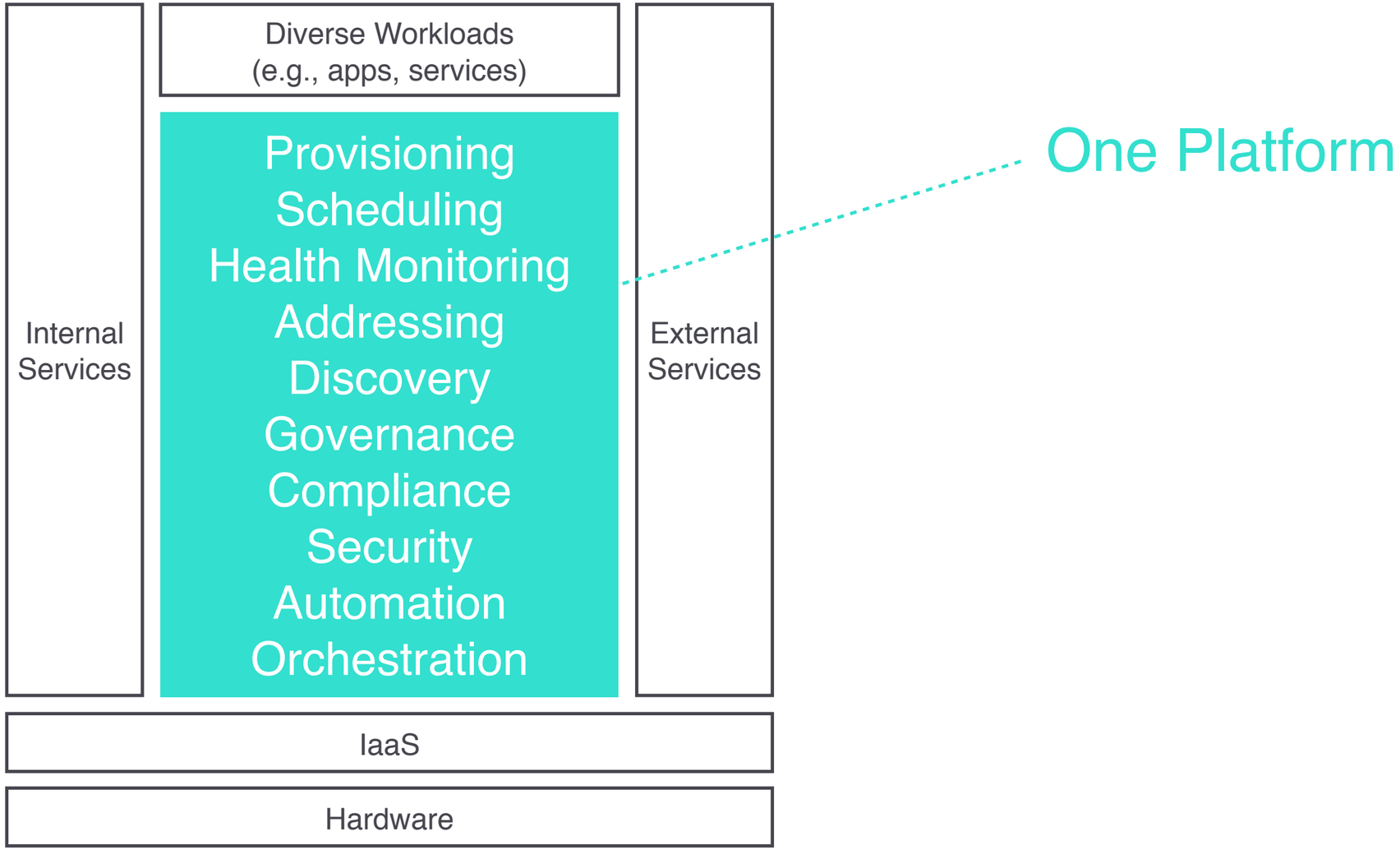


Intelligent Canarying

- A lot of data needed
  - resource utilizations: CPU, Mem, Storage
  - communication patterns: cascading effects
  - temporal awareness
- All data will feed into automated, anomaly detection services
  - Utilizing unsupervised deep machine learning



The Future of IT – Platform OS



Summary

Summary

- Composeable platforms
- Intelligent workloads sans code changes
- Policy aware...
  - Packaging and Dependency Management
  - Job Scheduling and Provisioning
  - Addressing, Discovery, Networking
  - Monitoring and Management
  - Lifecycle Management and Intelligent Canarying

A POLICY OF INNOVATION

Resources

- Docker - https://www.docker.io
- Mesos - http://mesos.apache.org
- CoreOS - https://coreos.com
- Fleet, Etcd - https://coreos.com/using-coreos/etcd
- Consul - http://www.consul.io
- Continuum - http://apcera.com/continuum

Thank You

Derek Collison - Apcera, Inc.
@derekcollison

June 12, 2014 - QCon New York