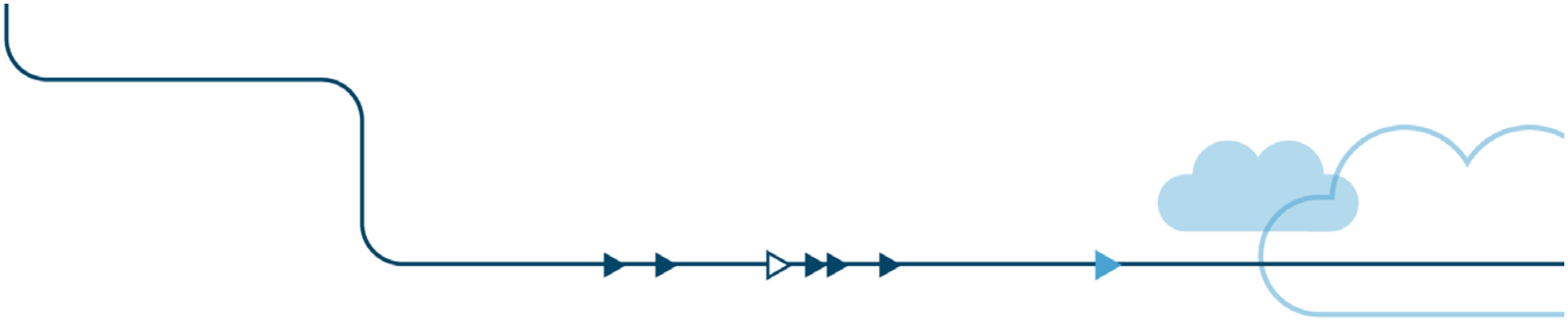
Cloud Native Cost Optimization

Adrian Cockcroft @adrianco
Technology Fellow - Battery Ventures
ICPE - Austin, February 2015

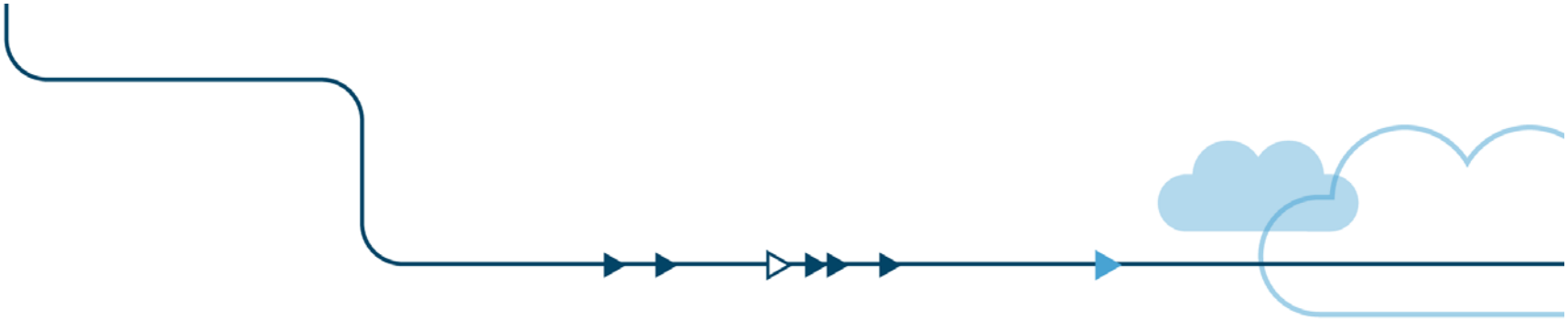
BV
Battery Ventures



Why Does Performance Matter?

@adrianco

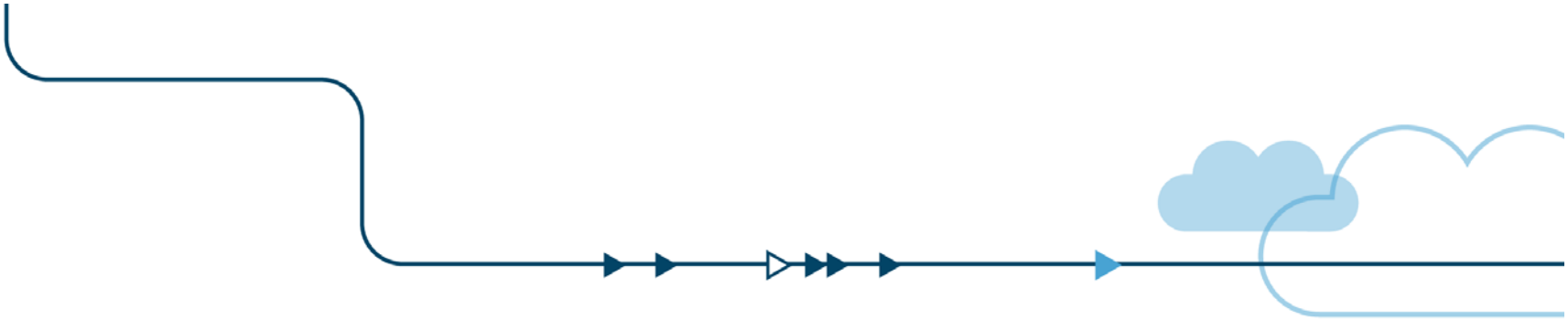
BV
Battery Ventures



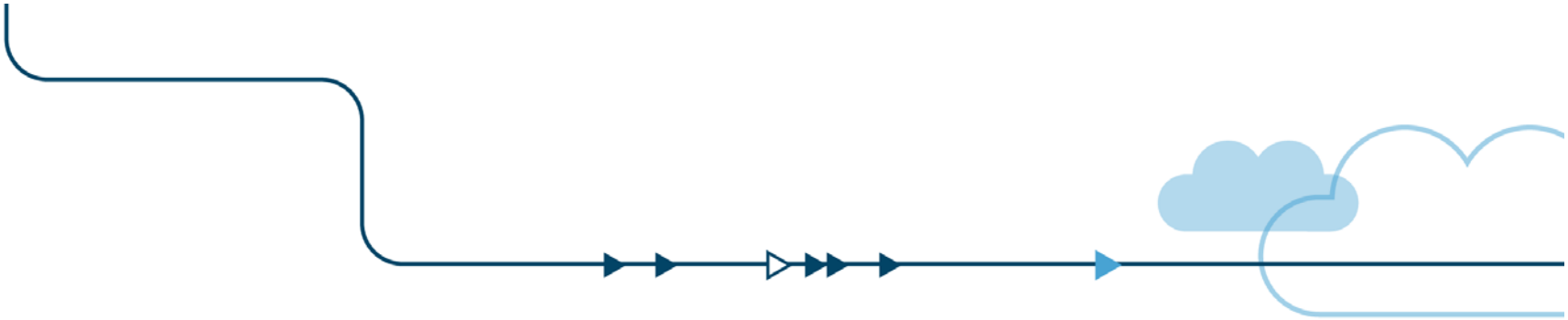
Latency Efficiency

@adrianco

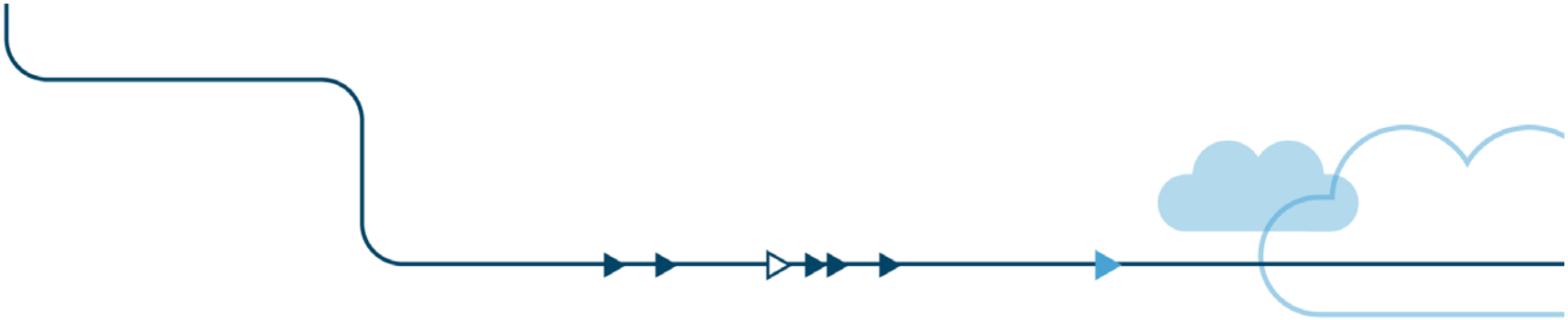
BV
Battery Ventures



Users: Response Latency
Developers: Release Latency
Operators: Efficiency

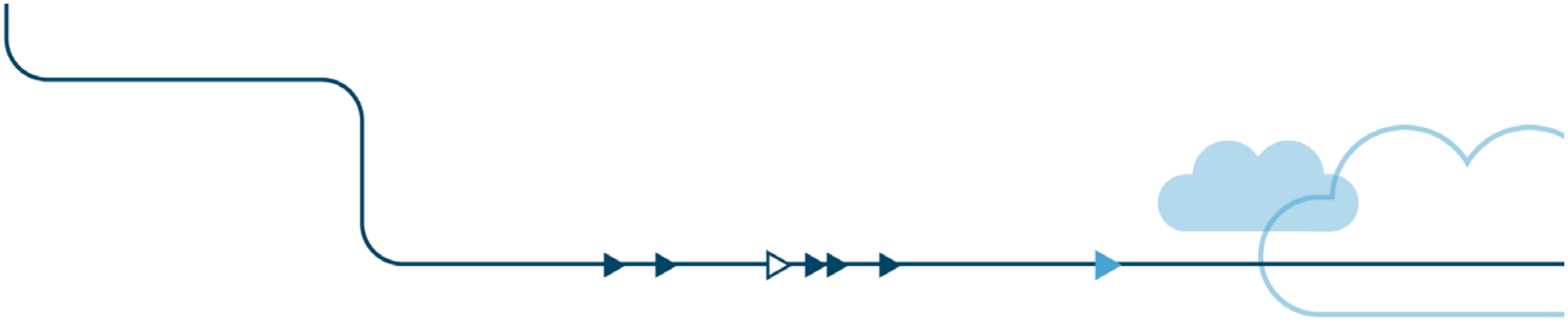


Less Time
Less Cost



Faster Delivery

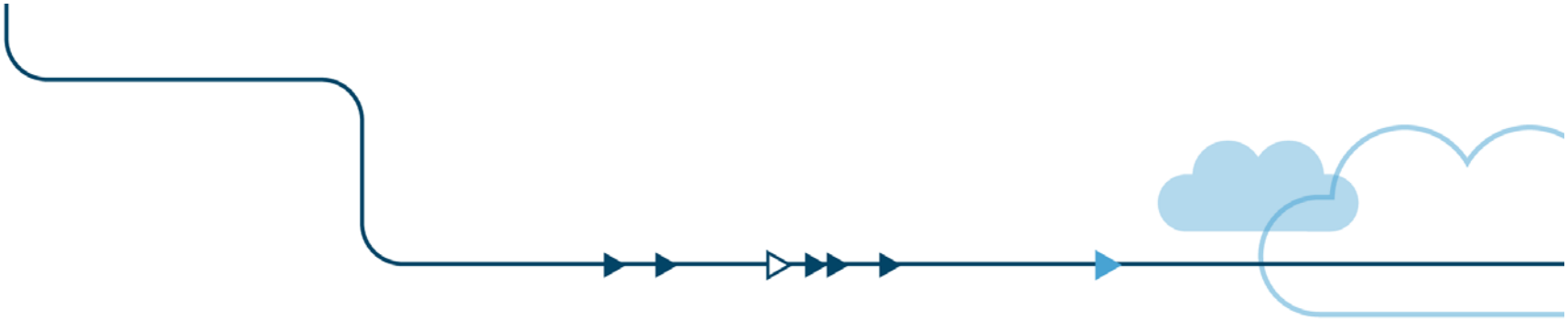
See talks by @adrianco
Speed and Scale - QCon New York
Fast Delivery - GOTO Copenhagen



Cheaper

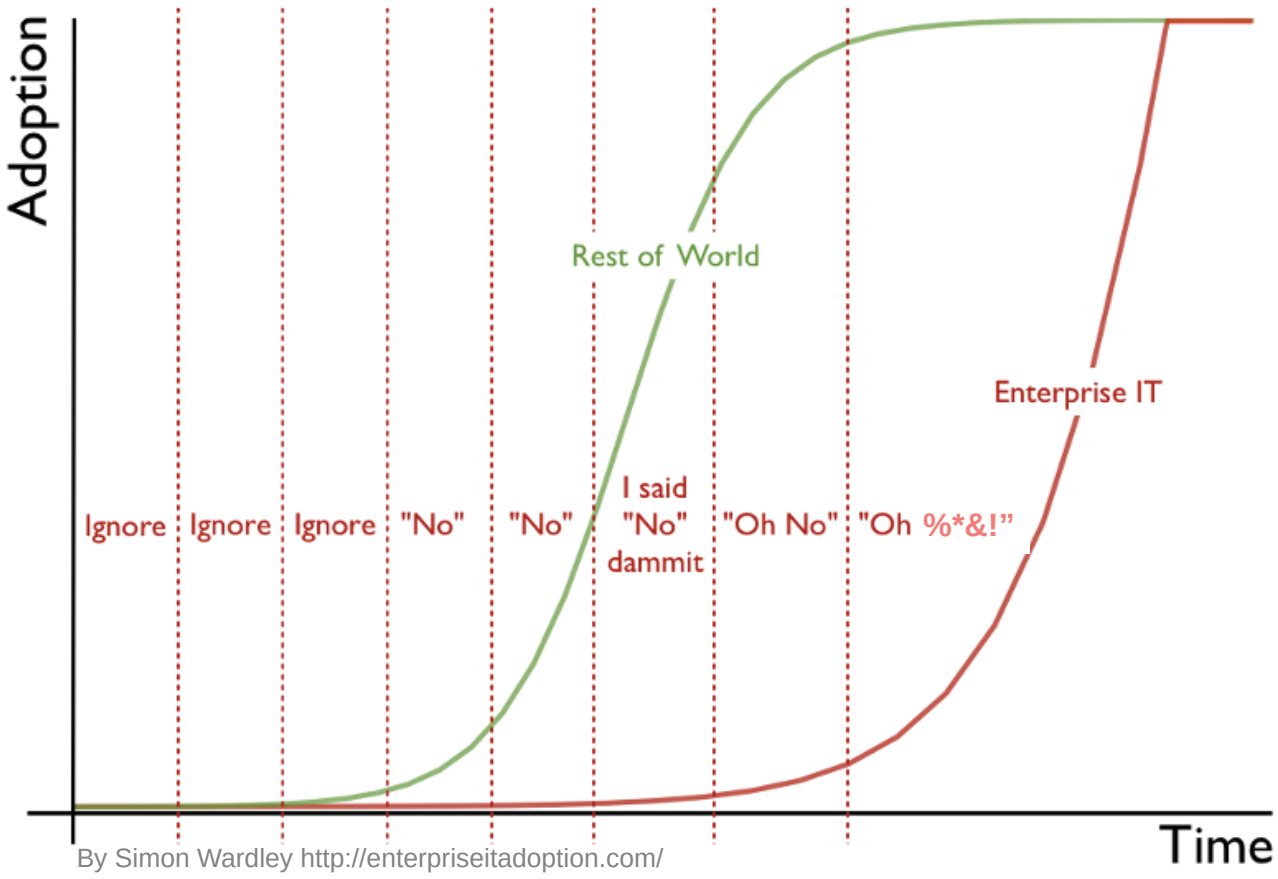
This talk:
How to use Cloud Native architecture to reduce cost without slowing down releases

@adrianco
BV
Battery Ventures



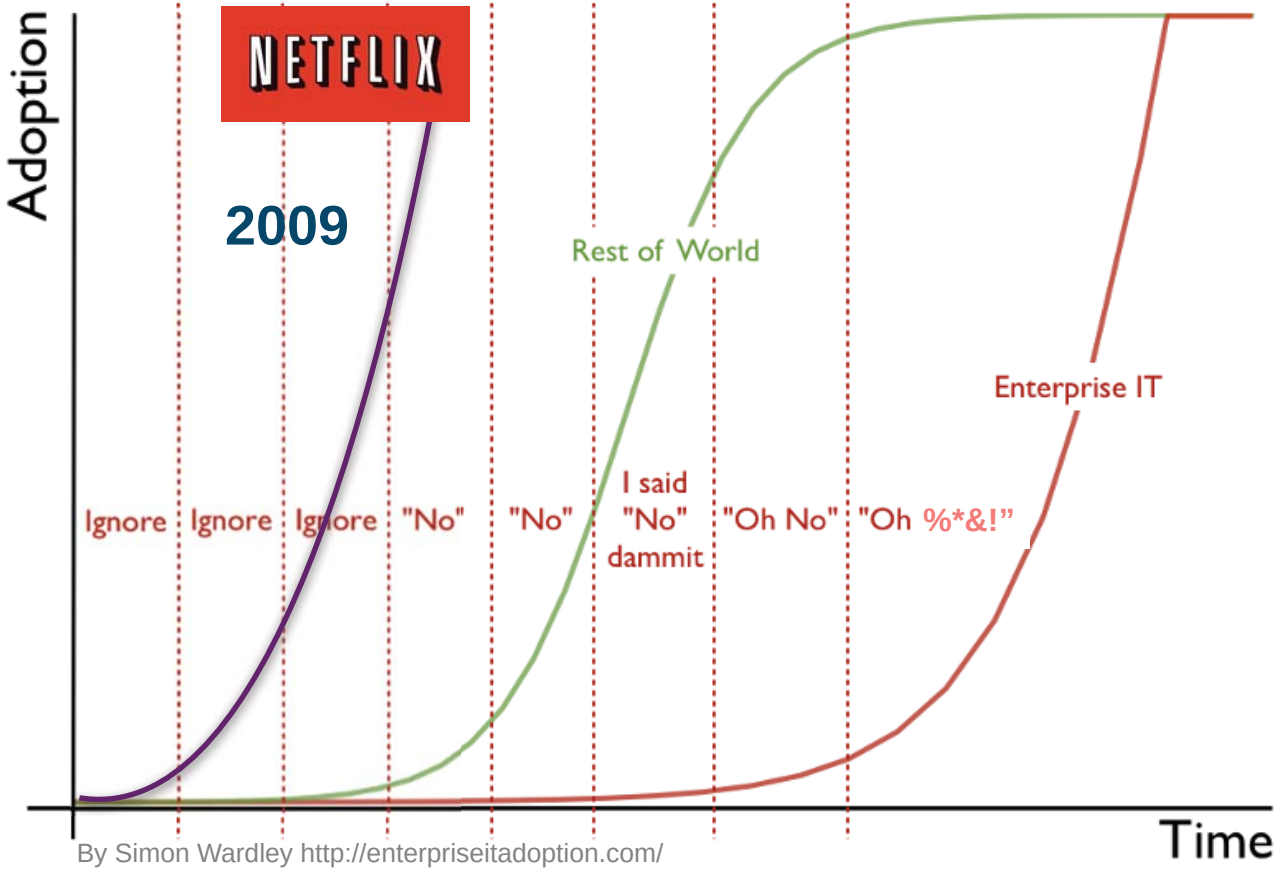
Speeding up Development Cloud Native Applications Cost Optimization

Why am I here?



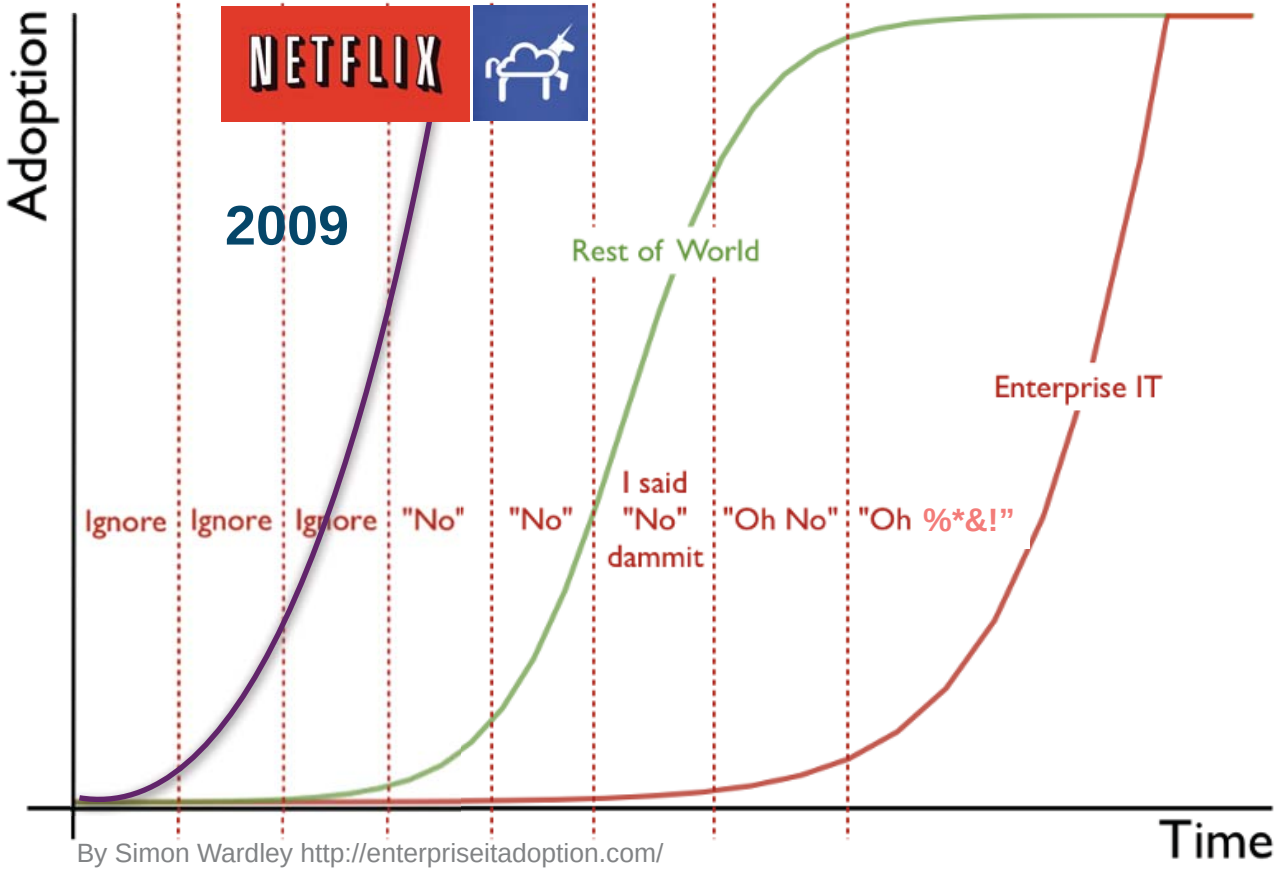
By Simon Wardley http://enterpriseitadoption.com/

Why am I here?



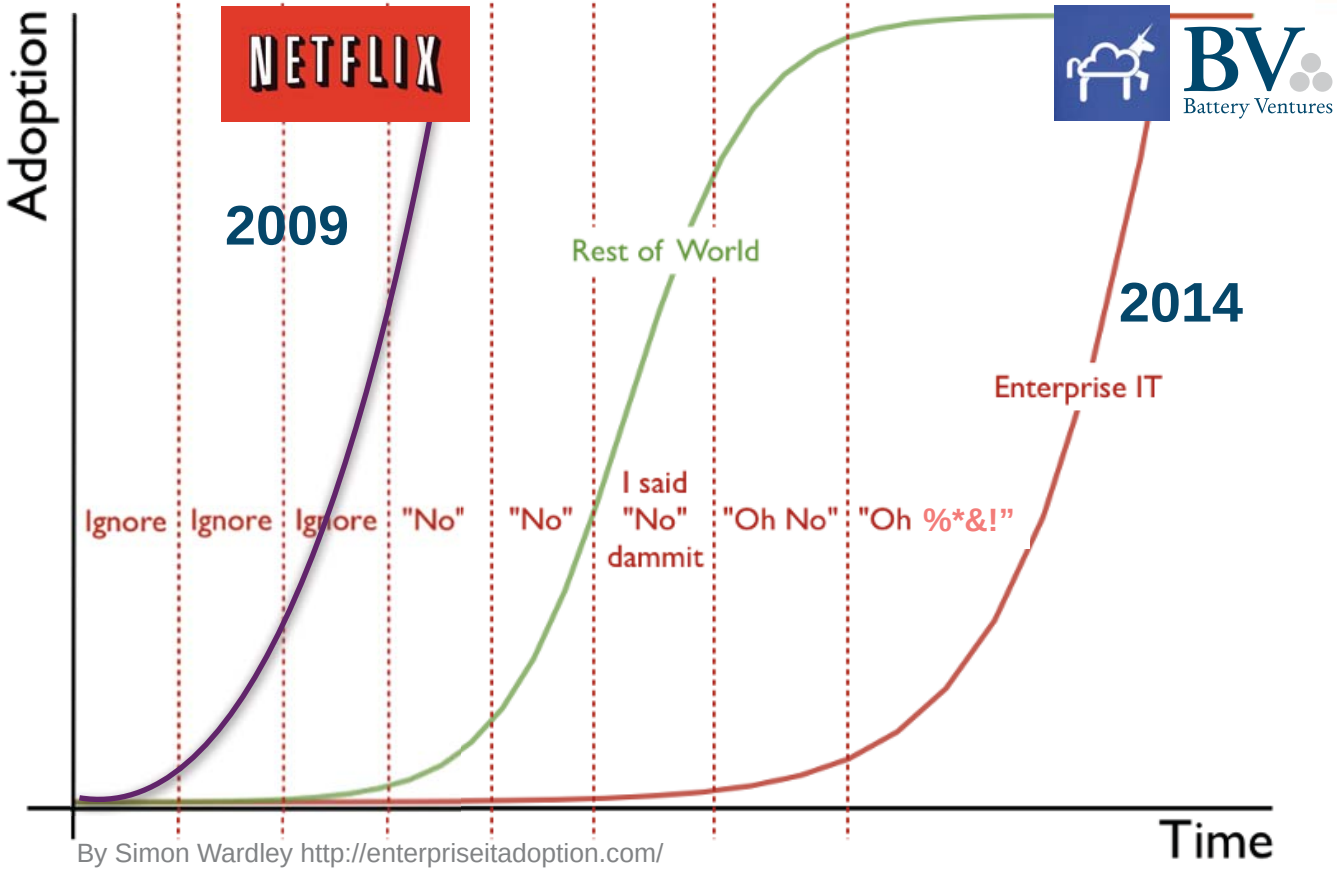
By Simon Wardley http://enterpriseitadoption.com/

Why am I here?



By Simon Wardley http://enterpriseitadoption.com/

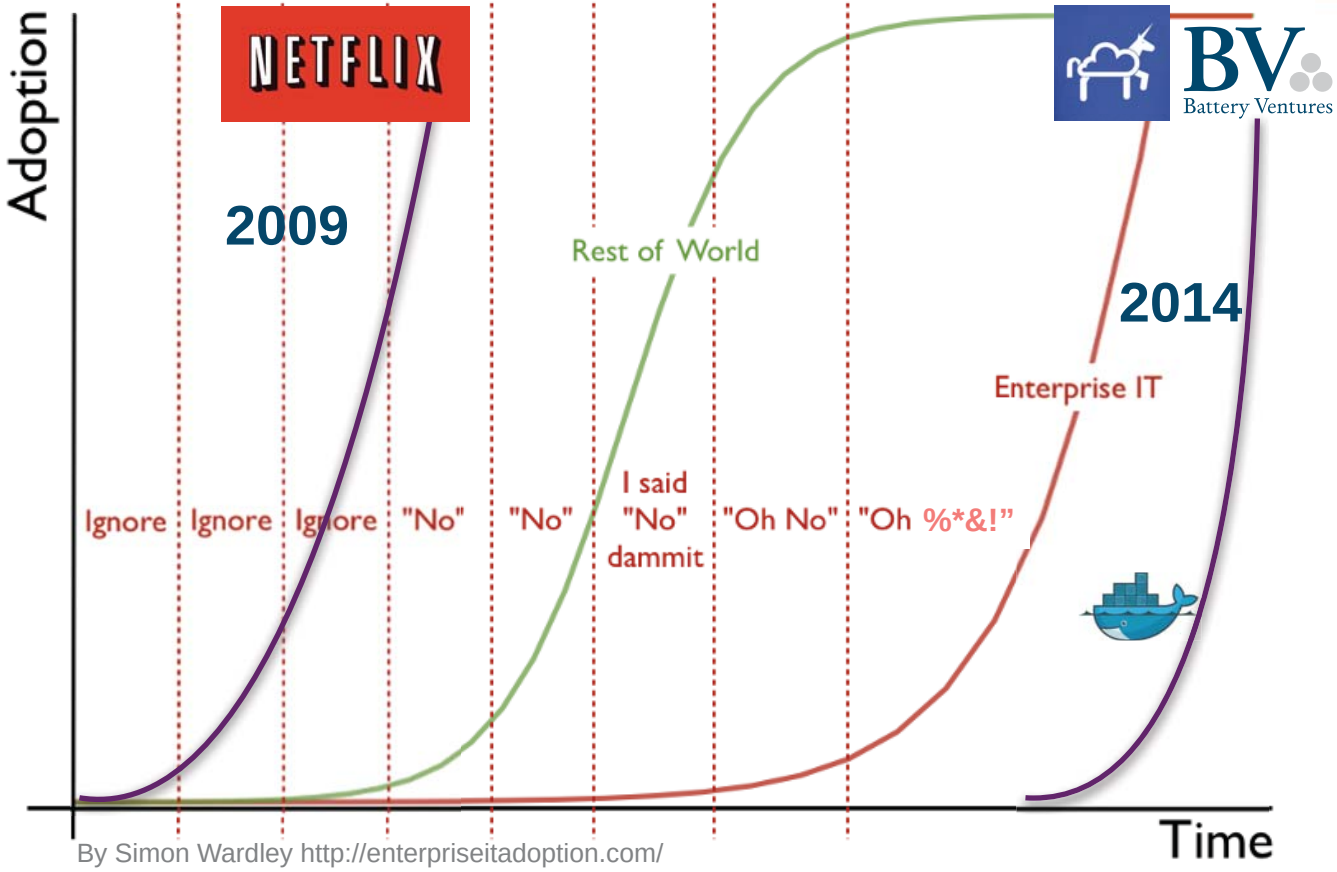
Why am I here?



BV Battery Ventures

@adrianco's job at the intersection of cloud and Enterprise IT, looking for disruption and opportunities.

Why am I here?



By Simon Wardley http://enterpriseitadoption.com/

@adrianco's job at the intersection of cloud and Enterprise IT, looking for disruption and opportunities.

Example: Docker wasn't on anyone's roadmap for 2014. It's on everyone's roadmap for 2015.

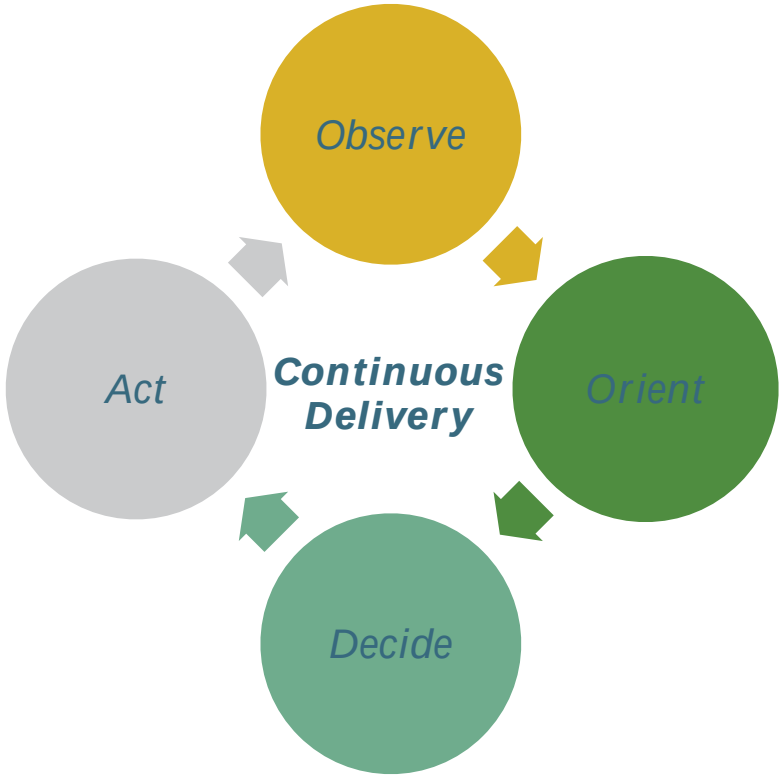
What does @adrianco do?

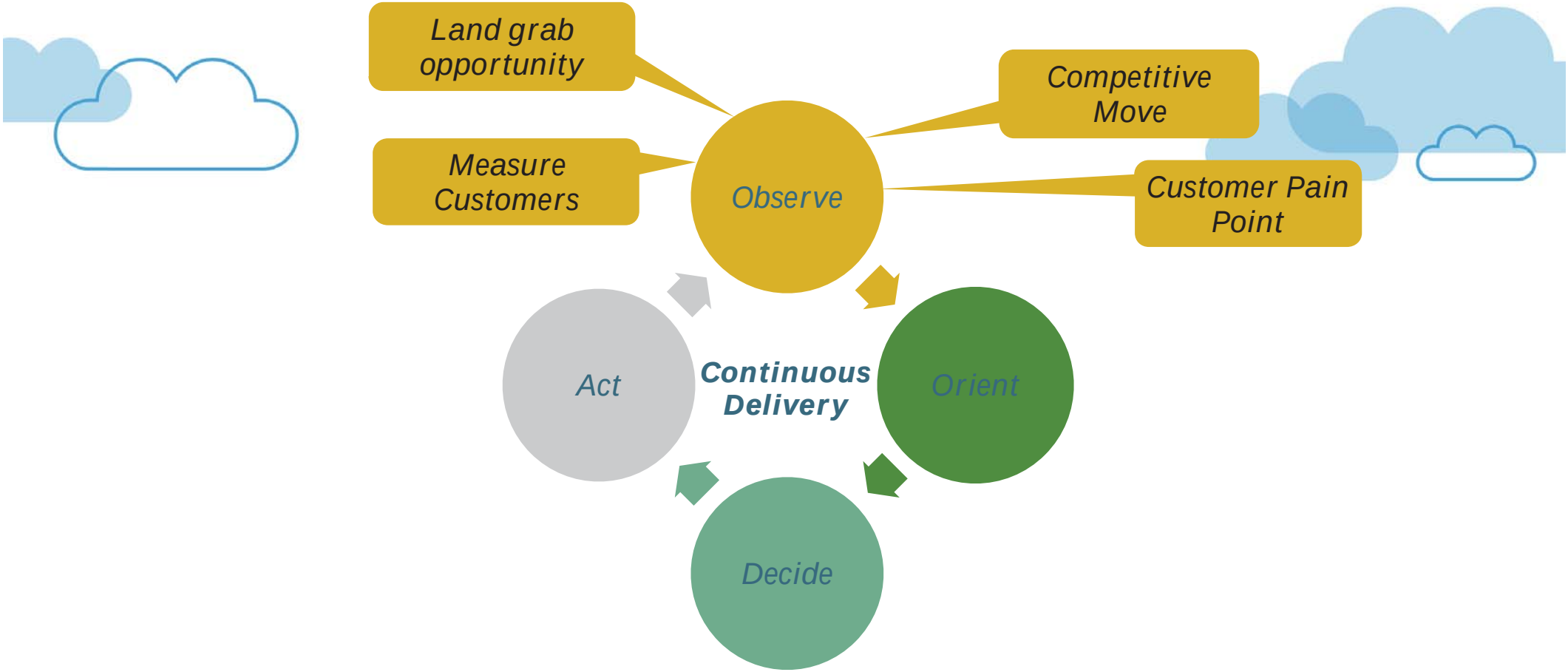
@adrianco

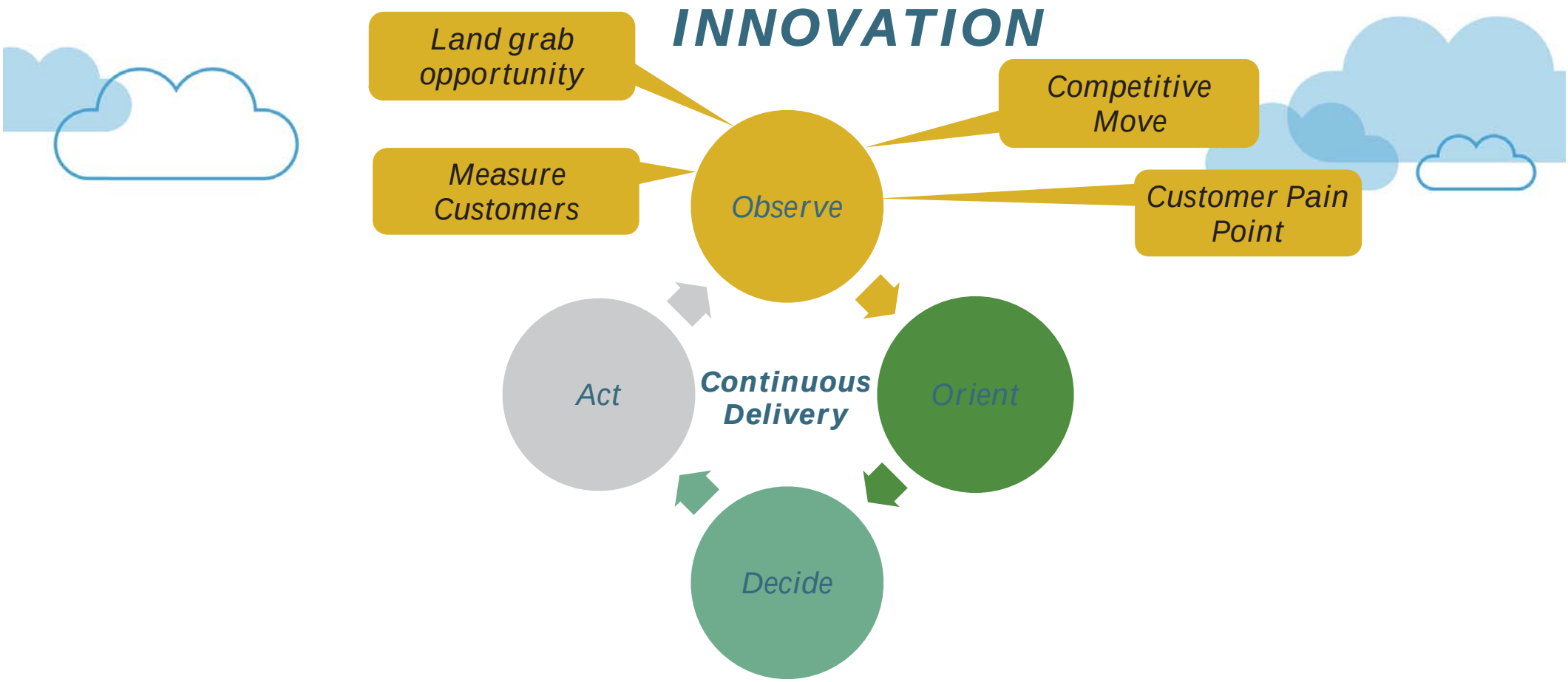
BV Battery Ventures

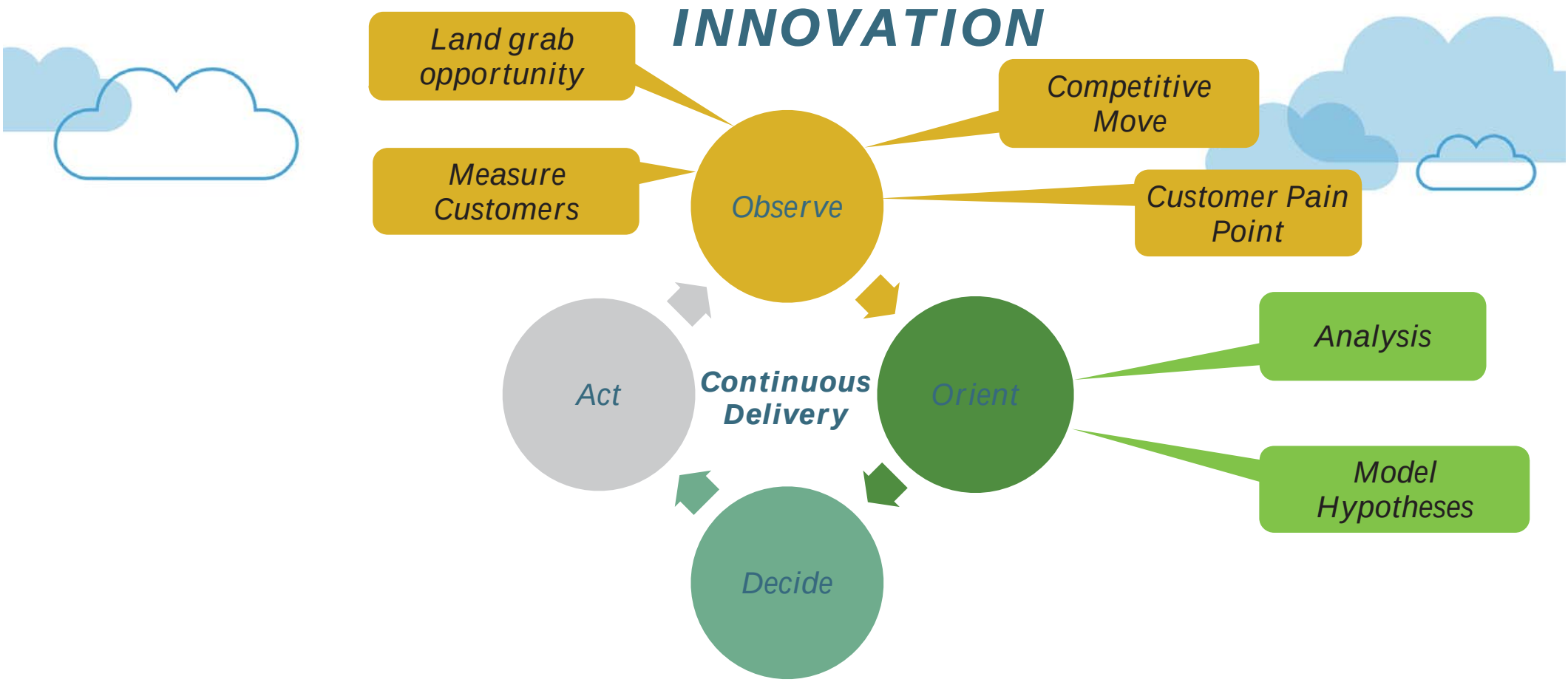
- Presentations at Conferences
- Maintain Relationship with Cloud Vendors
- Technology Due Diligence on Deals
- Presentations at Companies
- Technical Advice for Portfolio Companies
- Program Committee for Conferences
- Tinkering with Technologies
- Networking with Interesting People

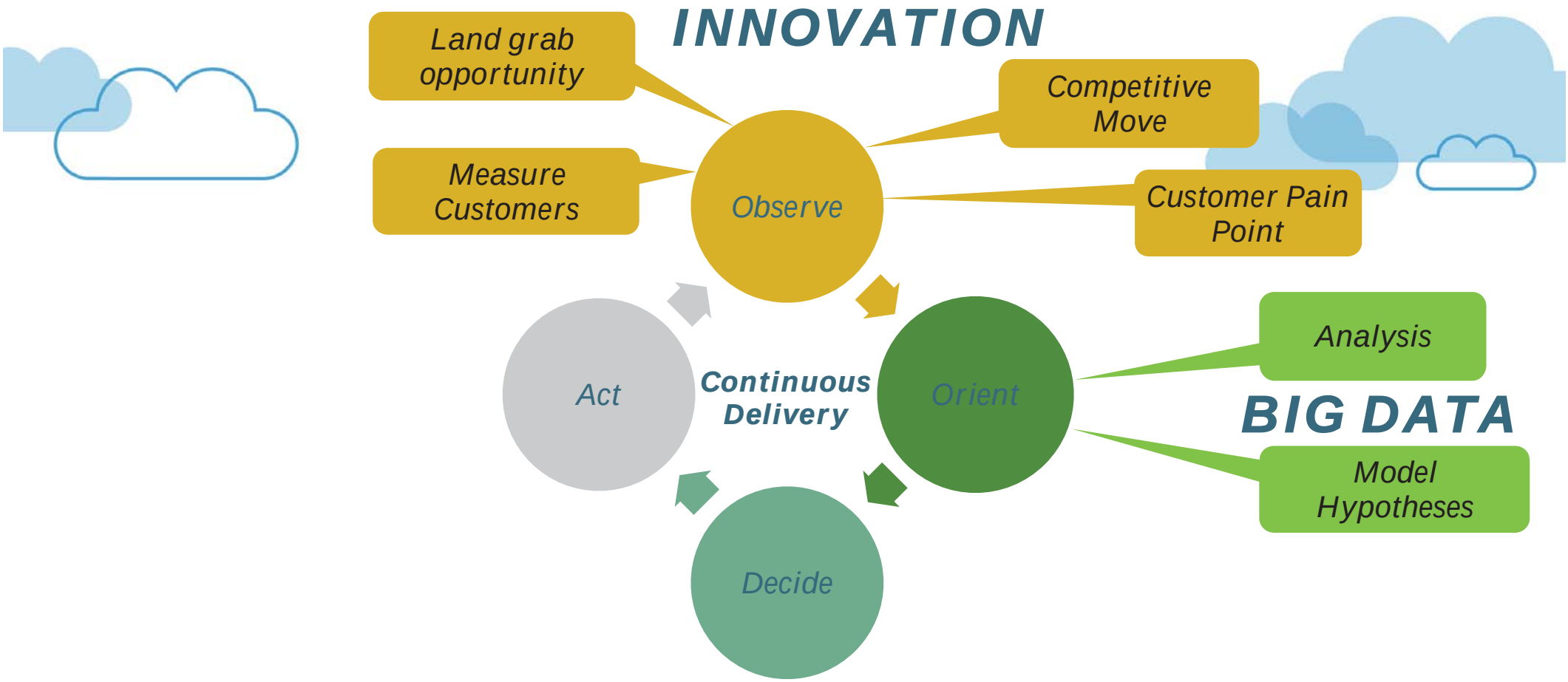
Speeding Up Development

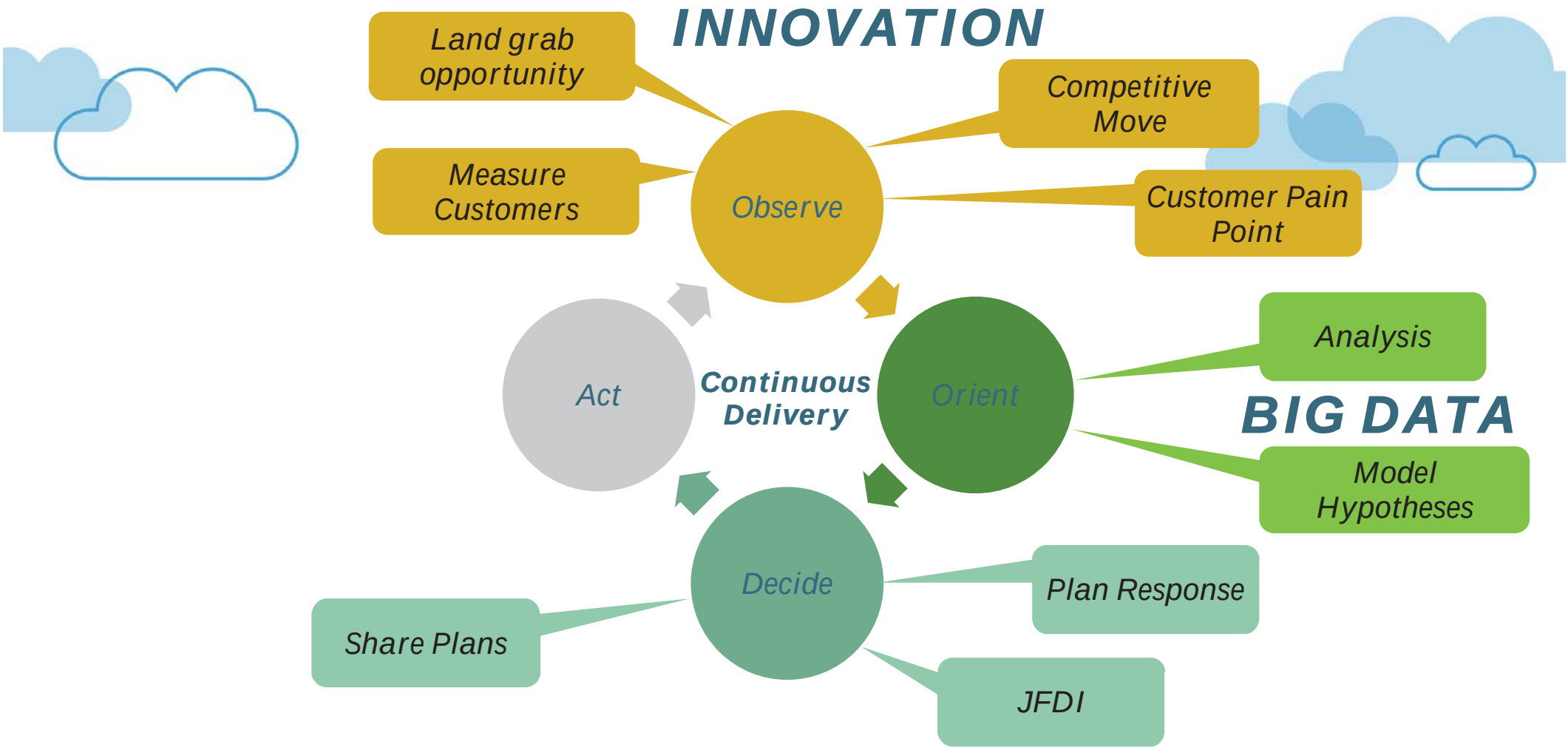

Observe
Orient
Decide
Act
Continuous Delivery

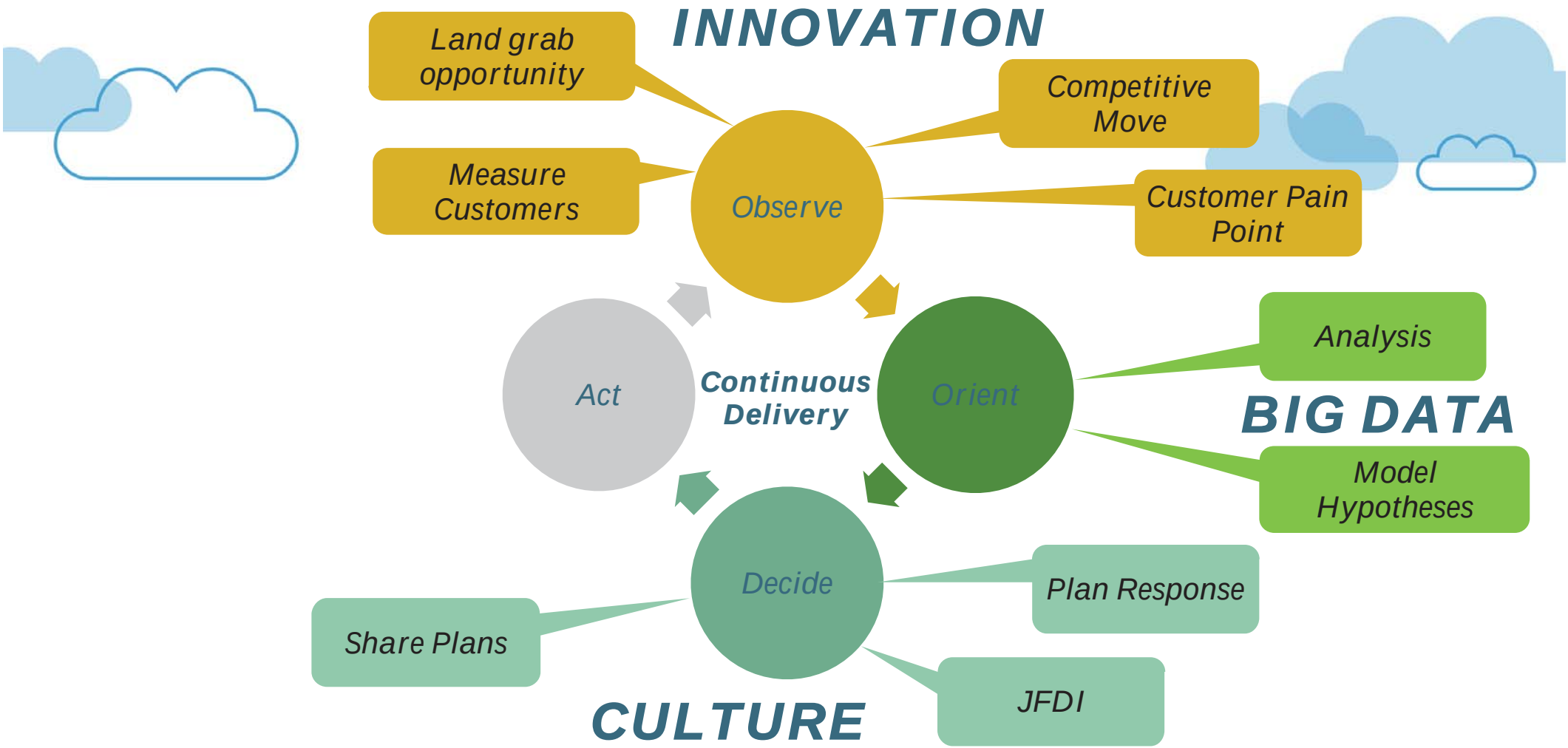

Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Act
Continuous Delivery
Orient
Decide

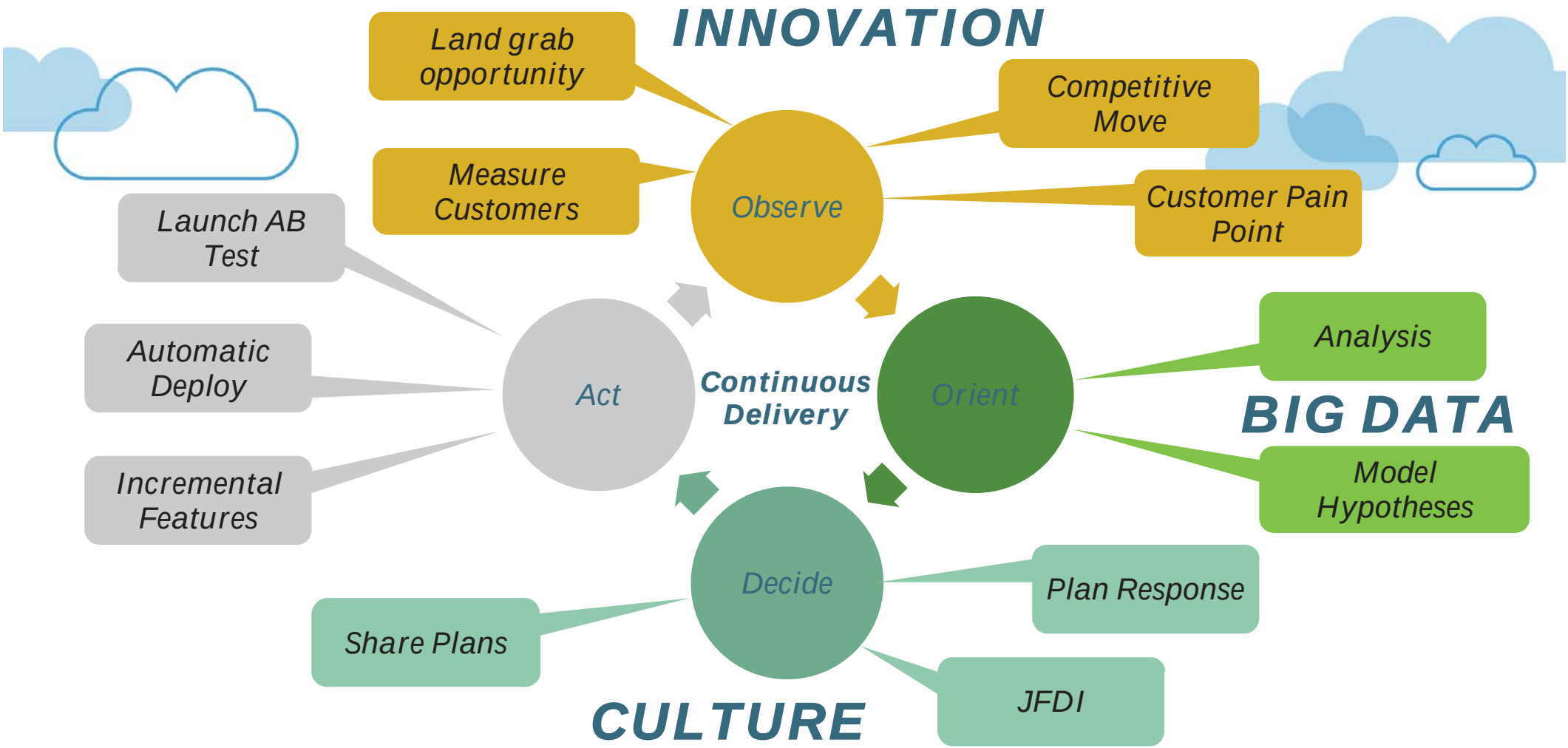

INNOVATION
Land grab opportunity
Competitive Move
Measure Customers
Customer Pain Point
Observe
Orient
Decide
Act
Continuous Delivery

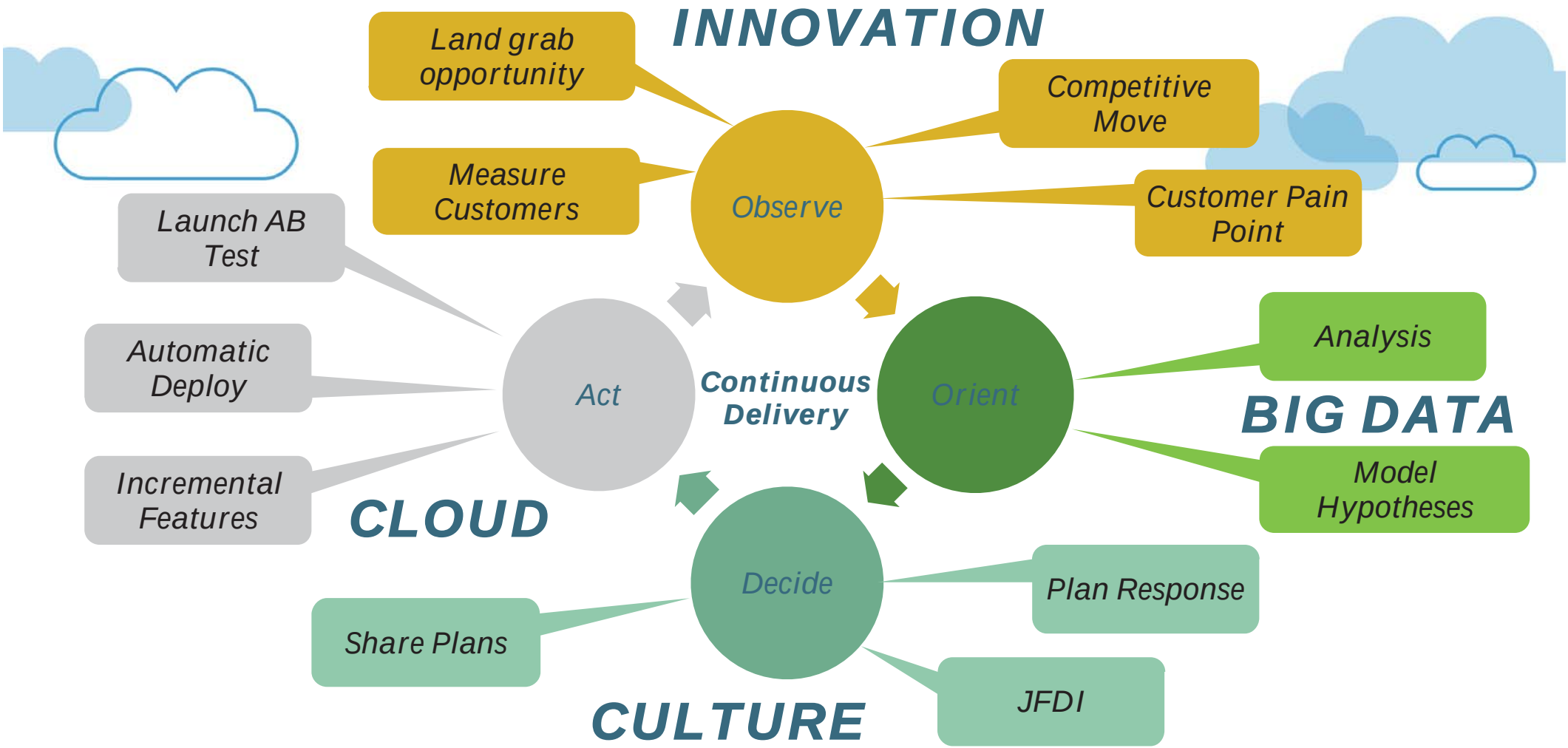

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Act
Continuous Delivery
Orient
Analysis
Model Hypotheses
Decide

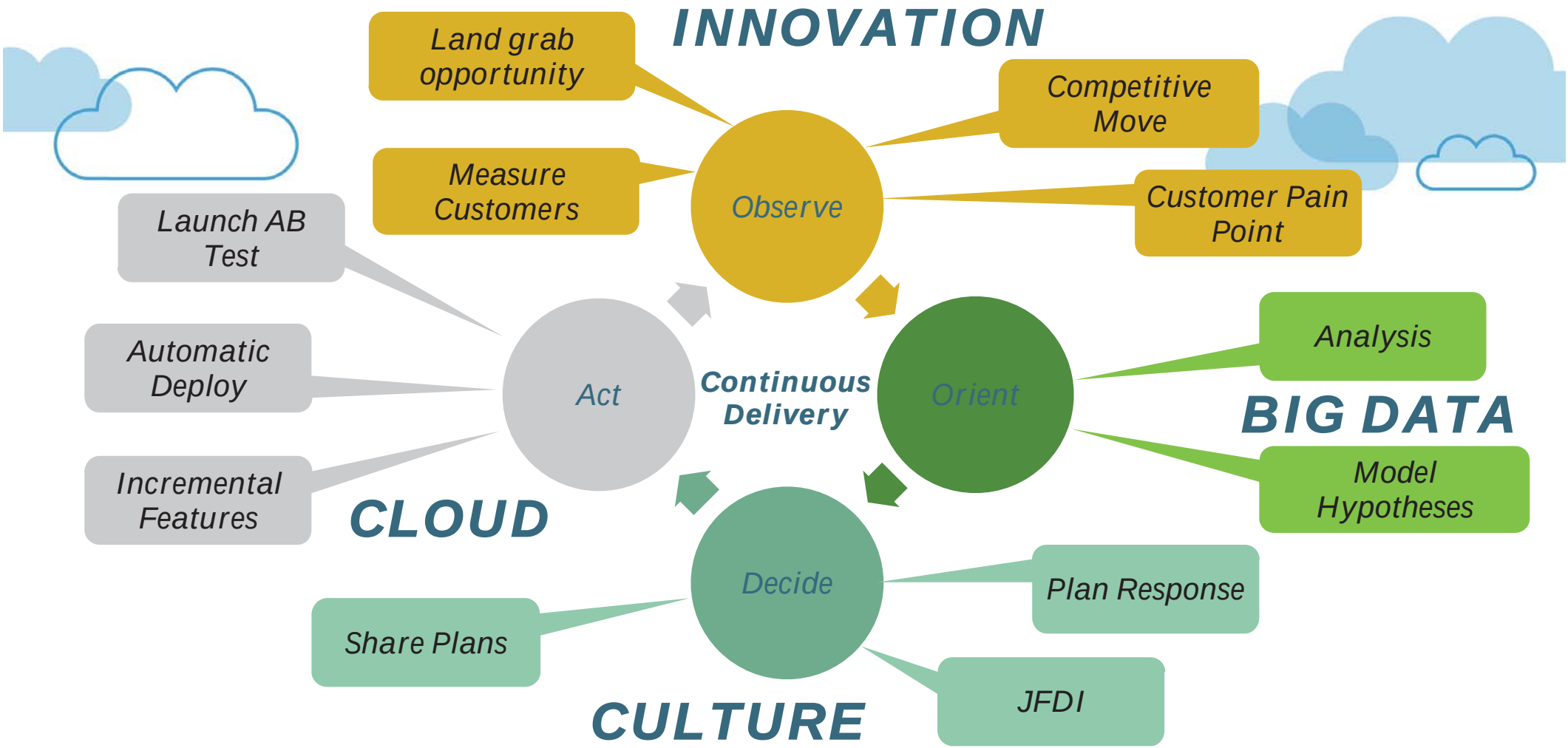

INNOVATION
Land grab opportunity
Measure Customers
Competitive Move
Customer Pain Point
Observe
Orient
Decide
Act
Continuous Delivery
Analysis
BIG DATA
Model Hypotheses

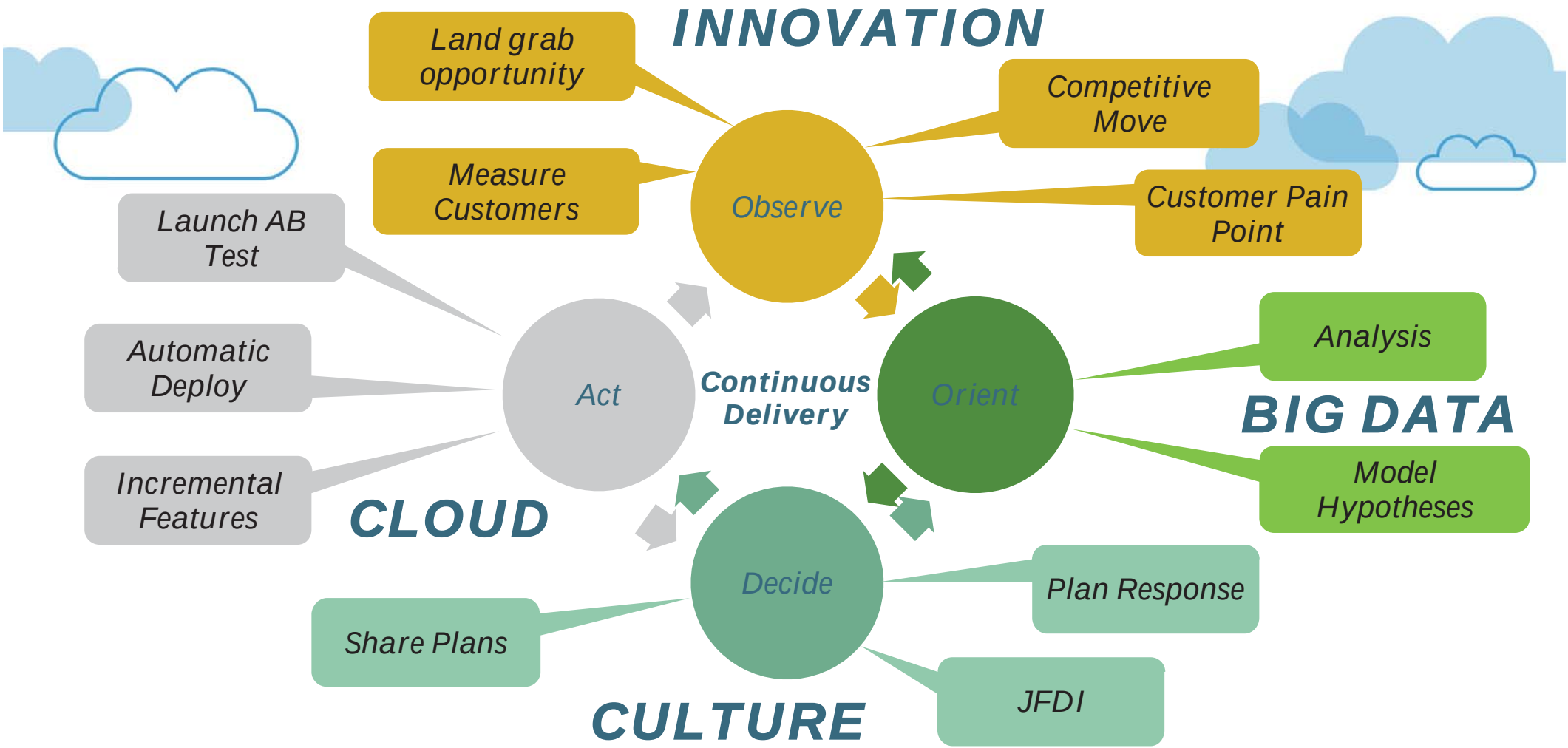

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Act
Continuous Delivery
Orient
Analysis
BIG DATA
Model Hypotheses
Decide
Plan Response
Share Plans
JFDI

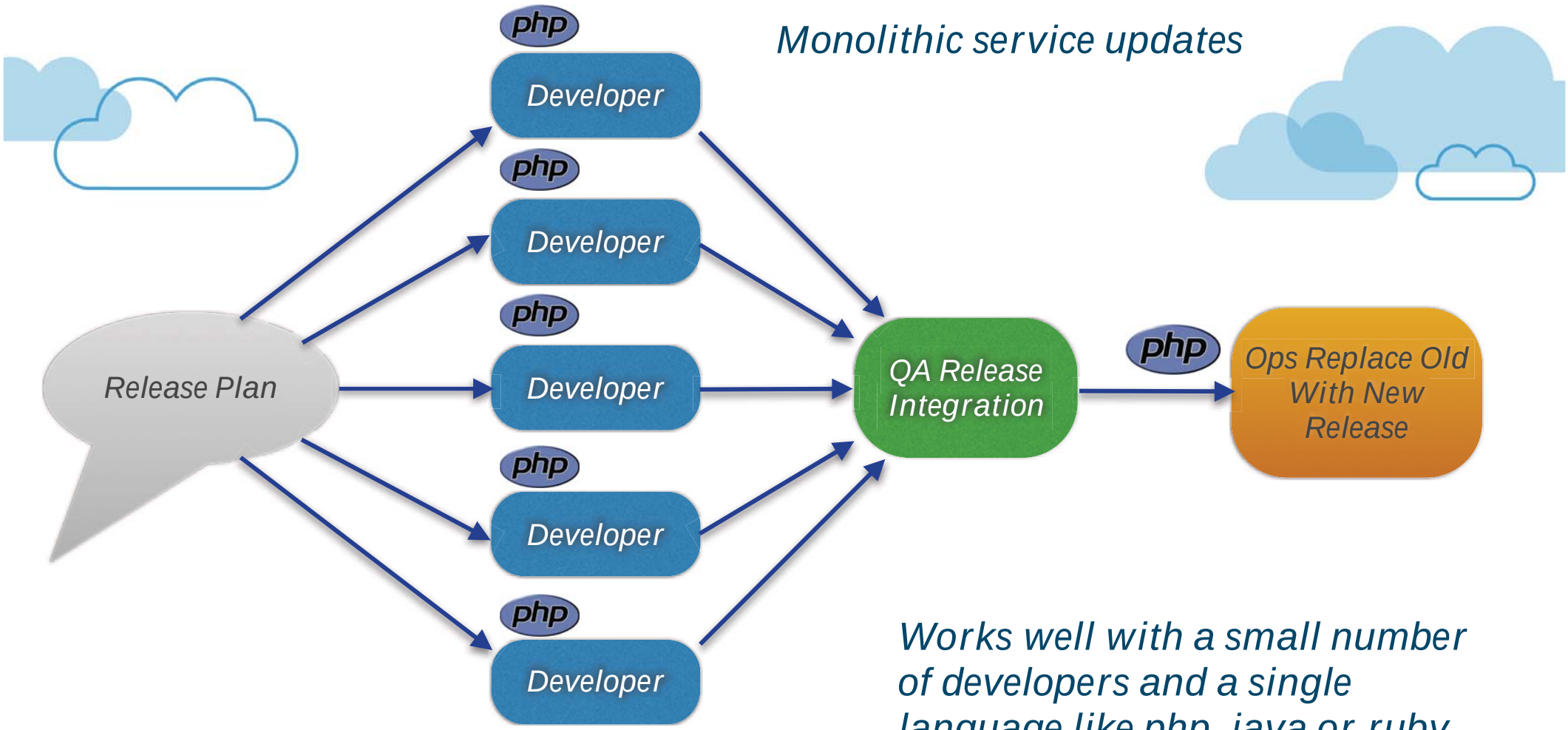

INNOVATION
Land grab opportunity
Measure Customers
Competitive Move
Customer Pain Point
Observe
Orient
Decide
Act
Continuous Delivery
Analysis
BIG DATA
Model Hypotheses
Plan Response
JFDI
Share Plans
CULTURE


INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Launch AB Test
Automatic Deploy
Incremental Features
Act
Continuous Delivery
Orient
Analysis
BIG DATA
Model Hypotheses
Decide
Plan Response
Share Plans
JFDI
CULTURE

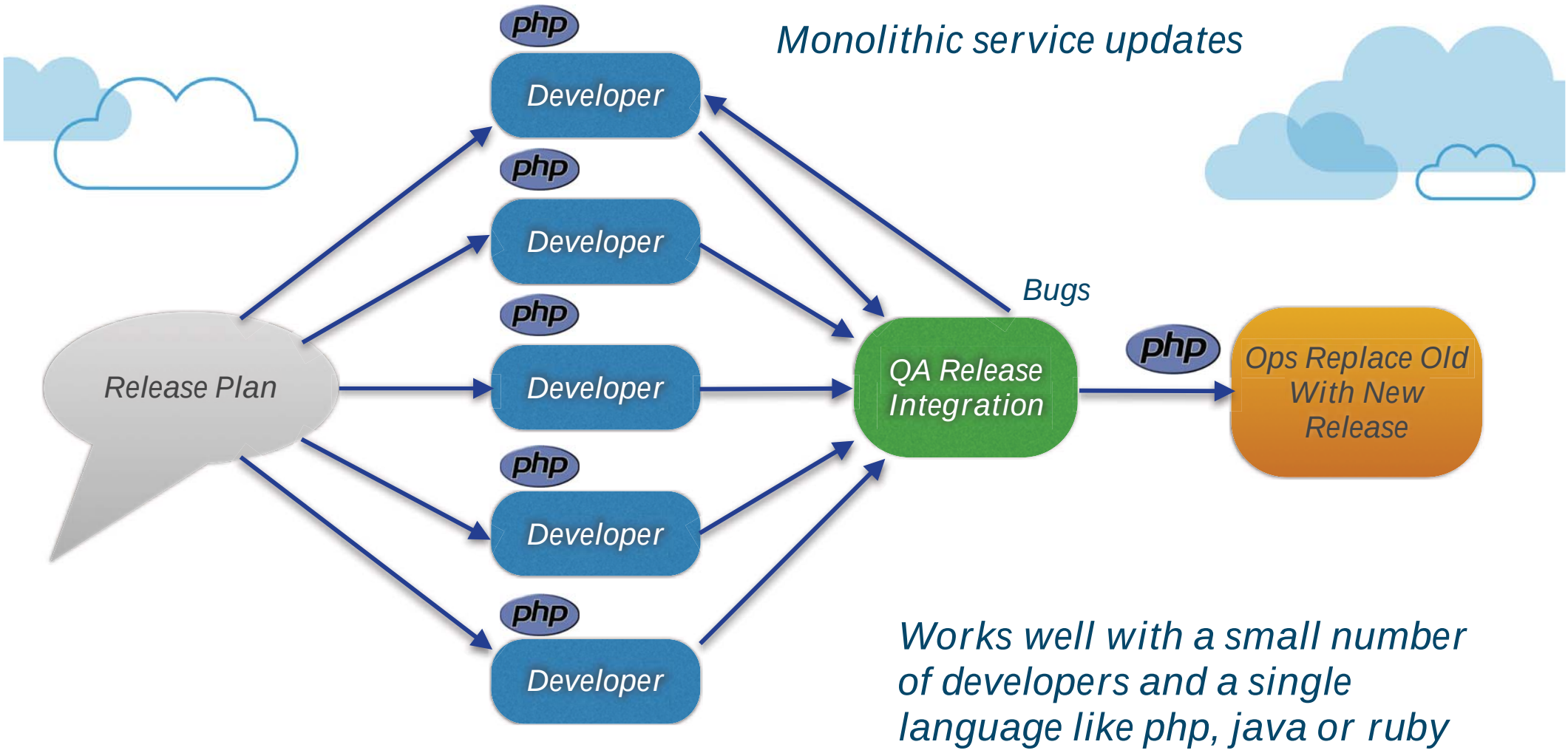

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Launch AB Test
Automatic Deploy
Incremental Features
Act
CLOUD
Continuous Delivery
Orient
Analysis
BIG DATA
Model Hypotheses
Decide
Share Plans
Plan Response
JFDI
CULTURE

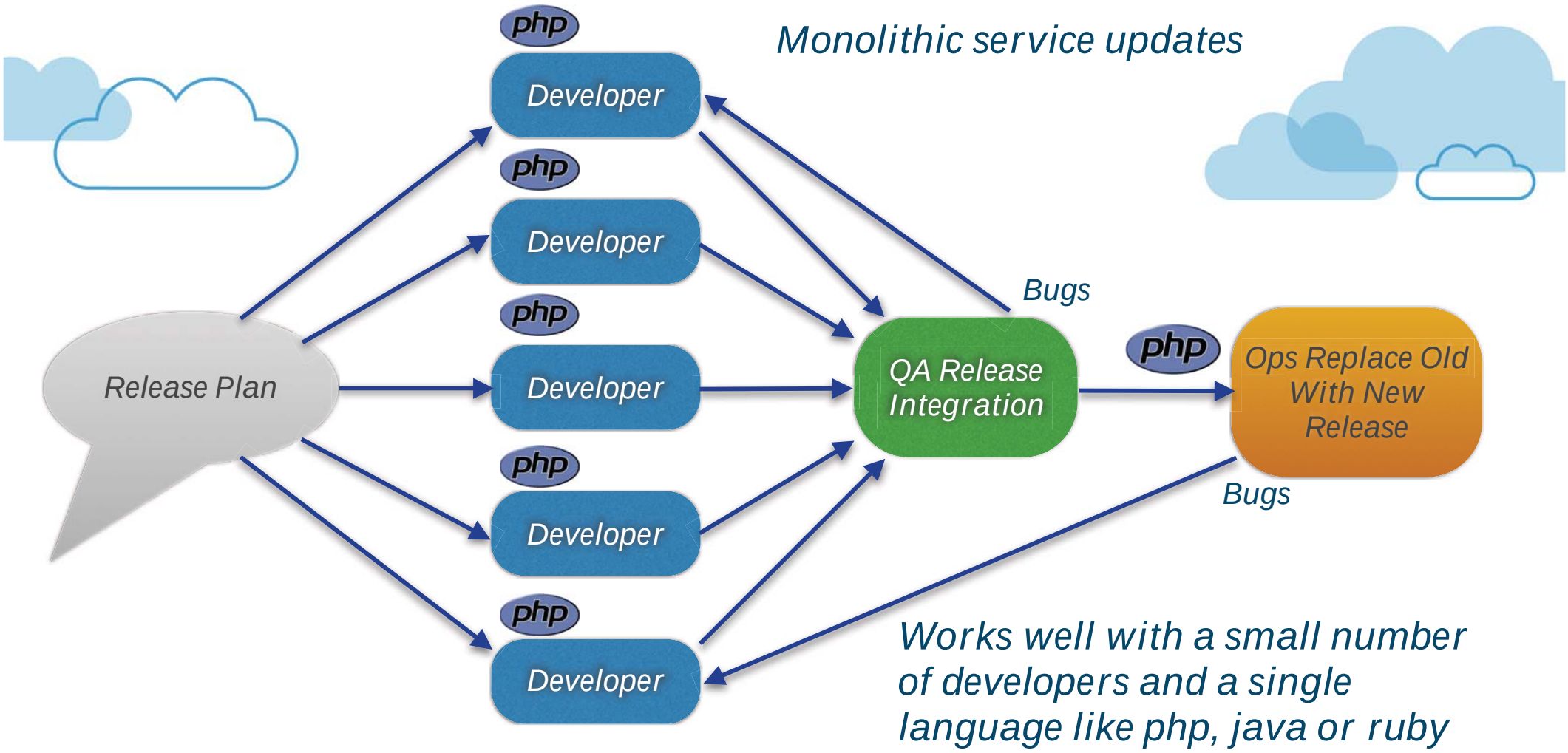

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Launch AB Test
Automatic Deploy
Incremental Features
Act
CLOUD
Continuous Delivery
Orient
Analysis
BIG DATA
Model Hypotheses
Decide
Share Plans
Plan Response
JFDI
CULTURE


INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Launch AB Test
Automatic Deploy
Incremental Features
Act
CLOUD
Continuous Delivery
Orient
Analysis
BIG DATA
Model Hypotheses
Decide
Share Plans
Plan Response
JFDI
CULTURE



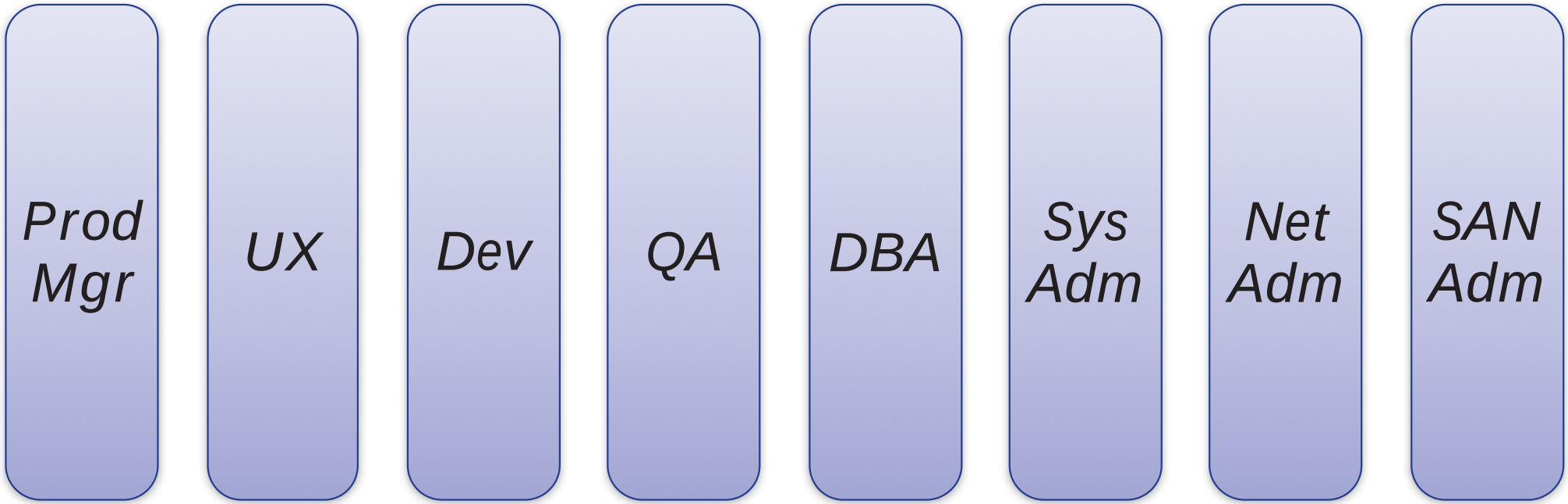
Works well with a small number of developers and a single language like php, java or ruby


Monolithic service updates
Release Plan
Developer
Developer
Developer
Developer
Developer
QA Release Integration
Bugs
Ops Replace Old With New Release
Works well with a small number of developers and a single language like php, java or ruby


Monolithic service updates
Release Plan
Developer
Developer
Developer
Developer
Developer
QA Release Integration
Bugs
Ops Replace Old With New Release
Bugs
Works well with a small number of developers and a single language like php, java or ruby

Breaking Down the SILOs

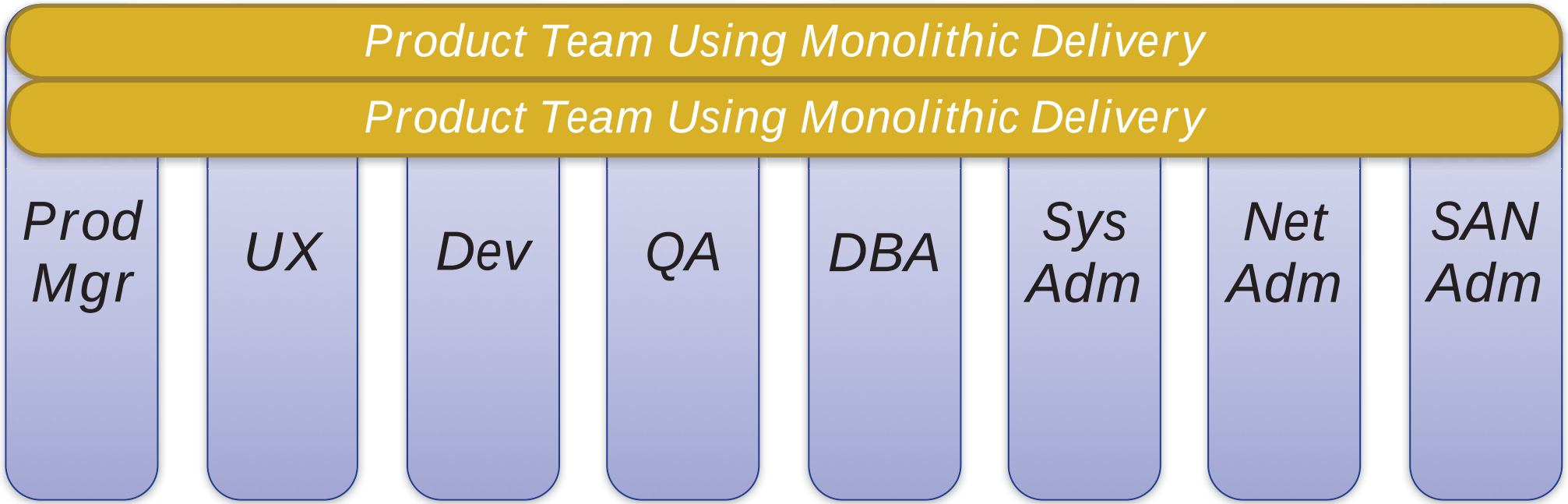
Breaking Down the SILOs



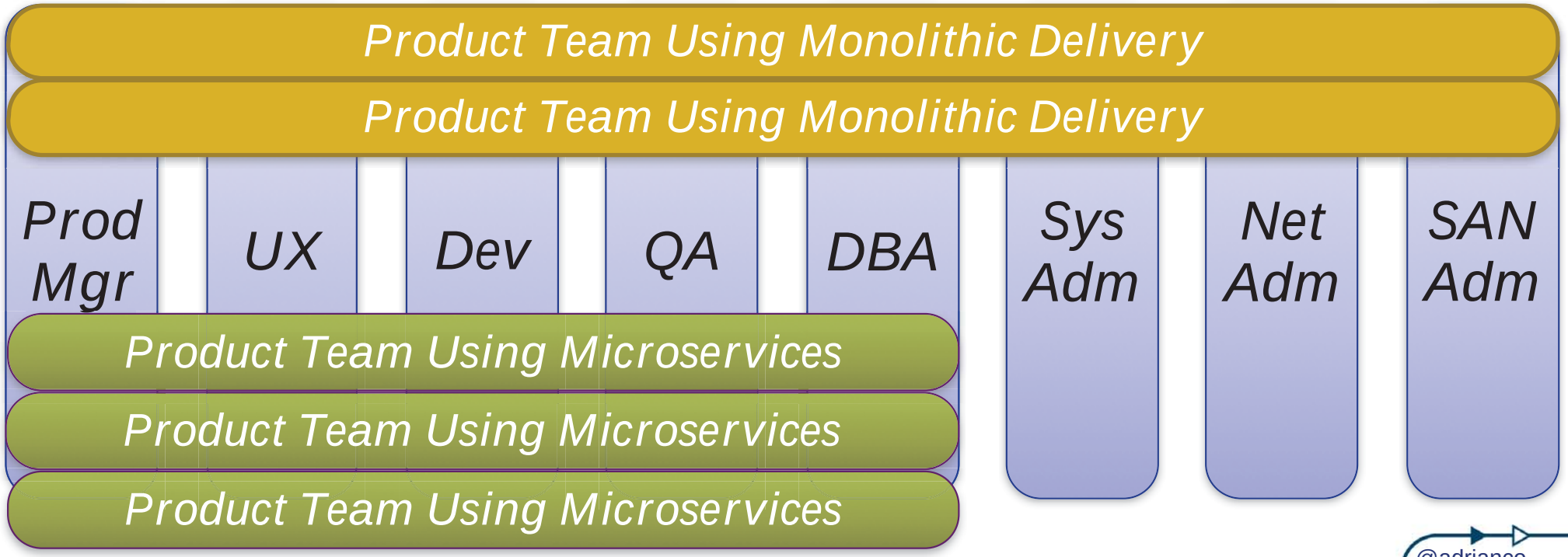
@adrianco

BV
Battery Ventures

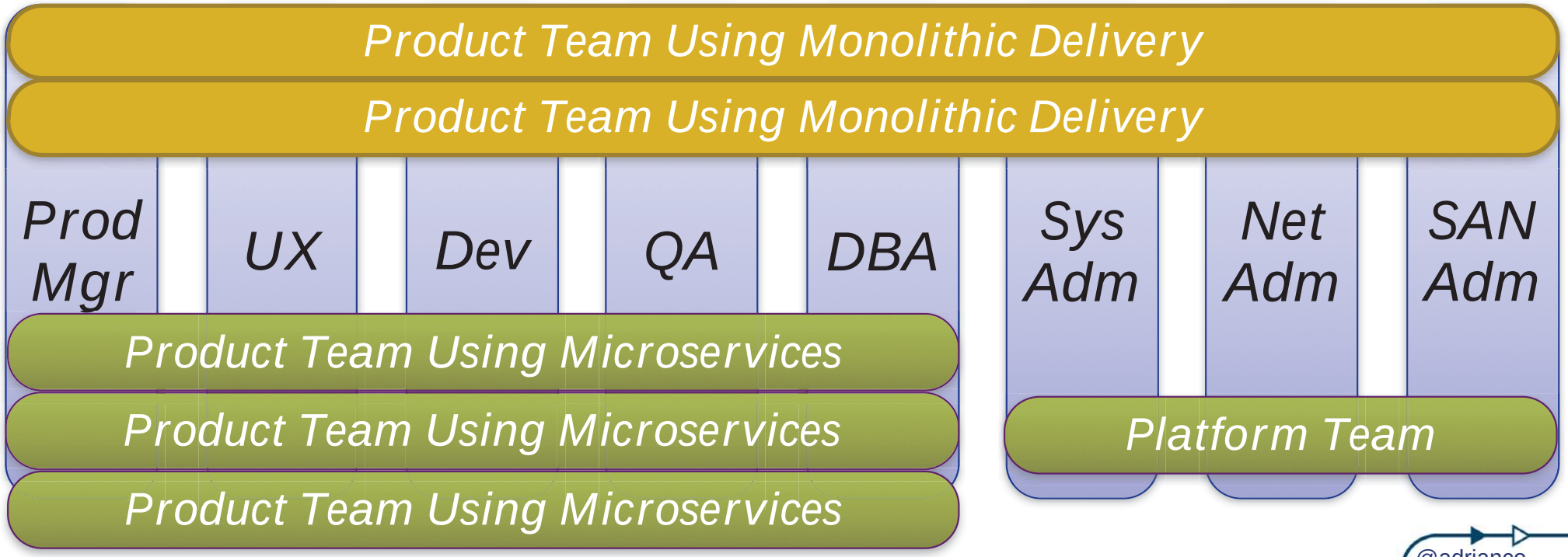
Breaking Down the SILOs



Breaking Down the SILOs



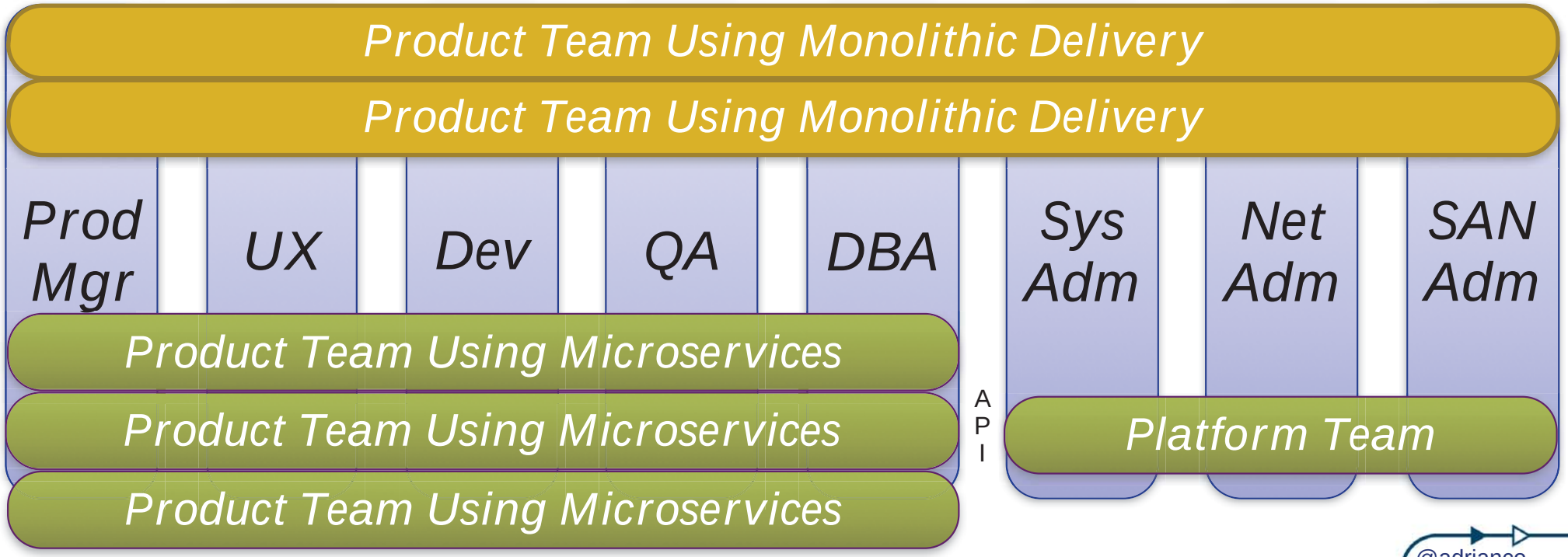
Breaking Down the SILOs



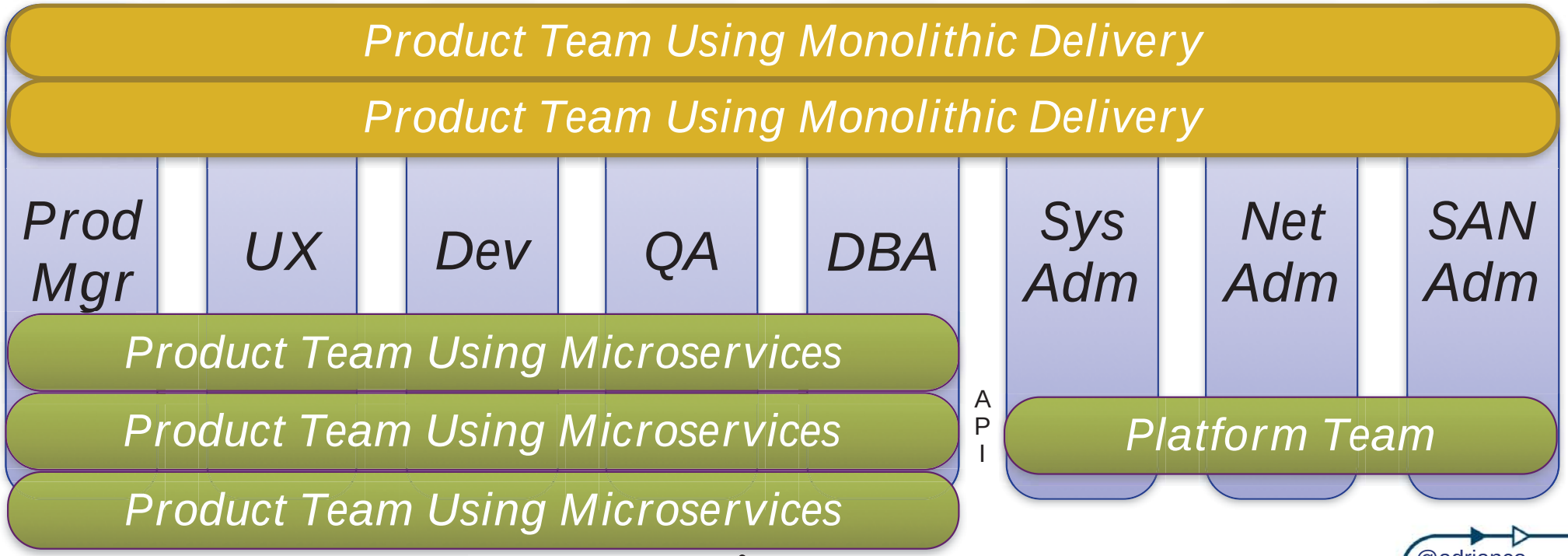
@adrianco

BV Battery Ventures

Breaking Down the SILOs



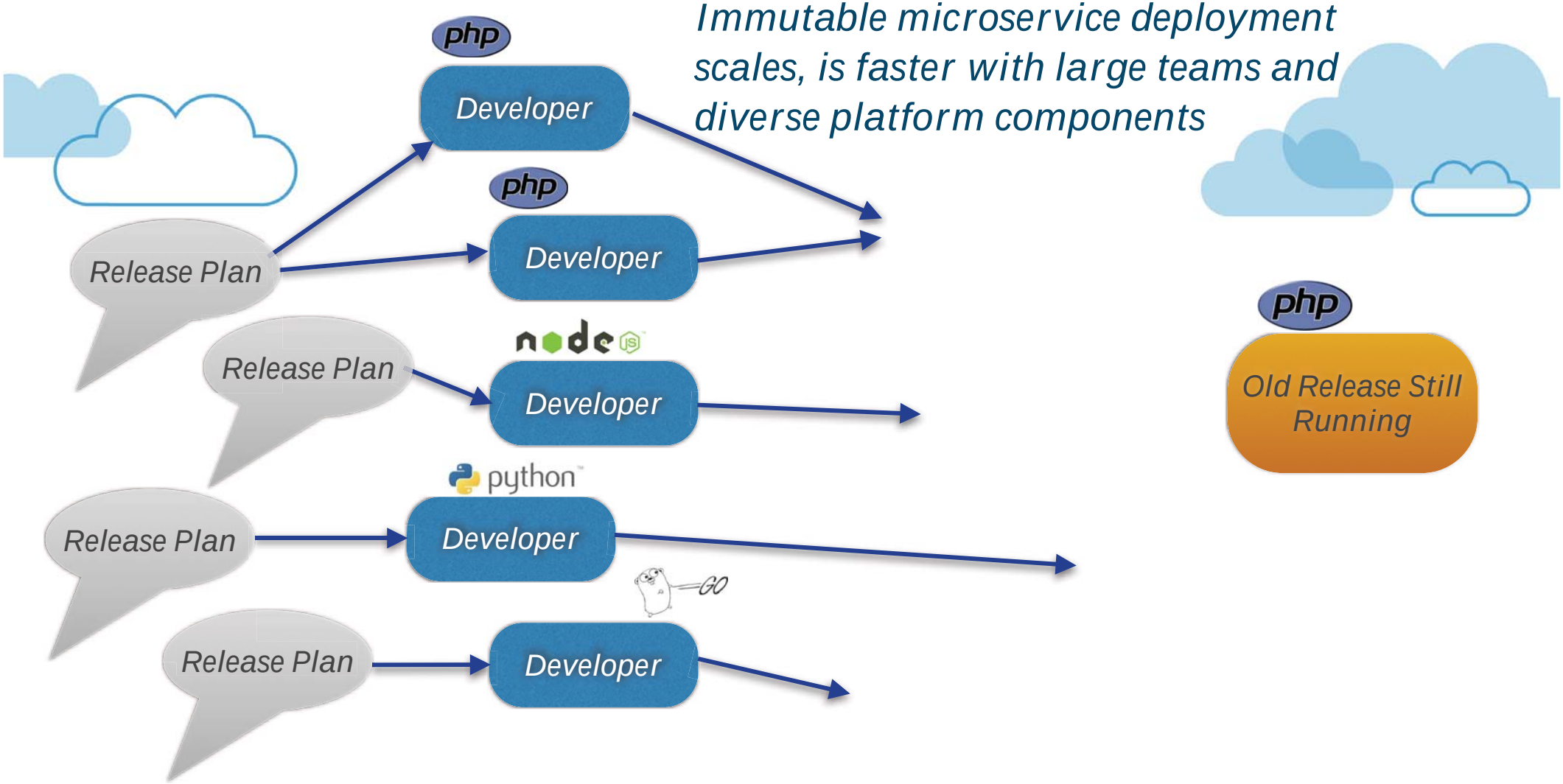
Breaking Down the SILOs



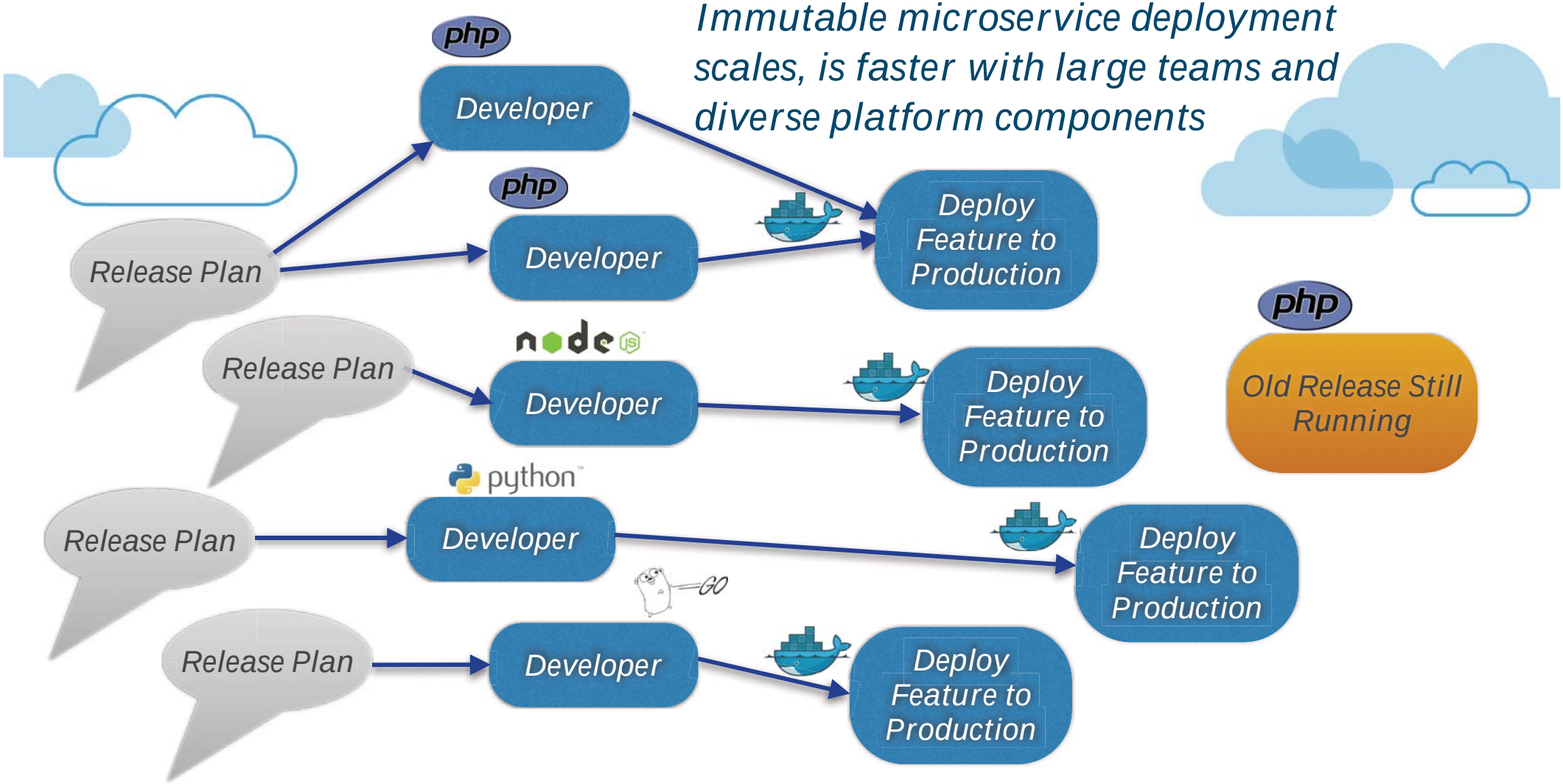
DevOps is a Re-Org!

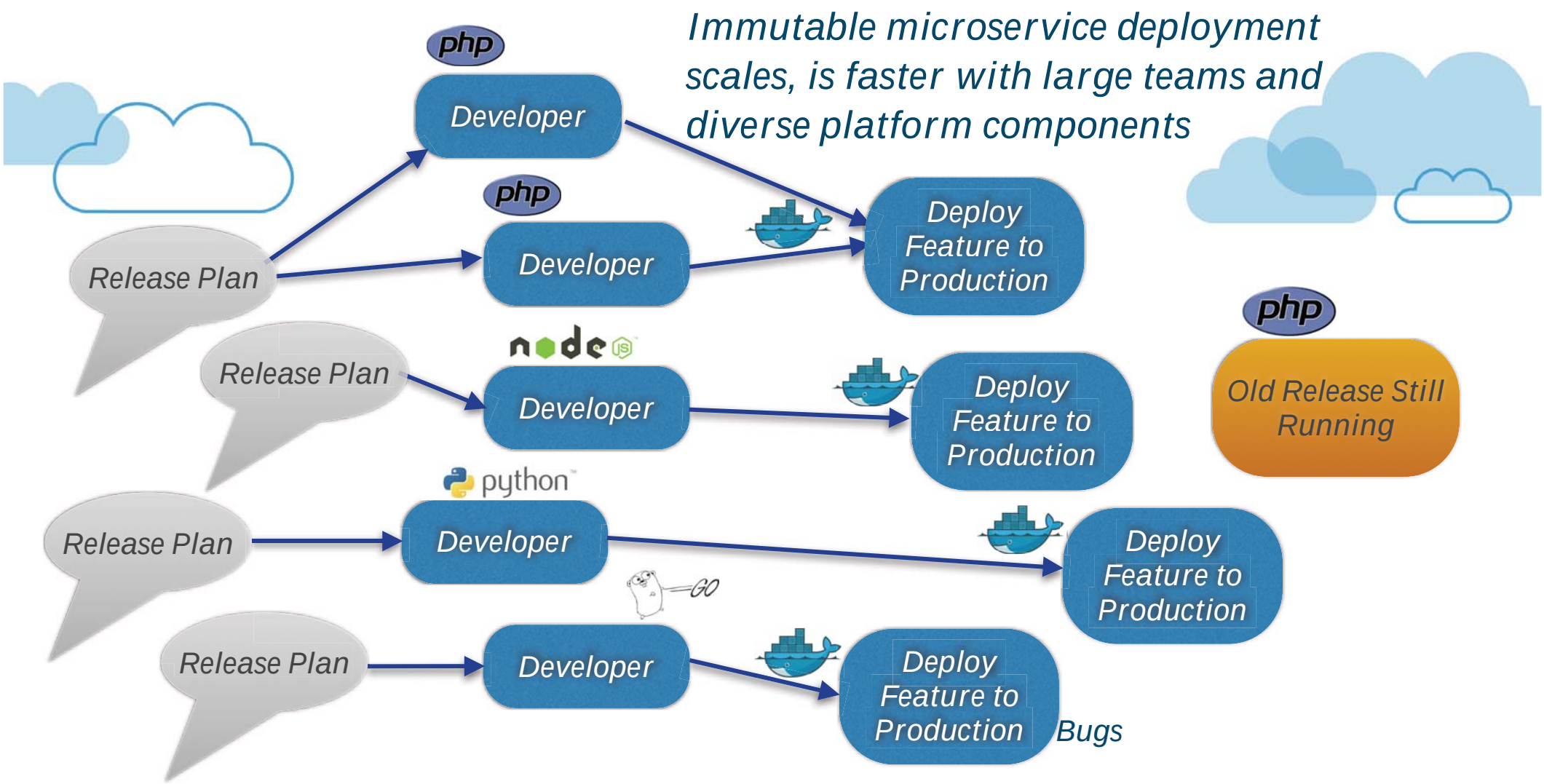
@adrianco
BV
Battery Ventures

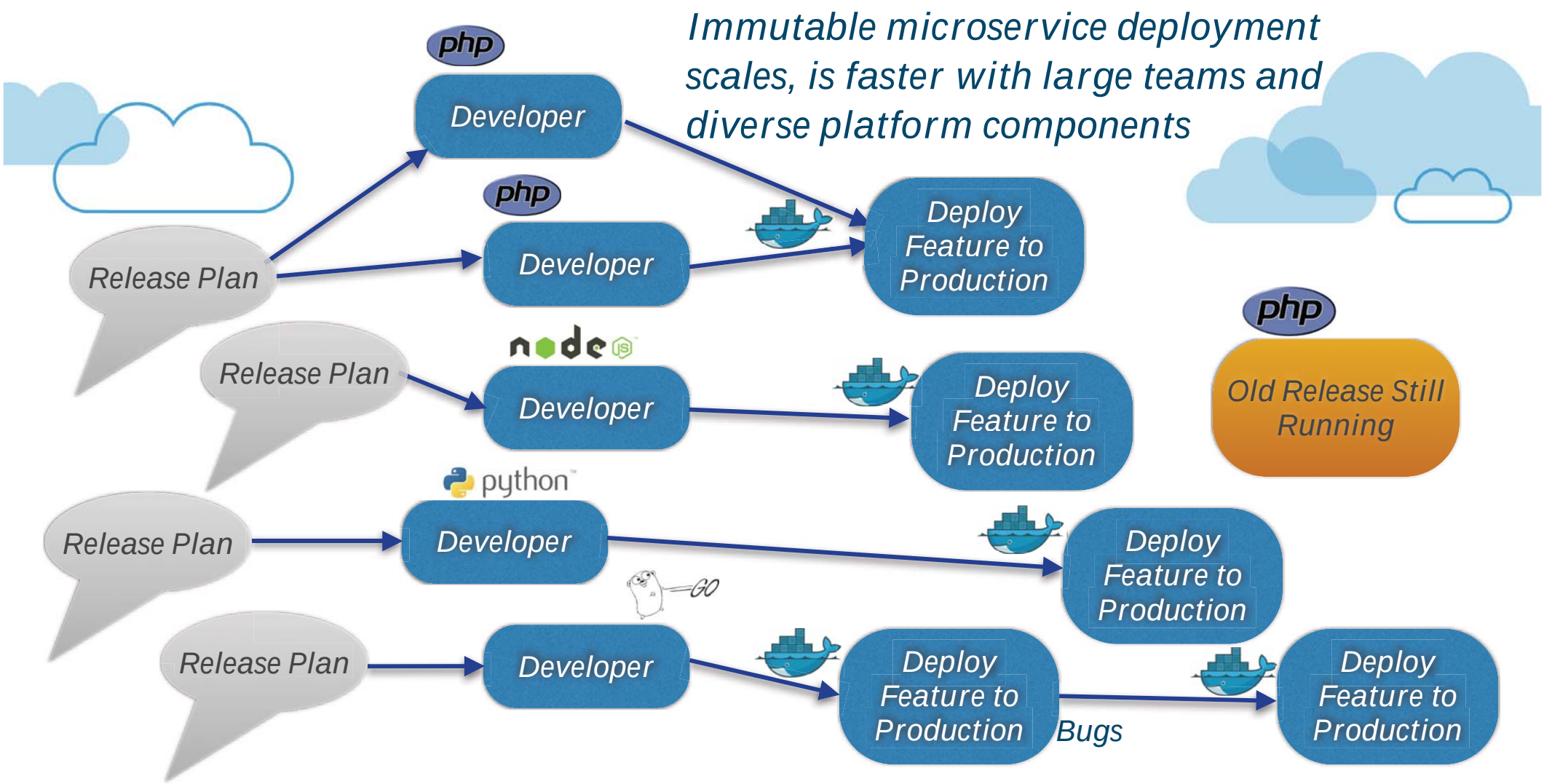
Immutable microservice deployment scales, is faster with large teams and diverse platform components



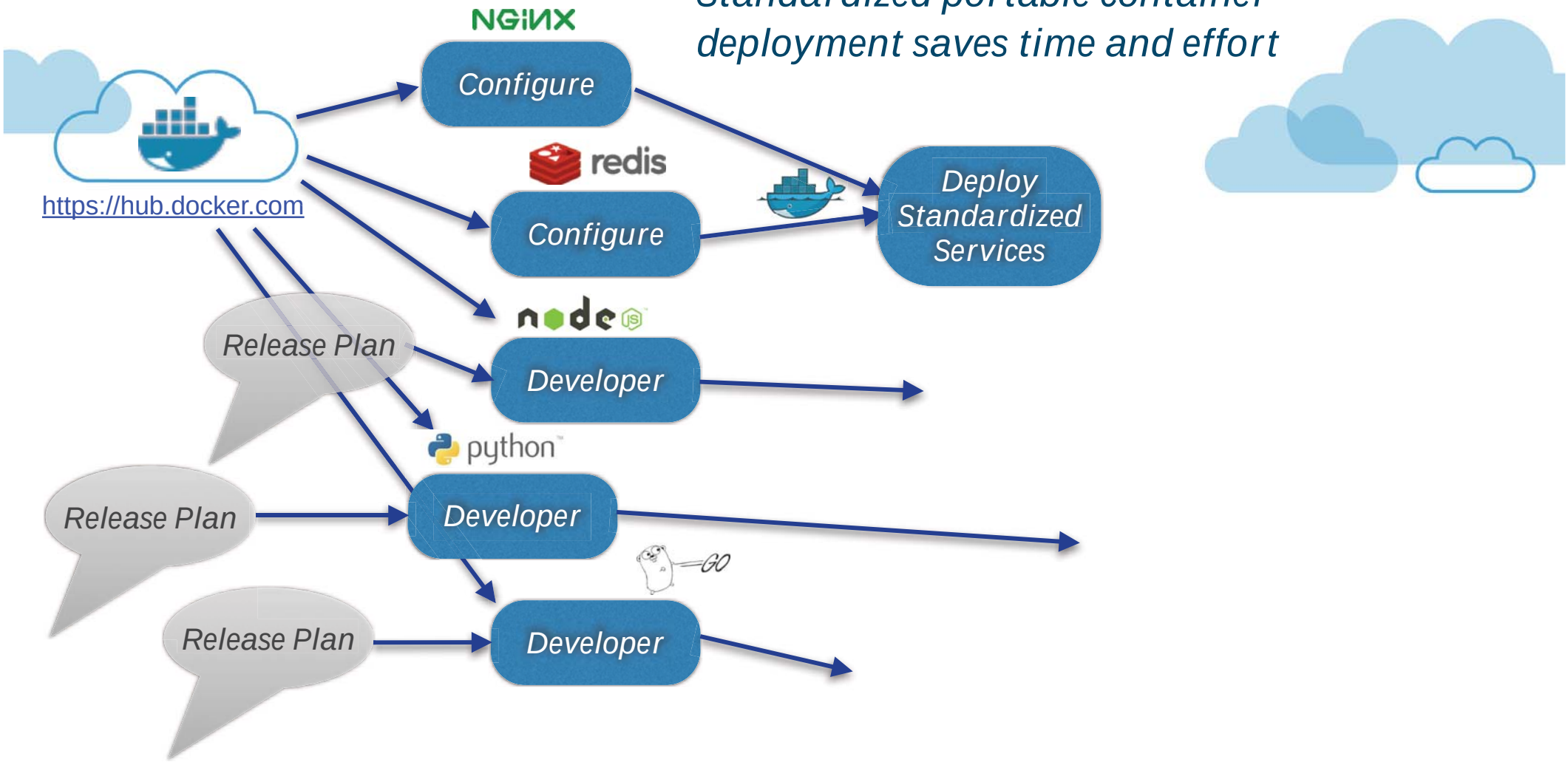
Immutable microservice deployment scales, is faster with large teams and diverse platform components

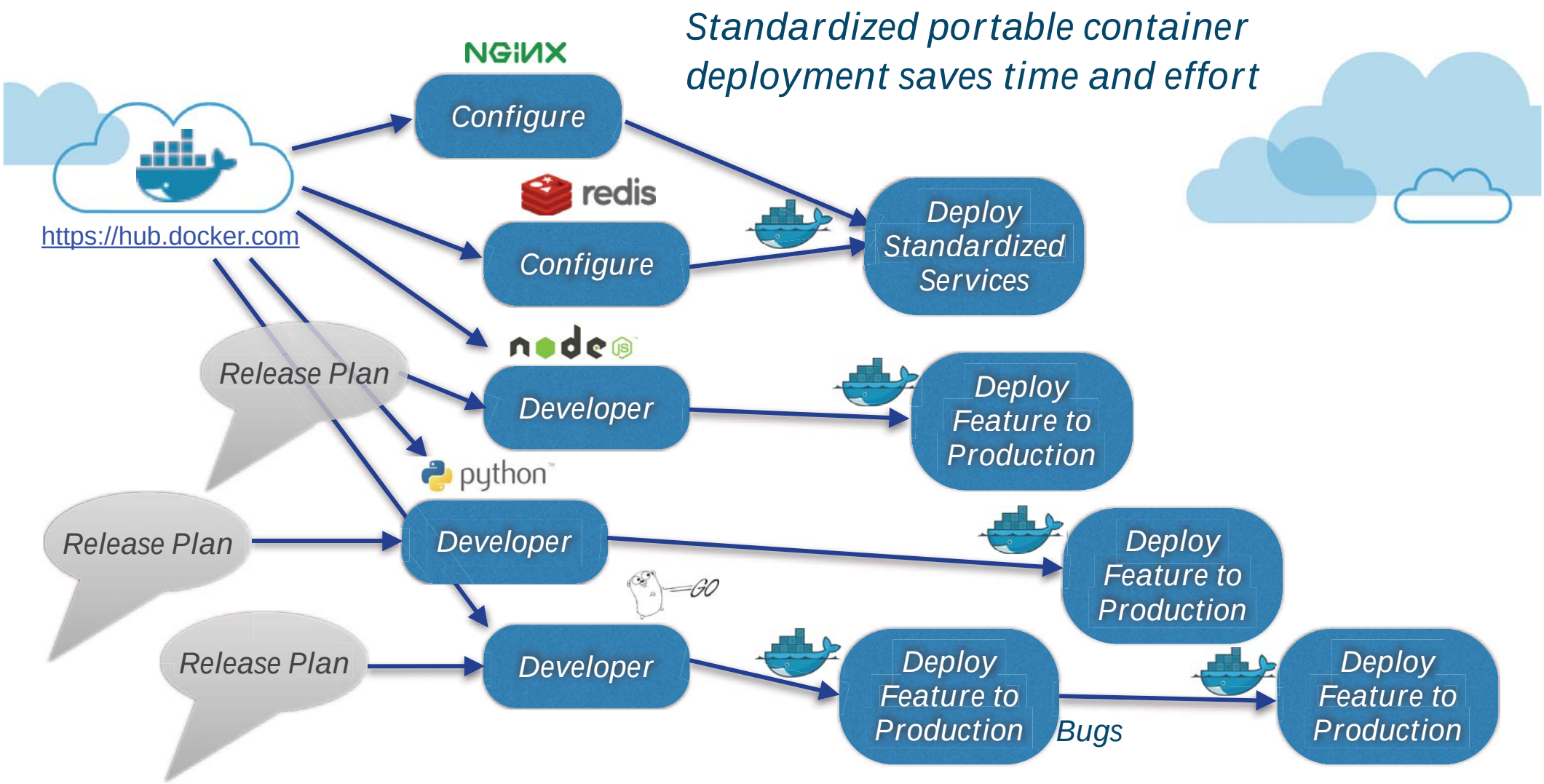



Immutable microservice deployment scales, is faster with large teams and diverse platform components
php
Developer
Release Plan
Deploy Feature to Production
node
python
GO
Old Release Still Running
Bugs


Immutable microservice deployment scales, is faster with large teams and diverse platform components
php
Developer
Release Plan
Deploy Feature to Production
node
python
GO
Old Release Still Running
Bugs

Standardized portable container deployment saves time and effort

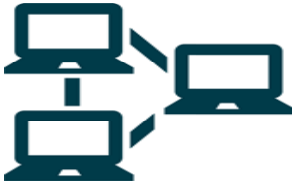



Standardized portable container deployment saves time and effort
NGINX
Configure
https://hub.docker.com
redis
Configure
Deploy Standardized Services
node
Release Plan
Developer
Deploy Feature to Production
python
Release Plan
Developer
Deploy Feature to Production
GO
Release Plan
Developer
Deploy Feature to Production
Bugs
Deploy Feature to Production

Developing at the Speed of Docker

Developers
- Compile/Build
- Seconds

Extend container
- Package dependencies
- Seconds

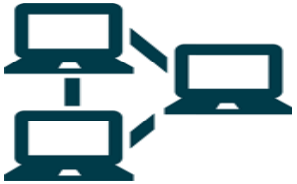


PaaS deploy Containers
- Docker startup
- Seconds

Developing at the Speed of Docker

Developers
- Compile/Build
- Seconds

Extend container
- Package dependencies
- Seconds



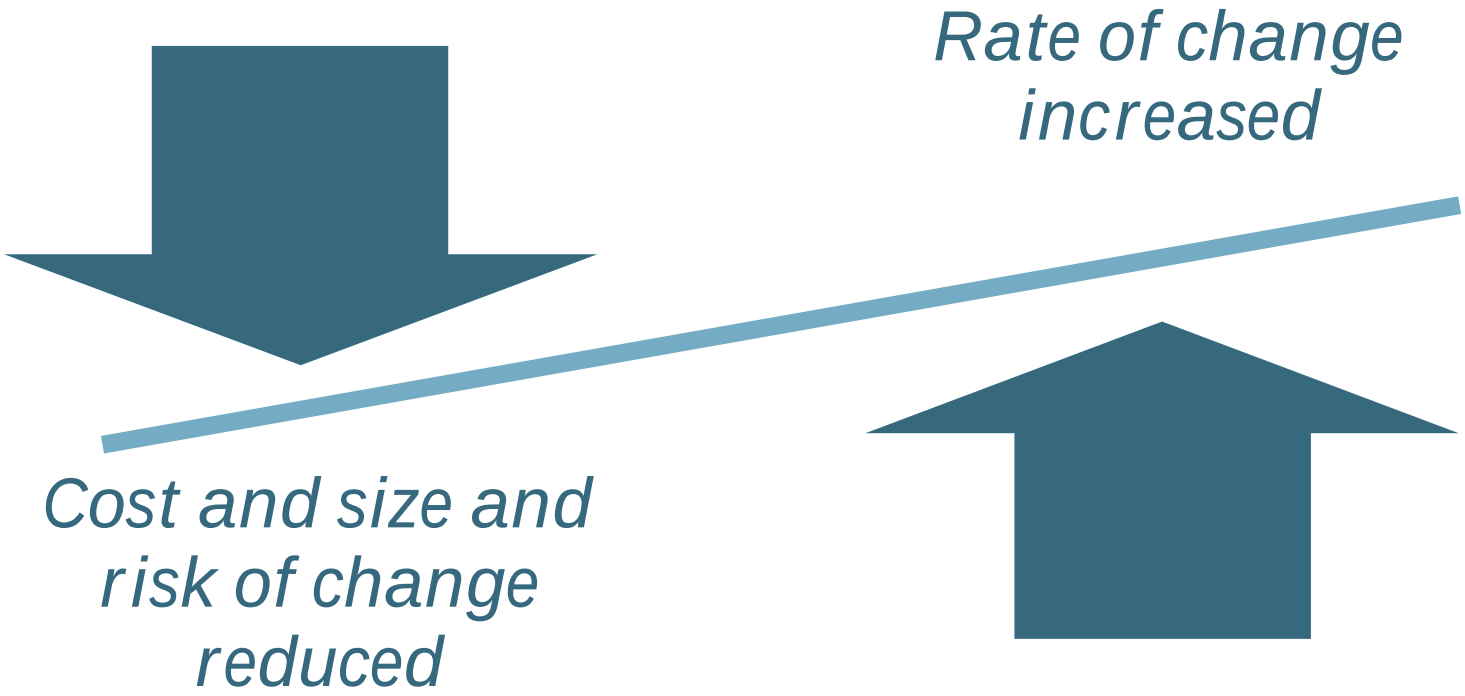
PaaS deploy Containers
- Docker startup
- Seconds

Speed is addictive, hard to go back to taking much longer to get things done

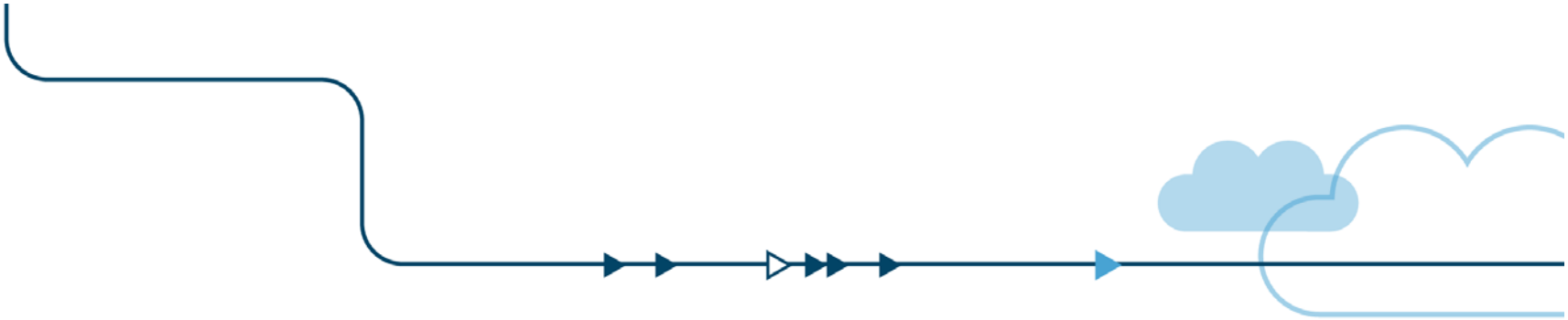
@adrianco

BV Battery Ventures

What Happened?



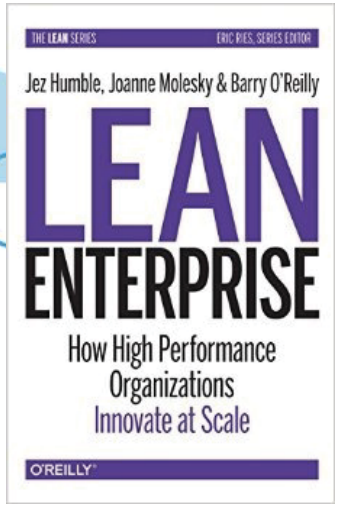
Cloud Native Applications

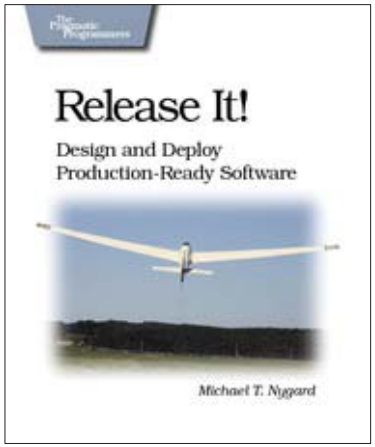


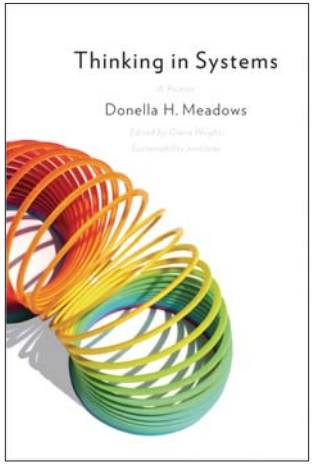
Cloud Native
A new engineering challenge

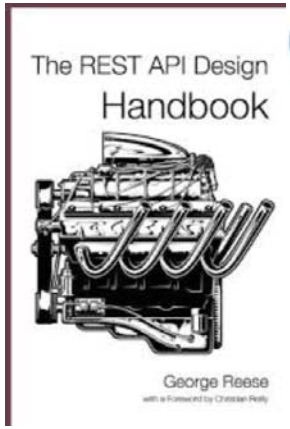
Construct a highly agile and highly available service from ephemeral and assumed broken components

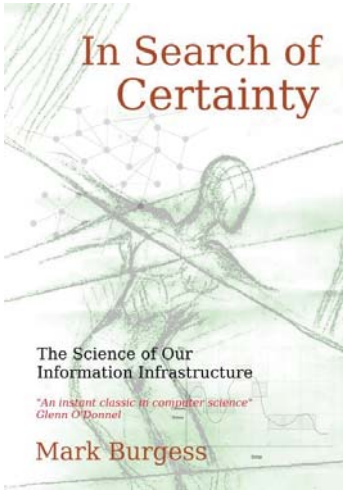
Inspiration

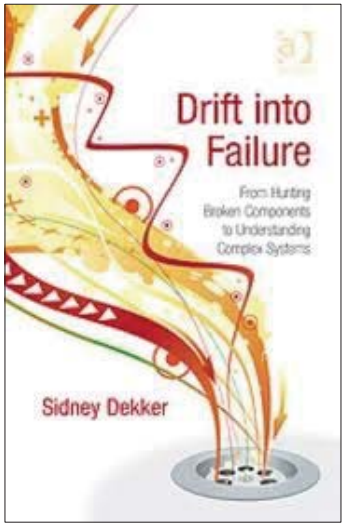


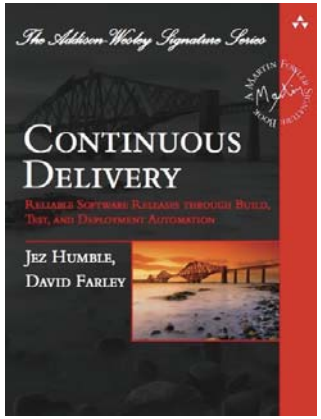


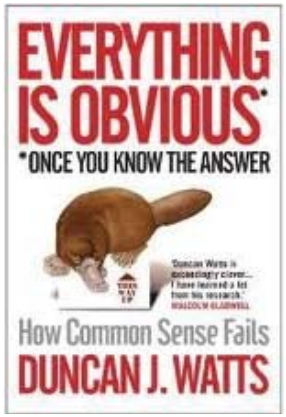


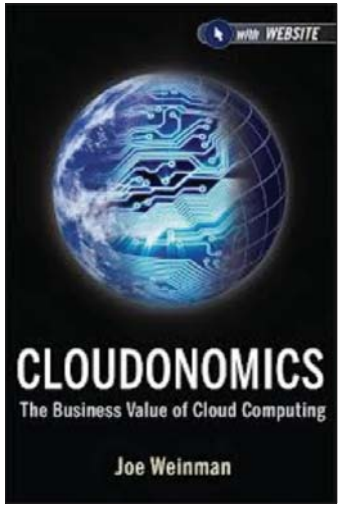




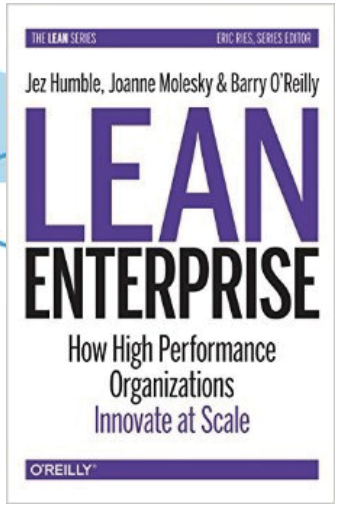


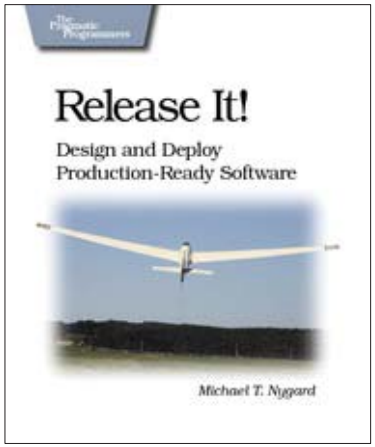


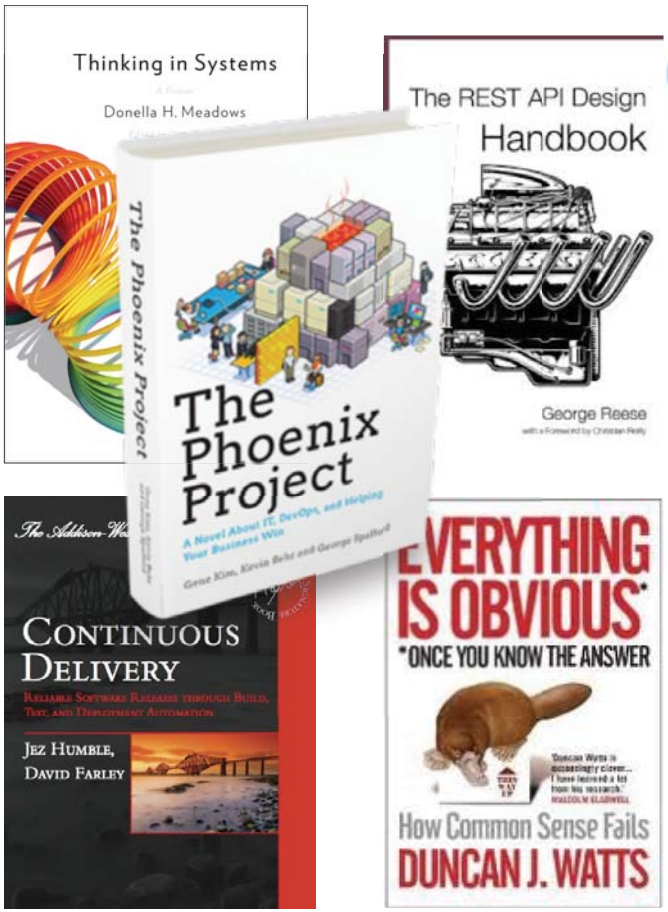


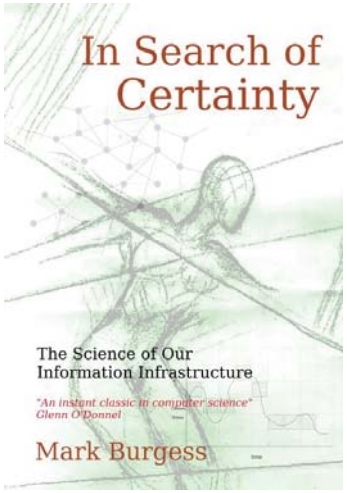


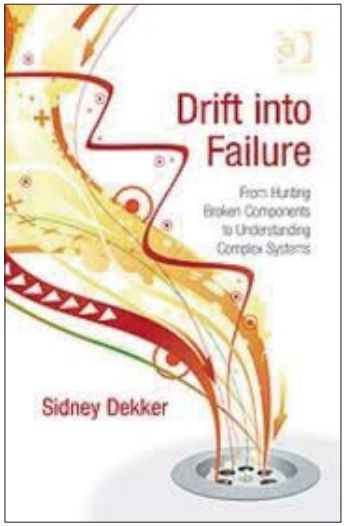
Inspiration

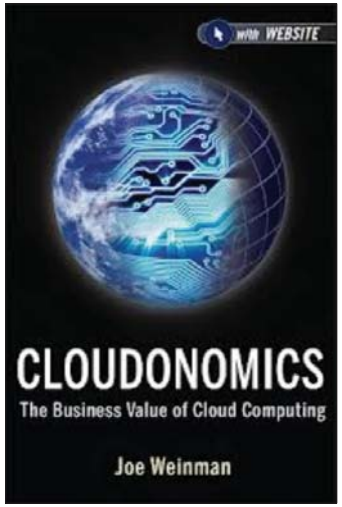












State of the Art in Cloud Native Microservice Architectures

NETFLIX | OSS

AWS Re:Invent : Asgard to Zuul https://www.youtube.com/watch?v=p7ysHhs5hl0
Resiliency at Massive Scale https://www.youtube.com/watch?v=ZfYJHtVL1_w
Microservice Architecture https://www.youtube.com/watch?v=CriDUYtfrjs

GILT

http://www.infoq.com/presentations/scale-gilt

GROUPON

http://www.slideshare.net/mcculloughsean/itier-breaking-up-the-monolith-philly-ete

http://www.infoq.com/presentations/Twitter-Timeline-Scalability
http://www.infoq.com/presentations/twitter-soa
http://www.infoq.com/presentations/Zipkin

HAIL O™

https://speakerdeck.com/mattheath/scaling-micro-services-in-go-highload-plus-plus-2014

NETFLIX | OSS Trust with Verification

- *Edda - the "black box flight recorder" for configuration state*
- *Chaos Monkey - enforcing stateless business logic*

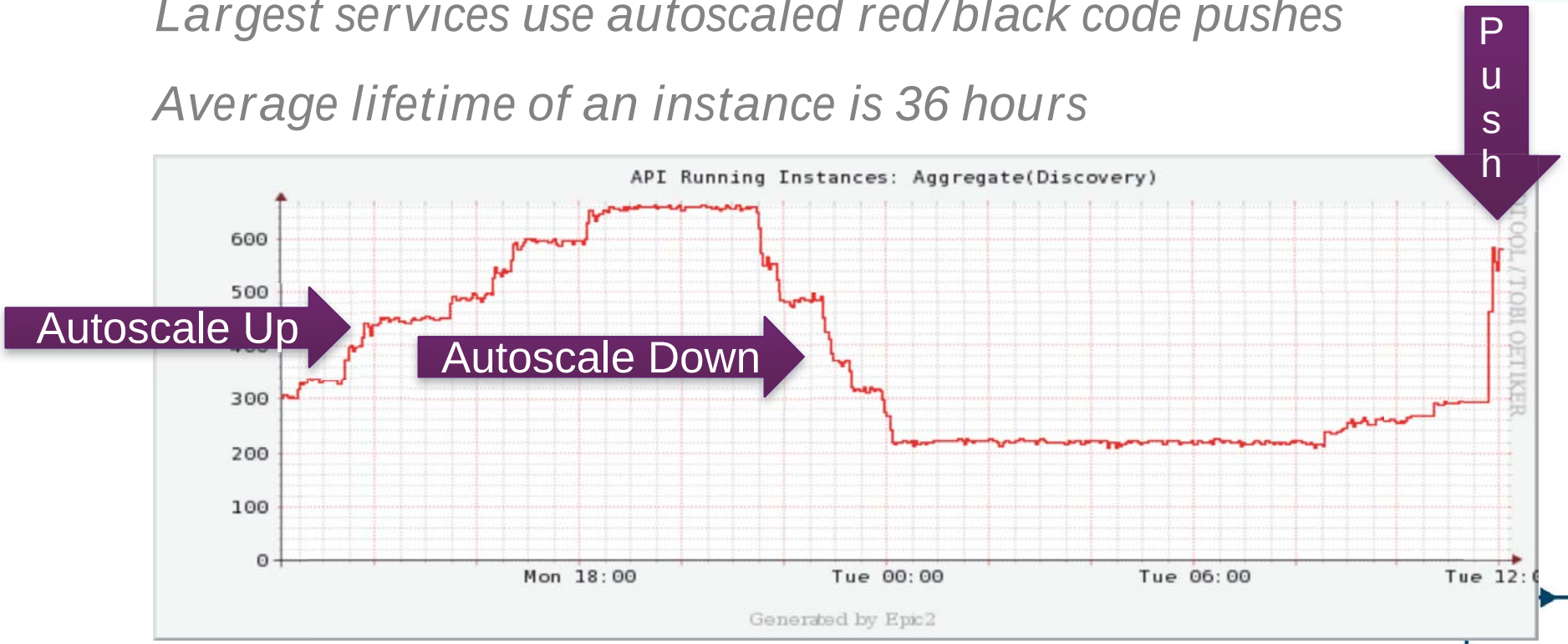
- *Chaos Gorilla - enforcing zone isolation/replication*
- *Chaos Kong - enforcing region isolation/replication*

- *Security Monkey - watching for insecure configuration settings*
- *See over 40 NetflixOSS projects at netflix.github.com*
- *Get "Technical Indigestion" trying to keep up with techblog.netflix.com*

Autoscaled Ephemeral Instances at Netflix

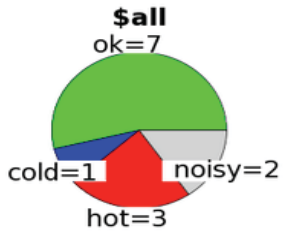
Largest services use autoscaled red/black code pushes

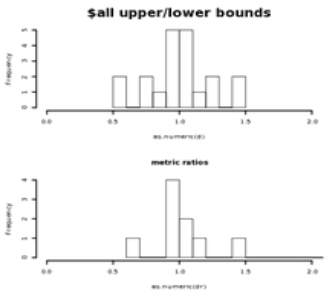
Average lifetime of an instance is 36 hours

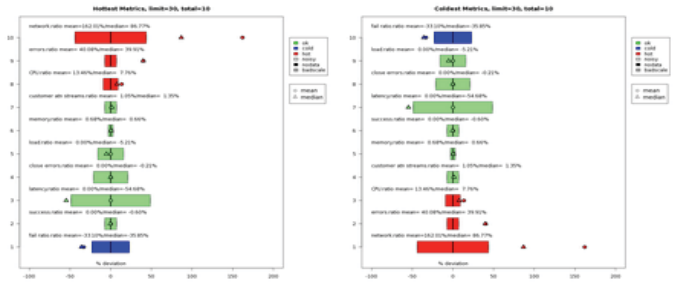


Netflix Automatic Code Deployment Canary Bad Signature

$all: state=hot (0%)



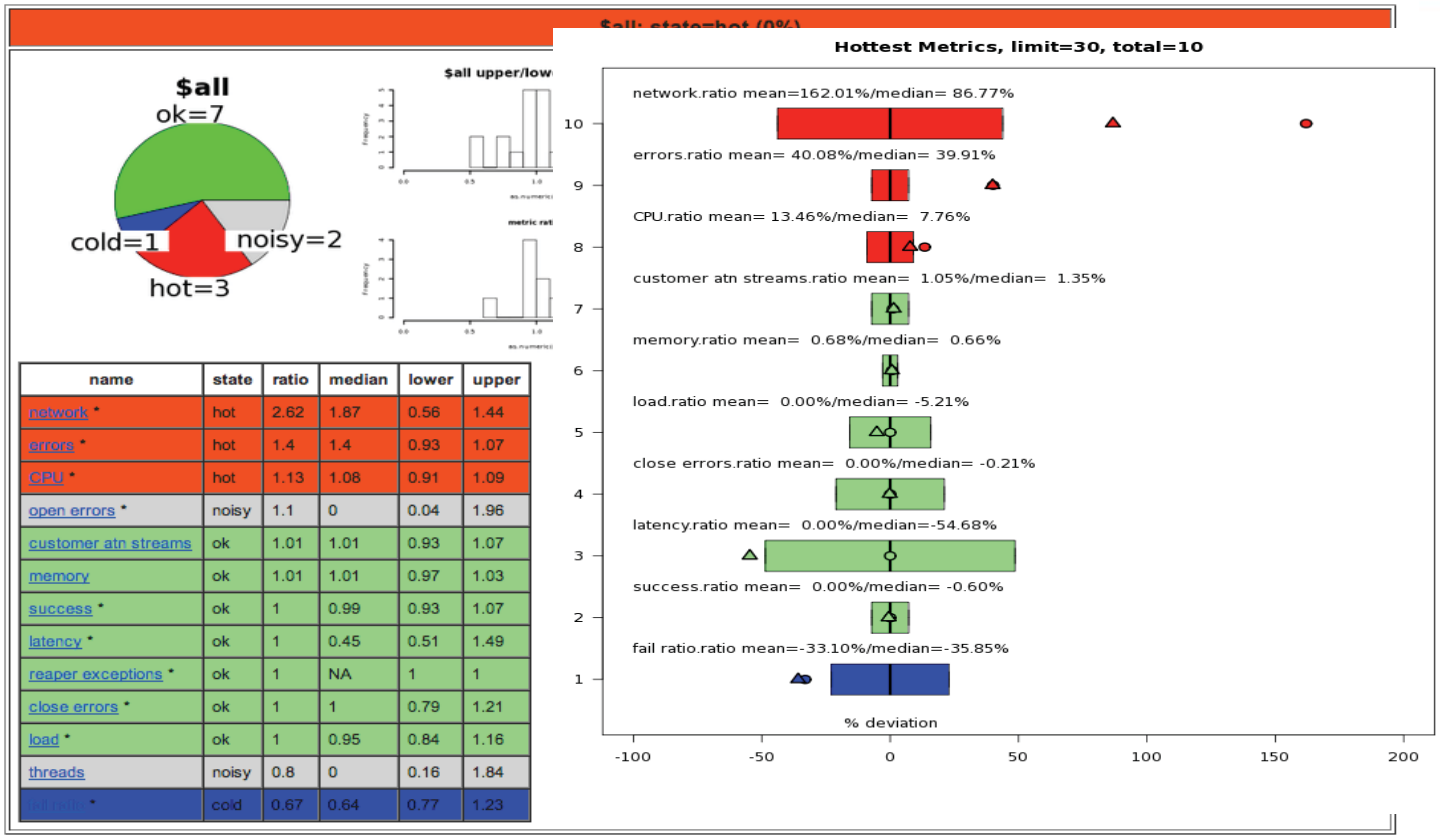




name	state	ratio	median	lower	upper
network *	hot	2.62	1.87	0.56	1.44
errors *	hot	1.4	1.4	0.93	1.07
CPU *	hot	1.13	1.08	0.91	1.09
open errors *	noisy	1.1	0	0.04	1.96
customer atn streams	ok	1.01	1.01	0.93	1.07
memory	ok	1.01	1.01	0.97	1.03
success *	ok	1	0.99	0.93	1.07
latency *	ok	1	0.45	0.51	1.49
reaper exceptions *	ok	1	NA	1	1
close errors *	ok	1	1	0.79	1.21
load *	ok	1	0.95	0.84	1.16
threads	noisy	0.8	0	0.16	1.84
[illegible] *	cold	0.67	0.64	0.77	1.23

Implemented by Simon Tuffs

Netflix Automatic Code Deployment Canary Bad Signature

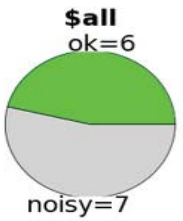


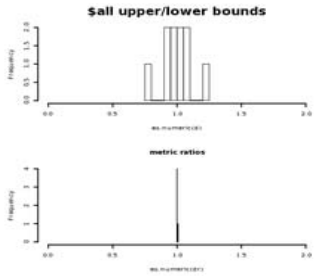
name	state	ratio	median	lower	upper
network *	hot	2.62	1.87	0.56	1.44
errors *	hot	1.4	1.4	0.93	1.07
CPU *	hot	1.13	1.08	0.91	1.09
open errors *	noisy	1.1	0	0.04	1.96
customer atn streams	ok	1.01	1.01	0.93	1.07
memory	ok	1.01	1.01	0.97	1.03
success *	ok	1	0.99	0.93	1.07
latency *	ok	1	0.45	0.51	1.49
reaper exceptions *	ok	1	NA	1	1
close errors *	ok	1	1	0.79	1.21
load *	ok	1	0.95	0.84	1.16
threads	noisy	0.8	0	0.16	1.84
fail ratio *	cold	0.67	0.64	0.77	1.23

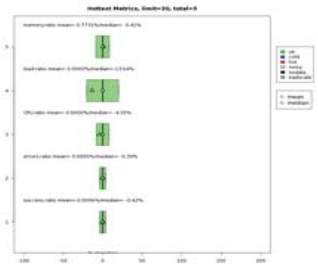
Implemented by Simon Tuffs

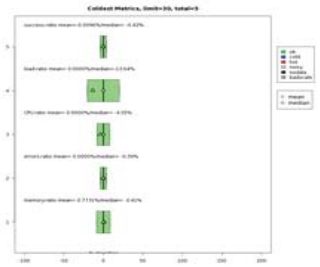
Happy Canary Signature

$all: state=noisy (98%)









name	state	ratio	median	lower	upper
close errors *	noisy	1.19	0	0	6.13
failure *	noisy	1.18	0	0	6.14
fail ratio *	noisy	1.15	0	0	7.05
memory	ok	1.01	1	0.91	1.09
errors *	ok	1	1	0.96	1.04
latency *	noisy	1	0.59	0.31	1.69
customer atn streams *	ok	1	NA	1	1
CPU *	ok	1	0.96	0.92	1.08
load *	ok	1	0.86	0.8	1.2
network *	noisy	1	0.21	0	2.34
success	ok	1	1	0.96	1.04
threads	noisy	0.83	0.84	0.29	1.71
recovered	noisy	0.75	0	0	2.54

Speeding Up The Platform

Datacenter Snowflakes
- Deploy in months
- Live for years

Speeding Up The Platform

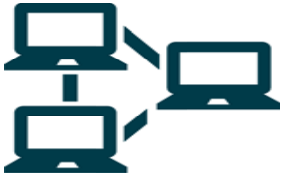
Datacenter Snowflakes
- Deploy in months
- Live for years

Virtualized and Cloud
- Deploy in minutes
- Live for weeks

Speeding Up The Platform

Datacenter Snowflakes
- Deploy in months
- Live for years

Virtualized and Cloud
- Deploy in minutes
- Live for weeks



Docker Containers
- Deploy in seconds
- Live for minutes/hours

Speeding Up The Platform

Datacenter Snowflakes
- Deploy in months
- Live for years

Virtualized and Cloud
- Deploy in minutes
- Live for weeks

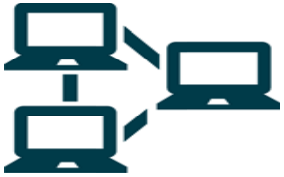
Docker Containers
- Deploy in seconds
- Live for minutes/hours

AWS Lambda
- Deploy in milliseconds
- Live for seconds

Speeding Up The Platform

Datacenter Snowflakes
- Deploy in months
- Live for years

Virtualized and Cloud
- Deploy in minutes
- Live for weeks

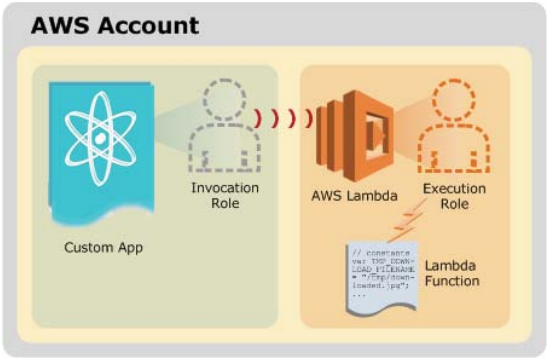


Docker Containers
- Deploy in seconds
- Live for minutes/hours

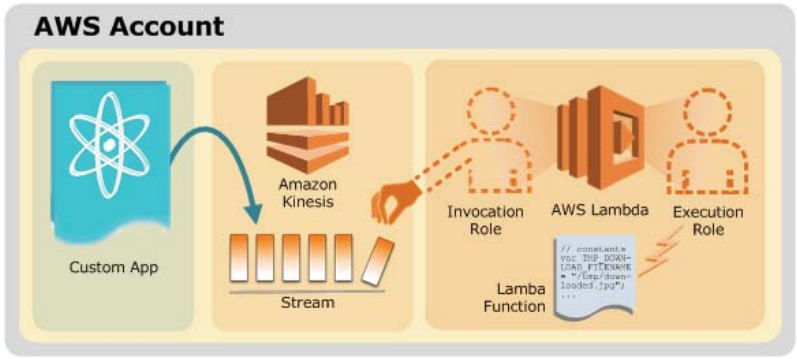
AWS Lambda
- Deploy in milliseconds
- Live for seconds

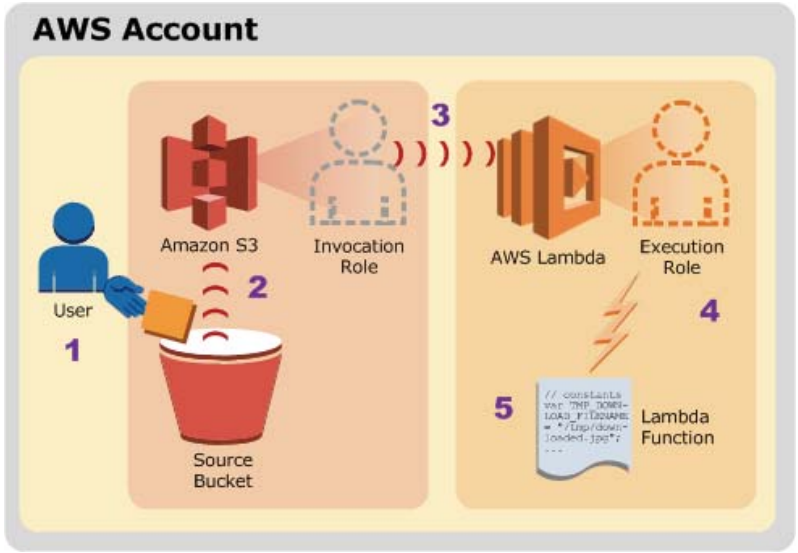
Speed enables and encourages new microservice architectures

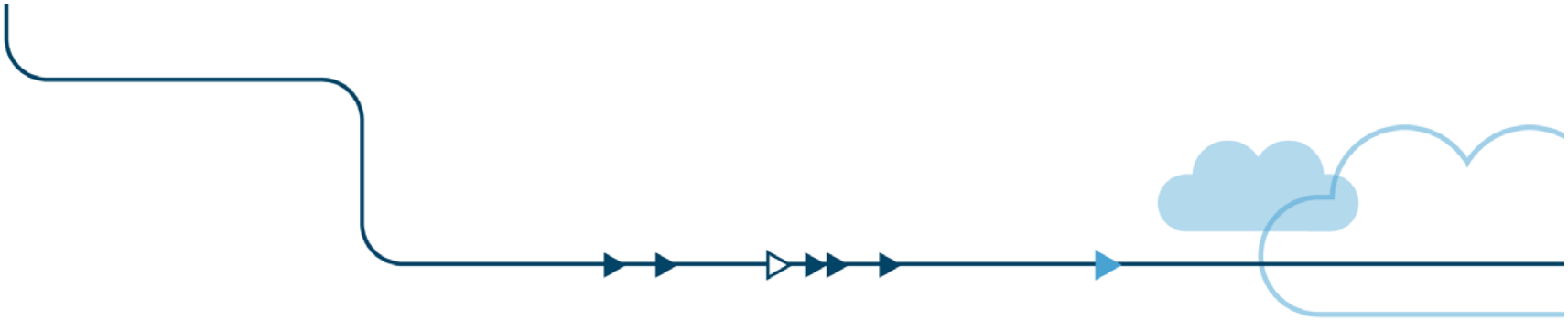
With AWS Lambda compute resources are charged by the 100ms, not the hour



First 1,000,000 node.js executions/month are free
First 400,000 GB-seconds of RAM-CPU are free







Monitoring Requirements

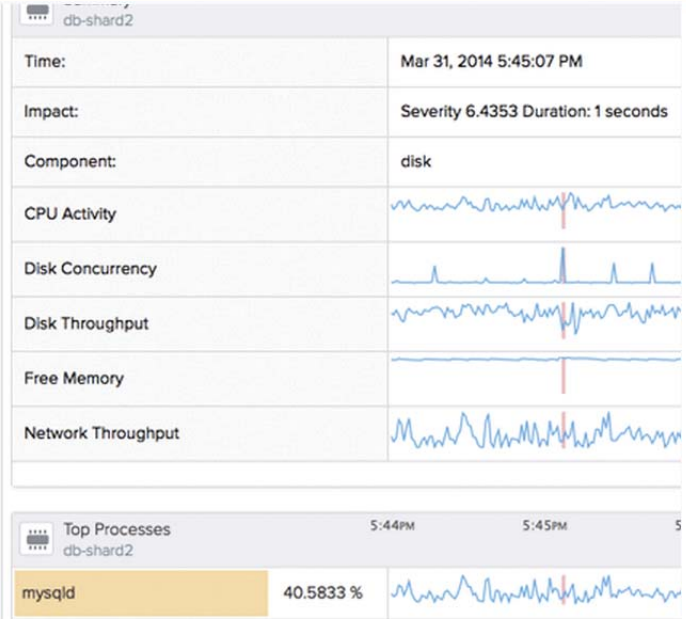
Metric resolution microseconds

Metric update rate 1 second

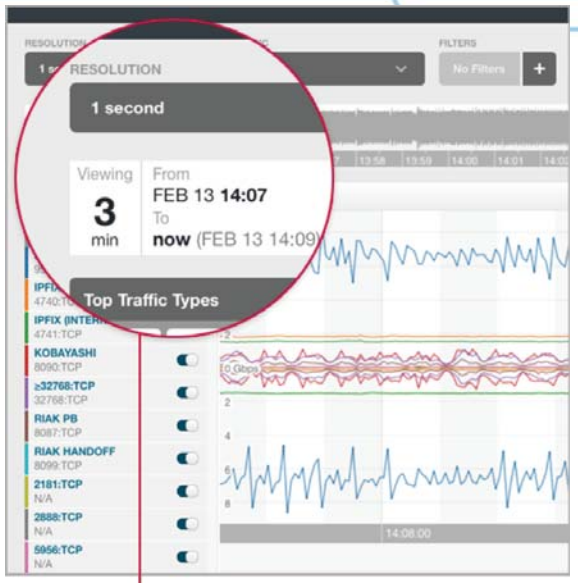
Metric to display latency less than human attention span (<10s)

Low Latency SaaS Based Monitors

VividCortex



boundary



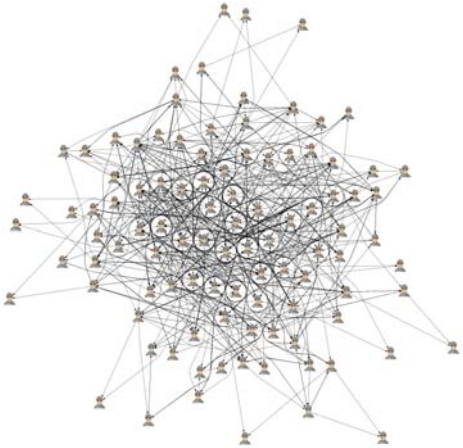
1-second data collection and real-time streaming processing on all components of the application stack

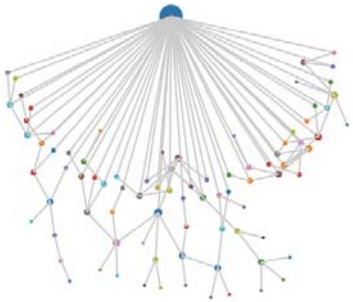
www.vividcortex.com and www.boundary.com

@adrianco

BV Battery Ventures

Adrian's Tinkering Projects





Model and visualize microservices
Simulate interesting architectures

See github.com/adrianco/spigo
Simulate Protocol Interactions in Go

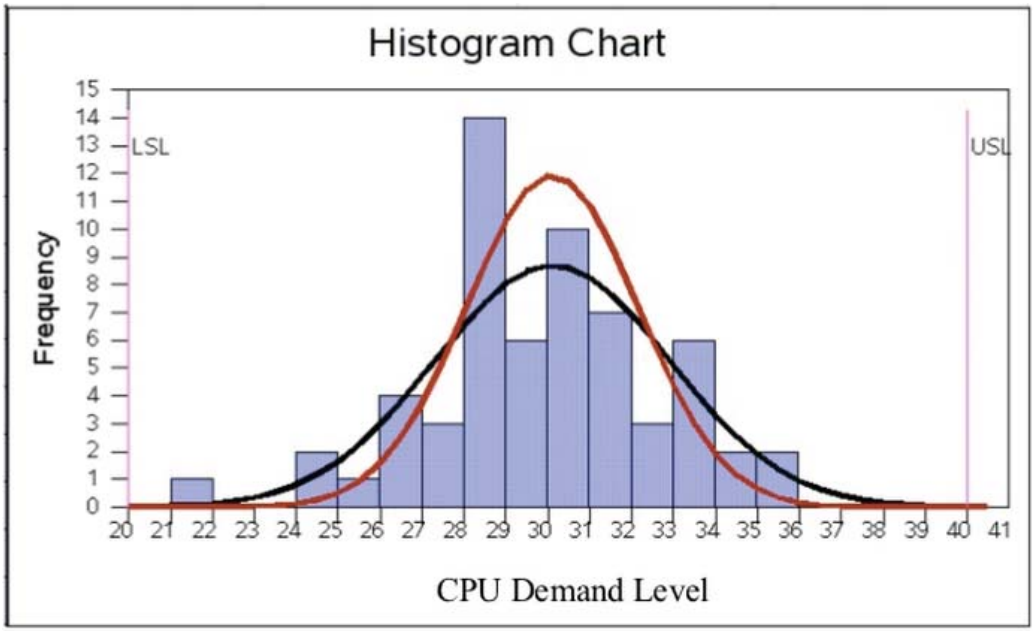
See github.com/adrianco/d3grow
Dynamic visualization

Cost Optimization

Capacity Optimization for a Single System Bottleneck

Lower Spec Limit

When demand probability is below USL by 3.0 sigma scale down resource to save money



Upper Spec Limit

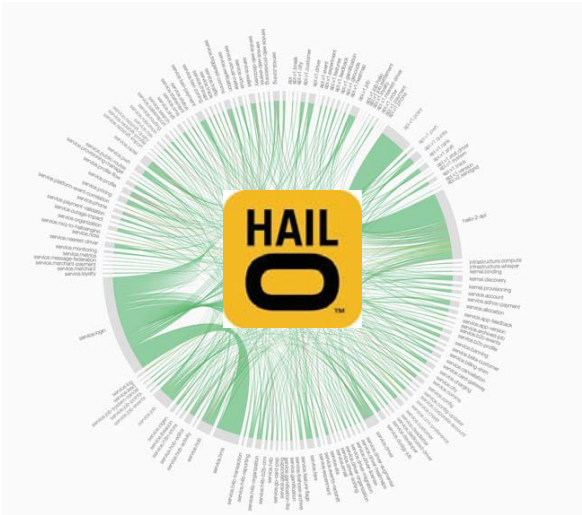
When demand probability exceeds USL by 4.0 sigma scale up resource to maintain low latency

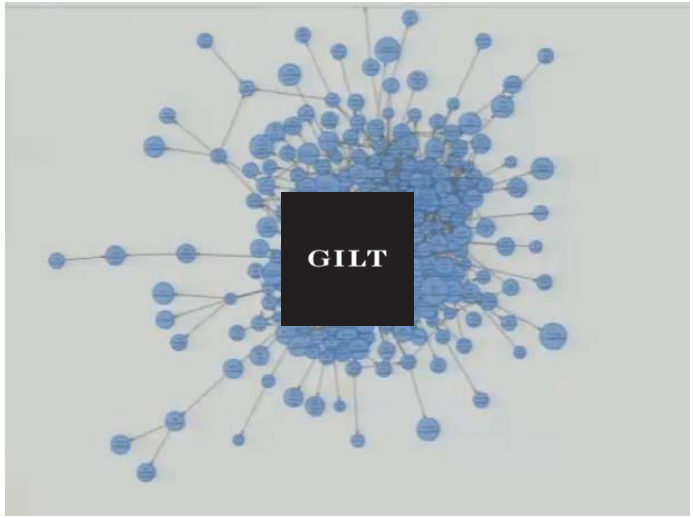
Documentation on Capability Plots

To get accurate high dynamic range histograms see http://hdrhistogram.org/

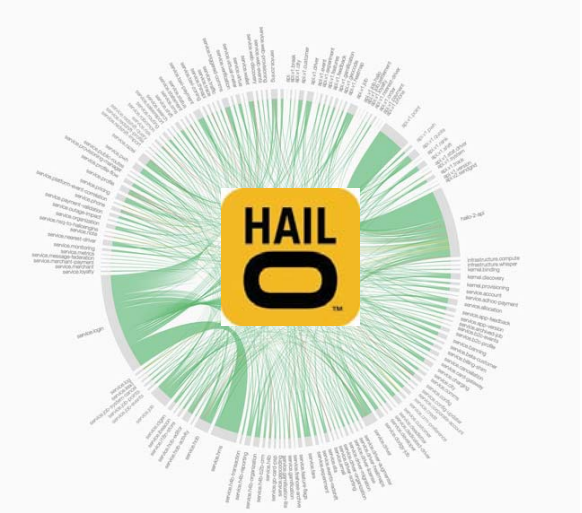
Slideshare: 2003 Presentation on Capacity Planning Methods See US Patent: 7467291

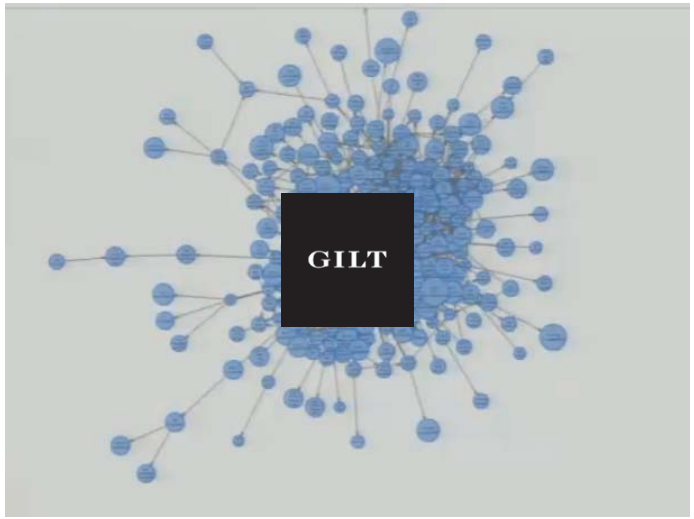
But interesting systems don't have a single bottleneck nowadays...

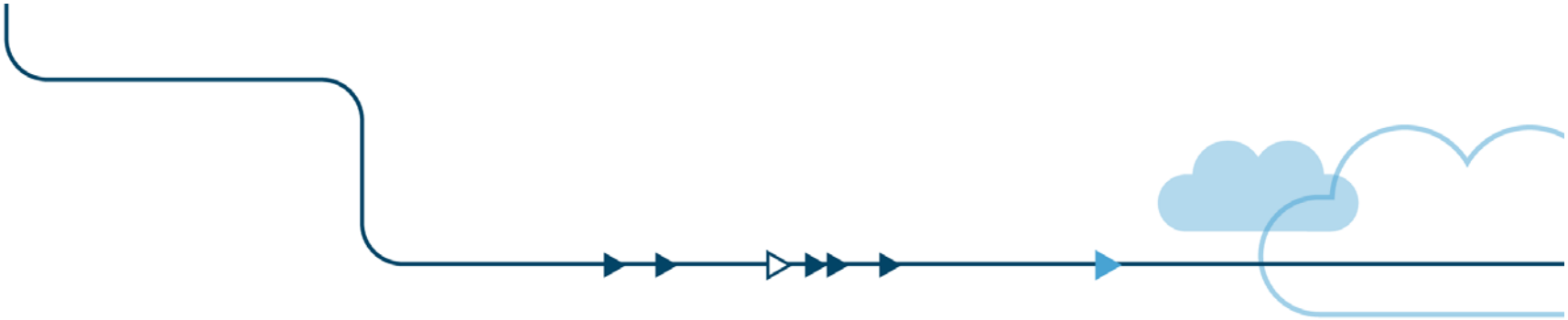




But interesting systems don't have a single bottleneck nowadays...







What about cloud costs?

Cloud Native Cost Optimization

$ $ $

Optimize for speed first

Turn it off!

Capacity on demand

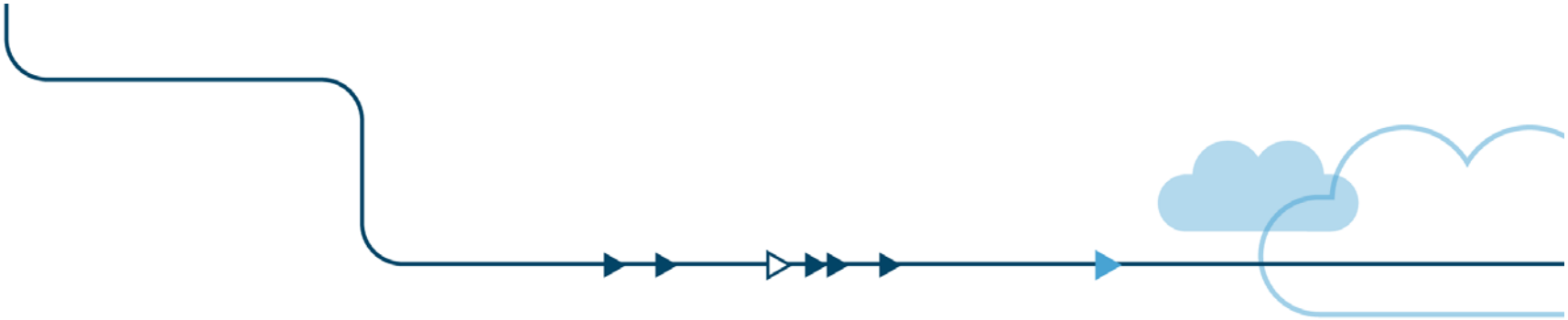
Consolidate and Reserve

Plan for price cuts

FOSS tooling

@adrianco

BV Battery Ventures



The Capacity Planning Problem

@adrianco
BV
Battery Ventures

Best Case Waste

Product Launch Agility - Rightsized

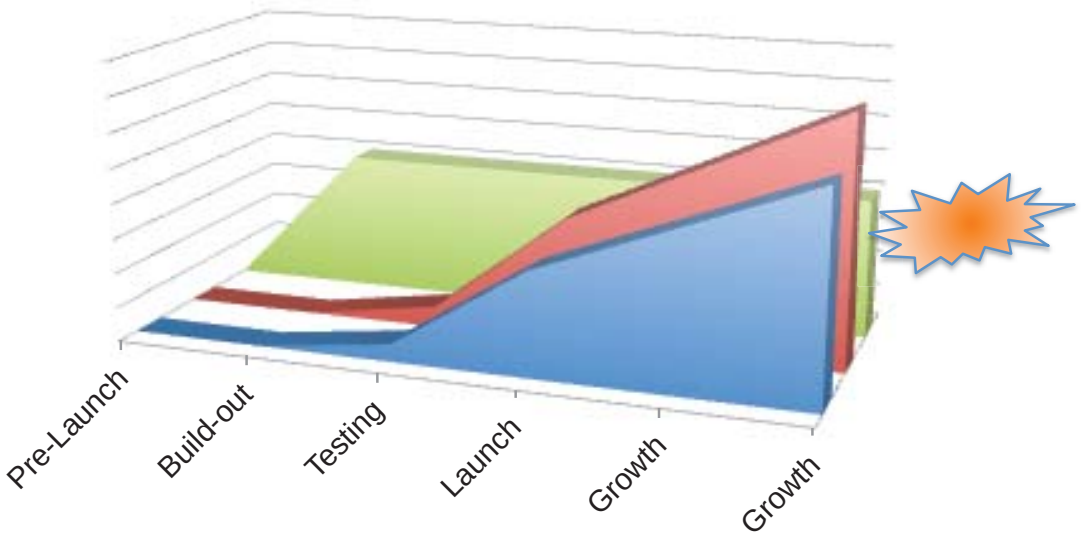
Cloud capacity used is maybe half average DC capacity

@adrianco

BV Battery Ventures

Failure to Launch

Product Launch - Under-estimated



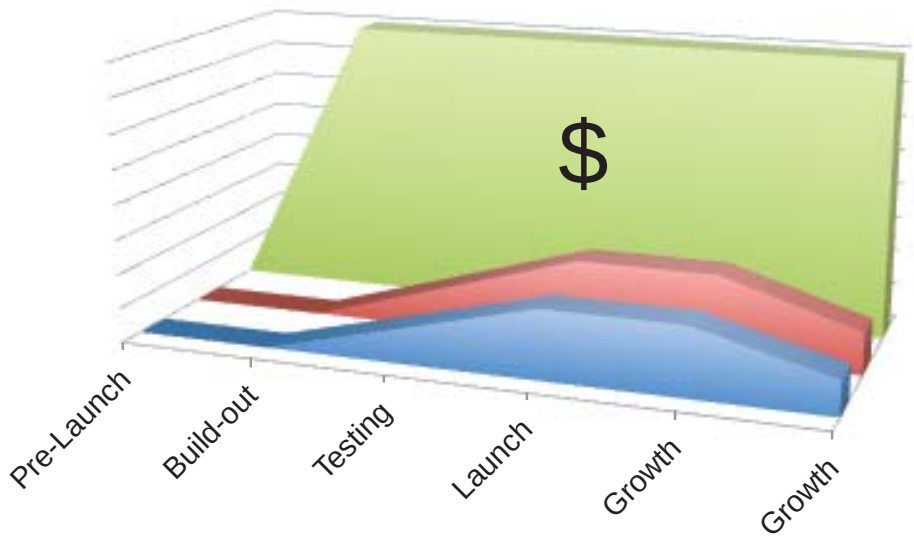
Mad scramble to add more DC capacity during launch phase outages

@adrianco

BV Battery Ventures

Over the Top Losses

Product Launch Agility – Over-estimated



Capacity wasted on failed launch magnifies the losses

@adrianco

BV Battery Ventures

Turning off Capacity

Off-peak production
Test environments
Dev out of hours
Dormant Data Science

Containerize Test Environments

Snapshot or freeze

Fast restart needed

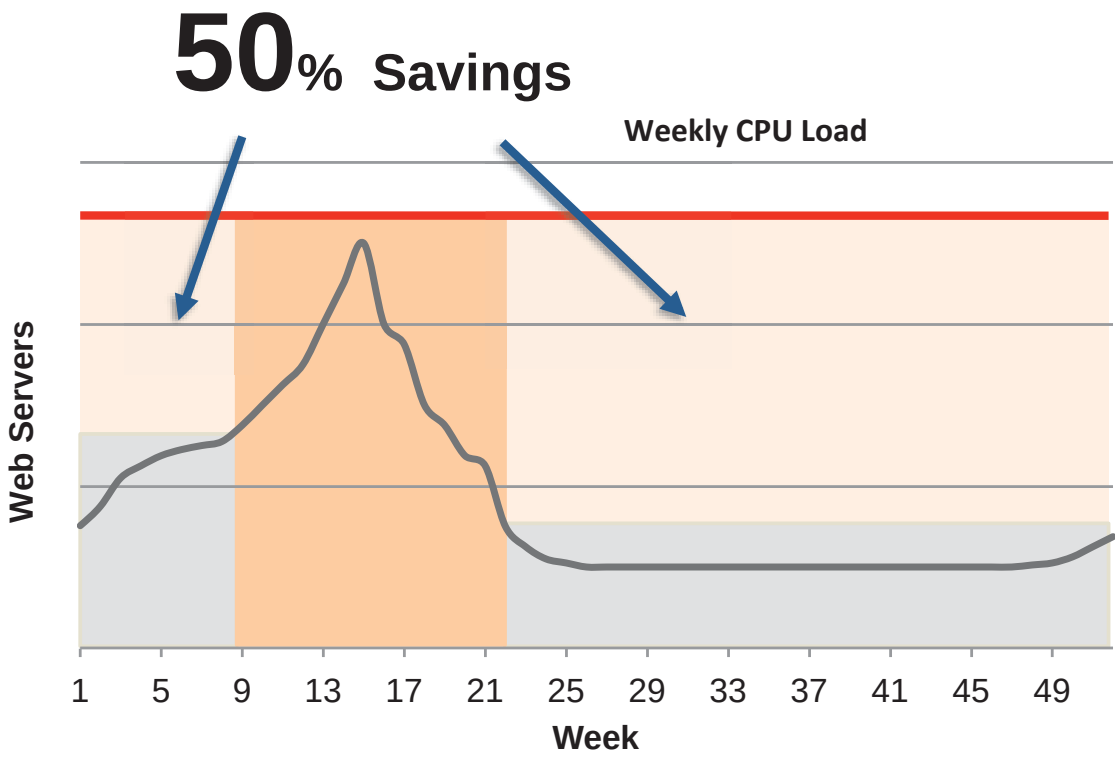
Persistent storage

40 of 168 hrs/wk

Bin-packed containers

shippable.com saved 70%

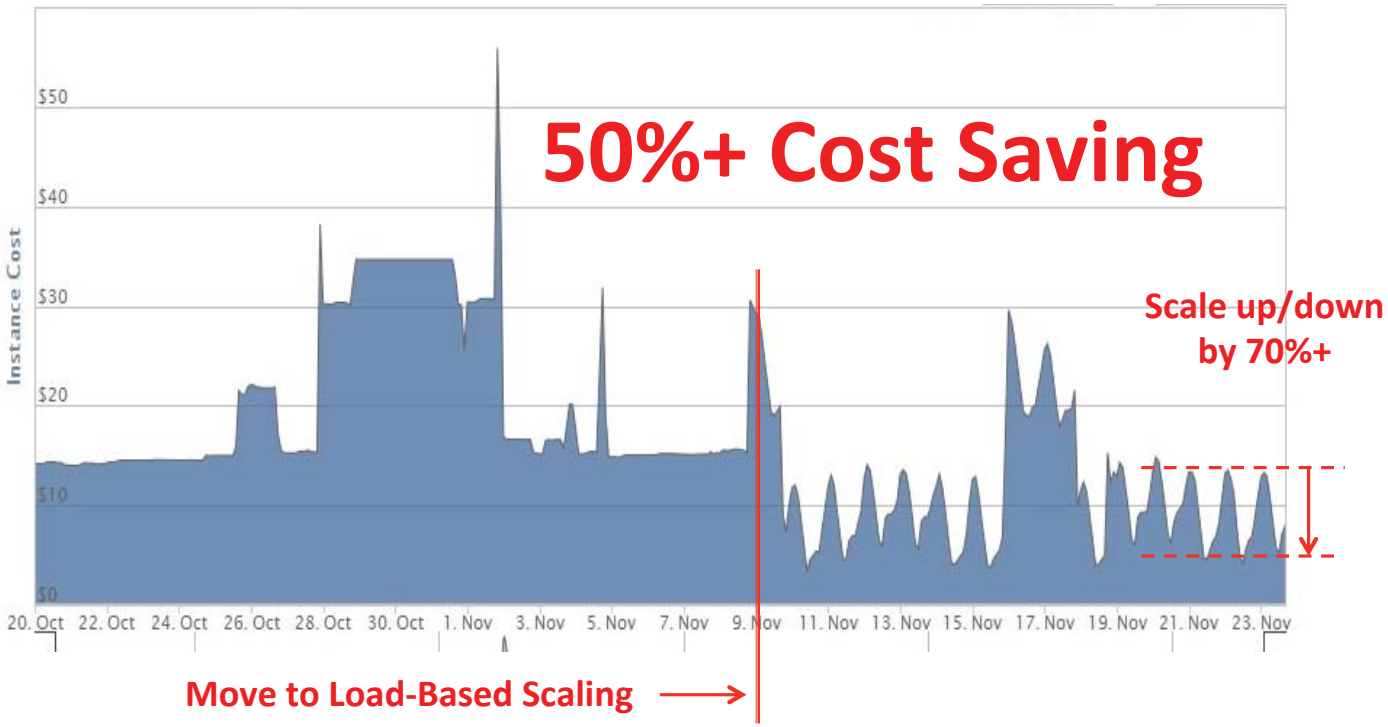
Seasonal Savings



@adrianco

BV
Battery Ventures

Autoscale the Costs Away

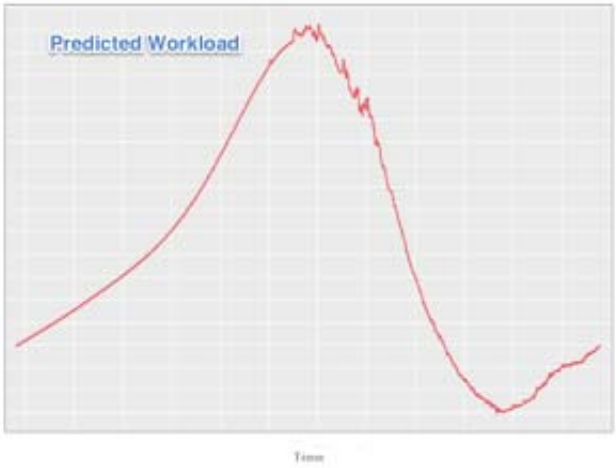


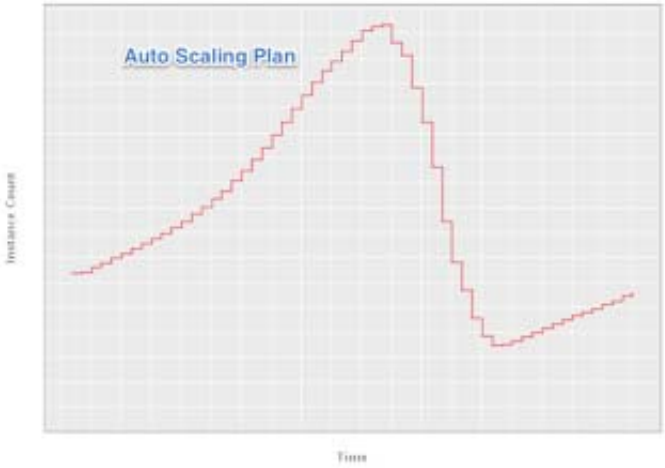
@adrianco

BV Battery Ventures

Daily Duty Cycle

Reactive Autoscaling saves around 50%





Predictive Autoscaling saves around 70%
See Scryer on Netflix Tech Blog

Underutilized and Unused

AWS Support – Trusted Advisor – Your personal cloud assistant

Trusted Advisor Beta

Expand All | Download Excel | Refresh All | Contact Support

The AWS Trusted Advisor program monitors AWS infrastructure services, identifies customer configurations, compares them to known best practices, and then notifies customers when opportunities may exist to save money, improve system performance, or close security gaps.

No issue detected | Investigation Recommended | Action Recommended

Cost Optimizing Checks

Unused Elastic IPs — Updated: **2012-06-14 00:00 PDT**

Summary: **0 of 6 Elastic IPs are not in use**

Underutilized EC2 Instances — Updated: **2012-06-13 22:27 PDT**

Summary: **27 EC2 instances are potentially underutilized**

@adrianco

BV Battery Ventures

Clean Up the Crud

Other simple optimization tips

- **Don't forget to…**
  - Disassociate unused EIPs
  - Delete unassociated Amazon EBS volumes
  - Delete older Amazon EBS snapshots
  - Leverage Amazon S3 Object Expiration

Janitor Monkey cleans up unused resources

Total Cost of Oranges

When Comparing TCO…

Make sure that you are including all the cost factors into consideration

Place
Power
Pipes
People
Patterns

@adrianco

BV
Battery Ventures

Total Cost of Oranges

When Comparing TCO…

Make sure that you are including all the cost factors into consideration

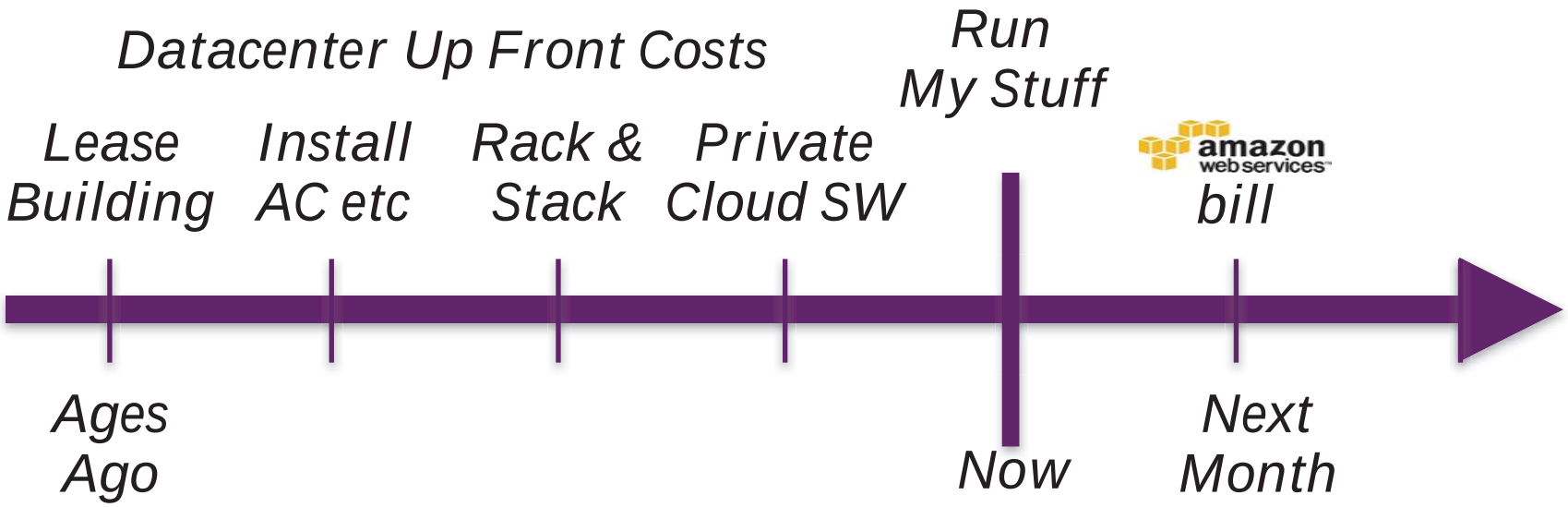
Place
Power
Pipes
People
Patterns

How much does datacenter automation software and support cost per instance?

@adrianco

BV Battery Ventures

When Do You Pay?



@adrianco

BV Battery Ventures

Cost Model Comparisons

AWS has most complex model

- *Both highest and lowest cost options!*

CPU/Memory Ratios Vary

- *Can't get same config everywhere*

Features Vary

- *Local SSD included on some vendors, not others*
- *Network and storage charges also vary*

Digital Ocean Flat Pricing

Hourly Price ($0.06/hr)	***Monthly Price ($40/mo)***
$ No Upfront	*$ No Upfront*
$0.060/hr	*$0.056/hr*
$1555/36mo	*$1440/36mo*
Savings	*7%*

Prices on Dec 7th, for 2 Core, 4G RAM, SSD, purely to show typical savings

Google Sustained Usage

Full Price Without Sustained Usage	Typical Sustained Usage Each Month	Full Sustained Usage Each Month
$ No Upfront	$ No Upfront	$ No Upfront
$0.063/hr	$0.049/hr	$0.045/hr
$1633/36mo	$1270/36mo	$1166/36mo
Savings	22%	29%

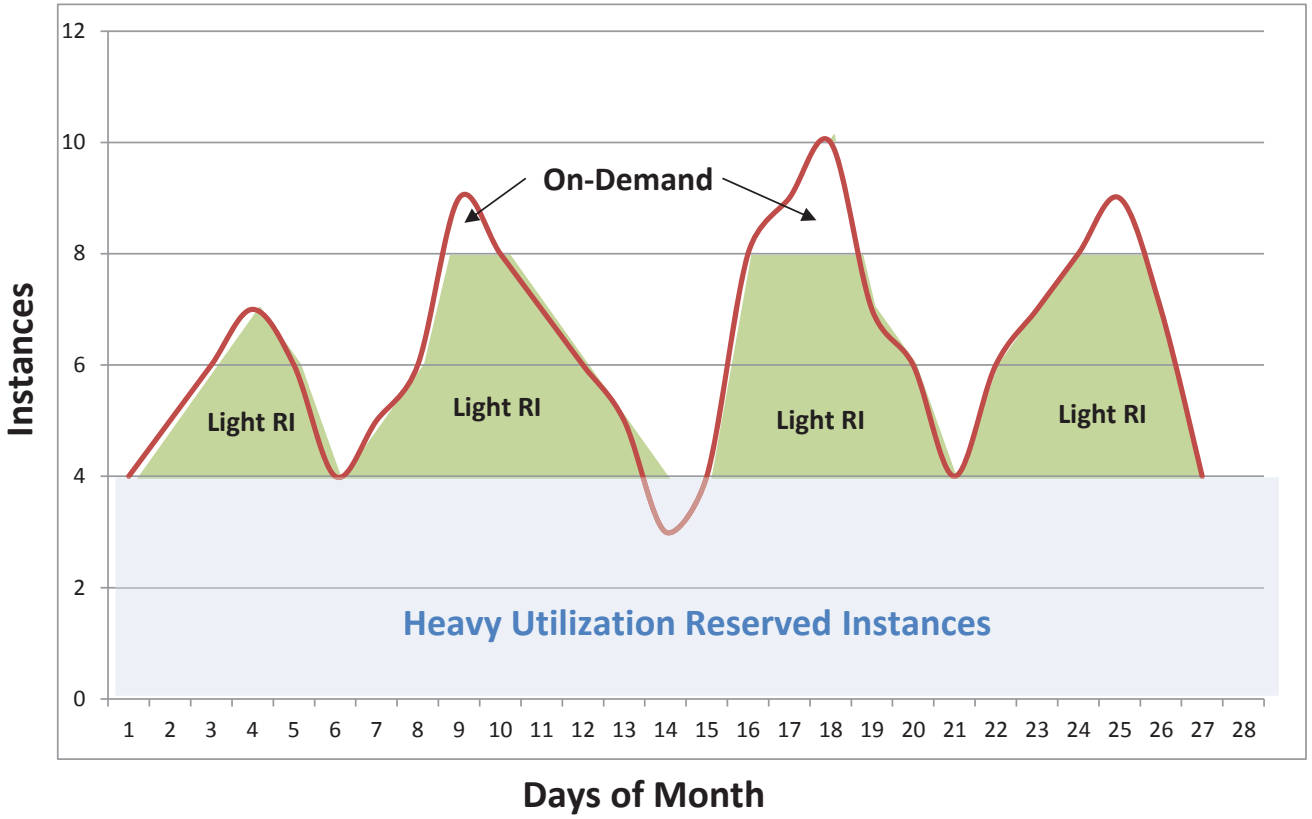
Prices on Dec 7th, for n1.standard-1 (1 vCPU, 3.75G RAM, no disk) purely to show typical savings

AWS Reservations

On Demand	No Upfront 1 year	Partial Upfront 3 year	All Upfront 3 year
$ No Upfront	$No Upfront	$337 Upfront	$687 Upfront
$0.070/hr	$0.050/hr	$0.0278/hr	$0.00/hr
$1840/36mo	$1314/36mo	$731/36mo	$687/36mo
Savings	29%	60%	63%

Prices on Dec 7th, for m3.medium (1 vCPU, 3.75G RAM, SSD) purely to show typical savings

Blended Benefits



On Demand

Partial Upfront

All Upfront

@adrianco

BV Battery Ventures

Consolidated Reservations

Burst capacity guarantee

Higher availability with lower cost

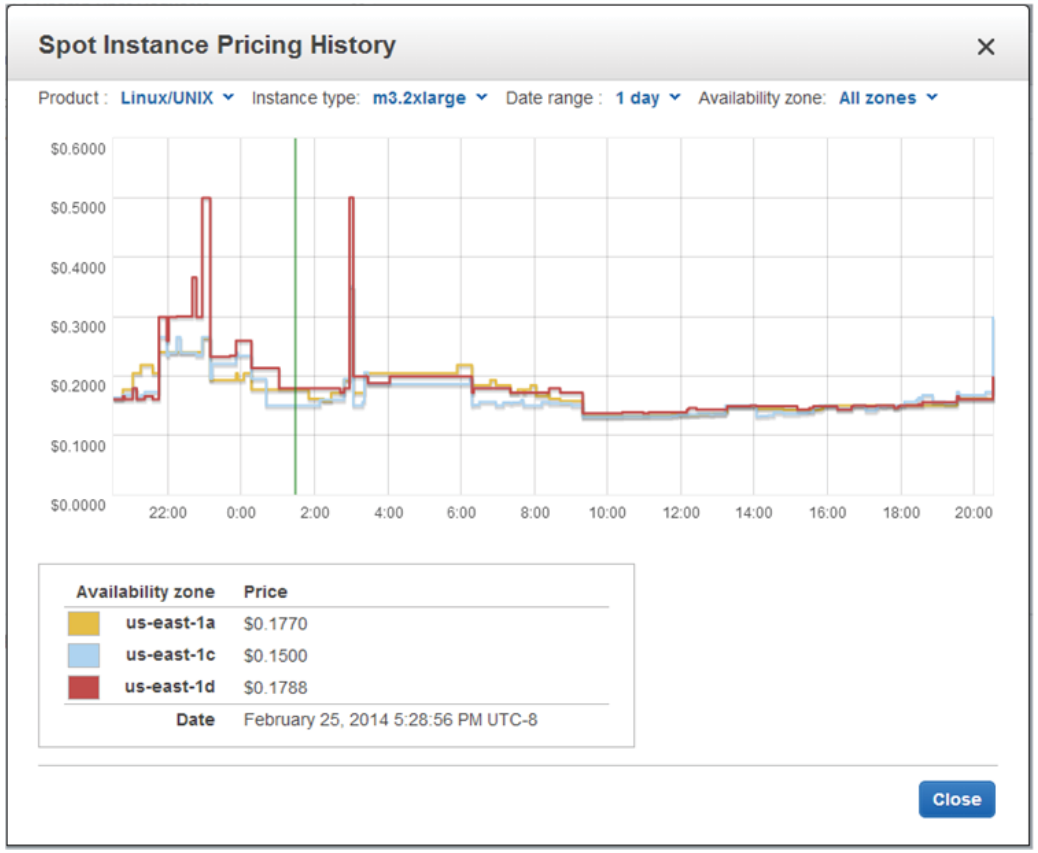
Other accounts soak up any extra

Monthly billing roll-up

Capitalize upfront charges!

But: Fixed location and instance type

Use EC2 Spot Instances



Cloud native dynamic autoscaled spot instances

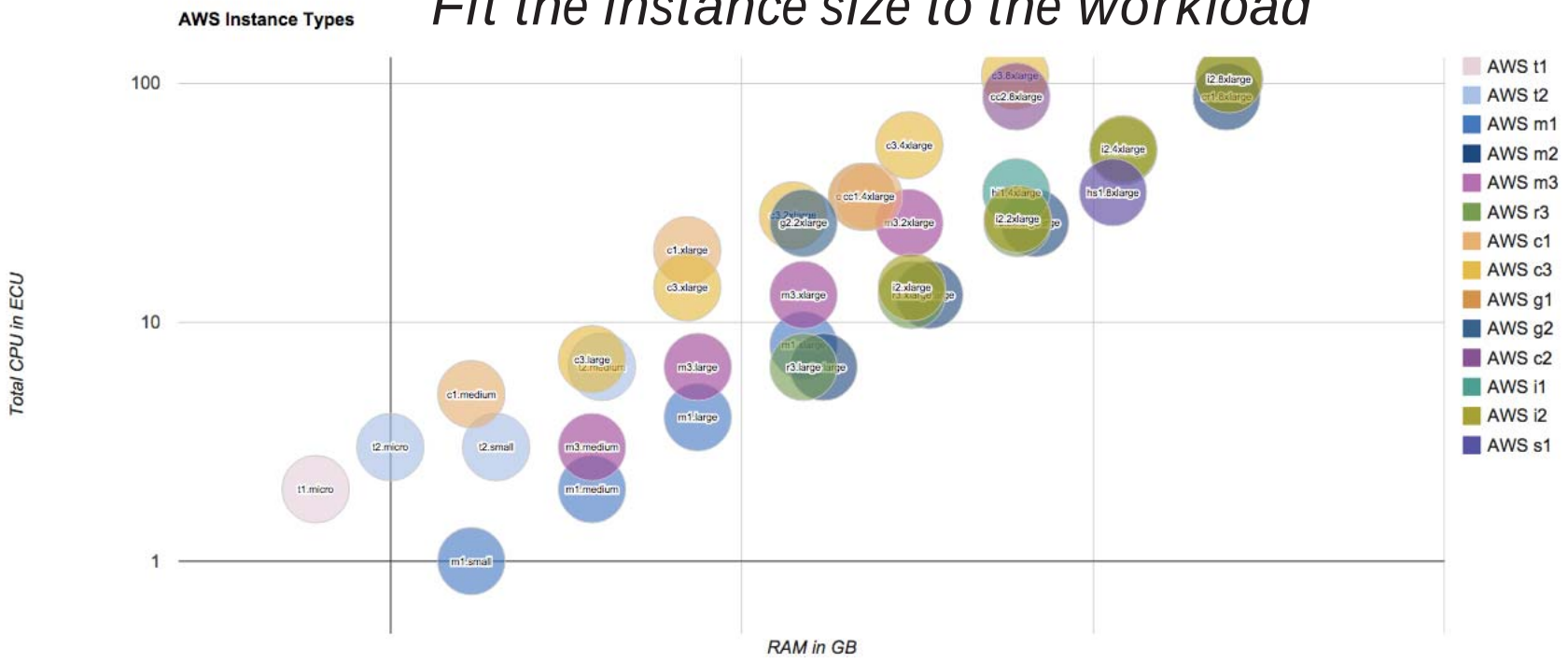
Real world total savings up to 50%

@adrianco

BV Battery Ventures

Right Sizing Instances

Fit the instance size to the workload



@adrianco

BV Battery Ventures

Six Ways to Cut Costs

#1 Business Agility by Rapid Experimentation = Profit

#2 Business-driven Auto Scaling Architectures = Savings

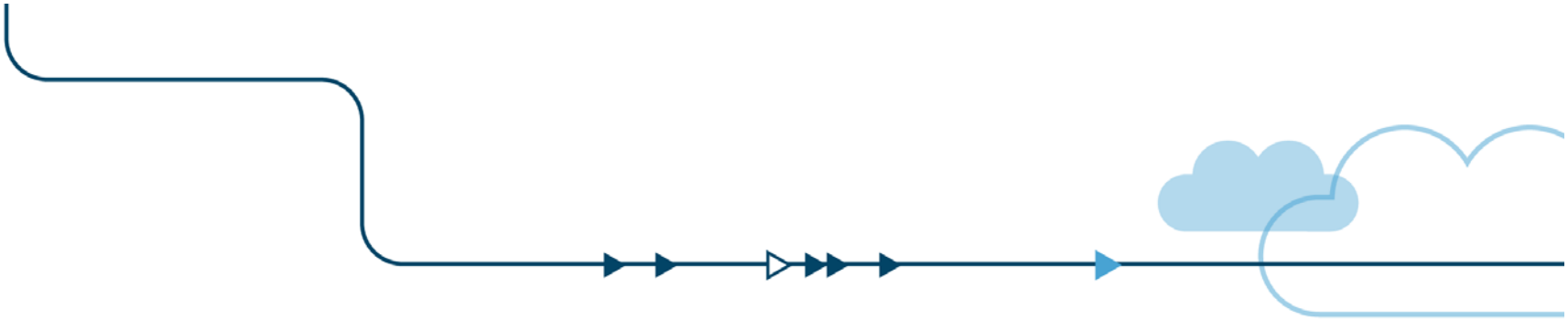
#3 Mix and Match Reserved Instances with On-Demand = Savings

#4 Consolidated Billing and Shared Reservations = Savings

#5 Always-on Instance Type Optimization = Recurring Savings

#6 Follow the Customer (Run web servers) during the day
Follow the Money (Run Hadoop clusters) at night

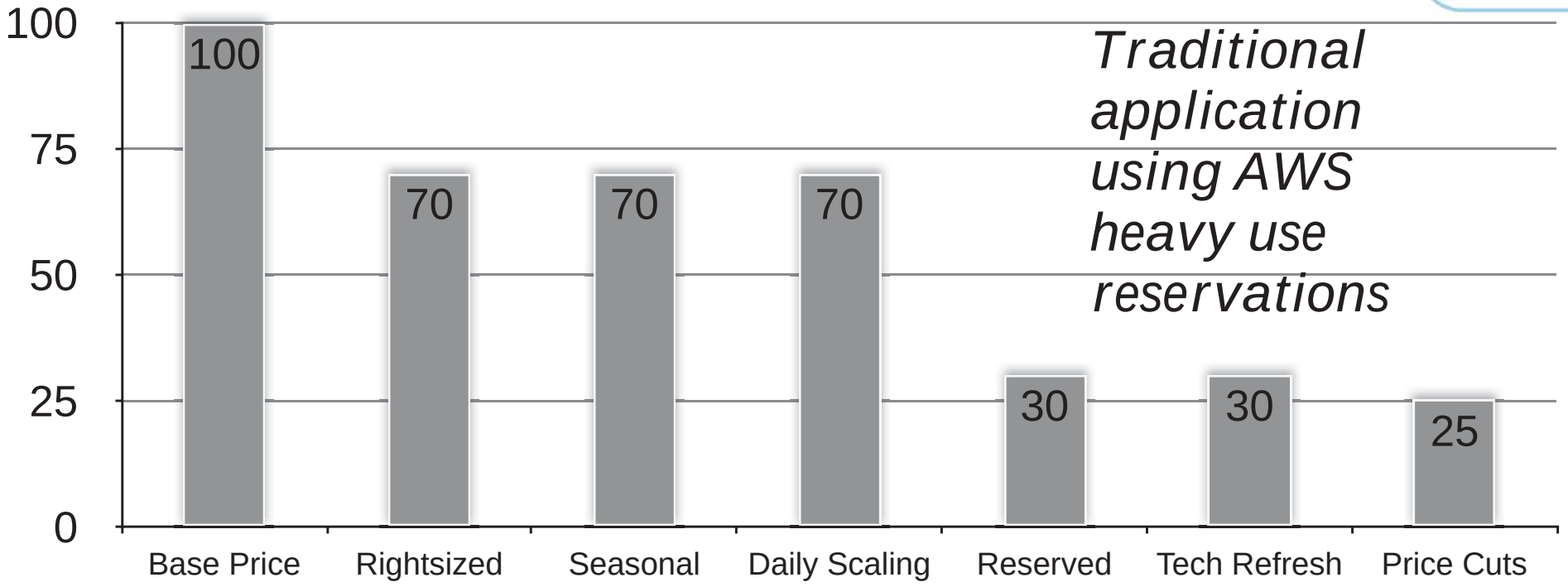
Credit to Jinesh Varia of AWS for this summary



Compounded Savings

@adrianco
BV
Battery Ventures

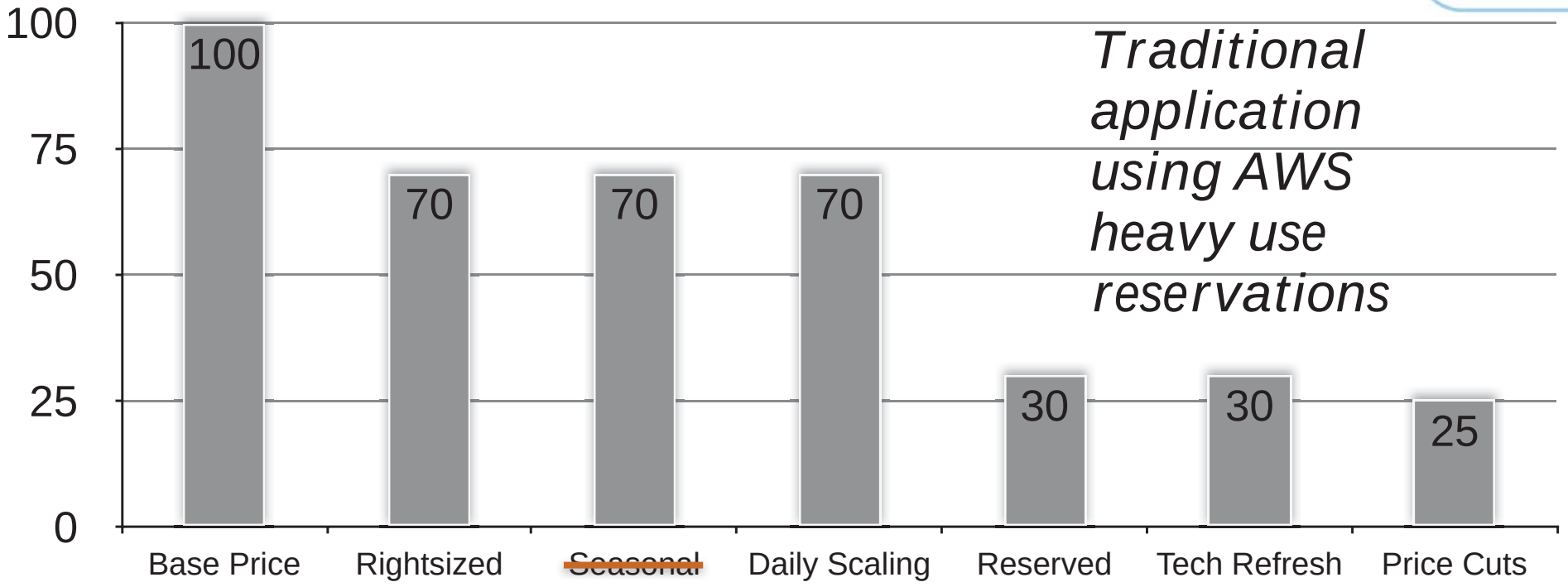
Lift and Shift Compounding



Base price is for capacity bought up-front

@adrianco
BV
Battery Ventures

Lift and Shift Compounding

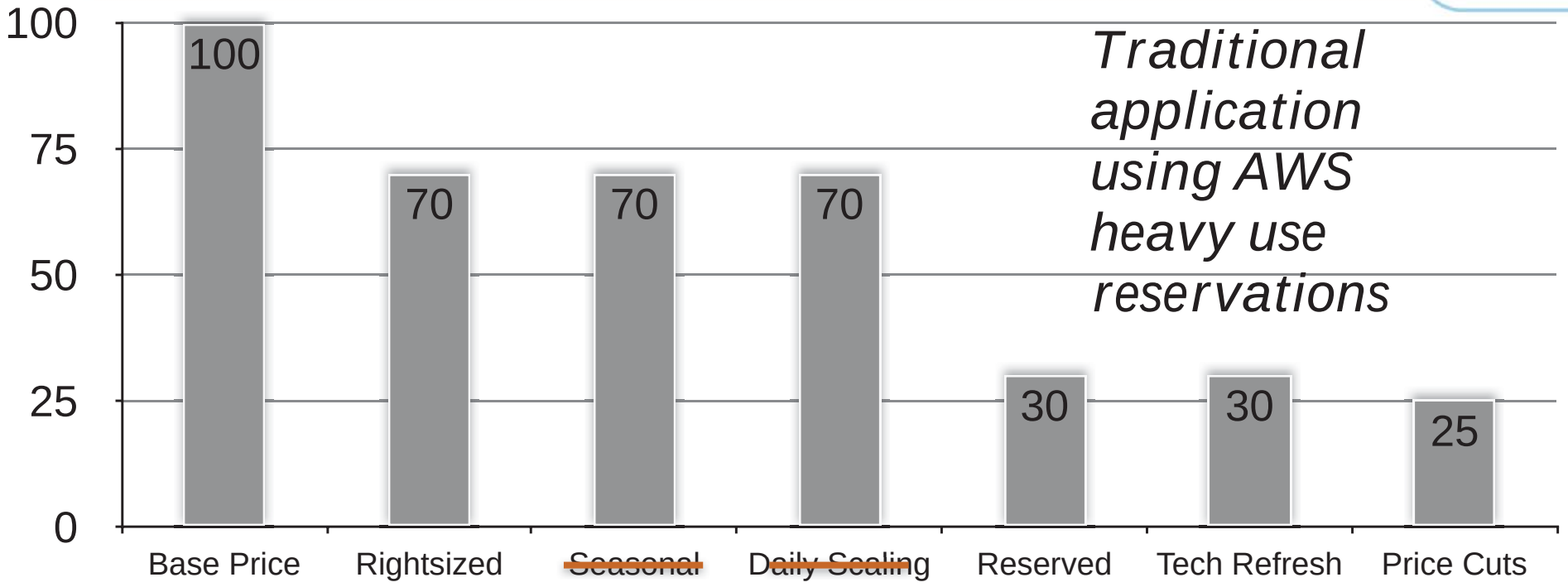


Base price is for capacity bought up-front

@adrianco

BV Battery Ventures

Lift and Shift Compounding

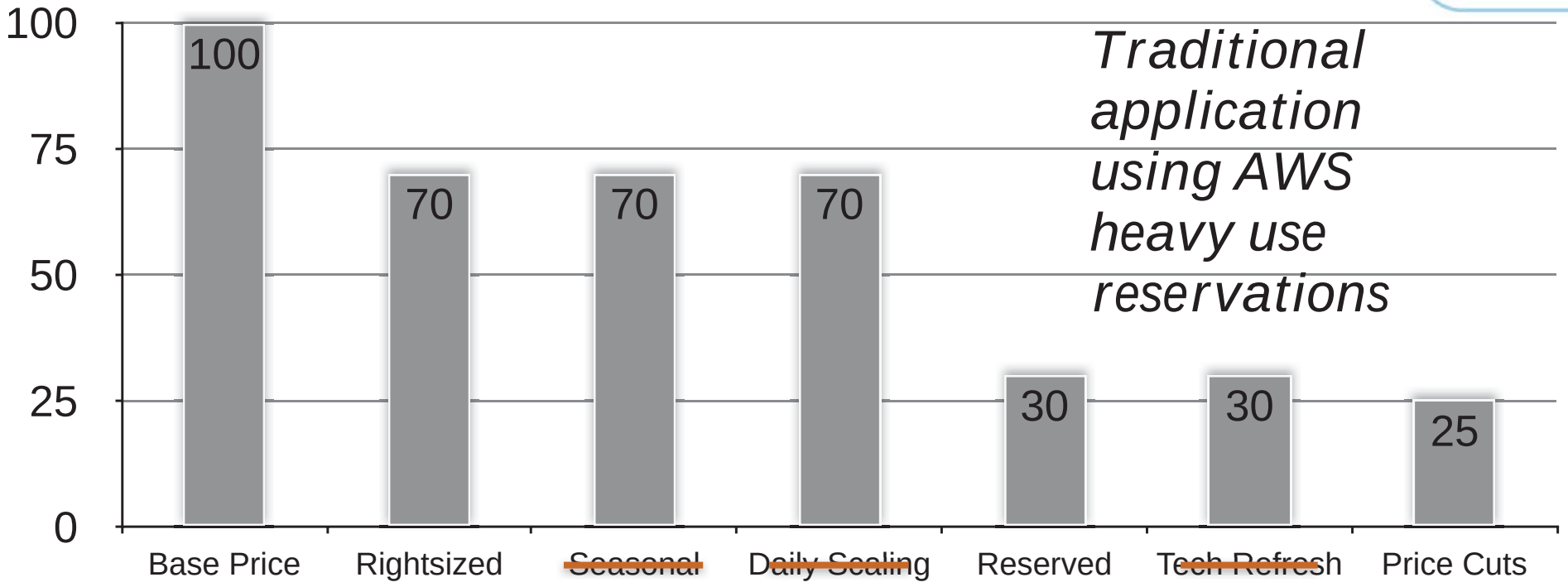


Base price is for capacity bought up-front

@adrianco

BV Battery Ventures

Lift and Shift Compounding

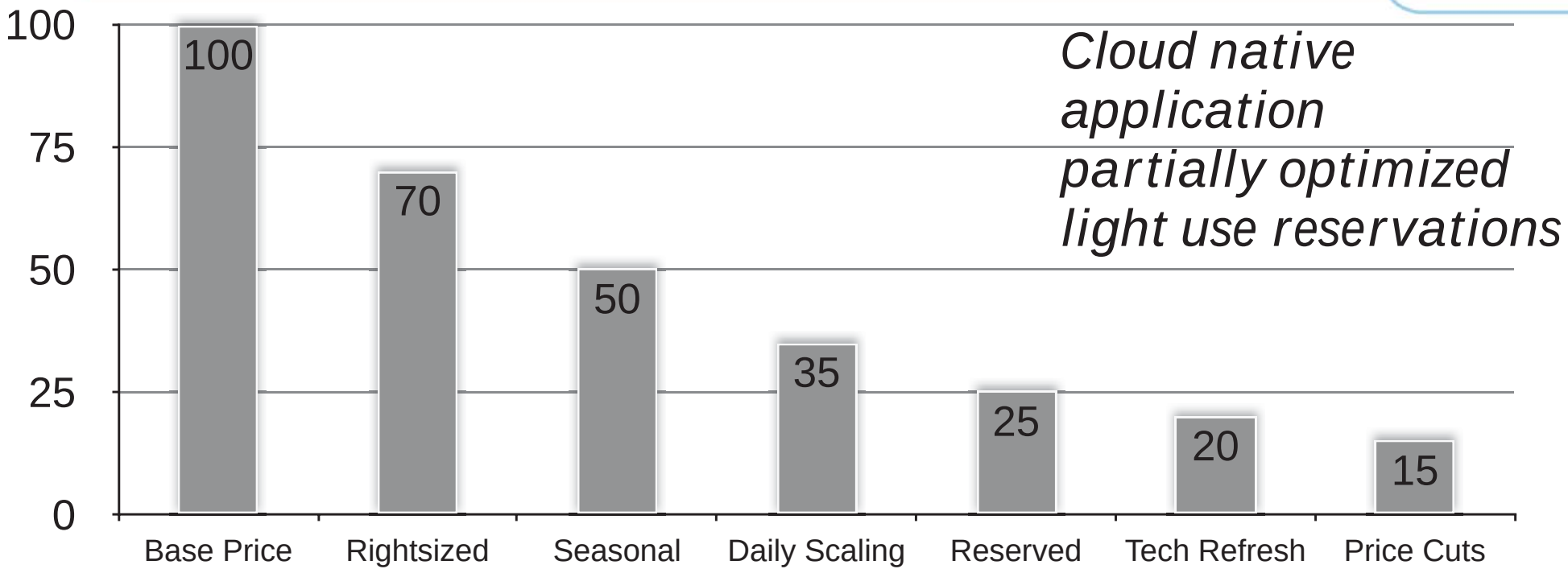


Base price is for capacity bought up-front

@adrianco

BV Battery Ventures

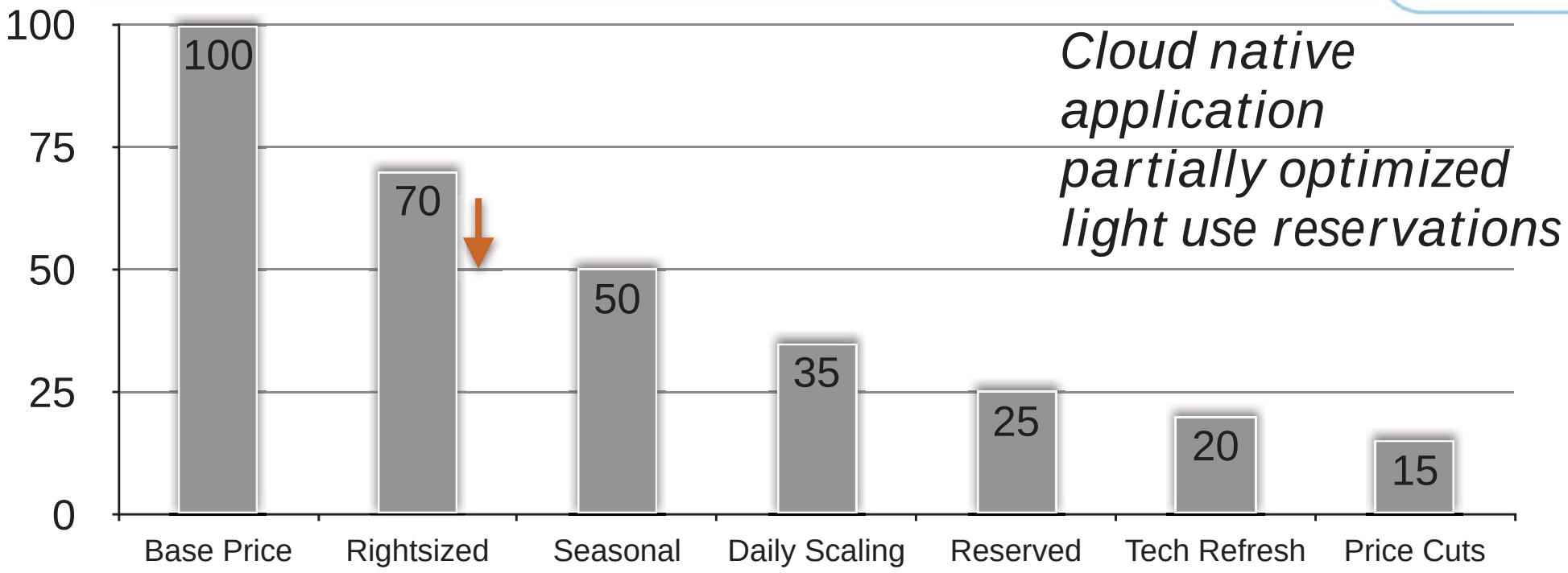
Conservative Compounding



@adrianco

BV Battery Ventures

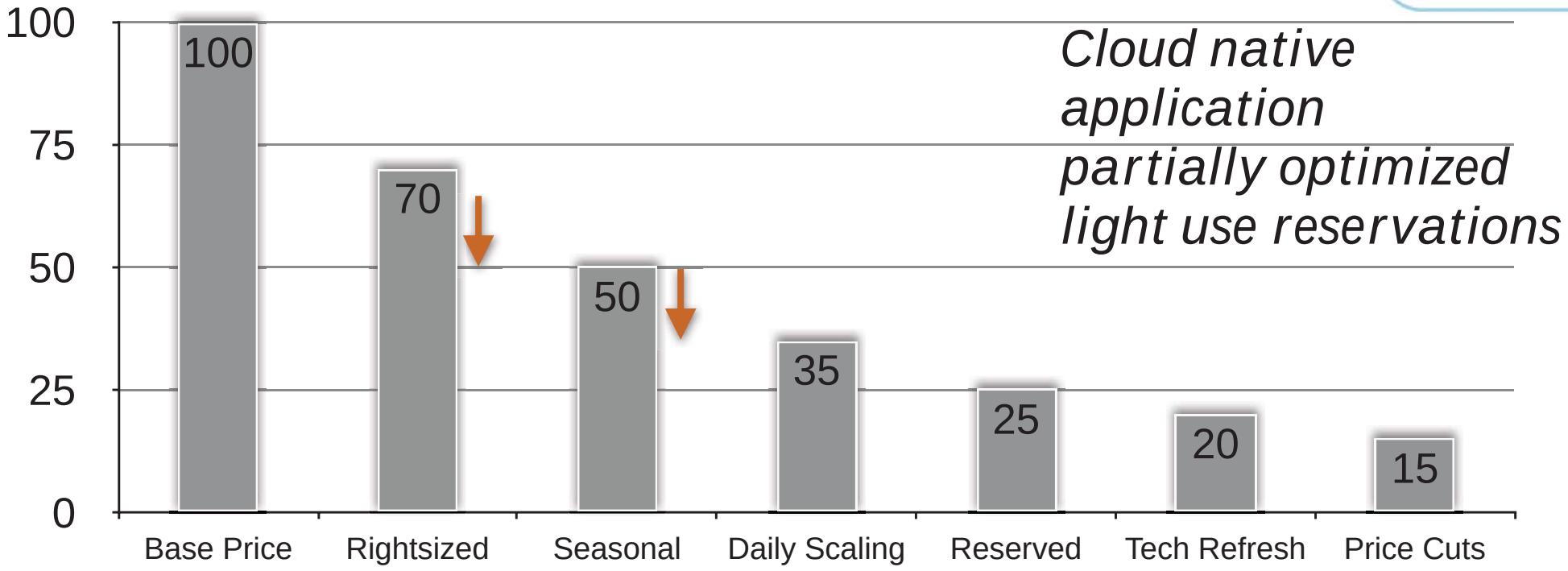
Conservative Compounding



@adrianco

BV Battery Ventures

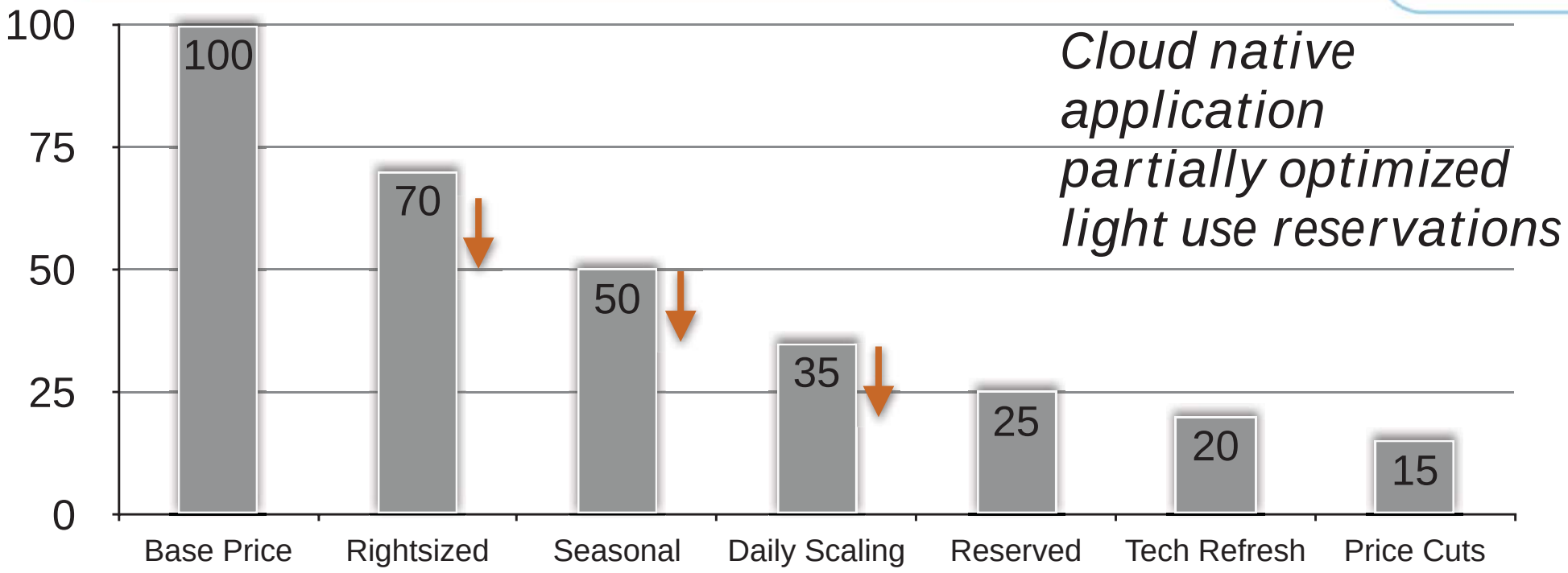
Conservative Compounding



@adrianco

BV Battery Ventures

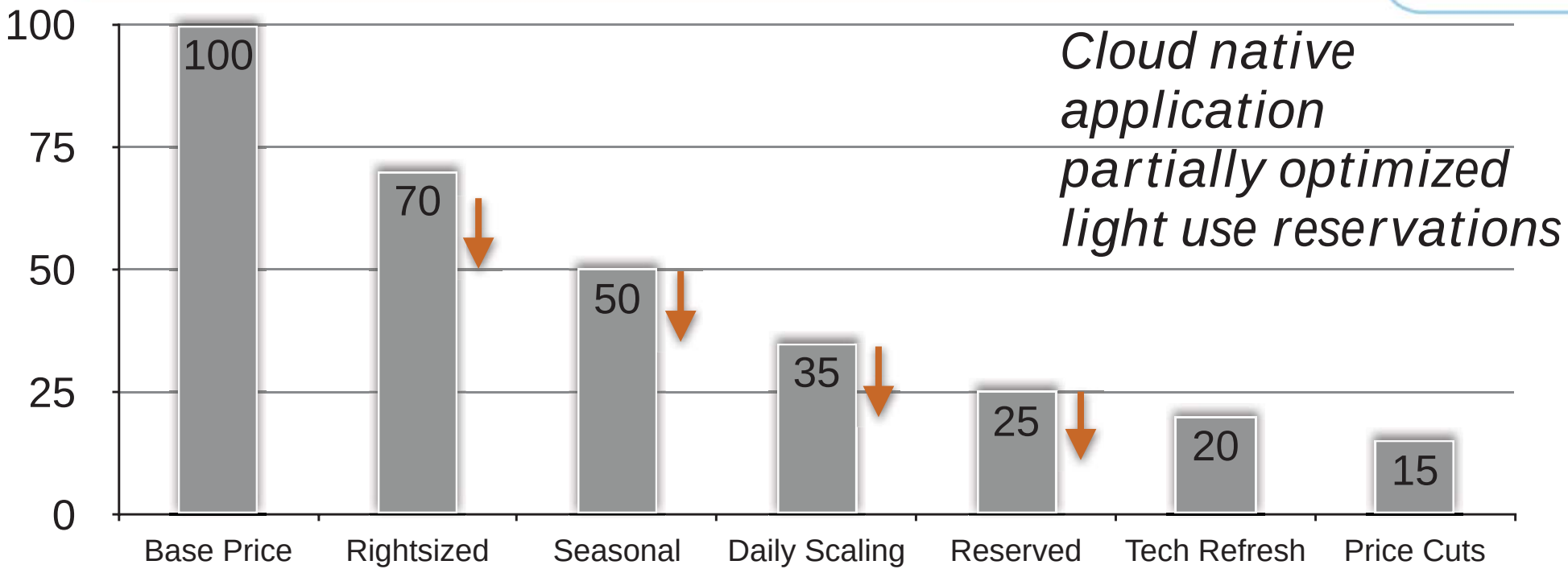
Conservative Compounding



@adrianco

BV Battery Ventures

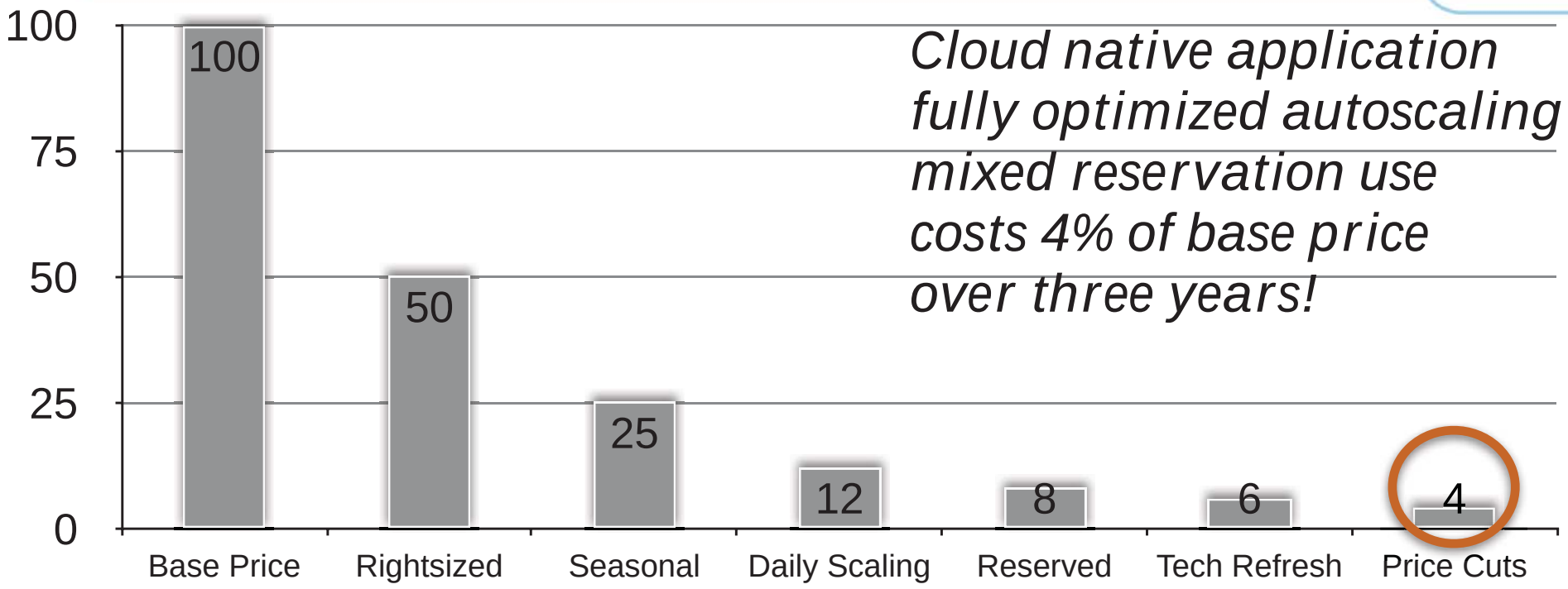
Conservative Compounding



@adrianco

BV Battery Ventures

Agressive Compounding

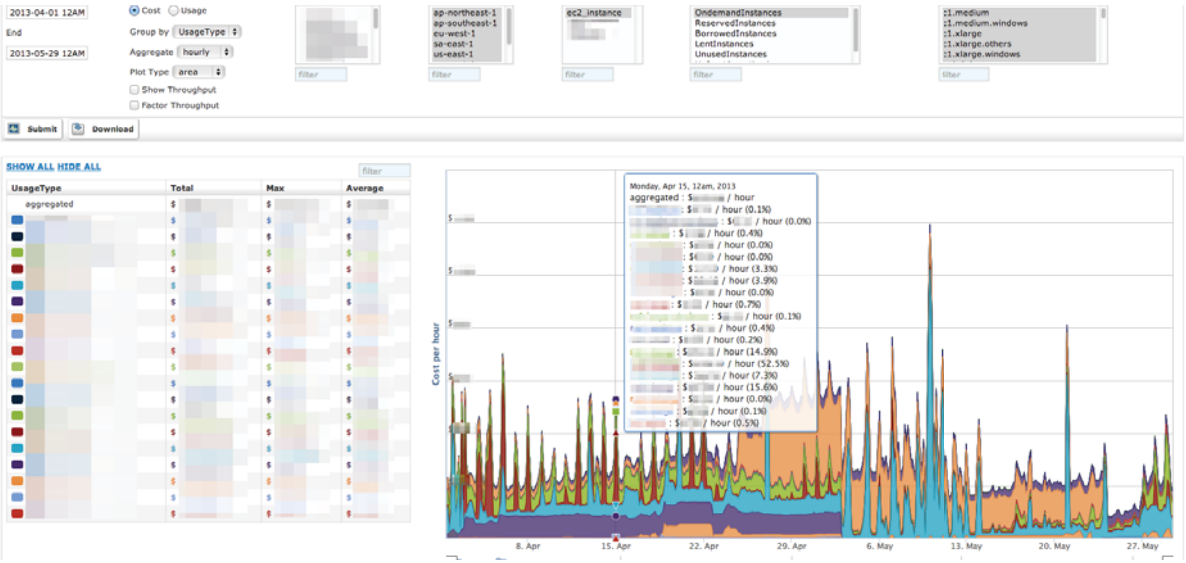


Cloud native application fully optimized autoscaling mixed reservation use costs 4% of base price over three years!

@adrianco

BV Battery Ventures

Cost Monitoring and Optimization



Cloud Options

cloudability

krystallize

cedexis

@adrianco

BV Battery Ventures

Final Thoughts

Turn off idle instances

Clean up unused stuff

Optimize for pricing model

Assume prices will go down

Go cloud native to be fast and save

Complex dynamic control issues!

Any Questions?

- *Battery Ventures http://www.battery.com*
- *Adrian's Tweets **@adrianco** and Blog http://perfcap.blogspot.com*
- *Slideshare http://slideshare.com/adriancockcroft*

- *Monitorama Opening Keynote Portland OR - May 7th, 2014*
- *GOTO Chicago Opening Keynote May 20th, 2014*
- *Qcon New York – Speed and Scale - June 11th, 2014*
- *Structure - Cloud Trends - San Francisco - June 19th, 2014*
- *GOTO Copenhagen/Aarhus – Fast Delivery - Denmark – Sept 25th, 2014*
- *DevOps Enterprise Summit - San Francisco - Oct 21-23rd, 2014 #DOES14*
- *GOTO Berlin - Migrating to Microservices - Germany - Nov 6th, 2014*
- *AWS Re:Invent - Cloud Native Cost Optimization - Las Vegas - November 14th, 2014*
- *O'Reilly Software Architecture Conference - Fast Delivery - Boston March 16th 2015*
- *High Performance Transaction Systems Workshop - http://hpts.ws September 2015*

@adrianco

BV Battery Ventures