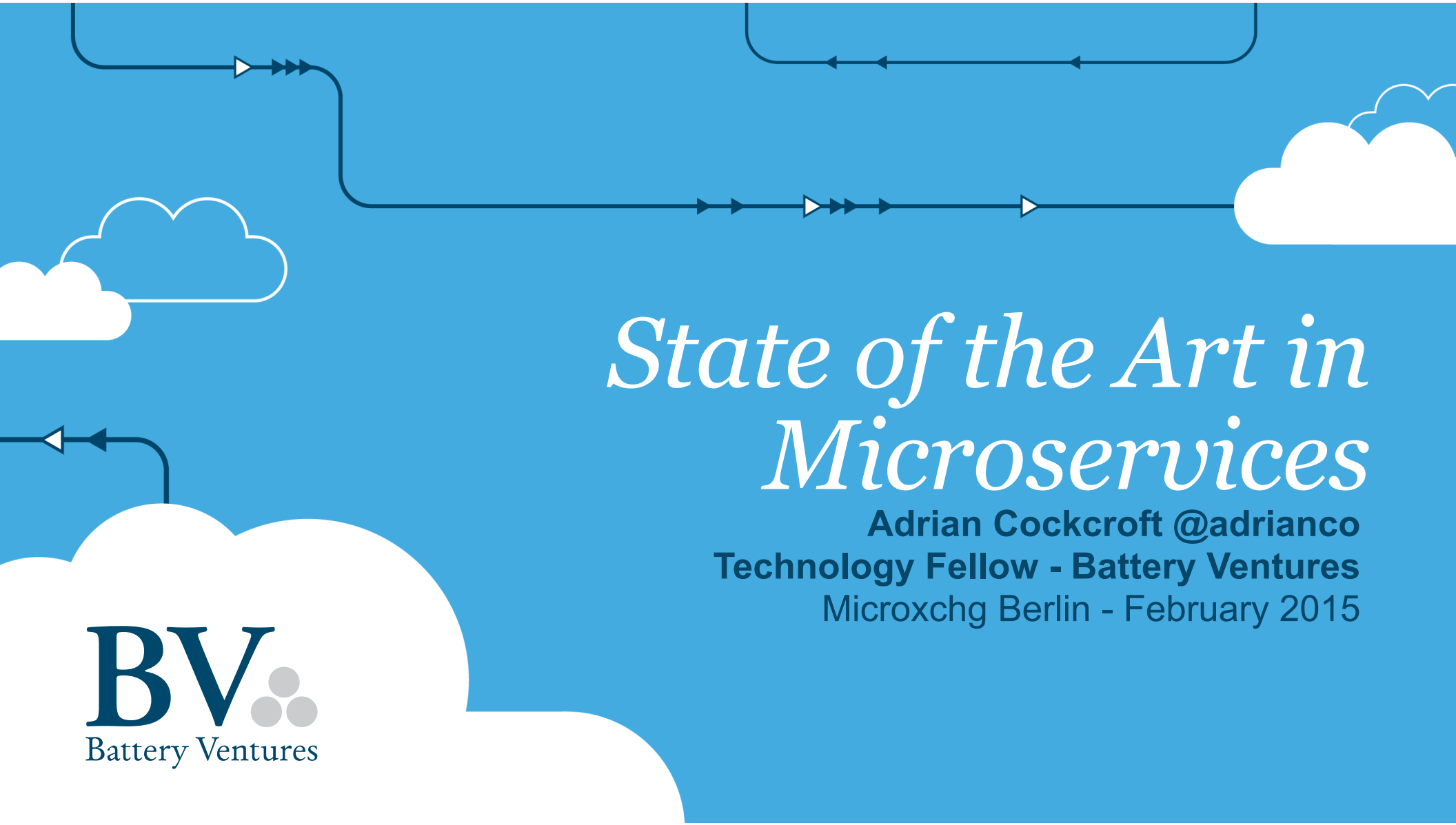
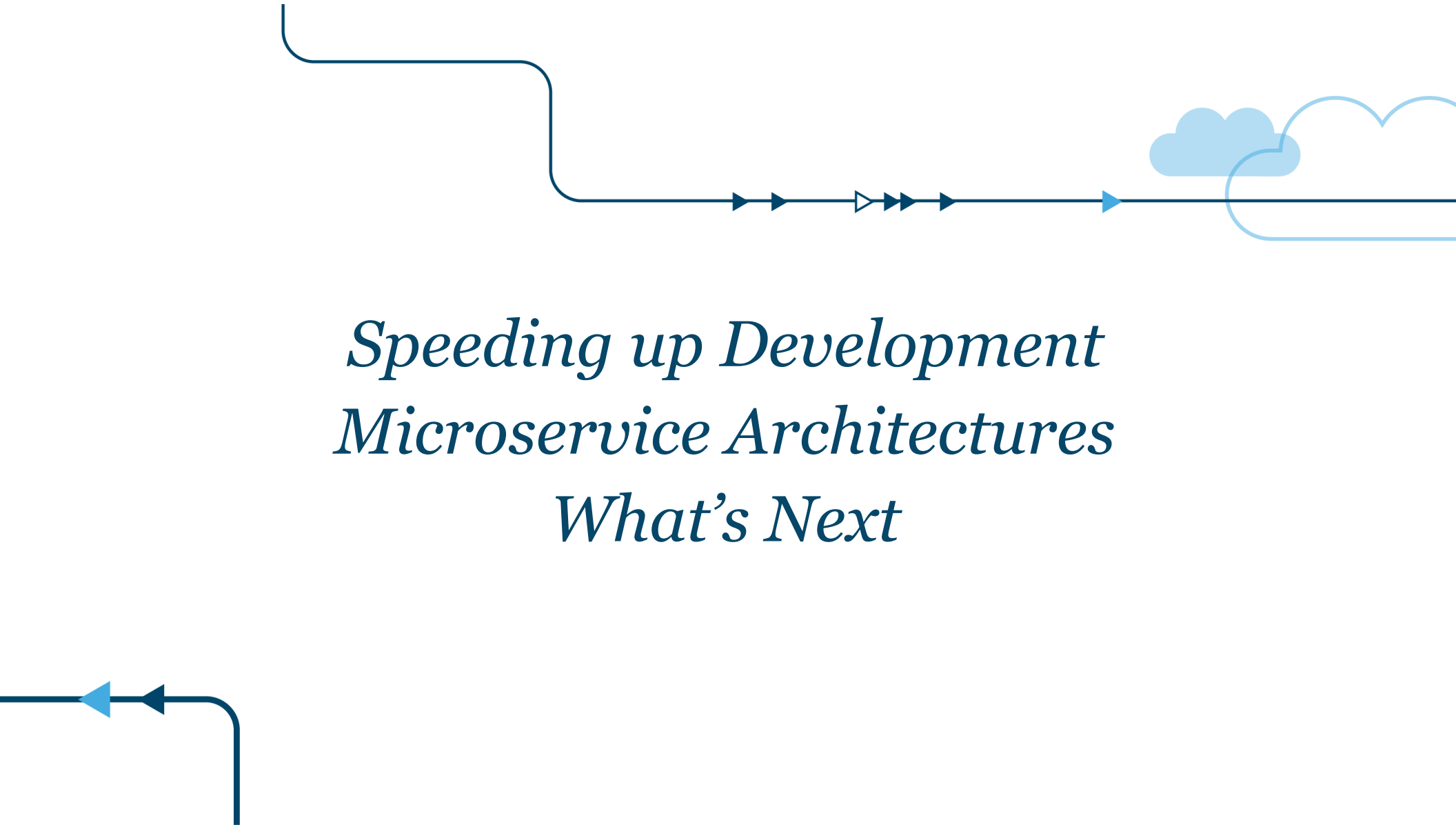


State of the Art in Microservices

Adrian Cockcroft @adrianco
Technology Fellow - Battery Ventures

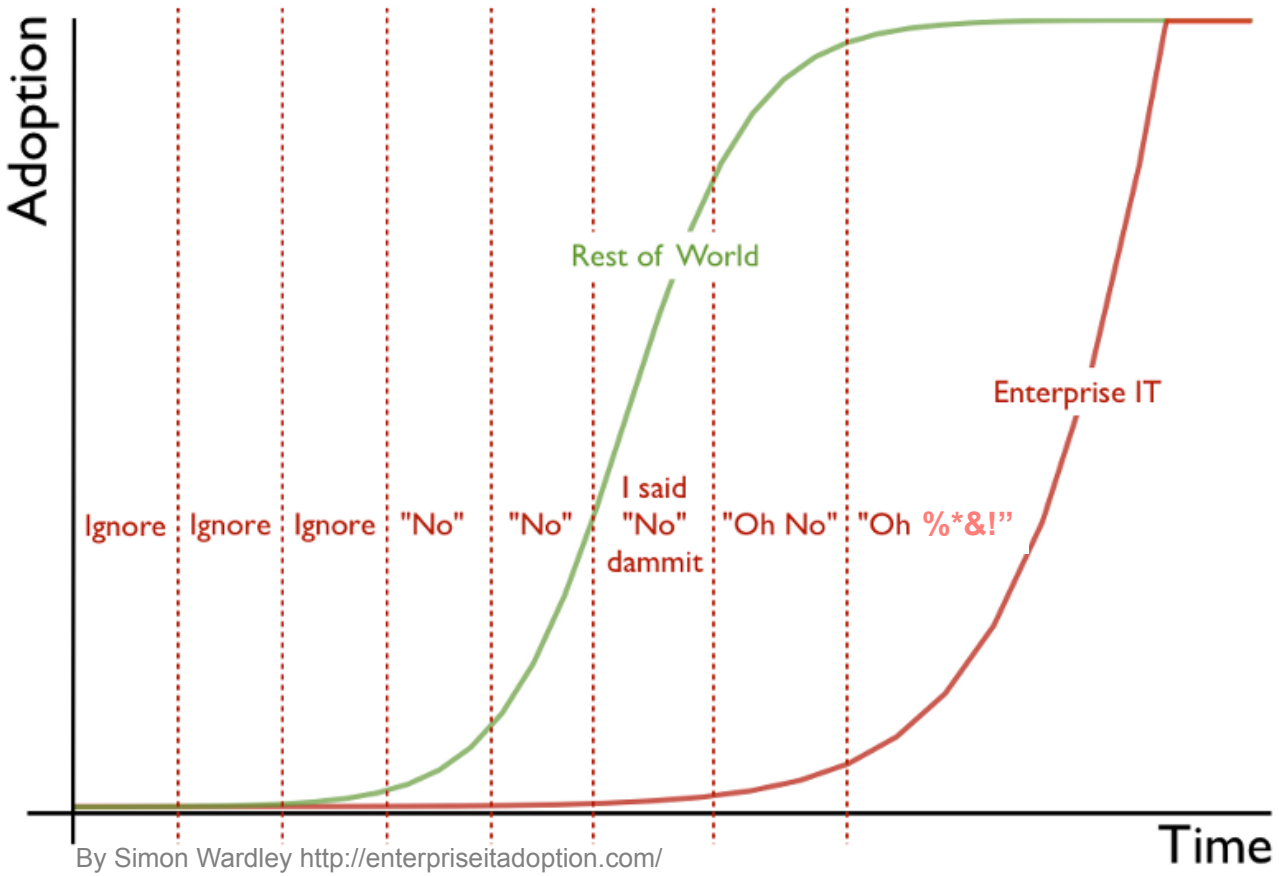
Microxchg Berlin - February 2015

BV
Battery Ventures



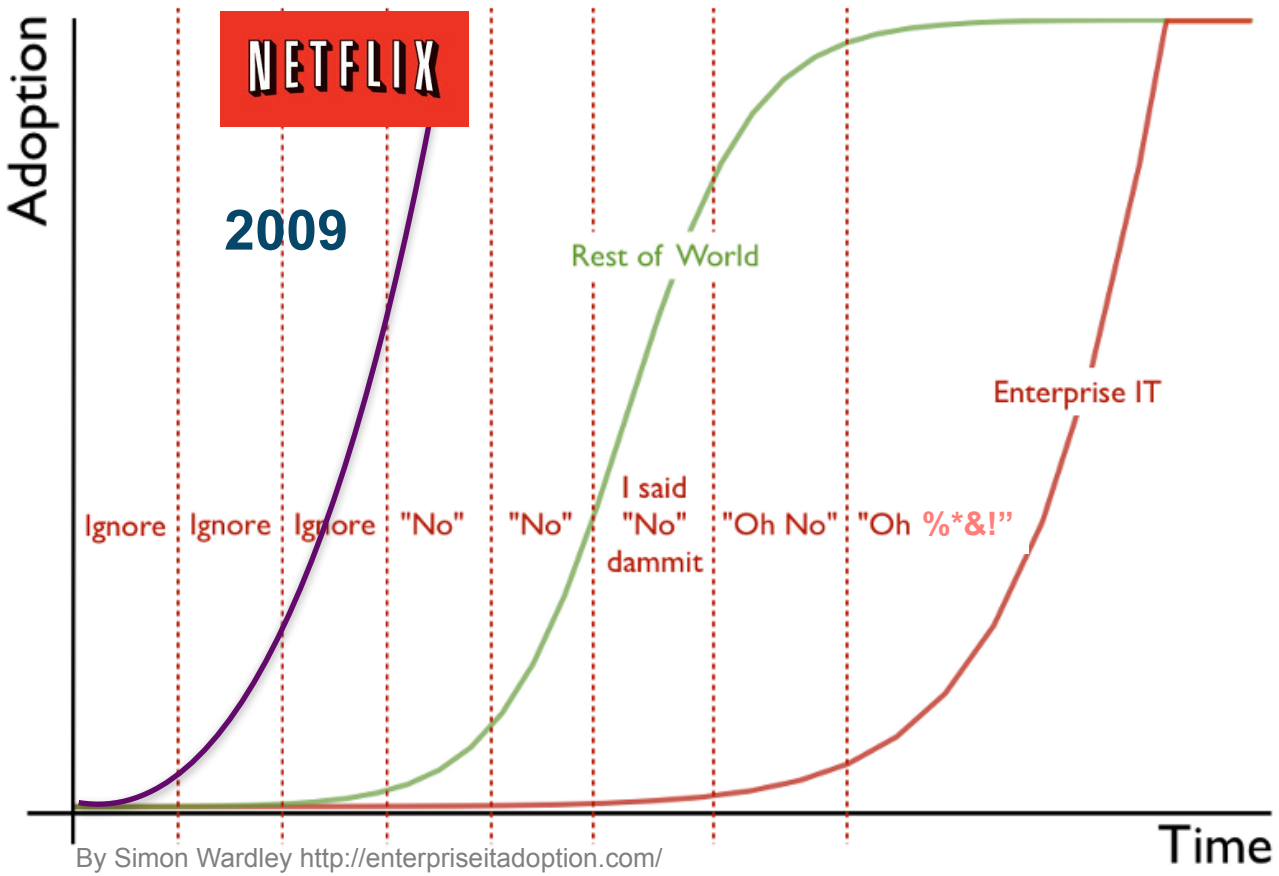
Speeding up Development
Microservice Architectures
What's Next

Why am I here?



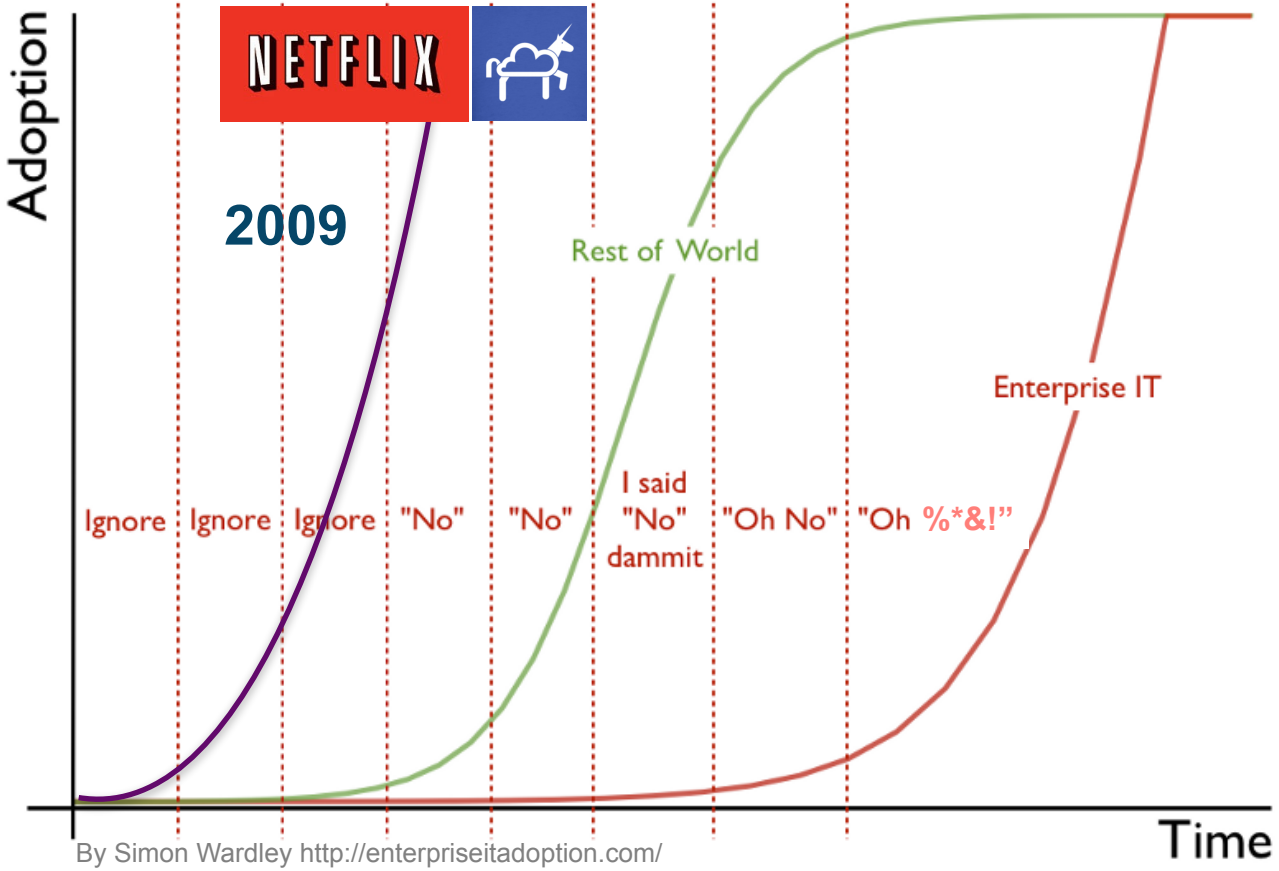
By Simon Wardley http://enterpriseitadoption.com/

Why am I here?

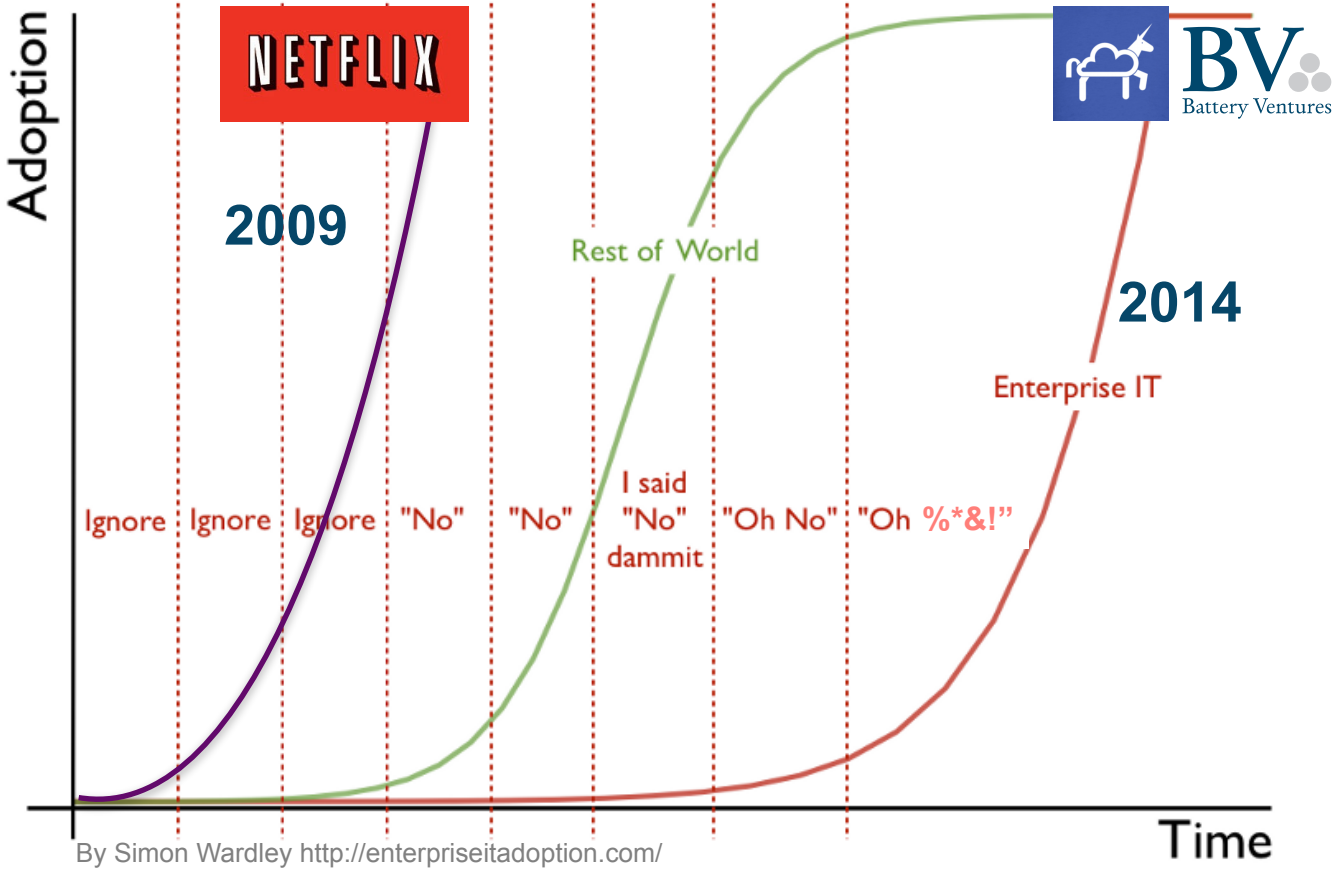


By Simon Wardley http://enterpriseitadoption.com/

Why am I here?

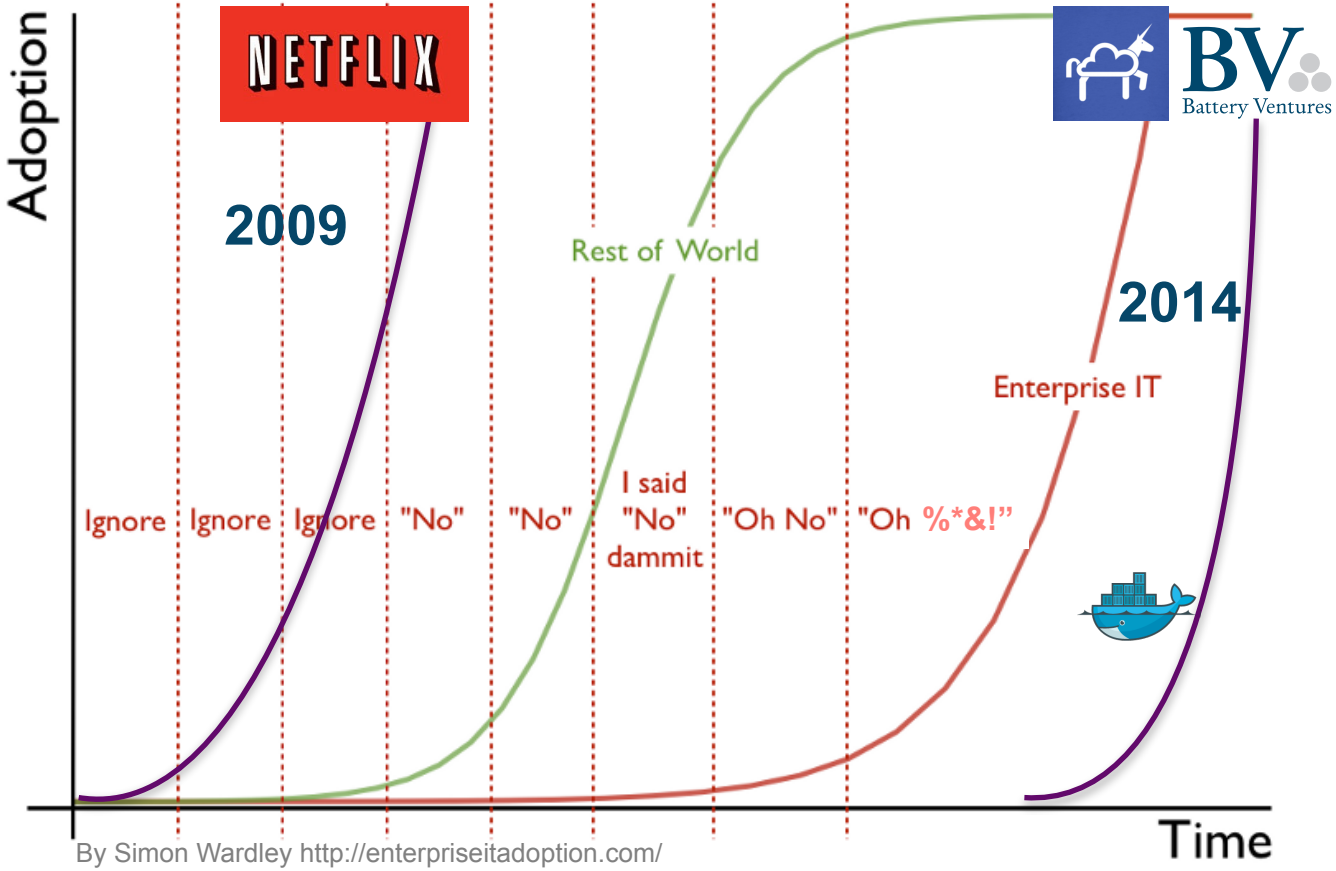

NETFLIX
2009
Adoption
Time
Rest of World
Enterprise IT
Ignore
Ignore
Ignore
"No"
"No"
I said "No" dammit
"Oh No"
"Oh %*&!"
By Simon Wardley http://enterpriseitadoption.com/

Why am I here?



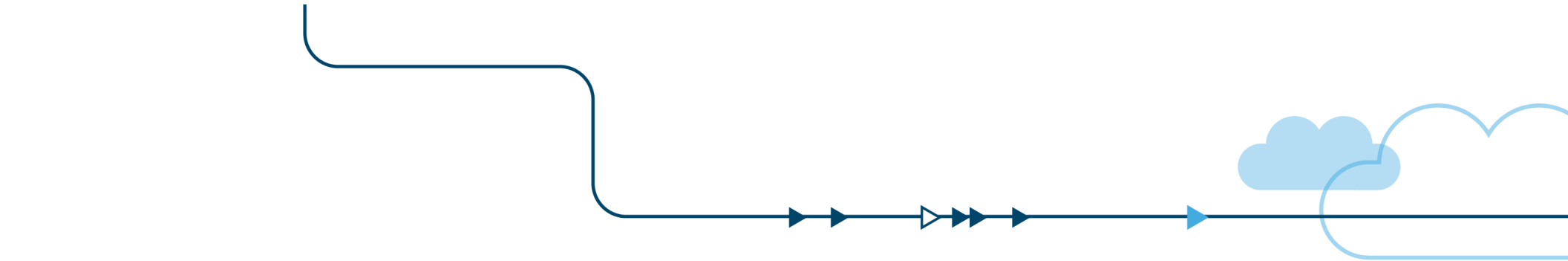
@adrianco's job at the intersection of cloud and Enterprise IT, looking for disruption and opportunities.

Why am I here?

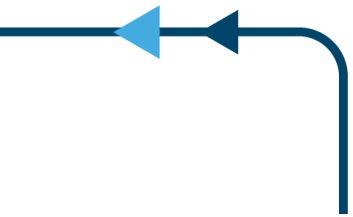


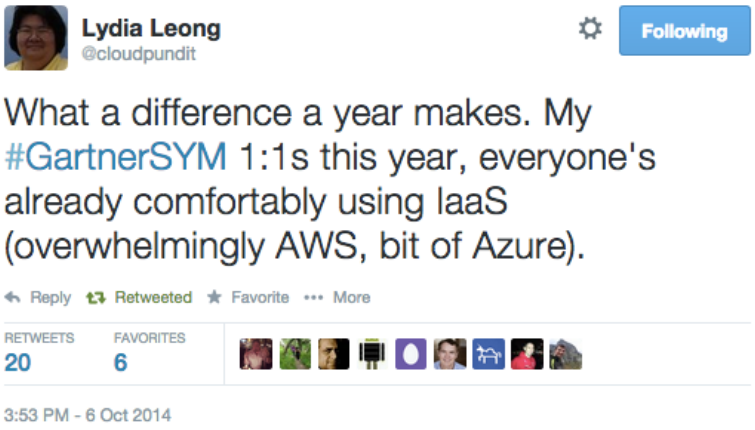
@adrianco's job at the intersection of cloud and Enterprise IT, looking for disruption and opportunities.

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

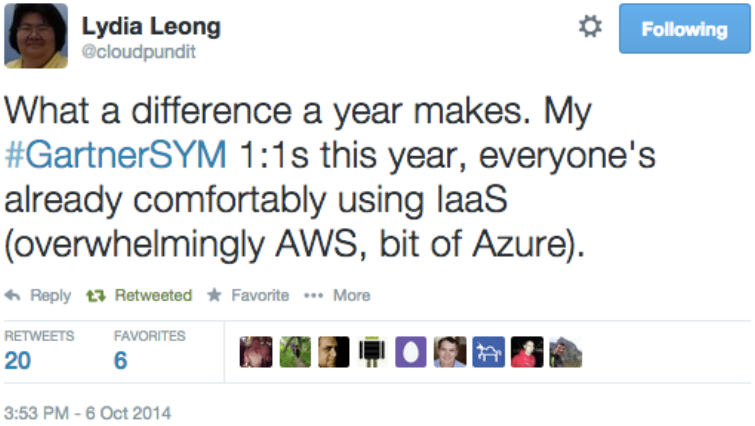


2014 was the year that Enterprises finally embraced cloud and DevOps.

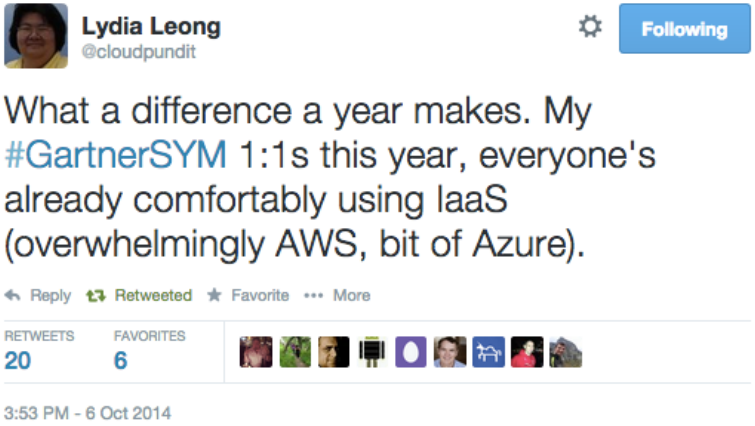




2014 was the year that Enterprises finally embraced cloud and DevOps.



2014 was the year that Enterprises finally embraced cloud and DevOps.



2014 was the year that Enterprises finally embraced cloud and DevOps.

What does @adrianco do?

Presentations at Conferences

Maintain Relationship with Cloud Vendors

Technology Due Diligence on Deals

@adrianco

BV Battery Ventures

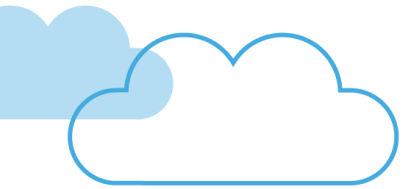
Presentations at Companies

Technical Advice for Portfolio Companies

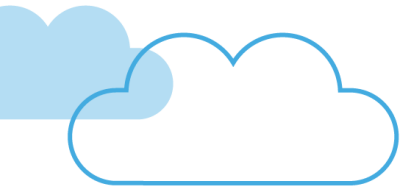
Program Committee for Conferences

Tinkering with Technologies

Networking with Interesting People

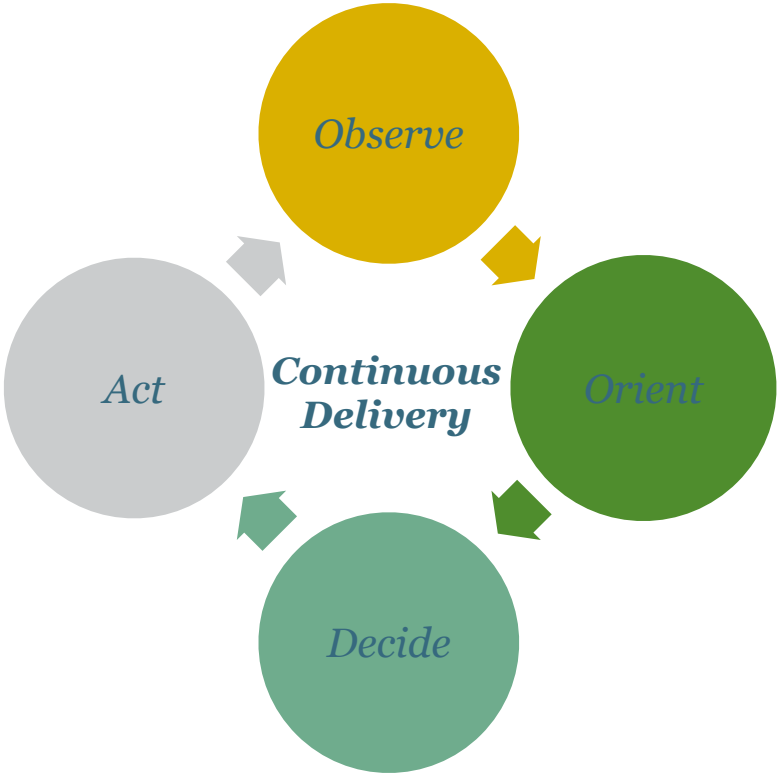


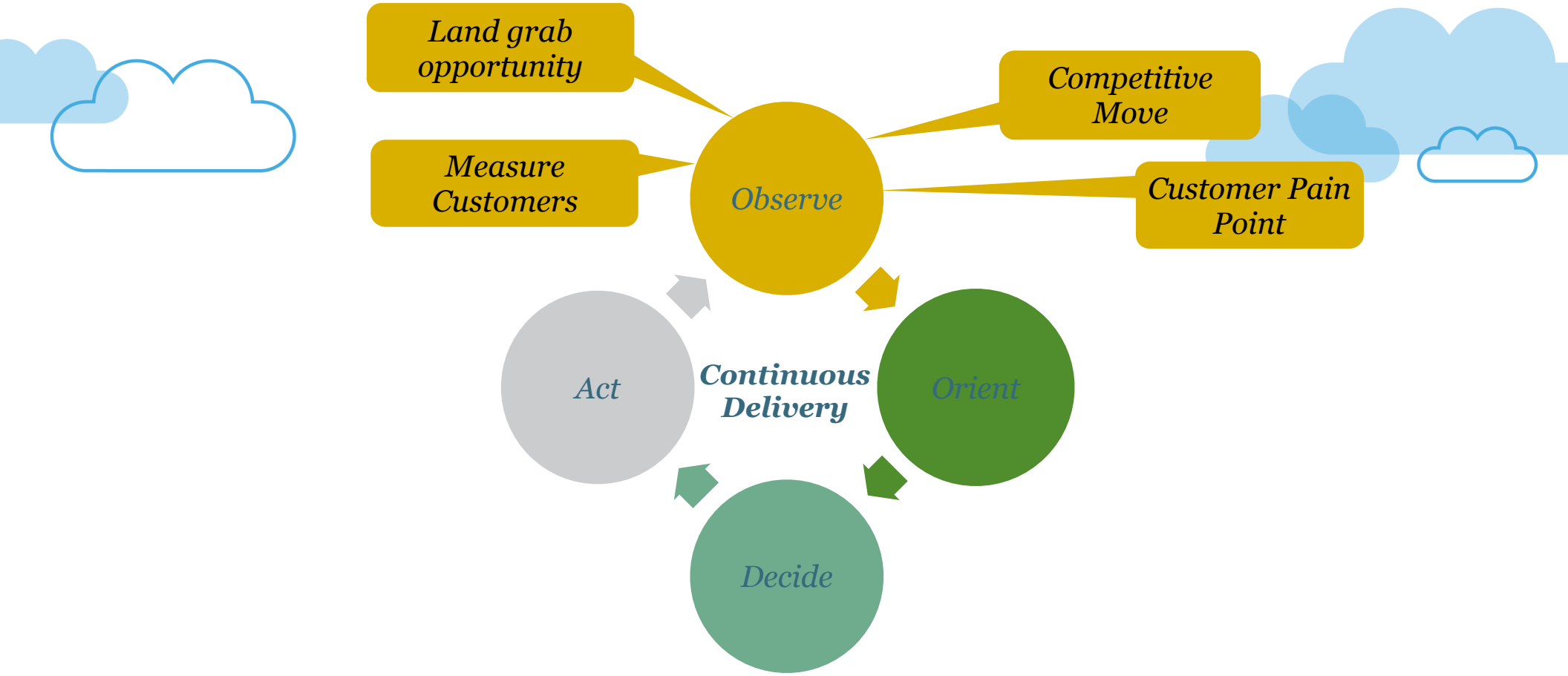
Product Development Processes

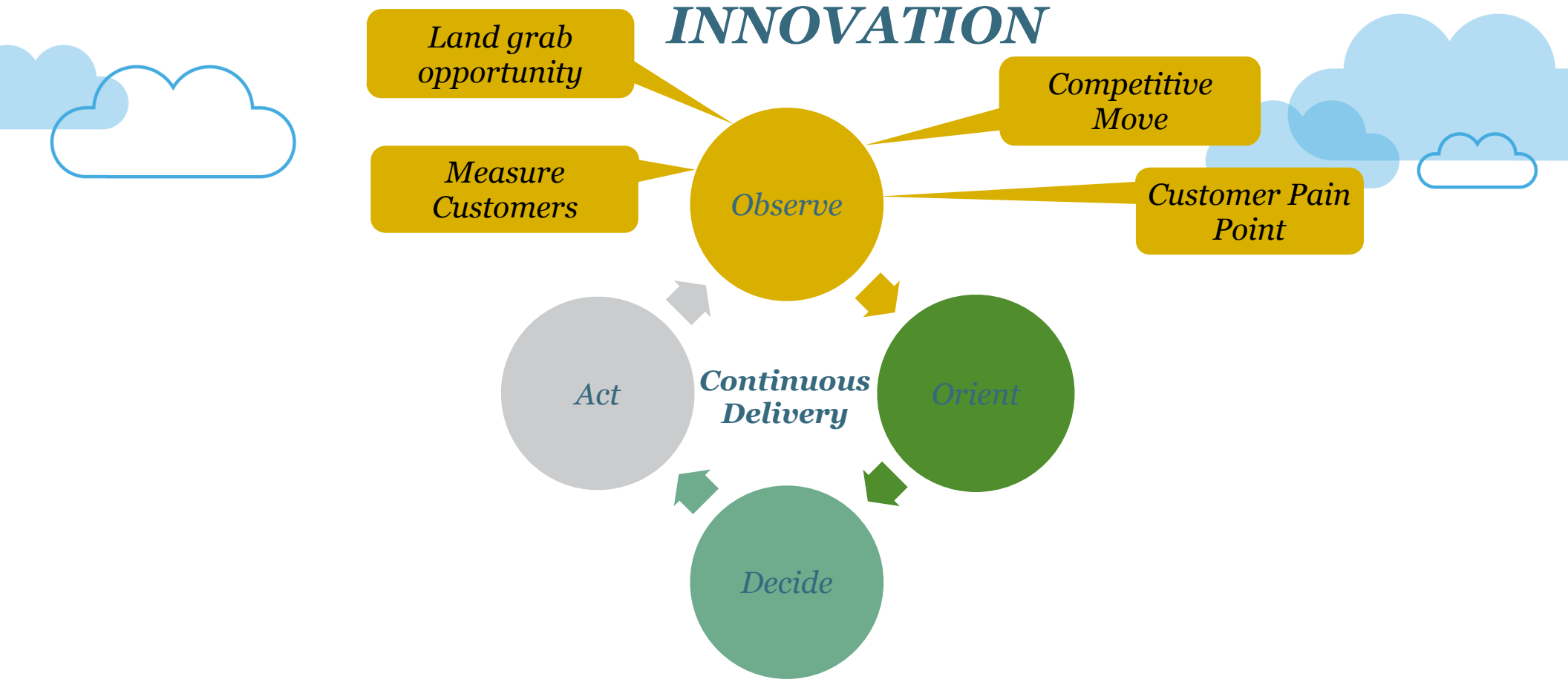


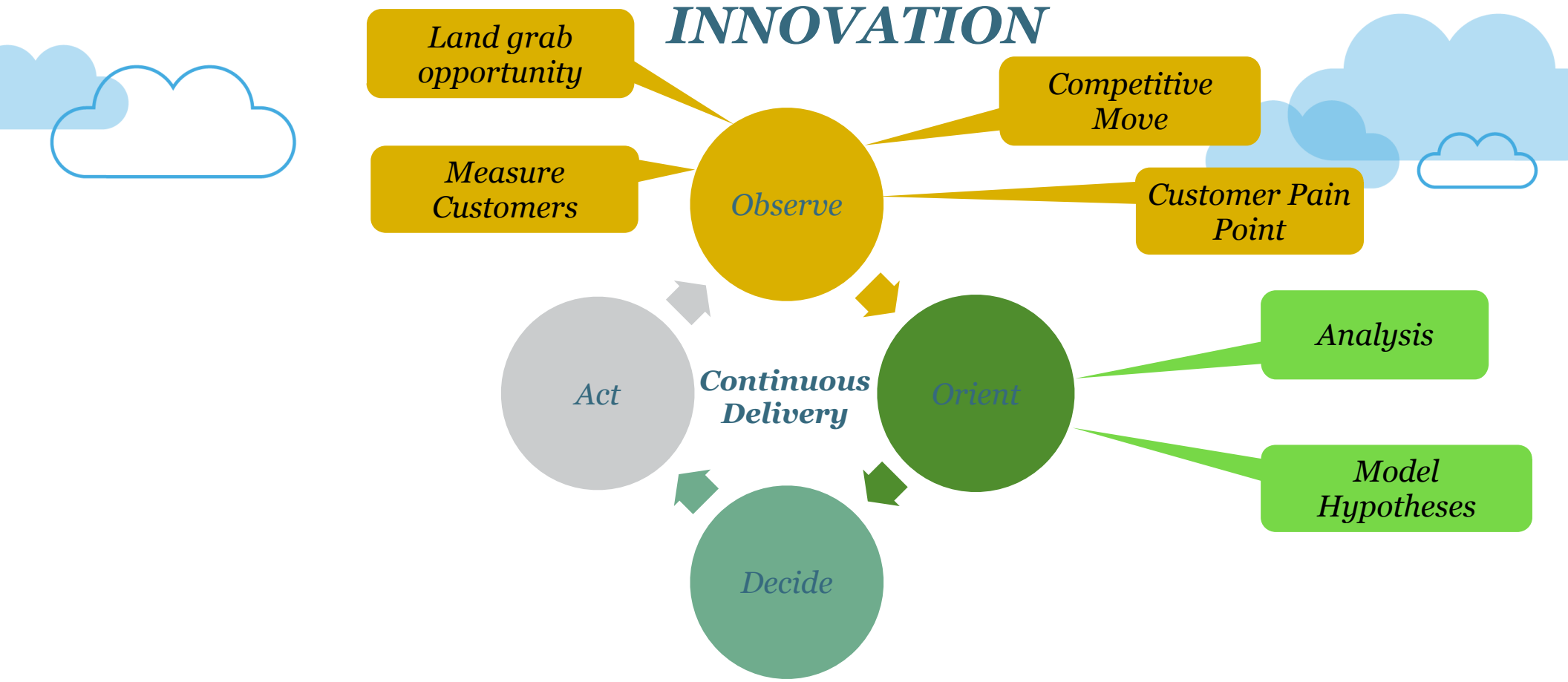

Assumption: Process prevents problems

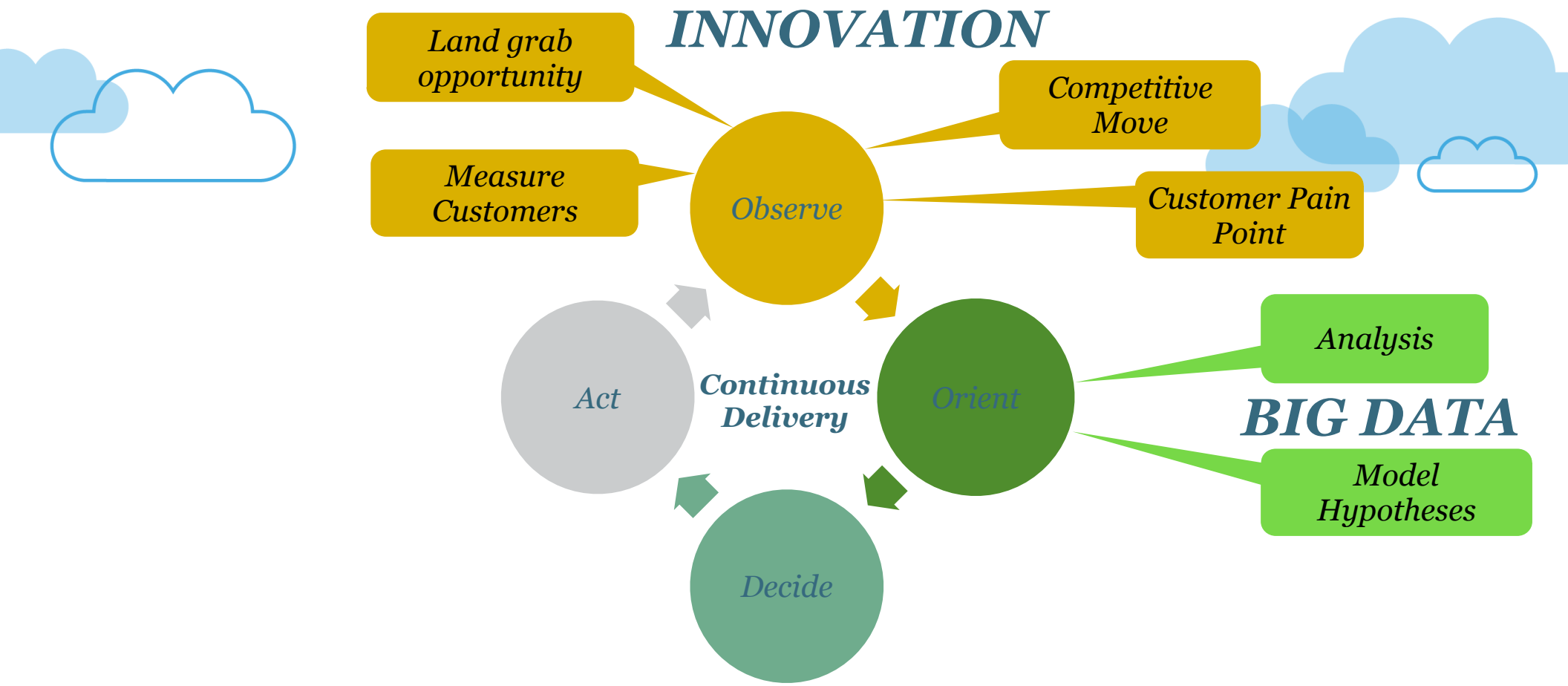
Organizations build up slow complex "Scar tissue" processes

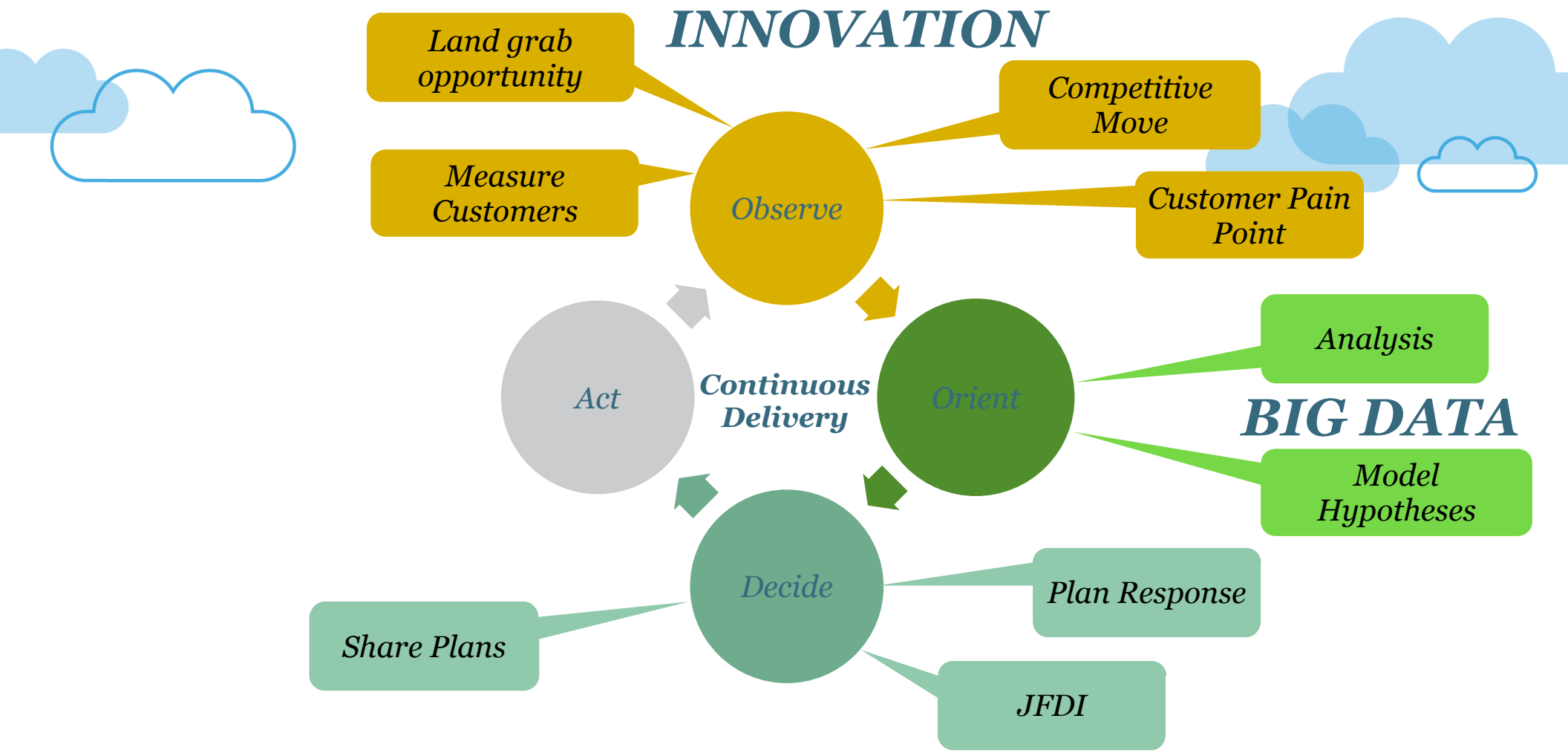

Observe
Orient
Decide
Act
Continuous Delivery

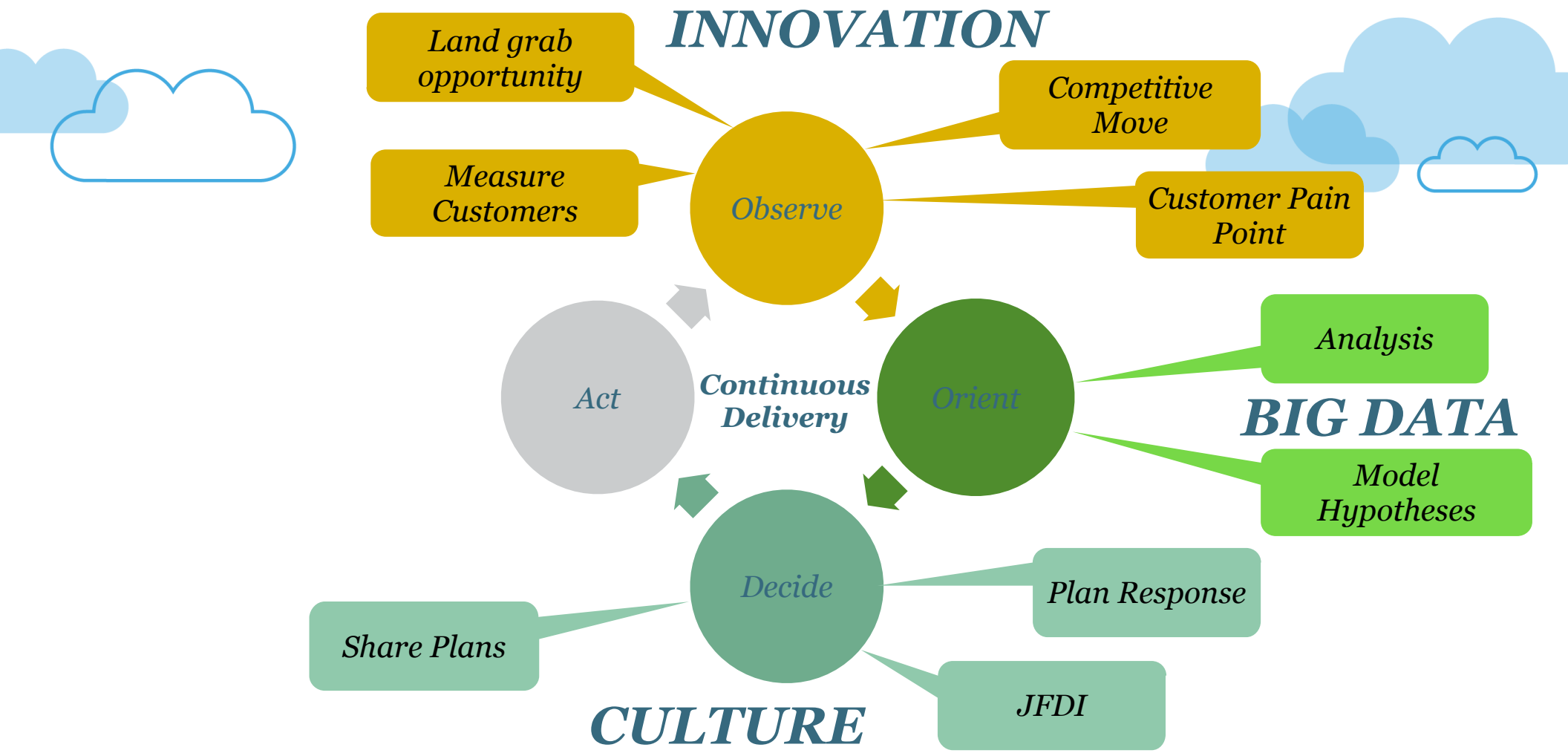

Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Act
Continuous Delivery
Orient
Decide

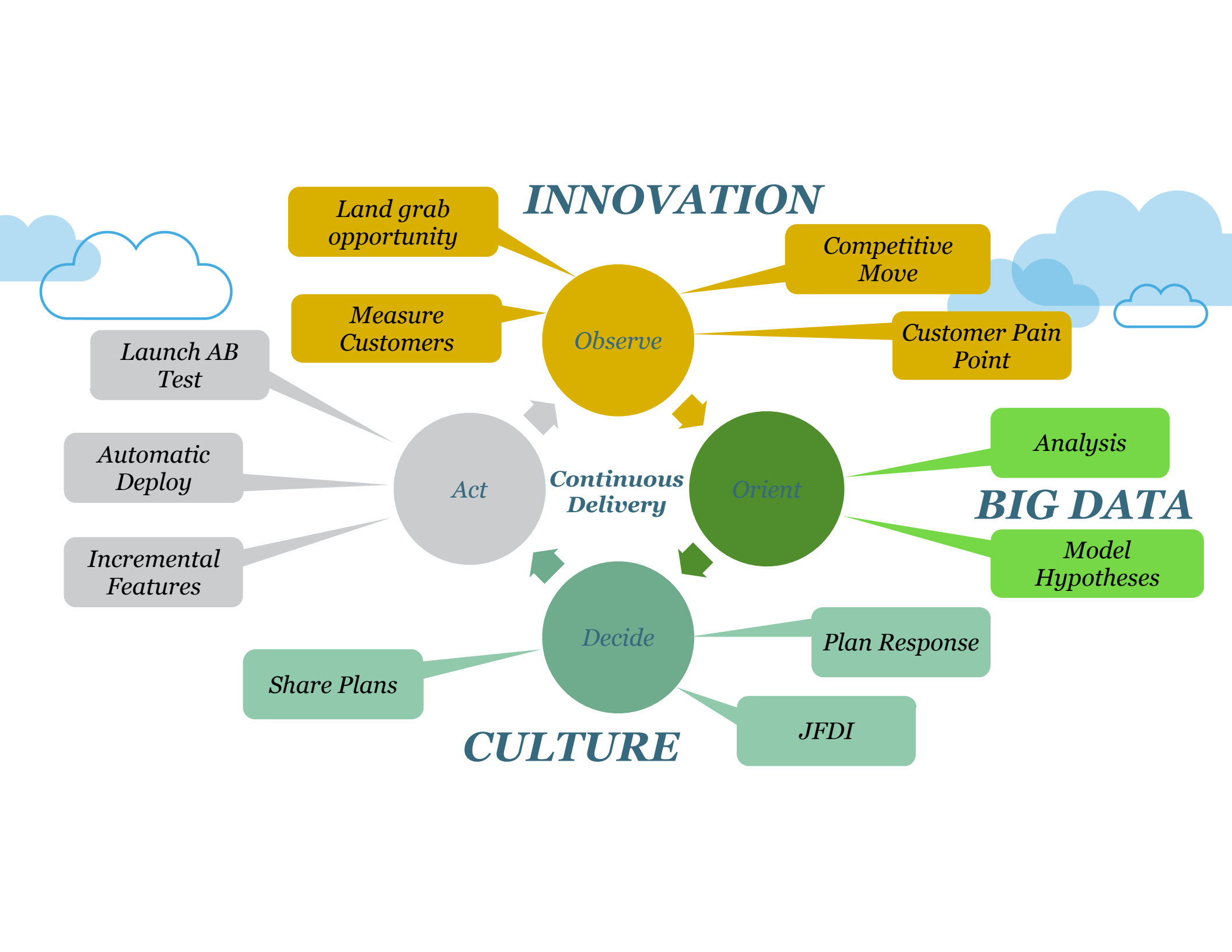

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Orient
Decide
Act
Continuous Delivery

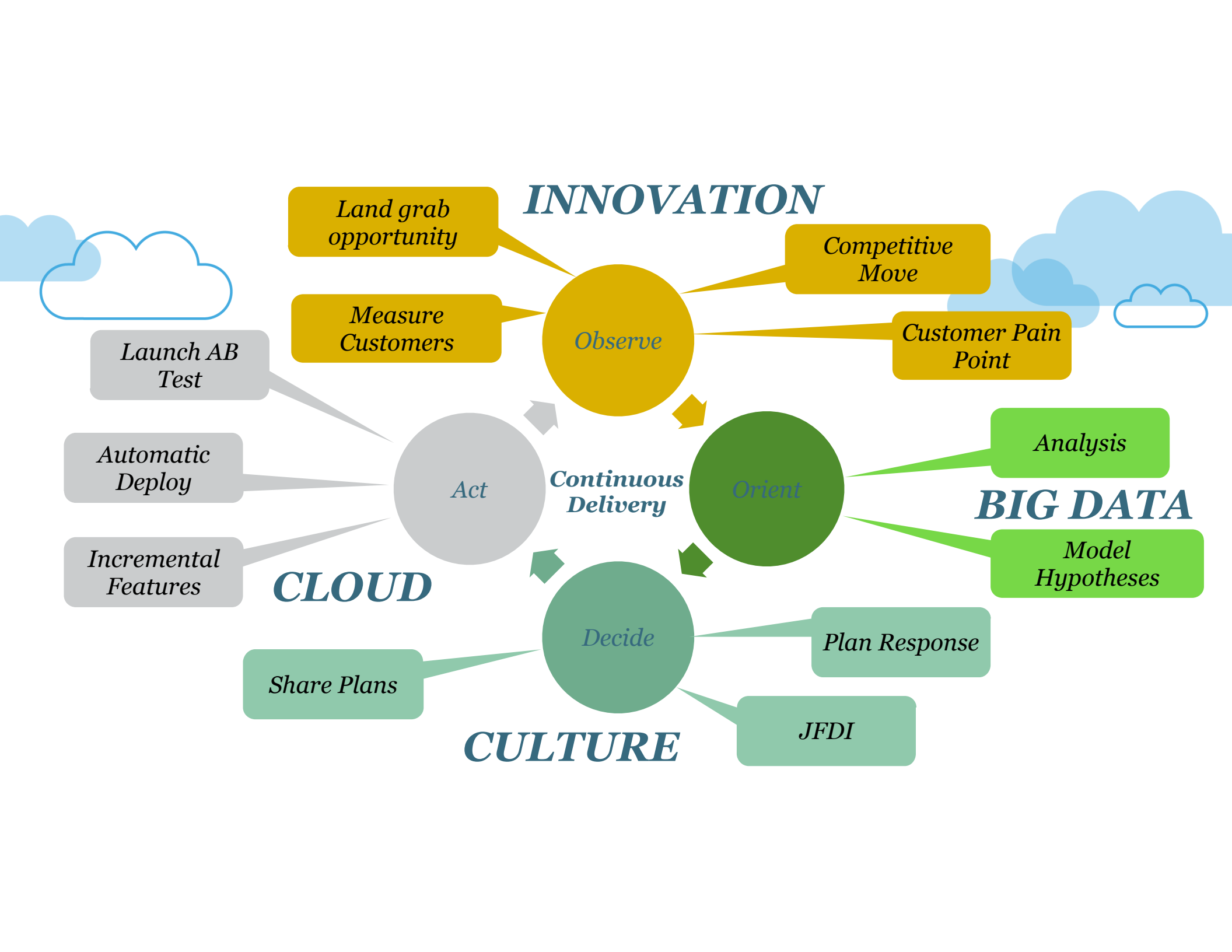

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Orient
Analysis
Model Hypotheses
Decide
Act
Continuous Delivery

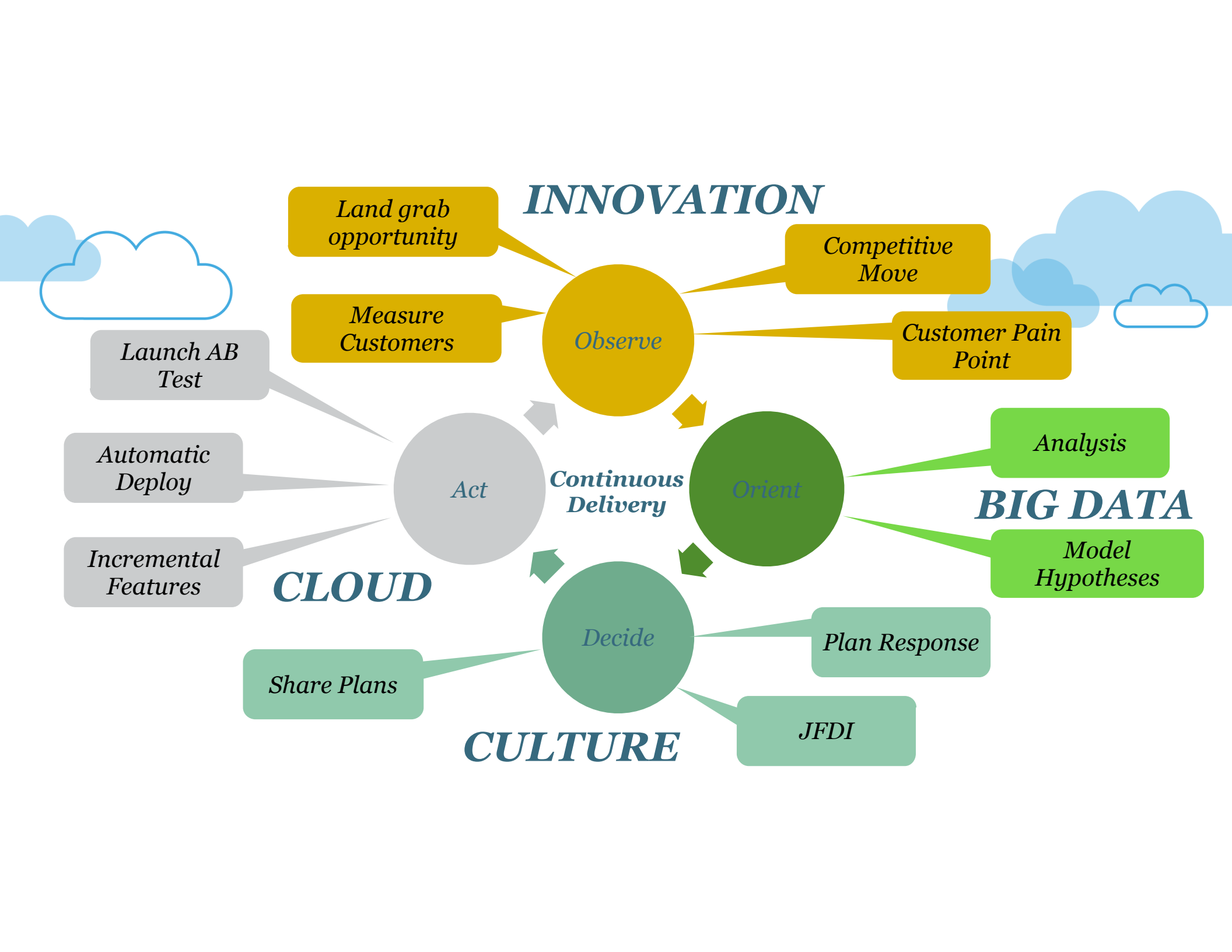

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Orient
Analysis
BIG DATA
Model Hypotheses
Decide
Act
Continuous Delivery

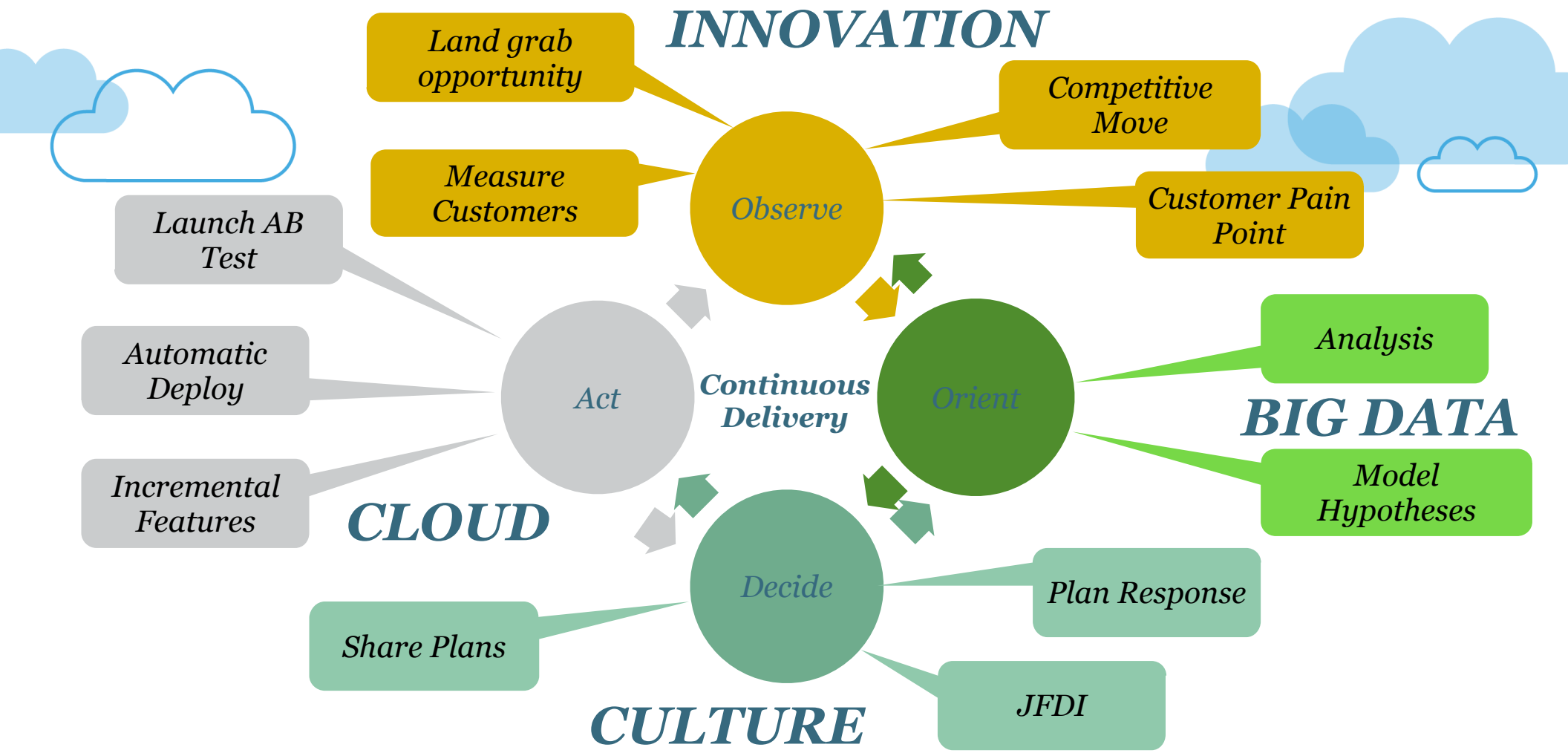

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Act
Continuous Delivery
Orient
Analysis
BIG DATA
Model Hypotheses
Decide
Share Plans
Plan Response
JFDI


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

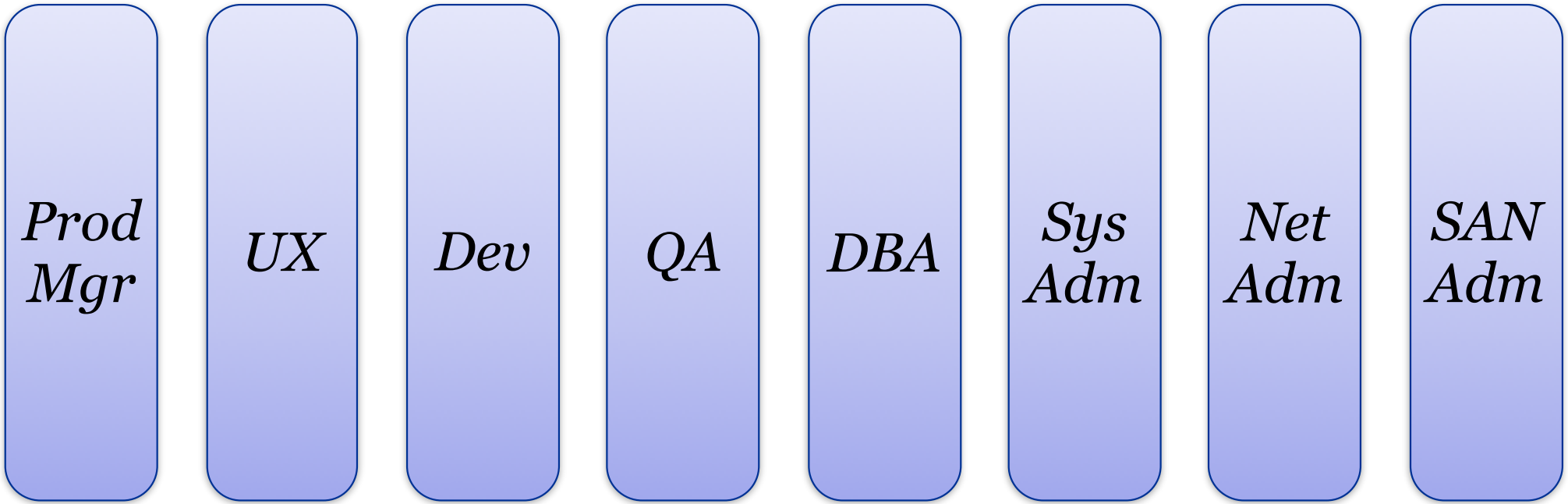

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Launch AB Test
Automatic Deploy
Incremental Features
Act
Continuous Delivery
Orient
Analysis
BIG DATA
Model Hypotheses
CLOUD
Decide
Plan Response
Share Plans
JFDI
CULTURE


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

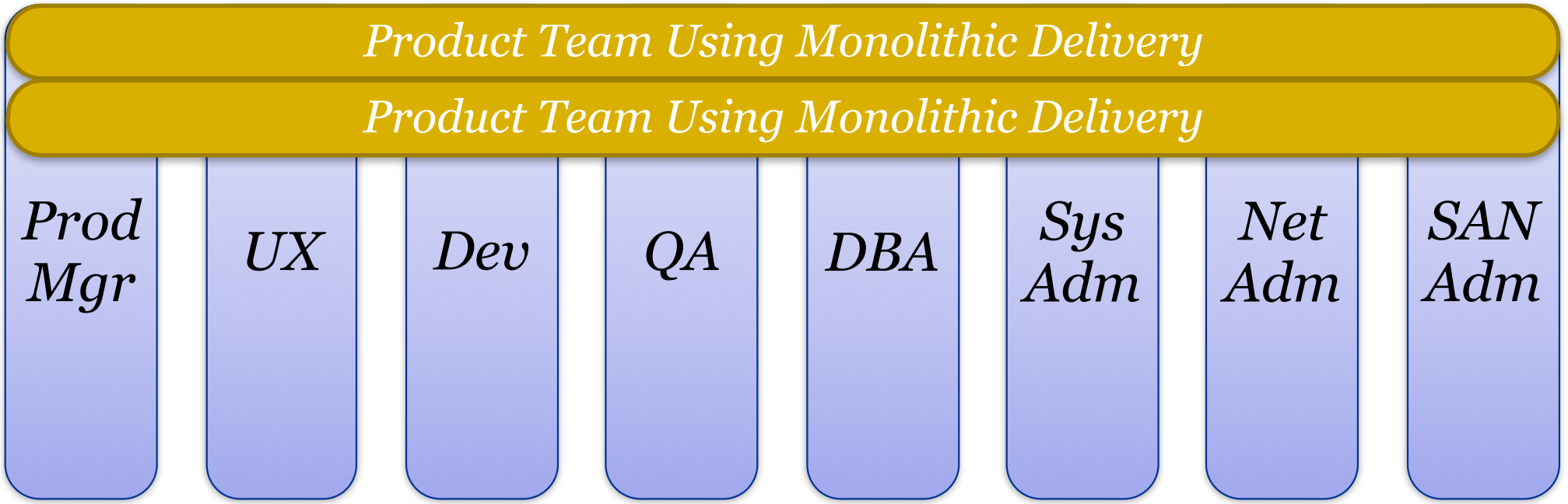

INNOVATION
Land grab opportunity
Measure Customers
Observe
Competitive Move
Customer Pain Point
Launch AB Test
Automatic Deploy
Incremental Features
Act
Continuous Delivery
Orient
Analysis
BIG DATA
Model Hypotheses
CLOUD
Decide
Plan Response
Share Plans
JFDI
CULTURE

Breaking Down the SILOs

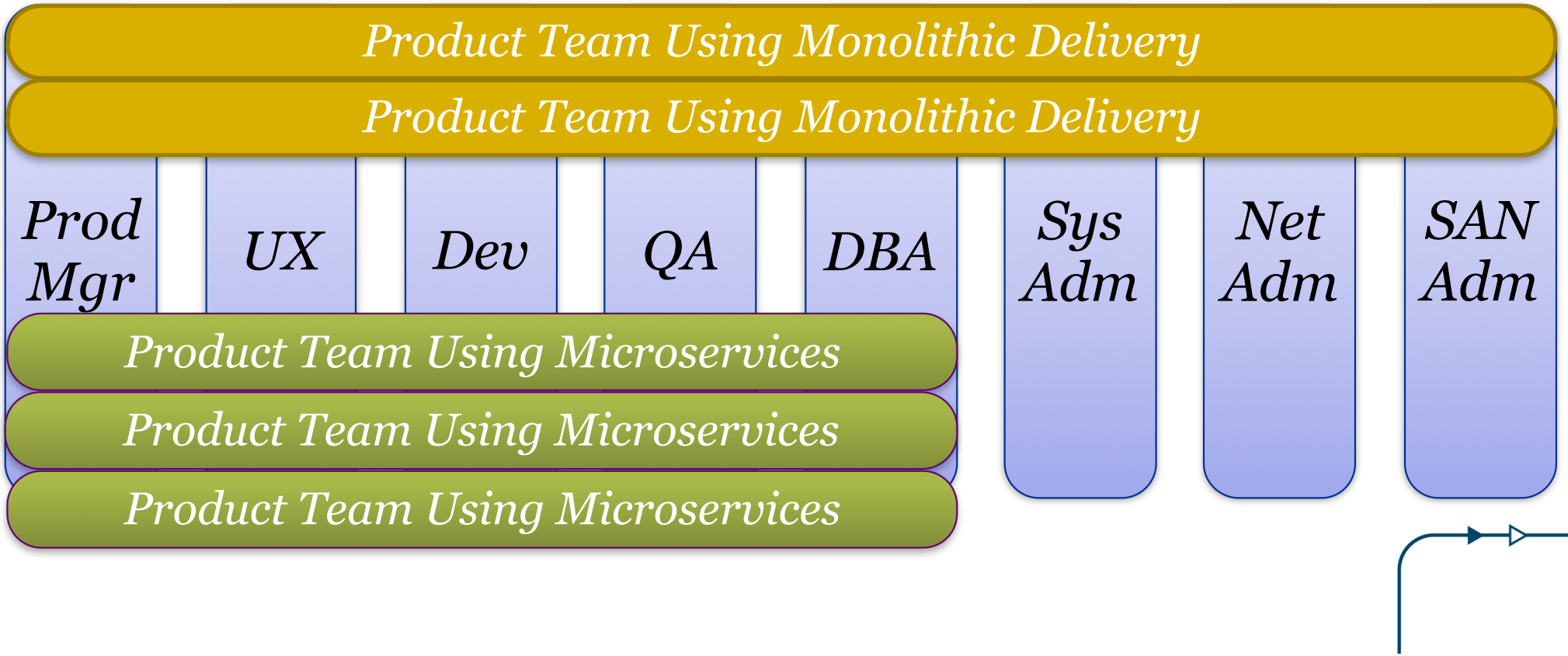
Breaking Down the SILOs



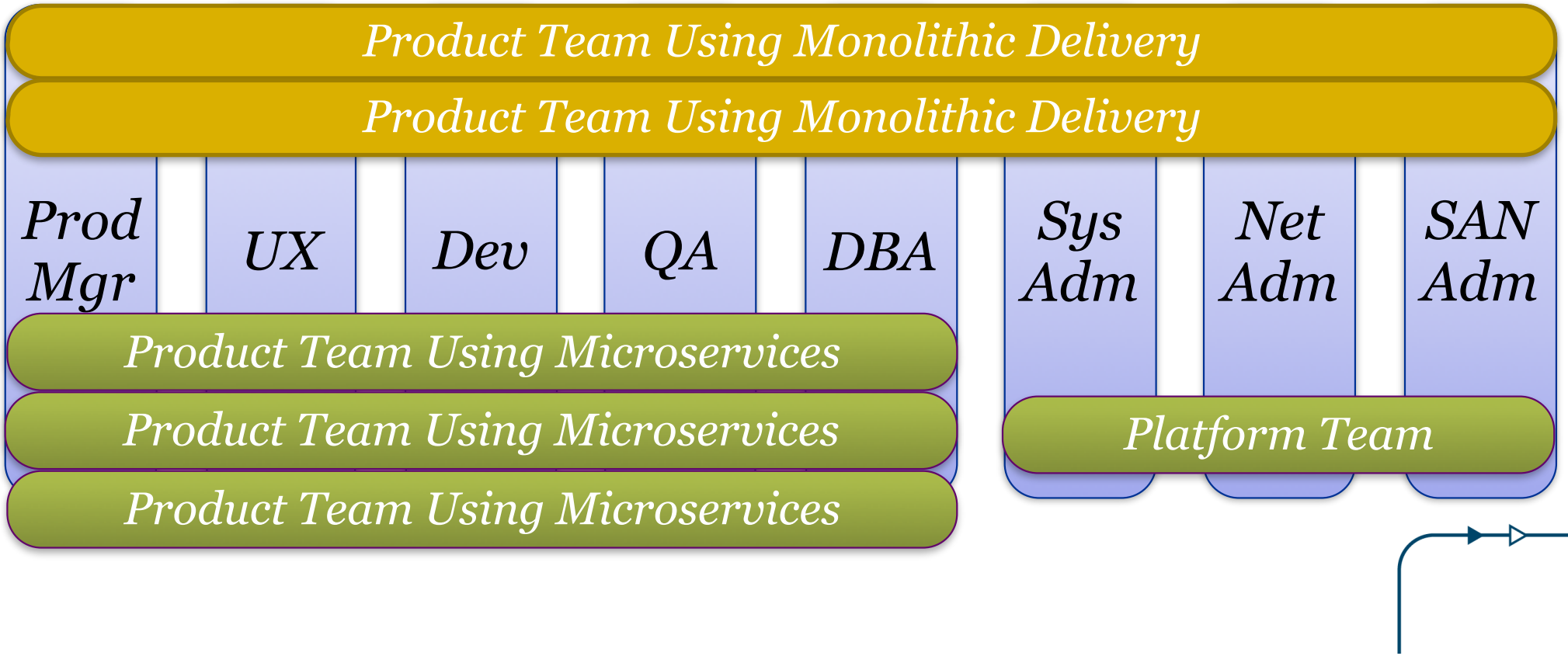
Breaking Down the SILOs



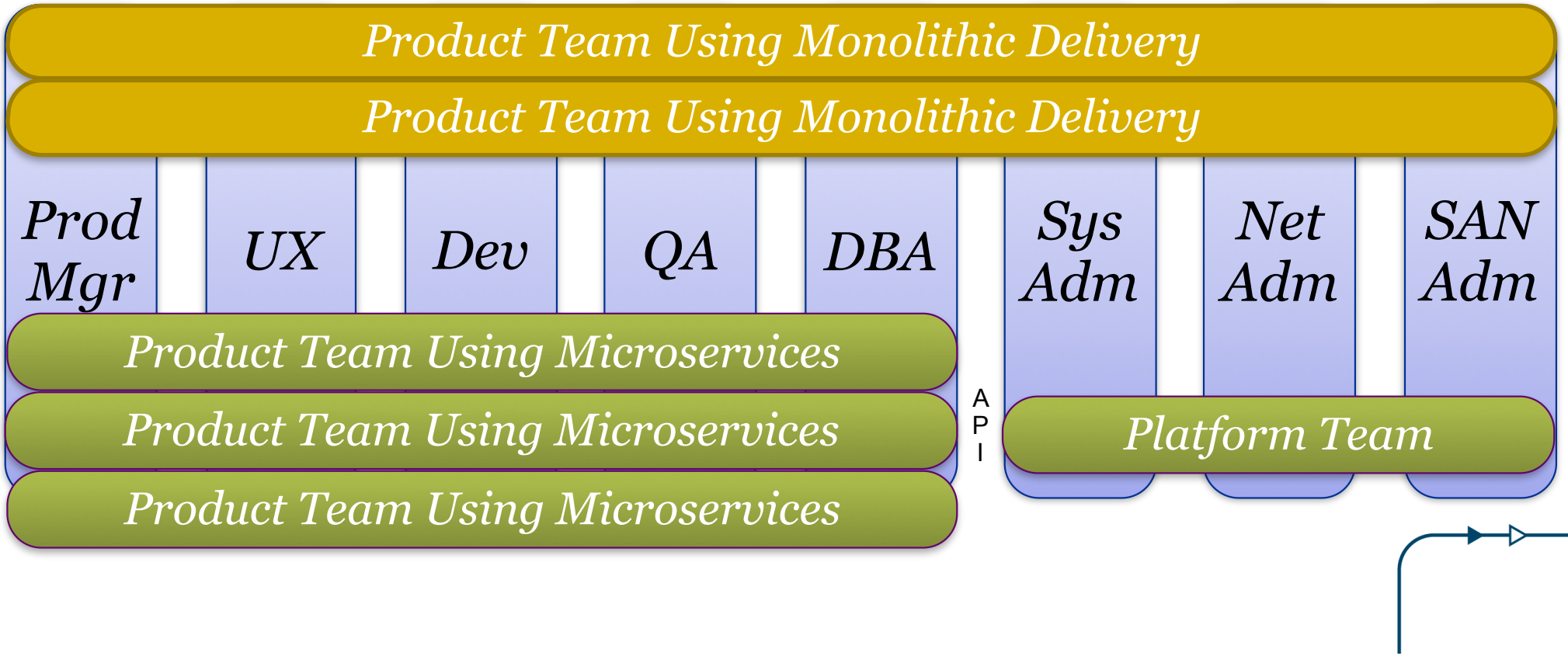
Breaking Down the SILOs



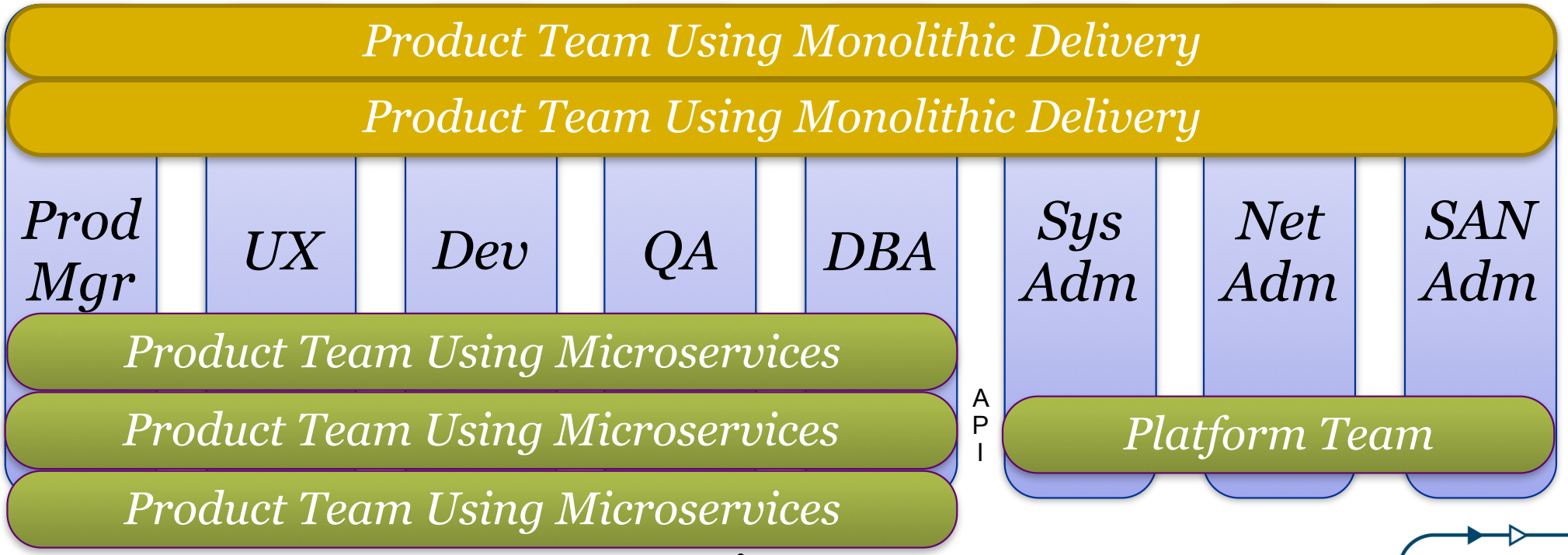
Breaking Down the SILOs



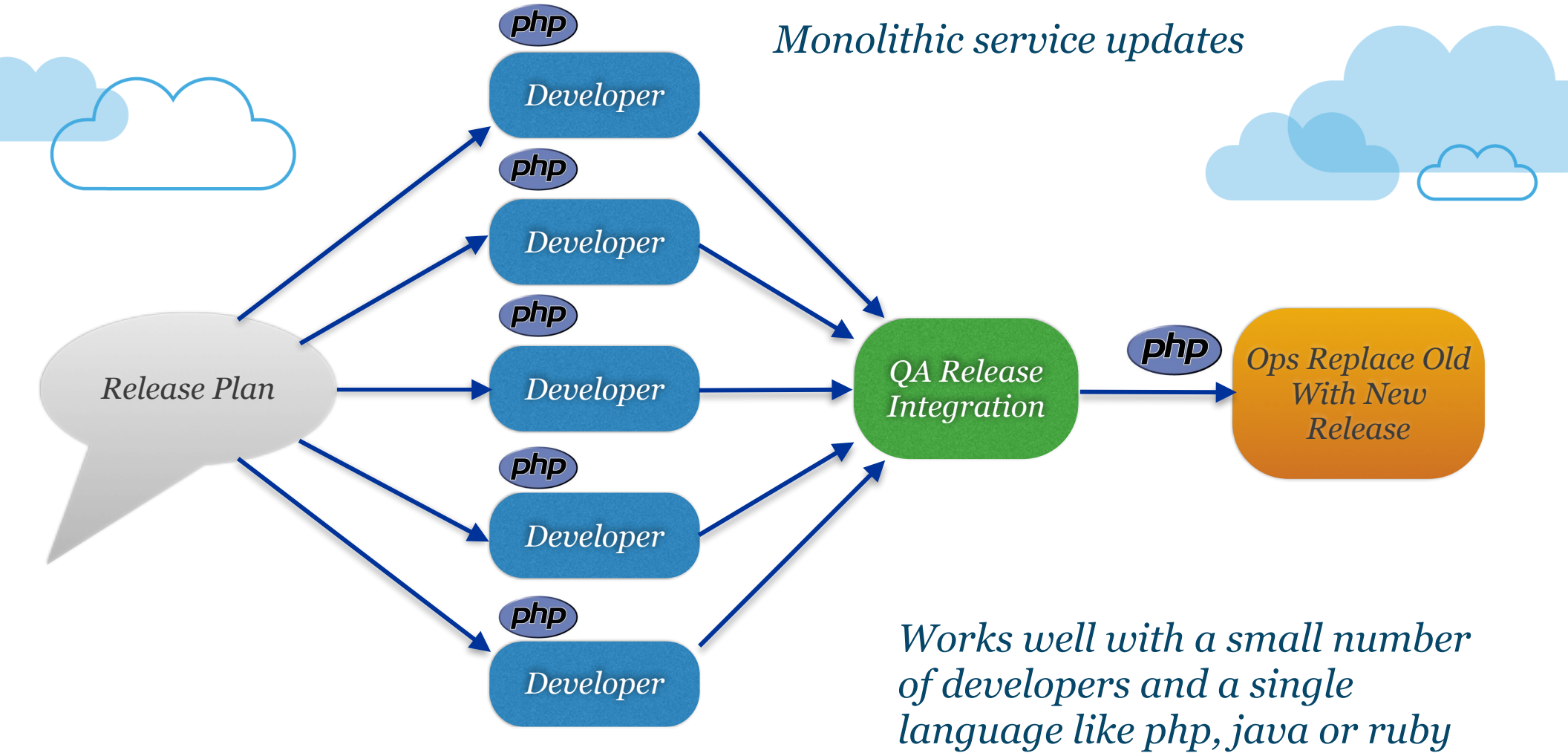
Breaking Down the SILOs

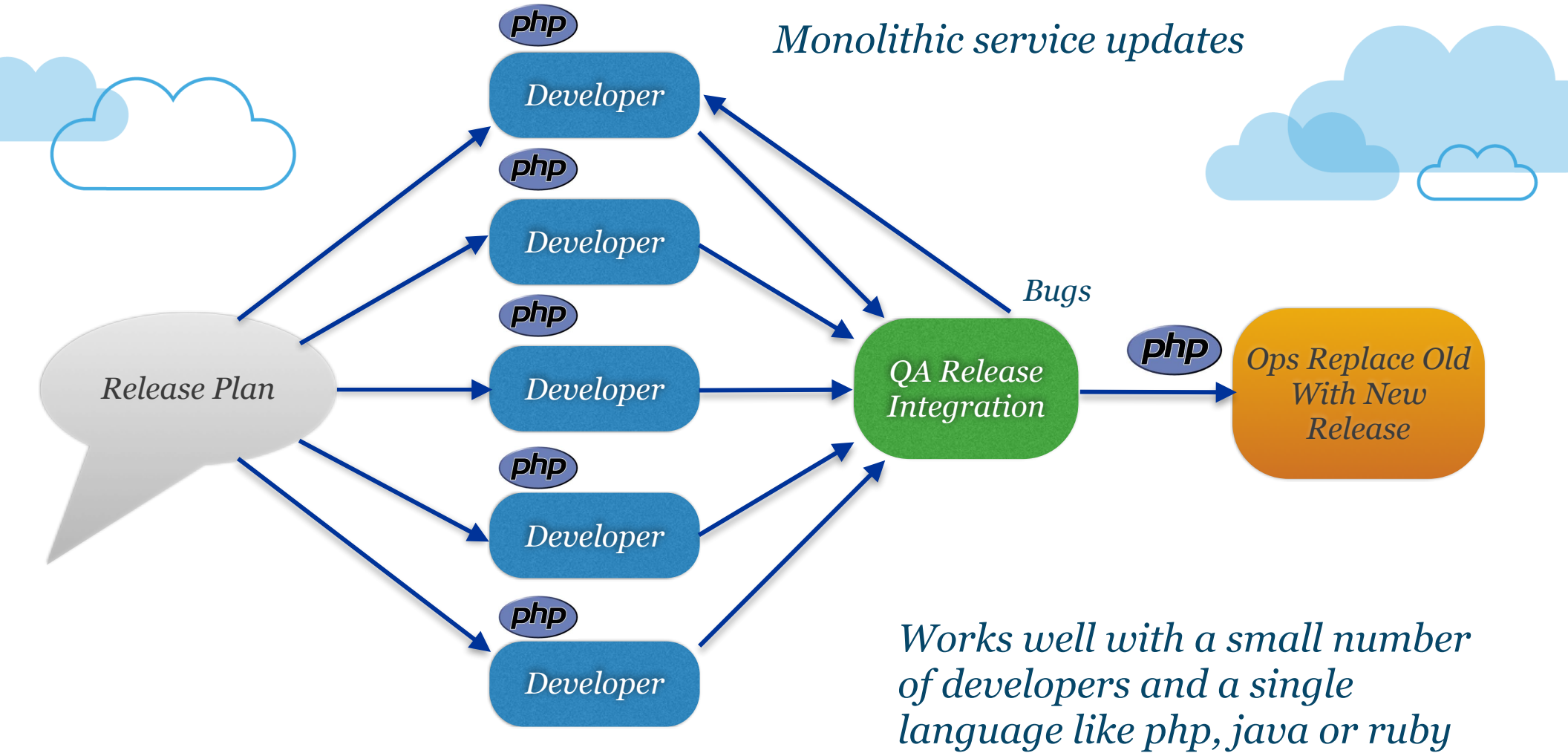


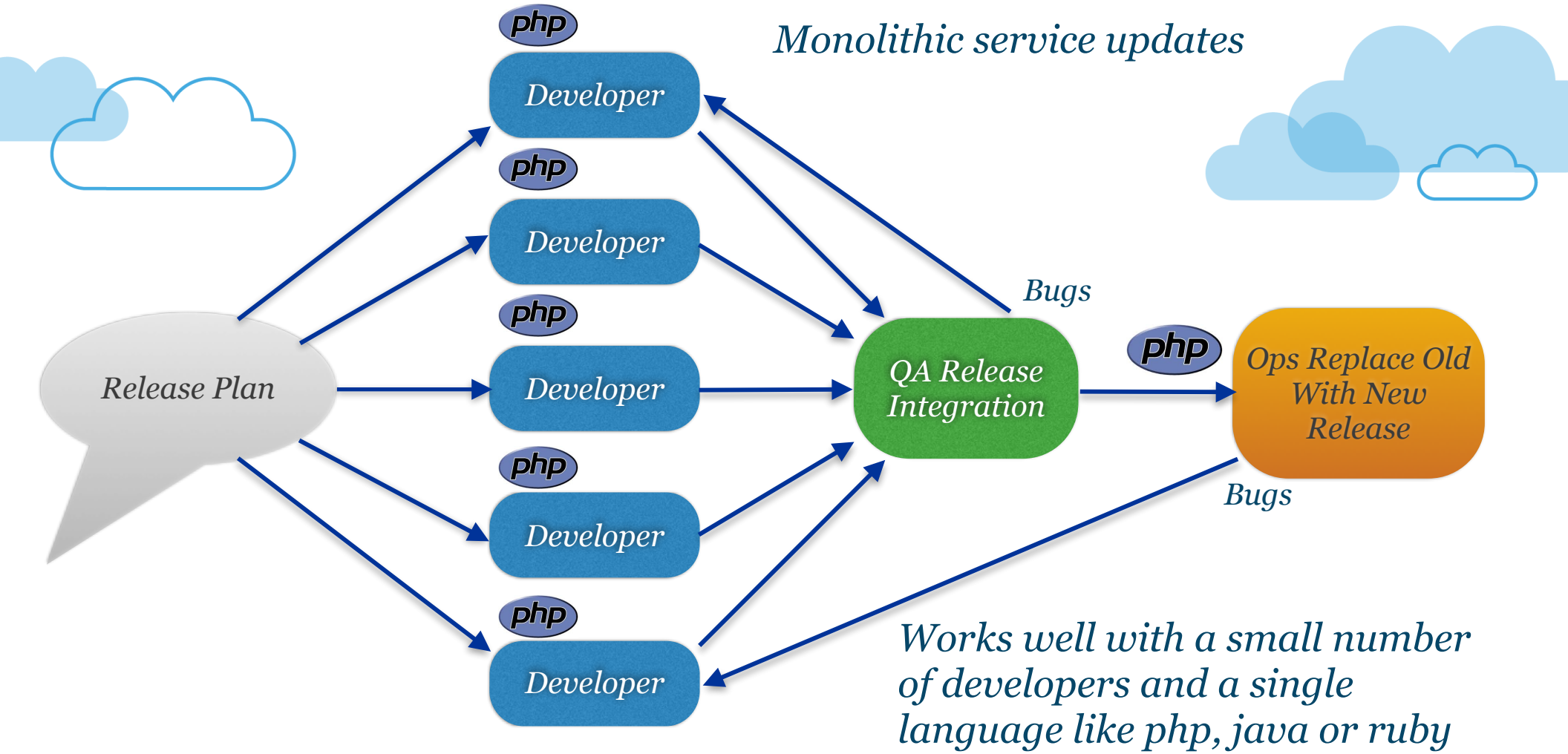
Breaking Down the SILOs

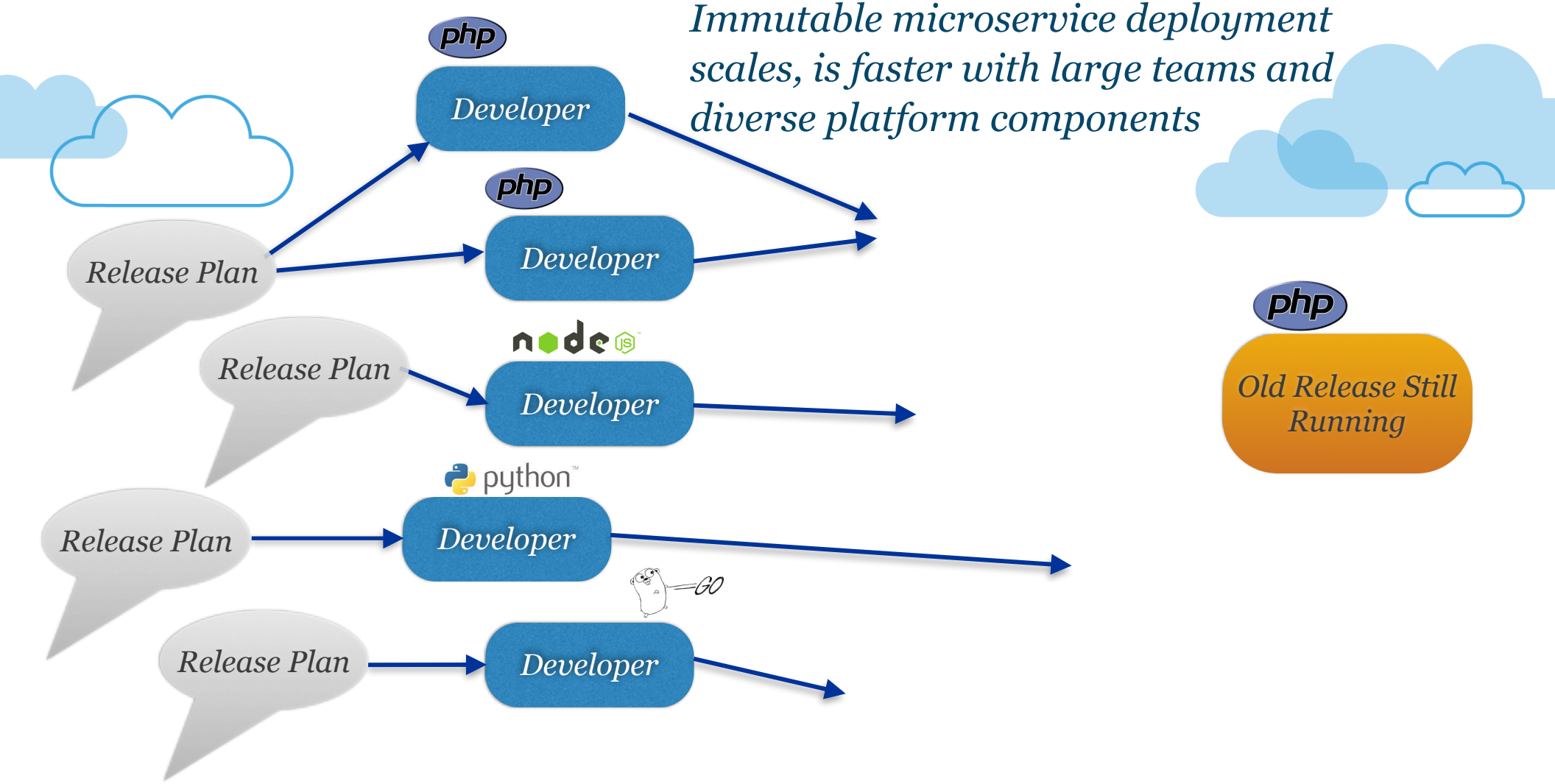


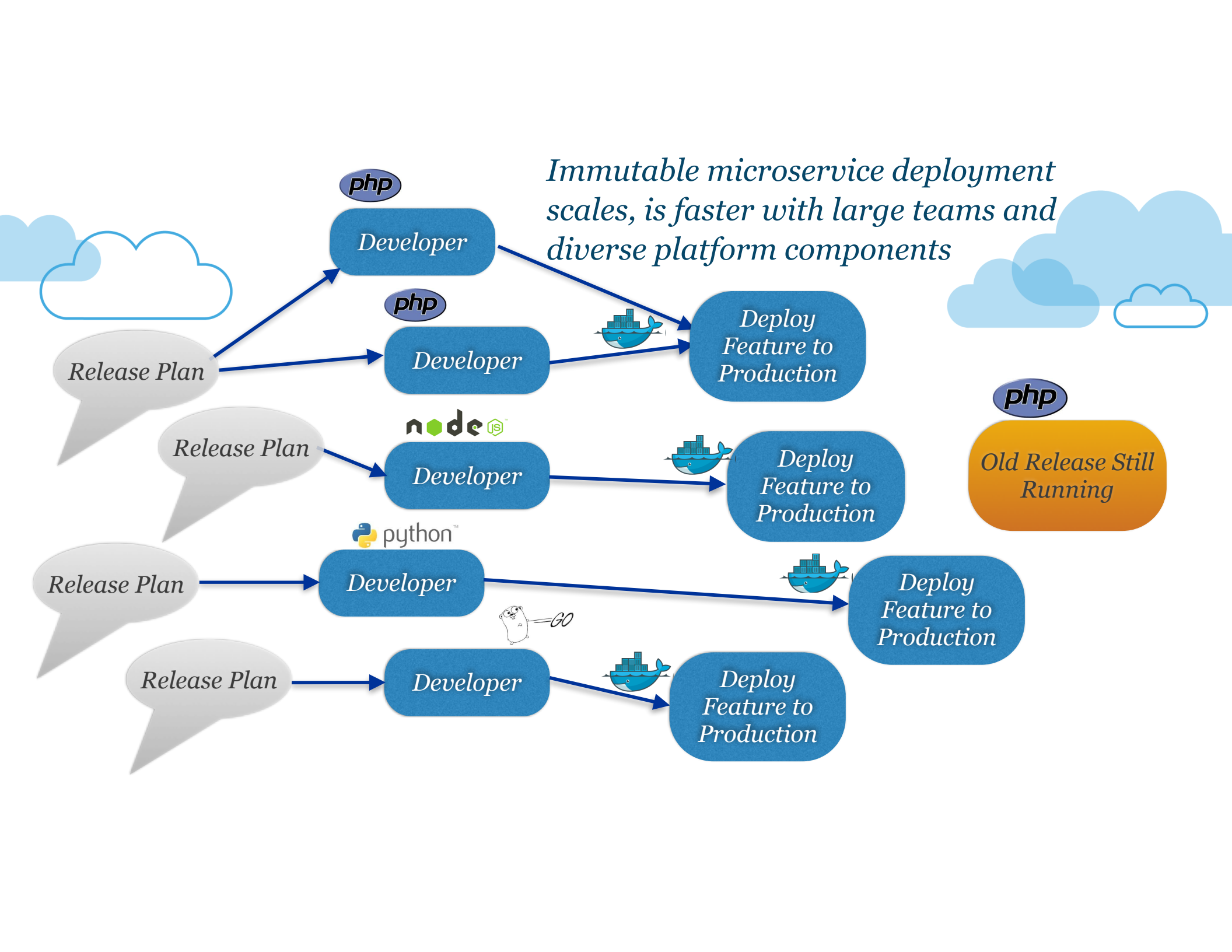
DevOps is a Re-Org!

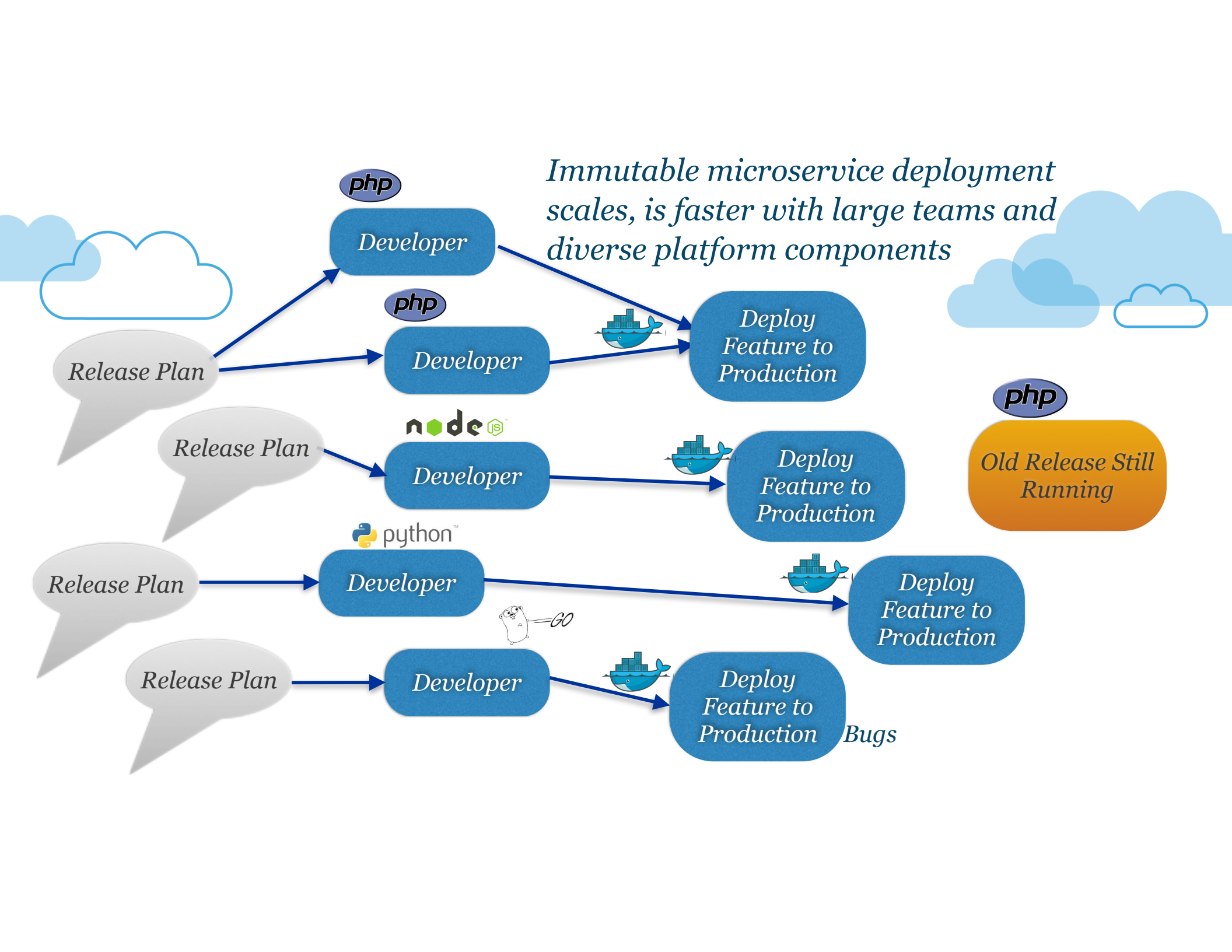

Monolithic service updates
php
Developer
php
Developer
php
Developer
php
Developer
php
Developer
Release Plan
QA Release Integration
php
Ops Replace Old With New Release
Works well with a small number of developers and a single language like php, java or ruby

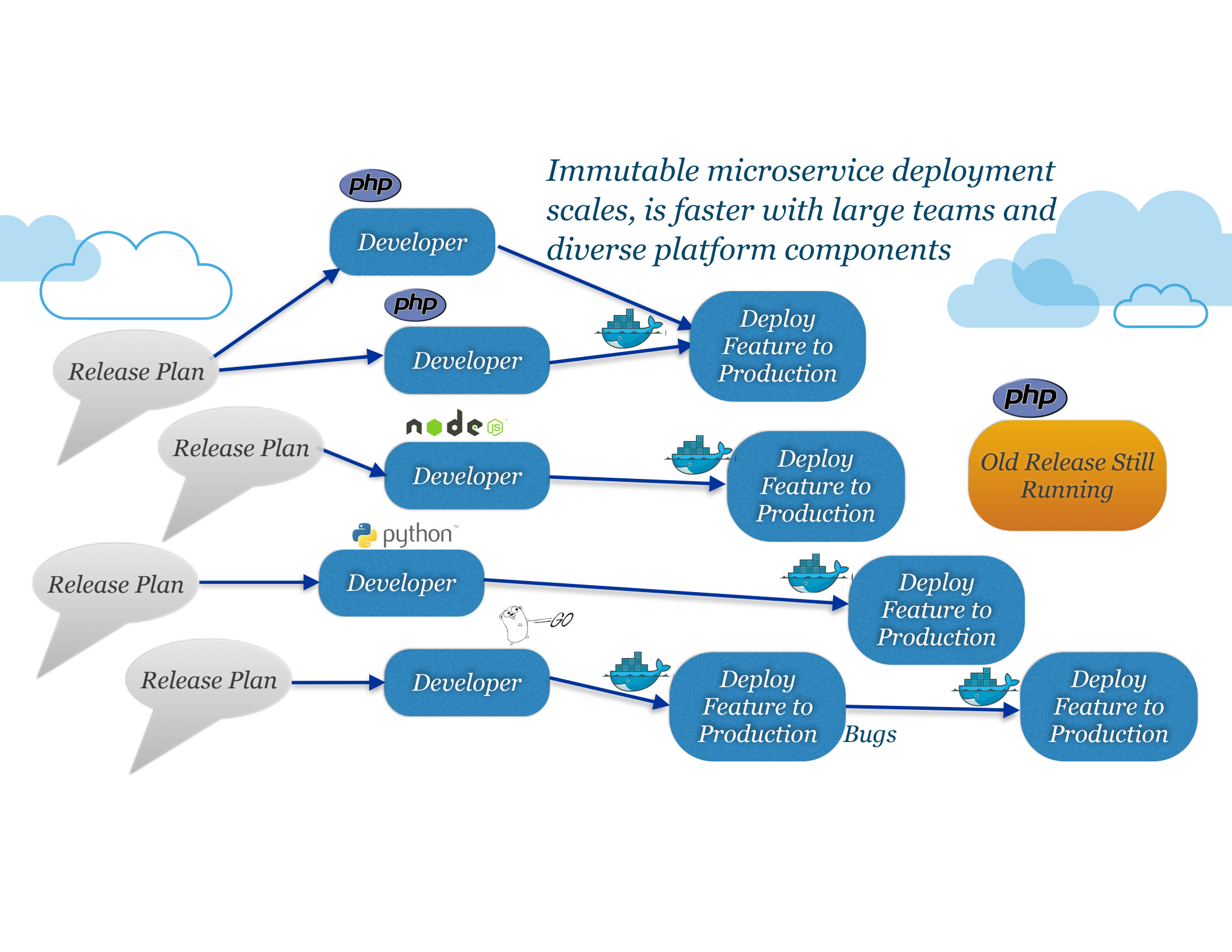

Monolithic service updates
Release Plan
Developer
Developer
Developer
Developer
Developer
Bugs
QA Release Integration
Ops Replace Old With New Release
Works well with a small number of developers and a single language like php, java or ruby


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

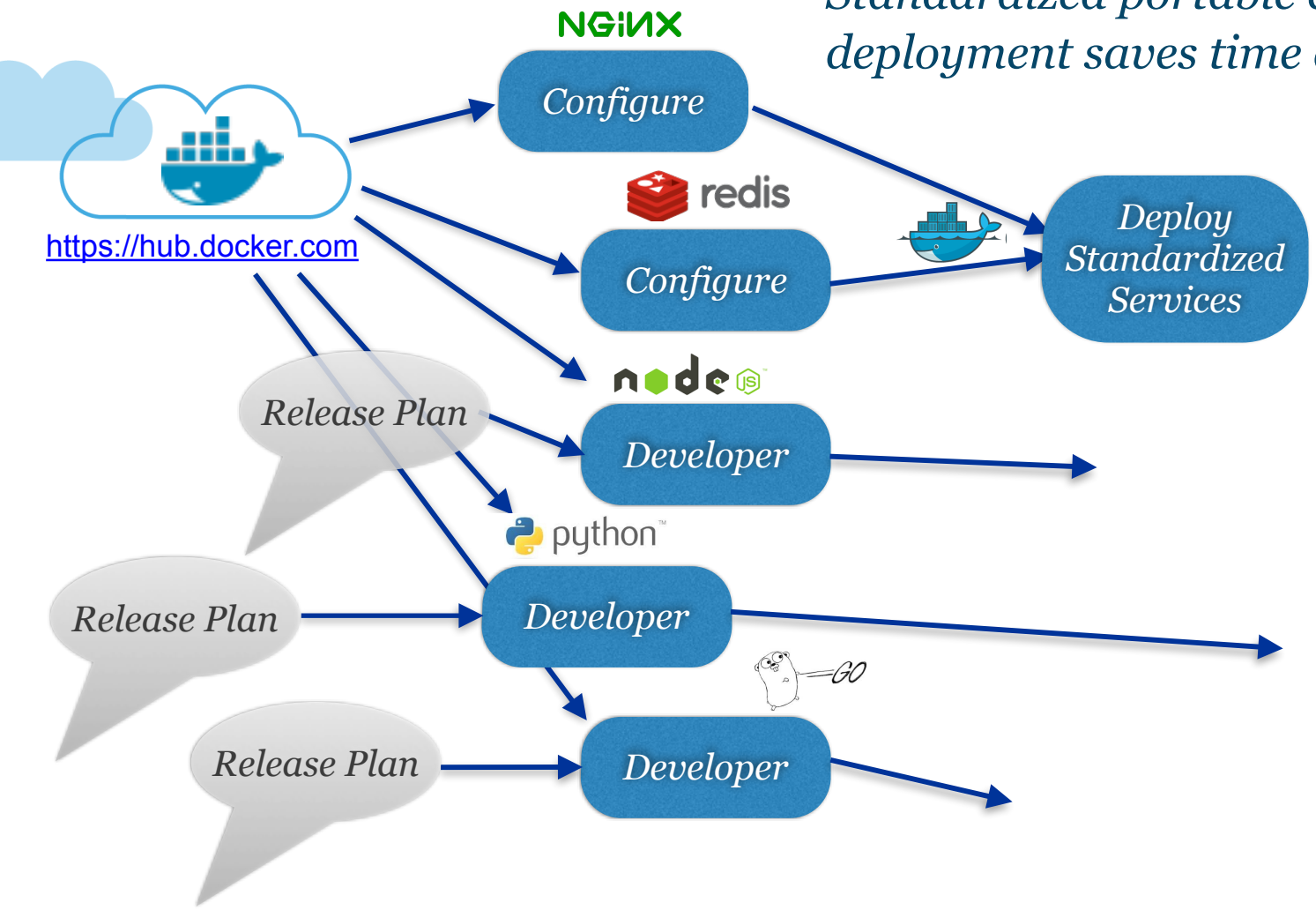

Immutable microservice deployment scales, is faster with large teams and diverse platform components
php
Developer
Release Plan
php
Developer
Release Plan
node
Developer
Release Plan
python
Developer
GO
Release Plan
Developer
php
Old Release Still Running

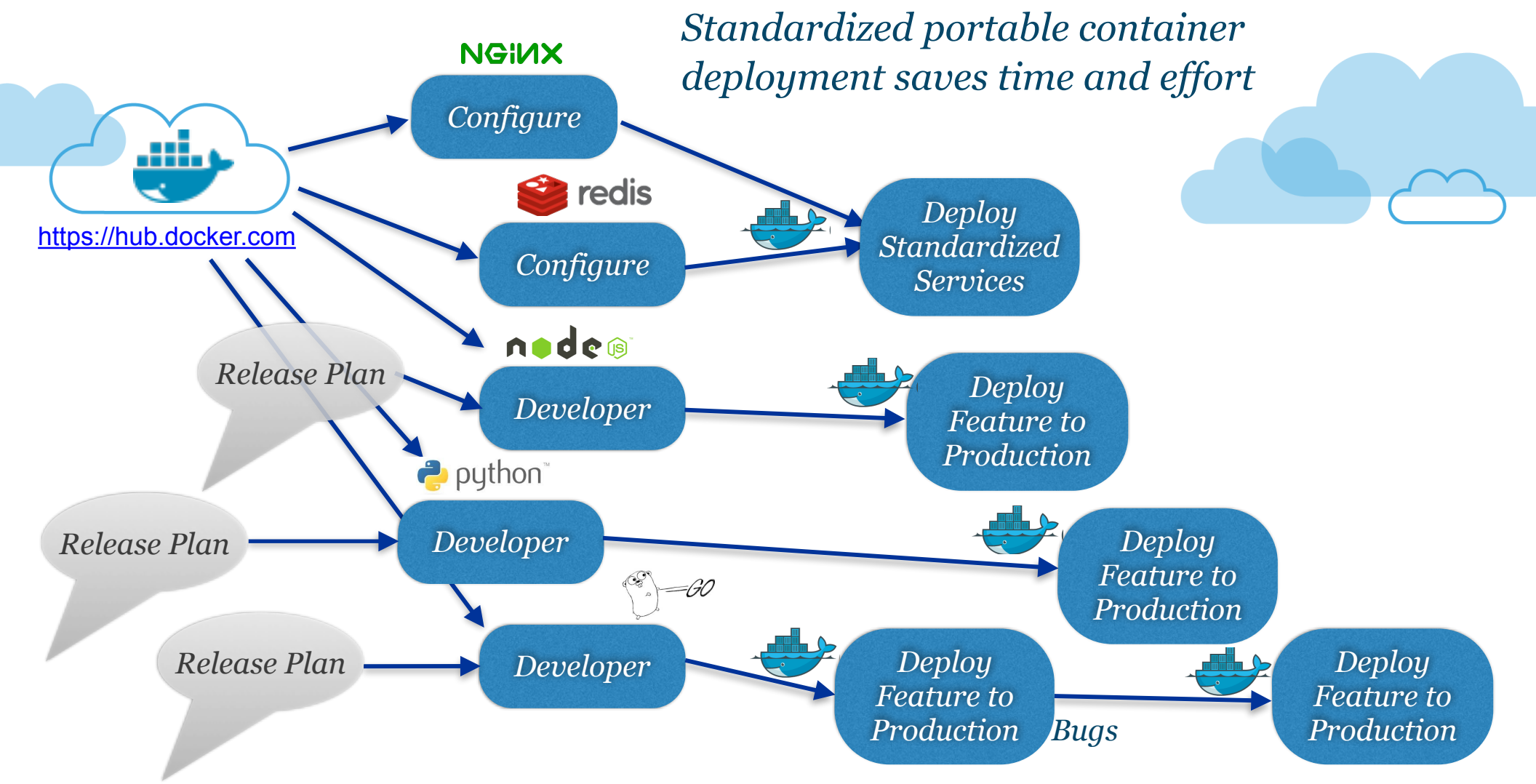

Immutable microservice deployment scales, is faster with large teams and diverse platform components
php
Developer
Release Plan
php
Developer
Deploy Feature to Production
node
Release Plan
Developer
Deploy Feature to Production
php
Old Release Still Running
python
Release Plan
Developer
Deploy Feature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


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

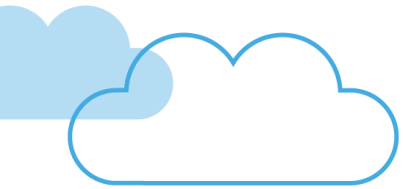

Immutable microservice deployment scales, is faster with large teams and diverse platform components
php
Developer
Release Plan
php
Developer
Deploy Feature to Production
Release Plan
node
Developer
Deploy Feature to Production
php
Old Release Still Running
Release Plan
python
Developer
Deploy Feature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

Standardized portable container deployment saves time and effort



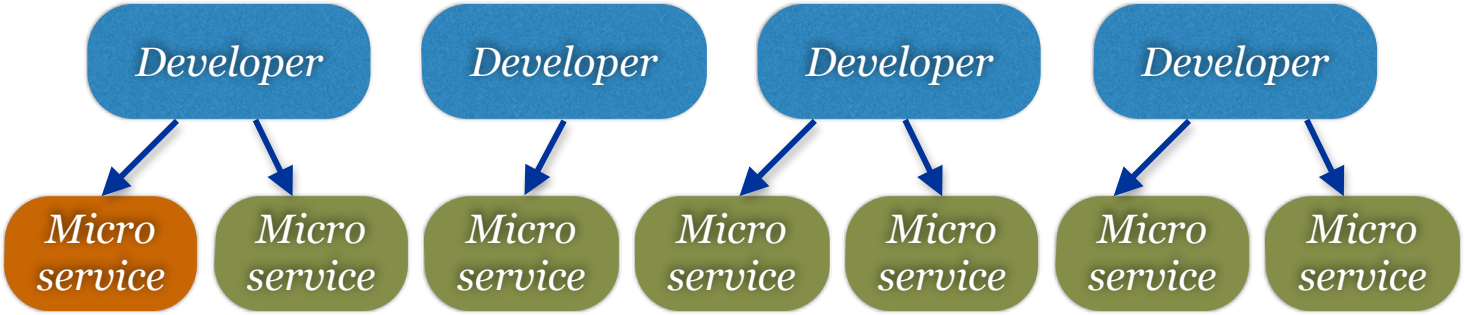

Standardized portable container deployment saves time and effort
NGINX
Configure
https://hub.docker.com
redis
Configure
Deploy Standardized Services
node
Release Plan
Developer
Deploy Feature to Production
python
Release Plan
Developer
Deploy Feature to Production
GO
Release Plan
Developer
Deploy Feature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

Run What You Wrote

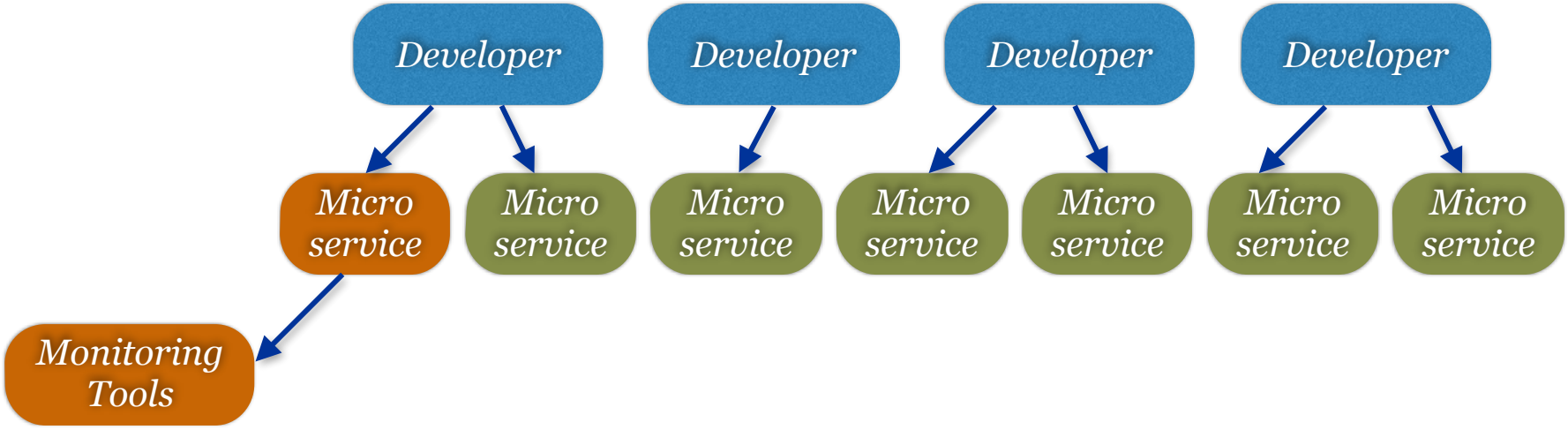


Developer *Developer* *Developer* *Developer*

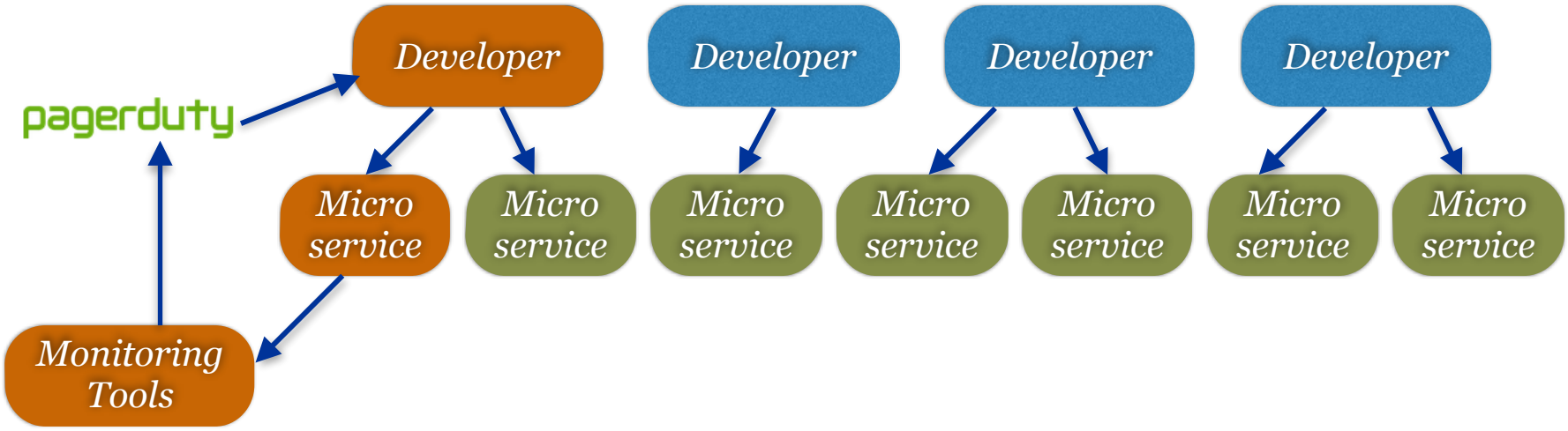
Run What You Wrote



Run What You Wrote

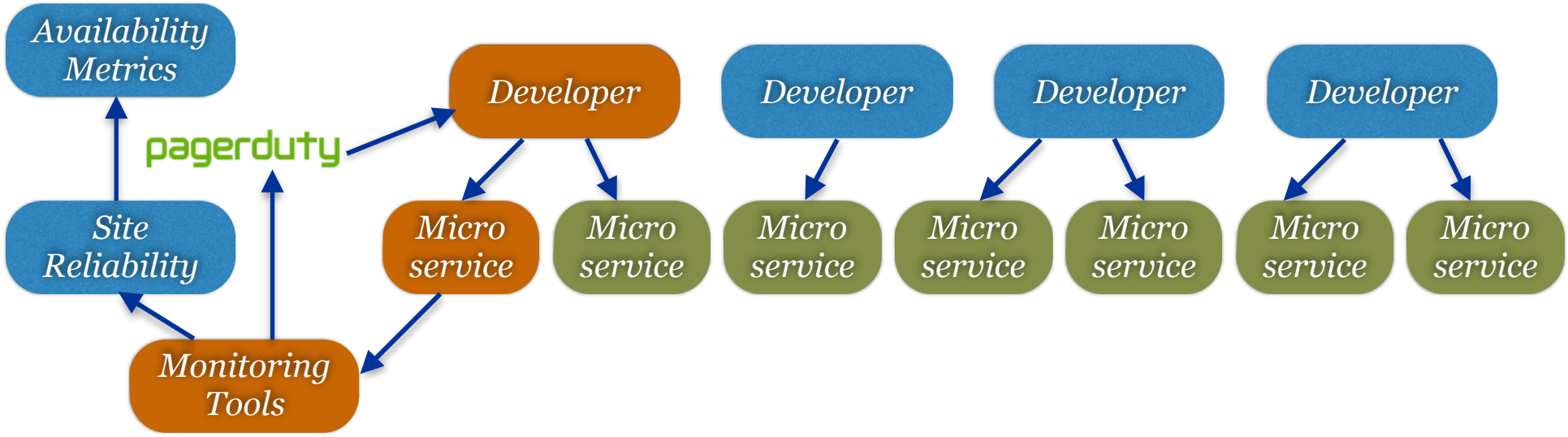


Run What You Wrote



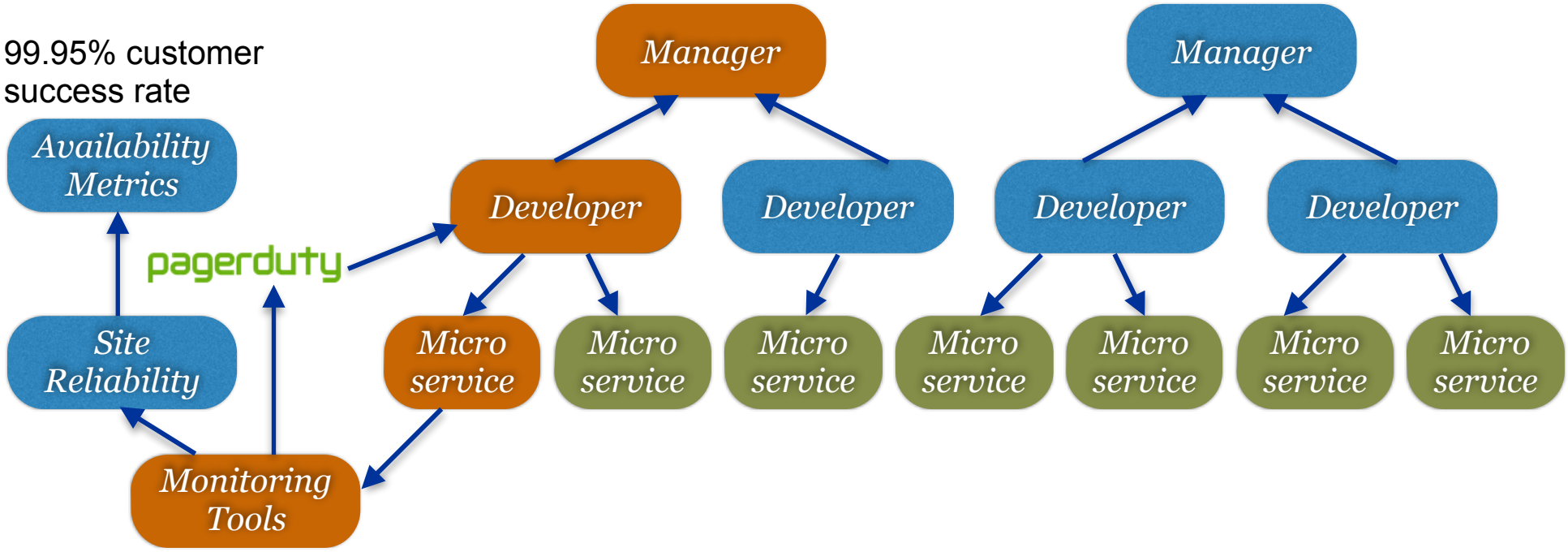
Run What You Wrote

99.95% customer success rate



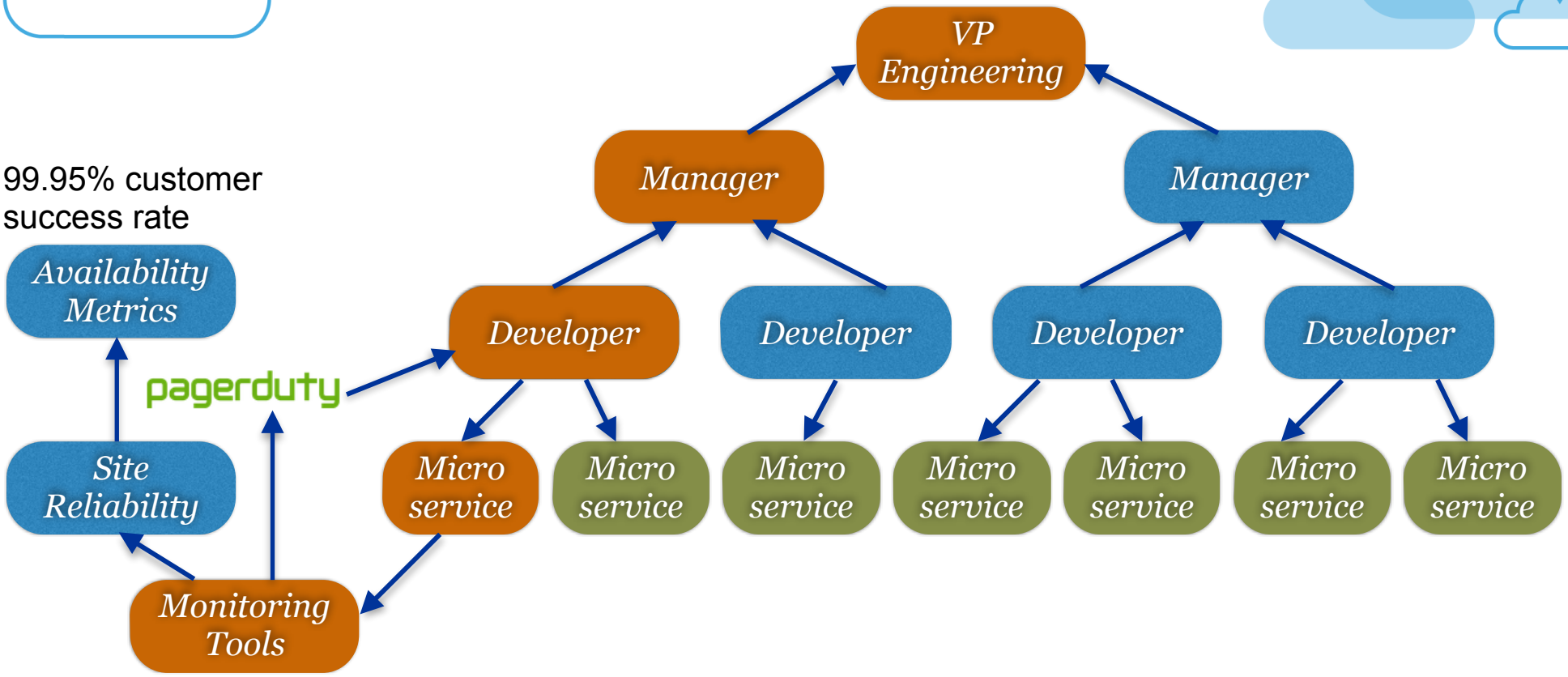
Run What You Wrote

99.95% customer success rate

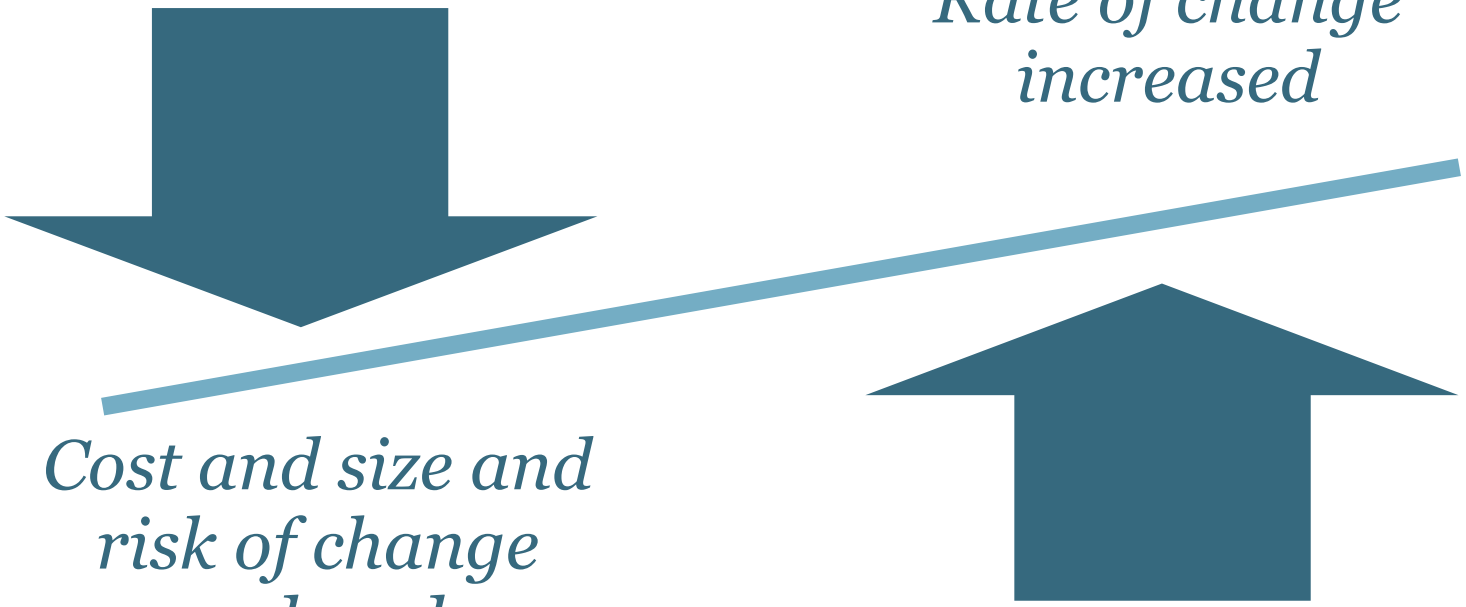


Run What You Wrote

99.95% customer success rate



What Happened?

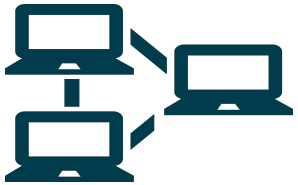


Developing at the Speed of Docker

Developers
- Compile/Build
- Seconds

docker

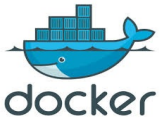
Extend container
- Package dependencies
- Seconds



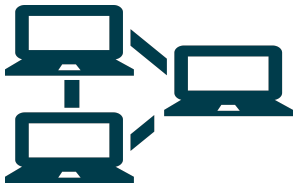
PaaS deploy Container
- Docker startup
- Seconds

Developing at the Speed of Docker

Developers
- Compile/Build
- Seconds

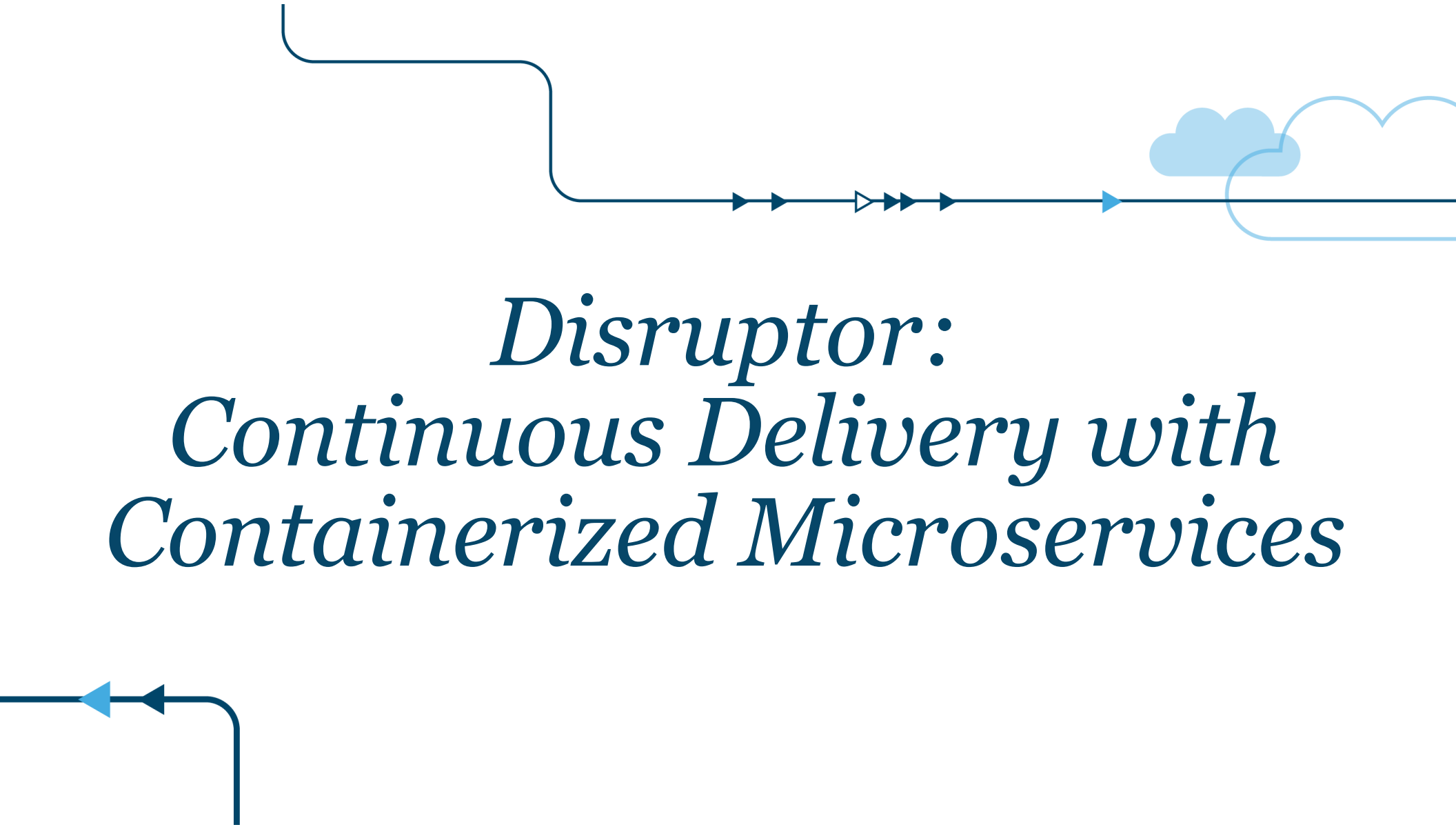


Extend container
- Package dependencies
- Seconds



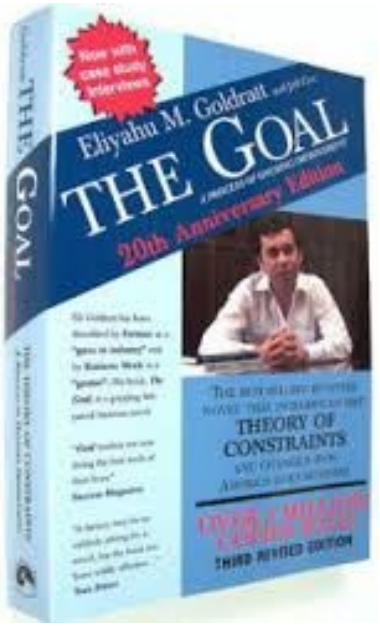
PaaS deploy Container
- Docker startup
- Seconds

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



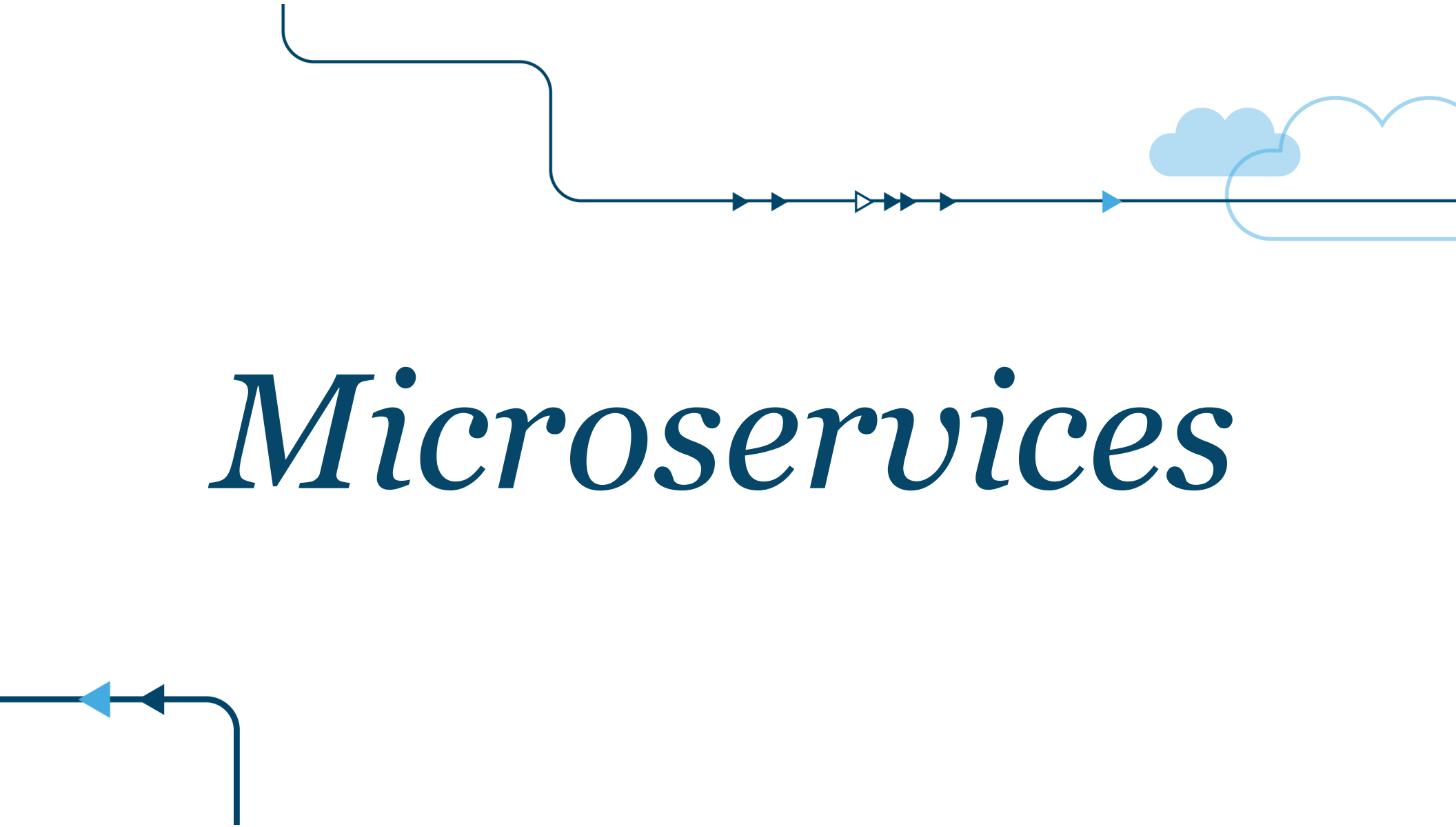
Disruptor: Continuous Delivery with Containerized Microservices

"This is the IT swamp draining manual for anyone who is neck deep in alligators."

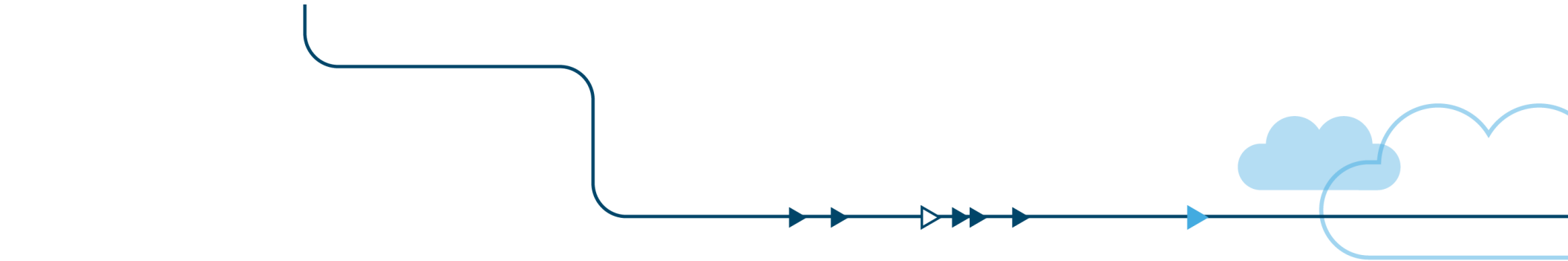


1984

2014

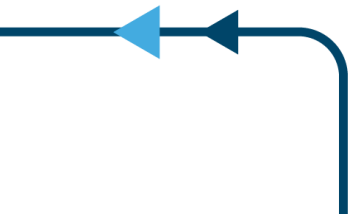


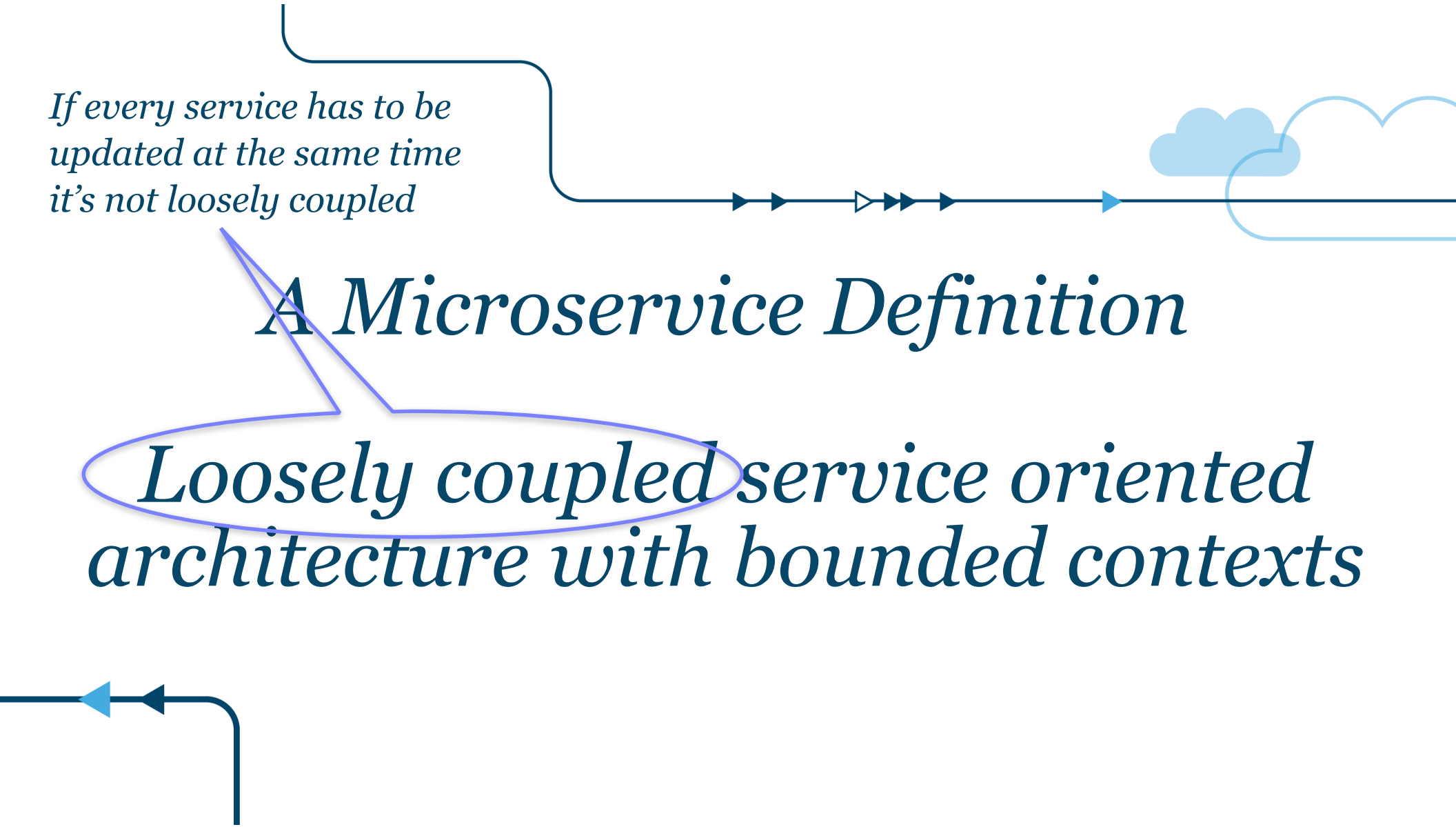
Microservices

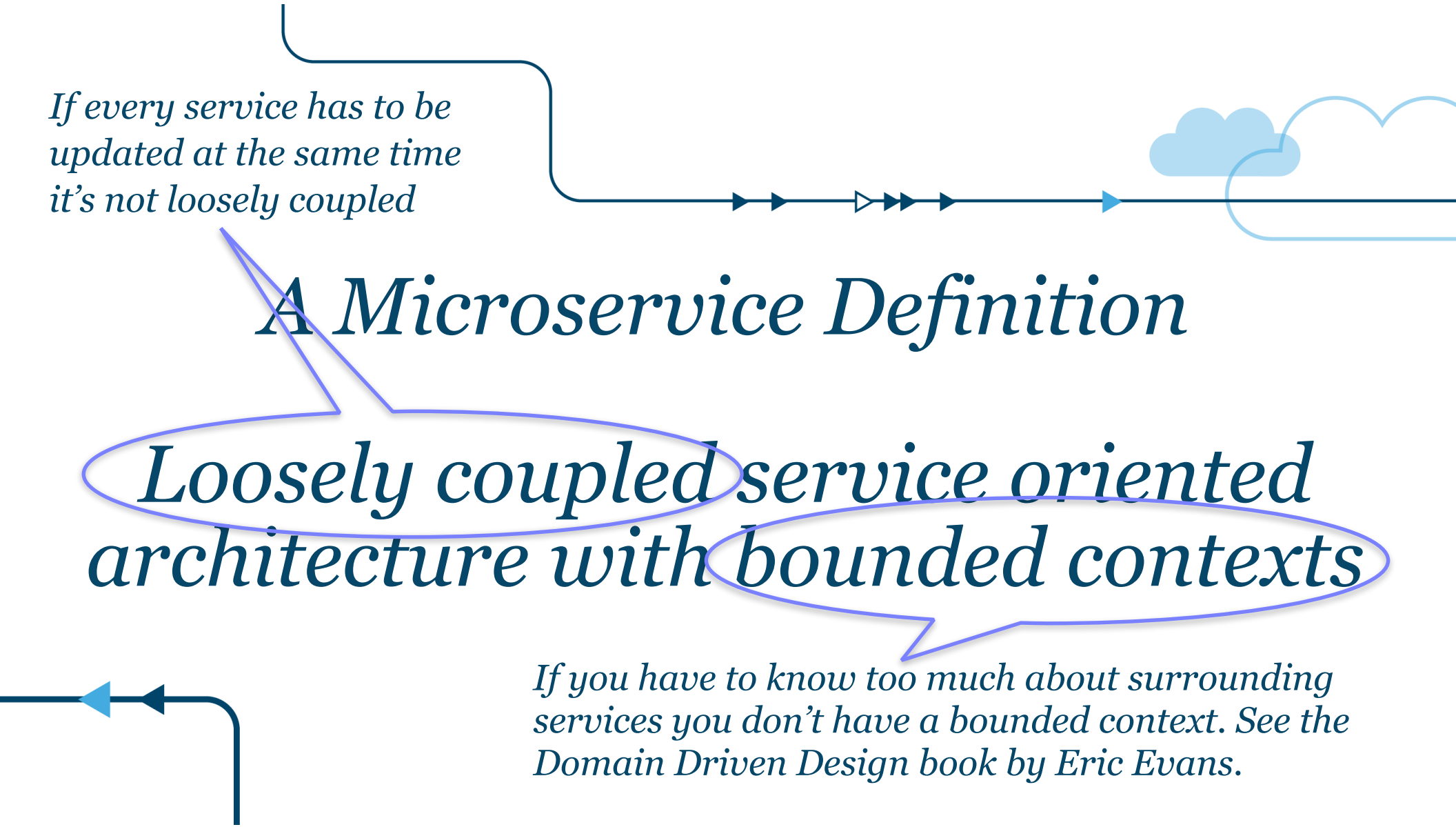


A Microservice Definition

Loosely coupled service oriented architecture with bounded contexts




If every service has to be updated at the same time it's not loosely coupled
A Microservice Definition
Loosely coupled service oriented architecture with bounded contexts


If every service has to be updated at the same time it's not loosely coupled
A Microservice Definition
Loosely coupled service oriented architecture with bounded contexts
If you have to know too much about surrounding services you don't have a bounded context. See the Domain Driven Design book by Eric Evans.

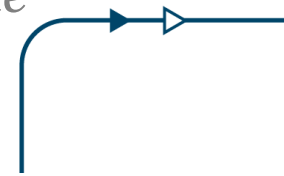
Coupling Concerns

- *Conway's Law - organizational coupling*
- *Centralized Database Schemas*
- *Enterprise Service Bus - centralized message queues*
- *Inflexible Protocol Versioning*

http://en.wikipedia.org/wiki/Conway's_law

Non-Destructive Production Updates

- *"Immutable Code" Service Pattern*
  - *Existing services are unchanged, old code remains in service*
  - *New code deploys as a new service group*
  - *No impact to production until traffic routing changes*
- *A|B Tests, Feature Flags and Version Routing control traffic*
  - *First users in the test cell are the developer and test engineers*
  - *A cohort of users is added looking for measurable improvement*
  - *Finally make default for everyone, keeping old code for a while*



Speeding Up The Platform

Datacenter Snowflakes
- Deploy in months
- Live for years

Speeding Up The Platform

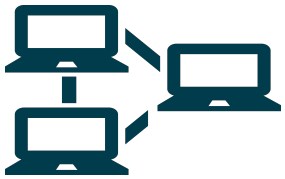
Datacenter Snowflakes
- Deploy in months
- Live for years

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

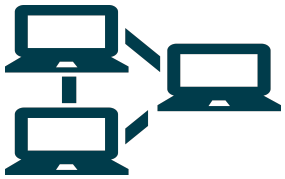


Docker Containers
- Deploy in seconds
- Live for minutes/hours

Speeding Up The Platform

Datacenter Snowflakes
- Deploy in months
- Live for years

Virtualized and Cloud
- Deploy in minutes
- Live for weeks



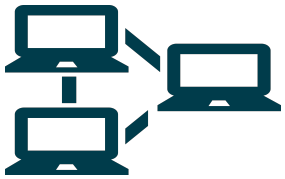
Docker Containers
- Deploy in seconds
- Live for minutes/hours

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

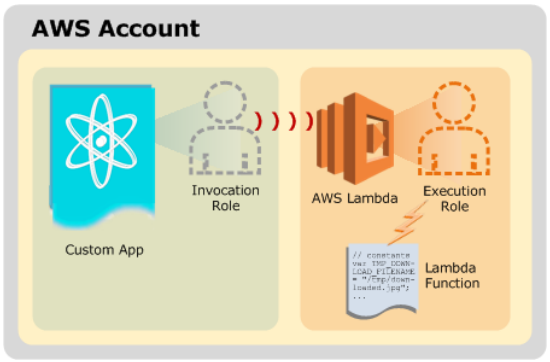


Docker Containers
- Deploy in seconds
- Live for minutes/hours

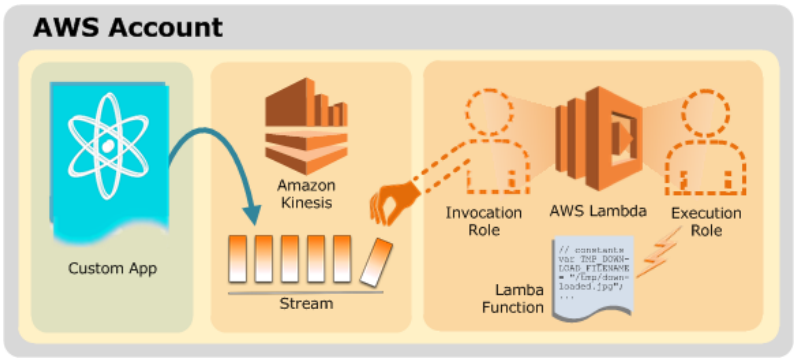
AWS Lambda
- Deploy in milliseconds
- Live for seconds

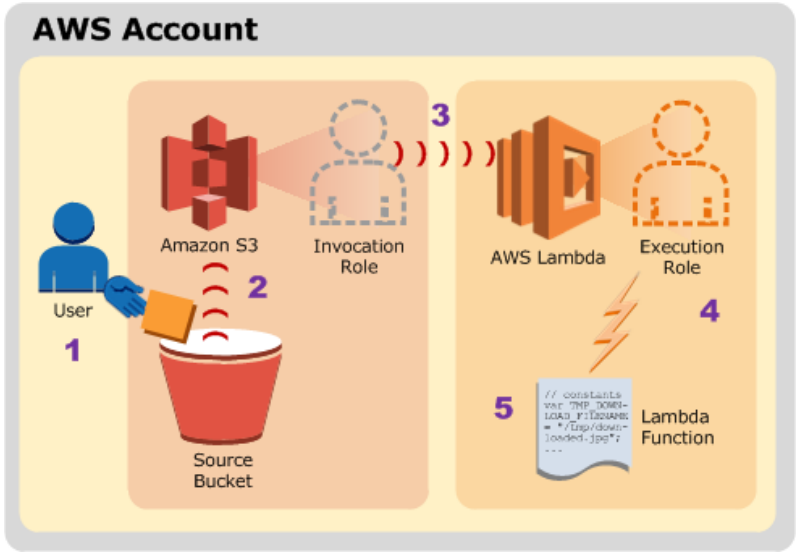
Speed enables and encourages new microservice architectures

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

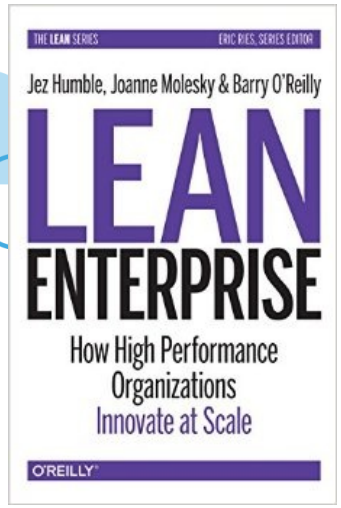


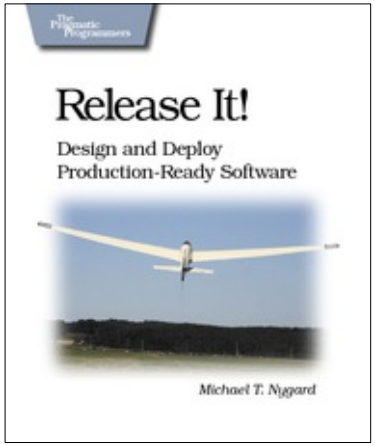
First 1M node.js executions/month are free

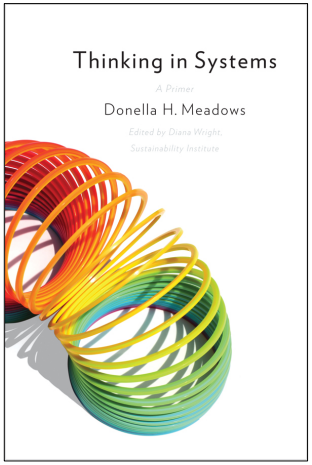


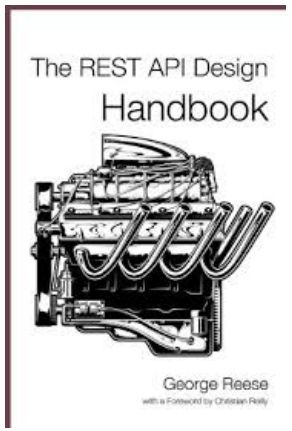


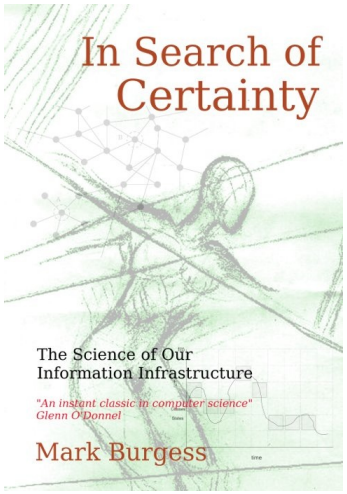
Inspiration

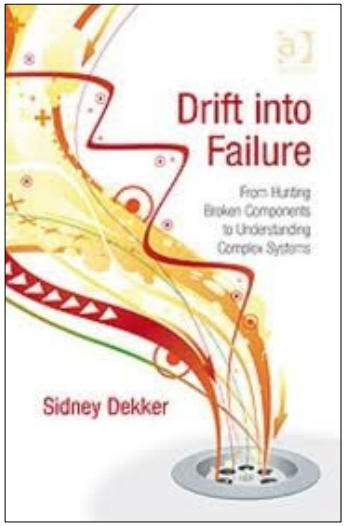


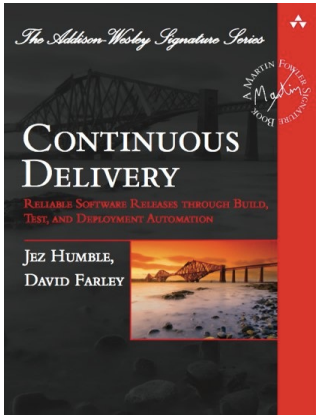


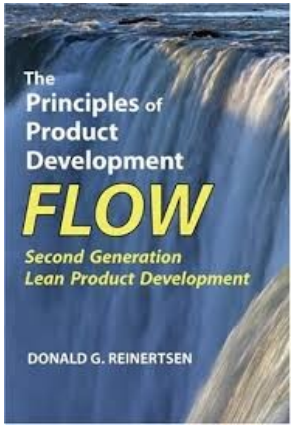


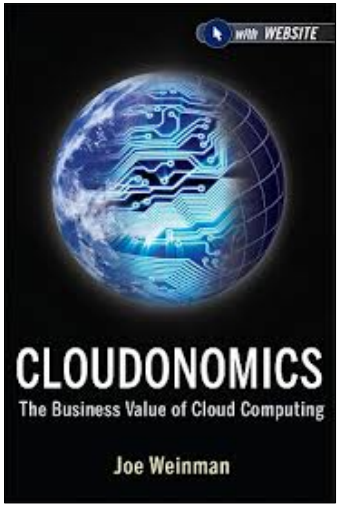




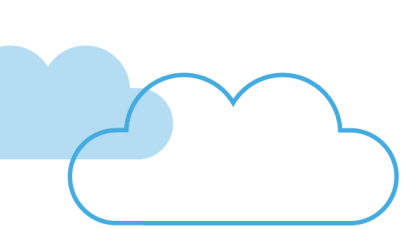








State of the Art in Web Scale Microservice Architectures



NETFLIX | OSS

GILT

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

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

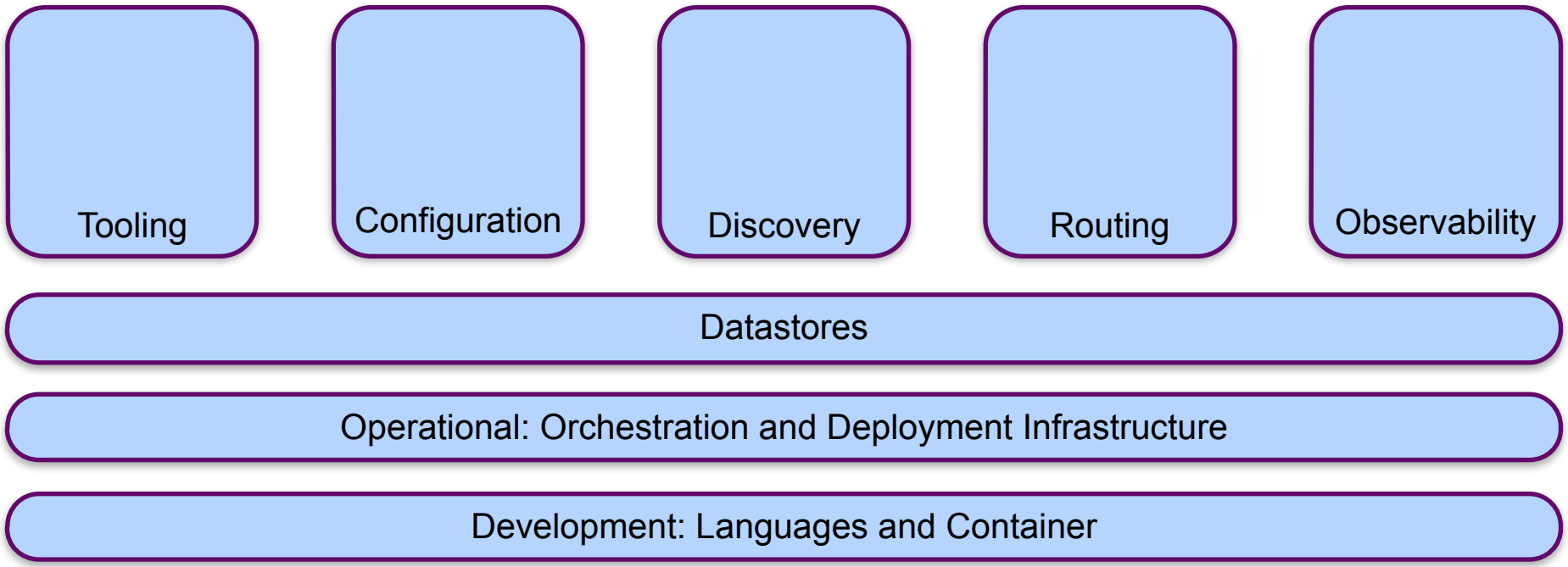
GROUPON

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

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

Microservice Concerns



NETFLIX | OSS Microservices

Asgard Aminator Tooling	Edda Archaius Configuration	Eureka Prana Discovery	Denominator Zuul, Netty Ribbon 2.0 Routing	Hystrix Pytheus SALP Observability

Ephemeral datastores using Dynomite, Memcached, Astyanax, Staash, Priam, Cassandra

Manual Orchestration with Asgard and deployment on AWS or Eucalyptus

Java, Groovy, Scala, Clojure, Python, Node.js with AMI and Docker Containers

NETFLIX | OSS Microservices

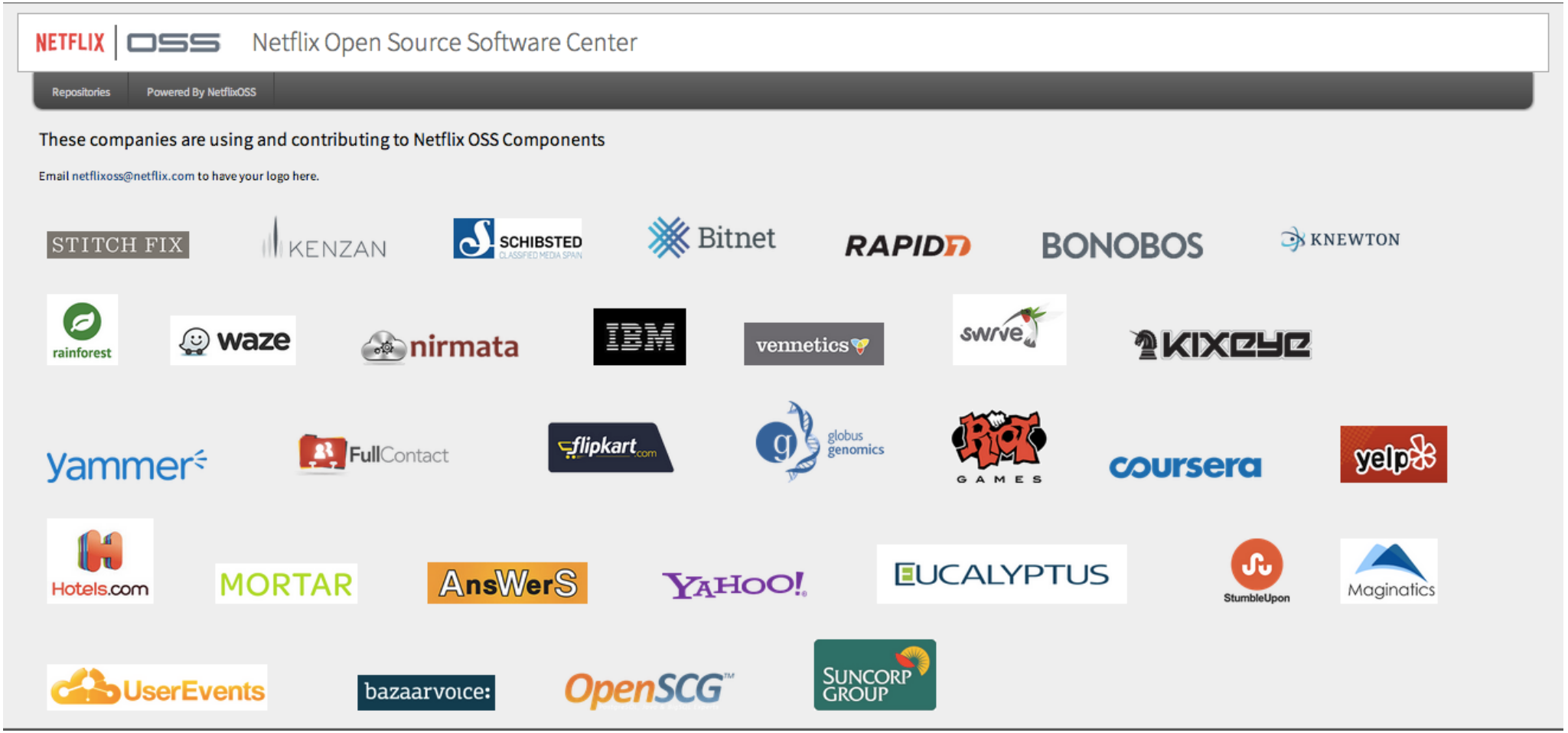
Asgard Aminator Tooling	Edda Archaius Configuration	Eureka Prana Discovery	Denominator Zuul, Netty Ribbon 2.0 Routing	Hystrix Pytheus SALP Observability

Ephemeral datastores using Dynomite, Memcached, Astyanax, Staash, Priam, Cassandra

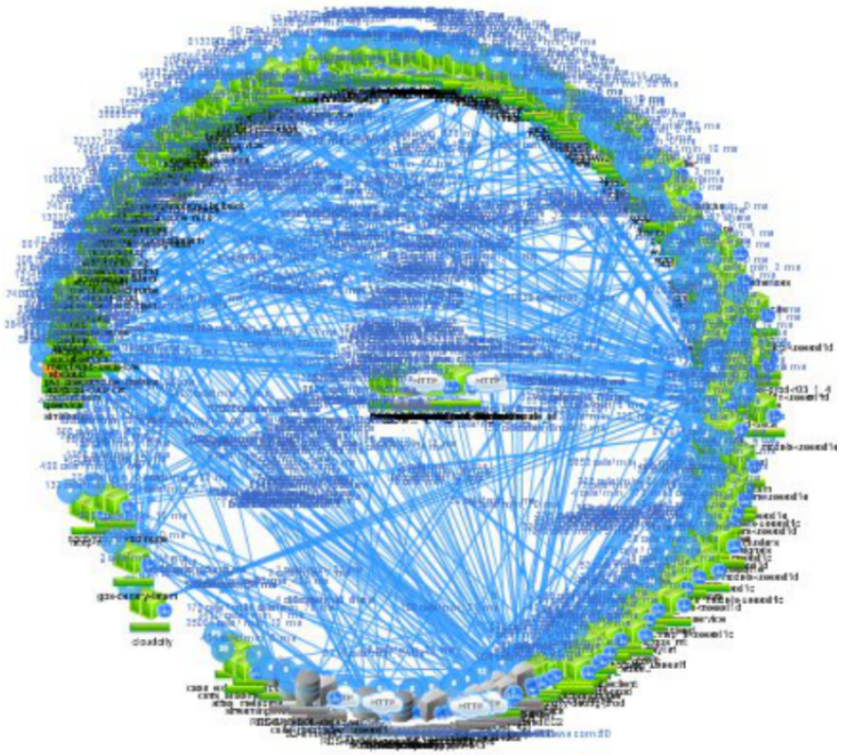
Manual Orchestration with Asgard and deployment on AWS or Eucalyptus

Java, Groovy, Scala, Clojure, Python, Node.js with AMI and Docker Containers

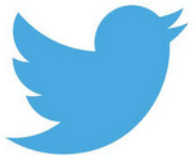
Focus on global distribution, high scale and availability

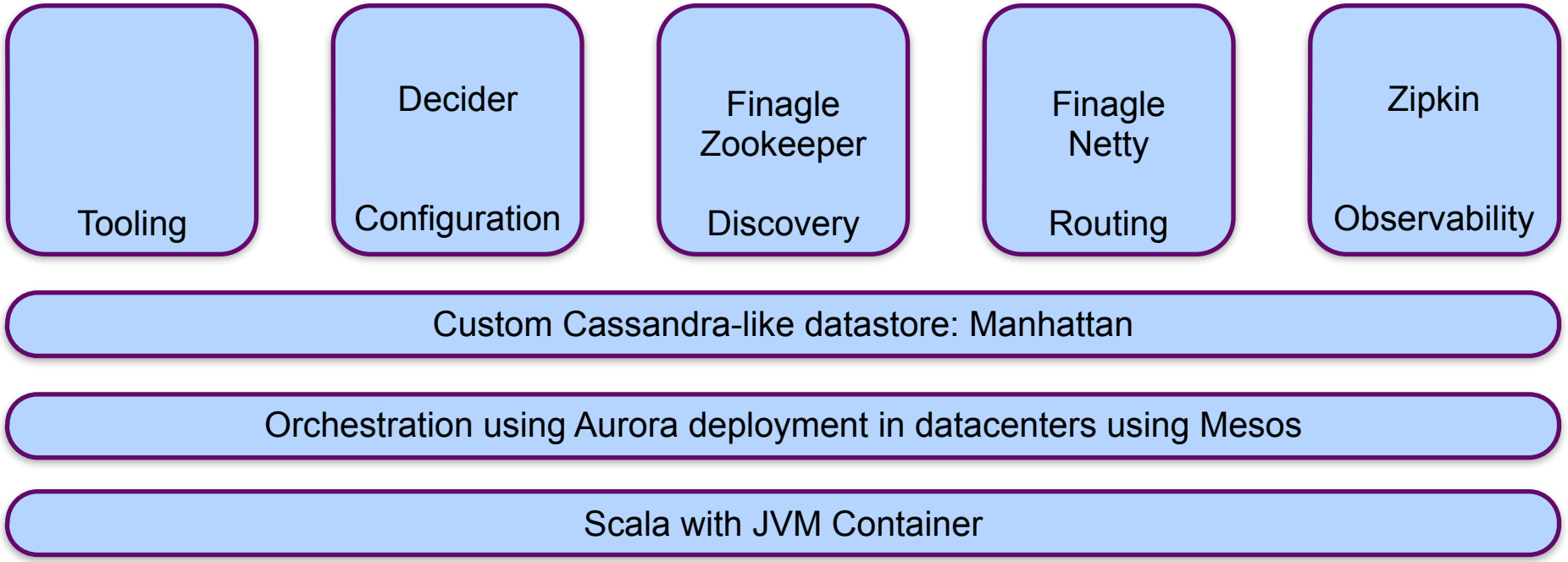

NETFLIX OSS
Netflix Open Source Software Center
Repositories
Powered By NetflixOSS
These companies are using and contributing to Netflix OSS Components
Email netflixoss@netflix.com to have your logo here.
STITCH FIX
KENZAN
SCHIBSTED
CLASSIFIED MEDIA SPAIN
Bitnet
RAPID7
BONOBOS
KNEWTON
rainforest
waze
nirmata
IBM
vennetics
swrve
KIXEYE
yammer
FullContact
flipkart.com
globus genomics
RIOT GAMES
coursera
yelp
Hotels.com
MORTAR
AnsWerS
YAHOO!
EUCALYPTUS
StumbleUpon
Maginatics
UserEvents
bazaarvoice:
OpenSCG
SUNCORP GROUP

NETFLIX

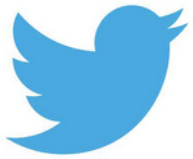


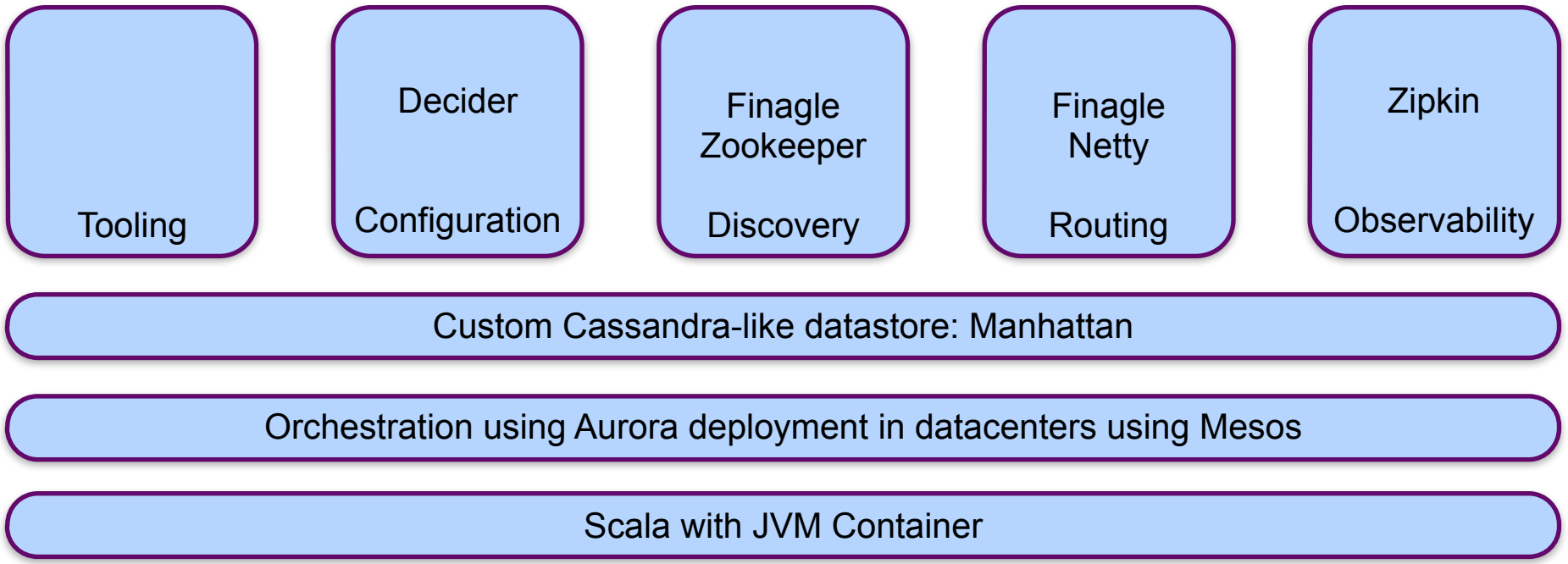
Twitter Microservices



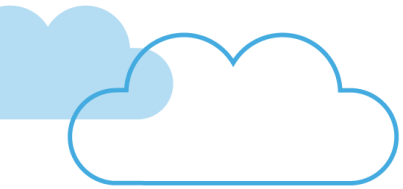


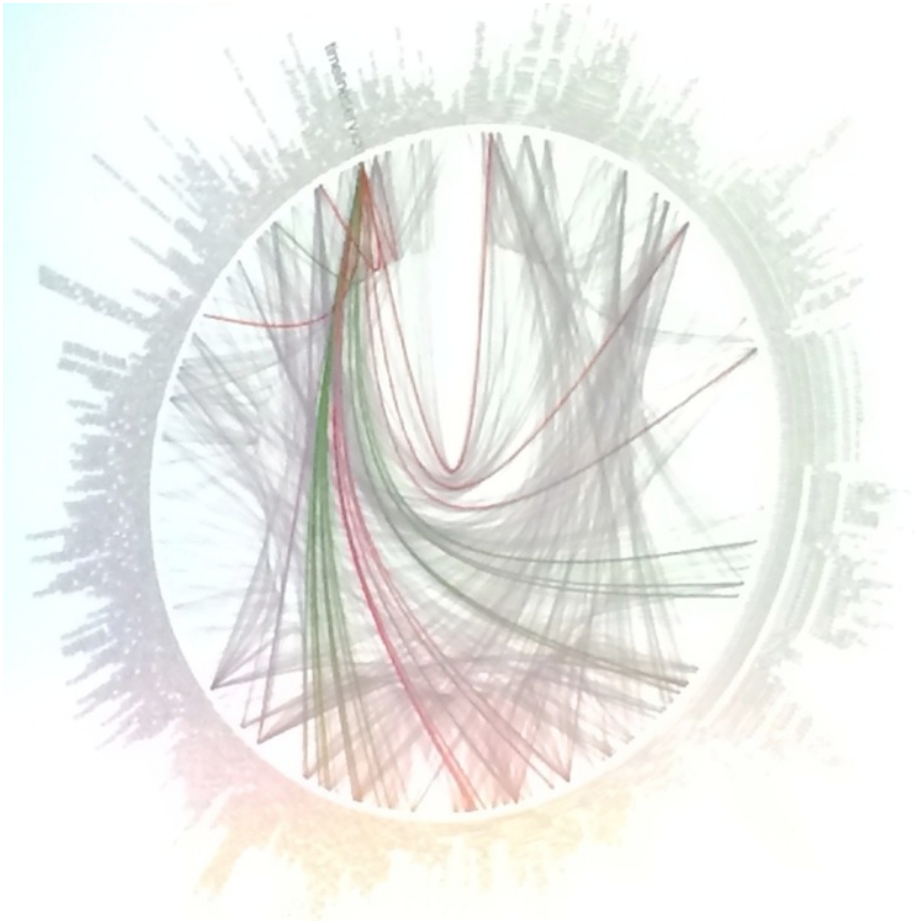
Twitter Microservices





Focus on efficient datacenter deployment at scale





Gilt Microservices

Ion Cannon
SBT
Rake

Tooling

Decider

Configuration

Finagle
Zookeeper

Discovery

Akka
Finagle
Netty

Routing

Zipkin

Observability

Datastores per Microservice using MongoDB, Postgres, Voldemort

Deployment on AWS

Scala and Ruby with Docker Containers

Gilt Microservices

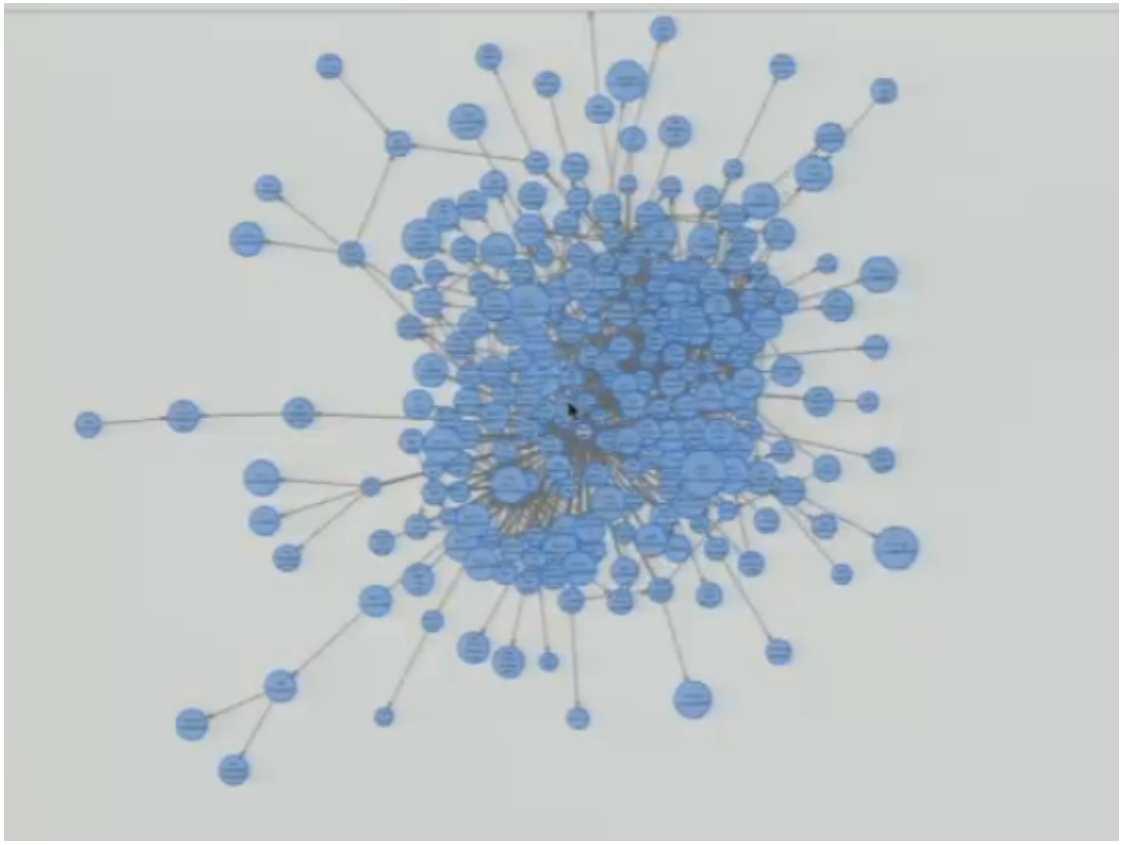
Datastores per Microservice using MongoDB, Postgres, Voldemort

Deployment on AWS

Scala and Ruby with Docker Containers

Focus on fast development with Scala and Docker

GILT



Hailo Microservices

Hubot
Janky
Jenkins

Tooling

Configuration

go-platform

Discovery

go-platform
RabbitMQ

Routing

Request trace

Observability

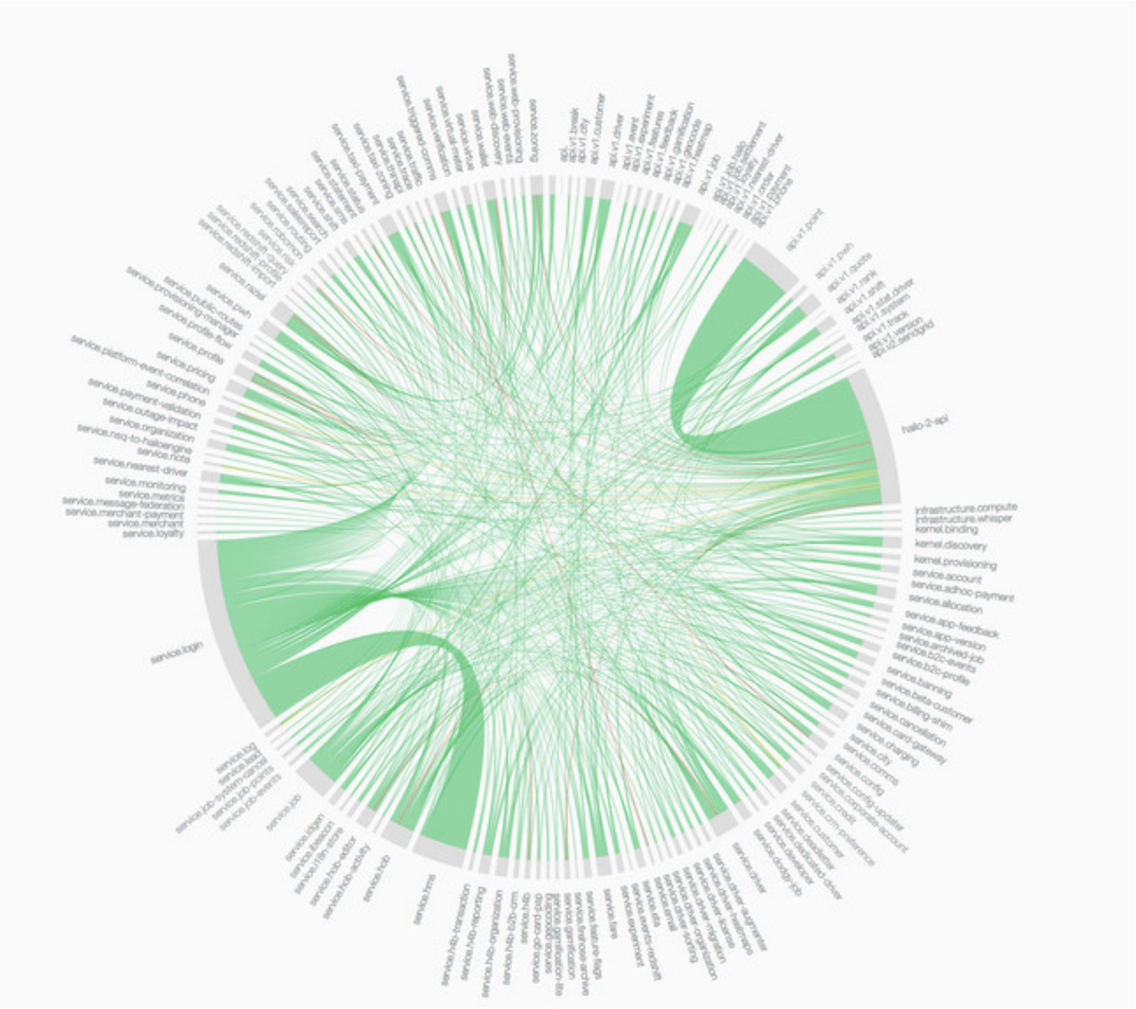
Datastore based on Cassandra

Deployment on AWS

Go using Docker

Hailo Microservices

Focus on fast development at scale using Go



Node.js Microservices

GROUPON

@WalmartLabs

http://senecajs.org/

amazon web services™

http://aws.amazon.com/lambda/

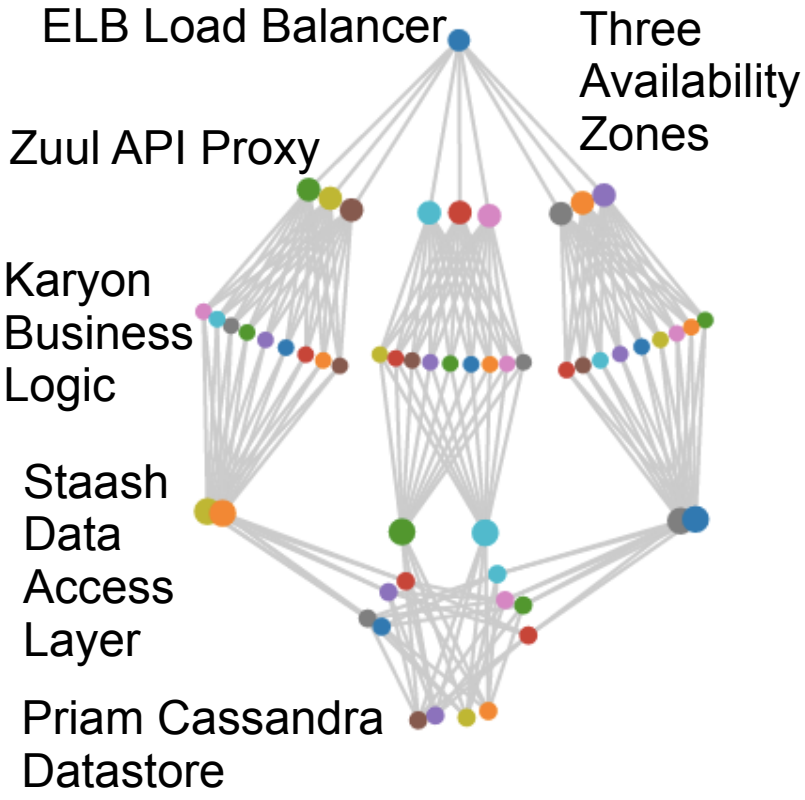
Several different approaches

Mostly small simple microservices

Focus on easy interface with presentation code in javascript

AWS Lambda - preview only

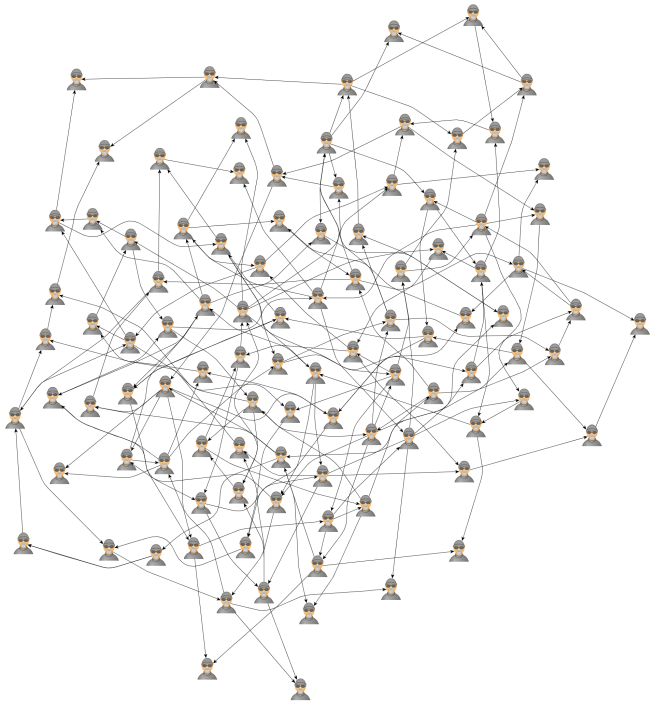
Adrian's Tinkering Projects



Model and visualize microservices
Simulate interesting architectures
Generate large scale configurations
Eventually stress test real tools

See github.com/adrianco/spigo
Simulate Protocol Interactions in Go
Visualize with D3

What is Spigo?



Creates and animates microservices
Single Go program on this laptop
Generates 100,000+ instances
About 250,000 messages/sec
Uses Go channels rather than http
Supports social network architecture
Supports NetflixOSS architecture
Simple code patterns to extend

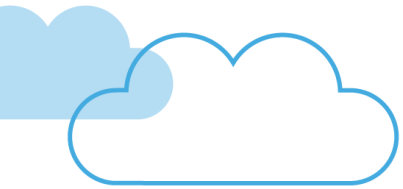
Why Build Spigo?

Generate test microservice configurations at scale
Stress monitoring tools display capabilities

Eventually (i.e. not implemented yet)
Dynamically vary configuration: autoscale, code push
Simulate microservice, zone, region failures
D3 websocket dynamic browser interface
Timescale: Monitorama Conference June 2015

Spigo Nanoservice Structure

```
func Start(listener chan gotocol.Message) {
        ...
        for {
                select {
                case msg := <-listener:
                        switch msg.Imposition {
                        case gotocol.Hello:
                                ...
                        case gotocol.NameDrop:
                                ...
                        case gotocol.Chat:
                                ...
                        case gotocol.GetResponse:
                                ...
                        case gotocol.Goodbye:
                                gotocol.Message{gotocol.Goodbye, nil, time.Now(), name}.GoSend(netflixoss)
                                return
                        }
                case <-chatTicker.C:
                        ...
                }
        }
}
```



What's Next?

Web Scale Characteristics

- *Brand new Microservices are deployed infrequently*
- *New versions deployed automatically/frequently*
- *No real need for general purpose orchestration*
- *Architectures use hundreds of microservices*
- *Each deployment is heavily customized*

Orchestration for Applications

- *Standard portable microservice based applications*
- *New versions deployed automatically/frequently*
- *Orchestration automated and standardized*
- *Architectures likely based on tens of microservices*
- *Opportunity: Docker Hub as **the** enterprise app store*

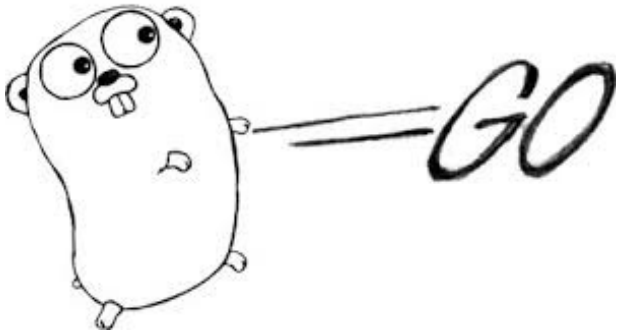
Next Generation Applications

Next Generation Applications

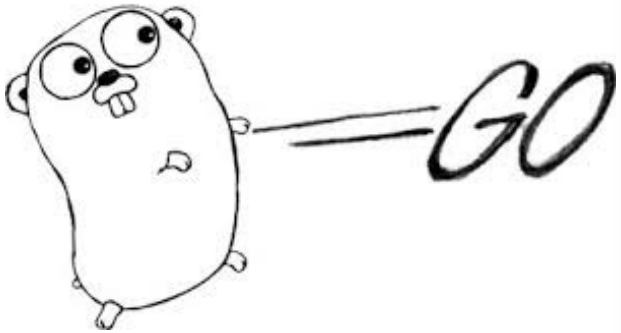
Fill in the gaps, rapidly evolving ecosystem choices

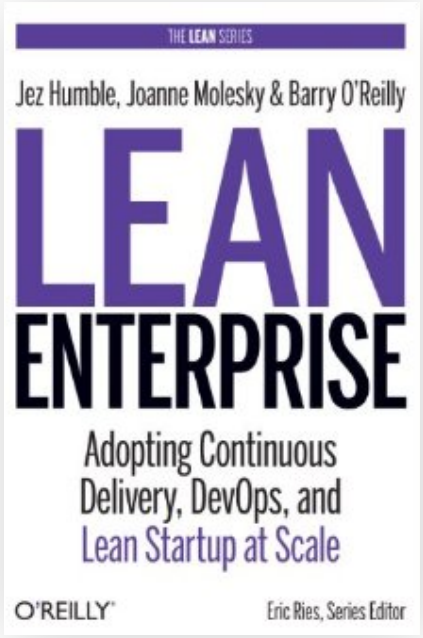
Forward Thinking

Forward Thinking

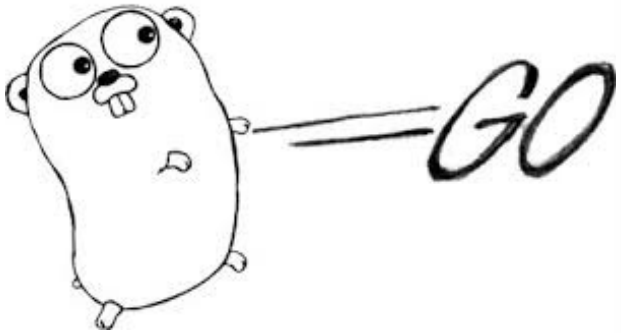


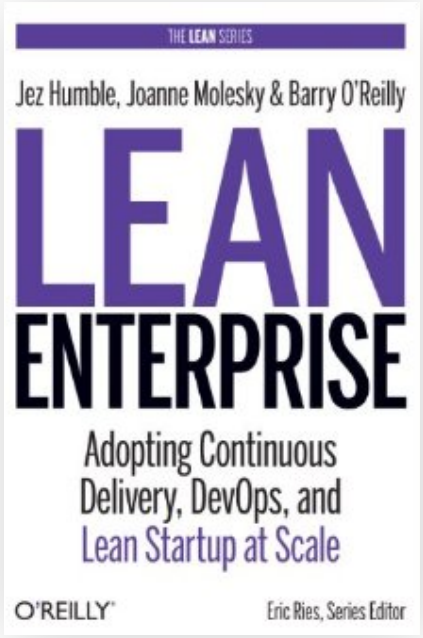
Forward Thinking

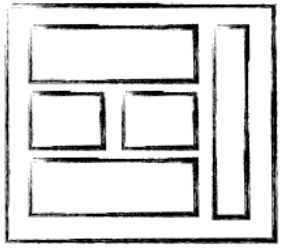




Forward Thinking







MONOLITHIC/LAYERED

MICRO SERVICES

http://eugenedvorkin.com/seven-micro-services-architecture-advantages/

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*

Disclosure: some of the companies mentioned may be Battery Ventures Portfolio Companies
See www.battery.com for a list of portfolio investments