Monitoring Microservices: A Challenge
Adrian Cockcroft @adrianco
Technology Fellow - Battery Ventures
March 2015
What does @adrianco do?

- Presentations at Conferences
- Presentations at Companies
- Program Committee for Conferences
- Maintain Deep Relationship with Cloud Vendors
- Tinkering with Technologies
- Technology Due Diligence on Deals
- Technical Advice for Portfolio Companies
- Networking with Interesting People
- Maintain Deep Relationship with Cloud Vendors
What does @adrianco do?

- Presentations at Conferences
- Presentations at Companies
- Program Committee for Conferences
- Tinkering with Technologies
- Maintain Deep Relationship with Cloud Vendors
- Technology Due Diligence on Deals
- Technical Advice for Portfolio Companies
- Networking with Interesting People
What does @adrianco do?

- Presentations at Conferences
- Presentations at Companies
- Program Committee for Conferences
- Maintain Deep Relationship with Cloud Vendors
- Tinkering with Technologies
- Technology Due Diligence on Deals
- Technical Advice for Portfolio Companies
- Networking with Interesting People
- Maintain Deep Relationship with Cloud Vendors
What does @adrianco do?

- Presentations at Conferences
- Presentations at Companies
- Program Committee for Conferences
- Maintain Deep Relationship with Cloud Vendors
- Tinkering with Technologies
- Networking with Interesting People
- Technology Due Diligence on Deals
- Technical Advice for Portfolio Companies
What does @adrianco do?

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

Loosely coupled service oriented architecture with bounded contexts
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.

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.
Speed
Speeding Up Deployments

Datacenter Snowflakes
- Deploy in months
- Live for years
Speeding Up Deployments

Datacenter Snowflakes
- Deploy in months
- Live for years

Virtualized and Cloud
- Deploy in minutes
- Live for weeks
Speeding Up Deployments

Datacenter Snowflakes
• Deploy in months
• Live for years

Virtualized and Cloud
• Deploy in minutes
• Live for weeks

Container Deployments
• Deploy in seconds
• Live for minutes/hours
Speeding Up Deployments

Datacenter Snowflakes
- Deploy in months
- Live for years

Virtualized and Cloud
- Deploy in minutes
- Live for weeks

Container Deployments
- Deploy in seconds
- Live for minutes/hours

AWS Lambda Events
- Respond in milliseconds
- Live for seconds
Speeding Up Deployments

- **Datacenter Snowflakes**
  - Deploy in months
  - Live for years

- **Virtualized and Cloud**
  - Deploy in minutes
  - Live for weeks

- **Container Deployments**
  - Deploy in seconds
  - Live for minutes/hours

- **AWS Lambda Events**
  - Respond in milliseconds
  - Live for seconds

*Measuring CPU usage once a minute makes no sense for containers... Coping with rate of change is the first challenge for monitoring tools.*
Scale
It’s much more challenging than just a large number of machines

<table>
<thead>
<tr>
<th>A Possible Hierarchy</th>
<th>How Many?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continents</td>
<td>3 to 5</td>
</tr>
<tr>
<td>Regions</td>
<td>2-4 per Continent</td>
</tr>
<tr>
<td>Zones</td>
<td>1-5 per Region</td>
</tr>
<tr>
<td>Services</td>
<td>100’s per Zone</td>
</tr>
<tr>
<td>Versions</td>
<td>Many per Service</td>
</tr>
<tr>
<td>Containers</td>
<td>1000’s per Version</td>
</tr>
<tr>
<td>Instances</td>
<td>10,000’s</td>
</tr>
</tbody>
</table>
Flow
Some tools can show the request flow across a few services.
But interesting architectures have a lot of microservices! Flow visualization is a challenge.

See http://www.slideshare.net/LappleApple/gilt-from-monolith-ruby-app-to-micro-service-scala-service-architecture
Failures
Simple Netflix OSS style microservices architecture on three AWS Availability Zones

ELB Load Balancer

Zuul API Proxy

Karyon Business Logic

Staash Data Access Layer

Priam Cassandra Datastore
ELB Load Balancer

Zuul API Proxy

Karyon Business Logic

Staash Data Access Layer

Priam Cassandra Datastore

Simple NetflixOSS style microservices architecture on three AWS Availability Zones
What should you do?

What should monitors show?

Simple Netflix OSS style microservices architecture on three AWS Availability Zones

Zone partition/failure
What should you do?
What should monitors show?
Simple Netflix OSS style microservices architecture on three AWS Availability Zones

Zone partition/failure
What should you do?
What should monitors show?

By design, everything works with 2 of 3 zones running. This is not an outage, inform but don’t touch anything! Halt deployments perhaps?
Simple NetflixOSS style microservices architecture on three AWS Availability Zones

By design, everything works with 2 of 3 zones running. This is not an outage, inform but don’t touch anything! Halt deployments perhaps?

Challenge: understand and communicate common microservice failure patterns.
Testing
Testing monitoring tools at scale gets expensive quickly...
Simulated Microservices

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

See [github.com/adrianco/spigo](https://github.com/adrianco/spigo)
Simulate Protocol Interactions in Go
Visualize with D3
Code for an architecture

Create(cname, "priamCassandra", Regions, priamCassandracount, cname)
Create(sqname, "store", Regions, mysqlcount, sqname)
Create(evname, "store", Regions, evcachecount)
Create(tname, "staash", Regions, staashcount, sqname, evname, cname)
Create(pname, "karyon", Regions, phpcount, tname)
Create(nname, "karyon", Regions, nodecount, tname)
Create(zuname, "zuul", Regions, zuulcount, pname, nname)
Create(elbname, "elb", Regions, 0, zuuname)
Create(dns, "denominator", 0, 0, elbname)
My challenge to you: Build your architecture in Spigo. Stress monitoring tools with it. Fix monitoring for microservices!
Office Hours Lunch: 12:15pm, Ballroom A

- **Microservices Challenges**
  - Speed and Scale
  - Flow and Failures
  - Testing and Simulation

- Battery Ventures [http://www.battery.com](http://www.battery.com)
- Adrian’s Tweets [@adrianco](https://twitter.com/adrianco) and Blog [http://perfi](http://perfi)
- Slideshare [http://slideshare.com/adriancockcroft](http://slideshare.com/adriancockcroft)

Disclosure: some of the companies mentioned may be Battery Ventures Portfolio Companies. See [www.battery.com](http://www.battery.com) for a list of portfolio investments.