Microservices Architectures

By Camilo Aguilar - Former Ceiba employee and founder at Hooklift, Inc.
@c4milo at Twitter

What is it?

“It is a service-oriented architecture composed of loosely coupled elements that have bounded contexts.”

– Adrian Cockcroft, Netflix, Inc.

Bound what?

Let’s just say that they allow you to update services independently.

There is no need for coordination among teams in order to make changes.

Evolution

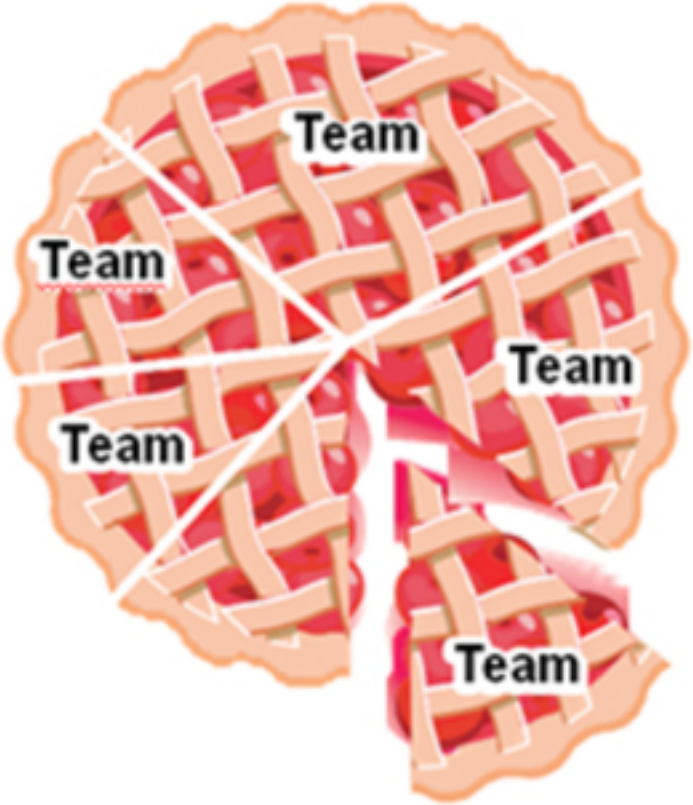
1990s and earlier

Pre-SOA (monolithic)
Tight coupling

For a monolith to change, all must agree on each change. Each change has unanticipated effects requiring careful testing beforehand.

2000s

Traditional SOA
Looser coupling



Elements in SOA are developed more autonomously but must be coordinated with others to fit into the overall design.

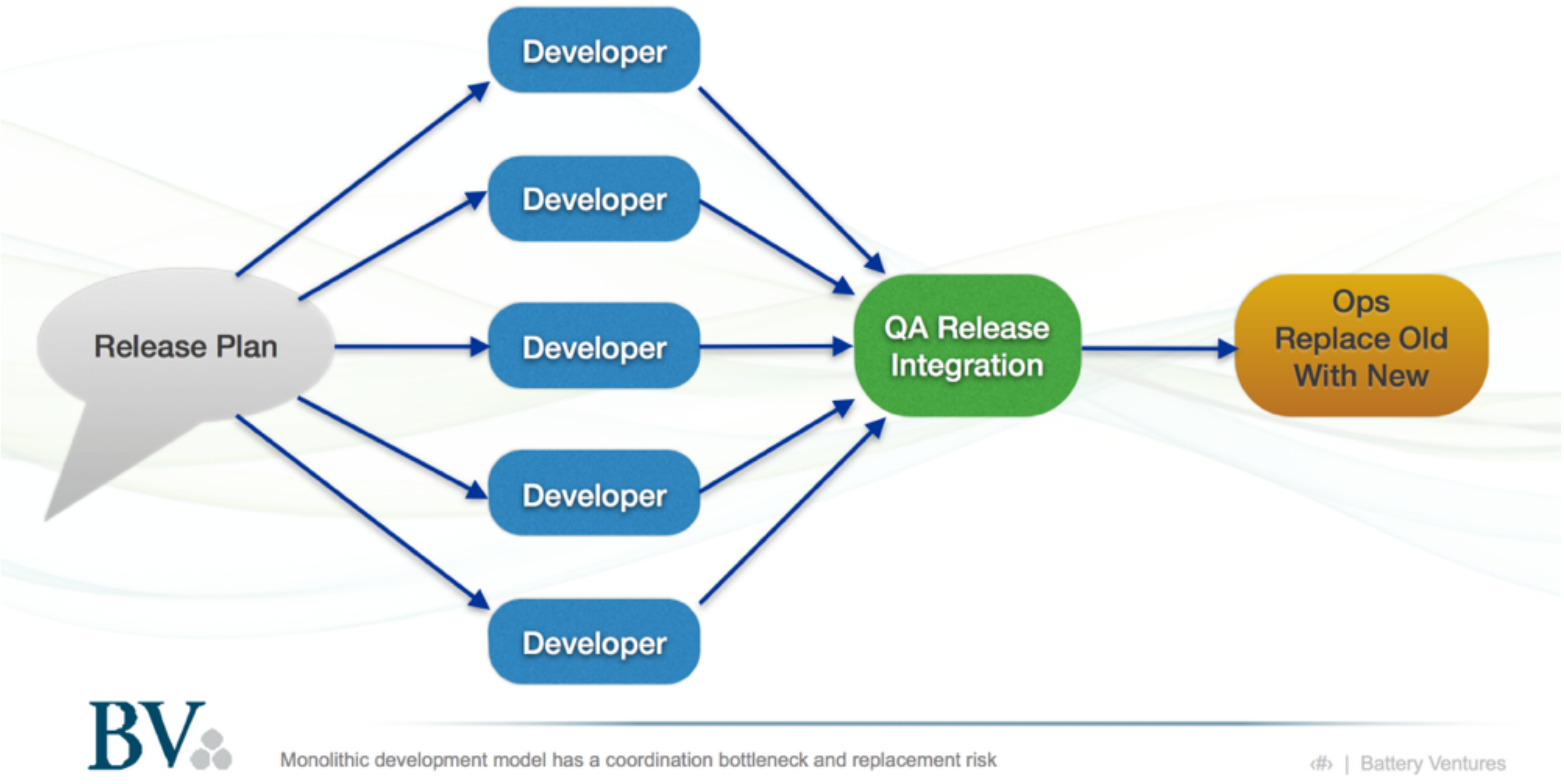
2010s

Microservices
Decoupled

Developers can create and activate new microservices without prior coordination with others. Their adherence to MSA principles makes continuous delivery of new or modified services possible.

Image source: http://www.pwc.com/us/en/technology-forecast/2014/cloud-computing/features/microservices.html

Traditional Release Plan



Release Plan

Developer

Developer

Developer

Developer

Developer

QA Release Integration

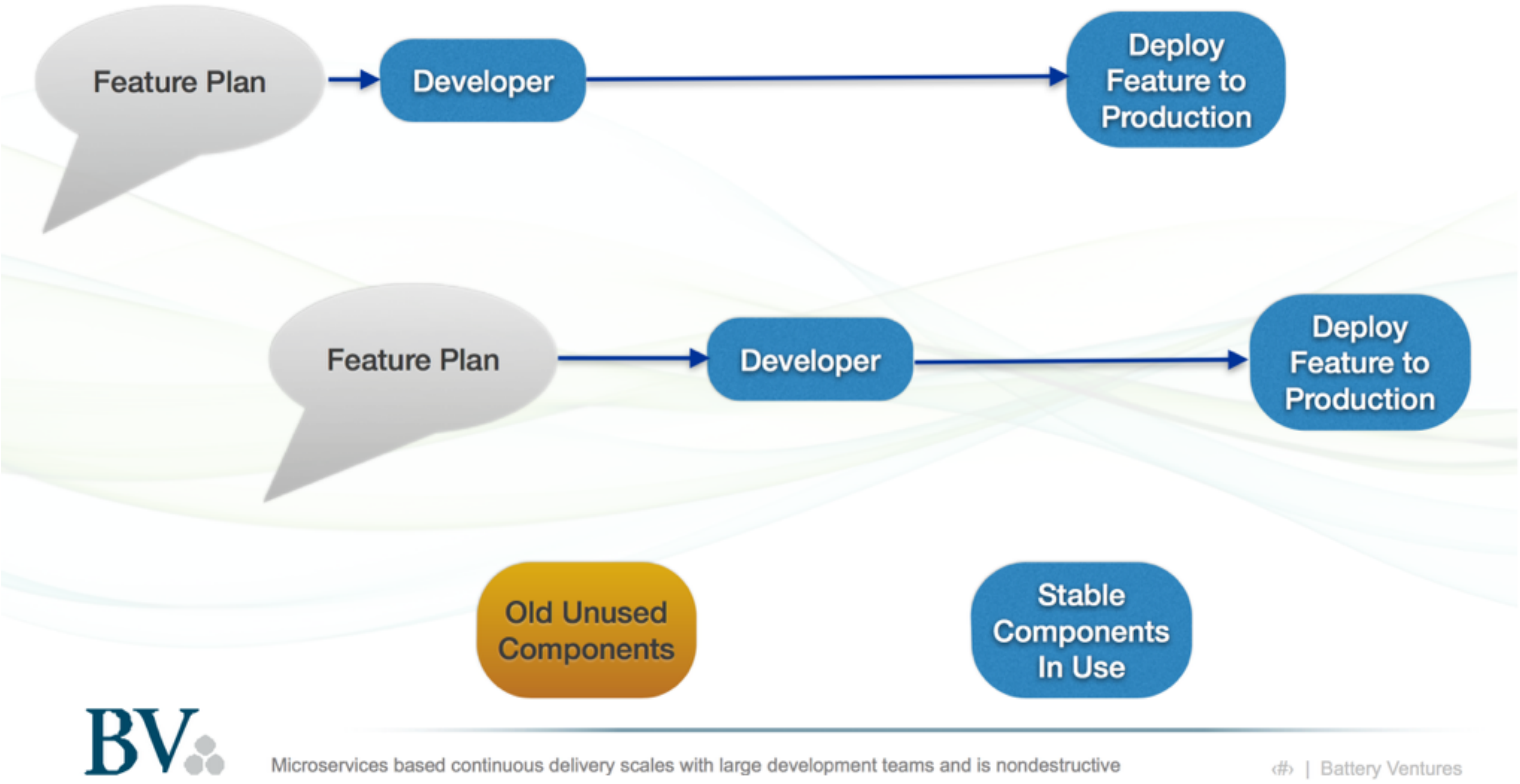
Ops Replace Old With New

Monolithic development model has a coordination bottleneck and replacement risk

‹#› | Battery Ventures

Image source: http://www.infoq.com/presentations/migration-cloud-microservices

Continuous Delivery



Feature Plan

Developer

Deploy Feature to Production

Feature Plan

Developer

Deploy Feature to Production

Old Unused Components

Stable Components In Use

BV

Microservices based continuous delivery scales with large development teams and is nondestructive

‹#› | Battery Ventures

Image source: http://www.infoq.com/presentations/migration-cloud-microservices

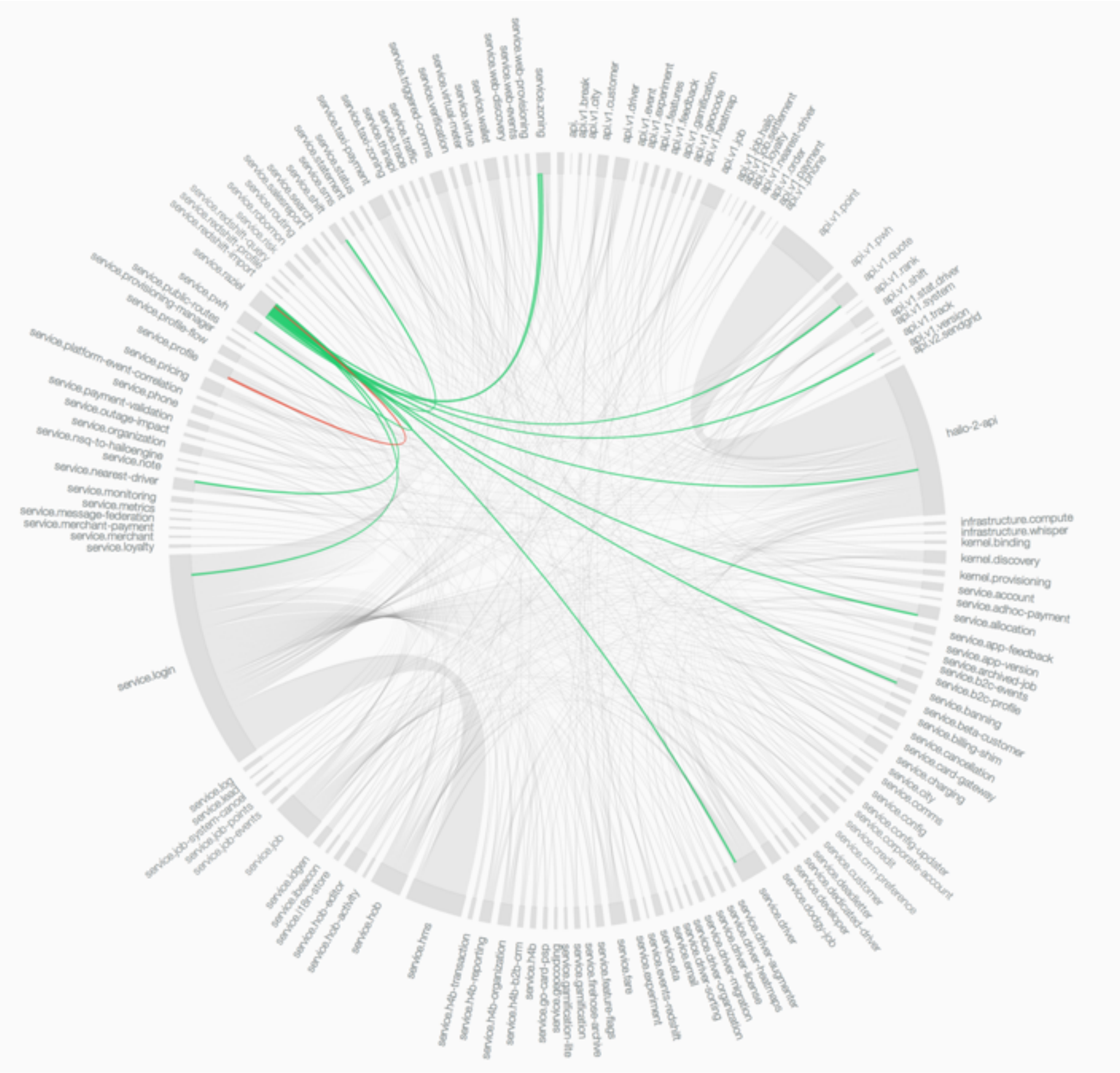
Benefits

- Delivers business value much quicker
- Less down time and lower risk
- No waiting for IT to provision VMs, no meetings
- Developers:
  - Feel more empowered and less frustrated
  - Run their own code
  - Increase product ownership
- Fosters a high trust culture
- Quick roll-backs (seconds)
- Enables progressive refactoring as opposed to full rewrites
- Enables easier experimentation:
  - New databases, programming languages, libraries, etc
  - Product feature A/B testing

Challenges

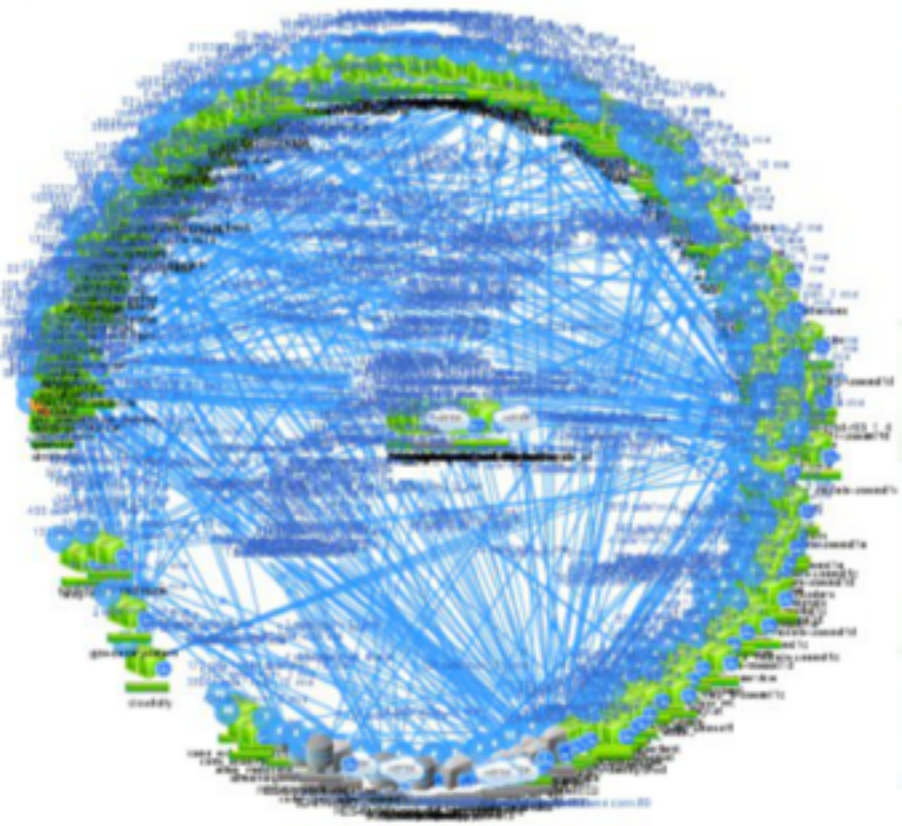
- Increased operational complexity
- You are moving into the distributed systems realm
  - Network latency and partitions
  - Message formats
  - Load balancing
  - Fault tolerance
- Conway’s Law

Wheel of Doom

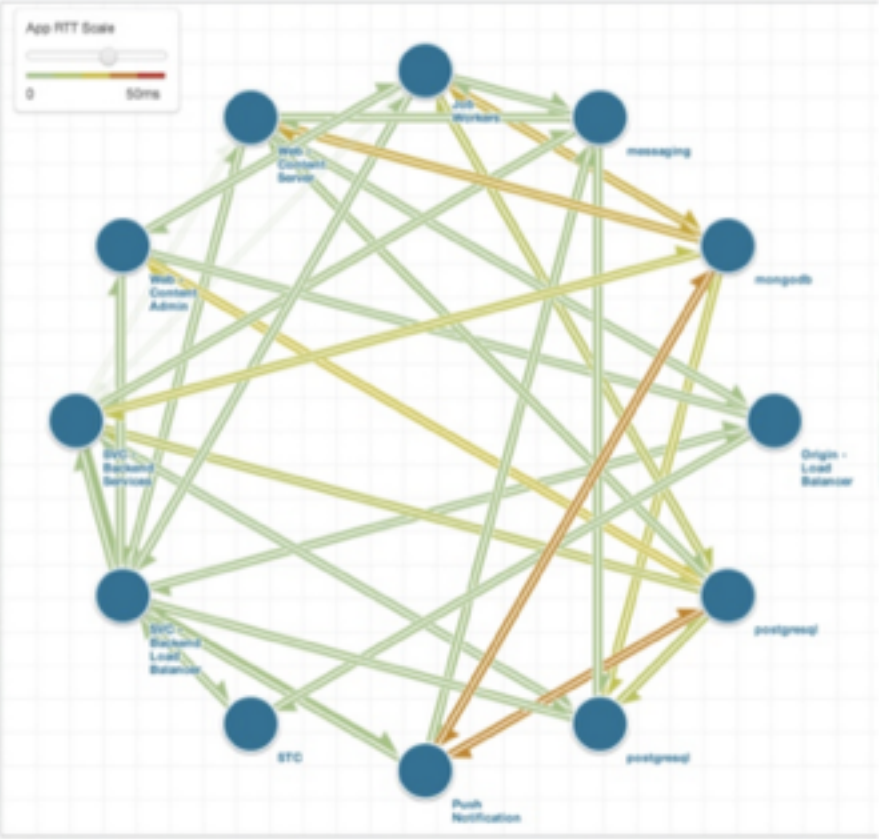


Hooklift, Inc.

“Death Star” Architecture Diagrams



Netflix



Gilt Groupe (12 of 450)

Twitter

BV

As visualized by Appdynamics, Boundary.com and Twitter internal tools

Image source: http://www.infoq.com/presentations/migration-cloud-microservices

LOOKS GOOD TO ME
memecrun

Deployments per day

- Etsy: 50
- Flickr: 10
- HubSpot: 300
- Amazon: 7800
- Netflix: 100
- Twitter: 7200

Microservices enables delivering business value at full speed

How to cope with the new complexity?

Must haves

- Excellent runtime visibility at all times
  - Grafana, InfluxDB, cAdvisor, Sysdig etc
- Log aggregation and search
  - papertrail, loggly, ElasticSearch, etc
- Service discovery:
  - Consul or Etcd
- Continuous Integration and testing
  - TravisCI, CircleCI, Jenkins, etc
- Distributed commit log: Kafka
- Circuit breakers

Nice to have

- Blue/Green deployments
- Canary deployments
- Service isolation aka containers
  - Docker, Rocket, systemd-nspawn, etc
- Continuous deployments
- Feature flags
- A way of doing distributed consensus and locking
  - Zookeeper
  - Consul
  - EtcD

How about DevOps?

It is a concept very often misunderstood

DevOps is not a team or role, it is an organizational change in practices and communication to enable agility delivering business value

Q&A