a Diminishing Delta between Development & Delivery
the Macro of Microservices

Josh Long (龙之春)
@starbuxman
jlong@pivotal.io
github.com/joshlong
say open source!
Josh Long (龙之春)
The Spring Developer Advocate

- http://cloudnativejava.io
- @starbuxman
- jlong@pivotal.io
- Java Champion
- open-source contributor
  (Spring Boot, Spring Cloud, Spring Integration, Vaadin, Activiti, etc etc)
microservices
continous delivery
agility
cloud computing
continuous integration (the feedback loop)
Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas
We follow these principles:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity—the art of maximizing the amount of work not done—is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
to the cloud!
Product team using monolithic delivery

Prod Mgr  UX  Dev  QA  DB Admin  Sys Admin  Net Admin  SAN Admin

Source:
Adrian Cockcroft
http://www.slideshare.net/adrianco
The First Way emphasizes the performance of the entire system, as opposed to the performance of a specific silo of work or department — this as can be as large a division (e.g., Development or IT Operations) or as small as an individual contributor (e.g., a developer, system administrator).

Gene Kim - “the Devops Handbook”
http://itrevolution.com/the-three-ways-principles-underpinning-devops/
cloud helps automate operations

Simon Wardley, Value Chain Mappings
http://www.computerweekly.com/opinion/Value-chain-mapping-learning-to-use-IT-as-a-strategic-weapon
Bezos: Approximately nine years ago we were wasting a lot of time internally because, to do their jobs, our applications engineers had to have daily detailed conversations with our networking infrastructure engineers. Instead of having this fine-grained coordination about every detail, we wanted the data-center guys to give the apps guys a set of dependable tools, a reliable infrastructure that they could build products on top of.

The problem was obvious. We didn’t have that infrastructure. So we started building it for our own internal use. Then we realized, “Whoa, everybody who wants to build web-scale applications is going to need this.” We figured with a little bit of extra work we could make it available to everybody. We’re going to make it anyway—let’s sell it.
THEY SAID I COULD BE ANYTHING

SO I BECAME A CLOUD

-> (IBM Bluemix™ | HP Helion | Pivotal CF ..)
Imagine if architects had to be the janitor for every building they designed. This is how the development team felt prior to moving to Windows Azure.

Duncan Mackenzie - Nov 07, 2011
http://infoq.com/articles/Channel-9-Azure
the platform improves governance

...less tickets, more software, faster, safer
writing cloud-native software
a monolith might be the right answer
microservices enable runtime scale

Martin L. Abbot, Michael T. Fisher, *the Art of Scalability*

http://theartofscalability.com/
microservices enable feature teams

Source:
Adrian Cockcroft
http://www.slideshare.net/adrianco
Mel Conway’s law:

..organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations

corollary:

you can refactor your organization to improve your systems
Bounded Context is a central pattern in Domain-Driven Design. It is the focus of DDD's strategic design section which is all about dealing with large models and teams. DDD deals with large models by dividing them into different Bounded Contexts and being explicit about their interrelationships.
modern computer science values **quantity, velocity** over **quality**

agile programming
unit testing & TDD
scrum
continuous integration
...

if you wait for all the data to make an informed decision, it may be too late to make any decision at all
iteration enables OODA

Source: Adrian Cockcroft
http://www.slideshare.net/adrianco
OODA requires continuous delivery

source:

**Continuous Delivery**: Reliable Software Releases Through Build, Test, and Deployment Automation / Jez Humble, David Farley.

“We may deploy 20 times a day, but we wouldn’t deploy 20 times a day if we went down 20 times a day. The only reason we got to 20 times a day, is that the first time we deployed 5 times a day, it worked out.”

John Allspaw - Flickr (not Etsy)
http://www.slideshare.net/jallspaw/10-deploys-per-day-dev-and-ops-cooperation-at-flickr
so how do i build it?
spring boot

supports rapid development of production-ready applications and services

it’s going to take an API.. lots of them!
microservices introduce complexity

- service registration & discovery?
  (and client-side load balancing)

- joined up views of data?

- centralized configuration management?

- distributed tracing?

- stream processing?

- single sign-on
distributed systems are hard
These logos are all *trademark/copyright* their respective owners (T-B, L-R):

Netflix, amazon.com, Apache Software Foundation, Cloud Foundry, Hashicorp

they are ALL great organizations and we love their open-source and their APIs!!
Questions?

Josh Long (龙之春)
@starbuxman
jlong@pivotal.io
speakerdeck.com/joshlong
github.com/joshlong
http://spring.io