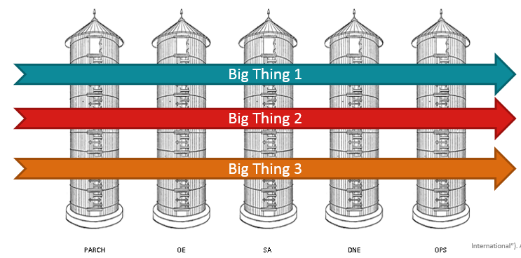
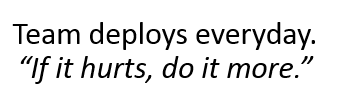
DevOps A Horse’s Perspective

Just a horse, not a unicorn, 1000 deploys per/second not feasible.

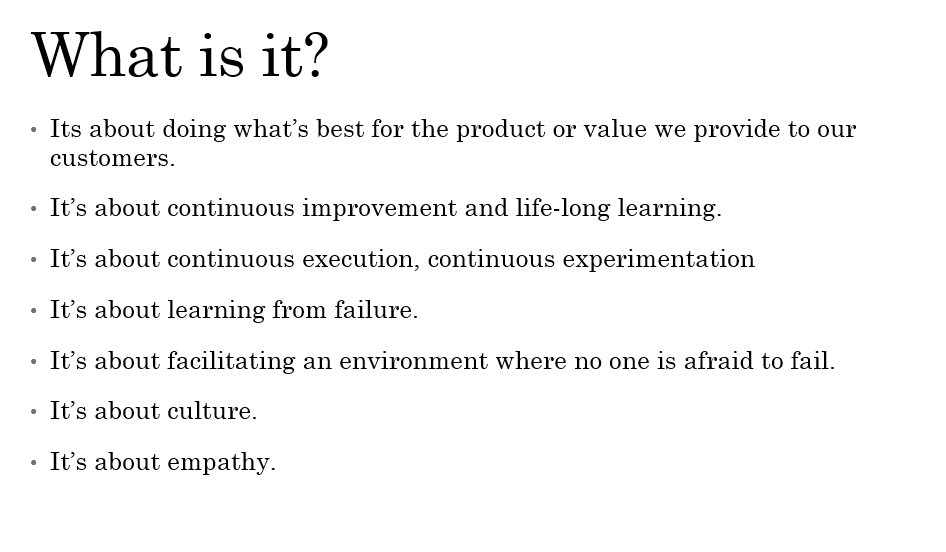
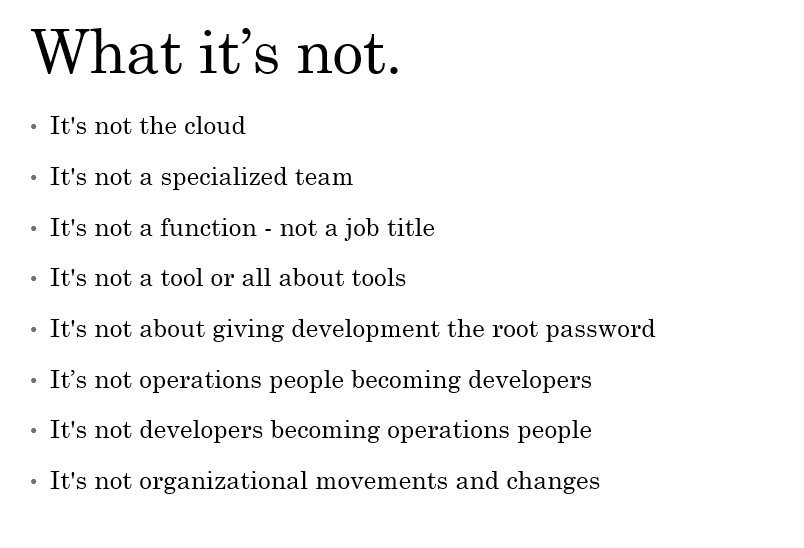
* History – How I Got Here?
  + Contentious relationship existed between development and operations.
  + Left development because I was frustrated that it took so much time to get my products into production.
    - * Thought I would get over there fix “their” problems.
      * Discovered, multiple teams involved in snowflake environment support.
      * The practice team wasn’t the same as the game time team.
  + Segregated technical and business functions
    - * Development
      * Operations (Operational Waterfall)

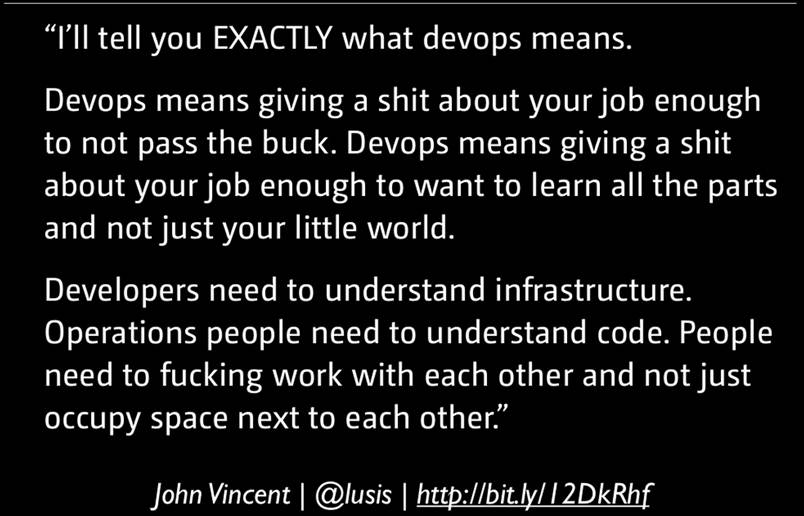
Infrastructure Ops

Product Ops

* + - * Product Management
  + Every technology under the sun
    - * Solaris, Windows, Linux
      * Apache, IIS, TCServer, etc.
      * Oracle, DB2, SQL Server
* How we got better
  + - We read and we studied.
    - Created a self-improvement project
      * + 2 week iterations, planning and demos
        + Put everything into a repository (configurations, scripts, etc.) Infrastructure as Code
        + Turned over repeatable and automatable tasks to the product team wherever possible. **Control our destiny**
        + Started to engineer solutions for issues related to product configuration and deployment
    - Built a strong relationship with development (built empathy)
      * + Development began to see operational issues, and usability problems
        + The relationship is less **adversarial** and more **supportive**.
        + Individuals are cross-trained on each other’s concerns – empathy
    - Combined the development support team with the production support team.
      * + Operations becomes familiar with new features in QA, rather than **in front** of and with the customer – shorter feedback loop
        + More opportunities to fail, iterate, and improve.
        + **Incented** innovation and optimization.
        + Environment homogenization and assimilation – no snowflakes
        + Deployment methodologies, automation, monitoring, and management tested continuously.
        + Overall Quality improvements, “it” goes in better, and with less issues.
        + Maintenance window changed from once a week to everyday
      * Expanded Sphere of Influence
        + Machine Imaging
        + DEG and 3rd Party Application Installations
        + Route Adds – requires heightened security access
        + Database Data Script Execution
        + Load Balancer Node Disablement
        + OS and Security Patching
        + Requesting access to technology specific dashboards and consoles.

Aides in troubleshooting and encourages technical learning.

* How we need to improve
  + - Sharing ideas, code, Community of Practice, etc.
    - Test Driven Infrastructure
    - Blue – green deployments
    - Combining DevOps Scrum – planning, standups, boards, and backlogs
    - Making work visible (it’s still hidden)
    - Poor measurements, velocity, capacity, etc.
    - Still not widely adopted
* Advice
  + Do embark on a agile self-improvement effort.
  + Do mix in or schedule your recurring maintenance.
  + Do define a primary purpose and measure it.
  + Do process map.
  + Do focus on Quality first.
  + Do start a book club.



Book List

* + - *Scaling Lean & Agile Development*: Thinking & Organizational Tools for Large-Scale Scrum, Craig Larman
    - *Continuous Delivery*: Reliable Software Releases through Build, Test, and Deployment Automation, Jez Humble and David Farley
    - *The Phoenix Project*: A Novel About IT, DevOps, and Helping Your Business Win, Gene Kim, Kevin Behr, and George Spafford.
    - *The Lean Primer*, Craig Larman and Bas Vodde