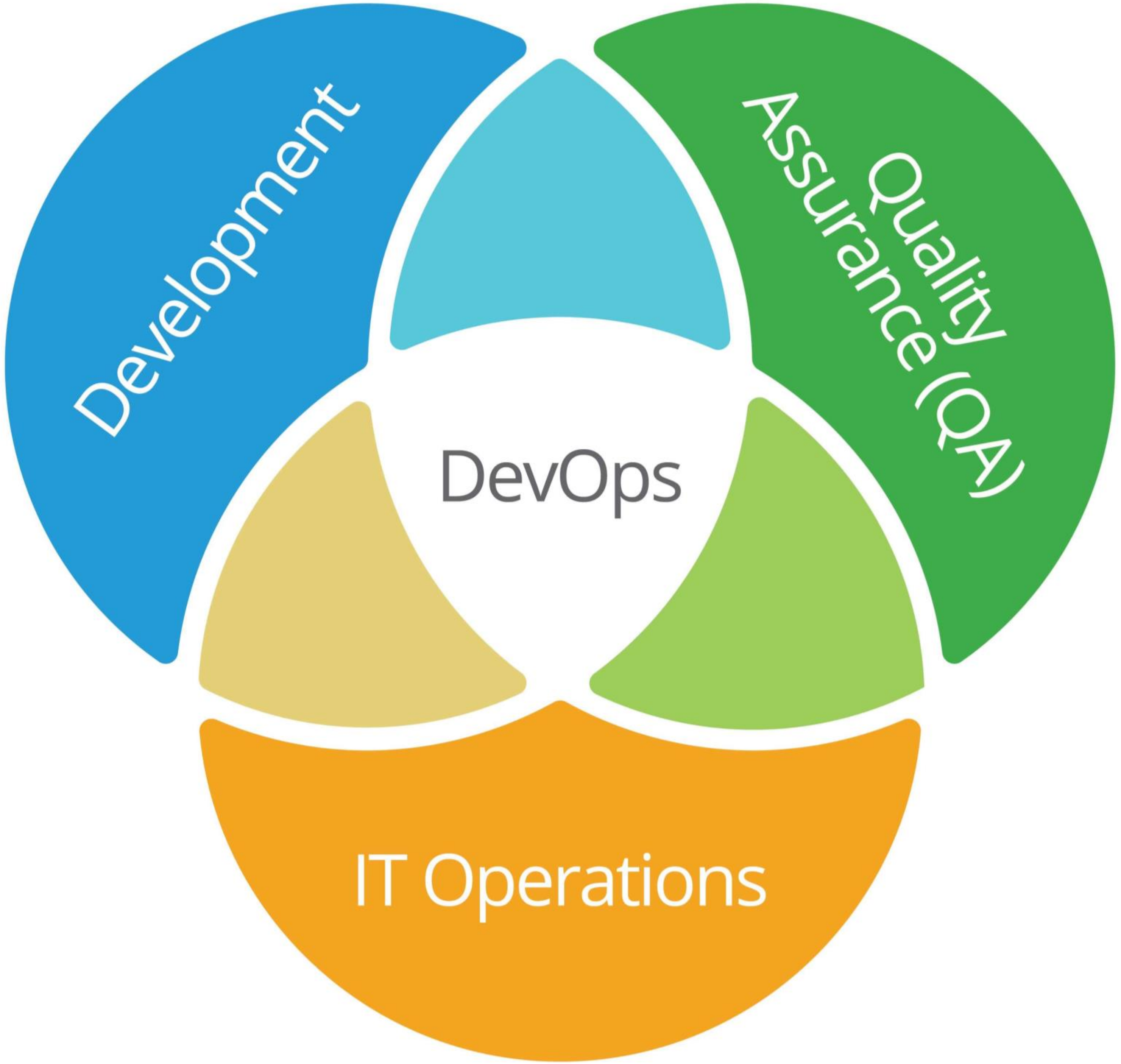
Demystify DevOps

Srikanth Tanniru (tanniru.srikanth@live.com)

DevOps Venn Diagram



What DevOps is Not ?

- It’s Not combining the Development and Operations team and leaving it at that
- It’s Not replacing or removing Ops
- It’s Not NoOps
- It’s Not (Just) Tools
- It’s Not (Just) Culture
- It’s Not (Just) Devs and Ops
- It’s Not (Just) A Job Title
- It’s Not a one-size-fits-all strategy
- It’s Not SRE
- It’s Not a separate team, nor department
- It’s Not (necessarily) Agile or Lean
- It’s Not an end goal
- It’s Not sacrificing governance and compliance
- It’s Not just automation
- It’s Not using Cloud
- It doesn’t mean that anyone can release any change to production at any time
- It doesn’t mean “Developers in Charge”
- There is no single DevOps playbook
- It’s not a plan, it’s a reaction
- It’s not a judgment
- It’s not meant to be an exclusive club
- It's not just a bunch of really smart people
- It's not a product
- It's not a run around traditional IT

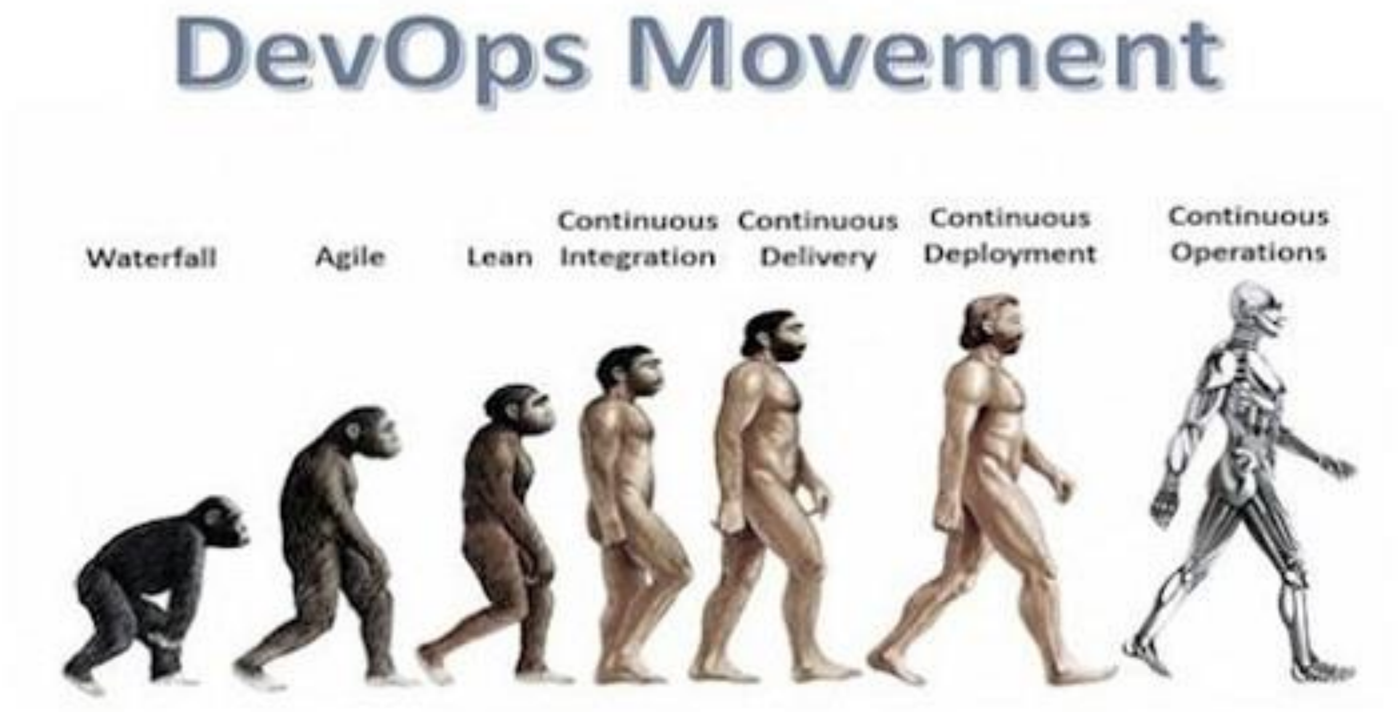
What DevOps Is !?

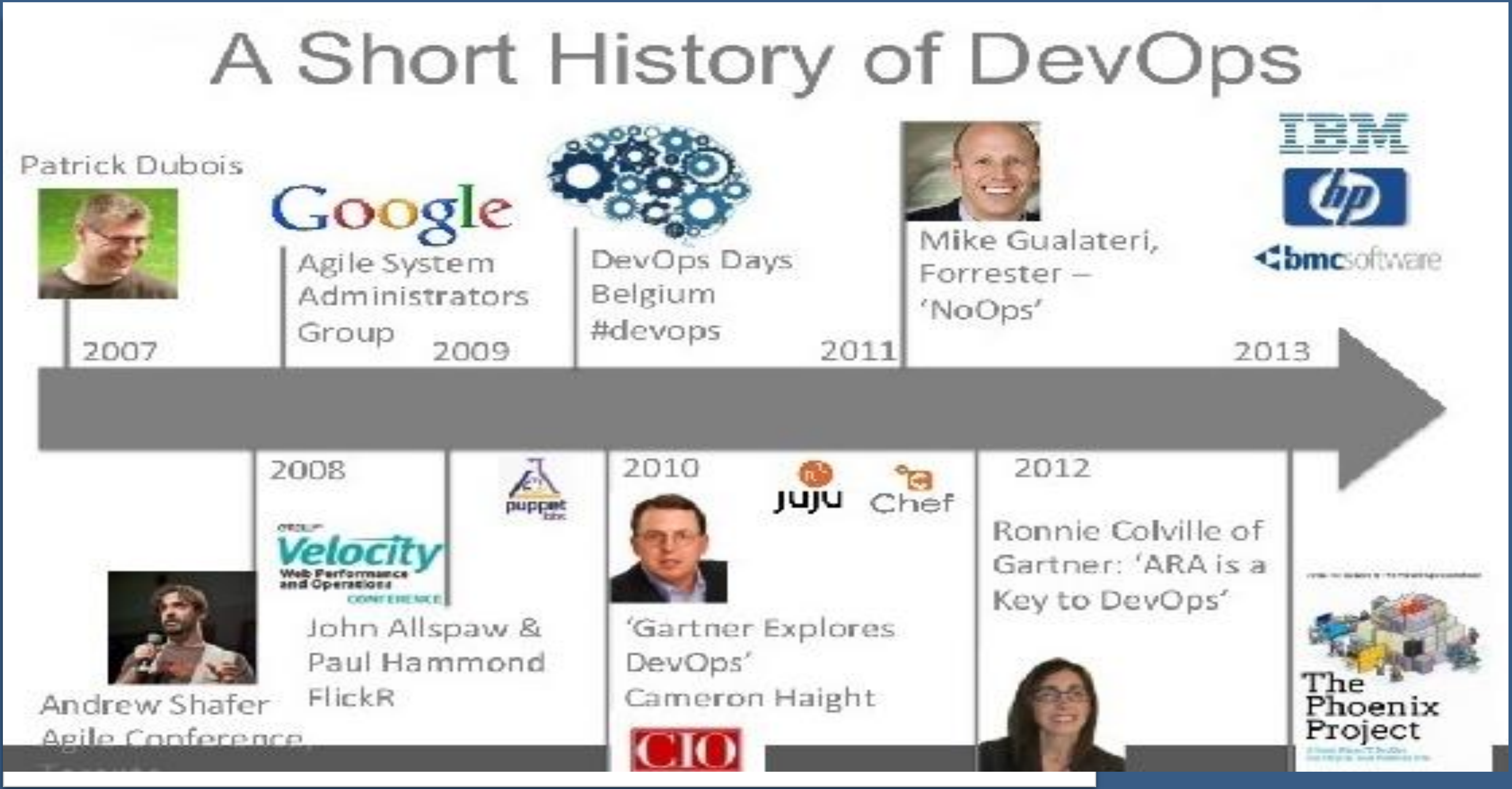
DevOps is the process of removing all friction between the developer and customer value.

- **Value**: typically, the products and services customers use.
  - *e.g.,* software or a web site.
- **Customer**: a value consumer.
  - *e.g.,* target audience or end user.
- **Developer**: value creator and contributor.
- **Friction**: anything that slows, diminishes, or reduces the value delivery.
  - *e.g.,* manual hand-offs, separation of duties, silos of responsibility, or isolation from the entire value stream.
- **Process**: the methodology to accomplish work

“DevOps is not about a technology; DevOps is about a business problem.

DevOps is a culture shift or a movement that encourages great communication and collaboration (aka teamwork) to foster building better-quality software more quickly with more reliability.





A Short History of DevOps

2007
Patrick Dubois

Andrew Shafer
Agile Conference

2008
Velocity Web Performance and Operations Conference
John Allspaw &
Paul Hammond
FlickR

2009
Google
Agile System Administrators Group

puppet labs

DevOps Days Belgium #devops

2010
'Gartner Explores DevOps'
Cameron Haight
CIO

juju
Chef

2011
Mike Gualateri, Forrester – 'NoOps'

2012
Ronnie Colville of Gartner: 'ARA is a Key to DevOps'

2013
IBM
hp
bmcsoftware
The Phoenix Project

https://groups.google.com/g/agile-system-administration/c/HKCTSee2u4w

Common DevOps Myths

- MYTH - DEVOPS IS ONLY FOR STARTUPS
- MYTH - DEVOPS REPLACES AGILE
- MYTH - DEVOPS IS INCOMPATIBLE WITH ITIL
- MYTH - DEVOPS IS INCOMPATIBLE WITH INFORMATION SECURITY AND COMPLIANCE
- MYTH - DEVOPS MEANS ELIMINATING IT OPERATIONS, OR “NOOPS”
- MYTH - DEVOPS IS JUST “INFRASTRUCTURE AS CODE” OR AUTOMATION
- MYTH - DEVOPS IS ONLY FOR OPEN-SOURCE SOFTWARE
- MYTH - DevOps requires Agile
- MYTH - DevOps can't work with legacies
- MYTH - DevOps is only for continuous delivery

Common DevOps Myths

- MYTH - DevOps is only for continuous delivery
- MYTH - DevOps requires new tools
- MYTH - DevOps is a skill
- MYTH - DevOps is a software
- MYTH - It is exclusive to native internet companies
- MYTH - There is no DevOps without the cloud
- MYTH - DevOps only matters to development (engineering) and operations team
- MYTH - DevOps will make the traditional IT roles redundant
- MYTH - DevOps doesn’t work for large, complex systems
- MYTH - DevOps requires teams’ physical proximity

Common DevOps Myths

- MYTH - Soft skills aren’t necessary
- MYTH - There’s no direct business value for adopting DevOps practices.
- MYTH - There’s no significant return on investment in applying DevOps principles to legacy apps.
- MYTH - There’s not enough time (or the right people) to implement DevOps.
- MYTH - DevOps doesn’t play nice with regulatory and compliance requirements.
- MYTH - There’s no reason to adopt DevOps because it can’t solve the kind of problems you have.
- MYTH - DevOps is just for startups or unicorns, not enterprise businesses.
- MYTH - It is all about CI/CD
- MYTH - DevOps means NoOps
- MYTH - Automation eliminates bottlenecks

Common DevOps Myths

- MYTH - You can have one-size-fits-all CD pipeline
- MYTH - Tools will solve your DevOps problems.
- MYTH - You should start with CI/CD.
- MYTH - DevOps transformation and cloud transformation can't happen at the same time.
- MYTH - The role of security is for vulnerability scanning.
- MYTH - Software Release Is the Same As In Amazon/Facebook/Google
- MYTH - Release All the Time
- MYTH - In order to deploy applications to the public cloud, devops toolchains need to operate there as well.
- MYTH - Using the cloud means not needing devops.
- MYTH - Devops leads to cloud security issues.

Three Principle-Based Frameworks for DevOps

01

The Three Ways as described in *The Phoenix Project* and *The DevOps Handbook*

02

Mature capabilities in technical and management practices found in high-performing DevOps teams, based on the research presented in *Accelerate*

03

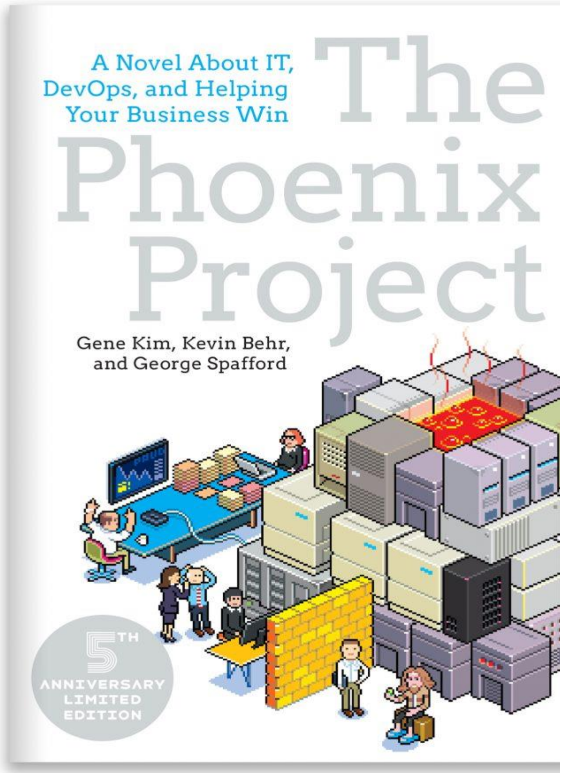
The CALMS (Culture, Automation, Lean, Measurement, Sharing) framework for assessing DevOps

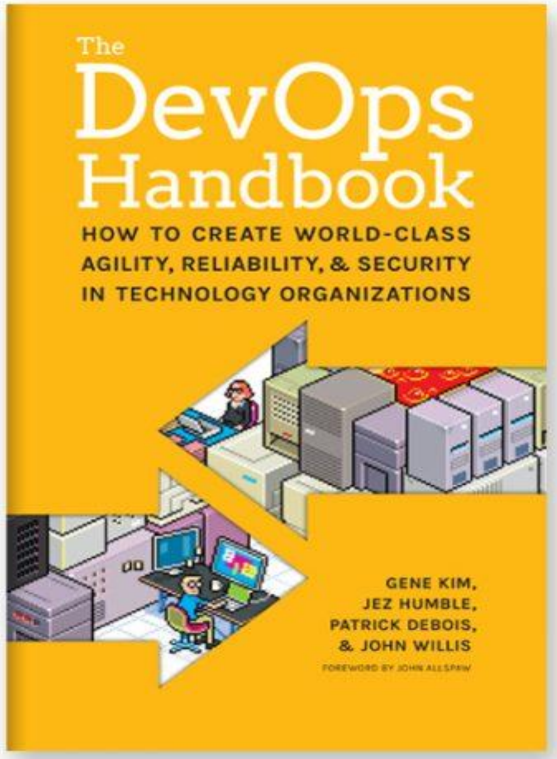
The Phoenix Project/DevOps Handbook's Three Ways

The First Way: Principles of Flow

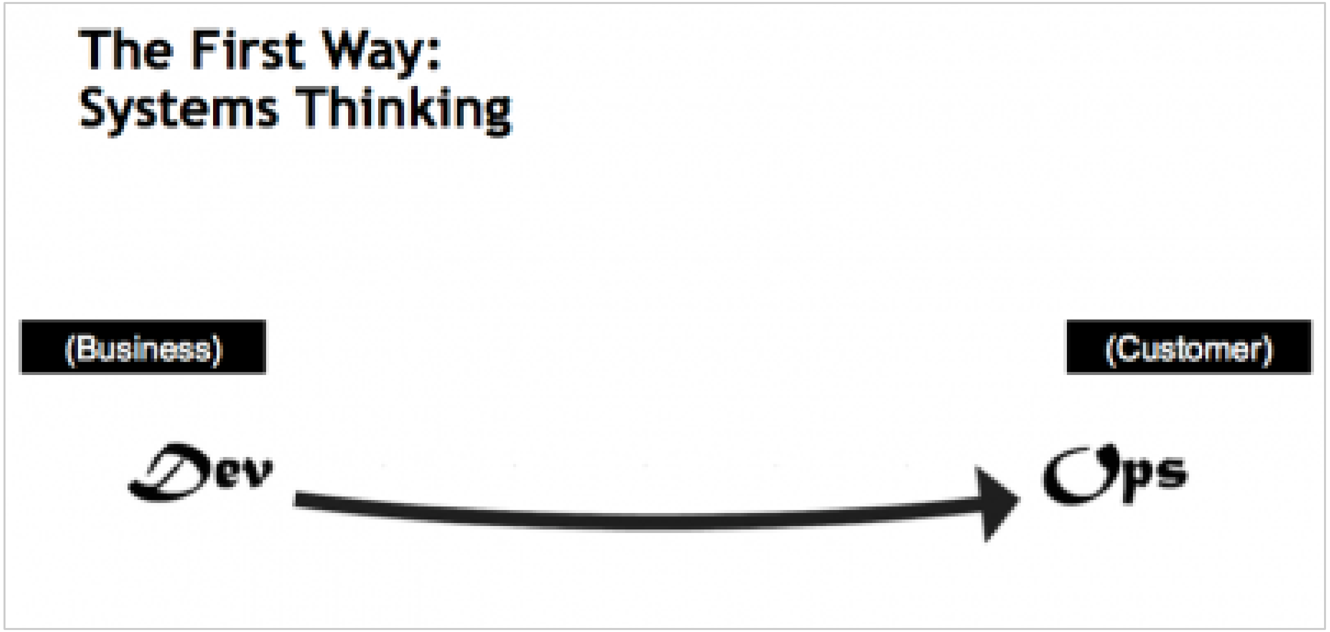
The Second Way: Principles of Feedback

The Third Way: Principles of Continuous Learning


A Novel About IT, DevOps, and Helping Your Business Win
The Phoenix Project
Gene Kim, Kevin Behr, and George Spafford
5TH ANNIVERSARY LIMITED EDITION

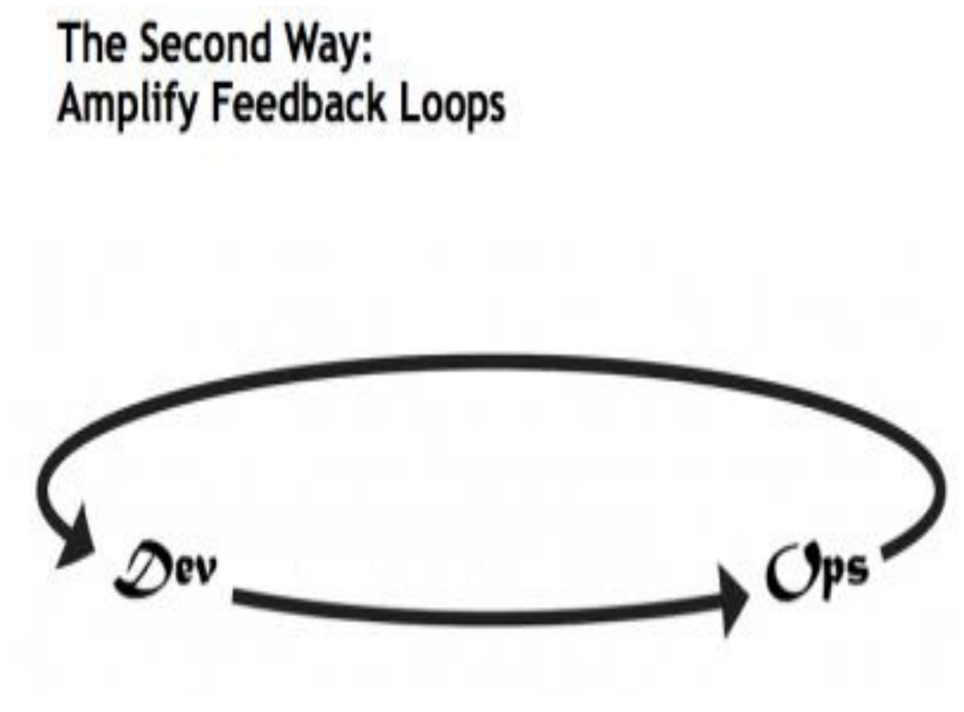

The DevOps Handbook
HOW TO CREATE WORLD-CLASS AGILITY, RELIABILITY, & SECURITY IN TECHNOLOGY ORGANIZATIONS
GENE KIM, JEZ HUMBLE, PATRICK DEBOIS, & JOHN WILLIS
FOREWORD BY JOHN ALLSPAW

Principles of Flow



- Making work "visible"
- Limiting work-in-progress (WIP)
- Reducing batch sizes
- Reducing hand-offs between teams
- Identifying and removing constraints and waste

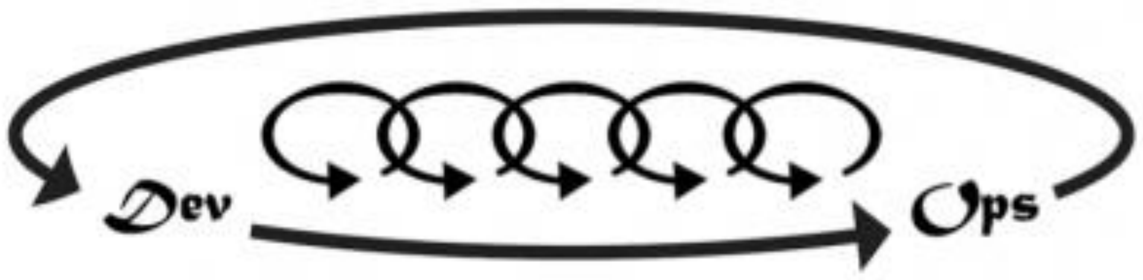
Principles of Feedback



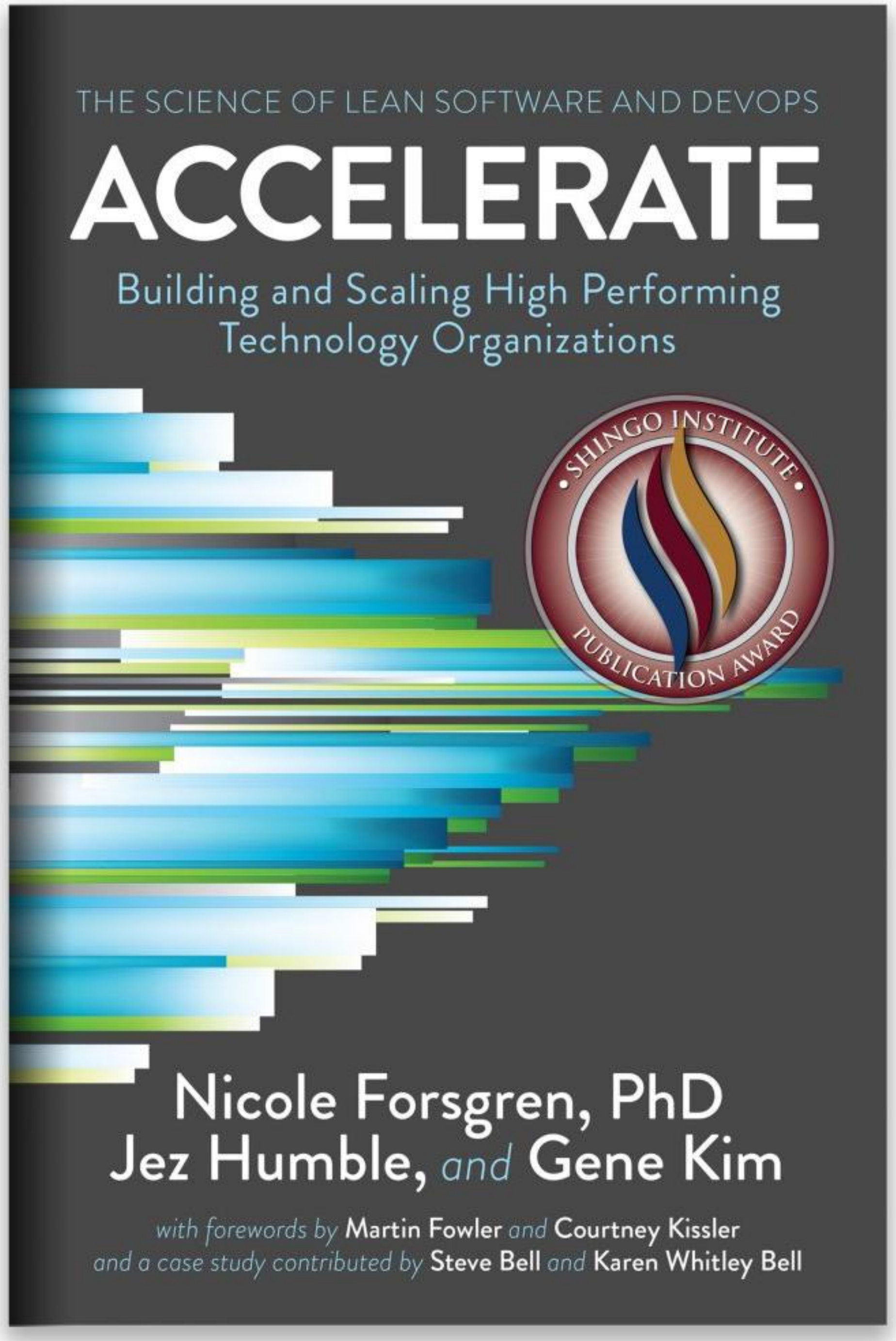
- **Swarming and solving problems to build new knowledge**
- **Pushing quality closer to source**
- **Optimizing for downstream work centers**

Principles of Continuous Learning

The Third Way:
Culture Of Continual Experimentation And Learning



- Enabling organizational learning and a safety culture
- Institutionalizing the improvement of daily work
- Transforming local discoveries to global improvements
- Injecting resilience patterns into daily work
- Leaders enforcing a learning culture


THE SCIENCE OF LEAN SOFTWARE AND DEVOPS
ACCELERATE
Building and Scaling High Performing Technology Organizations
SHINGO INSTITUTE PUBLICATION AWARD
Nicole Forsgren, PhD
Jez Humble, and Gene Kim
with forewords by Martin Fowler and Courtney Kissler
and a case study contributed by Steve Bell and Karen Whitley Bell

Accelerate's Technical and Management Practices of High-Performing DevOps Teams

Technical Practices

- Continuous Delivery
- Architecture
- Product and Process

Management practices

- Lean Management and Monitoring
- Cultural

Technical Practices

- **Continuous Delivery**
  - Version Control
  - Deployment automation
  - Continuous integration (CI)
  - Trunk-based development
  - Test automation
  - Test data management
  - Shift left on security (DevSecOps)
  - Continuous delivery (CD)

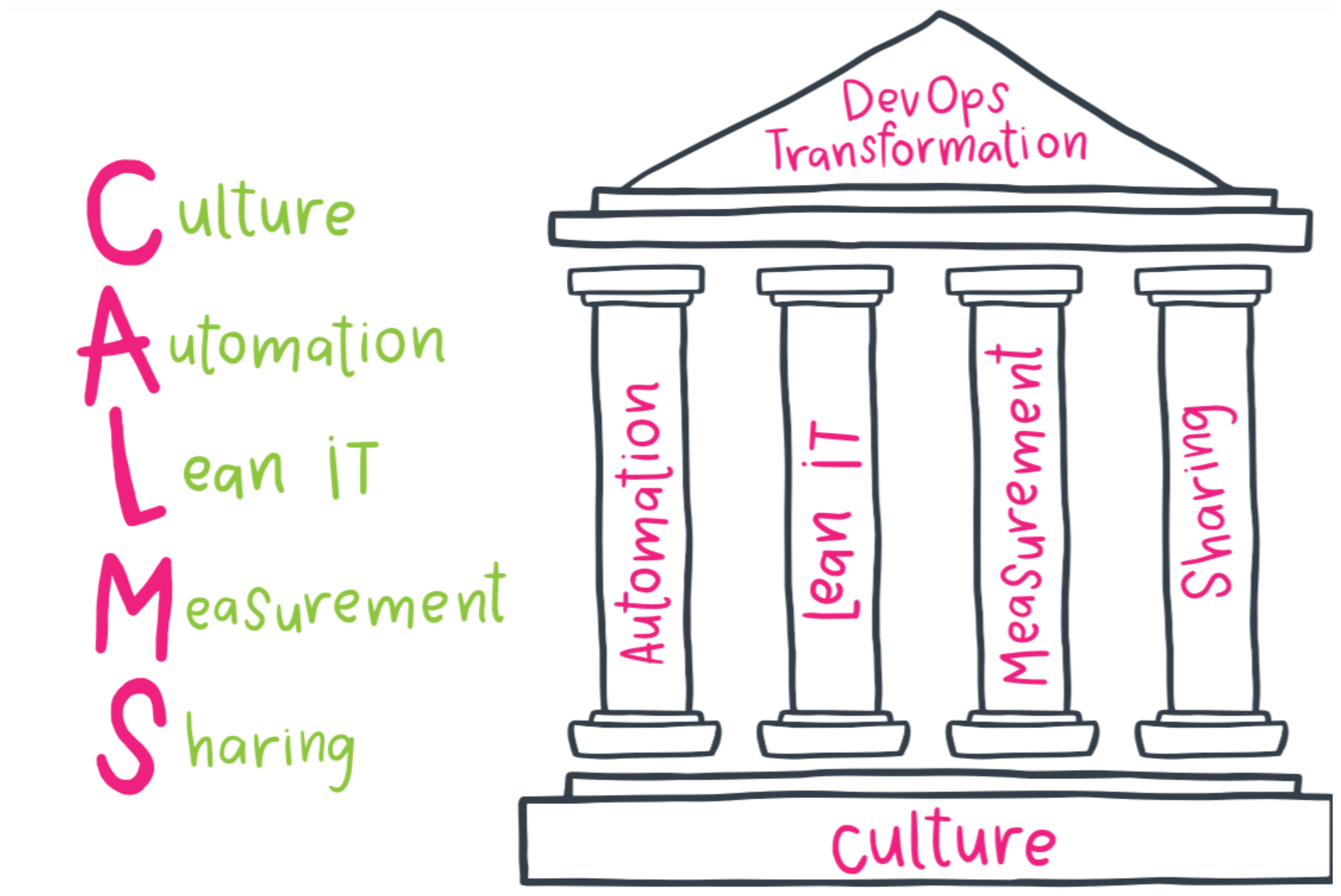
- **Architecture**
  - Loosely coupled architecture
  - Empowered teams

- **Product and Process**
  - Customer feedback
  - Value stream
  - Working in small batches
  - Team experimentation

Management practices

- **Lean Management and Monitoring**
  - Lightweight change approval processes
  - Monitoring
  - Work in Progress (WIP) limits
  - Visualizing work

- **Cultural**
  - Supporting learning
  - Collaboration among teams
  - Job satisfaction
  - Transformational leadership



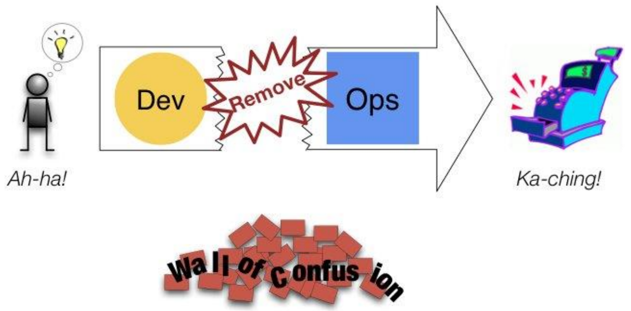
CALMS (Culture, Automation, Lean, Measurement, Sharing) Framework for DevOps

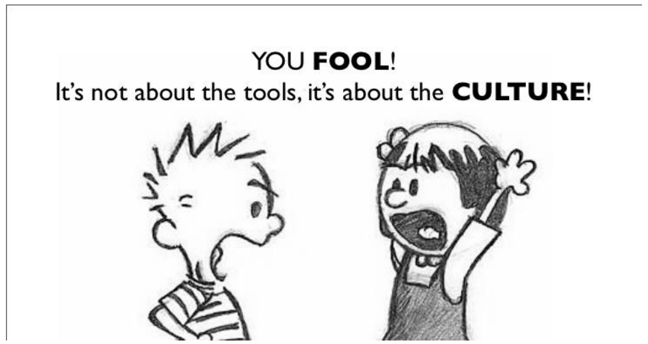
John Willis and Damon Edwards coined the acronym CAMS in 2010 which was later expanded to CALMS by Jez Humble.

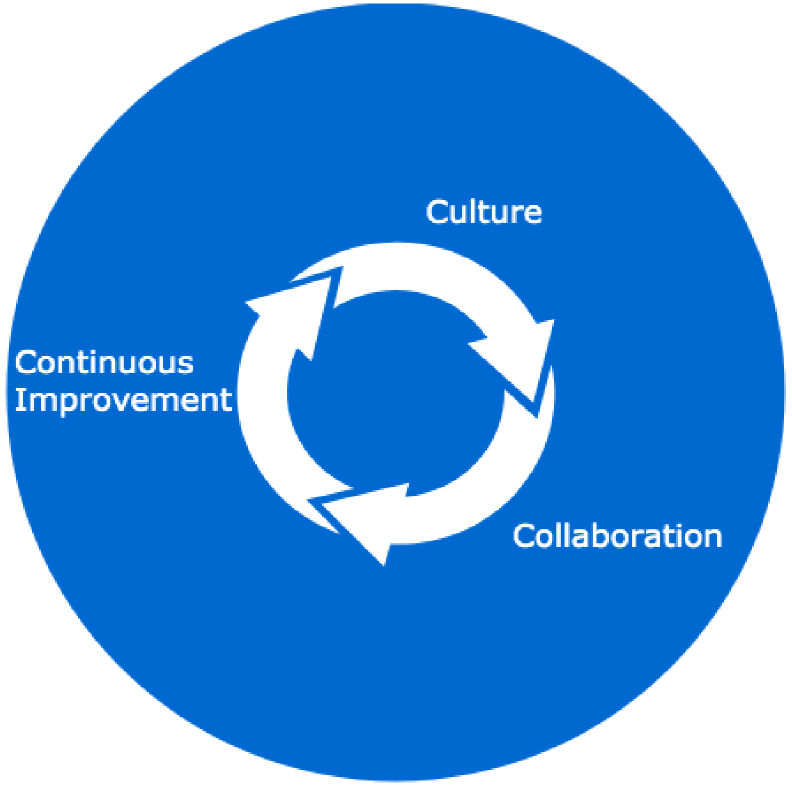
CALMS stands for:

- Culture
- Automation
- Lean
- Measurement
- Sharing

A Continuous Virtuous Cycle of 3Cs







A Legacy Mindset

- “It worked on my laptop!”
- “Keep your changes away from my code.”
- “I don’t know how a change will impact production; we don’t touch production.”
- “That server resource is unique/one-off/snowflake/bespoke/hand maintained by the ops team” = pet infrastructure.
- “We will have to live with a single point of failure” = pet.
- “Unconfigurable features, unrevertable changes, incompatible APIs.”
- “Backups are a good enough recovery strategy.”

To paraphrase the journey to DevOps: we all must become DevOps, working to continuously improve

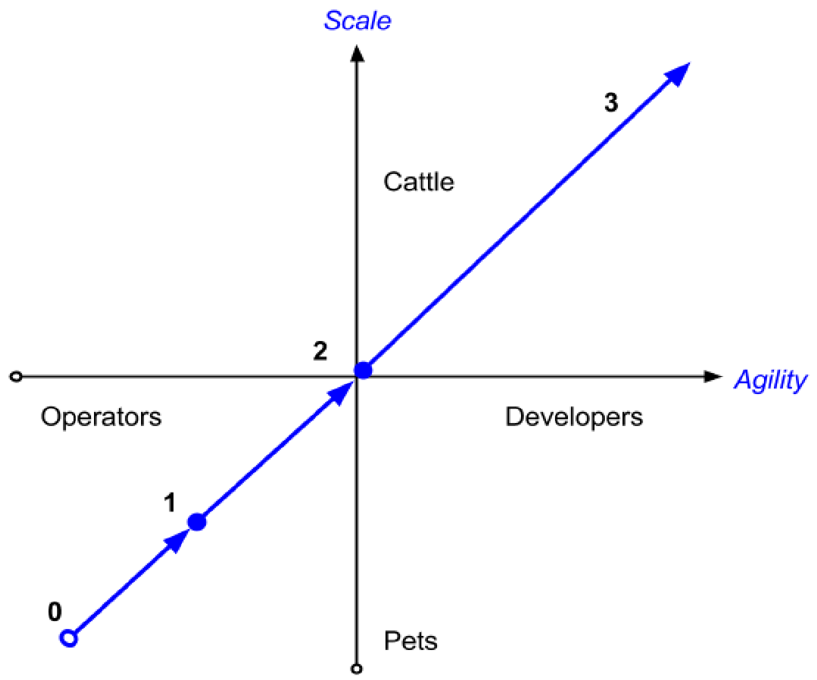
Agility = DevOps
Scalability = Removing Pets

- **Cultural change** to reduce dev+test+ops silos
- **Automation** increasing value delivery, by accomplishing:
  - Agility by democratizing expertise and distributing work
    - Results: continuous integration, delivery, and deployment
  - Scalability by eliminating single points of failure
    - Results: cattle infrastructure and operations
- **Feedback** measuring closed loop value4
  - Results: monitors + logs + metrics for Key Performance Indicators

Proverb: *What gets measure gets managed.*

DevOps Journey

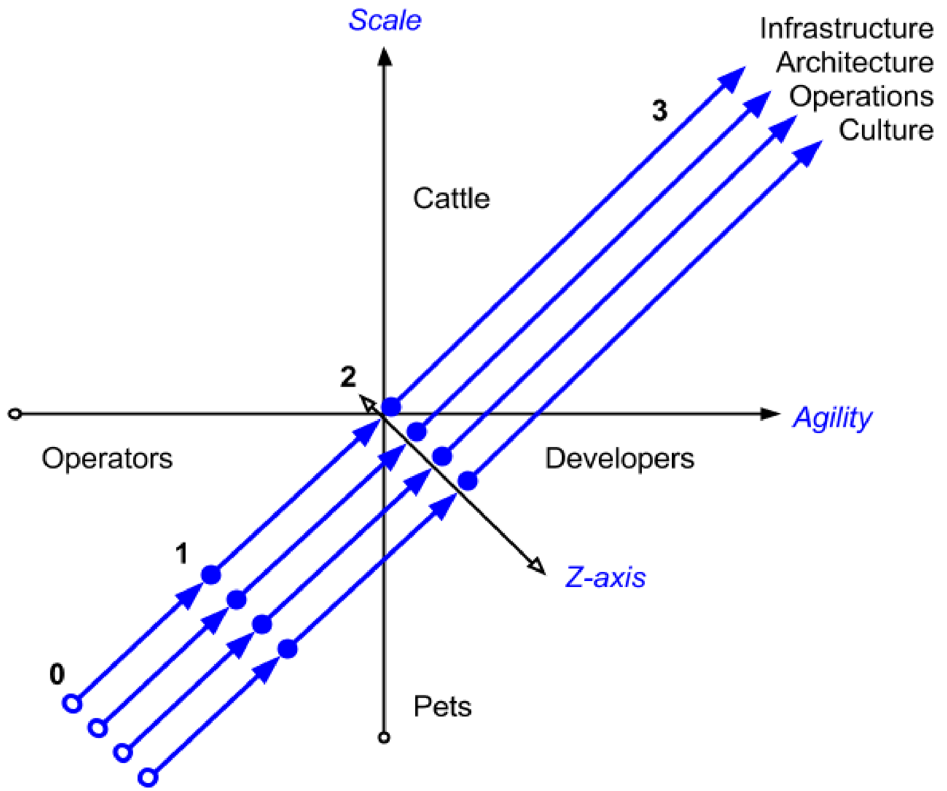
The combination of infrastructure and operational automation implied multiple aspects of the journey



Stage	Outcomes
0	Hand maintained monoliths, in a pet data center, with backups for revision control
1	Programmatic change controls and operations
2	Deploy new workloads with configuration management
3	Cattle everywhere: build test driven infra-artifacts, hybrid cloud deployments, KPI driven operations

DevOps Maturity

Four parallels, each representing infrastructure, architecture, operations, and culture when aligned and streamlined, the overall organization could achieve advanced stage outcomes, yielding the below graph



Stage	Infrastructure	Architecture	Operations	Culture
0	Single server, single datacenter	Monolithic	Hands-on	Silos
1	Synthesized	Distributed	Repeatable	Governable
2	Ephemeral	Scale out	Delegable deploy+ops	Testable with metrics
3	Hybrid clouds	Global + active	KPI driven lifecycle	Data-driven experiments

Miniscule Understanding of Pets Vs Cattle

DevOps is an extension of _Agile_ infrastructure in which its process is _Iterative_ or repeated in cycles.

Ideology

Category	Pets	Cattle
Organization:	Single Point of Failure (SPoF)	Fleet (no SPoF)
Naming:	Members of a theme	Naming Conventions and Numbers
Remediation:	MTTF	Replace to maintain SLA
Infrastructure:	Scale up	Scale out
Architecture:	Monolith	Distributed, Microservice

Technology

Category	Pets	Cattle
Naming:	earth.corp	webtier-01.ahv01.prod.corp
Uptime Goal:	Years	Seconds
Failure:	MTTR of minutes + hours	No SPoF!
Network:	Hardware	Software Defined
Storage:	SAN	Distributed File System
Datacenter:	Single AWS Region, Cluster, or DC	Multiple Enterprise/Hybrid Clouds

Culture

Category	Pets	Cattle
People:	Heroic admin at HQ2	Global Operations team
Operations:	Hands on: manual ops, change controls	Hands off: monitors, KPIs, ChatOps
Values:	DevOps team	DevOps mindset distributed across entire organization

How to Do DevOps ?

- http://dev2ops.org/2013/12/how-to-initiate-a-devops-transformation-video/
  1. **Build the "Why?"**
  2. **Building Organizational Alignment**
     - **Teach the basic concepts**
     - **Getting everyone on the same page** through
       - Value Stream Mapping
       - Timeline Analysis
       - Waste Analysis
     - **Developing metrics chains**
     - **Identify projects/experiments against baseline**
     - **Repeat steps 2-4**
  3. **Continuous Improvement Loops**

  HPE on DOES2015 : https://www.youtube.com/watch?v=q9nNqqie_sM

DevOps Best Practices

- **<u>Infrastructure as Code</u>**
- **<u>Continuous Integration, Delivery, and Deployment</u>**
- **Immutable Infrastructure**
- **Microservices and Containers**
- **Instrumentation**

7 Secrets for DevOps Success by Gene Kim

1. **Change often begins in operations**
2. **DevOps transformations start small—but not too small**
3. **Business-savvy technologists take the lead**
4. **DevOps change agents take risks**
5. **DevOps demands a culture of trust**
6. **DevOps expansion requires leaders to evolve**
7. **CIOs are key enablers of DevOps**

Source: https://www.hpe.com/us/en/insights/articles/gene-kims-7-secrets-of-devops-success-1702.html

7 Tips for DevOps Success by Gene Kim

Tip #1: Deploy smaller changes more frequently

- Decouple feature releases from code deployments.
- Deploy features in a disabled state, using feature flags.
- Require all developers check code into trunk daily (at least).
- Practice deploying smaller changes, which dramatically reduces risk and improves **MTTR**.

Tip #2: Fearlessly enable testing in production

- When we can continuously deploy disabled code into production, we can test under production-like loads long before the feature release.
- When we build our code and environments with resilient design patterns, we can rehearse and simulate large-scale failures so we're ready (à la **Chaos Monkey**)

Tip #3: Build a one-step environment-creation process

- Make environments available early in the development process.
- Make sure dev builds the code and environment at the same time.
- Create a common dev, QA, and production environment-creation process.

7 Tips for DevOps Success by Gene Kim

Tip #4: Synchronize the schedules of dev and ops

- it’s important to schedule updates in the middle of the day, when both groups are working. Not nights, weekends, and holidays when the devs are out of the office and ops folks are bearing the entire burden.

Tip #5: Change the way sprints work

- At the end of each sprint, we must have working and shippable code, demonstrated in an environment that resembles production.

Tip #6: Version control is more important for ops than dev

- That’s because there are more configuration settings in the environment than there are in the code, and “most failures have to do with environment errors, not code errors.”

Tip #7: Allocate 20% of cycles to technical debt reduction

- It’s the price you pay to get great dev and ops.

Challenges in Adopting DevOps

- Lack of a Standard Definition for DevOps
- Dearth of Vision
- Shortage of Tool Knowledge
- Choice of Tools
- Lack of Tool Integration
- Cultural Challenges
- Isolated Teams
- Risk Analysis
- Scarcity of SMEs

Who Should Not Do DevOps ?

- **Your business doesn't need regular releases**
- **Your business is satisfied with the current state of software**
- **You operate in a highly regulated industry**
- **Your business has lots of M&A activity on the horizon**
- **Legacy processes or architectures still rule**

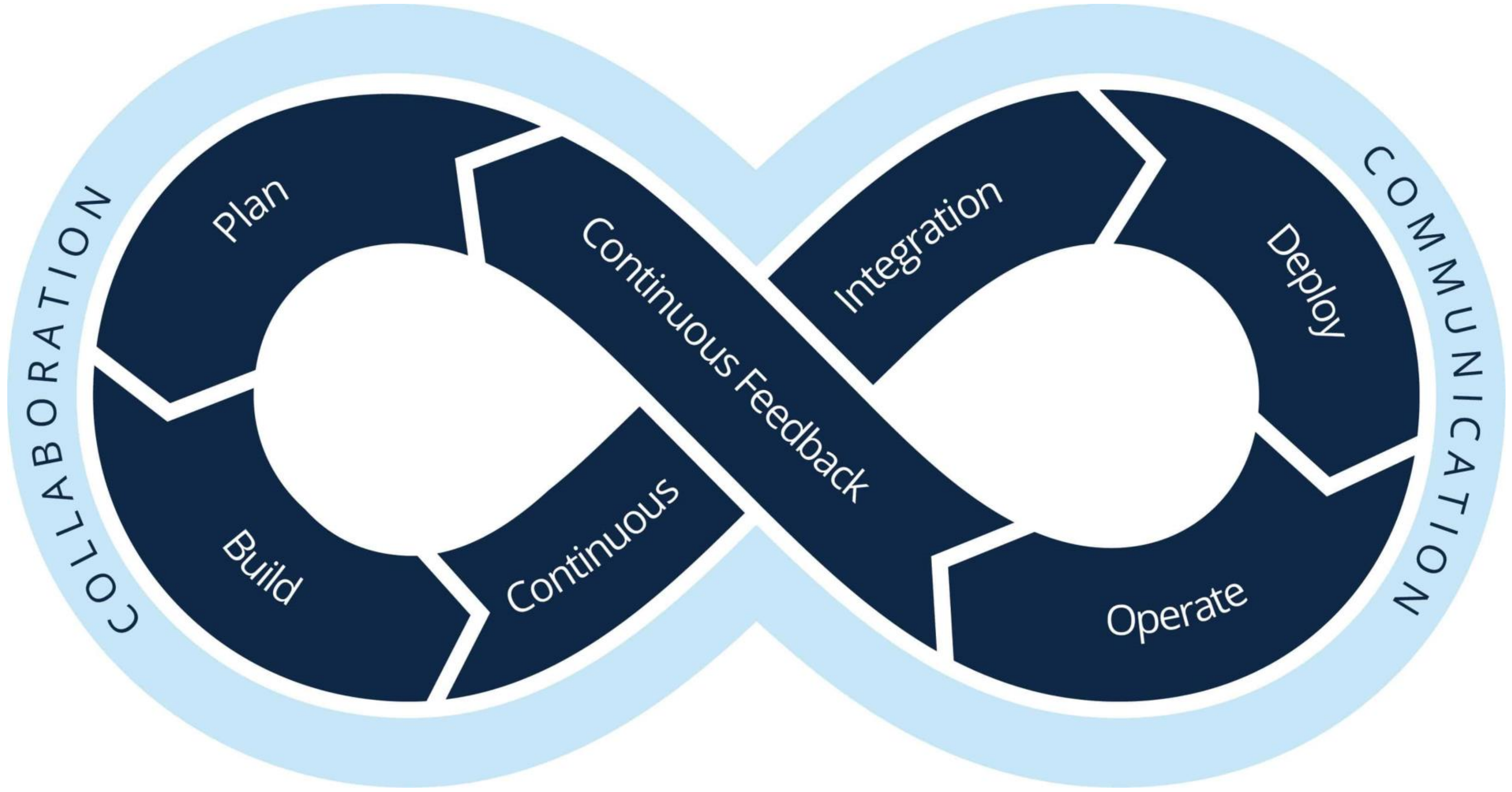
https://medium.com/faun/the-must-know-checklist-for-devops-system-reliability-engineers-f74c1cbf259d

https://www.devops-research.com/research.html

Summary

DevOps resolves the Core Chronic Conflict between Innovation & Stability

Q & A


COLLABORATION
Plan
Build
Continuous
Continuous Feedback
Integration
Deploy
Operate
COMMUNICATION