DevOps Culture and Principles

Diego Pacheco

About Me

- ❑ Cat's Father
- ❑ Principal Software Architect
- ❑ Agile Coach
- ❑ SOA/Microservices Expert
- ❑ DevOps Practitioner
- ❑ Speaker
- ❑ Author

diegopacheco

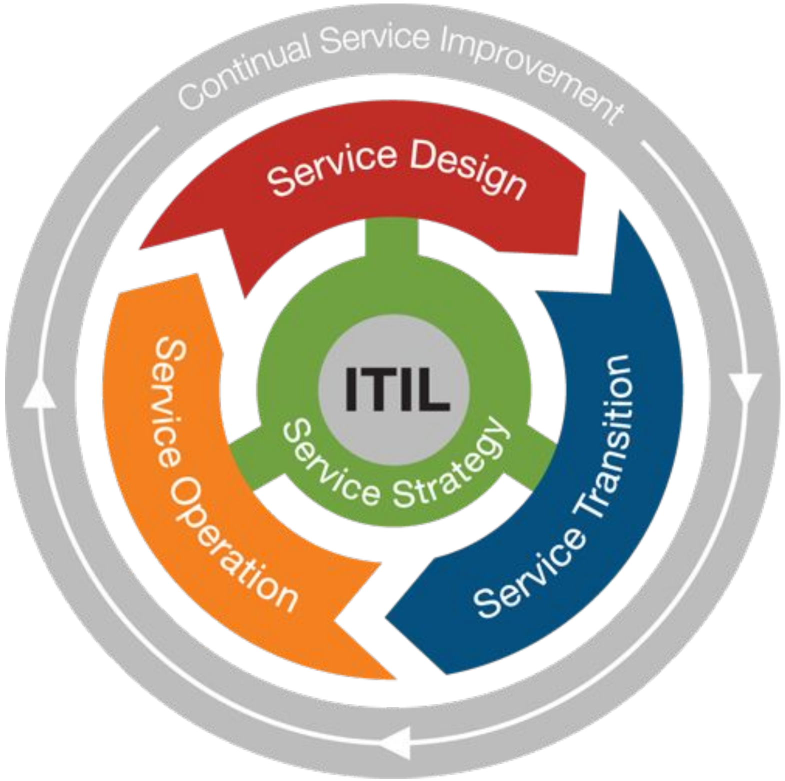
@diego_pacheco

http://diego-pacheco.blogspot.com.br/

Diego Pacheco
Building Applications with Scala
Write modern, scalable, and reactive applications with the power of Scala
Packt

Diego Pacheco
Building Effective Microservices
Explore microservices and their implementation hands-on
Packt

ITIL


Continual Service Improvement
Service Design
Service Transition
Service Operation
ITIL
Service Strategy

DevOps Founders and Big Names

Patrick Debois

Jez Humble

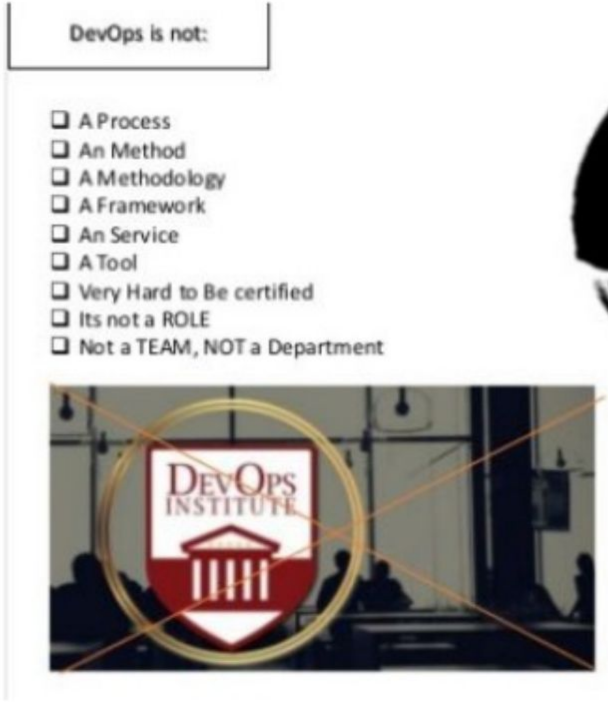
Gene Kim

John Willis

[Before] DevOps Movement

- ❑ COST Oriented
- ❑ Short Tern Focus
- ❑ Only focused on MGMT
- ❑ CHEAP Contractors
- ❑ Wrong Solutions (Not Specialists)
- ❑ High Process / Control
- ❑ Lack of Innovation and LED time

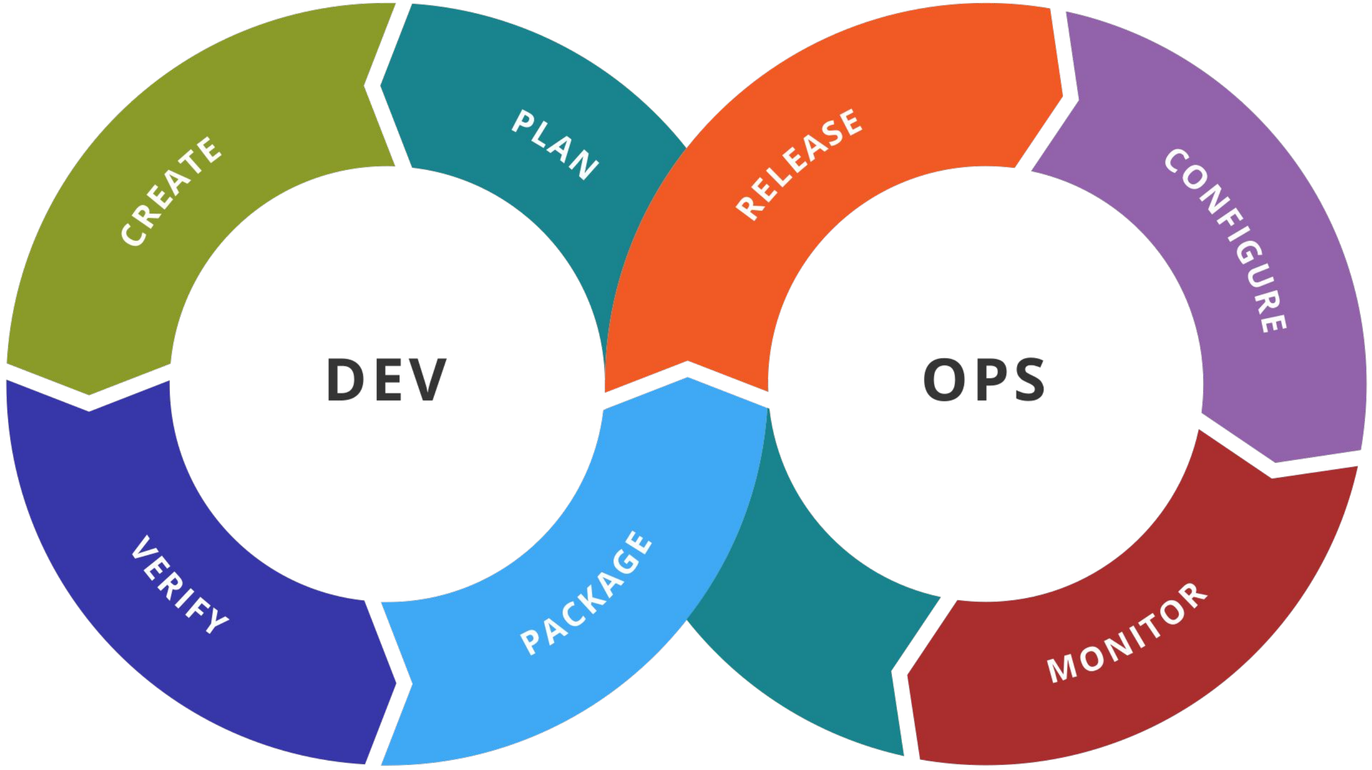
200[7..8] DevOps


DevOps is not:

- A Process
- An Method
- A Methodology
- A Framework
- An Service
- A Tool
- Very Hard to Be certified
- Its not a ROLE
- Not a TEAM, NOT a Department

DevOps
Institute

More than Just “Dev” working with “Ops”



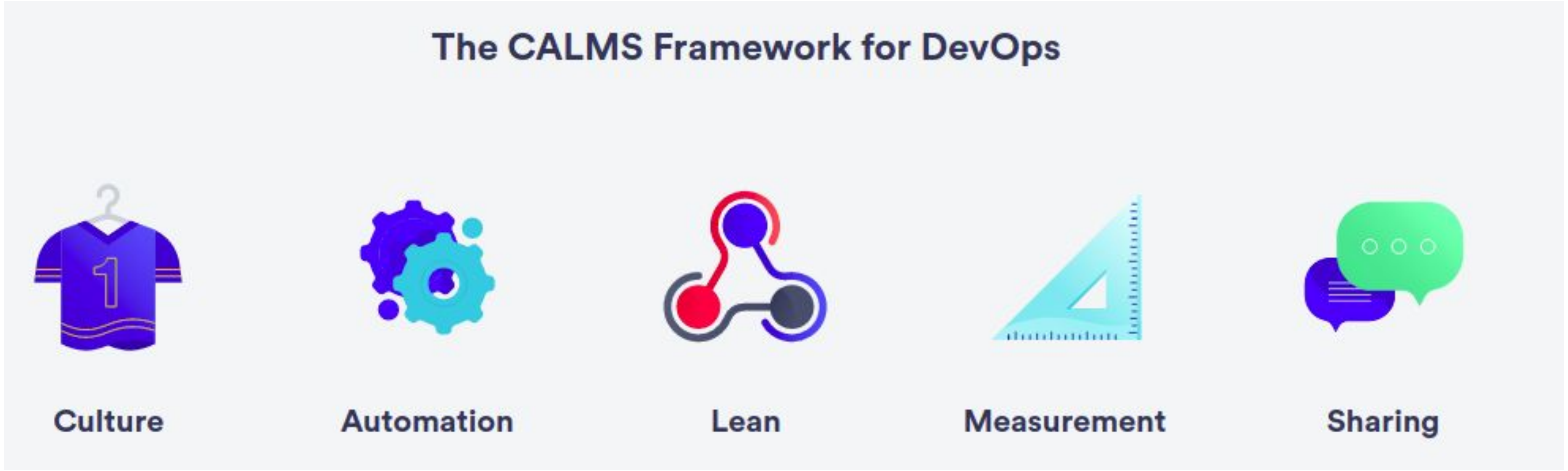
My Definition of DevOps

DevOps são formas de se pensar, escrever e operar software para atingir alto desempenho e excelência de TI para trazer retorno para o negócio.

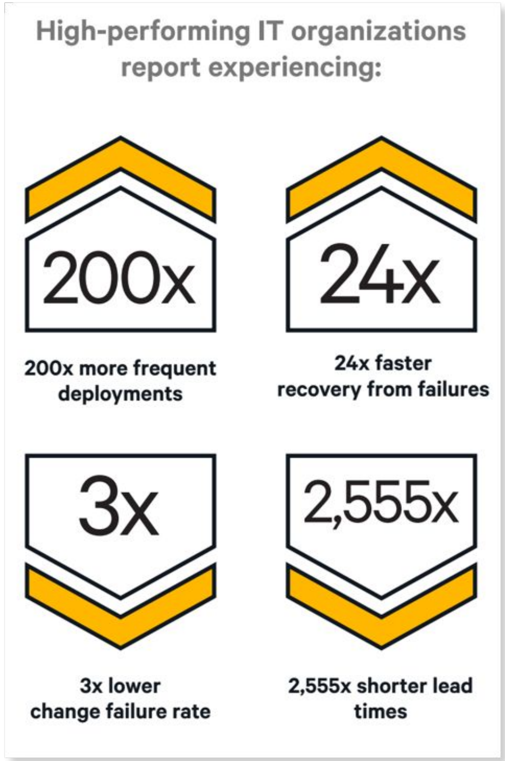
DevOps Cultural Shift

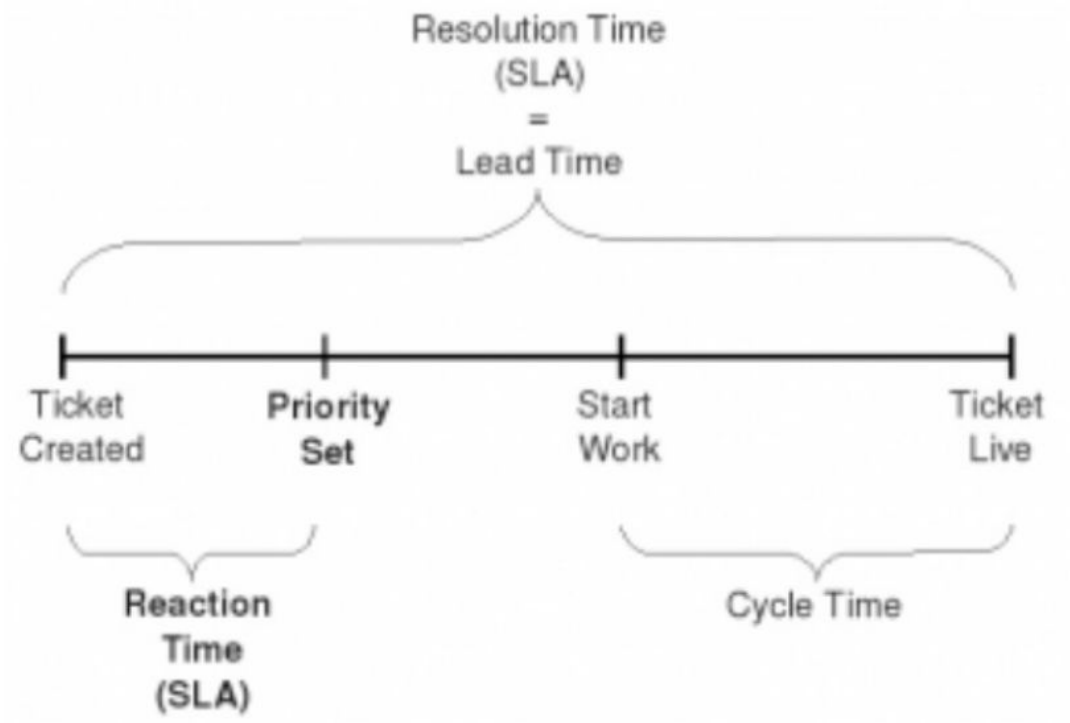
- ❑ Short Tern -> Long Tern
- ❑ High LEAD TIME -> Low LEAD TIME
- ❑ Long Downtimes -> Low rollback time / Low recovery time
- ❑ Lots of Incidents -> Few Incidents
- ❑ Poor User Experience -> Awesome User Experience
- ❑ Coupling -> Independence
- ❑ Side Effects -> Performance Degradation
- ❑ Long Innovation cycles -> Short Innovation Cycles
- ❑ COST Oriented -> Value Oriented
- ❑ Manual -> Automated

CALMS

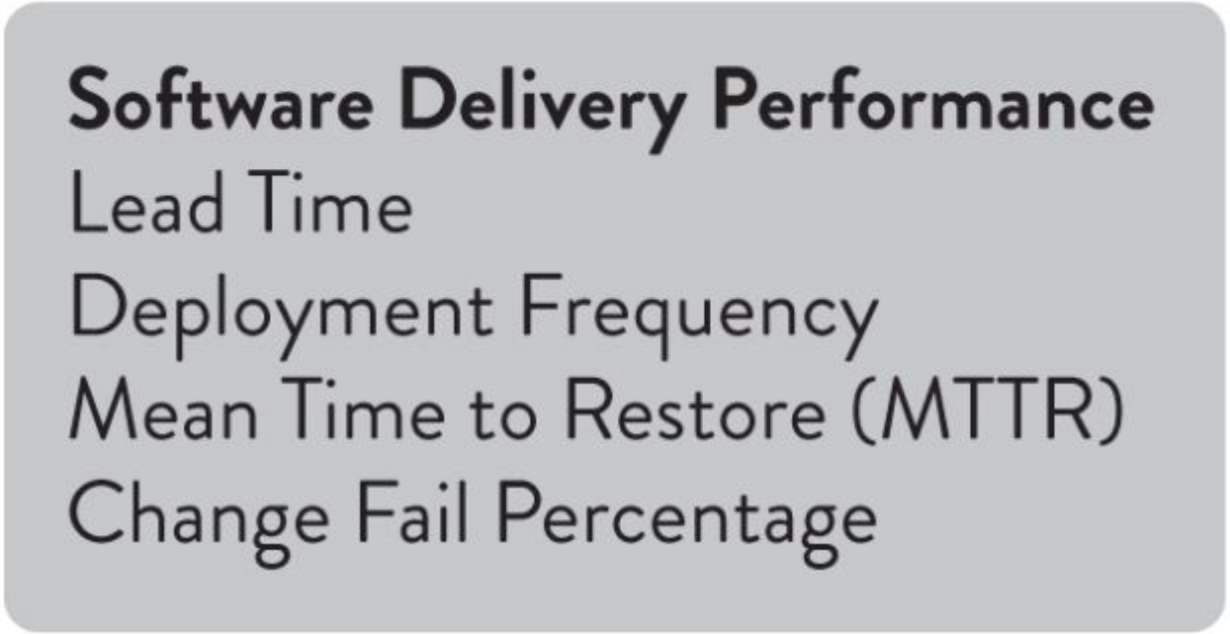

The CALMS Framework for DevOps
Culture
Automation
Lean
Measurement
Sharing

DevOps is about Time to Market



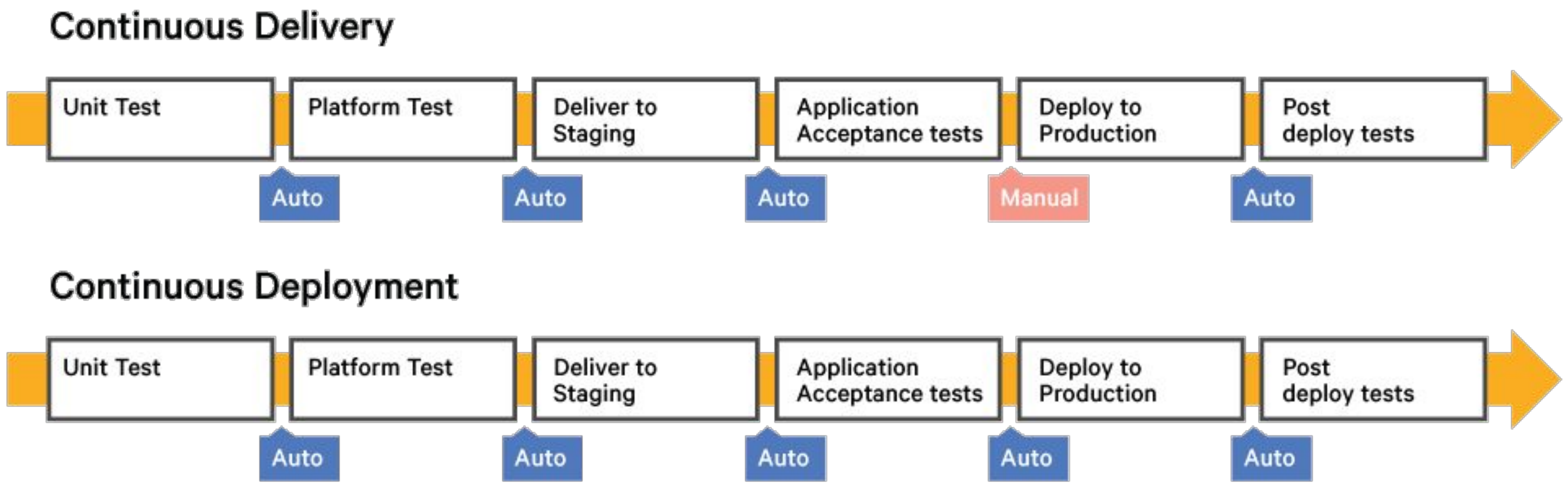


How to Measure DevOps Adoption?



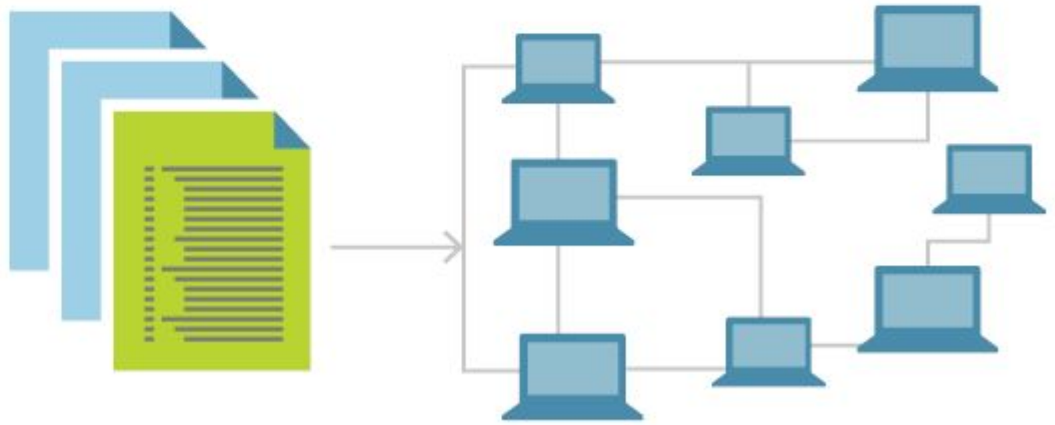
Software Delivery Performance
Lead Time
Deployment Frequency
Mean Time to Restore (MTTR)
Change Fail Percentage

CI/CD



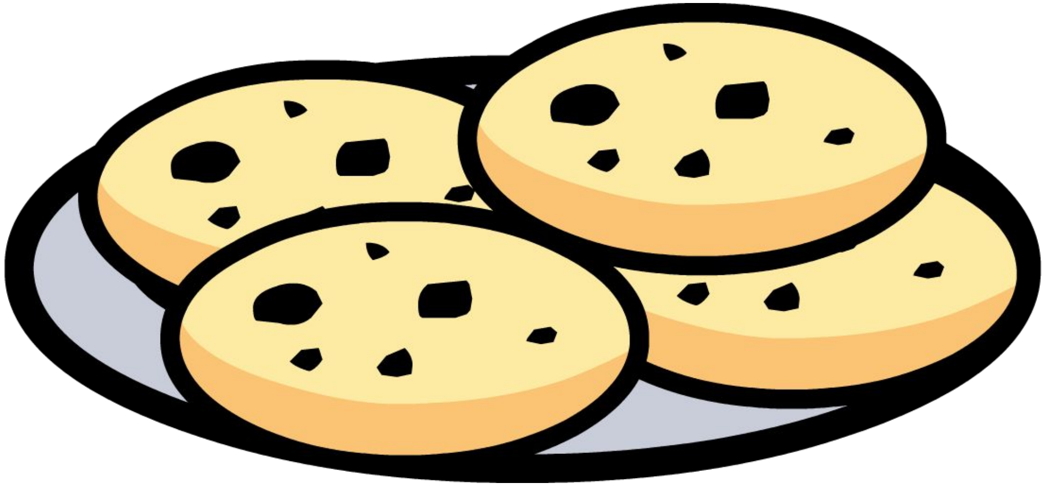
Continuous Delivery
Unit Test
Platform Test
Deliver to Staging
Application Acceptance tests
Deploy to Production
Post deploy tests
Auto
Auto
Auto
Manual
Auto
Continuous Deployment
Unit Test
Platform Test
Deliver to Staging
Application Acceptance tests
Deploy to Production
Post deploy tests
Auto
Auto
Auto
Auto
Auto

Infrastructure as Code



- ❑ Software Defined
  - ❑ Storage
  - ❑ Network
  - ❑ Infrastructure
- ❑ Big Role by Cloud Providers
- ❑ Benefits:
  - ❑ Automation
  - ❑ Create-Recreate
  - ❑ Recovery Benefits
  - ❑ Isolation (N-Envs)
  - ❑ Versioning

Immutable Infrastructure



- Versioning for each component
- Make sure you:
  - Known each change
  - Keep track
  - Changes make a new version
  - Enable “Testing”
  - “Easy” Rollback
- Every new FIX is a new version
- Several Companies disable SSH
- You have mutable state by software but not by Ops or Humans.
- Done via IMAGES for OS
  - Every cloud has they format
  - Packer (later)
- Both for OS and Whole Infrastructure(Cloud based or SD*)

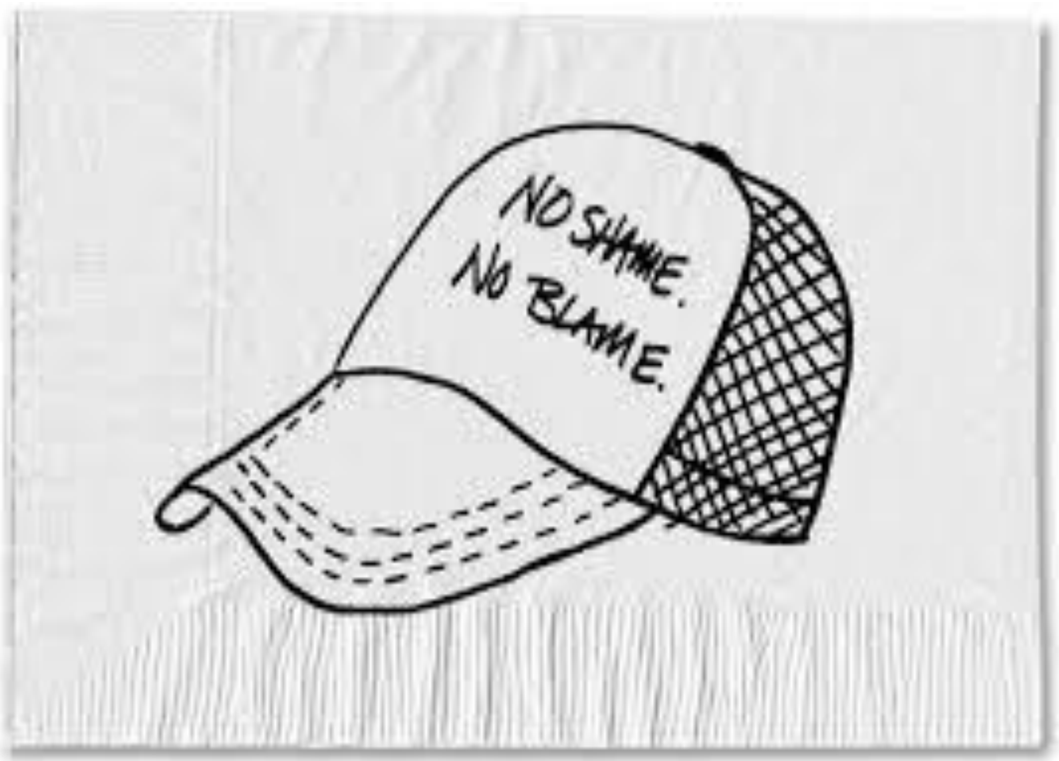
Baking Hell

- Slow if poor hardware provisioned or
- Too much software of BASE AMI
  - Bottleneck
  - Other people break you
- Onions Model - like Netflix
  - Stable
  - But Slow - Super Slow
- Roles Sharing Model
  - Bake 1 Image per component
  - Share roles with common sw
- Provisioning requires:
  - Stability Mindset
  - Testing
  - Tracking

Self Service Solutions

- ❏ How cloud Works
- ❏ Generic Solutions
- ❏ Favor CONFIG over CODE
- ❏ Stop making Developers/Ops to Code in order to USE - ALWAYS when possible.
- ❏ Don't do DevOps work for Developers - Develop tools and solutions to improve they work
- ❏ Dev vs DevOps Ownership
- ❏ Only way to SCALE
- ❏ Documentation also scales and avoid meetings and waste of time.

Blameless Postmortems

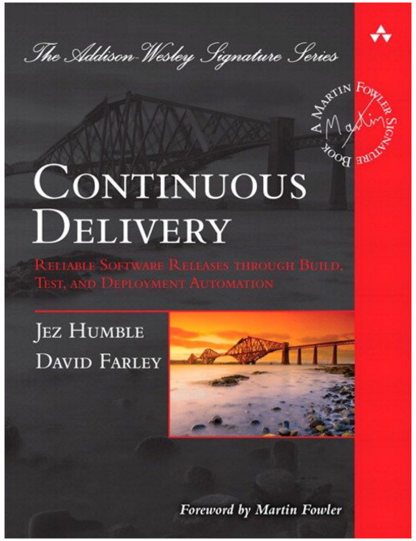


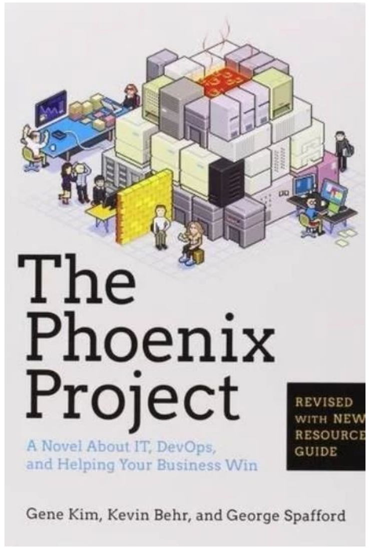
- ❑ No Trainee Jokes or Fault.
- ❑ Postmortems
  - ❑ Shit Happens
  - ❑ Great Practice
  - ❑ Learn from others
- ❑ Most important thing:
  - ❑ NO finger pointing
  - ❑ No Blame
  - ❑ No Shame
- ❑ Always make sure people learn and improve software so the world ends up being a better place.

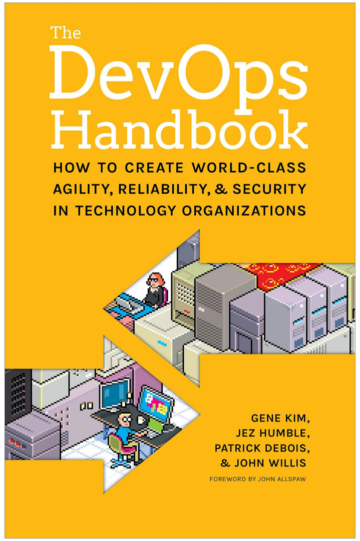
Stability Mindset

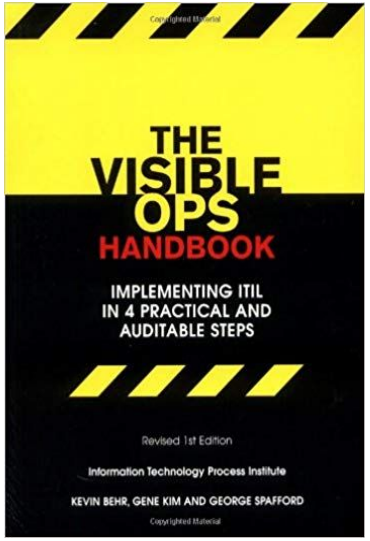
- First and more important:
  - CARE about it.
- Don't CHANGE and GO.
- Check lists are great practice.
- Don't trust your tests
- Lack of tests also don't make you safe
- Different Rollback / Bugs Cost:
  - Microservices
  - Database
- Pipelines don't avoid Outages - Just because you can rollback it does not mean you should be careless.

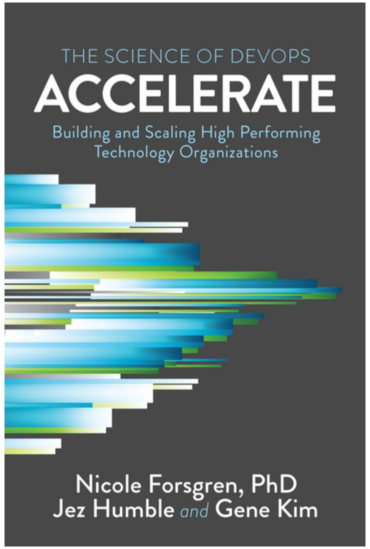
Recommended Books


The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fowler


The Phoenix Project
A Novel About IT, DevOps, and Helping Your Business Win
Revised with New Resource Guide
Gene Kim, Kevin Behr, and George Spafford


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


The Visible Ops Handbook
Implementing ITIL in 4 Practical and Auditable Steps
Revised 1st Edition
Information Technology Process Institute
Kevin Behr, Gene Kim and George Spafford



DevOps Culture and Principles

Diego Pacheco