DevOps

Arun Gupta, @arungupta
Red Hat

Delivering Software: Reality vs Goal

BUDGET	TIME	VALUE
45% OVER	7% OVER	56% LESS

Source: Delivering large-scale IT projects on time, on budget, and on value
McKinsey & Company, October 2012

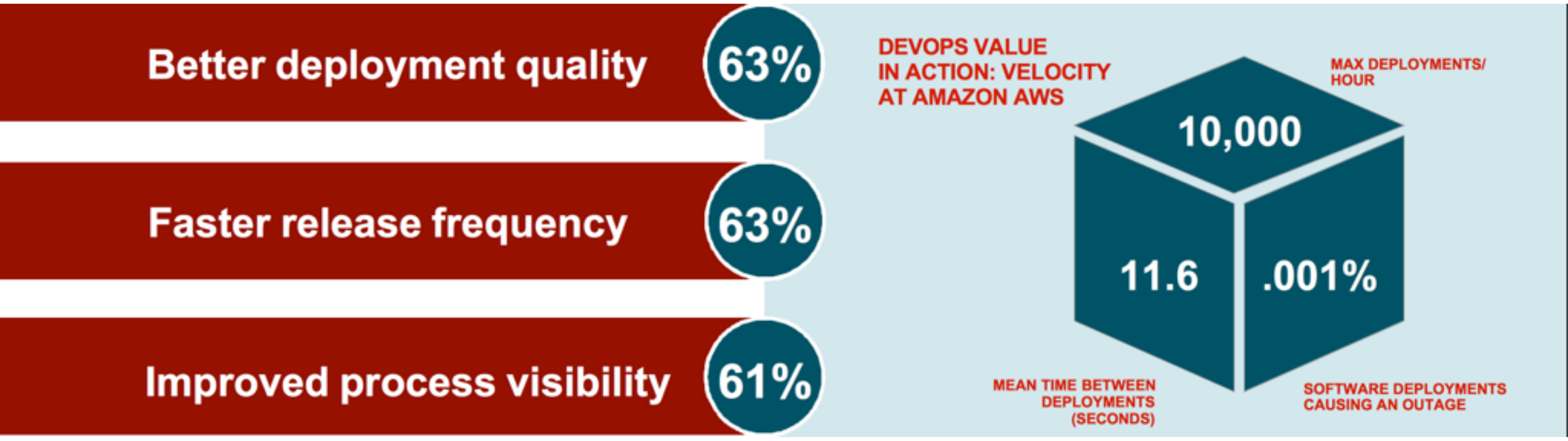
Is DevOps for you?

Ship code 30x faster
and complete those deployments 8,000 times faster than their peers.

Have 50% fewer failures
and restore service 12 times faster than their peers.

https://puppetlabs.com/wp-content/uploads/2013/03/2013-state-of-devops-report.pdf

Organizations implementing DevOps



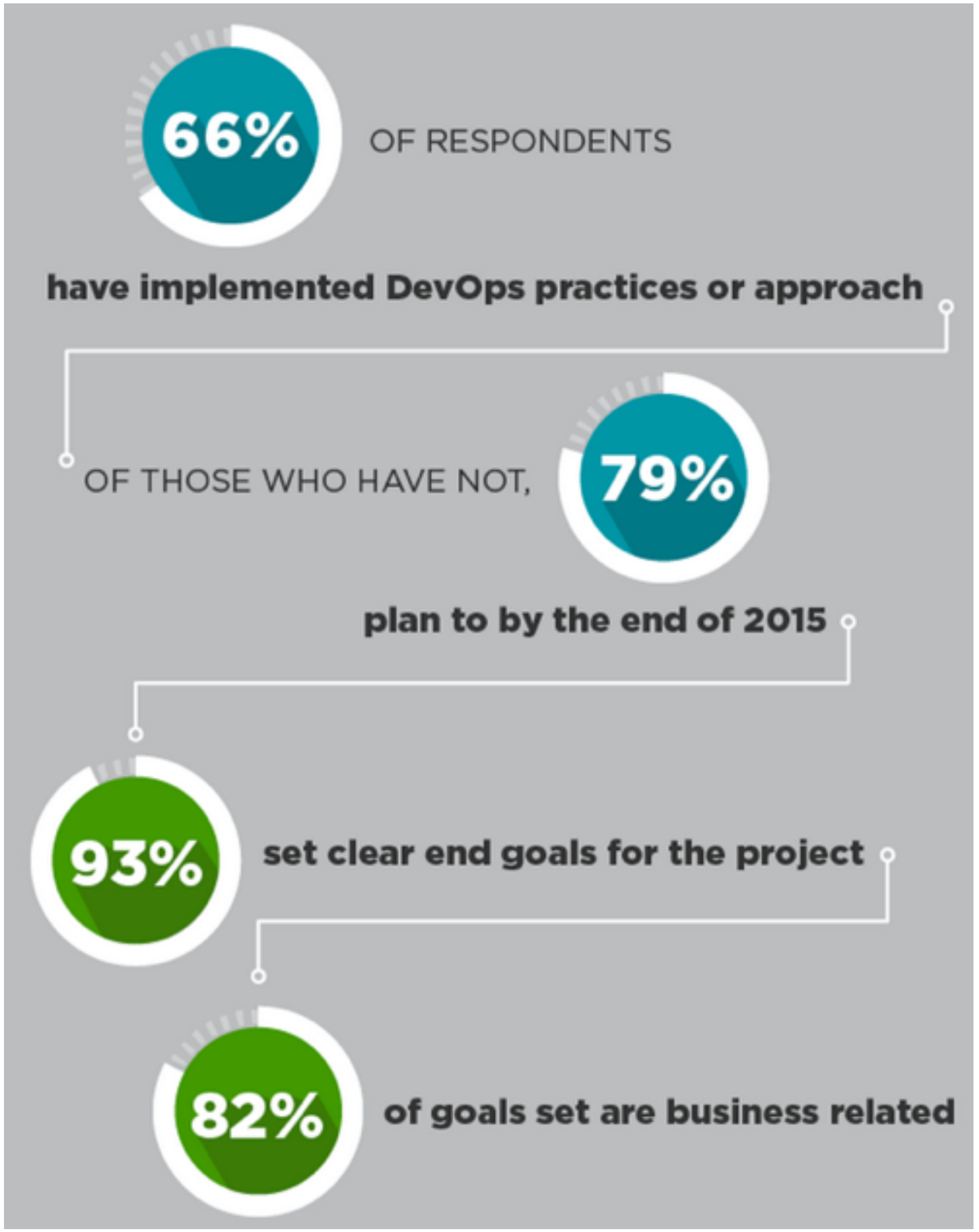
https://puppetlabs.com/wp-content/uploads/2013/03/2013-state-of-devops-report.pdf

Organizations implementing DevOps

Numbers missing?

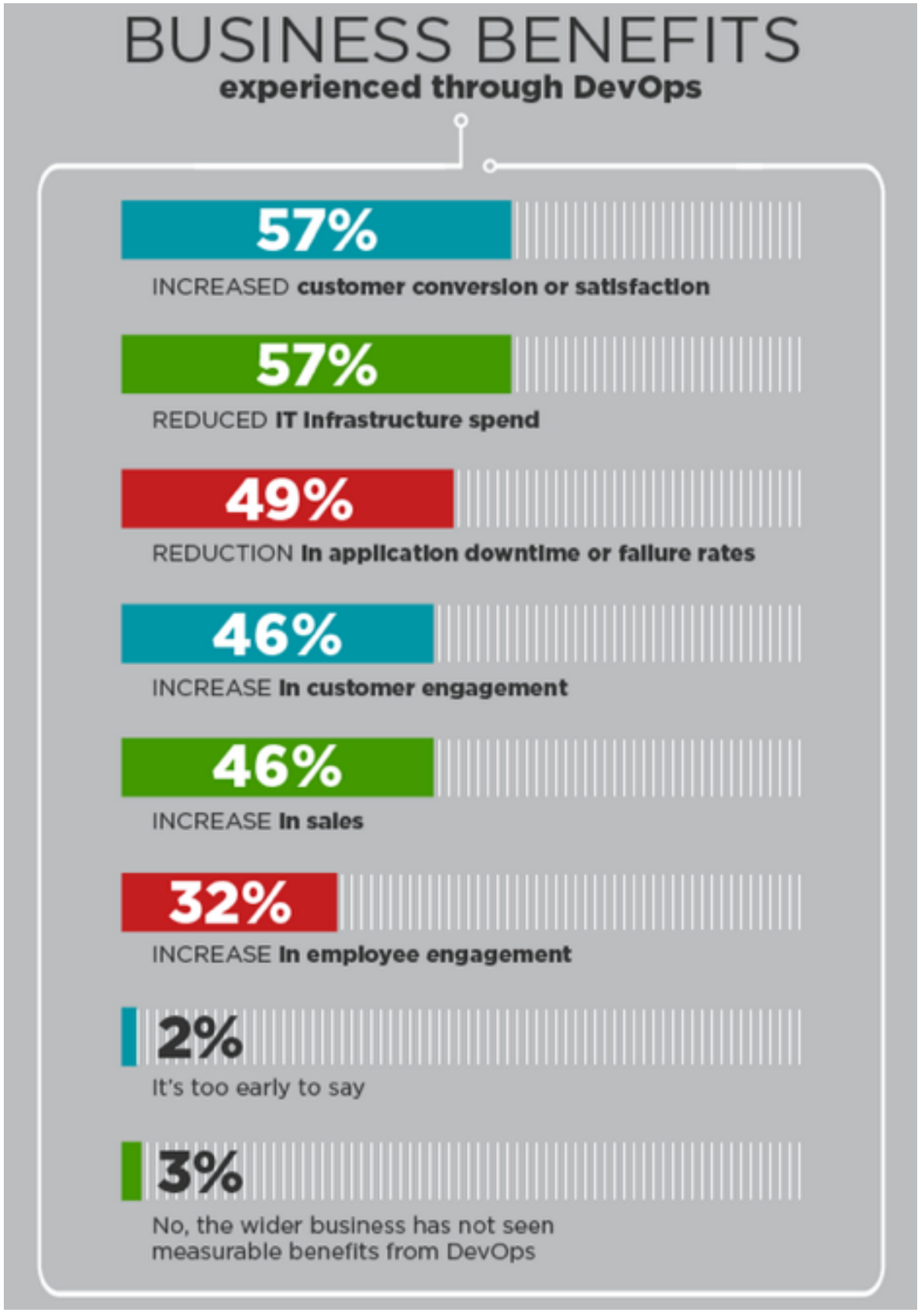
http://puppetlabs.com/sites/default/files/2014-state-of-devops-report.pdf

Oct 2014



http://www.rackspace.com/blog/a-devops-snapshot-rackspace-survey-shows-businesses-benefits-of-devops-infographic/

Oct 2014

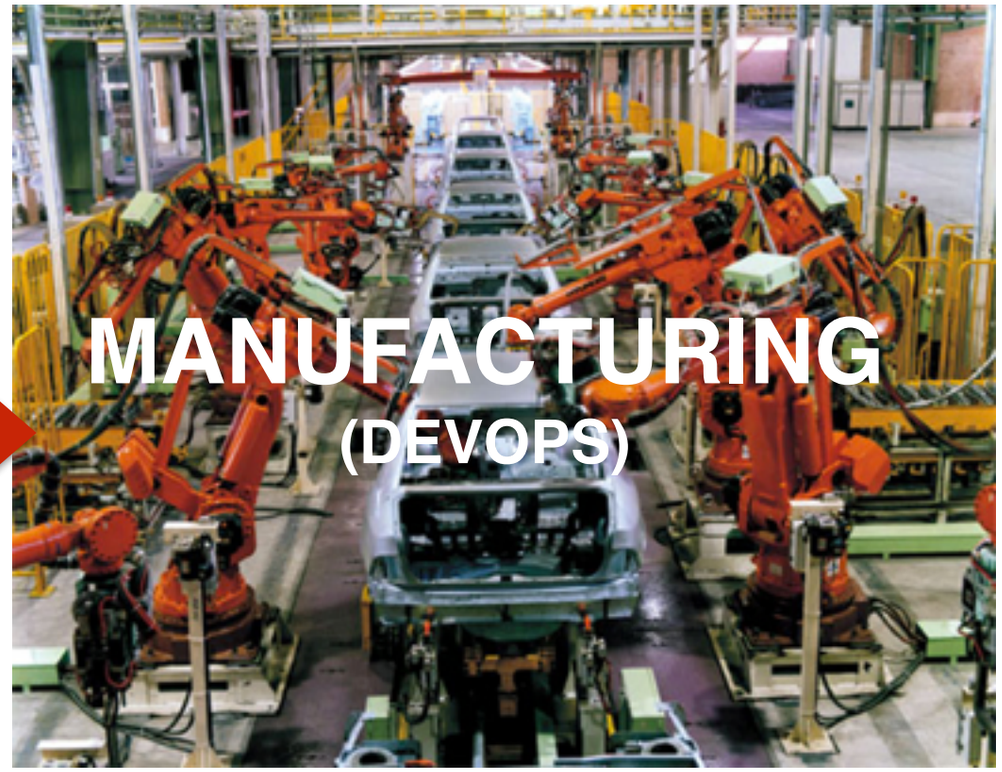


http://www.rackspace.com/blog/a-devops-snapshot-rackspace-survey-shows-businesses-benefits-of-devops-infographic/


CRAFTWORK


WORKSHOP

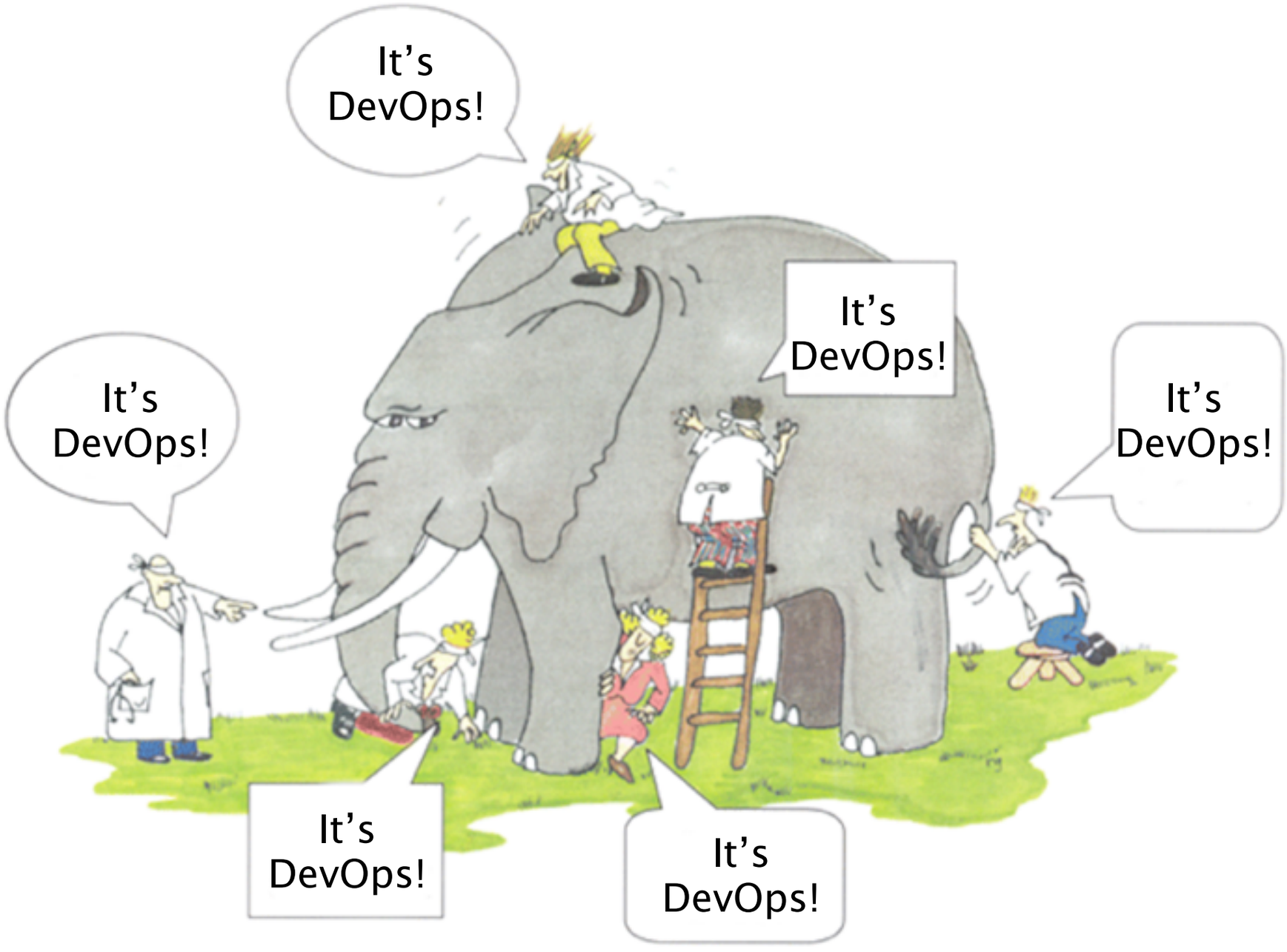

CRAFTWORK


MANUFACTURING
(DEVOPS)


WORKSHOP


FACTORY
(CLOUD)

What is DevOps?



DevOps is...

* A philosophy that starts with passion
* A cultural, professional movement with attitude and values
* A reaction to poor communication
* About creating visibility between dev and ops
* About the symbiotic relationship between dev and ops
* Cross-functional teams over organizational silos
* Products not projects
* Automation over documentation (and more automation... and more...)
* About creating self-service infrastructure for teams
* Knowing that good software doesn't end with development / release
* Software that doesn't require support
* Ensuring a continual feedback loop between development and operations
* Cross-functional teams over organizational silos
* Creating products that are owned by the delivery team
* Knowing that a project is only finished when it is retired from production
* Something you can do without doing agile

https://sites.google.com/a/jezhumble.net/devops-manifesto/

DevOps is...

* A philosophy that starts with passion
* A cultural, professional movement with attitude and values
* A reaction to poor communication
* About creating visibility between dev and ops
* About the symbiotic relationship between dev and ops
* Cross-functional teams over organizational silos
* Products not projects
* Automation over documentation (and more automation... and more...)
* About creating self-service infrastructure for teams
* Knowing that good software doesn't end with development / release
* Software that doesn't require support
* Ensuring a continual feedback loop between development and operations
* Cross-functional teams over organizational silos
* Creating products that are owned by the delivery team
* Knowing that a project is only finished when it is retired from production
* Something you can do without doing agile

https://sites.google.com/a/jezhumble.net/devops-manifesto/

DevOps is...

* A philosophy that starts with passion
* A cultural, professional movement with attitude and values
* A reaction to poor communication
* About creating visibility between dev and ops
* About the symbiotic relationship between dev and ops
* Cross-functional teams over organizational silos
* Products not projects
* Automation over documentation (and more automation... and more...)
* About creating self-service infrastructure for teams
* Knowing that good software doesn't end with development / release
* Software that doesn't require support
* Ensuring a continual feedback loop between development and operations
* Cross-functional teams over organizational silos
* Creating products that are owned by the delivery team
* Knowing that a project is only finished when it is retired from production
* Something you can do without doing agile

https://sites.google.com/a/jezhumble.net/devops-manifesto/

DevOps is...

* A philosophy that starts with passion
* A cultural, professional movement with attitude and values
* A reaction to poor communication
* About creating visibility between dev and ops
* About the symbiotic relationship between dev and ops
* Cross-functional teams over organizational silos
* Products not projects
* Automation over documentation (and more automation... and more...)
* About creating self-service infrastructure for teams
* Knowing that good software doesn't end with development / release
* Software that doesn't require support
* Ensuring a continual feedback loop between development and operations
* Cross-functional teams over organizational silos
* Creating products that are owned by the delivery team
* Knowing that a project is only finished when it is retired from production
* Something you can do without doing agile

https://sites.google.com/a/jezhumble.net/devops-manifesto/

DevOps is...

* A philosophy that starts with passion
* A cultural, professional movement with attitude and values
* A reaction to poor communication
* About creating visibility between dev and ops
* About the symbiotic relationship between dev and ops
* Cross-functional teams over organizational silos
* Products not projects
* Automation over documentation (and more automation... and more...)
* About creating self-service infrastructure for teams
* Knowing that good software doesn't end with development / release
* Software that doesn't require support
* Ensuring a continual feedback loop between development and operations
* Cross-functional teams over organizational silos
* Creating products that are owned by the delivery team
* Knowing that a project is only finished when it is retired from production
* Something you can do without doing agile

https://sites.google.com/a/jezhumble.net/devops-manifesto/

DevOps is...

* A philosophy that starts with passion
* A cultural, professional movement with attitude and values
* A reaction to poor communication
* About creating visibility between dev and ops
* About the symbiotic relationship between dev and ops
* Cross-functional teams over organizational silos
* Products not projects
* Automation over documentation (and more automation... and more...)
* About creating self-service infrastructure for teams
* Knowing that good software doesn't end with development / release
* Software that doesn't require support
* Ensuring a continual feedback loop between development and operations
* Cross-functional teams over organizational silos
* Creating products that are owned by the delivery team
* Knowing that a project is only finished when it is retired from production
* Something you can do without doing agile

https://sites.google.com/a/jezhumble.net/devops-manifesto/

Key Components

- People - “Dev” and “Ops”
- Processes - Continuous Delivery, Agile, …
- Technology - Jenkins, Chef, Puppet, Arquillian, …

Five “C”s of DevOps

Five “C”s of DevOps

- **C**ollaboration between “dev” and “ops”

Five “C”s of DevOps

- **C**ollaboration between “dev” and “ops”
- **C**ulture

Five “C”s of DevOps

- **C**ollaboration between “dev” and “ops”
- **C**ulture
- **C**ode everything - application and configuration

Five “C”s of DevOps

- **C**ollaboration between “dev” and “ops”
- **C**ulture
- **C**ode everything - application and configuration
- **C**onsistency - automation over documentation

Five “C”s of DevOps

- **C**ollaboration between “dev” and “ops”
- **C**ulture
- **C**ode everything - application and configuration
- **C**onsistency - automation over documentation
- **C**ontinuous delivery

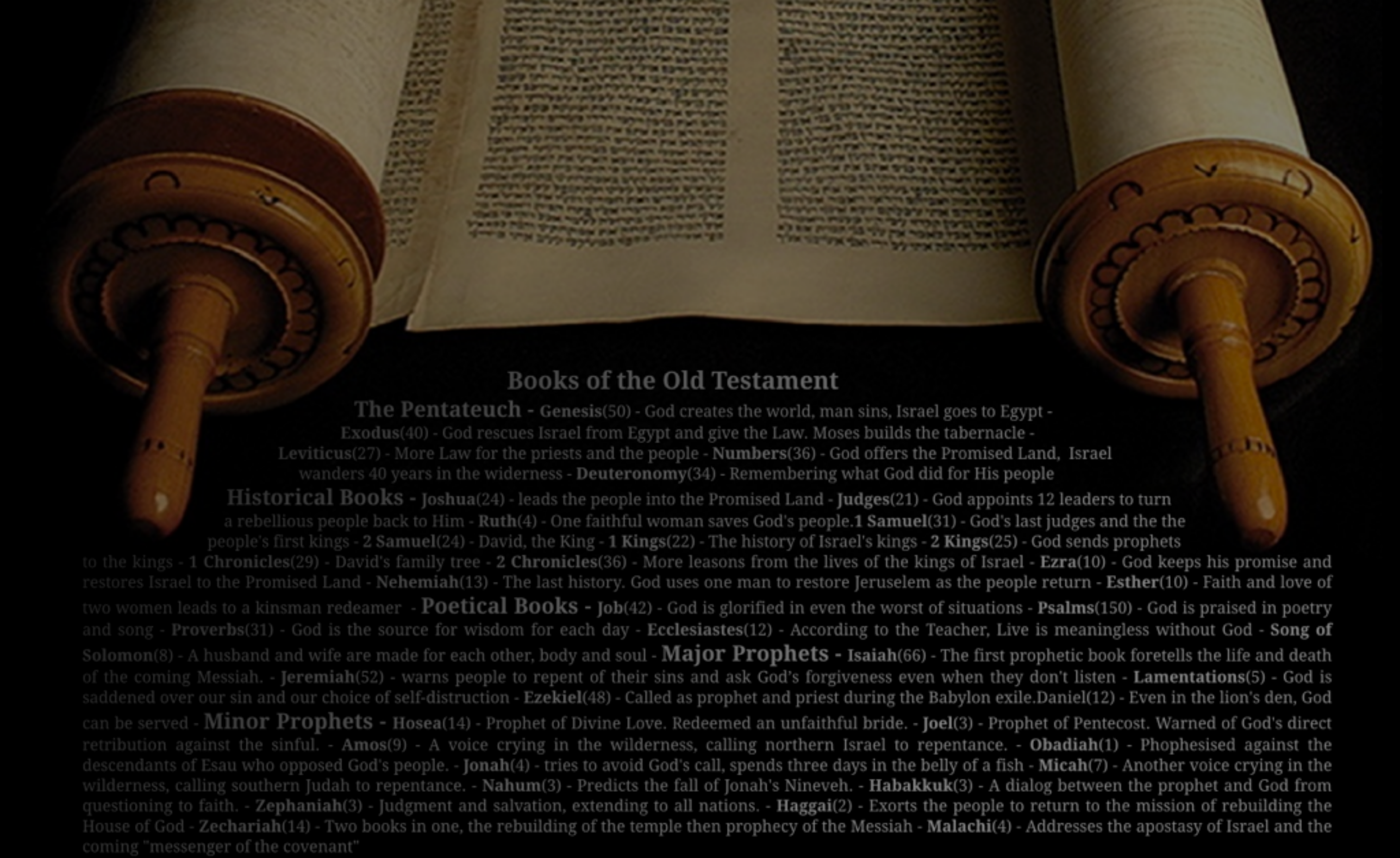
Collaboration

- "Dev"
  - Engineering
  - Test
  - Product management
- "Ops"
  - System administrators
  - Operations staff
  - DBAs
  - Network engineers
  - Security professionals

Collaboration

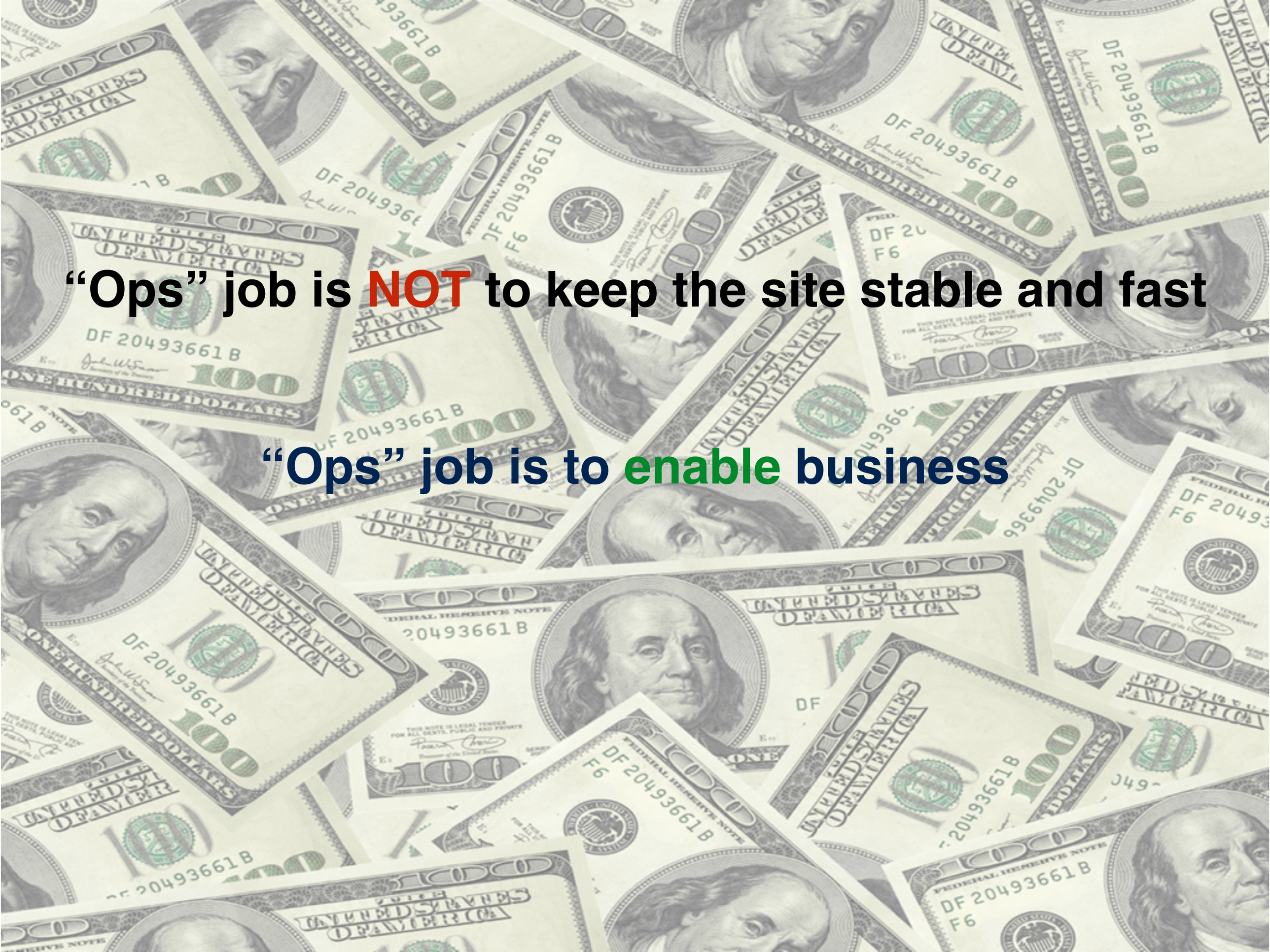
- "Dev"
  - Engineering
  - Test
  - Product management

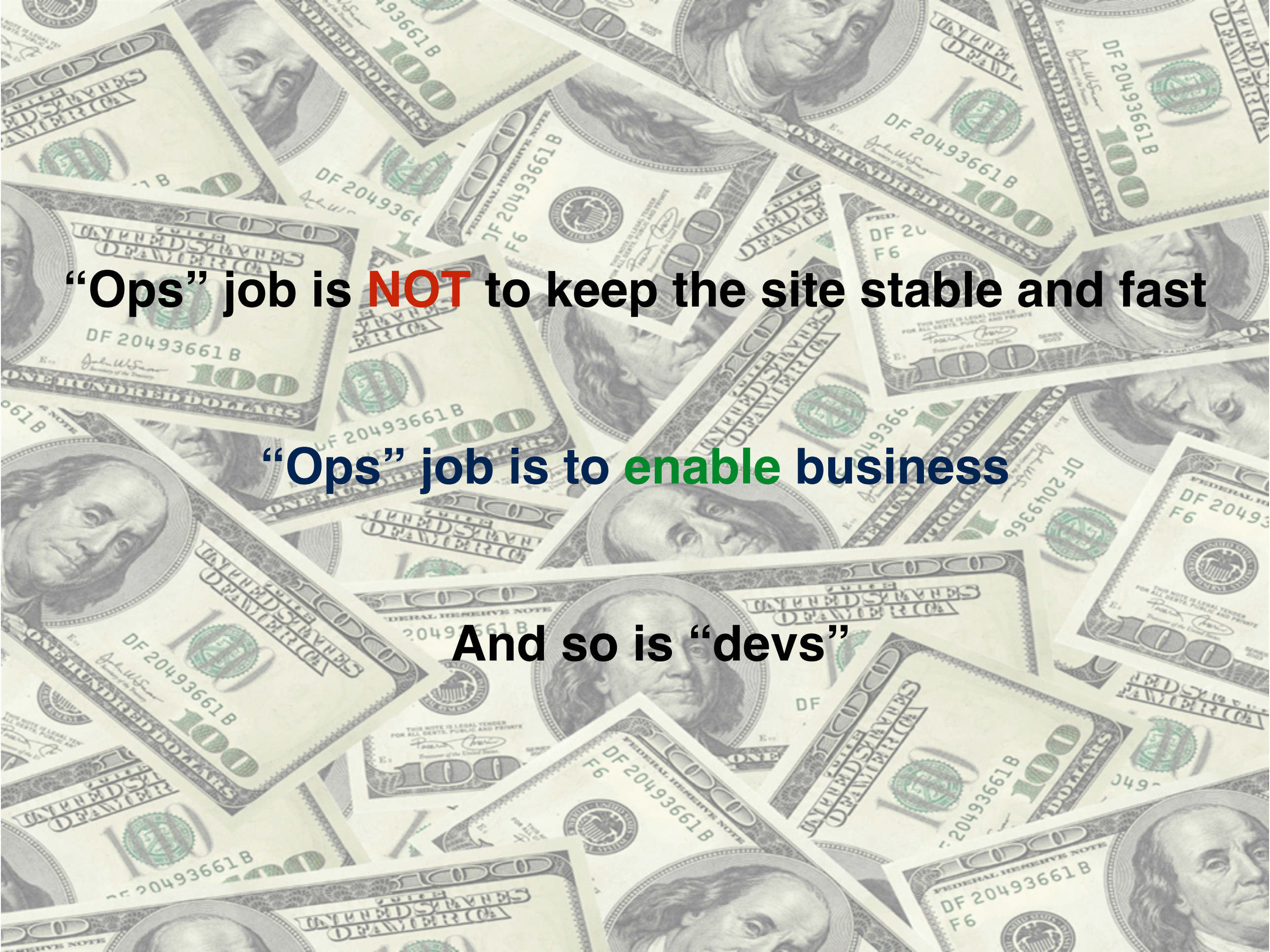
- "Ops"
  - System administrators
  - Operations staff
  - DBAs
  - Network engineers
  - Security professionals



“Devs” job is to add new features
“Ops” job is to keep the site stable and fast

“Ops” job is NOT to keep the site stable and fast


“Ops” job is NOT to keep the site stable and fast
“Ops” job is to enable business


“Ops” job is NOT to keep the site stable and fast
“Ops” job is to enable business
And so is “devs”


“Dev” “Ops”

Culture

Culture

- Respect other's expertise, opinions, responsibilities

Culture

- Respect other's expertise, opinions, responsibilities
- Trust: Ops to think like devs, devs to think like ops

Culture

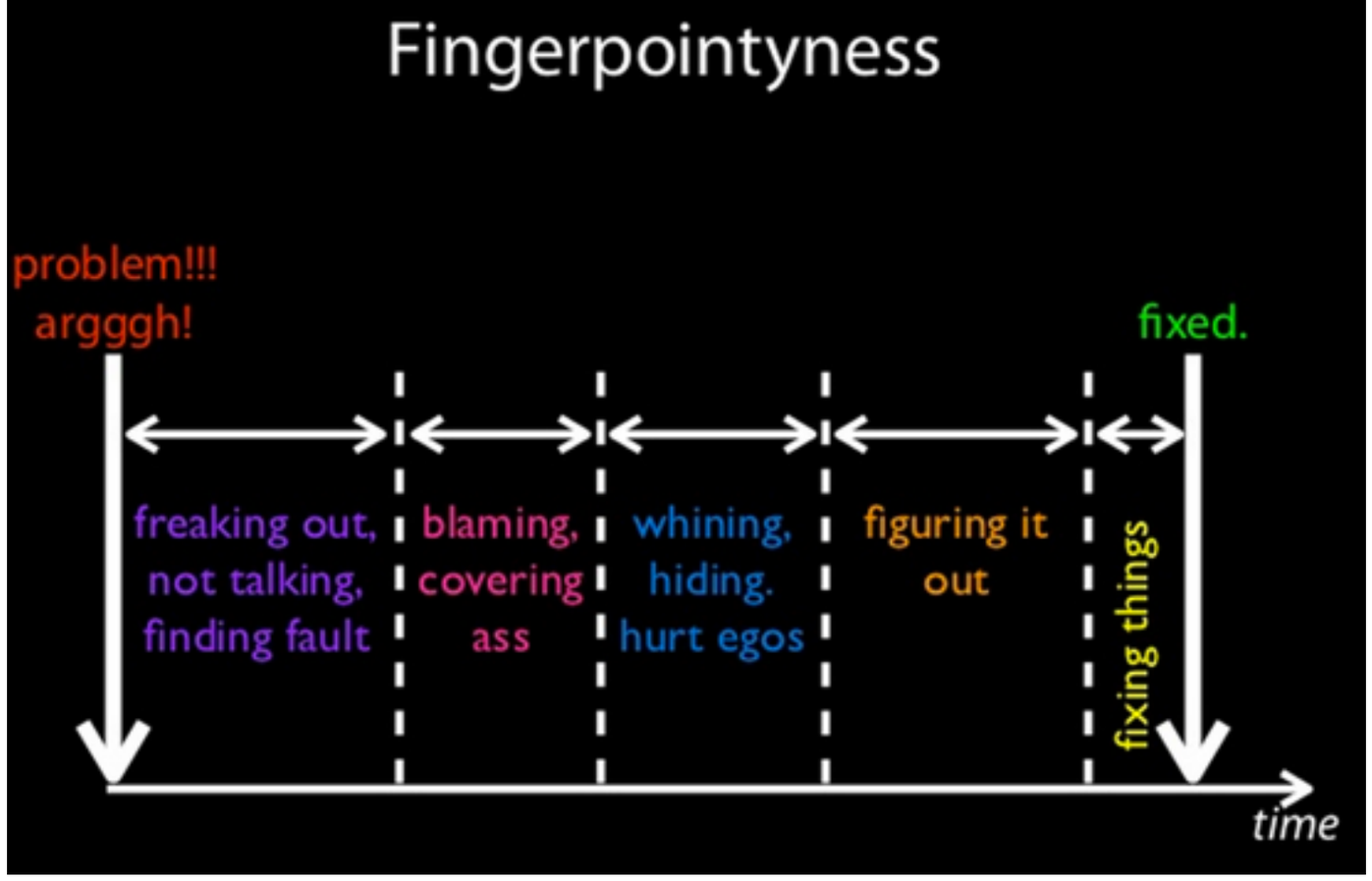
- Respect other's expertise, opinions, responsibilities
- Trust: Ops to think like devs, devs to think like ops
- Leads to transparency

Culture

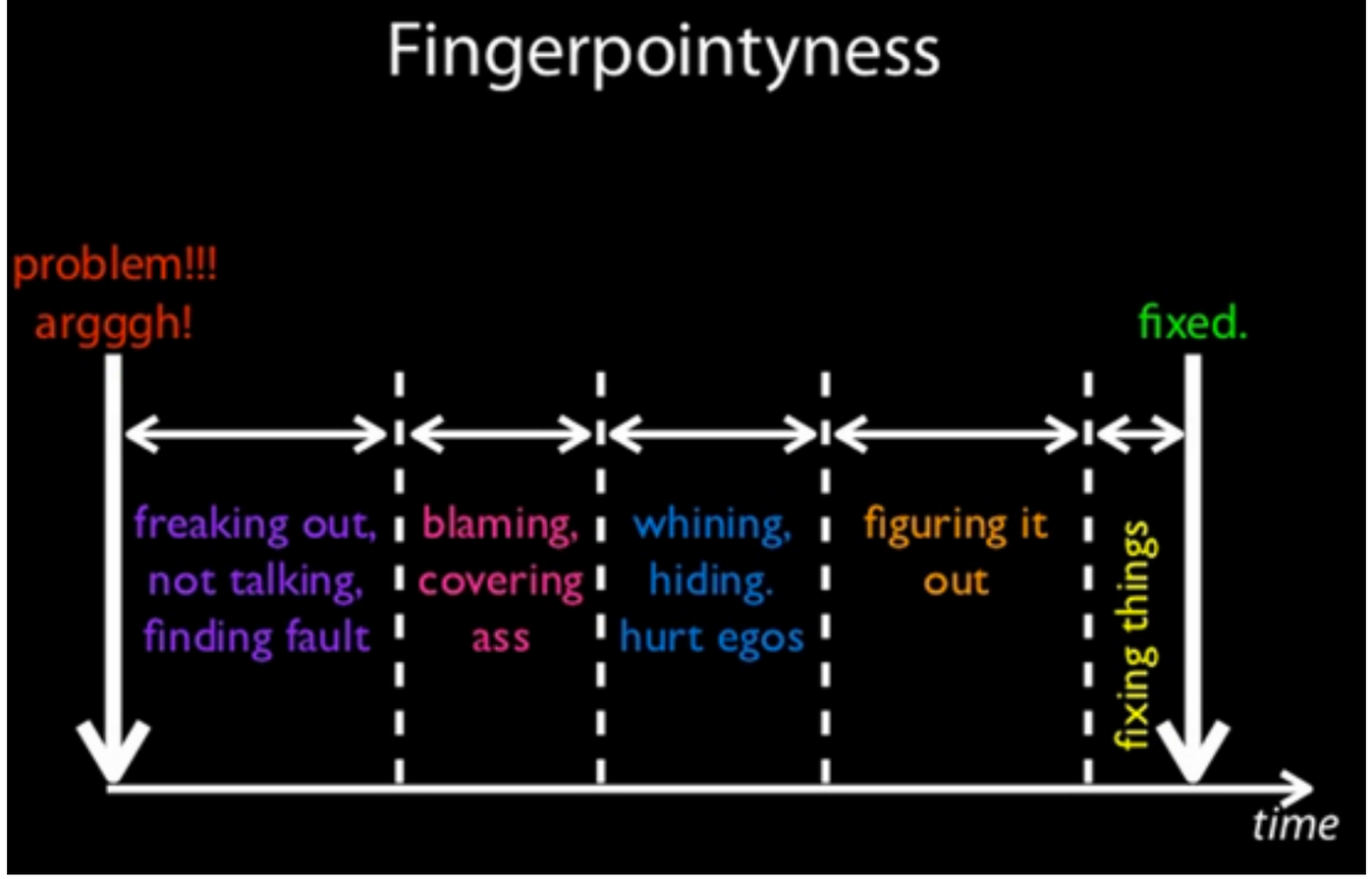
- Respect other's expertise, opinions, responsibilities
- Trust: Ops to think like devs, devs to think like ops
  - Leads to transparency
- Don't ignore failure, build joint recovery plans

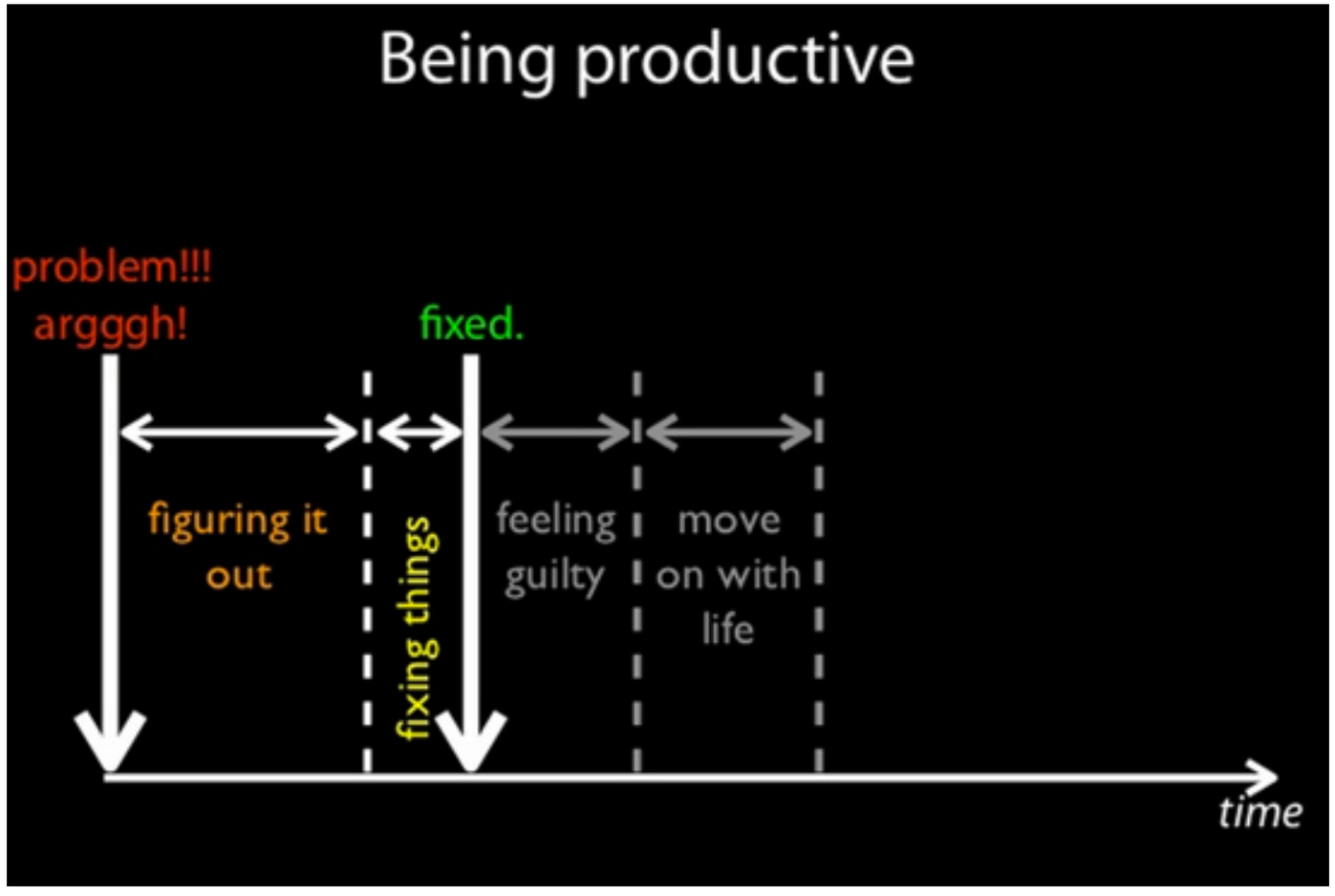
Culture

- Respect other's expertise, opinions, responsibilities
- Trust: Ops to think like devs, devs to think like ops
  - Leads to transparency
- Don't ignore failure, build joint recovery plans
- Amplify feedback loops

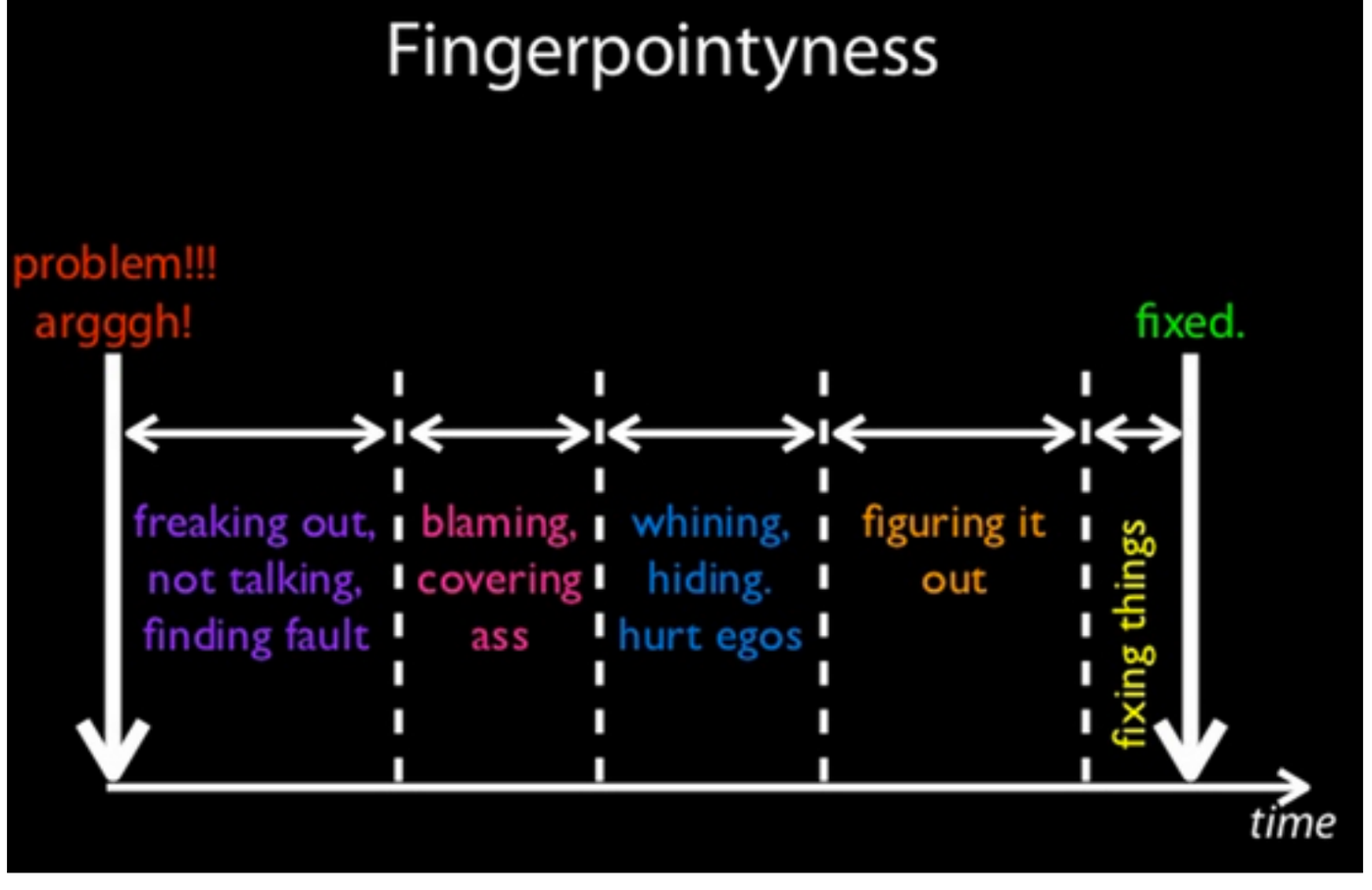


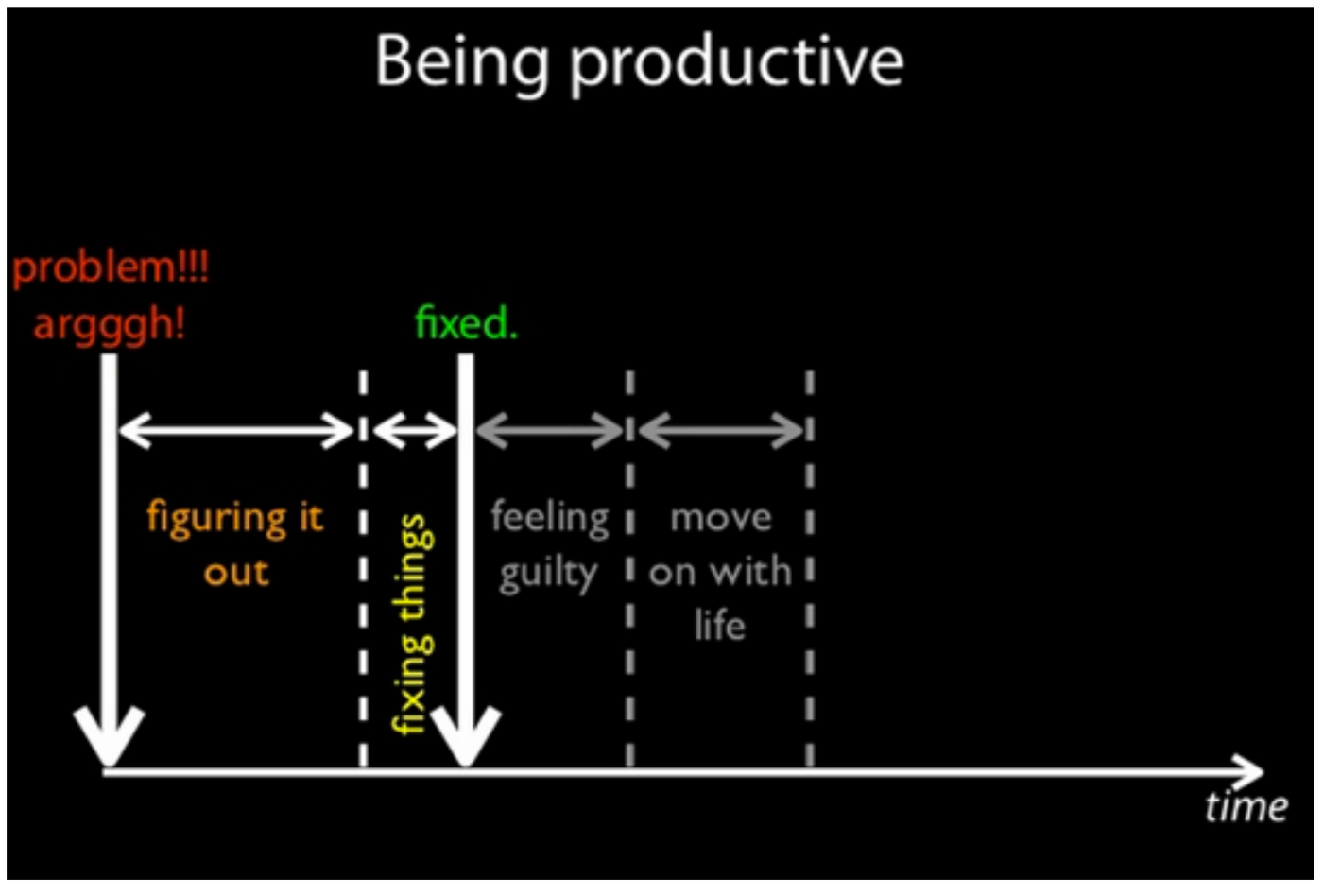
http://www.slideshare.net/jallspaw/10-deploys-per-day-dev-and-ops-cooperation-at-flickr





http://www.slideshare.net/jallspaw/10-deploys-per-day-dev-and-ops-cooperation-at-flickr



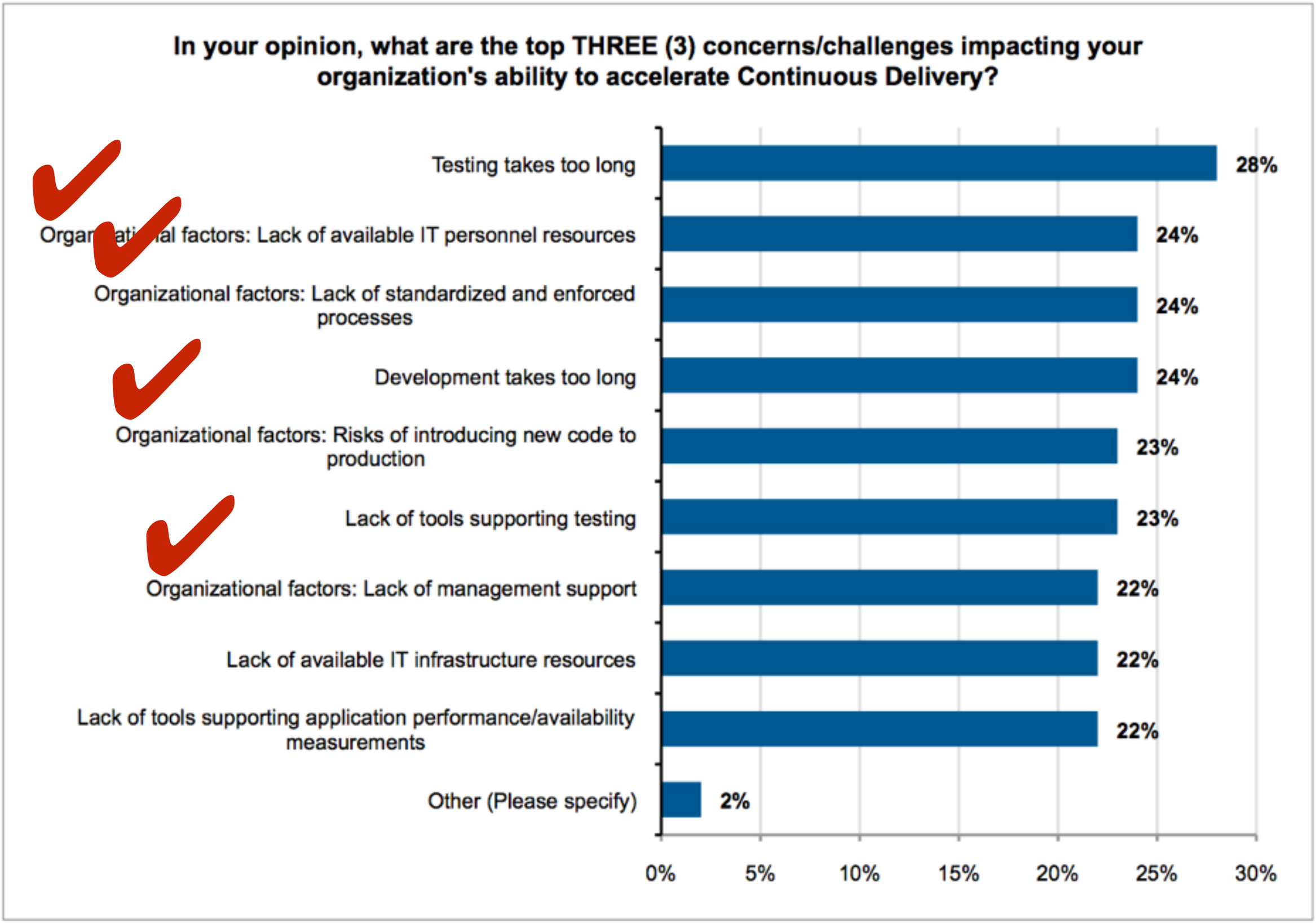


http://www.slideshare.net/jallspaw/10-deploys-per-day-dev-and-ops-cooperation-at-flickr

"Treat people warmly, issues coldly!"

With great power, comes great responsibility

"you build it, you run it!"



http://pages.cloudbees.com/rs/cloudbees/images/CloudBees-EMA-DevOps_2014.pdf

Code everything

- Application code
- Build scripts
- Database schema
- Configuration files
- IDE configurations
- Infrastructure
- Deployment scripts
- Test code and scripts
- Provisioning scripts
- Monitoring
- Logging
- …

“Never send a human to do a machine’s job”

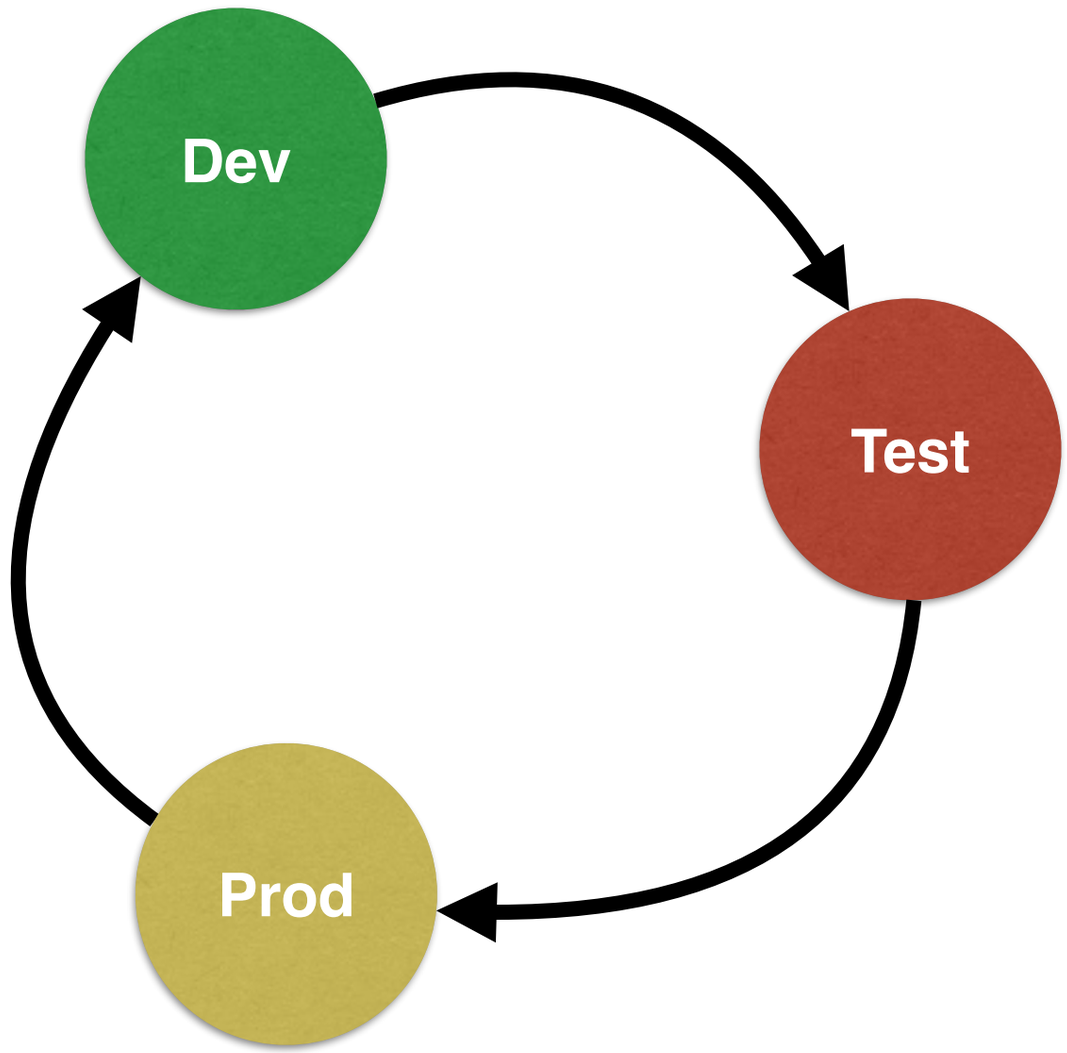
Consistency

Consistency

- Automation over documentation

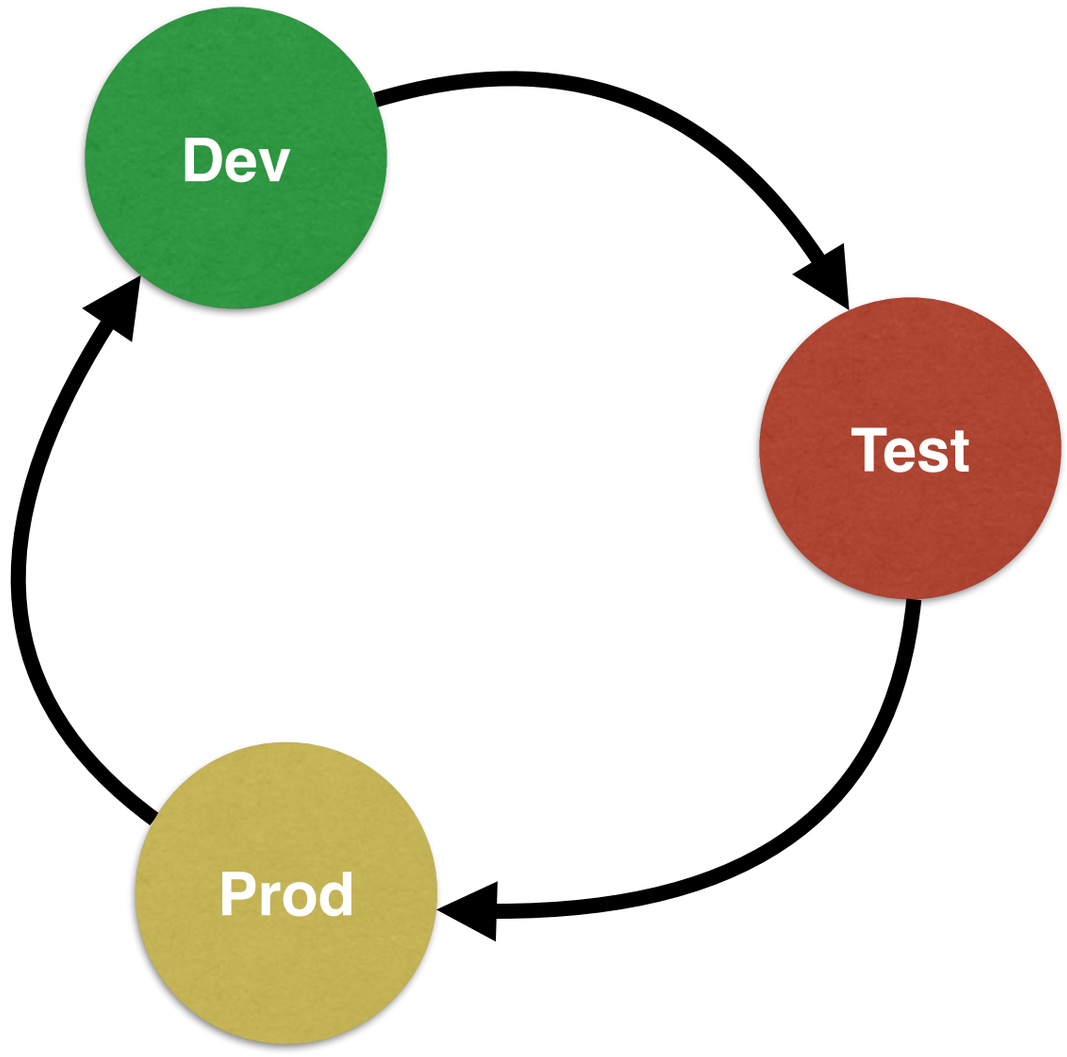
Consistency

- Automation over documentation
  - Repeatability



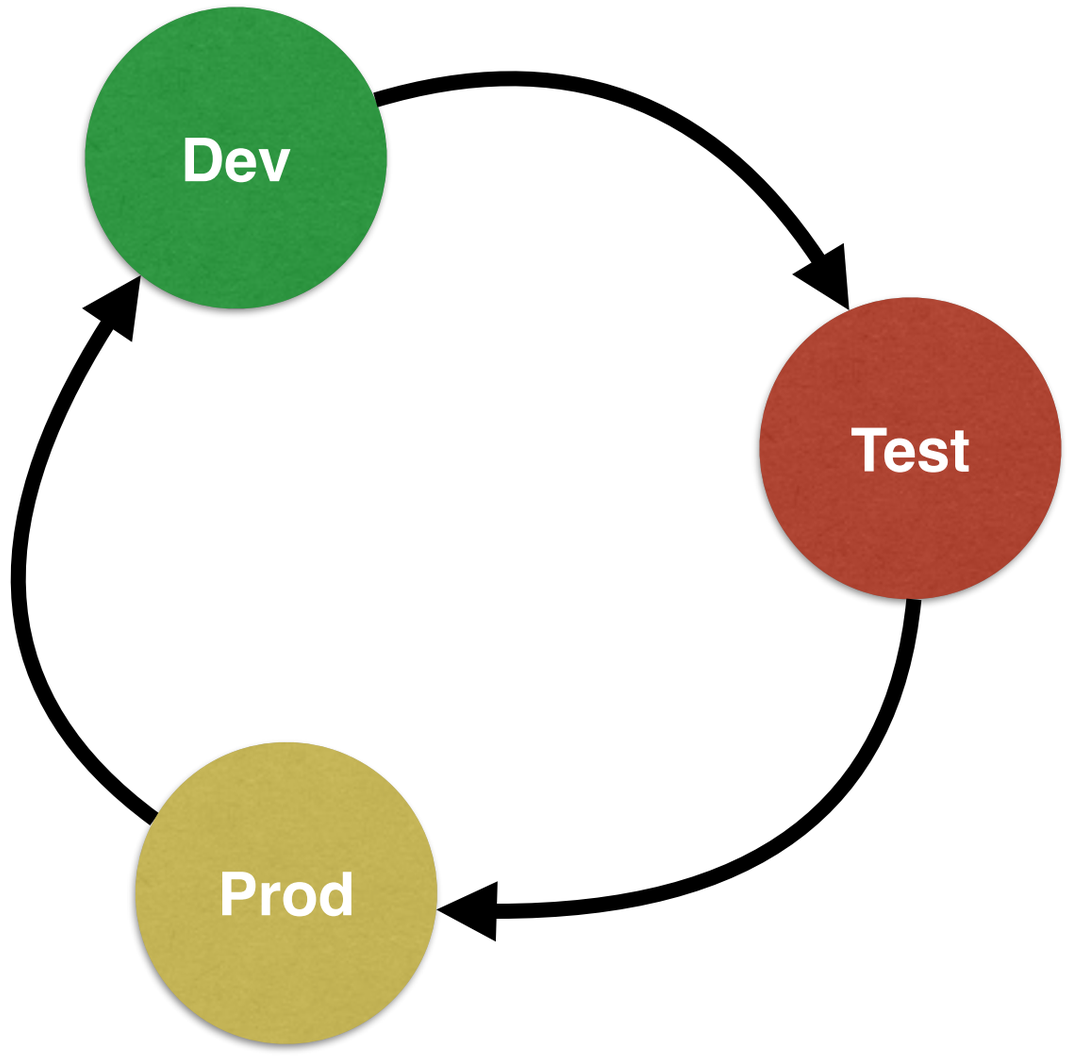
Consistency

- Automation over documentation
  - Repeatability
  - Push-button deployments



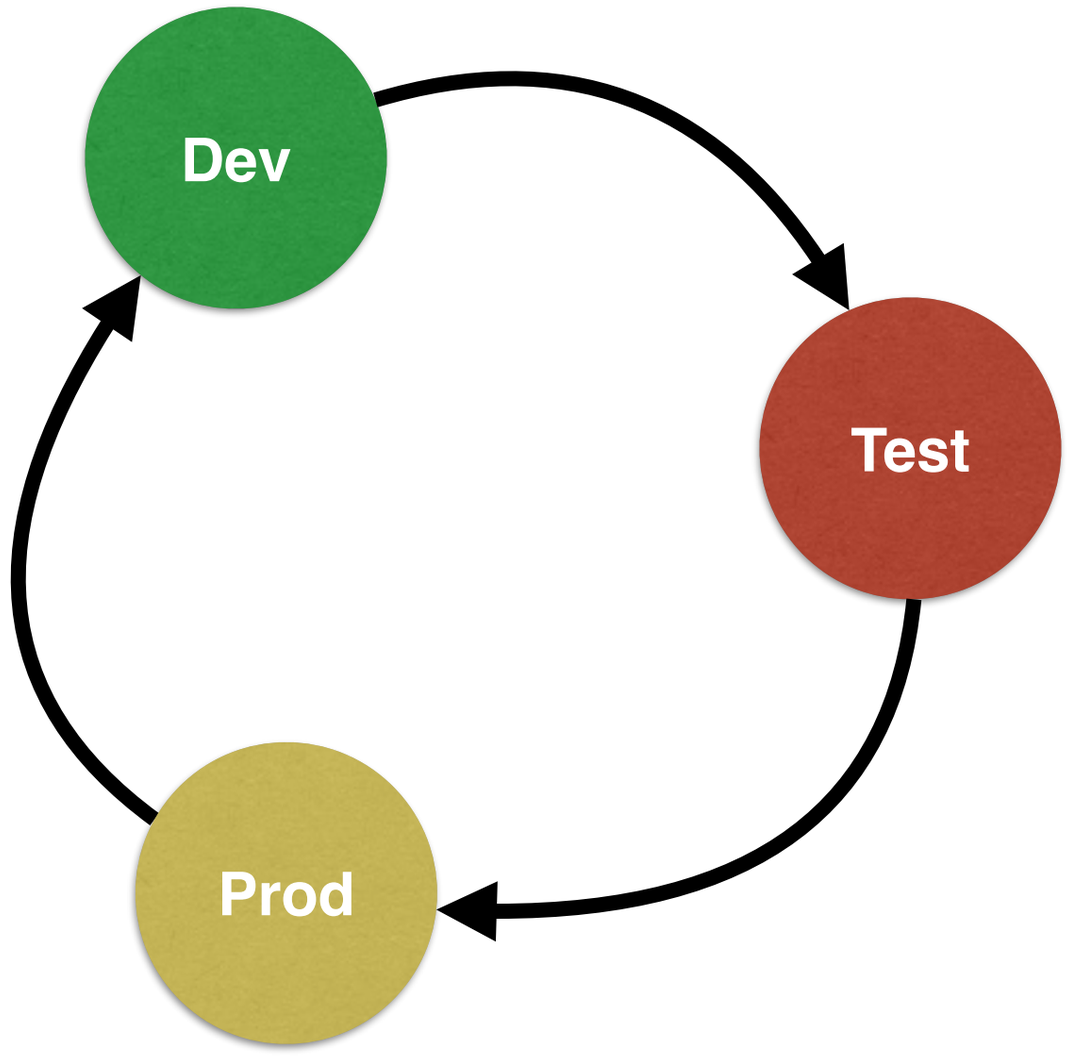
Consistency

- Automation over documentation
  - Repeatability
  - Push-button deployments
  - Managing environments



Consistency

- Automation over documentation
  - Repeatability
  - Push-button deployments
  - Managing environments



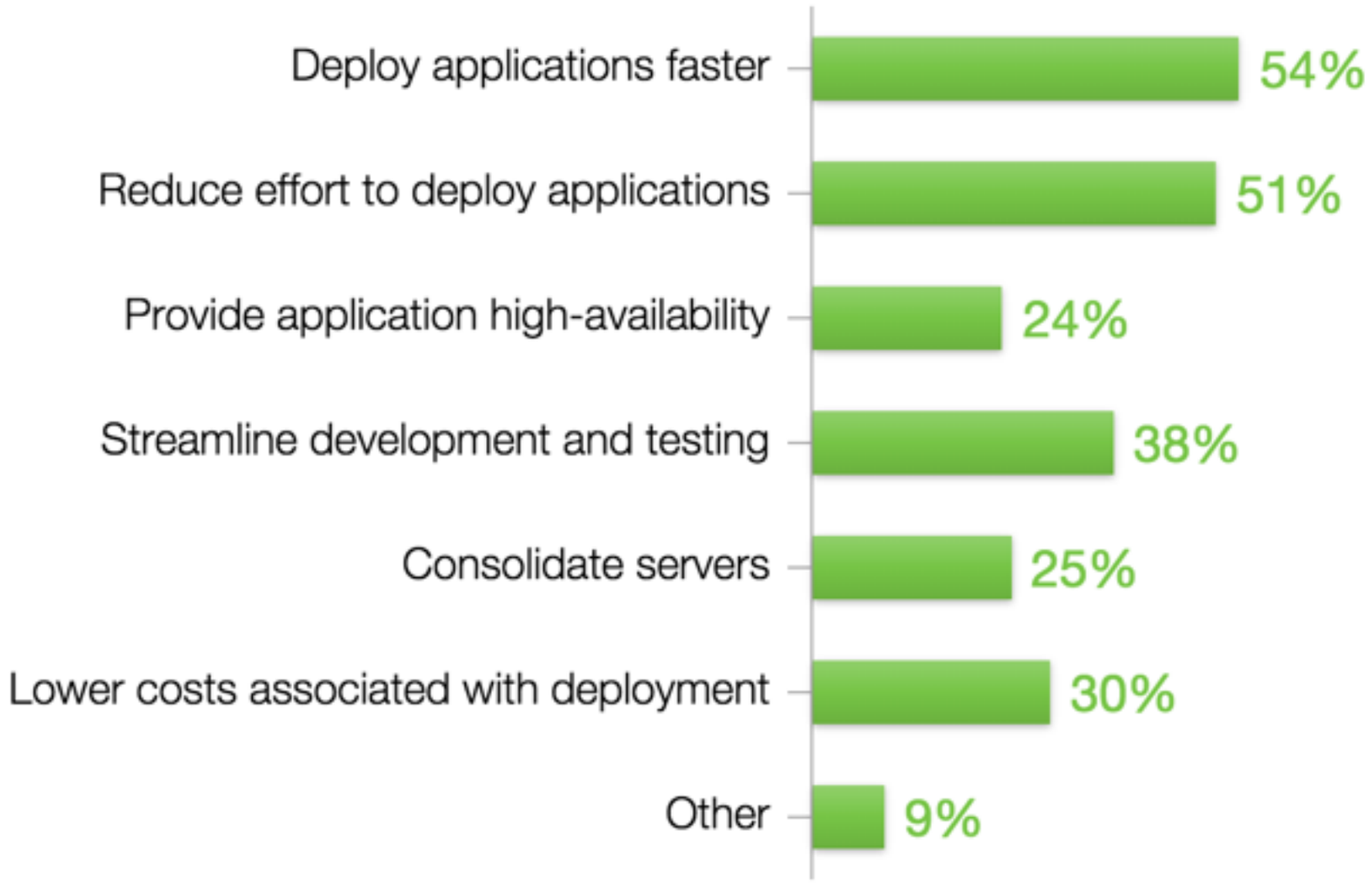
Manage environments

- Cloud computing: PaaS, OpenShift, …
- Virtualization: Virtual Box, Vagrant, …
- Containers: Docker, Rocket, …
- Configuration tools: Puppet, Chef, Ansible, Salt
- Orchestration: Kubernetes, Swarm, …

- Simple and portable way to create VMs
- Works with VirtualBox, VMWare, Docker, …
- Environments packaged as “boxes”
- Commands: **`vagrant up`**, **`destroy`**, **`login`**, …
- Minimize impedance mismatch between Dev, QA, Prod

Benefits of containers

What are the top benefits you see with containers?



Note: this is a multiple-choice question – response percentages may not add up to 100.

Source: TechValidate survey of 79 IT professionals

http://www.techvalidate.com/product-research/red-hat-enterprise-linux/charts/E21-1EE-BF2

Containers

Containers

- Faster deployments

Containers

- Faster deployments
- Isolation

Containers

- Faster deployments
- Isolation
- Portability - “it works on my machine”

Containers

- Faster deployments
- Isolation
- Portability - “it works on my machine”
- Snapshotting

Containers

- Faster deployments
- Isolation
- Portability - “it works on my machine”
- Snapshotting
- Security sandbox

Containers

- Faster deployments
- Isolation
- Portability - “it works on my machine”
- Snapshotting
- Security sandbox
- Limit resource usage

Containers

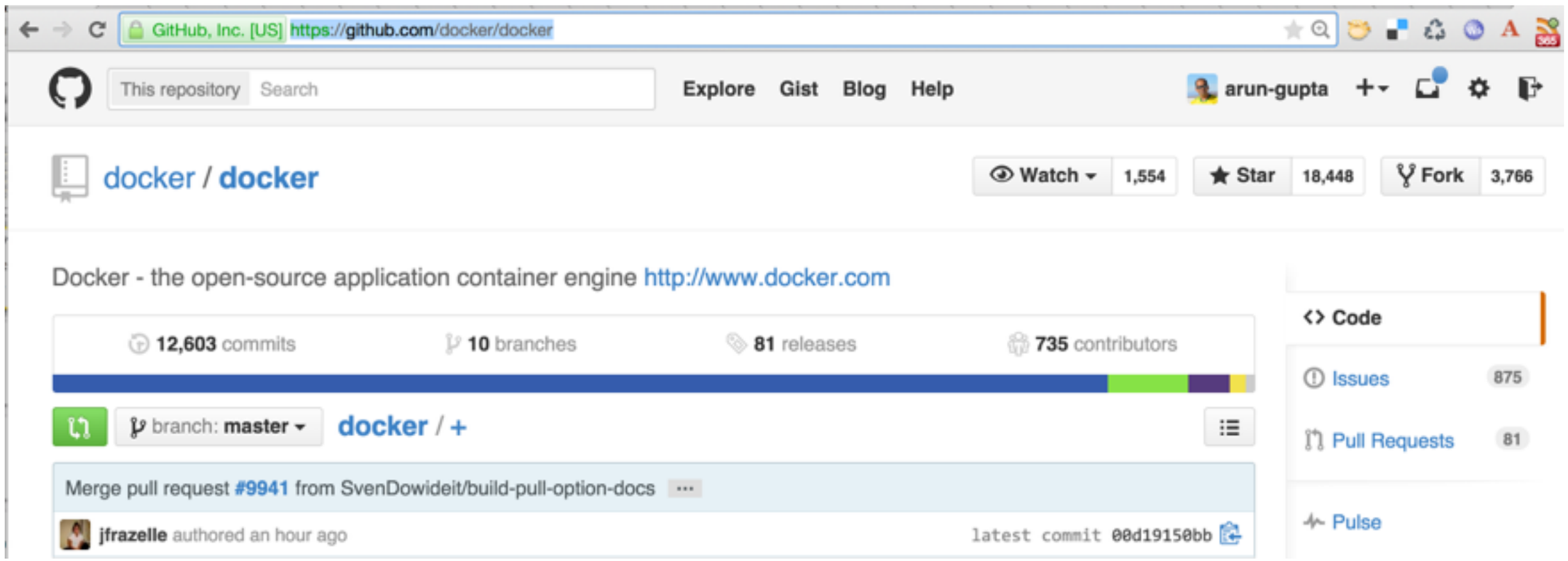
- Faster deployments
- Isolation
- Portability - “it works on my machine”
- Snapshotting
- Security sandbox
- Limit resource usage
- Simplified dependency

Containers

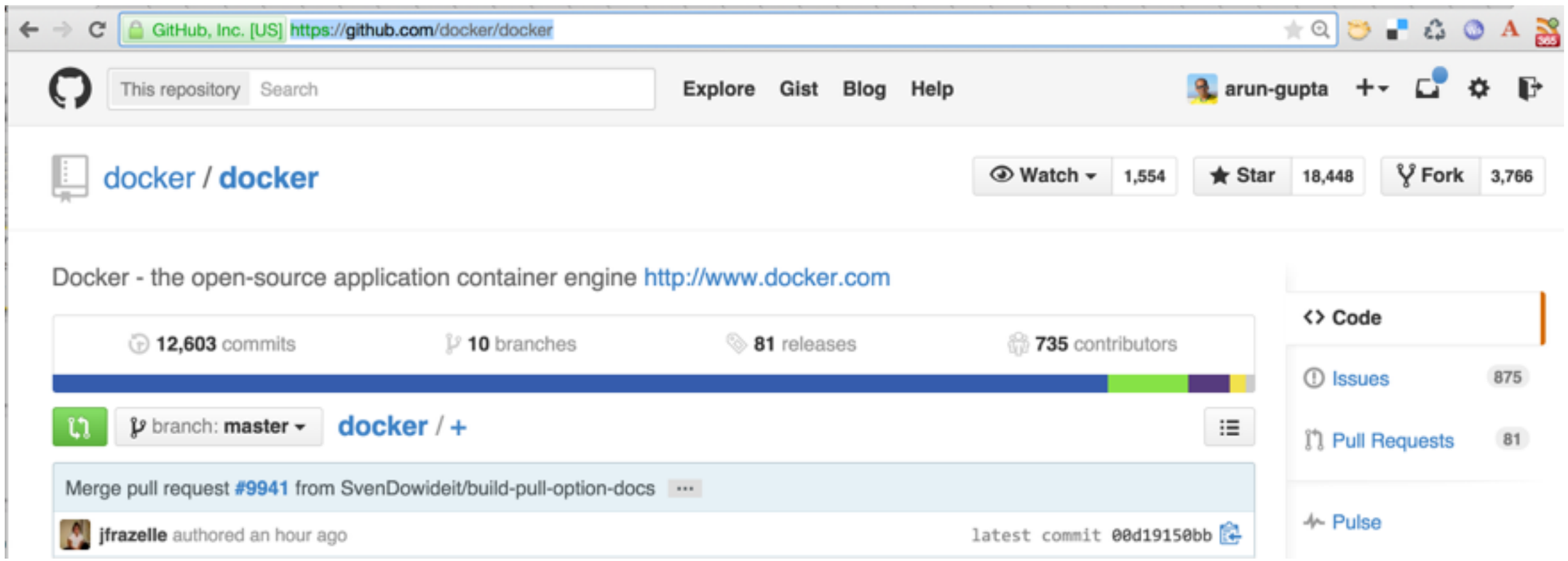
- Faster deployments
- Isolation
- Portability - “it works on my machine”
- Snapshotting
- Security sandbox
- Limit resource usage
- Simplified dependency
- Sharing


docker

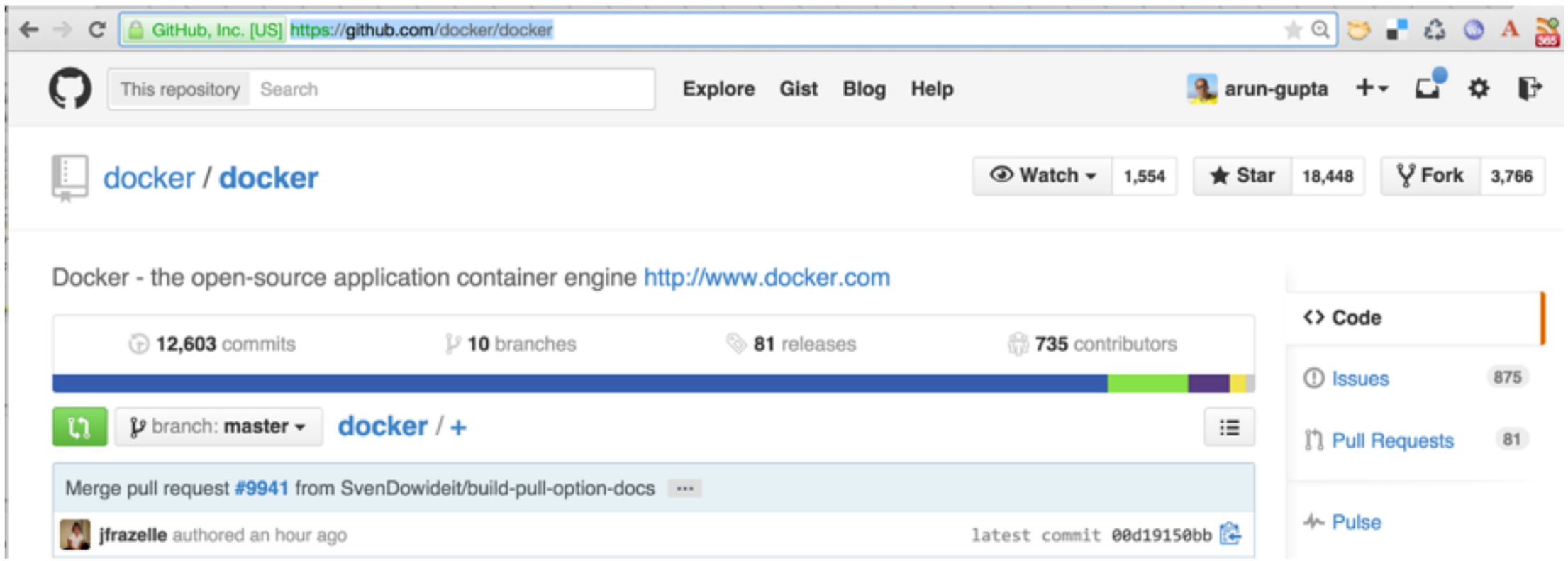
- Open source project and company

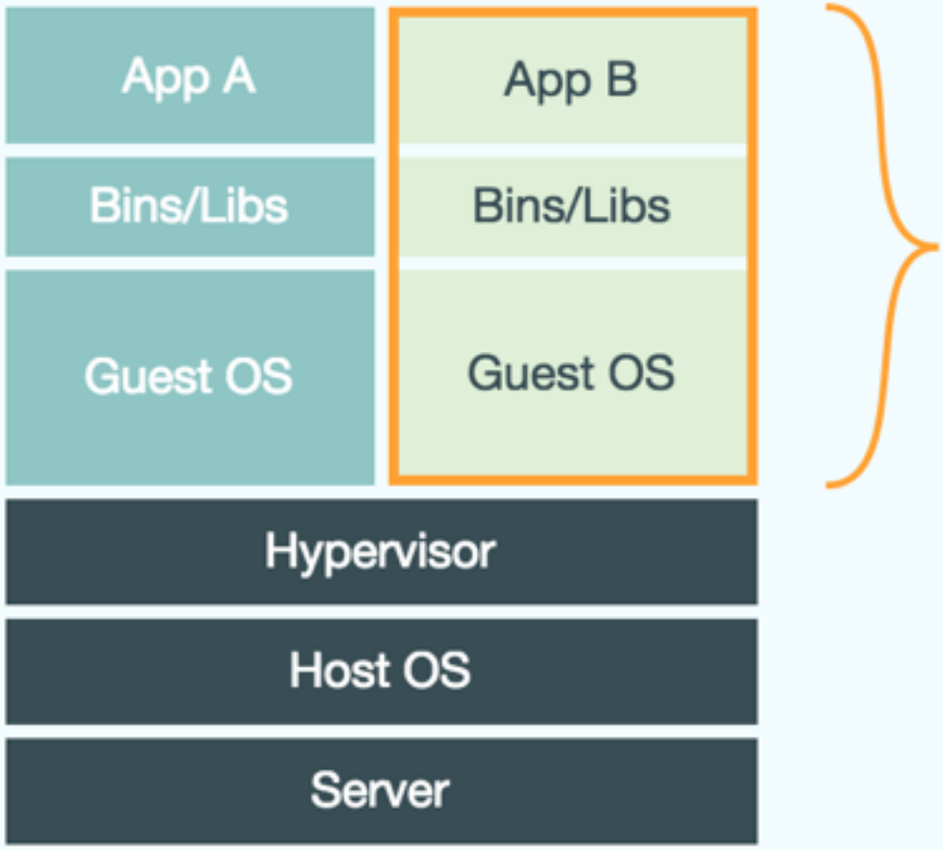


- Open source project and company
- Used to create containers for software applications



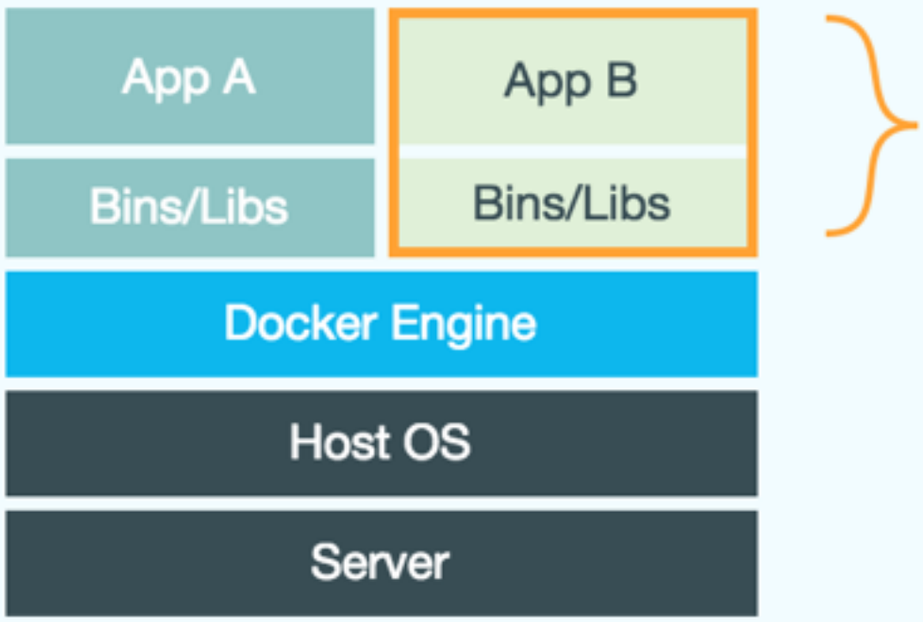
- Open source project and company
- Used to create containers for software applications
- Package Once Deploy Anywhere (PODA)





Virtual Machines

Each virtualized application includes not only the application - which may be only 10s of MB - and the necessary binaries and libraries, but also an entire guest operating system - which may weigh 10s of GB.



Docker

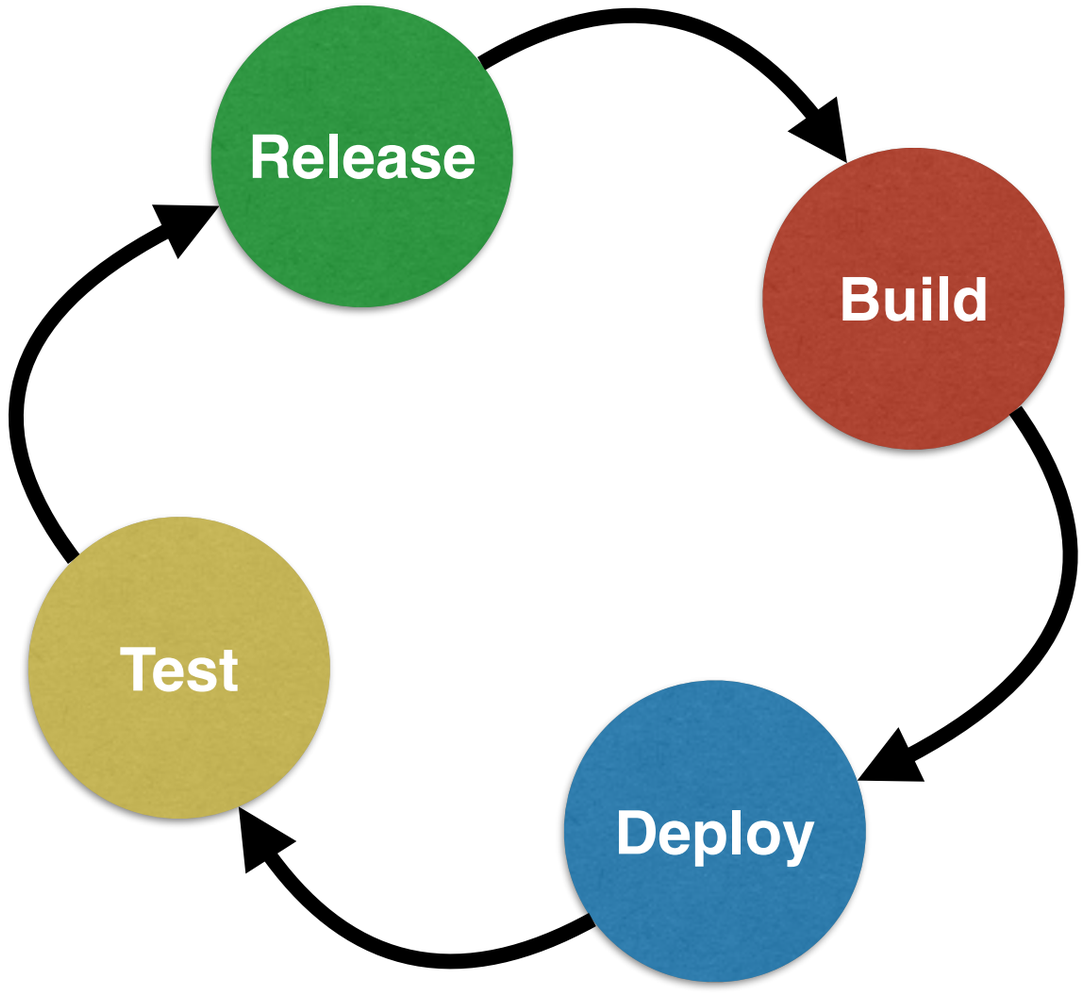
The Docker Engine container comprises just the application and its dependencies. It runs as an isolated process in userspace on the host operating system, sharing the kernel with other containers. Thus, it enjoys the resource isolation and allocation benefits of VMs but is much more portable and efficient.

Dashboards

- Build dashboards, improve transparency

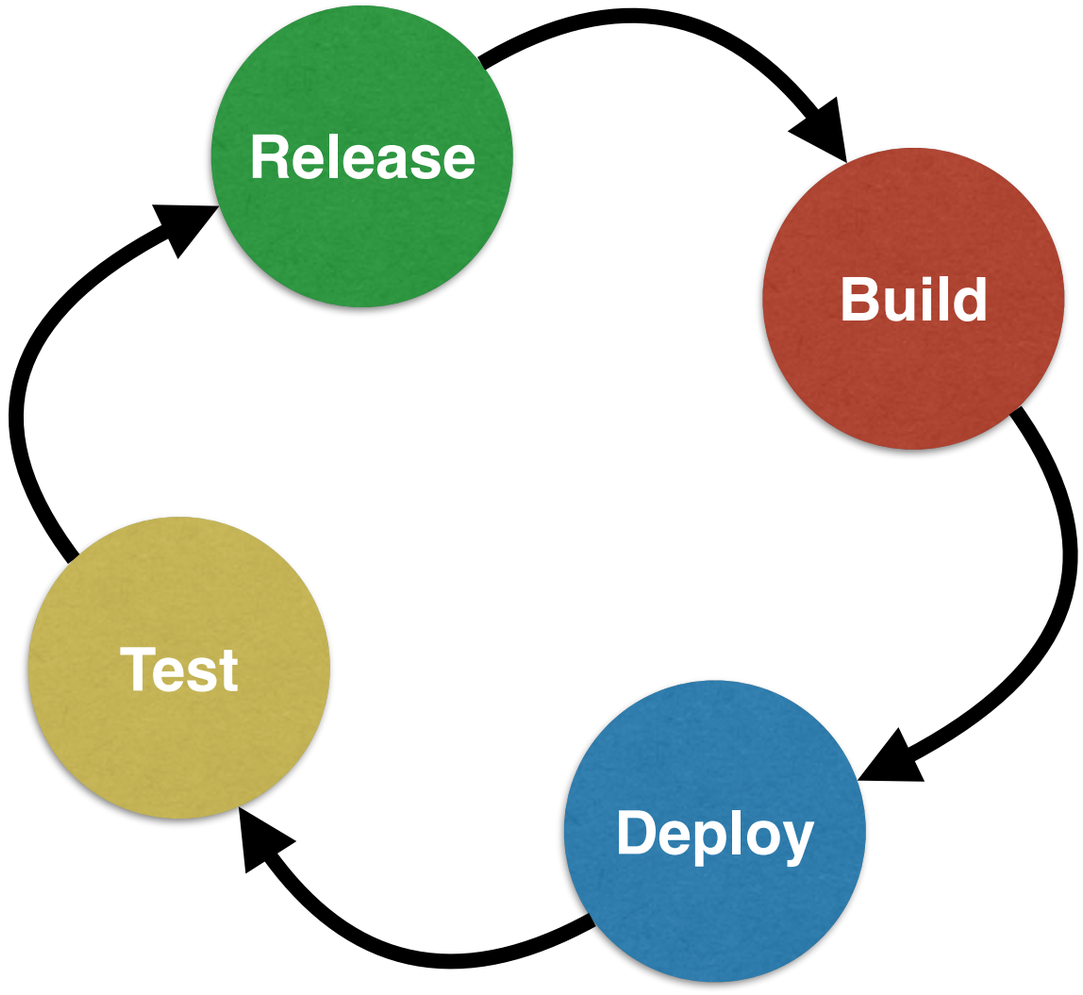
- stackexchange.com/performance

Continuous Delivery



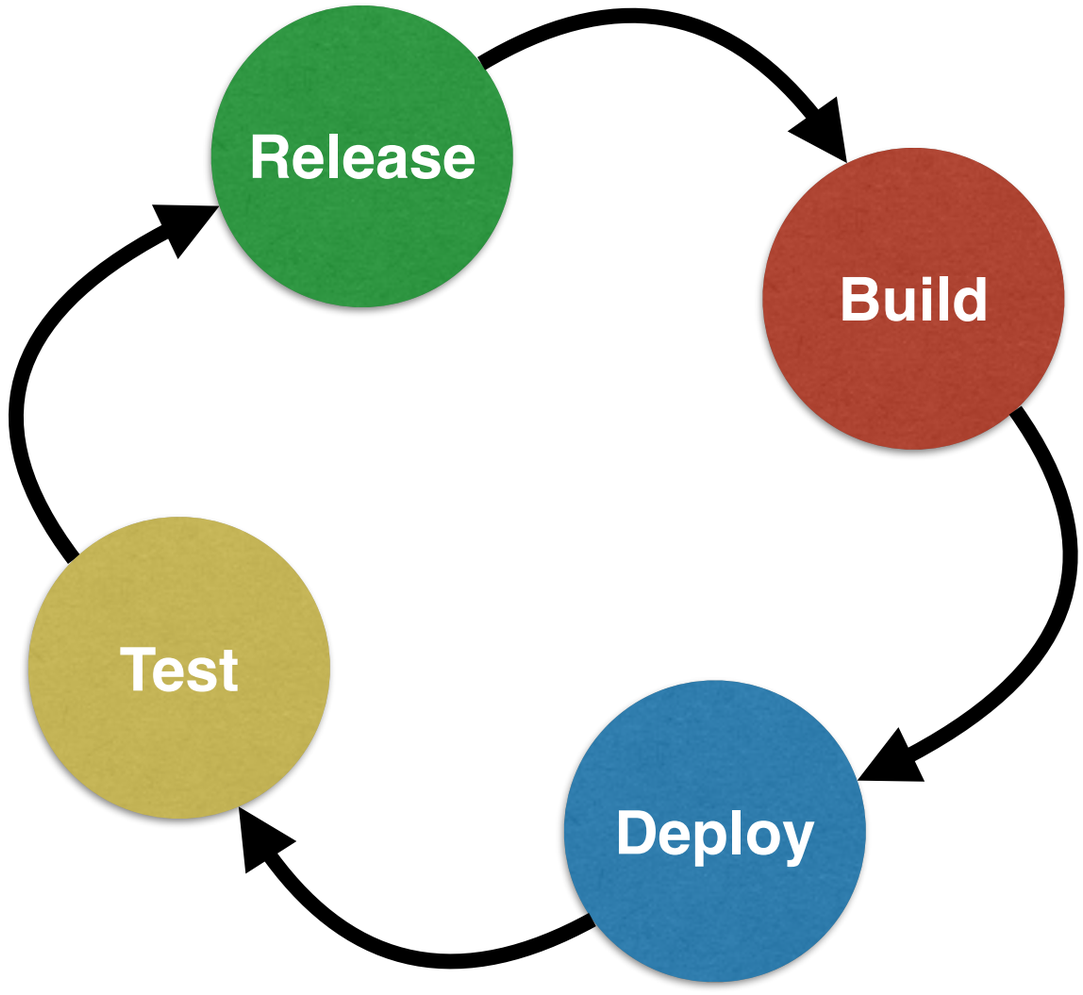
Continuous Delivery

- Agile Manifesto



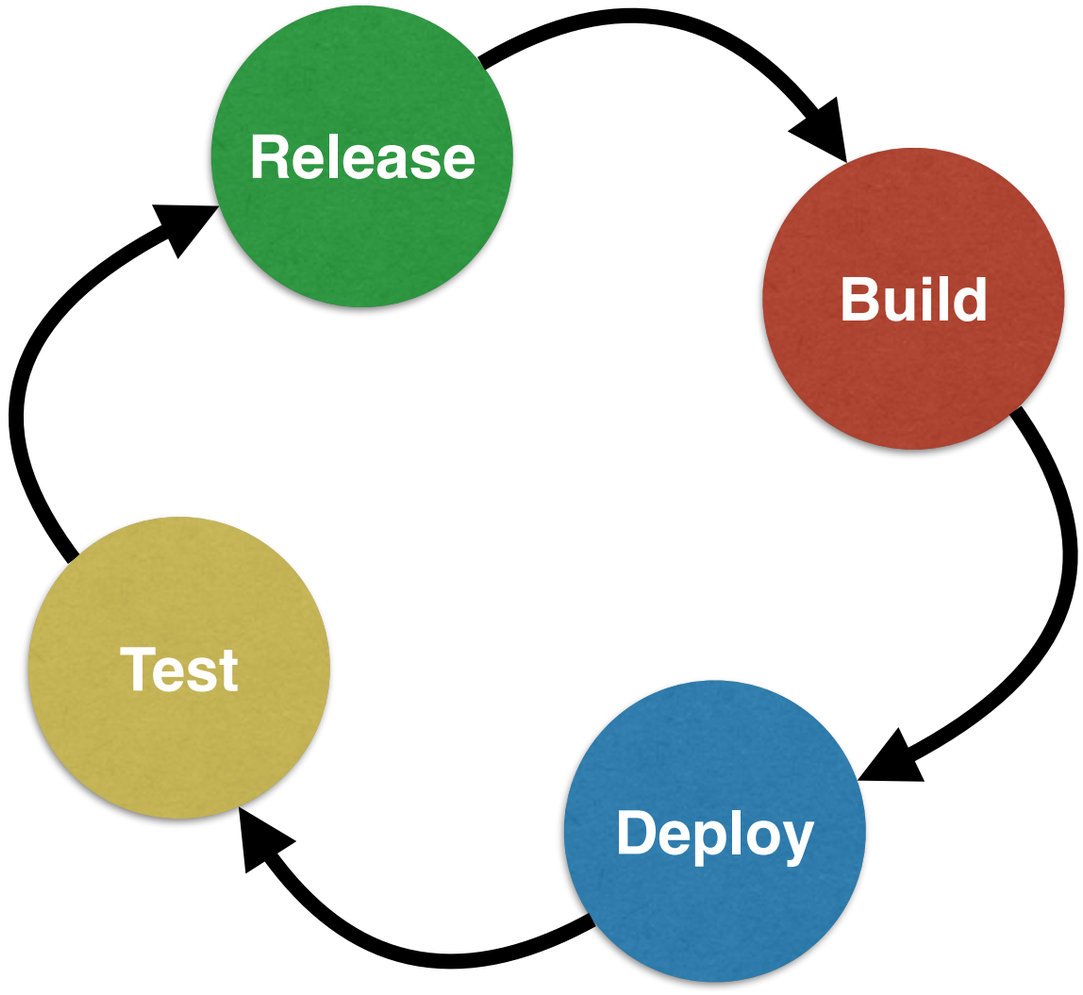
Continuous Delivery

- Agile Manifesto
- Continuous Integration



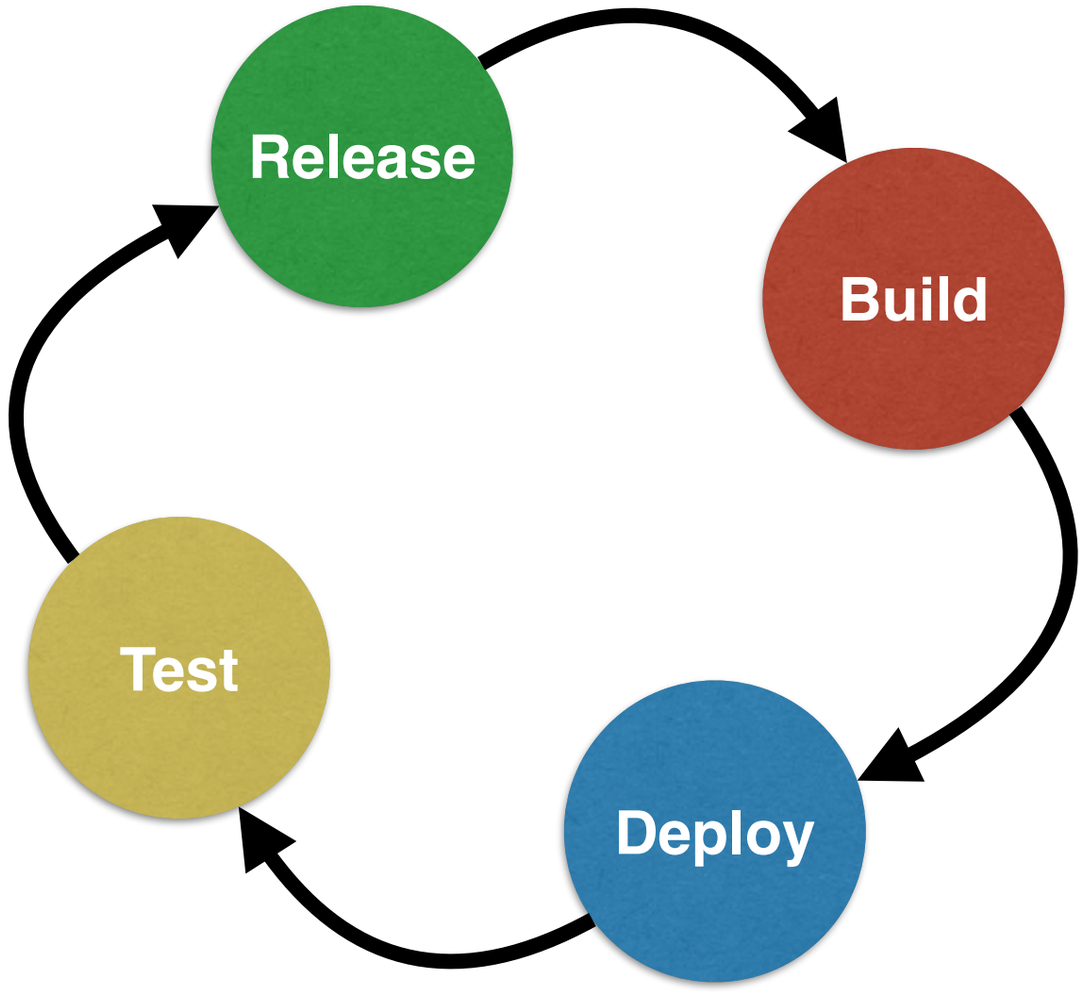
Continuous Delivery

- Agile Manifesto
- Continuous Integration
- Fail fast and recover



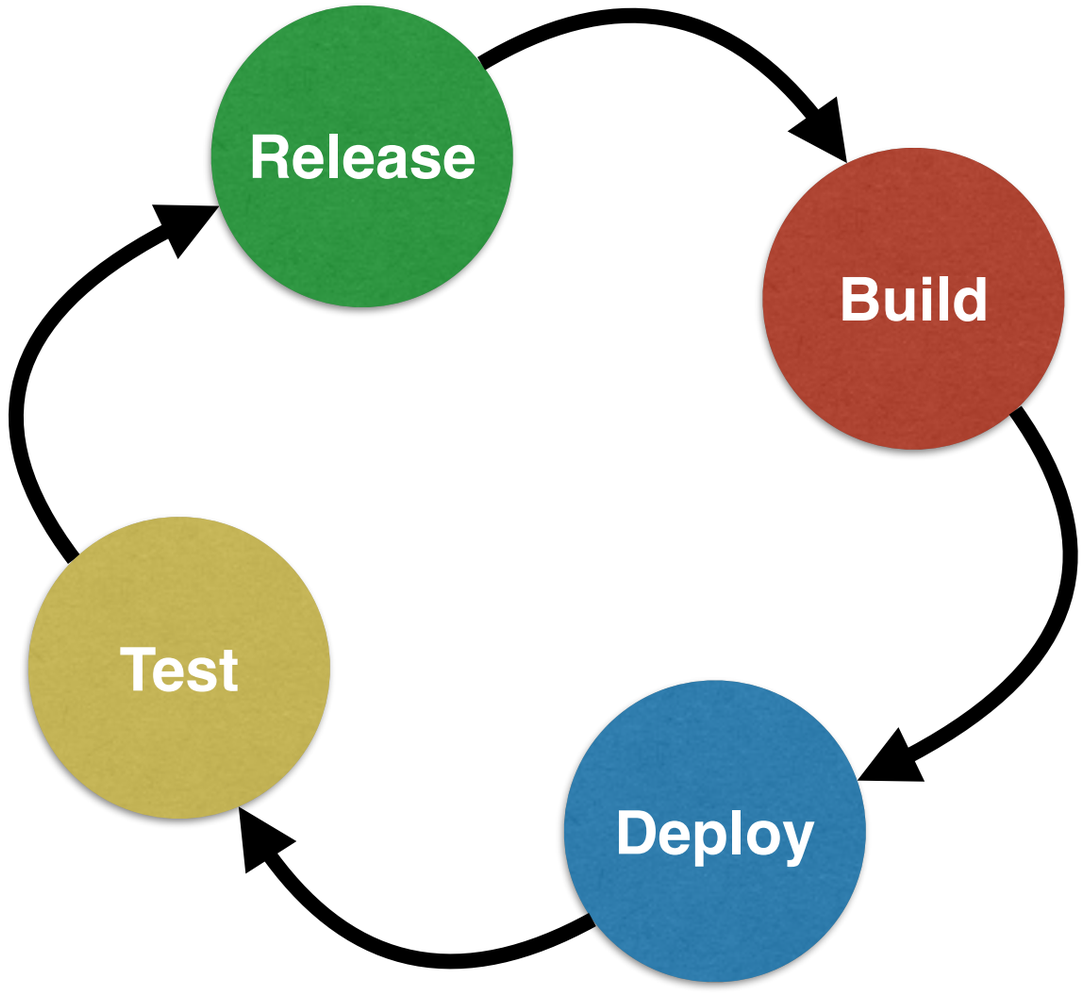
Continuous Delivery

- Agile Manifesto
- Continuous Integration
- Fail fast and recover
- Self service



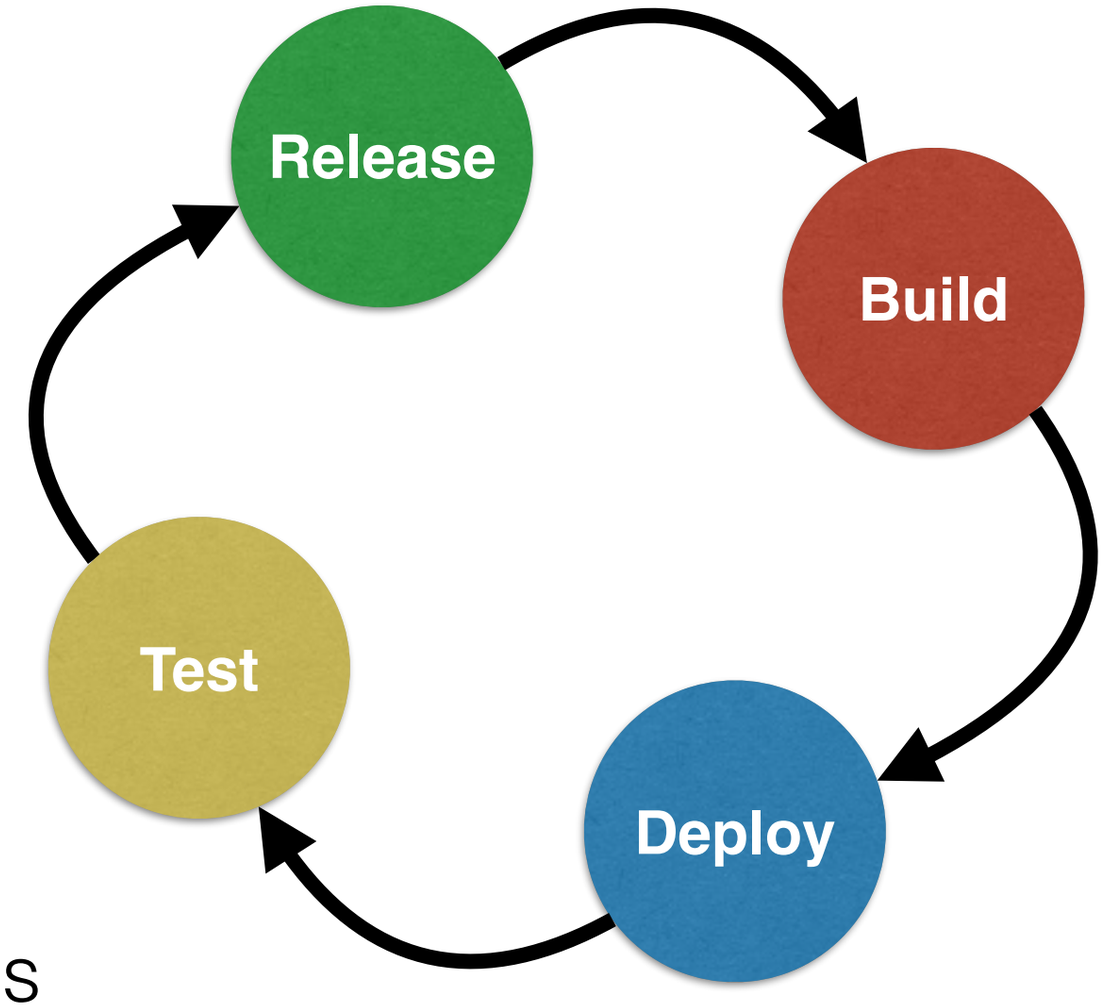
Continuous Delivery

- Agile Manifesto
- Continuous Integration
- Fail fast and recover
- Self service
- 100% Automation



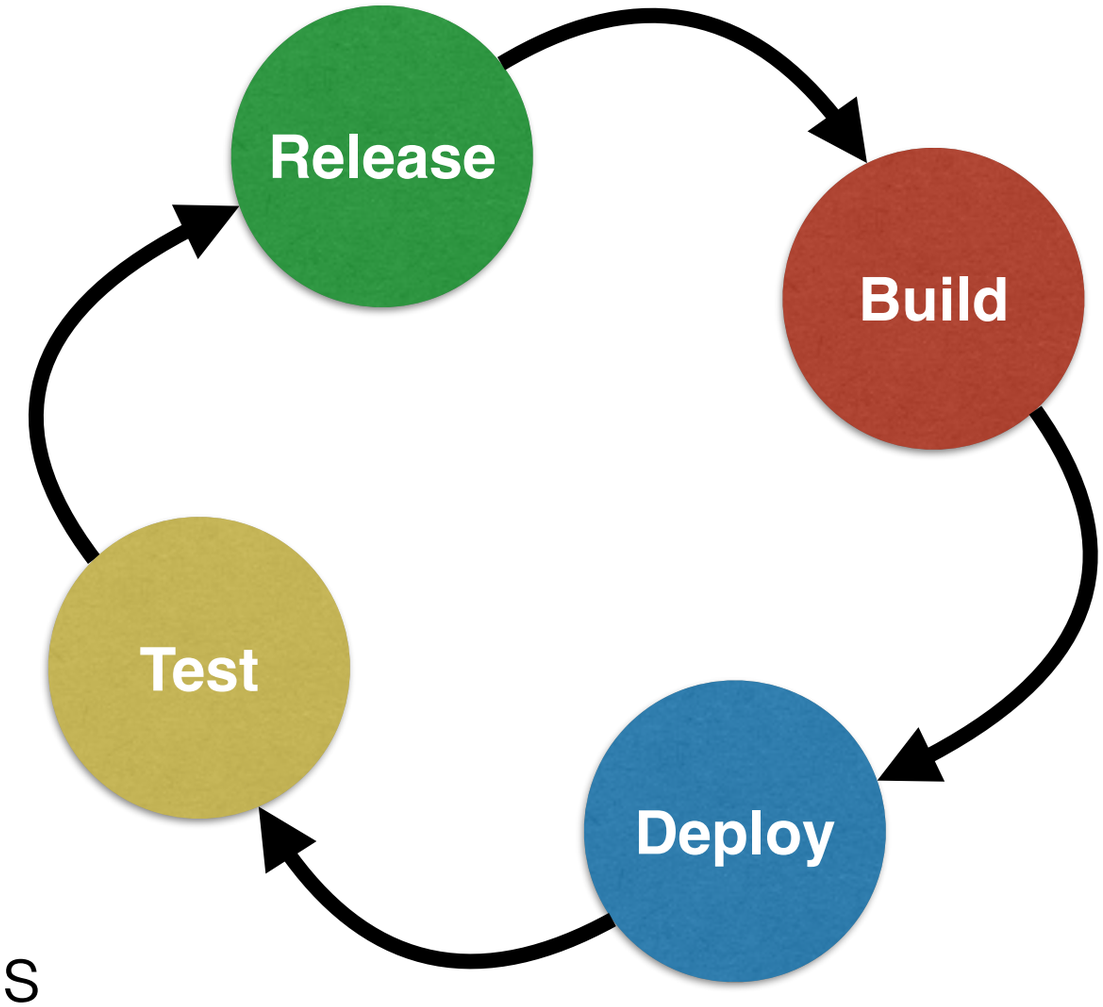
Continuous Delivery

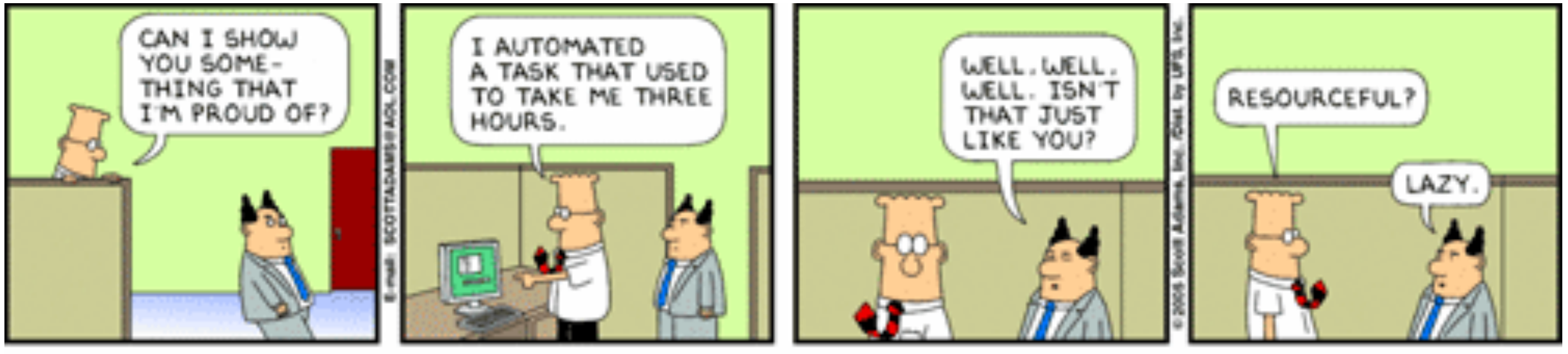
- Agile Manifesto
- Continuous Integration
- Fail fast and recover
- Self service
- 100% Automation
- Proactive monitoring and metrics



Continuous Delivery

- Agile Manifesto
- Continuous Integration
- Fail fast and recover
- Self service
- 100% Automation
- Proactive monitoring and metrics
- Push to Prod




CAN I SHOW YOU SOME-THING THAT I'M PROUD OF?
I AUTOMATED A TASK THAT USED TO TAKE ME THREE HOURS.
WELL. WELL. WELL. ISN'T THAT JUST LIKE YOU?
RESOURCEFUL?
LAZY.
DID YOU JUST TURN MY BRILLIANT ACCOMPLISHMENT INTO A CHARACTER FLAW?
COMPLAINER.
LETS JUST FORGET THE WHOLE THING.
QUITTER.

Our highest priority is to satisfy the customer through **early and continuous delivery** of valuable ~~software~~ service.

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.

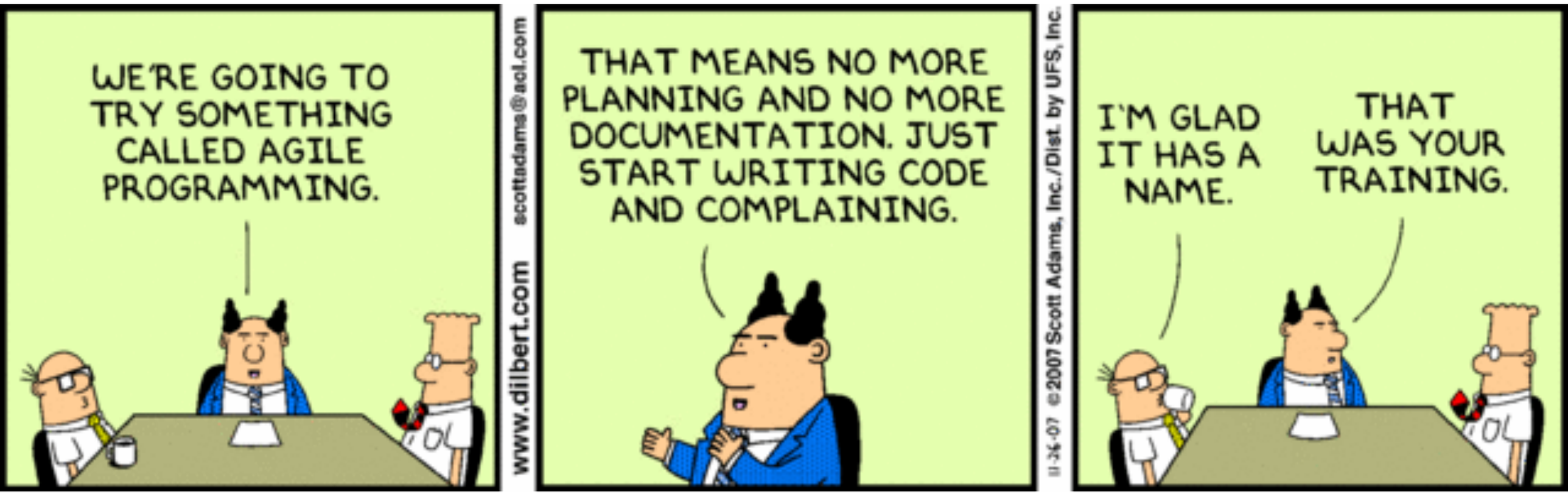
Continuous attention to technical excellence and good design enhances agility.

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.


WE'RE GOING TO INTRODUCE AGILE METHODOLOGIES
www.agileweboperations.com
WILL WE NEED TO CHANGE THE WAY WE WORK?
2012 © Matthias Marschall
I DON'T THINK SO. LET'S JUST REPLACE „PHASED APPROACH" WITH „AGILE APPROACH" IN OUR PROCESS DOCUMENTATION

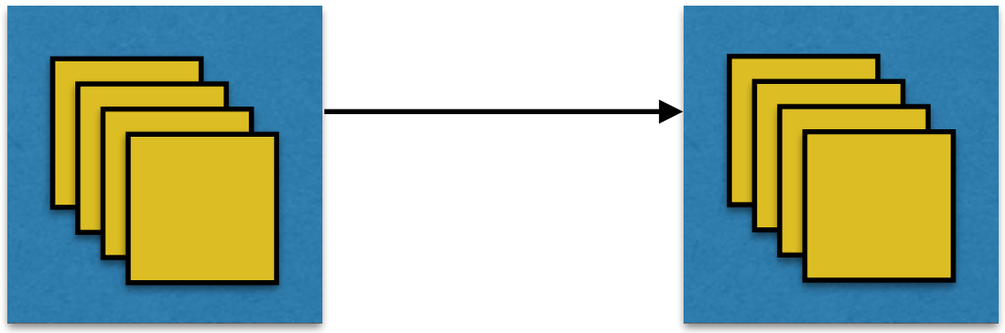

WE'RE GOING TO TRY SOMETHING CALLED AGILE PROGRAMMING.
www.dilbert.com scottadams@aol.com
THAT MEANS NO MORE PLANNING AND NO MORE DOCUMENTATION. JUST START WRITING CODE AND COMPLAINING.
I'M GLAD IT HAS A NAME.
THAT WAS YOUR TRAINING.
11-06-07 ©2007 Scott Adams, Inc./Dist. by UFS, Inc.

Tools

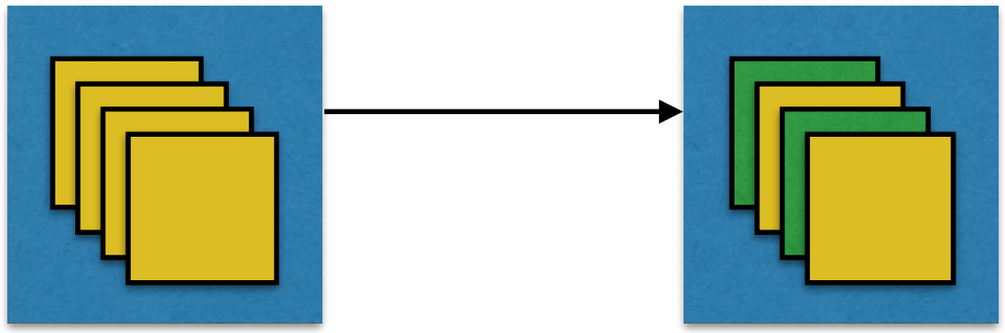
- CI: Jenkins, Travis CI, Bambo, …
- Versioning: Git, SVN, Mercurial, …
- Monitoring: Nagios, NewRelic, Pingdom, StatsD, …
- Logging: Log stash, ??

Continuous Integration - One Developer

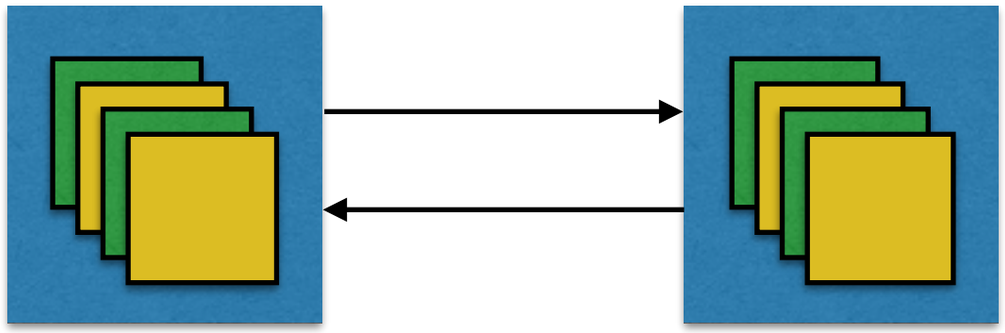
Continuous Integration - One Developer



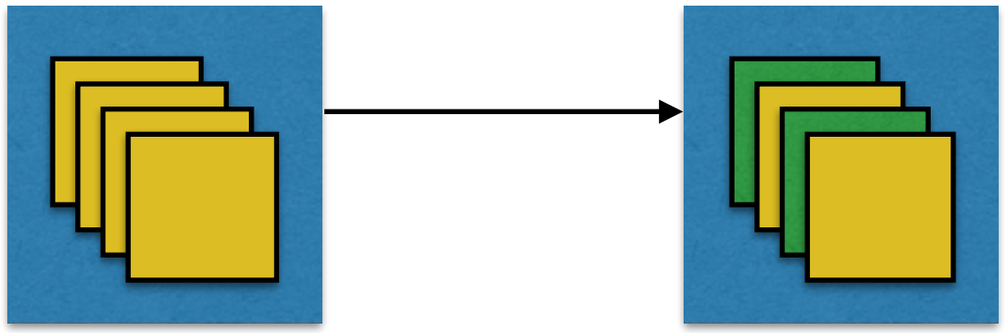
Continuous Integration - One Developer



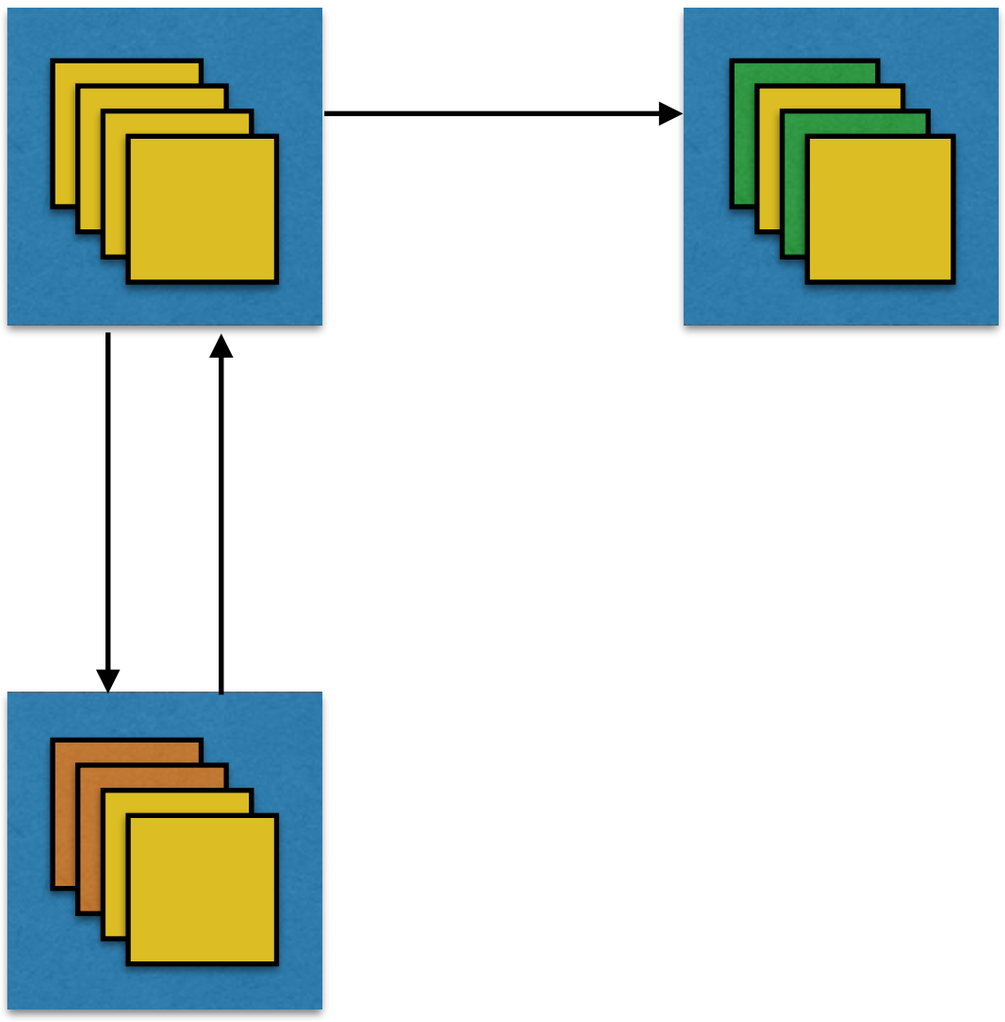
Continuous Integration - One Developer



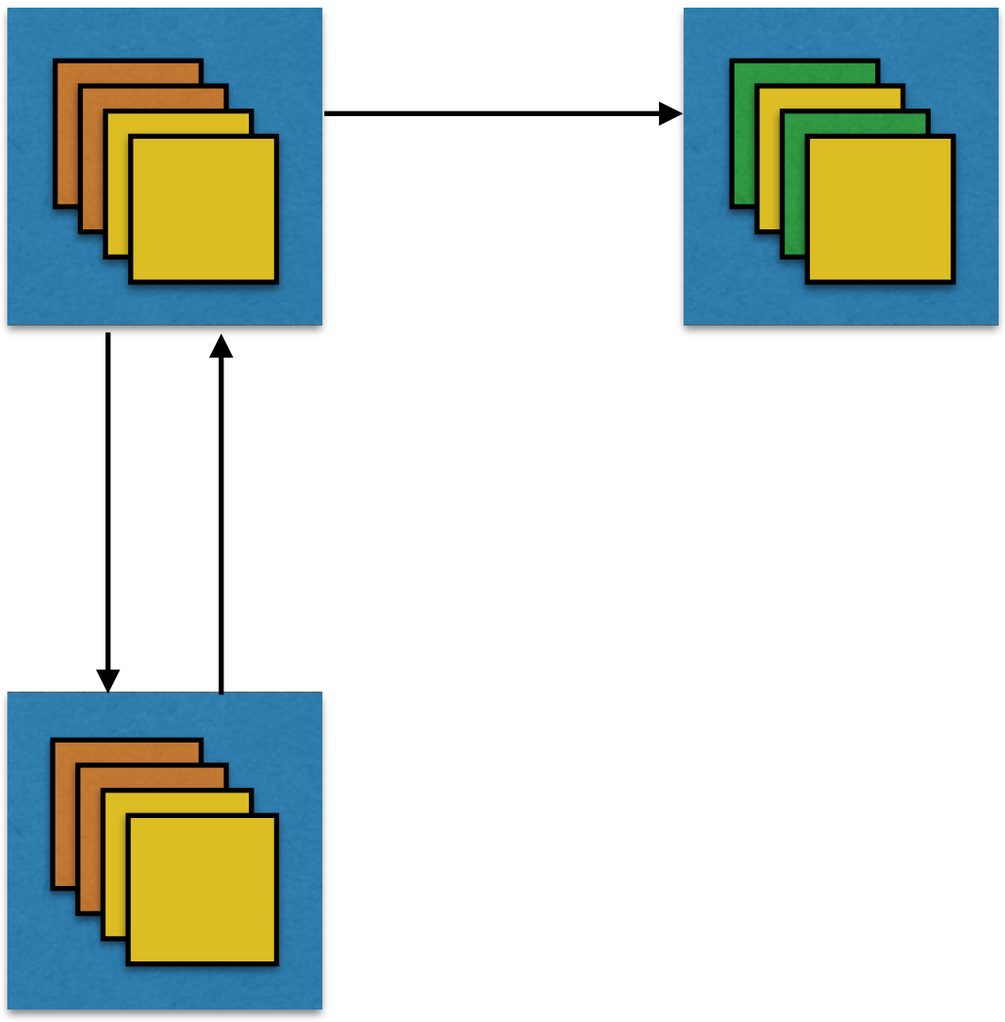
Continuous Integration - Multiple Developers



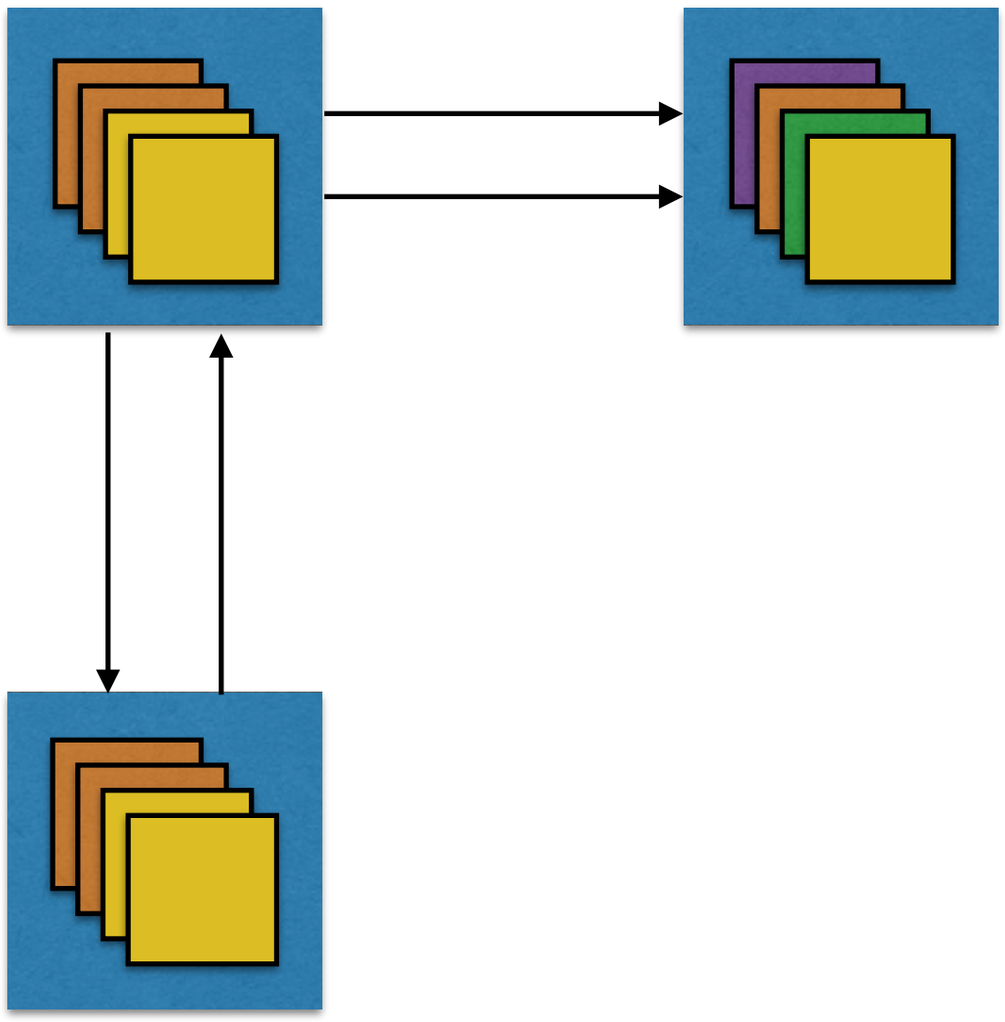
Continuous Integration - Multiple Developers



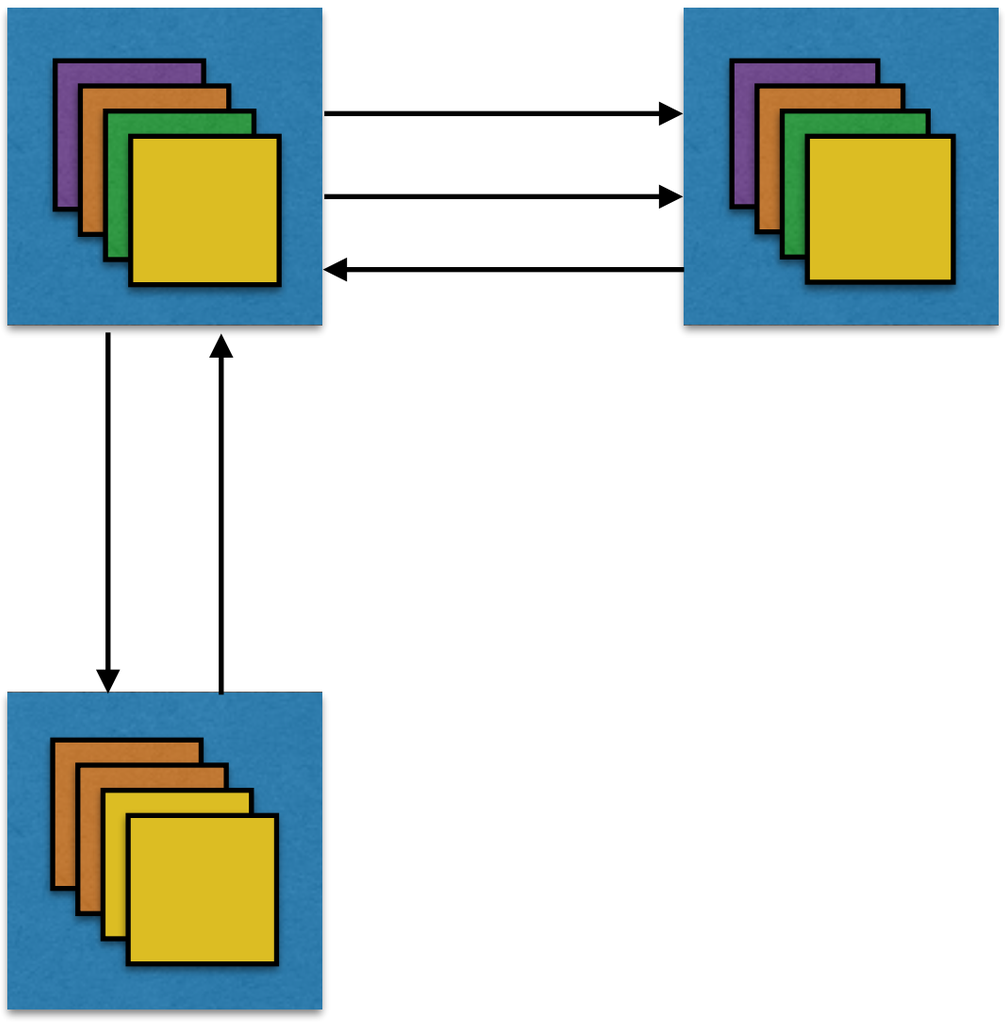
Continuous Integration - Multiple Developers



Continuous Integration - Multiple Developers



Continuous Integration - Multiple Developers



Tenets of CI

Tenets of CI

- Source code repository is the “single source of truth”

Tenets of CI

- Source code repository is the “single source of truth”
  - All source code - application and test

Tenets of CI

- Source code repository is the “single source of truth”
  - All source code - application and test
  - Include build/test scripts, schemas, IDE configurations, docs, …

Tenets of CI

- Source code repository is the “single source of truth”
  - All source code - application and test
  - Include build/test scripts, schemas, IDE configurations, docs, …
  - Tag your builds, define cadence (features?)

Tenets of CI

Tenets of CI

- Automate the build - single command builds the entire project

Tenets of CI

- Automate the build - single command builds the entire project

- Track who? when? what?

Tenets of CI

Tenets of CI

- Keep the build fast, rapid feedback (~10 mins)

Tenets of CI

- Keep the build fast, rapid feedback (~10 mins)
- Commit early and often, once every few hours, at least once a day

Tenets of CI

- Keep the build fast, rapid feedback (~10 mins)
- Commit early and often, once every few hours, at least once a day
- Check out, build, and test on a CI machine

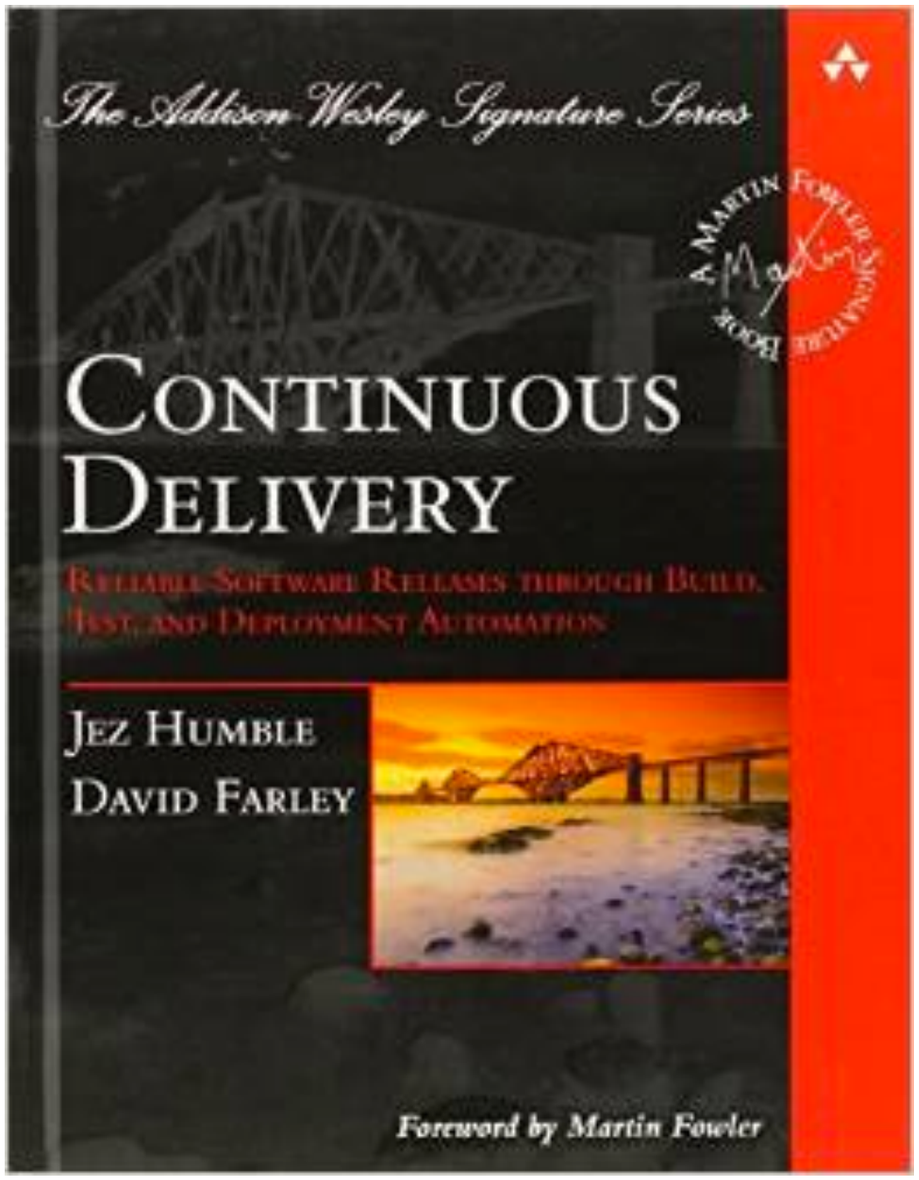
Tenets of CI

- Keep the build fast, rapid feedback (~10 mins)
- Commit early and often, once every few hours, at least once a day
- Check out, build, and test on a CI machine
- Nightly builds != CI

“Continuous Integrations doesn't get rid of bugs, but it does make them dramatically easier to find and remove”

Martin Fowler

http://martinfowler.com/articles/continuousIntegration.html

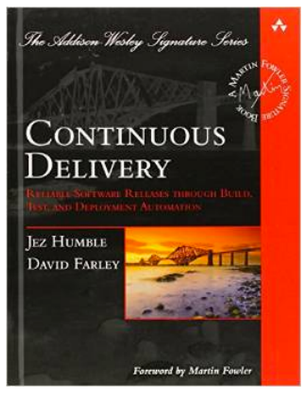


"it is the practice of releasing every good build to users"

Jez Humble

Deployment Pipeline

“an automated manifestation of your process for getting software from version control into the hands of your users"



CD Maturity Model

- Application code in source code control?
- Automated builds?
- Tests? Are they automated? TDD?
- What is the criteria for committing into “master”?
- What does it take to mature?

The Continuous Delivery Maturity Model

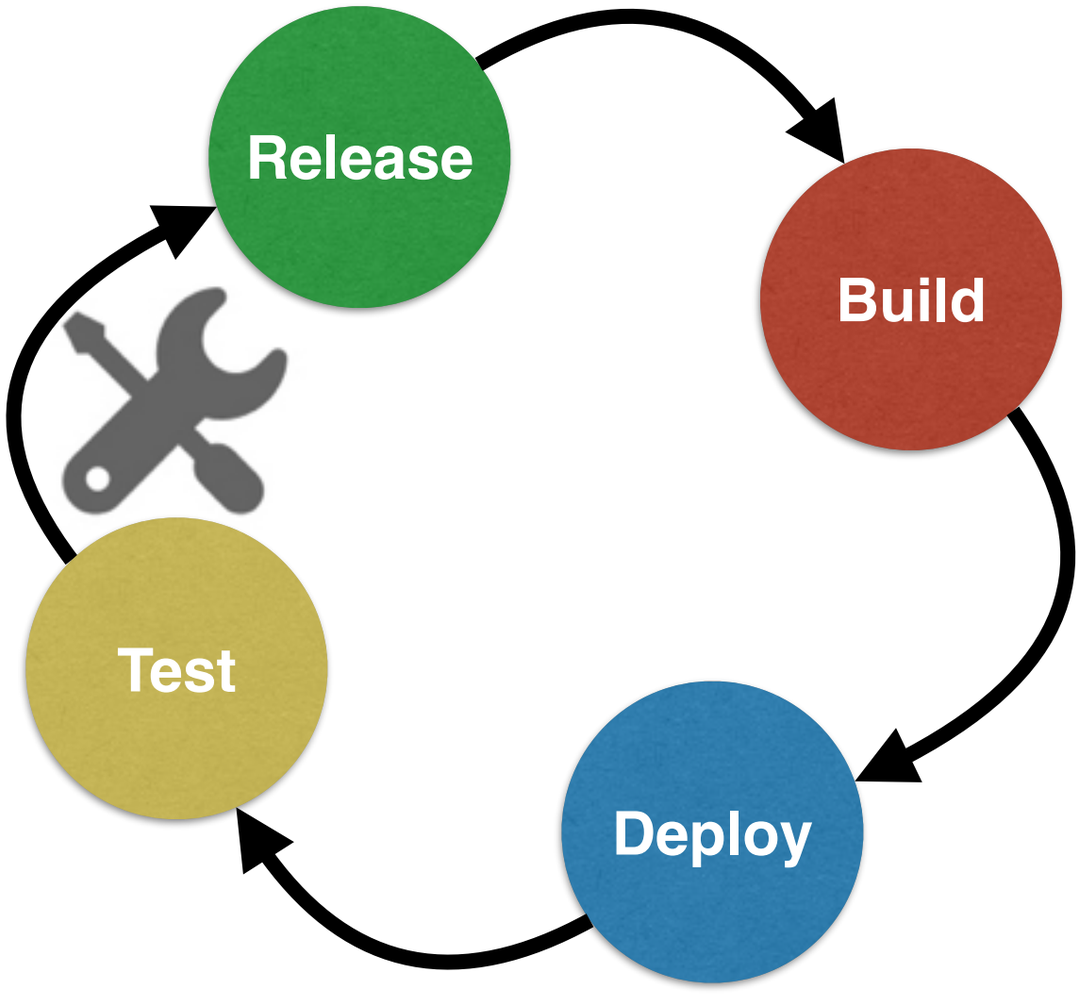
	Base	Beginner	Intermediate	Advanced	Expert
Culture & Organization	• Prioritized work • Defined and documented process • Frequent commits	• One backlog per team • Share the pain • Stable teams • Adopt basic Agile methods • Remove boundary dev & test	• Extended team collaboration • Component ownership • Act on metrics • Remove boundary dev & ops • Common process for all changes • Decentralize decisions	• Dedicated tools team • Team responsible all the way to prod • Deploy disconnected from Release • Continuous improvement (Kaizen)	• Cross functional teams • No rollbacks (always roll forward)
Design & Architecture	• Consolidated platform & technology	• Organize system into modules • API management • Library management • Version control DB changes	• No (or minimal) branching • Branch by abstraction • Configuration as code • Feature hiding • Making components out of modules	• Full component based architecture • Push business metrics	• Infrastructure as code
Build & Deploy	• Versioned code base • Scripted builds • Basic scheduled builds (CI) • Dedicated build server • Documented manual deploy • Some deployment scripts exsists	• Polling builds • Builds are stored • Manual tag & versioning • First step towards standardized deploys	• Auto triggered build (commit hooks) • Automated tag & versioning • Build once deploy anywhere • Automated bulk of DB changes • Basic pipeline with deploy to prod • Scripted config changes (e.g. app server) • Standard process for all environments	• Zero downtime deploys • Multiple build machines • Full automatic DB deploys	• Build bakery • Zero touch continuous deployments
Test & Verification	• Automatic unit tests • Separate test environment	• Automatic integration tests	• Automatic component tests (isolated) • Some automatic acceptance tests	• Full automatic acceptance tests • Automatic performance tests • Automatic security tests • Risk based manual testing	• Verify expected business value
Information & Reporting	• Baseline process metrics • Manual reporting	• Measure the process • Static code analysis • Scheduled quality reports	• Common information model • Traceability built into pipeline • Report history is available	• Graphing as a service • Dynamic test coverage analysis • Report trend analysis	• Dynamic graphing and dashboards • Cross silo analysis

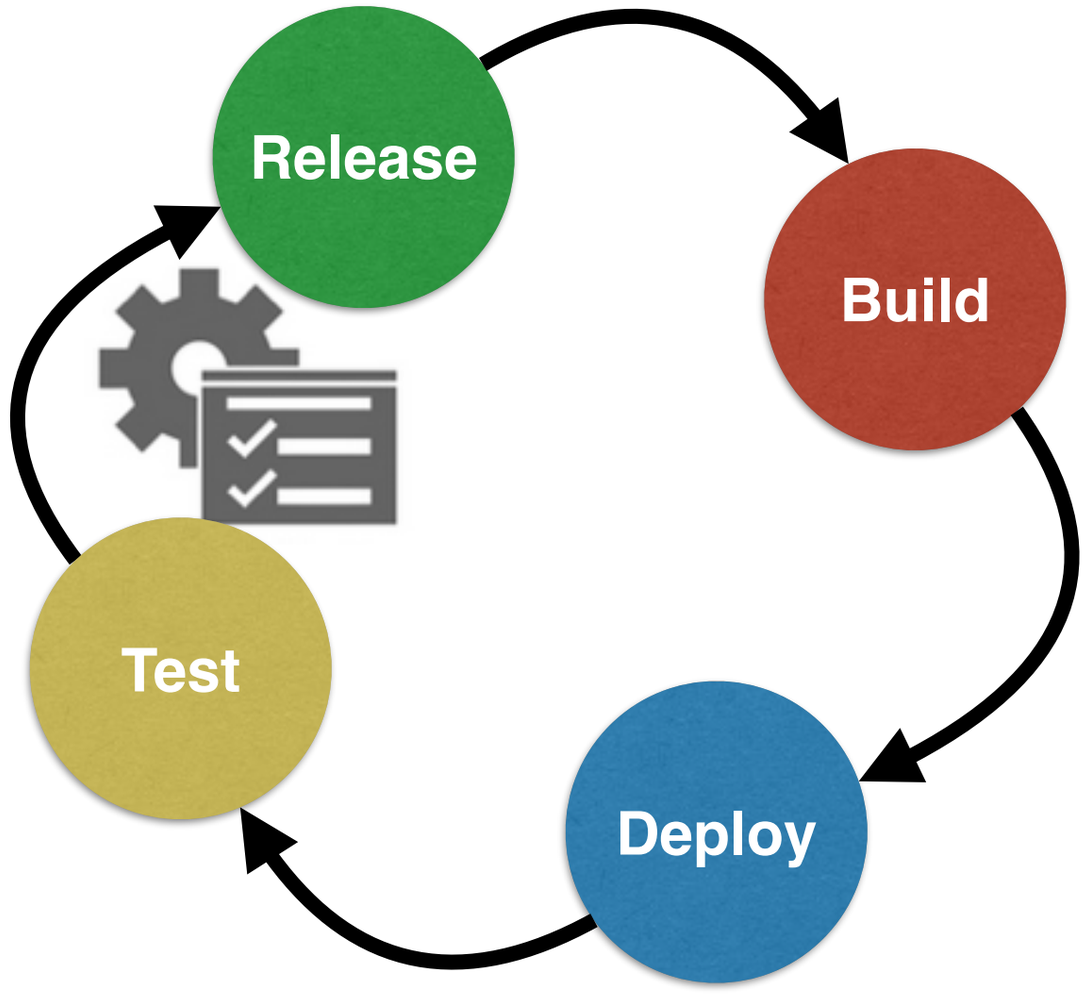
http://www.infoq.com/articles/Continuous-Delivery-Maturity-Model

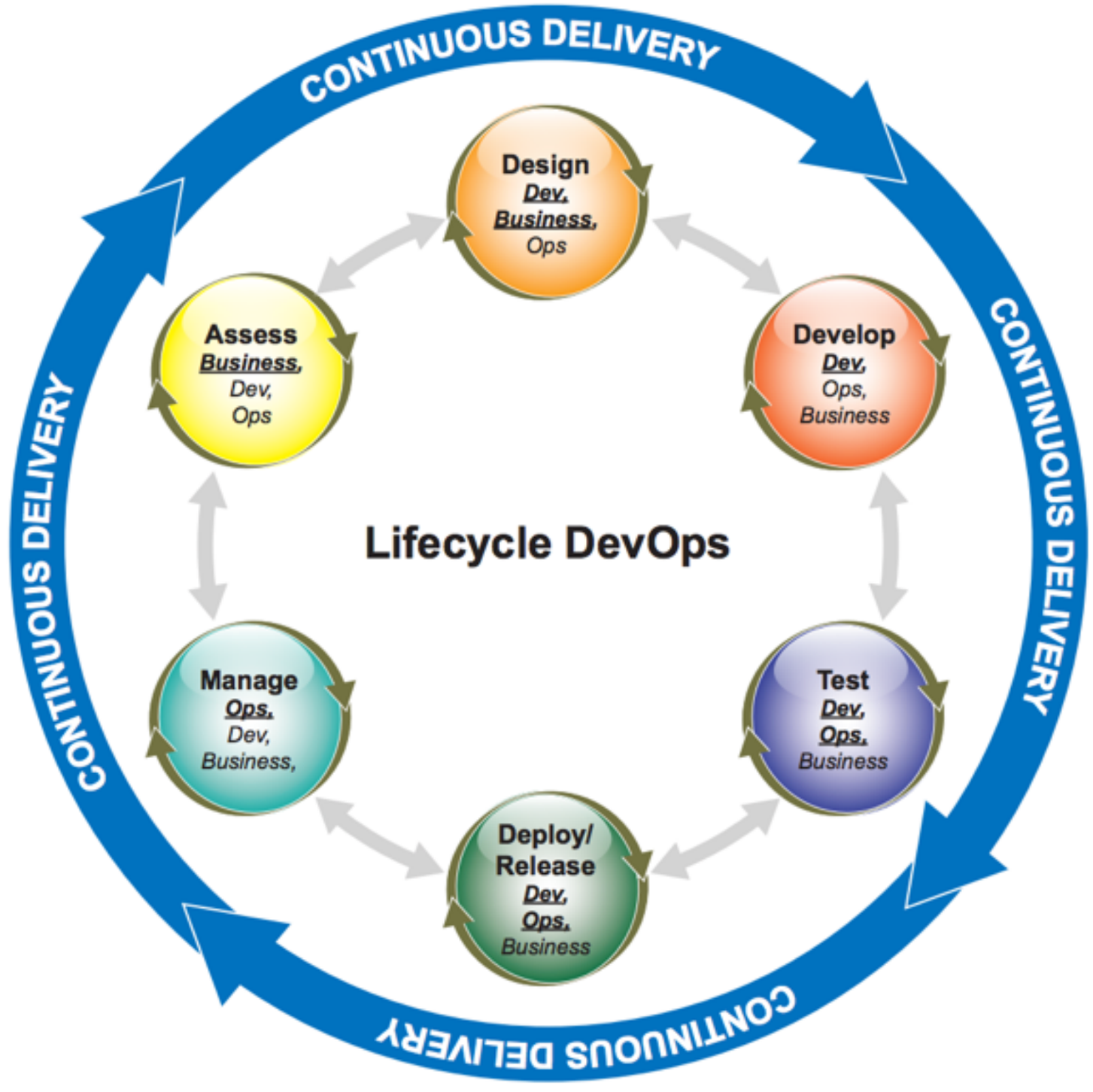
CD Maturity

Deploy Frequency ~ DevOps Maturity

Continuous Deployment







http://pages.cloudbees.com/rs/cloudbees/images/CloudBees-EMA-DevOps_2014.pdf

- 1000 releases/day
- No Chef/Puppet or QA/Release team
- 70% instances changed in 2 weeks, 90% every month
- Deployment **fully** automated

https://www.youtube.com/watch?v=WvpmiGURFYQ

NETFLIX | OSS

- **Aminator**: Create AMI from base AMI, include code + configuration (can be extended for other clouds)
- **Asgard**: Deploys images to cloud, manages resources (~Kubernetes)
- Canary analysis with red/black push
- **Glisten**: Workflow for automated deployment
- **Hysterix**: Builds resiliency, graceful fallback
- **Simian Army**: test resilience at runtime
  - Chaos, Latency, Janitor, Conformity, …

http://netflix.github.io/

http://www.allthingsdistributed.com/2014/11/apollo-amazon-deployment-engine.html
http://servicevirtualization.com/profiles/blogs/making-the-transition-to-a-devops-culture

- Switched from self-hosted data center to AWS

http://www.allthingsdistributed.com/2014/11/apollo-amazon-deployment-engine.html
http://servicevirtualization.com/profiles/blogs/making-the-transition-to-a-devops-culture

- Switched from self-hosted data center to AWS
- New update every 11.6 seconds on weekdays

http://www.allthingsdistributed.com/2014/11/apollo-amazon-deployment-engine.html
http://servicevirtualization.com/profiles/blogs/making-the-transition-to-a-devops-culture

- Switched from self-hosted data center to AWS
- New update every 11.6 seconds on weekdays
- Peak was 3x (1079 deployments in an hour)

http://www.allthingsdistributed.com/2014/11/apollo-amazon-deployment-engine.html
http://servicevirtualization.com/profiles/blogs/making-the-transition-to-a-devops-culture

- Switched from self-hosted data center to AWS
- New update every 11.6 seconds on weekdays
- Peak was 3x (1079 deployments in an hour)
- Picked up by 10k different hosts

http://www.allthingsdistributed.com/2014/11/apollo-amazon-deployment-engine.html
http://servicevirtualization.com/profiles/blogs/making-the-transition-to-a-devops-culture

- Switched from self-hosted data center to AWS
- New update every 11.6 seconds on weekdays
- Peak was 3x (1079 deployments in an hour)
- Picked up by 10k different hosts
- 0.0001% deployments resulted in outage

http://www.allthingsdistributed.com/2014/11/apollo-amazon-deployment-engine.html
http://servicevirtualization.com/profiles/blogs/making-the-transition-to-a-devops-culture

- Switched from self-hosted data center to AWS
- New update every 11.6 seconds on weekdays
  - Peak was 3x (1079 deployments in an hour)
  - Picked up by 10k different hosts
  - 0.0001% deployments resulted in outage
- Apollo - “secret sauce” for rolling updates

http://www.allthingsdistributed.com/2014/11/apollo-amazon-deployment-engine.html
http://servicevirtualization.com/profiles/blogs/making-the-transition-to-a-devops-culture

- Switched from self-hosted data center to AWS
- New update every 11.6 seconds on weekdays
  - Peak was 3x (1079 deployments in an hour)
  - Picked up by 10k different hosts
  - 0.0001% deployments resulted in outage
- Apollo - “secret sauce” for rolling updates
  - aws.amazon.com/codedeploy/

http://www.allthingsdistributed.com/2014/11/apollo-amazon-deployment-engine.html
http://servicevirtualization.com/profiles/blogs/making-the-transition-to-a-devops-culture

Graphite · cacti · New Relic. · Chartbeat · Nagios

node · python · php · Java · Ruby

ASGARD · amazon web services · ubuntu · puppet labs

Scott VanDenPlas
@scottvdp
Follow

4Gb/s, 10k req/s, 2k nodes, 3 datacenters, 180TB and 8.5 billion req. Design, deploy, dismantle in 583 days to elect the President. #madops

http://arstechnica.com/information-technology/2012/11/how-team-obamas-tech-efficiency-left-romney-it-in-dust/

Add RHAT portfolio

References

- github.com/arun-gupta/devops