

DevOps

Arun Gupta, @arungupta
Red Hat

Delivering Software: Reality vs Goal



Is DevOps for you?

Ship code 30x faster

and complete those deployments 8,000 times faster than their peers.

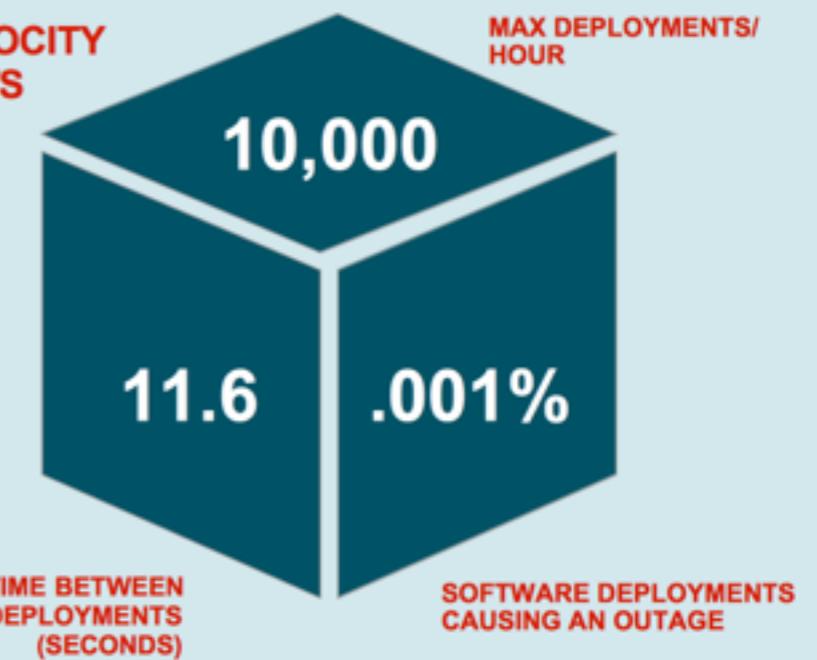
Have 50% fewer failures

and restore service 12 times faster than their peers.

Organizations implementing DevOps



DEVOPS VALUE
IN ACTION: VELOCITY
AT AMAZON AWS



Organizations implementing DevOps

Numbers missing?

Oct
2014



Oct
2014

BUSINESS BENEFITS experienced through DevOps

57%

INCREASED customer conversion or satisfaction

57%

REDUCED IT Infrastructure spend

49%

REDUCTION in application downtime or failure rates

46%

INCREASE in customer engagement

46%

INCREASE in sales

32%

INCREASE in employee engagement

2%

It's too early to say

3%

No, the wider business has not seen measurable benefits from DevOps



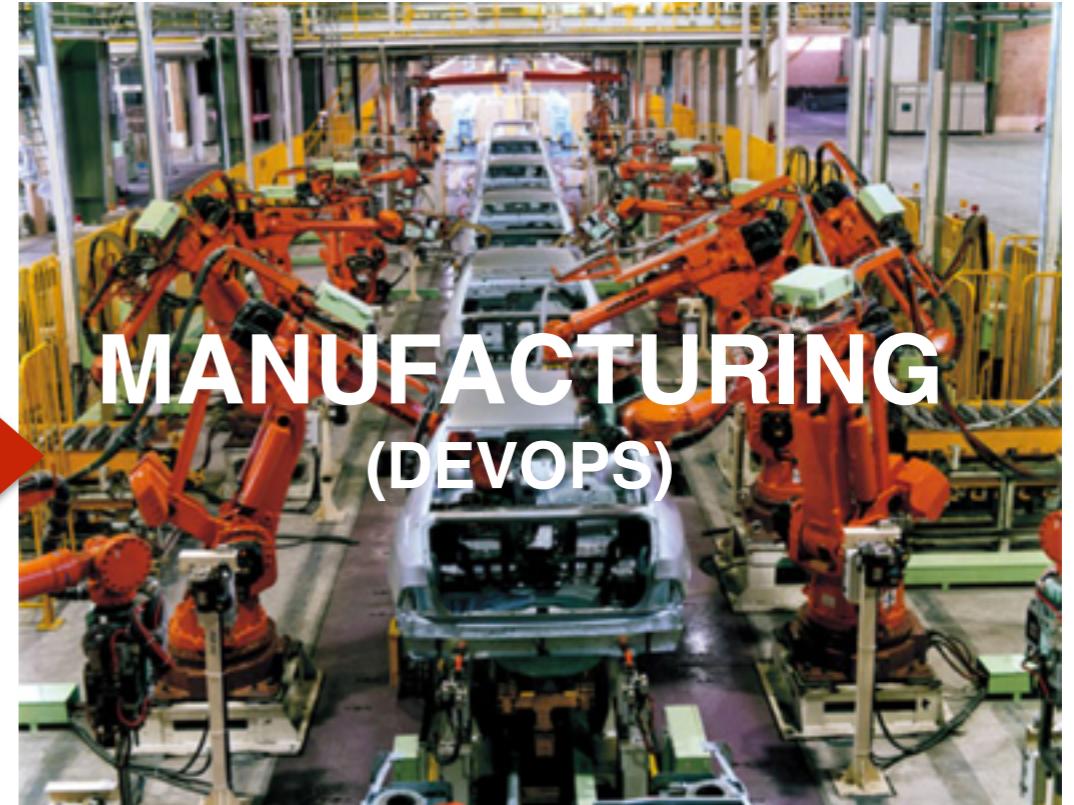
CRAFTWORK



WORKSHOP



CRAFTWORK



MANUFACTURING
(DEVOPS)

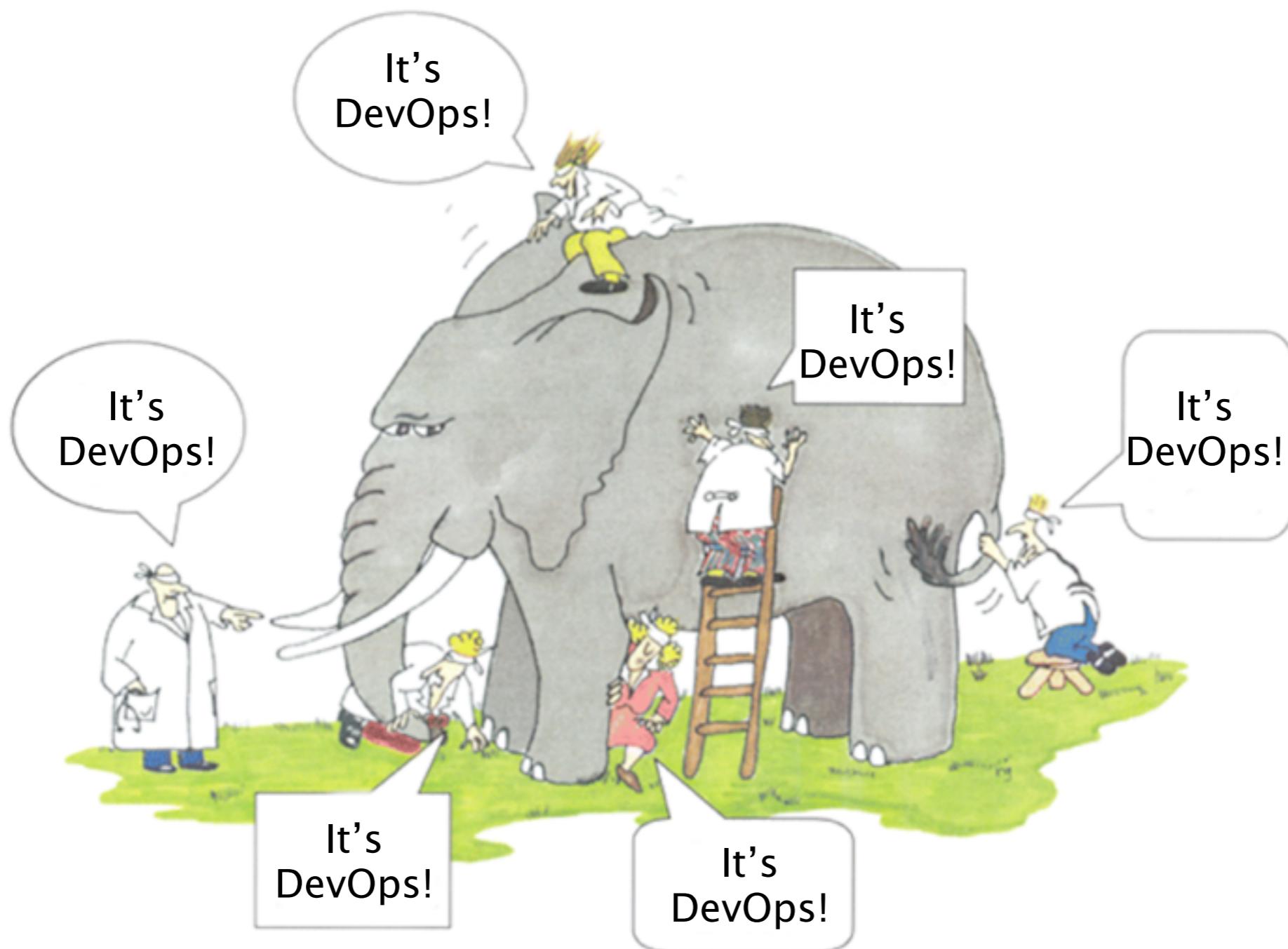


WORKSHOP



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

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

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

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

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

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

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

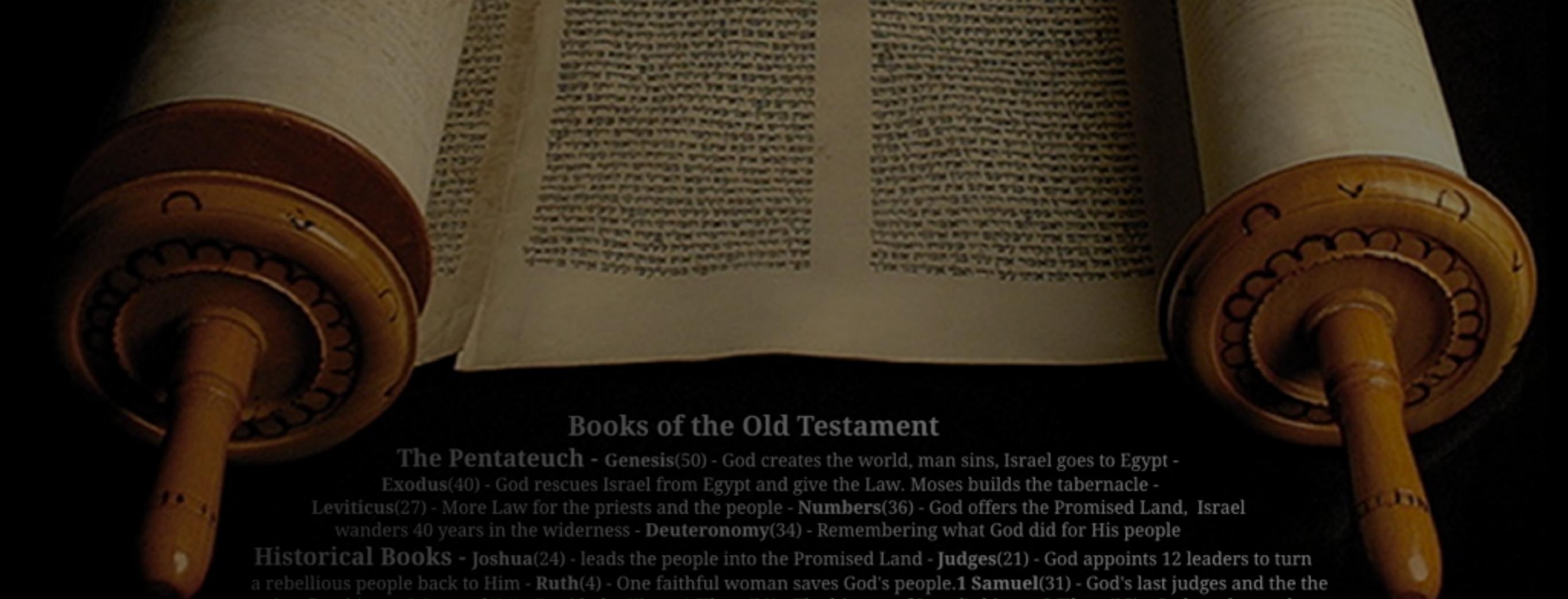
Collaboration

- “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



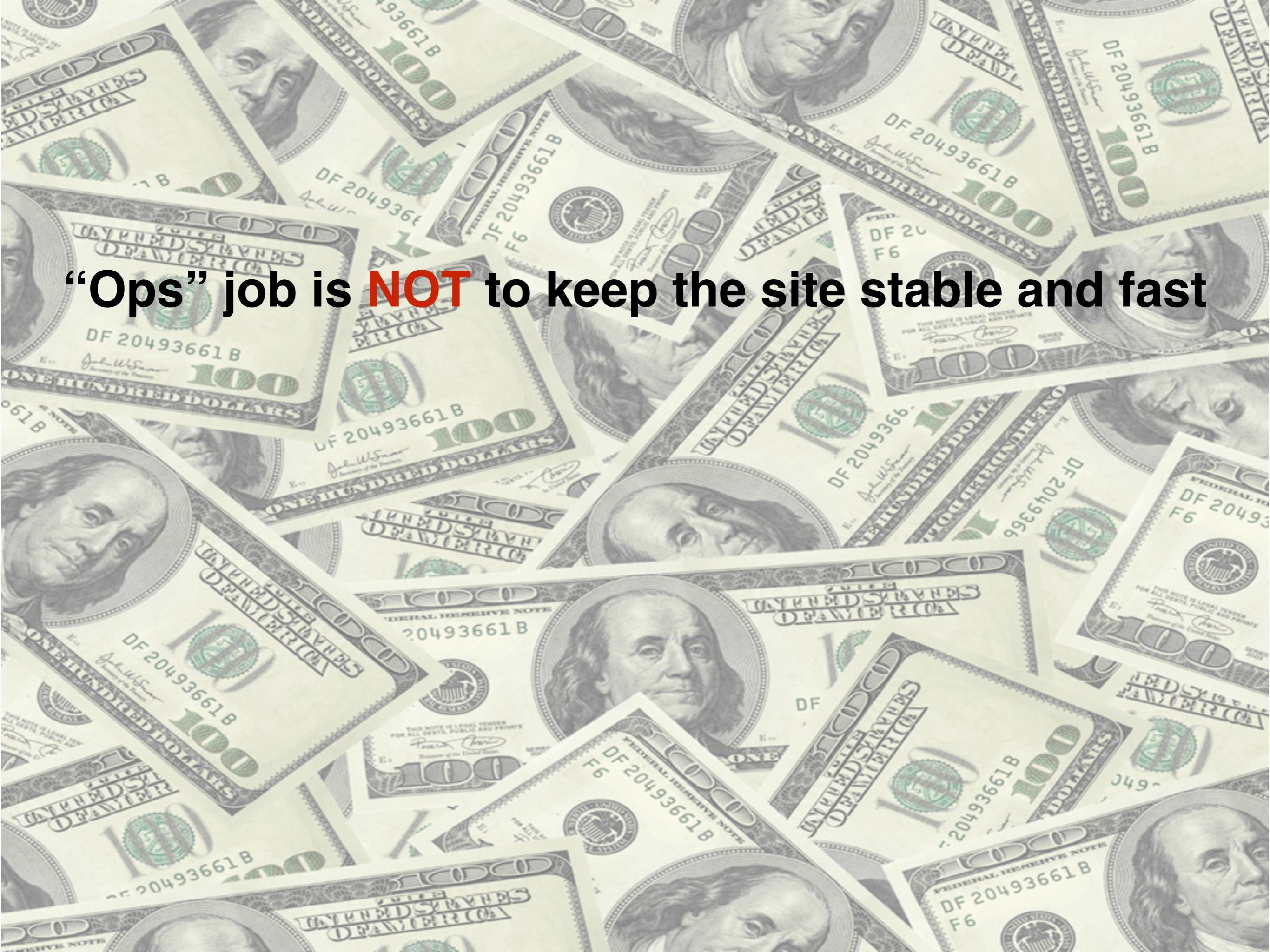


Books of the Old Testament

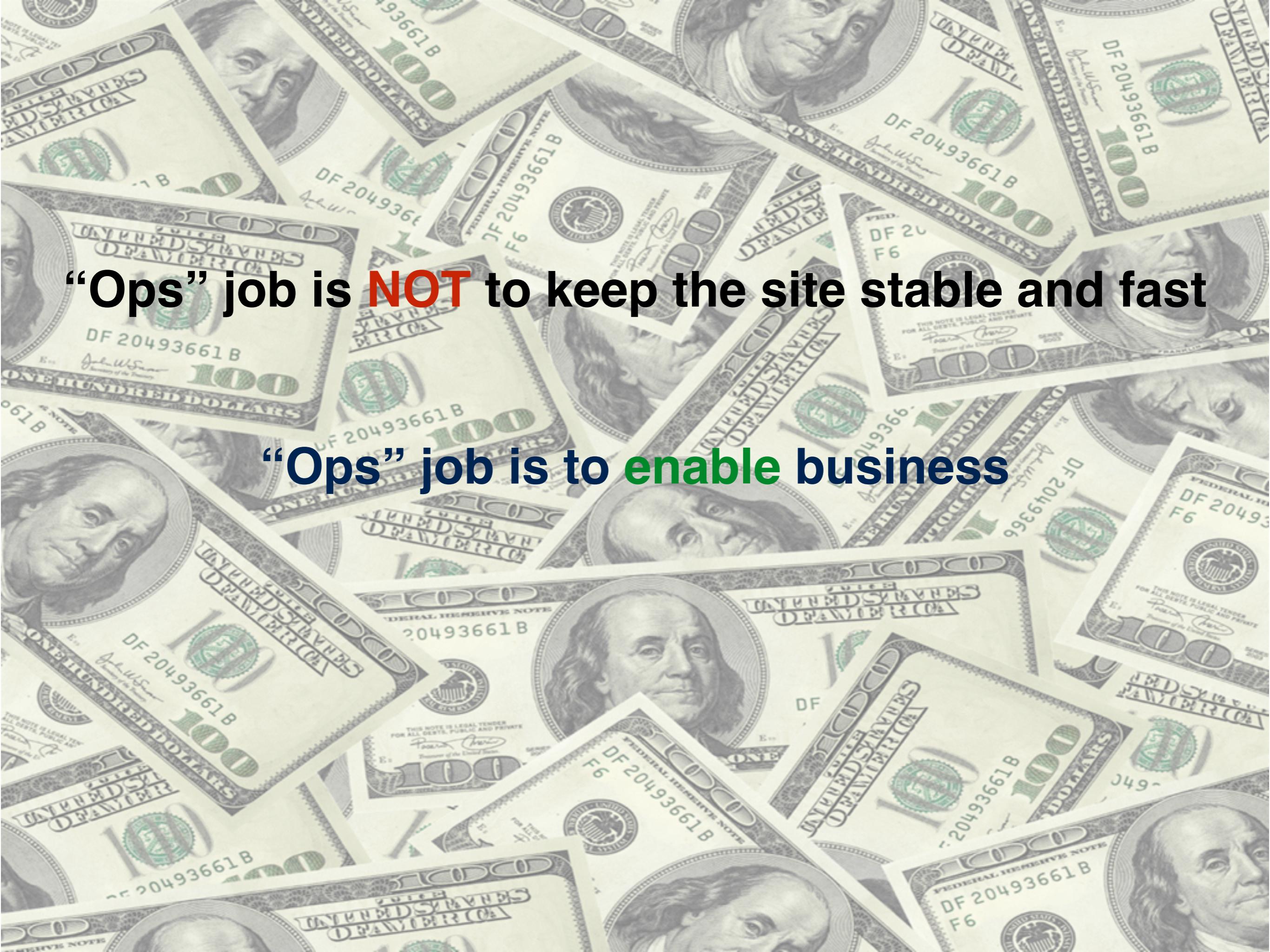
The Pentateuch - Genesis(50) - God creates the world, man sins, Israel goes to Egypt - Exodus(40) - God rescues Israel from Egypt and give the Law. Moses builds the tabernacle - Leviticus(27) - More Law for the priests and the people - **Numbers**(36) - God offers the Promised Land, Israel wanders 40 years in the wilderness - **Deuteronomy**(34) - Remembering what God did for His people

Historical Books - Joshua(24) - leads the people into the Promised Land - Judges(21) - God appoints 12 leaders to turn a rebellious people back to Him - Ruth(4) - One faithful woman saves God's people. 1 Samuel(31) - God's last judges and the the people's first kings - 2 Samuel(24) - David, the King - 1 Kings(22) - The history of Israel's kings - 2 Kings(25) - God sends prophets to the kings - 1 Chronicles(29) - David's family tree - 2 Chronicles(36) - More leasons from the lives of the kings of Israel - Ezra(10) - God keeps his promise and restores Israel to the Promised Land - Nehemiah(13) - The last history. God uses one man to restore Jeruselem as the people return - Esther(10) - Faith and love of two women leads to a kinsman redeamer - **Poetical Books** - Job(42) - God is glorified in even the worst of situations - Psalms(150) - God is praised in poetry and song - Proverbs(31) - God is the source for wisdom for each day - Ecclesiastes(12) - According to the Teacher, Live is meaningless without God - Song of Solomon(8) - A husband and wife are made for each other, body and soul - **Major Prophets** - Isaiah(66) - The first prophetic book foretells the life and death of the coming Messiah. - Jeremiah(52) - warns people to repent of their sins and ask God's forgiveness even when they don't listen - Lamentations(5) - God is saddened over our sin and our choice of self-distruction - Ezekiel(48) - Called as prophet and priest during the Babylon exile. Daniel(12) - Even in the lion's den, God can be served - **Minor Prophets** - Hosea(14) - Prophet of Divine Love. Redeemed an unfaithful bride. - Joel(3) - Prophet of Pentecost. Warned of God's direct retribution against the sinful. - Amos(9) - A voice crying in the wilderness, calling northern Israel to repentance. - Obadiah(1) - Phophesised against the descendants of Esau who opposed God's people. - Jonah(4) - tries to avoid God's call, spends three days in the belly of a fish - Micah(7) - Another voice crying in the wilderness, calling southern Judah to repentance. - Nahum(3) - Predicts the fall of Jonah's Nineveh. - Habakkuk(3) - A dialog between the prophet and God from questioning to faith. - Zephaniah(3) - Judgment and salvation, extending to all nations. - Haggai(2) - Exorts the people to return to the mission of rebuilding the House of God - Zechariah(14) - Two books in one, the rebuilding of the temple then prophecy of the Messiah - Malachi(4) - Addresses the apostasy of Israel and the coming "messenger of the covenant"

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

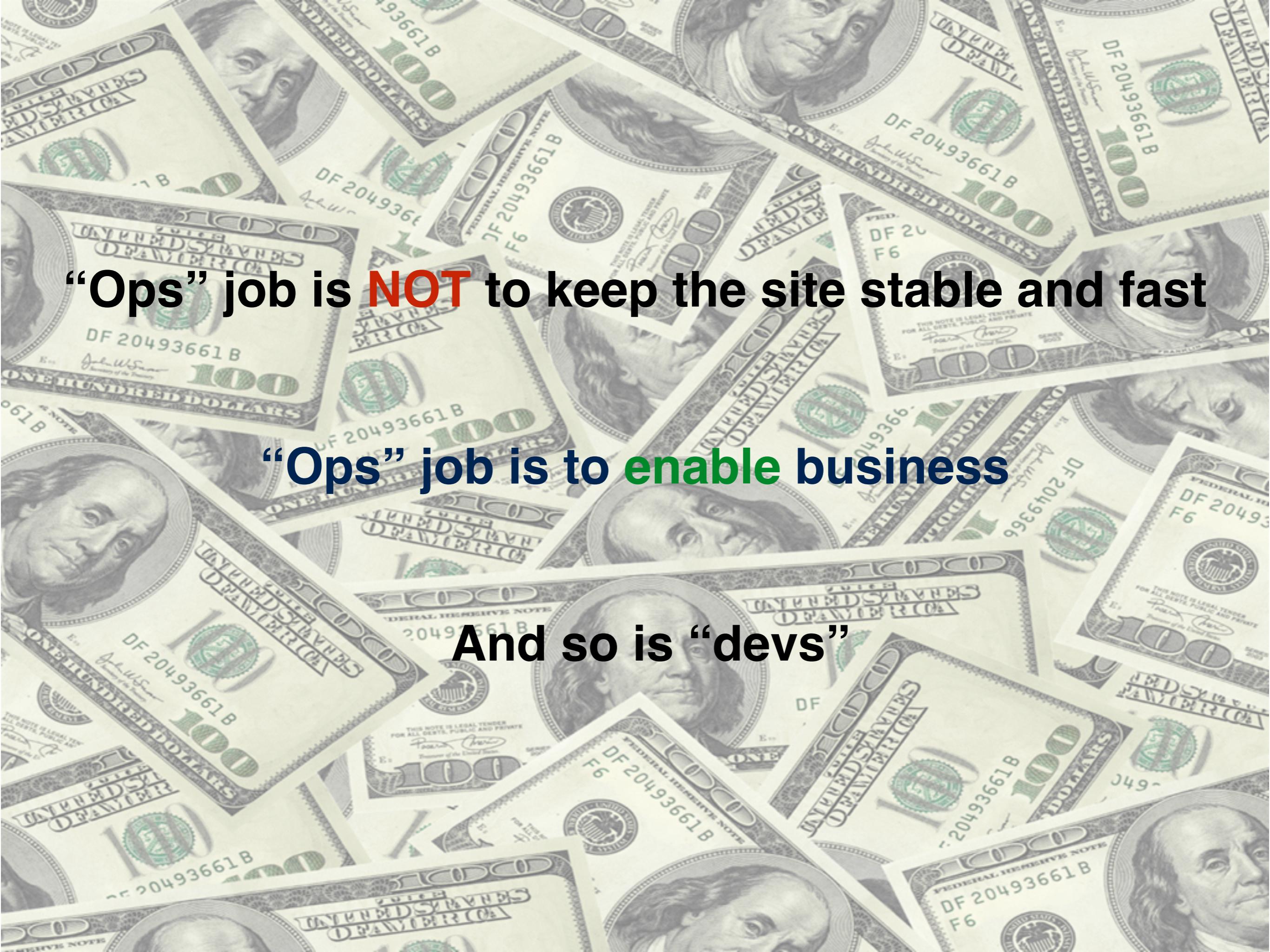


“Ops” job is NOT to keep the site stable and fast



“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

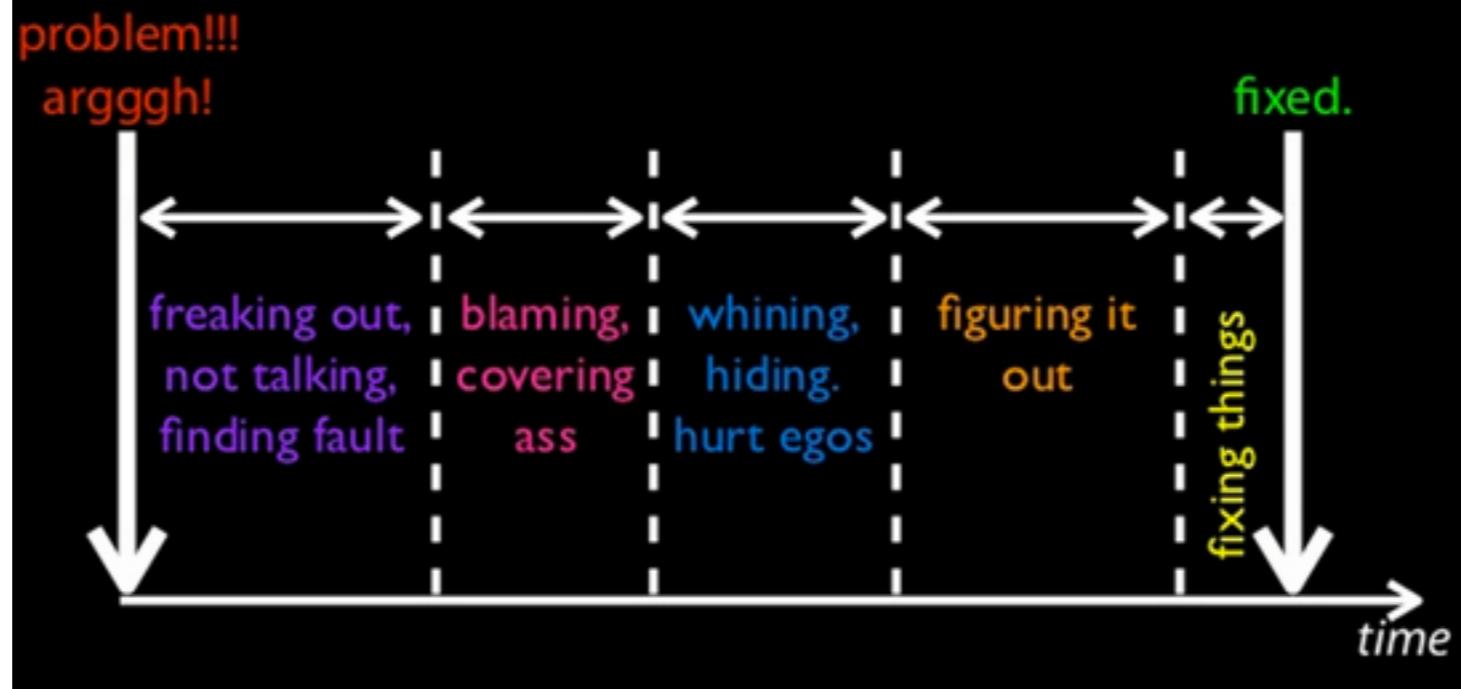
Culture

- 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

Fingerpointyness

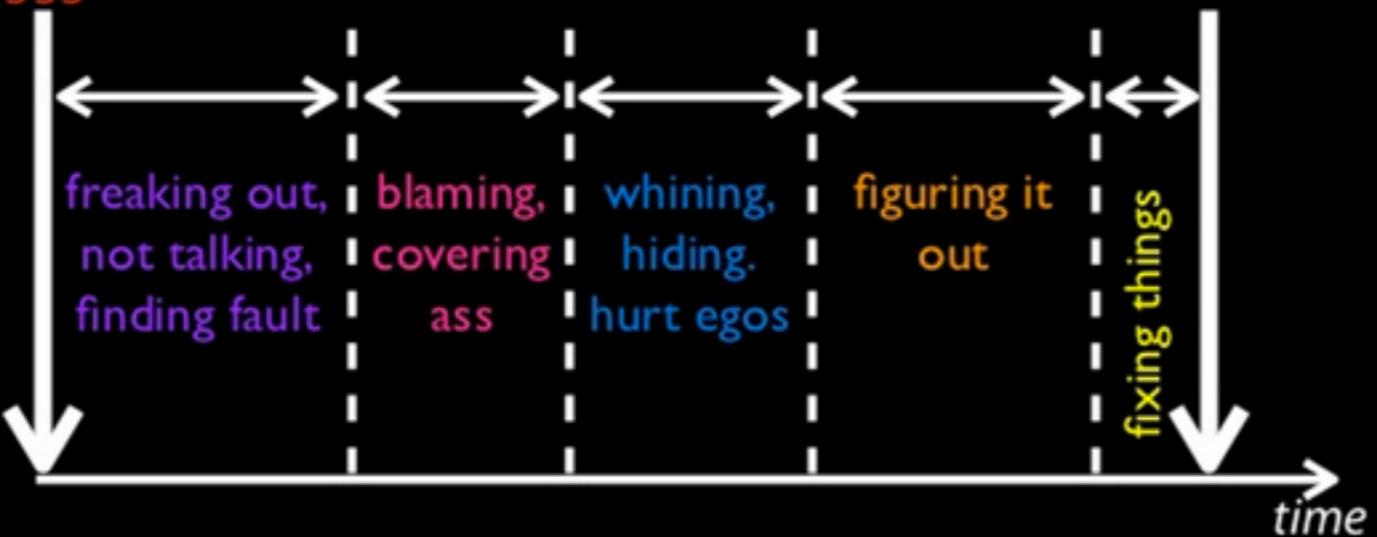


Fingerpointyness

problem!!!

argggh!

fixed.

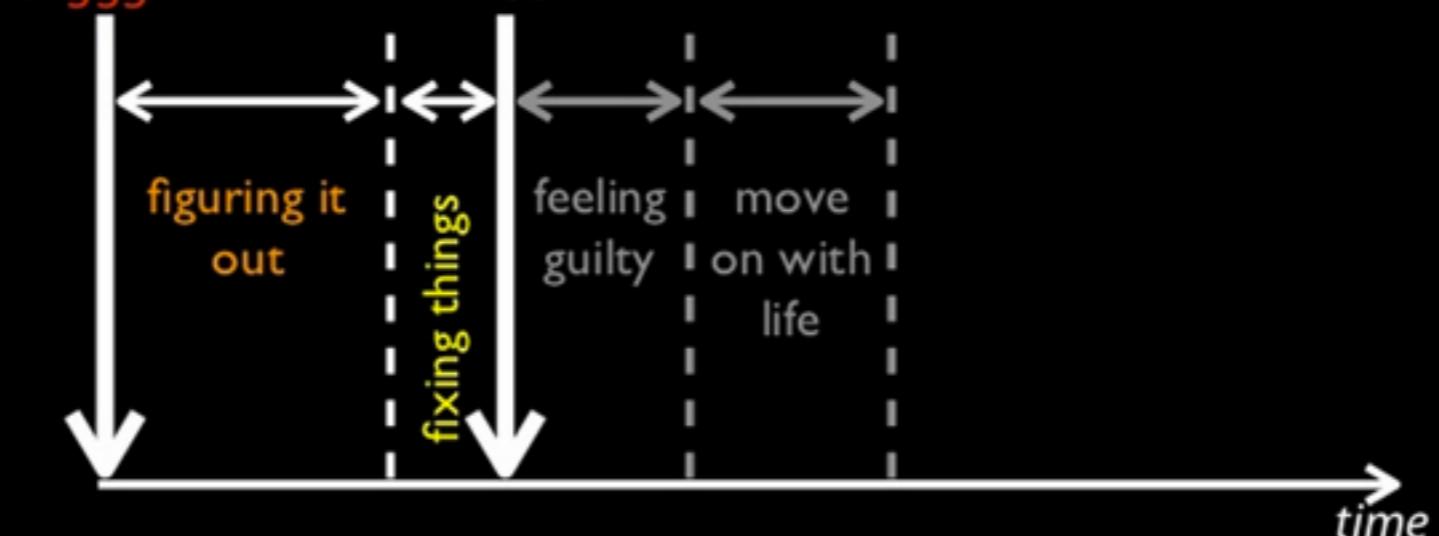


Being productive

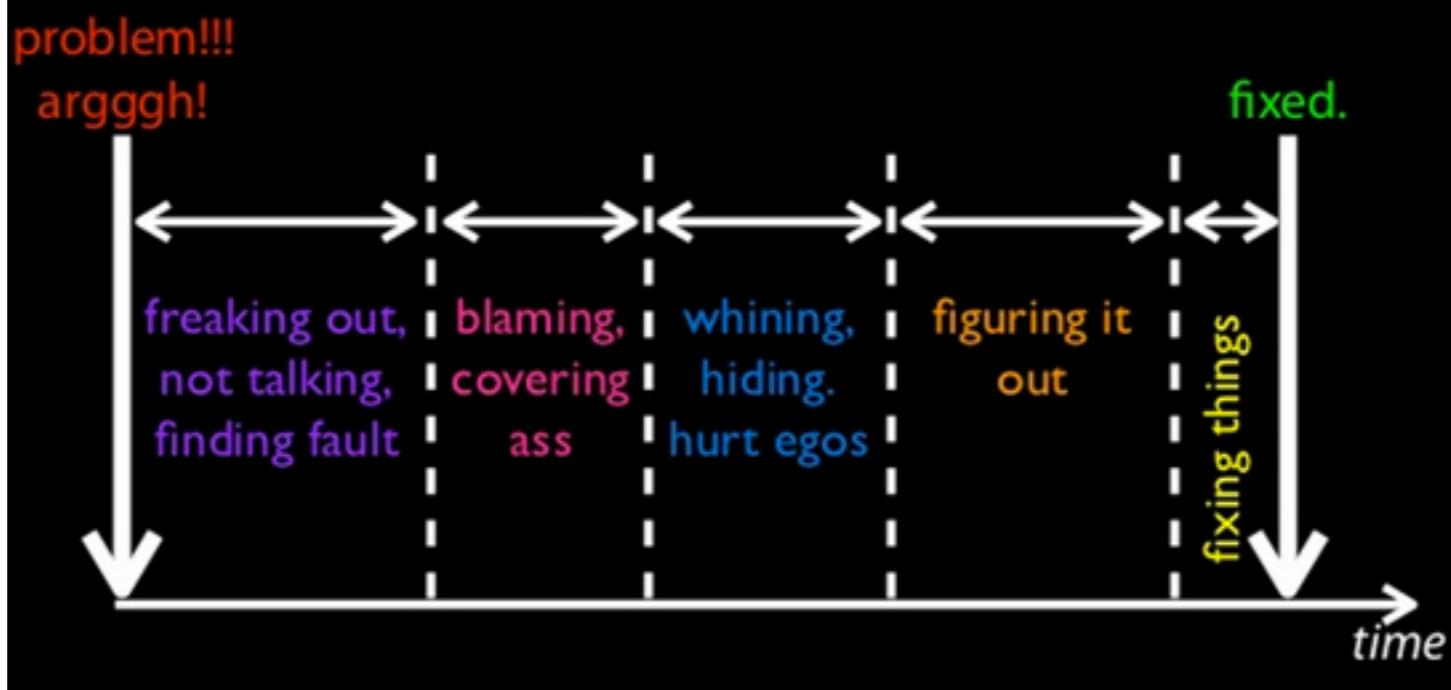
problem!!!

argggh!

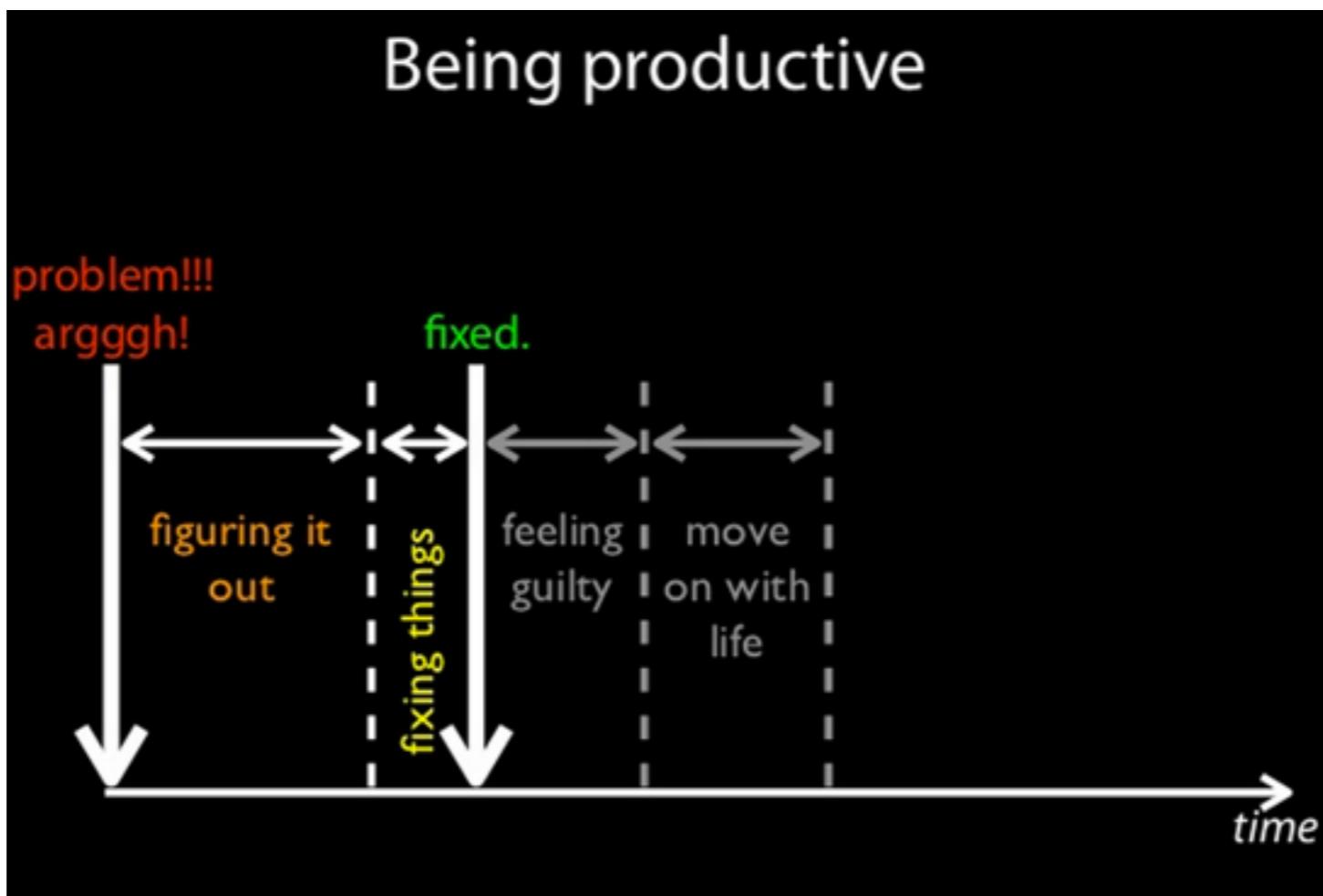
fixed.



Fingerpointyness



Being productive



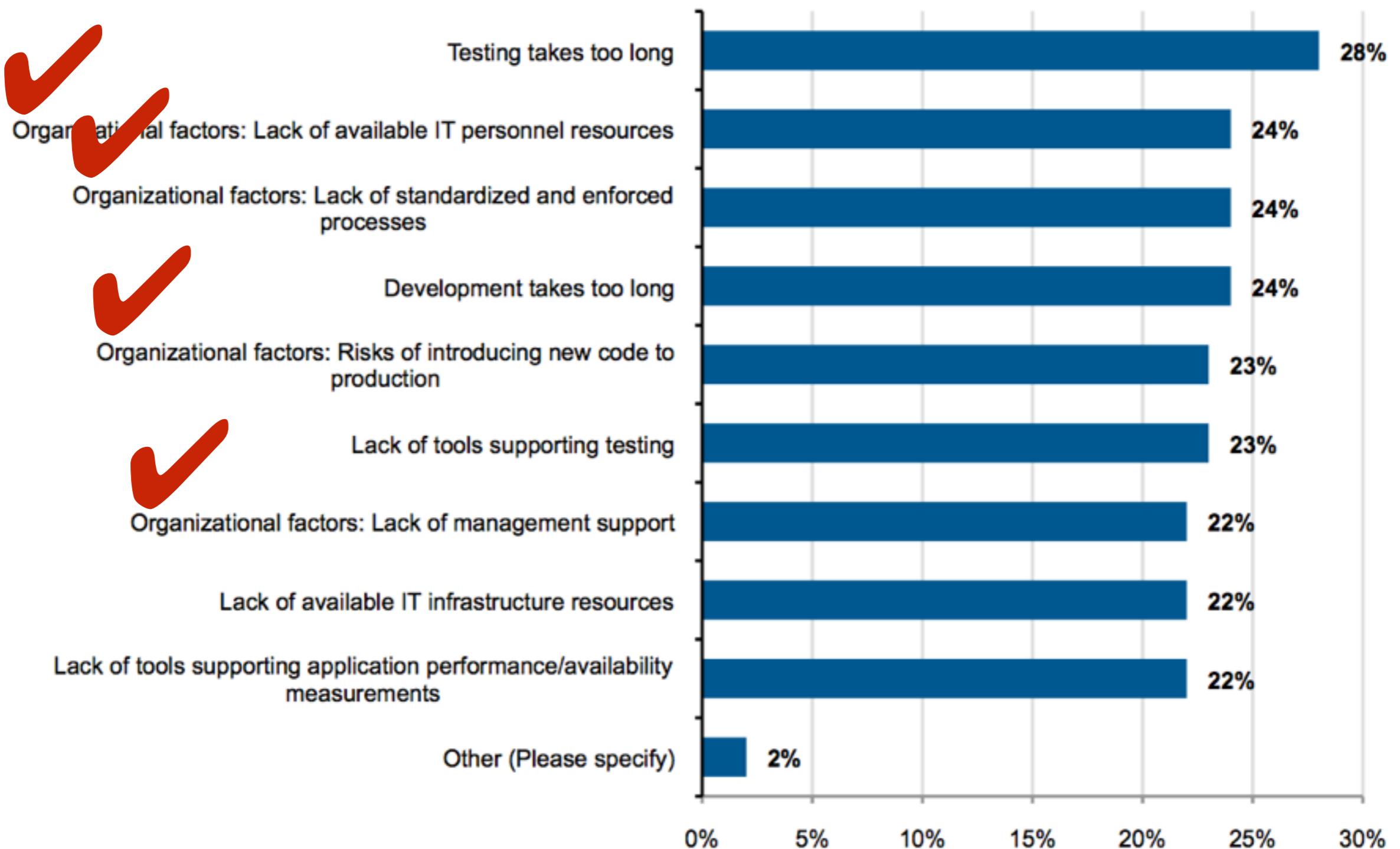
“Treat people warmly, issues coldly!”



With great
power, comes great
responsibility

“you build it, you run it!”

In your opinion, what are the top THREE (3) concerns/challenges impacting your organization's ability to accelerate Continuous Delivery?



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”

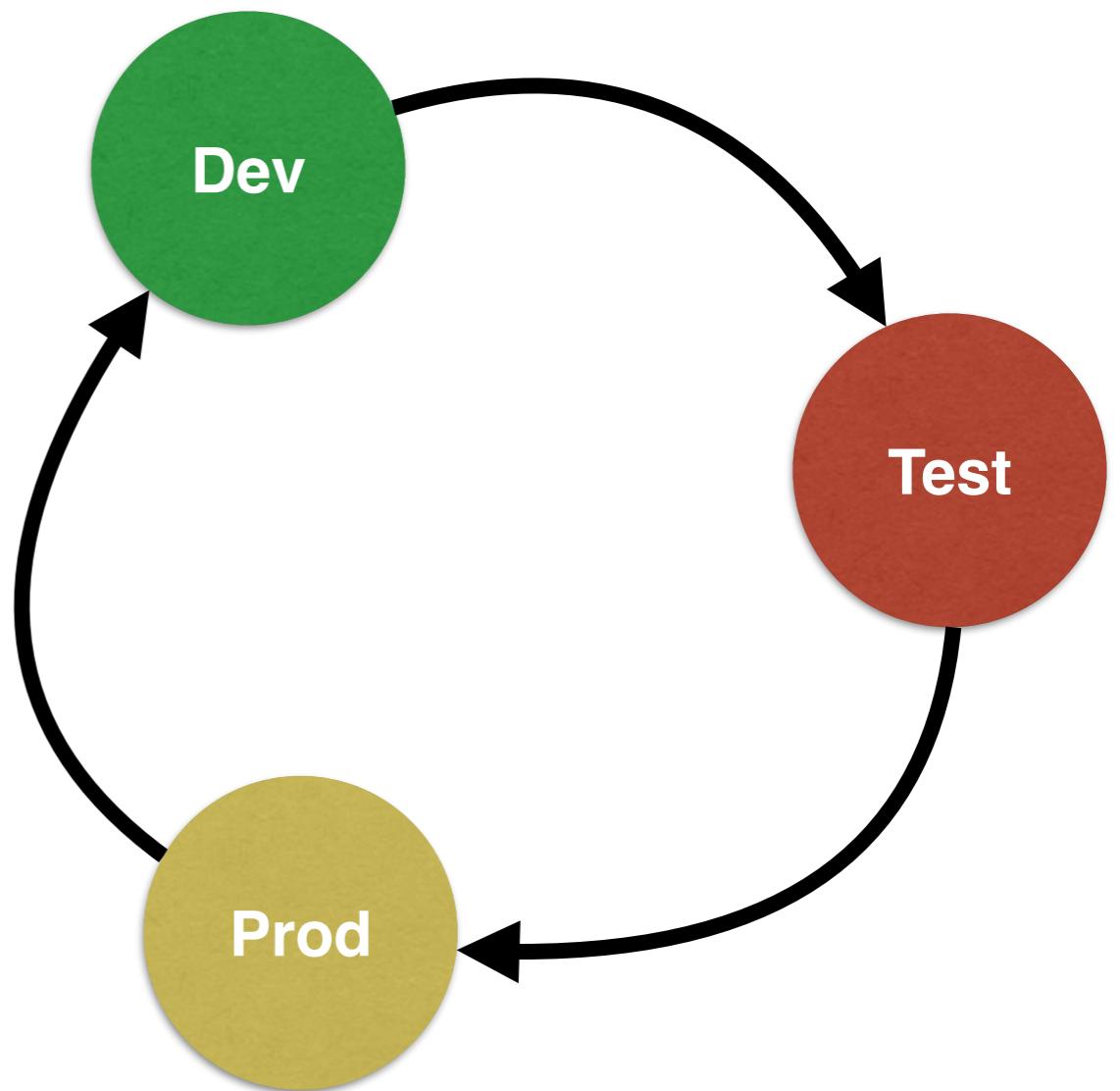
Consistency

Consistency

- Automation over documentation

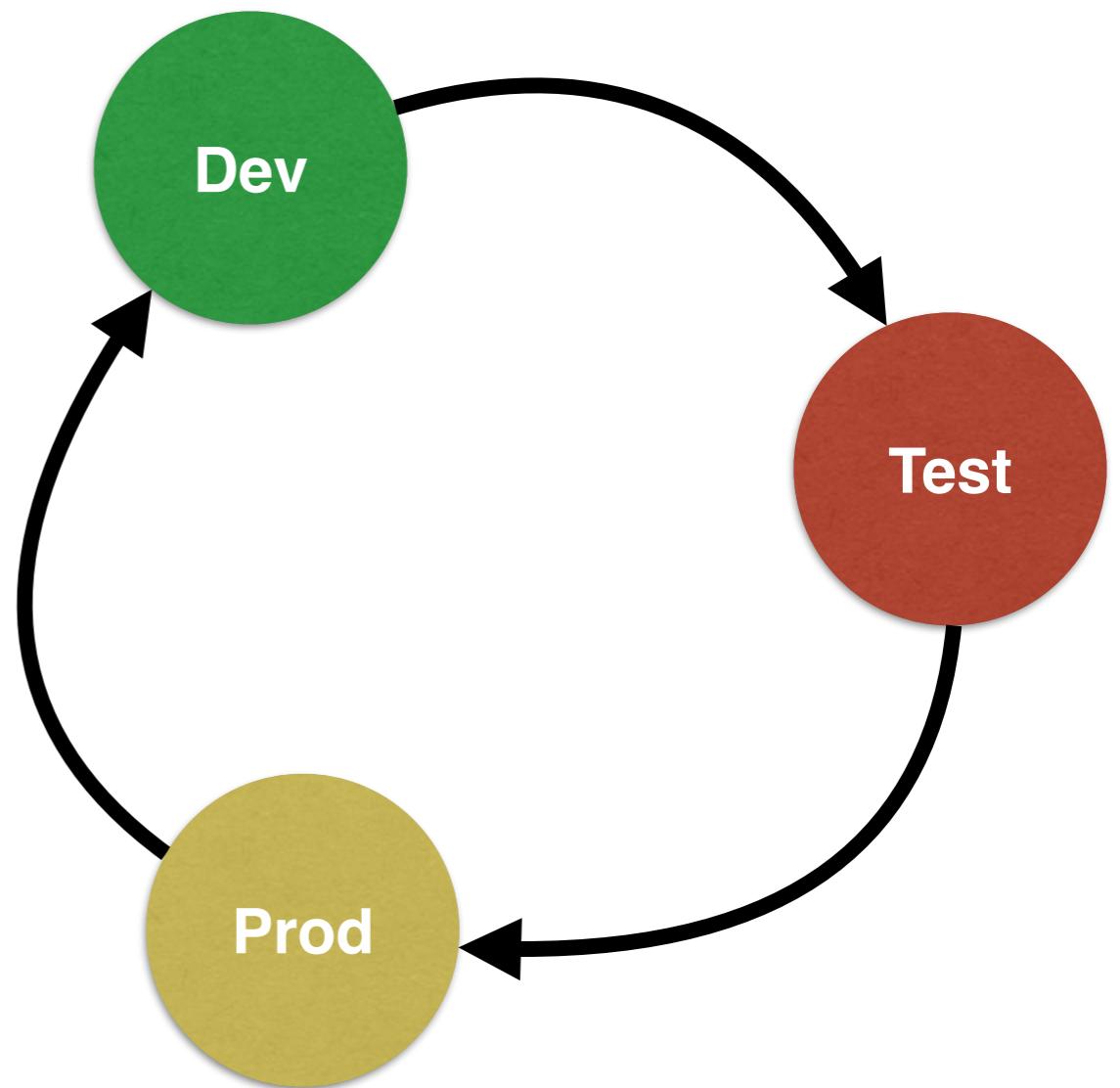
Consistency

- Automation over documentation
 - Repeatability



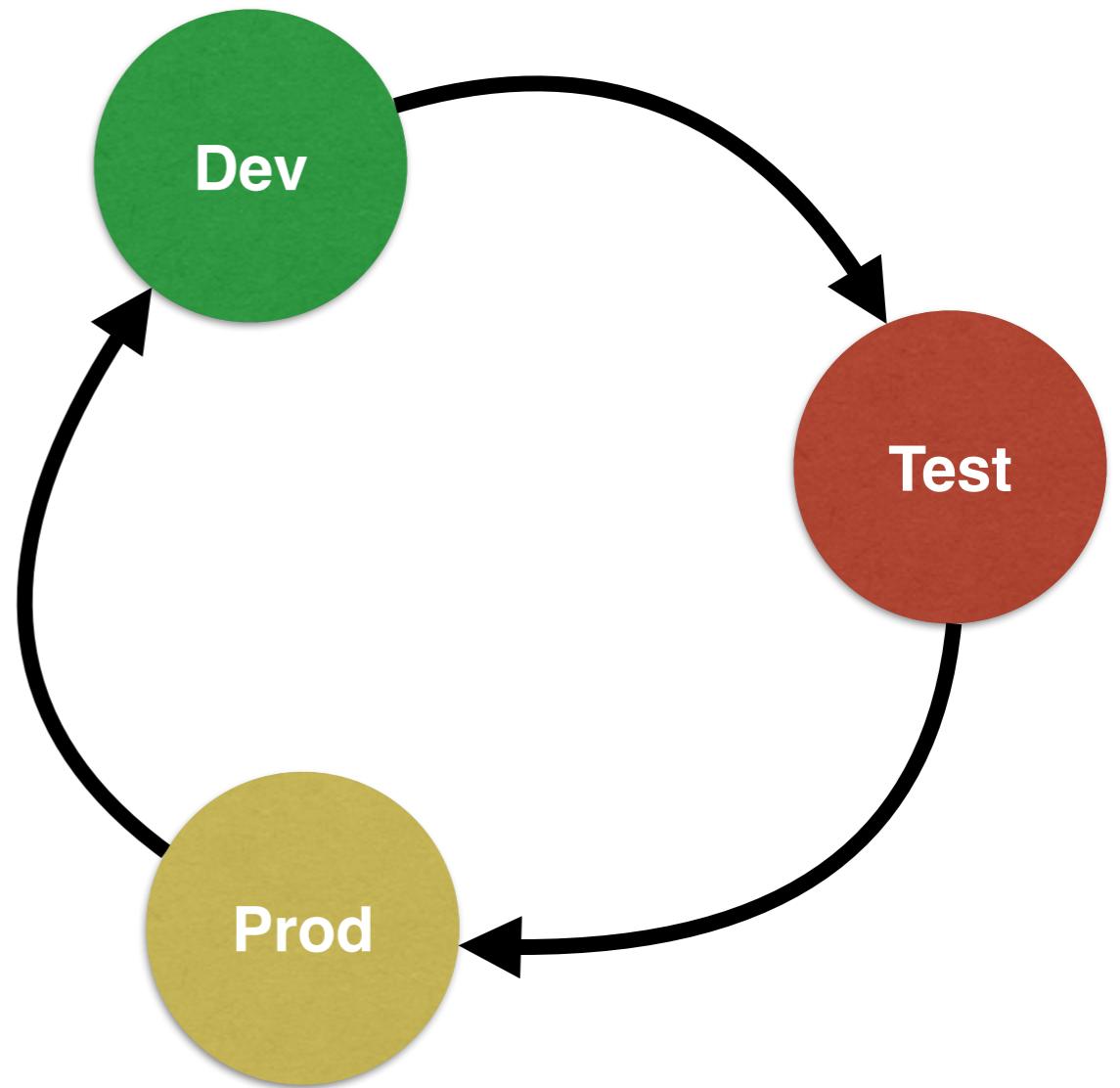
Consistency

- Automation over documentation
 - Repeatability
 - Push-button deployments



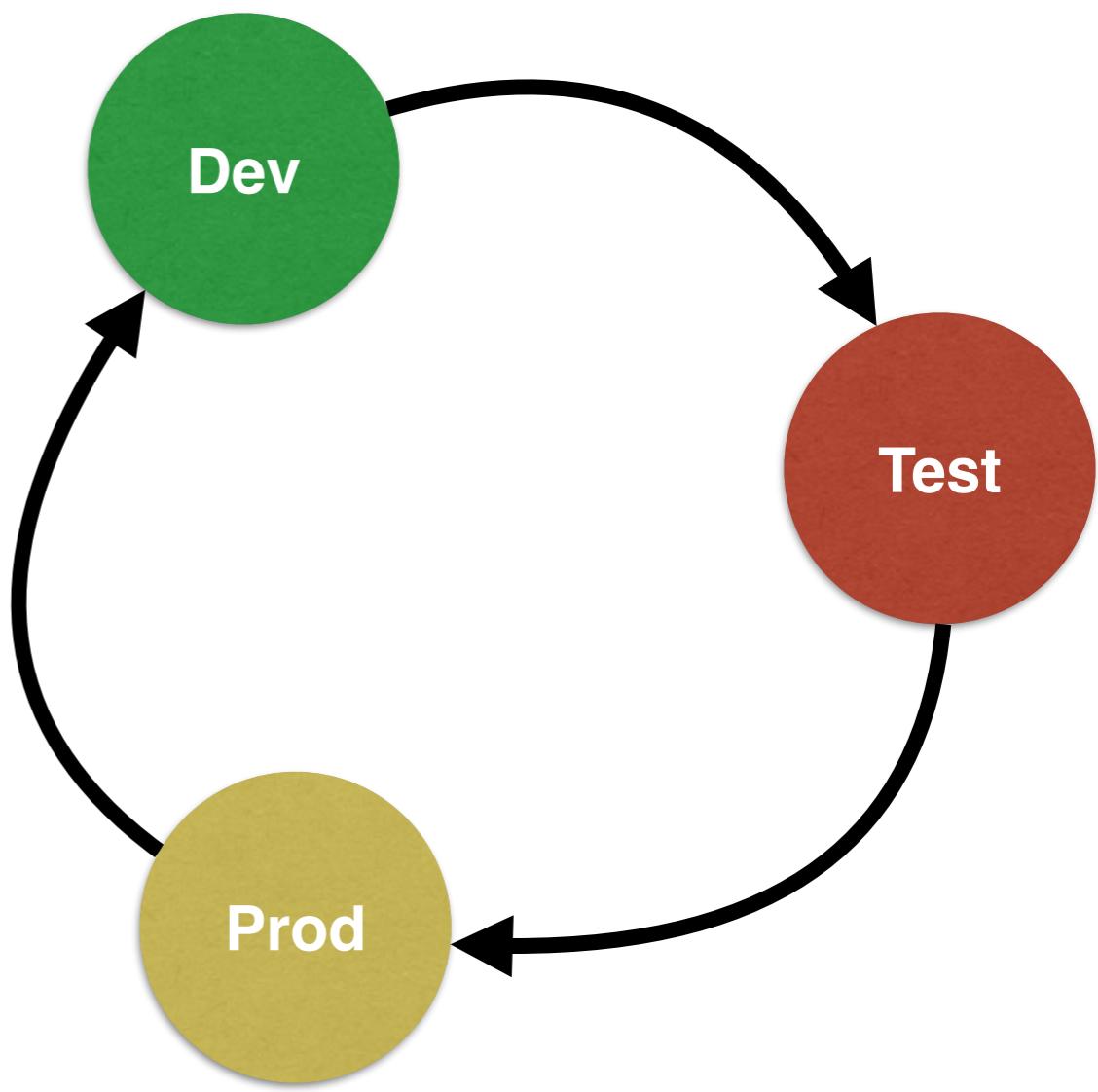
Consistency

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



Consistency

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



Manage environments

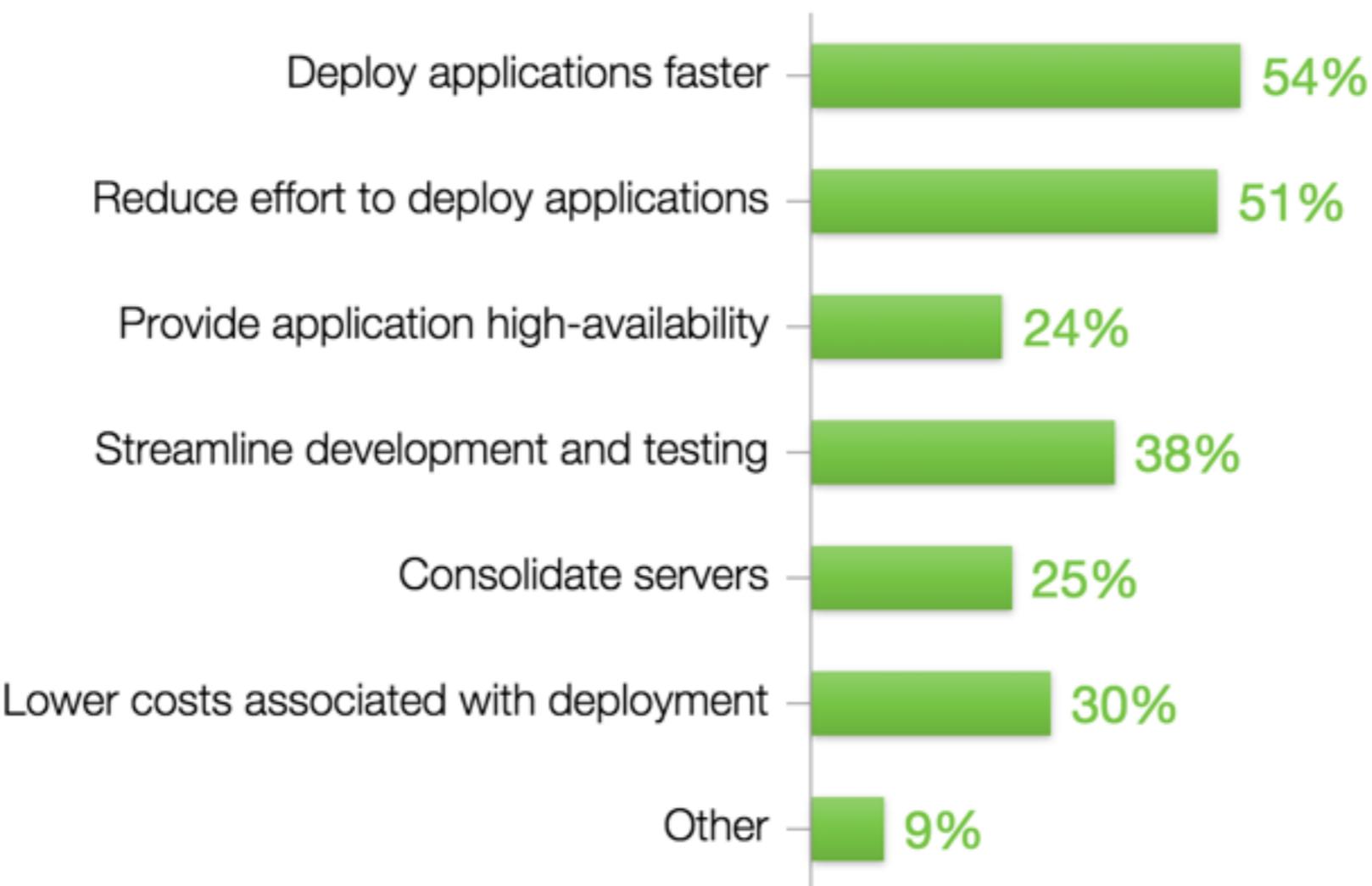
- Cloud computing: PaaS, OpenShift, ...
- Virtualization: Virtual Box, Vagrant, ...
- Containers: Docker, Rocket, ...
- App server: JBoss EAP, Tomcat, ...
- Configuration tools: Puppet, Chef, Ansible, Salt
- Orchestration: Kubernetes, Swarm, ...



- 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

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





- Open source project and company

Screenshot of the GitHub repository page for "docker/docker".

Repository details:

- Owner: arun-gupta
- Watchers: 1,554
- Stars: 18,448
- Forks: 3,766

Description: Docker - the open-source application container engine <http://www.docker.com>

Statistics:

- 12,603 commits
- 10 branches
- 81 releases
- 735 contributors

Branch selection: branch: master +

Recent activity:

- Merge pull request #9941 from SvenDowideit/build-pull-option-docs ... by jfrazelle an hour ago
- latest commit 00d19150bb

Side navigation:

- Code
- Issues (875)
- Pull Requests (81)
- Pulse



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

A screenshot of a web browser displaying the GitHub repository page for "docker/docker". The URL in the address bar is <https://github.com/docker/docker>. The page shows the repository's metadata: 12,603 commits, 10 branches, 81 releases, 735 contributors, 1,554 watchers, 18,448 stars, and 3,766 forks. A summary message at the top reads "Docker - the open-source application container engine <http://www.docker.com>". On the right side, there are sections for "Code", "Issues" (875), "Pull Requests" (81), and "Pulse". At the bottom, a merge pull request from "SvenDowideit/build-pull-option-docs" is shown, along with a commit by "jfrazelle" and the latest commit information.

Docker - the open-source application container engine <http://www.docker.com>

12,603 commits 10 branches 81 releases 735 contributors

branch: master docker / +

Merge pull request #9941 from SvenDowideit/build-pull-option-docs ...

jfrazelle authored an hour ago

latest commit 00d19150bb

Code

Issues 875

Pull Requests 81

Pulse



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

A screenshot of a web browser displaying the GitHub repository page for "docker/docker". The URL in the address bar is <https://github.com/docker/docker>. The page shows the repository's statistics: 12,603 commits, 10 branches, 81 releases, and 735 contributors. It also displays the master branch status, merge pull requests, and a recent commit by jfrazelle. The right sidebar includes links for Code, Issues (875), Pull Requests (81), and Pulse.

Docker - the open-source application container engine <http://www.docker.com>

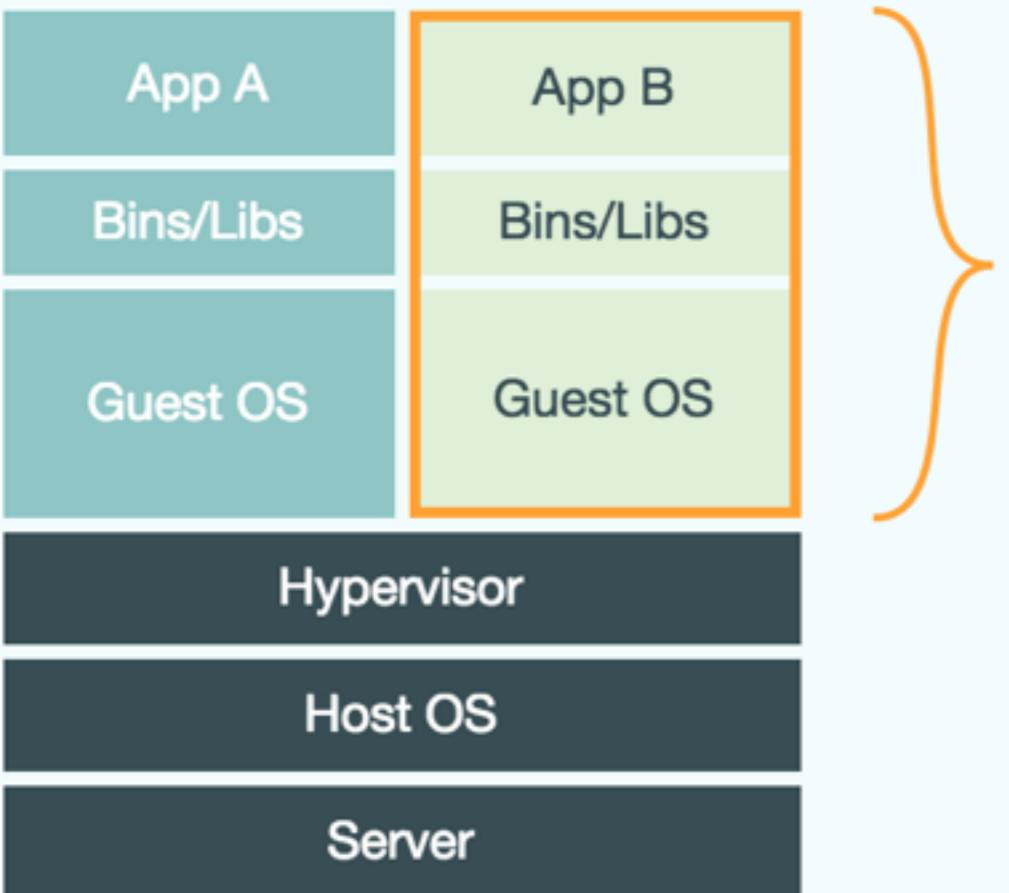
12,603 commits 10 branches 81 releases 735 contributors

branch: master docker / +

Merge pull request #9941 from SvenDowideit/build-pull-option-docs ...

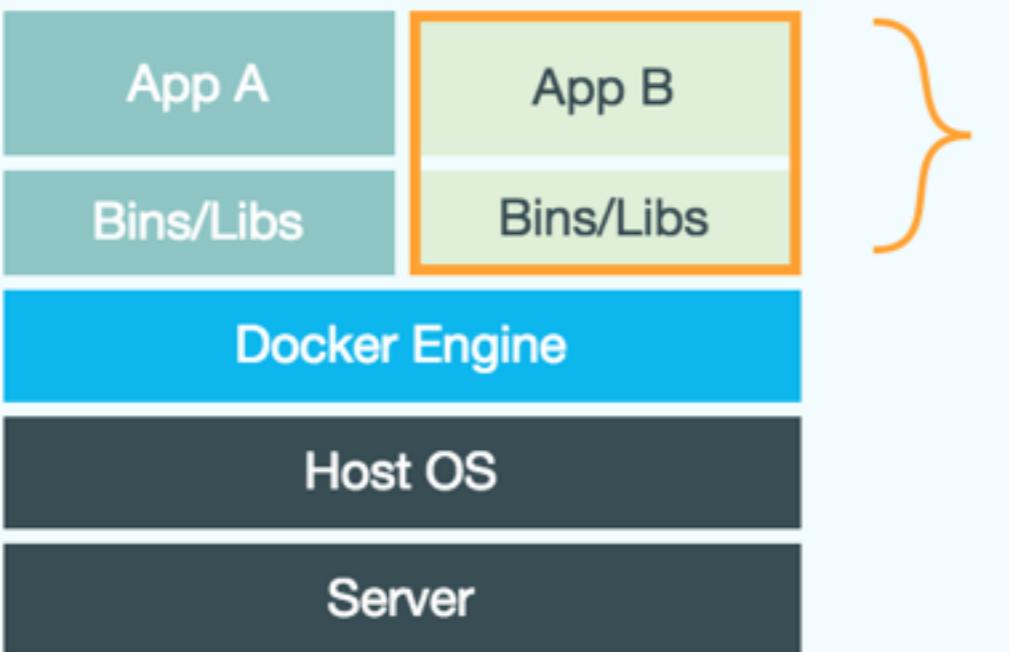
jfrazelle authored an hour ago latest commit 00d19150bb

Code Issues 875 Pull Requests 81 Pulse



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.



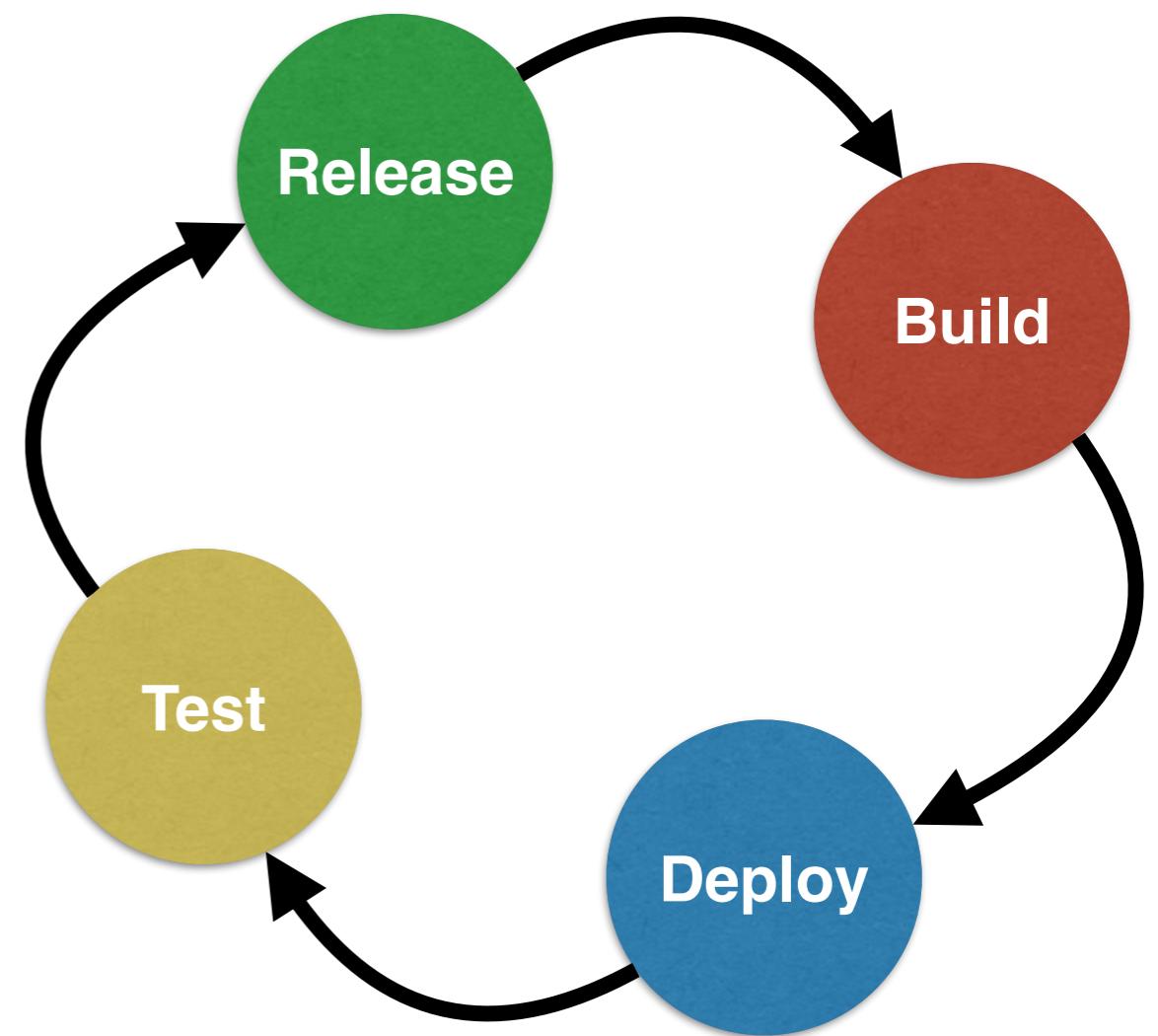
Docker

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

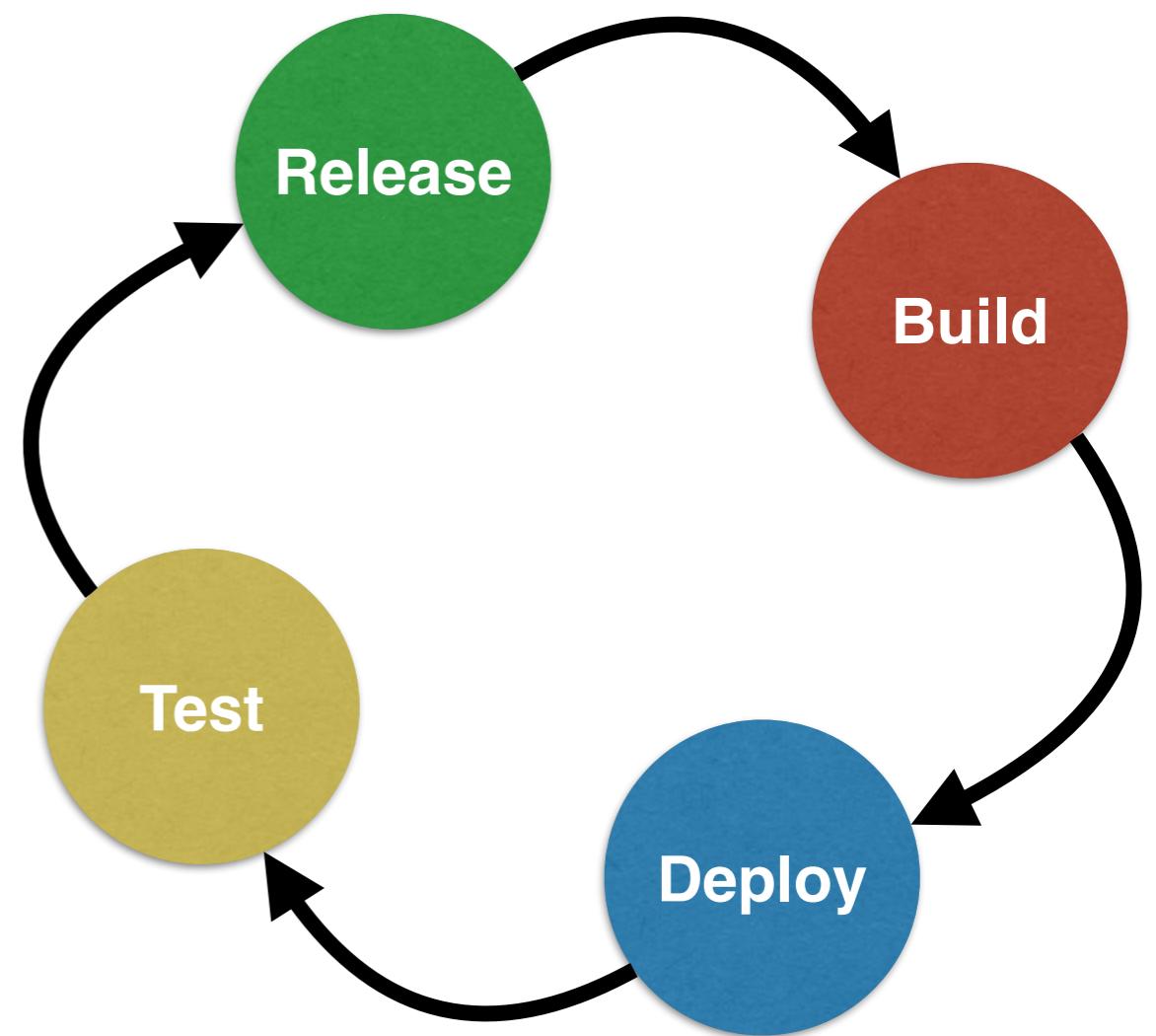
- Build dashboards, improve transparency
- stackexchange.com/performance

Continuous Delivery



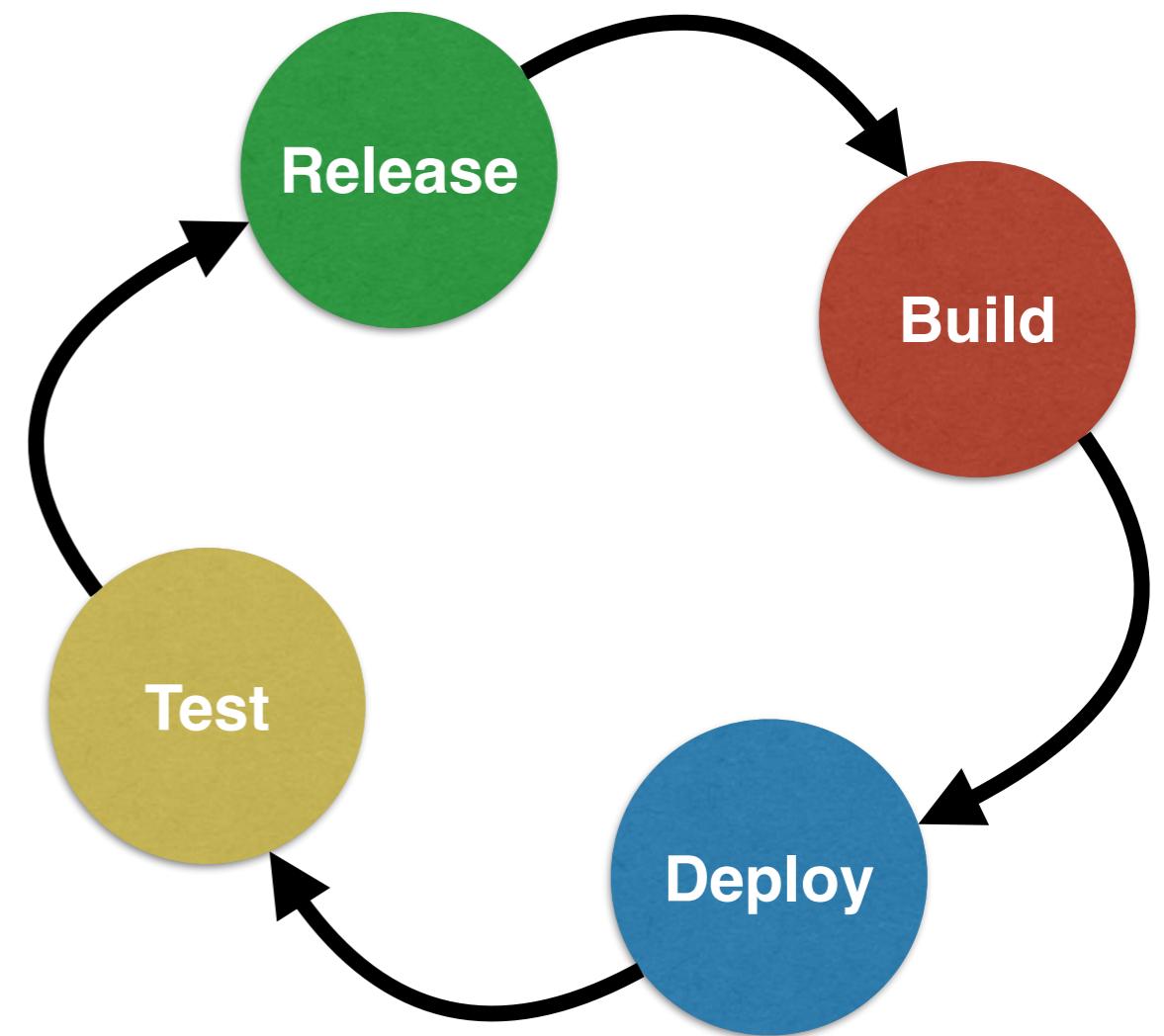
Continuous Delivery

- Agile Manifesto



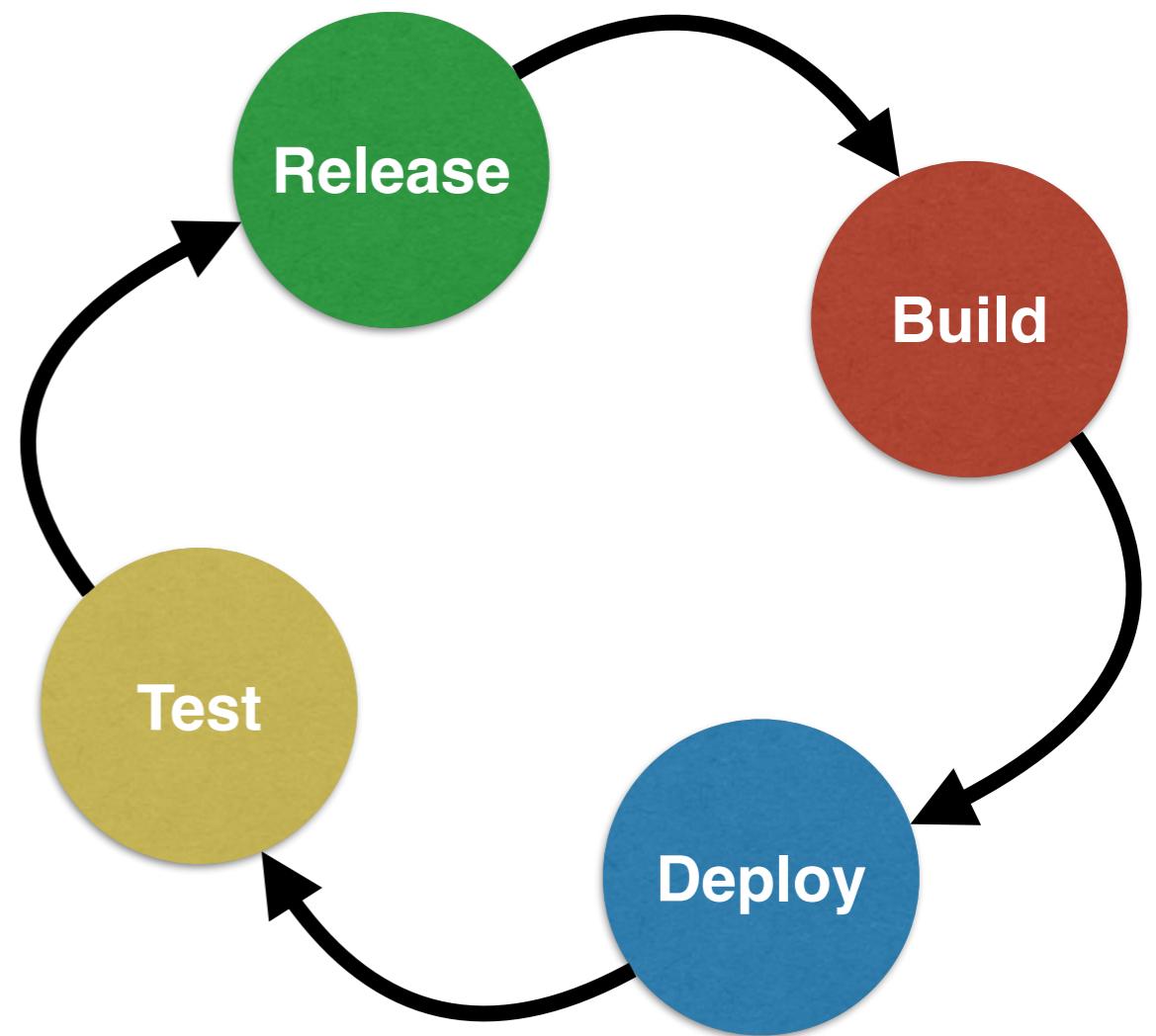
Continuous Delivery

- Agile Manifesto
- Continuous Integration



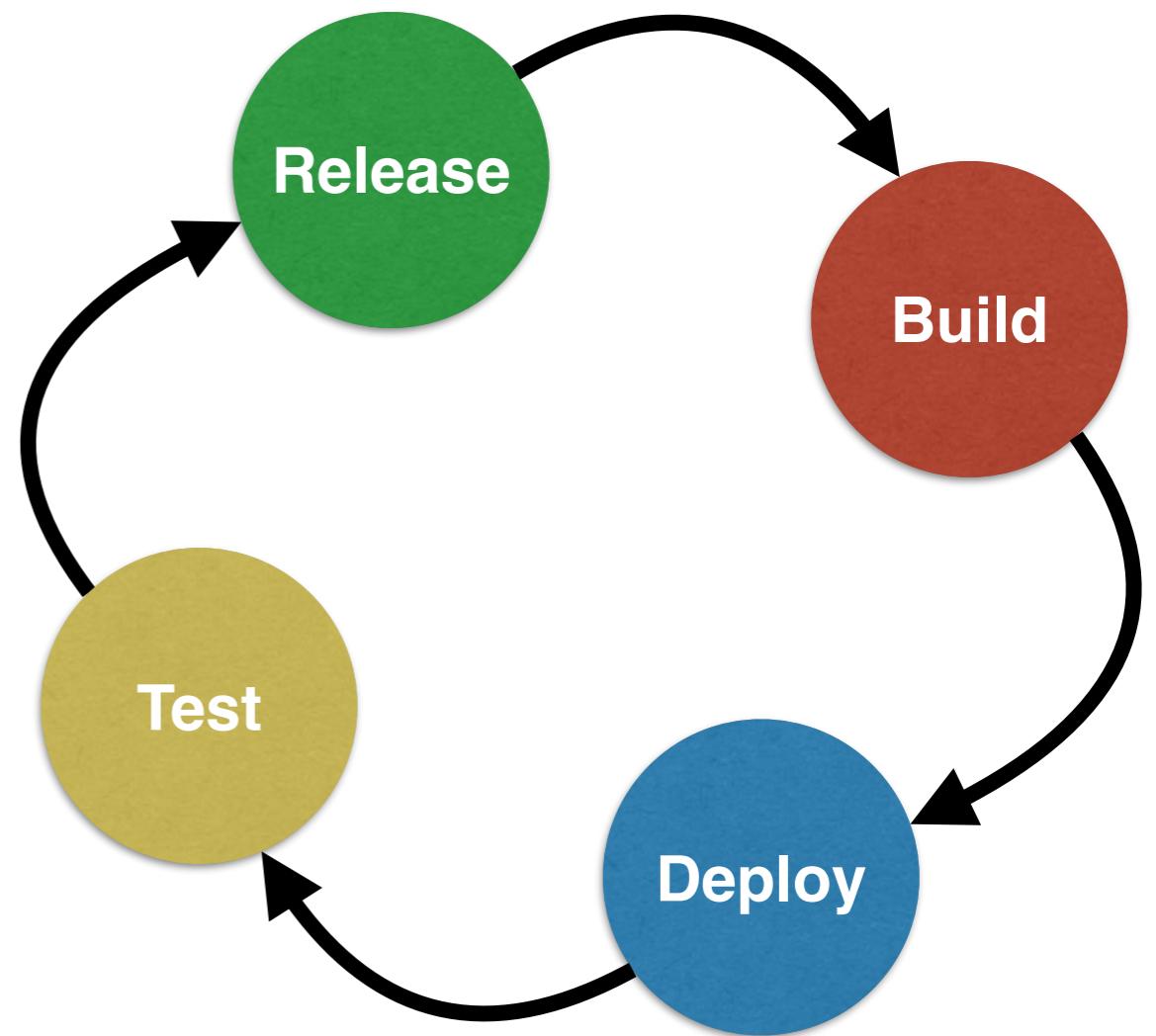
Continuous Delivery

- Agile Manifesto
- Continuous Integration
- Fail fast and recover



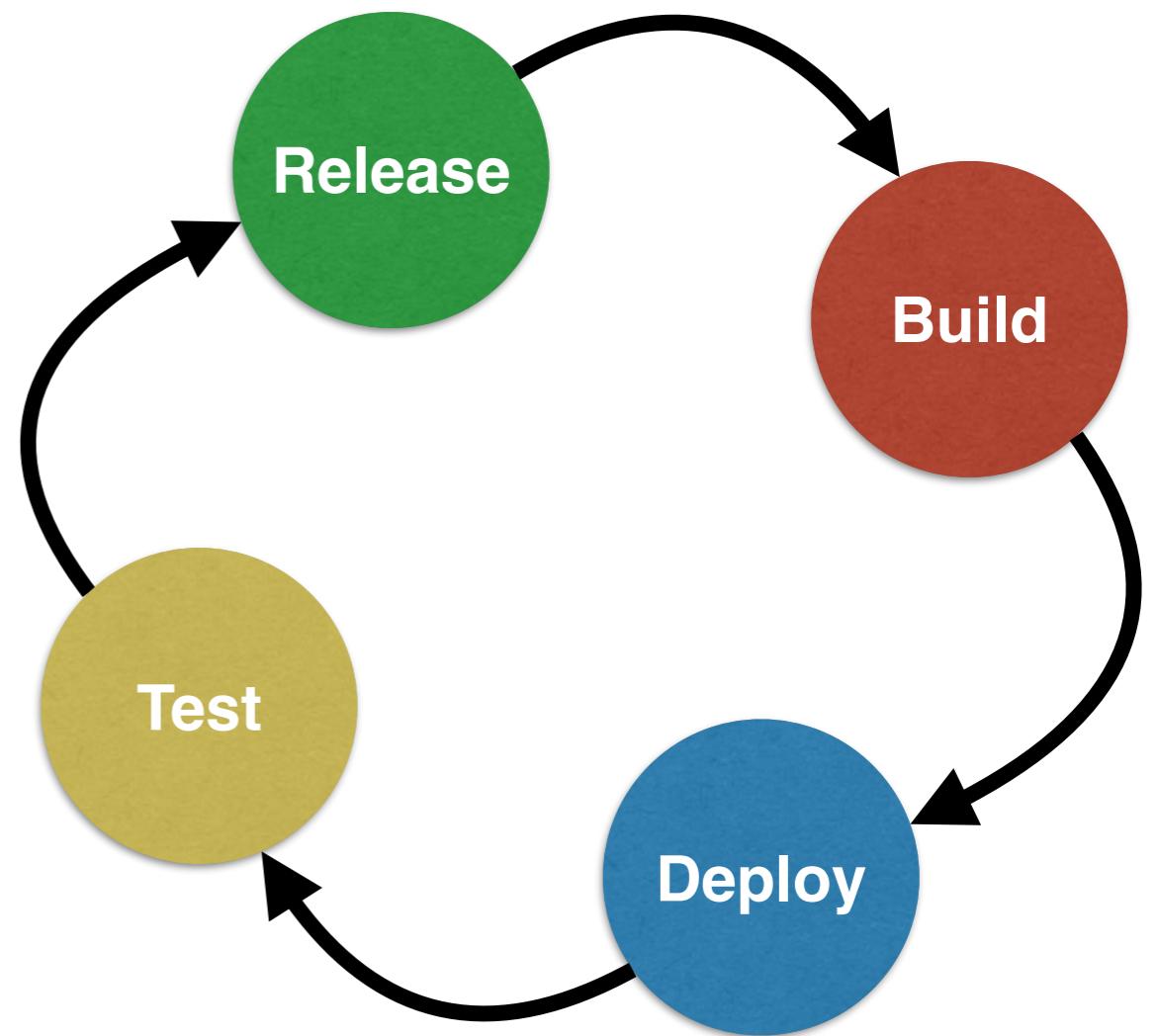
Continuous Delivery

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



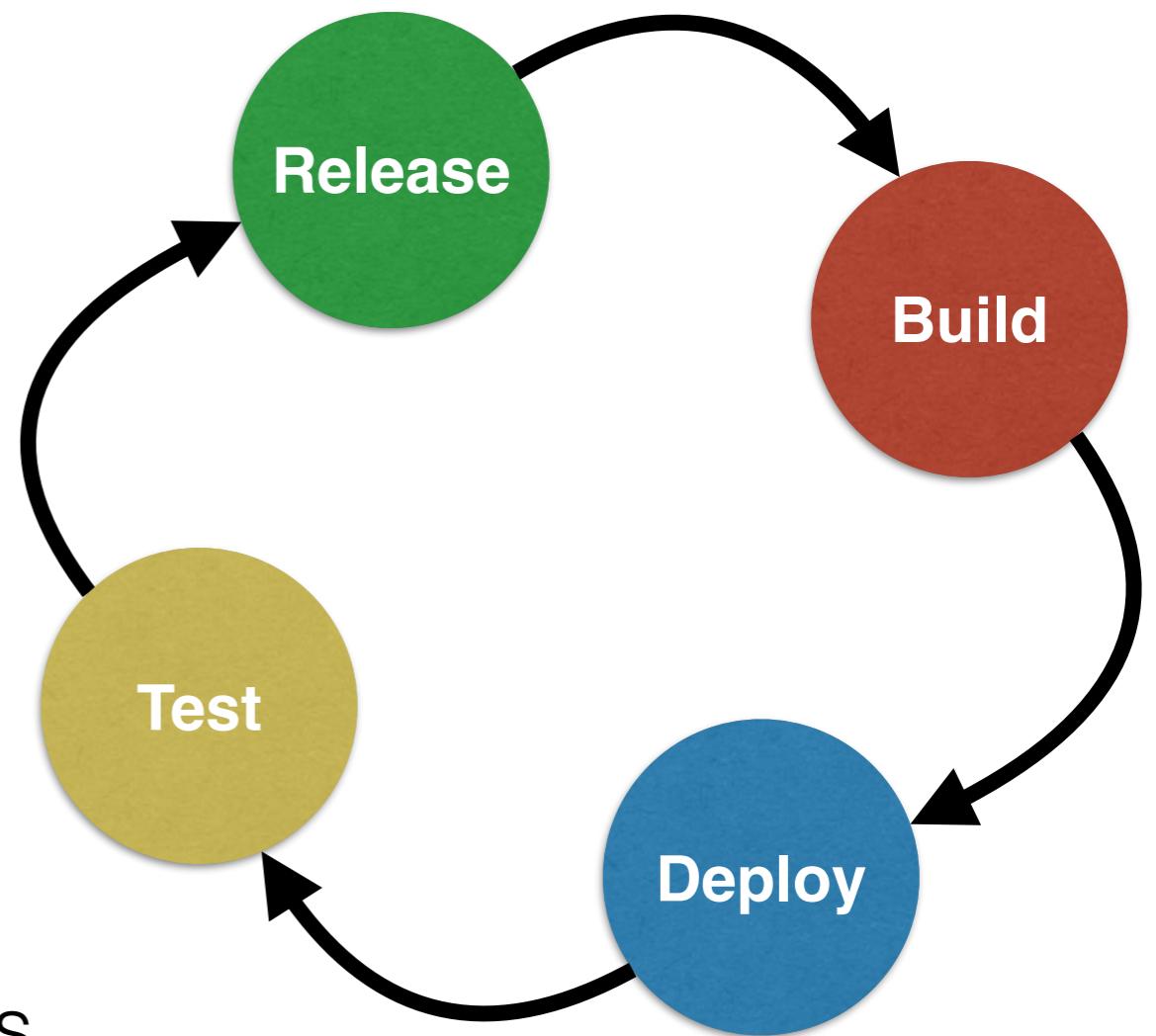
Continuous Delivery

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



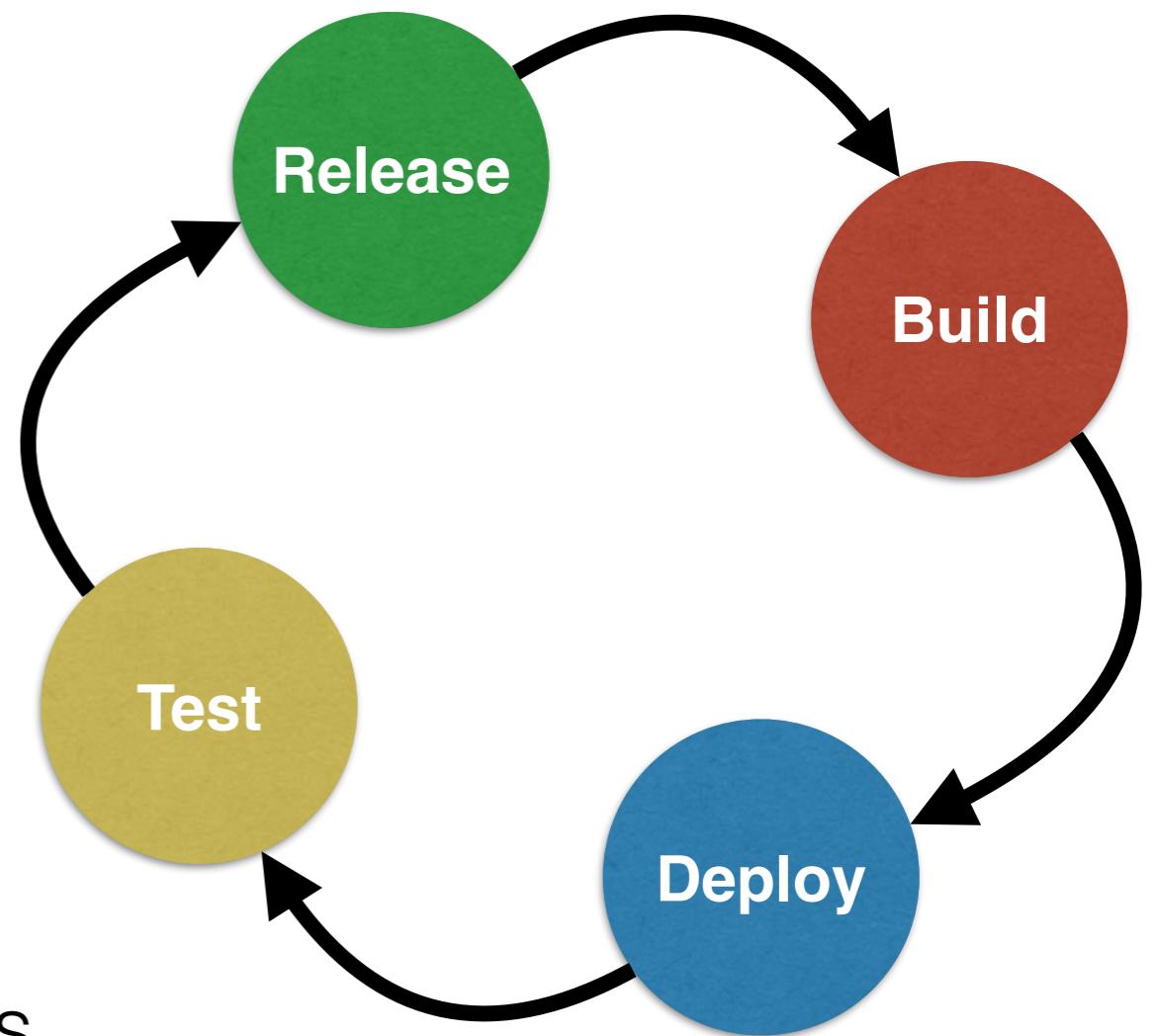
Continuous Delivery

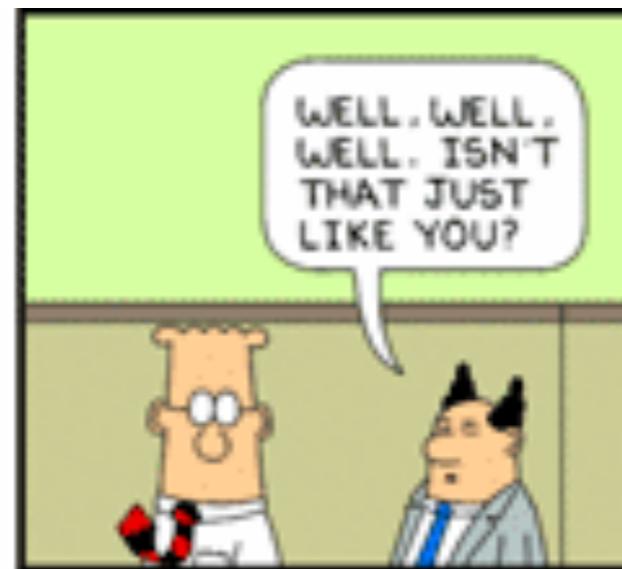
- Agile Manifesto
- Continuous Integration
- Fail fast and recover
- Self service
- 100% Automation
- Proactive monitoring and metrics

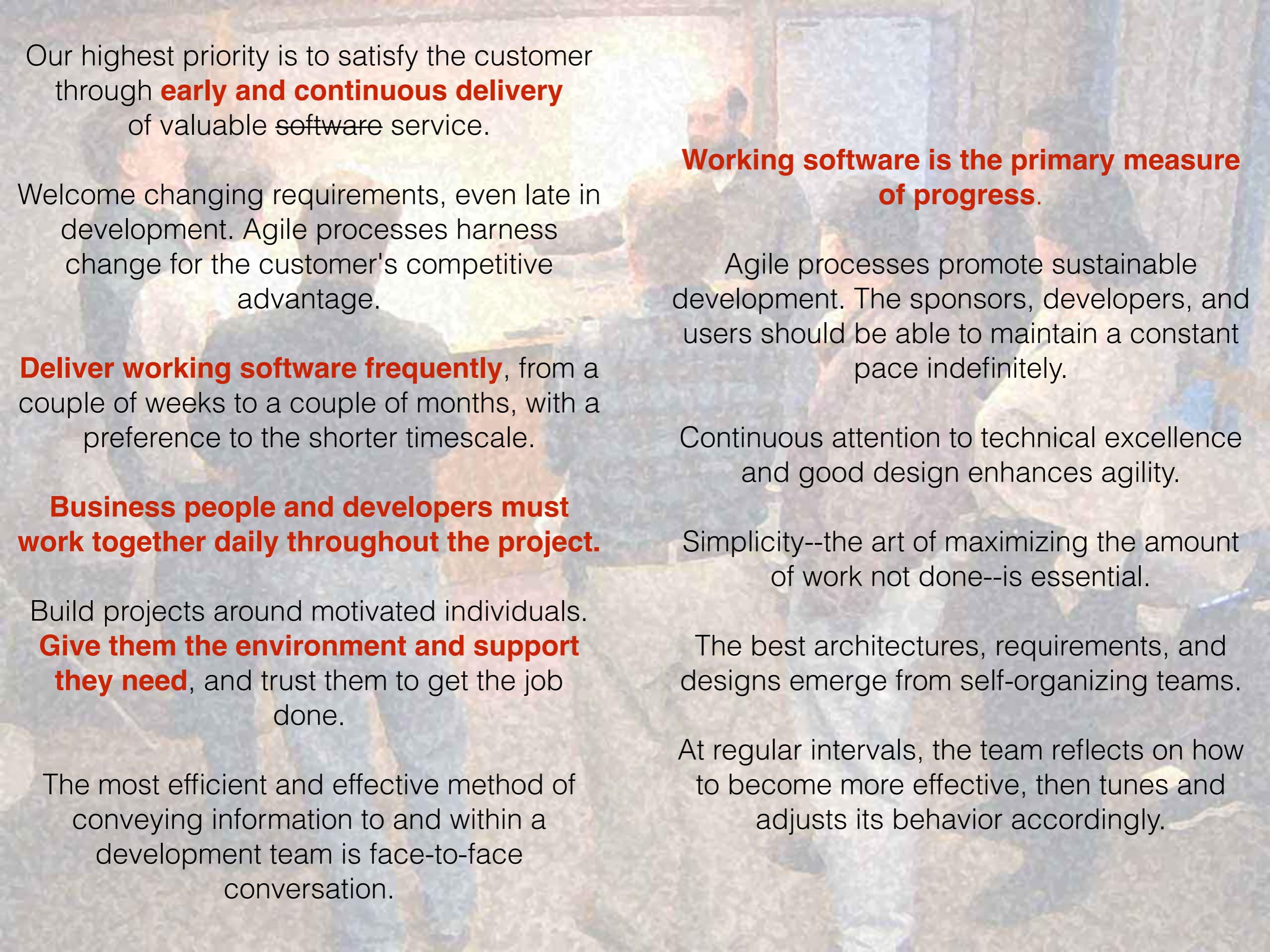


Continuous Delivery

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







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.

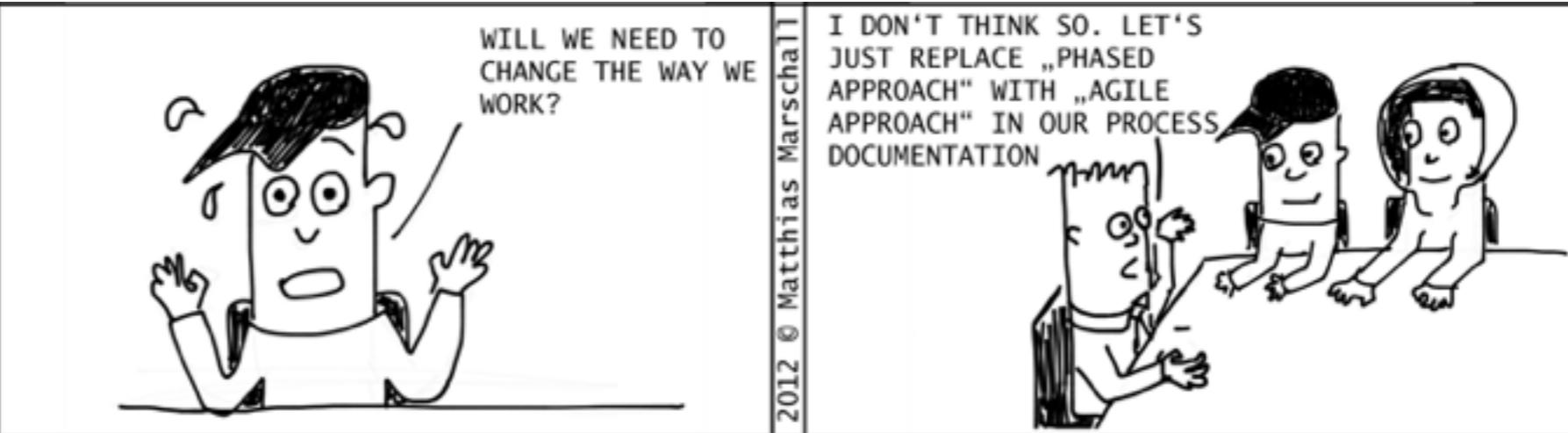
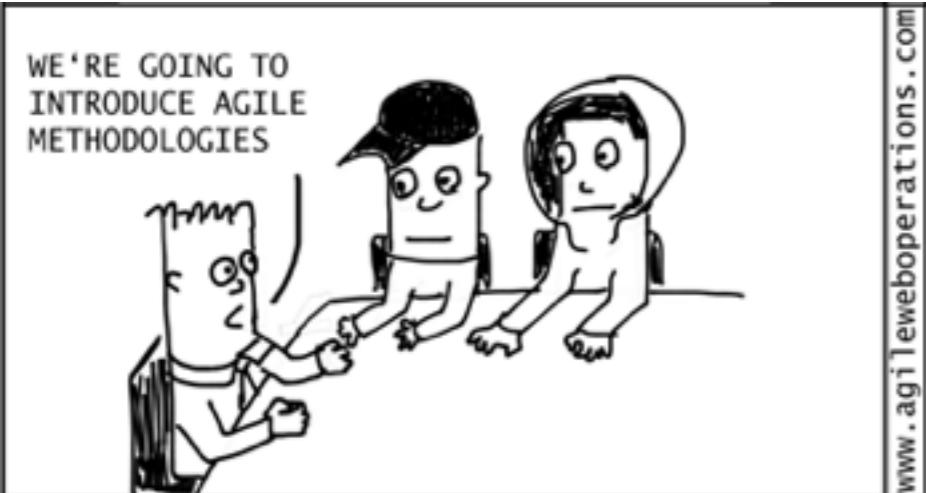
Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

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.

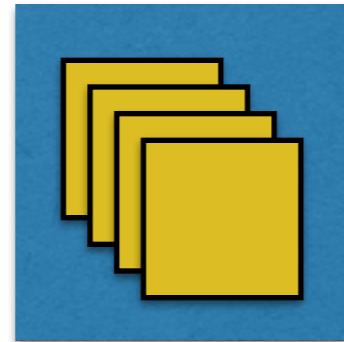
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.



Tools

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

Continuous Integration - One Developer



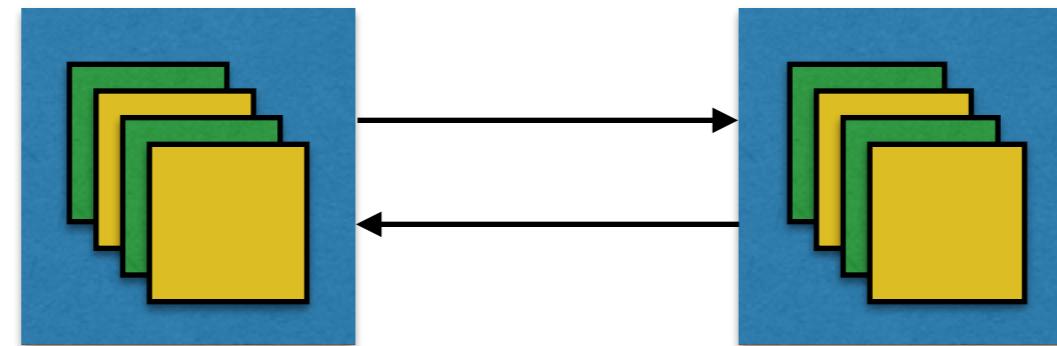
Continuous Integration - One Developer



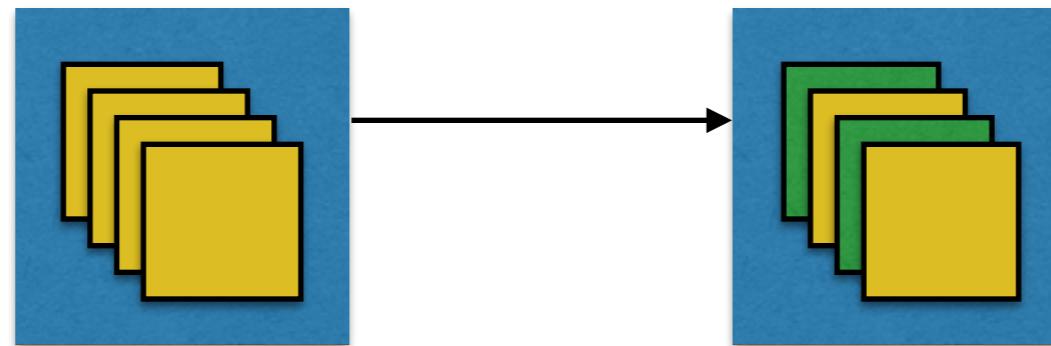
Continuous Integration - One Developer



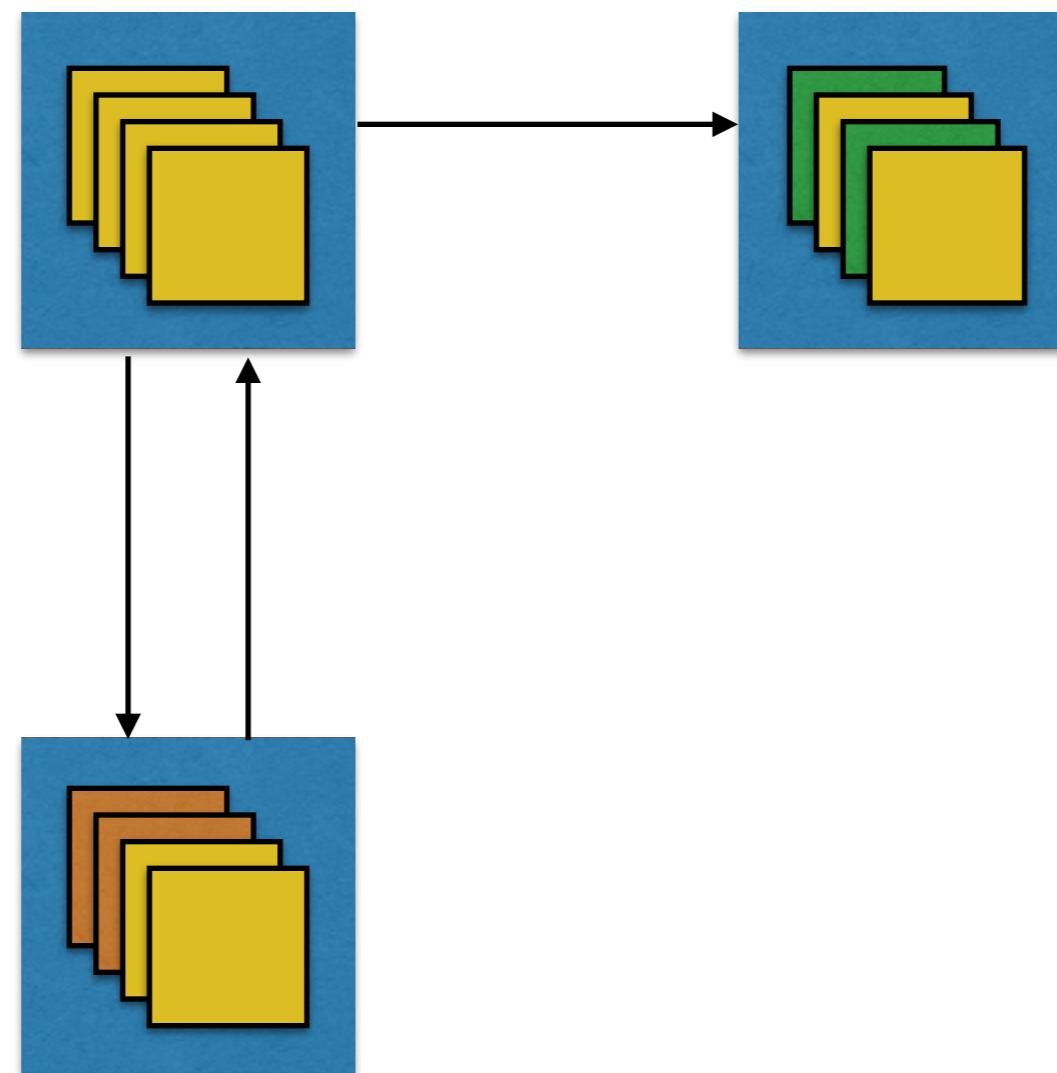
Continuous Integration - One Developer



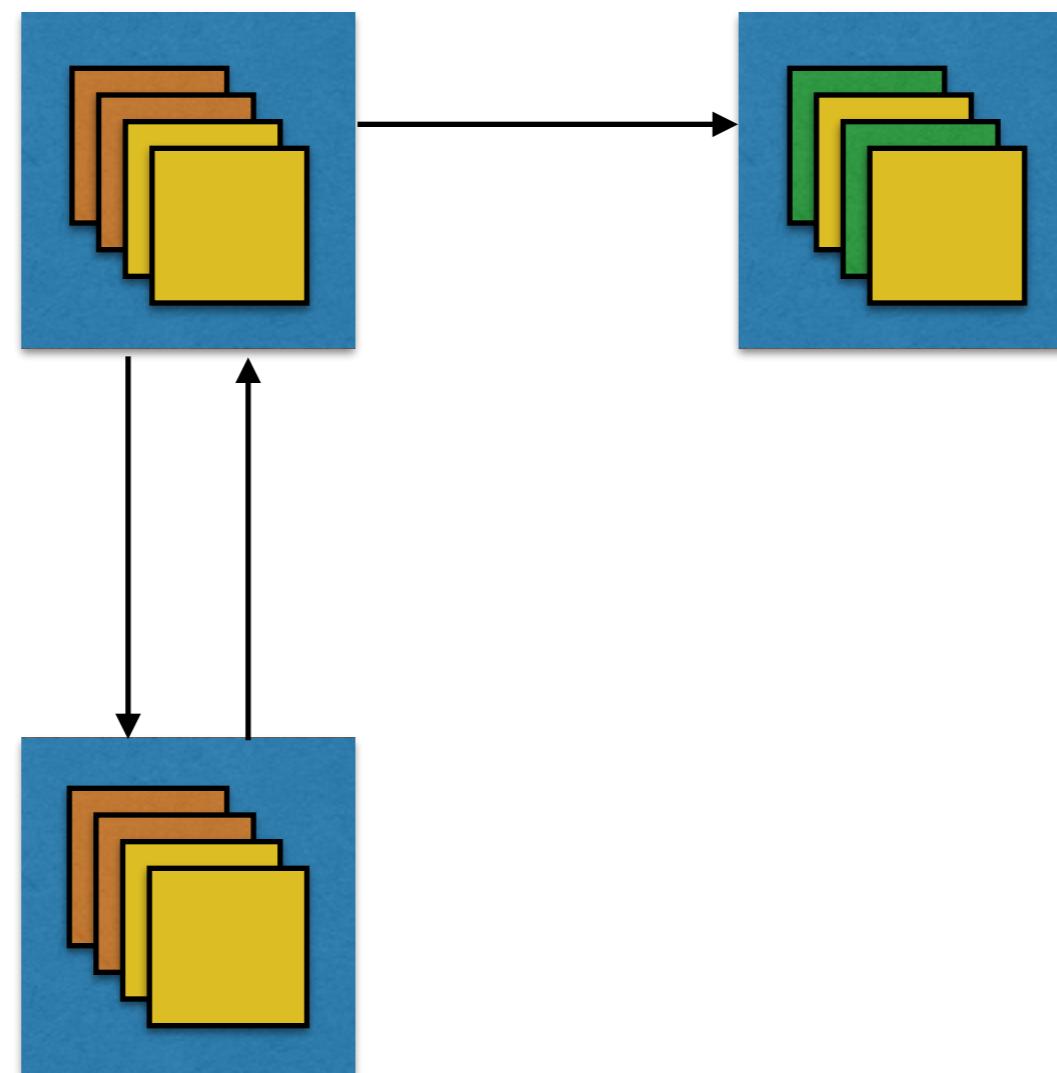
Continuous Integration - Multiple Developers



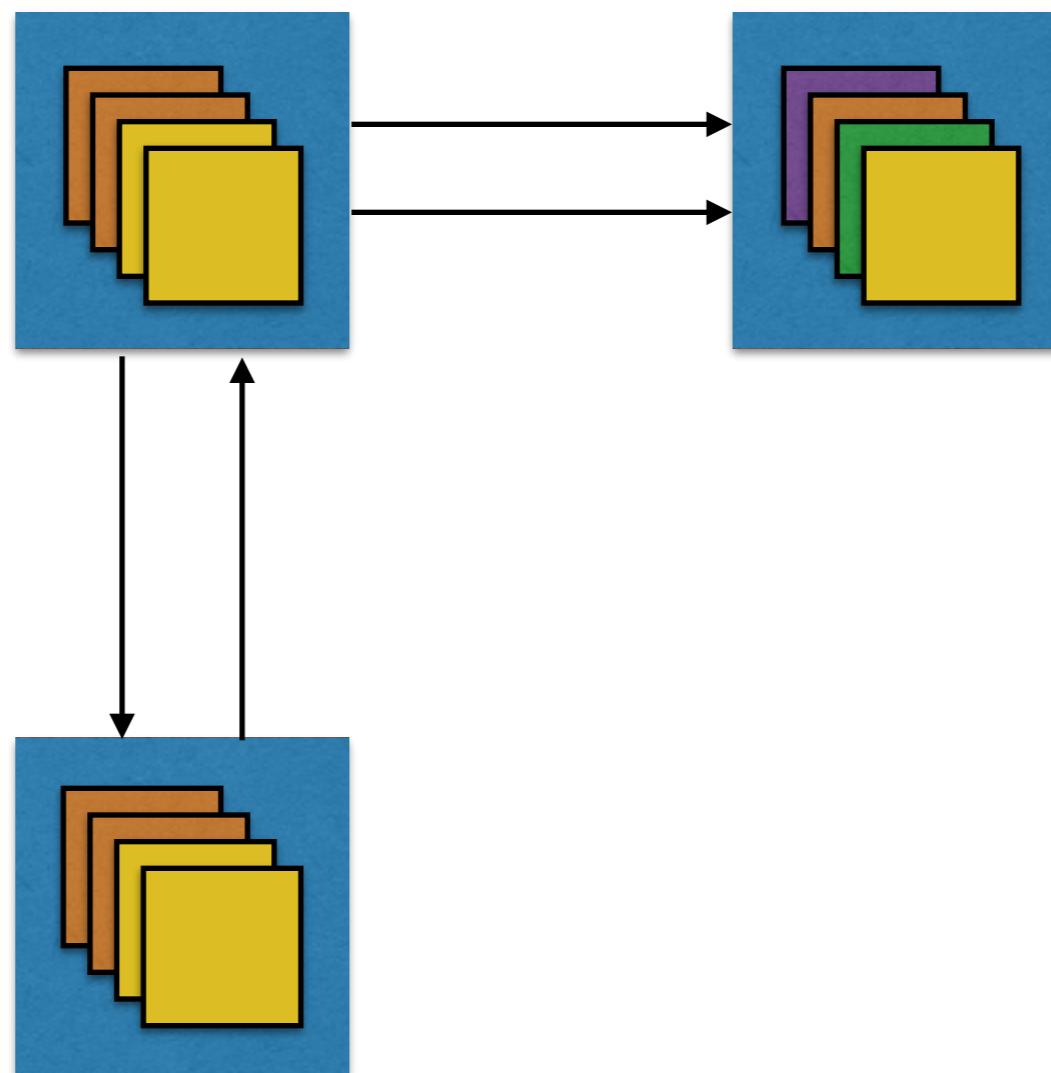
Continuous Integration - Multiple Developers



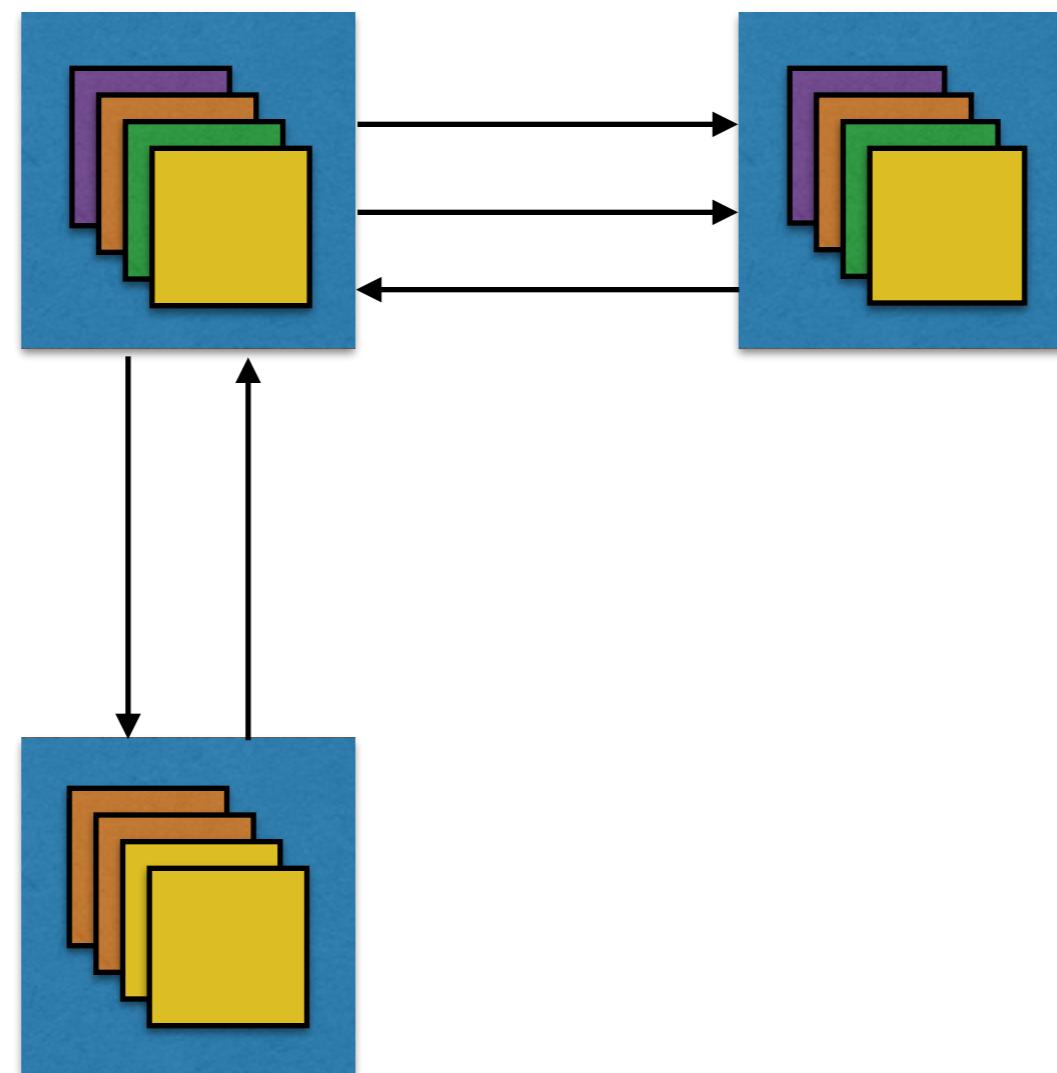
Continuous Integration - Multiple Developers



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

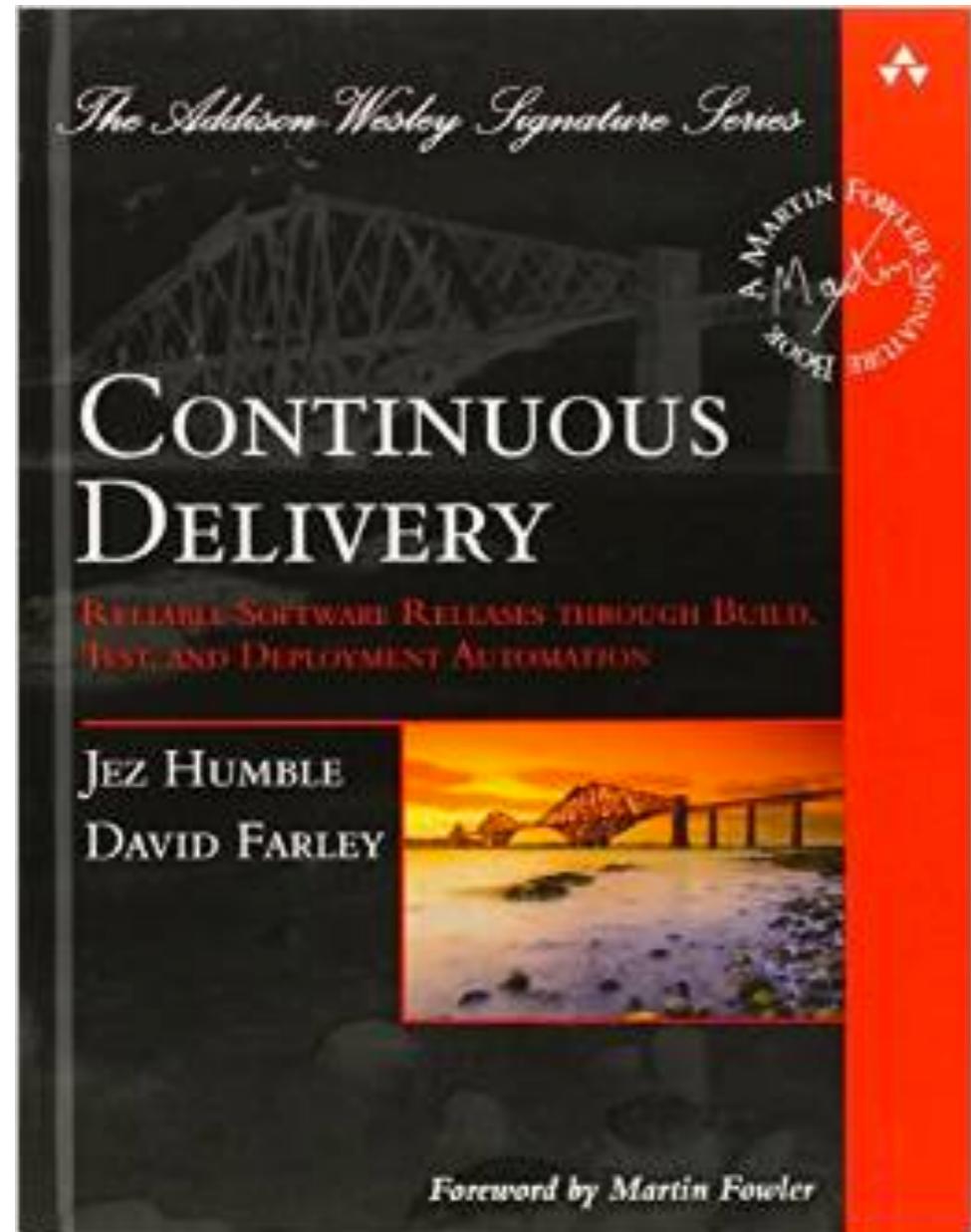
- Keep the build fast, rapid feedback (~10 mins)
- Commit early and often, once every few hours, at least once a day
- Check out, build, and test on a CI machine
- Nightly builds != CI

*“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>

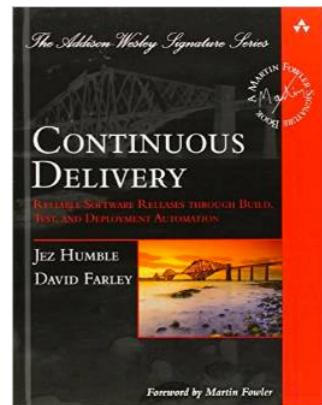
*“it is the practice
of releasing
every good build
to users”*



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?

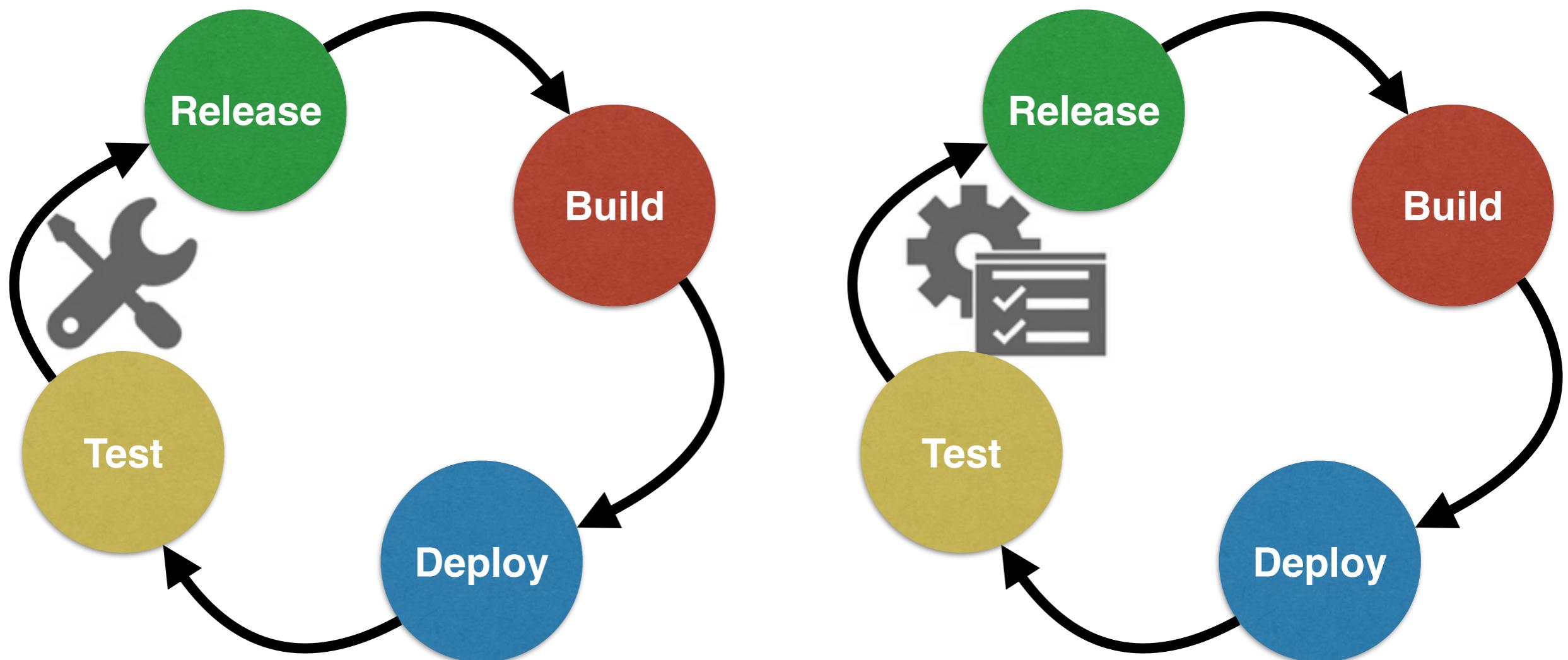
The Continuous Delivery Maturity Model

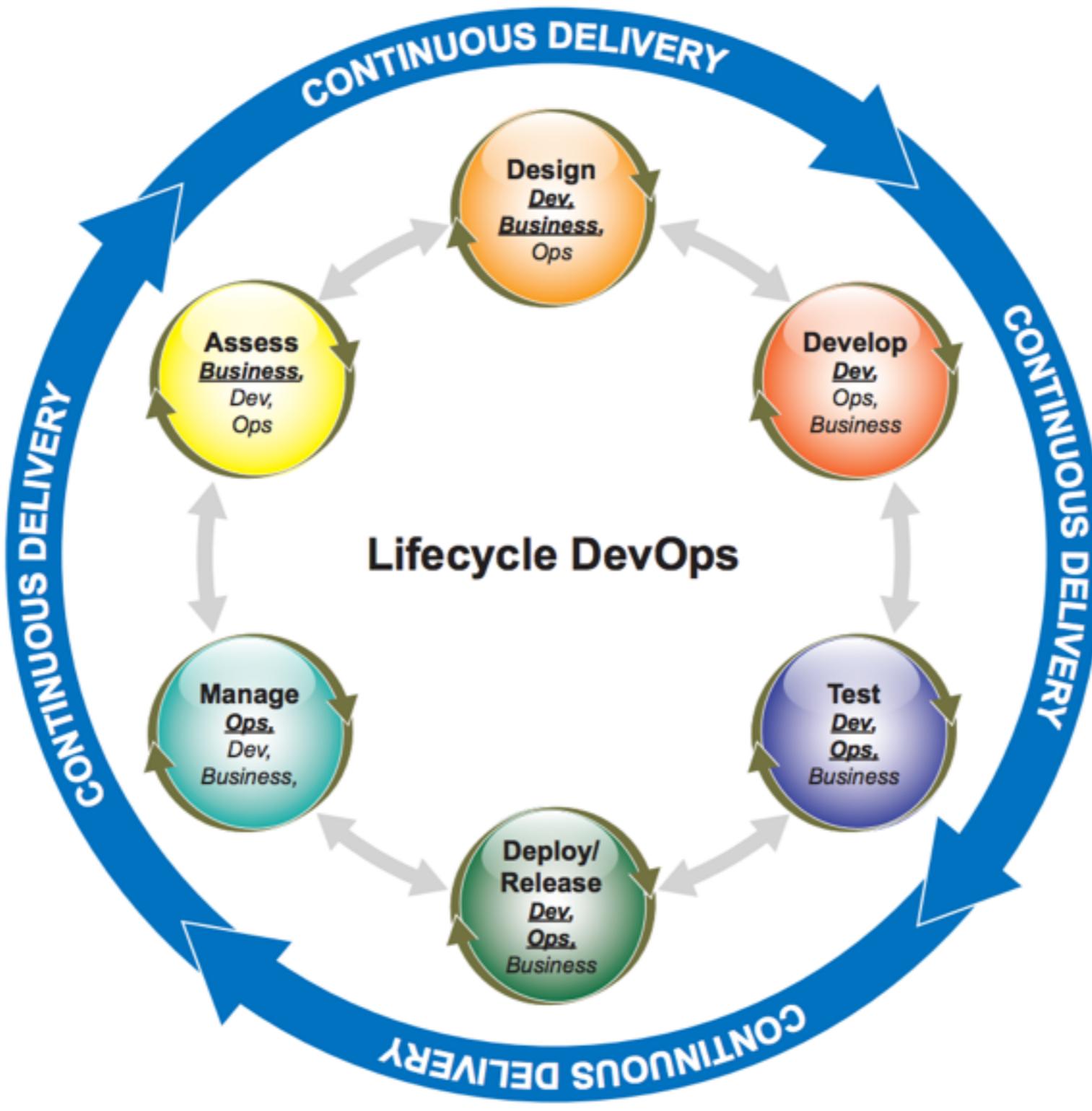
	Base	Beginner	Intermediate	Advanced	Expert
Culture & Organization	<ul style="list-style-type: none"> Prioritized work Defined and documented process Frequent commits 	<ul style="list-style-type: none"> One backlog per team Share the pain Stable teams Adopt basic Agile methods Remove boundary dev & test 	<ul style="list-style-type: none"> Extended team collaboration Component ownership Act on metrics Remove boundary dev & ops Common process for all changes Decentralize decisions 	<ul style="list-style-type: none"> Dedicated tools team Team responsible all the way to prod Deploy disconnected from Release Continuous improvement (Kaizen) 	<ul style="list-style-type: none"> Cross functional teams No rollbacks (always roll forward)
Design & Architecture	<ul style="list-style-type: none"> Consolidated platform & technology 	<ul style="list-style-type: none"> Organize system into modules API management Library management Version control DB changes 	<ul style="list-style-type: none"> No (or minimal) branching Branch by abstraction Configuration as code Feature hiding Making components out of modules 	<ul style="list-style-type: none"> Full component based architecture Push business metrics 	<ul style="list-style-type: none"> Infrastructure as code
Build & Deploy	<ul style="list-style-type: none"> Versioned code base Scripted builds Basic scheduled builds (CI) Dedicated build server Documented manual deploy Some deployment scripts exists 	<ul style="list-style-type: none"> Polling builds Builds are stored Manual tag & versioning First step towards standardized deploys 	<ul style="list-style-type: none"> 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 	<ul style="list-style-type: none"> Zero downtime deploys Multiple build machines Full automatic DB deploys 	<ul style="list-style-type: none"> Build bakery Zero touch continuous deployments
Test & Verification	<ul style="list-style-type: none"> Automatic unit tests Separate test environment 	<ul style="list-style-type: none"> Automatic integration tests 	<ul style="list-style-type: none"> Automatic component tests (isolated) Some automatic acceptance tests 	<ul style="list-style-type: none"> Full automatic acceptance tests Automatic performance tests Automatic security tests Risk based manual testing 	<ul style="list-style-type: none"> Verify expected business value
Information & Reporting	<ul style="list-style-type: none"> Baseline process metrics Manual reporting 	<ul style="list-style-type: none"> Measure the process Static code analysis Scheduled quality reports 	<ul style="list-style-type: none"> Common information model Traceability built into pipeline Report history is available 	<ul style="list-style-type: none"> Graphing as a service Dynamic test coverage analysis Report trend analysis 	<ul style="list-style-type: none"> Dynamic graphing and dashboards Cross silo analysis

CD Maturity

Deploy Frequency \propto DevOps Maturity

Continuous Deployment







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



- **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
- **Hystrix**: Builds resiliency, graceful fallback
- **Simian Army**: test resilience at runtime
 - Chaos, Latency, Janitor, Conformity, ...





<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



- 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



- 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>



Barack Obama

33,277,146 likes · 12,248,455 talking about this

Like

▼



Chartbeat **Nagios**



ubuntu®



 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

DevOps with Java EE

DevOps with Java EE



DevOps with Java EE



DevOps with Java EE



DevOps with Java EE



DevOps with Java EE





WildFly



WildFly

- Red Hat's open source Java EE 7 compliant application server



WildFly

- Red Hat's open source Java EE 7 compliant application server
- 8.0 (Feb 2014), 8.1 released (May 2014), 8.2 (Nov 2014), 9.0 next year



WildFly

- Red Hat's open source Java EE 7 compliant application server
- 8.0 (Feb 2014), 8.1 released (May 2014), 8.2 (Nov 2014), 9.0 next year
- Commercial support coming in JBoss Enterprise Application Platform (EAP) 7



JBoss Tools

- Eclipse plugins for JBoss technology

JBoss Tools

- Eclipse plugins for JBoss technology





Forge

Forge

- Fastest way to build Maven-based Java EE projects

Forge

- Fastest way to build Maven-based Java EE projects
- WORAI for Java EE (using Maven)

Forge

- Fastest way to build Maven-based Java EE projects
- WORAI for Java EE (using Maven)
 - Aids in automation of repetitive tasks

Forge

- Fastest way to build Maven-based Java EE projects
- WORAI for Java EE (using Maven)
 - Aids in automation of repetitive tasks
 - Generation of boilerplate code

Forge

- Fastest way to build Maven-based Java EE projects
- WORAI for Java EE (using Maven)
 - Aids in automation of repetitive tasks
 - Generation of boilerplate code
 - In-built support for Java EE 6 and 7

Forge

- Fastest way to build Maven-based Java EE projects
- WORAI for Java EE (using Maven)
 - Aids in automation of repetitive tasks
 - Generation of boilerplate code
 - In-built support for Java EE 6 and 7
- Extensible by design

Forge

- Fastest way to build Maven-based Java EE projects
- WORAI for Java EE (using Maven)
 - Aids in automation of repetitive tasks
 - Generation of boilerplate code
 - In-built support for Java EE 6 and 7
- Extensible by design
- Test-driven development



OpenShift



OPENSIFT

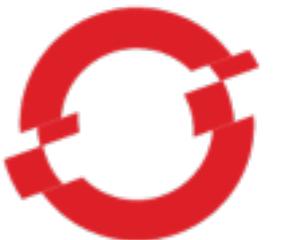
OpenShift

- OpenShift Origin, Online, Enterprise

The screenshot shows the OpenShift Origin landing page. It features a large laptop icon at the top left, followed by the word "ORIGIN" and "Community PaaS". Below this, there is a brief description: "Explore the community-driven open source upstream of OpenShift. Download the bits, join the growing community, and help extend the functionality of OpenShift." A "Learn more" button is located below the description, and a prominent red "JOIN THE COMMUNITY" button is at the bottom right.

The screenshot shows the OpenShift Online landing page. It features a large cloud icon at the top left, followed by the word "ONLINE" and "Public PaaS". Below this, there is a brief description: "Host your applications in the public cloud. OpenShift Online automates the provisioning, management and scaling of applications so that you can focus on development and creativity." A "Learn more" button is located below the description, and a prominent red "SIGN UP FOR FREE" button is at the bottom right.

The screenshot shows the OpenShift Enterprise landing page. It features a large server icon at the top left, followed by the word "ENTERPRISE" and "Private PaaS". Below this, there is a brief description: "Accelerate your IT service delivery and streamline application development by leveraging PaaS in your own datacenters or private cloud." A "Learn more" button is located below the description, and a prominent red "REQUEST EVALUATION" button is at the bottom right.



OPENSIFT

- OpenShift Origin, Online, Enterprise
- Pluggable cartridges

The screenshot shows the OpenShift Origin landing page. It features a laptop icon and the word "ORIGIN" in large letters. Below it, the text "Community PaaS" is visible. A descriptive paragraph encourages users to explore the community-driven open source upstream of OpenShift. A "Learn more" button and a prominent red "JOIN THE COMMUNITY" button are at the bottom.

The screenshot shows the OpenShift Online landing page. It features a cloud icon and the word "ONLINE" in large letters. Below it, the text "Public PaaS" is visible. A descriptive paragraph explains that OpenShift Online automates the provisioning, management and scaling of applications so that you can focus on development and creativity. A "Learn more" button and a red "SIGN UP FOR FREE" button are at the bottom.

The screenshot shows the OpenShift Enterprise landing page. It features a server icon and the word "ENTERPRISE" in large letters. Below it, the text "Private PaaS" is visible. A descriptive paragraph accelerates IT service delivery and streamlines application development by leveraging PaaS in your own datacenters or private cloud. A "Learn more" button and a red "REQUEST EVALUATION" button are at the bottom.



OpenShift

- OpenShift Origin, Online, Enterprise
- Pluggable cartridges
- Multiple languages, databases, app servers

ORIGIN
Community PaaS

Explore the community-driven open source upstream of OpenShift. Download the bits, join the growing community, and help extend the functionality of OpenShift.

Learn more →

JOIN THE COMMUNITY

ONLINE
Public PaaS

Host your applications in the public cloud. OpenShift Online automates the provisioning, management and scaling of applications so that you can focus on development and creativity.

Learn more →

SIGN UP FOR FREE

ENTERPRISE
Private PaaS

Accelerate your IT service delivery and streamline application development by leveraging PaaS in your own datacenters or private cloud.

Learn more →

REQUEST EVALUATION



OPENSIFT

- OpenShift Origin, Online, Enterprise
- Pluggable cartridges
- Multiple languages, databases, app servers
- Gears: small, medium, large

ORIGIN
Community PaaS

Explore the community-driven open source upstream of OpenShift. Download the bits, join the growing community, and help extend the functionality of OpenShift.

Learn more

JOIN THE COMMUNITY

ONLINE
Public PaaS

Host your applications in the public cloud. OpenShift Online automates the provisioning, management and scaling of applications so that you can focus on development and creativity.

Learn more

SIGN UP FOR FREE

ENTERPRISE
Private PaaS

Accelerate your IT service delivery and streamline application development by leveraging PaaS in your own datacenters or private cloud.

Learn more

REQUEST EVALUATION



OPENSIFT

- OpenShift Origin, Online, Enterprise
- Pluggable cartridges
- Multiple languages, databases, app servers
- Gears: small, medium, large
 - FREE account: 3 gears (1 GB disk, 0.5 GB RAM)

ORIGIN
Community PaaS

Explore the community-driven open source upstream of OpenShift. Download the bits, join the growing community, and help extend the functionality of OpenShift.

Learn more →

JOIN THE COMMUNITY

ONLINE
Public PaaS

Host your applications in the public cloud. OpenShift Online automates the provisioning, management and scaling of applications so that you can focus on development and creativity.

Learn more →

SIGN UP FOR FREE

ENTERPRISE
Private PaaS

Accelerate your IT service delivery and streamline application development by leveraging PaaS in your own datacenters or private cloud.

Learn more →

REQUEST EVALUATION



Arquillian

Highly **extensible testing platform** for the JVM that enables to easily create **automated tests**

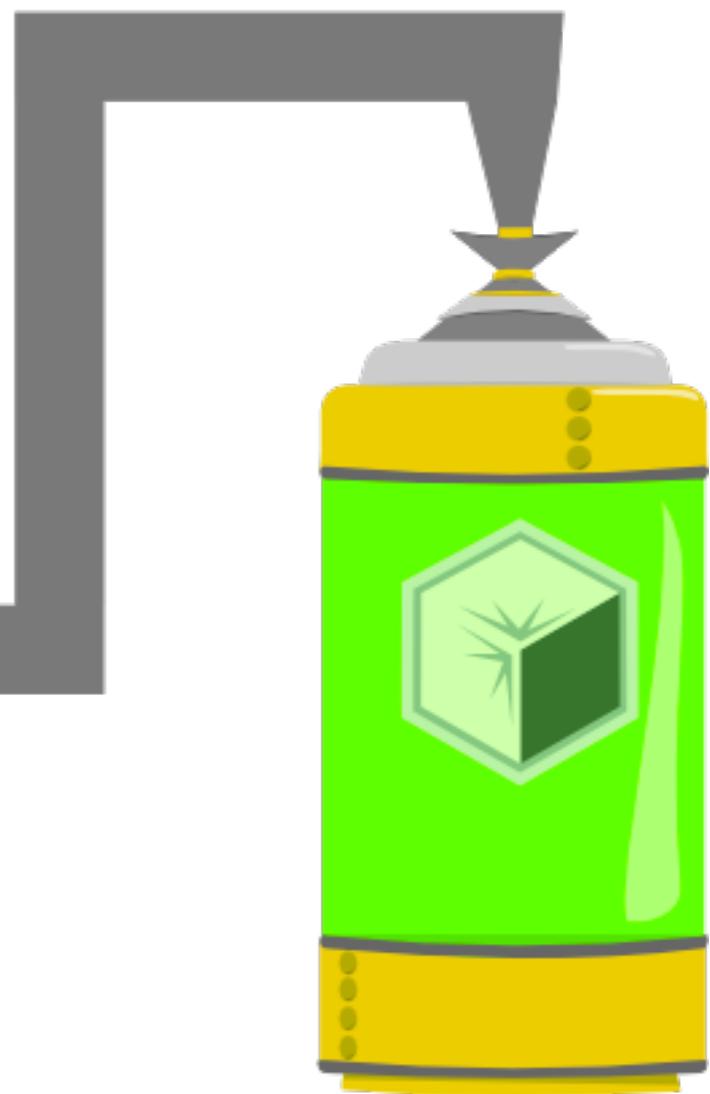
1 Select a container



2 Start or connect to a container

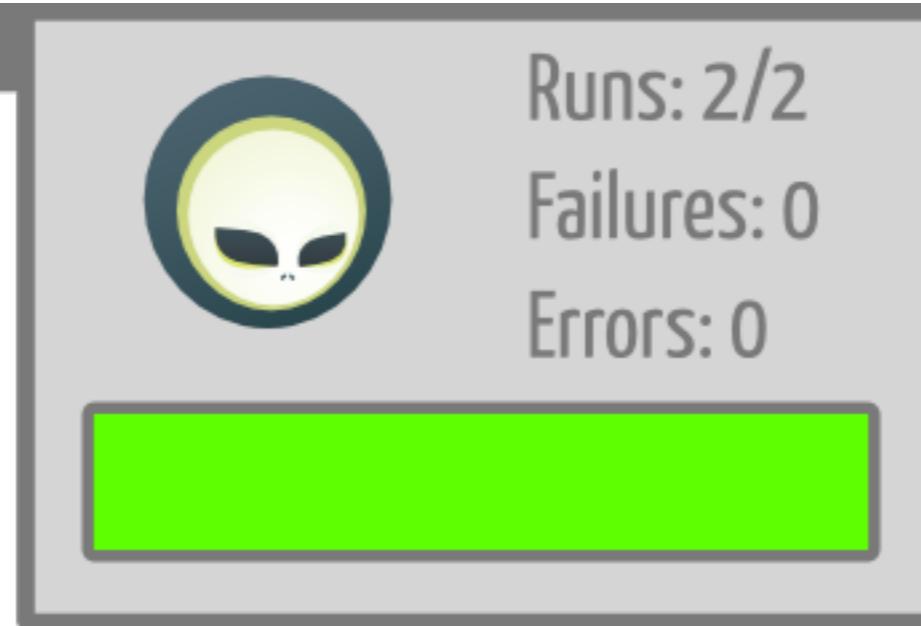


③ Package test archive
and deploy to container

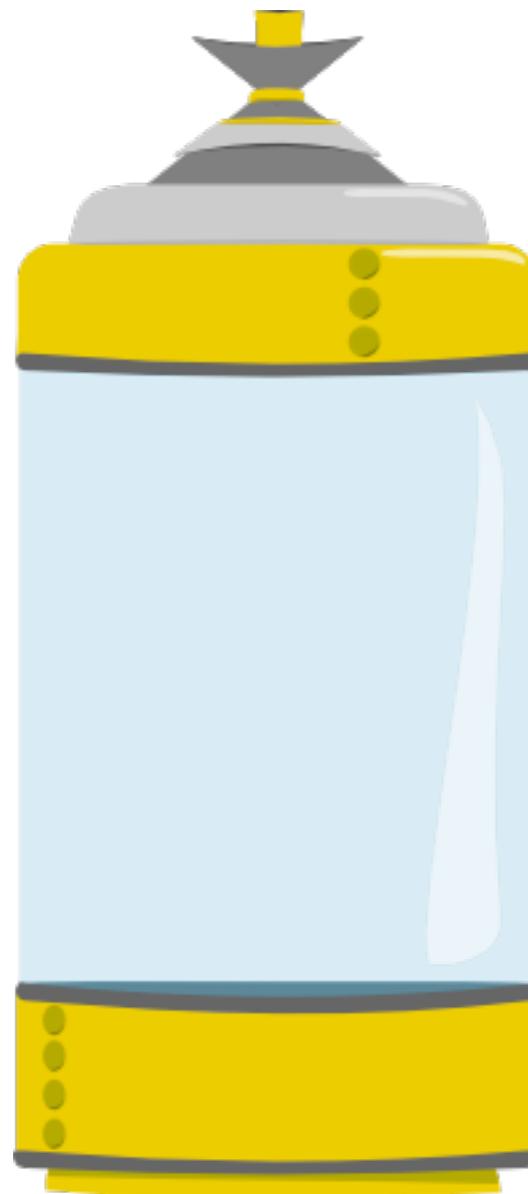


④ Run test in-container





⑤ Capture and report test results



- ⑥ Undeploy test archive and stop or disconnect from container

Getting Started with Arquillian

- Testable Java EE 7 application

```
mvn  
--batch-mode  
archetype:generate  
-DarchetypeGroupId=org.javaee-samples  
-DarchetypeArtifactId=javaee7-arquillian-archetype  
-DgroupId=org.samples.javaee7.arquillian  
-DartifactId=arquillian
```

Testable Java EE 7 App

- Generated profiles
 - wildfly-remote-arquillian
 - wildfly-managed-arquillian
 - glassfish-remote-arquillian
 - glassfish-embedded-arquillian

Test on WildFly

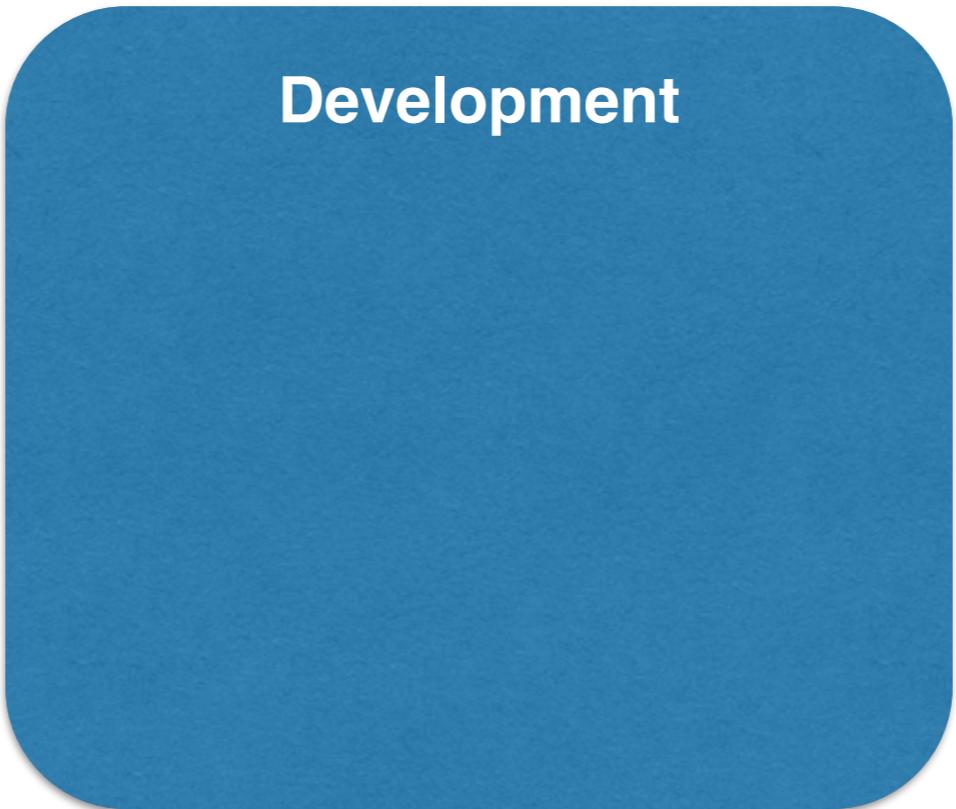
- Download WildFly 8.1 from wildfly.org
- Unzip and start as `bin/standalone.sh`
- Run the test as:

`mvn test -Pwildfly-remote-arquillian`

Enable Arquillian on Existing Java EE Projects

- Using Forge
 - addon-install-from-git --url <https://github.com/forge/addon-arquillian.git>
 - arquillian-setup --testFramework junit --containerAdapter wildfly-remote

Stage 1: Test Driven Development



Development

Stage 1: Test Driven Development



Stage 1: Test Driven Development



Stage 1: Test Driven Development



Stage 1: Test Driven Development



Code review
Code coverage
APM

Stage 2: Continuous Integration

Development



Stage 2: Continuous Integration

Development



Stage 2: Continuous Integration

Development



JUnit

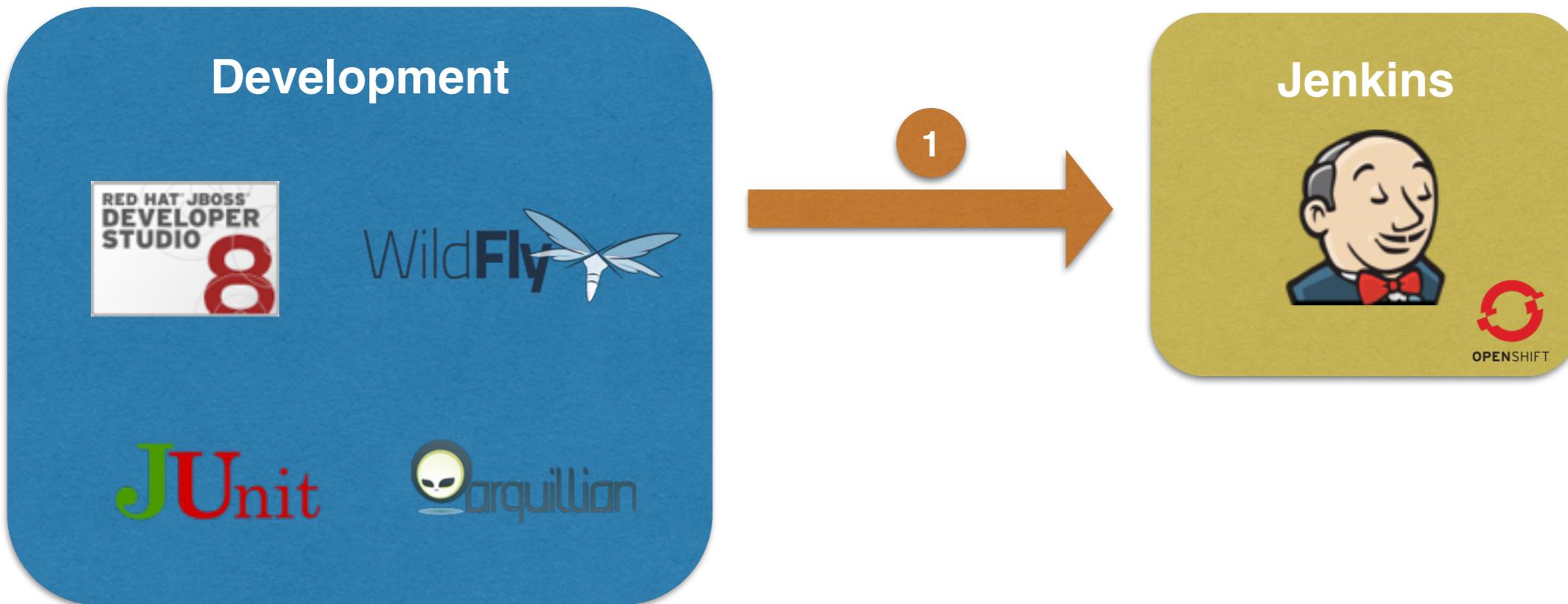


Jenkins



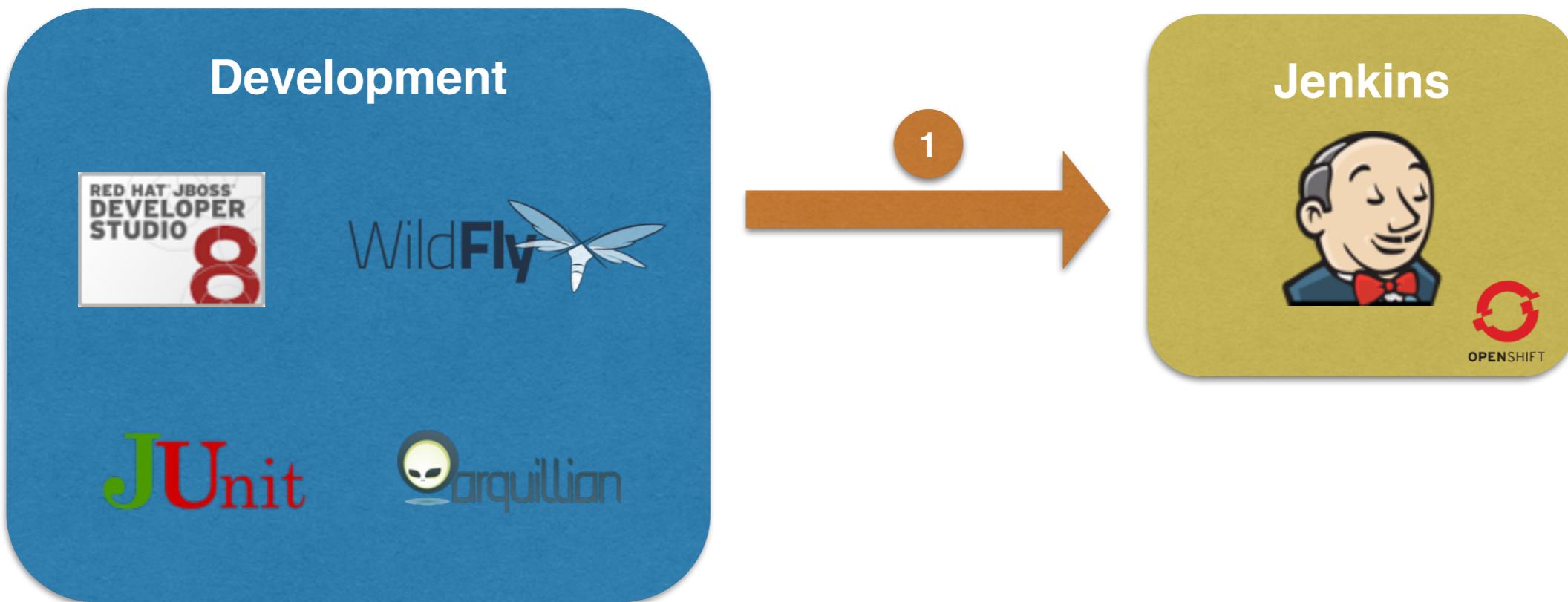
OPENSIFT

Stage 2: Continuous Integration



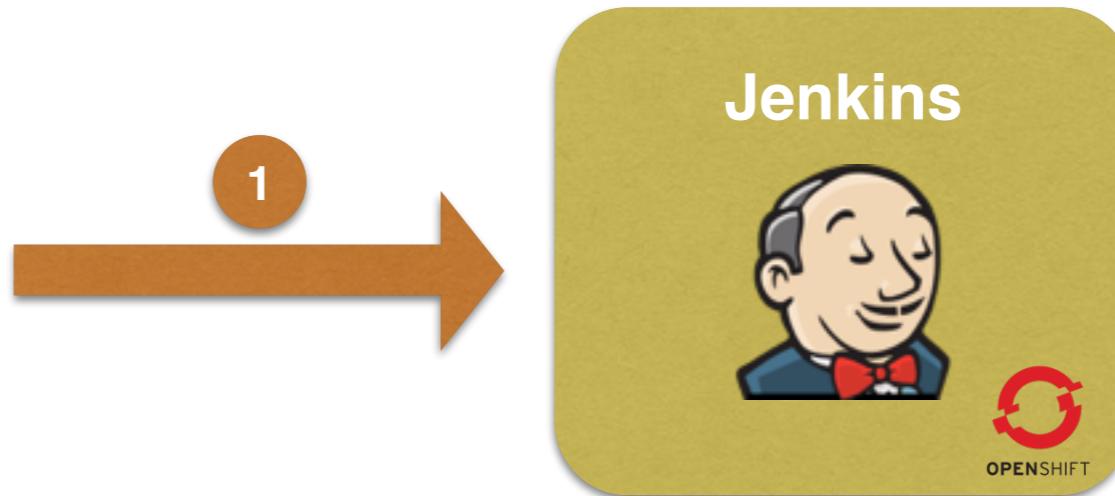
- 1 Started by git push

Stage 3: Automated



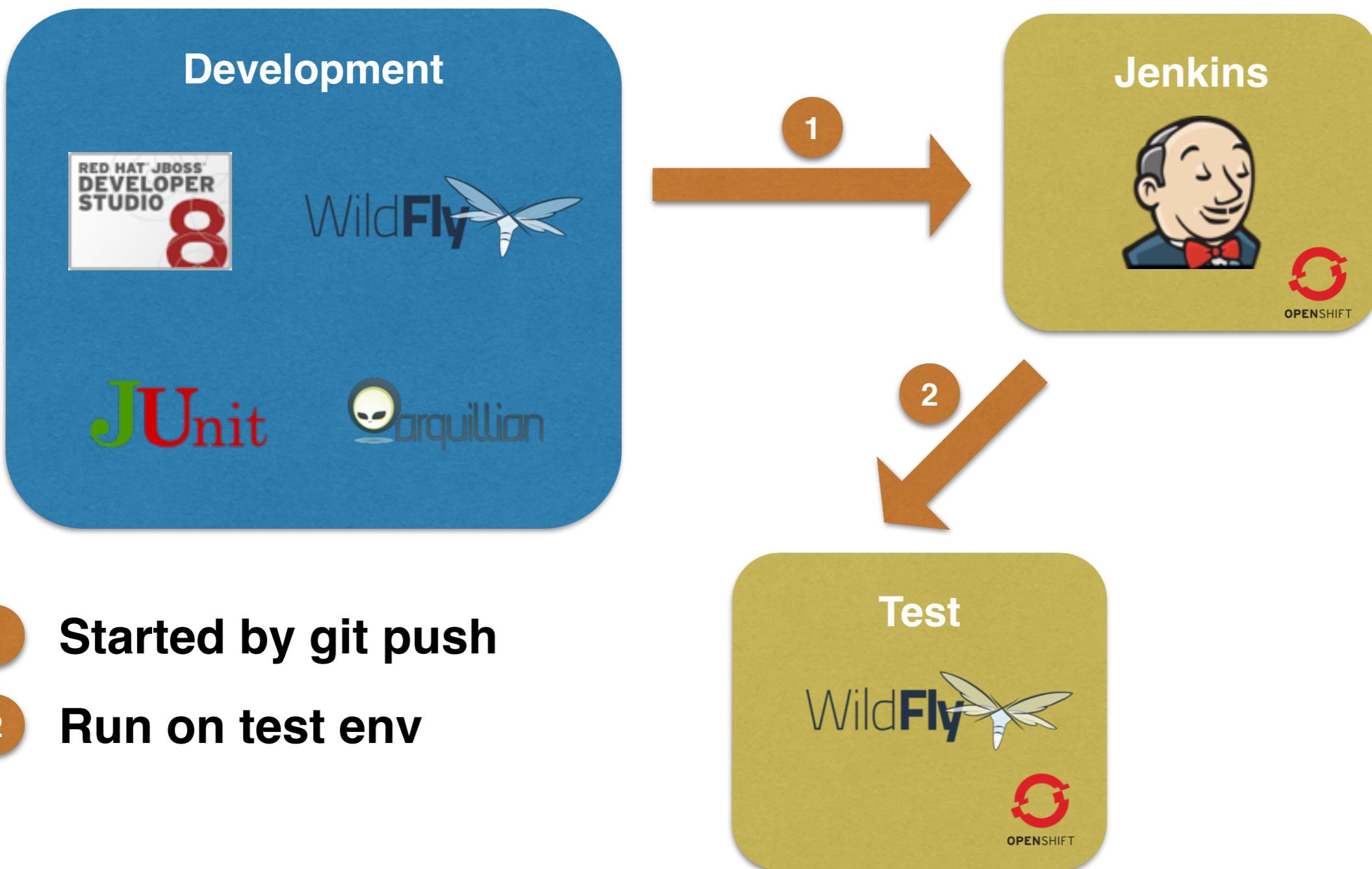
- 1 Started by git push

Stage 3: Automated

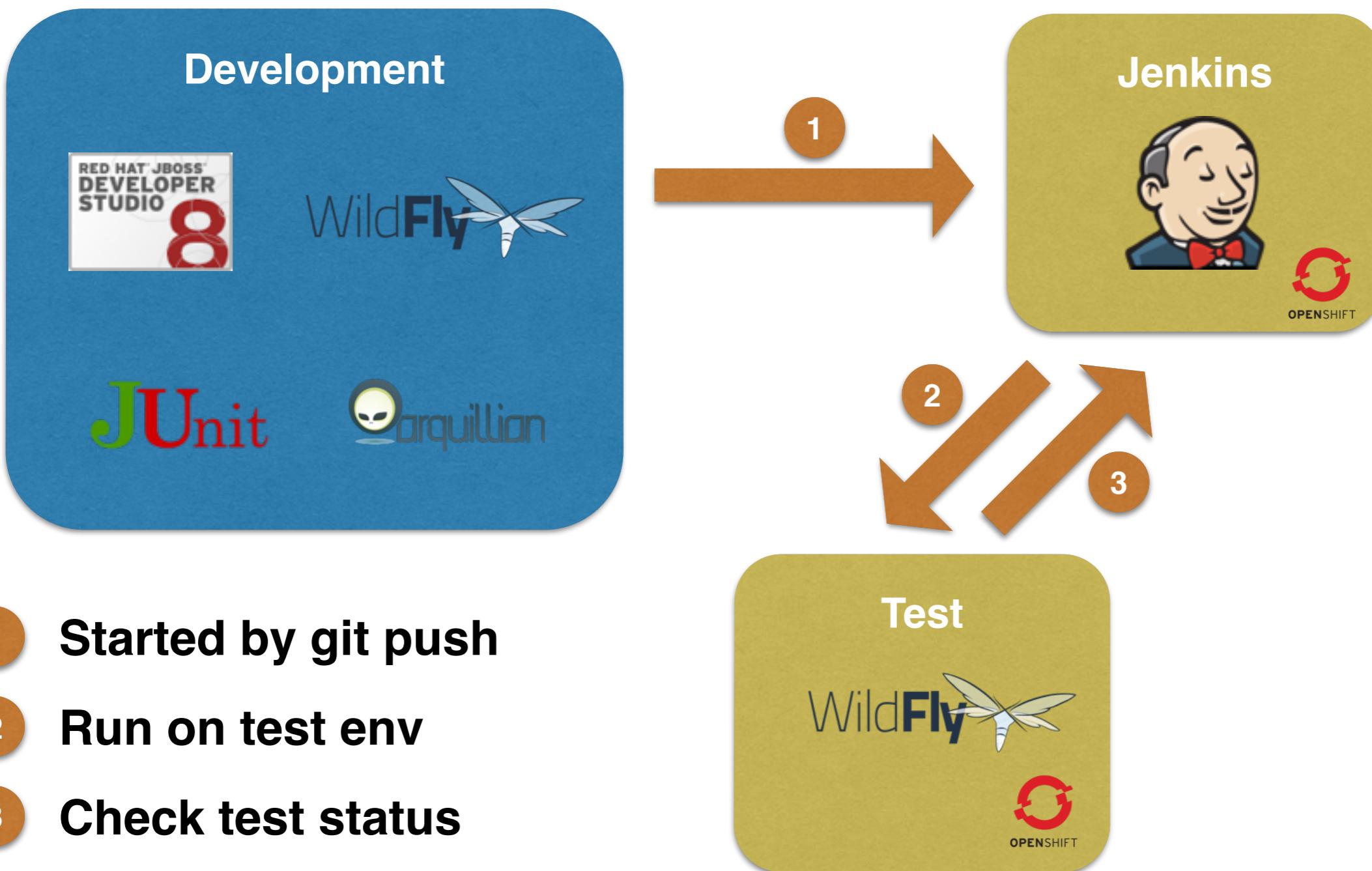


1 Started by git push

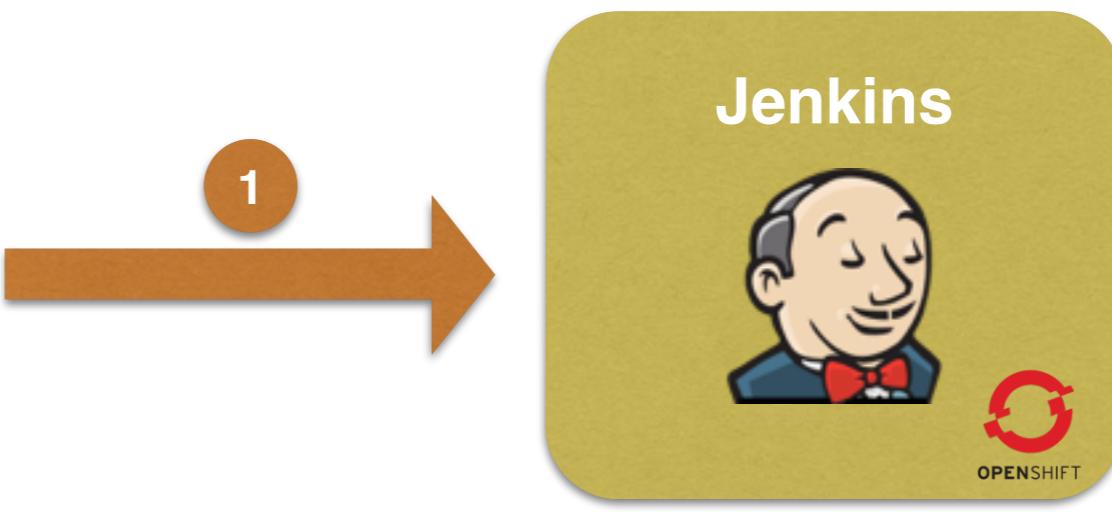
Stage 3: Automated



Stage 3: Automated

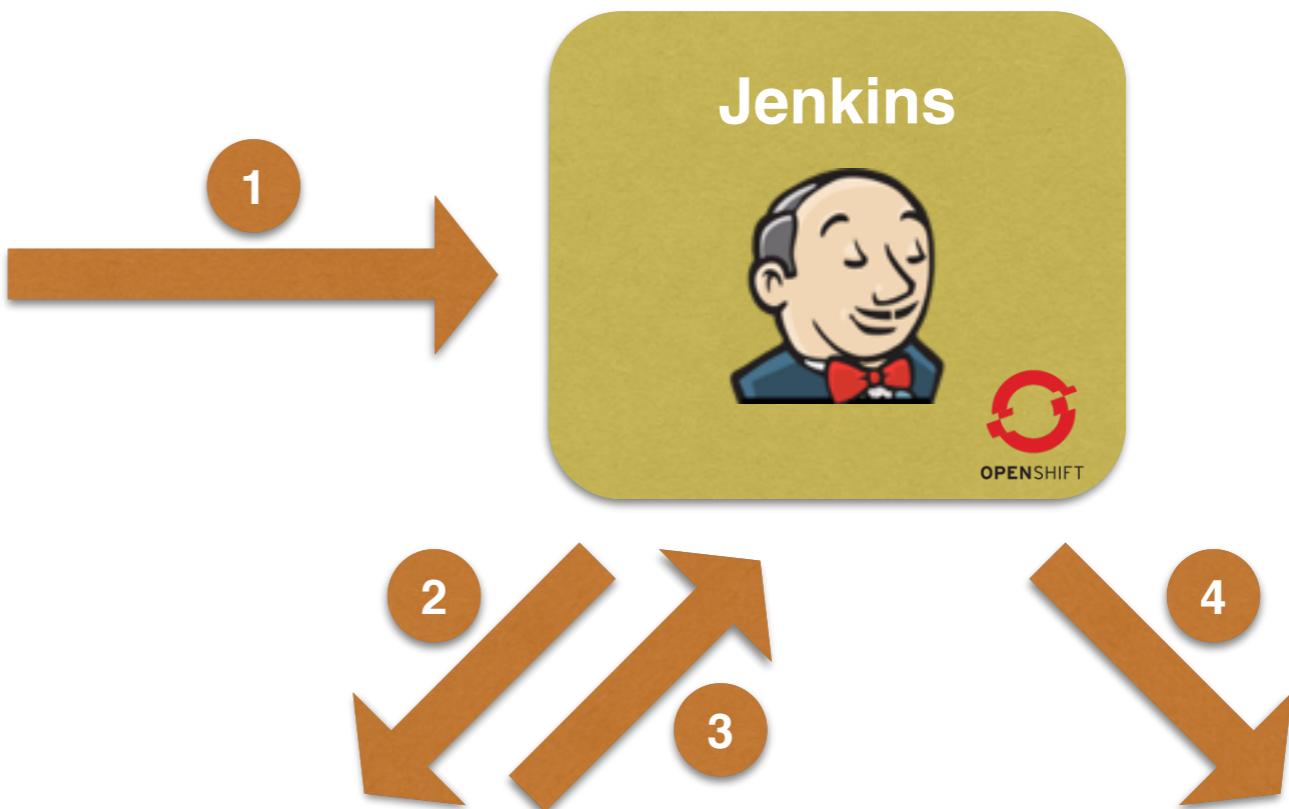


Stage 3: Automated



- 1 Started by git push
- 2 Run on test env
- 3 Check test status

Stage 3: Automated



- 1 Started by git push
- 2 Run on test env
- 3 Check test status
- 4 Push changes to prod



References

- github.com/arun-gupta/devops