

Deployment Pipelines

for Java Apps

using OpenShift

Arun Gupta, Red Hat

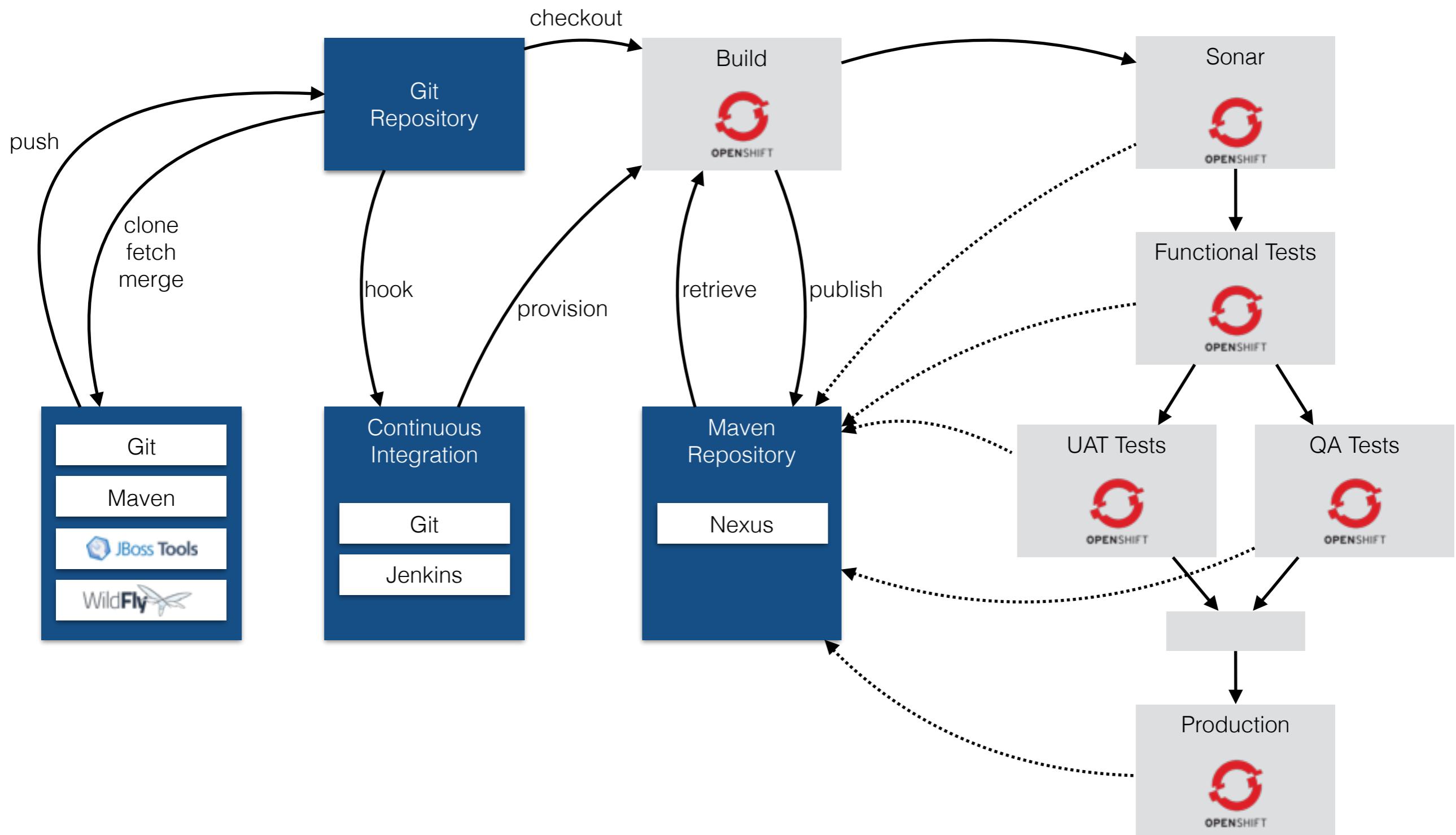


Arun Gupta

Director, Developer Advocacy & Technical Marketing

@arungupta
blog.arungupta.me
arungupta@redhat.com

Deployment Pipeline with PaaS





**WORKED FINE IN
DEV**

OPS PROBLEM NOW

Is DevOps for you?

Ship code 30x faster

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

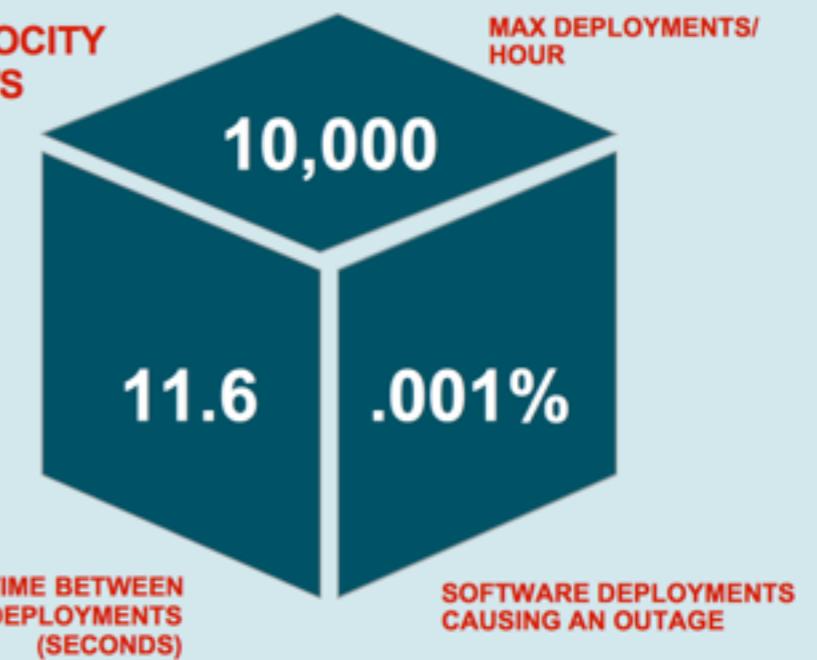
Have 50% fewer failures

and restore service 12 times faster than their peers.

Organizations implementing DevOps

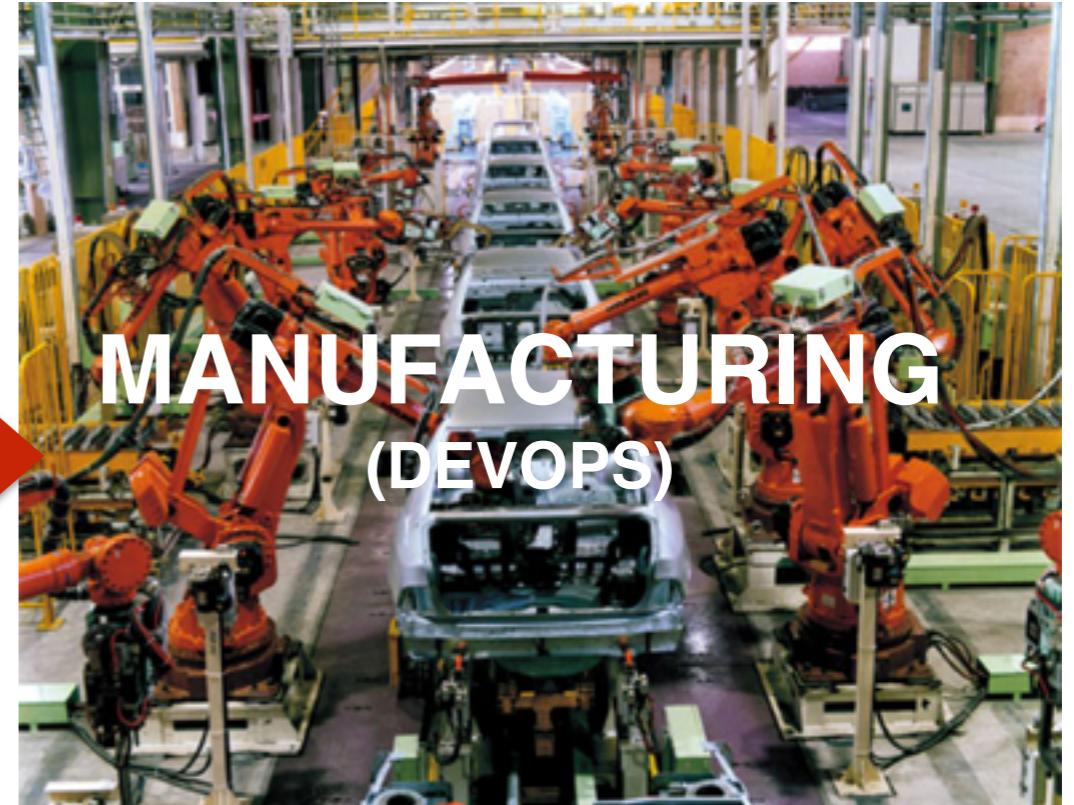


DEVOPS VALUE
IN ACTION: VELOCITY
AT AMAZON AWS





CRAFTWORK



MANUFACTURING
(DEVOPS)

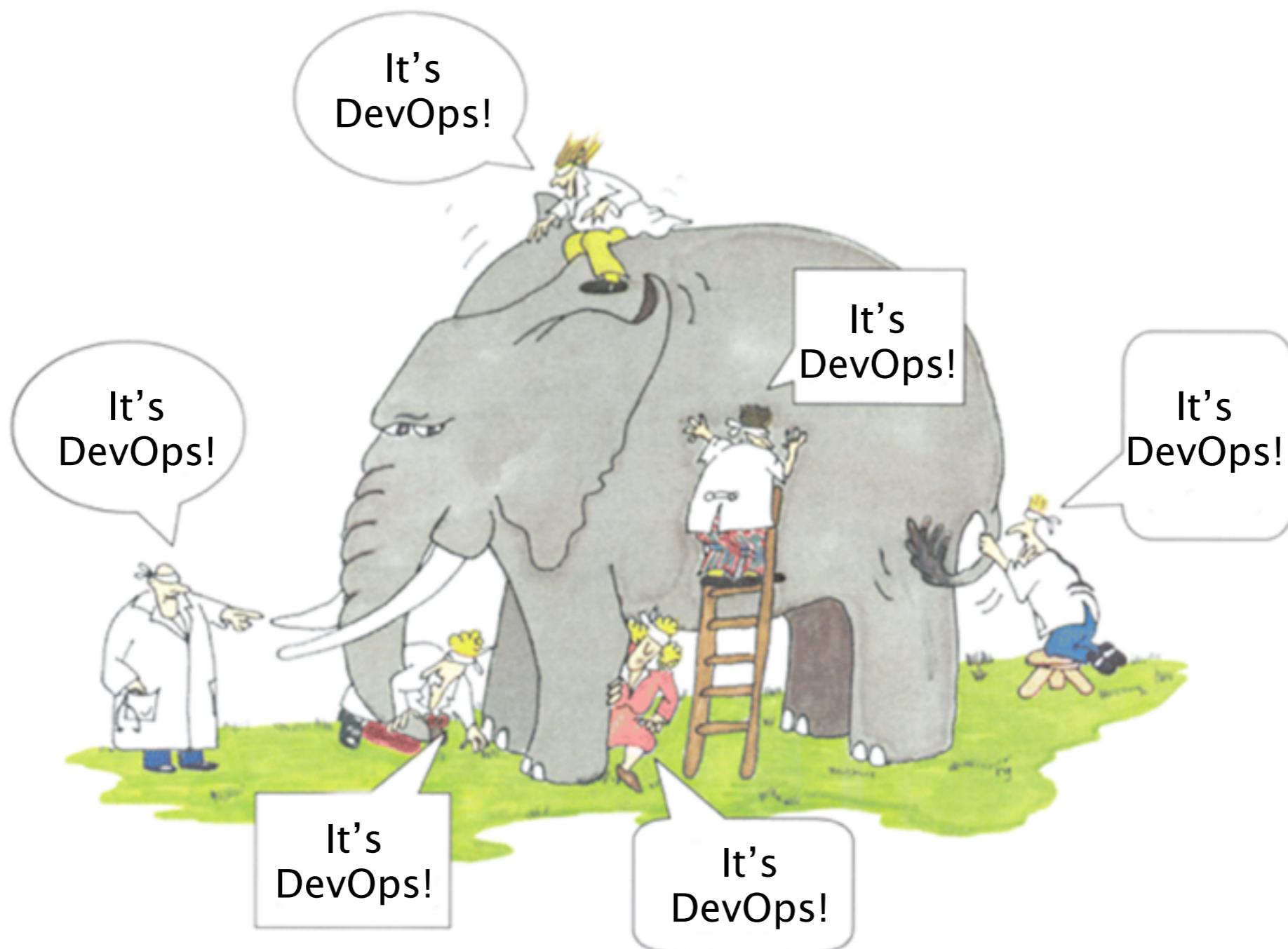


WORKSHOP



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

Key Components of DevOps

- **People:** “Dev” and “Ops”
- **Processes:** Continuous Delivery, Agile, ...
- **Tools:** Jenkins, Chef, Sonar, Arquillian, PaaS, ...

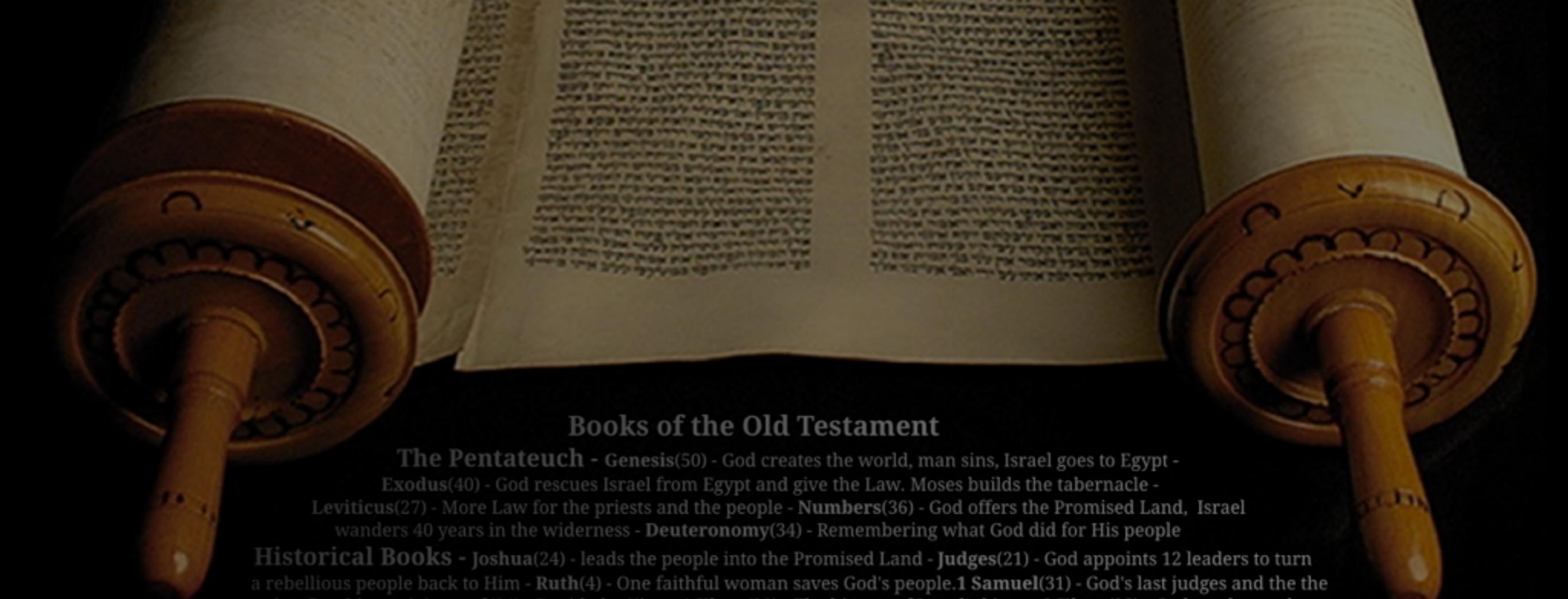
Five “C”s of DevOps

- **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





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 lessons 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-destruction - 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 to enable business

And so is “devs”

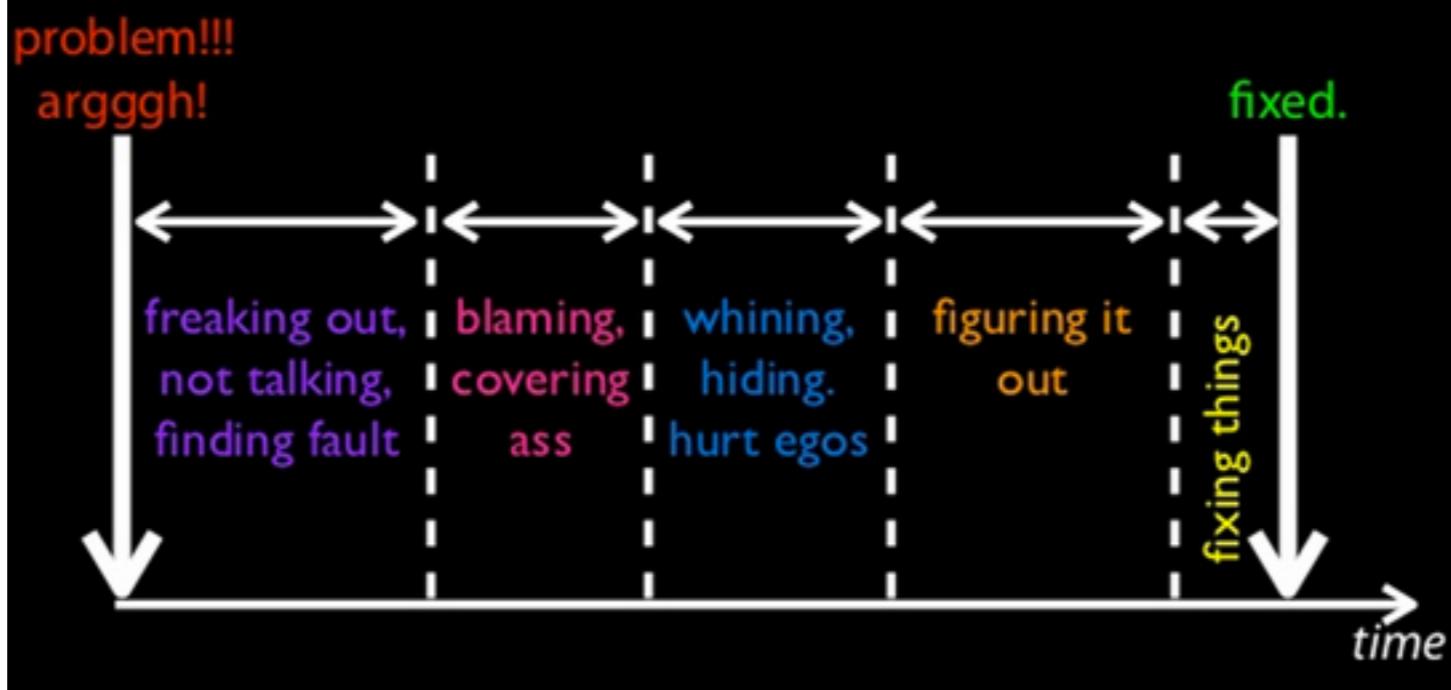


“Dev” “Ops”

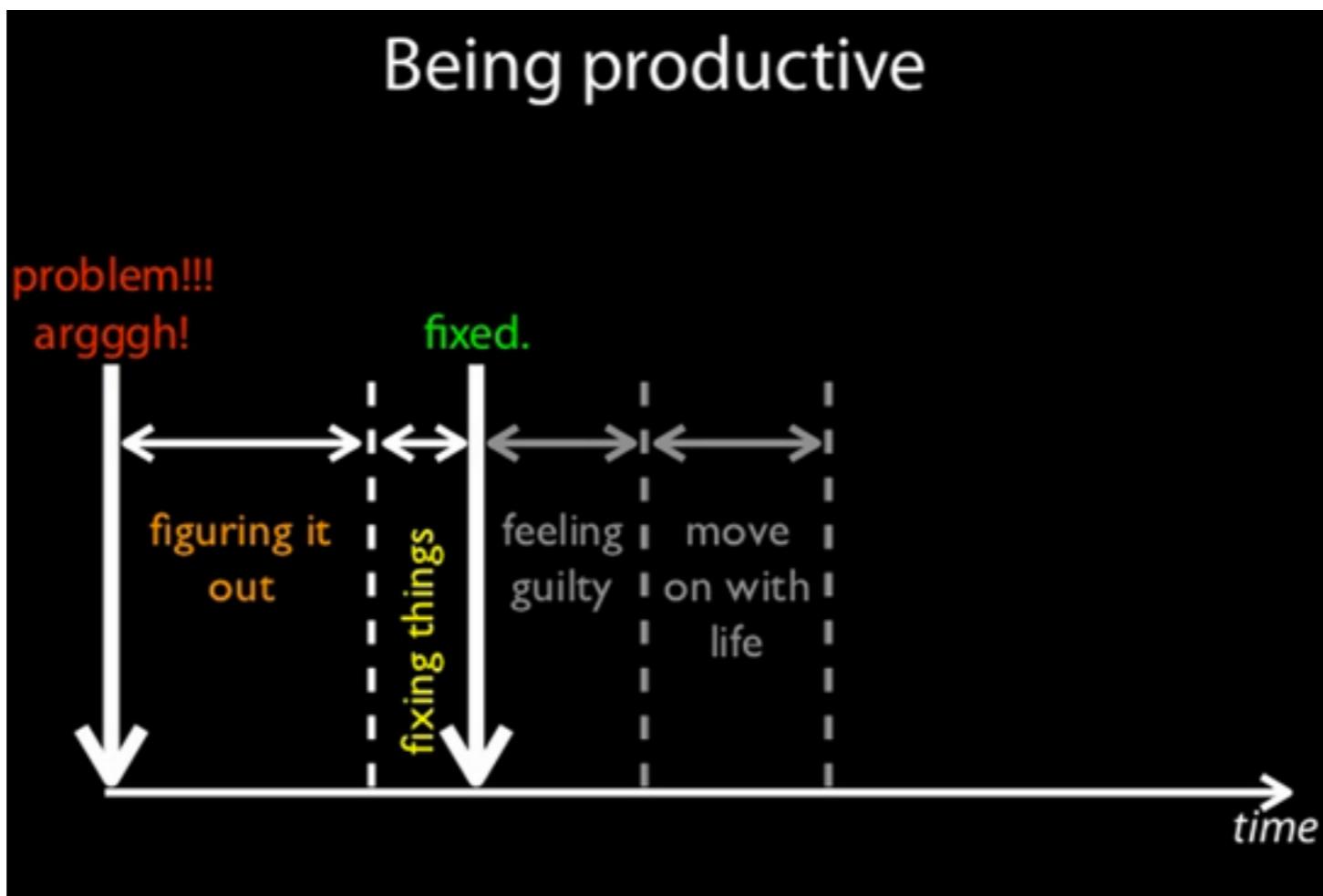
Culture

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

Fingerpointyness



Being productive



“Treat people warmly, issues coldly!”



With great
power, comes great
responsibility



“you build it, you run it!”

Code everything

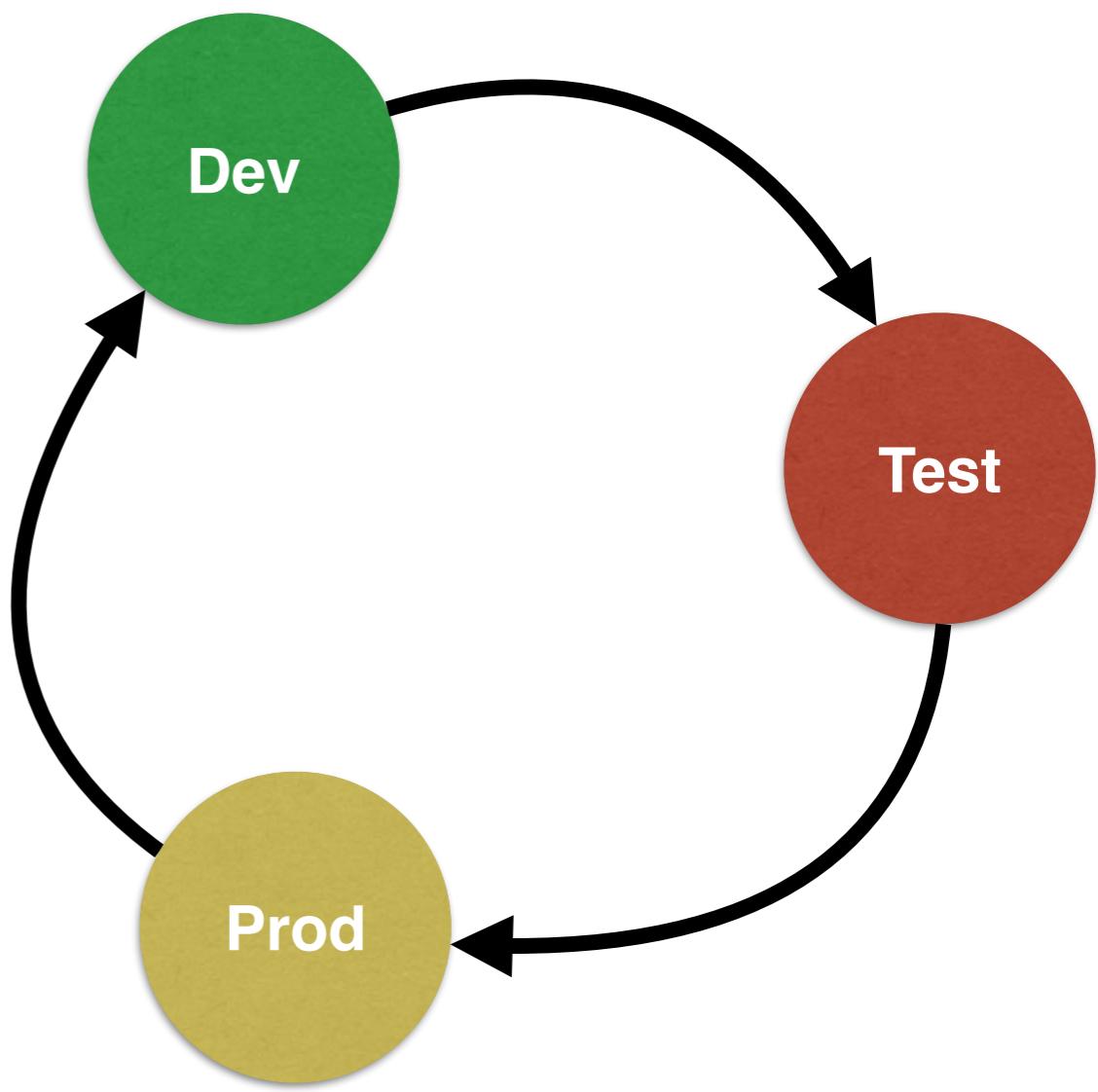
- Application code
- Build scripts
- Database schema
- Configuration files
- IDE configurations
- Infrastructure
- Deployment scripts
- Test code and scripts
- Provisioning scripts
- Monitoring
- Logging
- ...

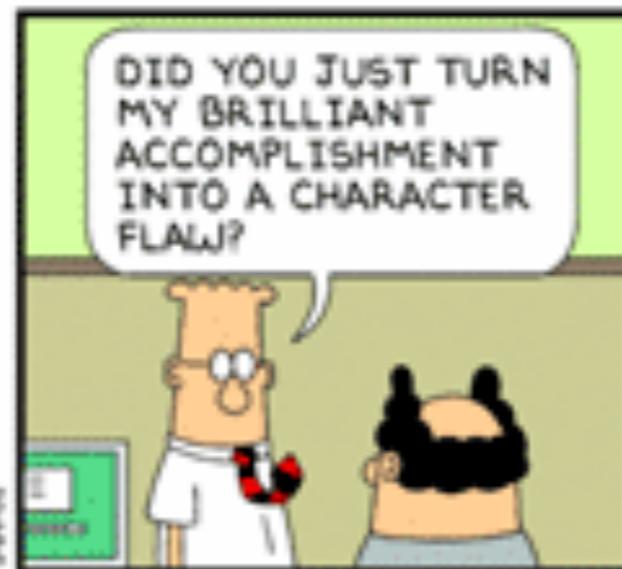
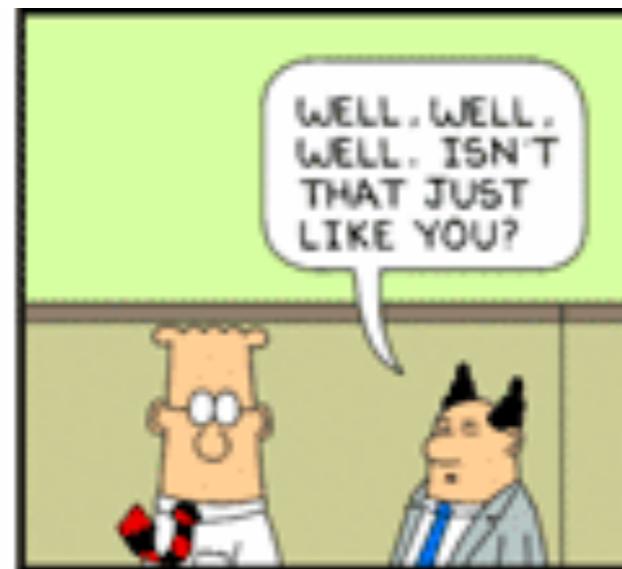


“Never send a human to do a machine’s job”

Consistency

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



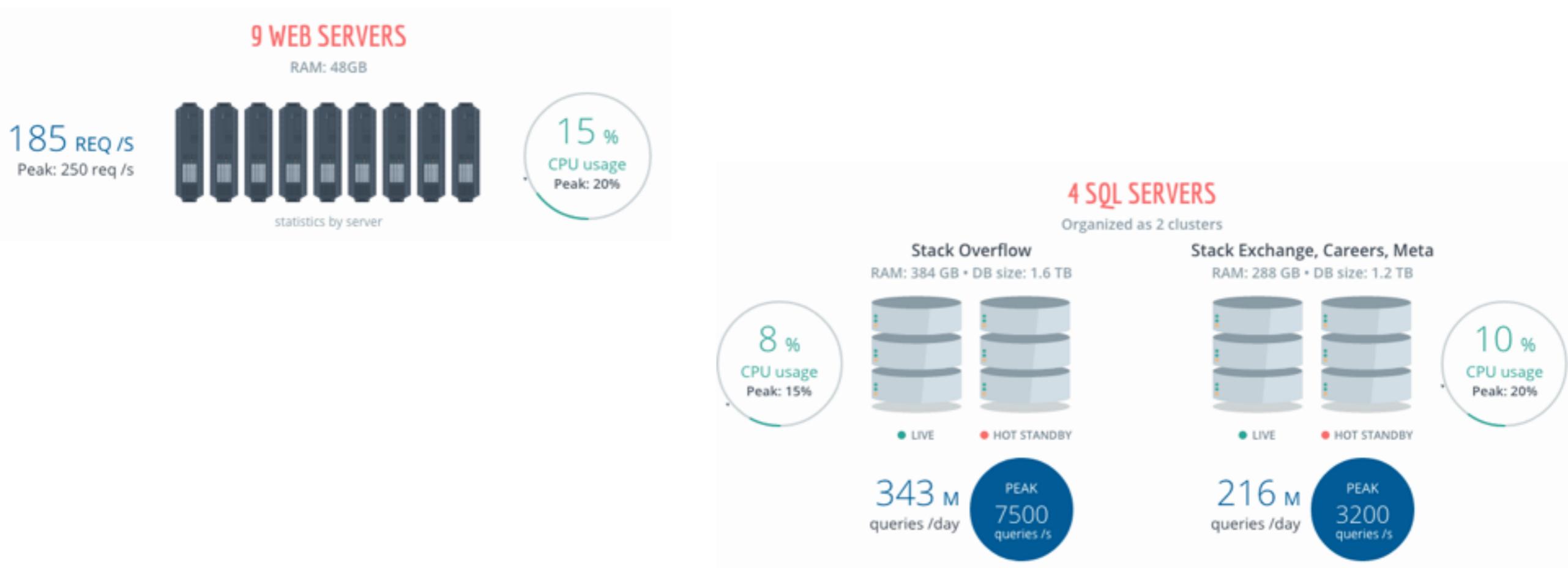


Manage environments

- **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, ...

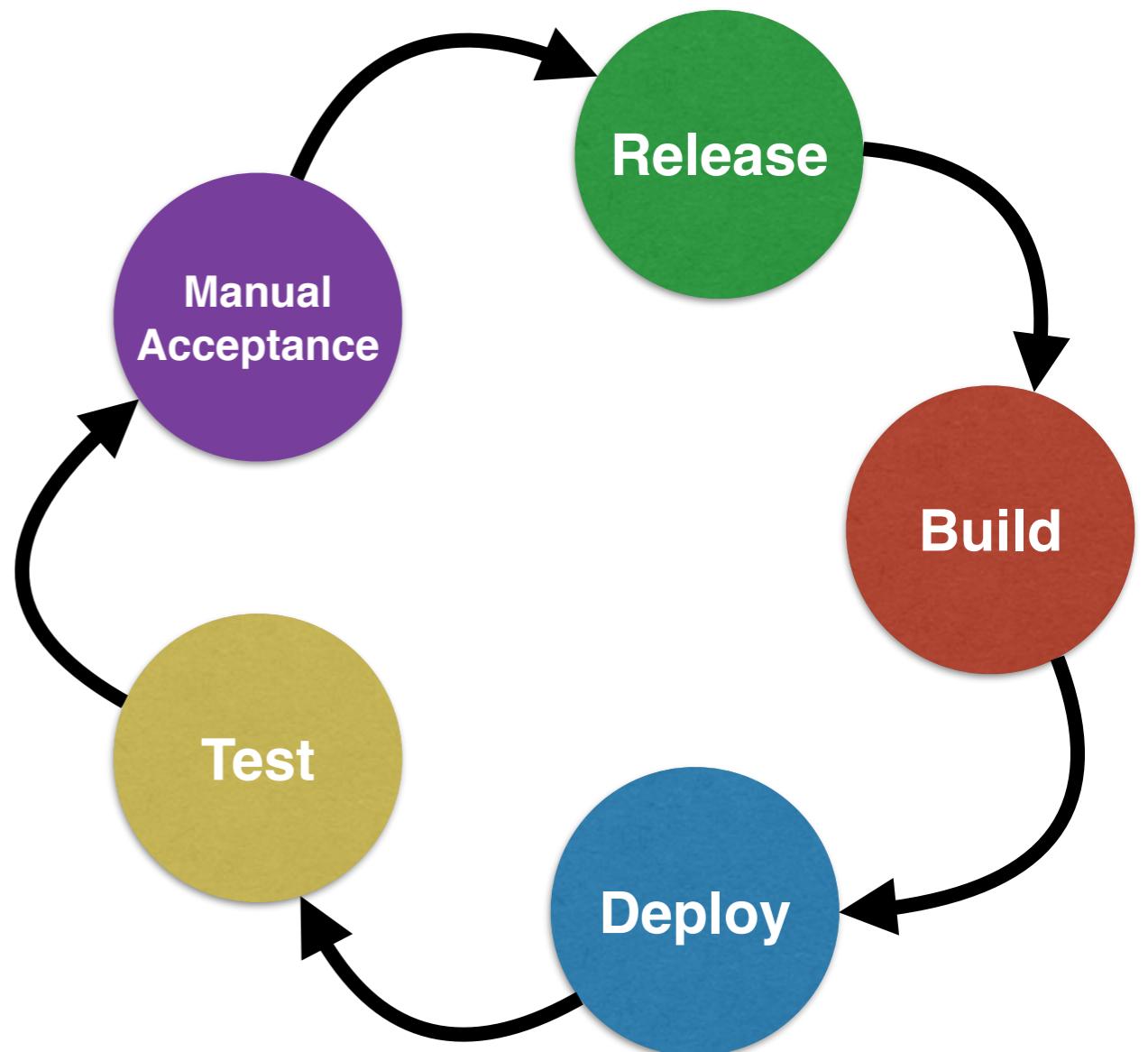
Dashboards

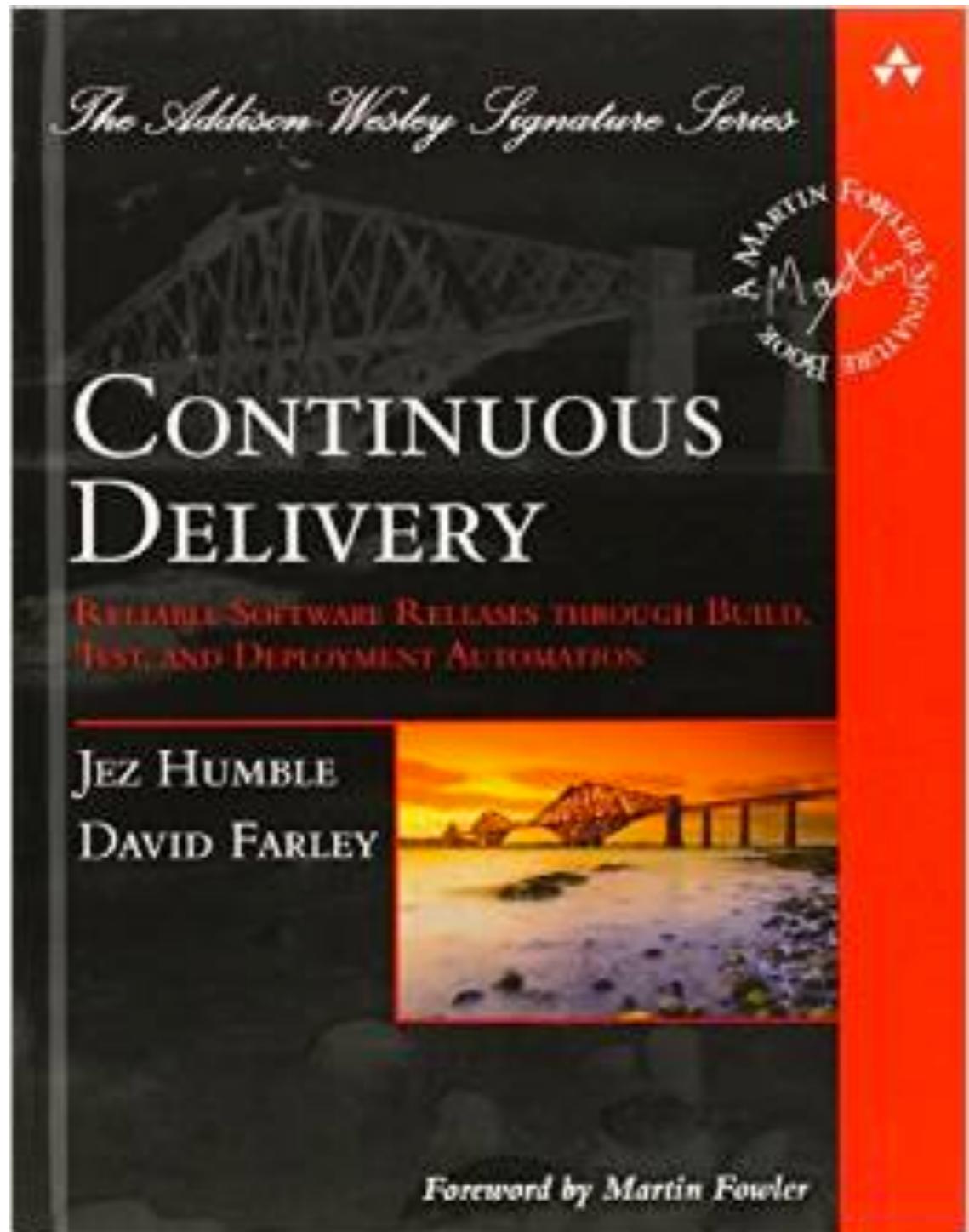
- Build dashboards, improve transparency
- stackexchange.com/performance



Continuous Delivery

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



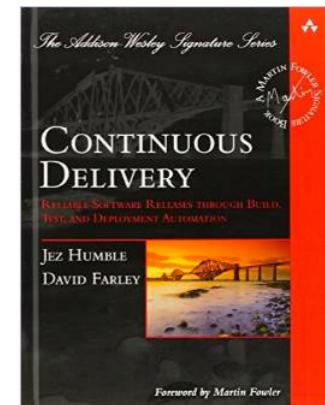


*“it is the practice
of releasing
every good build
to users”*

*“continuous
integration to its
logical
conclusion”*

Deployment Pipeline

*“an automated implementation of
your application’s build, deploy,
test, and release process”*



| | Initial | Managed | Defined | Quantitatively Managed | Optimizing |
|-------------------------|--|---|---|---|--|
| Culture & Organization | <ul style="list-style-type: none"> Teams organized based on platform/technology Defined and documented processes | <ul style="list-style-type: none"> One backlog per team Adopt agile methodologies Remove team boundaries | <ul style="list-style-type: none"> Extended team collaboration Remove boundary dev/ops Common process for all changes | <ul style="list-style-type: none"> Cross-team continuous improvement Teams responsible all the way to production | <ul style="list-style-type: none"> Cross functional teams |
| Build & Deploy | <ul style="list-style-type: none"> Centralized version control Automated build scripts No management of artifacts Manual deployment Environments are manually provisioned | <ul style="list-style-type: none"> Polling CI builds Any build can be re-created from source control Management of build artifacts Automated deployment scripts Automated provisioning of environments | <ul style="list-style-type: none"> Commit hook CI builds Build fails if quality is not met (code analysis, performance, etc.) Push button deployment and release of any releasable artifact to any environment Standard deployment process for all environments | <ul style="list-style-type: none"> Team priorities keeping codebase deployable over doing new work Builds are not left broken Orchestrated deployments Blue Green Deployments | <ul style="list-style-type: none"> Zero touch Continuous Deployments |
| Release | <ul style="list-style-type: none"> Infrequent and unreliable releases Manual process | <ul style="list-style-type: none"> Painful infrequent but reliable releases | <ul style="list-style-type: none"> Infrequent but fully automated and reliable releases in any environment | <ul style="list-style-type: none"> Frequent fully automated releases Deployment disconnected from release Canary releases | <ul style="list-style-type: none"> No rollbacks, always roll forward |
| Data Management | <ul style="list-style-type: none"> Data migrations are performed manually, no scripts | <ul style="list-style-type: none"> Data migrations using versioned scripts, performed manually | <ul style="list-style-type: none"> Automated and versioned changes to datastores | <ul style="list-style-type: none"> Changes to datastores automatically performed as part of the deployment process | <ul style="list-style-type: none"> Automatic datastore changes and rollbacks tested with every deployment |
| Test & Verification | <ul style="list-style-type: none"> Automated unit tests Separate test environment | <ul style="list-style-type: none"> Automatic Integration Tests Static code analysis Test coverage analysis | <ul style="list-style-type: none"> Automatic functional tests Manual performance/security tests | <ul style="list-style-type: none"> Fully automatic acceptance tests Automatic performance/security tests Manual exploratory testing based on risk based testing analysis | <ul style="list-style-type: none"> Verify expected business value Defects found and fixed immediately (roll forward) |
| Information & Reporting | <ul style="list-style-type: none"> Baseline process metrics Manual reporting Visible to report runner | <ul style="list-style-type: none"> Measure the process Automatic reporting Visible to team | <ul style="list-style-type: none"> Automatic generation of release notes Pipeline traceability Reporting history Visible to cross-silo | <ul style="list-style-type: none"> Report trend analysis Real time graphs on deployment pipeline metrics | <ul style="list-style-type: none"> Dynamic self-service of information Customizable dashboards Cross-reference across organizational boundaries |

Tools

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



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



Barack Obama

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

Like

▼



Chartbeat **Nagios**



ubuntu®

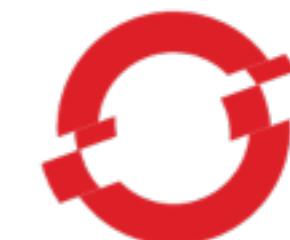


 Scott VanDenPlas
@scottvdp

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

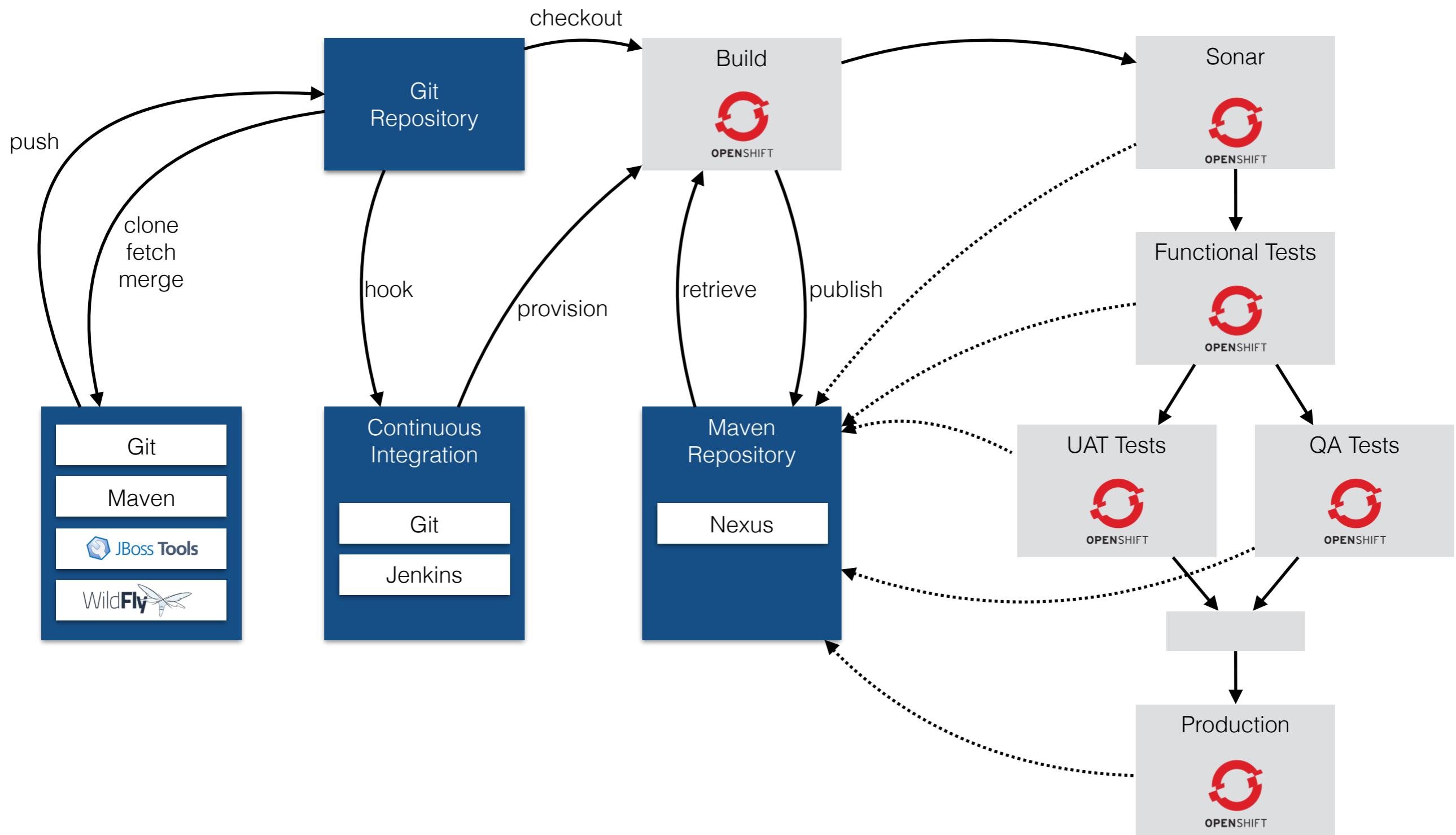
Tools for DevOps with Java EE



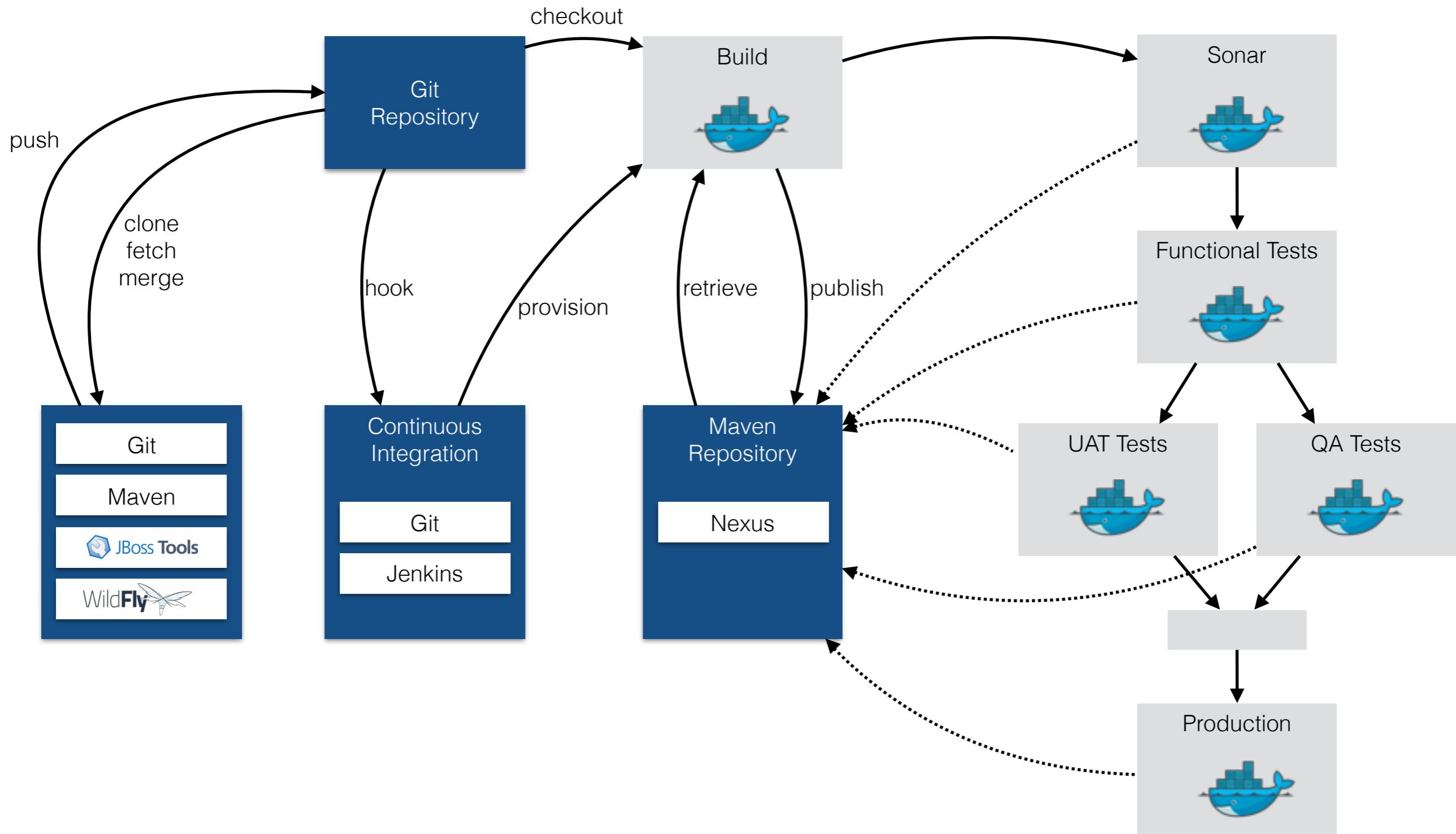
maven



Deployment Pipeline with PaaS



Deployment Pipeline with Containers



Build Server

Ticket Monster

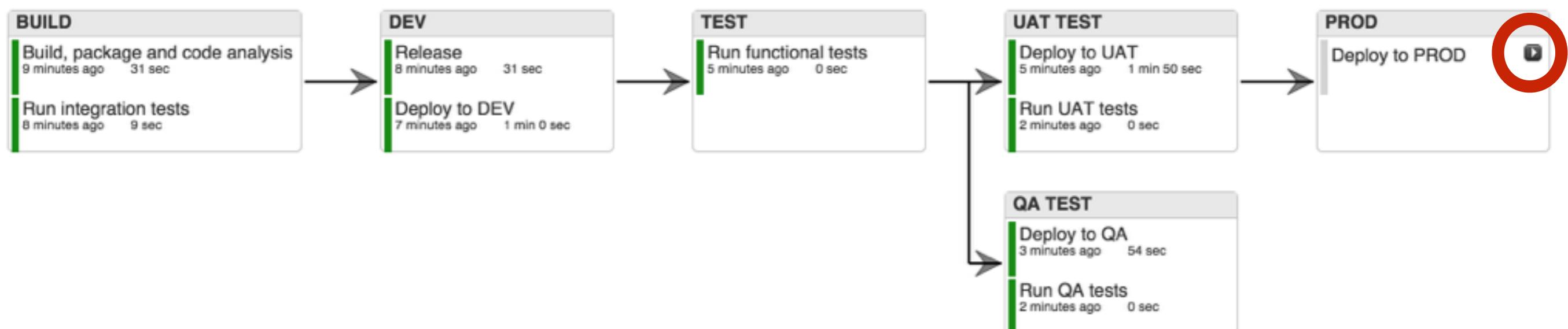
#57 triggered by user jenkins started 4 minutes ago



UAT and QA Tests

Ticket Monster

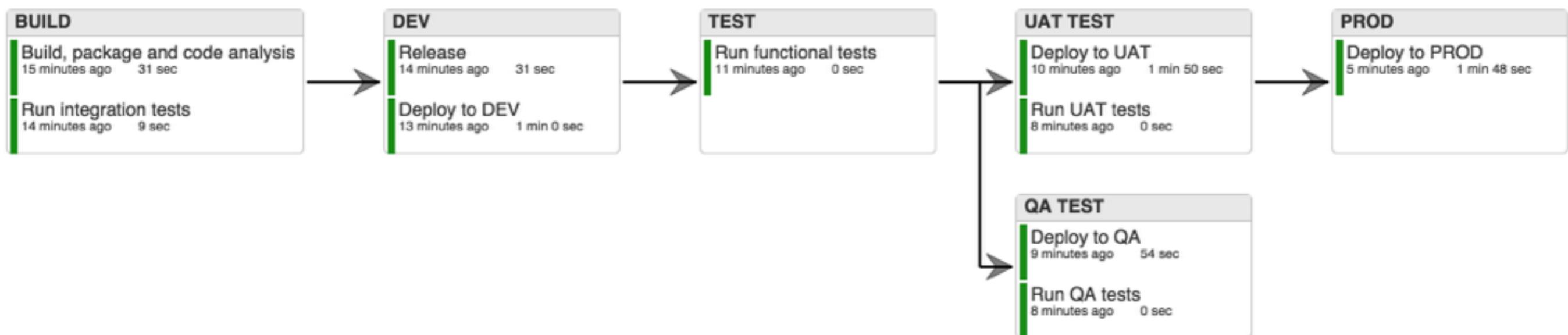
#57 triggered by user jenkins started 9 minutes ago



Deployed to Production

Ticket Monster

#57 triggered by user jenkins started 15 minutes ago



Failed Tests

Ticket Monster

#53 triggered by user jenkins started 16 minutes ago



References

- github.com/arun-gupta/devops-javaee