



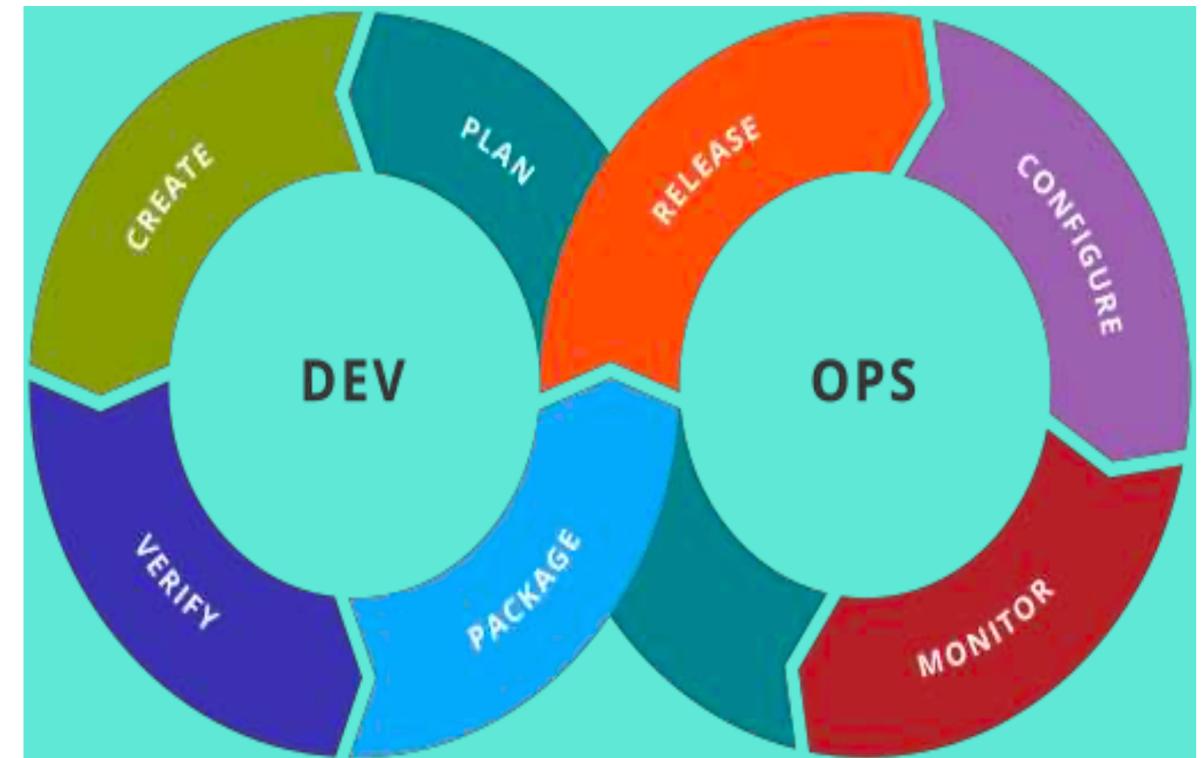
DevOps

Software Engineering

*Most of the slides are freely taken from Slideshare
We are thankful to authors for their work and sharing of material and
knowledge*

In a word

- In simple words the DevOps means “it is a combination of software development and operations”
- DevOps is characterized by operations staff making use many of the same techniques as developers for their systems work



Current Business Problems & Symptoms

Problems in a Nutshell

- ✓ Need more time to respond to market changes
- ✓ Deployments held off to avoid risk
- ✓ Slow and error prone releases
- ✓ Fix and maintain rather than innovate
- ✓ Unstable operations as fixes take more time
- ✓ IT is frequently seen as the bottleneck in the transition of “concept to cash.”

Symptoms

- ✓ Works on my machine / environment
- ✓ Need prod environment access to diagnose issues
- ✓ Servers not available for deployment
- ✓ Deployment failed due to incorrect configuration
- ✓ Lets fix it after this big event / day
- ✓ “Manual error” is a commonly cited root cause
- ✓ Releases slip / fail

So, Why DevOps?

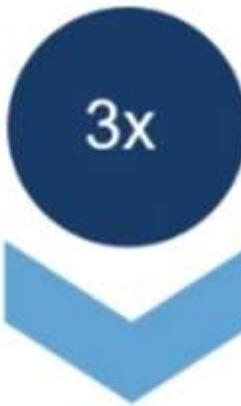
High-performing IT organizations report experiencing:



200x more frequent deployments



24x faster recovery from failures



3x lower change failure rate



2,555x shorter lead times



2.2x higher employee Net Promoter Score

Faster
Better
Cheaper
Happier



Customer Satisfaction



Source: <https://puppet.com/resources/whitepaper/2016-state-of-devops-report>

Why DevOps? Put Simply...

Increase Velocity.

Reduce Downtime.

Reduce Human Error.

Your competition is already doing this.

What DevOps is Not / Common Myths

DevOps is 100% end to end Automation

DevOps is all about tools

DevOps involves only Development and Operations

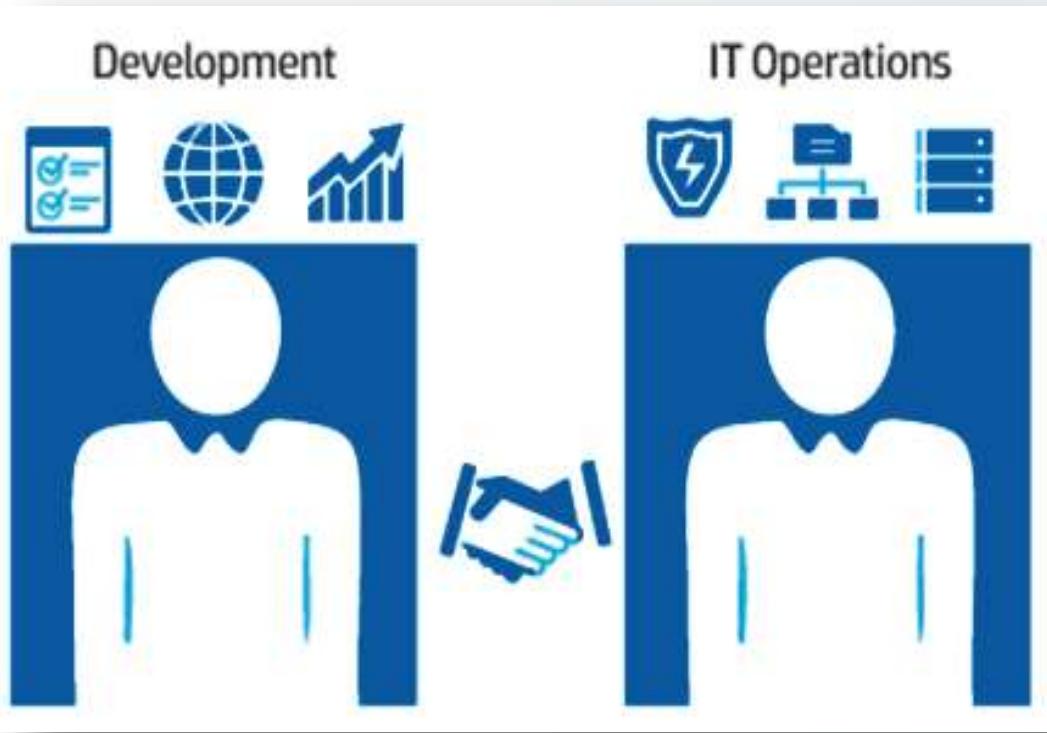
There is only “One Way” to do DevOps

DevOps is about reducing the staff by half

DevOps only works well with Startups

What is DevOps?

Take 1 of 3



Change / Modify /
Test Features

Enhance Stability /
Services

With DevOps, Everyone is Responsible for Quality

QUALITY

Moving from...

- Single person responsibility
- Individualism
- Departmental

...towards

- Collective responsibility
- Shared understanding
- Service delivery

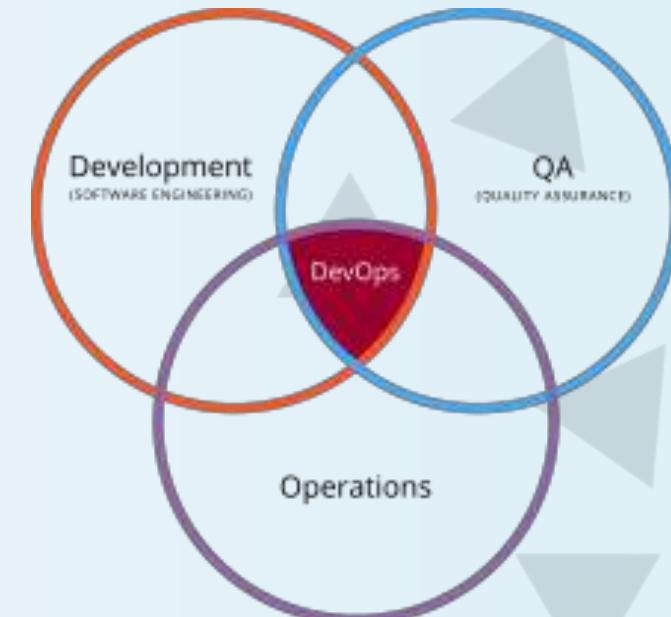
DevOps isn't one team's job. It's *everybody's* job. And DevOps culture is all about shared responsibility. That means a shift toward transparency, communication, and collaboration across development, IT/ops, and "the business".

<https://www.atlassian.com/team-playbook/examples/devops-culture>

What is DevOps?

Take 2 of 3

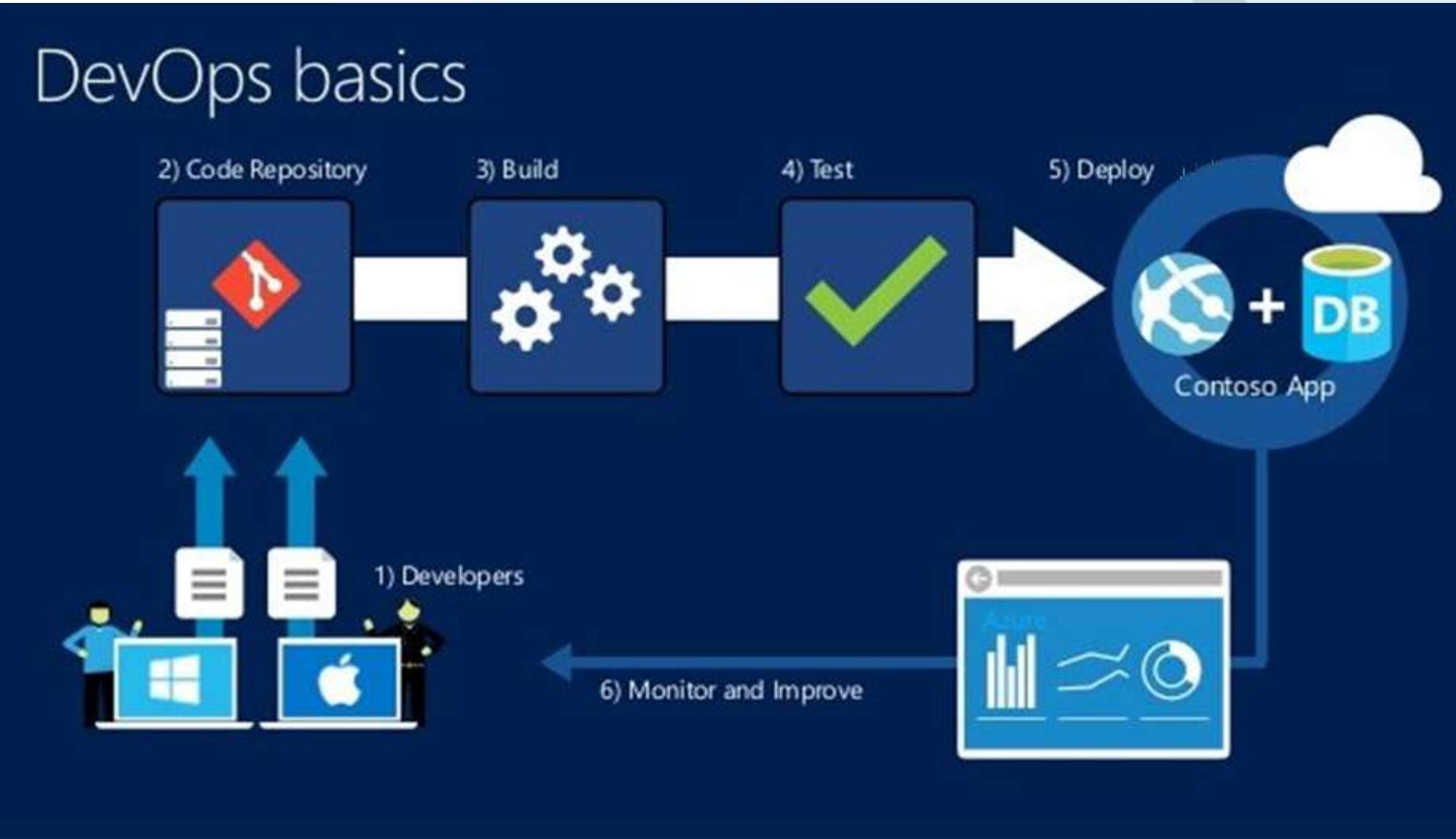
- Set of practices that emphasize the **collaboration** and communication of both software developers and information technology (IT) professionals while **automating** the process of software delivery and infrastructure changes. ([Wikipedia](#))
- Approach to **bridge the gap** between agile software development and operations. ([agileweboperations.com](#))
- Union of **people, process, and products** to enable **continuous delivery** of value to our end users. ([Donovan Brown, Principal DevOps Program Mgr., Microsoft - http://donovanbrown.com/post/what-is-devops](#))
- **Cultural** and operational model that fosters collaboration to enable high performance IT to achieve business goals. ([DASA](#))



Source: Wikipedia

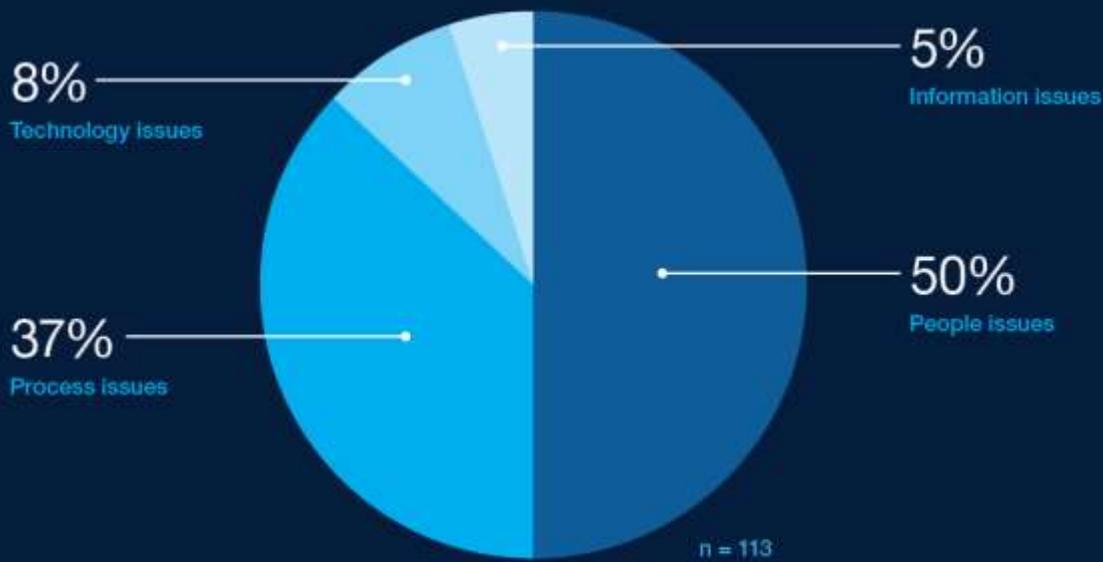
What is DevOps?

Final Take – The Basics



DevOps – Culture and Mind-set

Which of these is the biggest challenge for your organizations' expansion of the use of DevOps?



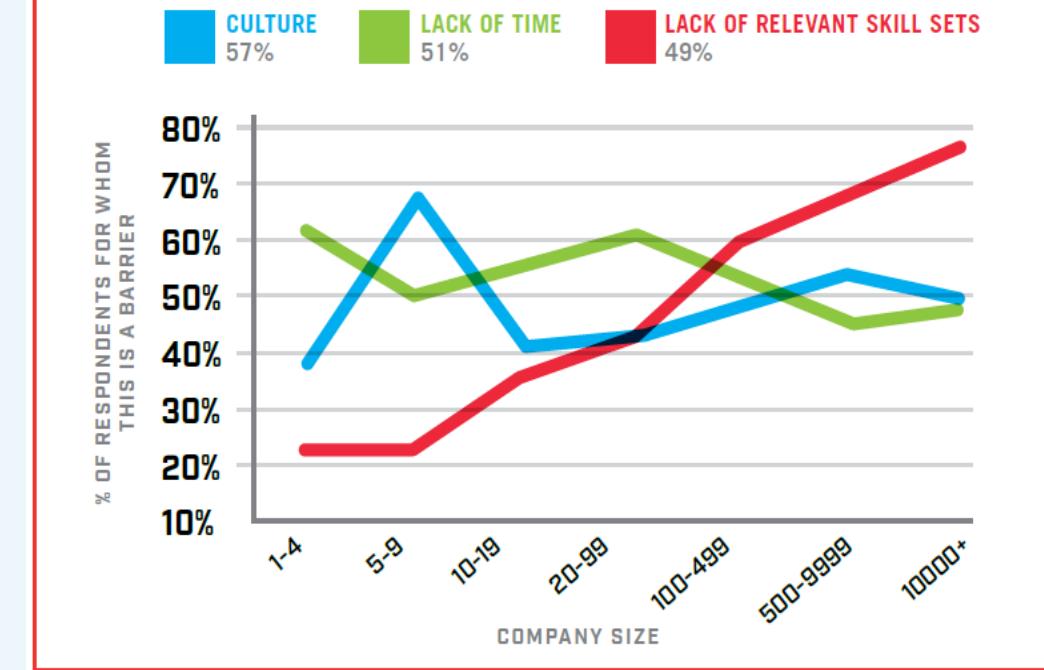
gartner.com/SmarterWithGartner

Source: Gartner (January 2016)

© 2016 Gartner, Inc. and/or its affiliates. All rights reserved. Gartner is a registered trademark of Gartner, Inc. or its affiliates. For more information, email info@gartner.com or visit gartner.com.

Gartner

CD BARRIERS AS PERCENTAGE OF COMPANY SIZE

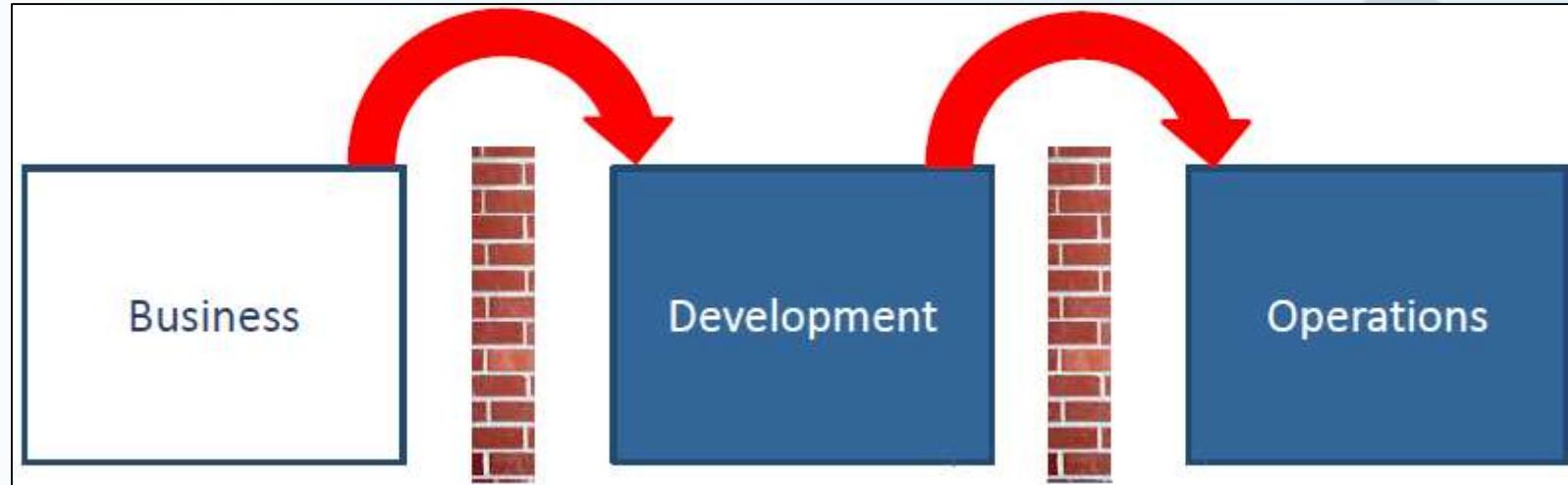


Barrier to DevOps is primarily the Culture.

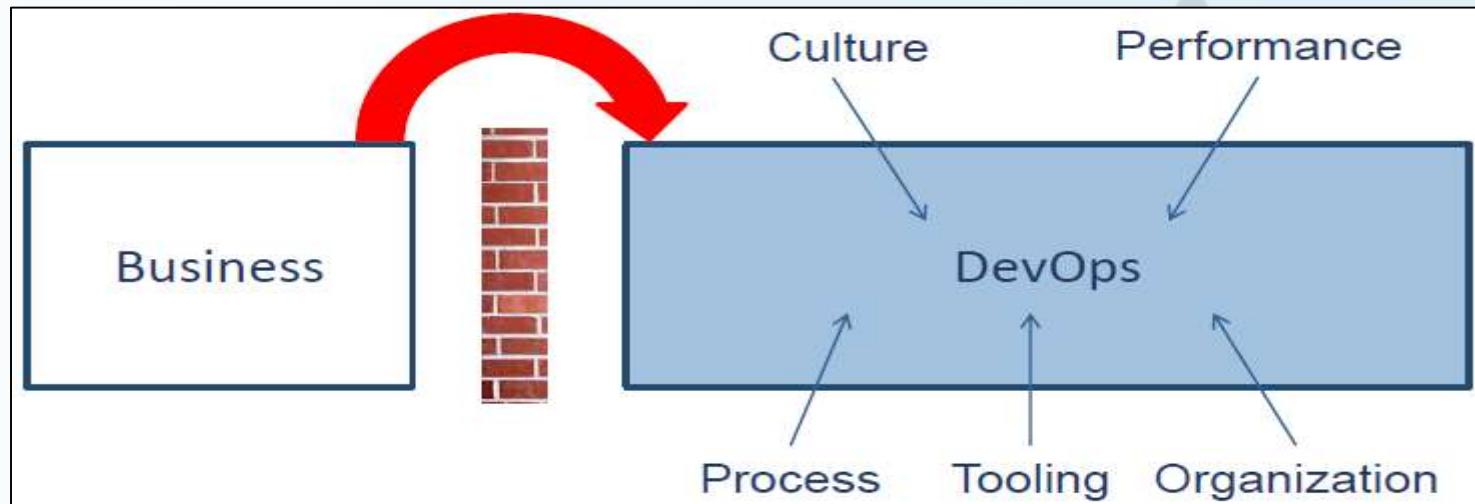
Source: [Dzone Continuous Delivery Ref Card 2016](http://dzone.com/refcardz/continuous-delivery)

DevOps – The Solution

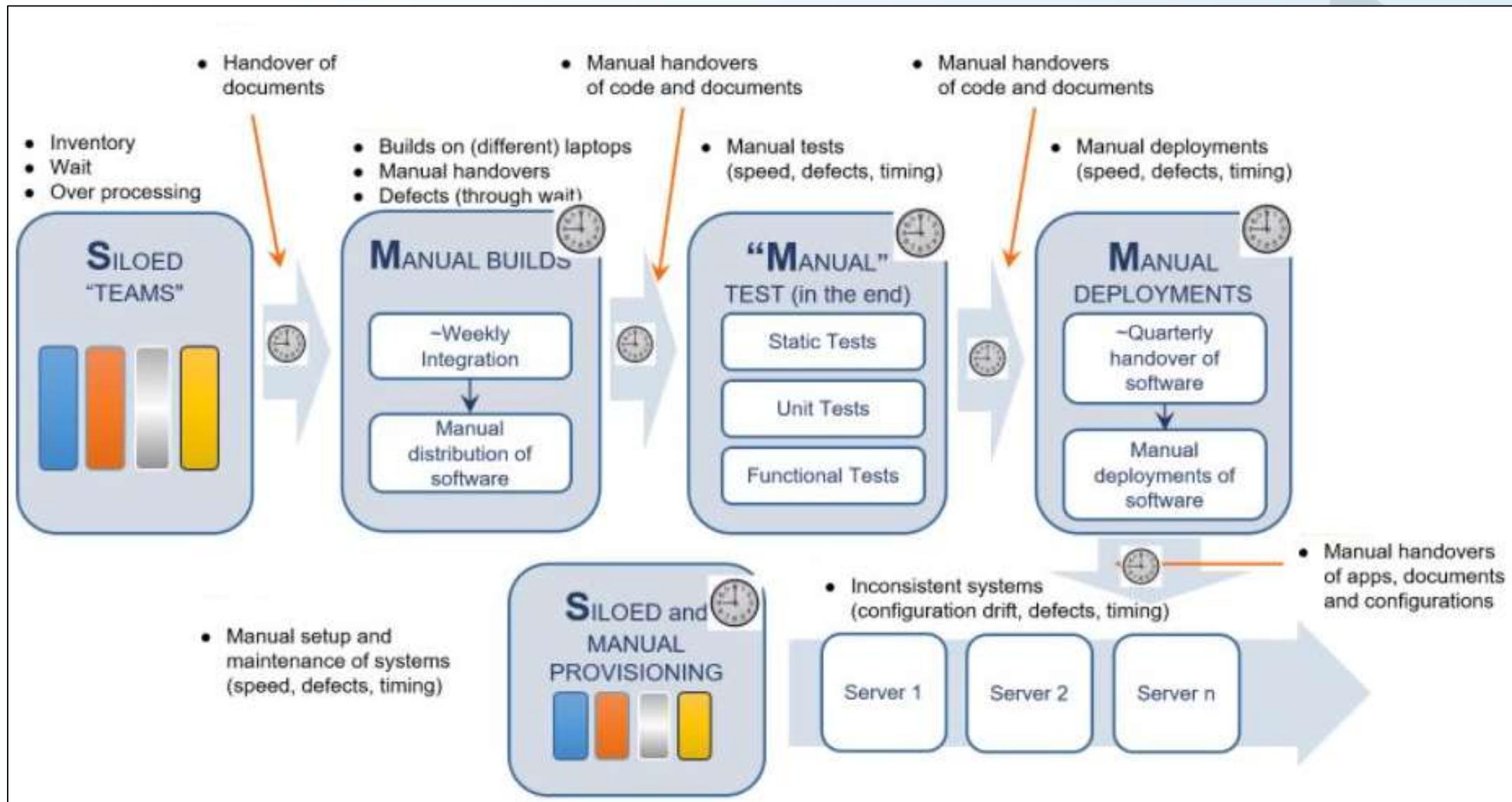
Current Situation



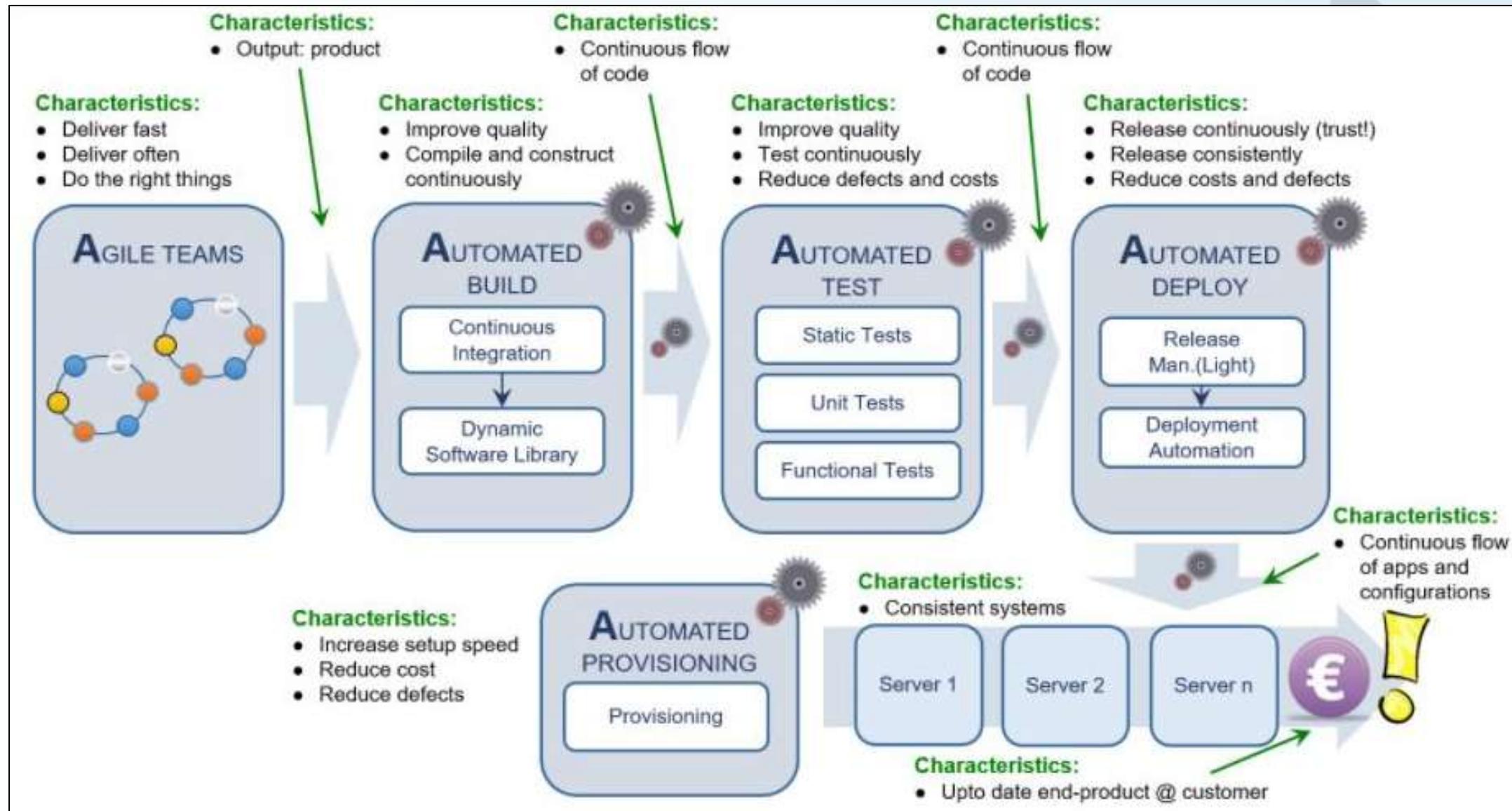
New Situation



Traditional Delivery Cycle



DevOps - Continuous Delivery Cycle



DevOps – Concepts and Jargons

Continuous Integration (CI)

Continuous Delivery (CD)

Automated Testing

Configuration Management

Infrastructure as Code (IaC)

Application Performance Monitoring/Management (APM)

Improved communication and collaboration

What is Continuous Integration (CI)?

Simply put, CI is the process of **integrating** code into a mainline code base

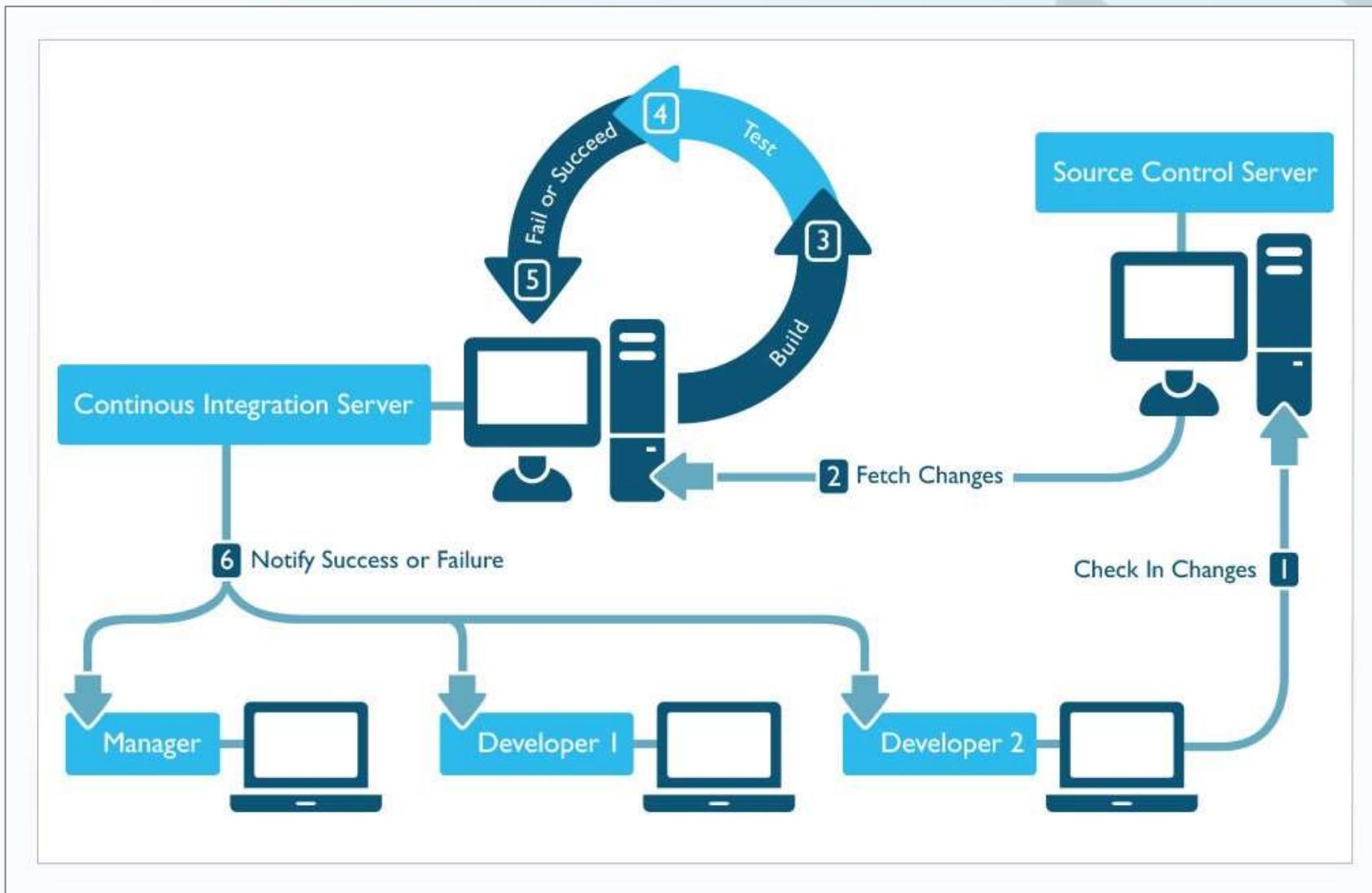
Source:

<https://devops.com/continuous-integration-vs-continuous-delivery-theres-important-difference/>

Development practice that requires developers to **integrate** code into a shared repository several times a day.

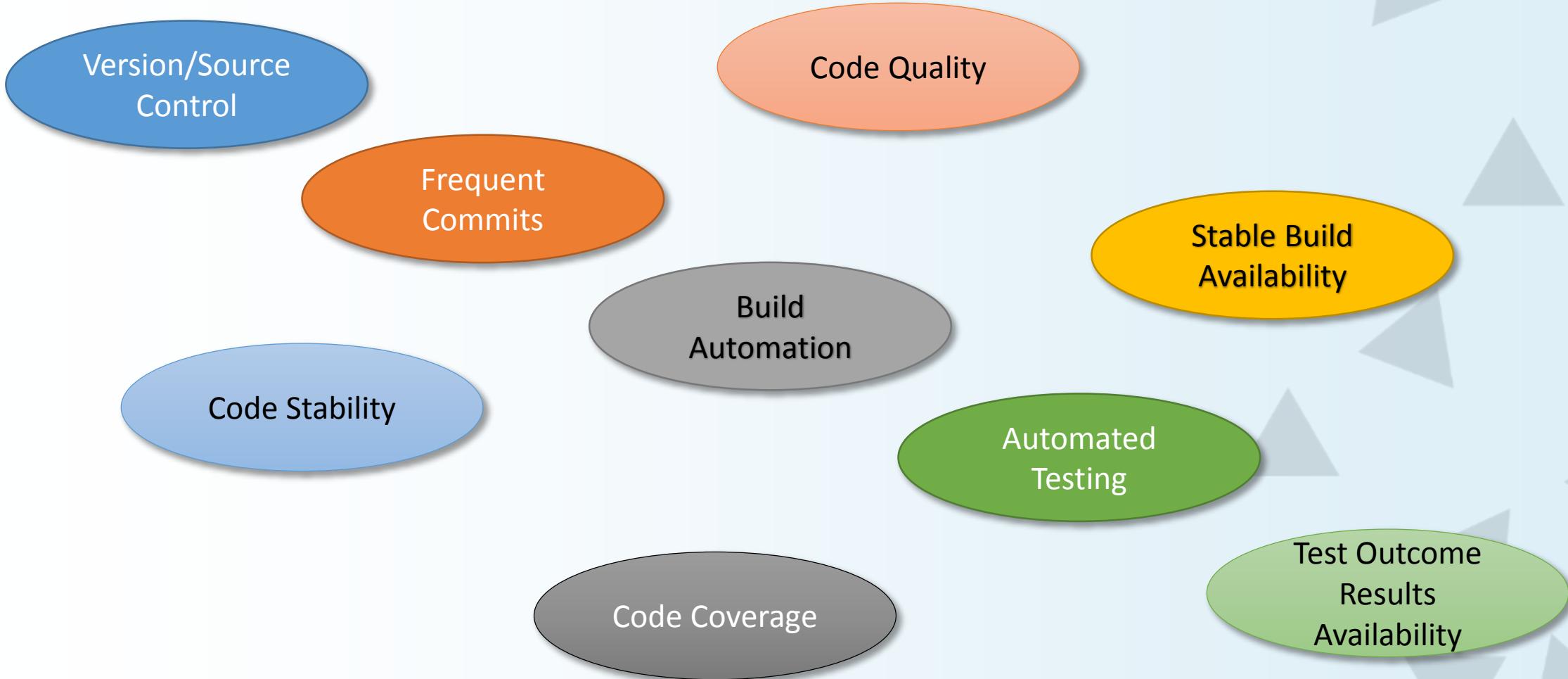
Source:

<https://www.thoughtworks.com/continuous-integration>



Source: [Carnegie Mellon University's Software Engineering Institute](http://se.cmu.edu/white-papers/continuous-integration.html)

Continuous Integration (CI) – Key Elements



Why CI?

Benefits

Early detection of bugs / issues

Immediate feedback on system-wide impact of local changes

Constant availability of a "current" build for testing, demo, or release purposes

Enforces discipline of frequent automated testing

Faster time to release with repeatable processes

Downsides

Automated test suites require considerable amount of work to set up and also for ongoing needs.

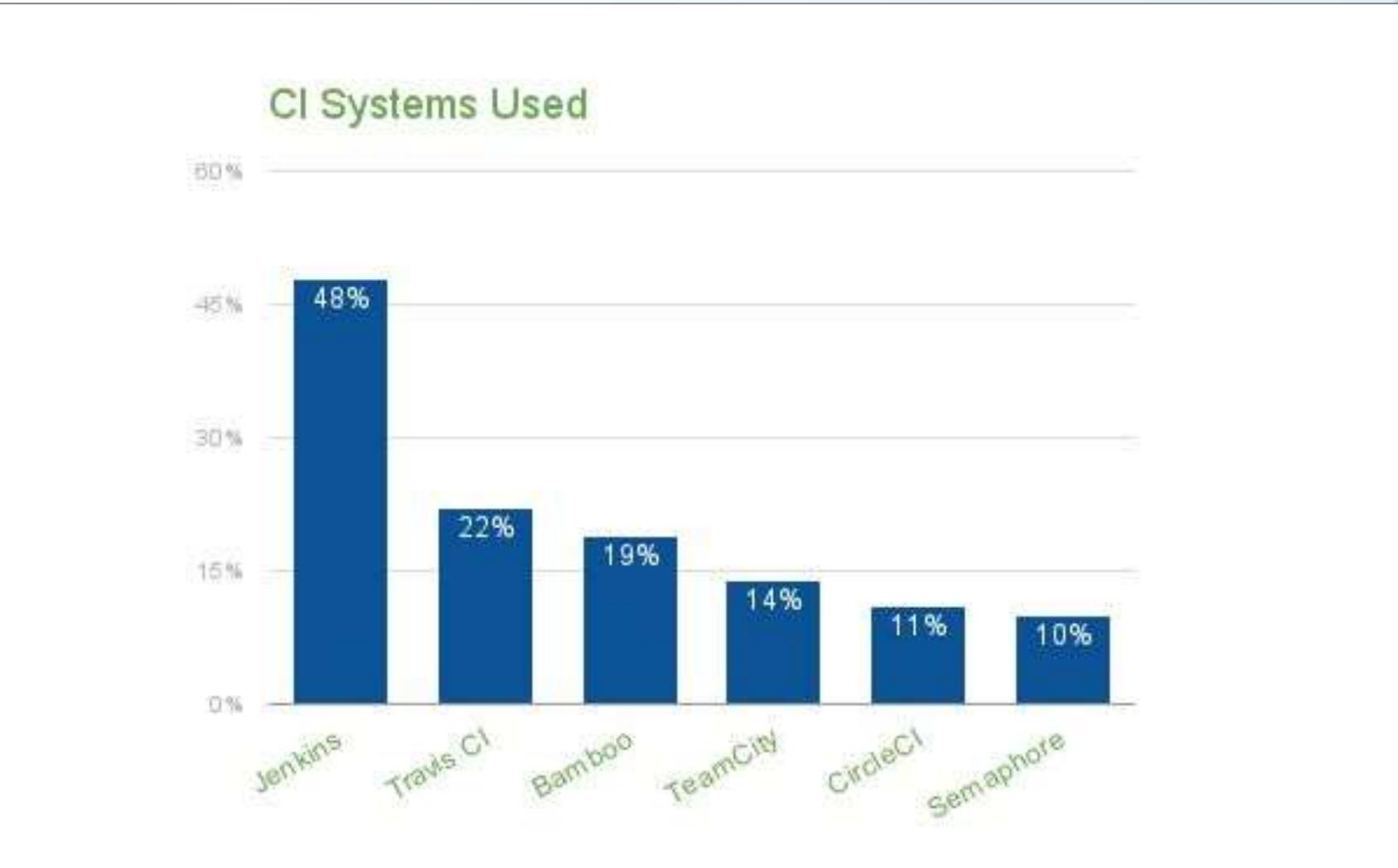
Work involved to set up a build system

Value added depends on the quality of tests and how testable the code really is

Builds queueing up can slow down everyone

Partial code could easily be pushed and therefore integration tests could fail until the feature is complete

CI Tool Box Usage

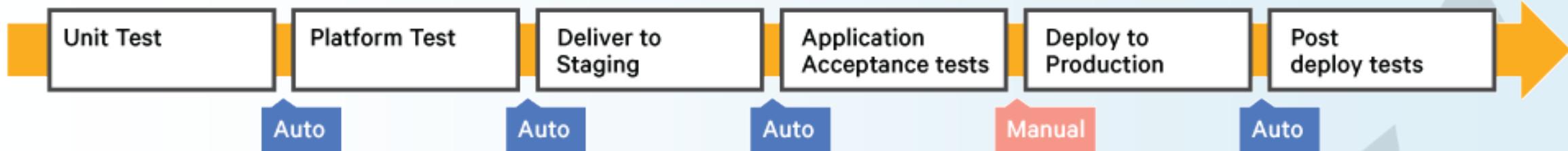


<https://blog.1and1.com/2016/08/11/continuous-integration-trends-from-bitnami-s-user-survey/>

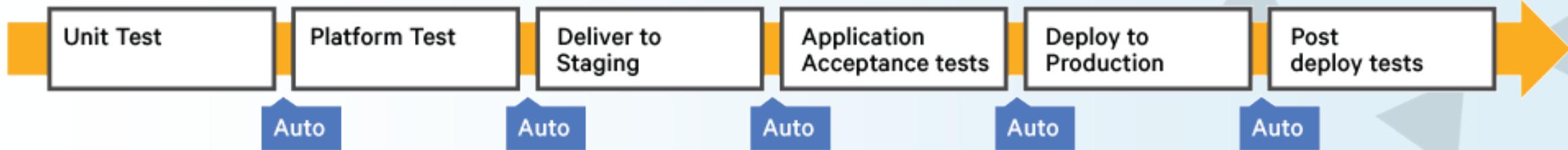
What is Continuous Delivery / Deployment (CD)?

Take 1 of 2

Continuous Delivery



Continuous Deployment



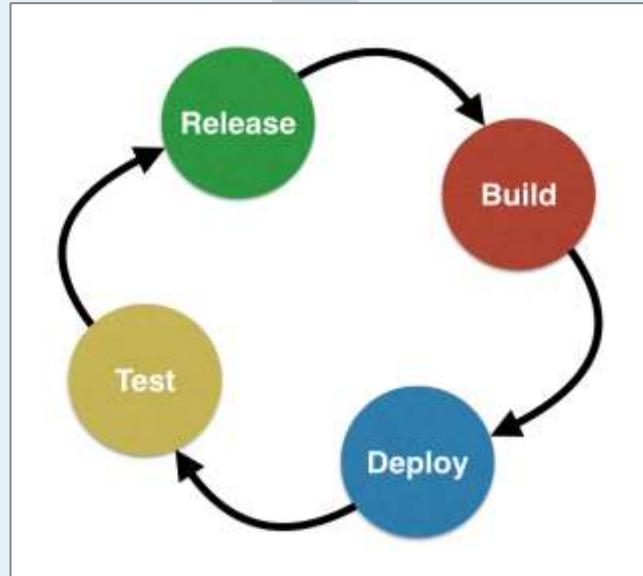
Source: [puppet](#)

What is Continuous Delivery / Deployment (CD)?

Take 2 of 2

Continuous Delivery

- ✓ Implementing continuous delivery means making sure your software is always production ready throughout its entire lifecycle - that any build could potentially be released to users at the touch of a button using a fully automated process in a matter of seconds or minutes.



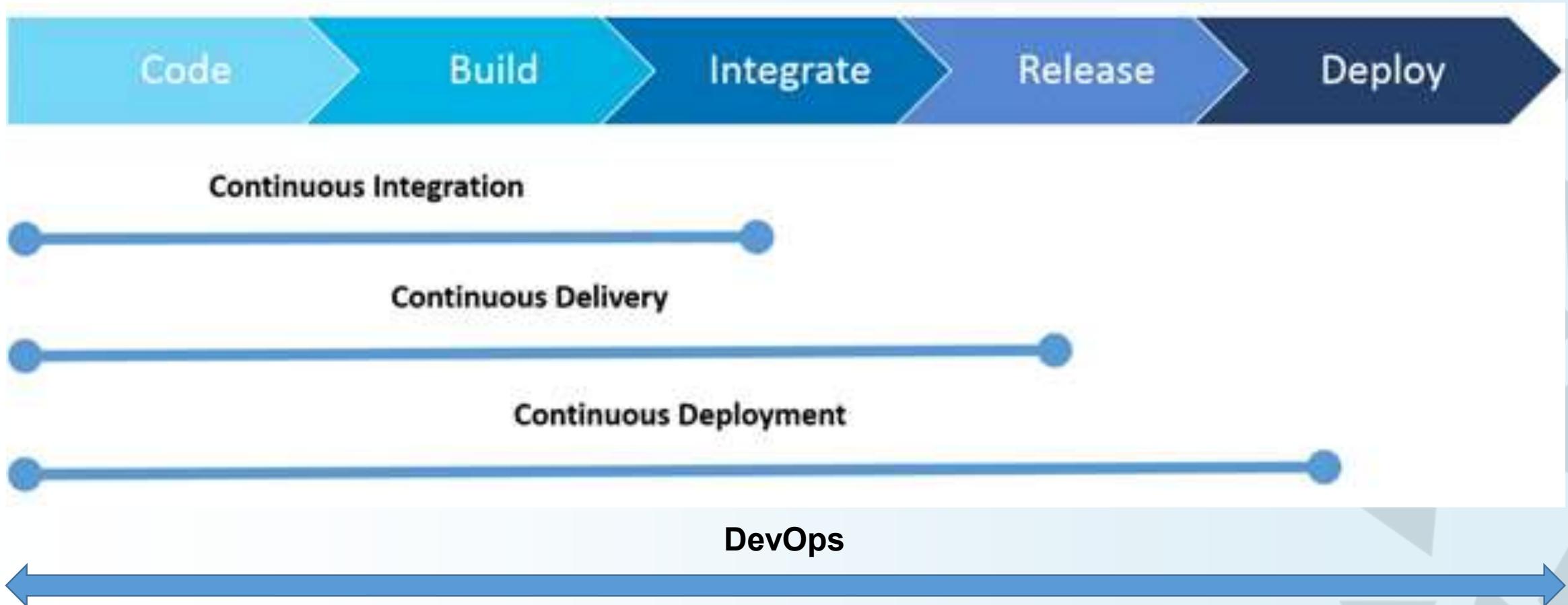
Continuous Deployment

- ✓ Essentially, it is the practice of releasing every good build to users
- ✓ What makes continuous deployment special is deploying every change that passes the automated tests to production
- ✓ Continuous deployment is the next step of continuous delivery. (*Source: puppet*)

While continuous deployment implies continuous delivery the converse is not true

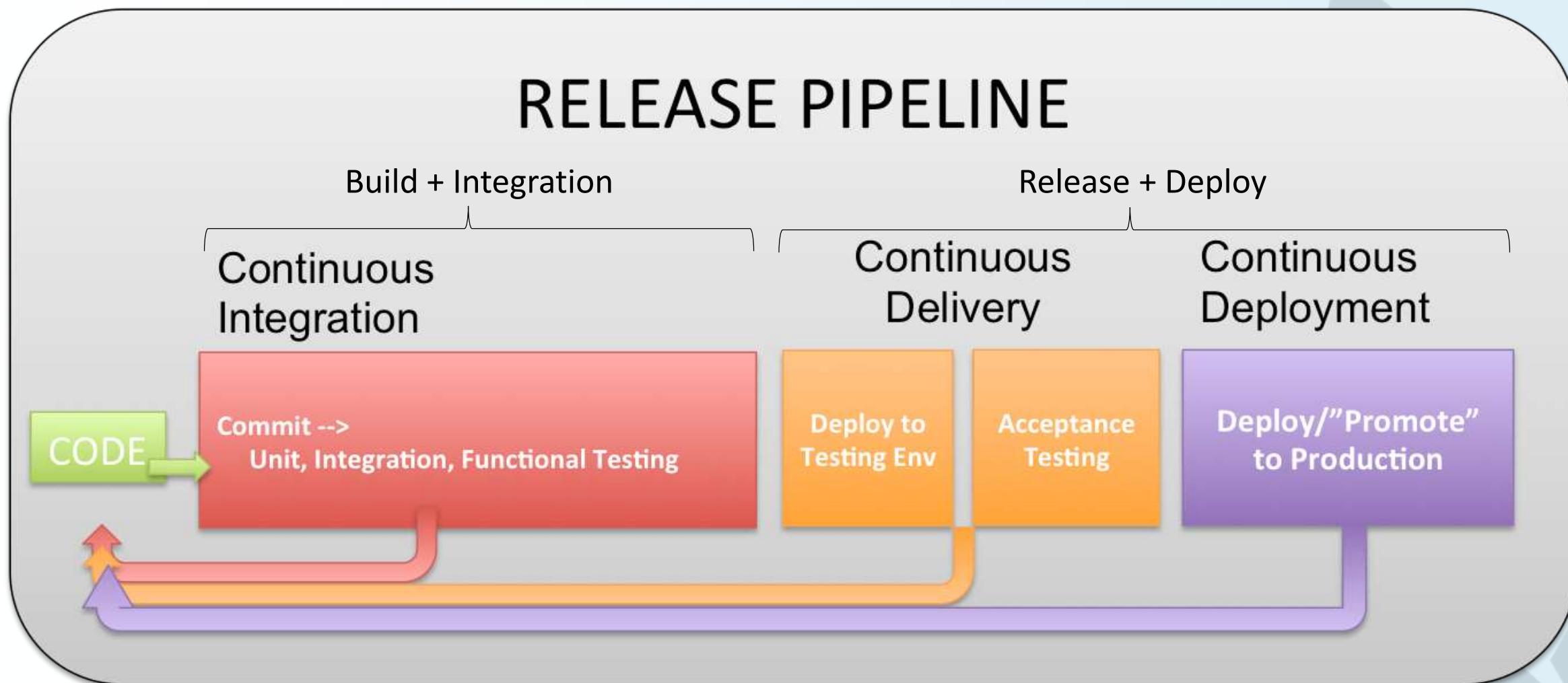
<https://continuousdelivery.com/2010/08/continuous-delivery-vs-continuous-deployment/>

Continuous - Integration vs Delivery vs Deployment



<http://www.saviantconsulting.com/blog/difference-between-continuous-integration-continuous-delivery-and-continuous-deployment.aspx>

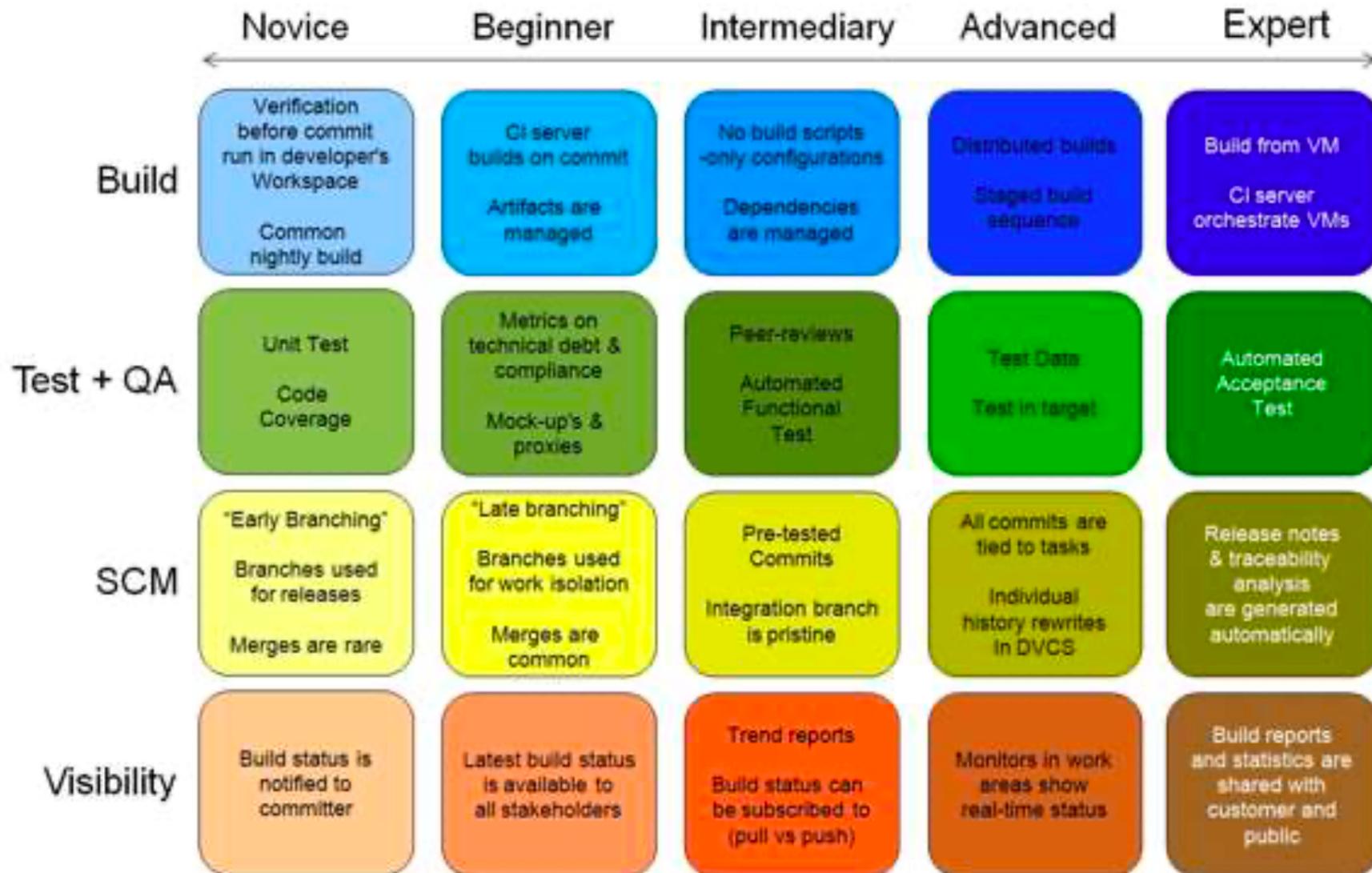
CI / CD Pipeline (With Feedback System)



Source: <https://devops.com/i-want-to-do-continuous-deployment/>



CI maturity matrix



DevOps – Using the right tools to achieve DevOps



Please see Appendix section for a more elaborate list of tools

Types Of DevOps Tools

- There are 9 types of DevOps tools which has known before choosing for the project...
- **Collaboration Tools:**
 - This type of tool is crucial to helping teams work together more easily, regardless of time zones or locations.
 - A rapid action oriented communication designed to share knowledge and save time. (See: **Slack, Campfire**).
- **Planning Tools:**
 - This type of tool is designed to provide transparency to stakeholder and participants.
 - Working together, teams can plan towards common goals, and better understanding of dependencies. Bottlenecks and conflicting priorities are more visible. (See: **Clarizen and Asana**).
- **Source Control Tools:**
 - Tools of this sort make up the building blocks for the entire process ranging across all key assets. Whether code, configuration, documentation, database, compiled resources and your web site html – you can only gain by managing them in your one true source of truth.(See: **Git, Subversion**).

Types Of DevOps Tools

- **Issue Tracking Tools:**
 - These tools increase responsiveness and visibility.
 - All teams should use the same issue tracking tool, unifying internal issue tracking as well as customer generated ones.
(See: **Jira** and **ZenDesk**).
- **Configuration Management Tools:**
 - Without this type of tool, it would be impossible to enforce desired state norms or achieve any sort of consistency at scale.
 - Infrastructure should be treated exactly as code that can be provisioned and configured in a repeatable way.
(See: **Puppet**, **Chef**, **Salt**).
- **Database DevOps Tools:**
 - The database, obviously, needs to be an honored member of the managed resources family. Managing source code, tasks, configuration, and deployments is incomplete if the database is left out of the equation.(See: **DBmaestro**)

Types Of DevOps Tools

- **Continuous Integration Tools:**
 - Continuous integration tools provide an immediate feedback loop by regularly merging code. Teams merge developed code many times a day, getting feedback from automated test tools. (See: **Jenkins, Bamboo, TeamCity**).
- **Automated Testing Tools:**
 - Tools of this sort are tasked with verifying code quality before passing the build. The quicker the feedback loop works – the higher the quality gets, and the quicker you reach the desired "definition of done".(See: **Telerik, QTP, TestComplete**)
- **Deployment Tools:**
 - In an effective DevOps environment, application deployments are frequent, predictable, and reliable.
 - Deployment tools are essential to checking those boxes. Continuous delivery means that applications can be released to production at any time you want in order to improve time to market, while keeping risk as low as possible. (See: **IBM uDeploy, CA Release Automation, XebiaLabs**)

DevOps Tools

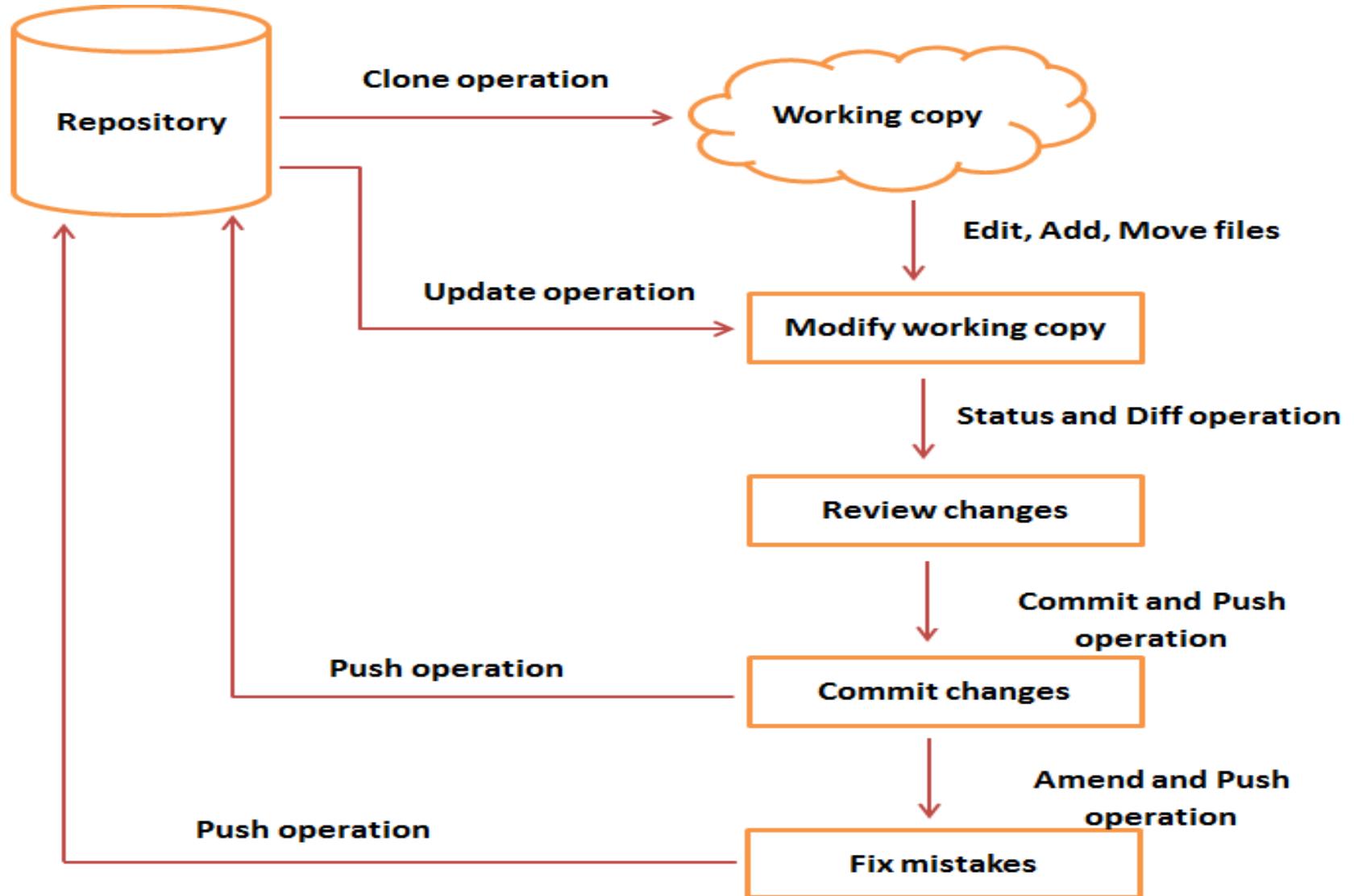


Git

- In recent years, Git has become incredibly popular for source code management, particularly as the site GitHub has become more popular for hosting open source projects.
- It stands out from other version control management for the ease with which it handles branching and merging.
- It's also very easy to use with distributed development teams, and it offers fast performance.
- Many DevOps teams use it to manage the source code for their applications.
- Its list of well-known users includes many of the biggest firms in the technology industry, such as Google, Facebook, Microsoft, Twitter, LinkedIn, Netflix, the Linux kernel and many others.



DevOps Tools



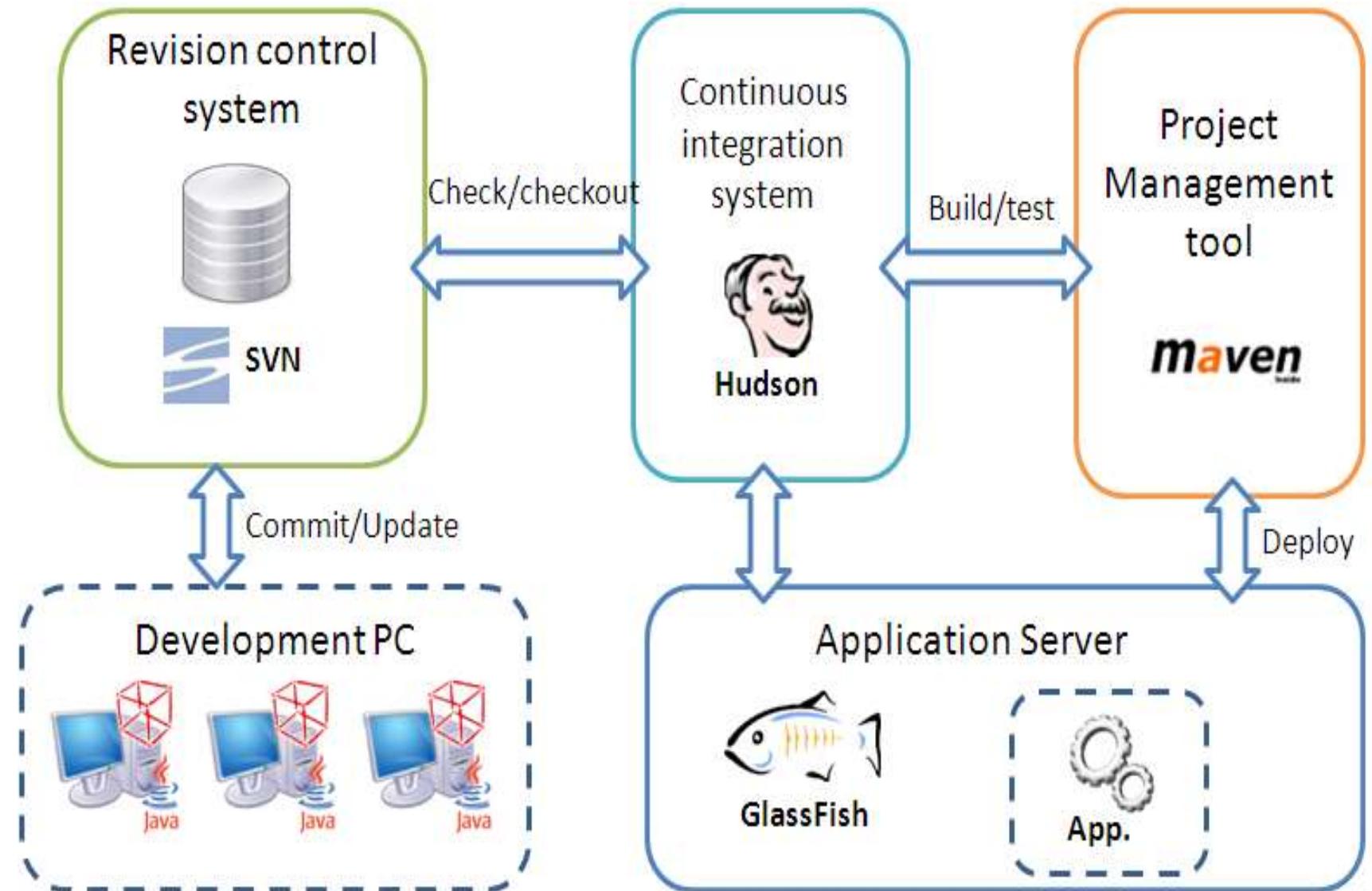
DevOps Tools



Hudson

- When Oracle bought Sun, it declared its intention to trademark the Hudson name, and development began on a commercial version.
- Continuous integration is an integral part of the DevOps approach, and Hudson.
- It is a tool for monitoring and managing continuous integration and testing.
- Its key features include easy installation and configuration, change set support, real-time notifications of test failures, file fingerprinting and support for a wide variety of source code management systems, build tools, testing frameworks, code analysis tools, application servers and other DevOps tools.
- Hudson is managed by the Eclipse Foundation, and there is a huge library of plug-ins that extend its capabilities.

DevOps Tools



DevOps Tools



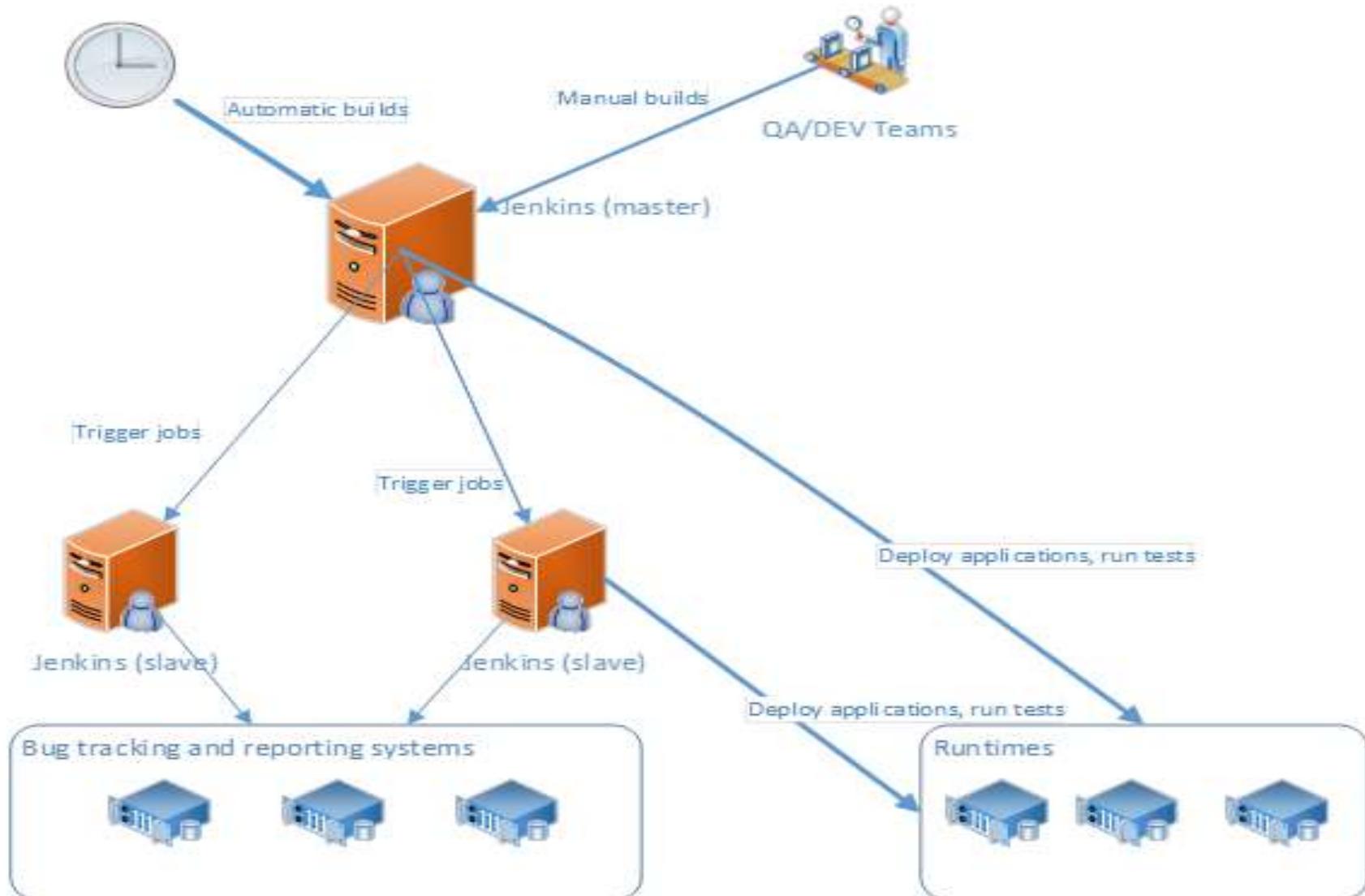
Jenkins

- The "leading open source automation server," Jenkins was forked from Hudson and offers many of the same capabilities.
- It boasts easy installation and configuration, hundreds of plugins, extensibility and a distributed architecture that allows it to speed the process of testing.
- It has a very active user community with lots of scheduled events that offer opportunities to learn more about the software.
- There is also plenty of documentation on the website, including a blog that is updated regularly.
- Jenkins Pipeline is a suite of plugins which supports implementing and integrating continuous delivery pipelines into Jenkins.
- Pipeline provides an extensible set of tools for modeling simple-to-complex delivery pipelines "as code".

DevOps Tools

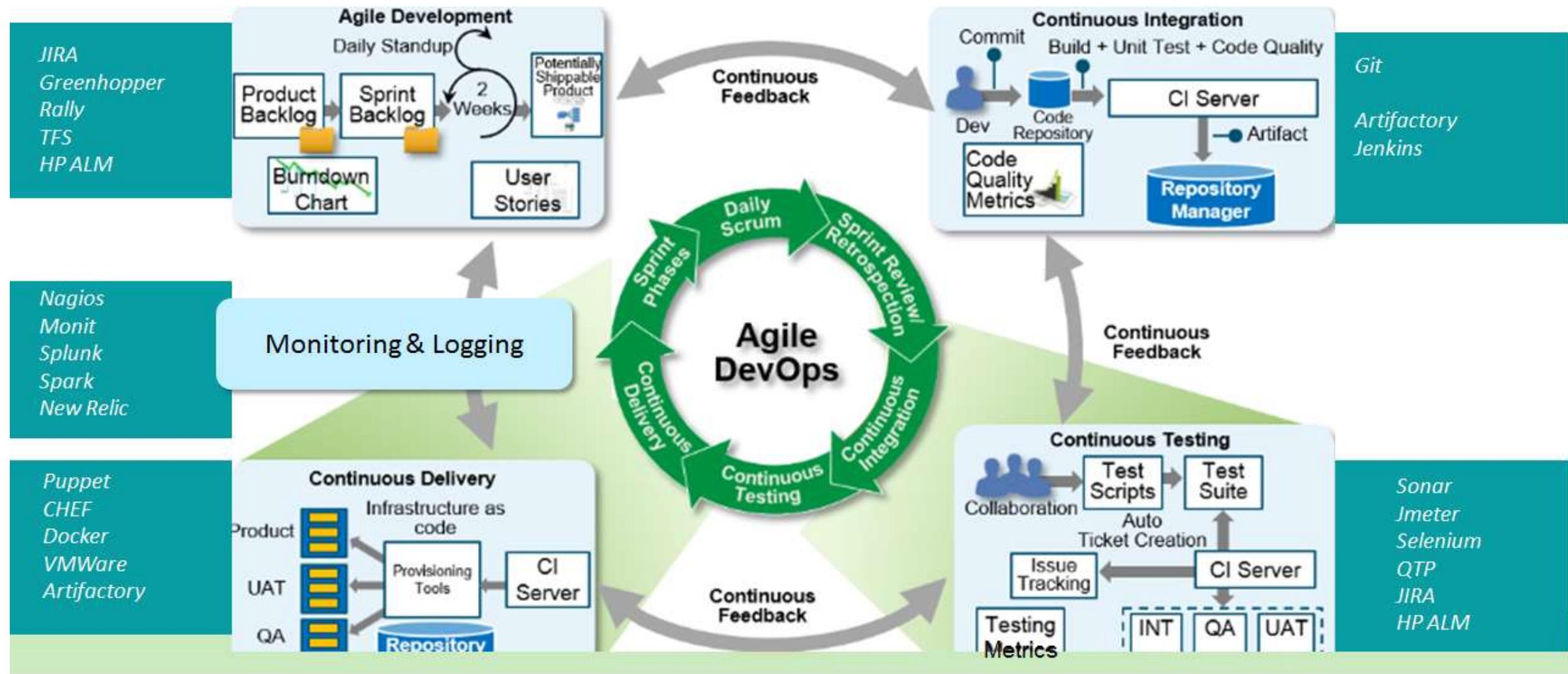


Jenkins



DevOps – A Representative Reference Architecture

Agile Methodology



Appendix 1 – Periodic Table of DevOps Tools

-<https://xebialabs.com/periodic-table-of-devops-tools/>

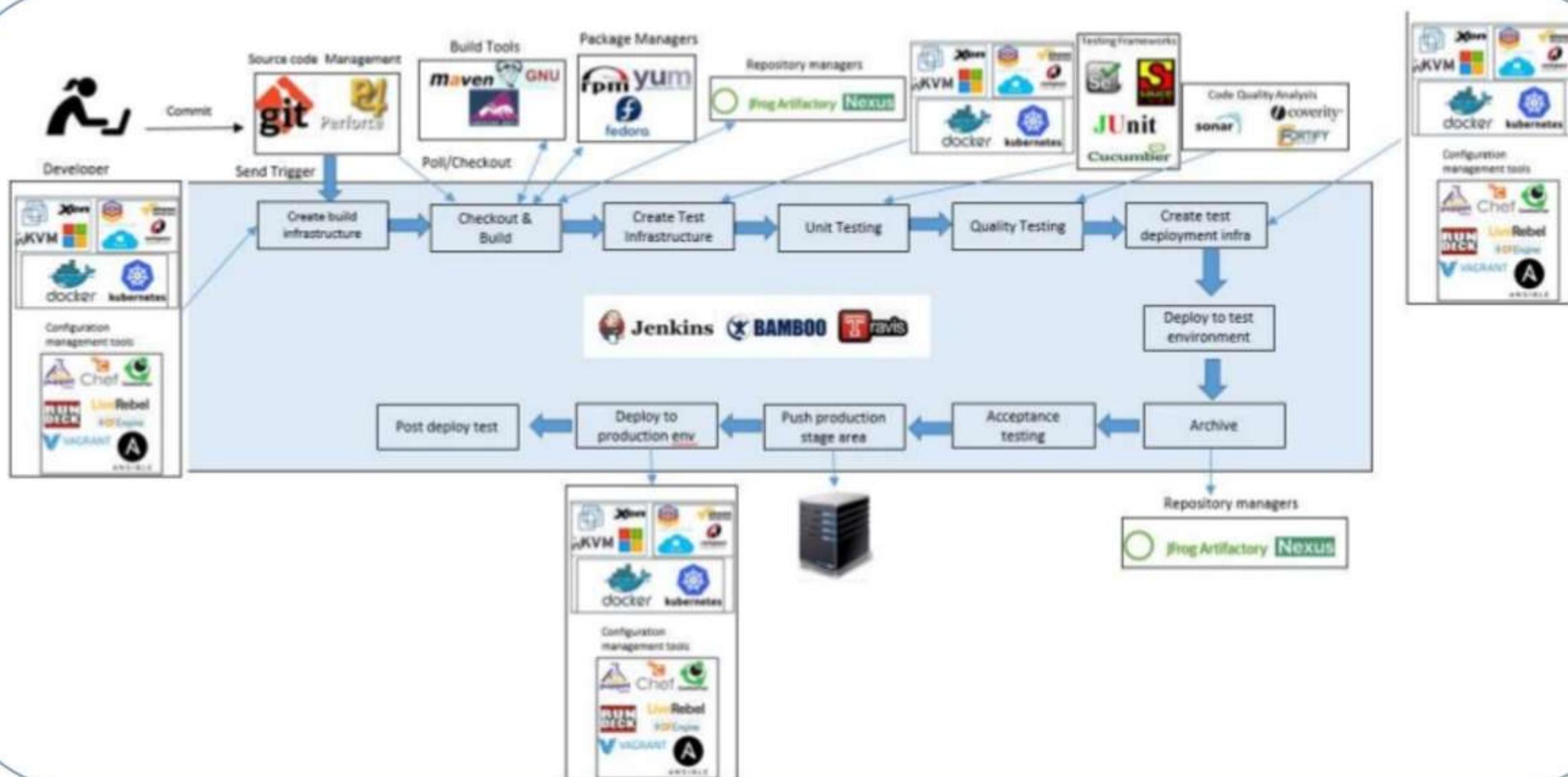
XebiaLabs

Follow @xebialabs

Appendix 2 – DevOps Tooling Landscape



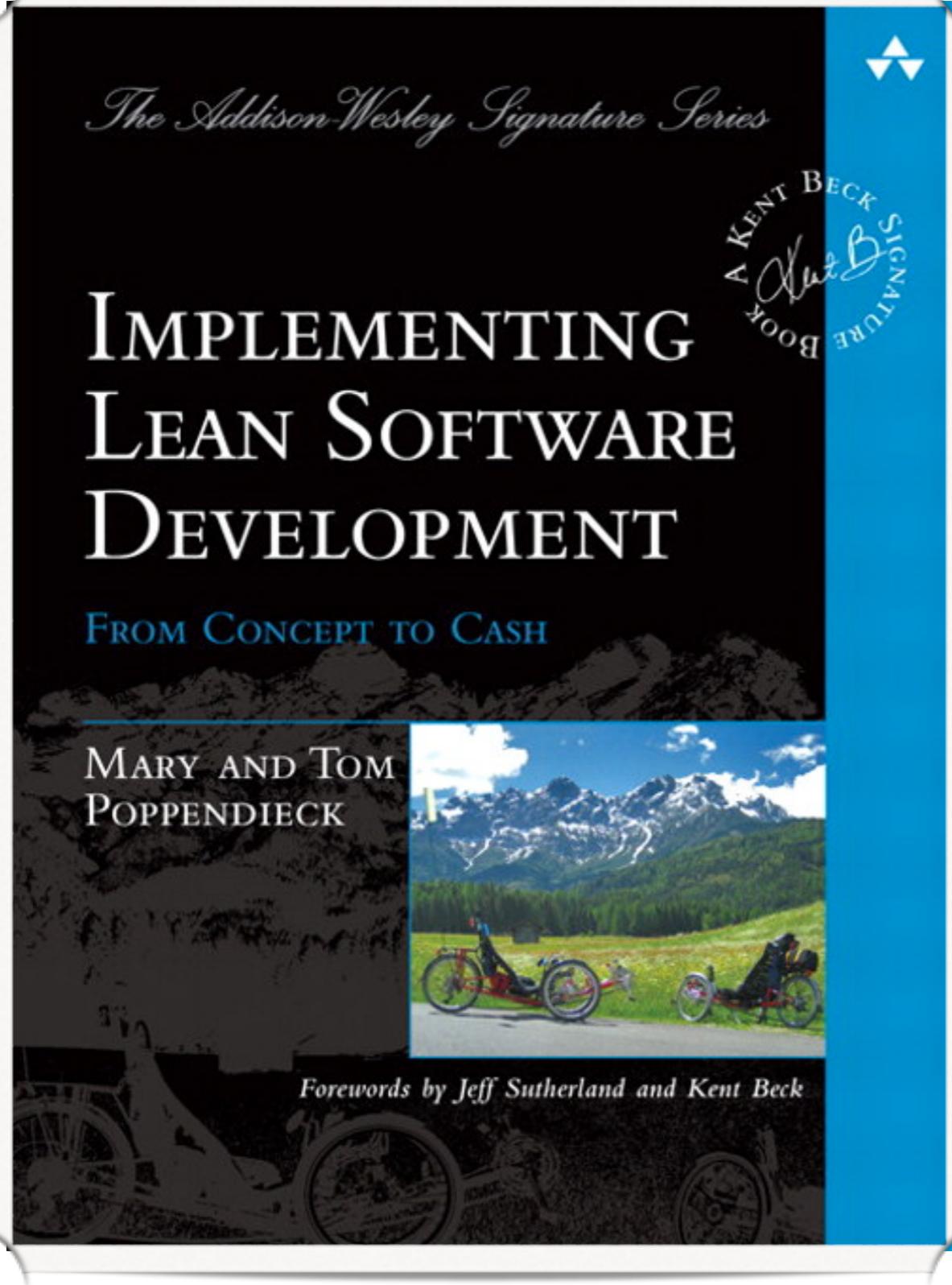
Appendix 3 – Continuous Deployment Pipeline Architecture



Recommended Reading

FROM CONCEPT TO CASH

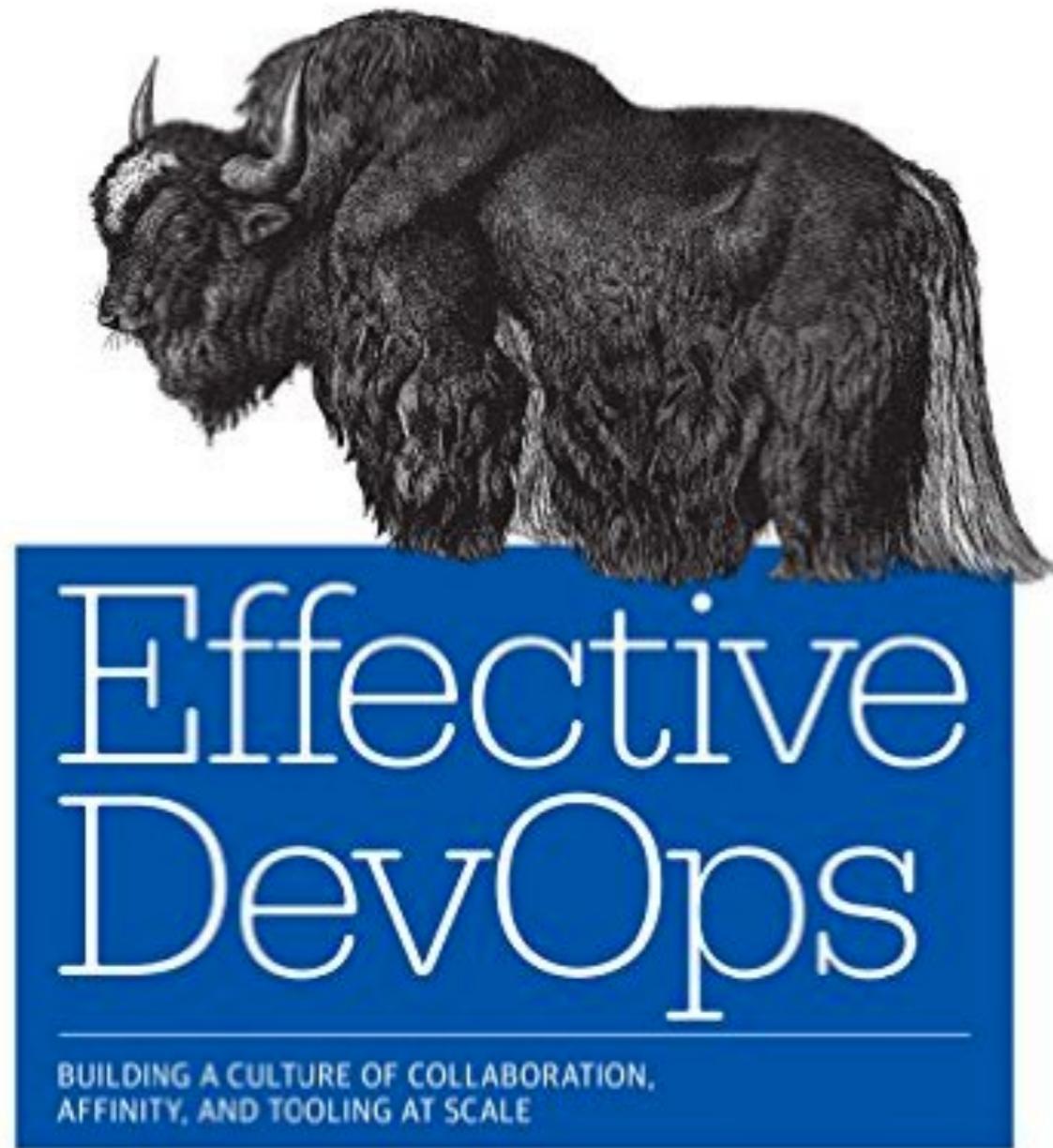
.....



- Lean Software Development book provides the foundation for DevOps; its principles are:
 - Eliminate waste
 - Amplify learning
 - Decide as late as possible
 - Deliver as fast as possible
 - Empower the team
 - Build quality in
 - See the whole
- Read it to perform value stream mapping for DevOps

"*Implementing Lean Software Development: From Concept to Cash*", Mary Poppendieck, Tom Poppendieck, Addison-Wesley Professional, 2006

<https://amzn.com/0321437381>



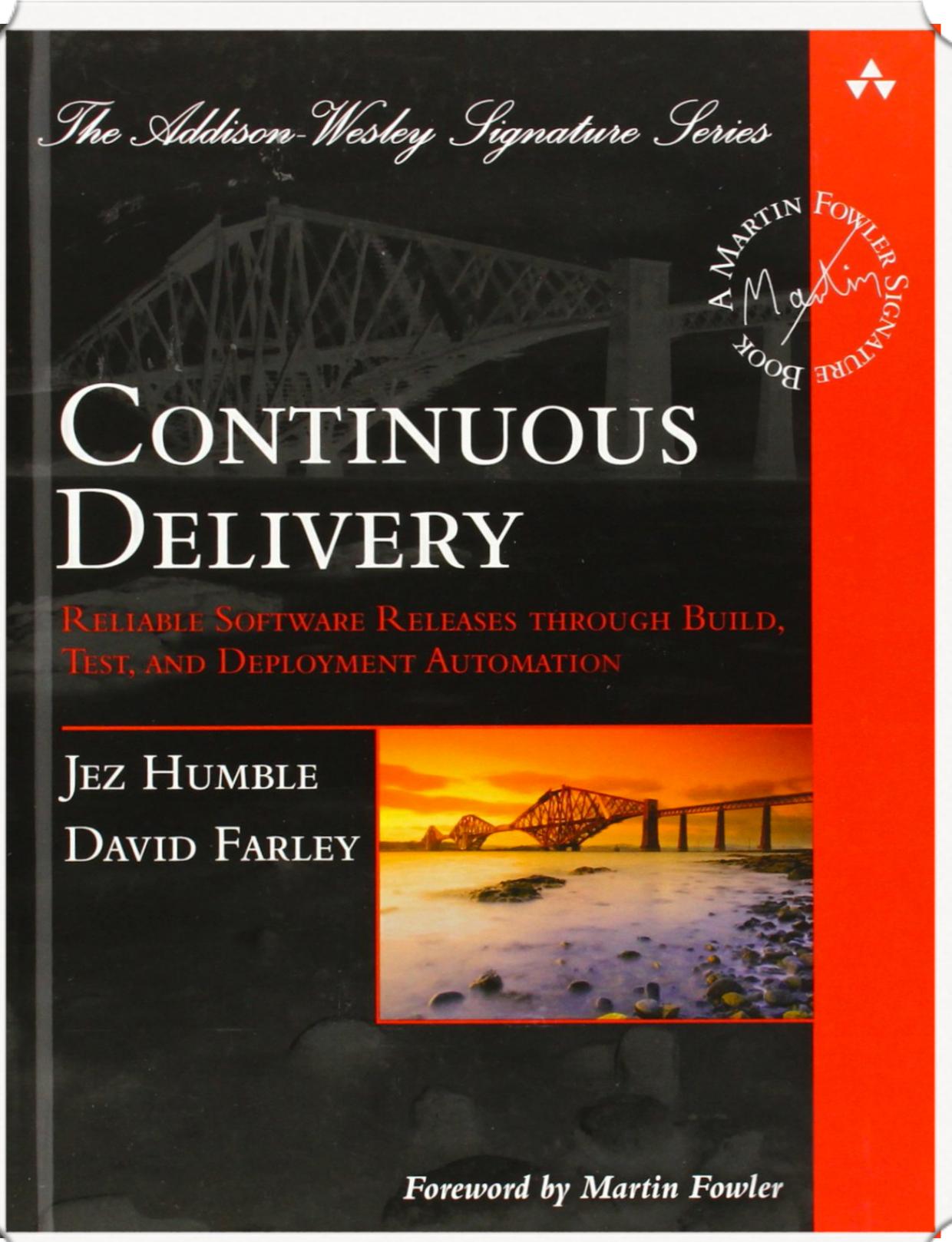
Effective DevOps: Building a Culture of Collaboration, Affinity, and Tooling at Scale, Jennifer Davis, Katherine Daniels, O'Reilly Media, 2016

<https://amzn.com/1491926309>

A BROAD PERSPECTIVE

.....

- DevOps is more than bringing in DevOps engineers or using tools
- This book provides an overall perspective of DevOps by covering its history and covers foundational concepts
- Covers four pillars of effective DevOps: Collaboration, Affinity, Tools, and Scaling
- Has interesting case studies, clears misconceptions, and helps troubleshoot



“Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation”, Jez Humble, David Farley, Addison Wesley, 2010

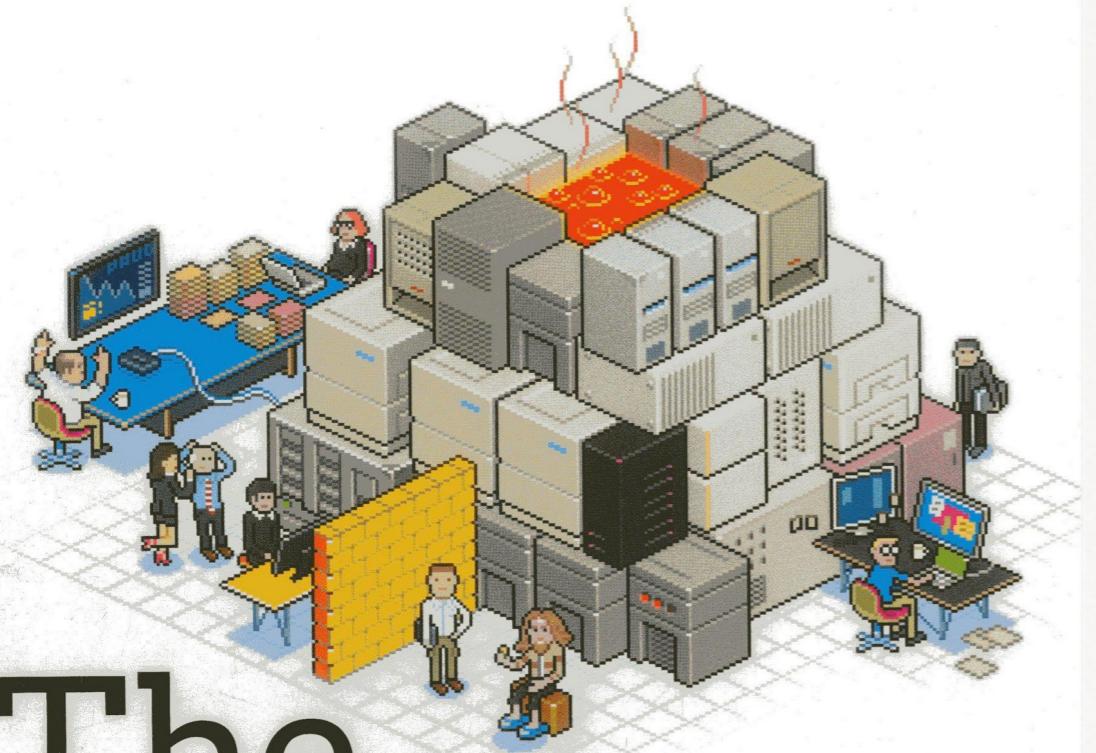
<http://www.amazon.in/dp/0321601912>

CD DISTILLED

.....

- An early book on Continuous Delivery
- Released before DevOps became a buzzword - but it covers dev ops concepts in its essence
- Book divided into three parts: Foundation, The Deployment Pipeline, and The Delivery Ecosystem
- Written based on the practical experience of authors and covers important aspects such as feature toggles; hence must read

From the authors of *The Visible Ops Handbook*



The Phoenix Project

A Novel About IT, DevOps,
and Helping Your Business Win

Gene Kim, Kevin Behr, and George Spafford

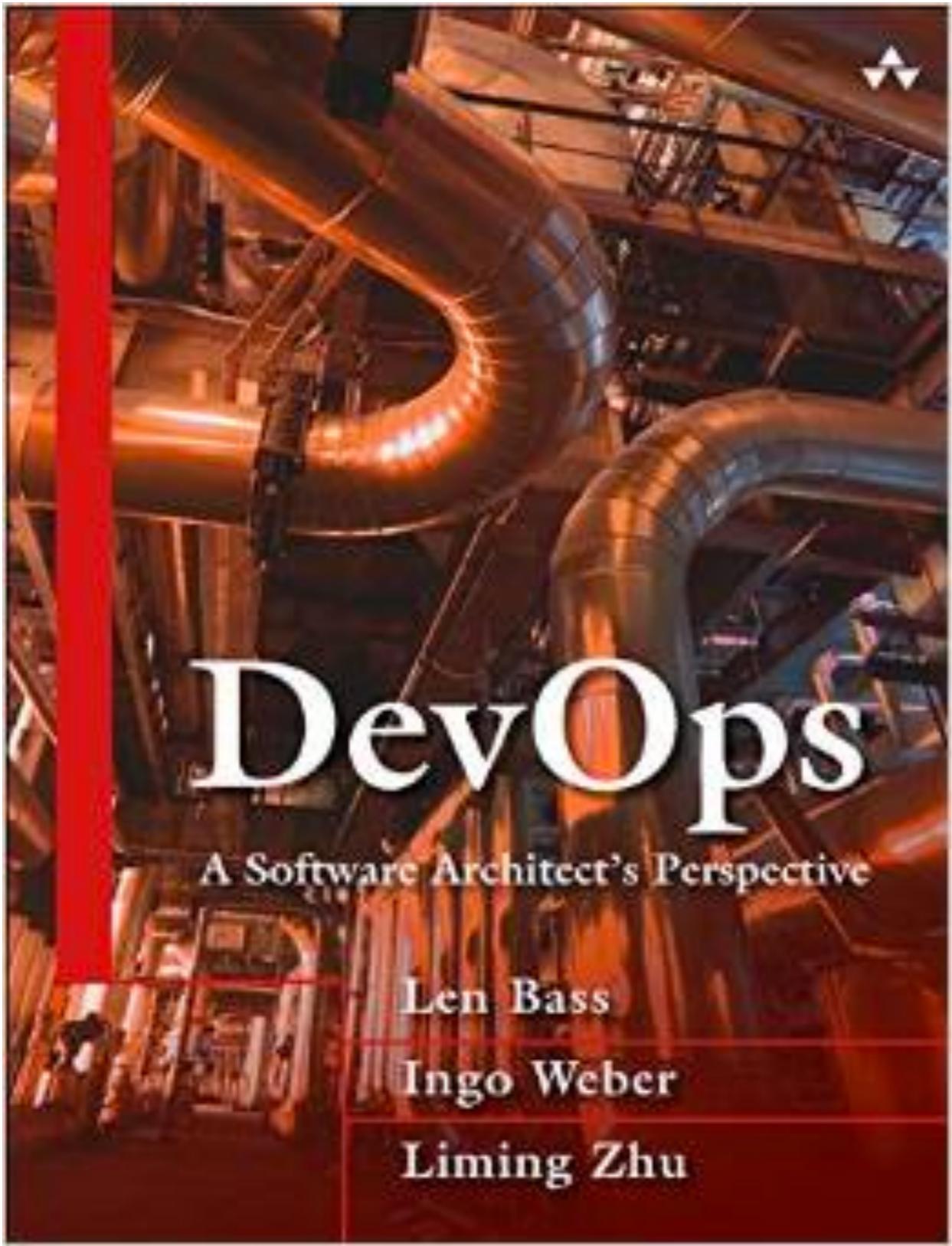
"The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win",
Gene Kim, Kevin Behr, George Spafford, IT Revolution Press, 2013

<https://amzn.com/0988262592>

WHY DEVOPS?

.....

- From the authors of the popular “The Visible Ops Handbook”
- This book shows why DevOps and how it helps business win
- It tells the story of an IT manager who has to rescue an IT project in crisis
- Written in a fiction style, so easy to read
- An important read for managers for effective adoption of DevOps practices



DevOps: A Software Architect's Perspective, Len Bass, Ingo Weber, Liming Zhu,
Addison-Wesley Professional, 2015
<https://amzn.com/0134049845>

ARCHITECT'S PERSPECTIVE

- Most books on DevOps talk about tools, process and technology perspective
- This book is a take on an architect's perspective on DevOps
- Covers deployment pipeline, cross-cutting concerns (monitoring, security, ...), and case studies
- A bit-dry (written in more of an academic style)

