DevOps

Capgemini

CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.

Table Of Contents

Part I – Introduction

DevOps – the business need?

What is DevOps?

What DevOps is not?

How does DevOps work?

Part II – Implementation and Tools

DevOps Practices

Infrastructure as Code

Learn a Tool - Chef and Puppet

Source Code Management

Learn a Tool - GIT

Continuous Integration

Learn a Tool - Jenkins

- Automated Testing
- Learn a Tool - Selenium
- Continuous Deployment
- Release Management

Part III – Introduction

- DevOps Cloud
- Cloud and DevOps using BlueMix
- BlueMix DevOps – Example

Part IV – Demo

- DevOps Delivery Pipelin
- DevOps App Monitoring & Scaling
- DevOps Web Editor

DevOps- Introduction

DevOps – the business need

The Developer

- New Products
- New Features
- Security Update
- Bug Fixes

- Old Code
- Pending Code
- New Products
- New features

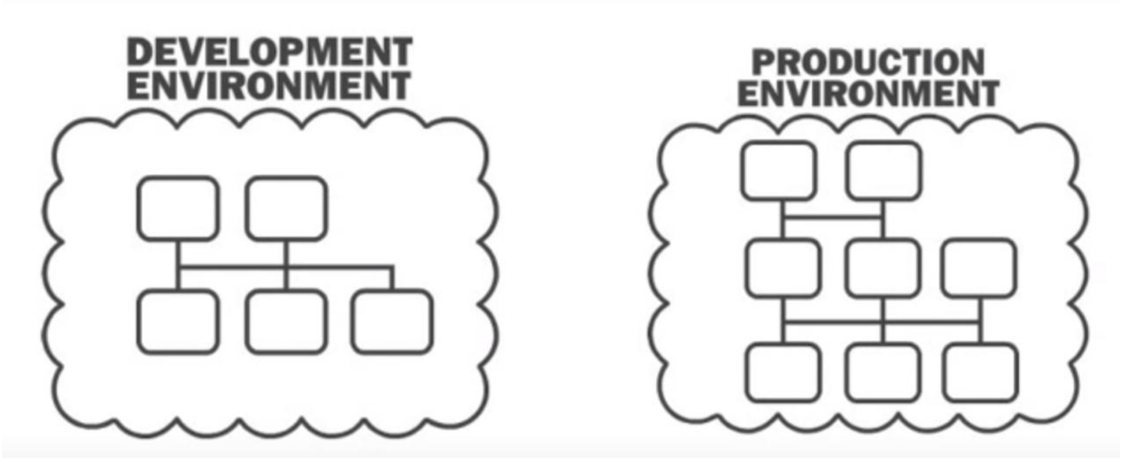
Time to Market

Dependency Error

DevOps – the business need

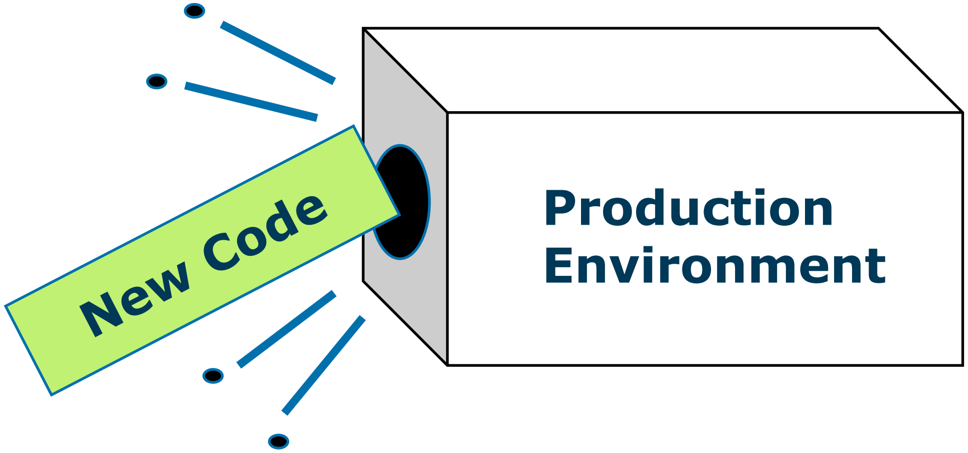
The Developer

As a developer I have always dabbled lightly in operations. I always wanted to focus on making my code great and let an operations team worry about setting up the production infrastructure.

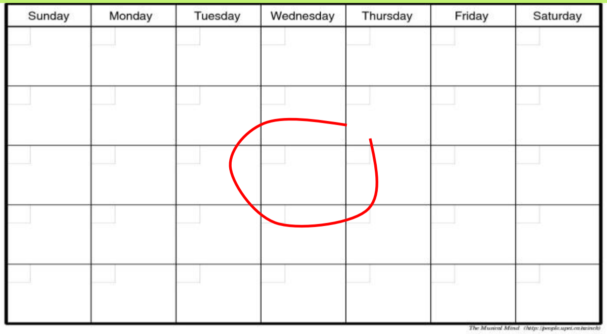


DevOps – the business need

The Operations team



Deployment Schedule



Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday

The Musical Mind (http://people.upei.ca/hsinch)

DevOps – the business need

I am responsible for maintaining 99% uptime. I think of servers and new code deployment mostly introduces bugs which I need to fix to ensure availability. These developers are pushing **their** *work to me.*

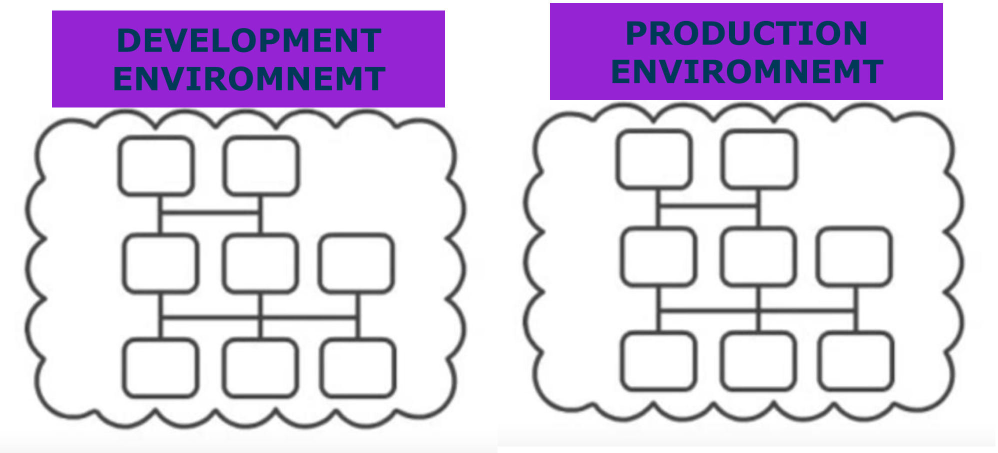
+ *New Code*

The Operations team

DevOps – the business need

DevOps

- ✓ **Worked Better together**
- ✓ **Thought more alike**
- ✓ **Broke down silos**
- ✓ **Shared responsibilities?**



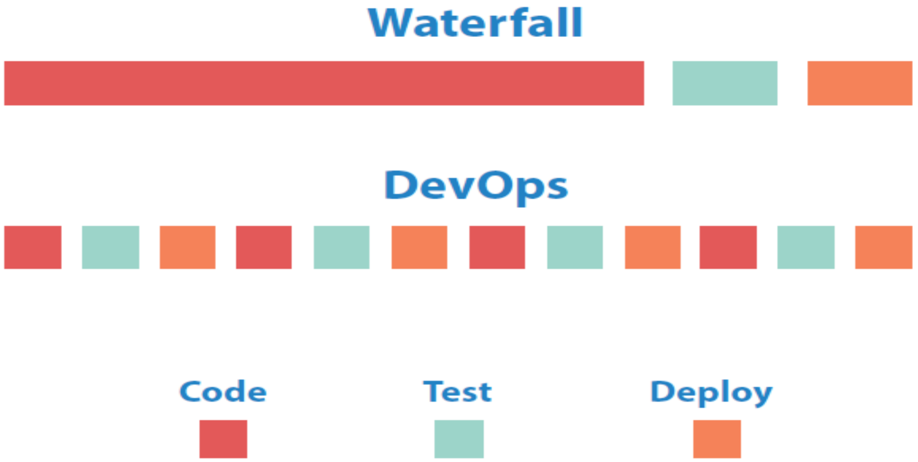
DEVELOPMENT ENVIROMNEMT

PRODUCTION ENVIROMNEMT

What is DevOps?

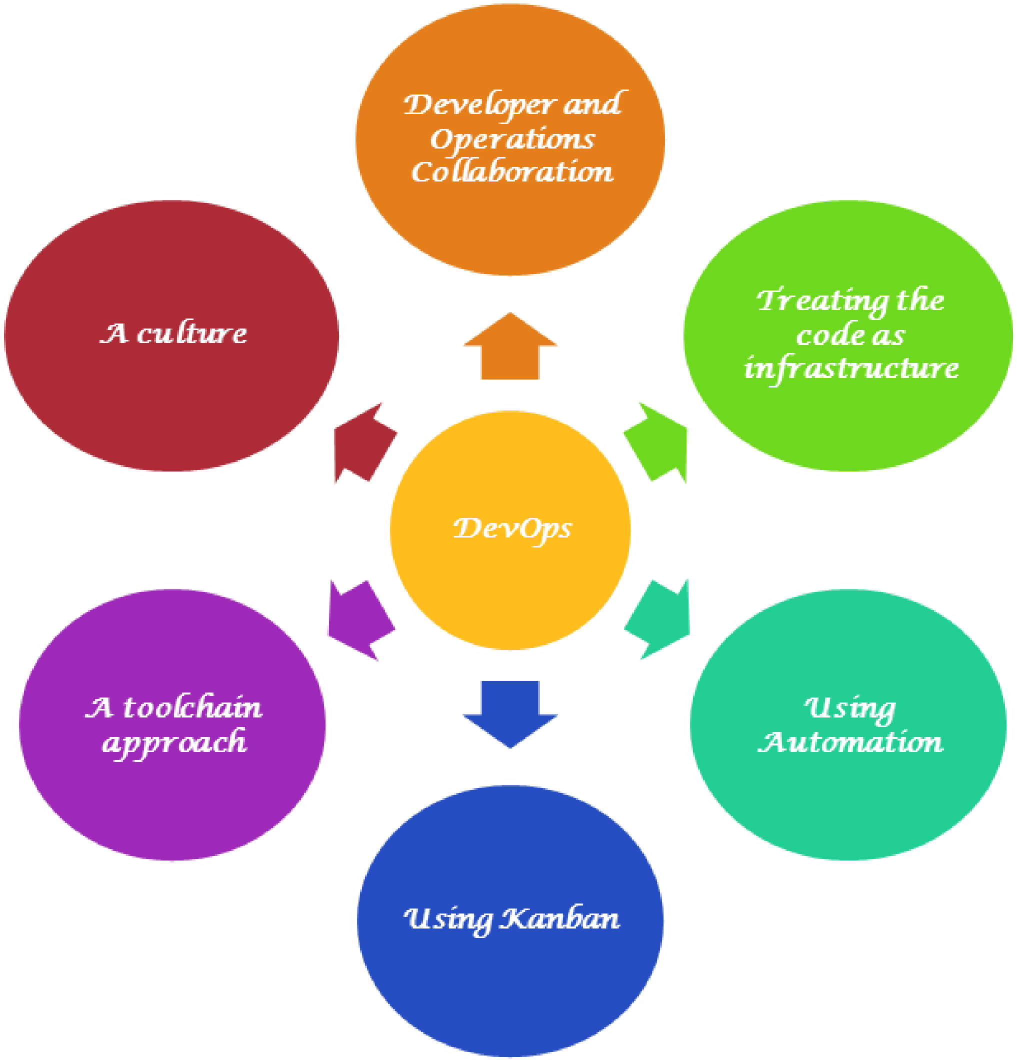
The Definition:

- ✓ **“ a software development method that stresses communication, collaboration & integration between software developers and IT professionals.” - wikipedia**
- ✓ **“DevOps is simply operations working together with engineers to get things done faster in an automated and repeatable way.”**



C.A.L.M.S.
C – Culture
A – Automation
L – Lean
M – Measurement
S – Sharing

What is DevOps?



DevOps

Developer and Operations Collaboration

Treating the code as infrastructure

Using Automation

Using Kanban

A toolchain approach

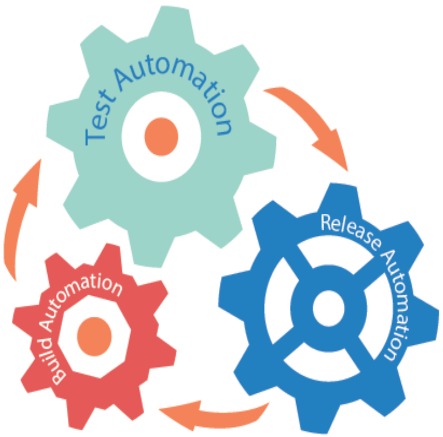
A culture

DevOps is also characterized by operations staff making use many of the same techniques as developers for their systems work.

What is DevOps?

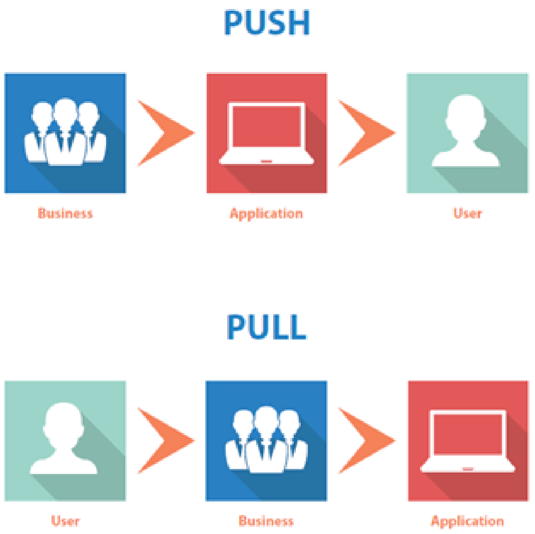
Automation - Optimizing the Entire Pipeline

- ✓ **The best way to quicken processes across the pipeline is to automate them.**
- ✓ **Build automation can be approached using Continuous Integration (CI) tools like Jenkins.**
- ✓ **Test automation requires frameworks like Selenium and Appium.**
- ✓ **And release automation, which is still maturing, can be handled with tools like Automic.**
- ✓ **DevOps is about optimizing processes across the entire pipeline, and automation is key to realizing this goal.**



PUSH vs PULL

- ✓ **The Lean approach to building apps involves a pull system where customers define what you should focus on, how fast you should go, and what you should ship, as opposed to the traditional top-down model of building applications.**



What is DevOps?

Agile and DevOps – A parallel definition

	Agile	DevOps
Values	Individual, Interaction, Communication, Collaboration, Change	People over Process over Tools
Principles	Customer Satisfaction, Changing Requirements, Continuous delivery	Infrastructure as code
Methods	XP, Scrum	Kanban, Scrum
Practices	Standups, planning poker, backlogs, CI	Continuous Integration and Continuous deployment
Tools	JIRA Agile (aka Greenhopper), planningpoker	Jenkins. Puppet, Chef, AWS

What DevOps is Not?

- **It’s Not NoOps :**

  DevOps is not that Developers take over Ops!

- **It’s Not (Just) Tools:**

  DevOps is also not simply implementing a set of tools.

- **It’s Not (Just) Culture**

  DevOps consists of items at all the levels

- **It’s Not (Just) Devs and Ops**

  What about security people! And network admins!

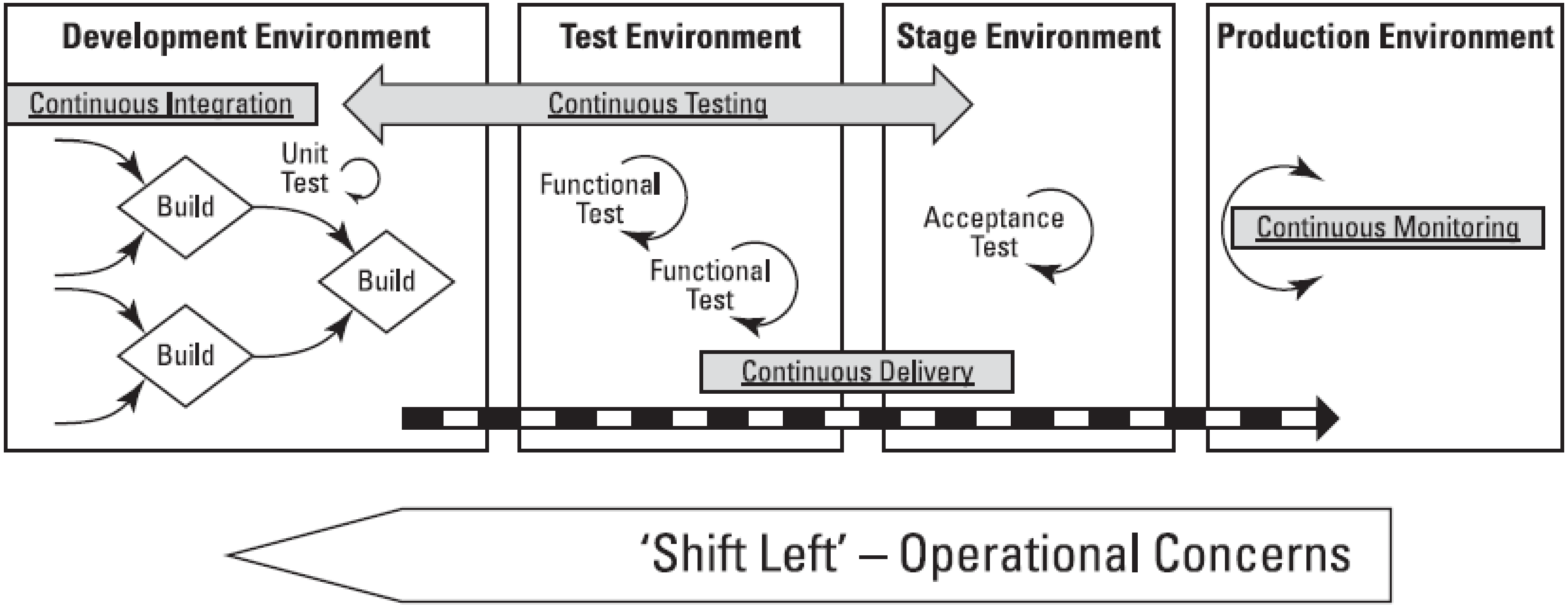
- **It’s Not Everything**

  It is part of an overall, hopefully collaborative and agile corporate culture, but DevOps is specifically about how operations plugs into that

How does DevOps Work?

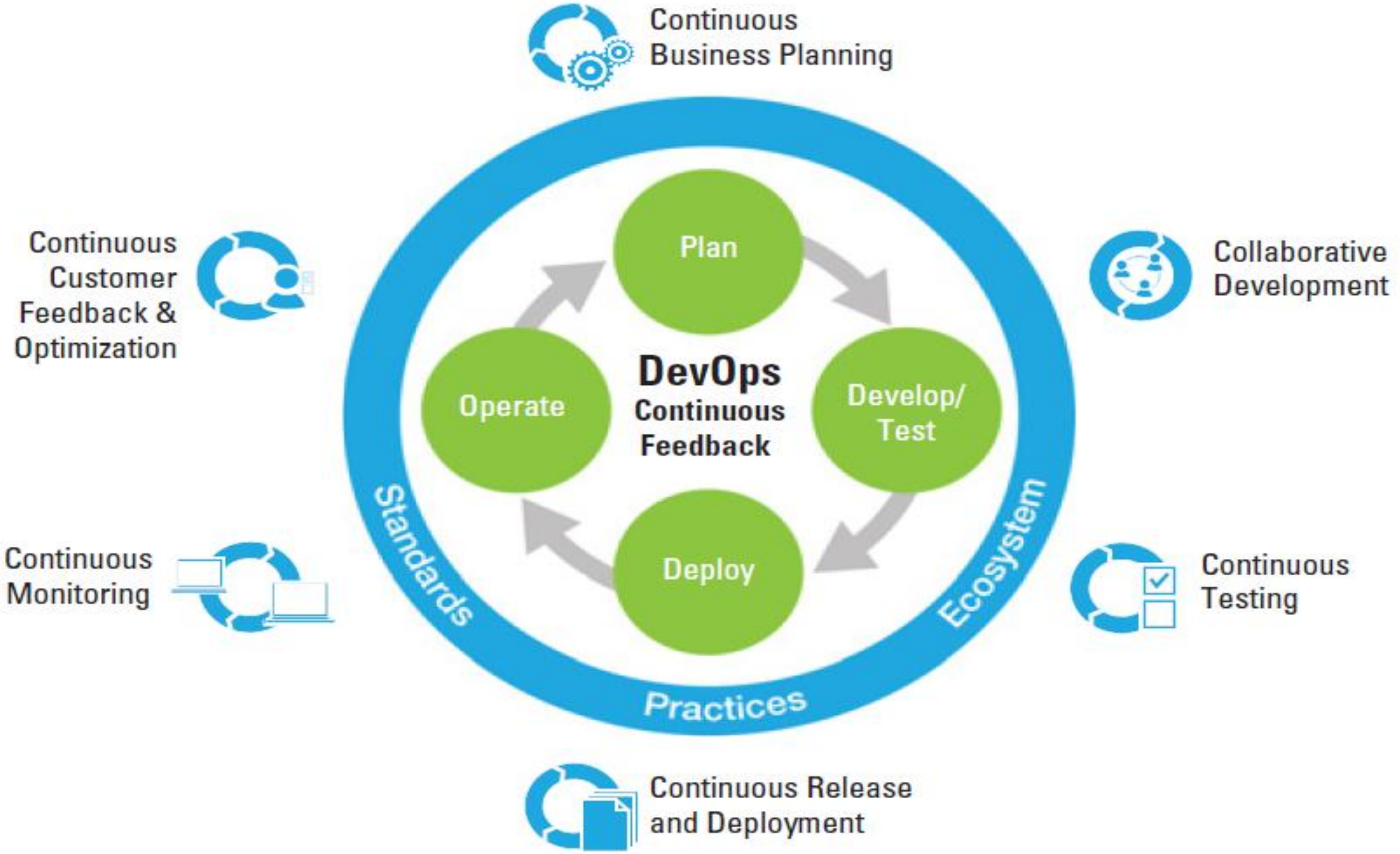
- **The 4 principles:**

✓ Develop and test against production-like systems
✓ Deploy with repeatable, reliable processes
✓ Monitor and validate operational quality
✓ Amplify feedback loops

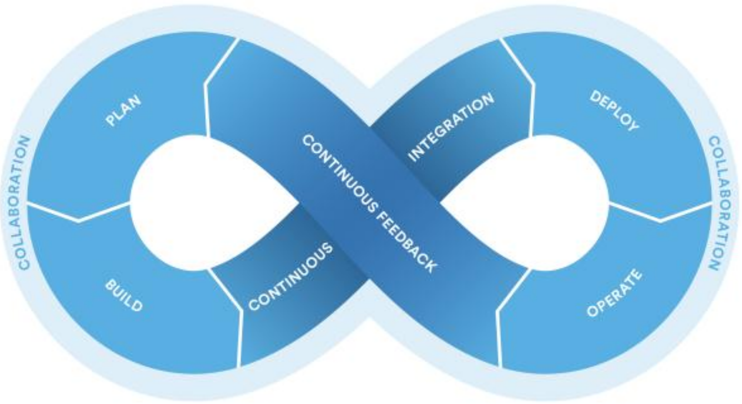


How does DevOps Work?

The Reference Architecture:



How does DevOps Work?



The Reference Architecture:

- **Plan:**

Focuses on establishing business goals and adjusting them based on customer feedback: continuous business planning .

- **Develop/Test:**

Forms the core of development and quality assurance (QA) capabilities. It involves two practices - collaborative development and continuous testing.

- **Deploy**

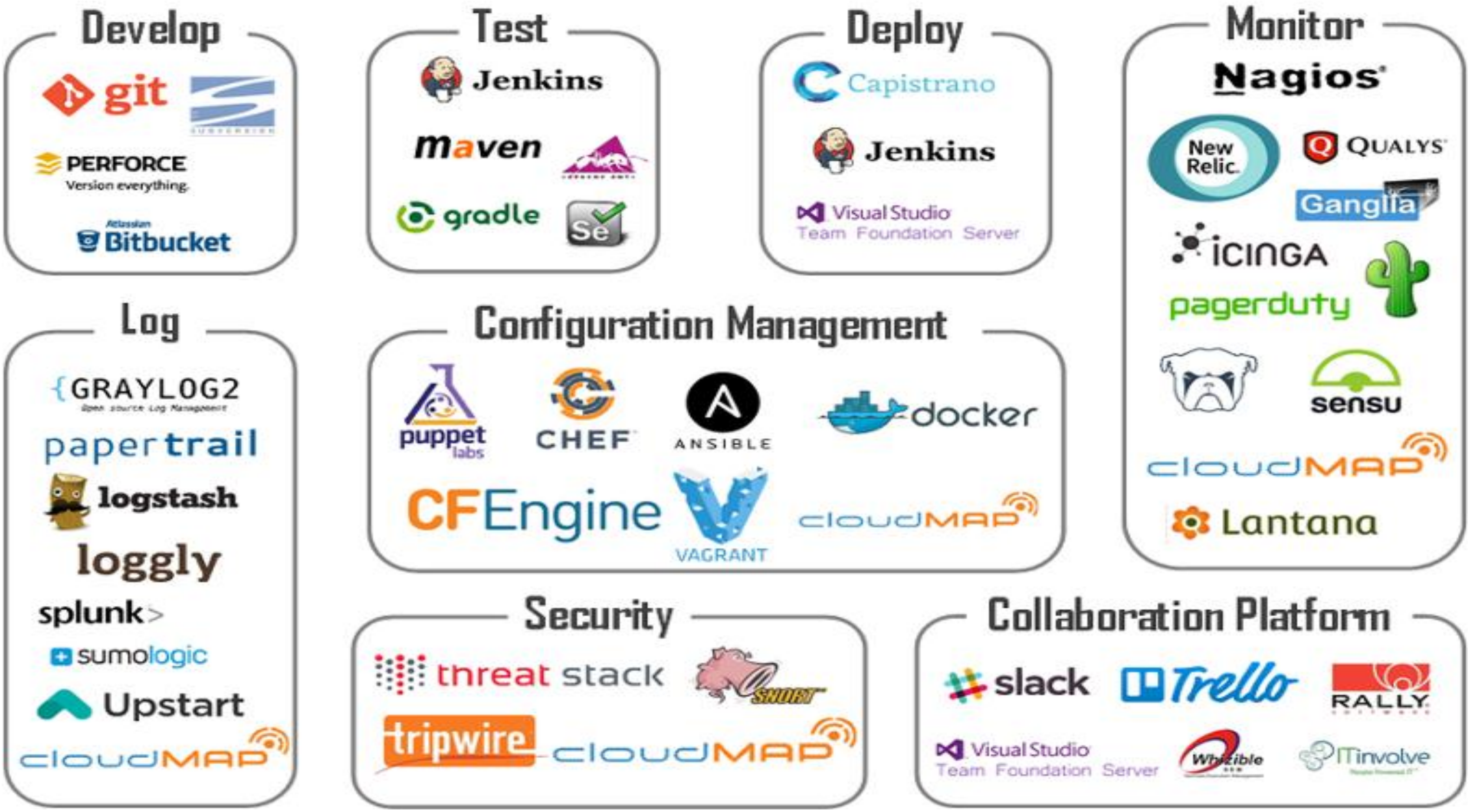
Continuous release and deployment take the concept of continuous integration to the next step

- **Operate**

It involves two practices - continuous monitoring and continuous customer feedback.

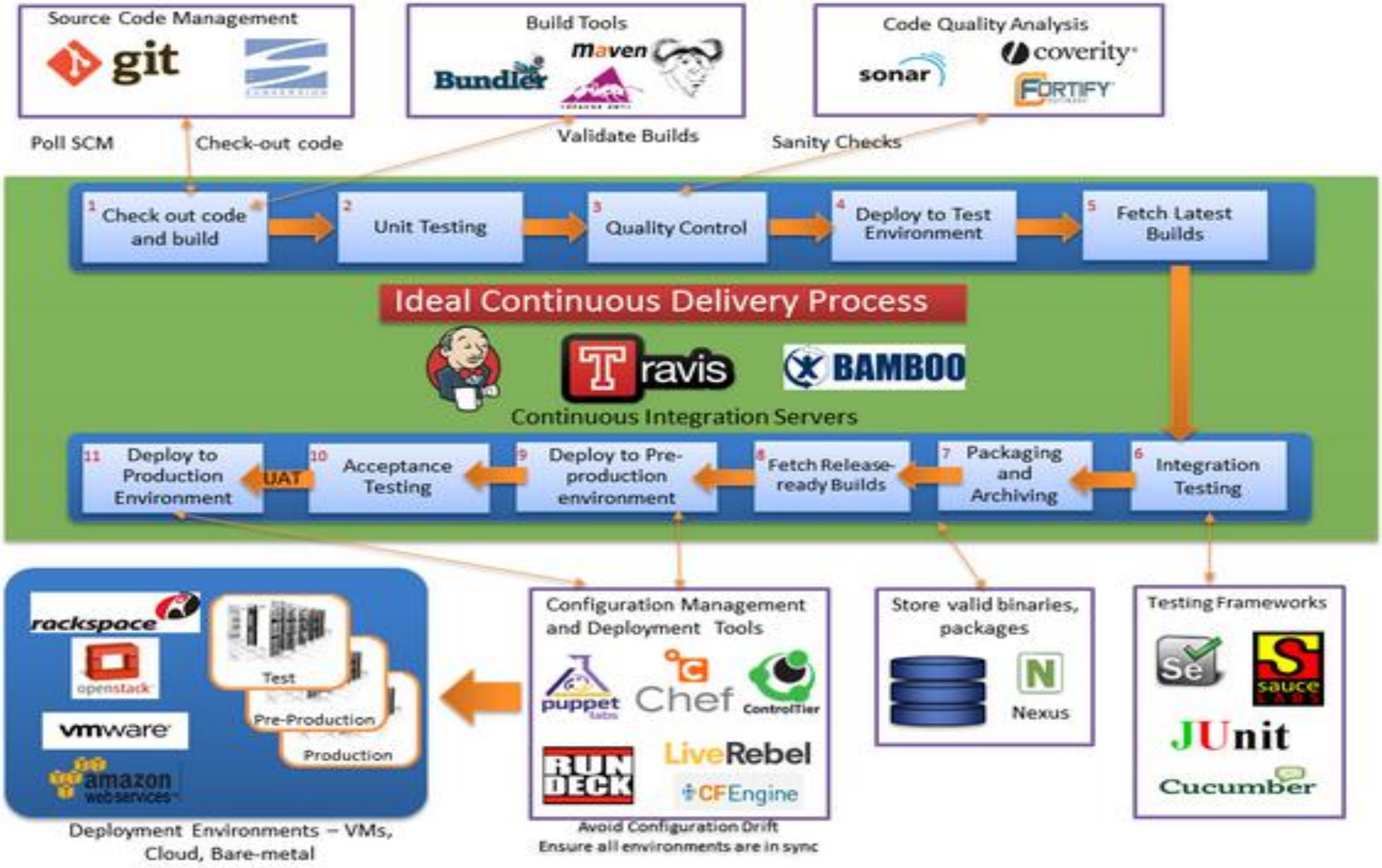
How does DevOps Work?

No tool will magically make the team DevOps.

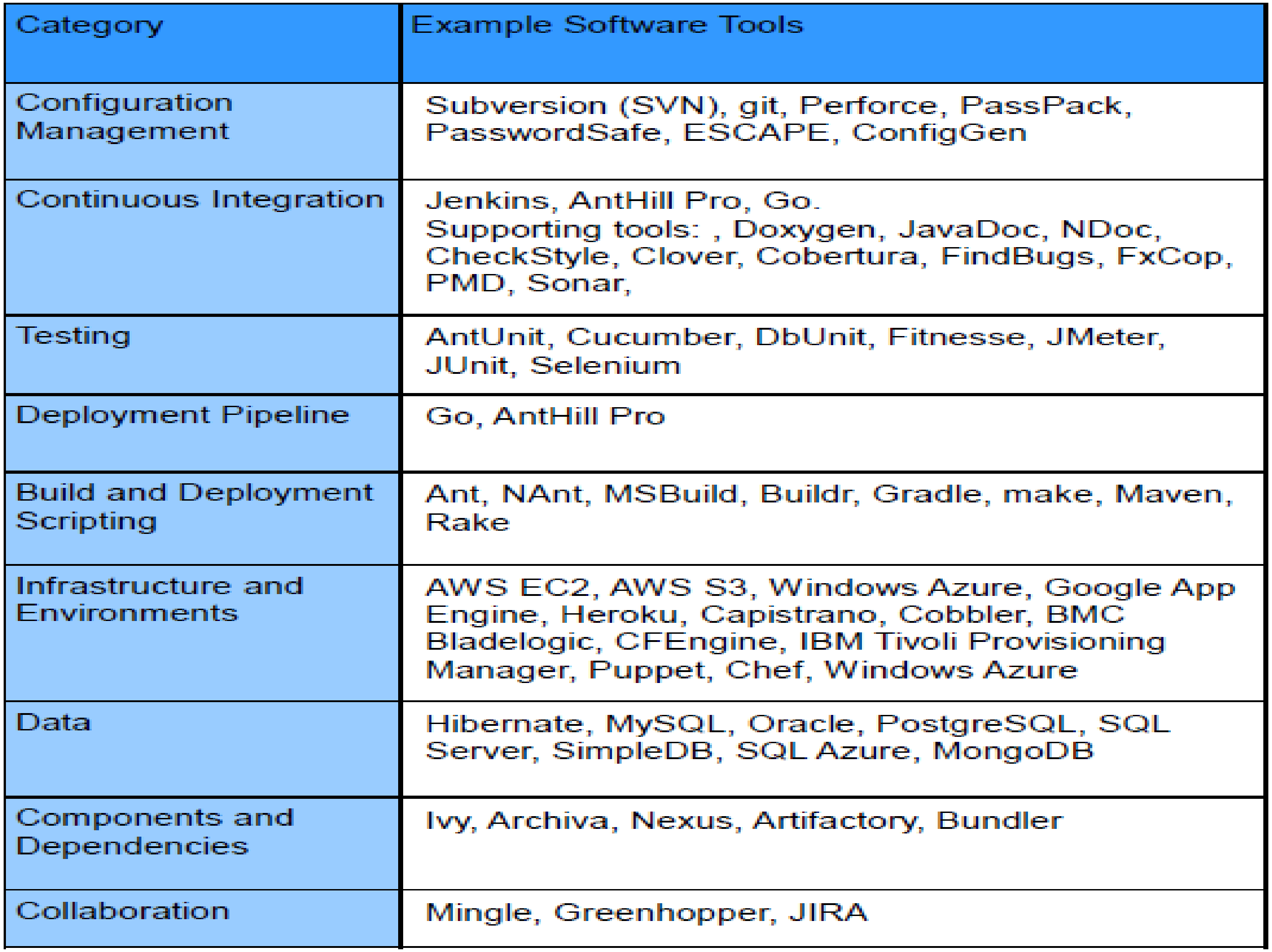


How does DevOps Work?

The Continuous Delivery Pipeline


Source Code Management
git
Build Tools
maven
Bundler
Code Quality Analysis
coverity
sonar
FORTIFY
Poll SCM
Check-out code
Validate Builds
Sanity Checks
1 Check out code and build
2 Unit Testing
3 Quality Control
4 Deploy to Test Environment
5 Fetch Latest Builds
Ideal Continuous Delivery Process
Travis
BAMBOO
Continuous Integration Servers
11 Deploy to Production Environment
UAT
10 Acceptance Testing
9 Deploy to Pre-production environment
8 Fetch Release-ready Builds
7 Packaging and Archiving
6 Integration Testing
rackspace
openstack
vmware
amazon web services
Test
Pre-Production
Production
Deployment Environments – VMs, Cloud, Bare-metal
Configuration Management and Deployment Tools
puppet labs
Chef
ControlTier
RUN DECK
LiveRebel
CFEngine
Avoid Configuration Drift
Ensure all environments are in sync
Store valid binaries, packages
Nexus
Testing Frameworks
Se
sauce LABS
JUnit
Cucumber

Tools for Adopting DevOps



Category	Example Software Tools
Configuration Management	Subversion (SVN), git, Perforce, PassPack, PasswordSafe, ESCAPE, ConfigGen
Continuous Integration	Jenkins, AntHill Pro, Go. Supporting tools: , Doxygen, JavaDoc, NDoc, CheckStyle, Clover, Cobertura, FindBugs, FxCop, PMD, Sonar,
Testing	AntUnit, Cucumber, DbUnit, Fitnesse, JMeter, JUnit, Selenium
Deployment Pipeline	Go, AntHill Pro
Build and Deployment Scripting	Ant, NAnt, MSBuild, Buildr, Gradle, make, Maven, Rake
Infrastructure and Environments	AWS EC2, AWS S3, Windows Azure, Google App Engine, Heroku, Capistrano, Cobbler, BMC Bladelogic, CFEngine, IBM Tivoli Provisioning Manager, Puppet, Chef, Windows Azure
Data	Hibernate, MySQL, Oracle, PostgreSQL, SQL Server, SimpleDB, SQL Azure, MongoDB
Components and Dependencies	Ivy, Archiva, Nexus, Artifactory, Bundler
Collaboration	Mingle, Greenhopper, JIRA

DevOps- Implementation and Tools

DevOps Practices

DevOps Practices:

- Infrastructure as Code (IaC)
- Source Code Management
- Continuous Integration
- Automated Testing
- Continuous Deployment
- Release Management

Infrastructure as Code (IaC)

IAC is not a product, it's a methodology

Infrastructure as Code (IaC)

- Organizations looking for faster deployments treat infrastructure like software
- Infra as code that can be managed with the same tools and processes software developers use, such as version control, continuous integration, code review and automated testing.
- Makes infrastructure changes more easy, rapid, safe and reliable.

To properly embrace IAC, you need three things:

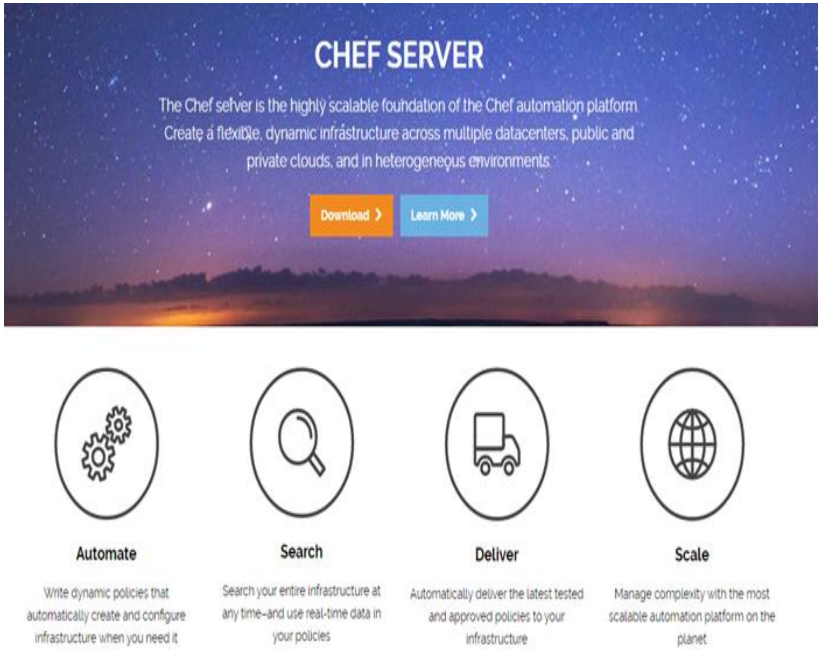
- agile development processes
- a DevOps environment
- the tools to write the code.

Example : Chef / Puppet

Learn the Tool – Chef and Puppet

Infrastructure as Code (IaC) Tool

https://www.chef.io/chef/



CHEF SERVER

The Chef server is the highly scalable foundation of the Chef automation platform. Create a flexible, dynamic infrastructure across multiple datacenters, public and private clouds, and in heterogeneous environments.

Automate

Write dynamic policies that automatically create and configure infrastructure when you need it

Search

Search your entire infrastructure at any time–and use real-time data in your policies

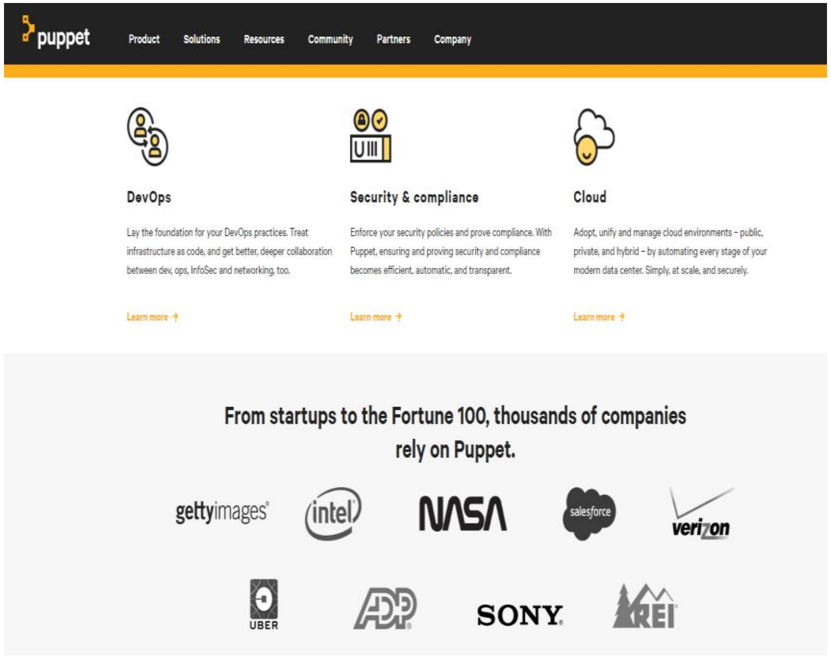
Deliver

Automatically deliver the latest tested and approved policies to your infrastructure

Scale

Manage complexity with the most scalable automation platform on the planet

https://puppet.com



DevOps

Lay the foundation for your DevOps practices. Treat infrastructure as code, and get better, deeper collaboration between dev, ops, InfoSec and networking, too.

Security & compliance

Enforce your security policies and prove compliance. With Puppet, ensuring and proving security and compliance becomes efficient, automatic, and transparent.

Cloud

Adopt, unify and manage cloud environments – public, private, and hybrid – by automating every stage of your modern data center. Simply, at scale, and securely.

From startups to the Fortune 100, thousands of companies rely on Puppet.

Learn the Tool – Chef

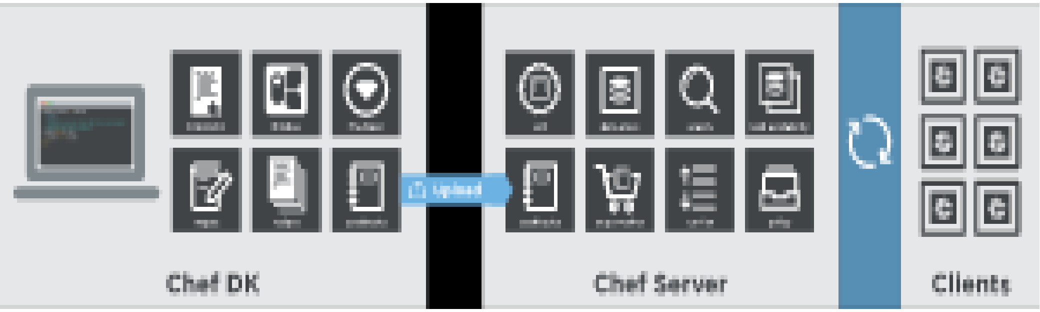
Chef:

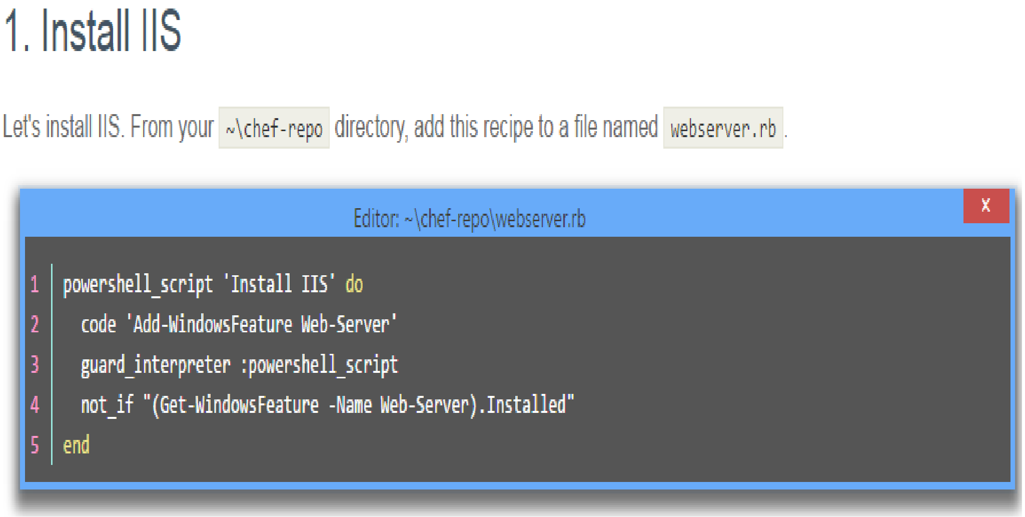
- Express your infrastructure policy – how your software is delivered and maintained on your servers – as code.

- The normal Chef workflow involves managing servers remotely from your workstation.

- A Chef resource describes some piece of infrastructure, such as a file, a template, or a package.

- A Chef recipe is a file that groups related resources, such as everything needed to configure a web server, database server, or a load balancer.





1. Install IIS

Let's install IIS. From your `~\chef-repo` directory, add this recipe to a file named `webserver.rb`.

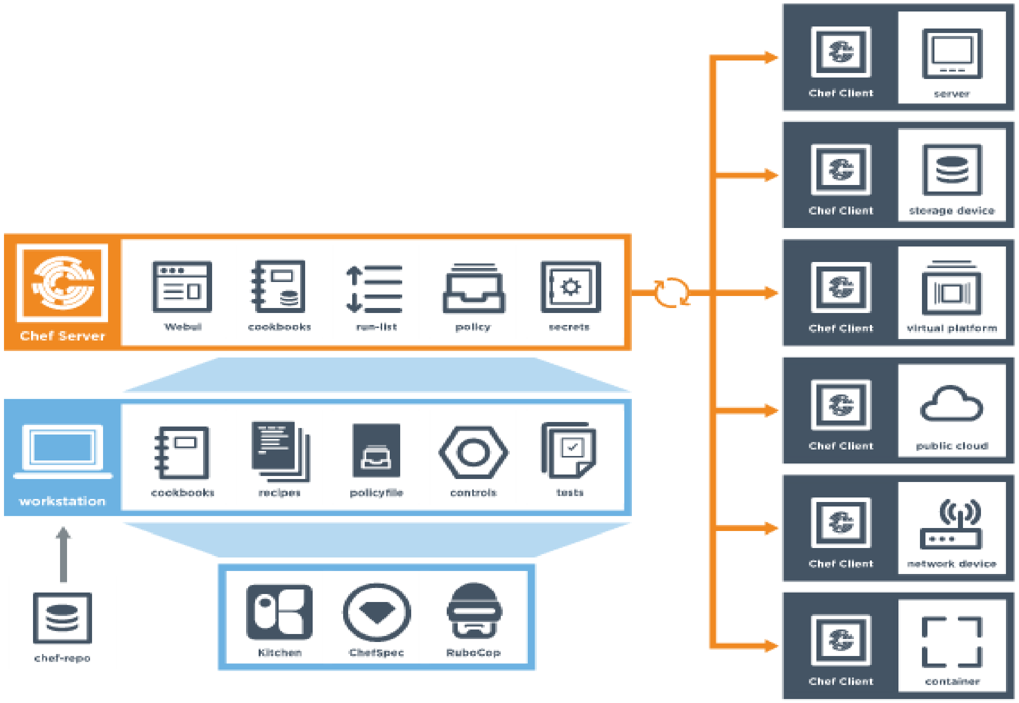
Editor: ~\chef-repo\webserver.rb

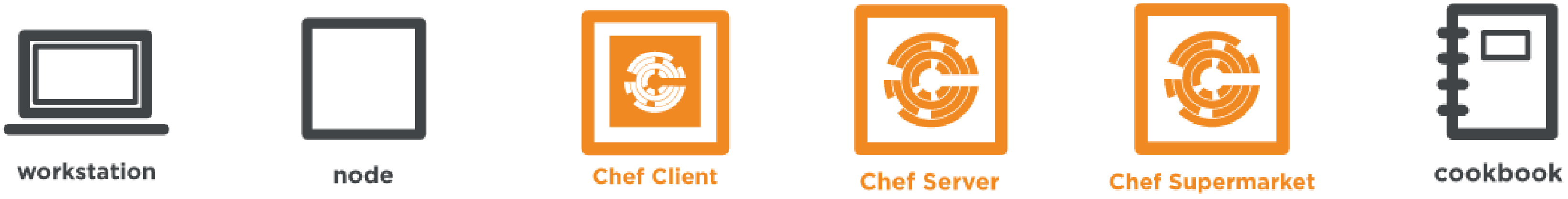
```
powershell_script 'Install IIS' do
  code 'Add-WindowsFeature Web-Server'
  guard_interpreter :powershell_script
  not_if "(Get-WindowsFeature -Name Web-Server).Installed"
end
```

Learn the Tool – Chef

Relationships between the various elements of Chef :

- Includes the nodes, the server, and the workstation.
- These elements work together to provide the chef-client the information and instruction that it needs so that it can do its job.

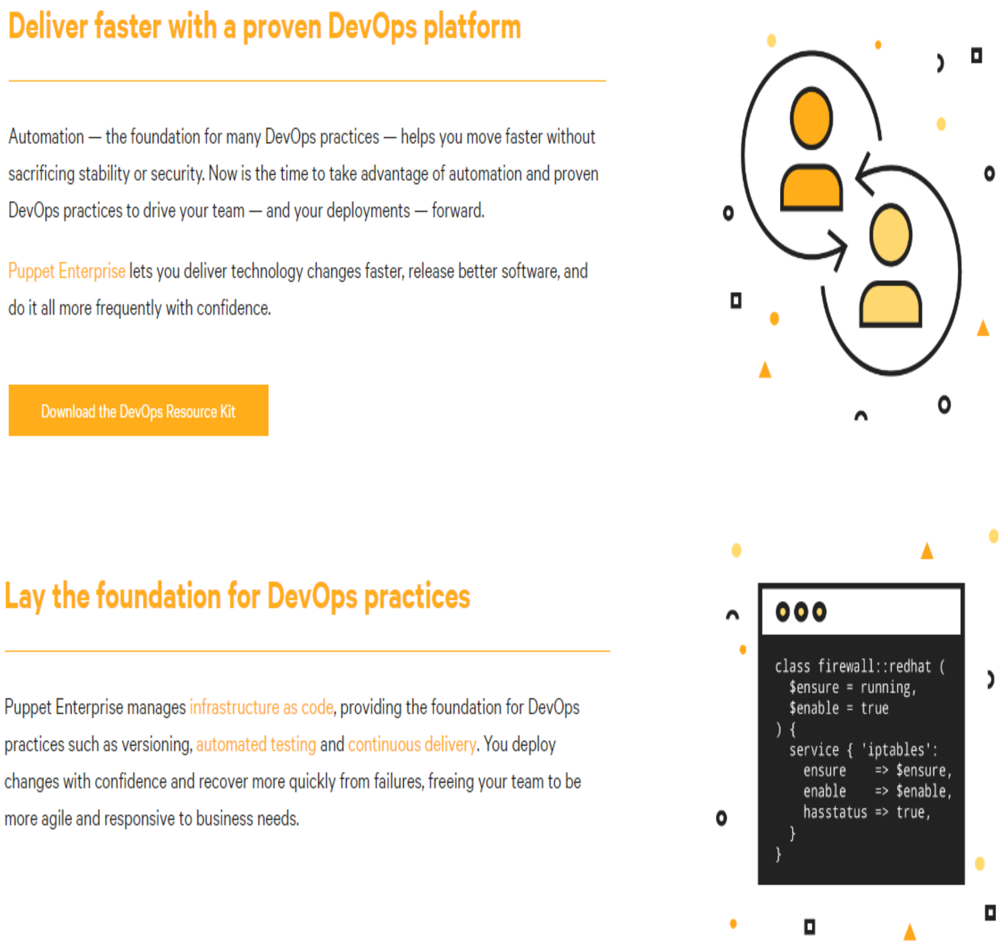




Learn the Tool – Puppet

Puppet:

- Lets you define the desired state of your infrastructure and what you want it to do.
- Puppet automatically enforces that desired state and remediates any unexpected changes.
- Deploy faster, with greater reliability, because one no longer have to map out and manually deploy every step
- *Capabilities:*
  - Orchestration
  - Automated provisioning
  - Configuration automation
  - Visualization & reporting
  - Code management
  - Node management
  - Role-based access control



Deliver faster with a proven DevOps platform

Automation — the foundation for many DevOps practices — helps you move faster without sacrificing stability or security. Now is the time to take advantage of automation and proven DevOps practices to drive your team — and your deployments — forward.

Puppet Enterprise lets you deliver technology changes faster, release better software, and do it all more frequently with confidence.

Lay the foundation for DevOps practices

Puppet Enterprise manages infrastructure as code, providing the foundation for DevOps practices such as versioning, automated testing and continuous delivery. You deploy changes with confidence and recover more quickly from failures, freeing your team to be more agile and responsive to business needs.

```
class firewall::redhat (
  $ensure = running,
  $enable = true
) {
  service { 'iptables':
    ensure    => $ensure,
    enable    => $enable,
    hasstatus => true,
  }
}
```

Source Code Management

Source Code Management:

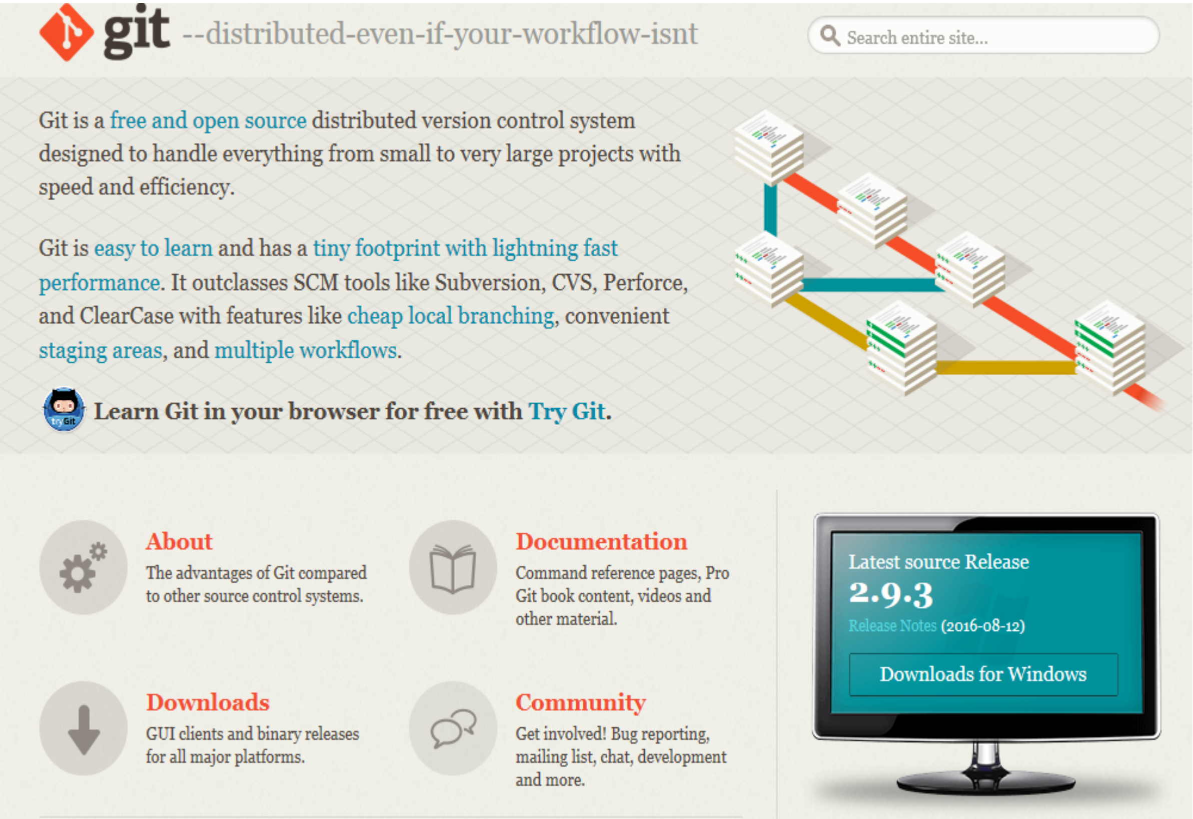
- Continually merges source code updates from all developers on a team into a shared mainline.
- A source code manager (SCM) is a software tool used by teams of programmers to manage source code.
- SCMs are used to track revisions in software.
- Each revision is given a timestamp and includes the name of the person who is responsible for the change.
- Various revisions may be compared, stored, and merged with other revisions.

Example : GIT

Learn the Tool – GIT

Source Code Management Tool

https://git-scm.com/



git --distributed-even-if-your-workflow-isnt

Search entire site...

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

Learn Git in your browser for free with Try Git.

About
The advantages of Git compared to other source control systems.

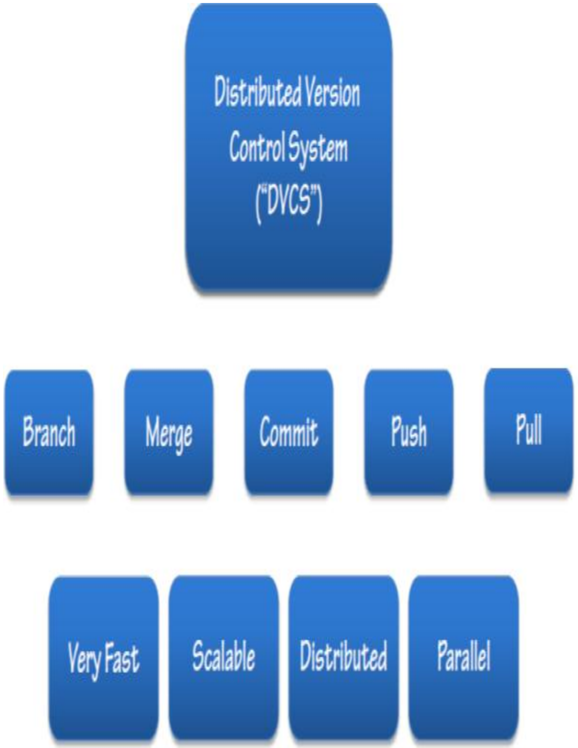
Documentation
Command reference pages, Pro Git book content, videos and other material.

Downloads
GUI clients and binary releases for all major platforms.

Community
Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release
2.9.3
Release Notes (2016-08-12)

Downloads for Windows



Distributed Version Control System ("DVCS")

Branch

Merge

Commit

Push

Pull

Very Fast

Scalable

Distributed

Parallel

Learn the Tool – GIT

Distributed Version Control using GIT

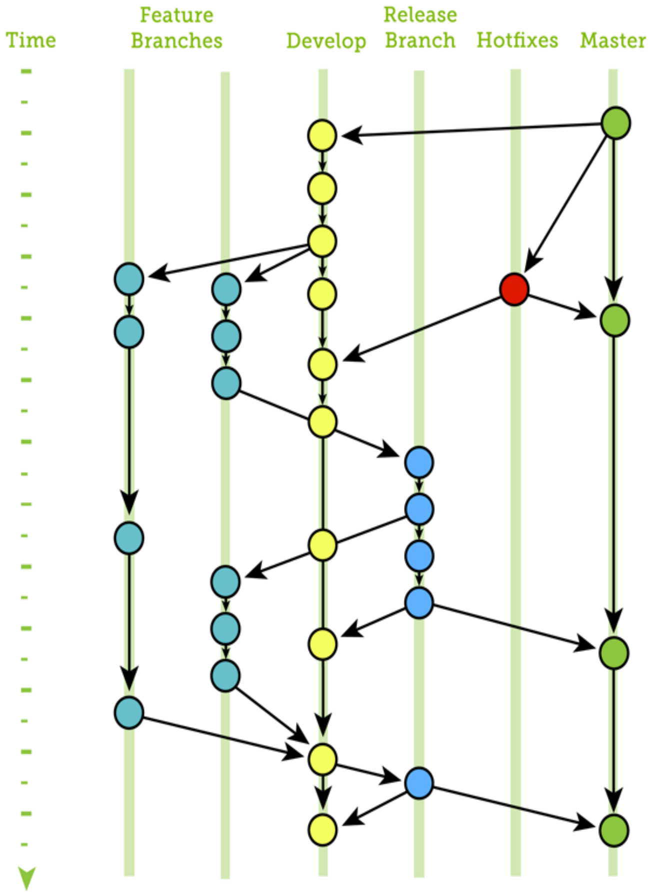
- Git is a distributed version control system.
- A distributed version control system does not necessarily have a central server which stores the data.

- The user can copy an existing repository. This copying process is typically called cloning

- Git allows the user to synchronize the local repository with other (remote) repositories.

- Users with sufficient authorization can push changes from their local repository to remote repositories.

- They can also fetch or pullchanges from other repositories to their local Git repository.



Fail Fast

Continuous Integration (CI)

Continuous Integration

- Continually merges source code updates from all developers on a team into a shared mainline.

- Prevents a developer's local copy of a software project from drifting too far afield as new code is added by others, avoiding catastrophic merge conflicts.

- CI involves a centralized server that continually pulls in all new source code changes as developers commit them and builds the software application from scratch, notifying the team of any failures in the process.

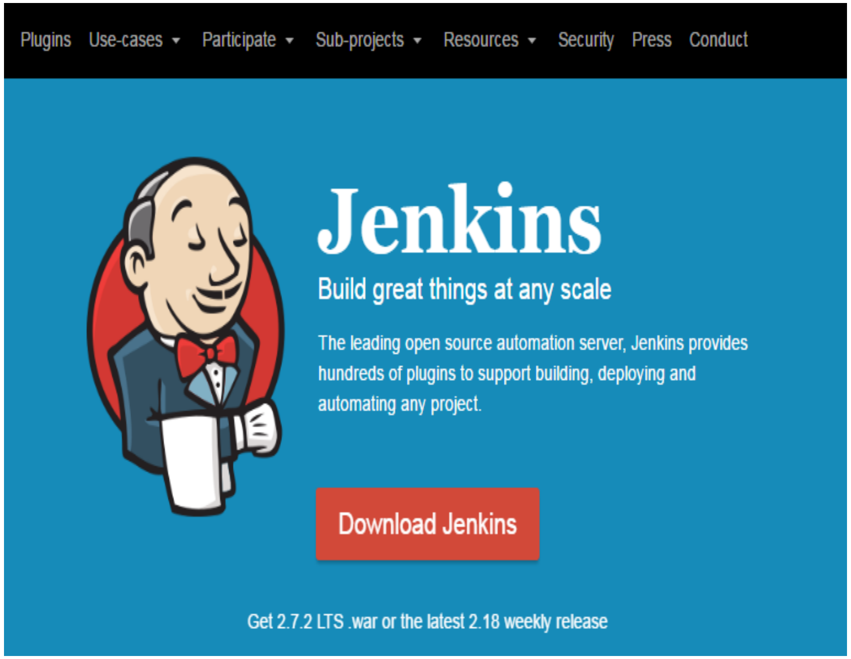
- If a failure is seen, the development team is expected to refocus and fix the build before making any additional code changes.

Example : Jenkins / Bamboo / Go

Learn the Tool – Jenkins

Continuous Integration (CI) Tool

https://jenkins.io

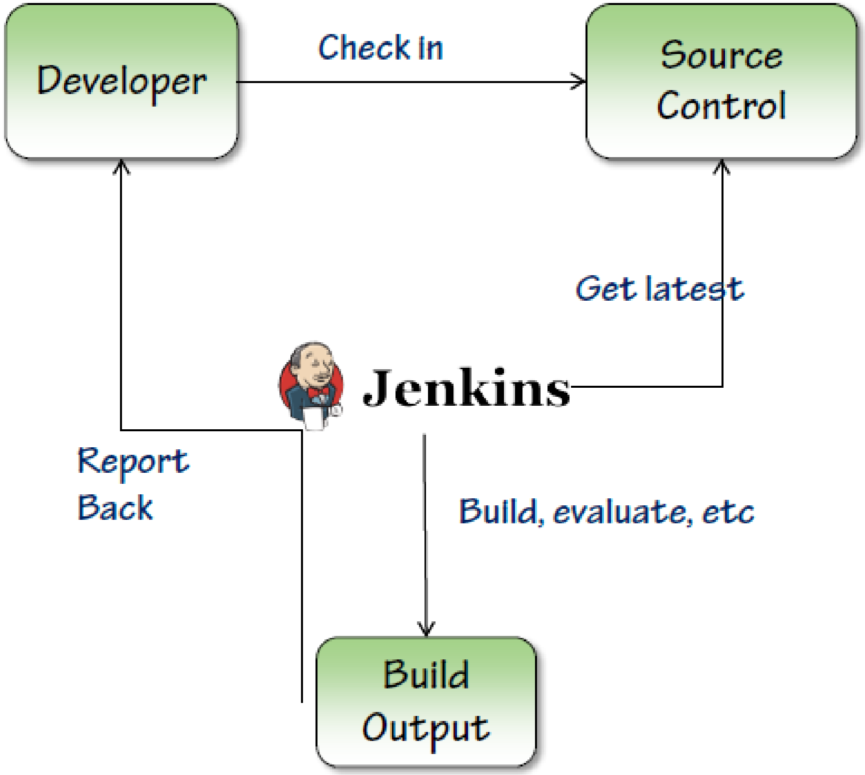


Jenkins

Build great things at any scale

The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating any project.

Get 2.7.2 LTS .war or the latest 2.18 weekly release



Learn the Tool – Jenkins

Open Source CI Tool:

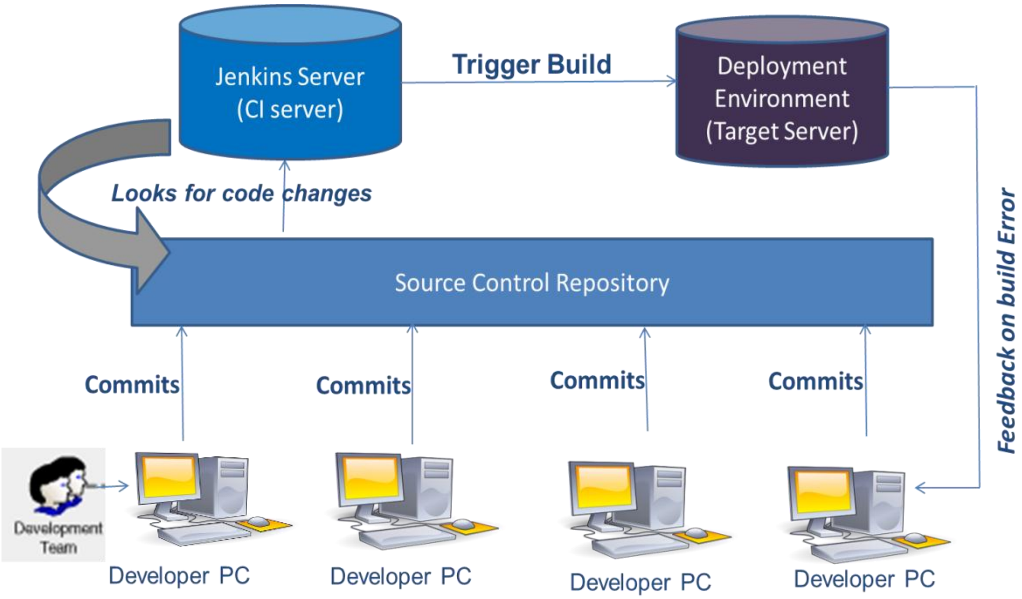
- Jenkins is an open source continuous integration tool written in java developed by Kohsuke Kawaguchi.

- Monitors the change in the source control systems like SVN, CVS, etc.

- Builds the application using various build tools like ANT, MAVEN, etc.

- Provides a fresh build whenever there is a change in the source control system

- Sends messages on the status of the build through Email, SMS, etc.



- Can support software releases, documentation, monitoring, and a number of use case secondary to continuous integration

Automated Testing

Automated Testing

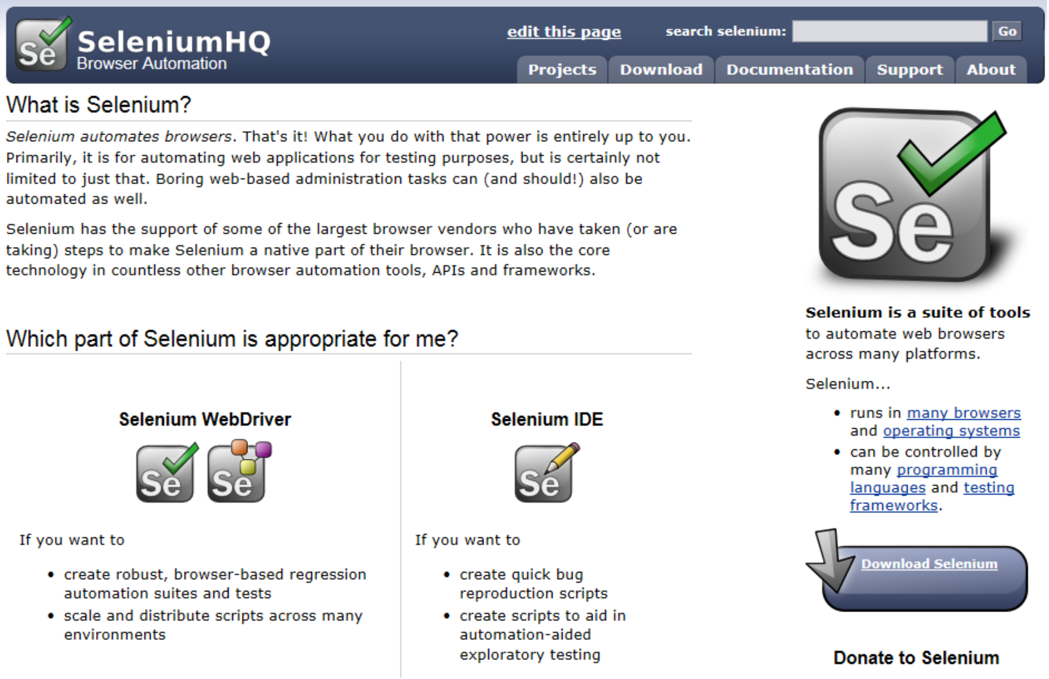
- The objective of automated testing is to simplify as much of the testing effort as possible with a minimum set of scripts.
- Automated testing tools are capable of executing repeatable tests, reporting outcomes, and comparing results with faster feedback to the team.
- Automated tests perform precisely the same operation each time they are executed, thereby eliminating human errors – and can be run repeatedly, at any time of day.
- Includes testing for each environment in the pipeline
  - ➢ Dev. Environment
    - ○ Unit, Sanity Testing
  - ➢ CI Environment
    - ○ Incremental Integration Testing
  - ➢ QA Environment
    - ○ Functional , Usability Testing
  - ➢ Compatibility Testing

Example : Selenium

Learn the Tool – Selenium

Automated Testing Tool

http://docs.seleniumhq.org/



What is Selenium?

Selenium automates browsers. That's it! What you do with that power is entirely up to you. Primarily, it is for automating web applications for testing purposes, but is certainly not limited to just that. Boring web-based administration tasks can (and should!) also be automated as well.

Selenium has the support of some of the largest browser vendors who have taken (or are taking) steps to make Selenium a native part of their browser. It is also the core technology in countless other browser automation tools, APIs and frameworks.

Which part of Selenium is appropriate for me?

Selenium WebDriver

If you want to

- create robust, browser-based regression automation suites and tests
- scale and distribute scripts across many environments

Selenium IDE

If you want to

- create quick bug reproduction scripts
- create scripts to aid in automation-aided exploratory testing

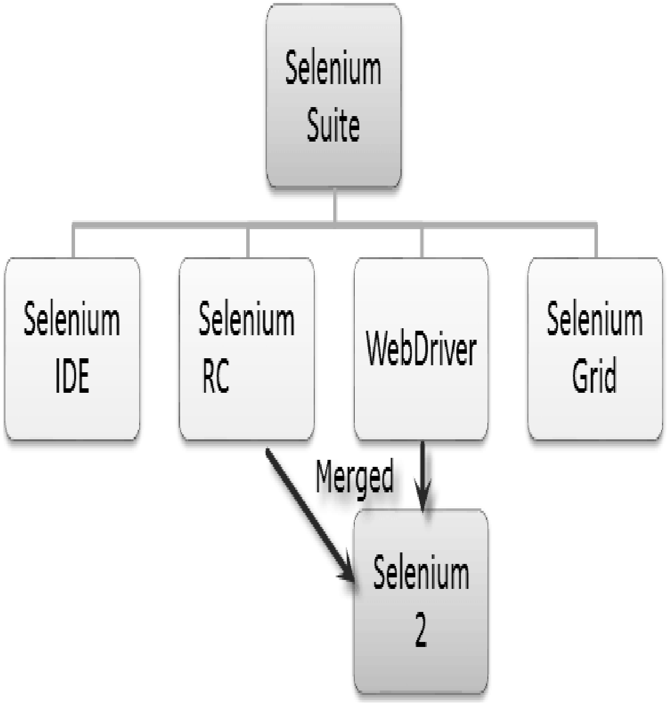
Selenium is a suite of tools to automate web browsers across many platforms.

Selenium...

- runs in many browsers and operating systems
- can be controlled by many programming languages and testing frameworks.

Download Selenium

Donate to Selenium



Learn the Tool – Selenium

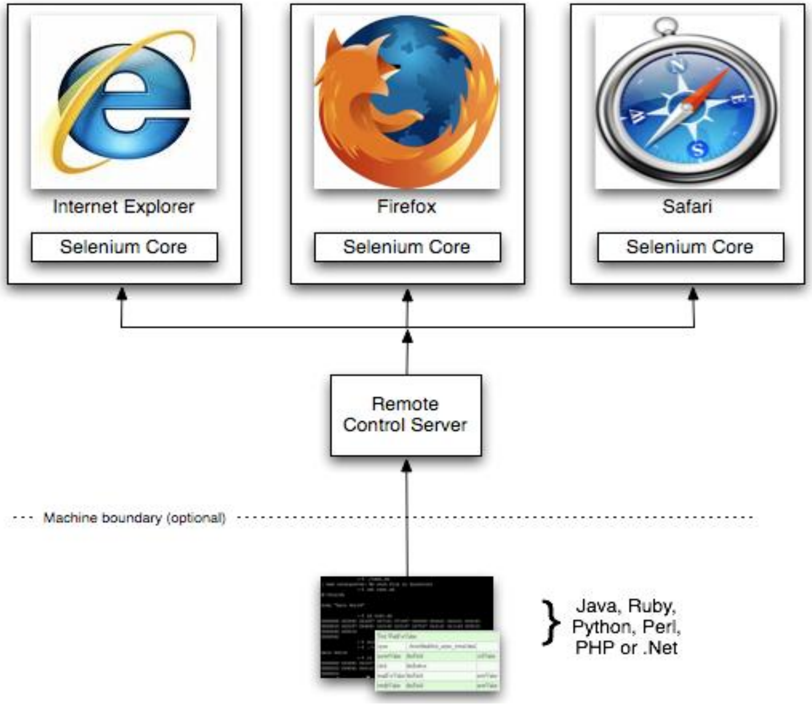
Overview of Selenium IDE

- Allows you to record, play back, edit, and debug tests in browser.

- Generate scripts from recorded user actions in most of the popular languages like Java, C#, Perl, Ruby etc.

- Run them using Selenium Web Driver.

- Allows the user to pick from a list of assertions and verifications for the selected location



- Selenium Remote Control (RC) is a test tool that allows you to write automated web application UI tests in any programming language against any HTTP website using any mainstream JavaScript-enabled browser.

Continuous Deployment and Release Management

Continuous Deployment & Release Management:

- Continuous deployment and release management raise the concept of continuous integration to the next level enabling creation of the delivery pipeline .

- This pipeline automates continuous deployment of software to QA environment, then to production in an efficient manner.

- Continuous release and deployment makes it possible to release new features to customers and users at the earliest possible..

- Correct selection of tooling and processes make up the core of DevOps to facilitate continuous integration, continuous release, and continuous deployment.

DevOps Typical Stories

Software Delivery Lifecycle (Integrated Development and Operations Lifecycles)

Story 0: Dev and Ops collaborate to develop environment definitions

- Value: Ensures that Dev understands and deals with production-like environments; avoids architectural miscommunications

Story 1: Dev continuously delivers application changes to a realistic environment for testing

- Value: Shared technology ensures testable environments and script reuse for repeatable delivery; Test org always has known good builds,properly deployed.

Story 2: Release Applications from Test /Staging to production

- Value: Shared technology and automation ensures no gratuitous differences between dev/test and prod.

Story 3: Collaborative incident management

- Value: ensures an integrated process for reproducing and resolving defects and issues between dev, test,and ops.

Story 4: Dev and Ops use the same analysis and instrumentation in dev, test,and ops

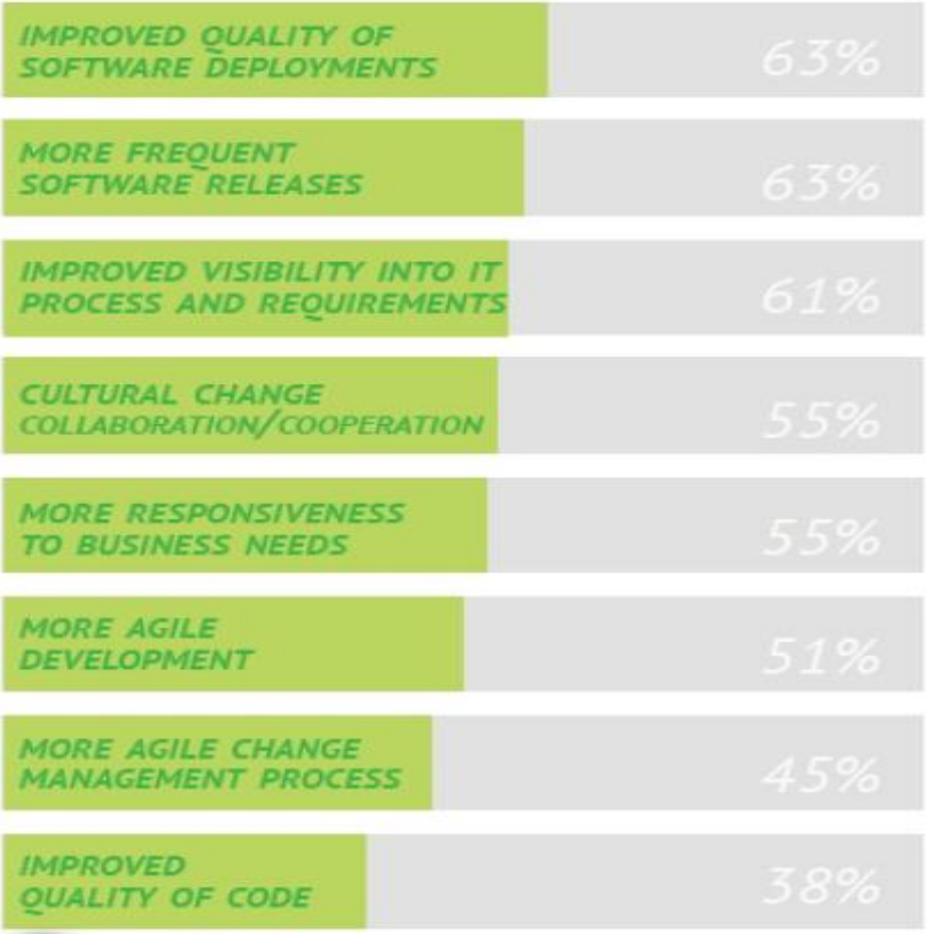
- Value: Ensures a common understanding of quality and performance (and no fingerpointing)

Story 5: Manage the entire delivery pipeline with end-to-end visibility and dashboards

- Value: Enables end-to-end delivery metrics and visibility into bottlenecks.

DevOps Benefits

Key benefits identified by the organizations that implement DevOps


IMPROVED QUALITY OF SOFTWARE DEPLOYMENTS
MORE FREQUENT SOFTWARE RELEASES
IMPROVED VISIBILITY INTO IT PROCESS AND REQUIREMENTS
CULTURAL CHANGE COLLABORATION/COOPERATION
MORE RESPONSIVENESS TO BUSINESS NEEDS
MORE AGILE DEVELOPMENT
MORE AGILE CHANGE MANAGEMENT PROCESS
IMPROVED QUALITY OF CODE