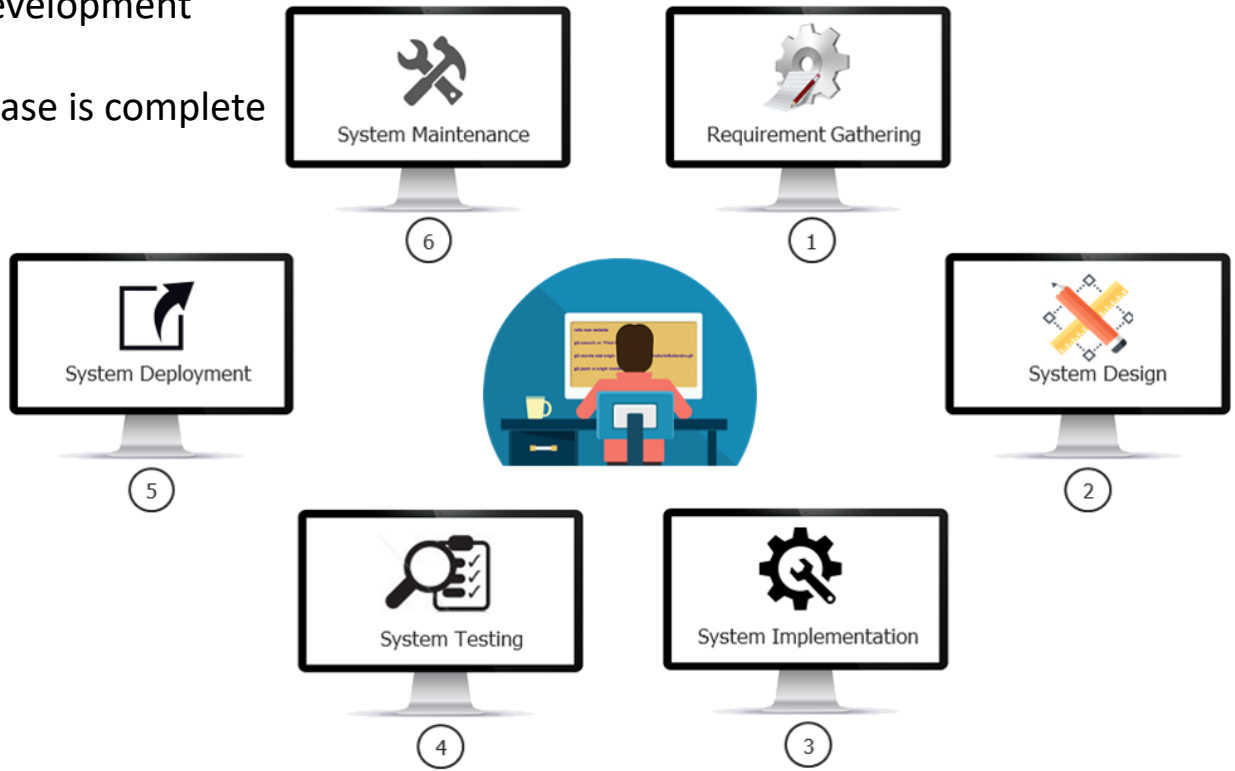
Introduction to DevOps

Traditional Development Models: Waterfall

Waterfall Model: New phase in the development process begins only if the previous phase is complete



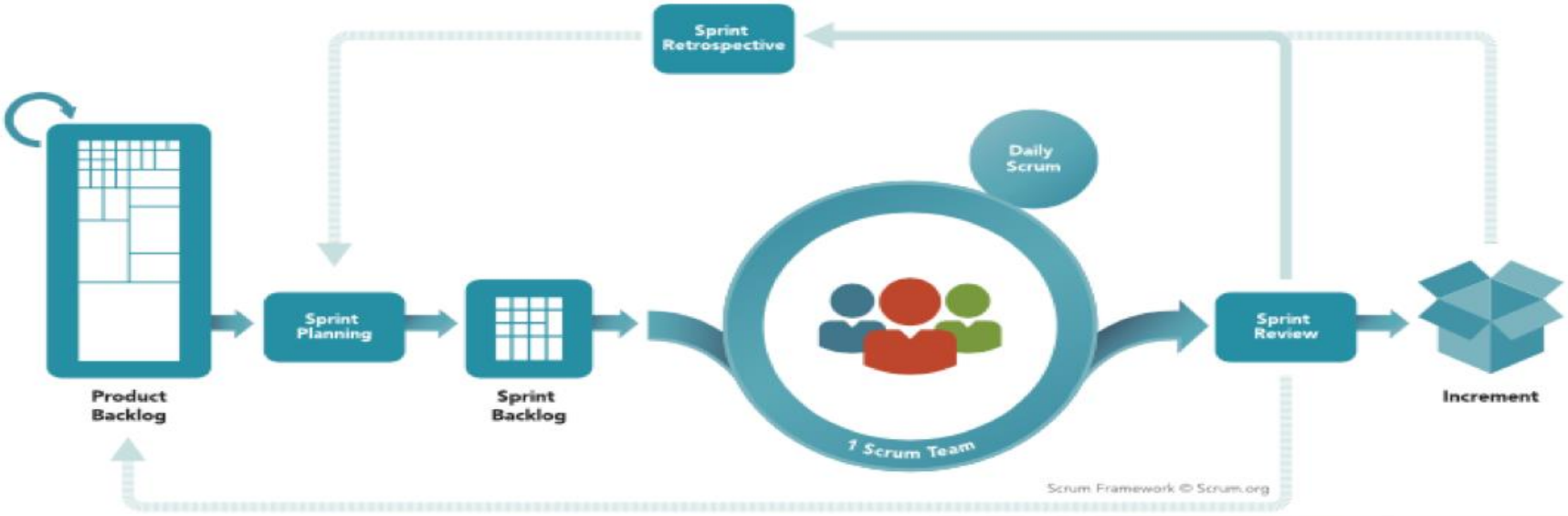
DevOps

- DevOps is short for **Dev**elopment **Op**eration**s**
- It focuses on collaboration between developers and other parties involved in building, deploying, operating, and maintaining software systems
- It is about defining a flow from development through full scale operation of a system
- It is about Systems Thinking with feedback to earlier stages of a DevOps workflow
- The emphasis is on automating processes required to release and change software
    - With a view to rapid, frequent, and reliable releases

Agile

- Agile Software Development is an umbrella term for a set of methods and practices based on the values and principles expressed in the Agile Manifesto
- Solutions evolve through collaboration between self-organizing, cross-functional teams utilizing the appropriate practices for their context

- From agilealliance.org



- Some of the principles of Agile manifesto that lead to a DevOps environment
  - Our highest priority is to satisfy the customer through early and continuous delivery of valuable software
  - 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
- DevOps not only talks about collaboration between Development and Operations, but improved collaboration between all teams including business
- DevOps also promotes using a lot of automation to complete the many tasks and steps that go into making a deliverable potentially shippable increment
- Using tools to replace the non-human steps like build creation, environment setup, trigger test cases, automated deployments multiple times in a day

What DevOps is Not

- A title
- A separate team
- A tool
- Only culture
- Only automation
- Anarchy
- A one size fits all strategy

Evolution of IT

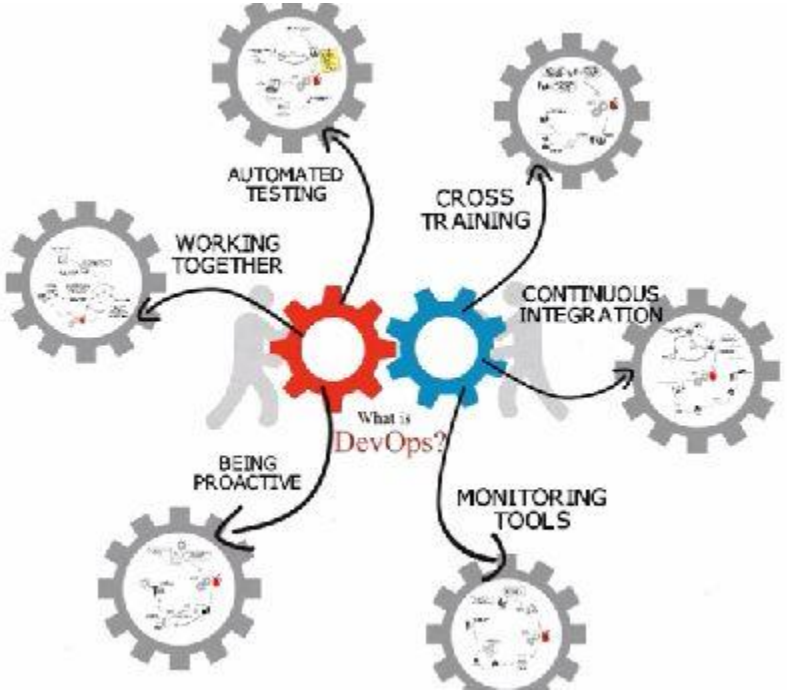
Development Process	Application Architecture	Deployment & Packaging	Application Infrastructure
Waterfall	Monolithic	Physical Servers	Datacenter
Agile	N-Tier	Virtual Servers	Hosted
DevOps	**Microservices**	**Containers**	**Cloud**

DevOps Goals

- Smaller, more frequent releases
- Reduced effort and risks
- Reduced cost of product iterations and delays
- A culture of communication and collaboration
- Consistency and speed through automation

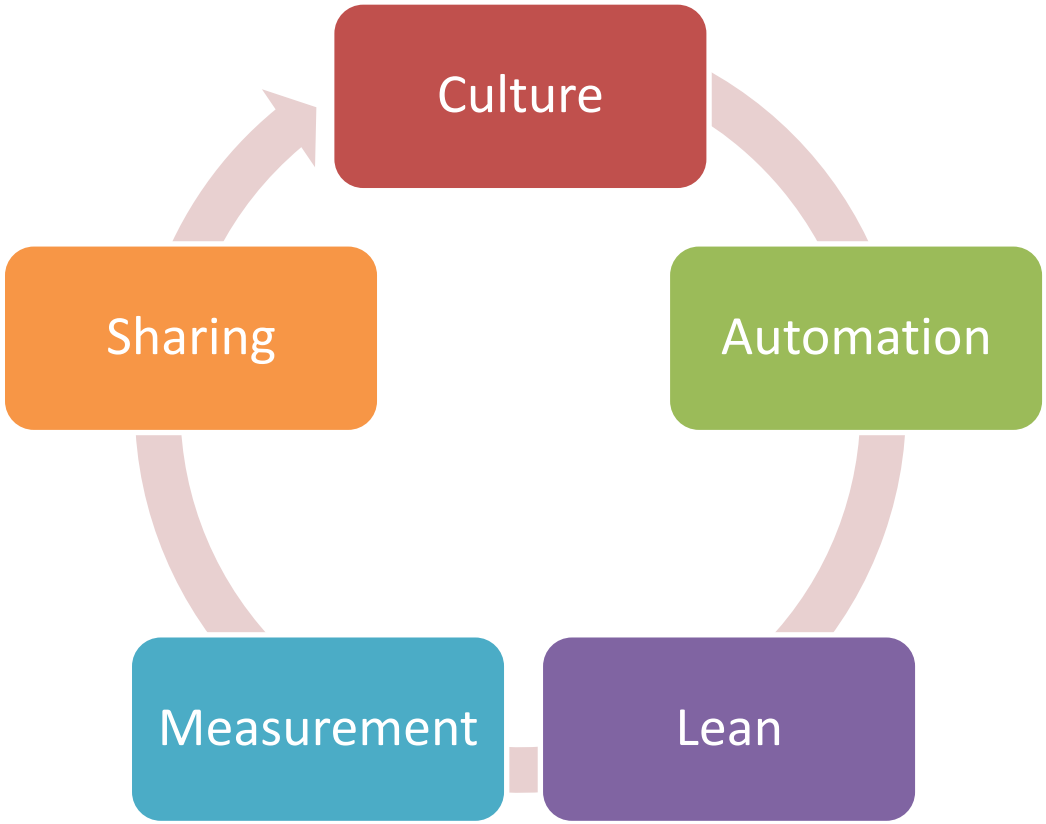
Characteristics of DevOps

- Treating Infrastructure and code as the same thing: Software
- That Software is built and deployed to achieve fast feedback cycles
- Those feedback cycles are achieved through continuous delivery pipelines
- Those pipelines contain the automated coded policy of the organization
- Those pipelines are as human free as possible
- Humans share knowledge and work creatively with work centered around pipelines
- Knowledge silos are broken in favor of supporting a continuous pipeline flow



- Logging & Traceability of all changes

DevOps Values



DevOps Stakeholders

- Architects,
- Business representatives,
- Customers,
- Product Managers,
- Project Managers,
- Quality - QA,QC
- Suppliers

Dev

- Systems Engineers,
- Systems Administrators
- Information security,,
- IT Operation engineers,
- Release engineers,
- Database administrators,
- Network engineers,
- Support Professionals,
- Suppliers

Ops

Why IT & Business are Driving DevOps

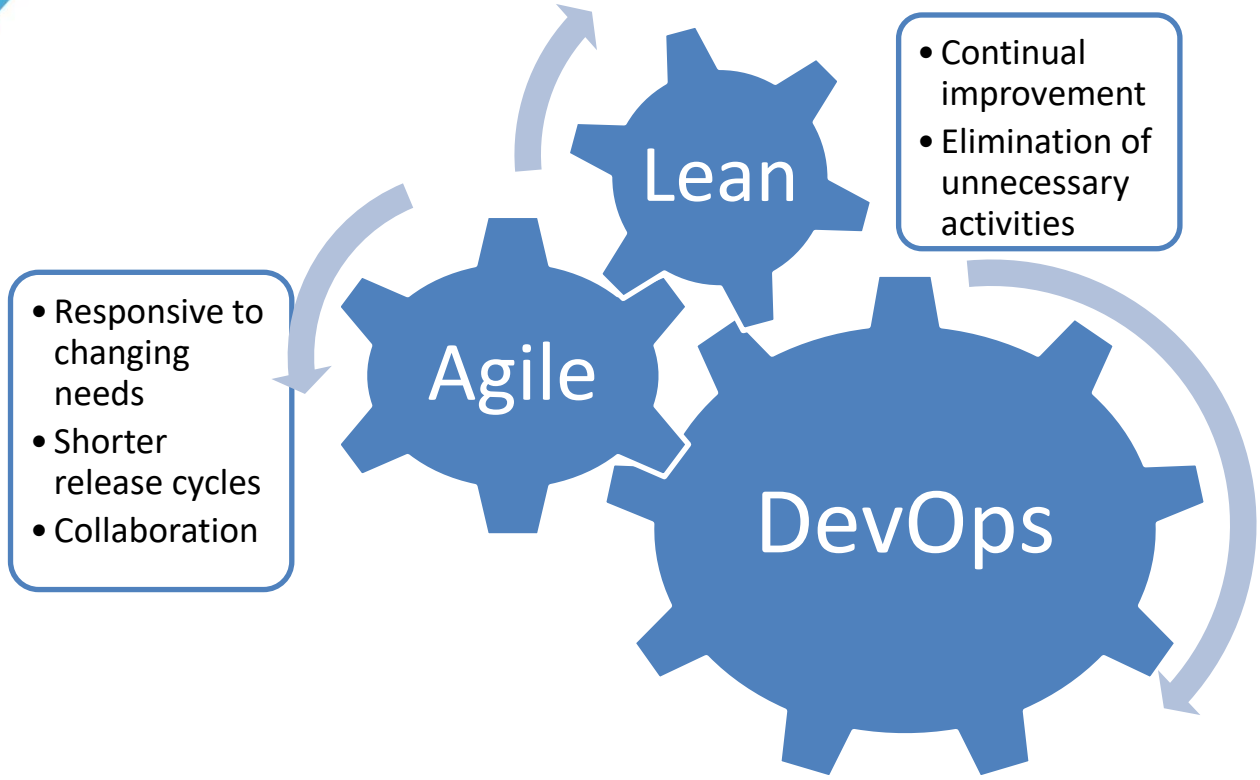
- Every business has become a tech business
- IoT is rapidly increasing
- Consumers have developed “app” mentalities
- Customers value outcomes, not products
- Time to value is replacing time to market
- Intelligent data must shape direction quickly
- Customer delight is more important than customer satisfaction

Benefits of DevOps

- A sustainable, successful business is more than the development and operations teams
- Shift focus from what to why – question, innovate, collaborate
- Repeatable, reliable, and predictable
- Faster reaction to business need
- Higher return on investment in software
- DevOps principles can also be applied to the business


DEPLOYMENTS AT AMAZON.COM
11.6s
Mean time between deployments (weekday)
1,079
Max number of deployments in a single hour
10,000
Mean number of hosts simultaneously receiving a deployment
30,000
Max number of hosts simultaneously receiving a deployment

DevOps Fits with Lean & Agile Scrum Practices



Pillar of Effective DevOps

Collaboration

- Targeting a specific outcome through supporting interactions and the input of multiple people

Affinity

- Shared organizational goals, empathy and learning between different groups of people

Tools

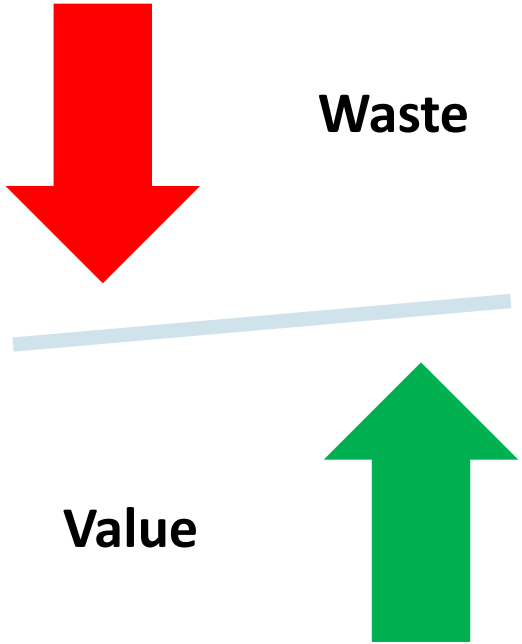
- An accelerator and cost saver but must fit with working methods

Scaling

- How DevOps can be applied in different organizations as they grow, mature, or even shrink

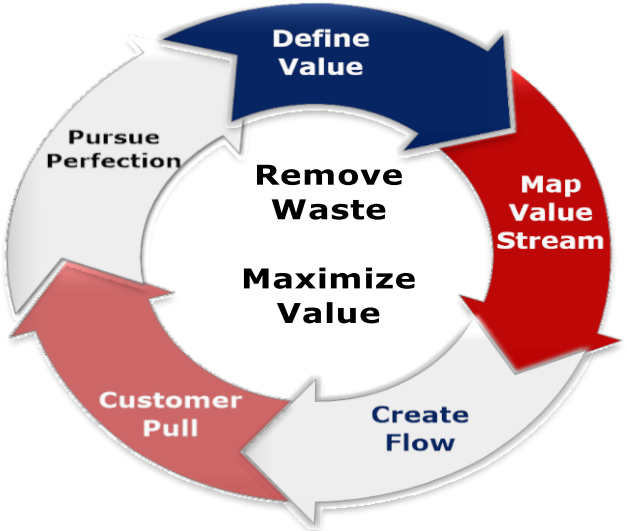
- Maximization of customer value

- **Take Customer Advice (and Credit Them for It)**
- **Be There When Customers Need You**
- **Help Customers Do Something They Love**
- **Give Customers Something Your Competitors Aren’t**
- **Be More Convenient than Anyone Else**
- **Solve a Problem for Your Customer**
- **Make Quality a Priority**

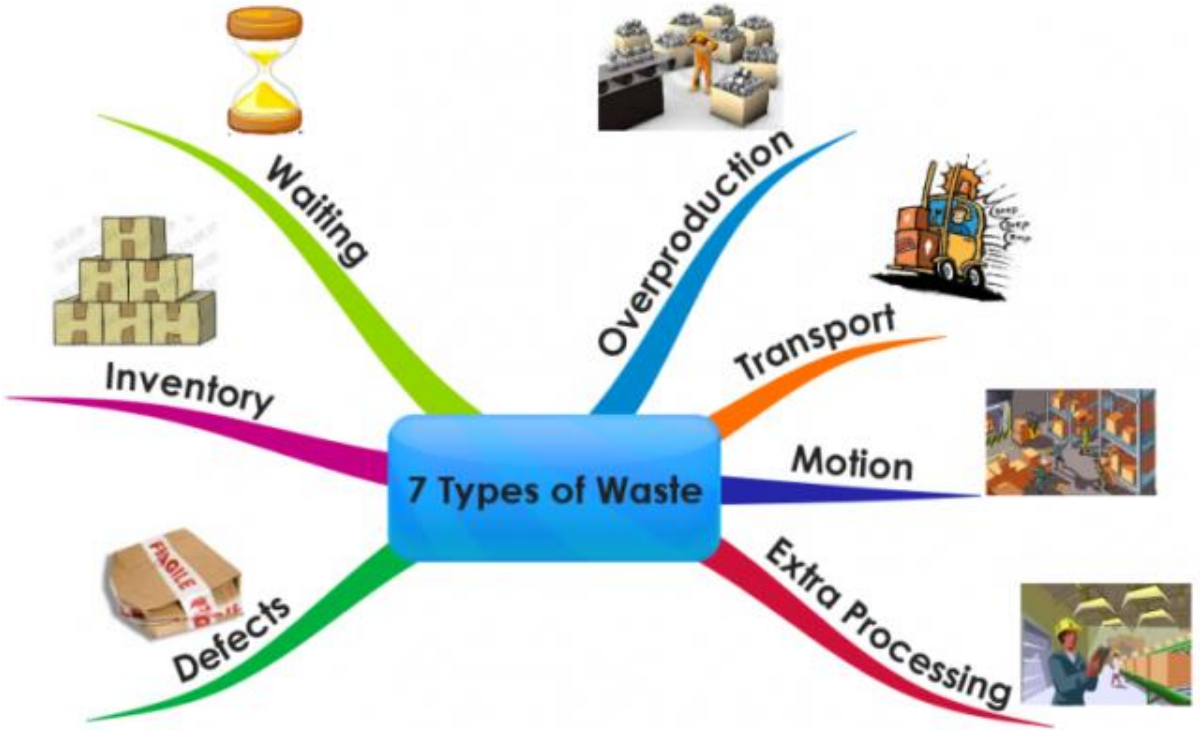


Five Principles of Lean Thinking

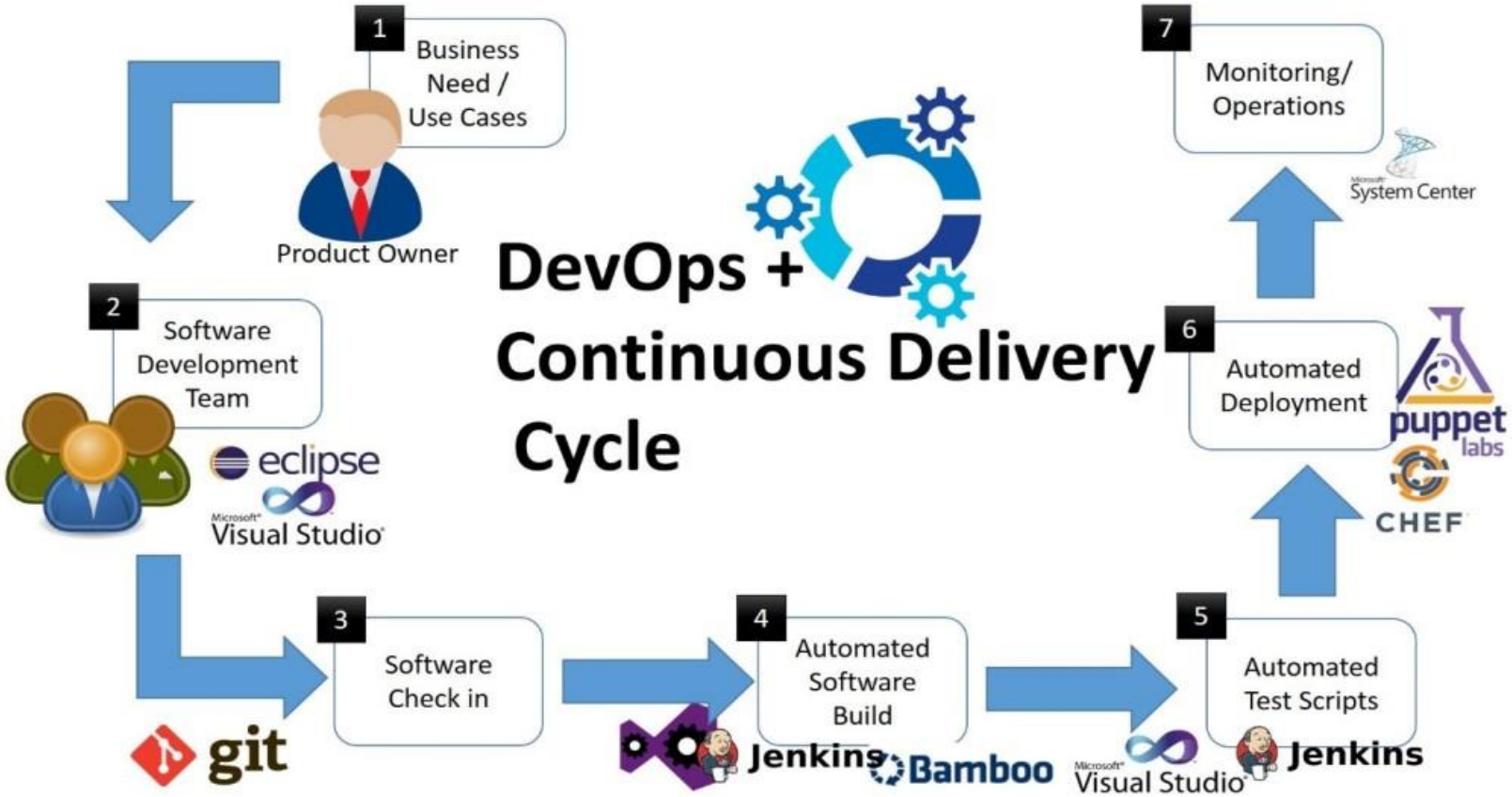
- Define value precisely from the perspective of the end customer
- Identify the entire value stream for each service, product or product family and eliminate waste
- Make the remaining value-creating steps flow
- As flow is introduced, let the customer pull what the customer wants when the customer wants it
- Pursue perfection



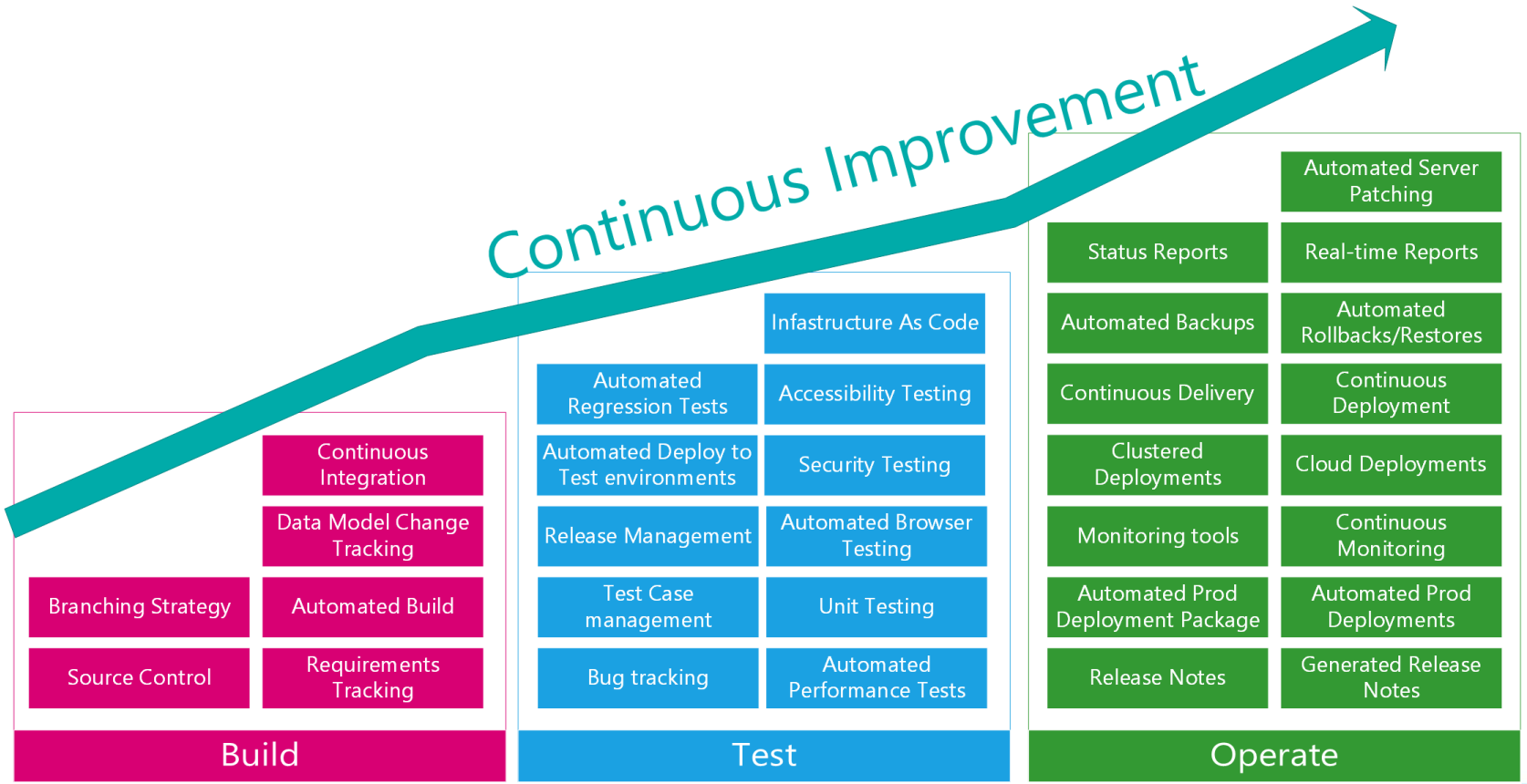
The voice of the customer (VOC) process captures and analyzes customer requirements and feedback to understand what the customer wants.


7 Types of Waste
Waiting
Overproduction
Transport
Inventory
Motion
Defects
Extra Processing

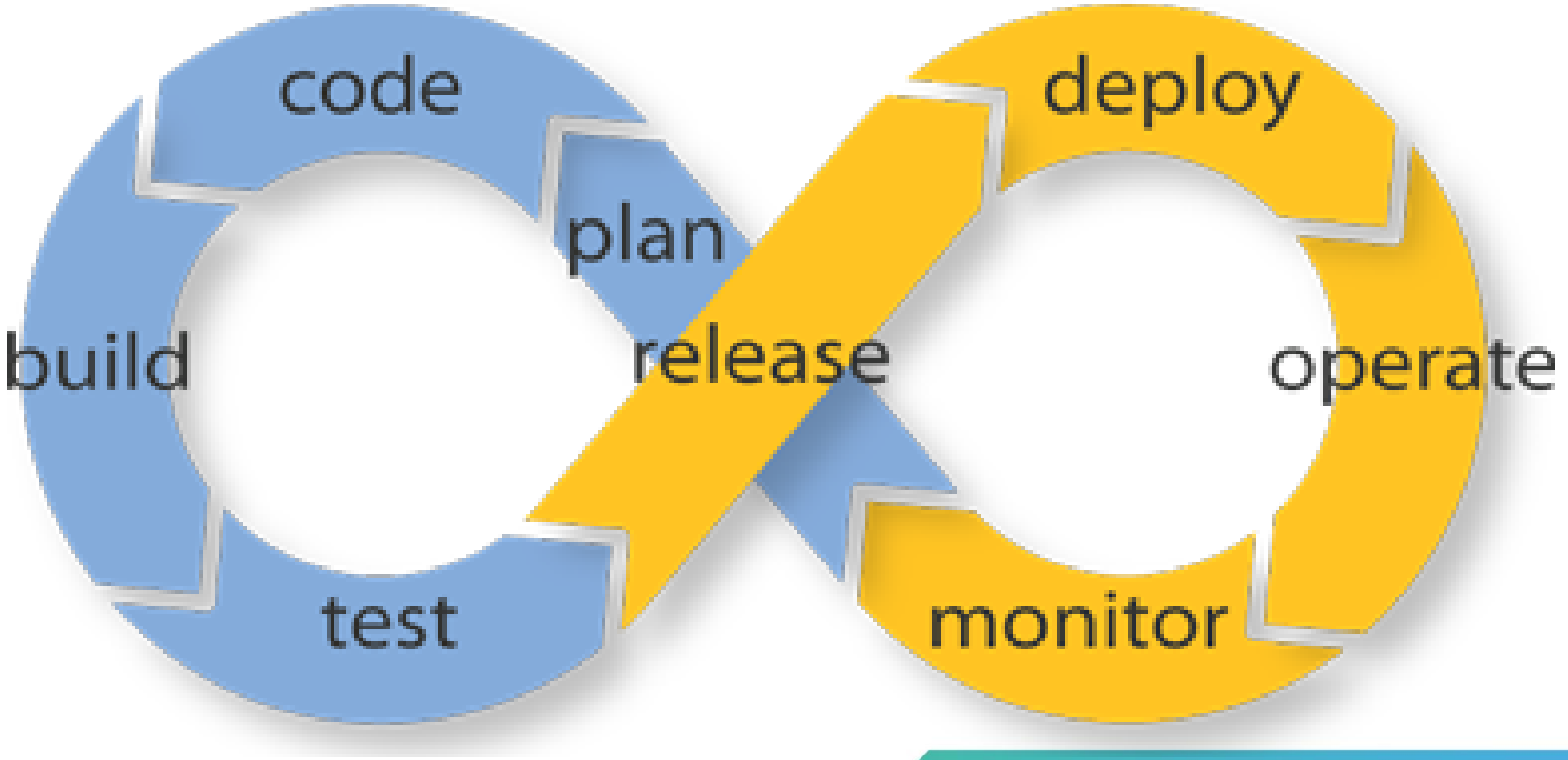
DevOps Workflow



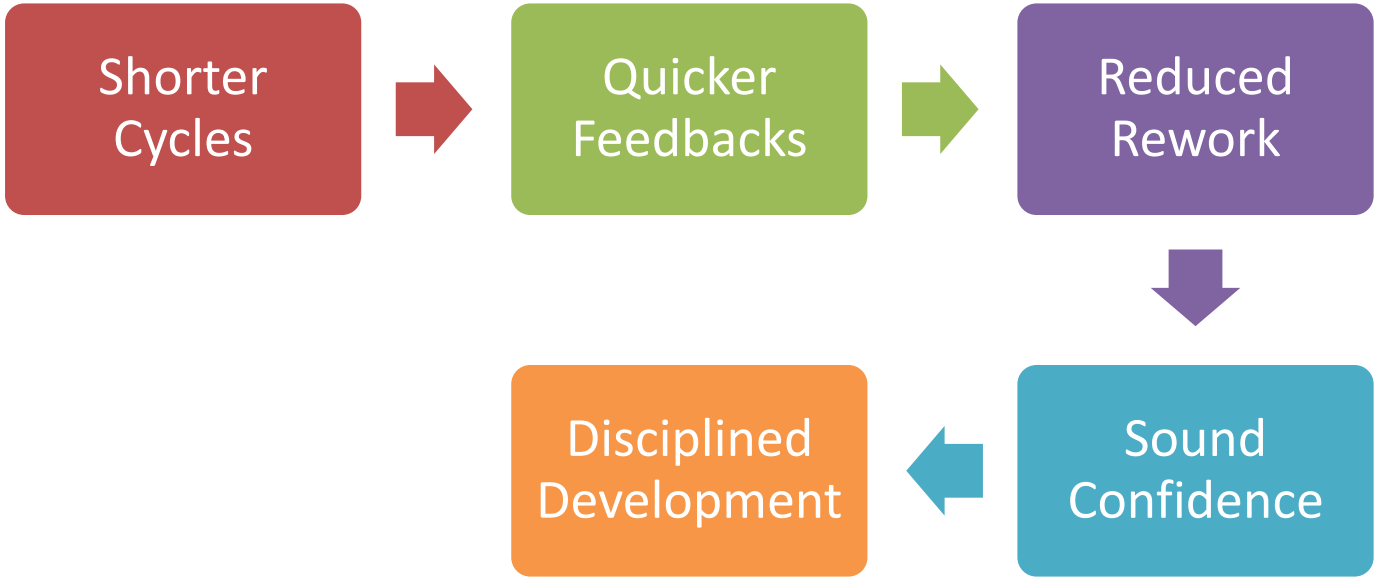
Continuous Improvement



Continuous Integration & Continuous Delivery



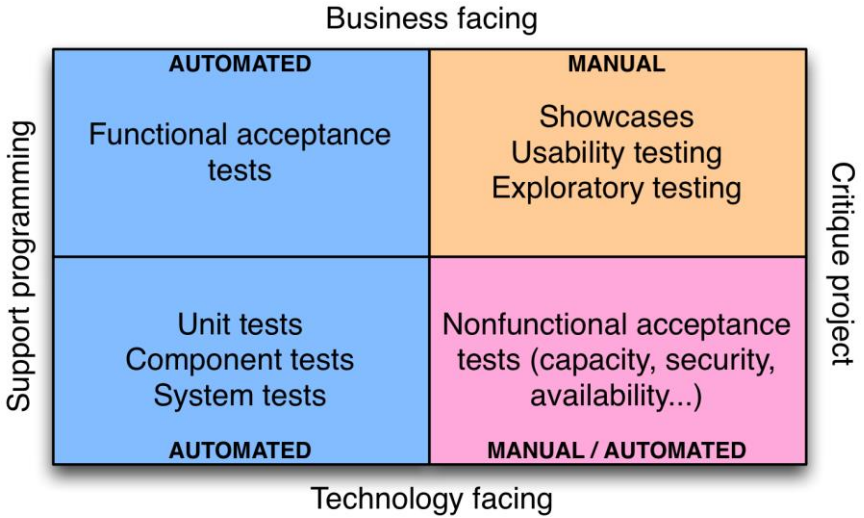
Continuous Development



Continuous Testing

- Manual regression testing takes a long time and is relatively expensive to perform, creating a bottleneck that prevents us releasing software more frequently, and getting feedback to developers weeks (and sometimes months) after they wrote the code being tested.
- Manual tests and inspections are not very reliable, since people are notoriously poor at performing repetitive tasks such as regression testing manually, and it is extremely hard to predict the impact of a set of changes on a complex software system through inspection.
- When systems are evolving over time, as is the case in modern software products and services, we have to spend considerable effort updating test documentation to keep it up-to-date.

The goal is to run many different types of tests—
both manual and automated—
continually throughout the delivery process.

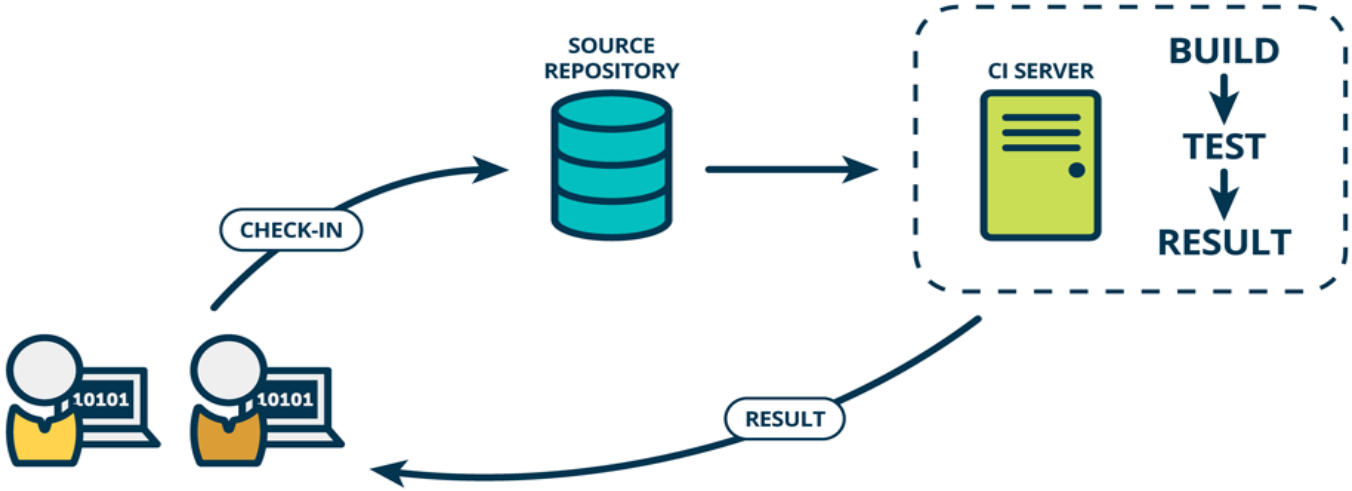


Continuous Integration

- Development practice that requires developers to integrate code into a shared repository several times a day.
- Each check-in is then verified by an automated build, allowing teams to detect problems early.
- Continuous Integration is cheap. Not integrating continuously is expensive.
- Continuous Integration brings multiple benefits to your organization:
    - Say goodbye to long and tense integrations
    - Increase visibility enabling greater communication
    - Catch issues early and nip them in the bud
    - Spend less time debugging and more time adding features
    - Stop waiting to find out if your code’s going to work

“Keep software always in a working state”

“Continuous Integration doesn’t get rid of bugs, but it does make them dramatically easier to find and remove.” - Martin Fowler, Chief Scientist, ThoughtWorks

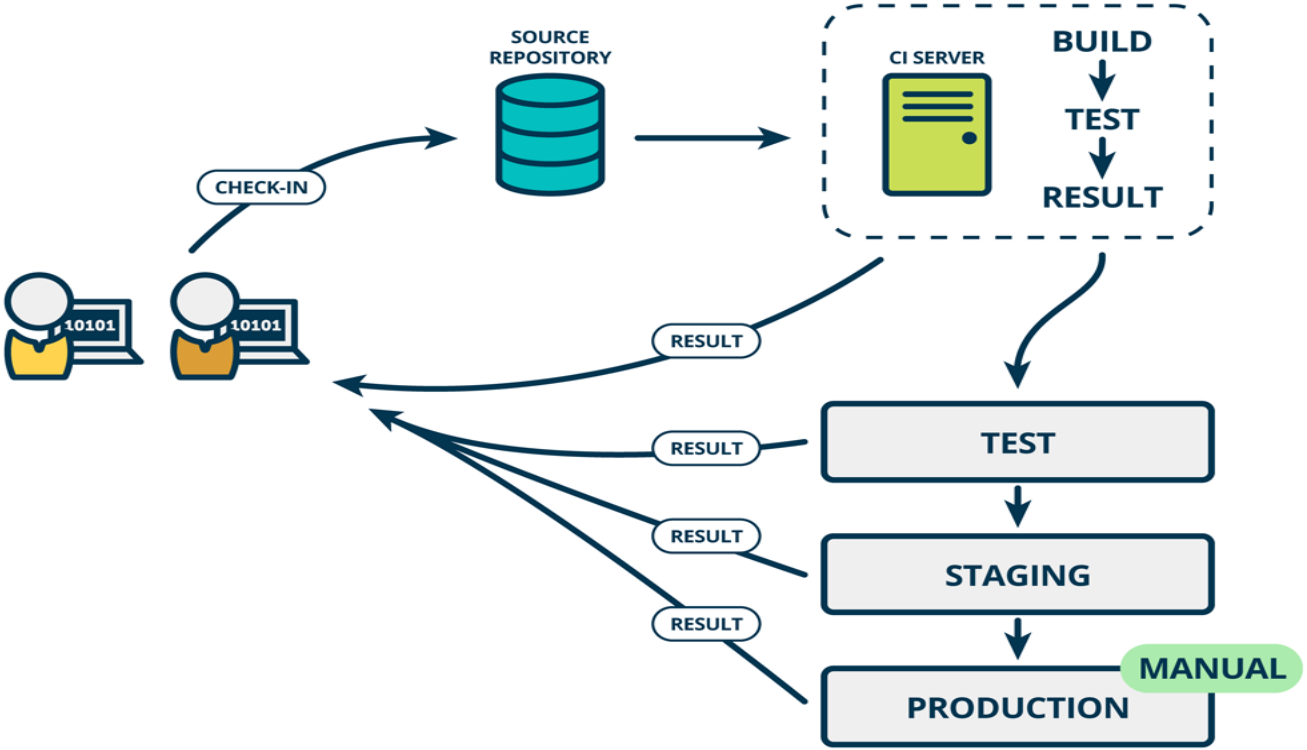


Continuous Delivery

- Process of releasing new software frequently through the use of automated testing and continuous integration.
- Automated implementation of the application build, deploy, test and release processes.
- To establish end to end deployment pipeline
- Through reliable, low-risk releases,
- Continuous Delivery makes it possible to
- continuously adapt software in line with user feedback, shifts
- in the market and changes to business strategy.
- Test, support, development and operations must work together as one delivery team to automate and streamline the build-test-release process.

Continuous Delivery doesn't mean every change is deployed to production ASAP. It means every change is proven to be deployable at any time

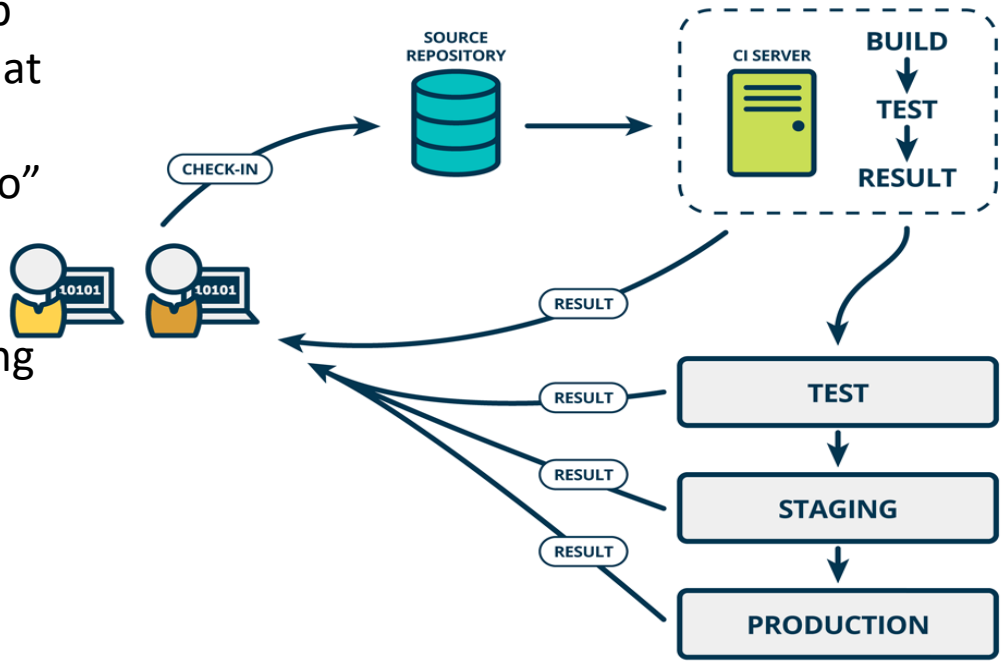
Continuous Delivery



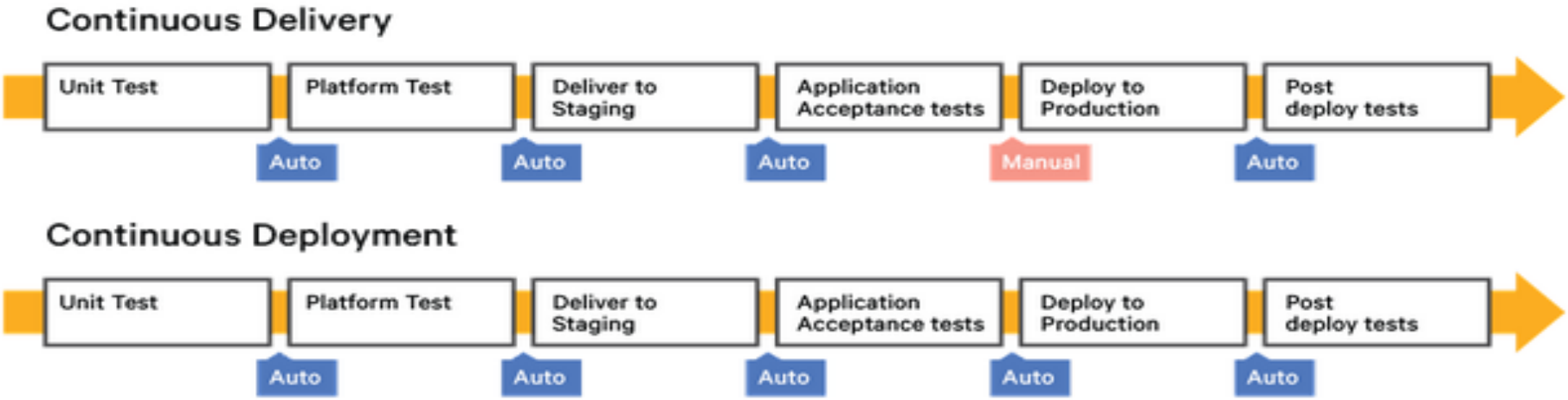
Continuous Deployment

- Continuous deployment is the next step of continuous delivery: Every change that passes the automated tests is deployed to production automatically. “Push to go”

- There are business cases in which IT must wait for a feature to go live, making continuous deployment impractical.
- While continuous deployment may not be right for every company, continuous delivery is an absolute requirement of DevOps practices.



Continuous Delivery Vs Continuous Deployment



Continuous Integration

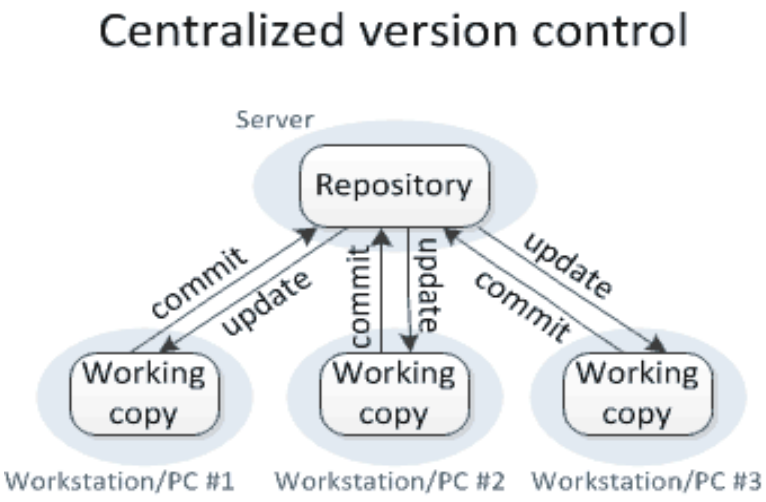
- Check symptoms
- How they could have been spotted earlier
- Root cause analysis
- Fix the cause
- Use risk management as required

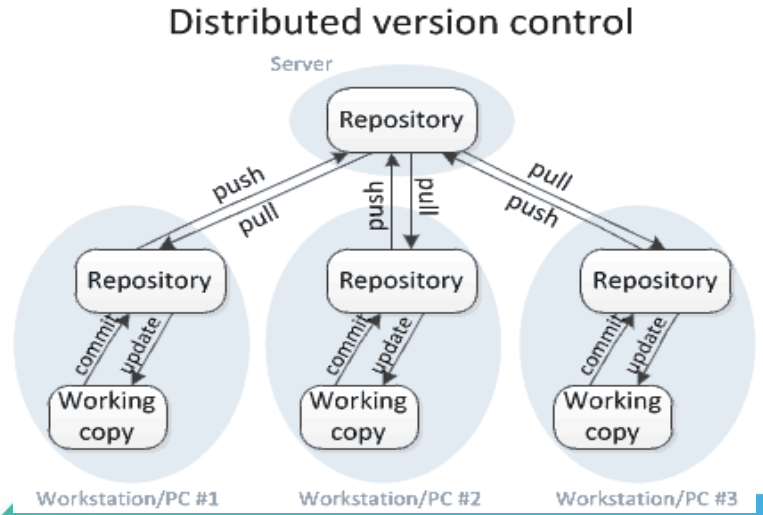
Continuous Delivery

- Check In code regularly (ideally twice a day)
- Create a Comprehensive Automated Test Suite
- Keep the Build and Test Process Short
- Managing Your Development Workspace

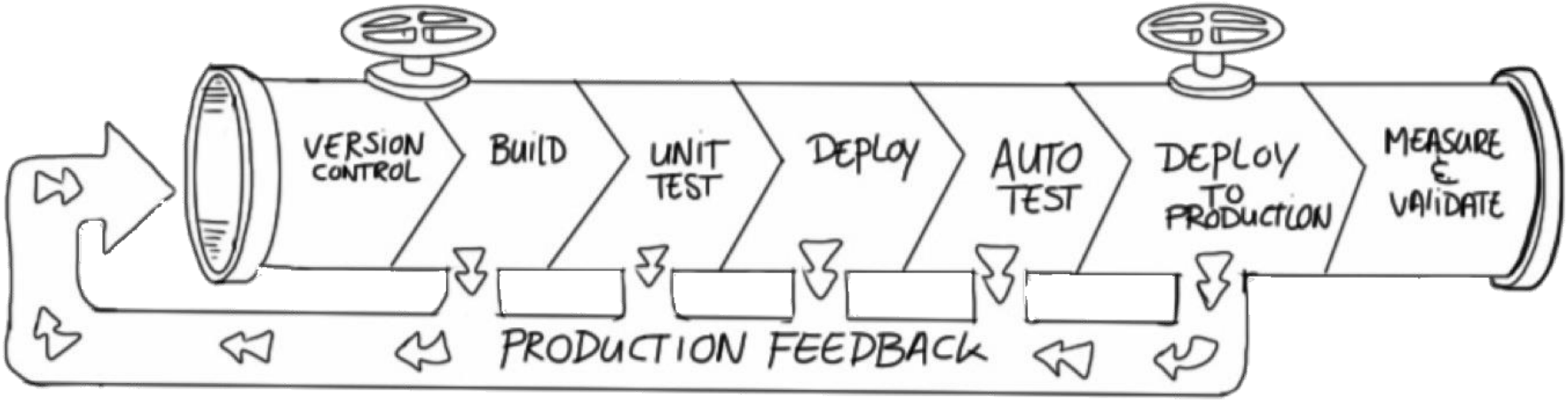
CI With Distributed Version Control System

- Distributed version control system
- **Automated build** – will combine code from various sources
- **Automated testing** – ensure that the build is not broken; if it is, whoever broke it must fix or revert it so that others can continue working
- **Automated release and deployment** (but test locally before committing code for release)
- **Electronic communication tools** (IM and VoIP) are essential for constant communication





Deployment Pipeline

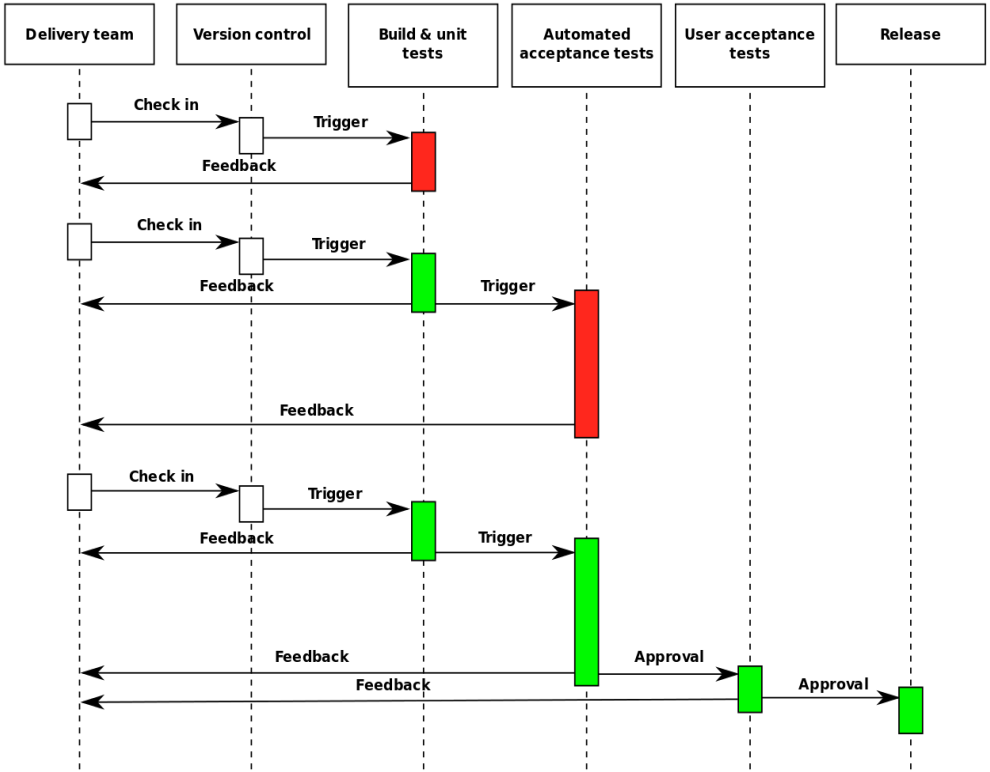


Purpose of Deployment Pipeline

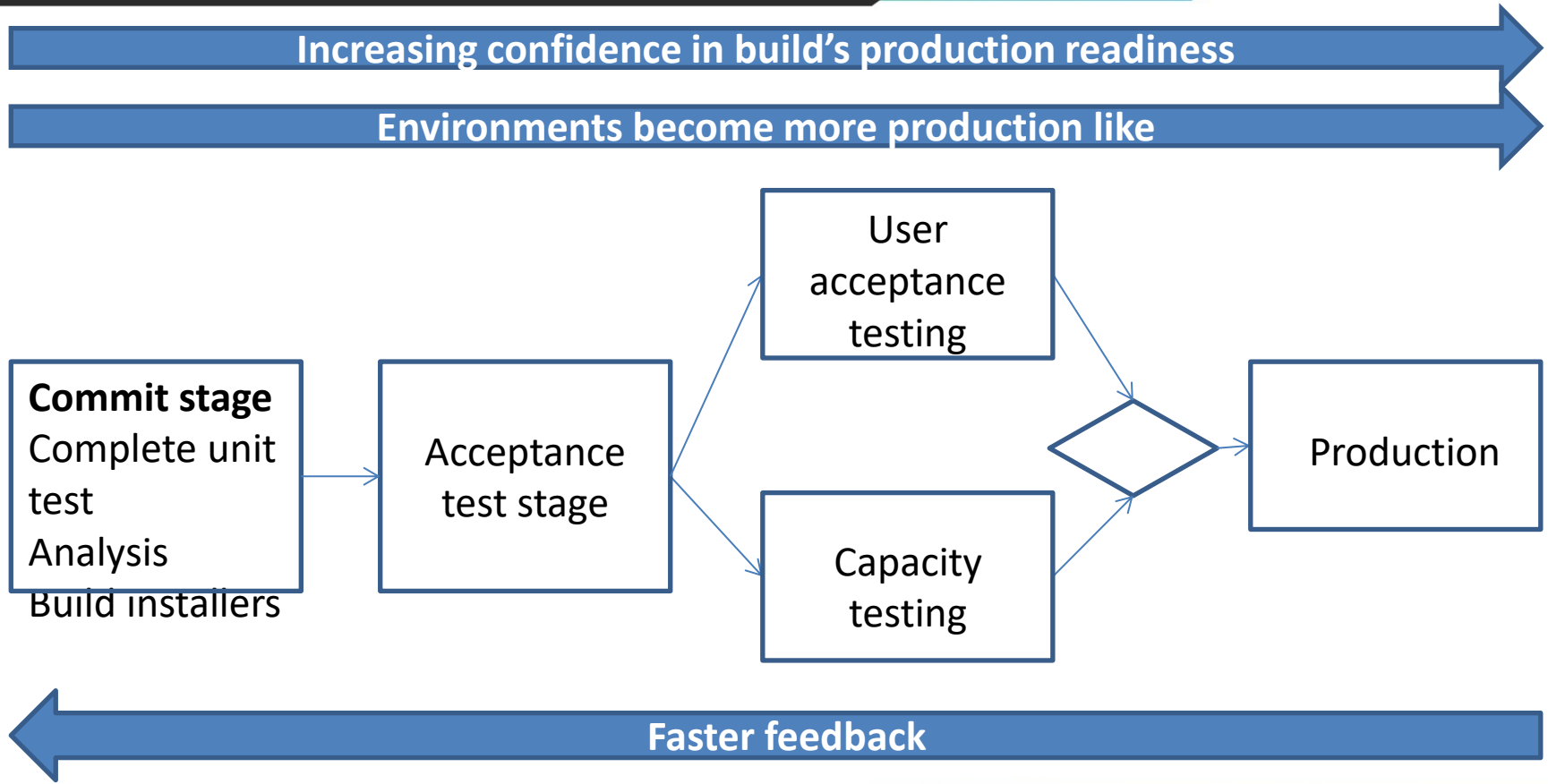
- **Visibility**:
- All aspects of the delivery system are visible to all team members promoting collaboration.

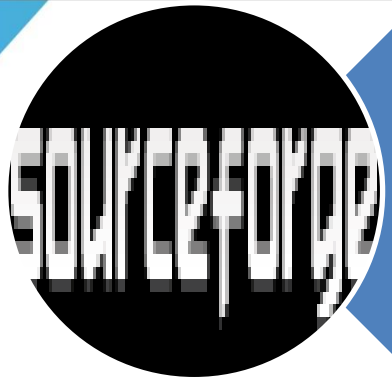
- **Feedback:**
- Team members learn of problems as soon as they occur so that issues are fixed as soon as possible.

- **Continually Deploy**:
- Through a fully automated process, you can deploy and release any version of the software to any environment.



DevOps Deployment Pipeline





Collectl gather and store statistics about the system on which they run and are much more flexible than other tools. It collects information every couple of seconds and then ships to allow us to run reports and send alerts. Allow users to measure the values of multiple system metrics unlike other log collection tools which measure specific system parameters.

Most common source code management tool available today.

GitHub also has plug-ins that can connect with Jenkins to facilitate integration and deployment.

DevOps Tools

Great ecosystem of plug-ins and add-ons to build code, create Docker containers, run tons of tests, and push to staging/production.

It's a great tool, but there are some issues regarding scaling and performance. Explore other cool solutions such as Travis and CircleCI,

It eases configuration management, control issues, and scaling by allowing containers to be moved from one place to another.

Ansible is a configuration management tool that is similar to Puppet and Chef. They consists of richer feature set and much more complex. Best suit to push changes and re-configure newly-deployed machines instead upgrades.

Nagios

itoring solution that is highly effective because of the large community of contributors who create plug-ins for the tool.

A watchdog who monitors the services and in case of failures, starts the services and send the alerts.

Logz.IO

Collects logs from all services, applications, networks, tools, servers, and more in an environment into a single, centralized location for analytical purposes

CONSUL

Provides internal DNS names for services. Enabling you to access service names instead of specific machines.

DevOps Toolset

Cloud Computing Platforms - Fundamentally accelerates the access to compute resources to facilitate faster software development and improved application scalability.

Amazon Web Services
Pivotal Cloud Foundry
Google Cloud Platform
Microsoft Azure
Rack space
OpenStack

Virtualization Platforms – Uses software to abstract hardware and make it possible to run multiple applications at the same for more efficient use of compute resources.

VMware
KVM
Xen
VirtualBox
Vagrant
QEMU

Linux OS Installation – Performs unattended operating system installation and configuration.

Kickstart
Cobbler
Fai

Configuration Management and Infrastructure as Code - Allows teams to declare and automate system configuration for better application manageability and IT productivity.

Puppet
Chef
Ansible
HashiCorp Terraform
AWS CloudFormation

Test and Build Systems – Facilitates continuous integration and continuous delivery by automating repeatable jobs as part of the software development process.

Solano Labs
Jenkins
Gradle

Monitoring, Alerting, and Trending - Allows digital teams to measure the impact of frequent release cycles across complex, dynamic IT environments.

New Relic
PagerDuty
VictorOps
Sensu
xMatters
BlueMedora

DevOps Toolset

Containerization – **Wraps software in a complete filesystem that contains everything it needs to run to enable automated deployments and more reliable software.** ***Docker*** ***rkt*** ***LXC***	**Process Supervisors –** **Eases the management of operating system services.** ***Monit*** ***runit*** ***Supervisor*** ***God*** ***Blue Pill*** ***Upstart*** ***systemd***	**Security –** **Helps you keep your applications safe from threats even as you release faster and adopt new technologies.** ***Snorby Threat Stack*** ***Snort*** ***Veracode*** ***Sqreen*** ***Sonatype*** ***Evident.io***	**Orchestration - Software systems that facilitate the automated management, scaling, discovery, and/or deployment of container-based applications or workloads.** ***Kubernetes*** ***Apache Mesos, DC/OS*** ***HashiCorp Atlas*** ***Docker Swarm*** ***AWS Elastic Container Service***
Application Deployment - Automates the process of making a new version of an application available on one or more web servers. *Capistrano*	**Logging –** Provides an intuitive interface for analyzing machine-generated data for easier application-issue detection and resolution. *PaperTrail* *Logstash* *Loggly* *Logentries* *Splunk* *SumoLogic*	**Collaboration and Ticketing** - Allows diverse DevOps teams to break down organizational silos through better, more transparent communication. *Slack* *HipChat* *JIRA* *ServiceNow*	

Source Code Management

Version Control/ SCM

git

Jenkins

Continuous Testing

Jenkins

Continuous Deployment

Continuous Monitoring

Nagios

puppet

docker

ANSIBLE

Configuration Management

Containerization/ Virtualization

feedback

Continuous Integration

Version Control/ SCM

Jenkins

Continuous Testing

Jenkins

Continuous Deployment

Continuous Monitoring

puppet

docker

ANSIBLE

Configuration Management

Containerization/ Virtualization

feedback

Continuous testing

Version Control/ SCM

Jenkins

Continuous Testing

Jenkins

Continuous Deployment

Continuous Monitoring

Configuration Management

Containerization/ Virtualization

feedback

Continuous Deployment: Configuration Management

Version Control/ SCM

Jenkins

Continuous Testing

Jenkins

Continuous Deployment

Continuous Monitoring

puppet

ANSIBLE

Configuration Management

docker

Containerization/ Virtualization

feedback

Continuous Deployment: Containerization

git

Version Control/ SCM

Jenkins

Continuous Testing

Jenkins

Continuous Deployment

Nagios

Continuous Monitoring

puppet

ANSIBLE

Configuration Management

docker

Containerization/ Virtualization

feedback

Continuous Monitoring

git

Version Control/ SCM

Jenkins

Continuous Testing

Jenkins

Continuous Deployment

Nagios

Continuous Monitoring

puppet

ANSIBLE

Configuration Management

docker

Containerization/ Virtualization

feedback

Automation Enablers

- Continuous integration and continuous delivery
- Configuration management
- Build, testing and deployment processes
- On-demand creation of development, test, staging and production environments
- Treating infrastructure as code
- Monitoring and dashboards
- Experimentation
- Ongoing operations and support

- Automation supports
- Faster lead times
- More frequent releases
- Less turbulent releases
- Fewer errors
- Higher quality
- Improved security and risk mitigation
- Faster recovery
- Business and customer satisfaction

Automation gives rote tasks to computers and allows people to

- Weigh evidence
- Solve problems
- Make decisions based on feedback
- Use their skills, experience and judgment


Cognixia
A Collabera LEARNING SOLUTIONS COMPANY
THANK YOU
45%