Course Introduction

In this course, you will learn the key principles of DevOps and the value that it can bring to teams that adopt this culture. Key topics that we will discuss in this course:

* What is DevOps
* DevOps Principles
* Various Stages in DevOps Lifecycle
* Metrics to measure the effectiveness of DevOps

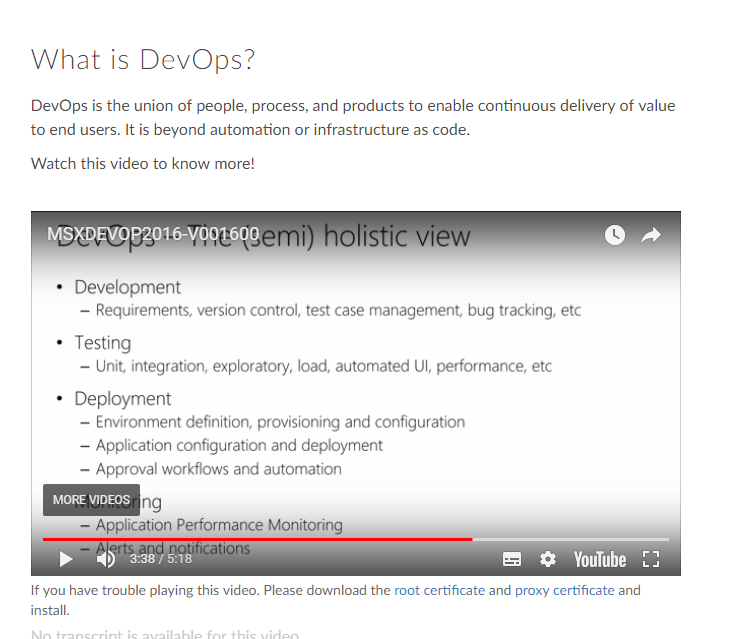
##### What is DevOps?

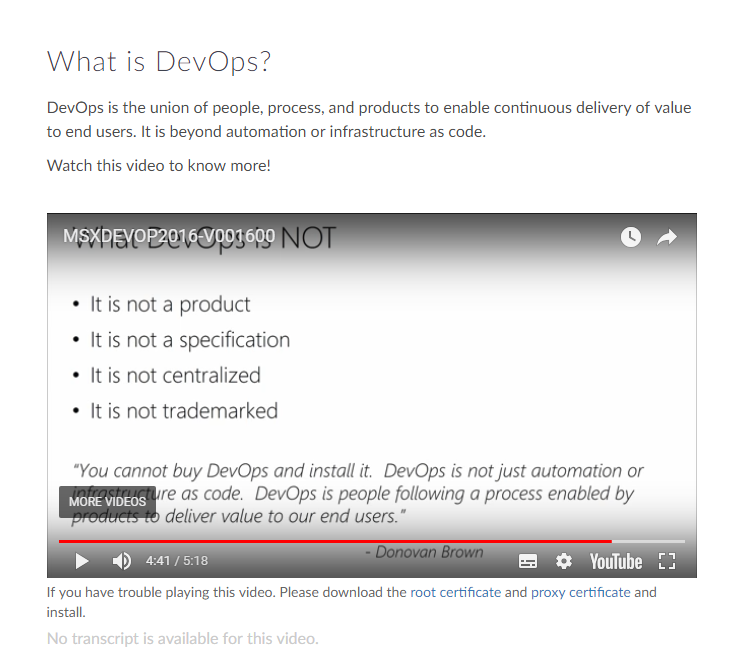
DevOps is the union of people, process, and products to enable continuous delivery of value to end users. It is beyond automation or infrastructure as code.

Watch this video to know more!









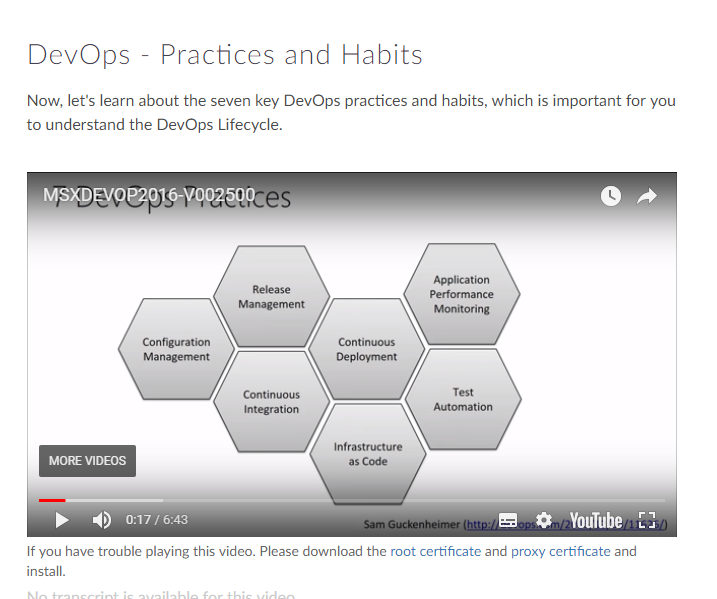
Value of DevOps

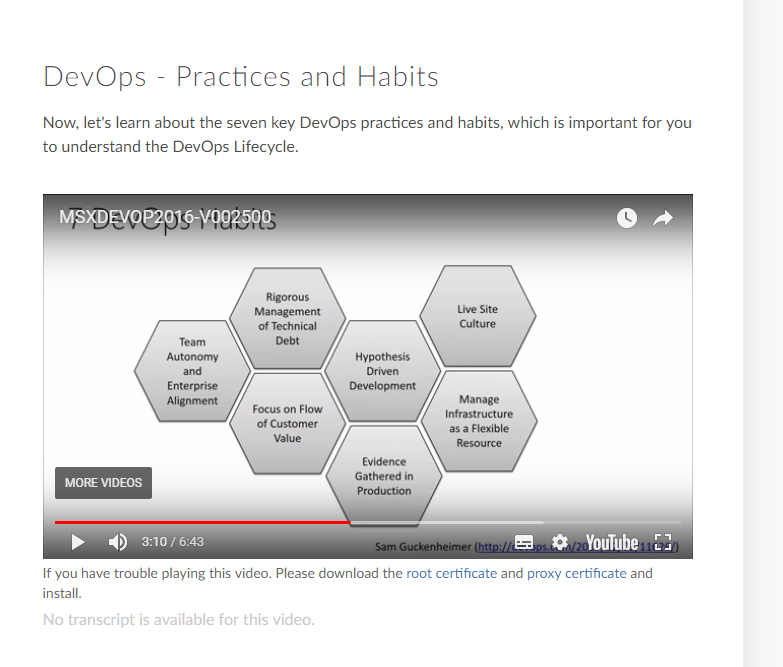
DevOps culture nurtures better communication and collaboration between teams, thus breaking the silos formed among teams.

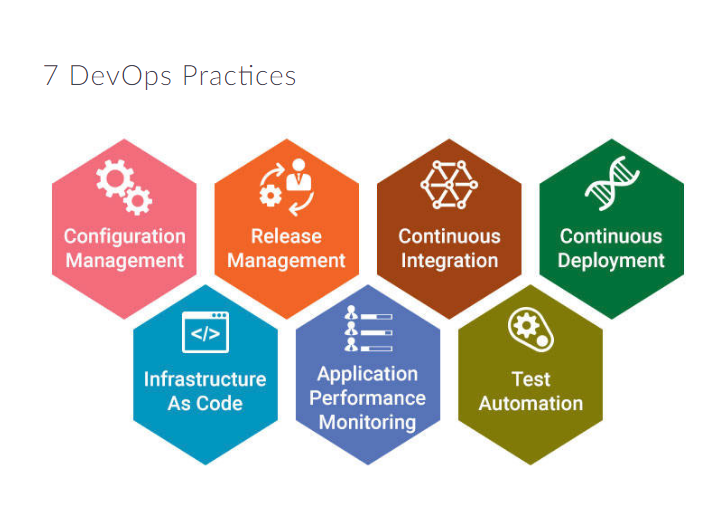
DevOps aims at:

* Developing high quality software
* Deploying in frequent cycles
* Reducing time to move from idea to implementation



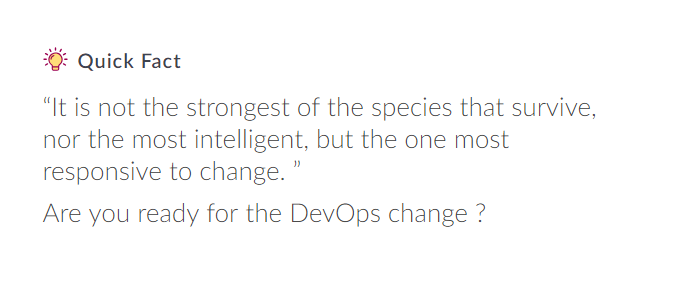












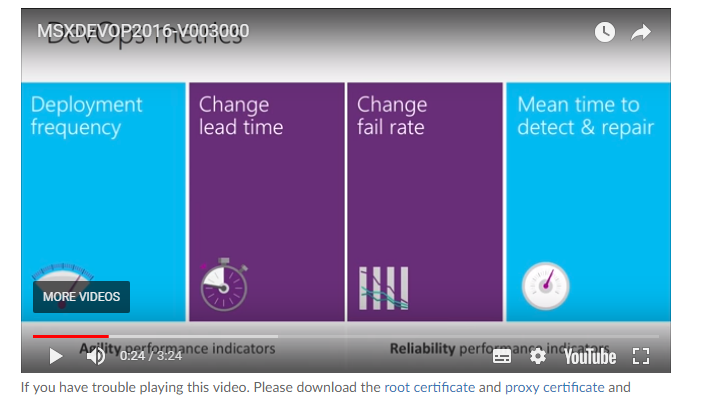
##### Module Outline

In this module, you will understand, KPIs and metrics that are typically analyzed in DevOps culture.

Also, you will learn about the agility performance and reliability performance indicators.

##### DevOps - Metrics

This video will introduce you to the DevOps Metrics Agility performance indicators and Reliability performance indicators that are typically collected in DevOps.



Metric - Deployment Frequency



***Deployment Frequency*** is a direct or indirect measure of

* Response Time
* Team Cohesiveness
* Developer Capabilities
* Effectiveness of Development tool
* Efficiency of the team

Ideally Deployment Frequency should show an upward trend or remain stable week on week.

##### Metric - Change Lead Time



**Change Lead Time** measures complexity of the code and the efficiency of the development systems.If the change lead time is too long, it may be an indication that the development/deployment process is inefficient in certain stages, or that it includes performance bottlenecks.

##### Quick Fact

“In Ops you can never exceed expectations, because the expectation is 100% up-time”

##### Metric - Reliability Performance



**Change Failure Rate** is a good measure of the value of deployments in DevOps.

A low failure rate is a good indication of a healthy DevOps team

**Mean Time To Recover (MTTR)** is a measure of failure rate. This can be seriously affected by code complexity, number of additional features being added and environmental changes.

##### Source Control in DevOps

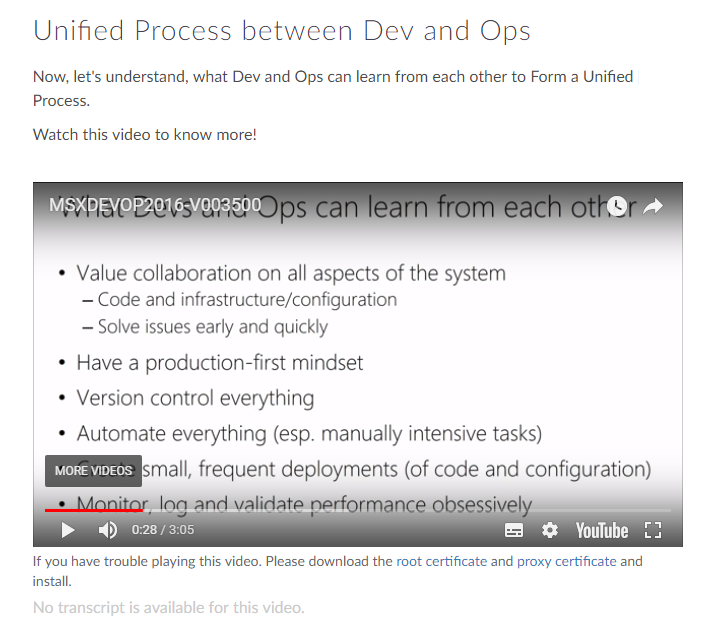
In DevOps, we treat infrastructure as code. Hence version or source control is an integral part of the culture.

**The ultimate goal is to treat the infrastructure code in the same lines on the application code**.

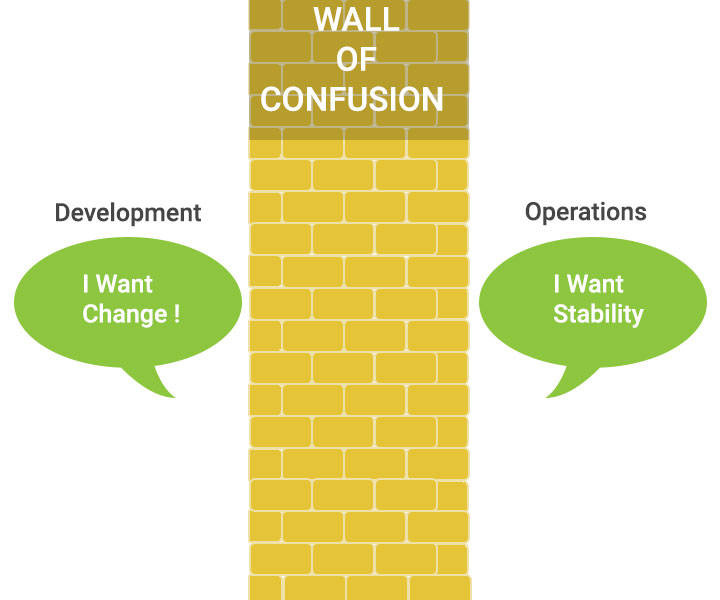
Watch this video to learn more about importance of version control in DevOps.

##### Unify Dev and Ops

In this module, you will learn how to deliver incremental value and the importance of communication and collaboration in Unifying Dev and Ops.



Collaboration and Communication



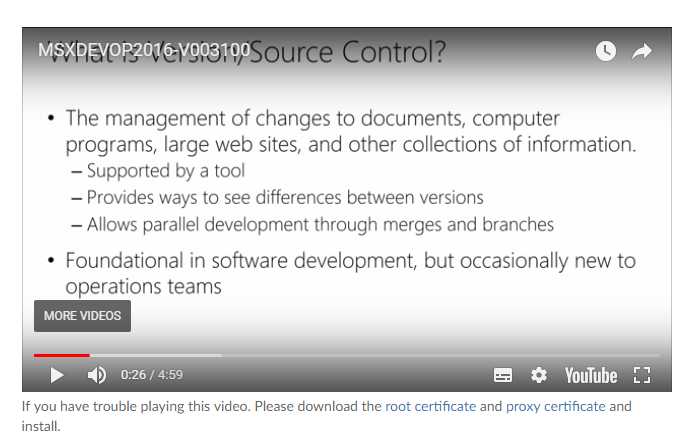
Collaboration and Communication are essential for the success of DevOps team. Fostering a culture of Collaboration and Communication helps:

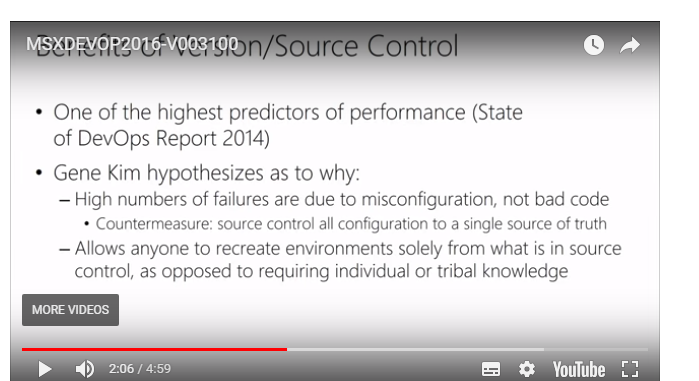
* Break barriers among team members
* Create visibility

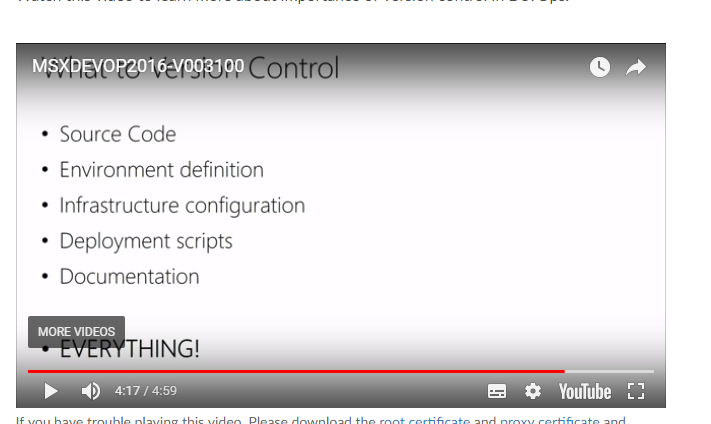
##### Quick Fact

“Devs are from Venus, Ops are from Mars”

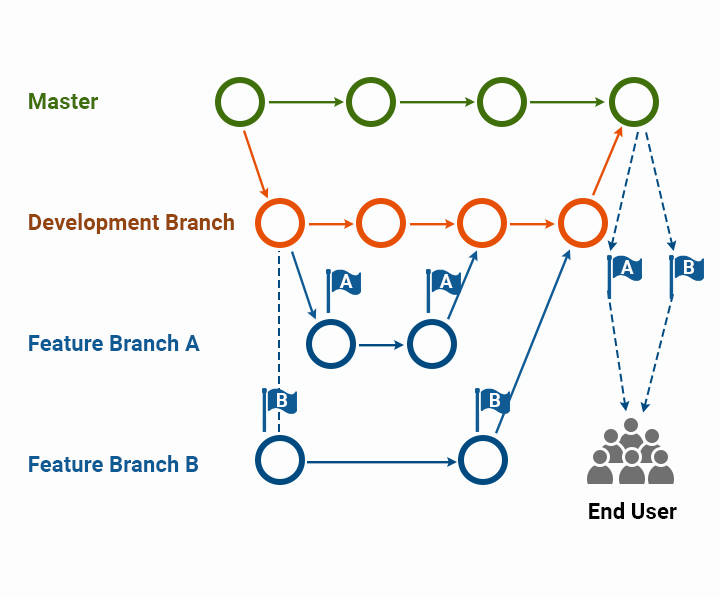
Time to break the Wall of Confusion!







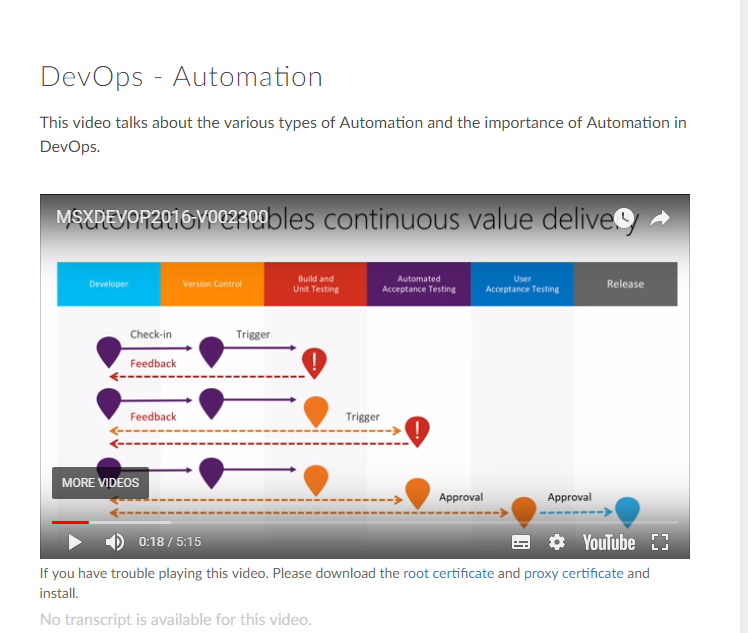
##### Branching Illustrated

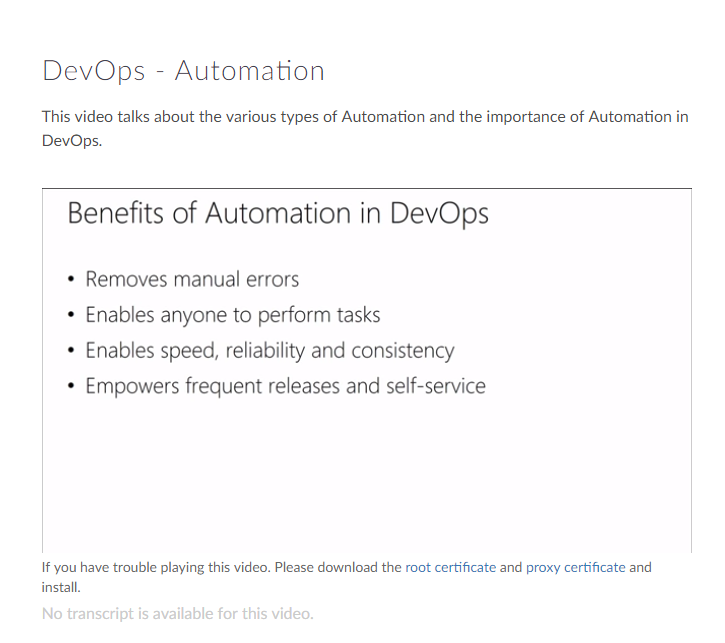


Branching Strategy

Ideally your branching strategy should accomplish the following three goals:

* Minimize conflicts when merging code
* Track code changes in the development pipeline
* Add a degree of separation between code at different stages of development







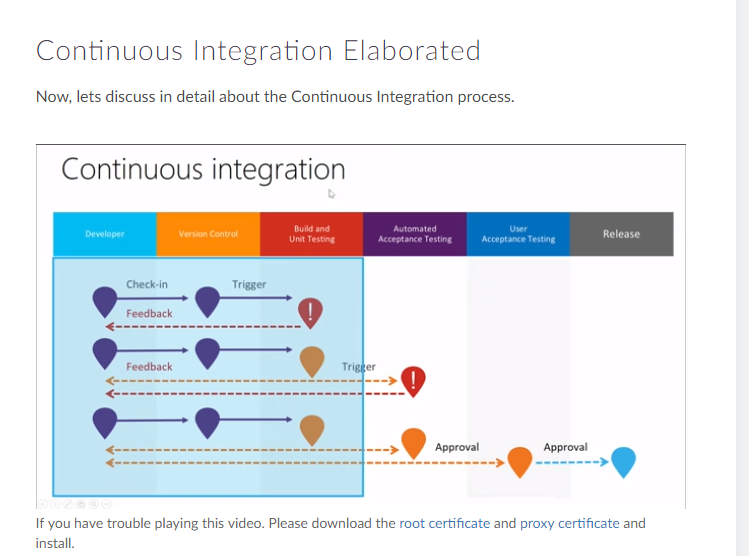
Automation Recommendation

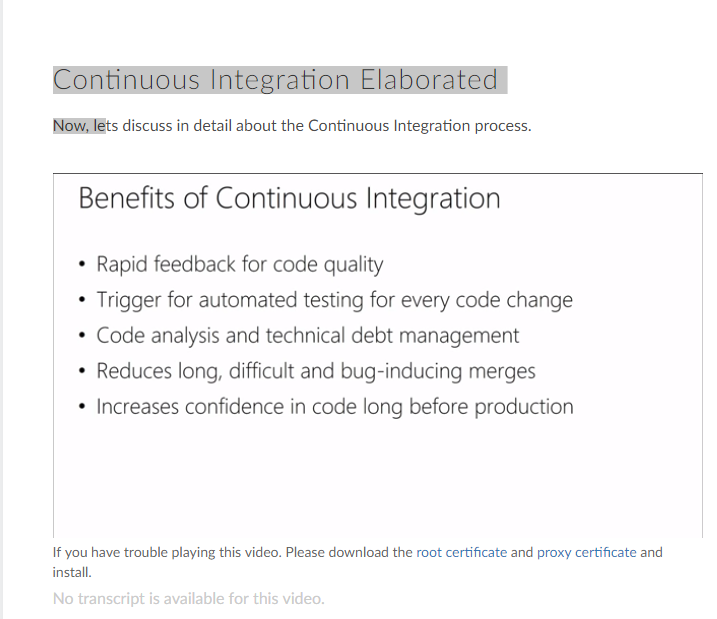
* It is recommended not to implement full automation. i.e. automate just what is necessary.
* Automation is not an independent discipline in itself.
* Automation is done to obtain fast feedback.

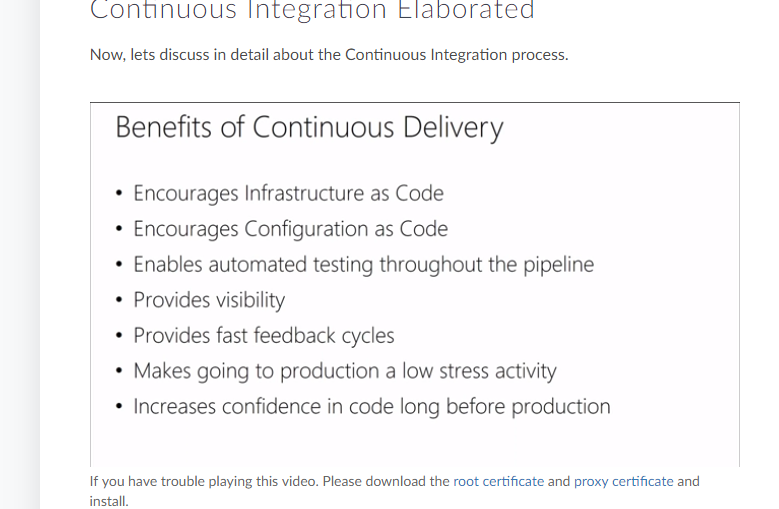
##### Continuous Integration

**Continuous Integration** forces the developers to integrate their code at an early stage which prevents integration issues from happening towards the end of the lifecycle.









Benefits of Continuous Integration

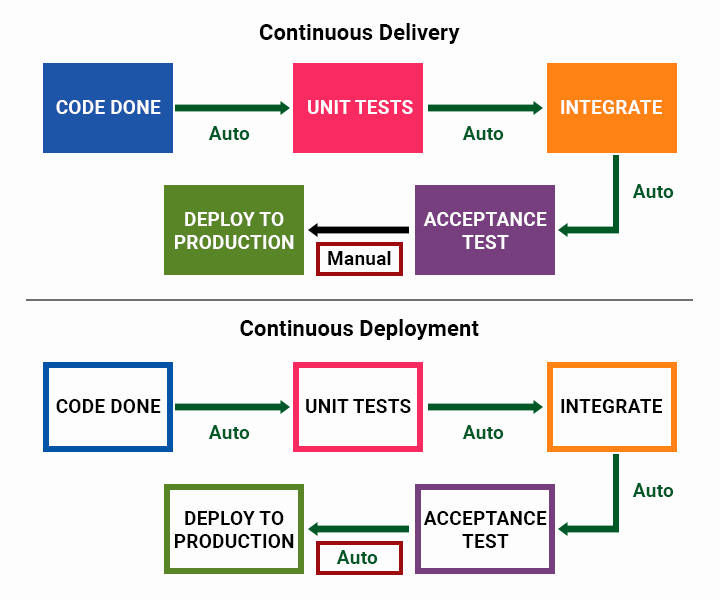
* ***Improve Productivity*** of Developers
* ***Address bugs and issues*** at a very early stage
* ***Faster Delivery*** of software updates

##### Continuous Deployment

**Continuous Deployment** is a Continuous Delivery pipeline with no manual gates between initial code commit / check-in and production.

Some organizations have matured to an extent to deliver multiple updates on the same day.

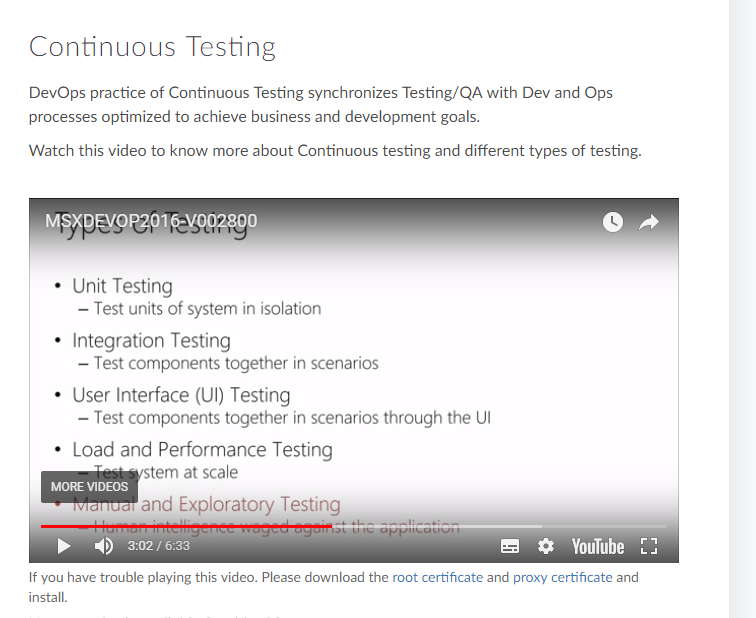
##### Continuous Delivery and Continuous Deployment



##### Continuous Testing

DevOps practice of Continuous Testing synchronizes Testing/QA with Dev and Ops processes optimized to achieve business and development goals.

Watch this video to know more about Continuous testing and different types of testing.



Key Elements of Continuous Testing

* Traceability
* Comprehensive Analysis
* Policy Analysis
* Risk Assessment

Types of Testing

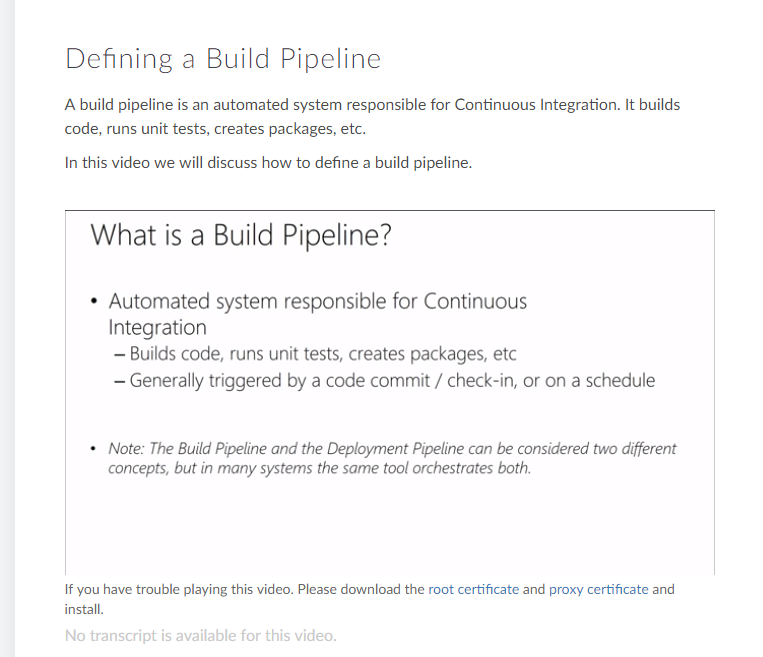
Different Types of Testing that can be automated

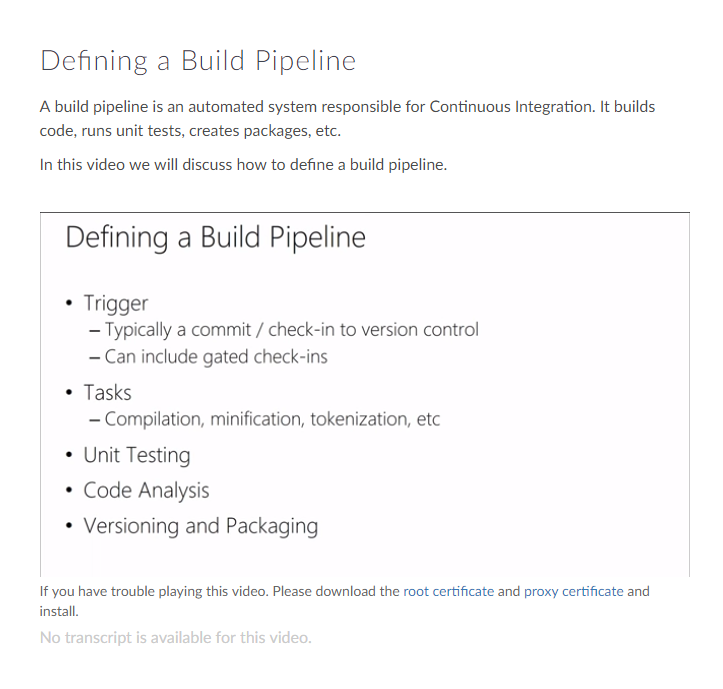
* Unit Testing
* Integration Testing
* Functional Testing
* Smoke Testing
* Regression Testing

##### Defining a Build Pipeline

A build pipeline is an automated system responsible for Continuous Integration. It builds code, runs unit tests, creates packages, etc.

In this video we will discuss how to define a build pipeline.



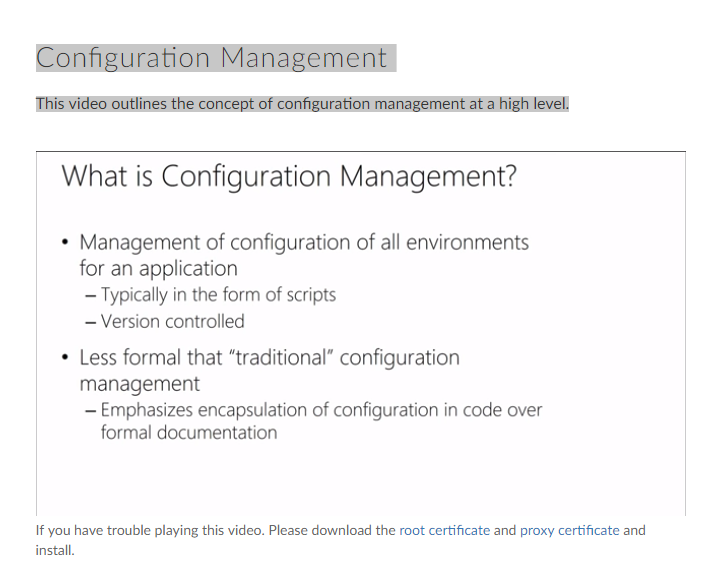


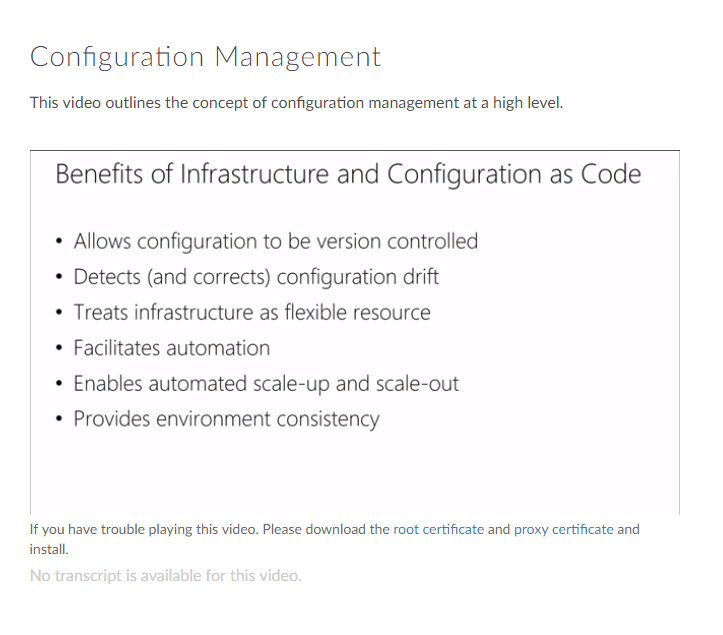
##### Infrastructure as Code

When Infrastructure is treated as code, the knowledge of deployment, configuration management and provisioning are not confined to just System Admins. They can be a developer's tasks.

##### Configuration Management

This video outlines the concept of configuration management at a high level.





Configuration Drift

It is the process where servers running in an infrastructure become different over a period of time due to human induced changes.

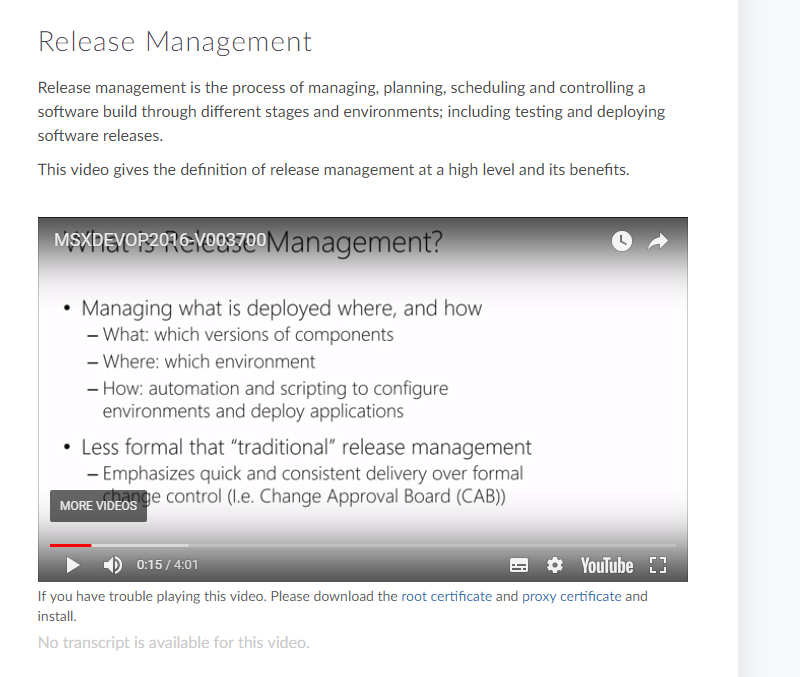
Ways to counter Configuration Drift:

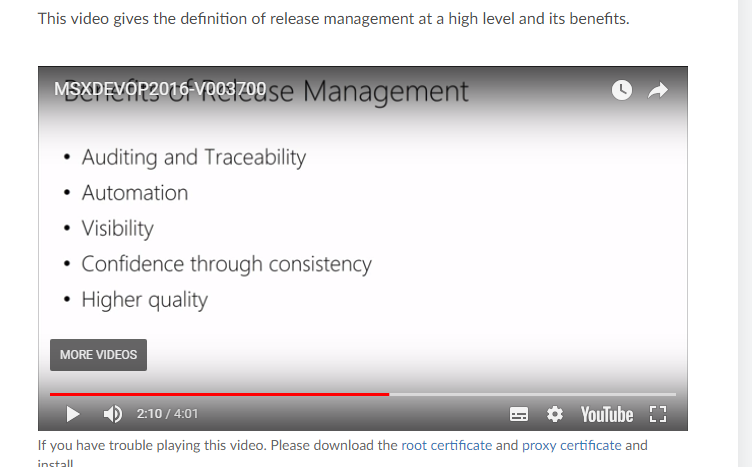
* Rebuilding machine instances frequently
* Make use of automated configuration tools and run them at regular intervals to keep the machines in sync

##### Release Management

Release management is the process of managing, planning, scheduling and controlling a software build through different stages and environments; including testing and deploying software releases.

This video gives the definition of release management at a high level and its benefits.

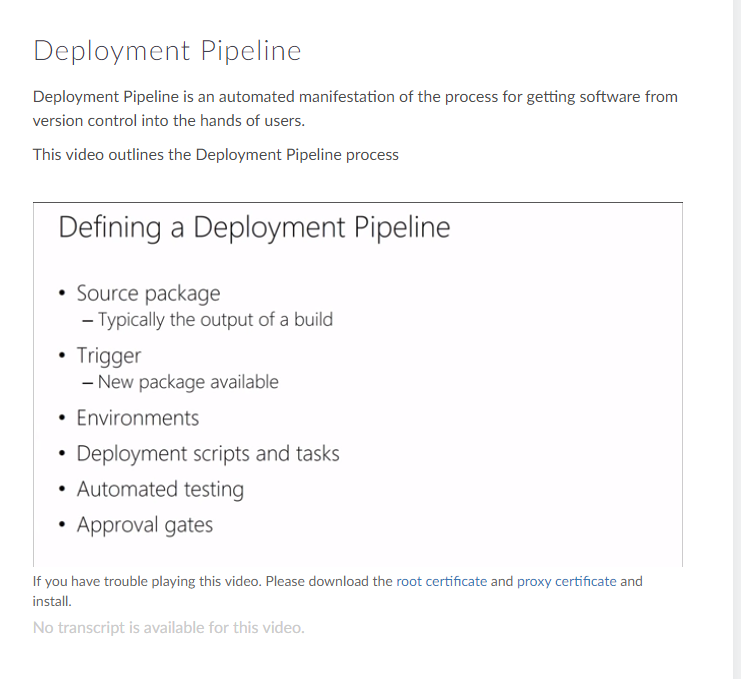




##### Deployment Pipeline

Deployment Pipeline is an automated manifestation of the process for getting software from version control into the hands of users.

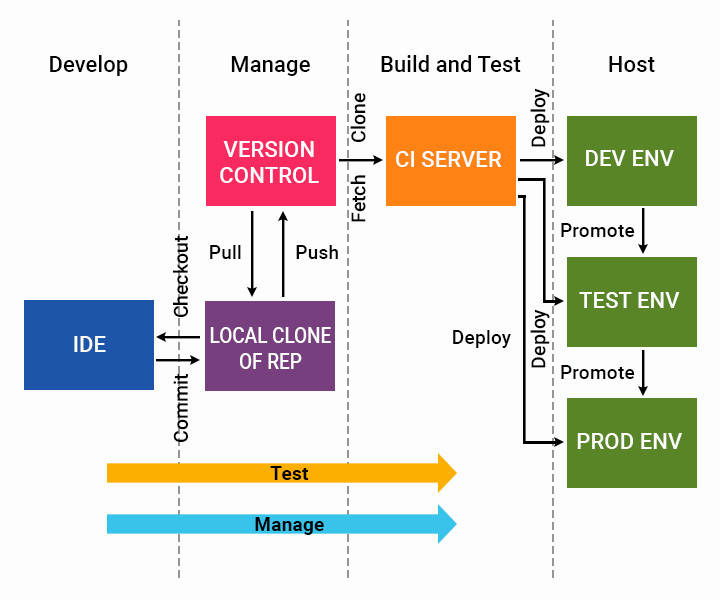
This video outlines the Deployment Pipeline process



Patterns in Deployment Pipeline

* Build Things Once
* Execute Tests in Parallel
* Design Parallel Workflows
* Verify on Environments similar to Production
* Manage Environment in a Pipeline

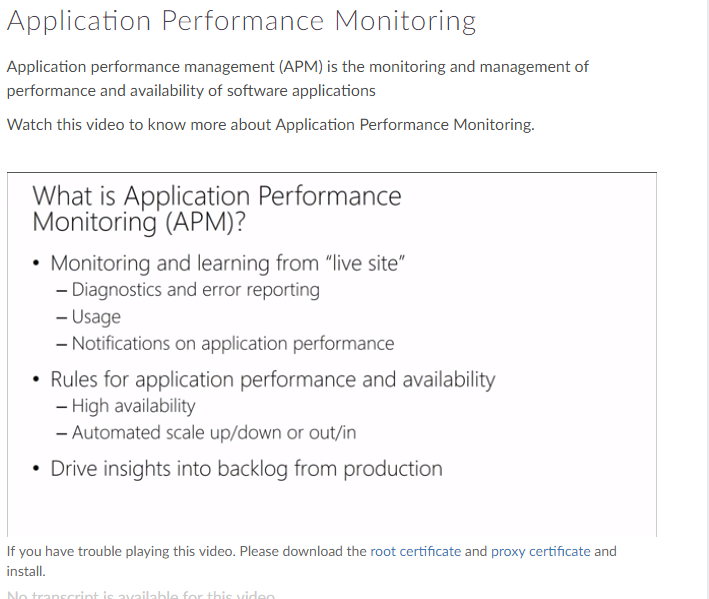
##### Deployment Pipeline Illustrated

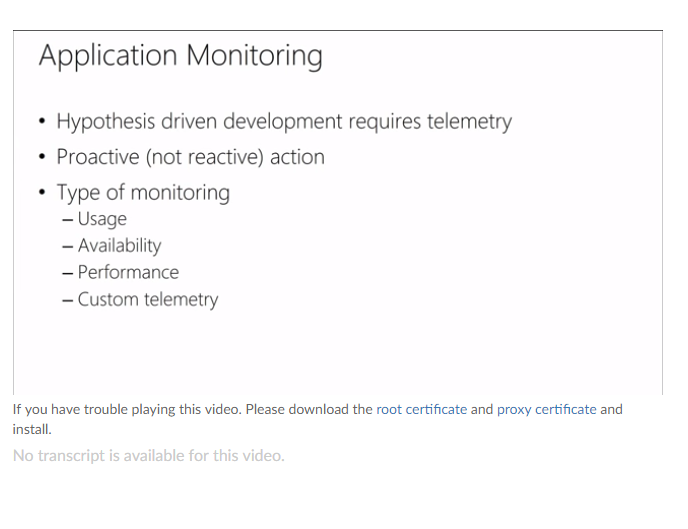


##### Application Performance Monitoring

Application performance management (APM) is the monitoring and management of performance and availability of software applications

Watch this video to know more about Application Performance Monitoring.





Steps to improve Application Monitoring:

* Consolidate Tool Set
* Enrich data with Contextual Information
* Rethink reporting of operational problems
* Automate what doesn't scale

Types of Monitoring

* Usage
* Availability
* Performance
* Customer

Other Metrics in DevOps



Now that you have understood the DevOps lifecycle better, following are some of the metrics that can help you run your team effectively in DevOps.

* ***Cycle Analytics -*** Cycle Analytics measures the time it takes to go from an idea to production for each project you have.
* ***Defect Density -*** Defects detected divided by total lines of code or number of components.
* ***Cycle Time -*** The cycle time is the period required to complete one cycle of an operation, function, or process.

Course Summary

Through this course we have covered the following

* ***DevOps Outline***: Definition and Value of DevOps.How Agile Principles have been used in DevOps.
* ***KPIs and Metrics***: The various metrics used to measure the effectiveness and efficiency of DevOps.
* ***Source Control and Automation*** : The importance of Version Control and Automation in the context of DevOps.
* ***CICD*** : The benefits of Continuous Integration and Continuous Delivery in a DevOps environment
* ***Continuous Testing and Build Pipeline*** : Value Continuous testing can bring to DevOps culture
* ***Infrastructure as Code*** : Benefits of treating Infrastructure as Code , Configuration and Release Management
* ***Application Performance Monitoring*** : The importance of Application Performance Monitoring in the context of DevOps

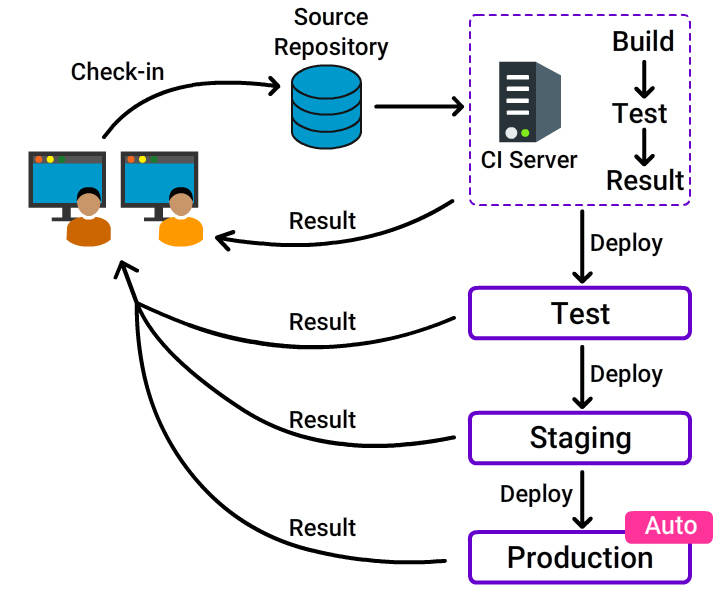
Hope you enjoyed the journey so far.

##### DevOps Lifecycle

##### Plan -> code-> build->test->release🡪deploy🡪operate->monitor

##### Course -2

##### What is Continuous Deployment?



**Continuous Deployment** is a software development practice, where software is built in such a way that it can be released automatically to production any time.

What is Continuous Deployment?...

**CD** is a logical extension of **C**ontinuous **I**ntegration, which **together** aims at giving developers and end users an incredible experience by:

* ***Detecting bugs at an early stage***.
* ***Ensuring seamless deployments*** to all environments.

Why CD?

A reliable release is highly at stake when:

* The software is deployed manually.
* Development is completed and then deployed to a production-like environment.
* Lack of proper configuration tools.

***Adapt*** Continuous Deployment ***for a stable and reliable release***.

##### Why CD?...

* **Reduced cycle time**: Reduces the time it takes from the development of code to the changed code being used by users in production.
* **Zero downtime releases**: Frequent deployment of changes enables negligible production downtime.
* **High quality**: Issues detected early and fixed immediately, ensures the best quality of the end product.
* **Immediate user feedback**: Frequent deployment enables quick user feedback.

##### Continuous Delivery vs Continuous Deployment

We need to **practice *C*ontinuous *D*elivery** in order **to do *C*ontinuous *D*eployment** but the reverse is not mandated.

Foundations of CD

* ***Configuration Management***
* ***Continuous Integration***
* ***Continuous Testing***

##### Configuration Management

**Configuration Management** often used as a **synonym** for **version control**, is the practice where every artifact from source code to build executables, test and deployment scripts, environment configuration details are:

* Stored
* Uniquely identified
* Retrieved
* Modified when needed

Configuration Management...



Configuration management enables:

* ***Reproducibility***
* ***Traceability***

***CM is the backbone*** without which it is impossible to do continuous integration, deployment pipelining and release management.

Environment Configuration

An environment is the:

* ***Set of resources needed for an application to work***.
* ***Configuration of those resources***.

Environment Configuration...

***Attributes*** that ***describe*** an ***environment*** are:

* ***Hardware configuration*** (amount of memory, the number, type of CPUs and so on).
* ***Network infrastructure***
* ***Operating environment***
* ***Middleware configuration*** (software such as database servers, messaging systems, application and web servers).