

|  |
| --- |
|  |

**Continuous Deployment**

**Find A Grave**

# Table of Contents

[Table of Contents 2](#_Toc423524654)

[1. Introduction 3](#_Toc423524655)

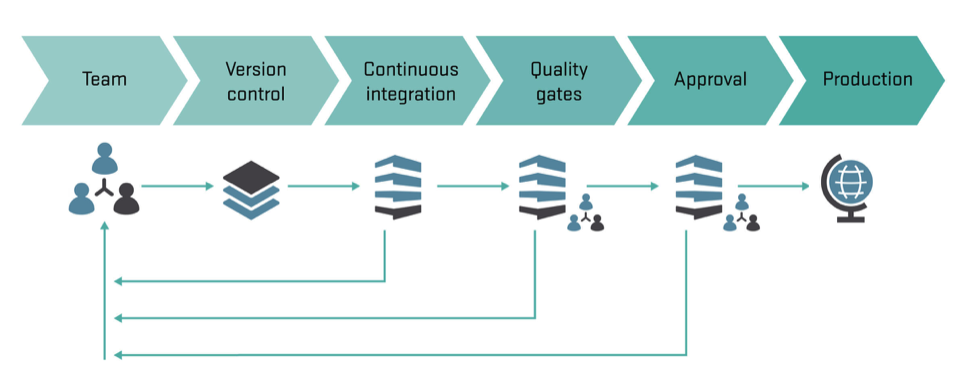
[2. Princilpe to continious dePloyment 3](#_Toc423524656)

# 

# Introduction

**Continuous Delivery (CD) is a software engineering approach in which teams keep producing valuable software in short cycles and ensure that the software can be reliably released at any time. It is used in software development to automate and improve the process of software delivery.**

# PIpeline for Contituous dePloyment



# benefits of Continious deployment

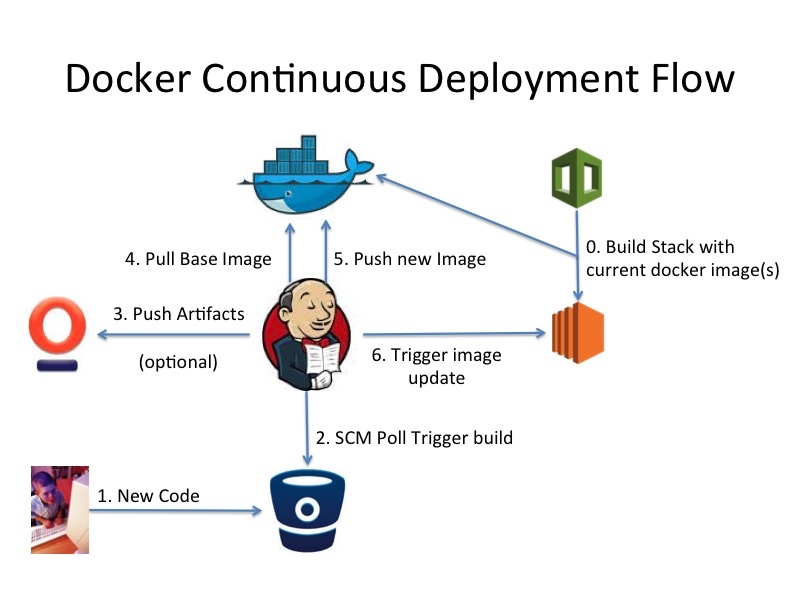
* **Accelerated Time to Market: CD lets an organization deliver the business value inherent in new software releases to customers more quickly. This capability helps the company stay a step ahead of the competition, in today’s competitive economic environment.**
* **Building the Right Product: Frequent releases let the application development teams obtain user feedback more quickly. This lets them work on only the useful features. If they find that a feature isn’t useful, they spend no further effort on it. This helps them build the right product.**
* **Improved Productivity and Efficiency: Significant time savings for developers, testers, operations engineers, etc. through automation.**
* **Reliable Releases: The risks associated with a release have significantly decreased, and the release process has become more reliable. With CD, the deployment process and scripts are tested repeatedly before deployment to production. So, most errors in the deployment process and scripts have already been discovered. With more frequent releases, the number of code changes in each release decreases. This makes finding and fixing any problems that do occur easier, reducing the time in which they have an impact.**

# What is Docker

**Docker is an open-source project that automates the deployment of applications inside software containers, by providing an additional layer of abstraction and automation of operating-system-level virtualization on Linux. Docker uses resource isolation features of the Linux kernel such as cgroups and kernel namespaces to allow independent "containers" to run within a single Linux instance, avoiding the overhead of starting and maintaining virtual** **machines**.

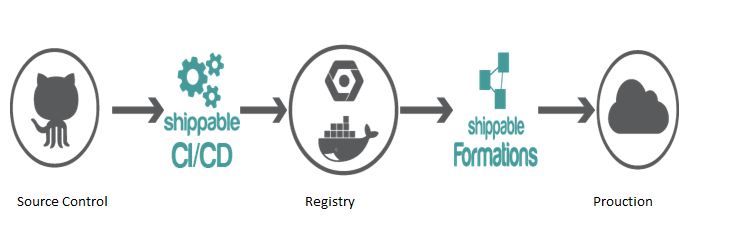
# benefits of using docker in deployment

* **Rapid application deployment – containers include the minimal runtime requirements of the application, reducing their size and allowing them to be deployed quickly.**
* **Portability across machines – an application and all its dependencies can be bundled into a single container that is independent from the host version of Linux kernel, platform distribution, or deployment model. This container can be transferred to another machine that runs Docker, and executed there without compatibility issues.**
* **Version control and component reuse – you can track successive versions of a container, inspect differences, or roll-back to previous versions. Containers reuse components from the preceding layers, which makes them noticeably lightweight.**
* **Sharing – you can use a remote repository to share your container with others. Red Hat provides a registry for this purpose, and it is also possible to configure your own private repository.**
* **Lightweight footprint and minimal overhead – Docker images are typically very small, which facilitates rapid delivery and reduces the time to deploy new application containers.**
* **Simplified maintenance – Docker reduces effort and risk of problems with application dependencies.**

****

# Find A Grave Jenkins and Docker Setup

* **For coninious deployment we follow this Pipeline**



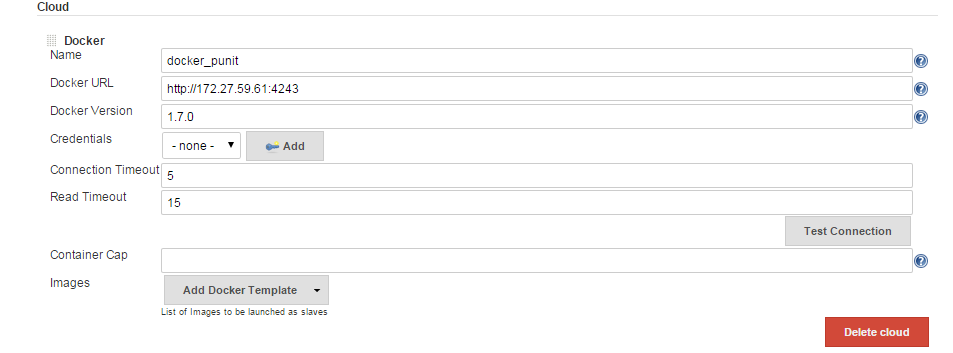
* **First source code of pushed to Source control system (GIT).**
* **This triggers the Jenkins Continuous Server Build which creates platform + Application which are easily deployable on any server.**
* **Then we commit this Artifact and pushed it into the Repository.**
* **This this artifact which contains source code and it’s platform are deployed to the production server.**

# Jenkins Configuration

* **For setting up the entire pipe line we have used various Jenkins plugins and fired various commands on the Execute shell..**
* **List of plugins which are used in the Project :**
  + GIT
  + Docker Plugin(CD)(It depends on Commons Plugin)
  + Docker Commons Plugin
  + Cloud Bees Docker Custom Build Environment Plugin(For CI)
  + Cloud Bees Docker Build and Publish plugin(For CD)
  + Hudson SCP publisher plugin
  + Email Extension Plugin

# Jenkins Global Configuration

* **Setting up Docker for Jobs**

****