



CI/CD

Pipelines



Cegos Group

inspire
qualify
change

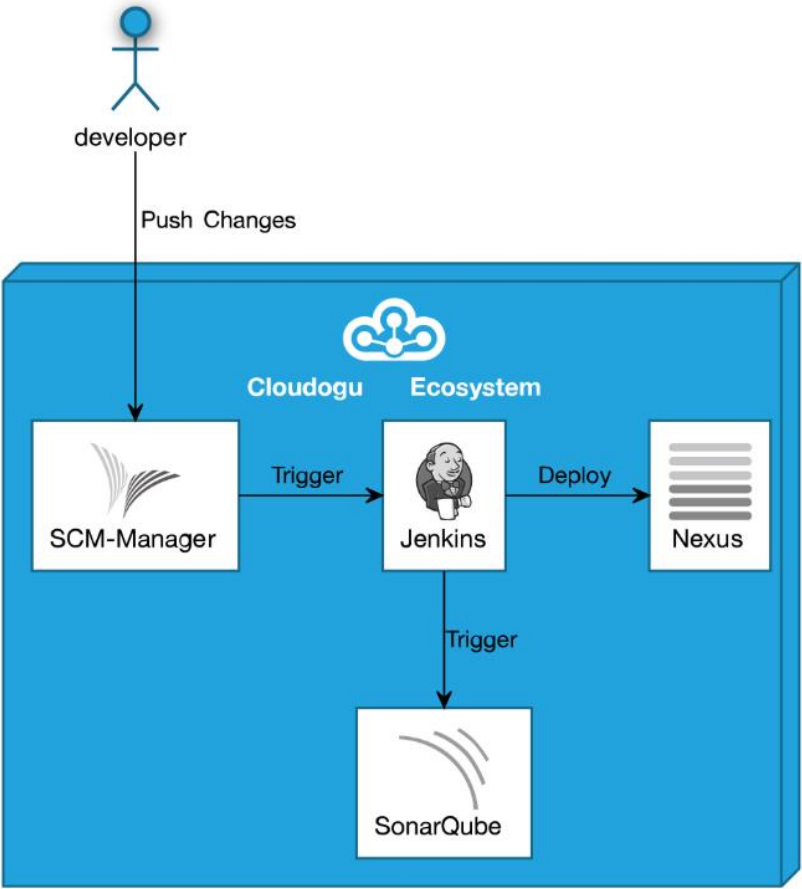
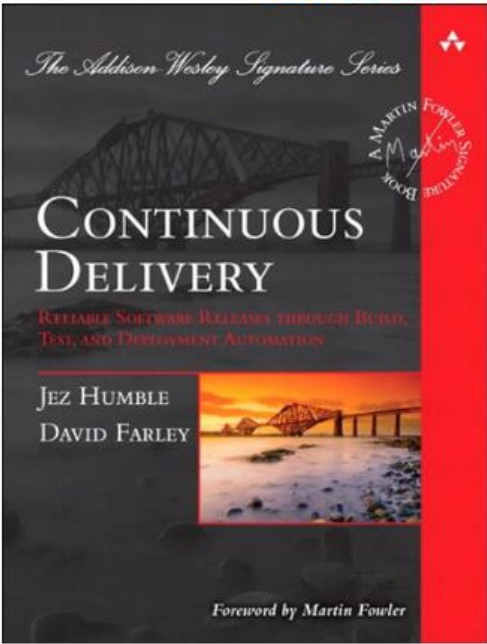


Marktdruck

- steigende Anforderung hinsichtlich der Entwicklungsgeschwindigkeit neuer Features
 - immer schneller implementieren
- Features müssen aber auch in Produktion gebracht werden.
- Deployments (Bereitstellung der Software)
 - erfolgen oft manuell
 - sind damit fehleranfällig
 - binden Ressourcen
 - dauern möglicherweise lange

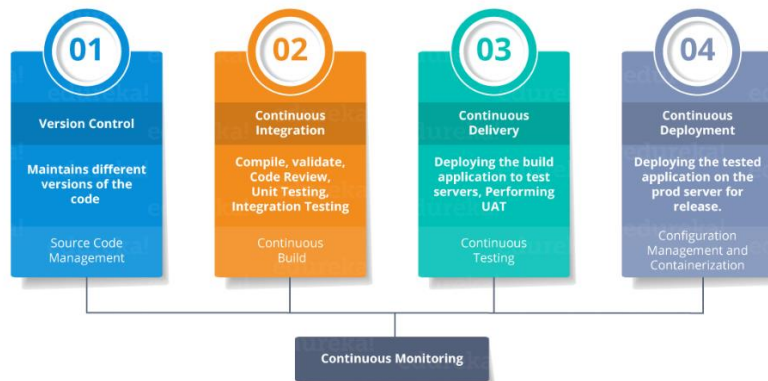
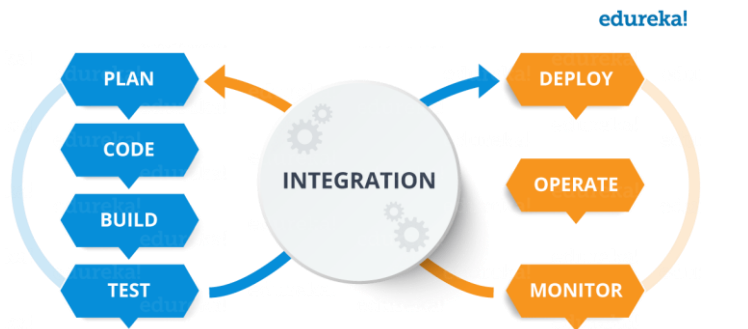
Lösung

- vollständige **Automatisierung** = Continuous Delivery
 - mittels Delivery Pipeline





DevOps








- **DevOps** ist ein Softwareentwicklungsansatz
- Er umfasst:
 - kontinuierliche Entwicklung,
 - kontinuierliches Testen,
 - kontinuierliche Integration,
 - kontinuierliche Bereitstellung
 - und kontinuierliche Überwachung der Software während ihres gesamten Entwicklungslebenszyklus
- Dies ist der Prozess, den alle Spitzenunternehmen anwenden, um qualitativ hochwertige Software und kürzere Entwicklungslebenszyklen zu entwickeln, was zu einer größeren Kundenzufriedenheit führt, etwas, das jedes Unternehmen wünscht.



Tools für CD-Pipeline

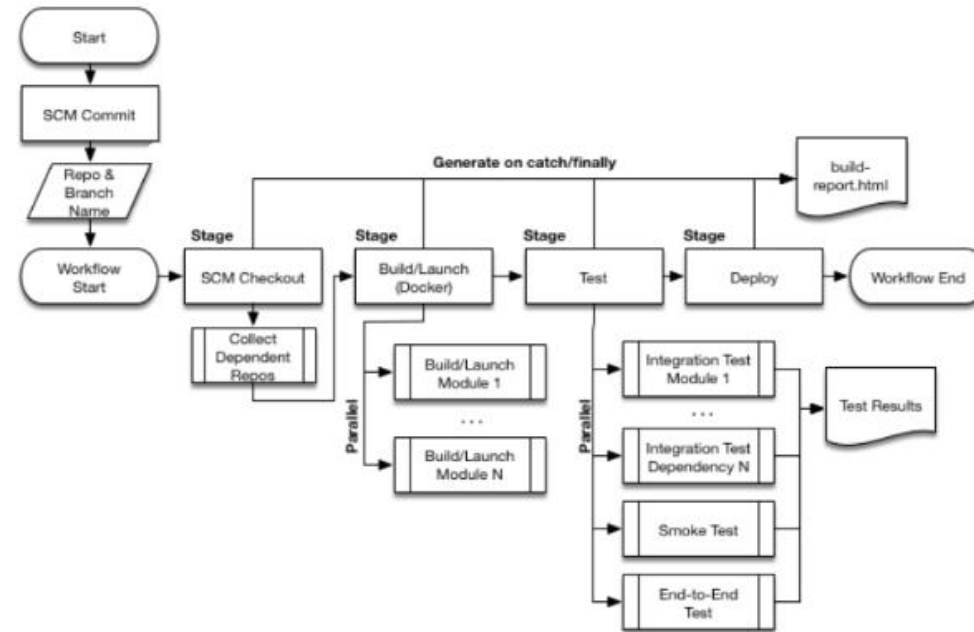
- Zur Umsetzung einer CD-Pipeline können diverse Tools eingesetzt werden.

| |  Jenkins |  circleci |  TeamCity |  Bamboo |  GitLab |
|-----------------------------|---|--|--|--|--|
| Open source | Yes | No | No | No | No |
| Ease of use & setup | Medium | Medium | Medium | Medium | Medium |
| Built-in features | 3/5 | 4/5 | 4/5 | 4/5 | 4/5 |
| Integration | ★★★★★ | ★★★ | ★★★★★ | ★★★ | ★★★★★ |
| Hosting | On premise & Cloud | On premise & Cloud | On premise | On premise & Bitbucket as Cloud | On premise & Cloud |
| Free version | Yes | Yes | Yes | Yes | Yes |
| Build Agent License Pricing | Free | From \$39 per month | From \$299 one-off payment | From \$10 one-off payment | From \$4 per month per user |
| Supported OSs | Windows, Linux, macOS, Unix-like OS | Linux or MacOS | Windows, Linux, macOS, Solaris, FreeBSD and more | Windows, Linux, macOS, Solaris | Linux distributions: Ubuntu, Debian, CentOS, Oracle Linux |

Top 5 CI/CD tools in 2020



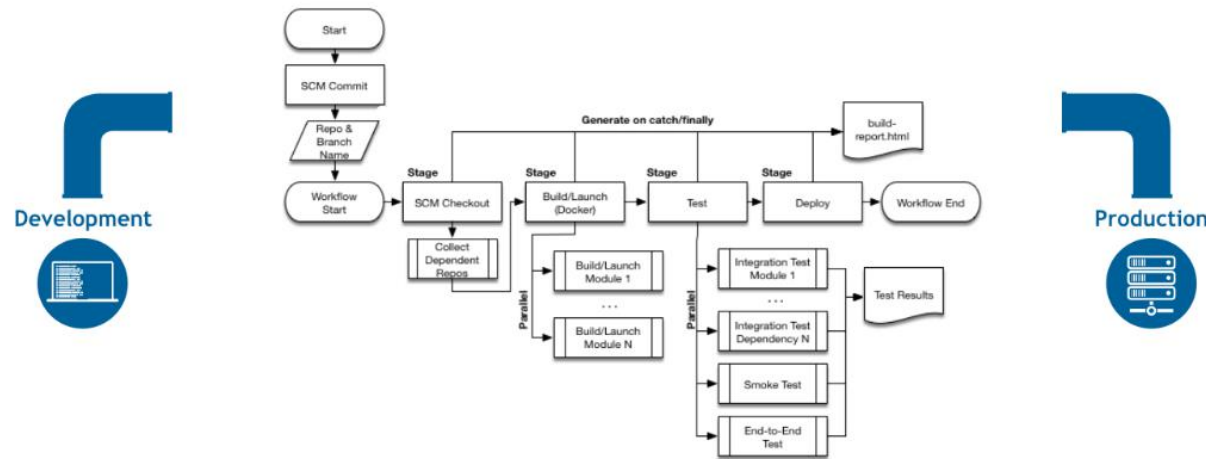
Delivery-Pipeline



- Auf abstrakter Ebene ist eine **Delivery-Pipeline**
 - eine **automatisierte Manifestation Ihres Prozesses**, um Software aus der Versionskontrolle in die Hände Ihrer Benutzer zu bekommen.



Delivery-Pipeline



- Die **Delivery-Pipeline** modelliert diesen Prozess,
- Ein kontinuierlichen Integrations- und Release-Management-Tool ermöglicht,
 - den **Fortschritt jeder Änderung** zu sehen und zu kontrollieren,
 - während sie von der **Versionskontrolle** über verschiedene **Test- und Bereitstellungsphasen** bis zur **Freigabe an die Benutzer** verläuft.
- In Ermangelung einer automatisierten Pipeline müssten Beteiligte diese Schritte immer noch manuell und damit weit weniger produktiv durchführen.



Delivery-Pipeline Ausbaustufen

- **Continuous Deployment, Delivery und Integration** sind drei unterschiedliche „Ausbaustufen“ ein und derselben Idee:
 - Der kontinuierlichen Weiterentwicklung von Software.
- **Continuous Delivery** stellt dabei die höchste Ausbaustufe dar
 - hier werden alle Änderungen automatisch bis zum Kunden hin ausgeliefert. Es gibt also keine klassischen Releases.
- Bei **Continuous Deployment** werden Änderungen zwar automatisch auf einem Server deployed, aber noch nicht an den Kunden ausgeliefert.
 - Die Änderungen können auf diesem Server noch manuell getestet werden um dann in Form eines Releases an den Kunden übergeben zu werden.
- **Continuous Integration** ist die schwächste Form der kontinuierlichen Entwicklung.
 - Bei ihr werden alle Änderungen automatisch gebaut und getestet, es findet aber kein Deployment auf einem Server statt.



Die Situation

- Mehrere Entwicklungsteams,
 - die agil arbeiten wollen,
 - stehen vor dem gleichen Problem,
 - während sie ein neues Projekt aufsetzen.
- Dies beinhaltet die Aufgabe,
 - Vereinbarungen über die verwendeten Werkzeuge zu treffen,
 - eine Entwicklungsumgebung einzurichten
 - und sogar die Notwendigkeit, diese zu betreiben und zu warten.



Die Situation

- Jeder Entwickler
 - baut seine eigene lokale Entwicklungsumgebung,
 - was zu einem Mangel an Konsistenz führen kann.
- Während der erste Quellcode erstellt wird,
 - lassen sich lokale Änderungen möglicherweise nicht leicht pflegen
 - oder gar integrieren.
- Aufgrund des Mangels an konsistenten Repositories, Build-Prozessen und Berichten
 - können wichtige Kommunikation und Integration verloren gehen.

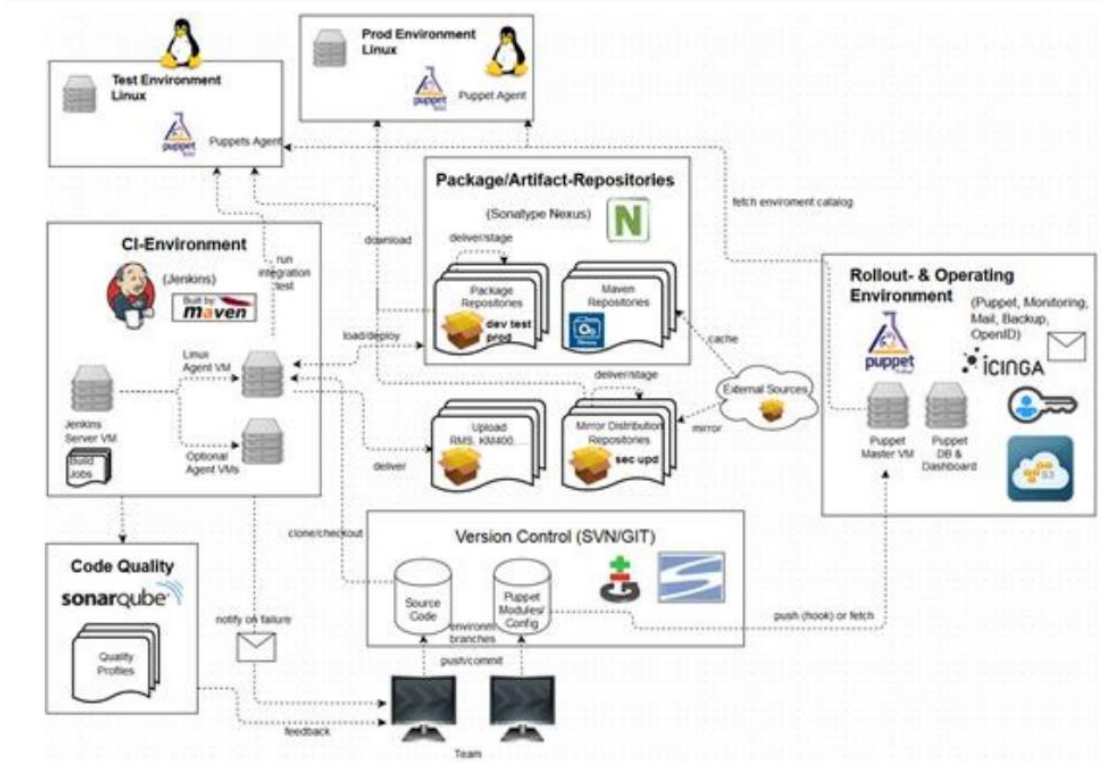
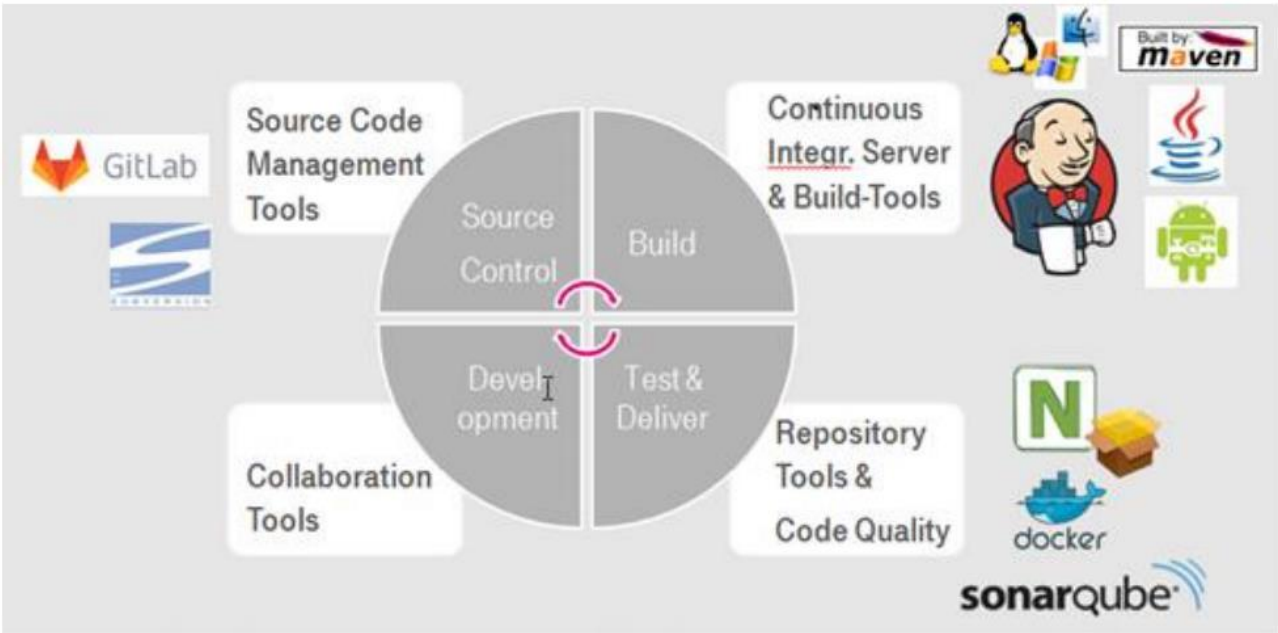


Die Lösung

- Eine einfache Lösung ist die Verwendung von CI.
- Die "Telekom Workbench" liefert alle notwendigen Werkzeuge in einer bereits vorkonfigurierte CI-Umgebung.
- Es ist kein zusätzlicher Aufwand erforderlich.
- Installation, Wartung und sogar Backups werden von der "Telekom Workbench" zur Verfügung gestellt.
- Continuous Integration in jeder Zielumgebung einsetzen
 - Continuous Integration Server **Jenkins** als Kernkomponente
 - sowie weiteren unterstützenden Werkzeugen wie
 - Versionskontrolle (**Gitlab**), Qualitätssicherung (**SonarQube**), dem Artefakt-/Paket-Repository-Manager (**Nexus**) sowie weiteren Tools und Plugins

Telekom Workbench

- Telekom Workbench is
 - a selected and pre-configured **combination of OpenSource** tools for Continuous Integration (CI).
 - Tools will be available for developer teams in dedicated Workbench instances.





GitLab

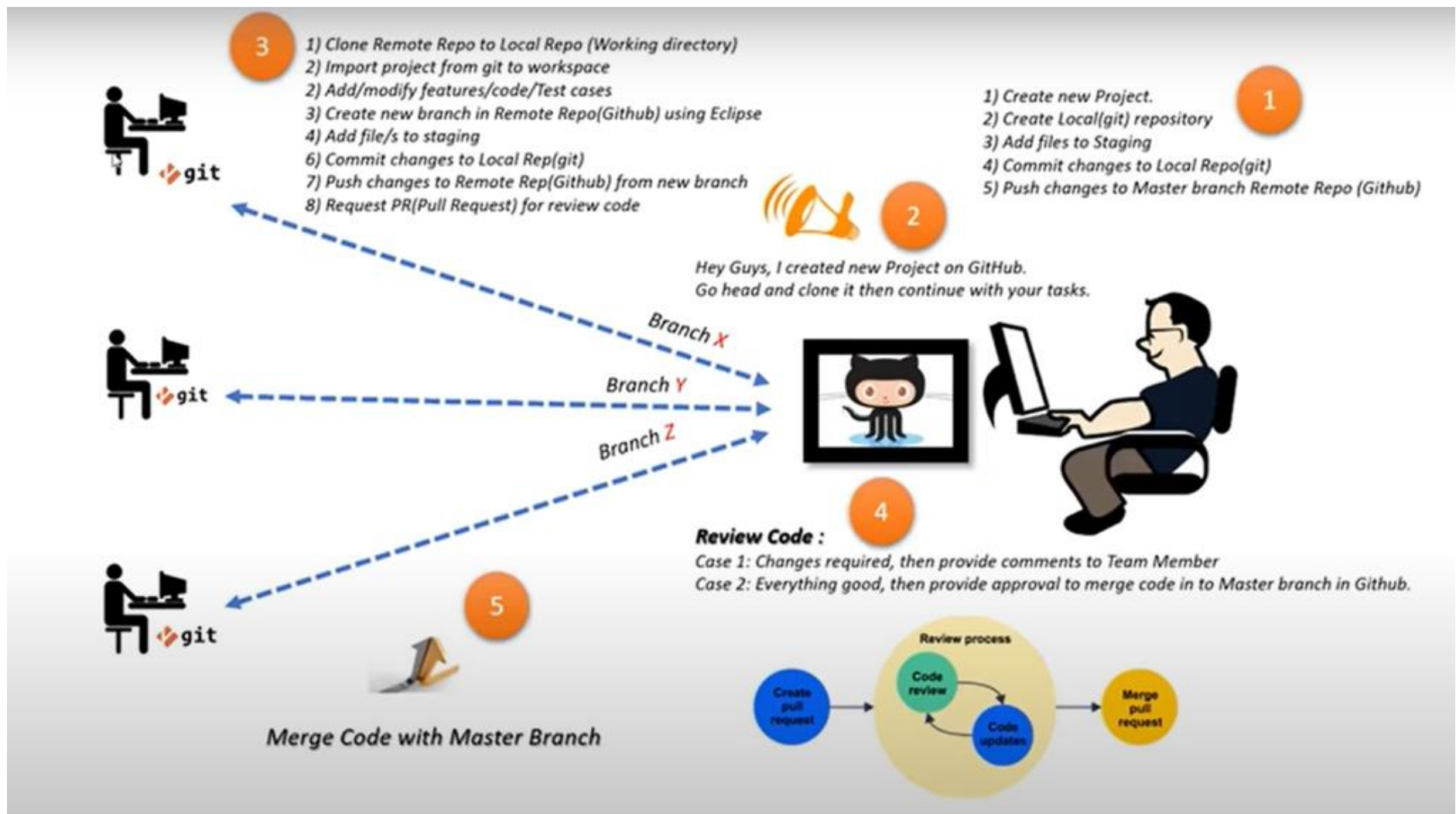


- **GitLab** ist ein **Version Control System(VCS)**.
- Es basiert vollständig auf **Git**, einem verteilten **Versionierungssystem**, das als Open-Source-Software zur Verfügung gestellt wird.
- **Git** ist das mit Abstand am weitesten verbreitete VCS der Welt.
- Hauptaufgabe der webbasierten Versionsverwaltung:
 - alle Änderungen an Dateien und ihrem Quellcode speichern und dokumentieren, sodass diese jederzeit nachvollzogen werden können.
- <https://rogerdudler.github.io/git-guide/index.de.html>

Gitlab@Telekom Workbench

Summary: Gitlab is the Workbench **Source Code Management Tool** provided with additional features like Issues, Wiki, Code Review and Private Docker Repository.

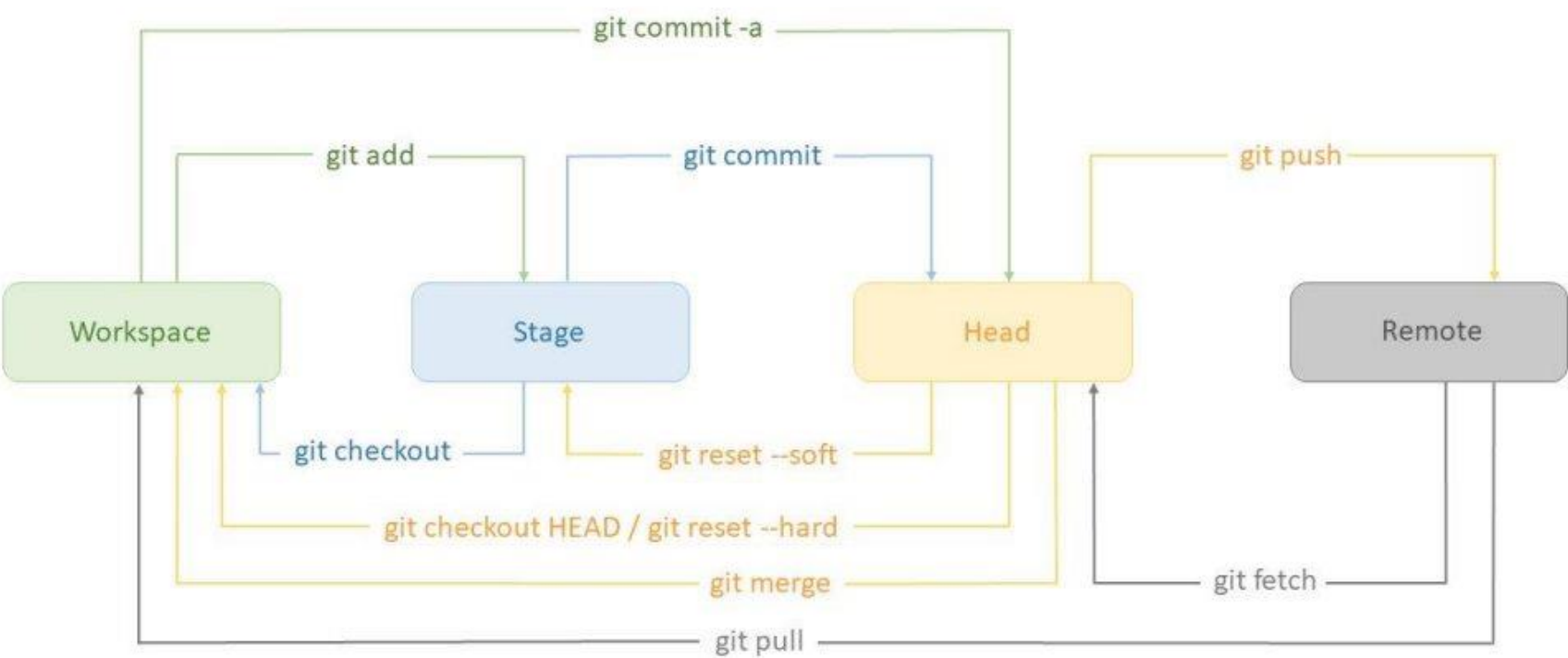
Git Workflows





Git Workflows

- Die wichtigsten Befehle für den täglichen Workflow

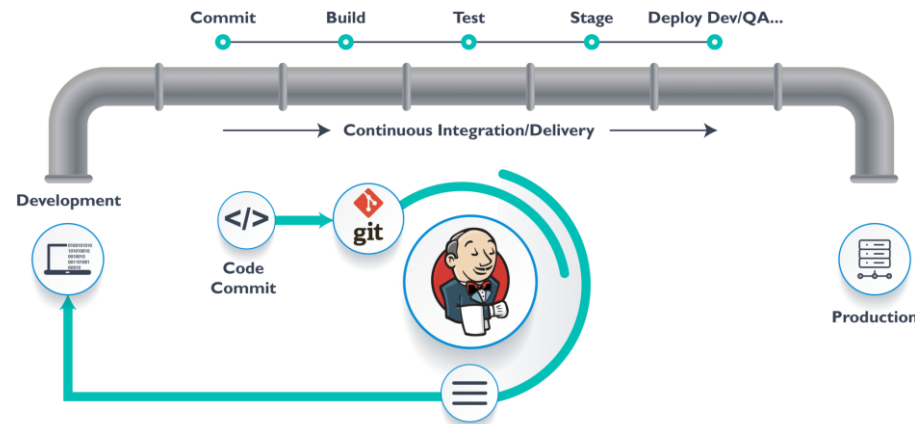




Jenkins



- Jenkins ist ein **Continuous Integration Server**
- stellt verschiedene Schnittstellen und Werkzeuge zur Verfügung, um den gesamten CI/CD Prozess zu automatisieren

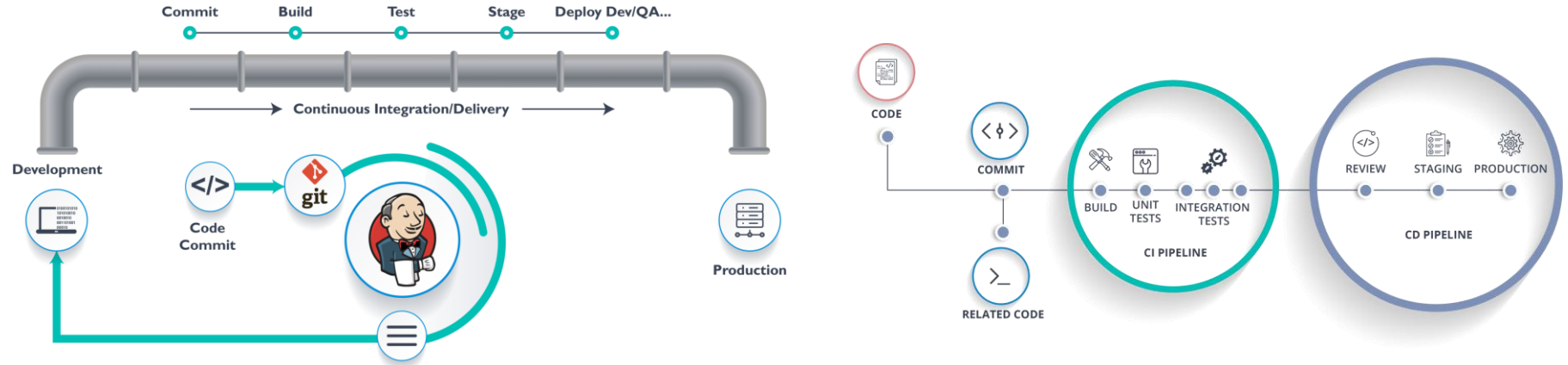


Jenkins@Telekom Workbench

Summary: Jenkins is the Workbench **Continuous Integration Server** for build and test automation.



CD Pipeline mit Jenkins



- **Jenkins** zieht den Code aus **Git**
 - und bringt ihn dann in die **Commit-Phase**, wo der Code aus jedem Branch übergeben wird.
 - Die **Build-Phase** ist die Phase, in der wir den Code kompilieren.
- Wenn es sich um Java-Code handelt, verwenden wir Werkzeuge wie **Maven** in Jenkins und kompilieren dann diesen Code, der zur Ausführung einer Reihe von Tests eingesetzt werden kann.
- Diese Testfälle werden wieder von **Jenkins** beaufsichtigt.
- Anschließend wird der Code auf den **Staging-Server** übertragen, um ihn mit Docker bereitzustellen. Nach einer Reihe von **Unit-Tests** oder Zustandsprüfungen geht er in die Produktion über.



Nexus



- **Nexus** ist ein sogenannter **Repository-Manager** für das **Build-Tool Maven**,
- organisiert die Verwaltung von internen und externen Repositories, einschließlich des Maven Central Repositories.
- Damit bietet der Repository-Manager Zugriff auf Bibliotheken und Plug-Ins (sogenannte Artefakte), die für die Erstellung eines Builds benötigt werden.

Nexus3@Telekom Workbench

Summary: Nexus3 is the new Workbench **Artifact Repository** with repositories for libraries and packages in various formats.



SonarQube



- Dieses Tool wird für die **Analyse von Code** und **Codequalität** verwendet
- Auffangen von Fehlern und Schwachstellen in der Anwendung
 - mittels automatisierten Regeln für die statische Code-Analyse.

SonarQube@Telekom Workbench

Summary: SonarQube is the Workbench **Code Quality Management** tool with support for different languages and build integration.



Maven



- Build-Management-Tool Maven
 - basiert auf Java
 - Damit kann man Java-Programme standardisiert erstellen und verwalten.
- Buildprozess
 - Compile
 - Test
 - Deploy
 - Build
- unabhängig von der IDE beschreibbar



Maven

- Maven versucht, den Grundgedanken „**Konvention vor Konfiguration**“ (englisch *Convention over Configuration*) konsequent für den gesamten Zyklus der Softwareerstellung abzubilden.
- Dabei sollen Software-Entwickler so unterstützt werden, dass möglichst viele Schritte automatisiert werden können.
 - Anlage eines Softwareprojekts über das Kompilieren,
 - Testen und „Packen“
 - Verteilen der Software auf Anwendungsrechnern
- Folgt man dabei den von Maven vorgegebenen Standards,
 - braucht man für die meisten Aufgaben des Build-Managements nur sehr wenige Konfigurationseinstellungen zu hinterlegen,
 - um den Lebenszyklus eines Softwareprojekts abzubilden.
- Standardlebenszyklen
 - https://de.wikipedia.org/wiki/Apache_Maven



Gradle



- Build-Management-Tool Gradle
 - basiert auf Java , Groovy
 - Java-Programme standardisiert erstellen und verwalten
- Buildprozess
 - Compile
 - Test
 - Deploy
 - Build
- unabhängig von der IDE beschreibbar

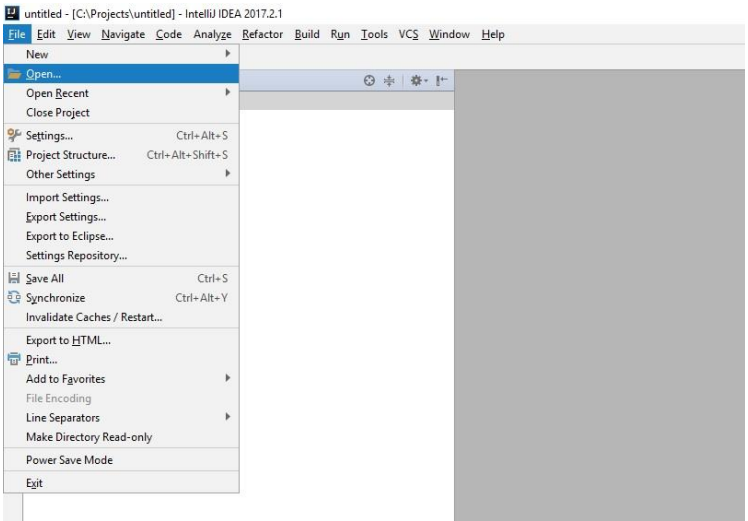


Gradle

- Nutzt Standardverzeichnislayout
 - Dadurch nichts zu konfigurieren
- Build erzeugt build-Verzeichnis
- Buildablauf definieren in Datei **build.gradle**
 - liegt im Hauptverzeichnis
 - enthält Tasks wie Kompilieren oder Ausführen von Tests
 - enthält Hinweise benötigten externen Libraries
- Sourcen kompilieren
 - **gradle build**
- Ausführen von Unit-Tests
 - **gradle test**



IntelliJ



- integrierte Entwicklungsumgebung (IDE)
 - für Programmiersprachen Java, Kotlin, Groovy und Scala.
 - gebaut von JetBrains
 - ab der Version 9.0 existieren zwei verschiedene Editionen,
 - kostenpflichtige Ultimate Edition
 - kostenfreie Open Source Community Edition.

Copyright und Impressum

© Integrata AG

Integrata AG
Zettachring 4
70567 Stuttgart

Alle Rechte, einschließlich derjenigen des auszugsweisen Abdrucks, der fotomechanischen und elektronischen Wiedergabe vorbehalten.