Continuous Integration

Dem Grundsatz nach ist Continuous Integration (CI) eine Summe von Prinzipien die an den täglichen Arbeitsablauf eines Entwickerteams angepasst sind.
Ein VCS bildet die Basis des CI Prozesses. Wenn ein Entwickler seinen Code in das Repository Hochlädt, prüft ein automatisches System den geänderten Code mittels vorab definierter Tests, dass das gesamte Programm weiterhin einwandfrei funktioniert. Diese Verifizierung soll Maschinenunabhängig durchgeführt werden, damit sowohl innerhalb der Entwicklung als auch bei der Auslieferung am Ende eine unabhängige und kontinuierliche Funktionalität garantiert ist.(vgl. Web4)

Üblicherweise wird CI in Verbindung mit einer Agilen Softwareentwicklung genutzt. Dieses Vorgehen unterstützt schon, sowohl die Einteilung in kleine Aufgaben als auch die Fertigstellung dieser innerhalb eines kurzen Zeitraums. (vgl. Web5)

CI beinhaltet eine automatisierte Software-Release-Pipeline bestehend aus drei Schritten und umfasst den kompletten Entstehungsprozess, von der Idee bis Freigabe an den Kunden.(vgl. Web5)

(Foto2)

Das Ziel von CI ist die Optimierung des Entwicklungsprozesses. Dadurch ist es den Teammitgliedern möglich parallel und unabhängig voneinander zu arbeiten und ihre Änderungen nicht nur in das gleiche Repository, sondern auch in den gleichen Branch Hochladen zu können. Die Beteiligten sind dazu angehalten regelmäßig beziehungsweise täglich ihre Änderungen Hochzuladen um den Testlauf durch das Prüfsystem möglichst kurz zu halten. Hinzu kommt das im Falle eines Fehlers, die Korrektur bei kleinen Änderungen wesentlich schneller möglich ist, wie beispielsweise bei einer kompletten neuen Funktionalität. (vgl. Web4)

Der erste Schritt, die **Continuous Integration**, beinhaltet genau diesen Vorgang des Hochladens seitens mehrerer Entwickler und den Versuch ihren Code in das Gesamtprojekt zu integrieren.

Daraufhin folgt der zweite Schritt, die **Continuous Delivery**, welcher die Funktion des automatischen Systems beschreibt. Die Aufgabe dieses automatischen Systems übernimmt ein so genannter Continuous Integration Server, dessen einzige Aufgabe es ist, ein oder mehrere Repository auf Änderungen zu Überprüfen, gegebenenfalls den Code Auszuchecken und mittels vorab definierter Kommandos den Build-Prozess zu starten. Wie genau ein Build aussieht variiert abhängig von der Verwendeten Programmiersprache und des verwendeten Frameworks.
Idealerweise besteht ein Build nicht nur aus dem compilieren des Codes, sondern enthält auch diverse Tests um das Programm als ganzes auf korrekte Funktionalität zu prüfen. Das Ziel sollte hierbei immer die höchstmögliche Testabdeckung sein, ohne dabei die dauer eines Builds länder als 10 Minuten werden zu lassen. (vgl. Web4)

Die dritte und letzte Phase, **Continuous Deployment**, umfasst die automatische Verteilung des zuvor Bereitgestellten Programmes an den Kunden. Dazu kommt es nur, wenn die bisherigen Schritte fehlerfrei durchlaufen wurden. Wurde ein Build nicht fehlerfrei durchlaufen, wird der gesamte Prozess unterbrochen, solange bis der Fehler behoben wurde. Umgesetzt wird die Verteilung an den Endnutzer beispielsweise durch Skripte oder andere Software, die automatisch das Programm auf öffentliche Server, App Stores oder andere Systeme verteilt.(vgl. Web5)