Die Abkürzung CI/CD steht für Continuous Integration/Continuous Delivery und bezeichnet eine Vorgehensweise in der Softwareentwicklung, bei der der gesamte Entwicklungsprozess von der Codeänderung bis zur Bereitstellung der Software automatisiert und ständig überwacht wird.

CI/CD-Pipeline mit UP

Continuous Improvement

Test und Release Automatisierung

Die Automatisierung im CI/CD-Prozess ist von entscheidender Bedeutung für eine erfolgreiche und effiziente Softwareentwicklung. Im Folgenden sind einige der wichtigsten Vorteile der Automatisierung im CI/CD-Prozess aufgeführt:

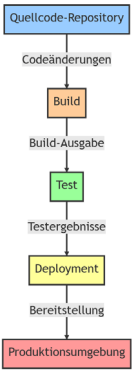

Abb: Beispiel CI/CD Prozess

- **Beschleunigung des Entwicklungsprozesses:** Die Automatisierung von Aufgaben wie Build, Test und Bereitstellung kann die Entwicklungszeit erheblich verkürzen.
- **Verbesserung der Softwarequalität:** Automatisierte Tests können dazu beitragen, Fehler und Schwachstellen der Software frühzeitig zu erkennen und zu beheben.
- **Erhöhung der Produktivität:** Automatisierte Prozesse können Entwickler von Routineaufgaben entlasten.
- **Reduzierung von Fehlern:** Durch die Automatisierung von Prozessen können menschliche Fehler minimiert werden.
- **Skalierbarkeit:** Automatisierte CI/CD-Prozesse können leicht auf eine wachsende Anzahl von Anwendungen, Repositories und Entwicklern skaliert werden.

Insgesamt kann die Automatisierung im CI/CD-Prozess Entwicklern helfen, schneller bessere Software bereitzustellen, die weniger Fehler enthält und eine höhere Produktivität und Effizienz aufweist.

Effizientes Testen mit Jenkins-Multibranch-Pipelines

Die Verwendung einer Jenkins-Multibranch-Pipeline für das Testen bietet eine effiziente Möglichkeit, Änderungen an Quellcode-Repositories automatisch zu testen und zu überwachen. Hier sind die Schritte, um Jenkins-Multibranch-Pipelines für das Testen zu verwenden:

- Erstellung einer Multibranch-Pipeline-Konfiguration in Jenkins, die die verwendeten Quellcode-Repositories und die Art des Builds angibt.
- Hinzufügen von Testschritten zu jedem Schritt der Pipeline, um sicherzustellen, dass der Code ordnungsgemäss kompiliert, getestet und bereitgestellt wird.
- Überwachung von Code-Änderungen durch das SCM-Tool (Source Code Management) und Auslösen der Pipeline, um Tests automatisch auszuführen, wenn neue Änderungen erkannt werden.
- Implementierung von automatisierten Testfällen und Integrationstests, um sicherzustellen, dass die Anwendung wie erwartet funktioniert und mit anderen Komponenten in der Umgebung interoperabel ist.
- Konfiguration von Jenkins, um benachrichtigt zu werden, wenn Tests fehlschlagen oder wenn ein Build nicht erfolgreich war, damit schnell Massnahmen ergriffen werden können.

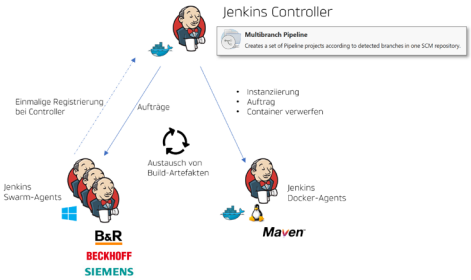

Abb: Build- und Testanordnung mit Jenkins Agenten

Durch die Verwendung von Jenkins-Multibranch-Pipelines können Entwickler sicherstellen, dass ihr Code automatisch getestet und bei Bedarf sofort repariert wird. Die Überwachung von Änderungen und automatisierte Tests können dazu beitragen, sicherzustellen, dass Fehler frühzeitig erkannt und behoben werden, was zu einer höheren Qualität und schnelleren Bereitstellung der Software führt.

Der Einsatz von Jenkins-Agenten ermöglicht das gleichzeitige Testen auf verschiedenen Windows-VMs oder auf Linux-Docker-Instanzen.

Testautomatisierung mit Jenkins und UPact Testframework

UPact von UP ist ein Testframework, das in Kombination mit Jenkins zur Durchführung von automatisierten Tests in der CI/CD-Pipeline verwendet werden kann. Die Integration von UPact in Jenkins ermöglicht eine nahtlose Durchführung von Tests und der Automatisierung der Testausführung bei jeder Änderung des Codes. Mit UPact können Berichte im JUnit Format erstellt werden, um die Ergebnisse der Tests in Jenkins anzuzeigen. Die Automatisierung der Testausführung mit UPact kann dazu beitragen, die Qualität und Zuverlässigkeit der Anwendung zu verbessern und Entwicklern schnelleres und effektiveres Feedback zu geben.

UP bietet nicht nur die Möglichkeit, PLC-Code headless aus dem Applikationsmodell zu generieren, sondern auch eine nahtlose Integration mit Jenkins. UP stellt fixfertige Skripte zur Verfügung, um den generierten Code und die durch UPact automatisierten Testfälle in der CI/CD-Pipeline von Jenkins einzubinden.

CI/CD-Pipeline mit Bitbucket, Jenkins, JFrog und UPact: Ein Praxisbeispiel

In einem aktuellen Projekt wird Bitbucket, Jenkins, JFrog und UP bzw. UPact als Techstack eingesetzt. Die Testpipeline beinhaltet folgende Schritte:

1. Commit auf dem Git Repository aktiviert die Pipeline auf dem Jenkins Build Server
2. Die Test Pipeline beinhaltet folgende ‚Stages':
   a. Das UP Applikationsmodell wird vom Git Repository geklont
   b. UP generiert aus dem Applikationsmodell den B&R Code
   c. Der B&R Code wird kompiliert und auf eine simulierte Steuerung geladen und gestartet
   d. UPact führt die verschiedenen Testszenarien aus
   e. Der Testreport im JUnit-Format wird durch den Jenkins interpretiert und angezeigt
3. Bei erfolgreichen Durchlauf wird ein Paket mit der kompilierten und lauffähigen Applikation auf dem JFrog Repositoryserver bereitgestellt

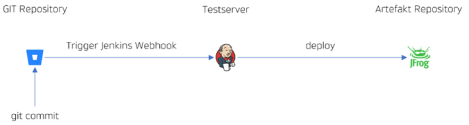

Abb: CI/CD Prozess mit Bitbucket, Jenkins, UP und JFrog

Gerne helfe ich Ihnen weiter!

René Zwingli
AVM Engineering AG
+41 71 544 60 86
rene.zwingli@avm.swiss