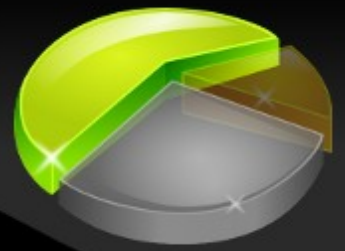
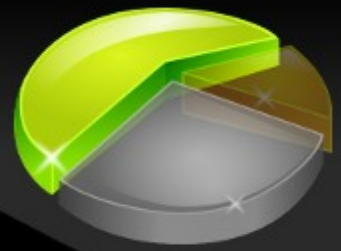
Von Continuous Integration zu Continuous Deployment



Manuel Pichler <mapi@pdepend.org>

31. Mai 2010

Über mich

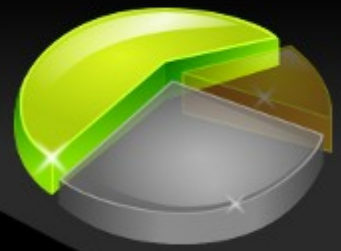


- Manuel Pichler
  - Jahrgang 1978
  - Diplom Informatiker
  - Softwarearchitekt
  - Entwickler von:
    - PHP_Depend
    - phpUnderControl
    - PHPMD

Warum kontinuierlich integrieren?

- Frühzeitige Erkennung von Fehlern
- Minimierung manueller Arbeitsschritte
- Lauffähige Softwareversion zu jeder Zeit
- Transparenz des Entwicklungsprozesses
- Zufriedenheit mit dem eigenen Produkt

Agenda

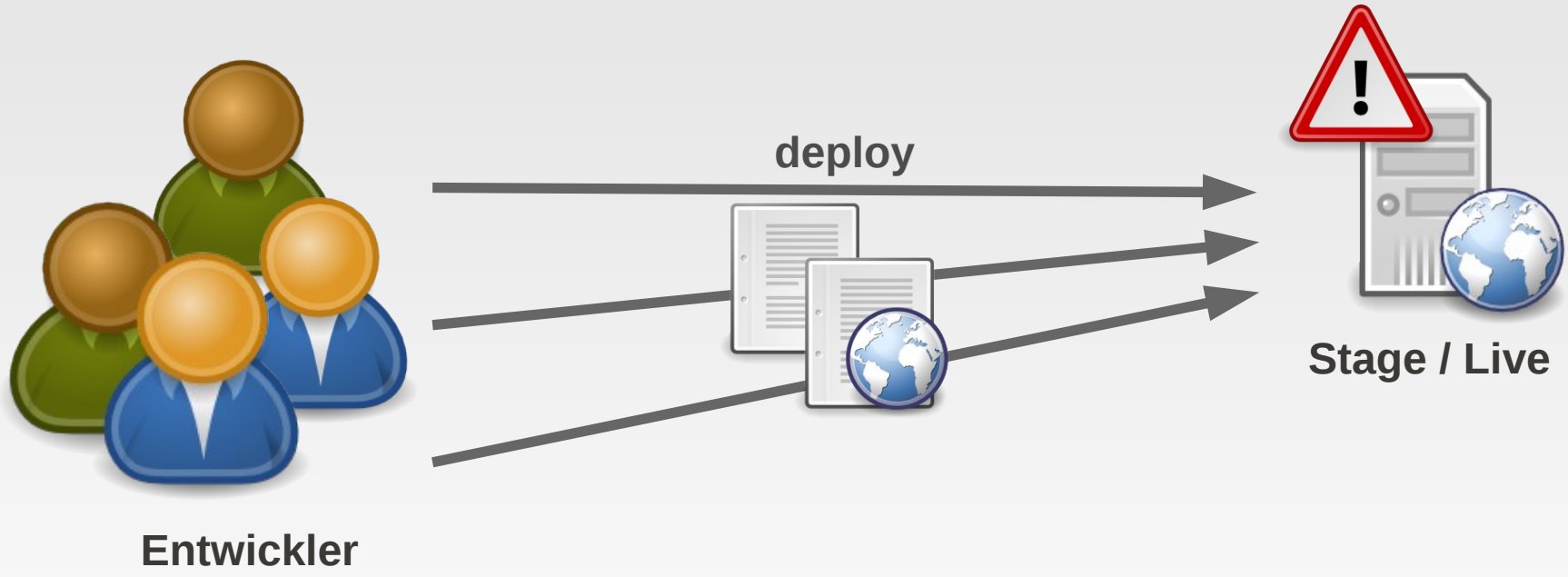


- Klassische Integration
- Grundlagen für kontinuierliche Integration
- Kontinuierliche Integration
- Prozesse und Techniken im Detail
- Continuous Deployment

Klassische Integration

Klassische Integration

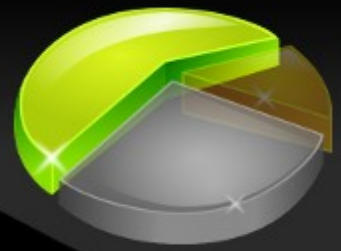
- Dieser Ansatz scheitert spätestens:
  - Mit größer werdenden Teams



Klassische Integration

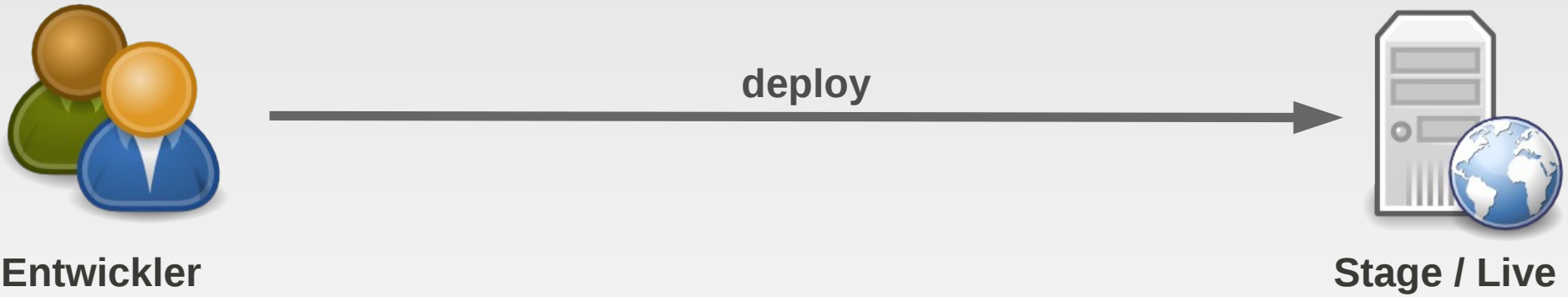
- Dieser Ansatz scheitert spätestens:
  - Mit größer werdenden Teams
  - Mit komplexeren Projekten

Agenda



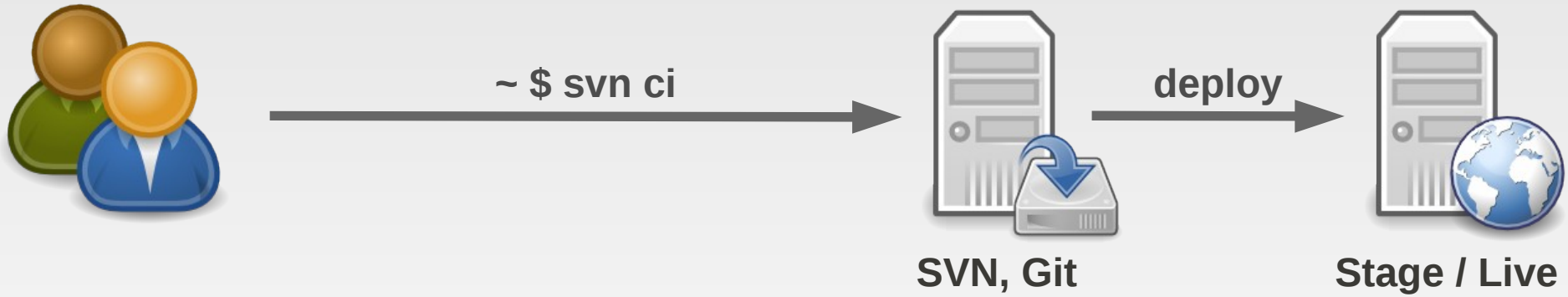
- Klassische Integration
- Grundlagen für kontinuierliche Integration
- Kontinuierliche Integration
- Prozesse und Techniken im Detail
- Continuous Deployment

Zurück zum Ausgangspunkt



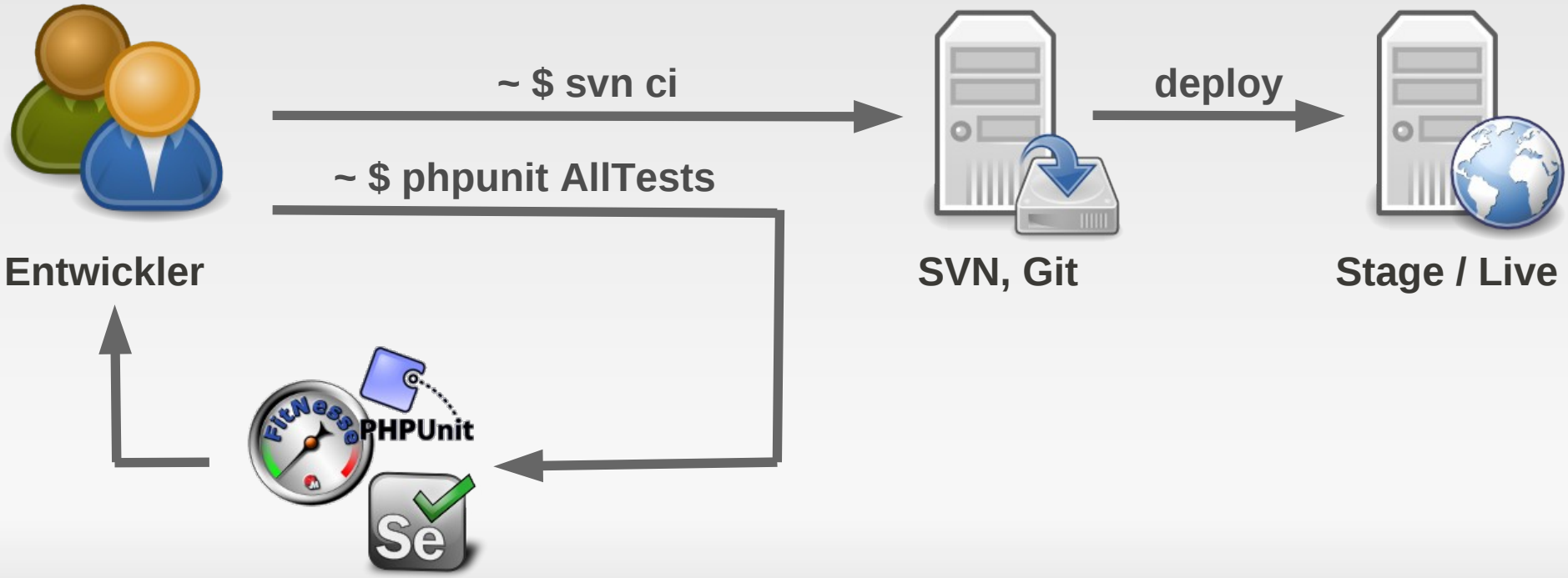
Grundvoraussetzungen

- Versionsmanagement



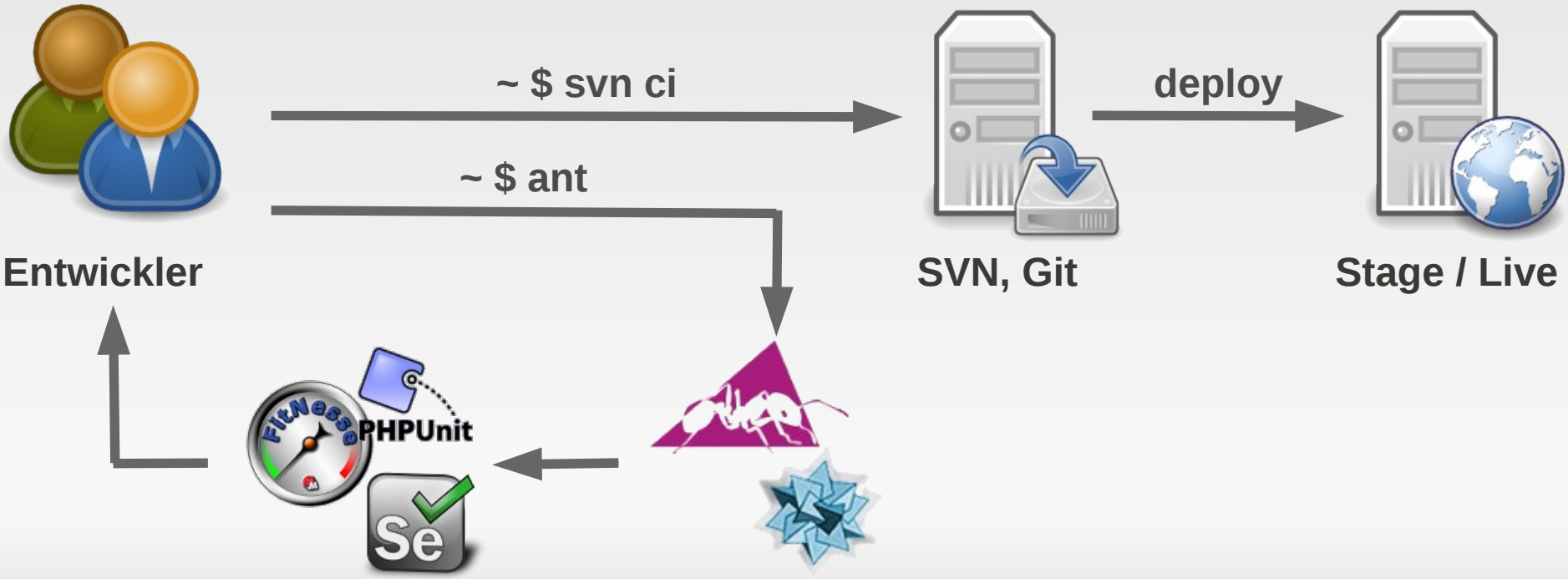
Grundvoraussetzungen

- Versionsmanagement
- Automatisierte Tests

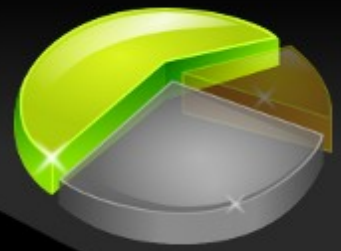


Grundvoraussetzungen

- Versionsmanagement
- Automatisierte Tests
- Buildmanagement

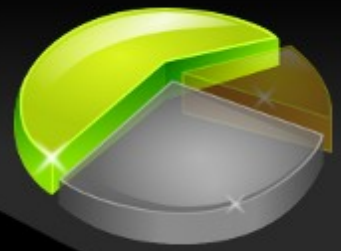


Zwischenfazit



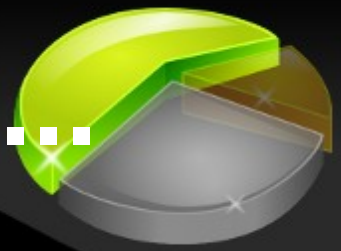
- 25% - Frühzeitige Erkennung von Fehlern
  - Automatisierte Tests
- 50% - Minimierung manueller Arbeitsschritte
  - Buildautomatisierung
- 25% - Lauffähige Softwareversion zu jeder Zeit
  - Versionsverwaltung
- 25% - Transparenz des Entwicklungsprozesses
  - Commithistorie
- 31% - Zufriedenheit mit dem eigenen Produkt

Agenda



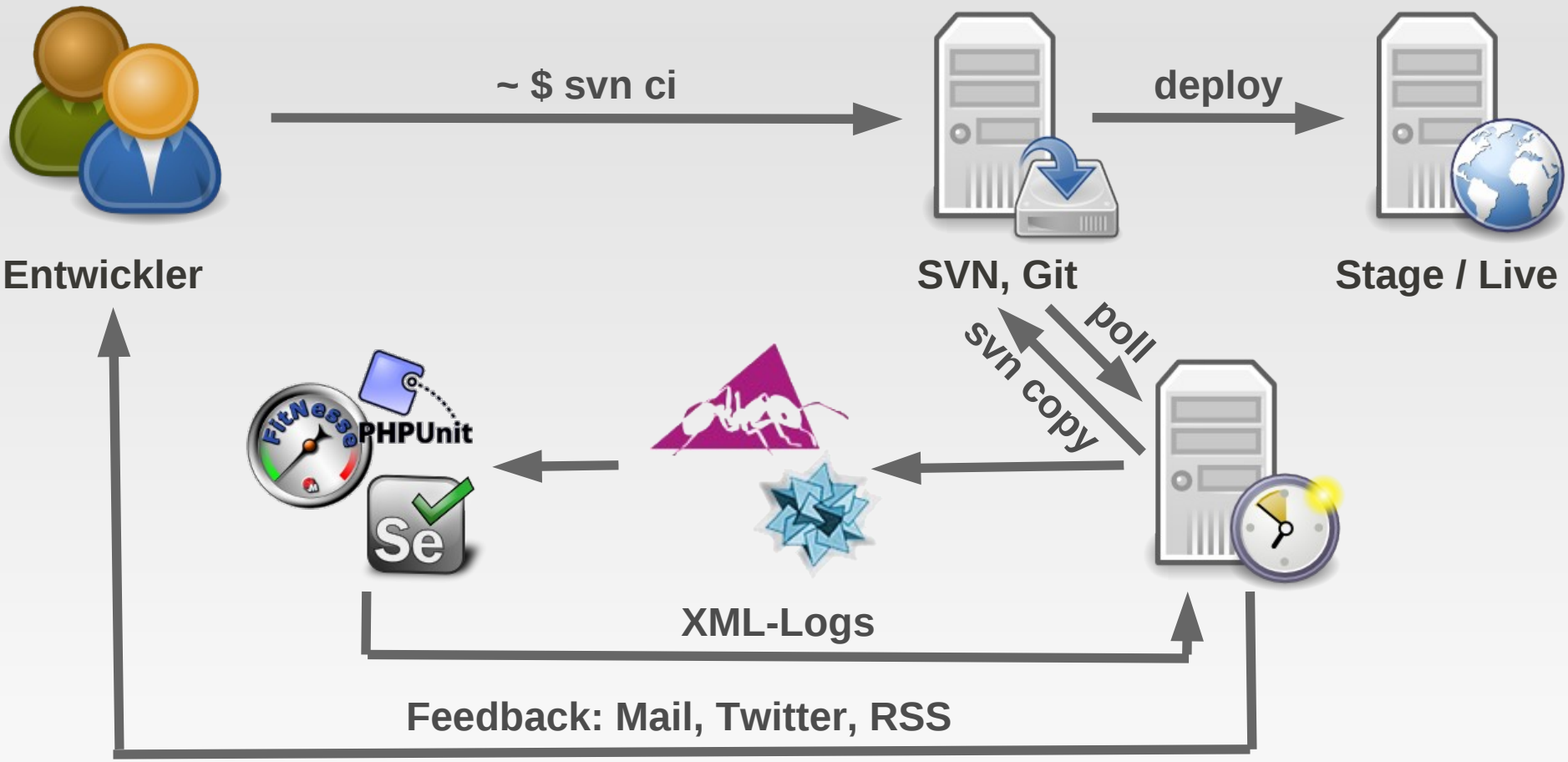
- Klassische Integration
- Grundlagen für kontinuierliche Integration
- Kontinuierliche Integration
- Prozesse und Techniken im Detail
- Continuous Deployment

Kontinuierliche Integration ist...

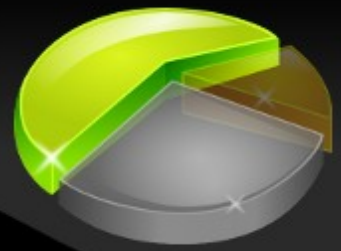


- ...nicht nur ein Werkzeug, sondern hat Einfluss auf den gesamten Entwicklungsprozess
  - Jede Änderung wird zeitnah eingecheckt
  - Die Versionsverwaltung ist kein Zwischenspeicher für fehlerhaften Quelltext
  - Fehlgeschlagene Builds werden umgehend gefixt, denn oberstes Ziel ist eine lauffähige Software
  - Mittels Buildskript führt jeder Entwickler vor einem Commit eine lokale Integration durch

Etappenziel erreicht

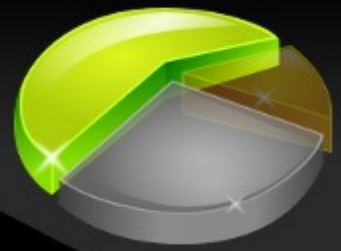


Zwischenfazit



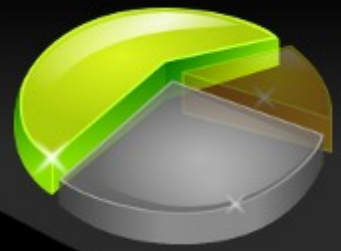
- 50% - Frühzeitige Erkennung von Fehlern
  - Automatisierte Tests, Testlauf nach jedem Commit
- 50% - Minimierung manueller Arbeitsschritte
  - Buildautomatisierung
- 50% - Lauffähige Softwareversion zu jeder Zeit
  - Versionsverwaltung, CI-Version
- 50% - Transparenz des Entwicklungsprozesses
  - Commithistorie, CI-Feedback
- 50% - Zufriedenheit mit dem eigenen Produkt

Agenda



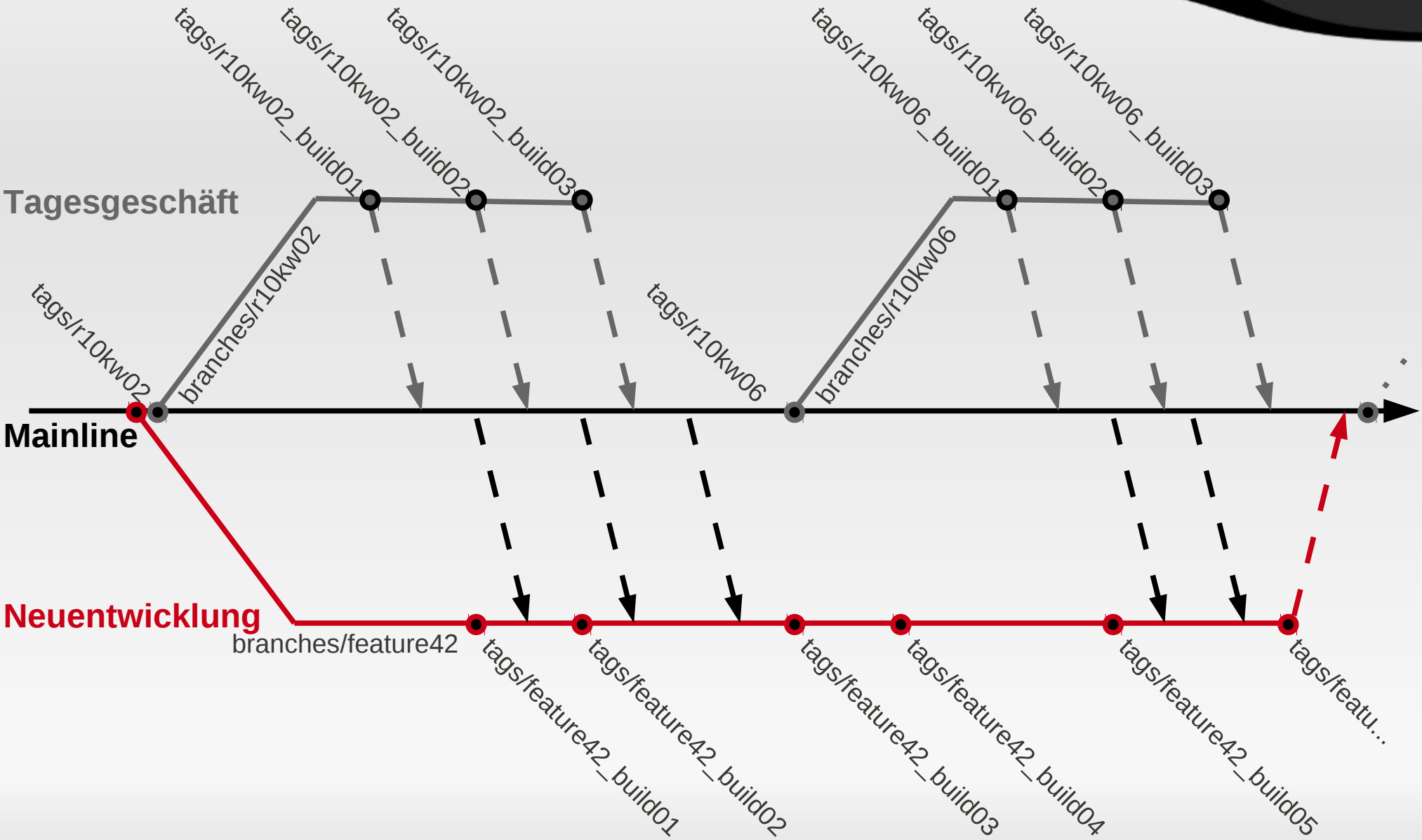
- Klassische Integration
- Grundlagen für kontinuierliche Integration
- Kontinuierliche Integration
- Prozesse und Techniken im Detail
- Continuous Deployment

Versionsmanagement



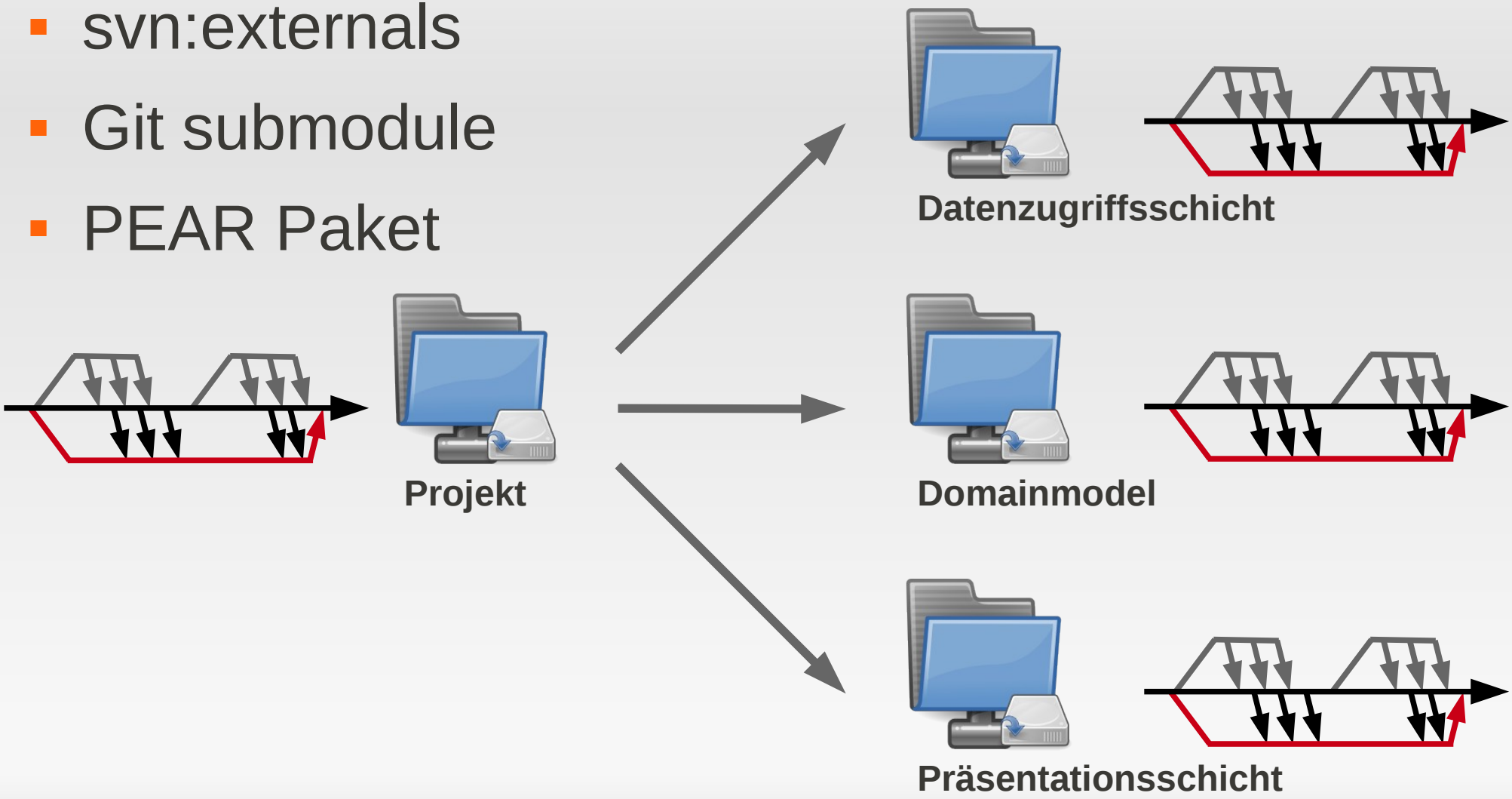
- Artefakte die versioniert werden sollten
  - Quelltexte jeglicher Art
  - Konfigurationsdateien
- Branches zur Trennung von Produktlinien
  - Releases und Features bekommen einen Branch
- Tags
  - Releases und erfolgreiche Builds werden mit einem Tag versehen

Tagging & Branching



Komponenten bilden

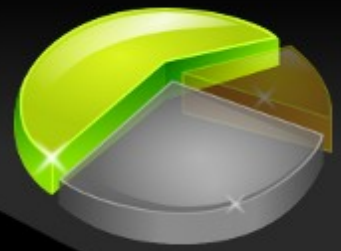
- svn:externals
- Git submodule
- PEAR Paket



Vorteile der Komponentenbildung

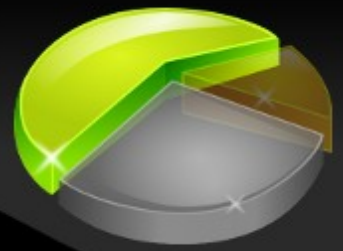
- Komponenten sind getrennt von einander testbar
- Kürzere Zeitenspanne bis zum Feedback
- Es existieren immer stabile Versionen von Komponenten und der gesamten Software
- Definierte und gut sichtbare Abhängigkeiten

Testverfahren



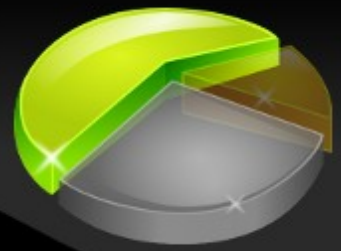
- Software- & Akzeptanztests mit
  - PHPUnit, Selenium, Fitnesse
- Statische Tests
  - Lint, PHPCPD, PHP_CodeSniffer, PHPMD
- Last- und Performancetests
  - JMeter

Akzeptanztest...



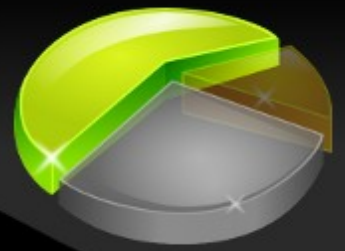
Wer kennt Fitnesse?

Statische Tests



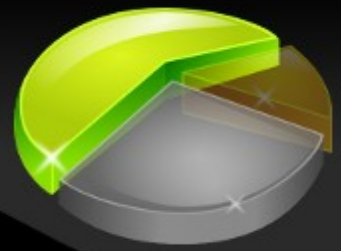
- Gerade bei modernen, browserbasierten Anwendungen ist syntaktische Korrektheit
  - Die einfachste Form der statischen Codeanalyse ist das lintern:
    - php -l
    - W3C CSS Validator
    - xmllint –html –noout http://www.example.com/
    - Douglas Crockfords JSLint

Programmierkonventionen



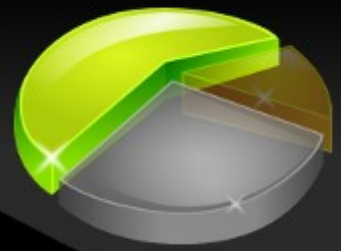
- Ein häufig unterschätzter Qualitätsaspekt
- Wo hinterlässt man eher Unrat?
  - Im keimfreien Operationssaal
  - Neben einer überquellenden Mülltonne
- Reduziert die Einarbeitungszeit in Quelltext
- PHP_CodeSniffer ist hier der Defacto-Standard

Softwaremetriken



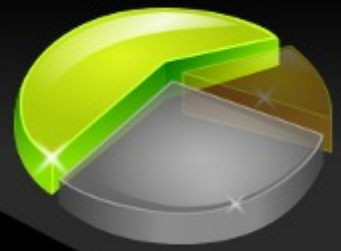
- Eine Softwaremetrik ist eine Maßzahl für Qualitätsmerkmale von Software
- Mathematische Funktion zur Ermittlung von Kennzahlen
- Softwaremetriken ermöglichen die Kontrolle der qualitativen Entwicklung von Software

Buildmanagement



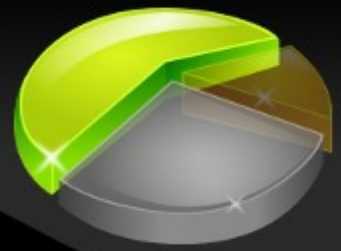
- Datenbanken
  - Anlegen/Löschen von Schemata und Testdaten
  - Automatisierung von Schema-Migrationen
  - Für jede Änderung existiert der passende Rollback
    - Doctrine Migrate, DbDeploy
- Automatisierung aller Schritte zur Erzeugung einer frei wählbaren Softwarekonfiguration
  - One Click Install

Zwischenfazit



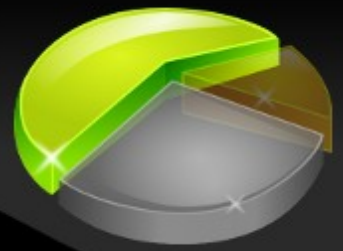
- 75% - Frühzeitige Erkennung von Fehlern
  - Automatisierte Tests, Testlauf nach jedem Commit
- 75% - Minimierung manueller Arbeitsschritte
  - Buildautomatisierung, One Click Install
- 75% - Lauffähige Softwareversion zu jeder Zeit
  - Versionsverwaltung, CI-Version, Tags
- 75% - Transparenz des Entwicklungsprozesses
  - Commithistorie, CI-Feedback
- 75% - Zufriedenheit mit dem eigenen Produkt

Agenda



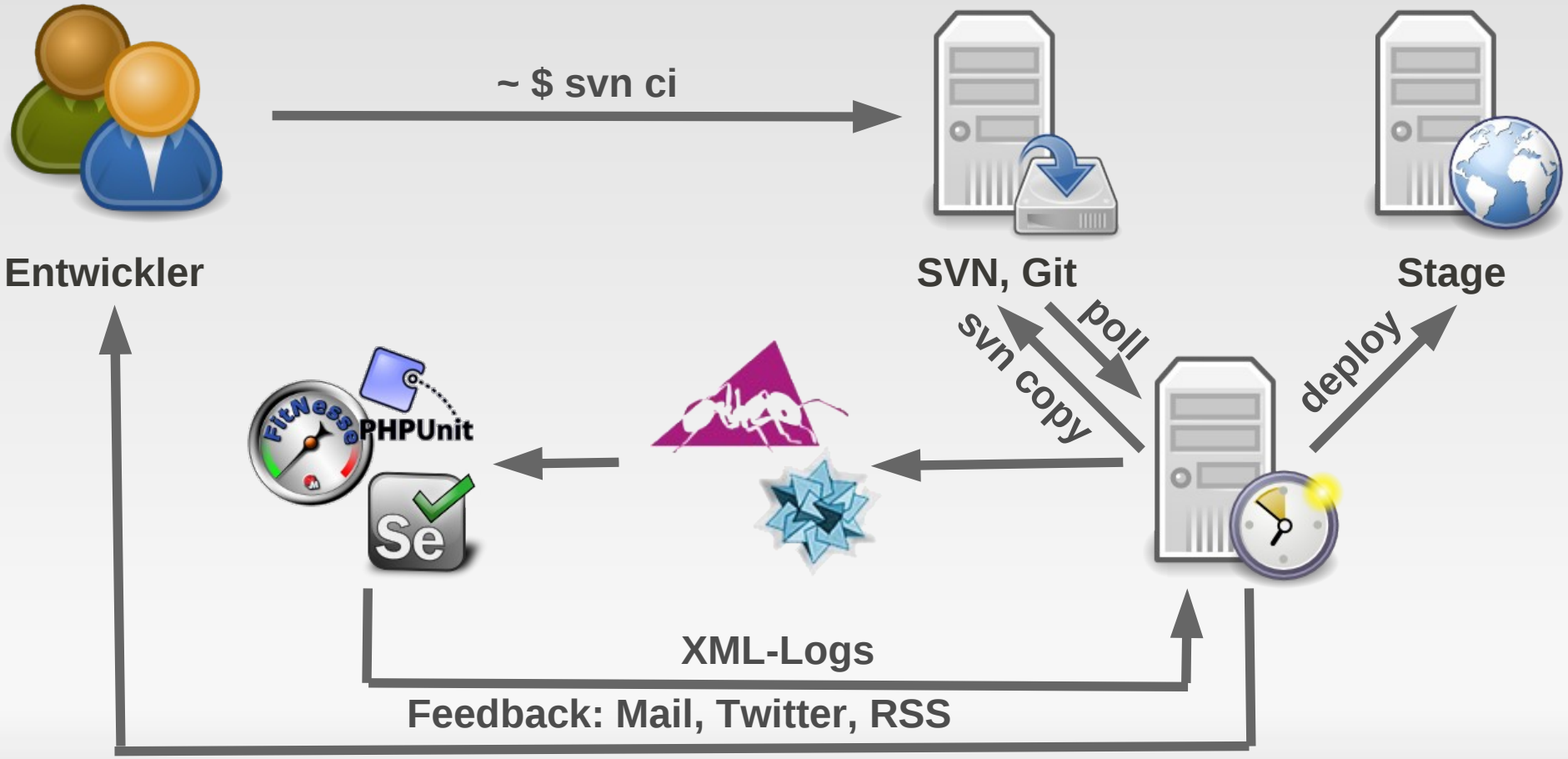
- Klassische Integration
- Grundlagen für kontinuierliche Integration
- Kontinuierliche Integration
- Prozesse und Techniken im Detail
- Continuous Deployment

Continuous Deployment

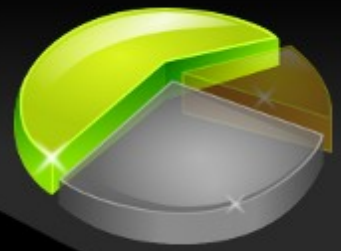


- Was ist zu beachten?
  - CI-, Stage- und Live-System müssen identisch sein
  - Klare Abgrenzung zwischen Tagesgeschäft und Neuentwicklungen
  - Deployments erfolgen nur aus einem durch die QA abgenommenen Branch
  - Deployt wird immer auf eine Stage-Umgebung, wo die letzte Abnahme erfolgt

Continuous Deployment



Danke



?

Contact:

Manuel Pichler
http://manuel-pichler.de
mapi@pdepend.org
@manuelp

Kommentare, Anregungen & Bewertung:

http://joind.in/1719