Orientation in Objects

Von Continuous Integration zu Continuous Delivery

international PHP2013 conference

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Steffen Schluff

Version: 1.1

Gliederung

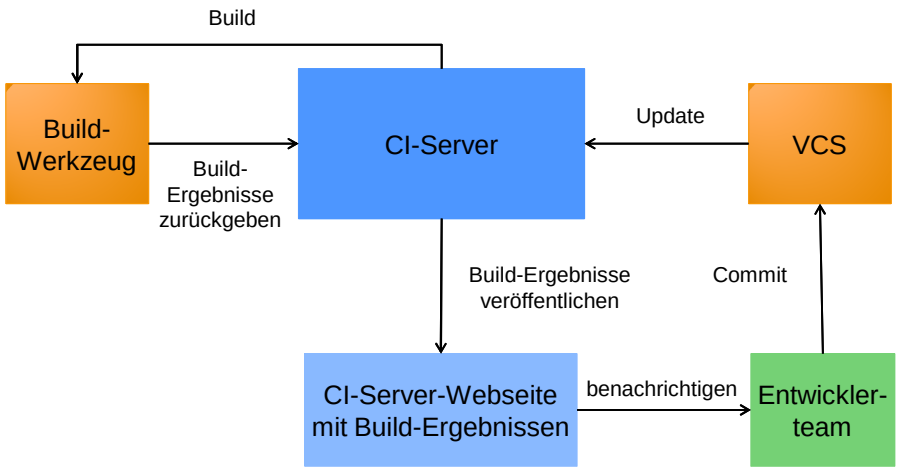
Orientation in Objects

- Einleitung
- Continuous Delivery
- DevOps
- Zusammenfassung

Gliederung

- **Einleitung**
- Continuous Delivery
- DevOps
- Zusammenfassung

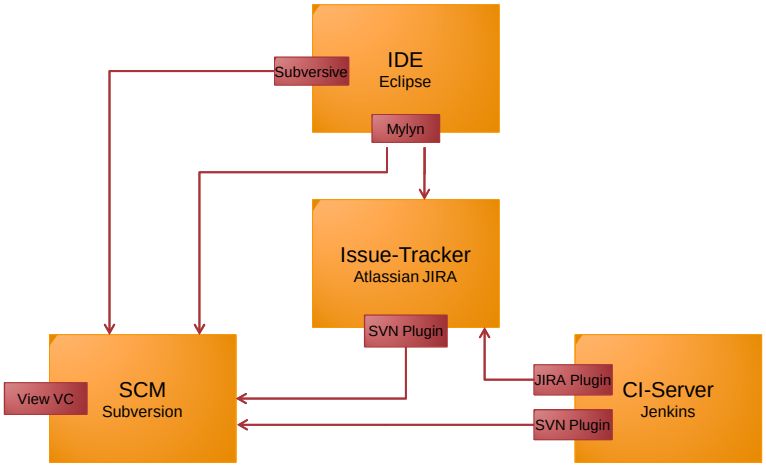
Been there, done that (1)



Been there, done that (2)

- „Daily Build and Smoke Tests“ sind schon ein alter Hut
  - Erste Veröffentlichung von Steve McConnell im Jahre 1996
  - Thema war bereits davor schon bekannt
- „Continuous Integration“ Artikel von Martin Fowler im Jahre 2000
  - Themenbereich bekam einen klingendem Namen
  - Definition „Key Practices“ (Automate the build, Make it self-testing, …)
  - Erste Bereitstellung von „fertigen“ Tools (CruiseControl)
- „Continuous Integration“ gehört heute zum guten Ton
  - Wahlfreiheit zwischen diversen Servern (Jenkins, Hudson, Bamboo, …)
  - Definition von Best Practices, Patterns und Anti-Patterns
  - Probleme der zweiten Generation: Testlaufzeiten, Virtualisierung, …

Entwickler Kosmos



Da war doch noch was? (1)

Orientation in Objects

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

First principle behind the Agile Manifesto
(http://agilemanifesto.org/principles.html)

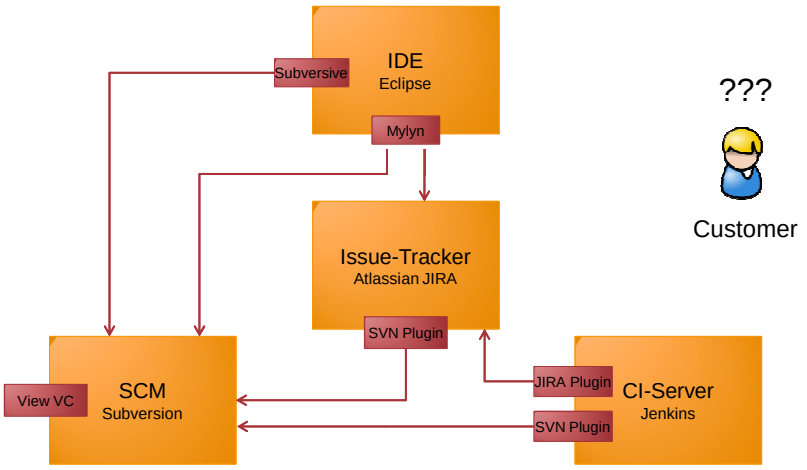
Da war doch noch was? (2)

Orientation in Objects

Our highest priority is to ***satisfy the customer*** *through early and continuous delivery of valuable software.*

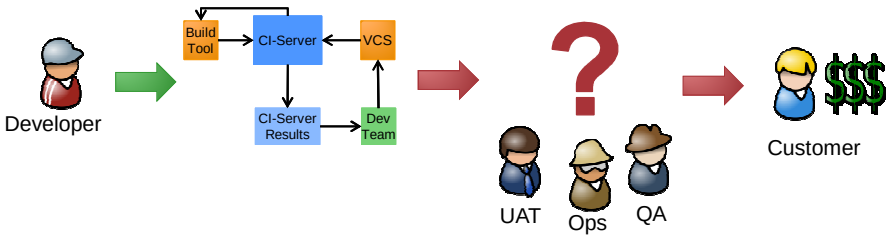
First principle behind the Agile Manifesto
(http://agilemanifesto.org/principles.html)

Ha Ha Only Serious



Bis hierher sollst du kommen und nicht weiter

- Continuous Integration konzentriert sich auf Entwickler
- CI Server grün, Commit erfolgreich, Entwickler glücklich
- Aber kein Bereitstellen zum Testen und keine Produktivsetzung



(http://www.public-domain-photos.com/free-cliparts)

Denkt denn niemand an den Kunden?

Orientation in Objects

- Kunde möchte Verfügbarkeit seiner Funktionalität
  - Kein Interesse, ob CI Server rot, grün, gelb oder blau ist
- Zwischen gutem CI Build und Kunden Verfügbarkeit liegt Release
  - Release Schritt oft nicht so gut beherrscht wie CI Ökosystem
- Release Modell „Big Bang“™ (alias der Klassiker)
  - Manuell, Zeitintensiv, Kompliziert, viele Beteiligte, Fehleranfällig

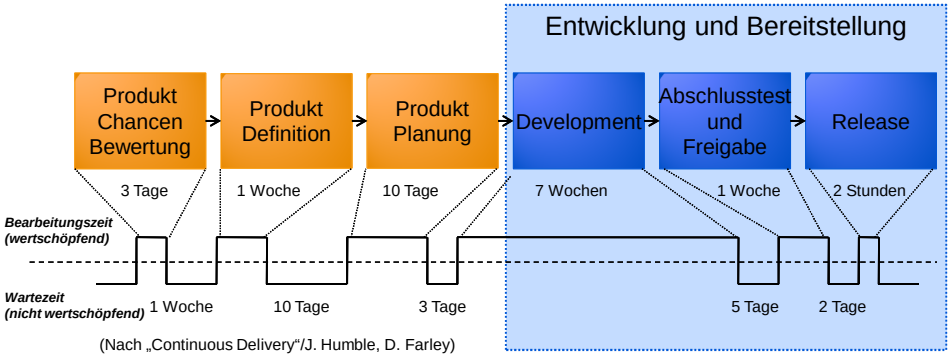
Don‘t do that then!

Orientation in Objects

- Seltene dafür aber große Releases als Konsequenz
  - Organisatorische Vermeidungsreaktion
- Zugleich hoher Stressfaktor bei ungeplanten Releases
  - Wenig Übung führt zu hohen Fehlerrate bei Hotfixes
- Frustrierend für den Kunden
  - Auch einzelne Features brauchen scheinbar sehr lange
- Wir wollen den Kunden nicht frusten, hier muss sich etwas ändern!

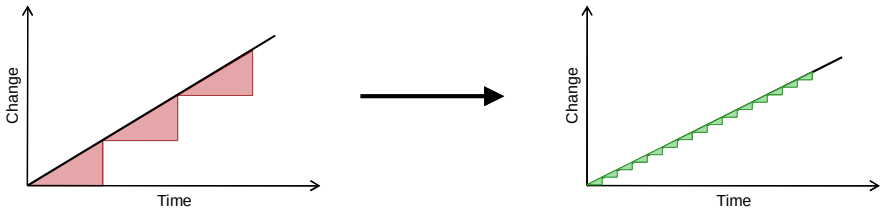
Gut Ding will Weile haben

- Dauer einer Kunden Idee bis Produktivsetzung („Concept to Cash“)
- Visualisierung als „Value Stream Map“



(Nach „Continuous Delivery“/J. Humble, D. Farley)

Stay on target



“[…] *the ability to rapidly, reliably and repeatedly push out enhancements and bug fixes to customers at low risk and with minimal manual overhead.*”

(http://en.wikipedia.org/wiki/Continuous_delivery)

Buzzword (1)

Orientation in Objects

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

First principle behind the Agile Manifesto
(http://agilemanifesto.org/principles.html)

Buzzword (2)

Orientation in Objects

Our highest priority is to satisfy the customer through early and ***continuous delivery*** *of valuable software.*

First principle behind the Agile Manifesto
(http://agilemanifesto.org/principles.html)

Gliederung

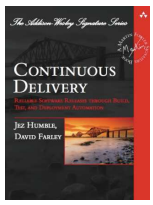
Orientation in Objects

- Einleitung
- **Continuous Delivery**
- DevOps
- Zusammenfassung

Continuous Delivery

Orientation in Objects

- *„Continuous Delivery is not Continuous Integration. Continuous Delivery is being in the position to ship your product whenever you want, day or night." (Neal Ford)*
- Frühere Begriffsverwendung und Wurzeln
  - „Agile Manifesto" (2001)
  - „Deployment Pipeline" (2004/2005)
- Gleichnamiges Buch von Jez Humble & David Farley
  - Eigentliche Begriffsprägung (2010)
- Schwerpunktthemen „Automation" und „Collaboration"



Continuous Delivery – Kerngedanken

Orientation in Objects

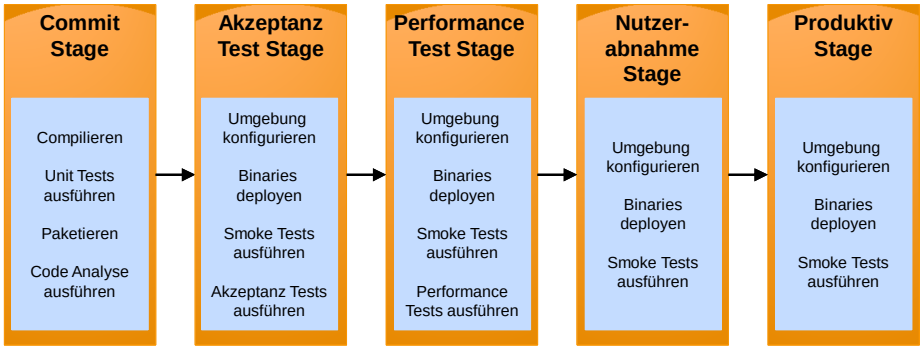
- “Create a Repeatable, Reliable Process for Releasing Software”
  - “If It Hurts, Do It More Frequently, and Bring the Pain Forward”
  - “Automate Almost Everything”
  - “Keep Everything in Version Control”
- “Everybody Is Responsible for the Delivery Process”
- “Done Means Released”
- Und wie soll das alles umgesetzt werden?

(Nach „Continuous Delivery“/J. Humble, D. Farley)

Deployment Pipeline – Ein erster Blick

Orientation in Objects

- Zentrale Abstraktion „Deployment Pipeline“
  - Visualisierung aller Prozessteile für alle Beteiligten
  - Verbessertes Feedback während der Ausführung
  - Möglichkeit eines vollautomatischen Releases in alle Umgebungen



(Nach „Continuous Delivery“/J. Humble, D. Farley)

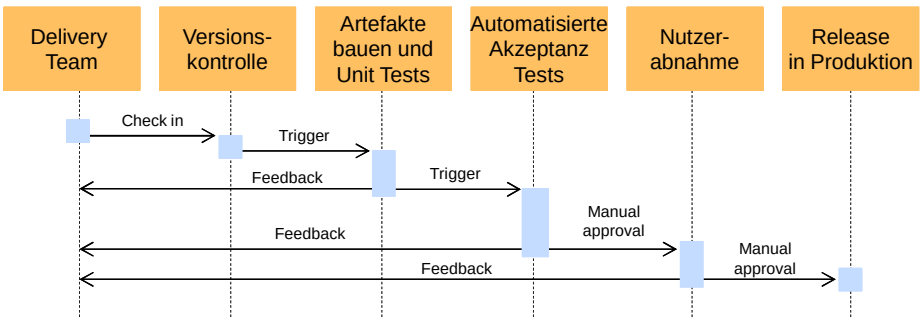
Deployment Pipeline – Bestandteile

Orientation in Objects

- Die **Deployment Pipeline**
  - Macht Status der Produktentwicklung sichtbar
  - Liefert Feedback zu jeder Änderung
  - Technisch-konzeptuelle Basis des Release Prozesses
- Die Pipeline besteht aus einer Folge von **Stages**
  - Commit Stage als zentrales Eingangs-Gate
  - Typische Stages: UAT, Performance Tests, Production Deployment
  - Stages verbunden durch Trigger (automatisch oder manuell)
- **Jobs** sind die Bausteine der Stages
  - „Unit of Work“
  - Bestehen aus Tasks wie Build, Deploy, Copy, Test, …

Deployment Pipeline – Sequenzdiagramm

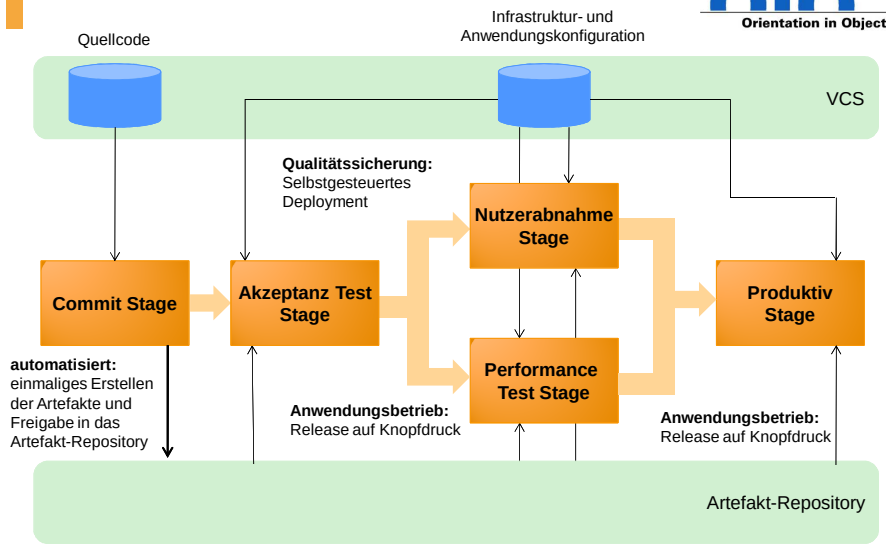
Orientation in Objects



- Jede Ressourcen Änderung startet neue Pipeline Instanz
- Erste Stage produziert alle Artefakte
- Durchlaufen aller Stages bis Fehlschlag („Stop the line“) oder …
- Pipeline Ende erreicht ist (letzte Stage führt Deployment aus)

(Nach „Continuous Delivery“/J. Humble, D. Farley)

Deployment Pipeline – Ein zweiter Blick



(Nach „Continuous Delivery"/J. Humble, D. Farley)

Continuous Delivery – Prinzipien & Methoden (1)

- Fortlaufende Optimierung
  - In Verantwortung aller Beteiligter (Development, Operations, …)
- Artefakte
  - Werden **einmal** gebaut und in einem Artefakt Repository verwaltet…
  - und allen Stages zur Verfügung gestellt
  - Ziel ist identisches Deployment in allen Umgebungen
  - Umgebungsspezifika durch eigene Konfigurationen
- Configuration-Management
  - Basis für einmalig erstellte Artefakte
  - Umfasst Software und Infrastruktur („Infrastructure as code")
  - Konfigurationen werden versioniert

Continuous Delivery – Prinzipien & Methoden (2)

Orientation in Objects

- Automatisierung
  - So umfangreich wie möglich
  - Umfasst auch alle Aspekte der Infrastruktur (inklusive OS)
  - Prägung durch Development und Operations
- Tests
  - Basis für Automatisierung und Pipeline Processing
  - Geben Sicherheit für erfolgreiche Änderungen
  - Smoke Tests speziell für Deployment
- Monitoring
  - Ermöglicht Feedback für Operating
  - Basis für fortlaufende Optimierung

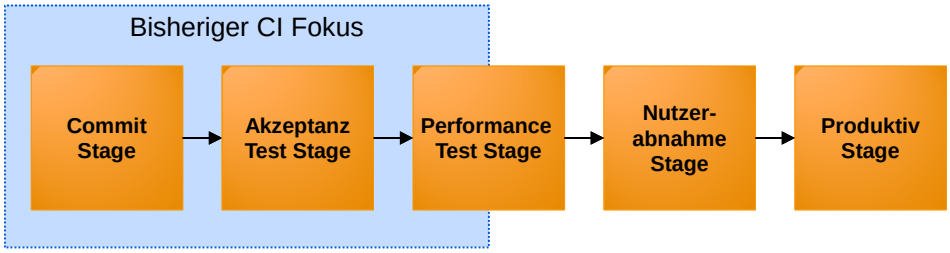
Alle Theorie ist grau

Orientation in Objects

- Erfolgsfaktoren von Continuous Integration rückblickend waren
  - Eingängiger Name
  - Konkrete „Key Practices“
  - Einsetzbare Tools
- Zum Erfolg fehlt Continuous Delivery also noch ein gutes Tool
  - Erster Impuls oft selbstgemachte Lösungen („Home grown“), aber …
  - häufig schnell veraltet bei schlechtem Kosten-Nutzen Verhältnis
- Jedes Projekt hat in der Praxis seine eigenen Spezialitäten
  - Web App vs. Mobile vs. Rich Client, Programmiersprache, OS, usw.
  - Somit sind auch Continuous Delivery Umsetzungen verschieden
- Gibt es also gar kein Continuous Delivery Tool?

Continuous Delivery – Tooling (1)

- *„The deployment pipeline has its foundation in the process of continuous integration and is in essence the principles of continuous integration taken to its logical conclusion.“ (J. Humble, D. Farley)*



Continuous Delivery – Tooling (2)

- CI Server werden bereits für alle möglichen Projekt Arten eingesetzt
  - Und integrieren dabei diverse Tool Arten (Build, Test, Lint, Coverage, …)
- Tools für einzelnen Continuous Delivery Konzepte sind vorhanden
  - Artefakt Repositories (CI-Server eigene Repos, Maven, …)
  - „Infrastructure as code“ (Puppet, Chef, …)
- Continuous Integration wird Continuous Delivery Server durch …
  - Integration der neuen CD spezifischen Tool Arten
  - Bereitstellung einer Deployment Pipeline (samt Stages, Jobs, Triggern)

Namen sind Schall und Rauch

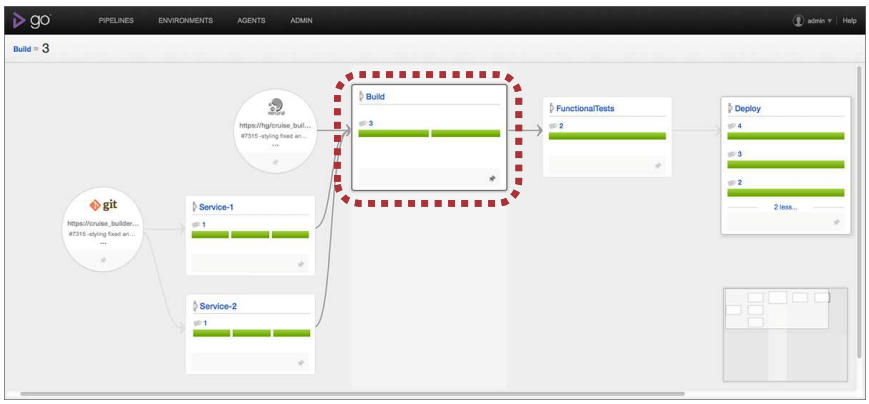
Orientation in Objects

- Jeder gängige CI Server bietet Pipeline Bausteine an
  - Und ist somit ein CD Server
- Konkrete Namen können variieren
  - Teils historische Gründe, teils Abgrenzung zur Konkurrenz
- Exemplarische Beispiele („*Your Mileage May Vary*“)
  - **Jenkins**: *Build Jobs, Build Steps, Post-build Actions, diverse Plugins*
  - **Go**: *Go Pipelines, Stages, Jobs, Tasks*
  - **Bamboo**: *Build Plans, Stages, Jobs, Tasks*
- Erster logischer Schritt ist Visualisierung der Pipeline
  - Bisherige Aktivitäten einer Pipeline, i.e. vergangene Pipeline Instanzen

Build Pipeline Plugin in Jenkins

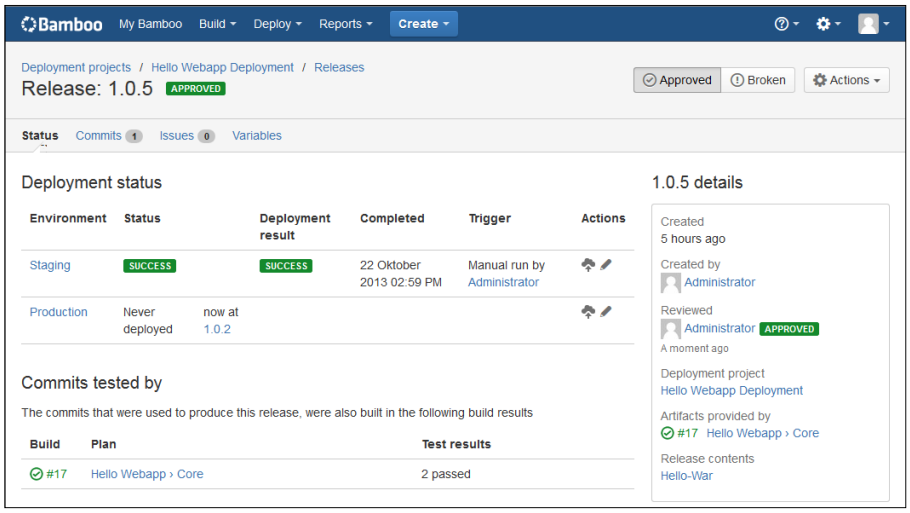
Orientation in Objects

Value Stream Maps in Go



(http://www.thoughtworks.com/products/docs/go/13.3/help/)

Release Details in Bamboo



Gliederung

Orientation in Objects

- Einleitung
- Continuous Delivery
- **DevOps**
- Zusammenfassung

Die Tücke im Detail

Orientation in Objects

- Continuous Delivery ist Fortsetzung von Continuous Integration
  - Continuous Integration Tooling ist bekannt und etabliert
- Aber: Jetzt nicht mehr nur Development beteiligt, sondern auch…
  - Quality Assurance in Stage „Manual Testing“
  - Operations in Stage „Release“
- Problem wird „verschärft“ durch „neue“ Methoden und Technologien
  - Agile Projekte releasen häufiger als Wasserfall Projekte
  - Verfügbarkeit von Server Hardware stetig wachsend (Cloud)
  - Bisheriger Release Stress und Sockelkosten nicht mehr tragbar
- Beteiligte Gruppen müssen stärker zusammenrücken
  - Betrifft vor allem Development und Operations

First things first

Orientation in Objects

- Neues Buzzword notwendig: *DevOps*
- Portmanteau (Kofferwort) aus „Development“ und „Operations“
- Erste Verwendung „DevOpsDays 2009“ in Belgien

DevOps […] is a software development method that stresses communication, collaboration and integration between software developers and information technology (IT) professionals.

(http://en.wikipedia.org/wiki/Devops)

Houston, we have a problem (1)

Orientation in Objects

- Zwei beteiligte Parteien mit gegenläufigen Zielen
  - Development will neue Features (Change)
  - Operations will hohe und schnelle Verfügbarkeit (Stability)
  - Konkurrenz, da häufig keine übergreifende Sicht in Unternehmen
- Trennung durch „Wall of Confusion“
  - Unterschiedliche Ziele und unterschiedliche Tools
  - „Wir werfen Operations dann den nächsten Release über den Zaun“

(Quelle: http://dev2ops.org/2010/02/what-is-devops/)

Houston, we have a problem (2)

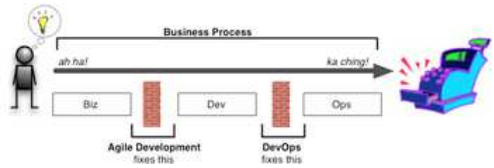
Orientation in Objects

- Nach fehlgeschlagenen Relase spielen alle „The Blame Game“
  - *Ops*: Artefakte, Skripte, Config Files, … waren fehlerhaft
  - *Dev*: Bei uns hat es funktioniert, ihr habt was falsch gemacht
  - *Ops*: Ihr müsst selber drauf schauen, was nicht stimmt
  - *Dev*: Wir kommen doch gar nicht auf die Prod Maschinen (usw.)
- In der Zwischenzeit kann niemand arbeiten
  - Fachabteilung ist es egal, ob Development oder Operations schuld
- Rollback eines teilweise durchgeführten Releases oft unmöglich
  - Releases nicht atomar, häufig im Bereich Datenbankänderungen
- Releases wird als Konsequenz „irgendwie“ lauffähig gemacht
  - Oft „Dirty Hacks“ ohne Lerneffekt dafür mit versteckten Fehlern

Déjà vu

Orientation in Objects

- „Wall of Confusion“ ist nicht neu
- Frühere Trennung zwischen Business und Development
- Lösung war agile Entwicklung
- DevOps soll gleiches für Development und Operations leisten



(Quelle: http://dev2ops.org/2010/02/what-is-devops/)

Hund, Katze, Maus, …

Orientation in Objects

- Zur Zeit noch kein DevOps de facto Standard verfügbar
  - … auf dem Level von Fowlers CI Artikel oder dem CD Buch
- Selbst die Definition variiert noch je nach Quelle
  - „Devlopment Method“ (Wikipedia), „Movement“ (Gartner), …
- Mögliches Standardwerk „The DevOps Cookbook“ angekündigt
  - Vertrauensvorschuss durch Autoren (Patrick Debois, Gene Kim, …)
  - Vermutlich weitreichender als nur DevOps (eher *bus-qa-sec-net-ops*)
- Zur Zeit hauptsächlich Online Material und Quellen für DevOps
  - Inhalte stark abhängig von jeweiligen Autoren

Patterns again, …

Orientation in Objects

- „Agile DevOps“ Reihe von Paul Duvall als exemplarisches Beispiel
  - Online Liste von DevOps Pattern bei IBM DeveloperWorks
  - Starker Fokus auf „Infrastructure as code“ Themen
- Scripted environments
  - Automatisierte und versionierte Bereitstellung einer Server Umgebung
  - Server Umgebung ist Teil des Deployments auf Produktion
  - Mögliche Tools: Chef oder Puppet
- Test-driven infrastructures
  - Wenn Infrastruktur Code ist, dann muss Infrastruktur getestet werden
  - Testet ob alle Bestandteile der Infrastruktur verfügbar sind
  - Beispiel: Test, ob Apache in richtiger Version läuft

… again …

Orientation in Objects

- Chaos Monkey
  - *„Everything fails, all the time“ (Werner Vogels)*
  - Terminiert regelmäßig Instanzen in einer Gruppe von Systemen und …
  - Testet indirekt, ob das System trotzdem weiterläuft
  - Schlägt in kontrollierten Zeiten zu, um für den Ernstfall bereit zu sein
- Transient environments
  - Umgebungen sind so kurzlebig wie möglich (Stunden bis Tage)
  - Keine „heiligen“ Server mehr, mit nicht reproduzierbarer Konfiguration
  - Forciert Konzept, dass alles automatisiert sein muss

… and again.

Orientation in Objects

- Version everything
  - Ziel ist Etablierung einer „single source of truth“
  - Maximal ein Checkout zum Loslegen für einen neuen Entwickler
- Delivery Pipeline
  - Vergleiche Deployment Pipeline aus Continuous Delivery
- DevOps Dashboard
  - Anzeige wie Änderungen das System in welcher Stage wie beeinflussen
  - Für alle beteiligten Gruppen zugänglich
  - Häufig im Continuous Integration Server verankert

Frage des Standpunkts

Orientation in Objects

- Continuous Delivery und DevOps sind miteinander verwandt
  - Ähnliche oder zumindest überlappende Prinzipien und Kernbegriffe
- Keine gegenseitige Ignoranz sondern vielmehr gleiches Ziel
  - Veröffentlichungen beziehen sich auf den jeweils anderen Begriff
- „Flattening“ von Release Prozessen und Organisationsstrukturen
  - Wunsch sind schnellere, billigere und bessere Releases

Gliederung

Orientation in Objects

- Einleitung
- Continuous Delivery
- DevOps
- **Zusammenfassung**

Zusammenfassung

Orientation in Objects

- Kundenzufriedenheit erfordert Auslieferung von Software
  - Klassische Release Modelle passen nicht zu moderner Entwicklung
- Continuous Delivery ist logische Folge von Continuous Integration
  - CI Server unterstützen jetzt auch Deployment Pipelines
- Auslieferung erfordert Team übergreifende Zusammenarbeit
  - Vor allem zwischen Development und Operations
- Zwischen Development und Operations steht „Wall of Confusion“
  - DevOps will diese Wand mit Methoden und Verfahren überwinden

If you remember one thing

Orientation in Objects

“In order for you to keep up with customer demand, you need to create a deployment pipeline. You need to get everything in version control. You need to automate the entire environment creation process. You need a deployment pipeline where you can create test and production environments, and then deploy code into them, entirely on demand.”

(“The Phoenix Project”)

Links

Orientation in Objects

- Schulung: Versionsverwaltung mit Git
  - http://www.oio.de/git-schulung-versionsverwaltung-seminar-dvcs-training.htm
- Schulung: Versionsverwaltung mit Subversion
  - http://www.oio.de/subversion-svn-schulung.htm
- Schulung: Hudson Grundlagen
  - http://www.oio.de/schulung-hudson-seminar-continuous-integration-training-jenkins.htm
- Schulung: JIRA – Fachliche Administration
  - http://www.oio.de/seminar/methodik-prozess-management-soft-skills/seminar-training-atlassian-jira-schulung.htm

Links

Orientation in Objects

- Artikel aus dem Java Spektrum:
  Optimiertes Testen (PDF)
  - http://www.oio.de/public/softwaretest/optimiertes-testen-gateways-gatekeeper-keymaster_JS_03_11.pdf
- Beratung zu Open Source Tools:
  - http://www.oio.de/beratung-consulting/open-source-software/tools/

Orientation in Objects

Fragen ?

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Orientation in Objects

Vielen Dank für ihre Aufmerksamkeit !

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de