



Martin Heider, infomar software

Im Fluss bleiben

Continuous Integration mit Hudson

# Agenda

## Kontinuierliche Integration

- Ziel
- Grundprinzipien
- Adressierte Risiken
- Besser, schneller, günstiger
- Der Build und seine Inhalte
- Wie starten?
- Mit 7 Schritten in den CI Himmel



# Agenda

## Hudson

- Historie ...
- Funktionsweise
- Einrichtung
- Features & Plugins
- Gemachte Erfahrungen

## Ein kleiner Film zum Schluss



# Kontinuierliche Integration

## Ziel

**Wir wollen  
Software ... entwickeln**

Günstiger

Transparenter

Schneller

Zuverlässiger

Besser



# Kontinuierliche Integration

## Grundprinzipien

Integration  
ist aufwändig  
darum machen wir  
es jetzt öfter 😊

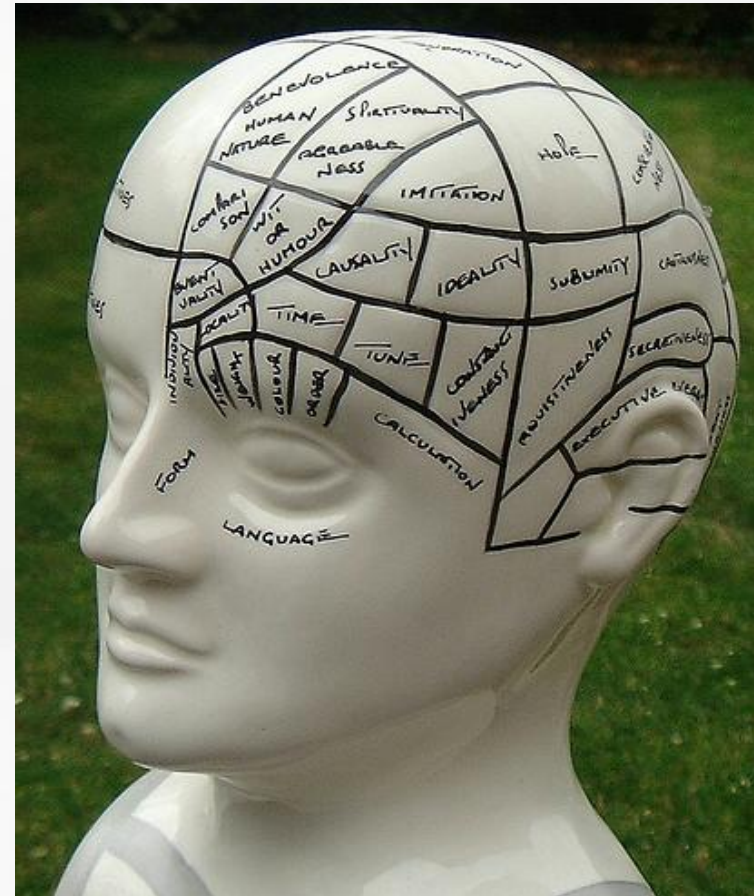


# Kontinuierliche Integration

## Grundprinzipien

### Entwicklungsmethode

- Mehr Mindset als Technik
- Unterstützt von Tools



# Kontinuierliche Integration

## Grundprinzipien

### Entwickler integrieren fortlaufend

- Kein Problemstau
- Im Fluss bleiben

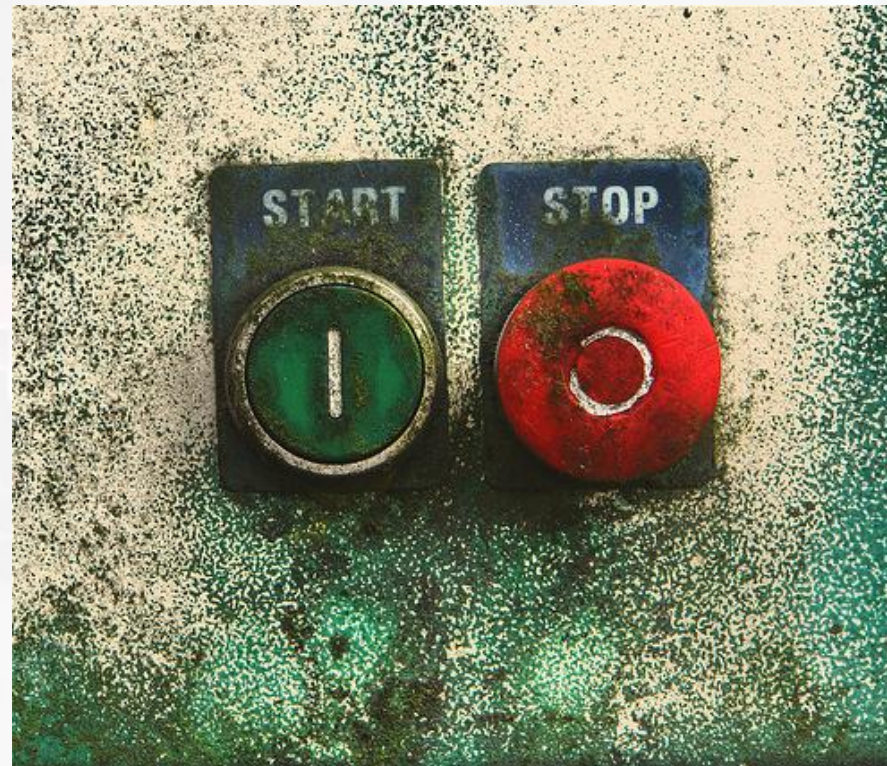


# Kontinuierliche Integration

## Grundprinzipien

### Vollautomatisierter Build

- „One green button“
- „Stop the line“





# Kontinuierliche Integration

## Grundprinzipien

### Was es nicht ist

- Nightly builds
- Entwickler branches
- Vereinbarte Integrationstermine
- Bauen mit der IDE



# Kontinuierliche Integration

## Adressierte Risiken

<b>Lesen Sie die Packungsbeilage!</b>	<b>Lesen Sie die Packungsbeilage!</b>	<b>Lesen Sie die Packungsbeilage!</b>
<b>NEIN</b>	<b>NEIN</b>	<b>NEIN</b>

zur bisherigen Integration

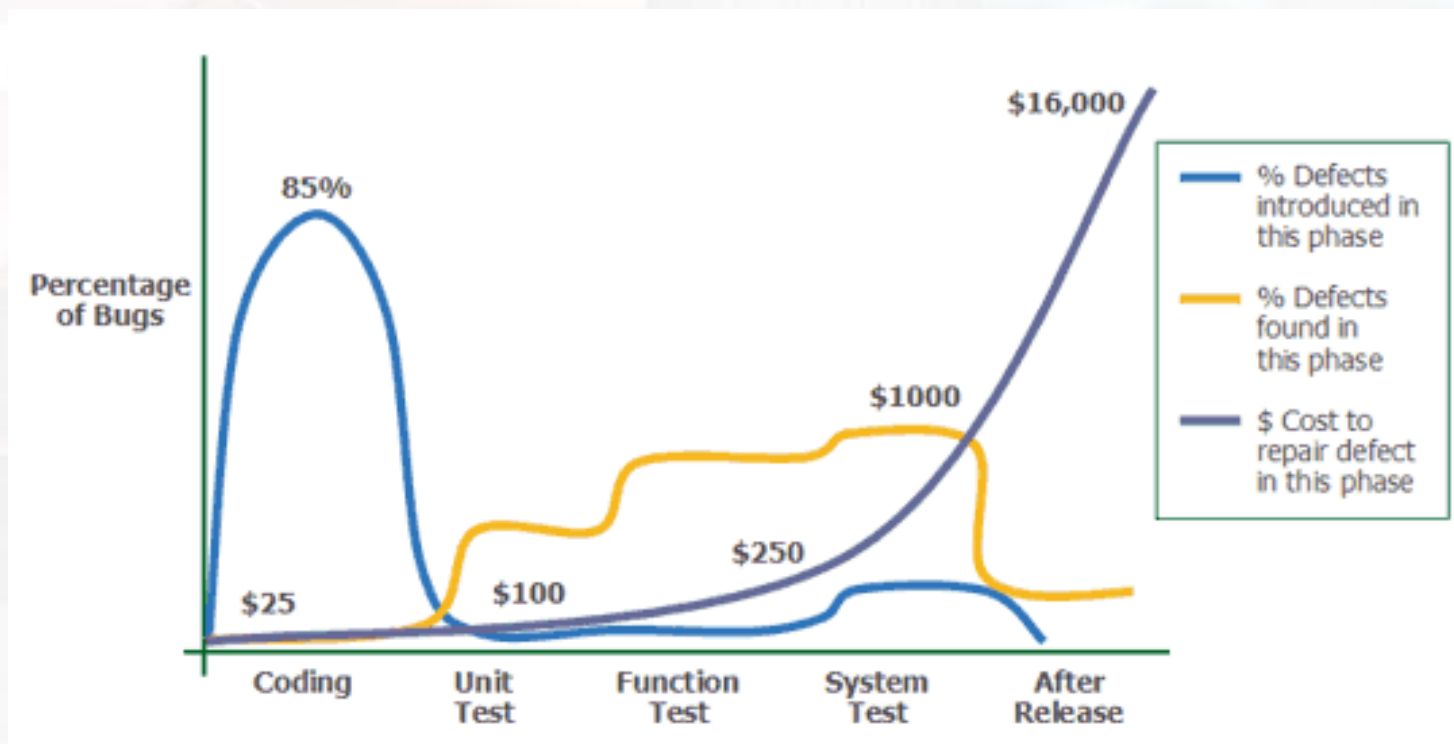
Zu Risiken fragen Sie Ihren Projektleiter oder Integrator



# Kontinuierliche Integration

## Addressierte Risiken

### Risiko I: Späte Fehlerbehebung ist teurer



[http://www.agitar.com/solutions/why\\_unit\\_testing.html](http://www.agitar.com/solutions/why_unit_testing.html)



# Kontinuierliche Integration

## Addressierte Risiken

### **Risiko II: Mangelnde Teamabstimmung**

- „Deine Änderung passen nicht mit meinen zusammen“
- „Hattest Du das nicht bereits vor 2 Monaten gefixt“

### **Risiko III: Schlechte Code-Qualität**

- „Wieso machen drei verschiedene Klassen das Gleiche?“
- „Der Code von Team XY schaut ja ganz anders aus“



# Kontinuierliche Integration

## Addressierte Risiken

### **Risiko IV: Mangelnde Transparenz / Sichtbarkeit**

- „Welche Tests laufen nicht?“
- „Was beinhaltet Build 1.2.3?“
- „Wo stehen wir mit der Code-Abdeckung?“

### **Risiko V: Nicht verfügbare Software**

- „Bei mir geht's“
- „Eigentlich läuft's“
- „Ich brauche noch einen Build zum Testen“
- „Morgen kommt der Chef-Chef, wir brauchen eine Demo“



Kontinuierliche Integration  
Besser, schneller, günstiger

Silver Bullet ??

Wohl kaum, aber ...



# Kontinuierliche Integration

## Besser, schneller, günstiger

### **Besser**

- Oft und frühzeitig getestet
- „Coding Standards“ und „Best Practises“ einhaltend

### **Schneller**

- Tests finden parallel zur Entwicklung statt
- Aufwändige Integrationen werden zum „Nicht-Ereignis“

### **Günstiger**

- Fehler werden früher gefunden
- Behebung der Fehler zum frühesten, günstigsten Zeitpunkt
- Einfach wiederholbare Tests



# Kontinuierliche Integration

## Der Build und seine Inhalte ...

### THE GRAND CHALLENGE EQUATIONS

$$\begin{aligned}
 & B_i A_i = E_i A_i + \rho_i \sum_j B_j A_j F_{ji} & \nabla \times \vec{E} = - \frac{\partial \vec{B}}{\partial t} & \vec{F} = m \vec{a} + \frac{dm}{dt} \vec{v} \\
 & dU = \left( \frac{\partial U}{\partial S} \right)_V dS + \left( \frac{\partial U}{\partial V} \right)_S dV & \nabla \cdot \vec{D} = \rho & Z = \sum_j g_j e^{-E_j/kT} \\
 & F_j = \sum_{k=0}^{N-1} f_k e^{2\pi i j k/N} & \nabla^2 u = \frac{\partial u}{\partial t} & \nabla \times \vec{H} = \frac{\partial \vec{D}}{\partial t} + \vec{J} \\
 & & p_{n+1} = r p_n (1 - p_n) & \nabla \cdot \vec{B} = 0 & P(t) = \frac{\sum_i W_i B_i(t) P_i}{\sum_i W_i B_i(t)} \\
 & - \frac{\hbar^2}{8\pi^2 m} \nabla^2 \Psi(r, t) + V \Psi(r, t) = - \frac{\hbar}{2\pi i} \frac{\partial \Psi(r, t)}{\partial t} & & -\nabla^2 u + \lambda u = f \\
 & \frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla) \vec{u} = - \frac{1}{\rho} \nabla p + \gamma \nabla^2 \vec{u} + \frac{1}{\rho} \vec{F} & \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = f
 \end{aligned}$$

- NEWTON'S EQUATIONS • SCHRÖDINGER EQUATION (TIME DEPENDENT) • NAVIER-STOKES EQUATION •
- POISSON EQUATION • HEAT EQUATION • HELMHOLTZ EQUATION • DISCRETE FOURIER TRANSFORM •
- MAXWELL'S EQUATIONS • PARTITION FUNCTION • POPULATION DYNAMICS •
- COMBINED 1ST AND 2ND LAWS OF THERMODYNAMICS • RADIOSITY • RATIONAL B-SPLINE •

**SAN DIEGO SUPERCOMPUTER CENTER**

*A National Laboratory for Computational Science and Engineering*

# Build != Kompilierung





# Kontinuierliche Integration

## Der Build und seine Inhalte ...

- Kompilierung
- Test Ausführung (Unit Tests, Akzeptanztests, etc.)
- Integration (Datenbank, Drittsysteme)
- Statische Analysen (Code & Architektur)
- Automatisches Deployment auf Staging / Produktiv Server
- Generierung der Dokumentation



# Kontinuierliche Integration Wie starten?



# Kontinuierliche Integration

## Wie starten?

### **Wann baue ich?**

- Nach jeder Code Änderung
- Nach jeder Änderung von Abhängigkeiten

### **Wie baue ich?**

- Mit einem einzigen Build-Skript
- Startbar auf der Kommandozeile
- Nicht in Abhängigkeit einer IDE

### **Was brauche ich dazu?**

- Konfigurationsmanagement Software
- CI Software und Server



# Kontinuierliche Integration Wie starten?

## Worauf ist noch zu achten?

- Schnelles Feedback
- Einfach zugreifbar
- Kein Aufwand für Entwickler
- Schlüsselmetriken identifizieren
- Wichtiges deutlich visualisieren
- Auf Schlüsselmetriken sofort reagieren

Ruth Ablett, Ehud Sharlin, Jörg Denzinger, Frank Maurer, Craig Schock  
Department of Computer Science, University of Calgary

Build notifications for Agile software development teams

Java Lava Lamps use simple red and green lights to convey build state. BuildBot uses sounds, lights and movement.

Both compared with Email

Accountability for failed tests in a playful, fun way

Self-supervision: An fun reminder is better than an angry manager!



**SPECIAL ROBOT TIMES. EDITION**  
25 June 2007. Your Best Source of News-worthy Items. Price Two Cents.

## ROBOT ATTACKS DEVELOPER!

BuildBot, cute puppy overseer, alerts team to potentially disastrous error.

Software developer slightly embarrassed, unhurt after breaking software build.

Build fixed, development continuing as normal, biscuits enjoyed.

It was a scene of deafening calm with a smattering of giggles Thursday when Bob Bobson, developer employed by YesWeAreAgile Software, made a small error in his code, accidentally omitting a small 23-character method call that would have ensured that his code pass the tests he had previously written for it.

"Well, that was silly, wasn't it," commented the smiling 25-year old native of Agileville, Alberta, Canada, also home to the small software development firm.

When Bobson's code was uploaded to the team repository, the error was caught by the continuous integration server when sixteen of the forty-three prewritten tests failed due to the oneline omission. It was then that the robot watchdog, BuildBot, began to move from its power station towards Bobson's desk.

"At first I didn't think it was me" said Bobson, whose code had not been the cause of any previous build breakages. "It started moving, and we could all hear it, and we all started wondering who broke the build. I was thinking, 'It couldn't be me, or could it?'"

Susie Derkins, the project manager, shrugged when hearing about the breakage. "They break it, they fix it. As long as they are aware and someone's taking steps to fix the situation, I'm not worried at all."

As soon as the cuddly robot approached Bobson's desk, it slowly sat down. BuildBot alerted him via a cute but menacing growl that it was in fact Bobson's breakage. Bobson then corrected the error, BuildBot made its way back to its station and life continued more or less as normal. At that point, Bobson took a break to enjoy some of his mother's homemade ginger snap biscuits, which he had been saving for just such an occasion.



# Kontinuierliche Integration

## 7 Schritte

1. **Früh und oft** einchecken
2. **Keinen** Code einchecken, der nicht läuft
3. Build Fehler **sofort** beheben
4. Probleme **früh angehen** und **schnell scheitern**
5. Aufgrund von Metriken **(re)agieren**
6. Auf **allen** Zielplattformen bauen
7. Artefakte für **jeden** Build erstellen



# Hudson



# Hudson

## Historie ...

### **Vorfahren & Verwandte**

- Urvater CruiseControl
- Vater Kohsuke Kawaguchi
- Continuum, Bamboo, TeamCity, Luntbuild, AnthillPro

### **Geburtsort und Datum**

- Sun Microsystems, seit 2005, aktuell 1.300

### **Geburtsurkunde (Lizenzmodell)**

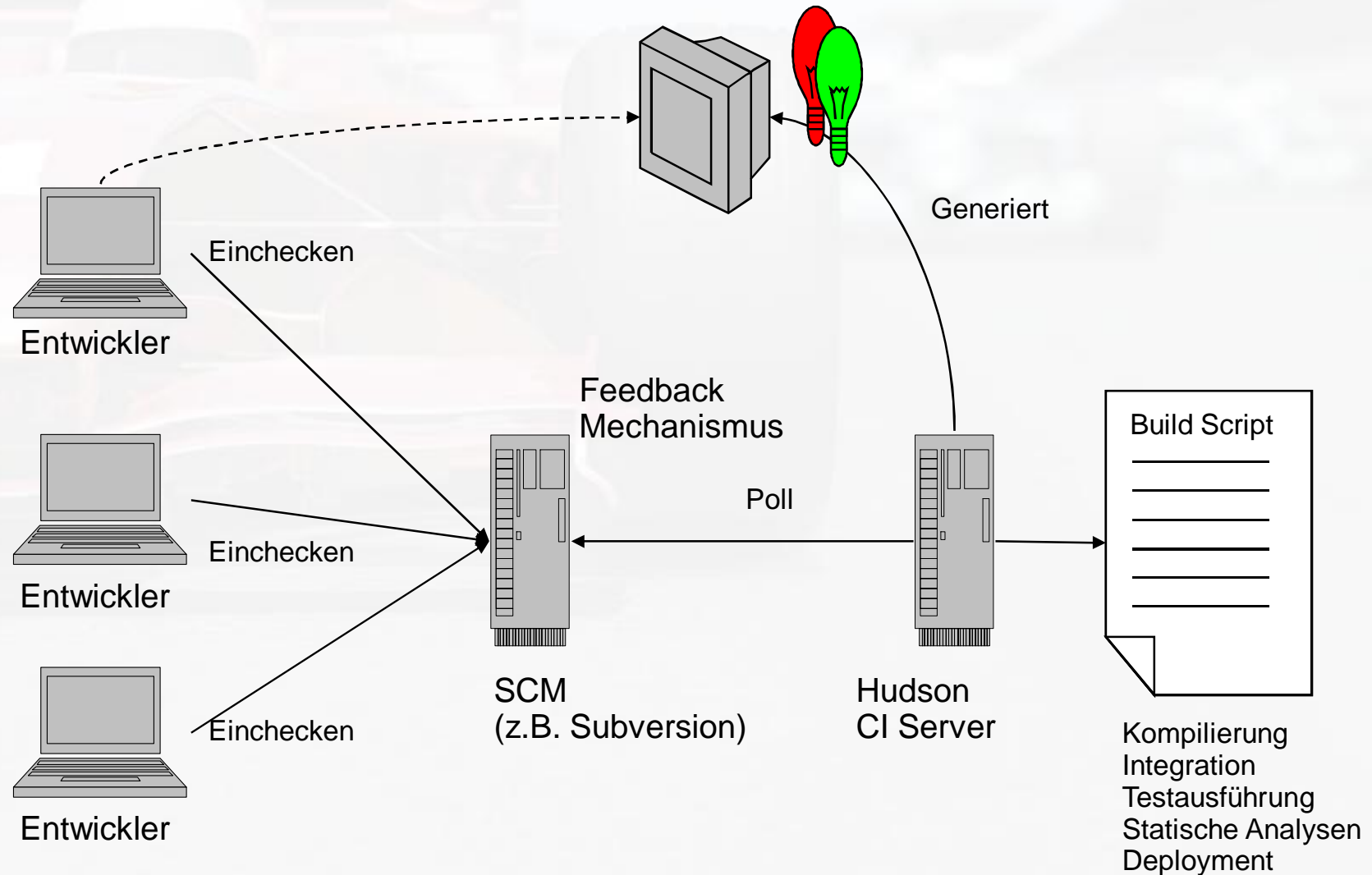
- Mischung aus MIT und Commons Creative

### **Notwendig für die ersten Schritte**

- J2SE 1.5



# Hudson Funktionsprinzip





# Hudson

## Funktionsprinzip

- Sehr schnell “Up und running” (~ 1 Stunde)
  - Konfiguration des Hudson Servers
  - Einrichtung erster Build Jobs
- Skalierbar
  - Master & Slave
  - Einbindung Virtuelle Maschinen
  - Pipe Lining von Jobs
- Erweiterbar
  - Umfangreicher Plugin Support (135)



# Hudson Live Demo



<http://localhost:8080>



# Hudson

## Einrichtung allgemein

### Hudson installieren & verwalten

- System konfigurieren
- Plugins & Knoten verwalten
- Als Service installieren

### Betrieb

- Neu laden
- Herunterfahren vorbereiten

### Diagnose

- Logging, System Log, Statistik
- Skript-Konsole

### Hudson verwalten



#### System konfigurieren

Globale Einstellungen und Pfade konfigurieren.



#### Konfiguration von Festplatte neu laden

Verwirft alle Daten im Speicher und lädt die Konfiguration neu.



#### Plugins verwalten

Plugins installieren, deinstallieren, aktivieren oder deaktivieren.



#### Systeminformationen

Zeigt Umgebungsinformationen an, z.B. zur Unterstützung bei Problemen.



#### Systemlog

Zeigt protokollierte Ereignisse im Hudson System.



#### Nutzungsstatistiken

Ressourcenauslastung überwachen und überprüfen.



#### Skript-Konsole

Führt ein beliebiges Skript aus zur Administration.



#### Knoten verwalten

Knoten hinzufügen, entfernen, steuern und überwachen.



#### Benutzer verwalten

Anlegen, Aktualisieren und Löschen von Benutzern.



#### Install as Windows Service

Installs Hudson as a Windows service to this system.



#### Herunterfahren vorbereiten

Verhindert die Ausführung neuer Builds, so daß das System heruntergefahren werden kann.



# Hudson

## Einrichtung Jobs

### Job Typen

- Free Style
- Maven 2
- Multikonfigurationsprojekt
- Externen Job überwachen

Job Name

**"Free Style"-Softwareprojekt bauen**  
Dieses Profil ist das meistgenutzte in Hudson, beschränkt, sondern kann darüber hinaus

**Maven 2 Projekt bauen**  
Dieses Profil baut ein Maven 2 Projekt. Hudson es ist aber bereits einsetzbar, um Rückme

**Multikonfigurationsprojekt bauen (alpha)**  
Dieses Profil eignet sich sehr gut für Proje

**Externen Job überwachen**  
Dieses Profil erlaubt die Überwachung von Protokollierung von automatisch ausgefü

**Kopiere bestehenden Job**  
Kopiere von



# Hudson

## Einrichtung Jobs

### Multikonfigurationsprojekt

- Definition von Achsen
- Varianten der Achsen werden als Variablen an den Build übergeben
- Alle Varianten werden durch mehrfachen Aufruf des gleichen Build-Skripts abgearbeitet

 **Build #19 (21.04.2009 18:50:23)**



Revision: 8

Keine Änderungen.



Started by user [infomar](#)

#### Konfigurationsmatrix

	1.5	1.6
oracle		
mysql		

#### Permalinks

- [Buildnummer](#)



# Hudson

## Einrichtung Jobs

### Externe Job überwachen

- Besteht lediglich aus Namen
- Benachrichtigung über Job Ausführungen erfolgt mittels
  - XML:

```
<run>  
  <log encoding='hexBinary'>...</log>  
  <result>integer. 0 is success and everything else is failure</result>  
  <duration>milliseconds it took to execute this run</duration>  
</run>
```
  - HTTP: [http://myhost/hudson/job/<\\_jobName\\_>/postBuildResult](http://myhost/hudson/job/<_jobName_>/postBuildResult).

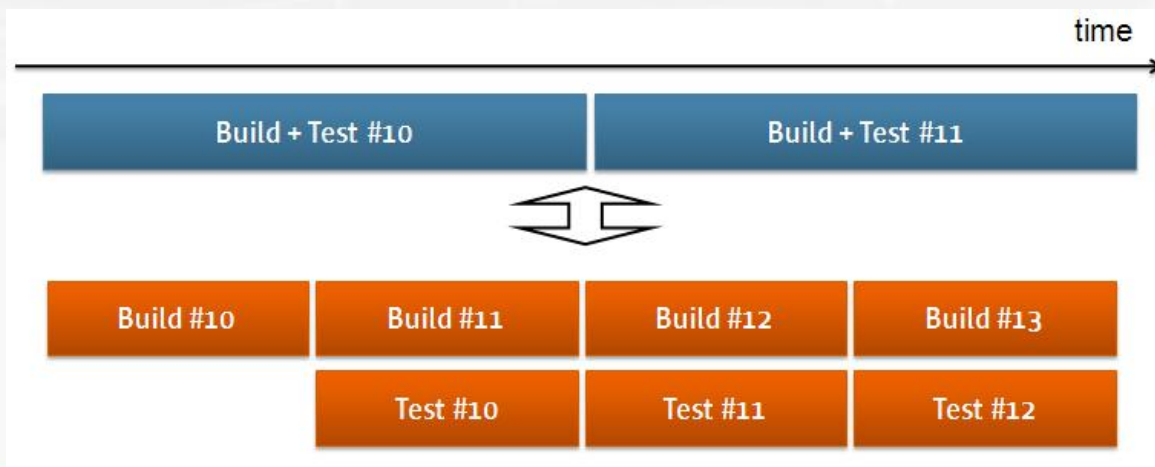


# Hudson

## Einrichtung Jobs

### Jobs miteinander verknüpfen (Pipe Lining)

- Vor- oder nachgelagerte Projekte definieren
- Artefakte in vorgelagerten Projekten archivieren
- Archivierte Artefakte in nachgelagerten Projekten holen
- Fingerprinting von Artefakten in vor- und nachgelagerten Projekten hilft bei deren Zuordnung



#### Projekt ConnectFourStream1

-  [Arbeitsbereich](#)
-  Artefakte des letzten erfolgreichen Builds
  - [connectfour.jar](#)
-  [Letzte Änderungen](#)

**Nachgelagerte Projekte**

- [ConnectFourStream2](#)

**Permalinks**

- [Letzter Build \(#21\), vor 23 Minuten](#)
- [Letzter stabiler Build \(#21\), vor 23 Minuten](#)
- [Letzter erfolgreicher Build \(#21\), vor 23 Minuten](#)
- [Letzter fehlgeschlagener Build \(#19\), vor 33 Minuten](#)



# Hudson

## Einrichtung Master & Slave

- Konfiguration der Slaves auf Master
- Einrichtung der Slaves
- Abstimmung Jobs & Slaves, damit Builds auf korrekte Maschinen kommen
- Master verteilt Builds

**Hudson**

Hudson » [nodes](#)

[Zurück zur Übersicht](#)

[New Node](#)

**Geplante Builds**  
Keine Builds geplant.

**Build-Processor Status**

#	Master
1	Bereit
2	Bereit

**linus (offline)**

1	<a href="#">Offli</a>	Name	snoopy
2	<a href="#">Offli</a>	Beschreibung	
		Anzahl der Build-Prozessoren	1
1	<a href="#">Offli</a>	Stammverzeichnis in entferntem Dateisystem	c:\temp\hudson
2	<a href="#">Offli</a>	Labels	Windows
		Usage	Utilize this slave as much as possible
		Startmethode	Starte Slave-Agenten über JNLP
		Verfügbarkeit	Slave immer angeschaltet lassen.

**Node Properties**

Umgebungsvariablen

Tool Locations

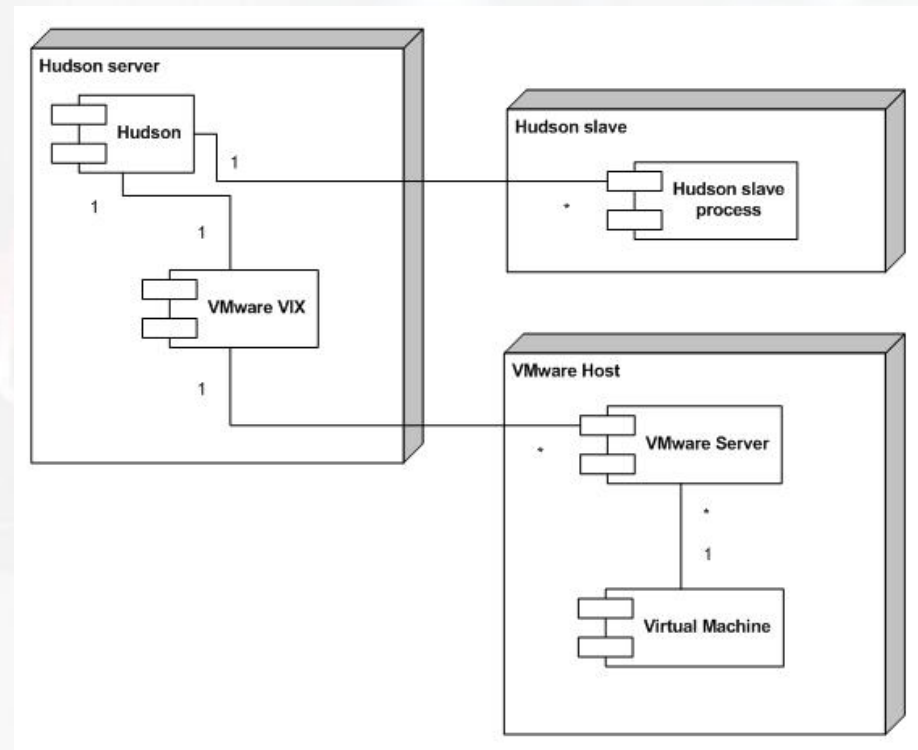




# Hudson

## Einrichtung virtuelle Welten

- Konfiguration des Hudson Servers zur Kommunikation mit Vmware Host
- Konfiguration der Jobs zur Zusammenarbeit mit bestimmten virtuellen Rechnern
- Aktuell auf einen Host begrenzt (besser wäre Schnittstelle zu VMware Virtual Infrastructure Server)



# Hudson

## Einrichtung Zugriff

### Benutzer & Rechte

Hudson absichern

TCP-Port für JNLP-Slave

Zugriffskontrolle

Statisch :   Zufällig  Deaktivieren

#### Benutzerverzeichnis (Realm)

Hudsons eingebautes Benutzerverzeichnis

Benutzer dürfen sich selbst registrieren

LDAP

An Servlet-Container delegieren

#### Rechtevergabe

Matrix-basierte Sicherheit

Angemeldete Benutzer dürfen alle Aktionen ausführen

Jeder darf alle Aktionen ausführen

Project-based Matrix Authorization Strategy

Legacy-Autorisierung

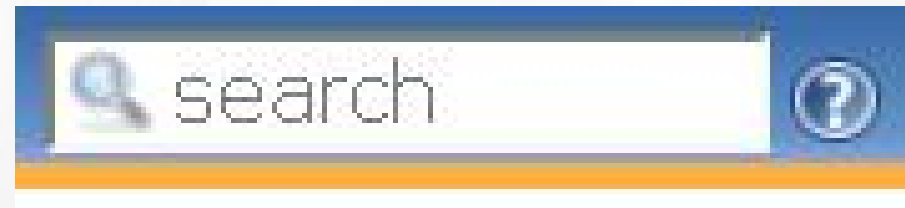


# Hudson

## Features & Plugins

### Schnelle Navigation über ...

- Knotentypen, z.B. job, ws, changes, configure, etc.
- Über Knotennamen, z.B.
  - Job Name
  - Build Nummer
  - Slave Rechner



# Hudson

## Features & Plugins

### Remote Access API

- REST Style
- Ermittlung von Daten für programmatische Weiterverarbeitung
- Anstoßen eines Builds
- Erzeugung oder Kopieren von Jobs

### Remote API

Many objects of Hudson provides the

#### XML API

Access data exposed in [HTML](#)

You can also specify optional result will be sent with `text/` parameter to specify the name

Similarly `exclude` query parameter. This query parameter can be :

#### JSON API

Access the same data as JSON

#### Python API

Access the same data as Python identical to that of JSON. However malicious Python programs.



# Hudson

## Features & Plugins

- SCMs (GIT, Mercurial, ClearCase, etc.)
- Benachrichtigungsmechanismen (z.B. RSS, E-mail, IM, Twitter, GoogleCalendar, etc.)
- Statische Analysen (z.B. PMD, Checkstyle, Sonar, etc.)
- Tests (z.B. JUnit, TestNG, Selenium, JMeter, Grinder, etc.)
- Integration mit anderen Sites/Tools (z.B. Jira, Mantis, etc.)
- Meine Lieblinge
  - Permanente Links (z.B. für LATEST builds)
  - Nachträgliches Tagging von Builds
  - Trends
- Spezielles (z.B. Claim, Task Scanner, CI Game)



# Hudson

## Gemachte Erfahrungen

### Wo ist noch Potential ☹

- Für manche Plugins bis heute nicht klar, ob oder was sie tun
- Flüchtigkeitsfehler (z.B. in 1.219 bei gleichen Build-Verfahren und Build-Skript aber unterschiedlichen Targets)
- Wenn Hudson und Plugins verschiedene Versionen der gleichen Library verwenden wollen (z.B. commons-lang)
- Umgang mit Passwörtern in Plugins (z.B. Google Cal)
- Schreiben von Plugins ist nicht einfach (Einarbeitung in Jelly, nur mit Maven)



# Hudson

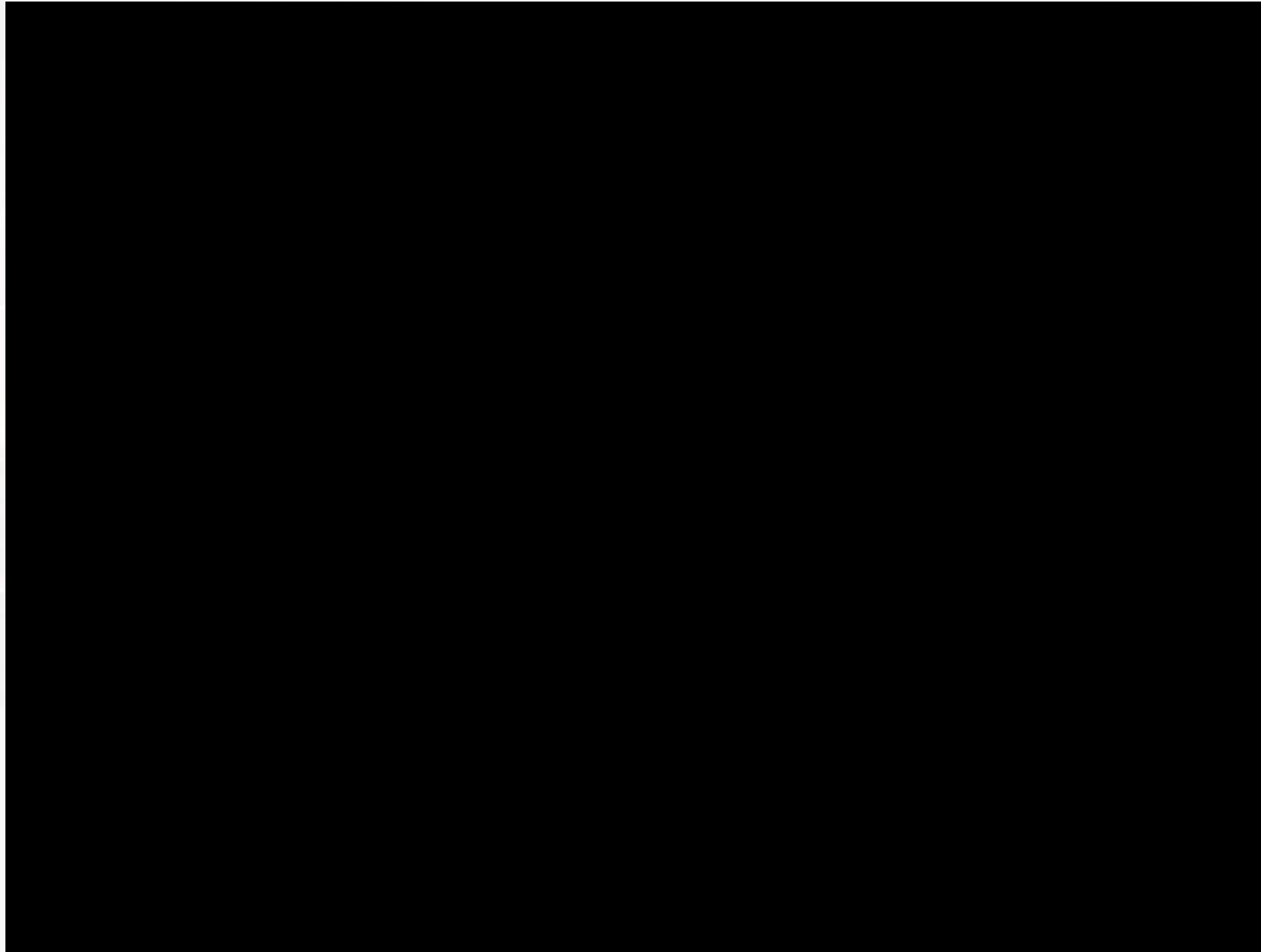
## Gemachte Erfahrungen

### Positiv aufgefallen ☺

- Sehr schnell „Up and Running“ (~ 1 Stunde)
- Eine Oberfläche für Konfiguration und Betrieb
- Permanente Links (z.B. für LATEST builds)
- Nachträgliches Tagging
- Berechtigungskonzept
- Starke Skalierungsunterstützung
- Schnelle Bearbeitung von Fehlern und Feature Wünschen
- Unzählige Plugins (~ 135)



# Ein kleiner Film zum Schluss ...





# Literatur und Links

## Bücher

- Continuous Integration, Paul M. Duval

## Links

- <https://hudson.dev.java.net/>
- <http://www.martinfowler.com/articles/continuousIntegration.html>
- <http://www.slideshare.net/drluckyspin/continuous-integration>
- <http://www.slideshare.net/carlo.bonamico/continuous-integration-with-hudson/>
- <http://jboss-qa.blogspot.com/2007/10/taking-continuous-integration-to.html>
- <http://www.it-agile.de/build-flashfilm.html>
- <http://www.sonatype.com/people/2009/01/the-hudson-build-farm-experience-volume-i/> (sowie deren Fortsetzungen)



# Literatur und Links

## Bilder Referenzen (cc creative commons)

- <http://www.flickr.com/photos/carstingaxion/1103931822/>
- <http://www.flickr.com/photos/lukepdq/99418297/>
- <http://creativecommons.org/licenses/by-nc-nd/2.0/deed.en>
- <http://www.flickr.com/photos/paopix/2413495787/>
- <http://www.flickr.com/photos/sonosalvo/171714927/>
- <http://www.flickr.com/photos/heyjules/2144592427/>
- <http://www.flickr.com/photos/destinme/1267500829>
- <http://www.flickr.com/photos/dullhunk/359634390/>
- <http://www.flickr.com/photos/k9/556002530/>
- <http://www.flickr.com/photos/ppdigital/2329376071/>
- <http://www.flickr.com/photos/iko/739595/>



# Noch Fragen? Aber gern



Martin Heider arbeitet seit mehr als 15 Jahren im Bereich Software-Entwicklung. Als Freiberufler unterstützt er Kunden in verschiedenen Rollen als SW-Entwickler, SW-Architekt, Testmanager, Team- und Entwicklungsleiter oder Coach.

Seine Erfahrung umfasst international verteilte Projekte sowie Teams verschiedener Größen. Sein besonderes Interesse gilt agilen Methoden und der Herausforderung Software-Entwicklung einfach zu machen, damit alle Beteiligten mit mehr Spaß bessere Ergebnisse erzielen.

Sie erreichen mich unter: [mh@infomar.de](mailto:mh@infomar.de)

