DR.WIEST
SOFTWARETECHNIK

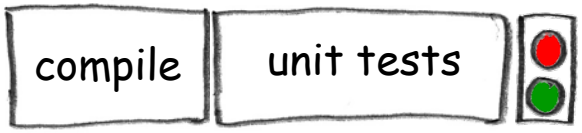
Liebling, ich habe den Build geschrumpft!

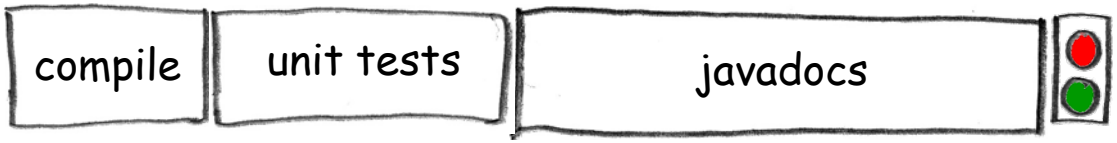
Endlich (wieder) schnellere Builds mit Hudson

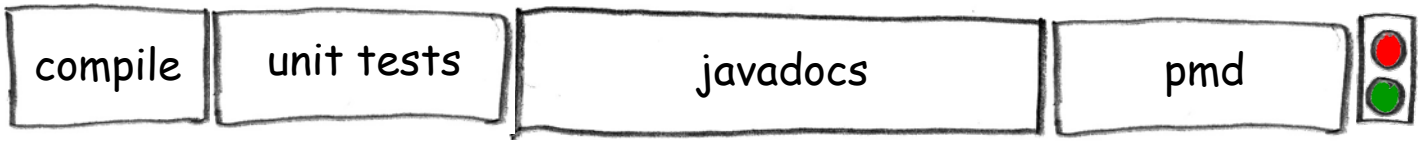
Dr. Simon Wiest

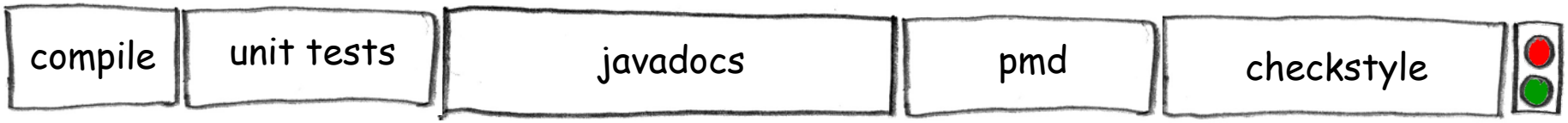
XP-Days 2009
27.11.2009, Karlsruhe

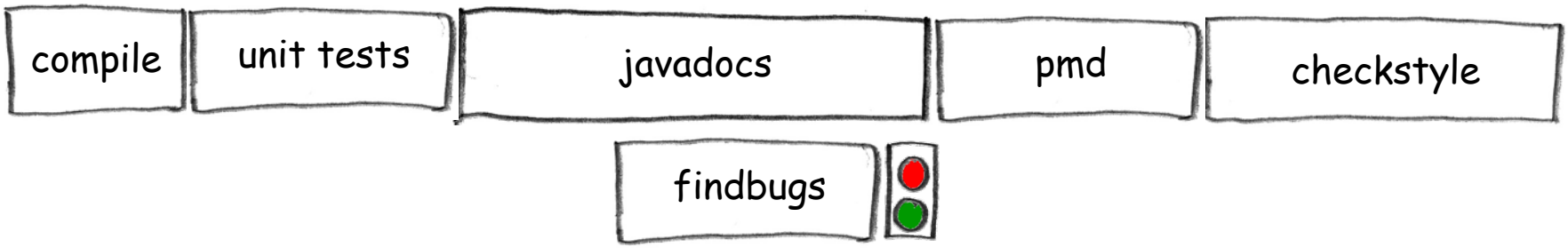

compile

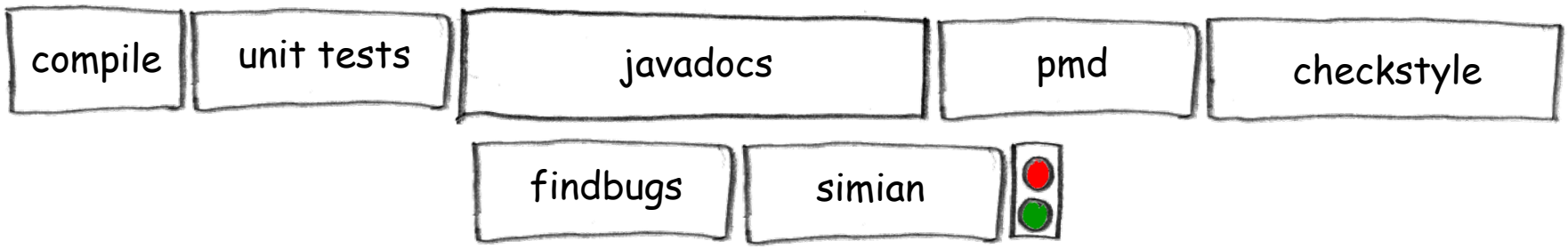

compile
unit tests

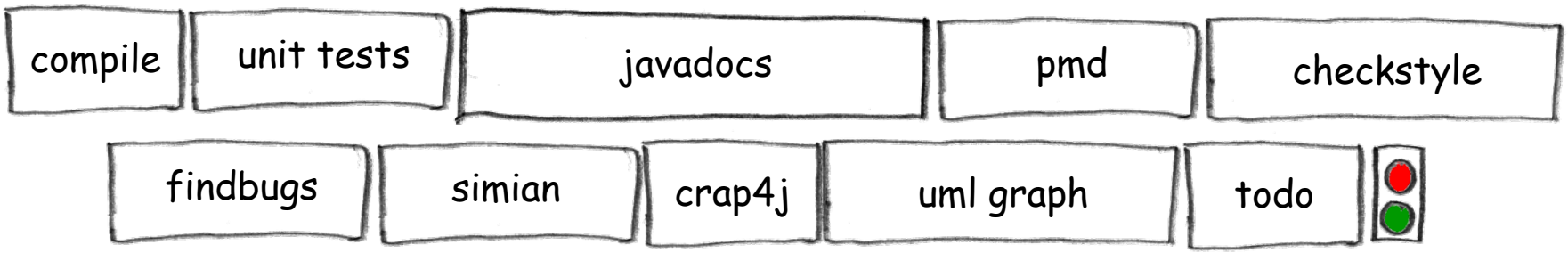

compile
unit tests
javadocs

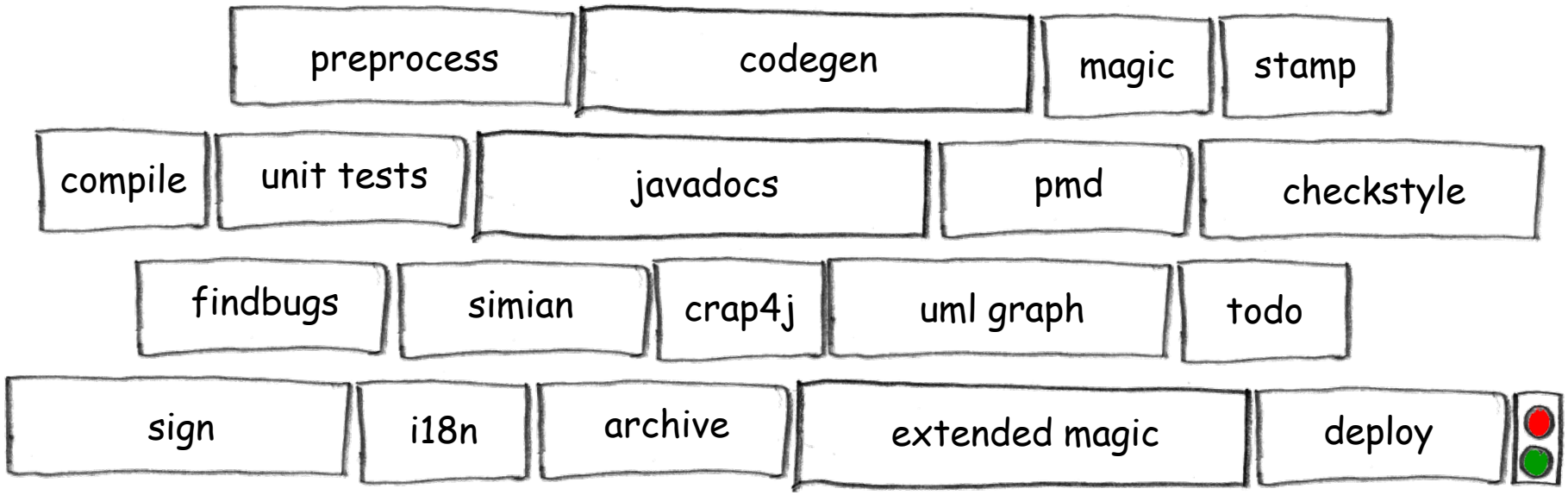

compile
unit tests
javadocs
pmd

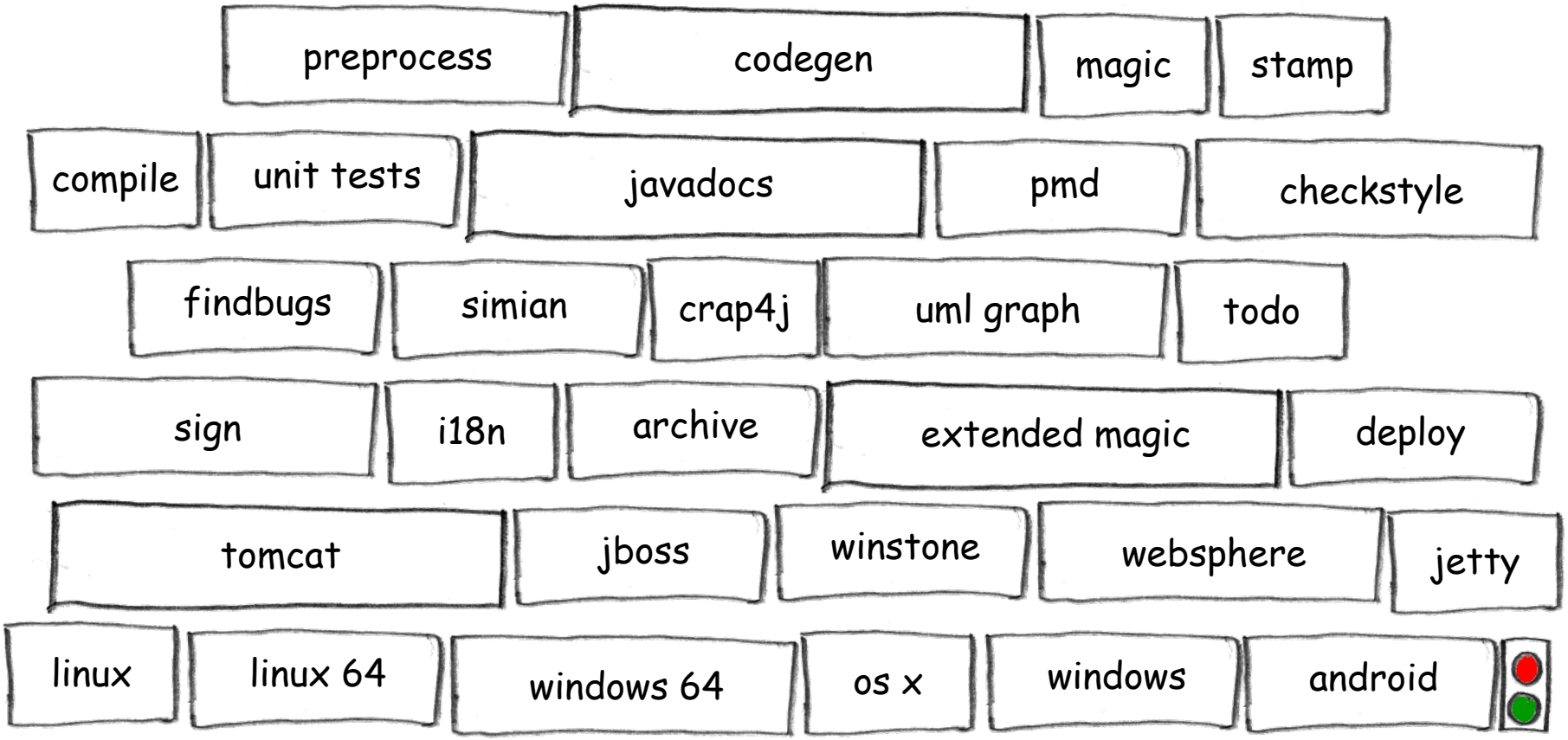

compile
unit tests
javadocs
pmd
checkstyle

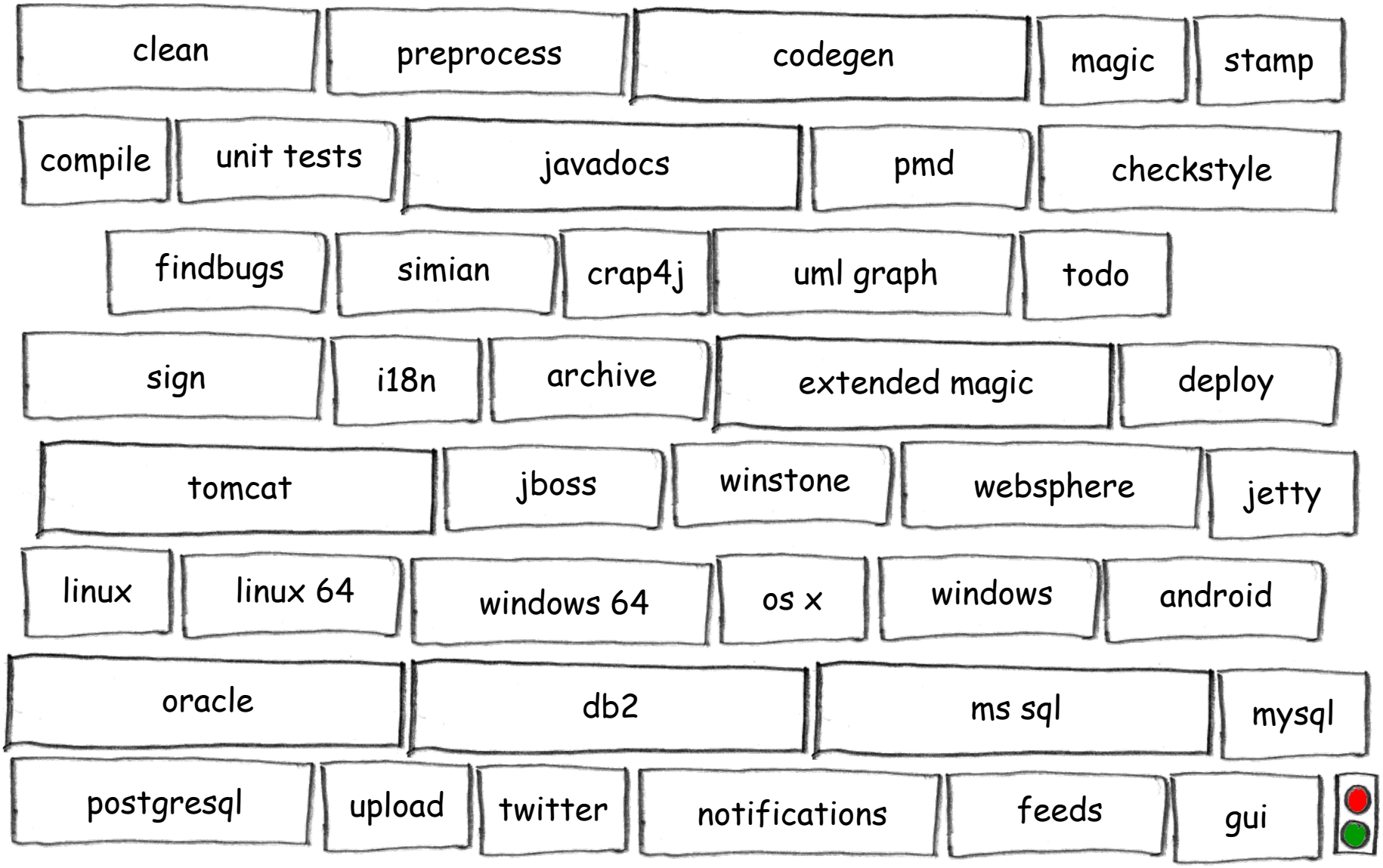

compile
unit tests
javadocs
pmd
checkstyle
findbugs

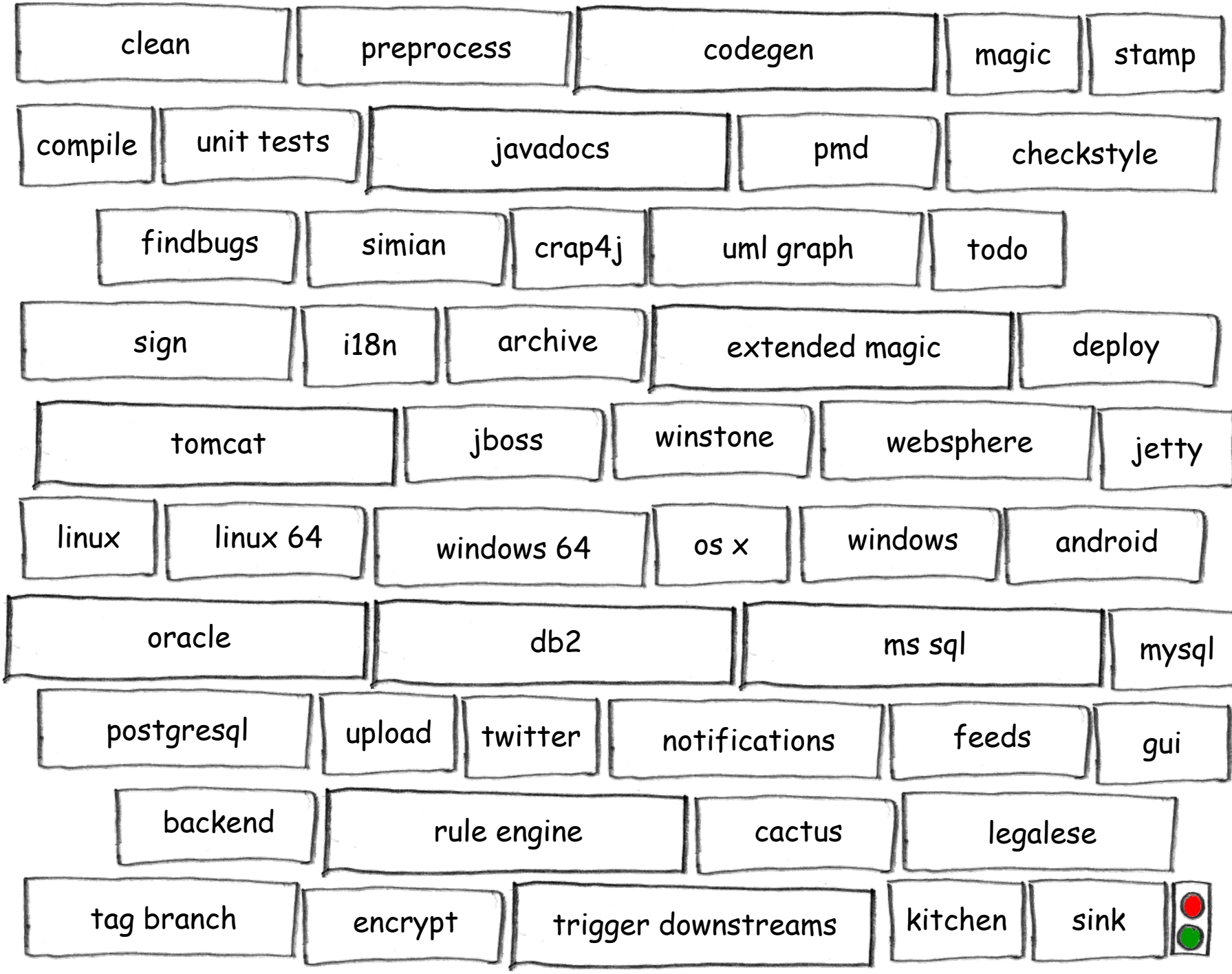

compile
unit tests
javadocs
pmd
checkstyle
findbugs
simian


compile
unit tests
javadocs
pmd
checkstyle
findbugs
simian
crap4j
uml graph
todo

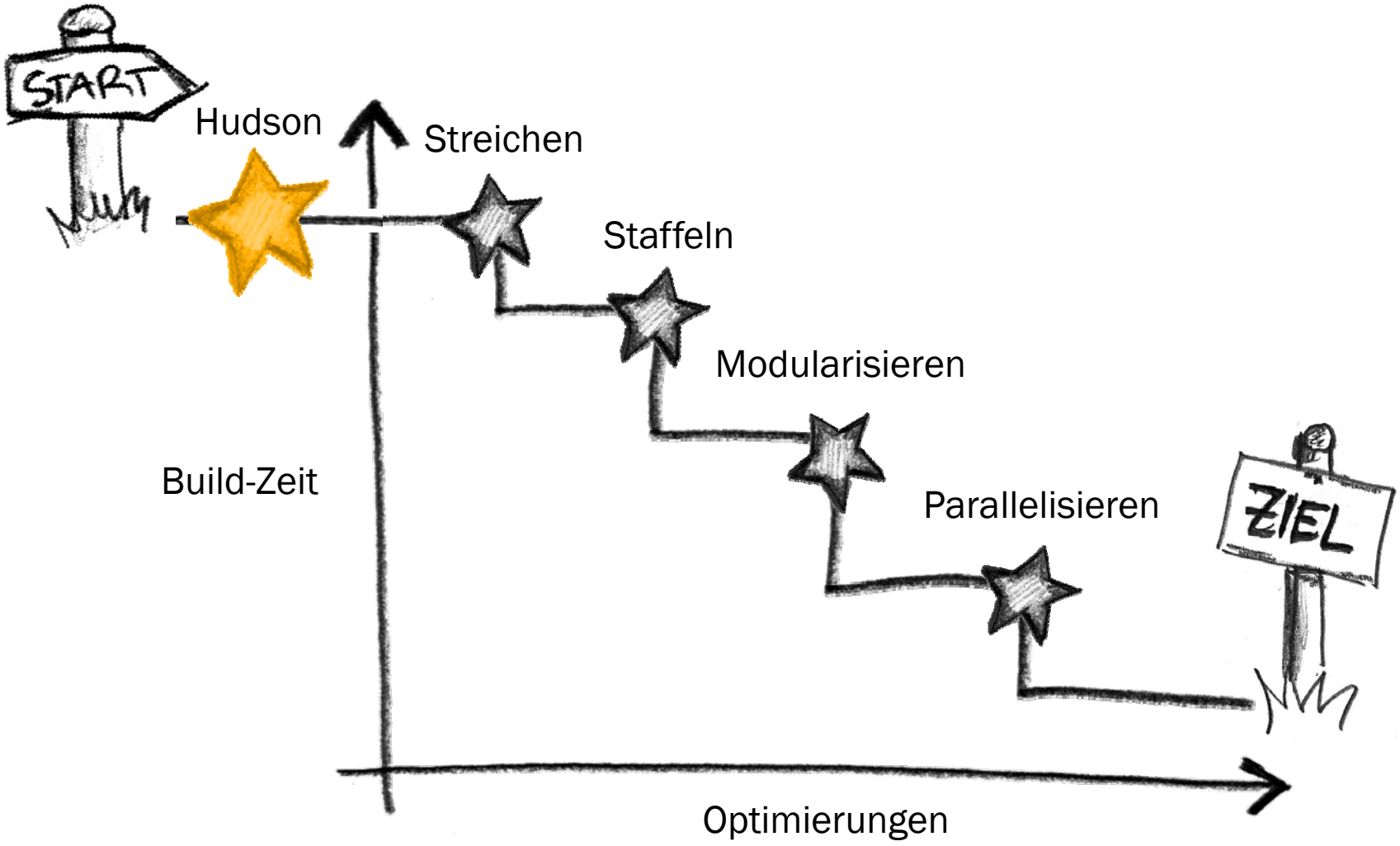

preprocess
codegen
magic
stamp
compile
unit tests
javadocs
pmd
checkstyle
findbugs
simian
crap4j
uml graph
todo
sign
i18n
archive
extended magic
deploy


preprocess
codegen
magic
stamp
compile
unit tests
javadocs
pmd
checkstyle
findbugs
simian
crap4j
uml graph
todo
sign
i18n
archive
extended magic
deploy
tomcat
jboss
winstone
websphere
jetty
linux
linux 64
windows 64
os x
windows
android


clean
preprocess
codegen
magic
stamp
compile
unit tests
javadocs
pmd
checkstyle
findbugs
simian
crap4j
uml graph
todo
sign
i18n
archive
extended magic
deploy
tomcat
jboss
winstone
websphere
jetty
linux
linux 64
windows 64
os x
windows
android
oracle
db2
ms sql
mysql
postgresql
upload
twitter
notifications
feeds
gui

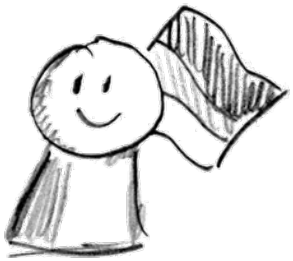

clean
preprocess
codegen
magic
stamp
compile
unit tests
javadocs
pmd
checkstyle
findbugs
simian
crap4j
uml graph
todo
sign
i18n
archive
extended magic
deploy
tomcat
jboss
winstone
websphere
jetty
linux
linux 64
windows 64
os x
windows
android
oracle
db2
ms sql
mysql
postgresql
upload
twitter
notifications
feeds
gui
backend
rule engine
cactus
legalese
tag branch
encrypt
trigger downstreams
kitchen
sink

Agenda


START
Hudson
Streichen
Staffeln
Modularisieren
Parallelisieren
ZIEL
Build-Zeit
Optimierungen

Über den Referenten: Dr. Simon Wiest

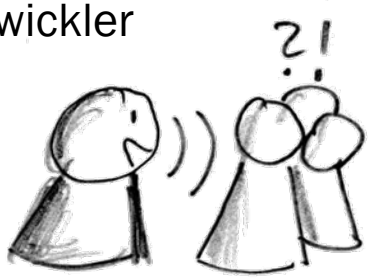
Privat: Hudson-Committer



Lokalisierung DE

Internationalisierung

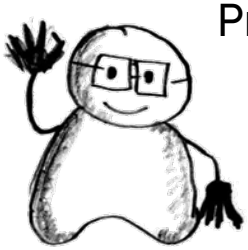
Plug-In Entwickler



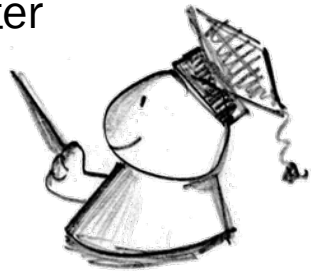
Support (Mailing Listen)

Beruflich: Hudson-Anwender

Projektleiter



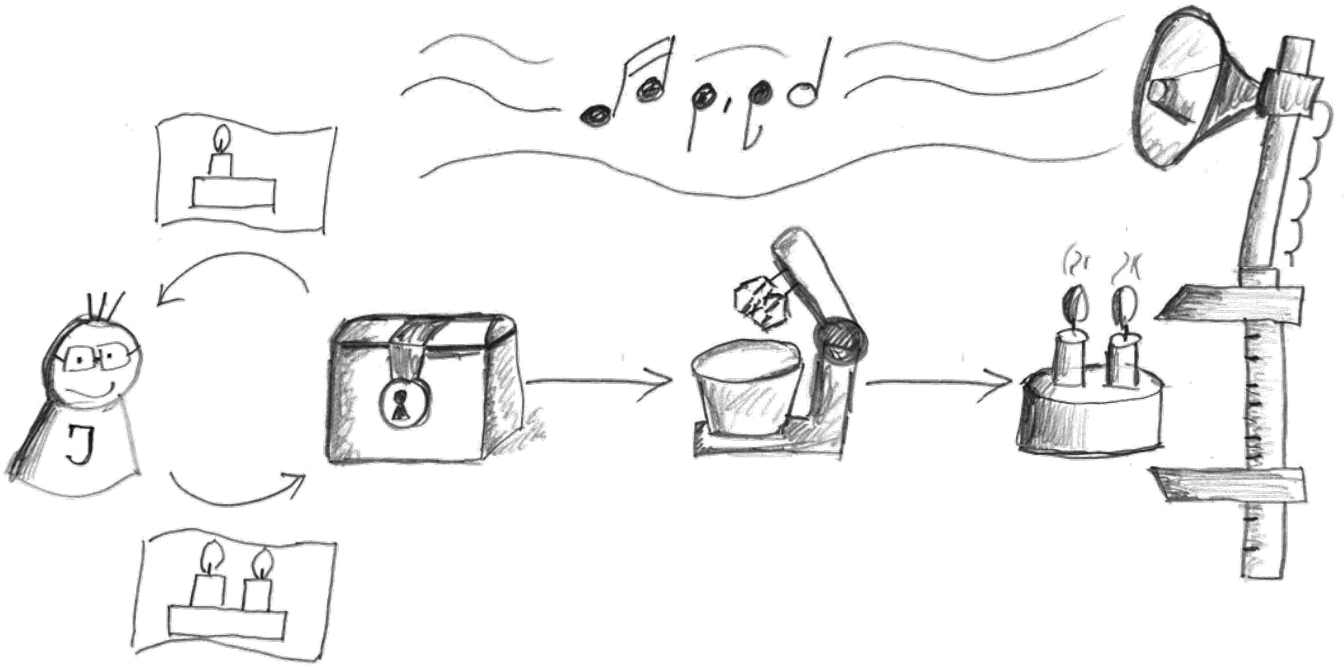
Java Architekt



Coach

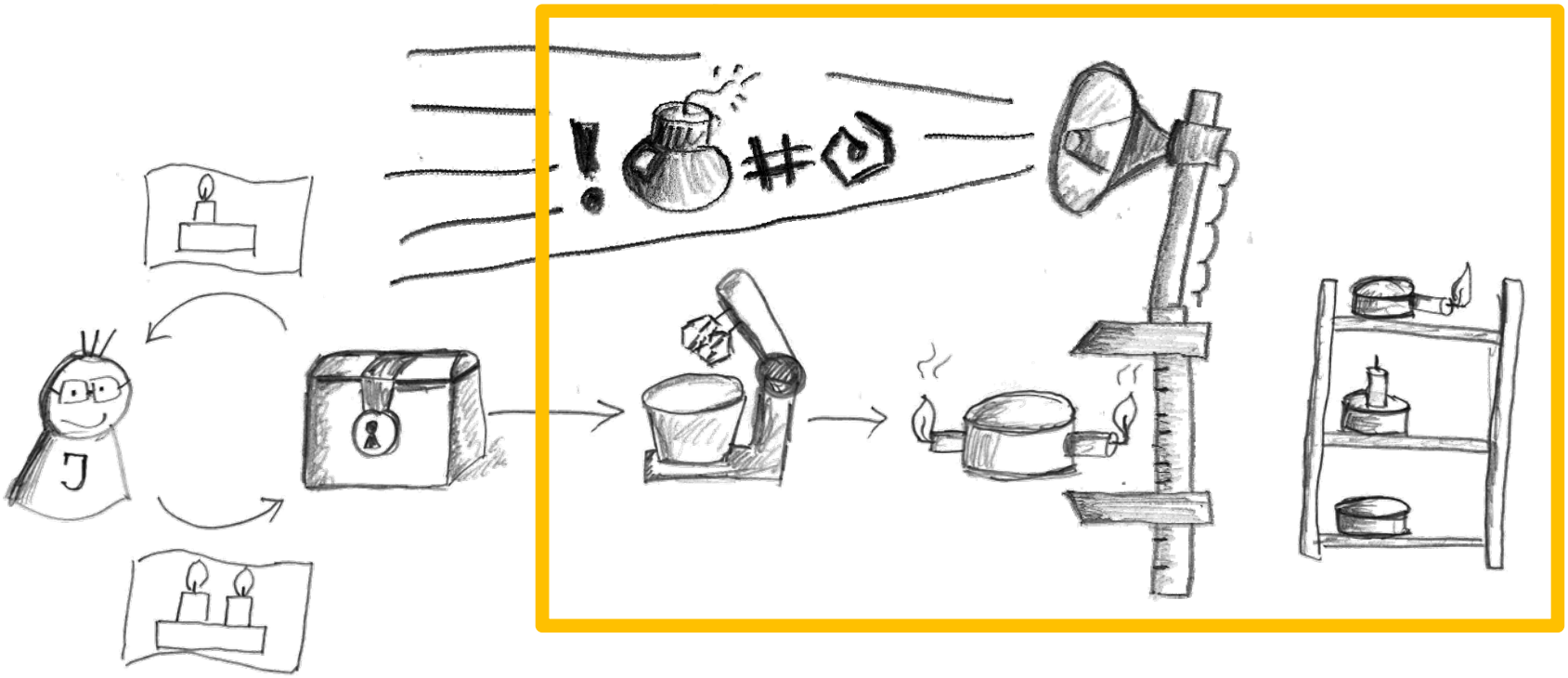
IT-Freiberufler

Was ist kontinuierliche Integration (CI)?

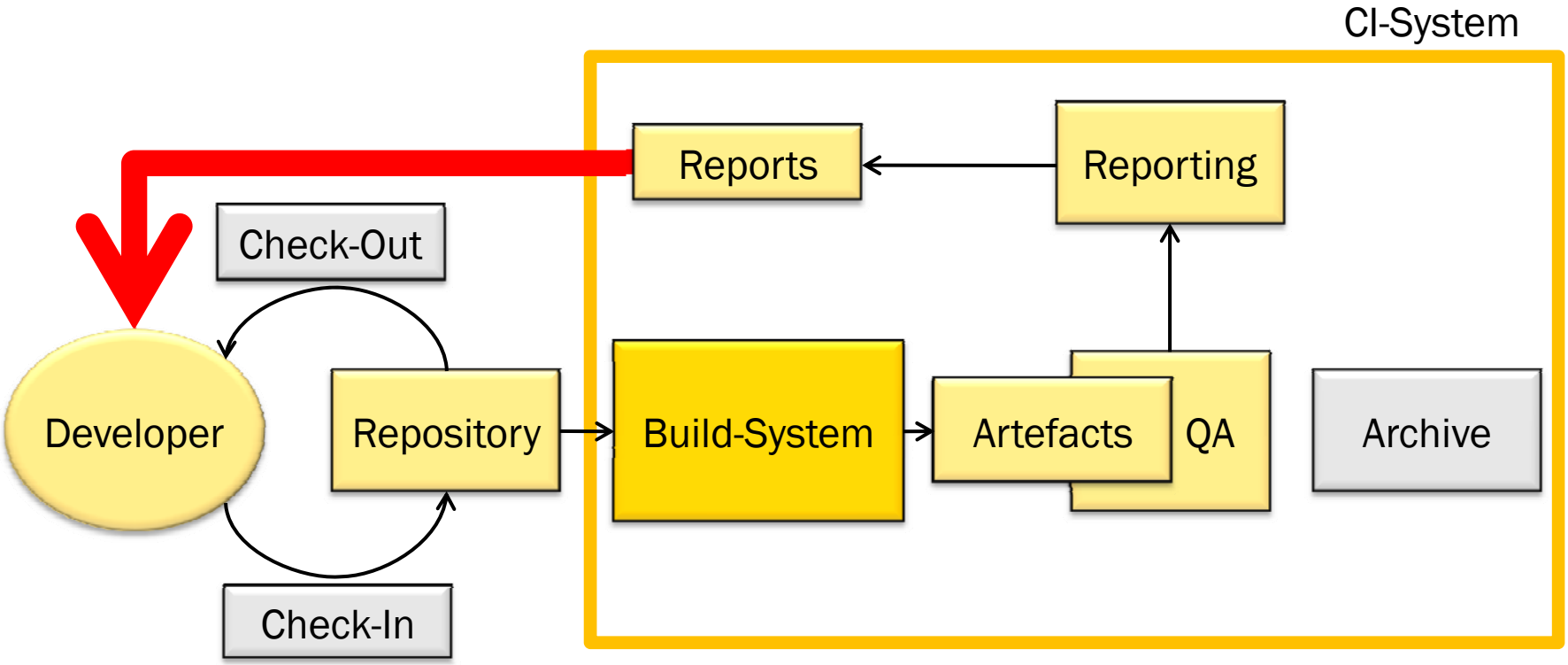


Was ist kontinuierliche Integration (CI)?

orchestriert durch CI-System



Was ist kontinuierliche Integration (CI)?



- Subversion
- CVS
- Perforce
- Git
- ...

- Ant
- Maven
- Shell-Skript
- Batch-Datei
- ...

- JUnit
- TestNG
- CheckStyle
- PMD
- ...

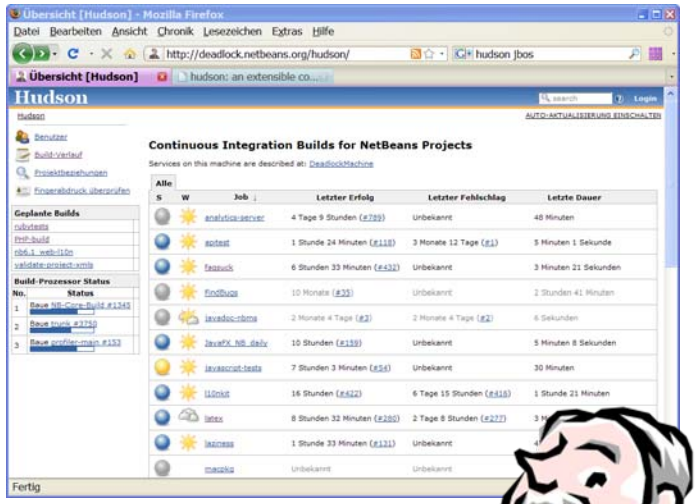
Rückmeldung mit Xtreme Feedback Devices (XFDs)




Photos: Kohsuke Kawaguchi, violet.net, Simon Wiest, Hartmut Lang

Hudson auf einen Blick

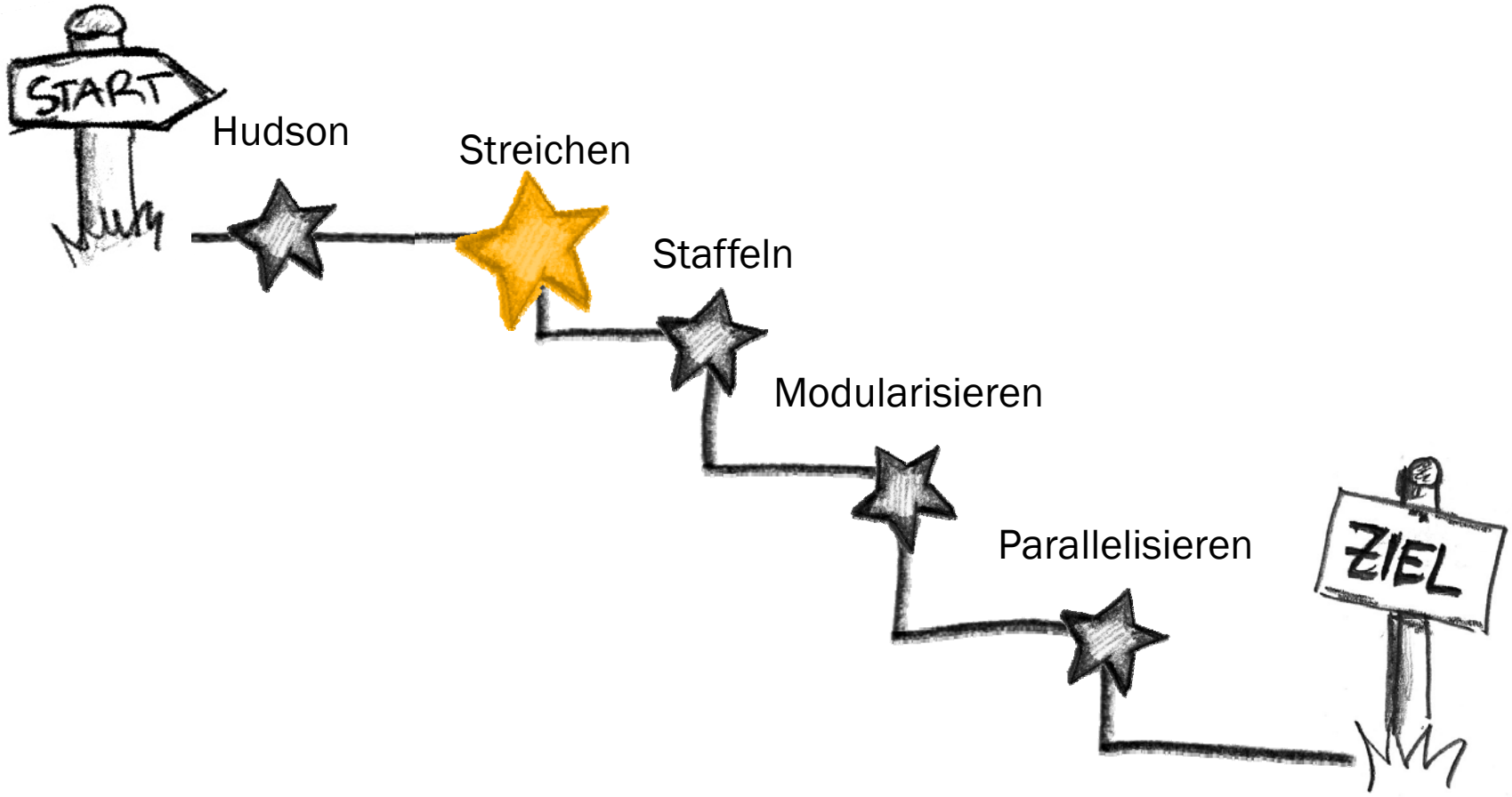
- Java-basierte Web-Anwendung
- Initiiert 2006/2007 von Kohsuke Kawaguchi (Sun)
- Teil des Projekts Glassfish
- Open Source (MIT Lizenz)
- 1,24 Mio. LOC (mit Plugins)
- Zur Zeit 180+ Plug-Ins
- 130+ Beitragende



Wer verwendet Hudson (und darf es zugeben)?

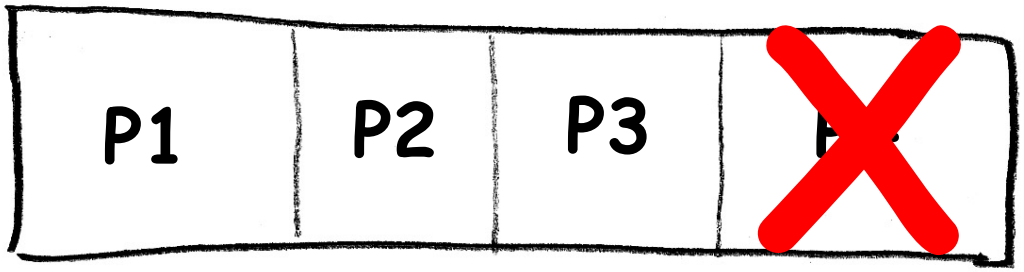
Quelle: Kohsuke Kawaguchi, wiki.hudson-ci.org//x/EYQ5Ag

Agenda

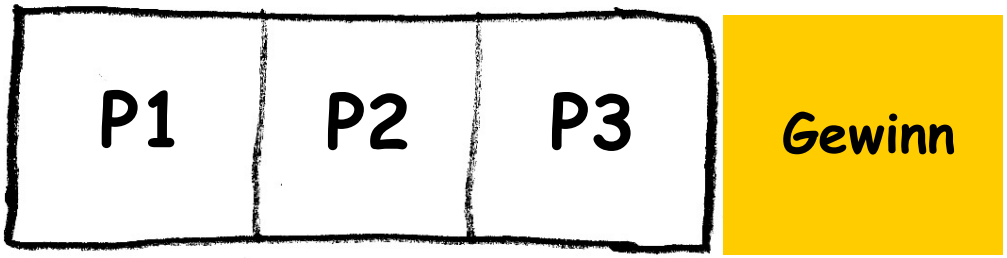

START
Hudson
Streichen
Staffeln
Modularisieren
Parallelisieren
ZIEL

Streichen: Was kann entfallen?

vorher

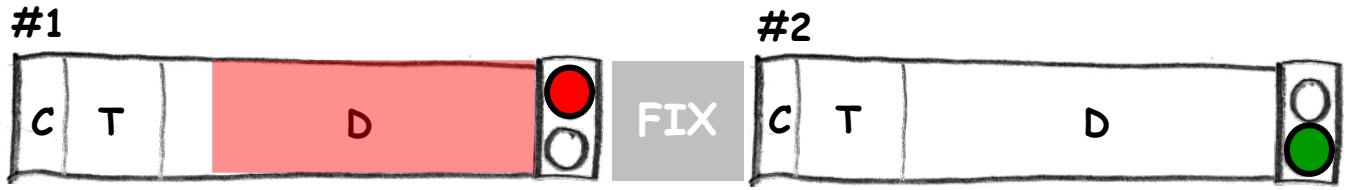


nachher

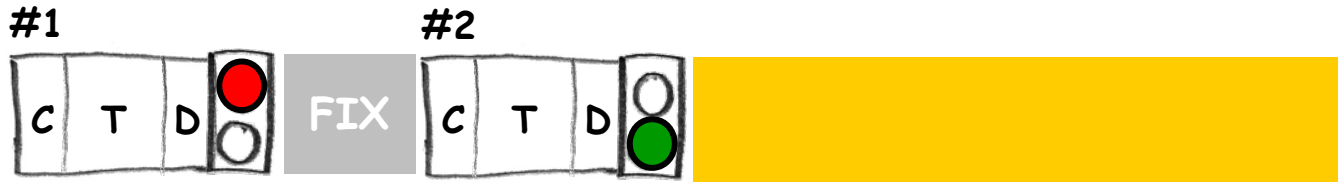


Streichen: Beispiel

vorher

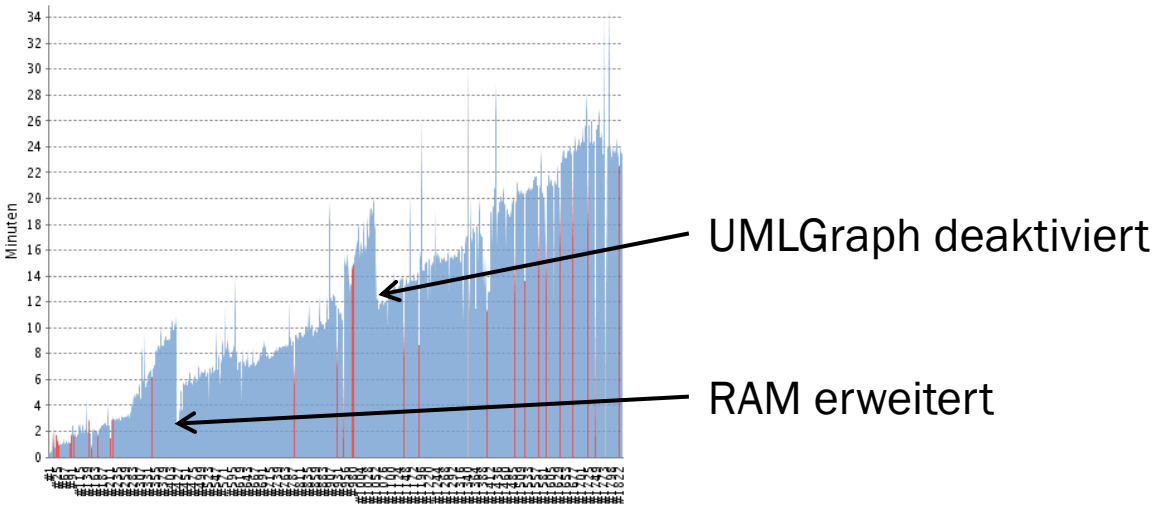


nachher



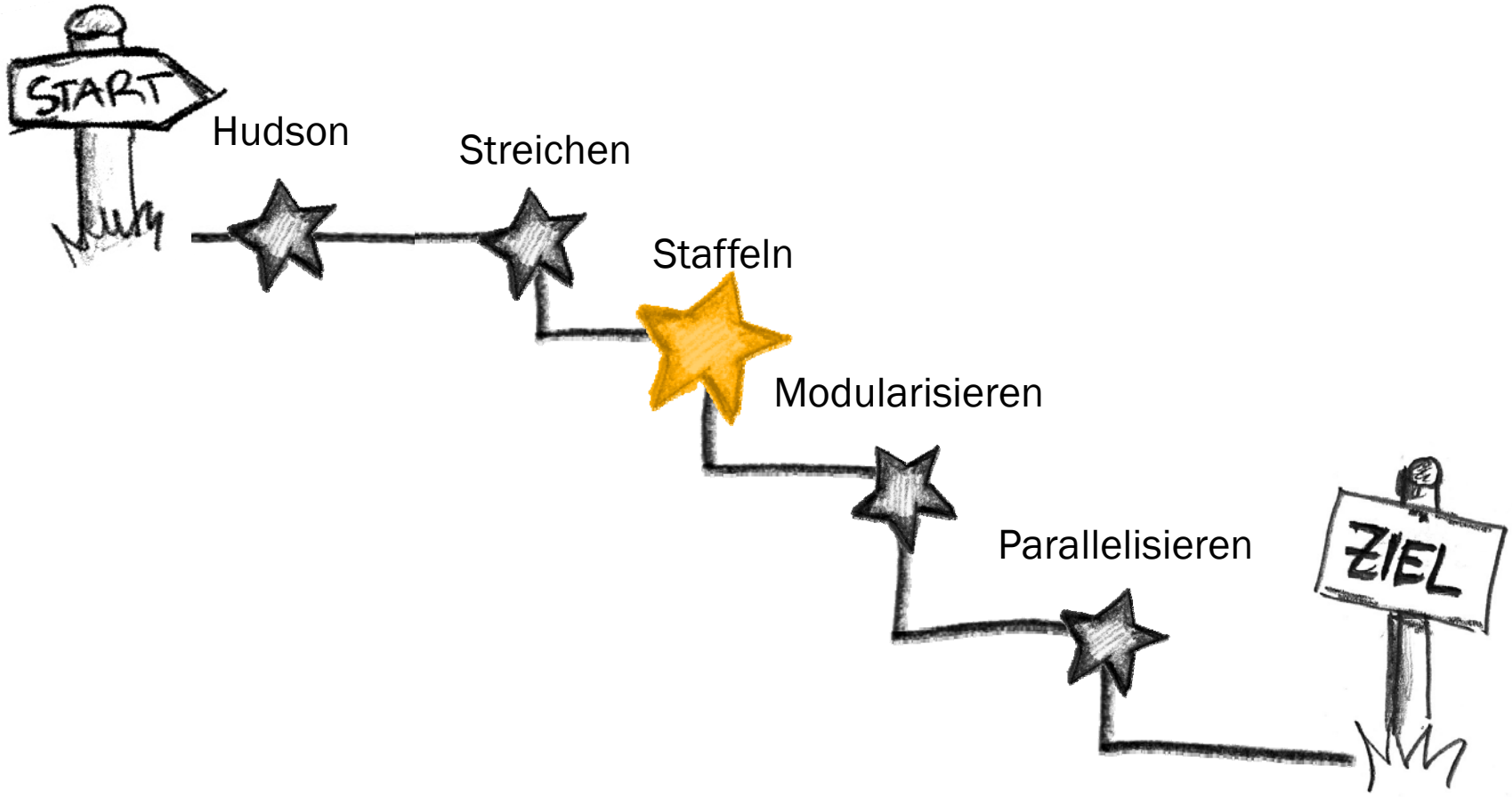
Streichen: Wie unterstützt hier Hudson?

- Trends über Projektverlauf, z.B. Buildzeit



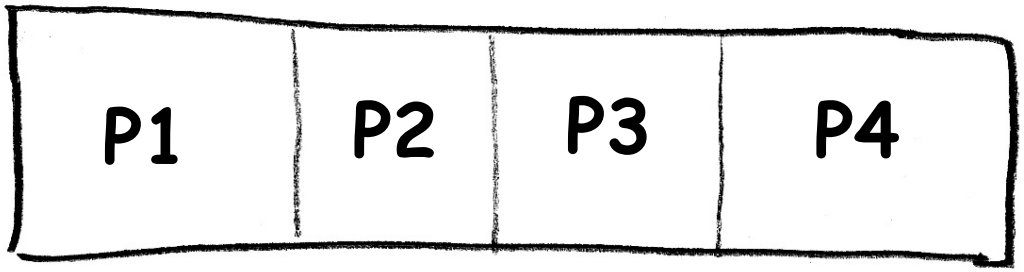
- Intuitive Ad-Hoc-Analyse in Web-Oberfläche

Agenda

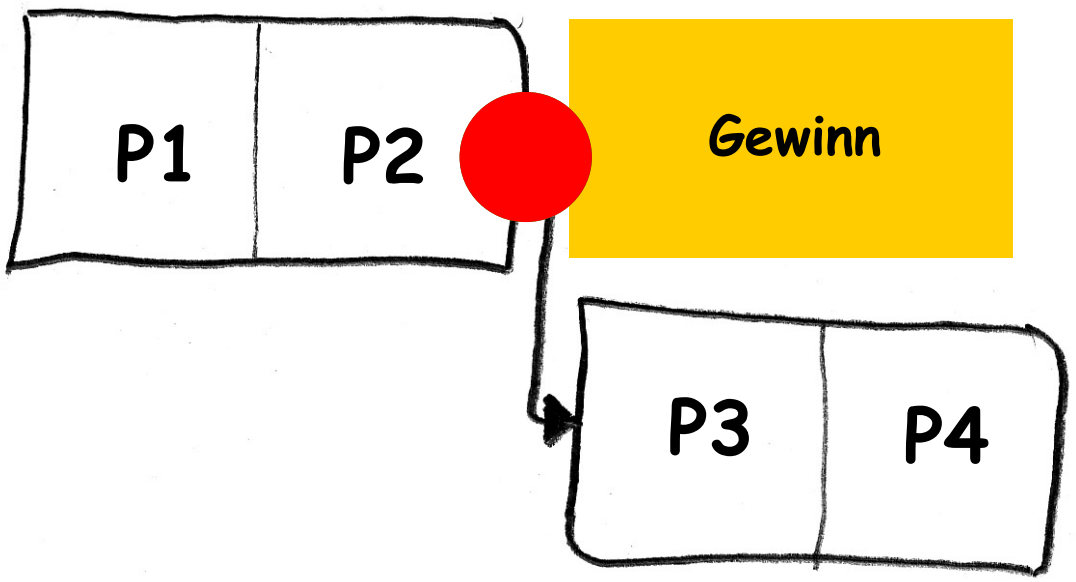

START
Hudson
Streichen
Staffeln
Modularisieren
Parallelisieren
ZIEL

Staffeln: Das Wichtigste zuerst!

vorher



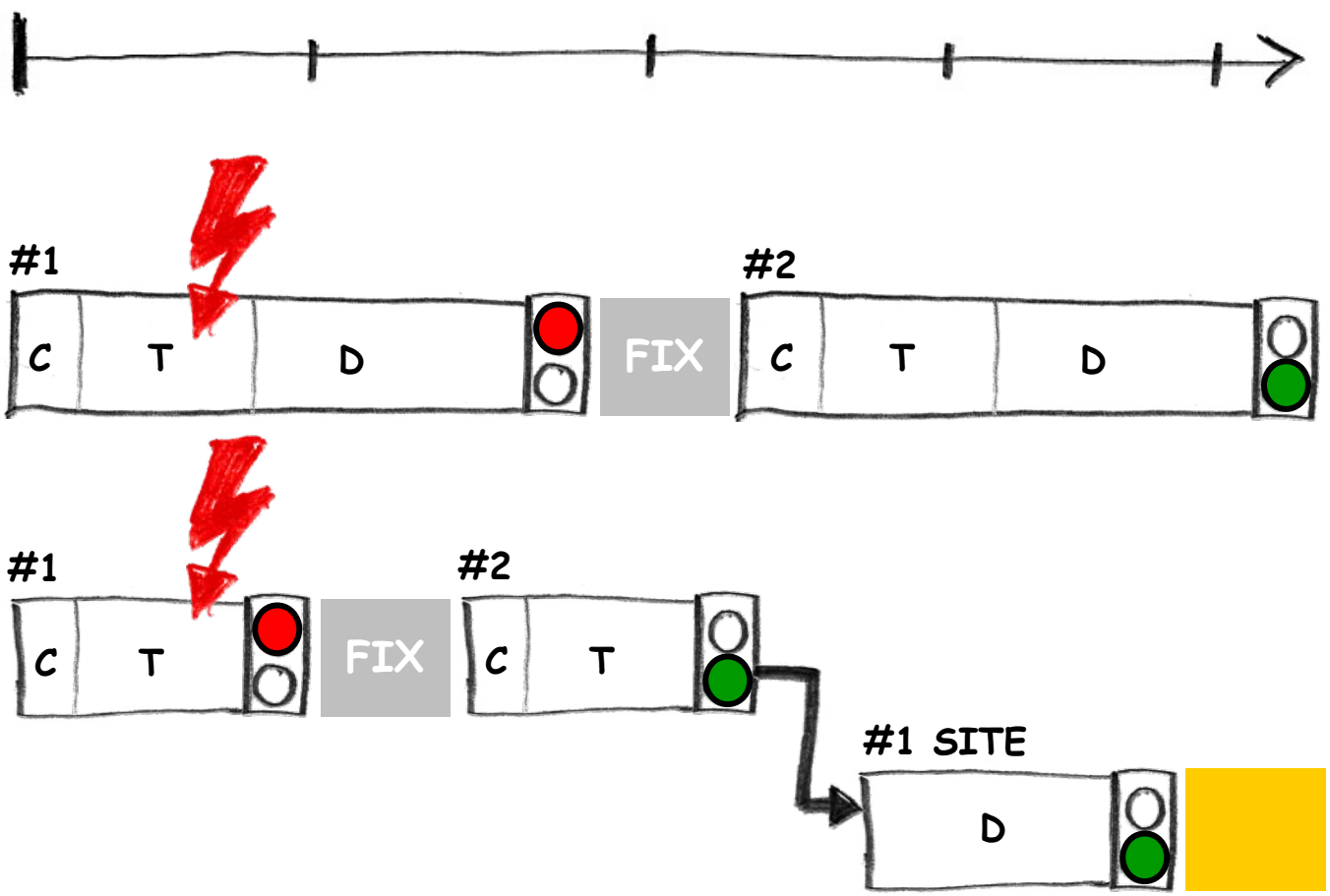
nachher



Staffeln: Beispiel

vorher

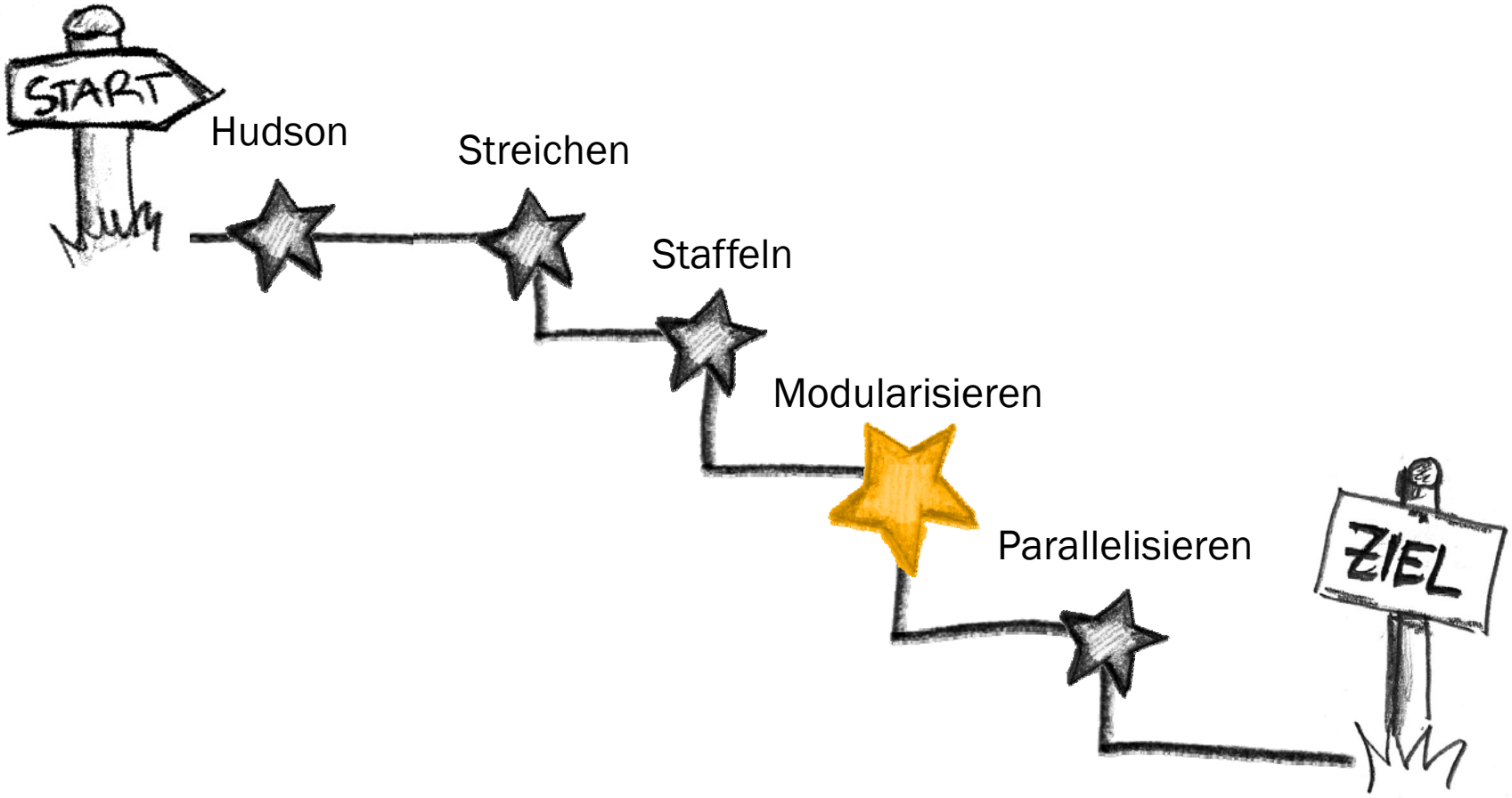
nachher



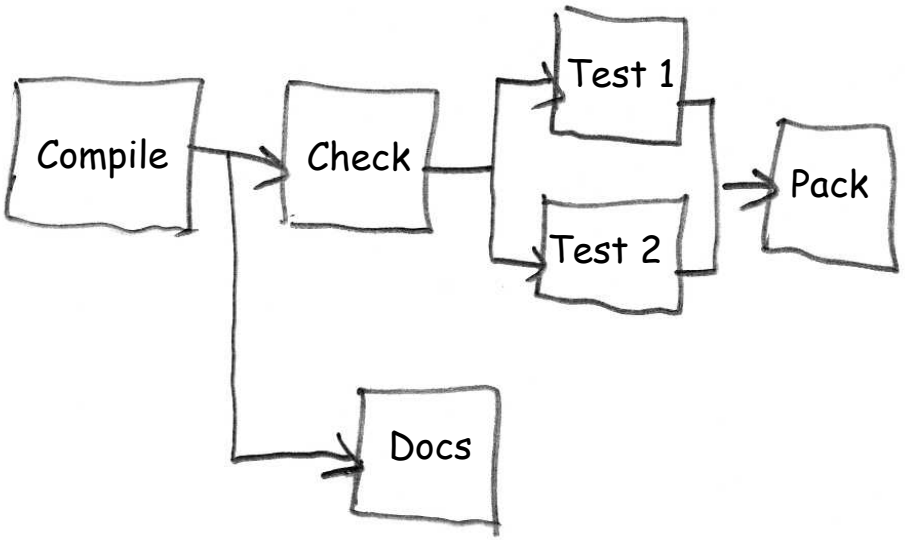
Staffeln: Wie unterstützt hier Hudson?

- Projektbeziehungen (vor-/nachgelagerte Projekte)
- Speicherung von Fingerabdrücken
- Nachvollziehbarkeit des Buildprozesses über verknüpfte Projekte hinweg

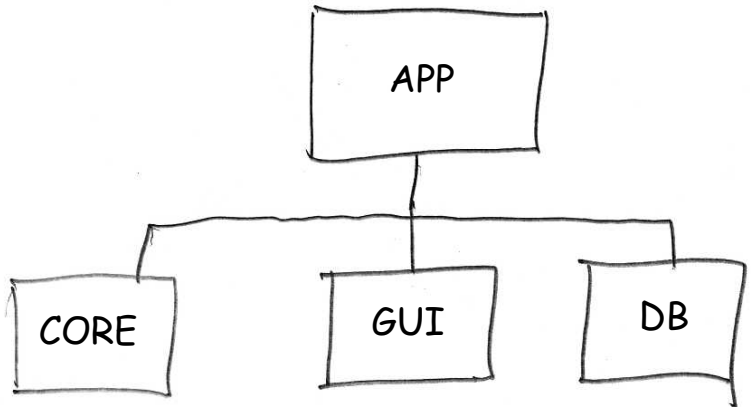
Agenda


START
Hudson
Streichen
Staffeln
Modularisieren
Parallelisieren
ZIEL

Die nächste Ebene: Wir betrachten Module statt Phasen.



Abhängigkeiten zwischen **(Build-)Phasen**

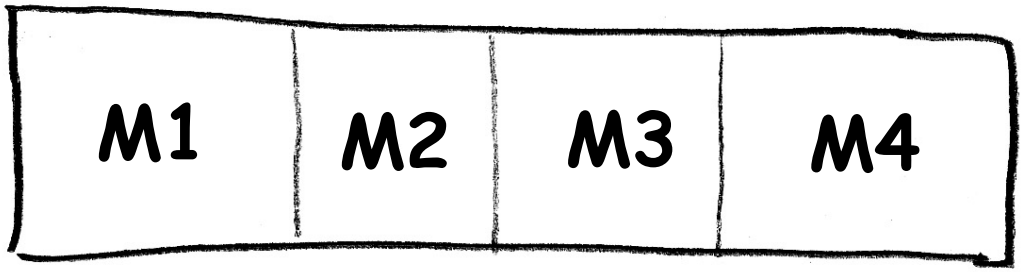


Abhängigkeiten zwischen **Modulen**

Modularisieren: Was lässt sich wiederverwenden?

#1

vorher



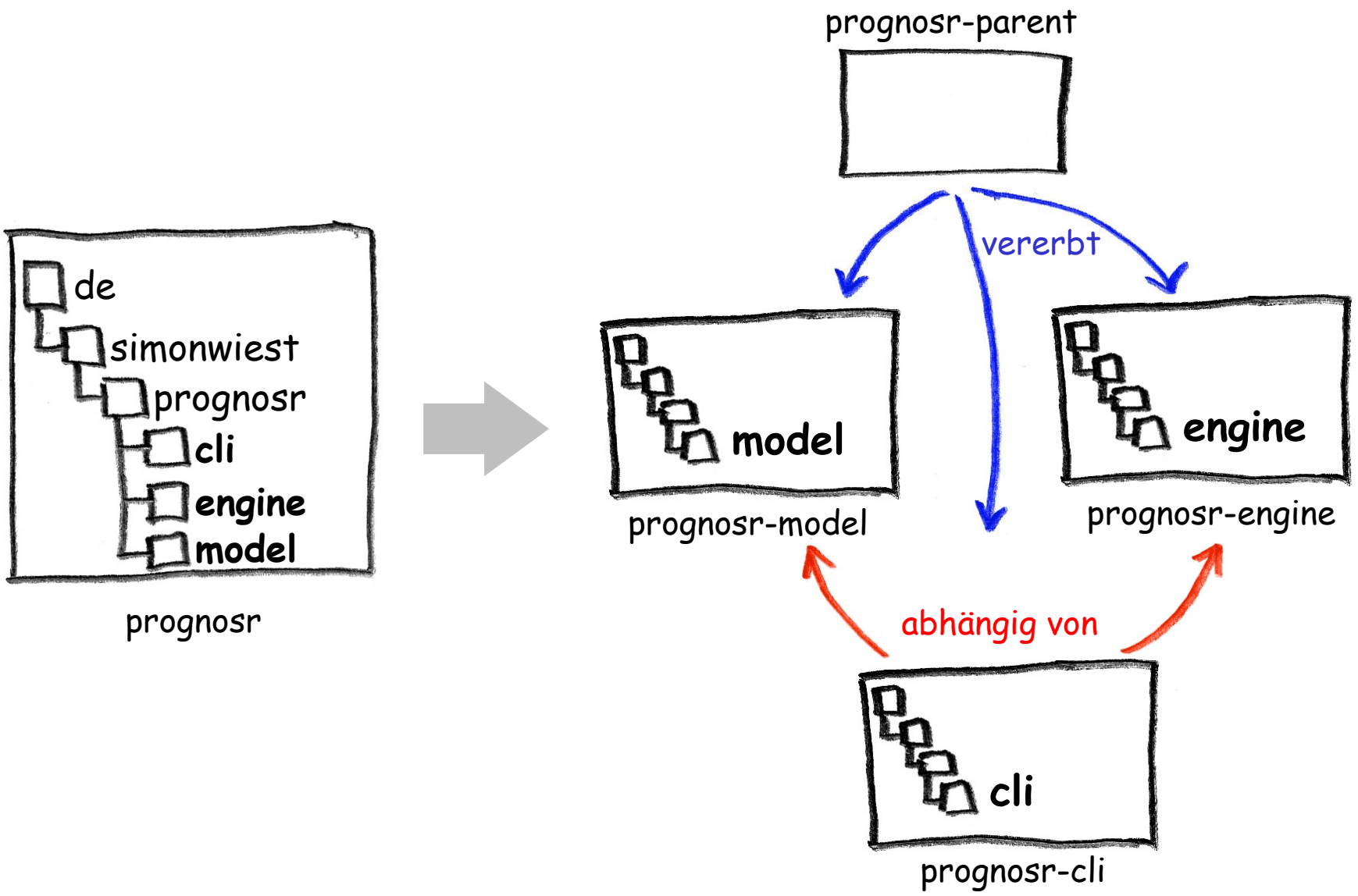
wiederverwenden

#2

nachher

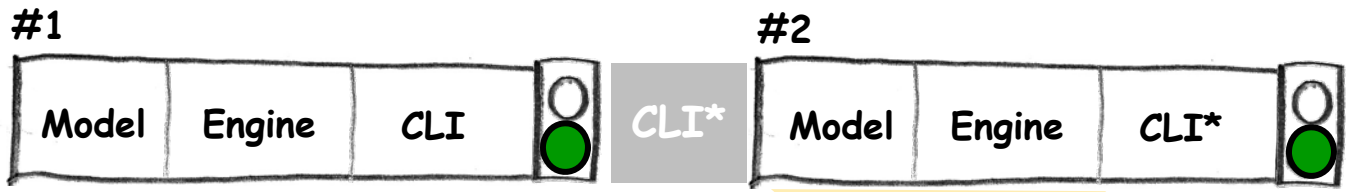
neu bauen

Modularisieren: Das Projektlayout wird verändert.

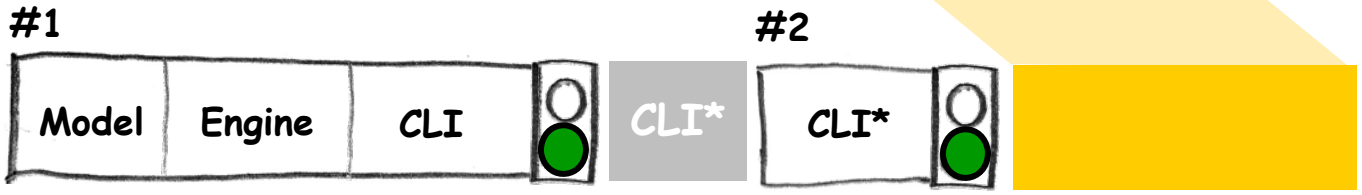


Modularisieren: Beispiel

vorher



nachher

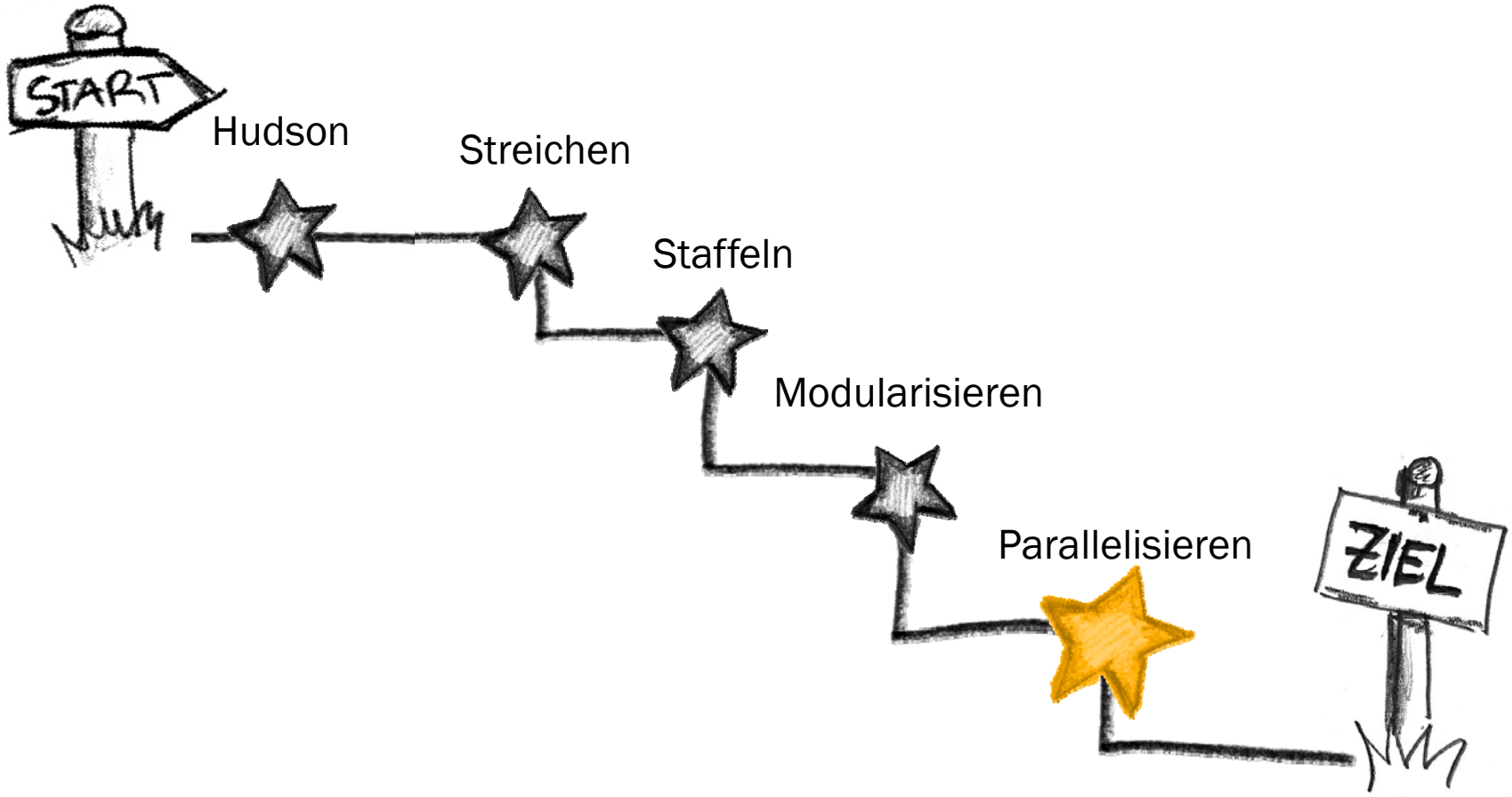


Voraussetzung: Artefakte müssen archiviert werden, z.B. in Maven Repository.

Modularisieren: Wie unterstützt hier Hudson?

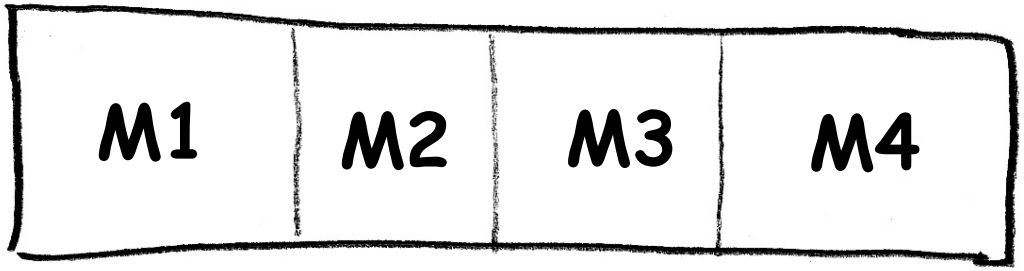
- Direkte Unterstützung des Maven-Modulkonzeptes
- Verwendungsnachweis der Build-Ergebnisse
- Filterung der angezeigten Projekte über reguläre Ausdrücke in Ansichten

Agenda

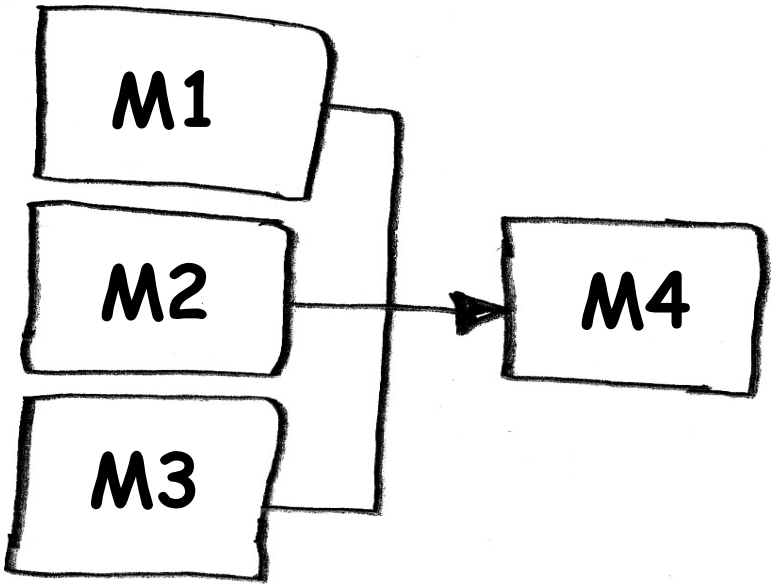

START
Hudson
Streichen
Staffeln
Modularisieren
Parallelisieren
ZIEL

Parallelisieren: Was lässt sich gleichzeitig bauen?

vorher

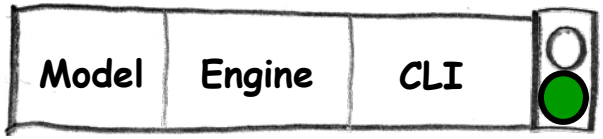


nachher

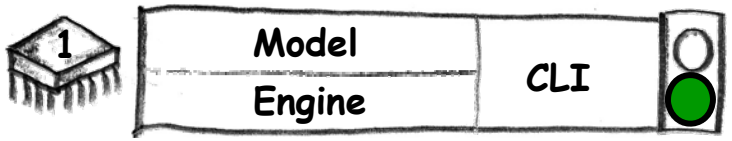


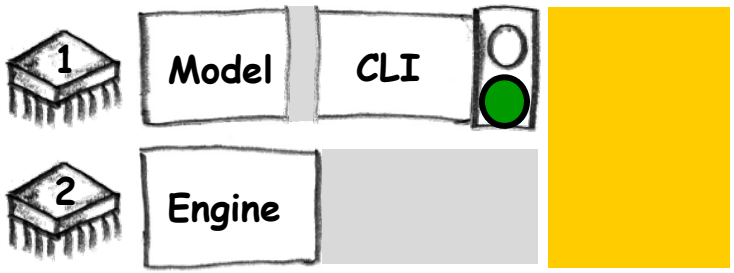
Parallelisieren: Beispiel

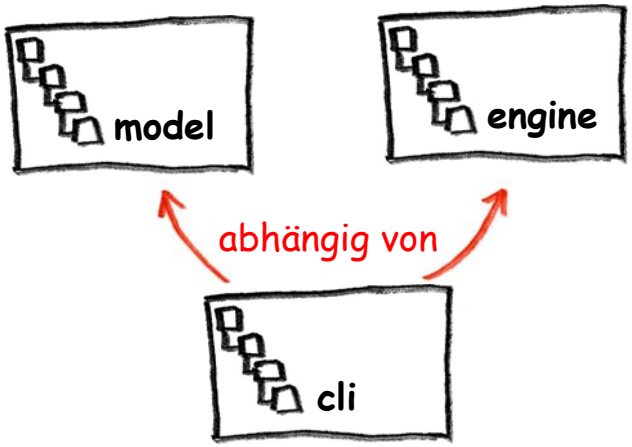
vorher



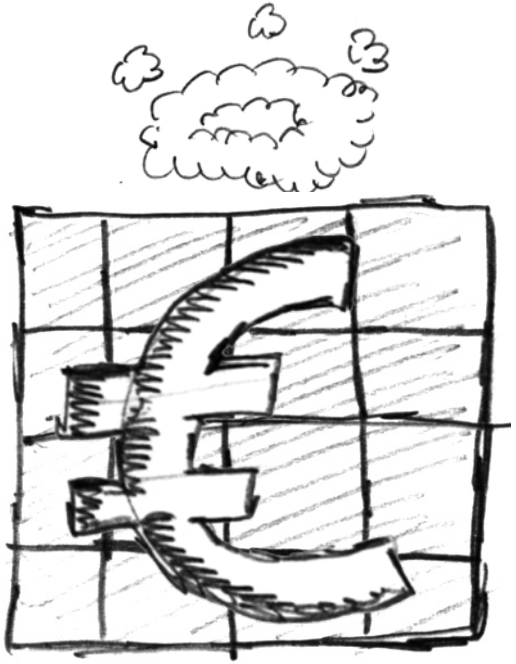
nachher



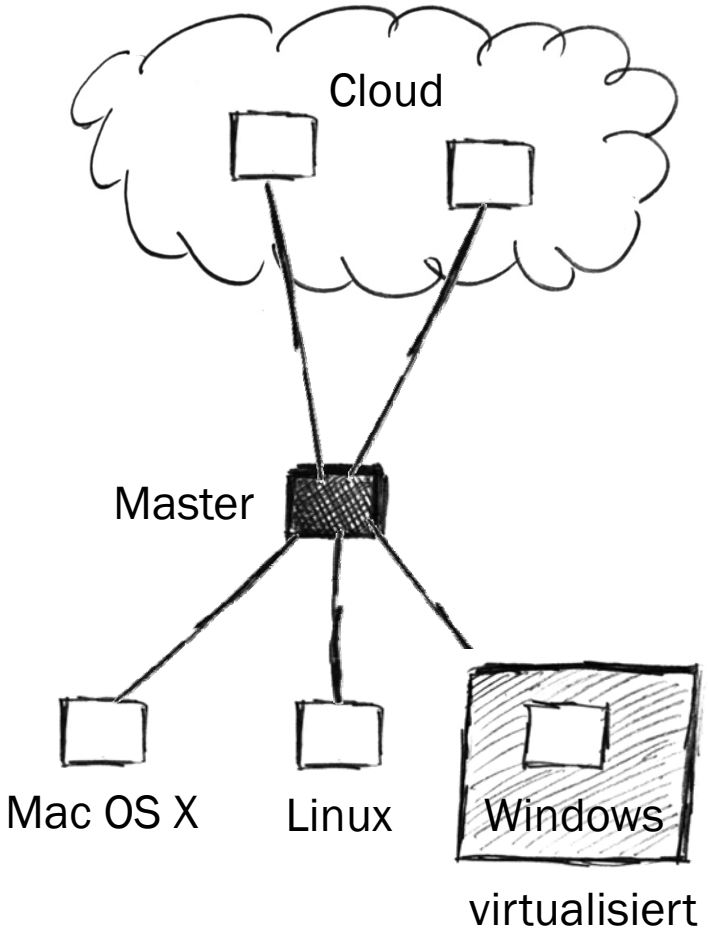




Parallelisieren: Verteilte Builds sind meist wirtschaftlicher.



lokale Builds

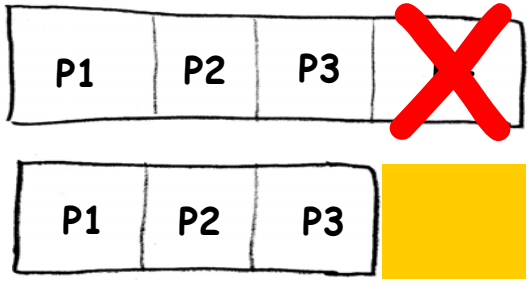


verteilte Builds

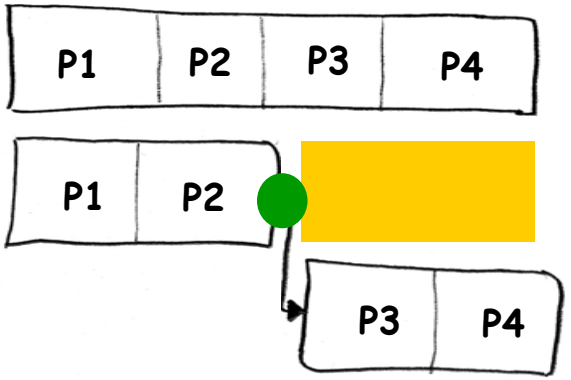
Parallelisieren: Wie unterstützt hier Hudson?

- Master-/Slave-Architektur
- Einfache Konfiguration von Knoten
- Automatischer Start/Stopp von Slave-Knoten
- Zusammenfasste Darstellung der Ergebnisse verteilter Builds
- Breite Unterstützung von Betriebssystemen und Startmechanismen

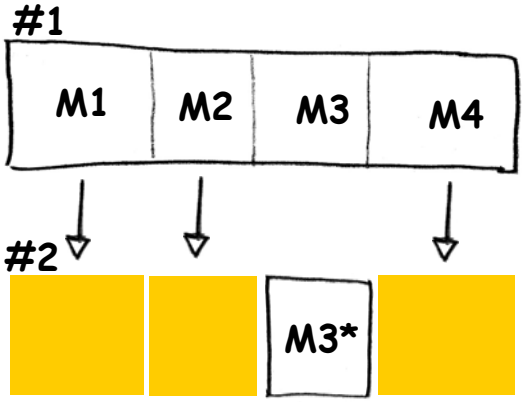
Die vier Strategien zusammengefasst:



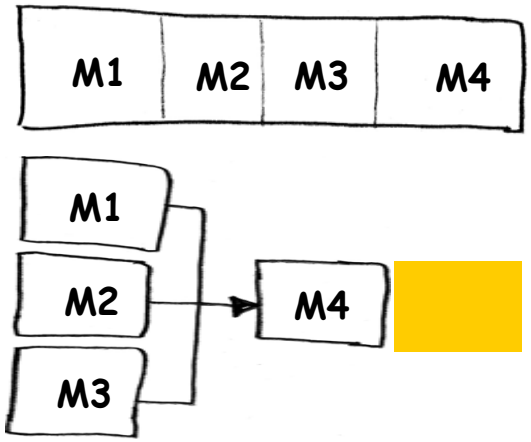
Streichen



Staffeln

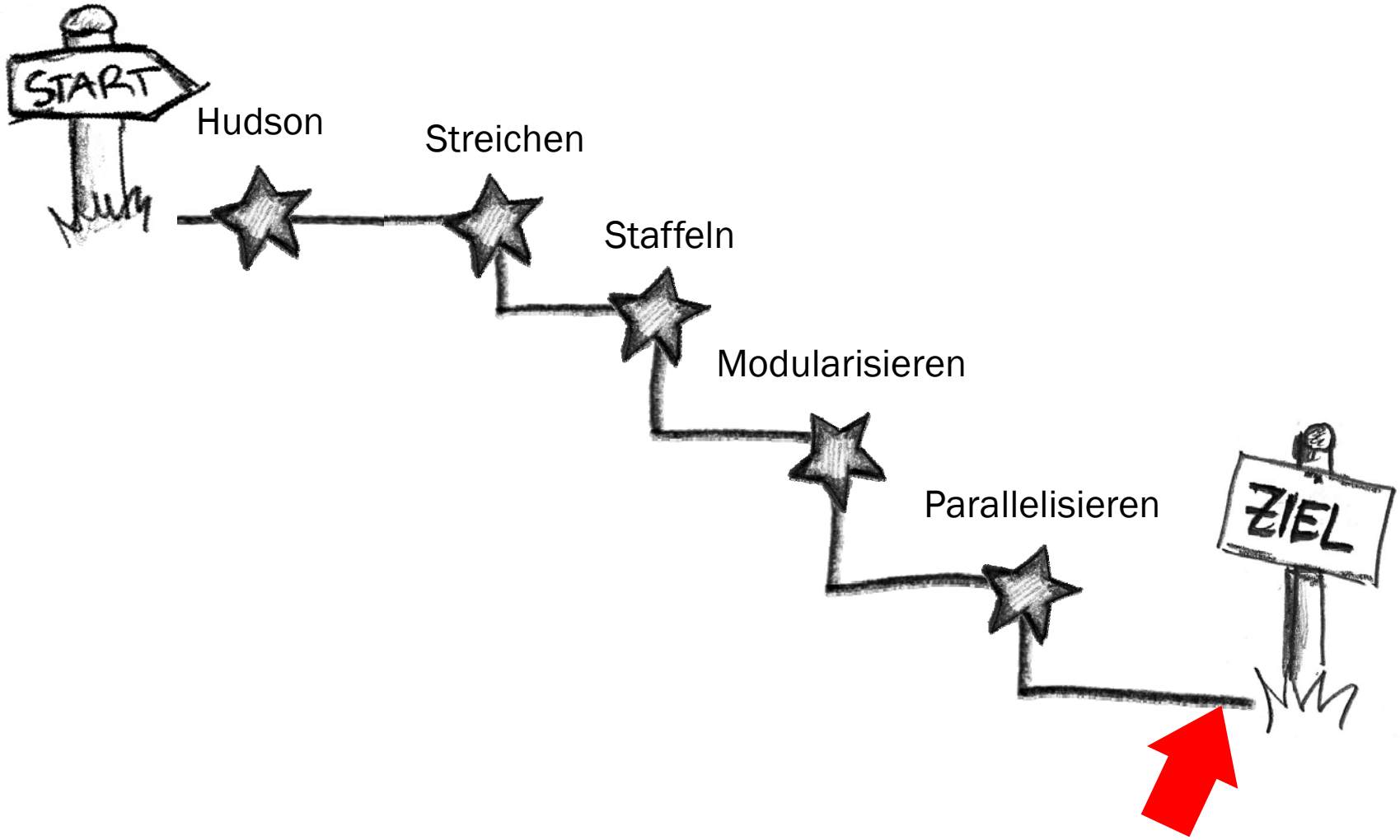


Modularisieren



Parallelisieren

Agenda


START
Hudson
Streichen
Staffeln
Modularisieren
Parallelisieren
ZIEL

HERUNTERLADEN

EINSETZEN

WEITERSAGEN

MITMACHEN

DR.WIEST
SOFTWARETECHNIK

hudson.dev.java.net

Vielen Dank fürs Zuhören.

Dr. Simon Wiest

Ingenieurbüro für Softwaretechnik
Wiesfleckenstrasse 13
72810 Gomaringen

www.simonwiest.de

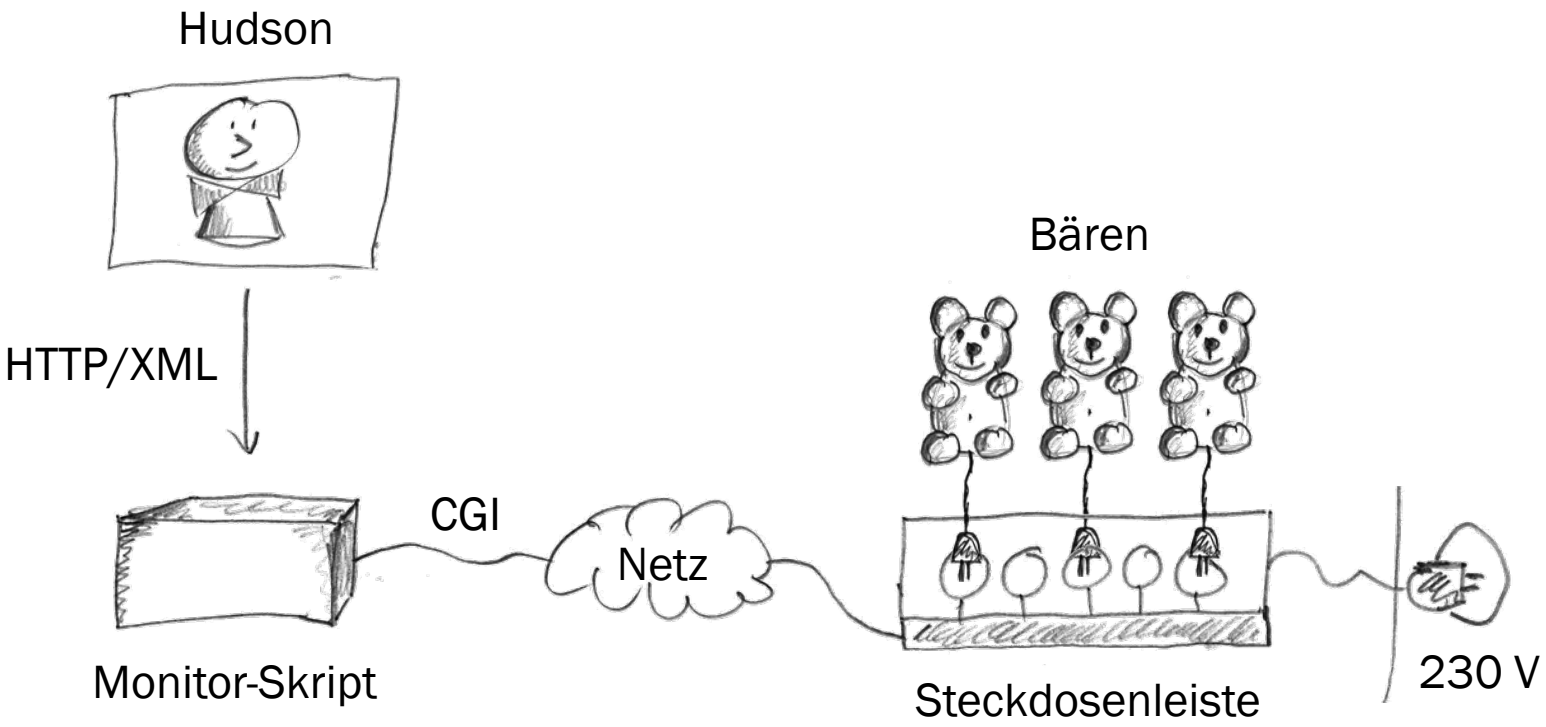
BACKUP

Empfohlene Nachlese

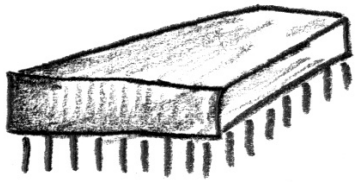
- Hudson
  - Projektseite. hudson.dev.java.net
  - JBoss Hudson CI Server. hudson.jboss.org/hudson
  - Interview mit Hudson-Initiator Kohsuke Kawaguchi
    blogs.sun.com/glassfishpodcast (Episode #007)
- Continuous Integration
  - White Paper von Martin Fowler. www.martinfowler.com
  - CI Feature Matrix. confluence.public.thoughtworks.org
  - P.M. Duvall: Continuous Integration. 2007.
- Projektautomatisierung (allgemein)
  - M. Hüttermann: Agile Java-Entwicklung in der Praxis. 2007.
  - G. Popp: Konfigurationsmanagement mit SVN, Maven, Redmine. 2009.
  - J. F. Smart: Java Power Tools. 2008.

Wie funktionieren die Bären?

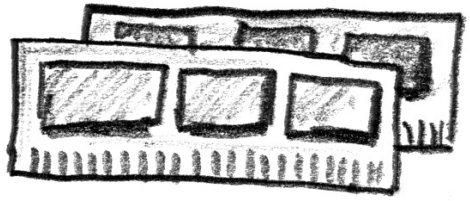
Mehr dazu im Hudson Wiki
wiki.hudson-ci.org//x/AQA1AQ



Bonus-Folie: Hardware aufrüsten lohnt.



CPU



Hauptspeicher

Festplatte

Netzwerk

Quelle: „Schneide Blog“ der Softwareschneiderei GmbH, http://schneide.wordpress.com

Cloud: Verteilung, Virtualisierung und Rechenzeit auf Abruf