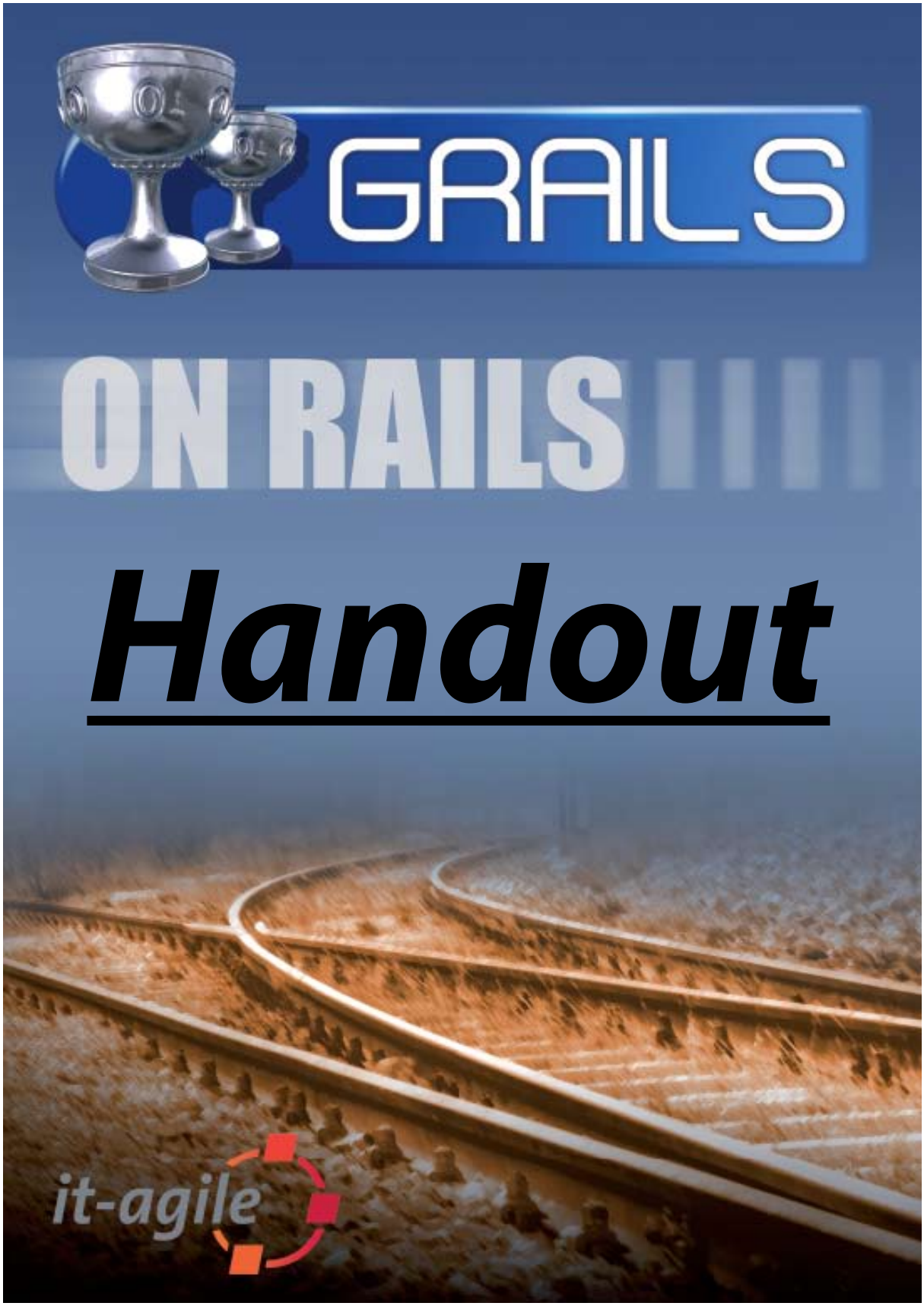

GRAILS
ON RAILS
Handout
it-agile

Grails in zwei Minuten

*Grails ist **das** Framework, um hochproduktiv Webanwendungen mit etablierten Java-Technologien zu entwickeln. Als Programmiersprache wird dabei Groovy eingesetzt, die standardisierte dynamische Skriptsprache für die Java-Plattform. Der Groovy-Quellcode wird in Java-Bytecode compiliert, wodurch der Betrieb auf der Java-Plattform sichergestellt ist. Darüber hinaus punktet Groovy auch bei der Integration, denn Groovy- und Java-Code lassen sich leicht mischen. Aufrufe in beide Richtungen sind problemlos möglich. Dadurch können alle Java-Libraries auch aus Groovy und damit unter Grails verwendet werden. Außerdem kann existierender Java-Code leicht in Grails-Projekte eingebunden werden.*

In Grails wird Hibernate als OR-Mapper eingesetzt, Spring für Dependency-Injection und das Transaktionshandlung sowie Quarz als Job-Scheduler. Mit Grails lassen sich sehr einfach „AJAXifizierte" Webanwendungen erstellen. Und das mächtige Plugin-System bietet bereits jetzt über 100 Plugins für eine Reihe von Aufgaben, die Sie nicht mehr selber lösen müssen.

Grails fügt den etablierten Java-Technologien kaum etwas hinzu, sondern widmet sich im Wesentlichen der Aufgabe, diese etablierten Technologien pfiffig zu verbinden. Daher sind Performance-Bedenken auch nicht angebracht. Natürlich ist Groovy als dynamische Sprache in der Ausführung langsamer als Java. Da aber letztlich nur ein sehr geringer Anteil von Grails-Anwendungen auf Groovy basiert, fällt das nicht weiter ins Gewicht. Ähnlich verhält es sich mit der Skalierung der Anwendung: Grails-Anwendungen lassen sich genauso wie Java-Anwendungen skalieren. Allenfalls ist mit Grails die Wahrscheinlichkeit geringer, versehentlich Bottlenecks in die Anwendung einzubauen, die die Skalierung verhindern.

Anwendungsbereich und Vorteile von Grails

Grails eignet sich für die Entwicklung von Webanwendungen für Internet und Intranet. Dabei folgt Grails dem Paradigma „Konvention statt Konfiguration" und spart dadurch Aufwände bei der Entwicklung ein. Es gibt Anwendungen, bei denen der Code der Grails-Anwendung kürzer ist als das, was man in einer vergleichbaren Java-Anwendung an XML-Konfiguration erstellen muss.

Grails basiert auf der Java-Plattform und etablierten Java-Technologien. Daher ist dieses Framework relativ einfach zu erlernen, wenn bereits Java-Know-How vorhanden ist. Für den Betrieb der Anwendungen gilt sogar, dass gar nicht umgelernt werden muss, denn Grails-Anwendungen werden genau so betrieben wie Java-Anwendungen auch: mit einer Servlet-Engine wie Tomcat oder Jetty. Was immer Sie heute zum Betrieb ihrer Java-Webanwendungen einsetzen, funktioniert auch mit Grails - ein einigermaßen aktuelles JDK vorausgesetzt.

Unsere Erfahrungen

Wir haben bisher drei Internet-Anwendungen mit Grails entwickelt. Interne Analysen haben dabei gezeigt, dass die Entwicklung der ersten Grails-Anwendung etwas aufwändiger war als es in Java der Fall gewesen wäre (6% mehr Aufwand). Wir führen das auf das initiale Erlernen der Technologie sowie einige Kinderkrankheiten zurück, die es damals (2006) in Grails gab. Inzwischen sind wir mit Grails aber produktiver als mit Java - um ca. 20%.

Zum einen haben wir heute geringere Lernaufwände. Wir konnten das letzte Projekt nicht vollständig mit Grails-erfahrenen Entwicklern bestücken. Allerdings war ein erfahrener Grails-Entwickler dabei und hat als Mentor fungiert - ein Verfahren, dass gut funktioniert hat. Zum anderen sind die meisten Kinderkrankheiten in Grails heute beseitigt.

Detaillierte Informationen zu unseren Erfahrungen mit Grails finden Sie im Anhang.

Fazit

Wie bei jeder neuen Technologie fallen am Anfang Lernaufwände an. Ist diese Investition aber einmal getätigt, kann man mit Grails produktiver Webanwendungen entwickeln als in Java. Dabei können Entwickler und der Betrieb viel von ihrem Java-Wissen im Grails-Umfeld wiederverwenden. So muss man sich auch keine Sorgen machen, dass irgendeine Aufgabe in Grails nicht lösbar wäre. Wenn sie in Java lösbar ist, ist sie auch mit Grails lösbar.

Trotz der oben genannten Herausforderungen können wir Grails für die Entwicklung von Webanwendungen uneingeschränkt empfehlen.

Fragen zu Groovy und Grails?

Sprechen Sie uns an oder schreiben Sie eine E-Mail an unsere Experten:

- *bernd.schiffer@it-agile.de*
- *stefan.roock@it-agile.de*

Die Referenten

*Dipl.-Inform. **Stefan Roock** ist Senior IT-Berater bei it-agile in Hamburg. Er verfügt über mehrjährige Erfahrung aus agilen Softwareprojekten (Scrum, eXtreme Programming, Feature Driven Development) als Projektleiter, Coach, Trainer, Scrum-Master/Facilitator und Entwickler. Darüber hinaus hat er zahlreiche Artikel und Tagungsbeiträge über Agile Softwareentwicklung verfasst und ist Autor der Bücher „Software entwickeln mit eXtreme Programming" und „Refactorings in großen Softwareprojekten".*

*Dipl.-Inf. **Bernd Schiffer** ist Agiler Berater bei it-agile in Hamburg. Er hat mehrjährige Erfahrung aus agilen Softwareprojekten (vor allem: eXtreme Programming, Scrum) als Entwickler, Projektleiter, Coach und Trainer, sowie als Sprecher auf Konferenzen. Darüber hinaus interessiert er sich für TDD, Groovy, Grails, DSLs und guten Code. Er studierte Informatik an der Universität Bremen.*

it-agile

Wir entwickeln Individualsoftware mit Groovy und Grails oder klassisch in Java. Dabei legen wir besonderen Wert darauf, dass der Kunde auch wirklich das System bekommt, das er braucht - und zwar möglichst schnell. Deshalb stimmen wir uns auch während der Entwicklung immer wieder eng mit dem Auftraggeber ab und unterstützen ihn ggf. auch bei der Definition und Formulierung der Anforderungen.
Zu unseren Kunden gehören u.a. das Land Bayern, die Berliner Wasserbetriebe, HDI Gerling, hama und Signal Iduna.
Außerdem unterstützen wir unsere Kunden mit erfahrenen Agilen Entwicklern und bieten Schulungen und Coaching zu Grails sowie verschiedenen agilen Themen wie Scrum, Lean Software Development, testgetriebene Entwicklung oder flexible Architekturen.
Für eine komplette Referenzliste und ausführliche Projektbeispiele besuchen Sie unsere Webseite, oder schreiben Sie uns eine E-Mail: info@it-agile.de

it-agile

Anhang A: Erfahrungen

Die Programmiersprache Groovy

Mit Java-Kenntnissen sind die ersten Schritte mit Groovy einfach. Die Groovy-Syntax ist Java sehr ähnlich. Java-Code lässt sich daher auch mit wenig Aufwand in Groovy-Code transformieren. Damit nutzt der Code noch lange nicht alle Groovy-Features und ist aus Groovy-Sicht auch unnötig kompliziert. Aber immerhin läuft schon mal etwas. Jetzt kann man sich schrittweise daran machen, die fortgeschrittenen Groovy-Konzepte zu verstehen und anzuwenden. Code im Groovy-Style ist deutlich kompakter und ausdrucksstärker als Java-Code: Mit Groovy brauchten wir weniger Code, der auch noch besser lesbar war.
In einfachen aber realitätsnahen Beispielen haben wir bzgl. des reinen Codes eine Ersparnis von 25-30% festgestellt bzgl. Lines of Code (LoC). Noch interessanter als der reine LoC-Vergleich ist allerdings eine Betrachtung der Kontrollstrukturen: In Groovy-Code findet man nur einen Bruchteil der Schleifenkonstrukte (for, while), die man in vergleichbaren Java-Anwendungen hätte.

Entwicklungsumgebung

Die Unterstützung für Groovy in Entwicklungsumgebungen ist bisher nicht so ausgefeilt wie man das von Java gewohnt ist. Immerhin bieten IntelliJ Idea, Eclipse und Netbeans Groovy/Grails-Plugins, die kontinuierlich verbessert werden. Die umfangreichste Unterstützung findet sich heute in IntelliJ Idea - inkl. Syntax-Highlighting, Auto-Vervollständigung, Cross-Referencing und einiger Refactorings.

Konvention statt Konfiguration

Am Anfang ist das Arbeiten mit Konventionen statt Konfiguration gewöhnungsbedürftig, und man muss auch erst mal verstehen, wie die Konventionen funktionieren. Dann sieht man aber schnell den Vorteil: Man kann auf hunderte oder tausende Zeilen von Konfigurationscode verzichten. Und wenn man doch mal von den Konventionen abweichen muss, ist das möglich: Letztlich basiert ja alles auf Spring, und man kann auch manuell konfigurieren.

Technologien

Die Verwendung etablierter Java-Technologien bringt viele Vorteile mit sich. Allerdings versteckt Grails diese nicht vollständig. Insbesondere in Fehlersituationen wird man mit den grundlegenden Technologien wie Hibernate oder Spring konfrontiert. Und dann sind grundsätzliche Kenntnisse dieser Technologien auf jeden Fall hilfreich.

AJAX

Die Verwendung von AJAX mit Grails ist sehr einfach und klar.

REST

Der REST-Architekturstil ließ sich in unseren Projekten mit Grails besonders leicht umsetzen. Die Konfiguration der REST-URLs ist mit der URL-Mappings-DSL ganz einfach, und Grails bringt Hilfsklassen mit, um Objekte nach XML oder JSON und zurück zu wandeln.
So bietet eine unserer Anwendungen eine REST-Schnittstelle, damit sie von einem anderen System des Kunden mit Daten befüllt werden kann.

Unit-Tests und testgetriebene Entwicklung

Groovy und Grails bringen bereits von Haus aus Unterstützung für Unittests mit. In den frühen Grails-Versionen dauerte die Ausführung der Tests relativ lange, so dass insbesondere testgetriebene Entwicklung behindert wurde. Inzwischen ist dieses Problem durch den „interactive mode" von Grails deutlich entschärft worden.

Agile Entwicklung

Grails eignet sich sehr gut für agile Projekte: Man kann schnell erste Versionen der Anwendung zeigen und diese schrittweise weiterentwickeln. Die Entwicklung geht sogar so schnell, dass man für Teilbereiche direkt den Fachexperten neben sich sitzen haben kann - z.B. wenn es um die Gestaltung der Oberflächen geht.

Anhang B: Grails-Projekte (Auszug)

Unter anderem wurden die folgenden Internet-Anwendungen mit Grails entwickelt:

1. *WerKannWann (Terminfindung): http://www.werkannwann.de*
2. *kaufDa (Produktsuche): http://www.kaufda.de*
3. *Blerp (Diskussion über Web-Sites): http://www.blerp.com/*
4. *CITYPendler (Car-Sharing): http://www.citypendler.de/*
5. *Team-Spider (Team-Assessment): http://team-spider.it-agile.de*
6. *Lean-to (Projekttracking): http://www.lean-to.com/*
7. *SplendiCon (Konferenzsystem): http://www.splendicon.net/*
8. *TwitCaps (Twitter-Bilder): http://twitcaps.com/*
9. *Empora (Shopsystem): http://www.empora.com/women/*

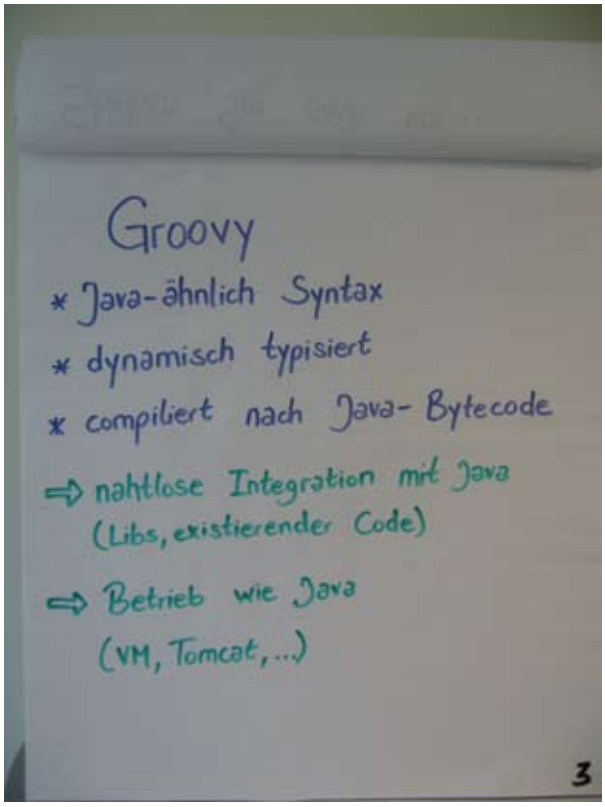
Eine vollständige Liste findet sich unter http://grails.org/Success+Stories

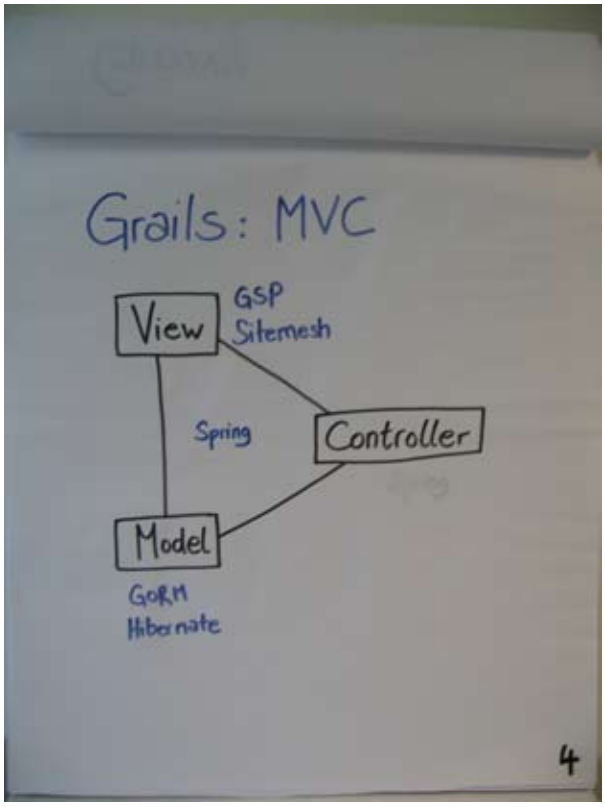
Fragen zu Groovy und Grails?

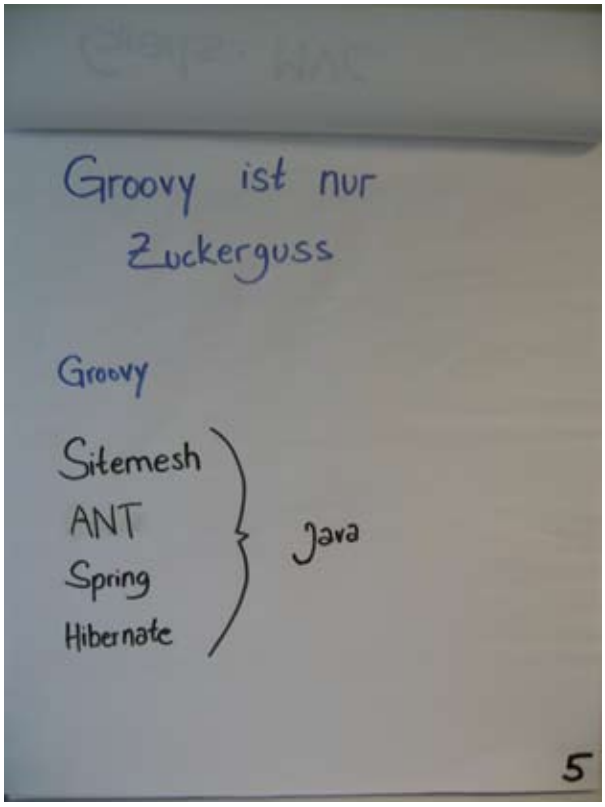
Sprechen Sie uns an oder schreiben Sie eine E-Mail an unsere Experten:

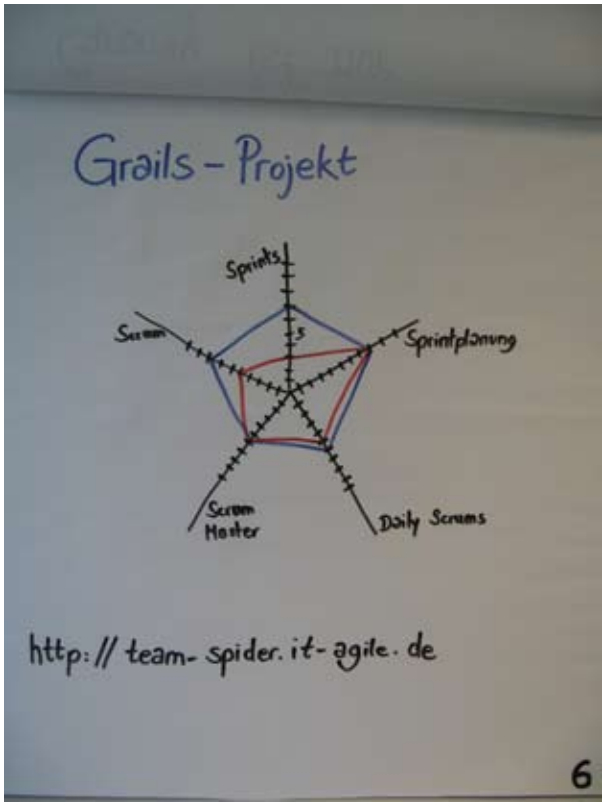
- *bernd.schiffer@it-agile.de*
- *stefan.roock@it-agile.de*

Anhang C: „Folien“ des Vortrages





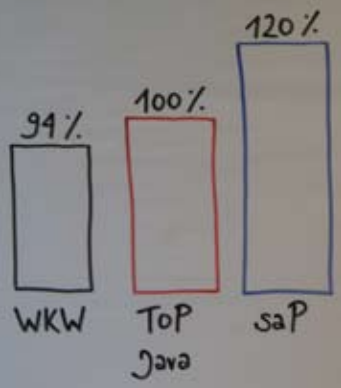

Groovy ist nur
Zuckerguss
Groovy
Sitemesh
ANT
Spring
Hibernate
Java
5


Grails - Projekt
Sprints
Sprintplanung
Daily Scrums
Scrum Master
http://team-spider.it-agile.de
6


Grails - Projekt
werkannwann.de
Freitag
14.
April 2000
7


Grails - Projekt
LfU saP
8

Produktivität



9

Grails - Gesamt

Grails A	Grails B
* nur Theoriewissen und aus einfachen Beispielen	* Entwickler aus Grails A im Team
* Kinderkrankheiten	* Kinderkrankheiten weitgehend beseitigt

10

Groovy

Grails A	Grails B
* vorher erste Groovy-Erfahrungen	* vorher erste Groovy-Erfahrungen
* Java-Libs wiederverwendet (z.B. E-Mail)	* Java-Libs wiederverwendet (z.B. E-Mail)
* unbrauchbarer IDE-Support	* nur Editor verwendet

11

Konventionen

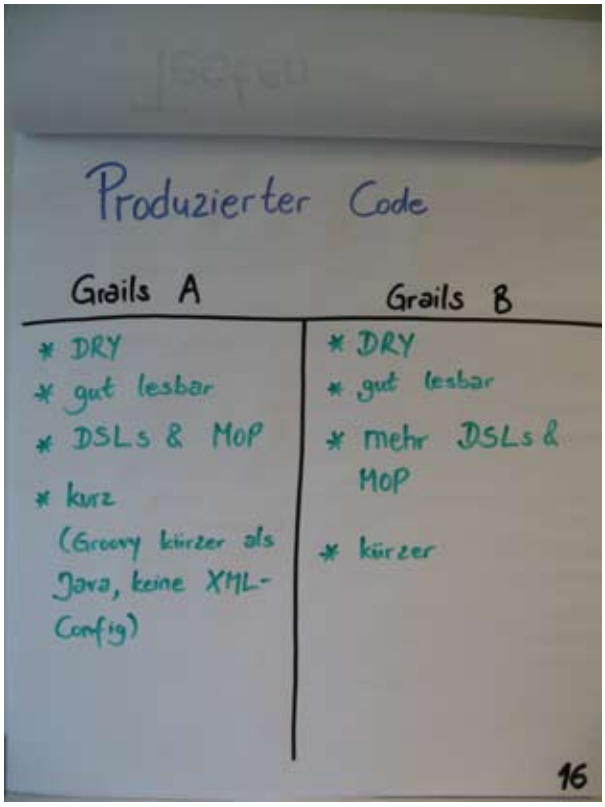
Grails A	Grails B
* Fehlerfälle wg. Unkenntnis	* unproblematisch
* schwer zu debuggen	* produktivitäts-steigernd
* weniger Code	* weniger Code
* geringere Kopplung	* geringere Kopplung

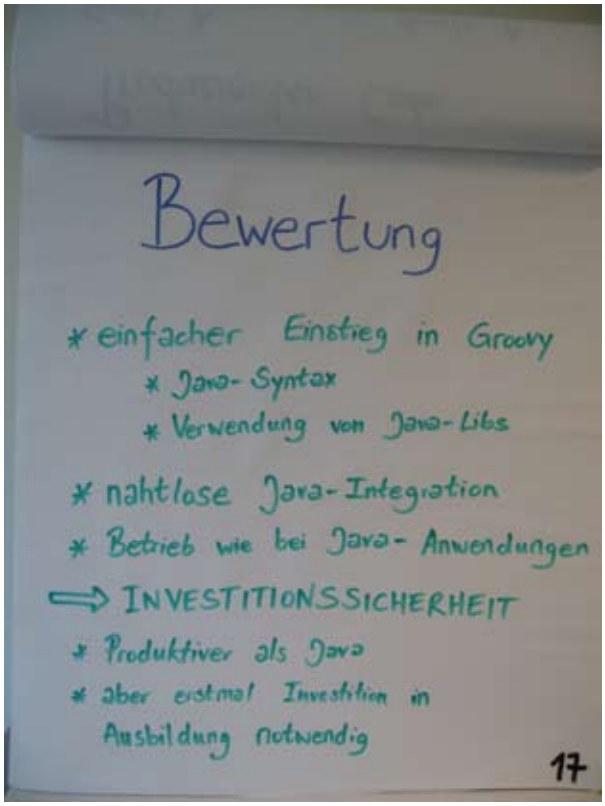
12


Technologiestack
Grails A
Grails B
* Technologien nicht vollständig versteckt
* Fehlersituationen z.T. Komplex, häufig Hibernate, Spring, Groovy im selben Stacktrace
* mehr Erfahrung erleichtert die Problemanalyse, aber es bleibt inhärent komplex
13


Externe Infrastruktur
Grails A
Grails B
* Infrastruktur (z.B. Hudson) bietet keine Groovy / Grails-Unterstützung
⇨ viel Handarbeit
* fast überall Groovy / Grails-Unterstützung vorhanden
14


Testen
Grails A
Grails B
* Testunterstützung eingebaut
* zu langsam für TDD
* Test Interface First
* durch „interactive mode" viel schneller
* Test Interface First auch für REST-Schnittstelle
15


Produzierter Code
Grails A
Grails B
* DRY
* gut lesbar
* DSLs & MoP
* kurz (Groovy kürzer als Java, keine XML-Config)
* DRY
* gut lesbar
* mehr DSLs & MoP
* kürzer
16



Fragen zu Grails-Projekten?

Wir verfügen über praktische Projekterfahrung mit Grails. Treten Sie mit uns in Kontakt, und wir überlegen gemeinsam, ob Grails für Ihre Projekte geeignet ist und wie ein praktikabler Einstieg/Umstieg aussehen kann!

040 88173 - 300
info@it-agile.de