Interview mit Graeme Rocher

von Marc-Oliver Scheele; Januar 2010 (Veröffentlicht auf jaxenter/Java Magazin)

Pünktlich zum vergangenen Weihnachtsfest wurde die neue Grails Version 1.2 veröffentlicht. (http://www.grails.org). Vor diesem Hintergrund hat Java Magazin Autor und Grails-Experte Marc-Oliver Scheele ein Interview mit Graeme Rocher geführt. Er ist Co-Gründer, Projektleiter und Lead-Developer des Grails-Frameworks.

Im Jahre 2006 veröffentlichte Graeme die erste Version 0.1 von Grails. Damals war es als CTO der Firma SkillsMatters beschäftigt. Er gründete mit anderen Groovy-Experten die Firma G2One, die sich auf Entwicklung und Beratung von Groovy- und Grails-Projekten spezialisierte. Die Firma wurde im Jahr 2008 von SpringSource übernommen, welche wiederum in 2009 von VmWare aufgekauft wurde und als eigenständige Division dort weitergeführt wird. Dort ist er als "Head of Grails Development" aktiv. Graeme Rocher ist darüberhinaus Autor des Grails-Standardwerkes "The Definitive Guide to Grails" und regelmäßiger Speaker auf Konferenzen.

Graeme, lass uns kurz die Zeit zurückdrehen. Erzähle bitte kurz was Deine Motiviation war das Grails Framework zu erschaffen? Hattest Du ein konkretes Projekt zu implementieren oder ist es als eine Art persönliches "Forschungsprojekt" entstanden?

Ich war CTO in einer Firma, die kundenspezifische Learning-Management (LMS)- und Content-Management(CMS)-Systeme entwickelte. Die Systeme wurden als maßgeschneiderte Lösungen verkauft, die Kosten für die Anpassungen der LMS und CMS schon inklusive. Mit diesen fixen Preisen mussten wir wirklich ultra-produktiv arbeiten, sonst hätten die Kosten für die Anpassungen den Gewinn minimiert.

Die Basis des LMS- und CMS-Systems wurde mit Spring und Hibernate gebaut, und obwohl mir Projekte mit Ruby on Rails und Django gefielen, ließen sich diese so gut wie nie auf die Umgebungen unserer Kunden anwenden. Allerdings hatte ich schon mal Groovy in einem anderen Projekt genutzt, genauer gesagt ging es da um ein Buildsystem für die Erstellung von eLearning-Inhalten aus Word-Dokumenten. Daher wollte ich sehen, ob man mit Groovy eine ähnliche Produktivität erreichen kann.

Wie sich herausstellte, gab es in der Groovy-Mailingliste Bestrebungen, ein Rails-ähnliches Framework zu entwickeln, also half ich dabei, das auf den

Weg zu bringen. Daraus entstand Grails. Heute habe ich immer noch Kunden, die CMS-Systeme basierend auf dieser ersten Grails Version 0.1 produktiv einsetzen. Mit der Zeit wurde Grails sehr populär, daher haben wir die Firma G2One gegründet, die später von SpringSource übernommen wurde.

Mittlerweile sind vier Jahre vergangen und Grails hat weite Verbreitung gefunden. Der "Unique Selling Point" gegenüber anderen Java basierten Frameworks ist der sehr geringe Entwicklungsaufwand, um anspruchsvolle Webapplikationen zu implementieren. Welches sind die aus Deiner Sicht beeindruckende Grails-Projekte in Sachen Entwicklungsgeschwindigkeit und Komplexität?

Klar! Es gibt viele Projekte da draußen, die Grails nutzen. In Großbritannien hat zum Beispiel der größte Runfunksatellitenanbieter Sky TV sein komplettes Sky.com nach Grails portiert – Sky TV gehört zu News Corporation, denen wiederum Fox in den USA gehört. Das Sky-Team hat ein nicht-triviales maßgeschneidertes CMS-System im Backend implementiert, das Content für über 150 Millionen Aufrufe pro Monat liefert.

Walmarts Gegenstück zum Apple iTunes Store ist ebenfalls in Grails geschrieben und ich glaube, auch hier gibt es erheblichen Traffic. Es gibt noch mehr große Namen wie zum Beispiel LinkedIn und natürlich viele viele Unternehmen, die es intern nutzen, weil es leicht ist, Grails-Applikationen in eine existierende Java-basierte Infrastruktur zu deployen.

Gerade ist die Version 1.2 von Grails erschienen. Was ist der Focus dieses Releases ? Welches sind Deine Top-3 Features dieser Version?

Der Fokus in diesem Release lag auf Stabilität und Geschwindigkeit. Wir haben die GSP-Rendering-Engine überarbeitet und es ist gut zu sehen, dass Grails seine Position als performantes Framework einer dynamischen Sprache auf dem Markt stärken konnte. Wir haben auch einige Build-Infrastruktur-Verbesserungen implementiert, zum Beispiel die Möglichkeit, JAR-Abhängigkeiten mit einer Dependency-Resolution-DSL zu spezifizieren, die auf Ivy basiert.

Auch GORM (der ORM-Layer gebaut mit Hibernate) hat Veränderungen erfahren – es gibt eine neue Syntax für dynamische Finder und die Möglichkeit, wieder verwendbare Named-Queries zu spezifizieren, die mit zusätzlichen Merkmalen verbunden werden können. Hierdurch können Queries wesentlich besser das DRY Prinzip unterstützen. Apropos DRY: wir haben ebenfalls globale Constraints und ORM-Mapping-Definitionen hinzugefügt, so dass man nun einfacher das Standardverhalten der GORM-

Domain-Klassen global spezifizieren kann. Abgesehen davon gab es Hunderte kleinere Features und Verbesserungen, die man natürlich in den Release Notes nachlesen kann.

Was steht auf der Roadmap für Grails? Welche Erweiterungen sind für die nächste Major-Version geplant?

Wir planen eine kurze Iteration für Grails 1.3 mit dem Ziel Groovy 1.7 zu unterstützen. Das sollte Ende März passieren. Dann wird man auch Grails-Plug-ins in Standard-Maven-Repositories veröffentlichen und aus diesen herunterladen können – das ist wichtig für Organisationen, die ihr existierendes Maven-Repository verwalten und kein neues Repository-Format einführen wollen – derzeit verlangt Grails das noch.

Für Grails 2.0 liegt der Fokus auf einer stärkeren Modularität. Dafür wird Grails echte Runtime-Modularität unterstützen, so daß Grails-Plug-ins zur Laufzeit gestartet, gestoppt und aktualisiert werden können. Wir werden uns dieser Sache in der zweiten Jahreshälfte 2010 annehmen.

Abgesehen davon wollen wir, dass Grails stabil bleibt und das Plug-in-Ökosystem weiter wächst. Einige Grails-Plug-ins sind schon in der Planung, es wird also ein spannendes Jahr.

Wie viele andere erfolgreiche Softwaresysteme ist Grails im Kern ein Plugin-System, welches insbesondere durch Plugins seine Funktionalität erreicht. Hast Du einen Überblick wieviele Plugins es aktuell gibt? Wieviele Core-Plugins gibt es, die direkt vom Grails-Team gepfelgt werden?

Momentan werden fast 350 Plug-ins im zentralen Grails-Repository gehostet, weitere sind extern verfügbar. Solche Plug-ins, die offiziell vom Grails-Team verantwortet werden, sind im Grails-Paket enthalten (Hibernate, Tomcat und Webflow), ebenso die Plug-ins, die von SpringSource unterstützt werden.

Die Qualität und Aktualität der Plugins schwankt stark. Gibt es Pläne Plugins in Qualitätsstufen zu unterteilen bzw. eine Art Kompatiblitätstests oder Zertifizierung zu realisieren?

Aus SpringSource-Perspektive sind solche Plug-ins „vertrauenswürdig", die wir unterstützen. Abgesehen davon gibt es viele weitere Plug-ins, die nicht offiziell unterstützt werden, die aber einen maßgeblichen Wert beisteuern. Es gibt gute Argumente ein Zertifizierungsprogramm für Grails-Plugins zu kreieren. Wir werden uns auch hiermit 2010 beschäftigen.

Seit einem Jahr gehört das Grails-Framework zur SpringSource Familie. Welche Synergien gibt es und inwiefern profitiert die Grails Community davon?

Ich glaube, die Grails-Community erkennt immer mehr den Zusammenhang, wie SpringSource nun für jeden Schritt des Entwicklungszyklus eine Lösung anbieten kann – angefangen beim Build (SpringSource Tool Suite) bis hin zum Deployment (Tomcat / tc Server).

Dank der Expertise von SpringSource können wir im Grails-Team nun Bugfixes viel schneller lösen, da wir jetzt offen mit den Mitgliedern, z.B. den Core-Spring- und Tomcat-Teams, zusammenarbeiten können. Außerdem hat Grails nun mehr Einfluss auf die Richtung, in die sich Projekte entwickeln, die auf Grails aufbauen. Wenn das Grails-Team zum Beispiel ein bestimmtes Feature in Spring oder Tomcat braucht, ist es jetzt leichter, dieses Feature „on demand" implementiert zu bekommen.

Wie läuft die Zusammenarbeit mit den anderen SpringSource-Teams? Gibt es beispielsweise regelmäßigen Austausch, gemeinsame Releasepläne oder arbeitet jedes Team sehr autark?

Die Teams sind ziemlich autark, die Kommunikation untereinander wird durch regelmäßige Meetings gesichert. Der Releaseplan wird zusammen mit den anderen Projekten abgestimmt – beispielsweise war Grails 1.2 ja abhängig von dem Spring 3.0 Release. Wir mussten warten, bis Spring 3.0 final war, bevor wir mit Grails 1.2 herauskommen konnten.

Nach meinen Beobachtungen schreibst Du nach wie vor den größten Teil aller Grails Erweiterungen selber. Wieviele weitere wirklich aktive Submitter, neben Dir, gibt es noch? Ist die Unterstützung ausreichend oder würdest Du dir weitere Unterstützung aus der Community wünschen?

2009 war wirtschaftlich ein ziemlich hartes Jahr für die meisten Firmen, aber das Grails-Projekt hat glücklicherweise einige wichtige externe Beiträge bekommen. Lari Hotari hat viele Performanceverbesserungen beigesteuert, insbesondere ein Feature um GSP-Seiten vorzuübersetzen. Luke Daley hat die Testing-Infrastruktur umgeschrieben. Jeff Brown von SpringSource und ich haben dauerhaft am Grails Projekt gearbeitet – eine Sache die sonst nicht möglich ist, wenn man Open-Source Projekte nebenberuflich betreibt.

Also würde ich sagen, dass das Grails-Projekt 2009 ziemlich viel Community-Support hatte und ich hoffe, dass das 2010 so weitergeht. Hier bei SpringSource/VMWare suchen wir neue Kollegen und haben für dieses Jahr

im Grails-Projekt ein paar Stellen frei. Dadurch dürften sich die Vollzeit-Ressourcen für Grails dieses Jahr verdoppeln.

Was ist Deine Empfehlung für Entwickler, die sich an der Grails-Framework -Entwicklung beteiligen wollen? Welche Formen der Beteiligung gibt es und was muss geschehen, um Grails-Submitter werden zu dürfen?

Es gibt jede Menge Möglichkeiten, sich zu beteiligen. Entweder über das Submitten von Patches in unserem Issue Tracker, das Beisteuern eines Plug-ins oder einfach über das Beantworten von Fragen in der Mailingliste Unser Prozess zum Grails-Comitter ermuntert Entwickler ihre Beiträge via „Git Pull Requests" zu erstellen. Wenn die Anzahl und Qualität hoch genug ist, gewähren wir Comit-Rechte.

Das Grails/Groovy Team arbeitet weltweit verteilt. Als Head of Grails Development bist Du am entwickeln, abstimmen, promoten, beraten und wahrscheinlich noch vieles mehr... Du mußt ziemlich beschäftigt sein. Kannst Du Deine typische Arbeitswoche beschreiben? Von wo arbeitest Du? Wieviel Zeit bleibt Dir für die eigentliche Programmierung des Frameworks?

Mein Ziel ist es eigentlich, den SpringSource Service, also Consulting, Support und Trainings, zu unterstützen und nicht direkt diese Aktivitäten auszuüben. Dadurch habe ich mehr Zeit für die Entwicklungsarbeit, sprich das Arbeiten mit unseren Projektmanagern, um Ziele und Deadlines zu definieren, das Lösen von Grails spezifischen Fragen, das Entwerfen und Entwickeln von neuen Features und das Koordinieren des Grails-Teams intern und extern. Ich würde sagen, ich kümmere mich zu 80-90 Prozent meiner Zeit um die Entwicklungsarbeit, und den Rest investiere ich in Marketing-Aktivitäten, wie Konferenzteilnahmen, User-Groups-Treffen, Artikel schreiben und mit dem Marketing-Team und dem SpringSource-Service zu arbeiten.

Derzeit arbeite ich in Spanien, aber das Groovy/Grails-Team ist über Großbritannien, Frankreich, Deutschland und den USA verteilt.

Hat sich an der Arbeitsweise bzw. den Verpflichtungen irgendetwas geändert, seit der Akqusitation durch VmWare?

Nein, es hat sich nichts verändert. Das Unternehmen VMWare hat die Bedeutung der Entwickler-Community sehr gut verstanden und weiß, dass eine anhaltende Investition in die Spring- und Grails-Communities entscheidend ist. Wie schon erwähnt, stellen wir neue Leute für das Grails-Team ein, also war die Übernahme auf jeden Fall positiv für uns.

Welchen Mehrwert bringt VmWare als Muttergesellschaft?

Bei VMWare haben wir eine gewaltige Gelegenheit, eine nahtlose End-to-End virtualisierte Umgebung für das Entwickeln und Deployen von Anwendungen zu schaffen. Die Entwicklungsmethoden wurden in den letzten fünf Jahren durch Frameworks wie Spring und Grails enorm optimiert, während der Betrieb von Anwendungen fast der selbe geblieben ist. Es muss daran gearbeitet werden, in Punkto Betrieb dasselbe Level an Produktivitätssteigerung zu erreichen, damit den Leuten Zeit für Innovation bleibt. VMWare und SpringSource sind in einer Position, eine Umgebung zu schaffen, um virtualisierte Anwendungen zu bauen, zum Laufen zu bringen und zu managen. Ihr könnt euch auf ein paar ziemlich aufregende Produkte in nächster Zeit freuen.

Ein gewagter Blick in die Zukunft. Was ist Deine Vision für Grails im Jahr 2015? Wirst Du 2015 weiterhin "Head of Grails Development" sein?

Wer weiß, was in der Zukunft passiert, ich könnte von einem Bus überfahren werden! Aber ernsthaft, auch wenn wir in Punkto Optimierung des Entwicklungsprozesses weit gekommen sind, können noch viele weitere Vereinfachungen eingebaut werden. 2015 sehe ich Grails die Daten in NoSQL Datenbanken zu persistieren, Rich Internet Applikationen zu rendern und in eine virtualisierten Umgebung direkt lauffähig zu sein. .

Graeme, vielen Dank für das Gespräch!