Scala – Reif für Enterprise?

Dr. Halil-Cem Gürsoy
adesso AG
Dortmund

Schlüsselworte:

Scala, Java, JVM, Enterprise, Entwicklungsumgebungen

Einleitung

Vor zwei Jahren hat A. Blewitt in seinem Blog ein für die Scala Gemeinde recht ernüchterndes Fazit gezogen: „Scala is not enterprise ready“. Wen wundert es, sagten sich die Skeptiker, kommt die Sprache doch aus einem rein akademischen Umfeld. In der Zwischenzeit hat sich einiges in der Scala-Welt getan und Scala ist „angekommen“. In dem Vortrag möchte ich mit Hilfe verschiedener Aspekte beleuchten, ob die JVM-Sprache Scala und das Ökosystem um die Sprache nun in der Zwischenzeit den Reifegrad erreicht hat, der für den Einsatz in einem „Enterprise“-Projekt notwendig erscheint.

Vorgeschichte

Im Jahr 2009 sorgte Alex Blewitt mit seinem Blog *„Is scala ready for the Enterprise ?“* (http://alblue.bandlem.com/2009/08/is-scala-ready-for-enterprise.html) für Aufsehen, in dem er, mit einem offen vorgetragenen Sarkasmus, bezweifelte, dass die JVM-Sprache für den Einsatz in Enterprise-Projekten überhaupt geeignet ist. In diesem Blog führte er mehrere Punkte auf, die nach seiner Meinung dazu führen, dass Scala nicht für den Enterprise-Einsatz geeignet ist:

- Nicht vorhandene Abwärtskompatibilität der verschiedenen Scala-Versionen und ein verworrener Umgang mit Versionsnummern
- Nicht vorhandene Modularität von Scala
- Komplizierte Sprach-Features wie *Implicits* und symbolische Namen
- Dokumentation im PDF-Format
- Fehlen von erfahrenen Entwicklern
- Mangelnde Unterstützung innerhalb von Entwicklungsumgebungen

Es entstand nach der Veröffentlichung des Blogs eine Rege Diskussion innerhalb der Blogkommentare als auch in der Communit um das Thema, in das sich auch der „Erfinder“ von Scala, Prof. Martin Odersky (*École polytechnique fédérale de Lausanne*), einschaltete.

Alex Blewitt beließ es bei seinem ersten Blog nicht und nahm das Thema in einem weiteren Blog auf, der nach einigen Monaten dem ersten Blog folgte. In diesem Blog mit dem Titel *„Scala is still not enterise ready“* unterstreicht er die Aussagen, die er bereits in seiner ersten Veröffentlichung zu dem Thema dargelegt hatte.

Die Blogs von Blewitt griffen zu einer Zeit ein Thema auf, das bereits rege diskutiert und bereits auch vom Referenten in seine Architektur-Planungen von JEE-Projekten betrachtet wurde und hat nichts in der Zwischenzeit an Brisanz verloren. Das Thema ist eher noch stärker in den Fokus gerückt, da durch

die schleppende Modernisierung der Sprachfeatures von Java und der befürchteten Umwälzungen in diesem Umfeld (Stichworte Änderungen von Lizenzbedingungen) Entwickler und Projektleiter nach vernünftigen Alternativen Ausschau halten müssen.

Diese Diskussionen werden allerdings häufig nicht unbedingt Wertfrei geführt (die oben aufgeführten Blogs und die Folgekommentare sind hierfür ein gutes Beispiel) und sind geprägt durch Vorurteile gegenüber der Sprache, wie z.B. sie sei zu Akademisch, sowie auf der anderen Seite des Spektrums einem recht ausgeprägten *„Fanboyism“*, in dem Scala als Allheilmittel für alle Probleme der Java-Welt dargestellt wird.

Die Motivation des Referenten ist, einen sachlichen Überblick über den aktuellen Entwicklungsstand von Scala und dessen Ökosystem anhand der Kritikpunkte von Alex Blewitt zu geben.

Letztendlich stellt sich aber eine zentrale Frage unter all dem: Kann Scala für den Einsatz in Enterprise-Projekt empfohlen werden?

Was ist überhaupt *„Enterprise“*? Ein Definitionsversuch.

Alex Blewitt stellte eine interessante Definition des Begriffs *„Enterprise“* auf. Twitter als Beispiel genommen, auch wenn täglich Millionenfach genutzt, sei kein *„Enterprise“* sonder nur ein „Start-up“. Vielmehr seien Unternehmen mit mehreren tausend Mitarbeitern und mehreren Millionen Euro (bzw. Dollar) Umsatz „*Enterprise“*-Unternehmen; die EU-Kommision differenziert da übrigens ebenfalls nach diesen Kennzahlen, wobei die Latte wesentlich niedriger liegt.

Die Definition des Referenten lautet dahingegen folgendermaßen:

„Enterprise“ sind alle Unternehmen und Unternehmungen,
von deren Erfolg Menschen existentiell abhängen.

Enterprise-Applikationen im Sinne des Referenten sind also all die, von deren Erfolg Unternehmen und Menschen existentiell abhängen. Also auch z.B. Twitter oder auch ein kleines, mittelständisches Unternehmen, das seine neuesten Produkte exklusiv über ein selbst entwickeltes Portal vertreiben möchte.

Sprache und Syntax von Scala

Scala bietet mit seinen Sprach-Features und der Syntax einen großen Freiheitsgrad für Entwickler. Diese Freiheit, insbesondere im Zusammenspiel von Sprach-Features und der Sytax birgt allerdings das Risiko, das Quelltexte nicht mehr einfach zu lesen sind. Als Beispiel hier die Implementierung des Quicksort-Algorithmus in Scala, entnommen der Wikipedia.de:

```
def quickSort[A <% Ordered[A]](xs: List[A]): List[A] = xs match {
      case Nil => xs
    case y :: ys => ys partition (_ <= y) match { case (l1, l2) =>
                quickSort(l1) ++ (y :: quickSort(l2)) }
}
```

Tatsächlich muss ein Betrachter dieses Quelltextes erst einmal nachdenken um zu verstehen, was genau geschieht. Abgesehen davon, dass „nachdenken“ immer ein guter Ratschlag ist, spielt hier die Erfahrung mit einer Programmiersprache und hier mit Scala im Besonderen eine große Rolle.

Dies gilt auch für alle Programmiersprachen; ohne weitergehende Kenntnisse der Sprache und dessen Möglichkeiten erscheint ein Quelltext kryptisch.
Auch mächtige Sprachfeatures wie *Implicits* (diese dienen der Typumwandlung), die komplexen Regeln unterliegen, tragen dazu bei, dass Scala als eine komplexe, schwer zu verstehende und lernende Sprache gilt. Es ist in der Tat so, dass die Lernkurve für Scala sicherlich flacher ausfällt als für andere Sprachen wie Java oder C#.
Aber ist diese Komplexität tatsächlich ein Problem? Wiegt die einfache Syntax einer Sprache vielmehr die Beteiligten nicht in einer vermeintlichen Sicherheit, die vorliegende Sprache gänzlich zu verstehen im wahrsten Sinne des Wortes? Ist es in der Tat nicht so, dass viele (Java-)Entwickler ihr Wissen und damit auch ihre Fähigkeiten aufgrund dessen überschätzen?
Dies ist nach meiner Meinung eher ein Problem als ein Vorteil einer Sprache wie Java, auch wenn dies der weiten Verbreitung dieser Sprachen viel beigetragen hat. Genau so wenig stellt sich daher die Frage, ob genügend gute ausgebildete Entwickler im Scala-Umfeld zur Verfügung stehen.
Auch die Dokumentation von Scala ist im Übrigen sehr komplett und eine große Anzahl an Beispielen sowie Fachbüchern, akademischer als auch praktischer Ausprägung, stehen dem Lernwilligen zur Verfügung.

Scala-Versionen und Kompatibilität

Ein Punkt, der bei den ersten Kontakten mit Scala auffällt ist die binäre Inkompatibilität der Scala Versionen untereinander. Während in Java, z.B. mit javac 1.5 compiliert, auch unter einer JRE 1.6 lauffähig ist, ist diese Abwärtskompatibilität unter Scala nicht gegeben. Dies wird häufig, auch von A. Brewitt, als ein Haupthemmnis für den Enterprise-Einsatz betrachtet.
Allerdings stellt sich im Gegenzug die Frage: wie oft wird in einem Projekt die elementare Basis gewechselt? Wie oft wechselt ein großes Enterprise-Projekt im Java-Umfeld die JDK ohne einen zwingenden Grund und ohne dafür zusätzlichen Aufwand einzukalkulieren? Ich behaupte: nicht sehr häufig. Auch in Scala-Projekten dürfte der Drang nicht so groß sein, die Scala-Version sua sponte zu ändern. Allerdings, ein Wermutstropfen bleibt: Scala-Bibliotheken, die ggfs. in einer neuen Version nur compiled gegen eine aktuelle Scala-Version vorliegen, dürften den Wechsel-Druck auf ebendiese erhöhen. Aber alles in allem: auch kein zwingendes Gegenargument, Scala nicht in einem Enterprise-Projekt einzusetzen.

IDE und Tool-Support

Während sich tatsächlich vor zwei Jahren der Support für Scala unter den gängigen Java-Entwicklungsumgebungen in Grenzen hielt hat sich die Situation erfreulich geändert. Inzwischen bieten Eclipse, NetBeans und IntelliJ eine gute Unterstützung für die Entwicklung mit Scala an. Auch dieses häufig ins Feld geführte Argument der fehlenden Unterstützung kann daher nicht mehr ernsthaft weitergeführt werden.
Wird das weitere Ökosystem um Scala betrachtet, kann festgestellt werden, dass zudem alle Werkzeuge der Java-Welt ebenfalls vollständig zur Verfügung stehen. Neben dem dediziert für Scala entwickelten Build-Werkzeug *„Simple Build Tool“* (SBT) stehen auch die Java-Platzhirsche Maven und Ant zur Verfügung. Auch die Unterstützung mit Frameworks und Bibliotheken ist vollständig: fast alle Java-Frameworks können auch in Scala-Projekten eingebunden und verwendet werden.

Support und Wartung

Zu guter Letzt stellt sich häufig die Supprt-Frage: kann diese gewährleistet werden? Gibt es kommerziellen Support. Auch hier hat sich einiges im Scala-Umfeld getan. So bieten inzwischen

Unternehmen kommerziellen Support für Scala an, wie z.B. Typesafe, das von dem Erfinder der Sprache, Prof. Martin Odersky, mitgegründet wurde.

Fazit

Zusammenfassend kann festgestellt werden, dass Scala unter einigen Kinderkrankheiten gelitten hat, und an manchen sicherlich auch noch auskurieren mag. Auch wenn die Sprache sich komplexer darstellt und die Lernkurve ohne Zweifel flacher ist als bei anderen modernen Sprachen wie Java oder C#, gereicht dies nicht zum Nachteil sondern vielmehr zum Vorteil. Entwickler gehen die Sprache mit Respekt an und lesen sich vielleicht die Dokumentation auch ein zweites Mal durch, was der Qualität sicherlich nicht abträglich ist.
Letztendlich bleibt festzuhalten, dass Scala guten Gewissens in Enterprise-Projekten eingesetzt werden kann. Aber auch hier gilt: nicht blind auswählen sondern unabhängig von aktuellen Trends abwägen, was tatsächlich benötigt wird.

Kontaktadresse:

Dr. Halil-Cem Gürsoy
adesso AG
Stockholmer Allee 24
D-44269 Dortmund

Telefon:	+49 231 930-9330
Fax:	+49 231 930-9331
E-Mail	halil-cem.guersoy@adesso.de
Internet:	www.adesso.de
G+:	http://goo.gl/hljRS
Twitter:	@hgutwit