Erlang Programming
Francesco Cesarini and Simon Thompson
First Edition Juli 2009
ISBN 978-0-596-51818-9
494 Seiten, broschiert
EUR48.00, SFR79.90
http://www.oreilly.de/catalog/9780596518189/

Buchbesprechung

Das Buch *Erlang Programming* ist als Einführung geschrieben und richtet sich an Leser jeden Kenntnisstandes. Die Wahl des Stils einer Überblicksdarstellung ist dabei sehr passend, denn bei Erlang handelt es sich um eine Sprache, die alles andere ist, als eine „gewöhnliche“ Programmiersprache.

Im ersten Kapitel werden Enstehungsgeschichte und wichtige Charakteristiken von Erlang kurz dargestellt, wobei auch anschaulich auf die Eigenschaft der Programmierung von Nebenläufigkeit eingegangen wird. Ausserdem werden zwei Erlang-Einsatzgebiete (der AXD301 ATM Switch und die Datenbank CouchDB) vorgestellt und ein Vergleich zwischen Erlang und C++ vorgenommen.

Das zweite Kapitel gibt einen Überblick über die grundlegenden Datenstrukturen von Erlang: Integer, Floats, Atome, Bools, Tuples und Listen sowie die zur Verfügung stehenden Verglechsoperatoren. Im Abschnitt über Variablen erfährt man, dass Erlang nur die *einmalige* Bindung von Werten an Variablen erlaubt, d.h. deren Redefinition ausschließt und ferner, dass die Zuweisung von Werten an Variablen eigentlich ein Pattern Matching-Vorgang ist. Es wird außerdem erläutert, wie man (polymorphe) Funktionen schreibt, diese in Module auslagert und diese Module dann einbindet bzw. kompiliert.

Das dritte Kapitel behandelt Kontrollstrukturen in Erlang, bzw. deren funktionale Entsprechung. Vorgestellt wird bedingte Evaluation, Guards und Rekursion. Weitere Themen dieses Kapitel sind vordefinierte Funktionen (BIFs) und Bibliotheksmodule, sowie Fehlermeldung und deren Handhabung durch *try...throw...catch*-Funktionen. Insbesondere das letzte Thema setzt jedoch etwas Vertrautheit mit dieser Konzeption aus anderen Programmiersprachen voraus und hätte für den Einsteiger etwas anschaulicher dargestellt werden können. Ebenfalls vorgestellt wird der Erlang-Debugger.

Im vierten Kapitel dringen wir nun schließlich zum Herzstück von Erlang vor: der nebenläufigen Programmierung. In klaren Worten wird auch der in diesem Bereich völlig Unerfahrene mit den wichtigsten hierfür benötigten Strukturen bekannt gemacht: dem Erzeugen von Prozessen mit *spawn*, dem Senden von

messages mit ! und dem Erhalten von messages mit *receive... end*. Wir erfahren, wie man Prozesse unter einem bestimmten Namen registriert und wie man Timeouts programmiert. Im Abschnitt Benchmarking wird uns eindrucksvoll vorgeführt, wie Erlang damit umgeht, wenn rekursiv 10 Millionen (!) parallele Prozesse gestartet werden. Sehr anschaulich ist der Abschnitt Process Skeletons, der zusammenfasst, wie ein typischer nebenläufiger Prozess strukturiert ist. Abgerundet wird das Kapitel durch Hinweise auf Speicherplatzprobleme, die bei Rekursion auftreten können, und weitere kritische Fragen der Programmarchitektur. Schließlich erfahren wir noch etwas über das Tool des Prozessmanagers.

Im fünften Kapitel werden weitere prototypische Musterlösungen für nebenläufige Prozesse vorgestellt. Implementiert werden eine Client/Server-Struktur, ein endlicher Automat und ein Ereignismanager mit austauschbaren Steuerungsprogrammen.

Im sechsten Kapitel werden die Möglichkeiten zum Abfangen von Crash-Signalen durch Verlinkung und Monitoring von Prozessen behandelt. Am Ende des Kapitels werden robuste Systeme vorgestellt, in denen jeder Prozess an einer Supervisor-Worker-Hierarchie teilhat.

Im siebten Kapitel werden die beiden Präprozessorstrukturen Records und Macros behandelt.

Im achten Kapitel wird sehr anschaulich demonstriert, wie Module und Prozesse im laufenden Betrieb aktualisiert werden können, ohne ein laufendes System anzuhalten.

Das neunte Kapitel behandelt Funktionen höherer Ordnung in Erlang. Eingegangen wird dabei zunächst auf Lambda-Ausdrücke (funs) und deren Anwendung. Danach werden einige vordefinierte Funktionen höherer Ordnung vorgestellt (z.B. map, filter, fold u.a.). Wichtig ist auch der Hinweis, dass Erlang von Haus aus keine lazy-Evaluation kennt, sondern diese durch Lambda-Ausdrücke im CDR-Teil einer Liste simuliert werden muss. Es folgen Abschnitte über die Funktion zur automatischen Listenerzeugung und über die Behandlung von Binaries. Etwas unglücklich geraten ist allerdings der Abschnitt über Bitsyntax der Form Expr:Size/Type, der für meine Begriffe völlig unverständlich geschrieben ist und unbedingt überarbeitet werden sollte. Am Ende des Kapitels finden sich noch zwei Abschnitte zur Darstellung von Binärbäumen und zu Message-Referenzen.

Bevor in Kapitel dreizehn auf das Datenbankmodul Mnesia eingegangen wird, sind einfache Datenbank-Tabellenstrukturen Gegenstand des zehnten Kapitels. In der ersten Hälfte des Kapitels werden ETS-Tabellen vorgestellt, die sich kom-

fortabel als datenbankähnliche Datenstruktur nutzen lassen. Kurz eingegangen wird auch auf Dets-Tabellen, die zwar langsamer sind als ETS-Tabellen, aber durch die Tatsache, dass sie als File abgelegt werden, Datenpersistenz erlauben. Am Ende des Kapitels wird die Programmierung eines Datenbank-Backends für einen Telefonkundenserver ausführlich kommentiert dargestellt, sowie das dazugehörige Datenbank-Servermodul aufgelistet. Eine hervorragende Verdeutlichung des Denkens in Erlang an passender Stelle!

Kapitel elf behandelt ein weiteres Zentralthema in Erlang: die verteilte Programmierung. Hier wird in erstaunlich einfachen Worten beschrieben, wie verteilte Systeme in Erlang aufgebaut werden können. Schon nach wenigen Sätzen merkt man, dass verteiltes Programmieren in Erlang eigentlich genauso funktioniert, wie nebenläufiges Programmieren, außer dass eben mit der Server-Knotenadresse ein zusätzliches Argument hinzukommt. Ausgehend von verteilten Knoten auf einem Rechner wird das Beispiel auf Knoten auf verschiedenen Rechnern ausgeweitet, wobei einfache Sicherheitsaspekte (cookies) behandelt werden, die für geschlossene Systeme ausreichen. (Verteilte Programmierung über TCP/IP wird in Kap.15 behandelt.) Versteckte Knoten helfen die Monitorlast sich gegenseitig überwachender Prozesse zu senken. Schließlich wird ein einfaches Beispiel für einen Remote Procedure Call implementiert. Am Schluß des Kapitels wird noch auf die Möglichkeit eines eigenen Portmapping mittels epmd hingewiesen.

Das zwölfte Kapitel behandelt OTS (Open Telecom Platform), ein standardisiertes Framework zur Erstellung von nebenläufigen Applikationen in einer Supervisor/Worker-Prozeßhierarchie. Zur Veranschaulichung wird der in Kapitel zehn erstellte Mobiltelefonkunden-Datenbankserver nun als generischer OTP-Server aufgesetzt. Danach wird ein Supervisor-Prozess implementiert, der die Ausführung des Servers überwacht. Zum Schluss werden Worker- und Supervisormodule in eine Applikation gebündelt und schließlich ein Versionrelease erstellt, aus dem ein Boot-File erzeugt werden kann, mit dem die Anwendung auf einem Erlang-Knoten gebootet werden kann.

Kapitel dreizehn gibt einen kurzen Einblick in die Funktionsweise der Datenbank Mnesia, die als abgesicherte Variante für den Gebrauch auf verteilten Knoten bei einer Datenbankgröße von ca. 2GB sinnvoll eingesetzt werden kann. Im Kapitel wird die Dets-Datenbank aus dem Beispiel aus Kapitel zehn nach Mnesia umgesetzt.

In Kapitel vierzehn wird die GUI-Programmierung mit wxErlang behandelt. Obwohl darauf hingewiesen wurde, dass GUI-Programmierung nicht gerade eine Stärke von Erlang darstellt, hätte dieses Kapitel ruhig etwas umfangreicher ausfallen können.

Kapitel fünfzehn schildert – ebenfalls sehr kurz – die IP-Kommunikation mit TCP- und UDP-Paketen.

In Kapitel sechzehn wird die Interoperabilität mit Erlang anhand von Beispielen aus Java, C, Ruby und von Interaktion über die Shell veranschaulicht. Außerdem werden Erlang-Ports und Linked-In Drivers als Interfacingmethoden erörtert.

Kapitel siebzehn befasst sich mit Tracing in Erlang. Dabei werden zunächst die low-level-BIFs trace/3 und trace_pattern/3 verwendet und später mit den etwas benutzernäheren Funktionen der dbg-Library operiert. Am Ende des Kapitels widmen sich zwei sehr umfangreiche Abschnitte dem Verfahren der Match Specifications, die wir schon in Kapitel zehn im Zusammenhang mit ETS kennengelernt hatten. Zunächst wird dabei darauf eingegangen, wie man Match Specifications beim Tracing einsetzt, die zuvor mithilfe des tools fun2ms aus Funktionen erzeugt werden. Danach wird die Syntax von Match Specifications nochmal detailliert erklärt. Dem fortgeschrittenen Zustand des Buches entsprechend wird der Stoff hier sehr kompakt dargestellt und erfordert seitens des Lesers erheblich mehr Konzentration als am Anfang des Buches. Es könnte jedoch auch sein, dass die Konzentration der Autoren an dieser Stelle bereits etwas abgenommen hat, so dass pädagogisch-didaktische Erwägungen in der Darstellung auf der Strecke geblieben sind. Bei einer Überarbeitung wäre eine verständlichere Darstellung der Inhalte eben dieses relativ wichtigen Kapitels unbedingt wünschenswert!

In Kapitel achtzehn geht es um Typdeklaration mit dem -type-Befehl, sowie die Analyse von Programmcode auf Typfehler und andere Inkonsistenzen mit den Tools typer und dialyzer. Am Ende wird die Dokumentation von Programmcode mit dem edoc-Dystem erläutert.

Kapitel neunzehn erklärt in einfachen und klaren Worten das Tool EUnit, mit dem man Tests für einfache Erlang-Module herstellen und ausführen kann. Nur ganz kurz gestreift wird die Problematik des Testens nebenläufiger Systeme.

Im abschließenden zwanzigsten Kapitel geben die Autoren Hinweise für eine solide Programmiertechnik in Erlang. Dabei gehen sie auf Programmiertechniken und -strategien, Stilfragen, Effizienz und besondere Probleme der Nebenläufigkeit ein. Das Kapitel ist wie eine narrative Aufzählung von Dos und Don'ts geschrieben, was das Lesen etwas erschwert. Besser wäre es gewesen, die Hinweise durchzunummeriren und inhaltlich besser zu gliedern, so dass man die entsprechenden Hinweise später leicht wieder auffinden kann. So muss man sich die vielen wirklich guten Tipps beim Lesen leider alle merken.

Abgerundet wird das Buch durch einen kurzen Anhang, der Informationen zur

Installation, die Bedienung der Erlang-Shell, Tools und weitere Informationsquellen enthält. Lediglich der Abschnitt über die Erlang-Shell könnte ruhig noch etwas detaillierter sein.

Fazit: Ein Buch, das man unbedingt auch dann lesen sollte, wenn man nicht gerade konkret mit Erlang programmieren muss. Die hierin vermittelten Grundlagen zur nebenläufigen Programmierung sind wertvolles Allgemeinwissen, um das man eigentlich nicht drumherumkommt. Klar und umfassend dargestellt kann man sich eigentlich keine bessere Darstellung des hierin verfolgten Programmierparadigmas vorstellen. Ein Buch, das Türen öffnet und den Verständnishorizont enorm erweitert.

Einziges leider wirklich gravierendes Manko: Es fehlt ein Anhang mit den Lösungen zu den Übungsaufgaben. Alternativ wären zumindest Lösungscodes im Zipfile von der Website zum Buch wünschenswert. Diese wären insbesondere deshalb hilfreich, da es sich bei Erlang ja nicht um eine „normale“ Sprache handelt, bei der man anderswo Gelerntes einfach übertragen kann, sondern um eine Sprache, in die man sich teilweise völlig neu eindenken muss.

Das Buch ist größtenteils sehr anschaulich und verständlich geschrieben. Erst im letzten Viertel des Buches wird die Darstellung manchmal etwas anspruchsvoller, bzw. das didaktische Handwerk nicht mehr so gepflegt, wie noch in den ersten fünfzehn Kapiteln, was ich mitunter kritisieren möchte. Trotz kleiner derartiger Mängel, wie sie oben im Detail beanstandet wurden, handelt es sich hier aber im großen Ganzen um eine absolut gelungene Rundum-Einführung in die vielen Aspekte des Programmierens in Erlang.

Harald Vajkonny