Buchbesprechung: E. Freeman, S. Hüpfer, K. Arnold — JavaSpaces Principles, Patterns and Practice *

Ernst-Erich Doberkat†

23. Juni 2000

In [CG90] zeigten N. Carriero und D. Gelernter, wie mächtig ein Mechanismus zur Koordination ist: Mit dem Linda-Mechanismus gelingt es, eine sequentielle Sprache — in diesem Fall C — in eine Sprache zur parallelen Programmierung zu verwandeln. C-Linda entstand, und weil die Mechanismen der Koordinierung bestechend einfach sind, konnten weitere Lindas entstehen.

Zentral im Ansatz von Carriero und Gelernter ist ein Tupel-Raum, der als Multimenge Tupel beliebiger Länge mit heterogen getypten Komponenten enthält. Ein Prozeß kann ein Tupel in einem Tupelraum ablegen, ein anderer Prozeß kann den Tupelraum fragen, ob er ein vom Prozeß spezifiziertes Tupel enthält. Linda enthält Operationen, mit denen Tupel gelesen oder aus dem Tupelraum entfernt werden können. Diese Operationen können blockieren, so daß sich einfache Möglichkeiten zur Synchronisation ergeben. Auf diese Weise ergibt sich ein einfacher, aber mächtiger Zugang zur parallelen Programmierung. [CG90] geben einige Beispiele zur parallelen Programmierung in C-Linda an, die recht schnell von den Vorteilen des Zugangs überzeugen.

Das vorgelegte Buch macht den Linda-Ansatz für Java nutzbar (es finden sich weitere Ansätze, z.B. Jada und Laura [CTV+98] oder die bei IBM entwickelten T-Spaces). Java verfügt bekanntlich über `threads`, ist also für die parallele Programmierung bereits ausgestattet. Warum also Linda in Java? Gelernter gibt in seinem Vorwort eine recht plakative Antwort: er möchte gern einen weltweiten Informationspool geschaffen sehen, der allen offensteht, und auf den weltweit jeder mit derg geeigneten Technik zugreifen kann (also eine Art *MEMEX* mit aktiven Operationen). Diesen weltweiten Informationspool sieht er als Tupelraum realisiert, auf den man etwa mit JavaSpaces zugreifen kann.

Technisch ist JavaSpaces eine Entwicklung, die der JINI-Klasse zugerechnet werden muß [OW00], also einer API, mit deren Hilfe ein weites Spektrum von Rechnern autonomer oder eingebetteter Art ihre Dienste über Netzwerke anzubieten gestatten. Sie werden also in Zukunft Ihren Kühlschrank über das Handy mit geeigneten JINI-Objekten abfragen können. Da JINI Pate stand, werden in JavaSpaces zwei wichtige Operationen zugrunde gelegt, nämlich Transaktionen und Leases. Transaktionen sind aus dem Bereich der Datenbanken bestens bekannt und dienen zur Synchronisation zusammengesetzter Operationen. Leases sind ein originärer Beitrag von JINI, sie dienen grob gesagt dazu,, die Verfügbarkeit eines Objekts zeitlich einzugrenzen (eine Art zeitlich befristete Pacht).

Gut.

*Addison-Wesley, Reading, Mass, 1999, ISBN 0-201-30955-6, US $ 39.95

†Lehrstuhl für Software-Technologie, Universität Dortmund, `doberkat@acm.org`

Das Buch ist als Tutorium angelegt und setzt voraus, daß der Leser fließend Java spricht. Eine gewisse Vertrautheit mit verteilten Arbeiten ist sicher hilfreich, wird aber im Laufe der Lektüre des Buchs auch entwickelt. Kapitel 1 gibt einen Überblick über die, wie es so schön heißt, JavaSpaces Technologie; Kapitel 2 führt die primitiven Operationen aus JavaSpaces ein, die aus Linda übernommen worden sind und die sich nun in einer Java-Verkleidung wiederfinden. Für die kritischen JINI-Konstruktoren werden stets voreingestellte Werte verwendet, so daß sich der Leser auf das Erlernen der Linda-Operationen konzentrieren kann und sich nicht mit JINI vertraut gemacht haben muß. Kapitel 3 geht auf spezifische Konstuktionen ein, in Linda vertraute Datenstrukturen mit einem Tupelraum aufzubauen, und Kapitel 4 zeigt einige Möglichkeiten der Synchronisation mit Hilfe der Linda-Operationen auf (natürlich werden die Philosophen zum Essen eingeladen). Damit ist der simple Teil vorbei, jetzt wird´s technischer. Kapitel 5 zeigt, wie sich einige Kommunikationsprimitive in enger und in loser Kopplung darstellen lassen, und Kapitel 6 handelt von Anwendungsmustern (sehr ähnlich zu [CG90, Kap.9]): die Muster `Command`, `Marketplace` und `Replicated Worker` werden eingeführt und am Beispiel diskutiert. Die drei folgenden Kapitel sind eine Art Import aus JINI: Pachtverträge, verteilte Ereignisse und Transaktionen werden kurz, knapp, aber heftig soweit hingeführt, daß sich der Leser bei der Anwendung der Konzepte auf JavaSpaces sicher fühlt. Zwei substanzielle Anwendungen (ein interaktiver Botendienst und ein Ansatz, kryptographische Codes zu brechen) schließen den Hauptteil des Buchs. Drei Anhänge befassen sich mit JINI und der JavaSpaces Spezifikation.
Das Buch ist angenehm zu lesen, die Beispiele sind ordentlich durchdacht, auch wenn gelegentlich einige Einzelheiten offenbar ausgelassen wurden JINI iin a Nutshell — A Desktop Quick Reference und nur in der Netzversion vollständig zur Verfügung stehen. Die Schrittweite der Argumentation erschien mir angemessen gewählt, nur gelegentlich wird über allzu arge technische Probleme hinweggewischt, was man in amerikanischen Hörsälen gern als *hand waving* bezeichnet. Ich fand insgesamt die erneute Bekanntschaft mit Linda erfreulich, denke, daß wir es hier mit einer sehr wichtigen und sehr vielversprechenden API zu tun haben, und empfehle das Buch gern.

Literatur

[CG90] Nicholas Carriero und David Gelernter. *How To Write Parallel Programs — A First Course.* The MIT Press, Cambridge, MA, 1990.

[CTV+98] Paolo Ciancarini, Robert Tolksdorf, Fabio Vitali, Davide Rossi und Andreas Knoche. Coordinating multiagent applications on the WWW: A reference architecture. *IEEE Trans. Softw. Eng*, 24(5):362 – 375, 1998.

[OW00] Scott Oaks und Henry Wong. *JINI in a Nutshell — A Desktop Quick Reference.* O'Reilly Associates, Köln, 2000.