Mark Rambow

Implementierung einer Referenzanwendung für den JBoss Application Server unter Verwendung der Java 2 Enterprise Edition

Diplomarbeit

GRIN

Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de/ abrufbar.

Impressum:

Copyright © 2004 GRIN Verlag
ISBN: 9783638818599

Mark Rambow

Implementierung einer Referenzanwendung für den JBoss Application Server unter Verwendung der Java 2 Enterprise Edition

GRIN Verlag

Implementierung einer Referenzanwendung für den JBoss Application Server unter Verwendung der Java 2 Enterprise Edition

Mark Rambow

Diplomarbeit

zur Erlangung des akademischen Grades
Diplom-Informatiker (FH)

eingereicht an der
Fachhochschule Brandenburg
- Fachbereich Informatik und Medien -

Brandenburg an der Havel, 19. Mai 2004

Danksagung

Durch die Auswahl des Themas und die gemeinsame Bestimmung der Schwerpunkte mit meinem betreuenden Prof. Dr. Stefan Edlich, ist die vorliegende Diplomarbeit zum interessantesten Teil meines Studiums geworden.
Für die Unterstützung bei der Umsetzung möchte ich mich hiermit bei meinen Betreuern Prof. Dr. Stefan Edlich und Prof. Dr. Thomas Preuß bedanken.

Ebenfalls möchte ich mich an dieser Stelle bei meinen Kommilitonen Christian Koth, Jens Ziegler, Oliver Kalz, Andreas Graff und Mathias Meyer bedanken. Durch ihre konstruktive Kritik und ihr fachliches Wissen konnten einige Fragen, die im Verlauf der Arbeit auftraten, geklärt werden.

Meinen Eltern möchte ich für die großzügige Unterstützung während des gesamten Studiums danken.

Mark Rambow

Inhaltsverzeichnis

1. Einleitung

Durch zunehmende Globalisierung und gemeinschaftliches Arbeiten bei räumlicher Trennung gewinnt die Vernetzung von Unternehmensanwendungen immer mehr an Bedeutung. Das World Wide Web ermöglicht den Zugriff auf die Daten von Unternehmen. Um dieser Entwicklung gerecht werden zu können ist es notwendig, Konzepte für eine Infrastruktur zu schaffen, welche die dabei auftretenden Probleme beherrschbar macht. Dieser Herausforderung stellen sich Unternehmen wie Sun Microsystems oder Microsoft.
In dieser Arbeit soll auf die von Sun eingeführte, *Java 2 Platform, Enterprise Edition* (J2EE) eingegangen werden. Um deren Vorteile nutzen zu können, wird ein Application-Server verwendet, welcher die benötigten Dienste bereitstellt. In der vorliegenden Diplomarbeit soll eine Evaluation des Open-Source-Projekt *JBoss Application Server* durchgeführt werden, dabei wird JBoss auch kommerziellen Produkten gegenübergestellt.
Ziel ist es, dem Leser sowohl ein Verständnis für das Erstellen von Anwendungssoftware auf Basis von J2EE zu vermitteln, als auch den JBoss Application Server detailliert und kritisch zu untersuchen.

1.1. Thematik und Aufbau der Arbeit

Diese Arbeit beschäftigt sich mit der Erstellung einer Referenzimplementierung für den JBoss Application-Server. In diesem Zusammenhang wird in die Grundlagen für die Erstellung von komponentenbasierten Systemen auf J2EE-Basis eingeführt, einige „best practises“ in Form von Entwurfsmustern (Pattern) vorgestellt und Quellcodebeispiele zur Veranschaulichung angeführt.
Der zweite Schwerpunkt dieser Arbeit befasst sich mit Application-Servern, speziell mit dem JBoss. Dabei wird der JBoss genau untersucht und seine Konfiguration erläutert. Um den JBoss Application Server im Markt einordnen zu können, wird ein kurzer Vergleich mit

den Konkurenzprodukten durchgeführt. Es soll gezeigt werden, welche Kriterien für den Einsatz eines Application-Server sprechen und was beim Einsatz zu beachten ist.

1.2. Motivation

JBoss ist ein Open-Source Application-Servern für J2EE Anwendungen. Dadurch ist er sowohl für kosteneffizient arbeitende kleinere Unternehmen interessant, die ihn kostenlos einsetzen können als auch für große Unternehmen, die in der Verfügbarkeit des Quellcodes mehr Sicherheit für ihr Unternehmen erkennen. Durch seine große Entwicklergemeinde, die als JBoss Group gemeinsam das Projekt vorantreiben, gehört er zu den sich am schnellsten weiterentwickelnden Application-Servern.
Durch den Einsatz neuester Technologien gehen vom JBoss immer wieder Impulse aus, die ihn besonders interessant für die Untersuchung machen.

1.3. Aufgabenstellung

Die Aufgabenstellung umfasst sowohl die Erstellung einer Referenzimplementierung auf Basis des JBoss als auch die genaue Untersuchung des JBoss Application Servers. Das zu entwickelnde Programm stellt ein Beispiel für J2EE-Anwendungen dar. Sie soll nicht den Anforderungen eines Produktivsystems gerecht werden, sondern vielmehr als Studienobjekt Entwickler bei der Erstellung von J2EE-Anwendungen und dem Betrieb des JBoss Application Server unterstützen.

2. Theoretische Grundlagen

Die Basis für komplexe Anwendungen im J2EE[1]-Bereich bildet die Programmiersprache Java[2], auf der die Enterprise Edition aufbaut. Im Rahmen dieser Diplomarbeit kann keine vollständige Einführung in die Softwareentwicklung mit Java gegeben werden. Vielmehr geht es um die Vermittlung von Konzepten, die bei der Entwicklung von J2EE-Applikationen Verwendung finden.
Weitere wichtige Grundlagen stellen die Auszeichnungssprache XML[3] und die Datenbank-Anfragesprache SQL[4] dar. In welchem Zusammenhang sie mit der Entwicklung von Enterprise Systemen stehen, wird im ersten Abschnitt dieses Kapitels veranschaulicht.
Der daran anschließende Abschnitt „Verteilte Anwendungen“ geht tiefer auf die Programmierung mit der Java 2 Enterprise Edition ein. Es soll ein Überblick über die verfügbaren Komponenten der J2EE gegeben und diese mit anderen Modellen verglichen werden. Den Abschluss des Kapitels bildet der Abschnitt „Softwareentwicklung für komponentenbasierte Systeme“. Hier wird auf das Softwareengineering bei J2EE Anwendungen eingegangen und es werden Standards vorgestellt, die diesen Prozess unterstützen.

2.1. Einführung in plattformunabhängige Programmierung

Unter Plattformunabhängigkeit versteht man die Unabhängigkeit einer entwickelten Software von einer spezifischen Zielplattform. In die Betrachtung müssen dabei sowohl die Hardware als auch die verwendeten Betriebssysteme einbezogen werden. Unter anderem ist dabei sicherzustellen, dass alle Datentypen den gleichen Wertebereich ausschöpfen z. B.

[1] Java 2 Platform, Enterprise Edition
[2] Java ist eingetragenes Markenzeichen der Sun Microsystems Cooperation
[3] Extensible Markup Language
[4] Structured Query Language

ist der Integer-Wertebereich maschinenabhängig. Eine weitere Anpassung bezieht sich auf die Haltung von Daten im Hauptspeicher. Dieser kann als „Big-Endian“ oder „Little-Endian“ organisiert sein. Dabei bedeutet Big-Endian, dass die Bytes in einer Speicherzelle vom höchstwertigen Byte links zum niederwertigsten Byte rechts angeordnet werden. Beim „Little-Endian“-Format, welches hauptsächlich von Intel bevorzugt wird, steht das niederwertigste Byte links, das höchstwertige Byte rechts [Cur94].
Auch der Zugriff und Austausch von Daten muss geregelt werden. Immer häufiger kommt daher eine Datenaustauschsprache zum Einsatz, die öffentlich spezifiziert wurde - XML. Auch offen gelegte Standards bei Netzwerkprotokollen helfen hier Plattformunabhängigkeit zu erreichen. Um dieses Konzept zu verwirklichen ist es erforderlich, eine Vermittlerinstanz (Middleware) zwischen den Programmcode und die Zielplattform zu schalten. Mit einer solchen Middleware kann z. B. ein Java-Programm mit einem in COBOL[5] implementierten Programm auf einem Mainframe kommunizieren. Hierbei könnte z. B. CORBA[6] als Middleware eingesetzt werden. Ein gutes Beispiel für offene Protokolle zur Kommunikation findet man u. a. bei den WebServices. Hier wird mittels SOAP[7] und einer Servicebeschreibung in XML der Zugang zu Programmen unterschiedlicher Programmiersprachen und Systeme hergestellt. Ein weiterer Ansatz, der bei portablen Programmen unterschiedlicher Sprachen zum Einsatz kommt, ist eine Laufzeitumgebung, wie sie bei der Sprache Java oder der .NET-Sprachfamilie zu finden sind. Diese Sprachen nennt man *interpretierte* Programmiersprachen. Im Fall von Java wird kein nativer Code erzeugt, sondern Bytecode, der von der JVM[8] interpretiert wird. Das hat den Vorteil, dass man Programmcode auf jedem Rechner, für den eine JVM verfügbar ist, ohne Änderungen ausführen kann. Dieses Prinzip nennt Sun „Write Once, Run Anywhere“.

2.1.1. Die Java-Plattform

Als Sun 1995 die Programmiersprache Java veröffentlichte, wuchs deren Verbreitung mit der gleichzeitig wachsenden Popularität des Internets. Java setzte sich zum Ziel, die unterschiedlichsten Hard- und Softwareplattformen in der Art zu vereinen, dass Java-Programme ohne Codeanpassungen überall dort lauffähig sind, wo eine JVM installiert ist.

[5] Common Business Oriented Language
[6] Common Object Request Broker Architecture
[7] Simple Object Access Protocol
[8] Java Virtual Maschine

Die ersten Java-Programme waren Applets, die in Browsern ausgeführt und von einem Server bei Bedarf geladen wurden. Der Vorteil dieser Programme war, dass eine Oberfläche für dynamische Interaktion von Webseiten mit deren Benutzern geschaffen wurde. Oftmals wurden mit Applets jedoch auch einfach nur Webseiten, z. B. durch animierte Logos, aufgewertet.
Die Benutzer dieser Programme können sicher sein, dass diese keinen Schaden verursachen. Java-Applets laufen in der sogenannten *Sandbox* ab. Das heißt, dass sie nicht auf Systemressourcen zugreifen können. Aufgrund der Applet-Technologie wird Java oft als die Sprache des Internets bezeichnet.
Doch die Java-Plattform kann weitaus mehr und entwickelt sich in rasantem Tempo weiter. Durch die Auslegung Javas als objektorientierte Programmiersprache mit einer Vielzahl von APIs[9], z. B. für Netzwerkprogrammierung und Sicherheit, wurde Java unter anderem auch für Serverprogrammierung interessant. Der eigentliche Durchbruch auf diesem Gebiet gelang Sun mit der Spezifikation von J2EE mit den Teilspezifikationen der Servlets und JavaServer Pages (vgl. A.1) sowie der Enterprise JavaBeans (EJB), die in Kapitel 3 beschrieben werden. Die Plattformunabhängigkeit half dabei der Marktdurchdringung, so dass Java eine der am häufigsten verwendeten Programmiersprachen ist.

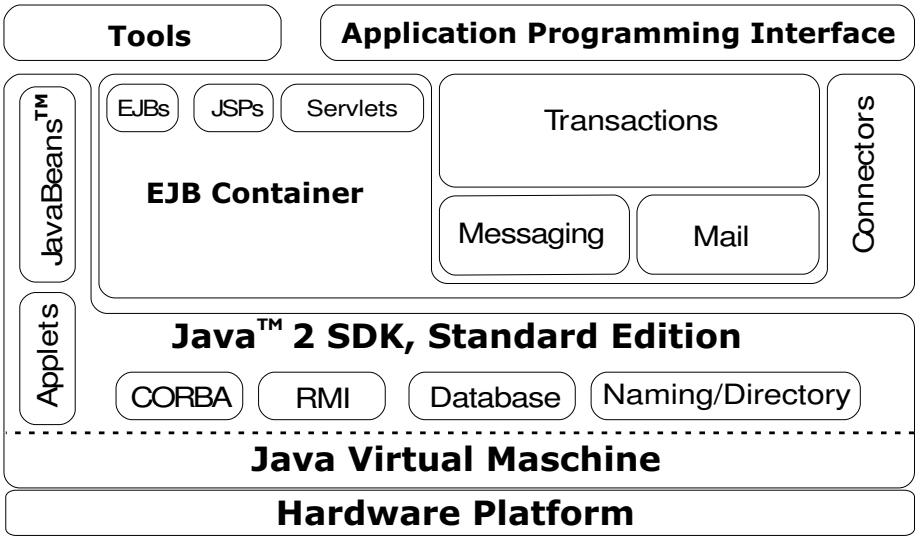

Abb. 2.1: Java und J2EE Quelle: [SM97]

[9] Application Programming Interface