TURING 图灵程序设计丛书 Java系列

Apress®

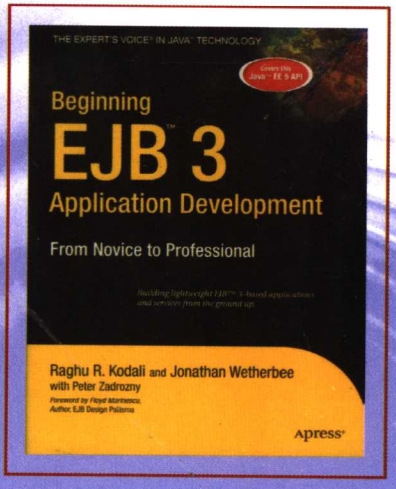
Beginning EJB 3 Application Development

EJB 3基础教程

[美] Raghu R. Kodali
Jonathan Wetherbee
Peter Zadrozny 著

马朝晖 杨 艳 等译

- 世界级EJB专家力作
- 透彻易读，覆盖全面
- Amazon四星图书



人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书 Java系列

EJB 3 基础教程

Beginning EJB 3 Application Development

Raghu R. Kodali
[美] Jonathan Wetherbee 著
Peter Zadrozny

马朝晖 杨艳 等译

人民邮电出版社
北 京

图书在版编目（CIP）数据

EJB 3 基础教程 /（美）科达利（Kodali, R.R.），（美）韦瑟比（Wetherbee, J.），（美）扎德罗兹尼（Zadrozny, P.），著；马朝晖等译. —北京：人民邮电出版社，2008.1
（图灵程序设计丛书）
ISBN 978-7-115-16622-7

Ⅰ.E… Ⅱ. ①科…②韦…③扎…④马… Ⅲ. JAVA 语言—程序设计—教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2007）第 114914 号

内 容 提 要

本书从 EJB 3 最基本的内容讲起，接着逐步讲解了 EJB 3 应用程序开发的整个过程，涉及 EJB 3 规范的各个方面，并将许多实践经验融入整个 EJB 架构，提供了对 EJB 3 架构和 EJB 3 编程全面的实战性指导，充分体现了 EJB 3 强大的功能和易于使用的特点。

本书主要面向 Java 和 J2EE 开发人员。

图灵程序设计丛书

EJB 3 基础教程

◆ 著 [美] Raghu R. Kodali Jonathan Wetherbee Peter Zadrozny
译 马朝晖 杨 艳 等
责任编辑 傅志红 王慧敏

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
三河市海波印务有限公司印刷
新华书店总店北京发行所经销

◆ 开本：800×1000 1/16
印张：21.5
字数：577 千字 2008 年 1 月第 1 版
印数：1 – 5 000 册 2008 年 1 月河北第 1 次印刷
著作权合同登记号 图字：01-2007-1967 号

ISBN 978-7-115-16622-7/TP

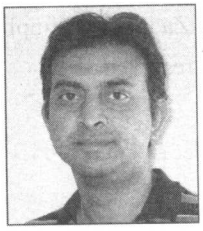
定价：49.00 元

读者服务热线：(010)88593802 印装质量热线：(010)67129223

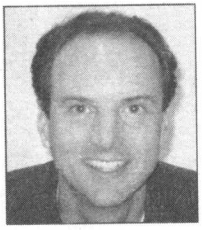
版 权 声 明

作者简介

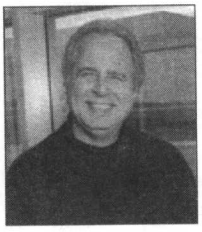
本书的三位作者均为世界级的EJB专家，拥有10年以上的软件开发经验。



Raghu R. Kodali是世界级的EJB专家，OASIS SOA Blueprints规范技术委员会成员。他是Oracle公司资深工程师，领导开发Oracle公司EJB产品多年，目前担任Oracle Application Server顾问产品经理和SOA技术推广官。Kodali在Java EE社区非常活跃，除了经常在各种世界性技术大会上演讲之外，还是*Java Developer's Journal*等杂志的专栏作家。他的博客地址是www.jroller.com/page/raghukodali。



Jonathan Wetherbee是Oracle公司最出色的EJB专家之一，自EJB 1.1以来一直负责Oracle的核心EJB工具包，并开发了各种O/R映射工具。目前是Oracle JDeveloper IDE EJB开发工具的顾问工程师和技术主管。



Peter Zadrozny是世界顶级的软件技术专家之一，目前担任领先的数字消息基础架构厂商StrongMail公司的首席技术官。他拥有超过20年的软件开发经验，曾担任BEA公司欧洲、中东和非洲的首席技术专家，和Oracle公司Oracle Application Server集团副总裁和首席技术推广官。除本书外，他撰写过其他多部著作，还是*WebLogic Developer's Journal*杂志的首任主编。

序

EJB 3是EJB规范的一个非常重要的里程碑，不仅因为它更加容易使用，而且因为它是第一次（在我看来）根据开发社区的需求创建的，是已有的最佳实践经验的标准化，而不是委员会的设计[①]结果。本书的主要读者应该是开发人员，所以我将从开发者的角度介绍一些EJB形成以及现在为什么如此重要的历史背景。

我经常在各种会议上演讲，在演讲时我会问有多少人使用过EJB。通常有90%的听众举手。然后，我问他们："有多少人把EJB作为分布式对象使用——也就是说，有独立的物理层放置业务逻辑，还有独立的物理层放置表示（Servlet/JSP）层？"通常那90%的听众中只有15%举手。无论是在北美、欧洲还是日本的会议中，情形都是如此。这个结果至今让我吃惊，因为早期的EJB迫使你在代码中应用分布式语义，这对大型的多层项目是有用的，但实际上，EJB的大多数使用者都把它作为中小型Web应用程序的本地框架。

为什么会这样呢？如果了解这段时间（1999—2003年）更为广泛的历史背景，你就会明白其中的原因了。分析一下EJB 1+所提供的核心价值，可以简单地把它们归结为以下3类。

- 框架优势。当时，如果使用Java进行过Web或者企业开发工作，你会发现自己处于一个专有技术的混乱世界中；要么就只能是使用RMI和Servlet这样的基本工具并创建自己的框架。加之当时大多数软件开发者都是随着网络公司的迅猛发展而入行的新手，因此软件行业普遍期待标准的、有共识的企业开发途径。EJB提供了这一途径——一个能够处理事务、安全、有状态组件、对象持久性等的标准框架。这一标准框架解决了实际的和当前的需求——因为当时没有开源运动，也没有出现各种Web框架。
- 分布式优势。EJB标准化了使用分布式对象构建业务逻辑的编程模型和平台。这满足了随RMI和CORBA之后紧接着产生的市场需求。如果希望使用分布式对象，EJB是首选。
- 组件优势。这一点有些古怪。在我看来，Sun公司在此方面的努力对Visual Basic（VB）组件市场成功的一种反应，希望在Java业务组件领域也出现同样的局面。我记得早期围绕EJB的宣传和营销主要集中在组件重用上。甚至有人曾试图为EJB组件构建在线商场。其结果是，为了EJB组件的二进制程序能跨应用程序服务器一致运行，EJB在API、部署和打包语义方面变得非常臃肿。

需要特别提到的是，Java社区对持久性标准／解决方案非常感兴趣，因为当时对象持久性的解决方案非常少，而且自己开发O/R映射程序并不是一件简单的事情。

这就是当时的背景。现在我们讲述EJB 2.1这个阶段，为此先简要回顾一下2003年年末和2004年年

① 委员会设计（Designby Committee）是一种反模式，参阅《反模式》一书（人民邮电出版社）。——编者注

初的情形。在会议上谈这一话题时，我接着会从下面3个方面来介绍社区的方向（这是在EJB 3计划公之于众之前）。

- 框架。这时我们有了一些可提供事务功能、池、安全和所有其他良好程序设计模型的有益于EJB的开源框架，且这些开源框架是通过更加简单和更好的方式来提供这些功能的。诸如XDoclet和Spring这样的工具，以及AOP这样的方法，都将EJB所有的框架优势引入了轻量型环境。最值得注意的是，由于普遍认为实体bean是不良的O/R映射解决方案（实际上，它们是围绕持久性组件，而不是围绕对象设计的），因而Hibernate成为了Java中最流行的O/R映射解决方案，这使得实体bean变得无足轻重了。
- 分布式。RMI不再是Java远程调用事实上的标准；此时已有了其他开源API / 协议。另外，SOA（面向服务的架构）原则也不赞成在分布式中使用远程对象。EJB 2.0中本地接口的引入显得很蹩脚，在很多方面使规范毫无必要地进一步复杂化了。
- 组件。谁还关心它呢？企业级组件的市场消亡了。当Sun关注VB市场的成功时，他们没有注意到市场大部分是由可以在项目内重用的UI小部件和“实用工具”式的组件构成的。我们没有看到工资单类或者看到VB组件形式的“用户”。实际情况是，业务逻辑一般不能在项目之间重用，因此EJB添加的与组件相关的内容都是没有必要的。EJB JAR之间的二进制兼容性也是实际上用不到的一个特性。对于不同的应用程序服务器，通常的做法是简单地把EJB源代码添加到构建中，并且在构建时打包。如果开发可重用组件不是EJB规范制定小组的目的，那么我们在2001年就可能已经有了接近EJB 3的东西。

现在回来思考我经常在演讲中提出的那个调查问题，请注意大部分人没有利用EJB的组件或者分布式特性。他们利用的只是EJB的框架特性，而轻量型开源框架能够更好地提供这些框架特性。在2003年，我认为EJB带来的唯一真正的价值是，用作真正大型系统的分布式对象框架——这是EJB的最初目的。因此，我对听众说：“你们误用了EJB。”我的意思是说，实际使用EJB跨物理层分布式处理业务逻辑的那15%的人没有误用EJB，他们把EJB用于必须使用分布式的情况中，而EJB确实是应对这类情况的正确技术。

那么现在的情况如何呢？依据通过Spring和Hibernate这样的框架在团队中产生的最佳实践，EJB 3最终被重新构造，为另外85%的听众进行了优化。现在我们有了一个功能强大、易于使用并且基于POJO的框架规范，该规范提供事务、安全、O/R映射和分布式，而不是一个用于构造分布式、事务和持久性组件的规范。基本拦截、依赖注入和注解驱动配置的添加也将这几年开发社区中流行的已经检验的最佳实践引入了EJB 3。

还有一件让人高兴的事，那就是我的*EJB Design Patterns*（中文名为《EJB设计模式》）一书（2002年出版）不会再推出新版本了。对于其中的许多模式，都已有解决方案让那85%的听众，不，是那些所有的听众能更充分地使用EJB。

EJB 3终于满足了开发者的需求，因此阅读本书是一种愉快的享受。

Floyd Marinescu
EJB Design Patterns（《EJB设计模式》）的作者
InfoQ.com社区创始人和主编
TheServerSide.com（J2EE社区）创建者

致　谢

虽然我在这里只提到了一些名字，但是必须要说明的是，我非常幸运地得到了朋友、家人、同事和出版社专业人士的大力支持。没有这一强大的后盾，我们可能现在还处在构思本书的阶段呢。

感谢我的朋友Peter Zadrozny，他鼓励我写作本书，并把我引荐给Apress出版社。没有他的经验、构思和指导，本书是不可能完成的。

感谢Apress出版社的Sofia Marchant、Damon Larson和Kelly Winquist，感谢他们耐心地帮助我整理和出版本书。

在顾及日常工作的同时还要写书是很困难的。感谢我的主管Roel Stalman，他鼓励我写作本书，并对此给予了热切的关注。

还要感谢的是Floyd Marinescu，他为本书写了前言，并根据他的专业经验着重介绍了EJB技术的实际情况。

特别感谢我的朋友Sri Rajan一直以来对我的鼓励。他为本书提供了相关的硬件，我们使用这些硬件完成了本书的性能测试部分。

感谢我的父亲Chandra Sekhara Kodali，他对我常说的一句话是：在职业生涯中耐心和坚持总能得到回报，并一直鼓励我勇攀新高。

感谢我的妻子Lakshmi和我们两岁的儿子Yash，感谢他们陪伴我完成了本书的编写。

Raghu R. Kodali

很多人在完成本书的过程中给了我技术支持和灵感，本书是他们努力和智慧的结晶。我要感谢John Bracken和Doug Clarke召开的很多关于EJB和JPA最佳实践的设计会议和讨论。Chris Carter和Michel Trudeau，支持我的追求即使这影响了我对JDeveloper的注意力。他们知道，从研究、测试和编写本书获得的经验肯定会给团队带来收获。在我们频繁地交流和处理本书繁杂事务的过程中，Raghu Kodali让我见识了他的风度、耐心和智慧。Dave Clark为我提供了一个优秀的论坛来检验我的知识并获得反馈的信息。Apress出版社的Steve Anglin、Sofia Marchant、Damon Larson、Kelly Winquist以及他们优秀的编辑团队和出版团队证明了他们是其所在领域的真正专家。

在前面所提到的技术的支持下，如果没有每日的各种消遣作为调剂，围绕我深深喜爱的主题写书只能是一个马拉松长跑的过程！说到这些消遣，我要感谢的人是：David Silver，他使我在艰难的工作中保持清晰的思路。我们在公园长跑时的交流是一种纯粹的精神享受。Adam Beyda，他自始至终都在传授我经验并告诉我判断问题真正关键的角度；Moby Coquillard，他在我编写本书过程中给予的实际引导和在心理方面的帮助是至关重要的。

感谢我的父母Andrea和Peter Wetherbee，他们给予我关爱和鼓励，并且时时提醒我他们是我最有力的支持者。还有Bob和Holly Spain，如果没有他们在每个周末悉心地照顾孩子，从而让我能够进行研究和写作，那么我就不可能完成本书。

最后，完成本书的主要动力是我渴望陪伴我的妻子Laurel和我们的孩子。这是我们在一起度过的最忙碌的一年，当然，我们要祝福我们享有的快乐时光！

Jonathan Wetherbee

目　录

第1章 EJB 3架构简介

我们编写本书的目的是把EJB（Enterprise JavaBeans）3介绍给开发者，深入讨论如何把这种技术应用于日常的实际应用程序中。EJB 3是一种深层次的规范，包含3个文档，面向初级开发者和资深用户，这可是很广泛的读者群，而EJB 3规范的文档作为一部参考指南很好地满足了这一群体。编写一本讲解如何使用EJB 3的书，我们不得不缩小读者的范围；但我们相信本书能够满足大多数Java EE开发者的需求。

本书面向具有Java经验的开发人员，要求他们曾经使用EJB早期版本或者其他技术构造过单层或者多层应用程序，并乐于接受使用最新技术来构建企业应用程序的挑战（以及收益）。考虑到一份800页的参考资料会让人望而却步，因此我们将为开发者逐步进行介绍，每次展开EJB 3的一部分，并且提供他们在实际工作中所需要的信息和代码示例。

展开讲解每一章时，不仅会介绍规范的新内容，而且还会通过专门的示例讲解如何把规范应用到你自己的应用程序中。其中很多示例直接来源于第7章和第12章讲解的综合的完整的Java EE企业酒类在线（Wines Online）应用程序，因此你可以了解它们是如何应用到更大的背景中去的。建议演练这些示例并运行它们。在你常用的IDE或者开发环境中试验这些示例，并且修改它们来尝试新的处理方法。EJB 3——特别是使你可以在EJB容器之外使用实体（即以前的实体bean）的新的容器外持久化特性——可以提供很多功能。熟悉基本的构建、部署和测试之后，你会发现EJB 3组件不仅功能强大，而且容易构建和使用。

我们和Java EE的其他使用者一起使用EJB 3构造了很多应用程序，本书将给出我们所获得的实用模式、所找到的成功策略以及需要避免的某些陷阱。本书的大部分章节主要是介绍EJB 3，但是也包含了从EJB 2.*x*移植到EJB 3、评测EJB 3应用程序的性能以及把你的选择部署到Java EE应用程序服务器的内容。本章最后还有一个介绍性的1.3节，以便让你开始运行本书中许多有用的示例应用程序。

我们希望本书不仅是EJB 3的参考指南，而且也是实际示例的引导性指南和知识库，以供你在构建自己的应用程序时参考。祝大家阅读愉快！

1.1 EJB简介

大约在1996年，随着面向大型应用程序需求的技术（比如RMI和JTA）的出现，Java得到了加强，这继而产生了另一种需求，即一种能够统一这些技术并且能在标准的开发模型下融合这些技术的业务组件框架。EJB的诞生满足了这一需求。在之后的十年中，EJB发展成为包含大量特性（同时排斥其他一些特性）且能在分布式多层环境中部署和执行业务组件的健壮而标准的成熟框架。

1.1.1　EJB是什么

EJB 3由JSR 220：Enterprise JavaBeans 3定义。以前的EJB版本只有一个文档，现在这个JSR扩展到了3个文档（在下面列出）。第一个文档综合介绍了新版本的高级特性，着重介绍了用于构建EJB组件的新的简化开发模型；后两个文档分别讲解核心企业bean框架和持久化模型的技术细节。

- EJB 3简化API（EJB 3 Simplified API）高度概括了新的EJB 3开发模型。
- EJB核心约定和要求（EJB Core Contracts and Requirement）重点讲解会话bean和消息驱动bean。
- Java持久化API（Java Persistence API）讲解实体和持久化框架。

构成EJB 3规范的这三个文档所呈现的既是组件模型也是框架。

1.1.2　EJB组件模型

作为组件模型，EJB定义了开发者可以构建和定制的如下3种对象类型：

- 会话bean（session bean）执行业务服务操作、安排事务和访问控制行为。
- 消息驱动bean（message-driven bean，MDB）是异步调用的，以便通过关联消息队列或者主题（topic）响应外部事件。
- 实体（entity）是具有唯一标识的对象，表示持久的业务数据。

会话bean和消息驱动bean就是EJB[①]，经常被统称为企业bean（enterprise bean）。在EJB的早期版本中，实体被称为实体bean，并且也是企业bean。但是在EJB 3中，实体由持久化提供器管理，而不是由EJB容器管理，并且不再被认为是真正的企业bean。

1.1.3　EJB框架

EJB框架提供了EJB组件在其中进行操作的支持环境，包括容器事务和安全服务、资源的池管理和缓存、组件生存周期服务、并发支持，等等——这些本书中都将进行探讨。EJB组件使用EJB专有的元数据（要么在运行时由容器捕获并被应用到EJB行为，要么在EJB组件部署到EJB容器时被解释并用于构造包装），指定如何与支持它们的容器进行交互的细节。

1.1.4　EJB核心特性

在其发展过程中，EJB一直专注于提供具有一些核心特性的组件，而EJB 3全面改进了这些特性。

1. 声明式元数据

EJB组件模型的特点之一是开发者可以使用他们所选择的JDK 5.0注解（annotation）和/或XML描述文件声明式地（和编程式相反）指定企业bean和实体的行为。这样在很大程度上简化了开发过程，因为很多定制处理可以添加到bean，而无需使用Java编写任何逻辑。为了适应开发者的偏好和应用程序的灵活性，EJB 3允许开发者选择使用注解和XML，可同时在同一个EJB或者实体内使用这两种方法，来指定元数据中的行为。在同时使用注解和XML定义相同元数据片段的情况中，XML声明具有优先权，以避免冲突。这种方式的其他优势将在1.1.6节中介绍。

2. 按异常配置

与声明式指定行为相结合的，是EJB 3中大量使用的智能默认值。很多行为被自动附加到EJB或者实体中，无需显式地声明，比如会话bean方法的事务行为，以及用于持久化实体及其公有属性的表和列的名称。只有需要非默认行为时，才需要显式地指定注解或者对应的XML。在大多数常见的情况（其

① 请注意，EJB这个缩写词既可指规范和技术，也可以指具体的组件，后者等同于bean或企业bean。——编者注

中需要默认行为）中，这种方式将产生非常稀疏、清洁的代码。这种开发模型称为按异常配置（configuration by exception），因为只有在异常（非默认）情况下才必须显式地配置bean的行为。

3. 可伸缩性

大型应用程序要求具有随着客户负载的增长而伸缩的功能。EJB服务器使用资源池达到对象重用的最大化，使用持久化缓存避免重复查询或者创建相同对象，并且在中间层实现优化的锁定策略，以便降低关系数据库管理系统（relational database management system，RDBMS）的负载以及避免并发锁定的情况。

4. 事务性

Java事务API（Java Transaction API，JTA）为分布式事务定义标准API，并且EJB服务器作为EJB的JTA事务管理器。从一开始，EJB规范就定义了标准模型，用于在企业bean上声明式地指定容器管理的事务行为。

5. 多层安全

可以声明式地指定EJB方法级别的访问控制，由应用程序服务器管理员强制定义用户级别和角色级别的权限。

6. 可移植性

至少从理论上说，合乎规范的企业bean可以部署到实现EJB的任何应用程序服务器上。在实际应用中（对EJB 3之前的版本尤其如此），厂商提供它们自己的元数据定义，企业bean开发者们逐渐对它们产生依赖，把它们锁定到特定厂商的实现中。随着EJB不断成熟，它不断地合并了很多这些平台原先专有的特性，所以和过去相比，现在实现的EJB的可移植性要高得多。

7. 可重用性

EJB是松耦合的组件。可以重用或者打包到多个应用程序中，虽然EJB必须捆绑或者访问其他相关的EJB。

8. 持久性

实体bean——在EJB 3中被替换为POJO（plain old Java object，普通旧式Java对象）类，简称为实体——代表具有唯一标识的持久化域对象。实体类对应数据库表，每个实体实例由这个表中的单一行表示。

1.1.5 EJB 规范的发展

每次EJB的新版本发布时，都会带来新的重要特性，满足大众需求并采用新兴技术。下面是从1996年EJB出现以来，或者说更重要的是从1998年EJB的第一个商业实现以来，其规范发展的简要总结。

1. EJB 1.0

最初的版本1.0开始支持有状态和无状态服务对象（称为会话bean），以及可选支持持久化域对象（称为实体bean）。在可移植性方面，EJB通过提供可移植性和远程特性的专门远程接口达到访问的目的，但是受到了远程基础结构和按值传递的语义的开销的影响。

2. EJB 1.1

其后的版本1.1要求厂商支持实体bean，并且引入XML部署描述文件来替换存储在专门的串行化类文件中的元数据。

3. EJB 2.0

EJB 2.0通过引入本地接口解决远程接口造成的开销和按值传递的缺陷。只有运行在J2EE容器内的客户才能通过其本地接口访问EJB——但是按引用传递的方法调用允许组件之间更加有效地进行交换