HZ BOOKS

Sun公司核心技术丛书

J2EE核心模式

Core J2EE Patterns

Best Practices and Design Strategies

Sun microsystems

本书由Rational公司首席科学家Grady Booch作序

Deepak Alur
(美) John Crupi 著
Dan Malks

牛志奇 丁天 田蕴哲 等译

机械工业出版社
China Machine Press

PH PTR

Sun公司核心技术丛书

J2EE核心模式

Deepak Alur
（美） John Crupi 著
Dan Malks
牛志奇 丁天 田蕴哲 等译

机械工业出版社
China Machine Press

本书主要描述 J2EE 关键技术的模式、最佳实践、设计策略和经过验证的解决方案。涉及 J2EE 包括的 15 个模式的分类和大量的策略，可以使读者更好地掌握 Java 技术。本书适合 J2EE 的爱好者、程序员、设计师、开发者和技术管理者参考。

本书版权登记号：图字：01-2001-3871

图书在版编目（CIP）数据

J2EE 核心模式/（美）阿卢尔（Alur, D.）等著；牛志奇等译．-北京：机械工业出版社，2002.1

（Sun 公司核心技术丛书）

书名原文：Core J2EE Patterns: Best Practices and Design Strategies

ISBN 7-111-09511-1

Ⅰ.J… Ⅱ.①阿…②牛… Ⅲ.Java 语言-程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2001）第 078843 号

机械工业出版社（北京市西城区百万庄大街 22 号 邮政编号 100037）

责任编辑：高智勇 张鸿斌

北京昌平奔腾印刷厂印刷 ·新华书店北京发行所发行

2002 年 1 月第 1 版第 1 次印刷

787mm×1092mm1/16·18.75 印张

印数：0 001-5 000 册

定价：35.00 元

译 者 序

最近几年，J2EE 技术作为建立企业应用的标准平台出现，并且逐渐成熟。该平台为开发和配置企业应用提供坚实、稳固服务的同时，也显示出了它的挑战性。开发者经常无法分清学习技术和利用技术学习设计的区别。在本书中，Sun Java Center 的资深设计师、Sun Java 咨询机构等与读者一起分享他们丰富的 J2EE 技术经验。

本书主要描述 J2EE 关键技术的模式、最佳实践、设计策略和经过验证的解决方案。J2EE 关键技术包括 Java Server Pages（JSP）、Servlet、Enterprise JavaBeans（EJB）、Java Message Service（JMS）等。其他的 J2EE 技术，如 JDBC 和 JNDI，也因与模式相关而在本书中有所描述。J2EE 所包含的 15 个模式的分类和大量的策略均为本书所述的内容，它们可以使读者更好地掌握Java 技术。

本书不但具有一定的理论深度，而且还非常实用。通过对本书的学习，完全可以理解在理论部分所讲解的知识，最终实现理论和实践的紧密结合。

我们很荣幸有机会承担本书的翻译工作。在翻译过程中，我们经常为一句话、一个术语进行反复的讨论，到处查找资料，力图使本书能正确、贴切地反映原文的意思，同时注意使句子、段落符合中国人的语言习惯。我们真挚地希望读者能够从本书中有所收获，这是作者的初衷，也是我们的良好愿望！

本书由牛志奇、丁天、田蕴哲组织翻译，万方工作室的全体同仁都参加了本书的翻译、校正和录入等工作。参加本书翻译、录排、校对工作的人员为：牛志奇、丁天、葛丽、罗贤锋、罗浩、王宾、赵伟、夏欣伟、李红玲、孙南、田敏、龚露娜、马军、马丽、田军、牛献忠、田蕴哲、金荣学、薛彪、叶哲、邓海燕、邢倩、王育红、李军、刘彬、钱斌、赵策、姜南、李浩、王天凌、李林、张巧莉、范玉春、袁雷、邓涛、李卓林、聂宛析、王小将、李素丽、天海鹏等。本书的出版是集体劳动的结晶，在此特别感谢万方工作室的全体工作人员。

由于时间仓促，且译者经验和水平有限，译文难免有不妥之处，恳请读者批评指正。

万方工作室

2001 年 8 月

序　言

在软件行业里，模式最能体现一个公司所具有的特点。模式对普遍的问题提供了通用的解决方法。在某一个公司或者在某一个领域内，命名并确立一个模式往往意味着正式形成了一个普遍的、被以往经验证实正确的解决方案。在开发中，拥有一个好的模式语言等于拥有一个强大的专家队伍：通过使用众多模式中的一个，就会从专家们来之不易的知识中获取极大的收益。好的模式也并不是大量存在的，因为首先要发现模式，然后在现有系统中运行成功之后再推出。因此，在一个成熟的模式中，包含大量的去粗取精、去伪存真的过程，也包含设计者总结出来的大量基本原理和启示。

模式很深奥，很有用且早已存在，你也许会说："我以前用过。"其实真正的"模式"你以前也许没用过，这样的模式最终将会使你的系统变得更为简单。

模式不仅可以帮助你建立简单可执行的系统，而且可以帮助你建立完美的程序。在时间就是金钱的今天，写一个十全十美的程序往往是不可能的，更多的时候只是努力去创造合格的产品，这对于专业人士来说不能不是一种遗憾。但是，如果能够较好地使用模式，完全可以提高系统的完美性。否则，是不可能做到这一点的。

本书的作者在书中贡献了一系列非常有实用价值的模式。J2EE 确实是一个非常了不起的平台，它能使团队开发出功能非常强大的系统。然而事实上，在 J2EE 提供的抽象和服务与团队最终必须建立的应用之间仍然存在着很大的差距。本书所涉及的模式正是一步步缩小这种差距的解决方案。通过应用这些模式，可以找到降低软件风险的主要路径：编写更少的软件。你不用自己去发掘这些解决方案，相反，可以直接使用现成的模式，因为它们已经在现有系统中被证实是正确的。

除了命名一系列模式以外，作者还使用 UML 来详细解释它们的语义，以便模式更容易被理解。另外，本书还说明了如何应用这些模式，如何利用它们来重组系统等。总之，还是前面的那句话：利用模式就等于拥有一个强大的专家队伍。

Grady Booch

Rational 软件公司首席科学家

前　言

本书主要讲述企业版 Java 2 平台（J2EE）的模式。J2EE 模式为 J2EE 平台软件应用设计者提供典型问题的解决方案。本书中所有模式都经过了用户的 J2EE 应用验证。

本书主要描述经过验证的 J2EE 平台解决方案，着重于 J2EE 的以下关键技术：Java 服务器页面（Java Server Pages，JSP）、Servlet、Enterprise JavaBeans（EJB）组件、Java 消息服务（Java Message Service，JMS）、JDBC 和 Java 命名与目录接口（Java Naming and Directory Interface，JNDI）。通过 J2EE 模式目录和 J2EE 重组来提供对 J2EE 平台重复出现问题的解决方案。你可以应用这些思想开发新系统或完善现有系统的设计。本书中的模式可以使读者快速掌握并精通开发强壮、高效企业应用系统的技能。

无论是现在还是过去，很多人都错误地以为学习技术和学习设计是一回事。当然，学习技术是进行成功设计的重要环节，因为设计需要技术。现在有很多优秀的 Java 书籍，它们只讲述技术细节，却不涉及如何应用技术。学习设计则与此不同，它主要来自成功或失败的经验和教训。

本书讲授的经验来自我们在这个领域内的亲身实践。我们隶属于 Sun Microsystems 公司的 Sun Java Center（SJC）咨询机构。在我们的工作中，经常会遇到这样的情况：由于技术发展得太快，设计者和开发者们总是费力地去理解和掌握技术细节，却不会想到如何利用技术来进行设计。

只是一味地指导设计者和开发者去写好的代码并不是一个好主意，或者只是建议使用 Servlet 和 JSP 去开发表示层和使用 EJB 组件[○]去开发业务层等也是不够的。

因此，从以上分析中我们就可以回答下列问题：一个积极的 J2EE 设计者不仅要知道应该做什么，而且还应知道不该做什么。什么是最佳实践？什么是失败的实践？如何才能成功实现从问题到设计到实施的过程？

Sun Java Center（SJC）和 J2EE 模式目录

从诞生的那天起，SJC 的设计者就和全世界的用户一起去成功地设计、构筑、建造和发布各种基于 Java 和 J2EE 的系统。SJC 是一个快速成长的咨询机构，并且至今还不断地有经验丰富的设计师加入。

当我们意识到需要捕捉和共享那些经过验证的设计和构架以后，便在 1999 年开始将我们对 J2EE 平台的经验积累整理、归档成模式的形式。尽管我们参考了现有的许多文献，但还是无法找到一个可以代表 J2EE 平台模式目录。我们发现，很多文献都涉及一个或多个 J2EE 技术细节，并且详细地讲述了这些技术细节，甚至介绍了最细微的差别。有些书籍还通过描述一些

○ 对于初次接触 J2EE 平台的读者，请参见第 2 章“J2EE 平台概述”，主要讲述 J2EE 平台和这些相关技术。

设计注意事项来扩充这些技术细节。

自从 2000 年 7 月我们在 JavaOne 大会上首次公开宣传我们关于 J2EE 模式的思想后，得到了来自设计师和开发者们热烈的响应。一些人表示很想进一步了解模式；而另一些人则坚持这并不是一个新鲜事物，因为他们以前应用过模式，只是从未命名它们或形成文档而已。总之，所有这些对 J2EE 平台模式的热情关注促使我们将此项工作继续进行下去。

因此，我们开始整理 J2EE 平台的模式目录，并于 2001 年 3 月在 Java Developer Connection 中首次形成，同时将 beta 版在整个 J2EE 社区中试用。在广泛的 Java 团体反馈基础上，我们以 beta 版为基础，形成了本书的内容。

我们希望这些模式、成功的实践和策略、失败的实践以及 J2EE 平台的重组，能够与带给我们好处一样，也给读者带来收获。

本书主要内容

本书主要讲述以下内容：

- 使用 J2EE 平台模式。

在搜集和整理关于 J2EE 平台经验的基础上，我们在本书中形成了模式目录。J2EE 模式目录描述了与设计和构筑 J2EE 平台应用有关的各种成功实践。本书着重于下列四种 J2EE 技术：Servlet、JSP、EJB 组件和 JMS。

- 应用成功的实践去设计应用（使用 JSP、Servlet、EJB 组件以及 JMS 技术）。

仅仅学习技术细节和 API 是不够的。学习使用技术如何进行设计同样重要。我们整理出的经验，是 JSP、Servlet、EJB 组件和 JMS 等技术的成功实践。

- 形成 J2EE 平台的设计和构架，可以避免他人重复劳动。

模式可以使设计重复使用。重复使用已知的解决方案可以缩短设计和开发应用的周期，包括 J2EE 的应用在内。

- 识别现存设计中失败的地方，使用 J2EE 模式重组为更好的解决方案。

知道可以做什么当然好，但知道不可以做什么也同样重要。因此，本书也记述了一些我们经历的设计 J2EE 平台应用失败的实践。

本书不包括以下内容：

- 如何使用 Java 或 J2EE 技术编程。

本书不讲述如何编程。尽管本书内容很大程度上以 J2EE 技术为基础，但我们却不讲述具体的 API。如果读者想要学习 Java 或 J2EE 编程，有许多很好的书籍和在线资源可供参考。如果读者想要学习具体的技术细节，我们特别推荐使用位于 Java 官方网站 http: // java. sun. com 上的在线教程。在该网站的主页上还有正式的 J2EE 技术详述。

- 使用什么样的过程和方法论。

我们不建议使用任何类型的过程或方法论，因为本书所述内容与这两者均无关系。因此，本书不会讲述任何关于项目中的过程或方法论。如果读者想了解过程和方法论，有许多阐述各种面向对象方法论的书籍和有关初级过程的新书，如《Extreme Programming》（超级编程）等。

- 如何使用 UML。

本书不讲述有关 UML 的知识。我们广泛地使用 UML（特定的类和图表系列）的目的是来归档模式、描述静态和动态的交互操作。如果读者想更多地了解 UML，请参考 Grady Booch、Ivar Jacobson 和 James Rumbaugh 所著的《UML User Guide》（UML 用户指南）和《UML Reference manual》（UML 参考手册）。（这两本书已由机械工业出版社引进出版。——编者注）

本书适用读者

本书适用于所有 J2EE 的爱好者、程序员、设计师、开发者和技术管理者。一句话，适用于对设计、构筑和开发 J2EE 平台应用的所有人。

我们希望本书能成为 J2EE 设计师有益的培训指南。因为我们知道一个好的设计项目的重要性，因此我们需要本书来指导。

使用这些经过整理的模式、成功的实践和失败的实践去共享、传递知识和经验，可以提高开发团队的整体素质和产品价值，我们衷心地希望本书能够做到这一点。

本书结构

本书分为三部分：

第一部分：模式和 J2EE。本部分由第 1 章和第 2 章组成。

第 1 章：引言。简要介绍一些概念，包括模式、J2EE 平台、模式定义和模式目录等，最后介绍 J2EE 模式的类型。

第 2 章：J2EE 平台概述。为 J2EE 初学者全面介绍 J2EE 平台概况，也适用于想了解 J2EE 最新内容的读者。

第二部分：设计注意事项、失败的实践和重组。主要介绍 JSP、Servlet 和企业 beans 的设计注意事项。此外还包括 J2EE 平台失败的实践和重组。本部分由第 3 章、第 4 章和第 5 章组成。

第 3 章：表示层设计注意事项和失败的实践。

第 4 章：业务层设计注意事项和失败的实践。

第 3、4 章分别讲述了表示层和业务层设计注意事项和失败的实践。设计注意事项指出了 J2EE 的开发者、设计者、设计师应用 J2EE 平台时需要考虑的问题。该章节中涉及到的概念，读者可参照其他文献（如正式的说明书或有关这些概念的书籍），以获得更多的信息。

第 5 章：J2EE 重组。讲述来自我们实际工作中的重组经验，这些重组使原本不是很好的解决方案变得更加完善。重组是本书后面要讲述的另外一个重要内容，我们相信它对模式目录有非常大的帮助。本章内容很大程度上得益于 Martin Fowler 和他的著作《Refactoring》。对于了解《Refactoring》的读者来说，一定非常熟悉本章的形式，但本章内容全部是关于 J2EE 技术层面的，而 Martin Fowler 的书则从另外的一个角度来描述重组。

第三部分：J2EE 模式目录。讲述了 J2EE 模式的类型。J2EE 模式目录体系中包含 15 个模式类型，这些模式类型构成了本书的核心内容。本部分由第 6 章、第 7 章、第 8 章、第 9 章和第 10 章组成。

第 6 章：J2EE 模式总览。概要介绍了 J2EE 模式类型。从本章开始详细介绍模式的思想，

解释了将模式分为层的方法。同时还说明了本书中讲解模式所用的模板。本章讲述了所有J2EE的模式类型，并使用图表说明它们之间内在的联系。此外，还在模式类型中提出了“路标”的概念。“路标”向公共J2EE设计和结构相关的问题给出提供这些问题解决方案的模式和重组。理解模式之间的关系和路标的概念是应用的关键。

第7章：表示层模式。介绍了与使用Sevlet、JSP、JavaBean和自定义标签来设计J2EE平台基于Web应用相关的六个模式类型。在这些模式中，提供了大量的实现策略，并且讲述了一些具有普遍性的问题，如请求处理、应用分割和生成复杂的界面等。

第8章：业务层模式。介绍了与使用EJB技术来设计J2EE平台业务组件相关的七个模式类型。该章中的模式提供了使用EJB和JMS技术最好的实践。同时，这些模式也涉及了其他方面的技术，如JNDI和JDBC等。

第9章：集成层模式。介绍了与用于集成J2EE应用、资源层及外部系统相关的两个模式类型。这些模式主要应用于使用JDBC和JMS集成业务层和资源层的组件。

第10章：J2EE模式应用。介绍了使用这些模式的一个具体实例。探讨并展示了这些模式是怎样联系在一起的。该章强化了这样一个概念：模式存在于一个相互关联的环境中。每一个模式都支持其他模式，同时也被其他模式所支持。

联系信息

在官方网站上，我们将提供本书的更新信息和其他参考资料，具体地址为：http://www.phptr.com/corej2eepatterns。

J2EE模式兴趣小组的邮件地址为：j2eepatterns-interest@java.sun.com。该地址向大家开放，欢迎参加。欲加入该兴趣小组并浏览相关文档，请访问：http://archives.java.sun.com/archives/j2eepatterns-interest.html。

目　录

第三部分 J2EE 模式目录

第一部分　模式和 J2EE

第一部分包含两章：

- 第 1 章——引言。
- 第 2 章—— J2EE 平台概述。

第 1 章是模式和 J2EE 的高级概述。主要介绍了多种模式定义、有关模式目录的信息以及应用模式的好处等。本章定义了 J2EE 模式工作的环境，提出了 J2EE 模式目录的基本原理和出发点。

第 2 章提供了 J2EE 平台的高级概述、J2EE 背景以及平台价值等。本章还阐述了 J2EE 平台和 J2EE 模式目录的关系。

第1章 引　言

本章主要包含如下内容：

- 什么是 J2EE?
- 什么是模式?
- J2EE 模式目录。
- 模式、构架和重用。

近些年来，人们非常重视企业软件开发的前景，Java 2 企业版平台（J2EE）在此过程中备受关注。J2EE 为分布式和服务器为主的应用提供了统一的平台，并且通过广泛采用各种策略，提供开放和标准的开发环境，J2EE 使企业以服务为基础的应用系统得以建立。

与此同时，人们经常将学习 J2EE 技术和学习使用 J2EE 技术进行设计二者混为一谈。现存的许多 Java 书籍在阐述具体技术细节方面做得非常出色，但是却很少去描述如何应用技术进行设计。

J2EE 设计师不仅应当懂得相关的 API，而且还应该知道：

- 什么是最佳实践?
- 什么是失败实践?
- 什么是普遍重现的问题，什么是对应的经过验证的解决方案?
- 如何重组代码和应用模式使得一个不够优化的应用或失败的实践达到更完美的程度?

以上就是本书所要讲述的全部内容。好的设计源于工作中经验的积累。当设计使用标准的模板以模式的方式进行交流时，模式就成了交流和重用的强大机制，并且可以改善设计和开发软件的方式。

1.1 什么是 J2EE

J2EE 是开发分布式企业软件应用的平台。Java 语言自诞生以来，经历了大量的扩充和发展。越来越多的技术被融入 Java 平台中，并且不断有新的 API 和标准被开发，以便更好地适应不同的需求。最终，Sun 和一些工业巨头在开放的 Java Community Process（JCP）的基础上，将所有企业相关的标准和 API 统一到了 J2EE 平台上。

J2EE 平台为企业应用提供了大量好处：

- J2EE 在各种领域内创建了适用于企业计算需要的一系列标准，如数据库连接、企业业务组件、面向信息的中间件（MOM）、Web 相关组件、通信协议、协同工作等。
- J2EE 基于开放的标准，可以使先进的技术更好地发展，保护技术投资。
- J2EE 提供开发组件的标准平台，适用于不同的供应商，有效地避免了供应商独立封闭的局面。
- J2EE 缩短了产品投入市场的时间。绝大多数供应商的产品的基础架构或部件都是遵循

J2EE 指定标准而开发，因此，现在的 IT 机构可以摆脱中间件的困扰而专注于自己的业务应用。

- J2EE 提高了编程效率，因为 Java 程序员可以在 Java 语言基础上相对容易地掌握 J2EE 技术。所有企业软件的开发都可以在 J2EE 平台上使用 Java 作为编程语言来实现。
- J2EE 提高了现有不同种类开发环境的协同工作能力。

我们将在第 2 章更详细地讨论 J2EE 平台。若想了解更多的内容，请参阅第 2 章。现在，我们已对模式有了简单的认识，如它的形成、J2EE 模式目录中的模式种类等。更多的有关模式目录的详细内容请参阅本书的第三部分。

1.2 什么是模式

1.2.1 模式的形成历史

在 20 世纪 70 年代，Christopher Alexander 发表了很多关于工程和建筑模式方面的书籍。随后，软件行业才逐渐接纳了这种最初建立于工程和建筑方面的模式的思想，尽管在此之前软件业中已有此种思想的萌芽。

软件业中模式概念的普及是由 Erich Gamma、Richard Helm、Ralph Johnson 和 John Vlissides 四人（又被称为“GoF”或“四人帮”）合著的《设计模式：可重用的面向对象的软件元素》（Design Patterns: Elements of Reusable Object - Oriented Software）一书引发的。今天，当 GoF 提出的“模式”一词已成为全世界软件开发中一个共同的话题时，我们应当记住：模式不是这些作者发明出来的，相反，是他们意识到大量的项目中有可重用的设计思想后，将这些设计思想收集、分析并加以整理而形成了文字。

在 GoF 的书问世之后，许多软件模式方面的书籍相继出版。这些书籍覆盖了各种领域，有着各种各样的用途。我们在本书后面列出了这些书籍的名字供大家参考，并希望读者继续探索这些书籍中没有提到的其他类型的模式。

1.2.2 模式定义

模式用来描述所交流的问题及其解决方案。简单地说，模式可以帮助我们在一个特定的环境里整理并记录已知的可重现的问题及其解决方案，并且通过模式来与他人交流这些知识。这个描述中，一个很关键的词就是“重现”。因为模式的目标就是提倡随着时间的推移，在概念上进行重用。

我们将在第 6 章的“什么是模式”一节中详细探讨这个问题。

以下是我们收集的一些有关模式的著名定义。首先介绍 Christopher Alexander 在《Pattern Language》（模式语言）一书中关于模式的定义：

每个模式都是由三部分组成的一个规则，这个规则描述特定环境、问题和解决方案之间的关系。

—— Christopher Alexander

Alexander 后来扩展了这个定义，并且解释了模式的特征。而 Richard Gabriel [Gabriel] 更详尽地论述了这个定义。在 Alexander 定义的基础上，他提出了自己的应用于软件的模式定义：

每个模式都是由三部分组成的一个规则，这个规则描述特定环境、特定系统作用，以及特定软件配置之间的关系。其中特定系统作用力可以在特定环境中反复出现，并且特定软件配置可以使特定系统有能力解决自身存在的问题。（请参考 A Timeless Way of Hacking。）

—— Richard Gabriel

这可以算是一个严密的定义，但也存在许多不足。例如，Martin Fowler 在《Analysis Patterns》（分析模式）[Fowler2] 一书中提出了下列定义：

模式是一种思想，它在特定的环境中起作用，并且也可能在其他环境中起作用。

—— Martin Fowler

正如你所见，现在有很多关于模式的定义，但是所有定义都有一个共同的主题：与在特定环境下成对出现的问题/解决方案的重现相关。

模式的共同特征如下：

- 模式来源于经验。
- 模式总是以一种结构化格式记录（参见 6.4.3 节“模式模板”）出现。
- 模式的出现避免了重新设计和创造。
- 模式存在于不同程度的抽象当中。
- 模式总在不断地被完善。
- 模式是可重用的人为总结的经验。
- 模式可以用来交流设计和最佳实践。
- 多个模式可以一同使用，以解决复杂的问题。

很多伟大的思想家花费了大量的时间试图去定义模式或精炼软件模式的定义。我们不想成为什么伟大的思想家，也不想浪费时间去扩大这些讨论的范围。相反，我们专注于简单和重现这两点，力图将这些表面上五花八门的定义变成现实。

1.2.3　模式分类

模式代表着对特定环境中重现问题的专业解决方案。因此，我们可以在多个抽象的层次和多个领域内定义模式。软件模式的分类方法有很多，常用的模式分类方法有：

- 设计型模式。
- 构架型模式。
- 分析型模式。
- 创建型模式。
- 结构型模式。
- 行为型模式。

上述分类列表尽管简单，但也存在多种层次的抽象和交叉的分类体系。因此，尽管有许多理论上的分类法做指导，我们还是无法找到一个正确的分类方法来整理和归档这些思想。

我们将分类中的模式简单地称为 J2EE 模式。每个模式作用于设计模式和构架模式之间的某些方面，而策略则是在较低的抽象层次记录每个模式。我们这里只介绍一种分类方式，即在下列三个逻辑结构层内对模式进行分类：