"J2EE开发者的必读之作，振兴模式运动的经典论著。"
——Martin Fowler，ThoughtWorks首席科学家

Sun 公司核心技术丛书

（原书第2版）

J2EE 核心模式

Core J2EE™ Patterns

Best Practices and Design Strategies, Second Edition

（美） Deepak Alur John Crupi Dan Malks 著
刘天北 熊节 等译

经过更新和修订的最新版本！

- 由Grady Booch和Martin Fowler作序
- 来自Sun Java中心的J2EE模式目录
- 用于J2EE和Web Service的模式和策略
- J2EE开发中的重构和不佳实践
- 各种模式、策略和重构的示例代码

机械工业出版社
China Machine Press

Sun 公司核心技术丛书

（原书第2版）

J2EE 核心模式

Core J2EE™ Patterns

Best Practices and Design Strategies, Second Edition

（美） Deepak Alur John Crupi Dan Malks 著

刘天北 熊 节 等译

机械工业出版社
China Machine Press

本书讲解使用J2EE核心技术实现企业应用过程中的模式、最佳实践、设计策略以及经过验证的解决方案，涵盖了JSP、servlet、EJB、JMS等技术，其中J2EE模式目录包括21个模式以及大量策略，帮助读者迅速熟练掌握J2EE技术，从而构建出健壮、高效的企业应用。本书是计算机应用开发人员、架构师、技术经理等人员的必备参考书。

本书版权登记号：图字：01-2003-4987

图书在版编目（CIP）数据

J2EE核心模式（原书第2版）/（美）阿卢（Alur, Deepak.）等著；刘天北等译. -北京：机械工业出版社，2005.3

（Sun公司核心技术丛书）

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

ISBN 7-111-15942-X

Ⅰ.J… Ⅱ.①阿… ②刘… Ⅲ.Java语言-程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2004）第142282号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）

责任编辑：王 晶 李云静

北京瑞德印刷有限公司印刷 · 新华书店北京发行所发行

2005年3月第1版第1次印刷

787mm × 1092mm 1/16 · 32.5印张

印数：0 001-5 000册

定价：55.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

对《J2EE核心模式》的好评

“《J2EE核心模式》的作者们提取了一组真正实用的模式。他们介绍了应该如何应用这些模式、如何重构系统以便从模式中获益。拥有这本书就像有一个专家组坐在你旁边一样。”

——Grady Booch，Rational软件公司首席科学家，序言节选

“本书是对J2EE模式的一次出色的汇集。它固化了J2EE开发的重要经验。没有这本书，就别开发EJB……而且，越来越多的人选用“重构”的方法，来对现有系统做出更动。本书的作者把我在重构方面的研究应用在一个新方向——也就是J2EE设计的世界之中，这样做还尚属首次。我不仅因为有人在我的研究基础上进行工作而心怀感激，而且，读到他们依据自己的经验，实际描述了如何进行系统重构的转换，我也感到非常喜悦。”

——Martin Fowler，ThoughtWorks 首席科学家，序言节选

“《J2EE核心模式》是一本福音书，应该与每个J2EE应用服务器一起配套发售。这本书提供了一套明确无误的、经过实战检验的模式语言以及多种重构策略，对于在真实环境下设计、实现、维护健壮的J2EE应用系统很有帮助。本书的作者都是J2EE架构设计战壕中的老兵，本书总结了他们的多年经验，囊括了J2EE平台下可供架构师使用的多种技术和API，令人信服地解答了为何、何时、如何使用J2EE平台。”

——Sean Neville，Macromedia公司JRun企业版架构师

“作者们介绍了大量对于应用程序架构极有帮助的模式，这是一项了不起的工作。单单是书中的‘重构’部分就值整本书的价钱！”

——Craig McClanahan，Struts首席架构师

“随着J2EE平台渐趋成熟，在应用系统设计的过程中采用成熟的工程原则，对目前来说比以前更为重要。有很多架构师、分析师都要决定使用何种策略在J2EE平台上实现企业应用，对于他们，《J2EE核心模式》是一部价值连城的技术指南。书中详细分析了用于各种最佳实践的通用模式，同时也讲解了避免不佳实践的方法。”

——Craig Russell，JDO架构师

译者序

为一部由Grady Booch和Martin Fowler作序的作品写序言，这个念头本身就足够荒谬和僭越，不啻于在莎翁之后再写一个关于丹麦王子复仇的剧本。大师们的判断是中肯而毫不含糊的："没有这本书，就别开发EJB。"他们的担保足以确认本书在其论域中舍我其谁的地位。是的，这就是"那本"J2EE书。

当然，对于广大中国开发者而言，我们早就已经在"没有这本书"的条件下开发了大量J2EE乃至EJB应用系统。那些波折的、不乏磨难的开发历程似乎使不少人具备了一种不无理由的自信，在掌握了若干API细节、若干应用服务器配置诀窍、若干框架类库用法之后，他们或是公开、或是暗自地把自己当成了当之无愧的Java企业开发专家。——不，这些话没有任何揶揄的意思：我们想说的其实是，本书恰恰是为以上这一类开发者写的。对于他们想成为"Java企业开发专家"的隐秘欲望，本书就是最大限度的补救和成全。如果说，此前的各种教程都是在介绍J2EE开发中的"内容"要素——也就是，教给我们"做什么"的话，本书关注的则是这里的"形式"要素，即"怎样做"才能开发出高效的、优雅的J2EE系统。读者从中学到的，将不仅仅是"J2EE技术"，而是"如何使用J2EE技术进行设计"。

换句话说，如果你以前没有进行过J2EE实践，但明早将应聘一个需要"1年J2EE开发经验"的职位，本书中不包含你今晚要彻夜吞咽的那一类知识；相反，如果你，这位未来的"Java企业开发专家"，追求的职位是"资深Java应用系统架构师"，如果你预料到未来的上司明天将问起"怎样实现访问控制"、"何时采用细粒度的接口设计"等"高阶"问题，那么恭喜你，今晚——乃至今后——阅读本书，你选对了补课的读物。

作为本书第1版的忠实读者，我们（半是欣喜、半是惊讶地）发现，眼前的这部第2版构成了全新的阅读体验。作者们按照最新版J2EE技术规范（尤其是EJB 2.1）全面修订了技术细节；根据模式社区的研究交流，作者们补入了若干模式；即使是一些不涉及技术更新的部分，论述方式、示例也完全不同于第1版；原有的PSA项目（第1版"尾声"一章）融入了其余各章的"示例代码"部分；而新增的讨论"微架构"的尾声、对Web Service等技术的关注、对各种的持久化方案（定制持久化、EJB、JDO等）的深入讨论，都体现出作者们对本书新版的大量投入。

受益于本书有年，在此，我们想冒昧地为本书的中国读者们建议一条高效的阅读路径：与第1章相比，第5章"J2EE模式概览"是读者更合理的起步点。请特别关注其中对"分层"、"术语"和模式/策略区别的讨论，这些都是贯穿全书的重要概念！其次，应该通读第2章"表现层设计考虑和不佳实践"和第3章"业务层设计考虑和不佳实践"：即使你不打算使用任何模式，甚至，即使你根本不关心J2EE开发，只要你的工作与分布式企业应用系统有关，这两章涉及的问题都是你迟早会遇到的。至于每个具体模式本身，我们则推荐读者留意其中详尽的"策略"部分和那些散布其中的"设计手记"。前者讨论了对同一个模式的多种实现方案，后者则突出介绍了特定开发领域的一些核心概念和考虑。

一部英文技术论著在汉语中的旅行，永远是一段难以捉摸的行程。对于本书的汉语译者，“技术难度”并非挑战：全书讨论的正是译者们最为熟知的一个领域，所以我们能够负责任地说，在这个中译本里，没有任何技术细节会因为译者的无知或生疏而发生变形或曲解。这次翻译的原则和前提是对原文的彻底领会。事实上，译者在翻译工作中遇到的困难主要发生在“语汇”层面。简单地说，J2EE专著的译者总要面对“翻，还是不翻”的两难处境：对象、函数的名称，UML图中的各种元素，这些内容由英语表示早就是约定俗成，即使是英语程度略低的开发者大概也都能读懂，所以，在读者能够理解的部分尽可能保留原文似乎是一种合理的做法——毕竟开发工作最终是与代码有关，而代码则肯定是要用“英文”的。但在另一方面，翻译的责任就在于让不谙英文的读者也能通达作品，如果译文中大量段落（不包括示例代码）都仍保留为英文或“类英文”，那么读者也就无法直观地获得原文包含的信息。反复权衡之后，在这个译本中译者的解决方式还是折衷的。工作中我们采取了以下原则：

1) 术语尽可能采用通用文献定译，不自创译法。对于各个模式的名称、模式文档模板各部分名称、重构手法名称，我们参考了李英军等译《设计模式》（机械工业出版社，2000年）、熊节等译《重构》（中国电力出版社，2003年）等译作，以及IBM DeveloperWorks中文网站的部分资源。

2) 本领域的一些常见术语，如果没有定译，本书也不自创新语，强译为中文，而是保留英文原字。这一类的术语包括：applet、servlet、bean、JavaBean、entity bean、session bean、EJB、finder、Context、cookie、RowSet、null、scriptlet、Web Service。根据我们的观察，国内的开发者在日常工作中已经习惯按原文使用以上术语。在一些情况下，我们也以注释形式澄清了这些术语的用法。另外，一些非常直观的英文表达方式，比如“versus/vs”（“A versus B”即“A对B”、“A与B相比较/对照”），我们也径用原文——改为汉语既罗嗦，也不直观。

3) 模式中的对象名称，往往按照代码风格命名，比如“BusinessObject”、“CustomerTO”等。如果对此完全不加翻译，那么很多充斥这类表达的段落就很难理解。我们的原则是，在每个自然段第一次出现某个这类表达方式时，用括号注明，比如“BusinessObject（业务对象）”、“CustomerTO（客户传输对象）”等。希望这个做法能够维持易懂和简洁之间的平衡。

4) 书中示例代码占有相当大的比重，而代码注释则是理解这些代码的关键。我们把所有代码注释译为中文。而对在视图中显示特定结果的代码（比如调试信息等），我们没有改为中文，只是在必要时对输出信息的含义加以注解。如果读者更信赖代码原貌，还可以从本书官方网站http://www.corej2eepatterns.com/ 下载原始代码。

5) 原书不包含注解，目前的所有注解都是译注。

6) 原书申义未畅处，译文中以方括号[]加以解释、补足，略去生涩。这与上面三条原则一样，都类似于在原作讲话时的插嘴——但翻译任务本身，似乎本就已经是一种“插嘴”了。在博学的读者看来，有时候译者或许还不如保持体面的沉默——但我们只能力图做到插嘴而不多嘴。

7) 原书引用了Apache项目的若干代码，所以附录中包含Apache软件授权协议一页。中译本照录了这份法律文件，未加翻译。

8) 几个关键术语的译名考虑：

• application：一般译为“应用程序”或“应用”。本书中这个词单独出现时，往往指的是

“企业应用”，亦即企业信息应用系统。考虑到“应用程序”容易被理解为“桌面程序(desktop application)”，在该词含有“企业应用”意味时，我们译为“应用系统”，其他情况下则译为“应用”，以示区别。

- client：译为“客户端”。但本书中所说的“客户端”常常是指特定组件的调用者，不一定是“桌面程序客户端”，反倒很可能本身也是另一种组件、甚至一个子系统。希望读者注意该词在书中的用法。
- POJO：软件方法论大师Martin Fowler在《Patterns of Enterprise Application Architecture》(PEAA)中创造的说法，是plain old Java object的缩写，指普通Java对象（而不是EJB等组件)。中译本仍采用“POJO”名称。
- enterprise bean：直译为“企业bean”，在本书中就是“enterprise JavaBean/EJB”的另一说法。为了直观，我们统一译为“EJB”。
- tier/layer：字面上都是“层”/“层次”。本书中“tier”指的往往是“架构”意义上的分层，比如“表现层”、“业务层”、“集成层”等，而“layer”既分享了前者的含义，有时也指tier内部的中间层次，比如“会话门面”就构成了客户端和业务服务之间的一个“layer”。这两种意思实在很难区分，中译本只能都译为“层”、“层次”。希望读者在阅读中体察这种细微差别。
- delegate：是设计模式中的重要概念。一般译为“委派”。但在我们看来，这个译法还不完整，因为“委派”在汉语中只是动词，而delegate往往还充当名词。这次中译本的做法是，动词delegate仍译为“委派”，比如“A把功能F委派给业务层的B”，而名词delegate则译为“代表”，比如“B是A在业务层的代表”。希望读者体察，并推荐更好的译法。

原书中所有模式、重构手法、策略的名称以斜体标出，要点以黑体标出。中译本一仍其旧。

原书经多人、多版修订完成，难免有错漏、乱排之处。译者根据本书官方网站的最新勘误表订正，并结合参照本书第1版《Core J2EE Patterns: Best Practices and Design Strategies》(Addison Wesley，2001)，另外修正了数十处错误。

刘天北　熊节

Grady Booch序

在软件世界中，每个开发机构就像是一个部落，而一个模式就是对部落的某种共同记忆的一种有形表现。模式是对共通问题的共通解决方案，因此，对于某个特定开发机构的文化氛围、某种特定的问题领域来说，命名、确定一个模式，也就是把已经在先前的经验中获得证明的共通解决方案进一步整理成文，设为圭臬。如果你有一套不错的模式语言能随时派上用场，那就像是在开发过程中身旁一直坐着一个专家组：在采用专家们提出的一个模式时，你也就实际获益于他们那些来之不易的经验知识。事实上，那些最好的模式大都不是凭空发明的，而往往是专家们从现有的成功系统中发现、提取而来的。所以，一个成熟的模式中充满了实际有效的内容，不存在空泛不实的内容，同时，它也体现了设计者的智慧和设计思路。

那些深刻的、真正有用的模式大多都是很古老的东西，见到这么一个模式的时候，你往往会说："嘿，我从前就这么做过。"但是，只有当专家为这个模式命名之后，你才获得了一整套讨论这个问题的语汇；此前，由于缺乏这种语汇，你往往想不到怎样使用这个模式，因此命名有助于我们更好地应用模式。最终，这样一个模式的功效在于，它能让你的系统变得更为简单。

而模式还不仅有助于构建更简单而又实际有效的系统，它们还有助于构建优美的系统。在一种极度缺乏时间的文化氛围中，编写优美的软件常常是不可能的，这是很可悲的事情。因为，我们作为专业人士，本来都应该致力于构建高质量的产品。而通过应用一组恰当的模式，你就有可能为自己的系统注入某种程度的优雅——缺乏模式的帮助，这种优雅往往就无从谈起。

《J2EE核心模式》的作者们提取了一组真正实用的模式。别误解我的意思：J2EE当然是一种重要的软件平台，它能够让开发团队构建出特别强大的软件系统。但是，目前的现实却是，在J2EE提供的抽象层次、服务与开发团队必须构建的最终应用之间，还存在着非常巨大的语义断裂。本书描绘的这些模式，事实上正是人们一次又一次用来填补这种断裂的共同解决方案。应用这些模式，也就是采用了一条最主要的避免软件风险的措施：只要编写更少代码就能获得相同的效果。你也不必再重新自己动手寻找解决方案，这些模式已经在很多现存系统的应用中得到了验证，所以只需应用它们就是了。

作者们不仅完成了一组模式的命名，还利用UML确定了模式的语义，使它们更容易为人理解。并且，他们也介绍了应该如何应用这些模式、如何重构你的系统以便从模式中获益。再说一遍，拥有本书就像一个专家组坐在你旁边一样。

Grady Booch

Rational软件公司首席科学家

Martin Fowler序

在1998年末，我所在的ThoughtWorks公司就开始使用J2EE了。在那个时候，我们发现了很多很酷（虽然有点儿不成熟）的技术，但是很少有人能说明怎样才能恰当地应用这些技术。也许是因为我们具备在其他OO服务器环境下编程的大量经验，所以我们自己能够应付这些问题。但是我们也见到很多客户费尽了周折——并不是因为技术本身的问题，而是因为不知道怎样才能恰当地应用这些技术。

使用模式，能够固化设计经验——也就是说，模式有助于将经常重现的问题的实际解决方案进行归类、编目。多年以来，对于模式的这种用途，我一直都是个由衷的爱好者。最近的几年里，业界的很多先驱者都在使用J2EE，都在寻找构成一个有效的J2EE解决方案的核心模式。本书对这些模式进行了出色的汇集，其中揭示的很多技术都是我们自己通过多次尝试、多次错误才获得的。

这也就是本书的重要之处。对API倒背如流是一回事；知道如何设计优秀的软件则是另一回事。本书确实致力于固化这一类设计知识，这也是我见到的第一本这么做的书，看到作者们做得这么漂亮，我由衷感到欣慰。如果你在J2EE平台下工作，你也就需要了解这些模式。

另外，本书也提出了这样一种观点：当编码开始后，设计并非就结束了。人们在设计时做出的一些决定常常不符合实情。在这种情况下，在构建过程中还需要修正原本的设计，而这种修正必须以一种规范的形式进行。越来越多的人选用“重构”的方法来对现有系统做出更动。本书的作者把我在重构方面的研究应用在一个新领域——也就是J2EE设计的世界之中，这样做还尚属首次。我不仅因为有人在我的研究基础上进行工作而心怀感激，而且，读到他们依据自己的经验，实际描述了如何进行系统重构的转换，我也感到非常喜悦。

说到底，这种经验正是最宝贵的东西。把设计经验固化在书本中，这是最难做的一件事，但是要想让我们的行业进一步发展，这件事又非做不可。本书固化了J2EE开发中的重要经验。没有这本书，就别开发EJB。

Martin Fowler

ThoughtWorks 首席科学家

前　言

自从本书第1版出版以来，关于最初那15个模式，我们收到了大量反馈意见。最近几年来，J2EE模式社区目录服务器（JPCLS）上的活动一直都非常活跃、非常成功，每天都有很多精彩的意见交流。在这段时间里，我们也和客户一起进行了不少重要的大型J2EE架构设计、开发项目。把这段时期的经验和反馈植入到原有模式的更新工作和新模式的归档工作中，也确实是一个费力而艰苦的过程。我们特别关注了反馈中提及最多的内容：对J2EE技术规范和Web Service最新版本的支持。

我们完全修订、更新了最初的15个模式，使得本书覆盖了J2EE技术1.4版的规范。我们在这些最初的模式中加入了很多新的策略。另外，我们还记录了6种新模式，以便改进模式语言，为构建、理解、使用J2EE框架提供更好的概念抽象。虽然这些模式中的每一个本身都极为实用，但我们还进一步相信，当开发者将其组合起来解决大型问题时，它们更能显出威力。因此在本书的新版中，引入了一个我们正在探究的、与此相关的全新领域，我们称此为“**微架构**”。

所谓“微架构”，就是搭建应用程序和系统的积木块。与列入目录的那些单独模式相比，这个概念是一种更高层面的抽象，它常常表现为一组相互关联的模式组合，用于解决在应用架构中经常重现的一些共通问题。

我们乐于把“微架构”当作一种由相互关联的模式组成的网络，由此形成一种现成的解决方案，用于解决一个粒度更大的问题，比如子系统的设计。

本版中包括了一个叫Web Worker的微架构。它所解决的问题是：一个J2EE应用怎样与一个工作流系统集成。它特别讨论了使用系统集成模式让工作流系统中的用户与J2EE应用进行交互的问题。

本书讲述的是Java 2企业版平台（J2EE）的模式。本书新版中记录的J2EE模式，能够用于解决在J2EE平台下进行软件应用开发的设计者常常遇到的那些问题。在这个模式目录中记录的模式都是在设计实战中发现的，正是因为使用了它们，我们才能为自己的客户创建出了成功的J2EE应用。

本书描述了很多在J2EE平台下证明可行的解决方案，重点强调了以下核心J2EE技术：JavaServer Pages (JSP)、servlet、Enterprise JavaBeans (EJB) 组件、Java Message Service (JMS，Java消息服务)、JDBC以及 Java Naming and Directory Interface (JNDI，Java命名与目录接口)。对于那些在J2EE平台下经常重现的问题，我们通过J2EE模式目录和J2EE重构给出了解决方案。在开发新系统或是改进现有系统的设计时，你可以应用这些想法。本书记录的这些模式能够有助于你迅速熟练地掌握J2EE技术，从而构建出健壮、高效的企业应用。

今天，正如以往一样，我们中间有很多人天真地以为，**学会了一种技术**，也就等于是**学会了用这种技术进行设计**。诚然，对于利用某一技术进行设计来说，懂得这种技术是成功的重要元素之一。但现在有很多Java图书，对技术细节（比如API的一些专门用法等等）做出了出色的

讲解，但对如何应用这种技术却未作深入考察。要想学会设计，就需要实际设计经验，需要和其他开发者一起分享关于最佳实践和不佳实践的知识。

本书中传达的经验来自我们的工作实战。我们属于Sun公司的Sun Java中心（SJC）咨询机构。在工作当中，我们经常遇到一些情况，因为技术发展过于迅速，设计者和开发者都仍然在奋力理解技术本身，而无暇理解如何使用该项技术进行设计。

因此，简单地告诉设计者和开发者怎样写出优秀代码，或是建议他们使用servlet和JSP开发表现层，用EJB组件开发业务层，这都是不够的。

那么，在这样的情况下，一个热心的J2EE架构师又怎样才能不单单是学到“做什么”、还能学到“不做什么”呢？哪些实践构成了最佳实践？哪些是不佳实践？怎样完成从问题到设计，再到实现的整个过程？

Sun Java中心与J2EE模式目录

从初创时期以来，Sun Java中心的架构师们就在与来自全球的客户一起合作，致力于成功地设计、规划、构建、部署各种不同类型的基于Java和J2EE的系统。Sun Java中心是一个快速成长的咨询机构，一直在招募新员工，加入它经验丰富的架构师队伍。

目前已经有大量已验证有效的设计和构架，将这些设计经验固化下来并和其他人一起分享，是我们行业的一项重要需要。我们很早就认识到了这种需要，从1999年就开始以模式的形式记录我们在J2EE平台下的工作经验。虽然我们翻阅了各种现有文献，却没能发现有哪个模式目录是专门记载J2EE平台下的模式的。有很多书论及J2EE技术中的一种或多种，出色地介绍了技术，剖析了技术规范中的微妙细节。我们发现其中有些书还提供了一些设计上的考虑思路，因此也特别有益。

在2000年6月的JavaOne大会上，我们第一次公开发表了我们关于J2EE模式的想法。从那以来，我们收到了来自架构师和开发者的大量热忱反馈。其中一些人表示特别乐意进一步学习模式，还有一些人则说，他们使用过这些模式，只不过没有加以命名、也没有记录下来罢了。人们体现出来的对J2EE模式的兴趣鼓励我们进行进一步的工作。

因此，我们整理出了J2EE模式目录，在2001年3月，这个目录的beta版通过Java开发者联盟（JDC）首次公布给了J2EE社区。基于整个社区的大量反馈，那一份beta版的文稿最终发展成了你现在见到的这本书。

我们希望这些在J2EE平台下的模式、最佳实践、策略、不佳实践和重构能让大家从中受益。

本书的讨论范围

本书讨论的内容包括：

- 在J2EE平台下使用模式。

  基于我们在J2EE平台的经验，我们编纂了本书中的模式目录。这一份J2EE模式目录描述了在J2EE平台下架构和设计应用的最佳实践。本书着重考察了以下J2EE技术：servlet、JSP、EJB组件和JMS。

- 通过最佳实践来设计应用了servlet、JSP、EJB组件和JMS技术的应用系统。

仅仅学会了技术本身和API还不足够，同样重要的是要学会怎样使用技术进行设计。我们记录了在我们的经验中应用这些技术的最佳实践。

- 防止在J2EE平台的设计和架构中“重新发明轮子”。

模式鼓励设计的重用。重用现成的解决方案，能够缩短设计开发应用程序的周期——这也当然包括J2EE应用。

- 鉴别出现存系统中的不佳实践，并利用J2EE模式重构这些设计，以形成更好的解决方案。

知道哪些做法有效，这是一件好事。但知道哪些做法无效也同样重要。我们在本书中记录了自己在设计J2EE应用时遇到的一些不佳实践。

本书不讨论的内容

本书没有讨论以下内容:

- 如何使用Java或J2EE技术编程

本书讨论的不是编程。虽然很多内容都基于J2EE技术，但我们没有描述API细节。如果你希望学习Java编程，或是学习使用J2EE中的任何一种技术，现有很多种出色的著作，还有不少在线资源，都可以作为教程。如果你想要学习某一门特定的技术，我们强烈推荐Java官方主页http://java.sun.com上的各种在线教程。J2EE技术的官方技术规范也可通过Java主页获得。

- 采用哪种开发过程和方法论

我们并不特别推荐任何一种开发过程或方法论，因为本书讨论的内容与这两方面都关系不大。所以，本书不会教授任何可以用于开发项目的过程或方法论。如果你想要学习过程和方法论的话，现已有很多论著讨论各种面向对象的方法论，对于那些轻量级的过程，比如极限编程，也有不少新书论及。

- 怎样使用统一建模语言（UML）

本书不会教你如何使用UML。我们大量地使用了UML（特别是类图和序列图）来记录模式，描述静态和动态交互关系。如果要学习UML，请参考Grady Booch、Ivar Jacobson 和James Rumbaugh的著作《UML用户指南》[Booch]以及《UML参考手册》[Rumbaugh]。

谁应该读这本书

本书写给所有热心关注J2EE的人，程序员，架构师，开发者以及技术经理。简单地说，就是任何对在J2EE平台下设计、架构、开发应用程序有点儿兴趣的人。

我们力图让这本书成为一部写给J2EE架构师和设计者的培训指南。我们认为良好的设计、架构得当的项目具有很高的重要性，所以我们需要优秀的架构师达到这个水准。

对于那些开发者水准参差不齐的开发团队，如果我们把模式、最佳实践和不佳实践都做出详尽的归档，以此在团队中实现知识与经验的共享和传播，这可能会起到难以估价的帮助作用；我们也希望本书能部分地满足类似需求。

本书的组织

本书的组织分为两部分。

第一部分

第一部分“模式和J2EE”是一个关于J2EE和模式的导论。它考察了开发JSP、servlet和EJB时的设计考虑。这一部分也包括了J2EE平台下的不佳实践和重构。

第1章“导论”简要地讨论了多个问题，包括模式、J2EE平台、模式的定义以及模式的归类。最后引入了J2EE模式目录。

第2章“表现层设计考虑和不佳实践”、第3章“业务层设计考虑和不佳实践”分别讨论了表现层以及业务/集成层的设计考虑和不佳实践。这里所说的设计考虑，是指在J2EE平台下工作时，一个J2EE开发者/设计者/架构师需要考虑的问题。在阅读这两章中的论题时，可以参照其他的多种资源（比如官方技术规范以及一些出色的相关论著）来获得相关问题的一些细节信息。

第4章“J2EE重构”考察了一些重构，我们在自己的实际工作中遇到了这些重构，它们也确实帮助我们把原本不够理想的设计提升为更好的方案。这些重构也提供了看待本书其他内容的另一种思路，我们认为这对于模式目录是一种有价值的补充材料。本章体现出Martin Fowler和他的著作《重构》[Fowler]对我们的影响。对于熟悉《重构》一书的读者，本章的形式也应该相当眼熟。但是，这一章的内容完全基于J2EE技术，而Martin Fowler在他的论著中则是在另一个层面考察重构的。

第二部分

第二部分“J2EE模式目录”列出了J2EE模式目录。目录中包含的模式构成了本书的核心内容。

第5章“J2EE模式概览”，是J2EE模式目录的一个综述。这一章一开始对模式的理念进行了高层次的讨论，并且解释了我们按照系统的分层对模式进行归类的原因。该章也介绍了我们用来记录本书所有模式的“J2EE模式模板”。该章考察了所有的J2EE模式，并且用一张图描述了模式之间的相互关系。另外该章还包括了一种我们称为“模式目录路线图”的东西。这张路线图列举了一些与J2EE设计和架构相关的常见问题，并且把这些问题与特定的模式或重构关联起来，通过这些模式、重构给出了问题的解决方案。理解模式之间的关系以及这张路线图，对于实际应用这些模式至关重要。

第6章“表现层模式”描述了8种模式，它们处理的是在J2EE平台的Web应用设计中，怎样使用servlet、JSP、JavaBeans和定制标记的问题。在这些模式中描述了多种实现策略，并且也提出了一些常见问题，比如请求处理、应用分隔、生成复合视图等。

第7章“业务层模式”，描述了9种模式，它们处理的是怎样应用EJB在J2EE平台下设计业务组件的问题。该章介绍的模式提供了应用EJB和JMS技术的最佳实践。另外，这些模式的相关部分还涉及了其他技术——比如JNDI、JDBC等——的讨论。

第8章“集成层模式”描述了4种模式，它们处理的是怎样把J2EE应用与资源层和各种外部系统集成起来的问题。这些模式使用了JDBC和JMS技术在业务层和资源层之间实现集成。

“尾声”讨论的是一个高层次的主题：怎样利用多个模式一起解决一个大型问题。该章详尽

地讨论了“Web Worker微架构”这个示例，展示了如何通过多个模式来集成一个J2EE应用和一个工作流系统。

本书的官方网站和联络信息

在本书的官方网站上，我们会提供内容的更新信息以及其他一些资料。网址是:

http://www.corej2eepatterns.com

这个网站也附属于Sun Java蓝图网站:

http://java.sun.com/blueprints/corej2eepatterns

你的评论、建议、反馈都可以通过以下邮箱寄给作者:

j2eepatterns-feedback@sun.com

另外还有J2EE模式社区邮件列表服务，邮箱为j2eepatterns-interest@java.sun.com，可以免费订阅和参与。通过以下网址，你可以订阅兴趣小组的邮件，也可以浏览以往的讨论存档:

http://archives.java.sun.com/archives/j2eepatterns-interest.html

致谢

我们想感谢Sun全球软件服务副总裁Cheryln Chin、Sun杰出工程师和首席服务架构师James Baty，如果没有他们的支持、远见以及他们对我们工作的信赖，本书的工作就不可能完成。

我们愿将最大的感激和谢意致予Rajmohan“Raj”Krishnamurthy。如果没有他的帮助，本书就不会有这么多示例代码，而且我们也从他出色的评论意见中受益匪浅。他对本书新版的规划、开发、评审工作做出了不可或缺的帮助。

本书内容经过多位专家的审读，他们的深刻见解、评论意见、反馈建议，为本书的最终成型做出了重要贡献，通过他们的帮助，各个模式的表述比初稿更加清晰、实用；因此我们也愿对以下专家表示谢意：ThoughtWorks公司首席科学家Martin Fowler；Sun J2EE 蓝图团队的Sean Brydon和Inderjeet Singh；Sun公司的Craig Russel，他是 Java数据对象（JDO）技术规范的负责人/产品架构师；ObjectIdentity公司的JDO专家David Jordan；Sun公司的JSP技术规范负责人Mark Roth；Domain Language的 Eric Evans；BEA系统公司的解决方案架构师Mario Kosmiskas；LogicLibrary负责技术的副总裁Brent Carlson；Macromedia 的Sean Neville；Sun Java中心的Java架构师Sameer Tyagi；Chris Steel；Bill Dudney；Gary Bollinger；以及ThoughtWorks公司的Gregor Hohpe。

像这样一本书，肯定需要来自各方面的难以计数的帮助才能得以完成，所以我们很难面面俱到地感谢每一个人为此做出的贡献。

我们想感谢James Gosling和Michael Van de Vanter 领导的Sun Jackpot团队，他们的工作将本书推进到了全新的舞台上。

我们还想感谢Chuck Geiger领导的eBay.com V3团队、Terry Bone领导的福特金融中心的ATD框架团队，他们在企业中实际应用了J2EE模式来构建下一代的系统架构和平台。

感谢Sun Java中心的同事Murali Kaundinya、Ashok Mollin、Ramesh Nagappan和Heidi Schuster。

我们想感谢JetBrains公司提供的IntelliJ IDEA开发工具，为本书编写示例代码时我们使用了这种工具，相当满意。

我们还想感谢J2EE模式社区邮件列表（j2eepatterns-interest@sun.com）上的很多成员，多年以来他们的讨论和反馈一直很有帮助。

特别要对本书的技术编辑Solveig Haugland说一声“谢谢”。她是我们团队的重要一员。她在技术上的编辑工作大大提高了本书终稿的质量。

我们想感谢Prentice Hall出版社的Greg Doench和Debby Van Dijk给予我们的信任和鼓励。

特别感谢无糖红牛饮料提供的动力，让我们能每天写作16小时。

第1版致谢

我们想感谢Sun全球Java中心的主管Stu Stern和负责.COM咨询的副总裁Mark Bauhaus，如果没有他们的支持、远见以及对我们工作的信赖，本书的工作就不可能完成。

我们想感谢Ann Betser，要不是她的支持、鼓励和循循善诱的建议，我们的工作也不会成功。

我们想对Sun Java中心（SJC）PSA/iWorkflow参考实现开发团队的架构师们表达诚挚的感谢，他们是：Fred Bloom、Narayan Chintalapati、Anders Eliasson、Kartik Ganeshan、Murali Kalyanakrishnan、Kamran Khan、Rita El Khoury、Rajmohan Krishnamurty、Ragu Sivaraman、Robert Skoczylas、Minnie Tanglao和Basant Verma。

我们想感谢Sun Java中心J2EE模式工作组的成员们： Mohammed Akif、Thorbiörn Fritzon、Beniot Garbinato、Paul Jatkowski、Karim Mazouni、Nick Wilde和Andrew X. Yang。

我们想感谢Sun Java中心的首席方法专家Brendan McCarthy，他令我们的工作诸事协调，并提出了大量建议。

我们想感谢把这些模式介绍给客户的Jennifer Helms和John Kapson。

我们想对以下来自世界各地的Sun Java中心架构师表达谢意，他们的支持、反馈、建议都令我们受益匪浅，他们是：Mark Cade、Mark Cao、Torbjörn Dahlén、Peter Gratzer、Bernard Van Haecke、Patricia de las Heras、Scott Herndon、Grant Holland、Girish Ippadi、Murali Kaundinya、Denys Kim、Stephen Kirkham、Todd Lasseigne、Sunil Mathew、Fred Muhlenberg、Vivek Pande、John Prentice、Alexis Roos、Gero Vermaas、Miguel Vidal。

我们想对支持、鼓励我们的管理者Hank Harris、Dan Hushon、Jeff Johnson、Nimish Radia、Chris Steel和Alex Wong表达谢意。

我们还想感谢在Sun公司中与我们合作的以下同事：

Sun软件系统组的Bruce Delagi； Sun软件工程部门的Mark Hapner、Vlada Matena; Forte产品组的Paul Butterworth和Jim Dibble； iPlanet产品组的Deepak Balakrishna； J2EE蓝图团队的Larry Freeman、Cori Kaylor、Rick Saletta和Inderjeet Singh； Heidi Dailey； Java开发者联盟的Dana Nourie、Laureen Hudson、Edward Ort、Margaret Ong和Jenny Pratt。

我们想感谢以下各位对本书的反馈、建议和支持：

ThoughtWorks公司的Martin Fowler和Josh Mackenzie； Richard Monson Haefel; Goldman

Sachs公司的 Phil Nosonowitz 和Carl Reed; Rational软件公司的Jack Greenfield、Wojtek Kozaczynski和Jon Lawrence; TogetherSoft的Alexander Aptus; Zaplets.com 的Kent Mitchell ; Bill Dudney; David Geary; Hans Bergsten; J2EE模式兴趣小组 (j2eepatterns-interest@ java.sun.com) 的成员。

我们想对本书的首席技术编辑Beth Stearns表示特别的谢意和感激，她负责整理我们的手稿，让全书明了可读，与此同时还要随时掌控我们的工作进度，同我们一道完成一个高强度的工作计划。

我们想感谢技术编辑Daniel S. Barclay、Steven J. Halter、Spencer Roberts和Chris Taylor，他们出众的专业能力、细致的评审反馈对本书的完成非常重要。

我们想感谢Prentice Hall出版社的Greg Doench、Lisa Iarkowski、Mary Sudul和Debby Van Dijk; Sun公司出版社的Michael Alread和Rachel Borden，他们使本书的诞生成为可能。

目　录