HZ BOOKS

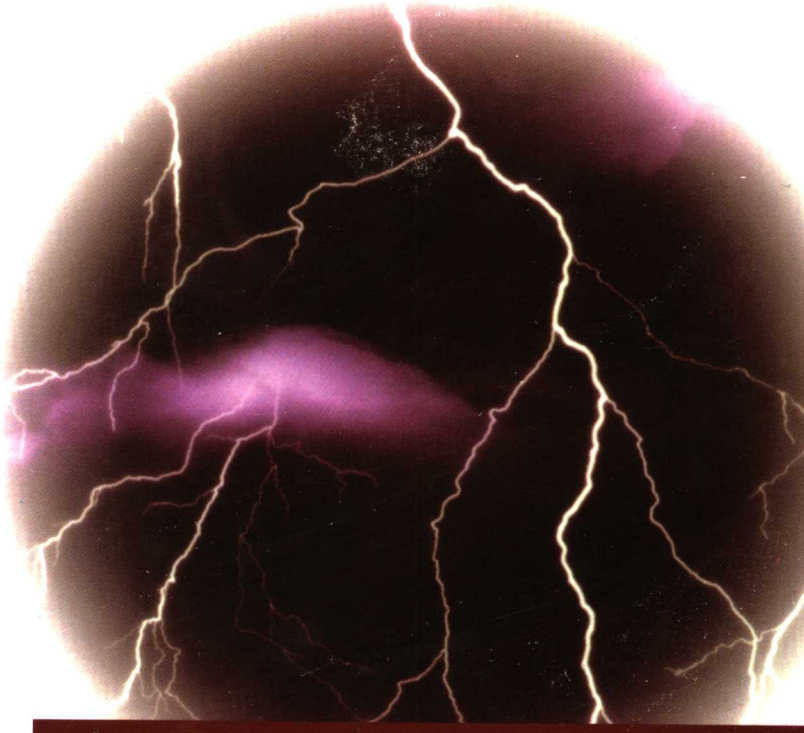
敏捷数据

Agile Database Techniques

Effective Strategies for the Agile Software Developer

(加) Scott W. Ambler 著

李巍 译
UMLChina



机械工业出版社
China Machine Press

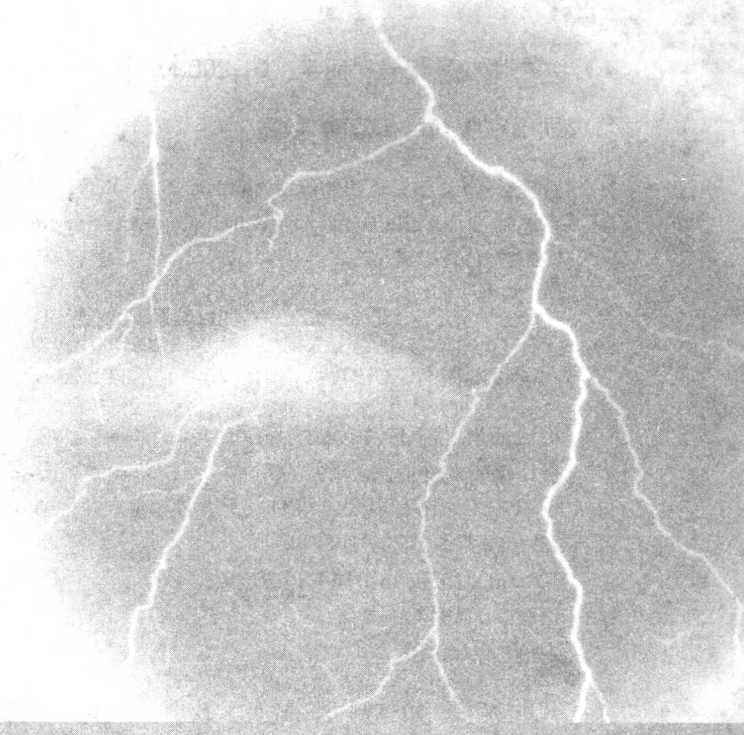
敏捷数据

Agile Database Techniques

Effective Strategies for the Agile Software Developer

(加) Scott W. Ambler 著

李巍
UMLChina 译



机械工业出版社
China Machine Press

本书作者在数据及对象技术方面都有很深造诣，多年的经验使他深刻地认识到：数据专业人员常常过于专注数据而忽视对象开发人员所面临的困难；而对象开发人员又没有或有很少的数据方面的经验。本书作者探索了有机结合数据和对象两个开发团队的方式，将敏捷方法拓展到了应用程序开发的一个关键领域——数据库，阐述了数据架构设计师、数据库管理员掌握敏捷方法进行面向数据开发的必要性。

本书分四部分。第一部分描述数据专业人员和对象专业人员所需的基本技能和方法，第二部分介绍进行渐进式数据库开发的方法，第三部分概述有效地结合使用对象技术、关系数据库技术和XML技术的方法，第四部分总结如何成功地采用本书所描述的技术方法。本书适合应用程序开发人员及数据处理人员阅读。

Scott W. Ambler: Agile Database Techniques: Effective Strategies for the Agile Software Developer（ISBN:0-471-20283-5）

Authorized translation from the English language edition published by John Wiley & Sons, Inc.

本书版权登记号：图字：01-2003-5

图书在版编目（CIP）数据

敏捷数据/（加）艾姆布勒（Ambler，S. W.）著；李巍译.－北京：机械工业出版社，2006.1
书名原文：Agile Database Techniques: Effective Strategies for the Agile Software Developer
ISBN7-111-17576-X

Ⅰ.敏…　Ⅱ.①艾…　②李…　Ⅲ.数据库系统　Ⅳ.TP311.13

中国版本图书馆CIP数据核字（2005）第119023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：刘亚姝　　刘立卿
北京中兴印刷有限公司印刷·新华书店北京发行所发行
2006年1月第1版第1次印刷
787mm×1020mm 1/16·19.25印张
印数：0 001-4000册
定价：39.00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

译者序

——谁说数据不能跳舞

敏捷联盟、敏捷宣言、敏捷建模、敏捷文档、敏捷Web开发……

敏捷，当今软件开发的又一个buzzword。作者这一次又将敏捷思想的触角伸及到数据领域。作为数据建模和对象建模专家以及敏捷方法的积极布道者，Ambler无疑是该领域最合适的人选。可是，这又是一本关于数据方面的书籍吗？

答案既“是”又“不是”。之所以说“是”，是因为本书涵盖了数据建模的方方面面，从最基本的数据规范化、关系－对象映射，到高级的数据库重构、并发控制与性能优化，一应俱全；而且本书的阅读门槛并不高，它沿袭了著者一贯的论述风格，即便对那些相对高级的主题也是娓娓道来、点到为止。如果你还嫌不够，作者还为你推荐了各个相关主题的经典著作并给出中肯的评价，足够你就某一主题掘地三尺。

之所以说“不是”，是因为这不是一本单单写给数据专业人员的书。作者强烈建议应用程序开发人员也应该读一读本书，因他深知对象和数据技术由于底层范型的不同而导致这两大阵营在观念上存在差异，甚至彼此对立、相互倾轧。因而本书开篇就着重探讨这两种技术之间的阻抗失配，同时恳请应用程序开发人员和数据专业人员能够真正地抛开成见、通力协作。

诚然，那些关于数据的章节或许是本书的主要论述内容，但著者更想与你分享如何敏捷地去做事情、去思考，以及如何建立有效的相互协作的敏捷团队，就像具体的技能或框架或许可以供你一时享用，而思维方式和行事之道则更有助于解决问题和拥抱变化。从这个意义上来说，敏捷更是一种做事态度：你必须首先选择这样去做，并且愿意尝试变化，而不是墨守成规，把自己框定在熟悉的领地不放。

同那些顽固的数据专业人员一样，对象阵营中也有一部分人沉溺于“纯对象主义”的光环固步自封。作者在书中反复强调，这个世界色彩绚丽，你不只有黑和白两种选择，不要从一个极端走向另一个极端。作为敏捷思想的提倡者，作者建议那些想实践敏捷思想的开发人员致力于成为一名通用型的专家，而非传统意义上的专家。尝试以别人的思路去考量问题，开放思维，取长补短。关系数据库中的数据规范化，过程化设计中模块的耦合与内聚性，以及OO（面向对象）设计中的单一职责原则、依赖反转原则又何尝不昭示着思想上的共通性呢？

权衡的重要性在本书各章的论述中反复加以印证。无论是应用程序开发还是数据开发，都应该根据自己的切身情况做出权衡，找出适合自己情况的最佳点。想想数据建模中采用非规范化的手法改善性能，程序设计中以增加间接层获得灵活性的思想，也无处不体现了权衡的艺术。

建议读者仔细看看本书最后关于“有效成为一名敏捷数据开发人员”的建议，这同样适用

于应用程序开发人员。我想这两章会给读者带来一些震撼，试着以别人的方式看待问题，像敏捷宣言里提倡的那样拥抱变化、轻装前进。谁说数据开发不能敏捷呢?

最后，感谢UMLChina的潘加宇先生在翻译过程中所给予的鼓励和帮助。感谢我的妻子何晓梅为本译稿所做的大量的文字工作，你对译文的推敲质疑使我如履薄冰，而你在旁边的陪伴则使我安心和专注。

李巍，UMLChina
2005年9月于北京

Jon Kern序

2001年2月，敏捷运动在犹他州11 000英尺的雪鸟峰的阴影之下初见端倪。自那时起，大众和博学者似乎都在谈论和实践敏捷性。许多开发团队厌倦了重量级过程失败的允诺，以及在未知的时间表内朝着未知的目标前进的死亡之旅，他们正在寻找一套适合“人们解读”的开发哲学和原则。

Scott Ambler在敏捷社区内享有很强的发言权，并且在2001年2月创立了敏捷建模论坛（Agile Modeling Forum）。对于所有“认识”Scott的人来说，毫无疑问都知道他会积极热情地帮助开发团队成功地为他们的涉众（以及最终为他们的客户）服务。

有许多研究对象建模、敏捷方法、UML（统一建模语言）、特定语言、数据库设计、SQL等方面的资源。许多好书都阐述了关于如何开发一个好的面向对象应用程序的信息。同样，也有许多关于开发和优化数据库的优秀文献。你却很难碰到描述一种渐进的、敏捷的、面向数据的开发方法的书籍。

在本书中，Scott探寻了应用程序开发的一个关键领域——数据库，他扩展了敏捷方法的边界，使其能够跨越应用团队——从开发人员到数据架构设计师，阐述了敏捷方法不再只限定于开发人员的领域之内。对于数据库，DBA（数据库管理员）也能应用同样的原则——增量和迭代式地开发，就像开发人员使用他们的代码库那样。现在，DBA能够理解敏捷方法学，并将其应用于面向数据的开发。他们能够深入了解和学习如何融入更大的团队，如何更好地运用他们关于特定DBMS（数据库管理系统）的丰富经验，以及如何有效地支持团队的持久需要。即使你工作的小团队并未“配备”DBA，这本书仍十分有用，因为它深入探讨了解决所有开发团队所面临的通用的持久性问题的关键方法。

对于我们中那些愿意向开发团队引入面向对象方法学的人来说，数据建模一直是个让人感兴趣的主题。一方面，好的对象模型看上去非常像是一个构成良好的实体－关系图（如果不是这样，你需要寻求对象建模方面的指导！），而且许多对象建模人员会从他们的类图上推导出数据库的设计；另一方面，一些数据库专家坚持认为这个世界围着数据库转（尤其是最近10年来那些由遗留数据库驱动业务的地方）。这两种情况都有优点——却都不全对。Scott提出了一种能在团队内达成施受平衡的、渐进式的方法学。他指出，在整个开发迭代的过程中，现实世界的应用程序通常都对数据提出了不止一种要求。

大多数现代开发方法学本身就是迭代式的，而且要求一种渐进的方式。大多数铁杆的数据建模人员可能更熟悉具有详细事先设计的瀑布方法。有时候这会导致团队内部出现摩擦，并造成各个阵营之间的火并。本书向开发人员讲授基本的数据库知识，向DBA教授成为敏捷开发项目成员所必需的技能。有效地结合（intertwining）敏捷对象开发和敏捷数据库开发能够帮助团

队探索他们的成功之路。

我真希望自己八年前就能有这样一本书，那时我在开发自己的第一个主要是瘦客户端的、面向对象的应用程序（其具有数据管理层和所有其他随之而来的类似功能）。使用本书，你可以避免遭受许多“沉重的打击”。如果你曾经在一个分布式环境中考虑过“脏标志”（dirty flag）、两段式提交，或者在“谁”应该负责引用完整性（数据库或对象）的问题上存有争议，那么本书可让你受益匪浅。而且，如果你无法确定这些术语的含义，那么你真的必须要考虑这本书。

由于几乎每个（业务）应用程序开发项目都会面临数据存储的需求，因此对于大多数开发团队来说本书都是极具价值的资源。确保你们的开发人员和数据库人员人手一本。通过学习敏捷数据库方法，开发人员可以提高他们的技能，而DBA则可以学到如何沿着更为敏捷的路线开发他们的数据库——更为有效地支持开发工作。总之，每个人都能取得成功，从而获得某种激励，用功研读，然后开始相互协作。

Jon Kern

《Agile Manifesto》的合著者

Douglas K. Barry序

我希望每个阅读这篇序的人都能立刻翻到本书的第23章，看看关于如何变得更加敏捷的建议表。是不是这个表中的条目使你感到有些不舒服？很好。你是否认为这些建议没有什么必要？为什么会这样？听我的建议：你的确需要照Scott建议的去做。为了提高你们项目的成功几率，这些是你能够采取的第一步。而且它们做起来可能非常不舒服。

某些精神上的不适，对那些想要有所改变的人来说是一件好事。毫无疑问，我们构建（或构建失败的）软件系统的方式需要改变。其中包括数据库。如果我们都在做自己能够找到的、舒服的事情，那么变化的可能性将很小。

对我来说，Scott在第23章中所建议的花一段时间从别人的角度考虑问题是一件好事情。确实需要坚持。能够与其他人进行交谈，并设身处地地关注和理解他们的处境，这一点很好，但还不够。实际去尝试其他人的工作是一段非常不同的经历。

我知道这一点是因为我经历过许多。在我职业生涯初期，我是一家大公司的数据建模的领头人。于是，我参与软件设计，并不得不处理其他人的数据库设计。之后，我成为一家刚成立的数据库公司的CIO。随后从事了许多年数据库相关标准的开发工作。同时，我开始帮助人们理解什么才是满足他们需要的最佳架构——这正是我当前的工作。让我来告诉你，这已经成为一门教育，而且时常会让我感到不舒服，但现在我的状况在逐渐好转。从我的经历来说，Scott似乎为你展示了一条可以沿循的好路线。

在整本书中，Scott囊括了使用敏捷方法进行数据库开发的实用建议。你可能不一定总是认同，但这很可能会挑战你的思维方式。这同样是一件好事情。

Scott还提出了一些常识性的设计建议，来开发数据库以及不同类型系统之间的数据映射。这些建议非常重要，而且在那些基础的建模文章中你并不是总能找到它们。

正是由于这些使你变得敏捷的不一般的建议和常识性的设计建议，注定了这是一本极为全面的好书，尤其是对那些想跨越基础的建模文章的人。你在这里能够找到实用的、有现实意义的建议。

Douglas K. Barry
Barry & Associates公司的创始人和总裁
（www.barryandassociates.com）

致　谢

我要感谢以下人士，感谢他们的想法和怀疑精神（这一点尤为重要）：Dave Astels、Philippe Back、Francois Beauregard、Graham Berrisford、Charles Betz、Chris Britton、Steven Brown、Steve Cohen、Mike Colbert、Warren Cotton、Dale Emery、Neal Fishman、Adam Geras、Steven Gordon、Jason Gorman、Michael M. Gorman、David C. Hay、Michael Haynes、Mats Helander、Daniel Honig、Ron Jeffries、Jon Kern、Jan Emil Larsen、Kevin Light、Floyd Marinescu、Les Munday、John Nalbone、Pan Wei Ng、Paul Oldfield、Oscar Pearce、Chris Roffler、Dave Rooney、John Roth、Adrian Rutter、Yonn M. Samuels、Dwight W. Seeley、Paul Tiseo、Jason P. Tryon、Patrick Vanhuyse、Micheal Vizdos、Michael Vorburger、Sebastian Ware、David Waters、Gary Williams、Dawn M. Wolthuis和Michael Zimmer。

我还要感谢John Wiley & Sons 出版公司的编辑，他们是：James H. Russell、Angela Smith、Terri Hudson、Kathryn A. Malm和Robert M. Elliott。

前　言

敏捷的结论：这是一本真正的好书。购买它、阅读它、推广它。

从20世纪90年代初以来，我就一直在工作中使用对象和关系数据库（RDB）技术，来构建业务应用程序，并且从90年代中期开始，我在这个主题上已经撰写颇多文章。这些文章被发表在《Software Development》（www. sdmagazine.com）上、我的几本书籍（特别是《Building Object Applications That Work》和《The Object Primer》）中以及我的个人主页上。我站点上的两本白皮书非常受欢迎，一本是关于对象－RDB映射的，另一本则描述持久层的设计，这些年来它有成千上万的下载量。有关持久层的文章已被用作多个开源产品的基本原理。尽管对我来说，通过这些著作分享我的想法是非常值得的，但是我从未花时间把这些作品整理到一处，而且我也没有写完关于该主题我所有想说的内容。本书将扭转这种局面。

作为一名顾问，我会与对象和数据专业人员一起工作，并使用他们相关的技术，当然还有他们的方法。在这样做的过程中，无论是采用近似连续的方式开发的传统环境，还是使用一种敏捷和渐进式开发方式的更现代的环境，我都在其中工作过。随着时间推移，我在许多不同的项目团队上充当过各种角色。对于每个项目的成功而言，面向数据的问题很重要，有时候甚至是至关重要的。尽管传统的项目团队看起来能够处理如何解决数据的问题，但是更加敏捷的团队常常对此产生质疑——一部分原因是这些组织机构内的数据专业人员倾向于使用一种连续性的方式，一部分原因是对象开发人员没有意识到面向数据的问题的重要性。作为一名前数据专家（哦不，泄露了我可怕的秘密！）并拥有在对象技术方面的经验，我常常可以找到这两个团体相互合作的方式。我的经验是数据专业人员常常过于专注数据，而将对象开发人员所面对的各种挑战排除在外；同样对象开发人员有很少的或没有数据相关的经验。因此，我会帮助这两个团体寻找相互合作的方式，针对其他每种方法对他们进行指导，并且帮助他们克服被称做对象－关系阻抗失配的问题。这两个团体要想有效地一起工作，他们都需要理解和重视其他团体所专注的东西，而且我甚至质疑各个团体最初持有的判断力（wisdom）。本书描述了数据专业人员和对象专业人员在构建现代软件时所需的技能。

作为一名方法学者，我积极致力于寻找有效开发软件的方法，而且这些年来，已涵盖了从正规方法（例如，有关过程模式（process pattern，www.ambysoft.com/processPatternsPage.html）和企业统一过程（Enterprise Unified Process，EUP，www.enterpriseunifiedprocess.info）的工作）到敏捷方法（如敏捷建模AM（www.agilemodeling.com）和现在的敏捷数据库方法）的方方面面。从某个方面来说，本书是AM的延续，有助于描述数据专业人员如何采用一种渐进（迭代和增量）的开发方式。尽管数据阵营内的许多人坚决反对渐进的方式，有趣的是我经常发现那些反对者从来都没有实际去尝试它；事实是敏捷软件开发是真实的，并且能够留存下来。数据专业人员要想与时俱进，他们必须做好以一种敏捷方式工作的准备，否则项目团队很有可能找到

绕开他们工作的方式（我猜想你总能在自己的组织机构中看到这种事情的发生）。在实际项目中我的经验是，对面向数据的活动采取一种敏捷的方式，你确实能够获得极大的成功，如果你选择这样去做的话。许多人将告诉你它没有用，但是他们真正所说的是他们可能无法让它运转，或者他们不想让它有用。本书描述了多种验证过的支持渐进式面向数据开发的方法。

在刚开始编写本书时，我想专注于敏捷数据（AD）方法（www.agiledata.org）。该方法（总结于第1章中）描述了数据专业人员和应用程序开发人员如何能够一起有效地在敏捷项目上工作。它还描述了企业专业人员（如企业架构设计师和数据管理员）如何能够有效地支持敏捷开发团队。由于我采用了一种迭代和增量的方式来书写本书，我便迅速意识到真正的价值在于具体的开发方法，而非另一种方法学。因此我重新调整自己的关注点。

本书的读者

谁是本书的读者？简单的答案是任何敏捷软件开发团队成员，或至少与这个团队打交道的人。更为复杂的答案是：

敏捷/极限编程人员。如果你所构建的软件是对数据进行操作，那么机会难得：因此，你需要采纳本书中所描述的许多方法。

数据库管理员。本书描述了你如何在一个敏捷软件开发团队上取得成功。请从头至尾细细研读。

数据管理员。随着时间的推移，你需要支持越来越多的敏捷开发团队，因此你需要理解他们是如何工作的，以及他们为什么这样工作。本书提供了你帮助这些团队提高效率所需的见解。

架构设计师。敏捷、渐进式开发正快速成为大多数组织机构内的规范。本书描述了你可以在这些团队中有效工作而采取的方法。

团队领导者/指导员/经理。为了有效地领导敏捷软件开发团队，你必须了解你们团队所使用的方法，他们为什么要使用这些方法，以及这么做意味着什么。本书不但描述了这些方法，而且还论述了它们之间该如何权衡，使你能够帮助团队做出明智的决定。

为何把注意力放在敏捷DBA上面

尽管我在本书中描述的大多数技能都同时适用于应用程序开发人员和数据库管理员（DBA），但是我选择从敏捷DBA的角度来表达它们。一个敏捷DBA会专注于面向数据的问题，包括传统的数据库管理和任何涉及数据的应用程序的开发。敏捷DBA还会与企业专业人员进行协作，以确保项目团队的工作能够反映企业的现实情况。更重要的是，他们需要以一种敏捷的方式做这项工作。敏捷DBA的角色可以由你们项目中的多个人来担任，由多个人共同轮流分担，或者由一个人单独担任。尽管敏捷DBA的技能集（skillset）看上去比较难以掌握，而且确实如此，但是你可以通过与其他具备你所缺少的技能的人一起工作，通过培训，或者通过自己实际地尝试，来不断地获取这些技能。

概述

本书由四个部分组成。第一部分为有效地进行面向数据的活动奠定了基础，描述了所有IT

专业人员所需的基本技能和原则。第二部分描述了进行渐进式数据库开发的方法，说明了采取一种迭代和增量式的面向数据的开发方式是可能的。第三部分概述了有效地结合使用对象技术、关系数据库技术和XML（Extensible Markup Language，可扩展标记语言）技术的详细实现方法。第四部分概括论述了如何成功地采用本书所描述的概念。

第一部分

IT业界一个显著的问题是，大多数数据书籍没有涉及到面向对象开发的问题，而大多数对象书籍看起来也忽略了数据问题。需要对这种局面喊停。第一部分描述了敏捷项目团队中的每个人应该具备的基本技能和知识。包括面向对象、关系数据库、对象－关系阻抗失配、数据建模，以及如何处理遗留数据问题的基础内容。如果没有这种通用的知识基础，应用程序开发人员和数据专业人员将很难有效地一起工作。

第二部分

第二部分专注于如何针对数据采取一种渐进的方式。该部分为模型驱动开发（model-driven development，MDD）——或者更为准确地说是敏捷模型驱动开发（agile model-driven development，AMDD）——的方法打下了基础，在该部分中的应用程序代码和数据库方案都是以敏捷模型为基础的。这不是唯一的一种工作方式；你还可以决定采取测试驱动开发（test-driven development，TDD）的方法，或者更好的是能够将其与AMDD结合起来。这两种方法都支持渐进式开发，但由于MDD在数据阵营内非常普遍，我想开发人员更倾向于AMDD方式，而不是TDD方式。然而，有些敏捷开发人员，特别是极限程序员，更倾向于TDD（与AMDD相比）。幸运的是，这两种方式一起使用的效果非常好，因此无论你选择哪一个都不是问题。这意味着在以后的日子里，TDD对于数据专业人员会变得愈加重要。该部分还描述了数据库重构，一种使你能够在很少的几步内改善自己数据库设计的渐进式方法。在许多方面，数据库重构是事后的规范化。该部分描述了对象－关系数据库映射、性能优化、数据库封装和支持性工具，这是因为它们使渐进式开发成为可能。

第三部分

第三部分专注于实现方法和策略，例如并发控制、安全访问控制、在关系数据库中查找对象、引用完整性和有效使用XML。一个重要的现象是，许多主题从传统意义上都被认为是数据问题，但正如你看到的那样，事情远非如此——这不是一个黑与白的世界。

第四部分

第四部分描述了采用敏捷数据库方法的策略。该部分对那些想要成为敏捷软件开发人员的个人和想要采取敏捷方法的组织机构分别给出了建议。

目　录

译者序

Jon Kern序

Douglas K. Barry序

致谢

前言

第一部分　基础背景

第1章　敏捷数据方法 ……2

1.1　缘何当前难以相互合作 ……2

1.2　发现问题 ……4

1.3　敏捷运动 ……5

1.3.1　敏捷软件开发宣言 ……5

1.3.2　敏捷软件开发原则 ……7

1.4　敏捷数据的基本原理 ……7

1.5　敏捷数据概述 ……8

1.5.1　敏捷DBA ……8

1.5.2　应用程序开发者 ……9

1.5.3　企业管理员 ……10

1.5.4　企业架构设计师 ……11

1.6　敏捷软件开发者 ……12

1.7　敏捷数据能解决我们的问题吗 ……13

1.8　总结 ……14

第2章　从用例到数据库——现实世界的UML ……15

2.1　面向对象概念简述 ……15

2.2　UML介绍 ……18

2.2.1　核心UML图 ……18

2.2.2　辅助UML图 ……21

2.3　数据建模的UML档案 ……26

2.3.1　标识模型或存储机制的类型 ……27

2.3.2　建模表、实体和视图 ……28

2.3.3　建模关系 ……31

2.3.4　建模数据属性和列 ……31

2.3.5　建模键 ……32

2.3.6　建模约束和触发器 ……34

2.3.7　建模存储过程 ……34

2.3.8　建模数据库内的段 ……34

2.3.9　建模其他概念 ……34

2.4　总结 ……35

第3章　数据建模基础 ……36

3.1　敏捷DBA的角色 ……36

3.2　什么是数据建模 ……36

3.2.1　如何实际运用数据模型 ……37

3.2.2　基本标记：如何阅读数据模型 ……39

3.3　如何建模数据 ……41

3.3.1　识别数据实体 ……41

3.3.2　识别属性 ……42

3.3.3　应用数据命名规范 ……42

3.3.4　识别关系 ……42

3.3.5　应用数据建模模式 ……44

3.3.6　分配键 ……44

3.4　如何更好地进行数据建模 ……47

3.5　总结 ……48

第4章　数据规范化 ……49

4.1　为什么需要数据规范化 ……49

4.2　敏捷DBA的角色 ……49

4.3　数据规范化的准则 ……50

4.3.1　第一范式（1NF） ……50

4.3.2　第二范式（2NF） ……52

4.3.3　第三范式（3NF） ……53

4.3.4　超越3NF ……54

4.4　总结 ……54

第5章 类规范化 ……55
5.1 如何将类规范化和其他面向对象的设计实践联系起来 ……55
5.2 敏捷DBA的角色 ……56
5.3 类规范化的准则 ……56
5.3.1 第一对象范式（1ONF） ……56
5.3.2 第二对象范式（2ONF） ……57
5.3.3 第三对象范式（3ONF） ……58
5.3.4 超越3ONF ……59
5.4 总结 ……59
第6章 关系数据库技术，无论你喜欢与否 ……60
6.1 关系数据库技术 ……60
6.1.1 关系数据库的简明特性 ……60
6.1.2 关系数据库的高级特性 ……61
6.2 耦合：你最大的敌人 ……63
6.3 关系数据库的额外挑战 ……65
6.4 封装：你最好的盟友 ……66
6.5 超越关系数据库：根据实际情况做出选择 ……68
6.6 总结 ……71
第7章 对象－关系的阻抗失配 ……73
7.1 敏捷DBA的角色 ……73
7.2 技术上的阻抗失配 ……73
7.2.1 虚假的相似性 ……74
7.2.2 微妙的差异性 ……75
7.3 文化上的阻抗失配 ……78
7.4 总结 ……80
第8章 遗留数据库——所有你需要了解却害怕应付的事物 ……81
8.1 敏捷DBA的角色 ……81
8.2 遗留数据源 ……83
8.3 理解与遗留数据相关的常见问题 ……84
8.3.1 数据质量挑战 ……84
8.3.2 数据库设计问题 ……87
8.3.3 数据架构问题 ……88
8.3.4 过程错误 ……90
8.4 使用遗留数据的策略 ……91
8.4.1 尽量避免使用遗留数据 ……92
8.4.2 开发一种数据错误处理策略 ……92
8.4.3 迭代和增量式地工作 ……92
8.4.4 优先考虑只读的遗留数据访问 ……93
8.4.5 封装遗留数据访问 ……94
8.4.6 为简单的遗留访问引入数据适配器 ……94
8.4.7 为复杂的数据访问引入中间数据库 ……95
8.4.8 采用现有的工具 ……96
8.5 数据集成技术 ……96
8.6 总结 ……97

第二部分

第9章 革命万岁 ……100
9.1 方法灵活性的需要 ……100
9.2 谨防面向数据的BDUF ……101
9.3 针对项目的渐进式开发 ……103
9.4 事物的“自然顺序”和渐进式开发 ……106
9.5 总结 ……107
第10章 敏捷模型驱动开发 ……108
10.1 敏捷DBA的角色 ……108
10.2 什么是敏捷建模 ……108
10.2.1 AM价值观 ……109
10.2.2 AM原则 ……109
10.2.3 敏捷建模实践 ……111
10.3 什么时候模型才算敏捷 ……112
10.4 什么是敏捷模型驱动开发 ……113
10.5 敏捷文档 ……113
10.6 总结 ……115
第11章 测试驱动开发 ……116
11.1 TDD是如何工作的 ……116
11.2 TDD的步骤 ……117
11.3 TDD与传统测试 ……118
11.4 TDD与文档 ……119
11.5 测试驱动的数据库开发 ……119

11.6 TDD与AMDD……120
11.7 总结……121
第12章 数据库重构……122
12.1 重构……122
12.2 数据库重构……123
12.2.1 保持语义……124
12.2.2 什么不是数据库重构……124
12.2.3 数据库重构类别……125
12.3 为什么数据库难以重构……125
12.4 如何重构数据库……127
12.4.1 步骤1：在开发沙箱中开始……127
12.4.2 步骤2：在集成沙箱中实现代码……133
12.4.3 步骤3：实地安装代码……134
12.5 常见的数据库重构味道……134
12.6 在你的组织机构内采用数据库重构…135
12.7 数据库重构最佳实践……136
12.8 现实中的数据库重构……137
12.9 总结……137
第13章 数据库封装策略……139
13.1 数据库封装层……139
13.2 敏捷DBA的角色……140
13.3 封装层的架构……141
13.4 封装层的实现策略……142
13.4.1 蛮力方式（并非一种封装策略）…142
13.4.2 数据访问对象……144
13.4.3 持久化框架……145
13.4.4 服务……148
13.4.5 每种策略的适用场合……150
13.4.6 策略间的转化……151
13.5 整编和数据验证……152
13.6 错误处理……152
13.7 总结……153
第14章 对象－关系数据库映射……154
14.1 敏捷DBA的角色……154
14.2 基本的映射概念……155
14.2.1 影子信息……157
14.2.2 映射元数据……159
14.3 映射继承结构……159
14.3.1 将整个类层次体系映射成一个表……161
14.3.2 将每个具体类映射成它自己的表……162
14.3.3 将每个类映射成它自己的表……162
14.3.4 将类映射成一个通用结构……163
14.3.5 映射策略的对比……164
14.3.6 多继承的映射……165
14.4 映射对象的关系……166
14.4.1 关系的类型……167
14.4.2 如何实现对象间的关系……167
14.4.3 如何在关系数据库内实现关系…169
14.4.4 关系映射……170
14.4.5 映射有序集合……173
14.4.6 映射递归关系……175
14.5 映射类作用范围的特性……175
14.6 为何数据方案不应该驱动对象方案……177
14.7 对对象的实现影响……179
14.8 模型驱动架构的启示……180
14.9 映射模式化……180
14.10 总结……181
第15章 性能优化……183
15.1 性能优化概述……183
15.2 敏捷DBA的角色……184
15.3 步骤1：识别性能问题……184
15.4 步骤2：剖析问题……184
15.5 步骤3：优化解决问题……185
15.5.1 系统优化……185
15.5.2 数据库访问优化……186
15.5.3 数据库优化……188
15.5.4 应用程序优化……193
15.6 总结……195

第16章 渐进式数据库开发的工具 ……196
16.1 工具 ……196
16.2 沙箱 ……198
16.3 脚本 ……199
16.4 总结 ……200

第三部分 实用的面向数据的开发方法

第17章 实现并发控制 ……202
17.1 敏捷DBA的角色 ……202
17.2 冲突 ……202
17.2.1 锁定类型 ……203
17.2.2 解决冲突 ……205
17.3 理解事务 ……206
17.3.1 事务的基本概念 ……207
17.3.2 实现事务 ……208
17.4 总结 ……212
第18章 在关系数据库内查找对象 ……213
18.1 敏捷DBA的角色 ……213
18.2 查找策略 ……213
18.2.1 蛮力方式（嵌入式SQL） ……213
18.2.2 查询对象 ……214
18.2.3 元数据驱动 ……215
18.2.4 每种策略的应用时机 ……216
18.3 实现方法 ……217
18.3.1 使用原生的错误处理策略 ……217
18.3.2 能够预见“逻辑”错误 ……217
18.3.3 总是返回一个集合 ……218
18.3.4 针对查询列表采用代理和惰性初始化 ……218
18.3.5 对高开销的属性使用惰性读取 ……218
18.3.6 为他人编程 ……219
18.4 表示查找的结果 ……219
18.5 总结 ……221
第19章 实现引用完整性和共享的业务逻辑 ……222
19.1 敏捷DBA的角色 ……222
19.2 对象技术是如何将引用完整性复杂化的 ……222
19.2.1 多种实体/关系的表现形式 ……224
19.2.2 对象关系管理 ……224
19.2.3 惰式读取 ……226
19.2.4 缓存 ……227
19.2.5 聚合、组合和关联 ……227
19.2.6 架构分层 ……228
19.2.7 从内存中移除与永久删除 ……228
19.3 应该在何处实现引用完整性 ……229
19.3.1 引用完整性的实现选择 ……229
19.3.2 业务逻辑的实现选择 ……231
19.3.3 通用实现策略 ……232
19.4 总结 ……232
第20章 实现安全访问控制 ……233
20.1 敏捷DBA的角色 ……233
20.2 身份验证 ……233
20.3 授权 ……234
20.3.1 问题 ……234
20.3.2 数据库实现策略 ……235
20.3.3 安全设计模式 ……237
20.3.4 面向对象的实现策略 ……238
20.3.5 启示 ……240
20.4 有效的安全策略 ……240
20.5 总结 ……241
第21章 实现报表 ……243
21.1 敏捷DBA的角色 ……243
21.2 数据库部署架构 ……244
21.3 在应用程序内部进行报表统计 ……246
21.4 在应用程序外部进行报表统计 ……248
21.5 数据库设计策略 ……248
21.6 实现策略 ……249
21.7 使报表统计变难的挑战 ……250
21.8 总结 ……250
第22章 现实中的XML ……252
22.1 敏捷DBA的角色 ……252

22.2 XML入门 ……252
22.2.1 XML的优势 ……254
22.2.2 XML的弱点 ……255
22.3 XML的实际应用 ……256
22.4 词汇表 ……256
22.5 如何建模XML ……258
22.6 XML映射和数据绑定 ……260
22.7 如何在关系数据库中持久化XML …262
22.8 如何在XML数据库中持久化XML …262
22.9 XML开发策略 ……264
22.10 总结 ……264

第四部分 采用敏捷数据库方法

第23章 你如何才能变得敏捷 ……268
23.1 不必非要做一个超人 ……268
23.2 敏捷性其实只是一个思维集 ……268
23.3 成为一名博学型的专家 ……269
23.4 总结 ……270
第24章 将敏捷性带到你的组织机构中 …271
24.1 改变你看待软件开发的方式 ……271
24.2 理解你所面对的挑战 ……272
24.3 实际去尝试它 ……273
24.4 阻止非敏捷的合作者 ……273
24.5 切合实际 ……274
24.6 临别感想 ……275
附录 数据库重构目录 ……276
参考资料及推荐读物 ……286