TURING 图灵程序设计丛书 程序员修炼系列

WILEY

AntiPatterns
Refactoring Software, Architectures, and Projects in Crisis

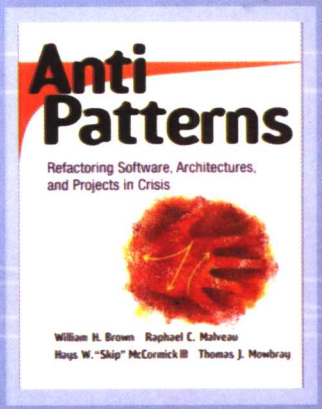
反模式

危机中软件、架构和项目的重构

第9届Jolt生产效率大奖

[美] William J. Brown
Raphael C. Malveau
Hays W. McCormick III
Thomas J. Mowbray 著

宋锐 等译



- 软件工程圣经之一
- 涵盖软件项目中开发、架构和管理各个方面
- 从独特的视角审视软件开发

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书 程序员修炼系列

AntiPatterns
Refactoring Software, Architectures, and Projects in Crisis

反模式
危机中软件、架构和项目的重构

[美] William J. Brown
Raphael C. Malveau
Hays W. McCormick III
Thomas J. Mowbray 著

宋锐 等译

人民邮电出版社
北 京

图书在版编目（CIP）数据

反模式：危机中软件、架构和项目的重构 /（美）布朗（Brown, W.J.）等著；宋锐等译. —北京：人民邮电出版社，2008.1
（图灵程序设计丛书）
ISBN 978-7-115-16279-3

Ⅰ. 反… Ⅱ. ①布…②宋… Ⅲ. 软件开发 Ⅳ. TP311.52

中国版本图书馆 CIP 数据核字（2007）第 074692 号

内 容 提 要

模式是可以复用的优秀解决方案。本书从一个新的角度审视模式，提出了反模式的概念，介绍了在软件开发中常常出现的问题——将设计模式错误应用于不适当的上下文环境。首先，定义了软件开发参考模型和文档模板来说明这些反模式。然后，从开发人员角度、架构角度和管理角度三个方面对这些反模式逐一说明，并说明了与特定反模式相关的背景、原因、症状和后果，让读者可以迅速地检验身边的项目是否出现了这些状况，同时也针对每个反模式给出了相应的解决方案。

本书适用于从事项目管理和软件开发的相关人员。

图灵程序设计丛书

反模式：危机中软件、架构和项目的重构

◆ 著 [美] William J. Brown Raphael C. Malveau Hays W. McCormick III Thomas J. Mowbray
译 宋 锐 等
责任编辑 陈兴璐

◆ 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
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：800×1000 1/16
印张：14.75
字数：363 千字 2008 年 1 月第 1 版
印数：1 – 5 000 册 2008 年 1 月北京第 1 次印刷
著作权合同登记号 图字：01-2007-2465 号

ISBN 978-7-115-16279-3/TP

定价：45.00 元

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

反盗版热线：(010) 67171154

对本书的赞誉

“本书的作者们在软件开发的管理领域显然身经百战。我自己曾经见证了太多陷入困境的开发项目，因此书中一个又一个真知灼见都能引起我的强烈共鸣。书中字里行间都充分体现出作者们的丰富经验。”

——John Vlissides，软件开发大师，《设计模式》作者之一

“本书可以称得上我们行业的圣经，将使你能更全面地理解面向对象的程序设计。我从本书中受益匪浅。”

——Thomas E. Davis，Internet计划的首席技术官

“本书深入探讨了软件项目中的问题及其解决方案。它没有局限于技术模式，而是涵盖了软件开发的方方面面，包括管理和组织问题。它为我们行业创立了一个标准词汇表，大大方便了开发人员、架构师和经理之间的交流。”

——Angelika Langer，著名技术专家，*Agile Java*一书作者

“只要你从事软件开发工作，无论是程序员、架构师，还是技术经理，你和你的同事就都应该熟悉本书中的内容——不仅要了解问题所在，还要掌握可能的解决之道。”

——Francis Glassborow，ACCU（C和C++用户协会）前主席，
曾任ISO C和C++标准委员会主席

“软件管理人员、架构师和开发人员都可以从本书中吸取前人的惨痛教训。书中关于软件架构性的反模式对软件工程的贡献极大。”

——Kyle Brown，IBM资深模式和Java技术专家，
Enterprise Java Programming for IBM WebSphere一书作者

“本书秉承了《设计模式》一书开创的传统。作者们揭示并命名了反模式——糟糕的管理或者架构所导致的许多常见问题，也是大多数有经验的软件从业人员能够识别的错误。对于各种反模式，作者们还提供了解决之道。”

——Gerard Meszaros，ClearStream咨询公司首席科学家，Mock Object模式发明者

“我是一口气读完本书草稿的，其间不断被书中的深刻见解打动，而作者的幽默表述又常常使我会心一笑。作者们叙述的许多反模式都可以在我自己的经历中得到验证。强烈推荐。”

——Al Stevens，Dr. Dobb’s Journal

“我喜欢本书，读起来非常有意思。我把书借给了一位英雄所见略同的项目经理，他居然不还我了……”

——Jeff Grigg，资深软件顾问

译 者 序

从1994年以来，有关设计模式的文献出版量呈指数形式增长。对有经验的面向对象架构师来说，现在有许多的、而且还在不断增长的可复用设计，可以利用它们来简化软件开发工作。但是，许多使用设计模式的人未能正确评估特定设计模式对他们所面对的特定问题的适用性，试图在完成领域分析之前就把所有的事情都归并到某个设计模式，或者用一组特定的设计模式来解决所有的问题。而反模式所针对的，就是在软件开发过程中各种反复出现的问题，其中的相当一部分就是将设计模式应用于不正确的环境而造成的。

模式可以帮助你识别和实现有益的过程、设计和代码，而反模式的作用与模式正好相反，它们让你留意软件开发过程中潜在的各种陷阱与危险，这些东西都可能会导致项目的毁灭。本书的四位作者长期从事与软件开发、项目管理和培训相关的工作，都具有丰富的行业经验。他们在本书中以现实主义的态度介绍了数十种常见反模式及其相应的解决方案。根据本书的指导，项目管理者可以避免很多常见的问题，提高开发过程的生产率，更及时地提供更能满足需求的系统。

本书主要由宋锐、肖国尊等翻译。如果广大读者需要对本书的内容进行讨论，可以发送电子邮件至coldmoon75@163.com。此外，参与本书翻译的还有马蓉、焦贤龙、邝祝芳、杨明军、张杰良、肖枫涛、刘齐军、闫志强、韩智文。Be Flying工作室负责人肖国尊负责本书翻译质量和进度的控制与管理。敬请广大读者提供反馈意见，读者可以将意见e-mail至be-flying@sohu.com，我们会仔细查阅读者发来的每一封邮件，以求进一步提高今后译著的质量。

译 者

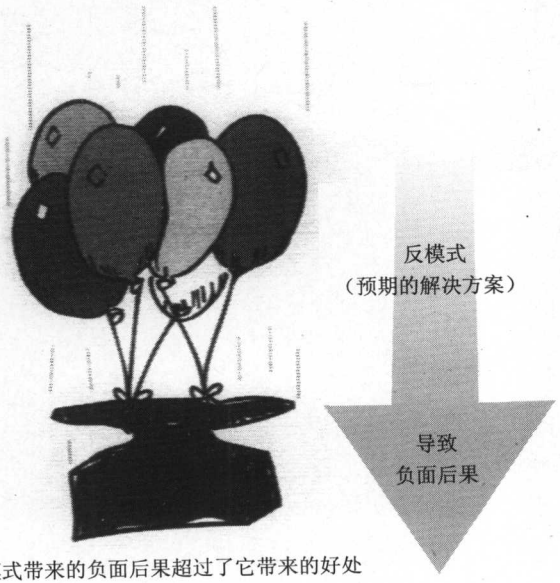
2007年1月于湖南长沙

序

我们非常荣幸地受邀来为这本有关反模式的书作序。第一次听别人提到这个术语时，我们更多的是迷惑。除非你已经了解反模式是什么，否则很可能也会感到不解。但在更深入的研究之后，我们发现这确实是一个有趣的主题，并且具有巨大的实际价值。

我们中的大多数人都相当熟悉（或者至少听说过）设计模式的概念，无论是在软件开发环境中还是在别的环境中。设计模式这一术语本身是无须解释的：设计是过去（经过实践）被证实可以取得成功，因此可以成功地复用的。

但什么是反模式？它是某种不是设计模式的东西吗？或者是以前没有被采用过，缺乏稳健设计的做法？还是那些不起作用的东西？使用设计模式时可以通过复用经过试验和测试的设计来节省金钱、时间和精力，而漫不经心地应用反模式则会导致完全相反的后果。



有时模式带来的负面后果超过了它带来的好处

在英文中，pattern（模式）一词可以是：对重复使用的配件的特定排列；用来指导布料裁减的设计或形状；一个模型或样品。但是pattern也可以是一种行为。本书的作者们见证了生活的各

个方面（尤其是软件开发）中行为的负面模式，可能更引人注意的是，这些负面模式在各种规模和层次的经历中都会出现。

反模式可以告诉你要避免哪些情况，而确定要避免哪些情况对成功的软件开发是非常关键的。本书详细说明了与软件相关的反模式的多种类别，并以幽默轻松的方式指出了应该尽量避免的各种软件设计反模式、架构反模式和管理行为反模式。

Jack Hassall / John Eaton

对象管理组织（OMG）

金融领域特别工作组联合主席

前　言

阅读反模式或者和朋友一起讨论它们是很有意思的事情。但是不要因此而掉以轻心！本书讲述的是软件技术和开发真实的一面。在本书中，我们定义了在技术和软件项目中实实在在发生的事以及你可以采取的措施。反模式指出了那些会导致劣质软件和项目失败的不良设计概念、技术途径和开发实践。本书还解释了项目中应如何发现和避免这些问题，改善设计与实践，使软件获得成功。

你能从容面对真相吗？真相令人吃惊地难以表达，而且往往被人看作是不正确、不通世故的。真相并不会让所有人都感到高兴。为了使对我们这个行业的揭示更易于接受，我们尽可能地采用了一些喜剧的方法，但是常言道乐极生悲。事实是，软件工程的现状就是一场悲剧：5/6的公司开发项目被认为是不成功的[Johnson 1995]。大部分软件系统交付时提供的功能远少于期望，几乎所有的系统都是烟囱系统，无法适应变化的业务需要。

在试图解释软件上普遍缺乏成功的过程中，我们得出的结论是实践中的反模式的数目远远多于设计模式。在网络时代，技术变化如此迅速，以至于昨天的设计模式会迅速变成今天的反模式。本书介绍了在实践中、在产品中以及在软件文献中反复出现的反模式。然后，我们在每个反模式中都包含了一个重构方案来指出解决方法，该重构方案来自我们在现实世界中使用过的、或者看到过是有效的解决方案。

内容概述

本书的第一部分（第1章～第4章）介绍了设计模式和反模式。接着提供了一个反模式参考模型来建立将在反模式说明中使用的共同定义。有经验的读者应该从第2章中的参考模型说明开始。第二部分（第5章～第7章）包含对反模式的说明，是从对开发性反模式的讨论开始的。然后介绍了架构性反模式和管理性反模式。最后，本书的第三部分为更深入地研究和应用反模式提供了资源。

反模式与设计模式的关系

反模式是下一代的设计模式研究。它们针对已有实践、遗留设计和前向工程方案，覆盖了更为广泛的应用。

补充材料

本书的主页位于www.serve.com/hibc/AntiPatterns/index.htm。作者们会在该网站提供更新。

致　谢

虽然我们希望能够感谢所有让本书成为可能的人，但实在无法列出所有为我们提供观点、帮助和鼓励的人，在此特别感谢：

Rohit Agarwal
Don Awalt
Tom Beadle
Pier-Yves Bertholet
Irv Boeskool
John Brant
Frank Buschmann
Bruce Caldwell
Ian Chai
Hugh Chau
Vic DeMarines
David Dikel
John Eaton
Karen Eaton
Ken Kinman
John Kogut
Gary Larson
Steve Latchem
Eric Leach
Dave Lehman
Barry Leng博士
David Lines
Pat Mallett博士
Mark Maybury博士
Dave Mayo
Dennis Egan
Marty Faga
Jack Flannagan
Brian Foote
Alejandra Garrido
Tom Gleeson
Julie Gravallese
Steve Gulick
Patrick Harrison博士
Dan Harter
Jack Hassall
Hebden兄弟
Christy Hermansen
Ruth Hilderberger
John Polger
Don Roberts
Darrel Rochette
Mark Rosenthal
Henry Rothkopf
Bill Ruh
David Samuels
Thad Scheer
Robert Silvetz博士
Bruce Simpson
Theresa Smith
Roy Hiler
Steve Hirsch
Michael Hoagland
Bill Hoffman
Jon Hopkins
Barry Horowitz博士
Bill Ide
Tom Jenkins
Ralph Johnson
Pat Jones
Michael Josephs
David Kane
David Kekumano
Ajay Khater
Kurt Tran
Kevin Tyson
Doug Vandermade
James Van Guilder
Robert Wainwright
Kim Warren
美国华盛顿特区软件图书学习俱乐部
John Weiler
Diane Weiss

Kate Mowbray
Lewis Muir
Diane Mularz
Eiji Nabika
Jason Novak
Jeanne O'Kelley
Ed Peters
Richard Soley博士
Ed Stewart
Shel Sutton
Fred Thompson
Bhavani博士
Thuraisingham
Pat Townes
Anthony Whitson
Jerry Wile
Deborah Wittreich
Joe Yoder
Ron Zahavi
Tony Zawilski

本书概要

本书将帮助你识别和克服那些在软件开发过程中普遍存在、反复出现并会影响到软件成功开发的障碍。反模式清晰地定义了大部分人在开发软件时经常会犯的错误。实际上，大部分人犯这些错误的一致性甚至可以达到ISO 9001对一致性的要求！反模式还提供了对这些错误的解决方案：如何解决已经出现的问题以及如何避免在将来重复这些灾难。简而言之，反模式描述和解决了现实世界中的问题。下面这些问题是本书内容的一些例子：

（1）最常见的两个软件设计错误是什么？如何才能识别它们？参阅第5章中的The Blob反模式和Poltergeists反模式。

（2）如何修复（或重构）不良的软件？参阅第5章中的Spaghetti Code反模式和第6章中的Stovepipe System反模式。

（3）设计项目正在原地打转，如何才能让它回到正轨？参阅第7章中的Analysis Paralysis反模式和第6章中的Design by Committee反模式。

（4）我如何才能知道被软件供应商误导了？参阅第6章中的Vendor Lock-in反模式和第7章中的Smoke and Mirrors反模式。

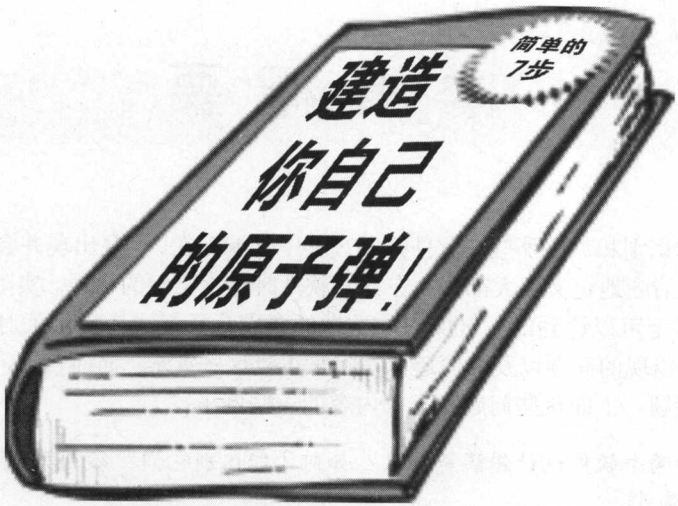
（5）最新的标准或技术突破可以解决问题吗？参阅第6章中的Wolf Ticket反模式和第5章中的Continuous Obsolescence反模式。

（6）软件项目是否正走向灾难？参阅第7章中的Death by Planning反模式和第5章中的Mushroom Management反模式。

（7）软件复用中的常见陷阱是什么？参阅第5章中的Cut-and-Paste Programming反模式和Golden Hammer反模式。

反模式阐明了那些导致开发障碍的负面模式，并包含了经过验证可以把软件开发问题转变成机会的解决方案。反模式可以服务于两个重要目的：帮助识别问题和帮助实现对问题的解决方案。理解问题是进行恢复的第一步。如果要没有确定要解决的问题，解决方案就没有什么用处。反模式有多种原因，并具有相关的症状和后果。我们对每个反模式的这些方面都进行了说明，以澄清为什么要进行改变。然后我们为每个反模式提供了经过验证的、被频繁使用的解决方案。

反模式和另一个重要的软件概念——设计模式密切相关，后者记录了被频繁使用的解决方案。在导致的问题比它解决的问题更多时，设计模式就会成为一个反模式。



图E-1　这是个恶梦

所有的模式都会产生一定的后果。有些情况下，一个模式是可以解决某个问题的好方案，而在另一些情况下，它会成为一个反模式。要想做出可能带有副作用的合理决策，理解模式的这些依赖于上下文环境的后果很重要。我们将从下面这些视角对每个模式进行检查，并说明了反模式在什么时候会提供益处：

- 管理性的（管理过程和人员）。
- 架构性的（定义技术策略）。
- 开发性的（编码实践）。

管理性的、架构性的和开发性的反模式在第5章～第7章中依次进行了定义。如果你刚接触设计模式或者反模式，参阅第1章～第3章提供的介绍性材料。对设计模式实践者，第2章说明了反模式参考模型；第3章说明了使用的模板。这些介绍性的章节有助于让反模式发挥最大的作用。

阅读本书的原因

反模式对软件的成功是至关重要的，主要有以下关键原因：

- 反模式随处可见。失败的软件项目远多于成功的软件项目。不良的软件设计、决策和项目远比良好的更为普遍。现实世界中的软件充满了反模式，当然也存在极为高效的解决方案。随着深入学习本书，你将会更清楚地看到这一点。
- 反模式澄清了最常见的软件设计错误。不良的设计和软件是一贯的原因、常见的误解和经典的错误的结果。反模式解释了为什么会出现不良的软件、如何重构不良设计和不良

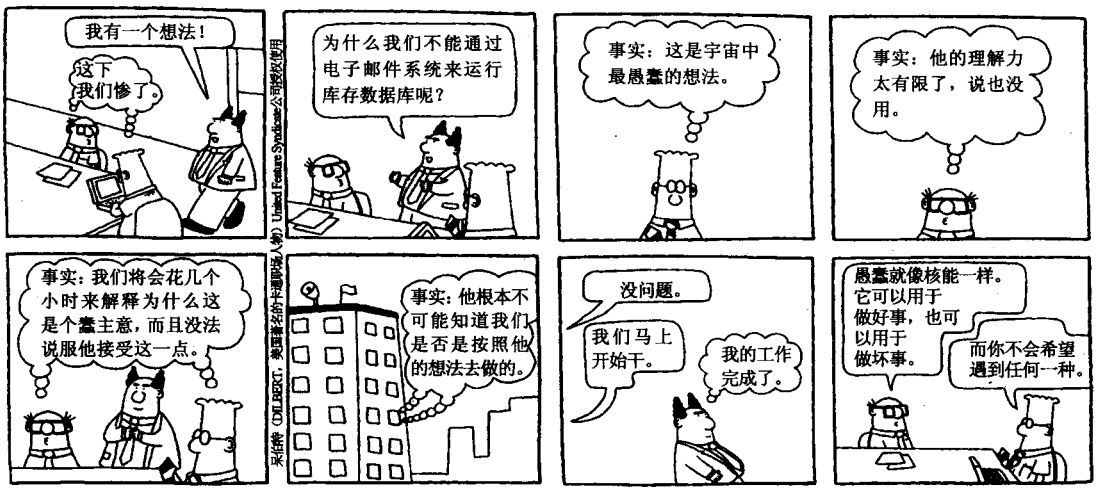
软件，以及如何避免重复这些错误。通过反模式，你可以学会如何及时发现和修复错误以避免严重的后果。

- **反模式揭示了有关软件行业的真相。**商业软件技术中充斥着缺陷、矛盾、虚假承诺和反模式。本书从业内人士的角度揭露了有关软件技术的真相。反模式记录了有关商业技术的关键信息，而这些信息对于你适应新兴的商用现货（COTS）驱动的开发范型是必需的。
- **反模式解释了软件项目的现实。**大部分软件项目是混乱、不可预计的，而且对职业生涯是危险的。反模式解释了软件项目实际上是如何运作的，以及如何管理它们来避免不良后果。
- **反模式对变化管理是必需的。**反模式给负面的软件实践建立了清晰的定义。它们是有用的说明，指出了你所在的机构会希望改变的实践。本书提供了对反模式的全面分类，这可以推动变化管理。
- **反模式定义了重要的术语。**每个反模式都为常见的软件实践定义了广为使用的术语。有很多反模式术语用于说明为什么事情会出错。与设计模式相似，反模式通过使用关键短语来指代复杂的概念，所以没有必要重新发明这些概念。使用这些术语的机构和个人可以提高工作效率。
- **反模式是更有效的设计模式形式。**我们是设计模式运动的信徒。*CORBA Design Patterns* 一书解决了以模式作为一种文字形式时的一些缺点[Malveau 1997]。本书更进了一步：对模式范例进行了重新定义和扩展，得到了一个全新的形式，也就是反模式。

反模式形式零星地出现在因特网上，它们主要是设计模式社区的作者们编写的非正式反模式。根据网上论坛上的观点，模式社区一致赞同反模式是值得研究的领域。本书对反模式概念进行了全面的发展，所得出的结论是十分有效的问题解决形式。

有一些关键原因让我们相信反模式是有效的。普通的设计模式开始于对上下文环境和作用力的长篇讨论，在逻辑上自然导向特定的解决方案。虽然与为了引出设计模式解决方案所需的长篇文字讨论相比，方案本身似乎更显而易见，但是许多读者发现这种风格冗长乏味。与之相反，反模式一开始是会导致严重后果的。通过强调可能产生的灾难，反模式描述了更吸引人的现实情况而不是抽象的作用力。在此之后提出了一个建设性的解决方案。通过详细说明症状和后果，反模式为要采纳的变化提供了具有说服力的论据，而普通设计模式在这一点上是无法与之相提并论的。而且，反模式可以吸引你的兴趣和想象，而任何设计模式都没有这种魔力。

请带着开放的心态来阅读本书，你会发现学习反模式并与同事讨论它们是很有趣的事。反模式是根据实际软件开发中的成败得到的。我们希望你在阅读本书时能够得到与我们编写和分享反模式时获得的同样乐趣。



图E-2　行政决策造成反模式

目　录