HZ BOOKS 华章科技

PEARSON

软件工程技术丛书

再工程领域的集大成之作，由有20余年实践经验的资深软件架构师撰写，亚马逊全五星好评

通过大量真实案例，充分展示利用再工程方法持续优化现有软件系统的工具、思维、方法和最佳实践

软件再工程

优化现有软件系统的方法与最佳实践

Reengineering .NET *Injecting Quality, Testability, and Architecture into Existing Systems*

（美）Bradley Irby 著　张帆 翟林丰 译

机械工业出版社
China Machine Press

软件工程技术丛书

软件再工程

优化现有软件系统的方法与最佳实践

Reengineering .NET *Injecting Quality, Testability, and Architecture into Existing Systems*

（美）Bradley Irby 著　张帆 翟林丰 译

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

软件再工程：优化现有软件系统的方法与最佳实践/（美）厄比（Irby, B.）著；张帆，翟林丰译 .—北京：机械工业出版社，2013.12
（软件工程技术丛书）
书名原文：Reengineering .NET: Injecting Quality, Testability, and Architecture into Existing Systems

ISBN 978-7-111-44881-5

Ⅰ.软… Ⅱ.①厄… ②张… ③翟… Ⅲ.软件工程 Ⅳ.TP311.5

中国版本图书馆 CIP 数据核字（2013）第 280701 号

再工程领域的集大成之作，由有 20 余年实践经验的杰出 CTO、资深软件架构师撰写，亚马逊全五星好评，指引你学会如何以最小的花费，在最短的时间通过再工程方法使旧版系统焕发新生。本书结合真实案例和示例代码，充分展示探究旧有代码真实状态、制定再工程计划、引入最新的工具和方法以提升性能的思维、方法和最佳实践，从而将新架构以及开发进展集成到不可离线的关键业务系统中。

全书共分两部分。第一部分包括第 1 ～ 5 章。第 1 章以面向服务的架构为基础，讲解如何明确最终目标的架构。第 2 章分析 3 种常见架构模式的区别、工作机制，以及如何选择适用的架构和设计模式。第 3 章以一个较高的层次重新认识单元测试的理念和实现。第 4 章讲解如何利用依赖倒置原则打破应用程序的紧耦合，以便进行优化。第 5 章介绍对单元测试使用测试替身的要点。第二部分包括第 6 ～ 13 章。第 6 章介绍如何从整体上认识再工程项目的原解决方案，辨析当前代码的真实状态。第 7 章分析再工程团队在项目规划和管理方面所面临的挑战，并给出解决方案。第 8 章介绍高效的再工程工具、方法。第 9 章介绍如何清除旧版解决方案。第 10 章讲解具体添加核心服务、类、库的方法，以建立再工程基础。第 11 章讨论如何将新的架构整合到旧版系统。第 12 章介绍架构重构中的一些重难点任务。第 13 章讲解将窗体重构为控制器的方法和具体步骤。

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：肖晓慧
三河市杨庄长鸣印刷装订厂印刷
2014 年 3 月第 1 版第 1 次印刷
186mm × 240mm • 16.75 印张
标准书号：ISBN 978-7-111-44881-5
定 价：69.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991 88361066 投稿热线：（010）88379604
购书热线：（010）68326294 88379649 68995259 读者信箱：hzjsj@hzbook.com

华章计算机
HZBOOKS | Computer Science and Technology

译 者 序

科技日新月异，用户的需求也在不断提高和变化，而现有旧版系统常常已在其中投入了非常多的时间和成本，问题不断但弃之可惜。如何在不影响企业和用户的前提下，花最小的代价、最少的时间提升或改造旧版系统，使其重新稳定、高质量地满足用户需求呢？再工程无疑为困惑者指明了方向。和原书作者一样，我也是极力地推崇软件工程技术。与学校的书本教条不同，本书中的再工程，更多的是来自于行业的经验总结，带有实用主义的色彩。这也是我最终决定翻译本书的原因之一。另外，这本书适应了当下的需求，能为很多软件的改造提供指引和帮助，起到抛砖引玉的作用。如此，我也做了一件益事吧。

过去的几年里，断断续续也翻译过一些科技文章，技术书籍等，但是那些工作，要么是比较零星，要么是半途夭折。能够像这本书一样，从头到尾，自始至终坚持翻译下来，自觉不易。

浸淫国外文化多年，有的时候，读起来比较顺口的英文，也要花心思琢磨。另外，有些技术词汇，国内的和国外的叫法区别很大。为了更接近读者习惯，常翻字典和查阅科技文献。

翻译这本书的过程，也同时伴随了我自己第一个孩子的诞生。所以想把这本译作，当作是送给我亲爱的妻子祝骅和儿子文骏的一份礼物。人生贵在坚持。能够坚持，把一个事情做完整、做好，这就是一种幸福。希望我的家人能够体会到这种幸福。

最后，还要感谢协助我参与本次翻译工作的翟林丰同学，以及中君科技的同事程欢、王丹婷和卢许骏等人。没有他们的配合，本书无法如期面世。

谢谢帮助过和支持过我的人，也感谢自己的坚持和努力，更要感谢购买本书的读者。

张 帆

前　言

什么是软件再工程

任何一个有多年工作经验的软件开发者都曾经面临过这样一个困难的局面：如何改进应用程序。这种困难之处包括：浏览代码很困难，确定从哪里开始追踪缺陷很困难，对缺陷进行修复也很困难。总之，关于应用程序的所有一切都很难。改进和修复缺陷耗时长、风险高而且代价大。

解决旧版应用程序问题的一种可能的选择是：离线一年或更久的时间，从头开始重写。通常情况下，这些应用程序对业务运营是非常关键的，所以，功能开发不能停滞如此长的一段时间。因此，对旧版应用程序的修复工作需要不间断地进行，以期在下一个发布周期之前修复所有缺陷并推出补丁。

此外，还有一种选择来帮助解决这些旧版系统的问题，那就是“软件再工程”。

什么是旧软件

一个应用程序构建完成后，就会立即步入老化的过程。软件工程是一门年轻的学科，构建应用程序的方法每天都在推陈出新。随着业界不断引入新的开发工具，如果不更新现有的应用程序从而使用这些工具，那么它们会变得越来越难以维护。

软件老化的原因

软件老化有很多原因。最明显的原因莫过于当今世界科学技术的飞速发展。几年前才开发的软件技术很快就会变得陈旧，并且难以维护。

快速的工作变动正在成为一种潮流，这同样加剧了代码质量的下降。随着原先的开发人员转投新的公司，关于代码结构设计的最初的一些理念就会丢失，导致留下的开发人员拼尽全力收拾残局。

然而，通过使用软件再工程持续不断地改进系统，这种对原代码设计者的依赖程度就会削弱。新的开发人员也能很容易适应和了解系统的架构，这主要是因为系统的架构将是与时俱进的，同时在网络上也可以找到很多有用的信息。

警示标志

特定标志可以警示一个系统是否真的到了需要再工程的时候。

开发者对功能请求存在抵触情绪

如果开发人员抵制来自于管理人员或用户的功能请求，不愿意改进应用程序，那么很有可能是因为该系统真的已经到了举步维艰的地步。随着时间的推移，该软件应用会变得非常脆弱，从而导致任何功能的开发都变得极其困难且令人沮丧。

发布后存在着大量缺陷修复工作

如果开发团队在软件新版本发布后马上就面临缺陷警告，则表明软件在开发过程中并没有使用现代的质量保证工具。为了显著减少缺陷率，部分软件再工程流程需要引入这些自动化的质量保证工具。

软件质量问题长期存在

旧软件中缺陷及其修复工作的多少显示了旧软件的老化程度。软件的老化和脆弱程度越高，做非破坏性的修复就越难。如果你发现，每当刚修复一个缺陷，马上就有两个或者更多的缺陷涌现出来的时候，那么说明这个应用程序需要进行再工程。

由于不支持诸如单元测试和系统模拟等新的质量保证方法，旧版应用程序特别容易出现质量问题。如果不使用这些工具，那么在系统修复时所产生的那些缺陷，看上去会与刚刚所做的修复完全无关。

软件再工程的目标和优点

软件再工程的目标是通过引入现代的架构和软件开发技术来逐步改善现有系统，同时在保证在线的情况下，持续不断地为系统增加新的功能。这意味着对目前已有的系统来说，我们无须费尽心思重写代码。通过软件再工程，我们可以有条不紊地改进现有系统，使之逐渐达到符合最新的软件开发标准的水平。在整个软件再工程项目期间，系统随时都可以进行产品发布。换句话说，这就好比我们可以在飞机飞行的同时对飞机进行维修。

引入现代架构

软件系统的架构决定了需要建立的必要细节的数量。有的时候，我们试图使用最新的方法为一个旧系统建立一种数据输入的形式，这就像是为一架双翼飞机加上一个喷气式发动机一样，这样做飞机就算飞离地面也飞行不了多久。

大多数管理者听到“需要更新架构”的第一反应就是应用程序必须重写。这并不一定正确。事实上，只要保证新的组件在接入的过程中，始终保持一个合理的步骤和顺序，那么架构的更新就可以逐步实现。

引入新架构无须劳师动众。不管开发团队有多少人，或者现有系统有多少行代码，新架构的引入完全可以由一个小团队甚至一个独立的设计师来完成。引入新架构的每一步都是独立的，并且不影响应用程序的其余部分。引入新架构和质量管理无须大量的预算或者专家团队。

在线增加新功能

本书中关于如何在线增加新功能的所有步骤都是特别设计的，这样既保证了这些步骤不会对其他开发造成干扰，同时保证了这些步骤之间可以并发进行。这些新功能的结构也是特别设计的，为的是保证引入的新功能不会带来任何不利影响，并在完成时通过几行代码就可启用新功能。这确保：在前端，产品经理能够持续不断地接受产品新的功能请求，发布更新的版本；在后台，每个新的版本都会在前一版本的基础上有所改进。

对于任何一个新增的功能，如果它需要一个独立程序开发员花费一整天的时间来完成，那么为该应用增加新功能的过程将会被重新分解成多个小的步骤。 这样做的目的在于：一方面，保证该应用的部分功能能够持续更新；另一方面，保证该应用的其他部分功能仍然能够使用旧的结构。

随着该应用每一个部分的逐步更新，应用程序将会变得更可测，缺陷率也会大幅度地降低。

灵活使用敏捷方法

很多开发公司都采用敏捷开发策略。对于不熟悉这种方法的公司来说，敏捷开发可以简单地解释成快速周期开发。这种周期（又称为“冲刺”）通常以几天或者几周来度量。每个快速周期开发结束时，应用程序处于一个潜在可发布状态。

保持应用程序处于一个潜在可发布状态，就是软件再工程。 每一次所做的更新都必须是完整和独立的，这样才能够保证在提交代码的时候，一方面系统仍然可以正常运行；另一方面系统能够处在一个稳步提升质量的过程中。这就是本书在设计这些步骤时所遵循的设计方法。每个步骤都可以在一个冲刺阶段中完成，而这往往是一天。极少数软件再工程的工作需要比较长的时间。这样的工作就需要特殊设计，以保证当它们还没完成，或者完成一半的时候，整个系统受到的影响也

是微乎其微的：系统可以照常运行，只不过部分功能已经进行过更新，部分功能还处于旧版本的状态下。

降低风险

一个软件系统使用多年之后，用户就会形成对该软件系统固定的业务流程操作模式。重写整个系统会破坏这些未文档化的业务流程，逼迫用户在工作流中使用新系统。

软件再工程维持了工作中的这些未文档化的流程。通过在现有系统上慢慢引入新的架构，这些流程就可以不受干扰。如果引入新架构确实以某种形式扰乱了正常的业务流程，那么一定会有一个及时的反馈，通常情况下，应用软件的产品化必须是快速而敏捷的。

降低成本

从头重写系统要求对所有业务逻辑的开发也要推倒重来。对于一个旧版系统来说，重写软件摈弃了之前投入的大量时间和知识，而事实上这些时间和知识在软件重写过程中不可避免地需要被重新创建。

再工程尽可能地复用现有的业务逻辑代码，从而可以显著节省大量的时间和金钱。在再工程项目中，有显著数量的任务可以跳过而无须改动。而这些任务的总体数量是非常可观的。由于现有系统已包含绝大多数的需求，所以只需新增极少数的需求文档。这可以为需求分析人员节省下几周甚至几个月的调研和归档用户需求的时间。有了实现这些需求的业务逻辑，会节省更多的时间。

1990 年，W.M. Ulrich 为《American Programmer》杂志的十月刊撰写了一篇文章。在文章中，他描述了一个商业系统，该系统完全重写的成本约为 5 亿美金。事实上，该系统成功地进行了再工程，其总成本仅为 1.2 亿美金。相比全新的开发，再工程是非常经济的。

本书的读者对象

本书适合从事维护和运行系统工作的读者。对于技术经理和产品经理来说，本书描述了提高系统可靠性的必要过程，其使得系统运行更快，更易维护以及拥有更少的缺陷。对于架构师和开发人员来说，本书包含了新架构的选择方法以及完成关键部分的必要代码的详细描述。对于通过添加现代架构和自动化测试方法增量地改进应用程序的结构和质量，本书给出了详细的建议。按照本书中概括的步骤，可以提高应用程序的质量并使得添加新功能变得更加容易和快速。

再工程一个现有系统比从头开始构建一个等效的系统更加容易、便宜且风险低。如果你遵循本书中总结的建议，就可以提高开发的速度和最终的质量，同时还可以保持系统正常运行。

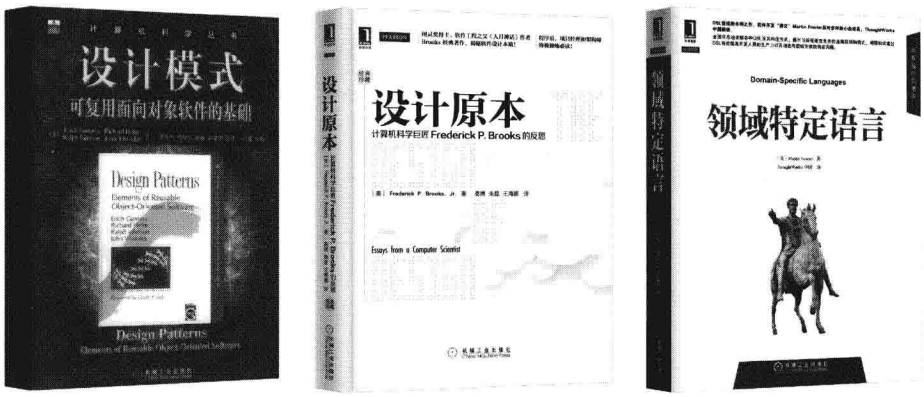
我希望你发现本书是一种很好的资源，并且可以节约时间。

致谢

没有许多朋友的帮助，我不可能完成这本书。首先，我想要感谢我的技术编辑 Joey Ebright 和 Peter Himschoot，他们在确保我的代码整洁、正确和切中要点方面做得非常出色。他们的架构洞察力和许多中肯的建议使得这本书更加完美。我还想要感谢我的编辑 Christopher Cleveland 和 Jeff Riley，他们对本书的结构和流程提供了有价值的建议。最后，我想要感谢 Joan Murray，她给了我自信去接受这个挑战。我希望有机会在另一本书中再次和所有人一起合作。

——Bradley Irby, bradirby.com

推荐阅读



设计模式：可复用面向对象软件的基础

本书是全球软件技术人员的圣经和经典，凝聚了软件开发界几十年的设计经验。四位顶尖的面向对象领域专家尽心挑选了最具价值的23种设计实践，加以分类整理和命名，并用简洁而易于重用的形式表达出来。这23个模式已成为开发界进行技术交流所必备的基本知识和语汇。四位作者由于在设计模式领域的杰出贡献，被广大开发者昵称为GoF (Gang of Four)，本书也因此而被称为GoF Book。

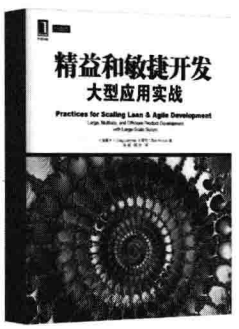
设计原本（精装本）

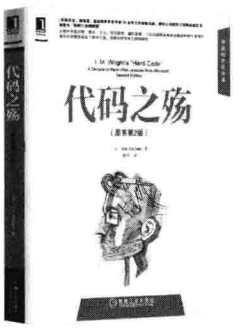
如果说《人月神话》是近40年来所有软件开发工程师和项目经理们必读的一本书，那么本书将会是未来数十年内从事软件行业的程序员、项目经理和架构师必读的一本书。它是《人月神话》作者、著名计算机科学家、软件工程教父、美国两院院士、图灵奖和IEEE计算机先驱奖得主Brooks在计算机软硬件架构与设计、建筑和组织机构的架构与设计等领域毕生经验的结晶，是计算机图书领域的又一史诗级著作。

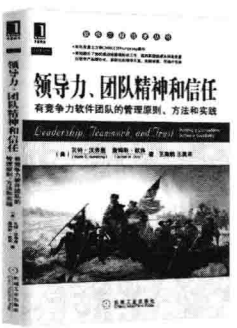
领域特定语言

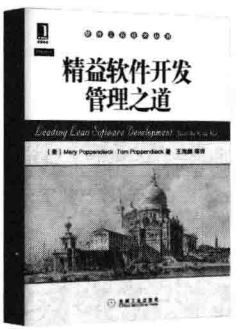
本书是DSL领域的丰碑之作，由世界级软件开发大师和软件开发“教父”Martin Fowler历时多年写作而成。全面详尽地讲解了各种DSL及其构造方式，揭示了与编程语言无关的通用原则和模式，阐释了如何通过DSL有效提高开发人员的生产力以及增进与领域专家的有效沟通，能为开发人员选择和使用DSL提供有效的决策依据和指导方法。

推荐阅读









- **精益和敏捷开发：大型应用实战**
  作者：（加）Craig Larman 等
  ISBN：978-7-111-32647-2
  定价：69.00元

- **代码之殇（原书第2版）**
  作者：（美）Eric Brechner
  ISBN：978-7-111-41682-1
  定价：79.00元

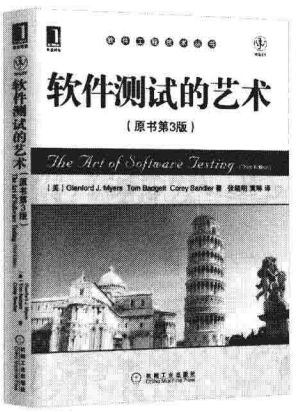
- **测试驱动开发：实战与模式解析**
  作者：（美）Kent Beck
  ISBN：978-7-111-42386-7
  定价：59.00元

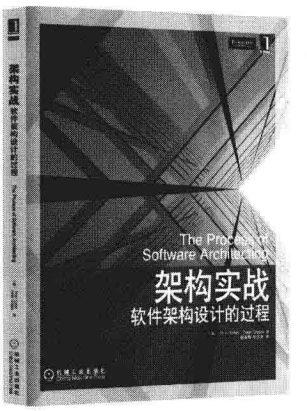
- **领导力、团队精神和信任**
  有竞争力软件团队的管理原则、方法和实践
  作者：（美）Watts S. Humphrey 等
  ISBN：978-7-111-38225-6
  定价：59.00元

- **死亡之旅（原书第2版）**
  作者：（美）Edward Yourdon
  ISBN：978-7-111-35998-2
  定价：49.00元

- **精益软件开发管理之道**
  作者：（美）Mary Poppendieck 等
  ISBN：978-7-111-34082-9
  定价：49.00元

推荐阅读





软件测试的艺术（原书第3版）

作者：Glenford J. Myers 等 译者：张晓明 等 ISBN：978-7-111-37660-6 定价：39.00元

面向使用的软件设计

作者：Larry L. Constantine 等 译者：刘正捷 等 ISBN：978-7-111-34575-6 定价：75.00元

架构实战——软件架构设计的过程

作者：Peter Eeles 等 译者：蔡黄辉 等 ISBN：978-7-111-30115-8 定价：45.00元

精益和敏捷开发大型应用实战

作者：Craig Larman 等 译者：孙媛 等 ISBN：978-7-111-32647-2 定价：69.00元

软件开发经济学

作者：Walker Royce 等 译者：苏敬凯 等 ISBN：978-7-111-30146-2 定价：29.80元

测试之美

作者：Tim Riley 等 译者：张爽 等 ISBN：978-7-111-30239-1 定价：55.00元

敏捷项目管理

作者：Jochen Krebs 译者：陈宗斌 等 ISBN：978-7-111-29698-0 定价：39.00元

软件项目管理与敏捷方法

作者：Michele Sliger 等 译者：李晓丽 等 ISBN：978-7-111-30193-6 定价：42.00元

软件开发成功路线图：敏捷模式

作者：Amr Elssamadisy 译者：初悦欣 等 ISBN：978-7-111-29943-1 定价：45.00元

目　录