/THEORY/IN/PRACTICE

精益软件开发艺术

The Art of Lean Software Development

Curt Hibbs, Steve Jewett & Mike Sullivan 著

章显洲 译

O'REILLY®

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

O'REILLY®

精益软件开发艺术

The Art of Lean Software Development

Curt Hibbs
Steve Jewett 著
Mike Sullivan

章显洲 译

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内容简介

这本简洁之作将向你解释如何应用精益软件开发的实践来显著提高生产力和质量。基于对日本制造业产生革命性改变的实践，精益原则正被成功应用到产品设计、工程、供应链管理等领域中，现在也被应用到软件开发领域中了。书中覆盖了从开始精益软件开发之旅时，所能采用的最重要的五项实践的详情。这些都是简单、可增量递进的步骤，一步一个脚印的前进，将能使软件开发不断获得精益的效果！

本书适用于那些新近接触精益软件开发（也许还包括敏捷软件开发）的软件开发者和管理人员。也适用于那些想要快速了解“为什么精益软件开发是重要的”，以及“它可以为我做什么”的读者。

版权贸易合同登记号　图字：01-2009-2862

图书在版编目（CIP）数据

精益软件开发艺术／（美）希布斯（Hibbs,C.），（美）朱（Jewett,J.），（美）沙利文（Sullivan,M.）著；章显洲 译. —北京：电子工业出版社，2009.6
书名原文:The Art of Lean Software Development
ISBN 978-7-121-08866-7

Ⅰ. 精… Ⅱ.①希…②朱…③沙…④章… Ⅲ.软件开发 Ⅳ.TP311.52

中国版本图书馆 CIP 数据核字（2009）第 077026 号

责任编辑：杨绣国
项目管理：梁　晶
印　　刷：北京智力达印刷有限公司
装　　订：北京中新伟业印刷有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：787×980　1/16　印张：9.25　字数：180 千字
印　　次：2009 年 6 月第 1 次印刷
定　　价：28.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。
质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。
服务热线：（010）88258888。

O'Reilly Media, Inc.介绍

为了满足读者对网络和软件技术知识的迫切需求，世界著名计算机图书出版机构O'Reilly Media, Inc. 授权电子工业出版社，翻译出版一批该公司久负盛名的英文经典技术专著。

O'Reilly Media, Inc.是世界上在Unix、X、Internet和其他开放系统图书领域具有领导地位的出版公司，同时也是联机出版的先锋。

从最畅销的《The Whole Internet User's Guide & Catalog》（被纽约公共图书馆评为20世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到WebSite（第一个桌面PC的Web服务器软件），O'Reilly Media, Inc.一直处于Internet发展的最前沿。

许多书店的反馈表明，O'Reilly Media, Inc. 是最稳定的计算机图书出版商——每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly Media,Inc. 具有深厚的计算机专业背景，这使得O'Reilly Media, Inc. 形成了一个非常不同于其他出版商的出版方针。O'Reilly Media, Inc. 所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly Media, Inc. 还有许多固定的作者群体——他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly Media, Inc. 依靠他们及时地推出图书。因为 O'Reilly Media, Inc. 紧密地与计算机业界联系着，所以O'Reilly Media, Inc. 知道市场上真正需要什么图书。

译者序

在现今，敏捷软件开发已经不是什么新鲜事物了——市面上已经有大量的书籍，网络上也有大量的页面聚焦于此领域。如果有人还总把“敏捷”挂在嘴上喋喋不休，反倒可能会被认为是后知后觉，落伍久矣。

简而言之，“敏捷”在口头上的热度已经慢慢冷却，在概念层面上的新鲜劲已经过去。现在，是让嘴巴休息而让双手忙碌的时候了！

但是，“千江有水千江月”，究竟该如何有效实施**“我的敏捷开发”**呢？

“琳琅满目的敏捷方法中，哪个最适合我们团队？该选 Scrum 还是极限编程（XP）呢？如果是 XP，它包含有一大堆的实践，我们要把这些实践一次性全部导入吗？但是，如果我们是在维护一个已有产品，大量的代码没有自动化单元测试，该怎样进行“持续集成”呢？作为经理，又该如何说服老板获得他的支持呢？……”

一堆问题！

由于无法在这些问题上获得令人满意的答案，这使得很多软件开发组织始终徘徊在“敏捷”的门口，不敢举足踏入；在没有深入思考和回答出这些问题前，就贸然地光凭一腔热情一头扎入，也正是很多软件组织实施“敏捷”失败的原因——仅模仿得“敏捷”之皮相，当然无法带来真正的实效。

知行合一，诚非易事！如果你有以上类似的困惑，如果你真诚期望团队能够更快更好地开发软件来创造真正的客户价值，但是感觉无从下手，那么，本书值得你拨冗一读。

本书将向你介绍一种可以以稳健的步伐逐步导入的“精益软件开发方法”。

它根源于“精益思想”这一通泛的哲理。“精益思想”涌现于日本的汽车制造业。丰田公司通过其独特的“丰田生产系统（Toyota Production System）”获得了巨大的成功，很快，日本汽车工业的高速发展对美国汽车工业带来了极大的威胁和挑战，麻省理工学院的研究小组对它进行调查研究后，挖掘出丰田的成功哲学，以“精益思想”名之。之后，“精益思想”在多个领域得到广泛深入的应用。

本书正是要向你提供一个在软件领域实施“精益生产”的导航图。依循本书指出的方向和路线，将可以通过实施持续的改善，释放出“精益思想”的强大力量。

到这里，你可能会问**“精益软件开发和敏捷软件开发之间是什么关系？精益软件开发难道不是诸多敏捷开发方法学流派中的一支吗？”**

☺ 请允许我先笑而不答！

且回到古代的日本看看，在日本战国时期，有高超的剑士从禅僧那里获得启示而迈入“剑禅合一”之境的案例。在禅门中，有“**守破离**”之说，禅尚不立文字，虽如以指指月，且强为说之：首先以前人所定之规矩为准，以之作为第一阶段的修炼，是为“**守**”；经历这种修炼，取得不断进步而到达某种程度后，藉由自己的用功和机智等将之突破，是为第二阶段“如桶底脱”的“**破**”境；然而依然持续“勇猛精进”的进行修炼，技艺更趋精纯，于不知不觉中升离，但一切还是不失法、不逾矩，到达一个独立开拓的境地，这是第三阶段的“**离**”境。

守，破，离，Shu，Ha， Ri……“万里无云万里天”！

对于一本题涉“精益”的书，它的译序也应该言简意赅才好，我还是赶紧闭上嘴巴隐退到文字之后，让本书作者展示他们关于软件开发的智慧吧！

反馈

虽然我们力求准确传递作者的哲思，但人非圣贤，译文中可能仍然难免有错误存在。

如果你发现书中有错讹之处，请联系我，以便添加到勘误表中。我的电子邮件地址是 *agileway.cn@gmail.com*。

致谢

非常感谢博文视点对我的信任，使我在离开机械工程专业 9 年之后，有了这个机缘，把软件开发和本科期间所学的机械工程知识进行了一次联结，让我有做完了一道 Cloze 题目的感觉！

感谢爸妈、爱妻对我长期的支持和宽容，使我能够在周末躲进书房安心伏案工作！特别想对小儿“多多”略表歉意，老爸抢了你的鼠标，让你好久不能玩 Ubuntu 上的小企鹅游戏，☺。

另外，要特别感谢阿里巴巴网站技术部的同学们，尤其是 Peleus 虚拟小组和“风林火山”项目组的同学们，和你们一起就书中的若干主题进行思考和探讨，实为快事！

最后，感谢读者朋友们，在繁忙中停驻脚步，翻开此书！

章显洲
于杭城城西
2009 年 5 月

前言

所有才华中最有价值的是，一个字能表达清楚的，绝对不用两个字。

——托马斯·杰斐逊

凡事都有起因。

有时候，能够指出引发某件事情的单一事件；而其他时候，则是各种想法、活动和思考，以某种我们永远不能完全理解的神秘方式凑到了一起，引发了某件事情。对于本书的写作缘由，无疑，“单一事件”的模式正好适用。

Curt 还记得整个事情就像是昨天刚发生的一样——他第一次坐下来与一位同事讨论精益软件开发。当他们讨论着把精益思想应用到软件开发上意味着什么时，同事问了一个简单的问题：“如果我只能做一件事情时，该做的是什么呢？”

这是一个很好的问题，这个问题就像一首歌一样萦绕在 Curt 的心中无法拂去。思考这个问题使他更好地了解到一点：软件开发者和管理人员是多么想相信精益和敏捷开发的方法能够帮助他们。他们觉得应该做些什么，但出于对失败的恐惧——这也很合情合理——他们不愿意（或不能）一下子就采纳一整套的方法学。

这就像给某人背上绑上一个时灵时不灵的新型喷气推进器，在要他跳过悬崖时，对他说：“相信我，这玩意儿会更快地把你带到深渊的对面去。”也许……但如果它失灵了，后果又会怎样呢？

如果这是你的感受，那么这本书就是写给你的。

数十年来，精益方法已经在制造业取得了显著的成果，现在，精益方法正同样地被成功应用到供应链、产品设计、工程，甚至是软件开发中！与此同时，敏捷软件开发方法也表明，它们之中的核心实践，那些和精益软件开发所推荐的非常类似的部分，具有很大的价值。

在每一种敏捷方法和精益软件开发的实现中，这些核心实践是一致的。绝妙的是，这些实践可以一次只采用一个，但仍然会带来相当大的好处——你无须在使用了整个庞大的实现后，才能看到好处。

很多人错误地认为，精益和敏捷是同样一件事情的两个不同名字。精益方法和敏捷软件开发具有相同的目标——提高质量和生产力，但它们却是采取不同的哲学方法（方法论）来达成目标的。本书第一部分将介绍精益软件开发的原则。我们会讨论精益和敏捷观点的不同及相似之处。

本书第二部分将根据价值大小，依次介绍这些核心实践。我们会告诉你，哪一项实践要首先采用，以使付出的努力获得最大的回报；而如果你已经这样做了，则会为你指出下一步应采用的实践。

这本书覆盖了从开始精益软件开发之旅时，所能采用的最重要的 5 项实践的详情。这些都是简单、可增量递进的步骤，一步一个脚印地前进，将能使软件开发不断获得精益的效果！

在掌握了每一项实践后，你将会看到显著的可衡量的结果。这些成功将会让你更深入地了解到把精益思想应用到软件开发中所产生的威力。

对软件开发过程做出越多改善，你就越会乐于去发现仍然存在的那些障碍（用精益的术语讲，叫“浪费”）。这将带给你知识和能力，并赖之开始做出自己的价值判断，持续改进自己的软件开发过程。毕竟，“精益”是一个旅程，而不是目的地！

谁应该读这本书
Who Should Read This Book?

这本书适用于那些新近接触精益软件开发（也许还包括敏捷软件开发）的软件开发者和管理人员。也适用于那些想要快速了解“为什么精益软件开发是重要的”，以及“它可以为我做什么”的读者。

我们特意把这本书写成章节短小的简洁之作。我们知道，你也一样忙碌，我们也信奉“不在书中充斥无用东西”的信条。书中每一个章节都尽量简明扼要，并做到尽可能地一语中的。我们的目标是要向你介绍重要的专题和资源，让你知道当需要更详细的信息时可以去什么地方寻找。

惯例约定
Conventions Used in This Book

下面列出了本书所采用的一些格式的含义：

斜体（*Italic*）

代表一些新的术语、链接地址、电子邮件地址、文件名和文件扩展名。

等宽字体（`Constant width`）

代表广义的计算机代码，包括命令、可选项、分支开关、变量、属性、快捷键、函数、类型、类、命名空间、方法、模块、参数、值、对象、事件、事件句柄、XML 标签、HTML 标签、文件内容，以及命令的输出结果。

使用代码示例
Using Code Examples

本书希望能帮助你做好工作。一般来说，你可以在程序和文档中使用本书中的代码。你不需要联系我们获取许可，除非你是要复制使用相当大段的代码。举例而言，使用本书中的一些代码片段来编写程序不需要许可。销售或分销 O'Reilly 书籍并包含样例的 CD-ROM

需要获得许可。引用或摘录本书中的示例代码来回答问题，不需要许可。把本书中相当大数量的代码纳入你自己的产品文档，则需要请求许可。

如果引用本书的内容，不要求你一定明确标明出处，不过如果你这么做，我们会很感激。标明内容，通常包括书名、作者、出版商和 ISBN。比如，“*The Art of Lean Software Development* by Curt Hibbs, Steve Jewett, and Mike Sullivan. Copyright 2009 Curt Hibbs, Stephen Jewett, and Mike Sullivan, 978-0-596-51731-1”。

如果你对于使用本书中的示例代码是否属于侵权行为还不太确定，随时可以通过下面的邮件 *permissions@oreilly.com* 和我们联系。

我们希望看到您的来信

We'd Like to Hear from You

如果你想就本书发表评论或有任何疑问，敬请联系出版社：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

奥莱利技术咨询（北京）有限公司

北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室
邮政编码：100055
网页：*http://www.oreilly.com.cn*
E-mail：*info@mail.oreilly.com.cn*

北京博文视点资讯有限公司（武汉分部）

湖北省武汉市洪山区吴家湾邮科院路特 1 号湖北信息产业科技大厦 1402 室
邮政编码：430074
电话：（027）87690813　　传真：（027）87690595
网页：*http://bv.csdn.net*
读者服务信箱：
reader@broadview.com.cn（读者信箱）
bvtougao@gmail.com　　（投稿信箱）

我们为本书准备了一个 Web 页面，在那里列出了勘误表，示例和其他附加信息。可以通过下面的网址访问：

http://oreilly.com/catalog/9780596517311/（原书）
http://www.oreilly.com.cn/book.php?bn=9787121088667（中文版）

想就本书发布评论或者询问技术问题，请发信到：

bookquestions@oreilly.com

想更详细了解我们的书籍、会议、资源中心，以及 O'Reilly Network，可以通过下面的网址访问网站：

http://www.oreilly.com

致谢

Acknowledgments

如果没有感谢到所有帮助本书成为现实的人，那是我们的粗心失职。感谢包括 O'Reilly 里那些认为本书值得出版的好人，以及我们的家人，他们允许我们像失踪了一般关起门来写作本书。

我们还想感谢那些给我们反馈，让这本书变得更好的那些早期的审校者，他们是：Kelly Carter、Ted Davis、Laurent Julliard、John McClenning、Phyllis Marbach、Bill Niebruegge、Ian Roth、Beth Simon、Tim Sullivan、Ed Thoms 和 Brian Wells。

但最重要的是，我们要感谢 Beth Simon，因为他问了那个简单的问题！

目录

Table of Contents

第 1 章

精益之由

Why Lean?

乐观主义是编程活动中的职业病。

——Kent Beck

几十年来，软件开发实践一直被惊人的低成功率所困扰。与此同时，软件产品和服务的数量却每年以惊人之势持续增长。如果仅依这两种态势发展下去，那么无疑我们将会迈向灾难。

幸运的是，敏捷软件开发方法正向世人昭示：高成功率是可能的。目前，精益技术（在近50 年中已在制造业显著提高了成功率）正被应用于软件开发领域；并验证着敏捷方法的成功之处。

精益原则及思维，已被证明非常适用于任何致力于提升生产力和质量的活动。精益方法已成功应用于制造业、分销、供应链、产品开发、银行业、工程、后台管理等诸多领域。然而，仅在最近这几年，精益原则和技术才被应用到软件开发领域中。

在本章，我们将就长期困扰软件开发的诸多问题，进行更详细的说明；同时也将就敏捷软件开发、精益开发的起源，以及它们用于改善各种过程的独特方法，作一个概览性的介绍。

1.1 软件开发中的问题

The Problem with Software Development

你曾经在这样的软件开发项目中工作过吗？

- 进度超期；
- 预算超支；
- 不符合客户需求；
- 被中途取消。

如果你说“没有”，那你可能是“初出茅庐”并正在做你人生中的第一个项目。

如果你说“是的”，我敢肯定，你并不孤单！

实际的统计结果，令人震惊。

1994 年，Standish Group 发表的 CHAOS 报告是关于“IT 项目之失败”研究的一个里程碑。截至 1994 年，Standish Group 在研究了超过 8000 个软件开发项目后发现：只有 16%软件项目是成功的。这意味着其他 84%的项目要么彻底失败，要么存在严重的问题。 到 2004 年，经过了又一个 10 年，其研究所覆盖的项目已增至 40 000 个，结果表明，成功率已提高到 29%。虽然这已是一个显著的进步，但仍然不值得吹嘘。

你能想象到任何其他一个行业中存在如此惊人的低成功率吗？

1.1.1 CHAOS 研究

The CHAOS Study

Standish Group 的调查覆盖了主要行业中的大、中、小型企业，这些行业包括：银行业，证券业，制造业，零售业，批发业，医疗保健，保险业，服务业和地方、州及联邦机构（总共 365 家企业在内）。在近 10 年间，他们研究了近 40000 个项目。此外，他们还使用了“焦点小组（focus groups）”和“个人访谈（person interviews）”的方法，来为调查结果提供定性分析背景（qualitative context）。

CHAOS 把研究的项目分为下面三类。

成功的项目

该项目在预算范围内按时完成，并交付初始定义的全部特征和功能。

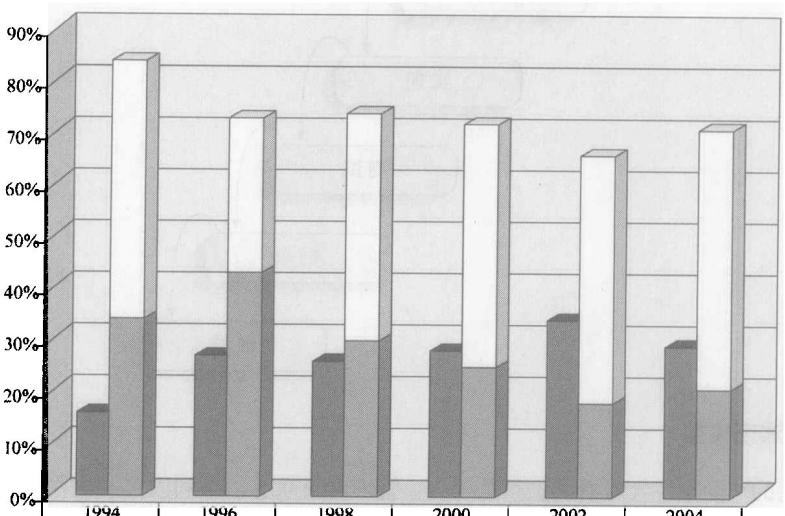
受质疑的项目

该项目虽然完成，但超出了预算，超出了预估的时间进度，并且交付的特征和功能少于初始的定义。

失败的项目

该项目在开发周期的某个点上被取消（在 CHAOS 研究中，称之为“**项目破损**”）。

图 1-1 是在 10 年间，这三类项目各占的百分比情况。



	1994	1996	1998	2000	2002	2004
■成功的	16%	27%	26%	28%	34%	29%
□受质疑的	53%	33%	46%	49%	51%	53%
■失败的	31%	40%	28%	23%	15%	18%

图 1-1：CHAOS 研究数据

诚如所见，成功率随着时间的推移是在提高，但却是以每年1.3%的极缓速度在增长。按照这种增长率，直到 2020 年，成功项目的比重也不会突破 50%大关！

公平地讲，一些专家不同意 CHAOS 研究中对“成功”的定义。他们指出，许多项目虽然被列为“受质疑的项目”，但却推出了非常成功的产品。然而，即使把这种争议因素考虑在内，软件项目的成功率仍有很大提升空间。

1.1.2 瀑布方法

The Waterfall Method

问题变成：为什么失败率如此之高？有很大一部分责任可以追溯到被广泛采用的**瀑布**方法。

软件开发的瀑布模型将开发过程分解为一系列依次完成的单独阶段：从需求、设计、实现、测试、部署到维护（参看图 1-2）。整个开发过程看起来就像是瀑布一样，稳定地向下依次经过这些阶段。每个阶段都有一个开始点和结束点，一旦到达下一阶段，就不能再回到上一阶段 （正如瀑布不会向山上倒流一样）。

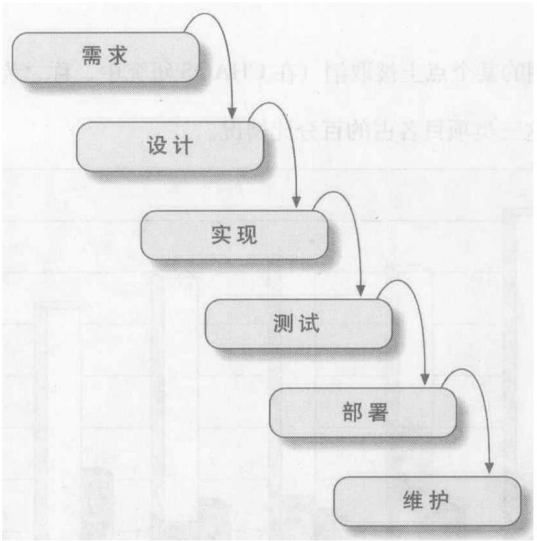


图 1-2：瀑布模型

瀑布方法的“简单”气质看上去很是诱人。管理者喜欢有一系列固定的里程碑，这看上去能够很方便地跟踪项目进度。然而，由于模型不允许变化发生，这种可管理性，其实只是一个幻象。