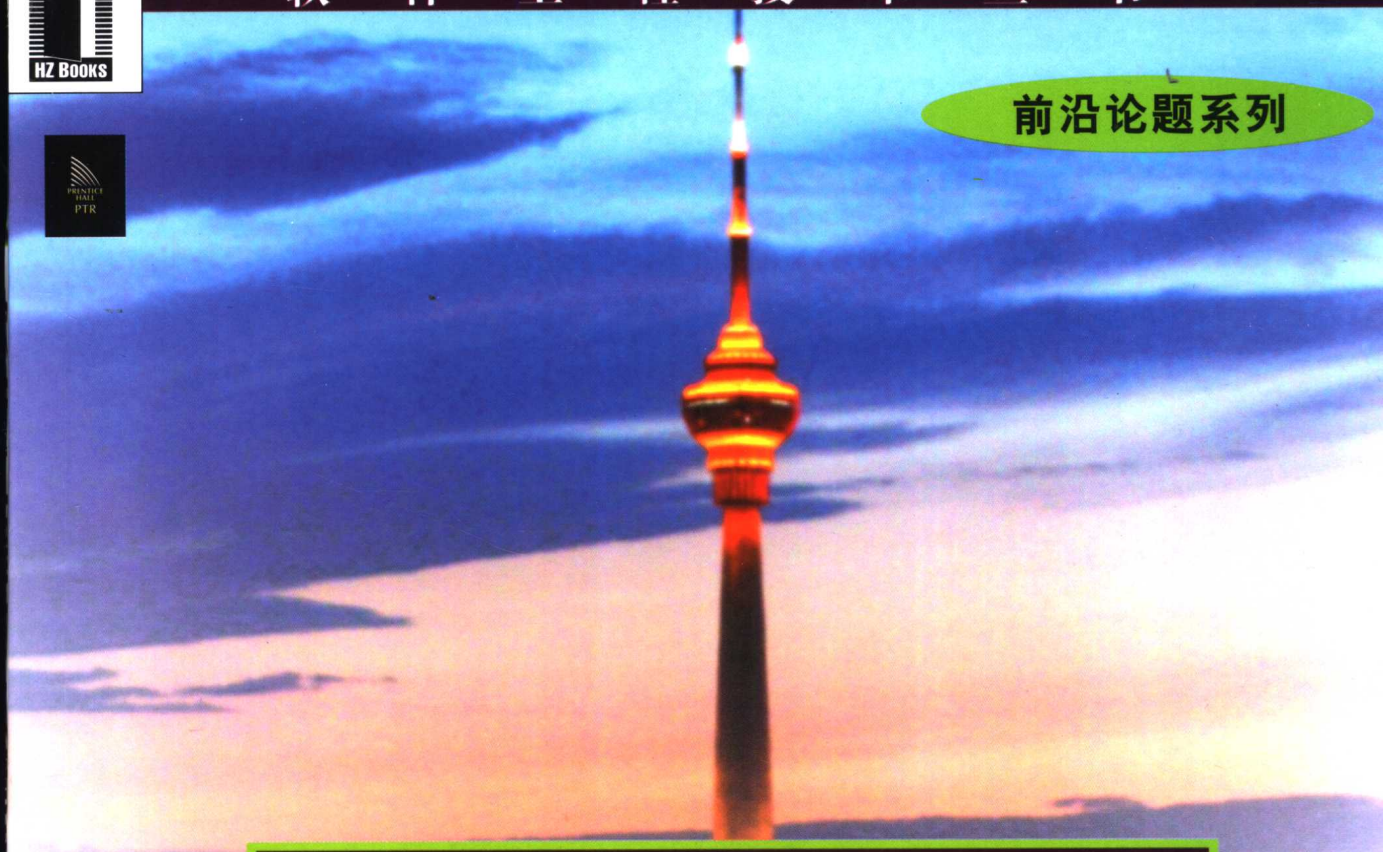
软 件 工 程 技 术 丛 书

HZ BOOKS

PRENTICE HALL PTR

前沿论题系列



软件工程的敏捷管理

Agile Management for Software Engineering

Applying the Theory of Constraints for Business Results

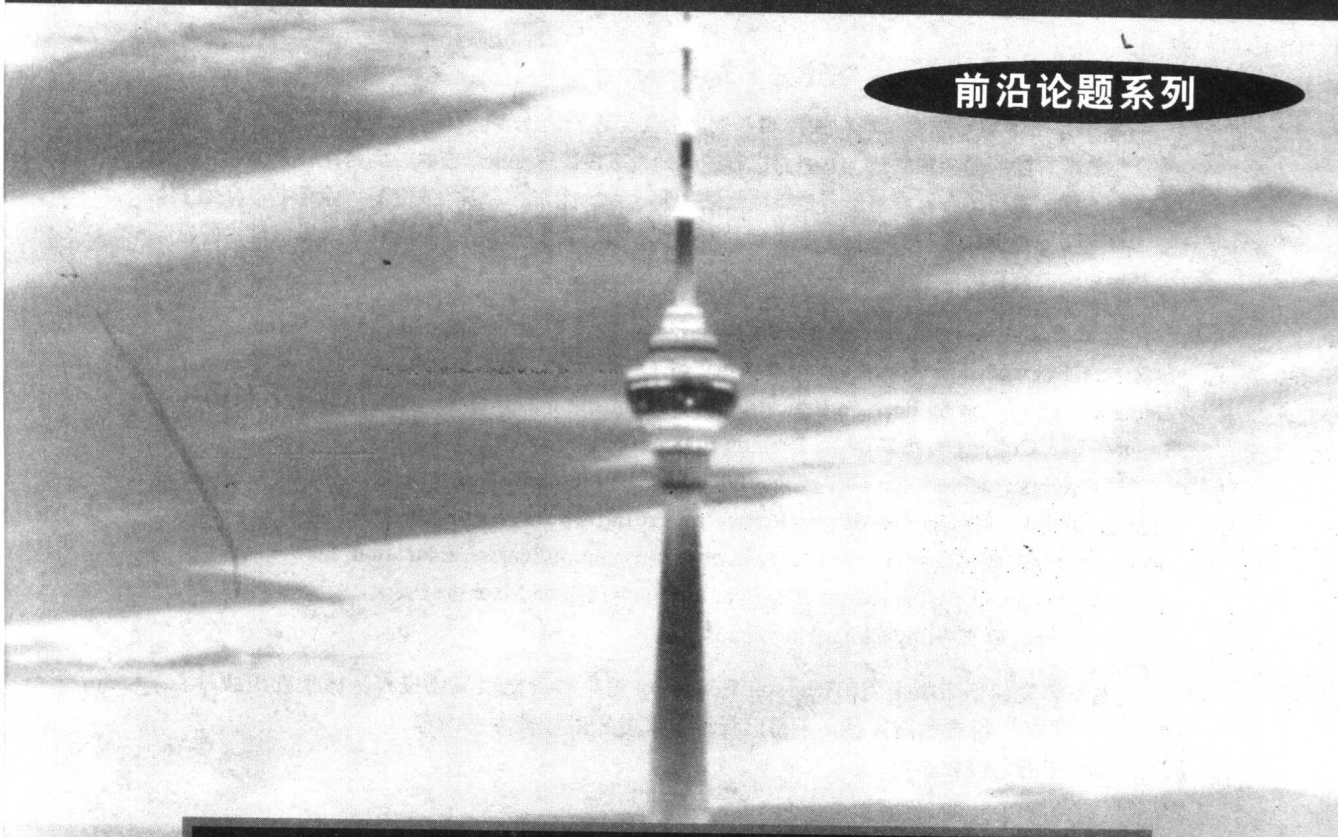
（美） David J. Anderson 著

韩 柯 等译

机械工业出版社
China Machine Press

软 件 工 程 技 术 丛 书

前沿论题系列



软件工程的敏捷管理

Agile Management for Software Engineering

Applying the Theory of Constraints for Business Results

（美） David J. Anderson 著

韩 柯 等译

机械工业出版社
China Machine Press

本书是一本观点鲜明、新颖独特的专著，全面论述当前比较流行的软件生产敏捷方法，着重介绍敏捷方法的理念和创新。书中并没有简单地否定传统软件生产方法，而是比较全面地分析了各种方法的适用场合。本书作者是特征驱动开发这种敏捷方法的创始人之一，他在书中介绍了很多自己亲身负责和参与的项目管理实例。本书具有很好的参考价值，适合软件开发经理、开发人员、用户以及在校学生阅读。

本书版权登记号：图字：01-2003-8123

图书在版编目（CIP）数据

软件工程的敏捷管理/（美）安德森（Anderson, D. J.）著；韩柯等译. - 北京：机械工业出版社，2004.8

（软件工程技术丛书　前沿论题系列）

书名原文：Agile Management for Software Engineering: Applying the Theory of Constraints for Business Results

ISBN 7-111-14519-4

I. 软… II. ①安… ②韩… III. 软件工程 IV. TP311.5

中国版本图书馆CIP数据核字（2004）第052366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：李晓燕　姚　蕾

北京瑞德印刷有限公司印刷 · 新华书店北京发行所发行

2004 年8月第1版第1次印刷

787mm × 1092mm 1/16 · 18.75印张

印数：0 001-4 000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

序　言

本书对软件界，对软件工程背后，尤其是管理软件机构背后的一些基本业务假设提出挑战，这实在令人兴奋。在撰写这篇序时，全世界的软件企业都在面临巨大的困难。我希望这些困难会使人们产生一种重新审视企业的愿望，一种鼓起勇气思考变革的愿望。其他行业，特别是制造业，在20世纪80、90年代已经完成了这种根本变革。这种变革当然不容易，不过根据我个人的经验看，我们非常需要这种变革。

1985年，我在以色列创办了自己的软件公司，对于自己开发的供经过认证的公共会计师使用的软件包感到很自豪。但是，即使我的软件包在市场中具有很强的竞争力，我还是注意到一种现实的业务问题：需要越来越多的开发人员对这个软件包提供支持。在这种情况下，我该怎样判断自己的投入？事实上，我不知道只占市场一席之地的小型软件公司是否是一个好的企业，尽管其产品本身受到市场的热烈欢迎。我感到即使我已经拿到了MBA，但仍然需要对企业有更深入地理解。

后来我见到了Eli Goldratt博士。我已经听到过很多关于Goldratt博士的国际软件公司Creative Output的信息，他的公司看起来远远不只是勇于创新的出色软件公司，它向企业的一些最神圣的规则提出了挑战，例如产品成本的概念。我不理解怎么会有人否认一个产品单元具有与其关联的一定成本这一基本概念。我很有兴趣准备接受新的工作：加入Creative Output公司，开发一种供经理使用的视频游戏，为其提供一些新的管理思想。那时的计算机游戏还只限于锻炼玩家的手指和协调能力，当然不能吸引成年人。怎样才能使计算机游戏能够被成年经理们接受，并向其提供新的管理思想呢？

这是深入思考一种抓住现实问题不放的新管理哲学的开端。我自己做了管理顾问，专门研究针对机构特定目标的改进。软件成为重要的支持工具，但并不是变革工作的中心。

约束理论（Theory of Constraints，TOC）可以在以下两个方面用于软件：

1）极大地改进推向市场的新产品流。

2）确定待开发项目，甚至只是某个特性对最终用户的实际价值。其背后的假设是，如果我们知道产品或特性对于用户的实际价值，就能够制定出合适的营销策略，为用户和软件公司带来这种实际价值。

David Anderson在这本书中主要关注第一个问题，包括研究业务案例，确保有能力进行改进。在已经改变了传统西方工业的新一代管理理念的帮助下，软件公司肯定会得到改进。David很好地将一般管理思想和观点与软件方法结合起来，形成了一种一致的企业改进方法。

阅读本书时应报有这样的目标：**学会怎样做到事半功倍。**不要只因为是David这样说的，就全盘接受他的所有观点。如果确实想做到事半功倍，就需要有能力使这些观点成为自己的观点。

译者序

这是一本观点鲜明、新颖独特的专著。作者是特征驱动开发这种敏捷方法的创始人之一，从20世纪制造业的发展成熟开始，令人信服地论述了软件工程的发展历史，站在很高的层次，叙述了敏捷方法的根源和基本原理，具有很高的权威性。

本书以软件生产系统为背景，涉及从构想逐步发展为实际运行代码的全部过程，提出软件生产系统的评估准则，全面论述当前比较流行的软件生产敏捷方法，着重介绍敏捷方法的理念和创新。作者并没有简单地否定传统软件的生产方法，而是比较全面地分析各种方法的适用场合。

本书并没有单纯论述技术问题，而是使用了大量篇幅描述敏捷项目的管理和心理学问题。作者在书中介绍很多自己亲身负责和参与的项目管理实例，具有很好的参考价值，相信会对关心软件生产系统发展和运营的经理、开发人员、用户，以及将要成为信息系统的经理、开发人员和用户的各类在校学生，有很大的帮助。

在翻译过程中，我们力求忠实原文。由于译者的知识水平和实际工作经验有限，不当之处在所难免，恳请读者批评指正。参加本书翻译、审校和其他辅助工作的还有耿民、朱军、孟海军、黄慧菊、屈健、刘芙蓉、王威、李津津、原小玲、韩文臣等。

译 者

2004年5月

我要说的是，应该为这种转变提供一种机会，花时间仔细地对自己的环境进行反思，然后自己确定应该怎么办。战胜自我，对于任何机构中的所有真正优秀的经理来说，都是最大的挑战。当然，草率跟随新时尚会更糟糕。保持思想开放，正视与基本假设矛盾的新思想，这才是我建议读者应该追求的目标。本书是供读者用来奋斗的，它既不是讨论分支末节的，也不是一种时尚。如果读者喜欢自己正在采用的方法，那么是否要检验新的思想改进自己的方法则完全取决于读者自己。

以下是评估新特性，特别是新的软件包对潜在客户价值的一些简要观点。

只有当新特性能够消除或明显降低原有产品的限制时，才能对用户带来价值。价值量取决于所消除的限制，不取决于特性本身的强弱。举一个简单例子。在字处理系统的某一发展时期，有人曾经有过一种想法：为什么不在软件包中增加拼写检查器呢？

现在已经成为普通特性的拼写检查的价值是什么呢？它所消除或降低的限制是什么呢？对于具有很高语言水平的人来说，拼写错误是由于写得太快。因此，如果没有拼写检查器，这些人需要仔细阅读已经写下的内容。语言水平不太高的人（例如我这个以色列人）需要经常查阅字典，这很费时间。

这种需要使我们意识到以下另外两点。

人们制定规则是为了克服这些限制。使用字处理系统的人在把文档发送给别人之前，需要把所写的全部内容再看一遍。语言水平不太高的人则需要借助字典。

一旦明显降低了这种限制，人们就会采用具有清除限制这一优势的新规则取代旧规则。如果没有取代，则该特性没有价值。

下面讨论在现有字处理系统上增加拼写检查器是否带来价值。假设读者具有很高的语言水平，在把最近写好的文档发送出去之前，是否不再仔细阅读了呢？拼写错误并不是仔细检查要让别人阅读的任何文档的主要理由。因此，对于具有很高语言水平的人来说，拼写检查器并没有提供价值。但是对于我这个掌握了希伯来文，不过英文水平并不高的人来说，英文拼写错误是很头疼的事。可是我能够仅靠使用拼写检查器真正避免编写错误吗？只要拼写检查器没有提出如何正确拼写单词的建议，限制就仅仅是被部分地降低，因此没有产生多大价值。这意味着如果要为语言水平不太高的特定用户群提供很大的价值，就需要补充提出应该如何拼写单词的建议。

在这个经过简化的例子中，我们已经看到在限制被清除之前和之后，都需要检查行为规则。所有人都很清楚应该增加什么新的行为规则吗？假想用户非常了解应该增加什么新规则是非常常见的陷阱。

假设要在销售图表显示模块中增加一个新特性，分析具有统计意义的曲线发展趋势。其限制就是不了解市场需求实际是增加还是降低，或者是正态统计曲线的一部分。管理层当前的做法是：如果销售量上升，要表扬销售代理，并相应地发放奖金。如果销售量下降，就不会发放奖金，并提出批评。

一旦管理层知道销售量的情况并打算采用新管理规则时，恐怕在绝大多数情况下采取的措施都一样。因此，新增加的特性没有对客户增加新的价值，尽管有些客户可能会要求增加这样的特性，甚至施加压力实现该特性。实际上对于开发该特性的软件公司来说，价值是负的。

当然，对于帮助更好地完成决策过程的优秀管理咨询公司来说，有的特性会为咨询公司和客户都带来巨大的价值。在这种情况下，咨询公司和软件公司之间的战略伙伴关系可以为所有人（包括客户）带来收益。

改进真正能为客户带来的价值，同时又为软件公司提供产生收益的很好机会的特性，正是这本独特的著作所要讨论的内容。《敏捷宣言》的原则是："我们的最高任务，就是通过尽早并且持续提供有价值的软件满足客户的需要"，这是充分结合"约束理论"的目标。更确切地说，约束理论要努力更接近机构的目标。但是为了做到这一点，机构必须为其客户提供更多的价值。尽早并且持续提供真正产生价值的软件，既会帮助软件公司也会帮助其客户更接近各自的目标。

请记住，改进的准则只有一个：更接近目标。真正改进软件机构运营方式的途径也许就从本书开始。

Eli Schragenheim

引　言

“与其他因素相比，管理水平差将导致软件成本更迅速地增加。”

——Barry Boehm[⊖]

为什么要提出敏捷管理

正如Barry Boehm所指出的那样，管理差会导致金钱的浪费[1981]。今天，管理差还会浪费工作。对不断攀升的软件开发成本感到困惑、对差的结果和服务以及没有透明性感到沮丧的高级执行经理，只能耸耸肩膀说：“如果没有办法能够把软件开发搞好，那就让我们把成本降下来吧。”这种思路的结果，就是把软件开发工作转移到海外低薪国家并在本土裁员。由于2003年的经济状况不佳，这种做法已经逐渐形成发展趋势。如果不让这种发展趋势形成潮流，IT公司中的管理就需要做得更好。软件开发的成本必须降下来，要生产更好的产品，即产品要更可靠，提供更好的客户服务，更具有透明性。本书就是要教会“敏捷经理”如何做到这一点。

经济上的必要性

构建软件需要大量资金，因为软件开发是劳动密集型知识工作。

软件工程师及其项目管理和相关部门的同事得到的报酬都很高。这是一个基本的供给与需求问题。在我经历的大部分时间里，对IT专业人员的需求一直大于供给，报酬水平也在相应地提高。大多数40岁以下软件工程师的收入，都要高于传统职业（例如医药、法律和财会等）同等条件员工的收入。在很多美国公司中，软件工程师的收入要高于营销人员的收入。

最近，随着全球经济的衰退，大型公司和很多小型公司都开始关注压缩成本提高效益或减少损失。IT业的高投入正在面临压力。首席信息官们正在削减预算。结果是，工作岗位被转移到海外，把软件开发外包给亚洲、澳大利亚和东欧的公司。知识工作正在从富裕国家转移到比较贫穷的国家。印度外包提供商的员工成本一般只是美国同等软件开发人员成本的25%。

如果要把软件知识工作保留在富裕的发达国家，并且美国、欧洲和日本软件工程师又要维持自己已经习惯了的很高的工资水平，就必须提高竞争力。软件开发已经形成全球市场，通信系统（例如因特网）的发展，已经使北美客户和印度等地供应商之间的时间和距离差变得不再重要。

工作岗位在动摇！就像20世纪后50年西方制造业受到亚洲崛起的威胁一样，西方知识工人

⊖ [Boehm 1981]《软件工程经济学》（Software Engineering Economics）。

界也会受到只需1/10~1/4的报酬就可以完成同样工作的接受过良好教育的勤奋劳动力的威胁。

答案并不是软件开发人员要想保住饭碗就必须更勤奋地工作。问题不在软件工程师身上。答案是管理手段必须改进，工作实践必须改变，以便交付更大的价值，从而提高竞争力。

敏捷管理观点

本书要讨论软件工程管理，还要讨论企业。本书要讨论的是怎样为了获得业务成功而管理软件工程，要证明敏捷软件开发方法是企业管理软件工程的更好方法。

IT界一直没有解决好软件工程管理问题，一直没有人表现出管理和过程控制方面的才能。结果，IT企业常常要靠凭直觉的即兴发挥和粗略估计运营。IT项目很少能够按计划完成，经常延期、超支、不能交付所承诺交付的软件特性，这种情况很普遍，已经到了正在被行业作为标准实践接受的程度。

软件工程管理工作一直做得很差，这可能是因为培训做得很差或很不够。只是最近，我所在地区的大学，即华盛顿大学才开始开设高技术管理MBA课程。这样的课程开设得很少，使得软件业几乎没有什么管理方面的专门知识。

但是，确实存在着很多能够提高软件开发企业竞争力的技术。这些技术已经在其他行业得到验证，问题是怎样才能运用到软件开发上来。像“约束理论”[Goldratt 1990a]、“精益生产”[Womack 1991]、“系统思维”[Senge 1990]这样的技术，以及从“复杂自适应系统”这种最新科学演化出来的新思想，都有助于发挥知识工人天才的潜在能力。

在经济上可行的软件工程的秘密，是以新的管理科学为基础的新的工作实践。敏捷经理必须构建一种使知识工人能够发挥才能的敏捷学习机构，一旦做到这一点，效果就会是很明显的。最起码能够得到4倍的改进，10倍的改进肯定也是有可能的。想像一下，如果自己的软件工程机构能够以目前一半的时间完成5倍的工作会是怎样的效果，这对你、你的工作岗位和你的机构意味着什么？

接受不确定，依靠透明性管理

知识工作与制造工作不同。完成在汽车车身上印字这样工作，具有很高的确定性，可以在很低的误差范围内很可靠地完成，很少出现失效和错误。在两辆汽车车身上印字所需的时间，几乎精确地等于在一辆汽车车身上印字所需时间的两倍。在100辆汽车车身上印字所需时间，可以通过在一辆汽车车身上印字所需时间乘以100计算出来。制造工作在很多方面都是可预测的、线性的，如果是机械过程，还是可以用科学规则定义的。

知识工作既不是线性的，也不是已定义的，因此知识工作与制造工作不同。人们一直认为，既然知识工作与制造工作不同，是不可预测的、非线性的，那么就不能采用相同的管理方式。事实上，尝试将传统管理方法引入软件工程的过程往往都会碰壁。软件项目极少，如果说还有的话，能够按计划执行，估计开发过程很难做到，对最终结果的估计往往纯粹是天方夜谭。从

董事会的观点看，软件开发是不可控制的。

本书将说明，放任软件工程成为一种不可控制的过程是错误的。软件工程可以像企业其他部门那样得到管理。秘密就是正确管理，就是以透明的方式管理。与制造生产工作相比，软件工程有更大的不确定性，这并不意味着管理方法就是无效的，而只是意味着这些方法必须适应更大的不确定性。约束理论教导经理人员如何缓解不确定性，而本书则要解释如何把这种手段运用到软件开发中。重要的是价值链合作伙伴、管理层和有关利益方理解软件开发管理的正确模型，即适应不确定性的模型，并学会信任敏捷经理手段。

敏捷经理的新工作，变成研究确定控制软件生产系统的支配规则。敏捷经理需要了解要跟踪什么、如何跟踪、如何解释跟踪结果以及要向高级管理层报告什么。本书要解释敏捷经理要做什么、为什么这样做和怎样做。

挫折感

美国西海岸的一些IT专业人员正在放弃自己的专业，谋求其他新的职业发展。在全世界范围内，IT人员都在醒悟。他们开始认识到高技术工作岗位并不值得牺牲家庭生活、社会生活或自己的健康，发现考虑到按公司期望所作出的无报酬加班牺牲，自己按小时计算的报酬并不算高，认识到生活中一定还有更有价值的东西。

我以前在新加坡的一位同事最近经过培训改行当了艺术家和摄影师，我在堪萨斯城工作时的一位同事辞职搬迁到法国的巴黎，在一个非盈利机构内工作，另一位同事最近为了经营汽车保养公司而辞职。还有其他一些作为承包商的同事只准备承担非全职工作。其中一位选择利用其余时间在一家鞋店工作，另一位则选择在插花店里工作。我所遇到的业内人士也都提到过类似的事情。到底怎么了？

人们选择IT工作有四种原因：事业（机构的长远发展和领导岗位）、对技术的热爱（通常是充满几乎是宗教激情的选择）、金钱（通常报酬相当不错）和老板（人们实际上是为人而工作）。下面分别讨论这些因素。

事业和技术常常可以放在一起讨论，包括公司的使命、未来前景、正在采用的技术以及采用这些技术的行业。例如，有些IT人员只是永远不在国防公司中工作。建立宏伟事业吸引人员是伟大领导考虑的事情。近年来，有很多文章讨论领导问题。也许很快就有论述IT业领导的专著问世，但是本书并不想讨论领导问题。

金钱很重要。IT人员很好找工作，需求大于供给。即使在困难时期，对IT人员的需求也很强烈。劳动力需求衰退常常是因为自动化系统可以替代其他工人，并降低成本。例如航空公司最近普遍采用了自动化的检票系统。

老板与本书要讨论的内容有很大关系。如果老板不能理解员工的愿望，员工就会醒悟并离开。IT公司很高的人员流动率通常说明管理“不能理解员工的愿望”。管理是很重要的。人们喜欢在管理有序、井井有条的环境中工作，希望有明确的目标和干一番大事业的环境。

本书要为IT界的老板提供一组新的管理工具，讨论怎样像评价企业内部其他部门那样地评价IT部门，讨论怎样演示IT业是否真正提供了增值，并产生了合适的投资回报。

把软件工程作为一种一般业务经营，实际上会产生更优化的使用资源、更高效地生产代码以及为员工提供更好的发挥创造性和职业发展环境的效果。如果能够理解员工的愿望，员工就会感觉到并喜欢企业。降低人员流动率、提高软件开发机构性能的关键是更好的管理。

敏捷宣言

最近，业界正在出现一种对不断严重的性能低下、超时过长、质量低劣、客户不满和开发人员有挫折感问题的反抗。这是对不良管理的反抗。一个充满激情的软件开发者团体提出，一定有更好的方法，交付软件应该具有更好的可预测性。这些充满激情的开发者支持采用一些新的软件开发方法，他们认为这些方法能够使软件开发更快、更省、更好，能够按时、按预算交付所约定的软件。这些新方法合在一起叫作敏捷方法。

“敏捷”这个词意味着灵活、易反应，按照达尔文的思想，就是具有适应变化的天生的能力。具有敏捷特性的事物具有“普遍适应”的能力。这意味着敏捷软件开发方法应该在不断变化的环境中生存，并取得成功。

2001年2月，软件过程方法论学者峰会接受了敏捷方法的定义，并形成了《敏捷软件开发宣言》[⊖]。这份宣言是由一个17人的小组起草的，目的是说明什么是敏捷软件开发，那时叫作“轻型”方法。轻型方法开始于“快速应用开发”（RAD）。RAD方法寻求严格交付时间的时间盒软件版本，为其分配项目中的所有其他要素，包括预算、特性集和人员，以满足交付日期要求。“快速”这个词来自时间盒特性的启发，这要比传统软件开发快得多，即两周到三个月。

敏捷方法主要是从RAD方法导出的。这些方法补充了另一个考虑要素，主要是因为人们认识到软件开发是一种人员活动，必须按照人员活动的客观规律进行管理。

敏捷软件开发宣言

我们正在通过实践和帮助其他人实践，揭示更好的软件开发。我们的价值观是：

人和交互重于过程和工具

能够发挥作用的软件重于面面俱到的文档

客户协作重于合同谈判

随时应对变化重于遵循计划

也就是说，虽然我们也关注以上右侧的内容，但是更关注左侧的内容。

Kent Beck、James Grenning、Robert C. Martin、Mike Beedle、Jim Highsmith、Steve Mellor、Arie Van Bennekum、Andrew Hunt、Ken Schwaber、Alistair Cockburn、Ron Jeffries、Jeff Sutherland、Ward Cunningham、John Kern、Dave Thomas、Martin Fowler和Brian Marick.

⊖ http://www.agilemanifesto.org/.

人们逐渐认识到，《敏捷宣言》是一份寻求反传统软件开发观点的非常简明的声明。它的基础是以下12条原则[⊖]：

我们的首要任务，是通过尽早并且持续提供有价值的软件满足客户的需要。

欢迎发生变更的需求，即使需求在开发后期发生变更。敏捷过程驾驭变更使客户获得竞争优势。

频繁提供能够发挥作用的软件，间隔时间从几周到几个月，时间间隔越短越好。

在整个项目期间，业务人员和开发人员每天都必须在一起工作。

围绕士气旺盛的人进行软件开发，向他们提供所需的环境和支持，相信他们能够完成任务。

向开发团队和在开发团队内部传递信息最高效、最有效的方法，就是面对面的交流。

能够发挥作用的软件是工作进展的主要度量。

敏捷过程提倡可持续开发。出资方、开发方和用户都应该能够坚定地维持一种稳定的步调。

对卓越技术和良好设计的不断追求可提高敏捷性。

简单——尽可能较少工作量是至关重要的。

最好的体系结构、需求和设计都来自自愿组合团队。

团队定期反思如何提高有效性，并相应地调整自己的行为。

有一些敏捷方法。本书将在第二部分详细讨论其中的三种方法：极限编程（Extreme Programming，XP）、FDD（Feature Driven Development）和Scrum。虽然没有讨论其他敏捷方法，但是本书还要提供基本指南和测度指标，通过与更传统的软件开发方法进行比较，对每种方法作出恰当的评估。

敏捷方法的问题

敏捷方法提出一些不同寻常的工作实践。顾名思义，极限编程有一些更激进的因素，从名称上听起来，往往会与极限运动联系起来。陌生的术语和陌生的实践吓退了大型公司的管理人员，他们能拿自己的职业发展、声望和优厚报酬冒险吗？

敏捷方法引入了令人恐慌的违反直觉的工作实践。如果要消除管理层的疑虑，就必须向管理层提供减轻这些疑虑的方法，需要提供报告管理层能够相信和熟悉，并且对业务有意义的统

⊖ http://www.agilemanifesto.org/principles.html。Kent Beck、James Grenning、Robert C. Martin、Mike Beedle、Jim Highsmith、Steve Mellor、Arie Van Bennekum、Andrew Hunt、Ken Schwaber、Alistair Cockburn、Ron Jeffries、Jeff Sutherland、Ward Cunningham、John Kern、Dave Thomas、Martin Fowler和Brian Marick.

计结果的方法。需要展示经济上的优点并关注实际业务收益。软件开发是为了获得更大的效益，而不是为了生产更多的代码。通过进行经济上的讨论，大型公司的高级经理可以确信报酬很高的IT人员已经理解了真正的目标。本书将讨论如何使软件工程机构更加成熟，达到能够报告可以相信的财务测度的程度，讨论应该选择什么测度，如何进行计算。

敏捷方法：是时尚还是发展趋势

敏捷方法宣称能够解决很多问题，但是证据何在？敏捷方法论学者会回答说，“证据就在甜点中。”换句话说，自己尝试一下就知道了。大家以前听到过这类说法。读者还记得第四代语言倡导者曾经宣称要消灭开发人员，让普通开发人员自己创建代替手工工作的工具吗？读者是否也曾被卷入通过组件组装可视软件的潮流中？Visual Basic的诞生是否真正消灭了开发人员？IT界充满了宣称。为什么敏捷方法就不能像其他方法那样宣称自己的目标呢？

敏捷方法究竟是改变工作实践的一种大趋势，还是使软件人员“逃避”现实的另一种时尚？本书将说明，敏捷方法是为制造业带来变革的“精益生产”和“约束理论”的翻版。

敏捷软件开发实际上要改变工作实践，改变管理风格。敏捷方法理解管理层真正想要的是什么，理解管理并不只是经济和工程，更多的是关注人员。“准确地说，管理是一门从有助于人们使自己和自己的世界更理性的原则中提取出来的人文教育艺术。这就是值得进行管理的最本质的原因”[Magretta 2002，第3页]。由于有这种现有经验的基础，我坚信敏捷方法是一种大趋势，要对软件生产方法工作实践和范例进行变革，而不是一种时尚。

企业效益

为了在大型企业中采纳敏捷方法，只是到董事会上说“别人是这样说的。我们为什么不能试一下呢？”是不够的。首席信息官很可能是实用主义者，不大可能很早采用新思想，不愿意冒风险。需要以能够展示更高的效益和投资回报的过硬数据为基础，例举业务实例。这是使敏捷方法具有吸引力，与把软件工程外包到海外的短期思维进行抗争的惟一方法。

本书将向敏捷经理提供使企业具有敏捷性的材料。可以通过提高增值和投资回报正确认识敏捷方法。本书将帮助敏捷经理收集和利用财务数据打消别人的疑虑。

本书提出一种科学地度量和评估敏捷方法的框架。使用测度确定增值和投资回报水平。这种框架的基础在很大程度上是Eli Goldratt及其同事所完成的工作。这是一种叫作“产出核算”[Corbett 1997]知识框架。基于约束理论在制造业生产中的应用的生产率核算被用作财务论证的基础。

创造软件经济奇迹

20世纪70年代和80年代，当西方国家正在致力于通过在装配生产线上使用机器人提高自动化程度的时候，日本通过采用变革工作实践的管理手段取得了好得多的效果。这些工作实践源

自丰田公司，所以称为“丰田生产系统”，也就是Kanban法。

在过去的30年中，技术行业也与西方国家制造业一样，一直致力于寻找技术解决方案。人们提出了第三代语言和第四代语言、建模和抽象工具、自动集成开发环境和自动测试工具。在一定程度上，敏捷学派拒绝这些还在不断发展的技术方法，采用新的管理手段改变工作实践。从这个意义上说，敏捷方法很像最初由丰田公司提出、现在被西方国家称作“精益生产”的原则。

在20世纪的下半叶，精益生产手段使经济提高了20~50倍。例如，Womack及其同事[Womack 1991]指出，丰田公司在最近的一年时间里，利用不到通用汽车公司5%的人员，生产出达到通用汽车公司一半的汽车。换句话说，丰田公司利用精益生产，与美国的大型生产竞争对手相比，生产率提高了10倍。有些敏捷倡导者提出精益生产可提高经济4倍[Highsmith 2002; Schwaber 2002]。这相当于20世纪下半叶初期日本汽车制造业的改进水平。例如马自达公司在20世纪60年代到1980年，生产率提高了4倍。在最近的20年中，其中一些制造商继续取得提高生产率5倍以上的成绩，所产生的累积经济增长达到20倍以上。恰恰是这类增长创造了20世纪末期的亚洲经济奇迹，并为全世界提供了巨大的财富。

如果敏捷软件开发能够提高生产率4倍，在9个月内完成项目，为什么还要把软件工程外包给生产成本只有1/4的印度提供商，在3~4年内完成呢？

目前，全世界软件业共聘用员工3 000多万人[㊀]，世界最富有的公司——微软公司也在其中。如果有可能再创造一个经济奇迹会有什么结果呢？如果软件开发生产率相当于制造业1925年的水平，会出现什么情况？敏捷方法只是代表了对如何更好、更快、更经济地生产软件方法理解的开始。这是否预示着95%数量级的潜在经济增长正在等待释放呢？敏捷方法是向精益IT行业道路发展迈出的第一步。敏捷方法确实产生出经济效益，本书要说明如何计算这种效益。

本书对象

Joan Magretta在他的《What Management is》(管理是什么？)[2002,第2页]这本书的序言中指出，“大多数管理专著都只是为经理们写的。而这本书却是为所有人写的，道理很简单：今天，所有人都生活在管理层制造的世界中。不管我们是否意识到，我们每一个人都把自己的幸福压在管理层的表现上。”[㊁]正如Tom DeMarco在他的专著《Slack》(懈怠)[㊂]所指出的一样，全世界的Dilbert[㊃]都放弃了责任。他们怪罪经理。Dilbert不理解帮助经理提高有效性是自己的责任。管理是事关企业从股东到最下层员工每一个人的任务。因此，本书针对的对象范围很广，包括关心软件公司是否能够很好运营的每个人。

㊀ Gartner集团和IBC估计全世界的软件开发人员为1 500万人左右。估计相关部门，例如项目、程序和生产管理部门聘用的员工为1500万人是合理的。

㊁ [Magretta 2002]《What Management is》，第2页。

㊂ [Demarco 2001]《Slack》.

㊃ Dilbert是United Feature Syndicate公司的注册商标，是一个由Scott Adams发明的卡通人物。这个人物在一个不能理解员工的老板手下吃尽了苦头。

本书内容针对关心改变软件开发工作实践以提高有效性和竞争力的每位读者。本书主要面向所有与软件有关企业内各个层次上的管理人员，以及立志在不远的将来担任单个产品或产品线高级经理的读者。本书也适合希望在与软件有关的公司中谋求管理职位的拥有硕士学位和MBA的学生阅读。大型软件开发公司的CEO、CFO、COO和CIO需要理解第一部分给出的范例和理论。Lou Gerstner在他的IBM论文集中指出，文化变革要想取得效果，就必须由最高管理层领导[Gerstner 2002]。如果变革由最高管理层领导，老板就必须采纳敏捷开发实践的最新思想模型以便作出决策，并理解以后开展的反传统的活动。

新管理实践的主角

本书将定义4种基本管理角色，并描述与每种角色的实践集。这些角色是：开发经理、程序经理、项目经理和产品经理，都在第8章“敏捷经理的新工作”中描述。开发经理角色的具体细节在第5章、第9章中定义。第10章定义程序经理，第7章定义项目经理，第16章定义产品经理。

本书的观点是，开发经理负责正在开发的软件系统，这种系统必须通过以细粒度单元生产活动为基础的测度进行管理。但是程序和项目必须在粗粒度层次上度量，以通过汇聚细粒度任务降低不确定性。第4章将解释如何缓冲可变性。产品经理必须定义作为有价值可交付产品的有意义的细粒度特性分组，即由程序和项目经理跟踪的粗粒度项。

总之，所有4种角色交互定义一种设置系统控制规则，但是内部具有高度委派性和自组织性的两层管理系统。成功的敏捷管理需要向开发人员放权的具有高度委派性的系统。敏捷管理的实质就是自愿组织的生产、经过策划的有价值组件组装内部的框架设计以及频繁交付的产品，以产生企业所需的投资回报。

敏捷成熟度模型

第11章介绍机构学习更好地使用敏捷方法逐渐成熟的概念。本书给出财务测度，关注真正的业务目标，研究管理层如何组织和报告软件生产系统的日常运作，以交付所需的财务收益。这种方法用来展示选用敏捷软件开发令人信服的理由。

在实践中，形成容错、敏捷、有学习能力的机构的途径是从内向外进行的。首先是工作实践，然后是可跟踪性，其次是测度，再次是学习，最后是财务测度和效益。“敏捷成熟度模型”描述这种发展过程。

阅读建议

第一部分是总体介绍，适合所有对经营软件开发公司获得更好结果感兴趣的读者阅读，适合从团队领导到CIO、CEO、CFO和GM的各级管理人员阅读。第一部分将介绍敏捷管理及其实践和理论，介绍如何将约束理论和精益生产方法作为一般实践运用到软件工程。对于很多读者来说，阅读第一部分已经足够。

第二部分针对的是需要在自己的公司中进行敏捷软件开发实践的读者，需要理解为什么要进行敏捷软件开发，以及怎样实现要达到的目标的读者。第19章和第20章将概述传统软件开发方法，帮助敏捷经理认清当前的现实情况并创建着手改进度量的基线。第二部分的其他部分介绍了一部分敏捷软件开发方法。这部分内容还很多，例如，这一部分要说明如何将具体的敏捷方法与第一部分给出的理论联系起来。第21章到第30章将讨论敏捷软件开发机构未来可能的发展，介绍如何对这些机构进行度量以展示经济效益的提高，并对照第一部分给出的理论比较FDD、极限编程、Scrum和快速应用开发，强调对这些方法的解释，而不是进行相互对比。应对比较留在第三部分中完成。

第三部分针对的是需要从多种方法中选出一种的读者，以及寻求理解敏捷方法并开拓敏捷软件开发未来管理的读者。这一部分将讨论当前不断变化的已有敏捷方法的异同，讨论如何针对不同的软件项目类型、规模和等级，恰当地选用这些方法。第三部分主要针对的是敏捷方法论学者，以及进一步研究敏捷软件开发未来的读者。

致　谢

一个人是不可能完成任何一本书的工作的。我有幸能够在合适的地点、合适的时间把自己认为有价值的材料写出来。对于我个人来说，如果我没有碰巧有7年多的时间处于一种特殊环境，能够“看到”约束理论和敏捷软件开发方法之间关系，是不能完成这本书的。我需要感谢很多人。

我从“特性驱动开发”创建小组那里学到很多东西，特别是Jeff De Luca、Peter Coad、Stephen Palmer、Philip Bradley和Paul Szego。如果没有与他们一起工作的经历，以及后来在世界其他地方管理FDD项目，是不能写出本书的。

此外，Mac Felsing指出了S曲线的重要性，以及开发经理应该怎样预测S曲线。Mike Watson提出了导致提出S曲线作用正式解释的重要见解。Jason Marshall帮助我“发现”累积流图（Cumulative Flow Diagram）是直观表示价值流的理想方法。Ken Ritchie不断鼓励我，并提出很多早期审阅意见。他的FDD和约束理论知识在开始撰写本书之前和之中，对我都有极大帮助。Daniel Vacanti通过每天上午与我的小组一起召开每天例行会议，帮助我理解了这种会议的意义。他还就第31章中的“蒸发云图”内容提出了自己的意见。Vahid Nabavizadeh和Chris Bock提供了项目跟踪证据，说明FDD特性汇聚于一个均值上，标准差很小。Martin Geddes在提出“易退化的需求”概念上作出很大贡献，推动我在第三部分提出了一些思想并进行讨论。如果没有Mac、Mike、Ken、Jason、Dan、Vahid、Chris和Martin提出的观点，在本书中我就没有多少好写的了。

我要感谢Peter Coad给了我撰写这种不同寻常的著作的机会，感谢Prentice Hall出版公司的Paul Petralia对我自始至终的鼓励和帮助，他们两人都坚信撰写本书是很值得做的工作。我还要特别感谢Prentice Hall出版公司的责任编辑Anne Garcia和Carlisle Communications公司的责任协调人Ann Imhof。

John Yuzdepski在2001年3月送给我一本《The Goal》时，对本书的内容结构表示了支持。我在去日本的飞机上看了这本书，又在回来的途中完成了36张演讲幻灯片，这些幻灯片成为后来向出版公司正式提出建议的基础。

正式审阅小组的Martin Geddes、Mike Cohn、Ken Ritchie、Luke Hohmann和Eli Schragenheim一直审阅最初比较难读的稿子，并进行指导、修改和鼓励，保证本书的最终稿能够以易懂的方式论述复杂的主题。

我要特别感谢所有非正式审阅人，他们无偿地拿出自己的时间提出本书修改意见。他们提出的一些很小的意见最终完善了本书的结构和表述。感谢Keith Mitchell、Alana Muller、Pujan Roka、Les Novell、Mary Poppendieck、John Resing、Frank Patrick、Keith Ray、R. A. Neale、Anthony Lauder、Hal Macomber、Alan Barnard、Lawrence Leach、Steven Holt、Ken Schwaber、