

# 《可伸缩敏捷开发》

## 图书基本信息

书名：《可伸缩敏捷开发》

13位ISBN编号：9787121082160

10位ISBN编号：7121082160

出版时间：2009-5

出版社：电子工业出版社

作者：兰芬维奥

页数：304

译者：李冬冬,冯雁,娄嘉鹏

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：[www.tushu111.com](http://www.tushu111.com)

## 前言

如果你刚刚涉入敏捷方法的领域，那么打开这本书时你可能会充满忧虑和怀疑，这不足为怪。关于敏捷方法似乎有一些奇怪的表述，例如，冲刺（sprint）、速度（velocity）、并列争球（scrum）（这是橄榄球比赛吗？）、极限编程（extreme programming）（我们是穿着滑雪板跳过悬崖吗？）、用户故事（user stories）、事迹和传奇（epics and sagas）（这是作家工作室吗？）等，还有一些怪异的社交形式，例如，结对编程（pair programming）、用户审查（user retrospectives）、聚集或者每日站立例会（huddles or daily stand-up meetings）（开发人员和测试人员在工作中互相拥抱吗？）等。敏捷团队似乎也消耗大量的彩色便签和4×6索引卡，他们在可以触及的任何墙面贴满了这些东西，整个事情似乎都不太对劲。这一切似乎是Scott Adams的Dilbert漫画的好素材！但是请不要误会：敏捷软件开发过程就是“穿越鸿沟”，这里用到了Jeffrey Moore创造的一个术语。我们今天已经远远不再是说些搞笑且令人讨厌的行话了，而是形成了有效的、有生产力的并且可扩展的开发方法。你必须向前迈进，否则就要落后。

另外，你可能看到或听说过，敏捷开发通常是反对“计划驱动”开发的，你可能认为这是一件非常混乱或者无法无天的事情，比起既系统又有计划的行动倒退了一大步。但是实际上，敏捷项目是经过精心策划的，只是他们的计划不同，并且该计划的修订和完善更为频繁。可能也有人告诉你，敏捷方法能够很好地支持小型团队（7~12人）、短期项目（2~9个月），但是它不能适用于大型的、长期的和分布在全球各地的软件开发项目。然而，请你不要合上这本书。随着世界各地的众多项目对这些界限的推动，以及敏捷在软件成果的高生产力和高质量方面取得的成功，一切都在快速地变化。

这就是Dean Leffingwell在本书中的重要贡献，他以XP、Scrum、Lean、DSDM、FDD、Unified Process等不同的敏捷过程之间的争论为基础，找到了这些方法之间的共性并作为基准，然后才进入他的主要目标，说明如何扩展这些敏捷方法并使其超越当前的适用范围。他不是在本已很长的名册中补充一个新的敏捷过程方法，相反，他利用一套新的实践方法扩展了敏捷方法，并把这些包括技术和管理在内的更高层次的实践方法融合集成到现有的已经建立的敏捷实践方法（名字很有趣）中。除了综合并且扩充了敏捷方法中所共有的最佳工程实践方法之外，他还描述了用于大型敏捷项目管理的方法：发布计划等主题，协调大规模的分布式团队，建立项目的企业价值观，处理大型的、长生命周期的开发过程，等等，我这里仅举这几例。

作者的工作不是学术性的，他不只是提出一些新的、大胆的猜测让你去尝试。他的建议根植于他多年积极的自身实践，这些实践来自于许多公司的众多项目，所涉及的行业范围极为广泛，从维持生命的医疗设备到软件工具，从游乐园骑乘设备到大型IT基础设施的应用。我知道这第一手资料是因为我曾有幸先后在Rational Software公司和Rally Software Development公司与Dean一起工作了大约10年。

# 《可伸缩敏捷开发》

## 内容概要

《可伸缩敏捷开发:企业级最佳实践》简体中文版由电子工业出版社和Pearson Education 培生教育出版亚洲有限公司合作出版。未经出版者预先书面许可，不得以任何方式复制或抄袭《可伸缩敏捷开发:企业级最佳实践》的任何部分。

# 《可伸缩敏捷开发》

## 作者简介

兰芬维奥 (Dean Leffingwell)，是一位知名的软件开发方法论者和作者，也是一个软件团队指导。他是8equisite公司的创始人和前CEO，也是Rational软件公司的前副总裁。在过去的五年里.他的工作角色是独立顾问，并担任Rally软件公司的顾问兼方法论者。Leffingwell先生致力于将敏捷方法应用于跨国公司分布式大型开发团队。Leffingwell先生也是《Managing Software Requirements》(Addison—Wesley公司，2003年出版)的第一作者。

## 书籍目录

### 第1部分 软件敏捷概述

#### 第1章 敏捷方法介绍

1.1 在软件经济中获得竞争优势软件开发方法与行业一起发展

1.2 走进敏捷方法

1.3 敏捷的规模

1.4 了解敏捷方法敏捷宣言

1.5 采用敏捷方法的趋势

1.6 软件敏捷的企业效益

1.6.1 提高生产力

1.6.2 提高质量

1.6.3 提升团队士气和工作满意度

1.6.4 更快地面市

1.7 XP、Scrum及RUP的简介

1.7.1 极限编程 (XP)

1.7.2 Scrum

1.7.3 Rational统一过程

1.8 小结

#### 第2章 为什么瀑布模型不适用

2.1 瀑布模型的问题

2.2 瀑布模型的假设

2.2.1 假设1：如果我们花时间来理解的话，存在着一套定义相当明确的需求

2.2.2 假设2：改变是小型且便于管理的

2.2.3 假设3：系统集成会顺利进行

2.2.4 假设4：我们完全可以按计划交付

2.3 利用敏捷方法来纠正行为

#### 第3章 XP的本质

3.1 什么是XP

3.2 有关XP的争议

3.3 有关XP的极限

3.4 XP的基本原则

3.5 XP的价值、原则及实践方法

3.5.1 XP的5个核心价值

3.5.2 基本原则

3.5.3 XP的13个关键实践技巧

3.5.4 结对编程的注释

3.6 XP的过程模型

3.7 XP方法的应用

阅读参考

#### 第4章 Scrum的本质

4.1 Scrum是什么

4.2 Scrum的角色

4.3 Scrum的哲学根基

4.4 Scrum的价值观、原则及实践方法

4.5 Scrum的关键实践方法

4.6 Scrum的基本原则：经验过程控制

4.7 Scrum的过程模型

4.8 对Scrum和组织的变更

## 4.9 方法的应用

### 阅读参考

## 第5章 RUP的本质

### 5.1 什么是RUP

### 5.2 RUP的关键特征

### 5.3 RUP的根源

#### 5.3.1 RUP的原理与实践

#### 5.3.2 迭代：RUP的基本原则

#### 5.3.3 架构驱动和用例中心化

#### 5.3.4 RUP开发过程模型

#### 5.3.5 时间轴

#### 5.3.6 规程轴

#### 5.3.7 RUP生命周期迭代类型

### 5.4 敏捷RUP变体

#### 5.4.1 开放统一过程（OpenUP）

#### 5.4.2 敏捷统一过程

### 5.5 方法的适用性

### 阅读参考

## 第6章 精益软件开发、DSDM和FDD

### 6.1 精益软件开发关于精益软件开发的阅读参考

### 6.2 动态系统开发方法

#### 6.2.1 背景

#### 6.2.2 DSDM的基本原则

#### 6.2.3 DSDM的核心实践

#### 6.2.4 访问DSDM

### 6.3 特征驱动开发FDD的最佳实践

## 第7章 敏捷的本质

### 7.1 敏捷正在改变什么

#### 7.1.1 成功的新措施

#### 7.1.2 不同的管理文化

#### 7.1.3 需求、架构和设计的不同方法

#### 7.1.4 修正编码和实现实践

#### 7.1.5 测试和质量保证实践的转变

#### 7.1.6 规划和进度安排的新方法

#### 7.1.7 最大的变化：范畴VS日期，优先考虑日期

### 7.2 敏捷的重要动力：短时间盒内的工作代码

### 7.3 总结

## 第8章 可伸缩敏捷的挑战

### 8.1 方法的明显障碍

#### 8.1.1 小团队规模

#### 8.1.2 客户是团队的一部分

#### 8.1.3 配置

#### 8.1.4 架构形成

#### 8.1.5 缺乏需求分析和规范文档

#### 8.1.6 文化和物理环境

### 8.2 企业的障碍

#### 8.2.1 过程和项目管理组织

#### 8.2.2 现有正式的策略和流程

#### 8.2.3 企业文化

- 8.2.4 固定日程、固定功能授权
- 8.2.5 开发部门和用户/客户代理团队之间的摩擦
- 8.2.6 通过纪律组织人力而不是生产线
- 8.2.7 高度分布
- 8.3 总结
- 第2部分 7种可伸缩的敏捷团队实践
- 第9章 定义/构建/测试模块团队
- 9.1 什么是定义/构建/测试模块团队
- 简单故事的生命周期
- 9.2 解除功能单元
- 9.3 敏捷模块团队的角色和职责
- 9.4 创建自组织、自管理的定义/构建/测试团队
- 9.4.1 团队中有合适的人
- 9.4.2 团队是被领导而不是被管理
- 9.4.3 团队了解任务
- 9.4.4 团队不断交流与合作
- 9.4.5 团队为结果负责
- 9.5 分布式的团队
- 第10章 计划和追踪两个级别
- 10.1 通用敏捷框架
- 10.1.1 定义迭代
- 10.1.2 剖析迭代
- 10.1.3 定义发布
- 10.1.4 剖析发布
- 10.1.5 计划发布
- 10.1.6 为发布分配需求
- 10.1.7 发布计划
- 10.2 小结：两个级别的计划
- 第11章 掌握迭代
- 11.1 迭代：敏捷的推动力
- 11.2 标准的两周迭代
- 11.3 计划和执行迭代
- 11.4 迭代计划
- 11.4.1 为迭代计划会做准备
- 11.4.2 参与者
- 11.4.3 迭代计划会议
- 11.4.4 结果：迭代计划
- 11.4.5 附加的迭代计划指导原则
- 11.4.6 分布式团队的迭代计划
- 11.5 迭代执行
- 11.5.1 承担职责
- 11.5.2 开发
- 11.5.3 交付故事
- 11.5.4 宣布故事完成
- 11.5.5 接收迭代
- 11.6 迭代追踪和调整
- 11.6.1 追踪每日站立例会
- 11.6.2 每日站立例会指导原则
- 11.6.3 追踪迭代状态

- 11.6.4 追踪剩余时间表
- 11.7 迭代节奏日历
- 第12章 更小、更频繁的发布
  - 12.1 小型发布的好处
  - 12.2 定义发布和制定发布的日程
    - 12.2.1 日程驱动发布
    - 12.2.2 最简单的模型：固定周期发布日期
    - 12.2.3 估算特征集
  - 12.3 计划发布
    - 12.3.1 参与者
    - 12.3.2 准备
    - 12.3.3 发布计划过程
    - 12.3.4 结果：发布计划
    - 12.3.5 附加的发布计划指导原则
  - 12.4 发布追踪
    - 12.4.1 为发布状态审查做准备
    - 12.4.2 发布状态审查会
    - 12.4.3 成果/文档
  - 12.5 发布路线图
  - 12.6 大规模敏捷的预览：全面的发布计划和追踪
    - 12.6.1 组织大规模的敏捷
    - 12.6.2 多团队发布计划
    - 12.6.3 发布追踪
- 第13章 并发测试
  - 13.1 敏捷测试介绍构建本质上可测试的系统
  - 13.2 敏捷测试原则
  - 13.3 单元测试
    - 13.3.1 迭代过程中的单元测试
    - 13.3.2 单元测试和测试驱动开发
  - 13.4 接收测试自动接收测试实例：FIT方法
  - 13.5 组件测试
  - 13.6 系统和性能测试
  - 13.7 小结：简述敏捷测试策略迭代和发布测试模式
- 第14章 持续集成
  - 14.1 什么是持续集成非持续集成：微观世界的问题
  - 14.2 持续集成
  - 14.3 实现持续集成的3个步骤
    - 14.3.1 源代码集成
    - 14.3.2 自动化构建管理
    - 14.3.3 自动构建验证测试
  - 14.4 什么是持续集成成功
- 第15章 定期反省和调整
  - 15.1 迭代回顾
    - 15.1.1 迭代回顾的形式
    - 15.1.2 定量评估
    - 15.1.3 定性评估
    - 15.1.4 要求行动
  - 15.2 发布回顾
    - 15.2.1 定量评估



15.2.2 定性评估

15.2.3 利用迭代回顾消除组织的障碍

第3部分 创建敏捷企业

第16章 有意识的架构

16.1 什么是软件架构

16.2 敏捷和架构

16.2.1 极限编程：架构形成

16.2.2 Scrum

16.2.3 在FDD中的架构

16.2.4 RUP：以架构为中心

16.3 关于重构和可伸缩系统

16.4 你在创建什么

16.5 用于企业级系统的敏捷架构方法基于组件的系统：组织遵从架构

16.6 创建架构跑道

16.6.1 架构的脆弱性和临时性本质

16.6.2 扩展架构跑道

16.6.3 通过产品记录重构

16.6.4 扩展架构跑道：与迭代同步

16.6.5 扩展架构跑道：一种精益的、基于拉的方法

第17章 伸缩时的精益需求：愿景、路线图、适时的细化

17.1 概述：需求金字塔

17.1.1 利益相关者的需要

17.1.2 解决方案的“特性”

17.1.3 软件需求

17.1.4 传统的需求方法

17.2 敏捷方法中需求的不同

17.2.1 在XP中的需求

17.2.2 Scrum、产品拥有者和产品记录

17.2.3 在RUP中的需求

17.3 一种可测量的、敏捷的需求方法：概要、路线图以及适时的细化

17.3.1 细化用户故事

17.3.2 细化用例

17.3.3 细化接收测试用例

17.4 小结

第18章 系统的系统及敏捷发布序列

18.1 敏捷组件发布日程

18.1.1 驱动敏捷序列的经验教训

18.1.2 敏捷发布序列的原则

18.2 敏捷发布序列

18.2.1 序列是同步的

18.2.2 序列是由愿景、主题和端到端用例驱动的

18.2.3 保持序列被跟踪并符合日程

18.2.4 测量过程和速度

18.2.5 观察系统级模式

18.2.6 管理相互依赖关系

18.3 发布序列审查

第19章 管理高度分布式开发

19.1 在规模上，所有的开发都是分布式开发

19.2 案例研究1PINGIDENTITY公司：

分布式定义/构建/测试组件团队

19.2.1 PingIdentity案例研究背景

19.2.2 学到的其他经验教训

19.3 案例研究2BMC软件公司：高度分布式的、大规模企业中的敏捷改革

19.3.1 背景

19.3.2 IMD应用敏捷

19.3.3 结果

19.3.4 从编码到编程：大范围采用敏捷

19.3.5 吸取的经验：贯穿大型组织的可伸缩敏捷实践

19.3.6 下一步骤：敏捷成功的第一年后

19.4 重视沟通

19.4.1 穿梭访问

19.4.2 通信基础设施

19.5 企业级敏捷的基础设施建设

19.5.1 源代码管理

19.5.2 网络基础设施

19.5.3 在早期迭代中提供基础设施

19.6 小结

第20章 对客户和操作的影响

20.1 敏捷方法对销售和市场的益处

20.2 对产品市场/产品管理的影响

20.3 更小、更频繁的发布更小、更频繁发布的挑战

20.4 优化敏捷发布过程

20.4.1 发布选择1：忽略敏捷

20.4.2 发布选择2：追求敏捷

20.4.3 发布选择3：通过从外部发布中分离出开发发布，进行优化

20.5 来自真正的销售和市场执行人员关于敏捷的真实挑战和错觉

第21章 组织变更

21.1 概述

21.2 为何敏捷需要改变组织

21.3 为Scrum和敏捷做准备

21.3.1 让软件过程和团队都“Scrumming”

21.3.2 让执行主管成为组织变更的Scrum主管

21.3.3 当心：变更是很困难的

21.4 消除软件生产率的障碍

21.5 给执行管理层的敏捷模型

21.5.1 支持采用敏捷

21.5.2 实践你宣扬的理论：把敏捷作为执行管理层实践

21.6 在大型组织中全面开展Scrum/敏捷

21.6.1 概观、评估和先导准备

21.6.2 先导项目

21.6.3 组织扩张

21.6.4 获得影响

21.6.5 度量、评估和调整

21.6.6 扩展和胜利

21.7 小结

第22章 度量业绩

22.1 敏捷测量：主要区别

22.2 测量团队业绩

22.2.1 敏捷项目度量

22.2.2 敏捷过程度量

22.2.3 评估成果

22.3 关于度量、“过程策略”和团队自评估

22.4 扩展至组织业绩：综合评价卡方法

22.4.1 效率

22.4.2 质量

22.4.3 价值交付

22.4.4 敏捷性

22.5 可伸缩的敏捷度量:为企业实现一个灵活的、自动化的和有意义的BSC

22.5.1 第1步：量化BSC矩阵元素

22.5.2 第2步：转为字母等级

22.5.3 第3步：聚合成产品线、业务单元和企业

结论：敏捷是可伸缩的

# 《可伸缩敏捷开发》

## 媒体关注与评论

尽管公司实施大型敏捷项目已经许多年了，但是“敏捷方法只适用于小型项目”这样的话依旧是新手所面临的普遍障碍，并且成为制定敏捷标准的战斗扫号。关于敏捷开发的资料很多，但是缺少一本关于使用敏捷方式开发大型项目细节的可靠且具有实用性的书。DeanLeffingwell的这本《可伸缩敏捷开发：企业级最佳实践》极好地填补了这一空白。它为架构、需求开发、多级发布计划及团队组织等大型项目问题提供了实际的指导。Leffingwell在本书中也为大型项目和大型组织向敏捷开发过渡提供了必要的指导。——JimHighsmith，主管，AgilePractice，CutterConsortium，（敏捷项目管理）

《AgileProjectManagement》的作者快速构建软件与交付可持续软件之间，以及保持对市场变化的响应与维持稳定程度之间存在着矛盾。DeanLeffingwell在《可伸缩敏捷开发：企业级最佳实践》中，介绍了如何实现这些方面之间的实际平衡。Leffingwell对问题的观察、对解决方案的建议及对结果的最佳实践的描述都来自于他的经验：他本人...直参与在敏捷实践当中，并且看到了效果。——GradyBooch，IBMFellow（IBM院士，即IBM最高级别的专家）

# 《可伸缩敏捷开发》

## 编辑推荐

“ 尽管公司实施大型敏捷项目已经许多年了，但是“敏捷方法只适用于小型项目”这样的话依旧是新手所面临的普遍障碍，并且成为制定敏捷标准的战斗口号。关于敏捷开发的资料很多，但是缺少一本关于使用敏捷方式开发大型项目细节的可靠且具有实用性的书。Dean Leffingwell 的这本书《可伸缩敏捷开发（Scaling Software Agility）》极好地填补了这一空白。它为架构、需求开发、多级发布计划以及团队组织等大型项目问题提供了有实际意义的指导。Leffingwell 在这本书中也为大型项目和大型组织向敏捷开发过渡提供了必要的指导。” ——Jim Highsmith，主管，Agile Practice，Cutter Consortium，敏捷项目管理（Agile Project Management）的作者

“快速构建软件与交付可持续软件之间，以及保持对市场变化的响应与维持稳定程度之间存在着矛盾。Dean Leffingwell 在其最新的著作《可伸缩敏捷开发（Scaling Software Agility）》中，介绍了如何实现这些方面之间的实际平衡。Leffingwell 对问题的观察、对解决方案的建议以及对结果的最佳实践的描述都来自于他的经验：他本人一直参与在敏捷实践当中，并且看到了效果。” ——Grady Booch，IBM Fellow（IBM院士，即IBM最高级别的专家）

# 《可伸缩敏捷开发》

## 精彩短评

- 1、前面关于Scrum的介绍以及各种敏捷实践的介绍比较全面。
- 2、读的是本书中文版，没去看英文的，懒得啃。

先说这本书的翻译，让我很觉得无语。懒得再写一遍评论，直接复制过来之前在敏捷中国论坛上的发言。

词汇翻译错误：

- CSM被翻译成“注册Scrum主管”
- PO被翻译成“产品主管”
- 还有一些其他词汇，完全不顾及这些词汇已有约定俗成的译法

也不消说整本书里的逻辑混乱，我自认也还算对敏捷了解有一定基础的人，都看得我头脑发昏，差点看得走火入魔，充斥着不通畅的中文。

更不消说此书的编辑不晓得干嘛去了，同一个词汇居然前后几章翻译得都不一样，user story都翻译不对，太让人失望。书中对wiki这个词没有翻译，直接留着英文，其索引列表里却赫然列着“wiki -- 维基”，哎。。

- 3、图书不错，又买了一本。
- 4、翻译的问题我一般都自动纠错了，呵呵呵
- 5、还可以，本书的精华内容在后半段
- 6、内容还算是比较充实的，值得一读。

# 《可伸缩敏捷开发》

## 版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:[www.tushu111.com](http://www.tushu111.com)