特性团队简介

作者： Craig Larman 与 Bas Vodde

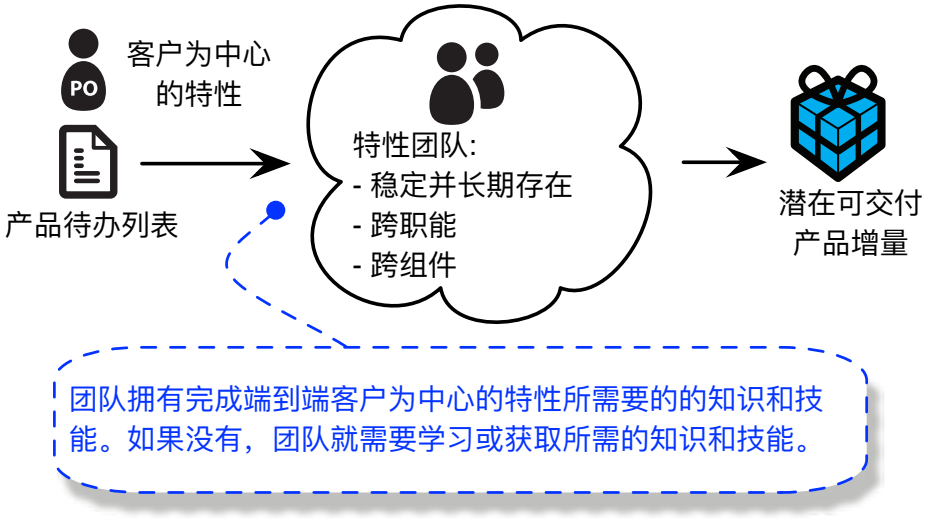
版本 1.3

特性团队与**需求领域**是规模化精益和敏捷开发的关键要素。《精益和敏捷开发大型应用指南》的特性团队和需求领域章节对它们进行了深入分析。这篇简短的论文总结了一些关键的思考点，你也可以在《精益和敏捷开发大型应用实战》中找到它们。

特性团队介绍

如图1所展示的，**特性团队**，是长期存在的、跨职能的、跨组件的，逐个完成端到端客户特性的团队。

图 *1.* 特性团队



特性团队的特征列举如下：

特性团队
• 长期存在 - 团队成员“凝聚”在一起以获得更高绩效；随时间的推移他们不断承担新特性的开发 • 跨职能并跨组件 • 理想情况下，坐在一起 • 跨越所有的组件和职能（分析、开发、测试……），共同工作在一个完整的以客户为中心的特性上 • 由通用型专家组成 • 在Scrum的定义中，通常 7 ± 2 人

采用特性团队时，应用现代工程实践，特别是持续集成至关重要。持续集成促进共享代码拥有权，这是多个团队同时工作在同一组件时必不可少的。

有个普遍存在的误解：每一位特性团队成员都需要了解整个系统。但并非如此，因为：

- 团队是作为一个整体，而非每一位成员个体，都需要具备实现整个以客户为中心的特性所需要的技能。这包含了组件知识以及职能所对应的技能，如测试、交互设计，或编程。但是在团队内，成员仍有其专注的领域……理想中是多个领域。
- 特性并非是随机分配给特性团队的，需要考虑到团队现有的知识和技能。

在一个特性团队组织内，当专业化成为一个限制…学习就会发生。

特性团队组织从专业化中获取速度上的优势，只要需求与团队技能相匹配。

但是当需求与团队技能不匹配时，学习就会“强制”发生，来打破过度专业化的限制。

特性团队平衡了专业化和灵活性。

表 1 和 图 2 展示了特性团队与传统组件团队的差异点。

表 *1.* 特性团队 *vs.* 组件团队

特性团队	组件团队
为交付最大客户价值而优化 [a]	为交付最多代码行数而优化
聚焦在高价值特性和系统生产力（价值产出）	聚焦在实现“简单”低价值特性以提高个体生产力
负责完整的以客户为中心的特性	负责部分的以客户为中心的特性
用现代的方式组织团队 [b] — 避免康威定律	用传统的方式组织团队 — 遵循康威定律 [c]
促成关注客户的、可见的并且更小的组织	导致产生“虚构”的工作及一个不断变大的组织
最小化团队间依赖以增加灵活性	团队间的依赖导致额外的计划 [d]
聚焦于多个专业领域	聚焦于单个专业领域
共享产品代码所有权	个人/团队代码所有权
共享团队责任	清晰化个体责任
支持迭代开发	导致“瀑布”开发
利用灵活性；持续并有广度地学习	利用已有的专业知识；学习新技能的水平低
需要熟练的工程实践—效果广泛可见	应用松散的工程实践—效果是局部的
为编写易于维护和测试的代码提供了动机	与人们的看法相反，通常导致组件内的低质量代码
看上去难以实现	看上去易于实现

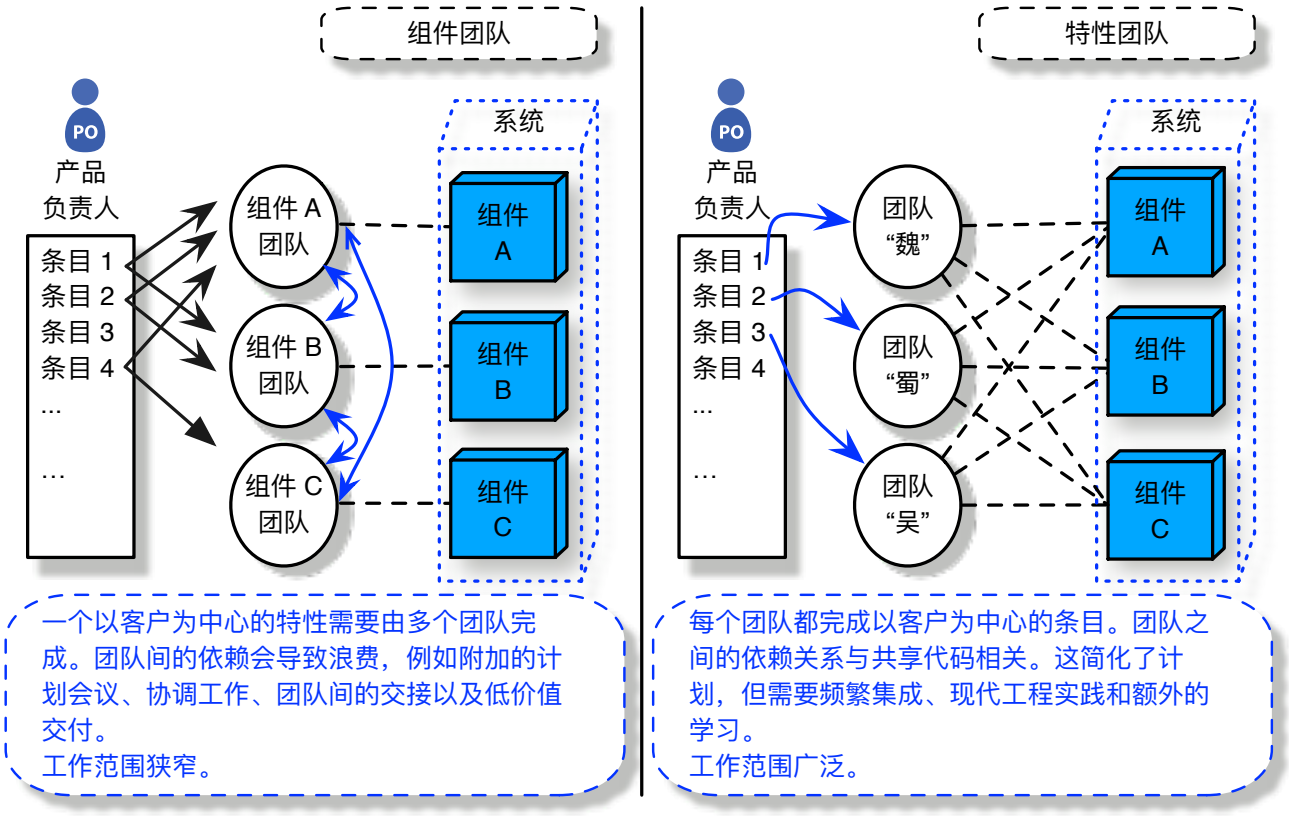
a. 不同的优化目标 *(*从局部角度来看*)* 通常使特性团队感到工作缓慢。

b. 所谓“现代”是相对的，因为特性团队在大规模开发领域已有很长历史，例如微软和爱立信。

c. 马尔文·康威在 *1968* 年注意到这种不合适的结构。事实上，他并不推荐使用。

d. 这种额外的计划常见于更多的“发布计划会议”或 “发布火车”和其它管理开销。

图 2. 特性 *vs.* 组件团队

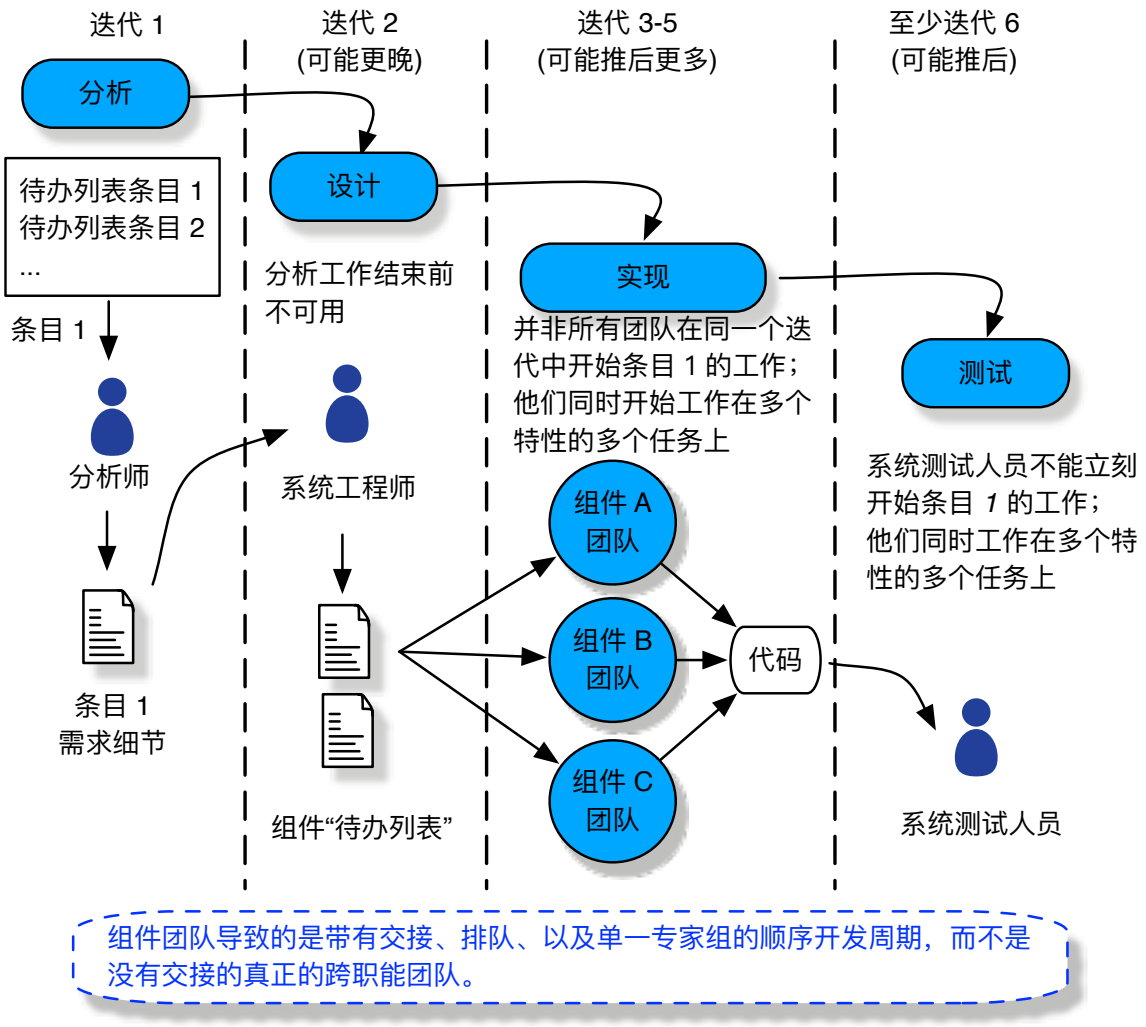


下表总结了特性团队与传统项目或特性群体的不同。

特性团队	特性群体或项目群体
稳定的团队，合作多年并工作在许多特性上	为了一个特性或者项目临时创建的群体
对于所有工作共享团队责任	基于各自的专长，个人对“他们”的部分负责
自管理团队	由项目经理控制
导致简单的单线组织（不是矩阵！）	导致配有资源池的矩阵组织
团队成员是专职的，100%分配给团队	由于专业化，成员在多个项目中兼职

大多数组件团队的缺点在《精益和敏捷开发大型应用指南》的“特性团队”一章中讨论了，图 3 总结了一些。

图 *3.* 组件团队的一些缺点



有时被忽略的是组件团队的结构加强了顺序开发（“瀑布”或V型），有许多不同大小工作包的排队，大量WIP，许多交接，以及不断增长的多任务和部分分派。

选择组件团队还是特性团队?

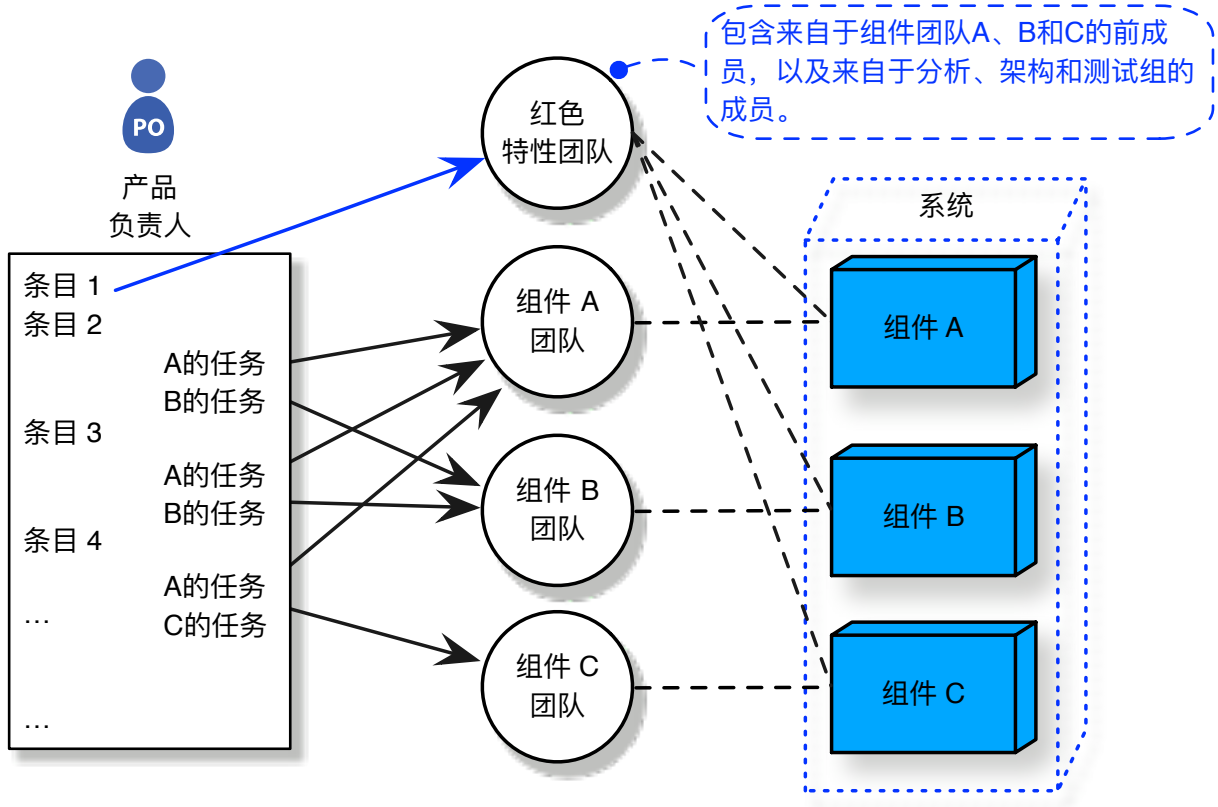
从价值交付和组织灵活性的角度来说，纯粹的特性团队组织是理想的。然而，价值和灵活性并不是组织设计的唯一标准，许多组织因而采用了一个混合模型 - 尤其是处于从组件团队到特型团队的转变中时。注意：混合模型具有两个世界的缺点并可能...很痛苦。

一个频繁被用来支持混合组织的理由是搭建基础设施，构建可重用的组件，或清理代码这些工作通常属于组件团队。但是这些活动也可以属于纯粹的特性团队 - 无须建立永久的组件团队。如何做到呢？就像对待一个以客户为中心的特性一样，把基础设施、可充用组件或清理代码的工作加到产品待办列表中然后交给现有的特性团队。该特性团队临时地（如同产品负责人期待的一样）工作在其上，完成后再去构建以客户为中心的特性。

转变到特性团队

不同的组织需要使用不同的从组件到特性团队的转变策略。我们有过许多成功过的策略但在不同的上下文中却失败了的经历。一个安全但是缓慢的转变策略是在现在的组件团队组织里先建立一个特性团队。当这个团队表现良好时，再成立第二个特性团队。就这样以该组织适合的速度持续进行下去。如图 4 所示。

图 *4.* 逐步从组件团队转为特性团队

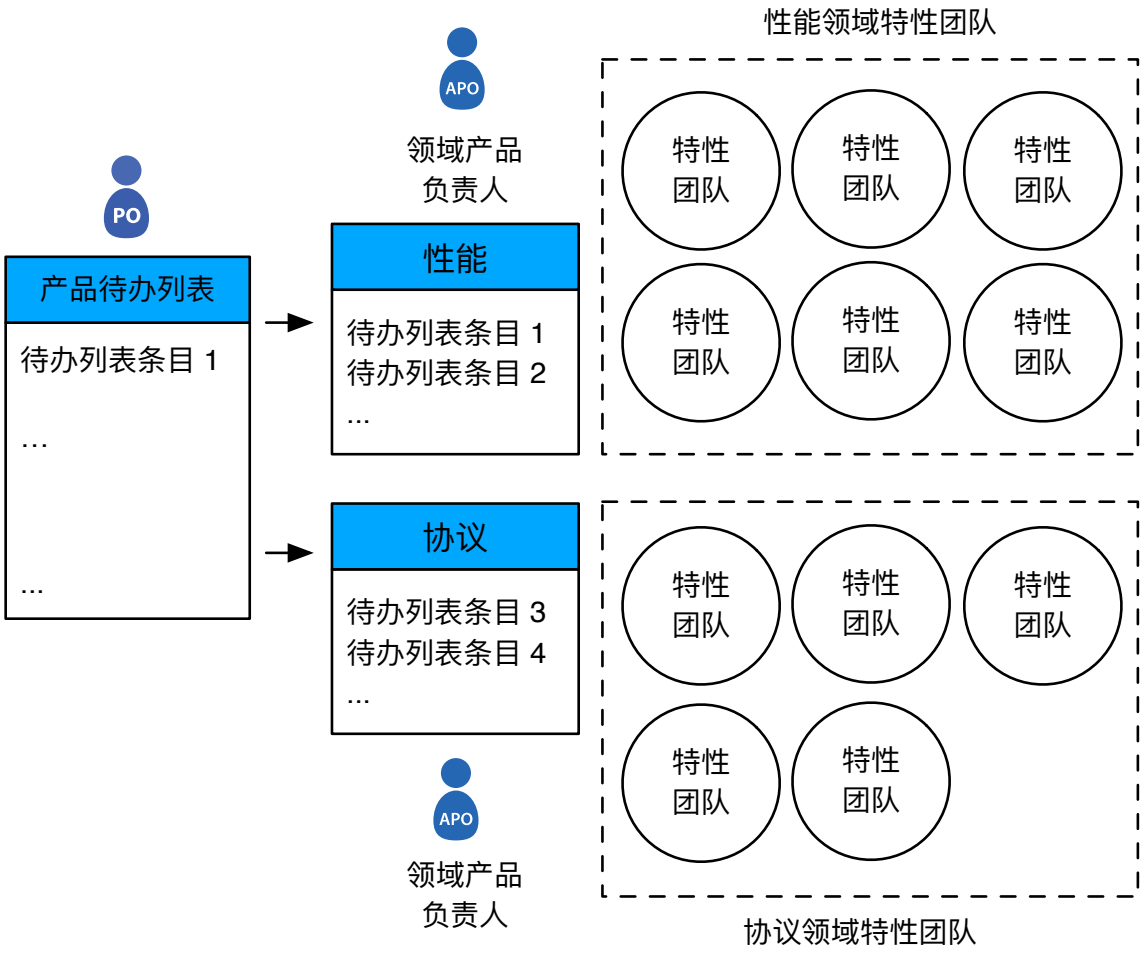


需求领域介绍

特性团队很方便扩展，但是当团队的数量超过十个，即大约一百人时，就需要额外的结构。需求领域提供了这个结构并补充了特性团队背后的概念。**需求领域**是对需求进行分类，从而形成产品待办列表的不同视图。

产品负责人将每一个在同一需求类别（即需求领域）的产品待办列表条目分组。然后他生成产品待办列表的不同视图，称为**领域待办列表**。领域待办列表由一个专门聚焦在产品某个部分（从客户的视角）的**领域产品负责人**进行排序。每个需求领域有几个特性团队在其中工作，如图5显示。

图 *5.* 需求领域



需求领域是规模化扩展了的特性团队。依据产品架构来扩展特性团队的规模化方式叫做**开发领域**。表 3 总结了两者的不同。

表 3. 需求领域 *vs.* 开发领域

需求领域	开发领域
围绕以客户为中心的需求来组织	围绕产品架构来组织
没有子系统代码所有权	子系统代码所有权
临时性；会在产品的生命周期中发生变化，但不是发生在每个迭代	趋向于在产品的生命周期中不变
聚焦客户，使用客户语言	聚焦架构，使用技术语言

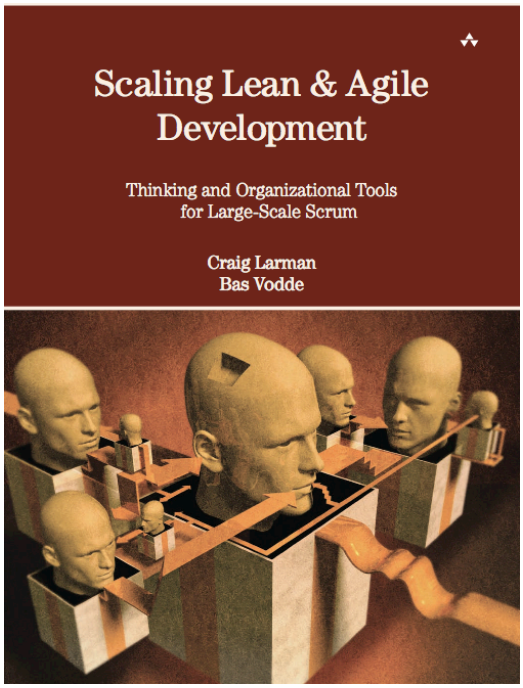
最后，领域产品负责人不同于为了帮助产品负责人而与一或两个团队一起工作的辅助型产品负责人。一个领域产品负责人拥有不同的责任和关注点，而且与（可能）至少四个而非仅仅一个团队一起工作。这避免了专注于一个团队活动的局部优化。

结论

特性团队是工作在完整的以客户为中心特性上的稳定的团队。这些团队解决了组件团队组织导致的局部优化和额外协调开销的问题。然而，特性团队也依旧需要挑战自我。

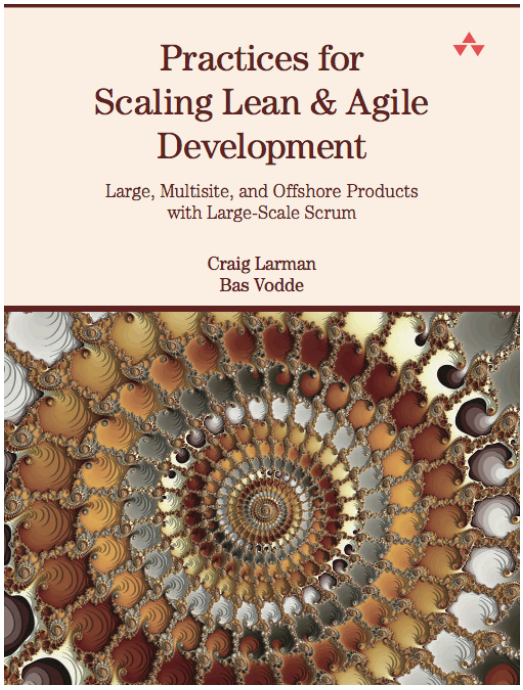
需求领域通过创建以客户为中心的视图在规模上扩展了特性团队概念，因而创建了一种允许特性团队扩展到任何大小的结构。

参考



章节:

- 简介
- 系统思考
- 精益
- 排队理论
- 虚假二分谬误
- 呈现敏捷状态
- 特性团队
- 团队
- 需求领域
- 组织
- 大规模Scrum



章节:

- 大规模Scrum
- 测试
- 产品管理
- 计划
- 协调
- 需求
- 设计
- 遗留代码
- 持续集成
- 检视与适应
- 多地点
- 离岸
- 合同