企业级敏捷攻略

目录

前言

在全球化经济带来的压力下，当今企业日益将软件生产能力作为核心竞争优势。无论开发的软件是为了管理生产和客户交付流程，还是为了改进日常活动的效率，软件确实已经触及到了商业的每个方面。

然而，很多 CXO 都发现他们的软件开发实践仍然和 20 世纪 80 年代没什么两样。尽管大量事实已经证明，预测型的、基于计划的瀑布式流程经常无法及时交付有实际价值的软件，还阻碍了公司及时响应日新月异的客户需求和市场环境，但他们仍然普遍沿用了这些流程，并且这样的状况一直没有得到改善。

现今的 IT 组织必须将过时应用程序重构成更灵活的、面向服务的架构，同时有效地协调全球分布式软件开发团队。显然，我们需要一种新的方法来管理和开发软件，从而保持竞争力。

为了应对这些挑战，人们开始广泛应用更敏捷、适应性更强的软件开发技术。运用这些技术，组织能够更快速地交付高价值的软件，Scrum 就是其中一个 被众多组织广泛使用、经过验证的流程。这份白皮书介绍了 CXO 或者其他高管如何在组织层面实施 Scrum，其中包括：如何将 Scrum 的应用进一步扩大， 如何用于大规模应用和多层团队，在实施过程中遇到的挑战以及得到的回报，为那些以软件为市场制胜法宝的企业应用 Scrum 提供了攻略。

管理分布式组织和有效地转换到面向服务的架构都需要一种新的软件开发流程

这是一份关于如何在企业中实施 Scrum 的“攻略”。这是一本手册而不是说明书，因为每个组织都是唯一的。在一个组织中实施 Scrum 的方式必然和 在另外一个组织中不同。由于实施中遇到的阻碍、需要改变的东西、变革的难度以及实施变革的人员不同，因此相关的时间表、优先级以及需要做的工作也会有所不同。

Scrum 与软件敏捷概述

从表面上看，Scrum 是一个非常简单的流程：一种只有少量相互影响的实践和规则，没有过度死板的条条框框，易于上手，并且几乎可以立即提高生产效率的软件管理方法。

Scrum 本质上专注于让整个组织开发出成功的产品。即使在技术不稳定的情况下，随着需求、架构和设计的涌现，它也能够帮助组织每隔一段时间就能交付有用的特性。你可以在项目初期引入 Scrum，也可以在项目中期引入。Scrum 已经将很多开发项目从困难中拯救出来。

Scrum 可行是因为它能够优化开发环境，减轻组织负担，并且通过快速交付同步响应市场需求。Scrum 基于现代流程控制理论，根据可用的资源、可接受的质量水平以及要求的发布日期，Scrum 开发出了尽可能最好的软件。

Scrum 的核心在于它以迭代增量式的流程开发产品或者管理工作，在每个迭代结束时，开发出潜在可交付的功能集。Scrum 具有以下这些属性：

- Scrum 是可以用于达成敏捷的工具。
- Scrum 是管理和控制开发工作的一种敏捷流程。
- Scrum 是对现有工程实践的一种包装。
- Scrum 是在需求快速变化的条件下以团队为基础的系统开发流程。
- Scrum 能够控制利益和需求的冲突所引起的混乱。
- Scrum 能够改善沟通并且最大程度地促进合作。
- Scrum 能够发现并移除在产品开发和交付过程中出现的任何障碍。
- Scrum 是一种最大程度地提高生产力的方式。
- Scrum 既适用于某个独立的项目也适用于整个组织，它能够管理多个互相影响的、组织人数超过一千人的产品和项目的开发。
- Scrum 能够让每个人都对自己的工作以及自己作出的贡献感到高兴，而且每个人都认识到自己尽其所能做到了最好。

Scrum 的实践细节并不在这份白皮书的讨论范围之内（详见 Schwaber 2004 和 Schwaber 2002），它的特征在于它拥有产品待办列表——该表根据优先级管理需求（如图 A3.1 所示）。产品负责人负责审批对产品待办列表的变更。经过大概为期 30 天的迭代才能实现需求，这样的迭代就叫做 Sprint。在这段时间内，只关注产品待办列表中最高优先级的需求。每个 Sprint 的目标就是交付一个潜 在可交付的产品增量。在 Sprint 进行期间，通过每日 Scrum 会议来观察需求的检查点。在会议上大家会讨论团队当前的进度以及活动状态，还会提出有可 能会阻碍个人或者团队工作的障碍。通过这个会议，Scrum Master 就能够检查现在距离 Sprint 的承诺目标还有多远，并在 Sprint 中途就如何确保成功地完成 Sprint 给出修正建议。图 A3.1 显示了 Scrum 的整体流程：

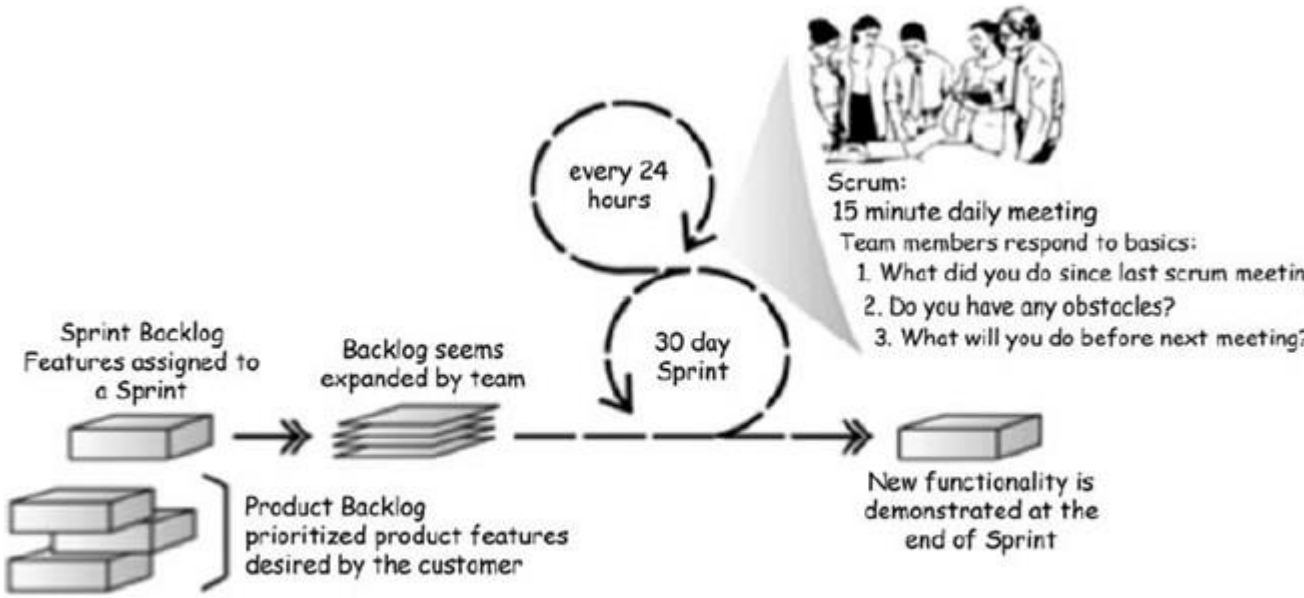

Figure A3.1 An Empirical Process Model for Scrum

图 A3.1 Scrum 的经验型流程模型

图字：Sprint Backlog Features assigned to a Sprint: Sprint 待办列表中分配到一个 Sprint 的特性；Backlog seems expaned by team: 被团队展开的待办列表；Product Backlog prioritized product features desired by the customer: 产品代表列表 客户需要的按照优先级排序的产品特性；every 24 hours: 每 24 小时；30 day Sprint: 为期 30 天的 Sprint；15 minute daily meeting: 15 分钟的每日例会；Team members respond to basics: 团队成员需要回答这些问题：；1. What did you do since last scrum meeting?: 1. 自从上一次会议之后做了什么？；2. Do you have any obstacles?: 2. 遇上什么困难了吗？；3. What will you do before next meeting?: 3. 在下一次会议之前你会干什么？；New functionality is demonstrated at the end of Sprint: 新的功能在 Sprint 结束时演示。

Scrum 的原则

当了解过 Scrum 的一些机制后，CXO 们应该要理解指引 Scrum 的一些关键原则：

- 坚信经验行的流程而非计划型流程才是最有效的软件开发流程。
- 坚信一旦组织的障碍移除了，一个自组织和自管理的团队自然而然能够交付比其他管理状态下更好的软件。
- 你能够在规定的时间和预算内交付最有价值的软件，但是肯定你无法精确地预测团队能够交付什么功能。

Scrum 断言只要能够认清这些关键原则，就能够将组织从众多妨碍有效软件开发的束缚中解救出来。然而，CXO 们还必须意识到，要采用这些关键原则同时也预示着对组织的潜在重大变革。由于这些原则组成了 Scrum 的根基，因此每项原则都值得我们进一步地探讨。

采用经验型流程还是计划型流程

Scrum 相信很多系统开发都基于一个不正确的思想基础——通过更多更好地计划就可以获得更可预测和更高质量的结果。Scrum 认识到应用程序开发 是一项不可预测并且异常复杂的过程（试想一下其中包括了几十万行人工编写的代码），其中的价值也只能够使用经验来度量。毕竟你所开发的应用程序从来没有被 别的团队在任何地方开发过，你的团队开发过同样的程序的可能性就更小了。因此，像手册或者按部就班的计划这样方法是无法有效地解决软件开发中固有的不可预 测性的问题的。

Scrum 将系统开发流程定义成一套宽松的实践，其中包含了一些已知可行的工具和技术，还有一个和客户或者产品负责人密切相关的被授权的团队。由于 其中很多实践是宽松的，因此需要加以控制——例如实施经常性的检查和掩饰——来管理风险和在每个时间点提供项目状态的实时经验证据。

是否使用 Scrum 需要的权衡非常简单：

是要利用 Scrum 然后每天都知道实际进展如何？

还是要

以为你正按照非常完善的计划按部就班地前进，却很久以后才发现你是错的？

消除障碍团队就可以专注工作

多年以来，一个公司的组织流程和软件开发实践通常都会在开发软件变得非常困难之前变得越来越繁琐。然而，当他们引入 Scum 之后，这些妨碍有效软件开发的“组织性”障碍将无所遁形，因为它们会降低团队按照 Scrum 的方式——快速迭代增量式——交付软件的能力。通过改变这些流程和实践，人们可能会发 现 CXO 或者执行负责人（executive champion）需要启动、驱动并监督一项主要的变革工程（稍后会进一步讨论）。 此外，在 Scrum 中团队是最重要的——因为他们是真正负责设计、开发和交付程序的人。因此，帮助他们消除障碍来提高他们的表现就能提升组织交付商业价值 给客户的表现。而管理层的使命是帮助团队消除障碍；而团队的使命就是完成他们承诺需要完成的 Sprint 待办列表项。

换而言之，在 Scrum 中团队是被授权并且负责交付产品的。团队必须要自我组织、自我管理并且自我达成 Sprint 的目标。对于很多组织来说，这样 就等于

要在组织里引起翻天覆地的变化。那种层次型技术型的管理思想会从本质上被 Scrum 淘汰。现在，只要产品负责人设定好目标和优先级，团队就可以自己 找出达成目标的方法，而不再一直需要有人告诉他们怎么做。

更少却更好的预测 vs 错误的自信

Scrum 开始于这样一个前提：开发软件是一项在瞬息万变的技术关键中施展的非常复杂的业务，没有人能够可靠地预测或者确定地计划出团队将会交付什么、何时交付以及交付的质量和成本如何。Scrum 相信团队可以做这样的预测，根据不同的风险通过讨论制定一个短期的计划，随着项目的进行还可以根据实际 情况进行调整。在这里需要达成的共识是：团队需要在给定的条件下交付最好的软件，遵循任何手册都不会改变“最好”的定义，而只会减缓团队对现实世界的复杂 度和不确定性的响应速度。

根据过往的经验，忽略以上这些原则会给组织带来众多问题：

- 管理层相信他们可以预测成本、交付时间表和能够交付的功能，然后根据这些预测制定计划。
- 开发人员和项目经理被迫生存在谎言中：他们假装能够可靠地计划、预测和交付。他们以一种方式工作，然而却必须假装正在使用另外一种方式。到最后，他们将会变得完全不受控制。
- 在系统马上将要交付的时候，通常需要进行无关紧要的或者重大的修改。导致这种问题的关键就在于过高的迭代成本让人们直到最后一刻才能看清团队正在开发的东西是否有用。 认清这些事实同样也是一项挑战——试想一下哪位经理会告诉高管他不清楚在给定的日期团队会交付什么东西？然而，这样做的好处是组织真正地能够自由地为其终端用户制造更好的产品，同时由于能够更迅速地这些产品，因此也毫无疑问地为业务创造更多竞争优势。

Scrum 和软件敏捷

从上个世纪 90 年代开始，Scrum 就已经开始被使用了，直到现在她已经被全球数千个项目采用。在这段期间，除了 Scrum 以外的一些新型迭代式流程也开始进入人们的视线。正如 Scrum 一样，这些流程都融合了新旧思想，但是她们都无一例外地强调：

- 开发团队和业务专家之间的紧密合作
- 面对面的交流比书面文档更有效率
- 经常性地交付新的可以部署的具有商业价值的软件
- 目标、流程和构件都需要具有透明性
- 封闭的自我组织团队
- 构件代码的方法以及团队应该能够持续地适应变化的需求

在 2001 年，包括 Scrum 领袖在内的各个流程的各路创始人和支持者进行了会晤，讨论了这些流程的共通之处。他们使用“敏捷”作为涵盖性术语，并且制定了《敏捷宣言》，其最重要的方面可以用以下对共同价值观的描述表达：

我们通过亲身体验去发现更好的软件开发方法，同时也在帮助其他人这么做。通过这样的过程我们得出了这样的价值观：

- 个人和互动优于流程和工具
- 工作的软件优于面面俱到的文档
- 客户协作优于合同谈判
- 响应变化优于遵循计划

也就是说，其中位于右边的内容虽然也有其价值，但是左边的内容最为重要。

《敏捷宣言》引起了大家的共鸣，也导致了成千上万新的敏捷项目的诞生。从这些项目中获得的成果和经验进一步地增强了那些采用了多种形式的敏捷实践的 流程。正如任何人类实践一样，有些项目成功了，有些失败了。然而，这些成功中得到的最令人振奋的消息是无论是商业人士还是技术人员都热爱他们的项目，他们 都认为这才是他们想要的软件开发方式，并且还得到了客户和终端用户的认可。这些成功的项目带来了更多的敏捷爱好者，正如一个成功的 Sprint，敏捷的巨 轮一直转动到今天。

为 Scrum 做准备

一旦 CXO 们对 Scrum 和敏捷能够带来的商业和文化上的好处有所了解，通常他们就会采取下一步行动看看这种开发流程能为他们的组织带来什么样的改进。

在 Scrum 诞生后的前 15 年间，很多 Scrum 的推行都是自下而上的。换而言之，如果一个项目团队采用了 Scrum 并且产生了不错的效果，另一个 团队也会接着尝试，然后很快 Scrum 的项目就遍布了整个组织。然而从最近开始，很多组织希望自上而下指令式地实施 Scrum，试图提高组织的响应速度和 生产效率。

由于 Scrum 非常依赖于团队授权和“让团队做决定”，因此自上而下地实施需要充分的考虑和准备，这就是这部分要讨论的内容。

用 Scrum 武装软件开发流程和整个组织

很多组织已经忍受了各种低效和障碍多年了，Scrum 却可以帮助他们快速找到这些问题并促使他们找到解决方案。解决这些问题需要费一些周章，幸运的是有来自 Scrum 项目的生产效率和价值的提升作为回报。

要实施 Scrum 组织必须要完成两项工作：首先，必须指引开发团队使用 Scrum 进行软件开发；其次，为团队消除优化创新和软件交付的障碍。第一项工作能够改进软件交付；第二项工作则消除在第一项工作中找到的提高投资回报率和生产效率的障碍。

这两项工作都充满挑战并且需要艰苦的付出，要完全实施 Scrum 可能需要长达 5 年。它们是组织变革的核心，因此无论管理层的要求有多强烈，也无论他们许下了什么承诺，这两项工作都不能草草了事。

Scrum 中每日和每月的检视和调整使得所有东西都变得可见——代码、流程还有公司内的障碍。使用 Scrum 的项目能够定期地找出各种需要被记录、评估、排列优先级和采取行动的障碍。

实施 Scrum 的快慢直接取决于以下这些方面：

- 在组织中需要变革的程度
- 改进软件开发和交付流程在组织内的迫切程度
- 领导力在组织内的有效性

CXO 作为组织级的 Scrum Master 在持续变革中所扮演的角色

在 Scrum 中，Scrum Master 的任务就是要保证 Scrum 团队以价值为生和并遵循 Scrum 的实践。Scrum Master 通过保证团队不会承诺超出他们在一个 Sprint 内

能够完成的工作量来保护他们，同时还不断地消除妨碍团队达成 Sprint 目标的障碍。

然而，当团队遇到组织层面的障碍的时候就需要 CXO 和高管们出马了。他们需要移除这些有可能会阻碍敏捷开发模型的成功的组织障碍。

CXO 是组织变革的 Scrum Master

组织级的 Scrum Master 的工作就是通过观察、确认和在组织内引起变革来消除各种障碍。也就是说，作为组织级的 Scrum Master，CXO 主要担任变革代理人，障碍的清单就是产品待办列表，而 CXO 的 Scrum 支持者则扮演这些障碍的“产品负责人”为这些障碍排列优先 级。组织内以团队为单位着手处理产品待办列表中的障碍，而交付物就是这些障碍被移除。组织变革的产品待办列表在试点项目期间就开始建立，只要在 Scrum 的检查和调整周期中能够找到需要改变的地方，这个列表就一直存在。

组织级的 Scrum Master 需要定期和所有 Scrum Master、产品负责人和支持者进行讨论，研究如何进一步挖掘组织级的变革产品待办列表。在 Sprint 中，团队负责驱动变革的发生。在 Sprint 评 审会议上，大家会评审已经发生的变革还有可以用于检测变革进度的指标。CXO 通过这种方式参与到所有组织级的持续变革，尤其是提升软件开发团队的生产力和 质量的流程中。

注意！变革没那么容易！

变革是一项困难的工作，同时也没有捷径可走。有些正在实施 Scrum 的组织有时会把实施过程中遇到的困难错误地归结为某个人的错误，他们以为只要没 有这些“错误”他们遇到的问题就会不翼而飞。这种错误的责备足以毁掉本该能够帮助组织开发更好的软件的整个 Scrum 的实施。当你遇到挫折、遭遇失败的时 候，你必须意识到这只是 Scrum 实施过程中必不可少的一部分，而且你还可以借此机会和大家共同探讨如何一起解决这些问题。

Scrum 的实施是没办法详细计划，也不可能遵照清单、步骤和表格按部就班地展开的。Scrum 只是一个能够找出所有在组织内会阻碍更好地开发软件 的障碍的简单框架。而管理和消除这些障碍正是实施 Scrum 的困难所在，并且由于每个组织都是不同的，因此每个组织都会遇上不同的问题。

没有人希望遇上挫折和困难，有些障碍已经在组织的思考和运作方式中根深蒂固，变得非常难以消除。就算事先计划得再好，也没有办法可以减少将要遇到的 困难，只会提醒每一个人：要成为世界级的竞争者就必须通过艰苦的奋斗。Scrum 需要高级管理层全身心地投入到障碍的分析和消除中，也要求主张采用 Scrum 的 CXO 成为组织变革的代理人领袖。

CXO 通过这种方式参与到所有组织级的持续变革，尤其是提升软件开发团队的生产力和质量的流程中。这不是一件容易的事，CXO 的领导力在其中起到了非常重要的作用。肯•施瓦伯在给某 CEO 的一段话中写到：

来自：肯•施瓦伯 给：XX，某某公司的 CEO

“Scrum 提供了非常诱人的可能结果——提升的生产效率、更好的工作环境、更强的竞争力以及更高的产品质量；然而，其实施难度却非常大。为了实施 Scrum 所需要作出的大量改变都是非常困难的。

尽管改变对开发人员及其客户（产品负责人）是很困难的，但是毕竟他们能够通过更好的职业满足感作为即时回报，从而帮助他们减缓变革带来的压 力和焦虑。然而，对于中层管理来说情况就不一样了，因为他们无法获得即时的回报。他们被要求将组织从使用传统流程转变成使用更精益的流程，在这个过程中他 们无法清晰地看到自己的个人目标。人们不禁会想：“我将会做些什么？我要怎么样才能融入转型后的组织呢？”。这样的问题尤其难以解决，然而由于中层管理是 组织转型后的中坚力量，因此这些问题不解决的话将是非常危险的。可想而知，由此可能引发的冲突和办公室政治将是非常可怕的。

根据过往我自上而下地实施 Scrum 的经验，我相信实施成败的关键就在于您——是否能够展望未来并且将愿景传达给管理层、是否能够耐心地带领他们完成变革、否能够让中层管理们认识到自己的价值并且使他们形成团队——这些都将影响到您是否能够成功变革并实现价值。”

Scrum 实施攻略

一旦你决定了要在组织里实施 Scrum，你就开始了一次漫长的旅程。你必须坚信所有的付出都会有回报，你会得到更有效的软件开发流程以及响应更快更有竞争力的公司。你还需要意识到大量的组织级变革将要到来。

当 CXO 谋划这项变革的时候，需要想到组织行为必然产生一系列能够产生实质变化的步骤。这些步骤包括：

- 寻求“传道者”和支持者
- 采用小幅的起始行动来试水
- 仔细分析成功和失败，然后继续一步步前进

下面这个部分将讲述一些如何在组织内推广 Scrum 的典型例子。这份“手册”列出了一些达成必要变革的方法的例子。

阶段 0 ——总览、评估和为试点做准备

这个阶段的目标就是为后面要实施的行动做准备，要准备的事情有：a）评估组织对敏捷的准备情况；b）为早期的参与者提供基本的培训；c）为初始项目建立产品待办列表。这个阶段的实施细节如下：

（i） 总览和评估

描述：为期两天的工作室，内容包括：

- Scrum 倾向性测试：向管理层透露 Scrum 会带来的变化，并帮助他们决定是否要进行。
- 介绍 Scrum：让组织对实施 Scrum 有总体认识，并向整个组织介绍 Scrum 的概念
- 评估组织的准备情况并制定下一步的行动
- 制定计划：找出潜在的试点项目，为培训制定时间表并为试点项目留出资源
- 和高级管理层一起晚餐评审下一步的行动

期限：2 天

支持：外部

（ii） 为试点做准备

组织已经为培训和启动第一个试点项目需要的结构调整做好准备。这里需要的活动有：

Scrum Master 培训

描述：为试点项目培训 Scrum Master

期限：2 天

支持：外部

产品负责人培训

描述：培训产品负责人懂得如何使用 Scrum 最大化投资回报率。

期限：2 天

支持：外部

团队（开发人员）培训

描述：培训所有团队成员，教他们如何成为跨职能自组织的团队，以及如何使用现代工程实践交付“完成”的功能增量。

期限：5 天

支持：外部

创建指标

描述：评审和修改用于检测 Scrum 在组织内的使用情况的指标，并且定义试点项目的价值。

期限：1 周

支持：外部

创建变革产品待办列表

描述：创建产品待办列表来跟踪和评估在试点项目中找到的障碍。这个列表将是推动组织变革的基础。

期限：1 天

支持：外部

阶段 1 ——试点项目

这个阶段的目标是通过在组织的实际项目中实施 Scrum 来展示敏捷软件开发的好处。一个或多个的试点项目在这个阶段开始展开。Scrum Master 和管理层需要密切地关注这些项目并尝试找出实施 Scrum 的障碍。一旦障碍被找出，就需要在适当的时候消除它们，或者将它们记录在组织级的改 变代表列表中，然后为稍后更好地处理进行归类。

（i） 试点项目

期限：3-6 个月

支持：外部 / 内部 Scrum Master

描述：运行 3 到 6 个迭代的试点项目，这样会交付一些功能增量，也能够找出一些优化软件开发的障碍。然后评估并调整计划，为障碍进行评估并排列优先级。

（ii） 回顾会议

期限：2 天

支持：外部 / 内部 Scrum Master

描述：评审试点项目、指标和障碍。评估有什么进展顺利，有什么可以改进；计算投资回报率；评估对包括各个部门之间的关系以及客户关系在内的业务运营的影响。

（iii） 重新计划

期限：1 天

支持：外部 / 内部 Scrum Master

描述：修改实施 Scrum 的总体计划，这个计划要一直保持高度概括，目的是让项目计划和组织变革计划由各自的产品待办列表驱动。

阶段 2 ——扩展到组织

借助成功的试点项目的经验，这个阶段的目标就是将 Scrum 的应用及其好处推广到开发部门的重要分部中。到目前为止，大家都对什么样的工程实践是有益的、发现了什么障碍以及需要什么样的深入培训有所了解了。例如，下面这些更广泛的培训项目到这个阶段也许有必要了：

- Scrum Master 培训：在将 Scrum 推广到更大范围应用之前，你必须增加 Scrum Master 的数量。这个时候应该从组织中选出拥有匹配技能的候选人，并且培训未来会带领 Scrum of Scrums 的 Scrum Master 学习驱动团队和收集指标数据。
- 产品负责人培训：当客户和产品经理在管理承诺和风险的时候，他们将会学习到如何优化投资回报率。通过充当产品负责人——利用管理进度来优化价值和避免产生意外的角色——他们将会学到这项本领。
- 开发人员培训：参与敏捷项目的开发人员需要学会如何运作自我组织的跨职能团队，并且学习如何利用指定的部分现代工程实践交付功能增量。
- Scrum/敏捷培训：一项成功的 Scrum 实施十分依赖于全民参与。为了到达这样的效果，可以对组织 30%到 50%的员工进行 2 到 4 小时的简介。

另外，你可能还需要其他活动来提升 Scrum 在组织中的可见度和接受程度。

- 信息发射源（Information radiator）：利用简单但强大的信息发射源来交流 Scrum 项目的状态，例如使用白板来展示任务（任务板）、产品和发布待办列表、工程与项目燃尽图。
- 阅读：通过向人们推荐一系列文章和书籍鼓励进一步的知识传播。
- CXO 带领的研讨会和简餐会议：变革领袖们应该经常坦率地向大家传达组织中正在发生什么的信息。像简餐会议或者匹萨聚会等非正式的会议都会为变革带来正面效果。
- 来自试点项目的图表/英雄事迹/反馈：试点项目的结果应该对每个人公开，这样才能增加整个组织中所有阶层对 Scrum 的讨论和投入度。

阶段 3 ——形成影响

试点项目已经证明了敏捷软件开发流程会带来实际价值，这个阶段的目标就是从根本上形成更大的影响，而这样的目标只能通过运作更多和更大的项目来达到。通过前面的这些阶段，组织已经获得足够的知识，因此能达到这一目标的成功率应该很高。到这个时候，组织中应该有四分之一的人参与到 Scrum 的实施 中。

这个时候实质的变革应该在开发部门内外同时发生：在部门内，变革最好由开发团队完成；在部门外，消除障碍应该由组织级的 Scrum Master 带领并由受影响的部门各自完成。

（i） 开发项目

期限：永远

支持：内部

描述：启动使用投资回报率检测的开发项目。

（ii） 变革工程

期限：大多数工作在前 1 到 2 年完成，其余的工作看具体需要

支持：内部

描述：各个部门的组织变革工程能够使障碍涌现和改变。

（iii） 评估并调整

期限：每个 Sprint

支持：外部 / 内部的 Scrum Master

描述：评审定性和定量的指标。当发生意外的时候加入额外的指标并评审如何获取指标所需的数据。

阶段 4 ——度量、评估并调整

这个阶段的目标是评估组织的实施进度并且创建更多指标作为进一步扩展实施的基础。CXO 们应该意识到，下面关于各种指标的讨论有可能争议和娱乐并存，因为在采用 Scrum 之前的一些传统指标（例如“文档完成率”）也许已经不复存在了。值得庆幸的是，Scrum 和敏捷实践实际上是可以解释和度量的，其参与者密切关注能够在流程和项目级别提供定性和定量反馈的各种指标。

在开始讨论之前，我们需要先将众多传统软件开发流程和 Scrum 以及敏捷流程加以区别：

*敏捷软件开发流程的主要指标就是：是否有可以工作的软件实际存在，其是否能够应用到实际的需求中。在 Scrum 中，这个关键指标**在每个 Sprint 结束**的时候通过演示的方式根据经验进行度量。*

这种主要对软件质量和生产率的度量是敏捷软件开发的本质，因此，只要用了 Scrum 你就不可能在自己毫不察觉的情况下远离目标。所有其他指标都是根据这个本质来制定的，同时它也是“更频繁地交付软件”的主导思想。

在 Scrum 实施这项事业的这个时间点，组织中的大部分部门和员工都已经加入了敏捷的行列。从初始项目每个 Sprint 中获得的信息是度量新实践和新流程对团队的有效性的主要指标，因此这些数据应该被公开并加以分析。

此外，这时正是制定引导你的组织如何实施 Scrum 的次级指标的最佳时机，其中以下两种指标可能会被应用到：

流程指标：主要表明团队和组织实施 Scrum 的有效性的各种定性指标，包括团队在管理产品待办列表的有效性，各种 Scrum 流程（如每日 Scrum 例会，Scrum 计划会议）的有效性等等。

项目指标：在项目层面额外的一组指标，用于度量某个特定的 Scrum 团队及其负责的服务、组建或者系统。这些指 标有可能包括一些传统指标，例如缺陷数、单元测试的覆盖率、自动化回归测试的覆盖率等等；也会包含一些 Scrum 的指标，例如在每个 Sprint 结束时能 够完成并演示的用户故事数。

关于质量和 Scrum

客户总是希望开发部门能够比现实可能更早交付特性。有些组织为了满足这样的需求而降低了产品质量、放弃了代码重构、减少了测试还有其他重要的工程实 践。这在 Scrum 中是不允许的，因为这些系统和产品是公司的财产而不是一次性交付的项目，需要不断地维护和客观地度量。由于交付期限压力而放弃了

质量的 组织最终只会开发出毫无设计可言的系统，这样的系统难以有效地维护和改进。结果就是组织不得不花费高额成本重写现有的代码。要避免这样的情况发生，必须限 制只有组织中的高层才能做降低质量的决定。

阶段 5 ——扩展并获得成功

随着前面介绍的一系列活动在组织中开展，你已经拥有了制定好的指标来指导和预测整个组织将来的进展，现在是时候将 Scrum 推广到整个组织了。在这个阶段需要做的是集中精力使 Scrum 的力量在组织内进一步扩张。

跟随着组织中大概 25%到 30%的先行者的步伐，其他人也陆续开始转向 Scrum。现有的实践需要进一步提炼，形成组织内统一的敏捷实践，然后在所 有团队中实施。只有现在才可以调整 Scrum 严格的敏捷实践来更好地满足组织的需要。可以邀请客户参与产品负责人或者 Scrum Master 的培训，使他们更多地参与到 Scrum 实施中来。这个阶段一直持续到所有团队都使用 Scrum，而 Scrum 的“检查和调整”机制将会一直保 持，帮助你进一步改进流程和实践。

这个时候，组织就可以收获 Scrum 实质上带来的生产率、业务和文化的回报了。

在开始讨论如何扩展 Scrum 到最大范围应用之前，我们需要先看看有哪些组织障碍会阻碍 Scrum 实践的有效进行。

实施 Scrum 的组织障碍

所有应用程序的开发，无论在任何组织里都是为了提升公司满足其业务任务的能力。然而，随着时间的推移这些组织并不总是朝着有利于软件开发团队的生产效率的 方向演变。实际上，有些组织的软件开发实践变得非常紊乱，尽管他们屡败屡战，然而由于组织的架构、政策和约束使变革无法有效进行。这个部分将讲述这些阻碍 的来源和性质，帮助 CXO 更好地迎接即将来临的工作。

用 Scrum 曝光所有障碍

Scrum 的特质就是不断要求快速地开发出高质量的软件，她需要持续地和终端用户沟通来保证软件的有效性，她利用“检查和调整”的机制快速暴露失效的实践和障碍。当使用 Scrum 作为在组织中实施和推广 Scrum 的流程的时候，这样的特质就更加明显了。

要提前找出所有需要的组织工作是不可能的

由于很多障碍已经在组织中根深蒂固，大家都对它们太熟悉了，因此你不可能提前找出所有障碍，直到你开始实施 Scrum 它们才会显现出来。实施的计划也会在实施的过程中随着变革的需要以及组织想要变革的意愿一起涌现出来。

为障碍定性

障碍主要出现在四个领域：

Scrum 流程本身：有什么阻碍了 Scrum 流程的运作？

人为实践：有什么实践阻碍了开发、分配、支持和使用产品来最大化每个参与者的参与度？

产品工程实践：有什么实践阻碍了对投资回报率的优化，或者从产品角度阻碍了最大化组织的职责？又有什么阻碍了对产品开发和交付的优化？

组织问题：有什么存在于组织中的体系问题明显超出了团队的控制范围，而又阻碍了团队更快地为其用户交付软件？

我们希望将组织级阻碍产品待办列表中的条目分类，因为解决不同种类的问题分别需要独特的技能。此外，这些条目应该按照其影响程度按优先级排序，同时应该给出谁是组织里解决这个问题的最佳人选的一些意见。

Scrum 的扩张

Scrum 和敏捷给业务带来的好处通常是通过在同一地点工作的小型跨职能团队达成的，最理想的团队由不多于 11 人组成（包括产品负责人、Scrum Master 和开发团队），每个 Scrum 团队都能够在没有外界的帮助下独立完成一个产品或者应用程序的设计、开发、测试和交付。

但事实上，使用 Scrum 获得的成功必然会促使其在更大的项目、系统和应用程序上使用，而必然需要更多团队，甚至需要使用分布式团队来开发和交付。 幸运的是，Scrum 已经被证明在拥有数百名开发人员的项目中同样适用，因此 Scrum 确实能够通过扩张来接受开发更大规模的企业软件的挑战。但是，扩张会带来一系列必须解决的问题，尤其是下列的几个：

1. 扩张团队结构：建立 Scrum 团队之上的团队
2. 为企业级敏捷准备工具和设施
3. 协调团队之上的团队

下面我们来看一些如何解决这些问题。

扩张团队结构：建立 Scrum 团队之上的团队

为了保持“少即是多”的原则，Scrum 只有很少量的规则。然而，这些规则中的大多数都是固定的和不能违反的。其中一条基本的规则就是团队由不多于 11 人组成，而且这些团队成员应该尽可能地被安排在同一个工作区域内。这种方式是最有效也是生产效率最高的，因为这样可以使团队成员之间的经常性非正式交 流更加方便、加强团队合作精神、使每天一起工作并互相认识的团队成员可以为了共同的 Sprint 目标一起努力。此外，如果团队人数超出 8 到 10 人的话，像 Sprint 计划会议和每日 Scrum 例会这些强制的 Scrum 实践会轻易地瓦解。组建分布式团队和过大团队应该需要仔细权衡。

即使要将 Scrum 的应用范围增大（如图 A3.2 所示），也不能放弃这条原则。因此，如果要将 Scrum 推广到 300 人的应用中，那么就需要组建大 概 30 个 Scrum 团队。每个团队需要满足前面所述的条件，并且能够在每个 Sprint 开发出潜在可交付的功能模块。对很多组织来说，这就需要围绕着产品 的特性、服务、组建或者子系统重新组建团队，而不是根据每个人的职能（例如开发人员、测试人员等等）来划分。当我们在讨论组织障碍的时候已经发现随着团队人数的增多，团队组建的难度也会增大。

组织架构跟随产品架构

我们不理解每个团队如何相对整体地交付终端用户可用的功能，就无法容易地组建 Scrum 团队。这就要求我们将程序的架构分解成独立的组件或者子系 统，使每个部分都可以独立地交付商业价值 ①。这种组织架构调整可以在早期两个

Sprint 之间由先实行 Scrum 的团队完成，这个方法在 Scrum 扩张到更大的项目中时尤其有效。具体方法如下， 先行的团队一边交付客户需要的价值，一边对系统进行调整使正在进行 Scrum 培训的新团队可以在加入后负责独立的模块。这样，当每个新团队组建好的时候， 其在系统中的角色就清晰可见了，如图 A3.2 所示。

①这种架构的分享和沟通在开发面向服务的架构（SOA）的软件时的确是一项挑战，因为现在团队间需要更多的交流，而现存的组织架构很可能是按照原来各自为政的程序模块来设立的，不需要太多的相互协作就能向用户交付足够灵活的服务。

协调团队上的团队

当然，团队间的沟通和协作在团队数量巨大时是一项重大的挑战，同样也预示着系统级的每日和每月的检查实践在团队中运用时可能会出现大量问题。过往 Scrum 扩张的经验演变成了一套用于协调不同团队以及处理在多团队项目中进行 Sprint 计划会议、版本计划会议和追踪系统级集成和测试的有用实践。

每日沟通：Scrum of Scrums

和普通的 Scrum 实践一样，大型的分布式团队同样需要每天在每日 Scrum 例会中交流，而这种交流就叫做每日 Scrum of Scrums 例会，目的是为了让各个团队协调他们之间的工作。在这个会议中，来自每个组件团队的代表和单个团队的每日 Scrum 例会一样，需要回答同样的 三个问题：

1. 我的团队昨天为达成 Sprint 目标做了什么？
2. 我的团队今天会做什么？
3. 有什么障碍出现了使我的团队无法完成 Sprint 的承诺？

理想上，这个会议应该在每个团队的每日 Scrum 例会后马上举行。当团队分布在不同地点时，通常会采用电话会议的形式进行，而时间则应该选择 Scrum of Scrums 团队中能够参与人数最多的时候。

系统级的版本计划和追踪

你可能以为像图 A3.2 显示的那样，将组织按照特性、服务或者子系统分割是一件很简单的事，只要给团队授权，然后美妙的系统集成就自然地发生了—— 经验告诉我们那是不可能的。就算每个团队都获得充分授权，能够达成每个 Sprint 的目标，还能很好地和其他团队的子系统集成，事实上仍存在诸多挑战—— 如何能够将系统作为一个整体构建。如何在所有子系统中实施 ant ②测试，如何让所有子系统协同工作满足客户的边界需求的同时又不影响系统整体的质量、性能和可靠性的需求——这些都是需要解决的问题。因此，现在我们要求 只有在所有团队的模块集成和测试的工作完成之后，一个单独团队的工作才能被认定为完成。如图 A3.3 所示。

②译者注：一种软件构建工具（http://ant.apache.org/）。

为了迎接这些挑战，很多团队都在系统级别增加了技术领袖的角色。架构师、团队领袖、产品经理和质量保证部门经常会组成额外的 Scrum 团队负责系统 级别的事务。此外，他们还可以在系统级别实施 Scrum 来为每个 Sprint 设定目标，并创建待办列表项来追踪强制进行的系统集成、系统级的演示、质量检查、预发布和其它里程碑，从而保证系统的开发走在正轨。只要这样做，你就会看到组织呈现出图 A3.3 的状态。

为企业级敏捷准备工具和设施

尽管拥有这种水平的组织结构和组织协作，大项目和分布式团队中的成员仍然可能觉得他们缺乏内部和团队之间的合作，以及在快速、完整测试的迭代中可靠 地交付软件所需的项目透明性。虽然 Scrum 为软件开发的项目管理提供了已经被证明可行的框架，但是她并没有规定特定的软件工程实践，也没有推荐任何工具 来支持 Scrum 的流程。Scrum 在这方面的哲学是：保持简单，让团队做决定。当众多组织还在为现代工程实践挣扎时，Scrum.org 已经引入了 Scrum 开发者（Scrum Developer）项目和围绕现代应用程序生命周期管理（ALM）工具的培训。

实际上，对于理想的少于十人并且都在同一地点工作的团队来说，用于 Sprint 计划和追踪特性、任务和团队进度的最佳工具通常是一份经过定制的表 格，由 Scrum Master 负责定制和维护。而一些和需求、测试用例和缺陷相关的工程工件也可以用类似轻量级的解决方法，例如利用索引卡片、白板或者 团队的 wiki。

（i） 人员和沟通

要将 Scrum 推广到分布式团队和带层次结构的团队会带来特别的挑战，多个团队之间如何协同工作实现共同的需求、跟踪特性状态、找出障碍是其中最主要的问题。这个时候，*“需要一种机制来让各个团队经常性地分享他们的进度。同时，还需要更仔细地划分产品和技术架构，才能更清晰地将工作分配到每个团队。”*

扩张 Scrum 会为沟通带来特殊的挑战。

传统的项目管理工具可能能够胜任显示理想化的任务的开始和结束日期，提供在漫长的瀑布式项目中也许没有实际意义的关键路径分析。而在短周期的迭代中，整个团队都专注在使优先级最高的少数几个特性满足验收标准，这些以计划驱动的管理分析就不管用了。在小型中，只有一个人负责维护独立的任务数据库，这 个数据库却和团队日常用于计划和实施的工件（如用户故事和测试）没有直接联系，而大型的项目则需要能够反映各个团队在开发、测试和集成产品待办列表中特性 的实时工作状态的协作环境。为了尽量和本地团队保持一致，这个敏捷项目管理环境必须能让每个人都能快速地查看和更新某个特性在

生命周期中所处的位置、要完 成这个特性还需要多少工作量、是否有什么障碍出现。

除了需要新的方法进行迭代计划和追踪外，大型项目对用于定义、组织、分享系统工件也有新的要求——需求、验收测试、缺陷的所有信息在整个 Sprint 期间必须是非常容易查阅，而不应该让这些信息淹没在和团队承诺不相干的信息里。实际上，在快速的迭代中这些工件之间的关系正式团队最关心的问 题。因为团队在每个 Sprint 都会开发出很多可工作并经过测试的代码，所以他们必须非常清楚这些工程工件之间的关系，也必须能够随时查阅它们的状态。

(ii) 建设工具和设施

作为开发人员，自然希望用到的工件能够组织得更好，能够自动化一些 Scrum 的流程，这样就可以把精力集中在软件开发本身上。一些团队尤其希望有设施能够使以下软件生命周期中的活动和工件更便捷：

- **待办列表管理：**随着系统复杂度的增加，团队会希望能够有更好的管理特性列表、功能性和非功能性需求、用 例、用户故事以及它们的优先级、估算、状态和负责人的方法。随着 Scrum 应用到更大型的项目，这些工件的数目会增长到数千个，因此一个能够按照系统或者 子系统组织、支持和管理这些工件的方法变得至关重要。
- **项目报表：** Scrum 避免了使用传统的类瀑布式的项目计划，但是其具有策略性的日常项目管理的本质是坚定不移的。因此，团队需要一个简单的方法来让每个人都能够很容 易地输入他们对任务估算、任务的状态、剩余的工作量，这样燃尽图就能够自动生成并且持续有效。此外，还需要在待办列表项在变换生命周期中位置时向团队发出 信号。高级员工需要从观察每个团队各自的迭代和发布计划才能从全局上评估整个项目所处的状态。
- **及时的需求分析：**很多小型的 Scrum 项目通过非正式的需求管理机制获得了成功，例如团队直接和产品负 责人讨论需求。然而，随着项目的复杂度和重要性的增加，需要更深层次和更多方式的需求讨论以及需求版本管理的可能性也会增加。例如，将对多个团队有影响的 接口记录在文档中是非常重要的。对接口的更改或者增加跨团队的新特性都有可能对整个项目造成重大影响。这些需求都应该在需要时，也就是说只有在要实现这些 特性的 Sprint 开始之前才进一步分析。要解决这个问题，团队可能需要一种能够更丰富地展现需求、整理后待评审的文档以及自动化变更通知的方法。
- **先行的测试：**因为每个 Sprint 都会交付潜在可交付代码作为下一个 Sprint 的基线，先行的测试用 例开发和自动化测试使团队能够跟上 Scrum 快速迭代的步伐。如果能使用工具将需求或者用户故事卡片直接生成测试用例，就可以提高开发的速度并提供证明通 过特性的验收所需的固有可追溯性。需要注意的是，对成百上千持续增长的回归测试的持续管理很可能是决定 Sprint 的成功和速度的关键因素。
- **发布计划：** Scrum 是“追求近期可能性的艺术”，而不是去尝试预测 6 到 12 个 Sprint 之后能够交付什么。这种哲学是团队中思想上的突破，

因为它可以让团队专注 在当前 30 天的事情上，从而更可靠地开发出可工作的软件。但是随着团队的增大增多，为将来的 Sprint 做额外的严谨的分析有助于避免日后对架构做过多重 构。虽然在敏捷中高度推荐重构，但是随着程序的范围和部署的数量增多，过多的重构就变得不现实了。为了更好地看清楚系统架构将来可能演变的道路，进行额外 的发布计划也是合适的。因此，Sprint 计划会议中可以“向后看”几个 Sprint，也可以进行一些假设性的计划，这样能够帮助团队更好地权衡产品待办 列表并向资助者传达一个更合理的愿景和产品路线图。

另外，团队还可能希望将这些设施和工具都集中到一个中央仓库中，使全球每个角落的每个团队成员都可以全天候地访问，还可以显示即时的工程和项目状态，并且当项目发生重要变更是能够自动发送变更通知。

(iii) 在 Sprint 中改善设施

在 Scrum 中，这种程度的设施建设是不可能由一个负责设施的团队提前一次准备好的。相反，应该由 Scrum 团队自己根据从过去的 Sprint 中吸 取的教训找出需要购买或者建造哪些设施来帮助他们解决遇到的问题。还需要说明的是设施的建设应该是在 Sprint 中进行的，因此团队需要为建设设施在产品 待办列表中添加相应的列表项，如图 A3.4 所示。当然，和客户相关的功能优先级应该更高，但是有经验的团队能够认识到在产品的范围和团队数量增加时，他们 必须在持续地安排设施相关的工作地同时保持他们的速率和生产效率。

总结

Scrum 是一项已经被证明确实可行的软件开发实践，她可以迅速地提高生产效率、交付速度和软件开发团队的质量。

难道会有什么开发组织是无法从成功的 Scrum 实施中体会到下面这些好处的吗？

- 减少的开发周期次数
- 对终端用户输出更高的价值
- 更好的质量
- 更低的开发风险
- 更高的用户满意度
- 更高的员工士气

实施 Scrum 虽然看似简单，但是实际上却通常需要重大的组织性变革来消除有效地开发和交付的障碍。作为变革代理领袖的 CXO 或者其他高管应该肩负 起消除障碍的重任，他们是否能够称职是实施成败的关键。当然，这并不是一件容易的事，而许诺要用 Scrum 改善软件产出的 CXO，应该作为先行者迈出保证 企业正在从更快更好地交付软件中获得商业利益的正确道路上的第一步。

此外，Scrum 在大规模的企业软件开发中也是相当高效的，她能够使几百位开发人员同时开发一个应用程序。Scrum 的扩张预示着在设施和工具上将遇到新的挑战，这些问题都需要由团队自己解决，然而只要克服这些挑战就很有可能在市场竞争中获得竞争对手无法想象的优势。

翻译

这份《企业级敏捷攻略》是由李麟德(Derek Li)和王军(Jim Wang)翻译的,原文来自于 Ken Schwaber 和 Jeff Sutherland 撰写的 《Software in 30 Days》一书。