异步图书
www.epubit.com.cn

Pearson

超越需求

敏捷思维模式下的分析

[美] Kent J. McDonald 著　霍金健 译

Beyond Requirements

Analysis with an Agile Mindset

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

第一部分　理　　念

第 1 章

指导原则

1.1 简介

敏捷方法对我最大的影响也许正是这样一种理念，即团队做事方法应基于价值观和原则而不是基于实践。实践往往对情境非常敏感——用于 Web 应用程序的实践与用于商业佣金系统的实践不同，而用于商业佣金系统的实践与用于大型机的工资系统的实践也不同。在这 3 种情况下采用同样的实践就是制造麻烦。而价值观和原则往往更广泛适用。“敏捷软件开发宣言”和“敏捷宣言背后的原则”通常被认为是敏捷价值观的代表。本章将论述我对知识工作方法的核心思想。

下面这些基于敏捷原则的指导原则，描述了任何**自发活动**（initiative）的理想特征：

- 交付价值；
- 合作；
- 迭代；
- 简化；
- 考虑情境；
- 明智决策；
- 反思与适应。

1.2 交付价值

价值很难界定。在很多方面，这就像美和质量一样：“当我看到它的时候就会知道它。”对一个人很有价值、很重要，但对其他人也许一点儿也不重要。和其他

很多事情一样，交付价值也是和具体情境非常相关的。对我而言，评估价值就像试图理解它是否是值得的，而这里的“它”可能指的是承担或继续一项自发行动或交付一个具体的特性。

当你所交付的（产出）满足了利益相关者的需求（提供了预期的结果）时，你的团队就是在交付价值（deliver value）。交付价值也为**项目**（project）决策和衡量成功提供了不同的依据。你仍然需要关注成本、时间和范围的**三重限制**（triple constrains），但**范围**（scope）的定义是以是否取得了预期的成果为基础的，而不是以团队交付的产出量为基础的。你会发现，你的团队需要采取行动，寻求以最小的产出取得最大的结果，同时确保满足成本和时间限制。

范围的定义由产出变为结果，这使对是否交付了事先约定的范围进行量化更加困难。这正是**目的**（goal）、**目标**（objective）和**决策过滤器**（decision filter，这一技术将在第13章详细介绍）派上用场的地方。目的、目标和决策过滤器既提供了一种清晰的方法来描述你所寻求的结果，也提供了一种方法来判断你何时取得了那个结果。把打算满足的范围定义为这些方面，也为团队在满足这个范围时提供了更多灵活性，并且能够避免团队制造不需要的产出，这也为能够满足项目的成本和时间限制增加了机会。

项目会积累很多潜在的特性，它们在当时看起来是不错的想法，但最后对于项目的终极目标可能无足轻重。交付价值最简单的方法之一就是，把不能对预期结果产生直接贡献的产出砍掉。这些产出包括很酷且也是某个利益相关者声称想要的，但对项目要解决的问题无所助益的**功能**（functionality）。我们把特性限定为具有真正可识别价值的功能，并且我们只想关注用户真正使用的极少数功能，这就回到了我们的前提 ——用户基于他们的行为感知有价值的特性。

正如 Gojko Adzic 在审阅本书初稿时所提醒的，把重点放在能驱动达成预期结果的事情上。当用户可能使用的功能或者利益相关者提出的需求和预期结果毫无关系时，不为其分心是个不错的主意。

会议投稿系统（将在第7章介绍）就是一个专注结果而非产出的案例。这个项目的主要目标是支持 Agile2013 大会的会议投稿流程。我们确实建立了一个该系统要包含的特性的待办列表（项目的产出），但这一待办列表更多的是进行估算和计划的起点，而不是必须遵守的范围定义。在项目进行的过程中，我们根据我们的发现在待办列表中增加了几个特性，也推迟了几个特性，因为在时间固定的指导原则下，这几个特性对于达成业务目标并非绝对必要。

一个重要的附加说明需要记住，有些用来满足法规监管或安全要求的产出，

诸如系统文档和文书工作，对预期结果仍然是有贡献的，否则如果没有其他原因而未能完成这些产出，**组织**（organization）要么无法得到批准交付解决方案，要么可能招致惩罚。

我将在第 5 章讨论更多关于交付价值的想法。

1.3 合作

合作（collaboration）有两个方面的含义。一个方面是团队成员尽可能高效协同工作的能力。这方面可能是在所有项目中——实际上在所有团队工作中——都具有的，这方面总能得到提升。实际操作中，这意味着从团队环境中移除所有阻止有效沟通的障碍，并且有团队成员可以高效地**引导**（facilitate）团队合作。

合作的另一个微妙但同样重要的方面是，实际上完成项目核心工作的团队成员也是做计划并报告进度的人。这是从项目的传统视角看到的转变，传统项目中这些类型的任务由项目负责人完成。团队成员是最熟悉这项工作的人，同时他们处在最合适的位置，能够确定要完成什么工作。因此，团队成员应该自愿完成不同的工作项，而不是由别人进行指派。

无论采用任何**方法论**（methodology）和具体方法，这一指导原则都应该被用在每个项目中。然而这一原则可能是团队最难采用的。某些项目经理具有根深蒂固的指挥控制习惯。团队也倾向于由别人告诉他们做什么，即便他们并不喜欢被指派工作。改变这一现状的过程往往会违背人类的本性。

如同其他所有事物一样，合作最好适度。为了确保团队意识到关键信息，有些合作是必要的。但是，当确保每个人都熟知所有信息时，会降低团队以稳定速度交付价值的能力，导致合作变得混乱，而这可能就是一个团队无法很好地一起工作的一种原因。合作也并不意味着**共识**（consensus）。有些时候冲突也是适当的、合理的。但是，过多的冲突会带来过多的伤害，特别是当它没有通过在正确的渠道发生时更是如此。

我曾和一个无论如何都无法一起好好合作的团队共事过。我无法准确地找到原因，但我认为有几个因素在发挥作用。

该团队的一些成员来自有毒的环境。他们已经转到采用敏捷方法的团队，以摆脱他们以前的环境。但是，我不确信他们是否采纳了敏捷的原则和价值观。这些成员常以“我们正在敏捷”作为各种不正常行为的借口。例如，他们缺乏专注，据称是因为要和队友频繁互动。

该团队确定了几个“头儿”，这些人往往是在分析、开发、测试技能以及敏捷方法的知识方面具有更丰富的经验。但是这些人逐渐滑向了指挥控制的行为方式，有时甚至吩咐团队成员就某一个问题停止互相交谈。一个成员说他觉得自己从原来有一个老板变成了有4个老板。

该团队无法公开讨论他们的想法，他们只在极罕见的情况下才尝试进行讨论，他们将讨论包裹在大量过程中，致使讨论没有任何收获。

该团队最终解散，部分原因是没有足够的工作，但主要原因是它功能运转失灵。有趣的是，该团队总能兑现承诺。有些人可能会说应该再给他们一次机会，因为他们能够交付。

对比一下我所在的另一个团队。这个团队不是在敏捷环境中工作，但决定尝试敏捷方法。像前一个案例一样，开发人员没有完全认同敏捷方法。与前一个团队的想法相呼应，不认同的开发人员最初时说他觉得有2～3个老板而不是一个。我们能够解决这些问题，讨论彼此的关注点，并利用这些分歧来找出改善的机会。我们建立了一些工作协定，并且在这个团队中从未出现指挥控制的方式。这主要是因为我们互相尊重，并且能够无需借助流程就能讨论这些问题。最基本的是，合作意味着在一个项目中工作的人形成一个真正的**团队**（team），而不是**工作组**（work group）。这使得他们能够解决任何困难，而不会因为生闷气或急着去找身边最近的决策者。

也许是更重要的是，合作也意味着，团队成员承诺完成共同的目标，而且他们并不害怕走出自己的专业领域去帮助团队中的其他人。团队中的每个人都有自己的专业领域，如开发、测试或者分析，他们花了大量时间在这些领域，但是在有需要时，他们能够跳入其他领域并完成一些工作来帮助团队实现总体目标。对于一个业务分析师，他可以推进整个团队进行合作，利用团队成员和利益相关者的深刻洞察辅助分析，并在团队中其他人遇到困难时帮助完成测试和文档工作。这也意味着，分析师不再把所有分析工作对其他人都藏起来，也不再只做分析工作。他们作为团队成员将所有时间投入到团队中，而不再限定为业务分析师角色。这种合作的结果是，角色变得模糊，但团队处在更能频繁交付产品增量的位置，从而获得更有意义的反馈。

1.4　迭代

迭代（iterate）的一个鲜为人知的同义词是反复排练（rehearse）。仔细想一想，就会发现这是**迭代**的一个特征。可以通过一个又一个特性来构建软件应用，或者

建造车辆的几个样板模型来尝试装配流程并生产车辆用于测试。无论哪种方法，都是在排练方法和设计决策。迭代给团队一个提出方法并进行尝试的机会，所以就算你做了一个糟糕的决定，也不至于浪费大量的工作。

迭代方法的关键是针对每个迭代的产出获得可行动的反馈，这样你就知道自己是否是在朝着期望的结果前进。为了获得有用的可行动的反馈，需要解决方案的一部分能够工作，以便利益相关者可以看到并做出反应。通过一系列迭代获得可行动的反馈，就形成了持续的学习。

与运营工作不同，IT项目和其他类型的知识工作从不直接使用可重复的流程。当你从事操作工作时，例如组装车辆或者处理索赔，许多步骤都能够从一个单元复制到下一个。识别改进变得很容易，因为在一组特定的工作任务周期之间往往只有很短的时间。运营工作具有重复性和可预测性。如果你愿意，可以反复学习如何做得更好。

另一方面，知识工作有点儿不同。项目就像雪花，没有两片是一样的。如果有机会去体验不同的项目，从一个项目获得的经验很可能并不适用于另一个项目。关注持续学习，并将迭代作为其关键组成部分，就能提醒团队要经常停下来并找出可以改进的地方。这也有助于利用实际的工作产出来确定有意义的里程碑，而不是使用中间产物衡量进展。

在会议投稿系统的案例中，我们以不同的方式使用迭代获得了收益。首先，团队产生了一条特性的定期输出流，我作为产品负责人（product owner）会进行检查并提供反馈，要么是关于系统的感观，要么是关于其功能的。团队用我的反馈来影响他们后续将要完成的特性。我们还对用户发布了投稿系统的多个**版本**（release），并利用第一个版本的用户反馈来影响特性如何进一步设计，并作为输入供我们决定哪些特性在下一个版本发布，而哪些特性要推迟。

1.5 简化

敏捷宣言背后的一条原则是“最大化未完成工作的数量”。这意味着想要交付最少的产出（output）并使其结果（outcome）最大化（正如在 1.2 节所介绍的），并且只使用绝对必要的流程。我在 1.2 节中讨论过交付正确的工作，这里我想讨论一下如何简化交付正确工作的方法。

简化意味着团队一开始就应该采用一种**刚好够用**（barely sufficient）的方法。当团队启动一个新项目时，应该首先确定对项目成功绝对必要的活动，然后就只做这些活动。在项目进行的过程中，经过一些反思，你可能意识到需要一些额外

的活动来克服所经历的挑战。这很好，只要你的团队也愿意放弃一些不再需要的活动。你想要从一个勉强够用的方法开始，是因为团队往往会发现停止某些进行中的活动比增加新活动要困难。从一组小的活动开始，你就给了团队一个保持精简流程的战斗机会。

简化意味着采用最直接的路径到达目的地，并且毫不留情面地询问为什么有人想走其他路径。简化也意味着不要让完美成为良好的敌人。我这里主要指的是晦涩难懂的建模语义、用例格式、商业规则语言以及类似的东西。我见过把太多时间浪费在模型是否100%正确的学究式争论上的情形，而模型只不过是用来辅助进行有关最终产品的沟通的。在有些情况下，准确性是需要的也是必要的。例如，当为团队和利益相关者整理共同使用的**术语表**（glossary）时，或者记录业务规则以便未来参考时。但是，如果模型或工作是为了中间沟通，就没必要穷究细节，因为你总是能通过对话进行澄清。

简化意味着用简单的规则指导复杂的行为。不要指望规定的流程能控制团队的工作方式。给团队提供一些直接的指导原则，并让他们从中学习。有些最优雅的解决方案是最简单的。不要掉入开发不需要的复杂的解决方案的陷阱。

最后，简化意味着如果一个模型不容易被记住，那么它被使用的机会也就几乎为零。这就是我更喜欢 2×2 矩阵的原因，如**情境领导模型**（context leadership model）和**基于目的的对准模型**（purpose-based alignment model）（都会在第 12 章介绍）。限制为二维的想法可能过于简单，但这增加了模型被用于一些良好目的的可能性。另外，因为这些模型相当简单，它们很可能包含一些细微的差别，这使得他们非常强大，以适用于不同的情况。

当启动会议投稿系统时，我们从一个很小的流程开始，这一流程是我们经过一些调整后发现特别适合我们的（由于我们团队的规模较小且对该**领域**非常熟悉）。我们发现，冲刺计划会议（sprint planning meeting）、站会（standup）和演示（demo）给我们造成太多流程开销，因为我们对于一个版本中想处理的用户故事顺序已有共识，并且当为我这个产品负责人准备好用户故事进行验收并提供反馈时，我们通过版本库进行沟通。随着项目的推进，我们在必要时增加了一些额外活动，但整体上我们能保持整个方法非常有效。

1.6　考虑情境

最佳实践（best practice）通常用来描述在一个项目或组织中取得成功并被他人复制的技术或过程。但是，在一个项目中发挥很好作用的实践在另一个环境中

可能完全失灵。一个实践在一个给定项目上是否有效，受很多环境因素影响。正因为这样，我通常喜欢使用**恰当的实践**（appropriate practice）或良好实践（good practice）来强调这些实践并不是对所有项目都是最佳实践的事实。

当决定选择哪个流程、实践或技术时，团队需要考虑情境，以确保他们所做的事情会成功并且不会造成浪费。归根结底，也许考虑情境是唯一真正的最佳实践。

一个可能并非显而易见的关键点是项目团队需要决定采用哪些实践、流程和技术，并且随着项目的进行，学到更多东西时乐意改变哪些实践。项目团队经常是在组织定义的流程内进行工作，他们相信必须严格遵循的这些流程，其实很多时候对项目反而有害。通常如果那些项目团队对实际情况稍做检查，就会发现他们的选择比他们想象的要多。

1.7 明智决策

在很多类型的组织中，成功取决于明智的、及时的决策，无论这些组织是营利的、非营利的还是政府组织。我曾在成功的组织的多个成功项目中工作，发现它们都有一个共同特点，就是明确的决策。相反，我也有机会在那些未达到理想情况的案例中进行学习，它们也有一个共同因素，就是糟糕的决策或者没有决策。

不要误解我的意思，我不是说："这都是关于决策的。"否则这将是我唯一的指导原则了。但决策在 IT 项目中确实起着巨大作用。在过去的几年中，这一启示在我身上已经结晶不见，真正将其拉回关注焦点的事情是写《Stand Back and Deliver》一书。该书的第 5 章"决策"侧重于使用**业务价值模型**（business value model）帮助梳理对话结构，以便做出明智决策。在我写那一章的同时，由我的朋友 Chris Matts 介绍的"真实期权"的深远影响一下子击中了我的脑袋。没错，决策的时机和决策本身一样重要。

决策的一个同样重要的方面是谁实际做出决策。人们希望这个做决策的人足够了解情况又处于一个可以做决策的位置。有趣的是，在很多组织中被期望做大多数决策的人——高级领导——对很多情况并不了解。这是因为决策需要深入、详细的知识，而这些知识是这些领导并不具备的，他们要么无法获得超过其负责范围的大量信息，要么因为在信息从个人到主管，到经理，到行政领导传递过程中有信息过滤，没有得到足够信息。解决这些问题的一个有效方法是，将决策权分散到组织中去。这有助于确保具有相关信息的人也是能够做决定的人。一个最好的例子就是团队自行决定项目进行的最佳方式，只要他们对项目的预期结果和约束条件有正确的理解就可以。

决策中的另一个重要概念是各种**认知偏见**（cognitive bias）使人们远离理性的人，而经济学家等人就愿意相信他们自己就是理性的人。Daniel Kahneman、Dan Ariely 和其他很多人在过去几年中对认知偏见进行了大量的研究。（参见本书最后提供的关于认知偏见的资源参考清单。）这些观念对于 IT 项目的决策具有重大影响，我认为值得进一步探讨。

决策没有在业务分析周期的覆盖范围内，这可能是因为进行业务分析的人可能不是最终的决策者。但这并不能改变他们经常要引导决策过程的事实，这在很多方面都是非常困难的。为了确保决策的制定并进行沟通，理解这一过程中涉及的挑战对于 IT 项目的成功是非常有帮助的，同时这些挑战也是本书中反复出现的主题。

我将在第 4 章对决策展开进一步讨论。

1.8　反思与适应

团队应当不断地从经验中学习，以改进所使用的方法和项目的结果。项目常常持续超过几个月的时间。在这段时间内，业务状况、团队成员对项目目的的理解和该项目的周围环境都会增长和变化。团队应该寻求利用这种变化的优势，以确保在结果交付时项目的结果符合利益相关者的需要，而不仅仅是符合项目启动时利益相关者所理解的需要。

项目团队一直在做事后分析或经验学习，团队成员在项目结束时聚在一起讨论所发生的事情——通常是消极的方面——寄希望于他们能够记住，以便下次能做得更好。如果这种方法被认为是一个良好的实践，那在项目过程中做同样的事情岂不更有意义？此时团队还有时间做出改变并影响结果。这就是**回顾**（retrospective）背后的理念，回顾为团队提供了一种讨论项目到目前为止发生了什么的机制——既有团队做的好的事情，也有改进的机会——进而决定需要实施什么改正措施。

不管你采用的是什么方法，回顾都是一个很有用的技术。举一个回顾很有帮助的例子。几年前我曾参与一个非常大的项目，是为一家大型金融机构修改加载决策过程[①]。我所在的团队负责从一个新的组合信用政策中提取业务规则。我们发现，除了要参考信用政策本身，我们还需要和一群来自公司各个部门的**行业专家**（subject matter expert，SME）紧密协作。为此，我们所能采用的最有效的方式就

① 通常的流程为“请求—数据加载—模型评估—风险决策—返回结果”。——译者注

是每周一次将 SME 们召集在一起，进行业务规则工作会议。每周结束时，我们会举行一个回顾会议，讨论业务规则会议进行得怎么样，并识别改进行动。在召开工作会议的几周时间里，我们发现每次会议都比前一次更加顺利，而且我们经常发现某一周被认定为需要改进的项目在几周后已经做得很不错了。

1.9 总结

我不是一夜之间想出这些指导原则的。这些原则来自于多年的经验和试错。最初引起我想创建一个清单的事件是后来称为敏捷项目领导力网络（APLN）的一次创立会议。作为一种自我介绍的方式，Alistair Cockburn 建议我们分享我们对世界的看法，实际上就是解释我们为什么在那里。当时我绞尽脑汁要表达清楚自己的观点，在回到家仔细思考之后，这个清单的第一个版本就在脑海里出现了。自此之后我修改了几次，这些想法已经固定成型保持不变了，而且它们是我的首要原则，每当我试图解决之前未曾遇到的情况时，都会回过头来基于这些首要原则进行思考。当你阅读本书其他部分时，我希望你把它们记在心里。

1.10 切记

- 当你以最少的产出让结果最大化时，就是在交付价值。
- 团队应当持续探索共同合作的方法来交付价值。
- 缩短反馈环以鼓励持续学习。
- 不要做对交付价值不是绝对必要的事情。
- 具体问题具体分析。
- 有目的地做出决策。
- 从过去学习以改进未来。

第 2 章

有用的概念

2.1 简介

本章将对书中其余章节介绍的方法和技术的背后概念进行说明。这些概念与我们如何思考分析的分类方法有关：

- 需要和解决方案；
- 结果和产出；
- 发现和交付。

通过介绍这些概念，希望为本书其他章节创建一个共同语言。这些概念所使用的术语也罗列在术语表中。

2.2 需要和解决方案

在前言中，对分析涉及的活动进行了介绍：

- 理解利益相关者；
- 理解情境；
- 理解需要；
- 理解解决方案；
- 组织并持久保存解决方案信息。

对我而言，一些关键定义是按顺序来的。为此，参考《Guide to the Business Analysis Body of Knowledge Version 3》（BABOK v3）一书中介绍的业务分析核心概念模型（Business Analysis Core Concept Model，BACCM），其中定义了 6 个核

心概念，如表 2-1 所述。

表 2-1　BACCM 中的核心概念

核心概念	描述
变更	响应需要的变化行为 变更是为了提高企业绩效
需要	待解决的问题或可利用的机会 需要能够通过激发利益相关者的行为来引起变更，变更也能通过削弱或加强现有解决方案交付的价值来引发需要
解决方案	在情境中满足一个或多个需要的具体方法 解决方案通过解决利益相关者面对的问题或帮助利益相关者更好地利用机会来满足需要
利益相关者	与变更、需要或解决方案相关的团体或个人 利益相关者通常被定义为对变更感兴趣，能够影响变更或受到变更影响的人 利益相关者通常根据其与需要、变更和解决方案的关系进行分类
价值	某种事物在情境中对利益相关者是值得的、具有重要性或很有用处 价值可以被看作是潜在或实现的回报、收益和改进，也有可能使损失、风险和成本减少而得到的价值 价值可以是有形的，也可以是无形的。有形的价值是直接可衡量的。有形价值往往具有重要的财务成分。无形的价值是间接可衡量的。无形价值通常具有重要的激励成分，例如公司的声誉或员工的士气 在某些情况下，价值可以用绝对值评估，但在多数情况下，只能通过相对的形式评估：从一组利益相关者的视角看一个解决方案选项比另一个更有价值
情境	影响变更、受变更影响并提供变更信息的环境 变更发生在情境中。情境是环境中与变更相关的一切。情境可能包含态度、行为、信念、竞争对手、文化、人口特征结构、目标、政府、基础设施、语言、损失、流程、产品、项目、销售、季节、术语、技术、天气和任何其他满足定义的元素

这里最相关的核心概念是需要和解决方案，因为它们描述的是分析主题。理解这两个概念之间的区别非常重要，很多 IT 项目因为在基于解决方案加速前行之前未能确定需要，遭受了巨大痛苦。

在启动一个项目之前，你可能已经知道，应该要明白为什么要启动——换句话说，你要解决的问题是什么。如果你理解了想要解决的问题，或想要利用的机会——需要——就有可能选择最有效的解决方案，并避免把无谓的时间和精力投入到开发一个不需要的解决方案上。我猜你也能说出几个参与的 IT 项目，其中团队跳过问题的理解直接冲进解决方案的交付。我也曾参与过这样的项目。

为什么团队即便知道理解需要是优秀的实践，但还是会反复地跳过对需要的理解？有时是由于受到项目发起人的压力，这些发起人偏好特定的解决方案并蒙受一些在第 4 章介绍的认知偏见的痛苦。更可能是，团队不知道如何描述需要，以帮助确定合适的解决方案。其实这样的技术一直存在而且就在我们眼皮底下：目的和目标。

BABOK v3 中包含业务目的和业务目标（本书中简化为目的和目标）的定义，如表 2-2 所示。这些定义的好处是提供了一种方法来区分这两个容易混淆的概念。

表 2-2 目的和目标

术语	定义	健康保险案例
目的	组织寻求建立并维持的状态或条件，通常定性表示而不是定量	提高处理能力以应对索赔的增加
目标	一个可衡量的结果用于表明目的已经实现	到 12 月 31 日把每周 1000 单纸质保险索赔降低到每周 500 单

具体而言，目的是想要完成的事情（想要满足的需要），目标是如何衡量成功实现目的的程度。在本书中对每个术语在什么时候使用以及在什么地方使用都会非常注意。

既然目标是要可衡量的，当团队建立目标时，把目标的一组特征记在心里是有好处的。这组特征通常称为 SMART，如表 2-3 所述。注意这一缩写代表不同的意思，这里选择使用这一缩写，而其中 A 代表“达成一致”（agreed upon）以加强团队对目标及其含义达成一致的理念，从而更可能对将要做什么形成共识。

表 2-3 良好目标的特征

属性	描述
具体的	你确切地知道想要实现什么并有明确的期望
可衡量的	当你朝目标前进时，要有能力辨别
达成一致的	达成目标所涉及的每个人都要赞同什么是真正的目标而且是值得达成的，以及当达成时如何辨别。这个概念加强了共识的理念。而如果一个目标是可达成的，但如果整个团队不理解或不认可它是好的目标的话，就不妙了
现实的	你不希望通过设定一个无法达成的目标让团队感到挫败。你可能需要把目标延伸一点儿，但设定一个在所处环境的约束条件下绝对无法达成的目标对你没有任何好处
时间限定的	你需要知道何时期望做完。否则你可以永远持续下去，最终无法完成任何事情

为了强化良好目标的特征，Tom Gilb 在《Competitive Engineering》中建议了表 2-4 所示的属性集，每个目标都能找出这些属性。

表 2-4　目标的属性

属性	描述	例子
名字	目标的唯一名字	降低每周收到的纸质索赔申请
单位	度量的内容（Gilb 称之为刻度）	每周收到的纸质索赔申请数量
方法	如何衡量（Gilb 称之为计量器）	计数每个日历周收到的提交类型为纸质的索赔申请次数
目标值	努力达成的成功基准	每周 500 单索赔
限制	努力避免的失败基准	每周 1000 单索赔
基线	当前绩效水平	每周 1000 单索赔

注意，在这个例子中限制和基线是一样的。这表明如果这是项目的目标，而你未能改变一周内收到的纸质索赔申请数量的话，项目就失败了，但任何改进都至少是在正确方向上前进了一步。在其他情况下，你会看到把基线和目标值之间的中间值设置为限制，这意味着如果你未能完成某些改进，项目就是失败的。设置这些属性的一个重要价值是为了决定目标值和限制应该是多少而引发的讨论，这让大家对项目成功的样子形成更加明确的理解。

首先理解需要并能够通过目的和目标进行描述，让你有机会与团队针对为什么考虑开始（或继续）一个特定项目建立共识。这也为提出“这个需要值得满足吗？”这一问题形成了一个基础。因而在表 2-4 纸质索赔申请的例子中，团队就可以问：

- 立刻提升我们处理纸质索赔申请的能力是值得的吗？
- 为什么我们认为收到的索赔将会增加？
- 纸质索赔申请是我们处理索赔能力的最大障碍吗？
- 我们提升纸质索赔能力的同时要放弃什么？

将需要和解决方案分开考虑，让团队有机会发现多个潜在解决方案，并且当要确定交付的解决方案时还能从中选择。拥有这些可选项，增加了团队高效交付解决方案的机会，而这些解决方案在满足利益相关者需要的同时，还能满足诸如时间和成本等限制。

将需要和解决方案分开考虑也有助于澄清利益相关者和团队的责任。需要来自于利益相关者，特别是**项目发起人**（sponsor），而解决方案来自于团队。现实