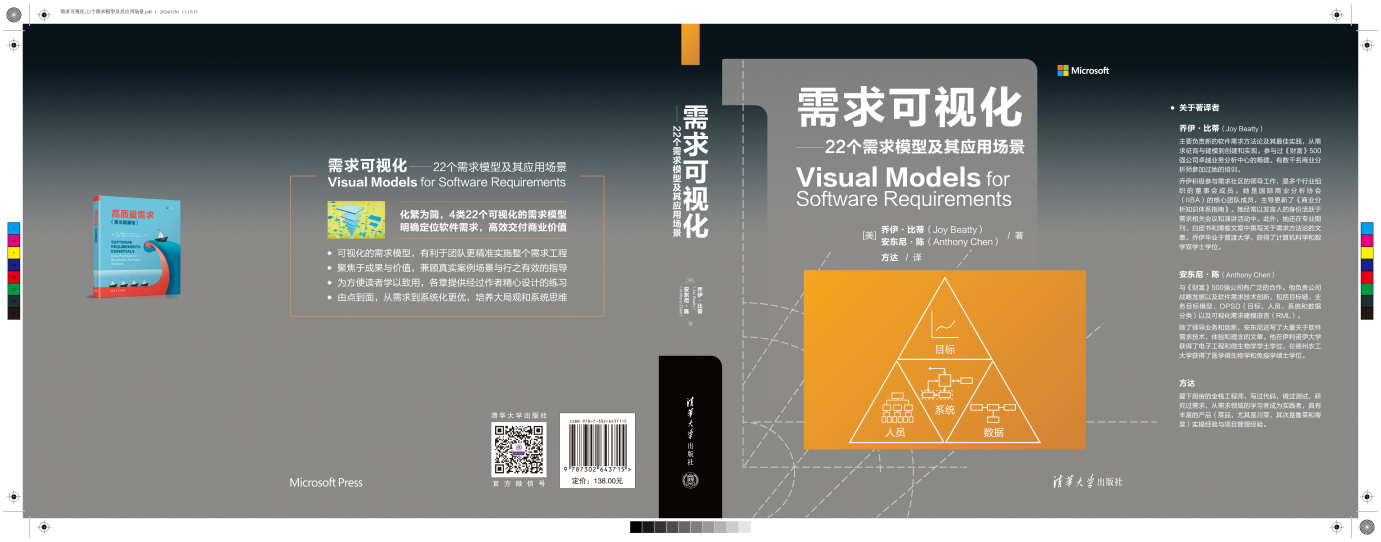
需求可视化：22 个需求模型及其应用场景

(原书名：软件需求与可视化模型)



(美) Joy Beatty / Anthony Chen 著

方达 译

清华大学出版社

北 京

本书献给所有默默无名的分析师，项目成功的军功章，应该有他们的一 半。

目录

推荐序

关于需求工作，最引人注目的一件事情是，在学术界的人认为它所涉及的东西和业内人士实际所做的事情之间，往往存在着巨大的差异。

学术界的人认为他们遥遥领先，因为他们有广泛的模型和技术，有完善的实验研究（与所谓的行业“专家”一起完成）、理论分析以及大量充满了优秀建议的教科书。他们不明白为什么业内人士在采用他们的方法方面如此缓慢。

业内人士也认为他们遥遥领先，因为他们有多年的经验，有能工作的软件（经过一番挣扎之后），有成熟的需求管理方法，包括可跟踪性矩阵、审查、配置管理以及优先级和状态属性。他们不明白为什么学术界的人半天都追赶不上现实。

这就像看着两名自行车赛手在一个圆形赛道上，相隔 180 度，无休止地绕圈。

这就是乔伊·贝迪和安东尼·陈的这本书是如此之好的原因。他们是以自己的经验说话的实干家。但这才是最关键的一点——他们熟悉研究人员所倡导的一系列模型。更妙的是，他们已经稳步地将越来越多的模型纳入自己的实践。现在，他们正在使用的模型使他们能方便和有效地分析他们遇到的所有需求。他们看到并听说过学者们在谈论的一些东西，例如使用 KAOS 或 i*的目标建模（goal modeling）。他们还看到过那些只需要一个环境模型（context model）就能让项目变得清晰的挑战，也看到了那些缺乏像数据字典这样简单和传统的东西的项目所遭遇的灾难。另外，他们对你必须同时使用所有这些东西这一基本事实有真切的把握。

他们清楚地认识到，在一个需求过程（requirement process）中，如同在任何系统或产品中一样，整体大于部分之和。一个机身，两台强大的发动机，一个航电系统，以及一个机组，只有当它们被整合在一起时，才能构成一架飞机。当他们一起工作时，就会出现一样新的、没有任何一方能单独实现的东西：飞行的能力。

为了使一个需求过程“飞起来”，第一步是理解存在不止一种需求模型。在一份合同中，一个需求购物清单是非常重要的，但凭借它本身很难检查其正确性和完整性，而且它没有提供任何关于如何发现需求的建议。需要不同的需求模型来帮助发现、检查和分析需求。“购物清单”是一个输出，而不是唯一的输入。

乔伊和安东尼在本书中确定了四类主要的需求模型：目标、人员、系统和数据。

他们所说的“目标”（objectives）最接近于传统的需求，但要开始于一个更早和更临时的阶段：调查企业的目标是什么，然后从那里开始研究应该如何满足这些需求。

显然，“人员”意味着要调查谁是所设计的系统的利益相关方，他们将如何使用系统，以及他们想从系统那里得到什么。

“系统”意味着探索新系统所要做的事情的上下文、接口和事件。这在很大程度上是一套传统的分析技术，经常被那些盲目相信软件时尚的人认为是过时的。乔伊和安东尼敢于正视这一点，并明确指出老旧——即便不完整——不一定意味着错误。当然，问题的关键在于，20 世纪 70 年代的系统分析本身是不充分的——例如，它经常因为缺乏对“目标”的恰当关注而失败。

最后，“数据”意味着定义业务用户需要的信息，并对它们在系统中的使用情况进行分析。同样，其中很多都非常传统，尽管它不仅包括数据分析，还包括状态模型和报告分析——对一个古老话题的现代理解。

这里存在一个必要的复杂性。需求模型是紧密联系的。目标与功能联系，功能与过程联系，过程与用例联系，用例与用户界面联系。乔伊和安东尼展示了这种需求架构——你可以称它为元模型（meta-model）——如何为单独的项目量身定制。他们已经试过了，一遍又一遍，而且很有效。

本书讨论的方法是为支持业务过程（business processes）的软件设计的。 其他类型的项目需要相关但截然不同的需求过程，例如在开发涉及硬件和软件的一个大众市场产品系列的时候。 乔伊和安东尼专注于一个领域，即业务软件（software for businesses）的领域。 结果是一种创新但引人注目的需求方法。

——伊恩 • 亚历山大（Ian Alexander）

前言

可视化需求模型（visual requirements models）是分析软件需求最有效的方法之一。他们帮助分析师确保所有利益相关方——包括主题专家、业务利益相关方、高管和技术团队——了解拟议的解决方案。可视化使利益相关方一直保持兴趣并参与，这是对需求进行查漏补缺的关键。最重要的是，可视化创建了解决方案的一个全景，帮助利益相关方了解解决方案将提供什么和不将提供什么。虽然可视化有这么多优点，但还是有许多业务分析师和产品经理在用列出了成千上万条需求陈述的电子表格或者文档来创建非可视化的需求。这种文件非常繁琐，令人不知所措，审查起来很枯燥，而且极难分析出缺失的需求。这种实践是反映了需求培训的一个症结，这些培训通常将重点放在如何写一个好的需求上，而不是放在如何对整个解决方案进行分析上。

本书帮助业务分析师、产品经理及其组织中的其他人使用可视化模型来征询（elicit）、建模（model）和理解（understand）需求。本书描述了一种简单而全面的软件需求可视化建模语言，称为 RML（Requirements Modeling Language，需求建模语言），它是具有最佳实践的模型的一个集合，经常在行业中以一种“特别”或“即席”（ad-hoc）的方式使用。

本书面向的读者

虽然本书主要面向业务分析师（business analysts）[1]和产品经理（product managers），但我们认为，项目经理、开发人员、架构师和测试人员也能从本书收获不小的价值，因为本书可以帮助他们理解所接收的信息的标准，使其工作变得更容易。在整本书中，我们一般将从事这项工作的人称为“分析师”，但该角色在不同组织中有许多不同的头衔。当书中提到“你”或者“我们”时，指的也是“分析师”。

事先声明，我们的经验主要来自那些在现有基础设施上运行的软件项目，例如为内部开发的信息技术（IT）系统、面向消费者的大规模“软件即服务”（Software as a Service，SaaS）系统以及云系统。虽然我们在独立软件（通用软件或 packaged software）以及嵌入式系统上使用过 RML，但这些类型的项目并不是我们的主要关注点。不过，基于我们对这些系统的有限经验，我们仍然认为，使用这些系统的读者会在 RML 中发现令人难以置信的价值，而且我们期待着能从这些读者那里得到反馈，以便对它进行改进。

假设

本书没有涵盖关于需求的基本信息；因此，本书假设你已经有了编写软件需求的基础知识。

[1] 译注：本书的主要读者是业务分析师。Business Analysts（BA）在许多地方都翻译为“商业分析师”。但是，并不是所有 business 都是“商业”。

另外，本书假设你对软件开发过程（例如迭代方法、瀑布方法和敏捷方法）有一个基本的了解，并理解需求如何与这些方法相适应。

本书不面向的读者

如果你是刚开始做业务分析师，那么可能应该在阅读本书之前阅读 Karl Wiegers 的《软件需求》第 3 版（清华大学出版社，2023 年重译版），以了解需求实践的概况。如果开发的是包装好出售的消费者软件，那么一些概念虽然有用，但你可能发现本书的业务导向会令自己分心。如果你是一名产品经理，专注于软件产品的战略或营销方面，而不是软件构建，那么本书可能不太适合，因为它非常强调如何设计功能以获得高的最终用户接受度和满意度。

本书的组织

本书进行了高效的组织，可以把它作为一个参考指南使用。

第 1 部分“模型简介”从常规意义上介绍了模型，然后继续讨论 RML 和模型的四个分类：目标模型、人员模型、系统模型和数据模型（OPSD）。

第 2 部分~第 5 部分的每一章都涵盖了一个 RML 模型，并且采用一致的布局，包括：

- 一个将模型和现实世界联系起来的开章故事。
- 模型的定义。
- 模型模板。
- 关于使用何种工具来创建模型的建议（工具提示）。
- 一个虚构的例子。
- 对如何创建和使用模型的解释。
- 一个练习，供你练习模型的使用。

每一章的练习都基于同一个样板项目，该项目贯穿全书所有章节。

第 6 部分“模型大局观”解释了如何选择模型，以及如何综合使用多种模型来推导需求。

附录 A 包含两个模型速查表，你随时可以参考它来选择模型。附录 B 罗列了创建模型的一般准则，其中包括所有模型的元数据和关于模板的提示。附录 C 包含书中所有练习的参考答案。最后还有一个词汇表，定义了全书使用的术语。

找到你的最佳起点

可以直接从头阅读本书，但对某些人来说，在深入了解每个模型的细节之前，可以先阅读

第 6 部分来体会一下上下文。下表提供了更多的指导。

如果你是	遵循这些步骤
初次接触需求建模或更广义的可视化建模	从头开始一直读到本书最后，这样可以了解需求模型的常规知识，了解各种具体的模型，最后学会综合运用它们。
熟悉可视化需求建模，而且是已在使用类似模型的业务分析师	建议所有章都读完，以了解 RML 处理可视化模型的方式与其他建模语言有何不同。然而，你也可能发现第 6 部分更有用，可以先了解如何选择模型并在项目中综合运用。然后，在做具体的项目时，可以根据需要随时参考具体的模型章。

模型快速入门

本书包含大量关于模型的信息。信息量有点大，可能会令人不知所措。所以，我们为你开发了一种方法来快速上手，它会使用尽可能少的模型，但仍然能为项目创造巨大的价值。这种快速入门的方法适合大多数基于 IT 的项目。以下“过程流程”图对这种方法进行了概述。

如图所示，首先创建一个过程流程。接着，根据过程流程中的步骤来创建一个需求映射矩阵（RMM）。然后，为每个屏幕都创建一个显示-操作-响应（DAR）模型，并建立它们与业务过程的映射。最后创建数据字典，以确保所有字段都被覆盖，而且校验规则是已知的。

虽然其他许多模型的价值还没有被发挥出来，但采用这一系列步骤无伤大雅。它的结果是，你的需求将按过程流程步骤组织起来，屏幕也将被映射到过程步骤，以确保 UI 反映了关键的过程。

本书约定和特色

本书采用了使信息具有可读性并易于理解的一些约定。

- 每章开头都有一个来自现实生活的故事，用特殊字体显示，帮助读者建立上下文。

- 所有 RML 模型名称都首字母大写。不属于 RML 的其他建模语言的模型则全部小写[①]。
- RML 模型的基本构建单元称为“元素”，这些模型元素没有首字母大写，这样它们就不会与模型名称混淆了。
- 本书末尾的词汇表包含了我们认为对 RML 很重要的术语。这些术语在本书中以斜体字出现[②]。
- 每个模型模板小节最后都提供一个“工具提示”，建议了用于创建当前模型的工具。

配套资料

欢迎下载 RML 模型模板。在你的项目中创建本书描述的模型时，可以直接使用这些模板。以下网址提供了一套完整的 RML 模型：

http://go.microsoft.com/FWLink/?Linkid=253518

在下载回来的压缩包中，我们解释了如何使用这些模板。这里简短重复一下：将压缩包解压到一个方便的位置。每个模型都有一个模板。作为 Microsoft Visio 文件的模型包括一个.vst 文件和一个.vss 文件，这两个文件是使模板正确工作所必需的。其余模板采用 Microsoft Excel 或 Microsoft Word 格式。模型速查表也在压缩包中。

致谢

从我们在 Seilevel 的团队，到我们在世界各地的需求同事，到多年来启发和帮助我们完善 RML 的客户，如果没有你们所有人的合作，这本书是不可能完成的。

非常感谢 Seilevel 的员工，他们帮助研究、审查、写作、编辑、起草模型，并提出了许多非常好的、非常有挑战性的问题：Joyce Grapes，James Hulgan，Betsy Stockdale，Michael Liu，Candase Hokanson，Jeremy Gorr，Balaji Vijayan，Marc Talbot，Matt Offers，Ajay Badri，Jason Benfield，Geraldine Mongold，Kell Condon，Clint Graham，David Reinhardt，Weston Eidson，Abdel Mather，Kristin DiCenso，Rob Sparks 和 Lori Witzel。

我们对众多的审稿人表示最诚挚的感谢，感谢他们花费宝贵的时间来阅读手稿，并提出看法和批评，帮助我们不断改进本书：Joyce Statz，Kent McDonald，Sarah Gregory，Ljerka

---

[①] 译注：这是英文版的约定，中文版在提到模型名称时未做特殊处理，而且除非为了避免歧义，甚至不会添加“模型”或者“图”作为后缀。例如，一般直接说“创建一个过程流程”，而不会说“创建一个过程流程图”。

[②] 译注：同样只适用于英文版。

Beus-Dukic，Mary Gerush，Karl Wiegers，Ellen Gottesdiener，Scott Sehlhorst，Ivy Hooks 和 Anne Hartley。特别感谢 Karl Wiegers 和 Ian Alexander，他们都提供了写作指导，并对我们的模型进行了大量测试和反馈。

非常感谢编辑团队，他们非常勤奋，而且还很有趣。是他们使本书成为现实。感谢组稿和策划编辑 Devon Musgrave，以及项目编辑 Carol Dillingham，他们都来自微软出版社。还要感谢项目经理和文字加工编辑 Kathy Krause；图文编辑 Jean Trenary；校对 Jaime Odell；图形艺术家 Jeanne Craver；以及索引制作员 Jan Bednarczuk。

最后，感谢我们的家人，他们和我们一起忍受了漫长的写作过程。乔伊感谢她的丈夫 Tony Hamilton，他在整个过程中帮助她保持了幽默感；感谢她的女儿 Skye，她在写这本书期间出生，并在书写完的时候掌握了如何睡上一整晚的精髓。事实证明，写这本书很像生孩子，都是几个月的孕育、待产和养育。安东尼感谢他的妻子 Gloria 的支持，感谢他的女儿 Mason，因为她在自己玩耍的时候允许老爸工作，而且在开电话会议的时候不会大声嚷嚷。最后，安东尼要感谢乔伊。如果没有她凭借其强大的意志力来推动本书的写作，你恐怕永远看不到这本书的诞生。

勘误和售后支持

我们已尽了一切努力来确保本书和其配套内容的准确性。自本书出版以来，任何上报的错误都会在我们的微软出版社网站上列出[①]，网址是：

http://go.microsoft.com/FWLink/?Linkid=253517

如果你发现了一个尚未列出的错误，可以通过同一个网址向我们报告。

如需要额外支持，请向微软出版社的图书支持部发电子邮件：mspinput@microsoft.com。

注意，上述地址不提供微软软件产品支持。

我们希望听到您的意见

在微软出版社，您的满意是我们的首要任务，而您的反馈是我们最宝贵的财富。请告诉我们您对本书的看法：

http://www.microsoft.com/learning/booksurvey

调查问卷很短，我们会阅读你的所有意见和想法。提前感谢您的意见！

[①] 译注：新的中文版已集成了截止 2023 年的所有勘误。

保持联系

让我们在 Twitter 上保持对话：http://twitter.com/MicrosoftPress。

第 1 部分 模型简介

第 1 章 RML 简介

在距离假日旺季还有九个月的时候，一家著名在线零售商确定了一组想在其网站上添加的关键新功能。它们能大幅改善客户体验，直接增加销售额，同时减少好几个国家的客户支持电话。据估计，这些功能每年的价值为 1400 万美元，而成本不到 200 万美元。产品经理确定了这些功能的需求和业务规则，开发团队和项目管理团队做了估算，显示项目团队应该很容易能在假期开始前完成。团队在截止日前努力工作，经常在晚上和周末加班，以保证到时能顺利发布。

八个月后，团队做了最后的测试，感觉非常不错。他们完成了一个非常长的改进清单，有望实现非常可观的回报。然后，一个测试人员注意到，税的计算不正确。不幸的是，这些计算只是冰山之一角。该团队忽略了与税务团队的沟通。事实上，他们甚至没有意识到需要这样做。如果真的这样做了，就会发现在某些国家运营的时候，需要遵循的计税规则相当复杂，需要与管理这些规则的第三方软件进行整合。项目被推迟了，那一季的 1400 万美元回报也就没有了着落。项目经理被解雇了，产品经理被重新分配到另一个不太重要的项目。

软件项目经常被遗漏、不完整或者不明确的需求所困扰（The Standish Group 2009）。由于错误的需求实践，大多数项目注定失败（Ellis 2008）。糟糕的需求是许多项目失败的根源。令人失望的是，在过去 20 年里，软件需求行业的成功率并没有大幅提高。虽然学术界一直在努力寻找对需求技术和工程方法的改进，但业务实践基本上没有变。随着各种新技术和大量工具的产生，软件编程实践已经相当成熟，但在开发软件需求时，往往还是要使用电子表格中的一长串“应……”陈述。采用了敏捷方法的项目也没有多大好转，往往还是要使用电子表格或工具中的一个长长的清单来维护他们的产品待办事项和用户故事。

定义 RML

RML（需求建模语言，Requirement Modeling Language）是专为需求的可视化建模而设计的一种语言，管理人员、业务和技术利益相关方都能方便地使用。RML 不是一种纯理论的建模语言。在开发 RML 的过程中，我们修改了现有模型以方便使用，并创建了新的模型以查漏补缺。其结果是一套完整的模型，专为软件需求建模而设计，经常受到复杂模型挑战的业务利益相关方（business stakeholder）可以方便地使用。我们已经在许多大规模的软件开发项目中成功运用了这些需求模型。

传统软件需求实践的挑战

遗憾的是，传统实践支持使用成千上万的“系统应……”需求，这类似于图 1-1 所示的一份冗长的清单。这些需求往往是通过与业务利益相关方的访谈和工作会议而生成的。普通人受到米勒魔数（Miller's Magic Number）的限制（参见下一节“人脑的局限性”），几乎不可能读懂成千上万的需求并有把握这些需求是完整的。另外，更严重的一个问题是范围蔓延（scope creep）[①]。在存在成千上万的需求时，如果没有某种方法将这些需求与可在整个解决方案内比较的价值联系起来，那么很难确定哪些需求应被删除。团队通常会将需求组织成合乎逻辑的分组，但这些分组通常还是太大，无法高效地处理。

像 Scrum 这样的敏捷方法有产品待办事项清单、用户故事和和验收标准等。许多 Scrum 传道者说，产品待办事项清单（product backlog）应该是一个没有嵌套的故事清单，但这并不比一个长长的需求清单好。验收标准也应该列出来，有时就列在一张便条卡（notecard）的一面上。但从事大型系统工作的人都知道，在一个可能存在数百个利益相关方的项目中，这种缺乏信息组织的做法是根本行不通的。

需求文档

REQ001 系统应具有名字、中间名和姓氏字段。

REQ002 如果存储的资料中有，系统应显示姓名。

REQ003 系统应要求填写姓名。

REQ004 系统应具有一个职位（position）或头衔/职称（title）字段。

REQ005 系统应要求填写头衔。

REQ006 如果存储的资料中有，系统应显示职位或头衔/职称。

REQ007 系统应具有一个电子邮件地址字段。

REQ008 系统应具有一个备用电子邮件地址字段。

REQ009 如果存储的资料中有，系统应显示电子邮件地址。

REQ010 如果存储的资料中有，系统应显示备用电子邮件地址。

REQ011 系统应要求填写电子邮件地址。

[①] 译注：范围蔓延（scope creep）是指项目范围不受控制地变化或持续增长。这通常是由于范围欠缺定义、记录或者控制，导致在项目进行期间，非预期的需求不断增加，最终脱离控制。

REQ012 系统应要求填写备用电子邮件地址。

REQ013 系统应具有一个白天电话号码字段。

REQ014 如果存储的资料中有，系统应显示电话号码。

REQ015 系统应要求填写电话号码。

REQ016 用户填写完毕后，系统应验证电话号码字段中的所有字符都是数字。

REQ017 如果电话号码字段中并非所有字符都是数字，系统应显示一条错误消息。

REQ018 系统应具有一个传真号码字段。

REQ019 系统应要求填写传真号码。

REQ020 如果存储的资料中有，系统应显示一个传真号码。

REQ021 用户填写完毕后，系统应验证传真号码字段中的所有字符都是数字。

REQ022 如果传真号码字段中并非所有字符都是数字，系统应显示一条错误消息。

REQ023 系统应具有两个街道地址字段。

REQ024 系统应要求填写第一个街道地址字段。

REQ025 如果存储的资料中有，系统应显示一个地址。

REQ026 系统应具有一个城市字段。

REQ027 系统应要求填写城市字段。

REQ028 如果存储的资料中有，系统应显示一个城市

REQ029 系统应具有一个州字段。

REQ030 如果存储的资料中有，系统应显示州名。

REQ031 系统应要求填写州字段。

REQ032 系统应具有一个邮政编码（zip code）字段。

REQ033 如果存储的资料中有，系统应显示邮政编码。

REQ034 系统应要求填写邮政编码字段。

图 1-1 一个冗长的需求清单

人脑的局限性

使用传统实践来创建软件需求的分析师在分析、组织和消费需求的过程中会遇到一些常见的问题。传统实践是使用一个冗长的文本需求清单，其形式包括各种“应……”陈述、用例、用户故事和产品待办事项清单。处理这种冗长的项目清单时，挑战来自于人类认知的一个基本限制。20 世纪 50 年代，认知心理学家 George A. Miller 发现，人类只能同时记忆和处理 7 正负 2 个数据项（Miller 1956）。这通常被称为米勒魔数（Miller's Magic Number）。

$$7 \pm 2$$

最新证据表明，这个数字甚至可能只有 3 或 4（Cowen 2001）。该数字代表的是大脑用于保存解决问题所需信息的一个“便签簿”（scratchpad）的容量。不管实际数字是多少，如果要求一个普通人同时思考 15 件事情，那么这 15 件事情中最多只有 9 件（可能更少）能真正得以保留并处理。如果需要处理更多的项目，那么只能一次处理几个，并迅速在记忆中切换。以去商店购买 15 样东西为例。如果没有一个购物清单，那么很可能在回到家时才发现少买了东西，或者买的东西不正确。同样地，如果有一个需求清单或产品待办事项清单，其中包含数百乃至数千项，那么除非将其分解成较小的分组，否则人的大脑根本无法理解这种复杂性。

图很容易，文字很难

既然我们这种原始哺乳动物的大脑存在这一基本限制，那么有什么解决方案呢？“一图胜千言”这句格言似乎很合适。对于正在开发的解决方案，其内部和周围会存在一些和过程、数据和交互有关的信息，对这种信息的可视化表示（图片）就是*模型*（Models）。你可能每天都在使用可视化模型而不自知。

在最近一次去赌场参加会议的过程中，在我登记入住并拿到房间钥匙后，前台女士告诉我如何去我的房间。她说：“从这里右拐，然后左拐，经过酒吧，经过老虎机，看到喷泉就右拐，你会经过一家餐厅，还有一家，然后会到达一个大厅，那里有几家商店，左拐，走到尽头，然后会在泳池入口附近找到电梯。”

我茫然地盯着她。那一刻，我所想到的是我下出租车后走到前台所经过的老虎机和赌桌的海洋。我猜的话，在去到房间的路上还会经过更多的老虎机和赌桌，这将使她刚才说的话更加令人困惑。但是，她马上给我带来了希望：“这里有一张地图，告诉你怎么走。”她画出了我从前台到电梯所需的路线，很像如图 1-2 所示的地图。我大大地松了一口气，因为除了最初的几个方向外，我完全没有能力记住她指出的所有方向。

但是，我现在至少有了一个模型，在我感到困惑时可以参考。一张地图！简单地说，当人

类解释信息时，图很容易，文字很难。

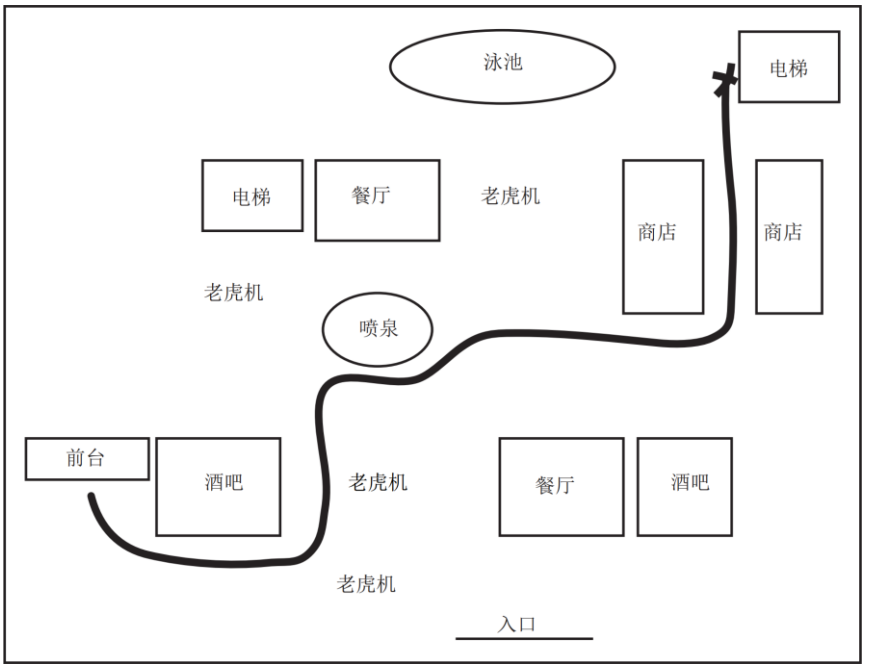


图 1-2 和怎么穿过一个赌场的口头指示配套的地图

需求模型

需求模型（requirements models）组织并呈现大量信息，帮助你识别缺失的信息，并为细节提供上下文（Gottesdiener 2002）。最重要的是，模型提供了可视化的分组，使你能通过有限的短期记忆快速分析大量不相干的信息。对于由数千条“系统应……”陈述组成的需求文档，阅读、解释和识别其中缺失的需求是很困难的，但模型可以提供帮助。

想象一下，在你面前有一系列如图 1-3 所示的杂乱无章的字母，必须按字母顺序找出缺失的字母。

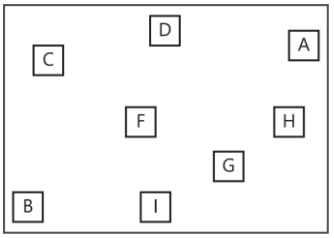


图 1-3 乱七八糟的字母——缺失了哪一个？

如果只是盯着这些字母，甚至把它们排成一排，但不按任何顺序，那么很难找出缺失的那一个（事实上，你刚才可能已经试过把它们一个接一个排列起来）。但是，如果按字母顺序排列，如图 1-4 所示，缺失的字母就一目了然了。

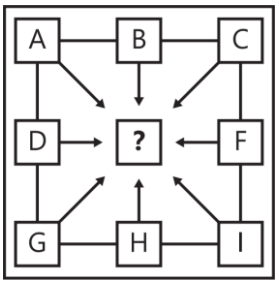


图 1-4 整理字母以找出缺失的那一个

为了找出缺失的需求，关键在于利用每个需求都以某种方式与其他需求相关这一事实。面对一长串“系统应……”陈述，要确保这个清单的完整性是非常困难的。但是，为需求添加结构，我们可以利用它们之间存在的关系，并通过每次都分析一个较小的信息组，从而大大简化任务。

需求模型被用于整个项目的生命周期。它们有助于分析需求，在与利益相关方的会谈中征询需求，向利益相关方验证需求，并与开发和测试人员沟通需求。

为什么不用 UML？

有人肯定会问，为什么不用统一建模语言（Unified Modeling Language，UML）？UML 是对软件系统的设计进行可视化的一种语言（Object Management Group 2007）。UML 是需求建模的合理基础，但它对需求建模来说是不完整的，因为它缺少将需求与业务价值联系起来的一些模型，也缺少从最终用户的角度呈现系统的一些模型。此外，它的技术根基使其对于业务利益相关方来说过于复杂，因其建模的是软件架构。最后，UML 旨在描述系统的技术设计和架构，充其量是对模型需求的一种改造。相反，需求的重点是业务收益（business benefit）、用户操作（user actions）和业务规则（business rules）。

当模型只关注解决方案的一个或两个方面时，它们是最有用的。如果一个模型中的信息类型太多，或者语法太复杂，难以理解，那么业务利益相关方根本不会使用它。事实上，我们的经验表明，模型的复杂性是大型组织没有采用一些现有建模语言的主要原因之一。

RML 模型是用最简单的语法设计的，但仍然允许模型传达需要的信息。RML 的意图是提供一致的语法和语义结构，供业务利益相关方分析和理解项目中的模型。该语言被设计成对整个团队来说易于学习和使用，其中包括但不限于业务利益相关方。

开发人员和测试人员。这些模型被简化为最基本的符号和格式，以实现需求空间内的预期

结果。RML 不限定某种软件开发方法，可以很容易地调整以适应任何开发方法或工具集。

需求与设计

许多 RML 模型都进入了分析师传统上认为是设计的领域。例如，显示-操作-响应（Display-Action-Response）模型使用线框（wireframes）或屏幕模拟（screen mockups）来描述用户如何与屏幕元素交互，而用户界面流程（User Interface Flow）显示用户如何在各种用户界面之间导航。

关于需求，人们常说一句话："需求是指需要构建*什么*（what），而设计是指它将*如何*工作（how）"。需求和设计的这个区别非常为重要，因为很多人认为，任何属于设计的东西都不应该和需求放在一起，设计文档也不应该由业务分析师来写。遗憾的是，这种严格的定义存在一个问题："一个层级的需求就是另一个层级的设计"。

一个层级的需求是另一个层级的设计

在自上而下解决方案的任何一个层级（level），如果将一个层级视为"什么"，那么向下一个层级，就是"如何"。因此，使用上述"什么"和"如何"定义，一个层级是需求，下一层级就是设计。

例如，某利益相关方可能提出一项需求，想要降低公司网站的购物车放弃率。在下一个细节层级，产品经理可能提出几个不同的解决方案来降低购物车放弃率。例如，团队可以减少结账步骤，可以提供保存购物车以便以后购买的选项，或者可以提供免费送货。每一个建议的解决方案都是对"什么"/"需求"的"如何"/"设计"回答，目的是降低购物车放弃率。此外，"修改系统以降低购物车放弃率"虽然是一个"什么"，但它本身也可能是上一个层级——即提高整个网站的转换率——的"如何"。

用"什么"和"如何"来区分需求和设计，这是一种糟糕的方式。

确定实际的业务需求

另一种常见的说法是，任何定义实际解决方案的东西——例如所用的算法、外观和感觉或者用户界面元素——都是设计，不属于需求。但在某些情况下，一个特定的要求有时是需求，有时则是设计。例如，在某些行业中，一个产品必须使用专门的加密算法才能有竞争力；因此，它是一个需求。而在一个不同的应用中，并不要求某一种专门的加密算法，唯一重要的是对信用卡号进行加密，任何算法都可以。

作为需求的东西和不作为需求的东西之间的关键区别在于，业务利益相关方是否真的需要它。我们都知道，利益相关方并不是真的需要他们声称需要的一切，所以你的角色是针对一个特定的要求，确定它是否真的是一个需求，即他们是否真的需要它。

定义需求

需求（requirement）是业务利益相关方（the business）需要在解决方案中实现的任何东西。因此，需求可能包括功能需求、非功能需求、业务规则甚至许多人在传统意义上称为设计的东西。我们不是直接告诉业务利益相关方他们能指定什么类型的东西。相反，我们使用模型来帮助他们真正理解其需求，从而专注于为其提供服务。

本节定义了一些将在全书使用的需求术语。**功能需求**（functional requirement）是解决方案在不考虑任何限定条件下可以提供的行为或能力。**业务规则**（business rule）是一种需求，代表对功能需求进行修饰的条件陈述（conditional statement），其中包括但不限于功能何时可用以及谁被允许执行该功能。业务规则包含了诸如“如果”（if），“当”（when）和“那么”（then）等词。**非功能需求**（non-functional requirement）是除功能需求（包括业务规则）之外的其他所有需求。**功能**（feature）是对解决方案最终包括进来以满足业务目标的功能区域（area of functionality）的一个简短描述。功能是需求的集合，用于阐述（articulate）和组织（organize）需求。表 1-1 展示了几个例子。

表 1-1 需求实例

需求	**类型**
系统应能自动批准或拒绝贷款	功能需求
当信用分高于 750 时，系统应自动批准贷款	业务规则
当信用分低于 750 时，系统应使用以下算法来自动确定是否批准贷款：[*在这里列出算法*]	业务规则
审批结果应在 30 秒内返回给用户	非功能需求

假设（assumption）是一种被视为真理的陈述，决策需在此基础上做出。在假设中，包括对未来的任何预测（predictions）或预报（forecasts）。假设是对需求来说至关重要的一个话题，因为它们经常被提出，但很少被理解或阐述。事实上，当分析师被要求写下他们的假设时，他们通常会写下一些没有实质影响的琐碎假设，而忽略重要的假设。下面列出了一些示例假设，如果它们被证明不正确，那么可能导致业务目标的失败：

- 许多人都愿意在网上搜索以解决他们遇到的技术问题。
- 50%遇到技术问题的人都愿意等待后续跟进（follow up）。
- 企业 90%的客户都能上网。

- 需解决的问题都能由客户自己解决。

需求模型不是尾声

虽然使用了需求模型，但不是说完全就用不着写需求陈述（requirement statements）了。模型提供了上下文，并创建了需求的全貌。但是，它们并不代表由系统开发人员和测试人员使用的最终需求。所以，需要采取额外的步骤从模型中推导（derive）出需求。就像一个按走道来组织的购物清单一样，需求构成了团队开发解决方案时的一个核对清单（checklist）。模型的价值在于以一种特别的方式对需求进行组织，使我们很容易看出缺失、不相干或不正确的需求。

应将自己创建的所有模型作为项目的完整需求工件（requirements artifacts）的一部分。但是，只有结合文本和可视化需求，才能描绘出需要构建的解决方案的全貌（Wiegers 2013）。

在项目中使用 RML

可将本书描述的 RML 模型看成是软件项目所用模型和模板的一个工具箱。多个模型通常应该一起使用，而且有一些通用的方式来定义在整个开发生命周期中何时应该使用特定的模型。在项目中应用需求模型的方式适用于许多开发方法，例如敏捷、迭代和瀑布方法（参见第 25 章）。

其他资源

- “RML Quick Reference for Business Analysts”是一篇两页的模型总结：http://www.seilevel.com/wp-content/uploads/RML-Language-for-Modeling-Software-Requirements.pdf。
- Karl Wiegers 在《软件需求，第三版》（Wiegers 2013）一书的第 11 章对模型的价值进行了深人浅出的解释。

参考资料

- Chen, Anthony 2010,“What vs How – BRD vs User Requirements vs Functional Requirements”：https://tinyurl.com/4jmh2r7r
- Cowan, Nelson 2001,“The Magical Number 4 in Short-Term Memory: A Reconsideration of Mental Storage Capacity”, Behavioral and Brain Sciences 24, 87-114 Ellis, Keith 2008,“Business Analysis Benchmark Report”，IAG Consulting：https://tinyurl.com/2anz6vuc
- Gottesdiener, Ellen 2002, *Requirements by Collaboration: Workshops for Defining Needs*, Boston, MA: Addison-Wesley Professional.

- Miller, George A 1956, “The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information”, *Psychological Review* 63, 81-97.
- Object Management Group. 2007, “OMG Unified Modeling Language Specification”: http://www.uml.org/#UML2.0
- The Standish Group 2009, “CHAOS Summary 2009”, West Yarmouth, MA: The Standish Group International, Inc.
- Wiegers, Karl E. 2013, *Software Requirements, Third Edition*. Redmond, WA: Microsoft Press.

第 2 章　模型分类

想象一下，你需要在胶合板上切一个圆孔。你有一个工具架，上面摆满了可供使用的工具。你会很快会将搜索范围缩小到那些能够切割的工具上。例如，你会立即放弃锤子、锉刀和螺丝刀等，而是将注意力集中在各种类型的切割工具上，例如剪刀、铁皮剪、电钻、铣刀、粗木锯、曲线锯和手锯等。然后，你会从中选择能以最简单的方式完成圆形切割的一种工具。有的工具可能需要更多的安装，因为它们使用空气压缩机而不是电池或电源插座。这里的重点在于，你已经按类型和用途对自己的工具进行了“分类”。

可以将同样的分类概念应用于需求模型，以帮助我们为特定类型的分析选择合适的模型。RML 模型被组织为目标模型（objectives models）、人员模型（people models）、系统模型（systems models）和数据模型（data models）等类别，统称为 OPSD。如图 2-1 所示，RML 分类为解决方案的分析提供了一个完整的模型工具箱。综合运用这些 RML 模型，我们可以了解解决方案的*目标*、使用解决方案的*人员*、*系统*本身以及所处理的*数据*。这些模型对分析过程进行了约束，使我们尽可能不会错过关键需求，同时避免包括不必要的需求。

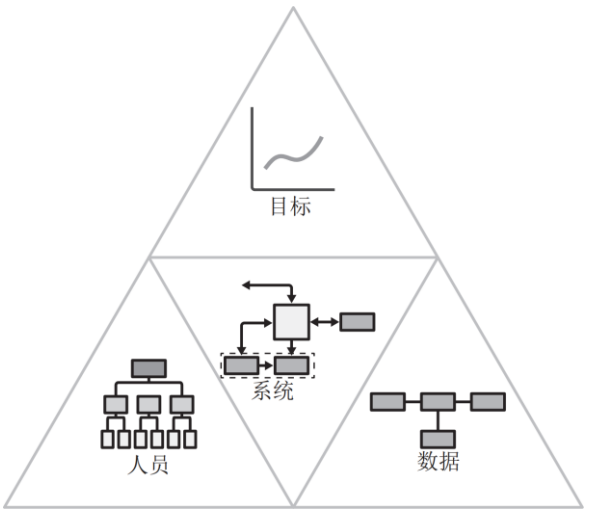


图 2-1 RML 模型的 OPSD 分类

目标、人员、系统和数据(OPSD)模型

所有软件都要处理数据。简单地说，数据进入系统，得到处理，然后退出系统。最早的软件开发模型（例如流程图和结构化设计）采用的就是这种以设计为中心的视角（DeMarco 1978）。这一视角还考虑了多系统环境中各个系统如何相互传输数据。过去 25 年，解决方案团队发现另一个视角也很重要，那就是最终用户的视角。基于这一发现，出现了用例

（Use Cases）、业务过程建模和其他形式以用户为中心的建模。最后（也是最近以来），管理层利益相关方（executive stakeholders）一直在尝试确定如何使软件开发与最终用户和组织的业务目标保持一致。

RML 的组织结构以这些传统的模型领域为基础，按目标、人员、系统和数据（OPSD）对模型进行分组。这些领域代表了为了全盘分析自己的解决方案而需考虑的 4 类信息。RML 模型通过确保信息的完整性，帮助你界定解决方案的范围。

表 2-1 展示了按 OPSD 组织的 RML 模型。

表 2-1 RML 模型分类

	描述	**模型**	**界定模型 (bounding model)**
目标	描述系统的业务价值，根据价值来确定功能和需求的优先级	业务目标模型（Business Objectives Model） 目标链（Objective Chain） 关键绩效指标模型（Key Performance Indicator Model） 功能树（Feature Tree） 需求映射矩阵（Requirements Mapping Matrix）	“业务目标模型”界定了目标空间
人员	描述谁使用这个系统，以及他们的业务过程和目标	组织结构图（Org Chart） 过程流程（Process Flow） 用例（Use Case） 角色和权限矩阵（Roles and Permissions Matrix）	“组织结构图”界定了人员空间
系统	描述存在哪些系统，用户界面是什么样子，系统如何交互，以及系统的行为方式	生态系统图（Ecosystem Map） 系统流程（System Flow） 用户界面流程（User Interface Flow） 显示－操作－响应（Display-Action-	“生态系统图”界定了系统空间

		Response） 决策表（Decision Table） 决策树（Decision Tree） 系统接口表（System Interface Table）	
数据	从最终用户的角度描述业务数据对象之间的关系、数据的生命周期以及如何在报告中用这些数据来做决策	业务数据图（Business Data Diagram） 数据流图（Data Flow Diagram） 数据字典（Data Dictionary） 状态表（State Table） 状态图（State Diagram） 报告表（Report Table）	“业务数据图”界定了数据空间

界定模型

每一类 RML 都有一个界定模型（bounding model），它有很高的概率能采集该模型所针对的全部信息。如果有很强的信心认为一个模型是完整的，就可以说完全界定了你的分析。例如，对于一些 IT 系统，可以从一个企业组织结构图（corporate organizational chart）中识别出所有可能的利益相关方。利用这些信息，我们可以创建一个名为“组织结构图”（Org Chart）的解决方案。因此，我们可以确定要与之会谈的所有相关利益相关方的一份完整名单。通过创建包含所有可能的利益相关方组别的一个“组织结构图”，我们界定了自己的分析，可以确信不会遗漏任何利益相关方。如表 2-1 所示，每一类的 RML 界定模型分别是业务目标模型（目标）、组织结构图（人员）、生态系统图（系统）和业务数据图（数据）。

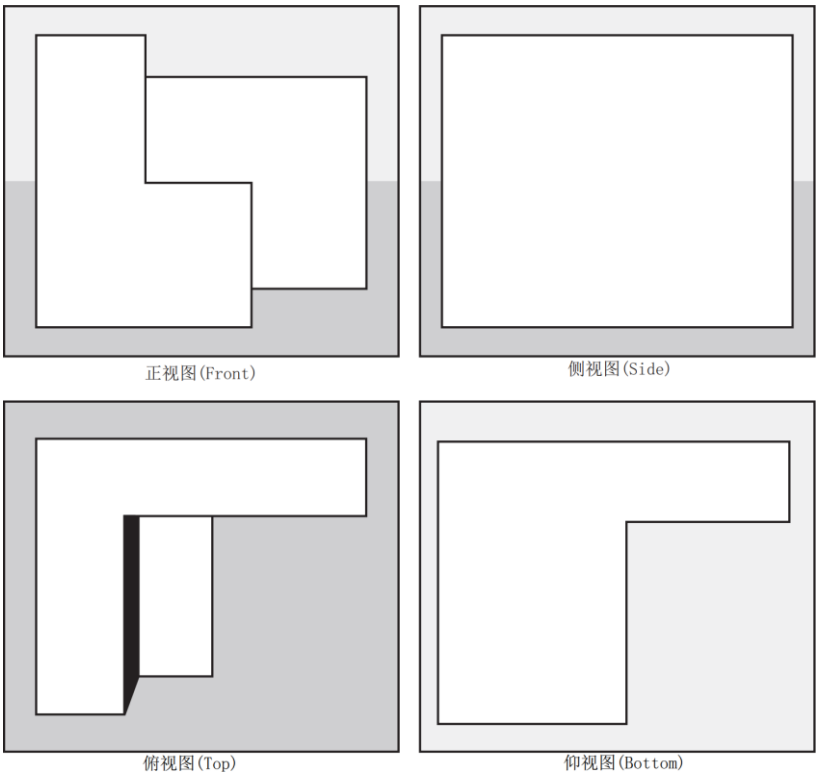
使用每一类的界定模型，我们可以建立一个完整的信息基础来界定分析范围。当分析进行到 RML 内更详细的模型时，判断该模型是否完整会变得越来越困难。例如，在深入了解个别用户的“过程流程”（Process Flow）[1]时，很难确保已经识别了用户完成的所有任务。综合使用不同的模型有助于填补这方面的空白（参见第 26 章）。

[1] 译注：“过程流程”是指一个过程的具体流程。一个“过程”涉及多项活动，这些活动具有离散性、整体性、时间性和逻辑性等特点。将所有这些活动组织起来，就形成了“流程”。详情参见第 9 章。

全部四个类别都需要

大多数解决方案在进行分析的时候，通常需要来自全部四个类别的模型。从不同角度分析解决方案的核心价值在于，它有助于确保从所有角度对解决方案进行检验。如图 2-2 所示，以一个物体的四个视图为例。在只有其中一个、两个或三个视图的情况下，不可能正确画出这个三维物体。只有在拿到全部四个视图之后，才能正确可视化物体的全貌，如图 2-3 所示。

说到软件，一些组织试图完全使用“用例”来分析一个系统，但这往往会遗漏有关数据及其在系统中如何流动的关键信息。在软件开发的早期岁月，流程图（flow charts）、数据流图（data flow diagrams）和环境图（context diagrams）[①]是软件分析的主要机制，它们生成的系统往往难以使用。在如今的软件开发中，一个常见的问题在于，由于没有对各种功能进行整体的价值分析，所以会浪费不少金钱来创建只有极少数用户才需要的功能——收益与成本不成正比。通过开发 RML 每一类别的模型，可以界定解决方案，并最大限度提高团队构建正确软件的机率。第 25 章将进一步讨论如何根据项目特征、项目阶段、受众和开发方法，从每个类别中选择合适的模型。



[①] 译注：也称为系统关系图。

图 2-2 物体的四个视图

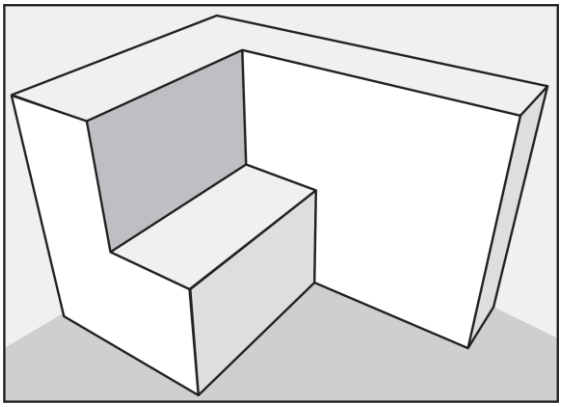
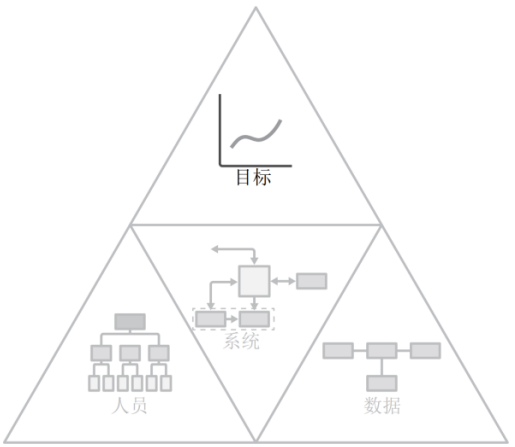


图 2-3 完整的三维物体

“目标”模型



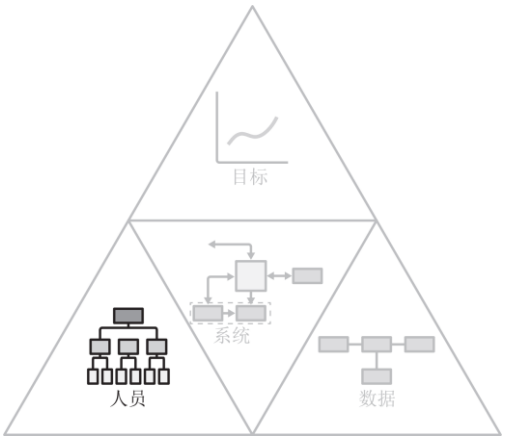
“目标”模型（objectives models）描述系统的业务价值，帮助你根据价值来确定功能和需求的优先级。项目浪费最严重的一个地方就是加入了对最终用户或组织没什么价值的功能。据统计，大约 65%的功能很少或从不使用（The Standish Group 2009）。如果这些功能在实现之前就被砍掉，那么将直接缩小项目的范围、风险和预算。只要功能没有开发，就相当于帮项目省钱了！

业务目标模型（Business Objectives Model）是 RML“目标”模型类别的界定模型。它使你能将功能映射到业务目标，以帮助利益相关方根据价值来确定功能的优先级，并砍掉价值最低的功能。另外，业务目标模型确定了需要解决的问题，以确保在实现过程中，开发团队不仅实现了功能，还解决了关键问题。

目标模型在项目的早期很有用，可以帮助项目利益相关方就项目的业务价值达成一致。项目进行期间中，它们能识别那些没有被需求满足的业务目标，从而引出事先没有考虑到的需求。另外，它们能识别那些不能创造显著价值的需求，从而帮助你缩小范围。使用业务目标模型，很容易将之前请求的许多功能或需求排除在范围之外，从而只关注那些有助于实现业务价值的需求。

通过开发业务目标模型，相当于围绕着系统范围创建了一个边界。其他目标模型将帮助你进一步细化范围，在需求和它们所创造的价值之间建立更细化的联系。

“人员”模型



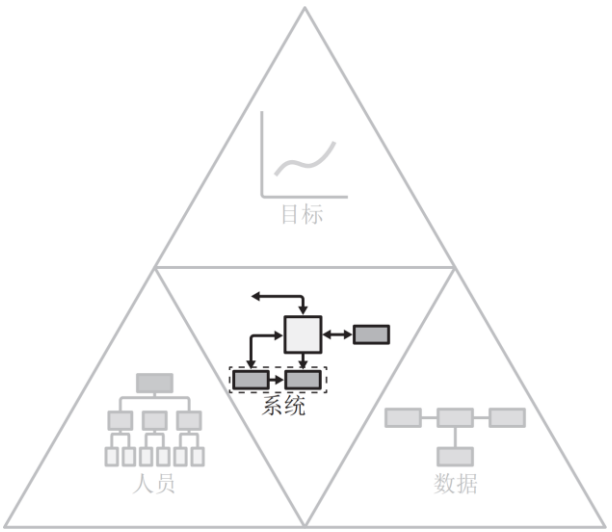
“人员”模型描述了系统的利益相关方、他们的业务过程以及他们的目标。组织结构图（Org Chart）是 RML“人员”模型类别的界定模型。它确保你考虑到了所有可能的利益相关方，从而对人员空间进行界定。一般来说，识别业务组织中的所有利益相关方是一项简单的任务，因为人们习惯于考虑与之共事的人。但是，有的业务团体（business group）并不是那么明显。组织结构图不仅包括使用系统的人，还包括运营、构建以及维护系统的人。可在每个团队中确定一个有代表性的领导，将他/她作为单一的决策者来代表用户群体，而且他/她能帮助确定那些在系统的某个方面有特殊知识的人。除此之外，还要确保识别了任何子群组（subgroups）。例如，如果有两种类型的销售代表以不同的方式使用系统，那么应将他们识别为不同的用户。最后，如果最终客户（他们显然不存在于你的组织结构图中）要使用系统，那么可以使用组织结构图来识别组织内部的利益相关方，他们直接与这些最终客户交互，因此最可能了解那些客户的需求。

对用户群体进行界定的其他方法包括查看有权访问系统的所有用户的一个数据库（其中包含他们常用的系统功能），确定将代码签入变更控制系统的所有开发人员，以及负责系统维护的所有运营团队成员。

通过确定利益相关方群体，我们有了一个牢靠的边界来限制未知因素。你可能会确定更多

的用户类别，但随着时间的推移，“人员”模型会趋于稳定。下一步是与各种用户见面，确定他们如何与系统交互。这些访谈将导致生成更多的人员模型，以描述用户期望如何使用系统以及他们期望的结果。

“系统”模型

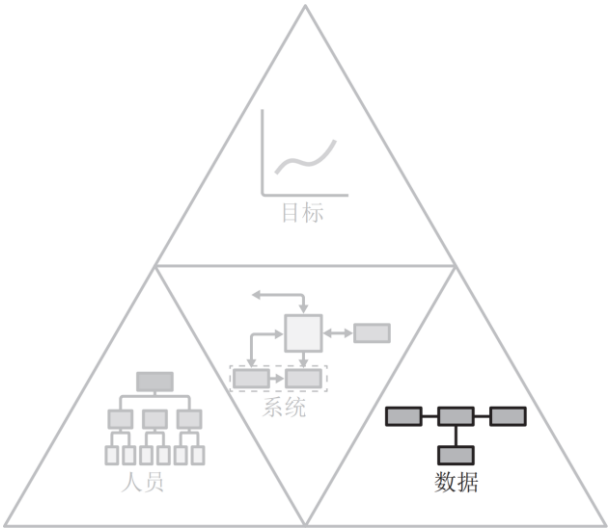


“系统”模型描述存在哪些系统，用户界面是什么样子，系统和系统之间如何交互，以及它们的行为方式。它们确定了参与多系统环境（所谓的“生态系统”）的所有主要应用程序，并描述了系统的具体方面，例如用户界面、系统间的接口以及系统执行的自动化过程等。生态系统图（Ecosystem Map）是 RML“系统”模型类别的界定模型。大多数 IT、软件即服务（SaaS）、云甚至移动软件项目都存在于一个应用程序生态系统的背景下。这个生态系统由其他无数系统和过程组成，它们要与被更改的系统交互。无论是为了一个新的开发项目，为了升级一个现有的系统，或是为了一次完整的系统迁移，当你首次审视一个现有的应用程序生态系统时，有时很难知道从哪里开始分析。更糟的是，现有的应用程序生态系统通常根本没有提供文档。

对于用户、运营人员和维护人员来说，列出涉及的系统通常并不难。然而，如果没有一些有组织的结构，那么要他们详细描述系统之间的所有交互会非常难。生态系统图提供了一个可视化的框架，它专注于任何两个系统之间的关系。因此，每次只需考虑这两个系统之间的关系即可。这使我们能够轻松捕捉关于这种关系的所有信息。生态系统图同时显示了所有系统，并在高层次上显示了交互的性质。这个包含所有系统的清单提供了另一个相当容易实现的边界。

确定一个相对完整的生态系统图，相当于界定了所有能影响你的开发工作的系统的范围。然后，可以使用额外的系统模型来描述这些系统应该是什么样子的，它们应该如何表现，以及它们之间应该如何交互的细节。

“数据”模型



“数据”模型从最终用户的角度来描述业务数据对象的关系、数据的生命周期以及如何在报告中利用这些数据来做决策。业务数据图是 RML“数据”模型类别的界定模型。业务数据图记录了全套业务数据对象及其层次结构。可以使用现有的数据输入屏幕和报告来充分识别业务数据对象。即使没有现有的屏幕，现有的表格和纸质报告也能达到同样的目的。因为这些屏幕和报告的数量是有限的，所以可以确信它们围绕所提议的解决方案的数据形成了一个完整的边界。

有了合理的业务数据图之后，就可以着手创建更多的数据模型，描述更多的数据处理细节、数据的确切形式以及用户如何使用数据来做决策。数据只能创建、更新、使用、移动、复制或销毁。所以，在确定了一个对象之后，可将这些操作应用于业务数据对象，从而以一种系统化的方式来确定需求，帮助确定用户具体如何与数据进行交互。

参考资料

- DeMarco, Tom， 1978. *Structured Analysis and System Specification.*New York, NY: Yourdon Inc.
- The Standish Group， 2009，“CHAOS Summary 2009”， West Yarmouth, MA: The Standish Group International, Inc.

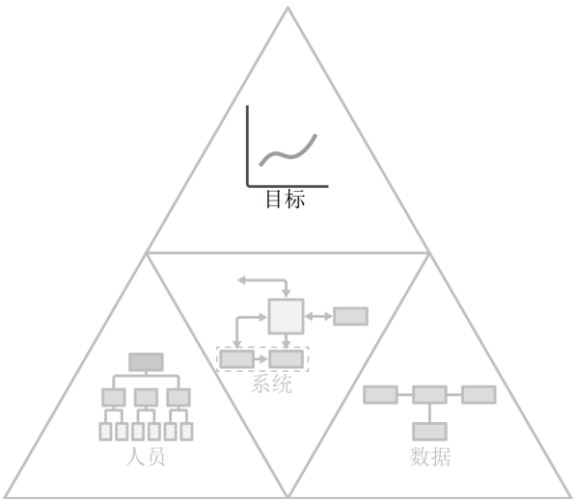
第 2 部分 “目标”模型

第 3 章 业务目标模型

某大型经销商每年向客户提供近 3 亿美元的返点。返点的计算是完全手动的，由一个海外团队执行。财务团队估计，每年会发生约 1500 万美元的错误返点，但无法确定准确数额。几笔高达 150 万美元的错误返点终于说服管理层启动一个项目来减少错误。之所以发生这些错误，主要是因为销售合同在规定返点的时候，允许客户就同一笔购买多次申请返点。这种重叠有时只是日期范围的错误，有时则是销售团队故意为之。为了纠正这个问题，管理层启动了一个项目，将这一过程自动化，并验证销售合同中的返点是否符合公司政策。

在对公司的一些客户进行为期六个月的新系统试点期间，该系统发现了大约 400 万美元的会计错误，这意味着新系统全年的回报可能超过 1500 万美元。项目耗资约 100 万美元。遗憾的是，该项目没有完全实现合同从销售系统到财务系统的自动化传输送；合同还是必须手动送到，需要三名全职人员，每个人每年的人力成本约为 25000 美元。结果，在过程完全自动化之前，业务团队拒绝全面部署。所以，新系统每年 1500 万美元的回报被延后了！

每个项目的核心都是它为最终用户或公司带来的价值。但很多时候，团队忽视了一个项目应该创造的本来价值。不通过对业务目标的关注来驱动项目的范围，项目就可能因为一些无关紧要的功能而不堪重负，这些功能对项目的核心目标没有什么帮助。利用业务目标模型（Business Objectives Model），利益相关方可以确定项目的价值，而且平时根据这个价值来做出需求决策。



业务目标模型是众多 RML“目标”模型之一。它定义了业务问题（business problems）、

业务目标（business objectives）、产品概念（product concept）和成功指标（success metrics），如表 3-1 所示。

表 3-1 业务目标模型包含的元素

元素	**定义**
业务问题	阻碍业务利益相关方达成其目标的问题。
业务目标	指出业务问题何时得到解决的一种可度量的目标。
产品概念	业务利益相关方为了达成其业务目标而选择实现的实际解决方案的愿景。它通常由包含高层次功能的一个清单来描述。
成功指标	要实际进行度量以确定项目是否成功的一个业务目标，或者与解决方案相关的其他指标。

图 3-1 展示了业务目标模型的各种元素，它们彼此相连。

图 3-1 业务目标模型元素

业务利益相关方考虑的可能是他们想要达成的目标（goals）。虽然这些目标不是业务目标模型的一部分，但理解它们并和业务利益相关方讨论是有帮助的。目标（goals）与业务目标（business objectives）相似，但前者是一种定性的陈述，而不是一种可度量的陈述[①]。

[①] 译注：goal 更一般化，是定性陈述。objective 更具体，是定量陈述。例如，我决定减肥（goal），明天起每天锻炼半小时（objective）。

业务目标模型模板

“业务目标模型”模板是一种框图，每个框都包含一个业务问题、业务目标或者带有功能描述的产品概念，并用箭头线来表示它们之间如何链接。“成功指标”要么和业务目标在同一个框中，要么单独标注。每个项目只能有一个“产品概念”框，但业务目标模型的其他所有元素都可以有多个。针对后面这种情况，应该使用多个元素框，每个框只包含元素的一个实例，同时为同类型的元素添加唯一的编号。例如，图 3-2 展示了一个业务问题有多个业务目标的情况。

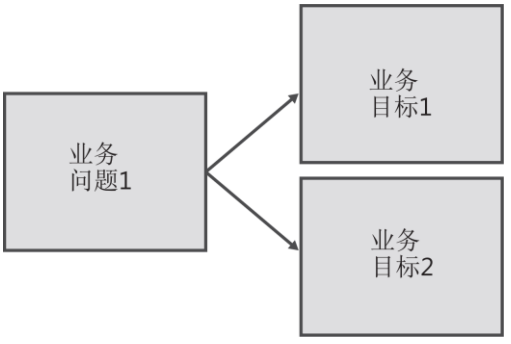


图 3-2 业务问题有多个业务目标的一个例子

业务问题和业务目标形成了一个层次结构，其中包含重复的“问题/目标”对。业务目标模型总是从一个业务问题开始，然后至少有一个业务目标。每个业务目标可能引出另一个业务问题，也可能引出一个产品概念。

成功指标可以是任何一个能度量的业务目标，它代表项目成功与否。这种指标在业务目标框中用“(SM)”来标注，即“Success Metrics”的缩写。理想情况下，所有业务目标同时也是成功指标。但遗憾的是，实情并非总是如此。因为对于业务目标的达成，解决方案所产生的影响往往难以直接度量。所以，这些业务目标不能被用作项目的成功指标。另外，可能有一些成功指标不属于业务目标，而是作为业务目标的一种可度量的代理（measurable proxies）。那些不是业务目标的成功指标，应作为标注记录在其代理的业务目标上[①]。

最终完成的业务目标模型可能不只一个层级，如图 3-3 所示。在本例中，有两个业务问题映射到三个不同的业务目标，这些业务目标被映射到四个业务问题，这些业务问题被映射

[①] 译注：标注是对一个框的补充说明，英文称为 callouts，参见图 3-3 的两个成功指标。

到四个业务目标。基于这些业务目标，我们就有了一个产品概念。通常，为了方便阅读，模型应从左向右排列。

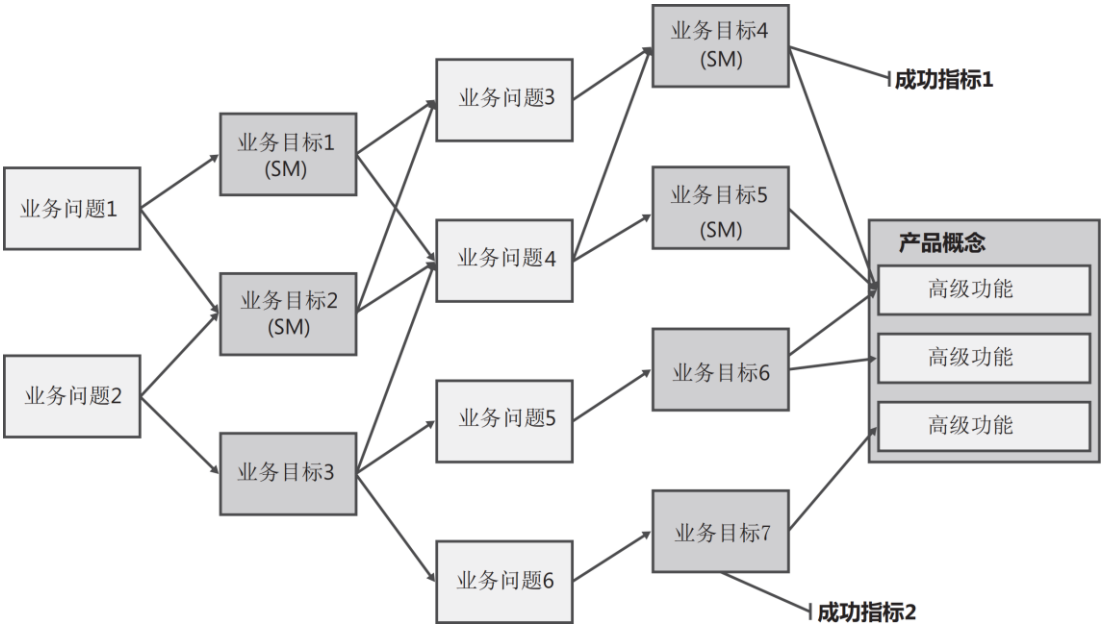


图 3-3 “业务目标模型”模板的一个例子

注意，一个产品概念可能跨越解决方案的多个组件，例如多个软件、硬件或业务过程。如果认为需要描述多个产品概念，那么针对用于实现这些产品概念的不同项目，很有可能需要多个业务目标模型来描述这些项目的业务价值。

工具提示：业务目标模型通常用 Microsoft Visio 或 Microsoft PowerPoint 来创建。创建时，可考虑利用一些思维导图工具。对于大型项目，可考虑将模型放到一个需求管理工具中，以追踪各元素之间的关系。

例子

一家打印机公司的管理层正在评估他们的财务问题。他们意识到，有几条产品线的利润已经下降。经过进一步研究，分析师帮助他们认识到，这些生产线之所以出现利润问题，原因是公司增加了呼叫中心的员工，用以支持大量客户来电。分析师与业务领导合作，确定了如图 3-4 所示的业务问题。

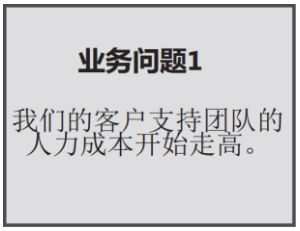


图 3-4 业务问题的一个例子

在做进一步讨论后，管理层决定，为了削减呼叫中心的开支，必须将 180 名员工调离呼叫中心，转移到能创造利润的岗位。但是，这些员工在转岗后，呼叫中心每天处理的来电数量将减少 3000 个。鉴于这个新目标，他们创造了一系列业务问题和业务目标，如图 3-5 所示。

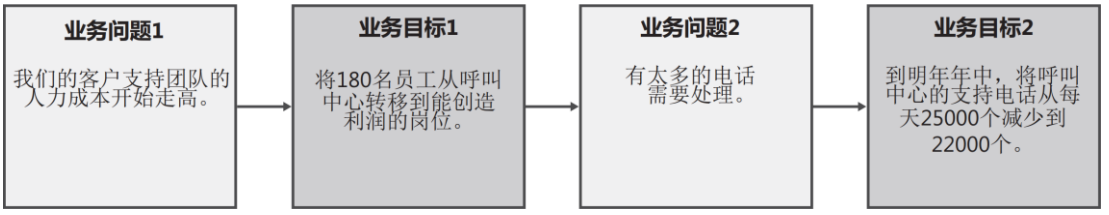


图 3-5 关联了业务问题和业务目标的一个例子

分析师提出一个问题：“当前是什么阻碍我们达成目标？”在评估了来自呼叫中心的数据后，管理层确定了人们打来电话的两个关键原因：

- 50%来电者希望找到解决其新问题的方法。
- 30%来电者之前打过电话说明了他们的问题，由于在通话过程中无法实时解决，他们又打电话来问确认情况。

这两方面的信息构成了下一层级的业务问题。如图 3-6 所示，分析师帮助管理层确定了两个业务目标。如果它们得到满足，业务问题就解决了。

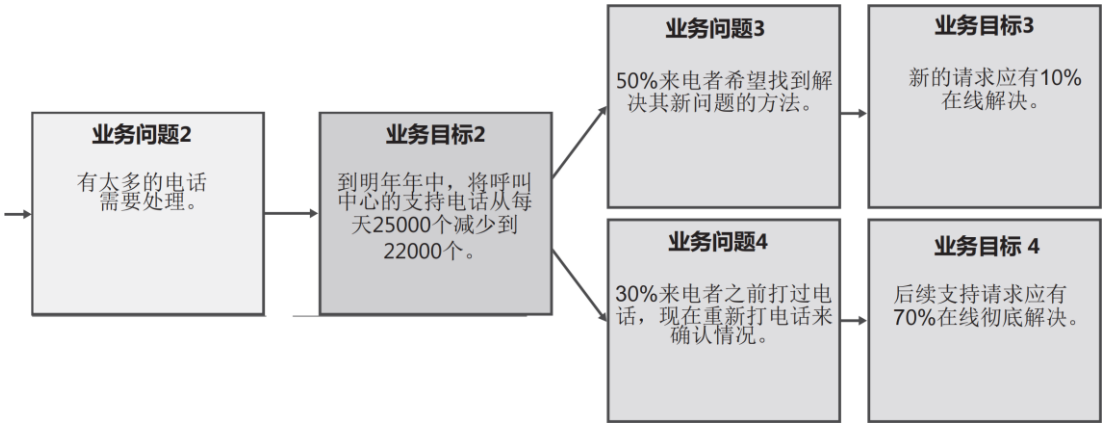


图 3-6 在业务目标模型中添加业务问题和业务目标的一个例子

然后，分析师可以提出同样的问题：“当前是什么阻碍了我们达成目标？”答案是没有在线支持系统。随后，团队就一组构成产品概念的功能达成了一致，他们认为这将减少呼叫中心接到的电话，如图 3-7 所示。他们确定了四个高级功能（high-level features），这些功能是达成业务目标的关键。注意，所有这些功能都直接包含在“产品概念”框中。

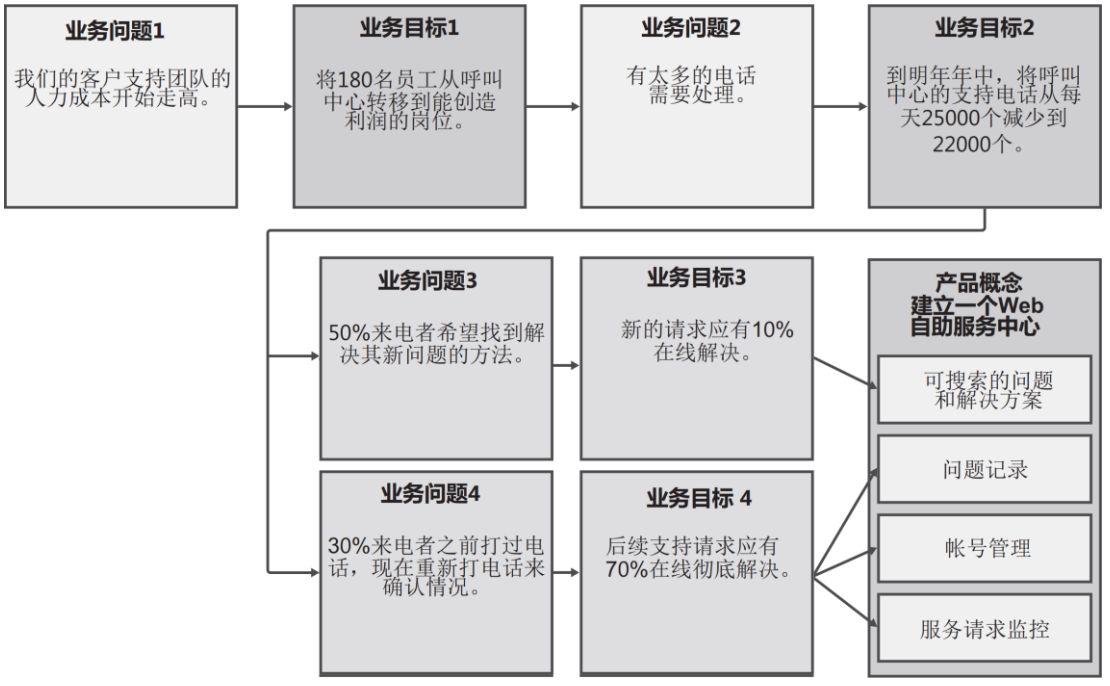


图 3-7 在业务目标模型中添加产品概念的一个例子

最后，团队确定哪些业务目标是可以实际度量的，并用这些目标来确定项目的成功指标（SM）。基于这种可度量的指标，他们可以确保项目的进展符合预期，最终实现业务目标。就当前这个例子来说，他们发现没有一个业务目标可以很容易地度量并与项目的成功联系起来。所以，他们创建了两个可以作为解决方案的一部分来度量的成功代理指标，如图 3-8 所示。

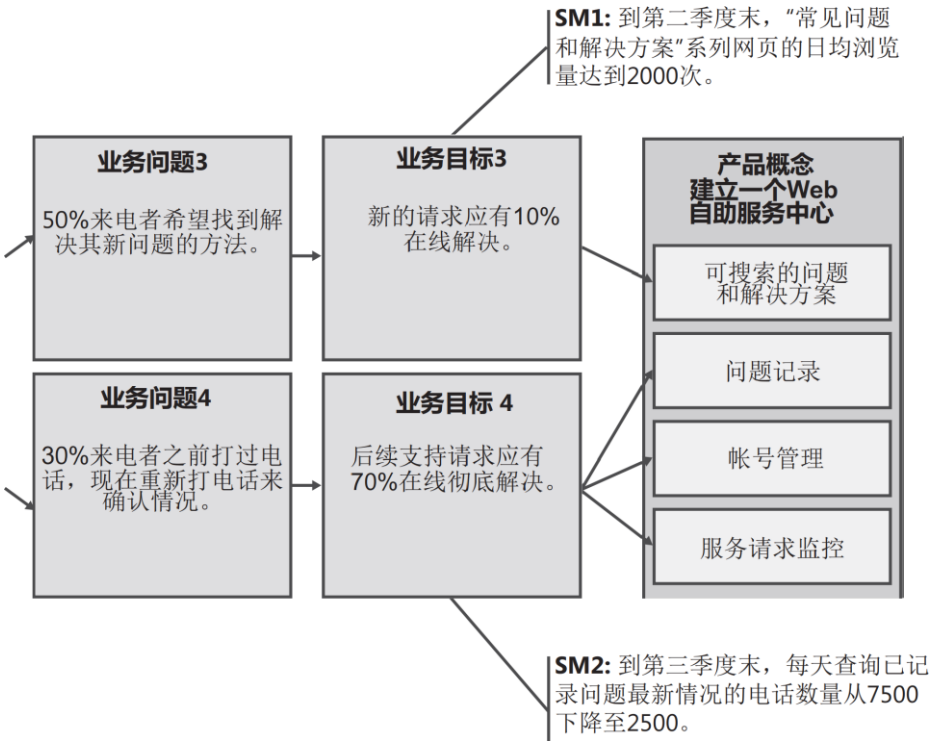


图 3-8 在业务目标模型中添加（代理）成功指标的例子

创建业务目标模型

业务目标模型中的元素以一个自然的顺序创建，如图 3-9 所示。业务问题和业务目标以迭代的方式创建，直到可以定义一个产品概念。在本章后面的“理解正在进行的项目”一节，我们将讨论在不遵循这个顺序的情况下创建和使用模型。

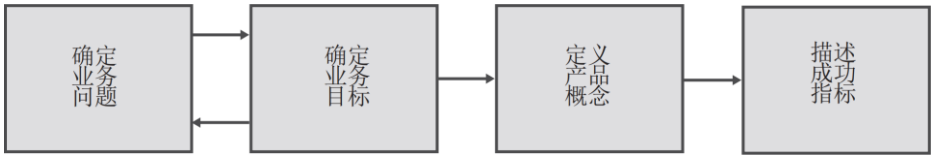


图 3-9 业务目标模型的创建过程

确定业务问题

一般来说，组织投资一个项目是为了解决特定的业务问题，即使这个问题还没有得到明确阐述（explicitly articulated）。但在许多情况下，项目的利益相关方对业务问题各自有不同的理解。好消息是，几乎所有业务问题都可以映射到收入的提高或者成本的降低。

收入需要提高

与收入有关的业务问题很少直接表述为：“我们需要赚更多的钱”。通常，企业会使用与赚更多钱有关的其他指标。例如，与客户保留（留客）、客户获取（获客）和客户消费习惯相关的问题最终都与试图提高收入的一个业务问题相关。

成本需要降低

与成本有关的业务问题可能更直接，例如，“客户支持部门成本太高”。这些问题也可能基于成本指标。成本指标（cost indicators）可能与执行某项任务所需的人数、第三方系统许可证费用、产品退货成本、监管不合规成本或者执行某项任务所需的时间有关。

许多人都认为，合规和监管问题是第三类业务目标。但是，不应将其与提高收入或降低成本分开，因为监管和合规问题与收入或成本直接相关。例如，某些类型的合规与收入相关——不合规的话，公司的产品根本不允许售卖。其他类型的合规可能与成本相关——不合规的话，公司可能会面临罚款。虽然合规与收入或成本息息相关，但对成本的影响可能是无法度量的（甚至可能是灾难性的）。因此，利益相关方经常认为监管问题胜过其他所有问题，而事实可能并非如此。

征询业务问题

业务问题和业务目标构成了一个包含重复“问题/目标”对的层次结构。在最高层级，业务问题应直接与金钱有关。在较低层级，问题则与上一个层级有关。为了征询业务问题，需要与主要利益相关方交谈，通常还需要与管理层交谈，以了解是什么在阻碍收入的提高或成本的降低。为了确定业务问题，你的起点通常是一个目标，也可能是一个产品概念（如果有人已经确定了他们想要的产品概念，但还没有分析业务需求）。然后，需要沿着这个层次结构向上走，找到一个直接与金钱相关的业务问题。或者向下走，找到与业务目标直接相关、可以帮助定义产品概念的问题。

例如，如果要进行的软件项目是为客户实现一个 Web 自助服务中心，分析师和管理层可以进行如图 3-10 所示的一次对话，以确定用于发起项目的业务问题。

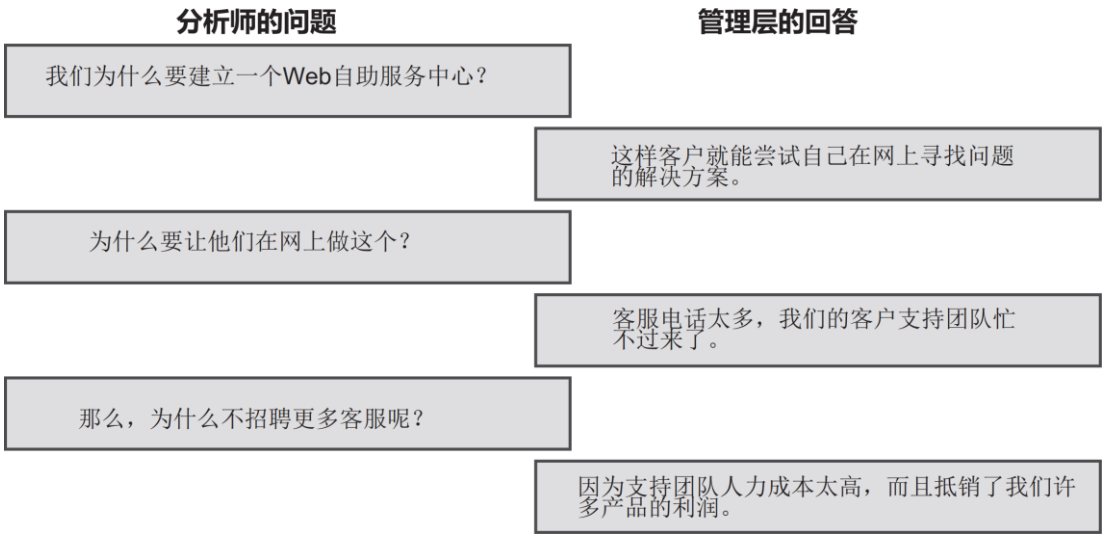


图 3-10 通过与管理层对话来确定业务问题的一个例子

在第三个回答中，分析师确定了一个新的、关于成本的业务问题。公司为客户建立自助服务中心的真正原因是为了降低支持团队的成本。这意味着，除了建立一个在线支持系统，利益相关方可以确保他们专注于开发减少来电数量的功能。自助服务中心的目标是降低成本。这似乎是显而易见的，但在执行项目时，利益相关方很容易会变得过于专注于功能，以至于忘了重点是减少来电数量以降低成本，而不是自助服务中心的功能本身。如果不保持对业务问题的关注，利益相关方往往在实现了议定的功能后，仍然没有解决业务问题。

在项目已经启动的情况下，可能有人对继续发现新的业务问题持反对意见。例如，有人会说这是在浪费时间。然而，整个团队只有在对解决的业务问题有了深刻的理解后，团队中的每个人才能在项目中做出更好的决策。这更有可能获得一个能真正解决业务问题的解决方案，而非仅仅是获得一个简单的、实现了需求的解决方案。理解业务问题还有可能促使一个组织完全重新定义一个项目，甚至在将资源浪费在错误的事情上之前及时叫停。

确定业务目标

为了确定与特定问题关联的业务目标，要问自己：“我如何确定这在当前是一个问题？如果解决了这个问题，业务会变成什么样？”业务目标必须是可度量的，而且通常有一个时间范围，在这个范围内度量才是合适的。

继续上一节的例子，在确定业务问题是支持团队人力成本过高之后，你可以拟定一个具体的目标成本或利润金额。如果无法拟定具体金额，那么考虑使用一个对成本的下降进行“代理”的指标，即从呼叫中心转岗至创收岗位的员工数量，因为员工数量直接与呼叫中心的运营成本相关。

业务目标规定了从当前数字到目标数字的变化。另外，应避免在业务目标中使用百分比来度量变化，因为人们往往会忘记百分比所适用于的原始基线数字。另外，它使人们更难真正理解目标数字。百分比变化要求读者知道、查询或假设实际数字，这导致利益相关方缺乏共同理解。相反，应使用具体的数字来表明当前水平和目标水平。在以下两个目标陈述中，第二个目标比第一个目标更容易记住，避免人们在不知不觉间试图改变项目的中期目标：

- 陈述 1：支持电话数量减少 12%。
- 陈述 2：到明年年中，将每天的支持电话数量从 25000 减少到 22000。

在确定业务目标时，分析师面临的一个挑战是，在需要对已确定的业务目标做出承诺时，人们往往会犹豫不决。很少有合理的理由需要将收入或成本目标保密。大多数时候，这种犹豫不决源于害怕对结果担责。如果真的无法就业务目标达成一致，你最起码也要尽最大努力使人们同意下一级的可度量结果——成功指标。不过，如果没有可度量的业务目标，项目很有可能无法为公司提供足够的价值，因为在需求的实现过程中，很可能偏离方向。

定义额外的问题和目标

在层次结构的顶层定义了最初的业务问题和业务目标后，可能还需要继续确定其他“问题/目标”对，最终形成一个明确的产品概念。顶层以下的每个“问题/目标”对都与解决上一层的问题有关。

为了确定额外的问题，分析师应该问：“当前是什么阻碍了我们达成业务目标？”这听起来很简单，但可能非常具有挑战性，因为它需要当前可能不没有的数据。例如，可能还没有数据说明为什么客户会频繁拨打支持电话。

在前面的例子中，之所以不能将 180 名员工立即调到其他岗位，原因是当前有太多的电话呼入，剩下的员工根本处理不过来。这个新确定的业务问题被添加到层次结构中。然后，一个新的目标被创建，以代表成功解决新问题（电话太多）的指标是什么：明年将每天的支持电话数量从 25000 减少到 22000。

在任何时候，几个潜在的问题都可能使你无法实现业务目标。例如，可能有许多可能的原因导致支持电话的数量无法减少。在本例中，公司的产品本身可能存在问题，网站可能让客户太容易打电话进来，或者手册可能太难使用。

为此，需要继续在定义问题和目标之间进行迭代，直到可以根据业务目标直接定义要实现的功能。这些功能形成了产品概念。一般来说，“问题/目标”对的层级不能太多。如果太多，就会有太多的信息无法有效地理解和使用。

定义产品概念

业务目标不会讨论具体如何实现，那是产品概念的事情。产品概念是企业为满足业务目标

而选择实现的解决方案的愿景。

产品概念包括设想的高级功能（high-level features）。产品概念可以描述一个解决方案的多个方面，包括软件、硬件和业务过程。所谓“功能”（feature），是指对解决方案最终将包含以满足业务目标的一个功能区域（area of functionality）的简短描述。功能（features）是需求的集合，用于阐述并组织需求。它们通常可以作为一个清单来交流，帮助不熟悉项目的人理解项目所提供的功能（functionality）的本质。

第 6 章将进一步讨论如何征询（eliciting）对产品概念进行定义的功能。

指导原则

有的时候，企业对需要在解决方案中关心的事情做了一般性的陈述。虽然这些不算是单独的“需求”，但在创建解决方案时仍然很重要。这些指导原则（guiding principles）为利益相关方提供了在开发业务问题的解决方案时应考虑的方向。

指导原则描述了对市场的期望或者利益相关方想要达到的目标（goals），它们适用于整个解决方案。例如，以下都可以称为指导原则：

- 开发面向用户的所有功能时，将重点放在新用户的易用性上。
- 在所有财务计算中应用“公认会计原则”（Generally Accepted Accounting Principles，GAAP）。
- 维持现有业务工作流（business workflow）。
- 更改业务过程（business process），而不是进行定制。

如果确实有现成的指导原则，那么可以把它们作为产品概念的一部分。

描述成功指标

一些业务目标的改善可以直接度量并归因于项目的成功。在这种情况下，应将这些业务目标标记为“成功指标”（Success Metrics，SM）。然而，许多业务目标（business objectives）度量的是大型的、长期的业务目标（business goals），这些目标（goals）经常受到当前开发的解决方案之外的因素的影响。和业务目标一样，成功指标在一个特定的时间范围内可以度量，但它们应用于正在开发的“产品概念”。某些情况下，成功指标就是业务目标本身。但在大多数情况下，成功指标都是一种间接指标（代理指标）。如果业务目标难以直接度量，或者会受其他许多因素的影响，以至于无法直接度量解决方案的成功与否，那么可以将成功指标作为对核心业务价值进行度量的一种“代理”。如果使用成功指标作为代理，那么需要跟踪将其视为有效代理而做出的任何假设。

为了确定成功指标，请考虑在当前实现的解决方案中，有哪些可以度量的方面促成了业务目标的实现。例如，自助服务支持系统的产品概念与“支持电话数量从 25000 减少到

22000”的业务目标关联——假设如果将信息放到网上，客户就会使用它。因此，对于这个项目，合理的成功指标是在指定时间范围内产生一定数量的浏览量，而且对于一定数量的产品，用户要有合理的“本文有帮助吗？”点击量。可能有其他因素会影响来电数量，例如新产品的推出。因此，在线帮助系统的使用人数是比呼入电话数量更准确的一个指标。

如果只是创建用于提供在线支持的功能，那么不一定能取得成功。所以，让利益相关方关注解决方案的成功指标是至关重要的，不能让他们仅仅关注产品概念的实现。就在线帮助系统而言，成功指标可能会推动其他变化，例如营销、网站文本和电子邮件宣传上的变化，以向客户推荐新的在线系统。

在阐述需求之前，首先应制定成功指标，因为某些需求能直接从这些成功指标推导得出。在确定了更详细的功能，并将其映射到目标链之后（目标链的主题请参见第 4 章），可以添加更多的成功指标。目标链中的“目标因素”（objective factors）提供了对于特定项目来说一种很好的指标，它们可能比业务目标更容易度量。

为完成业务目标模型需要提出的问题

为了完成业务目标模型的每一部分，表 3-2 推荐了应该问的一系列问题。通常，应该向管理层的发起人或者作为其代表的利益相关方提出这些问题，因为他们才是与底层业务问题联系最紧密的利益相关方。

表 3-2 帮助确定业务目标模型各个元素的一系列问题

模型中的元素	要问的问题
业务问题	阻碍组织提高收入或降低成本的关键问题是什么？ 不断问：“为什么这会成为问题？”，直至最终答案涉及到“钱”。 当前是什么阻碍我们达成目标？
业务目标	你用什么指标来判断问题已被解决？ 你期望在什么时间范围内看到结果？ 根据什么基线水平对变化进行度量？ 在这个项目外部，还有什么会影响业务目标？ 哪些指标可以用作判断业务问题是否解决的“代理”？
产品概念	必须构建或改变哪些产品或过程来达成业务目标？

	当前是什么阻碍公司达成目标？解决方案是什么？ 可以采取哪些方法来达成业务目标，从而解决业务问题？ 要解决哪些问题或者问题的哪些方面？以何种方式解决？ 为了达成业务目标，关键需求（功能）是什么？ 有哪些指导原则会对可能的解决方案或功能集产生限制？
成功指标	能否直接度量解决方案对业务目标的影响？ 如果不能，可以用什么“代理”指标来判断产品是否成功？ 利益相关方怎么知道解决方案完全实现了它预期对业务目标的贡献？

使用业务目标模型

业务目标模型应在项目的早期创建，通常应该是你创建的第一个模型。它应该在项目的整个生命周期中使用，使利益相关方一直专注于解决方案的价值。

提供对项目价值的共同理解

业务目标模型为利益相关方提供了一个框架，使他们对项目的目的和价值有一个共同的理解。该模型的结构允许在一个页面上查看项目价值，以方便他们使用。一旦利益相关方首先关注到的是业务目标模型，其他模型所描述的需求就必然是为特定业务目标提供支持的正确需求。管理层可以使用这个模型来验证他们做出的投资。

界定解决方案空间

阐述业务问题（articulation of the business problem）是在任何项目中首先要做的事情之一。一个清晰的业务目标模型可以帮助所有利益相关方理解并界定解决方案空间，确保其中的解决方案是专门用来解决问题和实现目标的。通过完成业务目标模型，可以确保整个利益相关方群体对项目要解决的业务问题达成一致。与其他模型相结合，业务目标模型可以确保只有最具价值的功能才会得到开发。这将在第 4 章进一步讨论。

理解正在进行的项目

业务目标模型是每个项目的基础。大多数项目在前期并没有主动使用业务目标来定义项目，更不用说做出日常决策了。因此，我们在设计模型时，就考虑到了要让它很好地支持已在进行的项目。

一种不会奏效的典型方法

大多数项目在开始时，相当于已经定义了一个产品概念。项目组可能会定义一些成功指标；但是，他们通常会立即跳到定义需求和设计的步骤。如图 3-11 所示，这种方法的问题在于，如果没有先就指导项目范围的业务目标达成一致，那么团队很可能会偏离业务的预期目标。

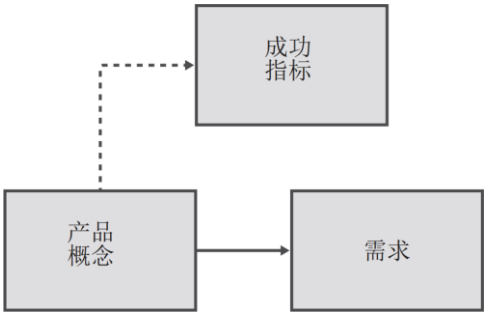


图 3-11 确定需求的典型方法

这很容易出问题。如果分析师继续在此基础上为每一个功能收集需求，那么可能出现几个问题：

- 开发出不必要的需求。如果一个需求不能映射回一个特定的可度量的业务目标，那么为什么要实现它呢？为什么要花时间来开发这种需求？
- 定义了需求后，开发团队回头可能发现，以其当前的人员和预算，根本无法实现所有需求。他们想知道哪些需求可以删除。而如果这些需求没有被映射到价值，那么很难冷静讨论可以删除哪些需求，以使项目在预算内完成。人们会争取自己喜欢的功能，而忽略对业务的量化价值。
- 即便确定了与业务目标一致的需求，构建的解决方案也可能无法达成业务目标。例如，在一家保险公司，团队认为业务目标是将 7 个 IT 系统合并成一个。但是，实际的业务目标是将 7 个业务团队合并成一个。不是 7 个具有完全不同技能的小团队，而是合并成一个具有重叠技能的大团队。只有这样，才能最大限度地减少周期性进行额外处理所需的人员。可以想象，虽然将 7 个系统合并成一个很容易，但无法实现真正的业务目标。

始于问题和目标的理想方法

如图 3-12 所示，一个理想的方法是从业务问题开始，以迭代的方式定义业务目标和其他问题，直至软件的功能可以达成业务目标。然后，使用高级功能和成功指标（high-level features and success metrics）来定义产品概念。你将使用这些功能和指标来驱动这一过程，

直到最后确定需求。这种方法之所以理想，是因为它确保最终得以开发的需求总是由业务问题的解决方案来驱动的。

这种方法的缺点在于，它不能反映组织现实的运作方式。大多数组织在考虑项目时都不会采用自上而下的方法。另外，这种方法通常不会奏效，因为分析师往往在定义业务价值（业务目标）的工作发生后很久才会参与到项目。

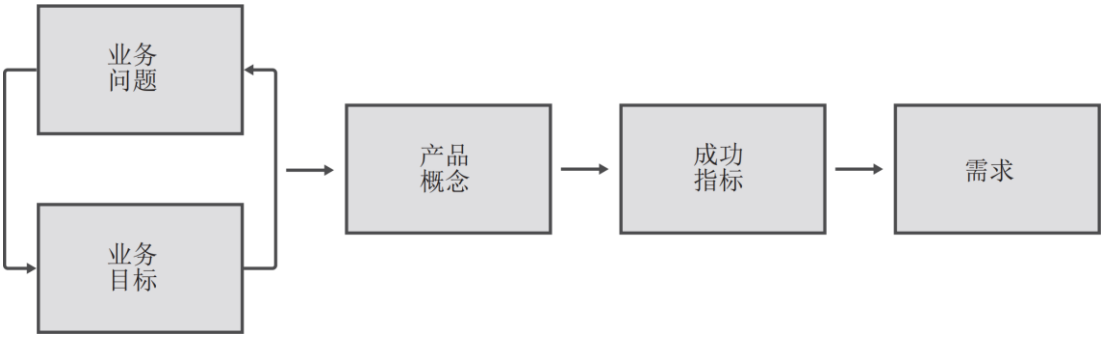


图 3-12 确定需求的理想方法

一种现实的方法

许多项目都从一个产品概念开始，而这仅仅是因为分析师那时才会参与进来。因此，前面描述的从业务问题开始的理想方法是不切实际的。不过，仍然可以创建一些元素来取得成功，如图 3-13 所示。

如果项目从一个产品概念开始，那么一组初步的需求往往已经定义好了。但是，我们完全可以从产品概念开始反方向推导，从而理解需要解决的业务问题。事实上，业务问题和目标可能是已知的，只是没有被正式记录。当分析师问：“为什么这会成为问题？”，最终就会遇到一个与创造收入或削减成本有关的业务问题。有了清晰定义的业务问题和业务目标，就可以确定，产品概念将满足业务目标并解决问题！

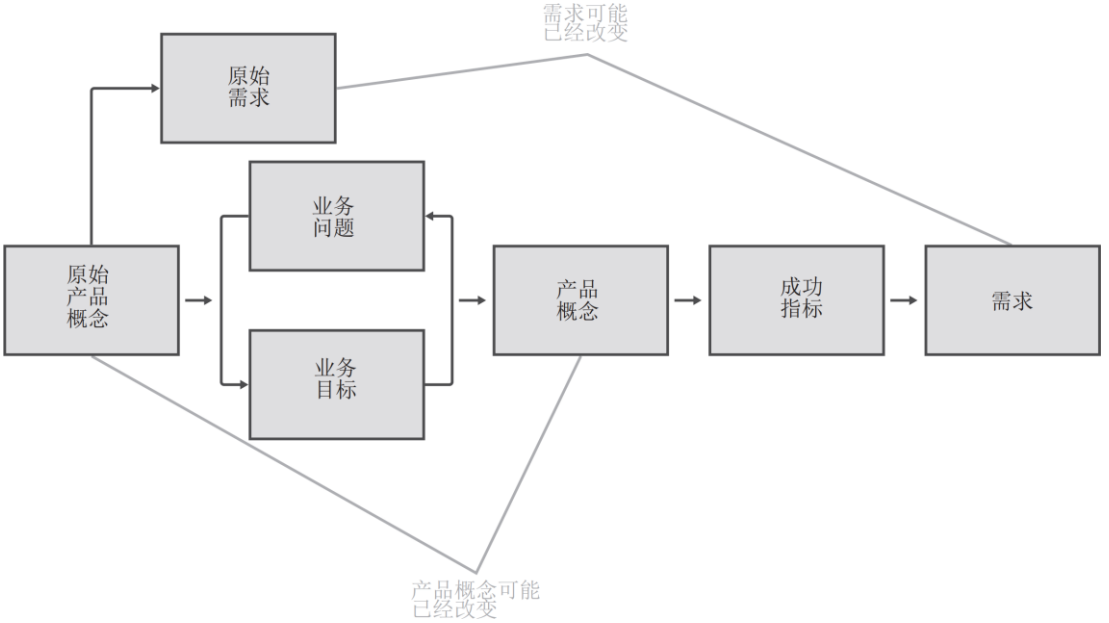


图 3-13 确定需求的现实方法

利用业务问题和业务目标，利益相关方可以定义包含高级功能（high-level features）的一个清晰的产品概念。由于对业务问题有了更好的理解，所以这个产品概念可能与原来的产品概念相同，也可能不同。然后，应该制定解决方案的成功指标。最后，可以使用业务目标模型的各种元素来确定需求的初稿。注意，相较于最初的建议稿，需求可能已经发生了变化。

分析师在拿到一个项目后，可以立即开始确定或定义业务问题/业务目标，即使当前已经定义了产品概念。对业务目标的仔细核实往往会引起产品概念的一些变化。如果可能发生这种情况，那么最好是在项目的早期发现需求并做出改变，而不是在后期。

推导需求

由于业务目标模型阐述的是高级（高层次）业务目标，所以很难直接从它推导出需求。然而，业务目标模型确实记录（document）了功能，后者相当于高级需求。我们需要基于业务问题和业务目标来确定高级功能。然后，利用这些功能，在一个需求映射矩阵（参见第 7 章）中组织详细需求。

何时适用

任何项目只要有新功能，而且这些功能可以映射到业务价值，就应该使用业务目标模型。其中包括增强功能、新的定制开发项目以及要进行大幅定制的商业现货（Commercial Off

the Shelf，COTS）部署。

何时不适用

利益相关方可能想对遗留系统进行转换，实现一个新系统来取代旧系统，而不是将新的功能映射到业务价值。他们想通过现有的业务过程来保持现有的关键绩效指标（KPI）。在这种情况下，关键绩效指标模型（Key Performance Indicator Model，KPIM）（参见第 5 章）会更有用。KPIM 有助于为现有过程流程（Process Flows）分配价值，方便利益相关方为现有功能制定优先级。

常见错误

下面列出了关于业务目标模型最常见的错误。

没有理解业务问题

如果分析师不了解一个项目背后真正的业务问题，那么业务目标和最终衍生的解决方案就很有可能无法解决问题。一个更常见的问题是，如果没有对业务问题和业务目标的清晰理解，利益相关方就没有客观的标准来确定什么时候应该从范围中删除功能。范围蔓延（scope creep）是项目超出预算或出现延误最常见的原因之一。

定义了不可度量的业务目标

业务目标可能已经确定，但却无法度量。这往往是管理层的一个错误。正如本章前面所讨论的，他们对可供度量的指标有一种恐惧感。当然，即使业务目标没有定好，但有总比没有强。不过，分析师还是应该努力定义可度量的业务目标。

在业务目标中阐述了错误的信息类型

业务目标应与金钱紧密联系。产品概念经常被错误地列为业务目标。例如，在前面介绍的保险例子中，业务目标被定为“将 7 个系统合并成一个”。但这是产品概念。真正的业务问题是，有 7 个业务团队无法完成彼此的工作。导致该问题的一个原因是，每个团队都在不同的系统上接受培训。产品概念是开发一个新系统，它将 7 个系统合并为一个。

相关模型

业务目标模型捕捉业务问题和业务目标，这和其他建模语言是相似的。不过，RML 模型非常简单易懂。另外还存在一个理性模型（rationale model），它增加了业务目标模型以外的信息，可以跟踪做出范围决策（scope decisions）的理由（rationale）。它跟踪为什么要做出一个决策，这有助于确保决策不会被不必要地重新审视（Alexander and Beus-Dukic

2009）。

另外还存在一个与目标链非常相似的概念，称为最小可销售功能（Minimum Marketable Features，MMF），它帮助分析师通过比较功能的价值来决定项目范围。

下面简要描述了影响业务目标模型或者被业务目标模型增强的一些最重要的模型。第 26 章会对所有这些相关模型进行更深入的讨论。

- **目标链（Objective Chains）**：直接使用业务目标，允许比较单独功能的价值。
- **功能树（Feature Trees）**：进一步发展出在业务目标模型中最初作为产品概念的一部分来定义的高级功能。
- **关键绩效指标模型（Key Performance Indicator Models，KPIM）**：用于代替业务目标模型。如果项目只是替换现有功能，并维持之前的 KPI，就用该模型来确定优先级。
- **需求映射矩阵（Requirements Mapping Matrices）**：用于按业务目标模型中的业务目标和功能来组织需求。

练习

以下练习帮助你更好地理解如何使用这种模型。练习是开放式的，因此你的答案可能与我们提供的答案大不相同。可能存在许多正确的解决方案。在答案中，我们对如何得出解决方案进行了解释。在看答案之前，你可以先尝试自己做一下，这样练习的收获最大。练习答案可以在附录 C 中找到。

说明

为以下场景准备一个业务目标模型。记录为创建模型而要提出的任何问题，以及期望从管理层那里得到的虚构答案。还要列出所做的任何假设。

场景

你的公司销售塑料火烈鸟和其他草坪摆件，大约有 10 万名客户，年收入为 1000 万美元。公司已经开始了一个建立网店（eStore）的项目，但你希望了解业务主管想用这个新系统解决什么问题，以帮助他们确定不同需求的优先级。

其他资源

- 《Discovering Requirements》一书讨论了如何确定业务要求（business needs），后者类似于业务目标（Alexander and Beus-Dukic 2009）。本书还归纳了理性模型和它为项目增加的价值。
- 在《More About Software Requirements》一书的第 21 章，Wiegers 将业务目标作为业

务需求的一部分讨论（Wiegers 2006）。

- “The Single Most Important Failure with Requirements”一文指出，项目失败的常见原因是它们没有业务目标，或者业务目标不佳：*https://tinyurl.com/3t9u8p4w*。
- BABOK 的 Enterprise Analysis Knowledge Area 有一项任务是定义业务要求（IIBA 2009）。
- Enterprise Business Motivation Model（EBMM）有一个和业务目标模型很像的目标；两者甚至用了同样的术语（Malik 2009）。

参考资料

- Alexander, Ian, and Ljerka Beus-Dukic，2009，*Discovering Requirements: How to Specify Products and Services*. West Sussex, England: John Wiley & Sons Ltd.
- Malik, A. Nicklas ， 2009 ， “ Enterprise Business Motivation Model ”： http://motivationmodel.com/
- Wiegers, Karl E.，2006。*More About Software Requirements: Thorny Issues and Practical Advice*. Redmond, WA: Microsoft Press.
- International Institute of Business Analysis（IIBA），2009，*A Guide to the Business Analysis Body of Knowledge*（*BABOK Guide*），Toronto, Ontario, Canada.

第 4 章 目标链

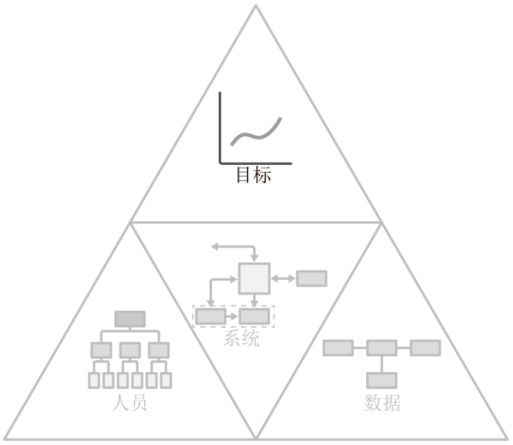
假设我去市场买一辆新车，我的主要目标是花最少的钱买一辆能满足我基本需求的车。我关心油耗、长期维修费用、维修频率、后备箱空间和安全性。为了决定哪些对我来说是最重要的，我考虑每一项对汽车整体价值的贡献有多大：

- 混动车能提供不错的里程，但这个功能将花费我 1 万美元，根据我驾驶的里程数，每年只能为我节省 700 美元。考虑到车子可能的寿命，这个功能的价值不足以证明购买混动车是合理的。然而，如果我每年开更多的里程，或者如果汽油价格上涨两倍，那么它可能是值得的。
- 研究表明，某些品牌的汽车比其他品牌的汽车需要更少的维护，因此长期维修成本更低，但这些品牌的汽车要比同类车贵 5000 美元。
- 我知道，如果买一辆 SUV，那么会有很大的后备箱和更好的越野能力。由于我喜欢户外活动，所以需要这些。另外，我还会有足够的乘客空间，可以载着我的小孩和他们的朋友四处游玩。但是，这些功能都很贵，一辆 SUV 比同品牌的非 SUV 汽车要贵 8000 美元左右。

基于这些考虑，我降低了油耗的优先级，并决定购买一辆评级良好但不是最好的品牌的 SUV。为此，我通过一个一个的陈述句将愿望（油耗、后备箱等）与目标（满足基本需求的一辆车）联系起来，说明每个愿望是如何满足目标的，而且我为每个愿望都设定了价值。最后，我知道了一辆车的哪些方面最重要，哪些方面可以忽略。

分析师经常遇到一些范围太大的项目，以目前可用的资源（包括时间、金钱和人员）无法完成。为了控制任何项目的范围，最有效的方法是将项目限制在只为用户或业务提供最大价值的那些功能上。一项研究发现，在典型的软件中，65%已开发的功能很少或从未使用（Standish Group 2009）。如果一开始就剔除那些只提供很少价值的功能，那么可以实现项目的大部分价值，同时大幅削减成本，加快推出速度。这一理念对于面向消费者的软件也是适用的。

目标链（Objective Chain）是一种 RML 目标模型，它以可度量的方式将功能与业务目标联系起来（参见第 3 章）。由于每个业务目标通常都会链接多个功能，而且每个功能可能会解决多个业务目标，所以有必要用一个模型来帮助确定每个功能的相对价值。



目标链是一个层次结构，其顶部包含业务目标，底部则包含功能。两者之间的每个层级都是一个定性的陈述句，说明了该层对链中的上一层有什么贡献。例如，在本章开头的故事中，我的目标是花最少的钱买一辆能满足基本需求的汽车，而我考虑的功能之一是混动发动机。定性地说，混动发动机增加了每加仑汽油的行驶里程数。而增加每加仑里程数能降低每英里的成本。

链中的每个层级也可以包含一个大致的计算，从而定量地解释一个功能对业务目标做出的贡献。还是那个买车的例子，厂商表示混动车每加仑可以行驶 50 英里，而油车每加仑只能行驶 25 英里。假设油价是每加仑 4 美元，我每月行驶约 1000 英里，混动车每英里的油钱是 8 美分，油车每英里的油钱则是 16 美分。因此，混动车每月的油钱是 80 美元，油车每月的油钱是 160 美元。

目标链提供了一种方法来包含对任意功能的价值的一个近似计算，即使该功能与目标的关系不大。如果为所有功能都创建目标链，就可以比较它们的相对价值，以确定应该从范围中删除哪些功能。该模型使团队能理性地辩论应如何计算功能的价值，而不是带着与功能相关的情绪去辩论（例如，因为特别喜欢某个功能，所以罔顾其成本）。

目标链模板

如图 4-1 所示，目标链是一种树状结构，由多个方框构成。左上角的方框包含了一个业务目标，而右下角的一系列方框（树的“叶子”）包含了功能。两者之间的分支由一系列包含目标因素和目标方程的方框构成。

其中，目标因素（objective factors）是层次结构中的一个层级对上一层级有什么贡献的定性陈述。目标方程（objective equations）则对目标因素之间的关系进行了量化。目标方程是在关联的目标因素框中采集的。至于功能，则用不同的颜色或阴影显示，以便和目标因

素区分。

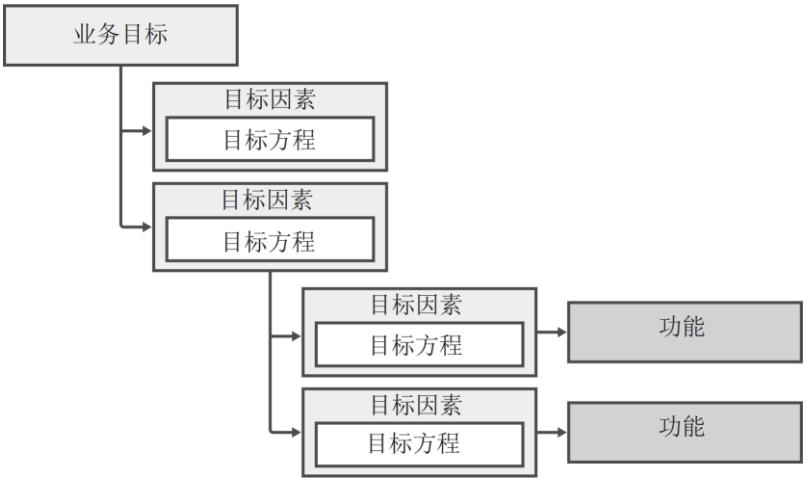


图 4-1 目标链模板

工具提示：刚开始可以使用便条和白板来创建层次结构，但创建目标链的最佳工具是某种软件，例如 Microsoft PowerPoint 或 Microsoft Visio。理想情况下，目标链应该存储在一个需求管理工具中，这样任何可追踪功能（traceability functionality）都可用来辅助维护层次结构中的链接。

例子

某销售组织了解到，相较于没有接受培训的同行，定期接受产品培训并通过了培训考试的销售代表能销售出更多的产品。该组织的业务目标是通过增加每名销售代表的销售额来增加收入。合理的推论是，培训很重要，因为它有助于销售代表了解新产品。而对这些新产品有了更多的了解后，销售团队就能更好地销售这些产品。

销售代表的考试分数（test scores）度量了他们通过培训课程学到了多少东西。销售代表通过的考试越多，他们了解的产品就越多，他们卖出的产品就越多。

例如，对于未通过某一产品考试的销售代表，其每天平均贡献的收入是 1000 美元。然而，如果该销售代表参加了培训并通过了相关考试，其贡献的平均收入就会增加到 1200 美元。这个指标可以清楚地告诉企业，拥有一支通过考试的销售队伍是有价值的。企业应该寻找机会，帮助更多的销售代表通过培训考试。

这个例子中的目标因素可以这样陈述：“增加通过考试的销售代表的人数可以增加销售额”，如图 4-2 所示。

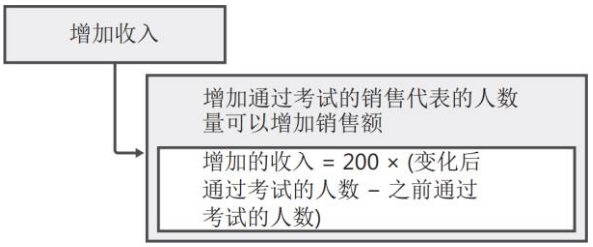


图 4-2 这个目标因素是增加通过考试的人数

我们可以想出几种功能（features）来增加通过培训考试的销售代表的人数。对这些潜在的功能进行分析，可以发现其中哪些能提供最大的价值。

一个可能的功能是提供“在线培训”。这使销售代表能在他们方便的时候参加培训，而不是每季度固定参加几次，后者是当前的培训计划。进一步的研究表明，由于销售团队的日程安排，大约 40%的员工会错过这些面对面的培训活动。在有 1000 名销售代表的情况下，通常约有 600 人能参加培训。而有了额外的在线培训后，预测表明这个数字可以增加到 900 人之多，即增加 30%的销售代表，培训的参与率达到 90%。历史数据还显示，先培训再考试能将通过率从 25%提升至 90%。这使通过考试的销售代表人数从 640 人增加到 835 人。(900 名参加培训的销售代表中的 90%，加上未参加培训的 100 名中的 25%，共有 835 名销售代表通过了考试)。此外，由于通过考试的人对收入的贡献平均会多出 20%，所以每天的收入将增加 39000 美元（在新增的 195 名通过考试的销售代表中，每人每天会创造出 200 美元的额外收入）。图 4-3 展示了这个例子。

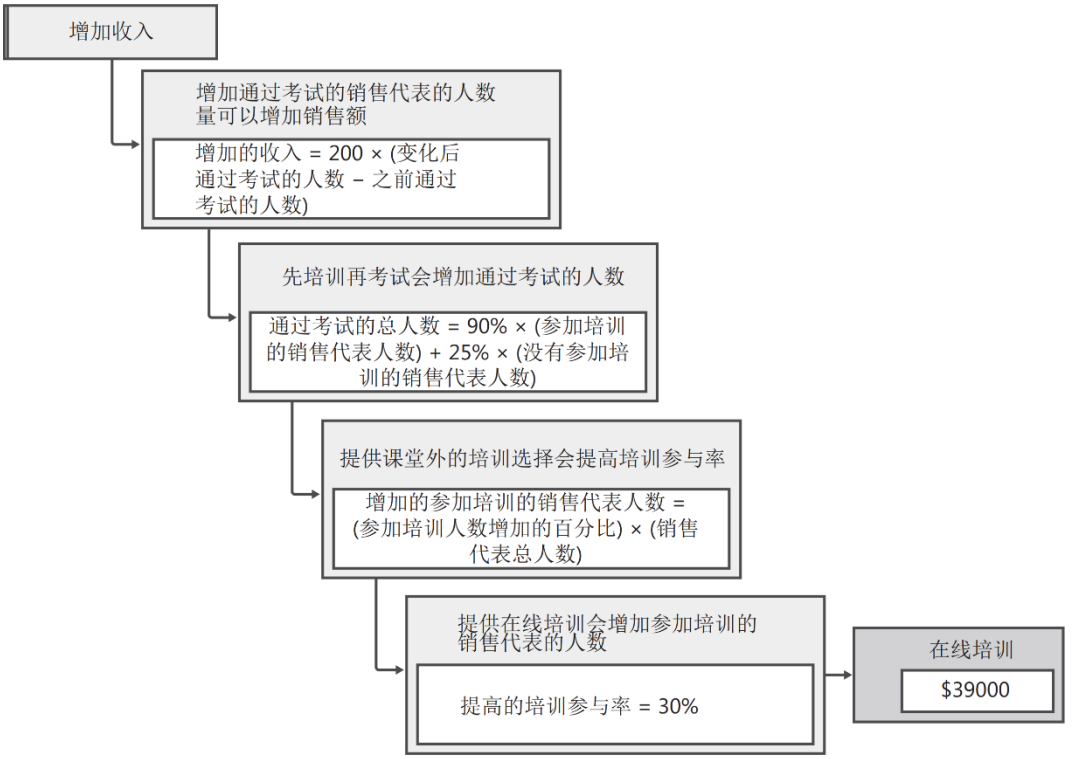


图 4-3 先培训再考试的目标因素和目标方程数据

注意，在图 4-3 中，前两个目标因素都有一个应用于现有数据的目标方程。这些方程有助于评估“在线培训”功能对业务目标的贡献程度。这些方程可能存在一些会对分析产生重大影响的假设。例如，分析假设如果提供了在线培训，那么参加培训的销售代表的人数将从 60%增加到 90%。如果出现了关于功能优先级的争论，那么争论的焦点应该是“在线培训会使参加培训的销售代表人数增加那么多”这一假设的有效性，而不应该就“是否应该实现在线培训”进行主观的争论。此外，在考虑在线培训的子功能（subfeatures）时，重心还是增加参加培训的销售代表的人数。这些子功能的例子包括通过电子邮件和自动提醒告诉销售代表参加培训。最后，分析师可以进行盈亏平衡分析（breakeven analysis），以确定需要增加多少参与率，才能使项目的收入刚好与实现在线培训的成本相当。即使团队不确定能达到多大的参与率，也可能会同意参与率极有可能小幅度增长。如果这个小幅度的增长仍然有利可图，那么实现在线培训项目潜在的风险就会小很多。

下一个要分析的功能是提供“可下载的培训”，它方便参与者在离线时完成。这个功能听起来很合理，有助于提高销售团队的整体通过率，因为培训变得更方便了。在与销售团队的交谈中，分析师可能发现，销售团队的大多数人几乎一直在线。有的销售会开车去见客户，但这段时间无论如何都不能拿来培训。还有一些人经常坐飞机去见客户，他们能使用离线培训选项。但是，总的来说，只有大约 2%的销售代表能从离线培训中受益。向这一

目标因素应用目标方程，我们可以确定，通过提供可下载的培训，只有 20 名额外的销售代表能接受这样的培训，而且其中只有 18 名销售代表能通过考试（20 名可能的销售代表中的 90%）。开发这个功能，每天能增加 3600 美元的收入（18 名销售代表每天贡献 200 美元的额外收入）。图 4-4 展示了最终完整的目标链。

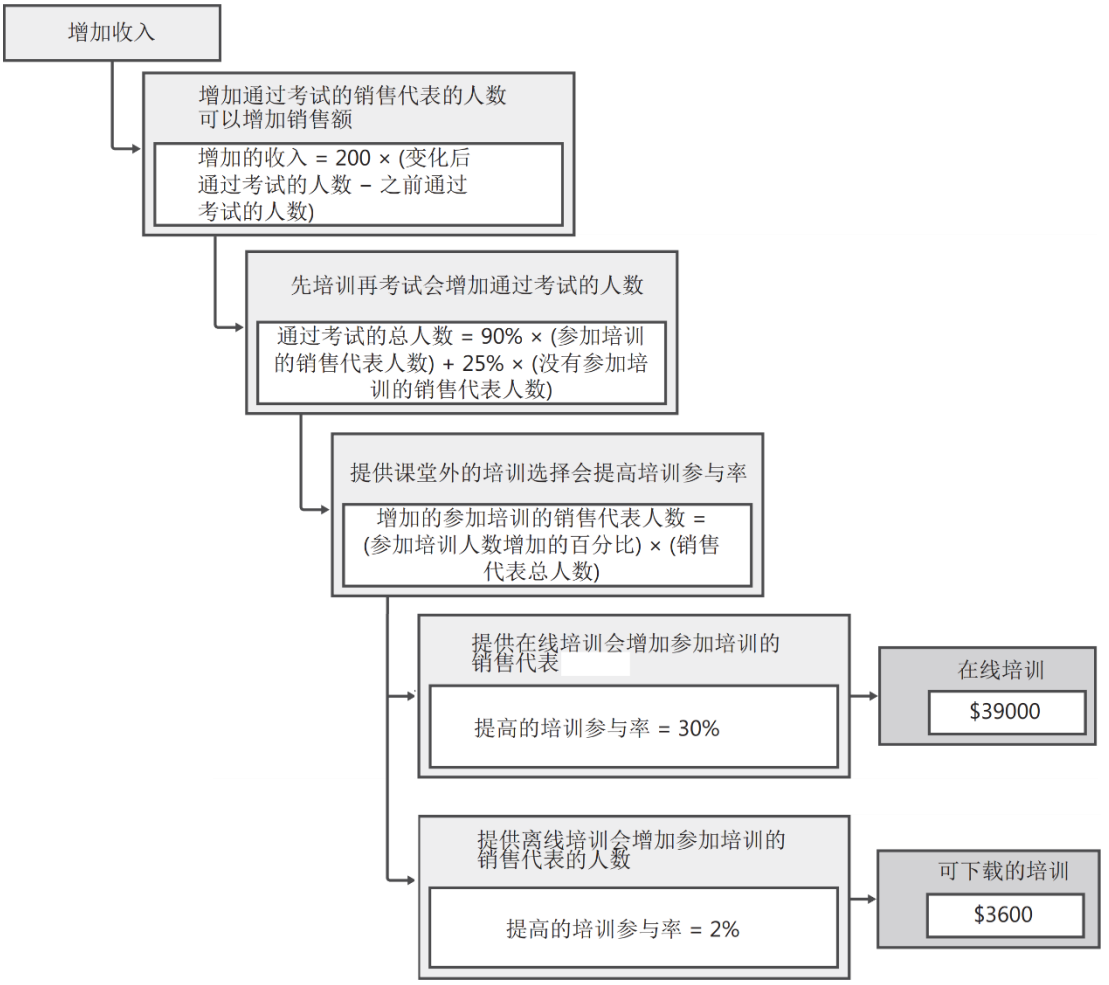


图 4-4 为“可下载的培训”功能新增了目标因素和数据

我们不必与顽固的利益相关方争论功能的有用性（“销售代表在飞机上参加培训的功能是必须的！”），相反，可以将争论的重点放在对回报的分析上。取决于开发离线培训功能的成本，团队可以做出不带感情色彩的、严格意义上的业务决定来保留或删除该功能。在本例中，分析师将目标方程应用于目标因素，发现虽然可下载的培训确实增加了价值，但这个价值和实现成本并不相当。使用带有估计的数据值的目标因素和目标方程，有助于将项目的范围限制在那些对项目最有利的功能上，并确保其符合业务目标。

创建目标链

如图 4-5 所示，创建目标链的过程是建立在项目早期创建的其他模型的结果之上的。目标链通常主要由分析师创建。但是，团队的其他成员也必须提供输入，以解释功能对于业务目标的贡献。

图 4-5 创建目标链的过程

确定业务目标和功能

首先澄清几个定义：

- 业务目标（business objectives）是标志业务问题何时得以解决的可度量目标。
- 功能（feature）是对解决方案最终将要包括以满足业务目标的一个功能区域的简短描述。功能是需求的集合，用于阐述和组织需求。

第 3 章讨论了如何确定作为业务目标模型一部分的业务目标。第 6 章将讨论如何使用功能树来确定功能。这两个模型结合起来使用，应该就可以提供要在目标模型中使用的一套足够完整的业务目标和功能。如果还没有创建一个功能树，那么在业务目标模型中确定的功能就是一个很好的起点。

选择要在目标链中分析的功能

不需要为每一个功能都创建目标链。相反，应该为最高级别的功能创建目标链，同时还要有做出范围决策所需的粒度。第 6 章将详细描述不同的功能级别（L1、L2 和 L3）。在选择适当的功能级别以进行分析时，可以遵循以下准则：

- 应选择每个功能都相对独立的一个功能级别。通常，可以在目标链中使用 L1 功能，只有在必要时才引入 L2 功能。例如，考虑一个“索赔管理”的 L1 功能，它包含“提交”、“调整”和“批准”等 L2 子功能。如果没有“调整”，那么“提交”的价值就很有限。而如果没有“提交”，那么“批准”和“调整”没有任何意义。因此，一个合理的做法是将“索赔管理”这个 L1 功能作为整体，基于它来开发目标链，而不是基于上述单独的 L2 功能来开发目标链。如果对这些 L2 功能的优先级出现疑虑，那么届时可以将目标链扩展到这些子功能。
- 选择一个功能级别时，要求开发每个功能的成本至少占价值的 1%。由于我们只关心数量级的功能价值，所以如果过于深入，去分析那些对整体价值或成本贡献太小的功能，回报就会衰减得越来越厉害。

- 如果不能在 L1 或 L2 级别删除任何功能，那么可以在所有 L1 的功能中寻找删除 L3 功能的机会。从 10 个不同的 L1 功能中删除 20%的 L3 功能，类似于彻底删除 10 个 L1 功能中的 2 个——假设实现每个 L3 功能所需的努力是相似的。如果需要以这种方式进行功能删除，那么必须做大量目标链建模来分析 L3 功能。
- 将目标链建模的重点放在业务目标模型中构成产品概念主要价值的功能上，而不是放在为其他功能提供支持的内务功能（housekeeping features）上。例如，许多解决方案都包括用户管理（user administration）功能。大多数情况下，用户管理并不是软件的目标，但它支持软件的真正目标，例如控制对信息的访问。因此，大多数项目都不需要在目标链中评估用户管理功能。

确定目标因素

为了确定目标因素，要逐一考虑从业务目标到功能的映射。从第一对业务目标和功能开始，回答“该功能对业务目标有什么贡献？”，如图 4-6 所示。答案应该是“增加或减少 X，会增加或减少 Y”（或者其他差不多的表述，例如“提供该功能会增加 Y”）。

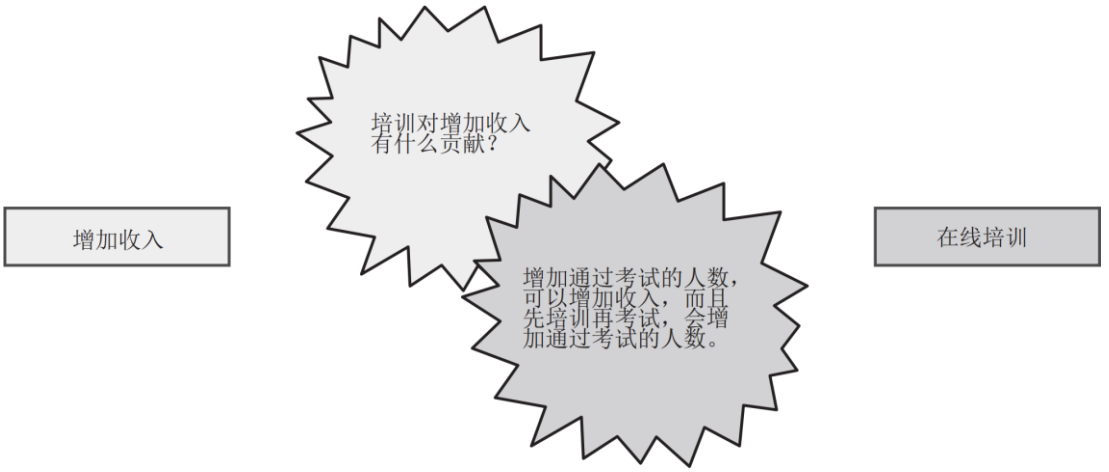


图 4-6 将“在线培训”功能与“增加收入”业务目标联系起来的思考过程

使用业务目标模型

业务目标模型将问题映射到业务目标，再映射到功能，但它并没有指出功能对实现业务目标的贡献。目标链旨在显示一个更细化的视图，传达每个功能与每个业务目标的关系。然而，完全可以使用业务目标模型中的问题和目标层次结构，帮助自己确定一个功能对业务目标的贡献。

例如，对于“增加收入”目标和“在线培训”功能，可以有如图 4-7 所示的层次结构。

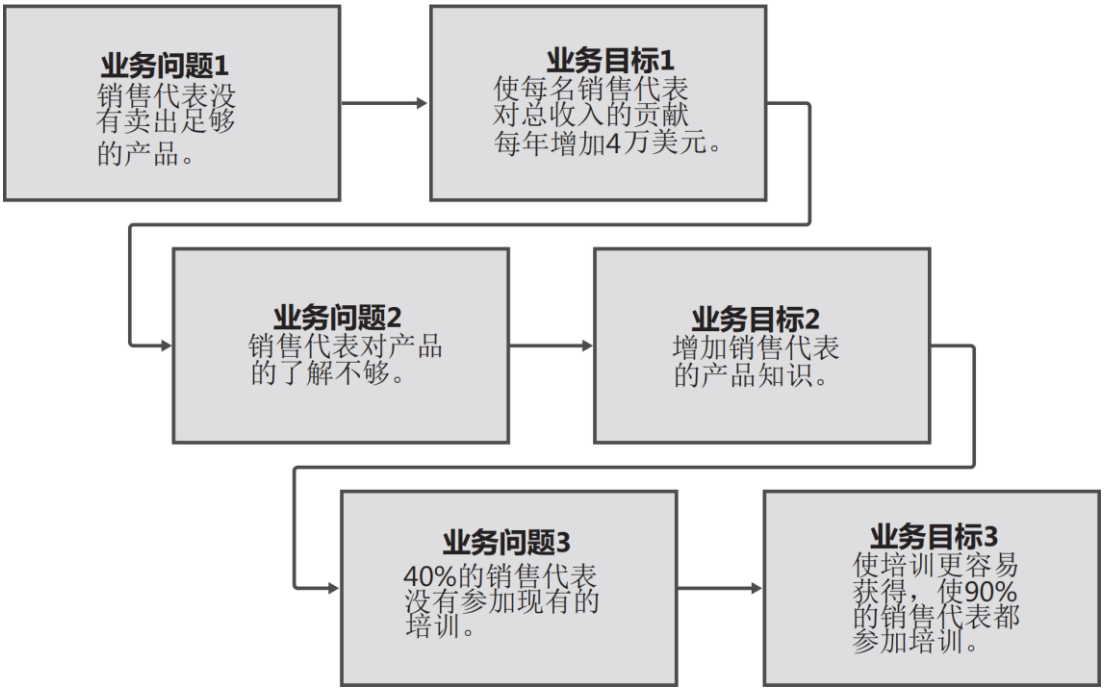


图 4-7 本例的业务目标模型

这些目标帮助你理解在线培训功能对增加收入的贡献。为了写出这个功能的实际目标因素，需要从底层开始，沿着链条往上写，如图 4-8 所示。

现实中并非总是存在“目标”到“目标因素”的一对一映射。在业务目标模型中，你可能会在自己的目标层次结构中进行了跳跃（因为它们显而易见），但在目标链层次结构中定义关系时，不能这样做。

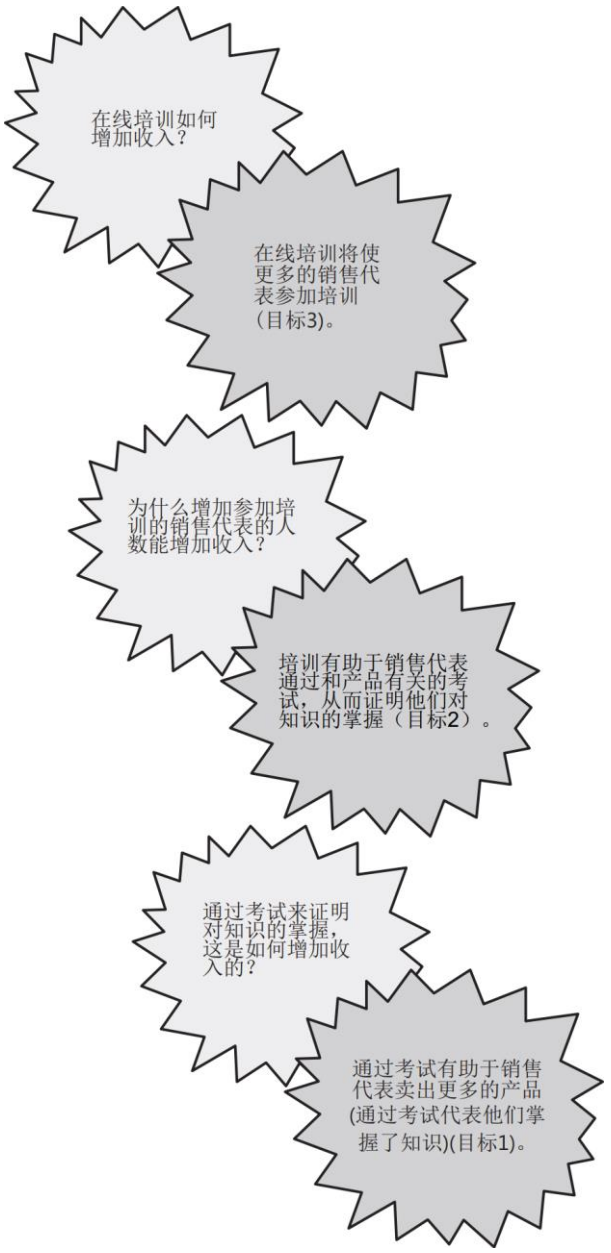


图 4-8 在业务目标模型的帮助下定义目标链层次结构的思考过程

简化目标因素陈述

回答一个功能对业务目标的贡献时，如果答案是一个复合句或者一段话，其中有像“而且”、“所以”和“因此”这样的词，那么可能存在多个级别的目标因素，应该把它们分离成独立的目标因素。在本例中，应该记录四个目标因素，如图 4-9 所示。

图 4-9 将“在线培训”功能与“增加收入”业务目标联系起来的目标因素

一个目标因素可能影响链条中的多个父级目标因素。为了简单起见，请组织每个目标因素，使它映射到受影响最大的一个父因素。

创建目标链层次结构

现在，每个功能都有了将其与一个业务目标联系起来目标因素。如图 4-10 所示，利用这些信息来创建一个有组织的树状结构，可以消除可能出现的冗余，并使信息更容易阅读和理解。

图 4-10 “在线培训”和“可下载的培训”功能的目标因素

为了开始创建目标链层次结构，请将第一个业务目标放在树的顶端。接着，从业务目标上链接出所有目标因素，最终将该业务目标链接到它的第一个功能。将功能添加到链条末端后，就形成了一个树状层次结构，如图 4-11 所示。

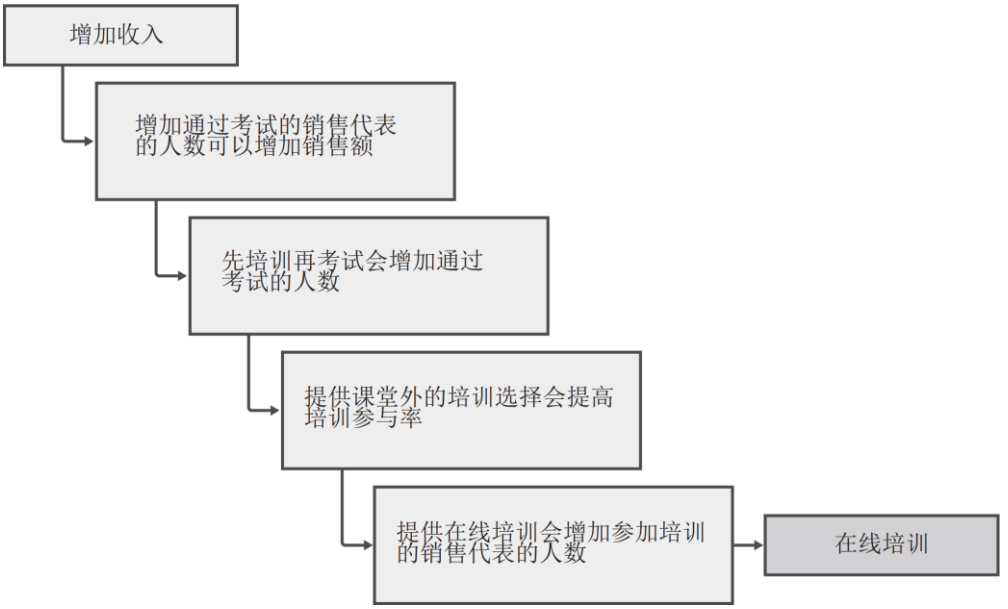


图 4-11 “在线培训”功能的目标链

现在，检查第二个功能以及将其链接到业务目标的目标因素。创建一个从业务目标到将其与第二个功能链接起来的所有目标因素的链，将这个链添加到与第一个功能相同的树状层次结构中。如果任何目标因素已经在树上了，就重用它们；不要重复。图 4-12 展示了生成的目标链。

继续为第一个业务目标在层次结构中添加目标因素和功能，直到所有选定的功能都在模型中。重复整个过程，为其余业务目标创建新的层次结构。

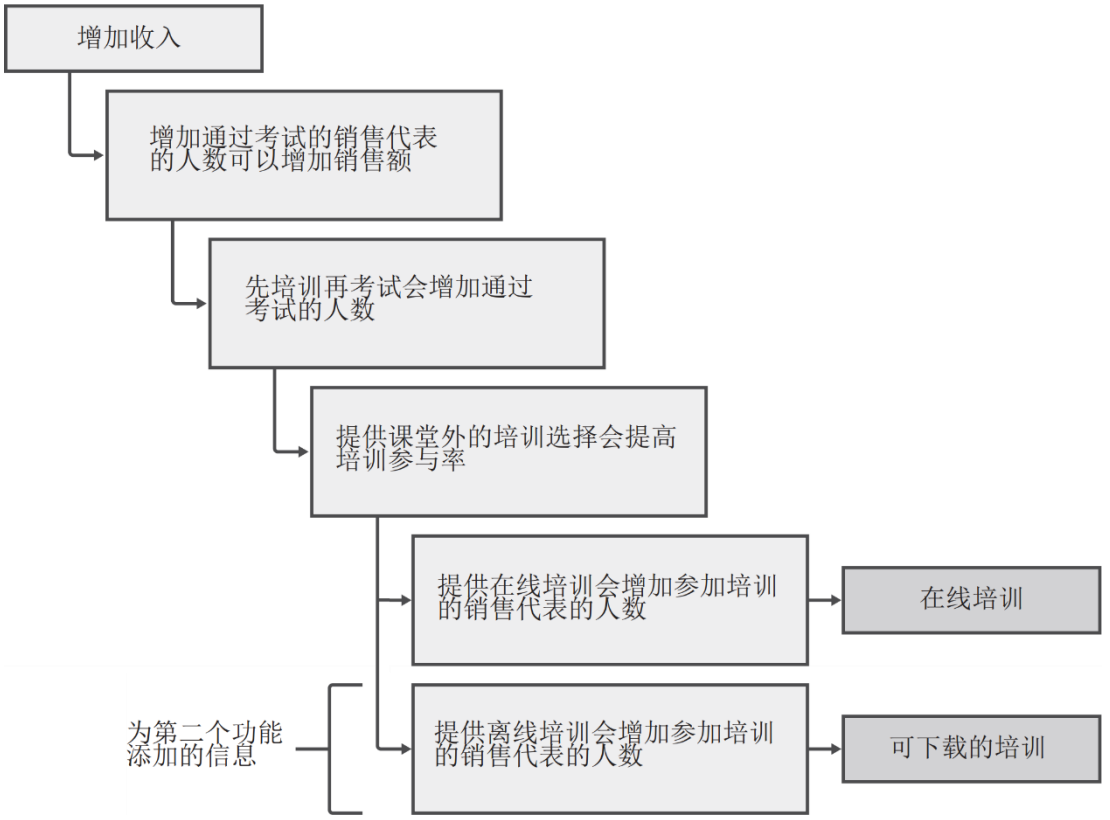


图 4-12 添加了“可下载的培训”功能的目标链

定义目标方程

在确定了目标因素后，要检查每个目标因素，以确定该功能对于业务目标的贡献。目标方程提供了为目标因素中的陈述提供定量支持的一个机制。这使分析师能够计算出任何特定功能对于金钱的贡献。目标方程的目的不是进行精确的计算；相反，它们的目的是提供可供比较的相对价值。

方程的格式化

在写目标方程时，应该使用相对价值，并以方程形式出现。以“增加的收入 = 200×(变化后通过考试的人数-之前通过考试的人数)”这个方程为例，它使用的是培训前通过考试的人数相对于培训后通过考试的人数。图 4-13 展示了这个例子的目标方程及其对应的目标因素。

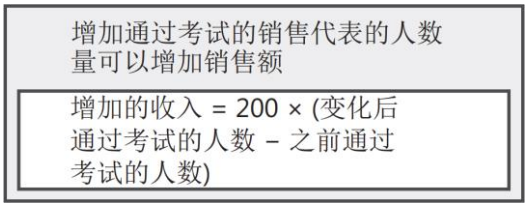


图 4-13 证明增加通过考试的人数会增加收入的目标方程

确定数据值

目标方程基于的是可获得的数据、可合理替代的数据或者近似值。可以从任何合理的来源获得数据，例如现有的系统、类似的项目、行业标准以及有根据的估计。业务利益相关方对这些指标的直观印象通常就是一个不错的开始。

创建目标方程的任务表面上很艰巨，尤其是在缺乏数据的情况下。然而，我们的目标是对功能的价值做出数量级的估算，从而按相同的标准进行比较。

在本章之前的示例目标方程中，每个值都代表目标方程中的数据。这些值包括：经过培训的销售人员每天能增加 200 美元的收入；当前有 600 名销售代表能参加培训；提供在线培训选择后，则有 900 名销售代表能参加培训；一个人先培训再考试的话，通过率会从 25% 上升到 90%。数据值有时是已知的，其他时候未知，但可以近似地计算。例如，提供在线培训选择后，有 900 名销售代表能参加培训，这是分析师对未来的一个预测。这是非常关键的一个假设，可以用来计算项目部署后的实际回报。

就数据值进行争论

注意，如果将目标方程记录下来，其他人就可以审查这个模型。如果他们有更好的数字或者其他想法，那么他们可以提供反馈意见。另外，围绕功能优先级的争论会自然地转移到每个功能的合理价值上，而不会带着情绪就每个利益相关方最喜欢的功能展开争论。团队不是在争论一个功能是否应该在范围内，而是争论一些假设是否靠谱。例如，在提供了在线培训的选择后，最终是否真的会有 900 名销售代表参加培训？争论的结果往往是，功能的发起人意识到他必须为自己的预测负责。结果，不需要的功能会从范围中删除，因为无人愿意为不靠谱的假设负责。

对假设进行度量

最后，如果使用的是假设而不是实际数据，那么应该增加允许对假设进行度量的需求，以便团队确定假设是否有效。随着时间的推移，团队所做的假设会越来越准确。在本章前面的例子中，分析师假设在提供了在线培训的选择后，培训的参与率能达到 90%。所以，分析师应该增加一项需求，度量销售团队的实际培训参与率。另外，项目计划应该包括一项

活动，在解决方案部署后验证之前的所有假设。

使用目标链

完成了目标链后，就可以采取一种可度量的方式，用它们来评估每个功能对业务目标的贡献。我们的目标是从范围中删除功能：

只要功能没有开发，就相当于帮项目省钱了。

应该在项目的早期使用目标链，避免在最终会被删除的需求上过多地投入。

比较功能的相对价值以缩小范围

可以比较每个功能相对于其他功能对于目标的贡献。从提供了最少价值的功能开始删除，从而实现范围的缩小。采用这种方法时，关于删除哪些功能的争论将围绕着为每个功能计算出来的价值而展开，而不是围绕着与每个功能相关的情感而展开。在本章之前的例子中，模型表明“可下载的培训”功能并没有为整个业务目标增加太大的价值。事实上，“可下载的培训”功能完全可以删除。

这种方法是自上而下的，并提供了一种系统化的方式来找出哪些功能组能增加业务价值。许多 L1 功能确实没有贡献那么大的价值，所以完全能自上而下地删除整个分支。在完成了最初的一轮删除之后，可以在检查剩余功能的下一层级的细节时再次使用这一技术。在前面的例子中，如果“在线培训”与其他不相关的功能进行比较，并最终从范围中删除，那么就没有必要分析在线培训的子功能了。

目标链的树状结构对这种分析很有帮助，因为它将功能的价值都清楚显示出来，可以方便地进行比较。

确定映射到多个业务目标的一个功能的价值

当一个功能映射到多个业务目标，并因此映射到多个目标因素和目标方程时，该功能的价值是适用的每个目标方程的价值之和。

例如，用于提供在线培训的功能除了“增加收入”业务目标，还可能链接到“减少培训师的成本”业务目标。培训师的数量减少了，成本自然会降低，而提供在线培训能减少对培训师的需求，这就形成了如图 4-14 所示的目标链。继续这个例子，我们可以确定，一名培训师培训 600 名学生的成本为 6000 美元。所有这些学生今后都可以改为在线培训，不再需要培训师。所以，这个功能在降低成本方面的总价值是 6000 美元。因此，“在线培训”功能的总价值是 45000 美元 = 6000 美元 + 39000 美元（后者是本例已经计算好的，参见之前的图 4-3）。

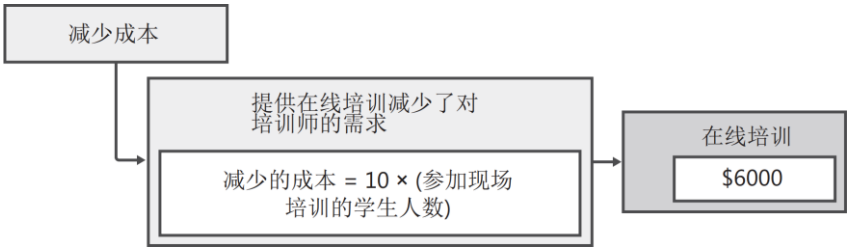


图 4-14 在目标链中将“在线培训”链接到“减少成本”

确定映射到同一目标链的多个功能的价值

如果多个功能被映射到同一目标链，就将目标方程的结果分散到各个功能上。不过，当功能之间相互吞噬的时候，分散结果就有点麻烦了。

例如，假定某公司有一个提供电话支持的系统，他们想减少支持成本。根据行业标准，利益相关方了解到，他们每将 10%的支持交互转换为在线支持，就能减少 4%的支持成本。因此，他们考虑添加聊天和电子邮件形式的支持功能。类似地，根据行业平均水平，他们估计仅增加电子邮件支持，能将 35%的支持转化为在线支持，从而减少 14%的成本。类似地，仅增加聊天支持，能将 40%的支持转化为在线支持，成本减少 16%。但是，如果同时添加聊天支持和电子邮件支持，那么仅有 50%的支持会转化为在线支持（而非 75%），因为使用电子邮件或聊天支持的客户存在一定的重叠。同时实现电子邮件和聊天支持，会使总成本降低 20%。表 4-1 总结了这些数据。

表 4-1 分析“在线支持”选项

功能	**在线支持转化率**	**成本减少百分比**	**每天 10000 美元的支持成本能节省**
仅实现电子邮件支持	35%	14%	1400 美元
仅实现聊天支持	40%	16%	1600 美元
同时实现聊天和电子邮件支持	50%	20%	2000 美元

如果假设“电子邮件支持”功能的价值为 1400 美元/天，“聊天支持”功能的价值为 1600 美元/天，并将这些功能加在一起得到 3000 美元/天，那么就夸大了这些功能的价值，因为从转化率可知，这两个功能加在一起，只降低了 20%的成本，即 2000 美元/天的综合价值（combined value）。另一方面，如果两个功能分别只值 1000 美元/天，那么两项功能或许都应该从范围中删除，因为它们的价值不够。

在这种情况下，应考虑开发和维护成本，挑选一个你认为最有价值的功能，将它的全部价值都分配给它。然后，将综合价值剩下的部分分配给另一个功能。在本例中，公司认为电子邮件支持最有价值，因为虽然实现聊天支持所获得的转化率会比电子邮件支持略高，但开发和维护聊天支持的成本约为电子邮件支持的三倍。有鉴于此，他们为电子邮件支持分配了它的全部价值，即 1400 美元/天，再将综合价值的剩余部分分配给聊天，即 600 美元/天（2000 美元 − 1400 美元）。如果有两个以上的功能映射到同一个目标链，就依次应用这个逻辑。

确定附属于情感目标的功能价值

有的时候，有一些映射到情感目标的功能可能具有无形的好处。它们自然无法映射到某个具体的价值。例如，这些功能可能有助于改善环境，能帮助一个非常小的用户群体，或者能避免管理层违法。在这些情况下，仍然可为它们以分配一个你选择“捐赠”（donate）的价值。基于组织愿意将多少资源花在这些无形的好处上，这些功能仍有可能会被删除。另外，还可以从环保意识和管理层不想违法的意愿出发，为这些功能赋予公关（PR）或营销（Marketing）价值。

确定项目是否成功

在解决方案部署完成后，可以使用目标链来确定项目所取得的价值。由于项目之外可能还有其他许多因素会影响收入，所以可能很难直接通过收入或成本数字看出解决方案的影响。还是之前的例子，如果在实现“在线培训”功能后，产品收入下降了，但培训参与率超过了假设的 90%，那么这个项目仍应被认为是成功的。即使可能无法度量项目对收入的直接影响，但只要参与率达到了 90%，那么根据目标链，就很有可能已经实现了项目的收入目标，现在是其他因素导致了收入下降。如果销售代表的参与率最终只有 80%，而其他所有假设都是正确的，那么可以用这个实际值来修改之前的假设，重新计算理论上的收入改善。

推导需求

我们不能从目标链直接推导出需求。相反，我们用它确定功能的优先级。另外，还可以用它缩小最终必须推导的需求集。使用目标链将某些功能从范围中删除后，就可以避免浪费时间去推导与这些功能相关的需求。

何时适用

在使用了业务目标模型的任何项目中都应该使用目标链，从而以一种系统化的方式为业务目标确定功能的优先级。

何时不适用

如果主要是为一个系统替换现有的功能，那么目标链的帮助不大，因为它们是用来对功能

进行优先排序的。在这种情况下，应改为使用关键绩效指标模型（参见第 5 章）来帮助自己了解哪些“过程流程”（Process Flows）是最重要的，并对它们进行优先排序。

常见错误

下面列出了关于目标链最常见的错误。

因为数据不存在就不创建目标链

如果没有目标方程的具体数据，团队选择不使用目标链。如前所述，如果使用合理的假设和有根据的猜测，那么目标链在这种情况下仍然是有价值的。

在层次结构中跳级

在业务目标模型中，经常可以在层次结构中跳跃，因为目标之间的关系很明显。但在目标链中，在层次结构中进行这样的跳跃通常是行不通的，因为必须说明一个层级如何影响下一级。如果过度简化这个层次结构，这些关系就没有了意义，也很难为各个功能评估具体的价值。

相关模型

有一个与目标链非常相似的概念，称为最小可销售功能（Minimum Marketable Features，MMF），它帮助分析师决定要哪些功能要纳入范围，以及何时开发。《Software by Numbers》一书详细描述了如何计算投资回报率来进行这种评估（Denne and Cleland-Huang 2004）。

下面简要描述了影响目标链模型或者被目标链模型增强的一些最重要的模型。第 26 章会对所有这些相关模型进行更深入的讨论。

- **业务目标模型（Business Objectives Models）**：为目标链提供业务目标和 L1 功能。
- **功能树（Feature Trees）**：为目标链提供功能以及关于这些功能的附加信息。

练习

以下练习帮助你更好地理解如何使用这种模型。练习是开放式的，因此你的答案可能与我们提供的答案大不相同。可能存在许多正确的解决方案。在答案中，我们对如何得出解决方案进行了解释。在看答案之前，你可以先尝试自己做一下，这样练习的收获最大。练习答案可以在附录 C 中找到。

说明

为以下场景准备一个目标链，并列出为完成目标链所做的任何假设。

场景

你的公司销售各种颜色和姿势的塑料火烈鸟和其他草坪摆件。团队已经确定了一个业务目标：“到明年年底，将年收入从 1000 万美元增加至 1250 万美元”。目前每年有 10 万名客户，回头客不多。团队已经同意实现一个网店（eStore），通过提供完整的在线产品目录和在线下单功能来实现这一业务目标。目前，平均订单金额为 100 美元，没有业务是在网上完成的。

根据你之前的经验，仅仅通过提供网购途径，就有望吸引 20000 名新的网站访客，这是因为公司有能力在各种新渠道进行营销，而且行业数据显示，这些访问者中有 90%能在第一年内转化为客户。通过将产品放到在线目录，公司预计每个客户在每个网购订单中会多购买一件产品，你估计该产品的售价通常为 10 美元。

团队中还在争论是否该开发其他功能，包括交叉销售[1]推荐以及打分和评价。行业数据显示，交叉销售会使平均订单金额增加 3%，打分和评价功能则会使这个金额增加 10%。

其他资源

- “What Do You Do When the Client Isn’t Focused on the Business Outcome?”描述了一个例子，企业虽然应用了目标链，但并不想用结果来做出关于范围的决定：https://tinyurl.com/3dwudx46。
- 《Software by Numbers》一书详细描述了如何计算“最小可销售功能”（MMF）的价值，从而限定项目的范围（Denne and Cleland-Huang 2004）。

参考资料

- Denne, Mark, and Jane Cleland-Huang，2004，*Software by Numbers: Low-Risk, High-Return Development*. Santa Clara, CA: Sun Microsystems Press, Inc.
- The Standish Group, 2009，“CHAOS Summary 2009”，West Yarmouth, MA: The Standish Group International, Inc.

[1] 译注：交叉销售（cross-sell）是指向已经购买一种产品或服务的客户销售其他类型产品或服务的过程，从而满足客户的多样性需求。例如，在客户做美甲的时候，通常可以向其交叉销售修脚服务。

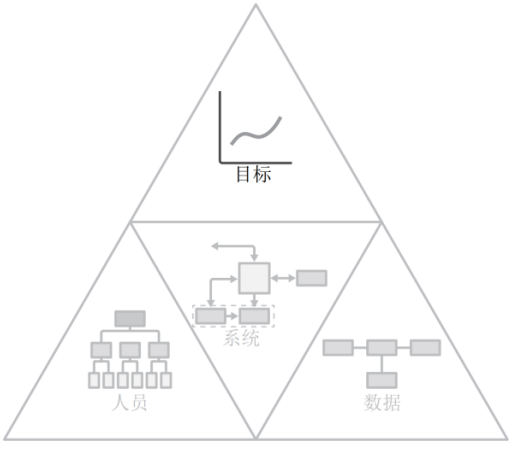
第 5 章 关键绩效指标模型

德州以烤肉闻名，尤其是低温慢烤的牛腩。通常，每磅牛腩的烧烤时间为 1.5 小时。由于一块典型的牛腩差不多 12 磅（5.44 公斤），所以一般从头天晚上就要开搞，让它烤上一晚上。收到我的新烧烤炉后，我决定为第二天的橄榄球派对烤上一块牛腩。这是我第一次烤牛腩，而且我知道需要多长时间。但是，我最终还是搞砸了。由于烤制时间过长，直到最后一个人都回家了，牛腩才真正烤好。不过，在烤好之后，它真的是我吃过的最好吃的牛腩！

在这个过程中，我测量了烧烤炉的温度和肉的温度。烧烤炉的目标温度是 250 华氏度（121 摄氏度），而肉的目标内部温度是 190 华氏度（88 摄氏度）。我监测了牛腩的内部温度，并计算了每小时的温度变化。根据每小时的温度变化，我可以预测牛腩能否在派对前完成。由于我进行了这种程度的监测，所以如果它没有按时完成，我就能调高火力，增大供热，加快烹饪速度。但问题是，火在晚上灭了。第二天早上起来时，我从前一晚的数据中得知，已经没有足够的时间完成烤制了。

关键绩效指标模型（Key Performance Indicator Model，KPIM）是一种 RML 目标模型。注意，关键绩效指标（KPI）是用来度量一个活动是否成功的指标。若应用于需求，KPIM 使用 KPI 作为与业务过程关联的度量标准，帮助团队对映射到这些业务过程的需求进行优先级排序。监测温度以预测烤肉的完成时间，这就是使用 KPIM 的一个例子。

如果一个新系统的重要部分复制了旧系统的现有功能，那么 KPIM 对于确保业务成果（business outcomes）的一致性特别有用。由于主要用于排定需求的优先级，所以 KPIM 是一种 RML 目标模型。



KPI 用于确保软件项目实现其价值。到目前为止，我们已经讨论了如何运用业务目标（第 3 章）和目标链（第 4 章）来确保项目实现目标价值。但是，这两个模型度量的是功能发

生改变的系统和过程是否取得成功，它们并不适合度量表现[1]是否至少和改变前一样好。

对一个现有的系统进行替换时，总会出现新系统与现有业务过程不匹配的情况。大多数时候都可以采用新系统所定义的新业务过程，从而降低因此而产生的风险。某些情况下，你可能决定对软件进行定制。但是，从旧系统转换到新系统时，如何排定现有业务规则、需求和业务过程的优先级，以确定哪些过程需要改变，哪些功能需要开发？许多利益相关方会说：“现在这些都要保留”，并以为别人都明白那是什么意思。但是，这个指示是不清晰的。是说每个业务过程都要完全一样？每个屏幕看起来都一样？新系统的外观和工作方式与旧系统完全一样？事实上，在这种情况下，利益相关方的重点应该是确保他们的业务成果（business outcomes）一样，而 KPIM 可以帮助他们做到这一点。

KPIM 模板

KPIM 是通过在“过程流程”（Process Flow）上叠加 KPI 来描述的（第 9 章会更多地讨论过程流程）。一个单独的 KPI 是以文本加支架符号（理解成方括号也没关系）的方式显示的，如图 5-1 所示。

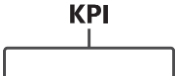


图 5-1 KPI 符号

这个符号框住的是 KPI 所适用的“过程流程”步骤。图 5-2 展示了完整 KPIM 模板的一个例子。

[1] 译注：KPI 中的 P 是指 performance，最初人们把它翻译为“绩效”，并延用至今。但是，performance 应取其原义，即“表现”。无论人，系统，还是过程，它的“表现”都是可以度量的。

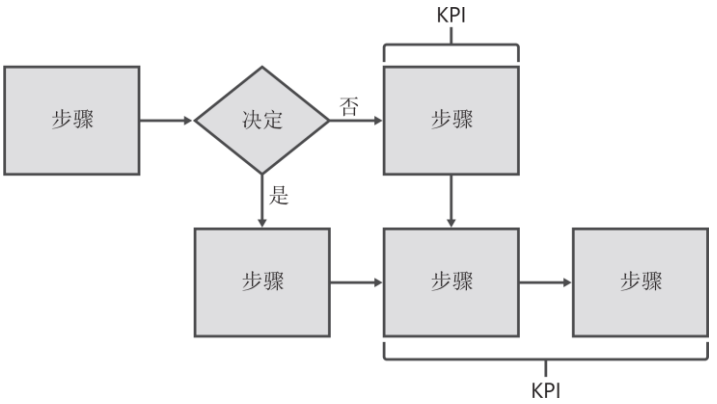


图 5-2 KPIM 模板

工具提示：KPIM 使用像 Microsoft Visio 这样的一个过程建模工具来创建。

例子

某抵押贷款公司想实现一个新的贷款发放系统。软件供应商的销售人员估计，在转到新系统后，公司每年将节省约 1 亿美元。这使得新系统 5000 万美元的许可证费用看起来似乎并不贵。系统提供了许多新功能，对旧系统进行了大量改进。另外，销售团队已经做了全面的差距分析[①]，表明新的贷款系统具有现有系统的全部功能，甚至提供了更多功能。这应该是一个容易的项目。但是，像这样的故事很少有一个美好的结局。

一名分析师开始调查具体应该如何配置软件以满足要求。他研究了贷款处理人员如何管理为贷款申请所收集的全部文件。这些文件包括房屋检查、个人财务报表和评估报告等等。所有这些资料都要以特定的方式排序，发送给核保部门进行审批（打印或转换成 PDF 格式），并以对方要求的格式发送给某家放贷银行。在收集资料的过程中，处理人员会来回发送大量文件，而且常常会遗漏资料，需要重新收集。

在当前的系统中，每名贷款处理人员每天能处理 60 笔贷款申请。为了达到组织规定的质

[①] 译注：差距分析（Gap Analysis）——又称缺口分析、差异分析——是战略分析方法之一。对公司制定的目标与公司预期可取得的结果进行比较，或者对公司制定的目标与公司实际取得的结果进行比较，分析两者之间是否存在差距。若存在差距，进一步分析造成差距的原因并制定措施（如改变目标、改变战略等）减少或消除差距。

量水平，贷款文件必须完整，确保核保团队和银行不再需要任何额外的资料来做出是否批准的决定。从收集抵押贷款申请资料，到将其发送给银行，时间跨度平均为五个工作日。这个过程涉及几个关键绩效指标，包括：一名处理人员每天能整理的申请数量、审计后发现的错误数量以及完成一份申请的总时间。图 5-3 的 KPIM 反映了这一过程。

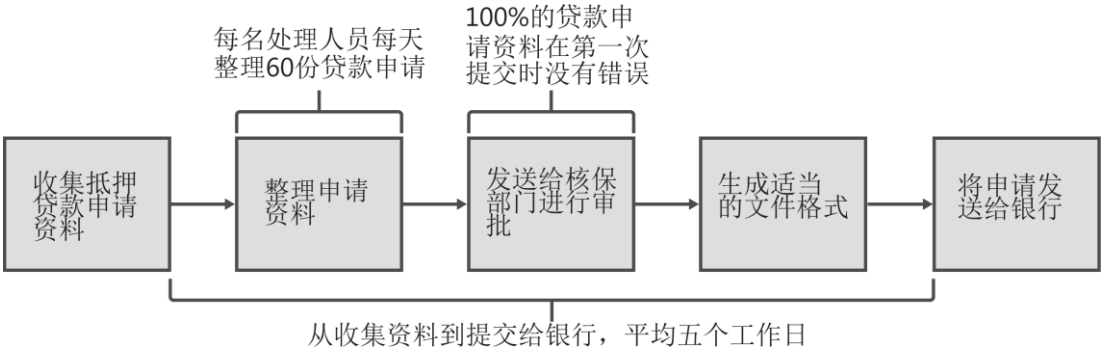


图 5-3 贷款发放 KPIM

旧系统有一个功能允许导入多个 PDF 文件，以电子方式分割，重新组织，合并成一个 PDF，然后以特定的顺序打印。遗憾的是，新系统不允许贷款处理人员重新排列 PDF 文件的顺序。所以，每个文件都要按照正确顺序扫描成单独的文件。不过，新系统支持打印带目录的大型分隔页，以便迅速找到信息，而旧系统需要手动插入分隔页。新系统还能跟踪记录哪些类型的信息尚需收集，以确保文件的完整性。旧系统支持键盘快捷键和宏，操作起来相当快；新系统则要求使用鼠标。这两个系统具有同样的一系列“功能”，只是这些功能以全然不同的方式实现。贷款处理人员非常喜欢他们当前的工作方式，因为可以方便地扫描一整叠文件，然后为扫描的页面手动分配正确的文件类型。

针对这种情况，我们需要解决两个问题：“是否需要为新系统添加将一叠文件扫描成 PDF 并以电子方式重新排序的功能？”以及“是否需要添加键盘快捷键？”。贷款处理人员坚称，这一功能对他们的工作至关重要。他们表示，一次扫描一份文件将是一场噩梦，会大大降低他们的工作效率。

KPIM 可以通过几种不同的方式帮助管理这方面的问题。一个新系统往往会在一个领域造成更多的成本，同时降低另一个领域的成本。整体净效应[①]可能为零，甚至为负。在这种情况下，分析师可以根据每天 60 份申请，100%文件完整率的总体目标来确定新系统是否

[①] 译注：一个经济行为可能会产生正的效应和负的效应，净效应（net effect）是两者相抵以后的效应，即，正效应与负效应的代数和。

需要定制。为此，可以简单地讨论一下新系统如何使一些操作变得更快，例如自动插入分隔页和目录。但是，这也可能需要开展一个试点项目，在其中进行时间运动研究[①]，以确定完成任务的实际速度，并通过对新系统的模拟来计算错误率。结果可能是，在新系统中执行任务的速度确实比较慢，贷款处理人员使用新系统每天真的只能处理 45 笔贷款。在这种情况下，要么必须对系统进行定制，要么必须增加贷款处理人员的数量以达成 KPI 目标。或者，如果贷款处理团队实际上还有潜力可挖，那么可以容许处理效率有一定的下降，因为这不会减少公司能够处理的贷款总笔数，也可能不会引起成本的增加。

如果有任何 KPI 目标不能实现，那么应该分析其他 KPI，以了解总体的净效应。如果新系统的其他方面为其他部门（例如营销团队或融资团队）带来的好处足够大，那么即使一个部门受到影响，另一个部门也可能会不成比例地受益，最终为公司带来总体的正收益。继续举例，如果对销售过程的改进大大增加了收入，那么贷款处理成本的增加也许是可以接受的。例如，营销团队目前可能没能力预测以前哪些客户可能会进行改善型购房。但新系统提供了一组功能，可以通过数据挖掘来预测以前的客户何时可能进行改善型购房。这些功能使营销团队能针对性地面向这些客户。软件供应商说，这一功能将增加平均贷款规模，进而使平均手续费[②]KPI 增加 25%。因此，即便贷款处理团队的效率下降了 10%，总收入的增加也将超过这个损失。

每个部门或团队都倾向于优化自己的一亩三分地。KPI 则能按相同的标准对跨越整个系统的需求进行比较，以确保在获得总体正收益的同时，最大程度地缓解范围蔓延。

创建 KPIM

图 5-4 展示了创建 KPIM 的过程。这个过程是建立在项目早期创建的其他模型的结果之上的。

图 5-4 创建 KPIM 的过程

[①] 译注：时间动作研究（time-motion studies），也称为工作研究，是指运用一些系统分析方法将工作中不合理、不经济、混乱的因素排除掉，寻求更好、更经济、更容易的工作方法，以提高生产率。

[②] 译注：贷款公司和放贷银行合作时，主要的收入来源就是手续费。

确定业务过程

第 9 章将讨论如何确定和创建“过程流程”（Process Flows），从而对业务过程（business processes）进行描述。要为 KPIM 确定业务过程，需选择要建模的特定业务过程，还需选择将在 KPIM 中使用的“过程流程”级别（L1、L2 或 L3）。

选择正确的级别

KPIM 通常使用 L2 过程流程来创建。L1 过程流程通常等级太高（过于笼统），以至于无法描述存在具体度量标准的步骤，而 L3 过程流程又过于详细。

为了确定使用哪个过程级别，请考虑以下几个指导原则：

- 选择能将业务过程映射到一个综合业务成果（consolidated business outcome）的级别。例如，一个用于“处理贷款”（process loan）的 L2 过程流程有许多 L3 过程步骤，例如“确保评估报告的完整性”（ensure appraisal is complete）和“检查征信”（check credit）。因此，更好的做法是在 L2 步骤而不是在这些单独的 L3 过程步骤上开发 KPIM。如果映射到 L3 过程的需求的优先级出了问题，再将 KPIM 扩展至 L3 过程流程也不迟。
- 选择已确定一个需求占软件价值至少 1%的过程级别。我们的目标尽量减少 KPI 的数量，以免尾大不掉。在对需求进行优先级排序时，我们就已经使用了各个需求估计的影响。而将需求分组（使用“过程流程”），可以避免为每个需求都建立 KPI。大多数部门都只有几个真正重要的 KPI，我们应专注于这些 KPI。

选择正确的过程

创建 KPIM 最具挑战性的步骤之一就是决定哪些过程需要有 KPIM。以下指导原则可以帮助你选择正确的过程：

- 从产生大部分成本的过程开始 KPIM 建模。换言之，这些过程花费时间最长，需要最多的资源，或者对于最终想要的结果最重要。
- 询问不同的部门，他们如何度量，或者他们的绩效目标（performance targets）是什么，并以此为基础来确定哪些与这些度量标准相关。
- 考虑每一个“过程流程”。可以和利益相关方共同列出一个过程清单，以确定哪些过程是维持或度量绩效的关键。

在现实中，关于过程级别的决定可能是与具体 KPI 的决定一起做出的。

确定 KPI

为了确定 KPI，需要检查已确定的每个“过程流程”的每一个步骤。对于每个步骤，都要

问以下问题：

- 该步骤的成果（outcome）是什么？是一个关键的业务交付成果（business deliverable）吗？
- 评定部门绩效时，是否需要度量该步骤的结果？

一般来说，没必要为每个过程步骤都创建 KPI。然而，如果上述问题的答案表明，某个特定的步骤需要严谨地度量，就需要做进一步的研究，为这个步骤确定一个或多个 KPI。每个步骤都可以有一个以上的 KPI，例如完成时间（time to complete）、典型量（typical volume）、最大量（maximum volume）和质量水平（quality level）KPI。其他 KPI 还可能包括完成一项任务所需的资源数量（number of resources），或者一些财务指标（financial measures）。以下问题有助于确定一个步骤的实际 KPI：

- 通过该步骤的量有多大（how much volume）？
- 如何确定该步骤的质量？
  - 可以接受多少错误（how many errors）？
  - 需要多大的人工干预（manual intervention）频率？
  - 能接受多严重的错误（how large of an error）？
- 完成该步骤需要多长时间？
- 完成该步骤需要多少人？
- 公司里有多少人执行该步骤？

创建 KPIM

创建 KPIM 的方法是采用标注（callouts）的形式，将 KPI 叠加到过程中的一个或多个步骤上，以方便识别。如果一个特定的步骤有多个 KPI，就将这些 KPI 记录到一个列表，并从“过程流程”中引用它们。

使用 KPIM

KPIM 应该在项目的早期使用，以帮助排定开发工作的优先级，使最重要的功能先完成开发。此外，KPIM 使团队能按相同的标准比较各个功能，以决定哪些功能需要删除。完成 KPIM 后，可以采取一种可度量的方式，用它们来评估每个需求对系统的业务价值的贡献。我们的目标是从范围中削减需求。只要功能没有开发，就相当于帮项目省钱了。

当业务目标不好用时排定 KPIM 的优先级

下面是将 KPIM 与业务目标进行对比的一个例子：最近，一个项目的目标是通过切换到一个具有较低许可证费用的系统以降低成本。通过切换到新系统，公司每年将节省大约 500 万美元的许可费。这个项目的业务目标很简单：“每年节省 500 万美元”。但是，并不能开

发一个功能来映射到“每年节省 500 万美元”。这是因为在许多项目中，特别是那些取代现有系统的项目中，有一个不成文的目标，即现有所有业务成果（business outcomes）都应保持在当前水平。许多时候，并没有人记录现有的业务成果，或者即使有，这些成果也分散在各个部门。KPIM 对跨越整个业务过程的业务成果进行识别、组织和记录，以确保对系统的任何改变都能将现有的业务成果保持在或者高于当前水平。

替换现有功能时排定优先级

在排定优先级时，你不能既为所有过程步骤使用 KPIM，又为所有功能使用目标链。相反，要根据项目类型选择最适宜的模型。一般来说，为现有系统的升级和业务过程的自动化使用 KPIM，为新功能或新系统使用目标链。更具体地说，需要维持或改进现有度量目标时使用 KPIM，需要估计所提议的功能的价值时使用目标链。

之所以要做这样的区分，原因是如果在系统中增加代表新业务过程的新功能，那么必然还没有这些功能的 KPI。但是，可能已经有了想要达成的业务目标。反之，如果在系统中用一个自动化过程取代一个手动过程，那么可能要确保该过程的总体绩效不会下降。即使真的有所下降，也要理解这背后的取舍。

在某些项目中，公司会尝试对一个过程进行改进。例如，假定已将一个过程从 90 分钟缩短至 15 分钟，这已经低于目标的 20 分钟。还可以做更多的改变，使这个过程自动化，并将时间进一步缩短至 10 分钟。但是，由于已经达成了目标，所以从项目的大范围内来讲，是否值得投入资源再从 15 分钟缩短至 10 分钟？或许可以更好地利用这些开发时间来改进另一个过程，使其达到目标水平。因开发团队的构成使然，团队很容易做太多的局部优化，而不考虑是否能更好地利用这些时间来优化另一些东西，为公司创造更大价值。有的时候，一些事物保持原样就已经很好了。所有部门或小组都会“情不自禁”地优化自己的工作，所以使用 KPIM 有助于跨越部门的界限进行优化。

比较需求的相对价值以缩小范围

对于大型项目来说，在系统转换过程中，可能有数百个 KPI 需要维持。使用之前为目标链解释过的相同的技术，我们可以计算出每个 KPI 的相对货币价值。由于 KPI 是连接到过程步骤上的，而过程步骤映射到需求（参见第 7 章），所以每个需求单独的价值贡献可与其他需求进行比较。可以找出那些提供了最少价值的 KPI，然后从范围中削减需求。当团队使用这种方法时，关于削减哪些功能的争论会围绕着需求对完成业务过程的贡献来进行，而不是围绕着与每个需求相关的情感。

基本上，我们使用 KPIM 来确定哪些过程步骤具有最高的优先级，然后检查需求，确保最高价值领域的 KPI 首先得到满足。

在本章之前讨论的放贷例子中，贷款处理人员坚持要求系统提供对 PDF 文件重新排序的功能。然而，在没有额外信息提供支撑的前提下，如果比较整个系统包含的多个 KPIM，

我们并不清楚 PDF 文件的电子重新排序功能是否提供了足够大的价值，值得不计成本地加入它。

推导需求

我们不是用 KPIM 直接推导功能需求（functional requirements）。相反，是用它们来确定各个“过程流程”（Process Flows）的优先级，并确保业务吞吐量能在需要的地方得以维持。另外，还可用它们来推导非功能需求。

何时适用

任何项目只要有需要维持的现有系统或者过程，就应使用 KPIM。

何时不适用

KPIM 对实现全新功能的项目没有帮助；对于这些项目，应该使用业务目标模型。然而，大多数项目都有一些现有的功能或过程需要维持或转换，所以可为这些项目使用 KPIM。对于完全没有业务过程的软件（例如一些通用软件，即 packaged software），KPIM 是没有帮助的。

常见错误

下面列出了关于 KPIM 最常见的错误。

因为没有 KPI 就不使用 KPIM

许多组织不使用运营指标（operational metrics）来监控业务过程，所以如果没有 KPI 的数据，团队就会选择不使用 KPIM。但在大多数情况下，KPIM 可以在项目期间创建，并且可以通过度量来评估当前状态和新的目标状态。另外，基于可能出问题的功能，可以采取一种及时（just-in-time）的方式来收集 KPIM。

由于担心被追责而不使用 KPIM

毫不奇怪，对于企业中各种各样的组织，他们对创建 KPI 往往有抵触情绪，因为一旦开始度量，就意味着企业现在要对他们之前可能从来没有遇到过的 KPI 负责。分析师可以帮助业务利益相关方了解如何在内部使用 KPI 来帮助改善他们的运营，并在新系统中做出更好的实现选择。

缺少持续监测

当团队选择使用 KPIM 时，他们往往忽略了提供一种机制来持续监测 KPI。

相关模型

KPI 在许多组织中是以列表（list）的形式存在的。KPIM 将这些 KPI 映射到不同的“过程流程”，以便将其置于具体的背景中。

下面简要描述了影响 KPIM 或者被 KPIM 加强的一些最重要的模型。第 26 章会对所有这些相关模型进行更深入的讨论。

- **过程流程（Process Flows）**：为 KPIM 提供业务过程。
- **需求映射矩阵（Requirements Mapping Matrices）**：用于按业务目标模型中的业务目标和功能来组织需求。提供从过程步骤到需求的映射。因此，一旦过程流程被删除，需求也可以删除。

练习

以下练习帮助你更好地理解如何使用这种模型。练习是开放式的，因此你的答案可能与我们提供的答案大不相同。可能存在许多正确的解决方案。在答案中，我们对如何得出解决方案进行了解释。在看答案之前，你可以先尝试自己做一下，这样练习的收获最大。练习答案可以在附录 C 中找到。

说明

为以下场景和提供的“过程流程”准备一个 KPIM。

场景

你的公司销售各种颜色和姿势的塑料火烈鸟和其他草坪摆件。销售火烈鸟产品的销售团队使用 Microsoft Excel 模板和电子邮件来创建订单，并发送给客户进行核实。然后，销售代表将批准的订单手动输入订单履行系统（order fulfillment system）。随着新的网店（eStore）系统的实施，一个全面的订单管理解决方案将被整合到网店中。

销售经理发现，基于当前使用的人工过程，他们的销售代表每小时能提交 30 个订单，销售经理不希望在使用新系统后，他们的工作效率不升反降。由于当前系统使用人工过程与客户一起核实订单，所以订单在提交给订单处理中心时通常是正确的。在提交时，唯一可能出现的错误是打字错误。当前，只有大约 2%的订单输入不正确。经理们不希望新系统的总订单处理时间有任何形式的减慢；目前，从收到订单到发货不超过 3 天。图 5-5 展示了这个场景的“过程流程”。

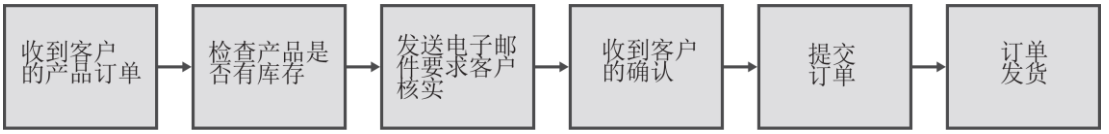


图 5-5 本练习的过程流程

其他资源

- 以下网站提供了关于 KPI 的大量信息和例子：http://kpilibrary.com
- Business Objectives, KPIs, and Legacy Conversions：https://tinyurl.com/5bk8tmz9

附录 A　模型速查表

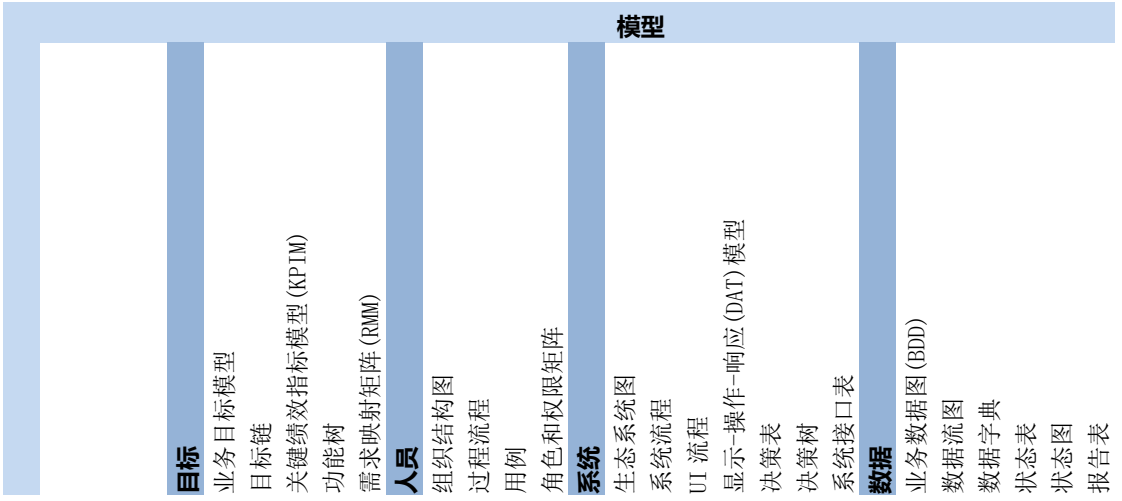
为自己的项目确定要使用的模型时，本附录可以作为一个快速参考。它以简单的网格格式总结了对于模型的选择（详情参见第 25）。可以下载这些网格的一个拷贝，以便在手边没有书的时候使用。事实上，我们喜欢在桌子上放一份压膜的拷贝，这样在做项目的时候就可以把它作为一个能够快速参考的工具。

图 A-1 是一个完整的网格，展示了按项目特征划分的模型。使用它的时候，请自行确定哪些行所代表的特征符合自己的项目。注意，特征应选尽选。然后，向右阅读，看看为当前项目类型和那些特征推荐了哪些模型。记住，我们并没有创建一个巨细无遗的项目特征集。所以，在遇见这里没有列出的项目特征时，你几乎肯定需要创建新的特征行。

如果一个单元格留空，表明单就当前行的特征来说，并不需要当前列的模型。可能仍然需要为自己的项目使用该模型，但这是仅仅是因为项目符合一个不同的特征，而且该特征建议使用该模型。反之，如果单元格中填写了实际的内容，就意味着该特征本身就足以表明该类型的项目需要该模型。表 A-1 解释了速查表的单元格中的值的含义。

表 A-1　基于项目特征的模型速查表——单元格中的值的含义

含义	**单元格中的值**
极可能需要该模型（Likely）	L
可能需要该模型（Might）	M
单就此特征来说，不需要该模型	留空



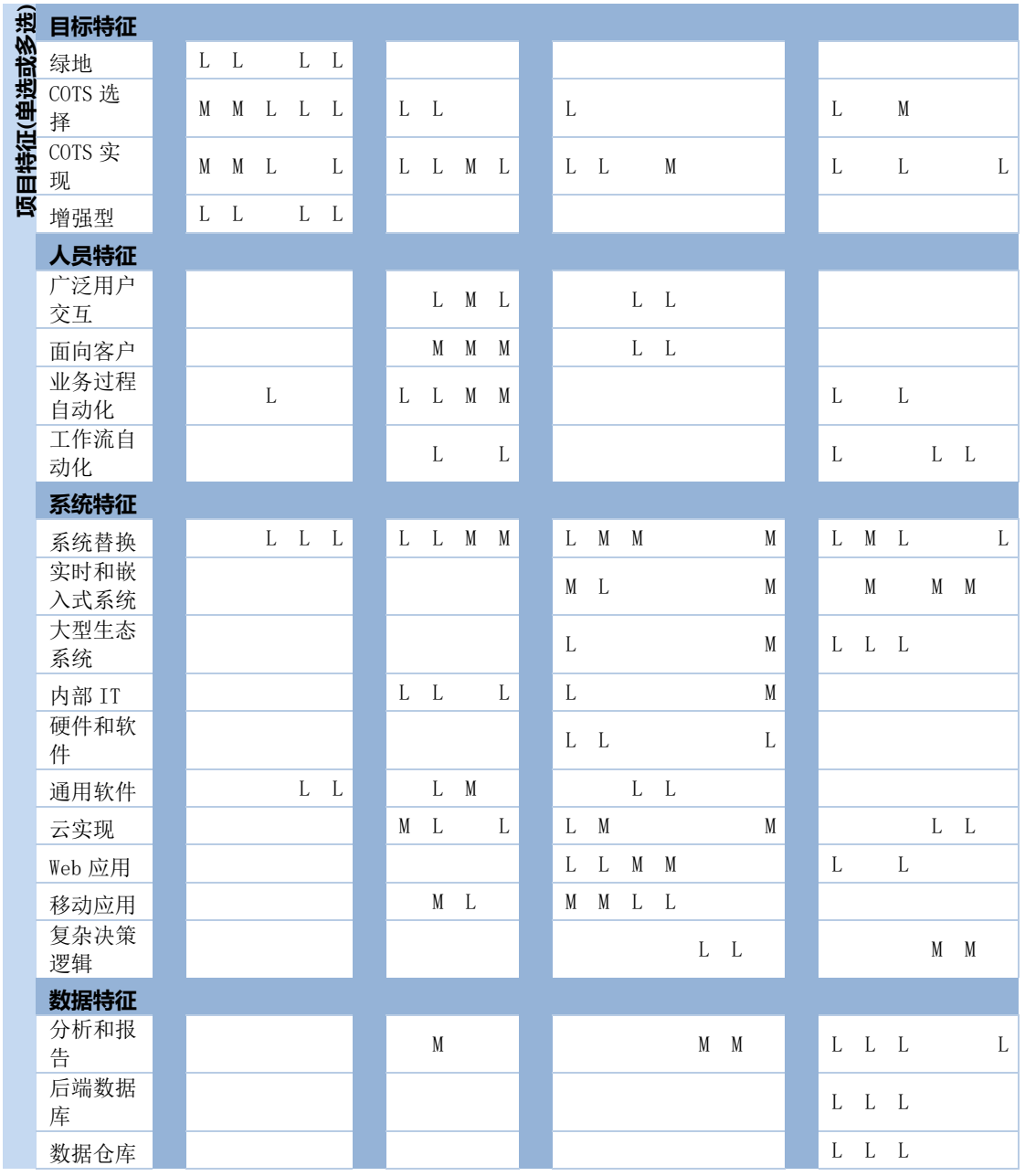


图 A-1 基于项目特征的模型速查表

图 A-2 归纳了不同项目阶段所需的模型，详情请参见第 25 章。这个速查表是为了帮助你确定在项目的什么阶段创建和使用模型。记住，不是所有项目都会完全照着这些阶段进行。例如，如果一个项目已经开始了，那么可能会在“计划”阶段创建一个业务目标模型，因为已经没有了“设想”阶段。表 A-2 解释了单元格中的值的含义。

表 A-2 基于项目阶段的模型速查表——单元格中的值的含义

含义	单元格中的值
创建（Create）	C
使用（Use）	U

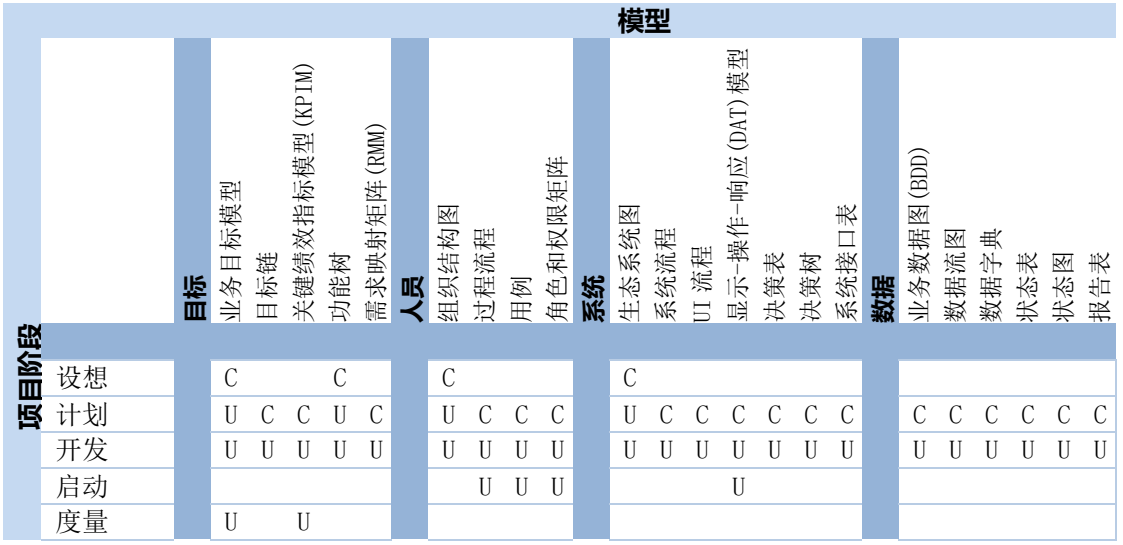


图 A-2 基于项目阶段的模型速查表

其他资源

- 模型速查表可以从 http://go.microsoft.com/FWLink/?Linkid=253518 下载。

附录 B 模型的一般准则

所有模型在某些方面都是一样的，这些共性有助于利益相关方更好地理解模型。本附录描述的一般准则适用于大多数模型，但具体在什么时候使用，还是要依据你自己的判断。

要包含到模型中的元数据

虽然模型模板的大部分内容都因模型而异，但如下表所示，有一些信息可能应该包括到你创建的每个模型中。如果使用的是某个需求管理工具，那么其中一些信息会自动为你保存。

元数据	**定义**
标题	模型名称
作者	模型创建人
项目	模型用于哪个项目
版本（Release）	模型用于项目的哪个版本
描述	对模型所描述内容的简短总结
唯一标识符	从其它模型引用该模型时使用的唯一标识符
相关文件	与该模型有关的其他文件的一个清单
版本历史	随着时间的推移，对模型所作更改的历史
创建日期	模型的创建日期
修改日期	模型的最后一次修改日期
批准人	已经阅读并将批准或已批准该模型的利益相关方
位置	模型的存储位置，方便拿硬拷贝的人能够找到源文件，也方便拥有本地拷贝的人找到最新版本

一般准则和提示

建议遵循以下一般准则来确保模型的可读性：

- 确保所有类似的形状都一样大小。例如，所有流程步骤的大小应该一致，决策菱形的大小也应该一致。该准则的一种例外情况是，如果需要更多的空间在一个形状内填写文本，那么可以让这个形状的一个实例比其他大，前提是它们都放大为一致大小的话会浪费很多空间。
- 确保所有类似形状中的文本都一样大小。例如，如果更改了一个方框中的字号，以便为流程步骤腾出空间，那么所有方框中的字号都可能需要更改，以保持一致。至少应该尝试在类似的形状（例如流程步骤）中使文本的字号一致。但最理想的情况是，一个模型中的所有形状（流程步骤和决策菱形）中的文本都一样大小。
- 对于具有层次结构的模型，对不同层级进行统一编号。例如，从左到右对 L1 过程流程步骤进行数字标注，然后通过引用具体 的 L1 步骤对 L2 过程流程进行命名和编号。为同一个项目的类似模型（例如过程流程和系统流程）使用相同的编号方案。
- 使用颜色来区分模型内的元素时，用一些技术来确保在模型以灰度打印时也能明显区分。例如，可以使用图案，或者更改形状的轮廓线类型。
- 模型中的对象之间要适当留空以保持美观。例如，几个对象不要挤在一堆，要善于利用空白。
- 当信息有一个基本的顺序时，通过从左上角到右下角的流动来强化这个顺序。
- 模型中的所有文本都使用主动语态。
- 在一个项目的类似模型中使用一致的大写形式（第一个单词首字母大写，或者每个单词首字母大写）。例如，对于项目的所有过程流程，所有步骤都只有第一个单词首字母大写。
- 在所有模型一致的位置呈现元数据。例如，如果为一组过程流程创建 Microsoft Visio 文件，并且它们共享元数据，那么为 Visio 文件创建一个背景页来包含元数据。然后，在包含模型的所有 Visio 文件中，都在同一位置创建一个同名的相似页面。

附录 C 练习答案

第 3 章

取决于所问的问题和为这些问题编造的答案，这个练习的结果可能会有很大的不同。下面这个清单提供了一些合理的问题和可能得到的答案：

- 你想通过建立一个网店来解决什么问题？我们认为，多一个销售渠道将增加我们现有的收入。
- 收入没有增加会怎样？管理团队希望在五年内出售公司。为此，我们必须达到 2000 万美元的收入。
- 你认为网店将如何帮助增加收入？这样一来，许多非本地区的人也会购买我们的产品。
- 当前是什么阻碍了向这些人销售？主要是他们不在我们的地区，我们没有符合成本效益的方法向他们推销。
- 希望明年增加多少收入？到明年年底，希望达到 1250 万美元。到五年结束时，希望年收入达到 2000 万美元。
- 是否计划将电话订单转换为网上订单？我们也许会将一些现有客户转化为网购。然而，我们期望吸引新客户。他们现在不喜欢打电话下单，但如果整个订单都在网上完成，他们也许就会向我们购买了。
- 你认为增加新客户是增加收入的唯一方式吗？不是，我们也希望现有客户在转化后能在网上比在电话上购买更多的东西，因为他们能看到更多可能感兴趣的产品。
- 如何吸引新客户？通过推广网店，我们希望能吸引那些以前不会通过电话来购物的新客户。
- 如何提高现有客户的购买量？我们的销售代表可以在与现有客户通话时告诉他们有新的网店了。我们还会发送推广电子邮件，让他们知道网店的推出。

利用上述问题和答案，可以开发如图 C-1 所示的业务目标模型。

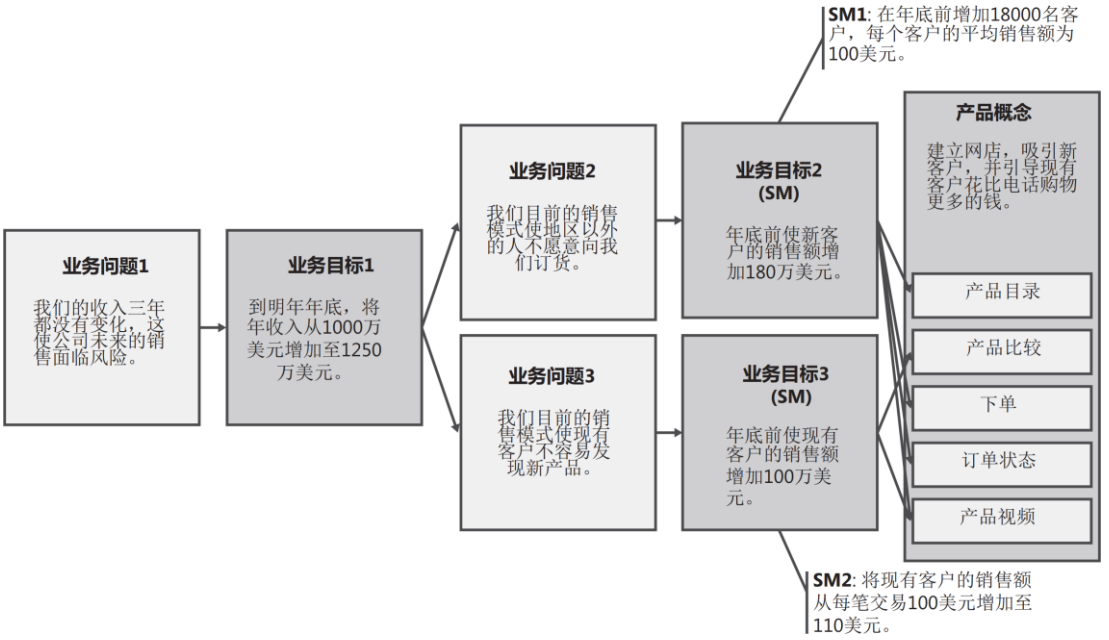


图 C-1 网店（eStore）的业务目标模型

第 4 章

业务目标和功能已在“场景”中提出，所以我们的工作是创建目标因素和目标方程来连接它们。为了确定目标因素，要问：“这个功能是如何增加收入的？”对于这个问题，一些可能的回答包括：

- 产品目录使顾客能注意到比电话订购时更多的产品，导致每笔订单购买的产品数量增加。
- 交叉销售可以推荐与当前购买的产品相匹配的其他产品。产品打分和评价为客户提供了更多关于产品的信息，让他们知道其他消费者已经认可了产品的质量。
- 增加一个网购途径，会吸引新的访问者来到网站，其中一些人将成为客户。

为了完成目标方程并计算每个功能的价值，团队可以做一些假设，例如：

- 新客户预计将花费与现有客户网购时一样多的钱。
- 在线客户的消费额与电话客户的消费额差不多。
- 交叉销售使平均订单金额增加 3%。
- 打分和评价使平均订单金额增加 10%。
- 由于看到了额外的产品，会导致每个客户多购买一件产品。
- 现有客户额外购买一件 10 美元的产品。
- 网店每年将吸引 20000 名新的访客。
- 90%的网店访客会下单。

由此得出的目标方程和功能价值计算如下：

- 在网站上增加产品目录，将使现有客户的每笔订单增加一件产品，每件产品增加 10 美元。目前有 100000 笔订单，这就是 100 万美元的额外收入。
- 交叉销售以及打分和评价功能分别增长 3%和 10%的平均订单金额。如果通过交叉销售，每笔订单多了 3 美元，10 万笔订单，那么收入会增加 30 万美元。如果通过打分和和评价，每笔订单多了 10 美元，10 万笔订单，那么收入会增加 100 万美元。
- 如果有 20000 名新的潜在客户访问网站，其中 90%的人在一年内购买了产品，那么会产生 18000 笔新订单。按每笔订单 100 美元计算，这相当于 180 万美元的新增收入。

最终的目标链如图 C-2 所示。基于目标链，团队现在可与开发团队和利益相关方合作以确定实现每个功能的成本，然后就能直接确定优先级。

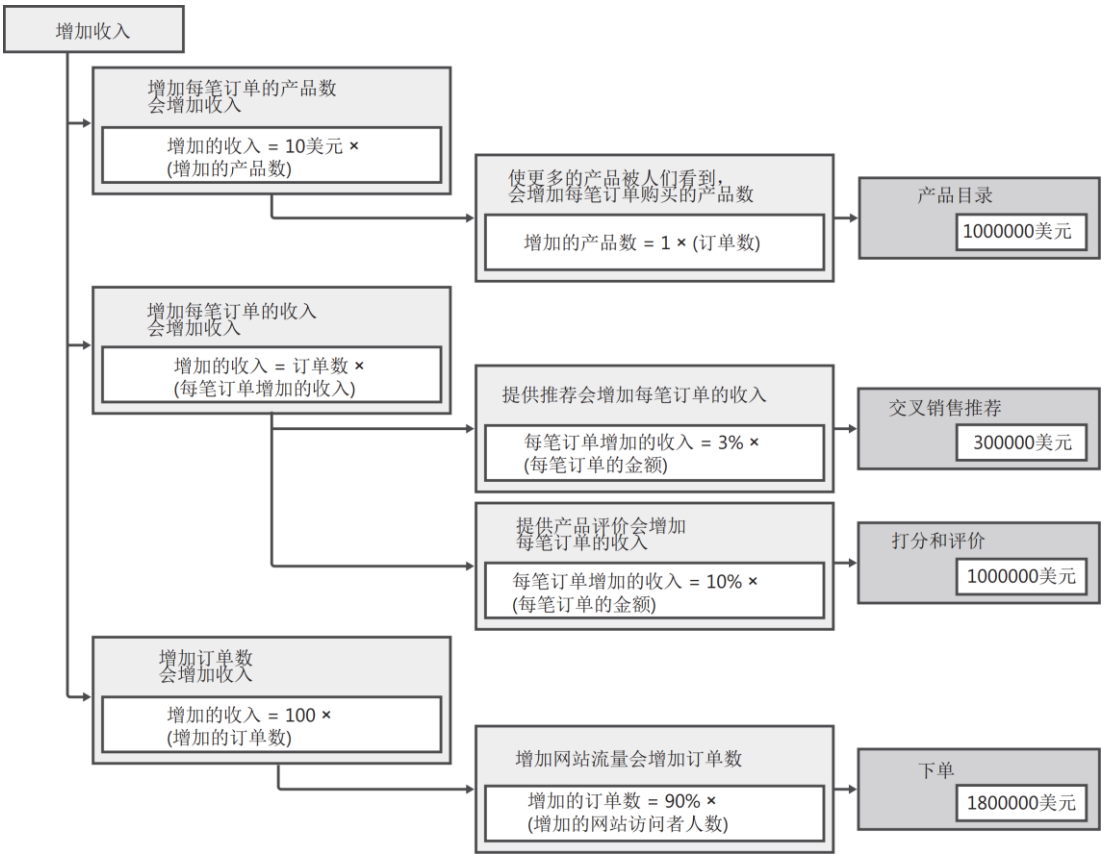


图 C-2 网店（eStore）针对 4 个功能的目标链

第 5 章

销售经理目前衡量的 KPI 有三个，并希望在新系统中保持。这些 KPI 被叠加在流程上，形

成图 C-3 所示的 KPIM。

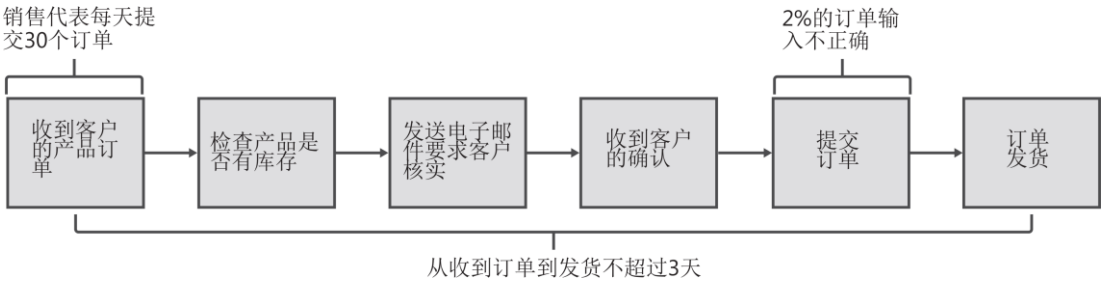


图 C-3 从准备订单到发货这一过程的关键绩效指标模型（KPIM）

第 6 章

为了创建功能树，首先将示例场景给出的功能写在便签上。你应该确定诸如“监控负载”、“相关产品”、“查看系统日志”、“查看网站使用数据”、“链接交叉销售产品”、“创建帐户”、“查看网站订单数据”、“更新火烈鸟目录”、“产品打分”、“浏览产品”、“产品愿望清单”、“产品评价”和“购买产品”等功能。

然后组织这些功能，形成第一轮的功能分组，并根据这些分组添加到功能列表中。例如，我们确定了“购买产品”、“产品愿望清单”、“管理帐户”、“管理产品”和“系统监控”等组别。在此基础上查漏补缺，我们发现缺少“购物车”和“将产品加入目录”功能。

有了一个好的功能列表初稿后，将功能转移到功能树中，并继续添加功能，直到最终完成，如图 C-4 所示。注意，在这个时候，为了保持一致性，功能要重新进行命名，统一为使用名词短语。最后，寻找功能树上那些单薄的分支，并重点调查。例如，管理员是否有需要做其他类型的帐户维护工作？

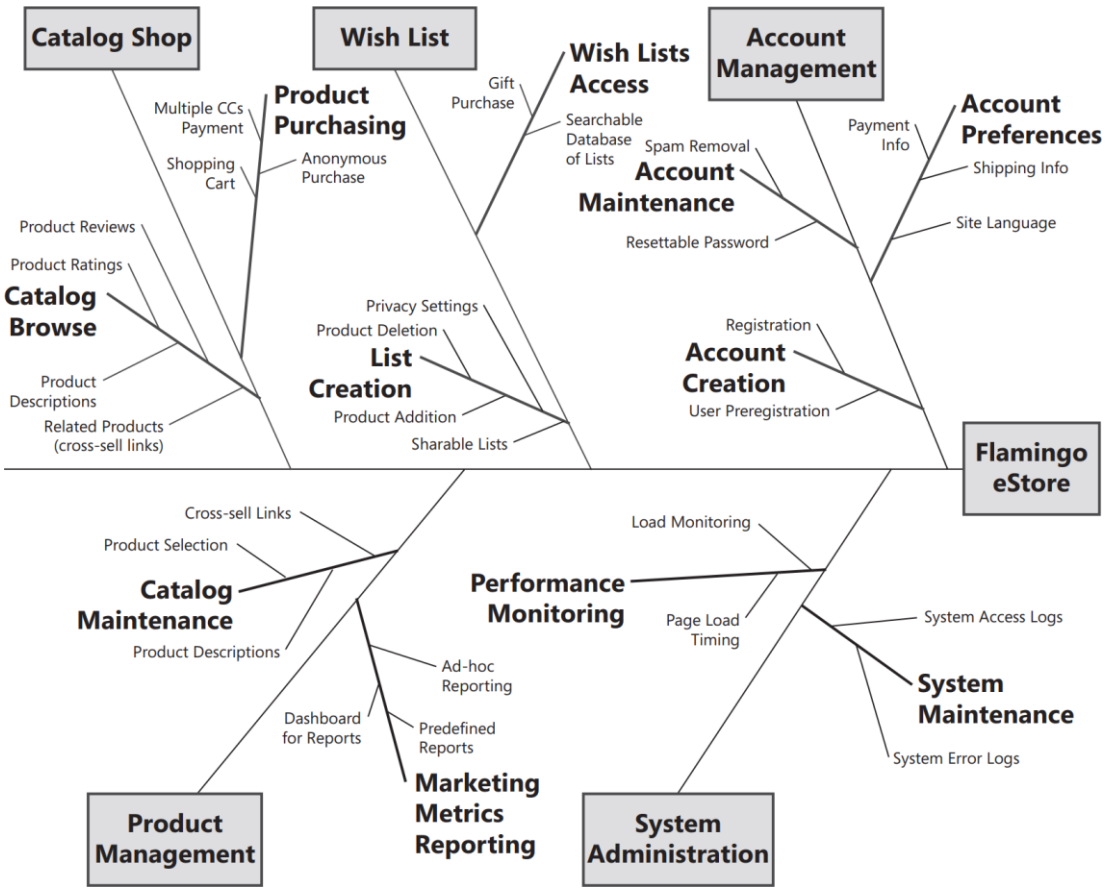


图 C-4 网店（eStore）的功能树

405 页图中译文：

Catalog Shop：商店目录

Wish List：愿望清单

Account Management：帐户管理

Catalog Browse：目录浏览

Product Reviews：产品评价

Product Ratings：产品打分

Product Descriptions：产品描述

Related Products (cross-sell links)：相关产品(交叉销售链接)

Product Purchasing：产品购买

Multiple CCs Payment：多张信用卡付款

Anonymous Purchase：匿名购买

Shopping Cart：购物车

List Creation：愿望清单创建

Privacy Settings：隐私设置

Product Deletion：产品删除

Product Addition：产品增加

Sharable Lists：可分享的愿望清单

Wish Lists Access：愿望清单访问

Gift Purchase：礼物购买

Searchable Database of Lists：可搜索的愿望清单数据库

Account Maintenance：帐户维护

Spam Removal：垃圾信息删除

Resettable Password：可重置的密码

Account Creation：帐户创建

Registration：注册

User Preregistration：用户预注册

Account Preferences：帐户首选项

Payment Info：付款信息

Shipping Info：收货信息

Site Language：站点语言

Flamingo eStore：火烈鸟网店

Product Management：产品管理

System Administration：系统管理

Catalog Maintenance：目录维护

Cross-sell Links：交叉销售链接

Product Selection：产品选择

Product Descriptions：产品描述

Marketing Metrics Reporting：营销指标报告

Dashboard for Reports：报告仪表盘

Ad-hoc Reporting：特别(ad-hoc)报告

Predefined Reports：预定义报告

Performance Monitoring：性能监控

Load Monitoring：负载监控

Page Load Timing：页面加载计时

System Maintenance：系统维护

System Access Logs：系统访问日志

System Error Logs：系统错误日志

第 7 章

L1、L2 和 L3“过程流程”步骤相互映射。在图 C-5 中，本练习的几个示例需求被映射到了 L3 过程流程中的相应步骤。注意，在本练习中，业务规则与需求放在同一行。

L1 过程步骤	L2 过程步骤	L3 过程步骤	功能	需求 ID	需求	业务规则
确定要补货的产品	确定每种商品的补货量	产品卖得快吗?	补货	REQ001	自动和事先定义的一个阀值比较历史销售率	BR1：默认期限为上三个月
确定要补货的产品	确定每种商品的补货量	产品卖得快吗?	补货	REQ002	销售率阀值可以基于利润（margin）、销售量（units）或收入（revenue）	BR1：默认阈值基于利润 BR2：默认利润阈值比上三个月的利润低 5%
确定要补货的产品	确定每种商品的补货量	产品卖得快吗?	补货	REQ003	可以比较任意期限的销售率阀值	
确定要补货的产品	确定每种商品的补货量	确定补货量	补货	REQ004	系统自动补货低于阀值的商品	BR1：所有产品的默认阈值是 20(件)
确定要补货的产品	确定每种商品的补货量	确定补货量	补货	REQ005	若计算的自动补货超出阀值，系统请求手动补货	BR1：系统每天在午夜检查库存
确定要	确定每	还有商	补货	REQ006	系统自动决	

补货的产品	种商品的补货量	品要检查吗？			定哪些商品需进行补货检查	

图 C-5 网店补货过程的“需求映射矩阵”（RMM）

第 8 章

为了创建部门组织结构图，先在总裁之下创建第二级，其中包括场景描述中提到的所有部门。然后，咨询利益相关方以确定是否遗漏了任何部门。例如，可能发现在“财务”部门下还有一个“税务”部，必须咨询该部门，了解如何正确计算网上订单的销售税。一定要包括所有部门，这样就可以在确定其中一些部门不在范围内时，自信地将其进行删除。在这个过程中，可能发现一个被忽视的部门，而你需要了解它的具体需求。例如，在这个练习中，公司的“IT”部门肯定需要更详细地了解。图 C-6 展示了这个练习的组织结构图。

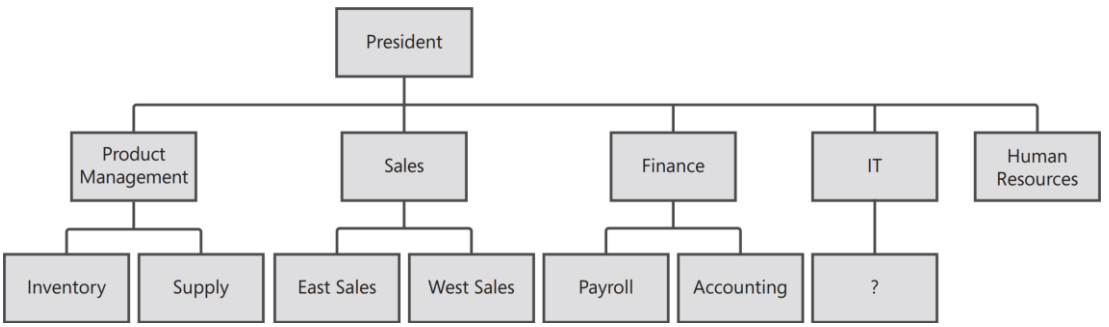


图 C-6 网店（eStore）团队的部门组织结构图

图中译文：

President：总裁

Product Management：产品管理

Sales：销售

Finance：财务

IT：IT

Human Resources：人力资源

Inventory：库存

Supply：供应

East Sales：东部地区销售

West Sales：本部地区销售

Payroll：薪资

Accounting：会计

在图 C-7 的角色组织结构图中，要写上每个部门中存在的实际角色。理想情况是，保留部门组织结构图中对项目重要的部门，并删去其他部门。记住，IT 部门此时尚未确定，这是一个危险信号，提醒你要开始与利益相关方交谈，了解该部门谁需要参与进来。

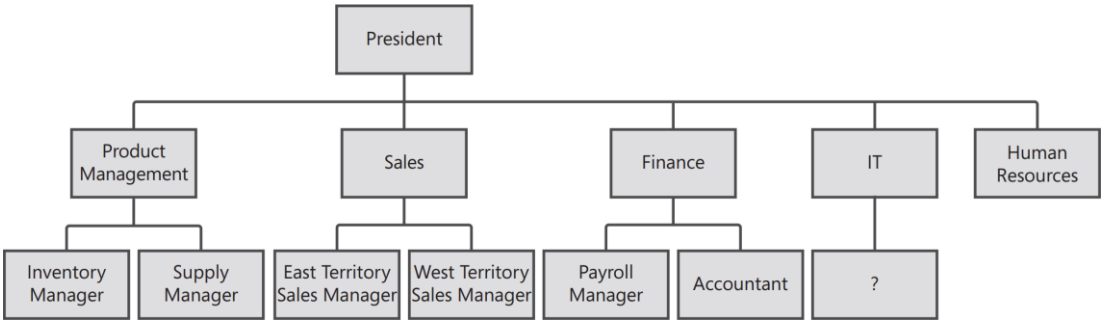


图 C-7 网店（eStore）团队的角色组织结构图

图中译文：

President：总裁

Product Management：产品管理

Sales：销售

Finance：财务

IT：IT

Human Resources：人力资源

Inventory Manager：库存经理

Supply Manager：供应经理

East Territory Sales Manager：东部地区销售经理

West Territory Sales Manager：本部地区销售经理

Payroll Manager：薪资经理

Accountant：会计师

为了创建如图 C-8 所示的个人组织结构图，可以在纸上写下包含姓名和角色的方框，然后按工作关系把他们连接起来。

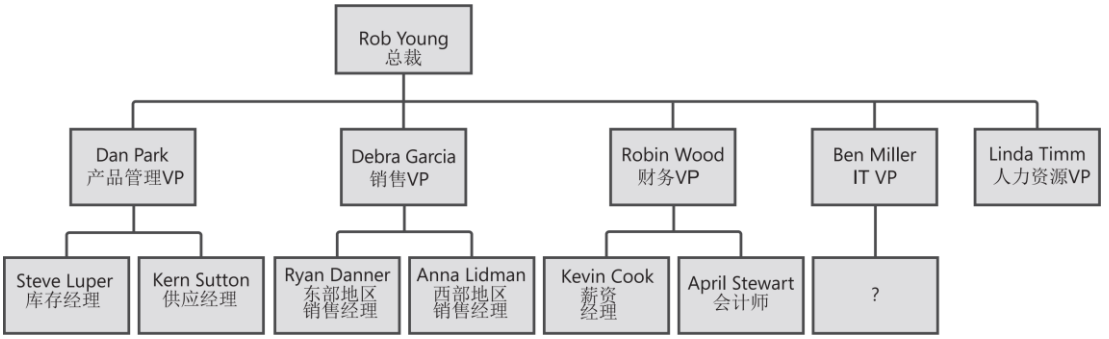


图 C-8 网店（eStore）团队的个人组织结构图

最后，可以看到在个人组织结构图的“IT”部门下仍然有一个问号。因此，要与 Ben Miller 合作来了解他的那个部门，从而确保没有人被排除在需求征询会议之外。这一点非常重要。

第 9 章

L1 过程流程从很高的级别显示了从头到尾的整个过程，如图 C-9 所示。

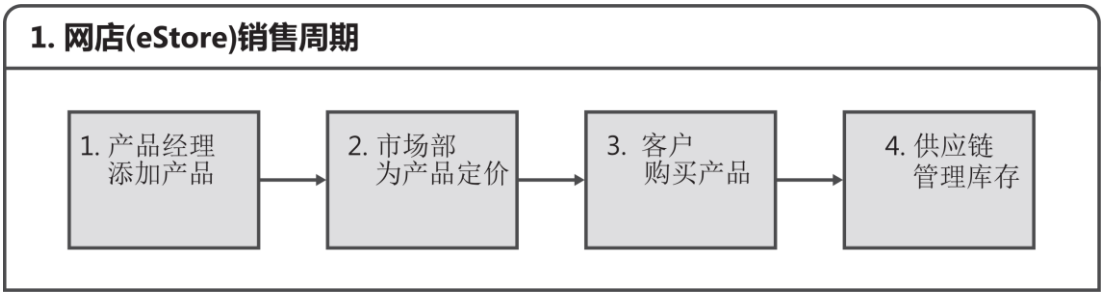


图 C-9 “网店（eStore）销售周期”L1 过程流程

然后，可以为“客户购买产品”L1 步骤创建一个 L2 过程流程。图 C-10 展示了一个例子。

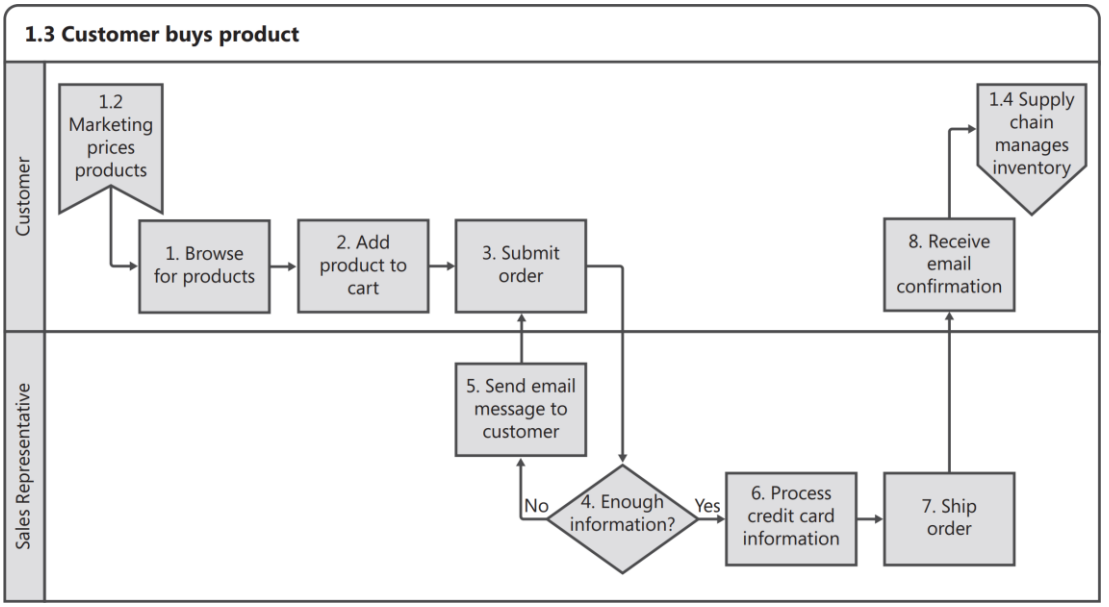


图 C-10 “客户购买产品”L2 过程流程

图中译文：

Customer：客户

Sales Representative：销售代表

1.3 Customer buys product：1.3 客户购买产品

1.2 Marketing prices products：1.2 市场部为产品定价

1.4 Supply chain manages inventory：1.4 供应链管理库存

1. Browse for products：1.浏览产品

2. Add product to cart：2.将产品添加到购物车

3. Submit order：3.提交订单

8. Receive email confirmation：8.接收确认电子邮件

No：否

Yes：是

5. Send email message to customer：5.向客户发送电子邮件

4.Enough information?：4.信息足够吗？

6. Process credit card information：6.处理信用卡信息

7. Ship order：7.订单发货

最后，可以创建一个 L3 过程流程，针对“验证是否有足够的信息来处理订单”过程来完善细节。图 C-11 展示了一个例子。

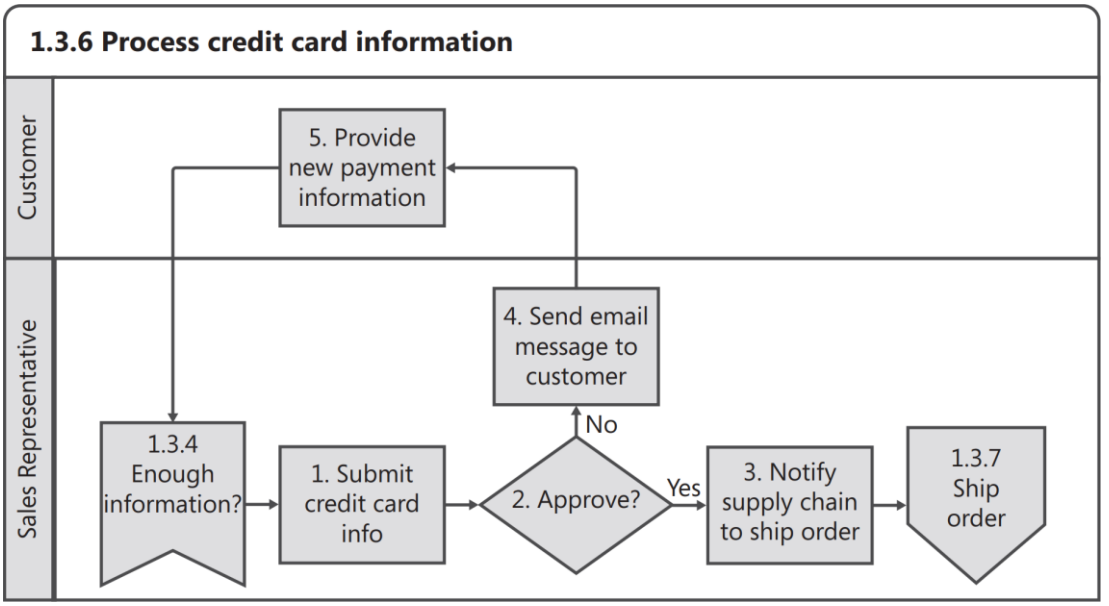


图 C-11 “处理信用卡信息”L3 过程流程

图中译文：

1.3.6 Process credit card information：1.3.6 处理信用卡信息

Customer：客户

Sales Representative：销售代表

5. Provide new payment information：5.提供新的付款信息

4. Send email message to customer：4.向客户发送电子邮件

1.3.4 Enough information?：1.3.4 信息足够吗？

1. Submit credit card info：1.提交信用卡信息

2. Approve?：2.核准了吗？

No：否

Yes：是

3. Notify supply chain to ship order：3.通知供应链发货

1.3.7 Ship order：1.3.7 订单发货

第 10 章

图 C-12 展示了针对该场景的一个示例用例。这是一个简单的用例，其中不包括搜索字段或产品显示字段。那些是从用例推导出的需求。

下面是来自这个用例的示例需求：

- 来自步骤 1：产品可以按照<product.SKU>或<product.name>进行搜索
- 来自步骤 3：搜索结果中包含<product.name>，<product.description>，<product.image>和<product.date>。
- 来自步骤 5：产品在显示时使用了产品数据系统中的以下字段：<product.name>，<product.description>，<product.image>，<product.datecreated>，<product.supplier>和<product.colors>。
- 来自步骤 5：用户可以选择发布从搜索结果中选择的一个产品

还要注意，如果产品已存在于网店中，那么会发生一个 异常情况。这可能会提示我们与产品经理对话，确定他们是否真的想让这些产品出现在搜索结果中。又或者，他们是真的想让这些产品出现在搜索结果中，但要用一个标志指出它们已存在于网店中。要点在于，在这个例子中，用例促使读者思考用户进行的交互，以帮助发现额外的需求。

名称	向网店添加产品
ID	UC002
描述	一名管理员从主产品数据库将一种产品添加到网店（上架），使顾客能看到并购买。
参与者	网店产品经理。
组织收益	让网购顾客看到并购买产品，以增加销售。
使用频率	每月更新两次网店产品和库存，每次大约上架 28 种新产品。
触发器	用户选择“添加新产品”选项。
前置条件	用户用管理员身份凭据登录。有关该产品的所有必要的信息已经存储在产品数据系统中。
后置条件	产品在网店中显示。
主路径	1. 系统显示 SKU 搜索参数。

	2. 用户输入产品的全部或者部分 SKU（例如一种手机的颜色、容量等），选择搜索(参见 AC1)。 3. 系统显示从产品数据系统查到的匹配产品。 4. 用户选择将产品添加到网店 (参见 AC2) 5. 系统显示网店产品数据系统中该 SKU 的信息，提供一个将产品发布到目录（上架）的选项 6. 用户选择将产品发布到目录(参见 EX1) 7. 系统显示产品在网店中的预览供审核。 8. 用户选择最终发布产品（参见 AC3）。 9.系统将用户跳转到网店中的新产品页面(参见 EX2)
替代路径	AC1 用户选择按名称添加产品。 1.系统显示名称搜索参数。 2.用户输入完整或者部分名称，选择搜索。 3.回到主路径的步骤 3。 AC2 产品不在搜索结果列表中，或者没有返回任何结果。 1. 回到主路径的步骤 1。 AC3 用户不喜欢产品的预览效果，希望取消。 1. 回到主路径的步骤 3。
异常情况	EX1：产品已经存在于网店目录中 1.系统显示报错消息。 2. 回到主路径的步骤 1。 EX2：系统添加产品到网店失败。 1.系统显示报错消息。 2. 回到主路径的步骤 9。

图 C-12 向网店（eStore）添加产品的用例

第 11 章

图 C-13 展示了一个角色和权限矩阵，其中覆盖了一些角色以及目录、购物车和账户功能显然，完整的系统会有更多的角色和权限，这里只列举了一部分。

在对场景进行建模后，你可以向业务团队提出更多的问题，例如：

- 是否有定制的销售报告？如果有，谁有权创建？
- 谁有权从目录中删除商品？
- 谁能向目录中添加商品——某个产品经理和某个经理都可以吗？这一点在场景描述中没有明确。

Roles and Permissions Matrix	External Users	Customer	Internal Users	Product Manager	Account Representative	Manager
Catalog and Cart						
Set up product catalog				X		
View products		X		X	X	X
Purchase products		X				
Add item to catalog				X		X
View sales reports						X
View shopping cart		X			X	
User Information						
View own order history		X				
View any customer's order history					X	
Edit user profile		X			X	

图 C-13 网店（eStore）目录、购物车和帐户功能的角色和权限矩阵

图中译文：

Roles and Permissions Matrix：角色和权限矩阵

Catalog and Cart：目录和购物车

Set up product catalog：设置产品目录

View products：查看产品

Purchase products：购买产品

Add item to catalog：向目录添加商品

View sales reports：查看销售报告

View shopping cart：查看购物车

User information：用户资料

View own order history：查看自己的订单历史

View any customer's order history：查看任意客户的订单历史

Edit user profile：编辑用户个人资料

External Users：外部用户

Customer：客户

Internal Users：内部用户

Product Manager：产品经理

Account Representatives：帐户代表

Manager：经理

第 12 章

图 C-14 展示了网店（eStore）的生态系统图。注意，场景所描述的基本系统在这个生态系统图中是一目了然的。另外，还标注了业务数据对象及其在系统之间的流动方向。

下面列出了基于生态系统图的这份初稿可以提出的一些问题：

- 库存系统似乎没有与产品目录系统集成，只在 eStore 和呼叫中心进行了集成，这是正确的吗？
- 订单的运费和税是如何计算的？例如，可能需要发现一个计算税或运费的系统。
- 还有什么遗漏吗？问问总没有坏处！你可能发现原本一无所知的系统。

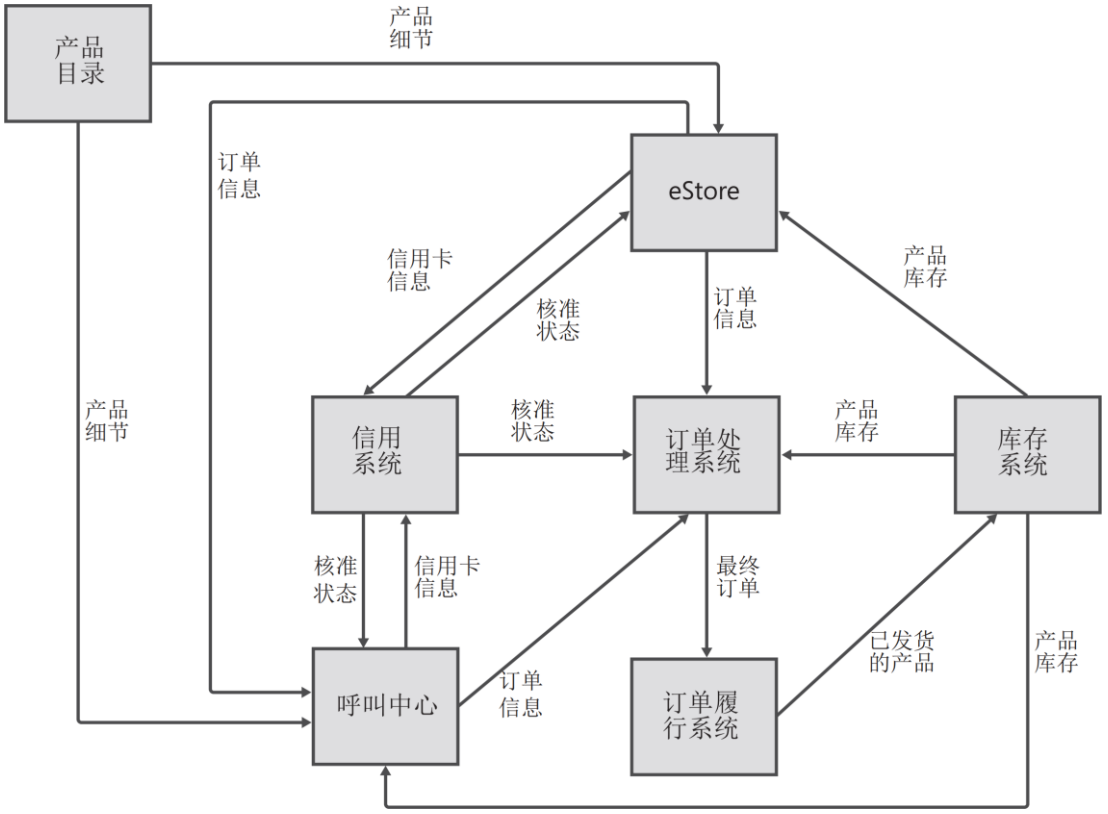


图 C-14 网店（eStore）的生态系统图

第 13 章

图 C-15 的 L2 系统流程展示了从处理订单到发货的系统步骤。

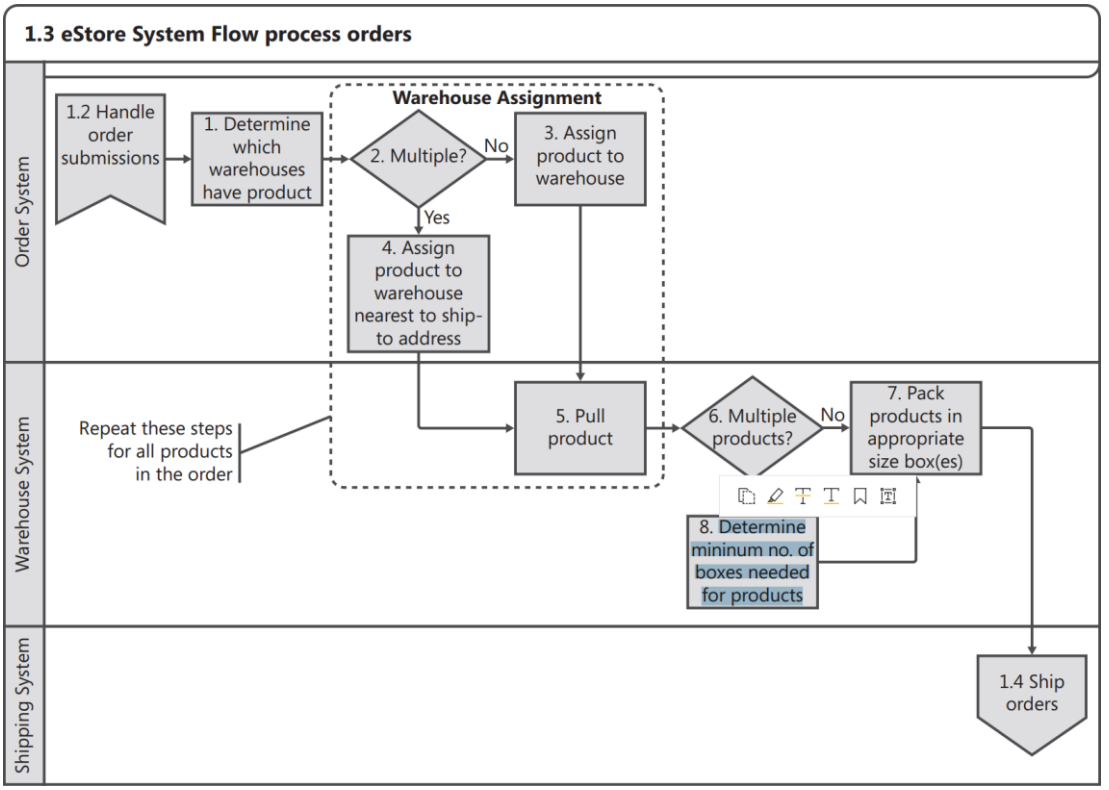


图 C-15 处理网店（eStore）订单并完成发货的 L2 系统流程

图中译文：

eStore System Flow process orders：网店的订单处理系统流程

Order System：订单系统

Warehouse System：仓储系统

Shipping System：发货系统

Handle order submissions：处理订单提交

Determine which warehouses have product：判断哪些仓库有产品

Multiple?：有多个仓库吗？

Yes：是

No：否

Warehouse Assignment：仓库分配

Assign product to warehouse：将产品分配给该仓库

Assign product to warehouse nearest to ship to address：将产品分配给离地址最近的仓库

Repeat these steps for all products in the order：为所有订单产品重复这些步骤

Pull product：从货架取出产品

Multiple products?：多种产品吗？

Pack products in appropriate size box(es)：用合适大小的包装箱打包产品

Determine mininum no. of boxes needed for products：判断产品所需的最小包装箱数量

Ship orders：订单发货

第 14 章

图 C-16 的 UI 流程包含从场景描述中确定的所有主要屏幕，并创建了其中最重要的一些（屏幕）过渡线。某些些过渡线上标注了 UI 触发器，因为这些触发器不是特别显眼。例如，从购物车到检查购物车是否存在商品的决定框，两者之间的过渡线加上了标注，因为不好一眼看出这是结账导航路径。

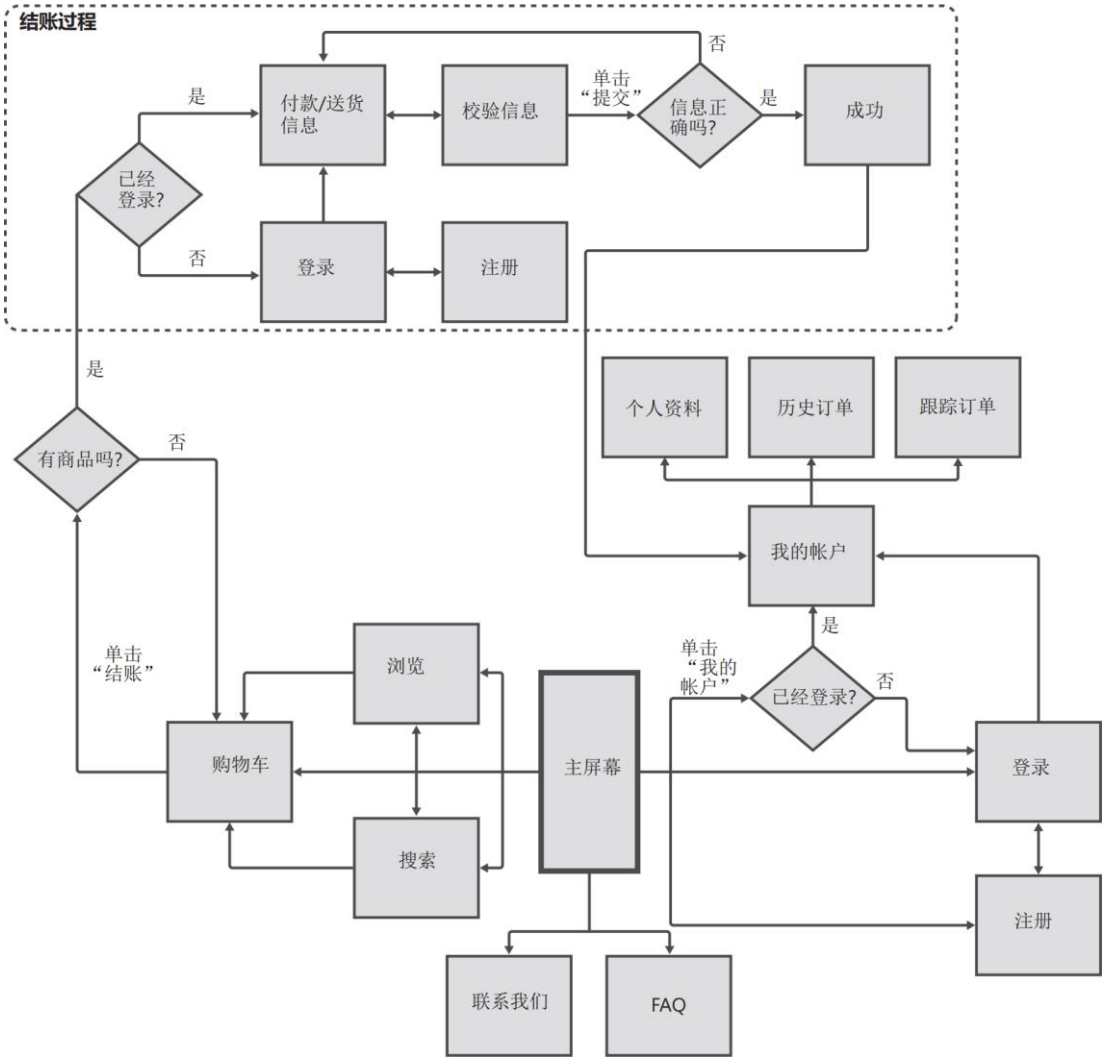


图 C-16 整个网店（eStore）的 UI 流程

第 15 章

图 C-17 显示了颜色选择器的 UI 元素表。

UI 元素：颜色筛选器		
UI 元素描述		
ID	ELMT_0048	
描述	按颜色筛选产品的颜色选择器	
UI 元素显示		
前置条件	**显示**	
始终	针对可用的产品，由<product.color>所有可能的值构成的一个网格，默认选择粉色。 	
UI 元素行为		
前置条件	**操作**	**响应**
始终	选择一个颜色	系统突出显示所选颜色。系统筛选搜索结果，只显示<product.color>和当前所选颜色匹配的产品。
始终	选择一个额外的颜色	系统突出显示所选颜色。系统筛选搜索结果，只显示<product.color>和当前所选颜色匹配的产品。同时应用其他所有选定的筛选器。
始终	选择已经突出显示的颜色	系统从当前选定的所有颜色中移除这种颜色。系统移除这种颜色的筛选器。
始终	选择“View all”（查看全部）	系统移除所有颜色的突出显示，显示<product.color>为任意颜色的产品。

图 C-17 网店（eStore）产品搜索颜色过滤器的显示-操作-响应表

第 16 章

因篇幅有限，完整决策表在书中放不下。所以，图 C-18 显示的是一个简化的决策表。注意，如果选择没有足够资金的礼品卡进行支付，那么会有两种有效的结果，而且它们的应用顺序是固定的。

决策表		规则 1	规则 2	规则 3	规则 4	规则 5	规则 6	规则 7
条件								
客户选择使用已保存的支付方式		Y	Y	N	N	N	N	N
支付方式有效		Y	N	-	-	-	-	-
新的支付方式		-	-	信用卡	信用卡	支票	礼品卡	礼品卡
信用卡类型		-	-	Contoso	A. Datum	-	-	-
有足够资金		-	-	-	-	-	Y	N
结果								
OC001 使用已保存的支付方式		X	-	-	-	-	-	-
OC002 提供对已保存的支付方式进行编辑的选项		-	X	-	-	-	-	-
OC003 获取信用卡号和 3 位安全码		-	-	X	-	-	-	-
OC004 获取信用卡号和 4 位安全码		-	-	-	X	-	-	-
OC005 获取账户信息、驾照号码和地址		-	-	-	-	X	-	-
OC006 使用礼品卡		-	-	-	-	-	X	1

OC007 为剩余待支付金额选择支付方式		-	-	-	-	-	-	2

图 C-18 该决策表描述了网店（eStore）在应用支付方式时的规则

第 17 章

为场景创建决策树时，要考虑决策的顺序。必须做出的第一个决策是客户是否在其个人资料中保存了一种支付方式。可能的选择是，客户要么保存了一种支付方式，要么没有。在确定了这些选择之后，接着要确定这些选择可能带来哪些新的决策或结果。例如，如果客户确实有一种支付方式，新的决策就是判断该支付方式是否有效。如此不断地确定决策、选择和结果，最终应该得到一个类似于图 C-19 的有序决策树。

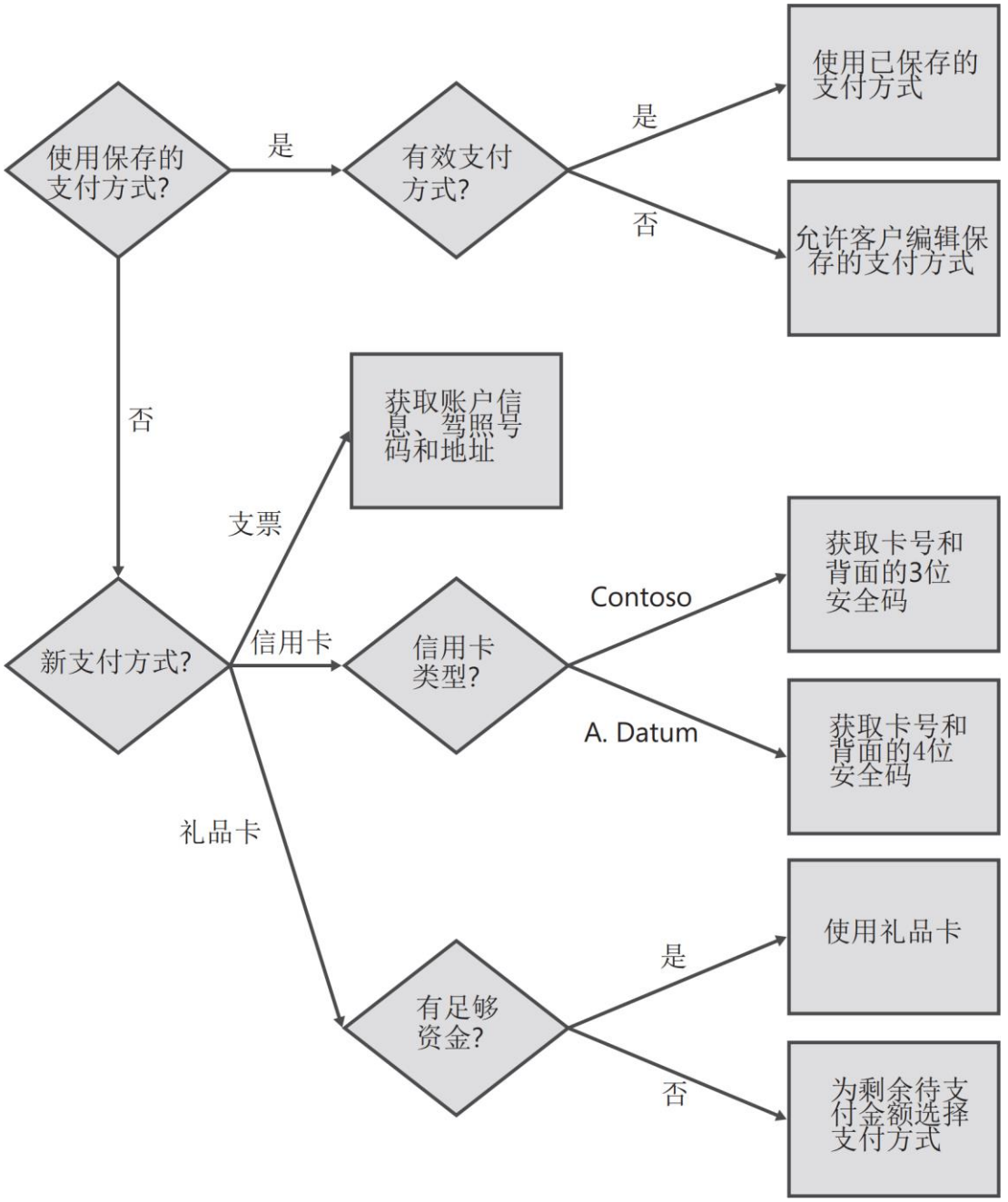


图 C-19 该决策树描述了网店（eStore）在应用支付方式时的规则

第 18 章

图 C-20 为库存系统和网店（eStore）之间的接口显示了系统接口表。存在多个接口，但只需描述产品库存数据传输的细节。注意，图中引用了用于错误处理的系统流程；在真实的

项目中，你需要找出确切的流程编号，并在这里引用该编号。在本例中，产品对象只有两个字段需要传输：作为标识符的 SKU，以及可用的产品数量。对于这些字段，没有特别的规则需要覆盖它们。

系统接口			
来源	库存系统		
目标	网店（eStore）		
ID	SI0012		
描述	库存系统用产品库存信息更新网店，以显示可用性		
频率	每天午夜更新一次		
数据量	所有产品（约 500 种）		
安全限制	无		
错误处理	参考用于错误处理的系统流程 1.1		
接口对象			
对象	**字段**	**数据字典 ID**	**校验规则**
产品	SKU	DD001	
产品	可用数量	DD001	

图 C-20 对库存系统和网店（eStore）之间的接口进行描述的系统接口表

第 19 章

在场景描述中，包含了一些无关紧要的信息，但一个好的分析师应过滤这些信息，以判断哪些与模型相关。例如，业务目标并不能直接引导你发现 BDD 中的对象，地址中的字段也不能。然而，功能列表应该能帮助你确定一些对象，例如客户、订单和商品。用户列表清楚地表明，其中的一些用户是公司，它们有自己的客户；还有一些个人客户，它们不属于任何公司。图 C-21 展示了最终的 BDD。注意，地址是和商品连接的，而不是和订单连接，因为客户可能选择将一个订单中的不同商品发到不同的地址。

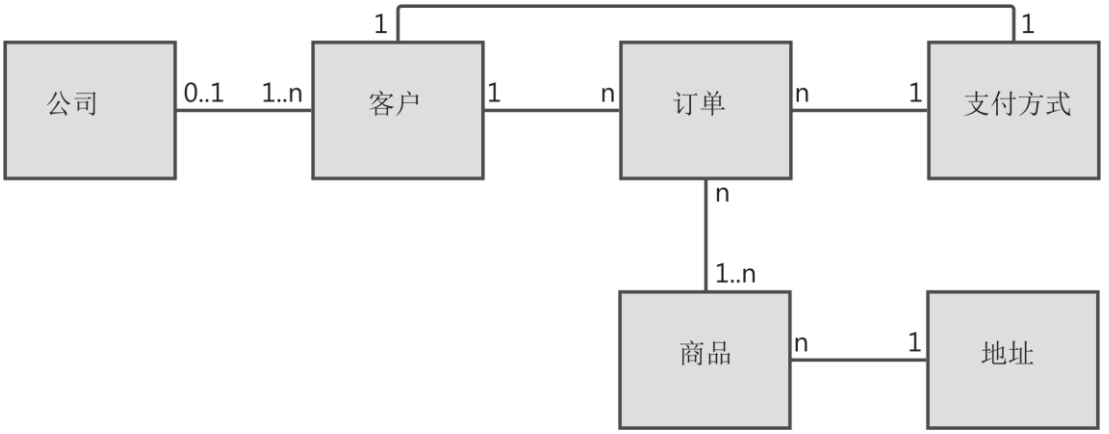


图 C-21 网店（eStore）业务数据对象的 BDD

第 20 章

在这个场景中，明显的业务数据对象有购物车、订单和产品。要考虑的第一组过程是创建购物车、确认订单和维护产品。明显的实体有购物者、产品经理和订单履行。将这些放到图中，更多的过程就变得明晰了。图 C-22 是 DFD 的初稿。

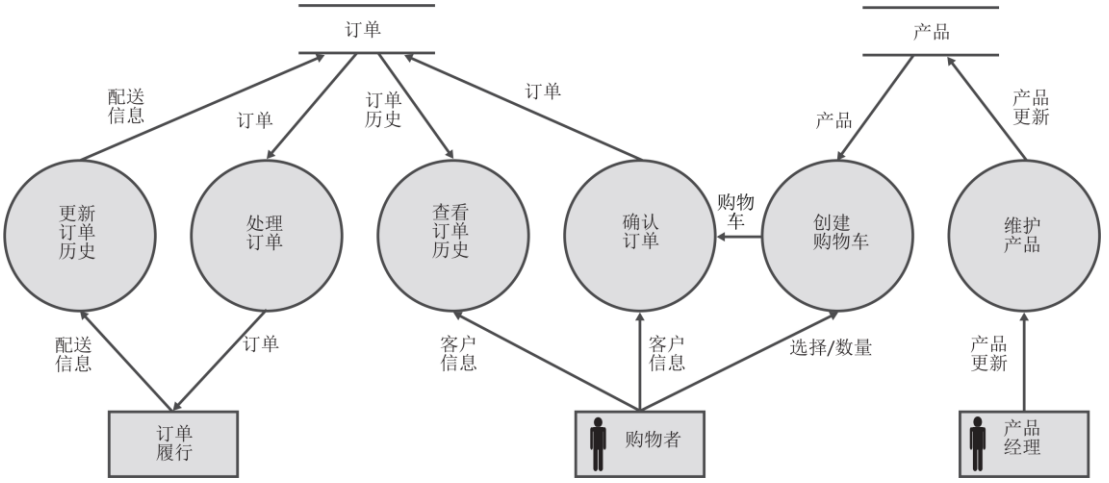


图 C-22 网店（eStore）的数据流图（DFD）

第 21 章

使用场景所提供的信息，并根据经验推断出其他信息，就可以创建一个如图 C-23 所示的数据字典。其中，“折扣价”和“是否有货”字段都是计算字段。

信息尽可能靠推断，这是一种很好的技术，能节省时间。当然，务必确认推断出的信息是正确的。在这个示例答案中，我们根据以往的经验来推断数据类型、有效值和所有字段的

长度。另外，还推断了如何计算“是否有货”字段值，并建议将“折扣价”四舍五入到“分”。

可能需要为其他属性获取信息。一般来说，应获取所有必要和推荐属性的信息。但是，如果时间紧张，或者信息存在于其他地方，那么可以暂且将数据字典限制在一组核心属性上。

ID	业务数据对象	字段名称	描述	数据类型	有效值	长度	值需要唯一吗？	必须吗？	默认值	计算
DD01	产品	成本	网店采购或生产该产品的成本价	美元格式：($999,999.99)	0.00...999,999.99	10	否	是	$100.00	N/A
DD02	产品	标价	在网店目录中显示的产品售价	美元格式：($999,999.99)	0.00...999,999.99	10	否	是	$120.00	N/A
DD03	产品	折扣价	高级客户的折扣价	美元格式：($999,999.99)	0.00...999,999.99	10	否	否	N/A	0.80 * 标价，四舍五入到分
DD04	产品	SKU	对产品进行唯一标识的代码	Alphanumeric	10 个字母和数字的唯一组合	10	是	是	9999999999	N/A
DD05	产品	数量	该 SKU 目前的存货数	Integer	0...999,999	6	否	是	0	N/A
DD06	产品	是否有货	指出产品是否	Boolean	True，False	N/A	否	是	False	数量 > 0

有货

图 C-23 网店（eStore）的“产品”业务数据对象的数据字典

第 22 章

需要状态表的对象是“订单”。该对象的第一组状态可能包括“挑选”（drafted）、“决定”（finalized）、“计价”（priced）、“完成”（completed，这个状态要求收集好配送和支付信息）、“确认”（confirmed）、“组装”（assembled）、“发货”（shipped）和“收货”（received）。状态表的初稿如图 C-24 所示。随着项目的进行，所了解情况的增多，它可能会有进一步的变化，但现在已接近完成。

		目标状态							
		挑选	**决定**	**计价**	**完成**	**确认**	**组装**	**发货**	**收货**
初始状态	**挑选**	否	用户选择结账	否	否	否	否	否	否
	决定	否	否	计算好税和折扣	否	否	否	否	否
	计价	用户编辑订单	否	否	用户输入有效配送和支付信息	否	否	否	否
	完成	用户编辑订单	否	否	否	用户确认购买	否	否	否
	确认	工厂不能履行订单	否	否	否	否	工厂生产	否	否
	组装	否	否	否	否	否	否	工厂发货	否
	发货	否	否	否	否	否	否	否	订单送达
	收货	否	否	否	否	否	否	否	否

图 C-24 网店（eStore）的“订单”业务数据对象的状态表

第 23 章

需要状态图的一个显而易见的对象就是“订单”。图 C-25 是和所提供的状态表对应的状态图。

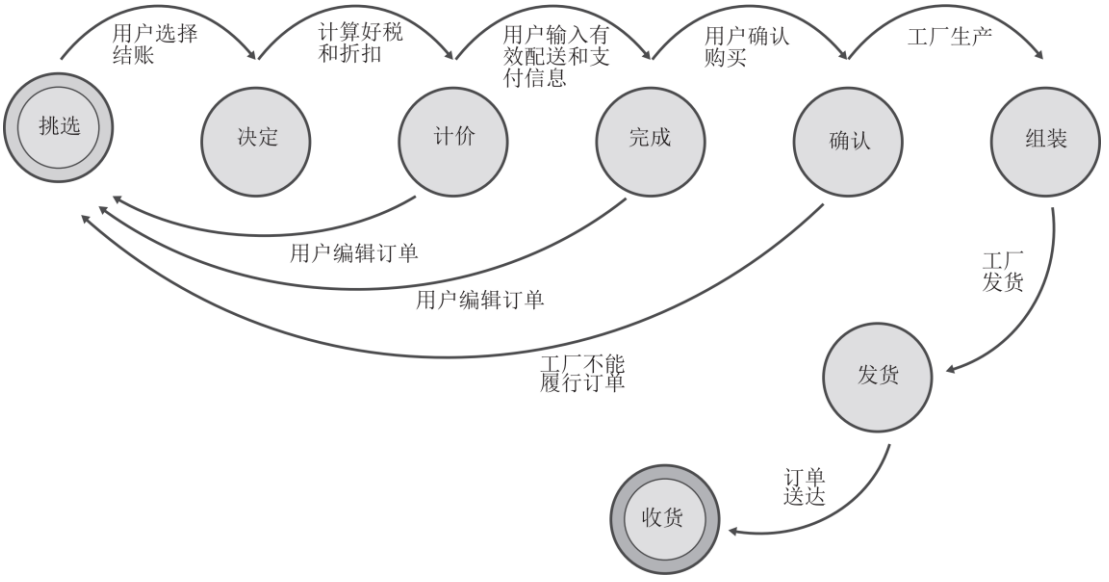


图 C-25 网店（eStore）的“订单”业务数据对象的状态图

第 24 章

图 C-26 针对书中给定的模型报告给出了一个“报告表”。

	元素	**呼叫处理报告**
顶级元素	**唯一 ID**	REP003
	名称	CSR 呼叫处理时间
	描述	该报告显示了每名 CSR 应答的支持电话数量，以及每名 CSR 解决客户问题的平均时间。
	依据报告做出的决策	每季度哪名 CSR 应得到晋升。
	目标	业务目标 3 – 将员工保留率提升至 80%。
	优先级	3 of 5。参考目标链 003。

	功能区域	销售
	相关报告	N/A
	报告所有者（负责人）	呼叫中心总监
	报告(的)用户	呼叫中心总监及其助理
	触发器	用户请求报告
	频率	按需生成。每月第三个周五访问
	延迟	数据应该是实时的
	事务处理量	每天在报告上添加约 30 个呼叫
	数据量	报告平均返回 300 个呼叫
	安全性	仅供呼叫中心总监及其助理查看
	持久性	保存报告执行之间的所有设置（下次看到上次的设置）
	显示格式	矩阵显示，CSR 姓名作为行，日期作为行，“每天呼叫数”和“平均呼叫处理时间”作为列
	交付格式	在应用程序中显示，可作为 Excel 文件通过电子邮件发送。在应用程序中，一屏显示不全的数据可以滚动查看。
	交互性	CSR 姓名可以展开或折叠，以显示或隐藏每名 CSR 的呼叫日期。默认为展开。
	下钻	N/A
字段元素	**筛选依据**	系统预设筛选器是<呼叫.日期> = 当月。用户可从 12 个月中选择不同的一个月。
	分组依据	系统预设分组，先按<CSR.姓名>，再按<呼叫.日期>，再按所有呼叫。
	排序依据	系统预设排序，先按<CSR.姓名>以字母顺序排序，再按<呼叫.日期>以时间顺序排序
	用户输入参数	N/A

	分组计算	按日期和按 CSR，显示指定月份的呼叫总数 按日期和按 CSR 的平均呼叫处理时间 = sum(当天所有单独呼叫的处理时间) / 当天呼叫总数 按月和按 CSR 的平均呼叫处理时间 = sum(当月所有单独呼叫的处理时间) / 当月呼叫总数
	计算字段	N/A
	显示字段	<CSR.姓名> <呼叫.日期> 呼叫计数 平均呼叫处理时间 平均呼叫处理时间应四舍五入为两位小数
	模拟(what-if)分析	N/A

图 C-26 为网店（eStore）呼叫处理报告设计的一个“报告表”

第 25 章

场景所描述的是一个“绿地”项目（一个从头开始定制的全新系统），其特征包括广泛的用户交互、面向客户、包含分析和报告组件、工作流自动化、业务过程自动化以及 Web 应用。

肯定需要创建一个业务目标模型和一个目标链，以确定需求的优先级。KPIM 也许有帮助，因为当前过程是手动的（依靠人工电话）。可以使用 KPIM 来确保企业在使用新系统接单时不会损失任何总体运营效率。功能树有助于向高管传达完整的功能集合。RMM 对于将需求映射回过程流程和业务目标非常关键，目的是尽可能防止范围蔓延（scope creep），这是实现新系统时最害怕的。

可能需要创建一个组织结构图来确定系统的内部用户。肯定需要创建过程流程来显示用户如何在新系统中完成订单。生态系统图对于展示网店与其他现有系统（例如订单履行）的交互是必要的。系统流程可以显示订单提交后发生的自动交互。UI 流程和 DAR 模型对于确保用户轻松使用新网站至关重要。BDD 和数据字典有助于记录业务数据的主要部分，例如客户和订单对象及其字段。

除此之外，角色和权限矩阵、状态表、状态图、报告表、DFD 和用例也可能有帮助。如

果要为内部用户建立一个安全模型，那么需要用到角色和权限矩阵。如果主要是外部用户在访问 UI，而且他们都有相同的权限，那么这个模型就没什么帮助了。状态表和状态图可以用来表示订单在经过一个定义的工作流时的不同状态。报告表可以用来记录负责人对于报告的要求。用例可以进一步描述用户在系统中发生的交互。可能需要一个 DFD 来描述报告所显示的数据流。

图 C-27 展示了项目特征、为这些特征推荐的模型以及最终选定的模型。

	目标						**人员**					**系统**							**数据**					
	业务目标模型	目标链	关键绩效指标模型 (KPIM)	功能树	需求映射矩阵 (RMM)		组织结构图	过程流程	用例	角色和权限矩阵		生态系统图	系统流程	UI 流程	显示−操作−响应 (DAT) 模型	决策表	决策树	系统接口表	业务数据图 (BDD)	数据流图	数据字典	状态表	状态图	报告表
绿地	L	L		L	L																			
广泛用户交互								L	M	L				L	L									
面向客户								M	M	M				L	L									
业务过程自动化			L				L	L	M	M									L		L			
工作流自动化								L		L									L			L	L	
Web 应用												L	L	M	M				L		L			
分析和报告								M								M	M		L	L	L			L
为此场景选择的模型	x	x	x	x	x		x	x	x	x		x	x	x	x				x	x	x	x	x	x

图 C-27 网店（eStore）项目的特征和推荐模型

第 26 章

可以采取以下方式来综合运用场景所描述的各种模型：

- 过程流程记录网店所支持的主要行动（操作）。
- 过程流程中的每个步骤都可与一个或多个 DAR 模型联系起来，以确保用户界面易于导航和使用。
- BDD 定义了业务数据对象。每个对象都包含了在一个数据字典中定义的字段。
- DAR 模型还要配合数据字典使用，使所有数据需求都能在屏幕和底层存储系统中得以体现。
- 可以使用数据字典来确定哪些业务数据对象包含的状态会影响网店并需要状态图。
- RMM 可以将过程流程映射到 DAR 模型，再映射到需求。

词汇表

7±2：参见“米勒魔数”。

活动图（activity diagram）：用于可视化呈现复杂流程的一种模型，通常用于为用例提供补充。使用类似于过程流程的符号，但在同一幅图中显示用户操作和系统响应。参见“过程流程”和“用例”。

参与者（actor）：参见“用户”。注意，系统不是参与者。

亲和图（affinity diagram）：一种简单地将信息项组织成相关分组的图。在需求征询期间时，是将大量信息快速组织成较小分组的一种好方法。经常在头脑风暴会议上使用。如果向其应用米勒魔数，那么效果更佳。

方法（approach）：实现一个项目所遵循的过程的类型。

界定模型（bounding model）：创建这种模型后，有很高的概率能捕捉（采集）该模型所针对的全部信息。RML 界定模型包括业务目标模型、组织结构图、生态系统图和业务数据图。参见“模型”。

业务数据图（Business Data Diagram，BDD）：从业务利益相关方或客户的角度显示业务数据对象之间关系的一种 RML 数据模型。显示对象之间的包含（has a）关系，以及集合的基数或 cardinality（多对多、多对一、一对一等）。

业务数据对象（business data object）：任何对业务利益相关方有意义的概念性数据。由字段以及/或者其他业务数据对象构成。

业务对象（business object）：参见“业务数据对象”。

业务目标（business objective）：一种可度量的目标，规定了如何判断业务问题已得到解决。

业务目标模型（Business Objectives Model）：一种 RML 目标模型，它定义并关联了业务问题、业务目标、产品概念和成功指标。用于确定一个项目的价值。

业务问题（business problem）：阻碍业务利益相关方实现其目标的一个问题。

业务过程（business process）：由业务用户执行的一组活动。可以用一个过程流程来描述。

业务规则（business rule）：一种需求（requirement），代表对功能需求（functional requirement）进行修饰的条件陈述（conditional statement），其中包括但不限于功能何时可用以及谁被允许执行该功能。业务规则包含了像“如果”（if），“当”（when）和“那么”（then）这样的词。参见“需求”和“功能需求”。

业务利益相关方（business stakeholder）：要使用软件或者与软件利益攸关的人（或团体），

他们通常为生产软件的公司或组织工作。业务利益相关方是利益相关方的一种类型。参见“利益相关方”。

业务用户（business user）：要使用软件的人，通常为生产软件的公司或组织工作。

基数（cardinality）：一个业务数据对象有多少个实例与另一个业务数据对象相关。

类图（class diagram）：通过系统类（system classes）来描述系统结构的一种 UML 图（Eriksson, Hans-Erik, and Magnus Penker. 2000. *Business Modeling with UML*. New York, NY: Wiley）。参见“统一建模语言”（UML）。

协作图（collaboration diagram）：对一组软件对象之间的交互进行描述的一种 UML 图（Eriksson and Penker 2000）。参见“统一建模语言”（UML）。

组件图（component diagram）：对一个系统的技术组件之间的关系进行描述的一种 UML 图（Eriksson and Penker 2000）。参见“统一建模语言”（UML）。

跨功能过程流程（cross-functional process flow）：参见“过程流程”。

客户（customer）：在实现软件的那个公司或组织外部的人员，他们获得并使用由公司或组织提供的产品或服务。

数据（data）：参见“业务数据对象”。

数据字典（Data Dictionary）：一种 RML 数据模型，描述了系统中的任何业务数据对象的字段。

数据流图（Data Flow Diagram，DFD）：一种 RML 数据模型，显示了信息在解决方案中的流动以及业务数据对象的转换过程。

数据对象（data object）：参见“业务数据对象”。

决策表（Decision Table）：一种 RML 系统模型，描述一组条件的所有可能组合以及相应的结果，用网格表示。

决策树（Decision Tree）：一种 RML 系统模型，描述相关的条件组合以及相应的结果，用树状结构表示。

设计（design）：需求将如何在解决方案（包括 UI）中实现。

图（diagram）：有组织的信息的一种可视化表示。

显示-操作-响应（Display-Action-Response，DAR）模型：一种 RML 系统模型，记录了系统显示屏幕的方式，以及如何响应用户采取的行动。

生态系统（ecosystem）：组织中的一整套解决方案组件，可以包括硬件、软件、人员和数

据等。

生态系统图（Ecosystem Map）：一种 RML 系统模型，显示了解决方案生态系统中所有系统之间的关系。参见“生态系统”。

元素（element）：一个模型的任何组件（例如一个方框）都可以称为一个元素。

实体关系图（Entity Relationship Diagram，ERD）：一种数据库设计模型，用于显示存储在数据库中的概念或物理数据之间的关系。

功能（feature）：对解决方案最终包括进来以满足业务目标的功能区域（area of functionality）的一个简短描述。功能是需求的集合，用于阐述（articulate）和组织（organize）需求。

功能树（Feature Tree）：一种 RML 目标模型，用一种树状结构来显示进行了逻辑分组的所有系统功能。

功能需求（functional requirement）：解决方案在不考虑任何限定条件下可以提供的行为或能力。

目标（goal）：关于业务利益相关方想要实现什么的定性陈述。虽然和业务目标（business objectives）相似，都包含“目标”一词，但前者是定性陈述，而不是可度量的陈述（定量陈述）。

信息技术（Information Technology，IT）：本书特指“IT 部门”或“IT 团队”，即负责具体实现软件项目的群体，旨在为公司内部用户提供服务。

关键绩效指标（Key Performance Indicator，KPI）：度量业务活动成功与否或者目标进展情况的一种量化指标。

关键绩效指标模型（Key Performance Indicator Model，KPIM）：一种 RML 目标模型，将 KPI 与业务过程联系起来，以评估过程的表现（绩效）。参见 KPI。

图（map）：map 是一种特殊的 diagram，中文将两者都翻译为“图”，少数地方会把 map 翻译为“地图”，例如“用户故事地图”。事实上，map 专门用于呈现元素之间关系的一个全景。做动词用的时候（map to），是“映射到”或者“链接到”的意思。

矩阵（matrix）：一种网格结构，以行列交叉点来捕捉（采集）模型细节。矩阵是“表”或“表格”的一种更广义的称呼。

方法论（methodology）：某些活动或学科所遵循的、由方法、原则和规则构成的一个系统。

米勒魔数（Miller's Magic Number）：认知心理学家 George A. Miller 发现，人类只能同时记忆和处理 7 正负 2（记为 7±2）个数据项（Miller, George A 1956,“The Magical Number

Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information”, *Psychological Review* 63, 81-97）。

思维导图（mind map）：一种对信息进行结构化的方式，支持信息的快速组织。进行头脑风暴时，要求信息被快速激发出来，而不是按照特定的顺序，这时思维导图很有用。

模型（model）：对于正在开发的解决方案，其内部和周围会存在一些和过程、数据和交互有关的信息，对这种信息的可视化表示（图片）就是模型。

非功能需求（non-functional requirement）：除功能需求（包括业务规则）之外的其他所有需求。参见“需求”和“功能需求”。

对象（object）：参见“业务数据对象”。

对象图（object diagram）：经常与类图一起使用的 UML 图，用于描述类图中对象的实例（Eriksson and Penker 2000）。参见“类图”。

<*对象.字段*>表示法：本书中用来精确引用数据字段的一种标准表示法。其中，“对象”是业务数据图中的业务数据对象，“字段”是数据字典中的数据字段。

目标链（Objective Chain）：一种 RML 目标模型，使用目标因素和目标方程，以可度量的方式将功能与业务目标联系起来。

洋葱模型（onion model）：一种环状模板，用来显示利益相关方、他们相互之间的关系以及正在开发的产品（Alexander, Ian. 2005. “A Taxonomy of Stakeholders: Human Roles in System Development”， *International Journal of Technology and Human Interaction*，https://tinyurl.com/4x6bfbfv）。

操作（operation）：系统所支持的一个单独的功能。可以是概念性的，也可以是 UI 中的一个实际物理元素。

OPSD：RML 模型分类为目标模型（objectives models）、人员模型（people models）、系统模型（systems models）和数据模型（data models），统称为 OPSD。

有序决策树（ordered Decision Tree）：最常见的决策树类型，在有隐含的决策顺序时使用。使用方向箭头来表示决策顺序和结果。参见“决策树”和“无序决策树”。

组织结构图（Org Chart）：一种 RML 人员模型，显示一个组织内所有人员的结构。用它来帮助我们确定需要从其收集需求的所有可能的用户、利益相关方和主题专家（SME）。

人员（people）：由利益相关方构成的任何群体/群组。

画像（persona）：一个典型的用户，包含了用户的背景信息和使用系统的动机。

过程（process）：为了特定目的或者为了实现特定结果而执行的一系列活动。

过程流程（Process Flow）：一种 RML 人员模型，描述了一个将由人来执行的业务过程（business process）。过程流程显示了要执行的活动、活动的执行顺序以及用户为实现预期结果而做出的不同决定（决策）。过程流程可以使用泳道（swim lanes）来显示在过程流程中执行活动的各种人员。参见“泳道”。

产品概念（product concept）：业务利益相关方为了达成其业务目标而选择实现的实际解决方案的愿景。它通常由包含高层次功能的一个清单来描述。

项目（project）：为了创造一个独特的产品、服务或结果而做出的临时性努力。Project Management Institute (PMI). 2008. *A Guide to the Project Management Body of Knowledge (PMBOK Guide) – Fourth Edition*. Newtown Square, PA: Project Management Institute, Inc.

原型（prototype）：本书特指 UI 的一种可工作的模型（working mockup），用于征询和澄清需求。可以是低保真线框图（wireframe），也可以是高保真模型（支持用户的交互式体验）。

报告表（Report Table）：一种 RML 数据模型，以结构化的方式描述报告需求。包括数据应该如何显示，输出应该如何格式化，还可以包括下钻（drilldown）视图以及操作和交互方面的需求。注意不要将“report table”理解为“报表”，后者是 report 的一种中文翻译。本书的“report”都是“报告”。

需求（requirement）：业务利益相关方（the business）需要在解决方案中实现的任何东西。可以包括功能需求、非功能需求、业务规则和设计。参见“功能需求”、“非功能需求”和“业务规则”。

需求架构（requirements architecture）：需求信息的一种组织方式，包括需求的结构是什么、模型与模型的关系以及模型与需求的关系。

需求映射矩阵（Requirements Mapping Matrix，RMM）：一种 RML 目标模型，将需求和业务规则映射到一个模型，使需求以一种更容易使用的方式进行分组。最常见的是将过程流程或系统流程步骤映射到需求和业务规则。

需求建模语言（Requirement Modeling Language，RML）：一种可视化需求建模语言；由目标、人员、系统和数据（OPSD）模型组成。专为需求建模而设计和使用。参见“OPSD”。

RML：参见“需求建模语言”。

角色（role）：参见“用户角色”。

角色和权限矩阵（Roles and Permissions Matrix）：一种 RML 人员模型，定义了角色的类型及其在系统中执行操作所需的权限。参见“用户角色”。

序列图、时序图或顺序图（sequence diagram）：一种 UML 图，描述了对象之间发送的消息的顺序（Eriksson and Penker 2000）。

解决方案（solution）：用于解决一个业务问题的完整实现。可以包括硬件、软件、业务过程、用户手册和培训。参见“业务问题”。

利益相关方（stakeholder）：与项目利益攸关的个人或团体。IT 部门、业务部门和客户都是利益相关方的例子。他们可能参与、受影响以及/或者影响结果（Wiegers, Karl E. 2013, *Software Requirements, Third Edition*. Redmond, WA: Microsoft Press）。参见“业务利益相关方”、“技术利益相关方”和“客户”。注意，中文版没有采用“干系人”的译法？这是因为有时一个 stakeholder 并不是“人”，而可能是一个部门、一个团队等。

状态（state）：在业务数据对象的生命周期中，不同的阶段可能对系统的行为有不同的影响。状态是对这些阶段的简短描述。

状态图（State Diagram）：一种 RML 数据模型，显示了业务数据对象所有可能的状态，还显示了状态之间的过渡。参见“状态”。

状态表（State Table）：一种 RML 数据模型，显示了业务数据对象所有可能的状态，还显示了状态之间的过渡，以网格形式表示。参见“状态”。

状态图（statechart diagram）：显示系统状态的一种 UML 图（Eriksson and Penker 2000）。类似于 RML“状态图”。

主题专家（SME）：也称为“行业专家”，是与解决方案的某个主题相关的专家。SME 可以是 IT、业务或客户利益相关方。

成功指标（success metric）：为确定项目是否成功而实际度量的业务目标；也可以是与解决方案相关的其他指标。参见“业务目标”。

泳道（swim lane）：过程流程、系统流程或 UI 流程中的一个分组，它们将图划分为多个部分，以直观传达是什么实体在执行步骤。

泳道图（swim lane diagram）：参见“过程流程”。

系统（system）：一个具体的实现，包括用于解决一个业务问题的硬件、软件和业务过程。

系统流程（System Flow）：一种 RML 系统模型，用于描述系统自动执行的活动。它们显示了所执行的活动、执行的顺序以及为了达成一个结果所做出的不同决定（决策）。系统流程可以使用泳道来显示在系统流程中执行活动的各种系统。

系统接口表（System Interface Table）：一种 RML 系统模型，从业务利益相关方的角度描述了两个系统之间的通信，包括必须传输什么信息、传输多少以及传输频率。

表或表格（table）：参见“矩阵”。

团队（team）：参与执行一个项目的全部人员，包括分析师、业务利益相关方和技术利益相关方。

技术利益相关方（technical stakeholders）：负责实际制作软件或者对系统进行配置的实现团队。通常指数据库设计师、架构师、软件开发人员和测试人员。

技术团队（technical team）：参见“技术利益相关方”。

可跟踪性（traceability）：两种类型的对象的所有实例之间的映射。我们比较各种关系，以确保一种对象完全被另一种对象所覆盖（被跟踪到，可追溯到）。

可跟踪性矩阵（traceability matrices）：参见“需求映射矩阵”。

过渡（transition）：一个业务数据对象从一种状态变成化为另一种状态。

UI 元素（UI element）：用户界面中的任何实体，具有依赖于数据的显示或行为属性。UI 元素的例子包括按钮、显示表格、图像或者复选框等。参见“用户界面”。

UI 流程或用户界面流程（UI Flow）：一种 RML 系统模型，显示用户如何在用户界面的屏幕之间导航。参见“用户界面”。

统一建模语言（Unified Modeling Language，UML）：一种用于可视化语言，用于描述面向对象软件系统的设计（Object Management Group. 2007. “OMG Unified Modeling Language Specification.” http://www.uml.org/#UML2.0）。

无序决策树（unordered Decision Tree）：没有方向箭头的决策树；当没有隐含的顺序时，用于简化一组决策。参见“决策树”和“有序决策树”。

用例（Use Case）：一种 RML 人员模型，描述用户和系统之间的交互。用例描述了用户需要做什么，他想完成什么，以及他在使用软件时系统如何响应。

用例图（use case diagram）：描述用例之间关系的一种 UML 图（Eriksson and Penker 2000）。

用户（user）：直接使用一个系统来达成目标的人。

用户界面（User Interface，UI）：软件系统中的屏幕，用户通过它与系统进行交互。

用户界面设计需求（user interface design requirement）：表示系统外观和特定 UI 元素行为方式的一种需求。

用户角色（user role）：共享相同的功能并访问同一个系统的用户集合。

用户故事（user story）：一种捕捉需求的敏捷方法。包含一个名称，对用户想要做什么的描述，以及用于确定用户故事在什么时候被软件正确实现的验收标准。

确认（validation）：检查需求，确保这些需求是实现项目的业务目标所需要的。

验证（verification）：检查需求，确保它们将导致一个起作用的解决方案。

线框或线框图（wireframe）：用来显示系统的用户界面的屏幕模拟。可以是低保真的（草图或方块图），以鼓励受众思考组件和功能，而不是思考外观和感觉；也可以是高保真的（屏幕截图），向受众展示解决方案开发完成后屏幕的实际样子。

作者简介

乔伊·贝迪（Joy Beatty）是 Seilevel 公司副总裁。乔伊推动了新方法论和最佳实践的创建和实现，改善了需求征询和建模。她协助财富 500 强企业建立卓越业务分析中心。Joy 已经为数千名商业分析师提供了培训。

乔伊积极参与需求社区的领导工作，在多个行业组织的董事会任职。她目前是国际商业分析协会（International Institute of Business Analysis，IIBA）的核心团队成员，负责更新《A Guide to the Business Analysis Body of Knowledge》（商业分析知识体系指南），简称《BABOK Guide》。她曾在许多与需求相关的会议和演讲活动中发言。此外，她还在期刊、白皮书和博客文章中撰写关于需求方法论的文章。Joy 毕业于普渡大学，拥有计算机科学和数学的理学学士学位。

安东尼·陈（Anthony Chen）是 Seilevel 公司总裁。在过去 15 年中，安东尼与众多财富 500 强公司有过合作。他负责 Seilevel 的战略发展以及创新软件需求技术的开发，包括目标链；业务目标模型；目标、人员、系统和数据分类（OPSD）；以及 RML。

除了领导业务和创新，安东尼还撰写了大量关于软件需求技术、体验和理念的文章。其中一些文章可以在 Seilevel 博客中找到。他在伊利诺伊大学获得了电子工程和微生物学的理学士学位，并在德州农工大学获得了医学微生物学和免疫学的硕士学位。

要想联系我们，您可以在 Seilevel 博客上留言（https://argondigital.com/），给我们发电子邮件（RML@seilevel.com），或者在 Twitter 上加入我们的对话（@seilevel）。