第 1 章

需求建模语言入门

离节假日旺季还有九个月，一家著名的网上零售商确定要在其网站上添加一组重要的新功能，这将大大增强消费体验，直接增加销售额，同时减少好几个国家的客户电话服务量。据估计，这些新功能会为公司每年增加1400万美元的收入,而实现这些功能的费用却不到200万美元。产品经理确定这些功能的要求和业务规则，开发团队和项目管理团队预估可以在节假日旺季节到来时轻松地完成项目。为了能按期交付产品，开发团队努力赶工，晚上和周末经常加班加点。

八个月之后，该团队进入最后的测试阶段，感觉良好。他们完成了一长串功能增强以求获得高额的回报。然而，一位测试人员发现税收的计算是不正确的。不幸的是，这些计算错误只是冰山一角，因为团队忽略了和税收团队交流。实际上，他们当时没有意识到必须这样做。如果与税收团队交流，他们会发现在有些国家营业时要遵守的税务规则极为复杂，必须与管理税收规则的第三方软件进行集成。项目被推迟，那个旺季 1400 万美元的回报也成泡影。项目经理被解雇,产品经理被调往其他小项目。

项目经常因为需求的缺失、不完整或者不明确而受到困扰(The Standish Group 2009)。错误的需求实践普遍存在，所以大部分项目注定会失败(Ellis, Keith. 2008)。需求没做好是许多项目失败的根源，令人失望的是在过去 20 年中软件需求水平并没有显著提高。尽管学术界一直在稳步改善需求技术和工程方法，但是业务实践行为在很大程度上并没有什么变化。软件编程技术已经相当成熟，创造出各种新的技术，开发出丰富的工具，但是在写需求时，人们还是常常使用一长串的“应该”语句，把语句存在电子表格中。使用敏捷方法的项目也没有多少改进，还是经常把产品工作清单和用户故事作为一长串列表存在电子表格或其他工具中。

定义 RML

RML(需求建模语言)是为建立需求视觉模型而专门设计的语言，它便于企业管理、业务和技术等项目干系人使用。RML 不是一种学术上的建模语言。

在开发 RML 期间，我们改进了现有模型的易用性，创建新的模型来弥补功能上的缺失。结果就有了一套完整的模型，是专门为软件需求建模而设计的，对于那些常常搞不懂复杂模型的项目干系人来说，更容易接受。我们已经在许多大型软件开发项目上成功使用了这些需求模型。

传统软件需求实践的挑战

传统的做法不得不使用几千行“系统应该”这样的需求描述句，其繁复程度如图 1-1 所示。这些需求通常是通过与企业项目干系人面谈或者举行工作会议之后产生的。因为一般人都受米勒魔数的制约(参见下一节“人脑的限制”)，下面的事情几乎是不可能发生的：人们在阅读了数以千计的需求条款后，突然确信项目的需求是全面的。此外，另一个较大的问题是需求规模会逐渐变化。等你有了上千个需求，如果没有某种方法把这些需求与价值联系起来，力求在解决方案层面上进行比较，将很难决定应该砍掉哪些需求。团队经常把需求进行逻辑分组，但这些分组通常还是过大，无法得到有效的处理。

敏捷方法，如 Scrum，有产品工作清单、用户故事和验收标准。许多 Scrum 布道者说，产品工作清单应该是非嵌套的故事列表，这种做法比传统的需求列表好不了多少。验收标准也要求列出来，有时就列在便条卡的某一面上。做过大型系统的人都知道，在可能会有几百个项目干系人参加的项目上，这种缺乏信息组织结构的做法是行不通的。

人脑的限制

使用传统实践来创建软件需求的业务分析师在分析、组织和使用需求上会遇到同样的问题。传统的做法使用冗长列表来列出需求的文字描述，其形式是“应该”语句、用例、最近又增加的用户故事和产品工作清单。受限于人类的基本认知能力，冗长的清单列表使用起来都很困难。在 20 世纪 50 年代，认知心理学家乔治 • 米勒发现，人类只能记住和处理 7 加或减 2 项内容(Miller, George A. 1956)，这通常称为“米勒魔数”。

7 +/-2

后来的证据表明基数甚至可能少到 3 或 4(Cowan, Nelson. 2001)。这个数字代表大脑“暂存器”解决问题时所能保存的信息容量。无论实际数目是多少，如果要求普通人同时考虑大约 15 件事情，实际上最多只能记住和处理其中 9 件(可能更少)。如果要求处理的事情更多，一次只有几件可以同时处理，其他的会被快速切入或切出暂存器。想想去杂货店购买 15 件东西，如果没有一份写好的购物清单，你很可能漏掉东西或者买回的东西数量不正确。同样的道理，

如果需求列表或产品工作清单中的事项成百上千，那么你的大脑根本没有办法处理这种复杂性，除非把它分解成更小的结构化分组。

需求文件

REQ001 系统应该有姓、中间名首字母和名等字段。
REQ002 系统应该显示名字如果存储的个人资料中已有一个。
REQ003 系统应该要求姓名是完整的。
REQ004 系统应该有职位或头衔字段。
REQ005 系统应该要求头衔是完整的。
REQ006 系统应该显示职位或头衔如果存储的个人资料中已有一个。
REQ007 系统应该有电子邮件地址字段。
REQ008 系统应该有备用的电子邮件地址字段。
REQ009 系统应该显示电子邮件地址如果存储的个人资料中已有一个。
REQ010 系统应该显示备用电子邮件地址如果存储的个人资料中已有一个。
REQ011 系统应该要求电子邮件地址是完整的。
REQ012 系统应该要求备用的电子邮件地址是完整的。
REQ013 系统应该具有白天电话号码的字段。
REQ014 系统应该显示电话号码如果存储的个人资料已有一个。
REQ015 系统应该要求电话号码是完整的。
REQ016 系统应该在验证电话号码字段中所有字符是数字当用户退出该字段时。
REQ017 系统应该显示错误消息如果在电话号码字段不是所有字符都是数字。
REQ018 系统应该有传真号码的字段。
REQ019 系统应该要求传真号码是完整的。
REQ020 系统应该显示传真号码如果存储的个人资料已有一个。
REQ021 系统应该验证在传真号码字段中的所有字符是数字当用户退出该字段时。
REQ022 系统应该显示错误信息如果在传真号码字段里不是所有字符都是数字。
REQ023 系统应该有街道地址的两个字段。
REQ024 系统应该要求街道地址字段是完整的。
REQ025 系统应该显示地址如果存储的个人资料已有一个。
REQ026 系统应该有城市的字段。
REQ027 系统应该要求城市字段是完整的。
REQ028 系统应该显示城市如果存储的个人资料已有一个。
REQ029 系统应该有状态的字段。
REQ030 系统应该显示状态如果存储的个人资料已有一个。
REQ031 系统应该要求状态字段是完整的。
REQ032 系统应该有邮政编码的字段。
REQ033 系统应该显示邮政编码如果存储的个人资料已有一个。
REQ034 系统应该要求邮政编码字段是完整的。

图 1-1　冗长的需求列表

图比文字更容易理解

如何解决原始哺乳动物大脑的基本限制呢？有句话说得好：“一图胜千

言。”模型是信息的视觉表现方式(图形)，它描述了流程、数据和解决方案内部和环境的互动。你可能每天都在使用视觉模型但可能没有意识到这一点。

最近我出差，参加一个在赌场酒店举办的会议，我在前台登记后领到了房间的钥匙，前台女服务员给我指路，告诉我怎么去我的房间。她说了类似这样的话：“你从这里沿着右边走出去，然后沿着路向左行，穿过一个酒吧，路过几台老虎机，在喷泉附近向右转，走过一家餐厅，再走过一家餐厅，然后你会走到一个大厅，在那里向左转走过一条商业街，在路的尽头你会发现游泳池入口处有一个电梯。”

我茫然地盯着她。那一刻我能想起的就只有我刚刚从出租车走到前台时所看到的很多老虎机和赌桌。我假设在去房间的路上会走过更多的老虎机和赌台，女服务员刚才讲的已经记不清楚，反而把我搞得更糊涂。后来她给了我一点儿希望：“这张图画了怎么去那里。”她画出从前台到电梯的路线图，如图 1-2 所示。我松了一口气，因为我只记住了她所说的前几步，其他记不住，但现在我有了一个模型，我糊涂的时候可以参考。一张图！总而言之，当人类解释信息时，图比文字更容易理解。

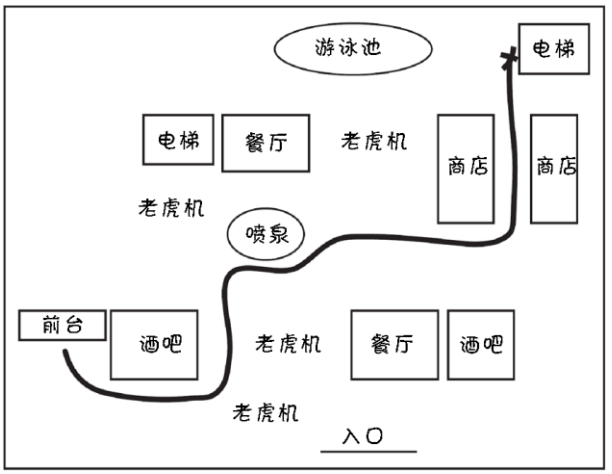

图 1-2　一张穿过赌场的地图，对应于女服务员所说的路线

需求模型

需求模型组织和展示了大量信息，帮助你发现缺失的信息，并给出上下文细节(Gottesdiener, Ellen. 2002)。最重要的是模型可以从视觉上进行分组，使你能够通过短期记忆能力快速分析大量截然不同的信息。在有几千条“系统应该”句式的需求文档中，阅读、解释并找出差异是很困难的，而视觉模型能够提供帮助。

想象在你面前零乱摆放的字母(如图 1-3 所示)，要你找出字母表中哪些字母没有出现。

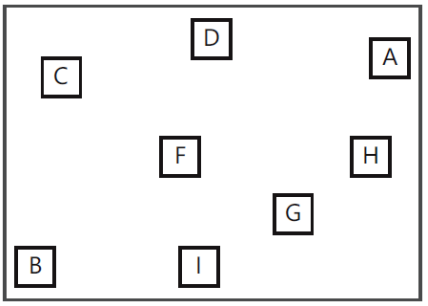

图 1-3　零乱摆放的字母中，缺了哪个字母

如果你只是盯着混乱的字母或甚至把字母无序地排成一行，是很难发现缺了哪个字母的(事实上，你可能刚刚试着把它们按顺序排列起来)。如果按字母顺序排列(如图 1-4 所示)，瞬间就会发现缺失的字母。

要找到缺失的需求，关键是利用一个事实，每一个需求与其他需求都有着某种联系。当你得到一长串“系统应该”的需求条款时，要想保证其完整性是极其困难的，但如果重新组织需求就可以利用这种联系，每次只在较小的分组里分析信息从而大大简化任务。

需求模型用于项目的整个生命周期。这些模型可用于多种场合，有助于分析需求，有助于向项目干系人提出需求和验证需求，有助于与开发人员和测试人员沟通需求。

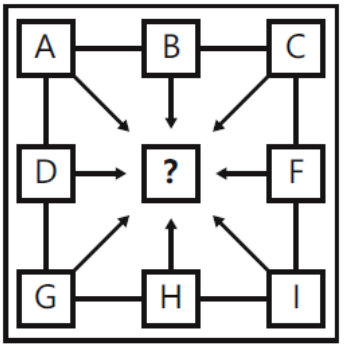

图 1-4　排列已有字母，找出缺失的字母

为什么不用 UML

一个直接的问题出现了：“为什么不使用统一建模语言(UML)？”|UML 是一种专门用于以可视化方式设计软件系统的语言(请参阅文献 Object Management Group. 2007)。UML 为需求建模奠定了合理基础，但它不满足需求建模的全部要求，因为它缺少有关需求与业务价值的模型，缺少从最终用户的角度展示系统结构的模型。此外，它在技术上过于复杂使得业务项目干系人

难以掌握，因为它的模型侧重于软件系统的架构建模。最后，UML 用于描述系统的技术设计和结构，顶多在建模方面对 UML 进行翻新改造以支持业务收益、用户操作和业务规则。

当一个模型只聚集于解决问题的一个或两个方面时，它是最有用的。如果一个模型具有许多类型的信息或者模型的语法规则过于复杂和难于理解，项目干系人绝对不会用。事实上，我们的经验说明，模型的复杂性是造成大型企业不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出来的，还可以传达必要的信息。RML 的目的是提供一致的语法和语义结构供业务干系人分析和理解项目模型。设计该语言的目的是让整个团队容易学习和使用，包括但又不局限于业务项目干系人、开发人员和测试人员。模型简化到只有最基本的符号和格式，但还能保证在需求方面取得预期的结果。RML 不只针对软件开发方法，也可以容易适应于与任何开发方法或工具套件结合使用。

需求与设计

许多 RML 模型在业务分析师看来通常属于设计领域。例如，显示-操作-响应模型使用线框或屏幕截图来描述用户如何与屏幕元素交互，用户界面流模型展示用户如何浏览各种用户界面。

有一句关于需求的谚语：“需求关注的是要建什么，设计关注的则是它如何起作用。”需求和设计之间的区别很重要，因为很多人强调任何设计都不应该和需求混在一起，设计文档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：“一个层面的需求是对另一个层面的设计。”

一个层面的需求是对另一个层面的设计

在自上而下的解决方案中有不同的概念层面，如果考虑一个层面是有关“什么”的，那么它的下一层面将是有关“如何作用”于它的。因此，基于这种“什么”与“如何作用”的定义，如果一个层面是需求，那么下面的层面就应该是对需求的设计。

例如，项目干系人可能需要降低网站的购物车放弃率。在下面的细节层面，产品经理可能会提出几个不同的解决方案力求降低购物车放弃率。例如，该团队可以减少结账过程中的步骤，可以提供保留购物车内容的功能方便顾客下次购买，或者可以提供免费送货服务。提出的每一个解决方案都是有关“如何作用”(即“设计”)的，满足的是“什么”需求(即降低购物车放弃率)。此外，最初的“什么”(即改进系统降低购物车放弃率)可能同样也是“如何作用”(即试图改进整个网站转化率)。

不要用“什么”与“如何作用”的关系来区别“需求”和“设计”。这种方式不好。

确定业务的实际需要

另一个常见的定义是，任何有关实际解决方案的都属于设计而不是需求，例如算法的使用、外观和感觉、用户界面元素等。但是，在有些情况下，特定的请求有时可能是需求，而有时可能是设计。例如，在某些行业中，一个产品为了竞争的需要必须使用专门的加密算法，因此它是一种需求。对于另一个应用程序，它完全不关心使用专门的加密算法，唯一重要的是应用程序必须使用某种算法来加密信用卡号码。

用户要求是不是需求，关键要看业务项目干系人是否真正需要它。我们都知道，项目干系人实际上并不需要他们宣称自己想要所有的特征，所以要对请求作出判断，它是否真正是需求，用户是否真的需要。

定义需求

需求是企业需要在解决方案中实现的。因此需求可以包括功能性需求、非功能性需求、业务规则，甚至包括许多人传统上所称的设计。可以使用模型方法帮助项目干系人真正明白需要什么，而不是告诉他们允许他们选择什么类型的需求。

本节定义一些用于全书的需求术语。功能性需求是一个解决方案所提供的行为或功能而不加任何限定词。业务规则表示在修改功能性需求时必须满足的条件语句，包括但不限于什么时候该功能可以用以及允许谁执行该功能。业务规则包含诸如“如果”“何时”和“然后”等词汇。非功能性需求是任何不属于功能性的需求(包括业务规则)。特性是一个功能区域的简短描述，解决方案将最终实现该特性以达到业务目标。特性是需求的集合，用来清楚描述和组织需求。表 1-1 给出了几个例子。

表 1-1　需求的例子

需求	类别
系统应该能够自动批准或驳回信用申请	功能性需求
当信用指标高于 750，系统应该自动批准信用申请	业务规则
对于信用指标低于 750，当自动决定是否批准信用申请时，系统应该使用下列算法：[算法将加在这里]	业务规则
审批过程应该在 30 秒内返回给用户	非功能性需求

假设是做决策时所依据的真实陈述。假设包括对未来的任何预测或预报。假设对于需求非常关键，因为这些假设会不断被引用，但很少有人理解或能够有人讲清楚。事实上，如果让业务分析师写下自己的假设，他们通常写下一些琐碎的小事，既不具有影响力又缺乏重要性。如果这些例子中的假设被证明是不正确的，可能会导致业务目标无法实现。

- 很多人都愿意在网上搜索以解决他们的技术问题。
- 当遇到技术问题时，50%的人愿意等待以后再试。
- 90%的业务客户都在上网进行。
- 待解决的问题中有些问题可以由客户自行解决。

需求模型不等于游戏的结束

需求模型的使用并没有完全取消写需求。虽然模型提供上下文又创建了有关需求的完整图形，但是模型还不是系统开发人员和测试人员最终使用的形式，必须采取额外的步骤从模型中提炼出需求。就像按照货架走道来组织购物清单一样，要为开发项目的团队产生需求清单。模型的价值在于以某种方式帮助组织所有的需求，以便很容易看出需求有缺失、无关或不正确的情况。

创建的所有模型都应该纳入项目的全部需求中。文字需求和可视化需求共同描绘出待建的解决方案的全景(Wiegers, Karl E. 2003)。

在项目中使用 RML

可以把这本书中介绍的 RML 模型想象成软件项目的模型模板工具箱。通常情况下，建议综合使用多个模型，有些常用的方法定义了在整个开发周期中何时使用特定的模型。把需求模型应用到项目时，可以和许多开发方法一起使用，例如敏捷方法、迭代方法和瀑布方法(参见第 25 章)。

其他资源

- “业务分析师的 RML 快速参考指南”，两页篇幅的模型相关摘要：*http://www.seilevel.com/wp-content/uploads/RML-Language-for-Modeling-Software-Requirements.pdf*
- 《软件需求》(第2版)第11章系统介绍了模型的价值(Wiegers, Karl E. 2003)

参考文献

- Chen, Anthony. 2010. “What vs. How – BRD vs. User Requirements vs. Functional Requirements”: *http://requirements.seilevel.com/blog/2010/04/what-vs-how-brd-vs-user-requirements-vs-functional-requirements.html.*
- 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. *http://www.iag.biz/images/resources/iag%20business%20analysis%20benchmark%20-%20executive%20summary.pdf.*
- 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. 2003. *Software Requirements, Second Edition*. Redmond, WA: Microsoft Press.