第3章.　需求工程过程

3.1. 概述

过程是一组相关活动的集成，通过这些活动的执行，可以完成一项任务或者达到一个目标。需求工程过程是系统开发当中需求开发活动的集成，它以用户所面临的业务问题为出发点，进行分析和各种转换，最终产生一个能够在用户环境下解决用户业务问题的系统方案，并将其文档化为明确的规格说明。

为了有效的理解用户问题，分析各种可能的系统解决方案，并最终产生一个适宜的规格说明文档，需要将需求开发活动组织成一个系统化和严格的需求工程过程，这是人们随着系统开发的进展而逐渐认识到的[Siddiqi1996]。在最初的时期，系统开发的唯一焦点就是编码，此时不论系统大小，开发都同样是一个单独的活动——编码，这个时期人们还不认为存在独立的需求开发活动。其后，随着生命周期模型的引入，对系统开发活动的认知取得重大进展，人们认识到需求开发是系统开发当中一个独立的阶段，即软件开发生命周期模型的第一个阶段。在此后的进一步发展当中，人们逐渐认识和接受了系统的演化式开发思想，认识到系统的实现往往是开始于一个并非完备的需求体系，发现需求开发也是一个递进的过程，包含一系列的独立活动。今天，大多数的软件专业人士已经意识到需求工程也有属于它自己的生命周期模型，即存在针对需开发的需求工程过程，这个过程又作为系统工程和软件工程的一个子过程，部署在系统开发的初期阶段。

目前看来，如果所开发系统的类型不同，或者开发公司的规模和文化不同，或者系统开发资源获取的途径不同，需求工程过程都会表现出极大的差异[Martin2002]。例如，对一个大规模的军用或航空系统而言，通常存在一个正式、严格的需求工程过程，它详细定义了过程的各个阶段，要求产生能够详细描述系统需求和软件需求的规格说明文档。而对开发创新型软件的小公司而言，需求工程过程可能仅仅包含一些头脑风暴会议，它产生的文档也可能仅仅是对系统期望的一段简短描述。但不管实际上执行的需求工程过程为何，它们都拥有一些共同的需求工程活动：需求获取、需求分析、需求规格说明、需求验证和需求管理。其中，需求获取、需求分析、需求规格说明、需求验证为需求开发活动，需求管理为项目管理活动。

在需求工程的开始，必须要获取用户期望系统表现出来的各种行为，这些期望并不是外在和可以直接得到的，系统分析师需要利用一些方法和技术才能得到正确的结果。

为了取得对需求的正确和深入理解，系统分析师还需要对获取的结果进行综合与整理，通过分析保证需求的正确性、完整性和可行性。

经过分析的需求被认为是有效的需求，它必须被进行记录和文档化，因为只有将需求文档化为正式的规格说明，才可能将它们传递给其他参与系统开发的人员，让所有相关人员形成对系统需求的一致和正确的理解。

规格说明文档可能被传递给设计人员、测试人员、项目管理人员等众多系统开发者，因此如果传递的规格说明文档中存在一些错误或偏差，将造成很大的影响。为了尽可能减小不利因素，尽最大可能产生完善的规格说明文档，就需要在规格说明文档产生之后和传递给相关人员之前，组织进行文档的验证，以尽可能的发现文档中的错误和偏差并进行纠正。

获取、分析、规格说明和验证这些需求开发活动并不是看上去的那样以线性、顺序地方式执行，实际上，这些活动之间是互相交织的，整个开发活动也是不断迭代和递增的（如图3 - 1）。

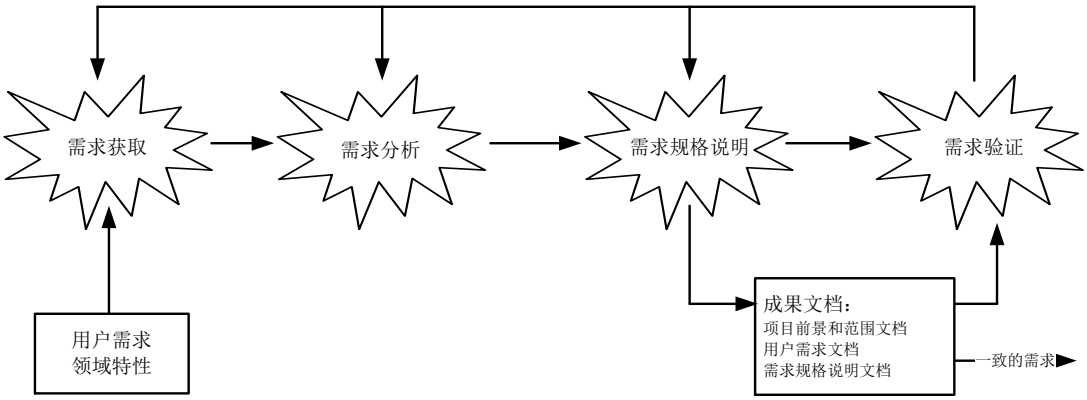


图3-1 递增的需求开发模型，修改自[Kotonya1998]

在需求开发活动当中，会产生各种各样的成果文档，比较常见有三种：项目前景和范围文档、用户需求文档和需求规格说明文档。项目前景和范围文档定义了系统的业务需求，明确了系统开发的努力方向和工作范围。用户需求文档定义了系统的用户需求，以用户的立场表达了对系统行为的期望，常见的用例文档就是用户需求文档的一种形式。需求规格说明文档定义了系统的系统级需求，指出了开发者应该完成的任务。需求规格说明文档又依文档内所定义的需求范围分为系统规格说明和软件规格说明，其中系统规格说明内定义的是对整个系统的需求，包括软件需求、硬件需求和其他需求，软件规格说明内定义的仅仅是软件需求。

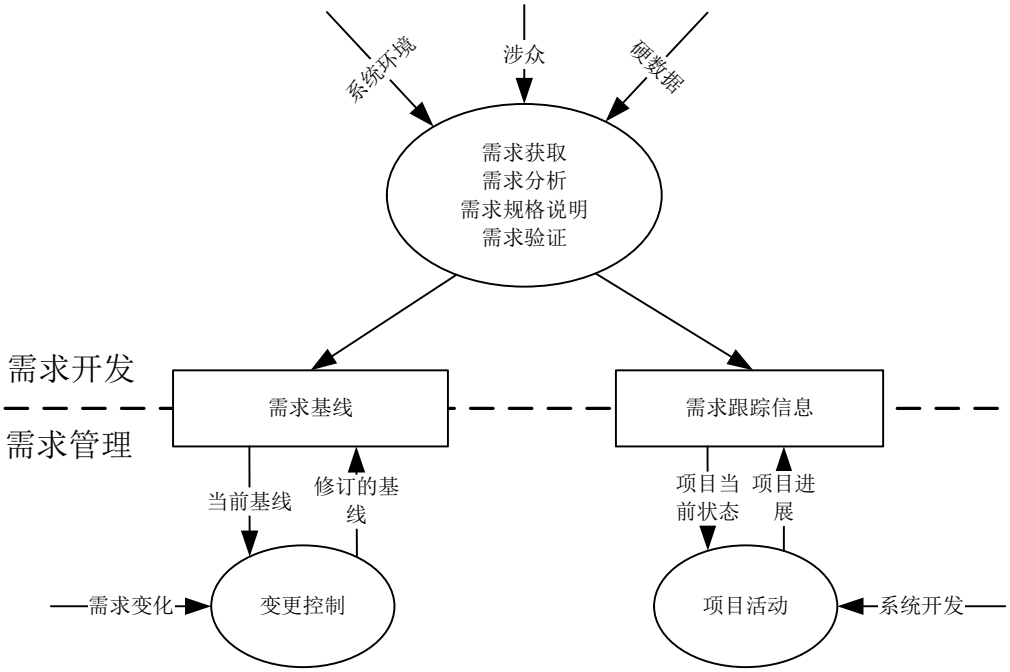


图3-2 需求管理与需求开发活动的界限

在所有的需求开发活动结束之后，定义良好的需求被转入系统开发的后继阶段——设计、实现、测试等，但这时系统开发人员却仍然常常要面对一个非常头疼的需求问题——需求变化。因为遭遇业务问题的现实世界的确处于不断变化之中，所以为了得到有效的系统解决方案，有些变化必须被进行妥善的处理，这

就需要在需求开发结束之后，在后继阶段当中采取有效的策略统一管理开发的需求和变化的需求，进行需求的管理和变化的控制。所以，和前面的四个需求开发活动不同，需求管理是项目的管理活动，它在需求开发活动结束之后才开始执行。需求管理和需求开发活动之间的界限如图3-2所示。

3.2. 需求工程活动

3.2.1. 需求获取

需求获取是从人、文档或者环境当中获取需求的过程。获取过程并非是将定义良好的需求从人、文档或者环境当中直接转移到获取的结果文档上那样简单，需求工程师必须要利用各种方法和技术来“发现”需求。

需求开发的过程包含有学习和认知的过程，而学习和认知的过程是递进的，即学习一点，增加一些认知，然后在新的认知的基础上继续学习。因此需求获取和需求分析是交织在一起的，需求工程师需要获取一些信息，随即进行分析和整理，理解、认知到一定程度后再确定要进一步获取的内容。

在需求获取当中，需求工程师通常需要执行的任务包括：

一、收集背景资料

获取的目的是发现用户的问题，并经过需求分析步骤转化为用户的需求。要想和用户就业务问题进行交流，需求工程师就应该先具备一个能够和用户进行交流的知识基础，否则需求工程师和用户根本无法形成有效的沟通。因此需求工程师需要先收集系统的背景资料以形成一个基础的知识框架，例如企业的业务状况等。此时如果需要对背景资料进行非常深入的了解（例如进行产品线开发），就需要应用相关的需求分析方法（例如领域分析等）对收集的资料进行整合与处理，当然这是需求分析的任务。

二、获取问题与目标，定义项目前景与范围

在有了一定的知识框架之后，需求工程师就可以通过收集数据和文档，观察环境，了解用户的需要、期望和关注点，综合推定用户在业务中所遇到的高层次问题。用户解决高层次问题的期望即为系统的业务需求，也是系统要达到的目标。

在进行信息获取时，不同的用户往往会从各自自己的立场出发考虑问题，提出相应的功能要求。这样，在软件系统涉及很多的用户时，就常常会发现用户相互之间对系统的期望有着很大的差距。因此，需要根据业务需求明确高层次的解决方案，确定软件产品将来的样子，定义项目的前景。有了共同的项目前景，不同的用户就可以从共同的方向上理解问题，提出对系统的功能要求。而且在用户之间发生需求冲突时，还可以利用项目前景指导需求冲突的协商解决。

需求获取当中面对的信息内容非常广泛，因此，要保证获取的有效性，就一方面要求不能在无关的内容上花费太大的代价，另一方面也要不遗漏应该获取的重要内容。因此在开展详细信息的获取工作之前，应该先根据业务需求确定项目的获取范围，然后在范围界限的指导下保证获取活动的正确和顺利进行。而且在项目开发的后期发生需求的变更时，还可以依据清晰的项目范围定义来排除不必要的变化请求。

项目的业务需求、前景和范围都会被记录到项目的前景和范围文档之中。

系统开发是一个不可见的过程，在最终产品完成之前，要客户完全根据对开发人员的信心来启动一项投

资无疑带有一定的风险，因此在项目早期产生的前景和范围文档还可以起到提高客户投资信心的附带效应。

根据业务需求确定项目前景和范围的过程也包含有需求分析的活动。

三、识别涉众，选择信息的来源

在大多数的系统开发当中，用户是需求的主要来源。一个复杂的系统往往拥有很多用户，因此在执行获取时要想覆盖所有的用户不仅费时费力，而且是困难的，也是不必要的。一个可行的方案是使用少数的用户代表来代表全体用户表达看法。但是系统的不同用户往往在很多方面存在差异，例如使用的产品功能，具有的计算机技能，使用产品的频率，等等。这些具有不同特性的用户对系统的需求往往也是不一样的，因此要想让选取的少数代表完整的表达用户的声音，就需要将用户分成不同的类型，然后在理解每种类型用户特征的情况下为其选择合适的用户代表。这个过程被称为涉众分析。

表单、报表、备忘录等硬数据是需求获取信息的另一个重要来源。在用户的工作当中，往往会产生大量的硬数据，它们以清晰、条理和准确的方式描述了实际业务的相关信息，因此是一种理想的信息来源。但同样的问题是，面对大量的硬数据，也需要使用恰当的方法进行采样，以保证采集的少量数据能够准确、完整的代表全部数据的相关信息，即进行硬数据采样。

除了用户和硬数据之外，相关的产品、文档和领域专家等也都有可能是需求的来源。

四、选择获取方法，执行获取，获取功能与非功能需求

需求获取的主要目的在于获取用户需求，了解用户在完成任务时遇到的问题与期望。获取有效用户需求的前提是能够正确的理解用户的问题，所以针对每个获取的需求都要同时获取相关的问题域特性，在问题域特性充足的情况下，需求才能够正确的体现用户的意图。问题域特性往往包含很多细节，而且要清晰的描述这些细节是非常困难的。所以简单的向用户提问“你需要系统为你做什么？”是无法达到预期目的的，需求工程师需要运用很多的获取方法和技巧才能够完成任务。

需求获取的方法和技巧有很多，常用的方法有面谈、调查表、观察、原型等等。它们都有自己的优缺点和适用情况，没有哪种方法可以有效的应用于各种场合，因此需求工程师需要根据信息的来源、信息的类型、获取的成本和时间等各种因素选择合适的获取方法，执行获取活动。例如在用户地理上广泛分布的情况下就可以使用调查表方法，在问题模糊和复杂的情况下可以使用原型方法，等等。

五、记录获取结果

需求获取阶段产生的主要成果有：业务需求、项目的前景和范围、用户需求以及问题域特性。它们都需要被及时的记录下来。

项目的前景和范围文档记录了业务需求和前景与范围信息。获取笔录记录了用户需求和问题域特性。

因为刚刚获取的信息还没有进行分析与处理，所以获取笔录记录的内容往往具有凌乱、模糊、冗余、遗漏等诸多问题。

3.2.2. 需求分析

需求分析的主要工作是通过建模来整合各种信息，以使得人们更好的理解问题。同时需求分析工作还会为问题定义出一个需求集合，这个集合能够为问题界定一个有效的解决方案。需求分析还需要检查需求当中存在的错误、遗漏、不一致等各种缺陷，并加以修正。

需求分析活动最后会产生一个需求的基线集，它指定了系统（或当前版本的系统）开发需要完成的任务。在资源受限的情况下，这个基线集往往只是用户所要求功能的一个子集，而且需求工程师和用户必须就该子集的取舍达成一致。需求基线集之中的需求要具有优秀需求的特性（见第二章），尤其是一些不一致和冲突的现象必须得到妥善的解决。

在需求分析阶段，需求工程师主要的任务包括：

一、背景分析

系统是作为用户业务问题的解决方案得以被开发的，但仅仅系统本身是无法帮助用户达成目标的，它必须和它所部署的环境形成互动才能解决用户的问题。所以，在进行系统开发，尤其是需求开发时，研究系统所将要部署的环境无疑具有重要意义。而且通过对环境的分析和理解，还可以帮助需求工程师形成一个关于用户业务的知识框架，这又进一步有利于需求工程师在细节的需求获取活动当中形成和用户的有效交流。背景分析就是研究系统环境的一个任务。

在多数情况下，系统的环境较为清晰和简单，因此进行背景分析并不需要太大的投入和太复杂的手段。但在规模较大系统的开发当中，系统环境往往难以梳理，这时就需要使用一些专门的分析方法，例如领域分析、企业建模等。

二、问题分析、目标分析、业务分析，确定系统边界

在获取系统的问题、目标、前景、范围之后，要使用问题分析、目标分析、业务分析等分析方法与技术对它们进行处理，并基于这些处理明确其解决方案，定义系统的边界。系统边界之内定义的是系统需要对外提供的功能，系统边界之外标识的是对系统有功能要求的外部实体或者对系统有所限制的环境要素。

系统边界的定义要保证系统能够和周围环境形成有效的互动，并且在互动中解决用户的问题，满足业务需求，这些都依赖于分析技术与方法的有效使用。

系统用例图和上下文图通常被用来定义系统的边界。

三、软件需求建模

建模是为展现和解释信息而进行的抽象描述活动。模型由一些基本元素和元素之间的关系组成，它含有丰富的语义。和文本化的自然语言相比，模型能够在有限的空间内表述更加严谨、准确和高密度的信息。

软件需求建模是需求工程当中最为重要和基础的一项任务。它将大量信息以清晰、条理的方式集成到一个模型当中，让需求工程师对问题形成更为深刻的理解。需求工程师还可以依据模型进行推理，以创建能够界定可行解决方案的需求集合。为系统建立的模型还可以更好地将信息传递给开发人员。

在为需求建模时，常用的技术包括数据流图、实体关系图、状态转换图、类图等半形式化建模技术。在有些要求严格的领域（例如，安全攸关的医疗器械控制），也会应用Z模型等更加严格的形式化技术。这些不同的建模技术各自为不同的应用目的而设计，适用于不同的建模要求，所以需求工程师在进行需求建模前需要进行合理的判断与选择。

目标分析、业务分析也是需求建模过程，只是从工作阶段划分来讲，这里更多的是指需求后期阶段的需求分析与建模。

四、细化需求

用户需求往往具有模糊、歧义等诸多不利的特征，这使得它们很难被加以评估和验证。所以很有必要在系统模型的帮助下发现更多的细节，并依此将用户需求转化为一些具有良好粒度和特性的细节需求，即系统级需求。

五、确定优先级

用户对系统往往有许多的需求，而且这些需求并不是处于同等重要的地位，因此需求工程师需要根据其重要程度为需求设定优先级。这些优先级对多版本系统的功能分布和后继开发活动中的功能取舍非常重要，尤其是在开发资源非常有限甚至不足的情况下。

在项目的整个开发过程当中，应定期评估和调整优先级，以适应客户需求、市场条件和业务目标的变化。

六、需求协商

在分析当中，有时会发生不同用户间的需求冲突，这种情况下用户各自的需求都是合理的，但却不可能在系统当中同时被加以实现。无法同时实现的原因有可能是用户们的需求互相敌对，不可调和，也可能是因为系统实现的资源有限，无法二者兼顾。

发生需求冲突时，需要让各方用户清楚的意识到冲突的存在，并且通过他们之间的协商来加以解决。在实践当中，在用户能够从他人的关注点出发的情况下，冲突往往是比较容易解决的。

需求工程师在这个过程中的工作是通过分析及时的发现冲突，并为处理冲突提供技术上的参考信息，组织和指导用户之间的协商。

3.2.3. 需求规格说明

获取的需求需要被编写成文档，业务需求被写入项目前景和范围文档，用户需求被写入用户需求文档（或者用例文档）、系统级需求被写入需求规格说明。

编写文档的主要目的是为了在系统涉众之间交流需求信息，因此编写的文档应该具有一定的质量。这些质量特性有些来自于文档内所有独立需求的质量之和，有些来自于编写者的写作技巧，最重要的质量要求是简洁、精确、一致和易于理解。

需求工程师在这个阶段的主要工作包括：

一、定制文档模版

开发团队通常都会在其内部为各种需要编写的文档维护一些文档模版，需求规格说明文档也不例外。模版为记录功能说明和其他与需求相关的信息提供了统一的结构。

通常组织都会参考[IEEE1998]推荐的规格说明文档，然后根据自己的特点和需要进行调整，建立组织的参考模版。在进行具体的项目开发时，需求工程师再依据项目的特点对组织的参考模版进行进一步的定制。

二、编写文档

有了定制的文档模版，就可以开始编写需求文档。在编写的过程当中，一方面要选择最准确的表达方式，另一方面又要注意保证文档的良好结构和易读性。通常，人们会同时使用模型语言（图形、表达式等）和自然语言（文本）两种表达方式，用模型语言来保证信息传递的准确性，用模型后附加的文本描述保证文档的可读性。

3.2.4. 需求验证

为了尽可能的不给设计、实现、测试等后继开发活动带来不必要的影响，需求规格说明文档中定义的需求必须要能正确、准确的反映用户的意图。为此，需求规格说明文档至少要满足下面几个标准：（1）文档内每条需求都正确、准确的反映了用户的意图；（2）文档记录的需求集在整体上具有完整性和一致性；（3）文档的组织方式和需求的书写方式具有可读性和可修改性。

为了保证以上标准的满足，需求规格说明文档，尤其是最终定稿的需求规格说明文档，在传递给相关人员之前要进行严格的验证。

需求验证阶段的主要任务包括：

一、执行验证

执行验证的方法有很多，同级评审其中最通、最有效的一个。在有些情况下，也需要使用原型或模拟等代价相对较高的验证方法。

二、问题修正

在需求验证的过程当中会发现一些问题，这些问题在验证之后必须要及时得到修正。问题修正的过程还应该得到跟踪，以保证修正的落实。

3.2.5. 需求管理

在需求开发活动之后，设计、测试、实现等后续的软件系统开发活动都需要以围绕需求开展工作。需求的影响力贯穿于整个软件的产品生命周期，而不是单纯的需求开发阶段。所以，在需求开发结束之后，还需要有一种力量保证需求作用的持续、稳定和有效发挥，需求管理就是这样一个管理活动。

而且，在需求开发建立需求基线之后，还需要在设计、实现等后继活动当中处理来自客户、管理层、营销部门以及其他涉众群体的变更请求。需求管理会进行变更控制，纳入和实现合理的变更请求，拒绝不合理的变更请求，控制变更的成本和影响范围。

在企业界的实践当中，需求变更被认为是导致项目失败的二个主要原因之一[Robert2002]。所以需求管理在项目的各项管理活动当中具有非常重要的地位，CMMI就将其作为所有二级成熟度企业都应该具备的一个关键过程域。

目前有很多的需求管理工具可以帮助进行需求管理工作，它们可以为每项需求定义属性，跟踪需求的状态，并在需求和其他系统开发产品间建立跟踪体系。

需求管理阶段的主要任务包括：

一、建立和维护需求基线集

建立良好的配置管理，对需求基线进行版本控制，是进行有效需求管理的前提和基础。

要实现需求基线的版本控制，首先要标识每项需求，记录它的相关属性，例如ID、来源、产生日期、产生理由、优先级、预计实现成本等等，然后再为每一个需求文档建立唯一的版本号标识。

在建立初步的版本控制之后，所做的变更必须被明确地加以记录。记录的内容包括变更情况、 变更日期、变更原因等等。所有的变更还要被及时地传达给受到影响的每一个人。

基线的版本控制工作可以使用版本管理工具来进行，也可以使用专业的商业需求管理工具来进行。

二、建立需求跟踪信息

建立需求在项目当中的可跟踪性（Traceability）是需求管理的一个重要任务。

需求的可跟踪性要求以系统级需求为出发点进行双向跟踪：一是后向跟踪（Post-Traceability），跟踪系统级需求被设计、实现为哪些制品，并回溯每个设计、实现制品是为哪些需求存在的；二是前向跟踪（Pre-Traceability），回溯每个系统级需求是为支持哪些用户需求及业务需求存在的，并跟踪每个业务需求、用户需求是如何被转化为系统级需求的。

三、进行变更控制

考虑到现实世界的多变性，一个成功的项目在一致的需求基线建立之后仍然应该积极的接受来自外界的需求变化请求，并做出及时的调整与反馈。但为了保证项目的顺利进行，减小需求变化带来的失败风险，这些需求变化的请求必须得到妥善的控制，随意的需求变化是应该被绝对禁止的。

实现变更控制首先需要建立一个控制变更的过程及相关策略，它们确定了组织应对变化的基本方法。进行变更控制还需要挑选一批有经验的用户和开发人员，组成变更控制团队，由他们来分析变化的利益得失，审定变化的接受与否。

有效的变更控制要求所有在需求开发阶段之后发生的需求变化都要被提交给变更控制团队处理，都要严格按照变更控制的过程进行分析、判断和落实。

3.3. 需求开发过程是迭代和并发的

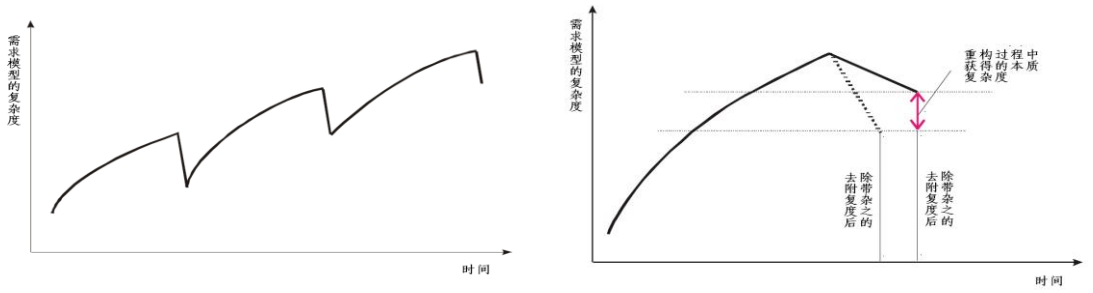
从需求工程各项活动的内容来看，需求开发活动似乎可以遵循着“需求获取→需求分析→需求规格说明→需求验证”的路线顺序执行，并成功的产生有效的成果文档。但正如软件开发过程的瀑布模型一样，实践者们很快就发现这个线性顺序的过程可以用于解释需求开发当中的活动内容，却无法用于组织成功的需求开发。

究其根本，作为软件开发的一个阶段，需求开发与软件开发一样存在着大量的不确定性，甚至于是软件开发中不确定性最多的一个阶段，所以需求开发与软件开发一样都应该是迭代的[Larman2003]。

需求开发不仅是迭代的，而且它的两个重要活动——需求获取与需求分析，还是交织的。需求获取与需求分析是需求开发中的两个主体活动，它们共同构成一个学习过程。

“不论是像早期一样认为需求是从涉众中获得的，还是最近一些观点认为有些需求是创造的、发明的或构建的，它们都包含了某种程度的学习过程——对涉众本身以及他们的任务、工作场所等的学习过程。学习是需求工程中很大的问题。”[Maiden2012]

[Nguyen2003]的发现更清晰地说明了需求开发的学习过程，如图3-3所示。[Nguyen2003]统计了随着时间发展的需求分析模型复杂度分布情况。理论上随着时间延伸，获取的内容逐渐积累，需求分析模型的复杂度应是线性上升的，可事实上却在总体上升趋势中伴随着间歇性下降。[Nguyen2003]认为下降的过程就是学习过程中知识积累到一定程度后发生的知识重构过程，简单来说就是理解更加深入所以复杂知识突然变得简单的“豁然开朗”现象。也就是说实际的需求开发过程应该是“获取、分析→重构→再获取、分析→再重构→……”这样一个获取与分析交织并迭代的过程。



a　　b

图3-3实际的需求开发过程中模型复杂度分布，源于[Nguyen2003]

如果综合考虑需求开发的迭代性，尤其是需求获取与需求分析的交织性则图3-4的需求开发过程描述更加准确。

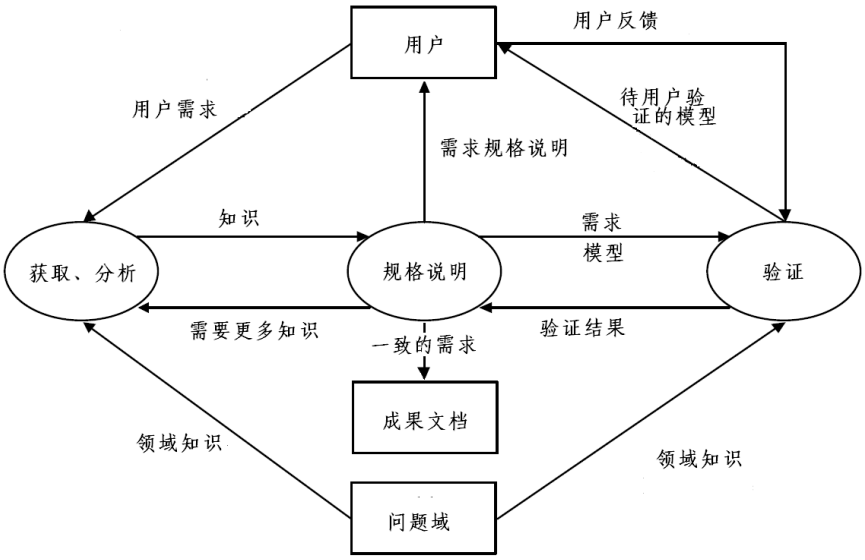


图3-4 迭代的需求开发过程模型，修改自[Loucopoulos1995]

需求开发过程不仅是迭代的，更进一步地说它的各个活动之间是并发的，如图3-5所示，其中分类（Triage）属于本书中的需求分析活动。需要说明的是，因为需求和分析是交织的，所以实践中很难准确区分二者的工作量，所以图3-5中获取与分析的分界也只是示意性的。

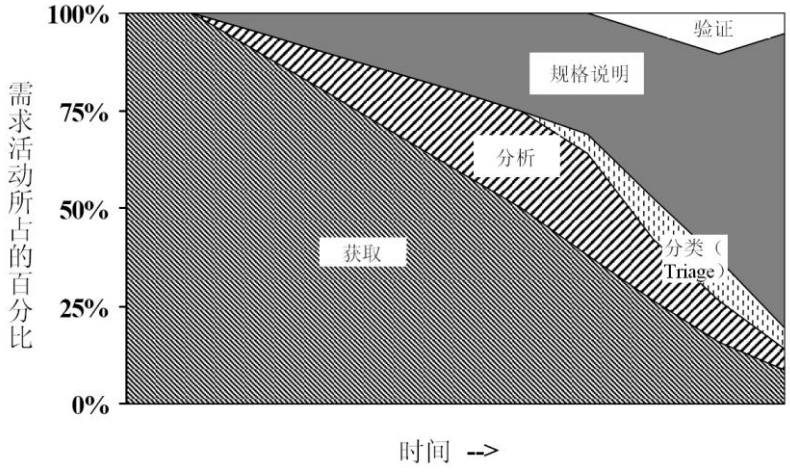


图3-5 需求开发活动的并发示意，源自[Hickey2003]

3.4. 实践方法的应用

通过分析需求工程过程，可以了解需求工程活动粗略的组织形式和高层视图，但是要进行一些具体的工作还需要更多的活动细节。这些活动细节是通过应用实践方法来实现的。

3.4.1. 细节知识的实践性

在任何一个知识领域当中，人们都需要在进行相当的探索之后才能为其建立学科化和系统化的知识体系。这个探索的过程通常很漫长，很多自然科学领域的知识体系都是人们在进行了数百上千年的探索之后才逐渐形成的。

在工程领域当中，如果能够建立比较完整的知识体系，那么就可以在知识体系的指导下进行规律性和系统化的生产。相反，在完全没有形成知识认知的全新工程领域当中，就只能纯粹依赖生产者的个人才智来进行工作。也有介于上述二种情况之间的工程领域：它们还没有形成完整的知识体系，所以无法实现大工业化的生产方式；同时这些工程领域又经过了相当时间的探索，从生产者大量的个人行为当中总结出了一些有效的工作方式和行为方法。这些有效的工作方式和行为方法虽然比较琐碎和孤立，和知识体系的要求还有很大的距离，但是它们却能够很好的帮助人们更快更好的进行工程实践，所以被称实践方法（Practice，又被称为原则Principle）。

实践方法是人们能够从陌生的知识领域当中得到的最早的知识片段和知识形式。在它们逐渐累积和丰富之后，人们就会从它们当中抽象出更加普遍的规律性知识，建立知识体系。

软件工程是一个“年轻”的工程领域，还没有形成一个完整的知识体系[Shaw1990]——有些部分完整，有些部分欠缺。在软件工程的各个子活动当中，有些领域已经形成了比较完整的知识体系，例如程序的编码和编译方法。也有一些领域还没有形成需要的知识体系，需要依赖大量的实践方法来指导工作的进行，例如设计活动和需求工程活动。

因此，除了能够在宏观上指导需求工程过程进行的过程模型之外，需求工程的成功应用还需要掌握很多能够指导工作细节的实践方法（参见附录2）。

3.4.2. 重要的实践方法

经过长期的实践，人们在需求工程领域发现和创造了很多用于处理这些需求工程细节的方法和经验，其中一些取得了显著的成功，被人们所广泛接受。因此，需求工程师的一项重要工作就是理解业界好的实践，并将它们成功的应用到组织的需求工程过程当中去。

在诸多的实践方法当中，有一些实践方法得到了广泛的应用。本书就从中选择了一些重要的实践方法（如表3-1所示），并在后面的章节当中进行具体的介绍。这些实践方法能够帮助读者更好的理解和执行需求工程工作的细节。

表3-1 本书后面将要介绍的实践方法

活动	有效实践	内容	技术、方法	章节
需求获取	定义项目前景	定义项目前景	问题分析 目标分析 业务过程分析 分析非功能需求	第5章
	控制项目范围	控制项目范围	定义系统边界 编写前景与范围文档	第5章
	实现用户价值	涉众识别	先膨胀后收缩 检查列表 涉众网络	第6章
		涉众描述	涉众的描述特征	第6章
		涉众评估	优先级评估风险评估共赢分析	第6章
	促进用户参与	涉众代表选择	代表采样 使用用户替代源	第6章
		参与策略制定	制定参与基本策略 敏捷方法——用户参与	第6章
	识别并使用各种需求源	涉众分析	涉众分析的各种方法（如前述）	第6章
		硬数据采样	硬数据采样	第6章
		需求重用		第9章
	有效的获取需求	建立有效交流机制	建立合作关系，维护交流气氛 利用适当的交流途径、交流方式	第4章 第6章
		正确使用需求获取方法	面谈/调查问卷/群体面谈 头脑风暴	第8章
			原型	第9章
			观察、民族志 文档分析/需求重用/需求剥离	第10章
	收集和组织需求获取的结果	建立收集和组织需求需求结果的机制	用例/场景模型	第7章
需求分析	为需求建模	通过建模手段明确和理解需求信息	结构化分析模型	第12章 第13章
			面向对象分析模型	第14章

表3-1 本书后面将要介绍的实践方法

活动	有效实践	内容	技术、方法	章节
		使用多种手段从多角度建模相同的内容	多视点方法 Wiegrnga框架 Zachman框架	第11章
	在合适的层次上描述需求	需求细化		第11章
	唯一的标识每一条需求	需求细化		第11章
	划分需求的优先级	确定需求优先级	累计投票 区域划分 Top-N 数据量化	第11章
需求规格说明	使用模版	使用需求文档模版	[IEEE1998]的模版	第15章
	进行良好的写作	综合使用各种描述手段	形式化、半形式和非形式化描述	第15章
		学习有效的写作实践	写作技巧 优秀需求规格说明文档特性	第15章
			需求写作事项 优秀需求特性	第2章
需求验证	验证需求	使用有效方法进行需求的验证和确认	需求评审 原型与模拟 开发测试用例 用户手册编制 利用跟踪关系 自动化分析	第16章
需求管理	建立和维护需求基线	建立和维护需求基线	配置管理 状态维护	第17章
	进行变更控制	进行变更控制	变更控制过程 变更控制事项（策略）	第17章
	建立需求跟踪信息	建立需求跟踪信息	低端/高端的需求跟踪使用 需求依赖	第17章

表3 - 1的实践方法描述了需求工程活动的工作细节，表3 - 2则说明了在需求开发当中涉及的管理实践。

表3-2 本书后面将要介绍的管理实践

	实践方法	内容	章节
过程管理	建立需求工程过程	建立需求过程过程框架	第18章
		选择需求工程过程工作组件	第18章
	维护和使用有效的实践方法	维护和使用有效的实践方法	第18章
	持续改进需求工程过程	评价需求工程过程	第18章
		持续改进需求工程过程	第18章
项目管理	制定需求开发计划	提供充足的资源支持	第19章
		选择需求开发的生命周期	第19章
	建立需求工程团队	组建团队	第19章
		维持团队内部的交流氛围	第19章
	管理需求风险	管理需求风险	第19章

除了上述被系统总结过的实践方法和经验之外，需求工程领域还有很多其他一再经过实践验证的方法、手段、策略和机制。它们当中除了少数的例外情况之外，都在取得成功的同时，也具有不可避免的局限性，它们的应用通常只适合于特定的环境和场景。需求工程师的一个重要任务就是了解这些实践，并为组织或项目选择、定制和应用一些有效的实践。

3.5. 需求开发过程实例

本书在内容组织上，使用的是图3-1所描述的线性需求开发过程，只是将部分需求分析内容（主要是前期需求阶段的分析方法）与需求获取方法组织在了一起。在实践中使用较多的典型需求开发过程是图3-4所描述的过程。但是除了上述两个典型的需求开发过程之外，实践中人们还会依据项目环境的不同建立其他类型的需求开发过程。

图3-6是[SIG2001]给出的一个需求开发过程，它能够非常好地适应软件开发过程的螺旋模型[Boehm1988]，可以用于风险驱动项目的需求开发，其实质就是螺旋模型前两个迭代的体现。图3-6中的协商与建模属于需求分析活动。

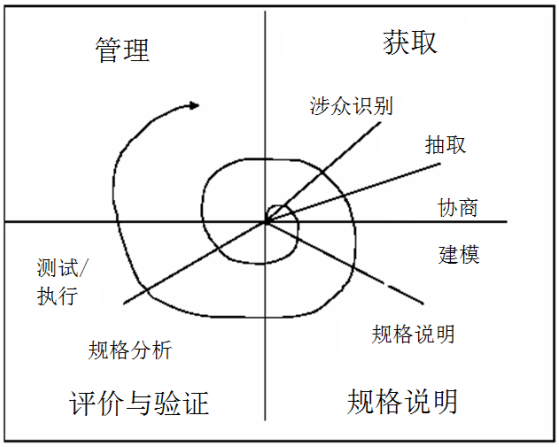


图3-6 适应于软件开发螺旋模型的需求开发过程模型

图3-7是[Hofmann 2001]提及的一个需求开发过程，较为重视原型法的使用，可以很好地适用于软件开发过程的原型模型。图3-7中领域分析和原型开发完成需求获取活动，领域分析和基础及高阶模型开发属于需求分析活动，同级评审、场景走查和涉众反馈属于需求验证活动，编写规格属于需求规格说明活动。可以发现图3-7的需求验证活动与常规过程有所不同，它借重于原型和分析模型的验证作用，再加上涉众评价、反馈，就可以在不依赖于软件规格说明文档的情况下完成需求验证工作，然后再将验证后的高质量需求写成规格文档。

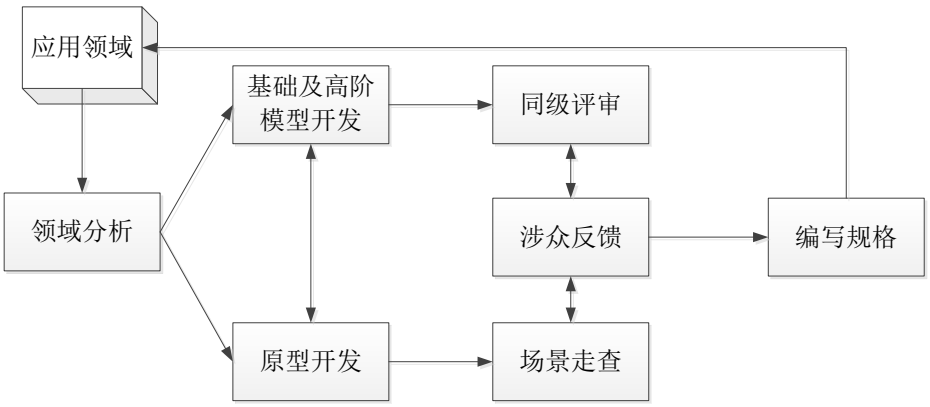


图3-7 适应于软件开发原型模型的需求开发过程模型，源自[Hofmann 2001]

图3-7是[Padula2004]是HP公司需求开发过程，将需求开发划分为两个阶段（系统工程的系统需求开发、软件工程的需求分析）进行，每个阶段都含有获取、分析、规格和验证活动，适用于创新型产品或大型项目开发。[Padula2004]需求开发过程在系统需求开发结束时要进行产品可行性分析，只有可行的产品才会真正进入软件产品开发阶段，可以实现利润的最大化。

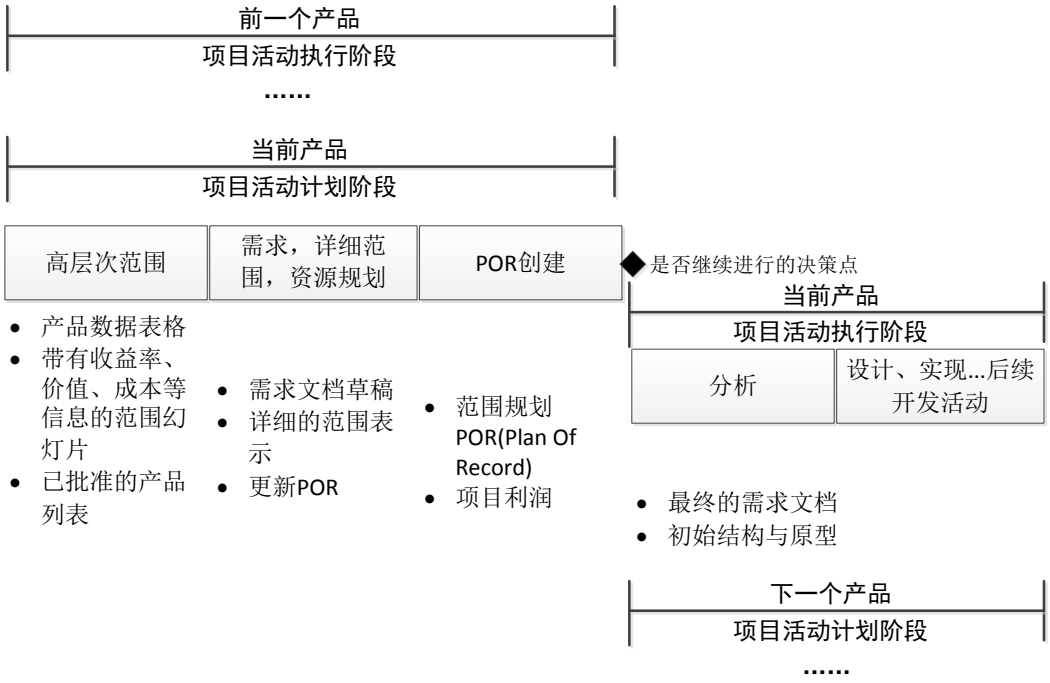


图3-8 HP两阶段的需求开发过程模型，源自[Hofmann2001]

图3-9是[Dörr2008]基于需求实践方法建立的需求开发过程，它将需求实践方法划分为多种类型以进行过程的评价与改进：基础实践完成最基本的需求开发任务，是需求开发过程必备的基础；进阶实践会让需求工程的某些方面变得更好；优化实践是非必需但是能进一步提升需求工程某方面效果的实践；依赖于上下文的实践是仅适用于特殊环境的实践，例如存在复杂业务过程时才可能使用“获取任务与业务过程”实践。

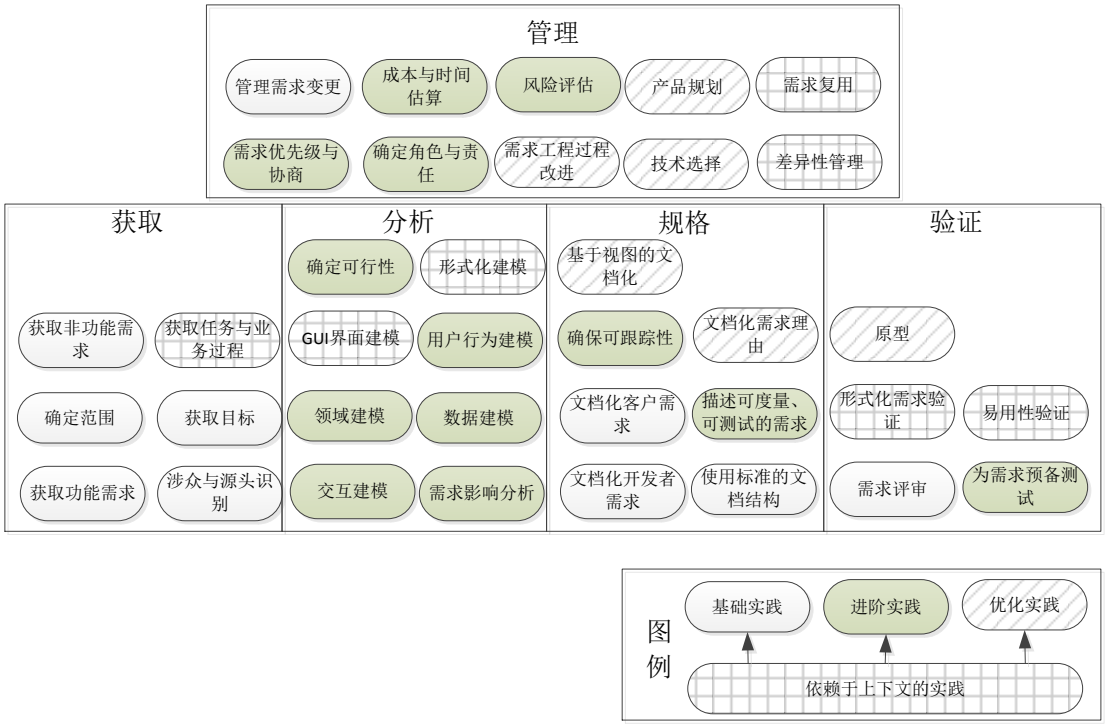


图3-9 集成需求实践方法的需求工程过程模型，源自[Dörr2008]

近些年来比较受欢迎的敏捷软件开发方法强调原则与实践，而不是固定的过程模型，这一特点在需求工程中也有体现。[Cao2008]在调查中发现敏捷软件开发方法主要通过7个实践完成需求开发：

- 面对面的交流胜过写规格说明文档；
- 迭代式需求工程；
- 将需求划分优先级做到极限；
- 通过持续规划管理需求变更；
- 原型法；
- 测试驱动开发；
- 用户评审会议与验收测试。

虽然不依赖特定的过程模型，但是敏捷方法的具体流派还是要执行获取、分析、规格说明、验证、需求管理这几个需求工程活动[Lucia2010]，如表3-3所示。

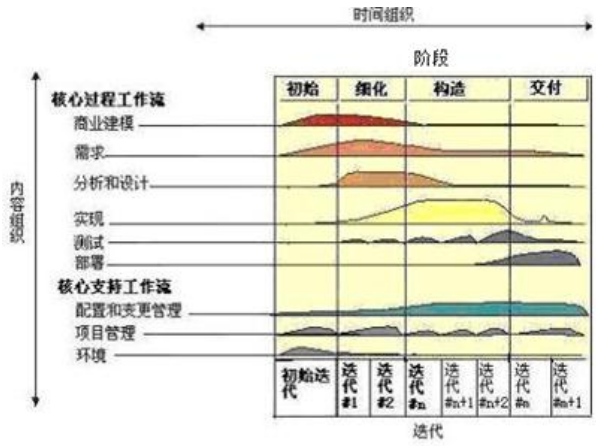
表3-3 XP和Scrum的需求工程活动组织，源自[Lucia2010]

需求活动	XP方法	Scrum方法
需求获取	将需求获取为Story 客户书写User Story	产品负责人（Product Owner）明确叙述产品功能(Product Backlog) 任何涉众都可以参与产品功能的确定

表3-3 XP和Scrum的需求工程活动组织，源自[Lucia2010]

需求活动	XP方法	Scrum方法
需求分析	并非独立阶段 开发时进行分析 客户为User Story划分优先级	功能(Backlog)精化会议 产品负责人划分产品功能优先级 产品负责人分析需求可行性
需求规格	User Story和验收测试用例作为需求文档 软件产品本身作为持久（文档）信息 面对面的交流	面对面的交流
需求验证	测试驱动 验收测试 频繁反馈	评审会议
需求管理	短的规划周期 跟踪User Story 按需重构	蓝图规划会议 跟踪产品功能项 对产品功能变更需求

RUP在强调采用最优实践的同时给出了一个可定制的过程框架[Jacobson1999]，如图3-10左边所示。在RUP过程框架中，业务建模、需求和分析与设计三个工作流共同完成需求开发工作，详细工作如图3-10右边所示。



商业建模：
1. 评价组织的当前状态，包括支持新系统的能力；
2. 探索当前业务的过程、角色和职责；
3. 识别与评价业务过程改造的潜在策略；
4. 开发反映业务内容的领域模型。

需求：
1. 与项目涉众紧密接触，理解他们的需要；
2. 定义项目的范围；
3. 通过恰当的建模，探索功能、非功能、业务规则、用户界面等需求；
4. 识别项目中新的及变更的需求，并明确他们的优先级。

分析与设计（仅列举需求开发工作）：
1. 理解并分析系统的需求。

图3-10 RUP的需求开发工作示意

3.6. 需求开发过程与软件工程过程的相互影响

需求的好坏对后续软件开发有着极其重要的影响这一观念已经是开发者共识了，近些年的实践研究进一步发现：不仅仅是最终制品——需求，整个需求开发过程都会对其后续的软件开发过程产生重要影响。

[Verner2005]发现相比于需求方法本身的好坏，需求方法与软件开发方法的适配性更会影响项目的成败。这也就是说，需求开发方法与软件开发方法是否适配，比结果需求的好坏更能影响项目的成败。

需求开发过程之所以对后续软件开发过程有重要影响，并不仅仅是因为它的结果制品——需求，是后续开发过程的工作基础，更要认识到需求开发过程中还会产生很多正性信息（例如前景与范围定义、涉众描

述、分析模型、需求特征描述等）。如果单纯从产生软件需求规格这个任务来看，这些正性信息都是不必要的，但它们对后续软件开发过程的影响则是明显的。也就是说，为了让整个软件开发团队的工作能够更加顺利，需求工程师需要完成很多看上去似乎不属于其本职工作的任务，这就是“团队”的含义。Daniela Damian等人的研究[Damian2003, 2005, 2006]证实了上述推断，其详细研究结论如图3-11和表3-4所示。

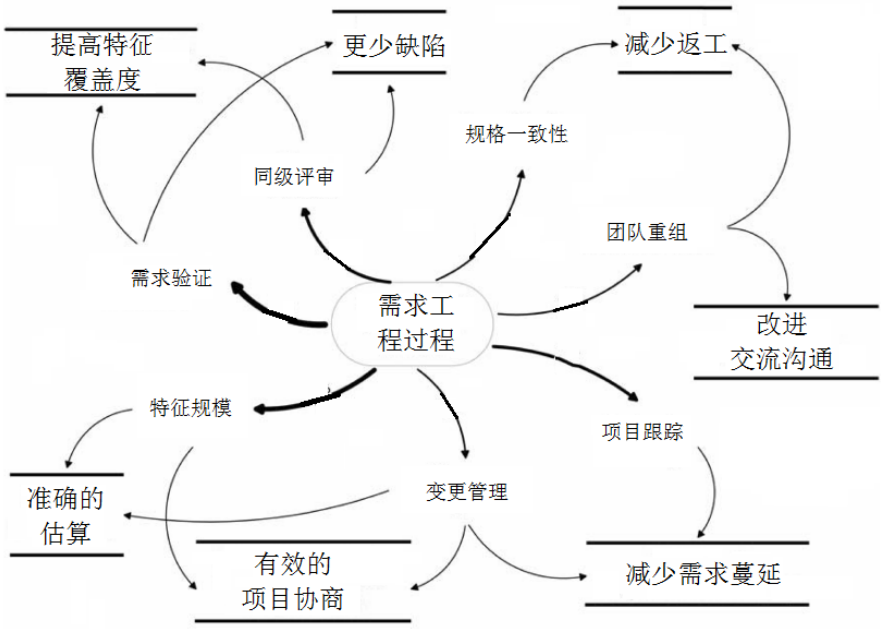


图3-11 需求工程过程对后续软件开发工作的影响示意

表3-4 需求工程过程对软件开发的影响效果及路径

需求工程收效	具体操作	影响趋向
提高生产率	问题理解	↑
	交流沟通	
	过度沟通	↓
	有效的沟通	↑
	开发者的非正式决策	↑
	返工	↓
提高质量	运营支持请求	↓
	交付后缺陷	↓
提升风险管理	估算	改进50%
	特征覆盖度	↑
	需求蔓延	↓
	项目协商	↑

引用文献

[Boehm1988] Boehm, B. W., A Spiral Model of Software Development and Enhancement, Computer, v.21 n.5, p.61-72, May 1988.

[Cao2008] Cao, L.,Ramesh, B., Agile Requirements Engineering Practices: An Empirical Study, IEEE Software, January/February 2008, pp60-68.

[Damian2003] Damian, D., Chisan J., Vaidyanathasamy, L., Pal, Y., An Industrial Case Study of the Impact of Requirements Engineering on Downstream Development,Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE'03), 2003.

[Damian2005] Damian, D., Chisan J., Vaidyanathasamy, L., Pal, Y., Requirements Engineering and Downstream Software Development: Findings from a Case Study, Empirical Software Engineering, 10, 255–283, 2005.

[Damian2006] Damian D., Chisan, J., An Empirical Study of the Complex Relationships between Requirements Engineering Processes and Other Processes that Lead to Payoffs in Productivity, Quality, and Risk Management, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 32, NO. 7, JULY 2006.

[Dörr2008] Dörr,J., Adam, S., Eisenbarth, M., Ehresmann, M., Implementing Requirements Engineering Processes: Using Cooperative Self-Assessment and Improvement, IEEE Software, May/June 2008, pp71-77.

[Hickey2003] Hickey, A. M., Davis, A. M., Requirements Elicitations and Elicitation Technique Selection: A model for Two Knowledge-Intensive Software Development Processes, In HICSS '03: Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 3, pp. 96.1. 2003.

[Hofmann 2001] Hofmann, H.F. and Lehner, F., *Requirements Engineering as a Success Factor in Software Projects*, IEEE Software, vol. 18, no. 4, July/Aug. 2001, pp. 58–66.

[IEEE1998] IEEE Std 830-1998, *IEEE Recommended Practice for Software Requirements Specifications*, Institute of Electrical and Electronics Engineering, Inc., 1998.

[Jacobson1999] Jacobson, I., Booch, G., and Rumbaugh, J., The Unified Software Development Process, MA: Addison Wesley Longman, Inc.1999.

[Kotonya1998] Kotonya, G. and Sommerville, I., *Requirements Engineering – Processes and Techniques*, John Wiley & Sons, UK, 1998.

[Larman2003] Larman, C., Basili, V. R., Iterative and Incremental Development: A Brief History, Computer, June 2003, pp47-56.

[Loucopoulos1995] Loucopoulos, P., and Karakostas, V., *System Requirements Engineering*,

McGraw-Hill Book Company Europe, 1995.

[Lucia2010] Lucia, A. D., Qusef, A., Requirements Engineering in Agile Software Development,JOURNAL OF EMERGING TECHNOLOGIES IN WEB INTELLIGENCE, VOL. 2, NO. 3, AUGUST 2010.

[Maiden2012] Maiden, N., Framing Requirements Work as Learning, IEEE SOFTWARE, MAY/JUNE 2012,pp8-9.

[Martin2002] Martin, S., Aurum, A., Jeffery, R., Paech, B., Requirements Engineering Process Models in Practice, AWRE'2002, 2002.

[Nguyen2002] Nguyen, L. and Armarego, J. and Swatman, P., *Understanding Requirements Engineering: a Challenge for Practice and Education*, School of Management Information Systems, Deakin University, 2002.

[Nguyen2003], Nguyen, L., Swatman, P. A., Managing the requirements engineering process. Requir. Eng. 8(1): 55-68, 2003.

[Padula2004] Padula, A., Requirements Engineering Process Selectionat Hewlett-Packard, Proceedings of the 12th IEEE International Requirements Engineering Conference (RE'04), 2004.

[Robert2002] Robert L. Glass, *Facts and Fallacies of Software Engineering*, Addison-Wesley, 2002.

[Shaw1990] M. Shaw, Prospects for an Engineering Discipline of Software, IEEE Software, vol. 7, no. 6, 1990, pp. 15–24.

[Siddiqi1996] Siddiqi, J. and Shekaran, M.C., *Requirements Engineering: The Emerging Wisdom*, IEEE Software, Mar. 1996, pp. 15-19.

[SIG2001] IPSJ Requirements Engineering SIG, SIG technical reports, Winter workshop in Kanazawa, 2001.

[Verner2005] Verner, J.M., Cox, K., Bleistein, S., Cerpa, N., Requirement Engineering and Software Project Success: An Industrial Survey in Australia and the U.S, in Proceedings of AWRE, Adelaide, Australia, 2005.