第 5 章

需 求 工 程

本章学习目标

- 掌握需求工程的基本概念，理解需求工程过程的基本活动和过程模型。
- 掌握典型的需求获取方法，并能在实践中灵活运用。
- 掌握典型的需求建模与分析方法，并能在实践中灵活运用。
- 能够根据需求规约描述模板进行定制，描述实际系统的需求规约。
- 理解需求确认与验证的基本概念和原则，并能在实践中灵活运用需求确认与验证的典型方法。
- 理解需求管理的基本概念，能够熟练运用需求管理工具进行需求管理。

需求工程是软件开发生命周期中至关重要的一个环节。通过在需求工程过程各阶段灵活运用各种需求工程方法和技术，可以帮助涉众明确软件项目开发的目的、范围、功能与非功能需求，从而为软件项目的成功研发奠定坚实的基础。本章首先对需求工程的基本概念和过程模型进行概述，进而详细介绍需求获取、需求建模与分析、需求规约、需求确认与验证、需求管理等需求工程各个阶段的活动，而后通过本章小结，概要总结本章的主要内容和学习重点。

5.8 节的综合习题有利于读者巩固和检查对本章所学基本知识的掌握情况；5.9 节的基础实践案例分析给出了如何灵活利用本章所学方法解决特定场景下需求获取规划、需求建模与分析、需求确认等方面的样例，同时还给出了相对应的基础实践练习帮助读者检验其灵活运用的能力；5.10 节给出的引申阅读材料可以帮助读者加深和扩展对相关知识的理解，学习学术界和工业界最新的研究成果和实践经验。

5.1 需求工程概述

需求工程（Requirement Engineering，RE）是系统、规范地利用需求获取、建模、分析、确认、验证等过程明确需求，通过需

求规约记录需求，并对需求全生命周期产生的制品进行管理的科学方法，是软件工程和系统工程的一部分。

需求工程的研究旨在提供一系列工程化的方法，以帮助涉众明确其对于待构建系统的目标与期望，对需求进行建模和分析，并通过确认与验证获得合理的需求，进而通过规约技术无歧义地描述需求，从而指导后续软件开发过程。

本节将分别从需求定义、需求类别和特点、需求工程过程三个方面对需求工程进行概要介绍。

5.1.1　相关定义

术语 1："需求"

IEEE 610.12—1990 标准将术语"需求(Requirement)"定义为：

(1) 用户解决某个问题或者达到某个目标所需要的条件或能力。

(2) 一个系统或系统组件为了实现某个契约、标准、规格说明(规约)或其他需要遵循的文件而必须满足的条件或拥有的能力。

(3) 对(1)或(2)中所描述的条件或能力的文档化表示。

其中，(1)主要强调对用户需要和目标的定义，(2)则强调对系统所需具备的条件和属性的定义。

由于"需求"定义具有一定的主观性，因此并不存在一个一致认可、清晰客观的"需求"术语。公认的"需求"定义需要遵循的原则是：**需求是定义做什么的，而不是定义怎么做的。**

例如，"用户希望获得个性化商品推荐服务"是需求，在定义需求时不需要考虑或明确描述用什么方法或模型为用户提供个性化商品推荐服务。

优秀的需求描述应该具备 7 种特性，包括完整性、正确性、无歧义性、可行性、有优先级、必要性和可验证性。

1. 完整性

完整性又分为单个需求描述的完整性和整体需求描述的完整性。单个需求描述的完整性，是指每一项需求都必须将期待系统实现的功能描述得清晰完整，使开发人员能获得理解、设计和实现这些功能所需的所有必要信息。整体需求描述的完整性，是指针对待建系统的所有需求描述是无遗漏的，即整体需求描述能够涵盖当前定义的待建系统的所有需要，将来的需求变更中"新需求"主要是由于外部环境的变化而产生的。整体需求描述的完整性是软件工程师追求的目标，但通常难以达到。

以外卖平台系统为例，若需求描述中仅包含与店家菜品浏览、购物车、提交订单、订单分配、外卖推荐等相关的描述，那么这样的需求描述是不完整的。其中，没有涉及安全性需求，如身份验证、访问约束、用户特权级别等。

2. 正确性

正确性又分为需求描述本身的正确性和需求描述意图的正确性。需求描述本身的正确性，是指每一项需求都必须准确地陈述要开发的功能，通常可以通过需求分析、需求检测或需求验证技术来检验。需求描述意图的正确性，是指每一项需求都是用户意图的真实反映，通常可以通过需求确认技术来检验。

以外卖平台系统的一项需求描述"系统提供的在线支付功能应能支持用户线下支

付”为例，由于在线支付和线下支付存在冲突，因此需求描述不正确；若需求“系统应该支持用户线下支付”经过需求确认被用户否认，则该需求描述意图不正确。

3. 无歧义性

同一读者对一项需求只能做出一种解释，且不同读者对其也能在理解上达成一致。歧义主要是由于自然语言语义的模糊性导致的。避免歧义的有效方法主要包括制定需求描述规范以及需求确认和验证。

以外卖平台系统为例，需求“锁屏状态下应该支持调整屏幕亮度”对不同的读者而言就存在歧义：一种理解是“锁屏状态下应该支持系统（根据设置自动）调整屏幕亮度”，另一种理解是“锁屏状态下应该支持用户调整屏幕亮度”。

4. 可行性

每一项需求都必须在已知系统和环境的权限和能力范围内是可以实施的。为评估需求的可行性，在每次获取需求后的需求整理阶段，最好由技术人员和领域专家分别对需求的技术可行性和业务可行性进行评估。

以外卖平台系统为例，对于需求“系统必须以 100％的可靠性提供 7×24h 服务”，技术人员可以将其评估为技术不可行，进而要求需求提出者进行修改。当然，要求开发团队对所有的需求都进行早期的可行性评估是很难操作的，故可以将评估放在重要的需求项上。

5. 有优先级

由于人力、物力有限，需要对需求进行优先级评估，以区分其迫切需要的程度。需求优先级可以从多维视角进行评估，如期望-必备-可选视角、业务-技术-项目管理视角、成本-价值视角、收益-损失-成本-风险视角等。由于背景、职责、偏好不同，不同涉众对于同一组需求的优先级评价往往不一致。因此，需求优先级评估具有一定的主观性，优先级往往是相对的。

以外卖平台系统为例，从期望-必备-可选视角考察需求“系统必须提供商家菜品推荐功能”的优先级，涉众既可能评估其为期望型需求，也可能评估其为必备型需求或可选型需求。

6. 必要性

必要性是指需求对于待建系统而言是必须满足的。由于每一个需求的满足必定会需要一定的成本，不必要的需求满足会导致成本增加、进度延误，更有甚者，还可能导致项目失败。因此，确认需求的必要性对于确定需求的优先级至关重要。但需求的必要性又是随着时间变化的，以前必要的需求可能成为可选需求或无意义的需求；而现在可选需求可能随着用户要求提升成为必要需求。

以外卖平台系统为例，要求“系统必须为店家提供个性化的打折促销定制接口”，在系统建设初期可能为非必要需求，暂时不予满足。但随着外卖平台的竞争加剧，该需求可能转变为必要需求。

7. 可验证性

每一项需求都应该能够通过设计测试用例或验证方法来确定系统是否满足该需求。如果需求不可验证，则难以判定其实施是否正确，进而给今后的系统验收带来隐患。

以外卖平台系统为例，要求“系统必须提供友好的用户界面”，由于没有定义客观

的度量标准，无法验证系统是否满足此项需求。

术语 2："涉众"

在需求工程中，"涉众(Stakeholder)"又称为利益相关者，特指与待建系统相关的所有利益相关方，包括客户、用户、上下游环境或资源提供商、开发团队、受到待建系统影响的特定人群等。其中，客户和用户最容易被混淆：客户专指为待建系统提供资金支持的利益相关方，用户专指系统的使用者。两者在项目中既可以是同一主体，也可以是不同主体。

例如，若某外卖平台是某公司或部门委托开发的，则该公司或部门就是其客户，平台交付后由客户运营，依托外卖平台开展业务的买方、卖方等则是其用户。若某外卖平台是某公司 IT 部门自行研发的，后期拟通过自行运营交易抽成的方式获取利益，则该公司将成为依托外卖平台开展服务的平台方，则该公司既是该平台的用户又是客户。此外，与这个平台相关的支付交易提供商、快递服务提供商等都是该平台的涉众。

术语 3："问题空间"与"解空间"

在需求工程中，"问题空间(Problem Space)"特指待建系统应解决的问题、满足的需求或达到的业务目标；"解空间(Solution Space)"特指针对问题的技术解决方案，即构建何种系统能够解决问题、满足需求，并最终达成目标。

需求工程方法和技术的主要目的就是帮助涉众理清需求，准确定义问题空间，并明确对解空间的设计约束。

5.1.2　需求分类

由于来源不同、层次不同、粒度不同，需求可以细分为以下类别。

1. 业务需求

业务需求又分为目标层业务需求和操作层业务需求。

目标层业务需求侧重于描述对系统开发具有决定权的主体对系统的高层次目标要求，即希望通过系统开发达成的目标。

注意：目标层业务需求通常来源于项目投资人、组织机构负责人、实际用户管理者、市场营销部门或产品策划部门等。需求规格说明书中使用前景和范围部分的内容往往来源于此。

操作层业务需求侧重于描述待建系统必须支持完成的实际业务、任务、工作流程或必须遵守的业务规则等。需求分析师通常可以从中进一步细化出具体的功能需求和非功能需求。

注意：操作层业务需求通常来源于待开发系统支持的各类业务的从业人员，这些从业人员很可能也是将来系统的用户。需求规格说明书中的功能需求与非功能需求往往来源于此。

以外卖平台系统为例，业务目标"外卖平台系统应该支持企业成为全国最大的餐饮中介服务提供商，为百万级餐饮服务提供商及亿级消费者提供餐饮服务"即为目标层业务需求，业务规则"只有已支付订单才能进行配送"即为操作层业务需求。

2. 用户需求

用户需求是指描述用户使用待开发系统需要完成什么任务以及如何完成的需求。用户需求通常是在业务需求定义的基础上通过用户访谈、调查等，获得用户使用的场

景，并进行整理得到的。

注意：由于用户背景、职位以及在待开发系统中承担的角色等的不同，其提出的需求往往具有零散性和片面性，容易导致用户需求之间存在不一致或矛盾，需要通过需求分析或需求协商来解决。

以外卖平台系统为例，消费者的需求“希望能够在下单后 20min 内拿到餐品”和快递小哥的需求“希望在用餐高峰期和天气不好时延长送餐时限”间存在矛盾。

3. 系统需求

系统需求是指描述系统的顶级需求。系统可以包含软件子系统和硬件子系统，也可以只包含软件子系统。如果一个系统只包含软件子系统，那么系统需求就等同于软件需求；如果一个系统既包含软件子系统又包含硬件子系统，则系统需求包含软件需求和硬件需求。

注意：本章中后续讨论的系统均是只包含软件子系统的系统，即软件系统，后续涉及的各类需求工程方法和技术作用的对象主要指软件需求。

以外卖平台系统为例，该系统可以包含外卖平台商家子系统、骑手子系统和消费者子系统，这三个子系统均为软件子系统，该系统的系统需求即软件需求。

4. 软件需求

软件需求是指用户对目标软件产品在功能、行为、界面、性能、设计约束、质量保障等方面的期望。软件需求往往是由需求工程师主导，通过与利益相关方的多次交流、协商而最终形成的面向软件设计人员的需求。可见，软件研发是否能够取得成功，取决于软件需求是否真正体现了用户需求。

注意：软件需求和用户需求不同，软件需求的主体是软件系统，描述的是对软件功能和非功能需求的期望，而用户需求的主体是用户，描述的是用户的期望。需求工程师需要能够从用户需求中识别软件需求。

以外卖平台系统为例，用户需求“希望在用餐高峰期和天气不好时延长送餐时限”对应的软件需求为“系统应该提供送餐时限自定义功能，并支持根据设定的条件和当前状态自动计算送餐时限”；用户需求“希望能够送餐入户”就不是软件能够完成的需求，因而不是软件需求。

5. 功能需求

功能需求是指目标软件产品必须实现的软件功能或软件系统应具有的外部行为。功能需求源于业务需求和用户需求，使用户能够利用这些功能来完成任务，达成业务目标，而开发人员则根据这些功能需求来设计软件。

注意：功能需求通常用以下三种方式描述。

(1) 系统应该支持何种功能或提供何种服务的描述。

(2) 系统在特定条件下的行为描述。

(3) 系统不能做什么的陈述。

以外卖平台系统为例，“系统应该支持根据设定条件和当前状态自动计算送餐时限”“用户账户余额不足时支付失败”等都是功能需求。

6. 非功能需求

非功能需求是指目标软件产品为满足软件需求而必须具备的、除功能需求以外的需求。非功能需求描述软件产品必须具备的品质，与功能需求相辅相成密不可分。非

功能性需求具有不易发现、不易表达、不易实现、不易测试等特点。

注意：非功能需求可以包括对外部界面的要求、对外部接口的要求、对输入数据的要求、对各种质量属性的要求等，具体的非功能特性可参考 1.3 节。

以外卖平台系统为例，界面需求“系统应该支持不少于 4 种界面风格”、数据需求“上传的菜品图片分辨率不低于 512×384”、性能需求“系统支付功能应该能够支持百万用户并发访问”等都是非功能需求。

7. 设计约束

设计约束是指软件开发人员在设计和实现目标软件产品时必须满足的要求和必须遵从的标准规范。设计约束包括企业或组织对目标软件产品设计、构造、运维上的特殊限制和源于法律法规、风俗文化的要求。

注意：非功能需求与设计约束的主要区别在于，非功能需求是对功能需求的补充约束，主要来源于操作层业务需求和用户需求；而设计约束则主要描述对目标软件产品宏观、整体的要求，主要来源于目标层业务需求。

以外卖平台系统为例，时间约束“系统必须在半年内上线”、资源约束“系统必须运行在云平台上”、文化约束“系统应该使用红色显示注意事项或警告标志”、法律约束“系统只能允许具有营业执照和餐饮服务许可证的商家上线提供服务”等都是设计约束。

各类需求之间的关系如图 5.1 所示。需求分析师通过调研获取业务需求和用户需求后，通过整理分析得到系统需求，再从中分离出由软件子系统实现的需求形成软件需求，而后将其进一步具体化为对目标软件产品的功能需求和非功能需求，对于目标软件产品宏观、整体的要求则记为设计约束。

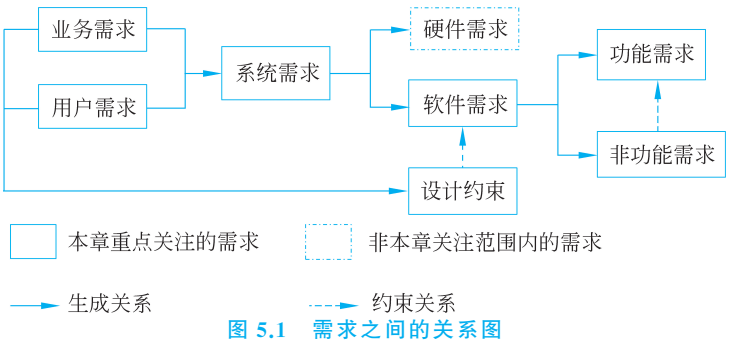

图 5.1　需求之间的关系图

5.1.3　需求工程过程

1. 需求工程过程概述

需求工程过程是指利用需求获取、需求建模与分析、需求规约、需求确认与验证、需求管理等活动收集、整理、建模、分析涉众需求，以最终获得待建目标软件的正确、完整、一致、无歧义的需求规格说明文档，并对整个过程进行管理的系列需求活动序列。

如图 5.2 所示，需求工程过程的输入包括领域知识、涉众需要和环境信息。

（1）**领域知识**。领域知识主要包括与待建系统相关的领域规章制度、运作规则、习惯等，往往容易被遗忘或忽略。

（2）**涉众需要**。涉众需要主要包括涉众对待建系统的各种显性或隐性需求。

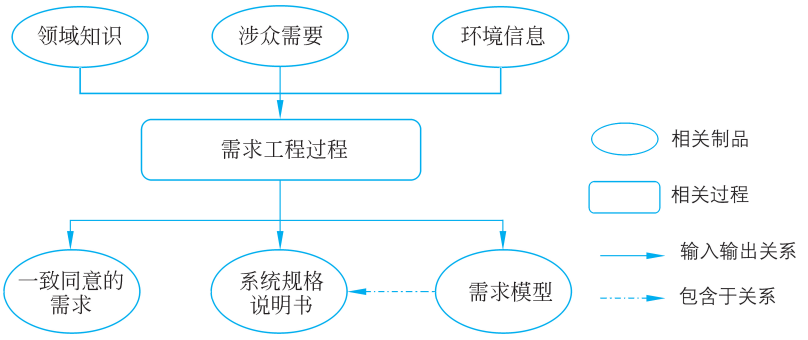

图 5.2 软件需求工程过程

(3) **环境信息**。环境信息主要包括将与待建系统进行交互的已有系统、运行环境中对待建系统运行所必需的信息和其对待建系统提出的能力要求。

需求工程过程的输出包括一致同意的需求、系统规格说明书以及需求模型。

(1) **一致同意的需求**。一致同意的需求是指消除了歧义和矛盾后得到的一致认可的需求,一般用自然语言描述,主要面向非技术人员。

(2) **系统规格说明书**。系统规格说明书是指系统功能的详细规格说明,一般需要按照特定需求规格描述规范撰写,主要面向软件开发人员。

(3) **需求模型**。需求模型是指按照特定需求建模与分析方法构建的模型,一般按照建模方法的要求撰写,常常出现在需求规格说明书的不同章节中,主要面向软件开发人员。

2. 需求工程活动

构成需求工程过程的需求工程活动主要包括需求获取、需求建模与分析、需求规约、需求确认与验证以及需求管理 5 个子活动。

1) 需求获取

需求获取主要指从人、资料或者环境中获取待建系统需求的过程。一般通过咨询涉众、查阅系统文档、整理领域知识、开展市场需求调研等方法获得。需求获取一般包含以下需求工程子活动:领域及项目背景资料收集、项目目标和范围确定、调查对象确定、需求获取方法选择、需求获取方法执行等。

2) 需求建模与分析

需求建模与分析主要指通过建模来整合需求获取阶段得到的各种信息,通过分析发现需求中存在的矛盾、冲突,通过协商解决问题,从而得到涉众共同认可的软件需求的过程。需求建模与分析一般包含以下需求工程子活动:需求建模、需求精化、需求优先级排序、需求协商等。

3) 需求规约(文档化)

需求规约(文档化)主要指将各方共同认可的软件需求编写成规范文档的过程。其中,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(或用例文档),软件需求被写入需求规格说明。需求规约一般包含以下需求工程子活动:文档模板定制、文档编写等。

4) 需求确认与验证

需求确认与验证主要指检查需求模型/需求规格说明文档,以确保模型/文档中所包含的需求能够真实、正确地反映项目意图的过程。其中,需求确认重在确保需

求描述能够真实地反映项目意图，需求验证重在确保需求满足正确性、无歧义性等特定属性。需求确认与验证一般包含以下需求工程子活动：需求确认、需求验证等。

5）需求管理

需求管理主要指对需求工程过程中产生的各种制品进行管理，以保障需求在软件产品整个生命周期中可利用、可管理、可追溯。值得一提的是，需求管理与其他活动不同，是贯穿整个软件生命周期的活动。需求管理一般包含以下需求工程子活动：需求跟踪、需求基线管理、需求变更管理等。

以上各个需求工程活动根据项目需要可组成各种不同的需求工程过程，最终得到符合要求的文档化需求规格服务。各需求工程活动之间的关系如图 5.3 所示。

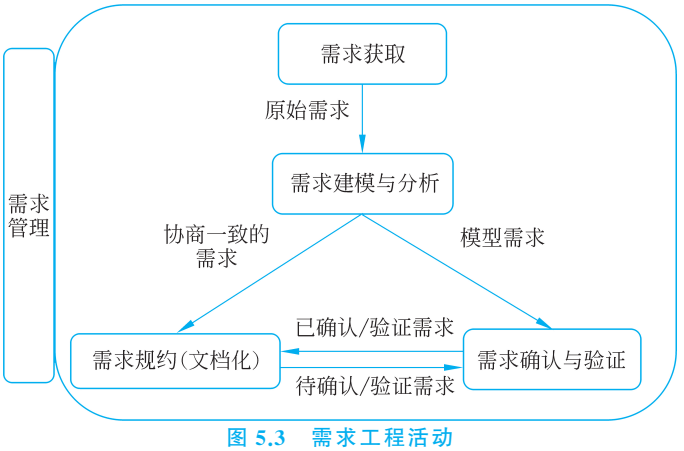

图 5.3　需求工程活动

3. 需求工程过程模型

同软件开发模型一样，需求工程过程模型也可以分为瀑布模型、迭代模型、增量模型等，可根据项目需要灵活选择。

1）瀑布模型

瀑布模型将软件需求工程过程划分成需求获取、需求建模与分析、需求确认与验证和需求规约 4 个步骤，4 个活动顺序开展，如图 5.4 所示。对于成熟、简单、小型系统的需求规约生成可以采用这种需求工程过程模型。

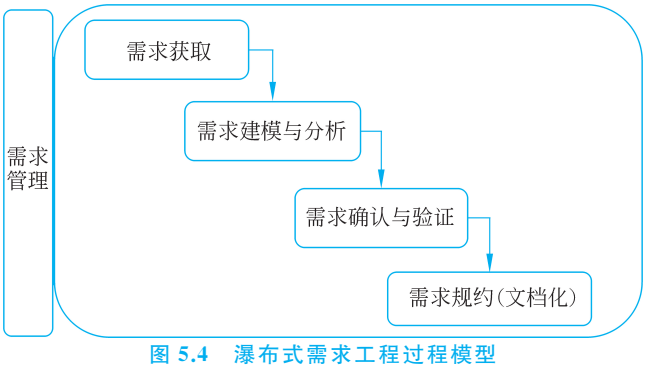

图 5.4　瀑布式需求工程过程模型

2）迭代模型

迭代模型是一种带反馈回路的需求工程过程模型，如图 5.5 所示，也是实际使用

过程中最常用的过程模型。对复杂、大型的系统而言，一次需求获取不可能获得所有需求，需要通过建模与分析，发现缺陷后补充获取；同样，经过建模与分析得到的需求也可能在确认与验证中发现问题，进而需要回溯等。

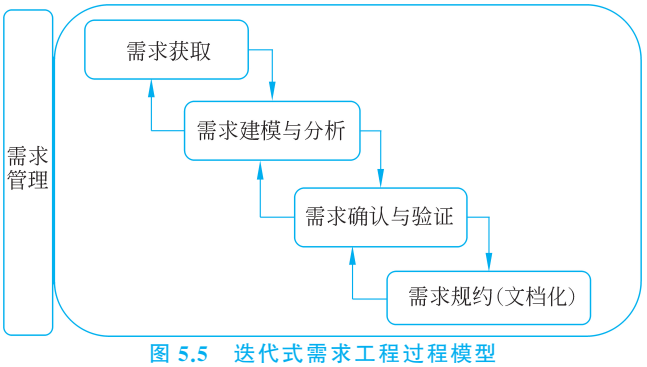

图 5.5 迭代式需求工程过程模型

3）增量模型

增量模型是一种划分子系统实施需求工程过程的模型，如图 5.6 所示。对于由多个子系统组成的复杂系统，可以根据每个子系统的特征，独立选择合适的需求工程模型开展工作。

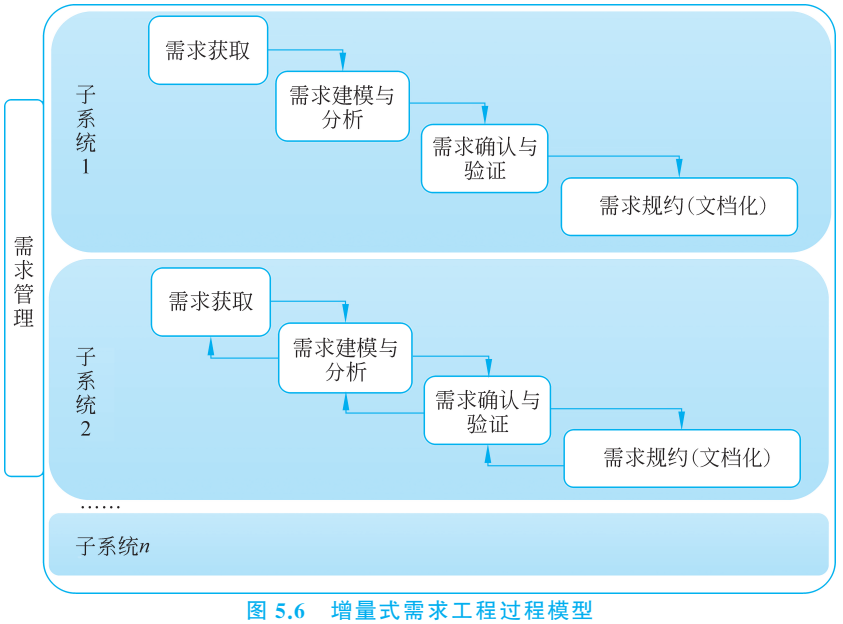

图 5.6 增量式需求工程过程模型

5.1.4 代表性的需求工程方法

需求工程过程的实践和需求活动的实施需要需求工程方法的指导。目前，需求工程领域的代表性方法主要包括面向目标的方法、面向主体的方法、问题驱动的方法、面向场景的方法、基于环境建模的方法等。其中，面向目标的方法将“目标”作为系统需求的源头，按照目标分解关系、精化关系、操作化关系等自顶向下对需求条目进行梳理，构建层次化的软件需求模型；面向主体的方法使用“主体(参与者)”来刻画能够主动适应环境变化、自主活动的软硬件实体，基于组织环境上下文发现主体之间的依赖

关系，以识别系统的早期需求；问题驱动的方法主要指问题框架方法，即从现实世界的实体以及这些实体产生的现象之间的期望关系出发，推断出待开发软件的需求；面向场景（也称情景）的方法是一种自底向上的需求工程方法，主要从具体的场景实例中获取用户的需求，并建立用户与系统交互的行为模型；基于环境建模的方法以系统环境模型为基础，帮助需求工程师识别系统的环境关注点，引导其对这些环境关注点进行系统化的分析，从中引导出系统和环境的交互能力需求。

限于篇幅，本章主要介绍面向主体的方法和面向场景的方法如何指导需求获取、需求建模与分析等需求工程活动。

5.2　需求获取

需求获取（Requirement Elicitation），又称作需求抽取，是指发现、识别并收集软件需求的活动，也是涉众之间互相沟通、识别其需要的过程。简单来说，需求获取是一个从相关人员、资料和环境中得到软件系统开发所需的相关信息的过程，因此，需求获取不但涉及技术问题，而且涉及社会交往问题。

鉴于需求获取的关键性、困难性和易出错性，需求获取成为整个开发过程中最需要交流的活动，必须通过涉众间高度的合作、有效的沟通才能成功。需求分析人员并不是简单照抄用户所说的话就能得到软件需求，而是需要利用科学的方法从涉众所提供的大量信息中分析和理解涉众真正的需求。

本节首先介绍了需求获取的任务和原则，进而介绍了软件需求获取的传统方法、辅助方法和智能化方法。最后，对常用的需求获取工具进行了简要介绍。

5.2.1　需求获取的任务和原则

需求获取任务通常可以划分为确定需求开发计划、建立项目的目标和范围、确定调查对象、实地收集需求信息、确定非功能需求 5 个子任务，如图 5.7 所示。

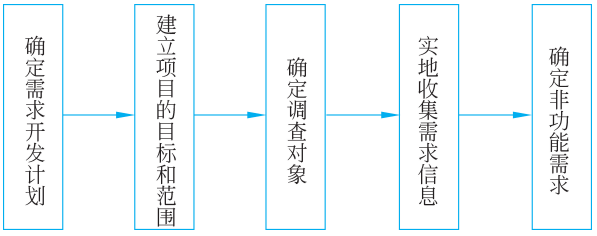

图 5.7　需求获取任务

1. 确定需求开发计划

基本任务：确定需求开发的实施步骤，安排好收集需求活动的具体工作与进度。

在确定需求开发计划时，需要遵循下列基本原则。

- 在安排需求开发的实施步骤、规划需求活动的进度和时间时，只考虑与需求开发相关的工作，不能将软件开发其他阶段（如设计阶段）的工作也在此考虑，以保证需求开发活动有充分的人员、时间和经费保障。
- 在安排进度时，应考虑困难性和灵活性。例如，在收集用户需求时，用户可能由于某些临时的计划或安排与约定的交流时间产生冲突，从而不一定能保证