第5章 需求分析

目标：

(1) 掌握需求分析的概念。

(2) 掌握需求分析的过程。

(3) 掌握需求分析的方法和工具。

(4) 掌握需求分析文档的撰写，了解需求评审的相关概念。

5.1 需求分析的概念

随着现代社会的工业化、信息化，以及计算机应用技术的迅速发展，各种各样的软件系统随之被大量开发出来。软件开发人员面临着应用系统的复杂性越来越高，规模越来越大，而且支持系统开发的基础软件本身也存在着复杂性、多样性、不间断性、自适应性等一系列问题。在这种状况下，软件需求分析的重要性和必然性就充分体现出来了。

软件需求分析是软件生命周期的一个重要阶段。只有通过软件需求分析，才能把软件功能和软件性能的总体概念描述为具体的软件需求规格说明，从而奠定软件开发的基础。软件需求分析质量的好坏直接影响着整个软件工程的进展和最终的结果。

软件工程所要解决的问题往往十分复杂，尤其是当建立一个全新的软件系统时，认识问题的本质是一个较为困难的过程。一般情况下，开发软件的技术人员精通计算机技术，但是并不熟悉用户的业务领域；而使用软件的用户虽然清楚自己的业务，但是并不掌握计算机技术。因此，通常对同一个问题，技术人员和用户之间可能存在着认识上的差异。面对这样的问题，在开始设计软件之前，就需要由既精通计算机技术，又熟悉用户应用领域的系统分析人员对软件方面的内容进行认真而细致的需求分析。

5.1.1 软件需求定义

美国电气和电子工程师协会(Institute of Electrical and Electronics Engineers，IEEE)软件工程标准词汇表(1997 年)中将“需求”定义如下。

(1) 用户为解决某一问题或者达到某个目标所需要的条件或能力。

(2) 系统或系统部件要满足合同、标准、规格说明及其他正式规定的文档所需要的条件或者能力。

(3) 反映上面两方面的文档说明。

目前虽然对软件需求的定义有着不同的看法，但是通常认为软件需求是指软件系统必须满足的所有功能、性能和限制。软件需求分析是将用户对软件的一系列要求、想法转变为软件开发人员所需要的有关软件的技术说明。

在实际工作中，通常把软件需求细化为三个不同的层次：功能需求、性能需求和领域需求。功能需求包含了组织机构或者用户对系统、产品的高层次目标要求和低层次的使用要求，定义了开发人员必须实现的软件功能，使得用户能够完成自己的工作，从而满足业务需求。图 5-1 描述了软件需求各组成部分之间的关系。

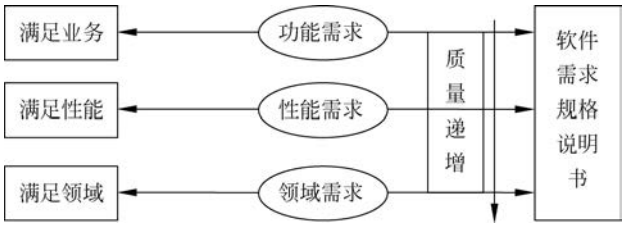

图 5-1　软件需求各组成部分之间的关系

1. 功能需求

功能需求是用来描述组织或用户的各层次目标，通常问题定义本身就是业务需求。业务需求必须具有业务导向性、可度量性、合理性及可行性。这类需求既来自于高层，如项目投资人、购买产品的客户、实际用户的管理者、市场营销部门或者产品策划部门；也来自于低层的具体业务要求，如为完成某项任务而采用的具体业务流程。功能需求是一类软件区分其他软件的本质需求，如财务软件的功能需求是不同于合同管理软件的功能需求。

2. 性能需求

为了有效地完成软件的功能需求，需要对软件的性能做出要求，如输入输出响应速度、界面的友好性、存储文件的大小、稳健性、可维护性、安全性等。性能要求是软件质量的高层次要求，CMM(软件能力成熟度模型)成熟度越高的软件公司，在性能需求方面做得越好。性能需求是所有软件的共性需求，不是软件彼此之间区分的本质特征。

3. 领域需求

软件的类别千差万别，不同领域的软件对需求有着比较明显的差别，涉及国家军事、政治和经济方面的软件有着特定的领域要求，如法律、法规和道德需求，高保密性和安全性需求。涉及自动控制的会引起生命危险的软件，对容错、纠错和维护响应时间的需求非常高。单纯的信息管理软件则对数据安全性的要求比较高。

5.1.2　软件需求分析

在软件工程中，软件需求分析是指在建立一个新的软件系统或改变一个现存的软件系统时，在描述新系统的目的、范围、定义和功能方面所做的全部工作。

软件需求分析是一项软件工程活动,它使系统分析人员能够描绘新软件系统的功能和性能,指明软件和其他系统元素的接口,并且建立系统必须满足的约束。通过对问题及其环境的理解与分析,对涉及的信息、功能及系统行为建立模型,将用户的需求精确化、完整化,最终形成软件需求规格说明。

实际上,软件需求分析是对系统的理解与表达的过程。理解是指开发人员充分理解用户需求,对问题及环境的理解、分析和综合,逐步建立目标系统的模型。表达是指经过调查分析后建立模型,并在此基础上把分析的结果用规格说明等有关文档完全地、精确地表达出来。这一系列的活动构成了软件开发生命周期的需求分析阶段。

在软件工程发展的历史中,人们在很长一段时间里一直都认为需求分析是整个软件工程中最简单的一个步骤。但是,近年来越来越多的人认识到,需求分析是整个软件设计过程中最为关键和重要的环节。在进行软件需求分析时,如果开发人员不能正确掌握用户的真正要求,那么最后开发出的软件产品是不可能满足用户需求的。

5.1.3 需求分析的要求

软件需求分析的目的是使用户需求具体化,并最终使需求满足用户要求。通常软件需求分析的基本要求包括以下几方面。

(1) 完整性。在需求分析中,没有遗漏用户的任何一个必要的要求。

(2) 一致性。在需求分析中,用户和开发人员对于需求的理解应当是一致的。

(3) 现实性。需求应当是以现有的开发技术作为基础来实现的。

(4) 有效性。需求必须是正确且有效的,保证可以解决用户真正存在的问题。

(5) 可验证性。已经定义的需求是可以准确验证的。

(6) 可跟踪性。已经定义的功能、性能可以被追溯到用户最初的需求。

5.1.4 需求分析的重要性

软件需求分析在软件开发过程中具有举足轻重的地位,它是开发出正确的、高质量的软件系统的重要保证之一。因此,无论是在学习软件工程的过程中,还是在开发软件产品的实践中,都要对软件需求分析有足够的重视。

软件需求分析在整个软件开发过程中的重要性主要表现为以下几方面。

1. 软件需求分析是获得用户需求的有效途径

开发软件产品是为用户服务的,要想开发出真正满足用户需要的软件系统,首先必须掌握用户的需求。对软件需求的深入理解是软件产品开发工作获得成功的前提条件,否则如果开发出的软件产品不能真正满足用户需要,软件开发人员即使把设计工作和编程工作做得再好也无济于事。

2. 软件需求分析是项目取得成功的关键因素

软件需求分析是一个项目的开始,也是项目建设的基础。在很多失败的项目中,大部分原因是项目需求不明确造成的。项目的整体风险表现在需求分析不明确、业务流程不合理

等方面，造成用户不愿意使用所开发出的软件产品，或者很难使用所开发的软件产品，从而使得项目失败。

3. 软件需求分析是软件设计的坚实基础

软件需求分析过程实际上就是确定用户需求的过程。由于用户掌握自己的需求，但是却不懂得如何使用计算机技术来加以实现，而软件设计人员往往缺乏实际事物的运作过程和商业过程的技巧，这时可以通过系统分析人员缩短商业领域和计算机技术之间的距离。从掌握需求信息的用户那里获得可用数据，并且把它转换成可以使用的形式，从而形成下一阶段软件设计的依据。

4. 软件需求分析是软件质量保证的重要阶段

软件需求分析阶段是项目的开始阶段，同时也是软件质量控制的开始时期。在软件生命周期的每一个阶段都要采用科学的管理方法和先进的技术手段，并且在每个阶段结束之前都要从技术和管理两个角度进行严格审查，待审查合格之后再开始下一阶段的工作。这就使得软件开发工程的整个过程以一种井然有序的方式进行，从而保证了软件的质量，提高了软件的可维护性。

5.2 需求分析的过程、内容和任务

软件需求分析是软件生命周期中非常重要的环节。在完成可行性研究之后，如果软件系统的开发是可行的，就要在软件开发计划的基础上进行需求分析。实际上，软件需求分析是一个不断认识和逐步细化的过程。在这个过程中将软件开发计划中确定的范围逐步细化到可以详细定义的程度，然后分析和提出各种不同的问题，并且为这些问题找到有效的解决方法。

5.2.1 需求分析的过程

软件需求分析的过程主要包括获取用户需求、分析用户需求、编写需求文档、进行需求评审几个阶段，如图 5-2 所示。

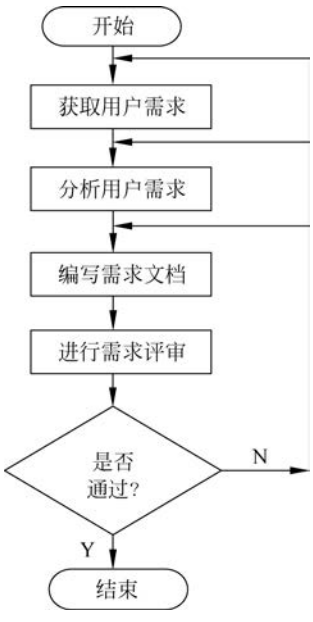

图 5-2 需求分析的过程

1. 获取用户需求

获取用户需求阶段，必须充分地了解用户目标、业务内容、系统流程，通过各种方式与用户进行广泛的交流，然后确定系统的整体目标和工作范围，弄清楚所有数据项的来源及数据的流动情况。

2. 分析用户需求

分析用户需求阶段，分析人员从数据流和数据结构出发，根据功能需求、性能需求和环境需求，分析是否满足要求、是

否合理,然后把其综合成系统的解决方案,给出目标系统的逻辑模型。分析和综合工作需要反复进行。

3. 编写需求文档

编写需求文档阶段,需要把已经确定的需求清晰、准确地描述出来,描述需求的文档称为需求规格说明书。需求文档可以采用结构化语言编写文本型的文档,也可以建立图形化的模型,还可以使用数学上精确的形式化逻辑语言来定义需求。

4. 进行需求评审

需求分析直接关系到软件项目能否顺利进行,因此要求进行需求评审来控制需求分析的质量。需求评审可以通过内部评审、同行评审、用户评审等方式进行。在需求分析评审中,用户的意见是第一位的。

5.2.2 需求分析的内容

软件需求分析的前提是准确、完整地获得用户需求。用户需求可以分为功能需求和性能需求两类。功能需求定义了系统应该做什么,系统要求输入哪些信息、输出哪些信息,以及如何将输入变换为输出;性能需求则定义了软件运行的状态特征,如系统运行效率、可靠性、安全性、可维护性等。

综合起来,应该获取的用户需求内容主要包括以下几个。

1. 物理环境

物理环境是指系统运行的设备地点及位置有哪些,如是集中式的还是分布式的;对环境的要求有哪些,如对温度、湿度、电磁场干扰等的要求。

2. 软件系统界面

软件系统界面是指与其他系统进行数据交换时的内容与格式、用户对于界面的特定要求、用户在操作时的易接受性等。

3. 软件系统功能

软件系统功能是指系统主要能完成的任务,对于运行速度、响应时间或者数据吞吐量的要求,系统运行的权限规定,对可靠性的要求,对扩充性或者升级的要求等。

4. 数据要求

数据要求是指输入输出数据的种类与格式,计算必须达到的精度,数据接收与发送的频率,数据存储的容量和可靠性,数据或者文件访问的控制权限及数据备份的要求等。

5. 文档规格

系统文档规格是指系统要求交付的各种文档、各类文档的编制规范,以及预期使用的对象等。

6. 维护要求

系统的维护要求是指当系统出错时，对错误修改的回归测试要求、系统运行的日志规格，以及可以允许的最大恢复时间的要求等。

5.2.3 需求分析的任务

软件需求分析要完成的任务是深入描述软件的功能和性能，确定软件设计的限制和软件与其他系统的接口细节，定义软件的其他有效性需求。即对目标系统实现的功能提出完整、准确、清晰、具体的要求。因此，需求分析的任务就是借助当前系统的逻辑模型导出目标系统的逻辑模型，重点解决目标系统“做什么”的问题。软件需求分析的实现步骤如图 5-3 所示。

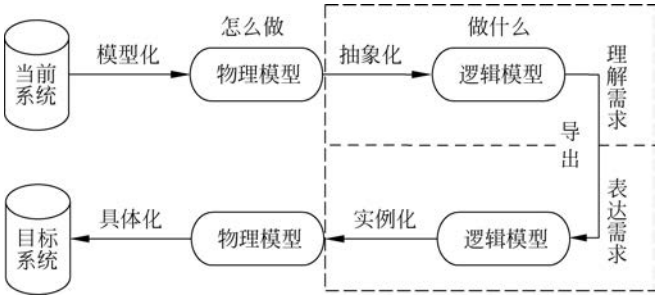

图 5-3 软件需求分析的实现步骤

由图 5-3 可知，软件需求分析的具体任务如下。

1. 确定对软件系统的要求

对软件系统的要求有以下 4 方面。

(1) 系统的功能要求。功能要求就是划分出系统需要完成的所有功能。

(2) 系统的性能要求。性能要求必须考虑到联机系统的响应时间、系统需要的存储容量，以及后援存储、重新启动、安全性等方面。

(3) 系统的运行要求。运行要求主要表现为系统运行时对环境的要求，如支持系统运行的系统软件是什么，采用的数据库管理系统是什么，数据通信接口是什么等。

(4) 将来可能提出的要求。应当明确列出那些现在不属于当前系统的开发范畴，但是据分析将来可能会提出的要求，目的是在设计过程中对系统将来可能的扩充和修改做好准备工作。

2. 分析软件系统的数据要求

任何一个软件系统的本质都是信息处理系统，系统必须处理的信息和系统应该产生的信息在很大程度上决定了系统的构成。因此，必须分析系统的数据要求，这是软件需求分析的一个重要任务。通常分析系统的数据要求采用建立概念模型的方法。软件系统的数据要求就是归纳出目标系统的数据结构和数据之间的逻辑关系，描述系统所需要的输入和输出

数据、数据库、数据类型及数据的获取和处理方法等。

3. 导出软件系统的逻辑模型

在确定目标系统的要求和数据的基础上,通过一致性分析检查,逐步细化软件功能及各个子功能。同时对数据域进行分解直至分解到各个子功能上,以确定系统的构成,最后通过用数据流图、数据字典和主要的处理算法建立目标软件系统的逻辑模型。

4. 修正软件系统开发计划

根据在分析过程中获得的对软件系统的更深入、更具体的了解,可以准确地估计软件系统的成本和进度,从而修正以前制订的开发计划。

5. 开发原型系统

在计算机硬件和其他许多工程产品的设计过程中经常使用原型系统。建造原型系统的主要目的是检验关键设计方案的正确性及系统是否真正满足用户的需要。对于软件系统的开发,使用原型系统可以使用户通过实践获得对未来系统在运行时的更直接和更具体的认识,从而可以更准确地提出和确定用户自身的要求。

5.3 需求分析的方法

软件需求分析方法是由对软件问题的信息域和功能域的系统分析过程及其表示方法组成。信息域包括三种属性:信息流、信息内容和信息结构。需求分析方法有很多种,根据目标系统被分解的方式不同,基本上可以分为三种方法:20 世纪 70 年代开发出的"结构化分析方法"、20 世纪 90 年代初推出的"面向对象分析方法"和 21 世纪初出现的"面向构件分析方法"。

结构化分析(structured analysis,SA)方法是 20 世纪 70 年代中期由 E. Yourdon 等提出的,适用于分析典型的数据处理系统,以结构化方式进行系统定义的分析方法。这个方法通常与 L. Constantine 提出的结构化设计(structured design,SD)方法结合起来使用。

软件需求分析方法首先用结构化分析对软件系统进行需求分析,然后用结构化设计方法进行系统的总体设计,最后是进行系统的结构化编程(structured programming,SP)。

1. 结构化分析思想

结构化分析方法要求软件系统的开发工作按照规定的步骤,使用一定的图表工具,在结构化和模块化的基础上进行分析。结构化分析是把软件系统功能作为一个大模块,根据分析与设计的不同要求进行模块分解或者组合。

在软件工程技术中,控制复杂性的两个基本手段就是"分解"和"抽象"。对于复杂问题,由于人们的理解力、记忆力有限,因此不可能触及问题的所有方面及全部细节。为了将复杂性降低到人们可以掌握的程度,可以把大问题分割成若干小问题,然后分别解决,这就是"分解"。分解也可以分层进行,即首先考虑问题最本质的属性,暂时把细节忽略,以后再逐步添加细节,直至涉及最详细的内容,这就是"抽象"。

结构化分析方法的基本思路如图 5-4 所示。

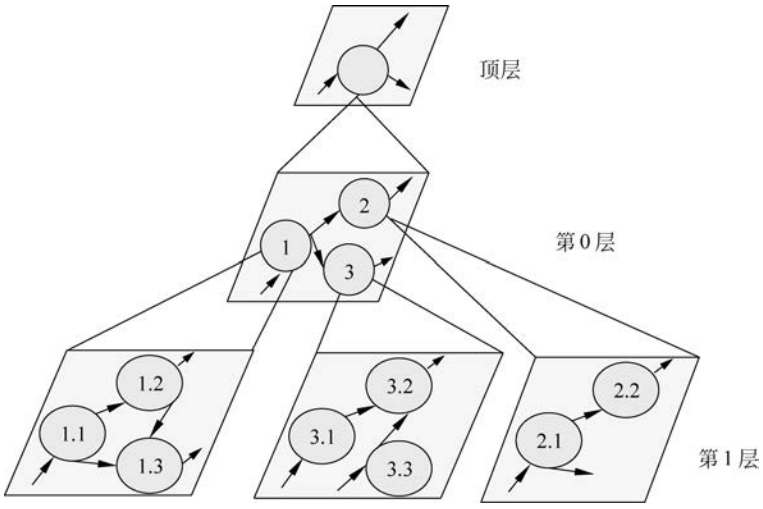

图 5-4 结构化分析方法的基本思路

结构化分析方法使用了“自顶向下,逐步求精”的方式,使人们不至于一下子陷入细节部分,而是有控制地逐步了解更多的细节内容,这有助于理解问题。图 5-4 中的顶层抽象地描述了整个系统,第 1 层(底层)具体地表示了系统的每一个细节,第 0 层(中间层)则是从抽象到具体的逐步过渡过程。按照这种方法,无论问题多么复杂,分析工作都可以有计划、有步骤地进行。

2. 结构化分析步骤

采用结构化分析方法进行系统需求分析,一般包括以下步骤。

(1) 熟悉当前系统的工作流程,构造出当前系统的物理模型。当前系统是指目前正在运行的系统,也是需要改进的系统。通过对当前系统的详细调研,了解当前系统的工作过程,同时收集有关资料、数据、报表等,将所有收集到的信息用图形或文字描述出来,也就是用物理模型来反映对当前系统的理解。

(2) 分析当前系统的物理模型,抽象出当前系统的逻辑模型。当前系统的物理模型反映了系统“怎样做”的具体实现,要构造逻辑模型就要去掉物理模型中的非本质因素,保留本质因素。本质因素是系统固有的,是不随运行环境的变化而变化的。可构造当前系统的逻辑模型,以反映出当前系统“做什么”的功能。

(3) 研究当前系统的逻辑模型,建立起目标系统的逻辑模型。目标系统是指拟开发的新系统。在当前系统逻辑模型的基础上,分析、比较目标系统与当前系统在逻辑上的差别,补充需要变化的部分,明确目标系统确实要“做什么”,这样就可以从当前系统的逻辑模型推导出目标系统的逻辑模型。

(4) 进行目标系统的进一步补充和优化。为了对目标系统做完整地描述,还需要对逻辑模型做进一步补充和优化,其中包括要说明目标系统的人机界面,说明系统中的出错处理、启动与结束、输入输出和系统性能方面等需求的细节。对于系统特有的一些性能和限

制,也需要用适当的形式做出书面记录。

分析阶段结束时,系统分析人员必须和用户再次认真审查系统文件,力争在系统开始设计之前,尽可能地发现其中还可能存在的错误,并且及时进行改正,直至用户确认这个模型确实表达了他们的需求之后,相关的系统文件才能作为用户和软件开发人员之间的"合同"而最终获得确定。

3. 结构化分析工具

结构化分析是一种建模活动,该方法使用简单易读的符号,根据系统内部数据的传递、变换关系,采用自顶向下、逐层分解的策略描述功能要求的软件模型。下面是使用结构化分析方法的一些指导性原则。

(1) 在开始建立分析模型之前,首先必须认真分析问题,切不可在问题没有被很好地理解之前就产生一个带有错误问题的软件模型。

(2) 开发设计模型,使用户能够了解如何进行人机交互。

(3) 记录每个需求的起源和原因,这样可以有效地保证需求的可追踪性。

(4) 使用多个需求分析视图,用来建立数据、功能和行为模型。

(5) 在需求分析中,应当尽量避免需求的含糊性和二义性。

结构化分析方法利用图形等半形式化的描述方法表达需求,形成需求规格说明书中的主要部分。常用的描述工具有以下几个。

(1) 数据流图。描述系统各部分组成及各部分之间的联系。

(2) 数据字典。定义数据流图中每一个图形元素。

(3) 结构化语言、判断表、判断树。详细描述数据流图中不能被再分解的每个加工。

由于分析中的主要依据是数据传递及数据变换所形成的数据流,因此结构化分析一般采用数据流图的分析方法,最终产生需求规格说明书。该文档包括一套数据流图、对数据流图中的成分进行定义的数据字典及对加工逻辑的描述。

4. 结构化分析特点

结构化分析方法一般包括以下特点。

(1) 结构化分析方法是面向数据流的分析方法之一,它采用图形描述方法来建立分析模型,把软件系统描绘成一个可见模型,为系统的审查和评价提供了有利的条件,也为软件开发人员和用户提供了交换信息的方法,还为系统的设计阶段提供了坚实的依据。

(2) 结构化分析方法简单实用,特别适合瀑布模型,易于开发者掌握,在成功率方面仅次于面向对象的方法。

(3) 结构化分析方法适合数据处理领域。为了使结构化分析方法适用于实时控制系统,还可以在数据流图中加入控制流,这是对结构化分析方法的一种扩充。

(4) 结构化分析方法不能提供对非功能需求的有效理解和建模。

(5) 结构化分析方法通常会产生大量的文档,系统需求的要素会被隐藏在许多细节的描述中。

(6) 采用结构化分析方法建立的分析模型,只能是提供给人们阅读的书面文档,不能被机器阅读和运行。

(7) 结构化分析方法一般不容易被用户理解,因而很难验证模型的真实性。

5.4 需求描述工具

软件需求分析方法最初是作为人工使用而开发出来的,但是对于一些大型的软件项目,用人工方法进行分析比较困难。因此,针对这些方法开发出了一些利用计算机的自动工具来帮助分析人员进行需求分析,这样在很大程度上改善了系统分析的质量,提高了系统分析的效率。

软件需求分析的自动工具按照其不同的表现形式分为如下两类。

第一类主要是利用图形记号进行分析,产生一些图示,协助问题进行分解,自动生成和维护系统的规格说明,其特点是将智能处理应用到问题的规格说明中。

第二类是一种特殊的、以自动方式处理的表示方法,用需求规格说明语言来描述需求,其特点是可以产生有关规格说明的一致性和组织方面的诊断报告。

软件需求分析的描述工具有数据流图、数据字典、结构化语言、判定表、判定树、层次方框图、Warnier 图、IPO 图、需求描述语言等,这里只介绍前 5 种工具。

5.4.1 数据流图

数据流图(data flow diagram,DFD)是一种从数据传递和加工的角度,以图形的方式描述数据流从输入到输出的移动变换过程。

在数据流图中一般要用到 4 种基本符号,如图 5-5 所示。

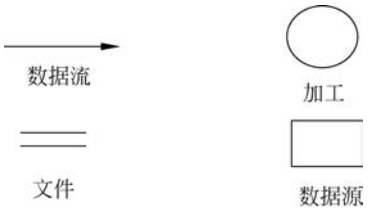

图 5-5 数据流图的基本符号

1. 数据流

数据流是数据在系统中的传输路径,由一系列成分固定的数据项组成,其方向可以是从加工流向加工,从加工流向文件,或者是从文件流向加工。在数据流图中,数据流用带箭头的直线表示。

2. 加工

加工也称为数据处理,用来表示对数据流进行某些加工或处理,是把输入数据转变成输出数据的一种变换,如对数据的算法分析和科学计算。每一个数据加工都应该有一个名字来概括其内容,用来表示其含义。

3. 文件

文件是存储数据的工具,表示数据的静态存储。数据可以存储在磁盘、磁带和其他存储介质中,必须对文件进行命名。文件名应与它的内容一致,写在文件符号内。在数据流图中要注意指向数据文件的箭头方向。读数据的箭头是指向加工处理的,写数据的箭头是指向数据存储的,双向箭头表示既有读数据又有写数据。

4. 数据源(终点)

数据源或终点代表系统外部环境中的实体,可以是人、物或者其他软件系统。它们发出或接收系统的数据,使用起来并不严格,其作用是提供系统和外界环境之间关系的注释性说明,使得数据流图更加清晰。

在绘制数据流图的过程中,应当注意以下几点。

(1) 数据的处理可以是一个程序、一个模块,还可以是一个连贯的处理过程。

(2) 文件是指输入或者输出文件,它可以是文件、文件的一部分、数据库的元素或记录的一部分等。

(3) 数据流和文件是两种不同状态的数据。数据流是指流动状态的数据,而文件是指处于静止状态的数据。

(4) 当目标系统的规模比较大时,为了能清晰地描述和易于理解,通常采用逐层分解的方法,然后画出各分层的数据流图。在分解时,要注意考虑其自然性、均匀性、分解度等一些因素。自然性是指概念上要合理、清晰;均匀性是指把一个大问题尽量分解为规模均匀的几个部分;分解度是指将问题分解的粒度,一般应分解到基本加工为止。

(5) 对数据流图分层细化时,必须保持数据的连续性,即细化前后对应功能的输入数据和输出数据一定要相同。

5.4.2 数据字典

数据字典(data dictionary,DD)是软件需求分析阶段的另一个有力工具。数据流图描述了系统的分解过程,直观而且形象,但是没有对图中各个成分进行准确且完整的定义。数据字典是为数据流图中的每一个数据流、文件、加工及组成数据流或文件的数据项做出说明。

数据流图和数据字典一起构成了系统的逻辑模型。没有数据字典,数据流图就不严格;没有数据流图,数据字典也不起作用。在数据字典中,建立严格一致的定义有助于提高分析人员和用户之间的交流效率,避免许多误解的发生。随着系统的改进,字典中的信息也会发生变化,新的数据会随时加入进来。

数据字典用于定义数据流图中各个图形元素的具体内容,为数据流图中出现的图形元素做出确切的解释。数据字典包含 4 类条目:数据流、数据存储、数据项和数据加工。这些条目按照一定的规则组织在一起,以构成数据字典。在定义这些规则时,常用的符号如表 5-1 所示。

表 5-1 数据字典的常用符号

符 号	含 义	示 例
=	被定义为	
+	与	X=a+b 表示 X 由 a 和 b 组成
[…|…]	或	X=[a|b]表示 X 由 a 或 b 组成
m{…}n	重复	X=2{a}b 表示重复 2～6 次 a
{…}	重复	X={a}表示 X 由 0 个或多个 a 组成
(…)	可选	X=(a)表示 a 在 X 中可能出现,也可能不出现
"…"	基本数据元素	X="a"表示 X 是取值为字符 a 的数据元素
..	连接符	X=1..9 表示 X 可取 1～9 中的任意一个值

例如，数据流“应聘者名单”由若干应聘者姓名、性别、年龄、专业、联系电话等数据项组成，那么“应聘者名单”可以表示为：应聘者名单{应聘者姓名＋性别＋年龄＋专业＋联系电话}。而数据项“考试成绩”可以表示为：考试成绩＝0..100。

又如，某教务系统的学生成绩数据库文件中的数据字典的描述可以表示如下。

- 文件名：学生成绩。
- 记录定义：学生成绩＝学号＋姓名＋{课程代码＋成绩＋必修[选修]}。
- 学号：由 6 位数字组成。
- 姓名：2～4 个汉字。
- 课程代码：8 位字符串。
- 成绩：1～3 位十进制整数。
- 文件组织：以学号为关键字递增排列。

数据字典的实现方法既可以采用全人工过程，也可以采用全自动过程或者混合过程。无论使用哪一种方法来实现，数据字典都具有以下特点。

(1) 通过名字可以方便地查询数据定义。

(2) 能够容易地修改和更新信息。

(3) 不重复在规格说明中其他组成部分已经出现的信息。

(4) 可以单独处理描述每个数据元素的信息。

(5) 定义的书写方法简单并且严格。

5.4.3 结构化语言

结构化语言是一种介于自然语言和形式化语言之间的半形式化语言。虽然使用自然语言来描述加工逻辑是最为简单的，但是自然语言往往不够精确，可能存在二义性，而且很难用计算机处理。形式化语言可以非常精确地描述事物，而且还可以使用计算机来处理，但是用户却不容易理解。因此，可以采用一种结构化语言来描述加工逻辑，它是在自然语言的基础上加入了一定的限制，通过使用有限的词汇和有限的语句来严格地描述加工逻辑。

结构化语言主要使用的词汇包括祈使句中的动词、数据字典中定义的名词或数据流图中定义过的名词或动词、基本控制结构中的关键词、自然语言中具有明确意义的动词和少量的自定义词汇等。一般不使用形容词或副词。另外，还可以使用一些简单的算术运算符、逻辑运算符和关系运算符。

结构化语言中的三种基本结构的描述方法如下。

(1) 顺序结构。由自然语言中的简单祈使语句序列构成。

(2) 选择结构。通常采用 IF…THEN…EISE…ENDIF 结构和 CASE…OF…结构。

(3) 循环结构。通常采用 DO WHILE…ENDDO 结构和 REPEAT…UNTIL 结构。

【例 5-1】 某学院依据每个学生每学期已修课程的成绩制定奖励制度。如果优秀比例占 60％以上，表现优秀的学生可以获得一等奖学金，表现一般的学生可以获得二等奖学金；如果优秀比例占 40％以上，表现优秀的学生可以获得二等奖学金，表现一般的学生可以获得三等奖学金。

可以对上述例题用结构化语言描述加工逻辑，具体表现形式如下。

计算某学生所获奖学金的等级：

```
IF 成绩优秀比例≥60 % THEN
    IF 表现 = 优秀 THEN
      获得一等奖学金
    ELSE
      获得二等奖学金
    ENDIF
ELSEIF 成绩优秀比例≥40 % THEN
    IF 表现 = 优秀 THEN
      获得二等奖学金
    ELSE
      获得三等奖学金
    ENDIF
ENDIF
```

5.4.4 判定表

判定表用来描述一些不容易用语言表达清楚或者需要很大篇幅才能用语言表达清楚的加工逻辑。在一些数据处理中，其数据流图的处理需要依赖多个逻辑条件的取值，但是这些取值的组合可能构成多种不同的情况，对应地需要执行不同的动作，在这种情况下使用结构化语言来描述就显得很不方便，应该使用一种描述机制来清晰地表示复杂的条件组合与动作之间的对应关系。判定表就是解决这一问题的有力工具。

一张判定表由 4 部分组成，如表 5-2 所示。

表 5-2 判定表的一般结构

所有的判断条件	各种条件的组合
所有可能操作	条件组合对应的操作

判定表左上部列出所有的判断条件；左下部列出所有的可能操作；右上部的每一列表示各种条件的一种可能组合，填入 T 或 Y 表示条件成立，填入 F 或 N 表示条件不成立，空白表示条件成立与否都不影响操作；右下部的每一列表示一种条件组合相对应的操作，填入√表示在该列上部规定的条件下做该行左边列出的操作，空白表示不做该项工作。

【例 5-2】 将例 5-1 中给出的奖励条件再进行细化。每个学生每学期已修课程成绩的比例情况：优秀比例占 60%以上，并且良以下比例小于 20%，其中表现优秀的学生可以获得一等奖学金，表现一般的学生可以获得二等奖学金；优秀比例占 60%以上，并且良以下比例小于 30%，其中表现优秀的学生可以获得二等奖学金，表现一般的学生可以获得三等奖学金；若优秀比例占 40%以上，并且良以下比例小于 20%，其中表现优秀的学生可以获得二等奖学金，表现一般的学生可以获得三等奖学金；若良以下比例小于 30%，其中表现优秀的学生可以获得三等奖学金，表现一般的学生可以获得四等奖学金。

采用判定表给出加工逻辑，其主要步骤如下。

(1) 列出所有的判断条件，填写判定表的左上限。

(2) 列出所有的可能操作，填写判定表的左下限。

(3) 计算所有可能的，且有意义的条件组合，确定组合的个数，填写判定表的右上限。

(4) 将每种组合所指定的操作填写在判定表右下限相应的位置。

(5) 合并相同的操作,简化规则。

(6) 将简化后的判定表重新排列。

奖学金发放判定表如表 5-3 所示。

表 5-3 奖学金发放判定表

条件	优秀比例≥60%	T	T	T	T	F	F	F	F
	优秀比例≥40%					T	T	F	F
	良以下比例≤20%	T	T	F	F	T	T	F	F
	良以下比例≤30%		T	T				T	T
	表现优秀	T	F	T	F	T	F	T	F
	表现一般	F	T	F	T	F	T	F	T
操作	一等奖学金	√							
	二等奖学金		√	√		√			
	三等奖学金				√		√	√	
	四等奖学金								√

5.4.5 判定树

判定树是判定表的图形化表示。由于判定表不直观,因此需要认真推敲才能看出其中的含义。判定树比判定表直观得多,它是采用一种树图方式来表示多种条件、多个取值所采取的操作。判定树的分支表示各种不同的条件,随着分支层次结构的扩充,各个条件完成自身的取值。判定树的叶子给出应完成的操作。

很明显,使用判定树的优点是直观、易于掌握和易于使用。但是它也有明显的缺点,判定树的简洁性不如判定表,数据元素的同一个值往往需要重复多遍,而且越接近判定树的叶子重复次数就越多。

【例 5-3】 采用判定树表示例 5-2,如图 5-6 所示。

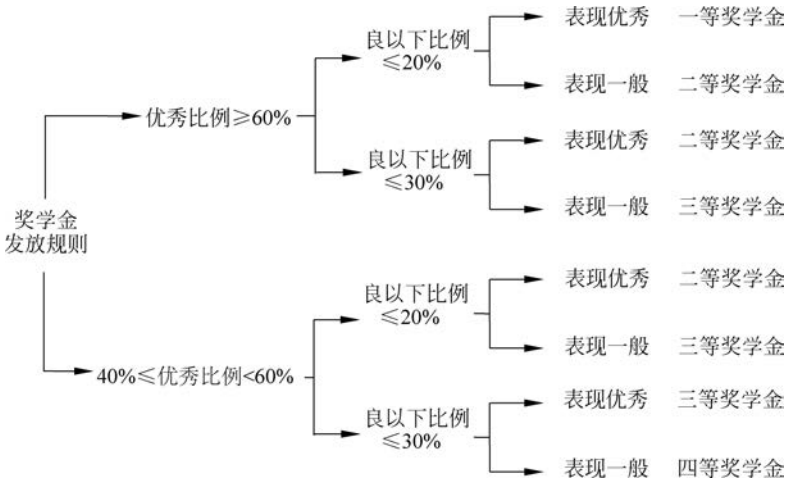

图 5-6 奖学金发放判定树

5.5 需求分析文档

在需求分析阶段,已经确定的用户需求应当得到清晰而准确的描述,软件需求规格说明书是描述需求的主要文档,它以完整的、正确的方式表达了目标系统应该实现的用户需求。软件需求规格说明书应围绕以下4方面组织。

(1) 系统规格方面。系统规格方面主要包括目标系统的总体概貌,系统功能、性能的要求,系统运行的要求,将来可能的修改扩充要求。

(2) 数据要求方面。数据要求方面主要包括建立数据字典,描绘系统数据要求,给出系统逻辑模型的准确的、完整的定义。

(3) 用户描述方面。从用户使用角度对系统进行描述,主要包括系统功能、性能概述,预期的系统使用步骤与方法、用户运行维护要求等。

(4) 开发计划方面。经过需求分析,对系统开发的成本估计、资源使用要求、项目进度计划等的要求。

5.5.1 需求文档完成的目标

软件需求规格说明书是软件工程项目的重要文档,它相当于用户和开发者之间的一项合同。软件需求规格说明书清楚地描述了软件产品做什么,以及产品的约束条件。它为软件设计提供了一个蓝图,为系统验收提供了一个标准集。所以软件需求规格说明书应当完成下列目标。

(1) 在软件产品完成方面,软件需求规格说明书为用户和软件设计人员之间建立的共同协议创立了一个基础,对要实现的软件功能做出了一个全面描述,帮助用户判断所开发的软件系统是否符合他们的要求,或者如何修改软件才能适合他们的要求。

(2) 在提高系统的开发效率方面,编制软件需求规格说明书的过程可以使用户在开始设计之前周密地思考其全部需求,从而减少以后可能的重新设计、重新编码和重新测试所带来的返工。在软件需求规格说明书中,对各种需求认真地进行复查,还可以在开发早期发现若干遗漏、错误和不一致,以便及时加以纠正。

(3) 在成本计价和编制计划进度方面,软件需求规格说明书对所开发的软件系统的描述是软件系统成本核算的基础,并且可以为各方面的费用提供依据。软件需求规格说明书对软件产品的清晰描述有助于估计所有可能用到的资源,并且可以作为编制计划进度的依据。

(4) 在软件系统的移植性方面,有了软件需求规格说明书,可以容易地开发出可移植的软件系统,从而适应新的用户或新的应用平台。用户也易于移植其软件系统到其他部门,软件设计人员同样也易于把系统移植给新的用户。

(5) 软件需求规格说明书是软件系统不断提高的基础。由于软件需求规格说明书所讨论的是软件系统,而不是开发这个系统的设计,因此软件需求规格说明书是软件系统继续提高的基础。虽然软件需求说明书也可能会被修改,但是原来的软件需求规格说明书还是软件系统进行改进的可靠基础。

5.5.2 需求文档的特点

一份好的软件需求规格说明书应该具有以下特点。

(1) 正确性。正确性是指软件需求规格说明书应当正确地反映用户的真实意图。正确性是需求规格说明书最重要的属性。为了确保各项需求是正确的,开发人员和用户必须对软件需求规格说明书进行确认。

(2) 完整性。完整性是指软件需求规格说明书中没有遗漏一些必要的需求。应该包括该系统应有的全部重要的用户需求;规定每种输入数据的软件响应;全部的术语、图表及文档必须完整,符合需求规范标准。

(3) 一致性。软件需求规格说明书中的各项功能、性能要求应该是相容的,不能互相抵触。如描述同一对象不能存在两个以上的不同术语;要求的某一数据的内部属性不能自相矛盾;两个规定的处理在时间上不能产生冲突。

(4) 清晰性。清晰的需求让人易读易懂,可以采用反问的方式判断软件需求规格说明书是否清晰。

- 文档的结构、段落是否混乱?上下文是否连贯?
- 文档的语句是否含糊其辞?
- 文档的内容是否表达明确?

(5) 可验证性。可验证性是指软件需求规格说明书中的每个功能、性能需求存在有限的人工或机器执行的过程,以确认该需求是否符合用户要求,如“软件系统具有良好的用户界面”的要求,用户和开发人员可以有着不相同的理解,因此是不可验证的。

(6) 可修改性。可修改性是指软件需求规格说明书的组织结构在需求发生变化时,对需求的修改能够保证其完整和一致。如果存在需求规格说明内容的列表、索引和交叉引用表,则当某个需求发生变化时,就可以方便地对软件需求规格说明书中必须修改的部分进行定位和修改。

(7) 可跟踪性。可跟踪性是指在软件系统开发中,每个需求在软件需求规格说明书中可以追溯出其来源。实现可跟踪性的常用方法是对软件需求规格说明书中的每个段落按层编号,每个需求给予唯一的编码。

5.5.3 需求文档编写的一般原则

对于用户需求通常有三种方法编写其需求规格说明书。第一种方法是用户自己描述并且自己编写需求;第二种方法是以用户为主,开发人员和用户共同编写需求;第三种方法是开发人员代替用户编写需求。用户是最终使用软件系统的人,对需求有着比较深入的了解,但是用户缺乏编写需求的技巧和方法,需要开发人员的指导,因此第二种方法比较好。在编写软件需求规格说明书的过程中要遵循以下基本原则。

1. 用户观点

一份好的用户需求应该站在用户的角度来思考问题,是用户能够利用系统来完成什么,而不是系统自己能够完成什么。

2. 整体观念

在软件系统开发初期,最关键的是建立一个高层的需求概况,而不是立即深入到细节。因此需要尽可能全面地发现需求,以及维持一个简单的需求列表。

3. 评估依据

以用户为主编写的需求为软件系统的评估提供了依据,在需求初期就进行适当的估算,可以让用户有一个直观的成本概念,为用户在制定需求实现的先后次序上提供指导。

4. 统筹安排

在制定最终需求时,虽然用户需求都是有用的,但在次序和数量上是不同的。在每一个用户需求明确了成本之后,用户就能够权衡实际成本和需求,安排需求的优先级。

用户对最终需求的选择直接影响到下一步的计划制订。在第一个版本中,用户希望能够实现哪些需求,经过估算后,这些需求是否还能够在这个版本中实现,且需要多长的时间等,这些都是需求对计划的影响。

5.5.4 需求文档的编写格式

编写需求文档(软件需求规格说明书)是为了使用户和开发人员对该软件系统的初始规定有一个共同的理解,使之成为整个开发工作的基础。每个软件开发组织都应该在其开发的项目中采用一种标准的软件需求规格说明模板来编写软件需求规格说明书。目前已有多种实用的模板可以使用,其中来自 IEEE 830—1998 的模板——"IEEE 推荐的软件需求规格说明的方法"(IEEE 1998)就是一个结构良好、适用多种软件项目的、灵活的模板。

下面就以一个从 IEEE 830 标准改写并扩充的软件需求规格说明书模板为例介绍软件需求规格说明书的编写工作,见附录 A.2。

需要指出的是,软件需求的内容是反映用户对拟开发系统特性的要求,而不是反映系统的开发特性。因此在软件需求规格说明书中一般不包括软件设计、开发里程碑、开发详细费用等方面的内容。

5.6 进行需求评审

有时由分析人员所提供的软件需求规格说明书初看起来是正确的,但是在具体实现时就有可能出现一些问题,如需求不清楚、需求不一致等。有时以需求规格说明书为依据编写测试计划时也会发现需求规格说明书中存在二义性。为了对需求分析阶段的工作进行验证和完善,应该对软件功能的正确性,软件需求规格说明书的一致性、完整性和准确性及其他需求予以评审。

5.6.1 需求评审的方法

需求评审具体所做的工作可以归纳为以下 4 方面。

1. 审查需求文档

对于所提交的需求文档，必须进行全面、认真的审查，以防止理解错误和遗漏需求情况的发生。组织一个包括分析人员、用户、设计人员、测试人员等不同代表组成的小组，对需求规格说明书及相关模型进行认真的检查，对于保证软件质量而言是一个非常有效的方法。

2. 设计测试用例

通过阅读需求规格说明书，一般很难确定在特定环境下系统的行为，可以将功能需求作为基础设计测试用例，让用户通过使用测试用例来确认是否达到了期望的要求。还可以从测试用例追溯功能需求以确保没有需求被遗漏，并且确保所有测试结果与测试用例相一致。同时要使用测试用例来验证需求模型的正确性。

3. 编写用户手册

在需求开发早期起草的用户手册可以作为需求规格说明书的参考资料来辅助需求分析，因为一份好的用户手册一般是用浅显易懂的语言描述出所有对用户可见的功能。

4. 确定合格标准

在需求评审阶段，需要确定合格的评审标准，让用户描述什么样的软件系统才是他们需要的和适合他们使用的。应将合格的测试建立在使用情景描述或使用实例的基础之上。

需求评审究竟需要评审什么？通常要细致到什么程度？严格地讲，应当检查需求文档中的每一个需求、每一行文字、每一张图表。评审需求优劣的主要指标有正确性、清晰性、一致性、必要性、完备性、可实现性、可验证性等。

为了保证软件需求定义的质量，通常评审工作应当由专门指定的人员负责，并按照规定程序严格进行。用户、开发部门的管理者、软件设计人员、软件实现人员、软件测试人员都应当参加评审工作。在评审结束时应该有负责人的结论意见及签字，然后才可以转入设计阶段。

5.6.2 需求评审的内容

需求文档的评审是一项精益求精的技术，它可以发现那些具有二义性的或不确定的需求，以及由于定义不清而不能作为设计基础的需求。评审的主要内容如下。

- 系统定义的目标是否与用户的要求相一致。
- 系统需求分析阶段所提供的文档资料是否齐全。
- 需求文档中的描述是否完整、清晰、准确地反映了用户要求。
- 与所有其他系统成分的重要接口是否都已经被描述。
- 所开发项目的数据流与数据结构是否充足且确定。
- 所有图表是否清楚，在不补充说明时能否被理解。
- 系统主要功能是否已经包括在规定的软件范围之内。
- 设计的约束条件或限制条件是否符合实际要求。
- 开发过程中的技术风险有哪些。

- 是否考虑过将来可能提出的软件需求。
- 是否详细制定了检验标准,它们能否对系统定义的成败进行确认。
- 用户是否检查了初步的用户手册。
- 软件开发计划中的估算是否受到了影响。
- 软件需求中是否还有遗漏、重复或不一致的方面。

需求评审一般按照预先定义好的步骤进行,评审内容需要记录在案,包括确定材料,评审员、评审小组对产品是否完整或是否需要开展进一步工作的评判,以及对所发现的错误和所提出问题的总结。评审小组成员对于评审的质量负责,而开发者对于所开发产品的质量负责。

5.6.3 需求评审的测试

需求评审的测试是对软件的需求分析,需求规格说明的最终审查,是质量保证工作中最为关键的一个环节。与测试相近的一个名词是纠错,其目的是定位和纠正错误,保证软件的质量。测试过程如图 5-7 所示。

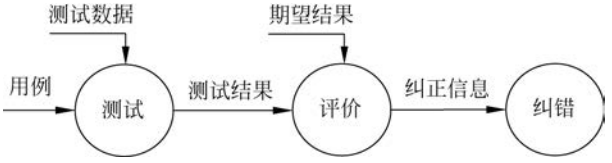

图 5-7 测试过程

如果只是通过阅读软件需求规格说明书,是很难想象出系统在特定环境下的行为的。以功能需求为基础的测试用例可以使项目的参与者了解系统的行为。虽然不可能在实际系统上执行测试用例,但是设计测试用例的过程就可以解释许多需求问题。如果在部分需求稳定时就开始设计测试用例,就可以及早发现问题并以较少的费用解决这些问题。

在开发过程的早期阶段,可以从使用实例中获得概念上的功能测试用例,然后就可以利用测试用例来验证需求规格说明书和分析模型。当分析人员、开发人员和用户通过测试用例进行研究时,他们将对系统如何运行的问题有着更为清晰的认识。这些基于模拟使用的测试用例可以作为用户验收测试的基础。在正式的系统测试中,可以把它们细化为测试用例和过程。

思考题

1. 为什么要进行软件需求分析?请叙述软件需求分析的主要过程。
2. 什么是结构化分析?其结构化体现在哪里?
3. 软件需求分析的原则主要有哪些?
4. 软件需求分析的描述工具有哪些?
5. 软件需求规格说明书由哪些部分组成?
6. 数据字典包括哪些内容?它的作用是什么?
7. 需求评审的主要内容是什么?