第 1 章

初识软件工程

本章要点

- 软件工程的产生；
- 软件的开发方法；
- 软件生存周期；
- 软件开发模型。

学完本章之后你将能够

- 了解软件的特点及软件工程的作用；
- 了解软件的开发方法；
- 了解软件开发模型的特点。

1.1 软件与软件危机

和计算机硬件一样，自 20 世纪 60 年代以来，软件也从规模、功能等方面得到了很大的发展，同时人们对软件质量的要求也越来越高。那么，究竟什么是软件？软件有哪些特征？

1.1.1 理解软件的概念

随着计算机技术的发展，对软件在不同阶段有不同的认识。计算机发展的初期，硬件的设计和生产是主要问题，那时的所谓软件就是程序，甚至是机器指令程序，它们处于从属的地位。软件的生产方式是个体手工方式，设计过程是在人的头脑中完成的，程序的质量完全取决于个人的编程技巧。其后，人们认识到在计算机上增加软件的功能会使计算机系统的功能大大提高，在研制计算机体系时既要考虑其硬件，又要考虑与之配套的软件，并且开始编制一些大型程序系统。这时的生产方式为互助合作的手工方式，认为软件就是程序加说明书。

随后社会对计算机提出了更高的要求，有的大型系统的设计和生产的工作量高达几千人/年(一个人工作一年其工作量为一人/年)，指令数百万条。现在，软件在计算机系统中的比重越来越大，而且这种趋势还在增加。人们感到传统的软件生产方式已经不适

应发展的需要，因此提出把工程学的基本原理和方法引进软件生产中，把软件生产分成几个阶段，每个阶段都有严格管理和质量检验，研制软件设计和生产的方法及工具，并用书面文件作为共同遵循的依据，这时软件的含义就成了文档加程序。

现在对软件的正确理解应该是，软件是计算机系统中与硬件相互依存的部分，它包括程序及其相关文档。程序是计算机程序所需的阐述性资料。

1.1.2 软件的特点

软件包括三个方面：

① 一个或多个计算机程序，这些程序执行时能够提供期望的功能和性能；

② 一个或多个数据结构，这些数据结构使得程序能够完全操纵信息；

③ 一个或多个文档，这些文档描述了程序的分析、设计、实现和维护的细节及使用说明。

软件不同于硬件，它只具备逻辑形式而不具备物理形式。因此，软件具有与硬件不同的特征。

1. 软件的质量是“开发”出来的，不是“制造”出来的

软件和硬件的质量都与“开发”有关，但软件的“制造”只是简单的复制。软件的质量主要取决于软件“开发”，在软件“制造”过程中几乎不会引入新的质量问题，即使存在，也很容易得到改正。而硬件在“制造”过程中有可能会引入新的质量问题。尽管在“开发”和“制造”两种活动中，人的因素都很重要，但两种活动中的参与人员及其所要做的工作却不一样。而且，在两种活动中，虽然目标都是要构造一个“产品”，但是两者采用的方法是不同的。

在 20 世纪 80 年代中期的时候，有人提出了“软件工厂”的概念。但这一概念也不意味着软件“开发”和硬件“制造”能够等同起来。实际上，“软件工厂”这一概念着重于使用自动化工具和环境来进行软件开发。

2. 软件可能会被“废弃”，但不会被“用坏”

软件从被提交给用户使用开始，只存在维护问题，而不存在在使用过程中被“用坏”的问题。硬件则不同，用户购买到硬件并开始使用之后，随着时间的推移，硬件会因为各种原因（如灰尘、震动、高温、磨损、滥用等）造成某些部件失效，从而导致硬件被“用坏”。

在硬件使用的早期，可能会因为设计或制造时的错误或缺陷造成一些问题。当错误或缺陷被改正之后，硬件即可正常运行一段比较长的时期（不同的硬件其时期长短不同），这一时期硬件出错的可能性非常小，称为硬件使用的中期。随着时间的推移，硬件出错的可能性会越来越高，这一时期可以称为晚期，晚期代表着硬件变得越来越不能正常使用了，也就是“用坏”了。

软件的使用过程中出错的可能性与硬件不同。图 1-1 是软件使用过程中随时间变化的错误率曲线的示意图。在软件使用的早期，错误率是比较大的，这主要是因为软件开发时的错误或缺陷造成的。随着时间的推移和错误的改正，软件中潜在的错误越来越少，因而曲线变得越来越平坦。换句话说，软件不会被“用坏”。当然，随着时间的不断推移，软件最终会由于不能满足用户的需要而被“废弃”，软件需要更新。

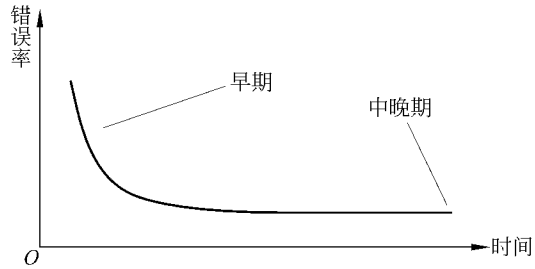

图 1-1　随时间变化的软件错误率曲线

3. 过去的软件大多是“定制”的，而不是“装配”的

现代工业社会中，硬件基本上都是通过集成不同生产厂商的各种零部件，进而“装配”起来的。典型的例子是工厂的流水线作业。硬件设计师的一个主要任务就是要从当前市场上可供选择的零部件中进行选择，通过自己的设计，进而制造出新产品。

软件设计师则没有这么幸运。过去，市场上很少会有类似于硬件“零部件”的软件“零部件”。即便有，大多也是具有完整功能的“软件”或独立功能的模块，而不是“零件”或“部件”。因此，软件大多是为用户专门“定制”的，而不是由现有的软件“零部件”“装配”而成的。现在，这种情况正在发生快速变化。面向对象技术和组件开发技术的广泛应用，使得越来越多的软件部件(也称为软件组件、软件构件)被开发出来，并且可以像硬件部件一样做到“即插即用”。当然，软件部件要做到和硬件部件同等程度的“即插即用”，还需要软件工程师进一步地研究和实践。

1.1.3　产生软件危机的原因

随着计算机应用的逐步扩大，软件需求量迅速增加，规模也日益增长。长达数万行、数十万乃至百万行以上的软件，已不鲜见。美国阿波罗登月计划的软件长达 10000 万代码行，航天飞机软件长达 4000 万行，就是两个突出的例子。

软件规模的增长，带来了它的复杂度的增加。如果说编写一个数十到数百行的程序连初学者也不难完成，则开发一个数万乃至数百万行的软件，其复杂度将大大上升，即使是富有经验的程序员，也难免顾此失彼。其结果是，大型软件的开发费用经常超出预算，完成时间也常常脱期。尤其糟糕的是，软件可靠性往往随规模的增长而下降，质量保证也越来越困难。

众所周知，任何计算机系统均由硬件、软件两部分组成。在计算机应用早期，软件仅包含少量规模不大的程序，应用部门花费在软件上的投资(成本)仅占很小的份额。随着应用面的不断扩大，软件的花费越来越大，所占的百分比也越来越高。B. Boehm 在 1973 年发表的一篇文章中预计，到 1985 年，美国空军的软件费用将上升到计算机总费用的 90%(参阅图 1-2)。即在每 100 元用于计算机投资总额中，软件将花费 90 元。这一预计早已为实践所证实。

庞大的软件费用，加上软件质量的下降，对计算机应用的继续扩大构成巨大的威胁。对这种严峻的形势，软件界的有识之士发出了软件危机的警告。

以下再从维护和生产两个方面，进一步说明出现软件危机的原因。

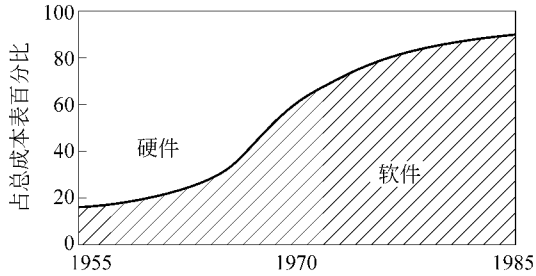

图 1-2 硬件/软件成本变化趋势

1. 软件维护费用急剧上升，直接威胁计算机应用的扩大

根据一些大公司的统计，软件维护费用大约占到软件总花费的 2/3，比开发费用高出一倍。一个大型软件，即使在开发时经过严格的测试与纠错，也不能保证运行中不再出现错误。维护的第一件事，就是纠正软件中遗留的错误，称为“纠错性维护”。在此后的运行过程中，还常常要为完善功能、适应环境变更等原因对软件修改，即进行所谓“完善性维护”和“适应性维护”。不言而喻，软件的规模越大，以上各种维护的成本必然越高。

维护既耗费财力，也耗费人力，为了维护，要占用计算机厂家或软件公司大批软件人员，使他们不能参加新软件的开发。难怪有些文献把维护工作中出现的问题比作冰海中横在前进航道上的冰山，或直称之为维护墙(Maintenance Wall)，视之为软件生产和维护中难以逾越的障碍。

2. 软件生产技术进步缓慢，是加剧这一软件危机的重要原因

有人统计，在过去 30 年中硬件的性能价格比增长了 10^6。一种新器件的出现，其性能较旧器件提高，价格反有所下降，这就是微电子技术创造的奇迹。软件则相形见绌。一方面，软件规模与复杂度增长了几个数量级，但生产方式长期未突破手工业的方式，创建新软件的能力提高得十分缓慢(如图 1-3 所示)。另一方面，很多在早期用“自由化”方法开发的、带有很强“个人化”特征的程序，因缺乏文档而根本不能维护，更加剧了供需之间的矛盾。结构化程序设计的出现，使许多产业界人士认识到必须把软件生产从个人化方式改变为工程化方式，从而导致了软件工程的诞生。

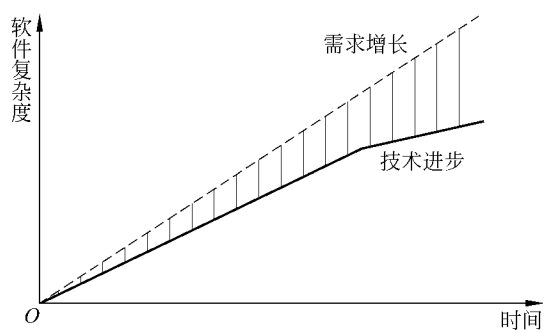

图 1-3 软件技术进步落后于需求的增长

1.2　软件工程的由来

正是由于软件的发展，使计算机应用逐步渗透到社会生活的各个角落，使各行各业都发生很大的变化。这同时也促使人们对软件品种、数量、功能和质量等提出了越来越高的要求。然而，软件的规模越大、越复杂，人们的软件开发能力越显得力不从心。于是，人们开始重视软件开发过程、方法、工具和环境的研究，软件工程应运而生。

1.2.1　软件工程的产生和发展

为了摆脱软件危机造成的困境，北大西洋公约组织（North Atlantic Treaty Organization，简称 NATO）的科学委员会于 1968 年在联邦德国召开的有关研讨会上，首先提出了“软件工程”（Software Engineering）的概念，其主要思路是把人类长期以来从事各种工程项目所积累起来的行之有效的原理、概念、技术和方法，特别是人类从事计算机硬件研究和开发的经验教训，应用到软件的开发和维护中。

软件工程是随着计算机系统的发展而逐步形成的计算机科学领域中的一门学科。软件工程的发展可分为 4 个时期。

1. 20 世纪 40 年代中期到 60 年代中期

这个时期计算机硬件从电子管电子计算机发展到晶体管电子计算机，价格昂贵，运算速度低，存储量小。软件通常是规模较小的程序，软件的设计开发者和使用者往往是同一个人。软件设计通常只注意如何节省存储单元、提高运算速度，除了程序清单之外，没有其他任何文档资料。

2. 20 世纪 60 年代中期到 70 年代中期

这个时期计算机硬件发展到集成电路计算机，运算速度和内存容量都相应提高了。出现了“软件作坊”，许多用户不再自己开发软件，而是去“软件作坊”购买软件。随着计算机应用的日益普及，软件需求量急剧增长。用户的需要和使用环境发生变化时，软件可修改性又很差，往往需要重新编制程序，其研制时间很长，不能及时满足用户要求，质量得不到保证。所谓“软件危机”由此开始。

如 IBM 公司的 OS/360 系统和美国空军后勤系统，在开发过程中都花费了几千人/年的工程量，最后都以失败告终。其中 OS/360 系统由 4000 个模块组成，共约 100 万条指令，花费了 5000 人/年的工程量，经费达数千万美元，结果还是失败了。

1968 年北大西洋公约组织的计算机科学家在联邦德国召开国际会议，正式提出了“软件工程”的术语。从此一门新兴的工程学科诞生了。当时“软件工程”还处于学术研究阶段，但已对软件开发产生了巨大影响。著名的例子，1971 年 IBM 公司运用软件工程技术成功地开发了“纽约时报情报库系统”和“空间实验室飞行模拟系统”，而且软件生产率比以前提高了一倍。

3. 20 世纪 70 年代中期到 80 年代

这个时期硬件发展到大规模集成电路计算机，计算机硬件的功能和质量都不断提高。计算机应用不断地扩大，软件开发生产率提高的速度远远跟不上计算机应用迅速普及深

入的趋势，软件产品供不应求，软件危机日益严重，为了维护软件还要耗费大量的成本。当时美国的统计表明，对计算机软件的投资占计算机软件、硬件总投资的70%，到1985年软件成本大概占总成本的90%。为了对付不断增长的软件危机，软件工程把软件作为一种产品批量生产。软件工程运用工程学的基本原理和方法来组织和管理软件生产，以保证软件产品的质量和提高软件生产率。

4. 20世纪80年代以后

计算机系统发展的第4代不再是单台的计算机和计算机系统，而是计算机软件和硬件的综合效果。由复杂操作系统控制的强大桌面机、局域网和广域网，与先进的应用软件相互配合，计算机体系结构从集中的主机环境转变为分布式的客户机/服务器环境。为此，软件开发的第4代技术可以举例如下：面向对象技术已在许多领域迅速取代了传统的软件开发方法；专家系统和人工智能软件从实验室进入了实际应用，解决了大量的实际问题；人工神经网络软件展示了模式识别与拟人信息处理的前景；并行计算、网络计算机、虚拟现实技术、多媒体技术和现代通信技术使人们可以采用和原来完全不同的方法进行工作。

1.2.2 浅谈软件工程的定义

关于软件工程，目前尚无统一、一致的定义，下面给出几个有代表性的定义。

1. Fritz Bauer在NATO会议上给出的定义

软件工程是建立和使用一套合理的工程原则，以便获得经济的软件，这种软件是可靠的，可以在实际机器上高效地运行。

2. IEEE在软件工程术语汇编中的定义

软件工程：①将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件；②在①中所述方法的研究。

3.《计算机科学技术百科全书》中的定义

软件工程是应用计算机科学、数学及管理科学等原理，开发软件的工程。软件工程借鉴传统工程的原则、方法，以提高质量、降低成本为目的。其中，计算机科学、数学用于构造模型与算法，工程科学用于制定规范、设计范型(paradigm)、评估成本及确定权衡，管理科学用于计划、资源、质量、成本等管理。

1.2.3 理解软件工程的作用

软件工程是软件行业的一门管理科学，也是系统分析员和项目经理以上人员必备的一种领导艺术，为了将我国的软件产业搞上去，使软件产业成为国民经济的支柱产业，使中国早日成为一个软件大国与软件强国，对于软件工程的作用在软件界怎么强调也不过分。

软件工程来自于软件企业，又服务于软件企业，所以讨论它的作用，主要是讨论它在软件企业中的作用。

从历史上讲，软件工程的作用，是为了克服20世纪60年代出现的软件危机(Software Crisis)，这种危机表现为软件开发的成本大、进度慢、维护难和质量得不到保障。

从当前来讲，软件工程的作用就是告诉人们怎样去开发软件和管理软件。具体地讲，它表现在与软件开发和管理有关的人员和过程上。为了说明这个问题，首先来分析软件企业的人才结构，看看这些人员的工作与软件工程有什么关系。

一般来说，软件企业的专业人才由下列几个层次组成。

① 高层管理人员。他们应具备的基本条件：软件专业宏观知识、软件工程管理知识，加上商业与资本运作知识。他们要用软件工程的理论和方法来管理整个公司的软件业务。

② 中层项目经理和软件工程师。他们应具备的基本条件：系统分析知识、系统设计知识，加上项目管理知识。他们要用软件工程的理论和方法，来管理项目组的软件开发。他们的个人奋斗目标是软件管理专家、分析设计专家、开发技术专家。这两部分人员，他们是软件工程的拥有者和实践者。

③ 软件蓝领工人。他们应具备的基本条件：掌握阅读文档的技能、程序设计的技巧，加上软件测试的知识。他们要用软件工程的理论和方法来实现软件项目的软件功能、性能、接口、界面。

④ 软件营销人员。他们应具备的基本条件：营销知识、售前知识，加上软件工程基本知识。他们要用软件工程的基本思路来与客户进行沟通，以赢得客户的信任。

⑤ 软件实施和维护人员。他们应具备的基本条件：软件客户化及安装、运行、维修技术。他们要用软件工程的基本方法来实现软件功能、性能与接口的实施和维护。

⑥ 软件售前人员。他们是软件公司的产品形象代表，其奋斗目标是既要成为某个行业领域的产品专家，又要成为该产品的实现顾问。只有这样，他们才能看懂招标书、写好投标书、讲好投标书。在制作和宣讲投标书的过程中，有许多与软件工程相关的知识和内容，如项目开发方法、开发工具、开发环境、运行环境、管理方法、质量和进度控制方法，只有把这些方法写清讲透，用户才能相信、认可，投标才有成功把握。这些知识和内容离不开软件工程知识的学习和教育。

在以上 6 种人员中，软件工程这门课是前三种人员的必修课。对后三种人员，若想在工作中寻求更大的发展空间，提升自己的知识结构和工作层次，也十分需要掌握软件工程的基本知识。当然，对于不同岗位，知识结构要求有所不同，侧重点也不同。但是，只要在软件行业工作，就会自觉或不自觉地参与软件岗位竞争，就必须重视软件工程，学好软件工程，用好软件工程，不断地将自己的实践经验上升到软件工程的理论与方法，又不断地用软件工程的理论与方法指导自己的实践活动，使自己不断地得到升华和发展，这就是软件工程的作用。

从软件项目团队来讲，软件工程的作用是在规定的时间内，按照规定的成本，完成预期质量目标(软件的功能、性能和接口达到需求报告标准)的软件。

从软件企业本身来讲，软件工程的作用是持续地规范软件开发过程和软件管理过程，不断地优化软件组织的个人素质和集体素质，从而逐渐增强软件企业的市场竞争实力。

从软件发展进程来讲，软件工程的作用是克服软件危机，控制软件进度，节约开发成本，提高软件质量。

由于软件工程的作用越来越大，它的地位也越来越高。以前，软件工程只是作为一

门课或一本书。现在，设立了软件工程专业和软件工程学位，即有软件工程硕士和博士学位。

1.2.4 软件工程研究的内容

软件工程作为一门独立的学科已经有30多年的发展历史，研究的主要内容是软件开发技术和软件管理两方面。在软件开发技术中，主要研究软件开发方法、软件开发过程、软件开发工具和环境。在软件开发管理中，主要是研究软件管理学、软件经济学、软件心理学等。

所有软件开发和维护都是由一系列过程所构成，它将方法和技术、工具和环境、标准和规范等结合起来，以合理、及时地进行软件开发和维护。软件生存周期模型则将软件工程过程有机结合起来，提供一个结构框架，明确主要活动和任务，忽略次要的细节，以便于软件开发和维护的各类人员理解并适应不同的项目，有时也称为过程模型，如瀑布模型。由此看来，过程强调的是具体的活动和任务等，模型则突出表现过程的有机结合。

软件开发和维护方法体现了软件开发和维护的人员看待系统的立场和观点，即方法论意义上的方法。例如，结构化方法认为系统是由一些结构化的功能相互联系、相互作用而构成，面向对象方法则认为系统是由一些对象的相互联系、相互作用而构成。技术则是方法的具体实现，由若干步骤组成，突出“如何做”，有时也不加区别地称之为方法，即技术方法。例如，SA/SD(System Analysis/System Design)方法是一种结构化分析与设计技术，OMT(Object Modeling Technique)方法是一种面向对象分析与设计技术，读者可从上下文来加以区分。除此之外，还有一些与软件开发和维护有关的辅助技术，如软件可靠性分析技术等。

工具为软件开发和维护的方法、技术提供了自动或半自动的软件支持，以提高软件生产效率，如编译程序等。将各种工具结合起来，连同有关的软硬件便形成软件开发和维护环境，其目的是使软件工具支持整个软件生存周期，如北大的青鸟系统等。

作为工程而言，标准化、规范化可以使各种工作有章可循，进而提高生产效率和产品质量。软件工程标准主要有五个层次：国际标准、国家标准、行业标准、企业规范和项目规范。同时，考虑到软件工程产品的特殊性和对社会的影响，加强“软件工程师”职业道德规范的建设和教育也具有特别重要的现实意义。

1.3 软件的开发方法

软件工程学的最终目的是以较少的投资获得质量较高的软件产品。研究软件开发方法的目的是使开发过程“纪律化”，就是寻找一些规范的“求解过程”，使开发工作能够有计划、有步骤地进行。

所谓软件开发方法就是使用定义好的技术(集)及表示符号来组织软件产生过程的方法。一般来说，软件开发方法必须在以下三个方面做出规定：开发步骤(包括每步相应的技术和符号)；软件文档格式；开发方案评价标准。

1.3.1 面向过程的方法

面向过程的方法包括面向过程需求分析、面向过程设计、面向过程编程、面向过程测

试、面向过程维护、面向过程管理。面向过程的方法又称结构化方法，习惯上叫做结构化分析、结构化设计、结构化编程、结构化测试、结构化维护。面向过程的方法也称面向功能的方法，它包括面向功能分析、面向功能设计、面向功能编程、面向功能测试、面向功能维护。这种方法包括：面向结构化数据系统的开发方法(Data Structured Systems Development，简称 DSSD)，面向可维护性和可靠性设计的 Parnas 方法，面向数据结构设计的 Jackson 方法，面向问题设计的嵌入式认证模块(Pluggable Authentication Module，简称 PAM)方法等。这些方法在宏观上都属于面向过程的方法，支持这些方法的是面向过程的结构化编程语言。

面向过程的方法的特点是：程序的执行过程不由用户控制，完全由程序控制。面向过程的方法的优点是简单实用，缺点是维护困难。

面向过程的方法开始于 20 世纪 60 年代，成熟于 70 年代，盛行于 80 年代。该方法的基本特点是，分析设计中强调“自顶向下、逐步求精”，编程实现时强调程序的“单入口和单出口”。这种方法在国内曾经十分流行，大量应用，非常普及。

对于软件行业来说，某一种方法论往往来自于某一类程序设计语言。面向过程的方法来自于 20 世纪 60～70 年代流行的面向过程的程序设计语言。例如，ALGOL，PASCAL，BASIC，FORTRAN，COBOL，C 语言等，这些语言的特点是用“顺序、选择(if-then-else)、循环(do-while 或 do-until)”3 种基本结构来组织程序编制，实现设计目标。

面向过程的方法，在军事上的实时跟踪监控系统中有很好的应用。如我方侦察卫星发射后其飞行轨迹的捕获、测量、跟踪和预报，导弹防御系统中敌方导弹发射后飞行轨迹的捕获、测量、跟踪和预报，其软件系统都是采用面向过程的方法设计和实现的。使用面向过程的方法，系统的执行路径可由系统自动控制，也就是程序自动控制，这是一切自动控制与跟踪系统所必须的。

1.3.2　面向数据的方法

面向数据的方法，也称为面向元数据(Metadata)的方法。元数据是关于数据的数据，组织数据的数据。例如，数据库概念设计中的实体名和属性名、数据库物理设计中的表名和字段名就是元数据。而具体的某一个特定的实例就不是元数据，它们叫做对象或记录，是被元数据组织或统帅的数据。面向数据的方法开始于 20 世纪 80 年代，成熟于 90 年代。20 世纪 80 年代中期，美国学者 James Martin 在《信息系统宣言》中提出了“以数据为中心”的学说，它是这种设计方法的萌芽与起源。20 世纪 90 年代中期，Sybase 和 Oracle 公司的 CASE 工具 Power Designer 和 Designer/2000(以后叫做 Oracle Designer)的出现，宣告这种设计方法已经进入工程化、规范化、自动化和实用化阶段，因为 CASE 工具中隐含了这种方法。概括起来，面向数据方法的要点如下：

① 数据(Data)位于企业信息系统的中心。信息系统就是对数据的输入、处理、传输、查询和输出。

② 只要企业的业务方向和内容不变，企业的元数据就是稳定的，由元数据构成的数据模型(Data Model)也是稳定的。

③ 对元数据的处理方法是可变的。用不变的元数据支持可变的处理方法，即以不变应万变，这就是企业信息系统工程的基本原理。

④ 企业信息系统的核心是数据模型。数据模型包括概念数据模型(Conceptual Data Model,简称 CDM)和物理数据模型 PDM(Physics Data Model)。数据模型的表示形式是 E-R 图,E-R 图要用 CASE 工具设计。例如,Power Designer,Oracle Designer 或 ERwin,它们不但具有正向设计功能,而且具有逆向分析功能,这样才能实现快速原型法。

⑤ 信息系统的实现(编码)方法主要是面向对象,其次才是面向数据和面向过程。

⑥ 用户自始至终参与信息系统的分析、设计、实现与维护。

面向数据方法的特点是程序的执行过程中,根据数据流动和处理的需要,有时由程序员控制(如数据库服务器上触发器和存储过程的执行),有时由用户控制(如用户浏览层上控件的选择与执行)。

面向数据方法的优点是通俗易懂,特别适合信息系统中数据层(数据库服务器)上的设计与实现;缺点是实现窗口界面较困难。

面向数据的方法来自于 20 世纪 80 年代开始流行的关系数据库管理系统 RDBMS,以及关系数据库程序设计语言,例如,Oracle,Sybase 关系数据库语言,这种关系数据库语言或命令提供了强大的面向关系表中数据的编程能力,典型的例子就是编写存储过程和触发器。Oracle 数据库管理系统自带的编程工具 Developer 2000,首先是一个面向数据的编程工具,其次才是一个面向对象的编程工具。Oracle Designer 加上 Developer 2000,构成了一个完整的面向数据的信息系统开发环境。

面向数据的方法与关系数据库管理系统紧密地捆绑在一起,只要面向对象数据库不能完全替代关系数据库,这种方法就不会终结。目前数据库管理系统的发展趋势是在关系型数据库的基础上,将面向对象的某些特性(如继承)添加上去,称为"对象-关系型数据库",但本质上仍然是一个关系型数据库。正如美国数据库专家 David M. Kroenke 所说的,"面向对象这样的数据库只是概念上的兴趣,它们在商用数据库处理中只起很小的作用。"

面向数据的方法在电子商务中应用有网站后台数据库服务器上的数据处理和数据传输,其软件都是利用面向数据的方法设计与实现的。实际上,不管网络应用系统结构是两层结构或三层结构,在数据库服务器上对数据的分析、设计和实现都自觉或不自觉地使用了面向数据的方法。

1.3.3 面向对象的方法

面向对象的方法起源于面向对象编程语言。自 20 世纪 80 年代中期到 90 年代,面向对象的研究重点已从编程语言转移到设计方法学上来,其中代表性的工作有:

① B. Henderson-sellers 和 J. M. Edwards 提出的面向对象软件生存周期的喷泉模型,以及面向对象系统开发方法。

② G. Booch 提出面向对象开发方法等。

③ P. Coad 和 E. Yourdon 提出面向对象分析(Object Oriented Analysis,简称 OOA)和面向对象设计(Object Oriented Design,简称 OOD);

④ J. Rumbaugh 等人提出的对象建模技术(Object Modeling Technique,简称 OMT);

⑤ Jacobson 提出的面向对象软件工程(Object Oriented Software Engineering,简称

OOSE)；

⑥ 由 G. Booch，J. Rumbaugh 和 Jacobson 等人发起，在 Booch 方法、OMT 方法加 OOSE 方法的基础上推出了统一的建模语言(Unified Modeling Language，简称 UML)。

面向对象的方法包括分析、设计和实现活动。它是一种把面向对象的思想运用于软件开发过程，指导开发活动的系统方法，建立在“对象”概念(对象、类和继承)基础上的方法学基于对象概念，以对象为中心，以类和继承为构造机制来认识、理解、刻画客观世界和设计、构建相应的软件系统。面向对象方法的应用目前有两种方式。

① 在分析、设计、实现活动中完全采用面向对象的技术。

② 传统的功能分解方法与面向对象方法结合使用。即功能分析、面向对象的设计和实现，以及面向对象分析和设计、实现用过程式语言。

面向对象分析的任务就是通过分析问题域建立系统的概念模型，并用相应的符号系统表示。而模型一般由五个层次构成，即主题层、类及对象层、结构层、属性层和服务层。因此，其步骤也是按其五个层次逐步展开。

面向对象设计是在面向对象分析的基础上进行系统设计，包括交互过程和用户接口、任务管理、全局资源协调并确定边界、各个类的存储和数据格式。

面向对象实现就是用面向对象程序设计语言来实现面向对象设计，因为该类语言支持对象、运行多态性和继承，因此比较容易，如果使用非面向对象程序设计语言，则需要特别注意和规定保留程序的面向对象式的程序结构。

用面向对象方法开发的软件，其结构基于客观世界界定的对象结构，因此与传统的软件相比较，软件本身的内容结构发生了质的变化，因而易复用性和易扩充性都得到了提高，而且能支持需求的变化。

作为软件开发方法，还应该叙述软件复用技术。且当今国际上已十分重视软件复用，提出了构件、体系结构和框架等一系列方法。这些将在本书的其他章中介绍。

面向对象的方法在电子商务中也有应用，网站前台界面的制作、信息的发布和处理、用户在网上浏览和录入信息等应用软件都是利用面向对象的方法设计与实现的。个人网页的制作也是面向对象方法的应用例子。窗口操作系统与互联网的出现，为面向对象方法开辟了无限的前景。

1.4　软件生存周期

如同人的一生，软件也有一个孕育、诞生、成长、衰亡的生存过程，这个过程称为软件的生存周期。

软件生存周期是指软件产品或软件系统从产生、投入使用到被淘汰的全过程。软件生存周期大致可以分为 6 个阶段：计算机系统工程、需求分析、设计、编码、测试、运行和维护。

1. 计算机系统工程

计算机系统包括计算机硬件、软件，以及使用计算机系统的人、数据库、文档、规程等系统元素。计算机系统工程的任务是确定待开发软件的总体要求和范围，以及该软件与

其他计算机系统元素之间的关系，进行成本估算，作出进度安排，并进行可行性分析。即从经济、技术、法律等方面分析待开发的软件是否有可行的解决方案，并在若干个可行的解决方案中作出选择。

2. 需求分析

需求分析主要解决待开发软件要“做什么”的问题，确定软件的功能、性能、数据、界面等要求，生成软件需求规约(也称软件需求规格说明)。

3. 设计

软件设计主要解决待开发软件“怎么做”的问题。软件设计通常可分为系统设计(也称概要设计或总体设计)和详细设计。系统设计的任务是设计软件系统的体系结构，包括软件系统的组成成分、各成分的功能和接口、成分间的连接和通信，同时设计全局数据结构。详细设计的任务是设计各个组成成分的实现细节，包括局部数据结构和算法等。

4. 编码

编码阶段的任务是用某种程序设计语言，将设计的结果转换为可执行的程序代码。

5. 测试

测试阶段的任务是发现并纠正软件中的错误和缺陷。测试主要包括单元测试、集成测试、确认测试和系统测试。

6. 运行和维护

软件完成各种测试后就可交付使用，在软件运行期间，需对投入运行的软件进行维护，即当发现了软件中潜藏的错误或需要增加新的功能或使软件适应外界环境的变化等情况出现时，对软件进行修改。

1.5 了解软件开发模型

根据软件生产工程化的需要，生存周期的划分也有所不同，从而形成了不同软件生存周期模型(soft ware life cycle model)，或称软件开发模型。软件开发模型可定义为：它是软件开发全部过程、活动和任务的结构框架。软件开发模型能清晰、直观地表达软件全过程，明确规定了要完成的主要活动和任务，用来作为软件项目开发工作的基础。对于不同的软件系统，采用不同的开发方法、使用不同的程序设计语言以及各种不同技能的人员参与工作、不同的管理方法和手段等。它还应允许采用不同的软件工具和不同的软件工程环境，软件开发模型都应该是稳定有效和普遍适用的。

在软件工程中，重要的概念是软件开发模型，以及软件生存周期过程，后者可分为基本生存周期过程、支持生存周期过程和组织生存周期过程，而基本生存周期过程又分为获取过程、供应过程、开发过程、运作过程和维护过程。需要注意的是开发模型仅对开发、运作、维护过程有意义。在此还需补充一点，在 ISO 12207 和 ISO 9000—3 中都提到软件生存周期模型，它和软件开发模型是同一概念，生存周期模型是一个框架，它规定了软件开发、运作和维护中所需的过程、活动和任务。

1.5.1　瀑布模型

瀑布模型遵循软件生存周期的划分，明确规定每个阶段的任务，各个阶段的工作顺序展开恰如奔流不息拾级而下的瀑布。

瀑布模型把软件生存周期分为计划、开发、运行 3 个时期。这 3 个时期又可细分为若干个阶段：计划时期可分为问题定义、可行性研究两个阶段，开发时期分为需求分析、概要设计、详细设计、程序设计、软件测试等阶段，运行时期则边运行边维护。

瀑布模型如图 1-4 所示。瀑布模型软件开发有以下几个特点。

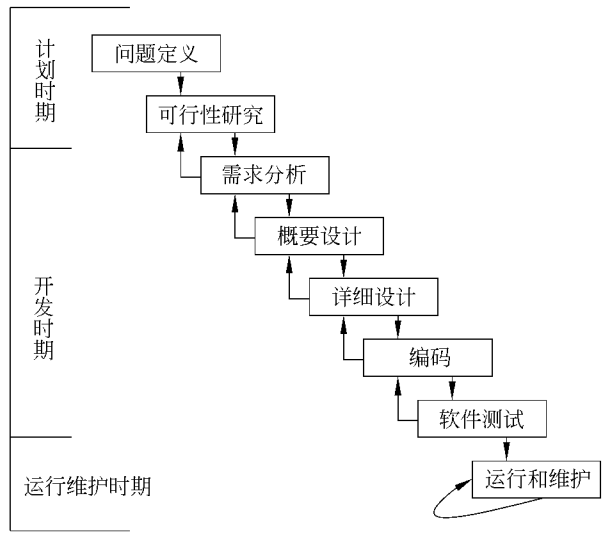

图 1-4　瀑布模型

(1) 软件生存周期的顺序性

顺序性是当只有前一阶段工作完成，后一阶段的工作才能开始；前一阶段的输出文档，就是后一阶段的输入文档。只有前一阶段有正确的输出，后一阶段才可能有正确的结果。如果在生存周期的某一阶段出现了错误，往往要追溯到在它之前的一些阶段。

瀑布模型开发适合于在软件需求比较明确、开发技术比较成熟、工程管理比较严格的场合下使用。

(2) 尽可能推迟软件的编码

程序设计也称为编码。实践表明，大、中型软件编码开始得越早，完成所需的时间反而越长。瀑布模型在编码之前安排了需求分析、概要设计、详细设计等阶段，从而把逻辑设计和编码清楚地划分开来，尽可能推迟程序编码阶段。

(3) 保证质量

为了保证质量，瀑布模型软件开发在每个阶段都要完成规定的文档，每个阶段都要对已完成的文档进行复审，以便及早发现隐患，排除故障。

1.5.2　快速原型模型

所谓快速原型是快速建立起来的可以在计算机上运行的程序，它所能完成的功能往

往是最终产品能完成的功能的一个子集(展示了目标系统的关键功能)。图 1-5 所示是软件开发的快速原型模型的示意图。由于快速原型方法要求能够快速地建立可以供用户使用的原型并从用户快速获取反馈,因此,快速原型模型不可能像最终产品一样面面俱到。例如,通常它不会对输入数据的合法性进行检查,也不会过多关注性能问题。

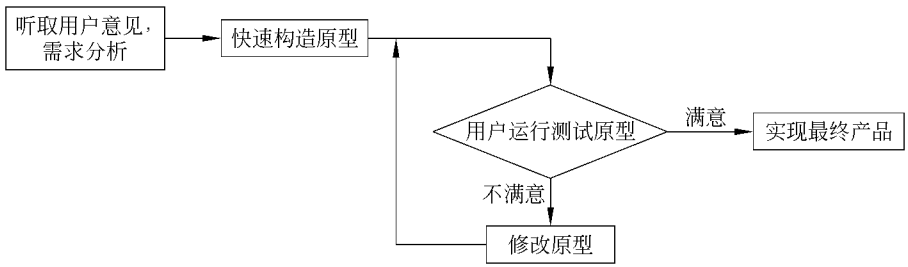

图 1-5 快速原型模型

快速原型模型的第一步是建造一个快速原型,实现客户或未来的用户与系统的交互,用户或客户对原型进行评价,进一步细化待开发软件的需求。通过逐步调整原型使其满足客户的要求,开发人员可以确定客户的真正需求是什么。第二步则在第一步的基础上开发客户满意的软件产品。

显然,快速原型方法可以克服瀑布模型的缺点,减少由于软件需求不明确带来的开发风险,具有显著的效果。

快速原型的本质是“快速”。开发人员应该尽可能快地建造出原型系统,以加速软件开发过程,节约软件开发成本。原型的用途是获知用户的真正需求。一旦确定了客户的真正需求,所建造的原型将被丢弃。因此,原型系统的内部结构并不重要,重要的是必须迅速建立原型,随之迅速修改原型,以反映客户的需求。

快速原型模型是不带反馈环的,这正是这种过程模型的主要优点:软件产品的开发基本上是线性顺序进行的。能做到基本上线性顺序开发的主要原因如下:

① 原型系统已经通过与用户交互而得到验证,据此产生的规格说明文档正确地描述了用户需求,因此,在开发过程的后续阶段不会因为发现了规格说明文档的错误而进行较大的返工。

② 开发人员通过建立原型系统已经学到了许多(至少知道了“系统不应该做什么,以及怎样不去做不该做的事情”)。因此,在设计和编码阶段发生错误的可能性也比较小,这自然减少了在后续阶段需要改正前面阶段所犯错误的可能性。

软件产品一旦交付给用户使用之后,维护就开始了。根据所需完成的维护工作种类的不同,可能需要返回到需求分析、规格说明、设计或编码等不同阶段。

当快速原型的某个部分是利用软件工具由计算机自动生成的时候,可以把这部分用到最终的软件产品中。例如,用户界面通常是快速原型的一个关键部分,当使用屏幕生成程序和报表生成程序自动生成用户界面时,实际上可以把得到的用户界面用在最终的软件产品中。

1.5.3　渐增模型

渐增式模型亦称为递增式或增量式模型。与建造大厦相同，软件也是一步一步建造起来的。在渐增模型中，软件被作为一系列的增量构件来设计、实现、集成和测试，每一个构件是由多种相互作用的模块所形成的提供特定功能的代码片段构成，如图 1-6 所示。

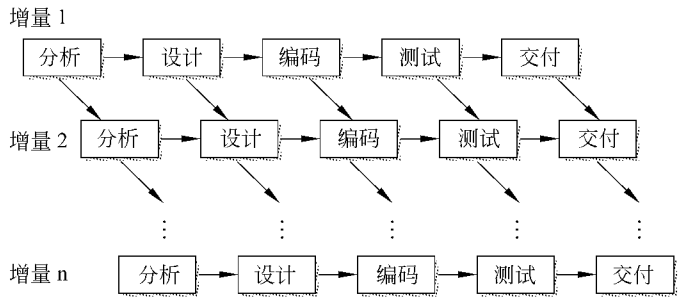

图 1-6　渐增模型

渐增模型在各个阶段并不交付一个可运行的完整产品，而是交付满足客户需求的一个子集的可运行产品。整个产品被分解成若干个构件，开发人员逐个构件地交付产品，这样做的好处是软件开发可以较好地适应变化，客户可以不断地看到所开发的软件，从而降低开发风险。但是，渐增模型也存在以下缺陷。

① 由于各个构件是逐渐并入已有的软件体系结构中的，所以加入构件必须不破坏已构造好的系统部分，这需要软件具备开放式的体系结构。

② 在开发过程中，需求的变化是不可避免的。渐增模型的灵活性可以使其适应这种变化的能力大大优于瀑布模型和快速原型模型，但也很容易退化为边做边改模型，从而使软件过程的控制失去整体性。

在使用渐增模型时，第一个增量往往是实现基本需求的核心产品。核心产品交付用户使用后，经过评价形成下一个增量的开发计划，它包括对核心产品的修改和一些新功能的发布。这个过程在每个增量发布后不断重复，直到产生最终的完善产品。

例如，使用渐增模型开发字处理软件。可以考虑，第一个增量发布基本的文件管理、编辑和文档生成功能，第二个增量发布更加完善的编辑和文档生成功能，第三个增量实现拼写和文法检查功能，第四个增量完成高级的页面布局功能。

1.5.4　螺旋模型

瀑布模型要求在软件开发的初期就完全确定软件的需求，这在很多情况下往往是做不到的。螺旋模型试图克服瀑布模型的这一不足。

螺旋模型通常用来指导大型软件项目的开发，它将开发划分为制订计划、风险分析、实施开发和客户评估四类活动。沿着螺旋线每转一圈，表示开发出一个更完善的新的软件版本。如果开发风险过大，开发机构和客户无法接受，项目有可能就此终止；多数情况下，会沿着螺旋线继续下去，自内向外逐步延伸，最终得到满意的软件产品。

图 1-7 显示了螺旋模型的原理，沿着螺旋线旋转，在笛卡儿坐标的四个象限上分别表示了四类活动。

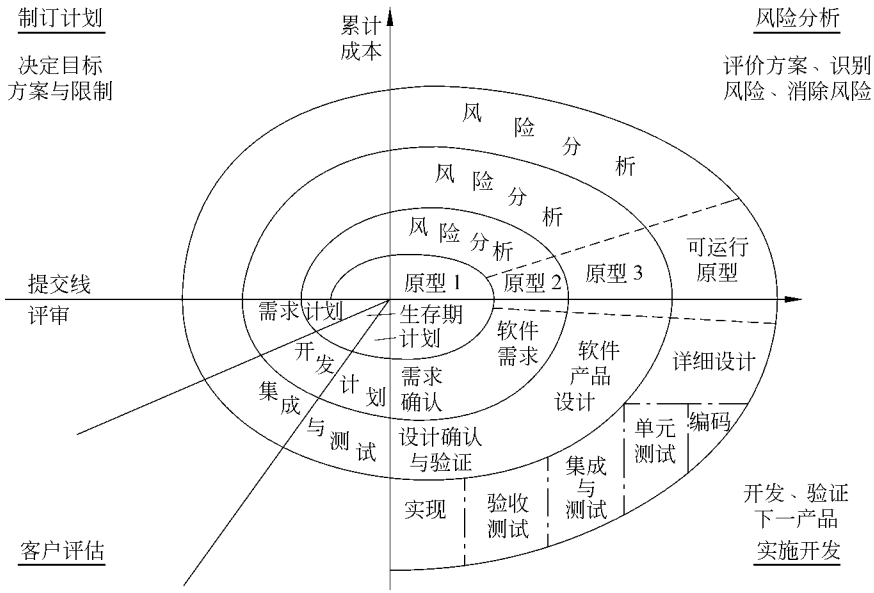

图 1-7 螺旋模型

① 制订计划。确定软件目标，选定实施方案，弄清项目开发的限制条件。

② 风险分析。分析所选方案，考虑如何识别和消除风险。

③ 实施开发。实施软件开发。

④ 客户评估。评价软件功能和性能，提出修正建议。

图 1-8 给出了螺旋模型的另一图示。

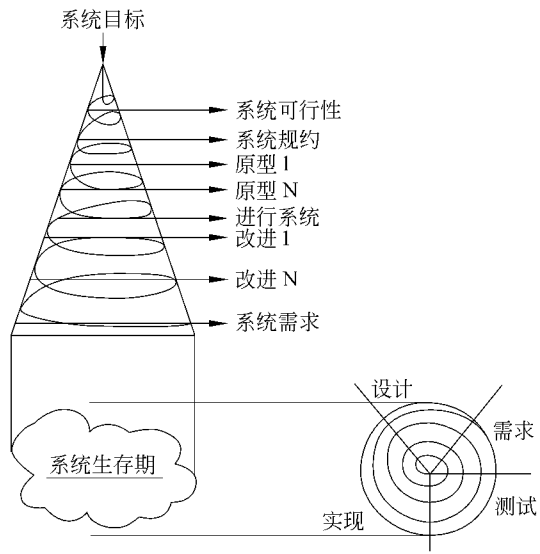

图 1-8 螺旋模型的另一图示

螺旋模型的优越性较为明显，但要求许多客户接受和相信并不容易，使用该模型需要具有相当丰富的风险评估经验和专门知识，如果项目风险较大，又未必能及时发现，势必造成重大损失。目前国内许多软件公司还未能及时掌握和运用，有待进一步积累经验。

软件风险是任何软件开发项目中普遍存在的问题，只是风险大小而已。系统分析人员必须回答：需求是什么？要投入多少资源？如何安排进度等一系列问题后才能制定计划，所以判断的关键是分析人员的经验。项目越大问题越复杂，资源、进度、成本等因素不确定性越大。风险分析就在于风险识别、采取对策，从而消除或减少所可能造成的损失。

1.5.5　快速应用开发模型

快速应用开发(Rap Application Development，简称 RAD)是一个线性顺序的软件开发模型，强调极短的开发周期。RAD 模型是线性顺序模型的一个“高速”变种，通过使用基于构件的建造方法获得了快速开发。如果需求理解得很好，且约束了项目范围，RAD 过程使得一个开发组能够在很短时间内(如 60～90 天)创建出“功能完善的系统”。RAD 方法主要用于信息系统应用软件的开发，它包含如下几个开发阶段。

① 业务建模。业务活动中的信息流被模型化，以回答如下问题：什么信息驱动业务流程？生成什么信息？谁生成该信息？该信息流往何处？谁处理它？

② 数据建模。业务建模阶段定义的一部分信息流被精化，形成一组支持该业务所需的数据对象。标识出每个对象的特征(称为属性)，并定义这些对象间的关系。

③ 处理建模。数据建模阶段定义的数据对象变换成为要完成一个业务功能所需的信息流。创建处理描述以便增加、修改、删除或获取某个数据对象。

④ 应用生成。RAD 假设使用第四代技术。RAD 过程不是采用传统的第三代程序设计语言来创建软件，而是复用已有的程序构件(如果可能)或是创建可复用的构件(如果需要的话)。在所有情况下，均使用自动化工具辅助软件建造。

⑤ 测试及反复。因为 RAD 过程强调复用，许多程序构件已经是测试过的，这减少了测试时间。但新构件必须测试，所有接口也必须测试。

1.6　小结

软件已经成了基于计算机系统和产品的一个关键元素。在过去的几十年中，软件已经由早期的求解专门问题的程序和信息分析的工具发展成为现今的一个产业(即软件产业)。但是，早期“编程”的缺点和历史导致了延续至今的一系列问题，软件变成了基于计算机系统和产品进一步发展的限制因素。软件工程的目的就是要提供一个构造高质量软件的框架，以解决这些问题。

软件工程是一门集过程和模型、方法和技术、工具和环境、标准和规范于一体进行计算机软件开发的工程学科。人们已经提出了许多不同的软件开发模型，每一种都有它的优点和缺点，实际应用时应该引起重视。方法和技术、工具和环境是软件工程师的必备知识。标准和规范对于确保软件质量是至关重要的。

1.7 习题

1. 什么是软件危机？为什么会出现软件危机？

2. 什么是软件工程？怎样运用软件工程消除或缓解软件危机？

3. 某班期末考试有以下课程：高等数学、英语、计算机应用、思想道德修养、C 语言。若要编制一个计算机软件，输入学生的学号、姓名、5 门课的成绩，打印出全班成绩单及每个学生的成绩单。你准备采用哪种方法开发？

4. 什么是软件生存周期？软件生存周期可以划分为哪几个阶段？

5. 比较几种软件开发模型的特点。

6. 哪些开发模型适用于面向对象的软件开发？

第 2 章

软件需求分析

本章要点

- 需求分析的任务；
- 需求分析的步骤；
- 需求的各种描述工具；
- 需求分析的文档。

学完本章之后你将能够

- 理解需求分析的任务及步骤；
- 掌握需求的各种描述工具：实体—关系图、数据流图、数据字典及一些图形工具；
- 掌握软件需求规格说明书的内容；
- 掌握用户手册的内容。

2.1 需求分析的任务

需求分析是研究用户要求，以得到目标系统的需求定义的过程。

需求分析的基本任务是软件开发人员和用户一起完全弄清用户对系统的确切要求。需求分析的结果是否正确，关系到软件开发的成败。正确的需求分析是整个系统开发的基础。需求分析是理解、分析和表达“系统必须做什么”的过程。

其中，理解就是尽可能准确地了解用户当前的情况和需要解决的问题。需求分析阶段并不马上进行具体的系统设计和需求实现，而是对用户提出的要求反复多次地细化，才能充分理解用户的需求。通过分析得出对系统完整、准确、清晰、具体的要求。表达是通过建模、规格说明和复审，说明“系统必须做什么”的过程。

建立模型就是描述用户需求，可使用的工具有实体—关系图、数据流图、数据字典、层次图、Warnier 图、IPO 图等。

下面介绍需求分析阶段的具体任务。

1. 确定目标系统的具体要求

需求分析阶段要确定目标系统的具体要求。

(1) 确定系统的运行环境要求

系统运行时的硬件环境要求,如外存储器种类、数据输入方式、数据通信接口等;软件环境要求,如操作系统、汉字系统、数据库管理系统等。

(2) 系统的性能要求

系统性能要求,如系统所需的存储容量、安全性、可靠性、期望的响应时间(即从终端输入数据到系统后,系统在多长时间内可以有反应,这对于实时系统来讲是关系到系统能否被用户接受的问题)。

(3) 系统功能

确定目标系统必须具备的所有功能,系统功能的限制条件和设计约束。

(4) 接口需求

接口需求描述系统与其环境的通信的格式。常见的接口需求有用户接口需求、硬件接口需求、软件接口需求、通信接口需求等。

2. 建立目标系统的逻辑模型

需求分析实际上就是建立系统模型的活动。

模型是为了理解事物而对事物做出一种抽象、无歧义的书面描述。模型由一组图形符号和组成图形的规则组成。建模的基本目标如下:

- 描述用户需求;
- 为软件的设计奠定基础;
- 定义一组需求,用以验收软件产品。

模型分为数据模型、功能模型和行为模型。为理解和表示问题的信息域,建立数据模型;为了定义软件的功能,建立功能模型;为了表示软件的行为,建立行为模型。可用层次的方式来细分数据模型、功能模型和行为模型,在分析过程中得出软件实现的具体细节。

为了达到上述目标,可以有三种不同的图形以及数据字典进行描述。数据字典用来描述软件使用或产生的所有实体。数据模型用实体—关系图来描述实体之间的关系。功能模型用数据流图来描述,其作用如下:

- 数据在系统中移动时如何变换;
- 描绘变换数据流动的功能和子功能。

行为模型可用状态转换图来描绘系统的各种行为模式(状态)和不同状态间的转换。

2.2 需求分析的步骤

在前一节中提出了需求分析的任务,但如何完成这些任务一个复杂系统的分析工作从何入手?传统的软件工程方法学采用结构化分析方法完成需求分析工作。

需求分析的步骤:进行调查研究,分析和描述系统的逻辑模型,复审。

1. 进行调查研究

对于不同的软件开发方法,在进行需求分析时具体步骤会有所不同。但有一点是相同的,需求分析阶段要作充分的调查研究。系统分析员要分析来自用户的各种信息,与用