第1章

软件工程概述

自1946年世界上第一台电子计算机诞生以来，计算机科学技术本身及其在各个领域的应用得到了飞速的发展。从计算机发展的历程来看，计算机技术是随着硬件的产生而形成的独立的技术学科。随后将用于算法的程序从硬件中分离出来而逐渐有了软件技术的概念。从此,计算机硬件技术与软件技术相互促进、共同发展,极大地促进了计算机在社会、经济、教育等领域的应用。特别是自60年代以后,计算机的应用几乎涉及到了社会和生活的各个方面。在所有的计算机应用领域，软件已经成为基于计算机系统的重要因素，同时对计算机功能的有效发挥起着举足轻重的作用。这不仅促使计算机硬件技术进一步发展，同时也使计算机软件的开发向着产业化方向迅速发展壮大起来。

1.1 软件的概念、特点以及分类

1.1.1 软件及其特征

从计算机应用的角度来看，计算机软件是计算机在各个方面应用的基础。计算机软件不仅包含计算机可以识别的代码形式，而且还包含每个项目内部构件的所有文件。计算机软件包含技术要求文件、设计文件、各种法律和财务上的文件、软件项目管理计划以及其他的管理文本和所有形式的手册。

计算机软件从其特征上来看，有以下主要特点：

1. 软件是逻辑意义上的概念而不是有形的系统中的部件。从这个意义上来说，软件是具体化的知识,软件的生产是一个知识化的脑力劳动过程,而不是一般的产品生产制造过程；

2. 软件的退化和失效不是因为使用过程中的消耗和老化而是由软件的自身缺陷以及维护修改不当导致的。因此,软件的维护与硬件的维修有着本质的区别,它是一个使软件持续发挥其效能的更复杂、更重要的过程。主要方面是为适应对象所做的软件改进和软件开发中存在缺陷的去除而不是因部件损坏或失效使其恢复性能的一般修理。从软件的应用特征来看，软件的维护在整个软件的生命周期中起着举足轻重的作用；

3. 由于软件不能完全摆脱硬件和软件环境而单独发挥作用，因此软件的开发必须考

虑计算机系统所能提供的基础条件，软件开发和运行对计算机系统有一定的依赖性；

4. 另外，由于软件只有与实际应用紧密相结合才能发挥出其独特的效力，所以软件一般都是由定制的专门化生产来满足需求。从这个意义上讲，一个软件的应用成功与否取决于软件能否最大限度地满足对象的使用要求；

5. 软件、特别是大型软件的开发需要投入大量的、复杂的和高强度的脑力劳动。这导致了软件的高成本特征。如今，软件开发的费用已大大超过硬件的开销。另外由于软件开发技术本身的局限性和其他各方面因素的影响，大量软件开发资金投入后并不能保证都能获得预期的成果和回报。据统计，美国军方每年花费数十亿美元来购买软件，其中可直接使用的仅占百分之二，另外有百分之三要做一些修改才能投入使用，剩下的百分之九十五由于基本不能满足使用要求而失去了使用价值。由此可看出，软件开发具有高风险的特征；

6. 相对计算机硬件技术的发展，计算机软件技术发展十分缓慢，不论是在理论研究中还是在实际开发的普及上都与计算机硬件的发展有相当的距离。面对计算机硬件的飞速发展，软件已经成为制约计算机技术发展和应用的瓶颈。

计算机软件技术的发展经历了程序设计（Program Design）时期（1947 年～60 年代初）、程序系统（Program System）时期（50 年代末～70 年代初）和软件工程（Software Engineering）时期（70 年代初至今）等三个阶段。其发展主线是由个体简单的开发方式向着复杂、大规模、标准化、工程化的方向发展。在其最高阶段，把软件的开发界定为两个方面的内容：软件开发和随软件应用要求而带来的软件维护。

1.1.2 软件的分类

计算机软件是一个涉及多个领域、应用广泛的概念。从各个不同的角度人们对计算机软件提出了许多分类的方法。目前一般的分类方法有 按软件的功能进行划分 按软件的规模进行划分 按软件工作方式进行划分 按软件服务对象的范围进行划分 按使用频度进行划分；按软件失效影响进行划分。

从某种程度上说，要兼顾多个不同类型的对象和范围对计算机软件给出一个科学、通用的分类确实是一件不容易的事情。因为不管是从哪个角度讲，随着软件复杂性的增加和应用领域及对象的多样性，各种分类方法都摆脱不了分类间的相互渗透，难以找出它们之间一般的差异性。因此，从应用领域的角度，可以简单地把软件做以下归类。

1. 系统软件：与计算机硬件紧密结合，构成用户在某一方面使用计算机的基础平台。它的功能既包括复杂信息的数据结构处理，也包括计算机资源的共享与复杂的进程管理。不管哪种情况，系统软件的工作通常都伴随着与计算机硬件的频繁交互，需要精细调度。它同时又具有良好的用户支持、资源共享及多外部接口的特征，如操作系统、数据库管理系统、设备驱动程序等。这些软件在某种程度上具有较大范围的适应性，一般由专业的软件公司有目的的开发并较好地维护。

2. 实时软件：计算机的高速处理能力使得应用计算机对事件和数据进行实时处理成为可能，例如，工业过程控制、卫星导弹的运行控制、管理信息处理等等。这些完成处理、反馈、控制过程的软件称为实时软件。这些软件的特征是，对事件的响应时间有严格限

定。它主要包括数据收集、实时分析和控制输出三个部分。实时软件既可以应用于信息处理也广泛应用于过程控制。

3. 嵌入式软件：随着智能化产品的不断出现，微型处理器(MCU)在消费产品和工业产品中的应用越来越普及。这些用于提供控制和专职功能的软件称之为嵌入式软件。嵌入式软件一般为某一单独的应用专门设计,驻留在只读内存中,执行有限的专职功能(如洗衣机、电冰箱等简单器具的操作及控制)或提供重要的功能及复杂的控制能力(如空调器的控制、数控机床等控制系统)另外还可以用来进行信息处理(如电子词典、PDA 等)。

4. 基于 Web 的软件：互联网的普及给计算机的应用提供了更广泛的空间,Internet 提供了无所不包、使用方便的软件资源。为了浏览和检索,基于 Web 的可执行指令(如 CGI、HTML、Perl 或 Java)以及数据(如超文本以及可视、音频等格式的数据)应用于浏览器的软件成为当今的又一个热点。

5. 实用软件：这些软件是针对计算机在某一领域或特定工作性质中应用的具有一定通用性质的软件。它通常可分为应用软件和支撑软件两大类，应用软件是在操作系统的基础上为某一特定领域应用而开发的软件,例如商业处理软件、科学计算软件、计算机辅助设计软件、人工智能软件等等；支撑软件一般用来辅助和支持开发人员开发和维护软件,例如需求分析工具、设计工具、编码工具、测试工具、维护和管理工具等等。

1.2 软件开发引发的问题

1.2.1 社会对软件需求的不断发展

随着社会的发展和科技的进步，人们需要解决的问题愈加复杂、有更高的实时性要求。面对上述问题若单靠人工解决,不光是成本高而且要耗费大量的时间,有时甚至是等到了问题的解决却由于时间过长而失去了意义。面对种种单靠人类自身能力难以解决的问题,人们使用计算机成功地达到了预期的目的。在计算机技术的应用中,针对多样化问题的解决,需要大量不同软件的支持。因为离开了软件,计算机不能解决任何问题。所以作为当今计算机信息技术灵魂的软件,随着计算机应用的日益普及和深化,它的数量正以惊人的速度急剧膨胀，复杂程度和规模也在迅速地增加。

1.2.2 软件开发理念和手段与客观实际存在的差异

随着计算机技术的普及和应用领域的日益广泛，人们对软件开发手段的掌握和技术内涵的理解也逐步深入。30 多年来，人们在软件开发的理念、方法及管理方面取得了长足的进步。但是,由于这些进步是一个逐步渐进的过程,虽然每当有新的思想和方法出现时都会对传统的方法和手段产生较大的变革，但是长期以来由于计算机及其软件技术发展的阶段性和种种制约，人们总是凭着对计算机技术的片面理解和对实际问题的不充分的认识进行软件开发。应用的开发理念和手段与客观实际的情况存在较大的差异，这些差异的存在对软件开发的各个环节形成了较大的障碍。目前，存在的差异主要表现在以

下几个方面。

1. 对软件本身的认识存在差异：

在计算机产生初期,计算机软件被认为是独立的程序,认识的角度是独立的、个体的。那时认为软件开发工作的主要任务就是编程，编程是程序设计人员个人技巧的充分发挥，工作是独立完成的而且是无章可循的。没有把个人的活动和整个软件开发活动结合起来,片面地夸大了个人的作用。整个软件的开发过程缺乏系统的理论和方法,对软件产品本身的认识也等同于一般的产品,把重点放到生产制造环节,缺乏维护的思想。

2. 对软件的服务对象认识不足：

在人们的传统认识中,一直把软件开发的重点放在开发者本身上,而对使用者一方没有给予足够的重视。随着软件的日益复杂和规模的不断扩大，软件的专业性也随之增强。这时软件开发人员对实际问题的理解和认识就成了软件开发成败的关键所在。现代软件工程要求对使用者提出的问题给予充分的认识和定义，并且进一步在软件开发中加以完善。

3. 对软件的开发缺乏科学的管理：

现代软件开发过程是一个复杂的系统工程，具有明显的工程特征和较高的技术含量。由于软件开发过程能见度底低，管理人员对开发人员的工作和进度都难以把握，所以现有的管理手段和方法也难以适应软件开发的要求。例如，用一般工作量的衡量方法，比如以人时、人月或人 年为计量单位的方法(简称人时、人月或人年)就难以准确地把握软件开发的工作进度。因为在软件的开发过程中，任务是不能简单分解的。分解任务后，人员的增加会带来沟通和交流的工作量增加，从而使总的工作量进一步增加。因此人员的数量和时间是不能简单替换的,他们的关系见图 1.1。

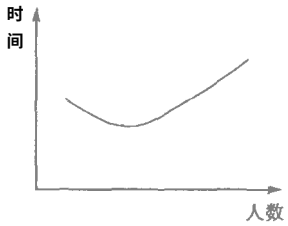

图1.1

4. 软件开发手段和工具的不完善：

在人们长期的劳动实践中可以看出，每一次先进工具的引入都会极大地提高生产效率。这一点在软件开发的过程中也不例外。虽然在长期的软件开发过程中人们一直在不断地探索、研究和制造有效实用的开发工具,但是,由于软件的开发工程是人类智慧的运用过程,所面临的对象又是复杂多样的,所以虽然目前出现了一些开发工具,但是都只能应用在软件开发的某一方面。尤其是在软件的分析、设计阶段,更无有效的工具。这一方面的工作还是主要由人工完成。

1.2.3 由软件开发导致的“软件危机”

软件危机是指在计算机软件开发和维护过程中遇到的一系列严重问题。这些问题主要体现在如何开发软件以满足用户日益增长的需求和如何对已有的软件进行维护。

对于这一系列源于软件的开发理念、方法与客观现实不适应的严重问题,人们在软件的开发过程中不是彻底失败了就是虽然软件开发出来了但是运行结果不甚理想，有的虽然按预定的功能要求完成了但是在工期或者成本上却大大超出了预算。综合来看，能真正全面满足用户要求的软件开发微乎其微。众所周知的 IBM 公司的 OS/360 系统花费了

几千人年的努力,耗资数亿美元,历尽艰辛,但结果令人大失所望。

软件开发的高成本与软件产品低质量之间的尖锐矛盾是导致软件危机的主要原因。它们的主要表现和产生原因可以归结为以下几个方面。

1. 软件开发成本和进度难以满足要求

在复杂多样的软件开发对象面前，没有系统的开发手段和方法。开发具有较大的盲目性，从而导致不是经费的大大超支就是工作完成期限一拖再拖。这不仅损害了软件开发者的信誉，也极大地损害了委托开发者的利益。

2. 软件系统不能符合用户的要求

在软件开发的初期,软件设计的依据是根据用户的需求提出的。由于种种原因,这些需求存在着用户表达的局限性和开发者理解的差异性。若在软件开发的过程中软件开发人员和用户没有及时沟通和不断地修正，则势必造成开发的软件与用户的需求产生巨大的分歧，从而给双方带来巨大的损失。

3. 软件难以维护

在传统的软件开发过程中,由于轻视软件开发的管理和缺乏统一的要求,导致软件设计人员存在较大的自由倾向性，加大了软件对设计人员的依赖性。在开发过程中对设计和实现过程的资料收集不够重视,开发人员间的接口部分不规范,这些都给软件的维护工作带来了极大的困难甚至使维护工作不可行。

4. 软件工作不可靠

在软件开发完成后由于种种原因不能保证软件的正确性。软件测试是保证软件正常工作的重要环节。但是由于技术和其他方面的原因，软件开发者未能对软件做好充分的检测工作，结果提交给用户的软件质量难以保证。这些存在缺陷的软件在运行中暴露出大量的问题,轻者影响系统的正常工作,重者会发生意想不到的重大事故。

在软件开发的几十年实践中，人们一直在程序设计的泥潭中苦苦挣扎。其结果使软件的开发成了既不能放弃又难以圆满解决的棘手问题。但是，由于信息技术对人类的巨大影响，迫使软件的开发者不断地吸取和借鉴人类从事其他工程项目所积累的行之有效的原理、概念、技术和方法。因此寻找解决软件危机途径的过程就是软件工程形成的过程。软件工程目前已成为软件开发中的一门综合技术与管理两个方面的新兴学科，逐渐成为计算机软件开发、维护和管理的重要理论根据。

1.3 软件工程的基本内容

由于软件本身具有不同于其他一些生产对象的特殊属性，这就决定了它需要采用特定的方法和技术进行生产。软件工程以计算机软件生产为对象，其核心是以工程化的原理和方法对软件进行规划定义、开发和维护。其宗旨是以较少的投入,最快的时间生产出高质量的软件。经过许多软件开发研究人员的深入研究和实践，软件工程这一新型的交叉学科正在逐渐发展壮大。

1.3.1 软件工程的概念及其要素

一般认为，软件工程学是指研究软件生产过程的理论和方法以及综合管理技术的应用学科，凝聚了软件实践者的成功经验和失败教训。软件工程一词最早是在 1968 年由北大西洋公约组织在联邦德国召开的一次会议上针对软件危机首先提出的。 30 多年来,人们对软件工程的概念的理解由模糊到逐步清晰和深化。在多方面的研讨中，专家学者对软件工程做了各种各样的定义，其中比较有影响的有以下几种：

P. Wenger 和 B. Boehm 认为“软件工程是科学知识在设计和构造计算机程序以及开发、运作和维护这些程序所要求的有关文档编制中的实际应用。”

F. L. Bauer 认为软件工程是“为了经济地获得可靠并能实际在计算机上运行的软件所需要的工程原理(方法)的确立和使用。”

1983 年 IEEE(国际电气与电子工程师协会）的软件工程术语汇编中将软件工程定义为‘对软件开发、运作、维护、退役的系统研究方法。”

1990 年 IEEE 又在新版的软件工程术语汇编中将软件工程重新定义为“对软件开发、运作、维护的系统化的、有纪律的、可定量的方法之应用,即是对软件的工程化应用。”

今天,作为一门新兴的交叉性学科,软件工程可以归结为 以计算机软件为对象,采用工程化的原理、技术和方法开发和维护软件。使开发的产品具有较高的质量、成本合理、满足用户需求,集技术、管理于一体的综合性、实践性学科。软件工程的主要思想是强调软件开发过程中应用工程化原则的重要性。

作为一门独立的学科,软件工程的内容包括三个方面的要素,即方法、工具和过程。

软件工程方法为软件开发提供了开发、维护的实现技术。它采用特殊的语言或图形方式、结合完善的质量保证手段为软件开发的各个阶段提供了可靠的技术保证。

软件工具是为软件工程方法的顺利实施提供的软件、技术的支撑环境,它为软件工程方法提供了自动化或半自动化的支持。如果将多个工具集成起来，由一个工具输出的信息就可以被另一个工具使用，这样就创建了支持软件开发的综合系统—— 计算机辅助软件工程(CASE)系统。

软件工程过程是软件工程方法和软件工程工具的综合，以人为主为软件开发过程制定的一系列可操作的步骤并规定了每一步使用的方法及结果,其目的是合理、及时地进行计算机软件开发。

软件工程作为工程学科家族的新成员，在自身的不断发展中形成了自己独有的形态。软件产品与其他工程中的产品,如计算机硬件与机床相比较既有相似的一面,也有重大的差别。因此对软件工程的认识既要借鉴传统工程的知识、方法和技术,又要充分注意软件自身的特殊性。在短短的 30 多年的发展历程中，软件工程的理论、方法和技术有了较大的发展,但是由于计算机技术本身就是一门新兴的学科,所以还不能说软件工程学科已经像经典的工程学科那样成熟。随着科技的不断进步，它一定能为人类的软件产业提供完善的理论指导和技术保障。

1.3.2 软件的生存期

软件工程强调使用生命周期的方法从时间的角度对软件开发和维护的复杂问题进行分解。使用这种方法把软件从形成概念开始,经过开发、使用和维护直到退役的漫长周期划分为若干个阶段,每个阶段都有相对独立的任务和解决的步骤和方法。这样,在软件开发过程中有利于软件开发工程的组织和管理，从而降低了整个软件开发过程的困难程度,对每个阶段都可选用最优的管理方法。同时也使每个阶段能够规定更明确的目标和恰当的审核标准。有利于保证软件的质量，大大地提高了软件开发的生产效率和成功率。

软件生命周期的划分涉及软件本身以及软件开发等方面的多种因素，而且根据不同的角度也有不同的划分方法。但它们都有一个统一的原则，即同一阶段内包含的任务性质尽量统一，各阶段任务间尽可能相对独立。目前通常采用的方法是把软件的生命周期划分为三个时期,即软件定义、软件开发和软件维护。

在上述三个时期的叙述方法中，比较明确地揭示了软件开发的基本过程。为了更加深入地揭示软件的生命周期,给软件开发提供更明确的指导,我们把上述的基本时期进一步展开从而揭示软件生命周期的更详细、更具有明确特征的 6 个阶段。即问题的定义及规划、需求分析、软件设计、程序编码、软件测试以及运行维护。

1. 问题的定义及规划

对于每一个软件的开发都有确定的应用领域及使用对象，要使开发出的软件符合使用要求,就必须明确软件要解决的问题。软件开发人员通过与用户的充分沟通,根据用户提出的要求,从开发的角度把问题明晰化以作为软件开发的依据。在问题明确后,接下来的工作就是针对问题进行项目实施的可行性研究,探讨解决问题的可能方案,结合软件开发、使用的可利用条件(计算机硬件、软件、人力等资源)开发费用以及软件投入使用后的经济效益等方面的问题,对定义的问题做出客观的评价,就问题的解决从技术、实施以及经济的角度做出是否可行的明确结论,以作为项目实施的决策依据。若可行,则在此基础上制定出完成开发项目的实施计划。

2. 需求分析

在软件开发的初期用户提交系统要求时,一般只是从使用的角度,对所要求的内容概略地确定了使用计算机解决问题的框架。对于这些要求,有些是开发工作能够满足的,有些需要在实现方法上做一些修改,有些实施起来存在困难,另外还存在软件可以实现的功能而用户没有涉及到的。针对上面的问题，需要软件设计人员在已经明确定义了问题的基础上,反复与用户讨论、沟通。使用户针对提出的问题进一步明确所达到的目的。使软件开发者与用户对开发软件的认识达成一致，从而得到更详细的定义。最后根据分析的结果写出软件需求说明书以及与软件相关的初步说明文件。

3. 软件设计

软件设计是软件实现的一个重要过程。要实现一个软件的良好设计，前提是对软件需求的准确分析。有了良好的分析基础,首先进行总体设计,接下来再做详细设计。在总体设计中，确定软件系统的总体结构，结构中的各部分由与特定需求相对应的模块组成,根据系统的实现要求确定模块之间的关系。在详细设计阶段，对总体设计确定的功能模

块逐步进行细化，为软件程序的编写打下基础。

在软件设计中,为了提高工作效率,有许多可供选择的方法和工具。随着对软件工程研究的不断深入,大量的新方法和有效的工具不断地涌现,这无疑给软件开发注入了新的活力。例如目前常用的结构化设计方法、面向对象设计方法、统一建模语言 UML 以及产品化的辅助工具 Rational Rose 软件等。

4. 程序编码

软件要发挥其功能最终是要有能在计算机上运行的、符合用户要求的程序。程序编码的任务是应用适当的程序语言，把软件设计的结果转换成计算机可运行的程序代码。其要求是语言结构清晰、能准确实现软件的功能、有较高的运行效率并且易于维护。

5. 软件测试

为保证软件的功能得以准确地实现,在软件设计完成后要经过严密的测试,以发现软件在整个设计过程中存在的问题并加以纠正。整个测试过程分单元测试、组装测试以及系统测试三个阶段进行。最后的结果是保证软件达到预定的要求。

6. 软件的维护

软件维护是软件生命周期中持续时间最长的阶段。在软件开发完成并投入使用后，由于多方面的原因,软件不能继续适应用户的要求。要延续软件的使用寿命,就必须对软件进行维护。软件的维护包括纠错性维护和改进性维护两个方面。

1.3.3 软件开发技术及过程管理

1. 软件开发模型

软件开发模型是在软件生存期基础上构造出的由软件开发全过程中的活动和任务组成的结构框架。它反映了软件开发中各种活动的组织衔接方式。它是软件项目开发工作的基础。它规定了软件开发、运作和维护中所需的过程、活动和任务。

软件开发中使用模型化的方法就是用一定的流程将软件开发的各个环节连接起来，用规范的方式操作的软件实现过程。就像一般产品在工厂中的整个生产历程。目前软件工程的研究学者和开发人员根据软件开发的实践经验提出了许多软件开发模型，例如：瀑布模型(图 1.2),渐增式模型(图 1.3),螺旋模型(图 1.4)等等。

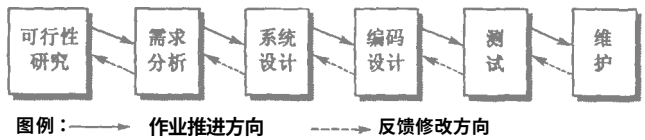

图 1.2 瀑布模型

(1) 瀑布模型——线性化的开发模型

最早出现的软件工程模型是瀑布模型（又称线性模型）。瀑布模型是一种理想化的、单纯的软件开发模式，它是一种文档驱动的模型。在这种模型中，软件开发过程是通过文档连接开发的各个阶段，其中每个阶段既不连续也不重叠。在这种模型中，人们成功地应用了“线性”这个最简单的原理解决软件开发的复杂问题。

虽然瀑布模型在软件开发中因为支持结构化软件开发，在控制软件开发复杂性、降低

计划管理费用、促进软件开发工程化等方面起到了较显著的作用，但是它缺乏灵活性，直到整个开发历程结束时它才能提供有形的软件成果。因此，无法通过开发活动澄清本来不够确切的软件需求。尽管它允许通过反馈修正前一阶段的错误，但是有时根本就行不通。例如，若在编码和调试阶段发现一个架构的缺陷，是很难反向进行修改的。因此随着软件规模的日益庞大，要求开发周期的日益缩短，该模型的不足所引发的问题显得更加突出。

(2) 渐增式模型

该模型实质就是分段的线性模型，如图 1.3 所示。渐增式模型将系统开发用一系列“渐近”过程实现，每次的‘渐近’都由一个瀑布生命周期组成。通常，设计从最显著的方面开始，然后根据用户的反馈信息增添、提炼原型。每‘渐近’一次都更加深入和详细，模型的每个周期都产生一个原型，经过若干个周期循环后，系统就建成了。这种循序渐进的方法，对减少由于软件需求把握不准确给软件开发造成的风险有显著的效果。

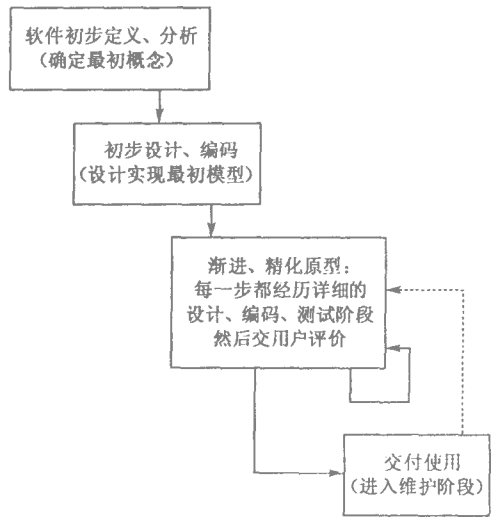

图 1.3 渐增式模型

(3)螺旋模型

依照渐增式模型的原理，螺旋模型则是连续的、弯曲了的线性模型。每个螺旋推进的过程都是渐进的实现过程。整个过程的实现，按照‘制定计划，风险分析，实施工程，客户评价’四个步骤循环实施。在实施过程中加入了风险分析，提高了风险识别和规避的能力。

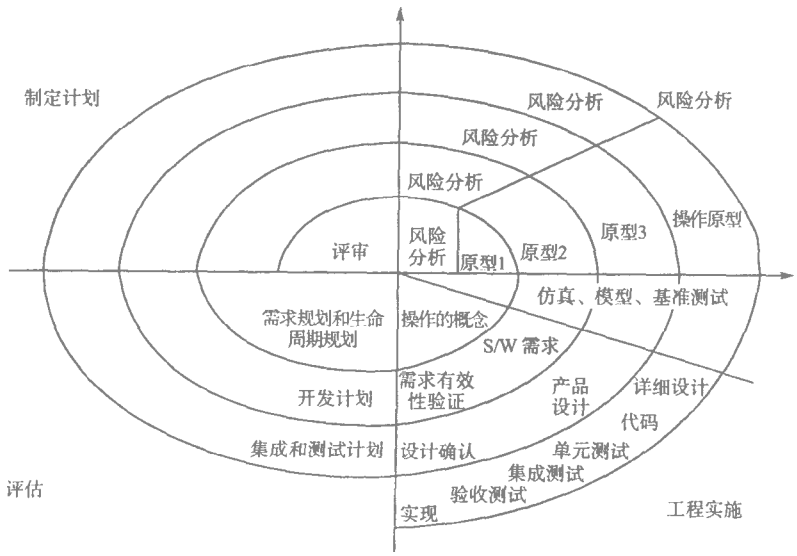

图 1.4 螺旋模型

实践证明，对于大型软件的开发，螺旋模型是最为实际的方法。它不但适用于面向规

格说明、面向过程和面向对象的软件开发方法,也适用于几种开发方法的组合而产生的组合模型。

对于实际的软件开发,由于问题的复杂性,一般一种模型很难符合要求。通常把几种模型组合在一起,配套使用,形成了组合模型。这种把模型组合的方法,允许一个项目沿着最有效的路径发展,从而容易达到降低软件开发成本、缩短软件开发时间、提高软件产品质量的目的。

2. 软件开发的主要技术

在软件开发中通常采用的技术方法有结构化分析和设计技术、Jackson 软件设计技术、原型设计技术以及面向对象的软件设计技术等。

结构化分析和设计技术 SADT(Structured Analysis and Design Technique)是 Softech 公司为解决复杂问题而提出的一种软件设计技术,它也是目前软件系统开发中使用最广泛的一种技术。SADT 首先对问题进行结构分析,然后逐层分解,直至明了做什么(在需求分析阶段)或怎样做(在设计阶段)为止,最后写出详细的系统说明书。SADT 技术软件系统的开发过程以瀑布式生命周期模型为基础,主要用于需求分析和功能设计阶段。

Jackson 技术为面向数据流的软件设计技术,称为 Jackson (也称面向数据结构的)软件设计技术。在许多应用领域,数据结构对软件设计的影响很大,不仅影响软件结构的设计,还影响软件过程的设计。应用领域中的任一数据结构都可以概括为以下结构 重复性的控制结构即循环结构、具有选择特性的条件语句结构和分层次的层次软件结构。所以,基于数据结构能充分揭示软件结构的思想,Jackson 方法为面向数据结构的设计定义了一组以数据结构为指导的映射过程。

原型设计技术(Prototypes Technology) 是针对传统瀑布模型的缺点而提出的一种新的软件开发技术。它通过建立和使用一系列原型来进行系统的设计、实现、测试和安装。其核心思想是快速建立一个具有若干功能(不要求完全)可执行的原型,用该原型启发用户和设计者不断试用和完善(甚至可抛弃)。每次完善都获得一个新原型,直到满足用户全部要求为止。因此,许多人把原型技术称为‘ 快速弱功能 ’开发技术。

面向对象(Object Oriented)设计技术 从 80 年代初以来,面向对象技术在计算机软件开发技术研究和应用领域中十分活跃。面向对象技术不仅已在计算机学科的传统领域,如程序设计语言、数据库、程序设计方法等方面取得成功,而且在 CASE(计算机辅助软件工程)、CAD(计算机辅助设计)、CAI(计算机辅助教育)、多媒体、超文本、超媒体等新领域中得到广泛应用。

一般认为,面向对象软件 =(数据 + 相应操作)的封装,以数据为中心。面向对象软件是事物的集合,通过对象以及对象和对象之间的通讯联系实现。它采用交互式、并行处理方式,由消息驱动控制。从应用的角度看,面向对象方法更加适合大型复杂的人机交互式软件和数据统计管理软件的开发。

目前,随着计算机技术的不断进步,面向对象技术正沿着 OOPL(面向对象的程序设计语言)→OOD(面向对象设计)→OOA (面向对象分析)的方向发展。面向对象方法代表了软件开发方法的发展方向。

3. 软件过程管理

软件工程过程是软件工程师在软件开发过程中为实现开发目标而完成的一系列软件工程活动。其关键部分是软件开发和维护中的管理和支持能力。一个好的管理是成功的基础，一个差的或者是不当的管理势必导致过程的失败。

为了成功地实施软件开发过程,管理者应对软件工程的目标、所需要的资源、经费以及工作量有一个基本的了解。制定一个科学的软件开发计划、有效实施的具体措施以及完善的质量控制标准，并在软件开发过程中合理地掌握进度。

软件工程是一门实践性很强的应用学科，它的对象主要是软件开发中的社会学——人的问题,因而在实践中有很大的灵活性和多样性。即使是相同性质项目的开发,针对不同的团队(人员)也不能照搬原有的开发管理模式,必须根据实际情况进行调整。因此科学的管理是软件项目成功不可缺少的重要组成部分。

1.4 软件工程的基本目标和原则

1.4.1 软件工程项目的基本目标

软件工程的目标是在给定成本、工期的前提下,达到要求的软件功能,取得较好的软件性能,开发的软件易于移植、可靠、有效、可重用。并且尽量提高软件质量与生产率,最终实现软件的工业化生产目标。在诸多的因素中质量是软件需求方最关心的问题，而生产率是软件供应方最关心的问题。从经济学的角度看，质量与生产率之间有着内在的联系,高生产率必须以质量合格为前提。质量与生产率之间不存在根本的对立,好的软件工程方法可以同时提高质量与生产率。

1.4.2 软件工程的基本方法及其原则

从特征上看,软件工程属于工程科学的范畴,许多成功的软件设计就是借鉴了工程设计中的基本方法。但是,从基本特征上看,软件设计与一般的工程设计还是有较大的区别的。由于软件是不可见的、抽象的、复杂的逻辑实体,所以软件的设计过程是一个交替进行比较、分类、推理、归纳、演绎、综合、分析的思考过程。软件设计中科学的方法和原则，对于保证软件工程项目的顺利进行、确保软件目标系统的质量是十分重要的。软件开发一般应遵守以下方法和原则 抽象、局部化与信息隐蔽、模块化、一致性、完整性和可验证性。上述过程的实现是通过模块化处理来实现的。

1. 抽象

抽象就是透过现象深入里层，抽取出本质的过程和方法。人们对客观事物的认识是通过一系列抽象思维来完成的,抽象思维的过程就是运用概念进行判断、推理的过程。工程设计中的许多重要技术都是以抽象为基础的,如系统、模型、设计流程等。软件工程属于工程科学范畴,软件项目的开发也继承了这些思想、方法和技术。

2. 局部化与信息隐蔽方法

局部化是指把一些有关的、具有特定目的的软件要素放在一起，以使程序的一个部分与另一个部分不产生相互牵扯和影响。信息隐蔽是指一个模块将一些具有特定目的的数据及对数据的操作封装起来使其内部与外界相隔离。模块信息隐蔽的结果意味着系统有效的模块化可以通过定义一组独立的模块来实现，这些独立的模块彼此之间仅仅交换那些为了完成系统功能所必须交换的信息。当测试或软件维护期间需要修改软件时，使用局部化与信息隐蔽原则设计的软件会把修改的影响限制在最小的范围之内。这对于提高程序的可靠性、可维护性以及模块的重用性尤为重要。

3. 模块化(高内聚,低耦合)

软件模块化，是指把软件系统划分为若干个逻辑上独立的模块，每个模块完成独立的子功能，所有的模块集合起来满足问题的要求。模块化是设计复杂软件的有效方法，也是为了有效地管理和维护软件经常采用的方法。模块化设计使软件结构清晰、设计方便、理解容易，并且易于修改和测试，有助于提高软件的可靠性和程序员的分工合作。

模块独立，是指一个模块的工作不依赖于另一个模块的存在，模块与模块之间的联系只是不可缺少的必要的数据联系。它是抽象、模块化、局部化和信息隐蔽的直接结果，也是获得良好设计的关键。对降低软件成本、提高软件开发的成功率至关重要。

模块的独立性可以由两个定性的标准来衡量，即内聚度和耦合度。内聚度是指模块内部各个成分之间联系的紧密程度，用来衡量模块的内部特性；耦合度是指模块与模块之间互相依赖的程度，用来衡量模块的外部特性。一个模块的内聚度愈高、耦合度愈低，模块的独立性就愈好。

4. 一致性

为了保证系统的良好性能，在软件设计时应使用统一的结构和规则。这包括系统接口的一致性、符号术语的一致性以及规格说明与实现对象的一致性。为实现一致性的原则,软件工程提供了许多有效的软件设计工具(如数据词典、数据库、文档自动生成与一致性检查工具）和设计方法以及编码风格的支持。

5. 完全性和可验证性

一个成功的软件应该完全、充分地实现系统所需的功能，而且要使系统有一定的“坚固性”，即在系统处于非正常状态时有一定的使系统行为保持正常的能力。同时，对于一个在开发中分解细化的大型软件系统应注意遵循容易检查、测试和评审的原则。

本章小结

本章介绍了软件以及软件工程所涉及的一些基本概念和软件设计中的有关理论以及实践中的方法和原则。从软件开发所引发的软件危机以及解决的方法导入了软件工程的概念、要素及其在软件开发中的方法和作用。涉及的内容有软件及其特征、软件工程的概念及要素、软件工程过程、软件的生存期和开发模型等概念。

习　　题

1. 通过你自己使用计算机的经历和对计算机认识分辨软件与程序的差别，指出区别的关键点是什么。

2. 简述软件危机产生的原因以及避免的方法。

3. 软件工程的要素有哪些？简述软件工程在软件开发中的作用和意义。

4. 软件的生命周期对软件的开发有哪些指导作用？

5. 分析瀑布模型和螺旋模型的异同，举例说明哪些软件适用于采用瀑布模型，哪些软件适用于螺旋模型,并说明原因。

第2章

软件的定义及规划

一般说来,把软件的生存期划分为软件定义、软件开发、软件运行三个时期,每个时期又可划分为若干个阶段。本章将讨论软件定义时期的主要任务。

软件定义时期是生命周期的第一个时期，也是软件开发的基础。根据软件开发的基本过程,这个时期可分为两个阶段 问题定义和可行性研究。这两个阶段的主要任务就是分析用户要求，在对用户要求充分了解的前提下,分析未来新系统(即目标系统)的主要目标，分析开发系统的可行性。参加这个时期工作的人员有用户和软件开发系统分析员。图 2.1 显示了这一时期的工作顺序。

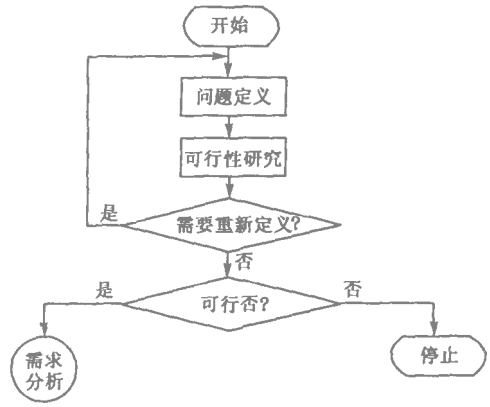

图 2.1 软件定义时期工作流程

2.1 问题定义

问题定义是软件定义时期的第一个阶段，作为软件的开发者，在这个阶段必须弄清用户“需要计算机解决什么问题?”如果在问题尚未明确的情况下就试图解决这个问题,那么就会白白浪费时间和精力，结果也毫无意义。因此，问题定义在软件生命周期中占有重要的位置。

2.1.1 问题定义的内容

问题定义的主要内容有：

(1) 问题的背景 弄清楚待开发系统现在处于什么状态,为什么要开发它,是否具备开发条件等问题。

(2) 提出开发系统的问题要求以及总体要求。

(3) 明确问题的性质、类型和范围。

(4) 明确待开发系统要实现的目标、功能和规模。

(5) 提出开发的条件要求和环境要求。

以上主要内容应写在问题定义报告（或系统目标和范围说明书）中，作为这一阶段的“工作总结”。

2.1.2 问题定义的方法

在问题定义阶段，需要用户和系统分析员共同协作、紧密配合，方能圆满地完成问题定义报告。

具体步骤如下：

首先，系统分析员要针对用户的要求做详细的调查研究，认真听取用户对问题的介绍，阅读与问题有关的资料，必要时还要深入现场，亲自操作，调查开发系统的背景，了解用户对开发的要求。

其次是与用户反复讨论，以把问题进一步确定。经过用户和系统分析员双方充分共同协商，确定问题定义的内容。

最后写出双方均认可的问题定义报告。具体做法示例如下：

例 1 某高校教务处提出用微机管理教务工作的要求，经过分析员的调查研究并与用户协商，写出了如下问题定义报告。

系统目标和范围说明书

1. 项目名称：××学院教务管理系统。

2. 背景：目前教务人工管理，效率低，易出错。

3. 项目目标：在网络上建立一个高效的、无差错的教务管理系统。

4. 项目范围：利用现有的校园网，教务处及各系均配备一台专用微机，软件开发费不超过 1 万元。

5. 初步设想：在网上实现上报各种信息(学生成绩、教师课时费等)、排课、下通知。

2.2 可行性研究

可行性研究是在问题定义之后进行的，它是软件定义时期的第二个阶段。可行性研究的目的是明确“问题是否能够解决？”和“是否值得去解决？”。也就是判断为开发系统所定的目标和规模是否能够实现，新系统是否能够带来经济效益。一般来说，在投入大量资金前，都要研究成功的可能性和所要冒的风险。

在这个阶段，往往要为前一步提出的问题寻求一种至数种在技术上可行且在经济上有较高效益的解决方案以供比较和选择。因此，可行性研究的实质是在高层次上做一次大大简化了的需求分析和设计。

作为可行性研究的成果，在最后要写出《可行性论证报告》。

2.2.1 任务

可行性研究的任务是对已提出的任何一种解决方案，都要从经济、技术、运行和法律诸方面来研究其可行性，并做出明确的结论供用户参考。

1. 技术可行性

从技术的角度去研究系统实现的可行性。主要包括：在给出的限制范围内，能否设计出系统,并实现必要的功能和性能 开发人员、硬件和软件是否存在问题 系统所用到的相关技术是否支持。

2. 经济可行性

是对软件开发项目进行成本／效益估算，分析实现这个系统有没有经济效益。

3. 运行可行性

指为新系统规定的运行方式是否可行。如果新系统建立在原来已担负其他任务的原系统上，就不能要求它在实时在线状态下运行，以免与原有的任务相矛盾。

4. 法律可行性

研究新系统的开发会不会在社会上和政治上引起侵权、破坏以及会不会与法律相抵触等问题。

可行性研究最根本的任务是对以后的行动方向提出建议。如果可行性研究的结果是问题没有可行的解，那么系统分析员应该建议停止这项工程的开发；如果可行性研究的结果是问题值得去解决，那么系统分析员应该推荐一个较好的解决方案，并且为工程制定一个初步的开发计划。

2.2.2 可行性研究的方法和步骤

可行性研究的整个过程是从分析《新系统目标与范围的说明书》开始到新系统的推荐方案通过审查为止。在整个过程中，要经过以下步骤：

1. 审核系统的规模和目标

系统分析员应对《新系统目标与范围的说明书》进行再审查，确保正在解决的问题是用户要求解决的问题。这一步要做的工作有：(1)再次确认新系统的规模和目标；(2)改正含糊不清或不确切的叙述；(3)再次确认新系统的一切限制和约束。

完成以上工作，通常采用的方法是：多次访问关键人员；仔细地阅读和分析有关资料；深入现场，熟悉处理流程。

2. 研究当前正在使用的系统

系统分析员可以通过阅读资料和深入现场，对当前正在使用的系统进行研究，总结出当前系统的特点，从而得到新系统的雏形。比如：当前系统和其他系统的接口情况就是设计新系统的重要约束条件。另外，还需要了解运行当前系统所需的费用。这是我们建立新系统时要考虑的，新系统的经济效益应超过当前系统。

3. 导出新系统的高层逻辑模型

一个好的设计应该完成以下工作：(1)由当前物理系统提出当前系统的物理模型；(2)由当前系统的物理模型，导出当前系统的逻辑模型；(3)参考当前系统的逻辑模型,设想出

新系统的逻辑模型；(4) 根据新系统逻辑模型建造新系统物理模型。见图 2.2。

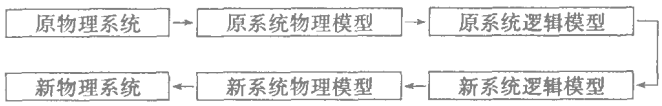

图 2.2 新系统建立过程

系统的逻辑模型通常用数据流图（详见 2.2.4 节)和数据辞典(详见 3.4.2 节)共同描述。系统的物理模型通常用系统流程图（详见 2.2.3 节)来描述。

在可行性研究阶段，我们只导出新系统的高层数据流图。关于数据流图的细化工作在需求分析阶段完成(详见 3.4.3 节)。

4. 重新定义问题

新系统的逻辑模型只表达了系统分析员对新系统“做什么”的看法，用户是什么看法还不知道。因此，分析员和用户应该一起对问题进行再定义，再次复审工程规模目标和约束条件，发现对问题的说明或对用户要求有遗漏应及时修改。

可行性研究的前四步构成了一个循环：审核问题→分析问题→导出一个试探性的解法→在此基础上，再审核问题→再分析问题→再修改解法……，继续这个过程，直到得到一个完全符合新系统目标的逻辑模型为止。见图 2.3。

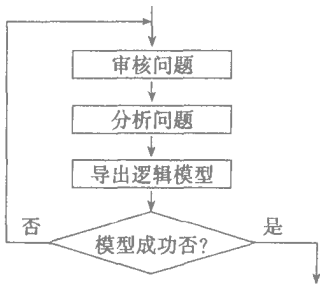

图 2.3 建立系统逻辑模型的过程

5. 提出和评价供选择的方案

分析员根据新系统逻辑模型，提出若干个较高层的(较抽象的)物理解决方案，供比较和选择。

对于已提出的多个解决方案，首先要进行技术可行性分析，排除那些技术上不现实的方案；其次再考虑经济方面的可行性，并与当前系统费用进行比较，看新系统经济效益是否有提高；最后，考虑每个方案的社会可行性，即是否违背国家法律，是否可能有版权纠纷，是否符合社会生产客观体制要求，是否能安全生产等。

6. 推荐一个方案和行动方针

在上一步各种方案分析的基础上，若分析员认为值得开发这个项目，那么就应该向用户推荐一个最好的解决方案，推荐中应表明：

(1) 项目的开发价值；

(2) 推荐这个项目的理由。

同时做出一个关键性的决定，表明是否进行这项开发工程。

7. 草拟项目开发计划

分析员要为被推荐的系统草拟一份开发计划。计划中应包括：

(1) 工程进度表：通常只估计生命周期每个阶段的工作量。

(2) 开发人员：估计生命周期每个阶段对于系统分析员、程序员、资料员等的需要状况。