计算机等级考试教程

（四级）

软件工程

姚淑珍　杨文龙　编著

机械工业出版社

计算机等级考试教程（四级）软件工程

计 算 机 等 级 考 试 教 程

（四 · 级）

软 件 工 程

姚淑珍　杨文龙　编著

机 械 工 业 出 版 社

本书全面阐述了软件工程的基本概念，以结构化方法为重点，详细分析了目前流行的各种软件开发方法，按瀑布模型介绍了软件开发过程，开发活动及文档规范，最后对软件质量保证，软件管理的任务及措施进行了综述。

全书内容深入浅出，配合有大量实例与习题，可作为全国计算机等级考试四级应试人员的教材，亦可作为高等院校计算机专业教材，从事计算机软件工作的技术人员和管理人员的参考用书等。

图书在版编目（CIP）数据

计算机等级考试教程(四级)：软件工程/姚淑珍　杨文龙　编著．北京：机械工业出版社，1996.4

ISBN 7-111-04994-2

Ⅰ．计…　Ⅱ．①姚…②杨…　Ⅲ．①电子计算机-技术等级标准-考核-教材②软件工程-技术等级标准-考核-教材　Ⅳ．①TP3②TP31

中国版本图书馆 CIP 数据核字（96）第 00268 号

出 版 人：马九荣（北京市百万庄南街 1 号 邮政编码 100037）

责任编辑：何文军　谢广钰　版式设计：张世琴　责任校对：刘志文

封面设计：郭景云　　责任印制：卢子祥

三河市宏达印刷厂印刷　·新华书店北京发行所发行

1996 年 4 月第 1 版第 1 次印刷

787mm×1092mm 1/16 · 11.25 印张 · 268 千字

0 001—5 000 册

定价：18.00 元

前　言

在计算机时代的前30年，主要问题是如何提高硬件水平，以降低其存储和处理的开销。80年代以来，随着微电子技术的发展，计算机性能不断提高，价格持续下降，现在的主要问题是如何以较高的质量和较低的成本来满足信息时代对软件开发和维护提出的要求，这正是软件工程急待解决的问题，因此，国家教委考试中心将软件工程作为计算机等级考试的一门主要课程，以此加快我国软件工程化进程，提高全民软件开发水平是十分必要和恰当的。

《软件工程》主要介绍如何运用工程学的原理和方法来组织和管理软件生产，以保证软件产品的高质量和高生产率。为提供全面的软件工程基础理论知识，提高读者的软件开发能力，本教材内容包括：

1）软件工程基本概念；

2）结构化分析与设计方法；

3）面向对象方法；

4）其他软件开发方法；

5）软件测试；

6）软件质量与质量保证；

7）软件维护；

8）软件管理；

9）软件开发工具与环境。

本教材是按计算机等级考试四级要求编写的，目的是使读者在软件工程方面能达到相当于计算机专业本科毕业生水平。

由于编者时间和水平的限制，难免有不妥之处，敬请读者批评指正。

作　者

1996年1月于北京

航空航天大学计算机系

目　录

第1章　软件工程基本概念

从世界上出现第一台计算机到60年代中期是计算机系统发展的早期。在这个时期，以计算机为基础的系统是采用面向硬件的管理技术进行开发的，工程管理者将主要精力集中在硬件上，因为硬件是系统开发预算中份额最大的一部分。为控制硬件成本，管理者建立了正式的控制和工艺标准，以在开发产品之前进行细致的分析和设计。简单地说，他们使用了所谓硬件工程的一套控制方法和工具。硬件工程发展至今，过去计算机主机的功能可由一个单片集成电路实现，计算机硬件这种巨大的处理和存储能力充分展示出计算机发展的潜力，计算机系统开发人员的主要工作就是如何利用和挖掘这个潜力，这正是软件开发所要完成的任务。但计算机系统发展早期所形成的一些错误观念和做法，已严重地阻碍了计算机软件的开发，以至所开发的许多大型软件几乎根本无法维护，只好提前报废，损失惨重。一些工业发达国家的计算机科学家把软件开发和维护过程中遇到的一系列严重问题统称为“软件危机”，并且在60年代后期开始认真研究解决软件危机的方法，从而逐步形成了计算机科学技术领域中的一门新兴的学科——计算机软件工程学，通常简称为，软件工程。

1.1　软件与软件危机

在计算机系统中，程序是不可缺少的。相对硬件而言，程序群称为软件。硬件与软件合成一体，完成系统的功能。也就是说计算机软件的实体是程序，而程序是使实际存在的计算机动起来的指令序列。软件的概念就是从这样实际的问题开始并逐渐明确下来的。

1.1.1　软件

软件定义形式有多种，比较公认的一种定义为软件由三部分组成：

1）在运行时能提供所希望的功能和性能的指令集，即程序；

2）使得程序能够正确运行的数据结构；

3）描述程序研制过程、方法及使用的文档。

随着计算机应用的日益普及，软件变得越来越复杂，规模也越来越大，超过100万条指令，由几百人经过几年时间才开发出来是常见的。如何使人与人、人与机间相互了解，保证开发与维护工作的顺利进行，文档（即各种报告、说明、手册等的总称）是不可缺少的。特别在软件逐步成为商品的今天，文档的作用更加重要。

相对硬件而言，软件有诸多特点。如：1）它是一种逻辑实体，因而具有抽象性，虽然可以记录在介质上，但无法看到软件本身的形态；2）它是在研制、开发活动中被创造出来的，研制（开发）成本远大于生产（拷贝）成本；3）软件在长期运行和使用过程中没有磨损，老化等问题；4）软件的开发和运行常常受硬件的限制；5）软件的开发至今尚未完全摆脱手工工艺的开发方式；6）软件开发费用愈来愈高。据统计，目前软件开销要占计算机系统费用的90%以上。由于软件所表现出的特有属性，使得在60年代末软件开发面临许多严重的急需解决的问题，即面临软件危机。尽管经过30年的努力，软件危机还没有得到彻底克服。

1.1.2 软件危机

具体地说，软件危机主要有下述一些表现：

1）由于软件开发的不可见性，缺乏软件开发的经验和关于软件开发数据的积累，使得软件开发成本和进度的估计常常很不准确；

2）由于开发过程没有统一的、公认的方法论或规范指导，使得在后期对软件进行修改和功能扩充时非常困难，即软件常常是不可维护的；

3）由于没有将软件质量保证技术（审查、复审和测试等）应用于软件开发的全过程，因而软件产品的质量往往靠不住；

4）在软件开发初期，软件开发人员常常在对用户要求只有模糊的了解，甚至对所要解决的问题还没有确切认识的情况下，就仓促上阵，匆忙着手编写程序，期间开发人员与用户又很少交换意见，从而导致用户对“已完成的”软件系统不满意的现象经常发生；

5）软件通常没有适当的文档资料。文档是软件的重要组成部分。软件开发组织的管理人员可利用文档来管理和评价软件开发工程的进展状况，软件开发人员可以利用文档作为通信工具，在软件开发过程中准确地交流信息。对于软件维护人员而言，文档是了解软件、记录修改的手段。缺乏必要的文档资料或者文档资料不合格，必然给软件开发和维护带来许多严重的困难和问题；

6）软件产品“供不应求”的现象使人们不能充分利用计算机硬件提供的巨大潜力，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。

软件危机的出现，向人们提出了以下问题：1）如何开发软件；2）如何维护已有软件；3）如何满足社会对软件的日益增长的需要。要解决这些问题，既要有技术措施（方法和工具），又要有必要的组织管理措施。软件工程正是从管理和技术两方面研究如何运用工程学的基本原理和方法来更好地开发和维护计算机软件的一门新兴学科。

1.2 软件生命周期与软件工程

人类解决复杂问题时普遍采用的一个战略就是“各个击破”，也就是采用对问题进行分解，然后再分别解决各个子问题的战略。软件工程采用的生命周期方法学就是从时间角度对软件开发和维护的复杂问题进行分解，把软件生存的漫长周期依次划分为若干阶段，每个阶段有相对独立的任务，然后逐步完成各个阶段的任务。这里所说的“软件生存的漫长周期”是指从提出软件产品开始，直到该软件产品被淘汰的全过程。研究软件生命周期是为了更科学地、有效地组织和管理软件的生产、维护，从而使软件产品更可靠，更经济。当采用软件工程方法论开发软件时，按软件生命周期的阶段划分，一个阶段一个阶段地进行开发。前一个阶段任务的完成是开始进行后一个阶段工作的前提和基础，而后一阶段任务的完成通常是使前一阶段提出的解决方案更进一步具体化，加进了更多的物理细节。每一个阶段的开始和结束都有严格标准。对于任何两个相邻的阶段而言，前一阶段的结束标准就是后一阶段的开始标准。在每一阶段结束之前都必须进行正式严格的技术评审和管理评审，即从技术和管理两方面对这个阶段的开发成果进行检查。检查通过之后，这个阶段才算结束；如果检查通不过，则必须进行必要的返工，并且返工后还要再经过审查。

目前，软件生命周期的阶段划分方法有多种，软件规模、种类、开发方式、开发环境及开发时使用的方法论都影响软件生命周期阶段的划分。划分软件生命周期阶段时应遵循的一

条基本原则就是，使各阶段的任务彼此间应尽可能相对独立，同一阶段各项任务的性质应尽可能相同，从而降低每个阶段任务的复杂程度，简化不同阶段之间的联系，以有利于软件开发工程的组织管理。

一般说来，软件生命周期由软件定义、软件开发和软件维护三个时期组成，每个时期又进一步划分成若干个阶段。下面的论述主要针对应用软件，不过对系统软件也基本适用。

软件定义时期的任务是：1）确定软件开发工程必须完成的总目标；2）确定工程的可行性；3）开发软件系统需要哪些资源（人力资源和设备资源）；4）作出成本估算；5）制定工程进度表；6）明确软件需求和软件验收标准；7）在经过管理评审和技术评审之后进入到软件开发时期。

软件开发时期是具体设计和实现在前一时期定义的软件，它通常由下述四个阶段组成：概要设计、详细设计、编码和测试。

软件维护时期的主要任务是使软件持久地满足用户的需要，即：1）当软件在使用过程中发现错误时应该加以改正；2）当环境改变时应该修改软件，以适应新的环境；3）当用户有新要求时应该及时改进软件，以满足用户的新需要。通常对维护时期不再进一步划分阶段，但是每一次维护活动本质上都是一次压缩和简化了的定义和开发过程。

由上可见，软件生命周期中，每个阶段都有确定的任务，并产生一定规格的文档，送交下一个阶段；而下一阶段是在前一阶段评审通过的基础上，继续开展工作。

软件工程强调使用生命周期方法学和成熟的技术方法来开发软件。它将一套形式化过程应用于计算机软件的定义、开发和维护，软件生命周期的每一步骤都要文档化和评审，各个过程都有先进、成熟的技术支持，软件开发要求管理和技术并重。因此，概括地说，软件工程是一门涉及软件计划、需求分析、设计、编码、测试和维护的原理、方法及工具的研究和应用的学科。

1.3 软件开发过程模型

软件工程是计算机软件走向工程化的途径。在软件开发过程中，软件工程思想具体为软件开发过程模型及其实施于过程模型中的软件开发方法和工具。计算机界人士对软件开发过程经过几十年的探索，提出了四种典型的软件开发过程模型。

1.3.1 瀑布模型（Waterfall Model）

典型的瀑布模型可采用 B. W. Boehm 的描述，他将软件生命周期划分为七个阶段，各阶段的任务分别为系统需求分析、软件需求分析、概要设计、详细设计、编码、测试和运行维护，每一阶段工作的完成需要评审等技术的确认，如图 1-3-1 所示。

瀑布模型的主要特点是：阶段间的顺序性和依赖性，开发过程是一个严格的下导式过程，即前一阶段的输出是后一阶段的输入，每一阶段工作的完成需要确认，而确认过程是严格的追溯式过程，后一阶段出现了问题要通过前一阶段的重新确认来解决。因此，问题发现得越晚解决问题的代价就越高。

瀑布模型的主要不足之处有以下两点：

1）从认识论上讲，人的认识是一个多次反复的过程，实践——认识——再实践—再认识，多次认识，多次飞跃，最后才能获得对客观世界较为正确的认识。软件开发是人的一个智力

认识活动，也不可能一次完成，也需要多次反复地进行，但瀑布模型中划分的几个阶段，没有反映出这种认识过程的反复性。

2）软件开发是一个知识密集型的开发活动，需要人们合作完成，因此，人员之间的通信和软件工具之间的联系、活动之间的并行和串行等都是必需的，但在瀑布模型中也没有体现出这一点。

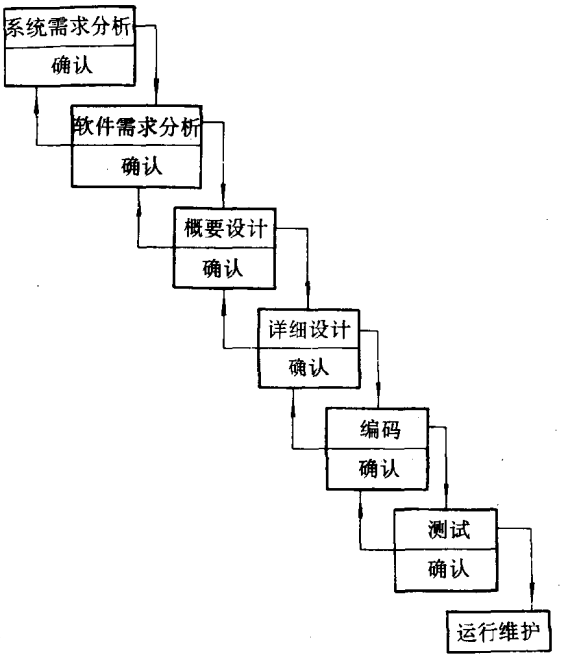

图 1-3-1　瀑布模型

1.3.2　原型模型（Prototyping Model）

原型模型是借助程序自动生成工具或软件工程支撑环境，尽快地构造一个实际系统简化的模型，以便供开发人员和用户进行交流，较准确地获取用户的需要，它的示意如图 1-3-2 所示：

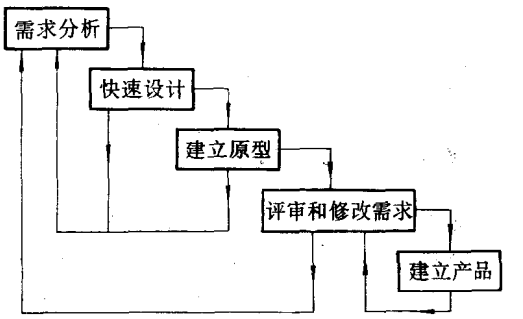

图 1-3-2　原型模型

原型模型的主要特点是：首先建立一个能够反映用户主要需求的原型，为用户展示未来软件系统的概貌，使用户可以比较直观地从最终软件产品的角度出发对原型提出修改意见，软件人员反复改进，最终建立完全符合用户要求的软件系统。这实际上是一个软件人员不断向用户提供样品，而用户对其做出迅速反馈，以进一步改善样品的过程，这就大大避免了在瀑布模型冗长的开发过程中，看不见最终软件产品雏形的现象。

原型模型在各个阶段用户反馈活动的基础上，突出了快速的改进过程，它改变了瀑布模型的线性结构，采用逐步求精方法使原型逐步完善，以满足用户的要求，是一种在新的高层次上不断反复推进的过程。

原型模型的不足之处有以下两点：

1）为了使系统尽快地运行起来，系统开发人员在初期往往考虑得不周全，经常采取一些折衷的方案，有可能使原型不能成为最终软件产品的一部分，只是一个示例而已（称这种原型为示例型）。这样，在实际开发软件产品时，仍然有许多工作要做。

2）原型模型需要大量完备和实用的软件工具的支持才能得以实现，即原型模型对工具和环境的依赖性较高。

1.3.3　喷泉模型（Fountain Model）

在面向对象（OO）方法中，提出了与瀑布模型相对应的喷泉模型，该模型认为软件生命周期的各个阶段是相互重叠和多次反复的，（见图 1-3-3），就像水喷上去又可以落下来，即可以落在中间，也可以落在最底部。

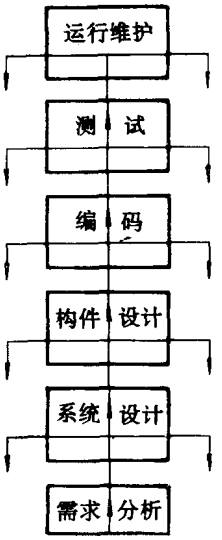

图 1-3-3 喷泉模型

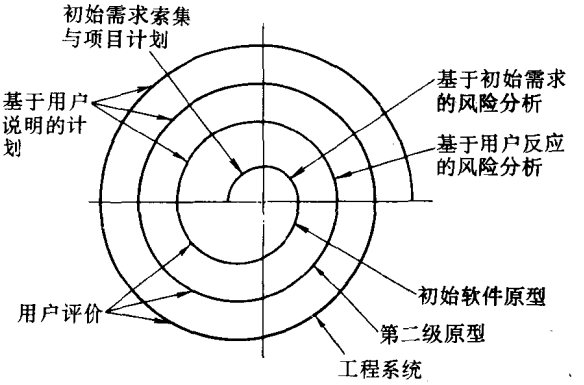

图 1-3-4 螺旋模型

1.3.4 螺旋模型 (Spiral Model)

相对瀑布模型而言，原型模型更符合人类认识真理的渐近修正的思维方式，它刺激了有良好工具和可重用成分的环境的发展。但在原型评价、改进的多次反复过程中，可能会引入其他风险，诸如计划的调整，需求的增加等。为解决这一问题，Boehm 等人在原型模型基础上又提出多次原型反复并增加风险评估的螺旋式开发模型，如图 1-3-4 所示。

1.4 软件开发方法

软件开发过程模型规定软件开发活动的组合应用方式，要保证开发活动的高质量，还需要有相应的软件开发方法作为技术支持。近 10 多年来，软件工作者研制出了许多工程化的软件开发方法，例如 70 年代初提出的用于编写程序的结构化程序设计方法，确实起到了提高效率，减少错误的效果。但是 70 年代中期，软件工作者认识到编写程序仅仅是软件开发的一个环节，而合理地建立系统结构比编写程序更为重要。所以研究的重点前移到设计阶段，出现了设计阶段的结构化设计 (SD) 方法和 JACKSON 等方法，到了 70 年代后期，人们又发现事先对用户的要求进行分析更为重要，故又把重点前移到分析阶段，出现了用于分析阶段的结构化分析 (SA) 方法、结构化分析与设计技术 (SADT) 等。随着计算机技术的迅速发展，在 80 年代初期的实时、并发和网络等软件的开发过程中，特别是在第五代计算机研究工作中，又提出了面向对象的设计方法。

现在流行的方法有多种，它们的适用范围也各不相同。有的适用于一般的数据处理系统，如 SA、SD（两者统称为结构化分析与设计方法，即 Yourdon 方法）、JACKSON 方法；有的适用于大型的复杂系统，如 SADT 技术；有的适用于实时事务处理系统，如 FSM 方法；有的适用于并发软件系统，如 PETRI 网方法；作为 90 年代代表作的面向对象方法，其应用已几乎遍布各个领域。这些方法除了适用范围不同外，方法形成的基础、处理规则和对所开发软件风格的要求等都各有侧重。用什么方法来说明用户的要求、用什么方法来设计软件以及用

什么方法对软件进行测试和维护，直接影响着所开发软件的质量。下面几章，我们将重点介绍结构化分析与设计方法、面向对象方法，概括论述其他常用的软件开发方法，并对这些方法进行总体比较。

习　题

（一）名词解释

1. 软件
2. 软件危机
3. 软件工程
4. 软件生命周期
5. 瀑布模型
6. 原型模型
7. 喷泉模型
8. 螺旋模型

（二）问答题

1. 何为软件，它包括哪几部分？
2. 软件危机主要有哪几种表现？
3. 什么是软件工程？试说明软件工程是如何克服软件危机的。

（三）论述与综合题

1. 列出两种以上软件开发过程模型，并说明它们各有什么特点。
2. 你认为书中所列的软件危机目前已彻底克服了吗？除此之外，是否又引入了新的软件危机？

第 2 章　结构化分析与设计方法

结构化分析与设计方法在软件工程中应用已很普遍，并且越来越成熟。有许多大、中型项目都采用了这种方法进行开发并取得了显著的成果。本章根据瀑布模型的阶段划分，按阶段介绍有关的任务、原则和基本方法。

按 B. W. Boehm 的描述，瀑布模型的软件生命周期可划分为七个阶段：系统需求分析、软件需求分析、概要设计、详细设计、编码、测试和运行维护。“系统需求”包括：问题定义、可行性研究及软件计划，下面我们诸节介绍。

2.1　问题定义

软件开发的第一步就是进行问题定义。问题定义阶段必须回答的关键问题是：“软件要解决的问题是什么?”如果不知道问题是什么就试图解决这个问题，显然是盲目的，只会白白浪费时间和金钱，最终得出的结果很可能是毫无意义的。尽管确切地定义问题的必要性是十分明显的，但是在实践中它却可能是最常被忽视的一个步骤。这里所说的问题，就是指用户的基本要求。说得通俗些，问题定义实际上就是了解用户到底要建立什么系统，并确定分析员下一步应该做什么。因此，问题定义的来源是用户。

通过问题定义阶段的工作，系统分析员应该提出关于问题性质、工程目标和规模的书面报告。这一阶段的分析员应尽可能站在较高的角度去抽象、概括所要干的事情，不要拘泥于问题实现的细节。尽管用户可能总是习惯于这样做，但分析员在这一阶段必须超脱出来，居高临下鸟瞰系统的全貌。通过对系统的实际用户和使用部门负责人的访问调查，分析员扼要地写出他对问题的理解，并在使用部门负责人的会议上认真讨论这份书面报告，澄清含糊不清的地方，改正理解不正确的地方，最后得出一份双方都满意的文档。

当用户的要求不是很多并且不太复杂时，一两个分析员用上一两天就可以完成这一工作了。但当系统比较大，且复杂时，恐怕就要组织一个问题定义小组，花上一两个星期，甚至数月来定义用户的问题。

如果分析员和用户及使用部门的负责人对所要解决的问题取得完全一致的看法，而且使用部门的负责人同意开发工程继续进行下去，那么开发工程将转入软件生命周期的下一个阶段——可行性研究。

2.2　可行性研究

并不是所有问题都有简单明显的解决办法，事实上，许多问题不能在预定的系统规模之内解决。如果问题没有可行的解，那么花费在这项开发工程上的任何时间、资源、人力和经费都是无谓的浪费。

可行性研究的目的就是用最小的代价在尽可能短的时间内确定问题是否能够解决。怎样达到这个目的呢?当然不能借主观猜想而只能借鉴已有经验，对待解决问题进行客观分析。必须分析几种主要的可能解法的利弊，从而判断原定的系统目标和规模是否现实，系统完成后

所能带来的效益是否大到值得投资开发这个系统的程度。因此，可行性研究实质上是进行一次大大压缩和简化了的系统分析和设计的过程，也就是在较高层次上以较抽象的方式进行的系统分析和设计的过程。

在可行性研究过程中，首先需要进一步分析和澄清问题定义。在问题定义阶段初步确定的规模和目标，如果是正确的就进一步加以肯定，如果有错误就应该及时改正，如果对目标系统有任何约束和限制，也必须把它们清楚地列出来。在澄清了问题定义之后，分析员应该写出系统的逻辑模型，然后从系统逻辑模型出发，探索若干种可供选择的主要解法，对每种解法都应该仔细研究它的可行性。一般说来，至少应该从下述五个方面研究每种解法的可行性：技术可行性、资金可行性、时间可行性、人员操作和维护的可行性、社会可行性。

1. 技术可行性

技术可行性所要考虑的问题是使用现有的技术能否实现这个系统。技术可行性工作一般可归纳为：

1）建立当前系统的物理模型；

2）抽象出当前系统的逻辑模型；

3）转化为可用计算机实现的新系统的逻辑模型；

4）落实到具体的新系统的物理模型；

5）分析新系统的物理模型关键技术的可解性。

以上五步工作实际上不仅为技术的可行性提供了依据，也为其他四方面的可行性提供了依据，我们把它放在技术可行性中介绍的目的，是由于这项工作为技术的可行性提供的依据最多最全面，这也是出于我国国情的考虑；当前国内要上马的系统，大部分都是先有一笔钱以后才进行系统分析和设计，而且有的项目时间已经确定，这样技术可行性就成为可行性研究中份量最重的一项，万一出了漏洞是会造成事故的，所以必须重点考虑。基于以上原因，技术可行性的问题就上升到了第一位。

2. 资金可行性

资金可行性所要考察的问题是这个系统的经济效益能否超过它的开发成本。在资金的可行性方面，应着重强调的一点是，我们研究的目的，并不仅仅是了解为完成用户提出的要求，是否有足够的资金支持（这是目前很多分析员重点做的事情），而更主要的是要把提高用户选择的每一种方案的投资利益比说清楚。因为聪明的用户并不只是看实现他们所要求的系统的绝对投资额，还要看这些钱花出去后能够收到多少利益。比如分析员为用户提供了三种可选（可解的）方案，第一种可满足用户最低的业务要求，第二种可满足用户中等业务要求，第三种可满足用户最高业务要求。它们的获利和投资利益比列于表 2-2-1。

表 2-2-1　成本/获利分析　　（万元）

	投资	获利总额	投资利益比
第一种方案	75	450	1∶6
第二种方案	140	1300	1∶9.3
第三种方案	220	2420	1∶11

用户根据分析员提供的这张表，再根据自己拥有的资金去选择系统，比单纯看系统的投资就有意义得多了。值得指出的是，其中的获利总额是根据系统的运行周期，乘其净增年利

润，再减去维护费用而得到的：

获利总额＝〔（系统运行周期×净增年利润）－维护费用〕×系数

其中系统运行周期按年计算，净增年利润是指使用新系统后平均每年比原系统能够增加的利润，维护费用是指在系统运行周期内，用于维护系统正常运转所需的费用。这种获利总额的估算，在可行性阶段可能与实际数值相差很大，因为系统尚未建立，不可能有很准确的答案。我们对净增年利润的估算只能在与用户的财务部门合作的前提下，对每一子系统所增长的利润进行估算，然后把它们相加，以此为资金可行性分析提供参考依据。

3. 时间可行性

时间可行性所要考虑的问题是，完成这样的一个系统所花的时间是否能够满足用户的要求？若经与技术和资金可行性紧密结合考虑之后，发现不能在用户预定的时间内完成系统，则要缩小项目规模，或采用分期实现的办法。时间的可行性是与投入的人力密切相关的，因此要从分析员和程序员的角度来研究时间可行性。

4. 人员操作和维护的可行性

人员操作和维护可行性所要考察的问题是当用户所要求的系统建立起来后，用户对它的操作是否感到方便，管理和维护是否容易。如果一个计算机系统的操作比原有的手工系统还麻烦，那么，它是不会受欢迎的。另一方面，如果管理和维护所开发的系统的人员数比原有的手工系统所需的还多，素质要求还高，那么这个系统对用户来说负担太重了。

5. 社会可行性

社会可行性是探讨法律方面和使用方面的可行性，诸如被开发软件的权利归属方面的问题。例如，要涉及到有关软件版权的争议时，就要考虑法律的有关规定以及可能带来的后果。自 80 年代初，我国先后制定了版权法、专利法、商标法、著作权法、计算机软件保护条例、反不正当竞争法等一系列知识产权保护条规。为与国际公约、惯例接轨，我国又对专利法和商标法进行了修改，制定了实施国际著作权等条约的规定。这些法律、条规都不同程度地对计算机软件这一客体实施保护，所以，在软件开发之前，要考察是否存在由于与法律、条规不相符造成的不良社会后果。

前面的讨论是从五个方面展开的，而实际上它们之间有着密切的联系，在分别研究之后，必须对它们综合考虑并向用户推荐一个较好的解决方案（若问题值得解的话），当用户选定方案后，分析员应对在问题定义阶段所规定的实现目标，即问题的定义进行修改，因为这时对系统有了更深入的了解，原来的问题定义可能有些不能实现，还有些要加上去，也就是说原有的问题边界不够准确，在这时要及时地把它纠正过来，以便于今后有一个非常明确的工作目标。

当以上的可行性研究工作做完之后，就应当着手进行最后一步工作：写可行性研究报告。该报告应包括以下内容：

1. 引言

1.1 编写目的

1.2 背景

1.3 定义

1.4 参考资料

2. 可行性研究的前提