鲁新登字 08 号

内 容 简 介

本书全面介绍了软件工程的概念、原理、方法和技巧。内容包括：软件工程概述，系统定义与软件计划，需求分析，软件的设计、实施、测试、维护和项目管理。

本书可作为“软件工程”课程的教材和各类软件培训班教材，也很适合管理人员、业务人员、计算机爱好者自学。

图书在版编目(CIP)数据

实用软件工程/陈彦仓编著. - 青岛：青岛出版社，2000.6
ISBN 7－5436－2216－5
Ⅰ. 实…
Ⅱ. 陈…
Ⅲ. 软件工程
Ⅳ. TP311.5

中国版本图书馆 CIP 数据核字（2000）第 05092 号

书　　名	实用软件工程
编 著 者	陈彦仓
出版发行	青岛出版社
社　　址	青岛市徐州路 77 号(266071)
邮购电话	(0532)5835124　5835844　5814750
责任编辑	樊建修　马　杰
特约编辑	邢　筠
装帧设计	申　尧
印　　刷	胶州市印刷厂
出版日期	2000 年 6 月第 1 版，2000 年 6 月第 1 次印刷
开　　本	16 开(787×1092 毫米)
印　　张	8.75
字　　数	195 千
印　　数	1－3000
ISBN 7－5436－2216－5/TP・291	
定　　价	16.50 元

前　言

随着信息技术的高速发展，计算机在各企事业单位逐步进入普及阶段，“无纸办公”、“电子商务”已开始走进我们的办公室，一种新的工作方式正在悄悄地改变着我们的生活，它将给我们的社会带来深刻的变革。面对这场迅速到来的信息革命，许多人陷入了困惑和迷惘，尤其是当人们怀着好奇的心情和新鲜的感觉坐在“观众席”上时，并未意识到这个今后对自己的工作方式将产生巨大影响的软件系统可能会质量低劣，甚至会把自己推向无休止的烦恼中去。

对于软件开发人员来说，能够认识到软件工程的重要性，掌握其原理、方法和技巧且能熟练运用的人并不多，且大量准备明天去做“程序员”的学生、电脑爱好者也只是把目光停留在学习一个又一个的语言上，至于用什么样的思路去开发程序的问题却很少考虑。事实上，所有这些问题都不同程度地影响了计算机的普及和发展，影响了信息技术的推广应用。随着 WTO 的加入，新世纪的来临，信息经济的迅速发展，世界和中国都需要更多的华人成为软件开发人员，那么一个软件人员在开发中的整体思路是什么呢？本书希望给出一个人人都能理解的、实用的答案。

本书的宗旨就是从实用的角度阐述软件工程的作用、原理、方法、技术以及实践经验，一方面为所有对计算机、信息技术感兴趣的人，尤其是本单位要实现计算机应用的人，全面、通俗地介绍软件工程，另一方面为软件开发人员提供一个开发依据和参考指南。

本书不涉及复杂的理论问题，以实用为出发点，以通俗易懂为前提，以实际工作经验为基础，以国内外资料为指导，力求文字精简，内容详实。

第 1 章　软件工程概述

1.1　软件危机

软件是计算机系统中的程序和有关文件。随着计算机的普及推广，软件的作用愈来愈重要，有人认为 21 世纪的两个生活必需品是石油和软件，但人类可以用其他能源代替石油，却无法找到软件的替代品。随着大规模集成电路的高速发展，计算机硬件性能/价格比不断提高，而软件的开发速度和性能却远远不能适应这种发展和变化，软件开发费用不断上涨，质量得不到保证，维护工作跟不上去等问题困扰着人类。我们把在软件开发和维护中所遇到的这些问题统称为“软件危机”。

(1) 软件危机常见的表现

① 软件失败。软件中一个极小的错误也可能导致运行的失败，如无人驾驶飞机冲入地面而爆炸。

② 开发失败。由于各种因素，造成开发工作的中断，永远无法完成；有些项目事实上已经失败，但出于其他考虑，仍然凑齐了各类文档和代码，表面上已开发成功，甚至召开了鉴定会，但开完鉴定会，软件就废弃不用了，鉴定日成了软件失败日。

③ 开发周期严重推迟。相当多的项目都存在实际开发周期比预期时间要长的问题，大型项目推迟一年的情况很普遍，甚至有些小型项目的开发时间也比合同预定时间延长了一倍以上，而在推迟期内，用户的需求和环境的不断变化，使得对软件的修改要求也在不断增加，形成了“愈干，活愈多”的局面。

④ 用户对产品不满意。开发初期，用户对未来产品充满了信心和热情，但当拿到最终产品时，却发现与想象的有一定差距，或者不能满足自己的实际需要。

⑤ 成本过高。包括开发成本过高和维护费用过高，并且它们还在不断增长(其中维护费用要占一半以上)。

⑥ 软件质量不稳定。除了功能的适用外，软件还应具备相当的可靠性、安全性、可维护性、可用性和反应速度较快等特点，这些非功能性质量要求对软件的整体质量有很大影响。

⑦ 软件供不应求。软件的质量和生产率问题实际上已形成了计算机应用的“瓶颈”，市场对各类软件有着极大的需求，但软件生产却严重滞后。

⑧ 软件寿命太短。除了版本更新等原因外，缺乏维护而造成运行的结束是主要原因。

尽管用户或开发人员都不愿承认也不愿谈论自己软件中的问题，但这些问题是广泛存在的，并且随着硬件的飞速发展、网络的普及，软件业似乎总也跟不上用户的要求以及硬件的发展，甚至有人认为，一个软件从业人员要用 1/3 的时间学习，1/3 的时间实践，剩下的 1/3 时间才能用于真正的软件开发，可见上述问题的存在严重地制约了软件产业的发展，因此有必要剖析软件危机的原因，找出相应的对策。

(2) 导致软件危机的原因

软件危机的产生既是由于软件本身的特点，又是由于开发方法中存在着问题。具体表现为：

① 软件的规模愈来愈大，复杂度愈来愈高，用户需求在不断地增加并随时变化。可以说任何软件都无法避免变动，而每次变动又面临着带入错误的危险等情况，所有这些问题都可能导致软件危机的发生。同时在开发中，一些错误的认识或方法把复杂的软件开发工作搞得更复杂了，甚至使开发工作偏离了正确轨道，这些问题的普遍存在，也是软件危机的主要原因。

② 缺乏必要的分析。软件就是思想，它的发展过程是客观存在的，我们把它从定义、开发、维护到死亡称为软件的生命周期，程序代码仅仅是这个生命周期中的某个阶段的表现形式。但在软件开发中有不少人没有认真分析软件发展过程中各个时期的特点，一步就进入了编程，其结果只能是设计出质量低劣的软件。

③ 维护工作不到位。维护是软件开发中投入较大的一个环节，但很多软件的结构很僵硬而无法维护，另一些软件又因不重视维护，缺乏文档和维护记录，导致新的错误不断进入系统。

④ 缺乏有效的组织管理。开发中各自为政，开发方法不一，标准不同，术语混乱，接口错误。

⑤ 用户与开发人员之间存在鸿沟。双方都难以理解对方的术语、思路和问题。

⑥ 用户确认工作流于形式。确认的目的是希望由用户来发现问题，但实际情况往往是用户稀里糊涂就签了字。

⑦ 开发人员水平参差不齐、人员变动频繁。开发人员经验、水平的差别，将导致质量的不稳定和进度的拖延；而人员的频繁变动，又使软件的完成变得遥遥无期。

综上所述，软件危机是客观存在的，它不会消失，但也不是不可救药。从软件发展的规律出发，吸收和借鉴几十年来的开发经验，运用各种工具，采取各种管理技术和开发技术，用系统工程的思想方法从宏观上进行控制，则是解决软件危机的有效途径。我们把这一新兴学科叫做软件工程。

1.2 软件工程

把系统工程的概念、原理、技术方法用于软件开发，把相应的组织管理技术和软件开发技术结合起来，这就是软件工程。

软件工程以软件的生长周期为前提，采用各种方法、工具，用工程的方法降低软件开发的复杂度和开发成本，提高软件质量。

(1) 生命周期法

生命周期法把软件开发过程分为系统定义和软件计划、需求分析、总体设计、详细设计、实施、测试、维护几个阶段。每个阶段的工作明确而且相对独立，使整个过程图解化、逻辑化。每个阶段的工作均以前一阶段为基础，工作结束时，把本阶段工作整理成文档，然后进入到下一阶段。一般来说，前期发生的差错，留到后期再去纠正，需要花费的代价更大，如果在进入下一阶段之前把住质量关，则是提高软件质量的有效途径，一般把这种分阶段方法叫做瀑布型生命周期法，见图 1.1。

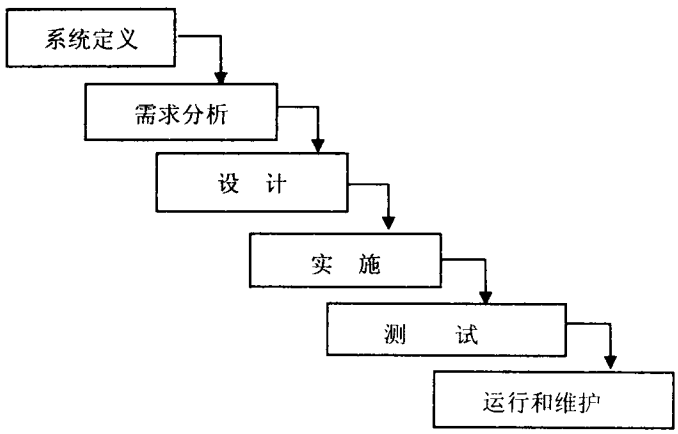

图 1.1　瀑布型生命周期法

为尽早发现错误，每个阶段工作完成后都要进行技术审查、管理复审、用户确认几项工作。技术审查实际上是对本阶段的工作进行测试，由技术审查测试小组负责。这个小组主要成员应是有经验、水平较高的技术骨干，但最好不要有开发小组人员参加，审查测试的目的就是发现本阶段文档或代码中的错误，交给开发小组返工，返工后再审查。见图 1.2。

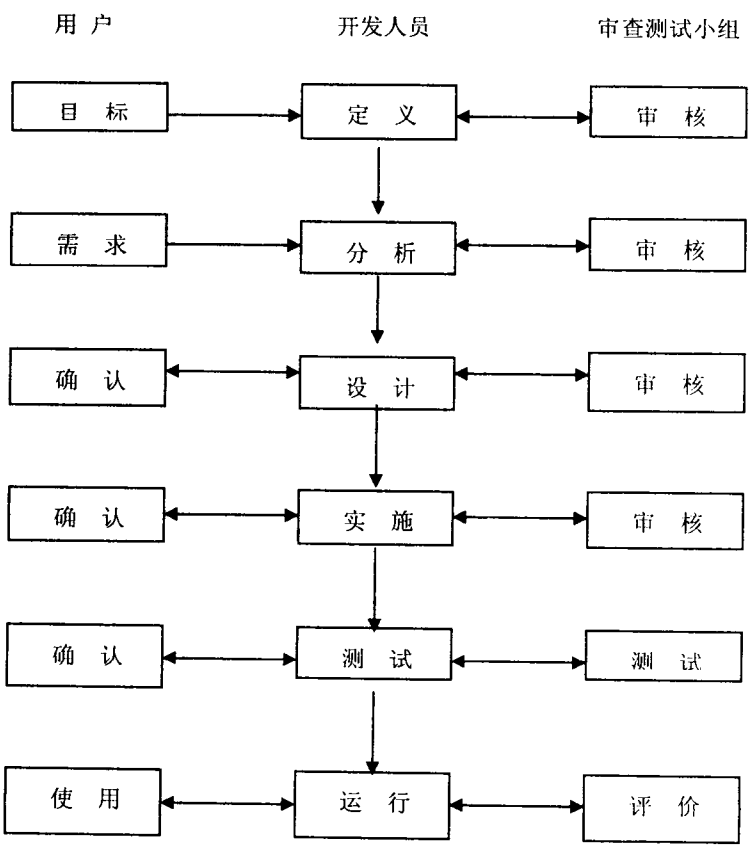

图 1.2　开发流程示意图

管理复审的工作是在本阶段结束后，对项目的成本进度、人员情况、环境要求等从管理角度进行审定，这项工作由开发小组负责人和用户的高层管理人员共同完成。

用户确定是重要的把关环节，本阶段的成果是否满足用户的要求，是否存在错误，是否存在理解上的不一致，均需经过用户审查，见图 1.2，一旦用户确认便可进入下一阶段。但是由于软件开发的前期工作结果均为文档，用户对其内容很难掌握，往往不知所措，使得用户确认拖延时间，最后草率签字，而问题却没有找出来，这是个通病，是生命周期法的一个弱点。

在问题复杂度方面存在着一条规律：若问题 A=B+C，函数 f(x)表示复杂度，则 f(A)>f(B)+f(C)，即把大问题分解成小问题，整个的复杂度降低了，显然生命周期法的一大优点就是降低了软件开发的难度。

生命周期法的另一个优点是使质量控制有了依据。每个阶段工作完成后都要以文档报告的形式总结本阶段的工作成果，并经过技术审查、管理复审和用户确认，的确找不出错误后，才能进入下一个阶段。

生命周期法中对各个阶段的划分是不统一的，有很多种划分方法，基本上大同小异，在一个具体的开发中，采用哪一种方法、哪一种划分并不重要，因为每种划分都是极力反映生命周期的不同特点，过分地拘泥于某种划分方法或无限制地比较评价方法间的差异，都会误入歧途。

(2) 原型法

在生命周期法中，当用户发现最终产品存在着与自己的需求不一致的问题时，往往需要花费较大的代价才能修正，为此近年来出现了一种尽早与用户见面的开发方法——原型法。在原型法中，要求在系统实现前，首先建立一个原型，交给用户去评价，由此发现问题，然后再改进，再评价……直到满意为止。见图 1.3。

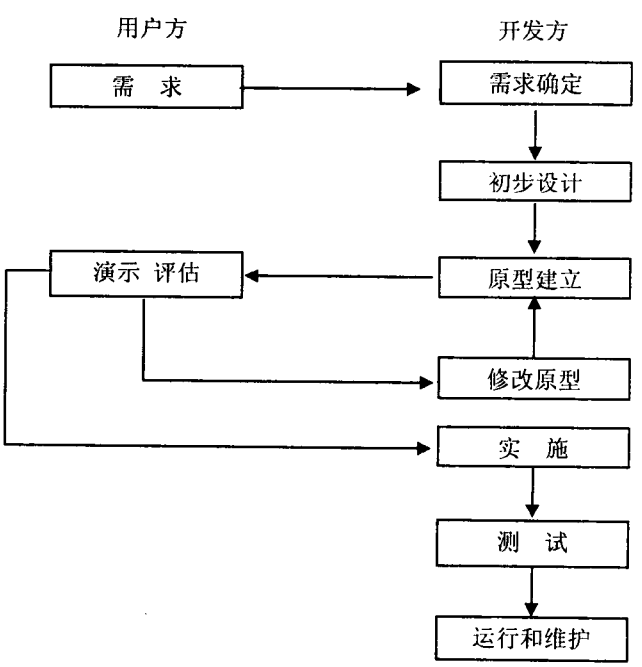

图 1.3 原型法开发方法

在原型法中，用户一方面可以及早发现不满意的地方和存在的问题，另一方面因为用户介入了开发过程，所以对产品的信心、满意度和喜爱程度都比较高。一般原型法分为以下几个步骤：

① 用户需求的最初确定。

② 系统的初始设计。

③ 原型的实现、评估、修改(循环过程)。

④ 系统开发。

⑤ 系统安装及检测。

⑥ 系统的维护和发展。

原型法与生命周期法的主要区别在于用户需求和设计阶段的开发方法不一样。生命周期法是从结构和微观出发把问题的所在和解决方案搞出来；而原型法是在用户提出要求后，大致了解一下用户需求，简单地进行初始设计，就宏观地用快速编程工具实现一个原型，交给用户去使用，评价原型的功能、性能和指标。这个快速编程工具一般是第四代语言、应用生成器或特殊原型工具。如果原型不满足用户的期望，用户提出意见，开发人员修改原型或增加功能，再交给用户，循环往复，直到用户认为满意为止。这时的原型就变成了一个动态的用户需求报告和系统设计报告，后面的实施可以在此原型基础上实现，亦可以放弃这个原型，而用另一种编程语言来实现，后续的各阶段工作与生命周期法差不多。

原型法实际上是用不同的途径去实现生命周期，它最大的特点就是能动性，可以准确、完整地搞清用户的需求。因为原型法并不是完美无缺的，在使用原型法时，一定要避免原型法自身带来的一些问题。由于原型自身就有需求说明、设计报告的作用，因此往往在使用中忽略了应有的系统分析和设计，这是很危险的，因为用户仅仅从使用角度提出意见和看法，但是深层的全局性问题往往提不出来而被忽略，这样最终产品将处于一个不断变动的昂贵的维护状态中。

在原型法中，文档同样是需要的，它可以帮助我们发现问题，为后继工作提供较完善的依据，尤其是在维护阶段，没有前期工作的文档支持，维护工作是很难进行的。

原型提交给用户，其作用就是演示、评价，其功能、工作负荷能力均不同于今后的实际系统，一旦到了实施阶段，原来在原型中已得到用户认可的功能可能会变得性能下降或无法实现。

原型的循环次数是不定的，只有在用户认可的时候才能停止，可是在实际运用中，用户往往不知道原型何时可以停止，他可以指出原型中还缺少什么功能，但难以确定原型已不缺少任何功能，结果就使得原型无限制地重复循环下去。

在生命周期法中，每进行一步都有相应的审查，但在原型法中，从原型的建立到原型的停止，实际上覆盖了用户需求、总体设计、详细设计几个阶段，原型结束了，这几个阶段的工作就完成了，但是其范围全面与内容正确与否均无法控制。

原型法是一种用户驱动的方法，如果用户的积极性不高或缺乏计算机知识，循环就根本进行不下去。用户的热情一般来说是先高后低的，而且热情一过，他们有充足的理由去远离计算机。由于国内计算机的普及教育程度还远远不够，使原型法的推广难度相对大得多。但是用户的参与，无论在哪种开发方法中都是极其重要的。

虽然原型法存在着缺点，但是原型法的灵活性和极强的能动性，对任何人都是一个很大的诱惑。只要妥善进行好项目的成本结算工作、资源管理工作、用户培训工作，并与生命周期法的成功经验和技术相结合，原型法的作用就会发挥得更好。

(3) 死亡周期法

死亡周期法并不是一种开发方法，它只是一个模型或概念，对生命周期法或原型法的实施做一个注解。

软件不死或没有磨损是很令人鼓舞的事，但是软件的变动或维护是不可避免的，而这个维护的代价又是巨大的，它随着时间的推移、设备平台的淘汰、人员的流动和技术的陈旧，其维护费用会急剧增加，见图 1.4。当软件的收益与运行维护费用相抵消时，即为软件的死亡点。任何一个软件，不管用什么方法开发，都可分为开发期、运行期、死亡期，开发期只有开发费用的投入，收益为负数，运行期有软件的收益和运行维护的费用，这两者相减形成实际收益线，当实际收益线取值为零时，进入死亡期。正确认识软件的这个变化规律，对项目的投资、效益分析等都是十分有益的。

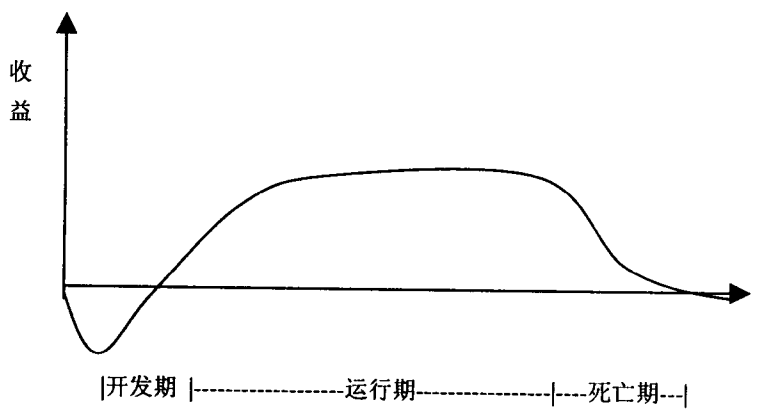

图 1.4 软件的死亡周期

1.3 软件工程的特点

软件工程的目标是用工程的技术和方法来指导软件开发过程，把整个过程分为若干个阶段，把原来个人艺术创造式的编程活动纳入有组织、有计划的工程开发轨道，尽早、尽快地发现各类问题和错误，保证软件质量，并通过资源管理技术降低开发成本。软件工程的应用，有效地解决了软件危机给人类带来的困惑。

软件工程主要依据以下几个规则：

(1) 分解

把一个大的、复杂的问题分解为小的简单的问题，从而降低问题的复杂度。把开发周期分为不同的阶段就体现了这一点。

(2) 分层

把问题按一定的原则分为不同的层次，高层在低层的基础上实现，低层为高层提供接口。这样不仅简化了系统的设计和实现，还减少了错误的发生机会。

(3) 统一性

遵循统一性的要求，使阶段之间、开发人员之间共同遵守统一的方法、技术、手段、术语和文档格式，这是软件成功的重要保证，也是分解、分层后系统集成的前提条件。

(4) 确定性

把用户的需要用明确的表达形式描述出来，使之变成一个结构化的、过程化的数据，对非功能性的要求如性能、易用性、安全性等也要用确切的数字表达出来。

(5) 包容性和系统性

软件工程包容了最新的管理技术、各类开发方法和开发工具，在系统工程的思想方法指导下，把复杂的软件开发——思想表达问题，变成可控制的工程化问题，它是从宏观上、方法上去解决问题，这也是软件工程的精髓所在。

软件工程还在不断发展，它的现状还不能为我们提供一个完全结构化和过程化的方法，这一方面是技术发展的问题，另一方面是软件的复杂度和用户需求的复杂度问题。

有些成功的软件，尤其是个人开发的软件，并未使用成熟的软件工程方法，但最终的软件质量却很好，而有些开发小组采用软件工程的技术方法，最终仍得到了质量低劣的产品。所有这些现象并不能抹杀软件工程的积极作用，只能说明软件开发、软件工程的特点："天下没有无敌的功夫，却有无敌的拳师"。开发方法仅提供了一个手段，它不能保证一定出高质量的产品，因为起主观能动作用的是人，而不是方法，只有积极遵循软件工程的原则、方法，最大限度发挥人的创造性，并把这个创造性与工程化要求融合到一起，才有可能创造出高质量的软件。相反，不采用软件工程的方法或违背软件工程、软件开发的规律，其结果就像在沙滩上盖高楼一样，最终得到的只能是一堆废瓦砾。

思考题

1. 什么是软件危机？常见的表现有什么？
2. 什么是软件工程？作用是什么？
3. 什么是生命周期法？有什么特点？
4. 什么是原型法？有什么特点？
5. 软件开发中主要有哪几方面的人员参加？
6. 用问题复杂度规律解释一件事情。
7. 软件何时死亡？
8. 软件工程的特点是什么？

第 2 章　系统定义与软件计划

2.1　本阶段工作内容

本阶段是生命周期法的第一步，它是整个过程的基础和出发点，主要说明用户的目标是什么，现行系统是个什么样的系统，有哪几种解决办法，是否可行，如何安排各种资源和进度。

如图 2.1 所示，系统定义主要包括系统分析和可行性分析，在理解最初要求和问题的基础上，对现行系统进行论证、评价，提出一个全面的、有逻辑的系统规格书；而可行性分析则是对现行系统分析、设计的前提下，考察各种方案，评价其可行性，提出可行性分析报告。

软件计划是对人力资源、设备资源、进度等做统一安排和部署。

系统定义工作、可行性分析、系统软件计划可由双方共同完成。

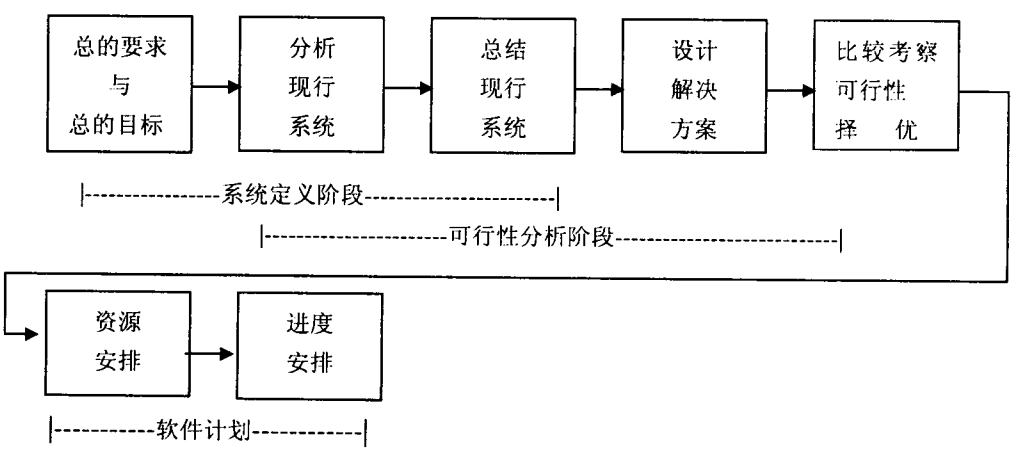

图 2.1　系统定义和软件计划

2.2　系统定义

当用户向开发人员提出他需要一个计算机系统时，他已经有了一个系统总体目标，不论这个目标仅仅是停留在口头上还是已经成文。这个目标往往是很粗糙、很笼统的，甚至互相矛盾，以此为基础进行分析设计将是危险的，只有以用户方为主，开发方参与，对这个目标要求进行系统地、详细地、有逻辑地归纳整理，才能形成系统规格说明书，这是系统定义工作的核心，也是后续分析设计工作的依据。

(1) 对系统总体目标要求的理解和复审

准确理解总体目标要求，了解提出要求的原因和背景，检查有无歧义性、逻辑错误和重复，用条理清楚的准确文字重新把总体目标要求整理成文。

(2) 调查现行系统

这个调查不要求全面、详细，但要从整体功能、要求和性能上进行，可以用户方的高层管理人员、中层管理人员和主要业务骨干为调查对象，方式为单独交谈、开会或发调查表。

(3) 现行系统的运行环境分析

任何一个信息系统的运行，与其环境都有着不可分割的联系，要用计算机取代现有的信息处理方式，必须对现行系统做一个环境分析，包括国家的政策法规、行业规定、管理制度、发展趋势和本单位环境等。

(4) 现行系统的运行情况

包括现行系统从什么时候开始运行，运行的费用，有过什么事故，反应速度如何，可靠性如何，用户使用评价如何。

(5) 现行系统存在的问题

分析现行系统是否有继续存在的必要，它主要在哪些方面不能适应要求。

(6) 现行系统的规模及分布情况

搞清现行系统的用户数有多少个，信息处理点有多少个和分布情况，可预测的系统规模扩充程度。

(7) 现行系统的功能及要求

分析现行系统主要实现了哪些功能，还需要完善哪些功能，在性能方面有什么要求。这部分的工作要结合总体目标要求来进行。一般来说，现行系统的功能，在总体目标要求中都应提到，即今后的系统规格说明书中应包括现行系统已实现功能以及现行系统无法实现的功能和性能。

(8) 现行系统的数据分析

包括存储数据分析和流动数据分析。存储数据是指各类记录、存档、流水账、台账、卡片等；流动数据是指各类报表、单据等。在这项工作中，既要统计和估算数据量的大小，又要分析各类数据的结构。

(9) 领导及主管部门的态度

软件开发不仅是个复杂度很高的工作，而且要牵涉到各个方面的工作，对所有人员、工作方式、业务环境都有极大的冲击，所以得到领导的支持、主管部门的认可是非常重要的，否则会出现非技术因素导致的开发失败。

(10) 业务人员态度

今后使用软件的业务人员或操作员对软件和软件工程的看法也必须引起注意，无论何种开发方法均需此类用户的介入，而且介入越多越好。如果业务人员对此很有信心，工作就会开展得很顺利；如果业务人员对此持怀疑、否定的态度或害怕计算机取代他的位置，开发人员就将面临一个极其困难的处境，软件质量将被打折扣。

(11) 用户素质

用户素质可从用户的综合素质和用户在计算机方面的素质两个方面分析。综合素质包括学历、工作能力、智商情况等，计算机方面的素质指用户是否受过计算机方面的教育、是否用过计算机和对计算机的爱好程度如何。显然一个综合素质高的用户，即使从未接触过计算机，也可经过培训达到要求；若用户的综合素质不太高，但计算机素质较高的话，也毫无问题；只有当用户综合素质不高，对计算机又毫无兴趣时，才会对今后的使用构成障碍。

(12) 用户方技术负责人分析

用户方技术负责人负担着与开发人员协调、沟通、技术方案选择、复审确认等重要任务，对软件的质量起着非常重要的作用。因此要求这个人必须有相当的实际开发经验和理论水平，最好能接近、达到或超过开发小组的水平。但在实际中往往是由一个经验水平一般或非专业人员担任，这就造成了开发中的进度缓慢、资源浪费、工作不到位现象。

(13) 开发人员分析

对开发小组的人员是求精不求多，随着项目规模的变化，开发小组可以有一个，可以有多个，实现分工协作。每个小组一般 3~5 人，设立小组长负责全面工作。开发小组中人员的素质水平不要相差太大，小组长要有较高的开发经验和水平，在系统分析和系统设计方面有出色的表现。

(14) 审查测试人员分析

审查测试小组中不应有开发小组人员参加，这样可以最大限度地发现问题。审查测试小组人员要有较高的理论水平和丰富的实践经验，如果开发方无法提供这样的队伍，可另行组织人员。

(15) 问题结构化程度分析

软件需要解决结构化、半结构化和非结构化三类问题。结构化问题是指可用形式化语言描述过程算法的问题；非结构化则反之；半结构化处于两者之间。显然，一个问题的结构化程度愈高，那么今后的分析、设计和实现就会愈容易，而且软件的质量也易于保证；若结构化程度低，那么难度就要大得多。

(16) 技术、设备环境分析

搞清现有的技术水平、设备情况能提供什么样的支持，软件平台有哪些，数据库平台有哪些，硬件可选范围。

(17) 同类项目经验

了解同类项目采用了何种方案，达到了什么水平，有哪些成功的经验或失败的教训。

(18) 项目的约束条件

包括项目要求的开发时间、开发资金和开发策略，是一次开发还是分期开发，项目的性能要求(包括可靠性、反应时间、可维护性、可用性等)，用户的培训条件，项目的风险。所有这些都要在系统定义中完成，最后把这些工作成果归纳整理，形成系统规格说明书。

2.3　系统规格说明书

系统规格说明书是用户需求的第一个正式文本，它是总体目标要求的详细体现，但它仍然是粗线条的，主要有以下内容：

① 系统的目标。
② 系统的环境。
③ 系统的范畴。
④ 系统的功能要求，包括输入、输出、处理。
⑤ 系统与其他部分的接口要求。
⑥ 系统的非功能性要求。
⑦ 约束条件。
⑧ 特殊要求。

2.4　系统可行性分析

当一个系统的目标确定后，首先要回答以下问题：这个问题有解吗，有几种方案，哪个方案最好。这时不必考虑怎样去做，只需考虑能否做，有无办法去做。如果方案不可行，那么后续工作便可停止，从而避免损失。

系统的可行性一般从经济、技术、环境三个方面分析。首先必须对现行系统进行分析并提出几个设计方案，然后比较选择，见图 2.1。实际上这是一次压缩了的分析、设计过程，这个分析和设计是在逻辑层上进行的，它的工作量可能不大，但涉及的信息种类繁多，表现为局部信息少，整体和外部环境信息多。

从图 2.1 可看到，可行性分析与系统定义有一定的交叉关系，它是在系统定义时所做的系统分析的基础上，进行多个方案的设计和评价选择，考虑其可行性的情况。对大型项目，在此期间可以使用系统流程图、数据流图、数据字典等工具来描述。

在系统定义时对现行系统的分析就是可行性分析的一个重要步骤，通过这阶段的工作，可以了解：现行系统的目标，现行系统的环境和发展历史，现行系统的功能及缺点，对新系统的要求及期望值，现行系统与环境的接口关系等等。要注意，这个现行系统的分析工作是高层次的，它不是详细的分析，只要求准确地把握住现行系统的状况就可以了。

在研究现行系统的基础上，会出现行系统的逻辑模型，即对现行系统做一个总结，进行高度的抽象概括，把系统的实质性问题抽取出来，形成与实际环境无关的逻辑模型，这个逻辑模型的表达可以用各种图表工具，也可以用简练概要的文字来描述。然后据此推出新系统的逻辑模型，并会同用户方有关人员检查这个目标是否满足了系统规格说明书的要求，有无遗漏，有无误解。

确认新系统逻辑模型后，可据此提出几种解决问题的方案。方案的提出，可以从经济角度考虑，提出高投入大规模方案、中投入中规模方案、小投入小规模方案；也可从技术角度出发，提出几种不同技术成分的方案；有时还需要结合硬件平台的情况来提出几种方案，

如网络式平台、小型机平台、高性能工作站平台等。当不同的实现方案做出后，便逐一分析它们的技术可行性、经济可行性和环境可行性。

(1) 技术可行性分析

① 主要分析这个方案所需的技术能否得到支持，这些技术是否成熟，是不是最佳方案，它能保持领先地位多少年，会不会很快被淘汰，与下一代技术能否兼容，有无成功的先例，采用这种技术对系统的响应时间、可靠性、安全性方面会有什么样的影响等等。

② 分析开发人员的力量和水平能否保证项目的顺利实施，审查测试小组能否把住质量关，用户方技术负责人的水平及能力如何。

③ 分析方案的技术局限是什么，它的缺陷是什么，对项目会有什么不利的影响，方案是冒险的还是稳妥的。

④ 分析设备情况，既要考虑设备的先进性、兼容性、存储容量、运行速度、可靠性、网络通信能力等，又要考虑软件平台的先进性、兼容性及支持能力。

通过上述问题的分析，把无法满足技术要求的方案去掉，剩下的方案分别说明每个方面的满足程度，并进行比较、排队。

(2) 环境可行性分析

① 分析这个方案的实施对国家、社会这个大环境有哪些积极影响，有哪些消极影响，是否违反法律或行业规定，社会对它会不会认同，是否存在版权纠纷等。

② 分析这个方案对本单位的规章管理制度有无消极影响。

③ 分析方案的操作方式是否能被业务人员接受，与旧的工作方式相比有哪些优点，有哪些缺点，改进大不大等等。

把无法满足环境可行性的方案去掉，并把其余方案按其技术可行性和环境可行性的满足程度进行排队。

(3) 经济可行性分析

① 分析方案的投资总额为多少，能否提供足够的经费支持。如果所需的投资得不到保证，那么这个方案的经济可行性便得不到满足；如果经费可以得到保证，下一步就是要考虑方案的市场前景、市场定位，这是定性的分析，主要考虑同类产品的现在情况、将来发展，市场是饱和还是空白状态，需求量大不大。如果方案在这些方面有乐观的保证，便可进行定量的分析了。

② 关于定量的成本、效益分析的方法，详见下节。

对每个方案进行经济可行性分析并淘汰后，剩下的按照技术可行性、环境可行性、经济可行性综合考虑，择优推荐，得出本阶段工作的关键性结论：本项目可行吗？

如果可行，介绍最佳方案，说明这个方案的优点，提出这个方案实施中的注意事项，给出相应的建议。最后，把本阶段工作成果整理成文，形成可行性分析报告。

2.5 成本/效益分析

在软件开发中，首先要经过开发期的投资，才能进入运行期，并在此期间获得收益，见图 1.4。当达到某个时间点后，系统死亡。从经济角度看，收益的总和若小于投入的总和，无论技术多先进也是失败的。

成本/效益分析主要分析总成本和总效益两个方面。

2.5.1　总成本分析

总成本包括开发成本和运行成本。

① 开发成本包括：

- 人员工资，如软件人员、审查测试人员、管理人员、操作人员的工资。
- 设备投资，如主机设备、外设、其他附属设备、网络设备、设备安装及调试的投资。
- 易耗品消耗，如磁盘、打印纸等的消耗。
- 管理费、办公费。
- 人员培训费。

② 运行成本包括：

- 硬件维修费。
- 人员工资，如操作员、程序员、业务人员。
- 易耗品消耗。
- 管理办公费。
- 软件维护费。

从图 1.4 中可看出，开发成本是近期投入，运行成本是陆续投入，直到系统死亡时停止。对于人员工资的计算，主要涉及到开发人员的工资难以计算，一般是根据程序量的大小、开发的速度和工资水平来决定。

对程序量的估计可用代码行技术，它首先估算出程序的总代码行数，再用一个平均日开发行数去除，得到所需的开发总时间，总时间乘以平均工资，便是软件开发人员的工资；也可在估算出程序总行数之后，乘以每行平均价格，即为软件的开发人员工资。例如一个程序的总行数估计为 26000 行，假定每行价格为人民币 5 元，那么软件开发人员的工资应为：26000×5=180000(元)。

当估计总行数有一定困难时，可以采用任务分解法来估计，方法为把整个开发按阶段、系统、难易程度分为不同的工作单元，对每个工作单元标明(估计出)它的工作量大小和难度系数，参考当前水平，确定出开发一个中等工作量中等难度的工作单元的费用，进行计算并累加，得到总的任务成本，见表 2.1。

表 2.1　工作单元成本计算表

工作单元	工作量	难度系数	工作单元	工作量	难度系数
1	1.5	1.0	5	0.5	0.5
2	1.0	1.2	6	1.8	1.2
3	2.2	1.5	7	3.0	1.0
4	1.5	2.0	8	2.0	1.5

如果当前的开发费用水平为 8000 元/工作单元(1.0 难度，1.0 工作量)，那么总成本为：

$$(1.5\times1+1.0\times1.2+2.2\times1.5+1.5\times2.0+0.5\times0.5+1.8\times1.2+3.0\times1.0+2.0\times1.5)\times8000=139440\text{ 元}$$

如果用户方与开发方已有合同，软件人员工资成本就按合同开发费来计算。

2.5.2　总效益分析

总效益包括直接经济效益、间接经济效益和不可预测的经济效益。

① 直接经济效益包括：

- 收益增加，如销售额增加，利润增长。
- 成本减少，如节省了原材料、能源耗费、人力物力等。

② 间接经济效益包括：

- 可计算间接经济效益。如使用计算机后，库存资金占用减少，那么这个间接效益库存(库存占用所产生的利息)是可以计算的。
- 不可计算间接经济效益。如使用计算机后，企业的管理水平和产品质量提高了，企业形象和产品品牌形象在市场中的价值迅速升值，这时计算机在这个无形资产升值中的作用是不可否认的，但也是难以计算的。
- 不可预测经济效益是指非经常性的、一次性的、计划外的收益。

2.5.3 成本/效益分析

在搞清总成本与总效益之后，便可进行成本/效益分析，一般从以下几个角度来分析。

(1) 纯收入

即计算全部效益与全部成本之差。若大于零，说明没有损失；若小于零，说明有损失，此项目不能投资；若等于零，说明既无损失也无收益，没有投资价值。只有当纯收入较大或占到全部投资的一定百分比时，才能认为项目具有投资价值。

在纯收入计算中，要估计软件的运行时间，它对计算起着重要的作用。由于计算机的发展速度极快，更新换代周期很短，按 5 年考虑比较符合实际情况。

在实际计算中，还要考虑资金的贴现问题。由于利息的原因，现在的 1 万元与 5 年后的 1 万元其价值是不等的。计算现在投资的将来价值公式为：

$$F=P\times(1+I)^N$$

其中，P 为现在投资，F 为将来价值，I 为利息率。假定 I ＝0.1，那么，现在的 1 万元，相当于 5 年后的：

$$10000\times(1+0.1)^5=16105.1\text{ 元}$$

那么，5 年后的 1 万元折合成现在是多少呢？这就是贴现。公式可从上面推出：

$$P=F/(1+I)^N$$

仍假定 I ＝0.1，那么：

$$P=1/(1+0.1)^5=6209.21\text{ 元}$$

在纯收入计算中，要将今后每年的收益逐年进行贴现，按当前价值比较成本与效益。若某项目在开发期(当年)投资 8 万元，运行期为 5 年，每年的效益与运行成本情况如下：

表 2.2 纯收入计算表

i=0.1　　单位：万元

年	效　益	运行维护费	(1+i)^n	效益贴现	运行贴现
0	0.0	0.0	1	共 17.281	共 4.753215
1	2.0	1.0	1.1	1.818182	0.909091
2	3.0	1.0	1.21	2.479339	0.826446
3	6.0	1.0	1.331	4.507889	0.751315
4	6.5	1.5	1.4641	4.439587	1.024520
5	6.5	2.0	1.61051	4.035989	1.241843

由表 2.2 可见，整个项目为投资 8 万元，总效益贴现为 17.280986，纯收入为：

17.280986 – 8 – 4.753215=4.527771 万元

纯收入占总投资的 56.5%。有了纯收入和这个百分比，便可以实现各方案间的比较了。

(2) 资金回收周期

资金回收周期是指累计效益达到投资额时的时间周期。虽然资金回收周期愈短愈好，但是资金回收周期之后的时间基本相当于项目的净收入期。资金回收周期若接近预期系统生存周期(例如 5 年)，则项目没有投资价值。

在上例中，三年的效益贴现之和为 8.8054 万元，而三年的运行维护费贴现之和为 2.486852 万元，加上开发投资 8 万元，为 10.486852 万元，显然还有 1.681442 万元尚未收回，设第四年的 a 时能收回投资，那么 1.02452×a 为第四年回收期内的运行维护费贴现数，显然有：

1.681442+1.02452×a=4.439587×a

计算可得：a=0.49，即资金回收周期为 3.49 年。

(3) 投资回收率

软件开发的投资行为是这样的：一次性把开发成本投进去，然后逐年收回一定效益，到生命周期结束时停止。如果同时把等于开发成本的另一笔资金存入一个特殊银行，每年从银行取出一笔钱，钱数与这年的效益相同，到运行周期结束时，银行就不再给利息了，并且本也不归还(特殊银行)，这样，年利率相当于多少呢？我们把这个年利率叫做投资回收率，年利率可从以下方程中解出：

T=F1/(1+i)+F2/(1+I)^2+…+Fn/(1+i)^n

其中，T 为投资额，F1 为第一年的效益，Fn 为第几年的效益，i 为投资回收率。

2.6　可行性分析报告

把本阶段的工作内容进行整理，形成可行性分析报告。报告的格式可参考 GB8567-88 来编写，也可根据实际情况进行增减，格式如下：

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
- 2.1 要求
- 2.2 目标
- 2.3 条件、假定和限制
- 2.4 进行可行性研究的方法
- 2.5 评价尺度

3　对现有系统的分析
- 3.1 数据流程和处理流程
- 3.2 工作负荷