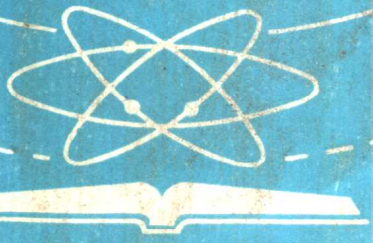
高等学校教材

软件工程概论

上海工业大学

孙振飞 应振澍编著



湖南科学技术出版社

软件工程概论

上海工业大学

孙振飞　应振澍编著

湖南科学技术出版社

内 容 提 要

本书是根据全国计算机与自动控制教材编审委员会审定的“软件工程概论”教学大纲而编写的。全书深入浅出，内容丰富，系统地介绍了软件工程的各个方面。该书既可作为计算机科学和工程类专业学生的教材，也可作为系统分析员和程序设计人员的重要参考读物。

高等学校教材

软件工程概论

上海工业大学

孙振飞　应振澍编著

责任编辑：古　华

*

湖南科学技术出版社出版

（长沙市展览馆路8号）

湖南省新华书店发行　湖南省新华印刷二厂印刷

*

1987年8月第1版第1次印刷

开本：787×1092毫米 1/16　印张：11.75　字数：291,000

印数：1—7,200

ISBN 7—5357—0104—3／TP·1

统一书号：15204·195　定价：2.90元

87年秋理工(3123—8)

前 言

软件工程是计算机科学的一个重要分支，软件工程通常包括软件开发方法学、软件经济学、软件管理和支撑环境等方面的内容，即通过改进程序结构、革新软件生产工具和生产方式，以及关于软件可靠性技术的研究，更有效更经济地开发软件产品，有力地促进软件生产的工程化和软件产品的商品化。

本书是根据全国计算机与自动控制教材编审委员会审定的"软件工程概论"教学大纲进行编写的。全书共分八章。第一章概要介绍计算机系统的开发过程，软件开发的前期工作和依据，以及软件生命周期等一系列软件工程的基本概念；第二章讲述软件计划，着重介绍软件估算模型和估算方法；第三章讲述软件需求分析，主要介绍数据流图和结构分析法；第四章介绍了按数据流的设计方法和数据结构设计的Jackson方法，并引用实例给出了具体的设计步骤，同时对LCP法也作了相应的讨论；第五章着重评述各类编码语言的风格，给出了对不同的编码语言进行比较和选择的原则和要点；第六章详细介绍了软件测试方法和纠错技术；第七章主要介绍了软件错误的估计方法和可靠性模型；第八章主要介绍维护的重要意义以及保证维护质量的要点。本书经全国计算机与自动控制教材编审委员会审定并推荐出版，可以作为计算机科学和工程类专业"软件工程概论"课教材，也可作为系统分析员和程序设计人员的重要参考书籍。

本书由孙振飞副教授主编，第一至第四章由孙振飞执笔，第五至第八章由应振澍执笔，复旦大学计算机科学系系主任施伯乐教授担任全书的主审工作。

本书编写过程中还得到了上海工业大学计算机工程系系主任张吉锋副教授的热情支持和帮助，在此表示衷心的感谢。

由于编者水平有限，书中存在的谬误和不妥之处，恳望读者勿吝批评指正。

编 者

1986年于上海

目　录

第一章 概 述

一般来说，计算机软件是计算机系统不可缺少的一部份，它是计算机系统能否充分发挥效能的主要成份。但其效能发挥又必须通过计算机系统的硬件部份表现出来。因此，研制计算机软件必须从计算机系统的总体出发，制定出软件目标进行设计，才能使最终的软件成品耦合于计算机系统之中，以发挥其最大功效。本章将概要介绍软件开发的前期工作。

由于软件不是物理产品而是信息产品，这就给它的开发、工程化带来了一系列的特有方法和手段，本章将就这方面的有关概念介绍如后。

§1.1 计算机系统的开发

计算机系统的开发是指从项目提出开始，经过论证决策、设计、实施直到交付使用的全过程。系统开发过程中确定的目标、作用范围是软件开发的依据，软件开发的结果是系统组成的一部分。所以软件开发是计算机系统开发的一部分。

计算机系统开发的目的是从系统项目提出的总目标出发，考虑到社会、经济、政治和技术等方面的因素，以及资源和时间上的限制，通过优化手段和方法，以求得一个在技术上先进、经济上合理、时间上最省、运行上可靠的计算机系统。

一般来说，计算机系统开发的任务是：揭示和分析系统的目标；确定系统的功能；把系统应具备的功能合理地分配给系统的元素（包括硬件和软件）和环境；进行设计和制造；组

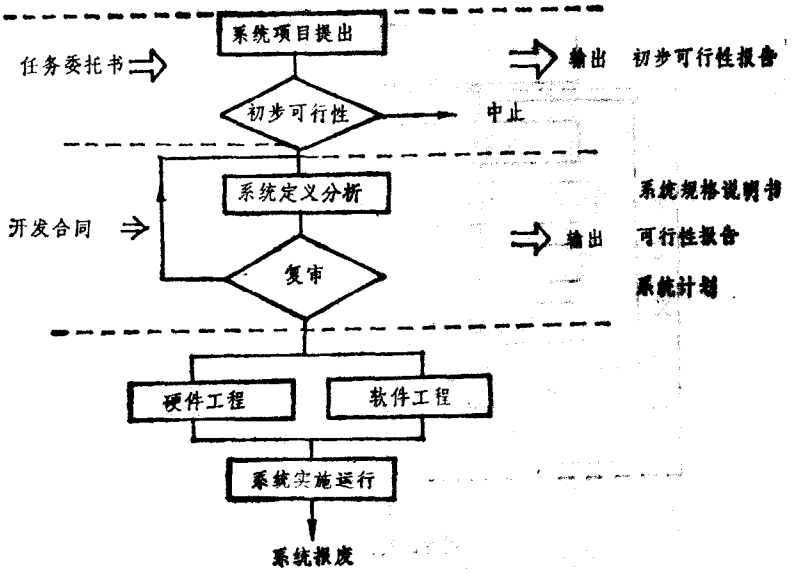


图1.1 计算机系统开发工作流程图

织实施和运行。

一般计算机系统的开发工作流程如图1.1所示。现将图中各框的内容分别介绍如下。

一、制定系统目标

建设一个计算机系统的初始目标往往是由领导部门或用户基于下述理由提出的：

- 基于生产和市场需要。
- 基于改善劳动条件、提高产品质量、提高经济效益等方面的需要。
- 适应技术进步、提高社会效益的需要。

这时提出的目标往往是原则性的、含糊不清的，必须进一步明确。首先应当明确建立计算机系统的目的是为了什么？是适应于哪一类用户对象？期望的性能价格比是多少？允许开发周期多长？总的开发费用多少等等。这些问题应尽量做到定量描述，即使无法定量说明，也需用严密的概念术语精确描述，使开发者和委托开发者对最终的系统总目标有一个共同的理解。

二、初步可行性论证

初步可行性论证主要是从以下几个方面来确定该计算机系统是否值得开发。即：

- 开发该计算机系统的先决条件是否具备。
- 成功的可能性有多大。
- 从技术进步、社会效益、经济效益来看开发该计算机系统是否值得。

初步可行性论证必须给出肯定或否定的回答，它将决定该计算机系统是否开发的决策依据。

三、系统定义和分析

系统定义和分析是计算机系统开发过程中关键性的一步，是关系到系统能否开发成功，能否获得最优的一步。在这一阶段中，我们工作的出发点不是拘泥于某一局部或某一阶段，而是从整体上寻求最优。其主要任务有：

- 依据系统总目标，定义出系统的详细目标、模型、功能、性能和界面。
- 确定系统与环境的界面。

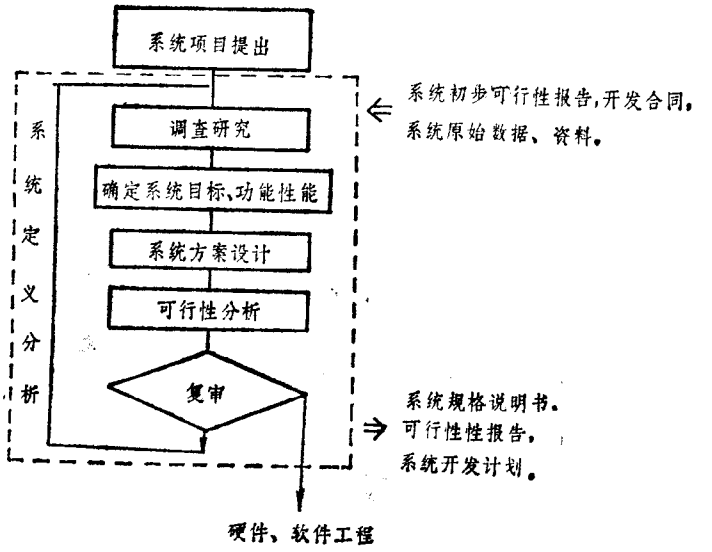


图1.2 系统分析定义工作流程图

·确定硬、软件功能的合理分担。

·进行多种可能的方案设计。

·对多种方案进行可行性论证。

·制定开发进度计划和投资计划。

一般系统定义分析阶段的工作流程图如图1.2所示。下面介绍其几个主要阶段的工作。

1.确定系统的功能和性能。

确定系统功能和性能是对系统总目标的一个求解过程。一般来说它有两种基本方法：一是基于归纳的归纳法，二是基于演绎的分解方法。

(1) 由底向上归纳法一般是用于计算机应用系统的功能确定。其基本思想是从应用对象的现状出发，剖析实际的业务活动，明确问题的处理过程，分析处理过程中的每一操作，以确定计算机系统所应完成的每一项功能，最终确定系统完成的目标。一般可用作业流程图来实现上述过程[7]。

(2) 自顶向下演绎法适用于目前还没有类似的未知系统，它是从系统既定的目标出发，进行逐步分解，直到功能不可分为止。

不管使用上述哪种方法，必须注意到最终功能块之间的相对独立性与相互关联性。

系统性能确定一般是指如下一些内容：

·信息的内容及长度。它随着计算机系统应用领域不同而有着不同的要求。如对科学计算来说希望有较大的字长，以保证有足够的计算精度。对于控制领域来说，一般只需十六位的字长即可，若字长太长，则花费的代价较高。

·信息输入频率。一般分为两种情况，一种是恒定的输入频率，如自动的数据采集系统。这类输入频率的大小确定，主要取决于系统应用对象的要求。另一类是随机输入频率，作为系统性能要求来说，一般取最大输入频率。

·信息输出频率。作为系统性能的要求，它与输入频率要求是用同一方式来确定的。

·系统的响应时间要求是指“用户”向系统提出要求开始（键入开始）到系统作出回答为止。对应实时控制的计算机，主要从系统控制精度和系统安全性来考虑，一般取毫秒级。对于分时系统和事务处理系统，它的响应时间要求主要取决于“服务质量”的需要。

·系统存贮能力的要求。它主要是指内存容量和外存容量。内存容量主要取决于程序量。外存容量主要取决于数据量。

以上讲述的只是性能要求的某些主要方面，其它如可靠性、可维护性、兼容性等一些性能要求就不一一列举了。

2.系统方案设计

系统方案设计的主要任务是根据确定的功能要求、性能要求合理地选择系统模式，确定硬、软件功能的合理分担，并提供多种可替换方案进行选择。

系统模式一般可归结为两大类。一类是集中式单机系统；另一类是多机分布式处理系统。但不管是那一类系统，问题的焦点都在于模块化、层次化、总线结构和通讯接口等方面。至于如何选择则取决于应用对象的特征与要求。

硬、软件功能的合理分担不仅仅是关系到硬、软件开发工作量谁大谁小的问题，而且是关系到计算机能否获得最优性能价格比的一个重要因素。从目前计算机发展水平来看，对系统功能要求既可由百分之一百的硬件来实现，也可由百分之十的硬件和百分之九十的软件来实现。这当然是极端情况。但面对这样一个广阔的选择范围，其选择可以按照如下三条准则

来进行：

(1) 技术性准则　对于任何一个计算机系统来说，都可以划分为可选择部分和不可选择部分。不可选择部分指的是一些基础硬件（如 CPU、存贮设备、通道、外围设备等）。可选择部分指的是各种程序，中断机构等。原则上可选择部分既可以由硬件来实现，也可以由软件来实现，其实现功能的要求在本质上是没有区别的，但就性能来说则各有千秋。其特点可对比如下：

硬　件	软　件
处理速度快	处理速度慢
应付变化的灵活性小	应付变化能力大
扩展功能时要增加部件	扩展功能无需增加部件

由上述分析可知，选择的绝对因素是系统要求的速度性能，选择的相对因素则是其它方面——特别是经济方面的因素。于是我们可确定如下选择准则：选定基本系统（不可选择部份）以后，在保证系统所要求的速度性能的前提下，功能由硬件还是由软件来承担，由所需耗费的投资来作综合权衡。

(2) 经济性准则　硬件的生产成本是与硬件生产的数量成正比的，软件的生产成本几乎与软件的需要数量无关。因此可确定如下准则：在满足系统性能要求的前提下，对于大批量生产的计算机系统，功能应尽可能由软件来承担，对于某些专用系统和小批量生产的系统，则既可由硬件来承担，也可由软件来承担——具体情况要由可靠性和投资耗费来作综合权衡。

(3) 可靠性准则　由于硬件可靠性技术已经比较成熟，而软件可靠性保证还不充分。所以，对可靠性要求高的系统，系统功能应尽可能由硬件承担。

可替换方案一般来说不宜过多，太多将使方案设计工作量增大。但也不宜太少，太少将有可能使最优方案遗漏掉。一般可拟定3～5个基本方案。

3.可行性研究　现在我们讨论的可行性不同于初步可行性论证，初步可行性论证是确定系统有无开发的必要，现在讨论的可行性是多种方案的选择问题。最终可行性报告应使所选择的方案是在资源、时间约束条件下获得投资最小、效益最大、技术上最先进、权益上合法的最优方案。最终可行性报告应当是：

- 领导者决策的依据。
- 向上级部门申请批准和认可的依据。
- 向银行申请贷款的依据。
- 签定有关契约的依据
- 是系统进一步发展的基本依据。

最终可行性报告应当从以下几方面来进行论证：

(1) 经济上的可行性　除经济上是最合理、最省以外，还应评价经济估算上是否合理、市场预测是否可信等等。

(2) 技术上的可行性　首先应评价是否能实现系统的目标，系统所采用的技术是否先进，实现的可能性如何，实现系统的人员素质是否具备等。

(3) 法律上的可行性　应当从合同的责任、专利权、版权等一系列权益方面予以考虑。

(4) 消耗——效益分析　消耗——效益分析是指在整个生存周期中，所花费的代价与所得到的效益之间的度量。对于不同的计算机系统，分析目标是不一致的，分析的方法也有所

不同。如对商品化的计算机系统，其分析目标是能否获得最大销售量，能否获得最大利润；对于应用系统来说，其分析目标应当则是能否用最小的代价，获取最大的经济效益、社会效益和技术进步。

4.系统规格说明

系统定义分析的最终任务是要写出系统的规格说明，它是一个应交付的文档资料，是下一步开发工作的依据。下面给出系统规格说明的参考提纲：

(1) 引言　引言部分首先要描述系统目标和系统将要运行的环境，其次应描述开发的要点：可行性、合理性及所要求的资源。最后是价格和进度的概述。

(2) 功能描述　本节应准确地描述系统的每一功能，其内容应包含输入、要执行的任务、输出结果和附加的接口数据等。

(3) 分配　把(2) 所描述的每一功能分配给合适的系统元素。同时，对系统硬件元素、软件元素应具有的特性、信息的特征、以及现有的数据库或文件系统的特征进行描述。

(4) 约束　约束的描述既包括技术和经济方面的约束，也包括管理上的约束。诸如外部环境、接口方面的约束、政府政策和法令方面的约束，以及资源、投资、进度等方面的限制等。

(5) 成本预算　包括整个系统生存期中物力、人力等资源的消耗。这是一种概算性质的预算。

(6) 进度　这里的进度估算一般是根据用户需要（市场需要），先确定一个项目的最终结束日期，采用倒排的办法，编制出一张粗略的进度表，以后再由硬件和软件开发进度加以协调和修改。

5.系统复审

复审的目的在于：对系统规格说明中描述的各个项目，进行一次正确性检查，以保证系统得到正确的刻划。复审的方式是由开发者和需求者，对规格说明中的各项逐项进行讨论，以取得供需双方一致的理解。其中主要的是指产品性能是否满足系统的目标要求以及实现目标是否困难等。

§1.2　软件及其组成

计算机“软件”这个概念，在不同的时期，不同的人有不同的理解。初学者往往从初始的狭隘观点去理解与认识软件，形成“软件即程序”的概念。这样简单的理解不仅不能概括整个软件的含意，而且影响着软件开发技术的进一步发展。因为“软件即程序”的概念会很自然地把软件开发技术与程序设计方法等同起来，而忽视软件开发技术中的其它方面，如系统需求分析方法、文档资料的编写、软件质量保证、软件维护和管理等。所以我们必须对“软件”有一个正确的理解与认识，以建立起完整确切的“软件”概念。

我们认为“软件是计算机系统中与硬件相互依存的另一部分，它是包含程序、过程及其相关文档资料的完整集合。”从另一个角度来看，软件由两大部分组成：其一是可执行部分，即以编码信息存放在存贮介质上的程序与过程；其二则是与程序和过程有关的文档资料。对于一个计算机系统，软件部分的完整配置可由以下四个主要部份构成：

• 应用程序——是直接面向用户的为解决各种特定问题而编写的程序，如实时控制程序、工程或科学计算程序以及信息管理程序等。

·系统程序——为应用程序服务所编制的程序汇总。它面向计算机硬件，是应用程序的支撑部分。如操作系统、编译程序及编辑程序等。

以上为软件的可执行部分。至于不可执行部分可以分为：

·面向用户的文档——指明如何使用、维护、修改程序。属于这类文档的有用户手册，操作手册以及程序维护手册等。

·面向开发者的文档——保证软件按质、按期有效地进行开发。属于这类文档的有可行性研究报告、项目开发计划、功能需求说明书、数据要求说明书、系统/子系统设计说明书、程序设计说明书、数据库设计说明书以及测试计划、测试分析报告等。

上述文档资料虽属不可执行部分，但它是不可缺少的重要部分。关系到软件能否有效地运行、最佳地维护等一系列问题。

§1.3 软件工程的由来和发展

1945年世界上第一台程序存贮式电子数字计算机的诞生，标志着计算机时代的开始。当时在人们的概念中，认为计算机硬件就是计算机。而对于用计算机机器码手编的程序，仅被视为计算机硬件的使用说明。根本不具有“计算机软件”的概念。1947年冯·诺依曼首先提出了用流程图描述计算机的运行过程，人们才开始意识到程序是计算机中不同于硬件的另一部分，程序设计是完全不同于硬件研制的另一种开发研究工作。于是，软件的开发和研究才开始独立地进行。三十多年来，随着计算机硬件的不断更新换代，应用领域的不断拓展和深入，计算机软件的规模、功效和性能也在不断增大和进步，与计算机硬件一样，已经成为计算机系统中必不可少的独立成份。因而，也就形成了一个独立的产业，即软件产业。与此同时，软件的概念在不断地演变和发展，软件的开发工具和开发方法也在不断地进步和创新。为了深刻理解软件的作用与意义，了解研制软件的基本途径，下面就软件的发展过程作一扼要阐述。

软件的开发从机器码手编程序到今天的软件工程方法，大致经历了以下几个时期。

程序时期(47年～60年代初)。人们直接感触到的是计算机硬件。因而一开始在认识上就形成了计算机硬件就是计算机这一概念。由于最初人们所耽心的是计算机能否持续、可靠地运行等问题，所以，主要致力于改善硬件结构和可靠性的研究，从事这些研究工作的主要是硬件工程师和物理学家。

对于计算机软件，当时并没有充分认识到是发挥计算机效能的主要组成部分，只将其作为机器运行时必须进行的准备工作。每一个程序都是为求解某一个问题而专门设计的，很少考虑到它的通用性。程序设计全凭个人经验和技艺单独进行，完全是一种个体生产方式。对于程序的运行、维护等工作全部由程序设计者自身承担，不需要向他人做任何交待和解释。这样，计算机软件除可执行部分的程序外，不可执行的文档资料部分就不是必须的，故当时人们只有程序的概念，没有软件的概念。

在这个时期的后半段，计算机硬件技术有了较大的发展，稳定性和可靠性都有了极大的提高。随着通道技术、中断技术的出现，外存贮设备、人机交互设备的改进，为扩大计算机应用奠定了基础。50年代末，计算机的应用已经从单一的科学计算扩展到数据处理和实时控制等方面，相应地出现了各种各样的应用软件。与此同时，人们为了摆脱机器码编程的困境，提高机器运行效率，相继研制出一批高级程序设计语言的编译系统软件（如 FORTRAN

II、ALGOL 60等）以及操作系统等支撑软件。无论是各种各样的应用软件，抑或编译、操作系统等系统软件，其规模都比较大，各成份之间的关系也比较复杂，通用性比较强，早期“程序”概念已不再适应，因而提出了“软件”的概念。

“软件＝程序＋说明”时期（50年代末～70年代初）。这个时期的特征主要表现在以下三个方面：（1）由于程序规模增大，那种完全由个人独立编程的“个体生产方式”已经不能适应生产要求，而是需要多人分工、共同协作来编制一个程序，即所谓“作坊式生产方式”。（2）程序的设计与程序的运行维护再也不是由一人来承担。（3）由于人们已认识到程序可以使计算机发挥巨大的灵活性，程序再也不是计算机硬件的附属成分，而是计算机系统中与硬件相互依存的不可缺少的部分。在计算机系统的开发过程中，起主导作用的不再只是硬件工程师，同时也包括了软件工程师。基于这三个特征，在开发过程中，人们如何互相进行通讯的问题就显得比较突出。例如，在开发一个大型软件时，对于所有参与设计的人员来说，必须要有能共同理解的约定说明。在运行时必须给运行维护人员一个软件使用维护说明。硬件工程师和软件工程师之间要有一个相互制约的要求说明等。

在这个时期，软件技术取得了很大的进展，比较有代表性的是多道程序设计技术、多用户人机交互系统、各种应用的实时系统以及文件管理系统等。在程序设计语言方面，不仅出现了多种通用的和专用的程序设计语言，如FORTRAN IV、PL/1、COBOL、LISP、SNOBOL等。在形式语言理论和编译理论方面也取得了重大的突破，这些都给计算机的广泛应用奠定了基础。

同时，由于各种通用的系统软件随机出售，各种应用软件包的大量出现，也就标志着软件产品化、商品化时期的到来。在美国和西欧一些大的制造计算机的厂家，均相继建立起庞大的软件公司，一种新兴产业——软件产业便应运而生了。

随着计算机应用领域的不断扩大，计算机的数量猛增，软件规模愈来愈大，复杂程度亦愈来愈高。但软件生产还停留在作坊式生产方式上，这样就势必造成软件质量不高，生产率过低，以致一个软件项目在开发过程中往往不得不中途夭折。这种局面的出现，人们称之为“软件危机”的开始。

为了摆脱这一困境，北大西洋公约组织成员国的软件工作者于1968、1969连续两年召开软件研讨会，即有名的NATO会议。集中研究对策，讨论如何摆脱软件危机。于是，“软件工程”这个术语开始出现，也就意味着进入了软件第三个时期即“软件工程”时期。

软件工程时期（1970年至今）。“软件工程”即如何克服“软件危机”中所遇到的困难问题。具体地说，即如何用科学管理方法去组织软件开发，以及解决软件开发计划、软件开发成本失控等一系列问题。另一方面它包括探索新的设计方法、手段和技术工具，以便提高软件的可靠性、生产率以及可维护性等一系列软件危机中所暴露出的问题。

近十五年来，软件工程已经取得了重大进展，在软件管理方面，如按软件生存周期编排进度计划，逐一确认与验证，主程序员负责制等。在设计方法方面，有按数据流结构化设计法（SD法）、按数据结构设计法（Jackson 方法）等。在支撑环境和技术工具方面，有美国TRW公司研制的SREP / SREM 系统（Software Requirement Engineering Program / Software Requirement Engineering Methodlogy）。美国密执安大学研制成的ISDOS 系统（Information System Design Optimization System）等。这些成就对开发软件、克服软件危机都起到了积极有效的作用，下面的实例便证实了这一点。IBM公司为纽约时报开发的一个情报系统，应用了软件工程中的方法，主要按数据流结构设计法，不仅按时交“货”，而且

质量也有很大提高，八万行程序只有二十五处错误[2]。又如日本的东芝电气公司，用Jackson方法设计了一个非常庞大的数据库管理系统，用于银行信贷管理，编程功效提高了将近一倍，数据库管理程序的错误减少了30%，其它程序错误减少了65%。

“软件工程”中的一系列技术和方法的引入，解决了软件开发中的许多难题，成绩是显著的。但“软件危机”并没有完全消失，这主要反映在以下几个方面：

(1) 软件生产不能满足日益增长的需要。自超大规模集成电路取得突破性进展以来，硬件价格迅速下跌，计算机设置台数迅速增加，特别是廉价的微型计算机。美国安隆联合公司预测1990年美国微型机销售台数将为1982年的9倍，达2800万台。届时，美国微型机设置台数将达到超过一亿台。如此巨大的装机量，势必导致软件需求量的增长(见图1.3)但另一方面，

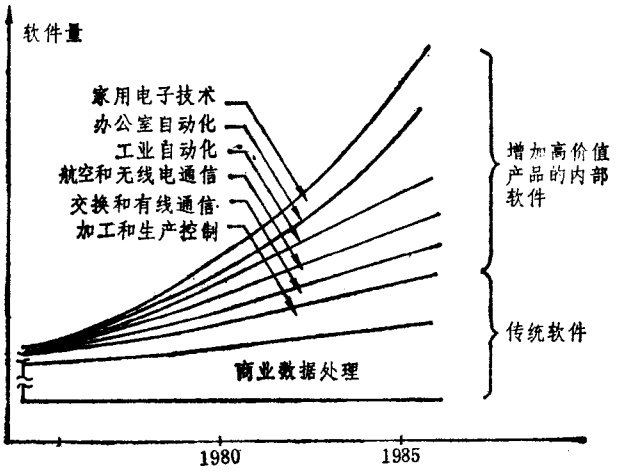


图1.3 对软件的需求趋势

软件是一种智力型产品，生产人员的增长和生产率增长均很慢。据美国预测，软件供需差的预测如图1.4所示，到1990年会更加增大，那时将缺少软件人员100万人左右。

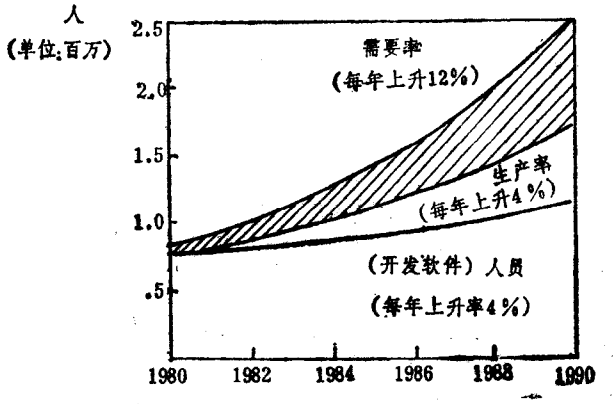


图1.4 软件供需差的预测(美)

(2) 软件成本日益增加，维护费超过开发费。1976年美国著名软件专家B.W.Boehm宣称，依据美国空军资料估算，计算机系统中硬件、软件开发和维护费用所占比例如图1.5所示。这种估计虽然不很精确，但足以说明软件费用和软件维护费的增长相当迅猛。

(3) 软件的复杂性增加，可靠性等质量要求愈来愈高。计算机应用领域的不断扩大和深入，使得软件的复杂性不断增加。例如由多台微处理机组成的分布式系统，软件需要适应并发执行、相互通讯等问题，无疑都增加了软件的复杂性。而可靠性要求也愈来愈高，如对银行信贷系统，任何微小差错都可能导致整个系统的失效，不允许有任何微小的数据差错。又如在工业控制的实时系统中，任何不允许的延时响应都可能导致整个生产过程的瘫痪或设备损坏、人身事故的发生等。

综上所述，为了扭转软件生产中的这种困难局面，必须进一步研究软件工程的方法和理论。

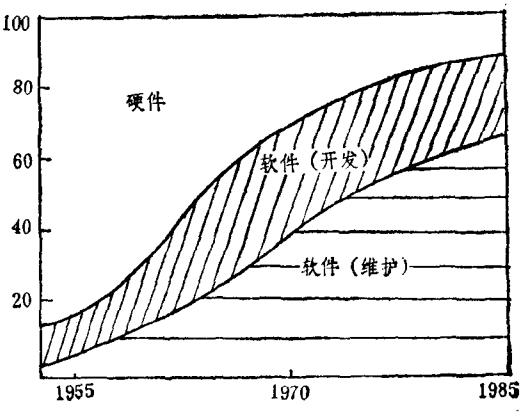


图1.5 硬件、软件开发及维护费用比率
（根据TRW公司统计资料）

§1.4 软件的生命周期

软件的生命周期是指某一软件项目被提出并着手实现开始，直到该软件报废或停止使用为止。研究软件生命周期的目的是为了更有效、更科学地组织和管理软件生产，以便最终能获得最经济、最可靠的软件产品。一般来说，软件的生命周期主要经历以下几个阶段：软件规划、软件需求分析和定义、软件设计、编码、验证和测试、软件维护运行等。

软件规划阶段。主要任务是确定将开发的软件是“做什么”，在经济上、技术上是否可行。具体地说，就是要确定软件的工作域，预测开发该软件所需的人力物力资源，对成本进行估计等。

软件需求分析定义阶段。主要任务仍然是解决“做什么”的问题。它把规划阶段初步拟定的软件工作域进行求精和细化，分析各种可能的解法，构造出系统的数据流或数据结构，配置各个软件元素。

软件设计阶段。主要任务是解决“如何做”的问题，根据对软件的需求及数据流或数据结构，选择软件设计方法，确定软件结构，把结构元素变换成过程描述等。

编码阶段。主要任务是对软件结构元素的过程描述进行编码，选取合适的程序设计语言来编写程序，并进行模块调试。

验证和测试阶段。主要任务是检查该软件是否符合要求，一般可分为两大方面：一是检查文档资料是否齐全，文档资料的内容是否符合规定要求；一是测试工作，即测试程序是否有逻辑性错误，测试程序执行的功能是否符合要求。

运行维护阶段。这个阶段占整个软件费用的比例最大，是相当重要的一个阶段。由于软件不存在零件的磨损、超负荷运行，所以软件维护的实质是对软件继续进行查错、纠错和修改的过程。一般把维护分为三种性质的维护：

(1) 订正性维护。对性能、功能、处理、实现等出现的错误，进行纠正。

(2) 适应性维护。对处理对象或数据环境变化时，作某些适应性修改。

(3) 完整性维护。为了提高性能或在可维护性方面所做的某些修改。

上面所讨论的软件生命周期的定义，仅就软件系统内部而言，对一些通用性较强、小型

软件产品来说是完全适用的，因为它往往作为独立系统而存在。对于大型的、专用性较强的一类系统，这样来定义它的软件生命周期，就显得不足了。首先，它的起始阶段——软件规划取决于整个系统的需求分析。其次，它是依附于整个的、特定的计算机系统，当整个计算机系统结束其“生命”，软件系统自然也就失效了。所以把系统的需求分析作为软件生命周期的开始部分是需要的。

系统需求分析的主要任务是把整个项目看成一个系统，根据对需求的初步理解，确定系统功能，把功能合理地分配到硬件、软件以及其它系统元素。因此，可将上面讨论的软件生命周期概括为如图1.6所示的“工艺流程”图，这个图只是简单地反映了软件生命周期各个阶段的顺序，并没有反映出软件生产的管理和控制。1976年B.W.Boehm 从保证软件可靠性

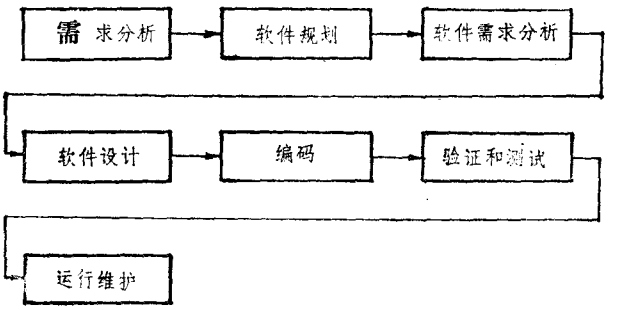


图1.6 软件“生产工艺”流程图

出发，提出了一种连续验证生命周期的模式，如图1.7所示。这种生命周期管理模式的特点是：

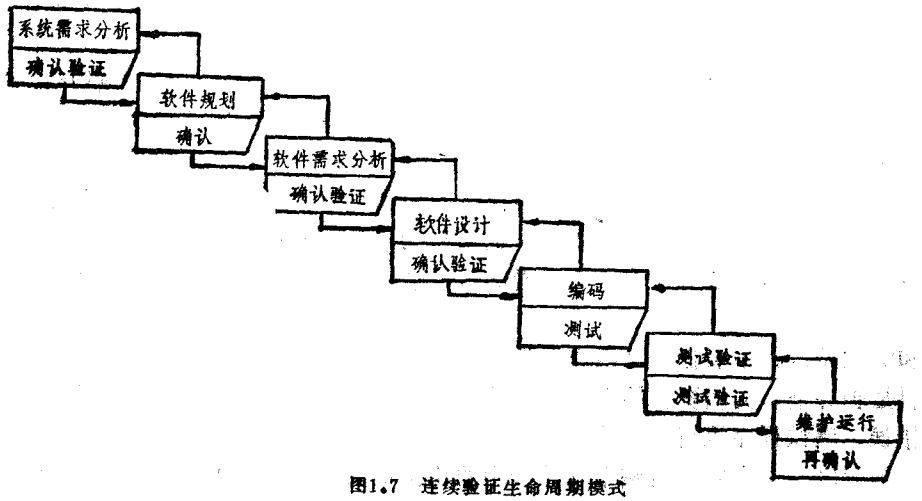


图1.7 连续验证生命周期模式

(1) 在从一个阶段进入下一阶段之前，要对本阶段所做的工作有明确的要求和确认验证的标准，并能为下一阶段工作人员所理解，这样做可以及早纠正错误。

(2) 在软件规划阶段的开始就要制定以后各阶段的测试与验证标准，以避免“削足适履”的情况发生。

(3) 对需求分析要制定出测试标准，从而避免需求分析中容易产生的歧义性。

所以，图1.7是一种有效的生命周期管理模式。

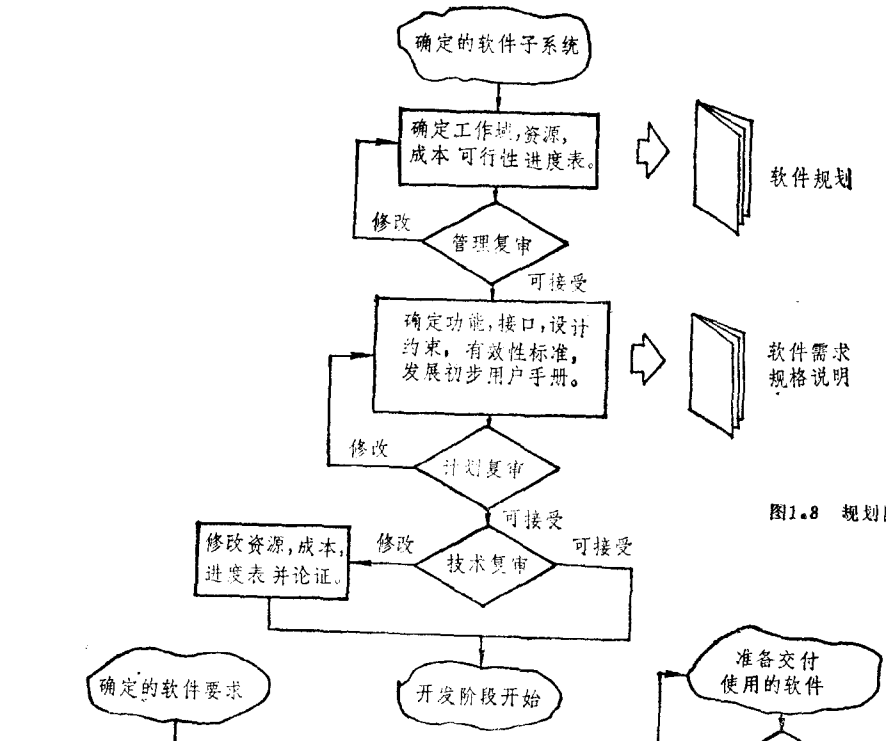


图1.8 规划阶段流程图

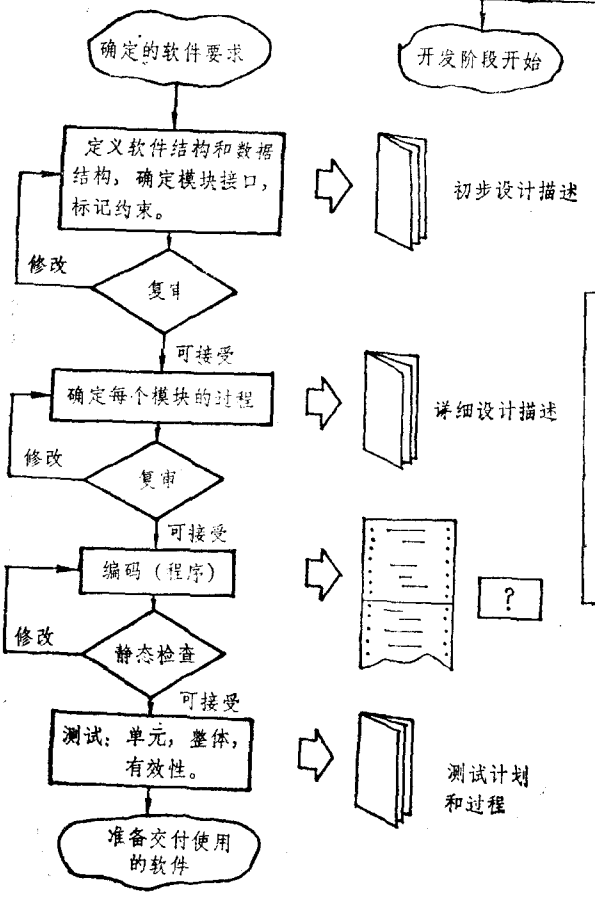


图1.9 开发阶段流程图

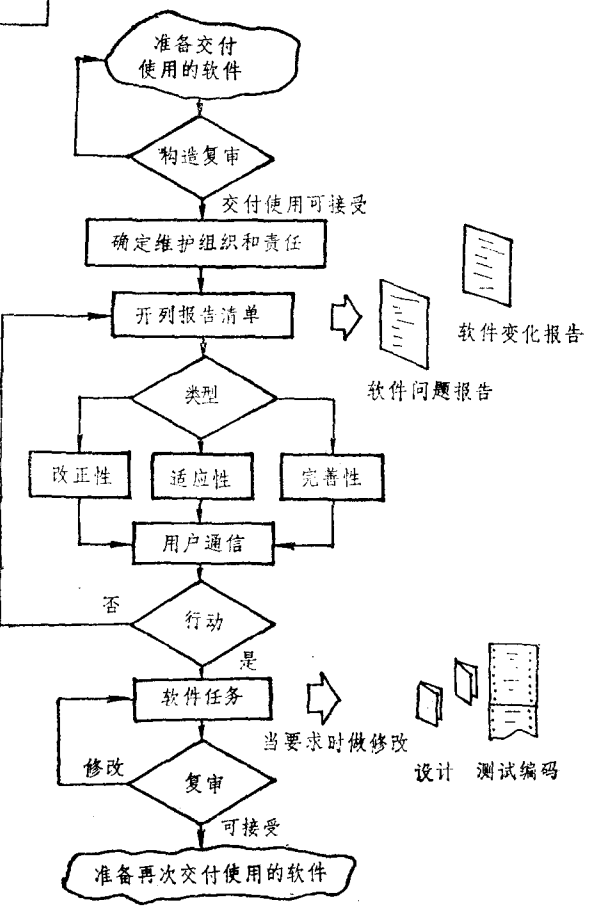


图1.10 维护阶段流程图