高等院校计算机经典教材

软件工程

Engineering

李代平　编著

冶金工业出版社

高等院校计算机经典教材

软 件 工 程

李代平　编著

北　京

冶 金 工 业 出 版 社

2002

内 容 简 介

本书介绍了软件工程学及应用，软件工程的特点，软件危机，软件工程的基本理论，可行性研究，软件需求分析，总体设计，软件细节设计，面向对象方法学，面向对象分析与设计，形式化方法，用户界面设计，软件质量，软件实现，软件测试，软件维护，软件项目管理与计划等知识，还介绍了软件工程的应用实例。

本书可以作为大专院校相关专业高年级学生的教材和参考用书，也可供计算机专业高级人员参考。

图书在版编目（CIP）数据

软件工程 / 李代平编著. —北京：冶金工业出版社，2002.8

ISBN 7-5024-3086-5

I. 软… II. 李… III. 软件工程 IV. TP311.5

中国版本图书馆 CIP 数据核字（2002）第 055822 号

出版人 曹胜利（北京沙滩嵩祝院北巷 39 号，邮编 100009）

责任编辑 程志宏

中山市新华印刷厂有限公司印刷；冶金工业出版社发行；各地新华书店经销

2002 年 8 月第 1 版，2002 年 8 月第 1 次印刷

787mm × 1092mm 1/16； 25.75 印张； 596 千字； 402 页； 1-2500 册

38.00 元

冶金工业出版社发行部 电话：（010）64044283 传真：（010）64027893

冶金书店 地址：北京东四西大街 46 号（100711） 电话：（010）65289081

（本社图书如有印装质量问题，本社发行部负责退换）

前　言

1. 关于本书

软件工程是指导计算机软件开发的工程科学。人们希望通过用工程技术和管理方法使软件开发工程化，由此产生了软件工程学。软件工程学是采用工程的概念、原理、技术与方法，把当前先进的技术与已经实践证明了的正确管理方法相结合来开发软件。从20世纪60年代提出软件工程的概念以来，软件工程技术逐渐成熟，现在已成为计算机科学技术中的一门重要学科。但是，还有些公司和个人仍然在随意开发软件，将编写高质量的程序与开发系统混为一谈，也有些软件专业的学生或软件开发人员还没有掌握软件开发出现的新技术，鉴于此我们编写了本书。

本书是作者根据近 10 年来对软件工程学、面向对象方法等教学与研究的经验，以及领导或参与的20项软件项目开发的实际应用经验，并结合软件开发新技术编写而成。根据过去的教学经验，读者学习一门新技术，教材是非常重要的。因此，我们在编写本书之前，在各方面进行了充分的准备。

2. 本书结构安排

本书由16章构成，内容包括：

第1章：绪论。介绍的主要内容有：软件的特点、软件的发展、软件危机，软件工程，软件工程与方法学，软件工程的基本原理等。

第2章：软件工程的基本理论。介绍的主要内容有：软件工程过程，软件生存周期，软件生存周期模型，软件开发方法，软件开发工具。

第3章：可行性研究。介绍的主要内容有：可行性研究的任务与步骤，系统分析，分析原理，结构化分析，系统流程图，数据流图，数据字典，成本-效益分析，可行性研究的文档，项目开发计划。

第4章：软件需求分析与概念模型。介绍的主要内容有：需求分析，IDEF方法，概念模型与规范化。

第5章：总体设计。介绍的主要内容有：软件设计的重要性，设计过程，软件总体设计，设计基本原理，体系结构设计，结构化设计，IDEF0图的设计方法，软件结构优化。

第6章：软件细节设计。介绍的主要内容有：细节设计任务与方法，设计表示法，结构化程序设计，结构化定理，图形工具，面向数据结构的设计。

第7章：面向对象方法学。介绍的主要内容有：传统方法学的缺点，面向对象的基本概念，对象模型，动态模型，功能模型。

第8章：面向对象分析。介绍的主要内容有：面向对象分析的基本过程，对象的发现和标识，发现对象方法，定义属性，定义服务，定义结构，实例连接，消息连接，建立功能模型。

第9章：面向对象设计。介绍的主要内容有：设计的准则，启发式规则，系统分解，设计问题域子系统，设计任务子系统，设计数据管理子系统，面向对象程序设计，软件重用，统一建模语言UML。

第 10 章：形式化方法。介绍的主要内容有：形式化方法的基础知识，有限状态机（FSM），Petri 网的基本原理，净室方法学，客户/服务器模式。

第 11 章：用户界面设计。介绍的主要内容有：界面软件开发综述，设计人-机交互子系统，图形用户界面设计，多媒体用户界面设计，用户界面模型，用户界面的描述方法与技术等。

第 12 章：软件质量。介绍的主要内容有：软件质量的概述，软件质量的度量和评价，软件质量保证，技术评审与审查，软件的可靠性。

第 13 章：软件实现。介绍的主要内容有：程序设计语言的特性及选择，程序设计风格，程序设计效率，冗余编程，软件容错技术。

第 14 章：软件测试。介绍的主要内容有：软件测试概述，测试方法，测试用例的设计，测试过程，调试。

第 15 章：软件维护。介绍的主要内容有：软件维护概述，软件可维护性，软件维护的特点，软件维护的实施，维护“老化代码”，逆向工程和再工程。

第 16 章：软件项目管理与计划。介绍的主要内容有：软件项目管理概述，项目管理过程，软件开发成本估算，风险分析，进度安排，软件项目的组织。

此外，本书的最后给出了一个附录，列出了软件产品的主要文件，以供读者参考。

3. 本书特点

本书侧重于理论联系实际，从实用性、易懂性出发、重点突出，内容丰富而实用。在详细介绍理论的同时，书中给出了部分示例，以利于读者掌握其实际应用的方法。此外，为了便于读者巩固所学的知识，在各章的后面都附有相应的小结与练习题。

4. 适用对象

本书可作为大专院校相关专业高年级学生的教材和参考书，也可供计算机专业的高级人员参考。

本书由李代平编写，另外张信一参加了第 9、11、12 章的编写，彭重嘉参加了第 13~16 章的编写。

由于作者水平有限，书中的不足之处在所难免，恳请读者批评指正。

编　者

2002 年 7 月

目　录

第1章 绪 论

软件工程是指导计算机软件开发和维护的一种工程科学，它涉及的知识相当广泛。在学习软件工程之前，必须对软件工程领域的一些基本概念有所了解，对软件工程有一个初步的认识，这样才能顺利地进入后面章节的学习。

1.1 软件概述

计算机系统由硬件和软件两大部分组成。电子计算机自 1946 年诞生后到 20 世纪 60 年代中期是计算机发展的早期。在早期，计算机系统还是以硬件为主，软件费用是总费用的 20%左右。到了中期（20 世纪 60 年代中期到 20 世纪 80 年代初期），软件费用迅速上升到总费用的 60%，软件不再只是技巧性和高度专业化的神秘机器代码。而到 1985 年以后，软件费用已上升到今天的 80%以上。软件相对硬件的费用比例在不断提高。我们可以从图 1-1 中看到硬件和软件费用比例的变化。

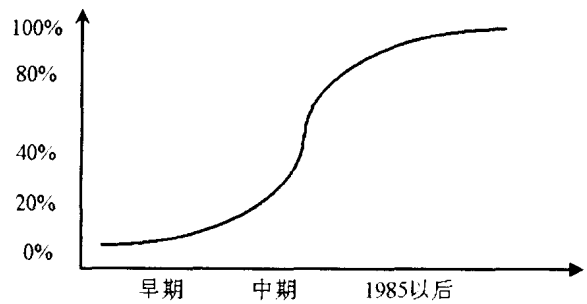

图 1-1 硬件和软件费用比例变化示意图

“软件”的定义是计算机程序及其说明程序的各种文档。在该定义中，“程序”是计算任务的处理对象和处理规则的描述；“文档”是有关计算机程序功能、设计、编制、使用的文字或图形资料。软件与硬件一起构成了计算机系统。

1.1.1 软件的特点

计算机系统中的软件与硬件是相互依存的，缺一不可。而软件与其他产品的特点不同。它是一种特殊的产品，具有下列特殊性质：

1）软件产品的生产主要是脑力劳动，还未完全摆脱手工开发方式，大部分产品是“定做”的。

2）软件是一种逻辑产品，它与物质产品有很大的区别，它是脑力劳动的结晶。软件产品是看不见摸不着的，因而具有无形性。它以程序和文档的形式出现，保存在存储介质上，通过计算机的运行才能体现它的功能和作用。

3）软件产品不会用坏，不存在磨损、消耗问题。

4）软件产品的生产主要是研制。其成本主要体现在软件的开发和研制上，软件开发

研制完成后，通过复制就产生了大量软件产品。

5）软件费用不断增加，软件成本相当昂贵。软件的研制工作需要投入大量的、复杂的、高强度的脑力劳动，它的成本非常高。

1.1.2　软件的发展

自第一台计算机诞生以来，软件的生产就开始了。随着计算机技术的飞快发展和应用领域迅速拓宽，自20世纪60年代中期以后，软件需求迅速增长，软件数量急剧膨胀。这种增长导致了软件的发展，我们可以将软件生产的发展划分为三个时代。

1.　程序设计时代（1946~1956年）

在这一时期，软件的生产主要是个体手工劳动的生产方式。程序设计者使用机器语言、汇编语言作为工具；开发程序的方法上主要是追求编程技巧和程序运行效率。在程序设计中还没有注意其他辅助作用。因此所设计的程序难读、难懂、难修改。这个时期硬件特征是价格贵、存储容量小、运行可靠性差。软件特征是只有程序、程序设计概念。不重视程序设计方法。

2.　程序系统时代（1956~1968年）

由于计算机的应用领域不断扩大，软件的需求也不断增长，软件由于处理的问题域扩大而使程序变得复杂，设计者不得不由个体手工劳动组成小集团合作，形成作坊式生产方式小集团合作生产的程序系统时代。生产工具是高级语言。开发方法仍旧靠个人技巧。由于大的程序需要合作，在程序设计中开始提出结构化方法。这时硬件的速度、容量及工作可靠性有明显提高，价格降低，销售有爆炸性增长。软件方面的程序员数量也猛增，其他行业人员大量进入这个行业。这时一方面大量软件开发的需求已提出，软件的规模越来越大，结构越来越复杂。发现错误必须修改程序。软件维护的资源耗费严重，许多软件最终成为不可维护。另一方面由于开发人员无规范约束，又缺乏软件理论方法。开发技术也没有新的突破，开发人员的素质和落后的开发技术不适应大规模、结构复杂的软件的开发。因此，这种尖锐的矛盾导致软件危机的产生。

3.　软件工程时代（1968年至今）

1968年在联邦德国召开的国际会议上讨论软件危机的问题，在这次会议上正式提出并使用了“软件工程”术语，新的工程科学就此诞生。软件工程时代的生产方式是采用工程的概念原理技术和方法。使用数据库、开发工具、开发环境、网络、分布式、面向对象技术来开发软件。硬件特征是向超高速、大容量、微型化以及网络化方向发展。软件特征是开发技术有很大进步，但是未能获得突破性进展，软件价格不断上升，没有完全摆脱软件危机。

1.1.3　软件危机

所谓软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题。这种“严重问题”不仅仅是“不能正常运行”。实际上几乎所有的软件都不同程度地存在问题。软件危机主要是指如何开发软件，怎样满足对软件日益增长的需求，如何维护数量不断膨胀的现有软件。

1. 软件危机的表现

软件危机表现在以下几方面：

1）对于软件开发的成本和进度的估计很不准确。由于缺乏软件开发的经验和软件开发数据的积累，使得开发工作的计划很难制定。主观盲目制定的计划，执行起来和实际情况有很大差距，使得开发经费一再突破。由于对工作量和开发难度估计不足，进度计划无法按时完成，开发时间一再拖延。

2）开发的软件产品不能完全满足用户要求，用户对已完成的软件系统不满意的现象常常发生。一般情况下软件开发人员在开发初期对用户的要求了解不够明确，未能得到明确表达，就开始着手编程。开发工作开始后，软件人员和用户又未能及时交换意见，使得一些问题不能及时解决，导致开发的软件产品不能完全满足用户要求。有些软件还因为合作与技术问题而导致失败。

3）开发的软件可靠性差。由于在开发过程中，没有确保软件质量的体系和措施，在软件测试时，又没有严格的、充分的、完全的测试，提交给用户的软件质量差，在运行中暴露出大量的问题。这种不可靠的软件，轻者会影响系统正常工作，重者会发生事故，造成生命财产的重大损失。

4）软件通常没有适当的文档。开发过程无完整、规范的文档，发现问题后进行杂乱无章的修改。程序结构不好，运行时发现错误也很难修改，导致可维护性差。

5）软件的可维护性差。由于开发过程没有统一的、公认的规范，软件开发人员按各自的风格工作，各行其是。很多程序中的错误非常难改，实际上不可能使这些程序适应新的硬件环境，也不可能根据用户要求在程序中增加新功能。

6）软件开发生产率提高的速度，远远跟不上计算机应用普及深入的趋势。软件产品“供不应求”的现象使人类不能充分利用计算机硬件资源提供的巨大潜力。

2. 软件危机的产生

软件发展第二阶段的末期，由于计算机硬件技术的进步，计算机运行速度、容量和可靠性有显著的提高，生产成本有显著下降，为计算机的广泛应用创造了条件。一些复杂的、大型的软件开发项目提了出来。但是，软件开发技术一直未能满足发展的要求。软件开发中遇到的问题因找不到解决的办法，使问题积累起来，形成了尖锐的矛盾，导致了软件危机。

3. 软件危机的原因

在软件的开发和维护过程中存在的这么多的问题，一方面与软件本身的特点有关，另一方面也与软件的开发和维护的方法有关。造成上述软件危机的原因概括起来有以下几方面：

1）软件的规模愈发庞大。1968 年美国航空公司订票系统达到 30 万条指令；IBM360OS 第 16 版达到 100 万条指令；1973 年美国阿波罗计划达到 1000 万条指令。这些庞大软件的功能非常复杂，体现在处理功能的多样性和运行环境的多样性。归纳当时软件应用的状况，可以说软件的规模越来越大，结构越来越复杂。软件不同于一般的程序，它的显著特点是庞大。随着计算机应用的日益广泛，需要开发的软件规模日益庞大，软件结构也日益复杂。有人曾估计，软件设计与硬件设计相比，其逻辑量要多达 10~100 倍。对于这种庞大规模

的软件，其调用关系、接口信息复杂，数据结构也复杂，这种复杂程度超过了人所能接受的程度。

2）软件开发的管理困难。软件不同于硬件，它是计算机系统中的逻辑部件。在写出代码并在计算机上试运行前，由于软件规模大，结构复杂，又具有无形性，软件开发过程的进展情况较难度量，质量也难评价。因此导致管理困难，进度控制困难，质量控制困难，可靠性无法保证。另外，软件在运行过程中不会因为使用时间长而被用坏。如果运行中出现问题，一般是在开发阶段引入的而在测试阶段没有发现的问题。因此，软件维护通常意味着对原有设计的改进。

3）软件本身的独有特点确实给开发和维护造成一些客观困难，但是人们在长期的实践中也积累了不少成功的经验。如果坚持使用成功的经验和正确的方法，许多困难是可以克服的。但是相当多的软件开发人员对于软件的开发和维护存在不少糊涂的观念，实践中或多或少地采用错误的方法和技术。这可能是软件危机的主要原因。

4）软件开发和维护中许多错误认识和方法的形成可以归结与计算机发展早期软件开发的个体化特点。其主要表现在对软件需求分析的重要性认识不够，错误地认为软件开发就是写程序并使之运行，不重视软件需求分析与维护等工作。

5）软件开发技术落后。在 20 世纪 60 年代，人们注重一些计算机理论问题的研究，如编译原理、操作系统原理、数据库原理、人工智能原理、形式语言理论等，不注重软件开发技术的研究，用户要求的软件复杂性与软件技术解决复杂性的能力不相适应，它们之间的差距越来越大。

6）生产方式落后。软件仍然采用个体手工方式开发，根据个人习惯爱好工作，无章可循，无规范可依据，靠言传身教方式工作。

7）开发工具落后，生产率提高缓慢。软件开发工具过于原始，没有出现高效率的开发工具，因而软件生产率低下。在 1960~1980 年期间，计算机硬件的生产由于采用计算机辅助设计、自动生产线等先进工具，使硬件生产率提高了 100 万倍，而软件生产率只提高了 2 倍，相差十分悬殊。

1.2　软件工程

软件工程是指导计算机软件开发和维护的工程科学。为了克服软件危机，人们从其他产业的工程化生产得到启示，采用工程的概念、原理、技术和方法来开发和维护软件，把经过时间考验而证明正确的管理技术与方法技术结合起来，这就是软件工程。

1.　软件工程的定义

软件工程是用科学知识和技术原理来定义、开发、维护软件的一门学科。该定义说明了软件工程是计算机科学中的一个分支，其主要思想是在软件生产中用工程化的方法代替传统手工方法。工程化的方法借用了传统的工程设计原理的基本思想，采用了若干科学的、现代化的方法技术来开发软件。这种工程化的思想贯穿到需求分析、设计、实现，直到维护的整个过程。

2.　软件工程的性质

软件工程是一门综合性的交叉学科，它涉及哲学、计算机科学、工程科学、管理科学、

数学和应用领域知识。计算机科学中的研究成果均可用于软件工程，但计算机科学着重于原理和理论，而软件工程着重于如何建造一个软件系统。

软件工程要用工程科学中的观点来进行费用估算、制定进度、制定计划和方案；要用管理科学中的方法和原理进行软件生产的管理；要用数学的方法建立软件开发中各种模型和各种算法，如可靠性模型，说明用户需求的形式化模型等。

1.2.1 软件工程与方法学

程序设计方法学和软件工程方法学是为了解决软件危机问题而逐渐形成的学科。1967年 Floyd 提出的断言方法证明流程图程序的正确性；1968 年 Dijkstra 的“GOTO”有害论；1969 年 Hoare 在 Floyd 断言法基础上提出的程序公理方法；1971 年 Wirth 的“自顶而下逐步求精”等对软件工程和程序设计方法学的形成和初期的发展有着深刻的影响。1969 年 IFIP（国际信息处理协会）成立了“程序设计方法学工作组”—— WG2.3，云集了当时许多著名的计算机科学家，专门研究程序设计方法学，这个国际组织对以后的程序设计方法学的发展起了很大的促进作用。

“软件工程”（Software Engineering）作为一个术语，是在 1968 年北大西洋公约组织的一次计算机学术会议上，正式提出来的。这个会议专门讨论了软件危机问题。这次会议是软件发展史上一个重要的里程碑。

事实上，在软件工程和方法学形成以后，研究工作一开始就分成了两种不同的角度和方法，形成了两种紧密相关、相辅相成又各有侧重的学科。一种是以数学理论为基础的理论性学科：程序设计方法学。程序设计方法学是主要运用数学方法研究程序的性质以及程序设计的理论和方法的学科；另一种是以工程方法为基础的工程学科：软件工程学。软件工程学是主要应用工程的方法和技术研究软件开发与维护的方法、工具和管理的一门计算机科学与工程学交叉的学科。软件工程学和程序设计方法学的研究都是研究软件开发和程序设计的学科，它们的研究对象、研究内容、出发点和目标都是一致的。

软件工程学与程序设计方法学的研究对象是软件和程序。它们的根本目标是以较低的成本开发高质量的软件和程序，具体包括：

1）提高软件的质量与可靠性。

2）提高软件的可维护性。

3）提高软件生产率，降低软件开发成本等。

但是，软件工程学和程序设计方法学研究的途径和侧重点有所差异，主要差异有：

1）研究方法和途径不同。软件工程学应用的是工程方法；而程序设计方法学依据的数学方法。软件工程学注重工程方法与工具研究，程序设计方法学则注重算法与逻辑方法研究。

2）研究对象有所侧重，软件工程的对象所指的软件，一般是指“大型程序”，是一个系统；而程序设计方法学的研究对象则侧重于一些较小的具体程序模块，早期的程序设计方法学研究重点是某个单独的程序的时空效率、正确性证明等问题。

3）软件工程学注重“宏观可用性”；程序设计方法学注重“微观正确性”。例如，软件工程学研究软件的“可靠性”的方法是“软件测试”，程序设计方法学研究的方法则是程

序的“正确性证明”。

随着软件技术的迅速发展，软件工程学和程序设计方法学的研究内容也都在不断发展，研究的内容和方法相互渗透。事实上，人们已经很少、也没有必要区分什么是软件工程学的范畴，什么是程序设计方法学的范畴了。逐渐地，这两条研究途径的界限又模糊化、一体化了。

一方面，程序设计方法学研究已发生了较大的变化，逐渐从“纯粹的程序”正确性证明等较老的课题转向“软件”的结构化、正确性、可靠性及软件设计方法方面的研究。例如，现在程序设计方法学及软件工程学都将面向对象的方法作为其重要的新的研究方向，“程序设计方法学”正逐渐发展成“软件设计方法学”。

另一方面，软件工程从一开始就是以程序设计方法学为基础的一门工程学科，而且还在不断吸收程序设计方法学和计算机科学理论新成果和新技术。从某种意义上可以说，软件工程学实际上就是“应用设计方法学”。

实际上“软件工程学”或“程序设计方法学”术语都已难以准确表达它们的研究内涵和含义了。也许采用“软件工程方法学”或“软件工程与方法学”更能概括当前软件工程学和方法学研究的内涵。

可以这样描述：软件工程方法学是指应用计算机科学理论和工程方法相结合的研究方法，研究软件生存周期一切活动（包括软件定义、分析、设计、编码、测试与正确性证明、维护与评价等）的方法、工具和管理的学科。

软件工程方法学既强调软件（一般指大型软件）开发的工程特征，又强调软件设计方法论的科学性、先进性。其基本内容包括：

结构化理论与方法。

模块技术与数据抽象。

软件测试与程序正确性证明。

软件分析与设计方法、工具及环境。

软件工程管理与质量评价。

1.2.2 软件工程的基本原理

1983 年 B.Weohm 提出了软件工程的七条基本原理。这七条基本原理是保证软件产品质量和开发效率的最小集合，又是相当完备的。现在虽然不能用数学方法严格证明是一个完备的集合，但是可以证明在此以前的 100 多条软件工程原理都可以由这七条原理组合与派生。这七条原理是：

1. 用分阶段的生命周期计划严格管理

这一条是吸取前人的教训而提出来的。统计表明，50%以上的失败项目是由于计划不周而造成的。在软件开发与维护的漫长生命周期中，需要完成许多性质各异的工作。这条原理意味着，应该把软件生命周期分成若干阶段，并相应制定出切实可行的计划，然后严格按照计划对软件的开发和维护进行管理。

在整个软件生命周期中应指定并严格执行六类计划：项目概要计划、里程碑计划、项目控制计划、产品控制计划、验证计划、运行维护计划。