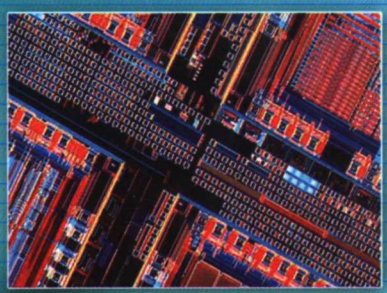
普通高等教育“十一五”规划教材

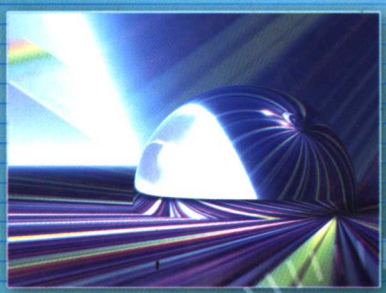
高等院校计算机技术系列教材

软件工程导论

刘怀亮　主编

潘如锋　郑杰鑫　沈金城　编著





冶金工业出版社

普通高等教育“十一五”规划教材
高等院校计算机技术系列教材

软件工程导论

刘怀亮　主编

潘如锋　郑杰鑫　沈金城　编著

北　京

冶金工业出版社

内 容 简 介

本书是根据普通高等教育“十一五”国家级规划教材的指导精神而编写的。

软件工程是软件工程专业的一门核心课程，也是计算机、信息、软件等相关专业的一门重要的计算机专业课程。本书由作者根据计算机等相关专业的教学计划，参考国内外多种教材及其他资料，理论联系实际，在教学和科研的实际过程中编写而成。

本书内容较为系统、全面，以传统软件工程过程为主线，介绍了软件工程的基本原理和技术方法，包括软件工程概述、系统分析、需求分析、总体设计、详细设计、程序编码、软件测试、软件维护，另外也介绍了面向对象技术、软件工程管理的相关知识。本书从基础入手，循序渐进，注重内容的可理解性和可操作性，提供了丰富的实例、习题和实训。

本书十分强调实践和应用，适合作为高等院校计算机专业及其相关专业的教材，也可供相关专业工程技术人员在实际开发过程中参考。

图书在版编目（CIP）数据

软件工程导论 / 刘怀亮主编；潘如锋，郑杰鑫，沈金城编著. —北京：冶金工业出版社，2007.6
普通高等教育“十一五”规划教材
ISBN 978-7-5024-4306-1

I. 软… II. ①刘…②潘…③郑…④沈… III. 软件工程—高等学校—教材 IV. TP311.5

中国版本图书馆 CIP 数据核字（2007）第 076321 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
ISBN 978-7-5024-4306-1
广州锦昌印务有限公司印刷；冶金工业出版社发行；各地新华书店经销
2007 年 6 月第 1 版第 1 次印刷
787mm × 1092mm 1/16； 18 印张； 415 千字； 280 页
35.00 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

一、关于本书

本书是根据普通高等教育"十一五"国家级规划教材的指导精神而编写的。

软件工程是用来指导计算机软件开发的工程科学。可以从工程化的角度来进行软件开发、测试和项目管理。软件工程的概念是于20世纪60年代提出的，在当时，软件开发面临很多问题，软件开发的进度一拖再拖，软件成本不断上升，质量得不到保证。人们希望能通过工程技术和管理方法使软件开发工程化。因此，软件工程逐步成为计算机科学与技术应用领域里一门重要的科学。软件工程发展到现在，对软件领域的发展起到了巨大的作用。

二、本书结构

本书一共分为11章，主要内容安排如下：

第1章：软件工程概述。主要介绍了软件工程的一些基本概念和相关知识，其中包括软件、软件工程、软件生命周期、软件生命周期模型，以及软件开发方法和开发工具等。

第2章：系统分析。主要介绍了基于计算机的系统和系统工程、可行性研究以及系统分析。

第3章：需求分析。主要介绍了需求分析的任务与原则、需求分析的过程及方法、需求规格说明与需求评审、需求管理及其工具等。

第4章：总体设计。主要介绍了总体设计的目标和任务、软件设计的原则、体系结构设计以及结构化设计方法。

第5章：详细设计。主要介绍了详细设计的任务与原则、处理过程设计、面向数据结构的设计方法以及详细设计说明书等。

第6章：面向对象技术。主要介绍了面向对象的基本概念、面向对象模型、面向对象分析、面向对象设计、面向对象分析与设计方法以及面向对象实现（包括面向对象程序设计和测试）。

第7章：程序编码。主要介绍了程序设计语言、编程规范与风格、程序效率、编程安全以及程序设计方法等程序编码知识。

第8章：软件测试。主要介绍了软件测试的基本概念、软件测试的基本方法、软件测试的过程与策略、软件的调试与排错以及软件测试的工具等内容。

第9章：软件维护。主要介绍了软件维护的基本概念、软件维护的特点、软件维护活动、软件维护的副作用、软件可维护性以及软件再工程技术。

第10章：软件工程管理。主要介绍了软件项目管理的相关概念和基本知识，重点介绍了软件项目计划与进度、软件质量管理和软件配置管理等内容。

第11章：上机实训。本书每一章都配有相应的上机实训，包括基础性、设计性和综合性等几种类型的实训。其中每个实训又包含实训概要、实训内容、实训过程和实训总结等

内容。

三、本书特点

本书在内容的编排、语言的叙述等方面都有其自身的一些特点。

（1）内容系统全面，结构清晰。先是软件工程概述，然后按照软件生命周期的各个阶段进行介绍，分别为系统分析、需求分析、总体设计、详细设计、程序编码、软件测试、软件维护，其中加入一章专门介绍面向对象技术，最后介绍软件工程管理的相关知识。

（2）描述简明易懂。本书从基本概念和原理出发，注重内容的可理解性，循序渐进，深入浅出；文字描述通俗易懂，简明扼要，重点突出。

（3）注重实训。注重实训是本书的最大特色。本书在阐述理论知识的同时配备相应的例题与详细的分析，易于学习，同时也便于教学。在每一章后面都有练习题，便于读者对本章知识进行巩固。本书还配有专门的上机实训，作为每一章的练习实训，通过实训可大大加深对所学知识的理解。

四、本书适用对象

本书十分强调实践和应用，适合作为高等院校计算机专业及其相关专业的教材，也可供相关专业工程技术人员在实际开发过程中参考。

本书由刘怀亮主编，潘如锋、郑杰鑫、沈金城参与编写。本书得以面世，许多同志对于本书的编写工作也提供了热心的帮助，在此表示衷心感谢！

由于编者水平有限，编写时间仓促，书中疏漏之处在所难免，欢迎广大读者对本书提出宝贵意见。联系方法如下：

电子邮箱：service@cnbook.net　　作者邮箱：great_liu@126.com

网址：www.cnbook.net

本书的电子教案和习题参考答案可从该网站下载，此外，该网站还有一些其他相关书籍的介绍，可以方便读者选购参考。

编　者

2007 年 5 月

目　录

第 1 章　软件工程概述

计算机系统由硬件系统和软件系统两大部分组成。计算机硬件和软件密切联系、相辅相成。硬件是计算机的物理设备，提供了计算机的可能性。软件是对硬件的支持和管理，使硬件在实际应用中发挥其巨大的潜能。

随着微电子技术和计算机技术的迅速发展，计算机硬件的性能和质量迅速提高，而其体积、功耗和成本却在不断下降。然而，和计算机硬件相比，计算机软件的开发效率却远远跟不上计算机应用的需求。所开发的软件功能越强大，规模越大，结构越复杂，人们的软件开发能力也越显得力不从心。这使得软件开发进度一拖再拖，软件成本逐年上升，软件质量得不到可靠保证，所开发的软件难以维护。软件生产率和软件质量满足不了计算机系统发展的需求，反而成为了制约计算机系统发展的关键因素。于是，自 20 世纪 60 年代末以来，人们开始重视软件开发过程、开发方法、开发工具和环境的研究，从而逐步形成了计算机科学技术领域中的一门新兴学科——计算机软件工程学，通常简称为软件工程。自此，计算机软件从最初的程序设计阶段过渡到软件系统阶段，最后发展到如今的软件工程阶段。软件工程强调软件开发的科学管理和规范化，对软件领域的发展起了很大的推动作用。

本章教学目标：

（1）理解与软件工程相关的一些基本的概念和知识，其中包括软件与软件的特点、软件工程的概念等。

（2）掌握软件生命周期的基本概念，熟悉一些典型的软件生命周期模型。

（3）了解有关软件开发方法和开发工具的知识。

1.1　软件

要正确地理解软件工程的概念，首先就要对软件进行正确的定义，认识软件的特点。另外，软件工程的产生与软件危机有很大的关系。

1.1.1　软件与软件的特点

软件是计算机系统中与硬件相互依存的另一部分。人们对软件的认识随着计算机技术的发展而不断深化。

计算机发展的初期，硬件的设计和生产是人们普遍关注的主要问题，而所谓的软件被简单地视为程序。软件在计算机系统中处于从属的地位。这时软件的生产采用的是个体的手工方式，程序设计只强调个人技巧，程序的目标特征是运行结果正确、速度快、节省空间等。这样一来，设计主要是靠程序员个人独立完成的，程序的质量完全取决于个人的编程技巧。其后，程序员之间的互助合作产生了所谓的程序说明，于是人们认为软件就是程序加上说明书。显然，这种原始的生产组织方式是无法适应生产力的飞速发展的。随着计算机系统的飞速发展，软件在计算机系统中所占的比重越来越大，软件的规模也越来越大，传统的软件生产方式已不适应发展的需要。工程学的基本原理和方法逐渐应用到软件设计

和生产中。软件开发被分成几个阶段，每个阶段都有严格的管理和质量检验，并在设计和生产过程中用书面文件作为共同遵循的依据。这些书面文件就是文档。文档是软件质量的基础，而程序则是文档代码化的表现形式。

现在，计算机软件被公认为是程序、数据及其相关文档的完整集合，即可表示为：

软件 = 程序+数据+文档

软件也可以简单地分为两部分：

（1）机器可以执行的程序和相关数据。

（2）不可执行的文档，包括软件开发、使用和维护全过程的相关信息和资料。

程序是按事先设计的功能和性能要求编写的指令系列。程序通常用程序设计语言描述，由程序设计语言翻译程序转换成机器指令在计算机上执行。程序的执行为系统和用户提供了期望的功能和性能。

数据是使程序能正常操纵信息的数据结构。一个程序对应的数据结构能使得该程序完全操纵它在运行时所需的必要信息。

文档是描述程序、数据和系统的设计、开发、维护以及使用的各种图文资料。它描述了系统的分析、设计、实现和维护的细节及其使用说明。文档具有永久性，可以存储在各种媒体上供人查阅。文档具有记录、通信交流、控制软件生产过程、管理维护软件、软件产品介绍等作用。文档的制定需遵循一定的规范。例如，我国国家标准局颁布的《计算机软件开发规范》、《计算机软件需求说明编制指南》、《计算机软件测试文件编制规范》、《计算机软件配置管理计划规范》等等。

同硬件相比，软件是一种特殊的工业产品，它具有下列一些特点：

（1）计算机软件是一种逻辑产品，它与物质产品有很大的区别。软件具有抽象性。人们可以把它记录在纸、内存、磁盘和光盘等各种存储介质上，但却无法看到软件本身的形态，必须通过观察、分析、思考和判断，才能了解它的功能、性能等特性。软件新产品是脑力劳动的结晶，总是以程序和文档的形式出现，通过计算机的运行才能体现它的功能和作用。

（2）软件的生产与硬件不同。软件产品的生产主要是研制。软件产品的开发过程中没有明显的制造过程，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，因此也产生了软件产品的保护问题。软件产品的知识产权应该在技术上和法律上得到必要的保障。

与硬件研制相比，软件开发更依赖于开发人员的业务素质和智力，以及开发人员的组织、合作和管理。软件的开发一般都是从头开始的，开发的成本、进度很难估计。软件在提交使用之前，尽管经过了严格的测试和试用，但仍然不能保证软件没有潜在的错误。所以，软件投入使用后，仍需要进行长期的维护。

（3）软件产品不会用坏，不存在硬件产品那样的机械磨损、老化等问题。但在使用过程中可能发生故障或为了适应硬件、运行环境及需求的变化而进行多次的修改，每一次的修改又可能引入新的错误，从而导致软件失效率升高，使得软件退化。随着时间的推移，软件运行环境和用户需求不断变化，当修改软件的成本变得难以接受时，软件就被“废弃”了。

如图 1-1 所示为软件和硬件的失效率曲线图，其中实线表示软件，虚线表示硬件。

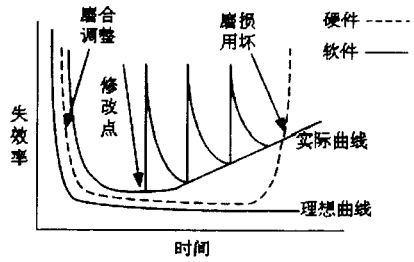


图 1-1 软件和硬件的失效率曲线图

（4）软件产品的生产主要是靠脑力劳动。目前大部分软件产品都是“定做”的，还未完全摆脱手工开发方式。近年来软件开发提出了许多新的方法，例如利用现成软件的复用技术、自动生成技术等，软件开发工具和软件开发环境的功能也不断增强。但总的来说，传统的手工艺开发方式仍然占据统治地位。

（5）软件费用不断增加，软件成本相当昂贵。软件本身是复杂的。软件的复杂性既可能来自它所反映的实际问题的复杂性，也可能来自程序逻辑结构的复杂性。软件产品的开发需要投入大量的、复杂的、高强度的脑力劳动，成本较高。软件产品的开发和使用依赖于一定的硬件条件和运行环境，而且软件的维护要比硬件复杂得多。

（6）软件工作涉及到各种社会因素。许多软件的开发和运行涉及法律法规、机构设置、体制运作和管理方式，以及人们的观念和心理等。人的因素往往成为软件开发的难点，直接影响到软件项目的成败。

软件技术从计算机诞生以来得到了巨大的发展，尽管软件产品的生产依然很难满足人们日益增长的需求，但无疑已经积累了丰富的软件开发经验和软件资源。目前要给软件做出科学的分类很难，而且从不同的角度出发，就会有不同的分类方法。通常有以下几种：

（1）按软件的功能进行划分，软件可以分为系统软件、支撑软件和应用软件。

（2）按软件规模进行划分，软件可分为微型、小型、中型、大型、较大型和巨大型。

（3）按软件工作方式划分，软件可分为实时处理软件、分时软件、交互式软件和批处理软件。

1.1.2 软件危机

从 20 世纪 60 年代末开始，“软件危机”一词在计算机界广为流传。事实上，软件危机几乎从计算机诞生的那一天起就出现了，只是到了后来才被普遍认识到。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。这些问题绝不仅仅是不能正常运行的软件才具有的，实际上，几乎所有软件都不同程度地存在这些问题。概括地说，软件危机包含下述两方面的问题：如何开发软件，以满足人们对软件的日益增长的需求；如何维护数量不断膨胀的软件。具体来说，软件危机主要有以下一些典型表现。

（1）对软件开发成本和进度的估计常常很不准确。由于软件的特殊性，不同类型的软件，其开发所需的工作量、成本往往差别很大。现有软件开发的经验和软件开发数据的不足，也使得计划难以制定，执行起来往往和实际情况有很大差距，以致软件开发经费预算经常突破，完成时间一再拖延。实际成本比估计成本高出一个数量级，实际进度比预期

进度拖延几个月甚至几年的现象也并不罕见。而为了赶进度和节约成本所采取的一些权宜之计又往往是以损害软件产品的质量为代价的。这些都会不可避免地引起用户的不满，从而降低了软件开发组织的信誉。

（2）用户对所交付的软件系统不满意的现象时有发生。在有限的时间里，软件开发人员常常还没清晰了解用户需求就已经着手编写程序了。软件开发人员和用户之间的信息交流往往很不充分，他们对软件功能的理解也有偏差，这样必然导致最终的软件产品偏离用户的实际需求。

（3）软件产品的质量往往靠不住。由于缺乏完善的确保软件质量的体系和措施，加上开发成本和进度等条件的限制，各种软件质量保证技术（复审、测试等）实际上很难贯彻软件开发的全过程，软件产品的质量难以保证。

（4）软件常常是不可维护的。维护是必不可少的，实际应用中的软件需要不断适应新的硬件环境和用户新的需求。许多软件在开发时没有考虑到将来的修改，发现错误以后很难改正。另外，开发过程常常缺乏统一的、公认的规范，软件开发人员按各自的风格工作，文档不够完整和规范，导致软件可维护性差。而不规范的小修小补往往只会引入更多的错误，也使软件最终成为不可维护的。

（5）软件文档资料通常不完整、不合格。软件文档资料是开发人员之间及开发人员与用户之间交流信息的依据，也是软件维护人员进行维护的依据。软件管理人员根据这些文档资料对软件开发工程进行管理和评价。然而开发组织通常对软件文档配置没有足够的重视，缺乏严格的文档制度，导致文档资料特别是与软件版本相匹配的文档资料不完整、不合格，给软件开发和维护工作带来了许多难以想象的困难和难以解决的问题，如软件开发进程、成本不可控制，软件维护、管理困难等。

（6）软件的价格昂贵，软件成本在计算机系统总成本中所占的比例逐年上升。随着微电子技术的迅速进步和生产自动化程度不断提高，计算机硬件成本逐年下降。然而软件开发需要大量的人力、物力和财力，随着对软件规模、数量和质量的要求不断提高，软件成本占计算机系统总成本的比例逐年上升。特别是软件维护成本迅速增加，已经占到软硬件总成本的40%到75%。

（7）软件开发生产率提高的速度，既跟不上硬件的发展速度，也远远跟不上日益增长的软件需求。

以上列举的仅仅是软件危机的一些比较明显的表现，可见在软件开发和维护过程中确实存在着严重的问题。这些问题不仅与软件自身的特点有关，也和软件开发与维护的方法有关，具体表现如下：

（1）软件的规模越来越大，结构越来越复杂。由于软件本身的复杂性和人类智力的局限性，存在人们无法解决的“复杂问题”。“复杂问题”的概念是相对的，会随着生产的进步而被解决，从而又出现新的“复杂问题”。

（2）软件开发管理困难。现代软件一般规模大，结构复杂，需要组织众多开发人员共同完成。因而软件开发管理困难，开发进度和软件质量难以控制，软件的可靠性无法保证。

（3）软件开发技术落后。计算机技术发展初期，人们只注重一些计算机理论问题的研究而不注重软件开发技术的研究，使得用户要求的软件复杂性与软件技术解决复杂性的

能力越来越不相适应。

（4）软件产品生产方式落后。软件开发仍过分地依靠程序设计人员在软件开发过程中的技巧和创造性，加剧了软件产品的个性化。

（5）软件开发工具落后。软件开发仍然缺乏高集成、高效率的开发环境和开发工具，因而软件生产率提高缓慢。自动化、智能化软件开发的目标仍非常遥远。

（6）用户对软件需求的描述不精确，软件开发人员对用户需求的理解有偏差。软件开发人员可能对用户的业务和系统工作环境不十分熟悉，加上前期问题定义工作做得不够，常导致最终软件产品与用户的需求不一致。另外，在软件的开发和维护关系问题上常存在错误的概念，使得后期维护困难，系统寿命缩短。

这一系列的问题得不到有效解决，软件的发展是没有出路的。软件危机的解决，需要从技术和组织管理两方面进行研究和采取措施。首先应该对软件有一个正确的认识，推广使用在实践中总结出来的开发软件的成功的技术和方法，研究探索更好、更有效的技术和方法。应该开发和使用更好的软件工具，构建功能强大的软件工程支撑环境。另外，软件开发应充分吸取和借鉴长期以来从事各种工程项目所积累的行之有效的原理、概念、技术和方法，采用科学的组织管理方法。软件工程正是从管理和技术两方面研究如何更好地开发和维护计算机软件的一门新兴学科。它用现代工程的原理、技术和方法进行软件的开发、管理、维护和更新，有效地缓解了软件危机所引发的种种问题。

1968年，北大西洋公约组织NATO（North Atlantic Treaty Organization）在联邦德国的一次学术会议上，首次提出了“软件工程”的概念，希望利用工程化的原则和方法来克服软件危机，从而形成了一门新兴的学科——软件工程学。

1.2 软件工程

软件工程是一个发展的概念，随着计算机的普及应用以及软件产业的不断发展，人们对软件工程的认识不断深化。软件工程的基本目标、基本原理和原则，是软件开发和研究人员对不断进步的软件工程实践的理论总结。

1.2.1 软件工程的定义

软件工程是一门工程学科，涉及软件生产的各个方面和整个过程。随着软件工程的不断发展和人们对软件和软件工程的深入认识，出现了对软件工程的各种各样的定义。

在首次NATO会议上Fritz Bauer给出的软件工程的定义是：软件工程是为了经济地获得可靠的和能在实际机器上高效运行的软件而确立和使用的一系列完善的工程原理（方法）。

Boehm对软件工程的定义为：软件工程是现代科学技术知识在设计和构造计算机程序中的实际应用，其中包括管理在开发、运行和维护这些程序的过程中所必需的相关文档资料。

1983年IEEE（国际电气与电子工程师协会）在其《IEEE软件工程标准术语》中对软件工程下的定义为：软件工程是开发、运行、维护和修复软件的系统方法。其中的“软件”是指计算机程序、方法、规则、相关的文档资料和程序运行所必需的数据。

1993年，IEEE给出了一个更加综合的定义：

（1）将系统的、规范的、可度量的工程化方法应用于软件的开发、运行和维护的全

过程。

（2）研究（1）中所提到的方法。

软件工程的定义虽多，但其主要思想都是在强调软件开发中应用工程化原则的重要性。这种工程化的思想一直贯穿需求分析、设计、实现和维护整个软件生命过程。软件工程是指导计算机软件开发和维护的一门工程学科。它应用计算机科学、数学及管理科学等原理，借鉴传统工程的原则、方法和经验来解决软件问题。软件工程以提高质量，降低成本为目的，采用了若干科学的、现代化的方法技术来开发软件，极大提高了软件开发生产的效率。软件工程所包含的内容也不是一成不变的，它必将随着软件系统开发和生产技术的发展而有所改变。

软件工程研究的主要内容包括软件开发技术和软件工程管理两个方面。软件工程管理主要是研究软件管理学、软件经济学和软件心理学等。软件开发技术主要研究软件开发方法学、软件开发过程及软件开发工具和环境。这三个部分称为软件工程的三要素。

软件工程中使用的各种方法是完成软件工程项目的技术手段，它们支持软件工程的各个阶段和环节，包括多方面的任务。例如，项目计划与估算、软件需求分析、系统总体结构设计、数据结构设计、软件测试以及维护等等。软件工程方法有其特有的一套质量保证标准，往往采用形式化和形象化的手段来描述，如特殊的形式化的语言或图形的表达方法。软件开发方法为软件开发提供了“如何做”的技术。传统方法学和面向对象方法学是目前使用得最广泛的两种软件工程方法学。

软件开发工具为软件工程方法提供了自动的或半自动的软件支撑环境，它是人类在开发软件的活动中智力和体力的扩展和延伸。人们开发出许多软件工具，能够支持软件工程方法和各种软件文档的生成等等。许多的软、硬件工具和软件工程数据库等按一定的方法或模型组织起来形成软件工具集，从而建立起一种称为计算机辅助软件工程（Computer-Aided Software Engineering）的软件开发支持系统，简称 CASE。

软件开发过程贯穿于软件开发的各个环节，它把软件工程方法和工具综合起来以开发出最终满足需求且达到工程目标的软件产品。这些活动主要包括系统的分析、设计、实现、确认以及支持等活动。软件开发过程定义了软件开发方法使用的顺序，要求交付的文档资料、为保证质量和协调变更所需要的管理等，它们是软件开发各个阶段完成的里程碑。在此过程中，项目管理人员可以对软件开发的质量、进度、成本等进行评估、管理和控制。

1.2.2 软件工程的基本目标

软件工程的基本目标是在给定的成本、进度等条件下，开发出满足用户需求的软件产品。软件项目的成功主要是要达到以下几个目标：

（1）付出较低的软件开发成本。

（2）达到用户所要求的软件功能。

（3）取得较好的软件性能。

（4）所开发的软件易于移植、可重用性好。

（5）需要较低的软件维护费用。

（6）能按时完成开发任务，及时交付使用。

在实际的软件开发项目中，要同时达到以上目标的理想程度几乎是不可能的。这些软

件工程目标之间相互联系、相互影响。其中有些目标之间是互补的关系，如易于维护和高可靠性之间、低开发成本与按时交付之间。还有一些目标则是彼此相斥的。例如，如果一味追求软件开发的低成本，很可能同时也降低了软件的可靠性和其他性能。若过于追求提高软件的性能，则很可能造成所开发出来的软件对具体硬件依赖太多，从而直接影响到软件的可移植性。如图 1-2 所示为软件工程目标之间的相互关系图。

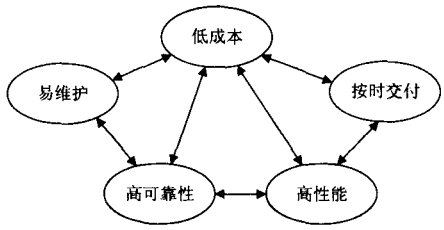


图 1-2 软件工程目标之间的相互关系图

软件工程的基本目标是人们判断软件开发管理方法优劣的衡量尺度。在实际应用中并不要求把以上目标同时实现，而是在实施软件开发项目的时候力图在以上目标的互补和冲突中取得一定程度的平衡。新的开发方法提出和使用，关键是要看它对满足哪些目标比现有的方法更为有利，而它所带来的不足并不影响总体目标的实现。

1.2.3 软件工程的基本原理

自从 1968 年提出“软件工程”这一术语以来，研究软件工程的学者和专家们从工程学的观点出发，提出了许多关于软件工程的准则和信条。其中著名的软件工程专家 B. W. Boehm 总结了 TRW 公司多年开发软件的经验，并综合了众多软件工程专家学者们的意见，于 1983 年提出了软件工程的 7 条基本原理。他认为这 7 条原理是确保软件产品质量和开发效率的原理的最小集合，它们相互独立、缺一不可，同时又是相当完备的。人们虽然很难用数学方法严格证明它们的完备性，但实际上在此之前已经提出上百条软件工程原理都可以由这七条基本原理的任意组合蕴含或派生。软件工程的这些基本原理概括了软件工程的方方面面，在软件工程实践中颇具指导意义。

以下是软件工程 7 条原理的简要概括。

1. 用分阶段的生命周期计划严格管理

在软件工程的漫长的生命周期过程中，需要完成许多不同性质的工作。应该把软件工程的生命周期划分成若干个阶段，并针对每一阶段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工作进行组织管理。B. W. Boehm 认为，在软件的整个生命周期过程中应该制定并严格执行六类计划，它们是项目概要计划、里程碑计划、项目控制计划、产品控制计划、验证计划和运行维护计划。据统计，在不成功的软件项目中，大约有一半是由于计划不周造成的。

2. 坚持进行阶段评审

软件的质量保证工作不能等到编码阶段结束之后再进行。大部分错误都是在编码之前造成的。据统计，软件设计错误约占软件错误的 63%，而编码错误仅占 37%。另外，错误发现与改正得越晚，所付出的代价也越高。因此，坚持进行严格的阶段评审，及时发现并改正软件开发过程中的错误，是一条必须始终坚持的重要原理。

3. 实行严格的产品控制

在软件开发过程中，需求的变更往往是难免的，但决不能因此随意改变需求。应该依靠科学的产品控制技术来处理用户提出的改变需求的要求。需求的改变必须保持软件各个配置成分的一致性，实行严格的产品控制，其中主要是实行基准配置管理（又称为变动控制）。基准配置即基线配置，是指经过阶段评审后的软件配置成分，即各阶段产生的文档或程序代码等。一切修改软件的建议，特别是涉及基本配置的修改建议，都必须按规程进行严格的评审，评审通过后才能实施。

4. 采用现代程序设计技术

从软件工程概念提出以来，人们一直致力于研究新的程序设计技术。新的面向对象技术与结构化程序设计技术相比，软件的开发效率、可维护性、可重用性等均有一定程度的提高。实践表明，采用先进的程序设计技术既可以提高软件开发与维护的效率，又可以提高软件的质量。

5. 结果应该能清楚地审查

由于软件产品是一种看不见摸不着的逻辑产品，软件开发小组的工作进展情况可见性差，难以评价和管理。这时应该提高软件开发过程的可见性，把定性的标准转化为定量的标准。应根据软件开发项目的总目标和完成期限，明确地规定出软件开发小组的责任和产品标准，从而能清楚地审查每一阶段所得到的结果。

6. 开发小组的人员应该少而精

软件开发小组人员的素质和数量是影响软件质量和开发效率的重要因素。素质高的人员的开发软件的效率可能比素质低的人员要高几倍至几十倍，而且素质高的人员所开发的软件中的错误也要少得多。而软件开发中开发小组的人数并不是越多越好，随着人数的增加，人员之间交流情况、讨论问题的通信开销将急剧增加。人员之间也可能会由于误解等原因增加出错的概率。显然，这并不能真正有效提高软件开发的效率。

7. 承认不断改进软件工程实践的必要性

遵循上述七条基本原理，已经能很好地实现软件的工程化生产。但软件工程不能停留在已有的技术水平上，应积极主动地采纳新的软件技术，注意不断总结经验，例如，收集进度和资源耗费数据，收集出错类型和问题报告数据等等。这些数据不仅可以用来评价新的软件技术的效果，而且可以用来指明必须着重开发的软件工具和应该优先研究的技术。把承认不断改进软件工程实践的必要性作为软件工程的基本原理，体现了人们发展的眼光和智慧。

1.2.4 软件工程的原则

（1）抽象（Abstraction）。抽象是指抽取事物最基本的特征和行为，忽略与问题无关或关系甚少的其他细节。通常采用分层次抽象的方法，自顶向下、逐层细化地控制软件开发过程的复杂性。抽象增强了软件的可理解性，并有利于软件开发过程的管理。

（2）模块化（Modularity）。模块化就是把一个问题划分成若干个较小的、较易解决的模块，每个模块完成一个子功能，将这些模块组装成一个整体即可完成指定的功能。模块是程序中相对独立的成分，是独立的编程单位，应该具有良好的接口定义。例如C语言中的函数、Java语言中的类等。模块化原则有助于表示复杂的系统，也有利于信息隐藏和