21世纪高等学校计算机规划教材

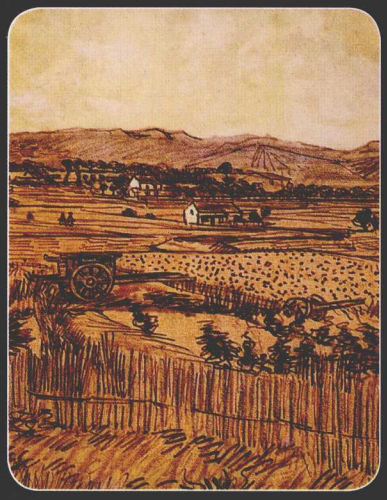
21st Century University Planned Textbooks of Computer Science

软件工程

Software Engineering

郑人杰 马素霞 麻志毅 编著

- 软件工程领域的经典教材
- 掌握软件开发的必经之路
- 软件专业人员的良师益友
- 投身软件事业的立足根基



名家系列

人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

软件工程

Software Engineering

郑人杰 马素霞 麻志毅 编著

名家系列

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

软件工程 / 郑人杰，马素霞，麻志毅编著. -- 北京
：人民邮电出版社，2009.11

21世纪高等学校计算机规划教材
ISBN 978-7-115-21026-5

Ⅰ. ①软… Ⅱ. ①郑… ②马… ③麻… Ⅲ. ①软件工程－高等学校－教材 Ⅳ. ①TP311.5

中国版本图书馆CIP数据核字(2009)第162050号

内 容 提 要

本书根据CC 2004对软件工程课程的要求，从软件的开发、维护和管理等方面阐述了软件工程的基本概念和常用方法。内容包括：软件工程基础、结构化软件开发方法、面向对象软件开发方法、软件生存期模型与软件体系结构、软件维护与管理。各章节均结合实例讲解，使读者易于理解和掌握。

本书可作为高等院校计算机专业或信息类相关专业本科生或研究生教材，也可作为软件开发人员的参考书。

21世纪高等学校计算机规划教材

软件工程

◆ 编　　著　郑人杰　马素霞　麻志毅
责任编辑　滑　玉
执行编辑　武恩玉

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街14号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市海波印务有限公司印刷

◆ 开本：787×1092　1/16
印张：25.5
字数：669千字　　2009年11月第1版
印数：1－3 000册　　2009年11月河北第1次印刷

ISBN 978-7-115-21026-5/TP

定价：39.80元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前 言

软件工程是计算机科学的重要分支，又是最为年轻的工程学科。然而，它却随着信息化社会的到来和计算机在人们生产和生活各个领域的广泛应用越来越显示出重要的作用。

许多高等学校计算机专业的毕业生工作多年以后，回顾他们在校所学的专业课时，往往提到软件工程课程是工作中最为实用的课程之一。事实上，在校学生学好这门课也将使他们在今后的职业生涯中受益。出于这一认识，本书作者深深地理解组织编写好这一教材的重大责任。为此在编写过程中开展了多次讨论和交流，并且吸取了以前的教学经验和读者对过去软件工程教材的反馈意见。

本书编写时作者力图遵循的原则如下。

（1）既要强调和突出基本概念、基本方法，又要尽可能使材料内容的组织符合学生的认识规律，由浅入深，由具体到一般。

（2）既要认真对待传统的经典开发方法（如结构化方法），又要兼顾当前流行和广为采用的开发方法（如面向对象方法）。

（3）既要充分重视技术性内容，使其作为初学者必须掌握的知识，并且占有足够的分量，同时也要兼顾软件工程实践中必不可少的管理知识（如本书的后三章），而其篇幅又不可过多。

参与本书编写的作者除本人外，还有华北电力大学的马素霞教授和北京大学的麻志毅教授。他们在软件工程的教学和软件开发的实践方面均有丰富的经验。

作者十分欢迎来自各方面的意见和建议，也希望吸收这些意见和建议后，在再版修订时采纳。

郑人杰

2009 年 4 月

目 录

第 1 部分 软件工程基础

第 2 部分 结构化软件开发方法

第 3 部分 面向对象软件开发方法

第 4 部分 软件生存期模型与软件体系结构

第 5 部分 软件维护与软件管理

第 1 部分　软件工程基础

第 1 章 软件及软件工程介绍

计算机技术经过了 50 年的发展历程，取得了突飞猛进的发展。计算机的应用领域已从单纯的科学计算发展到军事、经济、教育、文化等社会生产及生活的各个方面，推动了其他行业及领域的发展，改变了人们学习、工作及生活方式。进入 21 世纪，人类已从工业社会跨入了信息社会。

计算机软件系统是信息化的重要组成部分。计算机软件已形成了独立的产业，成为国民经济新的增长点和重要支柱。软件工程在软件开发中起着重要的作用，对软件产业的形成及发展起着决定性的推动作用。本章对软件、软件产业及软件工程相关的概念，软件开发的过程及方法进行简要介绍。

1.1　软件与软件危机

1.1.1　软件的作用

20 世纪 80 年代初我国大学生中知道软件的人并不多，甚至很多人从未听说过这个词，即使是当初软件专业毕业的学生也不曾想到软件的发展速度如此之快。今天的软件已无处不在，渗透到了各个行业之中。随着计算机大量进入家庭，计算机已经成为我们日常生活、学习和工作都离不开的工具，同时也改变了人们的学习方式、交流方式、思维方式及商业模式。

计算机软件已经成为世界舞台上最为重要的科技领域，商业、科学和工程都离不开软件技术。现在的软件技术具有产品和产品生产载体的双重作用。作为产品，软件显示了由计算机硬件体现的计算能力，扮演着信息转换的角色：产生、管理、查询、修改、显示或者传递各种不同的信息。而作为产品生产的载体，软件提供了计算机控制（操作系统）、信息通信（网络），以及应用程序开发和控制的基础平台（软件工具和环境）。

计算机软件的地位在 50 多年的时间中发生了巨大变化。硬件性能的极大提高、计算机结构的巨大变化、内存和存储容量的扩大，还有种类繁多的输入和输出方法都使得计算机系统的结构变得更加复杂，功能更加强大。计算机硬件的发展会受到物理极限的制约，而计算机软件的复杂程度却没有极限，复杂的结构和功能可以产生惊人的效果，具有无限的潜力。

1.1.2 软件的概念及特性

1. 软件的概念

虽然软件对于现代的人并不陌生，但很多人对于软件的理解并不准确，“软件就是程序，软件开发就是编程序”的这种错误观点仍然存在。软件并不只是包括可以在计算机上运行的程序，与这些程序相关的文件一般也被认为是软件的一部分。

一般可以将软件划分为系统软件、应用软件和介于这两者之间的中间件。其中系统软件为计算机使用提供最基本的功能，但是并不针对某一特定应用领域。而应用软件则恰好相反，不同的应用软件根据用户和所服务的领域提供不同的功能。

对于计算机软件的概念，现在尚无一个统一的定义。一种公认的传统定义为：软件是计算机系统中与硬件相互依存的另一部分，软件包括程序、数据及其相关文档的完整集合。其中，程序是按事先设计的功能和性能要求执行的指令序列；数据是使程序能正常操纵信息的数据结构；文档是与程序开发、维护和使用有关的图文材料。

在结构化程序设计时代，程序的最小单位是函数及子程序，程序与数据是分离的；在面向对象程序设计时代，程序的最小单位是类，在类中封装了相关的数据及指令代码。

2. 软件的特性

当今已有的人工制品千千万万，不可胜数，然而计算机软件却与任何传统的制造业产品不同，它具有许多突出的特性。

（1）形态特性。软件是无形的、不可见的逻辑实体。度量常规产品的几何尺寸、物理性质和化学成分对它却是毫无意义的。但绝不会因此否定它的存在和降低它的价值。

（2）智能特性。软件是复杂的智力产品，它的开发凝聚了人们的大量脑力劳动，它本身也体现了知识、实践经验和人类的智慧，具有一定的智能。它可以帮助我们解决复杂的计算、分析、判断和决策问题。

（3）开发特性。尽管已经有了一些工具（也是软件）来辅助软件开发工作，但到目前为止尚未实现自动化。软件开发中仍然包含了相当分量的个体劳动，使得这一大规模知识型工作（large scale knowledge work）充满了个人行为和个人因素。

传统制造业的工艺都已经相当成熟，早已摆脱了作坊式的手工生产，大规模采用自动化的生产。大多数的软件产品是根据用户的需求进行定制开发的个性化产品，虽然一直梦想软件的生产能够像硬件生产那样基于已有的零部件进行组装，但与这一目标还有相当长的距离。

（4）质量特性。软件产品的质量控制存在着一些实际困难，难于克服，表现在以下几个方面。

① 软件的需求在软件开发之初常常是不确切的，也不容易确切地给出，并且需求还会在开发过程中出现变更，这就使软件质量控制失去了重要的可参照物。

② 软件测试技术存在不可克服的局限性。任何测试都只能在极大数量的应用实例数据中选取极为有限的数据，致使我们无法检验大多数实例，也使我们无法得到完全没有缺陷的软件产品。

③ 已经长期使用或多次反复使用的软件没有发现问题，但这并不意味着今后的使用总不会出现问题。

这一特性提醒我们：一定要警惕软件的质量风险，特别是在某些重要的应用场合，需要提前准备好应对策略。

（5）生产特性。与硬件或传统的制造业产品的生产完全不同，软件一旦设计开发出来，如果需要提供多个用户，它的复制十分简单，其成本也极为有限。正因为如此，软件产品的

成本主要是设计开发的成本，同时也不能采用管理制造业生产的办法来解决软件开发的管理问题。

（6）管理特性。由于上述的几个特点，使得软件的开发管理显得更为重要，也更为独特。这种管理可归结为对大规模知识型工作者的智力劳动管理，其中包括必要的培训、指导、激励、制度化规程的推行、过程的量化分析与监督，以及沟通、协调，甚至是过程文化的建立和实施。

（7）环境特性。软件的开发和运行都离不开相关的计算机系统环境，包括支持它的开发和运行的相关硬件和软件。软件对于计算机系统的环境有着不可摆脱的依赖性。

（8）维护特性。软件投入使用以后需要进行维护，但这种维护与传统产业产品的维护概念有着很大差别，如建筑物、机械和电子产品的维修大都是由于使用（也包括非正常使用）中造成了材料的老化、腐蚀或是机械性的磨损等，有待于通过维修恢复其功能或性能，而软件产品使用中出现的问题并非是使用造成的，也不是使用时间久形成的，软件维护中需要做的往往是要修正开发时遗留下来、隐蔽的那些在特定运行条件下才暴露的缺陷。软件维护也可能是为了扩展与提升软件的功能或性能，以及为了适应运行环境的变更。

对软件的维护往往不能像硬件那样通过更换零件来解决，而需要对不适应的部分软件进行修改。

（9）废弃特性。当软件的运行环境变化过大，或是用户提出了更大、更多的需求变更，如果再对其实施适应性维护已不划算，该软件将走到它的生存期终点而被废弃（或称退役），此时用户将考虑采用新的软件代替。因此，与硬件不同，软件并不是由于被“用坏”而被废弃的。

（10）应用特性。软件的应用极为广泛，如今它已渗入国民经济和国防的各个领域，现已成为信息产业、先进制造业和现代服务业的核心，占据了无可取代的地位。

1.1.3　软件危机

20 世纪 60 年代，计算机已经在很多行业应用，解决问题的规模及难度逐渐增加，由于软件本身的特点及软件开发方法等多方面问题，软件的发展速度远远滞后于硬件的发展速度，不能满足社会日益增长的软件需求。软件开发周期长、成本高、质量差、维护困难，导致 60 年代末软件危机的爆发。

这些矛盾表现在具体的软件开发项目上，最突出的实例就是美国 IBM 公司在 1963—1966 年开发的 IBM 360 机的操作系统。这个项目的负责人 F.D.Brooks 事后总结了他在组织开发过程中的沉痛教训时说：“……正像一只逃亡的野兽落到泥潭中做垂死的挣扎，越是挣扎，陷得越深。最后无法逃脱灭顶的灾难，……程序设计工作正像这样一个泥潭，……一批批程序员被迫在泥潭中拼命挣扎，……谁也没有料到问题竟会陷入这样的困境……”。

除了软件本身的特点，软件危机发生的原因主要有以下几个方面。

（1）缺乏软件开发的经验和有关软件开发数据的积累，使得开发工作的计划很难制订。主观盲目地制订计划，往往与实际情况相差太远，致使常常突破经费预算，工期一拖再拖。而且对于软件开发工作，给已经拖延了的项目临时增加人力只会使项目更加拖延。

（2）软件人员与用户的交流存在障碍，除了知识背景的差异，缺少合适的交流方法及需求描述工具也是一个重要原因，这使得获取的需求不充分或存在错误，在开发的初期难以发现，存在的问题往往在开发的后期才暴露出来，使得开发周期延长，成本增高。

（3）软件开发过程不规范，缺少方法论和规范的指导，开发人员各自为战，缺少整体的规划

和配合，不重视文字资料工作，软件难以维护。

（4）随着软件规模的增大，其复杂性往往会呈指数型增长。

（5）缺少有效的软件评测手段，提交用户的软件质量差，在运行中暴露出大量的问题，轻者影响系统的正常使用，重者发生事故，甚至造成生命财产的重大损失。

与40年前相比，当前的软件开发技术已经有了长足的进步，早期的独立程序员也已经被专业的软件开发团队所代替。但由于目前软件要解决的问题越来越复杂和庞大，同时，用户对软件的质量和开发周期提出了更高的要求，因此软件开发人员依然面临开发周期长、成本居高不下、无法达到零错误等问题。

1.2 软件工程及其基本原理

1.2.1 软件工程的概念

为了克服软件危机，1968年10月在北大西洋公约组织（NATO）召开的计算机科学会议上，Fritz Bauer首次提出“软件工程”的概念，试图将工程化方法应用于软件开发。

许多计算机和软件科学家尝试，把其他工程领域中行之有效的工程学知识运用到软件开发工作中来。经过不断实践和总结，最后得出这样的结论：按工程化的原则和方法组织软件开发工作是有效的，是摆脱软件危机的一个主要出路。

虽然软件工程概念的提出已有40年，但直到目前为止，软件工程概念的定义并没有统一。在NATO会议上，Fritz Bauer对软件工程的定义是：“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，而建立和使用的健全的工程原则。”除了这个定义，比较有代表性的定义如下。

Boehm给出的定义是：“运用现代科学技术知识来设计并构造计算机程序及为开发、运行和维护这些程序所必需的相关文件资料”。此处“设计”一词广义上理解应包括软件的需求分析和对软件进行修改时所进行的再设计活动。

1983年IEEE给出的定义是：“软件工程是开发、运行、维护和修复软件的系统方法”。其中“软件”的定义为：计算机程序、方法、规则、相关的文档资料以及在计算机上运行时所必需的数据。

后来尽管又有一些人提出了许多更为完善的定义，但主要思想都是强调在软件开发过程中应用工程化原则的重要性。

概括地说，软件工程是指导计算机软件开发和维护的工程学科。采用工程的概念、原理、技术和方法来开发与维护软件，把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来，以经济地开发出高质量的软件并有效地维护它，这就是软件工程。

1.2.2 软件工程的目标

软件工程的目标是运用先进的软件开发技术和管理方法来提高软件的质量和生产率，也就是要以较短的周期、较低的成本生产出高质量的软件产品，并最终实现软件的工业化生产。生产率与成本密切相关，生产率的提高往往意味着成本的下降，开发周期的缩短。生产率与质量之间也有着内在的联系，表面上看，追求高质量会延长软件开发时间，并因此增加了成本，似乎降低了

生产率。但如果生产的软件质量差，虽然开发的时间可能缩短，但之后可能会造成返工，总的开发时间可能会更长。即使不返工，也无疑会增加维护代价。另外，低质量的软件很有可能给用户造成重大损失，这方面的历史教训已有很多。

实际上，质量和生产率之间不存在根本的对立，好的软件工程方法可以同时提高质量与生产率。

可以用功能性、可靠性、可使用性、效率、可维护性和可移植性 6 个特性来衡量软件的质量。功能性是指软件所实现的功能达到它的设计规范和满足用户需求的程度；可靠性是指在规定的时间和条件下，软件能够正常维持其工作的能力；可使用性是指为了使用该软件所需要的能力；效率是指在规定的条件下用软件实现某种功能所需要的计算机资源的有效性；可维护性是指当环境改变或软件运行发生故障时，为了使其恢复正常运行所做努力的程度；可移植性是指软件从某一环境转移到另一环境时所做努力的程度。

不同类型的应用系统对软件质量的要求不同，如对实时系统来说，其可靠性和效率比较重要；对生存期较长的软件来说，其可移植性、可维护性比较重要。

在实际的软件开发中，企图使所有的质量目标同时达到理想的程度往往是不现实的。因为有些质量目标之间是彼此冲突的，如若只考虑提高效率，很可能同时也降低了软件的可靠性、可维护性和可移植性。质量目标之间的关系如表 1-1 所示。

表 1-1　　质量目标之间的关系

	功能性	可靠性	可使用性	效率	可维护性	可移植性
功能性		△			△	
可靠性				▽		△
可使用性				▽	△	△
效率		▽			▽	▽
可维护性		△		▽		△
可移植性		▽		▽		

其中，△表示有利影响，▽表示不利影响。

1.2.3　软件工程的基本原理

自从“软件工程”诞生，提出运用工程学的基本原理和方法来组织和实施软件生产后，又发展了与软件有关的心理学、生理学、经济学等方面的学科。在此期间，研究软件工程学的科学家们陆续提出了 100 多条有关软件工程的准则，1983 年美国 TRW 公司 B.W.Boehm 将它们概括为著名的软件工程 7 条基本原理。

1. 按软件生存周期分阶段制订计划并认真实施

软件从定义、开发、运行和维护，直到最终被废弃，要经历一个很长的时间，通常称这样一个时期为软件生存周期。在软件生存周期中需要完成许多不同性质的工作，所以应把软件生存周期划分为若干阶段，为每一阶段规定若干任务，制订出可行的计划，并按照计划对软件的开发和维护活动进行管理。不同层次的管理人员都必须严格按照计划各尽其职地管理软件的开发和维护工作，不应受客户或上级人员的影响而擅自背离预定计划。

2. 坚持进行阶段评审

软件的质量控制工作不能等到编码阶段结束之后再进行。因为大部分错误是在编码之前造成

的，而且错误发现得越晚，为改正它所需付出的代价就越大，因此，在每个阶段都要进行严格的评审，以尽早发现在软件开发过程中产生的错误。

3. 坚持严格的产品控制

在软件开发过程中要及时识别所有影响软件产品质量的因素，并在开发过程中严格加以控制，如对于需求变更的控制。由于外界环境的变化或软件工作范围的变化，在软件开发过程中需求的变更是在所难免的，必须依靠科学的产品变更控制机制来顺应需求的变更。就是说，当变更需求时，为了保持软件各个配置成分的一致性，必须实施严格的产品变更审查和基线配置管理，确保变更的正确性和有效性。

4. 使用现代程序设计技术

随着软件产品规模和复杂性的不断增加，对于软件开发的方法和工具提出了更高的要求。先进的程序设计技术，如面向对象的程序设计技术、面向方面的程序设计技术、模型驱动开发方法等，能够以较低的费用、适宜的时限，生产出高质量的软件产品。实践表明，采用先进的程序设计技术可提高软件开发的生产率，还可提高软件产品的可维护性。

5. 明确责任

为了提高软件开发过程的可见性并有效地进行管理，应当根据软件开发项目的总目标、任务分解结构和完成期限，规定开发组织的责任和产品标准，使得工作结果能够得到清楚的审查。

6. 用人少而精

合理安排软件开发小组人员的原则是参与人员应当少而精，即小组的成员应当具有较高的素质，且人数不应过多。人员素质高能大大提高软件开发的生产率，明显减少软件中的错误。开发小组人数减少，可以降低因交流开发进展情况和讨论遇到的问题而造成的通信开销。

7. 不断改进开发过程

软件开发过程定义了软件工程方法使用的顺序、要求交付的文档资料、为保证质量和协调变化所需要的管理以及软件开发各个阶段应采取的活动和必要的评审。为保证软件开发的过程能够跟上技术的进步，必须不断地灵活改进软件工程过程。为了达到这个要求，应当积极主动地采用新的软件开发技术，注意不断总结经验。此外，需要注意收集和积累出错类型、问题报告等数据，用以评估软件技术的效果和软件人员的能力，确定必须着重开发的软件工具和应当优先研究的技术。

1.3 软件生命周期

如同任何其他事物一样，软件也有一个孕育、诞生、成长、成熟和衰亡的生存过程，我们称这个过程为软件生命周期或软件生存期。概括地说，软件生存期由软件定义、软件开发和运行维护 3 个时期组成，每个时期又可划分为若干个阶段。

软件定义时期的主要任务是解决“做什么”的问题，即确定工程的总目标和可行性；导出实现工程目标应使用的策略及系统必须完成的功能；估计完成工程需要的资源和成本；制订工程进度表。该时期的工作也就是常说的系统分析，由系统分析员完成。它通常又被分为 3 个阶段：问题定义、可行性研究和需求分析。

软件开发时期的主要任务是解决“如何做”的问题，即具体设计和实现在前一个时期定义的软件，通常由概要设计、详细设计、编码和测试 4 个阶段组成。

软件运行维护时期的主要任务是使软件持久地满足用户的需要，通常有 4 类维护活动：改正性维护，也就是诊断和改正在使用过程中发现的软件错误；适应性维护，即修改软件以适应环境的变化；完善性维护，即根据用户的要求改进或扩充软件，使它更完善；预防性维护，即修改软件为将来的维护活动预先做准备。

通常，这些活动与要交付的产品是密切相关的，如开发文档、源程序代码、用户手册等。里程碑可以用来标识项目进展状态的事件。例如，完成用户手册的事件可以是里程碑。由于管理人员用里程碑来评价软件开发的进展情况，所以，里程碑对于软件开发的管理非常重要。

开发过程中的典型文档包括以下几项。

① 软件需求规格说明书：描述将要开发的软件做什么。

② 项目计划：描述将要完成的任务及其顺序，并估计所需要的时间及工作量。

③ 软件测试计划：描述如何测试软件，使之确保软件应实现规定的功能，并达到预期的性能。

④ 软件设计说明书：描述软件的结构，包括概要设计及详细设计。

⑤ 用户手册：描述如何使用软件。

下面简要介绍上述各个阶段所要完成的基本任务。

（1）问题定义与可行性研究

本阶段要回答的关键问题是“到底要解决什么问题？在成本和时间的限制条件下能否解决问题？是否值得做？”为此，必须确定要开发软件系统的总目标，给出它的功能、性能、约束、接口、可靠性等方面的要求；由软件分析员和用户合作，探讨解决问题的可能方案，针对每一个候选方案，从技术、经济、法律、用户操作等方面，研究完成该项软件任务的可行性，并对可利用的资源（计算机硬件，软件，人力等）、成本、可取得的效益、开发的进度做出估算，制订出完成开发任务的实施计划，连同可行性研究报告，提交管理部门审查。

（2）需求分析

本阶段要回答的关键问题是“目标系统应当做什么？”为此，必须对用户要求进行分析，明确目标系统的功能需求和非功能需求，并通过建立分析模型，从功能、数据、行为等方面描述系统的静态特性和动态特性，对目标系统做进一步的细化，了解系统的各种需求细节。基于分析结果，软件分析人员和用户共同讨论决定：哪些需求是必须满足的，对其加以确切地描述，然后编写出软件需求规格说明或系统功能规格说明，确认测试计划和初步的系统用户手册，提交管理机构进行分析评审。

（3）软件设计

设计是软件工程的技术核心。本阶段要回答的关键问题是“如何做出目标系统？”为此，必须在设计阶段中制定设计方案，把已确定的各项需求转换成相应的软件体系结构。结构中的每一组成部分都是意义明确的构件，此即所谓概要设计。进而具体描述每个构件所要完成的工作，以及完成的顺序，为源程序编写打下基础，此即所谓详细设计。所有设计中的考虑都应以设计规格说明的形式加以描述。此外，基于设计结果编写单元测试计划和集成测试计划，再执行设计评审。

（4）程序编码和单元测试

本阶段要解决的问题是“正确地实现已做的设计”，即“如何编写正确的、可维护的程序代码？”为此，需要选择合适的编程语言，把软件设计转换成计算机可以接受的程序代码，并对程序结构中的各个模块进行单元测试，然后运用调试的手段排除测试中发现的错误。要求编写出的程序应当是结构良好、清晰易读的，且与设计相一致。