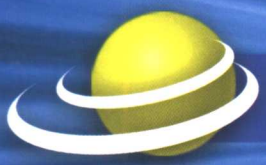

高等学校计算机科学与技术教材

- 原理与技术的完美结合
- 教学与科研的最新成果
- 语言精炼，实例丰富
- 可操作性强，实用性突出

软件工程

□ 万江平　编著

清华大学出版社　●　北京交通大学出版社

高等学校计算机科学与技术教材
国家自然科学基金(70471091)资助
高等学校博士点专项科研基金(20030561024)资助

软 件 工 程

万江平 编著

清华大学出版社
北京交通大学出版社
·北京·

内 容 简 介

本书运用了前沿的知识科学和系统科学来处理软件和软件工程的复杂性。使用了 IEEE 的软件工程知识体系（SWEBOK2004）、著名的软件过程成熟度模型（CMM/CMMI）和 NASA 的软件开发实践经验。针对中国的实际情况和软件产业的发展，突出了软件质量管理和软件过程改进，采用了面向对象技术（包括 UML）、项目管理等软件知识技术来解决中国软件工程教育和培训的实际问题。

本书内容丰富，结构合理，适合计算机及相关专业的本科生、研究生，以及软件技术和管理人员使用。

图书在版编目(CIP)数据

软件工程/万江平编著. —北京：清华大学出版社；北京交通大学出版社，2006.8
（高等学校计算机科学与技术教材）
ISBN 7－81082－733－2

Ⅰ. 软…　Ⅱ. 万…　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（2006）第 040272 号

责任编辑：谭文芳
出版发行：清 华 大 学 出 版 社　邮编：100084　电话：010－62776969　http://www.tup.com.cn
　　北京交通大学出版社　邮编：100044　电话：010－51686414　http://press.bjtu.edu.cn
印 刷 者：北京东光印刷厂
经　　销：全国新华书店
开　　本：185×260　印张：24.5　字数：627 千字
版　　次：2006 年 8 月第 1 版　2006 年 8 月第 1 次印刷
书　　号：ISBN 7－81082－733－2/TP·273
印　　数：1～4000 册　定价：36.00 元

本书如有质量问题，请向北京交通大学出版社质监组反映。对您的意见和批评，我们表示欢迎和感谢。
投诉电话：010－51686043，51686008；传真：010－62225406；E-mail：press@center.bjtu.edu.cn。

前　言

IEEE计算机学会（IEEE Computer Society，IEEECS）和计算机协会（Association for Computing Machinery，ACM）正致力于一项重要的项目：开发软件工程知识体系指南（the Software Engineering Body of Knowledge，SWEBOK，http://www.swebok.org/）。这清晰地说明一种知识的体系是发展一种专业所必要的，因为它代表对这个学科的内容达成了广泛的共识。SWEBOK指南把软件工程学科的本体知识分为10个知识域，各种重要概念之间的区别在每个知识域描述中进行阐述，方便读者迅速查找所感兴趣的知识域主题。本书认为用系统科学（结构复杂性理论）和知识系统工程（知识技术）能够很好地说明SWEBOK机理和规律，这是因为软件技术所面对的是信息的生成和处理，设计相应的结构、算法和程序，而处理过程的输入和输出之间是高度非线性的，还没有物理定律能够驾驭。

直到20世纪80年代中期，大多数研究人员都有一个信念：只要有好的软件开发方法和工具就可以开发出高质量的软件。直至大约1986年，美国工业界和军方才开始认识到软件过程的不断改进才是提高软件质量的第一要素。美国卡内基·梅隆大学的美国软件工程研究所（Software Engineering Institution，SEI）提出了SW-CMM（Software Capability Maturity Model，软件能力成熟度模型）来对软件质量乃至软件企业进行管理。SW-CMM是软件企业追求高质量发展的指南，它以几十年产品质量概念和软件工业经验教训为基础，为企业软件过程能力不断走向成熟提供了有效的步骤和框架。同时，美国航天局（National Astronautics Space Agency，NASA）的美国软件工程实验室（Software Engineering Laboratory，SEL）根据航天系统软件开发经验也总结出如下软件过程改进的要点：① 过程改进由内部目标和经验驱动；② 每个域按其特点用不同的方法对待；③ 按照软件组织相关的度量赋予环境特性；④ 成功与否的最终度量是软件组织交付的产品或服务的改进。

面向对象技术（Object-Oriented Technology）具有以下突出的优点：① 符合人们的思维习惯；② 对象技术开发的系统稳定性好；③ 系统的重用性好；④ 系统具有良好的可维护性。其理论基础是类型理论：人们经常使用3种组织方法来理清思路：① 区分对象及其特征；② 区分对象整体及其组成部分；③ 区分对象的类。当前，分布式网络计算的模式、开发环境、通信协议、标准等大多建立在对象技术之上。Java语言的出现，更进一步推动了分布对象计算，包括著名的Web Services服务协议和统一建模语言（Rational Unified Language，UML）及统一过程（Rational Unified Process，RUP）等。

为了更好地进行软件本质的研究和开发更为复杂的软件系统，研究人员在20世纪90年代初期提出了新的理论方法：软件体系结构（Architecture）。该方法试图以类似建筑学的观点去构造软件，并以更精确的方式刻画软件的结构。软件体系结构作为一门新的学科，其研究范围包括软件系统整体的组织和控制结构、通信、同步和数据存取的协议、系统单元的功能分配、物理分布、系统单元的集成、系统性能及在设计方案间进行选择等问题。软件体系结构是成功解决软件复杂度问题和质量问题的一把金钥匙。作为软件系统的自然属性，体系结构存在于每一个软件系统之中。软件体系结构奠基于程序算法、数据结构和对象设计，它是系统实施的抽象描述。体系结构概括了整体系统结构、功能部件分解、部件的本质和特性、部件的界面和部件之间的通信协议，以及整体性布局策略及法则。软件体系结构图帮助

各种不同背景的人，包括技术人员、用户或客户、项目管理人员、投资者等，共同就系统的总体性质和要求进行讨论分析，从而对将要诞生的系统取得共识。

本书包括12章，第1章是基本概念，包括软件及其复杂性、软件工程知识体系、软件过程成熟度模型、面向对象技术、软件体系结构和高质量软件生产知识集成支持结构。后面10章是按照IEEE提出的软件工程知识体系（SWEBOK，2004版）设置，包括第2章软件质量、第3章软件工程过程、第4章软件需求、第5章软件设计、第6章软件构造、第7章软件测试、第8章软件维护、第9章软件配置管理、第10章软件工程管理和第11章软件工程工具和方法。安排次序考虑到学习的方便。第12章案例研究，包括基于Web软件体系结构的DSS开发、NASA软件开发方法、印度Infosys公司软件过程管理、微软同步稳定法。书后有4个附录：UML中主要的图、简明SW-CMM、简明Fagan审查法和层次分析算法。

对于本科生，建议完成每章的复习题和思考题（选自同等学历硕士软件工程复习与考试指南和中国系统分析员试题），对于研究生和软件技术人员，建议还要学习参考文献，中国系统分析员论文试题及软件工程国际会议（International Conference of Software Engineering，ICSE，http://portal.acm.org/）的相关论文。之所以这样安排，是因为在中国目前情况下，各种类型的考试还是无法回避的，而且还是社会公平的重要保证。同时举行了20年的中国软件水平考试（系统分析员15年）在一定程度上反映了中国软件技术的现实需求和发展趋势，而已经召开的28届ICSE会议也在一定程度上反映了世界软件技术的现实需求和发展趋势，这样安排希望能够达到立足国内、放眼世界的目的。

本书参考了许多国内外的软件工程资料，尤其是Alan M.Davis先生的《201 Principles of Software Development》、Sami Zahran先生《Software Process Improvement—Practice Guidelines for Business Success》及其中译本和刘润东先生的《UML对象设计与编程》。同时我也要感谢李师贤教授、陈火炎教授、郭荷清教授，以及顾景华先生、候亚飞先生、乐建兵先生、刘利人先生、张积友先生、陈泽宁先生和王永松先生等对我的帮助。同时也对我的研究生卓永乐、黄德毅、安诗芳和王云凤等所做的大量辛勤工作表示感谢。

谨以本书怀念2006年2月19日逝世的恩师杨润生教授，他教会了我做研究要脚踏实地。

由于水平有限，不当之处，敬请指正。E-mail:csjpwan@scut.edu.cn。

万江平

2006年7月

广州华南理工大学

目　录

第 1 章　软件工程基本概念

内容提要：

- ☑ 软件的概念、特点及分类
- ☑ 软件的发展和软件危机的原因
- ☑ 软件工程及其基本目标和原则
- ☑ 软件工程知识体系指南的目的、内容及其相关学科
- ☑ 软件过程和软件过程改进
- ☑ CMM 模型的概念、特点及分类
- ☑ 面向对象技术的概念和特点
- ☑ 软件体系结构的概念、特点及分类
- ☑ 高质量软件生产知识集成支持结构

1.1　软件及其复杂性

软件危机的根本原因是软件系统高度复杂，难以驾驭。早期软件开发所带有的个人色彩。要解决软件危机，必须应用严格的技术，使用帮助管理复杂性的工具。

1.1.1　软件的特点和软件危机

软件（Software）是计算机系统中与硬件相互依存的另一部分，它包括程序、数据及相关文档的完整集合。

1. 软件的特点

软件具有以下特点。

(1) 软件是一种逻辑实体，而不是具体的物理实体，因而具有抽象性。这个特点使软件和计算机硬件，或是其他工程对象有着明显的差别。

(2) 软件的开发与硬件不同。在软件的开发过程中没有明显的制造过程。也不像硬件那样，一旦研制成功，可以重复制造，在制造过程中进行质量控制，以保证产品的质量。软件是通过人们的智力活动，把知识与技术转化成信息的一种产品。一旦某一软件项目研制成功，以后就可以大量复制同一内容的副本。所以，对软件的质量控制，必须着重在软件开发方面下功夫。

(3) 软件在运行和使用过程期间，没有硬件那样的机械磨损、老化问题。

(4) 软件的开发和运行常常受到计算机系统的限制，对计算机系统有着不同程度的依赖性。软件不能完全摆脱硬件单独活动，在开发和运行中必须以硬件提供的条件为依据。

(5) 软件的开发至今尚未完全摆脱手工艺的开发方式。软件产品大多数是“定做”的，

很少能做到利用现成的部件组装成所需的软件。对于软件开发人员来说，开发工作是一种高强度的脑力劳动，没有人认为这是一项轻松的工作。

(6) 软件是复杂的。软件的复杂性可能来自其所反映的实际问题的复杂性，例如，软件所反映的自然规律是人类社会的事物，都具有一定的复杂性。另外，软件的复杂性也可能来自程序逻辑结构的复杂性，例如，一个系统软件要能处理各种可能出现的情况。软件开发特别是应用软件的开发，常常涉及其他领域的专门知识，这对软件人员提出了很高的要求。

(7) 软件成本相当昂贵。软件的研制工作须投入大量的、复杂的、高强度的脑力劳动，它的成本是比较高的。

(8) 相当多的软件工作涉及社会因素。类似于企业管理类型的软件自然是不言而喻的。许多软件的开发和运行涉及机构、体制及管理方式等问题，甚至涉及人的观念和心理。对于这些人的因素重视得不够，常常是软件工作遇到的问题之一。

2. 软件的分类

软件按其功能可分为以下类型。

(1) 系统软件：是能与计算机硬件紧密配合在一起，使计算机系统各个部件、相关软件和数据协调、高效地工作的软件。

(2) 支撑软件：是协助用户开发软件的工具性软件，其中包括帮助程序人员开发软件产品的工具，也包括帮助管理人员控制开发进程的工具。

(3) 应用软件：是在特定领域内开发，为特定目的服务的一类软件。现在几乎所有国民经济领域都使用计算机，为这些计算机领域服务的应用软件种类繁多，其中商业数据处理软件是所占比例最大的一类，工程与科学计算软件大多属于数值计算问题。

3. 软件的发展

在计算机系统发展的初期（20 世纪 50—60 年代），硬件经历了不断的变化，而软件则被多数人作为一种事后的工作来看待，对它而言，几乎没有什么系统的方法可以遵循。大多数软件是由使用该软件的人或多个用户而研制的，使软件带有较强的个人色彩：软件设计是在某个人的头脑中完成的一种隐含的过程。而且，往往没有软件说明书。

在计算机系统发展的第 2 个时期（20 世纪 60—70 年代），这个时期以产品化的使用和“软件车间”的出现为特征。多道程序设计、多用户系统引入人机对话的新概念。人们开发软件是为了广泛销售。

在计算机系统发展的第 3 个时期（20 世纪 70 年代以后），分布式系统（多个计算机、各机器并行执行和相互通信）极大增加了以计算机为基础的系统复杂性。用于维护软件的费用占了数据处理预算的 50%以上，而软件开发的生产率又跟不上新系统对软件需求的步伐。为了对付不断增长的软件危机，软件工程开始得到认真对待，如表 1-1 所示。

表 1-1 计算机软件发展的 3 个时期及其特点

特点/时期	程序设计	程序系统	软件工程
软件所指	程序	程序及说明书	程序、文档、数据
主要程序设计	汇编及机器语言	高级语言	软件语言*
软件工作范围	程序编写	包括设计和测试	软件生存期
需求者	程序设计者本人	少数用户	市场用户
开发软件的组织	个人	开发小组	开发小组及大中型软件开发机构

续表

特点/时期	程序设计	程序系统	软件工程
软件规模	小型	中、小型	大、中、小型
决定质量的因素	个人程序技术	小组技术水平	管理水平
开发技术和手段	子程序 程序库	结构化程序设计	数据库、开发工具、开发环境、工程化开发方法、标准和规范、网络及分布式开发、面向对象技术、软件过程与过程改进
维护特征	程序设计者	开发小组	专职维护人员
硬件特征	价格高 存储容量小 工作可靠性	降价、速度、容量及工作可靠性	向超高速、大容量、微型化及网络化方向发展
软件特征	完全不受重视	软件技术的发展不能满足需要，出现软件危机	开发技术有进步，但未获突破性进展，价高，未完全摆脱软件危机

注：软件语言包括需求定义语言、软件功能语言、软件设计语言、程序设计语言等

1968 年，在北大西洋公约组织召开的计算机科学会议上，Fritz Bauer 首先提出了“软件工程”的概念，试图建立并使用正确的工程方法开发出成本低、可靠性好并在机器上能高效运行的软件，从而解决或缓解软件危机如表 1-2 所示。

表 1-2　NATO 关于软件工程问题报告（1968）

(1) 客户和设计者缺少对系统需求的了解
(2) 由于估算技术差、没有为客户需求的变化预留时间，以及没有很好地了解系统就将程序任务分块。从而使得对于开销和时间上的估计往往与实际的花费有巨大的差距
(3) 变数很大。例如，根据一项研究，程序员的生产率水平有可能相差 26 倍
(4) 很难区分设计和开发（编程）工作，有些设计方面的决策仍必须在编程中做出
(5) 由于程序开发通常并不是一系列步骤的简单叠加，每一个步骤都是相互关联的，因此很难监控软件项目的进度
(6) 软件系统规模的快速增长
(7) 同一项目下各工作小组之间缺乏沟通，许多事情没有有效地协调或充斥大量无用的信息，这使得沟通效果更加糟糕，并且处理所需信息的自动化程度不高
(8) 开发在线产品控制工具需要巨大的花费
(9) 很难客观地度量程序员绩效和系统的性能
(10) 软件开发人员之间在系统开发过程中，追求的不是“实用”而是“更新”和“更好”，这使得他们在单个项目中既有研究、又有开发和商品化，从而使得预测和管理变得极其困难
(11) 对程序员需求呈快速增长之势，然而训练有素、经验丰富的程序员数量有限
(12) 难以保证大型软件系统的可靠性（减少错误及提高系统的容错性）
(13) 软件依赖于硬件，这使得很难在不同的计算机之间实现软件的标准化
(14) 缺少可辅助构建新的程序的可重用的软件组件
(15) 软件维护成本往往超过系统最初的开发成本

来源：Compiled from Peter Naur and Brian Ramdell (eds.), Software Engineering: Report on a Conference Sponsored by the NATO Science Committee (Brussels: Scientific Affairs Division, NATO, January 1969), published in Michael A. Cusumano Japan's Software Factories (New York: Oxford University Press) P.67