高等院校计算机专业教育改革推荐教材

软件工程方法与实践

胥光辉　金凤林　丁　力　编著

机械工业出版社
CHINA MACHINE PRESS

高等院校计算机专业教育改革推荐教材

软件工程方法与实践

胥光辉　金凤林　丁　力　编著

机 械 工 业 出 版 社

本书全面系统地介绍了软件工程的概念、原理和典型的技术方法。本书旨在为软件工程领域的理论研究和实践应用架起一座沟通的桥梁，在注重实用的前提下，介绍软件工程领域最新的研究成果和成熟的实践经验。与同类教材的不同点包括将面向对象方法和结构化方法有机结合，注重标准化和过程改进，强调项目管理和软件测试等。

本书共11章，分成三部分：第1部分包括第1～3章，介绍软件工程的由来、软件过程模型和CMM。第2部分包括第4～8章，按软件生命周期的顺序介绍需求分析、系统设计、程序开发和软件测试几个阶段。其中第5章讨论了面向对象方法的基本原理。第3部分包括第9～11章，着重讨论软件项目的管理技术，包括软件度量、项目管理和项目管理实例研究。附录中简单介绍了统一建模语言UML。

本书可作为高等院校“软件工程”课程的教材或教学参考书，也可供有一定实际经验的软件工作人员和广大计算机用户阅读参考。

图书在版编目（CIP）数据

软件工程方法与实践/胥光辉等编著. —北京：机械工业出版社，2004.2
（高等院校计算机专业教育改革推荐教材）
ISBN 7-111-13928-3

Ⅰ.软… Ⅱ.胥… 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（2004）第006699号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）
策　　划：胡毓坚
责任编辑：陈振虹
责任印制：闫　焱

保定市印刷厂印刷·新华书店北京发行所发行

2004年2月第1版·第1次印刷
789mm×1092mm 1/16·17印张·418千字
0 001—5 000册
定价：24.00元

高等院校计算机专业教育改革推荐教材

编委会成员名单

主　任　刘大有

副主任　王元元

编　委　（按姓氏笔画排序）

刘晓明　李师贤　张桂芸　徐汀荣

耿亦兵　顾军华　黄国兴　薛永生

编者的话

计算机科学技术日新月异的飞速发展和计算机科学技术专业教育的相对滞后，已是不争的事实。

有两个发人深省的现象：一是，由于非计算机专业的学生既具有一门非计算机专业的专业知识，又具有越来越高的计算机应用技术水平，从而使计算机专业的学生感受到一种强烈的冲击和压力；二是，创建软件学院的工作已有近两年的历史，但软件学院的计算机专业教育的定位仍在探讨之中。

我们认为计算机科学与技术专业（以下简称计算机专业）教育的改革势在必行，正确认识和划分计算机专业教育的层次，对该专业的教育改革无疑是一个非常重要的问题。我国的计算机专业教育主要分三个层次。一般说来，这三个层次通常分布在以下三类高等院校：

第一层次主要以具有计算机一级学科博士学位授予权的教育部属重点高等院校为代表（包括具有两个博士点的大学）。这一类大学本科着重培养理论基础比较坚实、技术掌握熟练、有一定研究和开发能力的计算机专业学科型人才，其中部分学生（约本科生的 10 %）可攻读博士学位。

第二层次主要以具有一个计算机二级学科专业博士点的教育部属高等院校为代表。这一类高等院校本科着重培养有一定的理论基础、技术掌握比较熟练、有一定的研究或开发能力的计算机专业人才，其中一部分培养成学科型人才，另一部分培养成应用型人才，一小部分学生（约本科生的 5 %）可攻读博士学位。

第三层次主要以具有计算机二级学科专业硕士点的省属高等院校为代表。这一类高等院校本科面向企业应用，侧重培养对计算机技术或部分计算机技术掌握比较熟练，有一定的开发、应用能力的计算机专业应用型人才，其中很小一部分学生（约本科生的 2.5 %）可攻读博士学位。

国家教育部、计委批准的或省教育厅批准的示范性软件学院，就其培养目标和办学特色而言，分别与第二层次中应用型人才培养部分以及第三层次比较相近，但在如下方面有所不同：将软件工程课程作为专业教学重点；更加强调英语教学，更加重视实践能力培养，并对两者有更高的要求。

我们本着对高等院校的计算机专业状况的认识，主要面向与上述第二、第三两个层次对应的院校及与之相近的软件学院，总结多年的计算机专业的教改经验，在一定程度上溶入了 ACM & IEEE CC2001 和 CCC2002（中国计算机科学与技术学科教程）的教改思路，组织我国一直投身于计算机教学和科研的教师，编写了这套“高等院校计算机专业教育改革推荐教材”（以下简称“推荐教材”）。自然，“推荐教材”中所贯穿的改革思路和做法，也是针对上述第二、第三两个层次对应院校的计算机专业学生。这些思路和做法可概括成以下三句话：

- 适度调整电子技术基础、计算机理论基础和系统软件的教学内容。
- 全面强化计算机工具软件、应用软件的教学要求。
- 以应用为目标大力展开软件工程的教学与实践。

电子技术基础、计算机理论基础、系统软件教学关系到学生的基本素质、发展潜力和日后

的应变能力。“推荐教材”在调整它们的教学内容时的做法是:适度压缩电子线路、数字电路和信号系统的教学内容,变三门课程为两门,并插入数字信号处理的基础内容;合并“计算机组成原理”、“微型计算机接口技术”和“汇编语言”为“计算机硬件技术基础”一门课程;注意适当放宽“离散数学”课程的知识面,使之与 CCC2002 的要求基本接轨,但适度降低其深度要求;更新系统软件课程的教学内容,以开放代码的 Linux 作为操作系统原理的讲授载体,更加关注系统软件的实践性和实用性。

为了提高计算机专业人才的计算机应用能力,全面强化计算机工具软件、实用软件的教学要求是十分重要的,这也是上述改革思路的核心。为此,“系列教材”的做法是:强化程序设计技术,强化人机接口技术,强化网络应用技术。

为强化程序设计技术,“推荐教材”支持在单片机环境、微机平台、网络平台的编程训练;支持运用程序设计语言、程序设计工具以及分布式对象技术的编程训练。大大加强面向对象程序设计课程的组合(设计了三门课程:面向对象的程序设计语言 C++,面向对象的程序设计语言 JAVA 和分布式对象技术),方便教师和读者的选择。

为强化人机接口技术,“推荐教材”设计了“人机交互教程”,“计算机图形学”和“多媒体应用技术”等可供选择的、有层次特色的课程组合。

为强化网络应用技术,“推荐教材”设计了“计算机网络技术”,“计算机网络程序设计”,“计算机网络实验教程”和“因特网技术及其应用”等可供选择的、新颖丰富的课程组合。

将软件工程课程作为专业教学重点,以应用为目标大力展开软件工程的教学与实践,是“推荐教材”改革思路的又一亮点。为改变以往软件工程课程纸上谈兵的老毛病,“推荐教材”从工程应用出发,理论联系实际,突出建模语言及其实现工具的运用,设计了“软件工程的方法与实践”,“统一建模语言 UML 导论”和“ROSE 对象建模方法与技术”等可供选择的、创新独特的软件工程课程组合。对于各类软件学院,“推荐教材”的这一特色无疑是很有吸引力的。

强调实践也是计算机学科永恒的主题,对计算机应用专业的学生来说更是如此。重应用和重实践是“推荐教材”的一个整体特点。这一特点,一方面有利于解决本文开始所指出的计算机专业学生较之非计算机专业学生,在应用开发工作中上手慢的问题;另一方面,使计算机专业的学生能在更大范围内、更高层面上掌握计算机应用技术。这一特点正是许多高等院校计算机专业教育改革追求的一个目标,也是国家教育部倡导软件学院的初衷之一。

“推荐教材”由基础知识、程序设计、应用技术、软件工程和实践环节等五个模块组成。各模块有其对应的培养目标与功能,从而构架出一个创新的、完整的计算机应用专业的课程体系。模块化的设计,使各学校可根据学生及学校的特点做自由的选择和组合,既能达到本专业的总体要求,又能体现具有特色的个性发展。整套教材的改革脉络清晰,结构特色鲜明,值得各高等院校在改革教学内容、编制教学计划、挑选教材书目时借鉴和参考。当然,很多书目也适合很多相关学科的计算机课程用作教材。

“推荐教材”的组成模块和书目详见封底。显然它不能说是完备的(实践环节模块更是如此),其改革的思路、改革的举措也可能有值得探讨的地方。我们衷心希望得到计算机教育界同仁和广大读者的批评指正。

高等院校计算机专业教育改革推荐教材

编委会

前　言

软件工程是计算机学科的主干课程之一，主要研究如何应用科学理论和工程技术来指导大型软件系统的开发，使其发展成一门严格的工程科学，软件工程的最终目的是以较低的成本研制具有较高质量的软件。

计算机软件开发和软件工程领域的发展一直比较活跃，新的技术和方法层出不穷。从传统的结构化方法到目前的面向对象方法，对象建模技术已经从百家争鸣走向了统一的标准化道路，统一建模语言 UML(Unified Modeling Language)已经成为产业界的标准。软件过程对于软件开发和软件质量的重要性得到日益重视，能力成熟度模型 CMM(Capability Maturity Model)对于过程改进活动的有效性已经被大量实践所证实。项目管理是一种广泛应用于各行各业的技术管理过程，对项目实施有效的管理已经成为软件项目成败的关键。

目前高等院校使用的软件工程教材普遍存在知识老化与软件开发实践严重脱节等问题。大多数教材主要讲授传统的结构化开发方法，对业界流行的面向对象方法没有涉及或内容较少。本书旨在为软件工程领域的理论研究和实践应用架起一座沟通的桥梁，在注重实用的前提下，介绍软件工程领域最新的研究成果和成熟的实践经验。

与同类教材的不同点主要包括：

- 面向对象和结构化有机结合：在软件开发的各个阶段介绍面向对象和结构化方法的基本原理和思想，使两者相互融合。
- 注重标准化和过程改进：介绍 CMM 的结构和关键概念，使学生尽早树立软件过程改进的理念。介绍产业界广泛应用的 UML 建模技术。
- 软件度量与项目管理：从应用的角度介绍了软件度量指标体系及其操作实践。对于如何制定项目计划、成本估算、风险管理和跟踪项目进展等项目管理中的关键技术进行了较深入的探讨，并给出了一个项目管理的实例研究。
- 强调软件测试：比较详细地介绍工程实践中非常重视的软件测试的计划、组织、方法和评估。

本书的编者都是工作在教学和科研第一线的青年学者，对软件工程有着浓厚的兴趣，在软件工程的教学、科研和实践方面均有所研究。本书的第 1 章、第 6～11 章由胥光辉编写，第 2、3 章和附录由金凤林编写，第 4、5 章由丁力编写，全书由胥光辉统稿。

在本书编写过程中，一直得到王元元教授的关注和支持，他对本书内容的编选提出了宝贵的意见。张涛副教授提供了部分资料，在此一并致谢。

由于编者水平有限，书中难免存在不足之处，敬请广大读者批评指正。

编　者

目　录

第1章　软件工程的由来

1.1　计算机软件的发展

Microsoft 的比尔·盖茨在一次展览会的演讲中谈到：假如 GM(美国通用汽车公司，世界最大的汽车制造商)的技术能像计算机技术那样发展，我们现在应该能用 25 美元买到一辆 1 加仑汽油跑 1000 英里的汽车。

针对比尔·盖茨的讲话，GM 反唇相讥：如果 GM 发展的技术像 Microsoft 那样，我们现在开的汽车会有以下特点：

① 你的汽车可能毫无道理地每天抛两次锚。
② 每次公路上重新画线时，你都得买辆新车。
③ 有时候你的车在高速公路上莫名其妙地熄火，你必须 accept，然后 restart。
④ 有时候你的车在左拐弯时突然 shutdown 了，无法 restart，你必须重新安装(reinstall)发动机。
⑤ 当你买了“轿车 95”或“轿车 NT”后，每次车上只能坐一个人，你要给其他的人再买椅子。
⑥ Macintosh 牌汽车更可靠、更便宜、更快、也更容易开，但是只能在 5% 的道路上行驶。
⑦ 油量、水温和其他警告灯将由一个“GENERAL CAR FAILURE”的警告灯所代替。
⑧ 新座椅要求大家屁股的尺寸相同。
⑨ 气囊系统弹出前将询问“Are you sure?”，要求你加以确认。
⑩ 有时候你的汽车会锁死车门使你无法进入汽车，你得不停地提门把手、拿钥匙捅、晃天线，直到打开车门。

——摘自《扬子晚报》2002 年 8 月 23 日的 B14 版

我们暂且不考虑 GM 的言词是否过于偏激，但 Microsoft 公司作为全球最大的计算机软件生产商，拥有一大批优秀的软件工程师，具有丰富的软件开发和项目管理经验，开发出来的软件仍然不能很好地保证质量是不争的事实。我们几乎每个人在使用 MS Windows 系列操作系统时，都碰到过死机的现象。因此，在本书的开始，有必要深入研究下面几个问题。

- 与硬件相比，软件的本质特征是什么？
- 为什么硬件的可靠性可以达到很高的程度，而软件却不能做到 bug-free？
- 什么样的软件可以称作是一个好软件，或者说好软件具有什么样的特点？
- 目前存在哪些方法和技术能够有效地提高软件的质量？

1.1.1　什么是软件

1. 软件的定义

一般认为：软件是由能够完成预定功能和性能的一组计算机指令(计算机程序)；程序在执行过程中需要输入、处理和输出的数据结构；描述程序的设计和使用的文档三部分组成。

计算机是由硬件和软件两部分组成的，只有硬件、没有软件的计算机称为“裸机”。“裸机”

只能识别由“0”和“1”组成的机器指令，使用起来非常不方便。为了方便用户使用计算机，需要在“裸机”上覆盖软件。覆盖的第一层软件称为操作系统，它是管理计算机软硬件资源、方便用户使用的一个软件集合。在操作系统之上，计算机还提供多种系统软件和应用软件。系统软件与具体应用功能无关，如数据库管理系统、计算机语言编译系统等，使用系统软件可以开发出具有不同功能的应用软件（操作系统是一种特殊的系统软件）。应用软件与某个应用领域紧密相关，专门用于解决某一个或某一类应用问题，如图书管理系统、火车售票系统等。计算机的软硬件的层次关系如图 1-1 所示。

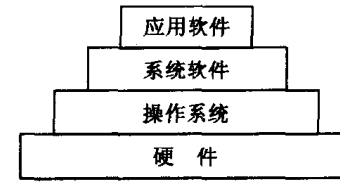

图 1-1　计算机的软硬件结构

2. 软件中的 fault 和 failure

尽管软件在我们的生活中被广泛接受，但生产出来的软件质量仍然有很大的改进余地，绝大多数的软件产品都不是 fault-free 的。很多计算机科学家和软件工程师相信没有办法来编写和测试软件以保证其充分的可靠性。例如，安全临界（safety-critical）系统出现失效的概率必须小于 10^{-9}，而目前生产的软件的可靠性很难达到这一标准。

软件中的缺陷（fault）常被称为 bug。依赖于上下文，bug 具有很多不同含义，可以指需求分析错误、语法错误或是引起系统崩溃的根源等。当一个人在软件开发的某个阶段犯了一个错误，就会在软件中引入一个 bug 或 fault。fault 是开发人员看到的软件系统的内部错误，而用户从外部观察到的软件行为与软件需求的偏差则称为失效（failure）。并不是每个内部的 fault 都会引起一个外部的 failure，因为一些条件不满足（如程序的某些路径没有被执行），使得有些 fault 可能不会表现出来，这也增加了软件测试的难度。

国内外有很多学者曾经试图采用形式验证、定理证明和程序自动生成等技术来保证软件的正确性，但这些形式化技术只在一些规模很小的程序上获得了成功，而且所需的代价极高。为了验证一个程序的正确性所需开发的代码长度甚至超过了被验证程序的长度，而且验证代码本身的正确性也无法保证。因此，形式化技术目前只用于验证安全临界系统和其他关键模块的正确性。

大量生产出来的软件的质量保障主要通过软件测试、过程改进以及软件复用等新技术来实现，软件测试可以发现软件中存在的 bug，但不能保证软件中没有 bug。

3. 软件的特征

软件是逻辑产品而不是物理产品，因此，与硬件相比，软件具有完全不同的特征。

（1）软件的成本集中在开发上，制造几乎没有成本

与硬件不同，软件的成本主要集中在开发和维护上，软件制造（复制）成本很低，1 张光盘的制作成本只有 1 元钱。软件开发是一项高智商的工作，需要很大的前期投入，但作为其成果的软件复制却非常容易。因此，各国都要加强知识产权的保护，严厉打击软件盗版的行为。

（2）软件不会“磨损”

图 1-2 刻画了随着时间的推移，硬件的故障率变化曲线，俗称“浴缸曲线”。硬件在其生命的初期具有较高的故障率，这些故障是由于设计和制造上的缺陷所引起的。随着这些缺陷不断地被发现和修正，硬件的故障率会逐渐减少，最后稳定在一个比较低的水平。经过一段时间，由于受到灰尘、振动、温度、湿度、摩擦及其他环境因素的影响，硬件的故障率又会逐渐增加，直至报废或被其他硬件所替换。即硬件会磨损。

图 1-3 给出的是理想情况下软件的故障率曲线。在软件开发的早期，由于程序中隐藏着较多的缺陷使得软件具有较高的故障率；随着这些缺陷被发现和更正，程序中的缺陷越来越少，因而软件的故障率维持在一个较低的水平（这里假设理想情况下，改正软件中的缺陷不会引入新的缺陷）。软件故障率曲线的前半段与硬件相同，但是软件的运行不会像硬件一样遭到磨损，因此，软件的故障率可以一直保持在同一个水平上。

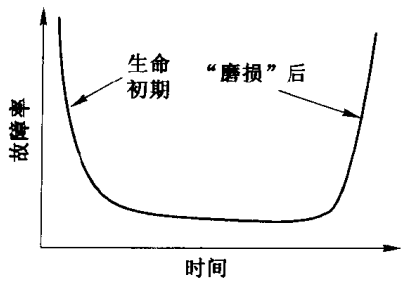

图 1-2 硬件的故障率曲线

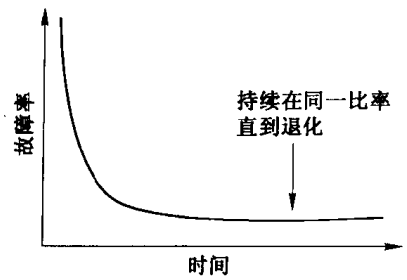

图 1-3 理想软件的故障率曲线

但是，在软件的整个生命周期中，用户的需求可能会发生变化，软件中可能发现了严重的bug，这都迫使软件必须被修改，这些修改（或称维护）常常会一直延续到软件生命周期的结束。每次增加或完善软件功能的维护工作可能会带来新的软件缺陷，使得软件实际的故障率曲线呈现出一种锯齿的形状，如图 1-4 所示。经过多次修改，软件原本良好的结构可能已经面目全非了，故障率曲线也在逐步抬高，最后，该软件被放弃了。

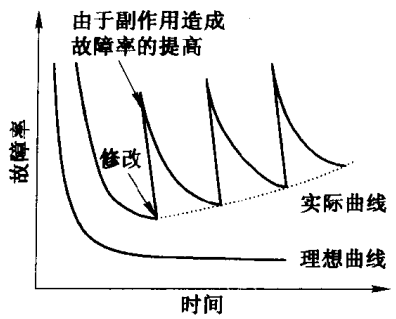

图 1-4 软件实际的故障率曲线

(3) 大多数软件都是从头开发的

硬件工程师设计一个电路板的过程是：首先画一个简单的数字电路图，接着做一些基本分析以保证预定功能的实现，然后在市场上采购所需的集成电路块搭建电路板，编写程序，最后进行调试和测试。每种集成电路块（芯片）都有惟一的编号、固定的功能、定义好的接口和标准的集成指南。因此，硬件的设计和实现相对来说是比较容易的。

尽管软件复用技术、构件技术、软件开发环境提高了软件开发的效率，但大多数软件开发仍然是从零开始的。目前，有些软件厂商提供自己开发的软件构件，但这些构件并没有统一的、标准化的接口，一般是一个厂家一套标准。另外，构件是否经过了充分的测试，其质量是否让人放心也是一个问题。

1.1.2 软件开发的历史

自 1946 年世界上第一台电子数字计算机在美国宾夕法尼亚大学的摩尔学院诞生以来，计算机技术以空前的速度飞速发展，先后经历了电子管时代、晶体管时代、集成电路时代和超大规模集成电路时代。著名的摩尔定律告诉我们计算机的速度平均每 18 个月增加 1 倍，价格降低 1 半，至今仍然适用。和计算机硬件的发展相适应，计算机软件的开发技术也发生了极大变化，可以大致以 10 年左右的时间来划分软件开发技术发展的各个阶段。

1. 20 世纪 40～50 年代

程序员直接使用机器指令来编写程序，机器指令一般由操作码和数据码两部分组成，两部