高等学校计算机专业“十二五”规划教材

软件详细设计教程

樊海玮　吕进　杜瑾
谢小军　赵祥模　编著

西安电子科技大学出版社
http://www.xduph.com

高等学校计算机专业“十一五”规划教材

软件详细设计教程

樊海玮　吕 进　杜 瑾
谢小军　赵祥模　编著

西安电子科技大学出版社

内 容 简 介

本书在软件工程知识体系框架下，围绕着软件形成过程，以软件详细设计这一关键环节为中心，系统讲述了软件详细设计的基本思想、理论、方法、技术，以及软件详细设计技术在软件工程中的应用方法、原则和技术规范。

本书首先从详细设计阶段前的先导过程出发，介绍了包括软件体系结构、统一建模语言、软件需求工程、软件设计工程在内的相关基础性知识；其次重点介绍了软件结构化详细设计和面向对象详细设计这两类主流技术，并与软件实现过程相结合，介绍了软件编码设计与规范，指出了面向对象软件实现的衔接方法；最后介绍了软件测试的方法、过程与技术，强调了软件详细设计与软件测试二者之间的应用关系和协作方法。

本书适合作为高等院校计算机、软件工程、信息工程、通信工程、自动化、电子技术等相关专业的本科及研究生教材，也可作为信息科学、系统工程等领域科研人员的参考书。

★本书配有电子教案，需要者可登录出版社网站免费下载。

图书在版编目(CIP)数据

软件详细设计教程/樊海玮等编著. —西安：西安电子科技大学出版社，2010.12

高等学校计算机专业“十一五”规划教材

ISBN 978-7-5606-2484-6

Ⅰ. ①软… Ⅱ. ①樊… Ⅲ. 软件设计—高等学校—教材 Ⅳ. ①TP311.5

中国版本图书馆 CIP 数据核字(2010)第 192249 号

策　　划　云立实

责任编辑　周　娟　阎　彬

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)88242885　88201467　　邮　　编　710071

网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com

经　　销　新华书店

印刷单位　陕西天意印务有限责任公司

版　　次　2010 年 12 月第 1 版　2010 年 12 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印 张 24.875

字　　数　586 千字

印　　数　1～3000 册

定　　价　35.00 元

ISBN 978 - 7 - 5606 - 2484 - 6/TP • 1236

XDUP 2776001-1

高等学校计算机专业“十一五”规划教材

编审专家委员会

前　言

软件工程专业是计算机领域发展最快的学科分支之一，国家非常重视软件行业的发展和人才培养，国家教育部在软件工程专业课程体系中把“软件详细设计”课程作为软件工程专业人才培养的重要课程。

多年来，在软件工程专业和计算机科学与技术专业所开设的课程中，涉及软件工程知识体系的课程只有软件工程学或软件工程导论，其教学用书或参考资料多以探讨软件工程思想、方法及准则为主。教学中，当需要引导学生从软件的设计阶段向软件实现阶段推进时总觉得比较困难，学生理解起来也感觉特别抽象难懂。我们认为产生上述问题的重要原因之一，就在于这些教材不能从实施层面系统、细致地阐述详细设计的方法、步骤和工具。基于此，我们萌生了编写本教材的强烈意愿，在规划教材结构和内容时，注重在软件工程知识体系框架下，围绕着软件过程，以软件详细设计这一关键环节为中心，系统讲述软件详细设计的基本概念、理论、方法、技术，以及软件详细设计技术在软件工程中的应用方法、原则和技术规范。特别加强了以下三个方面的内容组织：第一，从详细设计阶段前的先导过程出发，介绍包括软件体系结构、统一建模语言、软件需求工程、软件设计工程在内的相关基础性知识；第二，重点介绍软件结构化详细设计和面向对象详细设计这两类主流技术，并与软件实现过程相结合，介绍软件编码设计与规范，指出了面向对象软件实现的衔接方法；第三，介绍软件测试的方法、过程与技术，强调了软件详细设计与软件测试二者之间的应用关系和协作方法。

全书遵循统一软件开发过程，重点采用统一建模语言进行模型描述，并通过丰富的详细设计实例进行诠释。本书具有系统性强、中心明确、理论与实践结合等特点，注重软件详细设计技术与软件详细设计过程的统一，明确了软件开发项目中软件过程与文档过程的关系，强调了文档过程对软件过程的跟踪与评审作用。期望本书能够帮助读者完整、有效地完成软件设计和开发过程。

本书适合作为高等院校计算机、软件工程、信息工程、通信工程、自动化、电子技术等相关专业的本科及研究生教材，也可作为信息科学、系统工程等领域科研人员的参考书。

全书共分 12 章，其中第 1 章、第 4 章、第 12 章由杜瑾完成；第 2 章和第 8 章由谢小军完成；第 3 章、第 5～7 章由吕进完成；第 9～11 章由樊海玮完成；赵祥模教授负责全书的统稿、审校和组织安排。

本书的编写得到了西安电子科技大学出版社的大力支持，我们的同事周琳、闻怡、曹金山等同志还参加了本书部分章节的图表绘制和校稿工作，在此表示衷心的感谢！

在本书编写过程中，我们虽然力求准确、完整和深入，但限于本身学识和水平，书中仍会存在不足之处，殷切希望读者批评指正，以期在下次修订时予以纠正。

编 著 者

2010 年 5 月于西安

目　录

第1章 软件工程概述

软件是人类思维创造的杰作，并成为人类现代生活的催化剂。今天，软件遍布整个世界，在现代通信、商务处理、工业控制、生物工程、军事指挥以及宇宙探索等方面发挥出巨大的威力，推动了商业、科学和工程领域的跨越式发展，对整个社会的经济和文化产生了深远的影响。在计算机诞生的初期，软件仅仅是计算机硬件的附属品，其作用和成本微乎其微。如今，软件以各种形式嵌入在越来越多的产品中，不仅成为影响系统功能和性能的关键因素，而且在整个系统的成本中占据着越来越大的比重。因此，如何设计和开发高质量的软件是人们长期以来一直努力研究的主要问题。

软件工程是为了解决开发成本、效益和软件质量问题而产生的。从1968年NATO(North Atlantic Treaty Organization，北大西洋公约组织)会议首次提出“软件工程”概念至今，虽然人们并没有彻底解决软件危机的问题，但是正是软件工程的发展促使软件取得了如此令人瞩目的成就。四十多年来，人们更好地认识了软件的开发过程，在软件的需求、设计、实现、测试和维护等方面提出了许多有效的方法，新的开发方法和开发工具在大型、复杂软件系统的开发过程中起到了事半功倍的作用。如果没有这些复杂的软件，人们就不可能探索宇宙空间，也不可能拥有网络和现代化的通信技术，更不可能揭开人类基因的奥秘。

当前，软件工程仍然是一个正在迅速兴起的年轻学科，尚未形成完整的理论知识体系，需要大量的理论研究和工程实践。我们相信，随着软件工程学科的日益成熟，它必将对未来的软件开发产生更大的推动力。

1.1 软 件

1.1.1 软件的定义

在软件的发展过程中，软件从手工作坊式的程序演变为工程化的产品，人们对软件的看法也发生了根本性的变化。“软件=程序”显然不能涵盖软件的完整内容，除了程序之外，软件还应包括与之相关的文档和配置数据，用以保证这些程序的正确运行。

《IEEE Standard Glossary of Software Engineering Terminology》给出了有关软件的定义：软件是计算机程序、规程以及运行计算机系统可能需要的相关文档和数据。其中：① 计算机程序是计算机设备可以接受的一系列指令和说明，为计算机执行提供所需的功能和性能；② 数据是事实、概念或指令的结构化表示，能够被计算机设备接受、理解或处理；③ 文档是描述过程、方法及使用的图文材料。

然而，软件的真正含义却不是一个形式的定义所能体现的。从软件的内容来看，软件更像是一种嵌入式的数字化知识，其形成是一个通过交互对话和抽象理解而不断演化的过程。

软件的应用领域十分广泛，呈现形式也是多种多样的，在某种程度上很难对软件的类型给出一个通用的界定。根据软件服务对象的范围不同，一般可以将软件划分为通用软件和定制软件两种类型。

1．通用软件(Generic Software)

通用软件是由软件开发组织开发，面向市场用户公开销售的独立运行系统，像操作系统、数据库系统、字处理软件、绘图软件包和项目管理工具等都属于这种类型。有时，通用软件也被称为套装软件(Packaged Software)

2．定制软件(Customized Software)

定制软件是受某个特定客户委托，由软件开发组织在合同的约束下开发的软件，像企业 ERP 系统、卫星控制系统和空中交通指挥系统等都属于这种类型。

通用软件由开发组织根据市场调研自主提出产品需求，并以此进行设计和开发。其最大的优势在于，可以通过近乎零成本的复制来分摊最初投入的一次性开发成本，最终使原本昂贵的软件产品的价格降至众多用户可接受的程度，从而有效地提高市场份额和利润。但是，为了分摊最初的开发投入，通用软件必须面对足够大的市场空间，其功能设计也只能面向大规模用户普遍存在的共性需求。

对于不同的用户来说，除了共性需求之外还存在着个性化的需求，而这些个性化需求对于很多用户来说，恰恰是应用的关键所在。定制软件完全是订单开发，即按照单个客户的个性化要求，以软件项目的方式为其提交个性化的解决方案，从而更好地满足客户的需求。

1.1.2　软件的特性

计算机在使社会生产力得到迅速解放、使人类生活高度自动化和信息化的同时，却没有使计算机本身的软件生产得到类似的巨大进步。软件开发依然面临着过分依赖人工、软件难以重用、大量重复开发和生产率低下等问题，而导致这些问题的关键在于软件本身的特性。

(1) 软件是复杂的。软件是人类思维和智能的一种延伸和在异体上的再现，远比任何以往人类的创造物都复杂得多。在大型软件系统中，无数种数据、状态和逻辑关系的组合以及人类思维的复杂性和不确定性导致的理解歧义和差异，使整个系统的复杂性急剧增加，也使软件的设计、实现和测试都变得相当困难。

在著名的《没有银弹：软件工程中的根本和次要问题》一文中，Fred Brooks 认为正是软件固有的复杂性造成了软件开发的诸多问题。由于复杂性，人们难以全面理解问题，团队成员之间的沟通也变得非常困难，从而导致了产品缺陷、成本超支和进度拖延；由于复杂性，描述和理解软件系统所有可能的状态是极其困难的，影响了产品的可靠性；由于软件结构及其依赖关系的复杂性，软件的任何更改和扩充都有可能带来灾难性的后果，形成所谓的“雪崩效应”。

(2) 软件是不可见的。一般情况下，人们可以通过几何抽象模型准确地描述有形的物体，

例如建筑师可以用平面图描述建筑物的结构，硬件工程师可以用电路图描述计算机的系统结构，甚至化学家也可以用分子模型描述客观事物的微观结构。但是，软件是客观世界空间和计算机空间之间的一种逻辑实体，不具有物理的形体特征。人们一直试图用不同的图形技术来描述软件结构，即便是现在流行的面向对象技术，也仍然无法给出其准确、完整的描述。

由于软件的不可见性，定义“需要做什么”成为软件开发的根本问题。过去，几乎所有的技术、工具和过程都致力于解决“怎么做”这个次要问题，并且取得了很大的进展，但是在“做什么”的问题上，几十年来情况似乎并没有太大的改善。因此，我们需要知道描述哪些部分、忽略哪些部分，这才是软件的本质问题。

(3) 软件是不断变化的。软件是纯粹思维活动的产物，它不会像硬件一样发生磨损，而是需要随着应用、硬件、用户和社会等各种因素的变化不断地被修改和扩展。由于软件是人类思维和智能的一种延伸，因此当软件被真正应用之后，人们往往希望超越原有的应用边界进行软件功能的提升或扩展；另外，由于软件必须依附于硬件平台，因此需要随着硬件设备的更新和接口的不同而变化。

人们总是认为软件是很容易被修改的，通常忽视了修改带来的副作用，即引入新的错误，造成故障率的升高。图 1.1 显示了软件修改对其质量带来的冲击，不断的修改最终将导致软件的退化，从而结束其生命周期。

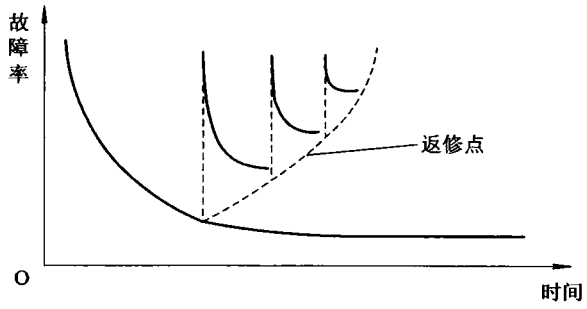

图 1.1　软件的故障率曲线

成功的软件都会发生演化，没有任何变化的软件一定是没有用的。虽然软件的易变性给软件开发带来了许多难题，但是这种固有的特性也给软件带来了异常的生命力。因此，不能回避软件演化的问题，需要采用积极的态度和有效的方法将变更在可控的情况下予以实施。

(4) 大多数软件仍然是定制的，而不是通过已有构件组装而成的。在软件的发展历程中，曾经涌现出许多开发技术和开发工具，当前流行的面向对象开发技术也日趋成熟，但是手工作坊式的软件开发方式仍占主导地位。随着数字化、网络化、智能化成为信息产业的发展趋势，软件复用和软件构件技术受到了广泛的关注，并成为一种社会化的开发方法，有助于软件工程化、工厂化生产的实现。

1.1.3　软件的发展

伴随着第一台计算机的问世，计算机程序就出现了。在以后几十年的发展过程中，人

们逐步认识了软件的本质特性，发明了许多有意义的开发技术与开发工具，同时软件的规模也在不断扩大，其应用几乎渗透到各个领域。纵观整个软件的发展过程，大致可以将其分成以下 4 个重要的阶段：

(1) 第一阶段：20 世纪 50～60 年代。在计算机发展的早期阶段，计算机的主要应用是快速计算，出现了以 Algol、Fortune 等编程语言为标志的算法技术。在这一时期，程序设计被认为是一种任人发挥创造才能的活动，不存在什么系统化的方法和开发管理，程序的质量完全依赖于程序员个人的技巧。随着软件规模的扩大，20 世纪 60 年代末期出现了“软件危机”。

(2) 第二阶段：20 世纪 70 年代。计算机应用开始涉及各种以非数值计算为特征的商业事务领域，交互技术、多用户操作系统、数据库系统等随之发展起来，出现了以 Pascal、Cobol 等编程语言和关系数据库管理系统为标志的结构化软件技术。在这一时期，软件的概念不再仅仅是程序，还包括开发、使用、维护程序需求的所有文档，软件的工作范围从只考虑程序的编写扩展到从定义、编码、测试到使用、维护等整个软件生命周期，瀑布模型被普遍使用。

(3) 第三阶段：20 世纪 80 年代。微处理器的出现与应用使计算机真正成为大众化的东西，而软件系统的规模、复杂性以及在关键领域的广泛应用，促进了软件开发过程的管理及工程化开发。在这一时期，软件工程开发环境 CASE 及其相应的集成工具大量涌现，软件开发技术中的度量问题受到重视，出现了著名的软件工作量估计 COCOMO 模型、软件过程改进模型 CMM 等。20 世纪 80 年代后期，以 Smalltalk、C++等为代表的面向对象技术重新崛起，传统的结构化技术受到了严峻的考验。

(4) 第四阶段：20 世纪 90 年代至今。Internet 技术的迅速发展使软件系统从封闭走向开放，Web 应用成为人们在 Internet 上最主要的应用模式，异构环境下分布式软件的开发成为一种主流需求，软件复用和构件技术成为技术热点，出现了以 Sun 公司的 EJB/J2EE、Microsoft 的 COM+和 OMG 的 CORBA/OMA 为代表的 3 个分支。与此同时，需求工程、软件过程、软件体系结构等方面的研究也取得了有影响的成果。

进入 21 世纪，Internet 正在向智能网络时代发展，以网格计算(Grid Computing)和网络服务(Web Services)为代表的分布式计算日趋成熟，从而实现了信息充分共享和服务无处不在的应用环境；高信度计算(Trustworthy Computing)普遍引起了人们的高度重视，从而为人们创造了一个安全、可靠、值得信赖的环境。

1.2 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遇到的一系列严重问题。软件危机在 20 世纪 60 年代末全面爆发，至今四十多年过去了，虽然软件开发的新工具和新方法层出不穷，但是软件危机依然没有消除。

(1) 软件开发的成本和进度难以准确估计，延迟交付甚至取消项目的现象屡见不鲜。1995 年，美国 Standish 咨询集团公布了题为“混沌”的研究报告，如图 1.2 所示的研究数据表明：在 20 世纪 90 年代初期，软件项目的平均成功率只有 16.2%，在这里，成功的含义

是指在计划的时间和预算内实现项目目标。仅 1995 年这一年间，有 30%的项目在完工之前就被取消了，其余 53.8%的项目由于各种原因在开发过程中遇到了这样或那样的问题，在开发成本、交付时间、产品功能或性能等方面没有实现预期的目标。近几年，有关统计资料显示：软件项目的平均成功率上升到 26%，但仍有 46%的项目超出预算和最后期限，另有 28%的项目没有完成。

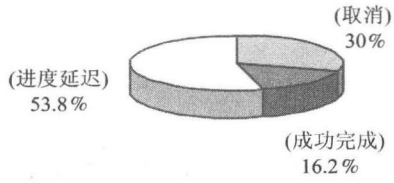

图 1.2　20 世纪 90 年代初期软件项目的成功率

(2) 软件存在着错误多、性能低、不可靠、不安全等质量问题。投入极大努力开发出来的软件常常出现人们无法预料的错误，甚至造成严重的后果。1996 年 6 月，Ariane 5 火箭在发射 37 秒之后突然发生爆炸，其错误原因是浮点数转换成整数时发生溢出，系统对此也没有提供相关的异常处理程序。就这样，一个简单的疏漏造成了这枚耗资 70 亿美元的火箭发射失败。另一个例子是在 1991 年的海湾战争期间，美国对抗伊拉克的飞毛腿导弹曾发生过几次对抗失利，其中一枚导弹在沙特阿拉伯的多哈误击了 28 名美国士兵，而问题的症结在于导弹的软件存在一个累加计时的故障。一个很小的系统时钟错误积累起来就可能产生 14 个小时的误差，从而使跟踪系统失去准确度。今天，随着计算机网络应用的不断普及，计算机病毒的传播、拒绝服务的攻击、网络信息的安全等问题日益突出，软件的质量保证成为人们关注的焦点。

(3) 软件成本在计算机系统的整个成本中所占比例越来越大。在过去的四十多年里，硬件性能至少跨越了 8 个重要的阶段，其成本也随着硬件技术的飞速发展而大幅度地下降。但令人遗憾的是，软件开发的能力却未能与硬件的发展保持同步。伴随着软件规模和复杂性的不断增长，软件成本在整个系统成本中所占的比例也持续上升，图 1.3 显示了软件成本的上升趋势。

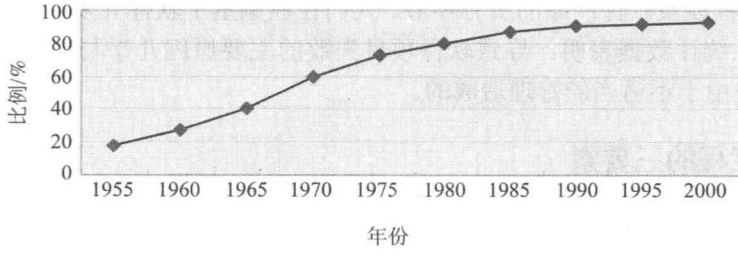

图 1.3　软件成本在系统总成本中所占比例

(4) 软件维护极其困难，而且很难适应不断变化的用户需求和使用环境。在软件交付使用的初期，需要识别和纠正软件的错误，改正软件性能上的缺陷，避免实施中的错误使用。即使软件进入了正常的使用期，由于计算机新技术的出现和用户新需求的提出，也需要修改和改进软件。然而，软件维护依然是一件非常困难的工作，常常出现诸如错误难以修改或者修改又带来新的错误等现象，长期不断的修改也引起了软件的退化。

另外，由于软件开发本身的缺陷和软件维护技术的不成熟，软件维护需要消耗大量的工作量，从而造成维护成本相当昂贵。统计数据表明，软件维护费用占软件整个生存周期总费用的 55%～70%。

1.3 软 件 工 程

软件工程采用工程的概念、原理、技术和方法来开发与维护软件，将经过时间考验而证明正确的管理技术与当前能够得到的最好的技术方法结合起来，其目的在于提高软件的质量与生产率，最终实现软件的工业化生产。

1.3.1 软件工程的概念

1968 年 10 月，NATO 科学委员会在德国的加尔密斯(Garmisch，Germany)开会讨论软件可靠性与软件危机的问题，Fritz Bauer 首次提出了“软件工程”的概念，他认为：软件工程是为了经济地获得能够在实际机器上高效运行的可靠软件而建立和使用的一系列好的工程化原则。

后来，人们曾经多次给出了有关软件工程的定义。这里引用《IEEE Standard Glossary of Software Engineering Terminology》给出的一个更为全面的定义：

① 软件工程是将系统性的、规范化的、可定量的方法应用于软件的开发、运行和维护，即将工程化应用到软件上；② 对①中所述方法的研究。

从上述定义中可以看出，软件工程包括以下两方面的内容：

(1) 软件工程是工程概念在软件领域里的一个特定应用。与其他工程一样，软件工程是在环境不确定和资源受约束的条件下，采用系统性的、规范化的、可定量的方法进行有关原则的实施和应用，这些原则一般是以往经验的积累和提炼，经过时间检验并证明是正确的。因此，软件工程师需要选择和应用适当的理论、方法和工具，同时还要不断探索新的理论和方法，解决新的问题。

(2) 软件工程涉及软件产品的所有环节。人们往往偏重于软件开发技术，忽视软件项目管理的重要性。统计数据表明，导致软件项目失败的主要原因几乎与技术和工具没有任何关系，更多的是由于不适当的管理造成的。

1.3.2 软件工程的三要素

软件工程以关注软件质量为目标，由过程、方法和工具三个要素组成，如图 1.4 所示。

软件工程的方法为软件开发提供了“如何做”的技术，通常包括某种语言或图形的模型表示方法、良好的设计实践以及质量保证标准等，其中使用最广泛的两种方法是传统的软件开发方法和当前流行的面向对象方法。

软件工程的过程是管理和控制产品质量的关键，它定义了技术方法的采用、工程产品(包括模型、文档、数据、报告、表格等)的产生、里程碑的建立、质量的保证和变更的管理，从而将人员、技术、组织与管理有机地结合在

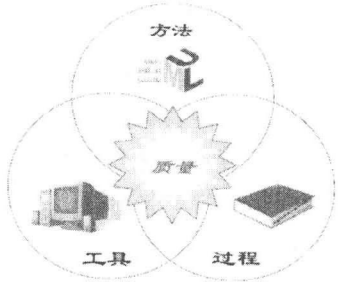

图 1.4　软件工程的三要素