高等学校教材

软件开发技术

——Windows高级编程技术

刘君瑞 姜学锋 主编

西北工业大学出版社

软件开发技术
——Windows 高级编程技术

刘君瑞　姜学锋　主编

西北工业大学出版社

【内容简介】 本书以教育部“卓越工程师教育培养计划”为指导，立足于培养学生的开拓创新能力，重视学生创新意识和应用能力的综合培养。

全书共9章，前4章为基础篇，分别为软件开发方法综述：介绍软件及软件工程的相关理论以及常用的软件开发方法和开发平台；面向对象的程序设计方法：介绍面向对象的程序设计方法和C++面向对象编程语言；数据结构：介绍应用问题求解中常用的线性表、队列、堆栈、树等数据结构；算法设计与分析：介绍算法分析的基本方法及几种常用算法。在这些软件及编程理论指导下，后5章的提高篇就常见的软件开发技术领域进行展开，分别介绍了Windows高级编程中界面的设计方法、数值计算方法、网络编程技术、数据库编程技术以及多媒体编程技术。全书系统全面地描述了软件开发过程用到的各种技术知识。

本书可作为高等学校相关专业的计算机技术基础课程教材，同时也可作为行业技术人员的参考用书。

图书在版编目(CIP)数据

软件开发技术：Windows高级编程技术/刘君瑞，姜学锋主编．—西安：西北工业大学出版社，2013.3

ISBN 978-7-5612-3638-3

Ⅰ.①软… Ⅱ.①刘…②姜… Ⅲ.①Windows操作系统—程序设计 Ⅳ.①TP316.7

中国版本图书馆CIP数据核字(2013)第062178号

出版发行：西北工业大学出版社

通信地址：西安市友谊西路127号 邮编：710072

电　　话：(029)88493844　88491757

网　　址：www.nwpup.com

印 刷 者：兴平市博闻印务有限公司

开　　本：727 mm×960 mm　1/16

印　　张：22.875

字　　数：398千字

版　　次：2013年3月第1版　2013年3月第1次印刷

定　　价：45.00元

前　言

计算机类课程具有实践性强的特点，强化应用性教育是计算机系列课程教学改革的必然趋势。在计算机技术飞速发展的今天，我们不断对教材内容进行修订，力求使教材体系结构保持先进性、系统性、稳定性、发展性以及实用性的特点，以满足创新性人才培养的需求。

本书编写的指导思想是以培养读者具有初步的应用系统软件应用开发能力为目标，以应用开发技术理论为主线，探索以学为主的教学模式，课堂教学覆盖不到之处让学生自学，重视创新能力和综合应用能力的培养。

培养工科大学生的综合应用能力应该是一个连贯的过程，大学本科生在学完“大学计算机基础”和“程序设计”课程后，实践环节掌握了计算机操作的基本技能或编程的基本方法，但是不具备开发一个软件应用系统的知识和能力。因此提高本科生整体计算机应用能力，培养工程创新的思维，是本书的建设目标。

根据目前软件技术的发展，先进的可视化高级语言开发平台，为大规模开展软件技术课程设计提供了良好的技术基础，并且考虑到读者的先修课程为“程序设计”，因此，本书内容以 C/C++作为语言基础，以 Visual C++作为开发平台，涵盖了软件工程、Windows 高级编程、嵌入式开发等方面的知识点和相关技术基础，使读者通过本课程的学习，能够根据软件工程的思想，借助于 Visual C++环境进行一些较高级的 Windows 应用软件开发。

本书共 9 章，其中第 1～7 章由刘君瑞执笔，第 8～9 章由姜学锋执笔，全书由刘君瑞统稿。

由于时间仓促，水平有限，书中不妥和疏漏之处在所难免，恳请读者批评指正。

编　者

2012 年 10 月

目　录

第 1 章　软件开发方法综述

随着软件的发展，计算机应用逐步渗透到社会生活的各个角落，使各行各业都发生了巨大的变化。日益广泛的软件应用促使人们对软件的功能和品质等提出了更高的要求，针对软件开发的研究得到了人们的高度重视。本章介绍软件及软件开发的基本概念，包括软件的定义及特征、软件的分类、软件危机、软件工程、软件开发模型及软件开发技术等。

1.1　软件概述

软件的发展历程可从多个方面描述：就体系结构而言，经历了主机结构到文件服务器，从客户端/服务器系统到基于 Internet 的浏览器/服务器结构的体系结构等的发展过程；从编程语言来讲，经历了从机器代码到汇编语言，再到高级程序语言以及人工智能语言的发展过程；就开发工具来看，经历了分离的开发工具（如代码编辑器、中间代码生成器和连接器）到集成的可视化开发环境，从简单的命令行调试工具到方便的多功能调试器等的发展过程。软件的发展速度是十分惊人的。

1.1.1　软件的定义及特征

1. 软件的定义

软件是计算机系统的重要组成部分，它是计算机程序、方法、规则、与程序有关的各种文档以及在计算机上运行所必需的数据的总称。计算机程序是用计算机的语言编写的，用于求解某个特定问题的算法和数据结构的代码序列的集合。软件是信息商品，为了使用户能正确地运行程序并及时排除故障，需要编写相应程序的文档资料，使用户能够理解进而正确维护程序。因此，软件的范畴比程序大得多，它不仅包括程序，还包括使用维护说明、指南以及培训教材等功能和性能的说明性资料。软件不仅有功能和性能要求，还有质量、成本、交货期、使用寿命要求。软件投入的资金主要是人工费用，研制周期的延长会使得成本陡增，会使软件变得毫无竞争力。按照软件行业标准工作并撰写文档是提高软件开发效率、方便软件产品维护的一个有效方法。

2. 软件的特征

软件生产需要密集的资金和人力投入。大型软件开发占用人力较多，研制时间长，开发费用高。认识软件的特征，对于开发软件具有重要的意义。

(1)软件是一种逻辑产品。软件和硬件是截然不同的两种产品和概念。硬件产品是有形的物理部件或设备，而软件是一种逻辑产品，具有无形性，是开发人员脑力劳动的结晶。软件产品是由程序和文档构成的，在设计和生产软件产品的过程中，首先要抽象出问题求解的数学模型或逻辑模型，再把这些模型转化为求解模型，然后根据求解模型写出程序，经过调试和运行程序，最后得到求解的结果。整个开发过程是通过人脑进行的逻辑思维完成的，其无形性的特征给软件的开发和生产过程的管理带来不便，进度难以控制，开发质量难以评价和保证。如果在软件运行中发现错误，很可能是遇到了一个在开发阶段隐藏的，并且在测试阶段没能检测出来的故障。因此，软件维护通常意味着修改原来的设计，这就在客观上决定了软件维护是相当复杂的工作。

(2)软件产品的质量需通过实践来验证。在整个生产过程中，硬件产品从器件的选购到产品的加工过程都有严格的质量控制和检验标准，来保证产品的质量。但是，如果在产品组装后期发现问题，则有可能造成产品的报废。软件产品在设计、编程和实现过程中的各个阶段中，其质量难以保证和检验，只有在实际问题求解过程中被证实是可行的，才能成为产品。而软件产品的质量问题可以通过测试和调试手段修改其错误来解决。因此，软件在质量保证方面比硬件具有更大的灵活性。

(3)软件产品的成本构成具有上升的趋势。在硬件产品生存周期中，其成本构成中有形的物质占了相当大的比例，例如，设计、生产环节占绝大部分，而售后服务只占少部分。在软件产品生存周期中，其成本构成中人力资源占了相当大的比例。软件的设计和生产只占很小比例，而维护却占了很大比例，据统计数据表明，软件维护的费用占软件总费用的 55%～70%。并且，由于实际问题的复杂性造成了实用软件的规模日趋庞大、结构日趋复杂，这使得软件的费用呈现上升趋势。例如，庞大系统中各个模块之间逻辑接口的定义、数据结构的描述、所有开发人员的协调和组织等。

(4)软件产品的故障率随着软件维护而下降。在硬件产品的生存周期中，生产定型投入使用后，在一定时间内故障率维持在一个较低的水平。但随着时间的推移，产品存在老化和折旧问题，长时间使用造成的磨损使硬件产品的故障率上升。

在软件产品生存周期中，软件在其生命周期的初始阶段存在较高的故障率，开发过程中的错误被纠正后，其故障率下降到一定的水平并保持相对稳

定，直到该软件被废弃不用。

软件是只有过时而无磨损的商品，用得越多，软件内的错误清除得越干净。所谓过时往往是它所在环境升级，导致配套软件必须做相应的升级，否则不能再用；或者同类软件产品功能已更新换代，使得本产品不具备竞争力。

1.1.2　软件分类

软件的分类方法有 6 种，下面分别介绍。

1. 按软件的功能分类

按软件的功能可以将软件分为系统软件、支撑软件、应用软件。

系统软件：与计算机硬件紧密配合在一起，使计算机系统中的各个部件、相关的软件和数据协调、高效工作的软件。例如，操作系统、数据库管理系统、设备驱动程序以及通信处理程序等都属于系统软件。

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，以及帮助管理人员控制开发进程的工具。

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，其中包括为特定目的进行的数据采集、加工、存储和分析服务的资源管理软件。

2. 按软件的服务对象的范围分类

按软件的服务对象的范围可以将软件分为项目软件、产品软件。

项目软件：也称定制软件，是受某个特定客户（或少数客户）委托，由一个或多个软件开发机构在合同的约束下开发出来的软件。例如，军用防空指挥系统、卫星控制系统等都属于项目软件。

产品软件：是由软件开发机构开发出来直接提供给市场，或为成百上千个用户服务的软件。例如，文字处理软件、财务处理软件、人事管理软件等都属于产品软件。

3. 按软件的规模划分

按开发软件所需的人力、时间以及完成的源程序行数可以将软件分为 6 种不同规模的软件，如表 1.1 所示。

任何规模软件的开发工作必须要有软件工程的知识作为指导，遵循一定的开发规范，其基本原则是一样的，只是对软件工程技术依赖的程度不同而已。

4. 按软件的工作方式分类

按软件的工作方式可以将软件分为实时处理软件、分时软件、交互式软件、批处理软件。

实时处理软件：指在事件或数据产生时，立即予以处理，并及时反馈信号，

对需要监测和控制的过程进行控制的软件，主要包括数据采集、分析、输出三部分。

分时软件：允许多个联机用户分时使用计算机的软件。

交互式软件：能实现人机通信的软件。

批处理软件：把一组输入作业或一批数据以成批处理的方式一次运行，按顺序逐个处理完的软件。

表 1.1　软件规模的分类

类　别	参加人员数	研制期限	产品规模(源代码行数)
微　型	1	1～4 周	500 以下
小　型	1	1～6 月	1 000～2 000
中　型	2～5	1～2 年	2 000～50 000
大　型	5～20	2～3 年	5 万～10 万
甚大型	100～1 000	4～5 年	10 万～100 万
极大型	200～5 000	5～10 年	100 万～1 000 万

5. 按使用的频度来划分

一次性软件：有的软件开发出来仅供一次使用，如人口普查、工业普查的软件。

多次用软件：具有较高的使用频度，如天气预报软件。

6. 按软件失效的影响来划分

一般性软件：软件失效的影响不大。

高可靠性软件：软件一旦失效，将出现灾难性后果。

1.1.3　软件危机

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，即随着大容量、高速度计算机的出现，计算机的应用范围迅速扩大，软件开发急剧增长，软件系统的规模越来越大，复杂程度越来越高，软件可靠性问题也越来越突出，原来的个人设计、个人使用的方式不再能满足要求，迫切需要改变软件生产方式，提高软件生产率。

软件危机的具体体现有以下几个方面：

(1)软件开发进度难以预测。拖延工期几个月甚至几年的现象并不罕见，这种现象降低了软件开发组织的信誉。

(2)软件开发成本难以控制。投资一再追加，令人难以置信。结果往往是

实际成本比预算成本高出一个数量级。而为了赶进度和节约成本所采取的一些权宜之计又往往损害了软件产品的质量，从而不可避免地会引起用户的不满。

(3)用户对产品功能难以满足。开发人员和用户之间很难沟通，矛盾很难统一。往往是软件开发人员不能真正了解用户的需求，而用户又不了解计算机求解问题的模式和能力，双方无法用共同熟悉的语言进行交流和描述。在双方互不充分了解的情况下，就仓促上阵设计系统、匆忙着手编写程序，这种“闭门造车”的开发方式必然导致最终的产品不符合用户的实际需要。

(4)软件产品质量无法保证。系统中的错误难以消除。软件是逻辑产品，质量问题很难以统一的标准度量，因而造成质量控制困难。软件产品并不是没有错误，而是盲目检测很难发现错误，而隐藏下来的错误往往是造成重大事故的隐患。

(5)软件产品难以维护。软件产品本质上是开发人员的代码化的逻辑思维活动，他人难以替代。除非是开发者本人，否则很难及时检测、排除系统故障。为使系统适应新的硬件环境，或根据用户的需要在原系统中增加一些新的功能，又有可能增加系统中的错误。

(6)软件缺少适当的文档资料。文档资料是软件必不可少的重要组成部分。实际上，软件的文档资料是开发组织和用户之间的权利和义务的合同书，是系统管理者、总体设计者向开发人员下达的任务书，是系统维护人员的技术指导手册，是用户的操作说明书。缺乏必要的文档资料或者文档资料不合格，将给软件开发和维护带来许多严重的困难和问题。

通过软件危机的表现和软件开发作为逻辑产品的特殊性可以发现软件开发危机的原因：

(1)用户需求不明确。在软件开发过程中，用户需求不明确主要体现在四个方面：①软件开发出来之前，用户自己也不清楚软件开发的具体需求；②用户对软件开发需求的描述不精确，可能有遗漏、有二义性，甚至有错误；③在软件开发过程中，用户还提出修改软件开发功能、界面、支撑环境等方面的要求；④软件开发人员对用户需求的理解与用户本来愿望有差异。

(2)缺乏正确的理论指导。由于软件开发不同于大多数其他工业产品，其开发过程是复杂的逻辑思维过程，其产品极大程度地依赖于开发人员高度的智力投入。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，加剧软件开发产品的个性化，缺乏有力的方法学和工具方面的支持，也是发生软件开发危机的一个重要原因。

(3)软件开发规模越来越大。随着软件开发应用范围的增广，软件开发规

模越来越大。大型软件开发项目需要组织一定的人力共同完成，而多数管理人员缺乏开发大型软件开发系统的经验，而多数软件开发人员又缺乏管理方面的经验。各类人员的信息交流不及时、不准确，有时还会产生误解。软件开发人员不能有效地、独立自主地处理大型软件开发的全部关系和各个分支，因此容易产生疏漏和错误。

(4)软件开发复杂度越来越高。软件开发不仅仅是在规模上快速地发展扩大，而且其复杂性也急剧地增加。软件开发产品的特殊性和人类智力的局限性，导致人们无力处理“复杂问题”。所谓“复杂问题”的概念是相对的，一旦人们采用先进的组织形式、开发方法和工具提高了软件开发效率和能力，新的、更大的、更复杂的问题又摆在人们的面前。

1.2 软件工程

1.2.1 软件工程概述

1968 年北大西洋公约组织的计算机科学家在德国召开国际会议，第一次讨论软件危机问题，并正式提出“软件工程”一词。作为一个新兴的工程学科，软件工程主要研究软件生产的客观规律性，建立与系统化软件生产有关的概念、原则、方法、技术和工具，指导和支持软件系统的生产活动，以期达到降低软件生产成本、改进软件产品质量、提高软件生产率水平，从而克服软件危机的目标。

软件工程学从硬件工程和其他传统工程中吸收了许多成功的经验，明确提出了软件生命周期的模型，发展了许多软件开发与维护阶段适用的技术和方法，并应用于软件工程实践，取得良好的效果。

借用传统工程设计的基本思想，采用工程化的概念、原理、技术和方法来开发与维护软件，突出软件生产的科学方法，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，降低开发成本，缩短研制周期，提高软件的可靠性和生产效率。这就是软件工程。它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，已成为指导计算机软件开发和维护的工程学科。

软件工程包括三个要素：方法、工具和过程。

软件工程方法为软件开发者提供了“如何做”的技术。它包括了多方面的任务，如项目计划与估算、软件系统需求分析、数据结构、系统总体结构的设计、算法的设计、编码、测试以及维护等。

软件工具为软件工程方法提供了自动的或非自动的软件支撑环境。目前已经有许多开发成功的软件工具用来支持上述的软件工程方法。

软件工程的过程是将软件工程的方法和工具综合起来以达到合理、及时地进行计算机软件开发的目的。

1.2.2　软件工程的目标

软件工程的目标是提高软件的质量与生产率，最终实现软件的工业化生产。质量是软件需求方最关心的问题，生产率是软件供应方最关心的问题。质量与生产率之间有着内在的联系，高生产率必须以质量合格为前提。从短期效益看，追求高质量会延长软件开发时间并且增大费用，似乎降低了生产率。从长期效益看，高质量将保证软件开发的全过程更加规范流畅，大大降低了软件的维护代价，实质上是提高了生产率，同时可获得很好的信誉。质量与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。

软件工程的基本目标是：

- 开发成本低
- 功能符合要求
- 性能较好
- 移植性好
- 维护费用低
- 开发周期符合要求

在实际的开发的具体项目中，以上的目标可能互相冲突，这就要力图在以上目标的冲突间取得一定的平衡。

1.2.3　软件工程的原理

1983 年，美国著明的软件工程专家巴利·玻姆(Barry Boehm)综合多位软件工程专家的意见，并总结了美国天合公司(TRW)多年的开发软件的经验，提出了软件工程的七条基本原理。他认为，这七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。它们是相互独立的，是缺一不可的最小集合；同时，它们又是相当完备的。下面简要介绍软件工程的七条原理：

(1)用分阶段的生命周期计划严格管理。统计表明，50％以上的失败项目是由于计划不周而造成的。在软件开发与维护的漫长生命周期中，需要完成许多性质各异的工作。这条原理意味着，应该把软件生命周期分成若干阶段，并相应制订出切实可行的计划，然后严格按照计划对软件的开发和维护进行

管理。玻姆认为，在整个软件生命周期中应指定并严格执行 6 类计划：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、运行维护计划。

(2)坚持进行阶段评审。统计结果显示：大部分错误是在编码之前造成的，大约占 63%。错误发现得越晚，改正它要付出的代价就越大，要差 2 到 3 个数量级。因此，软件的质量保证工作不能等到编码结束之后再进行，应坚持进行严格的阶段评审，以便尽早发现错误。

(3)实行严格的产品控制。开发人员最痛恨的事情之一就是改动需求。但是实践告诉我们，需求的改动往往是不可避免的。这就要求我们要采用科学的产品控制技术来顺应这种要求。也就是要采用变动控制，又叫基准配置管理。当需求变动时，其他各个阶段的文档或代码随之相应变动，以保证软件的一致性。

(4)采纳现代程序设计技术。从 20 世纪六七十年代的结构化软件开发技术，到最近的面向对象技术，从第一、第二代语言，到第四代语言，人们已经充分认识到方法的重要性。采用先进的技术既可以提高软件开发的效率，又可以减少软件维护的成本。

(5)结果应能清楚地审查。软件是一种看不见、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。为更好地进行管理，应根据软件开发的总目标及完成期限，尽量明确地规定开发小组的责任和产品标准，从而使所得到的标准能清楚地审查。

(6)开发小组的人员应少而精。开发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，应该少而精。这一条基于两点原因：高素质开发人员的效率比低素质开发人员的效率要高几倍到几十倍，开发工作中犯的错误也要少得多；当开发小组为 N 人时，可能的通信信道为 $N(N-1)/2$，可见随着人数 N 的增大，通信开销将急剧增大。

(7)承认不断改进软件工程实践的必要性。遵从上述六条基本原理，就能够较好地实现软件的工程化生产。但是，它们只是对现有的经验的总结和归纳，并不能保证赶上技术不断前进发展的步伐。因此，玻姆提出应把承认不断改进软件工程实践的必要性作为软件工程的第七条原理。根据这条原理，不仅要积极采纳新的软件开发技术，还要注意不断总结经验，收集进度和消耗等数据，进行出错类型和问题报告统计。这些数据既可以用来评估新的软件技术的效果，也可以用来指明必须着重注意的问题和应该优先进行研究的工具和技术。