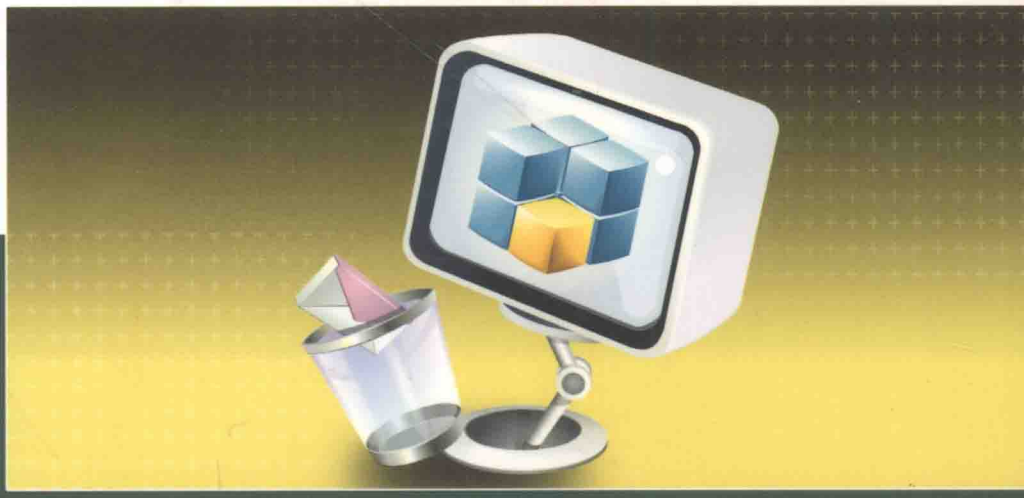
高职高专院校实践类课程系列规划教材

软件工程基础

SOFTWARE ENGINEERING FUNDAMENTALS

周 苏 王 文 编著



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高职高专院校实践类课程系列规划教材

软件工程基础

周 苏　王 文　编著
孙继红　参编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

软件工程是一门理论性和实践性都很强的学科，它采用工程化的概念、理论、技术和方法来指导计算机软件的开发与维护。本书针对计算机和其他IT专业学生的发展需求，系统、全面地介绍了软件工程的概念、原理、方法及应用，详细介绍了软件生存周期、面向对象软件过程和软件过程工程的思想和实现方法，力图反映软件工程领域的最新发展，具有较强的系统性和可读性。

本书的主要特色是：理论联系实际，结合一系列实训项目，把软件工程概念、理论和技术的相关知识融入实践当中，使学生保持浓厚的学习热情，加深对软件工程知识的认识、理解和掌握；按照一系列软件工程国家标准来表达和描述软件工程的知识，使软件工程技术具有很强的可操作性。本书还以实训和引导学生自主学习的方式，安排了丰富和生动的阅读内容，安排了对 Visio、PowerDesigner 和 Visual SourceSafe 等常用软件工具的学习。

本书可作为具有较强实践性的高职高专院校“软件工程”课程的教材，也可供有一定实践经验的软件开发人员、管理人员参考或作为继续教育的教材，还可作为各个级别的计算机软件专业技术资格和水平考试中相关内容的学习辅导用书。

图书在版编目（CIP）数据

软件工程基础 / 周苏，王文编著. —北京：中国铁道出版社，2010.8

（高职高专院校实践类课程系列规划教材）

ISBN 978-7-113-11569-2

Ⅰ. ①软… Ⅱ. ①周… ②王… Ⅲ. ①软件工程－高等学校：技术学校－教材 Ⅳ. ①TP311.5

中国版本图书馆 CIP 数据核字（2010）第 116245 号

书　　名：软件工程基础
作　　者：周　苏　王　文　编著

策划编辑：翟玉峰　王春霞
责任编辑：翟玉峰　　**读者热线电话：400-668-0820**
特邀编辑：李红玉　　**封面制作：白　雪**
封面设计：付　巍
责任印制：李　佳

出版发行：中国铁道出版社（北京市宣武区右安门西街 8 号　　邮政编码：100054）
印　　刷：河北省遵化市胶印厂
版　　次：2010 年 8 月第 1 版　　2010 年 8 月第 1 次印刷
开　　本：787mm×1092mm　1/16　印张：20　字数：487 千
印　　数：3 000 册
书　　号：ISBN 978-7-113-11569-2
定　　价：30.00 元

FOREWORD

前言

软件工程是一门理论性和实践性都很强的学科，它采用工程化的概念、理论、技术和方法来指导计算机软件的开发与维护，它主要研究软件结构、软件设计方法、软件工具、软件工程标准和规范以及软件工程的有关理论。采用工程化的概念、原理、技术和方法来开发与维护软件，把经过时间考验，证明正确的管理技术和当前能够得到的最好的开发方法结合起来，这就是软件工程。

另一方面，高等教育的新形势需要我们积极进行教学改革，研究和探索新的教学方法。在长期的教学实践中，我们体会到，坚持“因材施教”的重要原则，把实验实践环节与理论教学相融合，以实验实践教学促进学科理论知识的学习，是有效地改善教学效果和提高教学水平的重要方法之一。

本书是具有较强实践性的高职高专院校“软件工程”课程的教材。针对计算机和其他 IT 专业学生的发展需求，本书系统、全面地介绍了软件工程的概念、原理、方法及应用，详细介绍了软件生存周期、面向对象软件过程和软件过程工程的思想和实现方法，力图反映软件工程领域的最新发展，具有较强的系统性和可读性。

本书的主要特色是：理论联系实际，结合一系列知识的学习和实验，把软件工程的概念、理论和技术知识融入实践当中，使学生保持浓厚的学习热情，加深对软件工程知识的认识、理解和掌握；按照一系列软件工程国家标准来表达和描述软件工程的知识，使软件工程技术具有很强的可操作性。

在本书的设计编写中，包含以下几个愿望：

- 通过主要基于因特网的实验活动，培养学生在网络环境中自主学习的能力。
- 通过针对 Visio、WinRunner、PowerDesigner、Project 和 Visual SourceSafe 等常用软件工具的学习和实验活动，培养学生的动手能力。
- 通过广泛的专业文章和教材阅读，培养学生探究性学习、理性思考和创新思维的能力。

本书的编撰得到了浙江大学城市学院、浙江商业职业技术学院、温州大学城市学院、广州科贸职业技术学院等多所院校领导和师生的支持，在此一并表示感谢！本书相关的实验素材可以从中国铁道出版社网站（edu.tgbooks.net）的下载区下载。欢迎教师索取为本书教学配套的相关资料和交流。E-mail：zs@mail.hz.zj.cn，QQ：81505050，个人博客：http://blog.sina.com.cn/zhousu58。

编　者

2010 年 6 月

GUIDE

读者指南

本书是为高职高专院校相关专业“软件工程”课程编写的应用型、实践型教材，目的是通过一系列在网络环境下的学习和实训练习，把软件工程的概念、理论知识与技术融入实践当中，从而加深对该课程的认识和理解。

读者对象

高职高专院校相关专业的学生可以把此书作为课程学习的主教材、实训辅助教材或自学读物。教学实践证明，在主要强调实践性、应用性的相关课程中，本书是一本适用和优良的课程主教材。对于已经具备计算机应用基础知识，并希望通过进一步学习得到提高的读者来说，本书也是一本继续教育的良好读物。相信本书将有助于“软件工程”课程的教与学，有助于读者对理解、掌握和应用本课程内容建立起足够的信心和兴趣。

教学与实训内容

本书的教学内容和实训练习包含了软件工程知识的各个方面，包括可供选择的 14 个实训项目、1 个课程实训总结以及附录 A“课程设计”和附录 B“软件文档编写提示（部分）”。每个实训项目中都包含实训目的、所需工具和实训步骤指导等，以帮助读者加深对课程教材中所介绍概念的理解以及掌握主流软件工具的基本使用方法等。

第 1 章：软件工程概述，包括实训项目“软件工程的计算环境”；**第 2 章**：可行性研究与计划，包括实训项目“软件工具与软件开发环境”；**第 3 章**：软件需求分析，包括实训项目“软件开发绘图工具 Visio”；**第 4 章**：软件概要设计，包括实训项目“使用 Vision 绘制工程图形”；**第 5 章**：软件详细设计，包括实训项目“软件工程国家标准”；**第 6 章**：软件编码，包括实训项目“软件测试环境”；**第 7 章**：软件测试，包括实训项目“黑盒法与白盒法设计测试用例”；**第 8 章**：软件维护，包括实训项目“PowerDesigner 入门”；**第 9 章**：软件质量管理，包括实训项目“PowerDesigner 业务处理模型”；**第 10 章**：软件配置管理，包括实训项目“软件配置管理工具 VSS”；**第 11 章**：面向对象分析与设计，包括实训项目“PowerDesigner 概念数据模型”；**第 12 章**：面向对象的实现，包括实训项目“PowerDesigner 物理数据模型”；**第 13 章**：统一建模语言——UML，包括实训项目“PowerDesigner 面向对象模型”；**第 14 章**：软件文件，包括实训项目“软件产品开发文件编制指南”；**第 15 章**：软件工程实训总结。包括对所有实训项目的总结。

实训要求

本书的实训项目共有 14 个，教师可根据不同的教学安排、教学进度和要求等实际情况以及条

件，从中选取部分实训在课堂上完成，部分实训由学生作为作业选择完成。个别实训可能需要占用课下时间才能全部完成。

致教师

现有的“软件工程”教材大都有理论性很强，而实践与应用性偏弱的特点，对教学活动的开展，尤其是给强调教学型、应用型的高职高专院校相关课程教学的开展带来了一定的困难。但是，软件工程活动本身却具有鲜明的应用性，因此，我们应该充分重视这门课程的实训环节，以实训与实践教学来促进理论知识的学习。本书以一系列与网络学习密切相关的实训练习作为主线，来组织对软件工程课程的教学，以求掌握软件工程知识在实践中的应用。

为方便教师对课程实训环节的组织，我们在实训内容的选择、实训步骤的设计和实训文档的组织等诸方面都做了精心的考虑和安排。任课教师不需要作为专家来自己设计练习，相反，教师和学生都可以通过本书提供的实训练习来研究概念的实现。

本书的全部实训，都经过了严格的教学实践的检验，取得了良好的教学效果。根据经验，虽然大部分的实训确实能够在一次实训课的时间内完成，但学生中普遍存在着两个方面的问题：

(1) 常常会忽视对每个实训的相关知识的阅读和理解，而急功近利，只求完成实训步骤。

(2) 在实训步骤完成之后，没有投入时间对实训内容进行消化，从而不能很好地进行相关的实训总结。

因此，为了保证实训的质量，建议教师重视对教学实践环节的组织，例如：

(1) 在实训之前要求学生对相关课文内容进行预习。实训指导老师在实训开始时应该对学生的预习情况进行检查，并计入实训成绩。

(2) 明确要求学生重视对实训内容的理解和体会，认真完成“实训总结”等环节，并把这些内容作为实训成绩的主要评价成分，以激励学生对所学知识进行积极和深度的思考。

如果需要，教师还可以在现有实训的基础上，在应用实践方面做出一些要求、指导和布置，以进一步发挥学生的潜能和激发学习的主动性和积极性。

每个实训均有“实训总结”和“实训评价”部分；全部实训完成之后的实训总结部分还设计了“课程学习能力测评”等内容。希望以此方便师生交流对学科知识、实训内容的理解与体会，以及对学生学习情况进行必要的评估。如果有更多需要，请任课老师加以补充。

关于实训的评分标准

合适的评分标准有助于促进实训的有效完成。在实践中，我们摸索出如下评分安排，即对每个实训以 5 分计算，其中阅读相关课文（要求学生用彩笔标注，留下阅读记号）占 1 分，完成全部实训步骤占 2 分（完成了但质量不高则只给 1 分），认真撰写“实训总结”占 2 分（写了但质量不高则只给 1 分）。以此强调对相关课文的阅读和强调通过撰写“实训总结”来强化实训效果。

致学生

对于 IT 及其相关专业的学生来说，软件工程肯定是需要掌握的重要知识之一。但是，单凭课堂教学和一般作业，要真正领会“软件工程”课程所介绍的概念、原理、方法和技巧等，是很困难的。而经验表明，学习尤其是真正体会和掌握软件工程知识的最好方法是理论联系实际，进行充分的应用实践。

本书为读者提供了一个研究软件工程知识的学习方法，读者可以由此来学习和体验软件工程

的知识及其应用。

下面两点对于提高实训效果非常重要：

（1）在开始每一个实训之前，请务必预习各章的课文部分。课文部分包含课程知识的主体，也和实训内容有着密切的联系。

（2）实训完成后，请认真撰写每个实训的“实训总结”和最后的软件工程实训总结，完成“课程学习能力测评”等内容，把感受、认识和意见建议等表达出来，这能起到“画龙点睛”的作用，也可以此和老师进行积极的交流，以及对自己的学习情况进行必要的评估。

另一方面，可能仅靠书本所提供的实训还不够。如果需要，可以在这些实训的基础上，结合应用项目，来进一步实践网络管理技术知识，以发挥自己的潜能和激发学习的主动性与积极性。

实训设备

个人计算机在学生，尤其是专业学生中的普及，使得我们有机会把实训任务分别利用课内和课外时间来完成，以获得更多的锻炼。这样，对实验室和个人计算机的配置就有不同的要求。

实验室设备与环境

大多数用于软件工程实训的工具软件都基于 Windows 环境，用来开展软件工程实训的实验室计算机，其操作系统建议安装 Windows XP Professional 或 Windows Server 2003。

由于大多数实训都需要因特网环境的支持，因此用来进行软件工程实训的实验室环境，应该具有良好的上网条件。

个人实训设备与环境

用于软件工程实训的计算机环境，建议安装 Windows XP Professional 或 Windows Server 2003 操作系统。需要为实训准备足够的硬盘存储空间，以方便实训软件的安装和实训数据的保存。

在利用个人计算机完成实训时，要重视在操作中系统所显示的提示甚至警告信息，注意保护自己数据和计算环境的安全，做好必要的数据备份工作，以免产生不必要的损失。

没有设备时如何使用本书

当本书的读者由于某些客观原因无法获得必要的实训设备时，也不用失望，我们相信您仍将从本书中受益。全书以循序渐进的方式介绍了每个实训的背景知识和实训任务，其中也包含了相当一部分知识内容。读者通过认真阅读相关课文，仔细分析实训的操作步骤，相信也能在一定程度上有所收获。

Web 站点资源

几乎所有软件工具的生产厂商都对其产品的用户提供了足够的因特网支持，用户可利用这些支持网络来修改错误、升级系统，以及获得更新、更为详尽和丰富的技术资料。

由于网络资料的日新月异，我们不能在本书中一一罗列，有需求的读者可以上网利用搜索工具即时进行检索。

本书的相关实训素材可以从中国铁道出版社网站（edu.tg books.net）的下载区下载。下载资料中包含了与本书内容相配套的教学课件，帮助教师做一点基础的备课准备。这些教学课件也方便了学生课前课后的预习和复习。

CONTENTS

目 录

第 1 章 软件工程概述 1
1.1 计算机系统与软件 1
1.2 软件生存周期和软件生存周期过程 2
1.3 软件生存周期模型 3
1.3.1 瀑布模型 4
1.3.2 渐增模型 4
1.3.3 演化模型 5
1.4 软件工程定义 5
1.4.1 软件工程的内容 6
1.4.2 软件工程的基本目标和原则 6
1.4.3 软件工程与一般工程的差异 7
1.5 软件工具与开发环境 8
1.5.1 软件工具 8
1.5.2 软件开发环境 8
1.6 软件工程的发展 9
1.7 阅读：软件工程学科的相关学科 ... 10
1.8 习题与思考 11
1.9 实训：软件工程的计算环境 11
第 2 章 可行性研究与计划 14
2.1 可行性研究 14
2.2 软件计划 15
2.2.1 软件范围 15
2.2.2 资源 16
2.2.3 软件成本估算 16
2.3 进度安排 17
2.4 计划文件与复审 18
2.5 阅读：《人月神话》作者布鲁克斯 18
2.6 习题与思考 19
2.7 实训：软件工具与软件开发环境 20
第 3 章 软件需求分析 24
3.1 需求分析阶段的任务 24
3.2 结构化分析方法 25
3.2.1 结构化分析方法的内容 25
3.2.2 结构化分析方法的步骤 26
3.3 数据流程图 27
3.3.1 数据流程图的属性和成分 27
3.3.2 数据流程图示例 28
3.3.3 数据流程图绘制准则 29
3.4 数据字典 30
3.4.1 数据流条目 30
3.4.2 文件条目 30
3.4.3 数据项条目 31
3.4.4 加工条目 31
3.5 加工的分析与表达 32
3.5.1 加工的表达原则 32
3.5.2 结构化语言 32
3.5.3 判定表 33
3.5.4 判定树 34
3.6 需求分析文件与复审 35
3.6.1 GB/T 8567—2006 规定的文件 35
3.6.2 需求分析的复审 36
3.7 阅读：软件思想家杰拉尔德·温伯格 37
3.8 习题与思考 37
3.9 实训：软件开发绘图工具 Visio 38
第 4 章 软件概要设计 49
4.1 模块的划分 49

4.1.1 软件结构 49
4.1.2 模块划分的基本原则 50
4.1.3 内聚度 50
4.1.4 耦合度 51
4.1.5 高内聚和低耦合 52
4.1.6 模块划分的方法 52
4.2 结构化设计方法 53
4.2.1 变换与事务型数据流分析 53
4.2.2 模块化设计 53
4.2.3 模块结构图 54
4.2.4 从数据流程图导出模块结构图 55
4.3 Parnas 方法 57
4.3.1 信息隐蔽原则 57
4.3.2 加强系统各成分间的检查 57
4.4 Jackson 方法 58
4.5 程序的逻辑构造方法 59
4.6 概要设计文件与复审 59
4.6.1 概要设计说明书 59
4.6.2 概要设计的复审 59
4.7 阅读：Parnas：把软件工程作为一门真正的工程学科（1） 60
4.8 习题与思考 62
4.9 实训：使用 Visio 绘制工程图形 64
第 5 章 软件详细设计 69
5.1 概述 69
5.2 结构化构造 70
5.3 图形设计工具 70
5.3.1 程序流程图 70
5.3.2 方块图 72
5.3.3 HIPO 图 73
5.3.4 PAD 图 74
5.4 伪码与程序设计语言 75
5.5 各种详细设计工具的比较 77
5.6 详细设计文件与复审 78
5.6.1 详细设计说明书 78
5.6.2 详细设计的复审 78
5.7 阅读：Parnas：把软件工程作为一门真正的工程学科（2） 78
5.8 习题与思考 82
5.9 实训：软件工程国家标准 82
第 6 章 软件编码 88
6.1 结构化程序设计方法 88
6.2 程序设计风格 89
6.2.1 源程序 89
6.2.2 数据说明 90
6.2.3 语句结构 90
6.3 源代码文件 91
6.3.1 综合文件 91
6.3.2 程序组织文件 92
6.3.3 指令级注释 92
6.4 程序设计技术 92
6.4.1 冗余程序设计 93
6.4.2 防错性程序设计 93
6.4.3 程序设计的质量 93
6.4.4 编译程序和解释程序 94
6.5 编程语言的特点 94
6.5.1 过程性语言 94
6.5.2 说明性语言 95
6.5.3 脚本语言 95
6.5.4 低级语言 96
6.5.5 高级语言 96
6.5.6 面向对象语言 96
6.5.7 事件驱动语言 97
6.5.8 构件（组件） 97
6.6 选择编程语言 97
6.7 编码文件与复审 99
6.8 阅读：19 世纪的传奇合作——巴贝奇与阿达 99
6.9 习题与思考 100
6.10 实训：软件测试环境 103
第 7 章 软件测试 106
7.1 测试的基本概念 106
7.2 测试方法 107

7.2.1 静态分析技术 107
7.2.2 动态测试技术 108
7.3 单元测试 109
7.4 组装测试 110
7.4.1 组装测试的任务 110
7.4.2 组装测试的方式 111
7.5 确认测试 111
7.6 测试用例设计 112
7.6.1 白盒法 112
7.6.2 黑盒法 114
7.7 测试工具与测试自动化 116
7.7.1 基于 GUI 的自动化测试 116
7.7.2 自动化测试工具的特征 117
7.7.3 自动化测试工具的分类 118
7.7.4 α、β 测试 119
7.8 测试文件与复审 119
7.8.1 GB/T 8567—2006 规定的文件 120
7.8.2 GB/T 9386—1988 规定的文件 120
7.8.3 测试的复审 120
7.9 排错技术与系统转换 121
7.10 阅读：从程序员到软件测试工程师 121
7.11 习题与思考 124
7.12 实训：黑盒法与白盒法设计测试用例 126

第 8 章 软件维护 133

8.1 概述 133
8.1.1 软件维护工作的必要性 133
8.1.2 软件维护的内容 133
8.1.3 维护工作的过程 134
8.2 软件的可维护性 135
8.3 软件维护的管理 136
8.4 系统分析与建模工具 PowerDesigner 137
8.5 阅读：软件工程学科的内涵 139
8.6 习题与思考 141
8.7 实训：PowerDesigner 入门 142

第 9 章 软件质量管理 151

9.1 软件项目特点与软件管理职能 151
9.1.1 软件项目的特点 151
9.1.2 软件管理的主要职能 151
9.2 对软件质量的需求 152
9.2.1 用户的质量观 152
9.2.2 开发人员的质量观 152
9.2.3 维护人员的质量观 153
9.2.4 管理人员的质量观 153
9.3 软件质量度量 153
9.3.1 软件质量框架模型 153
9.3.2 软件质量特性 154
9.3.3 评估指标的选取原则 154
9.4 软件质量评估指标体系 155
9.4.1 功能度指标 155
9.4.2 可靠性指标 155
9.4.3 易用性指标 156
9.4.4 效率特性指标 156
9.5 CMM：软件能力成熟度模型 157
9.6 PowerDesigner 的 CDM 158
9.7 阅读：《未来之路》和《数字化生存》 159
9.8 习题与思考 160
9.9 实训：PowerDesigner 业务处理模型 161

第 10 章 软件配置管理 177

10.1 软件配置管理的概念 177
10.2 配置管理软件 VSS 177
10.2.1 VSS 的主要功能 177
10.2.2 软件配置管理员的任务 178
10.2.3 项目组其他人员的任务 179
10.2.4 与 Visual Studio IDE 集成 179
10.3 阅读：软件产业的设计大师和 VB 之父 Alan Cooper 179
10.4 习题与思考 180

10.5 实训：软件配置管理工具VSS.....181
第 11 章 面向对象分析与设计.....186
11.1 面向对象方法.....186
11.1.1 面向对象方法的特点.....187
11.1.2 面向对象软件工程.....188
11.2 面向对象的概念.....189
11.2.1 对象.....189
11.2.2 类.....189
11.2.3 消息传递.....190
11.2.4 多态性.....190
11.3 面向对象软件的开发过程.....190
11.3.1 类生存期.....190
11.3.2 面向对象的开发.....192
11.4 面向对象分析—— OOA.....193
11.4.1 OOA 的基本内容.....193
11.4.2 常用的 OOA 方法.....194
11.4.3 论域分析.....195
11.5 面向对象设计——OOD.....197
11.5.1 高层设计.....198
11.5.2 类设计的目标和方针.....199
11.5.3 通过复用设计类.....200
11.5.4 类设计方法.....200
11.6 PowerDesigner 的 CDM.....202
11.7 阅读：极限编程 XP 的先驱 Kent Beck.....204
11.8 习题与思考.....205
11.9 实训：PowerDesigner 概念数据模型.....206
第 12 章 面向对象的实现.....223
12.1 面向对象编程.....223
12.2 面向对象测试.....224
12.2.1 面向对象测试的特点.....224
12.2.2 面向对象的测试步骤.....225
12.2.3 面向对象的测试策略.....227
12.3 面向对象软件的测试用例设计.....228
12.3.1 传统测试用例设计方法的可用性.....228
12.3.2 基于故障的测试设计.....228
12.3.3 基于场景的测试设计.....229
12.3.4 测试表层结构和深层结构.....229
12.4 PowerDesigner 的 PDM.....229
12.4.1 表、列、视图、主键、候选键、外键.....230
12.4.2 存储过程和触发器.....230
12.4.3 默认值与规则.....231
12.4.4 完整性检查约束.....231
12.4.5 索引.....232
12.4.6 检查 PDM 对象.....232
12.5 阅读：CASE 与信息工程的创始人 James Martin.....232
12.6 习题与思考.....233
12.7 实训：PowerDesigner 物理数据模型.....233
第 13 章 统一建模语言——UML.....251
13.1 UML 概述.....251
13.2 PowerDesigner 的 OOM.....252
13.2.1 用例图.....253
13.2.2 时序图.....254
13.2.3 类图.....255
13.3 阅读：软件开发的教父 Martin Fowler.....255
13.4 习题与思考.....255
13.5 实训：PowerDesigner 面向对象模型.....256
第 14 章 软件文件.....281
14.1 目的和作用.....281
14.2 软件生存周期与各种文件的编制.....282
14.3 文件编制中考虑的因素.....282
14.3.1 文件的读者.....283
14.3.2 文件内容的重复性.....283
14.3.3 文件内容的灵活性.....283
14.4 文件编制的质量要求.....284
14.5 文件的管理和维护.....284
14.5.1 文件的形成.....284

14.5.2 文件的分类与标识.....284
14.5.3 文件控制 285
14.5.4 文件的修改管理.....285
14.5.5 《软件文档管理指南》（GB/T 16680—1996）....285
14.6 阅读：软件工程的七条基本原理.....286
14.7 习题与思考 287
14.8 实训：软件产品开发文件编制指南.....288
第 15 章 软件工程实训总结 293
15.1 实训的基本内容 293
15.2 实训的基本评价 295
15.3 课程学习能力测评 295
15.4 软件工程实训总结 295
15.5 实训总结评价（教师）.....296
附录 A 课程设计..... 298
附录 B 软件文档编写提示（部分）... 300
参考文献..... 305

第1章 软件工程概述

随着计算机系统的迅速发展和应用范围的日益广泛，计算机软件的规模越来越大，其复杂程度也不断增加。从20世纪60年代以来，人们开始逐渐认识到确实存在“软件危机”这样一个事实。例如：

① 软件生产不能满足日益增长的需要。

② 对软件开发成本和开发进度的估计往往不够准确。实际成本有时高出预计成本好几倍，预计完工的时间往往推迟几个月，甚至更长时间。

③ 软件开发人员和用户之间信息交流不充分，用户对完成的软件满意度很低。

④ 软件价格昂贵，软件成本在整个计算机系统中所占的比例急剧上升，软件已成为许多计算机系统中花费最高的项目。

⑤ 软件质量难以保证，质量保证技术还没有真正应用到软件开发的全过程。

⑥ 软件可维护性差，程序中的错误很难改正，或者当硬件环境发生变化时，想要进行适应性或完善性维护却极其困难。

导致这一系列问题的一个重要原因，就是软件的研制和维护本身是工程性的任务，但软件开发人员所采取的方式却未能工程化。

为了克服软件危机，促使人们开始考虑采用工程化方法和工程途径来研制和维护软件。20世纪60年代末至70年代初，逐渐发展起一组总称为“软件工程”的技术。这些技术把软件作为一个工程产品来处理：它需要计划、分析、设计、实现、测试以及维护。

1.1 计算机系统与软件

硬件工程和软件工程都可以看成是一门更广义的学科——计算机系统工程的一部分。

用于计算机硬件的工程技术是由电子设计技术发展起来的。今天，硬件设计技术已经达到比较成熟的水平。目前，制造方法仍在不断地改进，可靠性已是一种可以期待的现实，而不再是一种愿望。

但是，计算机软件工程却还处于某种困境之中。在以计算机为基础的系统中，软件已经取代硬件成为系统中设计起来最困难、最不容易成功（按时完成和不超过预计的成本），而且是最不易管理的部分。另一方面，随着以计算机为基础的系统在数量、复杂程度和应用范围上的不断增长，对软件的需求仍然有增无减。

计算机系统工程的主要内容是对系统所要求的功能加以揭示、分析，并把它们分配给系统

的各个部分，如图 1-1 所示。

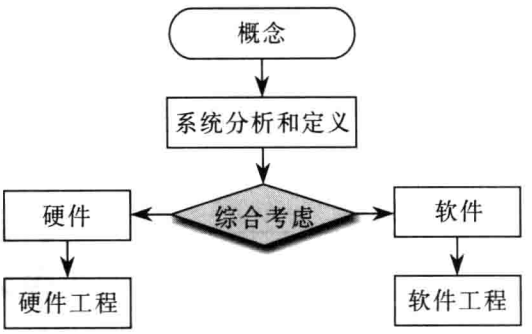


图 1-1 计算机系统工程

在大多数新系统创建时，对系统所要求实现的功能往往只有模糊的概念。系统分析和定义的目的在于项目的范围，这就要对需要进行处理的信息、所要求实现的功能、所期望的性能以及设计的约束和检验的标准等进行系统、详细的分析。

在范围确定之后，计算机系统工程师必须考虑多种能潜在地满足项目范围的、可供选择的配置。在综合考虑各项因素之后，选择其中的一种配置，并将系统的功能分配给系统的各个部分（例如硬件和软件）。

2006 年出版的《中国大百科全书》给软件下的定义是：软件是计算机系统中的程序和有关的文件。程序是计算任务的处理对象和处理规则的描述；文件是为了便于了解程序所需的资料说明。程序必须装入机器内部才能工作，文件一般是给人看的，不一定装入机器。程序作为一种具有逻辑结构的信息，精确而完整地描述计算任务中的处理对象和处理规则。这一描述还必须通过相应的实体才能体现。

也就是说，“软件”不仅仅是指程序，在软件研制过程中按一定规格产生的各种文件也是软件不可缺少的组成部分。

1.2 软件生存周期和软件生存周期过程

《软件工程术语》（GB/T 11457—2006）定义了软件生存周期，即从设计软件产品开始到产品不能再使用为止的时间周期。亦即一个计算机软件，从出现一个构思之日起，经过开发成功投入使用，在使用中不断增补修订，直到最后决定停止使用，并被另一个软件代替之时止，被认为是该软件的一个生存周期（或称生命周期、生存期）。

一个软件产品的生存周期可以划分成若干个互相区别而又有联系的阶段，每个阶段中的工作均以上一阶段工作的结果为依据，并为下一阶段的工作提供前提。经验表明，失误造成的差错越是发生在生存周期的前期，在系统交付使用时造成的影响和损失就越大，要纠正它所付出的代价也越高。因而在前一阶段工作没有做好之前，一定不要草率地进入下一阶段。

《软件生存周期过程》（GB/T 8566—2001，后升级为 GB/T 8566-2007）则根据软件工程的实践和软件工程学科的发展，进一步完善了软件生存周期的定义，即从概念形成直到退役，并且由获取和供应软件产品及服务的各个过程组成。该标准把软件生存周期中可以开展的活动分为 5 个基本过程（获取过程、供应过程、开发过程、运作过程和维护过程）、9 个支持过程（文档编

制过程、配置管理过程、质量保证过程、验证过程、确认过程、联合评审过程、审核过程、问题解决过程和易用性过程）和 7 个组织过程（管理过程、基础设施过程、改进过程、人力资源过程、资产管理过程、重用大纲管理过程和领域工程过程）。

软件生存周期过程中各阶段的划分，有助于软件研制管理人员借用传统工程的管理方法（重视工程性文件的编制，采用专业化分工方法，在不同阶段使用不同的人员等），从而有利于明显提高软件质量、降低成本、合理使用人才，进而提高软件开发的劳动生产率。

由于工作的对象和范围的不同以及经验的不同，对软件生存周期过程中各阶段的划分也不尽相同。但是，这些不同划分有许多相同之处。《计算机软件开发规范》（GB/T 8566—1988）（《软件生存周期过程》GB/T 8566—2001 的前身）将软件生存周期划分为以下 8 个阶段，即可行性研究与计划、需求分析、概要设计（即结构设计）、详细设计、实现（包括单元测试）、组装测试（即集成测试）、确认测试、使用和维护。

软件生存周期是对软件的一种长远发展的看法，这种看法把软件开始开发之前和软件交付使用之后的一些活动都包括在软件生存周期之内。应当注意的是，软件系统的实际开发工作不可能直线地通过分析、设计、编程和测试等阶段，出现各阶段间的反复是不可避免的。

软件生存周期的每个阶段都要产生一定规格的软件文件移交给下一阶段，使下一阶段在所提供的软件文件的基础上继续开展工作。《计算机软件产品开发文件编制指南》（GB/T 8567—1988，已升级为 GB/T 8567—2006《计算机软件文档编制规范》）建议在软件的开发过程中编制的文件有 14 种。而《计算机软件需求说明编制指南》（GB/T 9385—1988）、《计算机软件测试文件编制规范》（GB/T 9386—1988）等有关软件工程的国家标准对软件文件的编制提出了更为详尽的要求，《软件文档管理指南》（GB/T 16680—1996）则明确了对软件文件的管理要求。

图 1-2 和图 1-3 分别说明了在典型的情况下，软件生存周期各阶段的工作量所占的比重。图 1-2 说明运行维护工作量要占整个生存周期工作量的一半以上，图 1-3 则说明测试阶段（组装测试和确认测试）的工作量约占整个开发期工作量的一半。

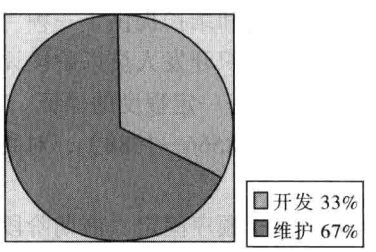


图 1-2　软件生存周期工作量分配

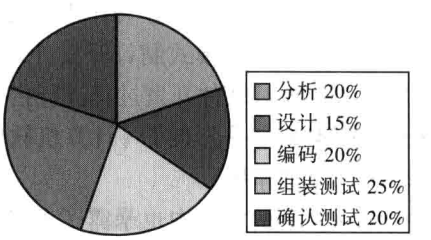


图 1-3　开发阶段工作量分配

1.3　软件生存周期模型

软件生存周期模型（又称软件开发模型）是软件工程的一个重要的概念，它可以定义为一个框架，它含有遍历系统从确定需求到终止使用这一生存周期的软件产品的开发、运行和维护中需实施的过程、活动和任务。

软件生存周期模型能清晰、直观地表达软件开发全过程，明确规定了开发工作各阶段所要完成的主要活动和任务，以作为软件项目开发工作的基础。对于不同的软件系统，可以采用不同的开发方法、使用不同的程序设计语言以及各种不同技能的人员参与工作、运用不同的管理

方法和手段等，以及允许采用不同的软件工具和不同的软件工程环境。软件生存周期模型是稳定有效和普遍适用的。

在软件生存周期过程中，软件生存周期模型仅对软件的开发、运行和维护过程有意义，在信息技术国际标准 ISO 12207 和 ISO 9000-3 中都提到软件生存周期模型，包括瀑布模型、渐增模型、演化模型、螺旋模型、喷泉模型和智能模型等。

1.3.1 瀑布模型

瀑布模型是 1970 年 W. Royce 提出的最早的软件生存周期开发模型（见图 1-4），它将软件开发过程中的各项活动规定为依固定顺序连接的若干阶段工作，形如瀑布流水，最终得到软件系统或软件产品。换句话说，它将软件开发过程划分成若干个互相区别而又彼此联系的阶段，每个阶段的工作都以上一个阶段工作的结果为依据，同时为下一个阶段的工作提供前提。

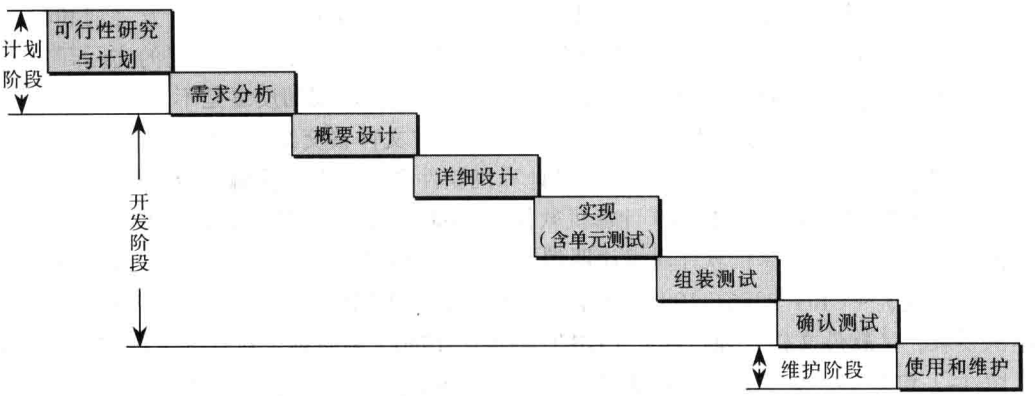


图 1-4 软件生存周期的瀑布模型

瀑布模型之所以能广泛流行，一方面由于它在支持开发结构化软件、控制软件开发复杂度、促进软件开发工程化方面起了显著作用；另一方面它为软件开发和维护提供了一种当时较为有效的管理模式，根据这一模式制订开发计划、进行成本预算、组织开发人员以阶段评审和文档控制为手段，有效地对软件开发过程进行指导，从而使软件质量有一定程度的保证。1988 年，曾依据该开发模型制定并公布了《计算机软件开发规范》（GB/T 8566—1988），对我国软件开发起到了较大的促进作用。

瀑布模型在大量的实践中也暴露了不足和问题。例如，由于顺序固定，前期阶段工作中造成的差错，越到后期阶段所造成的损失和影响也越大，为了纠正它而付出的代价也越高，尽管技术人员小心翼翼，但这种情况还是会经常发生。

1.3.2 渐增模型

渐增模型是指从一组给定的需求开始，通过构造一系列可执行的中间版本来实施开发活动。第一个中间版本纳入一部分需求，下一个中间版本纳入更多的需求，依此类推，直到系统完成。每个中间版本都要执行必要的过程、活动和任务，例如，需求分析和体系结构设计需要执行一次，而详细设计、编码和测试、软件组装和验收测试在每个中间版本构造过程中都执行，如图 1-5 所示。

在开发每个中间版本时，开发过程中的活动和任务顺序地或部分平行地使用。当相继的中间版本在部分并行开发时，开发过程中的活动和任务可以在各中间版本间平行地被采用。

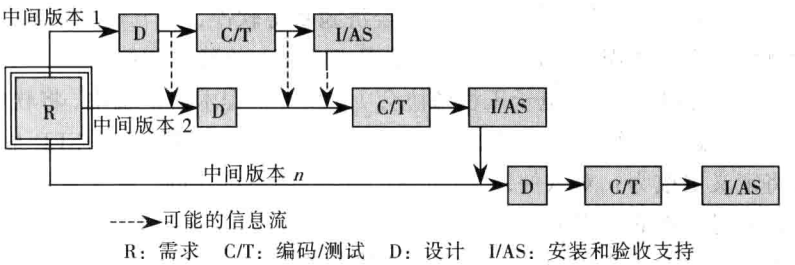


图 1-5　渐增模型示意

1.3.3　演化模型

演化模型（又称进化模型）主要针对事先不能完整定义需求的软件项目开发。由于人们对软件需求的认识模糊，许多软件开发项目很难一次开发成功，返工再开发难以避免。因此，人们对需要开发的软件给出基本需求，进行第一次试验开发，其目标仅在于探索可行性和弄清需求，取得有效的反馈信息，以支持软件的最终设计和实现。通常，把第一次试验性开发出来的软件称为原型。这种开发模型可以减少由于需求不明将给开发工作带来的风险。

演化模型（见图 1-6）也是通过构造系统的各个可执行的中间版本来开发系统的。但是，与渐增模型的区别是演化模型承认需求不能被完全了解，且不能在开始时就确定。在该模型中，需求一部分被预先定义，然后在每个相继的中间版本中逐步完善。该模型中，每个中间版本在开发时，开发过程中的活动和任务顺序地或部分重叠平行地被采用。

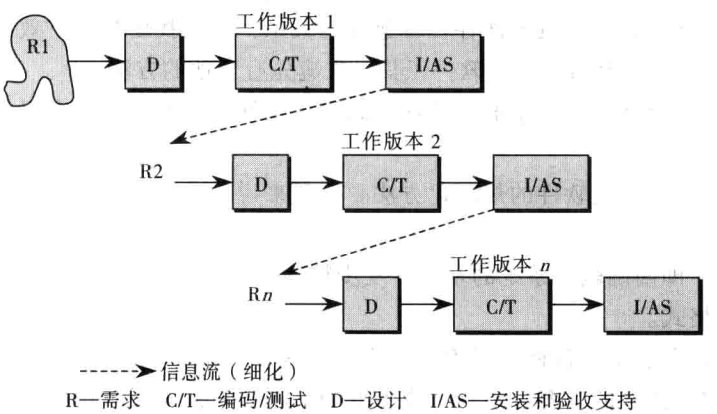


图 1-6　演化模型示意

对所有的中间版本，开发过程中的活动和任务通常按同一顺序被重复使用。维护过程和运行过程可以与开发过程平行地使用。获取过程、供应过程、支持过程和组织过程通常与开发过程平行地使用。

1.4　软件工程定义

人们对软件工程的定义和内涵有一个广泛讨论、研究和认识的过程，从而对软件工程有了各种各样的定义。例如：