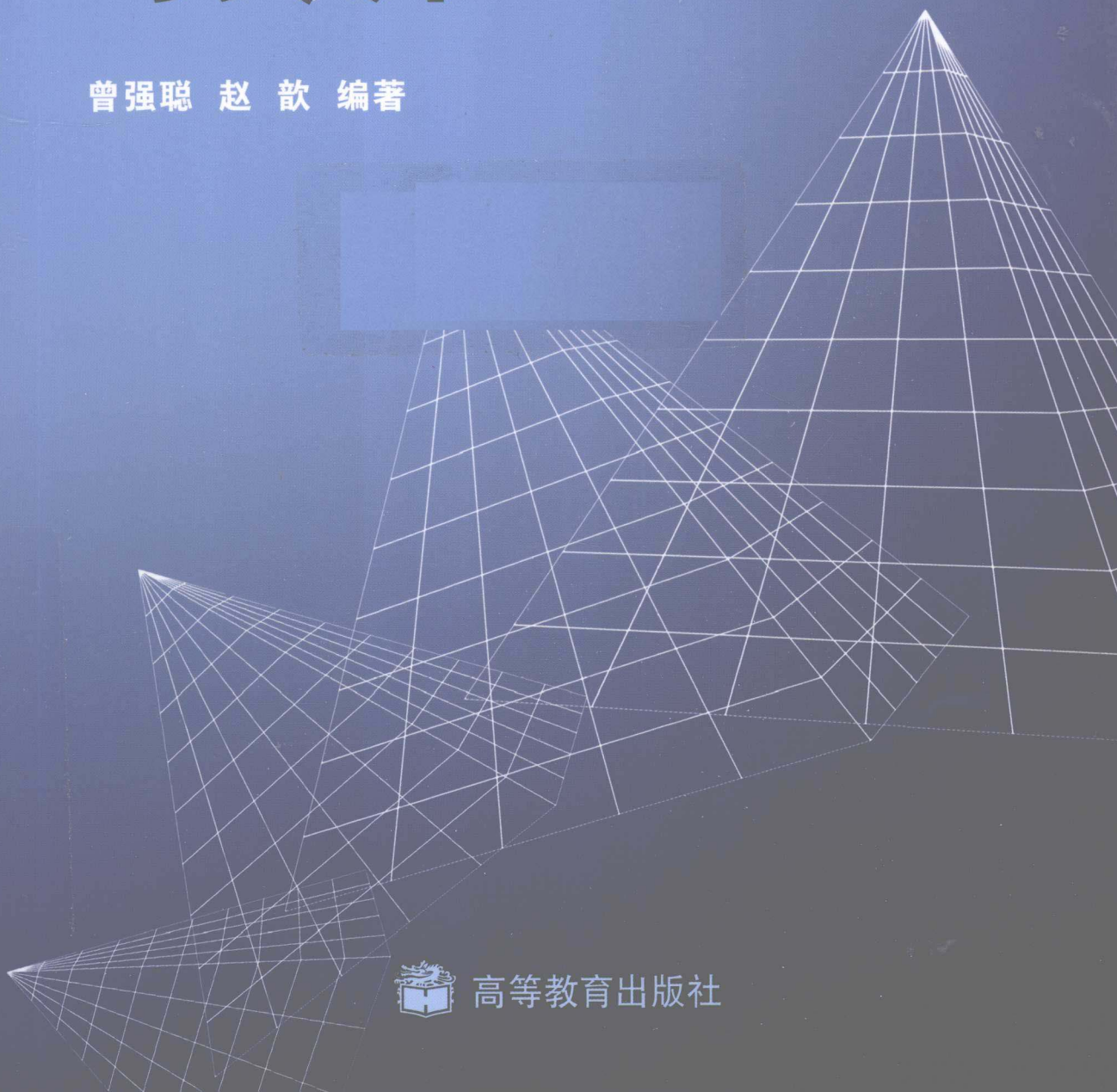
普通高等教育“十一五”国家级规划教材

# 软件工程方法与实训

曾强聪 赵 歆 编著



高等教育出版社

普通高等教育“十一五”国家级规划教材

# 软件工程方法与实训

Ruanjian Gongcheng Fangfa yu Shixun

曾强聪　赵　歆　编著

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

## 内容提要

本书是普通高等教育“十一五”国家级规划教材。

软件工程是很具实用性的工程方法学，是软件开发者开发和维护软件时的作业指南。本书基于实用的原则进行编写，不仅有较完整的软件工程知识体系，而且有较好的教学案例，有与工程实训相适应的编排结构，可较好地辅助软件工程实用人才的培养。

全书共14章。第1～3章为软件工程方法基础，以提供预备性学习，包含概述、过程模式、项目管理等内容。第4～14章基于软件生存周期编排，以利于理论教学与实训教学的同步，包含系统工程、需求分析、概要设计、结构化建模、面向对象建模、界面设计、算法设计、软件测试、软件维护等内容。

本书可作为应用性、技能型人才培养的应用型本科、高职高专教育软件工程课程的教材，也可作为软件工程领域技术人员的技术参考书，并可供广大软件工程学习者自学。

**图书在版编目(CIP)数据**

软件工程方法与实训/曾强聪，赵歆编著. —北京：高等教育出版社，2010.5

ISBN 978-7-04-028798-1

Ⅰ.①软… Ⅱ.①曾… ②赵… Ⅲ.①软件工程-高等学校-教材 Ⅳ.①TP311.5

中国版本图书馆CIP数据核字(2010)第052436号

**策划编辑** 冯 英 **责任编辑** 焦建虹 **封面设计** 张志奇 **责任绘图** 黄建英
**版式设计** 余 杨 **责任校对** 姜国萍 **责任印制** 陈伟光

| | | | |
|---|---|---|---|
| **出版发行** | 高等教育出版社 | **购书热线** | 010-58581118 |
| **社 址** | 北京市西城区德外大街4号 | **咨询电话** | 400-810-0598 |
| **邮政编码** | 100120 | **网 址** | http://www.hep.edu.cn |
| **总 机** | 010-58581000 | | http://www.hep.com.cn |
| | | **网上订购** | http://www.landraco.com |
| **经 销** | 蓝色畅想图书发行有限公司 | | http://www.landraco.com.cn |
| **印 刷** | 北京七色印务有限公司 | **畅想教育** | http://www.widedu.com |
| **开 本** | 787×1092 1/16 | **版 次** | 2010年5月第1版 |
| **印 张** | 15.5 | **印 次** | 2010年5月第1次印刷 |
| **字 数** | 380 000 | **定 价** | 23.00元 |



**物料号 28798-00**

# 前　言

软件工程是很具实用性的工程方法学，是软件开发者——软件项目负责人、软件分析师、软件设计师、程序员、测试员，开发和维护软件时的作业指南。

20世纪60年代，软件开发遭遇危机——难以满足用户需求、容易出错、难于维护，引发软件开发者的心理恐慌，使软件开发陷入困境，并因此开始探索用工程方法解决软件问题。软件工程即诞生于这样的背景之下，其谋求对软件开发、维护能有方法上的指导，能有规则上的约束，以使面对有待开发的复杂软件系统，开发者能更具预见力与信心。

软件工程已是计算机科学领域中的重要分支，其成长则与软件产业化发展密切相关。软件产业化发展需要软件工程方法提供理论支持，而伴随着软件产业的迅速发展，软件工程也在不断进步与完善，产生了结构化、面向对象等诸多方法学体系，涉及工程技术、工程管理、工程经济等诸多方面的内容，可对软件开发提供比较全面的工程支持。

软件工程能够带来高质量的软件产品，因此，软件工程人才备受业界重视，不仅软件工程师代替了早期的程序员，成为软件开发中的技术骨干，而且要求所有从事软件开发的人都具备一定的软件工程知识与技能，以保证软件工程规范的有效实施。

本书为应用性、技能型人才培养的应用型本科、高职高专教育计算机软件专业学生学习软件工程而编写。或许有学习者认为，应用型本科、高职高专层次的学生毕业之后一般不会从事软件分析、设计或项目管理等高层工作，而是从事编码、维护或整理文档等低层工作，因此软件工程方法学对他们来说价值不大。但实际上，目前软件行业中的许多高层分析师、设计师、项目经理就来自应用型本科、高职高专，并且即使这些学习者一直从事低层编码、维护、整理文档工作，也仍有必要学习软件工程，因为即使是低层软件实现，也必须遵循软件工程规范。

毫无疑问，软件行业对软件技术人才的要求越来越注重实际应用。因此，学习软件工程，不仅是学习知识体系，还应该接受一定的与项目有关的工程实训，以获得一定的软件工程实际应用能力。实际上，软件工程本身即是实践性学科，一系列的方法规则就建立于工程实践的基础上。因此，其学习也必然需要通过实践、实训，才能真正、有效把握。本书即立足于实际应用介绍软件工程，并从结构编排、教学案例等诸多方面考虑了实训教学的便利性。

本书有针对课程实训的专门结构编排。全书共14章，其中第1~3章为软件工程方法基础，用于完成预备性学习，积累必要知识，提出项目问题，组织适宜于学习的项目团队；第4~14章则基于软件生存周期编排，基本过程是项目可行性分析、软件需求分析、软件结构设计、程序界面设计、程序算法设计与编码、软件测试集成、软件维护。显然，这种编排有利于开展实训，可使理论教学与实训教学基本同步。

本书重点是软件分析与设计,并主要体现为分析设计建模。对于分析设计建模,本书有以下方面的教学突破,可利于学习者更好地把握建模方法。

其一,结构化分析中的数据流细化建模一直是一个学习难题,其实它也是实际建模中的一个应用难点,即不知如何进行功能细化。本书基于业务树实现功能逐级分解,可使这个难点得以较好处理。显然,其具有教学意义,同时也具有实用价值。

其二,对于结构化方法中基于数据流的程序结构映射,许多人认为实用性不大,主要是认为其与实际程序结构并不一致。本书基于映射结构到优化结构的过渡,较好地表现了映射结构与实际结构的一致性。

其三,面向对象建模过程是基于用例驱动的,以使有关系统的业务分析结论可延伸到系统的结构设计、界面设计、安装部署。然而,以往的大多数教科书仅局限于纲领性说明,而并无实际建模体现。本书则对此有较好的基于案例的建模过程说明,如分析中基于用例的活动建模,设计中基于用例的类图结构建模、对象协作建模。诸多案例无不体现出用例驱动的价值。

其四,无论结构化建模、面向对象建模,本书都有较好的基于案例的由分析到设计的全过程推演,以反映软件系统构建的严密性。

本书提供了比较丰富的教学案例。这些案例大多来自实际软件问题,有较强的可操作性与可应用性,并都按照教学要求精心设计,以满足软件工程理论与实训的结合。诸多案例还大多在习题中得到延伸,可供学习者做进一步探讨,以利于知识向应用能力的升华。

软件工程教学应该以有用、能用、实用为基本宗旨。显然,经过软件开发者长期实践与努力探索而获得的工程方法、规则肯定是有用的,值得认真学习。然而,有用还只能体现出软件工程的知识特性,若要使这些知识转化为学习者的能力,则必须能用。

教学实践表明,案例教学更有利于能力培养。本书有较丰富的案例,目的就在于开展案例教学,培养学习者的能力,以使书中知识能够被学习者活用。软件工程教学不仅需要能用,并还需要实用,可用来解决实际问题。教学实践还表明,项目实训有利于基础知识、一般能力与实际应用的结合。应该说,本书有较好的便于实训的结构编排。

项目实训的有效开展还依赖于教学者的精心组织。一种可行的项目实训组织模式是,三人一组进行实训,并以组为单位进行整体评价,但各成员应有特定任务,因此还需要对各成员进行评价。这种模式既可考查团队协作,又可体现个人成就。由于是团队合作,学习者大多有较高的实训热情,因此有较低的无效实训风险,能产生较好的实训效果。实际上,编者即一直基于这种模式开展项目实训教学,并产生了良好的实训效果。

本书篇幅不大,但实质内容不少。这也就是说,如要完成本书全部内容教学,则必须有足够的教学课时保证。通常情况下,需要安排 55 ~ 75 教学课时,其中理论课时 45 ~ 50,实训课时 10 ~ 25。显然,实训课时灵活性较大,如果要进行较完整的项目实训,必然需要较多的实训课时,通过压缩实训内容则可减少课时,但基本的分析设计建模实训总是需要的,否则软件工程课程学习实用价值不大。

下面是对各章节基于实训的教学说明,可供实际教学参考。

● 第 1 章软件工程概述(3 课时):有关软件工程的概括性描述,涉及与软件工程相关的一些概念、方法,可使学习者对软件工程有初步认识,以方便后续章节学习。本章还将介绍三个常

用的软件工具(Visio、PowerDesigner、Rational Rose),后续章节中的模型即依靠这些工具创建。

• 第2章软件开发过程模式(3课时):介绍软件生存周期,并在此基础上介绍几种常用的软件过程模式——瀑布模式、原型进化模式、增量模式。过程模式是项目框架,可针对有关项目进行问题讨论,以比较这几种过程模式的优劣。

• 第3章软件项目管理(4~6课时):涉及项目团队、项目计划、项目成本、项目成果等诸多管理要素。可在本章启动项目实训,如进行项目分组,指定项目负责人,说明项目评价机制,以使学生对项目管理有更直观的认识。本章中的甘特图、任务网络图由Microsoft Project创建,可安排2课时实训。

• 第4章计算机系统工程(4~6课时):本章是软件生存周期起点,需要从全局角度考查软件问题,涉及内容有计算机体系结构、软件系统高层分析、软件项目可行性分析。本章还涉及系统初步建模,其中的系统框架图、系统流程图都由Visio中的基本流程图创建,可安排2课时实训,以使学生初步熟悉Visio建模。

• 第5章需求分析(4~6课时):有关软件需求分析的一般性说明,涉及任务、过程,并对业务需求获取与建模有专门讨论。本章中的业务树由Visio中的组织图创建,业务用例、业务活动则由Rational Rose创建,可安排2课时实训。

• 第6章结构化分析建模(4~8课时):需求分析中结构化方法的延伸,内容有数据建模、功能建模、行为建模。本章中的实体联系图由PowerDesigner创建,数据流图由Visio创建,状态图由Rational Rose创建,需安排2~4课时实训。

• 第7章基于UML的面向对象分析建模(4~8课时):需求分析中面向对象方法的延伸,内容有用例建模、活动建模、类分析建模。本章诸多模型都由Rational Rose创建,需安排2~4课时实训。

• 第8章概要设计(4~6课时):有关软件概要设计的一般性说明,涉及内容包括任务及过程、系统构架、数据结构、程序结构。本章还对数据库设计有较详细的说明,并通过PowerDesigner创建设计模型,需安排2课时实训。

• 第9章结构化设计建模(4~6课时):概要设计中结构化方法的延伸,核心内容是基于数据流的结构映射。程序结构图由Visio创建,需安排2课时实训。

• 第10章基于UML的面向对象设计建模(4~8课时):概要设计中面向对象方法的延伸,内容有逻辑结构设计、动态过程设计、物理装配与部署。本章诸多模型都由Rational Rose创建,需安排4课时实训。

• 第11章用户界面设计(2~4课时):简要介绍界面设计,涉及界面设计特点、界面类型、界面功能、界面要素等内容。如项目要求实现,可安排2课时实训。

• 第12章算法设计与编码(2~4课时):简要介绍算法设计,说明几种算法设计工具,对Jackson方法有初步介绍。如项目要求实现,可安排2课时实训。

• 第13章软件测试(2~4课时):简要说明软件测试,涉及测试目的、测试任务、测试用例设计、程序调试等。如项目要求实现,可安排2课时实训。

• 第14章软件维护(2课时):简要说明软件维护,涉及维护分类、软件可维护性、维护实施、逆向工程与再工程等。可基于实际问题进行软件维护讨论。

本书创作过程中编者阅读了大量的图书文献。可以说,正是依靠这些极具价值的图书文献,编者的认识才更加深刻,创作思路才更加敏捷。这些图书文献已在书后参考文献中列出,学习者在学习本书之时,也可通过参阅这些书籍而开阔视野。

本书由曾强聪执笔创作,赵歆承担了部分案例设计,并提出了很好的与本书相适应的教学模式建议。无疑,写作本书是一件非常耗费心力的事情。首先是结构,它应该是严谨的,并能很好满足应用型本科、高职高专层次实训教学的需要。接着是语言,其应该既严密又畅快,以便于阅读,并能尽量吸引学习者去阅读。再接下来就是内容的取舍、概念的定义、案例的设计。实际上,本书中的每一细节都有反复推敲,因为毕竟是教科书,因此很担心因工作疏忽而给学习者带来误导。

为确保质量,书稿完成后特委托王四春教授对本书进行了全面审稿,熊署初教授、何小东教授、赵歆老师、曹伟老师、刘震老师、周纳老师、罗毅辉老师、丁启华老师、王雷老师、杨文东老师等参加了审稿讨论,并就适合于本书的教学模式进行了专题研讨。在此特向他们表示感谢!

然而,毕竟编者学识水平与时间都有限,因此本书难免会有缺点与不足,特请广大读者批评指正,以使本书再版时能够不断完善。

编 者

2010 年 3 月

# 目　录


第1章　软件工程概述 …… 1
1.1　软件 …… 1
1.1.1　软件概念 …… 1
1.1.2　软件特点 …… 2
1.1.3　软件分类 …… 4
1.2　软件工程 …… 6
1.2.1　工程技术 …… 7
1.2.2　工程管理 …… 9
1.2.3　工程目标 …… 11
1.3　主流方法学 …… 11
1.3.1　结构化方法学 …… 11
1.3.2　面向对象方法学 …… 12
1.4　常用软件工具 …… 14
1.4.1　Visio …… 14
1.4.2　PowerDesigner …… 15
1.4.3　Rational Rose …… 15
小结 …… 17
习题 …… 17
第2章　软件开发过程模式 …… 19
2.1　软件生存周期 …… 19
2.1.1　软件定义期 …… 19
2.1.2　软件开发期 …… 20
2.1.3　软件运行与维护期 …… 21
2.2　瀑布模式 …… 22
2.2.1　瀑布模式的特点 …… 22
2.2.2　瀑布模式的作用 …… 23
2.3　原型进化模式 …… 24
2.3.1　软件原型 …… 24
2.3.2　原型进化过程 …… 24
2.4　增量模式 …… 25
2.4.1　增量开发过程 …… 26
2.4.2　增量模式的特点 …… 27
2.4.3　增量模式的优越性 …… 27
小结 …… 28
习题 …… 28
第3章　软件项目管理 …… 29
3.1　开发团队 …… 29
3.1.1　软件开发机构 …… 29
3.1.2　软件项目组 …… 30
3.1.3　项目组管理机制 …… 31
3.2　项目计划 …… 34
3.2.1　任务分配 …… 34
3.2.2　项目进度计划 …… 35
3.2.3　项目计划书 …… 36
3.3　项目成本估算 …… 37
3.3.1　程序代码行成本估算 …… 37
3.3.2　软件功能点成本估算 …… 39
3.3.3　软件过程成本估算 …… 42
3.4　软件文档管理 …… 43
3.4.1　文档概念 …… 43
3.4.2　文档分类 …… 44
3.4.3　软件文档与软件生存周期之间的关系 …… 45
3.4.4　文档的使用者 …… 45
3.4.5　文档编码 …… 46
3.4.6　文档格式 …… 47
3.5　软件配置管理 …… 48
3.5.1　软件配置概念 …… 48
3.5.2　软件配置规划 …… 49
3.5.3　软件变更控制 …… 50
3.5.4　软件版本控制 …… 51
3.6　软件质量管理 …… 52
3.6.1　质量标准 …… 52
3.6.2　质量计划 …… 52

3.6.3 质量保证 …… 53
小结 …… 54
习题 …… 55
第4章 计算机系统工程 …… 56
4.1 计算机体系结构 …… 56
4.1.1 系统特征 …… 56
4.1.2 体系结构 …… 57
4.2 软件系统高层分析 …… 60
4.2.1 分析内容 …… 60
4.2.2 分析建模 …… 60
4.3 项目可行性分析 …… 63
4.3.1 分析目的 …… 63
4.3.2 分析内容 …… 64
4.3.3 分析报告 …… 65
小结 …… 66
习题 …… 67
第5章 需求分析 …… 68
5.1 分析任务与过程 …… 68
5.1.1 需求问题 …… 68
5.1.2 分析任务 …… 69
5.1.3 任务承担者 …… 70
5.1.4 分析过程 …… 70
5.2 获取用户需求 …… 72
5.2.1 识别用户 …… 72
5.2.2 从调查中收集用户需求 …… 74
5.2.3 建立需求规约 …… 76
5.3 需求建模 …… 77
5.3.1 业务树图 …… 77
5.3.2 业务用例图 …… 78
5.3.3 业务活动图 …… 79
5.4 需求验证 …… 80
5.4.1 通过原型进行需求验证 …… 80
5.4.2 通过评审进行需求验证 …… 81
5.5 需求规格说明书 …… 81
小结 …… 82
习题 …… 83
第6章 结构化分析建模 …… 84
6.1 分析建模特点 …… 84
6.2 数据建模 …… 85
6.2.1 实体及实体间关系 …… 85
6.2.2 传统 ER 建模 …… 86
6.2.3 基于工具的 ER 建模 …… 86
6.2.4 建模举例 …… 87
6.3 功能建模 …… 89
6.3.1 数据流图(DFD) …… 90
6.3.2 数据流细化 …… 91
6.3.3 建模举例 …… 92
6.4 行为建模 …… 94
6.4.1 状态转换图(STD) …… 94
6.4.2 建模举例 …… 95
6.5 数据字典 …… 96
6.5.1 数据定义 …… 96
6.5.2 功能定义 …… 99
6.5.3 行为定义 …… 99
小结 …… 100
习题 …… 101
第7章 基于 UML 的面向对象分析建模 …… 104
7.1 UML 特点 …… 104
7.1.1 建模语言 …… 104
7.1.2 建模过程 …… 105
7.1.3 建模管理 …… 106
7.2 用例建模 …… 107
7.2.1 图形元素 …… 107
7.2.2 参与者关系 …… 108
7.2.3 用例关系 …… 108
7.2.4 建模举例 …… 110
7.3 活动建模 …… 112
7.3.1 图形元素 …… 112
7.3.2 业务级活动建模 …… 112
7.3.3 用例级活动建模 …… 113
7.4 类分析建模 …… 115
7.4.1 实体类 …… 115
7.4.2 实体类关系 …… 116
7.4.3 建模举例 …… 118
小结 …… 119
习题 …… 120
第8章 概要设计 …… 121
8.1 设计任务与过程 …… 121

8.1.1 设计任务 …… 121
8.1.2 设计过程 …… 123
**8.2 系统构架** …… 124
8.2.1 软件系统支持环境 …… 124
8.2.2 软件系统体系结构 …… 124
**8.3 数据结构** …… 125
8.3.1 程序数据结构 …… 125
8.3.2 数据库结构 …… 126
**8.4 程序结构** …… 128
8.4.1 程序模块 …… 128
8.4.2 模块独立性 …… 129
8.4.3 结构化程序结构 …… 134
8.4.4 面向对象程序结构 …… 135
**8.5 概要设计说明书** …… 137
小结 …… 138
习题 …… 139
第9章 结构化设计建模 …… 141
**9.1 建模语言** …… 141
9.1.1 程序结构图 …… 141
9.1.2 HIPO 图 …… 143
9.1.3 框架伪码 …… 144
**9.2 基于数据流的结构映射** …… 145
9.2.1 变换流映射 …… 145
9.2.2 事务流映射 …… 146
9.2.3 混合流映射 …… 148
**9.3 设计举例** …… 150
小结 …… 155
习题 …… 155
第10章 基于 UML 的面向对象设计建模 …… 157
**10.1 面向对象设计方法** …… 157
10.1.1 两种设计方法的比较 …… 157
10.1.2 UML 设计建模 …… 159
**10.2 逻辑结构设计** …… 161
10.2.1 系统构架 …… 161
10.2.2 类体元素 …… 162
10.2.3 设计类图 …… 166
**10.3 动态过程设计** …… 171
10.3.1 协作图 …… 171
10.3.2 时序图 …… 172
10.3.3 状态图 …… 175
**10.4 物理装配与部署** …… 176
10.4.1 程序构件图 …… 176
10.4.2 系统部署图 …… 178
小结 …… 180
习题 …… 181
第11章 用户界面设计 …… 183
**11.1 界面设计特点** …… 183
**11.2 界面类型** …… 184
11.2.1 窗体 …… 184
11.2.2 Web 页 …… 186
**11.3 界面功能** …… 187
11.3.1 信息表示 …… 187
11.3.2 系统交互 …… 188
11.3.3 联机支持 …… 188
**11.4 界面行为导航** …… 189
**11.5 其他界面问题** …… 191
小结 …… 192
习题 …… 192
第12章 算法设计与编码 …… 193
**12.1 结构化流程控制** …… 193
**12.2 算法设计工具** …… 194
12.2.1 程序流程图 …… 194
12.2.2 NS 图 …… 196
12.2.3 PAD 图 …… 197
12.2.4 PDL 语言 …… 198
**12.3 Jackson 设计方法** …… 198
12.3.1 设计步骤 …… 199
12.3.2 设计举例 …… 200
**12.4 程序编码** …… 203
12.4.1 编程语言 …… 203
12.4.2 编程规范 …… 205
小结 …… 207
习题 …… 208
第13章 软件测试 …… 210
**13.1 测试目的、计划与方法** …… 210
13.1.1 测试目的 …… 210
13.1.2 测试计划 …… 211

13.1.3 测试方法 …… 211
**13.2 测试任务** …… 212
13.2.1 单元测试 …… 212
13.2.2 集成测试 …… 214
13.2.3 确认测试 …… 216
**13.3 测试用例** …… 218
13.3.1 白盒测试用例设计 …… 218
13.3.2 黑盒测试用例设计 …… 220
**13.4 面向对象程序测试** …… 222
13.4.1 面向对象单元测试 …… 222
13.4.2 面向对象集成测试 …… 223
13.4.3 面向对象确认测试 …… 223
**13.5 程序调试** …… 223
13.5.1 诊断方法 …… 224
13.5.2 调试策略 …… 224
小结 …… 225
习题 …… 226
第14章 软件维护 …… 228
**14.1 软件维护分类** …… 228
**14.2 软件可维护性** …… 229
**14.3 软件维护实施** …… 230
14.3.1 维护机构 …… 230
14.3.2 维护过程 …… 231
**14.4 逆向工程与再工程** …… 233
小结 …… 234
习题 …… 234
参考文献 …… 235

# 第1章

# 软件工程概述

软件工程是工程化软件开发与维护的方法论。软件的开发者、维护者和软件项目管理者都将是软件工程的实践者，而且都需要掌握与应用软件工程方法。

本章要点：

- 软件的概念、特点与分类
- 软件工程的概念、技术、管理与目标
- 软件工程主流方法学

## 1.1 软　件

### 1.1.1 软件概念

软件是附着在计算机硬件设备上的逻辑要素，如指令、程序、数据。如果将计算机系统与人的大脑做比较，硬件就如同人脑的生理构造，软件则如同基于人脑形成的知识体系、观念意识、判断力。

在信息时代，软件是这个时代技术进步与发展的最关键因素。通过软件人们获得了更高精度的工程计算，有了对科学问题的虚拟仿真，实现了纳米级的材料工艺控制。软件在促进传统产业（如农业、制造业、零售业）的信息化转型，并通过信息化获得了跨区域的资源配置、成本控制与客户服务。实际上，软件已经成为人们工作、学习与生活中不可缺少的元素，已经影响到人类的生存方式，使人的世界观与价值观有了全新的发展。

需要指出的是，软件在不断地发展，因此在不同时期，人们对软件有不同的认识。20 世纪 50 年代的时候，软件在人们的印象中就是指令或程序控制。但是到了 20 世纪 60 年代，由于软件应用已涉及大量数据处理，并需要依靠一些外部存储设备（如磁带、磁盘）保存数据，因此数据从早期程序中分离出来，成为软件系统中需要专门考虑的新元素。20 世纪 70 年代以来，软件工程方法的应用使得软件文档，如软件规格说明书、软件设计说明书、软件操作手册等，变得越来越重

要,因此成为软件系统中必须考虑的又一种新的元素。

按照软件工程的要求,软件被定义为程序、数据、文档等诸多元素的集合,图1-1所示的软件组成即反映了这种软件定义。

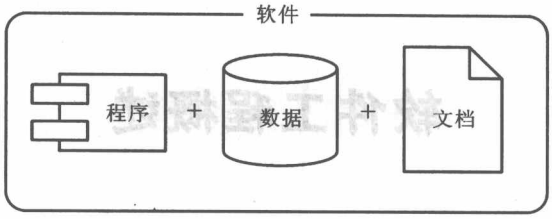



图1-1 软件组成

## 1.1.2 软件特点

### 1. 软件有对硬件不可缺失的依赖

计算机硬件是物理元素,因此具有一定的物理形态,能够独立存在。计算机软件则是逻辑元素,它是抽象的、无形的,因此不能独立存在,并对硬件有不可缺失的依赖。

显而易见的是,必须有磁盘、光盘、磁带这些有形的物理介质,抽象的逻辑软件才能存储;必须有CPU、内存的支持,抽象的逻辑软件才能进行物理运行;必须有键盘、鼠标、显示器、打印设备的支持,软件中的抽象数据才能有物理的输入、输出。

实际上,在任何时候,软件都离不开硬件的支持。

### 2. 软件有不同于硬件的生产流程

软件作为产品,也需要按工业化流程生产。然而,软件却有不同于硬件的生产流程。

实际上,软件不是制造出来的,而是开发出来的。通常,硬件生产的最大成本是复杂的制造工艺,而软件生产的最大成本是分析与设计。因此,针对产品质量监控,硬件主要体现为对产品制造工艺的流程监控,软件则主要体现为对产品的分析与设计的流程监控。

软件生产还有不同于硬件生产的资源需求。通常,硬件生产对原材料、能源、生产设备等有较高的依赖度,并难免为周围环境带来污染。软件生产则对原材料、能源、生产设备的依赖度很低,而是更加依赖于人的智慧,是高科技知识型产业,大多不会给周围环境带来污染。

### 3. 软件有不同于硬件的生命过程

硬件产品在使用过程中往往会经历磨合期、正常使用期、老化期三个阶段。

(1) 磨合期:硬件产品最初使用阶段。由于硬件产品中一些难以预料的缺陷,这个阶段往往会有一个相对较高的故障频率。

(2) 正常使用期:硬件磨合期的缺陷被修正后,将会经历的一个较长时间的稳定工作过程,故障频率相对较低。

（3）老化期：硬件产品寿命接近终点时的阶段。由于长久使用，硬件产品将出现过度磨损，因此故障频率会越来越高，并逐渐失效而不能工作。

图 1-2 所示为硬件产品的生命过程曲线。显然，高质量硬件产品应有较低的磨合期故障频率与较长的正常使用期。

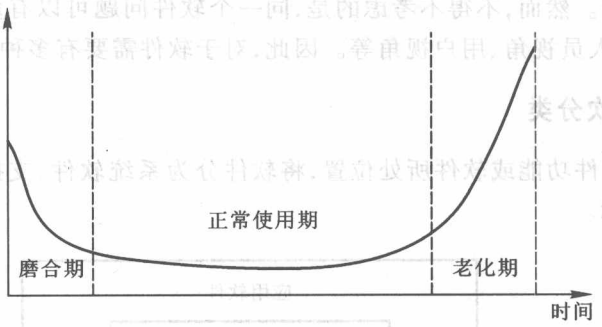



图 1-2　硬件产品生命过程曲线

软件产品也有类似于硬件产品的磨合期与正常使用期。软件磨合期的高故障频率通常由软件中隐藏的错误或缺陷所致，并随着这些错误的排除而进入正常使用期。但软件是用不坏的，故不会出现磨损现象。实际上，只要软件的工作环境没有变化，并且软件本身不做改动，则在其进入正常使用期后，可以一直使用下去。

但是，软件可能会遇到操作系统、数据库引擎等环境因素的变化。另外，由于业务发展，软件用户有可能会对软件提出新要求，如增加新功能、完善原有功能、改善工作性能、人性化交互界面。因此，软件在使用过程中可能需要更新和进化，以使软件能够有更长的使用寿命。然而，每一次因运行环境或用户需求导致的软件进化，都会给软件带来新的错误或缺陷，由此使软件需重新经历磨合期，并因此开始新的正常使用期。

图 1-3 所示为软件产品的生命过程曲线。显然，如同硬件一样，高质量软件产品也应该有较低的磨合期故障频率与较长的正常使用期。

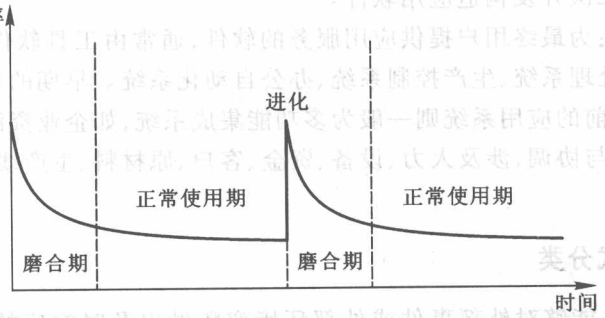



图 1-3　软件产品生命过程曲线

## 1.1.3 软件分类

开发的软件不同，则开发软件的过程、技术与方法都会不同。显然，为更好地认识软件，有必要对软件进行分类。然而，不得不考虑的是，同一个软件问题可以有多个不同的研究视角，如技术人员视角、管理人员视角、用户视角等。因此，对于软件需要有多种不同方式的分类。

### 1. 按功能层次分类

通常可根据软件功能或软件所处位置，将软件分为系统软件、支撑软件与应用软件，其一般关系如图1-4所示。

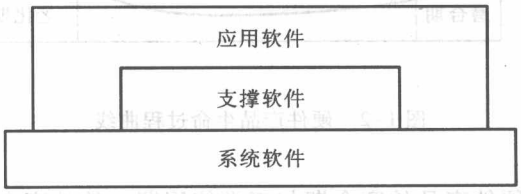



图1-4 系统软件、支撑软件与应用软件之间的关系

(1) 系统软件：为计算机底层软件，如操作系统、设备驱动程序、数据库引擎等。系统软件的特点是，与计算机硬件紧密相关，并通过与硬件的频繁交互，对其他软件的运行提供底层服务。例如，操作系统建立在硬件环境的基础上，并通过文件服务、进程服务、存储服务给其他软件提供更加适宜的运行环境。又例如，某种设备的驱动程序用于支持该设备与计算机主机的连接。

(2) 支撑软件：介于系统软件与应用软件之间的支持软件开发的软件。最常见的支撑软件是支持软件开发与维护的工具软件，如程序编译器、程序编辑器、错误检测程序、程序资源库等。显然，这些软件为高质、高效开发软件提供了便利。基于组件技术产生的中间件软件也是支撑软件。中间件种类很多，根据用途不同可分为数据中间件、对象中间件、通信中间件。可通过中间件的嵌入、组合或二次开发构造应用软件。

(3) 应用软件：为最终用户提供应用服务的软件，通常由工具软件开发，并依靠系统软件的支持运行，如财务处理系统、生产控制系统、办公自动化系统。早期的应用软件大多为离散结构，功能相互独立。目前的应用系统则一般为多功能集成系统，如企业资源管理系统，即集成企业中的各种资源的管理与协调，涉及人力、设备、资金、客户、原材料、生产线、半成品、产品等诸多方面的管理与控制。

### 2. 按工作方式分类

(1) 实时软件：能够对外部事件或外部环境变化做出及时响应的软件，如飞机自动导航程序、导弹目标跟踪程序、自动生产线监控程序。实时软件一般由数据收集器、数据分析器、响应控制器等诸多部件组成，如图1-5所示。通常，数据收集器负责从外部环境获取格式化信息，数据分析器负责对获取的格式化信息进行分析、判断，响应控制器则负责按数据分析器对外部事件的

判断进行有效的行为应对。以导弹目标跟踪程序为例，当导弹跟踪目标发生移动时，导弹可通过热感应方式获取移动信息，并通过数据收集器将移动信息格式化，然后通过数据分析器的分析、判断确定目标新的位置，再由响应控制器调整运行轨迹，以达到有效跟踪的目的。

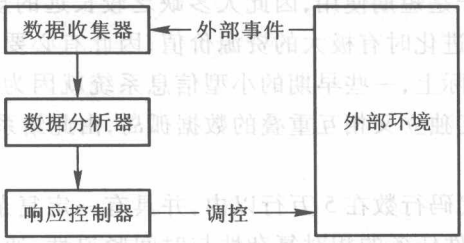



图 1-5　实时软件的部件组成

(2) 分时软件：能够把一个 CPU 工作时间段切片分配给多项任务的软件。例如，Windows 操作系统就是一个具有分时功能的多任务操作系统，能使多个程序同步运行。与分时软件相关的软件技术是多线程机制，即在一个程序进程内执行多个线程任务，C++、Java 等编程语言都支持多线程机制。

(3) 交互式软件：用于实现人机对话的软件，大多具有图形界面，如窗体程序、Web 页面。交互式软件的工作空间一般在客户端。窗体程序的特点是无论安装或是运行都在客户端。Web 页面则以 HTML、XML 等文档格式安装于 Web 服务器，但运行空间在客户端，图 1-6 所示即为 Web 页面的工作特征。交互式软件的作用是方便人对计算机的操作，因此开发交互式软件时需要考虑与人相关的多方面因素，如行为学、心理学、美学等。

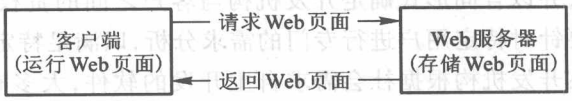



图 1-6　Web 页面的工作特征

(4) 嵌入式软件：专门为特定智能设备提供自动控制的软件，如洗衣机流程控制程序、微波炉流程控制程序、汽车定速导航程序。嵌入式软件一般驻留在智能设备拥有的只读存储器中，因此与智能设备融为一体。

(5) 批处理软件：能够成批处理大量数据或成批处理多项事务的软件。批处理软件通常需要面对大量数据，并往往以某个时间点(如年终、月末)为任务触发事件。批处理方式能够用来提高系统工作性能，如将可能影响系统日常性能的大量数据的处理安排在机器相对空闲时进行成批处理。批处理方式还能使事务处理延后，这将带来事务处理的灵活性，如一些涉及多方协作而不便于同步处理的事务，即可通过批处理方式而进行异步处理。

### 3. 按规模分类

(1) 微型软件：源程序代码行数在 500 行以内的程序，通常一个人几天内即可完成开发。微

型软件大多任务单一、操作简单。

(2) 小型软件:源程序代码行数在2 000行以内的程序,通常一个人半年之内即可完成开发。小型软件大多为用户单位自主开发,以满足其短期的、局部的业务应用需要。小型软件的程序结构一般比较简单,并由于是短期使用,因此大多缺乏较长远的规划。但是,小型软件运行中产生的数据可能在今后系统进化时有极大的资源价值,因此有必要对系统中的数据环境进行超越当前需要的整体规划。实际上,一些早期的小型信息系统就因为缺乏对数据环境的必要的整体规划,以致出现许多既相互独立又相互重叠的数据孤岛,由此给系统日后的进化与集成带来很大的困难。

(3) 中型软件:源程序代码行数在5万行以内,并具有一定复杂度的软件系统,大多要求一年内完成开发。由于中型软件任务的相对复杂性与时间紧迫性,通常需要组织项目小组进行开发,并需要采用一整套的软件工程方法进行任务约束,如制订项目计划、进行软件分析与设计、实施软件配置管理。

(4) 大型软件:源程序代码行数在5万行以上,具有综合功能,并涉及多用户任务协作的相当复杂的软件系统。大型系统一般有跨年度的开发周期,为了方便软件任务展开,有可能涉及多个项目小组的合作,需要实施二级管理。通常的做法是将软件系统按业务或功能划分成诸多相对独立的子系统,然后以子系统为单位将开发任务分配到各个项目小组。面对复杂的大规模软件系统,开发中还要有比较完善的软件工程方法的支持。

### 4. 按服务对象分类

(1) 用户定制软件:开发机构受特定客户委托开发的软件,如某特殊设备的控制系统、某企业的业务管理系统、某特定大厦的智能监控系统、某城市的交通监管系统。用户定制软件大多以招标方式获得开发任务,并以合同形式确定开发机构与客户之间的责权关系。由于定制软件由特定用户独享,因此需要针对特定用户进行专门的需求分析,以满足特定用户的特殊应用需要。

(2) 通用商业软件:开发机构根据社会需求自主开发的软件,大多有广泛的用户群,如通用办公系统、通用财务系统。一些具有通用特征的中间件也可看做通用商业软件,如生成报表中间件、统计分析中间件。为满足通用性,通用商业软件往往需要有面向用户的应用配置,以使软件能适应各种不同的工作环境和满足各种不同的应用需要。通用商业软件一般比用户定制软件有更高的技术要求,因此开发周期更长,开发费用更高。通用商业软件往往需要进行比较全面的市场调研,如调查市场中是否已有同类型软件,假如已有同类型软件,则考虑自己开发的软件是否具有一定优势,如价格优势、功能优势或性能优势。通用商业软件需要有一定的市场渠道实现应用推广,通常的市场渠道有专卖店销售、捆绑于硬件设备销售、网上销售。

## 1.2 软件工程

软件工程是关于软件开发、使用与维护的工程方法学,是一门涉及工程技术、工程管理与工程经济等诸多内容的综合性工程学科。软件工程建立在与软件有关的工程概念、原理与方法的基础上,它是对现实软件问题的工程方法探索,具有鲜明的工程实用性。

软件的产业化发展需要有软件工程方法的支持。实际上,自从软件成为工业化产品,它所面