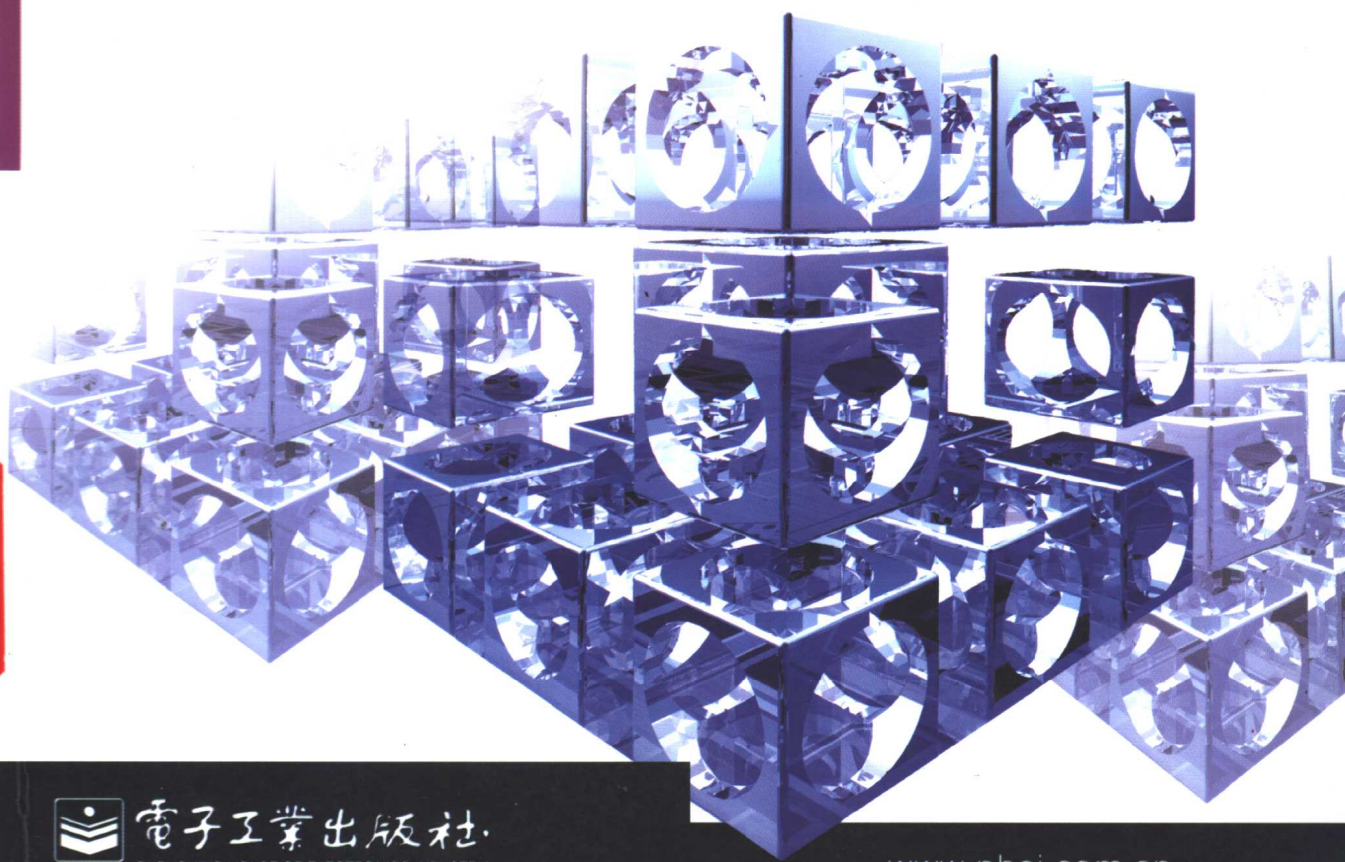
电子·教育

高职高专计算机系列教材

中国计算机学会高职高专教育学组推荐出版

软件工程基础

李成大　张　京
郑显举　许　珏　编著



電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
www.phei.com.cn

内 容 简 介

本书主要介绍软件工程及其应用的有关内容，包括可行性研究、需求分析、总体设计、详细设计、编码、测试、维护以及有关软件管理、软件开发工具和环境等方面的内容。为了保持教材内容的先进性，本书还介绍了面向对象软件工程学、统一建模语言 UML、软件工程标准与软件文档等方面的内容。本书内容新颖，实例丰富，各章均有小结与习题，便于教学和自学。

本书可作为高等院校“软件工程”课程的教材或教学参考书，也可供从事软件开发与应用的工程技术人员和管理人员阅读参考。

图书在版编目(CIP)数据

软件工程基础/李成大等编著 .—北京：电子工业出版社，2003.1
高职高专计算机系列教材
ISBN 7-5053-8194-6

Ⅰ.软… Ⅱ.李… Ⅲ.软件工程—高等学校：技术学校—教材 Ⅳ.TP311.5

中国版本图书馆 CIP 数据核字(2002)第 093608 号

责任编辑：程超群
印　　刷：北京李史山胶印厂
出版发行：电子工业出版社　http://www.phei.com.cn
　　北京市海淀区万寿路 173 信箱　邮编 100036
经　　销：各地新华书店
开　　本：787×1 092　1/16　印张：13.75　字数：352 千字
版　　次：2003 年 1 月第 1 版　2003 年 1 月第 1 次印刷
印　　数：7 000 册　定价：17.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。
联系电话：(010)68279077

出 版 说 明

高职高专的计算机专业面临着两方面的巨大变化，一是计算机技术的飞速发展，另一方面是高职高专教育本身的改革和重组。

当前，计算机技术正经历着高速度、多媒体网络化的发展，计算机教育特别是计算机专业的教材建设必须适应这种日新月异的形势，才能培养出不同层次的合格的计算机技术专业人才。为了适应这种变化，国内外都在对计算机教育进行深入的研究和改革。美国 IEEE 和 ACM 在推出了《Computing Curricula 2000》之后，立即又推出了《Computing Curricula 2001》。全国高校计算机专业教学指导委员会和中国计算机学会教育委员会在 1999 年 9 月也提出了高等院校《计算机学科教学计划 2000》（征求意见稿）。目前，国内许多院校老师、专家正在研究《Computing Curricula 2001》，着手 21 世纪的中国计算机教育的改革。

高专层次和本科层次的计算机教育既有联系又有区别，高专层次的计算机教育旨在培养应用型人才。自 20 世纪 70 年代末高等专科学校计算机专业相继成立以来，高等专科学校积极探索具有自己特色的教学计划和配套教材。1985 年，在原电子工业部的支持下，由全国数十所高等专科学校参加成立了中国计算机学会教育委员会大专教育学组，之后又成立了大专计算机教材编委会。从 1986 年到 1999 年，在各校老师的共同努力下，已相继完成了三轮高等专科计算机教材的规划与出版工作，共出版了 78 种必修课、选修课、实验课教材，较好地解决了高专层次计算机专业的教材需求。

为了适应计算机技术的飞速发展以及高职高专计算机教育形势发展的需要，中国计算机学会教育委员会高职高专教育学组和高职高专计算机教材编委会于 2000 年 7 月开始，又组织了一批本科高校、高等专科学校、高等职业技术院校和成人教育高等院校的有教学经验的老师，学习研究参考了高等院校《计算机学科教学计划 2000》（征求意见稿），提出了按照新的计算机教育计划和教学改革的要求，编写高专、高职、成人高等教育三教统筹的第四轮教材。

第四轮教材的编写工作采取了以招标的方式征求每门课程的编写大纲和主编，要求投标老师详细说明课程改革的思路、本课程和相关课程的联系、重点和难点的处理等。在第四轮教材的编写过程中，编委会强调加强实践环节、强调三教统筹、强调理论够用为度的原则，要求教学计划、教学内容适应高等教育发展的新形势。本套教材的编者均为各院校具有丰富教学实践经验的教师。因此，第四轮教材的特点是体系结构比较合理、内容新颖、概念清晰、通俗易懂、理论联系实际、实用性强。

竭诚希望广大师生对本套教材提出批评建议。

中国计算机学会教育委员会高职高专教育学组

2001 年 1 月

先后参加中国计算机学会教育委员会高职高专教育学组和高职高专计算机教材编委会学术活动的部分学校名单

太原理工大学阳泉学院
山西师范大学成人教育学院
承德石油高等专科学校
河北大学城市学院
保定职业技术学院
北京科技大学职业技术学院
北京工商大学应用技术学院
北京市机械工业管理局职工大学
北方工业大学
北京船舶工业管理干部学院
海淀走读大学
北京信息职业技术学院
北京信息工程学院
中国人民大学成人高等教育学院
沈阳电力高等专科学校
辽宁交通高等专科学校
吉林大学工程技术学院
吉林交通职业技术学院
吉林职业师范学院
哈尔滨学院
海南职业技术学院
海口经济职业技术学院
上海理工大学职业技术学校
上海第二工业大学
上海交通大学技术学院
上海商业职业技术学院
上海电机技术高等专科学校
上海旅游高等专科学校
上海应用技术学院
金陵联业大学
钟山职业技术学院
南京工程学院
南京师范大学
无锡职业技术学院
苏州市职工大学
空军后勤学院
连云港化工高等专科学校
淮南联合大学
滁州职业技术学院
兖州矿区职工大学
青岛职业技术学院
云南财贸学院
西安电子科技大学高等职业技术学院
陕西工业职业技术学院
兰州石化职业技术学院
兰州师范高等专科学校
重庆电子职业技术学院
重庆工业职业技术学院
天津职业技术师范学院
天津职业大学
天津轻工业学院
浙江大学
浙江工贸职业技术学院
宁波高等专科学校
湖州职业技术学院
福州大学职业技术学院
湖南大学
湖南计算机高等专科学校
湖南城市学院
中国保险管理干部学院
湖南税务高等专科学校
湖南民政职业技术学院
长沙大学
湖南财经高等专科学校
邵阳高等专科学校
湖南环境生物职业技术学院
湖南建材高等专科学校
襄樊职业技术学院
江汉大学
鄂州职业大学
武汉职业技术学院
河南机电高等专科学校
河南职业技术学院
郑州工业高等专科学校
平原大学
济源职业技术学院
郑州经济管理干部学院
中州大学
洛阳大学
漯河职业技术学院
广东女子职业技术学院
广州市财贸管理干部学院
广东轻工职业技术学院
广州航海高等专科学校
韶关大学
广西职业技术学院
南宁职业技术学院
广西水利电力职业技术学院
柳州职业技术学院
成都信息工程学院
成都电子机械高等专科学校
电子科技大学
成都航空职业技术学院
成都师范高等专科学校
四川托普信息技术学院
四川师范学院

前　言

软件工程是指导计算机软件开发与维护的工程学科，它采用工程的概念、原理、技术和方法来开发和维护软件，把经过时间检验证明是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，以便经济地开发出高质量的软件并有效地维护它。从20世纪60年代末提出“软件工程”概念以来，历经三十多年的飞速发展，软件工程逐渐成熟，现已成为计算机科学与技术领域中的一门重要学科。

随着计算机的日益普及，计算机软件已无处不在。以软件的说明、开发、维护和管理为内容，作为信息产业的一个支柱，软件工程这一学科已逐渐为人们所熟悉和广泛应用。现在大家都认识到，如果有哪个项目不遵循软件工程原则，必定会受到实践的惩罚。因此，认真学习并在实际工作中正确地运用软件工程，是摆在我们面前的一项十分迫切的任务。

软件工程是一门研究范围非常广泛的迅速发展的新兴学科，学科内的新技术、新方法不断涌现。本书着重从实用角度讲解软件工程的基本概念、基本原理和技术方法，同时也注意了该书的系统型和先进性。希望本书既能对实际的软件开发人员和管理人员有所帮助，又能为读者深入研究这门学科奠定较好的基础。

本书共有13章。第1章介绍了软件的概念、发展和软件危机，着重介绍了软件生存期、软件开发模型及软件工程的基本概念和基本内容。第2、3、4、5、6、9、10章是本书的重点，分别论述可行性研究、需求分析、总体设计、详细设计、编码、测试、维护阶段的各种方法和技术，对SA方法、SD方法、数据流图、数据字典、层次图、HIPO图、结构图、N-S图、PAD图、PDL语言、黑盒法、白盒法等逐一做了详细的介绍，读者开发软件时可根据需要灵活运用。为了保持教材内容的先进性，本书第7、8章介绍了面向对象软件工程学和统一建模语言UML，主要包括面向对象方法的基本概念、面向对象的分析、面向对象的设计、软件复用、UML的静态建模机制、UML的动态建模机制、UML软件开发过程等内容。第11章介绍了有关软件项目计划、软件项目组织、软件项目人员配备、软件项目的指导和检验、软件配置管理和配置管理工具等软件管理方面的内容。第12章介绍了软件开发工具和环境，并对计算机辅助软件工程CASE做了简要的介绍。第13章介绍了软件工程标准与软件文档，包括ISO 9000质量标准、ISO/IEC 12207软件生存周期过程标准、能力成熟度模型CMM等内容。在附录中给出了计算机软件开发文档编制指南，供实际应用时参考。

本书内容新颖，实例丰富，语言文字通俗易懂；各章重点、难点突出，原理、技术和方法的阐述融于丰富的实例之中；各章均有小结与习题，便于教学和自学。本书可作为高等院校“软件工程”课程的教材或教学参考书，也可供从事软件开发与应用的工程技术人员和管理人员阅读参考。

本书的第7、8、12、13章由李成大副教授编写，第1、2、3、4、5、11章由张京副教授编写，第9、10章由郑显举老师编写，第6章及附录由许珏老师编写，李成大老师负责统稿全书。

西南交通大学计算机与通信工程学院文登敏副教授仔细审阅了本书，并提出了非常宝贵的意见，特此表示深深的谢意。

由于编者水平有限，编写时间仓促，书中错误之处在所难免，恳请专家和读者批评指正。

编　者

2002年10月

目　录

第 1 章　软件工程概述 …… (1)
1.1　软件的概念、特点和分类 …… (1)
1.1.1　软件的概念 …… (1)
1.1.2　软件的特点 …… (1)
1.1.3　软件的分类 …… (2)
1.2　软件的发展和软件危机 …… (2)
1.2.1　计算机系统的发展历程 …… (2)
1.2.2　软件危机 …… (3)
1.3　软件工程 …… (4)
1.3.1　软件工程的定义 …… (4)
1.3.2　软件工程方法学 …… (4)
1.4　软件生存期和软件开发模型 …… (6)
1.4.1　软件生存期 …… (6)
1.4.2　软件开发模型 …… (7)
小结 …… (10)
习题 1 …… (11)
第 2 章　可行性研究 …… (12)
2.1　问题定义 …… (12)
2.1.1　问题定义的内容 …… (12)
2.1.2　问题定义的步骤 …… (12)
2.2　可行性研究的任务 …… (12)
2.3　可行性研究的步骤 …… (13)
2.4　系统流程图 …… (14)
2.5　成本/效益分析 …… (15)
2.5.1　成本估计 …… (16)
2.5.2　度量效益的方法 …… (16)
小结 …… (18)
习题 2 …… (18)
第 3 章　需求分析 …… (19)
3.1　需求分析的任务 …… (19)
3.2　需求分析的过程 …… (21)
3.3　需求分析的原则 …… (23)
3.4　结构化分析方法 …… (24)
3.4.1　数据流图 …… (24)
3.4.2　数据字典 …… (26)

3.4.3 加工逻辑描述工具 …… (28)
3.5 原型化方法 …… (30)
3.5.1 软件原型的分类 …… (30)
3.5.2 快速原型开发模型 …… (31)
小结 …… (33)
习题 3 …… (34)
第 4 章 总体设计 …… (35)
4.1 总体设计的过程 …… (35)
4.2 总体设计的图形工具 …… (37)
4.2.1 层次图 …… (37)
4.2.2 HIPO 图 …… (37)
4.2.3 结构图 …… (38)
4.3 软件设计的概念和原理 …… (39)
4.3.1 模块化设计 …… (40)
4.3.2 自顶向下逐步细化 …… (43)
4.3.3 启发式规则 …… (44)
4.4 面向数据流的设计方法 …… (46)
4.4.1 基本概念 …… (46)
4.4.2 SD 方法概述 …… (47)
4.4.3 SD 方法的步骤 …… (48)
4.4.4 设计优化 …… (51)
小结 …… (52)
习题 4 …… (52)
第 5 章 详细设计 …… (54)
5.1 详细设计的任务和原则 …… (54)
5.1.1 详细设计的任务 …… (54)
5.1.2 详细设计的原则 …… (55)
5.2 结构程序设计 …… (55)
5.3 详细设计的工具 …… (56)
5.3.1 程序流程图 …… (57)
5.3.2 N-S 图 …… (58)
5.3.3 PAD 图 …… (59)
5.3.4 PDL 语言 …… (60)
5.3.5 详细设计工具的选择 …… (60)
小结 …… (61)
习题 5 …… (61)
第 6 章 编码 …… (62)
6.1 程序设计语言 …… (62)
6.1.1 程序设计语言的分类 …… (62)
6.1.2 程序设计语言的特点 …… (63)

6.1.3 程序设计语言的选择 …… (65)
6.2 编码风格 …… (66)
6.2.1 代码文档化 …… (67)
6.2.2 数据说明 …… (68)
6.2.3 语句构造 …… (68)
6.2.4 输入/输出 …… (68)
6.3 程序效率 …… (69)
6.3.1 代码效率 …… (69)
6.3.2 存储器效率 …… (70)
6.3.3 输入/输出的效率 …… (70)
小结 …… (70)
习题 6 …… (71)
第 7 章 面向对象的分析和设计方法 …… (72)
7.1 面向对象方法的基本概念 …… (72)
7.1.1 面向对象方法概述 …… (72)
7.1.2 面向对象的概念 …… (73)
7.1.3 面向对象方法的主要优点 …… (76)
7.2 面向对象的分析 …… (79)
7.2.1 面向对象分析的基本过程 …… (79)
7.2.2 确定对象和类 …… (84)
7.2.3 确定属性 …… (87)
7.2.4 定义服务 …… (87)
7.2.5 对象间通信 …… (91)
7.3 面向对象的设计 …… (97)
7.3.1 面向对象设计的基本概念 …… (98)
7.3.2 面向对象设计的方法 …… (99)
7.4 软件复用 …… (103)
7.4.1 软件复用的概念 …… (103)
7.4.2 软件复用的效果 …… (103)
7.4.3 软件复用技术 …… (104)
7.4.4 面向对象方法与软件复用的关系 …… (104)
小结 …… (106)
习题 7 …… (107)
第 8 章 统一建模语言 UML …… (108)
8.1 UML 简介 …… (108)
8.1.1 UML 的由来 …… (108)
8.1.2 UML 的内容 …… (109)
8.1.3 UML 的主要特点 …… (110)
8.1.4 UML 的应用领域 …… (111)
8.2 UML 模型的基本概念 …… (111)

8.2.1 建模技术 …… (112)
8.2.2 标准建模语言 UML 建模框架 …… (112)
8.2.3 UML 模型的基本概念 …… (113)
8.3 UML 的静态建模机制 …… (115)
8.3.1 用例图 …… (115)
8.3.2 类图、对象图和包 …… (118)
8.3.3 构件图和配置图 …… (123)
8.4 UML 的动态建模机制 …… (124)
8.4.1 消息 …… (124)
8.4.2 状态图 …… (125)
8.4.3 顺序图 …… (125)
8.4.4 合作图 …… (125)
8.4.5 活动图 …… (126)
8.4.6 四种图的运用 …… (126)
8.5 UML 软件开发过程概述 …… (126)
8.5.1 UML 建模过程高层视图 …… (127)
8.5.2 UML 实际建模过程 …… (127)
小结 …… (128)
习题 8 …… (128)
第 9 章 软件测试 …… (129)
9.1 基本概念 …… (129)
9.1.1 软件测试的目标 …… (129)
9.1.2 软件测试的方法与技术 …… (129)
9.1.3 软件测试的步骤 …… (131)
9.2 测试用例的设计 …… (132)
9.2.1 黑盒测试法 …… (132)
9.2.2 白盒测试法 …… (135)
9.3 单元测试 …… (139)
9.4 集成测试 …… (141)
9.5 验收测试 …… (142)
9.6 系统测试 …… (143)
9.7 面向对象的软件测试 …… (144)
9.7.1 OOA 和 OOD 模型的测试 …… (144)
9.7.2 面向对象的测试策略 …… (145)
小结 …… (145)
习题 9 …… (146)
第 10 章 软件维护 …… (147)
10.1 维护的种类 …… (147)
10.2 可维护性 …… (147)
10.2.1 决定可维护性的因素 …… (147)

10.2.2 文档 …… (148)
10.2.3 可维护性复审 …… (148)
10.3 维护工作的步骤 …… (149)
10.4 维护的副作用 …… (151)
10.5 维护的管理 …… (152)
10.6 逆向工程和再生工程 …… (153)
小结 …… (155)
习题 10 …… (155)
第 11 章 软件管理 …… (156)
11.1 软件项目的特点和软件管理的职能 …… (156)
11.1.1 软件项目的特点 …… (156)
11.1.2 造成软件项目失误的原因 …… (157)
11.1.3 软件管理的职能 …… (157)
11.2 软件项目计划 …… (157)
11.2.1 制定计划的目标和进行风险分析 …… (158)
11.2.2 软件计划的类型 …… (158)
11.2.3 项目计划中任务的划分 …… (158)
11.3 软件项目组织 …… (159)
11.3.1 组织原则 …… (159)
11.3.2 组织结构的模式 …… (159)
11.3.3 程序设计小组的组织 …… (160)
11.4 软件项目人员配备 …… (161)
11.4.1 项目开发各阶段所需人员 …… (161)
11.4.2 配备人员的原则 …… (162)
11.4.3 对项目经理人员的要求 …… (162)
11.4.4 评价软件人员的条件 …… (162)
11.5 软件项目的指导和检验 …… (163)
11.5.1 软件项目指导 …… (163)
11.5.2 软件项目检验 …… (163)
11.6 软件配置管理和配置管理工具 …… (164)
11.6.1 概述 …… (164)
11.6.2 基线（baseline） …… (164)
11.6.3 软件配置项 …… (165)
11.6.4 软件配置管理的过程 …… (166)
11.6.5 配置管理工具 ClearCase 简介 …… (167)
小结 …… (169)
习题 11 …… (169)
第 12 章 软件开发工具和环境 …… (170)
12.1 软件开发工具 …… (170)
12.2 软件开发环境 …… (171)

12.2.1 按解决的问题分类 …… (171)
12.2.2 按现有软件开发环境的演变趋向分类 …… (171)
12.2.3 按集成化程度分类 …… (172)
12.3 软件开发工具和环境的应用及发展 …… (173)
12.4 计算机辅助软件工程 CASE …… (175)
12.4.1 CASE 工具 …… (175)
12.4.2 软件自动化 …… (175)
12.4.3 CASE 的作用 …… (175)
12.4.4 CASE 工具实例 …… (176)
小结 …… (176)
习题 12 …… (176)
第 13 章 软件工程标准与软件文档 …… (177)
13.1 软件工程标准化 …… (177)
13.1.1 什么是软件工程标准 …… (177)
13.1.2 软件工程标准化的作用 …… (177)
13.1.3 软件工程标准的级别分类 …… (177)
13.2 ISO 9000 质量标准 …… (179)
13.2.1 基本思想 …… (179)
13.2.2 ISO 9000-3 标准 …… (179)
13.3 ISO/IEC 12207 软件生存周期过程标准 …… (182)
13.3.1 标准制定的目的和适用范围 …… (182)
13.3.2 标准的基本内容 …… (182)
13.4 软件文档 …… (185)
13.4.1 软件文档的作用和分类 …… (185)
13.4.2 文档的管理和维护 …… (187)
13.5 能力成熟度模型 CMM …… (188)
13.5.1 能力成熟度模型的结构 …… (188)
13.5.2 能力成熟度等级 …… (189)
13.5.3 关键过程域 …… (191)
13.5.4 应用 CMM …… (192)
小结 …… (192)
习题 13 …… (193)
附录 计算机软件开发文档编制指南 …… (194)
参考文献 …… (208)

第 1 章　软件工程概述

1.1　软件的概念、特点和分类

1.1.1　软件的概念

软件是软件工程学中的一个重要概念，任何一种计算机系统都包含硬件（Hardware）和软件（Software）两大部分。Software 一词，有人译为“软制品”，也有人译为“软体”，现在统一称为“软件”。许多人认为软件就是程序，那么，软件究竟是不是程序呢？

软件的定义如下：软件是计算机系统中与硬件相互依存的另一部分，它是包括程序、数据及其相关文档的完整集合。其中，程序是按事先设计的功能和性能要求编写的指令序列；数据是使程序能正常操纵信息的数据结构；文档是与程序开发、维护和使用有关的图文材料。

从软件的概念可以看出，程序并不是软件，程序只是软件的组成部分。

1.1.2　软件的特点

为了深入理解软件工程，探讨软件的特点是非常重要的。通过对软件特点的介绍，能使读者更好地理解计算机软件，能更充分地认识到软件工程的重要性。软件的特点可归纳如下：

（1）软件是一种逻辑实体。人们可以把它记录在介质上，但却无法看到软件的形态，而必须通过测试、分析、思考、判断去了解它的功能、性能及其他特性。软件正确与否，是好是坏，一直要到程序在机器上运行后才能知道，这就给软件的设计、生产和管理带来许多困难。

（2）软件开发是人类智力的高度发挥，而不是传统意义上的硬件制造。在软件的开发过程中没有明显的制造过程。软件是通过人们的智力活动，把知识与技术转化成信息的一种产品。所以，要实现对软件的质量控制，必须着重在软件开发方面下功夫。

（3）软件维护与硬件维修有着本质的差别。在软件的生存期中，为了使软件能够克服以前没有发现的故障，能够适应硬件、软件环境的变化以及用户新的要求，必须修改软件，而每次修改都可能会引入新的错误。这样反复修改软件，必然导致软件失效率升高。

（4）软件的开发和运行常常受到计算机系统的限制，对计算机系统有着不同程度的依赖性。为了解除这种依赖性，在软件开发中提出了软件移植的问题，并且把软件的可移植性作为衡量软件质量的因素之一。

（5）软件的开发至今尚未完全摆脱手工艺的开发方式，使软件的开发效率受到很大限制。因此，应加快软件技术的发展，提出和采用新的软件开发方法。例如，可利用软件复用技术或软件自动生成技术，使用一些有效的软件开发工具或软件开发环境，以提高软件开发效率。

（6）软件的开发是一个复杂的过程。有人认为，人类能够创造的最复杂的事物就是计

算机软件。软件的复杂可能来自它所反映的实际问题的复杂性，也可能来自程序逻辑结构的复杂性。

（7）软件的成本非常高昂。软件的研制工作需要投入大量的、复杂的、高强度的脑力劳动，所以开发费用非常高，导致软件的成本相当昂贵。

1.1.3 软件的分类

人们在学习、工作和生活中会接触到各种各样的软件。那么究竟有哪些类型的软件呢？虽然找不到一个统一的严格分类标准，但可以从不同角度来对软件进行分类。

1. 基于软件功能的划分

（1）系统软件（System software）：是服务于其他程序的程序集，一般由计算机生产厂家配置，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。如果没有系统软件的支持，计算机将难以发挥其功能，甚至无法工作。

（2）应用软件（Application software）：是在系统软件的基础上，为解决特定领域应用而开发的软件。按其性质不同可以分为商业处理软件、科学计算软件、计算机辅助设计软件、人工智能软件等。

（3）支撑软件：是系统软件和应用软件之间的支持软件，一般用来辅助和支持开发人员开发和维护应用软件，以提高软件的开发质量和生产率。常用的支撑软件包括需求分析工具、设计工具、编码工具、测试工具、维护工具和管理工具等。

2. 基于软件工作方式的划分

（1）实时处理软件：指在事件或数据产生时，立即处理并及时反馈信号，控制那些需要监测和控制的过程的软件。主要包括数据采集、分析、输出三部分，其处理时间是应严格限定的，如果在任何时间超出了这一限制，都将造成事故。

（2）分时软件：允许多个联机用户同时使用计算机的软件。系统把处理机时间轮流分配给各联机用户，使各用户都感到只有自己在使用计算机的软件。

（3）交互式软件：能实现人机通信的软件。这类软件接收用户给出的信息，但在时间上没有严格的限定，这种工作方式给予用户很大的灵活性。

（4）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，按顺序逐个处理的软件。

除此以外，软件还有其他的一些分类的方法。按软件规模进行划分可分为小型软件、中型软件、大型软件、甚大型软件和极大型软件；按软件服务对象的范围可划分为定制软件和产品软件等。

1.2 软件的发展和软件危机

1.2.1 计算机系统的发展历程

迄今为止，计算机系统经历了四个不同的发展阶段。下面简要地介绍各个发展阶段的特点，使读者了解软件危机的产生和软件工程学诞生的历史背景。

20 世纪 60 年代中期以前，是计算机系统发展的早期时代，此时的软件发展为程序设计阶段。在这个时期通用硬件已经相当普遍，软件却是为每个具体应用而专门编写的，这时的软件实际上就是规模较小的程序，程序的编写者和使用者往往是同一个（或同一组）人。这种个体化的软件环境，使得软件设计往往只是在人们头脑中隐含进行的一个模糊过程，除了程序清单之外，根本没有其他文档资料保存下来。

从 20 世纪 60 年代中期到 20 世纪 70 年代中期，是计算机系统发展的第二代，此时的软件发展为程序系统阶段。在这 10 年中，计算机技术有了很大的进步，多道程序、多用户系统、实时系统、在线存储技术是这一时期的主要特征。在这一时期出现了“软件作坊”，但“软件作坊”基本上沿用了早期形成的个体化软件开发方法。随着软件需求数量的急剧膨胀，软件维护工作的急剧增长，“软件危机”就这样开始出现了。1968 年，北大西洋公约组织的计算机科学家在联邦德国召开国际会议，讨论软件危机问题，在这次会议上正式提出并使用了“软件工程”这个名词，一门新兴的工程学科就此诞生了。

计算机系统发展的第三代是从 20 世纪 70 年代中期到 20 世纪 80 年代中期，此时的软件发展为软件工程阶段。这个时期的主要特点是微处理器出现并得到了广泛的应用，分布式系统极大地增加了计算机系统的复杂性，局域网、广域网和宽带数字通信对软件开发者提出了更高的要求。但是，这个时期的软件仍然主要在工业界和学术界应用，个人应用还很少。

从 20 世纪 80 年代中期至今，计算机系统发展已进入第四代。这时人们已经不再看重单台计算机和程序，人们感受到的是硬件和软件的综合效果。由复杂操作系统控制的强大的桌面机、局域网和广域网，与先进的应用软件相配合，已经成为当前计算机发展的主流。计算机体系结构已迅速地从集中的主机环境转变成分布的客户机/服务器（或浏览器/服务器）环境，面向对象技术已经在许多领域迅速地取代了传统的软件开发方法。软件产业在世界经济中已经占有举足轻重的地位。

1.2.2 软件危机

20 世纪 60 年代末、70 年代初，西方工业发达国家经历了一场“软件危机”。这场软件危机表现在以下两点：一方面，软件十分复杂，价格昂贵，供需差日益增大；另一方面，软件开发时又常常受挫，质量差，指定的进度表和完成日期很少能按时实现，研制过程很难管理，即软件的研制往往失去控制。我们称软件开发和维护过程中所中遇到的这一系列严重问题为软件危机。软件危机包含下述两方面的问题：如何开发软件，以满足对软件日益增长的需求；如何维护数量不断膨胀的已有软件。

具体来说，软件危机主要有以下一些表现：

（1）软件功能不能满足用户的实际需要。软件开发人员和用户之间的信息交流往往很不充分，“闭门造车”必然导致用户对软件产品的不满意。

（2）软件产品的质量差。软件可靠性和质量保证的确切定量概念刚刚出现不久，软件质量保证技术还没有坚持不懈地应用到软件开发的全过程中，这些都导致软件产品发生质量问题。

（3）软件开发生产率低。软件生产率提高的速度远远不能满足客观需要，也跟不上硬件的发展速度，使人们不能充分利用现代计算机硬件所提供的巨大潜力。

（4）软件开发成本和进度的估计常常很不准确。实际成本比估计成本有可能高出一个数量级，实际进度比预期进度拖延几个月甚至几年的现象并不罕见。

（5）软件无配套的文档资料。软件开发人员可以利用文档资料，在软件开发过程中准确地交流信息；对于软件维护人员而言，这些文档资料更是至关重要、必不可少的。缺乏完整、合格的文档资料，必然给软件开发和维护带来许多严重的困难和问题。

（6）软件的可维护性差。很多程序中的错误非常难改正，并且不能使这些程序适应新的硬件环境，也不能根据用户的需要在原有程序中增加一些新的功能。

（7）软件的价格昂贵。软件成本在计算机系统总成本中所占的比例逐年上升。

1.3 软件工程

1.3.1 软件工程的定义

要克服软件危机，软件开发必须寻找新的方法，创造新的理论。人们从失败中吸取教训，经过思索，得出如下结论：过去软件由同一个（或同一组）人编写和运行，如果出了故障也由同一个（或同一组）人来排除，这种手工作坊式的软件生产方式已经过时；软件不能再是个人艺术作品，软件开发人员不能是艺术家，软件开发人员应该是工程师。

有人从制造一台机器或修建一座楼房的过程中受到启发，提出：可否像机械工程、建筑工程那样，有计划、有步骤、守纪律地开展软件的开发工作？回答是肯定的，于是萌生了一种想法：像处理“工程”问题一样来处理软件开发全过程，这就产生了“软件工程”这一概念。“软件工程”一词于 1968 年问世以来，经过许许多多研究人员卓有成效的研究，终于产生了一门新兴的学科——软件工程学。

软件工程是指研究软件生产的一门学科，也就是将完善的工程原理应用于经济地生产既可靠又能在实际机器上有效运行的软件。1983 年美国《IEEE 软件工程标准术语》对软件工程下的定义为：软件工程是开发、运行、维护和修复软件的系统方法，其中“软件”的定义为计算机程序、方法、规则、相关的文档资料以及在计算机上运行时所必需的数据。

1.3.2 软件工程方法学

通常把在软件生命周期全过程中使用的一整套技术的集合称为软件工程方法学。软件工程方法学包括三个要素：方法、工具和过程。

软件工程方法是完成软件开发的各项任务的技术方法，为软件开发提供了“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。软件工程方法中常采用某种特殊的语言或图形表达方法和一套质量保证标准。

软件工具为软件工程方法提供了自动的或半自动的软件支撑环境。目前已经开发出了许多软件工具，可以用于支持上述的软件工程方法。而且已经有人把诸多软件工具集成起来，使得一种工具产生的信息可以为其他的工具所使用，这样就建立起一种称之为计算机辅助软件工程（CASE）的软件开发支撑系统。

软件工程的过程则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。过程定义了方法使用的顺序、要求交付的文档资料、为保证质量和协调变化所需要的管理以及软件开发各个阶段完成的里程碑。

传统方法学和面向对象方法学是目前使用得最广泛的两种软件工程方法学。本书后面

各章将对传统方法学进行详细的介绍，并简要介绍面向对象方法学。以下先对这两种方法学做一介绍和对比。

传统方法学也称为生命周期方法学或结构化范型，它采用结构化技术来完成软件开发的各项任务，并使用适当的软件工具或软件工程环境来支持结构化技术的运用。这种方法学把软件生命周期的全过程依次划分为若干个阶段，然后顺序地逐步完成每个阶段的任务。采用这种方法学开发软件的时候，从对任务的抽象逻辑分析开始，分阶段地进行开发。前一个阶段任务的完成是开始进行后一个阶段工作的前提和基础，而后一阶段任务的完成通常是使前一阶段提出的解法更进一步具体化，加进了更多的实现细节。每一个阶段的开始和结束都有严格标准，对于任何两个相邻的阶段而言，前一阶段的结束标准就是后一阶段的开始标准。在每一个阶段结束之前都必须进行正式严格的技术审查和管理复审，从技术和管理两方面对这个阶段的开发成果进行检查，通过之后这个阶段才算结束；如果检查通不过，就必须进行必要的返工，并且返工后还要再经过审查。审查的一条主要标准就是每个阶段都应该交出“最新式的”（即和所开发的软件完全一致的）高质量的文档资料，从而保护在软件开发工程结束时有一个完整准确的软件配置交付使用。在完成生命周期每个阶段的任务时，应该采用适合该阶段任务特点的系统化的技术方法——结构化分析、结构化设计、结构化程序设计或结构化测试技术。

传统方法学的优点在于：把软件生命周期划分成若干个阶段，每个阶段的任务相对独立，而且比较简单，便于不同人员分工协作，从而降低了整个软件开发工程的困难程度；在软件生命周期的每个阶段都采用科学的管理技术和良好的技术方法，而且在每个阶段结束之前都从技术和管理两个角度进行严格的审查，合格之后才开始下一阶段的工作，这就使软件开发工程的全过程以一种有条不紊的方式进行，保证了软件的质量，特别是提高了软件的可维护性。总之，采用传统方法学可以大大提高软件开发的成功率和软件开发的生产率。

但是传统方法学并不适合于以下情况：软件规模庞大，或者对软件的需求是模糊的或随时间变化的。同时，使用传统方法学开发出来的软件，可维护性并不是最佳的。归结原因，在于传统方法学存在以下局限：这种技术要么面向行为（即对数据的操作），要么面向数据，没有既面向数据又面向行为的结构化技术。同时，离开了操作便无法更改数据，而脱离了数据的操作更是毫无意义的，数据和对数据的处理原本就是密切相关的。然而，传统方法学把数据和对数据的处理人为地分离成两个独立的部分，自然增加了软件开发与维护的难度。

为了克服传统方法学的局限，逐步发展并形成了一种新的方法学——面向对象方法学。面向对象方法把数据和行为看成同等重要，它是一种以数据为主线，把数据和对数据的操作紧密地结合在一起的方法学。

面向对象方法具有以下四个要点：

（1）把对象作为融合了数据以及在数据上的操作行为的统一的软件构件。面向对象的程序是由对象组成的，程序中任何元素都是对象，复杂对象由比较简单的对象组合而成。

（2）把所有对象都划分成类。每个类都定义了一组数据和一组操作，类是对具有相同数据和相同操作的一组相似对象的定义。数据用于表示对象的静态属性，是对象的状态信息，而施加于数据之上的操作用于实现对象的动态行为。

（3）根据基类与派生类的关系，把若干个相关类组成一个层次结构的系统。在类的等级中，下层派生类自动拥有上层基类所定义的数据和操作，这种现象称为继承。

（4）对象之间只能通过发送消息相互联系。对象与传统数据有着本质的区别：对象不

是被动地等待外界对它施加操作，相反，对象是进行处理的主体，必须向它发消息请求它执行它的某个操作以处理它的数据，而不能从外界直接对它的数据进行处理。也就是说，对象的所有私有信息都被封装在该对象内，不能从外界直接访问，这就是通常所说的封装性。

面向对象方法学的出发点和基本原则，是尽可能模拟人类习惯的思维方式、方法与过程，尽可能接近人类认识世界、解决问题的方法与过程，从而使描述问题的问题空间（也称为问题域）与实现解法的解空间（也称为求解域）在结构上尽可能一致。用面向对象方法学开发软件的过程，是一个主动地多次反复迭代的演化过程。面向对象方法在概念和表示方法上的一致性，也保证了在各项开发活动之间的平滑（无缝）过渡。

正确地运用面向对象方法学开发软件，最终的软件产品由许多较小的基本上独立的对象组成，而且大多数对象都与现实世界中的实体相对应，因此，降低了软件产品的复杂性，提高了软件产品的可理解性，简化了软件的开发和维护工作。由于对象是相对独立的实体，容易在以后的软件产品中重复使用，因此，与传统方法学相比，面向对象方法学促进了软件的重用。同时，面向对象方法学中对象所特有的继承性，又使软件的可重用性进一步提高。

1.4 软件生存期和软件开发模型

1.4.1 软件生存期

如同任何其他事物一样，软件也有一个孕育、诞生、成长、成熟、衰亡的生存过程，一般称之为计算机软件的生存期。

一般说来，软件生命期由软件定义、软件开发和软件维护三个时期组成，每个时期又可进一步划分成若干个阶段。

1. 软件定义时期

（1）问题定义：这是软件生存期的第一个阶段，主要任务是弄清用户要计算机解决的问题是什么。通过问题定义阶段的工作，系统分析员应该提出关于问题性质、工程目标和规模的书面报告。

（2）可行性研究：任务是为前一阶段提出的问题寻求一种至数种在技术上可行且在经济上有较高效益的解决方案。可行性研究阶段应该导出系统的高层逻辑模型（通常用数据流图表示），并且在此基础上更准确、更具体地确定工程的规模和目标。然后由系统分析员更准确地估计系统的成本和效益，对系统进行仔细的成本/效益分析。同时，还应制订出人力、资源及进度计划。

2. 软件开发时期

（1）需求分析：任务在于弄清用户对软件系统的全部需求，主要是确定目标系统必须具备哪些功能。系统分析员在需求分析阶段必须和用户密切配合，充分交流信息，以得出经过用户确认的系统逻辑模型。通常用数据流图、数据字典和简要的算法表示系统的逻辑模型。这些文档既是软件系统逻辑模型的描述，也是下一阶段进行设计的依据。

（2）总体设计：任务是设计软件的结构，即确定程序由哪些模块组成以及模块间的关系。通常用层次图或结构图描绘软件的结构。