

高等职业院校国家技能型紧缺人才培养培训工程规划教材
· 计算机应用与软件技术专业



软件工程与 UML

胡林玲 主编 黄奇 副主编



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

高等职业院校国家技能型紧缺人才培养培训工程规划教材·计算机应用与软件技术专业

软件工程与 UML

胡林玲 主 编

黄 奇 副主编

電子工業出版社

Publishing House of Electronics Industry

内 容 简 介

本教程内容按照软件工程的概念、软件开发的过程、软件开发过程的管理三大部分顺序安排，主要包括：软件工程的概念，统一软件开发过程（RUP）与统一建模语言（UML），软件过程的管理与改进，项目可行性分析与软件需求分析，系统分析与设计，系统实现，软件测试，软件配置管理与软件维护，软件项目管理等八章。每章节都通过案例的方式，以简洁的文字对相关内容进行介绍。一般在讲解内容后紧跟练习，可使读者进一步掌握相关的知识点；“综合实训”主要训练将本书理论与实际问题联系的能力。本教程各部分层次清楚、内容既相互联系又相对独立，读者可根据自己需要进行选择。

本书系统性强、内容实用，可作为高等职业技术学院或大学专科软件工程课程教材，也适合从事软件开发的软件工程师、UML的初学者自学与参考。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目(CIP)数据

软件工程与 UML / 胡林玲主编. —北京：电子工业出版社，2005.2
高等职业院校国家技能型紧缺人才培养培训工程规划教材·计算机应用与软件技术专业
ISBN 7-121-00962-5

I. 软… II. ①胡… III. ①软件工程—高等学校：技术学校—教材②面向对象语言，UML—程序设计—高等学校：技术学校—教材 IV. ①TP311.5②TP312

中国版本图书馆 CIP 数据核字（2005）第 012044 号

责任编辑：赵江晨

印 刷：北京牛山世兴印刷厂

出版发行：电子工业出版社

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 字数：341.2 千字

印 次：2005 年 2 月第 1 次印刷

印 数：5 000 册 定价：18.00 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。质量投诉请发邮件至 zltz@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前 言

为了克服与摆脱软件危机，人们希望通过工程技术方法和管理手段使软件开发纳入工程化的轨道，由此产生了软件工程学。软件工程是一门指导计算机软件开发与维护的工程学科，以提高软件生产的质量与效率为宗旨。经过三十多年的发展，软件工程学已逐渐成熟，现已成为计算机科学与技术领域中一门重要的学科。

编写目的 软件业人士已经认识到软件工程学对软件质量和生产效率影响的重要性，软件工程已成为一门软件技术相关人员必须掌握的学科。对于高等职业院校的学生或大学专科学生来讲，他们没有在实际的软件开发中吃足苦头，对软件危机的认识不足，对软件工程学的理论不感兴趣。但他们又是将来软件生产过程的主要生产者，是软件质量的直接影响者，必须要让他们掌握软件工程的基本理论和常用方法、工具，了解软件开发的过程。这就需要一本淡化理论、突出实用的教程，本书正是应这种需求而编写的。

主要内容 本书分为四个部分，第一部分介绍软件工程学的概论；第二部分介绍软件开发过程中项目可行性研究与软件需求分析、系统设计、系统实现、软件测试等核心工作流的任务和完成任务所采用的技术方法、利用的工具，具体的工作指导；第三部分介绍软件开发过程管理方面的内容，本书对 UML 是作为软件开发过程中需要用到的工具来介绍的，着重介绍 UML 在面向对象的软件工程学方法中的具体应用；第四部分实例给出软件生产与管理过程中具体的文档规范和实例。

本书特色 本书语言精练、准确、深入浅出；淡化理论性阐述，突出实践指导；将软件工程的理论和概念与具体的开发实例相结合进行讲解；重点讲解将软件工程的理论运用到指导实际的软件开发工作时，每一阶段该如何做的问题。本书不仅有助于读者深入了解软件工程的理论，而且有助于读者学会用软件工程的原理和方法指导实际的软件开发工作。

主要作者 本书由深圳信息职业技术学院胡林玲、黄奇编写，其中第 1, 2, 4, 5, 6 章与实例 A, B, C, D, F, G 由胡林玲编写，第 3, 7, 8, 9 章与实例 E 由黄奇编写。本书由上海应用技术学院的陆惠恩主审。蒋方纯、黄锐军、覃国蓉、易函平、易力、胡惠、谢霏、霍红影等对本书的编写提供了许多帮助，在此一并表示感谢！

由于时间仓促，加之作者水平有限，书中不当之处在所难免，恳请读者批评指正。我们的 E-mail 地址是：hull@szit.com.cn 和 huangq@szit.com.cn。

编 者
2005 年 1 月于深圳

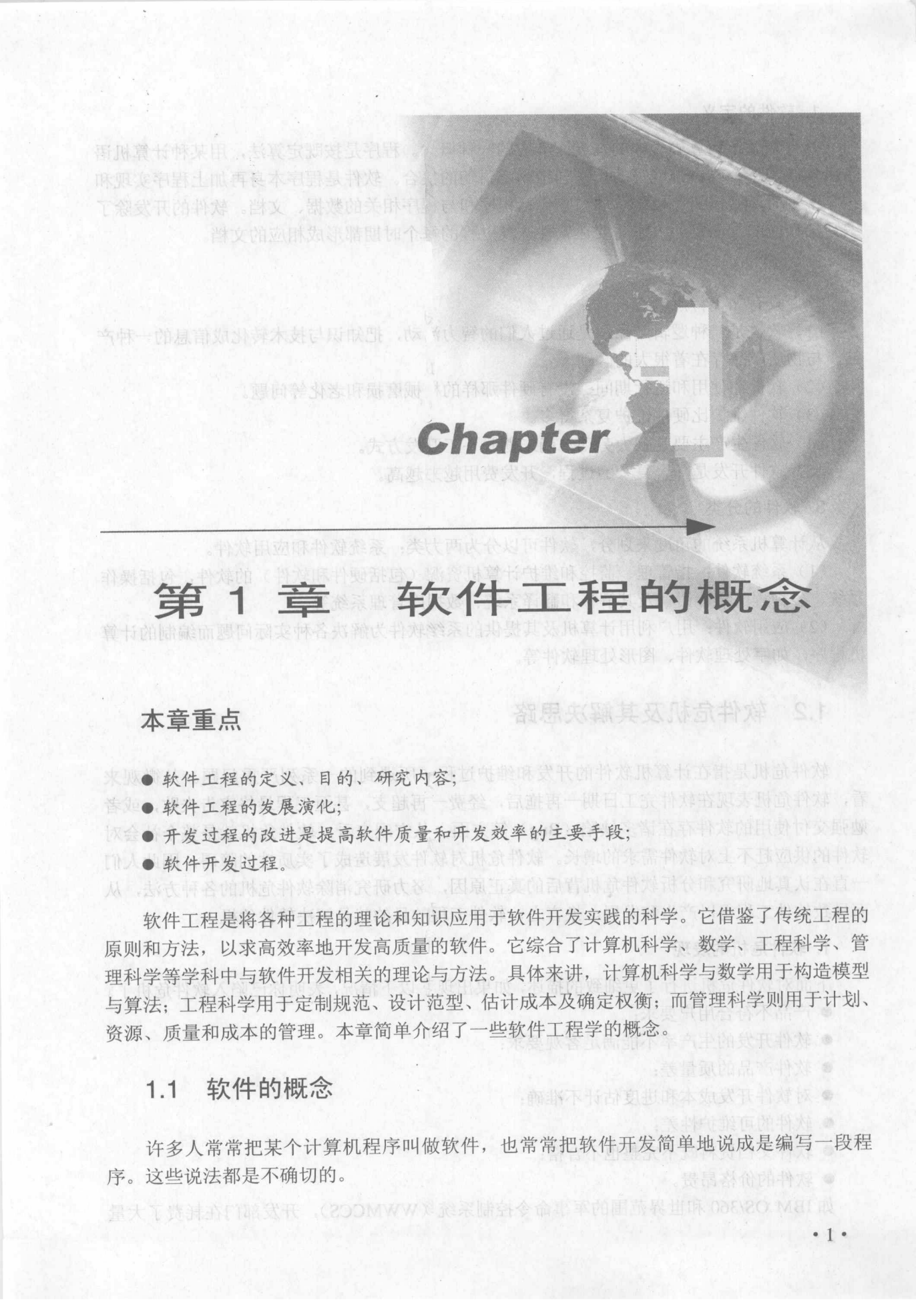
目 录

第 1 章 软件工程的概 念	(1)
1.1 软件的概念	(1)
1.2 软件危机及其解决思路	(2)
1.3 软件工程	(3)
1.4 生存周期模型	(5)
1.4.1 软件生存周期各阶段的基本任务	(5)
1.4.2 几种生存周期模型	(6)
1.4.3 统一软件开发过程	(8)
1.4.4 与模型相应的知识体系的形成	(8)
1.5 优秀软件的属性	(9)
1.6 内容总结	(10)
习题 1	(10)
第 2 章 统一软件开发过程 (RUP) 与统一建模语言 (UML)	(11)
2.1 软件开发的几个指导原则及模型的概念	(12)
2.2 统一软件开发过程 (RUP)	(13)
2.2.1 统一过程概述	(13)
2.2.2 统一过程的四个工作阶段	(14)
2.2.3 统一过程的五个核心工作流	(15)
2.2.4 统一软件开发过程的生命周期	(16)
2.2.5 RUP 开发过程的特点	(17)
2.3 UML 概述	(20)
2.4 UML 结构	(22)
2.4.1 UML 中的事物	(23)
2.4.2 UML 中的关系	(25)
2.4.3 用 UML 对系统建模	(26)
2.5 UML 中的图	(28)
2.5.1 UML 规则	(28)
2.5.2 类图	(32)
2.5.3 对象图	(33)
2.5.4 用例图	(33)
2.5.5 顺序图	(35)
2.5.6 协作图	(36)
2.5.7 状态图	(37)
2.5.8 活动图	(39)
2.5.9 构件图	(41)
2.5.10 实施图	(42)

2.5.11 绘制 UML 图	(43)
2.6 内容总结	(45)
习题 2	(45)
第 3 章 软件过程的管理与改进	(47)
3.1 软件过程管理与改进概述	(48)
3.1.1 软件过程的概念	(48)
3.1.2 软件过程改进的实际意义	(48)
3.1.3 软件过程建模与软件过程改进的理论与方法	(49)
3.1.4 过程改进的模式与体系	(50)
3.1.5 过程改进的原则和步骤	(50)
3.1.6 过程改进活动的组织与实施	(51)
3.2 度量软件过程	(52)
3.3 能力成熟度模型 CMM	(53)
3.4 个体软件过程 PSP	(57)
3.4.1 个体软件过程框架	(57)
3.4.2 个体软件过程策略和方法	(58)
3.5 团体软件过程 TSP	(60)
3.6 内容总结	(61)
习题 3	(61)
第 4 章 项目可行性分析与软件需求分析	(62)
4.1 可行性研究和需求分析的任务	(62)
4.2 项目的可行性分析	(63)
4.3 需求分析方法	(65)
4.4 结构化需求分析与 RUP 需求分析比较	(72)
4.5 RUP 获取用户需求实例	(73)
4.6 内容总结	(76)
习题 4	(76)
第 5 章 系统分析与设计	(77)
5.1 几种常用的体系结构	(78)
5.2 结构化概要设计	(79)
5.2.1 概要设计的目的和任务	(79)
5.2.2 概要设计的过程	(80)
5.2.3 概要设计的工具、方法和结果	(84)
5.3 结构化详细设计	(86)
5.4 RUP 的分析与设计	(87)
5.4.1 RUP 系统分析和设计	(87)
5.4.2 RUP 系统分析与设计实例	(88)
5.5 结构化方法与面向对象方法特点比较	(93)
5.6 内容总结	(93)
习题 5	(93)

第6章 系统实现	(94)
6.1 选择一种合适的程序设计语言	(94)
6.1.1 几种不同程序设计语言的特点	(95)
6.1.2 选择不同的程序设计语言	(96)
6.2 编码风格——程序员个人修养	(97)
6.2.1 程序的版式	(97)
6.2.2 输入/输出	(99)
6.3 程序效率	(99)
6.4 程序的安全性	(100)
6.5 内容总结	(103)
习题6	(103)
第7章 软件测试	(104)
7.1 软件测试概述	(105)
7.1.1 软件错误的根源	(105)
7.1.2 软件测试在软件项目中的地位	(106)
7.1.3 软件测试基础与软件测试策略	(107)
7.1.4 测试成熟度、测试过程及测试生命周期的模型	(108)
7.1.5 测试工具与自动测试	(109)
7.2 软件测试的基本原理与技术	(110)
7.2.1 白盒与黑盒原理	(110)
7.2.2 覆盖率	(110)
7.2.3 动态测试与静态测试	(112)
7.2.4 程序分析技术	(113)
7.2.5 测试分析方法与测试设计技术	(113)
7.2.6 自动测试生命周期方法学	(115)
7.2.7 测试工具及其分类	(115)
7.3 需求测试与设计测试	(117)
7.4 单元测试、集成测试与系统测试	(118)
7.4.1 单元测试	(119)
7.4.2 集成测试	(121)
7.4.3 系统测试	(122)
7.5 测试团队与角色	(123)
7.6 内容总结	(124)
习题7	(124)
第8章 软件配置管理与软件维护	(125)
8.1 软件配置管理	(125)
8.1.1 软件配置项与基线	(126)
8.1.2 软件配置管理的组织与计划	(126)
8.1.3 软件配置管理过程	(127)
8.1.4 配置管理工具	(129)

8.2 软件维护	(129)
8.2.1 软件维护的基本内容	(129)
8.2.2 软件维护的类型	(130)
8.2.3 软件维护代价	(130)
8.2.4 软件维护实施	(131)
8.2.5 程序修改	(132)
8.2.6 软件的可维护性	(133)
8.2.7 软件再工程	(134)
8.3 内容总结	(135)
习题 8	(135)
第 9 章 软件项目管理	(136)
9.1 项目管理的常见概念	(137)
9.2 项目计划	(138)
9.3 项目监控	(139)
9.4 软件质量管理	(141)
9.4.1 软件质量的基本问题	(141)
9.4.2 如何提高软件质量	(142)
9.4.3 全面质量管理	(143)
9.5 软件外包管理	(143)
9.6 软件项目的人员管理	(144)
9.7 项目结项管理	(146)
9.8 内容总结	(147)
习题 9	(147)
实例 A 应用于个人的 RUP 实例——计时项目开发过程	(148)
实例 B 需求规格说明书例	(157)
实例 C 概要设计说明书例	(167)
实例 D 详细设计说明书例	(176)
实例 E 一个完整的测试计划	(181)
实例 F 面向对象软件设计说明书例	(188)
实例 G 综合实训	(198)
参考文献	(200)



Chapter

第 1 章 软件工程的观念

本章重点

- 软件工程的定义、目的、研究内容;
- 软件工程的发展演化;
- 开发过程的改进是提高软件质量和开发效率的主要手段;
- 软件开发过程。

软件工程是将各种工程的理论和知识应用于软件开发实践的科学。它借鉴了传统工程的原则和方法,以求高效率地开发高质量的软件。它综合了计算机科学、数学、工程科学、管理科学等学科中与软件开发相关的理论与方法。具体来讲,计算机科学与数学用于构造模型与算法;工程科学用于定制规范、设计范型、估计成本及确定权衡;而管理科学则用于计划、资源、质量和成本的管理。本章简单介绍了一些软件工程学的概念。

1.1 软件的概念

许多人常常把某个计算机程序叫做软件,也常常把软件开发简单地说是编写一段程序。这些说法都是不确切的。

1. 软件的定义

软件是由于计算机程序的发展而形成的一种概念。程序是按既定算法，用某种计算机语言所规定的指令或语句编写的一系列指令或语句的集合。软件是程序本身再加上程序实现和程序维护时所必需的文档的总称。它包括程序和与程序相关的数据、文档。软件的开发除了编写出相应的程序之外，还需要在软件开发过程的每个时期都形成相应的文档。

2. 软件的特点

软件具有如下特点：

- (1) 软件是一种逻辑产品，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，与物质产品存在着很大的区别。
- (2) 软件在使用和运行期间，没有硬件那样的机械磨损和老化等问题。
- (3) 软件维护比硬件维护复杂得多。
- (4) 软件生产主要是脑力劳动，尚未摆脱手工开发方式。
- (5) 软件开发是一个复杂的过程，开发费用越来越高。

3. 软件的分类

从计算机系统的角度来划分，软件可以分为两大类：系统软件和应用软件。

(1) 系统软件：指管理、监控和维护计算机资源（包括硬件和软件）的软件，包括操作系统、各种程序设计语言及其解释和翻译系统、数据库管理系统等。

(2) 应用软件：用户利用计算机及其提供的系统软件为解决各种实际问题而编制的计算机程序，如字处理软件、图形处理软件等。

1.2 软件危机及其解决思路

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。从微观来看，软件危机表现在软件完工日期一再拖后，经费一再超支，甚至工程最终宣告失败，或者勉强交付使用的软件存在诸多缺陷（Bug）等方面；从宏观来看，软件危机就是整个社会对软件的供应赶不上对软件需求的增长。软件危机对软件发展造成了实质性的障碍，因此人们一直在认真地研究和分析软件危机背后的真正原因，努力研究消除软件危机的各种方法，从而导致软件工程得以产生和发展。换言之，软件工程的目的是解决软件危机。

1. 软件危机的表现

下面对软件危机进行了更细致的描述，如果出现了以下情况，表明你已陷入软件危机了。

- 产品不符合用户要求；
- 软件开发的生产率不能满足客观要求；
- 软件产品的质量差；
- 对软件开发成本和进度估计不准确；
- 软件的可维护性差；
- 软件文档资料既不完整也不合格；
- 软件的价格昂贵。

如 IBM OS/360 和世界范围的军事命令控制系统（WWMCCS），开发部门在耗费了大量

的人力和财力之后，由于离预定目标相差甚远不得不宣布失败；软件开发的生产率从来就不能满足客观要求，所以你满眼看到的都是加班加点拼命干活的软件开发人员；补丁和 Bug 满天飞，动不动就“非法操作”、“关机”、“重启”，甚至于弄到要重新格式化硬盘，这些都是软件产品质量不尽如人意的结果。至于软件开发的进度估计，请现在估计一下，立即动手开发一个学生成绩管理系统要多长的时间？估计之后立即动手开发直到完成，软件开发的进度估计得准确吗？Linux 作为一个免费的操作系统有许多应用软件都可以从网上下载得到，请将其中的任一款软件下载后再改一改功能，立即就会发现缺少相关的说明文档，读懂别人的源程序已属不易，要想修改别人的程序则根本无从下手，这些都是软件危机的表现。

2. 解决软件危机的途径

人们在认真地分析和研究软件危机背后的真正原因后，对解决软件危机提出了以下几条途径：

(1) 纠正错误认识。如果你有下列错误认识，就请立即改正它。

- 软件就是一个程序；
- 有一个对目标的概括性描述就可以编写程序了，许多细节可以在以后补充；
- 所谓软件开发就是编写程序并使它运行；
- 用户对软件的要求不断变化，然而软件是柔性灵活的，可以被轻易地改动；
- 软件投入生产性运行以后，需要的维护工作量不多，而且维护是一种很简单的工作。

以上这些错误的认识一定会让你陷入危机，也很具有代表性。

(2) 将软件开发活动工程化，借鉴其他领域的经验以指导软件开发。应该认识到软件开发不是某种个体劳动的神秘技巧，而应是一种组织良好、管理严密、各类人员协同配合共同完成的工程项目。人们发现：将传统工程学的原理、技术和方法应用于软件开发上，可以使软件生产规范化，有利于提高开发质量，降低成本和控制进度，也就是用软件工程来规范软件开发活动，并且在技术措施和组织管理措施两方面进行不断改进。软件生产工程化思想的提出，使软件生产走向正规化，为解决软件危机找准了努力的方向。

(3) 不断发展新的软件工具，以提高软件生产的效率和质量。在软件开发的每个阶段都有许多烦琐重复的工作要做，在适当的软件工具辅助下，开发人员可以把这类工作做得既快又好。如：各种集成开发环境、代码生成器、集成测试环境、统一建模语言（Unified Modeling Language, UML）等建模工具、数据字典管理系统、帮助绘制各种流程图的专用工具、项目管理工具甚至是操作系统本身，它们的功能都在不断加强。

1.3 软件工程

从计算机诞生的那一天起，软件的发展就跟不上硬件的发展，就无法满足日益增长的社会需求，就存在着软件危机。随着软件的发展，软件的规模越来越大，功能越来越复杂，软件开发的计划、监督和控制变得更加困难，软件危机加深了。解决软件危机的种种努力最终导致了软件工程的产生和发展。

1. 软件工程的定义和目标

软件工程是用工程科学的知识和技术原理来定义、开发、维护软件的一门学科。简单地讲，软件工程就是制定并使用完整的工程法规、优秀的管理策略、合适的工具和方法，在已

知的适当的资源设备条件下，获得具有明确意义的高质量的软件。更具体地讲，就是用工程科学的观点估算费用，制定进度、计划和方案；用管理科学的方法和原理进行生产管理；用数学的方法建立软件开发中的各种模型和算法。

软件工程的目的是：降低开发成本；达到要求的软件功能，取得较好的软件性能；开发的软件易于移植；降低维护费用；能按时完成开发任务，及时交付使用；开发的软件可靠性高。

2. 软件工程研究的问题

软件工程从软件的开发、运行、维护等方面研究软件生存的全过程，主要的研究内容包括开发技术与开发管理两大方面。软件与其他商业活动一样，包括人、过程与技术三大要素。据此软件工程学可分为下面这些分支：

(1) 软件工程技术。它的发展最早也最快，主要包括软件开发方法、软件开发生命周期模型、软件设计的交流语言、软件开发的工具与环境等。

(2) 软件开发过程、软件开发的工程过程模型。它主要包括软件工程基本的工程步骤，即需求分析、软件概要设计、软件详细设计、软件编写指南、软件开发标准、软件测试、软件实施与维护等。

(3) 软件项目管理。它主要包括项目的计划和组织，团队人员的管理，质量评估，项目的经济性指标，工作量估计，费用估计，项目管理的制度、标准、文档等诸多方面。这些都是软件开发管理者需要研究和探讨的问题，也是软件项目过程中与人相关的、直接影响软件项目成败的最重要因素。

(4) 软件实施与维护。它是对软件实施、维护、培训、升级等方面的指导，主要包括软件的实施步骤、实施实践的成功保障、软件的维护指南、用户使用的培训以及软件升级和软件再工程的实现等。

在软件工程的人、过程与技术这三大要素中，过程和人是导致成功或失败的关键因素，而人是过程不可分割的一部分，因为人是整个过程活动、过程监控、过程管理的关键。所以，要想彻底摆脱软件危机，除了发展先进的开发技术、有效的开发工具、合理有效的开发过程模型之外，最重要的是改变人这个要素在开发过程中的思想和行为。

3. 软件开发方法学

软件开发方法学是指编制软件的系统方法。它确定开发的各个阶段，规定每一阶段的活动、产品、验证步骤和完成准则。它是一种思想，指导软件开发用什么样的工具，经过哪些工作步骤（过程），用什么样的方法来完成开发任务。

目前使用最广泛的软件开发方法有两种：结构化方法和面向对象方法。

(1) 结构化方法。结构化方法又叫传统的软件工程学方法。它采用结构化的技术（结构化分析、结构化设计、结构化程序设计、结构化测试）来完成软件开发的各项任务。这种方法曾经给软件产业带来了巨大的进步，但它也有缺点：难以解决大型软件开发中的问题，代码重用率低，产品维护难。

(2) 面向对象方法。面向对象方法不仅是一些具体的软件开发技术与策略，而且是一整套关于如何看待软件系统与现实世界的关系，以什么观点来研究问题并进行求解，以及如何如何进行系统构造的软件方法学。面向对象的软件工程方法是面向对象方法在软件工程领域的全面运用。它包括面向对象的分析、面向对象的设计、面向对象的编程（又称为“面向对象的

实现”)、面向对象的测试、面向对象的软件维护。

4. 开发过程方法学

软件开发技术的发展,提升了软件开发人员个人的素质与能力,那么是不是光有优秀的开发人员就能保证软件开发的成功呢?让我们来看这样一个例子:现在有一组人要在规定的时间内开发一个软件,他们都努力工作。如果不用某种过程规则来约束他们的话,那么很可能不同的人采用不同的开发语言,使用不同的命名规则,当一个人对系统进行改动时,也不考虑其他人的情况,可以想像,这个小组最终只能得到一个混乱的结果。反过来,如果团队中的每个人都能遵守共同的过程规则,则情况就会大不相同:

- 为实现共同的目标,所有成员的活动都应相互协调;
- 为避免相互间产生矛盾与冲突,成员的活动应该统一;
- 每个成员所应该达到的目标是可以度量的;
- 能够重复团队以前的成果。

总之,组织中的每位成员都遵循相同的规则去约束个人的行为,为共同的目标而努力。随着软件规模越来越大,功能越来越复杂,作用越来越重要,软件开发的计划、监督和控制变得更加困难。软件工程从未停止对开发过程的研究,探索出了多种不同的开发过程模型(见 1.4.2 节)。

从软件开发的实践中认识到软件开发过程的管理是至关重要的。个体软件过程(Personal Software Process, PSPi)是对开发工程师个人在开发过程中行为的规范,群组软件开发过程(Team Software Process, TSPi)是对小组开发软件的过程进行规范,CMM 则是对企业开发软件过程进行规范。从 20 世纪 80 年代开始,由美国国防部资助,SEI 最先提出了“软件能力成熟度模型(Software Capability Maturity Model, SW-CMM)”理论,它得到了众多国家软件业的认可和广泛应用。CMM 提供了一个软件过程改进的框架,这个框架与软件生命周期无关,也与所采用的开发技术无关。软件企业根据这个框架开发企业内部具体的软件过程,可以极大地提高企业按计划的时间和成本提交有质量保证的软件的能力。

1.4 生存周期模型

软件从定义、存在到消亡持续的整个时期叫做软件的生存周期,又叫生命周期。软件开发工程化的思想将软件生存周期分成软件的定义、开发和运行维护三个时期。定义时期又可进一步分成:问题定义、可行性研究、需求分析三个阶段;开发时期又可进一步分成:概要设计、详细设计、编码和单元测试、综合测试几个阶段;维护活动实质上是一次压缩和简化了的定义和开发过程。软件生存周期模型规定把软件生存周期分成哪些阶段以及各阶段的任务、执行顺序,应该交付的文档资料,为保证软件质量和协调开发过程中的各种变化所需要采取的管理措施,标志软件开发各个阶段任务完成的里程碑。软件生存周期模型又叫过程模型。过程模型的发展演化反映了软件工程技术的发展。

1.4.1 软件生存周期各阶段的基本任务

下面简要介绍上述各个阶段应该完成的基本任务。

(1) 问题定义。确定“要解决的问题是什么”。给出关于问题性质、工程目标、工程规模的书面报告。

(2) 可行性研究。回答“上一个阶段所确定的问题是否有行得通的解决办法”。目的就是用最小的代价在尽可能短的时间内确定问题是否能够解决。它只研究问题是否值得去解决而不是解决问题本身。

(3) 需求分析。需求分析的任务是确定系统必须完成哪些工作，也就是对目标系统提出完整、准确、清晰、具体的要求。回答“目标系统必须做什么”。

(4) 概要设计。概要设计的任务是概括地回答“怎样实现目标系统”这个问题，它是实现目标系统的一个框架，又叫初步设计或总体设计，也可叫逻辑设计或高层设计。

(5) 详细设计。概要设计阶段以比较抽象概括的方式提出了解决问题的办法。详细设计阶段的任务就是把解决方法具体化，但它还不是编写程序，而是设计出程序的模块、每个模块的算法、采用的数据结构，又叫模块设计，也可叫物理设计或低层设计。

(6) 编码和单元测试。这个阶段的关键任务是：设计出正确的、容易理解和维护的程序模块，并且仔细测试每一个模块。

(7) 综合测试。这个阶段的关键任务是：把经过单元测试的模块按某种选定的策略装配起来，在装配过程中进行必要的测试。最后由用户对目标系统对照需求进行验收测试。

(8) 软件维护。每一项维护活动都要经过提出维护要求、分析维护要求、确定维护方案、修改软件、测试修改、验收修改等多个步骤，所以，维护实质上是一次压缩了的软件定义和开发的全过程。

1.4.2 几种生存周期模型

1. 瀑布模型

瀑布型生存周期模型本质上是开发一次性成功，并且每个阶段活动按顺序进行一次的方法。在这种方法中，前一阶段的工作成果是后一阶段工作开始的基础，所以，每个阶段都必须交出合格的文档，必须对前阶段的工作进行评审，前一阶段的工作完成之后才可以开始后一阶段的工作。实际的瀑布模型如图 1.1 所示。

瀑布模型是一次开发就将一个包含全部功能的产品交付使用，这未免太理想化。事实上，人在工作中不可能不犯错误，阶段评审也不能发现所有错误。在瀑布模型里，在越早阶段犯下的错误，在越晚的阶段暴露出来，排除故障的代价也越高。实际的瀑布模型是带有反馈环的，图 1.1 中向上的实线箭头线表示反馈，虚线箭头线表示有信息流动。

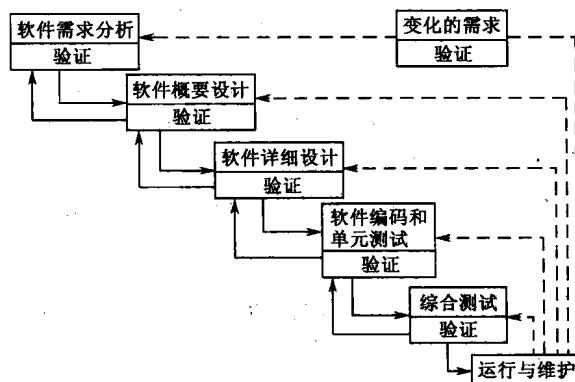


图 1.1 实际的瀑布模型

2. 增量模型

增量模型又叫有计划的产品改进模型，它从一组给定的需求开始，通过构造一系列的可执行工作版本来实施开发活动。第一个工作版本纳入一些需求，下一个工作版本纳入更多的需求，依次类推（图中用更大些的方框表示纳入了更多的需求），直到系统完成。每个工作版本都要执行需求分析、概要设计、详细设计、编码与单元测试、综合测试等过程，完成相应的任务。图 1.2 所示的增量模型中的虚线表示这些步骤间可能有信息流动。

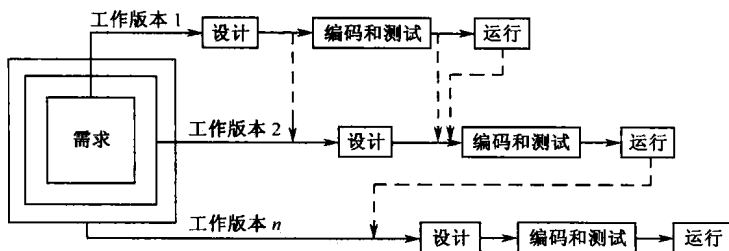


图 1.2 增量模型

3. 演化模型

演化模型与增量模型类似，也是通过构造系统的各个可执行工作版本来开发系统，但它与增量模型有区别，承认需求不能在初始时就完全被正确确定。在该方法中，需求的一部分被先定义，然后在每个相继的工作版本中再逐步对其加以完善。演化模型如图 1.3 所示。可以看出，增量模型与演化模型的区别主要在于增量模型一开始就有非常确定的需求；而演化模型的需求是在开发过程中，随着对问题理解的逐步加深，需求得到逐步补充，这比较符合实际的情况，可以说演化模型比增量模型优越。图 1.3 中的虚线表示这些步骤间可能有信息流动。

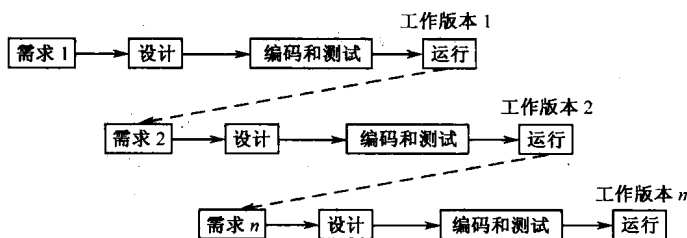


图 1.3 演化模型

4. 快速原型模型

所谓“快速原型”是指快速建立起来的可以在计算机上运行的程序，它所能完成的功能往往是最终的软件产品所能完成的功能的一个子集。用户通过使用这个原型，提出进一步的需求，开发人员按照用户意见快速修改原型，直到得到明确完整的需求，按照这个最后的需求开发出的系统应该能满足用户的真实需要。快速原型模型如图 1.4 所示，其中的实线表示工作顺序，虚线表示有信息流动。从图 1.4 中可以看到快速原型模型是不反馈的。

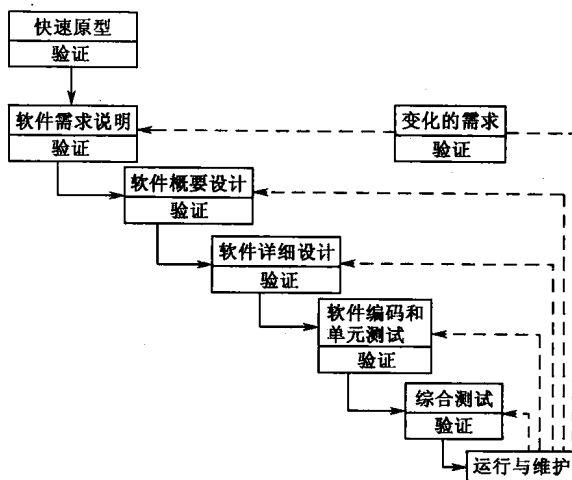


图 1.4 快速原型模型

1.4.3 统一软件开发过程

瀑布模型、增量模型、演化模型、快速原型模型都是与结构化的软件工程方法一起发展起来的，而统一软件开发过程则是与面向对象的软件工程方法一起发展起来的。随着 20 世纪 60 年代结构化编程的不断发展，goto 语句被函数调用所取代，程序的控制流变成顺序、分支、循环三种，从而简化了程序结构。随着软件系统的实现变得越来越复杂，导致了结构化软件工程技术的发展。瀑布模型是结构化软件工程应用最广的一种过程模型。其他几种模型都是后来对瀑布模型的改进。随着软件的规模越来越大，功能越来越复杂，对开发时间的要求越来越短，结构化的软件工程方法无法实现代码重用，导致了面向对象的软件工程的发展，面向对象的软件工程采用的过程模型可以认为是一种综合了瀑布模型、增量模型、演化模型、快速原型模型优点的混合模型，面向对象的软件开发过程仍遵从先分析，再设计，再实现，再测试的开发过程，各个过程间界线不是很严格，各个过程间有很多次反馈和迭代，迭代就是循环以前的过程，但不是简单的重复。像演化模型一样通过多个工作版本的开发来获得完整的用户需求，像增量模型一样，对获得的用户需求分解成相对独立又互相关联的部分来实现，这就是统一软件开发过程模型（Rational Unified Process，RUP）。

统一软件开发过程是在重复一系列组成生存周期的循环，每次循环都向用户提供一个产品。如图 1.5 所示，每次循环都包括四个阶段：初始、细化、构造、移交。每个阶段都经过多次迭代。图 1.5 表示了这些阶段和迭代。阴影的高度表示包含相应工作流的多少。

1.4.4 与模型相应的知识体系的形成

软件开发过程有三大组成部分：概念、表示、实现。

1. 概念

概念就是开发者怎样考虑待解决的问题。概念分为两种：一种是面向过程的结构化软件工程，它用过程（函数、过程和子程序）来刻画系统；另一种是面向对象的面向对象软件工程，它用对象和对象之间的关系来刻画系统。

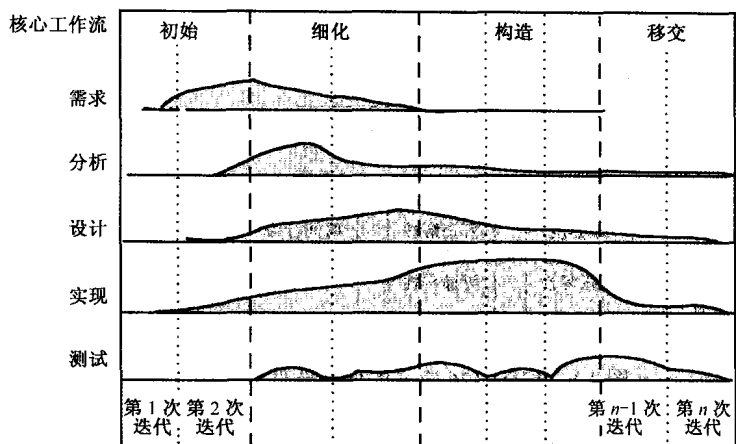


图 1.5 统一软件开发过程中每次循环包含的阶段和迭代

结构化概念主要的思想就是待开发系统（目标系统）可以按功能分解成多个相对独立又相互关联的过程（函数、过程和子程序），每个过程叫一个模块，模块间有信息交换。开发者对目标系统模块的认识和划分可以从最高层的抽象开始，随着对目标系统的了解加深，逐步深入到细节，即采用“逐步求精”的方法进行模块划分和模块设计。

面向对象概念的主要思想就是目标系统是由各种对象组成的，而对象之间是有关系的。开发者利用“逐步求精”的方法，构造出目标系统中的对象和对象之间的关系。

2. 表示

系统的表示又叫模型，就是用某种抽象方法描述待开发系统，如用表格、图、文字等描述系统。构造系统模型的过程叫系统建模。系统建模就是使用几何符号、标记及扩展的符号以一种清楚的、无二义的方式表示系统。

结构化软件工程中生存周期各个阶段采用不同的表示方法，建立各种不同的模型。需求分析阶段用数据流图、数据字典、系统流程图、层次方框图等来表示需求。设计阶段用结构图、HIPO 图、程序流程图、PAD 图、判定树、判定表、过程设计语言等来表示设计思想。所以，结构化方法的各种模型间存在着转换工作量和转换误差。

面向对象的软件工程用统一建模语言 UML 对系统进行全面建模。它在软件生存周期的各个阶段采用相同的表示方法，各种模型间无须转换，所以不存在转换工作量和转换误差。

3. 实现

实现就是指选择适当的程序设计语言来实现目标系统，结构化软件工程采用结构化程序设计语言；面向对象的软件工程采用面向对象的程序设计语言。

1.5 优秀软件的属性

软件工程的最终目标就是为了在已知的适当的资源设备条件下，在限定的时间内获得具有明确意义的高质量的软件。这时有一个重要的概念：什么样的软件才是高质量的软件？在 20 世纪 90 年代早期，国际标准化组织（ISO）提出了一个评估软件质量的标准模型，这个标准模型就是 ISO 9126，如图 1.6 所示。该模型由六个主要特征构成，每个特征都有进一步细