

21世纪高等院校计算机专业规划教材



实用软件工程教程

郭清宇 刘凤华 主编

21 世纪高等院校计算机专业规划教材

实用软件工程教程

主编 郭清宇 刘凤华

参编 李志民 贾晓辉 赵 冬 潘 磊 胡晓旻

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书结合实际软件项目开发讲解软件工程的关键概念,使学生加深对软件工程理论的理解,指导软件工程的实践活动。本书内容包含项目启动、项目计划、需求工程、系统设计、编码及代码审查、软件测试、项目交付和培训、项目实战等部分。

本书以实际项目为主线,并提供源码为参照,适合作为高校本科计算机软件工程、计算机应用技术等专业或软件职业技术学院软件工程和软件实训的教材,也可作为软件开发人员和软件项目管理人员的参考书。

图书在版编目(CIP)数据

实用软件工程教程 / 郭清宇, 刘凤华主编. —北京:
中国铁道出版社, 2011. 5
21 世纪高等院校计算机专业规划教材
ISBN 978-7-113-12723-7

I. ①实… II. ①郭… ②刘… 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 (2011) 第 053123 号

书 名: 实用软件工程教程
作 者: 郭清宇 刘凤华 主编

策划编辑: 周海燕

责任编辑: 周海燕 徐盼欣

封面设计: 付 巍

版式设计: 于 洋

读者热线电话: 400-668-0820

封面制作: 白 雪

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: 北京新魏印刷厂

版 次: 2011 年 5 月第 1 版

2011 年 5 月第 1 次印刷

开 本: 787mm×1092mm 1/16

印张: 22.5 字数: 546 千

印 数: 3 000 册

书 号: ISBN 978-7-113-12723-7

定 价: 33.80 元

版权所有 侵权必究

凡购买铁道版图书, 如有印制质量问题, 请与本社教材研究开发中心批销部联系调换。

软件工程专业学生需要扎实的理论基础,也需要较多的技能和实践经验。目前,软件工程理论方面的书籍很多,项目实践方面的书籍却很少,即使有一些案例教学之类的软件工程书籍,案例也大多是虚拟项目,和实际意义上的软件开发有很大距离。本书以实际项目(均来源于工程实践,已正式交付用户使用)为案例,将其贯穿于项目的软件工程生命周期中,使学生真正了解软件的实际开发过程,通过本书学习和选择一个完整的、难度适中的项目进行模仿、实训,达到提高实际动手能力、缩短就业后适应期的目的。

本书作者从事过多年的软件工程教学工作,大多主持开发过大型项目,有丰富的项目管理和开发经验,针对学生在软件工程实践过程中可能遇到的困难和问题,结合教学和项目开发过程实际编写了本书。

本书的总体编写思路如下:

(1) 全书共分为 10 章:前 8 章着重讲解软件工程的原理,包括项目启动、项目计划、需求工程、系统设计、编码及代码审查、软件测试、项目交付和培训,在讲解软件工程原理的同时引入项目案例;后 2 章详细介绍了两个项目的开发过程,其中第 9 章介绍的职业安全与健康法规标准管理系统是一个基于 B/S 模式的项目,第 10 章河流根石探测管理系统是一个基于 C/S 模式的项目,B/S 模式和 C/S 模式是当前的两种主流体系结构,两个项目均已完成并提交给用户使用。

(2) 每章均分为 4 部分进行编写:引言用于说明本章的主题思想、论述范围、主要内容,以及概略介绍有助于理解本章主题思想的有关内容,为阐述主题作准备。学习目标简明清晰地列出本章学习要达到的目标。主要内容均紧密结合项目工程实际,以工程项目开发流程为写作线索。每章后的小结对本章的内容进行简单回顾,帮助读者总结学习要点和学习重点。

与现有教材相比,本教材具有以下特色:

(1) 基于项目进行编写。本书不仅给出了两个实际工程项目的详细开发过程描述,而且在理论部分的写作过程中也穿插了案例的讲解。

(2) 以项目开发流程为写作线索。本书不仅在章节安排上以工程开发流程为线索,在描述项目开发过程中更是严格按实际开发流程进行,不仅内容真实严谨,而且可操作性强,读者可以一步一步模拟实践,深刻理解软件工程的原理及其应用。

(3) 重点难点突出。本书理论部分的讲解没有罗列详细的知识点,不介绍琐碎的和不太常用的知识,重点讲解最精华的理论知识。

(4) 所有项目资源齐备。本书详细介绍项目环境配置和发布运行的步骤,并提供源码参照。

本书第 1 章由郭清宇和贾晓辉编写;第 2、3、5、8 章由李志民编写;第 4 章由李志民、潘磊编写;第 6、7 章由郭清宇编写;第 9 章第 1~4 节由贾晓辉编写,第 5~8 节由潘磊和胡晓旻编写;第 10 章由刘凤华和赵冬编写,附录由贾晓辉编写。全书由郭清宇和刘凤华统稿并审定。

本书的编写得到中原工学院教材项目资金资助，在本书的编写过程中，豆振海在代码调试及内容组织上做了大量工作；写作过程中，本书参考和引用了很多优秀的文献，在此深表谢意！

虽然我们力求完美，力创精品，但由于水平有限，书中难免有疏漏和错误等不尽如人意的地方，还望广大读者批评指正。

编 者

2011 年 2 月

目 录

CONTENTS >>>

第 1 章 概论	1
1.1 软件和软件危机.....	1
1.1.1 软件的概念和特点	1
1.1.2 软件的发展历程	2
1.1.3 软件危机	3
1.2 软件工程.....	4
1.2.1 软件工程定义.....	4
1.2.2 软件工程的发展历史	4
1.2.3 软件生命周期模型	5
1.2.4 软件过程	6
小结	10
第 2 章 项目启动.....	11
2.1 项目开发流程与管理方式	11
2.1.1 软件项目形式.....	11
2.1.2 项目开发流程.....	12
2.1.3 精简并行过程.....	14
2.2 项目组织结构	15
2.2.1 项目角色	16
2.2.2 项目组织	16
2.3 项目里程碑	17
2.3.1 检查点、里程碑和基线.....	17
2.3.2 里程碑计划与验收	18
2.4 案例分析及描述.....	20
2.4.1 项目介绍	20
2.4.2 项目组织结构.....	21
2.4.3 项目里程碑.....	21
小结	22
习题	22
第 3 章 项目计划.....	23
3.1 项目估计.....	23
3.1.1 项目范围	23
3.1.2 软件规模度量.....	24
3.1.3 项目资源估算.....	25
3.1.4 成本估算	25

3.2 项目计划管理	26
3.2.1 项目进度计划管理	27
3.2.2 项目质量保证	29
3.2.3 项目配置管理	30
3.2.4 项目测试计划	32
3.2.5 风险管理	33
3.2.6 能力成熟度模型	34
3.3 案例分析及描述	35
3.3.1 项目估算	35
3.3.2 项目计划管理	36
小结	38
习题	38
第4章 需求工程	39
4.1 需求捕获	39
4.1.1 需求开发计划和目标	39
4.1.2 实地收集需求	40
4.1.3 软件需求	41
4.2 需求分析	43
4.2.1 原型分析法	43
4.2.2 结构化分析方法	44
4.2.3 用例分析方法	46
4.2.4 RUP 业务建模	47
4.2.5 用例分析	49
4.2.6 系统建模	52
4.2.7 相关案例分析	53
4.3 需求定义	56
4.3.1 需求报告	56
4.3.2 需求规格说明书编写指南	56
4.4 需求管理	59
4.4.1 需求变更控制	59
4.4.2 版本控制	62
4.4.3 需求跟踪	63
4.5 需求验证	64
4.5.1 需求验证的目的和任务	64
4.5.2 需求评审	65
4.5.3 需求测试	66
小结	67
习题	67

第 5 章 系统设计	68
5.1 设计概述	68
5.1.1 设计要素	68
5.1.2 设计原理	69
5.1.3 设计目标	69
5.1.4 软件设计方法	69
5.1.5 面向对象方法的设计原则	70
5.2 体系结构设计	71
5.2.1 软件体系结构建模	71
5.2.2 软件构架文档	72
5.2.3 MVC 架构	74
5.2.4 软件体系结构风格	75
5.2.5 Web 服务体系结构	79
5.3 数据库设计	79
5.3.1 数据库设计过程	79
5.3.2 数据库设计技巧	80
5.4 界面设计	82
5.4.1 界面设计原则	82
5.4.2 用户界面设计的工作流程	83
5.4.3 界面设计规范	84
5.5 案例分析及描述	84
5.5.1 系统架构风格设计	84
5.5.2 设计方法	84
5.5.3 数据库设计	84
5.5.4 界面设计	85
小结	85
习题	85
第 6 章 编码及代码审查	86
6.1 程序设计语言	86
6.1.1 程序设计语言的分类	86
6.1.2 程序设计语言的特性	88
6.1.3 编码语言的选择	90
6.2 编码规约	91
6.2.1 编码规约的目标	91
6.2.2 编码规约的一般性原则	91
6.3 代码审查	92
6.3.1 代码审查的目的和形式	92
6.3.2 代码审查小组和代码审查内容	93

6.3.3	代码审查过程	93
6.3.4	结对编程	95
6.4	案例分析与描述	95
6.4.1	编码语言选择及开发环境	95
6.4.2	编码规约	96
6.4.3	代码审查	103
小结		108
习题		108
第7章	软件测试	109
7.1	概述	109
7.1.1	软件测试的目的	110
7.1.2	软件测试原则	110
7.1.3	软件测试的基本过程	112
7.1.4	软件测试方法	112
7.1.5	软件测试人员的素质要求	116
7.2	软件测试过程	117
7.2.1	软件测试步骤	117
7.2.2	软件测试流程	121
7.2.3	单元测试	121
7.2.4	集成测试	123
7.2.5	确认测试	127
7.3	测试技术	128
7.3.1	白盒测试技术	128
7.3.2	黑盒测试技术	130
7.4	测试用例设计与执行	132
7.4.1	测试用例的基本要素	133
7.4.2	测试用例设计过程	134
7.4.3	基于场景的测试用例设计	135
7.4.4	基于补充规约生成的测试用例	136
7.4.5	测试用例执行过程	138
7.4.6	测试用例管理和维护	139
小结		140
习题		140
第8章	项目交付和培训	141
8.1	项目培训	141
8.1.1	培训需求确定	141
8.1.2	培训计划制定	141
8.1.3	培训计划实施	142

8.1.4 培训效果评估	142
8.2 项目实施	143
8.2.1 项目实施准备	143
8.2.2 模拟运行及用户化	143
8.2.3 制定工作准则与工作规程	143
8.2.4 系统运行	143
8.3 项目验收	143
8.3.1 项目验收准备	144
8.3.2 验收测试	144
8.3.3 项目交付	144
8.4 案例描述	145
8.4.1 项目培训	145
8.4.2 项目实施	145
8.4.3 项目验收	146
小结	146
习题	146
第9章 职业安全与健康法规标准管理系统	147
9.1 项目描述	147
9.1.1 项目背景	147
9.1.2 捕获需求	148
9.1.3 项目计划	148
9.2 需求分析	149
9.2.1 参与者	149
9.2.2 用例	149
9.2.3 系统用例图	150
9.2.4 用例规约	151
9.2.5 分析类	165
9.3 系统设计	172
9.3.1 体系结构设计	172
9.3.2 实体类设计	173
9.3.3 数据库设计	174
9.3.4 界面设计	178
9.4 系统实现	179
9.4.1 开发环境及配置	179
9.4.2 功能模块概要	180
9.4.3 项目测试	201
9.4.4 系统变更	216
9.5 项目交付	220

9.5.1 系统交付	220
9.5.2 用户手册	222
9.5.3 培训	225
小结	226
第 10 章 河流根石探测管理系统	227
10.1 项目描述	227
10.1.1 项目背景	227
10.1.2 业务描述	227
10.1.3 用户描述	231
10.1.4 项目计划	231
10.2 需求分析	234
10.2.1 需求描述	235
10.2.2 用例规约	236
10.2.3 用例图	263
10.3 系统分析与设计	266
10.3.1 领域模型	266
10.3.2 顺序图	269
10.3.3 设计类	286
10.3.4 数据库设计	289
10.3.5 界面设计	296
10.4 系统实现	300
10.4.1 环境配置	300
10.4.2 代码实现	301
10.4.3 测试	316
10.5 发布运行	322
10.5.1 项目发布运行	322
10.5.2 安装说明	325
10.6 xx河道整治工程根石探测管理系统使用说明	326
小结	339
附录 A UML 入门及 Rose 的使用	341
参考文献	349

第 1 章 | 概 论

引言

软件工程是一门指导软件系统开发和维护的工程学科。作为本书的开始，本章简要介绍软件工程的基本概念和原理，以便为下一步的学习打下基础。本章内容主要包括：软件工程的基本概念、软件过程、常见的几种软件开发模型简介。

已经系统学习过软件工程的读者可以跳过本章直接进入下一章的学习。

学习目标

通过本章学习，应掌握以下内容：

- 了解软件的基本概念和特征。
- 了解软件工程的基本概念。
- 了解常用的软件过程模型。

1.1 软件和软件危机

软件在计算机系统中担任着重要的角色。软件本身是一种产品，同时又是开发和运行产品的载体。作为一种产品，软件提供了以各种形式获取和处理信息的手段，它处理大量的个人数据、管理海量的商业信息、提供了通往信息网络的途径。作为开发运行产品的载体，操作系统软件是计算机控制的基础，网络管理和控制软件是信息通信的基础，而各种软件工具和开发环境也提供了创建和控制其他程序的平台。

1.1.1 软件的概念和特点

软件是计算机程序、数据结构和文档的有机结合。其中，计算机程序能够完成预定功能和性能的可执行的指令；数据结构使得程序能够适当地操作各种信息；文档全面描述了程序的分析、设计过程和使用方法。

软件和硬件虽然都是产品，但它们之间有着本质的区别。制造硬件时，人类创造性的过程（分析、设计、建造、测试）最终完全转换成物理的形式。制造一台新的计算机，从初始的草图、正式的设计图样和面板的原型到最后一步步演化成物理的产品，展现在人们面前的是真实存在的计算机。而软件是逻辑的而不是物理的产品，因此软件具有与硬件完全不同的特征：首先，软件和硬件的生产方式不同，软件是由开发或工程化而形成的，而不是传统意义上的制造产生的。虽然良好的设计都是获得高质量的基础，但硬件在制造过程中可能会出现质量问题，对于软件而言，其制造过程却相对容易，只是简单的复制。硬件可以有在规定范围内的误差，而软件不允许有误差。其二，软件不会“磨损”。通常，随着时间的改变，硬件构件由于种种环境原因不断磨损，故

障率会越来越高。软件本身并不受到这些引起硬件磨损的环境因素的影响。第三, 软件维护比硬件维护要复杂得多。当一个硬件构件磨损, 需要维护时, 可以用一个备件去替换, 而软件发生故障, 表明分析、设计或实现过程中存在错误, 必须经过深入的分析、测试才可能找到错误的原因并改正, 而且改正过程中还有可能引发新的错误。最后, 和硬件不同, 大多数软件是用户定制的, 而不是通过已有的构件组装而来的。市场上买到的商业化的通用软件, 都是一个完整的软件, 而不能作为构件再组装成新的程序, 软件复用还远远没有到实际工程使用的范围。

根据软件应用领域不同, 一般把软件分成如下几类:

系统软件: 系统软件是一组为其他程序服务的程序。如编译器、操作系统、驱动程序和通信进程控制等。系统软件与计算机硬件联系紧密, 一般支持多用户, 有复杂的数据结构及多外部接口, 同时, 系统软件还需要处理资源共享、进程管理、并发操作等问题。

实时软件: 管理、分析、控制现实世界中发生的事件的程序称为实时软件。其和交互系统或分时系统的最大区别在于: 实时软件在给定的时间范围内必须对外界环境变化给出正确的响应, 而交互系统或分时系统的响应时间则可以延迟。实时软件一般由数据采集、分析、控制、输出和管理部件几部分组成, 数据采集负责从外部环境获取和格式化信息; 分析部件负责将信息转换成应用时所需要的形式; 控制、输出部件, 负责响应外部环境; 管理部件负责协调其他各部件。

商业软件: 商业软件的目的是处理海量的商业信息, 如财务管理软件、库存管理软件、医院管理软件等。这些软件均属于管理信息系统 (MIS) 软件, 一般情况下, 均以一个或数个数据库为基础, 并能提供辅助商业操作和管理决策的功能。

工程和科学计算软件: 工程和科学计算软件的特征是“数值分析”算法。此类应用包含的范围很广, 几乎所有的自然科学领域都有大量的这类软件在起作用。随着计算机技术的发展, 目前工程和科学计算软件已不仅限于传统的数值算法。计算机辅助设计、系统仿真和其他交互应用已经开始具有实时软件和系统软件的特征。

嵌入式软件: 嵌入式软件是随着各类智能产品发展起来的, 目前已应用到人类生活的各个方面, 如手机、家电的按钮控制。嵌入式软件驻留在只读内存中, 用于控制这些智能产品, 执行有限但专职的功能。

个人计算机软件: 个人计算机软件主要包含字处理、电子表格、计算机图形、多媒体、娱乐、数据库管理、个人及商业金融应用、外部网络或数据库访问等, 它面向的是个人用户而不是一个企业或管理部门。

人工智能软件: 人工智能 (AI) 软件利用非数值算法去处理复杂的问题, 这些问题一般不能通过计算或直接分析得到答案。如基于知识的专家系统、模式识别 (图像或声音)、定理证明、人工神经网络等都是人工智能软件的典型例子。

1.1.2 软件的发展历程

在计算机发展的早期阶段, 软件开发时, 大多不做预先计划, 也没有系统化的开发方式和管理。软件开发基本上是个人的行为, 程序的编写者和使用者往往是同一个或同一组人。一旦计划出现延迟或变更, 也需要同一个人或同一组人员做出弥补, 工作的可变性很低。通用的硬件在当时已经非常普遍, 而软件应用范围却很有限, 几乎每个软件都需要重新设计和开发, 毫无通用性可言。在这种个人化的软件环境中, 软件的分析 and 设计仅仅是程序员的一些思想, 几乎没有成型、规范的设计和使用文档。

计算机系统发展的第二阶段跨越了从20世纪60年代中期到70年代末期的十余年。这个时期软件产品开始使用,软件开发的主体从个人开始向“软件作坊”转换。一些工作在主机和微机上的程序,已经开始商业化,甚至可能拥有上千用户。软件已经开始作为一个产业受到企业家们关注,并开始向软件业投资。从计算机技术的角度看,这个时期开始出现多用户系统和初期的数据库系统,实时系统也得到了广泛的应用。

随着计算机技术的发展,计算机软件库开始扩展,软件数量开始飞速增长,随之而来的软件维护问题开始显现。程序中发现错误时需要纠正这些错误;用户有了新需求时需要扩充软件功能;硬件环境变化时软件也需要随之变化以适应新的硬件环境。所有这些在软件维护上所花费的工作占用了惊人的资源,大量由个人开发的程序由于缺乏必要的文档根本无法维护,“软件危机”开始出现。

计算机系统发展的第三阶段始于20世纪70年代中期并跨越了整整十年。这个时期,计算机的硬件成本大大降低,分布式系统的出现极大地提高了计算机系统的复杂性,网络和通信技术快速发展,微处理器出现并得到广泛应用。

计算机系统发展的第四阶段,计算机体系结构迅速地从集中的主机环境转变为分布的客户机/服务器环境;世界范围的信息网基本结构开始形成;软件产业在世界经济中的地位快速上升;大量的新技术开始涌现;面向对象技术在许多领域中迅速取代了传统软件开发方法。

1.1.3 软件危机

从计算机系统发展的第二阶段开始,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就一直在困扰着管理人员和软件开发人员,而且这些问题随着计算机的普及和发展愈发严重,这一系列问题被称之为软件危机。软件危机主要包含两方面的问题:一是如何开发软件,以满足对软件日益增长的需求;二是如何维护数量不断膨胀的已有软件。

软件危机的主要表现有:

硬件的发展远远超前软件,软件难以发挥硬件的所有潜能;

人们开发新程序的能力远远不能满足社会对新程序的需求,同时开发新程序的速度也不能满足商业和市场的要求;

软件开发的成本越来越高,在计算机系统中,软件成本已占到了绝大部分;

软件的可靠性随着软件的复杂性提高日益下降,而计算机的普遍使用已使得社会越来越依赖于可靠的软件;如果软件失败,会造成巨大的经济损失,甚至有可能给人类带来灾难;

软件开发的成本和进度往往估计错误,有些甚至无法做出预计;

对已有软件的维护非常困难;很多软件缺乏必要的、规范的、正确的文档;

已开发的软件得不到用户的认同。

在软件开发和维护的过程中存在这么多严重问题,一方面与软件本身的特点有关,另一方面也和软件开发人员开发与维护的方法不正确有关。首先软件是个逻辑实体,而不是物理实体,具有抽象性,因此管理和控制软件都很困难;同时,随着社会发展,软件规模越来越大,结构越来越复杂,大规模的软件开发和管理非常复杂。其次,软件危机是由于软件开发和维护的方法不正确引起的,软件生产的知识密集和人力密集的特点是造成软件危机的根源所在。软件开发过程中,技术落后、生产方式仍采用手工方式,软件开发成功与否往往取决于软件人员的受教育程度和经验的积累;随着计算机应用领域的扩大,绝大多数软件已不再是数值计算问题,而是无法精确描述、多变的应用问题,软件开发人员对用户需求的理解与用户原意有差异,而很多开发人员对用户

要求没有完整准确认识就匆忙着手编写程序,只重视程序而忽视软件配置其余成分,从而造成后期维护工作量大大增加,甚至软件开发失败;大型软件往往要多人合作开发才能完成,需要在用户与软件人员之间、软件开发人员之间相互通信,由于此过程中经常会发生理解差异和彼此之间交流问题,从而导致设计或实现中出现错误;计算机技术和应用发展迅速,知识更新周期加快,软件开发人员经常处在变化之中,不仅需要适应硬件更新的变化,几乎每一项软件开发都必须调整自身的知识结构以适应新的问题求解的需要,而这种调整是人所固有的学习行为,难以用工具来代替。

为了消除软件危机,人们提出了各种各样的方法和途径,并不断在实践中加以完善。在软件开发过程中,首先要更新观念。必须认识到,软件开发不是某种个体劳动的神秘技巧,而应该是一种组织良好、管理严密、各类人员协同配合、共同完成的工程项目,因此必须采用工程化的方法和途径来开发与维护软件;同时,应该汲取系统工程的理念,采用必要的组织管理和技术管理措施,而这种组织和管理措施的正确与否往往决定了软件开发的成败;软件开发还应该使用更好的软件工具,提高软件生产率,不断地发现、完善、推广、使用人们在实践中总结出来的开发软件的成功技术和方法。

1.2 软件工程

1.2.1 软件工程定义

1968年在德国举行的NATO软件工程会议上,为应对“软件危机”的挑战,提出了“软件工程”的术语。这个时期有代表性的软件工程定义是“为了经济地获得在真实机器上可靠工作的软件而制定和使用的合理工程原则和方法”。

1972年IEEE的计算机学会第一次出版了《IEEE软件工程学报》。此后,“软件工程”这个术语被广泛用于工业、政府和学术界,众多的出版物、团体和组织、专业会议在它们的名称里使用“软件工程”这个术语,很多大学的计算机科学系先后开设了软件工程课程。

软件工程是为了理清软件工程过程的各种活动,从而提出软件生命周期的概念和软件开发的瀑布模型,进而制定的软件生命周期中主要活动的质量标准。这个时期的软件工程定义特别强调,软件工程以计算机科学和数学为基础,用系统的、可控制的、有效的方式建造高质量的软件。有代表性的定义包括:

定义1:软件工程是为了经济地获得能够在实际机器上高效运行的可靠软件而建立和使用的一系列工程化原则。

定义2:软件工程是以工程的形式应用计算机科学和数学原理,从而经济有效地解决软件问题。

定义3:软件工程是将系统性的、规范化的、可量化的方法应用于软件的开发、运行和维护。

定义4:采用工程的概念、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,以经济地开发出高质量的软件并有效地维护它。

1.2.2 软件工程的发展历史

1. 传统软件工程

“传统软件工程”是指1968年德国NATO会议之后产生的、以瀑布模型和结构化分析及设

计为代表的一系列软件工程方法，其主要特征是强调生命周期前期的需求分析，不鼓励后期的需求变化。传统软件工程又称为经典软件工程，其以结构化程序设计为基础，一般开发过程具有如下阶段：结构化分析—结构化设计—面向过程的编码—软件测试。

2. 面向对象软件工程

自 20 世纪 80 年代中期起，从注重于面向对象分析和设计的研究开始，逐步形成了面向对象方法学。面向对象方法的出现很快受到计算机软件界的青睐，并成为 20 世纪 90 年代的主流开发方法。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（OOA）、面向对象的设计（OOD）、面向对象的编程（OOP）、面向对象的测试（OOT）和面向对象的维护等主要内容。

3. 基于构件的软件工程

基于构件的软件开发技术是兴起于 20 世纪 90 年代的一种软件开发新模式，主要是在一定构件模型的支持下，复用构件库中的软件构件，通过高效率的构件组合，高质量的构造应用软件系统的过程，以软件复用为目标，以领域工程为基础，开发过程如下：领域分析和测试计划—领域设计—可复用的构件库—构件集成。

1.2.3 软件生命周期模型

跟其他事物一样，软件也同样有孕育、诞生、成长、成熟、衰亡的过程，我们称之为软件生命周期（systems development life cycle, SDLC）。概括地说，软件生命周期由软件定义、软件开发、软件维护三个时期组成。

定义阶段的主要任务是解决“做什么”的问题。确定软件的目标和可行性；导出软件实现的策略及功能；估计软件需要的资源和成本；制定软件开发的进度表。通常被分为三个阶段：问题定义、可行性研究、需求分析。

开发阶段的任务是解决“如何做”的问题。通常包括软件设计、实现、测试等工作。

维护阶段的主要任务是使软件保持用户满意的需要。

将软件生命周期划分为若干阶段，每个阶段有明确的任务，可以使规模大、结构复杂和管理复杂的软件开发变得容易控制和管理，是实现软件生产过程化的重要步骤。下面简要描述各个阶段的工作：

问题的定义：此阶段是软件开发方与需求方共同讨论，主要确定软件的开发目标及其可行性。

可行性分析：可行性分析是通过对项目的主要内容和配套条件，如市场需求、资源供应、建设规模、工艺路线、设备选型、环境影响、资金筹措、盈利能力等，从技术、经济、工程等方面进行调查研究和比较，并对项目建成以后可能取得的经济效益及社会环境影响进行预测，从而提出该项目是否值得投资和如何进行建设的咨询意见，为项目决策提供依据的一种综合性的系统分析方法。可行性分析应具有预见性、公正性、可靠性、科学性的特点。

需求分析：在确定软件开发可行的情况下，对软件需要实现的各个功能进行详细分析。需求分析阶段是一个很重要的阶段，这一阶段做得好，将为整个软件开发项目的成功打下良好的基础。“唯一不变的是变化本身”，同样需求也是在整个软件开发过程中不断变化和深入的，因此我们必须制定需求变更计划来应对这种变化，以保护整个项目的顺利进行。

软件设计：此阶段主要根据需求分析的结果，对整个软件系统进行设计，如系统框架设计，数据库设计等。软件设计一般分为总体设计和详细设计。好的软件设计将为软件程序编写打下良

好的基础。

程序编码：此阶段是将软件设计的结果转换成计算机可运行的程序代码。在程序编码中必须要制定统一、符合标准的编写规范。以保证程序的可读性、易维护性，提高程序的运行效率。

软件测试：在软件设计完成后要经过严密的测试，以发现软件在整个设计过程中存在的问题并加以纠正。整个测试过程分单元测试、组装测试以及系统测试三个阶段进行。测试的方法主要有白盒测试和黑盒测试两种。在测试过程中需要建立详细的测试计划并严格按照测试计划进行测试，以减少测试的随意性。

运行维护：软件维护是软件生命周期中持续时间最长的阶段。在软件开发完成并投入使用后，由于多方面的原因，软件不能继续适应用户的要求。要延续软件的使用寿命，就必须对软件进行维护。软件的维护包括纠错性维护和改进性维护两个方面。

有时也把软件生存周期划分为软件计划、软件开发和软件维护三个大的阶段，其中软件计划阶段包括问题定义、可行性研究，软件开发阶段包括需求分析、总体设计、详细设计、编码和测试。

实际从事软件开发时，软件规模、种类、开发环境及开发时所使用的技术方法等因素，都影响阶段的划分。

1.2.4 软件过程

对软件生命周期的研究导致了软件过程的诞生，软件过程即围绕软件开发所进行的一系列活动。

1. 从软件生命周期到软件过程模型

1976年，《IEEE 计算机学报》刊登 B.W.Boehm 的文章，建议将软件生命周期划分为问题定义、可行性研究、系统需求分析、概要设计、详细设计、编码、测试、运行维护等阶段。

早期软件工程中，软件开发过程与软件生命周期常常不加区分，比如典型的软件生命周期用来表示典型瀑布模型的过程和活动，将生命周期和软件过程同等对待，这是生命周期和软件过程的研究相结合的早期实例。

2. 软件过程的演变

随着时间推移，人们发现，线性开发模型不能适应大型复杂系统的开发。由于大型的系统开发周期长，引发需求改变的因素太多，需求的更改在线性模型应用中遇到了困难，正是由于这些变化，促使了软件开发模型的演变，陆续涌现了一些迭代、形式化等新的开发模型，下面将依次介绍。

(1) 传统的软件过程

① **瀑布模型 (waterfall model)。**瀑布模型是一种基于软件生命周期的线性开发模型，如图 1-1 所示，其由 Royce 于 1970 年首先提出。瀑布模型中至关重要的一点是，只有当一个阶段的文档已经编制好并获得软件质量保证小组的认可，才可以进入下一个阶段。

瀑布模型通过强制性的要求提供规约文档来确保每个阶段都能很好地完成任务，属于传统的结构化开发模式，其有下述三个特点：

阶段间具有顺序性和依赖性：顺序性即只有前一阶段的工作完成后，才开始后一阶段的工作；依赖性即前一阶段的输出文档是后一阶段的输入文档。