

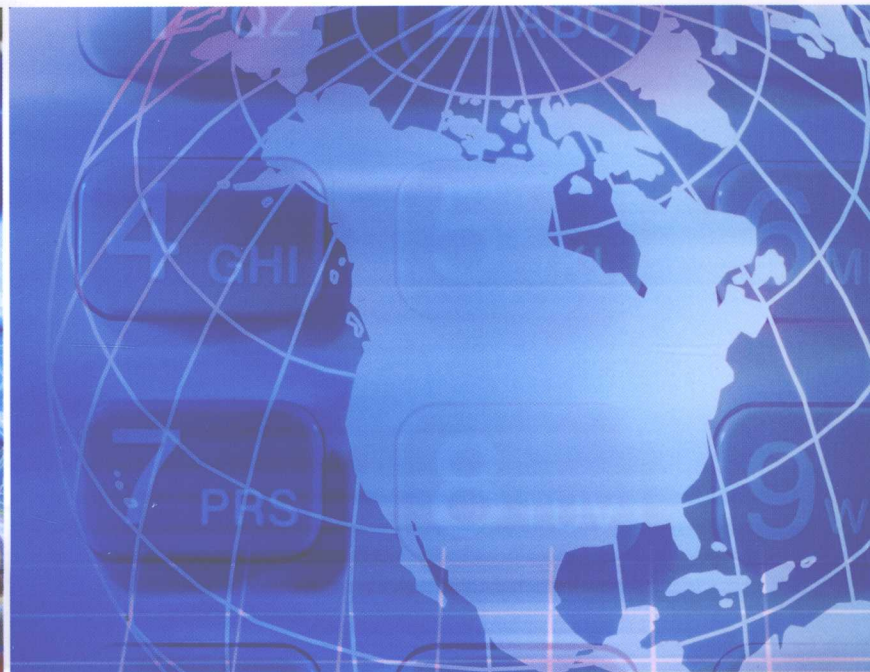


软件职业技术学院“十一五”规划教材

软件开发流程

实训教程

主 编 杜文洁 景秀丽
副主编 胡万德 郑学伟



中国水利水电出版社
www.waterpub.com.cn

软件职业技术学院“十一五”规划教材

软件开发流程实训教程

主 编 杜文洁 景秀丽

副主编 胡万德 郑学伟



中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书详细介绍了软件开发流程基础知识及其相关实训项目, 内容包括软件开发项目的前期准备、目标系统的可行性分析、需求分析、概要设计、详细设计、编码、软件测试、软件实施以及软件维护。

本书采用现在流行的 VB.NET 网络编程语言和 ASP.NET 网站技术设计章节实例, 从应用的角度出发, 采用“任务驱动”方式表述实训内容, 注重实训项目的先进性、科学性和实用性, 力求反映软件开发的系统流程。将软件开发的基础理论与实训项目紧密结合, 使读者可以更好地理解和掌握软件开发过程, 并迅速地运用到实际软件开发工作中。

本书可作为高等院校、高职高专院校及相关软件学院软件技术专业 and 计算机相关专业的教材, 也可作为企业公司软件开发的培训教程, 还可作为软件开发人员的参考资料。

本书配有电子教案, 读者可以到中国水利水电出版社网站或万水书苑免费下载, 网址: <http://www.waterpub.com.cn/softdown/>或 <http://www.wsbookshow.com>。

图书在版编目 (C I P) 数据

软件开发流程实训教程 / 杜文洁, 景秀丽主编. —
北京: 中国水利水电出版社, 2009

软件职业技术学院“十一五”规划教材
ISBN 978-7-5084-6809-9

I. ①软… II. ①杜… ②景… III. ①软件开发—高等学校: 技术学校—教材 IV. ①TP311.138

中国版本图书馆CIP数据核字(2009)第161734号

策划编辑: 石永峰 责任编辑: 张玉玲 加工编辑: 陈欣 封面设计: 无极书装

书 名	软件职业技术学院“十一五”规划教材 软件开发流程实训教程
作 者	主 编 杜文洁 景秀丽 副主编 胡万德 郑学伟
出版发行	中国水利水电出版社 (北京市海淀区玉渊潭南路1号D座 100038) 网址: www.waterpub.com.cn E-mail: mchannel@263.net (万水) sales@waterpub.com.cn 电话: (010) 68367658 (营销中心)、82562819 (万水)
经 售	全国各地新华书店和相关出版物销售网点
排 版	北京万水电子信息有限公司
印 刷	北京市天竺颖华印刷厂
规 格	184mm×260mm 16开本 13.75印张 340千字
版 次	2009年8月第1版 2009年8月第1次印刷
印 数	0001—4000册
定 价	24.00元

凡购买我社图书, 如有缺页、倒页、脱页的, 本社营销中心负责调换

版权所有·侵权必究

序

随着信息技术的广泛应用和互联网的迅猛发展，以信息产业发展水平为主要特征的综合国力竞争日趋激烈，软件产业作为信息产业的核心和国民经济信息化的基础，越来越受到世界各国的高度重视。中国加入世贸组织后，必须以积极的姿态，在更大范围和更深程度上参与国际合作和竞争。在这种形势下，摆在我们面前的突出问题是人才短缺，计算机应用与软件技术专业领域技能型人才的缺乏尤为突出，无论是数量还是质量，都远不能适应国内软件产业的发展和信息化建设的需要。因此，深化教育教学改革，推动高等职业教育与培训的全面发展，大力提高教学质量，是迫在眉睫的重要任务。

2000年6月，国务院发布《鼓励软件产业和集成电路产业发展的若干政策》，明确提出鼓励资金、人才等资源投向软件产业，并要求教育部门根据市场需求进一步扩大软件人才培养规模，依托高等学校、科研院所，建立一批软件人才培养基地。2002年9月，国务院办公厅转发了国务院信息化工作办公室制定的《振兴软件产业行动纲要》，该《纲要》明确提出改善软件人才结构，大规模培养软件初级编程人员，满足软件工业化生产的需要。教育部也于2001年12月在35所大学启动了示范性软件学院的建设工作，并于2003年11月启动了试办示范性软件职业技术学院的建设工作。

示范性软件职业技术学院的建设目标是：经过几年努力，建设一批能够培养大量具有竞争能力的实用型软件职业技术人才的基地，面向就业、产学结合，为我国专科层次软件职业技术人才培养起到示范作用，并以此推动高等职业技术教育人才培养体系与管理体制和运行机制的改革。要达到这个目标，建立一套适合软件职业技术学院人才培养模式的教材体系显得尤为重要。

高职高专的教材建设已经走过了几个发展阶段，由最开始本科教材的压缩到加大实践性教学环节的比重，再到强调实践性教学环节，但是学生在学习时还是反映存在理论与实践的结合问题。为此，中国水利水电出版社在经过深入调查研究后，组织了一批长期工作在高职高专教学一线的老师，编写了这套“软件职业技术学院‘十一五’规划教材”，本套教材采用项目驱动的方法来编写，即全书所有章节都以实例作引导来说明各知识点，各章实例之间并不是孤立的，每个实例都可以作为最终项目的一个组成部分；每一章章末还配有实习实训（或叫实验），这些实训组合起来是一个完整的项目。

采用这种方式编写的图书与市场上同类教材相比更具优越性，学生不仅仅学到了知识点，还通过项目将这些知识点连成一条线，开拓了思路，掌握了知识，达到了面向岗位的职业教育培训目标。

本套教材的主要特点有：

- (1) 课程主辅分明——重点突出，教学内容实用。
- (2) 内容衔接合理——完全按项目运作所需的知识体系结构设置。

(3) 突出实习实训——重在培养学生的专业能力和实践能力，力求缩短人才与企业间的磨合期。

(4) 教材配套齐全——本套教材不仅包括教学用书，还包括实习实训材料、教学课件等，使用方便。

本套教材适用于广大计算机专业和非计算机专业的大中专院校的学生学习，也可作为有志于学习计算机软件技术与开发的工程技术人员的参考教材。

编委会

2006年7月

前 言

现阶段，国内软件工程教程的开发已经处于一个非常成熟的阶段，但是针对具体软件开发流程的实训教程还比较少，处于起步阶段。我们依据高职高专软件工程学科教学大纲所规定的教学要求编写了本教程，把软件开发理论基础和实践经验溶入到教程中，按照理论与实践相结合、突出实践的思路设计教程内容。

本教程注意把握高职高专学生的专业知识背景与接受能力，阐明了软件开发流程的基本概念、基本思路，重点着眼于实际应用，围绕软件生命周期的各个阶段设计相关的实训项目。本教程以学生设计思路为本，通过实训项目培养学生运用科学化、系统化的软件开发方法解决实际问题的能力，力求做到理论知识简练化、体系化，实例阐述具体化、应用化。

“图书馆书目查询管理系统”是贯穿全书的实例，让读者感受到实践知识的连贯性。“图书馆书目查询管理系统”选择 ASP.NET 这样比较流行、实用性强的网络编程技术，采用 VB.NET 作为网络编程语言，使学生易于理解和掌握实例内容，加强学生应用意识、兴趣、能力的培养。

本教程的宗旨是让学生善于运用软件开发的相关技术设计出有效的实例，最终掌握软件开发的整个生命周期、完成软件产品的整个设计流程。在教程的结构安排上，各个章节根据需要安排了“本章引言”、“学习目标”、“基础理论”、“实训项目”等内容，把理论和实训有机地结合在一起，目的就是要达到层次清楚、循序渐进、理论和实践相呼应的效果。在每章内容后面均附有小结、习题和课后实训题。

本教程共分 9 章，系统介绍软件开发流程的主要内容及其相关实训，具体分布如下：

第 1 章软件开发项目的前期准备：介绍软件相关概念、软件生命周期、软件开发常用模型、软件开发方法和工具。

第 2 章可行性分析：系统介绍可行性分析的任务、要素和步骤，以及可行性分析报告的编写。

第 3 章需求分析：介绍需求分析的任务、需求分析的过程、数据流程图、数据字典、需求规格说明书，以及相关实训项目。

第 4 章概要设计：系统描述软件概要设计的任务、软件概要设计的实施、软件概要设计说明书的内容，以及“图书馆书目查询管理系统”的概要设计过程。

第 5 章详细设计：介绍软件详细设计的任务和“图书馆书目查询管理系统”的详细设计过程。

第 6 章编码：介绍编码风格、编码方法、源代码的调试过程和调试方法，并且详细叙述“图书馆书目查询管理系统”的代码编写情况。

第 7 章软件测试：详细介绍软件测试流程和方法、软件测试与软件开发的关系、软件测试方法、黑盒测试和白盒测试的相关方法、软件测试流程相关方法、测试用例设计，以及相关实训项目。

第 8 章软件实施：叙述系统实施概述，系统实施过程，系统实施原则，以及相关实训项目。

第 9 章软件维护：介绍软件维护基本理论、软件维护活动的内容，以及相关实训项目。

本书由杜文洁、景秀丽任主编，胡万德、郑学伟任副主编，第1章、第2章由郑学伟编写，第3章、第7章、第9章、附录A由景秀丽编写，第4~6章和附录B由胡万德编写，第8章由杜文洁编写。另外，参加本书部分编写工作的还有王茹、马岩、丛国凤等。全书由杜文洁统稿。

由于时间仓促及编者水平有限，书中难免出现一些疏漏甚至错误，恳请广大读者批评指正。

作者

2009年6月

目 录

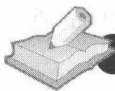
序	
前言	
第1章 软件开发项目的前期准备	1
1.1 计算机软件概述	1
1.1.1 计算机软件概念	1
1.1.2 软件的分类	3
1.1.3 软件工程的历史及发展	4
1.2 软件的生存周期	7
1.2.1 软件生存周期及各阶段的工作	7
1.2.2 软件生存周期各阶段的文档编制	12
1.3 软件开发常用模型	15
1.3.1 瀑布模型	15
1.3.2 增量模型	16
1.3.3 螺旋模型	17
1.3.4 变换模型	18
1.3.5 喷泉模型	19
1.3.6 智能模型	20
1.3.7 WINWIN 模型	20
1.3.8 原型实现模型	21
1.3.9 RAD 模型	22
1.3.10 并发开发模型	23
1.3.11 软件开发常用模型小结	24
1.4 软件开发方法和软件开发工具	25
1.4.1 软件开发环境	25
1.4.2 软件开发方法综述	26
1.4.3 软件开发工具	28
本章小结	28
习题 1	29
课后实训题	29
第2章 可行性分析	31
2.1 可行性分析的任务	31
2.2 可行性分析的要素	32
2.2.1 技术可行性分析	32
2.2.2 经济可行性分析	33
2.2.3 社会可行性分析	34
2.2.4 法律方面的可行性	34
2.3 可行性分析的步骤	34
2.4 可行性分析报告	35
2.4.1 可行性分析报告的格式	35
2.4.2 可行性分析报告范例	39
本章小结	46
习题 2	46
课后实训题	46
第3章 需求分析	48
3.1 需求分析的任务	48
3.1.1 需求分析的基本概念	48
3.1.2 需求分析的重要性	49
3.1.3 需求分析的任务	50
3.2 需求分析的过程	51
3.2.1 需求分析的过程	51
3.2.2 需求分析的注意事项	53
3.2.3 需求风险	56
3.3 数据流程图	57
3.3.1 数据流程图相关图示	57
3.3.2 数据流程图的设计	59
3.3.3 分层数据流图	60
3.4 数据字典	61
3.5 需求规格说明书	63
本章小结	74
习题 3	76
课后实训题	76
第4章 概要设计	77
4.1 软件概要设计的任务	77
4.2 软件概要设计的实施	79
4.2.1 面向对象设计	79
4.2.2 数据库设计	84

4.2.3 用户界面设计	92	本章小结	141
4.2.4 代码设计	94	习题 6	142
4.3 概要设计说明书	95	课后实训题	142
4.3.1 引言	95	第 7 章 软件测试	143
4.3.2 任务概述	96	7.1 软件测试概述	143
4.3.3 总体设计	97	7.2 软件测试与软件开发流程的关系	144
4.3.4 数据结构设计	97	7.3 软件测试方法划分	145
4.3.5 运行设计	100	7.4 黑盒测试的相关方法	149
4.3.6 出错处理设计	100	7.5 白盒测试的相关方法	151
4.3.7 安全保密设计	100	7.6 软件测试流程的相关方法	152
4.3.8 维护设计	100	7.7 测试用例设计	160
本章小结	100	本章小结	181
习题 4	101	习题 7	182
课后实训题	101	课后实训题	182
第 5 章 详细设计	102	第 8 章 软件实施	183
5.1 软件详细设计的任务	102	8.1 系统实施概述	183
5.1.1 软件详细设计的基本任务	102	8.2 系统实施过程	184
5.1.2 软件详细设计的原则	103	8.3 系统实施原则	187
5.2 图书馆书目查询管理系统的设计	105	本章小结	189
5.2.1 系统功能模块结构	105	习题 8	191
5.2.2 系统详细设计	107	课后实训题	191
本章小结	124	第 9 章 软件维护	192
习题 5	125	9.1 软件维护基础理论	192
课后实训题	125	9.2 软件维护活动的内容	195
第 6 章 编码	126	本章小结	202
6.1 建立编码风格	126	习题 9	203
6.2 编码方法	133	课后实训题	203
6.2.1 传统(结构化)编程方法	133	附录 A Visual Studio 2008 的安装	204
6.2.2 面向对象的程序设计方法	136	附录 B IIS 的安装与配置	209
6.3 调试过程和方法	138	参考文献	212

第 1 章



软件开发项目的前期准备



本章引言

软件开发在整个计算机软件生存周期中是最重要的一环。如何高效、高质地开发成功的软件产品是软件工程师们一直追求的目标。经过半个多世纪的研究与探索，人们在计算软件开发领域已经积累了丰富的经验与成果。



教学目标

1. 掌握计算机软件的概念与计算机软件的基本分类。
2. 掌握软件生存周期划分与相关档案的内容。
3. 掌握常用软件开发模型。
4. 掌握计算机软件开发方法。



基础理论和实例

1.1 计算机软件概述

软件是一个完整的计算机系统中与硬件相互依存的一部分，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执行的指令序列；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资料。

1.1.1 计算机软件概念

软件是计算机系统中与硬件相互依存的一部分，它是包括程序、数据及其相关文档的完整集合。计算机软件，是指计算机程序以及解释和指导使用程序的文档的总和。计算机程序包括源程序和目标程序。同一程序的源程序和目标程序应当视为同一作品。源程序是指用高级语言或汇编语言编写的程序；目标程序是指源程序经编译或解释加工以后可以由计算机直接执行的程序。所谓文档，是指用自然语言或者形式化语言编写的文字资料和图表，用来描述程序的内容、组成、设计、功能规格、开发情况、测试结果及使用方法。

软件是一种 20 世纪后半叶才出现的特殊商品（产品），本身就有很多同以往商品（产品）不一

样的特性，所以它的开发（制造）过程也有很多需要注意的地方。软件开发的特点是：

（1）由于软件并不是实际存在的物理形态，如办公桌、书籍、汽车等，它是一种抽象的逻辑过程，是一种逻辑实体，这个特点使它与其他工程对象有着明显的差异。软件不具备常见的物理特性，而具有一个非常重要的特性，即抽象性。人们可以把它记录在纸、内存、硬盘和光盘上，但却无法看到软件本身的形态，必须通过观察、分析、思考、判断才能了解它的功能、性能等特性。

（2）软件的生产过程也非常特殊，同传统意义上普通工厂里的加工不同的是，软件更像是一种纯粹的脑力劳动，同第一个特点一样它没有明显的物理加工过程。所以如果要加强对软件的质量控制，必须在软件开发期间下功夫。一旦研制开发成功，就可以大量拷贝同一内容的副本。

（3）由于软件产品其实是一种抽象的逻辑存在，所以在软件的运行和使用期间，没有硬件那样的机械磨损、设施老化问题。任何物理设备在运行和使用过程中，都有一个大致趋势相同的失效率曲线，即如图 1-1（a）所示的 U 形曲线（即浴盆曲线）。从这个曲线可以看出，随着使用时间的增加设备的失效率主要是由磨损用坏造成的。而软件的情况与此不同，由于其固有的特性，本身不存在磨损和老化问题，即便是它的载体出现 U 形曲线，软件本身仍然可以便利地移到另一台载体上。然而与物理设备不同的是它存在退化问题，随着时间的增加，软件本身会出现落伍的现象，因此必须要对软件本身进行一次或多次修改（维护），如图 1-1（b）所示。而这些修改又不可避免地引入错误，导致软件失效率升高，从而软件退化。当修改的成本变得难以接受时，软件就被抛弃。

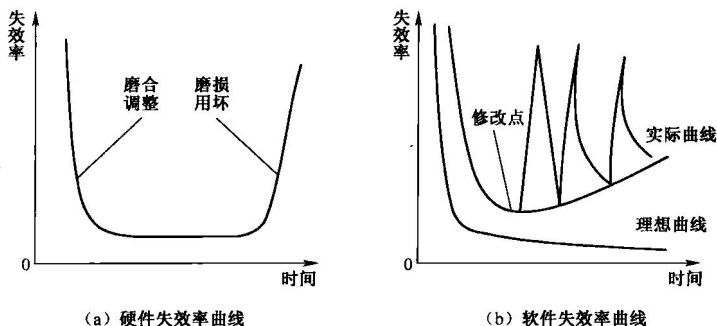


图 1-1 计算机软件失效率曲线

（4）软件的开发和运行不是完全由软件本身的技术实现的，它常常还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。软件对硬件和环境有着不同程度的依赖性，这导致了软件移植的问题。

（5）软件本身是一件结构非常复杂的产品。软件的复杂性来自于两个方面：一是它所反映的实际问题的复杂性（即需要解决问题的复杂性）；二是软件本身设计程序逻辑结构的复杂性。

（6）软件的成本问题。软件开发成本主要指软件开发过程中所花费的工作量及相应的代价。不同于传统的工业产品，软件的成本不包括原材料和能源的消耗，主要是人的劳动消耗。另外，软件也没有一个明显的制造过程，它的开发成本是以一次性开发过程所花费的代价来计算的。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，它的成本是比较高的。

（7）软件开发出来是要给“人”使用的，所以软件不可避免地要具备人的社会属性。相当多的软件工作涉及社会因素。许多软件的开发和运行涉及机构、体制及管理方式等问题，甚至涉及人的观念和人们的心理。这些问题都将直接影响软件开发的成败。

1.1.2 软件的分类

软件是计算机的灵魂，没有软件的计算机就如同没有磁带的录音机和没有录像带的录像机一样，与废铁没什么差别。使用不同的计算机软件，计算机可以完成许许多多不同的工作，它使计算机具有非凡的灵活性和通用性。也正是这一原因，决定了计算机的任何动作都离不开由人安排的指令。

1. 按软件的功能划分

一般把软件按功能分为两大类：应用软件和系统软件。

(1) 应用软件。应用软件是专门为某一应用目的而编制的软件，较常见的有以下几种：

- 文字处理软件：用于输入、存储、修改、编辑、打印文字资料等，如 Word、WPS 等。
- 信息管理软件：用于输入、存储、修改、检索各种信息。
- 辅助设计软件：用于高效地绘制、修改工程图纸，进行设计中的常规计算，帮助寻求好的设计方案。
- 实时控制软件：用于随时搜集生产装置、飞行器等的运行状态信息，以此为依据按预定的方案实施自动或半自动控制，安全、准确地完成任务。

(2) 系统软件。各种应用软件，虽然完成的工作各不相同，但它们都需要一些共同的基础操作，例如都要从输入设备取得数据、向输出设备送出数据、向外存写数据、从外存读数据、对数据进行常规管理等。这些基础工作也要由一系列指令来完成。人们把这些指令集中组织在一起形成专门的软件，用来支持应用软件的运行，这种软件称为系统软件。

系统软件在为应用软件提供上述基本功能的同时，也进行着对硬件的管理，使在一台计算机上同时或先后运行的不同应用软件有条不紊地合用硬件设备。例如，两个应用软件都要向硬盘存入和修改数据，如果没有一个协调管理机构来为它们划定区域，那么必然形成互相破坏对方数据的局面。

代表性的系统软件有：

- 操作系统：管理计算机的硬件设备，使应用软件能方便、高效地使用这些设备。在微机上常见的有：DOS、Windows、UNIX、OS/2 等。
- 数据库管理系统：有组织地、动态地存储大量数据，使人们能方便、高效地使用这些数据。现在比较流行的数据库管理系统有 FoxPro、DB 2、Access、SQL Server 等。
- 编译软件：CPU 执行每一条指令都只完成一项十分简单的操作，一个系统软件或应用软件要由成千上万甚至上亿条指令组合而成。为了提高效率，人们规定一套新的指令，称为高级语言，其中每一条指令完成一项操作，这种操作相对于软件总的功能而言是简单而基本的，而相对于 CPU 的一秒操作而言又是复杂的。因为 CPU 并不能直接执行这些新的指令，需要编写一个软件，专门用来将源程序中的每条指令翻译成一系列 CPU 能接受的基本指令（也称机器语言）使源程序转化成能在计算机上运行的程序。完成这种翻译的软件称为高级语言编译软件，通常把它们归入系统软件。

2. 按软件规模划分

根据所完成任务的不同以及开发任务、投入资金等因素，软件也可以按照规模来划分。

(1) 微型软件：这类软件一般只需要一个人几天即可完成，软件只有几 K 到几十 K 大小。由于规模很小，软件测试也极为容易。

(2) 小型软件：这类软件一般由十人内在半年内完成，开发前需要进行完整的开发设计。

软件一般有几百 K 到十几 M。完成时需要进行软件测试工作，但测试工作相对比较容易。

(3) 中型软件：中型软件一般由十几个人甚至几十个人开发，开发工作比较复杂，时限为半年至一年。软件大小一般有几十 M。对于这种规模的软件，软件测试是十分必要的。要进行严格的可行性分析、开发设计等工作。

(4) 大型软件：这类软件一般由几十人开发，开发工作十分复杂，时限为一年甚至两年以上。软件大小为上百 M 甚至几百 M。这类软件开发前要进行详细的可行性分析、需求分析、开发设计等工作，软件测试要贯穿于开发全过程。这类软件交付使用后往往还要进行一次甚至多次维护升级工作。

(5) 超大型软件：这类软件开发人员一般超过百人甚至几百人，开发工作十分艰辛，开发时限一般在两年以上，这类软件小则几百 M，大则甚至几 G。这类软件的开发十分复杂，要进行详细的可行性分析、需求分析、开发设计工作并反复论证。软件测试工作占很大的比重并贯穿于整个开发过程直至交付使用。这类软件交付使用后往往还要进行多次维护与升级。

以上只是简单地对软件按规模进行分类，并不能完全准确地写明软件按规模的分类标准。随着软件规模的增加，软件开发难度也随之成几何级数增加。规模大、时间长、很多人参加的软件项目，其开发工作必须要有软件工程的知识做指导。而规模小、时间短、参加人员少的软件项目也得有软件工程概念，遵循一定的开发规范。其基本原则是一样的，只是对软件工程技术依赖的程度不同而已。

3. 按软件工作方式划分

(1) 实时处理软件：指在事件或数据产生时，立即予以处理，并及时反馈信号，控制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 3 部分。

(2) 分时软件：允许多个联机用户同时使用计算机。

(3) 交互式软件：能实现人机通信的软件。

(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，按顺序逐个处理完的软件。

4. 按软件服务对象的范围划分

(1) 项目软件：也称定制软件，是受某个特定客户（或少数客户）的委托，由一个或多个软件开发机构在合同的约束下开发出来的软件。例如军用防空指挥系统、卫星控制系统等。

(2) 产品软件：是由软件开发机构开发出来直接提供给市场，或是为千百个用户服务的软件。例如文字处理软件、文本处理软件、财务处理软件、人事管理软件等。

1.1.3 软件工程的历史及发展

软件工程（Software Engineering, SE）是针对软件这一具有特殊性质的产品的工程化方法。软件工程涵盖了软件生存周期的所有阶段，并提供了一整套工程化的方法来指导软件人员的工作。

1. 软件工程的含义

“软件工程”是科学和数学的某种应用，通过这一应用，使计算机设备的能力借助于计算机程序、过程和有关文档成为对人类有用的东西。软件工程的成果是为软件设计和开发人员提供思想方法和工具，而软件开发是一项需要良好组织、严密管理且各方面人员配合协作的复杂工作，软件工程正是指导这项工作的一门科学。在过去一段时间内，软件工程已经取得了长足的进展，可以说在软件的开发和应用中起到了其应有的作用。高质量的软件工程可以保证软件工业中生产的软件是高

质量的产品、用户满意的产品。但是，对软件工程的界定总是存在一定的差异。

软件工程应该包括哪些知识？IEEE 在软件工程知识体系指南(SEWBOK, Guide to the Software Engineering Body of Knowledge 2004Version) 中这样定义：

(1) 软件开发、实施、维护的系统化、规范化、质量化的方法的应用，也就是软件的应用工程。

(2) 对上述方法的研究。

2. 软件开发的3个历史阶段

第一个写软件的人是Ada(Augusta Ada Lovelace), 在19世纪60年代他尝试为Babbage(Charles Babbage) 的机械式计算机写软件。尽管失败了，但他所做出的创新尝试却永远载入了计算机发展的史册。20世纪50年代，软件伴随着第一台电子计算机的问世诞生了。以写软件为职业的人也开始出现，他们多是经过训练的数学家和电子工程师。20世纪60年代美国大学里开始授予计算机专业的学位，教人们写软件。20世纪中期软件产业从零开始起步，在短短的50多年的时间里迅速发展成为推动人类社会发展的龙头产业。随着信息产业的发展，软件对人类社会越来越重要。软件发展的近60年历史中，人们对软件的认识经历了一个由浅到深的过程。

按时间可以将软件发展的历史大致分为如下3个阶段：

(1) 第一个阶段是20世纪50年代到60年代，是程序设计阶段，基本是个体手工劳动的生产方式。这个时期，一个程序是为一个特定的目的而编制的，软件的通用性很有限的。软件往往带有强烈的个人色彩。早期的软件开发没有什么系统的方法可以遵循，软件设计是在某个人的头脑中完成的一个隐蔽的过程。而且，除了源代码往往没有软件说明书等文档，因此这个时期尚无软件的概念，基本上只有程序、程序设计概念，不重视程序设计方法，主要是用于科学计算，规模很小，采用简单的工具（基本上采用低级语言），硬件的存储容量小，运行可靠性差。

(2) 第二阶段是20世纪60年代到70年代，是软件设计阶段，是小组合作的生产方式。在这一时期软件开始作为一种产品被广泛使用，出现了“软件作坊”。这个阶段，基本采用高级语言作为开发工具，开始提出结构化方法。硬件的速度、容量、工作可靠性有明显提高，而且硬件的价格降低。人们开始使用软件产品（可购买），从而建立了软件的概念。程序员数量猛增，但是开发技术没有新的突破，软件开发的方法基本上仍然沿用早期的个体化软件开发方式，软件需求日趋复杂，维护的难度越来越大，开发成本令人吃惊地高，开发人员的开发技术不适应规模大、结构复杂的软件开发，失败的项目越来越多。

(3) 第三个阶段是从20世纪80年代至今，为软件工程时代，是工程化的生产方式。这个阶段的硬件向超高速、大容量、微型化以及网络化方向发展，第三、四代语言出现。数据库、开发工具、开发环境、网络、分布式、面向对象技术等工具和方法都得到应用。

3. 软件危机

软件开发技术在不断地进步，但单纯软件开发技术的进步不能满足软件发展的要求。软件的数量、体积急剧膨胀，一些复杂的、大型的软件开发项目提出来了，但很多软件的开发最后都得到了一个悲惨的结局。很多的软件项目开发时间大大超出了规划的时间表，一些项目导致了财产的流失，甚至大量重复、复杂的高强度开发某些软件导致了人员伤亡。同时软件开发人员也发现软件开发的难度越来越大，在软件开发中遇到的问题找不到解决的办法，使问题积累起来，形成了尖锐的矛盾，失败的软件开发项目也屡见不鲜，因而导致了软件危机。软件危机就是指落后的软件生产方式无法满足迅速增长的计算机软件需求，从而导致软件开发与维护过程中出现一系列严重问题的现象。早

期出现的软件危机主要表现在:

(1) 软件开发费用和进度失控。费用超支、进度拖延的情况屡屡发生。有时为了赶进度或压成本不得不采取一些权宜之计, 这样又往往严重损害了软件产品的质量。

(2) 软件的可靠性差。尽管耗费了大量的人力物力, 而系统的正确性却越来越难以保证, 出错率大大增加, 由于软件错误而造成的损失十分惊人。

(3) 生产出来的软件难以维护。很多程序缺乏相应的文档资料, 程序中的错误难以定位, 难以改正, 有时改正了已有的错误又引入新的错误。随着软件的社会拥有量越来越大, 维护占用了大量人力、物力和财力。

进入 80 年代以来, 尽管软件工程研究与实践取得了可喜的成就, 软件技术水平有了长足的进步, 但是软件生产水平依然远远落后于硬件生产水平的发展速度。危机不仅没有消失, 还有加剧之势。主要表现在:

(1) 软件成本在计算机系统总成本中所占的比例居高不下, 且逐年上升。由于微电子技术的进步和硬件生产自动化程度的不断提高, 硬件成本逐年下降, 性能和产量迅速提高。然而软件开发需要大量人力, 软件成本随着软件规模和数量的剧增而持续上升。美国、日本两国的统计数字表明, 1985 年软件成本大约占总成本的 90%。

(2) 软件开发生产率提高的速度远远跟不上计算机应用迅速普及深入的需要, 软件产品供不应求的状况使得人类不能充分利用现代计算机硬件所能提供的巨大潜力。

最为突出的例子是美国 IBM 公司于 1963~1966 年开发的 IBM360 系列机的操作系统。这一项目花了 5000 人一年的工作量, 最多时 1000 人投入工作, 写了近 100 万行源程序, 包含 4000 多个模块, 耗资达数亿美元。系统运行后发现 2000 多个错误, 据统计, 这个系统每次发行的新版本都是从前一个版本中找出 1000 个程序错误修正的结果。最后该项目的负责人 Fred Brooks 在总结该项目时无比沉痛地说: “……正像一只逃亡的野兽落到泥潭中作垂死挣扎, 越是挣扎, 陷得越深, 最后无法逃脱灭顶的灾难, ……程序设计工作正像这样一个泥潭……一批批程序员被迫在泥潭中拼命挣扎, ……谁也没有料到问题竟会陷入这样的困境……。”

IBM360 操作系统的历史教训已成为软件开发项目中的典型事例被记入历史史册。出现这种现象的主要原因是:

(1) 用户需求不明确。软件开发过程中, 用户需求不明确问题主要体现在以下 4 个方面:

- 在软件开发出来之前, 用户自己也不清楚软件开发的具体需求。
- 用户对软件开发需求的描述不精确, 可能有遗漏、有二义性, 甚至有错误。
- 在软件开发过程中, 用户还提出修改软件开发功能、界面、支撑环境等方面的要求。
- 软件开发人员对用户需求的理解与用户本来的愿望有差异。

(2) 缺乏正确的理论指导。缺乏有力的方法学和工具方面的支持。由于软件开发不同于大多数其他工业产品, 其开发过程是复杂的逻辑思维过程, 其产品极大程度地依赖于开发人员高度的智力投入。由于过分地依靠程序设计人员在软件开发过程中的技巧和创造性, 加剧了软件开发产品的个性化, 也是出现软件开发危机的一个重要原因。

(3) 软件开发规模越来越大。随着软件开发应用范围的增广, 软件开发规模越来越大。大型软件项目需要组织一定的人力共同完成, 而多数管理人员缺乏开发大型软件开发系统的经验, 而多数软件开发人员又缺乏管理方面的经验。各类人员的信息交流不及时、不准确, 有时还会产生误解。软件开发项目的开发人员不能有效地、独立自主地处理大型软件开发的全部关系和各个分支, 因此

容易产生疏漏和错误。以美国宇航局的软件系统为例：

- 1963年 水星计划系统 200 万条指令
- 1967年 双子座计划系统 400 万条指令
- 1973年 阿波罗计划系统 1000 万条指令
- 1979年 哥伦比亚航天飞机系统 4000 万条指令

(4) 软件开发复杂度越来越高。软件开发不仅仅是在规模上快速地发展扩大，而且其复杂性也急剧地增加。软件开发产品的特殊性和人类智力的局限性，导致人们无力处理“复杂问题”。所谓“复杂问题”的概念是相对的，一旦人们采用先进的组织形式、开发方法和工具提高了软件开发的效率和能力，新的、更大的、更复杂的问题又摆在人们的面前。

由于软件危机的产生，迫使人们不得不研究改变软件开发的技术手段和管理方法。从此软件生产进入软件工程时代。

1968年北大西洋公约组织的计算机科学家在联邦德国召开的国际学术会议上第一次提出了“软件危机”(Software Crisis)这个名词。同时，讨论和制定摆脱“软件危机”的对策。在那次会议上第一次提出了软件工程(Software Engineering)这个概念，从此一门新兴的工程学科——软件工程学为研究和克服软件危机应运而生。

“软件工程”的概念是为了有效地控制软件危机的发生而被提出来的，它的中心目标就是把软件作为一种物理的工业产品来开发，要求“采用工程化的原理与方法对软件进行计划、开发和维护”。软件工程是一门旨在开发满足用户需求、及时交付、不超过预算和无故障的软件的学科。软件工程的主要对象是大型软件。它的最终目的是摆脱手工生产软件的状况，逐步实现软件开发和维护的自动化。

1.2 软件的生存周期

在很长一段时间里，人们认为开发软件就是编制程序，实际上，这种理解是狭隘的。首先来了解一下软件生存周期的定义，即以需求为触发点，提出软件开发计划的那一刻开始直到软件在实际应用中完全报废为止的一个完整的周期。这个周期可以认为是一个完整的软件生存周期。软件生存周期的提出是为了更好地管理、维护和升级软件。其中更大的意义在于管理软件开发的步骤和方法。它把整个的软件生存时间看做是一个整体，以时间的推移和软件开发的工作重心作为划分点，把软件开发和维护的工作细分为若干个相对独立的部分，从而更好地控制软件的开发进度和难度，同时也十分有利于降低软件的出错频率，协调各个部门间的工作配合和责任分配。

1.2.1 软件生存周期及各阶段的工作

软件生存周期各个阶段的划分并没有一成不变的法则，不同的开发方式、软件种类、软件规模和开发环境都会在不同程度上影响软件生存周期各阶段的划分，但无论最终把生存周期如何根据自己的实际情况进行划分，都是旨在更好地利用手中的资源（主要指人力资源、软件资源、技术资源和源码资源）降低软件的开发风险、复杂度和开发成本（主要以开发的时间和投入的资源为衡量标准），要做到对软件生存周期各阶段进行最好的划分，就必须遵循一条基本的原则，那就是各阶段

的任务应尽可能地相对独立,同一阶段各项任务的性质应尽可能地相同,从而达到降低每个阶段任务的复杂度,减少不同阶段任务之间的联系。这样做对软件项目开发的组织管理是十分必要的,同时对最终软件项目开发的成功也是不可或缺的。

尽管软件的生存周期各阶段的划分没有一个明确的法则,但就一般性而言,软件生存周期包括可行性分析和项目开发计划、需求分析、概要设计、详细设计、编写代码、软件测试、软件实施和软件维护等活动(有的文档资料和开发项目把概要设计和详细设计合在一起,统称为软件设计或设计),其中的每一个活动都是软件开发过程中必须要经历的,所以我们应该将它们按照项目的划分合理地安排到各个阶段中去。

既然软件开发周期这么重要,我们想要软件项目最终开发取得成功或是对软件进行管理和资源投入,就应当充分地了解周期里各个活动的定义和任务,这样才能合理、准确、客观地安排每一阶段的工作。下面就对各种活动的定义和任务做一下简单介绍,以对它们有一个初步的了解。

1. 可行性分析和项目开发计划

这两个活动通常被整合在一起进行,在实际工作中通常把它们归类到同一个阶段中。在某种程度上甚至可以把它们看成是一个活动整体,要做的事情就是回答“需要做什么?要如何去做?可不可能完成?”

在这个阶段中经验起到了决定性的作用,软件工程难就难在没有固定公式可供使用,很多时候都是靠系统分析员的经验来判断是否可行。在这个阶段中,可行性分析要依靠项目开发计划提供依据,而项目开发计划只有在初步得到可行性研究后才能再深入制定,两个活动可以说是互相制约、互相促进的关系。

同时在这个阶段中对要解决的问题定义十分重要,要注意和各方多沟通,得到尽可能准确的问题定义,再和各方再次沟通看看各方的理解是否相同,一般对问题的精确定义和理解在项目开发计划里解决比在需求分析阶段解决更合理,也可以更符合各方利益的要求,同时不会对软件开发方向造成隐患,亦不会给双方就软件开发报酬的商议造成不必要的麻烦。

在用户提出一个软件开发要求后,系统分析员要对此用户的机构进行了解,明确它是一个什么样的机构,它的作用是什么,这有利于分析所开发的项目的原由。同时对使用此软件的最终部门要进行一系列的观察研究,组织开会讨论,通过这一系列工作就可以确定软件项目的性质、目标和规模,其实这个工作有点像需求分析的简化版,但对项目的后期工作是一个奠基的作用。到此便应该能够得出可行性研究报告了。

如果可行性研究的结果是可行的,接下来的任务就是制定详细的项目开发计划。项目开发计划主要根据所开发的项目的目标、性能、功能、规模来确定所需的资源,主要包括3个方面,即硬件资源、软件资源和人力资源,除此之外还要对项目的开发费用、开发进度作出估计,可供决策者和用户参考。

至此,本阶段的工作任务已基本完成,这时候系统分析员应将《可行性报告》和《项目开发计划》一并提交管理部门审查。

2. 需求分析

软件开发最难的部分是什么?不用怀疑,就算是最初级的程序员也知道是需求分析,而另一个问题就是“需求分析为什么那么难呢?”要回答这个问题,必须在实际工作中把“两帮人”搞清楚,一帮是软件开发的相关人员,而另一帮则是使用软件的需求者。通常软件开发人员开发软件都不是为了自己使用,而是为某个组织开发的,这“两帮人”,一帮知道怎么用计算机解决实际问题而他