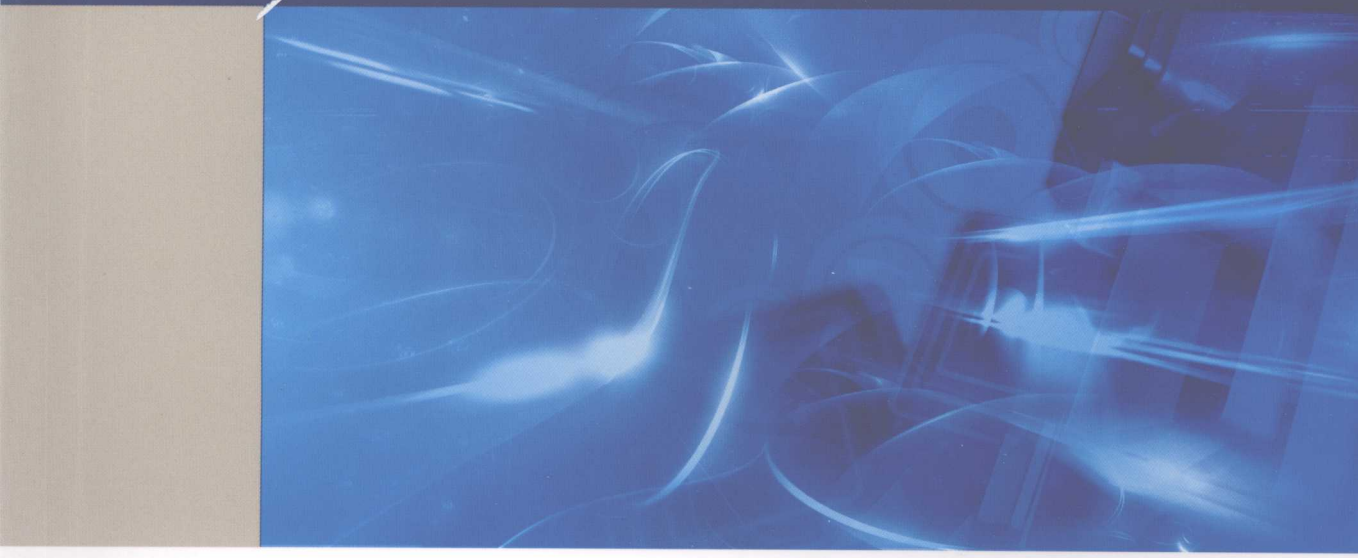
高职高专计算机技能型紧缺人才培养规划教材

计算机软件技术专业



面向对象软件工程与UML

张　京　主　编
李飞跃　郑显举　副主编
李成大　主　审

免费提供
教学相关资料

人民邮电出版社
POSTS & TELECOM PRESS

高职高专计算机技能型紧缺人才培养规划教材
计算机软件技术专业

面向对象软件工程与 UML

张京　主编
李飞跃　郑显举　副主编
李成大　主审

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

面向对象软件工程与 UML / 张京主编. —北京：人民邮电出版社，2008.9
高职高专计算机技能型紧缺人才培养规划教材. 计算机软件技术专业
ISBN 978-7-115-18202-9

Ⅰ. 面… Ⅱ. 张… Ⅲ. 面向对象语言，UML—高等学校：技术学校—教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2008）第 076303 号

内 容 提 要

本书系统介绍软件工程所涉及的各种概念、方法和新技术，重点讲解 UML（统一建模语言）的基本理论和应用以及使用 PowerDesigner 工具建立 UML 模型、进行面向对象的需求获取、软件系统的分析设计与实现等内容。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、UML、PowerDesigner 环境介绍、用例模型分析、逻辑模型分析等方面的内容。本书实例丰富，各章均有小结与习题，便于教学和自学。

本书可作为高职高专院校计算机专业的教材，也可供各类软件产品开发人员学习参考。

高职高专计算机技能型紧缺人才培养规划教材
计算机软件技术专业

面向对象软件工程与 UML

◆ 主　　编　张　京
副 主 编　李飞跃　郑显举
主　　审　李成大
责任编辑　潘春燕
执行编辑　刘雁斌

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京隆昌伟业印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：15
字数：357 千字　　2008 年 9 月第 1 版
印数：1－3 000 册　　2008 年 9 月北京第 1 次印刷

ISBN 978-7-115-18202-9/TP

定价：25.00 元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

高职高专计算机技能型紧缺人才培养

规划教材编委会

丛书出版前言

目前，人才问题是制约我国软件产业发展的关键。为加大软件人才培养力度和提高软件人才培养质量，教育部继在2003年确定北京信息职业技术学院等35所高职院校试办示范性软件职业技术学院后，又同时根据《教育部等六部门关于实施职业院校制造业和现代服务业技能型紧缺人才培养培训工程的通知》（教职成［2003］5号）的要求，组织制定了《两年制高等职业教育计算机应用与软件技术专业领域技能型紧缺人才培养指导方案》。示范性软件职业技术学院与计算机应用与软件技术专业领域技能型紧缺人才培养工作，均要求在较短的时间内培养出符合企业需要、具有核心技能的软件技术人才，因此，对目前高等职业教育的办学模式和人才培养方案等做较大的改进和全新的探索已经成为学校的当务之急。

据此，我们认为做一套符合上述一系列要求的切合学校实际的教学方案尤为重要。遵照教育部提出的以就业为导向，高等职业教育从专业本位向职业岗位和就业为本转变的指导思想，根据目前高等职业教育院校日益重视学生将来的就业岗位，注重培养毕业生的职业能力的现状，我们联合北京信息职业技术学院等几十所高职院校和普拉内特计算机技术（北京）有限公司、福建星网锐捷网络有限公司、北京索浪计算机有限公司等软件企业共同组建了计算机应用与软件技术专业领域技能型紧缺人才培养教学方案研究小组（以下简称研究小组）。研究小组对承担计算机应用与软件技术专业领域技能型紧缺人才培养培训工作的79所院校的专业设置情况做了细致的调研，并调查了几十所高职院校计算机相关专业的学生就业情况以及目前软件企业的人才市场需求状况，确定首批开发目前在高职院校开设比较普遍的计算机软件技术、计算机网络技术、计算机多媒体技术和计算机应用技术4个专业方向的教学方案。

同时，为贯彻教育部提出的要与软件企业合作开展计算机应用与软件技术专业领域技能型紧缺人才培养培训工作的精神，使高等职业教育培养出的软件技术人才符合企业的需求，研究小组与许多软件企业的专家们进行了反复研讨，了解到目前高职院校的毕业生的实际动手能力和综合应用知识方面较弱，他们和企业需求的软件人才有着较大的差距，到企业后不能很快独当一面，企业需要投入一定的成本和时间进行项目培训。针对这种情况，研究小组在教学方案中增加了“综合项目实训”模块，以求强化学生的实际动手能力和综合应用前期所学知识的能力，探索将企业的岗前培训内容前移到学校的教学中的实验之路，以此增强毕业生的就业竞争力。

在上述工作的基础上，研究小组于2004年多次组织召开了包括企业专家、教育专家、学校任课教师在内的各种研讨会和方案论证会，对各个专业按照“岗位群→核心技能→知识点→课程设置→各课程应掌握的技能→各教材的内容”一步步进行了认真的分析和研讨：

- 列出各专业的岗位群及核心技能。针对教育部提出的以就业为导向，根据目前高职高专院校日益关心学生将来的就业岗位的现状，在前期大量调研的基础上，首先提炼各个专业的岗位群。如对某专业的岗位群进行研究时，首先罗列此专业的各个岗位，以便能正确了解

每个岗位的职业能力，再根据职业能力进行有意义的合并，形成各个专业的岗位群，再对每个岗位群总结和归纳出其核心技能。

- 根据岗位群及核心技能做出教学方案。在岗位群及核心技能明确的前提下，列出此岗位应该掌握的知识点，再依据这些知识点推出应该学习的课程、学时数、课程之间的联系、开课顺序并进行必要的整合，最终形成一套科学完整的教学方案。

为配合学校对技能型紧缺人才的培养工作，在研究小组开发上述 4 个专业的教学方案的基础上，我们组织编写了这套包含计算机软件技术、计算机网络技术、计算机多媒体技术及计算机应用技术 4 个专业的教材。本套教材具有以下特点：

- 注重专业整体策划的内涵。对各专业系列教材按照“岗位群→核心技能→知识点→课程设置→各课程应掌握的技能→各教材的内容”的思路组织开发教材。
- 按照“理论够用为度”的原则，对各个专业的基础课进行了按需重新整合。
- 各专业教材突出了实训的比例，注重案例教学。每本教材都配备了实验、实训的内容，部分专业的教材配备了综合项目实训，使学生通过模拟具体的软件开发项目了解软件企业的运行环境，体验软件的规范化、标准化、专业化和规模化的开发流程。

为了方便教学，我们免费为选用本套教材的老师提供部分专业的整体教学方案及教学相关资料。

- 所有教材的电子教案。
- 部分教材的习题答案。
- 部分教材中实例制作过程中用到的素材。
- 部分教材中实例的制作效果以及一些源程序代码。

本套教材以各个专业的岗位群为出发点，注重专业整体策划，试图通过对系列教材的整体构架，探索一条培养技能型紧缺人才的有效途径。

经过近两年的艰苦探索和工作，本套教材终于正式出版了，我们衷心希望，各位关心高等职业教育的读者能够对本套教材的不当之处给予批评指正，提出修改意见，也热切盼望从事高等职业教育的教师以及软件企业的技术专家和我们联系，共同探讨计算机应用与软件技术专业的教学方案和教材编写等相关问题。来信请发至 panchunyan@ptpress.com.cn。

前　言

软件工程是一门研究范围非常广泛的发展迅速的新兴学科，学科内的新技术、新方法不断涌现。经过30多年的研究与发展，软件工程的方法从传统的结构化软件工程方法，发展到面向对象的软件工程方法，再到基于构件的软件工程方法，进而到基于Web的软件工程方法；在技术方面，软件复用、构件接口、Web Services等新技术也逐渐地发展和完善，成为传统的结构化技术之后的主流技术。此外，为了保证软件的质量，软件工程领域中项目管理这一部分也越来越受到重视，软件开发过程的管理方法也越来越科学，软件质量保证和开发过程的评估体系也逐渐完善。

软件工程领域在1995年至1997年取得了前所未有的进展，其中最重要的、具有划时代重大意义的成果之一就是统一建模语言（Unified Modeling Language，UML）的出现。在世界范围内，可以说至少在近10年内，UML将是面向对象技术领域内占主导地位的标准建模语言。UML代表了面向对象方法的软件开发技术的发展方向，具有巨大的市场前景。目前国内各类高校开设的软件工程课程中，利用具体的UML工具进行软件开发模型建立的有关内容介绍得较少。基于这种现状，我们组织了一批教学、实践经验丰富的教师编写了本教材，重点讲解面向对象的软件开发方法与UML，开发工具选择了目前性价比较高的PowerDesigner进行介绍。

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性和先进性。在内容设计方面，我们从软件工程的结构化分析方法开始，逐步过渡到面向对象的软件工程，重点突出了UML建模语言的基本理论和应用，以及教会学生使用PowerDesigner工具建立UML模型、进行面向对象的需求获取、软件系统的分析设计与实现，体现了“理论够用，方法先进，实现具体”的思路。

本书共12章，分为4大部分。第1部分为软件工程开发方法，第2部分为面向对象软件开发方法，第3部分为统一建模语言（UML），第4部分为使用PowerDesigner建立UML模型。在课时分配方面，建议开设64课时左右为宜，对于有软件工程基础的读者，可略讲或不讲第1部分内容。

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。本书突出实践动手能力和实用性，突出案例（第12章为综合应用实例），力图使学生学习本书后掌握软件工程开发的全过程，并学会用PowerDesigner这一工具建立UML模型。

本书的第1～5、7、11章及附录由张京编写，第8～10章由李飞跃编写，第6、12章由郑显举编写，张京负责全书统稿。李成大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，加之时间紧促，书中疏漏之处在所难免，恳请专家和读者批评指正。

编　者

2008年5月

目　录

第 1 章

软件工程

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。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，它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，读者能更好地理解计算机软件并且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，但无法看到软件的形态，必须通过测试、分析、思考、判断来了解它的功能、性能及其他特性。软件正确与否，是好是坏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）软件的开发是人的智力的高度发挥，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）软件维护与硬件的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，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，其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，这使软件的开发效率受到了很大限制。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，以提高软件开发的效率。

（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产物就是计算机软件。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，开发费用非常高，所以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？虽然现在没有一个统一的严格分类标准，但也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。按其性质的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，对需要监测和控制的过程进行监控的软件。它主要包括数据采集、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，都将会造成事故。

（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。按软件规模可划分为小型软件、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 4 个不同的发展阶段。下面简要地介绍各个发展阶段的特点，使读者能够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（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 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原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 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取得了广泛应用。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，计算机系统发展进入第四代。这时人们已经不再看重单台计算机和程序，而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、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：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。一般称软件开发和维护过程中所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：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的产品不满意。

（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，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以改正，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，必须在软件开发中寻找新的方法、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（组）人编写和运行，如果出了故障也由同一个（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，他们应该是工程师。

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、有纪律地开展软件的开发工作？回答是肯定的。于是人们萌生了一种想法：像处理“工程”问题一样来处理软件开发全过程，于是产生了“软件工程”的概念。“软件工程”一词自 1968 年问世以来，经过许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开发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 3 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被其他的工具所使用，这样建立起一种称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，并会简要介绍面向对象方法学。以下先对这两种方法学做一下介绍和对比。

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，是从对任务的抽象逻辑分析开始，分阶段进行开发的。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使前一阶段提出的解法更进一步具体化，加入了更多的实现细节。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，通过检查之后这个阶段才算结束；如果检查没通过，则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来交付使用。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，以便于人员分工协作，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的可维护性并不是最佳的。归结原因是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。但是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。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得同等重要，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。

面向对象方法具有下述 4 个要点。

（1）把对象作为融合了数据和在数据上的操作行为统一的软件构件。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的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。

正确运用面向对象方法学开发软件，则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。一般来说，软件生存期由软件定义、软件开发和软件维护 3 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或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，