21st CENTURY
规划教材

面向21世纪高等院校计算机系列规划教材

COMPUTER COURSES FOR UNDERGRADUATE EDUCATION

软件工程

薛德黔 等 编著

科学出版社
www.sciencep.com

面向21世纪高等院校计算机系列规划教材
COMPUTER COURSES FOR UNDERGRADUATE EDUCATION

软件工程

薛德黔 等 编著

科学出版社
北京

内 容 简 介

本书以传统的软件工程和面向对象的软件工程为主线，根据软件开发“工程化”思想，通过大量的应用实例，系统地介绍软件工程的基本概念、基本原理、软件开发的过程、开发方法、应用技术和实用工具。主要包括可行性研究、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。本书力图反映软件工程领域的最新发展，并从实用性出发，各章节均结合实例讲解，深入浅出，使读者易于理解和掌握。

本书适合作为高等院校计算机专业及相关专业教材，也可供计算机软件开发和管理人员参考。

图书在版编目（CIP）数据

软件工程/薛德黔等编著. —北京：科学出版社，2005

（面向21世纪高等院校计算机系列规划教材）

ISBN 7-03-016286-2

Ⅰ.软… Ⅱ.薛… Ⅲ.软件工程-高等学校-教材 Ⅳ.TP311.5

中国版本图书馆CIP数据核字（2004）第109634号

责任编辑：韩 洁 刘亚军/责任校对：郝 岚

责任印制：吕春珉/封面设计：飞天创意

科学出版社 出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

双青印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2005年10月第 一 版 开本：787×1092 1/16

2005年10月第一次印刷 印张：17

印数：1－3 000 字数：387 000

定价：23.00元

（如有印装质量问题，我社负责调换〈双青〉）

销售部电话 010-62136131 编辑部电话 010-62138978-8001（HI01）

前　言

软件工程是计算机科学的一个重要分支，所涉及的范围非常广泛，包括软件开发技术、软件工程环境、软件经济学、软件心理学和软件工程管理等方面的知识。本书以软件的生命周期为主线，重点讨论了结构化的软件开发方法和技术，包括结构化分析与设计、编码、测试。在软件工程的入门阶段，结构化软件开发方法是基本、实用的技术。通过对基本概念、原理、技术和方法的学习，读者能很快掌握软件工程的方法和软件开发技术。近年来，面向对象软件开发方法和技术的研究及其应用不断普及，本书利用一定篇幅介绍了面向对象的基本概念、分析和设计方法。面向对象方法符合人们认识客观世界、解决复杂问题的渐进过程；软件的稳定性、可重用性和易于维护性是当今流行软件开发方法的主要特点。本书最后一章介绍了我国软件工程的发展状况，并对软件工程的未来进行了展望，提出了当代大学生应肩负的历史使命。在附录部分提供了一些常用资料供读者查询。

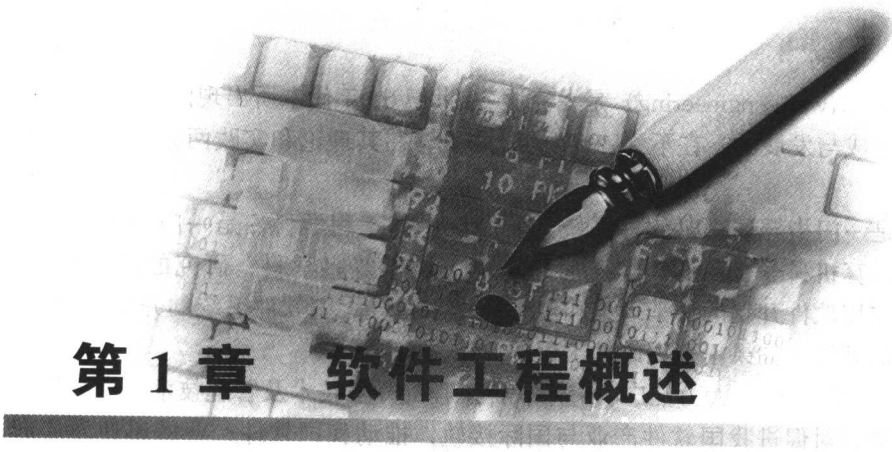
作者根据长期的教学实践和经验，参考国内外众多最新（版本）的教材和论文精选内容，注重基础性、系统性、实用性和新颖性，并结合大量软件项目的实例分析，深入浅出地阐述软件工程方法、应用技术和实用工具。本书重视基本知识和基本概念的传授，按照我国目前执行的软件开发国家标准组织教材内容，注意理论与实践的结合。通过本书的学习，学生能系统掌握软件开发全过程所必须具有的知识，初步掌握软件开发的基本思想、方法和相关技能，掌握软件项目开发和管理的实践技能，为适应软件工程的发展，更深入地学习和从事软件工程实践需要打下良好的基础。

另外，为了方便教师教学，本书配有教学指南、电子教案及习题答案（电子版），可以到科学出版社网站（http://www.sciencep.com）的下载区下载。

本书由薛德黔、王智群、徐明亮、肖浩等编著，薛德黔教授主审，其中第1、2章由徐明亮编写，第3、4章由王智群编写，第5、6章由肖浩编写，第7章由薛德黔编写。

由于时间仓促，加之作者水平有限，书中难免存在不足和疏漏之处，敬请读者批评指正。

目　录



第1章 软件工程概述

本章要点

- ◇ 软件工程产生的原因
- ◇ 软件工程的基本原理
- ◇ 软件工程方法学和面向对象方法学基本原理
- ◇ 软件过程
- ◇ 软件工程有关工具和环境

本章学习目标

- ◇ 了解软件工程产生的原因
- ◇ 掌握软件工程的基本原理
- ◇ 了解软件工程方法学和面向对象方法学基本原理
- ◇ 了解软件过程
- ◇ 了解软件工程有关工具和环境

1.1 软件工程的产生

软件工程（software engineering）是在克服20世纪60年代末所出现的“软件危机”的过程中逐渐形成与发展的。在不到40年的时间里，在其理论和实践两方面都取得了长足的进步。

软件工程是一门指导计算机软件系统开发和维护的工程学科，是一门新兴的边缘学科，它涉及计算机科学、工程科学、管理科学、数学等多学科，研究的范围较广，主要研究如何应用软件开发的科学理论和工程技术来指导大型软件系统的开发。例如，现代操作系统的开发，如果不采用软件工程的方法是不可能的。

在我国加入WTO后，大力推广、应用软件工程的开发技术及管理技术，提高软件工程的应用水平，对促进我国软件产业与国际接轨，推动我国软件产业的迅速发展起着十分重要的作用。

软件工程的产生和发展是与软件的发展紧密相关的。自从世界上第一台计算机诞生以来，就开始了软件的生产，到目前为止，软件生产的发展经历了以下三个阶段。

（1）程序设计时代（1946～1956年）

采用“个体生产方式”，即软件开发完全依赖于程序员个人的能力水平。

（2）程序系统时代（1956～1968年）

由于软件应用范围及规模的不断扩大，个体生产已经不能满足软件生产的需要，一个软件需要由几个人协同完成，采用“生产作坊方式”。

该阶段的后期，随着软件需求量、规模及复杂度的增大，生产作坊的方式已经不能适应软件生产的需要，出现了所谓的“软件危机（software crisis）”。

（3）软件工程时代（1968年至今）

该阶段的主要任务是为了克服软件危机，适应软件发展的需要，而采用“工程化的生产”方式。

“软件危机”的出现是由于软件的规模越来越大，复杂度不断增加，软件需求量增大。而软件开发过程是一种高密集度的脑力劳动，软件开发的模式及技术不能适应软件发展的需要，致使大量的质量低劣的软件涌向市场，有的软件虽花费了大量的人力财力，却在开发过程中就夭折了。例如，IBM公司的OS/360，共约100万条指令，花费了5 000多个人·年，经费达数亿美元，而结果却令人沮丧，错误多达2 000个以上，系统根本无法正常运行。OS/360系统的负责人Brooks这样描述开发过程的困难和混乱：“……像巨兽在泥潭中做垂死挣扎，挣扎得越猛，泥浆就沾得越多，最后没有一只野兽能够逃脱淹没在泥潭中的命运……”1963年，美国飞往火星的火箭因为一个软件错误而爆炸。1967年8月23日，前苏联“结盟一号”载人宇宙飞船，也由于忽略了一个小数点，在进入大气层时因打不开降落伞而烧毁。

“软件危机”主要表现在两个方面。

1）软件产品质量低劣，甚至在开发过程中就夭折。

2）软件生产率低，不能满足需要。

软件工程研究的目标是“以较少的投资获取较高质量的软件”。归结起来，软件工程研究的主要内容有以下两个方面。

1）软件开发技术：包括软件开发方法、技术和软件开发工具及环境、软件管理技术。

2）软件规范（国际规范）：包括以下内容。

①软件开发技术（软件结构、开发方法、工具与软件工程环境、软件工程标准化）。

②软件工程管理（质量管理、软件工程经济学——成本估算及计划安排等）。

1.2 软件工程及其基本原理

软件工程就是将工程学原理应用于软件开发与维护的实践，是人们从解决20世纪60年代开始出现的软件危机中逐步形成和发展起来的。随着软件产品的系列化，软件开发的工程化、标准化和产业化，软件工程学已成为软件产业发展的重要理论与技术基础。

1.2.1 软件工程定义

采用工程学的概念、原理、技术和方法，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、方法结合起来开发与维护软件，这就是软件工程。由表1.1，通过对名称、生产方式、质量、设计对象、开发工具、维护等方面进行比较，可了解软件开发方法各阶段的特点。从根本上说，软件工程学时代摆脱了软件“个体式”或“作坊式”的生产方法，把软件作为一种社会产品，并进行批量生产。软件工程学研究这种生产过程的有关基础理论、方法论和工具系统。

表1.1 软件开发方法的三个发展阶段

阶　段	程序设计时代	程序系统时代	软件工程学时代
名称	程序	软件	软件产品
生产方式	个人	作坊式项目小组	软件组织
质量	取决于个人水平	取决于小集团水平	生产管理可靠性评价和质量控制
设计对象	以硬件为中心	硬件/软件为中心	以软件为中心
开发工具	无	无系统工具且个人所有	软件生成器等，为组织所公有
维护	无	由开发者进行维护，且在设计中不重视设计维护问题	设计制作时均考虑维护问题，维护占成本主要部分，近年已达到80%以上
设计方法	没有系统的方法	自顶向下的方法	结构化程序设计及自顶向下和自底向上结合的方法

软件工程学中有如下主要概念。

1）程序：为了使计算机实现预期的目的（如解某一算题或控制某一过程）而编排的一系列步骤称为程序。程序可以用机器指令来编写，也可以用程序设计语言来编写。

2）软件：计算机的程序加上该程序的各种规格书或文档。软件方法是以大型程序为研究对象的，相应文档是软件的核心之一。

3）软件工程：生产软件的工程。研究软件工程的学问叫软件工程学（有时人们也把软件工程学简称为软件工程）。在某些书中，软件工程也称为软件产品工程学和软件

生产管理法。

4）软件可靠性：软件在所给条件下和规定时间内，能完成所要求的功能的性质。

5）软件可靠度：软件在所给条件下和规定时间内，能完成所要求功能的概率。很明显软件可靠性和硬件可靠性完全不同。对硬件而言，经过早期的故障排除之后，它稳定在一定的故障频率范围内，这个期间的故障是随机的。使用若干年后，故障率增大。但对软件而言，不存在由于工作中的损耗而发生故障。软件的故障差不多均是在设计、制作阶段中已存在但未被发现的。因此，只有通过维护才可能使软件可靠度随时间的增加而增加。

1.2.2 软件工程的基本原理

自从1968年在前联邦德国召开的国际会议上正式提出并使用了“软件工程”这个术语以来，研究软件工程的专家学者们陆续提出了100多条关于软件工程的准则或“信条”。著名的软件工程专家B.W.Boehm综合这些学者的意见并总结了TRW公司多年开发软件的经验，于1983年在一篇论文中提出了软件工程的七条基本原理。他认为这七条原理是确保软件产品质量和开发效率的原理的最小集合。这七条原理是互相独立的，其中任意六条原理的组合都不能代替另一条原理，因此，它们是缺一不可的最小集合，然而这七条原理又是相当完备的，人们虽然不能用数学方法严格证明它们是一个完备的集合，但是，可以证明在此之前已经提出的100多条软件工程原理都可以由这七条原理的任意组合蕴含或派生。

软件工程的七条基本原理如下所述。

（1）用分阶段的生命周期计划严格管理

有人经统计发现，在不成功的软件项目中有一半左右是由于计划不周造成的。把建立完善的计划作为第一条基本原理是在吸取了前人的教训之后提出来的。

在软件开发与维护的漫长的生命周期中，需要完成许多性质各异的工作。这条基本原理意味着，应该把软件生命周期划分成若干个阶段，并相应地制定出切实可行的计划，然后严格按照计划对软件的开发与维护工作进行管理。Boehm认为，在软件的整个生命周期中应该制定并严格执行六类计划，即项目概要计划、里程碑计划、项目控制计划、产品控制计划、验证计划和运行维护计划。

不同层次的管理人员都必须严格按照计划各尽其职地管理软件开发与维护工作，绝不能受客户或上级人员的影响而擅自背离预定计划。

（2）坚持进行阶段评审

当时已经认识到，软件的质量保证工作不能等到编码阶段结束之后再进行，其理由说至少有两个：第一，大部分错误是在编码之前造成的，（根据Boehm等人的统计，设计错误占软件错误的63%，编码错误仅占37%）；第二，错误发现与改正得越晚，所需付出的代价也越高。因此，在每个阶段都进行严格的评审，以便尽早发现在软件开发过程中所犯的错误，是一条必须遵循的重要原则。

（3）实行严格的产品控制

在软件开发过程中不应随意改变需求，因为改变一项需求往往需要付出较高的代

价。但是，在软件开发过程中改变需求又是难免的。由于外部环境的变化，相应地改变用户需求是一种客观需要，显然不能硬性禁止客户提出改变需求的要求，而只能依靠科学的产品控制技术来顺应这种要求。也就是说，当需求改变时，为了保持软件各个配置成分的一致性，必须实行严格的产品控制，其中主要是实行基准配置管理。所谓基准配置又称为基线配置，它们是经过阶段评审后的软件配置成分（各个阶段产生的文档或程序代码）。基准配置管理也称为变动控制：一切有关修改软件的建议，特别是涉及对基准配置的修改建议，都必须严格地按照规程进行评审，获得批准以后才能实施修改，绝对不能谁想修改（包括尚在开发过程中的软件）就随意进行修改。

（4）采用现代程序设计技术

从提出软件工程的概念开始，人们一直把主要精力用于研究各种新的程序设计技术。20 世纪 60 年代末提出的结构程序设计技术，已经成为绝大多数人公认的先进的程序设计技术。以后又进一步发展出各种结构分析（SA）与结构设计（SD）技术。实践表明，采用先进的技术既可提高软件开发的效率，又可提高软件维护的效率。

（5）结果应能清楚地审查

软件产品不同于一般的物理产品，它是看不见摸不着的逻辑产品。软件开发人员（或开发小组）的工作进展情况可见性差，难以准确度量，从而使得软件产品的开发过程比一般产品的开发过程更难于评价和管理。为了提高软件开发过程的可见性，并更好地对其进行管理，应该根据软件开发项目的总目标及完成期限，规定开发组织的责任和产品标准，从而使所得结果能够被清楚地审查。

（6）开发小组的人员应该少而精

这条基本原理的含义是，软件开发小组的组成人员的素质应该好，而人数则不宜过多。开发小组人员的素质和数量是影响软件产品质量和开发效率的重要因素。素质高的人员的开发效率比素质低的人员的开发效率可能高几倍至几十倍，而且素质高的人员所开发的软件中的错误明显少于素质低的人员所开发的软件中的错误。此外，随着开发小组人员数目的增加，因交流情况讨论、问题而造成的通信开销也会急剧增加。当开发小组人员数为 N 时，可能的通信路径有 $N(N-1)/2$ 条。可见，随着人数 N 的增大，通信开销将急剧增加。因此，组成少而精的开发小组是软件工程的一条基本原理。

（7）承认不断改进软件工程实践的必要性

遵循上述六条基本原理，就能够按照当代软件工程基本原理实现软件的工程化生产，但是，仅有上述六条原理并不能保证软件开发与维护的过程能赶上时代前进的步伐，能跟上技术的不断进步。因此，Boehm 提出应把承认不断改进软件工程实践的必要性作为软件工程的第七条基本原理。按照这条原理，不仅要积极主动地采纳新的软件技术，而且要注意不断地总结经验。例如，收集进度和资源耗费数据，收集出错类型和问题报告数据等。这些数据不仅可以用来评价新的软件技术的效果，而且可以用来指明必须着重开发的软件工具和应该优先研究的技术。

1.3 软件生存期

软件生存期又称软件生命周期，是指一个软件系统从目标提出到最后丢弃的整个过程。软件工程基本原理强调软件生命周期的阶段性，其基本思想是各阶段的任务相对独立，具有明确的完成标志，阶段的划分使得人员分工职责清楚，项目进度控制和软件质量得到确认。原则上，前一阶段任务的完成是后一阶段工作的前提和基础；而后一阶段的任务则是对前一阶段问题求解方法的具体化。

为了描述软件生存期的活动，提出了多种生存期模型，如瀑布模型、循环模型、演化模型、螺旋模型等。

整个软件生命周期可以划分为三个时期，即设计时期、开发时期、远行时期。一般用经典的瀑布模型来描述，如图 1.1 所示。

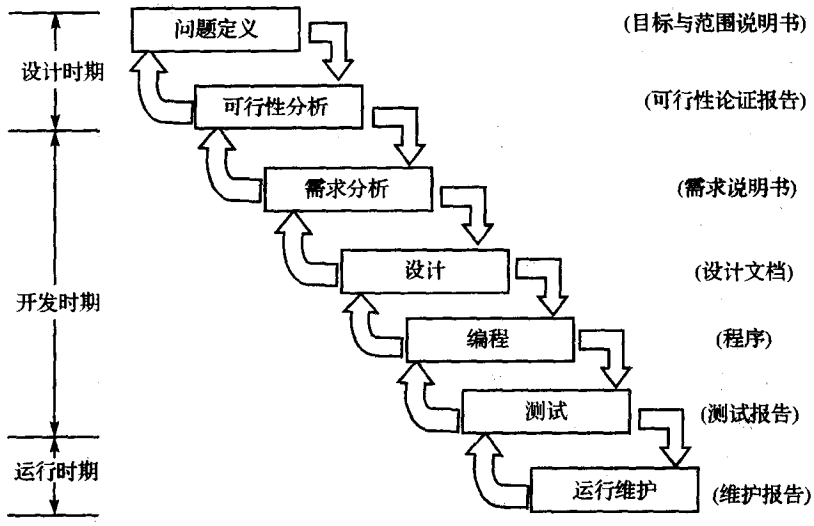


图 1.1 软件生命周期的瀑布型模型

GB 8567—88 中规定，软件生命周期分为七个阶段：可行性研究和项目开发计划、需求分析、概要设计、详细设计、编码、测试、维护。

在大部分文献中将生存周期划分为五个阶段，即要求定义、设计、编码、测试和维护，其中要求定义阶段包括可行性研究和项目开发计划、需求分析，设计阶段包括概要设计和详细设计。

下面简要介绍如图 1.1 所示的软件生存周期各个阶段的基本任务和结束标准。

(1) 问题定义阶段

问题定义阶段必须回答的关键问题是："要解决的问题是什么?" 因此，分析员通过对系统的实际用户和使用部门负责人的访问调查，扼要地写出他们对问题的理解，并在用户和使用部门负责人的会议上认真讨论这份书面报告，澄清含糊不清的地方，改正理解不正确的地方，最后得到一份双方都满意的文档，此文档中系统分析员应该写明问题的性质、工程目标和规模。

问题定义阶段是软件生存周期中最简短的阶段，一般只需一天甚至更少的时间。

（2）可行性研究阶段

此阶段的任务不是具体解决问题，而是研究问题的范围，探索这个问题是否值得去解决，是否有可行的解决办法。在这个阶段，系统分析员应该导出系统的高层逻辑模型，并且在此基础上更准确、更具体地确定工程规模和目标。然后分析员更准确地估计系统的成本和效益，对建议的系统进行仔细的成本/效益分析，这是这个阶段的主要任务之一。

可行性研究的结果是使用部门负责人做出是否继续进行这项工程的决定的重要依据。

（3）需求分析阶段

这个阶段的任务，主要是确定目标系统必须具备哪些功能。因此，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，根据字典和简要的算法用数据流图描述表示系统的逻辑模型。需求分析阶段确定的系统逻辑模型，是以后设计和实现目标系统的基础，因此必须准确、完整地体现用户的要求。

（4）设计阶段

1）总体设计阶段。设计阶段又分为总体设计和详细设计阶段。

这个阶段必须回答的关键问题是："应该如何解决这个问题?"

首先应该考虑几种可能的解决方案，一般包括以下几种。

①低成本的解决方案：系统只能完成最必要的工作，不能多做一点额外的工作。

②中等成本的解决方案：这样的系统不仅能够很好地完成预定的任务，使用起来很方便，而且可能还具有用户没有具体指定的某些功能和特点。

③高成本的"十全十美"的系统。这样的系统具有用户可能希望有的所有功能和特点。

系统分析员应该使用系统流程图或其他工具描述每种可能的系统，估计每种方案的成本和效益；还应该在充分权衡各种方案利弊的基础上，推荐一个较好的系统，并且制定实现所推荐的系统的详细计划。

要完成上述任务，通常采用结构设计的一条基本原理就是程序应该模块化。因此，总体设计还应设计软件的结构，通常用软件结构图表示。

2）详细设计阶段。详细设计阶段的任务就是把解法具体化，设计出程序的详细规格说明，包括必要的细节，程序员可以根据它们写出实际的程序代码。

通常用程序流程图、N-S图、PAD图、IPO图等描述详细设计的结果。

（5）编码和单元测试阶段

这个阶段的任务是程序员根据目标系统的性质和实际环境，选取一种适当的高级程序设计语言（必要时用汇编语言），把详细设计的结果翻译成用选定的语言书写的程序，并且仔细测试编写出的每一个模块。

程序员在书写程序模块时，应使它的可读性、可理解性和可维护性良好。

（6）综合测试阶段

这个阶段的任务是通过各种类型的测试，使软件达到预定的要求。

最基本的测试是集成测试和验收测试。集成测试是根据设计的软件结构，把经单元测试的模块按某种选定的策略装配起来，在装配过程中对程序进行必要的测试。验收测试是按照需求规格说明书的规定，由用户对目标系统进行验收。

通过对软件测试结果的分析可以预测软件的可靠性；反之，根据对软件可靠性的要求也可以决定测试和调试过程什么时候可以结束。

在进行测试的过程中，应该用正式的文档把测试计划、详细测试方案以及实际测试结果保存下来，作为软件配置的一部分。

(7) 软件维护阶段

维护阶段的任务，是通过各种必要的维护活动使系统持久地满足用户的需要。

通常，维护活动有四类：改正性维护，即诊断和改正在系统使用过程中发现的软件错误；适应性维护，即修改软件以适应环境的变化；完善性维护，即根据用户的要求改进或扩充软件使它更完善；预防性维护，即修改软件为将来的维护活动预先做准备。

每一项维护活动都应该准确地记录下来，作为正式的文档资料加以保存。

软件的生存周期划分为上述七个阶段，前两个阶段称为软件的定义阶段，第三至第六个阶段称为软件的开发阶段，最后一个阶段称为软件的维护阶段。在软件开发期间，测试的工作量最大，约占总开发量的 40%；而软件的维护阶段周期最长，工作量非常大。

软件系统的研制工作，不可能是直线进行，研制人员常常需从后面阶段回复到前面。为了减少返工现象，研制人员通常在各个阶段进行阶段复审，以确保研制工作顺序进行。

在软件生存周期的各个阶段完成研制任务后，应提交各阶段的格式文档资料。

1.4 软件工程方法学

软件开发的目标就是在规定的投资和时间限制内，开发出符合用户需求的高质量软件。软件开发是一种高智能的活动，必须用软件工程的方法和技术指导软件开发的全过程。

1.4.1 软件工程方法学

已经提出的各种软件开发方法和技术，对软件工程的发展和软件产业的进步，起到了不可估量的积极作用。各种开发方法和技术可归纳为三大类：瀑布型模型、原型化模型和变换型。

1. 软件开发的瀑布型模型

严格按照软件生命周期的阶段划分，顺序执行各阶段构成软件开发的瀑布型模型。瀑布型模型具有如下特点。

(1) 阶段间具有顺序性和依赖性

顺序性要求每个阶段工作开始的前提是其上一阶段工作结束。因此前一阶段输出