面向“十二五”高职高专规划教材·计算机系列

实用软件工程

王爱平　主编
王宏亮　张庾　副主编

清华大学出版社·北京交通大学出版社

面向“十二五”高职高专规划教材·计算机系列

实用软件工程

王爱平　主　编
王宏亮　张　庚　副主编

清华大学出版社
北京交通大学出版社
·北京·

内 容 简 介

本书从实用的角度出发，比较全面系统地介绍了软件工程的概念、原理和技术方法。主要内容包括软件工程的基本概念、软件过程模型、可行性分析、需求分析方法、软件系统设计方法、软件编码、软件测试及软件项目管理等，并对软件生命周期各环节的文档附有规格说明书。

本书内容充实、实用性强，可作为高职高专院校计算机软件专业软件工程课程的教材，也可作为有关软件工程师的培训教材，对从事软件开发工作的相关技术人员也具有一定的参考价值。

图书在版编目(CIP)数据

实用软件工程/王爱平主编. —北京：清华大学出版社；北京交通大学出版社，2009.8
（面向“十二五”高职高专规划教材·计算机系列）
ISBN 978－7－81123－589－0

Ⅰ. 实…　Ⅱ. 王…　Ⅲ. 软件工程－高等学校：技术学校－教材　Ⅳ. TP311.5

中国版本图书馆 CIP 数据核字（2009）第 096971 号

责任编辑：谭文芳
出版发行：清华大学出版社　邮编：100084　电话：010－62776969
　　北京交通大学出版社　邮编：100044　电话：010－51686414
印 刷 者：北京交大印刷厂
经　　销：全国新华书店
开　　本：185×260　印张：15.5　字数：398 千字
版　　次：2009 年 9 月第 1 版　2009 年 9 月第 1 次印刷
书　　号：ISBN 978－7－81123－589－0/TP·498
印　　数：1～4000 册　定价：26.00 元

本书如有质量问题，请向北京交通大学出版社质监组反映。对您的意见和批评，我们表示欢迎和感谢。
投诉电话：010－51686043，51686008；传真：010－62225406；E-mail：press@bjtu.edu.cn。

前　言

软件工程对软件产业的形成和发展起着决定性的推动作用，在计算机的发展和应用中至关重要，已成为新兴信息产业的支柱。随着计算机技术的发展及应用的普及，软件工程已经为人们所熟悉并得到广泛的应用。大量的实践证明，在软件开发过程中必须遵循软件工程的原则，采用软件工程的方法和工具，才能有效地保证软件的质量，满足用户的要求。

现在，软件工程课程已经成为计算机相关专业的必修课，是一门理论与实践相结合的专业课。本书通过对传统的面向过程的软件开发方法和面向对象的软件开发方法的介绍，使读者掌握开发高质量软件的方法；通过对软件开发过程和过程管理技术的学习，使读者了解如何进行软件管理，怎样进行质量保证活动，从而掌握软件设计、开发的能力。

作者建议在学习软件工程课程之前，应该掌握一两门高级程序设计语言、数据结构、操作系统和数据库技术等方面的知识。在学习过程中不但应注重概念、原理、方法和技术的掌握，也应注重方法、技术的实际应用。

本书参考教学时数为 50～60 学时。全书共 10 章：第 1 章讨论软件工程基本概念、软件过程模型；第 2 章介绍可行性研究的内容和方法；第 3 章介绍需求分析的定义、原则、方法和工具；第 4 章介绍系统总体设计的原则、原理及模块设计方法及优化准则；第 5 章讨论系统详细设计的内容和技术；第 6 章介绍面向对象的开发方法；第 7 章讨论软件编程语言的选择、程序设计风格及容错软件设计的基本方法；第 8 章介绍软件测试的基本方法及面向对象系统测试的方法；第 9 章介绍软件维护的定义、流程和技术；第 10 章讨论软件项目管理的基本内容。

本书的第 1～5 章由抚顺职业技术学院的王爱平老师编写，第 6～8 章由抚顺职业技术学院的王宏亮老师编写，第 9～10 章由抚顺职业技术学院的张庚老师编写，抚顺职业技术学院的李丹老师参与了本书的校对工作。

本书在编写过程中，参阅了很多国内外同行的著作和文章，汲取了该领域最新的研究成果。在此，对这些成果的作者表示深深的感谢！

由于水平和时间的限制，书中不可避免会出现一些错误，请各界同仁不吝赐教。

编　者

2009 年 6 月

目　录

第 1 章　软件工程概述

计算机技术发展到今天，软件已经成为限制计算机技术发展与应用的关键问题，“软件危机”仍然在时常地困扰着我们。软件工程这一学科的创立为解决“软件危机”提供了良好的途径和科学的方法。

1.1　软件的概念及特征

随着计算机硬件的飞速发展，软件也从规模、功能、性能方面得到了巨大的发展，同时人们对软件质量的要求也越来越高。那么，什么是软件？软件有哪些特征呢？

1.1.1　软件定义

随着计算机知识的普及，大多数用户都在一定程度上对软件有一些了解。很多人认为软件就是一个计算机程序，这种理解是很不完全的。现在一般认为软件由三部分组成：①能够完成预定功能和性能的一组计算机程序；②能被充分操作的数据结构；③描述程序设计和使用的文档。简明地把软件表示为：“软件 = 程序 + 数据 + 文档”。

程序是为了解决某个（些）特定问题而用程序设计语言描述的适合计算机处理的语句序列。它们是由软件开发人员设计和编码产生的，通常开发人员编制的程序源代码要经过编译，才能生成计算机可执行的机器语言指令序列。数据是软件的处理对象，数据的形式是多种多样的。程序在执行时，一般要输入一定的数据，也会输出中间结果和最终结果。文档是软件开发设计过程中各种活动的记录，主要供开发人员和用户阅读。这里所说的文档既用于开发人员和用户之间的通信和交流，也用于软件开发过程的管理和运行阶段的维护。为了提高软件开发的效率、提高软件质量、便于软件开发过程的管理及软件的维护，现在软件开发人员越来越重视文档的作用及其标准化工作。我国参照国际标准陆续颁布了有关软件开发的文档规范。

1.1.2　软件的特征

要对软件有一个全面的理解，必须了解软件的特征。软件具有如下特征。

1. 软件是一种逻辑实体，具有抽象性

软件与计算机硬件或其他工程对象有着明显的差别。虽然人们可以将软件记录在纸面上，或是保存在计算机的存储器里，或是存储在磁盘、磁带、光盘等存储介质中，但无法看到软件的形态，只有通过分析、思考、判断，或是运行软件去了解它的功能、性能及其他特性。

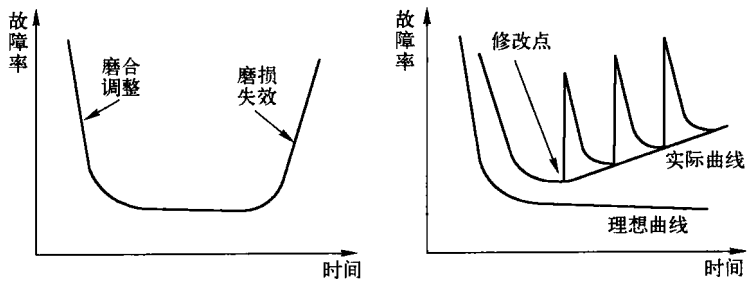
2. 软件开发的成本比较高

与硬件生产制造相比，软件的开发是人的智力的高度发挥，更依赖于开发人员的素质、

智力，以及人员的组织、合作和管理。在软件的开发过程中没有明显的制造过程，也不像硬件那样，在制造过程进行质量控制，以保证产品的质量。软件是通过人们的智力活动，把知识与技术转化成一种产品。所以，对软件的质量控制必须着重于其开发过程。就硬件而言，其成本往往只占整个产品成本的一小部分，而软件开发成本比较高，通常占整个产品的大部分。

3. 软件不会“磨损”

任何硬件设备在运行和使用过程中，都会随着时间的推移产生机械磨损和老化。其故障率变化曲线俗称“浴盆曲线”，如图 1－1（a）所示。硬件在投入使用初期，各部件尚未做到配合良好、运转灵活，容易出现缺陷和问题；经过一段时间运行，随着这些缺陷和问题不断被发现和修正，故障率会减少，最后，稳定在一个较低的水平；经过相当长的时间运行使用，设备会出现磨损和老化，故障率越来越大，最终设备被报废。软件与硬件有很大区别，它没有磨损、老化问题，但存在退化问题，其故障率曲线如图 1－1（b）所示。在软件运行使用过程中，为了使软件克服尚未发现的缺陷、使它更好地适应硬件和软件环境的变化及用户需求的变化，必须对软件进行各种维护（修改），而这些维护（修改）必不可免地引入新的错误，导致软件故障率不断地升高，使得软件退化，最终被放弃。



（a）硬件故障率曲线　　（b）软件故障率曲线

图 1－1　故障率曲线

4. 软件开发和运行对计算机系统有依赖性

计算机系统虽然是由硬件和软件两部分组成的，但是，软件不能脱离硬件而单独使用。任何软件的开发和使用都是以硬件提供的条件为依据的。有的软件这种对硬件的依赖性很大，常常为某个型号的计算机所专用，这给软件的使用带来许多不便。为了解除这种依赖性，人们在软件开发中提出了软件可移植性的问题，并把软件的可移植性作为衡量软件质量的要素之一。

5. 软件的开发是一个复杂的过程

软件要实现的功能是人类大脑的部分功能或是某部分功能的加强。软件所反映的实际问题是复杂的，例如，它所反映的自然规律，或现实社会中的各种事务，都有一定的复杂性。软件开发，特别是应用软件的开发常常涉及多个领域的知识。因此，软件必然是复杂的，其开发过程也必然是复杂的。

6. 软件的开发至今依然采用人工的开发方式

软件产品大多是根据特定用户的需求“定做”的，很少能做到利用现成的部件组装而

成，手工开发方式仍然是目前采用的主要开发方式。对于软件开发人员来说，开发工作是一种高强度的脑力劳动。

1.2　软件危机与软件工程

伴随硬件的发展，软件也走过了六十多年的历程。然而，这是一个坎坷的历程，软件危机始终伴随其中，因此说，软件的发展过程也是人们认识软件、解决软件危机的过程。

1.2.1　软件的发展

自 1946 年出现第一台计算机以来，软件就一直随着计算机硬件的发展而发展，经过几十年的发展，人们对软件有了更加深刻的认识。在这几十年中，软件经历了三个阶段。

1. 初始阶段（20 世纪 50 年代中期至 20 世纪 60 年代中期）

这一阶段，软件规模小，编写者和使用者往往是同一个人，完全依靠个人的程序技术和经验完成编程工作，除程序清单外，没有其他文档资料。

2. 发展阶段（20 世纪 60 年代中期至 20 世纪 70 年代中期）

随着计算机硬件的发展，软件的规模逐渐增大，功能日益增多，软件也由简单的程序发展为程序系统，开发人员也由个人发展成软件开发小组。但是，软件的开发仍然沿用早期形成的个体化软件开发方法，软件技术的发展不能满足需求，开发的软件要么开发时间过长，成本过高，要么其实现的功能和性能不能满足用户的要求，要么软件在运行过程中维护困难，不能适应环境的变化。这时出现了“软件危机”。

3. 成熟阶段（20 世纪 70 年代中期以后）

“软件危机”极大地阻碍了软件的应用与软件技术的发展，如何解决这一问题呢？人们开始重新认识软件。系统论和系统工程的创立为人们打开了一个新视野，也为解决“软件危机”提供了一个新途径。软件的发展从此进入成熟阶段。在这一阶段，计算机科学家正式提出“软件工程”这个名词，诞生了软件工程学。软件技术走上了一条科学、快速发展的道路，各种软件开发方法不断地应用于实践，取得了丰硕的成果。

纵观软件发展历程，可以看出，软件工程是人们不断总结实践经验，运用系统科学的理论和方法创立的，它是软件开发人员必须掌握的一门学科。

1.2.2　软件危机

软件危机是指计算机软件在它的开发和维护过程中所遇到的一系列严重问题。概括地说，主要包含两方面的问题：如何开发软件，怎样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。

软件危机的主要表现如下。

1. 软件开发经费预算常常很不准确，经常突破，完成时间一再拖延

在软件开发过程中，常常出现不能准确地估算所需经费和开发进度，实际费用往往比估计的费用高出很多，实际进度比预期进度拖延几个月的现象。这种现象降低了开发者的信誉，又由于进度和费用的缘故，软件产品的质量得不到保证，甚至被损害，从而引起用户的

不满。

2. 完成的软件常常不能完全满足用户要求

软件开发人员往往在对用户需求没有深入的分析，正确的理解，甚至对所要解决的问题还没有确切认识的情况下，就匆忙着手编写程序。在软件开发过程中，软件开发人员和用户之间的没有充分交流，结果导致最终的软件产品不符合用户需要，甚至与用户的要求南辕北辙。

3. 软件的质量低，可维护性差

在软件的开发过程中，缺乏有效的软件质量保证措施，不重视软件可维护性设计，使得程序中的错误很难改正，甚至不能改正，也不能根据用户的需求在原有程序中增加新的功能。造成软件产品质量低，可维护性差，容易发生质量问题。

4. 软件常常没有完备的文档资料

过去，人们头脑中的软件就是程序，这种对软件的不完全的认识导致软件开发人员只重视程序本身，不注重相关的文档资料，使得软件产品中没有一套完整的文档资料。其结果必然给软件的开发和维护带来许多严重的困难和问题。

5. 软件成本在计算机系统总成本中所占比例逐年上升

随着电子技术的进步和生产自动化程度的提高，计算机硬件成本逐年下降，然而，由于软件规模的增大，功能的增强，数量的增多，使软件开发需要大量的人力，软件成本随之不断扩大而逐年上升。有关调查表明，软件成本大约已占计算机系统总成本的90%。

软件危机产生的内在原因可归纳为两方面：一方面是由软件生产本身存在着复杂性，另一方面是与软件开发所使用的方法和技术有关。具体原因主要有以下几点：

① 软件的规模越来越大，结构越来越复杂；

② 软件开发管理困难而复杂；

③ 软件开发费用不断增加；

④ 软件开发技术落后；

⑤ 生产方式落后；

⑥ 开发工具落后，生产率提高缓慢。

软件危机促使人们开始对软件及其特性从新认识，进行更深入的研究，人们改变了早期对软件的不正确看法。现在人们普遍认为优秀的程序除了功能正确、性能优良之外，还应该容易看懂、容易使用、容易维护和扩充。为了解决软件危机，既要有技术措施，又要有必要的组织管理措施，软件工程正是从技术与管理两个方面研究如何更好地开发和维护计算机软件的一门新学科。

1.2.3 软件工程

1. 软件工程的概念

软件工程这一概念，主要是针对20世纪60年代“软件危机”而提出的。1968年第一届NATO（北大西洋公约组织）会议上，第一次提出了软件工程的概念，并给出了软件工程的一个早期定义：“软件工程就是为了经济地获得可靠的且能在实际机器上有效地运行的软

件，而建立和使用完善的工程原理。”这个定义明确地指出了软件工程的目标是经济地开发出高质量的软件，并强调了软件工程是一门工程学科，它建立并使用完善的工程原理。

此后，围绕软件项目，开展了有关开发模型、方法以及支持工具的研究，并且围绕项目管理提出了费用估算、文档复审等方法和工具。综观 20 世纪 60 年代末至 80 年代初，其主要特征是：前期着重研究系统实现技术，后期开始强调开发管理和软件质量。

1993 年 IEEE（国际电气电子工程师学会）进一步给了一个更全面更具体的定义：“软件工程是：①把系统的、规范的、可度量的途径应用于软件开发、运行和维护过程，也就是把工程应用于软件；②研究①中提到的途径。”

2. 软件工程的基本原理

自从 1968 年提出“软件工程”这一概念以来，计算机软件的专家学者们对软件进行了大量的研究和实践，陆续提出了 100 多条关于软件工程的准则。美国著名的软件工程专家 Boehm 综合这些专家的意见，1983 年提出了软件工程的七条基本原理。Boehm 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。它们是相互独立的，是缺一不可的最小集合；同时，它们又是相当完备的。

（1）用分阶段的生命周期计划严格管理

这是总结、吸取前人的教训而提出来的。统计表明，50% 以上的失败项目是由于计划不周而造成的。在软件开发与维护的漫长生命周期中，应该把整个软件生命周期分成若干阶段，明确各阶段的目标和任务，并相应制订出切实可行的阶段计划，然后严格按照计划对软件的开发和维护进行管理。Boehm 认为，在整个软件生命周期中应制订并严格执行 6 类计划：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。

（2）坚持进行阶段评审

通过对大量软件及软件缺陷的研究，人们发现大多数软件缺陷并非来源于编程错误。导致软件缺陷最大的原因是产品说明书（占 55%）；第二个原因是设计方案（占 25%）；第三个原因是代码（占 15%）；第四个原因是某些软件缺陷产生的条件被错误地认定（占 5%）。因此，在软件开发与维护的各阶段中，应该将软件质量保证工作贯彻始终，坚持进行阶段评审，尽早发现存在的问题，实现早期报警。

（3）实行严格的产品控制

虽然，用户需求的变化往往是不可避免的，但在软件开发过程中应该采用科学的产品控制技术来顺应这种需求的变化。也就是要采用变动控制，又称为基准配置管理。当需求变动时，其他各个阶段的文档或代码随之相应变动，以保证软件的一致性。

（4）采纳现代程序设计技术

自软件工程这一概念提出以来，围绕软件项目管理、程序设计技术等，专家学者进行了一系列的研究，提出了瀑布模型，开发了一些结构化程序设计语言、结构化设计方法等。并进一步研究了各种先进的软件开发与维护技术。实践表明，采用先进的技术可以提高软件开发与维护的效率，减少软件维护的成本，提高软件产品的质量。

（5）结果应能清楚地审查

软件是一种看不见、摸不着的逻辑产品。软件开发小组的工作进展情况可见性差，难于准确的度量，更难于评价和管理。因此，为了提高软件开发过程的可见性，更好地进行全过程的管理，应该根据软件开发的总目标及完成期限，尽量明确地规定开发小组的责任和产品

标准，从而使所得到的结果能清楚地审查。

（6）开发小组的人员应少而精

开发人员的素质和数量是影响软件质量和开发效率的重要因素，因此，开发小组人员的素质应该好，而人数不宜过多，要做到少而精。

这一条基于两点原因：第一，高素质开发人员的效率比低素质开发人员的效率要高几倍到几十倍，开发工作中犯的错误也要少得多；第二，随着开发小组人数的增加，因为交流情况而产生的通信开销也会急剧增加。

（7）承认不断改进软件工程实践的必要性

遵从上述六条基本原理，就能够较好地实现软件的工程化生产。但是，它们只是对现有的经验的总结和归纳，并不能保证赶上技术不断前进发展的步伐。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原理。根据这条原理，不仅要积极采纳新的软件开发技术，还要注意不断总结经验，收集进度和消耗等数据，进行出错类型和问题报告统计。这些数据既可以用来评估新的软件技术的效果，也可以用来指明必须着重注意的问题和应该优先进行研究的工具和技术。

3. 软件工程的基本原则

人们在研究、应用软件工程的过程中，围绕工程设计、工程支持以及工程管理，提出了以下四项基本原则。

（1）选择适宜开发模型

在系统设计中，软件需求、硬件需求以及其他因素之间是相互制约、相互影响的，经常需要权衡。因此，必须认识需求定义的易变性，应该采用适宜的开发模型进行控制，以保证软件产品能满足用户的需求。

（2）采用合适的设计方法

在软件设计中，通常要考虑软件的模块化、抽象与信息隐蔽、局部化、一致性以及适应性等特征。合适的设计方法有助于这些特征的实现，以达到软件工程的目标。

（3）提供高质量的工程支持

在软件工程中，软件工具与环境对软件过程的支持非常重要。软件工程项目的质量与开销直接取决于对软件工程所提供的支撑质量和效用。

（4）重视开发过程的管理

软件工程的管理，直接影响可用资源的有效利用，生产满足目标的软件产品，提高软件组织的生产能力等问题。因此，只有当软件过程得到有效管理时，才能实现有效的软件工程。

总之，软件工程的目标是可用性、正确性和经济性；实施一个软件工程要选取适宜的开发范型，采用合适的设计方法，提供高质量的工程支撑，实行开发过程的有效管理；软件工程活动主要包括需求、设计、实现、确认和支持等活动，每一活动可根据特定的软件工程，采用合适的开发模型、设计方法、支持过程以及过程管理。软件工程学科的研究内容主要包括：软件开发模型、软件开发方法、软件过程、软件工具、软件开发环境、计算机辅助软件工程（Computer Aided Software Engineering，CASE）及软件经济学等。

1.3　软件过程模型

软件过程模型就是把软件开发过程、开发方法以及开发工具和软件生命周期结合起来的结构框架，是软件开发各阶段之间关系的集合，是软件工程的重要内容。它为软件工程管理提供了进度表，为软件开发过程提供了原则和方法。

1.3.1　软件的生命周期

世界上任何事物都要经历形成、发展、衰落、消亡的过程。一个软件从它发生到消亡的过程被称为软件的生命周期。软件生命周期一般划分为三个阶段：软件定义、软件开发和软件维护。软件生命周期及软件生命周期中的主要活动如图 1－2 所示。

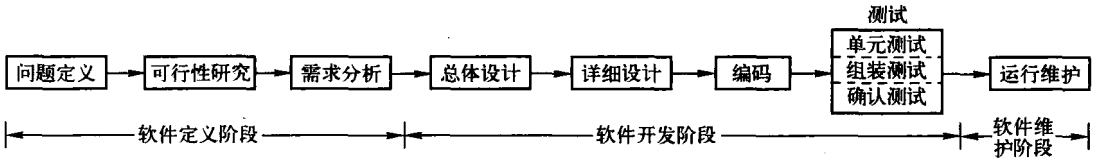


图 1－2　软件生命周期

软件定义阶段的主要任务是确定软件开发工作必须完成的总体目标；通过调查、分析确定软件的功能、性能、质量等方面的要求；对完成该软件项目需要的资源、成本、效益、进度做出估计，并制订出实施计划。软件定义通常由问题定义、可行性研究和需求分析三个部分组成。

软件开发阶段的主要任务是根据软件的总体目标和确定软件的功能、性能、质量等方面的要求具体设计和实现所定义的软件。通常由总体设计、详细设计、编码和测试四部分组成。其中前两个部分统称为系统设计，后两个部分统称为系统实现。

软件维护阶段的主要任务是保证软件正常地运行，完成规定的功能，持久地满足用户的需要。维护阶段通常不再进一步划分。

1. 问题定义

这一阶段主要是确定“用户要解决什么问题?”。首先由用户提出需要解决的问题，再由系统分析人员通过分析和对问题的理解，提出关于问题性质、软件目标及规模的报告，请用户审查认可。

2. 可行性研究

可行性研究是在初步调查、分析及开发方案构想的基础上，运用技术经济理论与方法，分析软件开发的必要性和开发方案的可行性，得出是否继续开发的明确结论，并对新软件系统实现的投入与产出作出全面的评估。可行性研究是在较抽象的层次上进行的分析与设计过程，其主要任务不是解决问题，而是确定给定的软件项目是否有解。

可行性分析的实质是进行一个大大压缩简化的软件分析与设计过程，也就是在较高层次上、以比较抽象的方式进行软件分析和设计。因此，可行性分析应该由有经验的分析人员来进行。它是在系统初步调查的基础上，分析现行系统存在的问题及目标系统与现行系统之间

的差别，构思新系统的初步方案。可行性研究着重考虑以下五个方面：

① 经济可行性；

② 技术可行性；

③ 组织可行性；

④ 社会及法律可行；

⑤ 风险因素及对策。

可行性研究的结果是“可行性研究报告”，它是用户决策是否继续进行该软件项目的重要依据。

3. 需求分析

确定软件开发可行后，需要对软件实现的各个功能进行详细分析。因此，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。需求分析就是运用系统的观点和方法，对软件需要实现的各个功能、环境和数据进行详细分析，从而得出软件目标和功能模型的过程。需求分析阶段的根本任务是准确地回答“为满足用户的需要系统必须做什么?”，即尽可能理解和表达用户对软件的需求，调查现行系统的资源、输入、处理和输出，确定新软件的基本目标和逻辑功能要求，建立成新软件的逻辑模型。需求分析的结果用“软件需求说明书”的形式准确地表达出来，它包括对软件的功能需求、性能需求、环境约束和限制条件、外部接口等描述。“软件需求说明书”既是对用户认可的软件系统逻辑模型的描述，也是进行软件设计的依据。另外，在整个软件开发过程中，用户的需求是不断变化和深入的，因此必须制订需求变更计划来适应这种变化，以保护整个项目的顺利进行。

4. 总体设计

系统总体设计是要根据需求分析的结果和用户的实际情况对新系统的总体结构形式进行的设计，是宏观上的规划。在需求分析的阶段，系统已经对系统“做什么”很清楚了，总体设计阶段就是确定系统“怎么做”，即将软件的用户需求合理地转变成目标系统的体系结构、模块逻辑、接口特征和数据结构等。

5. 详细设计

详细设计阶段的根本目标是根据总体设计阶段得到的软件模块结构，给出软件模块的内部过程描述，确定应该怎样具体地实现所要求的系统。这一阶段的任务不是具体地编写程序，而是要设计出程序的“蓝图”，以后程序员将根据这个蓝图编写出实际的程序代码。因此，详细设计的结果基本上决定了最终的程序代码的质量。

6. 编码

编码阶段是软件开发过程的核心，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。编程语言的性能和编码风格，对软件的质量和维护性能有很大的影响。因此，在程序编码中必须要制定统一、符合标准的编写规范。以保证程序的正确性、可读性、易维护性，提高程序的运行效率。

7. 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