普通高等教育“十三五”规划教材

（计算机专业群）

软件工程

主　编　田保军　刘利民
副主编　张林丰　张丽霞　许志伟

本书微课资源

中国水利水电出版社
www.waterpub.com.cn

普通高等教育“十三五”规划教材（计算机专业群）

软件工程

主　编　田保军　刘利民

副主编　张林丰　张丽霞　许志伟

中国水利水电出版社
www.waterpub.com.cn

·北京·

内 容 提 要

本书是一本基于丰富案例的软件工程实用教程，利用软件工程核心三要素——方法、工具和过程——贯穿全文，重点介绍了软件工程的基本概念、原理、软件工程国家相关规范和软件工程文档撰写国家标准以及传统软件工程方法学和面向对象方法学。

本书重点介绍当前主流的面向对象软件工程的开发方法，UML 与建模方法、工具以及统一过程 RUP。通过实例突出讲述面向对象分析、设计和实现流程。本书所有的概念、开发方法都通过实例来演示，内容精炼、表达简明、实例丰富，非常适合用作高等院校软件工程专业、计算机科学与技术专业及相关专业本科生、研究生的教材，也可以作为培训机构相关专业的培训教材和广大科技工作者、研究人员的参考用书。

本书视频资源可扫书中二维码观看。

图书在版编目（CIP）数据

软件工程 / 田保军，刘利民主编. -- 北京 : 中国水利水电出版社，2019.4
普通高等教育“十三五”规划教材. 计算机专业群
ISBN 978-7-5170-7596-7

Ⅰ. ①软… Ⅱ. ①田… 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(2019)第069197号

策划编辑：石永峰　责任编辑：张玉玲　加工编辑：王玉梅　封面设计：李　佳

书　名	普通高等教育“十三五”规划教材（计算机专业群） 软件工程 RUANJIAN GONGCHENG
作　者	主　编　田保军　刘利民 副主编　张林丰　张丽霞　许志伟
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	三河市鑫金马印装有限公司
规　格	184mm×260mm　16 开本　19 印张　468 千字
版　次	2019 年 4 月第 1 版　2019 年 4 月第 1 次印刷
印　数	0001—2000 册
定　价	46.00 元

前　言

软件工程作为支撑软件产业的一级学科，其发展正方兴未艾。软件工程课程是 SWEBOK 软件工程知识体系中一门基础、核心课程。软件工程课程涉及的内容广泛，其涉及的各项技术和项目管理方法对于即将从事 IT 产业的学生来说是非常重要的。但是由于种种原因，对于这门课程许多学生认为比较空洞乏味。本书结合作者多年的教学和工程实践经验，参阅大量国内外有关软件工程的教材和资料，遵循“理论为基础、实用为目的”的原则，理论联系实践，编写本书。

本书着重从实用角度出发，讲解目前软件工程比较成熟的、广泛使用的两大方法学：结构化方法学和面向对象方法学。本书以软件生命周期为主线，利用软件工程核心三要素——方法、工具和过程——贯穿全文，主要讲解了软件工程概述，结构化方法、工具和过程，可行性与计划研究，需求分析，软件设计，软件实现，软件运行与维护，软件过程管理，面向对象的软件工程概述，面向对象分析，面向对象设计与实现，同时介绍了当今流行的软件工程建模语言和工具，例如面向对象统一建模语言 UML、软件绘图工具 Microsoft Visio、数据建模工具 PowerDesigner、面向对象建模工具 Rational Rose 与 StarUML、测试工具 LoadRunner、Quality Center 与 QuickTest Professional 等。同时，为了帮助学生通过“做中学”的模式掌握扎实而实用的软件工程技术，本书以学生学籍管理系统为项目案例，贯穿全文。

本书内容丰富，组织结构严谨，原理、方法与案例相结合，讲解由浅入深，既体现知识点的连贯性、完整性，又体现知识在实际项目中的应用，适合作为高等院校软件工程专业、计算机科学与技术专业及相关专业本科生、研究生的教材，也可以作为培训机构相关专业的培训教材和广大科技工作者和研究人员的参考用书。

本书在内容的编排、语言的叙述等方面都有其自身的一些特点：

（1）内容系统全面，结构清晰。全书分为三大部分：面向过程的软件工程和面向对象的软件工程，按照软件生命周期的各个阶段分别进行讲述。

（2）描述简明易懂。本书从基本概念和原理出发，注重内容的可理解性，深入浅出，循序渐进；文字描述通俗易道，简明扼要，重点突出。

（3）注重案例分析。以学生学籍管理系统为案例贯穿全文是本书的最大特色。本书克服以往软件工程教材重理论、轻实践或案例少且知识点内容不连续的缺点，不仅增加了案例数量，而且保持案例的连续性，使读者更容易掌握相关知识。

（4）每章列出了学习目标和小结，配有精选的适量习题，便于读者对所学内容的复习和理解。

本书由田保军老师、刘利民老师任主编，张林丰老师、张丽霞老师和许志伟老师任副主编。参加编写的老师分工如下：田保军老师编写第 5、10、11 章；刘利民老师编写第 1、3 章；张林丰老师编写第 6、7、8 章；张丽霞老师编写第 4、9 章；许志伟老师编写第 2 章和附录。全书由田保军老师、刘利民老师统稿。

本书参考和引用了许多教材、著作和网站内容，除了确实无法查证出处以外，本书在参考文献中都一一列出，在此表示衷心感谢。张志林、胡皎月、王宇、胡培培等研究生在教材的编写过程中，也做了不少工作，一并表示感谢。限于时间和水平有限，书中不够完善乃至缺点和错误之处，恳请专家学者提出宝贵意见，以便我们再版时进行修订补充，使之日臻完善。

编 者

2018 年 12 月于内蒙古工业大学

目　　录

第 2 篇 面向对象的软件工程

第 1 篇　面向过程的软件工程

第 1 章　软件工程概述

本章学习目标

- 熟练掌握软件工程概念、原理和方法。
- 熟练掌握软件生命周期和软件过程模型。
- 掌握软件危机产生的原因、解决的办法。
- 掌握软件及其特点。
- 了解软件工程的发展历程。
- 了解软件工程相关规范。

本章首先向读者介绍软件工程的发展历程，然后介绍软件及其特点、软件危机产生的原因和解决的办法，继而介绍软件工程原理和方法，最后介绍软件工程相关规范。

1.1　软件工程的发展历程

随着计算机的广泛应用，软件在计算机系统中的地位越来越重要。市场需要的软件逐年增多，而且趋向大型化和复杂化，软件变得越来越复杂，软件开发人员越来越满足不了实际需要，软件产品的质量也变得难以满足各方面的要求，加上软件生产率低，软件成本上涨等，出现了一系列严重问题，导致软件危机产生。

软件产业在工业发达国家中占有非常重要的地位。软件危机的产生使人们认识到软件开发必须以新的方法作指导，必须改变原有的软件开发方法。1968 年北大西洋公约组织（North Atlantic Treaty Organization，NATO）在联邦德国召开国际会议，讨论软件危机问题，第一次提出了“软件工程”这个概念。

随着软件技术的发展，软件工程的研究范围和内容也在不断变化和发展。其发展大致经历了传统软件工程、过程软件工程、构件软件工程等三个阶段。

第一阶段，传统软件工程阶段。20 世纪 70 年代，为了解决软件项目错误率高以及软件维护任务重等问题，人们提出软件开发工程化的思想，希望使软件开发走上一条规范化的道路，并努力克服软件危机。软件工程的概念、框架、方法和手段因此形成。

第二阶段，过程软件工程阶段。20 世纪 80 年代末逐步发展起来的面向对象方法，形成了完整的面向对象技术体系，适应更大规模、更广泛的应用。这时，进一步提高软件生产率、保证软件质量成为软件工程追求的更高目标。人们认识到，应从软件生命周期的总费用及总价值

来决定软件开发方案。在重视发展软件开发技术的同时，提出了软件能力成熟度模型、个体软件过程和群组软件过程等概念。软件开发过程从目标管理转向过程管理。

第三阶段，构件软件工程阶段。20 世纪 90 年代以后，软件开发技术的主要处理对象为网络计算和支持多媒体信息的 WWW。为了适应超企业规模、资源共享、群组协同工作，需要开发大量的分布式处理系统。这时软件工程的目的在于不仅提高个人生产率，而且通过支持跨地区、跨部门、跨时空的群组共享信息、协同工作来提高群组的整体生产效率。因整体性软件系统难以更改、难以适应变化，所以提倡基于部件（构件）的开发方法。同时人们认识到计算机软件开发领域的特殊性，不仅要重视软件开发方法和技术的研究，更要重视总结和发展包括软件体系结构、软件设计模式、互操作性、标准化、协议等领域的重用经验。软件重用和软件构件技术正逐步成为主流软件技术。迭代、敏捷、持续集成的理念，正成为业界的共识。

随着软件产业的发展，软件工程必将朝着流水线装配软件工程的方向发展，以迎接软件蓬勃发展的趋势。流水线生产、网络化、服务化与全球化将成为主流，移动互联成为趋势。

1.2 软件危机

1.2.1 软件的概念、特点及分类

1. 软件的概念

计算机软件无所不在，已成为人们生活中的一部分。长期以来，人们关于软件的许多问题存在着一些争议：

在软件开发周期上，为什么需要那么长时间才能结束软件开发？

在软件的成本上，软件的成本为什么如此之高？

在软件的错误上，为什么我们不能在把软件交给客户之前就发现所有的错误？

在软件的度量上，为什么在软件开发过程中我们难以度量其进展？

……

以上这些，其实都与软件及其自身特点有着必然的联系。

一般公认的软件定义是：软件是计算机系统中与硬件相互依存的另一部分，它是程序、数据及其相关文档的完整集合。程序是按事先设计的功能和性能要求执行的指令序列，而数据是使程序能正常操纵信息的数据结构，文档则是与程序开发、维护和使用有关的图文材料。这三部分构成了软件，缺一不可。

1983 年，美国电气与电子工程师学会（Institute of Electrical and Electronics Engineers，IEEE）为软件作了如下定义：软件是计算机程序、方法、规则、相关的文档资料以及在计算机上运行程序时所必需的数据。

也有不少人认为，软件除了上述三部分，还应包括知识。软件都是为特定领域服务的，领域知识也是需要软件开发人员掌握的。

2. 软件的特点

软件作为一种产品，是一种不同于物质性产品的逻辑性产品，与其他制造类产品有着本质的不同。它除具有一般产品的诸多属性外，还具有自己特有的属性。具体表现在如下几个方面：

（1）软件形态的逻辑性。软件是一种逻辑实体，而不是具体的物理实体。因而它具有抽象性、不可见性。可以将软件存储在不同的介质上，但却无法直接看到软件的形态，开发过程的进度也难以衡量，质量难以评价，必须通过观察、分析、判断，利用抽象思维能力去了解软件的功能、性能和其他特性。软件的管理和控制相当困难。

（2）生产过程的非制造性。软件的生产过程与硬件不同，在它的开发过程中没有明显的制造过程。

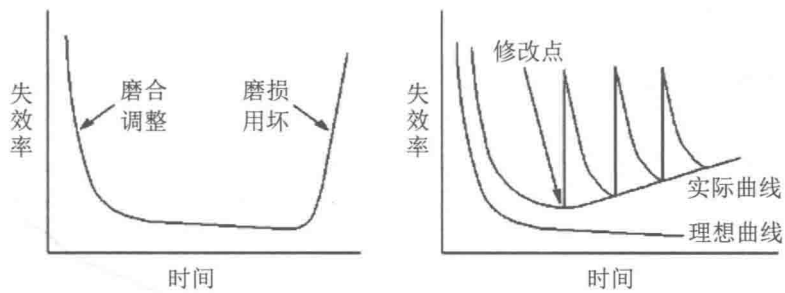
软件是由工程师开发或工程化形成的，而不是传统意义上的制造产生的产品副本。软件成本、质量集中于开发和研制上，这意味着软件项目不能像硬件制造项目那样进行管理。

但软件一经确定，需要批量生产时，相对容易，可以复制产生大量产品副本。即它可以被无限复制成若干副本。

（3）使用方式的无磨损性。在软件的运行和使用期间，没有硬件那样的机械磨损、老化问题。软件不会被“用坏”“用旧”，也不会因为长时间运行而被损耗或产生新的故障，不过它的功能可能会发生退化，需要开发人员不断地更新和维护。随着时间的推移和环境的变化，软件最终可能会被废弃。

此外。当一个硬件构件磨损时，可以用另外一个备用零件替换它，但对于软件则不然。每一个软件故障都表明了设计/编程中存在错误。因此，软件较难维护，维护意味着改正或修改原来的设计。

如图 1.1 所示为硬件与软件失效率经验曲线，其中硬件的失效率曲线又称 U 型曲线（或浴缸曲线）。而软件一开始需要一段时间磨合期，之后就可以稳定运行，随后逐渐退化。



（a）硬件失效率曲线　　（b）软件失效率曲线

图 1.1　硬件与软件失效率经验曲线

（4）开发和运行对环境的依赖性。软件的开发和运行常常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。或者对硬件有一定依赖，或者对特定的操作系统有一定依赖。为了解除或减少这种依赖性，在软件的开发过程中提出了软件的可移植性和跨平台问题。

（5）开发方式的手工化。软件的开发至今尚未完全摆脱传统的手工艺开发方式。大多数软件产品都是定制的，而不是通过已有的构件组装而来的。随着面向对象技术、构件技术等的发展，情况略有好转。但由于软件开发是一种高强度脑力劳动，开发人员必须利用自己的智力去理解需求，并综合运用软件技术提高开发效率和质量，因此软件开发还无法完全实现自动化。

（6）软件的复杂性。软件本身是非常复杂的，软件是人类能够创造的最复杂的产物。主要体现在实际问题的复杂性、程序逻辑结构的复杂性、其他领域专门知识的复杂性。

例如，下述软件所花费的人力资源相当大。

1）Windows 95 操作系统有 1000 万行，花费上千人。

2）Windows 2000 操作系统有 5000 万行，花费 5000 多人。

3）Lotus 1-2-3 V3.0 统计制表软件总计有近 40 万行，花费 263 人·年，成本 2000 多万美元。

4）WWMCCS（全球军事指挥与控制系统）花费 3500 多人，拖了几年，交付后发现 100 多个错误。最后失败。

因此如果把一个简单程序看作是独唱的话，那么一个较复杂程序就可以看作小合唱，而一个复杂的软件产品则可以看作合唱，甚至大合唱。可见它们的复杂程度有着显著的区别。

（7）成本的昂贵性。软件成本相当昂贵。在软件开发过程中，会投入大量的、复杂的、高强度的脑力劳动，投入的研发成本相对比较高。

随着计算机硬件技术的不断发展，在一个计算机系统中软件成本的比例在逐步增加，而硬件成本的比例逐步下降。如图 1.2 所示为软件成本的经验曲线。据统计，目前软件成本已占到了计算机系统的 90%以上。

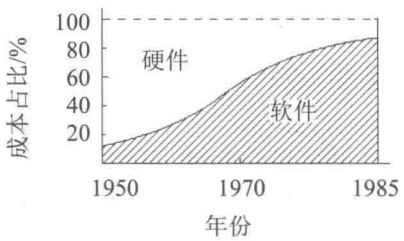


图 1.2　成本时间曲线

（8）社会因素。相当多的软件工作涉及社会因素。软件大大提高了工作的效率，必然释放劳动力，促进机构的重组，由此带来一系列的社会问题。

许多软件的开发和运行涉及组织机构设置、体制运作及日后管理方式等问题，甚至涉及企业文化、使用人员的观念和心理等，这些因素会直接影响软件项目的实际应用效果，甚至软件的成败。

3. 软件的分类

软件的分类有许多种方法，可以按软件的功能划分，也可以按软件的工作方式划分，或者按软件的规模划分。此处只介绍按软件的功能进行分类。

按功能划分，软件可以分为如下三类：系统软件、支撑软件和应用软件。

（1）系统软件。系统软件是负责计算机系统中各种独立的硬件部件、相关软件和数据，使得它们可以协调高效地开展工作的软件。系统软件使得用户和其他软件将计算机当作一个整体而不需要顾及底层每个硬件是如何工作的。例如操作系统、数据库管理系统、设备驱动程序、通信处理程序等软件均属于系统软件。

（2）支撑软件。支撑软件是为方便用户使用而开发的各种工具软件，是一种通用性较强的软件。例如支持需求分析、设计、实现、测试和支持管理的工具软件，微软公司的支持各种图表制作的 Visio 软件、SYBASE 公司的数据库设计建模工具 PowerDesigner 软件等均属于支撑软件。

（3）应用软件。各行各业都需要软件，但解决的问题不同。为解决特定应用领域问题

而开发的软件，称为应用软件。例如商业数据处理软件、工程与科学计算软件、计算机辅助设计/制造软件、系统仿真软件、智能产品嵌入软件、医疗制药软件、办公自动化软件、计算机辅助教学软件等均属于应用软件。

1.2.2　软件危机

随着软件规模的扩大，软件产品越来越复杂，在软件的开发和维护过程中不可避免地出现了一些问题。比如在软件运行过程中发现了错误，软件人员必须设法去改正；用户在软件使用过程中有新的需求，软件人员必须相应地修改程序；当计算机系统的硬件或操作系统发生了更新，通常也需要相应地修改软件。上述的软件维护工作，以令人吃惊的比例消耗着资源。甚至，有些软件是不可维护的，于是出现了软件危机。软件危机就是在计算机软件开发、维护过程中所遇到的一系列严重问题，导致软件的开发、维护出现风险。

最典型的是美国 IBM 公司在 1963 年至 1966 年开发的 IBM360 机的操作系统。这一项目花了 5000 人·年的工作量，最多时有 1000 人投入开发工作，写出了近 100 万行源程序。据统计，这个操作系统每次发行的新版本都是从前一版本中找出 1000 个程序错误而修正的结果。

这个项目的负责人 F. D. Brooks 事后总结了他在组织开发过程中的沉痛教训，在撰写的《人月神话》中写道："……正像一只逃亡的野兽落到泥潭中做垂死的挣扎，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成为软件开发项目的典型事例为人们所记取。

如果开发的软件隐含错误，可靠性得不到保证，那么在运行过程中很可能对整个系统造成十分严重的后果，轻则影响到系统的正常工作，重则导致整个系统的瘫痪，乃至造成无可挽回的恶性事故。例如：1963 年，美国用于控制火星探测器的计算机软件中的一个 "," 号被误写为 "."，而致使飞往火星的探测器发生爆炸，造成高达数亿美元的损失。

软件危机通常包含下述两方面的问题：

（1）如何开发软件，以满足对软件日益增长的需求。

（2）如何维护数量不断膨胀的已有软件。

1.2.3　产生软件危机的原因

计算机硬件技术的进步，要求软件及其相关技术能与之相适应。然而，软件技术未能快速满足社会发展的要求，致使一系列问题堆积起来，形成日益尖锐的矛盾，最终导致了软件危机。

软件危机产生的原因是多方面的，既有软件自身的特点，也有软件开发过程中采用的一些错误方法和技术。

1. 软件自身的特点

软件作为逻辑产品，具有与硬件不同的特性（见 1.2.1 节）。这些特性决定了软件开发与维护的复杂性，导致软件的开发和维护比硬件更难。

2. 在开发和维护过程中，采用了错误的方法和技术

（1）软件管理相对落后。由于缺乏软件开发的经验和相关数据的积累，使得开发工作的计划很难制订，以致经常超出经费预算，无法遵循进度计划，完成开发的期限一再拖延。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的现象并不罕见。这种现象降低了软件企业的信誉。而为了赶进度和节约成本所采取的一些权宜之计又往往损害了软件产品的质量，从而不可避免地引起用户的不满。

（2）软件需求不够清晰。软件需求在开发的初期阶段不够明确，或是未能得到确切的表达。软件开发人员常常在对用户需求只有模糊的了解，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，急于求成。开发工作开始后，软件人员和用户之间的沟通不够及时充分，造成矛盾在开发后期集中暴露。“闭门造车”必然导致最终的产品不符合用户的实际需要。

（3）软件测试不够充分。未能在测试阶段充分做好检测工作，提交给用户的软件质量差，在运行中暴露出大量的问题，轻则影响系统的正常工作，重则发生事故，甚至造成生命财产的重大损失。软件可靠性和质量保证确切的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。

（4）软件开发方法相对陈旧。早期的个体化开发特点，开发过程没有统一的、公认的方法论和规范指导，加上不重视文字资料工作，资料很不完整。忽视每个人与其他人的接口部分，发现了问题修修补补，这样的软件很难维护。很多程序中的错误是非常难改正的，不可能使这些程序适应新的硬件环境，也不能根据用户的需求在原有程序中增加一些新的功能。

（5）没有软件生命周期概念。认为软件就是编程、运行，轻视软件维护这个重要环节。其实编程工作量只占总工作量的20%左右，软件维护的工作量占到50%以上。

（6）忽视开发代价与时间的关系。软件开发过程中，在不同的时间段引入同一变动付出的代价具有明显的不同，引入变动的时间越晚付出的代价越高，代价随时间变化趋势如图1.3所示。

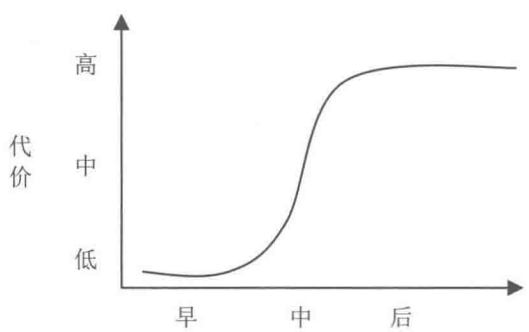


图1.3 代价随时间变化趋势

一个复杂的软件系统需要建立庞大的逻辑体系，而这些往往只存在于人们的头脑中。一个大项目的负责人曾经说过这样一段话：“软件人员太像皇帝新衣故事中的裁缝。当我来检查软件开发工作时，所得到的回答好像对我说：我们正忙于编织这带有魔法的织物，只要一会儿，你就会看到这件织物是极其美丽的。但是我什么也看不到，什么也摸不到，也说不出任何一个有关的数字。没有任何办法得到一些信息说明事情确实进行得非常顺利。我知道许多人最终只是编织了一大堆昂贵的废物而已，还有不少人最终什么也没有做出来。”

如果这些问题不能很好地解决，软件的发展是没有出路的。如何保证软件产品的质量，是非常复杂困难的问题，特别是对于规模庞大的软件尤为重要。

1.2.4　解决软件危机的方法

消除软件危机的途径，主要有下述几个方面。

1. 首先应该正确的认识计算机软件

“软件”不等于单一的“程序”，软件=程序+数据+相关文档，是程序、数据和文档的完整集合。

1983 年 IEEE 为软件作了如下定义：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程序时所必需的数据。方法和规则在文档中说明并在程序中实现。

2. 按工程化的原则和方法组织项目开发是软件开发的一个主要出路

20 世纪 50 年代到 20 世纪 60 年代时，程序设计曾经被看作是一种任人发挥创造才能的技术领域。写出的程序通篇充满了程序技巧，这些程序很难被别人看懂。然而随着计算机技术的发展和广泛使用，人们逐渐抛弃了这种观点。人们要求这些程序容易看懂、容易使用，并且容易修改和扩充。多个软件人员分工合作、共同完成，只能在项目的总体要求和技术规范的约束下充分发挥和施展自己的才能。

3. 通过技术和管理的方法手段提升软件的质量

必须意识到：软件具有自己的生命周期。大型软件系统的开发与其他工程项目如建造桥梁，制造飞机、轮船等的开发是同理的。必须采用工程化思想，运用先进的技术方法和管理手段来提升软件的开发质量和效率。

1.3　软件工程

1.3.1　软件工程的概念

1. 软件工程的概念

伴随着计算机科学技术的进步和软件产业的发展，软件工程已由最初的一个学科方向发展成为一个以计算机科学技术为基础的新兴交叉学科。同时，软件工程方法学的研究与实践也大大丰富了计算机科学，促进了软件产业的发展。

软件工程是指导计算机软件开发和维护的一门学科。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，这就是软件工程。

1968 年 NATO 给出了一个定义，软件工程是为了经济地获得可靠的且能在实际机器上运行的软件，而建立和使用完善的工程原理。

1993 年 IEEE 给出了一个更加综合的定义，软件工程是开发、运行、维护和修复软件的系统方法，软件工程包括两个方面：

（1）将系统化的、规范的、可度量的方法应用于软件的开发、运行和维护的过程，即将工程化应用于软件中。

（2）对（1）中所述方法的研究。

其他软件专家也提出了一些有关软件工程的定义，大同小异，就是采用工程化的思想和方法开发软件。

2. 软件工程的本质特性

（1）软件工程关注于大中型程序的构造。大中型程序相对比较复杂，因此有必要采用系统的工程理论与方法，规范软件的开发和维护过程，以保障软件的质量。

（2）软件工程的中心课题是控制复杂性。软件所解决的问题十分复杂，以致于不能把问题作为一个整体通盘考虑，只能采用模块化分解的方法解决。模块化方法并不能降低问题的整体复杂性，但是却使它变成可以管理的问题。许多软件的复杂性主要不是由问题的内在复杂性造成的，而是由必须处理的大量细节造成的。

（3）开发软件的效率非常重要。现实世界在不断地变化，软件为了不被很快地淘汰，必须随着所模拟的现实世界一起变化。目前，社会对新应用系统的需求超过了人力资源所能提供的限度，软件供不应求的现象日益严重。

（4）和谐地合作是开发软件的关键。软件处理的问题十分庞大，必须许多人协同工作才能解决这类问题。纪律是成功地完成软件开发项目的一个关键，良好的团队合作有利于项目的成功。软件开发人员必须学会合作与沟通，才能保证软件的质量和开发效率。

（5）软件必须有效地支持它的用户。软件提供的功能应该能有效地协助用户完成他们的工作。在软件工程领域中是由具有一种文化背景的人（计算机人员）为另一种文化背景的人（行业领域的人员）创造产品。缺乏应用领域的相关知识，是软件开发项目出现问题的常见原因。软件工程师不仅缺乏应用领域的实际专业知识，还缺乏该领域的文化知识。因此有必要通过访谈、阅读书面文件等方法了解用户组织的“正式”工作流程，然后用软件实现这个工作流程。

3. 软件工程项目的基本目标

一般而言，成功的软件工程项目希望达到以下几个目标：

（1）成本低，付出比较低的开发成本。

（2）功能全，达到用户要求的软件功能。

（3）性能好，取得较好的软件性能。

（4）易移植，开发的软件易于移植。

（5）费用少，需要较低的维护费用。

（6）能按时完成开发工作，及时交付使用。

在实际开发过程中，如果想让以上几个目标都达到理想的程度，往往是非常困难的，而且这些目标之间也可能发生矛盾。软件工程的这些目标之间有些是互斥的，有些是互补的，软件开发项目力图在以上目标的冲突中取得一定程度的平衡。

1.3.2 软件工程原理

著名软件工程专家 B. W. Boehm 于 1983 年综合研究了软件工程的专家与学者们的意见，提出了软件工程的七条基本原理，他认为这七条原理是保证软件产品质量和开发效率的原理的最小集合。

1. 用分阶段的生命周期计划严格管理

把软件生命周期划分为若干个阶段，并相应的制订计划，然后按计划对软件的开发与维护工作进行管理。整个生命周期中主要执行六类计划，分别是项目概要计划、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

在每个阶段都要进行严格的评审，以便尽早发现在软件开发过程中所犯的错误。各个阶段之间存在着必然的联系，尽早地发现前一阶段错误有助于后续阶段工作的继续，保证每个阶段的工作质量。

3. 实行严格的产品控制基准和配置管理

在软件开发过程中，改变需求往往会付出较大的代价。为了能使需求改变的代价降到最低，必须实行严格的产品控制，其中主要是实行基准配置（各个阶段产生的文档或程序代码）管理。当对基准配置进行修改时，要按照严格的规程进行评审，只有获得批准后才能实施修改。

4. 采用现代程序设计技术

采用先进的程序设计技术，不仅可以提高软件的开发效率，而且可以提高软件的维护效率。比如采用面向对象的程序设计技术，就可以有效提高软件的重用性和可扩展性，进而提高软件的开发效率和可维护性。

5. 结果应能清楚地审查

为了能更好地管理软件开发过程，规定开发组织的责任和产品标准，每个阶段所得到的结果都具有里程碑的意义，应该能够清楚地审查。

6. 开发小组的成员应该少而精

开发人员的水平参差不齐，应该不断提高开发小组人员的素质。人员多了必然带来沟通和接口等一系列问题，势必影响软件的开发进度和质量效果。

7. 承认不断改进软件工程实践的必要性

软件工程是实践者的工程方法，在实践的过程中，不可避免地会出现一些新的问题，需要软件开发人员不断地改进，以适应需求的变化。

二八定律又叫帕雷托法则，是 19 世纪末20 世纪初意大利经济学家巴莱多发现的。此法则指在众多现象中，80%的结果取决于 20%的原因。他认为，在任何一组东西中，最重要的只占其中一小部分，约 20%，其余 80%尽管是多数，却是次要的，因此又称二八定律。生活中普遍存在“二八定律”。

近几年，软件工作者总结了项目开发过程中的经验，提出了软件工程的二八定律。这些经验丰富了软件工程的思想。

（1）对软件项目进度和工作量的估计：一般主观上认为已经完成了 80%的，往往实际上只完成了 20%。

（2）对程序中存在的问题的估计：80%的问题往往存在于 20%的程序之中。

（3）对模块功能的估计：20%的模块实现了 80%的功能。

（4）对人力资源的估计：20%的人解决了软件中 80%的问题。

（5）对投入资金的估计：企业信息系统中 80%的问题可以用 20%的资金来解决。

1.3.3　常用的软件工程开发方法

1. 软件工程方法学

在软件生命周期全过程中会使用一整套的技术方法，这些方法的集合称为软件工程方法学。