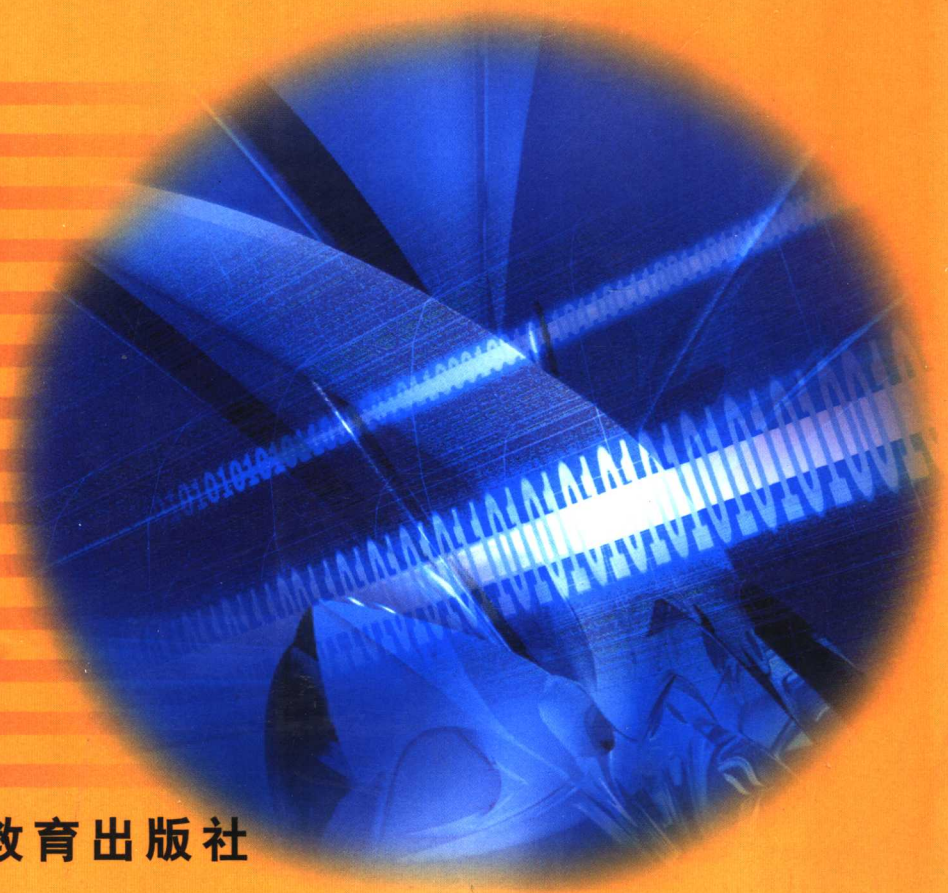
高等职业教育技能型紧缺人才培养试用

软件工程概论

陶华亭　主　编
梁建新　副主编



高等教育出版社

高等职业教育技能型紧缺人才培养试用

软件工程概论

陶华亭　主　编
梁建新　副主编

同济大学出版社

内容提要

本书是高等职业教育技能型紧缺人才培养试用教材。

本书概括论述了软件工程思想、软件工程方法论、软件生存周期理论、软件开发模型、传统的软件工程方法、结构化软件工程方法和面向对象的软件工程方法。对于继承、复用、UML、面向对象的建模、软构件等概念进行了详细介绍。全书共分 12 章。前 6 章介绍了软件工程的基本理论和基本概念及传统的和结构化的软件工程方法，第 7 章作为面向对象方法的引论，详细对比讨论了各种软件工程方法论的特点以及面向对象方法论的优点和开发背景。后 5 章主要讨论面向对象的软件工程方法，并且与 UML 结合起来，在 UML 的技术背景下介绍和讨论了面向对象软件工程的相关概念和方法。在介绍面向对象技术的过程中，没有机械地重复调研、分析、设计、编码、测试等各阶段的全部内容，而是重点剖析了面向对象方法的特殊之处。

全书内容紧凑，深入浅出，实例贯穿始终，突出实用。

本书适合于高等职业学校、高等专科学校、成人高校、本科院校举办的二级职业技术学院，以及示范性软件职业技术学院、继续教育学院、民办高校、技能型紧缺人才培养作为教材使用，还可供本科院校、计算机专业人员和爱好者参考。

图书在版编目(CIP)数据

软件工程概论 / 陶华亭主编. —北京：高等教育出版社，2004.11

ISBN 7-04-015689-X

Ⅰ. 软… Ⅱ. 陶… 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（2004）第 099217 号

策划编辑 冯 英 责任编辑 彭立辉 封面设计 王凌波 责任绘图 朱 静
版式设计 胡志萍 责任校对 金 辉 责任印制 韩 刚

出版发行 高等教育出版社
社 址 北京市西城区德外大街 4 号
邮政编码 100011
总 机 010 - 58581000

购书热线 010 - 64054588
免费咨询 800 - 810 - 0598
网 址 http://www.hep.edu.cn
http://www.hep.com.cn

经 销 新华书店北京发行所
印 刷 高等教育出版社印刷厂

开 本 787 × 1092 1/16
印 张 14.5
字 数 350 000

版 次 2004 年 11 月第 1 版
印 次 2004 年 11 月第 1 次印刷
定 价 18.50 元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换。

物料号：15689 - 00

前　言

众所周知，软件危机导致了软件工程的诞生。20 世纪 60 年代，北大西洋公约组织成员国的专家们，主张学习传统工业中工程化的方法和管理手段，像生产工业制品一样开发软件，以缓解软件危机，并提出了“软件工程”这一概念。

经过了 30 多年的风雨历程，作为一门学科，软件工程有了很大的发展并形成了不同方法、策略、工具和技术。它是来自实践者的经验和总结并在规范和指导着软件的开发活动。

学习软件工程，应悟其思想，抓其精髓，创造性地应用。软件工程是实践者的技术，它的概念、方法、策略和规范都不能死搬硬套。一个好的软件工程师的超人之处，往往体现在好的软件工程思想、创造意识和灵活应用的技能。

方法论是软件工程理论体系的大的支脉。不同的方法论来源于不同的时代和应用环境，不可同日而语，不可横向比较，更不可偏废，综合、创造性的应用才是最好的出路。只要不偏离软件工程思想，博采众家，综合应用，创造性地发挥，对软件工程学科的发展都是有益的。尤其不要以为某种方法论是传统的就束之高阁。本书以传统的、结构化的、面向对象的方法为线索组织内容，基本上是越靠后的方法，越能适应现在的开发环境。

本书从第 7 章到第 12 章主要讲述面向对象的软件工程方法并统一到 UML 平台上，占用了将近一半的篇幅，主要为了强调面向对象的方法和 UML 的重要性。在使用时，要根据教学要求，灵活选择。可以结构化方法为重点，面向对象的方法作为选修；也可以面向对象的方法为重点，结构化的方法占用尽量少一些的时间。

本书由陶华亭主编，梁建新任副主编，参编人员有陶华甫、胡耀东、邓大治、余文奇。另外，本书在编写过程中，得到了徐其兴等老师的大力支持，他们为本书提出了大量好的建议，孙国成教授对本书稿进行了全面审阅，在此一并表示衷心感谢！由于水平所限，时间紧张，书中错误之处在所难免，敬请业界专家、学者批评指正。

陶华亭

tao@zzmy.net.cn

2004 年 7 月

郑重声明

目　录

第1章 绪 论

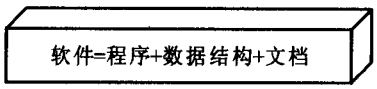
在详细讨论软件工程技术之前，本章首先介绍软件工程学科的基本内容，包括：学科背景、软件生存周期理论、软件开发模型和软件工程方法论等。这些知识是后续章节的基础。

学习目标：了解软件工程的学科背景，灵活理解和掌握软件生存周期理论。该理论的重要贡献是提出了要对软件开发过程进行阶段性划分，至于如何划分则要灵活掌握。软件开发模型是在软件生存周期理论指导下，通过实践不断探索而来的，是指导软件开发活动的理论参考，应灵活理解和应用并在实践中不断发展。

1.1 软件工程的产生

1.1.1 什么是软件

关于软件，目前一种理解是“软件=程序+数据结构+文档”，“程序”不是软件的全部。这里强调了文档的重要性，它是软件的一个重要组成部分。



- 程序：是计算机中可以被执行并用来完成处理任务的指令部分。
- 数据结构：是程序在执行过程中对数据进行存储、计算、读取的规则。
- 文档：是关于程序功能、设计、编码和使用说明的文字或图形资料。

软件是一种特殊的产品，搞清楚软件开发与一般产品制造的区别，对全面理解软件工程很重要。软件具有以下特点：

① 软件是一种逻辑产品。制造硬件时，经过分析、设计、制造、测试等过程，能够得到具体的、有形的实物；而软件是通过脑力劳动得到的看不见摸不着的逻辑的产品。

② 软件和硬件的制造过程不同。软件是经过创造性的开发活动得到的，而不是传统意义上的加工、制造生产的。两者的制造过程有着本质的不同：首先，虽然都可以通过良好的设计获得高质量的产品，但硬件在制造过程中可能会引入操作质量和精度质量问题，这种情况对于软件而言几乎不存在（或很容易改正）。其次，软件成为产品之后，其制造只是简单的拷贝而已。软件成本集中于开发环节上，这意味着软件项目不能像硬件制造项目那样来管理。

需要指出的是，虽然20世纪80年代中期提出了“软件工厂”的概念，但应该注意到这个术语并没有把硬件制造和软件开发认为是等价的，而是通过软件工厂这个概念提出了软件开发中应该使用工厂的理念和自动化的工具。

③ 软件只会退化，不会“磨损”，也不会被用坏和消耗。硬件在运行过程中存在磨损。随着时间的推移，硬件的故障率表现出如图1-1所示的曲线变化，这就是著名的“浴盆曲线”，它表明硬件在其使用初期具有较高的故障率，这些故障主要是设计或制造遗留的缺陷，这些缺陷

经过一段时间的磨合和维护之后，故障率会迅速下降。之后，故障率在一段较长的时间内会维持在一个较低的水平。随着运行时间的延长，到一定时期以后，故障率又会很快地上升。这是因为硬件构件由于种种原因会不断受到损害。

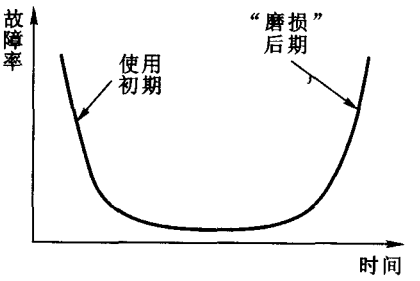


图 1-1 硬件故障浴盆曲线

理论上讲，软件的故障率曲线呈现如图 1-2 所示的形态。隐藏的错误会引起程序在使用初期有较高的故障率，但这些错误改正之后，故障曲线就会趋于平稳。

图 1-3 给出了实际的软件故障模型的一个简化图，其意义很清楚：软件不会磨损，但它会退化。在其生命期中，软件会经历若干次修改，每次修改都有可能引入新的错误，使得故障率骤然提高，表现为锯齿形曲线。在该曲线恢复到原来的稳定状态的故障率之前，又需要做新的修改，又会引起一个新的锯齿。慢慢地，最小故障率就开始提高了，这就是软件的“退化”，它是由于修改而引起的。

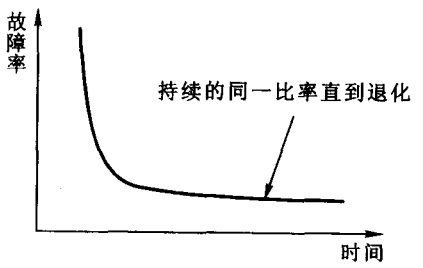


图 1-2 理想的软件故障曲线

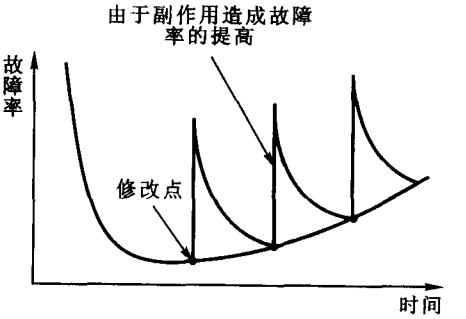


图 1-3 软件故障浴盆曲线

关于磨损的另一个侧面也表明了硬件和软件之间在制造和维护上的不同。当一个硬件构件磨损时，可以用另外一个备用零件替换，但软件没有可替换备件，每一个故障都表明了设计或编码的过程中存在错误。因此，软件维护要比硬件维护复杂得多。

④ 大多数软件是自建的，而不是通过已有的构件组装而来的。硬件设计制造过程，可以选用标准件手册中已有的构件来组装。比如，要生产一种型号的汽车，只需建造一个装配生产线即可，需要的零部件，都可以选择别的厂家生产好的。这里的成果，是建立在别人的劳动成果基础上的。在技术上，是对别人的研究成果的“继承”；在成果上，是对别人劳动成果的“复用”。从而降低了设计、开发成本。

在软件开发中，虽然现在也追求继承和复用，但继承和复用的难度很大。继承在面向对象的方法中得到了广泛的应用，但可复用的软件构件几乎等于零。硬件设计与生产效率的大幅度提高，主要得益于硬构件的复用。软件成本主要集中在设计开发阶段，目前这个阶段成本节约和效率提高主要靠继承。另外，软件构件的生产和管理还有一定的困难。

1.1.2 软件生产的发展

从第一台计算机诞生时起，就开始了软件的生产，到目前为止，已经经过了编程、软件开发和软件工程 3 个时代。

1．编程时代（1946—1956 年）

编程时代的生产方式是个体手工劳动，使用的工具是机器语言、汇编语言；开发软件时追求编程技巧，追求程序运行效率，代码不规范、难读、难懂、难修改；硬件特点是价格贵、容量小、速度慢、可靠性差；软件特点是只重视编程，不重视程序设计的方法。

2．软件开发时代（1956—1968 年）

软件开发时代提出了结构化设计方法，但仍旧靠个人技巧；小作坊式团队合作生产；使用高级语言编程；计算机性能不断提高，价格降低，普及速度加快；程序员数量猛增，但由于缺乏训练，开发人员素质差。这时软件需求不断增加，但开发技术没有新的突破，开发人员的素质和落后的开发技术不适应规模大、结构复杂的软件开发，矛盾日趋突出。

3．软件工程时代（1968 年以后）

软件工程时代提倡工程化生产，使用数据库、开发工具、开发环境、网络、分布式系统和面向对象的技术开发软件；硬件特征是向超高速、大容量、微型化以及网络化方向发展；软件特点是开发技术有很大进步，但未能获得新的突破。与软件市场需求的发展速度相比，软件工程仍然面临着许多问题。

1.1.3 软件危机

早期，在“软件开发”概念形成之前，大多数人认为“编程”是少数聪明人的事情，程序设计被看做是一门“艺术”，几乎没有规范化的方法。编程的专业性和挑战性，使其蒙上了一层神秘的色彩，管理人员很难控制开发过程。软件的开发活动很草率，没有任何管理措施，一旦进度延迟或成本超支了，程序员才开始手忙脚乱地弥补，完全处于一种无序的状态——编程好比是“捏泥巴”，程序员可以“为所欲为”。

在硬件技术逐渐趋于成熟和普及以后，编程的规模也越来越大，人们开始不得不重视软件的生产并形成了软件开发的概念。尽管如此，编程规模变大和软件需求膨胀还是带来了一系列问题。首先，这个时期，计算机应用系统的开发成本发生了戏剧性的变化，软件费用占了很大一部分比例，如图 1-4 所示。

与此同时，软件开发界陷入了一种莫名其妙的怪圈。

用户在抱怨：

- 为什么开发出来的软件不是我想要的东西。
- 开发一个软件就那么费劲吗？迟迟开发不出来，还那么贵。
- 软件质量太差，错误百出。

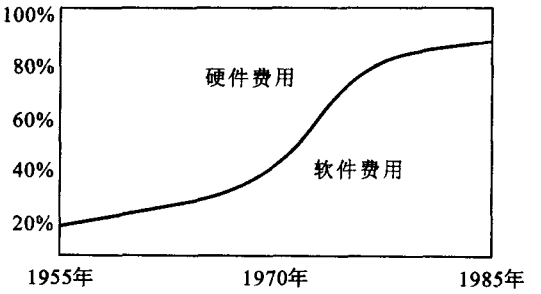


图 1-4 计算机应用系统中软、硬件成本变化趋势

- 业务变化了，面对缺少文档的程序，修改起来比重新开发还难。

……

软件开发商也在苦恼：

- 用户的需求为什么一直在变化，是讲不清，还是理解不了？
- 为什么需要那么长时间才能结束开发？成本太高了。
- 在把软件交给客户之前，难道我们就不能发现所有的错误吗？
- 沉重的售后服务负担，简直要把公司拖垮了。
- 为什么我们的开发过程难以度量，而且预算总是偏差那么大？

……

这种抱怨和苦恼的背后，就是所谓的“软件危机”的阴影，它给信息产业界带来了“剪不断，理还乱”的困扰。

“软件危机”发生的原因包括：

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。由于软件规模大，结构复杂，又具有无形性，导致管理困难，进度控制困难，质量和可靠性无法保证。

③ 软件开发费用不断增加。软件生产是一种智力劳动，它是资金密集、人力密集的产业，大型软件投入人力多，周期长，费用上升很快。

④ 软件开发技术落后。在 20 世纪 60 年代，人们注重计算机的理论问题研究，而对软件开发技术不太重视，因此，用户要求的软件复杂性与开发人员的能力不相适应，差距越来越大。

⑤ 生产方式落后。手工作坊式小团队开发，没有规范，开发过程管理落后。

⑥ 开发工具落后，生产效率提高缓慢。软件开发工具过于原始，没有出现高效率的开发工具，因而软件生产率低。1960—1980 年期间，计算机硬件的生产由于采用计算机辅助设计、自动生产线等先进工具，使硬件生产率提高了将近 100 倍，而同一个时期的软件生产率只提高了 2 倍。

事物总是在发展的，在这些阴影的笼罩下，却孕育着一个新时代的到来，那就是软件开发活动的工程化时代。

1.1.4 软件工程的诞生

“软件危机”形象地描述了软件业面临的挑战和严峻性，直到今天它仍然困扰着软件产业

的发展。为了克服软件危机，人们从其他产业的工程化得到启示，早在1968年，北大西洋公约组织就提出了“软件工程”的概念，提出要用“工程化”的思想方法和管理手段，像生产工业制造品一样开发软件，控制软件质量、提高软件产品生产率。从此，软件生产进入了软件工程时代。

30多年过去了，尽管软件的缺陷、漏洞还很多，但软件开发在质量、效率以及可靠性方面都实实在在地有了很大的提高。统计表明，软件产业的发展速度超过了任何传统工业，而且从未出现真正的软件危机，这要归功于“工程化”的思想和手段。如今通过不断总结经验和吸取教训，软件工程已经发展成了一门系统的学科。

软件工程学科形成的过程，也就是“工程化”思想创造性应用的过程。在软件开发活动中，引入“工程化”的思想并不是一件易事。面对看不见、摸不着的软件产品和创作式的脑力劳动形式，工程化思想无疑遇到了前所未有的尴尬，因为这里不同于工业品的加工，也不是建筑工程施工，照搬是不可行的。出路只有一个，那就是创造性地应用。**人们以工程化的思想为出发点，从不同的角度去研究、认识软件开发活动，诞生了不同的方法论。为支持各自的方法论，分别充实了相应的工具、方法，同时形成了应用这些工具和方法从事开发活动的技术**。如，结构化的方法、面向对象的方法等。所有这些努力，都为“软件工程”学科的形成和发展奠定了基础。它使得软件工程由刚提出时的一个单纯术语，发展并形成了一个内容详实，囊括了不同方法论、策略、工具和技术的完整的理论体系。

目前，软件工程在规范和指导着我们的开发活动并在应用中不断发展，而它发展创造的一个朴素的指导思想仍然是“工程化的思想”。由此可见，要想领悟软件工程的真谛，创造性地学习和发展，就必须牢固树立“工程化思想”的观点。

1.1.5 软件工程的概念

1．软件工程的定义

软件工程是一门针对软件生产，研究开发技术和开发活动管理技术的学科，旨在生产无故障的、能及时交付的、在预算之内又能满足用户需求的软件。实质上，软件工程就是采用工程的概念、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方法和最先进的软件开发技术结合起来，应用到软件开发和维护过程中，来解决软件危机问题。

1993年，IEEE为软件工程下的定义可以归纳为：软件工程是将系统化的、规范的、可度量的方法应用于软件的开发、运行和维护过程，即把工程化的思想应用于软件开发方法的研究。

该定义说明了软件工程是计算机科学中的一个分支，其主要思想是在软件生产中用工程化的方法代替传统手工方法。工程化的方法借用了传统的工程设计原理的基本思想，采用了若干科学的、现代化的方法技术来开发软件。这种工程化的思想贯穿到需求分析、设计、实现，直到维护的整个过程。

2．软件工程的性质

软件工程所涉及的理论极为丰富，包括计算机科学、工程学、管理学、数学等领域，着重于研究如何建造一个软件系统。软件工程要用工程学科中的观点进行费用估算、制定进度、制定计划和方案；要用到管理科学中的方法和原理进行软件生产过程的管理；要用到数学的方法建立软件开发中的各种模型和各种算法。

3．软件工程的目标

软件工程是一门工程性学科，目的是成功建造一个软件系统。软件工程学追求的总体目标可概括为：选择适当的方法论做指导，使用相应的工具做手段，运用成熟的技术从事软件开发活动，最终实现提高软件产品质量和开发效率，得到可靠性高的、经济适用的、易维护的软件产品。

为实现这个总体目标，在软件项目实施过程中，针对分析、设计、编程、测试、维护等具体的实践环节，人们又总结出了一系列具体的目标。如追求高可靠性、可适应性、有效性、可理解性、可互操作性、可重用性、可修改性、可追踪性和可维护性。这些目标是从事软件开发的人员在开发活动中力求做得更好的方面。这些目标不仅是分析活动、设计活动中追求的目标，也是衡量工作成果，评价产品质量优劣的目标。具体解释如下：

- 可靠性：一般认为，“可靠性”是指程序代码的可靠性。可靠性高的代码，在运行过程中就会少出错，避免给用户带来损失，甚至灾难性的后果。实际上，可靠性”是涉及软件开发全过程的一个目标，从可行性分析、用户需求定义、文档编写到设计、编程、测试、维护等各阶段的工作都要追求高可靠性才能最终保证软件产品的可靠性。
- 可适应性：软件在不同的系统约束条件下，使用户需求得到满足的难易程度。
- 有效性：软件系统在给定时刻根据规定成功运行的比率。
- 可理解性：系统具有清晰的结构，能直接反映用户的需求。
- 可互操作性：多个软件元素可以相互协调完成任务。
- 可重用性：软部件可以适应多种场合应用的程度。
- 可修改性：允许对系统修改而不增加复杂度。
- 可追踪性：根据软件需求对软件设计、程序设计进行正向追踪或逆向追踪的能力。
- 可维护性：软件产品交付用户后，能进行修改，以便改正潜伏的错误，改进性能和其他属性。
- 可移植性：软件从一个计算机系统或环境安装到另一个计算机系统或环境的难易程度。

需要特别指出的是，上述目标中，有些不是针对软件过程中某个环节的，而更可能是前后几个阶段共同努力才能做得更好的目标。如可靠性目标，不单是指程序代码的可靠性，而是要从可行性分析开始，每一个环节都要追求工作质量的可靠性，才能最终实现软件系统的可靠性。

4．软件工程的内容

软件工程研究的主要内容包括软件开发技术和软件开发过程管理技术两方面。在管理方面，主要研究软件项目管理的相关内容，体现为软件开发过程中的各项管理控制活动。在开发技术方面，它主要研究软件开发方法、软件开发过程、软件开发工具和技术。

“方法”是指方法论，是人们认识、理解和描述软件系统结构的一种思维模式，如结构化方法、面向对象的方法。不同的方法对软件系统的理解和描述是不同的。“工具”是针对不同的方法论研究的用于从事软件工程分析与设计的图表工具，如结构化方法中，有数据流图、功能结构图等；面向对象的方法有对象模型等。“技术”指研究如何应用“工具”从事分析设计。因此，过程、方法、工具和技术就构成了软件工程的内容体系结构。

5．软件工程的发展

按照不同时期人们对软件工程关注的焦点不同，可以将软件工程的发展过程分为如下几个阶段：传统的软件工程、结构化的软件工程、面向对象的软件工程、面向过程管理的过程软件工程、基于构件复用技术的构件软件工程。

1.1.6 软件工程的7条原理

① 应用生存周期的理论，分阶段地计划、管理和控制软件开发过程。

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。分阶段建立完善的计划的原则，是对前人经验、教训的总法。一个成功的参考作法是：把软件开发与维护的过程，分成3个时期、7个阶段。3个时期是软件定义期、软件开发期、软件维护期。7个阶段是调研、分析、概要设计、详细设计、编码、测试及维护。

② 书面文档是阶段性活动的成果，要坚持进行阶段性评审。

严格进行阶段性成果的评审，尽早发现软件开发过程中的错误，可以减少错误造成的损失。尤其是如果发现早期的一个错误，就可以避免在后期辐射出的成百上千个错误。

③ 对开发过程中的需求变动进行严格控制。

要充分认识到软件需求的变动性并采取适当的措施保证最终产品能充分地满足用户的需求。开发过程中，严格控制用户的需求变动，其中关键技术是实现基准配置管理，一切修改，特别是涉及对基准配置的修改，必须经过批准才能实施。

④ 采用先进的软件设计技术。

好的设计方法可以极大地方便软件开发，以达到软件工程的目标。软件设计中，通常考虑模块化、抽象与信息隐蔽、局部化、一致性等准则。这些是本书系统设计部分的重要内容。

⑤ 结果应能清楚地审查。

⑥ 开发小组的人员应少而精。

⑦ 承认不断改进软件工程实践的必要性。

1.1.7 4条指导性原则

① 可行性分析不仅是为新系统寻找可行方案，也包括否定项目的可行性。因此，项目经理在必要时应该果断地取消项目。

② 系统是为用户建立的，用户的需求才是开发人员追求的目标，不能把开发人员的喜好强加给用户。

③ 与用户联合开发的软件系统需得到用户方高层管理者的支持。

④ 开发过程不是线性的，允许重复、返回和增减活动。

为了实现软件工程的目标，在大量实践中，人们总结出了实施软件工程活动普遍适用的原理和指导性原则。不管采用什么样的方法论、什么开发模型以及什么工具和技术，这些原理和原则都是普遍适用的。

软件工程的理论方法是在实践中总结出来的，仍然需要在实践中通过创造性的应用不断发展。学习软件工程技术的人，最惨的也许不是因为学不会这些工具和技术，而是学会以后把它变成千古不变的教条，这才是最让人担心的。

1.1.8 软件工程面临的问题

软件工程有许多需要解决的棘手问题，如软件费用、软件可靠性、软件可维护性、软件生产率和软件重用等。

1．软件费用

由于软件生产基本上仍处于手工状态，是知识高度密集型产品，人力资源远远不能适应这种迅速增长的软件社会的要求，所以软件费用上升的势头必然还将继续下去。

2．软件可靠性

软件可靠性主要是指软件系统能否在既定的环境条件下运行并实现所期望的结果。在软件开发中，通常要花费40%的代价进行测试和排错，即使这样也不能保证交付后的软件不再发生错误。为了提高软件的可靠性，必然要付出足够的代价。

3．软件可维护性

统计数据表明，软件的维护费用占整个软件系统费用的2/3，而软件开发费用只占1/3。软件维护之所以有如此大的花费，是因为已经运行的软件还需要排除隐含的错误，新增的功能也要加进去，维护工作相当困难，效率也非常低。因此，如何提高软件的可维护性，减少软件维护的工作量，也是软件工程面临的主要问题之一。

4．软件生产率

计算机的普及，使得软件需求量大幅度上升，而软件的生产又处于手工开发状态，生产率很低，使得各国都感到软件开发人员不足。这种趋势将继续下去，所以，如何提高软件生产率，是软件工程的又一重要问题。

5．软件重用

提高软件的重用性，对于提高软件生产率、降低软件成本有着重要意义。当前的软件开发存在着大量的、重复的劳动，耗费了不少的人力资源。软件的可重用性，不仅仅是指编程时代码的可重用性，也包括分析成果和设计成果的可重用性。越是靠近前期的重用。越是高效的重用，更有助于提高质量和开发效率。但是软件重用的理论和技术至今尚未彻底解决。

1.2 软件生存周期理论

1.2.1 软件生存周期概念

引入软件生存周期概念，对于软件生产的管理、进度控制有着非常重要的意义，可使软件生产有相应的模式、相应的流程、相应的工序和步骤。

软件生存周期是指一个软件从提出开发要求开始,直到该软件报废为止的整个时期。把整个生存周期划分为若干阶段，使得每个阶段有明确的任务，把规模大、活动多、管理复杂的软件开发活动变得容易控制和管理。

软件规模、种类、开发方式、开发环境以及开发使用的方法都会影响软件生存周期的划分。在划分软件生存周期的阶段，应遵循的基本原则是各阶段的任务应尽可能相对独立，同一阶段的各项任务的性质尽可能相同，从而降低每个阶段任务的复杂程度，简化不同阶段之间的联系，

有利于软件项目的组织管理。通常，软件生存周期包括：系统调研、需求分析、概要设计、详细设计、编码、测试和维护等活动，可以将这些活动以适当的方式分配到不同阶段。

1．系统调研

系统调研是完成可行性研究和项目开发计划的阶段，必须回答的问题是“要解决的问题是什么”。该问题有行得通的解决办法吗？若有解决问题的办法，则需要多少费用？需要多少资源？需要多少时间？要回答这些问题，就要进行问题定义、可行性研究，制定项目开发计划。

用户提出一个软件开发要求以后，系统分析师首先要解决该软件项目的性质是什么，它是数据处理问题还是实时控制问题，它是科学计算问题还是人工智能问题等。还要明确该项目的目标是什么，该项目的规模如何等。

通过系统分析师对用户和使用部门负责人的访问和调查，开会讨论，就可解决这些问题。

在清楚了项目的性质、目标、规模后，还要确定该问题有没有行得通的解决方法。系统分析师要进行压缩和简化的需求分析和设计，也就是在高层次上进行分析设计，探索这个问题是否值得去解决，是否有可行的解决办法，最后要提交可行性研究报告。

经过可行性研究后，如果确定该问题值得去解决，就制定项目开发计划。根据项目的目标、功能、性能及规模，估计开发需要的资源，即需要的计算机硬件资源，需要的软件开发工具和应用软件包，需要的开发人员数目及层次。还要对软件开发费用做出估算，对开发进度做出估计，制定完成开发任务的实施计划。最后，将项目计划和可行性研究报告一起提交管理部门审查。

2．需求分析

需求分析阶段的任务不是具体地解决问题，而是准确地确定“软件系统必须做什么”，确定软件系统必须具备哪些功能。

用户了解他们所面对的问题，知道必须做什么，但是通常不能完整地、准确地表达出来，也不知道怎样用计算机解决他们的问题。而软件开发人员虽然知道怎样用软件完成人们提出的各种功能要求，但是，对用户的具体业务和需求不完全清楚，这是需求分析阶段的困难所在。

分析师要和用户密切配合，充分交流，充分理解用户的业务流程，完整、全面地收集、分析用户业务中的信息和对信息的处理，从中分析出用户要求的功能和性能并完整、准确地表达出来。这一阶段要给出软件需求说明书。

3．概要设计

在概要设计阶段，开发人员要把确定的各项功能需求转换成需要的软件体系结构，在该结构中，每个成分都是意义明确的模块，即每个模块都和某些功能需求相对应。因此，概要设计的核心内容就是设计软件的结构，弄清该结构由哪些模块构成，这些模块之间的层次结构是怎样的，每个模块的功能是什么。同时还要设计该项目的应用系统的总体数据结构和数据库结构，即应用系统要存储什么数据，这些数据是什么样的结构，它们之间有什么关系等。

4．详细设计

详细设计阶段就是为每个模块完成的功能进行具体描述，要把功能描述转变为精确的、结构化的过程描述。即该模块的控制结构是怎样的，先做什么，后做什么，有什么样的条件判定，有些什么重复处理等，然后用相应的表示工具把这些控制结构表示出来。

5．编码

编码阶段就是把每个模块的控制结构转换成计算机可接受的程序代码，即写成以某特定程序设计语言表示的“源程序清单”。当然，写出的程序应该结构好、清晰易读并且与设计一致。

6．测试

测试是保证软件质量的重要手段，其主要方式是在设计测试用例的基础上，检验软件的各个组成部分。测试分为模块测试、组装测试、确认测试。模块测试是查找各模块在功能和结构上存在的问题。组装测试是将各模块按一定顺序组装起来进行的测试，主要是查找各模块之间接口上存在的问题。确认测试是按软件需求说明书上的功能逐项进行的，发现不满足用户需求的问题，决定开发的软件是否合格，能否交付用户使用等。

7．维护

软件维护是软件生存周期中时间最长的阶段。已交付的软件投入正式使用后，便进入软件维护阶段，它可以持续几年甚至几十年。软件运行过程中可能由于各方面的原因，需要对其进行修改。其原因可能是运行中发现了软件隐含的错误而需要修改；也可能是为了适应变化了的软件工作环境而需要做适当变更；也可能是因为用户业务发生变化而需要扩充和增强软件的功能等。

以上划分的7个阶段是在GB8567中规定的，而实际工作中更多的是将生存周期划分为5个阶段，即分析、设计、编码、测试及维护。

1.2.2 软件生存周期法

软件生存周期法又称生存周期模型，它为我们提供了一种框架，一种对软件开发各项活动进行计划、组织、管理、协调与控制的实施模式。软件生存周期模型就是对软件生存周期方法学的运用，它强调使用软件生存周期理论组织开发活动。

人类解决复杂问题时普遍采用的一个策略是“分而治之”。这是一种先对问题进行分解，然后再分别解决各个子问题的策略。把软件开发过程按生存周期理论进行阶段划分，就是从时间角度对软件开发和维护的复杂问题进行了分解。

实践中，根据软件开发项目的进展情况不同，不同时期的主要任务和目标也不同。一般将软件的整个生存期划分为若干个阶段，每一阶段安排一系列活动并有明确的任务和目标。不同的应用领域，不同的软件项目对阶段的划分不尽相同，但大都包括软件定义、软件开发、软件维护3个时期。在具体应用中，每个时期又可进一步细分为若干阶段，如图1-5所示。

1．软件定义期

软件定义期的主要任务包括4项：

① 制定项目规划。包括：描述软件的范围、进行风险分析、提出开发软件所需的资源清单、估算软件项目的成本和进度，要求以成本和进度估算为基础，对软件项目进行可行性论证。最后，生成经过项目管理小组评审的软件项目规划。

可行性研究主要考虑经济性、可操作性和技术性等因素。回答类似于下面这样一系列的问题：

用户提出的新系统是否值得开发？新系统的开发、应用环境是否具备应有的条件？新系统应该开发成什么样的？新系统应实现多大规模？应该如何实现新系统？……

为这些问题寻找答案的过程就是可行性分析。如果确定新系统值得开发，就需要给出一个