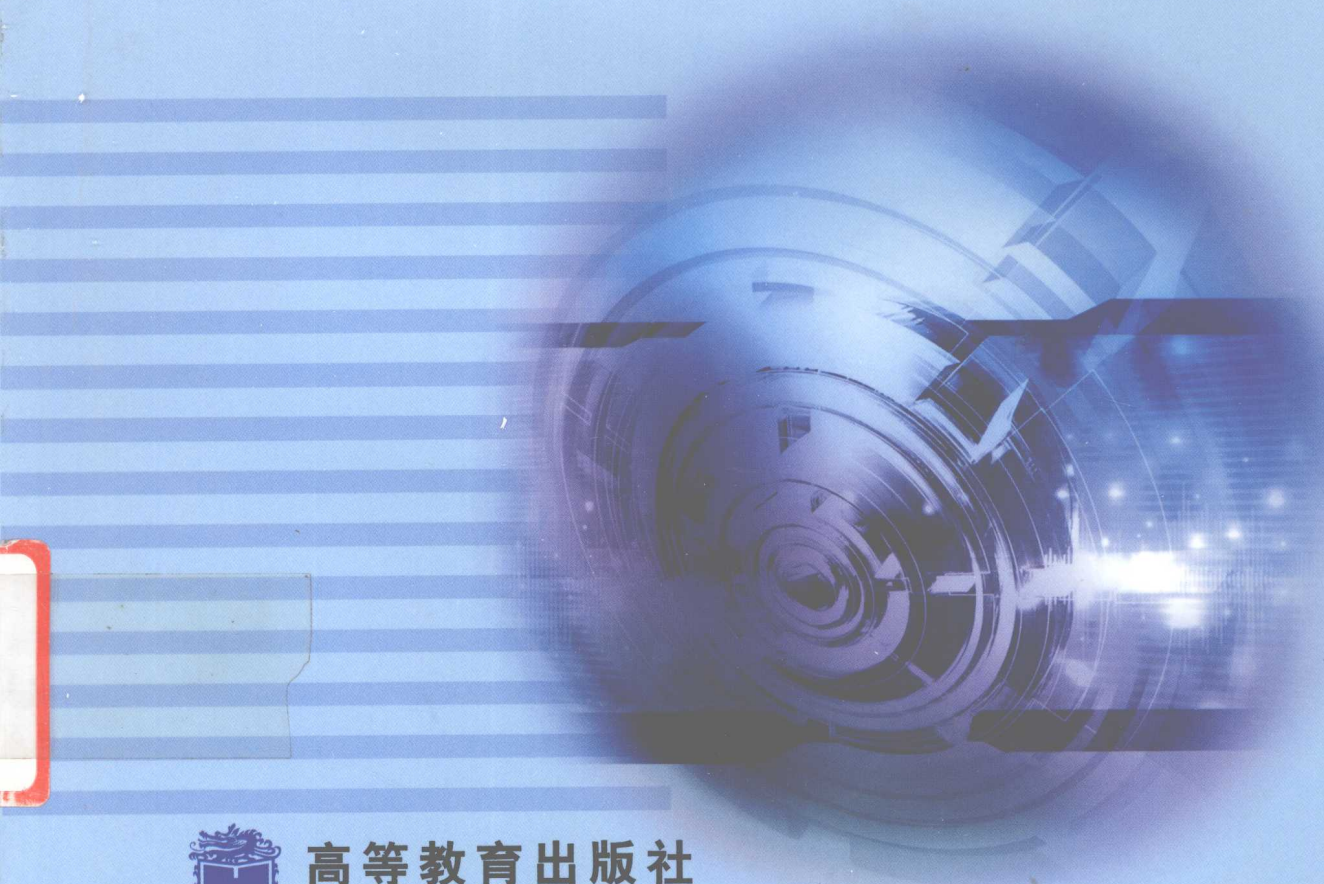
普通高等教育“十五”国家级规划教材

软件工程

曾强聪



高等教育出版社

普通高等教育“十五”国家级规划教材

软件工程

曾强聪

高等教育出版社

内容提要

本书是普通高等教育“十五”国家级规划教材。具有结构严谨、概念清晰、内容紧凑，深入浅出、突出实用、便于自学等特点。

全书内容紧凑，深入浅出。全书共10章正文，以软件生命周期为主线，主要内容包括：软件工程概论、软件工程过程模型、项目分析与规划、软件需求分析、软件概要设计、面向对象分析与设计、用户界面设计、程序算法设计与编码、软件测试、软件维护。附录部分包括软件文档管理规范和软件文档格式。

本书适合于高等应用型本科院校、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使用，也可供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使用，还可供本科院校、计算机专业人员和爱好者参考使用，并可用作软件技术人员资格（水平）考试的培训教材。

图书在版编目(CIP)数据

软件工程 / 曾强聪. —北京：高等教育出版社，2004.11

ISBN 7-04-015743-8

Ⅰ. 软… Ⅱ. 曾… 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105933号

策划编辑 冯 英 **责任编辑** 严 亮 **封面设计** 王凌波 **责任绘图** 黄建英
版式设计 胡志萍 **责任校对** 王效珍 **责任印制** 孔 源

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 北京市南方印刷厂

开 本 787×1092 1/16
印 张 14.75
字 数 360 000

版 次 2004年11月第1版
印 次 2004年11月第1次印刷
定 价 18.80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：15743-00

出版说明

为加强高职高专教育的教材建设工作，2000年教育部高等教育司颁发了《关于加强高职高专教育教材建设的若干意见》（教高司[2000]19号），提出了“力争经过5年的努力，编写、出版500本左右高职高专教育规划教材”的目标，并将高职高专教育规划教材的建设工作分为两步实施：先用2至3年时间，在继承原有教材建设成果的基础上，充分汲取近年来高职高专院校在探索培养高等技术应用性专门人才和教材建设方面取得的成功经验，解决好高职高专教育教材的有无问题；然后，再用2至3年的时间，在实施《新世纪高职高专教育人才培养模式和教学内容体系改革与建设项目计划》立项研究的基础上，推出一批特色鲜明的高质量的高职高专教育教材。根据这一精神，有关院校和出版社从2000年秋季开始，积极组织编写和出版了一批“教育部高职高专规划教材”。这些高职高专规划教材是依据1999年教育部组织制定的《高职高专教育基础课程教学基本要求》（草案）和《高职高专教育专业人才培养目标及规格》（草案）编写的，随着这些教材的陆续出版，基本上解决了高职高专教材的有无问题，完成了教育部高职高专规划教材建设工作的第一步。

2002年教育部确定了普通高等教育“十五”国家级教材规划选题，将高职高专教育规划教材纳入其中。“十五”国家级规划教材的建设将以“实施精品战略，抓好重点规划”为指导方针，重点抓好公共基础课、专业基础课和专业主干课教材的建设，特别要注意选择一部分原来基础较好的优秀教材进行修订使其逐步形成精品教材；同时还要扩大教材品种，实现教材系列配套，并处理好教材的统一性与多样化、基本教材与辅助教材、文字教材与软件教材的关系，在此基础上形成特色鲜明、一纲多本、优化配套的高职高专教育教材体系。

普通高等教育“十五”国家级规划教材（高职高专教育）适用于高等职业学校、高等专科学校、成人高校及本科院校举办的二级职业技术学院、继续教育学院和民办高校使用。

教育部高等教育司

2002年11月30日

前　言

软件工程诞生于20世纪70年代，现已发展成为计算机领域的一个专门学科。而在我国，从20世纪80年代末期以来，诸多的软件企业也逐步地由早期的软件作坊开发模式，进入到软件工程开发模式。可以说，最近十几年以来，软件工程作为一门有关软件开发的工程方法学，直接影响到了我们国家的软件产业的发展，它在软件开发中的指导意义与基础地位已经越来越多地得到了整个信息业界的高度重视。

软件工程在软件产业中的基础地位直接决定着它在教学上必将受到的重视程度。软件工程已经成为计算机及其相关专业的专业核心课程，许多有关软件技术的职业认证考试或等级考试也把软件工程列入考试大纲，而且所占考分比例逐年增大。

本书为国家十五规划教材，作者自感责任重大，不敢有半点疏忽，收集资料、潜心论证、精心创作，经过整整两年的精心创作与教学试验，终于有了现在这几十万字的收获。本书完稿之后，殷志云教授（留法博士）在百忙之中抽时间审阅了全部书稿，在此特表示感谢！

本书主要为高等院校应用型本科、专科和高职院校计算机相关专业“软件工程”课程教学所写，其以系统性、科学性、实用性为原则，并以结构严谨、布局合理、概念清晰、内容适度，能够更好地适应“软件工程”课程教学的需要为创作目标。

全书由正文和附录两个部分组成。正文章节包含概述、过程模型、项目分析与规划、需求分析、概要设计、面向对象分析与设计、界面设计、算法设计与编码、测试、维护等内容，它们基本上是按照软件生命周期的顺序进行组织。另外，两个附录则分别给出了软件开发中必然涉及的文档管理规范与文档格式这两个方面的说明。

本书是作者多年来从事软件工程教学、进行软件工程研究的结晶，教材所采用的以软件生命周期为线索实施教学的策略，能够使软件工程教学与软件工程实践得到更有成效的结合。一些正处于发展中的工程方法学，如组件技术、对象分布式技术、UML建模技术以及软件工程文化等，都在书中得到了体现。

本书也是对作者多年来进行软件工程实践的一次经验总结。教材中的诸多软件问题或工程实例，有许多就取材于作者的软件开发实践，并都按照教学的需要进行了模型简化。显然，这些源于实践的工程问题，对于提高软件工程教学的实践性与实用性，将具有很好的示范效应。为了方便读者自学，教材提供了比较丰富的实例与习题，并且每章都有小结。

本书以服务教学与广大读者为宗旨，但由于作者水平有限，书中难免有不足之处，恳请广大读者批评指正，以便本书再版时不断修正与完善。

曾强聪

2004年9月

目　录

第 1 章　软件工程概述

在计算机发展早期，软件被等同于程序，开发软件也就是编写程序。但是，随着软件应用的推广与规模的扩大，软件由程序发展成为了软件产品，软件项目成为了软件开发中的工程任务计量单位，软件作为工程化产品则被理解为程序、数据、文档等诸多要素的集合。应该说，软件工程即是为适应软件的产业化发展需要，而逐步发展起来的一门有关软件项目开发的工程方法学。

1.1　软　　件

1.1.1　软件特点

计算机系统由硬件、软件两部分组成。早期的计算机系统以硬件为主，软件费用只占系统总费用的 20%左右。但随着计算机应用范围的扩大，软件需求急剧上升，到 20 世纪 90 年代的时候，软件费用已经上升到了系统总费用的 80%以上。

早期的计算机系统主要用于一些科研机构的科学工程计算，软件任务单一，应用面狭窄，软件问题相对比较简单。因此早期软件开发主要着眼于程序编写，程序成为了早期软件的代名词。但随着软件系统功能的增多，软件应用范围与规模的扩大，软件变得越来越复杂了。因此，今天的软件系统已经具有了更多的内容，包含：计算机程序、用于设置程序正常工作的配置文件、描述软件构造的系统文档、指导用户正确使用软件的用户文档等。

软件是计算机系统中的逻辑成分，相对于硬件的有形的物理特性，软件则是抽象的，具有无形性。例如程序，它是操纵计算机工作的指令的集合，但它并不能以一种特殊的物理形态独立存在，而必须通过在它之外的物理存储介质，例如，磁盘、磁带等，才能得以保存；而当需要它工作时，则又需要将它调入到计算机的内部存储器上，并且通过计算机的中央处理器才能工作。

软件的无形特性使得我们只能从软件之外去观察、认识软件。可以把计算机系统与人的大脑进行比较，计算机硬件如同人脑的生理构造，而软件则如同基于人脑而产生的人的知识、思想等。

1.1.2　软件分类

计算机系统往往需要许多不同种类的软件协同工作，软件工程人员也可能会承担各种不同种类的软件开发、维护任务。因此，可以从多个不同的角度来划分软件的类别。

1．按软件功能划分

（1）系统软件：系统软件是计算机系统的必要成分，它跟计算机硬件紧密配合，以使计算机系统的各个部分协调、高效地工作。例如操作系统、数据库管理系统等。

（2）支撑软件：用于协助用户开发与维护软件系统的一些工具性软件。例如程序编译器、源程序编辑器、错误检测程序、程序资源库等。

（3）应用软件：用于为最终用户提供数据服务、事务管理或工程计算的软件。例如商业数据处理软件、工程设计制图软件、办公管理自动化软件、多媒体教学软件等。

2．按软件工作方式划分

（1）实时处理软件：能够及时进行数据采集、反馈和迅速处理数据的软件。其主要用于特殊设备的监控，例如，自动流水线监控程序、飞机自动导航程序。

（2）分时处理软件：能够把计算机 CPU 工作时间轮流分配给多项数据处理任务的软件。例如，多任务操作系统。

（3）交互式软件：能够实现人机对话的软件。这类软件往往通过一定的操作界面接收用户给出的信息，并由此进行数据操作；其在时间上没有严格的限定，但在操作上给予了用户很大的灵活性。例如，商业数据处理软件系统中的客户端程序。

（4）批处理软件：能够把一组作业按照作业输入顺序或作业优先级别进行排队处理，并以成批加工方式处理作业中数据。例如，汇总报表打印程序。

3．按软件规模划分

（1）微型软件：一个人几天内即可完成的源程序在 500 行语句以内的软件。这种软件主要供个人临时性使用，软件任务单一，操作简单。一般没有必要建立系统文档。

（2）小型软件：一个人半年之内即可完成的 2 000 行以内的程序。这种软件通常没有与其他软件的数据接口，主要用于用户企业内部专项任务，大多由最终用户自主开发，软件使用周期短，但软件运行过程中产生的数据则可能在今后系统扩充中有一定的价值。考虑到系统今后有扩充的可能性，软件创建过程中应该提供必要的系统文档支持。

（3）中型软件：由一个项目小组在一年时间内完成的 5 万行源程序以内的软件系统。中型软件在项目实施过程中有了软件人员之间、软件人员与用户之间的通信，软件开发过程中需要进行资源调配。因此，软件开发过程中已需要系统地采用软件工程方法，例如，项目计划、技术手册、用户手册等文档资源，以及技术审查制度等，都需要比较严格地建立起来。

（4）大型软件：由一个至几个项目小组在两年时间内完成的 5 万行源程序以上的软件系统。当有多个项目小组共同参入软件开发时，需要对参加软件开发的软件工程人员实施二级管理，一般将项目按功能子系统分配到各个项目小组，然后再在项目小组内将具体任务分配到个人。由于项目周期较长，在项目任务完成过程中，人员调整往往不可避免，并会出现对新手的培训和逐步熟悉工作环境的问题。对于这样大规模的软件，采用统一的标准，实行严格的审查是绝对必要的。由于软件的规模庞大以及问题的复杂性，往往在开发的过程中会出现一些事先难以预料的不测事件，对此需要有一定的思想与工作准备。

4．按软件服务对象划分

（1）通用软件：由软件开发机构开发出来的直接提供给市场的软件。例如通用财务软件、通用字处理软件，杀毒软件等。这类软件通常由软件开发机构自主进行市场调查与市场定位，并由此确定软件规格，大多通过一定的商业渠道进行软件销售。

（2）定制软件：受某个或少数几个特定客户的委托，由一个或多个软件开发机构在合同的约束下开发出来的软件。例如，某专门设备的控制系统、某特定企业的业务管理系统、某智能

大厦的监控与管理系统、某城市的交通监管系统。这类软件通常由用户进行软件描述，并以此为基本依据确定软件规格。作为软件开发机构，则必须按照用户要求进行开发。

1.1.3 软件发展历程

计算机系统总是离不开软件的，然而早期的硬件、软件是融于一体的，为了使得某台计算机设备能够完成某项工作，不得不给它专门配置程序。但是，随着计算机技术的快速发展和计算机应用领域的迅速拓宽，自20世纪60年代中期以来，软件需求迅速增长，软件数量急剧膨胀，于是，软件从硬件设备中分离了出来，不仅成为了独立的产品，并且逐渐发展成为了一个专门的产业领域。

观察软件的发展，可以发现软件生产有三个发展时代，即程序设计时代、程序系统时代和软件工程时代。

1．程序设计时代（20世纪50年代）

20世纪50年代是软件发展的早期时代，计算机主要应用于科研机构的科学工程计算，软件则是为某种特定型号的计算机设备而专门配置的程序。

这时的软件工作者是以个体手工的方式制作软件，他们使用机器语言、汇编语言作为工具直接面对机器编写程序代码。而这时的硬件设备不仅价格昂贵，而且存储容量小、运行速度慢、运行可靠性差。尽管程序要完成的任务在今天看来是简单的，但由于受计算机硬件设备条件的限制，程序设计者也就不得不通过编程技巧来追求程序运行效率。

这个时期的程序大多是自用，程序的编写者往往也就是使用者，软件还没有形成为产品。由于早期程序大多是为某个具体应用而专门编写的，程序任务单一，因此，对程序的设计也就仅仅体现在单个程序的算法上。早期程序还往往只能在某台专门的计算机上工作，很难将程序由一台设备移植到另一台设备。

2．程序系统时代（20世纪60年代）

20世纪60年代，计算机技术迅速发展。在硬件技术上，由于半导体材料、集成电路的出现，计算机设备不仅在运行速度、可靠性和存储容量上有了显著的改善，而且价格也大大降低了，这使得计算机得到了更加广泛的应用。例如，一些大型商业机构已经开始使用计算机进行商业数据处理。

在软件技术上，高级语言的诞生，显著提高了程序编写的效率，这使得一些更大规模的具有综合功能的软件被开发出来。操作系统也出现了，它有效地改善了应用软件的工作环境，并使得应用软件具有了可移植性。由于计算机的应用领域的扩大，软件需求不断增长，软件规模也越来越大；于是，“软件作坊”在这个时期出现了，伴随着“软件作坊”还产生出了具有一定通用性的软件产品。

“软件作坊”已在生产具有工业化特征的软件产品，并且这时的软件工作者已在使用系统的方法设计、制作软件，而不是孤立地对待每个程序。但是，“软件作坊”是一种比较疏散的组织机构，这使得软件开发还不能形成为工业流程。

这个时期的软件开发更多地依赖于个人创作。由于软件开发的主要内容仍然是程序编写，软件开发的核心问题仍是技术问题；于是，用户的意图被忽视了，除了程序之外的其他文档、技术标准、软件在今后运行过程中的维护等问题，也往往被忽视了。

软件已经开始成为产品，但软件的生产方式则是跟产品并不适宜的作坊创作方式。于是，随着软件规模的不断扩大，软件危机现象在这个时期最终爆发出来。

3. 软件工程时代（20世纪70年代起）

1968年在联邦德国召开的计算机国际会议上，专门针对软件危机问题进行了讨论，在这次会议上正式提出并使用了“软件工程”术语。于是，软件工程作为一门新兴的工程学科诞生了。“软件工程”如一线霞光，使软件发展步入到了一个新的时代。

在软件开发上，自20世纪70年代以来的30年里，结构化的工程方法获得了广泛应用，并已成为了一种成熟的软件工程方法学；而自20世纪90年代起，基于面向对象的工程方法，也已被应用于软件开发之中。应该说，采用工程的原理、技术和方法实施软件产品开发，以适应软件产业化发展的需要，成为了这个时期诸多软件企业的追求目标。

这是一个软件产业高速发展的时期，以软件为特征的“智能”产品不断涌现。尤其是网络通讯技术、数据库技术与多媒体技术的结合，彻底改变了软件系统的体系结构，它使得计算机的潜能获得了更大程度的释放。可以说，以计算机软件为核心的信息技术的高速发展，已经使得人们的生活方式与生活节奏发生了根本性的变化。

“软件工程”自产生以来，人们就寄希望于它去冲破“软件危机”这朵乌云。但是，软件危机现象并没有得到彻底排除，特别是，一些老的危机问题可能解决了，但接着又出现了许多新的危机问题，于是不得不去寻找一些更新的工程方法。应该说，正是危机问题的不断出现，推动着软件工程方法学的快速发展。

1.2 软 件 危 机

1.2.1 软件危机现象

软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题。例如，软件的开发成本、进度，软件质量，等等。这些问题绝不仅仅是不能正常运行的软件才具有的，实际上，几乎所有软件都不同程度地存在这些问题。

具体说来，软件危机主要有以下一些方面的典型表现。

1. 软件开发成本、进度的估计很不准确

软件开发机构制定的项目计划跟实际情况有很大差距，使得开发经费一再突破。由于对工作量和开发难度估计不足，进度计划无法按时完成，开发时间一再拖延，这种现象严重降低了软件开发机构的信誉。

2. 软件产品常常与用户的要求不一致

在开发过程中，软件开发人员和用户之间缺乏信息交流。开发人员常常是在对用户要求只有模糊了解的情况下就仓促上阵，匆忙着手编写程序。由于这些问题的存在，导致开发出来的软件不能满足用户的实际应用需要。

3. 软件产品质量可靠性差

软件开发过程中，没有建立起确实有效的质量保证体系。一些软件项目为了赶进度或降低软件开发成本，甚至不惜降低软件质量标准、偷工减料。

4．软件文档不完整、不一致

计算机软件不仅仅是程序，在软件开发过程中还应该产生出一系列的文档资料。实际上，软件开发非常依赖这些文档资料。在软件开发过程中，软件开发机构的管理人员需要使用这些文档资料来管理软件项目；技术人员则需要利用文档资料进行信息交流；用户也需要通过这些文档资料来认识软件，对软件进行验收，熟悉软件的安装、操作等。但是，由于软件项目管理工作的欠缺，软件文档往往不完整，对软件的描述经常不一致，很难通过文档去跟踪软件开发过程中软件产品规格的变更。

5．软件产品可维护性差

软件中的错误非常难改正，软件很难适应新的硬件环境，很难根据用户的需要在原有软件中增加一些新的功能。这样的软件是不便于重用的，以前开发的软件，一旦过时就不得不完全丢弃。

6．软件生产率低

软件生产率跟不上硬件的发展速度，不能适应计算机应用的迅速普及，以致现代计算机硬件提供的巨大潜力不能被充分利用。

1.2.2 产生软件危机的原因

软件危机现象最初出现在软件发展的第二个阶段——程序系统时代。自那时起，软件工作者就一直在探寻导致软件危机的原因，并期望能通过对软件危机原因的分析，从而找到一种行之有效地克服软件危机的方法、策略。

通过对一系列危机现象的研究，人们总结发现，产生软件危机的原因主要体现在以下几个方面。

1．软件的不可见特性

软件不同于硬件，它是计算机系统中的逻辑部件，缺乏“可见性”。硬件错误往往可以通过它的物理现象直接反映出来，例如，出现不正常的发热、噪音现象等；但软件错误没有这些直观表现，例如，软件中存在的程序行错误，就必须等到这行程序执行时才有可能被发现。因此，软件错误比起硬件错误来更难发现。软件的不可见特性也使得对软件项目的量化管理更难实施，对软件质量的量化评价更难操作。

2．软件系统规模庞大

软件成为产品以后已不同于早期程序，随着它的功能的增多，其规模、复杂程度越来越大。例如，1968 年美国航空公司订票系统达到 30 万条指令；IBM360OS 第 16 版达到 100 万条指令；1973 年美国阿波罗计划达到 1 000 万条指令。这些庞大规模的软件系统，其复杂程度已超过了人所能接受的程度；但是，面对不断复杂的软件系统，其开发手段却仍然需要依靠开发人员的个人创造与手工操作。

3．软件生产工程化管理程度低

软件生产的工程化管理是软件作为产品所必须的，这意味着软件也需要像硬件一样，在软件分析、设计完成之后，才能考虑软件的实现。应该说，工程化管理能够降低解决问题的代价。但是，许多软件的开发则往往是在分析、设计没有完成的情况下，就已经进入编码实现阶段。由于前期准备工作不充分，致使软件项目管理纷乱，严重影响软件项目成本、开发进度和软件

质量。

4. 对用户需求关心程度不够

软件开发机构不熟悉用户业务领域。软件技术人员所关注的仅仅是计算机技术，它们不太愿意和用户沟通，轻视对用户的需求调查，也缺乏有效的用户调查策略、手段。由于这些问题的存在，使得用户的需求意愿不能充分反映，或被错误理解。

实际上，软件是为用户开发的，只有用户才能真正了解他们自己的需要。由于没有对用户做大量深入细致的调查研究，以致软件需求规格定义不准确，并最终使得完成后的软件不能适应用户的应用需要。

5. 对软件维护重视程度不够

软件开发缺乏统一的规范。在软件产品开发过程中，开发者很少考虑到这个软件今后还需要提供维护。但是，软件的使用周期漫长，软件错误具有隐蔽性，许多年之后软件仍可能需要改错。另外，软件的工作环境也可能会在几年后发生改变；用户也可能在软件运行几年以后，要求对它增加新的功能。这些都是属于软件维护问题。实际上，软件的可维护性是衡量软件质量的一项重要指标，软件可维护性程度高，软件就便于修正、改版和升级，由此可以使软件具有更长的使用寿命。

6. 软件开发工具自动化程度低

尽管软件开发工具比30年前已经有了很大的进步，但直到今天，软件开发仍然离不开工程人员的个人创造与手工操作，软件生产仍不可能像硬件设备的生产那样，达到高度的自动化。

1.3 软 件 工 程

1.3.1 软件工程概念

软件工程是一门关于软件开发与维护的工程学科，它涉及软件生产的各个方面，包括：工程过程、工程原则、技术方法与工具，以及工程项目管理等，能够为经济、高效地开发高质量的软件产品提供最有效的支持。

实际上，我们可以从多个不同的角度来认识软件工程。

1983年国际权威机构IEEE给软件工程下的定义是：**软件工程是开发、运行、维护和修复软件的系统方法**。其中的“软件”被定义为：计算机程序、方法、规则、相关的文档资料，以及计算机程序运行时所需要的数据。

Fairly给出的定义是：软件工程学是为了在成本限额以内按时完成开发和修改软件产品时，所需要的系统生产和维护技术及管理学科。

FritZ Baner则给出了下述定义：软件工程是为了经济地获得可靠的且能在实际机器上有效运行的软件，而建立和使用的完善的工程化原则。

1.3.2 软件工程技术

软件工程技术是指软件工程所具有的技术要素。作为软件开发与维护的工程方法学，软件

工程具有三个方面的技术要素，即软件工程方法、软件工具和软件工程过程。

1．软件工程方法

软件工程方法是指完成软件开发与维护任务时，应该“如何做”的技术方法。它所涉及的任务贯穿于软件开发、维护的整个过程之中，包括：软件需求分析、软件结构设计、程序算法设计等诸多任务；而其方法则体现在使用图形或某种特殊语言的方式来表现这些任务中需要建立的软件系统模型，如：数据流模型、软件结构模型、对象模型、组件模型等。主要的软件工程方法有：结构化方法、JSD 方法和面向对象方法。

（1）结构化方法

结构化方法是传统的基于软件生命周期的软件工程方法，自 20 世纪 70 年代产生以来，获得了极有成效的软件项目应用。结构化方法是以软件功能为目标来进行软件构建的，包括：结构化分析、结构化设计、结构化实现和结构化维护等内容，能够很好地适应结构化编程工具，例如：C、Pascal 语言等。它主要使用数据流模型来描述软件的数据加工过程，并可以通过数据流模型，由对软件的分析顺利过渡到对软件的结构设计。

（2）JSD 方法

JSD 方法主要用在软件设计上，1983 年由法国科学家 Jackson 提出。它以软件中的数据结构为基本依据来进行软件结构与程序算法设计，是对结构化软件设计方法的有效补充。在以数据处理为主要内容的信息系统开发中，JSD 方法具有比较突出的设计建模优势。

（3）面向对象方法

面向对象方法是以软件问题域中的对象为基本依据来构造软件系统模型的，包括：面向对象分析、面向对象设计、面向对象实现和面向对象维护等内容。确定问题域中的对象成分及其关系，建立软件系统对象模型，是面向对象分析与设计过程中的核心内容。自 20 世纪 80 年代以来，人们提出了许多有关面向对象的方法，其中，由 Booch、Rumbaugh、Jacobson 等人提出的一系列面向对象方法成为了主流方法，并被结合为统一建模语言（UML），成为了面向对象方法中的公认标准。面向对象方法能够最有效地适应面向对象编程工具，例如：C++、Java 等，并特别适用于面向用户的交互式系统的开发。

2．软件工具

软件工具是为了方便软件工程方法的运用而提供的具有自动化特征的软件支撑环境。

软件工具通常也称为 CASE，它是计算机辅助软件工程（Computer-Aided Software Engineering）的英文缩写。CASE 工具覆盖面很广，包括：分析建模、设计建模、源代码编辑生成、软件测试等。表 1-1 所列是一些常用的 CASE 工具类别。

表 1-1 常用的 CASE 工具类别

工 具 类 型	举 例
项目管理工具	项目规划编辑器、用户需求跟踪器、软件版本管理器
软件分析工具	数据字典管理器、分析建模编辑器
软件设计工具	用户界面设计器、软件结构设计器、代码框架生成器
程序处理工具	程序编辑器、程序编译器、程序解释器、程序分析器
软件测试工具	测试数据生成器、源程序调试器

用来支持软件分析、设计的CASE工具，如数据字典管理器、分析建模图形编辑器、软件结构设计器等，被称为高端CASE工具；而用来支持软件实现和测试的CASE工具，如程序编辑器、程序分析器、源程序调试器等，则被称为低端CASE工具。

可以把诸多独立的软件工具集成起来，形成为一体化的CASE工作平台。这样的工作平台能够为软件开发提供更强有力的支持，平台中数据资源共享，界面风格、操作方式统一，诸多通用操作能够作为模块被许多工具调用，一种工具产生的信息也可以被其他的工具引用。

CASE工作平台最初应用于低端工具的集成上，例如，早期DOS环境下的Turbo C，即是一个集程序管理、编辑、调试、编译于一体的C语言程序创建工具。但在今天，CASE工作平台已被应用于软件开发的各种活动上，涉及软件的分析设计、安装部署、项目管理、版本控制等多个方面。

分析与设计工作平台是软件工程方法中的核心工作平台，由许多工具集成，如图1-1所示。分析与设计工作平台主要用于分析、设计阶段的系统建模，许多分析与设计工作平台既可用于结构化方法的系统建模，例如，创建数据流图、软件结构图等；也可用于面向对象方法的系统建模，例如，创建对象图、状态图等。

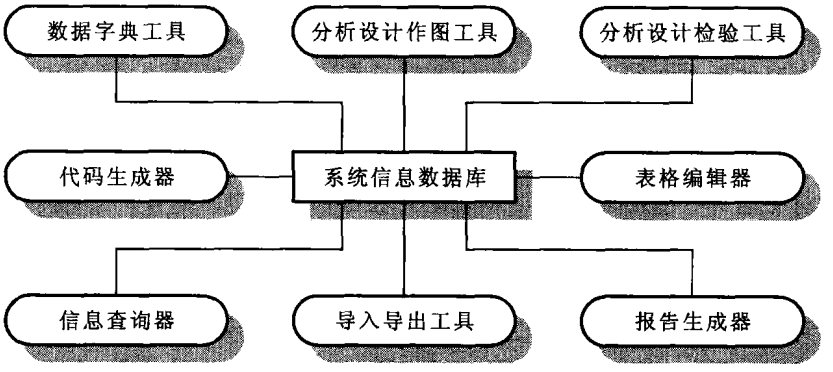


图1-1 分析与设计工作平台

3. 软件工程过程

尽管有软件工具与软件工程方法，但这并不能使软件产品生产完全自动化，它们还需要有合适的软件工程过程才能真正发挥作用。

软件工程过程是指为了开发软件产品，开发机构在软件工具的支持下，按照一定的软件工程方法所进行的一系列软件工程活动。实际上，这一系列的活动也就是软件开发中开发机构需要制定的工作步骤，它应该是科学的、合理的，否则将影响软件开发成本、进度与产品质量。因此，软件工程过程也就涉及到了软件产品开发中有哪些工作步骤，各个工作步骤分别具有什么样的工作特征，以及各个工作步骤分别需要产生一些什么结果等方面的问题。

软件工程过程并不是完全固定的，每个软件开发机构都可以专门制定更加适合自身特点的软件工程过程。实际上，软件产品不同，软件工程过程也可能会有所不同。但是，以下四项基本活动，则是绝大多数软件过程所必须的。