CENTURY 21

高等学校教材

Textbook for Higher Education

计算机软件工程与网络技术

孟东升 韩建宁 主编

西北工业大学出版社

计算机软件工程与网络技术

孟东升　韩建宁　主编

西北工业大学出版社

图书在版编目(CIP)数据

计算机软件工程与网络技术/孟东升,韩建宁编.—西安:西北工业大学出版社,2001.3
ISBN 7-5612-1007-8

Ⅰ.计... Ⅱ.①孟... ②韩... Ⅲ.①软件工程—高等学校-教材②计算机网络-高等学校-教材 Ⅳ.TP3

中国版本图书馆 CIP 数据核字(2001)第 027963 号

出版发行:西北工业大学出版社
通信地址:西安市友谊西路 127 号　**邮编**:710072　**电话**:029—8493844
网　　址:http://www.nwpup.com
印 刷 者:西安建筑科技大学印刷厂
开　　本:787mm×1 092mm　1/16
印　　张:10.25
字　　数:242 千字
版　　次:2001 年 9 月第 1 版　2001 年 9 月第 1 次印刷
书　　号:ISBN 7-5612-1007-8
印　　数:1～2 500
定　　价:14.00 元

前　言

目前国内外有关软件工程及计算机网络的教科书基本上是针对计算机专业的学生论述的。这些教材对95%以上的非计算机专业学生来讲，无论从学习时间、难度、深度等方面都显得要求过高，而适用于非计算机专业学生的软件开发和网络应用方面的教材却较少。随着教学改革的进一步深入和社会对毕业生需求质量的提高，越来越多的非计算机专业学生迫切要求开设软件工程及计算机网络等课程。为了适应这一要求，我们编写了《计算机软件工程及网络技术》一书。本书有机地将软件工程与计算机网络中最基本的内容结合在一起，目的是提高非计算机专业学生掌握软件开发方法和计算机网络的基本知识，以及培养应用这些方法和知识的能力。

本书是根据非计算机专业的特点和规律组织教学，改变了过去照搬计算机专业的教学方法，从而使理工、农医、经济管理等各非计算机专业的学生都能得到计算机的普及教育，培养和提高他们的计算机知识和应用能力

本书依据编者在计算机应用专业的讲义基础上，结合非计算机专业的教学要求编写的，并力求作到内容新颖、实用性强、概念清晰、通俗易懂。

总之在有限的计算机教学时间内，提高非计算机专业学生的软件开发水平以及运用计算机网络的能力与实践，是我们编写本教材的宗旨。

本书分为两大部分。第一部分软件工程共10章，由韩建宁主编，第二部分计算机网络共5章，由孟东升主编。中国计算机学会编辑出版委员会的有关专家对本书进行了评审，并提出了宝贵意见，在此表示衷心感谢。由于时间紧迫，加之水平与经验有限，书中难免有不当与错误之处，望广大读者批评指正。

编　者

1999年10月

目　录

第一部分　软件工程

试读结束：需要全本请在线购买：　www.ertongbook.com

第二部分 计算机网络

第一部分

软件工程

第1章　软件危机与软件工程

§1.1　什么叫软件

对于软件，没有严格的定义。有人说软件就是程序或是程序加以说明；还有人说软件是作为商品的程序。这些说法都有一定道理，反映了不同发展阶段，随着软件生产方式的不同，人们的认识差异，而且仍然在逐步发展变化之中。一般从广义上来讲可以把下列三者总称为软件：

(1) 程序：用某种程序设计语言表达计算机所处理的一系列步骤。

(2) 文档：软件开发过程中的分析、设计、实现、维护等文档资料。

(3) 使用说明书：用户手册、操作手册、维护手册等。

软件和程序常常被误认为同一含义的东西。其实，两者是有区别的。软件实际上包括计算机程序、方法、规则、相关的文档以及运行程序时所必须的数据。其主要功能就是利用计算机所提供的逻辑功能来合理地组织计算机工作，以便为人们提供一个便于掌握、操作简便的工作环境，例如 Windows 运行环境、Borland C++以及操作系统、编辑系统等各种开发环境都可称为软件。而程序则是我们利用这些环境编写的一些简单的例子和作业。当然，程序和软件的划分也是相对的，一般而言，程序的规模较小，通用性也不及软件。在大型软件系统的开发和维护过程中，文档及使用说明是软件"质"的部分，程序则是文档代码化的表现形式。

§1.2　软件危机

众所周知，现代计算机的应用日益普及和深化，与硬件相比，社会对软件的需求量正在以惊人的速度急剧膨胀，而且规模也往往十分庞大。其中包含数百万行代码，耗资几十亿美元，花费几千人年劳动才开发出来的软件产品，已是屡见不鲜了。例如，美国第四代宇宙飞船的软件包含 4 000 万行目标代码，阿波罗登月计划的软件则长达 1 000 万行代码。

由于软件规模的增大，使得开发软件的复杂度大大地增加。其结果是，大型软件的开发费用常常超出预算，不能按期完成。更为严重的是，软件的可靠性往往随着规模的增大而下降，质量也越来越难以保障。

现代微电子学技术的进步，使得计算机硬件性能价格比平均每十年提高二个数量级，而且质量也大大提高了；与此同时，计算机软件成本却逐年上升，质量无可靠的保证，软件开发生产率也远远跟不上普及计算机应用的要求。软件已成为限制计算机系统发展的关键因素

(见图 1.1)。

回顾计算机应用的早期,软件通常是规模较小的程序,软件投资比重很小。随着计算机技术的进步以及应用领域的不断扩大,对于软件的投资越来越大,所占比重也越来越高。B. Boehm 在 1973 年发表的一篇论文中曾预计到 1985 年,美国空军的软件成本费用将上升到计算机总费用的 90%,即在每 100 元用于计算机的总投资中,软件将占 90 元。

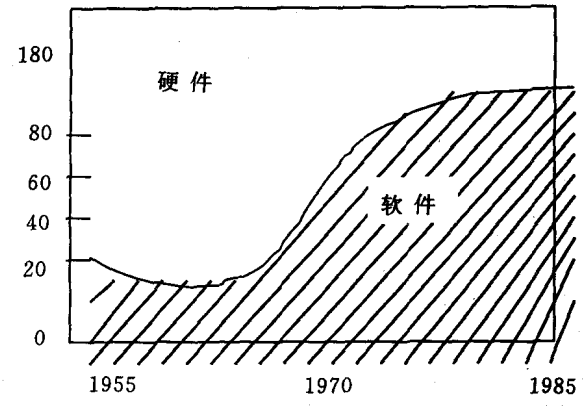

图 1.1

庞大的软件费用,加上软件质量的下降,对计算机应用的继续扩大构成了巨大的威胁。面对这种严峻的形势,软件界的有志之士发出了软件危机的警告。1968 年北大西洋公约组织的计算机科学家们在联邦德国召开国际会议讨论软件危机问题,正式提出并使用了"软件工程"这个名词,于是一门新兴的工程学科就此诞生了。

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。概括地说,软件危机包括两个方面的问题:即如何开发软件,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是:

- 高成本,很难在预算内完成
- 质量差,可靠性难以保证
- 开发过程难以控制,周期很长
- 维护困难,用户意见大
- 极不容易管理

产生软件危机的原因很多,但总的说来有以下两个方面:

(1)软件自身的特点。

软件不同硬件,它是计算机系统中的逻辑部件而不是物理部件,物理部件允许有误差,而逻辑部件则不然。软件开发过程的进展情况较难衡量,质量也较难评价,因此,软件管理和开发过程相当困难。另外,软件与一般程序不同,它的一个显著特点是规模庞大,其开发和维护的复杂度大大增加。

(2)方法不正确软件是逻辑产品,而不是物理产品,所以决定软件质量好坏的是人的智能的发挥。人们在长期使用计算机系统的实践过程中,积累和总结了许多成功的经验和启发性知识。如果坚持不懈地使用经过实践考验证明是正确的方法,许多错误是完全可以避免的。因此,尽快掌握正确的软件开发思想方法,是解决软件危机的主要途径。

§1.3 软件工程

软件工程研究的是:如何应用一些科学理论和工程上的技术来指导大型软件的开发。作为一门学科,它是指导计算机软件开发和维护的工程学科。采用工程的概念、原理、技术和方

试读结束:需要全本请在线购买: www.ertongbook.com

法来开发与维护软件，把各种正确的管理技术和最好的技术结合起来，就构成了软件工程的范畴。

习题

1. 什么叫软件？试述软件与程序的区别。
2. 什么是软件危机？为什么会产生软件危机？
3. 什么是软件工程？

第 2 章　软件生存周期

一切工业产品都有自己的生存周期，在机械工程中，一台机器的生命期（即从开始研制到机器最终被废弃不再使用为止）要经过分析、设计、制造、测试、运行等几个阶段。软件（产品）也不例外。一个软件从开始计划研制起，经过需求分析、设计、编程、测试和运行等五个阶段，每个阶段都有明确的任务，并需产生一定的文档资料交付给下一阶段，下阶段在上阶段交付的文档的基础上继续开展工作，直至软件废弃不用为止，称为软件的生存周期。一般说来，软件生存周期包括定义、开发和维护三个时期，每一个时期又进一步划分成若干个小阶段。

生存周期是软件工程学的一个重要概念。把整个生存周期划分成较小的阶段，每个阶段的任务相对独立，而且比较简单，便于不同人员分工协作，从而降低了整个软件开发工程的困难程度；便于从技术和管理两个角度对各阶段的工作进行严格的审查，合格之后才开始下一阶段的工作，使得软件开发的全过程以一种有条不紊的方式进行，从而保证了软件质量，特别是提高了软件的可维护性。

目前，生存周期的阶段划分方法有许多种，但总的来说都遵循一条基本原则就是使各阶段的任务彼此间尽可能相对独立。大体上讲，生存周期模型（SW Life Cycle Model）可以归为两类：即传统的生命结构模型或称瀑布模型和原型化模型（Prototype Model）。本章将简单介绍这两类不同的生命周期模型，使读者能对软件生命周期的全过程有一个总体的概念。

§2.1　生存周期模型

生存周期模型（即瀑布模型）可分为三个时期：定义、开发与维护。每一个时期又划分为若干个阶段。典型的瀑布模型如图 2.1 所示。在该模型中，各个阶段的工作相对独立，顺序展开，由上而下似瀑布一般。

目前，划分软件生存周期的方法也不尽一致，有许多种。一般来讲，划分方法受到软件规模、种类、开发方式、开发环境以及开发时使用的方法论等多种因素的影响。但无论哪种，都必须遵循的一条原则就是使各阶段的任务彼此尽可能独立，同一阶段的工作性质尽可能相同，从而降低每个阶段任务的复杂性，简化不同阶段间的联系，经过有限的步骤，把用户的问题从抽象的逻辑概念逐步转化为具体的物理实现。

下面以图 2.1 为例，扼要介绍生存周期各阶段的主要任务和结束标准。

一、定义时期

软件定义时期的任务是确定软件开发工程所必须完成的总目标，确定开发该系统的可行

性，采用的开发策略及确定系统必须完成的功能，并进行成本估计，制定软件开发进度表。这一阶段通常由系统分析员来完成。软件定义时期可进一步划分为以下三个阶段。

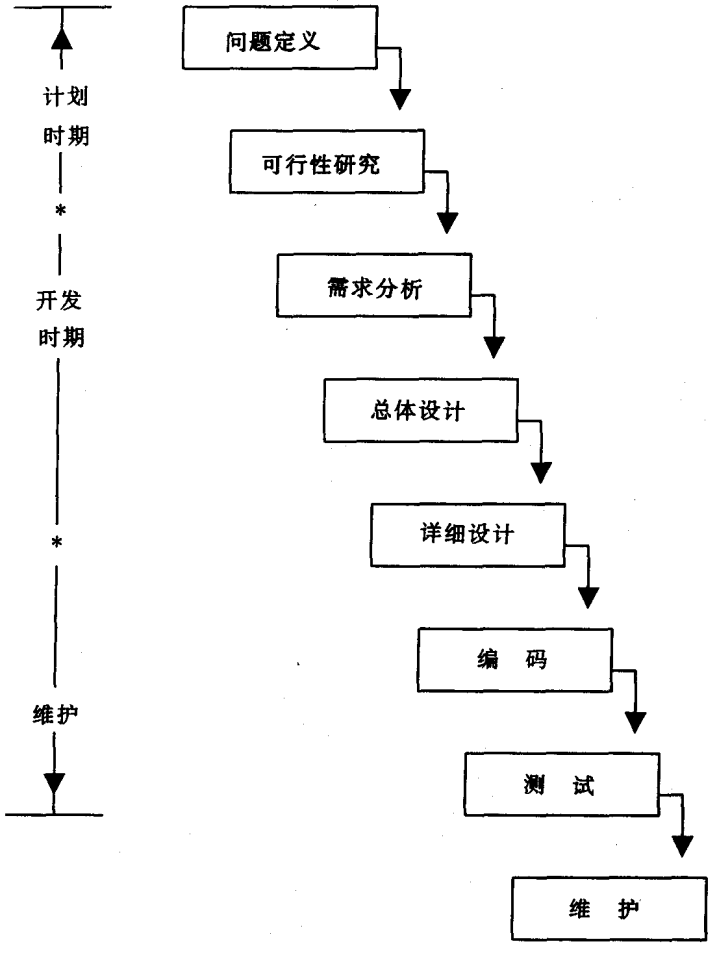

图 2.1　典型的瀑布模型

1. 问题定义

该阶段为整个生存周期的第一步，主要任务是弄清楚“用户要解决的问题是什么”，通过问题定义阶段的工作，系统分析员根据对问题的理解，提出关于问题的性质、系统目标和规模的书面报告，并与用户一起审查讨论，最后提出一份双方都满意的文档。此阶段一般需要一天甚至更少的时间。

2. 可行性研究

这一阶段主要是对上一阶段所确定的问题寻求一种或多种行得通的解决方法。为此，系统分析员需要进行一次大大压缩和简化了的系统分析和设计。可行性研究阶段应该导出系统的高层逻辑模型(通常用数据流程图)，并描述系统物理概貌(通常用系统流程图)。另外，对建议的系统进行仔细的成本/效益分析也是这个阶段的主要任务之一。一般来说，只有投资可能取得重大效益的那些系统项目才值得进行下去；否则，应立即中止，以免造成更大的浪费。

3. 需求分析

这个阶段的任务是提出未来"系统必须做什么",即系统必须具备的功能。这一阶段要求系统开发人员与用户密切配合,充分交流信息,以得出经用户确认的系统逻辑模型。通常用数据流图、数据字典和简要的算法描述来表示系统的逻辑模型。

二、总体设计

这个阶段的主要任务是确定如何解决需求分析阶段所提出的具体问题。这一阶段通常应首先考虑几种可能的解决方案。系统分析员应使用系统流程图或其它可能的工具描述几种可能的解决方案,并对每种方案进行成本/效益分析,最后推荐一个最佳实现方案,提请用户讨论确定。总体设计阶段的第二项任务是设计出软件的结构,即确定系统程序应该由哪些模块组成以及这些模块间的关系。通常采用层次图或结构图来描述软件的结构。

三、详细设计

这一阶段的主要任务是"如何具体地实现系统的所有功能"。通过这一阶段的工作,系统分析员应该导出系统模块内部结构,详细说明实现这一模块的算法和数据结构,以便程序员可以根据这一阶段的工作写出实际的程序代码,进入编码阶段。详细设计阶段通常采用 HIPO 图(层次图加输入/处理/输出图)或 PDL 语言(过程设计语言)来描述详细设计的结果。

四、编码和单元测试

这个阶段的主要任务就是要求程序员写出正确的、易理解、易维护的程序模块。为此,程序员应能根据系统的特点和现实环境,选取一种恰当的语言,把模块的过程性描述翻译成源程序,并对编好每个的模块进行单元测试(有必要的话,须将相关模块一同测试)。通常该阶段工作相对要简单得多,一般由有一定编程经验的程序员担任即可。

五、综合测试

这个阶段的任务是通过各种类型的测试,使所开发软件系统达到预定的功能及性能要求。按不同的层次,可分为测试、验收测试等。测试是保证软件质量的重要手段。通常大型软件的测试常由另一独立的部门和人员进行。最后,测试阶段应该以文档的形式将测试计划、详细测试方案、测试用例与实际测试结果保存下来。

六、软件维护

这个阶段是软件生存周期的最后一个时期,通常不再进一步细化。其主要任务是通过各种维护技术使系统持久地满足用户的需要。对于大型软件,维护是不可避免的,维护的工作量在软件的整个生存期内占有非常大的比重。通常有四类维护活动:改正性维护、适应性维护、完善性维护及预防性维护。每一项维护活动都必须经过提出维护报告、分析维护要求、提出维护方案、审批维护方案、确定维护计划、修改软件设计、修改程序、复查验收等一系列步骤,并应以正式的文档资料保存。

表 2.1　结构分析设计过程表

阶　段	任　　务	结束标准
问题定义	寻求可行的解	系统的高层逻辑模型，数据流图，成本/效益分析
需求分析	确定系统必须完成的功能	系统的逻辑模型： 数据流图 数据字典 算法描述
总体设计	总体上考虑如何解决系统的所有功能要求	可能的解法： 系统流程图 成本/效益分析 推荐的系统结构 层次图或结构图
详细设计	具体地实现系统中的每一个模块	编码规则说明：HIPO 图或 PDL
编码和单元测试	编写出正确的程序模块	源程序清单，单元测试方案和结果
综合测试	符合要求的软件	测试方案和结果
维　　护	持久地满足用户要求的软件	完整准确的维护记录

以上简要地介绍了软件生存周期的每一阶段，表 2.1 概括了软件生存周期各个阶段的名称、主要任务和结束标准。

§2.2　生存周期模型软件开发的特点

以上按结构化(即瀑布型)生存周期各个阶段的先后顺序，简要介绍了其开发的全过程。目前信息系统的开发主要采用结构化系统分析(SSA—Structure System Analysis)和结构化系统设计(SSD—Structure System Design)方法。该方法的优点是有严格的一套开发规范，各开发阶段都有完整的文档记录，国内外都有许多成功开发的例证。下面介绍其主要的特点。

1. 顺序观点

结构化生存周期模型强调将系统开发看做一工程项目，按部就班，有计划、有步骤地进行工作，前一阶段是下一阶段工作的基础，下一阶段是前一阶段的补充和细化。因此，各个阶段的工作必须确保尽可能的全面、准确，以免错误蔓延到下一阶段，甚至整个软件的开发过程。它强调构造系统逻辑模型及文档资料的重要。软件开发人员必须严格按照结构化模型的步骤顺序进行，对用户要求没有正确的认识就急于匆忙编写程序的错误做法应彻底抛弃。按照顺序开发的观点，把逻辑设计与物理设计清楚地划分开来，先逻辑后物理，先抽象后具体实现，是结构化生存周期的基本思想。

2. 强调文档

开发较高质量的软件，是软件工程学追求的目标。为此，结构化生存周期方法强调每一个阶段工作的结束，应以是否有准确、完整的文档为标准的。另外，对各个阶段产生的文档资料都必须进行严格的技术审查和管理复查。尽可能早的发现错误，及时修改，这是保证软件质量，降低软件开发成本的重要措施。

总之，运用软件工程的思想指导软件的开发，就会大大提高软件的质量，提高软件开发生产率，这是进行软件开发的惟一成功途径。

§2.3 快速原型方法

快速原型方法(Rapid Prototyping)是迅速地根据软件系统的需求产生出软件系统的一个过程。该原型要表现出目标系统的主要功能，行为特性，但不一定符合其全部的实时要求。软件开发者可利用原型得到系统的更详尽的需求信息；用户也可通过原型得到未来系统的概貌。快速原型法的主要优点是尽可能获得一个更完整、更准确的需求与设计；而且软件开发者与用户面对的是一个活生生的系统，便于软件开发人员与用户交流，以获得较完整、准确的需求。另外，通过不断地改进原型，也可使之成为最终的目标系统。

目前，在诸多软件开发方法中，快速原型法已获得了广泛的应用，它比传统的结构化生存周期模型具有更大的灵活性与适应性，若运用得恰当，可大大改善软件开发的总效益。目前，国内外已有许多成功的例证是运用快速原型法完成的。

一、原型法与传统方法的比较

传统的生存周期瀑布模型，对需求的确定是按照严格定义或预先定义的方法来进行的，运用这一模型来开发软件，只有当分析员能够做到准确的分析时，才能够得到严格的或预期的正确结果。然而，由于用户一般来讲不熟悉计算机，系统分析员又对用户业务了解不够深入。系统分析员与用户之间的交流必然存在或多或少的障碍。因此，在计划时期确立的用户需求其不完整、不准确就显得尤为突出。国内外都有一些大型软件系统开发失败的例证，很大一部分原因就是所开发的系统未能完整、准确地反映用户的需求。

实际上，运用传统的瀑布模型进行软件开发，要求对需求的严格，预先定义也是相对的。严格说来，完全准确、完整的用户需求对于大型软件的开发来说，是难以获得的。应用原型化方法进行软件开发，正是为了弥补传统瀑布模型方法的不足。

原型化方法的主要思想是：首先建立一个能够看得见，能反映用户需求功能的系统原型。让用户参与系统的开发，确定未来系统中哪些功能符合要求，哪些功能还需补充改进。在原型基础上，反复修改，最终建立起完全符合用户需求的新系统。

应用原型化方法的主要优点是快速建立一个未来系统的原型，它可以使用户一开始就参与整个系统的开发，用户面对的是一个直观的系统而不是一个想像中的系统。用户可通过原型向系统开发人员迅速“反馈”回有用的信息，便于双方的交流与合作，这是原型法的主要优点。

二、快速原型法的实现途径

在建立原型法时常采用下述三种方法。

1. 研究探索原型

该原型是为研究探索建立的原型，它主要强调澄清新系统的需求及所要求的特征，也讨论其实现方案，包含未来系统的主要功能，以及系统的重要接口，不包括系统的细节，例如异常处理，对无效输入的反应等，对硬件速度等系统性能暂不考虑。

试读结束：需要全本请在线购买： www.ertongbook.com