fei ji suan ji zhuan yie xi lie jiao cai

非计算机专业系列教材

软件工程简明教程

原理 模型 方法 工具 规范 应用

RUAN JIAN GONG CHENG
JIAN MING JIAO CHENG

刘润彬 张 华 编著

大连理工大学出版社

非计算机专业系列教材

软件工程简明教程

——原理、模型、方法、工具、规范、应用

刘润彬　张　华　编著

大连理工大学出版社

辽新登字(16)号

内容简介

软件工程作为支撑软件产业的新学科，其发展正方兴未艾。本书详述软件工程的开发模型，软件开发，软件维护，软件管理的概念和方法。结合实例介绍了软件项目开发规范标准与文档编写纲要。书中每章配有练习和CAI软件，可供计算机水平考试和等级考试辅导培训使用。本书是作者在大连理工大学讲授软件工程课讲稿基础上编写的，可作为高等学校计算机有关专业的教材，非计算机专业复合型人材培训教材，专业技术人员的参考书。

软件工程简明教程

——原理、模型、方法、工具、规范、应用

Ruanjiangongcheng Jianmingjiaocheng

刘润彬　张　华　　编著

*　　*　　*

大连理工大学出版社出版发行

（大连市凌水河）

（邮政编码：116024）

大连海事大学印刷厂印刷

*　　*　　*

开本：787×1092　1/16　印张：15.75　　字数：360千字

1994年12月第1版　　1994年12月第1次印刷

印数 0001—5000册

*　　*　　*

责任编辑：王佳玉　　责任校对：张　林

封面设计：孙宝福

*　　*　　*

ISBN 7-5611-0971-7
TP·69　　定价：12.80元

前　言

此同称工程制图是工业行业的技术与语言一样，称软件工程是信息产业的技术与行业语言。工业中根据工业蓝图制作工业产品，软件产业也是根据软件蓝图来制作软件产品。软件工程不仅作为取得技术职务任职资格水平考试和等级考试的必修课，还作为行业从业人员必备的第二文化的训练课。

软件工程学是支撑软件产业的新学科，它主要是研究软件结构，软件设计与维护方法，软件工具与环境，软件工程标准和规范，软件开发技术与管理技术的相关理论。本书分软件工程概述，软件计划，软件需求分析，软件概要设计，软件详细设计，软件编码，软件测试，软件维护，软件工程标准与文档编制，软件工程环境与工具，十章和两个附录，详实地介绍了软件工程的原理、概念、方法与应用。书中尽可能地吸收国内外软件工程领域的最新成果，编写大量练习，列出软件设计开发规范，文档编写提纲，给出实例，配有 CAI 软件，供水平考试与等级考试人员，从事实际软件开发人员参考使用。本书可作为高等学校计算机有关专业的教材，非计算机专业培训教材。

全书由刘润彬主编，编写一至五、七至十章、附录 A，张华编写第六章并录入全书，张瑞、魏新、编写附录 B。本书是作者教学讲稿的扩充，错误和不足之处在所难免，敬请读者指正。

刘润彬　张华

1994 年 12 月

目　录

第一章　软件工程概述

软件工程方法和技术源于软件开发的实践，是在解决与挽救软件危机中，经历了程序设计、程序系统、软件工程三个阶段的发展而形成的。同时，随着软件生产的系列化、产品化、工程化和标准化的软件产业的兴起，软件工程又成为指导软件开发实践的理论根据与支撑软件产业的理论依托。

1.1　软件危机

1.1.1　软件的发展阶段

软件工程和硬件工程都可以看作为计算机系统工程的一部分。硬件工程随着微电子技术的发展而日臻成熟。尽管神经网络计算机还没有研制成功，而网络分布计算环境已经建成，微内核水平结构的系统芯片已设计成功，可以说硬件设计技术已经很好地确立起来。微型计算机每两年更新一代，计算机硬件性能/价格比每五年提高一个数量级。硬件设计的可靠性已可如期实现。

在计算机系统发展的初期(50～60年代)，硬件已经通用化，而软件却是个体化的。这时期的软件的编写者和使用者往往是同一个人，程序规模小，几乎没有什么系统的方法可遵循，软件设计常常是设计者头脑中进行的隐含过程，除了程序清单，根本没有设计文档资料。通常称这个阶段为程序设计阶段，其生产方式是个体手工方式，程序设计被视为个人的神秘技巧。

计算机系统发展的第二阶段(60年代中期～70年代中期)，称作程序系统阶段。这个时期引入多道程序设计，多用户系统的人机对话的概念，引入数据库管理系统，软件规模增大已达几十万条源程序语句。生产方式由个体发展为软件车间，软件产品在广泛销售。软件数量急剧增加。软件运行错，软件要改正，用户需求改变，软件也要修正，硬件或操作系统更新，软件也要更新，软件维护耗资比例在巨增，更严重的是程序个体化特征引起软件产品不可维护，从而导致软件危机。1968年北大西洋公约组织的计算机科学家召开国际会议，讨论软件危机问题，这次会议上正式使用软件工程这个名词，从此在克服软件危机中诞生了软件工程这个新兴学科。

计算机系统发展的第三个阶段(70年代起至现在)为软件工程阶段。这个阶段以软件产品化、系列化、工程化、标准化为标志的软件产业的兴起为特征，软件车间联成软件工厂和公司，甚至是跨国公司。打破了软件设计的个体化特征，有了软件工程化的设计原则、方法和标准可以遵循。在这个阶段，软件产业与软件工程学科在挽救软件危机中不断地发展。然而软件工程阶段的客观需求又在不断提出更复杂的问题，如分布式系统，微内核系统，智能机系统。再加上维护费用占用了数据处理的50%以上预算比例，软件标准的规范统一还不完善，促使软件危机仍然与日俱增，客观要求仍然必须加快软件工程的发展。

计算机系统发展正向着第四个阶段，社会信息化、软件产业化的阶段过渡，从现在的技术性的软件工程阶段过渡到企业计算机化，社会信息化的计算机系统工程阶段。

1.1.2 软件危机的挽救

60年代以来软件危机在如何开发软件以满足客观需求，如何维护日益增多的软件方面的个体事实如下：

(1)软件生产不能满足日益增长的客观需要，软件生产率远远低于硬件生产率和计算机应用的增长率，出现了供不应求的局面。

(2)软件开发成本和开发进度往往估计不准确。实际开发成本高出预计成本好几倍，甚至一个数量级。实际进度拖期几个月到几年。为了赶进度和节约成本所采取的权宜之计，往往又损害了软件的质量。

(3)软件开发人员对用户需求缺乏深入准确的了解，软件开发人员与用户间信息交流不充分，甚至闭门造车，软件产品不符合需求。

(4)软件产品质量差，软件质量保证技术(审查、复审、测试)没有贯穿于开发的全过程，软件产品信誉下降。

(5)软件可维护性差，程序中错误难以改正，软件适应性差，与硬件环境改变不能适应。用户根据需要增加扩充功能做不到，完善性维护也做不到。软件没能实现再用性，仍然重复开发先前已被开发过的类似功能的软件。

(6)软件文档资料不完整不合格，给软件交流、管理、维护造成困难，开发很难成功。

(7)软件价格昂贵，占计算机总成本的比例逐年上升，与硬件性能价格比迅速提高形成反差，软件成本随着客观规模数量扩大以及社会通货膨胀而持续上升，美国自1985年以来软件成本已占计算机系统总成本的90%以上。

分析上述危机的事实，可将软件危机的原因归结如下：

(1)软件本身的特点导致开发和维护难。比如软件是依赖于物理部件而存在的逻辑实体，本身就是抽象和复杂的，在没有写出程序代码上机运行之前，检查开发的正确性，评价质量好坏是很困难的，另外测试本身又会引入新错误，软件维护确实困难。

软件不仅因客观问题复杂而决定其复杂，规模十分庞大，比如穿梭号飞船有40000000行目标代码，需要4000人年；而且还因为这样复杂而庞大的软件又与社会人的组织与管理因素相关，所以其开发和维护必然困难。

(2)软件开发方法不正确，没有采用工程化方法，忽视需求，闭门造车，导致无法维护，推倒重来。开发成本与开发进度估计错，造成开发费超支，工程拖期。

(3)开发人员与管理人员只重视开发而轻视问题的定义和软件维护，造成软件不满足需求，维护纠错不彻底使软件开发半途而废。

(4)软件开发技术本身落后于硬件技术水平，落后于客观需求，系统分析员领域知识面窄，经验不足，开发成功的比例很低。

(5)软件管理技术差。质量管理无统一的软件规范标准，缺乏全面量化管理。配置管理中，文档一致性与完整性差，残缺不全，失掉管理的依据。项目管理，由于费用估计错，分配混乱，人员组织不合理，进度无序，无法保证软件技术方法的实现。

摆脱软件危机的途径就是软件工程生成的过程，主要有：

(1)采用工程化方法和工程途径来研制与维护软件。软件开发不是某种个体劳动的神秘

技巧，而是组织良好、管理严密、各类人员协同配合、共同完成的工程项目。要吸取和借鉴人类长期以来从事各种工程项目行之有效的原理、概念、技术、方法来定义、开发、维护软件。采用工程化方法和途径的核心，是将软件问题也分成若干阶段。从时间上把软件设计分步化简以便分工协作。从结构上化简为若干个简单逻辑模块，然后再进行物理实现。在定义、开发、维护的流程中，将软件设计与工程实践相结合，开发技术与管理技术相结合，达到低成本、高质量、满足需求的软件开发目的。这就形成了软件工程的技术，把软件作为工程产品来处理，按照计划、分析、设计、实现、测试、维护的周期来进行生产。

(2)采用先进的技术、方法与工具来开发与设计软件。技术是指生产管理技术，有：软件的需求定义和分析技术，软件设计与设计审查技术，软件设计表现技术，软件测试技术，软件可靠性的理论及评价方法，软件扩充与维护，软件成本估算等。方法是开发规范方法与开发模型。规范方法有：基本开发步骤及每一步的条件、目的、结果；软件评价标准；软件文件格式。开发模型有：生存周期(瀑布)模型，原型模型，总体数据库规划模型。工具是提高软件开发效率的开发软件的软件，将各个开发阶段使用的软件工具集合成一个整体，构成支持软件开发全过程的软件工程支撑环境。

(3)采用必要的组织管理措施。软件产品与其它产业的产品不同，它把思维、概念、算法、组织、流程、效率、质量多方面问题融为一体。本身是无形的，没有产量，没有体积，所以说软件工程项目管理又不同于一般工程项目管理。它是通过人员组织管理，项目计划管理，标准化量化管理，配置管理，保证软件产品按期高质量完成。

为了挽救软件危机，采取了上述技术措施与管理措施。软件工程正是在挽救软件危机中，从技术与管理两个方面，研究软件方法、软件工具、软件管理，研究如何更好地开发和维护软件，形成了一门新兴学科。软件工程成为指导计算机软件开发、维护、管理的理论根据。反过来软件产业的建立又带动和促进软件工程的发展。

1.2 软件开发模型

软件开发模型是许多学者在软件工程实践中总结出来的软件开发的方法步骤。到目前为止有：B. W. Boehm 提出的瀑布模型(生存周期模型)，如图 1.1 所示，M. A. Jackson 与 F. P. Brooks 提出的原型模型(样机模型)，如图 1.2 所示，J. Martin 提出的总体数据库规划模型，如图 1.3 所示。

1.2.1 瀑布模型

B. W. Boehm 提出的瀑布模型，如图 1.1 所示，即生存周期模型，是软件工程的基础模型。在软件工程化过程中对原始的传统方法的改进。核心思想是，从制作时间上按工序把问题化简，将功能实现与制作分开，便于分工协作。用结构化分析与设计方法，将物理实现与逻辑表示分开，从规模上将问题化简，将设计与实践结合，开发技术与管理技术结合，提出三个开发阶段，七个流水依赖的步骤流程。其优点，奠定了软件工程方法的基础，流水依赖，便于分工协作，推迟物理实现，易于修改文档，有复审质量保证。其缺点，与用户见面晚，纠错慢，难于克服系统分析员不懂专业领域知识，用户不懂计算机的困难，成功率低，一般 25%成功。适用于系统要求明确的小系统。其阶段流程如下：

定义阶段，确定问题结构。又分成计划与需求分析。计划通过问题定义，做可行性研究。确

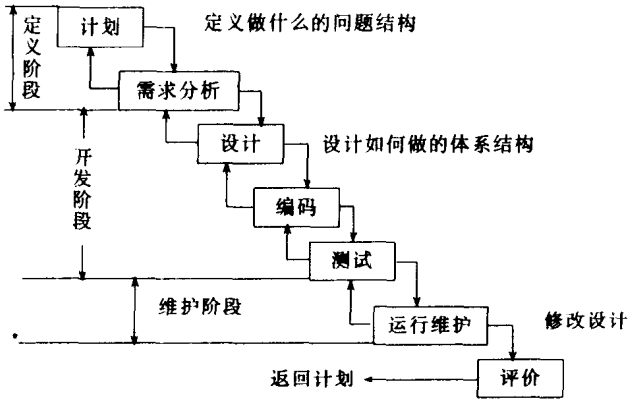

图1.1 瀑布模型

定开发软件总目标，给出系统功能、性能、可靠性、接口的设想，分析资源（软硬件资源、人力与开发时间）、成本、效益、开发进度，做可行性分析结论（技术、经济、社会法律），制订解决问题方案与开发实施计划，写出可行性分析报告，系统开发计划报告，提交审议。需求分析，详细定义做什么，确定技术目标。开发人员与用户共同决定可满足的要求，确切描述，写出需求规格说明书，初步用户使用手册，提交审议。

开发阶段，确定开发软件的体系结构。设计是解决如何将需求分析确定的技术指标转换成怎样做的体系结构。总体设计，完成划分模块功能与接口，并写概要设计说明书。详细设计，进行模块功能的算法与数据描述，写详细设计说明书。提交审议。编码，根据详细设计确定的算法描述，选择好一种程序设计语言，将算法描述转换成计算机可接收的程序，要求结构要好，清晰可读，易修改，与设计一致。测试，根据需求规格与软件设计说明、软件测试用例，验证是否符合需求，通过单元、组装、验收、系统、人工测试后，写出测试报告，交付使用。

运行维护阶段，交付使用后，进行改正性、适应性、完善性、预防性维护，保证软件运行合格。

复审评价，复审贯穿整个周期，最后鉴定评价软件的实用价值，评价全部的技术质量水平。防止了设计错误的放大效应。

1.2.2 原型模型

M. A. Jackson 和 F. P. Brooks 提出的软件开发原型模型（即样机模型），如图 1.2 所示。由于，不是所有用户对系统都能预先定义需求，往往只是模糊的设想，修改需求也经常发生。用户与开发者之间隔行如隔山，常常发生信息通讯障碍，需求无法准确表达。原型模型就是为了确定需求而提出的实际模型。打破传统的自项向下结构化开发模型方法，在计划和需求分析后，即把系统主要功能接口做为设计依据，快速开发出软件样机，及时征求用户意见，正确确定系统需求，然后再进一步准确地进行系统设计与实现。主要优点，与用户见面快，开发成功率高，适合于需求不确定的大系统。缺点是周期长，往往开发成本高。

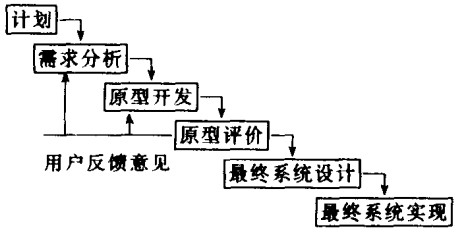

图 1.2 原型模型

原型模型改进型。由于验证需求的原型用后丢掉，造成开发成本高。故改进原型有如下几种。用于验证设计方案的原型，用后可保留，做为继续设计的部分，这样开发成本反而不会高。演进出目标系统原型或智能原型，不仅不会增加成本，反而使开发成本降低。这种模型是用知识与设计原理与高层原型规范交互翻译成低层规范，不断调节，自动翻译成源程序。

1.2.3 总体数据库规划模型

总体数据库规划模型(如图 1.3 所示)是 James Martin，分析了瀑布模型与原型模型的共同局限而提出来的。瀑布与原型的共同局限是从现行系统数据流程出发进行开发的，一旦系统本身陈旧，开发系统也将陈旧，无法摆脱被废弃的命运。总体数据库规划模型的思想是：确定企业长远计算机管理的企业目标模型，然后找出本行业的主体数据库，做数据结构分析，再进入应用项目的逻辑与物理设计，转入测试、维护。

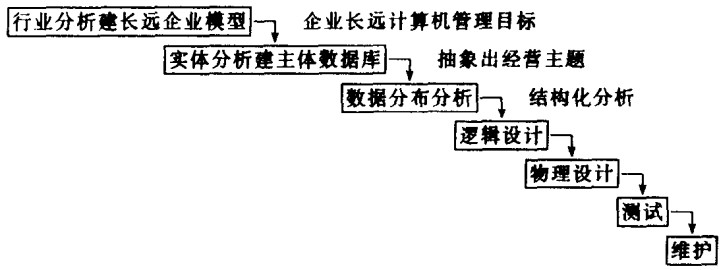

图 1.3 总体数据库规划模型

优点：起点高，不受行业水平限制。缺点：投资大，周期长，技术与社会性复杂，难度大。适应于现代化大型企业软件开发。

1.3 软件工程学及其基本原则

在软件应用领域中，尽管认识问题、解决问题的技巧和手段各不相同，但我们可以使用与具体应用无关的技术来研究如何设计软件，这些技术就构成了软件工程学的基础。

1.3.1 软件工程学的内容与目标

从内容上划分，软件工程学可分为理论、结构、方法、工具、环境、管理、规范等部分。理论和结构是软件开发技术的基础。方法、工具、环境构成软件开发技术。好的工具促进方法的研制，好的方法能改进工具。工具的集合构成开发环境。管理是技术实现与开发质量的保证。规范是开发遵循的技术标准。

- 软件工程学
  - 软件开发理论、结构
  - 软件开发技术
  - 软件工程管理、规范
- 软件开发理论、结构
  - 程序正确性证明理论
  - 软件可靠性理论
  - 软件成本估算模型
  - 软件开发模型
  - 模块划分原理
- 软件开发技术
  - 软件开发方法学
  - 软件工具
  - 软件环境
- 软件工程管理
  - 软件管理(人员、计划、标准、配置)
  - 软件经济(成本估算、质量评价)

软件工程学研究的基本目标是：

1. 定义良好的方法学，面向计划、开发维护整个软件生存周期的方法学。

2. 确定的软件成分，记录软件生存周期每一步的软件文件资料，按步显示轨迹。

3. 可预测的结果，在生存周期中，每隔一定时间可以进行复审。

软件工程学的最终目的，是以较少投资获得易维护、易理解、可靠、高效率的软件产品。

软件工程学是研究软件结构、软件设计与维护方法、软件工具与环境、软件工程标准与规范、软件开发技术与管理技术的相关理论。

1.3.2 软件工程学基本原则

为了开发出低成本高质量的软件产品，软件工程学应遵守以下基本原则：

一、分解

分解是分析解决复杂问题的重要手段和基本原则。其基本思想是从时间上或是从规模上将一个复杂问题分成若干个较小的、相对独立的、容易求解的子问题，然后分别求解。软件瀑布模型，结构化分析方法，结构化设计方法，Jackson 方法，模块化设计都运用了分解的原则。

二、尽量将可变因素隐藏在一个模块内，将怎样做的细节隐藏在下层，而将做什么抽象到上一层做简化，从而保证模块的独立性。这就是软件设计独立性要遵守的基本原则。模块化和局部性的设计过程使用了抽象和信息隐蔽的原则。

三、一致性

研究软件工程方法的目的之一，就是要使开发过程标准化，使软件产品设计有共同遵循的原则。要求软件文件格式一致，工作流程一致，软件开发过程要标准化，统一化。

四、确定性

软件开发过程要用确定的形式表达需求，表达的软件功能应该是可预测的，用可测试性，易维护性，可靠性，易理解性，高效率的指标来具体度量软件质量。

1.4 软件工程学的进展

随着软件工程学的广泛应用和不断发展，出现了许多新的思想和方法，如新软件开发模

型，软件重用，计算机辅助软件工程，软件工程的人工智能方法，软件自动生成等。

1.4.1 新的软件开发模型

B. W. Boehm 提出的瀑布(Water－fall)模型，关于计划、需求分析、软件设计、实现、测试、运行维护、评价的生存周期方法，不适于大型软件开发。M. A. Jackson 与 F. P. Brooks 提出的原型模型(Prototype)属新的软件开发模型，废弃型验证需求，进化型最后生成目标系统。适用于开发需求不确定的大型软件系统。有人不断改进原型模型，又提出智能原型模型，将进一步提高开发的效率。J. martin 提出战略数据库规划模型，是从信息化社会和计算机企业化管理的高度来构造各行业的系统目标，从而避免行业陈旧，软件也随之报废的悲剧。这一崭新模型，将伴随广域网络和全球网络的建成而发挥巨大作用。

1.4.2 软件再用

软件再用，将已开发的软件的知识，方法和标准，软件成分，不作修改或稍作修改可多次使用，构成新的软件，这与完全从头开发软件相比，软件再用能大大缩减开发成本，提高效率。设计出的软件能被再用，应具备如下条件：

(1)软件必须是模块化的，具有单一完整的功能，且被测试证明是正确的。

(2)软件结构清晰，具有很好的可读性、可理解性、规模适当。

(3)软件具有高度适应性，能适应各种使用环境。

利用可再用的软件成分来开发出新的软件的技术称为软件再用技术。再用方法有：

(1)软件组合。根据需要按照一定的规则把可再用的软件组合成所需要的软件系统或新的可再用软件。组合过程中保持再用性不变。一般要有公用数据库和软件库作为支持。组合 CAD 技术就是一例。

(2)软件生成技术。在可再用的软件库和知识库的支持下，根据需要，按照形式化的功能描述和生成机理，生成相似软件成分或软件系统。

1.4.3 计算机辅助软件工程 CASE

CASE(Computer－Adied Software Engineering)计算机辅助软件工程，是 80 年代中期发展起来，为提高软件工程设计效率，支持软件开发周期的前期的软件开发工具，现在正朝着支持软件开发全周期的方向发展。

1.4.4 软件自动生成器

不经过软件开发的漫长周期，直接用这个软件自动生成器生成软件需求的程序或目标。例如 APS 自动程序生成器，仅要用户做算法的构造，直接生成 C＋＋或 PASCAL 的目标程序，开发工作量降低到 80－95%，方便外行进行软件设计。

1.4.5 软件工程与人工智能

软件工程与人工智能是计算机科学的两个重要分支。软件工程是通过对软件生产周期的研究，寻求科学和工程化方法工具和支撑环境，达到开发出低成本高质量软件产品的目的。本身就是知识密集的智能活动。人工智能是对人的智能活动自身的研究，从中找出用计算机来替代人的智能活动一部分的软件设计过程。所以二者有着紧密的联系。人工智能软件的开发可