面向“十二五”高等学校精品规划教材
高等教育课程改革项目研究成果

软件工程
——理论与实践

宋礼鹏　张建华　编著

- 新：新思路、新领域、新技术、新变革
- 活：模块化、立体化、可扩展
- 精：精品、精心、精致

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

软件工程

理论与实践

宋礼鹏　张建华　编著

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

内容简介

本书包括4大部分，共13章：第1章介绍软件工程的历史和基本原理；第2章介绍常见的软件过程模型及其演化过程；第3章讨论软件工程的管理技术。第4章介绍软件需求分析的框架知识；第5章介绍了面向数据流的分析方法；第6章介绍面向对象的分析建模技术和UML建模语言。第7章介绍软件设计阶段的框架知识；第8章介绍了面向数据流的设计方法；第9章讨论从分析模型转换成设计模型的过程。第10章介绍软件验证的目标、软件验证与软件检验的区别等；第11章介绍白盒测试和黑盒测试技术；第12章介绍面向对象软件测试的案例设计以及面向对象的集成测试；第13章介绍遗留系统的进化问题、软件变更的不同策略、软件维护及其过程和软件再工程技术等。

图书在版编目（CIP）数据

软件工程：理论与实践/宋礼鹏，张建华编著．—北京：北京理工大学出版社，2011.7

ISBN 978-7-5640-4587-6

Ⅰ．①软…　Ⅱ．①宋…②张…　Ⅲ．①软件工程　Ⅳ．①TP311.5

中国版本图书馆CIP数据核字（2011）第093335号

出版发行 /北京理工大学出版社
社　　址 /北京市海淀区中关村南大街5号
邮　　编 /100081
电　　话 /(010)68914775(办公室)　68944990(批销中心)　68911084(读者服务部)
网　　址 /http：// www.bitpress.com.cn
经　　销 /全国各地新华书店
印　　刷 /天津紫阳印刷有限公司
开　　本 /787毫米×1092毫米　1/16
印　　张 /16
字　　数 /365千字
版　　次 /2011年7月第1版　2011年7月第1次印刷
印　　数 /1～4000册　　责任校对 /陈玉梅
定　　价 /32.00元　　责任印制 /王美丽

前　言

软件工程是计算机科学中年轻而又充满活力的重要分支，是人们为了克服软件危机而引入的一门工程学科。随着信息化社会的到来和计算机软件的广泛使用，软件工程越来越显示出重要的作用。

软件工程在高等学校计算机专业教学中有着重要的作用，是培养学生软件开发能力和项目管理能力的一门重要课程，其教学效果直接影响到学生毕业后从事软件开发和项目管理的能力。一本好的教材能帮助学生快速掌握软件开发的理论、技术和方法。出于这样的认识，在编写本书的过程中我们认真总结了多年的教学经验并充分讨论了各类软件工程教材的优缺点。希望本书能给读者带来一些收获。

在编写过程中，我们力求反映以下特色：

1）采用“两条线”的形式组织内容

首先，按照软件工程的基本阶段将整体内容划分为四个部分：需求、设计、软件验证和进化（概览除外）。其次，在每个部分均按一般理论、结构化方法、面向对象方法这一路线展开。

2）将“启发式”和“对比式”的思想方法运用在内容安排上

软件工程的新技术层出不穷，即使将最新的技术教给学生，毕业后也需要不停地学习。因此，我们在每个阶段对比结构化方法与面向对象方法，启发学生去思考软件工程方法进化的原则与方向。正所谓“授之以鱼”不如“授之以渔”。

3）一个案例贯穿整个过程

在需求和设计阶段使用同一个案例。该案例虽然规模不大，却有助于学生系统地学习软件工程各个阶段的技术。

本书的第一、二、三、四、五、七、八、十、十一章由宋礼鹏撰写，第六、九、十二、十三章由张建华撰写。限于作者的水平，难免有错误和不妥之处，希望得到广大读者的批评指正。

作　者

目 录

第一部分 概 览

第二部分 需 求

第三部分 设 计

第四部分 软件验证和进化

第一部分　概　览

第 1 章　概　述

【目标】

本章的目标是介绍软件工程这门学科和学习本书其余部分所需的框架知识。读完本章，读者将了解以下内容。

- 导致软件危机的原因；
- 什么是软件工程，学习软件工程的意义；
- 软件生命周期包括哪些阶段；
- 软件工程方法学及其演化过程。

【内容】

* *

软件是人类思维的产品，并且越来越成为国民经济和人类生活中必需的组成部分。软件的发展经历了一个阶段。最初，软件仅仅是计算机硬件的附属品，软件规模较小，成本相对于硬件也微乎其微。随着计算机硬件的普及，软件的需求越来越大，软件也变得越来越复杂。软件不仅成为影响系统性能的关键要素，在系统总成本中的比重也越来越大。因此，如何快速开发经济的、高质量的软件成为待解决的关键问题。

软件工程就是借助于工程化的思想、技术来解决软件开发过程中面临的问题，其目标是提高软件的质量与生产率，最终实现软件的工业化生产。从 1968 年北大西洋公约组织的计算机科学家在国际会议上首次提出软件工程的概念至今，软件工程的发展一直指引人们朝着彻底解决软件危机的方向前行。40 多年来，软件工程在与软件的互相推动中逐渐完善，形成了一整套理论知识体系。

软件工程方法学是在软件生命周期全过程中使用的一整套技术方法，包括：方法、工具和过程。软件工程方法学经历了 4 个阶段，代表了 4 种不同的解决软件危机的思想。

1.1 软件与软件危机

本节讨论软件的定义、特点，软件的发展过程，软件危机及其产生的原因。

1.1.1 软件

在20世纪50年代，软件伴随着第一台电子计算机的问世诞生了。接着，以写程序为职业的人也开始出现，这些人大多是经过训练的数学家和电子工程师。10年后，美国大学里开始出现计算机专业学位，专门教人们写软件。在随后的50年里，软件行业迅速发展成为推动人类社会发展的龙头产业，并造就了一批百万、千万富翁。随着信息产业的发展，软件对人类社会的发展越来越重要。

那么，如何给软件定义？软件是什么？

软件对于人类而言是一个全新的东西，其发展历史不过四五十年。人们对软件的认识经历了一个由浅到深的过程。在计算机系统发展的初期，硬件通常用来执行一个单一的程序，而这个程序又是为一个特定的目的而编制的。当时，大多数软件的使用者就是软件的编写者。这使得早期软件的开发方法存在许多先天的问题：① 软件的需求、设计是在某个人的头脑中完成的一个隐藏的过程，这个过程也没有什么系统的方法可以遵循。② 软件往往带有强烈的个人色彩，而且除了源代码，往往没有软件说明书等文档。

从20世纪60年代中期到20世纪70年代中期，计算机硬件系统经历了一个大的发展。在这一时期，软件开始作为一种产品被广泛使用，出现了专门给别人开发软件的“小型作坊”。但这些作坊仍然沿用早期的个体化软件开发方式，随着软件数量的急剧膨胀、软件需求的日趋复杂，维护软件的难度越来越大，软件开发的成本也令人吃惊的高，而失败的软件项目却屡见不鲜。“软件危机”就这样开始了！

“软件危机”促使人们开始对软件及其特性进行更深一步的研究，人们改变了早期对软件的不正确看法，认识到优秀的程序除了功能正确、性能优良之外，还应该容易看懂、容易使用、容易修改和扩充。

现在，被普遍接受的软件的定义如下。

软件是计算机系统中与硬件相互依存的另一部分，它包括程序、数据及其相关文档。其中程序是按事先设计的功能和性能要求执行的指令序列；数据是使程序能正常操纵信息的数据结构；文档是与程序开发、维护和使用有关的各种图文资料。

然而，软件远非一个形式化的定义所能刻画的。越来越多的软件包含有该软件应用领域的业务流程、规则等。因此，软件还是嵌入业务知识的系统。

软件是人工制品，但它是人类思维的产品，与传统的工业产品相比，软件有其独特的特点。

(1) 软件是一种逻辑实体，而不是具体的物理实体，具有抽象性。这个特点使它在开发、生产、使用、维护等方面与硬件相比有明显的差异。人们可以把它记录在纸上、内存、磁盘、光盘上，但无法看到软件本身的形态，必须通过观察、分析、思考、判断，才能了解软件的功能、性能等特性。

(2) 软件是开发，硬件是制造。一旦软件研制开发成功，就可以大量复制同一内容的副本。所以对软件的质量控制，重在软件的开发方面。

(3) 硬件存在机械磨损问题，但软件在使用过程中，没有磨损、老化的问题。软件在

生存周期后期不会因为磨损而老化，但会为了适应硬件、环境以及需求的变化而进行修改，而这些修改又不可避免地引入错误，导致软件失效率升高，从而出现软件退化。当修改的成本变得难以接受时，软件就会被抛弃。

（4）软件对硬件和环境有着不同程度的依赖性，包括不同的硬件平台和支持软件运行的其他软件系统。这导致了软件移植的问题。

（5）软件是定制的，而不是通过已有构件组装而成的。虽然人们提出了软件复用、构件技术，但是手工作坊式的软件开发方式仍然占主导地位，像硬件生产那样基于已有零部件进行组装，实现软件开发自动化一直是追求的梦想。

（6）软件是复杂的，而且会变得更加复杂。软件是人类思维和智能的产品，它的开发凝聚了大量的脑力劳动，是人类有史以来生产的复杂度最高的工业产品。在大型软件系统中，软件涉及大量的数据、状态和它们之间的逻辑关系，加上人类思维的复杂性和不确定性，导致系统的复杂性急剧增加，也使得软件的分析、设计、实现和测试变得相当困难。

此外，由于软件使用在社会的各行各业、方方面面，软件开发常常涉及其他领域的专门知识，这对软件工程师提出了很高的要求。

（7）软件的成本相当昂贵。软件开发是高智商者从事的工作，需要投入大量、高强度的脑力劳动，成本非常高，风险也大。现在软件的开销已远远超过了硬件的开销。

（8）软件工作牵涉到很多社会因素。许多软件的开发和运行涉及机构、体制和管理方式等问题，还会涉及人的观念和心理。这些人的因素，常常成为软件开发的困难所在，直接影响到项目的成败。

1.1.2 软件危机

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。20 世纪 60 年代末至 20 世纪 70 年代初，“软件危机”一词在计算机界广为流传。事实上，几乎从计算机诞生的那一天起，就出现了软件危机。如今，50 多年过去了，虽然软件开发的技术和工具不断改进，但是软件危机依然没有彻底消除。

概括来说，软件危机包含两方面问题：① 如何开发软件，以满足不断增长、日趋复杂的需求。② 如何维护数量不断膨胀的已有软件产品。

具体地说，软件危机主要有以下表现。

1）软件成本日益增长，在计算机系统的整个成本中所占比例越来越大

20 世纪 50 年代，软件成本在整个计算机系统成本中所占的比例为 10%～20%。但随着软件产业的发展，软件成本日益增长。相反，计算机硬件随着技术的进步、生产规模的扩大，价格却不断下降。这样一来，软件成本在计算机系统中所占的比例越来越大。到 20 世纪 60 年代中期，软件成本在计算机系统中所占的比例已经增长到 50%左右。而且，该数字还在不断地递增，图 1.1 所示显示了软件成本的上升趋势。

2）开发进度难以控制，延迟交付甚至取消项目的现象屡见不鲜

在软件开发过程中，用户需求变化等各种意想不到的情况层出不穷，令软件开发过程很难保证按预定的计划实现，给项目计划和论证工作带来了很大的困难。此外，由于软件是逻辑、智力产品，软件的开发需建立庞大的逻辑体系，在硬件生产中常用的加快进度的方法（如增加人力）不能用在软件开发上。IBM360 机的操作系统项目负责人 F. D. Brooks

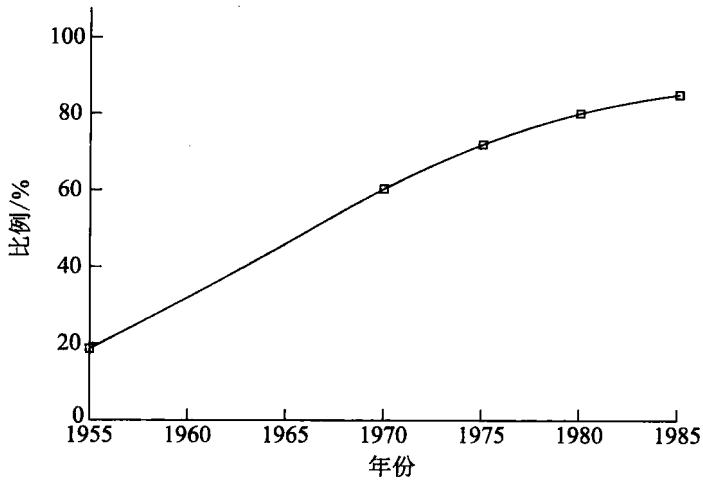

图 1.1 软件成本占系统总成本的比例

曾经提出："在已拖延的软件项目上，增加人力只会使其更难按期完成。"事实上，软件系统的结构很复杂，各部分联系极大，盲目增加软件开发人员并不能成比例地提高软件开发能力。相反，随着人员数量的增加，人员的组织、协调、通信、培训和管理等方面的问题将更为严重。

3）软件存在着错误多、性能低、不安全、不可靠等质量问题

软件项目即使能按预定日期完成，结果却不尽如人意。1965—1970 年，美国范登堡基地发射火箭多次失败，绝大部分故障是由应用程序错误造成的。程序的一些微小错误可以造成灾难性的后果。例如，有一次，在美国肯尼迪发射一枚阿脱拉斯火箭，火箭飞离地面几十英里高空开始翻转，地面控制中心被迫下令炸毁。后经检查，发现是飞行计划程序里漏掉了一个连字符。就是这样一个小小的疏漏，造成了这支价值 1 850 万美元的火箭试验失败。随着软件越来越多地用在国民经济和人类社会的高安全性、高可靠性系统中，软件的质量保证成为人们关注的焦点。

4）软件维护困难

正式投入使用的软件，总是存在着一定数量的错误，在不同的运行条件下，软件就会出现故障，因此需要维护。软件即使能够正常使用，由于软硬件环境的变化或用户提出新需求，也需要进行修改。但是，由于在软件设计和开发过程中，没有严格遵循软件开发标准，各种随意性很大，没有完整的真实反映系统状况的记录文档，给软件维护造成了巨大的困难。特别是在软件使用过程中，原来的开发人员可能因各种原因已经离开原来的开发组织，使得软件几乎不可维护。有资料表明，为维护软件支付的费用占全部硬件和软件费用的 40%～75%。

为了克服软件危机，首先需要分析导致软件危机的原因。从软件危机的种种表现和软件作为逻辑产品的特殊性可以发现，产生软件危机的原因包括以下几个方面。

1）用户需求不明确，致使软件开发周期延长、成本增加，甚至导致项目失败

用户需求是软件开发的基础，然而获取用户需求又是一件非常困难的任务。在软件开

发过程中，用户需求不明确问题主要体现在：在软件开发出来之前，用户自己也不清楚软件的具体需求；用户对软件需求的描述不精确，可能有遗漏、有二义性，甚至有错误；在软件开发过程中，用户还提出修改软件功能、界面、支撑环境等方面的要求；由于知识背景的差异、交流方法或描述工具的原因，软件开发人员对用户需求的理解与用户本来的愿望有差异。

2）缺乏正确的理论指导，开发过程不规范，开发人员各自为战，缺少整体规划

缺乏有力的方法学和工具方面的支持、没有对软件进行整体规划、没有约束开发人员的文档资料、过分地依靠程序设计人员在软件开发过程中的技巧和创造性，加剧软件产品的个性化，也是发生软件危机的一个重要原因。

3）软件规模越来越大

随着软件应用范围的不断扩大，软件规模也越来越大。大型软件项目需要组织一定的人力共同完成，多数管理人员缺乏管理方面的经验，而多数软件开发人员又缺乏开发大型软件系统的经验。各类人员的信息交流不及时、不准确，有时还会产生误解，致使软件开发活动无法有效进行，工期一拖再拖。

4）软件复杂度越来越高

软件不仅仅是在规模上快速地发展扩大，其复杂性也急剧地增加。软件产品的特殊性和人类智力的局限性，导致人们无力处理复杂问题。

与40年前相比，软件开发技术已经取得了长足的进步。然而，软件的规模和复杂性也呈现指数型增长，同时，对软件的质量也提出更高的要求。因此，软件危机依然是亟待解决的问题。

1.2 软件工程

认真研究和分析了软件危机背后的真正原因，便开始探索用工程的方法进行软件生产的可能性，即用现代工程的概念、原理、技术和方法进行计算机软件的开发、管理和维护。于是，计算机科学技术的一个新领域——软件工程诞生了。

1.2.1 软件工程的概念

软件工程的定义有许多，下面给出两个典型的定义。

1968年秋季，北大西洋公约组织举行的国际会议上首次给出了软件工程的定义：软件工程是为了经济地获得能够在实际机器上高效运行的可靠软件而建立和使用的一系列好的工程化原则。

1993年《IEEE Standard Glossary of Software Engineering Terminology》给出了一个更全面的定义：软件工程是，① 将系统性的、规范化的、可定量的方法应用于软件的开发、运行和维护，即将工程化应用到软件上。② 对①中所述方法的研究。

从软件工程的定义可以看出，软件工程不仅强调软件开发技术的研究，还包括软件项目管理。统计数据表明，大多数软件开发项目的失败，并不是由于软件开发技术方面的原因。它们的失败是由于不适当的管理造成的。遗憾的是，尽管人们对软件项目管理重要性的认识有所提高，但在软件管理方面的进步远比在设计方法学和实现方法学上的进步小。

此外，软件工程是将工程的思想应用于软件领域。与一切工程领域一样，软件工程师也需要不断积累经验，从已有的理论、方法和工具中选择最适宜的，同时还要不断探索新的理论、方法和工具。

1.2.2 软件工程的目标

软件工程的目标就是运用最先进的技术和经过时间检验证明正确的管理方法来提高软件的质量和生产率，也就是在给定成本、进度的前提下，开发出高质量的软件产品，最终实现软件生产自动化。通常质量和生产率是一对矛盾，但这两者又都是软件开发追求的目标。因此，好的软件工程方法要能同时提高软件质量和生产率。

概括地说，软件工程就是要解决软件危机问题，具体到当前软件工程主要面临如下问题。

1. 遗留系统带来的挑战

现在使用的许多大型系统都是几年前，甚至十几年前开发的。这些系统在使用过程中不断地被修改，因此不断地退化，但同时这些系统也不断地被注入大量的业务知识、规程。现在面临的问题是这些系统维护起来的成本很高，但彻底淘汰的代价也无法承受。如何以合理的成本维护和更新系统，同时又要不断交付基本的业务服务是带给软件工程的一大难题。

2. 软件开发方式和运行环境多样性的挑战

随着网络时代的来临，软件的开发由过去集中在专门的机构中封闭实施，转变为在网络环境下基于开放源码的方式由许多开发者协作完成。同时，系统也越来越要求像基于网络的分布式系统一样运行，但这些网络中包含各种不同类型的软硬件平台。因此，必须开发出新的技术来制作可靠的、灵活的软件来应对网络化的挑战。

3. 软件交付上的挑战

传统的软件工程技术开发高质量软件需要消耗大量时间，而今天的软件必须具有快速制作、快速响应变化的特点。因此，必须找到新一代软件开发技术，在不损害软件质量的前提下快速开发大型的、复杂的系统。

4. 高可信软件带来的挑战

随着软件在国民经济和人们生活中的广泛使用，人们更加关注软件的质量问题，尤其当软件是通过远程方式获取时。除了软件的正确性和性能外，软件的可靠性、安全性和保密性等可信性质是人们关注的焦点。传统的软件工程技术通过测试来检查软件中存在的问题，但无法验证软件的可信性。目前，基于严格数学模型的形式化开发方法为高可信软件的开发提供了一种解决思路。

虽然人类彻底解决软件危机的目标尚待时日，但软件工程的研究与应用已经取得很大成就，它在软件开发方法、工具、管理等方面的应用极大地缓解了软件危机造成的被动局面。

1.3 软件生命周期

概括地说，软件生命周期是由软件规格描述、软件开发、软件确认和软件演进 4 个基

本活动组成的，每个基本活动又进一步划分成若干个阶段。

软件规格描述的主要任务是解决“做什么”的问题，即确定工程必须完成的总目标和可行性，导出实现工程目标应该采用的策略及系统必须完成的功能；估计完成该项工程需要的资源和成本，并且制订工程进度表。这个时期的工作通常又称为系统分析，由系统分析员负责完成。软件规格描述通常进一步划分成 3 个阶段，即问题定义、可行性研究和需求分析。

软件开发的主要任务是解决“如何做”的问题，即具体设计和实现在前一个时期定义的软件，它通常由下述 3 个阶段组成：总体设计，详细设计，编码实现。

软件确认的主要任务是“确认实现的系统满足用户的要求”，即依据规格说明来测试所实现的软件。软件测试包括单元测试、集成测试、系统测试等。通常单元测试合并在编码实现阶段进行。

软件演进又称为软件维护，通常有 4 类维护活动：改正性维护，当软件在使用过程中发现错误时应该加以改正；适应性维护，当环境改变时应该修改软件以适应新的环境；完善性维护，当用户有新要求时应该及时改进软件以满足用户的新需要；预防性维护，即修改软件为将来的维护活动做准备。

下面简单介绍软件生命周期每个阶段的基本任务。

1. 问题定义

问题定义阶段必须明确要解决的问题是什么。尽管确切地定义问题的必要性是十分明显的，但是在实践中它却可能是最容易被忽视的一个步骤。通过问题定义阶段的工作，系统分析员应该提出关于问题性质、工程目标和规模的书面报告。通过对系统的实际用户和使用部门负责人的访问调查，分析员扼要地写出他对问题的理解，并在用户和使用部门负责人的会议上认真讨论这份书面报告，澄清含糊不清的地方，改正理解不正确的地方，最后得出一份双方都满意的文档。

问题定义阶段是软件生命周期中最简短的阶段，一般只需要 1 天甚至更少的时间。

2. 可行性研究

这个阶段要明确在成本和时间的限制条件下，上一个阶段所确定的问题是否有行得通的解决办法。为此，系统分析员需要在较抽象的高层次上进行一次系统分析和设计过程。在用户的配合下，由分析员提出解决问题的候选方案，然后对每个方案从技术、经济、法律和操作等方面进行可行性研究。

可行性研究的结果将为使用部门做出最终是否开发软件项目的决定提供重要依据。可行性研究有助于尽早发现那些不值得投资的工程项目，避免了后期更大的浪费。

3. 需求分析

这个阶段的任务主要是明确为了解决这个问题，目标系统必须做什么。为此，分析员要通过各种途径与用户沟通，获取他们的真实需求，并通过建模技术来表达这些需求。通常需要从功能、数据、行为等方面描述系统的静态和动态特性。

在需求分析阶段确定的系统逻辑模型是以后设计和实现目标系统的基础，必须准确完整地体现用户的要求。系统分析员通常都是计算机软件专家，技术专家一般都喜欢很快着手进行具体设计，然而，一旦分析员开始谈论程序设计的细节，就会脱离用户，使他们不

能继续提出他们的要求和建议。因此，分析人员必须和用户共同讨论决定哪些需求是必须的，然后编写出需求规格说明书和系统用户手册，在用户确认之后才能进入下一个阶段。

4. 总体设计

又称为概要设计，这个阶段的主要任务是确定系统的架构，即给出软件的体系结构。首先，架构设计师必须制订几种可能的设计方案。例如，目标系统的一些主要功能是用计算机自动完成还是用人工完成；如果使用计算机，那么是使用批处理方式还是人机交互方式；信息存储使用传统的文件系统还是数据库等。

选择最佳的方案，把上一阶段确定的需求映射成相应的软件体系结构。软件体系结构明确了系统由哪些子系统构成、子系统之间的关系、每个子系统由哪些模块组成、模块之间的关系，并将这样一种设计结果用适当的模型表示出来。

5. 详细设计

也称为模块设计，这个阶段的主要任务是进一步确定如何实现这个系统，即设计总体设计阶段所给出的每个模块的内部算法流程和数据结构。这个阶段不是要编写每个模块的代码，而是运用适当的模型来刻画软件设计人员对模块算法的构思。

6. 编码和单元测试

这个阶段的任务就是实现已做的设计，即写出正确的、容易理解和维护的程序代码。程序员应该选取一种适合的编程语言，把软件的设计结果转换成在机器上可以运行的程序代码。此外，在转换过程中，对每个设计模块要进行单元测试，并通过调试排除测试中发现的错误。

7. 集成和系统测试

这个阶段就是通过各种类型的测试来提高软件质量，使软件达到预定的要求。本阶段主要做两类测试：集成测试和系统测试。

集成测试又称为组装测试，即把经过单元测试的模块按设计规定的某种策略组装起来，在组装过程中对程序进行必要的测试，重点测试模块接口部分的正确性。

系统测试又称为验收测试，是根据软件需求规格说明书的要求，对各项需求逐一进行测试。系统测试是在用户的参与下对目标系统进行的验收。

8. 软件维护

软件维护发生在软件已经正式交付使用之后，维护的主要目标是使系统持久地满足用户的需要。软件维护的任务包括：提出维护要求，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，测试程序，复查验收等一系列步骤。

软件维护发生在软件运行后退役前，每一次维护都经历了一次压缩和简化了的软件定义和开发的全过程，所以软件维护的工作量很大，应该在开发过程中就考虑将来的维护，使软件具有尽可能好的可维护性。

1.4 软件工程方法学

软件工程方法学是指用在软件生命周期全过程中的一整套技术方法的集合。软件工程方法学包括 3 个要素。

(1) 过程，获取高质量软件所需要的一系列任务框架，它规定了任务完成的顺序，以及任务完成的交付物等。过程为软件开发明确了做什么的问题。

(2) 方法，为完成过程中规定的各项任务提供的技术集合，解决软件如何做的问题。软件工程的方法覆盖面很广，包括需求分析、设计、编码、测试和维护等各个阶段的技术，它经历了4个阶段的发展，每个阶段都是某种解决软件危机的思想在具体技术上的体现。目前较为常用的是传统方法（结构化方法）和面向对象方法。

(3) 工具。软件工具是为软件工程方法提供了自动的或半自动的软件支撑环境。

1.4.1 结构化方法学

20世纪60年代末到20世纪70年代，为了克服软件危机提出了“软件工程”一词，将软件的开发纳入了工程化的轨道，基本形成了软件工程的概念、框架、技术和方法，称为结构化方法学，又称为传统方法学或生命周期方法学。

人类解决复杂问题时普遍采用的一个策略就是“各个击破”，也就是对问题进行分解然后再分别解决各个子问题的策略。在分解问题时应该遵循的一条基本原则就是使各个子问题的任务彼此间尽可能相对独立，同一子问题的各项任务性质尽可能相同，从而降低每个子问题的复杂程度，简化不同子问题之间的联系，有利于整个问题的解决。

结构化方法学就是将人类的这种结构化划分思想应用到软件的分析、设计、实现和维护中，也就是从时间角度对软件开发和维护的复杂问题进行分解，把软件生存的漫长周期依次划分为若干个阶段，然后逐步完成每个阶段的任务。对于任何两个相邻的阶段而言，前一个阶段的结束标注是后一个阶段的开始标准。

结构化软件工程方法学划分的每个阶段有相对独立的任务，在每个阶段都采用科学的管理技术和良好的技术方法，而且在每个阶段结束之前都从技术和管理两个角度进行严格的审查，合格之后才开始下一阶段的工作，这就使软件开发工程的全过程以一种有条不紊的方式进行，保证了软件的质量，特别是提高了软件的可维护性。

结构化方法学是历史最悠久的软件工程方法学，为推动软件开发技术的发展起到重要的作用。然而，结构化划分要么面向行为，要么面向数据，没有将数据和对数据操作的行为结合起来的技术。众所周知，软件系统本质上是信息处理系统，数据和处理数据的操作本身是一体的、密不可分的，硬性将其分离的做法不符合人类认识现实世界的规律，也难以被人们用来描述现实问题。

1.4.2 面向对象方法学

与传统方法不同，面向对象方法是一种以数据或信息为主线，把数据和行为相结合的方法。面向对象方法把对象作为由数据及可以施加在这些数据上的操作所构成的统一体。对象与传统的数据有本质区别，它不是被动地等待外界对它施加操作，相反，它是进行处理的主体。必须发消息请求对象主动地参与它的某些操作，处理它的私有数据，而不能从外界直接对它的私有数据进行操作。

面向对象本身是一种思想，最初这种思想被用在程序设计语言上。在软件工程方法学演变的历史进程中，正是由于人们认识到结构化方法学存在的问题，才有意识地将面向对象思想引入到软件分析、设计中，逐渐形成完整的面向对象方法学。

面向对象方法尽可能模拟人类习惯的思维方式，使开发软件的方法与过程尽可能接近人类认识世界解决问题的方法和过程，也就是使描述问题的问题空间（问题域）与实现解法的解空间（求解域）在结构上尽可能一致。相较于传统的方法，面向对象方法的主要优势在于以下几个方面。

（1）与人类习惯思维方式一致，这种一致性有利于分析员理解问题域和系统任务，也有利于分析员与用户的交流。

（2）在整个开发过程中，面向对象方法保持了概念和模型表示的一致性，填平了语言之间的鸿沟，促使各项开发活动的平稳过渡，有利于软件开发的迭代进行。图 1.2 所示显示了面向对象方法与传统方法在这一点上的区别。

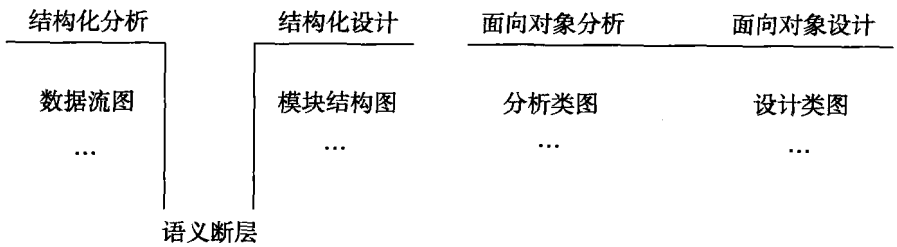

图 1.2　传统方法与面向对象方法

（3）用面向对象方法建立的系统可维护性好。首先，由于系统是由对象构成的，而各种维护通常是功能性变化，反映到系统里往往只需要修改对象内部的操作，系统的整体结构无须调整，因此系统稳定性好，易于维护。其次，对象是包括属性和操作的独立单元，对象间只能通过消息通信。因此，修改部分对象对其他部分的影响小，这种系统的可修改性好，易于维护。

（4）对象是从问题域提取出来的、封闭性和独立性很好的模块单元，使其具有普适性，也易于复用。

【习题】

1.1　“软件≠程序”，请解释这句话为什么是正确的。

1.2　软件和其他人工制品有何本质区别？

1.3　试分析软件危机出现的必然性。

1.4　什么是软件工程，软件工程为解决软件危机带来了什么？

1.5　软件生命周期有哪些阶段，划分的标准是什么？

1.6　试分析结构化方法学出现的历史背景及其必然性。

1.7　简述面向对象方法学的优点，其在方法学演化中的历史必然性。