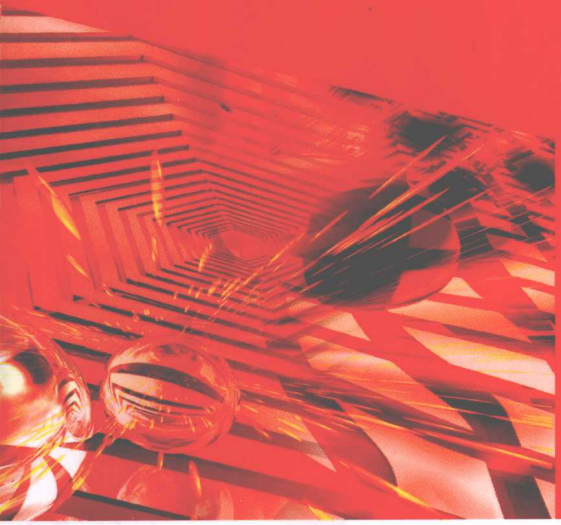
普通高等教育“十一五”计算机类规划教材



软件工程基础与实例分析

■ 王阿川 主编

机械工业出版社
CHINA MACHINE PRESS

普通高等教育“十一五”计算机类规划教材

软件工程基础与实例分析

王阿川　主　编
张剑飞　兰　翠　副主编

机 械 工 业 出 版 社

本书从实用的角度出发，系统地介绍了软件工程基本知识，包括传统的软件工程和面向对象的软件工程两大部分。在传统的软件工程部分，按照软件生存周期的顺序介绍各个阶段的任务、过程、方法、工具和文档编写规范；在面向对象的软件工程部分，介绍了面向对象的分析与设计方法以及统一建模语言 UML 的相关知识。同时，配有开发实例和软件文档模板。

本书理论与实践相结合，内容循序渐进、深入浅出、通俗易懂、侧重应用。

本书不仅可作为高等学校计算机专业软件工程课程的教材或教学参考书，也可作为通信工程、电子信息工程、自动化等相关专业的软件工程教材，还可供软件工程师、软件项目管理者和应用软件开发人员阅读参考。

本书配有免费电子课件，欢迎选用本书作为教材的老师登录 www. cmpedu. com 注册下载或发邮件到 xufan666@ 163. com 索取。

图书在版编目（CIP）数据

软件工程基础与实例分析/王阿川主编．—北京：机械工业出版社，2010. 8

普通高等教育“十一五”计算机类规划教材

ISBN 978-7-111-31122-5

Ⅰ. ①软… Ⅱ. ①王… Ⅲ. ①软件工程－高等学校－教材 Ⅳ. ① TP311. 5

中国版本图书馆 CIP 数据核字（2010）第 134845 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）

策划编辑：徐 凡 责任编辑：徐 凡

版式设计：张世琴 责任校对：李秋荣

封面设计：张 静 责任印制：乔 宇

北京机工印刷厂印刷（三河市南杨庄国丰装订厂装订）

2010 年 8 月第 1 版第 1 次印刷

184mm×260mm · 14 印张 · 345 千字

标准书号：ISBN 978-7-111-31122-5

定价：24. 00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务

社服务中心：（010）88361066

销 售 一 部：（010）68326294

销 售 二 部：（010）88379649

读者服务部：（010）68993821

网络服务

门户网：http：//www. cmpbook. com

教材网：http：//www. cmpedu. com

前　言

软件工程学是指导软件生产、维护的一门工程科学，从20世纪60年代末起迅速发展，现已成为计算机科学中的一个重要分支，它的研究范围非常广泛，包括技术、方法、工具和管理等许多方面。

本书从实用的角度出发，系统地介绍了软件工程基本知识，包括传统的软件工程和面向对象的软件工程两大部分。在传统的软件工程部分，按照软件生存周期的顺序，介绍各个阶段的任务、过程、方法、工具和文档编写规范。在面向对象的软件工程部分，介绍了面向对象的分析与设计方法以及统一建模语言UML的相关知识。

本书尽量用实例来解释概念，用案例来演绎方法和原理，并选择典型的软件工程开发实例及文档实例进行剖析，使读者能够在牢固掌握理论知识的同时，思考并尝试解决实际的问题。

本书文字通俗易懂、概念清晰、深入浅出、实例丰富、实用性强，既可作为高等学校计算机专业课程的教材或教学参考书，也可作为通信工程、电子信息工程、自动化等相关专业的软件工程教材，还可供软件工程师、软件项目管理者和应用软件开发人员阅读参考。

本书的第1章由东北林业大学王阿川编写；第6～8章由黑龙江科技学院张剑飞编写；第3～5章由黑龙江科技学院兰翠编写；第12、13章由黑龙江工程学院吕瑞编写，第2、9～11章由华北科技学院丁智斌编写。华北科技学院郭红担任本书主审。

在本书编写过程中，参考了大量的相关资料，同时得到了各方面有关专家的大力支持和帮助，在此一并表示感谢。由于时间仓促，水平有限，书中难免有不妥之处，敬请读者不吝赐教。

本书配有免费电子课件，欢迎选用本书作为教材的老师登录www.cmpedu.com注册下载或发邮件到xufan666@163.com索取。

编　者

2010年2月

目　录

第1章　软件工程概述

本章要点：本章主要介绍软件和软件工程的基础知识、比较常用的软件开发模型、典型软件开发方法以及软件文档写作等方面内容。

教学目标：了解软件的特点、软件的分类、软件危机产生的原因及解决途径，掌握软件工程的基本原理、软件生存周期的意义以及软件生存周期各阶段的基本任务，掌握瀑布模型、快速原型模型、螺旋模型、喷泉模型等常用的软件开发模型和面向数据流、面向数据结构、面向对象的软件开发方法，认识软件文档的作用、种类和写作要求。

1.1　软件

1.1.1　软件的概念及特点

1. 软件的概念

“软件”这一名词在20世纪60年代初从国外引进，当时人们无法说清它的具体含义，也无法解释英文单词“software”，于是有人把它翻译成“软件”或“软制品”，现在应该统一称其为软件。早期，人们认为软件就是源程序。随着人们对软件及其特性的更深层的研究，认为软件不仅仅包括程序，还应包含其他相关内容。目前，对软件通俗的解释为：

软件 = 程序 + 数据 + 文档资料

其中，程序是按照事先设计的功能和性能要求执行的指令序列；数据是程序运行的基础和操作的对象；文档是有关程序开发、维护和使用的各种图文材料。

2. 软件的特点

1）软件是一种抽象的逻辑实体。人们无法看到其具体形态，只能通过观察、分析、思考、判断等方式去了解它的特性功能。

2）软件是一种通过人们智力活动，把知识与技术转化为信息的一种产品，是在研制、开发中被创造出来的。

3）软件需要维护。主要是因为在软件的生存周期中，为了使它能够适应硬件、软件环境的变化以及用户新的要求，必须进行多次修改（维护）。

4）软件的开发和运行受到计算机硬件、操作系统的限制。

5）软件开发至今尚未摆脱手工开发方式。很多软件仍然是“定制”的，这使得软件的开发效率受到很大限制。

6）软件的开发是一个复杂的过程。

7）软件的成本较高。软件开发需要投入大量的、高强度的脑力劳动，成本较高。

1.1.2　软件的分类

为了便于人们根据不同的应用要求选择相应的软件，也鉴于不同类型的工程对象，对其

进行开发和维护有着不同的要求和处理方法，因此，从不同角度出发对软件进行分类，更加符合实际情况。

1. 基于软件功能的划分

（1）系统软件

系统软件是与计算机硬件紧密配合以使计算机的硬件部分与相关软件及数据协调、高效工作的软件。例如，操作系统、数据库管理系统、设备驱动程序以及通信处理程序等。

（2）支持软件

支持软件是协助用户开发软件的工具性软件，包括帮助程序人员开发软件产品的工具和帮助管理人员控制开发进程的工具。

（3）应用软件

应用软件是在特定领域内开发、为特定目的服务的一类软件，其中，商业数据处理软件占有很大比例，另外，还有工程与科学计算软件、计算机辅助设计/计算机辅助制造（CAD/CAM）软件、系统仿真软件、智能产品嵌入软件（如汽车油耗系统、仪表盘数字显示、刹车系统）以及人工智能软件（如专家系统、模式识别）等。此外，事务管理、办公自动化、中文信息处理、计算机辅助教学（CAI）等方面的软件也在迅速发展之中。

2. 基于软件规模的划分

按软件开发所需要的人力、时间以及完成的源程序行数，可划分为 6 种不同规模的软件，即微型软件、小型软件、中型软件、大型软件、甚大型软件和极大型软件，见表 1-1。

表 1-1　软件规模分类

类　别	参加人员数	研制期限	产品规模（源程序行数）
微型	1	1～4 周	500
小型	1	1～6 月	1000～2000
中型	2～5	1～2 年	5000～50 000
大型	5～20	2～3 年	50 000～100 000
甚大型	100～1000	4～5 年	1000 000 以上
极大型	2000～5000	5～10 年	10 000 000 以内

3. 基于软件工作方式的划分

（1）实时处理软件

实时处理软件是在事件或数据产生时，立即处理并回馈信号，控制需要监测和控制的过程的软件。主要包括数据采集、分析、输出 3 部分。其特点是对外界变化的反映及处理有严格的时间限定。

（2）分时软件

分时软件允许多个用户同时使用计算机。系统把处理机时间轮流分配给联机用户，但用户的感觉是只有自己在使用计算机。

（3）交互式软件

交互式软件实现人机通信。这类软件接受用户给出的信息，但在时间上没有严格的限定。这种工作方式给予用户很大的灵活度。

（4）批处理软件

批处理软件是把一组输入作业或一批数据以成批处理的方式一次运行，按顺序逐个处理完的软件。

1.1.3 软件危机的原因及解决途径

软件危机指的是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。其实这些问题并不仅仅是那些不能正常运行的软件才具有的毛病，几乎所有的软件都不同程度地存在或多或少的问题。1968 年北大西洋公约组织（NATO）的计算机科学家在联邦德国召开的国际学术会议上第一次提出了“软件危机”（Software Crisis）这个名词。

概括来说，软件危机包含两方面问题：一方面是如何开发软件，以满足不断增长、日趋复杂的需求；另一方面是如何维护数量不断膨胀的软件产品。

1. 产生软件危机的原因

软件危机产生的原因：一方面是与软件本身的特点有关，另一方面是由软件开发和维护的方法不正确有关。其根本原因，主要表现为：

（1）忽视软件开发前期的需求分析

经验告诉我们，对用户要求没有准确完整的认识就匆忙着手编写程序，编码阶段开始得越早，完成整个工程的时间反而越长。

（2）没有统一的、规范的方法论的指导

在开发过程中文件资料不齐全，忽视人与人之间的交流，缺乏有力的方法论的指导，加剧了软件产品的个性化，以致产生疏漏和错误。

（3）忽视软件文档，造成开发效率低下

软件文档的编制在软件开发工作中占有突出的地位和相当的工作量。高效率、高质量地开发、管理和维护文档对于软件开发人员、软件维护人员以及用户都至关重要。

（4）忽视测试阶段的工作，提交用户的软件质量差

事实上，对于任何软件来讲，错误是不可避免的。为了尽量减少提交给用户软件中的错误，就需要在测试阶段找出软件中存在的错误，并及时加以修改。

（5）轻视软件的维护

在一个软件漫长的维护期中，必须注意改正软件使用中发现的每一个潜在的错误，以满足用户不断增长的需要。

2. 解决软件危机的途径

为了更好地解决软件危机，既需要技术措施，还要有必要的组织管理措施。先进的开发方法和工具可以保证软件的质量。严密组织、严格管理和各类人员工作的协调一致都是必不可少的因素。经过不断实践和总结，得出一个结论：按工程化的原则和方法组织软件开发工作是有效的，是摆脱软件危机的一个主要出路。软件工程正是从管理和技术两方面研究如何更好地开发和维护计算机软件的学科。

1.2 软件工程概念

1.2.1 软件工程的定义和内容

1. 软件工程的定义

1968 年 10 月，北大西洋公约组织（NATO）科学委员会在联邦德国开会讨论软件可靠性与软件危机的问题，会议上，Fritz Bauer 首次提出了“软件工程”的概念。后来，人们曾经多次给出了有关软件工程的定义。1993 年 IEEE 为软件工程的定义是：软件工程是将系统化的、规范化的、可度量的途径应用于软件的开发、运行和维护的过程，即将工程化应用于软件的方法的研究。

2. 软件工程的内容

软件工程是一种层次化的技术，如图 1-1 所示。和其他工程方法一样，软件工程是以质量为关注焦点，以相关的现代化管理为理念。

软件工程的基础是过程层。软件工程的过程是为获得软件产品，在软件工具支持下由软件人员完成的一系列软件工程的活动，贯穿于软件开发的各个环节。它定义了方法使用的顺序、要求交付的文档资料，是软件开发各个阶段完成的标志。

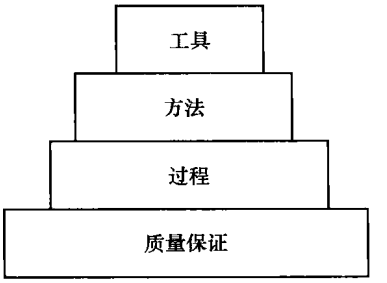


图 1-1 软件工程层次图

软件工程的方法是为软件开发提供了“如何做”的技术，通常包括以下内容：与项目有关的计算和各种估算方法、需求分析、设计的方法、编码、测试和维护等。

软件工程的工具为软件工程方法提供了自动的或半自动的软件支撑环境，辅助软件开发任务的完成。现有的工具覆盖了需求分析、系统建模、代码生成、程序调试和软件测试等多个方面，形成了集成化的软件工程开发环境 CASE（Computer-Aided Software Engineering，计算机辅助软件工程），以便提高软件开发效率和软件质量，降低开发成本。

1.2.2 软件工程的基本原理

自从 1968 年提出“软件工程”这一术语以来，研究软件工程的专家学者们陆续提出了 100 多条关于软件工程的准则或信条。美国著名的软件工程专家 Boehm 综合这些专家的意见，并总结了多年的开发软件的经验，于 1983 年提出了软件工程的 7 条基本原理。

1）用分阶段的生存周期计划严格管理开发过程。

2）坚持进行阶段评审。

3）实行严格的产品控制。

4）采用现代程序设计技术。

5）明确地规定开发小组的责任和产品标准。

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。

从上述7条基本原理，相互独立，缺一不可。实践过程中，可以对这些原理进行细化和再生，灵活运用这些原理指导软件开发。

1.3 软件生存周期

概括地说，软件生存周期就是从提出软件产品开始，直到该软件产品被淘汰的全过程。一般来说，软件生存周期包括计划、开发、运行三个时期，每一时期又可分为若干个更小的阶段。本节介绍的软件生存周期分为软件系统的可行性研究、需求分析、软件设计（概要设计和详细设计）、编码、软件测试、运行与维护。它们之间的关系如图1-2所示。

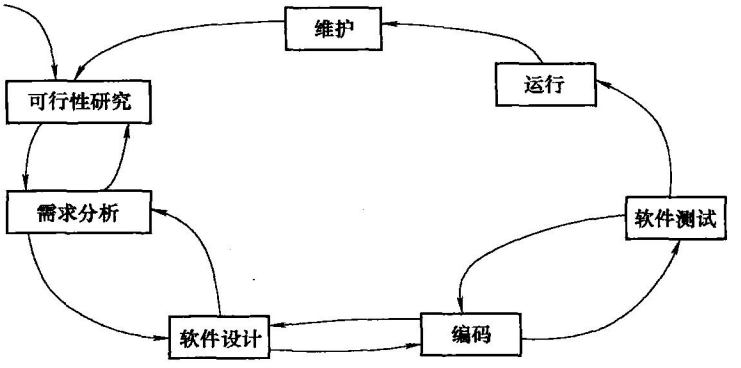


图1-2 软件的生存周期

软件生存周期每个阶段的基本任务如下：

（1）可行性研究

可行性研究阶段主要确定软件的开发目标及其可行性，给出其在功能、性能、可靠性以及接口等方面的要求。可行性研究由系统分析员和用户合作探讨，并且对可利用资源（计算机硬件、软件、人力等）、成本、可取得效益、开发的进度做出估量，制定任务实施计划，连同可行性报告提交给管理部门。

（2）需求分析

需求分析主要解决待开发软件要“做什么”的问题，包括需求的获取、分析、规格说明、变更、验证、管理等一系列工作。软件开发人员与用户共同讨论决定，哪些需求是可以满足的，并且加以确切地描述，然后，编写出软件需求说明书或系统功能说明书和初步的系统用户手册，提交给管理部门。

（3）软件设计

软件设计主要解决待开发软件“怎么做”的问题，软件设计通常可分为概要设计和详细设计。概要设计的任务是设计软件系统的体系结构，也就是确定程序由哪些模块组成以及模块间的关系。详细设计是具体设计每个模块，确定实现模块功能所需要的算法和数据结构。

（4）编码

编码是软件开发过程中的生产步骤。具体就是将软件转化为计算机代码，对其功能用某一种特定的计算机语言进行描述。编写出的程序要求具有结构性，并且易读，重要的是要与

设计要求一致。

（5）软件测试

软件测试的目的是确认软件的质量，一方面是确认软件做了你所期望的事情，另一方面是确认软件以正确的方式来做这些事情。首先进行单元测试，查找各个模块内部功能结构上存在的问题，其次进行集成测试，查找模块间联合工作存在的问题，最后进行确认测试、系统测试，决定软件产品质量是否过关，能否交用户使用。

（6）运行与维护

软件产品开发完成投入使用后可能运行若干年。在运行过程中可能因为各方面原因需要进行修改，硬件变更、操作系统升级、平台移植等问题都可能需要对软件进行维护。

1.4 常用软件开发过程模型

软件工程是基于问题求解的，要解决问题必须找到求解问题的策略。该策略包括软件的过程、方法、工具三个层次，被称为软件开发过程模型。该模型规定了把生存周期划分成哪些阶段及各个阶段的执行顺序。软件开发过程模型的选择依赖于项目和应用的性质、所采用的方法和工具、需要交付的产品和软件开发的控制。因此，选择开发过程模型对于项目的开发至关重要。

1.4.1 瀑布模型

瀑布模型是在1970年由W. Royce最早提出的软件开发模型，如图1-3所示。它将软件生存周期的各项活动规定为依固定顺序连接的若干阶段工作，这些工作之间的衔接关系是从上到下、不可逆转的，如同瀑布一样，因此称为瀑布模型。

每项开发活动均应具有下述特征：

1）以上一项活动产生的工作对象作为输入。

2）利用这一输入，实施本项活动应完成的内容。

3）给出该项活动的工作结果，作为输出传给下一项活动。

4）对实施该项活动的工作结果进行评审。若其工作得到确认，则继续进行下一项活动，否则返回前项，甚至更前项的活动进行返工。

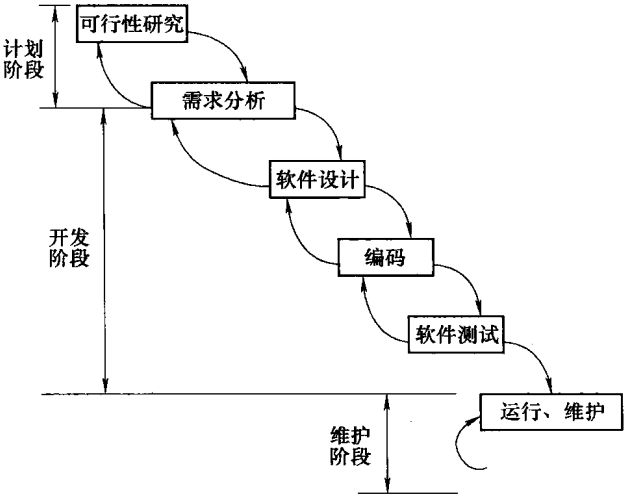


图1-3 瀑布模型

瀑布模型自提出以来，一直是一种广泛采用的开发模型，但是，在长期的实践中，人们发现这种模型有如下一些缺点。

1）在项目开始阶段，开发人员和用户对需求的描述常常是不全面的。如果需求阶段未

发现这些问题，就会影响到后面各阶段的工作。

2）瀑布模型中各阶段所做的工作都是文档说明，而对一般用户来说，很难全面理解文字描述背后的软件产品。当用户在提出一些意见时，加大了系统修改难度。

3）影响整个软件开发角度。例如在开发过程中，事先选择的技术或需求迅速发生变化，需要返回到前面某个阶段，对前面的一系列内容进行修改。

总的来说，瀑布模型是一种应付需求变化能力较弱的开发模型，因此，很多在该模型基础上开发出来的软件产品不能够真正满足用户需求。

1.4.2 快速原型模型

快速原型模型的第一步是建造一个快速原型，实现客户或未来的用户与系统之间的交互，用户或客户可以通过对原型的评价，进一步细化待开发软件的需求，由此通过逐步调整原型而进一步满足客户的要求，开发人员也可以确定客户的真正需求是什么；第二步则在第一步的基础上开发客户满意的软件产品。快速原型模型如图 1-4 所示。

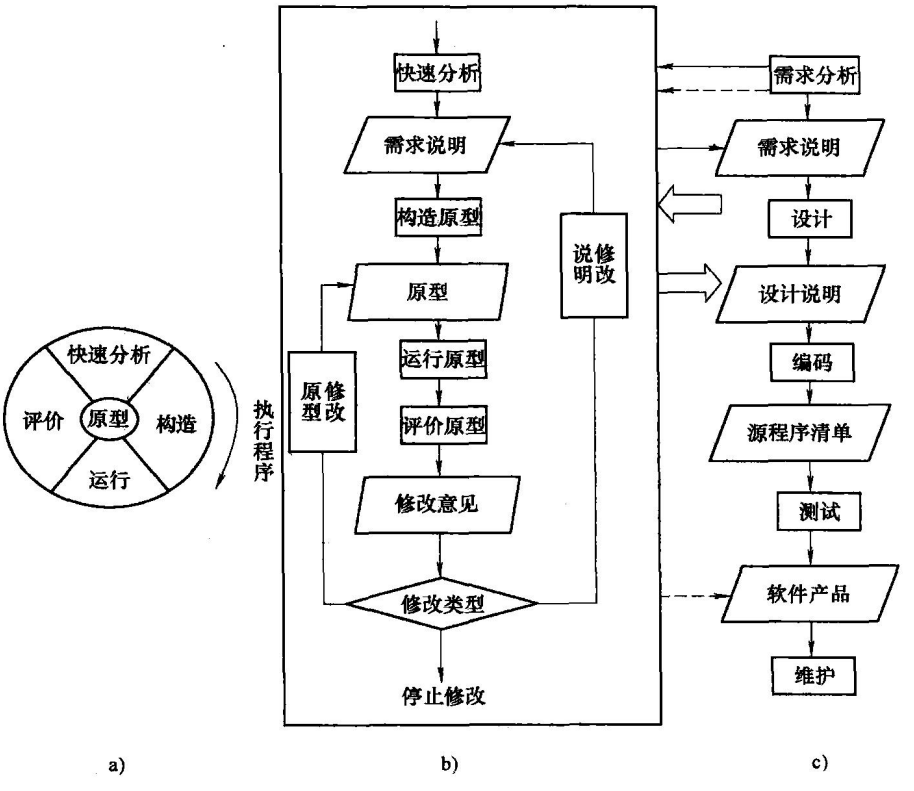


图 1-4 快速原型模型

a）原型本身的表示 b）原型本身的使用过程 c）快速原型模型的开发过程

显然，快速原型模型可以克服瀑布模型的缺点，减少由于软件需求不明确带来的开发风险，事实证明具有显著的效果。

快速原型模型的关键在于尽可能快速地建造出软件原型，一旦确定了客户的真正需求，所建造的原型将被丢弃。因此，原型系统的内部结构并不重要，重要的是必须迅速建立原

型，随之迅速修改原型，以反映客户的需求。

1.4.3 螺旋模型

1988 年，Barry Boehm 正式发表了软件系统开发的螺旋模型，它将瀑布模型和快速原型模型结合起来，强调了其他模型所忽视的风险分析，特别适合于大型复杂的系统。

风险是普遍存在于软件开发项目中的，所不同的只是风险有大有小。实践表明，项目规模越大，问题越复杂，资源、成本、进度等因素的不确定性就越大，承担的风险也越大。总之，风险是软件开发不可忽视的潜在不利因素，它可能在不同程度上损害到软件开发过程或软件产品的质量。软件风险驾驭的目标是在造成危害之前对风险进行识别、分析、采取对策，进而消除或减少风险的损害。螺旋模型沿着螺线进行若干次迭代，图 1-5 中的四个象限代表了以下活动：

1）制定计划。确定软件目标，选定实施方案，弄清项目开发的限制条件。

2）风险分析。分析评估所选方案，考虑如何识别和消除风险。

3）实施工程。实施软件开发和验证。

4）客户评估。评价开发工作，提出修正建议，制定下一步计划。

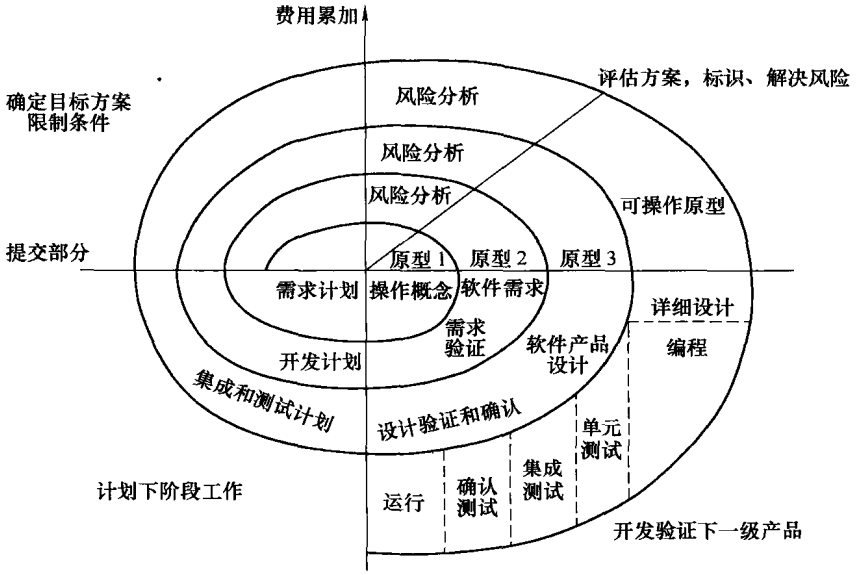


图 1-5 螺旋模型

从图 1-5 中可以看出，沿着螺旋线每转一圈，表示开发出一个更完善的软件版本。如果开发风险过大，开发机构和客户无法接受，项目有可能就此终止，多数情况下，会沿着螺旋线继续下去并向外逐步延伸，最终能够得到所期望的系统。但是，螺旋模型也有一定的限制条件，具体如下：

1）螺旋模型强调风险分析，但要求许多客户接受和相信这种分析，并做出相关反应是不容易的，因此，这种模型往往适应于大规模软件开发。

2）如果执行风险分析会大大影响项目的利润，那么进行风险分析就毫无意义，因此，螺旋模型只适合于大规模软件项目。

3）软件开发人员应该擅长寻找可能的风险，准确地分析风险，否则将会带来更大的风险。

1.4.4 喷泉模型

喷泉模型是一种以用户需求为动力，以对象为驱动的模型，主要用于描述面向对象的软件开发过程。该模型认为软件生存周期的各阶段是相互重叠和多次反复的，就像水喷上去又可以落下来，可以落在中间，也可以落在最底部，类似一个喷泉。各个开发阶段没有特定的次序要求，并且可以交互进行，可以在某个开发阶段中随时补充其他任何开发阶段中的遗漏。在喷泉模型中，存在交叠的活动用重叠的圆圈表示，一个阶段内向下的箭头表示阶段内的迭代求精。喷泉模型用较小的圆圈代表维护，圆圈较小象征采用面向对象设计后维护时间缩短了。喷泉模型如图 1-6 所示。

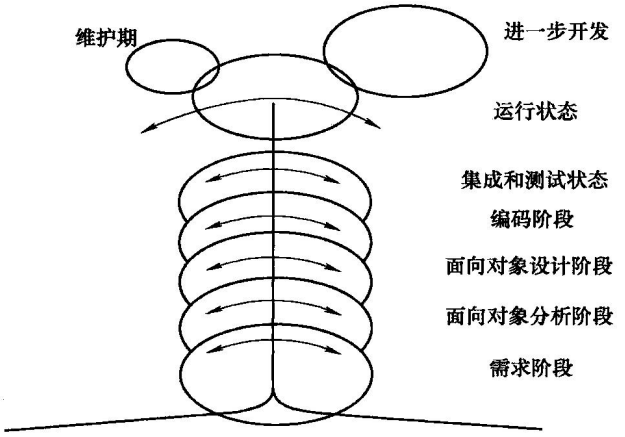


图 1-6 喷泉模型

1.5 软件开发方法简述

1.5.1 面向数据流的结构化方法

面向数据流的结构化方法，是由 E. Yourdon 和 L. L. Constantine 提出的，是 20 世纪 80 年代使用最广泛的软件开发方法。该方法建立在软件生存周期模型基础上，采用结构化分析方法对软件进行分析，然后用结构化设计方法进行概要设计和详细设计，最后进行结构化编程。

结构化分析以分析信息流为主，用数据流图来表示信息流，按照功能分解的原则，自顶向下逐步求精，直到实现软件功能为止。在分析问题时，一般利用图表方式进行描述。使用的工具有：数据流图、数据字典、问题描述语言、判定表和判定树等。其中数据流图用来描述系统中数据的处理过程，可以是一个程序、一个模块或一系列程序，还可以是某个人工处理过程；数据字典用来查阅数据的定义；问题描述语言、判定表和判定树用来详细描述数据处理的细节问题。

结构化设计是以结构化分析为基础，将分析得到的数据流图转换为描述系统模块之间关

系的结构图。

面向数据流的结构化方法主要问题是构造的系统不够稳定，它以功能分解为基础，而用户的功能是经常改变的，必然导致系统的框架结构不稳定。另外，从数据流图到软件结构图之间的过渡有明显的断层，导致设计回溯到需求有一定困难，但由于方法简单、实用，至今仍使用。

1.5.2 面向数据结构的 Jackson 方法

面向数据结构方法是根据数据结构设计程序处理过程的方法，侧重数据结构而不是数据流。在许多应用领域中信息都有清楚的层次结构，输入信息、信息的内部存储、输出信息也都可能有一定的数据结构，而数据结构与程序结构紧密相关。有一个著名的公式："程序 = 算法 + 数据结构"。可见，在程序设计中算法和数据结构是紧密相连的，不同的数据结构往往决定了不同的算法结构。面向数据结构方法，着重于问题数据结构到问题解的程序结构之间的转换，而不强调模块定义。因此，该方法首先要充分了解所涉及的数据结构，而且用工具清晰地描述数据结构，然后按一定的步骤根据数据结构，导出解决问题的程序结构，完成设计。

1975 年 M. A. Jackson 提出了一类至今仍广泛使用的软件开发方法，称为 Jackson 方法。该方法把每个问题分解为由三种基本结构相互组合而成的层次结构图。三种基本的结构形式是顺序、选择和重复。采用这一方法，从目标系统的输入、输出数据结构入手，导出程序框架结构，再补充其他细节，就可得到完整的程序结构图。这一方法对输入、输出数据结构明确的中小型系统特别有效，如商业应用中的文件表格处理。详细参见本书 5.3 节。

1.5.3 面向对象的方法

人们对于面向对象方法的研究最早起源于面向对象编程语言，在 20 世纪 60 年代开发的 Simula 语言，提供了比子程序更高的抽象机制。20 世纪 70 年代初期开发出 Smalltalk 语言，它引用了 Simula 语言中关于类的概念，应用了继承机制和动态连接，同时，它第一次提出了"面向对象"这一术语，之后面向对象语言不断发展。目前，面向对象语言已经成为应用最广泛的程序设计语言。与此同时，人们对面向对象的研究从编程语言开始向软件生存周期的前期阶段发展。也就是说，人们对面向对象方法的研究与运用，不再局限于用于系统实现的编程语言，而是从系统分析和系统设计阶段就开始采用面向对象方法。这标志着面向对象已经逐步发展成一种完整的方法论。

20 世纪 90 年代以来，一些专家按照面向对象的思想，对面向对象的分析和设计（OOA/OOD）工作的步骤、方法、图形工具等进行了详细研究，提出了多种实施方案。据不完全统计有五十几种，其中比较流行的有十几种。其中影响较大的方法如下。

（1）Booch 方法

Grady Booch 是面向对象方法最早的倡导者之一，他提出了面向对象软件工程的概念，并发明了 Booch 方法，该方法的分析能力较弱，是一种偏重设计的方法。

（2）OMT 方法

OMT 方法即面向对象的建模技术（Object-Oriented Modeling Technique），是由 Rumbaugh