高职高专计算机系列教材

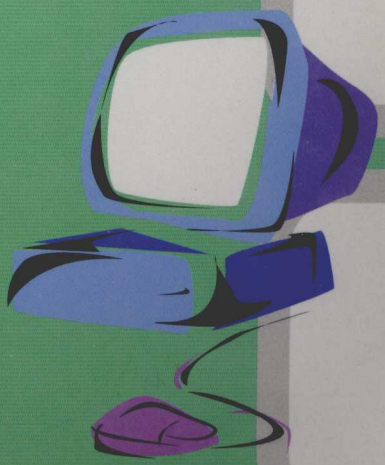
JISUANJI

软件工程

Ruanjian Gongcheng

主　编　罗先文

副主编　张颖淳



重庆大学出版社

软件工程

主　编　罗先文
副主编　张颖淳

重庆大学出版社

内 容 提 要

软件工程是20世纪60年代发展起来的一门新学科，它主要介绍软件开发技术思想和方法。本书作者在总结教学经验的基础上，编写了这本教材。其主要内容包括可行性分析、软件需求分析、概要设计、详细设计、程序编码、面向对象的分析与设计方法、软件质量、软件测试技术、软件维护、软件项目管理以及软件工程标准与软件文档等内容，并适当介绍了软件工程的发展问题。掌握软件工程知识，有助于读者在软件工程项目的开发上使用工程化标准。全书内容通俗易懂，实用性强，对软件工程的常用方法介绍，突出可操作性，内容精练，重点突出，概念清楚，针对性和实际操作性强。

本书主要作为高职高专计算机及相关专业学生的教材，也可作为软件开发人员的参考书和各类培训班教材。

图书在版编目(CIP)数据

软件工程/罗先文，张颖淳主编. —重庆：重庆大学出版社，2004.4
(高职高专计算机系列教材)
ISBN 7-5624-3064-0

Ⅰ.软… Ⅱ.①罗…②张… Ⅲ.软件工程—高等学校：技术学校—教材 Ⅳ.TP311.5

中国版本图书馆CIP数据核字(2004)第002200号

软 件 工 程

主　编　罗先文
副主编　张颖淳
责任编辑：曾令维　何建云　　版式设计：曾令维
责任校对：蓝安梅　　责任印制：张立全

*

重庆大学出版社出版发行
出版人：张鸽盛
社址：重庆市沙坪坝正街174号重庆大学(A区)内
邮编：400030
电话：(023) 65102378　65105781
传真：(023) 65103686　65105565
网址：http://www.cqup.com.cn
邮箱：fxk@cqup.com.cn (市场营销部)
全国新华书店经销
重庆升光电力印务有限公司印刷

*

开本：787×1092　1/16　　印张：13　字数：324千
2004年4月第1版　　2004年4月第1次印刷
印数：1—5 000
ISBN 7-5624-3064-0/TP·458　　定价：18.00元

前言

在20世纪60年代计算机发展初期，一群程序员、计算机科学家与工业界人士聚集一起共商对策。通过借鉴传统工业的成功做法，他们主张通过工程化的方法开发软件来解决软件危机，并冠以“软件工程”这一术语。软件工程这门学科从此诞生。

软件工程是一门实践性强的学科，它的形成得益于软件工程项目的推动，或者人们为解决“软件危机”而寻求的解决方法，同时，该学科也是软件开发项目的实际指导。尽管软件的一些毛病如人类的感冒一样无法根治，但软件的发展速度超过了任何传统工业，期间并未出现真正的“软件危机”，这主要得益于软件工程。

软件工程主要讲述软件开发的道理，基本上是软件实践者的成功经验和失败教训的总结。软件工程的观念、方法、策略和规范都是朴实无华的，平凡之人皆可领会，关键在于运用。

本书依据教育部关于《高职高专教育基础课程教学的基本要求》和《高职高专教育专业人才培养目标与规格》编写而成。全书共13章。内容主要包括：软件工程概述，可行性分析，软件需求分析，概要设计，详细设计，程序编码，面向对象的分析与设计方法，软件质量，软件测试技术，软件维护，软件项目管理，软件工程标准与软件文档，软件工程环境与发展等。全书内容简洁，便于教学与自学。

本书强调内容通俗易懂，实用性强，对软件工程的常用方法介绍，突出可操作性。内容精练，重点突出，概念清楚，针对性和实际操作性强，主要作为高职高专计算机及相关专业学生的教材，也可作为软件开发人员的参考书和各类培训班教材。

全书共分13章，分别由罗先文编写第7,11,12章，张颖淳编写第2,3章，蒋承延编写1,9章，马慧编写第4,5章，丁华锋编写第6,8章，黄煜编写第10,13章。全书由罗先文统稿，同时也得到重庆大学出版社的大力支持。特别是2000级的李杰，莫灼宇同学，在全书整理过程中做了大量工作，在此一并表示感谢。

编者

2004年2月

目录

第 1 章
软件工程概述

1.1 软件的概念、特点和分类

1.1.1 软件的概念

软件是计算机系统中与硬件相互依存的另一部分，是包括程序、数据及其相关文档的完整集合。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、维护和使用有关的图文材料。

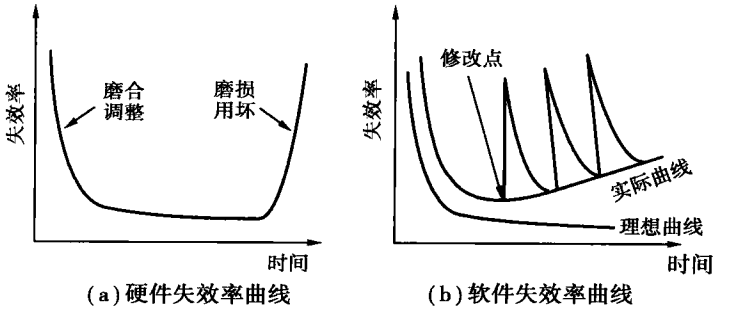
1.1.2 软件的特点

为了全面地了解软件开发的过程，必须先了解软件所具有的特点：

1）软件是一种逻辑实体，而不是具体的物理实体，因而它具有抽象性。

2）软件的生产与硬件不同，它没有明显的制造过程。对软件的质量控制，必须着重在软件开发方面下功夫。

3）在软件的运行和使用期间，没有硬件那样的机械磨损和老化问题。任何机械、电子设备在运行和使用中，其失效率大都遵循如图 1.1（a）所示的 U 型曲线（即浴盆曲线）。而软件



（a）硬件失效率曲线　　（b）软件失效率曲线

图 1.1　失效率曲线

的情况与此不同，因为它不存在磨损和老化问题。然而它存在退化问题，必须要多次修改（维护）软件，如图 1.1（b）所示。

4）软件的开发和运行常常受到计算机系统的限制，对计算机系统有着不同程度的依赖性。为了解决这种依赖性，在软件开发中提出了软件移植的问题。

5）软件的开发至今尚未完全摆脱手工工艺的开发方式。

6）软件本身是复杂的。软件的复杂性可能来自它所反映的实际问题的复杂性，也可能来自程序逻辑结构的复杂性。

7）软件成本相当昂贵。软件的研制工作需要投入大量的、复杂的、高强度的脑力劳动，它的成本是比较高的。

8）相当多的软件工作涉及到社会因素。许多软件的开发和运行涉及机构、体制及管理方式等问题，甚至涉及人的观念和人的心理。它直接影响到项目的成败。

1.1.3 软件的分类

（1）按软件的功能进行划分

• 系统软件：是指能与计算机硬件紧密配合在一起，使计算机系统各个部件、相关的软件和数据协调、高效地工作的软件。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。

• 支撑软件：是协助用户开发软件的工具性软件，其中包括帮助程序人员开发软件产品的工具，也包括帮助管理人员控制开发进程的工具。

• 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。

（2）按软件规模进行划分

按开发软件所需的人力、时间以及完成的源程序行数，可确定 6 种不同规模的软件，如表 1.1 所示。

表 1.1 软件规模的分类

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	0.5 KB
小型	1	1～6 月	1 KB～2 KB
中型	2～5	1～2 年	5 KB～50 KB
大型	5～20	2～3 年	50 KB～100 KB
甚大型	100～1 000	4～5 年	1 MB（1 024 KB）
极大型	2 000～5 000	5～10 年	1 MB～10 MB

规模大、时间长、很多人参加的软件项目，其开发工作必须要有软件工程的知识做指导。而规模小、时间短、参加人员少的软件项目也得有软件工程概念，遵循一定的开发规范。其基本原则是一样的，只是对软件工程技术依赖的程度不同而已。

（3）按软件工作方式划分

• 实时处理软件：指在事件或数据产生时，立即予以处理，并及时反馈信号，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。

• 分时软件：允许多个联机用户同时使用计算机。

• 交互式软件：能实现人机通信的软件。

• 批处理软件：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。

(4) 按软件服务对象的范围划分

• 项目软件：也称定制软件，是受某个特定客户(或少数客户)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。如军用防空指挥系统、卫星控制系统。

• 产品软件：是由软件开发机构开发出来直接提供给市场，或是为千百个用户服务的软件。例如，文字处理软件、文本处理软件、财务处理软件、人事管理软件等。

(5) 按使用的频度进行划分

有的软件开发出来仅供一次使用，如用于人口普查、工业普查的软件等；另外有些软件具有较高的使用频度，如天气预报软件等。

(6) 按软件失效的影响进行划分

有的软件在工作中出现了故障，造成软件失效，可能给软件整个系统带来的影响不大；有的软件一旦失效，可能酿成灾难性后果，如财务金融、交通通信、航空航天等系统的软件，我们称这类软件为关键软件。

1.1.4　软件的发展

自 20 世纪 40 年代出现了世界上第一台计算机以来，就有了程序的概念。经过几十年的发展，计算机软件历经了 3 个主要阶段：

• 程序设计阶段，约为 20 世纪 50 ~ 60 年代。

• 程序系统阶段，约为 20 世纪 60 ~ 70 年代。

• 软件工程阶段，约为 20 世纪 70 年代至今。

几十年来最根本的变化体现在：

1) 人们改变了对软件的看法。20 世纪 50 ~ 60 年代，程序设计曾经被看做是一种任人发挥、创造才能的技术领域。当时人们认为，写出的程序只要能在计算机上得出正确的结果，程序的写法可以不受任何约束。随着计算机的广泛使用，人们要求这些程序容易看懂、容易使用，并且容易修改和扩充。于是，程序便从个人按自己意图创造的“艺术品”转变为能被广大用户接受的工程化产品。

2) 软件的需求是软件发展的动力。早期的程序开发者只是为了满足自己的需要，这种自给自足的生产方式仍然是其低级阶段的表现。进入软件工程阶段以后，软件开发的成果具有社会属性，它要在市场中流通以满足广大用户的需要。

3) 软件工作的范围从只考虑程序的编写扩展到涉及整个软件生存周期。

1.1.5　软件危机

在软件技术发展的第二阶段，随着计算机硬件技术的进步，要求软件能与之相适应。然而软件技术的进步一直未能满足形势发展提出的要求。致使问题积累起来，形成了日益尖锐的矛盾，这就导致了软件危机。问题归结起来有：

1) 由于缺乏软件开发的经验和有关软件开发数据的积累，使得开发工作的计划很难制

定,经费预算常常突破,进度计划无法遵循,开发完成的期限一拖再拖。

2)在开发的初期阶段对软件需求提得不够明确,或是未能得到确切的表达。开发工作开始后,软件人员和用户又未能及时交换意见,造成开发后期矛盾的集中暴露。

3)开发过程没有统一的、公认的方法论和规范指导,参加的人员各行其是。加之设计和实现过程的资料很不完整,或忽视了每个人工作与其他人的接口,使得软件很难维护。

4)未能在测试阶段充分做好检测工作,提交用户的软件质量差,在运行中暴露出大量的问题。

如果这些障碍不能突破,进而摆脱困境,软件的发展是没有出路的。

1.2 软件工程的概念

1.2.1 软件工程的定义

Fritz Bauer 曾经为软件工程下过定义:"软件工程是为了经济地获得能够在实际机器上有效运行的可靠软件而建立和使用的一系列完善的工程化原则。"

1983 年 IEEE 给出的定义为:"软件工程是开发、运行、维护和修复软件的系统方法。"其中,"软件"的定义为:"计算机程序、方法、规则、相关的文档资料以及在计算机上运行时所必需的数据。"

后来尽管又有一些人提出了许多更为完善的定义,但主要思想都是强调在软件开发过程中需要应用工程化原则的重要性。

软件工程包括 3 个要素:方法、工具和过程。

软件工程方法为软件开发提供了"如何做"的技术。它包括了多方面的任务,如项目计划与估算、软件系统需求分析、数据结构、系统总体结构的设计、算法过程的设计、编码、测试以及维护等。

软件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。目前,已经推出了许多软件工具,这些软件工具集成起来,建立起称之为计算机辅助软件工程(Computer-aided software engineering,简称 CASE)的软件开发支撑系统。CASE 将各种软件工具、开发机器和一个存放开发过程信息的工程数据库组合起来形成一个软件工程环境。

软件工程的过程则是将软件工程的方法和工具综合起来以达到合理、及时地进行计算机软件开发的目的。过程定义了方法使用的顺序、要求交付的文档资料、为保证质量和协调变化所需要的管理以及软件开发各个阶段完成的里程碑。

1.2.2 软件工程的目标

组织实施软件工程项目,最终希望得到项目的成功。所谓成功指的是达到以下几个主要的目标:

- 付出较低的开发成本;
- 达到要求的软件功能;
- 取得较好的软件性能;

- 开发的软件易于移植；
- 需要较低的维护费用；
- 能按时完成开发工作，及时交付使用。

在具体项目的实际开发中，企图让以上几个目标都达到理想的程度往往是非常困难的。

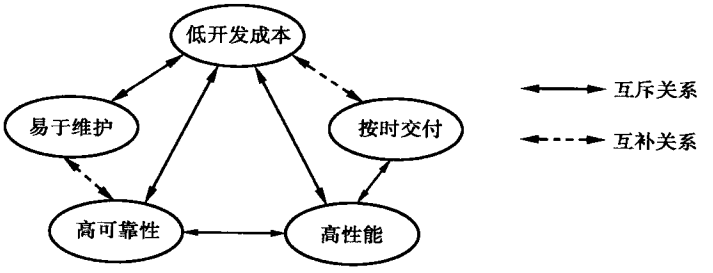


图 1.2　软件工程目标之间的关系

图 1.2 表明了软件工程目标之间存在的相互关系。其中有些目标之间是互补关系，例如，易于维护和高可靠性之间，低开发成本与按时交付之间。还有一些目标是彼此互斥的，例如，低开发成本与高可靠性之间，提高软件性能与软件可移植性之间，就存在冲突。

1.2.3　软件工程的原则

以上的软件工程基本目标适合于所有的软件工程项目。为达到这些目标，在软件开发过程中必须遵循下列软件工程原则。

- 抽象：抽取事物最基本的特性和行为，忽略非基本的细节。采用分层次抽象，自顶向下、逐层细化的办法控制软件开发过程的复杂性。
- 信息隐蔽：将模块设计成“黑箱”，实现的细节隐藏在模块内部，不让模块的使用者直接访问。这就是信息封装，使用与实现分离的原则。使用者只能通过模块接口访问模块中封装的数据。
- 模块化：模块是程序中逻辑上相对独立的成分，是独立的编程单位，应有良好的接口定义。如 C 语言程序中的函数过程，C ++ 语言程序中的类。模块化有助于信息隐蔽和抽象，有助于表示复杂的系统。
- 局部化：要求在一个物理模块内集中逻辑上相互关联的计算机资源，保证模块之间具有松散的耦合，模块内部具有较强的内聚。这有助于控制解的复杂性。
- 确定性：软件开发过程中所有概念的表达应是确定的、无歧义的、规范的。这有助于人们在交流时不会产生误解、遗漏，保证整个开发工作协调一致。
- 一致性：整个软件系统（包括程序、文档和数据）的各个模块应使用一致的概念、符号和术语。程序内部接口应保持一致，软件和硬件、操作系统的接口应保持一致，系统规格说明与系统行为应保持一致，用于形式化规格说明的公理系统应保持一致。
- 完备性：软件系统不丢失任何重要成分，可以完全达到系统所要求功能的程度。为了保证系统的完备性，在软件开发和运行过程中需要严格的技术评审。
- 可验证性：开发大型的软件系统需要对系统自顶向下、逐层分解。系统分解应遵循系统易于检查、测试、评审的原则，以确保系统的正确性。

使用一致性、完备性和可验证性的原则可以帮助人们实现一个正确的系统。

1.3 软件生存周期

如同其他任何事物一样，软件也有一个孕育、诞生、成长、成熟、衰亡的生存过程，我们称其为计算机软件的生存周期。软件生存周期模型是从软件项目需求定义直至软件经使用后废弃为止，跨越整个生存周期的系统开发、运作和维护所实施的全部过程、活动和任务的结构框架。根据这一思想，把上述基本的过程活动进一步展开，可以得到软件生存周期的7个阶段：制定计划、需求分析、软件设计、程序编写、软件测试、运行/维护、退役。其中软件设计可细分为概要设计和详细设计，软件测试则可分为组装测试和确认测试。

1.3.1 软件定义

软件定义可分为软件系统的可行性研究和需求分析两个阶段。

(1)软件系统的可行性研究

可行性研究的目的是了解客户的要求和现实环境，从技术、经济和社会等多方面进行调研、论证软件的可行性，而该过程需要在客户的参与配合下完成。其中，可行性论证包括技术可行性、操作可行性和经济可行性3部分。在对软件系统进行调研和可行性论证的基础上还应制定初步的项目计划，包括选用资源、定义任务、风险分析、成本估算、成本效益分析和进度安排等。

(2)需求分析

1)任务。确定待开发软件的功能需求、性能需求和运行环境需求，编制软件规格说明、软件系统的确认测试准则和客户手册概要。

2)重要性与困难。软件需求不仅是软件开发的依据，也是软件验收的标准。因此，确定软件需求是软件开发的关键和难点，其具体表现为：

- 客户、系统分析员和软件开发人员所重视软件需求的角度不同。
- 客户自身也不能确定其需要的软件系统应具备的功能。
- 客户、系统分析员和软件开发人员对软件需求的描述方式的意见不同。

3)需求分析过程。由于客户缺乏软件开发的知识和经验，系统分析员和软件开发人员不得不与客户对客户所提出的系统需求进行反复磋商，才能最终达成一致。在此过程中，就必须建立软件需求文档，有时还要对大型、复杂的软件系统的主要功能、接口、人机界面等进行模拟或建造原型。

4)软件需求规格说明(software requirements specification，简称SRS)。SRS需要指明软件系统的运行环境需求、功能需求、性能需求、接口需求、系统结构以及开发标准和验收标准等。

1.3.2 软件工程过程

许多计算机和软件科学家曾尝试，把其他工程领域中行之有效的工程学知识运用到软件开发工作中来。经过不断实践和总结，最后得出一个结论：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。

软件工程过程(software engineering process)是为获得软件产品,在软件工具支持下由软件工程师完成的一系列软件工程活动。软件工程过程通常包含 4 种基本的过程活动:

- P (plan): 软件规格说明,规定软件的功能及其运行的限制;
- D (do): 软件开发,产生满足规格说明的软件;
- C (check): 软件确认,确认软件能够完成客户提出的要求;
- A (action): 软件演进,为满足客户的变更要求,软件必须在使用的过程中演进。

事实上,软件工程过程是一个软件开发机构针对某一类软件产品为自己规定的工作步骤,它应当是科学的、合理的,否则必将影响到软件产品的质量。

1.3.3　软件使用、维护和退役

1)软件使用。软件使用是软件产生经济和社会效益的重要阶段。由于软件是具有逻辑性的产品,软件发行的分量越大,则所取得的经济和社会效益就越显著。而软件在市场中推广使用时,维护人员和客户都应该注意对软件在使用过程中发生错误的信息的收集。

2)软件维护。软件维护就是在软件使用过程中,发现了软件产品中的潜伏错误,或是客户提出要对软件需求进行修改,或是软件运行环境发生变化时,对软件代码、软件定义、开发各阶段生成的相关文档进行修改、完善的过程。

3)退役。软件生存周期的最后阶段,意味着该软件"生命"的终止。

1.4　软件开发模型

1.4.1　瀑布模型

瀑布模型规定了各项软件工程活动,包括:制定开发计划、进行需求分析和说明、软件设计、程序编码、测试及运行维护,参看图 1.3。并且规定了它们自上而下、相互衔接的固定次序,如同瀑布流水,逐级下落。

然而软件开发的实践表明,上述各项活动之间并非完全是自上而下,呈线性图式。实际情况是,每项开发活动均处于一个质量环(输入—处理—输出—评审)中。只有当其工作得到确认,才能进入下一项活动,在图 1.3 中用向下的箭头表示;否则需要返工,在图 1.3 中用向上的箭头表示。

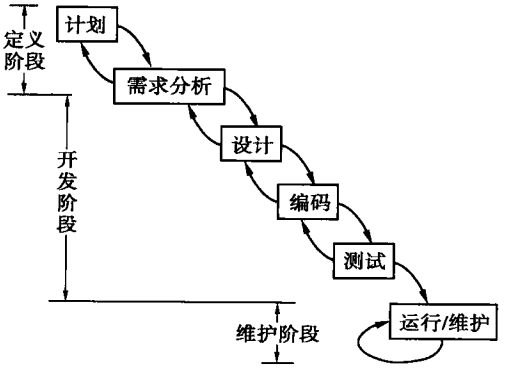


图 1.3　瀑布开发模型

1.4.2　原型模型

由于在软件开发初期确定软件系统需求存在困难,软件开发人员便借鉴建筑师在设计建筑原型方面的经验。首先根据客户提出的软件定义,快速地开发出一个原型,它向客户展示了待开发软件系统的全部或部分功能和性能,然后根据客户对原型的意见,对原型进行修改、完

善,确认软件系统的需求并达到一致的理解。快速开发原型的途径有3种:

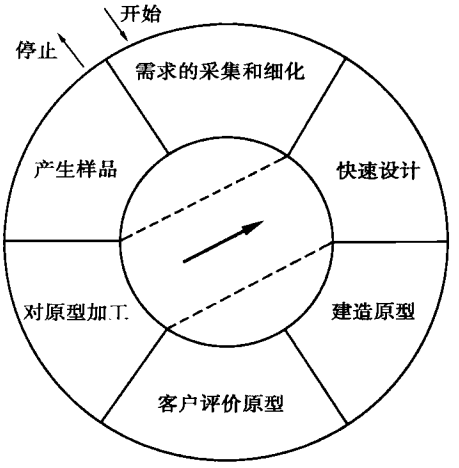


图1.4 原型开发模型

1)利用个人计算机模拟软件系统的人-机界面和人-机交互方式。

2)开发一个工作模型,实现软件系统的部分功能,而这部分功能是重要的,也可能是容易产生误解的。

3)利用一个或几个正在运行的类似软件向客户展示软件需求中的全部或部分功能。为了快速开发出原型,要尽量采用软件重用技术,在算法的时/空开销方面也可让步,以便尽快向客户提供原型。

原型开发模型如图1.4所示。

1.4.3 螺旋模型

对于复杂的大型软件,开发一个原型往往达不到要求。螺旋模型将瀑布模型与原型模型结合起来,并且加入这两种模型均忽略了的风险分析。螺旋模型沿着螺线旋转,如图1.5所示,在笛卡尔坐标的四个象限上分别表达了四个方面的活动,即:

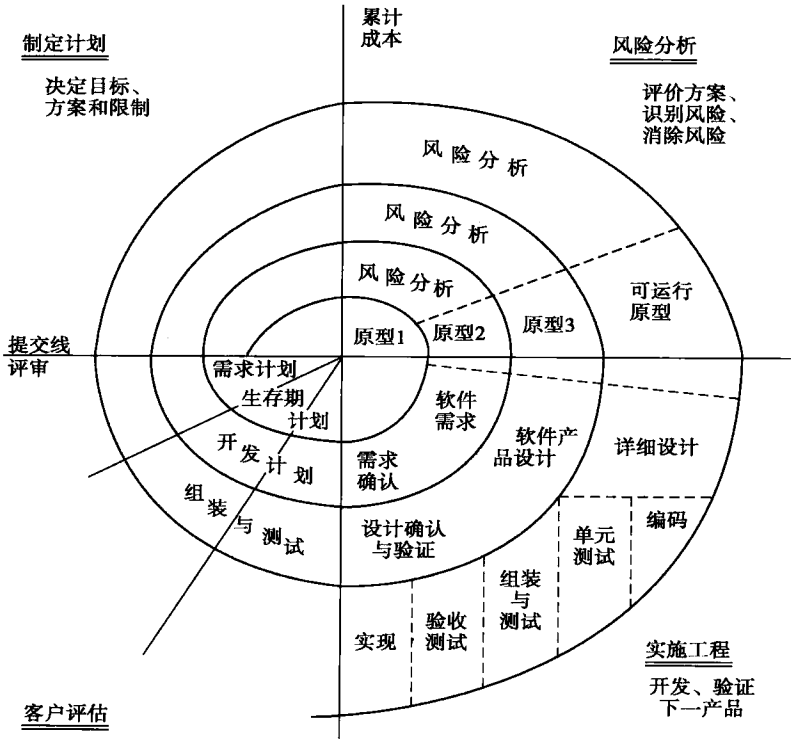


图1.5 螺旋模型

- 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;