Chapter
01

# 项目一

# 认识软件工程

## 项目要点

- 软件危机与软件工程
- 软件生存周期和开发模型
- 软件开发方法和软件开发工具
- 传统软件工程和面向对象软件工程

## 引言

本项目主要讲述软件的发展、定义及特征，软件危机的表现及解决途径，软件工程的定义及三要素，软件工程的基本原理及目标，软件生存周期的概念和内容，5 种软件的开发模型，软件开发方法和开发工具，传统软件工程和面向对象软件工程。

# 任务一:软件工程基本知识培训

## 任务描述

王一明是某科技公司的软件开发培训人员,需要对公司新招的一批应届毕业大学生进行软件工程培训,使他们了解软件工程的基本知识,为后续培训和实际工作的开展打下基础。

## 任务分析

软件经过长期发展而成,其中间也经历了危机与变革。随着软件开发技术的日渐成熟,也形成了具有一定规格的体系结构。此时,人们相继定义出软件工程的概念和基本原理。想要经济地开发出高质量的软件并进行有效维护,软件工程知识必不可少。

## 准备知识

计算机软件的发展与进步,是与计算机硬件的发展和计算机的普及分不开的。同其他事物的发展规律一样,也经历了从产生、发展到成熟的过程。在这个过程中也经历了软件的危机,为吸取历史经验教训,应该认真研究产生软件危机的原因,探讨消除软件危机的途径。

在计算机发展的初期,硬件的设计和生产是主要问题,那时的软件是机器指令程序,它们处于从属的地位。软件的生产方式是个体手工方式,设计过程是在一个人的头脑中完成的,程序的质量完全取决于个人的编程技巧。

随着计算机技术的发展,人们认识到在计算机上增加软件的功能会使计算机系统的功能得到大大提高,因此在编制大型程序系统时,不但要考虑硬件设备的配置,还要考虑与之相关的软件的功能,这样才能增强整个程序系统的功效。而且随着软件系统规模的逐渐扩大,软件不再是一个人编制完成,而需要多人合作。这时软件的生产方式是互助合作的手工方式,由于有多人合作编程,为了互相读懂程序,就需要增加说明书,所以这个时期软件的含义是“程序+说明书”。

随着软件规模的增大,它的复杂度也在增加。软件可靠性往往随规模的增长而下降,质量保证也越来越困难,软件的发展速度远不能满足用户的需求,出现了软件危机。人们感到传统的软件生产方式已不再适应计算机发展的需要。因此,提出把工程学的基本原理和方法引入到软件生产中,像做传统工程那样,把软件生产分成几个阶段,每个阶段都有严格的管理和质量检验,开始研制软件设计和生产的方法及工具,并用书面文件作为共同遵循的依据,而且大型系统的数据需求量也非常大,所以需要用专门的数据结构来表示和处理。这时软件的含义就成为“程序+数据+文档”。

现在对软件的一种公认的解释为：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，程序是按一定的功能和性能要求而设计并执行的指令序列；数据为进行通信、解释和处理而使用的信息的形式化表现形式。文档是与程序开发、维护和使用有关的图文材料。

**知识链接**

为了能全面、正确地理解计算机和软件，必须了解软件的特点。与硬件相比，软件主要有以下特点：

(1)软件是一种逻辑实体，不是具体的物理实体；

(2)软件与硬件的生产方式不同；

(3)软件与硬件的维护不同；

(4)软件是复杂的；

(5)软件成本相当昂贵。

## 任务实施

根据实际工作需要，王一明为应届毕业生准备了以下培训内容，以期让他们对软件工程有一个最基本的认识，方便他们后续培训及实际工作的开展。

### 1. 软件工程的概念

1968 年，北大西洋公约组织在原联邦德国召开计算机科学会议，在此次会议上 Fritz Bauer 首次正式提出了“软件工程”的概念。会议讨论要建立并使用正确的工程方法开发出成本低、可靠性高并能高效运转的软件，从而解决或缓解软件危机。

软件工程学是一门指导软件开发和维护的工程学科，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。

**知识链接**

软件工程学应用计算机科学、数学及管理科学等学科的原理，并借鉴传统工程的原则、方法来生产软件，以达到提高软件质量、降低开发成本的目的。

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，是指导研制软件的某种标准规范。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、算法的设计、编码、测试以及维护等。软件工程方法具有自己独特的语言或图形表达方式及一套质量保证标准。

软件工具是指软件开发、维护和分析中使用的程序系统，为软件工程方法提供自动的或半自动的软件支撑环境。

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及软件开发各个阶段完成的“里程碑”。

## 2. 软件工程的基本原理

著名的软件工程专家 B. W. Boehm 于 1983 年综合了软件工程专家学者们的意见，并总结了开发软件的经验，提出了软件工程的 7 条基本原理。这 7 条原理组成的集合被认为是确保软件产品质量和开发效率的最小准则集合，它们相互独立、缺一不可、相当完备。下面简单介绍软件工程的这 7 条基本原理。

### (1)用分阶段的生存周期计划严格管理

根据这条基本原理，可以把软件生存周期划分成若干个阶段，并相应地制定出切实可行的计划，然后严格按照计划对软件开发与维护进行管理。

**拓展提高**

需要制订的计划项目有概要计划、里程碑计划、项目控制计划、产品控制计划、验证计划和运行维护计划等。各级管理人员必须严格按照计划对软件开发和维护工作进行管理。

### (2)坚持进行阶段评审

据统计，在软件生存周期各阶段中，编码阶段之前的错误约占 63%，而编码错误仅约占 37%。另外，错误发现并改正得越晚，所付出的代价越高。坚持在每个阶段结束前进行严格的评审，就可以尽早发现错误。因此，这是一条必须坚持的重要原理。

### (3)实行严格的产品控制

由于外部环境的变化，在软件开发的过程中改变需求是难免的，但决不能随意改变需求，只能依靠科学的产品控制技术来顺应用户提出的改变需求的要求。为了保持软件各个配置成分的一致性，必须实行严格的产品控制。

### (4)采用现代程序设计技术

实践表明，采用先进的程序设计技术既可以提高软件开发与维护的效率，又可以提高软件的质量。多年来，人们一直致力于研究新的“程序设计技术”。例如，20 世纪 60 年代末提出的结构化程序设计技术，以及后来提出的面向对象的分析(OOA)和面向对象的设计(OOD)技术等。

### (5)结果应能清楚地审查

软件产品是一种看不见、摸不着的逻辑产品。因此，软件开发小组的工作进展情况可见性差，难以评价和管理。为了更好地进行评价和管理，应根据软件开发的总目标和完成期限，尽量明确地规定软件开发小组的责任和产品标准，从而能清楚地审查所得到的结果。

### (6)开发小组的人员应少而精

软件开发小组人员的素质和数量是影响软件质量和开发效率的重要因素。实践表明，素质高的人员与素质低的人员相比，其软件开发的效率可能高出几倍甚至几十倍，而且所开发的软件中的错误也要少得多。另外，开发小组的人数不宜过多，因为随着人数的增加，人员之间交流情况、讨论问题的通信开销将急剧增加，这不但不能提高生产效率，反而由于误解等原因可能会增加出错的概率。

### (7)承认不断改进软件工程实践的必要性

遵循上述 6 条基本原理，就能较好地实现软件的工程化生产。但是，软件工程不能停留在已有的技术水平上，应积极主动地采纳或创造新的软件技术，要注意不断总结经验，收集工作量、进度和成本等数据，并进行出错类型和问题报告的统计。这些数据既可用来评估新的软件技术的效果，又可用来指明应优先进行研究的软件工具和技术。

## 3. 软件工程的基本目标

从技术和管理上采取多项措施以后，组织实施软件工程项目的最终目的是保证项目成功，即达到以下几个主要目标：

- 付出较低的开发成本；
- 达到预期的软件功能；
- 取得较好的软件性能；使软件易于移植；
- 需要较低的维护费用；
- 能按时完成开发工作，及时交付使用。

在项目的实际开发中，使以上几个目标都达到理想的程度往往非常困难，而且上述目标很可能是相互冲突的，图 1-1 表明了软件工程目标之间存在的相互联系。有些目标是互补的，如高可靠性与易于维护之间、低开发成本与按时交付之间。有些目标之间则是互斥的，如低开发成本与高可靠性之间、高性能与高可靠性之间。

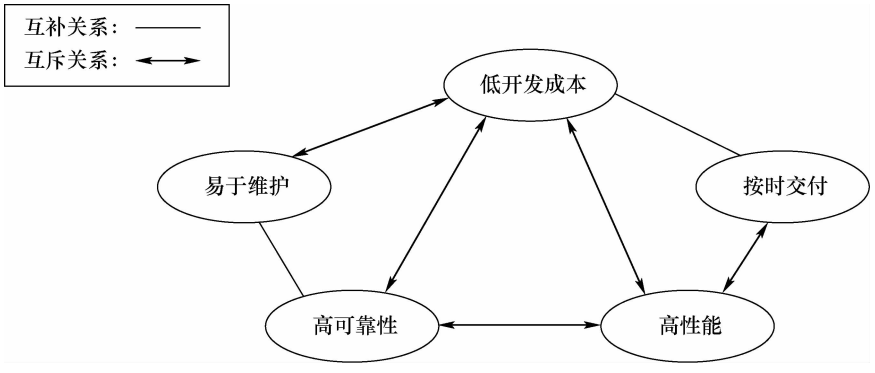


图 1-1 软件工程目标之间的关系

## 4. 软件危机

软件危机是指在计算机软件的开发、使用和维护过程中遇到的一系列严重问题。

### (1)软件危机的表现

应该说，自从计算机诞生以来，软件危机就一直存在。软件危机主要表现在以下几个方面：

①人们对软件开发的成本和进度的估计常常不够准确。由于软件的特殊性，不同类型的软件其开发所需的工作量、成本往往差别很大。因此，常常出现实际成本比估算成本高出一个数量级，实际进度比计划进度拖延几个月甚至几年的现象，这大大降低了开发商的信誉，也引起了用户的不满。

②用户对已完成的软件不满意的现象时有发生。这主要是由于在开发的初期，软件需求不够明确，开发过程中又未能和用户及时交换意见，致使开发出的软件不能满足用户的需求，甚至无法使用。

③软件常常是不可维护的。在软件开发过程中，没有统一、公认的方法和规范进行指导，且设计和实现过程的资料很不完整，这使得软件出现问题后很难维护。

④软件产品的质量往往不可靠。由于未做好测试阶段的工作，提交给用户的软件质量差，在运行中暴露出大量问题。

⑤软件开发生产率提高的速度远远跟不上日益增长的软件需求，满足不了社会发展的需要。

### (2)缓解软件危机的途径

到 20 世纪 60 年代末期，软件危机已相当严重。要想解决软件危机，必须做好以下几方面的工作：

①加强软件开发过程的管理，做到组织有序、各类人员协同配合，共同保证工程项目完成，避免软件开发过程中个人单干的现象；

②推广使用开发软件的成功技术和方法，并且不断探索更好的技术和方法，消除一些错误的概念和做法；

③开发和使用好的软件工具，支持软件开发的全过程。

**拓展提高**

如何做好上述三方面的工作来缓解软件危机呢？计算机科学家们经过多年的研究，提出了“软件工程”的概念，即用现代工程的原理、技术和方法进行软件的开发、管理、维护和更新。于是，开创了计算机科学技术的一个新的研究领域——软件工程。几十年来，软件工程在软件开发方法、工具和管理等方面的研究与应用已经取得了很多成果，并已大大缓解了软件危机所带来的影响。

# 任务二：选择软件生存周期模型

## 任务描述

王明被任命为一家软件公司的项目负责人，工作是管理该公司已被广泛应用的字处理软件的新版本开发。由于市场竞争激烈，公司规定了严格的开发完成期限并且对外发布了该信息。他需要据此确定合适的软件生存周期模型，以保证按期完成新版本的开发。

## 任务分析

类似于其他工程项目中安排各道工序那样，为反映软件生存期内各种活动应如何组织，软件生存期的 6 个阶段应如何衔接，需要用软件生存期模型做出直观的图示表达。软件生存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。

## 准备知识

### 1. 软件生存周期

同任何事物一样，软件也有一个孕育、诞生、成长、成熟、衰亡的生存过程。软件生存周期是指一个计算机软件从功能确定、设计，到开发成功投入使用，并在使用中不断地修改、增补和完善，直到停止该软件的使用的全过程。包括制订计划、需求分析、软件设计、程序编码、软件测试和运行维护等 6 个阶段。以下对这 6 个阶段的工作流程及主要任务做一概括的描述。

#### (1)制订计划

在软件系统开发之前，首先应当制订项目开发计划，该阶段是软件生存周期的第一阶段。其主要任务如下：

- 确定待开发软件系统的总目标；
- 给出功能、性能、可靠性以及接口等方面的要求；
- 完成该软件任务的可行性研究；
- 估计可利用的资源(硬件、软件和人力等)、成本、效益和开发进度；
- 制订出完成开发任务的实施计划，连同可行性研究报告，提交管理部门审查。

#### (2)需求分析

当完成开发计划的制订之后，需要对用户的需求去粗取精、去伪存真、正确理解，然后把它用软件工程开发语言表达出来。其主要任务如下：

- 去用户处做需求调研，让用户提出对软件系统的所有需求；
- 对用户提出的需求进行分析、综合，并给出详细的定义；
- 编写软件需求说明书及初步的系统用户手册，提交管理机构评审。

### (3)软件设计

需求分析和定义阶段结束之后，对于软件必须“做什么”的结论已经明确，下一步是如何实现软件的需求，即进入软件设计阶段，该阶段又可分为概要设计和详细设计两部分，其主要工作如下：

- 概要设计：把各项软件需求转化为软件系统的总体结构和数据结构，结构中每一部分都是意义明确的模块，每个模块都和某些需求相对应；
- 详细设计：即过程设计，对每个模块要完成的工作进行具体的描述，即给出详细的数据结构和算法，为源程序的编写打下基础；
- 编写设计说明书，提交评审。

### (4)程序编写

软件设计解决了软件“怎么干”的问题，而程序编写是在计算机上真正实现一个具体的软件系统，具体的工作包括以下两个方面：

- 把软件设计转换成计算机可以接受的程序代码，即写成以某一种特定的程序设计语言表示的“源程序清单”，这一步工作也称为编码；
- 要求写出的程序应该是结构良好、清晰易读的，且与设计相一致。

### (5)软件测试

软件分析、设计和程序编写过程中难免有各种各样的错误，需要通过测试来查找和修改，以保证软件的质量。其主要工作如下：

- 单元测试：查找各模块在功能和结构上存在的问题并加以纠正；
- 集成测试：将已测试通过的模块按一定顺序组装起来进行测试；
- 有效性测试：按规定的各项需求，逐项进行测试，判断已开发的软件是否合格，能否交付用户使用。

### (6)运行维护

软件项目开发成功后，要投入运行。软件系统在运行过程中，会不断受到系统内外环境变化及各种人为的、技术的、设备的影响，要求软件能够适应这种变化，不断地完善，这就要进行软件维护，以保证其正常而可靠地运行，并能使软件不断地得到改善和提高，充分发挥其作用。软件维护有 4 种类型，它们分别完成以下任务：

- 纠正性维护：运行中发现了软件中的错误而进行的修改工作；
- 适应性维护：为了适应变化了的软件工作环境，而做出适当的变更；
- 完善性维护：为了增强软件的功能而做出的变更；
- 预防性维护：为未来的修改与调整奠定更好的基础而进行的工作。

## 2. 瀑布模型

最早的软件开发模型是 1970 年 W. Royce 提出的瀑布模型，而后随着软件工程学科的发展和软件开发的实践，相继提出了瀑布模型、快速原型模型、增量模型、螺旋模型、喷泉模型、形式化方法模型等。

瀑布模型规定了各项软件工程活动，包括制订开发计划、需求分析和说明、软件设计、程序编码、测试和运行、维护，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，如图 1-2 所示。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的，呈线性图式。

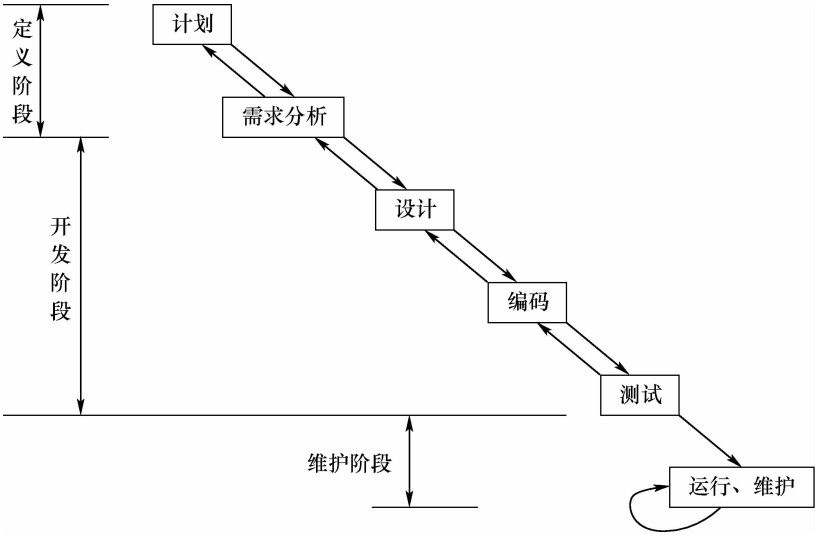


图 1-2 瀑布模型

实际情况是，每项开发活动均应具有以下特征：

①从上一项活动接受本项活动的工作对象，作为输入；

②利用这一输入实施本项活动应完成的工作；

③给出本项活动的工作成果，作为输出传给下一项活动；

④对本项活动实施的工作进行评审，若其工作得到确认，则继续进行下一项活动。否则返回前一项活动，甚至更前项的活动进行返工。

为了保障软件开发的正确性，每一阶段任务完成后，都必须对它的阶段性产品进行评审，确认之后再转入下一个阶段。如评审过程中发现错误和疏漏，应该返回到前面的有关阶段修正错误、弥补漏洞，然后再重复前面的工作，直至该阶段的工作通过评审后再进入下一阶段。

**知识链接**

严格按照软件生存周期的阶段划分，顺序执行各阶段构成软件开发的瀑布模型，是传统.的软件工程生存期模式。采用瀑布模型进行开发组织时，应指定软件开发规范或开发标准，其中要明确规定各个开发阶段应交付的产品，这为严格控制软件开发项目的进度，最终按时交付产品以及保证软件产品质量创造了有利条件。瀑布模型为软件开发和维护提供了一种有效的管理模式。

瀑布模型 20 多年来之所以广泛流行，是因为它在支持结构化软件开发、控制软件开发的复杂性、促进软件开发工程化等方面起着显著作用。但与此同时，瀑布模型在大量软件开发实践中也逐渐暴露出它的缺点，其中最为突出的缺点是该模型缺乏灵活性，特别是无法解决软件需求不明确或不准确的问题，这些问题可能导致最终开发出的软件并不是用户真正需要的软件，并且这一点往往在开发过程完成后才有所察觉。面对这种情况，无疑需要进行返工或不得不在维护中纠正需求的偏差，为此必须付出高额的代价，为软件开发带来损失。并且，随着软件开发项目规模的日益庞大，该模型的不足所引发的问题显得更加严重。

为弥补瀑布模型的不足，近年来已经提出了多种其他模型，现介绍如下。

### 3. 快速原型模型

瀑布模型的缺陷在于软件开发阶段的推进是直线型的。按照这一模型来开发软件，只有当分析员能够做出准确的需求分析时，才能够得到预期的正确结果。不幸的是，由于多数用户不熟悉计算机，系统分析员对用户的专业也往往了解不深，在需求分析阶段定义的用户需求，常常是不完全和不准确的。对于用户和分析员都未经历过的新的系统，这种情况就更加突出。

为了克服瀑布模型的这种缺陷，人们提出了快速原型模型。快速原型模型的基本思想是：软件开发人员根据用户提出的软件基本需求快速开发一个原型，以便向用户展示软件系统应有的部分或全部的功能和性能，在征求用户对原型的评价意见后，进一步使需求精确化、完全化，并据此改进、完善原型，如此迭代，直到软件开发人员和用户都确认软件系统的需求并达成一致的理解为止。软件需求确定后，便可进行设计、编码、测试等以后的各个开发步骤。可见，原型主要是为了完成需求分析阶段的任务而构建的。利用原型确定软件系统需求的过程如图 1-3 所示。

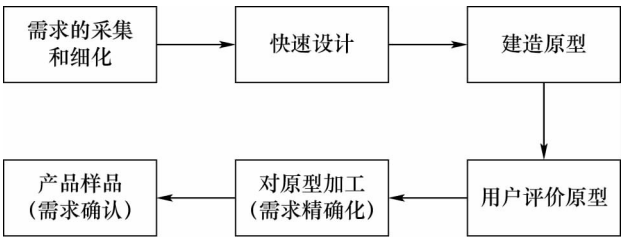


图 1-3　快速原型模型

**知识链接**

快速原型的开发途径有 3 种：

①仅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，实现软件系统中重要的或容易产生误解的功能；

③利用一个或几个类似的正在运行的软件向用户展示软件需求中的部分或全部功能。

总之，建造原型应尽量采用相应的软件工具和环境，并尽量采用软件重用技术，在运行效率方面可以做出让步，以便尽快提供；同时，原型应充分展示软件系统的可见部分，如人机界面、数据的输入方式和输出格式等。

虽然用户和开发人员都非常喜欢原型模型，因为它使用户能够感受到实际的系统，开发人员能很快建造出一些东西。但该模型仍然存在着一些问题，原因如下：

①用户看到的是一个可运行的软件版本，但不知道这个原型是临时搭建起来的，也不知道软件开发者为了使原型尽快运行，并没有考虑软件的整体质量或今后的可维护性问题。当被告知该产品必须重建，才能使其达到高质量时，用户往往叫苦连天。

②开发人员常常需要在实现上采取折中的办法，以使原型能够尽快工作。开发人员很可能采用一个不合适的操作系统或程序设计语言，仅仅因为它通用和有名；可能使用一个效率低的算法，仅仅为了演示功能。经过一段时间之后，开发人员可能对这些选择已经习以为常了，忘记了它们不合适的原因。于是，这些不理想的选择就成了系统的组成部分。

### 4. 增量模型

像建造一个建筑物一样，软件的开发也是逐步进行的。增量模型规定软件的开发过程是一次开发产品的一个部分。首先应该开发产品的基本部分，然后再逐步开发产品的附加部分。为了使所开发的产品的各个部分最后能有机地组合起来，首先总该有一个统一的体系结构设计。增量模型的开发过程如图 1-4 所示。在该模型中，产品的设计、实现、集成和试验是以一系列增量构件为基础进行的，构件是由一些模块的编码构成并能提供特定的功能。例如，在操作系统中，调度程序是一个构件，文件管理系统也是一个构件。在增量模型的每一个阶段，都要编码一个新的构件，然后集成到先前已构成的产品中并作为一个整体进行测试，当这个产品满足规定的功能时，即满足它的需求规范时，这个过程停止。开发者可以任意分解目标产品以得到一些构件，但是这些构件必须能集成为一个满足特定功能的产品。如果目标产品只能分解为很少的构件，那么增量模型可能退化为建造与修改模型。如果目标产品分解为太多的构件，那么每个阶段结束时，用户可能得不到需要的功能，并且浪费更多的时间在集成上，因此构件的大小应适中，这取决于用户的需求。一个典型的产品通常由 10～50 个构件组成。

瀑布模型和快速原型模型都是只提交可操作的、完整的和具有一定质量的软件。而客户希望交付的产品能满足其所有的需求，并且希望得到全面的和正确的文档，这

些文档能应用于各种维护。但是客户为了得到其产品，必须苦等数月或者数年。

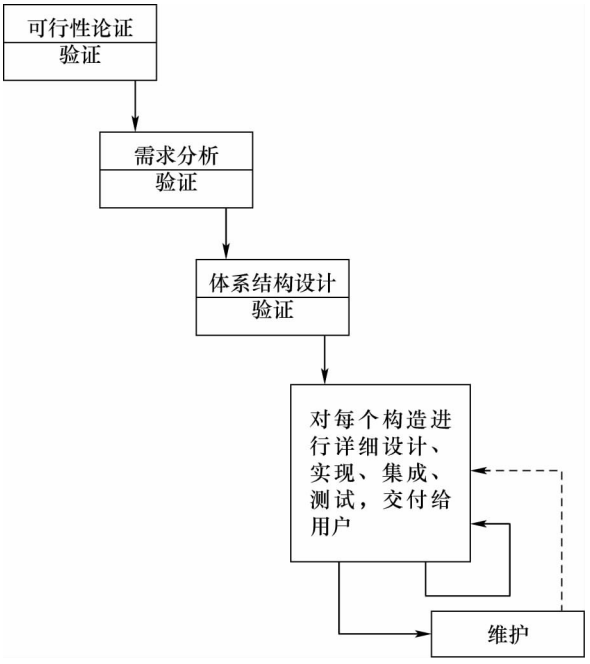


图 1-4　增量模型

而增量模型在每个阶段都交付一个可操作的产品，但它仅仅是满足客户需求的一个子集。完整的产品划分成一些构件，产品是以一次一个构件的方式开发的。在每个开发阶段之后，客户都能得到一个产品，从第一个构件交付开始，客户就能做有用的工作。使用增量模型，整个产品的某些部分在数量期之内是可用的。当使用增量模型时，第一个增量经常是核心产品，它满足用户的基本需求，而许多附加功能将在以后的增量中交付。当在规定的期限内没有足够的人员来开发完整的产品时，或者由于不可克服的客观原因而把交付期限规定得太短时，增量开发方式是特别有用的。

**拓展提高**

增量模型的一个难点是保证后来开发的构件必须能够集成到先前已开发的产品中而不毁坏已开发的功能。进而，现存的产品必须容易扩充，后开发的构件必须是简单和直观并容易集成的。因此，对于增量模型，产品体系结构的设计必须是开放的。

### 5. 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和原型模型结合起来，不仅体现了两个模型的优点，而且还增加了两个模型都忽略了的风险分析，弥补了两者的不足。

在此先简要说明什么是风险分析。“软件风险”是普遍存在于任何软件开发项目中的实际问题。对于不同的项目，其差别只是风险有大有小而已。在制订软件开发计划时，系统分析员必须回答：项目的需求是什么，需要投入多少资源以及如何安排开发进度等一系列问题。然而，若要他们当即给出准确无误的回答是不容易的，甚至是不可能的，但系统分析员又不能完全回避这一问题，而凭借经验给出的答案难免带来一定的风险。实践表明，项目规模越大，问题越复杂，资源、成本和进度等因素的不确定性越大，承担项目所冒的风险也越大。总之，风险是软件开发不可忽略的潜在不利因素，它可能在不同程度上影响软件的开发过程或损害软件产品的质量。软件风险驾驭的目标是在造成危害之前，及时对风险进行识别、分析，采取对策，进而消除或减少风险带来的损害。

螺旋模型的结构如图 1-5 所示，它由 4 部分组成：制订计划、风险分析、实施开发、客户评估。在笛卡儿坐标的 4 个象限上分别表达了 4 个方面的活动。

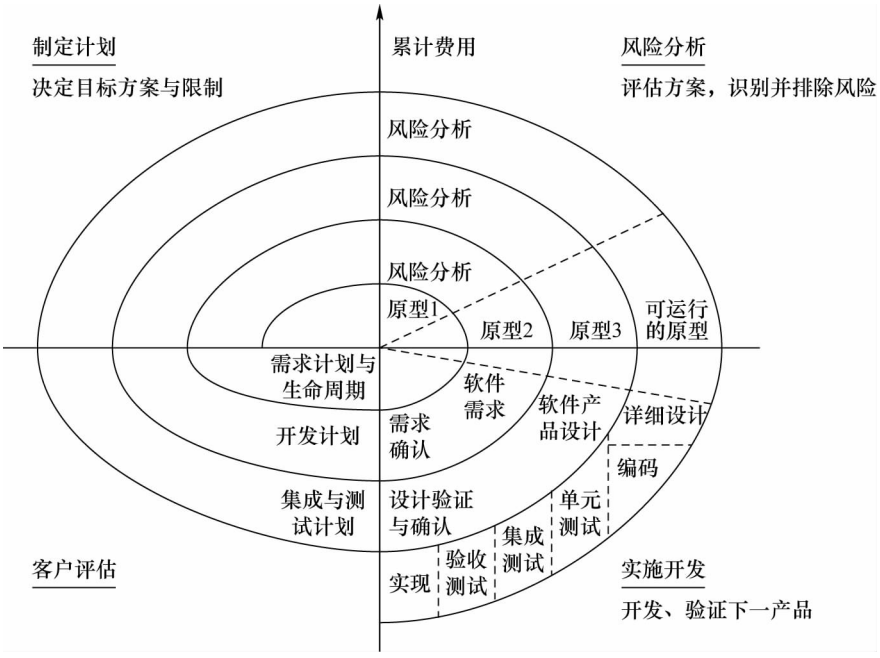


图 1-5 螺旋模型

沿螺旋线自内向外每旋转一圈便开发出一个更为完善的新的软件版本。例如，在第一圈，在制订计划阶段，确定了初步的目标、方案和限制条件以后，转入风险分析阶段，对项目的风险进行识别和分析。如果风险分析表明，需求有不确定性，但是风险可以承受，那么在实施开发阶段，所建的原型会帮助开发人员和用户，对需求做进一步的修改。软件开发完成后，客户会对工程成果做出评价，给出修正建议。在此基础上进入第二圈螺旋，再次进行制订计划、风险分析、实施开发和客户评估等工作。假如风险过大，开发者和用户无法承受，项目有可能终止。多数情况下，软件开发过程是沿螺旋线的路径连续进行的，自内向外，逐步延伸，最终总能得到一个用户满意的软件版本。

如果软件开发人员对所开发项目的需求已有了较好的理解，则无须开发原型，可采用普通的瀑布模型，这在螺旋模型中可认为是单圈螺旋线。相反，如果对所开发项目的需求理解较差，则需要开发原型，甚至需要不止一个原型的帮助，此时就要经历多圈螺旋线。但在实际开发中，应该尽量降低迭代次数，减少每次迭代的工作量，这样才能降低开发成本和时间；反之，由于时间和成本上的开销太大，客户无法承受，软件系统的开发有可能中途夭折。

**拓展提高**

螺旋模型不仅保留了瀑布模型中系统地、按阶段逐步地进行软件开发和“边开发、边评审”的风格，而且还引入了风险分析，并把制作原型作为风险分析的主要措施。用户始终关心、参与软件开发，并对阶段性的软件产品提出评审意见，这对保证软件产品的质量是十分有利的。但是，螺旋模型的使用需要具有相当丰富的风险评经验和专门知识，而且费用昂贵，所以只适合大型软件的开发。

### 6. 喷泉模型

喷泉模型是近几年提出来的软件生存周期模型。它是以面向对象的软件开发方法为基础，以用户需求为动力，以对象来驱动的模型。喷泉模型如图 1-6 所示。喷泉模型的特点如下：

①整个开发过程（包括维护过程）包括 5 个阶段，即需求分析、设计、实现、测试和维护。因此，使软件系统可维护性较好。

②各阶段相互重叠，表明了面向对象开发方法各阶段间的交叉和无缝过渡。

③整个模型是一个迭代的过程，包括一个阶段内部的迭代和跨阶段的迭代。

④模型具有增量开发特性，即能做到边分析，边设计；边实现，边测试，使相关功能随之加入到演化的系统中。

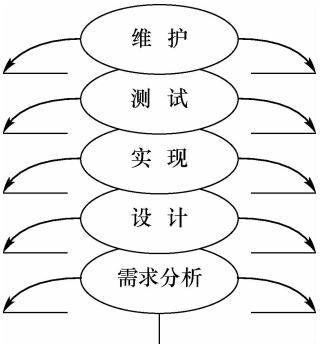

图 1-6 喷泉模型

⑤模型是对象驱动的，对象是各阶段活动的主体，也是项目管理的基本内容。

⑥该模型很自然地支持软件部件的重用。

### 7. 形式化方法模型

形式化方法模型包含了一组活动，它们带来了计算机软件用数学描述的方法。形式化方法使得软件开发人员能够通过采用一个严格的、数学的表示体系来说明、开发和验证基于计算机的系统。用于软件形式化开发方法的模型有两种：变换模型和净室软件过程模型。

### (1)变换模型

变换模型是一种用于软件的形式化开发的方法:在软件需求分析确定以后,便用形式化的规格说明语言将其描述为“形式化软件规格说明”,然后对其进行一系列自动或半自动的变换,最终得到软件系统的目标程序,变换模型如图 1-7 所示。由于一开始就要求做出正确、完全的需求分析在多数情况下是做不到的,因此,变换模型也应引入迭代机制,即将第一次用变换模型得来的目标程序作为“原型”。让用户评价,以便使用户需求精确化、完全化,再把精化后的需求作为输入,第二次用变换模型进行变换等。

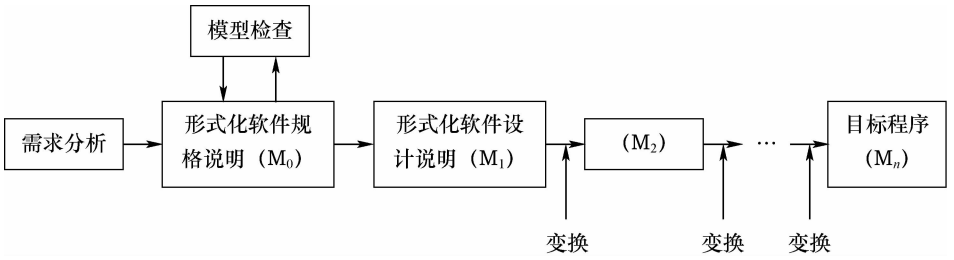


图 1-7 变换模型

### (2)净室软件过程模型

净室软件工程也是软件开发的一种形式化方法,它力求在分析和设计阶段就消除错误,确保正确,然后在无缺陷或“洁净”的状态下实现软件的制作,以生成极高质量的软件。它使用盒结构规约进行分析和设计建模,并且强调将正确性验证而不是测试,作为发现和消除错误的主要机制。它使用测试来获取认证,使被交付的软件的出错率达到最低。

净室软件过程模型是一种严格的软件工程方法,它是一种强调正确性的数学验证和软件可靠性认证的软件工程模型,其目标和结果是降低出错率,这是使用形式化方法难于或不可能达到的。

形式化方法提供了一种机制,它能够消除使用其他软件工程模型难以克服的问题,包括二义性、不完整性和不一致性都能被更容易地发现和纠正。它不是通过专门的复审,而是通过数学分析来实现。当在设计中使用形式化方法时,它们能作为程序验证的基础,从而使得软件开发人员能够发现和纠正在其他情况下发现不了的错误。

**知识链接**

形式化方法模型虽然不是主流的方法,但可以产生正确的软件。不过,对它在商业环境中的可用性还需要做出以下考虑:

①形式化模型的开发目前还是很费时和昂贵;

②因为很少有软件开发人员具有使用形式化方法所需的背景知识,所以尚需多方面的培训;

③难以使用该模型作为与对其一无所知的用户进行通信的机制。

### 任务实施

对这个软件开发项目的一个关键要求是，严格按照已对外公布的日期完成产品开发工作，因此，选择生命周期模型时应该着重考虑哪种模型有助于加快产品开发的进度。

这个项目是开发该公司已被广泛应用的字处理软件的新版本，从上述事实至少可以得出 3 点结论：第一，旧版本相当于一个原型，通过收集用户对旧版本的反映，较容易确定对新版本的需求，没必要再专门建立一个原型系统来分析用户的需求；第二，该公司的软件工程师对字处理软件很熟悉，有开发字处理软件的丰富经验，具有开发新版字处理软件所需要的技术水平；第三，该软件受到广大用户的喜爱，今后很可能还要开发更新的版本，因此，应该把该软件的体系结构设计成开放式的，以利于今后的改进和扩充。

综上所述，采用增量模型来完成这个项目比较恰当。

## 任务三：了解软件开发方法和软件开发工具

### 任务描述

周健是某计算机软件研发中心的技术人员，接到领导任务，要求其为新入职大学生做一次软件开发方法与开发工具方面的基础培训，为后续培训和实际工作的开展打下基础。

### 任务分析

选择合适的软件开发方法和开发工具有助于高效、高质量完成软件开发工作。因此，软件开发方法和软件开发工具的基本知识非常重要。

### 准备知识

软件工程学的最终目标是以较低的成本获得质量较高的软件产品，也就是说要“高产优质”。同其他工程学科一样，达到这个最终目标的两个主要途径是“纪律化”和“自动化”。

研究软件方法的目的是使开发过程“纪律化”，就是寻找一些规范的“求解过程”，使开发工作能够有计划、有步骤地进行。研究软件开发工具的目的是使开发过程“自动化”，就是使开发过程中的某些工作用计算机来完成或用计算机来辅助完成。方法和工具之间有着密切的联系，方法是主导，工具则是辅助。

## 任务实施

软件开发方法和开发工具的选择并不是一个简单的工作，但是万丈高楼平地起，相关基本知识是其根本。为此，周健确定以下培训内容，强化新入职大学生对软件开发方法和开发工具的基本认识，为将来的实际工作打下良好的知识基础。

### 1. 软件开发方法的概念

早期的程序设计是一种个体的、手工艺方式的劳动，对于一个提出的问题，程序员可以任意地“发明”出一个解答，而没有任何规章制度需要遵循。但是当软件进入大规模生产时期，软件就不再是个人的成果而是集体的劳动成果了，所以软件产业面临的第一个问题就是“纪律化”。

一项软件产品的诞生涉及许多人，包括用户和软件开发人员两方面。每一方面又有许多人，每个人对问题的理解可能不同，对问题的处理方式也可能不同；软件产品的开发周期又较长，即使是同一个人，在不同的时期，对问题的理解和处理还有可能不同；而且随着计算机技术的发展，处理问题的选择余地就更大了。软件开发过程中，缺乏强有力的内部纪律，各人可以任意地自行其是，这是造成软件危机的主要原因之一。

为了使软件研制走上工程化的轨道，必须寻找一些标准的规程，以便对开发人员进行指导和约束，使他们遵照一定的方式来理解和处理问题，这是成功开发软件的关键。

软件方法就是指导研制软件的某种标准规程，它告诉人们“什么时候做什么以及怎样做”。由于软件研制过程是相当复杂的，它涉及的因素很多，所以各种软件方法又有不同程度的灵活性和试探性。软件方法不可能像自动售货机的使用规程那样用简单的几句话就可以叙述清楚，也不应该像机器的操作规程那样机械地使用，一般说来，一个软件方法往往规定了明确的工作步骤、具体的描述方式以及确定的评价标准。

#### (1)明确的工作步骤

研制一个软件系统要考虑并解决许多问题，如果同时处理这些问题，将会束手无策或造成混乱。正确的方式是将这样的问题排好先后次序，每一步集中精力解决一个问题。像为加工机械产品规定一道道工序那样，软件方法也提出了处理问题的基本步骤，这包括每一步的目的、产生的工作结果、需具备的条件以及要注意的问题等。

#### (2)具体的描述方式

工程化生产必须强调文档化，即每人必须将每一步的工作结果以一定的书面形式记录下来，以保证开发人员之间有效地进行交流，也有利于维护工作的顺利进行。软件方法规定了描述软件产品的格式，这包括每一步应产生什么文档、文档中记录哪些内容、采用哪些图形和符号等。

#### (3)确定的评价标准

对于同一个问题，其解决方案往往不是唯一的，选取哪一个方案较好呢？有些软

件方法提出了比较确定的评价标准，因而可以指导人们对各个具体方案进行评价，并从中选取一个较好的方案。

**拓展提高**

在软件方法的指导和约束之下，面对错综复杂的问题，开发人员就可按统一的步骤、统一的描述方式，纪律化地开展工作。毫无疑问，这是“高产优质”的有力保证。

## 2. 软件开发的基本方法

软件开发的基本方法包括结构化方法、面向对象方法等，这些方法都是以软件生存周期为基本特征的软件工程方法。

### (1)结构化方法

结构化方法(structured method)是较传统的软件开发方法。在 20 世纪 60 年代初，就提出了用于编写程序的结构化程序设计(Structured Programming，SP)方法，而后发展到用于设计的结构化设计(Structured Design，SD)方法、用于分析的结构化分析(Structured Analysis，SA)方法等；在早期软件生存周期的各个阶段中，所使用的软件开发方法都是结构化方法。

结构化方法的基本思想可以概括为自顶向下、逐步求精，采用模块化技术和功能抽象将系统按功能分解为若干模块，从而将复杂的系统分解成若干易于控制和处理的子系统，子系统又可分解为更小的子任务，最后的子任务都可以独立编写成子程序模块，模块内部由顺序、选择和循环等基本控制结构组成。这些模块功能相对独立，接口简单，使用维护非常方便。所以，结构化方法是一种非常有用的软件开发方法，是其他软件工程方法的基础。

但是，由于结构化方法将过程与数据分离为相互独立的实体，因此开发的软件可复用性较差，在开发过程中要使数据与程序始终保持相容也很困难。这些问题通过面向对象方法就能得到很好的解决。

### (2)面向对象方法

面向对象方法(object－oriented method)是针对结构化方法的缺点，为了提高软件系统的稳定性、可修改性和可重用性而逐渐产生的：开始主要用在程序编码中，以后又逐渐出现了面向对象的分析(Object-Oriented Analysis，OOA)和设计(Object-Oriented Design，OOD)方法，是当前软件开发方法的主要方向，也是目前最有效、实用和流行的软件开发方法之一。面向对象方法的出发点和基本原则，是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、解决问题的方法与过程，将客观世界中的实体抽象为问题域中的对象。

**知识链接**

面向对象方法主要有以下几个特点：

①认为客观世界是由各种对象组成的，任何事物都是对象；

②即所有对象都划分为各种对象类，每个类定义一组数据和一组方法；

③按照子类与父类的关系，把若干对象类组成一个层次结构的系统；

④对象彼此之间仅能通过传递消息相互联系。

面向对象方法的主要优点是使用现实的概念抽象地思考问题，从而自然地解决问题，保证软件系统的稳定性和可重用性以及良好的维护性。但是面向对象方法也不是十全十美的，在实际的软件开发中，常常要综合应用结构化方法和面向对象方法。

### 3. 软件开发工具

软件开发工具也是软件工程的主要内容之一。软件开发工具为软件开发方法提供自动的或半自动的软件支撑环境。当一种方法提出并证明有效后，往往就会随之研制出相应的软件开发工具，来帮助实现和推行这种方法，提高软件设计效率，减轻劳动强度。

目前已经推出很多软件开发工具，如需求分析阶段的 PSL/PSA 系统；编码阶段的各种语言编译工具、编辑程序、连接程序等，都是软件编码阶段的软件工具；测试阶段的测试数据产生程序、动态分析程序、静态分析程序等软件自动测试工具；维护阶段的版本控制系统等。从广义上来讲，软件分析、设计阶段的各种图形工具，如数据流图（DFD 图）等也可以称为软件开发工具。

众多的工具组合起来，可以组成工具箱或集成工具，供软件开发人员在不同阶段按需选用。如果把诸多的软件开发工具集成起来，使得一种工具产生的信息可以为其他的工具所使用，这样建立起来的软件开发支撑系统被称为计算机辅助软件工程（Computer Aided Software Engineering，CASE），它是将各种软件工具、开发机器和存放开发过程信息的工程数据库组合起来形成的软件工程环境。

## 任务四：强化认识面向对象的软件工程

### 任务描述

为使新入职大学生对目前软件开发的主流方法——面向对象的软件工程有一个更加清晰的认识，公司领导要求老员工王明为此准备相关的培训资料。

### 任务分析

传统的软件工程曾经给软件产业带来了巨大的进步，部分地缓解了软件危机，但