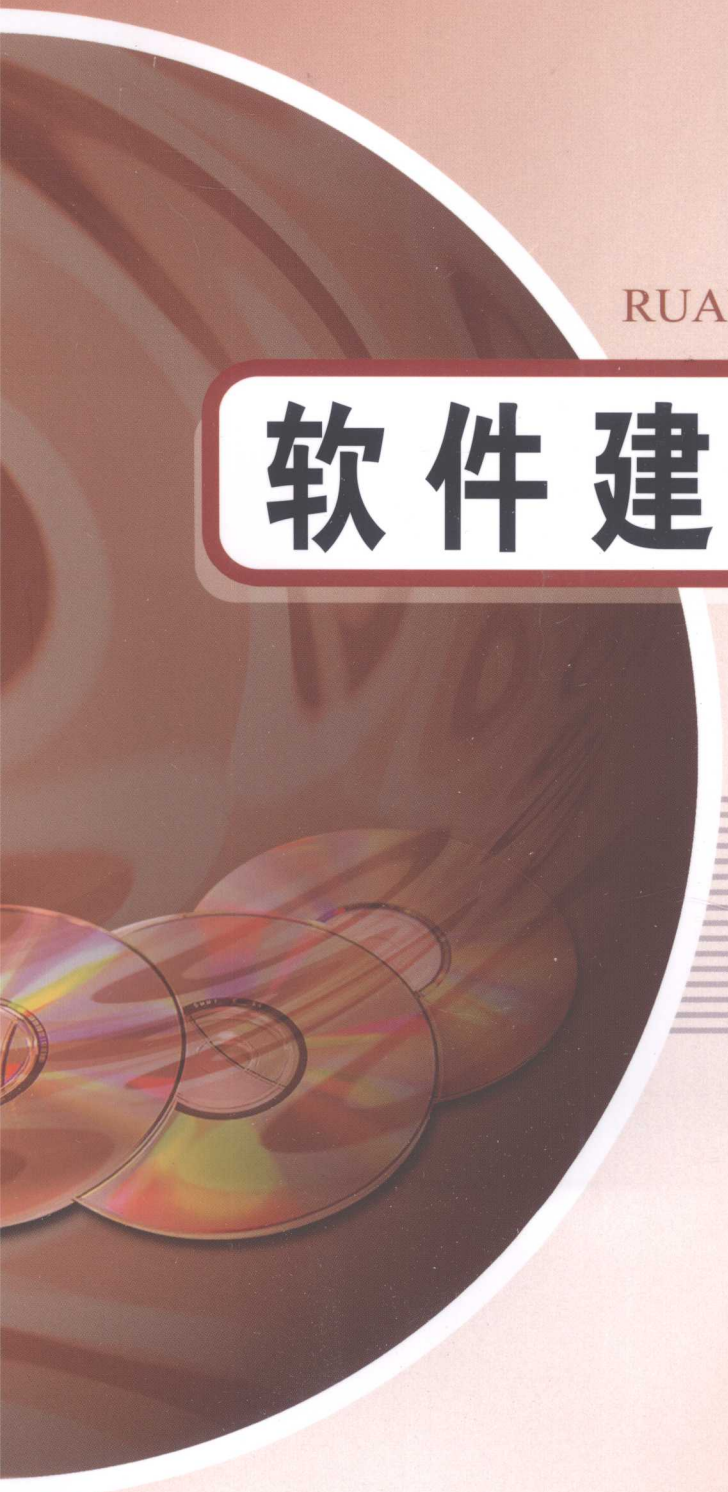
高等职业教育“十一五”精品课程规划教材



RUANJIAN JIANMO JISHU

软件建模技术

江 文 主编

北京邮电大学出版社
www.buptpress.com

高等职业教育“十一五”精品课程规划教材

软件建模技术

江 文 主 编

北京邮电大学出版社
·北京·

内 容 简 介

本书以实用够用为原则，介绍了 UML 的基本概念，全书内容由浅入深逐步地展开，并通过大量的案例和课堂练习，力图使初学者容易理解。

本书从最基本的内容讲起，对 UML 的强大功能进行了详细而系统的介绍，并着重论述了如何使用 UML 对应用系统进行建模；同时，为配合知识点的讲述，将一个实际项目作为案例与所讲述的内容融合在一起，力图做到从应用中来到应用去，如用类图和交互图来描述 HNS 软件学院的 JQQ 聊天系统中的静态和动态行为。因此，本书是一本以知识为导向，以实际应用为目标的 UML 教材。

图书在版编目(CIP)数据

软件建模技术/江文主编．—北京：北京邮电大学出版社，2009

ISBN 978-7-5635-2031-2

Ⅰ．软… Ⅱ．江… Ⅲ．面向对象语言，UML—程序设计 Ⅳ．TP312

中国版本图书馆 CIP 数据核字(2009)第 114907 号

书　　名：软件建模技术

主　　编：江　文

责任编辑：周　堃

出版发行：北京邮电大学出版社

社　　址：北京市海淀区西土城路 10 号(邮编：100876)

发 行 部：电话：010-62282185　传真：010-62283578

E-mail：publish@bupt.edu.cn

经　　销：各地新华书店

印　　刷：北京忠信诚胶印厂

开　　本：787 mm×1 092 mm　1/16

印　　张：8.75

字　　数：208 千字

印　　数：1—3 000 册

版　　次：2009 年 8 月第 1 版　2009 年 8 月第 1 次印刷

ISBN 978-7-5635-2031-2　　定　价：15.00 元

目　录

第1章 UML和软件工程

本章目标

UML (Unified Modeling Language,统一建模语言)是一种可视化的建模语言,主要应用于软件工程领域的建模。本章主要是对一些基本概念的描述,因此非常重要。通过本章的学习,可以理解UML和软件工程的主要概念,为后续的学习打好基础。本章的学习目标如下:

- 理解建模的概念
- 理解软件工程的基本概念
- 理解UML基本概念,以及UML描述模型的3种构造块(事物、关系、图)

1.1 UML概述

内容提要

UML是一种可视化的建模语言,它可以用来创建各种不同类型的模型。本节将首先讲述建模的概念,然后引出建模语言——UML。在UML这一小节中主要介绍了UML的历史、UML的基本概念。另外,必须使用一种工具来实现UML建模,因此在本节的最后介绍了UML建模工具——Rational Rose。总的来说,本节主要内容如下:

- 建模概述
- UML简述
- Rational Rose的使用介绍

1.1.1 建模

1. 什么是模型

什么是模型?在回答这个问题之前,先来回忆一下生活中常见的一些图表、文字:介绍天气情况的气象图;指示交通情况的交通地图;说明一部泡沫式灭火器如何打开的过程描述图……所有这些,构成了周围世界各种各样的模型。那么,模型是什么呢?简单地说,模型是对现实的简化。它可以是一个对象的微缩表示,是一种用于生产某事物的模式,也可以是一种设计或一个类型,还可以是一个待模仿或仿真的样例。一个好的模型包

括那些有广泛影响的主要元素，而忽略那些与给定的抽象水平不相关的次要元素。每个系统都可以从不同的方面用不同的模型来描述，就像中国一句古诗所说“横看成岭侧成峰”。另外，模型不一定是可视化的，模型也可以用文字来描述，如用文字描述车间里一个产品的生产流程，但是可视化模型可以更准确地展示模型所代表的含义。

2. 建模的目的和原则

为什么要建模？一个基本理由是，建模是为了能够更好地理解我们正在开发的系统。在开发一个系统的时候，建模可以帮助我们沟通设计思想，理解业务内容，澄清复杂的问题和场景，确保我们设计的系统在被实现之前更符合用户需求。按照这样的方式思考问题，我们会在没有规划之前就开工建设一栋大厦吗？也许造一个平房不需要详细的规划和设计，但想要建造好一栋大厦，最好先仔细设计一番。当使用一套好的设计图纸，并严格依照图纸施工，所建造的大厦才能经得起时间的检验。而且，越是复杂、庞大的系统，越是需要建模。因为人对复杂问题的理解能力是有限的，通过建模，可以帮助人们理解复杂的问题，一次只研究复杂系统的一个方面，即先把一个要解决的难题分解成一系列的小问题，解决了这些小问题也就解决了这个难题。

什么样的模型是合乎要求的呢？如果建立的模型对工作没有多大的帮助，或者反而有误导作用，这样的模型建立出来又有什么用呢？因此，建模时要有明确的目的性，不要为了建模而建模，也不要事事都建模。当需要专注于建模并希望它产生效力时，要先分析从建模中是否能获得收益，或者说建模值不值得。事实上，项目越简单，建模发挥的功效越小。一般来说，通过建模，要达到 4 个目的：

① 模型帮助我们按照实际情况对系统进行可视化。

② 模型允许我们详细说明系统。

③ 模型给出了一个指导我们构造系统的模板。

④ 模型对我们做出的决策进行模板化。

建模并不是一个刚刚冒出来的新鲜事物，事实上，在各种传统的工程科学领域都有丰富的建模使用历史，这些经验形成了建模的一些基本原则。

(1) 要仔细的选择模型

创建什么样的模型对问题的解决有着重大的影响。正确的模型将清楚的阐明所要开发的系统，而错误的或是有偏差的模型将误导我们将精力放在不相关的问题上。

(2) 每一种模型可以在不同的精度级别上表示所要开发的系统

举个例子来说，修一栋大厦，有时需要使用 3D 软件制作出大厦的整体视觉图，供投资者参考；有时需要制作一份详细的电气施工图，供大厦的施工员铺设电线、光纤。由此看到，在项目开发中，不同的角色因为其观察角度的不同，对模型的侧重方面和详细程度有不同的要求。系统用户主要考虑“能做什么”的问题，开发人员主要考虑“如何做”的问题，这两类人从不同角度以不同的精度级别对系统进行可视化建模。

(3) 模型要与现实相联系

一个模型如果和现实相脱离，显然这不是一个好的模型。因此，要注意一点，虽然模型对现实进行了简化，但不能简化掉任何重要细节，也不能改变或歪曲任何重要细节。

(4) 对一个重要的系统用一组几乎独立的模型去处理

对一个复杂的或是重要的系统，只用单个模型去描述可能是不充分的，有时需要用多

种模型对系统分别进行研究和描述，以加深对系统的理解。

3. 使用 UML 建模

前面讲述了在工程领域里建模的重要性，那么，该如何建模呢？看看我们身边其他的领域：在音乐领域，有五线谱，供作曲家和演奏家交流；在数学领域，有各种各样的数学公式和表示方法，供数学家、教师、学生交流学习。同样，在工程领域，也有一种可以供工程开发设计人员使用的公共语言：UML。UML 的中文意思是统一建模语言(Unified Modeling Language)，它是一种通用的可视化建模语言。UML 使用灵活，表达能力强，并且非常实用。有了建模语言，就方便我们对各种工程进行描述和交流，就本书而言，论述的是如何使用 UML 在软件工程方面建模，所以下面将简单地介绍一下有关 UML 的知识，关于软件工程，将在下一节论述。

1.1.2　UML 简介

1. UML 历史

若要理解 UML，就有必要了解它的起源。从 20 世纪 80 年代末开始，出现了许多面向对象的软件建模技术，这些技术是由不同的人发明的，使用了不同的建模技术和模型表示法。但是，使用面向对象方法的用户并不了解不同建模语言之间的差异，因此很难根据应用特点选择合适的建模语言。直到 20 世纪 90 年代中期，有 3 种建模方法逐渐占据了统治地位，分别是 Jim Rumbaugh 的对象建模技术(OMT)、Ivar Jacobson 的面向对象软件工程方法(OOSE)和 Grady Booch 的 Booch 方法。1994 年，Rational 公司聘请了 Rumbaugh 参加 Booch 的工作。两人开始合并 OMT 和 Booch 方法中使用的概念，并于 1995 年提出了第一个建议方案。同年，Jacobson 也加入了 Rational 公司，3 位最优秀的面向对象方法学的的创始人终于聚在了一起，他们共同的成果就是统一建模语言。1997 年，Rational 公司正式将 UML 1.0 版作为标准草案提交给独立标准化组织 OMG(Object Management Group，对象管理组织)并获得通过。此后，OMG 承担了进一步完善 UML 标准的工作，并先后推出了 UML 的多个版本。

有了若干年使用 UML 的经验之后，OMG 提出了升级 UML 的建议方案，以修正使用中发现的问题，并扩充一部分应用领域中所需的额外功能。建议方案自 2000 年 11 月开始起草，至 2003 年 7 月完成。之后，UML 2.0 规范被全体 OMG 会员采纳并正式发布。总的来说，UML 2.0 和 UML 1.0 大部分是相同的，尤其是核心特征。

2. UML 简述

统一建模语言是一种通用的可视化建模语言，用于对软件进行描述、可视化处理、构造和建立软件系统的工作文档。它记录了与被构建系统的有关的决策和理解，可用于对系统的理解、设计、浏览、配置、维护以及控制系统的信息。UML 包括语义概念、表示法和指导规范，提供了静态、动态、系统环境及组织结构的模型。UML 能够捕捉系统静态结构和动态行为的信息。静态结构定义了系统中重要对象的属性和操作，以及这些对象之间的关系。动态行为定义了对象随时间变化的历史和对象为完成目标而进行的相互通信。UML 从不同但相互联系的角度对系统建模，因此可以全方位地理解系统。

UML 体系包括 3 个部分：UML 基本构造块、UML 规则和 UML 公共机制。当我们理解了这些内容，才能够读懂 UML 模型，并能自己动手建立一些基本的模型。

① UML 基本构造块

UML 有 3 种基本构造块，分别是事物、关系和图。事物包括结构事物、行为事物、分组事物、注释事物 4 种。关系包括依赖关系、关联关系、泛化关系、实现关系 4 种。图包括类图、对象图、用例图、顺序图、协作图、状态图、活动图、组件图、部署图 9 种。关于事物、关系和图这 3 种构造块及其各自的组成部分，本书稍后将有更详尽的描述。

② UML 规则

一个结构良好的模型在语义上应该是前后一致的，并且与所有的相关模型协调一致。因此，我们不能简单地把 UML 的构造块随机地摆放在一起而堆砌成一个模型。UML 定义了一套规则来告诉我们如何使用 UML 的构造块搭建出一个结构良好的模型。UML 有用于描述“命名”、“范围”、“可见性”、“完整性”、“执行”等事物的语义规则。

③ UML 的公共机制

在 UML 中有多种贯穿整个语言且一致应用的公共机制，因此使得 UML 变得较为简单。

UML 是复杂、庞大的。但我们不必知道或使用 UML 的每一项特征，就像不需要知道或使用一个大型项目的每一个功能一样。被广泛使用的核心概念只有一小部分，其他的特征可以逐步学习，在需要的时候再使用。

UML 合并了许多面向对象方法中被普遍接受的概念，对每一种概念，UML 都给出了清晰的定义、表示法和相关术语。这样，一个开发者可以用 UML 绘制一个模型，而另外一个开发者可以无歧义地解释这个模型。

UML 本质上不是一门编程语言。但是，人们可使用代码生成器将 UML 模型转换为多种程序设计语言代码，或使用反向生成工具将代码还原成 UML 模型。UML 也不是一种可以用于定理证明的高度形式化的语言。UML 本质上是一种通用的建模语言。

3. UML 的应用领域

UML 的目标是以面向对象的方式来描述任何类型的系统，其中最常用的是建立软件系统的模型。UML 的设计初衷是为了支持面向对象系统的建模，以及基于构件的开发。但是，在 UML 的设计中也考虑了其他需求，今天，通过使用 UML 内置的扩展和用户定制能力，UML 同样也可以用来描述非软件领域的系统，如机械系统、企业机构或业务过程，以及处理复杂数据的信息系统、具有实时要求的工业系统或工业过程等。总之，UML 是一个通用的标准建模语言，可以对任何具有静态结构和动态行为的系统进行建模。

UML 适用于系统开发过程中从需求规格描述到系统完成测试后的不同阶段。例如，在需求分析阶段，可以用用例来描述客户的需求；在设计阶段，可以用 UML 动态模型来描述对象与对象之间的关系；在测试阶段，UML 模型还可以作为测试的依据。

1.1.3 建模工具

Rose 是美国 Rational 公司推出的面向对象建模工具。利用这个工具，可以建立用 UML 描述的软件系统的模型，而且可以根据描述模型自动生成 C++、Java、VB 等程序代码。

1. 启动 Rose

启动 Rational Rose 2003 后，进入到主界面，首先弹出如图 1.1.1 所示的对话框。在这个对话框上有 New、Existing、Recent 3 个选项卡。第一个选项卡 New 用来选择新建

模型时所采用的模板。可供选择的模板有：J2EE(Java 2 企业级版本)，J2SE(Java 2 标准版本)的 1.2 版、1.3 版和 1.4 版，JDK(Java 开发工具包)的 1.16 版和 1.2 版，JFC(Java 基础类库)的 1.1 版，Oracle8-datatypes，Rational Unified Process(Rational 统一过程)，VB6 Standard(VB6 标准版)，VC6 ATL(VC6 活动模板库)的 3.0 版，VC6 MFC(VC6 基础类库)的 3.0 版。Existing 选项卡用于打开一个已经存在的模型文件，其中 Recent 选项卡中列出最近打开的模型文件列表。

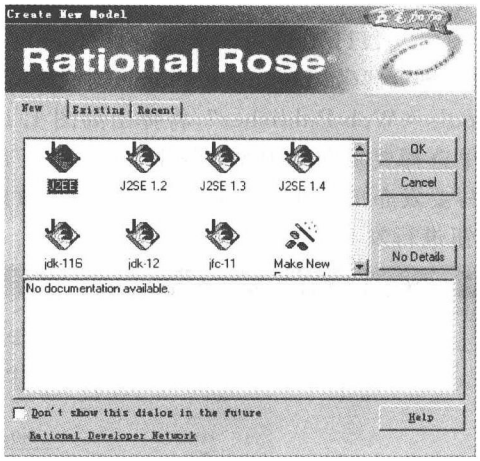


图 1.1.1

我们暂时不需要任何模板，只需要新建一个空白的模型，直接单击“Cancel”按钮，出现了默认的 Rational Rose 的主界面。它由标题栏、菜单栏、工具栏、工作区和状态栏组成，如图 1.1.2 所示。工作区分成 3 个部分：左边是树形视图和文档区，左边上面部分是树形视图，左边下面部分是文档区，每选中树形视图的某个对象，文档区就会显示其对应的文档名称；中间是编辑区，在该区中可以打开模型中的任意一张图，并可利用工具栏对图进行修改；中间下方是动作记录区，记录了对模型所做的动作。

图 1.1.2

2. 创建模型

Rose 模型文件的扩展名是. mdl，可通过以下步骤新建一个模型：

① 从菜单栏选择“File→New”。

② 弹出如图 1.1.1 所示的对话框，选择要使用的模板（如 J2EE、J2SE），单击“OK”按钮，则 Rose 自动装载这个模板的默认包、类和组件；或者不选择模板，单击“Cancel”按钮，则创建一个不使用默认模板的空模型，用户需要从头开始对系统建模。

3. 发布模型

Rose 可以把建立好的模型以 HTML 网页的形式发布，这样可以让其他即使没有装 Rose 软件的人员，也可以通过网页浏览器（如 Internet Explorer）浏览该模型。发布的步骤如下：

① 从菜单栏选择“Tools→Web Publisher”，将弹出如图 1.1.3 所示的对话框。

② 在弹出的对话框里选择要发布的模型视图和包，设定发布的细节内容。

③ 在黑线标记处填写要发布的 HTML 文件的名字和存放路径。

④ 单击“Publish”按钮进行发布。

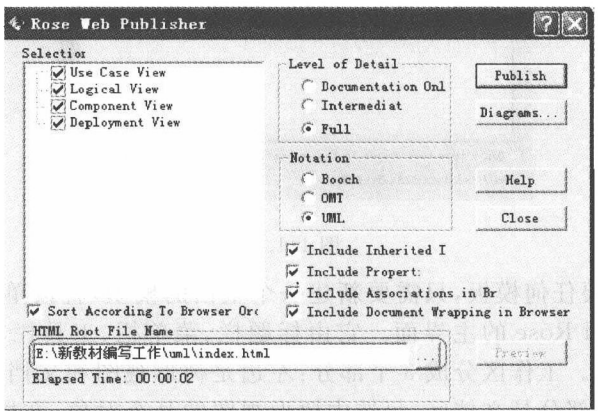


图 1.1.3

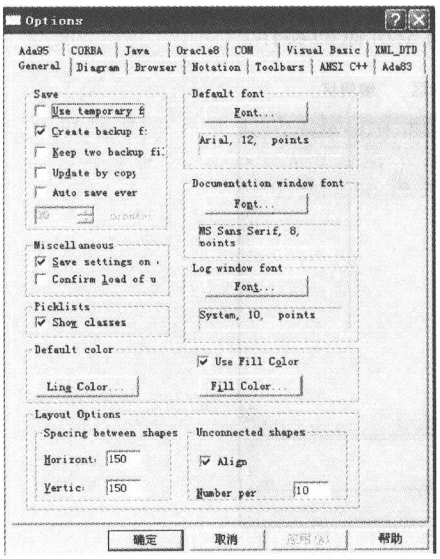


图 1.1.4

4. 设置全局属性

从菜单栏选择“Tools→Options”，将弹出如图 1.1.4 所示的对话框。在该对话框里可以设置一些全局属性，如字体、颜色，等等。

小　结

本节我们主要学习了以下知识：

1. 建模

（1）模型的概念

模型是对现实的简化。它可以是一个对象的微缩表示，是一种用于生产某事物的模式，也可以是一种设计或一个类型，还可以是一个待模仿或仿真的样例。

(2) 建模的目的和原则

建模是为了能够更好地理解我们正在开发的系统。

建模的原则包括:①要仔细地选择模型;②每一种模型可以在不同的精度级别上表示所要开发的系统;③模型要与现实相联系;④对一个重要的系统用一组几乎独立的模型去处理。

2. UML 概述

统一建模语言是一种通用的可视化建模语言,用于对软件进行描述、可视化处理、构造和建立软件系统的工作文档。UML 的组成部分包括 UML 基本构造块、UML 规则和 UML 公共机制。

3. UML 建模工具 Rational Rose

作　业

1. 什么是建模? 为什么要建模?
2. 模型必须是可视化的吗?
3. 缩写词 UML 的全称是什么?
4. UML 只适用于软件工程领域吗?
5. UML 包括哪些组成部分?

1.2　软件工程与 Rational 统一过程

内容提要

软件是包括程序、数据及其相关文档的完整集合,其开发过程至今尚未摆脱手工艺的开发方式,随着软件复杂度和开发难度的日益增加,逐渐形成了所谓的软件危机。为了解决软件危机,计算机科学家提出了软件工程的概念——使用工程化的原则和方法组织软件开发工作。其中重点介绍了软件的生存期和几种典型的软件生存期模型,最后介绍了 Rational 统一过程(Rational Unified Process,RUP)。本节主要内容如下:

- 软件
- 软件生命周期
- 软件生存期模型
- RUP

1.2.1　软件

在了解软件工程之前,我们首先需要知道什么是软件。一些人认为,软件就是程序,就是代码,这是比较片面的看法。软件一词最早是 20 世纪 60 年代初从国外传来的,当时很多人都无法说清它的确切含义。现在大家都比较认可的一种解释是:软件是计算机系

统中与硬件相互依存的另一部分，它是包括程序、数据及其相关文档的完整集合。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，数据是使得程序能够适当地操作信息的数据结构，文档是描述程序的开发、操作和维护的文字或图形资料。

要理解软件，以及最终理解软件工程，先了解软件的特征是重要的，据此你能够理解软件与人类建造的其他事物之间的区别。软件是一种逻辑实体，而不是具体的物理实体，因此，它具有与硬件完全不同的特征：

(1) 软件是被开发或设计的，而不是被制造的

虽然软件开发和硬件制造之间有一些相似之处，例如都可以通过良好的设计得到高质量，但两类活动在本质上是不同的。软件的开发过程中没有明显的制造过程，因此，硬件在制造过程中可以进行质量控制，而这种情况对软件而言几乎不存在的。如果要对软件进行质量控制，就必须在软件开发和测试方面下工夫。另外，就软件成本集中开发的特点而言，软件项目也不能像制造项目那样管理。

(2) 软件不会“磨损”，但会“退化”

硬件在运行和使用期间，会有机器磨损、老化等问题。例如，硬件在其生命初期有较高的故障率，这些故障主要是设计或制造的缺陷；在中期，随着缺陷的修正，故障率会维持在一个较低的水平上；到了后期，故障率又提升了，这是因为硬件已经进入了“磨损期”。而软件的情况不同，它不存在磨损和老化问题，但它存在着退化问题。在软件的生存期中，软件会经历多次修改或维护，使它能够适应软、硬环境的变化以及用户的新要求，但每次修改都可能会引入新的错误，这样一次次修改，导致软件故障率提高，从而使软件退化。

另外，当一个硬件零件磨损的时候，可以用另外一个来替换它，但软件就没有备用零件可换。每一个软件故障都表明了设计或者是编码中存在着错误。因此，软件维护比硬件维护复杂得多。

(3) 软件的开发至今尚未摆脱手工艺的开发方式

考虑这样一个案例：硬件设计工程师设计一个简单的数字电路图，其中的每一个集成电路都有一个零件编号，都有统一的制造工艺标准。每选定一个零件，我们都可以在货架上很方便地买到。硬件设计工程师可以专注于设计中的创新部分，而集成电路等零件是已经标准化的部件，可以不断复用，不需要硬件设计工程师再重新设计。类似地，一个软件设计师设计一个简单的应用系统，它包含多个子系统，我们可以把子系统想象成硬件世界中的“零件”，但这些“零件”不是标准化的工业产品，也不能在任何的货架上买到。因此，软件设计师完成了这个简单的应用系统的设计工作之后，还必须花额外的工夫去一一实现每个子系统。

在硬件世界，部件复用是工程过程的自然的一部分，但在软件世界，一切才刚刚起步。近年来，软件技术虽然提出了许多新的开发方法，例如，复用技术、自动生成技术，但在整个软件项目中采用的比例仍然比较低。大多数软件是根据客户需求订做的，而不是利用现成的部件组装成所需要的软件。由于在软件开发过程中，手工艺开发方式占主流，所以开发的效率自然比较低。

(4) 软件是复杂的

软件的研制工作必须要投入大量的高强度的脑力劳动，所以有人认为，人类创造的最

复杂的产物是计算机软件。软件的复杂性可能来自它所反映的实际问题的复杂性，例如，它所反映的自然规律或是人类社会的各种活动，都具有一定的复杂性；另一方面，软件的复杂性也可能来自程序逻辑结构的复杂性，例如，一个系统软件要能处理各种可能出现的情况。软件开发常常涉及其他领域的专业知识，这对软件人员提出了很高的要求。现阶段，软件技术的发展落后于复杂的软件需求，这确实是个很现实的问题。

以上讨论的是软件的特征。那么软件究竟有哪些类型呢？在某种程度上我们难以对软件给出一个通用的分类，但鉴于不同类型的工程对象，对其进行开发和维护有不同的处理方法，因此对软件的类型仍需进行必要的划分。下面依照软件的不同功能对软件进行了划分，需要注意的是，划分的方式不是唯一的，从不同的角度出发，可以做出不同的软件划分方式。

• 系统软件

系统软件指能与计算机硬件紧密结合在一起，使计算机系统各个部件、相关软件和数据协调高效地工作的软件。如操作系统软件、数据库管理软件、通信处理软件等。

• 支撑软件

支撑软件是指协助用户开发软件的工具性软件，包括帮助程序员开发软件产品的工具，也包括帮助管理人员控制开发的进程的工具。如 Java 开发工具 Eclipse 等。

• 应用软件

应用软件是指在特定领域内开发，为特定目的服务的一类软件。应用软件的种类非常繁多，如计算机辅助设计制造软件、系统仿真软件、人工智能软件、办公自动化软件、计算机辅助教学软件等。

1.2.2　软件危机

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是软件开发低级阶段的表现。随着计算机硬件技术的进步，生产硬件的成本降低了，在这一形势下，要求软件能与之相适应，所以，提出了一些复杂大型的软件项目。但是，软件技术的发展落后于复杂的软件需求，随着问题的日积月累，在 20 世纪 60 年代逐渐形成了所谓的软件危机。软件开发中出现的问题归结如下：

① 软件开发无计划性，进度的执行和实际情况有很大差距。

② 软件需求分析阶段工作做得不充分，前期问题不及时解决，造成后期矛盾的集中暴露。

③ 软件开发过程中没有统一的规范指导，参与软件开发的人员各行其事。

④ 软件产品无评测手段。

1.2.3　软件工程

从上述软件危机的现象中可以看出，摆脱危机不是一件简单的事情。我们如何开发软件、如何维护大量已有的软件以及开发速度如何跟上对软件越来越多的需求，等等，确实是一大堆苦恼的问题。经过许多计算机软件科学家的实践和总结，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，也是摆脱软件危机的一个主要出路。这

里就引出了软件工程的概念。软件工程指:将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护,即将工程化应用于软件开发。

如果不考虑应用领域、项目规模和复杂性,与软件工程相关的工作一般可分为 3 个阶段:

① 定义阶段

该阶段关注于"做什么"。软件开发人员在该阶段需要弄清楚要处理什么信息,完成什么样的功能,达到什么样的性能,希望出现什么样的系统行为,建立什么样的界面,有什么设计约束,以及实现一个成功系统的验收标准是什么。

② 开发阶段

该阶段关注于"如何做"。软件开发人员需要进行如下工作:数据如何被结构化,功能如何被实现于软件体系结构中,界面如何表示,设计如何被翻译成程序设计语言,测试如何进行。

③ 支持阶段

该阶段关注于"变化"。在已有软件的基础上,软件开发人员需要纠正软件中可能存在的错误,为适应外部环境的变化而修改软件,为扩展原来的功能需求而增强和升级软件。

如同任何事物一样,软件也有一个孕育、诞生、成长、成熟、衰落、死亡的生存过程,这称之为软件的生存期。根据这一思想,将上述 3 个阶段软件过程活动进一步地展开,就得到了软件生存期的 6 个步骤。

① 计划

确定要开发软件系统的总目标。由系统分析员和客户人员合作,进行该软件项目的可行性研究,探讨解决问题的可能方案,对开发成本、开发收益、开发进度做出估计,最后制订出详细的软件开发实施计划。

② 需求分析和定义

对客户提出的需求进行详细的分析。系统分析员和客户人员共同商讨哪些需求是可以满足的,并对其进行准确的描述,最后编写出项目需求说明书。

③ 软件设计(详细设计)

在设计阶段,系统分析员和软件设计人员一起把已经确定了的各个需求转换成一个相应的体系结构。结构中的每一组成部分都是意义明确的模块,每个模块和某些需求相对应。该阶段的成果物是设计说明书,该说明书对每个模块进行了详细的描述。

④ 编码

软件设计人员和程序员一起将软件设计转换成程序代码。该阶段的成果物是程序清单。

⑤ 软件测试

软件测试人员设计各种测试用例来检验软件。测试是保证软件质量的重要手段。该阶段的成果物是测试用例书、测试数据和测试结果。

⑥ 运行和维护

已交付的软件投入正式使用后,就进入维护阶段。软件在运行过程中可能由于多方面的原因,需要对它进行修改。维护的工作可能包括:在软件的运行过程中发现了错误需要修正;给软件做适当变更以适应变化了的软件工作环境。

为了反映软件生存期内各种活动应如何组织，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图。到现在为止，已经有了许多种软件生存期模型，下面只简要地介绍 2 种：

① 瀑布模型

该模型规定了计划、需求分析、设计、编码、测试、维护这 6 个步骤自上而下、相互衔接的固定次序，如同瀑布流水，逐级下落。如图 1.2.1 所示。

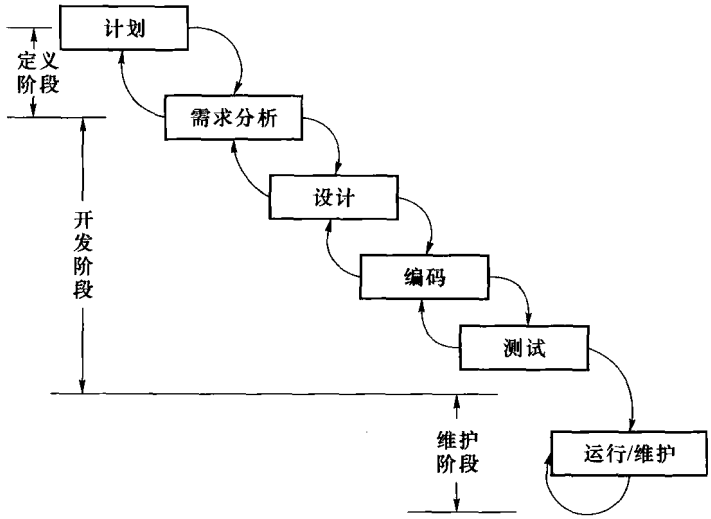


图 1.2.1

瀑布模型中的每一项开发活动有如下的特征：每一项活动的输出（工作成果）都是下一项的输入（除了“运行/维护”活动），在输出的时候要对该项活动实施的工作进行评估，若其工作得到确认则继续进行下一项活动（见图 1.2.1 中向下指的箭头），若工作不能通过评估，则返回前一项目，甚至更前项的活动进行返工（见图 1.2.1 中向上指的箭头）。

瀑布模型的缺陷是缺乏灵活性，特别是无法解决软件需求不明确或不准确的问题。

② 原型实现模型

考虑这样的情况，客户定义了软件的一般性目标，但不能标识出详细的输入、输出及处理要求，或者，软件开发人员不能确定算法的有效性或人机交互的形式，这时，由于人们对软件的认识不够清晰，因此项目无法做到一次开发成功，出现返工在所难免。在这种情况下，可以考虑用原型实现模型方式来开发软件项目。第一次的开发只是试验开发，其目的在于探索可行性，弄清软件需求。第一次得到的试验性产品称为“原型”。以“原型”为基础，再进行第二次、第三次的开发，逐步调整原型使其满足客户的要求，同时也使得软件开发人员对该软件的实现方法有更深入的理解。这是一个不断完善“原型”，不断迭代的过程，直到最后得到一个客户满意的模型为止。如图 1.2.2 所示。

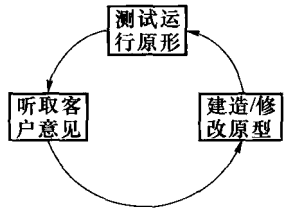


图 1.2.2

1.2.4 面向对象软件工程方法

面向对象技术是一个非常实用而强有力的软件开发方法，和传统的软件工程方法相比较，它有许多新特色。面向对象技术导致复用，而程序构件的复用导致更快的软件开发和高质量的程序。

为了讨论面向对象软件工程方法，首先必须明确什么是“面向对象”？一个比较统一的认识是：“面向对象＝对象＋类＋继承＋通信”。如果一个软件系统是使用这样的概念设计和实现的，就可以认为这个软件系统是面向对象的。

• 对象

对象是面向对象开发模式的基本组成部分。对象是一种看问题的观点，是对现实世界各种真实事物的一种抽象。一个对象是一组属性和一组操作的集合。属性描述了该对象区别于其他对象的一些重要特征。属性一般只能通过执行对象的操作来改变。操作又称为方法，在C＋＋中称为成员函数，它描述了对象所具有的行为，或者可以执行的功能。对象既含数据又含处理数据的功能，因此，对象被认为是迄今最接近真实事物的数据抽象。

• 类

类是具有相同属性、相同操作的一组对象的集合的抽象描述。类的定义包括一组数据属性和在数据上的一组合法操作。类定义可以视为一个具有类似特征与共同行为的模板，用来产生对象，因此，每个对象都是类的实例。

• 继承

继承是使用已存在的定义作为基础建立新定义的技术。一个子类 X 继承父类 Y 的所有属性和操作，这意味着，所有原本针对 Y 设计和实现的数据结构和算法对 X 是立即可用的，不需要进行进一步的工作。

• 通信

一个对象和另一个对象之间，通过消息来进行通信。消息是一个对象发出的让另一个对象做某个动作的请求。当一个对象收到发给自己的消息的时候，则调用消息中指定的操作。对消息的处理可能会改变对象的状态。可以认为，消息的传递大致等价于面向过程方法中的函数调用。

下面简要地介绍面向对象的应用开发过程：

① 分析阶段

在分析中，需要找到特定对象，根据对象的公共特征把它们组成集合，直到最后能够标识出对这个问题的一个抽象为止。另外，还要标识出应用系统的结构中对象之间的联系。

② 高层设计

在这一阶段，应该设计应用的顶层视图，这相当于开发一个表示系统的类的界面。

③ 类的开发

应用设计阶段基本上是类的开发，一个应用可以抽象成用一个类表示，也可以分成几个类。这一阶段将标识对各个类的要求，并给出类的定义。