现代软件工程专业系列教材

UML及建模

UML JI JIANMO

郭　宁　编著

清华大学出版社
http://www.tup.tsinghua.edu.cn
北京交通大学出版社
http://press.bjtu.edu.cn

现代软件工程专业系列教材

UML及建模

郭 宁 编著

清华大学出版社
北京交通大学出版社
·北京·

内 容 简 介

本书系统地介绍了面向对象的基本思想、主要概念，重点介绍了统一建模语言 UML 及其应用。全书内容丰富，除了介绍 UML 的用例图、顺序图、协作图、类图、状态图、活动图、组件图和部署图等图中涉及的术语、规则和应用外，还介绍了数据建模、对象约束语言、Web 建模、Rose 开发工具的主要用法等。本书注重理论与实践相结合，通过给出大量的例题、较为详尽的实例分析及对建模概念的运用，阐述了如何运用面向对象的技术建立软件系统模型的方法。

本书可作为大专院校计算机软件专业研究生和高年级本科生学习 UML 与面向对象技术的教材，也可作为广大软件开发人员自学 UML 与面向对象技术的参考书。

图书在版编目（CIP）数据

UML 及建模/郭宁编著. —北京：清华大学出版社；北京交通大学出版社，2007.1
（现代软件工程专业系列教材）
ISBN 978－7－81082－900－7

Ⅰ. U…　Ⅱ. 郭…　Ⅲ. 面向对象语言，UML－程序设计－教材　Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2006）第 130045 号

责任编辑：招富刚
出版发行：清 华 大 学 出 版 社　邮编：100084　电话：010－62776969
　　北京交通大学出版社　邮编：100044　电话：010－51686414
印 刷 者：北京鑫海金澳胶印有限公司
经　　销：全国新华书店
开　　本：203×280　印张：19.25　字数：544 千字
版　　次：2007 年 1 月第 1 版　2007 年 1 月第 1 次印刷
书　　号：ISBN 978－7－81082－900－7/TP・311
印　　数：1～5 000 册　定价：29.00 元

前言

面向对象技术以其显著的优势成为计算机软件领域的主流技术。产业界需要大量掌握面向对象方法和技术的人才。这些人才不仅能够使用面向对象编程语言来编程，更重要的是能运用面向对象方法进行系统建模，即通过面向对象的分析（OOA）和面向对象的设计（OOD）建立系统的分析模型和设计模型。面向对象的统一建模语言 UML 是一种定义良好、易于表达、功能强大且适用于各种应用领域的建模语言，已被 OMG 采纳为标准，目前 UML 已成为面向对象技术领域内占主导地位的标准建模语言。掌握 UML 语言，不仅有助于理解面向对象的分析与设计方法，也有助于对软件开发全过程的理解。

本书通过介绍 UML 和软件建模技术，可以使读者了解面向对象技术的基本概念，掌握面向对象的分析和设计方法、建立模型的过程及面向对象技术相关的一些软件开发技术。

本书包括 13 章的内容。第 1 章面向对象技术概述，主要是对软件工程中一些知识点的回顾。第 2 章软件建模和软件开发过程，是让读者对软件模型的概念和软件模型开发过程有一个总体的认识。第 3 章统一建模语言简介，是通过对 UML 的概述，对 UML 的体系结构、模型元素、模型结构和建模规则有一个总体的了解，并通过一个实际例子对 UML 中的主要图形模型有一个总体的印象。用例是 UML 中一个比较重要的概念，用例驱动的软件开发方法已得到广泛的认同，第 4 章用例建模，全面细致地介绍了用例模型的基本概念和建立方法。第 5 章结构建模，主要介绍了面向对象的系统分析的基本思想、过程和结构模型的基本概念，详细介绍了类图的基本概念、功能和识别对象类、属性、操作及关联的方法与技术，并介绍了接口、包、构造型等相关概念。第 6 章行为建模，介绍了行为模型、消息、交互机、状态和活动等基本概念，重点介绍了交互模型、状态模型的组成元素、建立方法和技术。第 7 章软件系统体系架构建模，对面向对象的系统设计进行了总体介绍，详细介绍了逻辑体系架构建模设计的概念、内容和典型的系统架构；并介绍了系统物理架构模型的组成，组件图和部署图的绘制方法；还介绍了面向对象约束语言的基本概念与用法。第 8 章系统详细设计，介绍了面向对象的设计原则、详细设计的方法、人机界面设计和数据建模等内容。第 9 章面向对象软件实现介绍了面向对象程序设计语言的技术特点、选择方法、程序设计风格和实现策略等内容，还简要介绍了面向对象软件测试的基本内容和方法。第 10 章 Web 建模主要介绍了 Web 应用系统的特征和体系结构，介绍了如何用 UML 中的扩展机制对 Web 应用系统建模。第 11 章软件设计模式，主要介绍了软件模式在软件设计过程中的作用，为了使读者对模式有进一步的了解，重点介绍了 Facade、Adapter、Abstract Factory 和 Observer 四个模式。第 12 章面向对象的软件建模工具，介绍了一些软件开发工具和 Rose 的使用方法。第 13 章介绍了一个手机游戏软件应用面向对象技术建立模型的开发过程。

本书是作者多年来进行软件系统开发实践的一次经验总结。教材中诸多实际问题或应用实例，有许多就取材于开发实践，并都按照教学的需要进行了模型简化。显然，这些源于实践的工程问题，对提高软件系统教学的实践性与实用性，将具有很好的示范效应。

作为一本实用的面向对象方法教材和工程技术用书，既要注重概念与表达内容的完整性，又要避免过分复杂。本书的最大特点是结合了丰富的实例，从而使对理论的介绍生动而不抽象，同时通过实例启发读者更好地将所学到的面向对象技术应用于软件系统的分析、设计与开发中。本

书不但可以作为大专院校课程教材，也可以作为软件开发人员的参考手册。

在本书的编写过程中，我们参阅了大量的资料，在此对所有编著者表示衷心的感谢。另外，李捷思、李志秀、张力东负责书稿的录入工作，郭林、田建勇、刘杰负责图表绘制、文稿编辑等工作，王广春、宋立东对本书提出了很多宝贵意见，在此一并表示感谢。在写作中，作者对书中的内容反复多次修改，以求尽量减少错误，但由于水平有限，难免会有各种错误和疏漏，敬请广大读者批评指正。

作者

2006 年 10 月于北京

目　录

第1章 面向对象技术概述

本章学习目标

- 掌握软件工程的基本概念
- 熟悉面向对象技术的发展历史及现状
- 掌握面向对象软件开发方法的特点
- 掌握面向对象的基本概念

面向对象技术使人本精神在计算机这个机器世界中获得了更加有效的发挥，并在面向用户的软件系统开发中表现出了显著的优势，成为了这个领域的主流技术。正是面向对象软件技术的广泛应用，推动了面向对象软件工程方法学的发展。本章简要地回顾了软件危机及软件工程的基本概念，并通过对软件开发的基本认识，引出面向对象方法的主要特点，重点介绍了面向对象技术的基本概念。

1.1 软件危机及软件工程

随着计算机技术的发展和应用领域的扩大，软件规模也越来越大，复杂程度不断增加，使得软件生产的质量、周期、成本难以预测和控制，出现了软件危机。软件工程正是为了解决软件危机而提出的，其目的是改善软件生产的质量，提高软件的生产效率。

1.1.1 软件及其特性

软件是计算机系统的思维中枢，是软件产业的核心。作为信息技术的灵魂——计算机软件，在现代社会中起着极其重要的作用。

1. 软件

计算机软件是由计算机程序的发展而形成的一个概念。它是与计算机系统操作有关的程序、规程、规则及其文档和数据的统称。软件由两部分组成：一是机器可执行的程序和有关的数据；二是与软件开发、运行、维护、使用和培训有关的文档。

程序是按事先设计的功能和性能要求执行的语句序列。数据是程序所处理信息的数据结构。

文档则是与程序开发、维护和使用相关的各种图文资料，例如，各种规范说明书、设计说明书、用户手册等。在文档中记录着软件开发的活动和阶段成果。

2. 软件的特点

软件是一种逻辑产品而不是实物产品，软件功能的发挥依赖于硬件和软件的运行环境，没有计算机相关硬件的支持，软件毫无实用价值。若要对软件有一个全面而正确的理解，应从软件的本质、软件的生产等方面剖析软件的特征。

（1）软件固有的特性

① 复杂性。软件是一个庞大的逻辑系统，比人类构造的其他产品都要复杂。一方面在软件中要客观地体现人类社会的事务，反映业务流程的自然规律；另一方面在软件中还要集成多种多样的功能，以满足用户对大量信息及时处理、传输、存储等方面的需求，这就使得软件变得十分复杂。

② 抽象性。软件是人们经过大脑思维后加工出来的产品，一般寄生在内存、磁盘、光盘等载体上，人们无法观察到它的具体形态，这就导致了软件开发不仅工作量难以估计，进度难以控制，而且质量也难以把握。

③ 依赖性。软件必须和运行软件的机器（硬件）保持一致，软件的开发和运行往往受到计算机硬件的限制，对计算机系统有着不同程度的依赖性。软件与计算机硬件的这种密切相关性与依赖性，是一般产品所没有的特性。为了减少这种依赖性，在软件开发中提出了软件的可移植性问题。

④ 软件的使用特性。软件的价值在于应用。软件产品不会因多次反复使用而磨损老化，一个久经考验的优质软件，可以长期使用。由于用户在选择新机型时，通常提出兼容性要求，所以一个成熟的软件可以在不同型号的计算机上运行。

（2）软件的生产特性

① 软件的开发特性。由于软件固有的特性，使得软件的开发不仅具有技术复杂性，还有管理复杂性。技术复杂性体现在软件提供的功能比一般硬件产品提供的功能多，而且功能的实现具有多样性，需要在各种实现中作出选择，更有实现算法上的优化带来的不同，而实现上的差异会带来使用上的差别。管理上的复杂性表现在：第一，软件产品的能见度低（包括如何使用文档表示的概念能见度），要看到软件开发进度比看到有形产品的进度困难得多；第二，软件结构的合理性差，结构不合理使软件管理复杂性随软件规模增大而呈指数增长。因此，领导一个庞大人员的项目组织进行规模化生产并非易事，软件开发比硬件开发更依赖于开发人员的团队精神、智力和对开发人员的组织与管理。

② 软件产品形式的特性。软件产品的设计成本高昂而生产成本极低。硬件产品试制成功之后，批量生产需要建设生产线，投入大量的人力、物力和资金，生产过程中还要对产品进行质量控制，对每件产品进行严格的检验。然而，软件是把人的知识与技术转化为信息的逻辑产品，开发成功之后，只需对原版软件进行复制即可，大量人力、物力、资金的投入和质量控制、软件产品检验都是在软件开发中进行的。由于软件的复制非常容易，软件的知识产权保护就显得极为重要。

③ 软件的维护特性。软件在运行过程中的维护工作比硬件复杂得多。首先，软件投入运行后，总会存在缺陷甚至暴露出潜伏的错误，需要进行“纠错性维护”。其次，用户可能要求完善软件性能，对软件产品进行修改，进行“完善性维护”。当支撑软件产品运行的硬件或软件环境改变时，也需要对软件产品进行修改，进行“适应性维护”。软件的缺陷或错误属于逻辑性的，因此不需要更换某种备件，而是修改程序，纠正逻辑缺陷，改正错误，提高性能，增加适应性。当软件产品规模庞大、内部的逻辑关系复杂时，经常会发生纠正一个错误而产生新的错误的情

况，因此，软件产品的维护要比硬件产品的维护工作量大而且复杂。

1.1.2 软件危机

20世纪六七十年代，由于软件规模的扩大、功能的增强和复杂性的增加，使得在一定时间内仅依靠少数人开发一个软件变得越来越困难。在软件开发中经常会出现时间延迟、预算超支、质量得不到保证、移植性差等问题，甚至有的项目在耗费了大量人力、财力后，由于离目标相差甚远而宣布失败。这种情况使人们认识到“软件危机”的存在。

1. 软件危机的突出表现

① 软件生产率低。软件生产率提高的速度远远跟不上计算机应用迅速普及和深入的趋势。落后的生产方式与开发人员的匮乏，使得软件产品的供需差距不断扩大。由于缺乏系统有效的方法，现有的开发知识、经验和相关数据难以积累与重用，另外，低水平的重复开发过程浪费了大量的人力、物力、财力和时间。人们为不能充分发挥计算机硬件提供的巨大潜力而苦恼。

② 软件产品常常与用户要求不一致。开发人员与用户之间的信息交流往往不充分，经常存在二义性、遗漏，甚至是错误。由于开发人员对用户需求的理解与用户的本意有所差异，以致造成开发中后期需求与现实间的矛盾集中暴露。

③ 软件规模的增长，带来了复杂度的增加。由于缺乏有效的软件开发方法和工具的支持，过分依靠程序开发人员在软件开发过程中的技巧和创造性，软件的可靠性往往随着软件规模的增长而下降，质量保障越来越困难。

④ 不可维护性突出。软件的局限性和欠灵活性，不仅使错误非常难改正，而且不能适应新的硬件环境，也很难根据需要增加一些新的功能，整个软件维护过程除了程序之外，没有适当的文档资料可供参考。

⑤ 软件文档不完整、不一致。软件文档是计算机软件的重要组成部分。在开发过程中，管理人员需要使用这些文档资料来管理软件项目；技术人员则需要利用文档资料进行信息交流；用户也需要通过文档来认识软件，对软件进行验收。但是，由于软件项目管理工作的不规范，软件文档往往不完整、不一致，这给软件的开发、交流、管理、维护等都带来了困难。

2. 产生软件危机的原因

软件危机是指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。产生软件危机的原因主要有以下几点。

① 软件独有的特点给开发和维护带来困难。由于软件的抽象性、复杂性与不可见性，使得软件在运行之前，开发过程的进展情况较难衡量，软件的错误发现较晚，软件的质量也较难评价，因此，管理和控制软件开发过程相当困难。此外，软件错误具有隐蔽性，往往在很长时间内软件仍可能需要改错。这在客观上使得软件维护较为困难。

② 软件人员的错误认识。相当多的软件专业人员对软件开发和维护还有不少的错误观念。例如，软件开发就是编写程序，忽视软件需求分析的重要性，轻视文档的作用，轻视软件维护等。这些错误认识加重了软件危机的影响。

③ 软件生产技术进步缓慢。长期以来软件开发一直沿用的是“手工作坊”的方式，这样不仅浪费了大量的财力、物力和宝贵的人力资源，无法避免低水平的重复性劳动，而且软件的质量也难于保证。此外，软件生产工程化管理程度低，致使软件项目管理混乱，难以保障软件项目成本和开发进度按计划执行。

④ 软件开发工具自动化程度低。尽管软件开发工具比30年前已经有了很大的进步，但直到今天，软件开发仍然离不开工程人员的个人创造与手工操作，软件生产仍不可能向硬件设备的生

产那样，达到高度自动化。

1.1.3 软件工程

为了克服“软件危机”，1968 年在北大西洋公约组织（NATO）召开的计算机科学会议上，FritzBauer 首先提出“软件工程”的概念，试图用工程的方法和管理手段，将软件开发纳入工程化的轨道，以便开发出成本低、功能强、可靠性高的软件产品。几十年来，人们一直在努力探索克服软件危机的途径。

1. 软件工程的形成与发展

自 NATO 会议上提出软件工程这一概念以来，人们一直在寻求更先进的软件开发方法与技术。当出现一种先进的方法与技术时，就会使软件危机得到一定程度的缓解。然而，这种进步又促使人们把更多、更复杂的问题交给计算机去解决，于是又需要探索更先进的方法与技术。软件工程的发展经历了以下三个阶段。

第一阶段：20 世纪 70 年代，为了解决软件项目失败率高、错误率高及软件维护任务重等问题，人们提出了软件生产工程化的思想，希望使软件生产走上正规化的道路，并努力克服软件危机。人们发现将传统工程学的原理、技术和方法应用于软件开发，可以起到使软件生产规范化的作用。它有利于组织软件生产、提高开发质量、降低成本和控制进度。随后，人们又提出了软件生命周期的概念，将软件开发过程划分为不同阶段（需求分析、概要与详细设计、编程、测试和维护等），以适应更加复杂的应用。人们还将计算机科学和数学用于构造模型与算法上，围绕软件项目开展了有关开发模型、方法及支持工具的研究，并提出了多种开发模型、方法与多种软件开发工具（编辑、编译、跟踪、排错、源程序分析、反汇编、反编译等）；并围绕项目管理提出了费用估算、文档评审等一些管理方法和工具，基本形成了软件工程的概念、框架、方法和手段，成为软件工程的第一代——传统软件工程时代。

第二阶段：20 世纪 80 年代，面向对象的方法与技术受到了广泛的重视，Smalltalk-80 的出现标志着面向对象的程序设计进入实用和成熟阶段。20 世纪 80 年代末逐步发展起来的面向对象的分析与设计方法，形成了完整的面向对象技术体系，使系统的生命周期更长，适应更大规模、更广泛的应用。这时，进一步提高软件生产率、保证软件质量就成为软件工程追求的更高目标。软件生产开始进入以过程为中心的第二阶段。这个时期人们认识到，应从软件生命周期的总费用及总价值来决定软件开发方案。在重视发展软件开发技术的同时，人们提出软件能力成熟度模型、个体软件过程和群组软件过程等概念。在软件定量研究方面提出了软件工作量估计 COCOMO 模型等。软件开发过程从目标管理转向过程管理，形成了软件工程的第二代——过程软件工程时代。

第三阶段：进入 20 世纪 90 年代以后，软件开发技术的主要处理对象为网络计算和支持多媒体信息的 WWW。为了适合超企业规模、资源共享、群组协同工作的需要，企业需要开发大量的分布式处理系统。这一时期软件工程的目的在于不仅提高个人生产率，而且通过支持跨地区、跨部门、跨时空的群组共享信息，协同工作来提高群组、集团的整体生产效率。因整体性软件系统难以更改、难以适应变化，所以提倡基于组件的开发方法——即部件互连及集成。同时人们认识到计算机软件开发领域的特殊性，不仅要重视软件开发方法和技术的研究，更要重视总结和发展包括软件体系结构、软件设计模式、互操作性、标准化、协议等领域的重用经验。软件重用和软件组件技术正逐步成为主流软件技术，软件工程也由此进入了新的发展阶段——组件软件工程时代。

2. 软件工程的基本概念

作为一门新兴的交叉性学科，软件工程所研究的对象、适用范围和所包含的内容都在不断发展和变化。

1）软件工程的定义

在NATO会议上，软件工程被定义为："为了经济地获得可靠的和能在实际机器上高效运行的软件，而建立和使用的健全的工程原则。"这个定义虽然没有提到软件质量的技术层面，也没有直接谈到用户满意程度或要求按时交付产品等问题，但人们已经认识到借鉴和吸收人类对各种工程项目开发的经验无疑对软件的开发是有益的。

软件工程是指导软件开发和维护的工程学科。它强调按照软件产品的生产特性，采用工程的概念、原理、技术和方法来开发与维护软件，把经过时间考验而证明是正确的管理技术和当前最好的技术结合起来，以经济地开发出高质量的软件并有效地维护它。

由于引入了软件工程的思想，把其他工程技术研究和开发领域中行之有效的知识和方法运用到软件开发工作中来，提出了按工程化的原则和方法组织软件开发工作的解决思路和具体方法，在一定程度上缓解了"软件危机"。

2）软件工程的目标

软件工程的目标是基于软件项目目标的成功实现而提出的，主要体现在以下几方面：

① 软件开发成本较低；

② 软件功能能够满足用户的需求；

③ 软件性能较好；

④ 软件可靠性高；

⑤ 软件易于使用、维护和移植；

⑥ 能按时完成开发任务，并及时交付使用。

在实际开发中，企图让以上几个质量目标同时达到理想的程度往往是不现实的。软件工程目标之间存在的相互关系如图1-1所示。

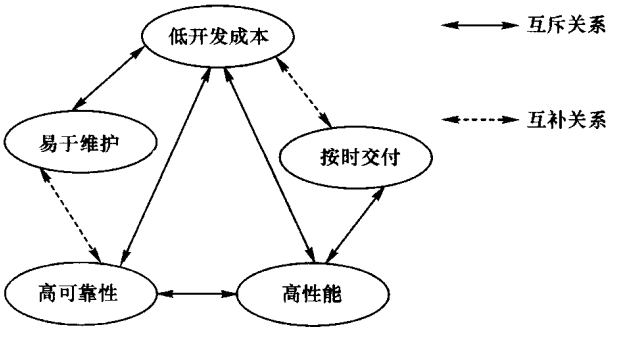

图1-1 软件工程目标之间的关系

从图中可以看出，有些目标之间是相互补充的，例如易于维护和高可靠性之间、功能强与可用性之间；有些目标是彼此相互冲突的，例如若只考虑降低开发成本，很可能同时也降低了软件的可靠性，如果一味追求提高软件的性能，可能造成开发出的软件对硬件的依赖较大，从而影响到软件的可移植性；不同的应用对软件质量的要求不同，例如，对实时系统来说，其可靠性和效率比较重要；对生命周期较长的软件来说，其可移植性、可维护性比较重要。

3）软件工程学研究范畴

软件工程学是研究软件开发过程、开发方法、工程开发技术和工具，指导软件生产和管理的