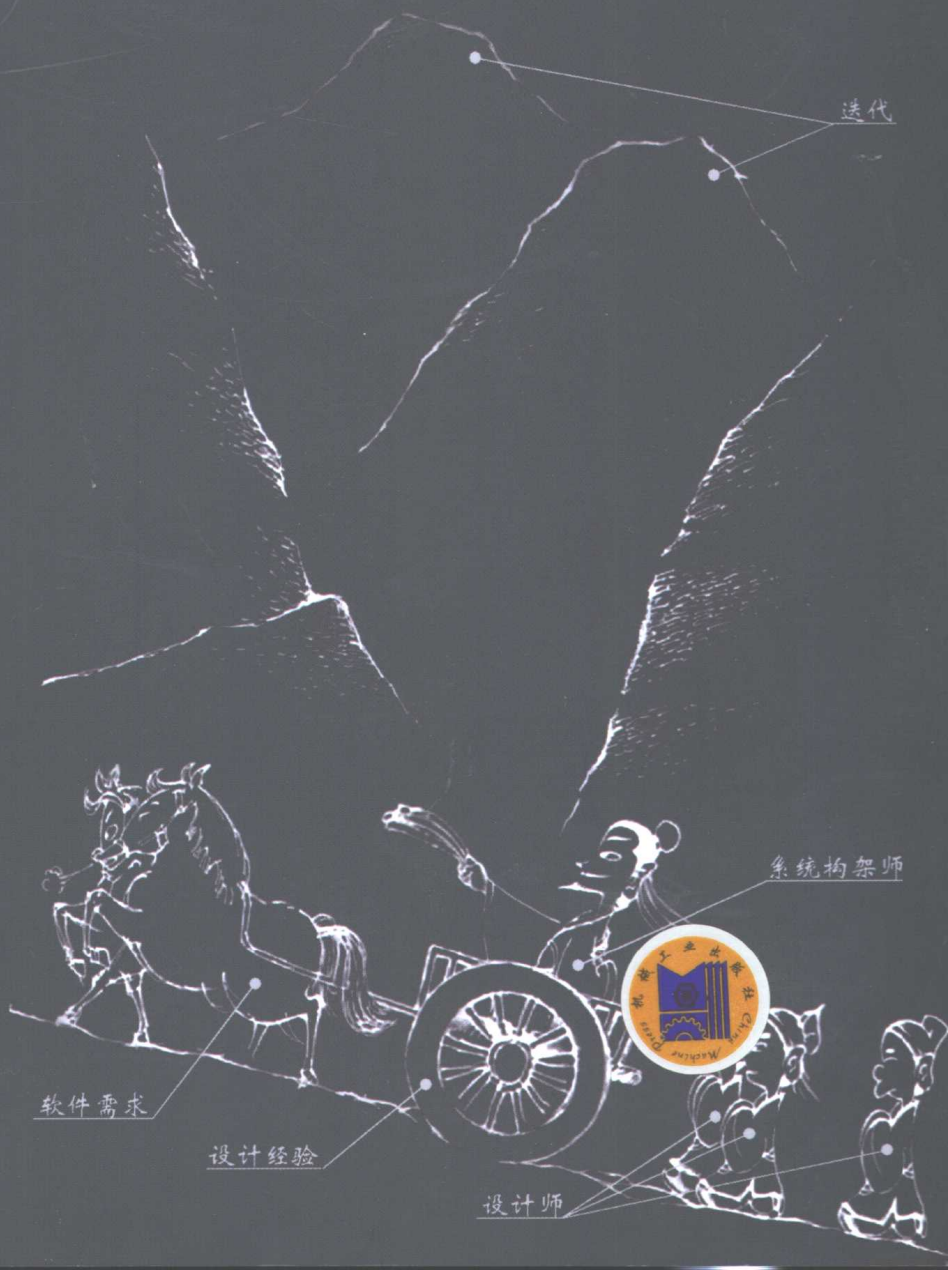
UML应用建模实践过程

尤克滨 编著

立足实践者视角

遵循RUP原则

贯穿全过程示例



机械工业出版社
CHINA MACHINE PRESS

UML 应用建模实践过程

尤克滨　编著

机 械 工 业 出 版 社

本书立足工程实践，以应用 UML 进行面向对象分析和设计为主题，帮助软件工程师在排除关键障碍的基础上，通过推敲实例，有步骤地掌握一套切实可行的方法和流程。

全书分为三个部分。第一部分，基本理念和准备知识。是本书的铺垫。解释分析和设计模型的含义和价值，概述面向对象技术的内涵、优势和原则，介绍模型内容的组织和相关的 UML 表述。第二部分，UML 应用建模实践过程。是本书的核心。详细展现分析和设计过程中的 5 项任务，即全局分析、局部分析、全局设计、局部设计和细节设计。其中包括 14 项基础活动、39 个核心概念、30 个关键步骤、52 条实践技巧以及贯穿全程的示例。本书的实践过程遵循 Rational 统一过程(RUP)的核心思想和基本原则，即以 Use Case 驱动的、体系构架为核心的迭代化面向对象分析和设计过程。第三部分，设计模型的沿用。是本书内容的延伸。概要地介绍与设计模型直接相关的活动和内容，包括设计模型向实施模型的过渡、设计模型和数据模型的关联以及如何整理主要的设计文档。

本书立足实践者的视角，适合于应用面向对象技术的软件工程师，尤其是系统构架师和设计师。本书可以作为应用 UML 进行面向对象分析和设计的实践课程教材。

图书在版编目（CIP）数据

UML 应用建模实践过程/尤克滨编著. —北京：机械工业出版社，2003.1
ISBN 7-111-11436-1

Ⅰ. U… Ⅱ. 尤… 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（2002）第 109104 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：胡毓坚
责任编辑：陈振虹
封面设计：于　鹏　王从然
责任印制：闫　焱

北京交通印务实业公司印刷·新华书店北京发行所发行

2003 年 1 月第 1 版第 1 次印刷
787mm×1092mm　1/16·14 印张·343 千字
0001-5000 册
定价：28.00 元

前　言

目　标

如果你只热衷于作“程序员”而无意成为“软件工程师”，这本书并不适合你。如果你想通过本书成为面向对象技术专家，坦率讲，也没有可能。但是，如果你打算应用统一建模语言（Unified Modeling Language，UML）、采用面向对象技术分析和设计软件，这本立足于工程实践的书能够给你实实在在的帮助。

本书以应用 UML 进行面向对象分析和设计为主题，帮助软件工程师在排除关键障碍的基础上，通过推敲实例，有步骤地掌握一套切实可行的方法和流程。但愿本书能成为你前行道路上的几块垫路石或一座里程碑。

背　景

20 世纪 90 年代末期，UML 成为国际对象管理组织（Object Management Group，OMG）认可的标准建模语言。UML 的几位创始人是面向对象方法学的世界级大师，UML 的核心价值和优势深刻地体现在面向对象的方法和流程之中。采用 UML 规范地表述面向对象分析和设计过程，能够显著提升开发效率并保障可持续发展，是公认的发展趋势。掌握 UML 已成为软件工程师极具潜在价值的技能。

在国内，相关的教学研究和工程实践越来越受重视，对面向对象分析和设计方面图书的要求越来越迫切。基于教学和咨询的实践，作者了解国内软件开发人员在接受、掌握和应用 UML 与面向对象方法过程中存在的主要困惑和实际困难。例如基础知识缺陷对理解概念的影响，东西方思维模式差异对领悟过程的影响，以及传统思维对接受面向对象方法的影响…，这些壁垒导致广大的实践者对这项先进技术的理解表面化、片面化甚至神秘化，失去大量实践与提高的良机。

内　容

基于对民族软件工业的责任感，立足于实践者的视角，作者力求在书中突出

方法和流程的可用性与针对性，不追求博大精深和面面俱到，但强调目标明确和重点突出。全书分为以下三个部分。

第一部分，基本理念和准备知识，是本书的铺垫。首先解释分析和设计模型的含义和价值，然后概述面向对象技术的内涵、优势和原则，最后介绍模型内容的组织和相关的 UML 表述。

第二部分，UML 应用建模实践过程，是本书的核心。详细展现分析和设计过程中的五项任务，分别为全局分析、局部分析、全局设计、局部设计和细节设计。其中包括 14 项基础活动、39 个核心概念、30 个关键步骤、52 条实践技巧以及贯穿全程的示例。本书描述的实践过程充分借鉴了 Rational 统一过程（RUP）的核心思想和基本原则。RUP 汇集了众多的成功经验并逐步成为该领域的事实标准，它倡导以 Use Case 驱动的、体系构架为核心的迭代化面向对象分析和设计过程。通俗地讲，就是软件需求牵引的、既往经验支撑的和风险前驱的开发过程。

第三部分，设计模型的沿用，是本书内容的延伸。概要地介绍与设计模型直接相关的活动和内容，包括设计模型向实施模型的过渡、设计模型和数据模型的关联以及如何整理主要的设计文档。

读　者

如果你是面向对象的程序员，你会发觉，缺乏面向对象分析和设计指引的面向对象编程只能得到似是而非的面向对象软件。本书帮助你将以往随机的分析和设计活动融入系统化的过程。在介绍 UML 重要语义的过程中，引用程序代码片段作为辅助诠释，帮助读者利用已有知识，快速建立更有价值的新概念。

如果你是应用面向对象技术的系统构架师或设计师，本书的核心内容正是你的主要职责。实践过程中详述了分析和设计任务从全局到局部再到细节的渐变和迭代，强调了系统构架师和设计师的不同侧重与配合，突出了软件需求在分析和设计活动中的牵引作用，说明了成熟设计经验用于简化和支撑分析和设计活动的价值。

对于项目经理和研发主管而言，UML 与面向对象技术提供了打破时间与质量之间僵局的机遇。本书的实践过程中，具体地说明如何在分析和设计活动中贯彻风险前驱的迭代化开发策略，明确地指出不同角色人员的基本素质要求，详细地介绍可操作的团队并行协作方式。

如果你读过《统一软件开发流程》、《设计模式》和《UML 用户指南》[⊖]，相信你学到的战略思想、战术策略和符号体系将给你带来更大的收获。当然，这不

[⊖] 三本书的原文名称分别为《The Unified Software Development Process》、《Design Patterns: Elements of Reusable Object-Oriented Software》和《The Unified Modeling Language User Guide》，Addison-Wesley 出版。

能脱离你严谨的态度、批判的眼光和不辍的实践。

致　谢

感谢 Ivar Jacobson 博士（UML 创始人之一）在建模问题上给作者高屋建瓴地点拨。和 Terry Quatrani 女士（UML 制定者之一）关于统一建模问题的讨论使作者获益非浅，在此表示深切感谢。

感谢参与审阅本书的朋友们。感谢来自著名企业的软件专家，他们是 IBM 的张晨曦、陈曦、梁朝东，Microsoft 的李峻，Rational Software 的吴穹，Motorola 的李玉山，Nortel 的白静原，用友软件的游志强，中国金融电子化公司的周夕崇，亚信公司的刘炜，中兴通讯的张雪敏。感谢 UMLCHINA 和《XProgrammer》的创始人潘家宇。感谢来自著名高校的老师，他们是北京航空航天大学软件学院的姚淑珍、张莉、林广艳，清化大学软件学院的刘强，王少峰，复旦大学软件学院的薛云蛟、吕钊、上海交通大学软件学院的步丰林、黄小平，西安交通大学软件学院的何亮、杨新宇，华中科技大学软件学院的胡迎松、陈中新，武汉大学软件学院的薛超英、胡启平。感谢所有为本书提出建议的朋友。

尤克滨

2003 年元旦 北京

目　录

第一部分　基本理念和准备知识

一　脚踏实地

第 1 章　分析和设计的逻辑模型

1.1　模型在认知和求解中的价值

模型（Model）可以帮助我们在简约繁复的基础上，捕捉现实问题（Problem）的本质，勾勒软件方案（Solution）的雏形。模型有助于在问题到方案的过渡过程中更好地认知、理解和沟通。

模型是什么

简单讲，模型是现实的简化。准确讲，模型是能动的抽象认知结果，它对应一个认知活动的主体和认知活动的原则。认知活动的主体决定了特定的视角，可以理解为简化的动机；认知活动的原则决定了特定的抽象层次，可以理解为简化的水平。

对于一个系统，基于不同的简化动机和简化水平，可以得到多个模型，有助于更深刻和更准确地把握系统的本质。模型可以描述系统的静态结构，也可以描述系统的动态行为；模型可以描述系统的宏观面貌，也可以描述系统的微观情境。

模型是现实的简化，并不隐含时间顺序。也许是巧合，如果将“现实的简化”的语序反过来，则变成“化简的实现”，也同样成立。面向方案的设计模型会先于方案而存在，模型提供了营造方案的蓝图，也可以包括详尽的规划。

利用价值高的模型就是好模型，它们通常会忽略那些与特定抽象层次无关的次要因素，强调那些具有广泛影响力的主要因素。换言之，内容多的模型未必是好模型，因为价值高的内容有可能被价值不高的内容淹没。

模型是一组具有完整语义的信息，包括两个方面的内容：一方面，对现实的简化或者对实现的化简；另一方面，认知主体的视角和抽象层次。前者是被认知的客体，表现为各种类型的图（Diagram）及其包含的元素和关联；后者反映认知的主体，表现为不同类型的视图（View）。两者都是模型不可或缺的要素。

尽管强调模型的简化和化简价值，这并不意味着可以片面地夸大图示信息的能量，好的模型应该图文并茂，关键是可用和易用。

建模的价值

对于问题，模型是现实的简化，对于方案，模型是化简的实现。建模

（Modeling）是捕捉系统本质[⊖]的过程。

为了降低风险和获得高回报，建模活动普遍应用于各种行业，软件开发更不例外。为说明建模的价值，Grady Booch 曾经给出过一个经典的类比：盖一个宠物窝棚，修一个乡间别墅和建一座摩天大楼，三种工作对建筑规划图纸的依赖程度有质的差异。建立一个简单的系统，模型可有可无；建立一个比较复杂的系统，模型的必要性增大；建立一个高度复杂的系统，模型则不可缺少。应用处理简单系统的方法对待复杂系统通常行不通，这好比用搭建一个宠物窝棚的方法来营造一座摩天大厦。

建模的意义随着系统复杂程度的增加而越发显著，从起初借助模型更好地理解系统，到后来不得不借助模型来理解系统，充分说明了这一切。人脑对复杂问题的理解能力是有限的，与模型相应的特定视角和抽象层次是简化复杂问题的有效出发点。

当今，我们开发的软件，特别是商业软件，通常一开始就很不简单，并且，复杂性随着时间的演进和技术的发展持续上升。一个复杂软件系统的开发必须面对多种未知因素，多个开发人员，复杂的开发工具和永远不够用的时间。开发人员没有可能，更没有必要去了解从问题到方案的所有细节。他们需要那些基于特定视角的、有助于解决问题的、并且是完整的某一部分信息，即所谓的模型。总之，建模对于复杂软件系统的开发是必要的。

广义讲，无论出于何种动机，只要在问题到方案之间作出一些过渡性的努力，哪怕只是在草稿纸或是白板上画了几笔，实际上就是在建模了。不过，有意识和无意识的建模活动对模型质量或价值的影响很大。有意识的建模活动通常是有计划的、有准备的和早动手的，得到的模型通常是完整的、一致的和可复用的；无意识的建模活动通常是随机的、无准备的和补救性的，得到的模型往往是零散的、混乱的和一次性的。

准确地讲，建模活动直观地记录下认知和求解的过程，支持团队成员之间的有效沟通，为重复利用各个阶段积累的智力成果创造有利的条件。

概括地讲，建模简化了认知过程，化简了求解过程。如果读者希望进一步拓展对模型和建模的理解，可以参考《UML 用户指南》。

统一建模语言 UML

统一建模语言 UML，全称 Unified Modeling Language。

首先，简要地回顾一下 UML 的由来。20 世纪 90 年代初，很多面向对象的方法已经拥有自己的符号体系，其中有三种比较突出：Jim Rumbaugh 的 OMT 方法，Grady Booch 的 Booch 方法以及 Ivar Jacobson 的 OOSE 方法。不同的方法和符号体系各有所长：OMT 擅长分析，Booch 擅长设计，OOSE 则擅长业务建模。那个时期的面向对象技术人员远没有我们这么幸运，为了建立比较丰满的模型并

[⊖] Modeling captures essential parts of the system. -Dr. James Rumbaugh

进行有效的沟通，他们需要掌握不同的符号体系，并且花费一些精力去翻译和转述用不同符号体系记录的模型。在后来的几年中，上述三位大师在各自的著作中自然而然地融入了其他两种方法的技术内容。Jim Rumbaugh 于 1994 年离开 GE 加入 Grady Booch 所在的 Rational 公司，开始和 Grady Booch 协同研究一种统一的方法。一年后，Unified Method 0.8 诞生了。同年，Rational 收购了 Ivar Jacobson 所在的 Objectory 公司，Ivar Jacobson 从此也成为 Rational 的一员。Unified Method 不久更名为 UML，仰仗三位面向对象方法学大师的威望，基于数十位行业内重量级人物历时两年的通力合作，并充分考虑到多方合作伙伴的反馈意见，UML 一步步趋向成熟。1997 年 9 月，UML 1.1 被提交到国际对象管理组织，同年 11 月被该组织认定为标准的建模语言。

统一建模语言，顾名思义，有三个要点：统一（Unified）、建模（Modeling）和语言（Language）。

很难想像如果没有五线谱，指挥家和乐手如何奏出交响乐。软件开发活动对统一表述符号的要求是类似的，并且，软件开发活动的规模和复杂程度通常不亚于大型交响乐团的排练和演出。

“统一”是 UML 的核心，它提升了软件开发团队的沟通效率，节约了以往用于翻译和转述的开销，屏蔽了藏匿于含糊语义中的风险。

UML 表述的内容能被各类人员所理解：包括客户、领域专家、分析师、设计师、程序员、测试工程师以及培训人员等。他们可以通过 UML 充分地理解和表达自己所关注的那部分内容。在传统概念中，他们各自拥有专用的符号系统，这也是长期以来潜在的沟通壁垒。

除了在多工种人员之间形成统一，在工具辅助下，UML 还可以通过双向工程（Round-trip Engineering）实现开发人员和开发环境的统一。

“建模”体现了 UML 的使用价值。UML 在制定过程中汲取了多种建模方法的精华，包括业务建模和数据建模等。UML 的使用价值不可能脱离特定类型的建模活动。对于学习者而言，如果以掌握 UML 的符号和规则为最终目的，你将所获甚微。尽管 UML 所表述的内容可以贯穿软件开发的生命周期，但 UML 不同于普通的程序设计语言，仅仅掌握 UML，并不能得到实际的解决方案。

“语言”是 UML 普遍价值的表现。语言的一层基本含义是一套按照特定规则和模式组成的符号系统，被拥有相同传统和习惯的人群所使用。我们在日常生活中将“拥有共同语言”视作有效沟通的必要条件。近年来，软件开发所涉及的技术飞速发展，不同技术门类所使用的建模语言自成体系，同时也具有很大局限性，表现形式的差异往往掩盖了本质内容的相通。幸好，与人类的自然语言不同，在软件开发过程中使用的建模语言不涉及宗教和文化等诸多历史或地域障碍。在博采众长的基础上，UML 作为一种共通的和可扩展的语言，其描述能力适用于软件开发中各种技术门类的建模活动。自然语言是人类对客观世界建模最直接有效的表述形式；类似地，UML 是迄今软件开发人员进行统一建模活动最直接有效的表述形式。不仅如此，UML 还是能被软件开发环境所理解的语言。

1.2 分析和设计的对立、关联和统一

客户根据业务需要（Needs）提出软件需求（Software Requirements），开发人员根据技术环境实施程序代码（Code），分析和设计是需求到代码之间平滑过渡的桥梁。概括地讲，分析是一个翻译软件需求和深入理解问题的过程，设计是一个逐步精化方案和适应实施环境的过程。尽管二者的基本目标对立，但两种思辨过程却紧密关联，为了在对立的目标和关联的思辨过程之间建立明确的映射和平滑的过渡，我们需要统一的思想原则、统一的表述形式以及融合思想原则和表述形式的统一实践过程。

目标的对立

分析的目标是理解问题并开发一个简要描述方案的可视化模型，不依赖于具体的实施技术环境。分析活动回答“要做什么”的问题，工作的重点是将功能性需求翻译成软件的概念，或者说用软件的概念来诠释问题所要求的功能。分析的核心是捕获问题的行为，在屏蔽实施细节的基础上得到构成方案的粗略对象模型。

设计的目标是精化方案并开发一个明确描述方案的可视化模型，保障设计模型最终能平滑地过渡到程序代码。设计活动回答“该怎么做”的问题，工作的重点是适应特定的实施环境和部属环境。设计的核心是规划方案的构造，在揭示实施细节的基础上得到方案的详细对象模型。实施环境指构建软件的环境，部属环境指运行软件的环境。

分析面向问题，是明确动力的过程，重在理解和翻译，灵活性高；设计面向方案，是排除阻力的过程，重在精化和适应，受约束大。两种活动的对立是客观的。因而，没有不存在的对立，只有未被意识到的对立。从整体上看，分析和设计的对立是保障问题和方案趋于一致的基本动力。开发一个好软件的基本任务之一就是让这种对立更明确地表现出来，更好地被认识和利用。

思辨过程的关联

人们从问题到方案的认知和求解思辨过程非常复杂，分析和设计的交错演进客观存在，试图将两种活动截然分开通常会带来不利影响。

有时，提出阶段性方案的目的就是为了更好地理解问题。在实际的建模过程中，一开始从特定的要点入手，创建简单的对象模型去模拟现实问题，即所谓的分析；然后，从分析的结果出发，以一些通用的元素为基础，创建较为复杂的对象模型来规划解决方案，即所谓的设计。设计的价值不仅在于它给出了一个方案空间对问题空间的映射，同时，设计往往展现出当前对问题理解的深度和精度的不足，这些具有启发价值的信息有助于在更明确的范围内展开进一步的分析活动。

当掌握了一些设计的经验和技巧之后，在分析活动中能够有意识地作出合理的假定，屏蔽和推延一些设计细节的出现，更好地保障分析和问题之间的简明对