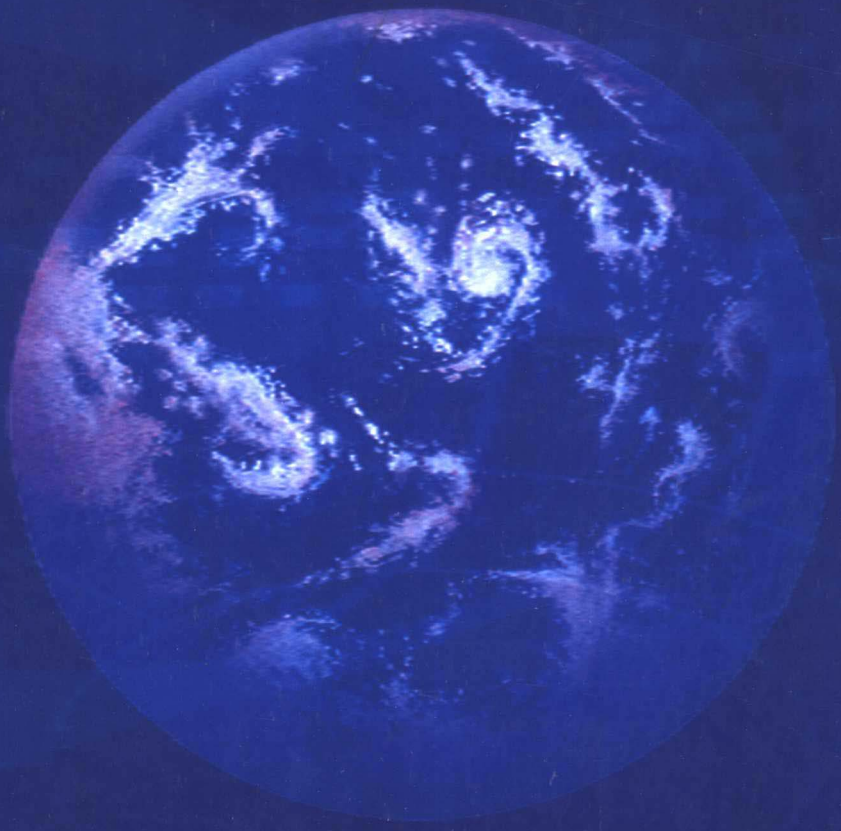
UML面向对象技术与实践

宋 波
刘 杰 编著
杜庆东



科学出版社
www.sciencep.com

UML 面向对象技术与实践

宋 波 刘 杰 杜庆东 编著

科学出版社
北京

内 容 简 介

本书是作者们多年来研究 UML 基础知识问题的概括和总结。主要内容包括：UML 的基础知识和在 Rational Rose 建模环境下创建九种 UML 图的方法；并以软件工程的开发为主线，把面向对系统——“图书管理系统”的需求分析、系统分析与设计、以及实现的过程进行了详细的描述，并给出了系统实现的全部源代码。

本书可供高等院校计算机专业的教学科研人员、研究生、本科生参考和使用。

图书在版编目(CIP)数据

UML 面向对象技术与实践 / 宋波，刘杰，杜庆东编著. 北京：科学出版社，2005
ISBN 7-03-016642-6

Ⅰ.U… Ⅱ.①宋… ②刘… ③杜… Ⅲ.①面向对象语言,UML一程序设计②JAVA 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2005)第 151511 号

责任编辑：范 欣　　责任校对：侯沈生
责任印制：任继革　　封面设计：张祥伟

科 学 出 版 社 出版
北京东黄城根北街 16 号
邮政编码：100717
http://www.sciencep.com
沈阳新华印刷厂印刷
科学出版社发行 各地新华书店经销
*
2006 年 1 月第 一 版　开本：787×1092 1/16
2006 年 1 月第一次印刷　印张：17 1/4
印数：1—2000　字数：410 千字
定价：32.00 元
(如有印装质量问题,我社负责调换)

前　言

统一建模语言 UML 是面向对象技术领域内占主导地位的标准建模语言，已被 OMG（Object Management Group，OMG）采纳为标准。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过程的理解和运用。

本书主要介绍 UML 知识及应用，目的是使读者掌握面向对象分析和设计的方法。书中通过一个典型的案例分析，比较全面、系统地描述了在 Rational Rose 建模环境下用 UML 和 Java 语言进行系统的规划、分析、设计、实现和部署的全过程。

本书的知识体系可以划分为两个部分：第一部分包括第一章至第九章，介绍了 UML 的基础知识和在 Rational Rose 建模环境下创建九种 UML 图的方法；第二部分包括第十章至第十四章，以软件工程的开发为主线，把面向对象软件开发方法与 UML 建模技术有机地结合起来，对一个实际的 Java 应用系统——“图书管理系统”的需求分析、系统分析与设计、以及实现的过程进行了详细的描述，并给出了系统实现的全部源代码。

第 1 章　UML 概述。本章介绍 UML 的历史、UML 建模要素，解释 UML 标准视图的概念，阐述 UML 图与 OOP 之间的关系，简要介绍 UML 的应用领域，以及 UML 与软件开发的各个阶段之间的对应关系。

第 2 章　Rational Rose 概述。本章简要介绍 Rational Rose 2003 建模工具的使用。

第 3 章　UML 用例图。本章介绍用例图中的参与者、用例、脚本、关系等概念和术语，以学生成绩系统为例，说明在 Rose 中创建用例图的方法和步骤。

第 4 章　UML 顺序图和协作图。本章介绍顺序图和协作图中的概念和术语，以学生成绩系统为例，说明在 Rose 中创建顺序图和协作图的方法和步骤。

第 5 章　UML 类图。本章介绍类图的基本概念，阐述类之间的关系，解释 UML 中边界类、控制类和实体类的概念，并以学生成绩系统为例，说明在 Rose 中创建 UML 类图的方法和步骤。

第 6 章　UML 数据建模。用 E－R 图设计数据库存在的主要问题是只能对数据建模，不能对行为建模。UML 类图的描述软件系统的能力更强，可以对 E－R 图的功能进行扩充。对于关系模型来说，可以用类图描述数据库模

式，用类描述数据库表。本章以实例说明如何将 UML 类之间的关系转换为关系模式。

第 7 章　UML 状态图和活动图。本章介绍 UML 状态图和活动图中的概念和术语，以学生成绩系统为例，说明在 Rose 中创建 UML 状态图和活动图的方法和步骤。

第 8 章　UML 组件图和部署图。本章介绍 UML 组件图和部署图中的概念和术语，以实例说明在 Rose 中创建 UML 组件图和部署图的方法和步骤。

第 9 章　RUP 软件开发过程。本章介绍 RUP 的简要发展史、RUP 的概念、RUP 与最佳实践、RUP 中的核心术语、RUP 软件开发生命周期以及 RUP 的特点。

第 10 章　Rose 业务视图。软件系统的业务视图能够帮助业务人员确定系统的需求分析是否准确地反映了用户需求，以及从用例分析中得到的用例是否完整地满足了用户需求。本章介绍 Rose 业务视图方面的知识。

第 11 章　Rose 用例视图——需求分析。从本章开始，将以一个实际的 Java 应用系统——“图书管理系统”作为案例，阐述和分析如何用 UML 与 Rose 建模工具为一个面向对象的应用系统建模的基本原理和方法。本案例实现的图书管理系统能够完成读者信息、书籍信息以及两者相互作用产生的借书信息、还书信息等计算机化管理，适用于图书馆以及企业、事业单位的图书资料室使用。本章介绍在需求分析阶段如何创建“图书管理系统”的 Rose 用例视图。

第 12 章　Rose 逻辑视图——分析模型。本章介绍在系统分析阶段如何创建“图书管理系统”的 Rose 逻辑视图。

第 13 章　Rose 逻辑视图——设计模型。本章介绍在系统设计阶段如何创建“图书管理系统”的 Rose 逻辑视图。

第 14 章　Rose 组件、部署视图——实现模型。本章介绍在系统实现阶段如何将创建的“图书管理系统”的 Rose 模型生成 Java 源代码。

本书中的第一章至第五章、第十三章至第十四章由宋波编写，第六章至第九章由刘杰编写，第十至十二章由杜庆东编写。本书从选题到立意、从酝酿到完稿，自始至终得到了沈阳师范大学科研处、教务处的领导和老师的关心与指导。本书也吸纳和借鉴了中外参考文献中的原理知识和资料，在此一并致以谢忱。

宋波

2005 年 10 月

目　录

第 1 章　UML 概述

UML 是一种可以用于任何软件开发过程的标记法和语义语言。UML 用一组图形符号描述软件模型，这些符号具有简单、直观、规范的特点。UML 可以从不同视角和方面描述人类所观察到的软件视图，描述在不同开发阶段中的软件形态。可以使用 UML 建立面向对象系统的需求分析、系统分析与设计、以及系统实现的软件模型。

本章首先介绍 UML 的历史，然后说明 UML 的建模要素，并解释 UML 标准视图的概念。在解释面向对象领域中的基本概念之后，阐述 UML 图与 OOP 之间的关系。最后介绍 UML 的应用领域，以及 UML 与软件开发的各个阶段之间的对应关系。

1.1　UML 的历史

如图 1.1 所示，是反映 UML（Unified Modeling Language，统一建模语言）发展历史的简图。

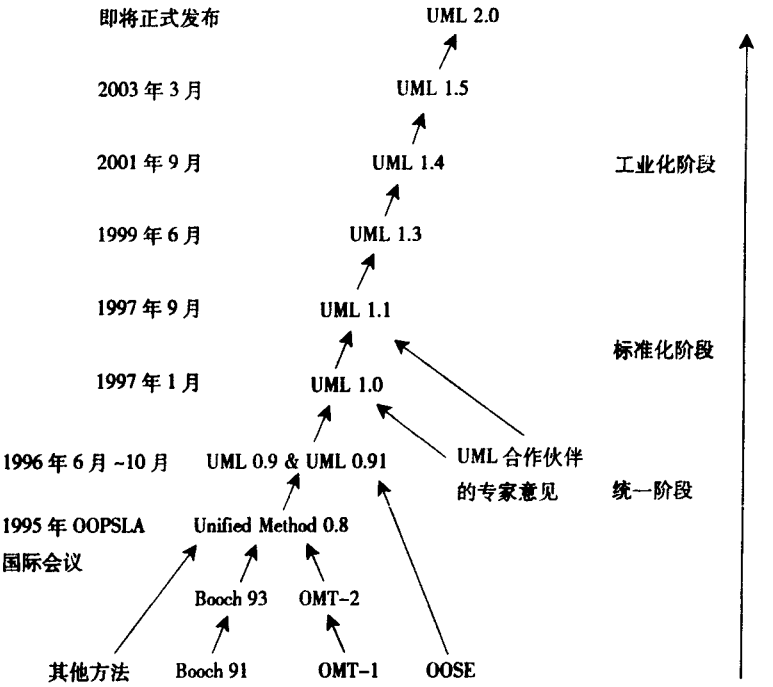


图 1.1　UML 发展历史

UML 是由世界著名的面向对象技术专家 Booch、Rumbaugh 和 Jacobson 发起，在

Booch 方法、OMT（Object Modelling Technique）方法和 OOSE（Object - Oriented Software Engineering）方法的基础上，汲取其他面向对象方法的优点，广泛征求意见，几经修改而完成的。目前 UML 得到了 IBM、HP、Oracle、Microsoft 等诸多大公司的支持，已成为面向对象技术领域内占主导地位的标准建模语言。

目前最新的 UML 规范说明是 2003 年 3 月发布的 UML 1.5 版本。OMG（Object Management Group）同时进行两个 UML 版本的改进工作。一个是对 UML 1.x 版本进行的改进工作，一个是对 UML 2.0 版本（2001 年开始）的改进工作。由于 UML 2.0 版本是一个比较大的升级工作，其发布时间也一再推迟。2003 年 8 月，OMG 发布了 UML 2.0 最后的征求意见版本，正式的版本将很快发布。

1.2 UML 建模要素

UML 作为一种对软件系统的建模语言，提供了描述事物实体、性质、结构、功能、行为、状态和关系的建模元素，并通过一组图描述由建模元素所构成的多种模型。UML 建模要素包括基本建模元素、关系元素和图三大类，如图 1.2 所示。

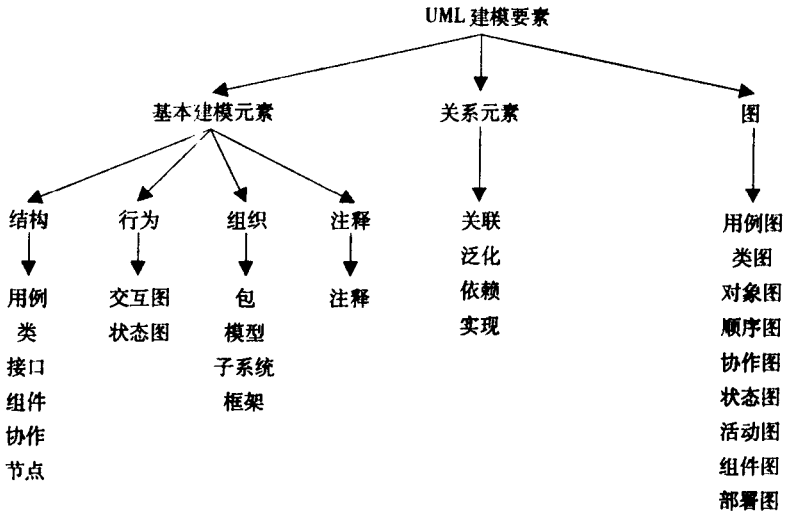


图 1.2 UML 建模要素

1. 基本建模元素

基本建模元素可以分为结构、行为、组织和注释 4 类。

- 结构建模元素——反映事物和描述性实体，包括用例、类、接口、构件、协作和节点等元素。

- 行为建模元素——反映事物之间的交互过程和状态变化，这类建模元素有交互图和状态图。
- 组织建模元素——描述通过一组模型元素所反映的模型、子系统、框架等的组织，包括包、模型、子系统、框架等元素。
- 注释建模元素——用来在建模过程中对模型进行注释和描述性说明。

2. 关系元素

关系元素反映了模型元素之间的关系，包括关联、泛化、依赖和实现关系。

3. 图

通过基本建模元素构成的图用来表示软件模型。UML中的图包括：用例图（Use Case Diagram）、类图（Class Diagram）、对象图（Object Diagram）、顺序图（Sequence Diagram）、协作图（Collaboration Diagram）、状态图（Status Diagram）、活动图（Activating Diagram）、组件图（Component Diagram）以及部署图（Deployment Diagram）共9种图。

4. 图形表示

UML定义了以下两类8种图形表示各种软件模型。

- 静态结构图——包括类图、对象图、组件图、部署图。
- 动态行为图——包括用例图、交互图（顺序图和协作图）、状态图、活动图。

1.3 UML标准视图

UML是用来描述模型的，而模型用来描述系统的结构或静态特征，以及行为或动态特征。UML定义了9种图，这9种图就是UML中的内容。另外，UML还提供了视图（View）的概念。

视图并不是一种图（Graph），它是由若干幅图（Diagram）组成的一种抽象。也就是说，每种视图可用若干个图描述。一幅图包含了系统某一特殊方面的信息，阐明了系统的一个特定部分或方面的内容。由于在不同视图之间存在一些交叉，所以一幅图可以作为多个视图的一部分。一幅图由若干个模型元素组成，模型元素表示图中的概念。UML中的诸如类、对象、用例、节点、接口、包、注解、组件等都是模型元素。用于表示模型元素之间相互连接的关系也是模型元素，UML中的诸如关联、泛化、依赖、聚集等也都是模型元素。

与图相比视图仅仅是一个观察视点而已，图才真正描述了系统。一般软件系统的分析是以图为主的，但是对图进行管理的时候，可以将它们组织为视图。视图的划分并无固定的方式，可以根据实际情况进行划分。

常用的UML工具软件，也不一定非要默认支持全部的视图。可以这样说，图是实

际软件系统模型的描述，视图是图的组织。UML 中有 5 种标准视图：用例视图（Use Case View）、逻辑视图（Logic View）、进程视图（Process View）、组件视图（Component View）以及部署视图（Deployment View）。

1. 用例视图

用例视图强调从用户的角度所看到的系统功能，被称为参与者的外部用户所能观察得到的系统功能的模型图。用例视图实际上并没有描述软件系统的组织，而是描述了形成系统体系结构的构成。

在 UML 中，用例视图的静态方面用用例图表现；动态方面用协作图、状态图和活动图表现。用例视图是中心，因为它的内容决定了其他视图的设计。用例视图还可用于确认和最终验证系统，用户根据用例视图来确认所构造的系统是否是所需要的，用例视图可以测试系统是否完成了指定的功能。用例视图只考虑系统应提供什么样的功能，对这些功能的内部运作情况一般不予考虑。

2. 逻辑视图

逻辑视图描述了系统内部的设计和协作状况，显示了系统内部的功能是怎样设计的。逻辑视图利用系统的静态结构和动态行为刻画系统的功能。其中，静态结构描述了类、对象以及两者之间的关系。逻辑视图针对分析者、设计者和编程者。静态结构在类图和对象图中描述，动态建模用状态图、顺序图、协作图和活动图描述。

3. 进程视图

进程视图体现了系统的动态行为特征。进程视图除了将系统分割成并发执行的控制进程外，还必须处理这些进程通信的同步。进程视图针对开发者和系统集成者，用于描述性能、可伸缩性、以及系统的吞吐量。在 UML 中，进程视图的动态和静态描述与逻辑视图相同。

4. 组件视图

组件视图显示了代码组件的组织方式，实现了模块之间的依赖关系。组件视图包含了用于装配和发布系统的组件和文件。组件视图主要针对发布的配置管理，其中的组件和文件可以用各种形式装配并产生运行时系统，描述这种装配方式的就是组件视图。组件视图的静态方面由组件图描述，动态方面由协作图、状态图、活动图描述。

5. 部署视图

部署视图显示了系统的物理框架，即系统如何在物理设备上进行部署。例如，计算机和其他设备以及它们之间的连接方式。其中，计算机和设备被称为结点。部署视图主要描述对组成物理系统的部件的分布、特性、以及交付情况。部署视图的静态内容由部署图描述，动态方面由协作图、状态图、活动图描述。

1.4 面向对象领域中的基本概念

本节将对面向对象领域中常见的几个概念和术语做简单的解释，这些概念和术语包括：对象、实例、类、属性、方法、封装、继承、多态、消息等。

1. 对象和实例

对象（Object）是系统中用来描述客观事物的一个实体，它是构成系统的一个基本单位。一个对象由一组属性和对这组属性进行操作的一组方法组成。从更抽象的角度来观察，对象是问题域或实现域中某些事物的一个抽象，它反映事物在系统中需要保存的信息和具有的功能，是一组属性和有权对这些属性进行操作的一组方法的封装体。客观世界就是由对象和对象之间的联系组成的。

消息（Message）是向对象发出的服务请求，它包含了提供服务的对象标识、服务（方法）标识、输入信息和应答信息。对象之间通过消息通信，一个对象通过向另一个对象发送消息激活某一个功能。

一个对象是类的一个实例（Instance）。实例这个概念不仅仅是针对类而言，UML建模元素也有实例。例如，在UML中，关联的实例就是链接。

2. 类

类（Class）是具有相同属性和方法的一组对象的集合，它为属于该类的所有对象提供了统一的抽象描述。类是一个独立的程序单位，它有一个类名并包括属性说明和方法说明。类的实例化结果就是对象，而对一类对象的抽象就是类。同类对象具有相同的属性和方法；是指它们的定义形式相同，而不是说每个对象的属性值都相同。

类是静态的，类的语义和类之间的关系在程序执行之前就已经定义好了，而对象是动态的，对象是在程序执行时被创建和删除的。

3. 封装

封装（Encapsulation）是把对象的属性和方法结合成一个独立的单位，并尽可能隐蔽对象的内部细节。封装包含以下两个基本涵义：

- 把对象的全部属性数据和对数据的全部操作（方法）结合在一起，形成一个不可分割的独立单位，也就是对象。
- 信息隐蔽，即尽可能地隐藏对象的内部细节，只保留有限的对外接口，对数据的操作都通过这些接口实现。

封装的原则在软件上的反映是：要求使对象以外的部分不能随意存取对象的内部数据（属性），从而有效的避免了外部错误对它的“交叉感染”，这样就可以使软件错误局部化，大大减少查错和排错的难度。

在面向对象程序设计中，一个类通过信息隐藏与封装形成了一个完整的、相对独立