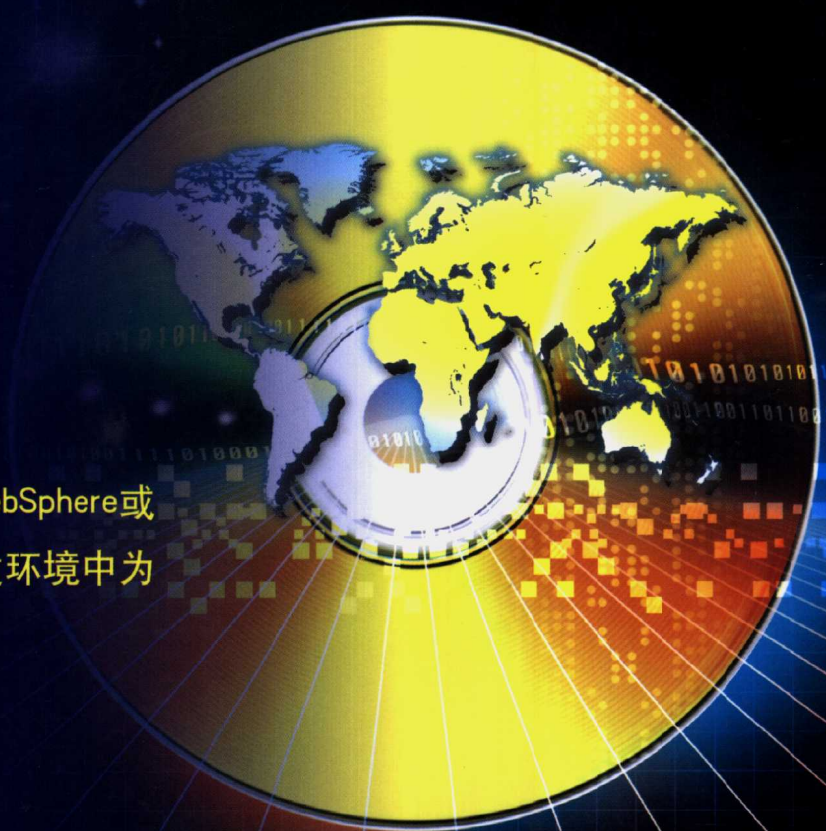
Mastering Rational XDE

Rational XDE

从入门到精通

[美] Wendy Boggs
Michael Boggs 著

邱仲潘 等译



掌握如何在熟悉的WebSphere或Visual Studio.NET开发环境中为自己的应用程序建模

SYBEX

PHEI

電子工業出版社
Publishing House of Electronics Industry
http://www.phei.com.cn

Mastering Rational XDE

Rational XDE
从入门到精通

〔美〕 Wendy Boggs
Michael Boggs 著

邱仲潘 等译

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 提 要

XDE是Rational公司为高效与有效开发软件产品而贡献的新工具。本书将概述XDE在软件开发周期中的角色与作用；介绍如何创建分析模型、设计模型与数据模型，如何生成代码并保持代码与模型同步，如何使用XDE模型中的模式和创建自己的模式，如何发表模型和生成报告材料，以及如何创建和使用可复用资源规范。

本书适合Rational XDE的初级与中级用户，也适合对软件开发感兴趣的读者阅读。

版权贸易合同登记号：01-2003-3204

图书在版编目（CIP）数据

Rational XDE从入门到精通/（美）博格斯（Boggs, W.）等著；邱仲潘等译.—北京：电子工业出版社，2003.11

书名原文：Mastering Rational XDE

ISBN 7-5053-9216-6

Ⅰ. R… Ⅱ. ①博… ②邱… Ⅲ. 软件开发 Ⅳ. TP311.52

中国版本图书馆CIP数据核字（2003）第089228号

责任编辑：李莹

印　　刷：北京天竺颖华印刷厂

出版发行：电子工业出版社　http://www.phei.com.cn

北京市海淀区万寿路173信箱　邮编：100036

北京市海淀区翠微东里甲2号　邮编：100036

经　　销：各地新华书店

开　　本：787×1092 1/16　印张：29.5　字数：750千字

版　　次：2003年11月第1版　2003年11月第1次印刷

定　　价：49.00元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换，若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

译者的话

本书翻译过程中得到了周阳生、刘文红、邹能东、彭振庆、黄志坚、李耀平、郭王旋等同志的大力帮助，刘文琼、邱冬金、邱燕明、黄素平等同志完成了本书的录入工作，刘云昌、刘联昌兄弟帮助进行了书稿与打印稿的校对，在此深表感谢。

简　介

啊，告别石器时代真好！在穴居人的时代，人们既要找吃的，又怕被恐龙吃掉。很快，他们就建立了一种相互交流的语言。后来，穴居人学会了制造石凿之类的简单工具。再后来，人类发生了更大的进化；虽然人们还要找吃的，但街角路边到处都是麦当劳和肯德基，要找饭吃也就不难了。恐龙呢？恐龙已经灭绝了，但我们没有灭绝。

在不断改变的软件世界中，我们也来自石器时代。在早期软件开发中，我们有开发语言，但建模与设计则要每个人自寻出路。和穴居人一样，我们开发了一些简单工具，以帮助自己度日。这些最初的开发工具随着时间的推移和计算机功能的日益强大而变得越来越捉襟见肘。后来出现了面向对象的概念。与面向对象的概念同时出现的是一些建模工具，包括统一建模语言（UML，Unified Modeling Language）。随着面向对象概念的不断进化，人们开发了更多的工具，以帮助进行建模与设计工作。许多年来，Rational Rose一直是主要的建模工具之一。

随着Rational Rose的引入，建模者可以按照面向对象的原则建立系统模型，甚至从模型产生框架代码。利用Rose还可以对现有代码进行逆向设计，亦即产生代码的模型。Rose使设计员、开发员和结构设计师有了高效与有效工作所需的工具，使软件开发世界焕然一新。从此，我们不再处于石器时代，而是进入了摩登时代。

在这个摩登时代，设计员和开发员有了漂亮的工具，但这些工具的集成仍然是个难题。设计员用自己的模型生成代码，而开发人员要改变代码，然后将其放回模型中。

有一天，设计员手持开发工具，走在街上；开发员也手持开发工具，走在街上。他们看着自己的工具非常得意，差点忘了自己姓什么。一不小心，他们撞上了，摔了一跤，手中的工具也全乱套了。他们爬起来，拍干净身上的灰尘，突然发现双方的工具已经混在一起。“你的开发环境在我的建模器中！”设计员说；“你的建模器在我的开发环境中！”开发员叫道。他们发现，新工具非常好，也非常集成。他们将其命名为XDE，表示不小心碰到一起的极端专心的工程师（Xtremely Determined Engineers）。

随着Rational XDE的推出，这个工具发生了最新的进化。现要，开发环境和建模器已经融为一体，代码与模型同步的问题不再困扰我们了。

本书适用对象

本书是为Rational XDE的初级与中级用户和刚刚对其行业感兴趣的读者设计的。创作这本书时，我们从熟悉面向对象分析、设计与开发的读者角度进行考虑。读者应熟悉统一建模语言（UML）和面向对象开发，特别是使用Java或Microsoft.NET进行面向对象开发。如果还不熟悉统一建模语言（UML），请先阅读附录“UML入门”。

本书将介绍Rational XDE的基础知识：

- 如何在XDE中导航
- 如何生成用例并将其与要求存储库相联系

- 如何生成顺序框图
- 如何生成类框图
- 如何生成状态表框图
- 如何生成Enterprise JavaBeans（EJB）
- 如何将模型与Java或.NET代码同步
- 如何建模数据库
- 如何生成组件与部署框图
- 如何与其他小组成员共享模型
- 如何使用与生成模式
- 如何将模式存储成可复用资源规范（RAS）

读者不一定要按顺序阅读本书。每一章详细介绍Rational XDE的一个方面，而每章末尾提供有练习题，帮大家练习XDE的使用。但这些练习应在前几章练习的基础上去做。

尽管本书介绍的是XDE与建模要素的基础知识，但没有介绍XDE的每个特性。如果读者不熟悉XDE，则建议顺序阅读各章内容，完成所有练习。这些练习将用样本系统演示建模过程。如果还不熟悉统一建模语言（UML），请先阅读附录“UML入门”。

本书的组织形式

本书分成三大部分共12章，最后有一个附录，见下面几节介绍。

第一部分：用Rational XDE自由自在地开发

第一部分介绍Rational XDE及如何在软件开发寿命周期中使用XDE。第二部分和第三部分的练习要继续介绍这个过程。这些练习将建立第3章引入的时间记录系统。到第一部分结束时，读者应了解XDE的功能，其在软件开发寿命周期中的地位，并要熟悉第二部分和第三部分的练习所处的背景环境。

第1章：Rational XDE简介 介绍建模工具的演变和统一建模语言（UML），以及XDE中的各种UML框图。

第2章：XDE与软件开发寿命周期 介绍XDE在软件开发寿命周期中的地位，以及Rational统一过程（Rational Unified Process）与极限编程（eXtreme Programming）方法论中的活动。

第3章：练习简介：实际应用程序开发分析 本书其余部分的练习要学习Rational XDE的用法。故本章主要介绍这些练习所处的背景环境——一家假想的专门开办旅馆连锁店的StayHere公司。StayHere公司准备开发一个时间记录系统，并且要用XDE进行开发。所有练习都与StayHere公司要开发的时间记录系统有关。

第二部分：XDE与可视建模

第二部分详细介绍XDE的用法，介绍如何生成各种UML框图，如何建模Java或Microsoft元素。我们还要介绍如何建模数据库，如何与其他小组成员共享模型。阅读第二部分后，读者就可以生成用例、顺序、类、组件、部署和自由框图，还可以将实体类转换成数据库表和生成数据库。最后，可以向Web上报告和发表模型，从而与其他小组成员共享模型。

第4章：集成用例管理　XDE可以在RequisitePro中生成用例框图和管理用例。我们将介绍如何在XDE中生成用例与框图，还要介绍集成RequisitePro及其详细的使用过程。本章的练习涉及如何用Rational XDE来生成用例和管理用例规范。

第5章：在XDE中建模Java与J2EE元素　介绍Rational XDE在Java开发环境中的使用，如何将XDE用于Eclipse Java平台或IBM WebSphere，如何生成各种UML框图和Java元素，如何同步代码与模型。我们还要介绍如何使用Enterprise JavaBeans（EJB）技术，以及Rational XDE对EJB开发人员提供的强大支持。

第6章：在XDE中建模Visual Studio.NET元素　介绍.NET中的Rational XDE。这个环境向Microsoft.NET开发人员提供了聚敛的建模与开发工具。还要介绍XDE可以生成的UML框图与结构，以及代码与模型同步。本章的练习演示如何对要开发的StayHere时间记录系统建模.NET元素。

第7章：用XDE建模数据库　数据库是大多数软件项目的重要方面，使用XDE也可以建模数据库与对象。在此，我们要介绍如何将实体类变成数据库表，生成数据库，以及如何对数据库进行逆向设计。练习中要把时间记录项目中的一些实体对象变成数据库表。

第8章：模型发表与报表　在XDE中建模应用程序之后，要与小组其他成员共享这个信息。XDE提供了Web发表功能和一些标准模型报表。可以用这些特性与小组其他成员共享这个XDE模型中的信息，即使他们无法直接访问XDE。

第三部分：模式

XDE包括一组“四人帮”模式，我们将详细介绍这些模式及其用法。利用XDE还可以生成自己的模式并将其保存起来，让别人一同使用。学完第三部分后，读者就可以对“四人帮”模式及其用法有个一般了解。此外，还可以生成自己的模式并将其保存为可复用资源规范（RAS）。

第9章：使用模式　XDE带有一些标准“四人帮”模式，可以在项目中使用。本章介绍如何在模型中使用这些“四人帮”模式，并介绍如何将模式关联到模型中。做练习时，我们要在时间记录项目中使用一个“四人帮”模式。

第10章：“四人帮”模式　XDE带有许多“四人帮”模式，我们要详细介绍每个模式。而且在本章的练习中，实践这些“四人帮”模式在时间记录项目中的用法。

第11章：可复用资源规范（RAS）简介　第9章和第10章介绍如何使用模式，本章介绍如何包装模式，以便让别人使用。这个功能主要适用于生成的模式，因此第12章才会进行将模式保存为可复用资源规范（RAS）的练习。

第12章：生成模式　本章介绍如何生成可以在应用程序中使用的模式。练习中，我们要生成模式并将其保存为可复用资源规范（RAS）。

附录：UML入门

附录是个UML启蒙，适合不熟悉UML规范的读者，涉及不同类型UML框图的基础知识，以及每个框图中使用的符号。如果读者不熟悉UML规范，则最好先阅读这个附录，然后再学习书中的内容。

关于Web站点

书中讨论XDE特性时，会对示范性的时间记录项目建立几个模型。在与本书相关的Web站点中，大家可以看到每个练习完成后的XDE模型。我们还提供了Rational Web站点的链接，供查找Rational软件与产品、UML和对象建模的各种信息。我们也提供了Rational开发网（RDN）的链接。Rational开发网是为Rational产品许可证用户提供服务的，包括白皮书、模式、过程模块，等等。

在Sybex Web站点（www.sybex.com）中，用Catalog或Search工具可找到本书的Web页面，单击Downloads按钮即能进行下载。

与作者联系

如果有关于XDE的问题或需要更多帮助，可以与作者联系。Wendy的电子邮件地址为wboggs@boggsconsulting.com，Mike的电子邮件地址为mboggs@boggsconsulting.com。

目　录

译者的话 iv
简介 v

第一部分　用Rational XDE自由自在地开发 1

第1章　Rational XDE简介 1
开始 1
可视化建模简介 3
什么是Rational XDE 7
小结 17

第2章　XDE与软件开发寿命周期 18
XDE与Rational统一过程 18
XDE与极限编程 33
XDE与配置管理 41
参考读物 51
小结 52

第3章　练习简介：实际应用程序开发分析 53
项目背景 53
现有体系结构 56
业务分析 58
小结 61

第二部分　XDE与可视建模 63

第4章　集成用例管理 63
需求管理简介 63
在XDE中建立用例框图 64
包装用例与角色 76
集成用例管理 80
在XDE中建立活动框图 84
用例分析 92
下一步 107
练习：生成时间记录系统的用例与分析模型 107
小结 123

第5章　在XDE中建模Java与J2EE元素 124
生成Java应用程序项目 124

在UML中建模Java元素 129
在XDE中建模Java元素 134
建立设计模型 149
设计J2EE元素 172
处理Java代码 183
练习：J2EE应用程序从分析到设计 194
小结 199
第6章 在XDE中建模Visual Studio.NET元素 200
关于术语 200
对.NET应用程序生成项目 200
在UML中建模.NET元素 204
在XDE中建模.NET元素 216
建立分析模型 235
处理.NET代码 258
练习：.NET应用程序从分析到设计 265
小结 268
第7章 用XDE建模数据库 269
对象模型与数据模型 269
生成数据模型 270
处理视图 290
从数据模型生成对象模型 293
从对象模型生成数据模型 294
从数据模型生成数据库 295
更新现有数据库 297
逆向设计数据库 299
练习：对时间记录系统生成数据库 299
小结 305
第8章 模型发表与报表 306
生成报表 306
发表XDE模型 308
练习：模型发表与报表 310
小结 313

第三部分 模式 315

第9章 使用模式 315
什么是模式 315
为什么使用模式 316
Pattern Explorer 316
采用模式 321

关联与扩展模式 326
使用模式收藏 327
练习：采用模式 328
小结 330
第10章　“四人帮”模式 331
生成性模式 331
结构性模式 342
行为性模式 358
练习：考虑模式 392
小结 392
第11章　可复用资源规范（RAS）简介 393
定义资源 393
使用资源 394
XDE中的资源 400
小结 403
第12章　生成模式 405
生成模式 405
模式库 425
关联与扩展模式 426
生成模式版型 427
将模式变成可复用资源规范 433
练习：生成模式 434
小结 439
附录　UML入门 440

第一部分　用Rational XDE自由自在地开发

第1章　Rational XDE简介

Rational公司开发了一个激动人心的新工具，专门供Java与Microsoft.NET应用程序设计人员与开发人员使用。这个工具把Rational Rose的建模功能与编程工具的集成开发环境（IDE）结合了起来。本章将介绍这个新工具——扩展开发环境（XDE，Rational eXtended Development Environment）的一些特性，以及如何将其用于系统设计与开发。

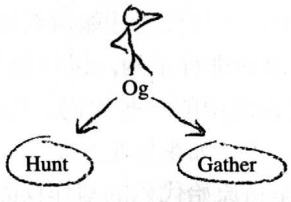
本章介绍下列内容：

- XDE的演变
- 可视建模简介
- Java与可视建模简介
- .NET与可视建模简介
- Rational XDE的特性
- XDE用户界面
- Rational XDE的版本

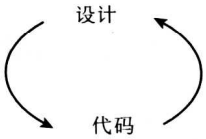
开始

有一天夜里，万籁无声，在亚热带草原的石墙周围，聚集着一群心情激动的穴居人。一只小小的蜥蜴爬进来，似乎感觉到了空气中紧张的气息，又悄悄地爬走了。穴居人的头领神色紧张地在石头上刻着一些标志，嘴里叽哩咕噜，另一个人则念出一个名称。渐渐地，一个明晰的形状出现了，然后是一个卵形。当头领在其他符号之间刻画出一条细线时，人群中有人开始大叫起来，开始指指点点。随着这些项目名称的不断列出，人们的心情也越来越激动："角色（Actor）"，"用例（Use Case）"。在这天晚上，一个系统诞生了。

这就是XDE之前的事情。

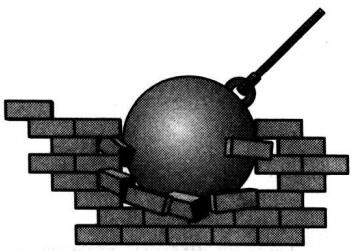


当然，可能没有这么原始，但我们的确经历了较长时间的演变。过去，我们用Rational Rose之类的统一建模语言工具来设计系统和体统结构，然后再将这些东西放进IBM公司的WebSphere之类的开发环境中，编写设计规范的实际代码。而UML工具可以从设计中自动生成一些框架代码，以提供过渡帮助。一旦采用新工具，则要改变系统设计，因此又要回到UML工具。设计中稍做修改后，还要切换回开发工具中，增加一些代码。这样，我们就要用UML工具的双向设计特性来回切换。一旦要改变系统设计，则又要回到UML工具。过一阵子，可能又要进行一些改变。也许我们忘了将这些改变同步回模型中，使模型与代码失去同步。经过一两次折腾之后，我们就会有点沮丧了。



这样下去，我们仍然可以设计与编码系统，但在不同工具间来回切换是很头痛的事。一旦模型与代码失去同步，则重新同步的难度越来越大。

一个问题是，来回不断切换环境会使焦点转移。我们要放下设计员的身份，以编程员的身份在WebSphere中工作，然后要放下编程员的身份，以设计员的身份在Rose中工作。这样来回变换很难得到良好的代码。开发人员在建立代码时还要考虑系统设计。也许，他们在建立代码时，突然发现设计中的某个方面可以进行一些优化。另一方面，设计员在建立系统设计时要牢记代码。在许多项目中，设计员与开发员常常是同一个人。除了角色重叠的问题外，他们还要使用多种工具并来回切换，才能完成任务。但Rational XDE使这一切得到改变。



XDE的目的就是拆掉设计与编辑之间的藩篱，让我们记住设计和编码只是同一方案的两个方面。就像不能隔着墙把要求扔给设计人员一样，也不能隔着墙把系统设计扔给开发人员。XDE可以让我们随时看到设计和编码，而不是在设计和编码之间来回切换。换句话说，XDE把设计与开发环境组合起来，形成了一个综合性工具。本书将花大量时间介绍XDE的设计和编码两个方面，介绍它们是如何集成到系统中的。

开发小组面临的另一个难题是让设计与编码保持同步。尽管在这个过程中可以使用一些帮助工具，但是劳动量仍然很大，常常会使设计和编码失去同步，结果，使系统变得很难维护。各小组必须浏览几百甚至几千行代码才能对系统设计有个明确了解。通常，改进其中一个系统就要在设计的相当大部分进行重构，以便触及这个系统并进行改变。

有时开发小组要重新处理系统中的一些代码，使其符合当前标准或利用新思想。这种改造在系统第一次开发时就会发生。如果你是开发人员，则可以看看项目初期编写的代码和项目后期编写的代码。你是想坚持原始代码的写作权呢，还是想要进行一些修改之后再公诸于众？

这种改造代码的概念也称为重新代管（refactor），此类工作要经常进行，但它是杂凑的而不是规范的。这种改造的结果是有些部分优化和标准化了，而有些部分则保持不动。不同的人可能改造系统中的不同部分，使用不一致的方法。这样就无法实现改造的许多真正好处——代码的标准化、优化、一致性与可读性。随着项目的进行，一定要寻找机会改进设计与编码。但既然改造这么重要，为什么没有更多的公司干这种事呢？我们听到的理由包括：

- 完成项目的时间已经不够了，更何谈改善。
- 改造过程是手工的，很难进行。
- 开发人员喜欢干新鲜的激动人心的事，而不喜欢处理现有的东西。

有这么多理由，自然改造的事就很少有人干了。但是，XDE可以使这个过程更简单也更有趣。当然，XDE无法打破时间与空间的法则，让人们在项目中有更多时间；但它可以帮我们，使设计与开发过程中一些麻烦的工作变得更方便更快捷。例如，XDE可以使随时保持同步，让开发人员把投在这方面的时间用于改造。XDE还可以减少使用模式所需的时间。出现XDE之前，要选择所要模式，进行设计，然后进行编码。现在则可以选择所要模式，然后用一个向导完成工作。这样又可以把省下的时间用于改造，或是准点回家！虽然XDE能帮我们腾出这么多用于改造的时间，但我们还有必要了解如何减少改造本身的工作。由于模型与代码保持同步，因此只需改造其中一个方面。如果没有同步，则要改造模型与代码，然后手工检查，以保证同步。

最后，分开设计与开发工具时遇到的另一个问题是很难复用。复用性是面向对象开发的神圣目标，但很少有开发人员真正实现了这个目标。原因包括：

- 项目小组没时间寻找其他项目中的可复用部分。
- 小组无法浏览可复用项目的仓储库。找到可复用项目时，已经失去了复用带来的时间优势。
- 每个项目差别很大，很难从其他项目中找到可复用部分。

提高复用率是Rational公司建立XDE的主要原因之一。它把设计与实现模式集成到一个环境中，可以利用其中的工具浏览和集成多个模型的元素。读者不必研究书本中的模式，学会其中所需模式的用法，然后手工设计与编码这个模式；相反，可以利用几个向导把模式加进模型与代码中。由于XDE能够在微粒层实现模式，因此可以使用在几乎任何系统中都适用的模式。即使公司的业务过程很独特，也可以充分利用复用。本书将花大量篇幅介绍模式与改造，介绍XDE在模式与改造方面有何帮助。

可视建模简介

顾名思义，可视建模就是把软件系统的设计与体系结构用可视的方式表现出来。这个概念由来已久，但标记方法经常改变。在大学里，我们编写代码之前要画流程图（实际上大多数人是写完代码之后才画流程图的，但这是另一回事）。可视建模实际上就是从这个概念演变来的，它是编码之前建立的蓝图，用以保证代码具有坚实的设计基础。

可视建模的一大优点是可以比较方便地改变软件系统。看看编写代码之前画流程图的情形。最终建档设计虽然好，但设计改变时则可能很困难，特别是做出大改动时。如果建立了可视建模，则可以分析模型，否定它，重构系统体系结构，然后才开始建立编码。在系统

寿命期间，这个方法可以节省成千上万的维护成本。

这在大规模系统中特别重要。如果没有蓝图，则不能保证不同开发人员所生成代码的集成，也不能保证系统不同部分采用相同的设计方法，并且具有代码一致性。要了解系统结构，惟一的方法是检查代码，想像系统如何集成，如何修改。

建筑大厦时，我们不会这么干（谁也不会不用蓝图就开始建摩天大楼），但信息技术行业的人们似乎经习惯这么干了，至少在一定程度上如此。这是因为，信息系统是比较独特的，可以对已经建立的系统改变体系结构。虽然不一定容易，也不一定便宜，但至少是可能的。但是，这种习惯的成本很高，信息技术行业正在放弃这种不规范的应用程序设计方法，转而使用更加系统化的方法。

可视建模的最新发展是建立UML，XDE使用了UML，本书也将大量使用UML。UML是一系列框图和一系列标注，专门用于可视建模系统设计。UML是20世纪90年代中期由三个方法学家Grady Booch、Ivar Jacobson与James Rumbaugh开发的（称为“三剑客”），用于统一他们的不同建模方法。这三位专家和其他方法学家一起，集成了许多不同建模标注系统中的最佳做法，如面向对象软件工程（OOSE，Object-Oriented Software Engineering）、Booch与对象建模技术（OMT，Object Modeling Technology）。UML 0.9版是1996年发布的。1997年，Rational软件公司组织的个人与公司联盟为响应对象管理组织（OMG）的要求，提出了描述软件集中系统的公共元模型。

随着时间的推移，UML不断修改与扩展，增加了新技术与新方法中的思想，如Web应用程序开发与规范业务建模。但是，UML依然独立于任何特定工具、编程语言和开发方法。事实上，这是个通用标注系统，可以分析与设计任何面向对象系统。

UML目前由对象管理组织（OMG）控制与维护，这是个厂家独立的标准组织。许多大小公司都和对象管理组织（OMG）一起为UML的增长与发展作着贡献。关于对象管理组织（OMG）的更多信息或最新UML规范，见www.omg.org。

UML包含许多不同类型的框图，见表1.1。每个框图描述系统的不同方面。例如，用例框图从客户角度看系统，类框图是从开发人员角度建立的，而组件框图是从系统集成人员角度建立的。这些框图加要一起，使小组可以得到系统及其体系结构的完整面貌。

表1.1 UML框图

框图类型	作用
用例	向客户显示项目范围中的功能（用例），谁是系统角色（任何与系统交互的人和事）
活动	显示业务过程的工作流程，或显示用例的步骤
协作	显示参与用例的对象，对象之间的关系和对象之间发送的消息
顺序	显示参与用例的对象，对象之间的关系和对象之间发送的消息。顺序框图与组件框图提供相同的信息，但格式不同
类	描述系统中的类（及其子集）及其关系、属性与操作。类框图也可以显示类或子系统的软件包
层次	显示对象的动态行为，包括其存在状态，状态之间如何过渡，每个状态如何表现
组件	显示构成系统的物理组件及其相互依赖性
部署	描述系统如何部署

利用UML框图提供的完整蓝图，小组成员可以进行讨论，然后记录系统体系结构方面的决策。更重要的是，框图成为信息仓储库，小组成员可以用其更有效地相互交流。任何人要查看与系统设计相关的信息时，只要检查UML框图即可。

UML很快成为了行业中使用的标准建模语言，并且已经成为多个组织采用的标准，支持UML标注的工具越来越多。

UML 1.4的新特性

最新的UML版本为1.4，是2001年公布的，这个版本中的主要改变包括：

改进了组件建模 更好地支持为使用Enterprise JavaBeans（EJB）与COM+的组件系统进行建模。

改进了协作与协作实例 UML规范正式描述了协作（一组角色及关联，用于定义完成特定任务所需的参与者）、交互（参与者之间的通信）、实例和协作实例组。新的UML规范还引入了参数化协作的概念，这是表示设计模式的一般化方式。这些协作可以用于建档设计模式与框架。

改进了配置文件 配置文件用于建模UML的特定实现。UML规范明确定义了如何针对特定域定制和扩展UML。它要求配置文件扩展（而不是修改）建档的UML规范。

此外，UML规范中还有许多其他改变和改进。UML规范的完整拷贝见www.omg.org站点。目前正在开发这个标准的新版本2.0。可视建模已经越来越重要。事实上，UML 2.0准备为可执行模型铺平道路。详见OMG的Web站点。

Java与可视建模简介

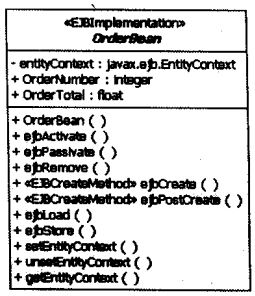
由于UML非常丰富，而且Java纯粹是面向对象的语言，因此大部分Java结构很容易加进UML模型中。UML可以用于Java 2标准版（J2SE）或Java 2企业版（J2EE）。

J2SE是20世纪90年代开发的面向对象语言。Java是个强类型语言，严格执行面向对象结构。其结构与C++相似，但不用指针。Java可以生成应用程序、Web小程序和其他类型的系统。J2SE支持并发性和持久性，J2EE更适合这些类型的使用。

J2EE的核心是一组类和接口，开发人员可以用其更方便地生成数据库和企业应用程序。小服务与EJB是J2EE中使用的主要结构，使开发人员可以编写瘦客户端，与应用程序服务器中运行的进程建立接口。EJB甚至可以自动处理会话信息的持久性，具有与其他EJB通信的内置功能。

第5章“在XDE中建模Java与J2EE元素”将介绍Java建模与XDE的细节。本节简要介绍标准J2SE和J2EE元素与UML的对应关系。

Java类 每个Java类在UML中表示为下图所示的分隔矩形。



这个矩形可以分解为下列组件：

类名 矩形上方显示有类名。模型中的类名应与代码中的类名相同。如果是抽象类，则类名用斜体字表示。

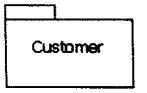
版型 上方还显示类的版型，版型是个UML机制，描述一个元素的不同类型。例如，接口与EJB会话Bean是特殊类型的类，都可以看成类版型。

属性 矩形中部存储类的变量（或属性）。属性名后面是冒号和数据类型。变量可以用类或与另一类的关系建模。

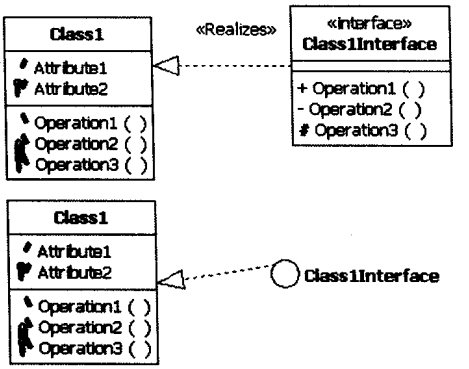
操作 矩形下方是类的方法（或操作）。

包

UML包与Java包相似。此外，UML还支持子系统，就是实现一个或几个接口的类包。包的UML符号如下：



Interfaces与Implements关键字 UML中的Java接口显示为类，版型为<<Interface>>。框图中可以用两种方法表示实现接口的类：用实现关系（如上图所示）或用接口的“棒糖”符号（如下图所示）。



Extends关键字 UML中Extends关键字表示为一般化关系。两个类之间的这种箭头表示继承关系。