经典
清华版
RUANJIAN RUMEN YU TIGAO CONGSHU
软件入门与提高丛书

新手入门　逐步进阶　实战提高
图解教学　范例练习

UML2

软件建模入门与提高

杨晓军　编著

影响百万人的经典清华版
全新改版震撼上市

清华大学出版社

软件入门与提高丛书

UML2 软件建模入门与提高

李 勇 杨晓军 编 著

清华大学出版社
北 京

内 容 简 介

本书从初学者的角度出发，由浅入深、循序渐进地介绍统一建模语言 UML 的相关知识，书中提供了大量操作 UML 的示例。另外，还向读者提供了很多实战案例和上机练习，用于演练。

本书共分为 16 章，内容包括面向对象思想和软件建模分类，UML 发展历史、组成元素、体系结构、建模流程和应用领域，常用的 UML 建模工具，用例图、类图、对象图和包图、状态机图、活动图、顺序图和时间图、通信图和交互概览图、组件图和部署图，UML 到关系型数据库的映射，UML 与统一过程，UML 与 Java 语言的映射，以及 UML 与设计模式等。最后一章提供了一个综合的案例。

本书示例新颖，内容丰富，涉及面广泛，适合所有的 UML 初学者学习，也可以帮助有基础知识的读者提高创建 UML 模型图的技能。另外，对于大中专学生和培训班的学生来说，本书更是一本不可多得的教材和自学用书。

图书在版编目(CIP)数据

UML2 软件建模入门与提高/李勇，杨晓军编著. --北京：清华大学出版社，2015
(软件入门与提高丛书)
ISBN 978-7-302-38610-0

Ⅰ. ①U…　Ⅱ. ①李…　②杨…　Ⅲ. ①面向对象语言—程序设计　Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2014)第 276811 号

责任编辑：杨作梅　宋延清
装帧设计：刘孝琼
责任校对：周剑云
责任印制：杨　艳

出版发行：清华大学出版社
　　网　　址：http://www.tup.com.cn，http://www.wqbook.com
　　地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
　　社 总 机：010-62770175　　**邮　　购**：010-62786544
　　投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
　　质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
　　课 件 下 载：http://www.tup.com.cn,010-62791865
印 装 者：清华大学印刷厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：27　　**字　　数**：658 千字
版　　次：2015 年 1 月第 1 版　　**印　　次**：2015 年 1 月第1 次印刷
印　　数：1～3000
定　　价：56.00 元

产品编号：057495-01

前　言

UML 是英文 Unified Modeling Language 的缩写，又称为统一建模语言，或者标准建模语言，它是始于 1997 年的一个 OMG 标准，是一个支持模型化和软件系统开发的图形化语言，可为软件开发的所有阶段提供模型化和可视化支持。面向对象的分析与设计方法在 20 世纪 80 年代末至 90 年代中出现了一个高潮，UML 正是这个高潮的产物，它不仅统一了 Booch、Rumbaugh 和 Jacobson 的表示方法，而且做了进一步的发展，并最终统一为大众所接受的标准建模语言。

本书内容

全书共分 16 章，主要内容如下。

第 1 章：面向对象和软件建模。从模型开始介绍，接着介绍面向对象思想的三大要素、三大模型、常用的三层开发方法、软件建模知识，最后介绍常用的建模分类。

第 2 章：UML 入门基础。着重介绍 UML 的基础知识，包括 UML 的概念、发展历史、目标、组成元素、体系结构、建模流程以及应用领域等内容。

第 3 章：UML 建模工具。从目前众多的 UML 建模工具中挑选出应用最广泛且在建模工具中最有影响力的 3 种工具(Visio、Enterprise Architect 和 PowerDesigner)进行介绍。

第 4 章：用例图。介绍用例图的构成、设计和使用，包括用例图的组成部分、各个部分成员的确定和使用，以及如何绘制完整的用例图等。

第 5 章：类图。详细介绍 UML 中的类图，包括类图中的类、抽象类、接口和各种关系的实现等内容，还介绍如何使用类图进行建模。

第 6 章：对象图和包图。首先介绍对象图，包括概念、组成、绘制和阅读，以及如何建模等；然后介绍包图，包括概念、组成、分类、设计原则以及如何建模等。

第 7 章：状态机图。详细介绍状态机图的绘制，首先介绍状态机图的基本内容，包括概念、标记、状态类型和状态机图的应用等，然后介绍状态机图中的转移元素，最后介绍组合状态。

第 8 章：活动图。详细介绍活动图的绘制，其内容包括活动图的定义、作用、与状态机图的区别、组成元素和活动转换等。

第 9 章：顺序图和时间图。介绍 UML 中的两种交互图，即顺序图和时间图。顺序图描述系统对象之间的交互顺序，但是这个顺序没有细致的时间刻度，只是一个大概的流程，而时间图弥补了这个不足，它们共同绘制了系统对象间交互的顺序和时间。

第 10 章：通信图和交互概览图。首先对通信图的概念进行介绍，然后介绍通信图中的消息、对象的创建和消息迭代等内容，最后介绍交互图和交互概览图的绘制。

第 11 章：绘制 UML 的实现图。首先介绍组件图的绘制和建模，然后介绍部署图的绘制和建模。

第 12 章：UML 到关系型数据库的映射。着重介绍如何将 UML 类图中的类和关系映射到关系型数据库表。首先介绍基本结构的映射，然后介绍泛化关系和关联关系的映射，最后介绍完整性与约束检查以及存储过程、触发器和索引等内容。

第 13 章：UML 与统一过程。首先讨论软件开发过程和成熟标准，然后详细介绍一种使用 UML 的过程，即统一过程，最后简单介绍使用 UML 过程的一般特征。

第 14 章：UML 与 Java 语言映射。主要介绍 UML 类图映射为 Java 语言实现的方法，包括转换为 Java 类、转换原则、转换类之间的关联、泛化关联以及包和接口等。

第 15 章：UML 与设计模式。首先介绍模式的一些基本概念，接着介绍 UML 对设计模式的支持，然后通过具体的示例讨论如何使用设计模式进行系统设计。

第 16 章：即时通信系统。综合 UML 建模系统的各类模型，通过对即时通信系统的分析，绘制该系统的 UML 模型图，包括用例图、静态图、行为图和交互图等多种图形。

本书特色

本书内容详细、示例丰富，知识面广，全面地讲解了 UML 的应用和开发。本书最大的特点体现在如下几个方面。

(1) 知识全面，内容丰富

本书紧密围绕 UML 的相关知识展开详细的讲解，涵盖了实际开发应用中的具体应用代码。

(2) 理论和示例结合

书中几乎每一个知识点都有丰富而典型的示例，而且每一章最后都会通过一个或多个综合的实战介绍该章的知识。作为一本 UML 入门类型的书，将理论和实践很好地结合起来进行讲解，让读者最容易快速掌握。

(3) 应用广泛，提供文档

对于大多数的精选实战案例，都会向读者提供详细的实现步骤，结构清晰简明，分析深入浅出，而且有些实战案例贴近实际。

(4) 网站技术支持

读者在学习或者工作的过程中，如果遇到实际问题，可以直接登录 www.itzcn.com 与我们取得联系，作者会在第一时间给予帮助。

(5) 贴心的提示

为了便于读者阅读，书中还穿插着一些技巧、提示等小贴士，体例约定如下。

- 提示：通常是一些提醒，让读者加深印象或提供建议及解决问题的方法。
- 注意：提出学习过程中需要特别注意的一些知识点和内容，或者相关信息。
- 技巧：通过简短的文字，指出知识点在应用时的一些小窍门。

读者对象

本书适合作为软件开发入门者的自学用书，也适合作为高等院校相关专业的教学参考书，还可供开发人员查阅、参考。本书特别适合下列人员阅读：

- UML 初学者。
- 各大中专院校的在校学生和相关授课教师。
- 准备从事与 UML 应用相关工作的人员。

作者团队

除了封面署名作者之外，参与本书编写的人员还有程朝斌、王咏梅、郝军启、王慧、郑小营、张浩华、王超英、张凡、赵振方、张艳梅等，在此表示感谢。在本书的编写过程中，我们力求精益求精，但难免存在一些不足之处，恳请广大读者批评指正。

编　者

目　录

第 1 章　面向对象和软件建模 1

1.1　模型 2
1.2　面向对象的思想 3
1.2.1　了解面向对象 3
1.2.2　面向对象的三大要素 5
1.2.3　面向对象的三大模型 7
1.2.4　面向对象的常用三层 8
1.2.5　面向对象的开发方法 10
1.3　软件建模 11
1.3.1　软件建模概述 11
1.3.2　建模的三要素 11
1.3.3　面向对象建模 12
1.4　建模分类 14
1.4.1　业务建模 14
1.4.2　数据建模 16
1.4.3　应用程序建模 16
1.5　思考与练习 17

第 2 章　UML 入门基础 19

2.1　UML 概述 20
2.1.1　UML 简介 20
2.1.2　UML 发展历史 21
2.1.3　UML 的目标 22
2.2　UML 的基本组成 23
2.2.1　建模元素 23
2.2.2　关系 26
2.2.3　图 28
2.2.4　规则 30
2.2.5　通用机制 31
2.2.6　UML 标准通用机制 34
2.3　UML 其他内容 40
2.3.1　UML 的体系结构 40
2.3.2　UML 建模流程 41
2.3.3　UML 的应用领域 42
2.4　UML 2.0 概述 42
2.5　思考与练习 44

第 3 章　UML 建模工具 45

3.1　使用建模工具需知 46
3.1.1　建模工具的作用 46
3.1.2　选择建模工具的方法 47
3.1.3　常用建模工具 48
3.2　Visio 2010 49
3.2.1　Visio 2010 简介 49
3.2.2　实战——绘制论坛系统的用例图 51
3.3　Enterprise Architect 8 54
3.3.1　Enterprise Architect 8 简介 54
3.3.2　实战——绘制论坛系统的类图 57
3.4　PowerDesigner 16.5 61
3.4.1　PowerDesigner 简介 62
3.4.2　实战——安装 PowerDesigner 16.5 63
3.4.3　实战——绘制活动图 65
3.4.4　实战——生成模型报告 68
3.4.5　实战——对 MySQL 进行反向工程 70
3.5　思考与练习 73

第 4 章 用例图 75
4.1 用例图简介 76
4.2 用例图的构成 78
4.2.1 系统 78
4.2.2 参与者 79
4.2.3 用例 79
4.2.4 关系 80
4.3 使用参与者 81
4.3.1 参与者的确定 81
4.3.2 参与者的使用 82
4.4 用例的使用 83
4.4.1 识别用例 83
4.4.2 用例描述 87
4.5 关系 88
4.5.1 关联关系 88
4.5.2 泛化关系 89
4.5.3 包含关系 90
4.5.4 扩展关系 91
4.6 实战——图书馆管理系统用例图 92
4.7 思考与练习 97
第 5 章 类图 101
5.1 类图和元素 102
5.1.1 类图概述 102
5.1.2 类 104
5.1.3 抽象类 109
5.1.4 接口 109
5.2 依赖关系 110
5.2.1 依赖关系概述 110
5.2.2 依赖关系分类 111
5.3 关联关系 113
5.3.1 关联关系概述 113
5.3.2 常见的关联 118
5.3.3 聚合关联 121
5.3.4 组合关联 122
5.3.5 建立关联 123
5.4 泛化关系 123
5.4.1 泛化关系概述 124
5.4.2 常用的泛化 125
5.4.3 泛化集 126
5.4.4 泛化约束 127
5.5 实现关系 128
5.6 类图建模步骤 130
5.7 实战——构建病房监护系统的类型 131
5.8 思考与练习 133
第 6 章 对象图和包图 137
6.1 了解对象 138
6.1.1 对象概述 138
6.1.2 对象符号 139
6.2 对象图 142
6.2.1 对象图概述 142
6.2.2 绘制和阅读对象图 143
6.2.3 使用对象图建模 144
6.3 对象图和类图 144
6.4 实战——绘制订单管理系统的对象图 145
6.5 了解包 146
6.5.1 包概述 146
6.5.2 包的符号 147
6.6 包图 149
6.6.1 包图概述 149
6.6.2 包图分类 150
6.6.3 包导入和包合并 152
6.6.4 使用包图建模 154
6.6.5 包图设计原则 154
6.7 实战——绘制剧院系统的包图 157
6.8 思考与练习 159

第 7 章 状态机图 161
7.1 状态机图简介 162
7.1.1 状态机概述 162
7.1.2 状态机标记 163
7.1.3 状态类型 165
7.1.4 状态机图的应用 166
7.2 转移 167
7.2.1 转移简介 167
7.2.2 事件 168
7.2.3 动作 170
7.2.4 活动与延迟事件 171
7.2.5 转移的类型 172
7.3 组合状态 174
7.3.1 顺序状态 174
7.3.2 并发状态 174
7.3.3 同步状态 175
7.3.4 历史状态 175
7.3.5 子状态机引用 176
7.4 实战——自动存取款系统状态机图 178
7.5 思考与练习 180
第 8 章 活动图 181
8.1 活动图的基本概念 182
8.1.1 活动图的定义 182
8.1.2 活动图的作用 182
8.1.3 活动图的主要元素 183
8.1.4 理解活动与动作 184
8.1.5 活动图与状态图的区别 186
8.2 活动图的元素详解 186
8.2.1 动作状态 186
8.2.2 活动状态 186
8.2.3 开始和结束状态 187
8.2.4 分支与合并 188
8.2.5 分叉与汇合 190
8.2.6 对象流 191
8.2.7 泳道 192
8.3 活动转换 193
8.3.1 转移 193
8.3.2 判定 194
8.3.3 发送和接收信号动作 195
8.3.4 事件和触发器 195
8.3.5 可中断区间 196
8.3.6 异常 197
8.4 实战——活动图的应用 198
8.5 思考与练习 200
第 9 章 顺序图和时间图 203
9.1 顺序图简介 204
9.2 顺序图的构成 205
9.2.1 对象 205
9.2.2 生命线和激活期 208
9.2.3 消息 209
9.2.4 序号 210
9.2.5 参数 212
9.2.6 激活期规范 213
9.3 消息类型 217
9.3.1 同步消息 217
9.3.2 异步消息 218
9.3.3 反身消息 219
9.3.4 接收发送消息 220
9.3.5 消息分支和从属流 220
9.4 组合片段 222
9.4.1 组合片段简介 222
9.4.2 选项组合片段 224
9.4.3 备选组合片段 225
9.4.4 循环组合片段 226
9.4.5 引用组合片段 227
9.5 时间图 228
9.5.1 时间图概述 228

9.5.2 时间图的构成.....229
9.5.3 时间约束.....234
9.5.4 替代表示法.....235
9.6 实战——团购系统顺序图.....236
9.7 思考与练习.....241
第 10 章 通信图和交互概览图.....245
10.1 通信图简介.....246
10.1.1 通信图概述.....246
10.1.2 对象与类角色.....247
10.1.3 关联角色与链接.....248
10.1.4 消息.....250
10.2 消息序号和控制点.....251
10.2.1 消息序号.....252
10.2.2 消息控制点.....252
10.3 创建对象.....253
10.4 消息迭代.....253
10.4.1 对象的迭代.....254
10.4.2 消息的迭代.....254
10.5 交互图.....255
10.6 交互概览图.....256
10.6.1 交互概览图简介.....256
10.6.2 绘制交互概览图.....257
10.7 实战——在线报考系统的交互图.....257
10.8 思考与练习.....259
第 11 章 绘制 UML 的实现图.....261
11.1 了解组件.....262
11.1.1 组件概述.....262
11.1.2 组件的符号.....262
11.2 组件图.....265
11.2.1 组件图概述.....265
11.2.2 组件间的关系.....267
11.2.3 组件图和类图.....268
11.2.4 实战——绘制组件图.....268
11.3 使用组件图建模.....269
11.3.1 对源代码建模.....270
11.3.2 对可执行体的发布建模.....270
11.3.3 对物理数据库建模.....271
11.3.4 对可适应的系统建模.....272
11.4 了解节点.....272
11.4.1 节点的符号.....272
11.4.2 节点和组件的区别.....273
11.5 部署图.....274
11.5.1 部署图概述.....274
11.5.2 部署图的关系.....275
11.5.3 实战——绘制部署图.....276
11.6 使用部署图建模.....278
11.6.1 对嵌入式系统建模.....278
11.6.2 对客户/服务器建模.....279
11.6.3 对全分布式系统建模.....280
11.7 思考与练习.....281
第 12 章 UML 到关系型数据库的映射.....285
12.1 关系型数据库与 UML 模型.....286
12.1.1 关系型数据库管理系统.....286
12.1.2 UML 模型.....286
12.2 基本结构映射.....287
12.2.1 主键的生成.....287
12.2.2 属性类型到域的映射.....288
12.2.3 属性到列的映射.....289
12.3 泛化关系的映射.....289
12.3.1 所有类的映射.....289
12.3.2 除无属性外类的映射.....291
12.3.3 父类属性下移.....291
12.3.4 子类属性上移.....292
12.3.5 映射方法比较.....293
12.4 关联关系的映射.....294
12.4.1 一对一关联的映射.....294

12.4.2 零或一对一关联的映射.....295
12.4.3 一对多关联的映射.....295
12.4.4 多对多关联的映射.....297
12.4.5 聚合和组合关系的映射.....298
12.4.6 映射时应避免的情况.....298
12.5 完整性与约束检查.....299
12.5.1 父表的约束.....300
12.5.2 子表的约束.....301
12.6 其他相关问题.....302
12.6.1 存储过程.....302
12.6.2 触发器.....302
12.6.3 索引.....303
12.7 实战——软件公司 UML 模型的映射.....303
12.8 思考与练习.....305
第 13 章 UML 与统一过程 309
13.1 软件开发过程简介.....310
13.2 定义和理解软件工程的过程.....311
13.3 软件成熟标准：CMM.....311
13.3.1 使用 CMM 的意义.....311
13.3.2 CMM 等级.....312
13.3.3 CMM 框架.....314
13.3.4 CMM 结构.....315
13.4 RUP 简介.....316
13.4.1 使用 RUP 的意义.....316
13.4.2 什么是 RUP.....317
13.4.3 RUP 的特点.....318
13.4.4 RUP 六大开发经验.....320
13.5 RUP 二维开发模型.....321
13.5.1 时间维.....322
13.5.2 RUP 的静态结构.....324
13.6 RUP 工作流程.....326
13.6.1 初始阶段.....326
13.6.2 细化阶段.....327
13.6.3 构造阶段.....328
13.6.4 交付阶段.....329
13.7 RUP 的核心工作流.....329
13.7.1 商业建模.....329
13.7.2 需求.....329
13.7.3 分析和设计.....331
13.7.4 实现.....332
13.7.5 测试.....332
13.7.6 部署.....333
13.7.7 配置和变更管理.....333
13.7.8 项目管理.....334
13.7.9 环境.....334
13.8 如何在过程中使用 UML.....334
13.8.1 以架构为中心.....334
13.8.2 用例驱动.....335
13.8.3 UML 对迭代开发的支持.....335
13.8.4 UML 图与工作流程之间的关系.....336
13.9 思考与练习.....336
第 14 章 UML 与 Java 语言映射 339
14.1 模型映射为 Java 的实现.....340
14.1.1 转换为 Java 类.....340
14.1.2 转换原则.....341
14.2 实现常见关联.....342
14.2.1 基本关联.....342
14.2.2 强制对可选或者强制关联.....343
14.2.3 可选对可选关联.....344
14.2.4 可选对多关联.....345
14.2.5 强制对多关联.....346
14.2.6 多对多关联.....346
14.2.7 关联类的实现.....347
14.2.8 聚合关联的实现.....348
14.2.9 组合关联的实现.....348

14.3 实现泛化.....349
14.4 特殊模型的映射.....350
14.4.1 包.....350
14.4.2 接口.....350
14.4.3 枚举.....352
14.5 实战——类图与 Java 的工程化.....352
14.5.1 正向工程.....352
14.5.2 逆向工程.....355
14.6 思考与练习.....357
第 15 章 UML 与设计模式 359
15.1 了解模式.....360
15.2 软件设计模式.....360
15.2.1 了解设计模式.....360
15.2.2 设计模式的诞生.....362
15.2.3 设计模式的原则.....363
15.2.4 设计模式的分类.....364
15.3 设计模式的元素.....366
15.3.1 关键元素.....366
15.3.2 其他元素.....367
15.4 创建型模式.....367
15.4.1 了解创建型模式.....368
15.4.2 简单工厂模式.....368
15.4.3 工厂方法模式.....372
15.4.4 抽象工厂模式.....375
15.4.5 单例模式.....380
15.5 结构型模式.....384
15.5.1 了解结构型模式.....384
15.5.2 适配器模式.....385
15.5.3 外观模式.....388
15.6 观察者模式.....390
15.6.1 了解观察者模式 390
15.6.2 实战——自定义观察者模式 391
15.7 思考与练习 393
第 16 章 即时通信系统 395
16.1 系统建模概述 396
16.1.1 系统开发的背景 396
16.1.2 系统建模的基本步骤 396
16.2 系统分析 397
16.2.1 系统结构 397
16.2.2 需求分析 398
16.3 用例图 398
16.3.1 确定参与者 399
16.3.2 确定用例 399
16.3.3 绘制系统用例图 400
16.4 静态图 402
16.4.1 即时通信类 402
16.4.2 即时通信类图 404
16.5 行为图 405
16.5.1 行为分析 405
16.5.2 用户聊天的活动图 406
16.5.3 可疑言论处理活动图 406
16.6 交互图 408
16.6.1 用户登录顺序图 408
16.6.2 离线消息顺序图 409
16.6.3 言论处理顺序图 409
16.6.4 在线通信顺序图 410
16.6.5 交互概览图 411
16.7 组件图 412
附录 各章思考与练习答案 415

第1章

面向对象和软件建模

面向对象是软件开发的方法，它的概念和应用已经超越了程序设计和软件开发，扩展到如数据库系统、交互式界面、应用结构、应用平台、分布式系统、网络管理结构、CAD 技术以及人工智能等领域。软件建模则体现了软件设计的思想，在系统需求和系统实现之间架起了一座桥梁。软件工程师按照设计者建立的模型，能够开发出符合设计目标的软件系统，而且软件的维护和改变也是基于软件分析模型的。

本书所介绍的 UML 是一种定义良好的、易于表达的、功能强大的且普遍适用的建模语言，它不仅可用于建立软件系统的模型，同样也可用于描述非软件领域内的系统以及具有实时要求的工业系统等。但是在介绍 UML 的内容之前，本章会向读者介绍面向对象和软件建模的知识。

本章重点：

- 了解模型的作用和特点
- 熟悉面向对象的优点
- 掌握面向对象的三大要素
- 熟悉面向对象的三大模型
- 掌握面向对象的常用三层

- 熟悉面向对象的常用开发方法
- 了解软件建模的目的
- 熟悉软件建模的三要素
- 掌握面向对象建模的 3 个特征
- 熟悉建模的分类

1.1 模　　型

由于人们对复杂性的认识能力有限，从而会导致系统的设计者在系统设计之初无法全面理解整个系统，这时，就需要对系统进行建模。举例来说，开发商要建造一座大楼，为了方便工人在动工前对大楼有清晰的认识，通常需要完成这座大楼的建筑设计图。建模可以使设计者从全局上把握系统及其内部的联系，而不会导致陷入每个模块的细节之中。而模型可以使具有复杂关系的信息简单易懂，使人们容易洞察复杂的原始数据背后的规律，并且能够有效地让人们将系统需求映射到软件结构上。

简单地说，模型是对现实的简化和抽象化。它是抓住现实系统的主要方面而忽略次要方面的一种抽象。因此，可以说模型既反映现实系统，却又不等同于这个现实系统。模型是理解、分析、开发或者改造现实系统的一种常用手段。图 1-1 显示了模型与现实系统之间的关系。

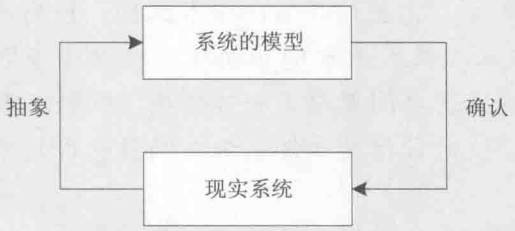

图 1-1　模型与现实系统之间的关系

模型是基于图形的表示，它以可视化方式、形象直观地描述系统的特征。一个模型往往针对同一个被建模事物，由多个图形组成，这些图大致可以分为结构图和行为图两类，分别描述事件的结构特征和行为特征。模型具有多种作用，说明如下。

(1) 促进项目有关人员对系统的理解和交流。

模型对于问题的理解、项目有关人员(例如客户、分析人员和设计人员等)之间的交流、文档的准备以及程序和数据库的设计都非常有益。它能促进人们对需求的理解，从而有利于人们直接研究一个大型的复杂软件系统。

(2) 缩短系统的开发周期。

模型实际上是通过过滤掉一些不必要的细节而刻画复杂问题或者结构的必要特性的抽象，它使问题更加容易理解。有了模型以后，软件系统的开发过程变得更快，同时也降低

了系统的开发成本。

(3) 有助于挑选出代价较小的解决方案。

在研究一个比较大型的软件系统的模型时，人们可以提出多个实际方案并对它们进行相互比较，然后挑选出一个最好的方案。

现实生活中模型的例子有很多。例如，房地产公司为了提前销售未完工的楼盘，通常会建立一个楼盘模型，其目的是使用户了解楼盘的性质，能够快速确认是否满足要求。楼盘模型不仅形象直观地说明每一栋楼的位置、层高、外观和采光条件等信息，还能表现出各个单元房间的结构布局，也能表现出周围道路、花草和娱乐场所等设施。

在一个软件工程中，要开发的软件具有复杂性，而且有多种角色的人员参与，往往需要建立一系列的软件模型，使不同的人员能进行交流、达成共识、协作配合。

模型具有 4 个特点，说明如下：

- 模型是局部性的，以反映事物的不同侧面。
- 模型不是实际的、物理性的系统，而是抽象的，且有不同的抽象级别。
- 模型的目的非常明确，一个模型总是出于特定的目的或意向而建立的。
- 模型与原型不同。原型是一个缩小的、局部的、可执行的系统；而模型无论多么详细，具体的模型都是难以直接执行的。

1.2 面向对象的思想

面向对象是一种对现实世界理解和抽象的方法，是计算机编程技术发展到一定阶段后的产物。它是一种以对象为基础，以事件或消息来驱动对象执行处理的程序设计技术。

从程序设计方法上来讲，面向对象是一种自下而上的程序设计方法，它不像面向过程程序设计那样，一开始就需要使用一个主函数来概括出整个程序，面向对象程序设计往往从问题的一部分着手，一点一滴地构建出整个程序。

1.2.1 了解面向对象

在面向对象出现之前，传统的程序设计方法大都是面向过程的，还有一少部分是面向数据结构的。面向过程的程序设计结构清晰，为缓解软件危机做出了贡献。但是，它的模块独立性较差，各个模块之间的耦合度非常高，一个模块的修改可能会造成许多其他模块功能上的改变。因此，面向对象的程序设计方法应运而生。

面向对象是一种新兴的程序设计方法，它是对面向过程程序设计强有力的补充。它使用类、对象、继承、封装和消息等基本概念来进行程序设计，从现实世界中客观存在的事物(即对象)出发来构造软件系统，并且在系统构造中尽可能运用人类的自然思维方式。开发一个软件是为了解决某些问题，这些问题所涉及的业务范围称作该软件的问题域，问题域关注的不仅仅是软件本身，还可以是计算机体系结构和人工智能等。

使用 UML 进行系统建模时，必须弄清楚什么是对象，以及在系统的分析与设计过程中如何利用对象。

1. 面向对象程序设计的优点

面向对象程序设计特别有利于大型、复杂软件系统的生成，它的优点如下。

(1) 方便理解

问题空间与解空间的结构一致，符合人们的日常思维习惯，降低了大规模系统的分析和设计难度。

(2) 概念连贯

软件开发全过程始终以“类和对象”为中心概念，方便阶段性结果的跟踪、管理和持续演进。

(3) 结构良好

对象具有良好的内聚性和局部独立性，从而使软件体系结构的可靠性、可维护性和可扩展性显著增强。

(4) 便于复用

它表现在两个方面：一是对象的内聚性和“粒度”便于复用；二是继承机制为代码复用提供了内在支持。

2. 面向对象程序开发

面向对象只是一种思想，或者说是一种开发方法，而不是一种编程技术。它的最大好处在于帮助规划人员、开发者和客户清晰地表达抽象的概念，并将这些概念互相传达。面向对象的思想已经涉及到软件开发的各个方面，例如面向对象分析、面向对象设计、面向对象编程等。

(1) 面向对象分析

面向对象分析的英文是 Object Oriented Analysis，简称 OOA。它是面向对象方法从编程领域向分析领域发展的产物。从根本上讲，面向对象是一种方法论，不仅仅是一种编程技巧和编程风格，更是一套可用于软件开发全过程的软件工程方法，OOA 是其中的第一个环节。

OOA 的基本任务是运用面向对象方法，从问题域中获取需要的类和对象，以及它们之间的各种关系。

(2) 面向对象设计

面向对象设计的英文是 Object Oriented Design，简称 OOD。OOD 在软件设计生命周期中发生于 OOA 之后或者后期。在面向对象的软件工程中，OOD 是软件开发过程中的一个大阶段，其目标是建立可靠的、可实现的系统模型；其过程是完善 OOA 的成果，细化分析。

OOD 与 OOA 的关系是：OOA 表达了“做什么”，而 OOD 则表达了“怎么做”。简单地说，OOA 只解决系统“做什么”，不涉及“怎么做”；而 OOD 涉及解决“怎么做”的问题。

(3) 面向对象编程

面向对象编程的英文是 Object Oriented Programming，简称 OOP。它就是使用某种面向对象的语言，实现系统中的类和对象，并使得系统能够正常运行。在理想的 OO 开发过

程中，OOP 只是简单地使用编程语言实现 OOA 和 OOD 分析和设计的模型。

1.2.2 面向对象的三大要素

面向对象包含抽象、封装、继承、多态、关联、聚合和消息等多个特征。一般情况下，将封装、继承和多态称为面向对象的三大要素或三大特性。

1. 封装(Encapsulation)

封装是指把对象的状态(属性)和行为(动作)绑定到一起的机制，把对象形成一个独立的整体，并且尽可能地隐藏对象的内部细节。封装包含两个含义：一是把对象的全部状态和行为结合在一起，形成一个不可分割的独立整体，对象的私有属性只能够由这个对象的行为来读取和修改；二是尽可能隐藏对象的内部细节，对外形成一道屏障，这样与外界的联系只能通过外部接口来实现。

封装的信息隐藏作用反映了事物的相对独立性，这样使开发者可以只关心它对外所提供的接口，即能够提供什么样的服务，而不用去关注其内部的细节问题。例如，对于一台计算机，我们不需要知道它的具体实现细节(例如 CPU、主板、显示器和内存等是怎么制造和工作的)，只要知道它能够做什么(例如上网购物、充话费和聊天等)即可。

【例 1.1】

有编程经验的读者都知道，Java 或者 C#等语言开发的程序都会用到封装。例如，如下代码显示了 C#中的一段封装代码：

```
public class CustomerInfo
{
    private string customNo;          //客户编号
    private int customAge;            //客户年龄
    public string CustomNo
    {
        get { return customNo; }
        set { customNo = value; }
    }
    public int CustomAge
    {
        get { return customAge; }
        set
        {
            if (customAge < 10 || customAge > 100)
                customAge = 10;
            else
                customAge = value;
        }
    }
}
```

上面一段代码首先通过 private 声明两个私有变量，然后将它们封装为公有属性。在 CustomAge 中，还对 customAge 变量的值进行了判断。这样，用户可以直接调用该类的公有属性并进行赋值，而不必关心是怎么实现的，直接调用即可。