HZ BOOKS
PRENTICE HALL

彩色UML建模

Java Modeling in Color with UML

Enterprise Components and Process

（美） Peter Coad　Eric Lefebvre　Jeff De Luca　著
王海鹏　等译

- 著名OO大师Peter Coad代表作
- UML之父Grady Booch高度评价
- UML China专家王海鹏倾情翻译
- UML China首席专家潘加宇鼎力推荐

机械工业出版社
China Machine Press

彩色UML建模

Java Modeling in Color with UML

Enterprise Components and Process

（美） Peter Coad　Eric Lefebvre　Jeff De Luca 著
王海鹏　等译

机械工业出版社
China Machine Press

本书系统地介绍了如何运用彩色来构建UML模型，书中使用4种颜色来代表4种架构型，给定一种颜色，您就知道这个类可能具有哪些属性、链接、方法和交互，从而得到一些彩色的构建块。本书包含6章展示61个领域所需的相关组件，本书讲解详细，实例丰富，展示了61个组件、283个类、46个接口、671个属性、1139个方法和65个交互序列图。

本书可作为UML建模人员、Java工程师、技术人员的参考用书。

本书版权登记号：图字：01-2008-3908

图书在版编目（CIP）数据

彩色UML建模 /（美）科德（Coad, P.）等著；王海鹏等译. —北京：机械工业出版社，2008.12

书名原文：Java Modeling in Color with UML: Enterprise Components and Process

ISBN 978-7-111-25481-2

Ⅰ.彩… Ⅱ.① 科… ② 王… Ⅲ.① 面向对象语言，UML-程序设计 ② JAVA语言-程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2008）第170903号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）

责任编辑：陈佳媛

中国电影出版社印刷厂印刷·新华书店北京发行所发行

2008年12月第1版第1次印刷

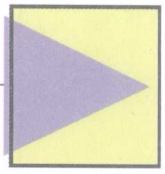
205mm×255mm · 12.75印张

标准书号：ISBN 978-7-111-25481-2

定价：55.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294



庖丁解牛

在很多行业里，不同岗位的人员，身上穿的工作服装颜色是不一样的，如酒店的接待员穿红色，行李员穿黄色，而厨师就是穿白色。您下次去银行、工厂、餐馆、商场、移动、电信等地方，不妨留心观察一下员工们的服饰。

Peter Coad认为各个领域之间也存在一些领域中性（Domain-Neutral）的基本结构，即四种基本的架构型（Archetype）：MomentInterval、Role、PartyPlaceThing、Description。Peter Coad用四种颜色来标记它们，并归纳出了它们之间交互的规律。以此为利器，可以如庖丁解牛般解剖各类系统的业务模型。

Peter Coad在1991年写的两本小册子《面向对象的分析》、《面向对象的设计》是最早引入国内的面向对象分析设计方法学书籍，成为很多人（包括我）的对象思想启蒙读物。Peter Coad的书一般都开门见山，只写自己的实作心得，很少东拉西扯引经据典，所以他的书一向比较薄，像本书，只有6章，但很值。

UMLChina首席专家 潘加宇

译 者 序

人们的学习都是从模仿开始的。学习书法时，一种重要途径就是临贴。学习围棋时，一种重要途径就是打谱。学习面向对象分析和建模时，对应的途径在哪里？

开发者希望看到真实世界中企业级开发的例子，而不只是ATM机的UML图。如果您是一名建筑设计师，您有机会看到上海金茂大厦的设计图纸吗？我们会看到最后的产品，但对产品创造的过程一无所知。模型并不重要，重要的是得到模型的过程。我们希望听到更多的设计大师的自战解说。开源让我们有机会看到优秀的代码，而这本书让我们有机会看到优秀的面向对象分析模型以及建模的过程。

建模能帮您做到什么？或者说，您为什么要建模？对我来说，建模是为了让软件开发更简单一些。虽然让复杂的事情变得简单并不容易，但这本书却做到了。通过4种彩色的架构型，让我们能够迅速地对复杂的业务领域建立起简单的模型。

系统分析师需要“体察千行百业之要义”，这本书对企业组件进行了全景式描述，为我们提供了学习的典范。

Booch在他的《面向对象分析与设计》一书中说，所有成功的软件项目都有两个显著的特点：一是很强的架构愿景，二是迭代增量式的开发。彩色UML和FDD做到了这两点。

一本好书会改变您对事物的理解，从而改变您的行为实践。对我来说，这本书就是这样的。每次我看到其他人给出的领域分析模型，都会用彩色UML的方法去印证，结果总是能得到更好的模型。

这本书是一本值得反复临写的贴，是一本值得反复打的谱。

参加本书翻译工作的人员除封面署名外还有：贾立群、王海燕、李国安、周建鸣、范俊、张海洲、谢伟奇、林冀、钱立强、甘莉萍。在这本书的翻译过程中，我学到了很多，因此郑重地向大家推荐它。如果这本书对于您改进软件开发实践有所帮助，我将十分高兴。

王海鹏

戊子年夏日于上海

前　言

架构型、颜色和组件将从此永久地改变您的Java建模方式。我们是从开发者的角度来构建Java模型的。在日复一日的现场指导中，我们提出并尝试新的想法，帮助那些开发者在建模中胜出。有一些想法被抛弃了；有一些想法提供了一定的帮助；还有一些想法，根据我们客户的说法，是全垒打。在这本书中，我们将介绍一些我们的全垒打。

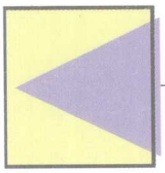
第1章探讨了颜色的重要性，引入了颜色编码。许多项目团队应用颜色编码在全球范围内取得了成功。这一章也介绍了领域无关的组件，这是一个模板，您会在后续的章节中一次又一次地看到它的应用。

第2章到第5章提供了立即可用的Java模型。这几章展示了61个领域相关的组件，每个组件都是一次有趣的实例教学。利用现成的组件，通过插件对它们进行扩展，通过添加自己的组件对它们进行扩展，或者利用它们作为一种外部意见（比较、对比您自己正在做的工作）。

第6章介绍了一个过程，这个过程将Java建模与经常提交的、实实在在的有效结果结合起来。

我们希望您享受这些新思想！

Peter Coad（pc@oi.com）
Eric Lefebvre（lefee@groupe-progestic.com）
Jeff De Luca（jdl@nebulun.com）



致　谢

如果没有许多人的大力帮助，这本书不会到您手中。我们特别要感谢：

- 我们的客户，是他们让我们首先在实践中提出这些思想。特别要感谢在Durham，Columbus(Ohio)、Hartford、Montreal、Oslo和新加坡的客户。
- John Kern，感谢他的许多深刻见解、建议和不知疲倦的工作，改进了本书的技术内容。
- Mark Mayfield和David North，他们与Peter协作完成了这本书的前一本书，《Object Modles: Strategies, Patterns, and Applications》。
- David Taylor，他阅读了早期的手稿，与我们分享了他的想法，提出了一些问题，否则这些问题就得不到解答。
- Phil Bradly，他温和地提醒我们将领域无关的组件扩展为一个总体模板，并且在我们考虑的词之中主张采用“架构型（archetype）”。
- Barbara Hanscome是《Software Development》杂志的主编，她鼓励我们对颜色理论追求更深入的理解。
- OCLC的Roger Thompson，他向我们指出Root-Bernstein关于形象思维的文章。
- Sun Microsystems公司的John Gage，他鼓励Jeff开始编写我们现在所谓的“特征驱动开发”的内容。
- Stephen Palmer，他在形成特征驱动开发时与我们合作。
- M. A. Rajashima，他向我们介绍了“进入条件、任务、验证、退出条件”的方式。
- United Overseas Bank的Lim Bak Wee，他向我们介绍了彩色进度报告的方法。
- Prentice Hall的Jeffery Pepper，他提供了鼓励和见解。
- Precision Graphics的工作团队，他们发挥自己的才能，将我们的手稿变成了拿在您们手上的书。

我们也要感谢那些在这本书形成过程中提供了反馈意见的人，他们是：David Anderson、Hakan Axelsson、Tom Considine、Rikard Dahlman、Terry Gliedt、Low Heng Sin、Mike Morrison、Ajay Kumar Rana、R. Mark Sharp、Paul Szego和Julia Tan。

如果没有一些重要的彩色UML建模软件，也不可能有这本书。因此我们要感谢我们的朋友，Together/J团队成员，尤其要感谢：Dietrich Charisius（首席架构师）、Alexander Aptus、Sergey Dmitriev、Valentin Kipyatkov、Igor Bazarny、Andrel Ivanov、Kate Gorentchuk、Fyodor Isakov、Eugene Belyaev、Konstantin Savvin、Robert Palomo、Maxim Livov、Frank Baker、Lee Youngblood、Robert Neher、Frank Sterkmann和Hanspeter Siegrist。

目　录

彩色UML建模

第1章

架构型、彩色与领域无关的组件

粉红色：这是我最喜欢的蜡笔。

Aerosmith

黑色与白色传递了基本的信息。彩色传递了更多信息，抓住了您的眼球。

正如从黑白照片向彩色照片的转换是如此意义深远，从黑白建模向彩色建模的转换也是令人敬畏的。

欢迎来到彩色建模的世界，这里有架构型（archetype）、领域无关的组件和61个领域相关的组件。

在这一章中，您将学习并应用4种架构型、彩色和领域无关的组件。在第2章至第5章，您将学习并应用大量不同的领域相关的组件。在第6章，您将发现特征驱动开发，一个将所有这些纳入最佳实践的开发过程。

Java在这里扮演了什么角色呢？许多建模的方式是受到Java启发的。您会发现聚合而不是继承。您也会看到聪明地使用了接口插入点——目地是为了增加可扩展性。

我们写这本书是让它成为《Java Design》的前端姊妹篇。那本书介绍了聚合设计、多线程设计和通知设计的具体策略。

在整个这本书中，我们使用了统一建模语言（UML）表示法。我们在本书中使用的类图表示法和惯例如图1-1所示。我们使用的序列图表示法和惯例如图1-2所示。我们建议您现在就快速看一下这些图，以后再经常回顾一下。

1.1 架构型

现在，让我们将注意力转向本章的第一个主要议题：架构型。

下面是我们想传达的概念：

针对为数不多的类分组的一种形式或一个模板。它规定了这个分组中类的常见属性、链接、方法、插入点和交互。

下面哪一个术语更适合这个概念？

构造型（Stereotype）

1. 一个不变的模型，就像从一个模子里出来的
2. 一个文本标签，标注一个UML图的元素
3. 一个更广泛的类的分组

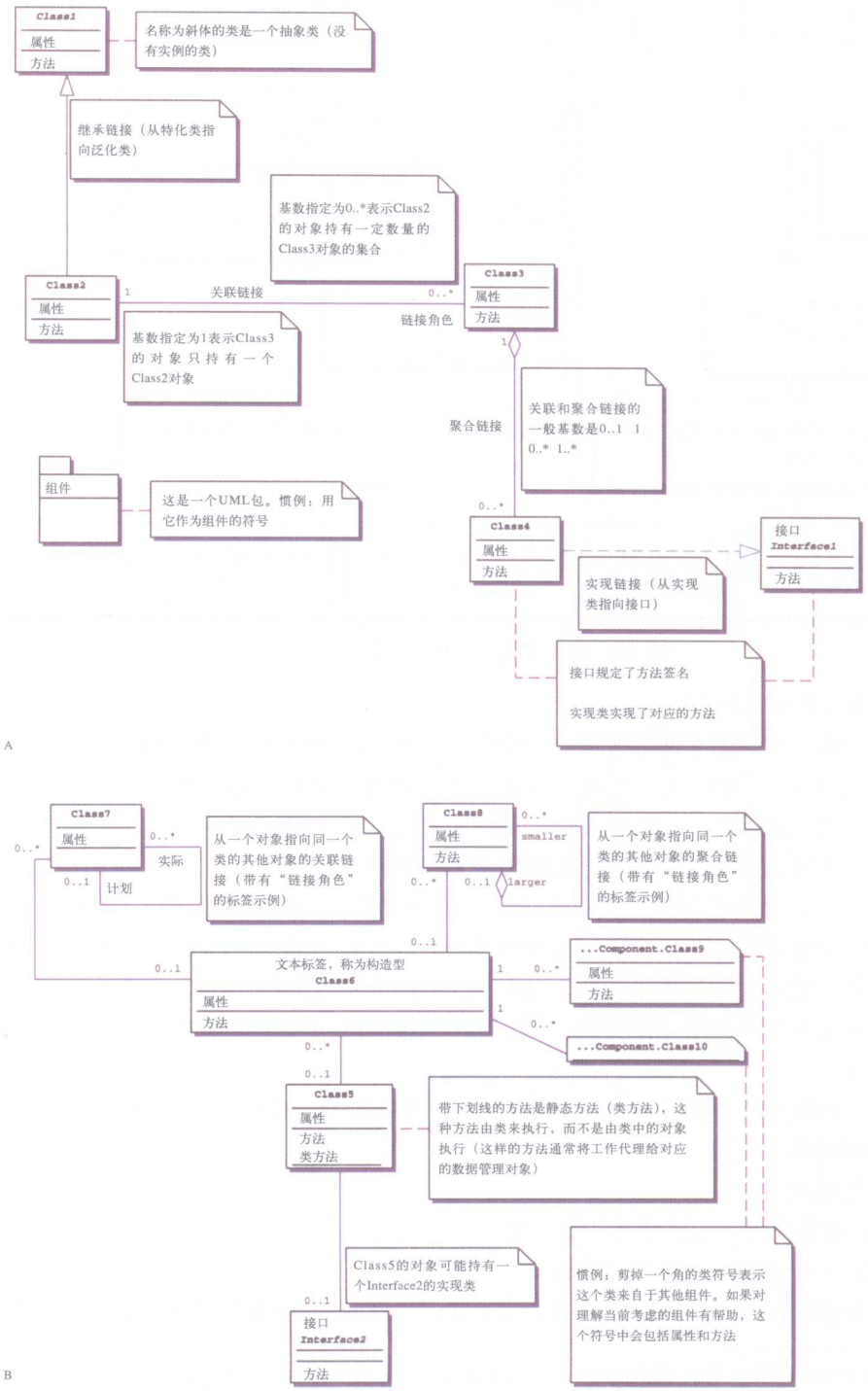


图1-1　类图表示法和惯例

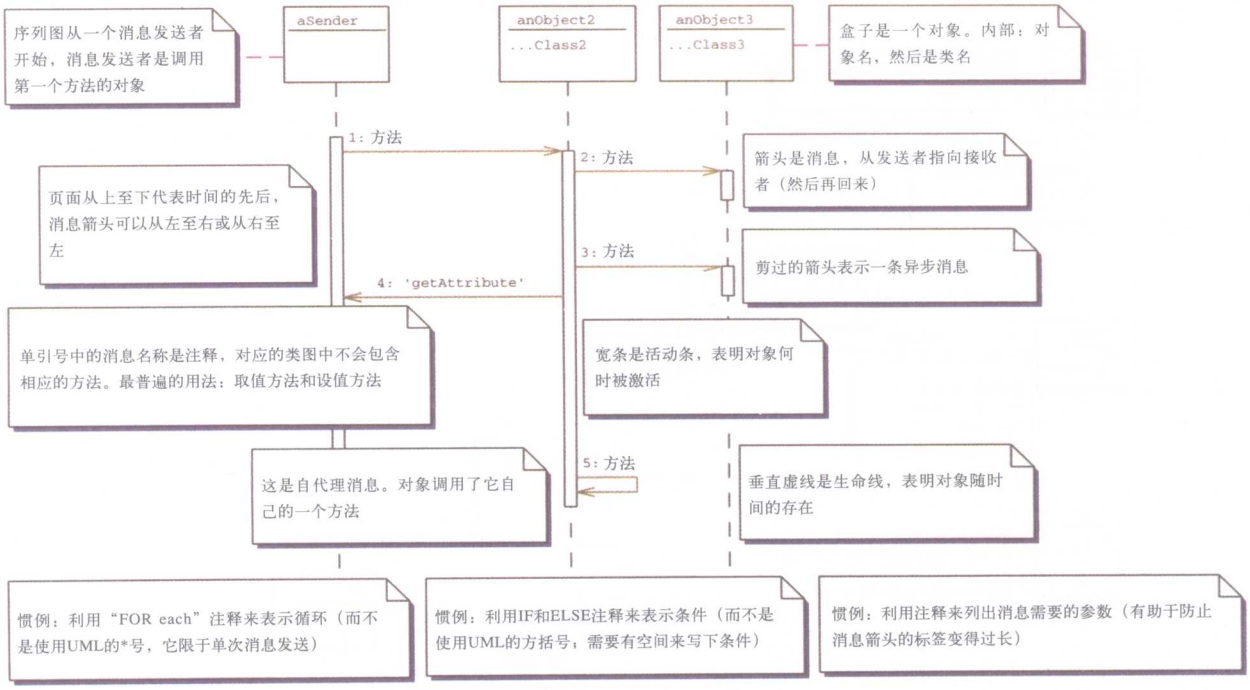


图1-2　序列图表示法和惯例

架构型（Archetype）

一种形式，所有的东西都或多或少地遵守。[摘自Webster75和Hayakawa68]

这就是事实。“架构型”更好地表达了这个概念。架构型是一种形式，属于同一种类型的类都或多或少地遵守——包括属性、链接、方法、插入点和交互。[⊖]

但是，哪些架构型是创建更好的模型过程中最有用的呢？

我们曾在几十个业务和工程领域开发了几百个模型。在这个过程中，我们不断地寻找一些方式，以期能够抽象出一个领域无关的组件，即一个小的架构型模型，它可以反复地在我们的研讨会和现场指导工作中应用。为什么？因为我们觉得我们可以在更少的时间里教授更多内容，在更少的时间里完成更多工作。这对我们的客户更好，我们自己也更有兴趣，这是双赢的结局。

随着时间的推移，我们发现了4种相互联系的架构型，它们形成了领域无关的组件：

1. 时刻时段架构型
2. 角色架构型
3. “分类目录条目似的描述”架构型
4. “参与方–地点–物品”架构型

我们要感谢Peter Coad和Mark Mayfield奠定了这4种架构型的早期基础工作，它们首先在

⊖ 架构型是“或多或少”遵守的形式。“或多或少”这一点是基本的。与它形成对比的是，构造型是不可变的。另外与它形成对比的是，继承和接口规定的名称是必须遵守的，不是或多或少遵守。——译者注

[Coad92]中得到描述，后来在[Coad95-97]中与David North一起进行了扩展。

1.1.1 时刻时段架构型

最重要的架构型就是一个时刻或一段时间。它代表了出于商业和法律上的原因，我们需要处理并追踪的某件事情，这件事情是在某个时刻或某一段时间内发生的。为了简便起见，我们称之为“时刻时段”，以此提醒我们在寻找的是问题域中具有重要意义的时刻或时段。

一次销售是在一个时刻完成的——这次销售的日期和时间。

一次租赁发生在一个时间段，从登记入住到归还。一次预订发生在一个时间段，从预订的时刻开始，直到被使用、被取消或超期。

如果您需要追踪销售过程本身的时间，也许是为了评估工作效率，那么一次销售甚至可以是一个时间段。

重要的是您意识到它属于这二者之一，而不是它具体属于这二者之中的哪一个。所以我们将它作为一种架构型，时刻时段。

1.利用架构型来确定类和更多东西

在任何领域中，人们都可以寻找时刻时段，并开始创建模型。在物料-资源管理中，我们可以从请求RFQ转到PO，再到交付，再到开票。在制造管理中，我们可以从一个计划的过程及其步骤转到实际的过程及其步骤。

所以架构型帮助指导建模的一种方式是，确定模型中需要包含的类。

但架构型不仅是类的分组。它们同时也是职责（属性、链接、方法、插入点和交互）的分组，属于这个架构型的类通常具有这样的职责。

2. 为架构型加上标签

我们需要的就是一个文本标签，这样我们就能在创建一个类时说明应用的是哪一种架构型。在UML中，这种文本标签被称为构造型，是表示法中的一种扩展机制（图1-3）。

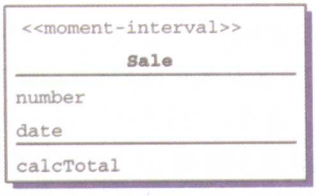


图1-3 利用UML文本标签来说明时刻时段架构型

问题是，像<<moment-interval>>这样的文本标签在过于普通和简单的文本标签中隐藏了非常重要的含义。在一系列的图里，这个小小的标签迷失在一片噪声中，因为它看起来就像所有其他的标签一样。这太糟糕了，架构型的表示非常重要，大大超过它所得到的关注。如果我们能够突出表示这一层增加的意思，那就太好了。所以它应该：

- 首先就让您注意到图中的那一部分；
- 帮助您发现时刻时段架构型随时间发展的情况；

- 指导您将其他类链接到正在处理的时刻时段架构型上；
- 安静地让您考虑哪些东西链接到时刻时段架构型上，它如何通过与其他类协作来完成工作。

用彩色来表示架构型是独辟蹊径，完全达到了上述要求，而且提供了更多好处。

3. 实现架构型

架构型在源代码中看起来是怎样的呢？

架构型描述了一个模型，属于这个架构型的类或多或少都遵守这个模型。这里重要的一点是“或多或少”。

能够将架构型实现为一个超类，然后从它继承吗？不能！原因是：架构型的本质是每个属于它的类只是或多或少的遵守它。对于架构型的概念，对于它们要表达的意思来说，继承太严格了。

另一种实现架构型的方式是利用包含特定关键词的注释，建模工具可以识别并有效地利用这种方式。在Java中，我们利用javadoc风格的注释来实现这一点，利用了特定的关键词。例如，下面的注释就包含了嵌入的关键词（在@符号后面）：

```
/** @archetype moment-interval*/
public class Sale {
 public BigDecimal calcTotal(){...
 }
 private int number;
 private Date date;
}
```

这样很好地完成了这项任务。

1.1.2 角色架构型

第二重要的架构型是角色。 角色是一种参与的方式，它由人、地点或物品来承担。

另一种说法是，角色是一种参与方式，它由参与方（人或组织机构）、地点或物品来承担。我们更喜欢这一种说法，因为很多时候在我们面对的问题域中，一个人或一个组织机构适合承担同一个角色（如拥有者）。通常，时刻时段会有一些组成部分，称为时刻时段明细。可以将它们看成是更小的时刻时段，我们需要它们来完成工作。

所以我们既对角色扮演者（参与方、地点或物品）建模，也对角色（参与方、地点或物品所戴的“帽子”）建模。角色扮演者记录了一些核心的属性和行为，不论它可能戴上几顶帽子。对于人来说，这通常包括法定的名称和出生日期等属性。它也包含一些方法，在它扮演的一组角色上强制实现一些业务规则。例如，person对象有一个“授权为”（authorized for）方法，与每个角色进行交互，然后我们对一组角色应用业务规则，确定某个人是否得到授权执行某个动作（图1-4）。

参与方、人和组织机构扮演角色是很常见的。有时候，您会发现地点和物品也扮演角色（例如，一个产品扮演两个角色：“在销售过程中的产品”和“在使用的产品”）。

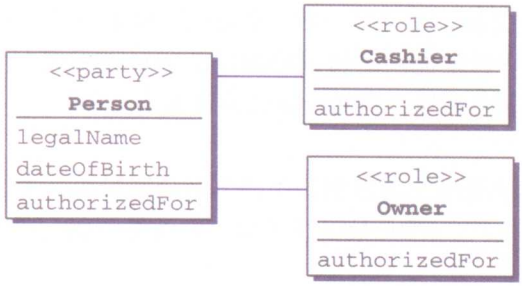


图1-4 参与方及其角色

1.1.3 描述架构型

第三个架构型是描述。更具体地说，它是“分类目录条目似的”描述。它是一组反复应用的值。它也为所有对应到某个描述的东西提供行为。

例如，您的红色小货车是一辆机动车，它有自己的序列号（称为车辆标识编号）、购买日期、颜色和里程表读数。对应的分类目录条目似的描述是车辆描述，它包括了制造商、型号、制造日期和可选颜色。一些业务相关的方法也适合放在这里，如“有多少辆这样的货车处于良好的工作状态？”

1.1.4 “参与方–地点–物品”架构型

参与方（意味着人或组织机构）、地点或物品是扮演不同角色的人或物。人可能既是雇员又是顾客。地点可能既是零售处也是批发处。物品可能在制造过程中扮演一个角色，在购买过程中扮演另一个角色。

1.2 彩色

在1997年9月，我们开始用4种彩色的即时贴来创建模型：粉红色、黄色、绿色和蓝色，每种颜色代表一种架构型。团队中对建模不熟悉的开发者和领域专家在这个过程中曾几次提出问题“如果没有彩色，你们怎么可能创建有效的模型呢？”这引起了我们的注意！所以我们在实践中发展了这种技术，公布了我们最初的发现[Coad97a]，并在OOPSLA’97会议上展示了这种方法[Coad97b]。

就像通常的情形一样，实践先于理论。看到这些想法在实践中效果很好，我们开始研究彩色，研究它为什么在构建更好的模型时产生了如此深刻的影响。

1.2.1 为什么使用彩色

我们为什么要在组件模型中使用彩色？彩色为我们提供了一种方式，加入了附加的信息层。这种使用彩色的方式巧妙地增加了我们要表达的信息量。

更重要的是，人们可以利用彩色为模型添加新的内容层次。这些层次可以从很远处看到，

这样就在您开始了解细节之前，表现出一种“全景式”模型内容。我们把这种效果称为“空间分层”，它意味着模型本身就能同时提供概述视图和详细视图，不需要切换视觉上下文，跳到另一种表现形式上。彩色使这种空间分层成为可能。

> 分层和分离技术从视觉上将数据的不同方面分出层次，这是减少噪声、丰富表现内容的最强有力的方式之一。[然后他描述了如何做到这一点：利用形状、明暗、大小，特别是利用颜色。]
>
> Edward R. Tufte[Tufte90]

所以，我们可以利用彩色来丰富模型的内容。实际上，我们可以利用颜色来实现4个目的：

> 在信息设计方面彩色的基本用法：用于标记（颜色表示名词）、用于测量（颜色表示数量）、用于表示或模拟真实性（颜色是一种展现方式）、用于活跃气氛或装饰（颜色表示美）。
>
> Edward R. Tufte[Tufte90]

对于建模的意义在于，我们可以利用颜色做到以下几点：

- 标识附加的信息层次（例如，具有类似特征的类组成的层次）。
- 表明时间进度（例如，可以利用不同的光影变化来表示进度）。
- 表示一个模型中关键的信息分类。
- 为模型增加视觉冲击力。

增加视觉冲击力是很重要的。建模就其本质来说是与视觉相关的活动。具有很好的空间想象力对创建模型和理解模型有很大的好处。

> 空间知识可以有多种科学用途，它可以是一种有用的工具、一种思考的辅助手段、一种记录信息的方式、一种阐明问题的方式，或者就是解决问题的方式。
>
> Howard Gardner[Gardner83]

> 这种将结果理想化的能力，透过真实世界的纷繁现象看到应该是什么的能力，正是天才的一种标志。
>
> Robert Scott Root-Bernstein[Root-Bernstein85]

加入彩色更好地保证了模型创建者和模型解读者的空间想象力。