



中国科学技术大学软件学院
SCHOOL OF SOFTWARE ENGINEERING OF USTC

面向对象建模与设计

ffh



目录

- ◆ 什么是面向对象
- ◆ 什么是面向对象开发
- ◆ 面向对象相关概念
- ◆ UML介绍
- ◆ UML世界的构成
- ◆ UML图及特征





什么是面向对象

◆ 什么是面向对象

“面向对象”是一种认识客观世界的世界观，这种世界观将客观世界看成是有许多不同种类的对象构成，每个对象有自己的内部状态和运动规律，不同对象之间的相互联系、相互作用就构成了完整的客观世界。

“面向对象”是从结构组织的角度去模拟客观世界的一种方法，这种方法的基本着眼点是构成客观世界的那些成分---对象。

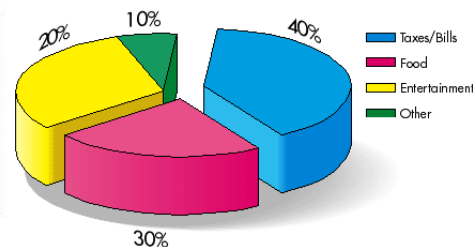
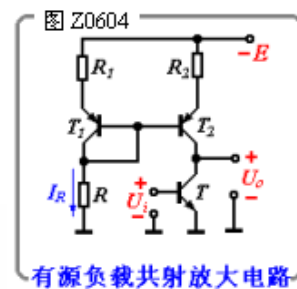
用“面向对象”的观点去认识客观世界，用“面向对象”的方法去模拟客观世界，这就构成了“面向对象”的完整含义。





什么是OO开发

- ◆ 开发指的是软件的生命周期
- ◆ 系统开发主要是建模
 - 模型是现实的简化
 - 建模是为了让我们更好地理解将要开发的系统。





什么是OO开发

建模要实现的目的：

模型有助于按原样或根据需要使系统可视化。

通过模型可以详细说明系统的结构或行为。

模型可以提供一个指导我们构建系统的模板。

模型可以记录已经做出的决策。

可以把软件开发理解为一系列模型的开发，可分为：

规格说明模型，分析模型，设计模型，代码模型，编码实现

◆ 三种模型

类模型：是描述系统内部对象及其关系的静态结构。

状态模型：描述了对象随时间而发生变化的部分。

交互模型：描述系统中的对象如何协作以完成更为宽泛的任务。





面向对象基本概念

◆ 抽象

抽象是从众多的事物中抽取出共同的、本质性的特征，而舍弃其非本质的特征。以便更充分地注意与当前目标有关的方面。

例如苹果、香蕉、生梨、葡萄、桃子等，它们共同的特性就是水果。

人在同一时间里，一般只能集中于7项左右的信息，而不受信息的内容、大小等因素的影响。





面向对象基本概念

◆ 封装（信息隐蔽）

封装就是将抽象得到的数据和行为（或功能）相结合，形成一个有机的整体（对象）。

隐藏对象的内部实现细节，仅对外公开接口。

只能通过接口访问对象。用户只需知道对象的功能，而不需了解对象内部的细节。

信息隐藏的过程

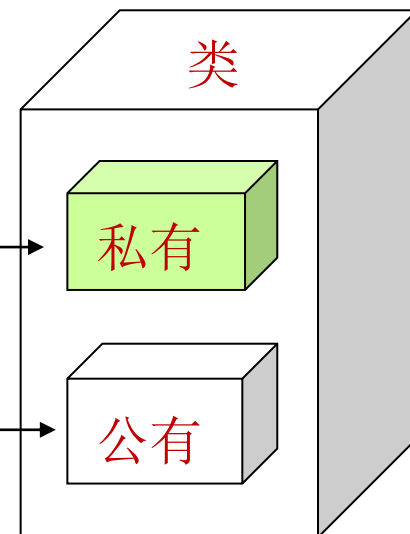
有选择的数据隐藏

防止意外的数据破坏

更易于隔离和修复错误

在类的外部不能访问

在类的外部可以访问





面向对象基本概念

◆ 对象

对象是现实世界中个体或事物的抽象表示，是其属性和相关操作的封装。属性表示对象的性质，属性值规定了对象所有可能的状态。对象的操作是指该对象可以展现的外部服务。

◆ 类和实例

类是某些对象的共同特性的表示，它描述了这些对象内部是如何构造的。相同类的对象在它们的操作和它们的信息结构两个方面都有相同的定义。

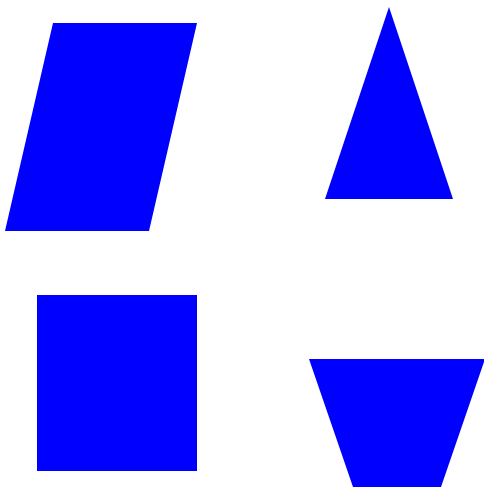
在面向对象系统中，每个对象都属于一个类。属于某个特定类的对象称为该类的实例。因此，常常把对象和实例当作同义词。实例是从某类创建的一个对象。





面向对象基本概念

多边形对象



抽象为



多边形类

属性

顶点
边的颜色
填充颜色

方法

绘制
擦除
移动

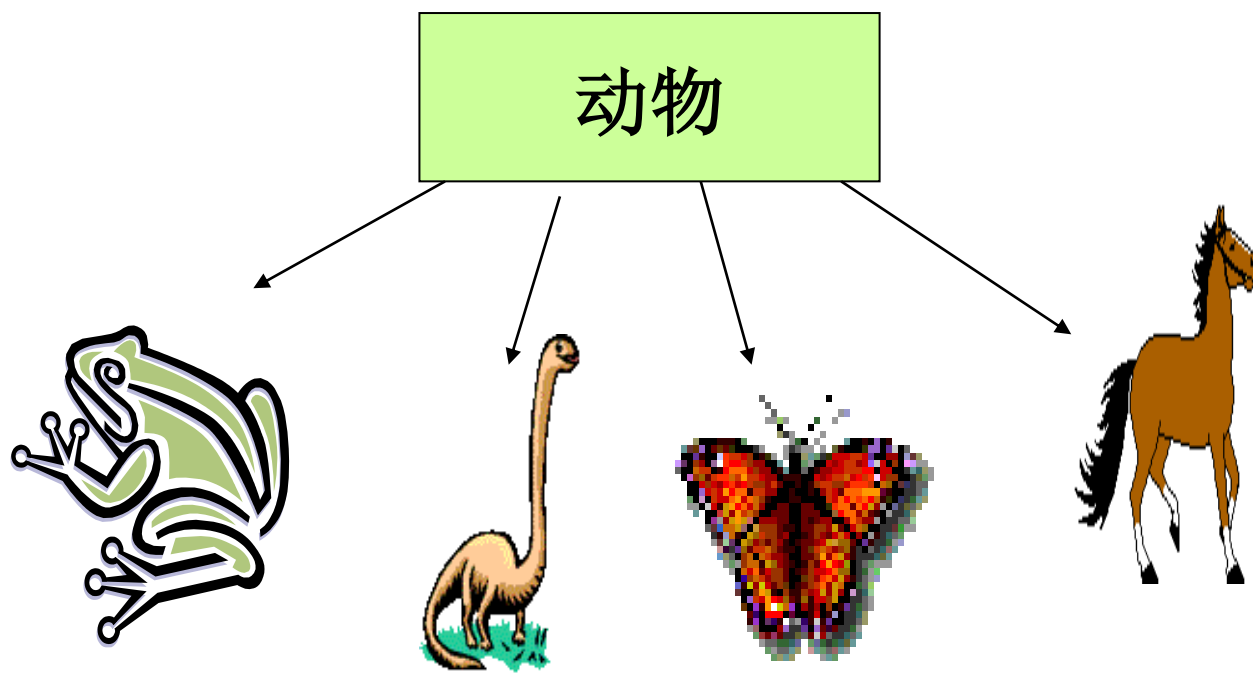




面向对象基本概念

◆ 继承

如果类B继承类A，那么类A中描述的操作和信息结构将成为类B的一部分。





面向对象基本概念

◆ 继承

借助继承，可以表示类之间的类似性，并且在其他类能继承的一个类中描述这些相似性。因此，就能够复用公共的描述。

继承常常被提倡为软件工业界中关于复用的一个核心思想。继承还有利于软件维护。

通过抽取和共享公共特性就能够通用化一些类，并且把它们放在继承层次的更高位置。同样，如果希望增加新类，可以寻找这样一个类，它已经提供了适用于该新类的某些操作和信息结构。然后，让新类继承这个类，只需增加该新类所独有的那些内容。然后，使这个类专用化。



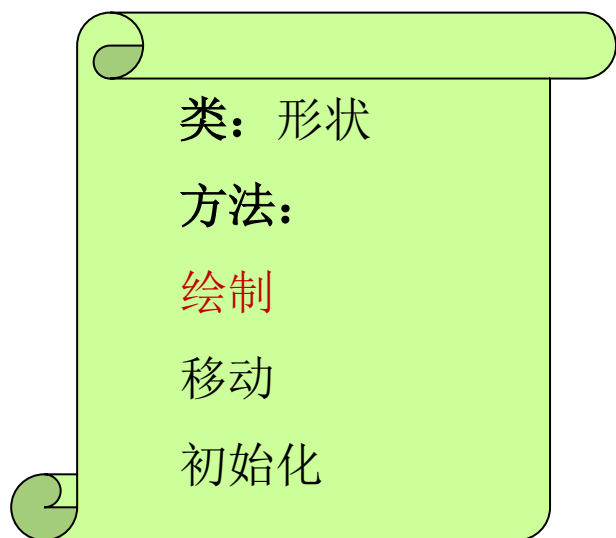


面向对象基本概念

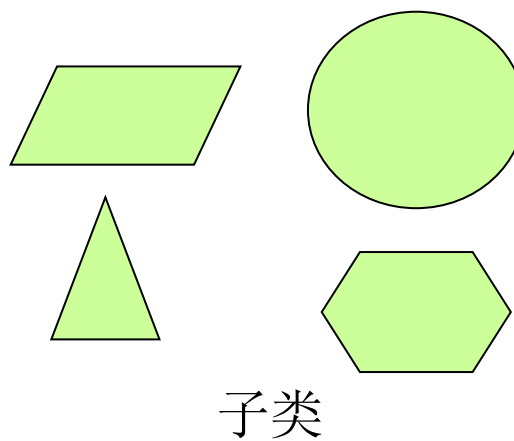
◆ 多态

指同一个实体同时具有多种形式(相同的操作会产生不同的动作)。

在面向对象语言中，接口的多种不同的实现方式即为多态（同样的函数在不同的类上有不同的行为）。



对各个子类实现的方法 将产生不同的结果





面向对象基本概念

◆ 接口

接口是一种约束形式，其中只包括成员定义，不包含成员实现的内容。

接口的主要目的是为不相关的类提供通用的处理服务。

当类实现一个接口，它就许诺实现在那个接口中执行所有的方法。

◆ 消息

对象之间的交互

包含三个方面的内容：

消息的接收者；接收对象应采用的方法；方法所需要的参数





目录

- ◆ 什么是面向对象
- ◆ 什么是面向对象开发
- ◆ 面向对象相关概念
- ◆ UML介绍
- ◆ UML世界的构成
- ◆ UML图及特征





UML介绍

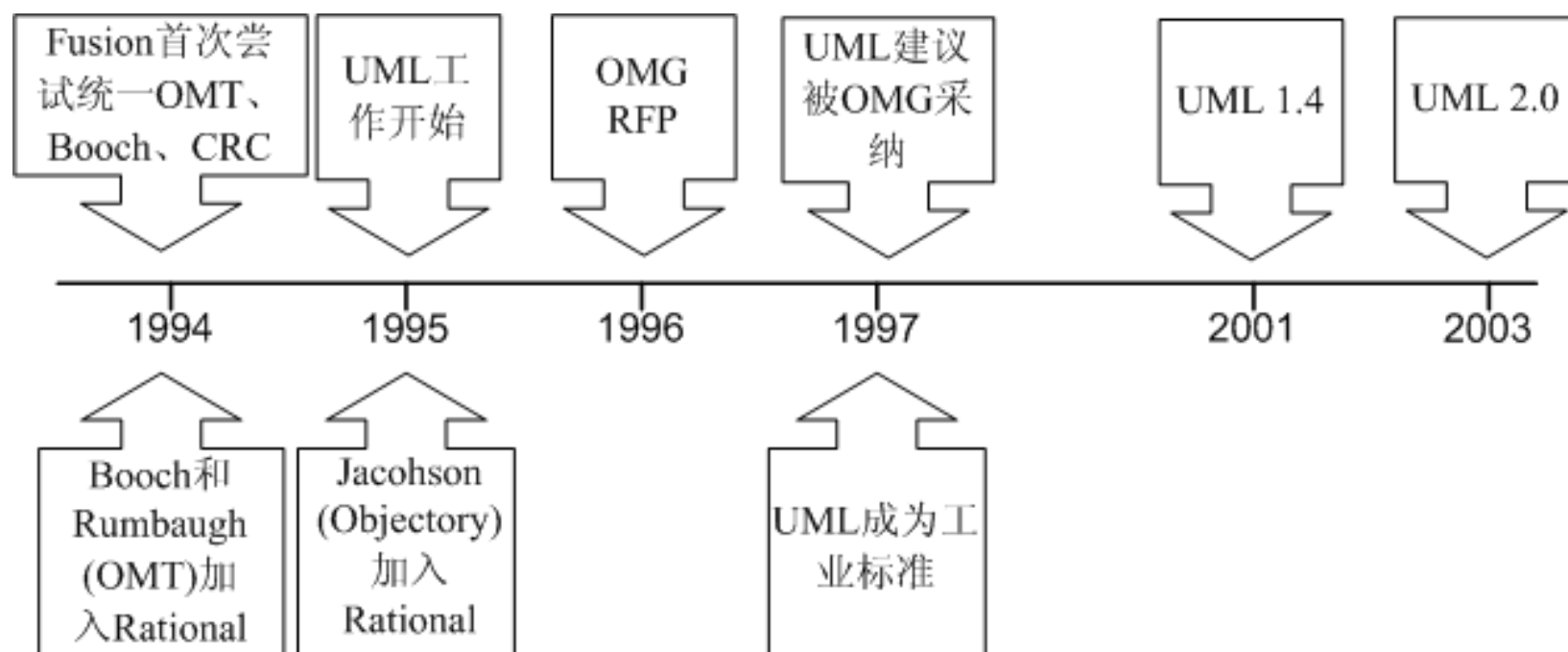
- ◆ 建模语言是建模中的一个非常关键的因素。
- ◆ **Unified Modeling Language**（统一建模语言）
 - 是一种通用的模拟语言，可用于确定、展示和记录软件系统。
 - UML中的图形标记非常适用于面向对象的软件设计。
- ◆ **UML的设计目标：**
 - 运用面向对象概念来构造系统模型
 - 建立起从概念模型直至可执行体之间明显的对应关系
 - 着眼于那些有重大影响的问题
 - 创建一种对人和机器都适用的建模语言





UML介绍

UML发展历程





标准建模语言UML

◆ UML发展历程

- UML由OMG与1997年11月批准为标准建模语言。
- UML是一种建模语言而不是一种方法，UML本身是独立于过程的。

◆ 用模型描述系统

UML为人们提供了从不同的角度去观察和展示系统的各种特征的一种标准表达方式。在UML中，从任何一个角度对系统所作的抽象都可能需要用几种模型图来描述，而这些来自不同角度的模型图最终组成了系统的完整模型。





标准建模语言UML

一般而言，我们可以从以下几种常用的视角来描述一个系统：

- 系统的使用实例：从系统外部的操作者的角度描述系统的功能。
- 系统的逻辑结构：描述系统内部的静态结构和动态行为，即从内部描述如何设计实现系统功能。
- 系统的构成：描述系统由哪些程序构件所组成。
- 系统的并发性：描述系统的并发性，强调并发系统中存在的各种通信和同步问题。
- 系统的配置：描述系统的软件和各种硬件设备之间的配置关系。





UML的语义与语法

- ◆ UML的定义包括UML语义和UML表示法两个部分。
 - UML语义：描述基于UML的精确元模型定义。

元模型为UML的所有元素在语法和语义上提供了简单、一致、通用的定义性说明，使开发者能在语义上取得一致，消除了因人而异的最佳表达方法所造成的影响。此外UML还支持对元模型的扩展定义。
 - UML表示法：定义UML符号的表示法，为开发者或开发工具使用这些图形符号和文本语法为系统建模提供了标准。

这些图形符号和文字所表达的是应用级的模型，在语义上它是UML元模型的实例





目录

- ◆ 什么是面向对象
- ◆ 什么是面向对象开发
- ◆ 面向对象相关概念
- ◆ UML介绍
- ◆ UML世界的构成
- ◆ UML图及特征



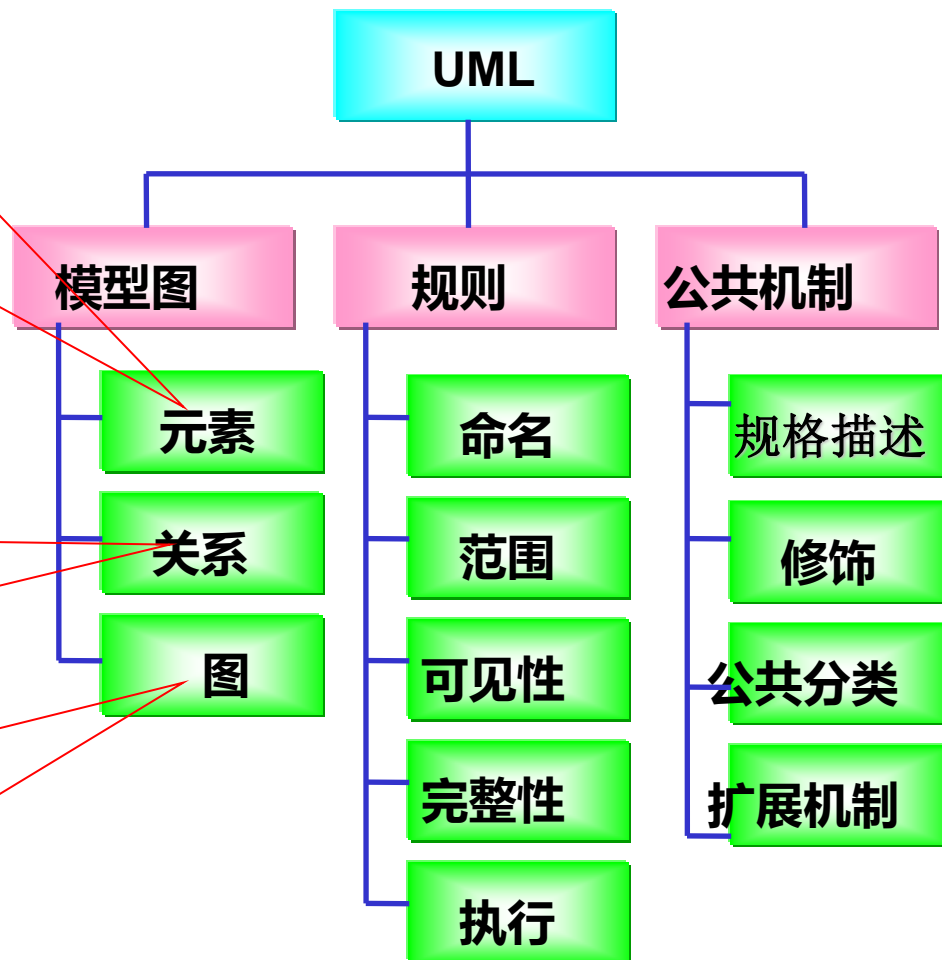


UML组成

- 1. 结构元素(类,接口,协作,用例,对象,构件,节点等)
- 2. 行为元素(交互,状态机)
- 3. 分组元素(包)
- 4. 注解元素

- 1. 关联
- 2. 依赖
- 3. 泛化
- 4. 实现

- 1. 静态模型(类图,构件图,部署图)
- 2. 动态模型(对象图,用例图,顺序图,协作图,状态图,活动图)





UML模型图的构成

◆ UML模型图的构成

– 事物(Things) (元素)

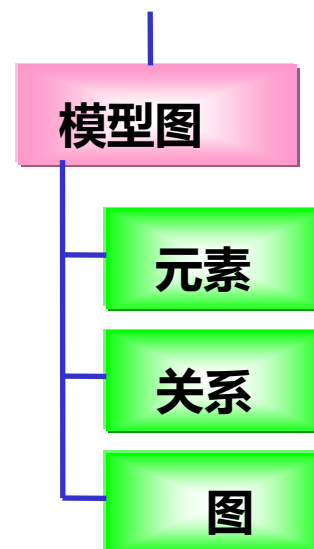
UML模型中最基本的构成元素，是具有代表性的成分的抽象

– 关系(Relationships)

关系把事物紧密联系在一起

– 图(Diagrams)

图是事物和关系的可视化表示





UML模型图的构成

◆ UML事物

UML包含4种事物：

- 构件事物：

UML中的名词，它是模型的基本物理元素。

- 行为事物：

UML中的动词，它是模型中的动态部分，是一种跨越时间、空间的行为。

- 分组事物：

UML中的容器，用来组织模型，使模型更加的结构化。

- 注释事物：

UML中的解释部分，和代码中的注释语句一样，是用来描述模型的。





UML模型图的构成

— 构件事物： UML模型的静态部分，描述概念或物理元素

它包括以下几种：

- **类 (class) 和对象 (object)**：具有相同属性相同操作 相同关系相同语义的对象的描述
- **接口 (interface)**：描述元素的外部可见行为，即服务集合的定义说明
- **协作 (collaboration)**：描述了一组事物间的相互作用的集合
- **用例 (use case)**：代表一个系统或系统的一部分行为，是一组动作序列的集合
- **构件 (component)**：系统中物理存在，可替换的部件
- **节点 (node)**：运行时存在的物理元素

另外，参与者、信号应用、文档库、页表等都是上述基本事物的变体





UML模型图的构成

◆ 构件事物

– 类 (class) 和对象 (object)

类是对一组具有相同属性、相同操作、相同关系和相同语义的对象的抽象

UML中类是用一个矩形表示的，它包含三个区域，最上面是类名、中间是类的属性、最下面是类的方法

对象则是类的一个实例



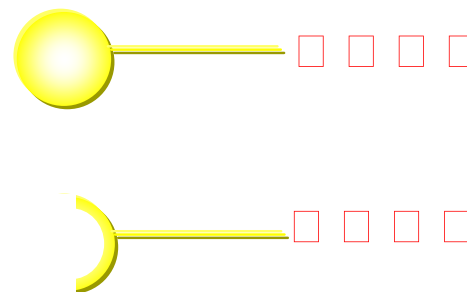


UML模型图的构成

◆ 构件事物

– 接口 (interface)

接口是描述某个类或构件的一个服务操作集





UML模型图的构成

◆ 构件事物

– 主动类 (active class)

主动类实际上是一种特殊的类。引用它的原因，实际上是在开发中需要有一些类能够起到启动控制活动的作用

主动类是指其对象至少拥有一个进程或线程，能够启动控制活动的类





UML模型图的构成

◆ 构件事物

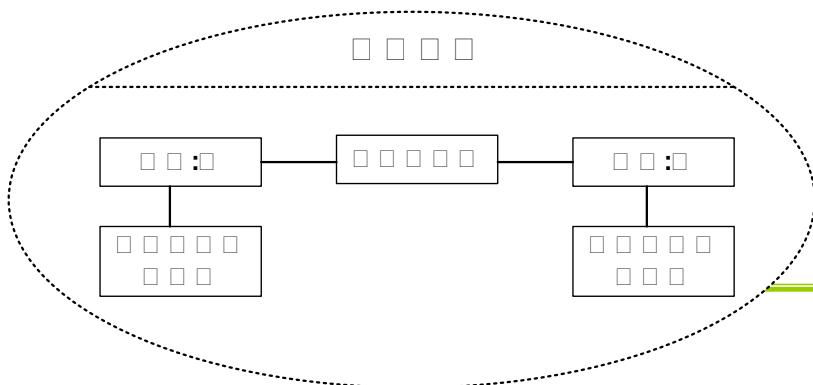
- 用例与协作 (use case & collaboration)

用例是著名的大师Ivar Jacobson首先提出的，现已经成为了面向对象软件开发中一个需求分析的最常用工具；

用例实例是在系统中执行的一系列动作，这些动作将生成特定执行者可见的价值结果。一个用例定义一组用例实例。

协作定义了一个交互，它是由一组共同工作以提供某协作行为的角色和其他元素构成的一个群体。

对于某个用例的实现就可以表示为一个协作





UML模型图的构成

◆ 构件事物

– 构件 (component)

在实际的软件系统中，有许多要比“类”更大的实体，例如一个COM组件、一个DLL文件、一个JavaBeans、一个执行文件等等。为了更好地对在UML模型中对它们进行表示，就引入了构件（也译为组件）

构件是系统设计的一个模块化部分，它隐藏了内部的实现，对外提供了一组外部接口。在系统中满足相同接口的组件可以自由地替换



1.0 □ □ □ □



2.0 □ □ □ □





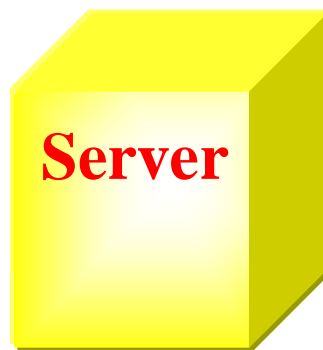
UML模型图的构成

◆ 构件事物

– 节点 (node)

为了能够有效地对部署的结构进行建模，UML引入了节点这一概念，它可以用来描述实际的PC机、打印机、服务器等软件运行的基础硬件

节点是运行时存在的物理元素，它表示了一种可计算的资源，通常至少有存储空间和处理能力





UML模型图的构成

◆ 行为事物

UML模型图的动态部分，描述跨越空间和时间的行为

– 交互 (interaction) :

是在特定语境中，共同完成某个任务的一组构件事物之间的信息集合，涉及消息、动作序列、链接

交互的表示法很简单，就是一条有向直线，并在上面标有操作名

– 状态机 (state machine) :

是一个事物或交互在生命周期内响应事件所经历的状态序列

在UML模型中将状态画为一个圆角矩形，并在矩形内写出状态名称及其子状态

等待





UML模型图的构成

◆ 分组事物

UML模型图的组织部分，描述事物的组织结构

对于一个中大型的软件系统而言，通常会包含大量的类，因此也就存在大量的构件事物、行为事物，为了能够更加有效地对其进行整合，生成或简或繁、或宏观或微观的模型，就需要对其进行分组。在UML中，提供了“包（Package）”来完成这一目标

包：把元素组织成组的机制



数据访问



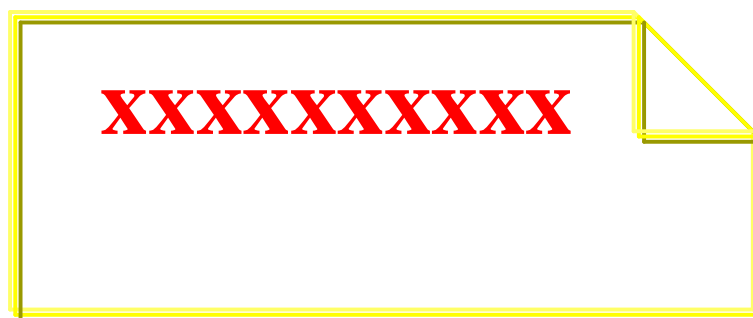


UML模型图的构成

◆ 注释事物：

UML模型的解释部分，用来对模型中的元素进行说明，解释
构件事物是模型的主要构造块，行为事物则是补充了模型中的动态
部分，分组事物而是用来更好地组织模型，似乎已经很完整了。
而注释事物则是用来锦上添花的，它是用来在UML模型上添加适
当的解释部分

注解：对元素进行约束或解释的简单符号





UML模型图的构成

◆ UML关系

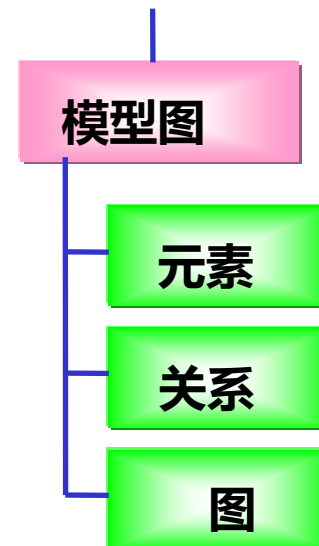
关系主要包括关联、泛化、依赖、实现

关联：关联(association)是一种结构关系，它指明一个事物的对象与另一个事物的对象间的联系

泛化：泛化(generalization)是一种特殊/一般的关系。也可以看作是常说的继承关系

依赖：依赖(dependency)是两个事物之间的语义关系，其中一个事物(独立事物)发生变化，会影响到另一个事物(依赖事物)的语义

实现：实现(realization)是类元之间的语义关系，其中的一个类元指定了由另一个类元保证执行的契约





UML模型图的构成

◆ 关系--关联关系

关联（Association）表示两个类之间存在某种语义上的联系。关联关系提供了通信的路径，它是所有关系中最通用、语义最弱的。

在UML中，使用一条实线来表示关联关系





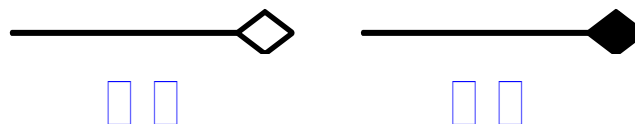
UML模型图的构成

◆ 关系--关联关系

在关联关系中，有两种比较特殊的关系：聚合(集)和组合(成)

聚合关系：聚合（Aggregation）是一种特殊形式的关联。聚合表示类之间的关系是整体与部分的关系

如果发现“部分”类的存在，是完全依赖于“整体”类的，那么就应该使用“组合”关系来描述



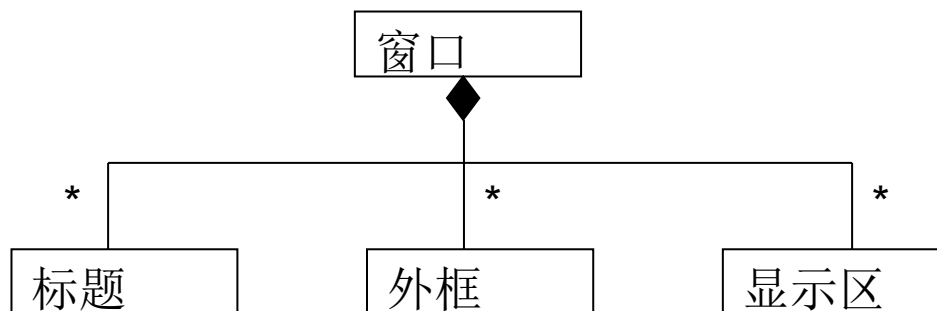
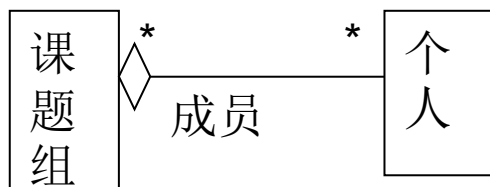


UML模型图的构成

◆ 关系--关联关系

组合是聚合的变种，加入了一些重要的语义。也就是说，在一个组合关系中一个对象一次就只是一个组合的一部分，“整体”负责“部分”的创建和破坏，当“整体”被破坏时，“部分”也随之消失

聚合就像汽车和车胎，汽车坏了胎还可以用。组合就像公司和下属部门，公司倒闭了部门也就不存在了！





UML模型图的构成

◆ 关系--泛化

泛化关系描述了一般事物与该事物中的特殊种类之间的关系，也就是父类与子类之间的关系。



Java	UML
<pre>public abstract class Employee { } public class Professor extends Employee { }</pre>	<pre>classDiagram Employee < -- Professor</pre>

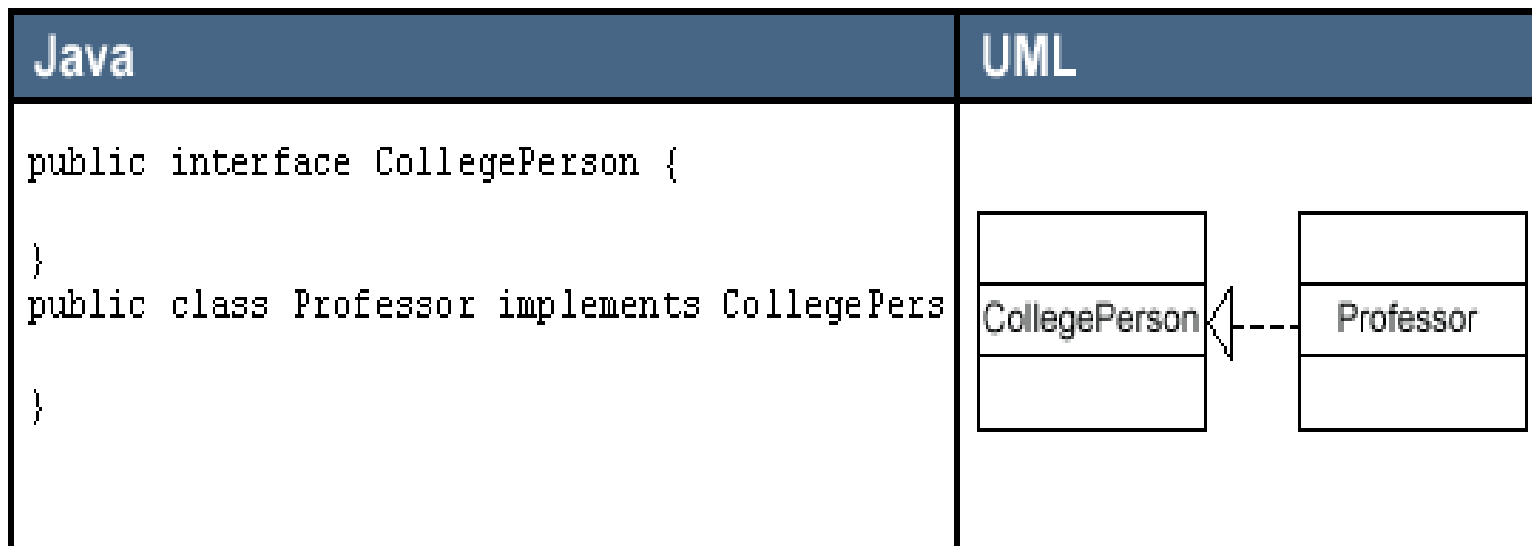
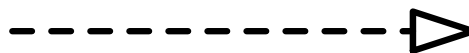




UML模型图的构成

◆ 关系--实现

实现关系是用来规定接口和实现接口的类或组件之间的关系。接口是操作的集合，这些操作用于规定类或组件的服务。

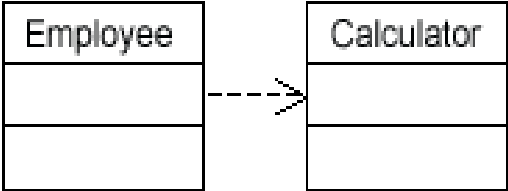




UML模型图的构成

◆ 关系--依赖

有两个元素X、Y，如果修改元素X的定义可能会引起对另一个元素Y的定义的修改，则称元素Y依赖（Dependency）于元素X。

Java	UML
<pre>public class Employee { public void calcSalary(CalculatorStrategy { ... } }</pre>	 <pre>classDiagram Employee ..> Calculator</pre>



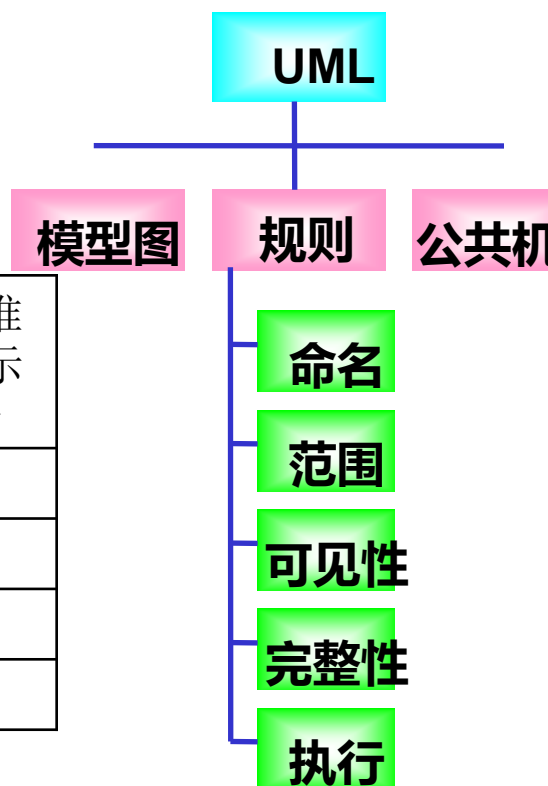


UML世界的构成

◆ UML规则

- 命名：也就是为事物、关系和图起名字。和任何语言一样，名字都是一个标识符
- 范围：与类的作用域相似.
- 可见性：

可见性	规则	标准表示法
public	任一元素，若能访问包容器，就可以访问它	+
protected	只有包容器中的元素或包容器的后代才能够看到它	#
private	只有包容器中的元素才能够看得到它	-
package	只有声明在同一个包中的元素才能够看到该元素	~

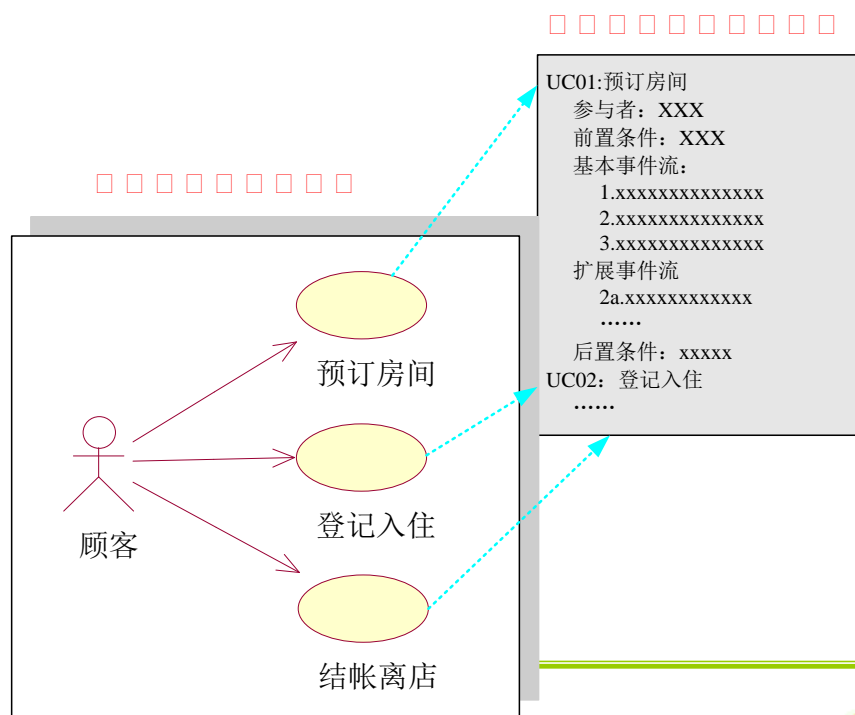




UML世界的构成

◆ UML公共规则--规格描述

在图形表示法的每个部分后面都有一个规格描述（也称为详述），它用来对构造块的语法和语义进行文字叙述。这种构思，也就使可视化视图和文字视图的分离：



公共机制

规格描述

修饰

公共分类

扩展机制





UML世界的构成

◆ UML公共规则--UML修饰与公共分类（通用划分）

在为了更好的表示这些细节，UML中还提供了一些修饰符号，例如不同可视性的符号、用斜体字表示抽象类

UML通用划分：

- 1) 类与对象的划分：类是一种抽象，对象是一个具体的实例
- 2) 接口与实现的分离：接口是一种声明、是一个契约，也是服务的入口；实现则是负责实施接口提供的契约

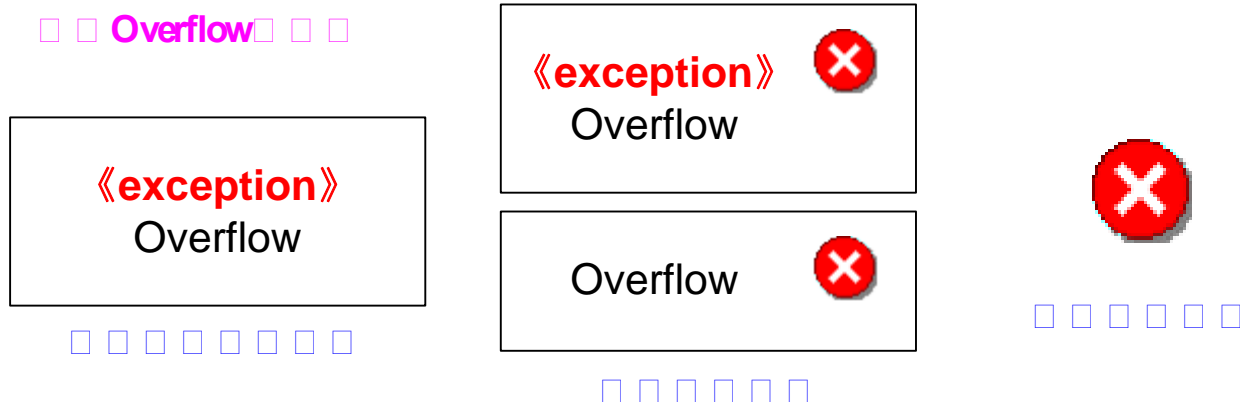




UML世界的构成

◆ UML扩展机制

构造型：在实际的建模过程中，可能会需要定义一些特定于某个领域或某个系统的构造块



标记值：则是用来为事物添加新特性的。标记值的表示方法是用形如“{标记信息}”的字符串

约束：是用来增加新的语义或改变已存在规则的一种机制（自由文本和OCL两种表示法）。约束的表示法和标记值法类似，都是使用花括号括起来的串来表示，不过它是不能够放在元素中的，而是放在相关的元素附近





UML世界的构成

◆ UML定义的图

图名	功能	备注
类图	描述类、类的特性以及类之间的关系	UML 1原有
对象图	描述一个时间点上系统中各个对象的一个快照	UML 1非正式图
复合结构图	描述类的运行时刻的分解	UML 2.0新增
构件图	描述构件的结构与连接	UML 1原有
部署图	描述在各个节点上的部署	UML 1原有
包图	描述编译时的层次结构	UML中非正式图
用例图	描述用户与系统如何交互	UML 1原有
活动图	描述过程行为与并行行为	UML 1原有
状态机图	描述事件如何改变对象生命周期	UML 1原有
顺序图	描述对象之间的交互，重点在强调顺序	UML 1原有
通信图	描述对象之间的交互，重点在于连接	UML 1中的协作图
定时图	描述对象之间的交互，重点在于定时	UML 2.0 新增
交互概观图	是一种顺序图与活动图的混合	UML 2.0新增





UML世界的构成

◆ UML视图和图

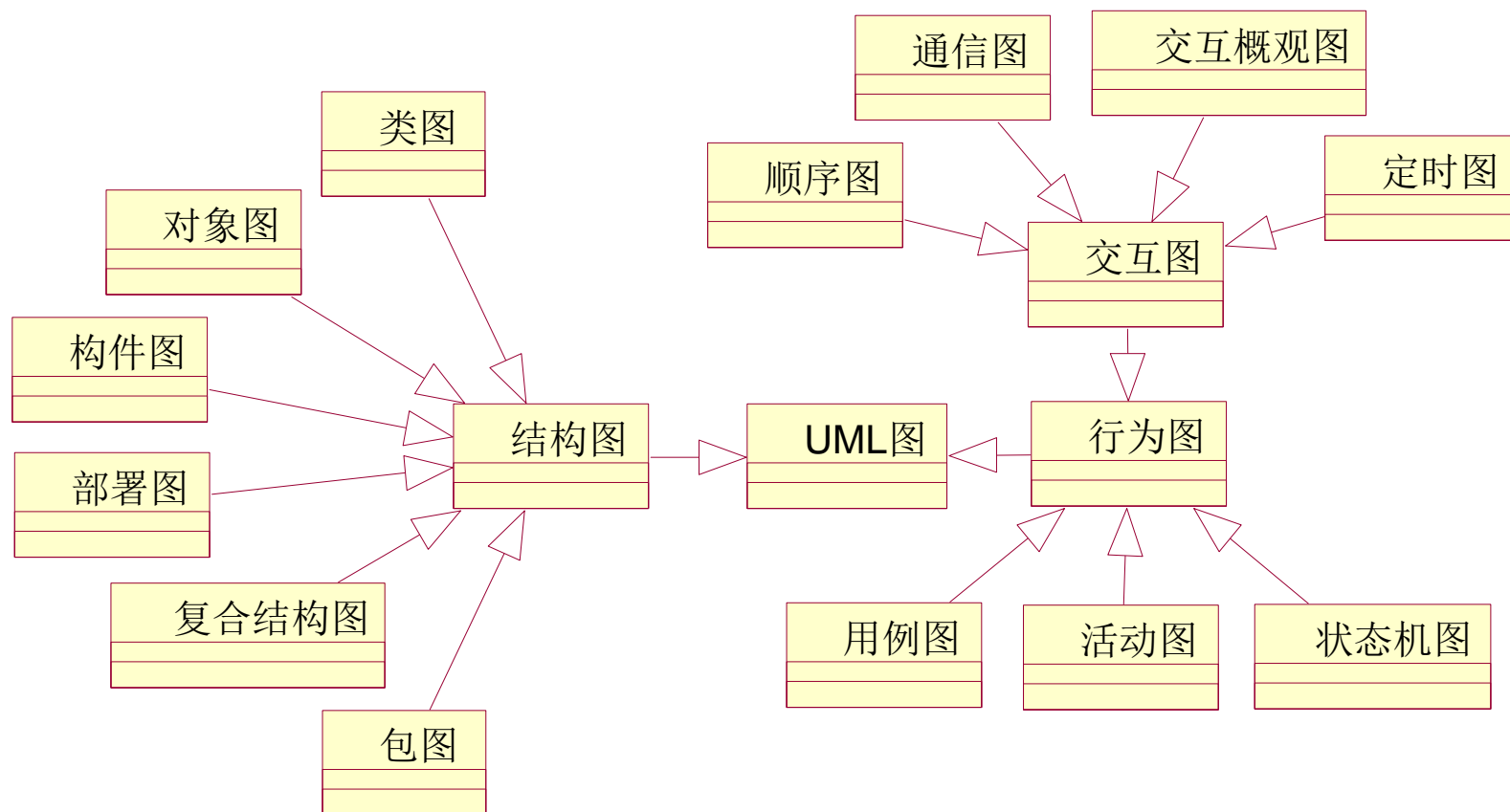
主要领域	视图	图
结构	静态视图	类图
	设计视图	复合结构图、协作图、构件图
	用例视图	用例图
动态	状态视图	状态机图
	活动视图	活动图
	交互视图	顺序图、通信图
物理	部署视图	部署图
模型管理	模型管理视图	包图
	特性描述	包图





UML世界的构成

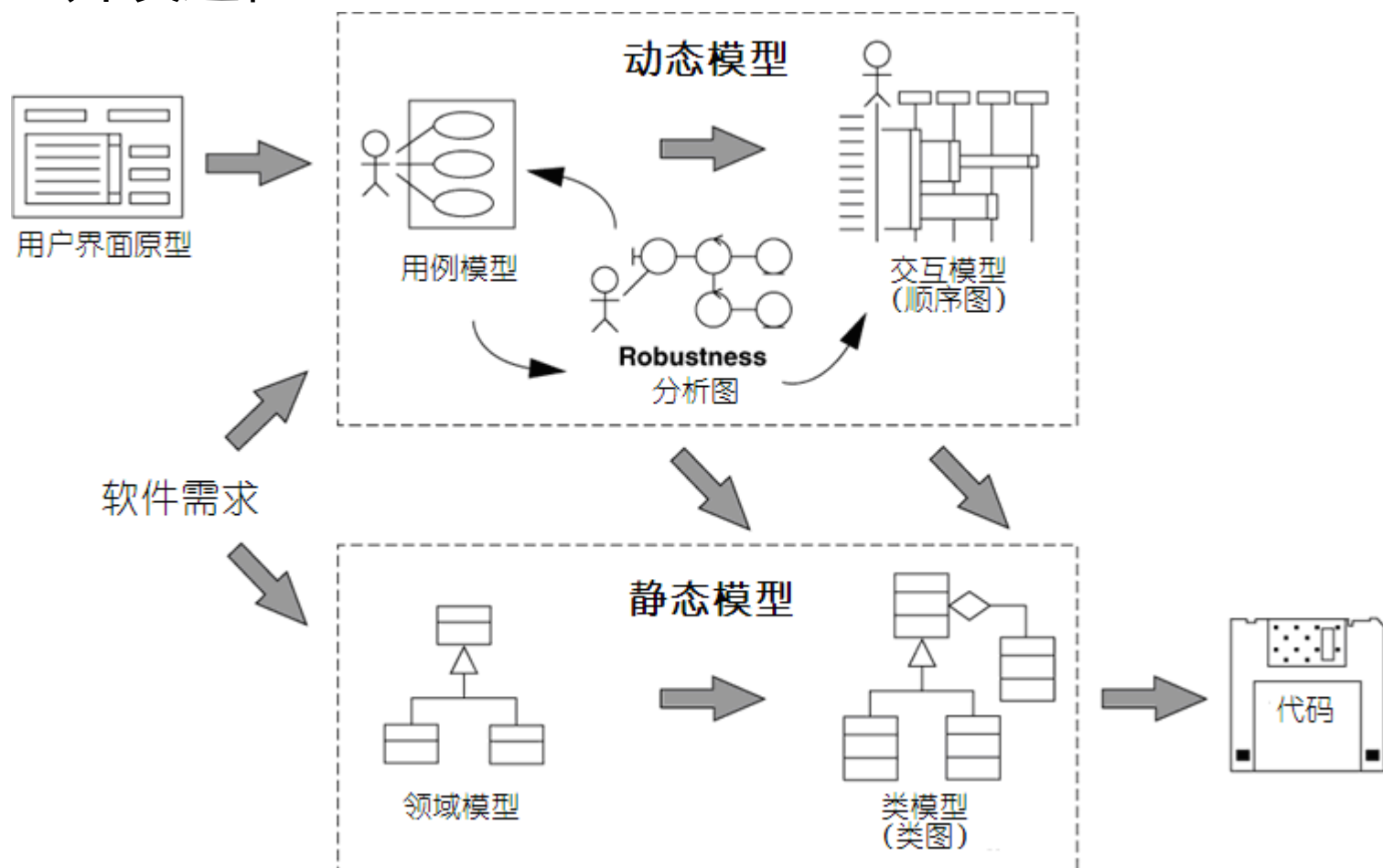
◆ UML图形分类





UML世界的构成

◆ 开发过程





目录

- ◆ 什么是面向对象
- ◆ 什么是面向对象开发
- ◆ 面向对象相关概念
- ◆ UML介绍
- ◆ UML世界的构成
- ◆ UML图及特征

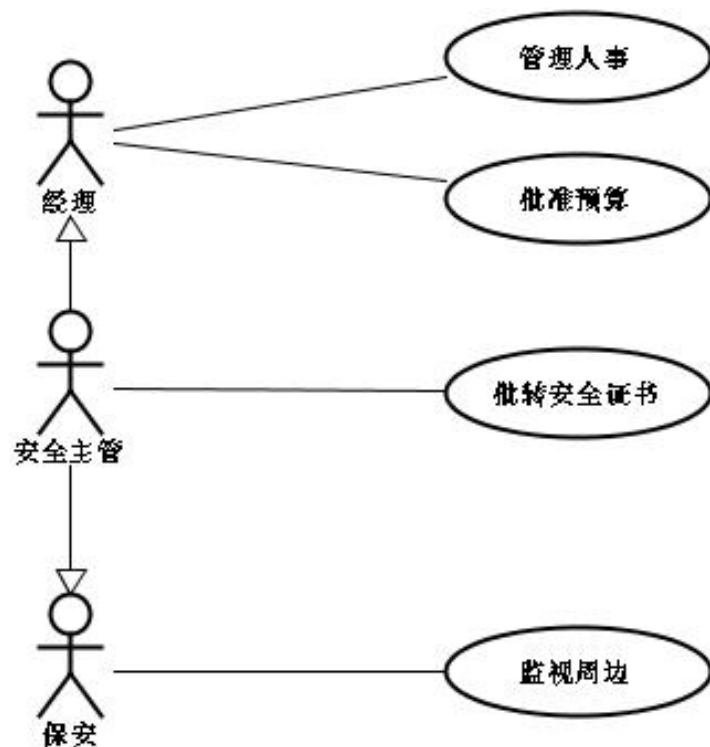




UML图及特征

◆ 用例图(Use Case Diagram)

用例图是从用户角度描述系统功能，
是用户所能观察到的系统功能的模
型图，用例是系统中的一个功能单
元



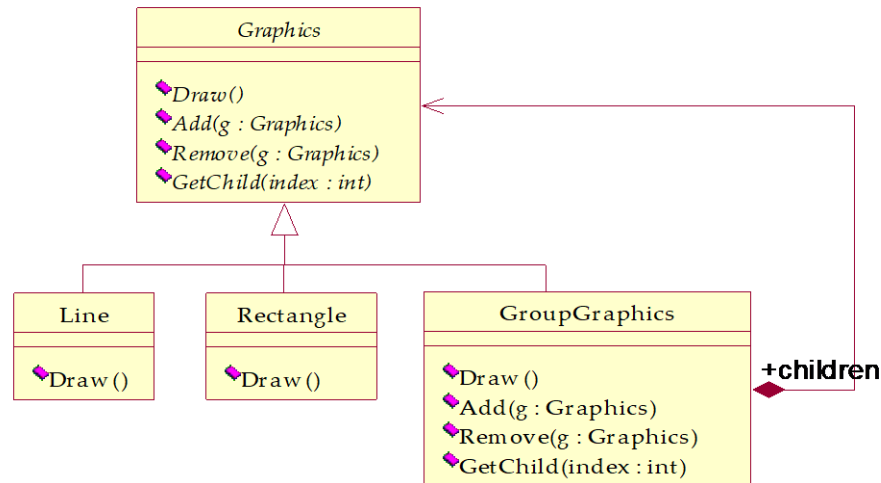


UML图及特征

◆ 类图(Class Diagram)

类图描述系统中类的静态结构。不仅定义系统中的类，表示类之间的联系如关联、依赖、聚合等，也包括类的内部结构(类的属性和操作)

类图是以类为中心来组织的，类图中的其他元素或属于某个类或与类相关联

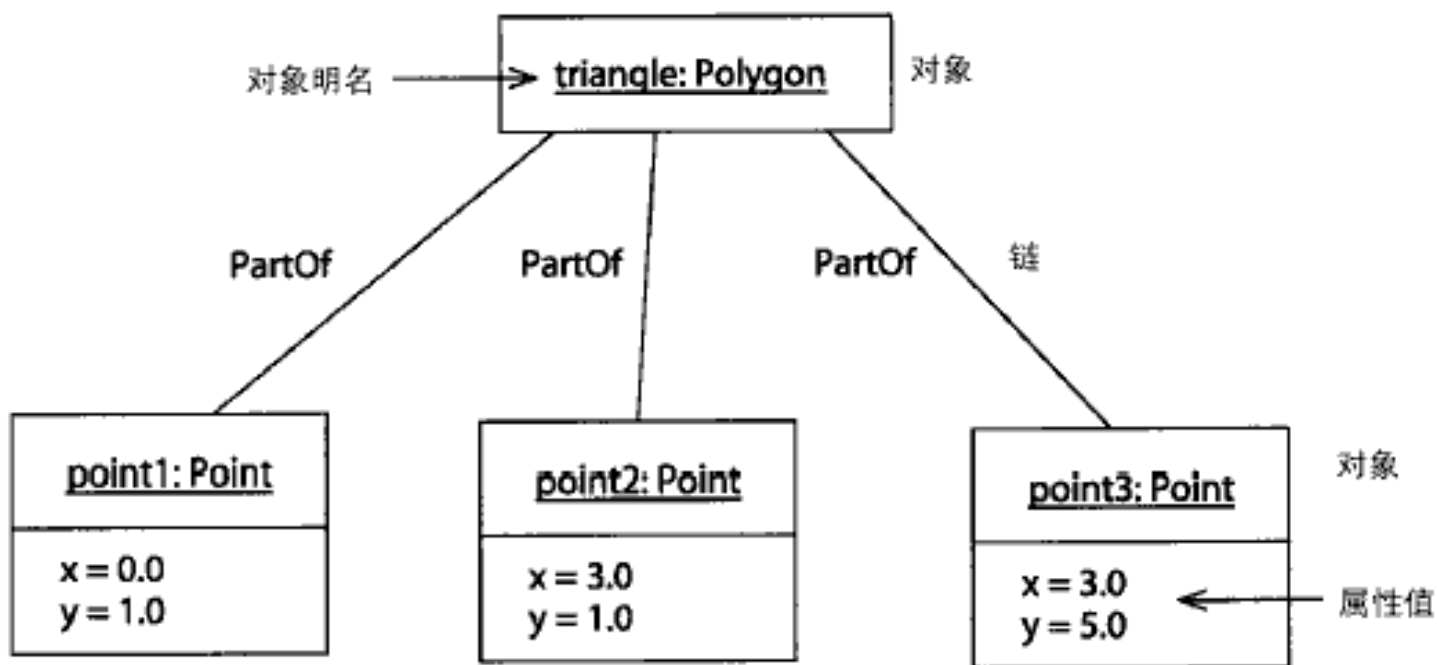




UML图及特征

◆ 对象图(Object Diagram)

对象图是类图的实例，几乎使用与类图完全相同的标识。他们的不同点在于对象图显示类的多个对象实例，而不是实际的类



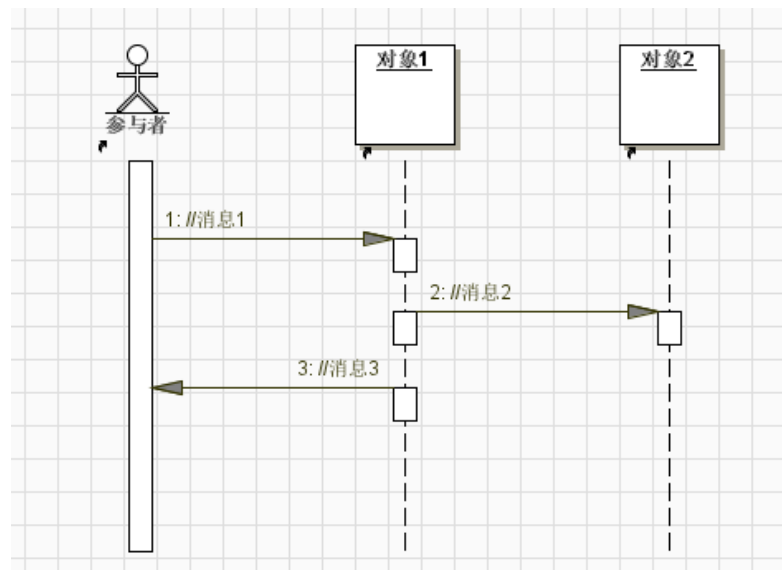


UML图及特征

◆ 顺序图(Sequence Diagram)

顺序图显示对象之间的动态合作关系，它强调对象之间消息发送的顺序，同时显示对象之间的交互

顺序图的一个用途是用来表示用例中的行为顺序。当执行一个用例行为时，顺序图中的每条消息对应了一个类操作或引起状态转换的触发事件



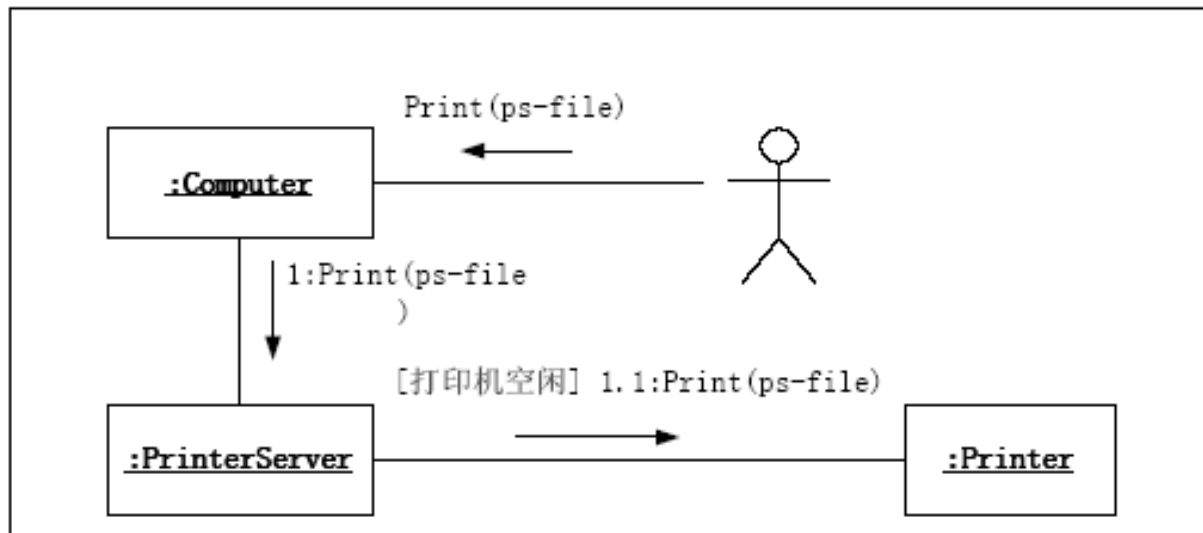


UML图及特征

◆ 协作图(Collaboration Diagram)

协作图描述对象间的协作关系，协作图跟顺序图 相似，显示对象间的动态合作关系。除显示信息交换外，协作图还显示对象以及它们之间的关系。

协作图的一个用途是表示一个类操作的实现

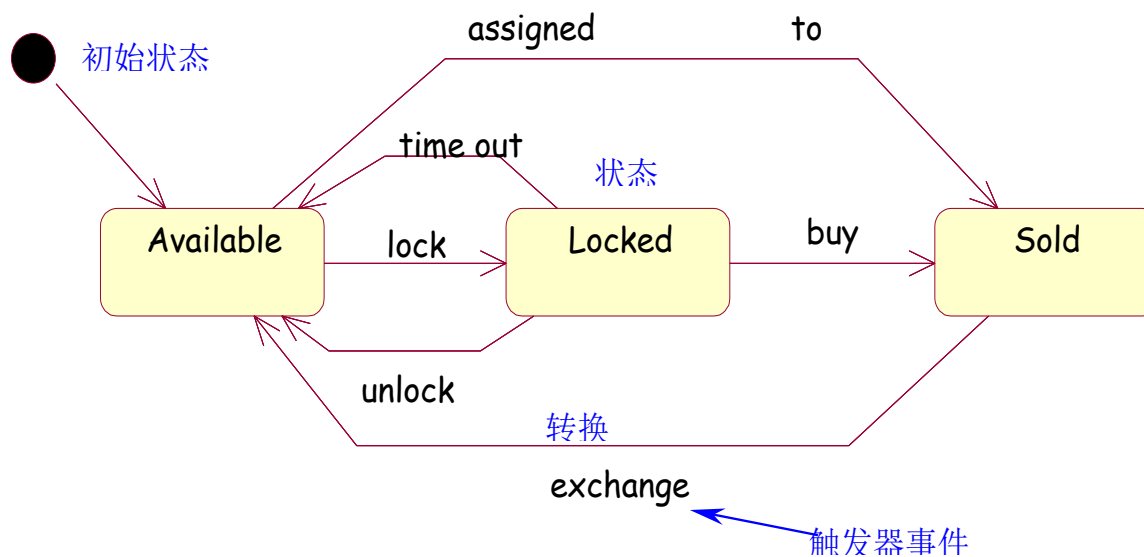




UML图及特征

◆ 状态图(State Chart Diagram)

状态图是一个类对象所可能经历的所有历程的模型图。状态图由对象的各个状态和连接这些状态的转换组成



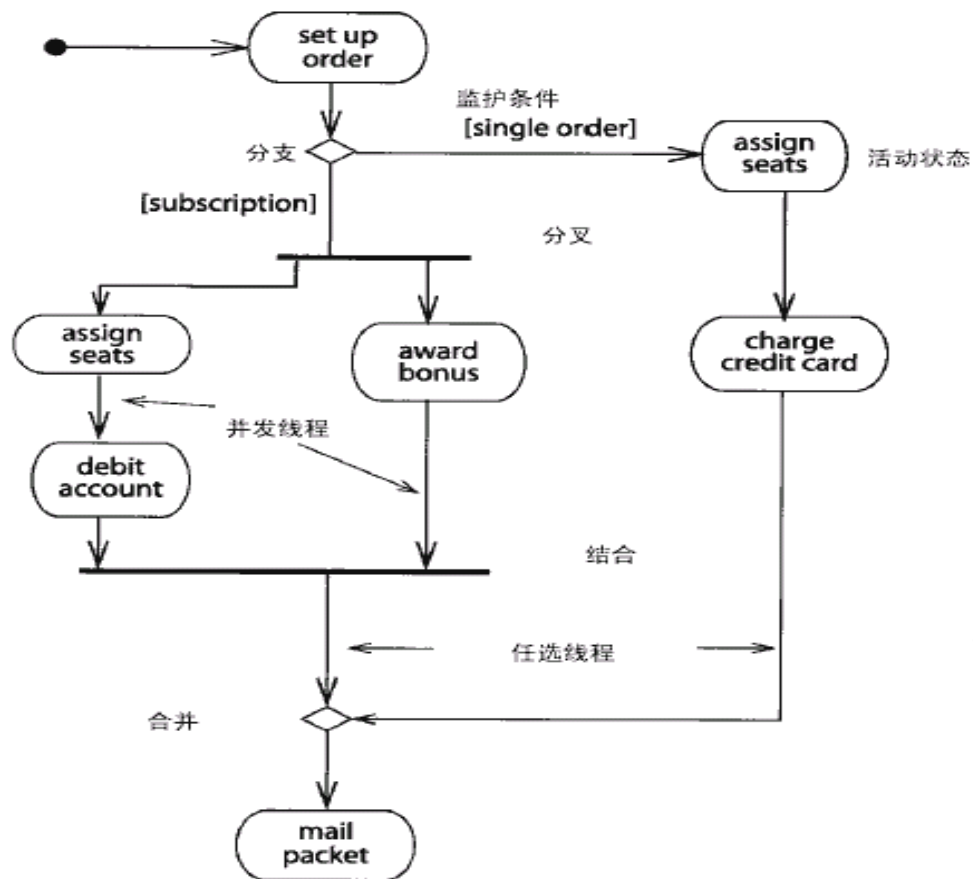


UML图及特征

◆ 活动图(Activity Diagram)

活动图是状态图的一个变体，用来描述执行算法的工作流程中涉及的活动

活动图描述了一组顺序的或并发的活动

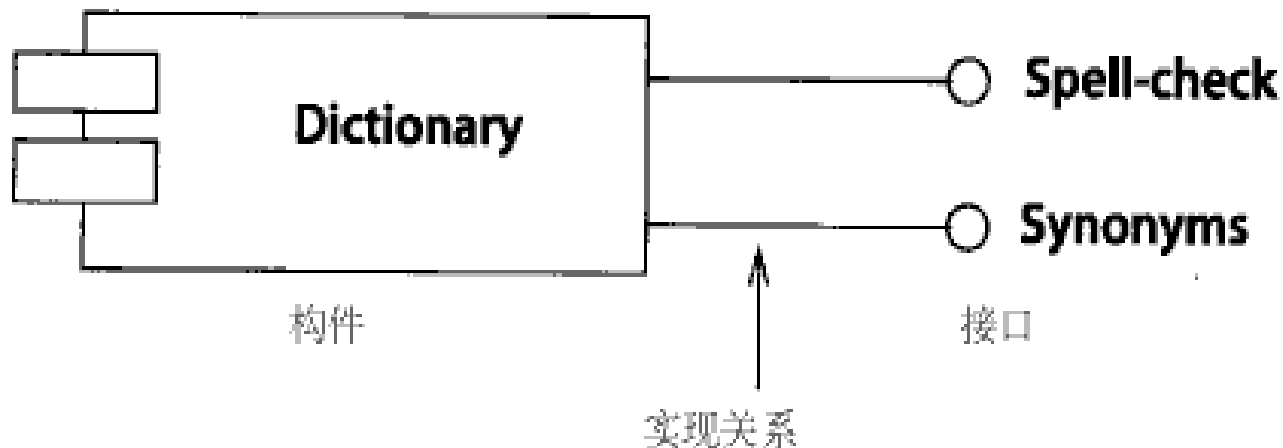




UML图及特征

◆ 构件图(Component Diagram)

构件图为系统的构件建模——构件即构造应用的软件单元——还包括各构件之间的依赖关系，以便通过这些依赖关系来估计对系统构件的修改给系统可能带来的影响

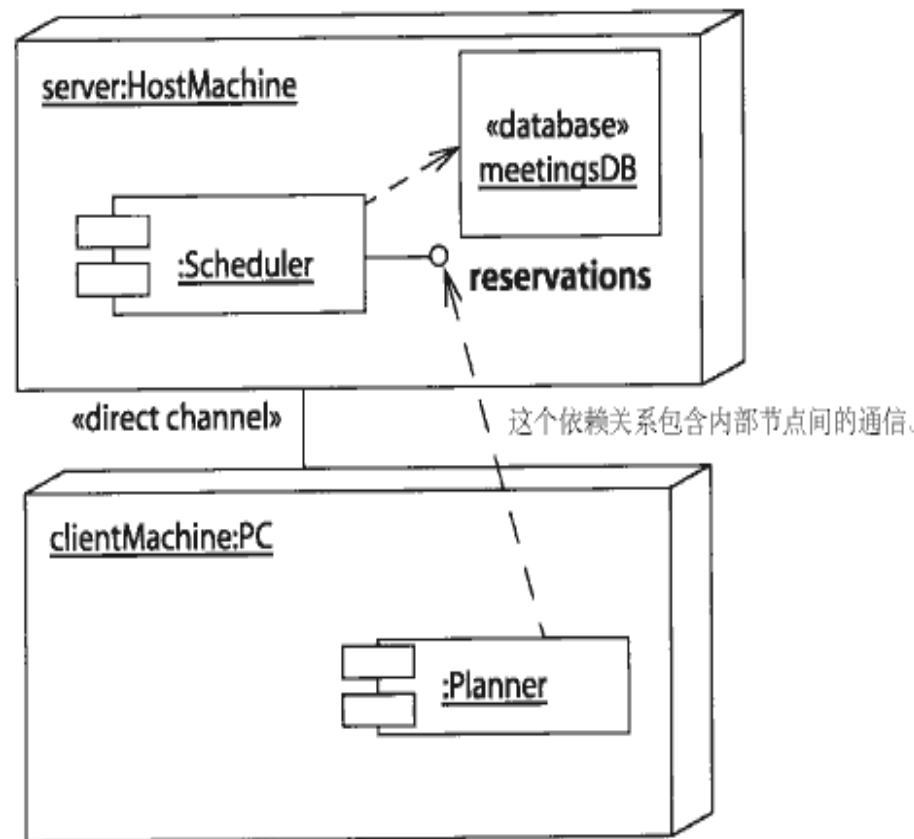




UML图及特征

◆ 部署图(Deployment Diagram)

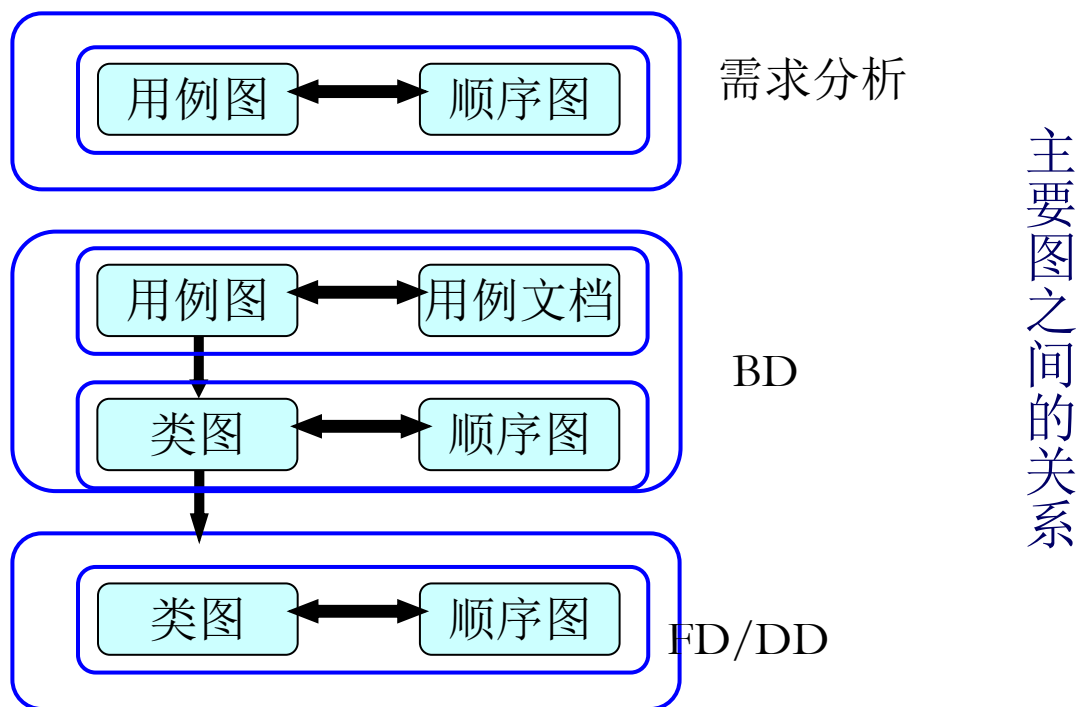
部署视图描述位于节点实例上的运行构件实例的安排。节点是一组运行资源，如计算机、设备或存储器。这个视图允许评估分配结果和资源分配





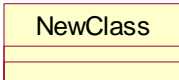
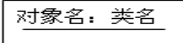
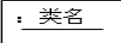




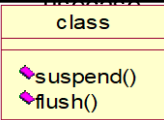
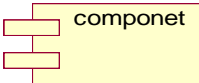
UML图及特征



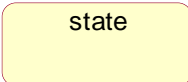
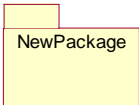




◆ 各UML图的关系





UML语法描述

类	是对一组具有相同属性、相同操作、相同关系和相同语义的对象的描述	
对象	  	
接口	是描述了一个类或构件的一个服务的操作集	
协作	定义了一个交互，它是由一组共同工作以提供某种协作行为的角色和其他元素构成的一个群体	
用例	是对一组动作序列的描述	
主动类	对象至少拥有一个进程或线程的类	
构件	是系统中物理的、可替代的部件	
参与者	在系统外部与系统直接交互的人或事物	

节点	是在运行时存在的物理元素	
交互	它由在特定语境中共同完成一定任务的一组对象间交换的消息组成	
状态机	它描述了一个对象或一个交互在生命期内响应事件所经历的状态序列	
包	把元素组织成组的机制	
注释事物	是UML模型的解释部分	
依赖	一条可能有方向的虚线	
关联	一条实线，可能有方向	
泛化	一条带有空心箭头的实线	
实现	一条带有空心箭头的虚线	