包　图

主要内容

- 概述
- 基本概念
- 建模方法

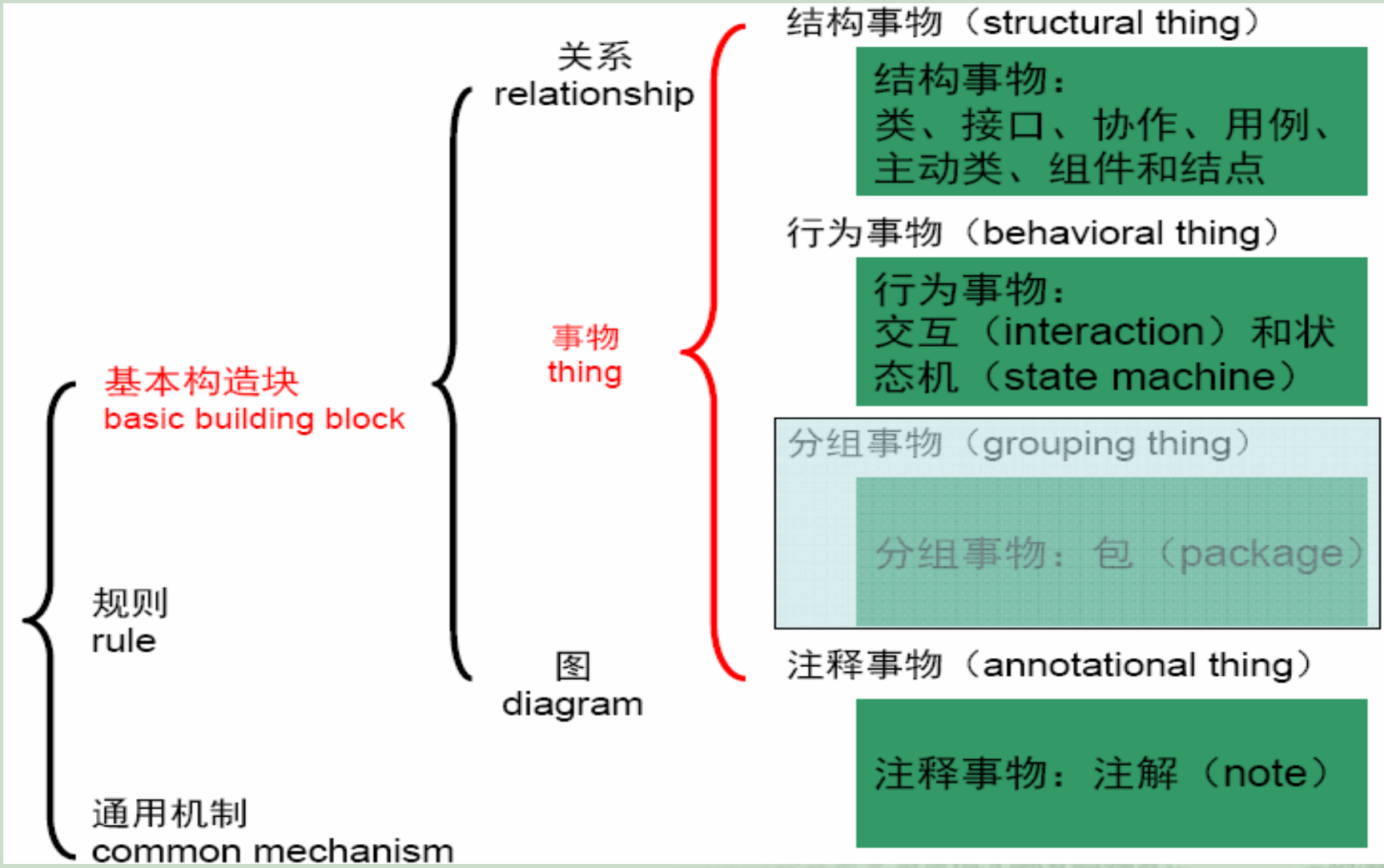
主要内容

- **概述**
- 基本概念
- 建模方法

概述

- 复杂的系统一般分成若干小的单元，以便人们可以一次只处理有限的信息，不会互相干扰。
- 在UML中，包是用于把建模元素组织成组的通用机制。
- 包图描述包和包之间的关系，是维护和控制系统总体结构的重要建模工具。

包


基本构造块
basic building block
规则
rule
通用机制
common mechanism
关系
relationship
事物
thing
图
diagram
结构事物（structural thing）
结构事物：
类、接口、协作、用例、
主动类、组件和结点
行为事物（behavioral thing）
行为事物：
交互（interaction）和状
态机（state machine）
分组事物（grouping thing）
分组事物：包（package）
注释事物（annotational thing）
注释事物：注解（note）

包图

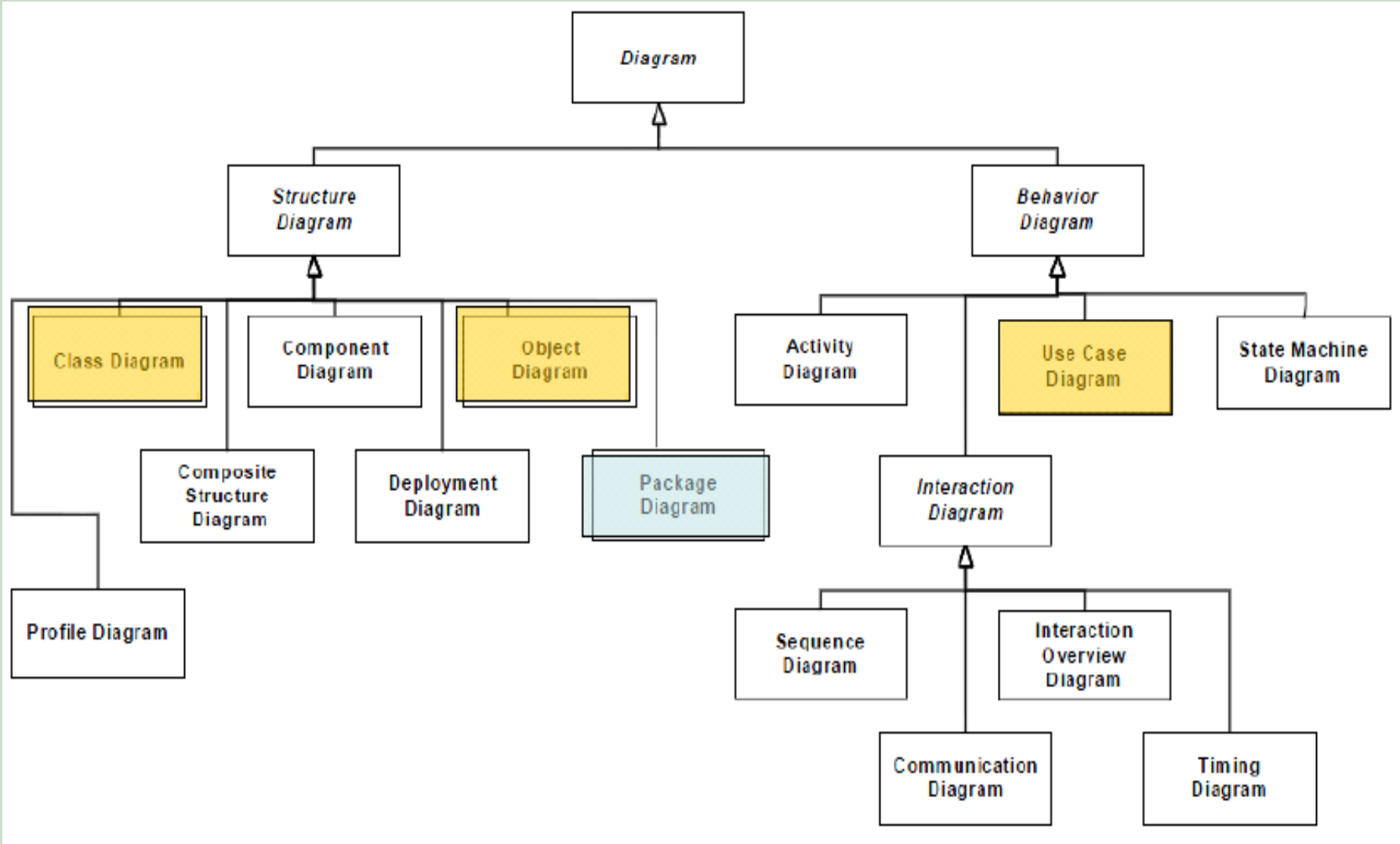


Diagram
Structure Diagram
Behavior Diagram
Class Diagram
Component Diagram
Object Diagram
Activity Diagram
Use Case Diagram
State Machine Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Interaction Diagram
Profile Diagram
Sequence Diagram
Interaction Overview Diagram
Communication Diagram
Timing Diagram

主要内容

- 概述
- **基本概念**
- 建模方法

包

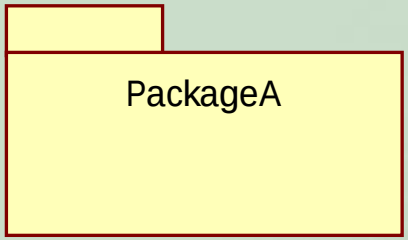
- A package is a general-purpose mechanism for organizing elements into groups
- 在UML中，包是
  - 一种分组事物
  - 一个建模元素的容器
- 通过包可将以下元素聚集在一起
  - 类
  - 用例
  - 构件
  - ……

包的作用

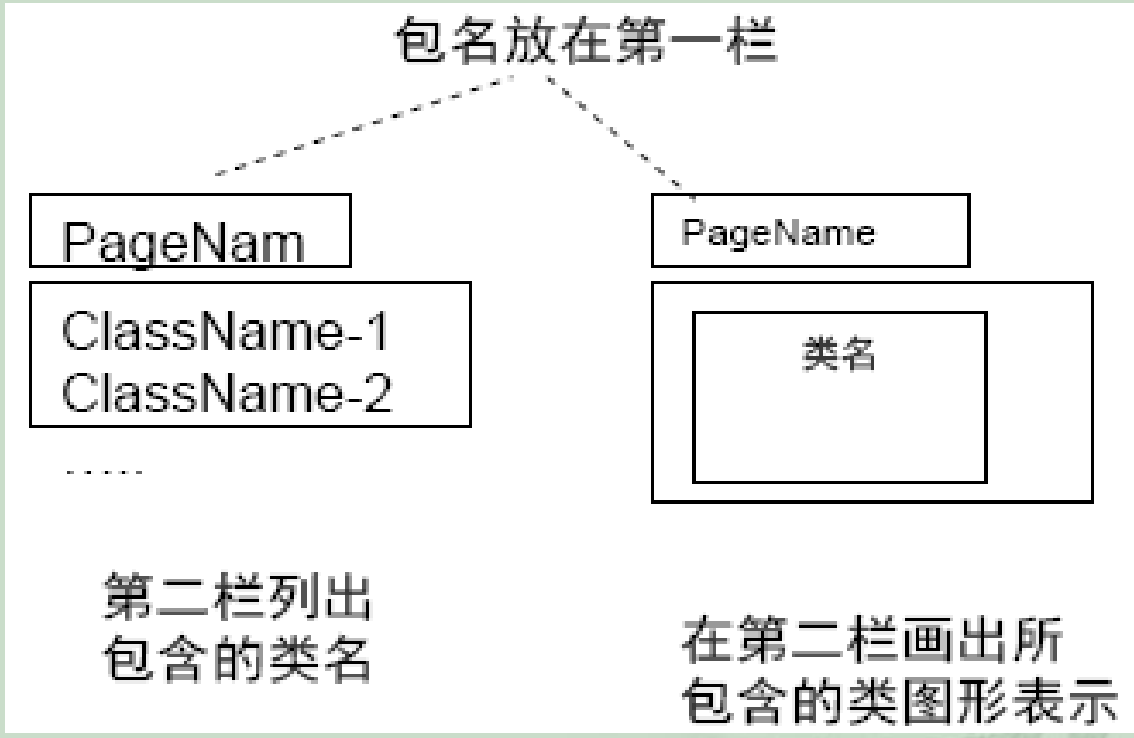
- 对语义上相关的元素进行分组
- 提供配置管理单元
- 在设计时，提供并行工作的单元
- 提供封装的命名空间，其中所有名称必须惟一

包的表示方法

- 在UML中，包用带标签的文件夹符号来表示



包的表示方法


包名放在第一栏
PageNam
ClassName-1
ClassName-2
.....
PageName
类名
第二栏列出
包含的类名
在第二栏画出所
包含的类图形表示

包的名称

- 每个包都必须有一个与其它包相区别的名称
- 标识包名称的格式
  - simple name，例如：Camera
  - path name，用该包的外围包的名字作为前缀，加上本身名字，比如：Sensors::Vision::Camera

包拥有的元素

- 包中可以包含其它建模元素
  - class
  - interface
  - component
  - node
  - use case
  - package
  - ……

包内元素的可见性

- 包内元素的可见性控制了包外部元素访问
  - “+”表示“public”
  - “#”表示“protected”
  - “-”表示“private”

包的版型

- 《system》版型：表示正在建模的整个系统
- 《subsystem》版型：表示正在建模的系统中某个独立的部分
- 《facade》版型：是其它包的视图，主要为复杂的包提供简略视图
- 《stub》版型：是一个代理包，通常用于分布式系统的建模
- 《framework》版型：表示一个框架，框架是一个领域中的应用系统提供可扩充模板的体系结构模式

包之间的关系

- 依赖关系
  - << import>>
  - << access>>
  - <<use>>
  - << trace>>
  - ……
- 泛化关系

《import》关系

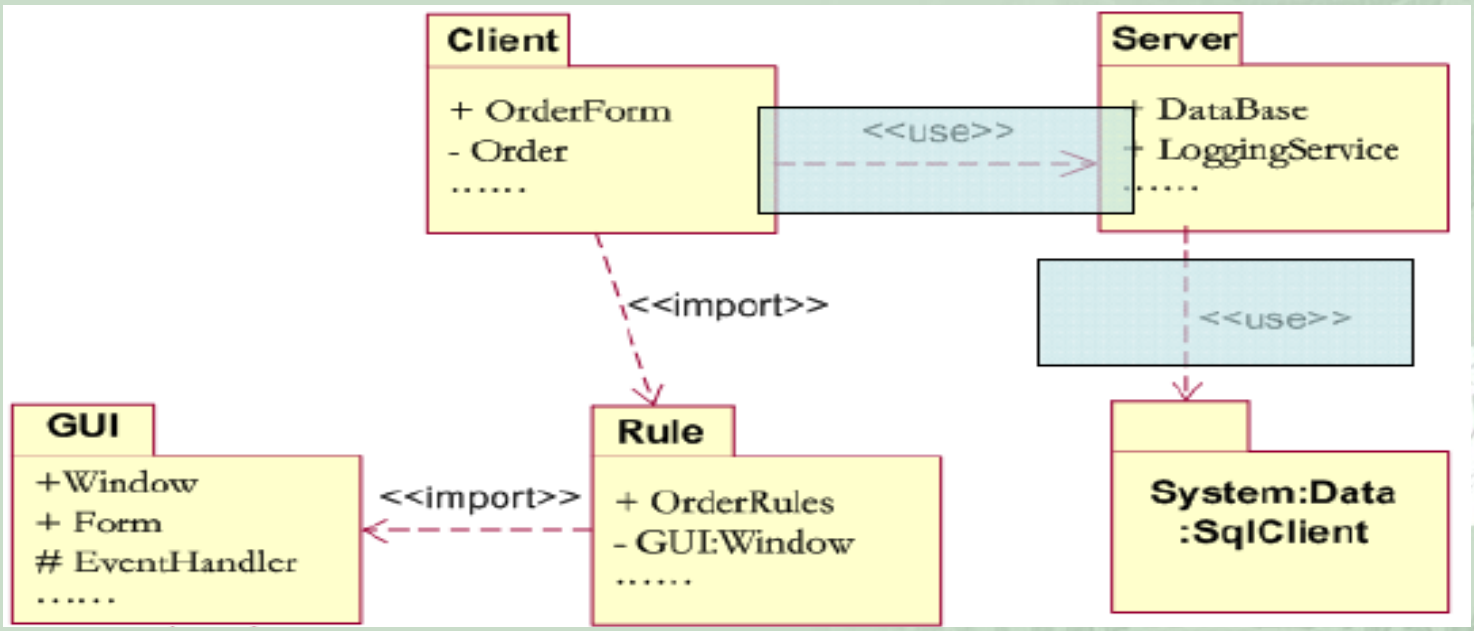
- 是最普遍的包依赖类型
- 说明提供者包的命名空间将被添加到客户包的命名空间中，客户包中的元素也能够访问提供者包的所有公共元素

《access》关系

- 客户包中的元素能访问提供者包中的所有公共元素，但是命名空间不合并

《use》关系

- 是一种默认的依赖关系
- 说明客户包（发出者）中的元素以某种方式使用提供者包（箭头指向的包）的公共元素
- 也就是说客户包依赖于提供者包

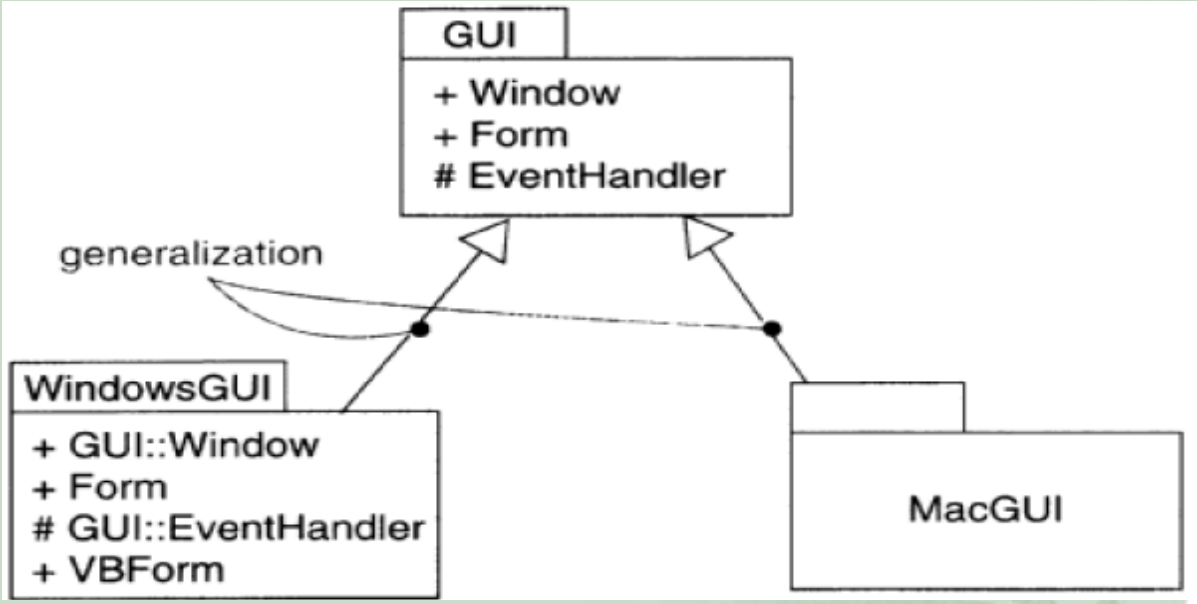


《trace》关系

- 表示一个包到另一个包的历史发展
- 不常用

泛化关系

- 包之间的泛化关系很少用到，Rose中也不支持



包图

- 包图描绘包以及包之间的关系
- UML2.X中的新概念，在之前一直是非正式的部分

主要内容

- 概述
- 基本概念
- **建模方法**

绘制包图的基本过程

- 浏览特定体系结构视图中的建模元素，找出语义或其它方面接近的元素
- 把这些元素放到一个包中
- 对每个包找出可以在包外访问的元素，将其标记公有，把其它元素标记为受保护或私有
- 确定包之间的依赖关系
- 确定包之间的泛化关系

包图建模原则

- 重用等价原则
  Reuse Equivalency Principle，REP
- 共同重用原则
  Common Reuse Principle，CRP
- 共同闭包原则
  Common Closure Principle，CCP
- 非循环依赖原则
  Acyclic Dependencies Principle，ADP

重用等价原则

- 把类放入包中时, 应考虑把包作为可重用的单元
- 重用的粒度就是发布的粒度
- 该原则倾向于把包做的越小越好

共同重用原则

- 不会一起使用的类不要放在同一个包中
- 一个包中的所有类应该是共同重用的，如果重用了包中的一个类，就应该重用包中的所有类
- 该原则倾向于把包做的尽可能小

共同闭包原则

- 把需要同时改变的类放在同一个包中
- 以下情况两个类放在一个包中：
  --一个类的改变要求另一个类做相应改变
  --删除一个类后，另一个类变成多余
  --两个类间有大量的消息发送
- 该原则倾向于将包做的尽可能的大

非循环依赖原则

- 包之间的依赖关系不要形成循环
- 包A依赖包B，包B依赖包C，包C依赖包A，形成了循环，怎么办？

  --将这些包中的元素放入一个更大的包中

几个原则的说明

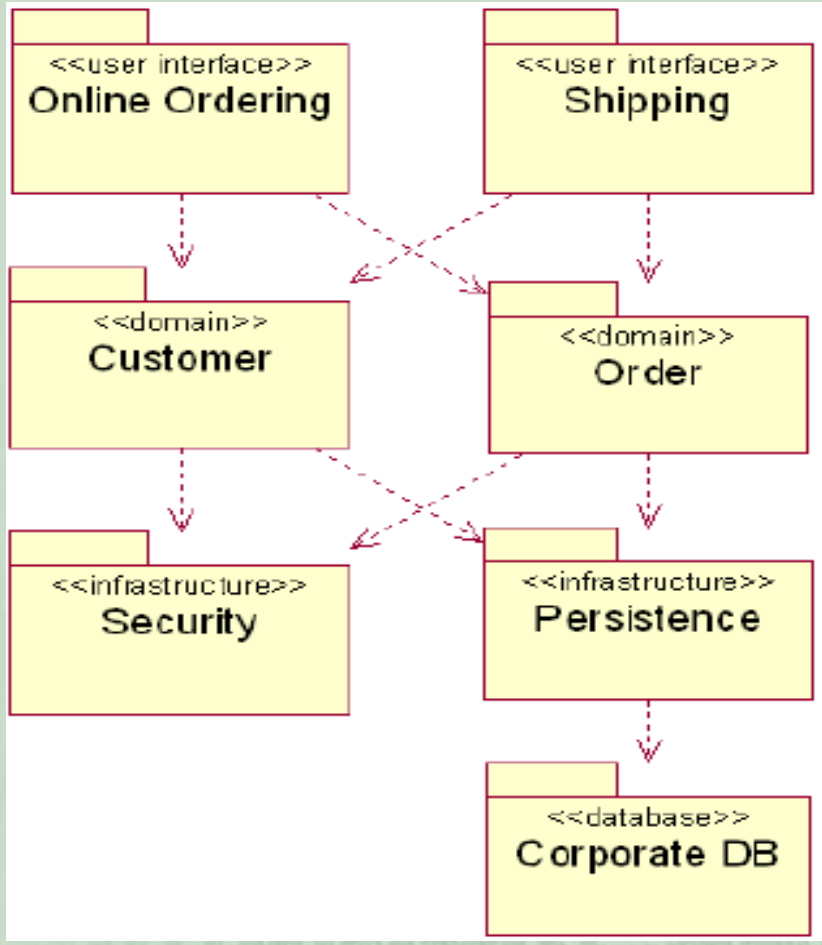
- 粒度角度（内聚性）：共同重用原则，重用等价原则，共同闭包原则
- 稳定性角度（耦合性）：非循环依赖原则
- 共同重用原则，重用等价原则是从重用者（包的使用者）角度考虑
- 共同闭包原则是从维护者（包的作者）角度考虑

包图建模风格

- 类的包图建议（使用类的包图从逻辑上对设计进行组织）
  - 一个框架内的类属于一个包
  - 一般位于同一继承层次上的类属于同一个包
  - 通过聚合或者组合关系相关联的类往往属于同一个包
  - 相互之间协作很多的类通常属于同一个包

包图建模风格

- 在垂直方向上分层组织类的包图
- 包的位置反映了系统体系结构的逻辑分层。按照惯例，一般是采用自上而下的方式体现分层结构
- 在包上用版型指明体系结构的层次



包图建模风格

- 在用例包图中包含参与者