软件系统分析与设计
System Analysis & Design

M210007B

Monday, May 15, 2023

/-1 软件系统静态建模

第7章

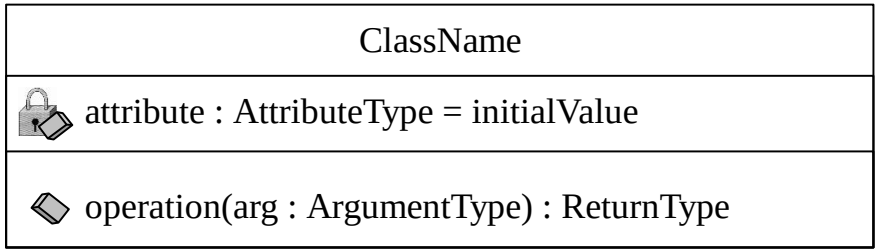
类图、对象图和包图

7-1 类图
Class Diagram

4-5 类

类

- 类是分享同样的属性、操作、关系和语义的对象的集合。
- 类是现实世界中的事物的抽象，当这些事物存在于真实世界中时，它们是类的实例，并被称为对象。类可以实现一个或多个接口。
- 类的UML符号是划分成3个格子的长方形。

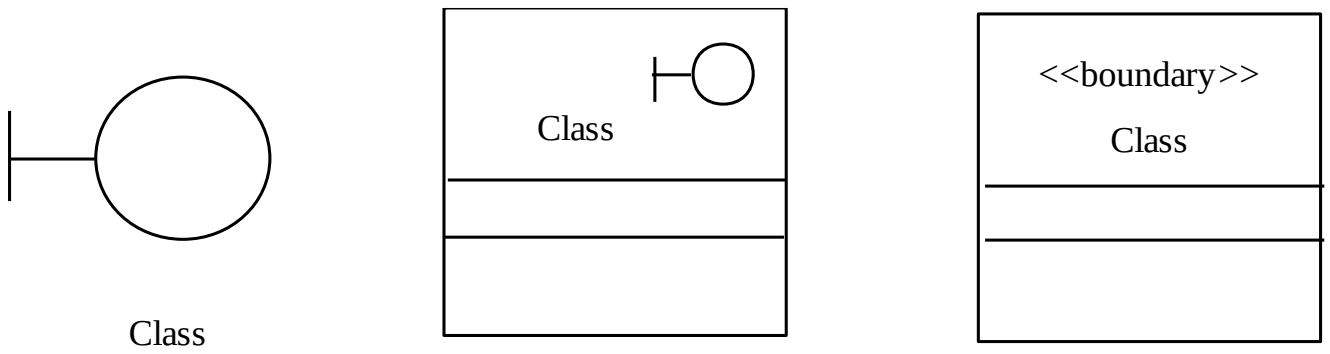


类

➢边界类

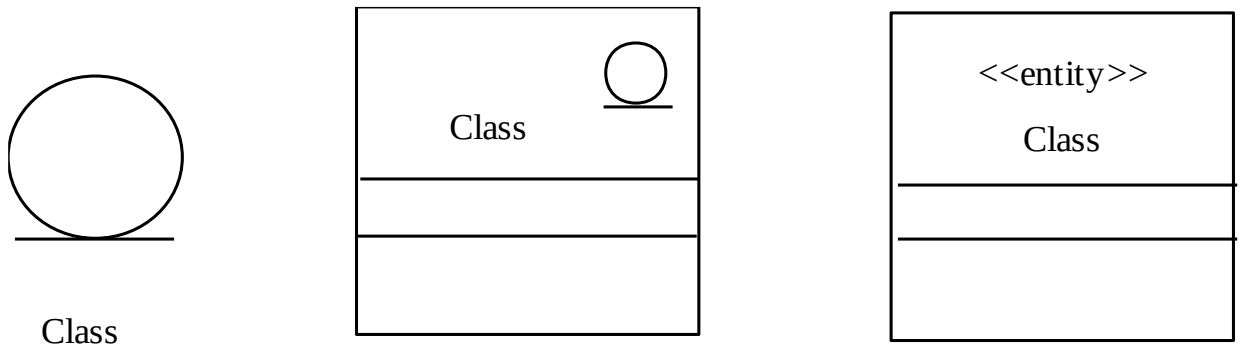
✓边界类处理系统环境与系统内部之间的通信，边界类为用户或另一个系统（即参与者）提供了接口。

✓边界类的UML符号表示



类

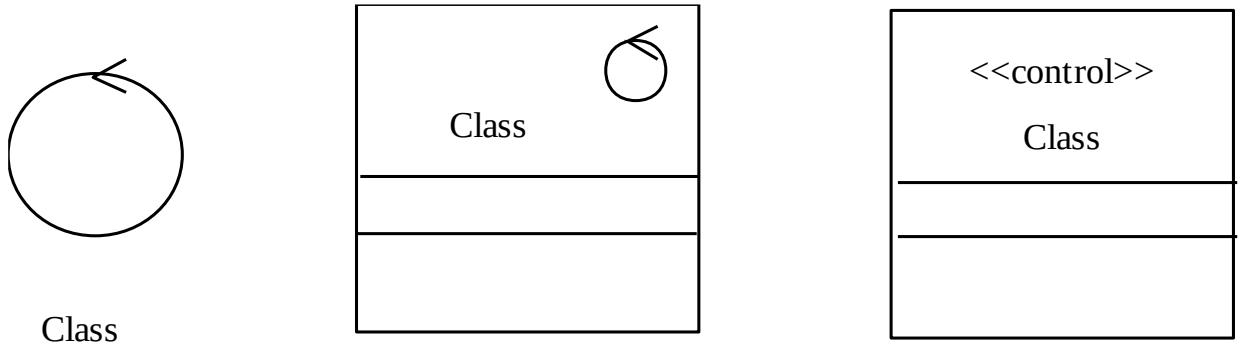
- 实体类
  - ✓实体类是模拟必须被存储的信息和其关联行为的类。
  - ✓实体类的UML符号表示



类

➢控制类

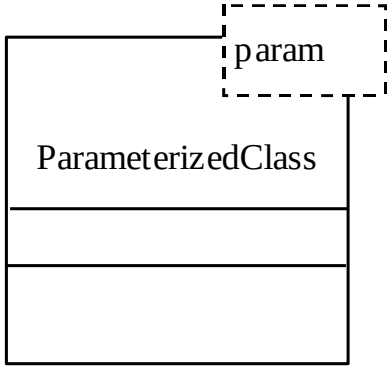
✓控制类是用来为特定于一个或多个用例的控制行为建模的类。



类

➢参数类

- ✓参数类又被称为模板类（Template Classes），模板类定义了类族。
- ✓模板不能直接使用，要首先实例化模板类，实例化包括将这些形式模板参数绑定到实际的参数。
- ✓参数类的UML符号是在类的UML符号表示的右上角加一个虚线框，在这个虚线框中列出模板参数。



第3章 UML的关系

内容

- 依赖（Dependency）关系
- 类属（Generalization）关系
- 关联（Association）关系
- 实现（Realization）关系

3-1 依赖关系
Dependency

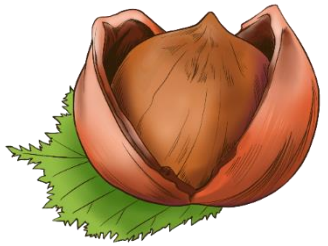
依赖关系

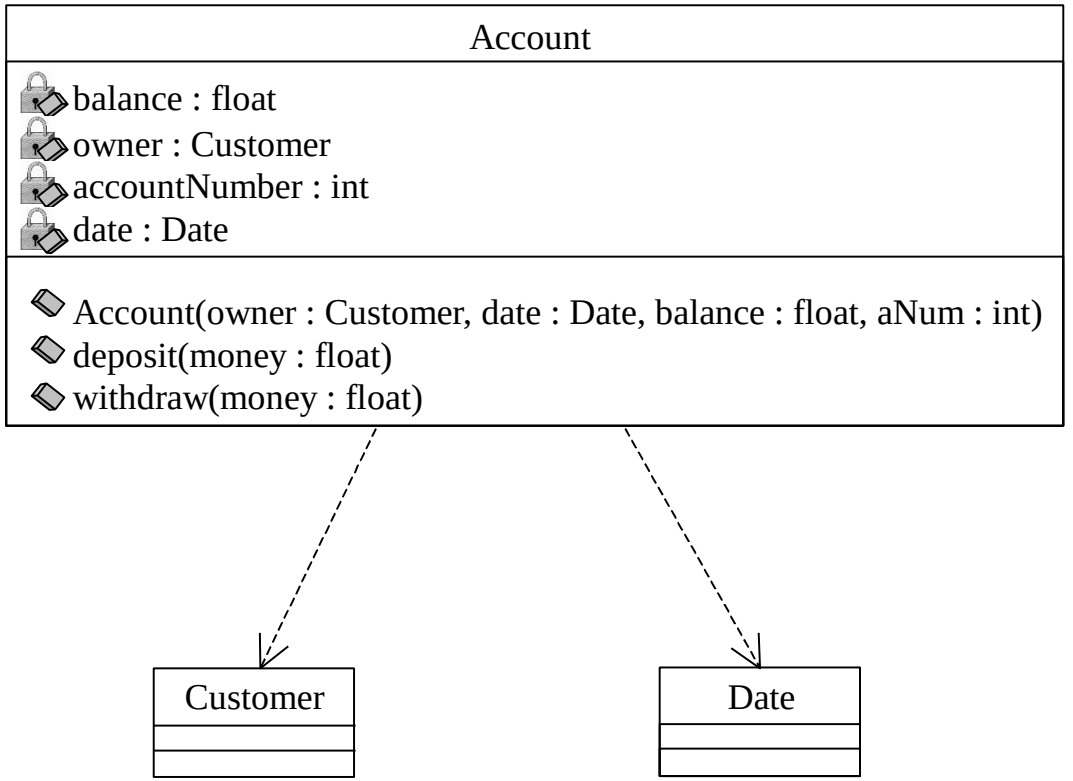
- 如果一个模型元素的变化会影响另一个模型元素（这种影响不必是可逆的），那么就说在这两个模型元素之间存在依赖关系。
- 依赖关系的UML符号表示是带箭头的虚线，指向被依赖的模型元素

依赖关系

举个例子

Examples





依赖关系：衍型

- UML定义了许多可以应用于依赖关系的衍型
  - 用于类图中类和对象之间依赖关系的衍型
    - (1) <<bind>>
    - (2) <<derive>>
    - (3) <<friend>>
    - (4) <<instanceOf>>
    - (5) <<instantiate>>
    - (6) <<powertype>>
    - (7) <<refine>>
    - (8) <<use>>

依赖关系：衍型

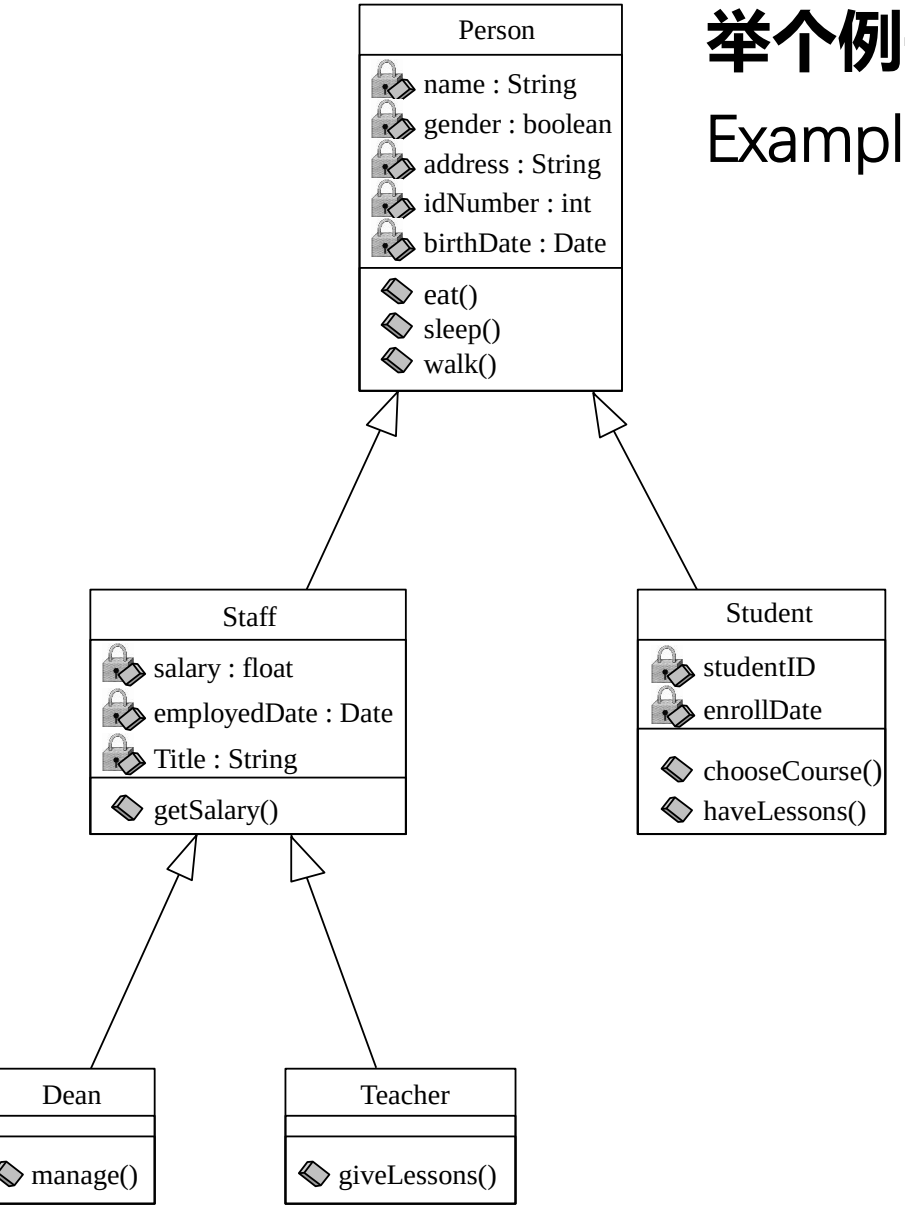
- 可以用于包间依赖关系的衍型
  - （9）<<access>>
  - （10）<<import>>
- 可以用于用例之间的依赖关系的衍型
  - （11）<<extend>>
  - （12）<<include>>
- 可以用于为对象间的交互作用建模的衍型
  - （13）<<become>>
  - （14）<<call>>
  - （15）<<copy>>
- 可以应用于状态机上下文中的衍型
  - （16）<<send>>
- 另外还有一个有用的衍型
  - （17）<<trace>>

3-2 类属关系
Generalization

类属关系

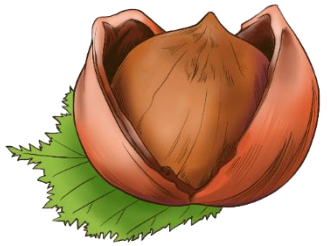
- 类属（Generalization）关系描述了一般事物与该事物的特殊种类之间的关系，也即父元素与子元素之间的关系。
- 在UML中，类属关系用带空心箭头的实线表示，箭头指向父元素。

类属关系



举个例子

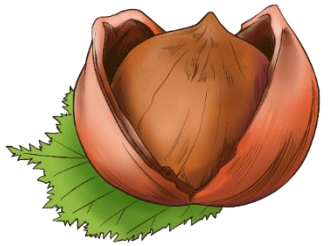
Examples

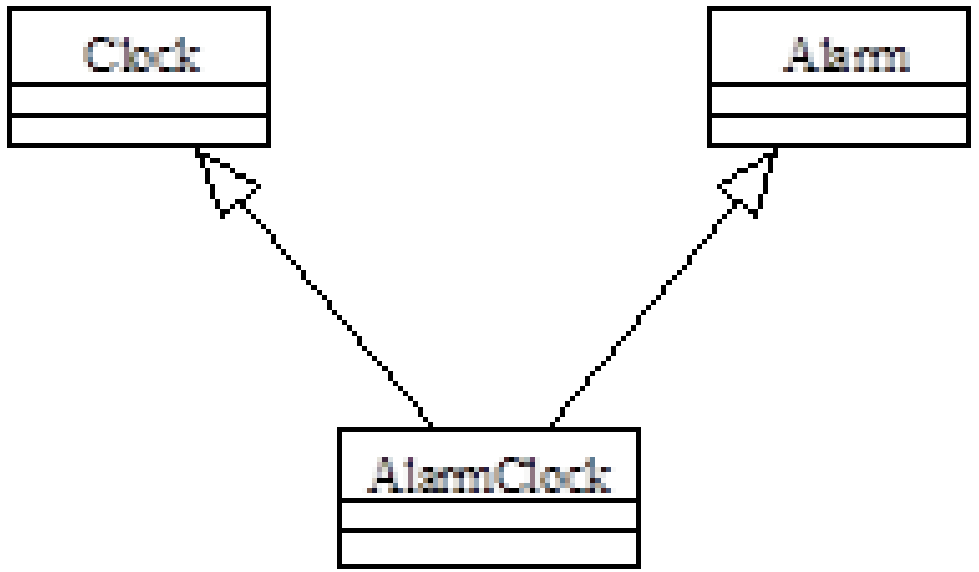


类属关系——多继承

举个例子

Examples





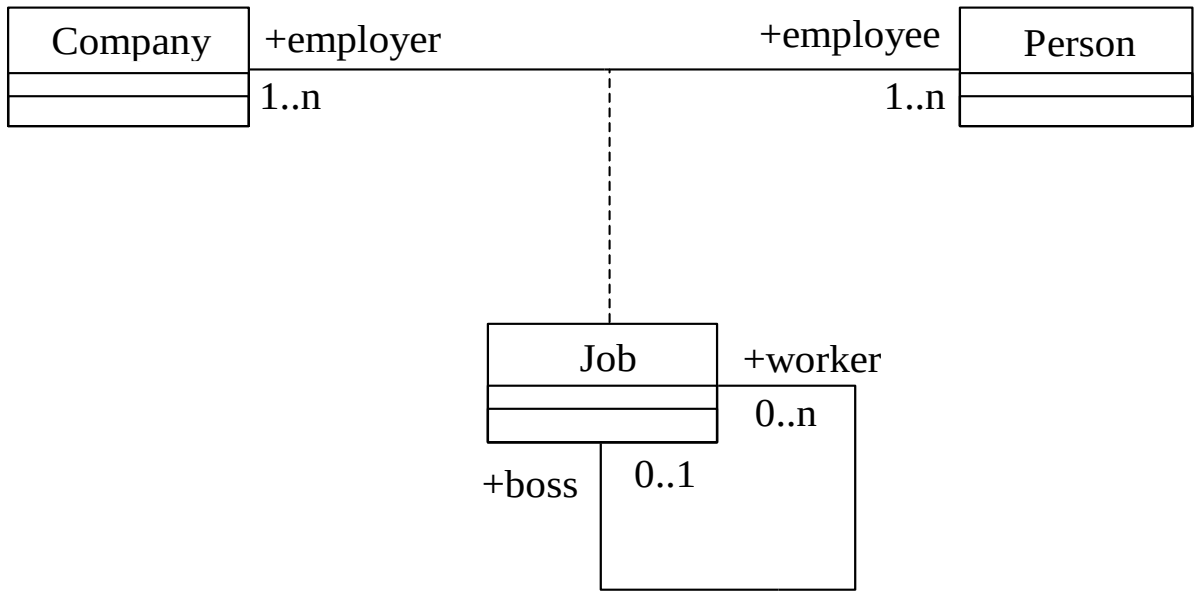
3-3 关联关系
Association

关联关系

- 关联关系表示两个类之间存在某种语义上的联系。它是一种结构关系，规定了一种事物的对象可以与另一种事物的对象相连。
- 关联关系的UML符号是一条实线。

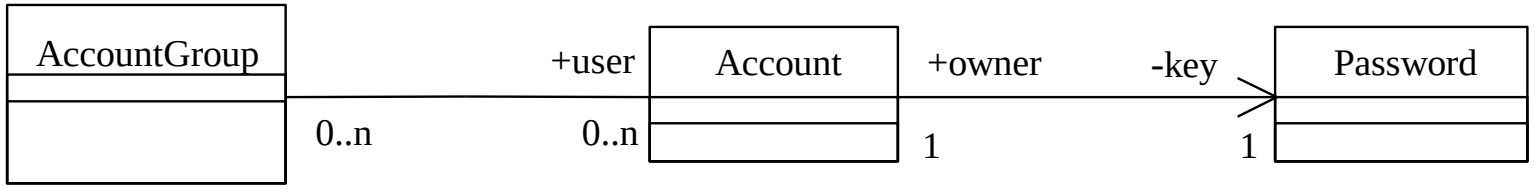
关联关系

- 角色（Role）与阶元（Multiplicity）
  - 关联两头的类都以某种角色参与关联。
  - 阶元表示有多少个对象参与该关联。



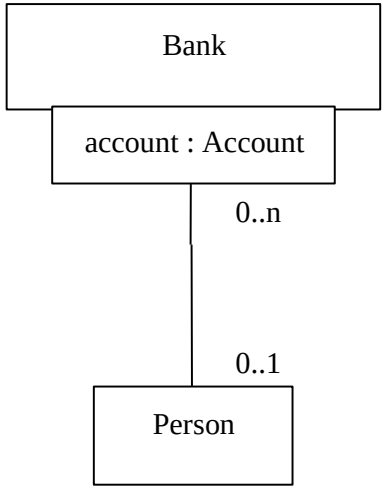
关联关系

- 导航
  - ✓关联关系是可导航的意味着给定一端的一个对象，可以容易、直接地到达另一端的对象，因为源对象通常含有对目标对象的引用。
- 可见性
  - ✓在UML中，通过对角色名附加可见性符号，可以为关联端规定3种可见性：公共可见性、私有可见性和保护可见性。



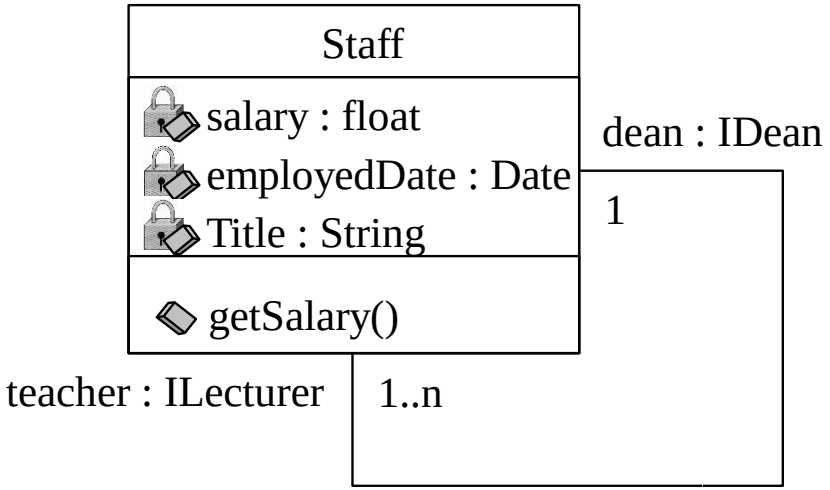
关联关系

- 限定符
  - ✓限定符是属性或属性列表，这些属性的值用来划分与某个对象通过关联关系连接的对象集。
  - ✓限定符是这个关联的属性。



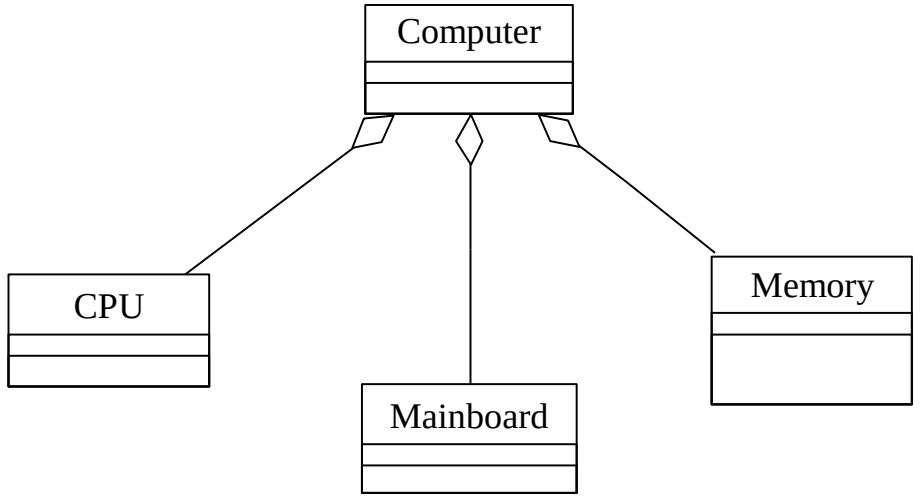
关联关系

- 接口说明符


Staff
salary : float
employedDate : Date
Title : String
getSalary()
dean : IDean
1
teacher : ILecturer
1..n

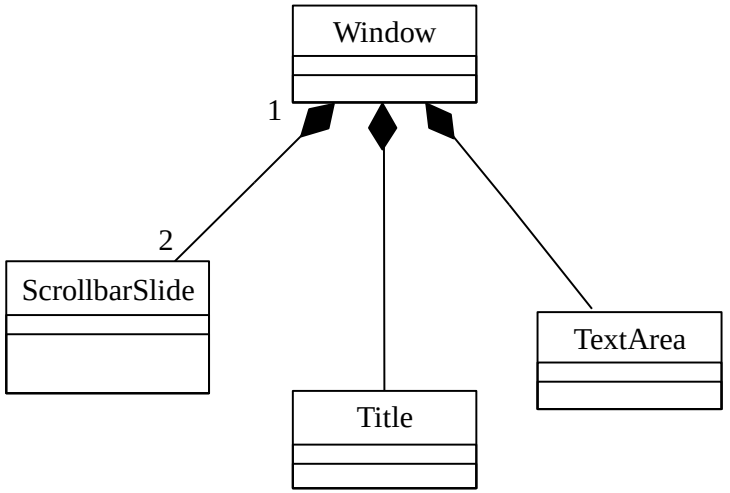
关联关系：聚合关系

- 聚合关系是一种特殊的关联关系。聚合表示类之间的关系是整体与部分的关系，它代表了“has-a”（拥有）关系，也即作为整体的对象拥有作为部分的对象。



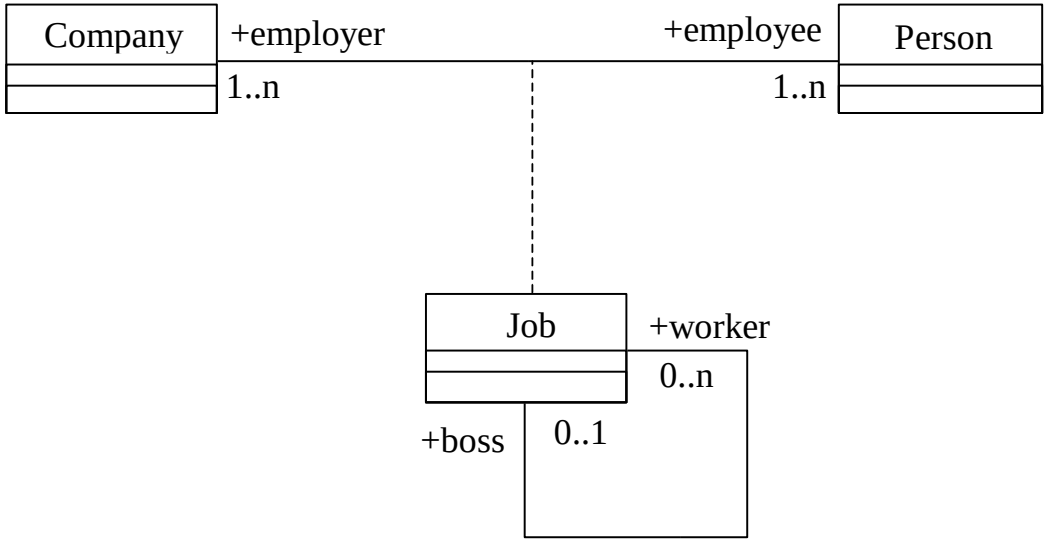
关联关系：组合关系

- 组合是聚合的变种，它加入了一些重要的语义。
- 在组合关系中，整体与部分之间具有很强的所有关系和一致的生命周期。



关联关系：关联类

- 在UML中，关联类是一个既具有关联属性又具有类属性的建模元素。
- 关联类可以被看作一个具有类特性的关联，或具有关联特性的类。



3-4 实现关系 Realization

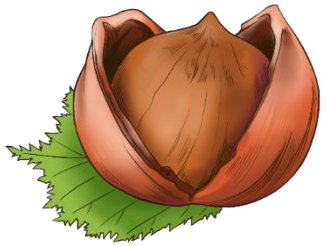
实现关系

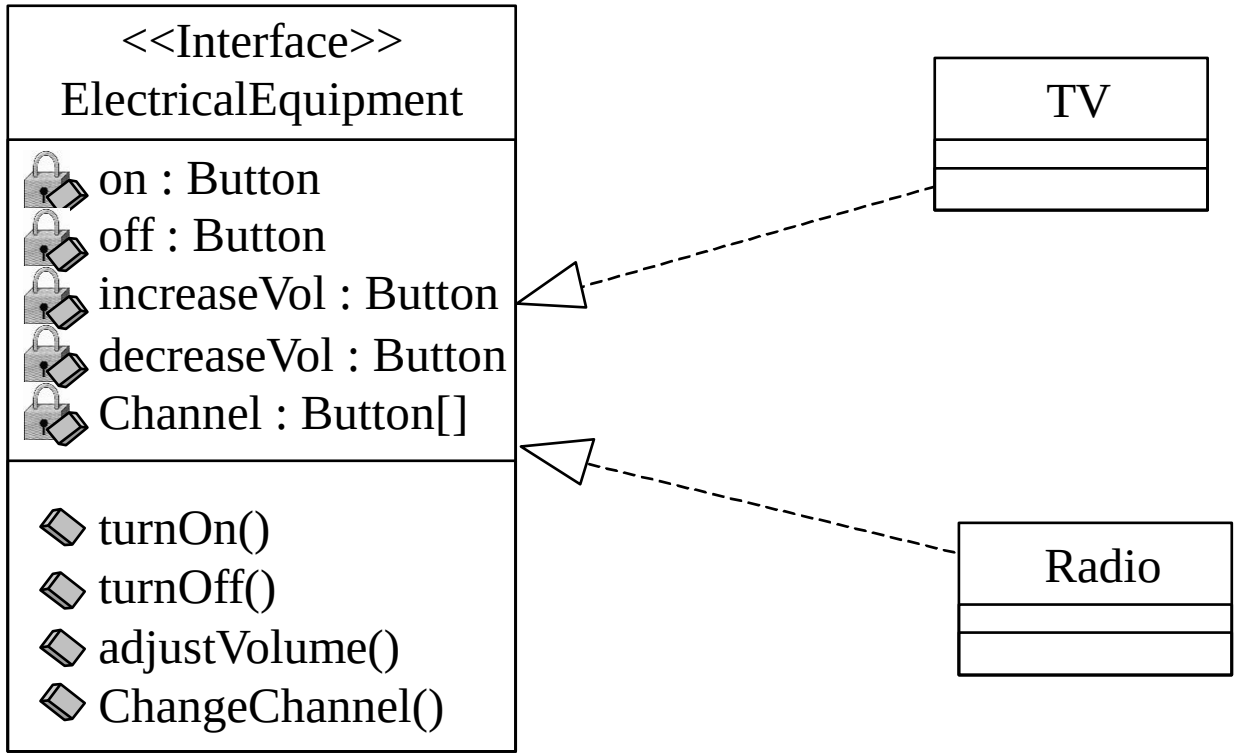
- 实现关系是分类器之间的语义关系，一个分类器规定协议，另一个分类器保证实现这个协议。
- 大多数情况下，实现关系被用来规定接口和实现接口的类或组件之间的关系。
- 实现关系的UML符号表示用带有空心箭头的虚线表示

实现关系

举个例子

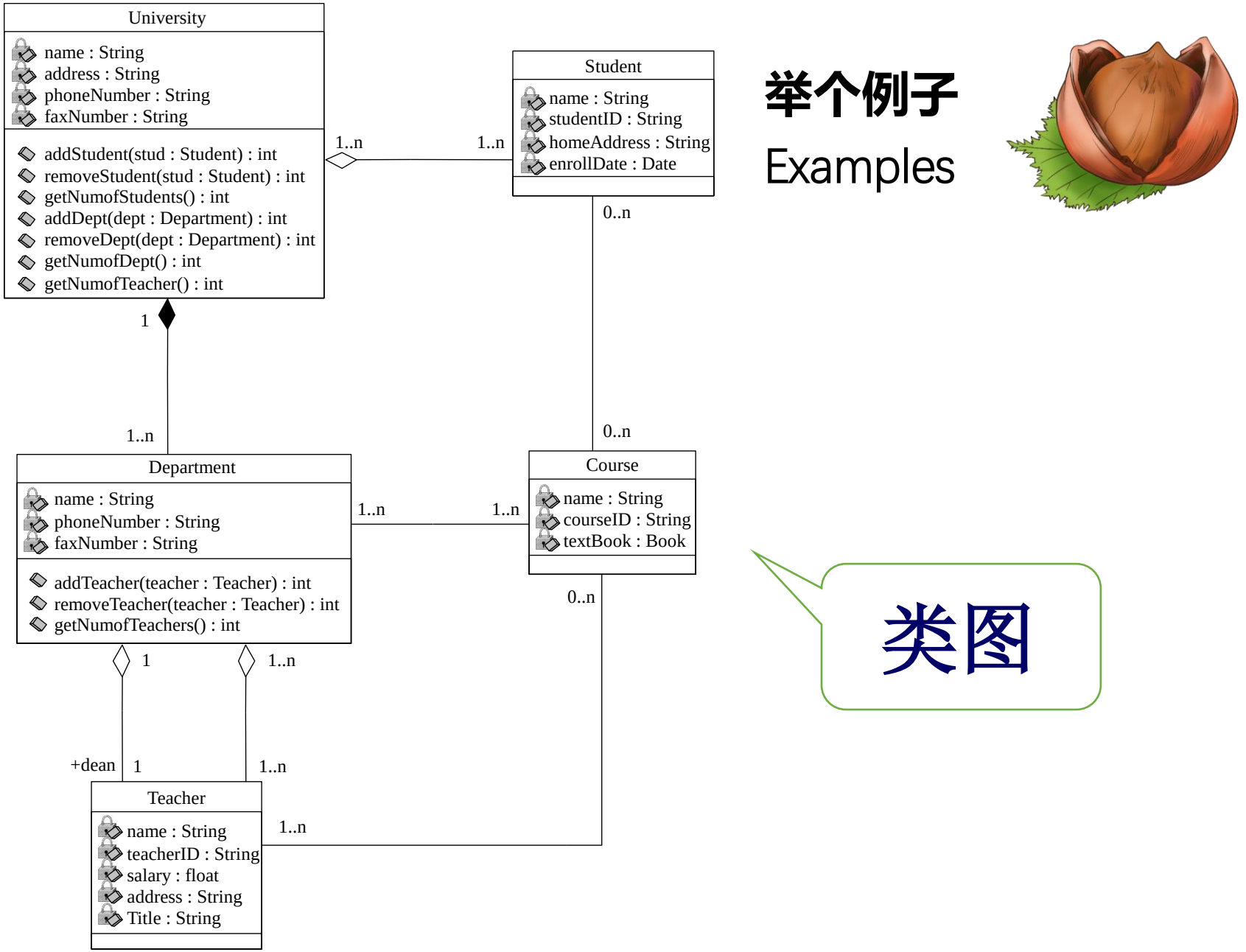
Examples





类图

- 类图是面向对象系统建模最常用的图，类图描述了类、接口、协作以及它们之间的关系。
- 类图用来为系统的静态设计视建模。
- 类图的组成部分包括:
  - ✓类。
  - ✓接口。
  - ✓协作。
  - ✓依赖、类属、实现或关联关系。
  - ✓类图还可以含有注释、约束、包或子系统。



举个例子

Examples

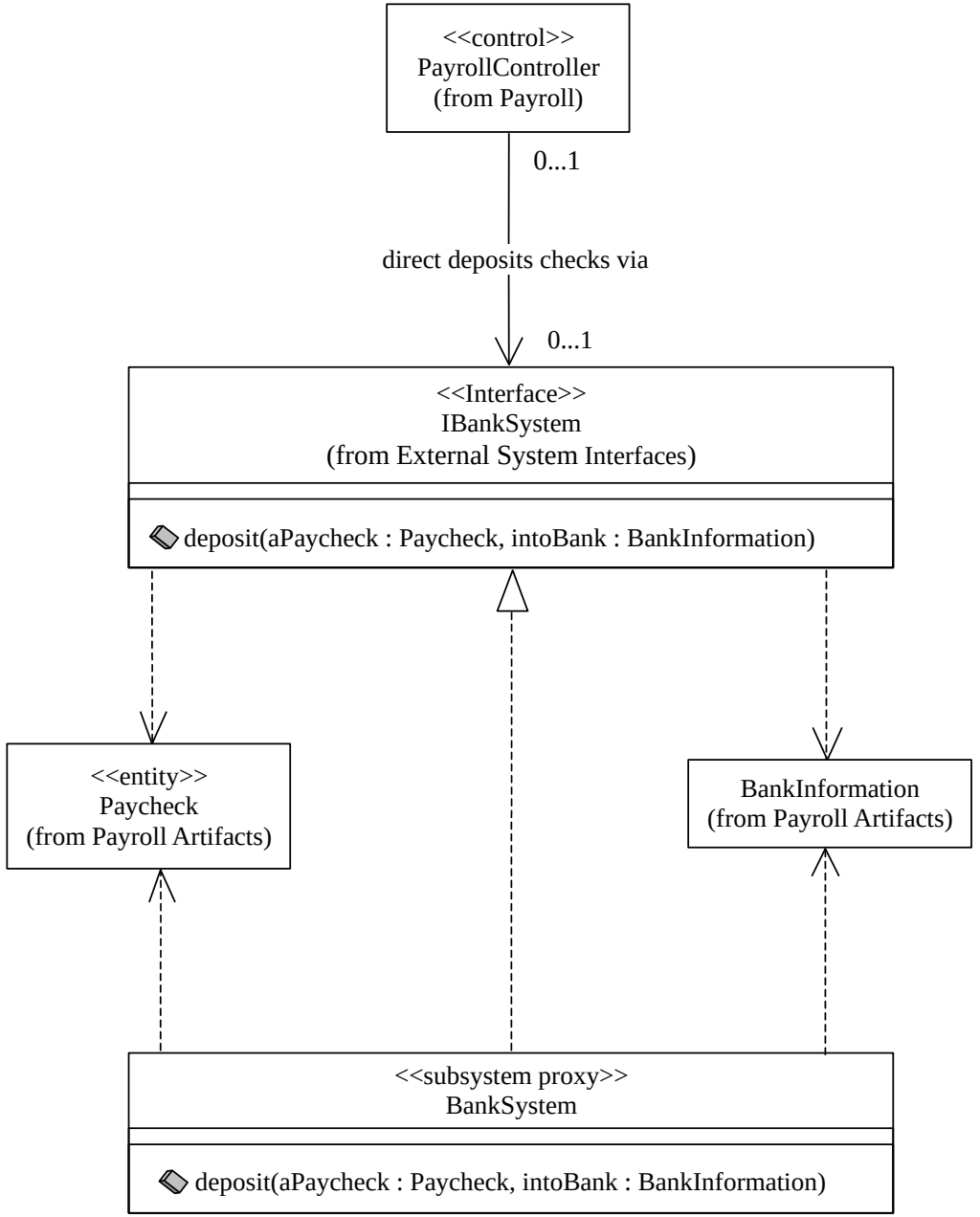
类图

类图

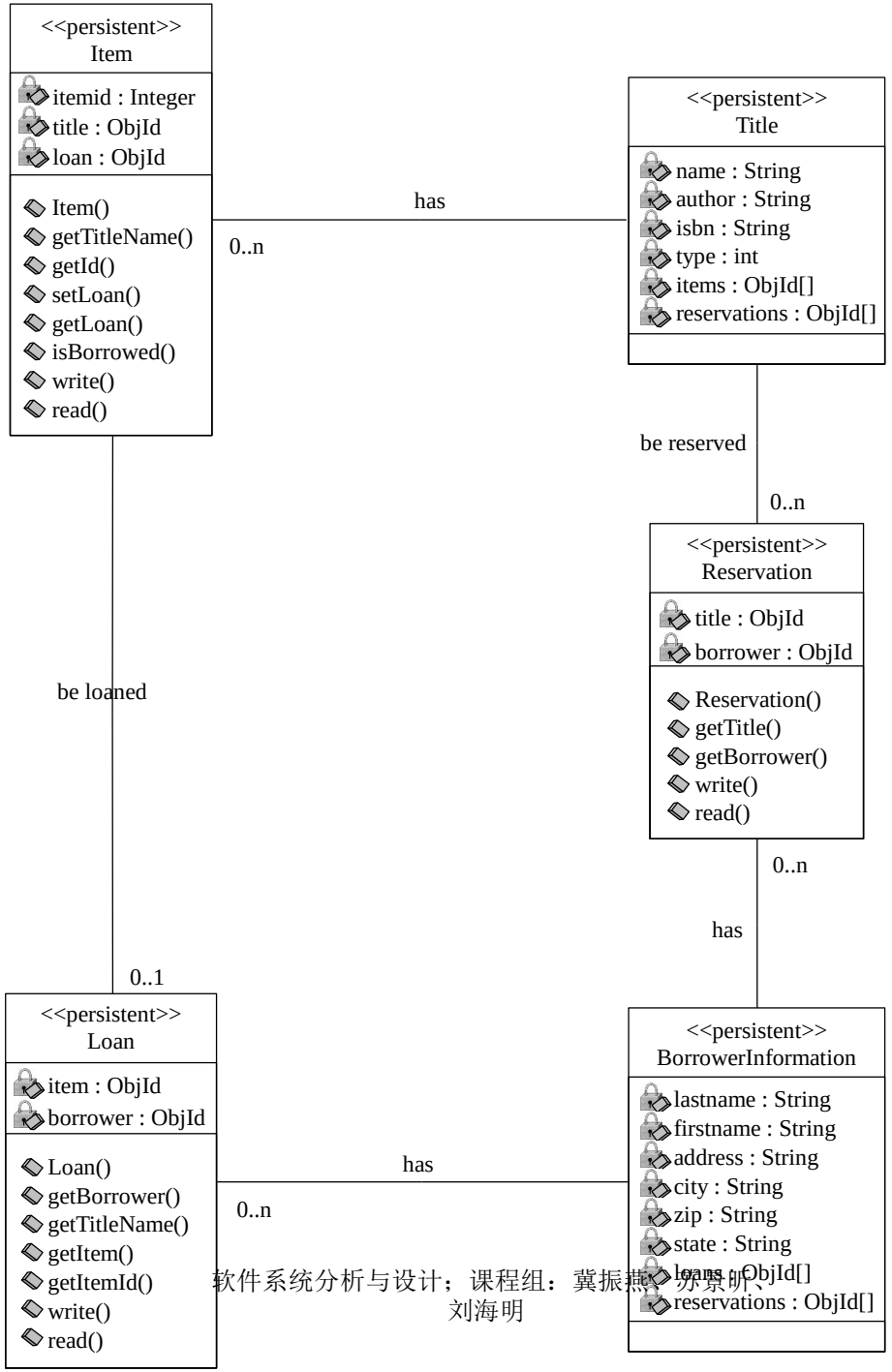
- 类图的划分
  - 概念层 (Conceptual)
  - 说明层 (Specification)
  - 实现层 (Implementation)
- 类图的应用
  - 为系统的词汇表建模
  - 为简单的协作建模
  - 为逻辑的数据库模式建模

类图的应用

模拟协作


<<control>>
PayrollController
(from Payroll)
0...1
direct deposits checks via
0...1
<<Interface>>
IBankSystem
(from External System Interfaces)
deposit(aPaycheck : Paycheck, intoBank : BankInformation)
<<entity>>
Paycheck
(from Payroll Artifacts)
BankInformation
(from Payroll Artifacts)
<<subsystem proxy>>
BankSystem
deposit(aPaycheck : Paycheck, intoBank : BankInformation)

类图的应用

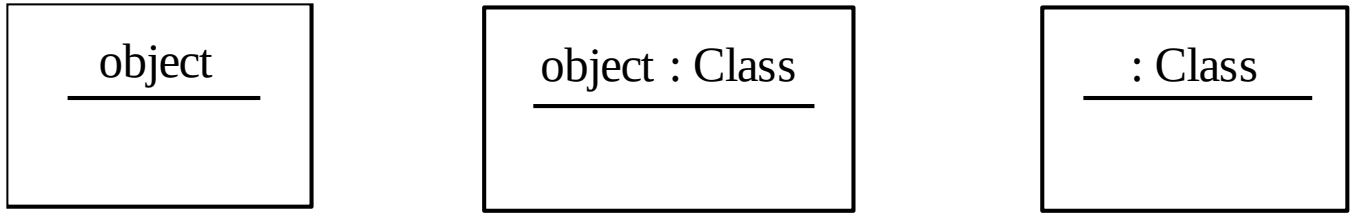


数据库的逻辑结构

7-2 对象图
Object Diagram

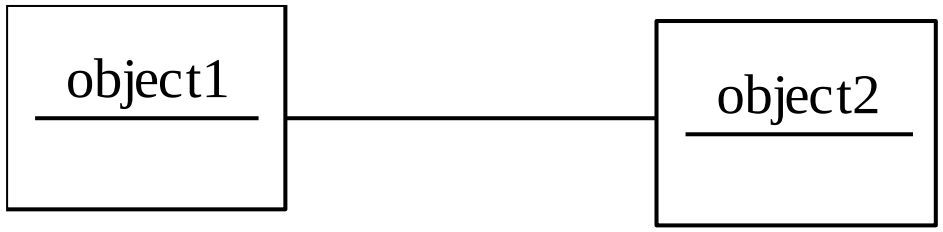
对象

- 对象代表了类的一个特定实例。对象具有身份（Identity）和属性值（Attribute Values）。

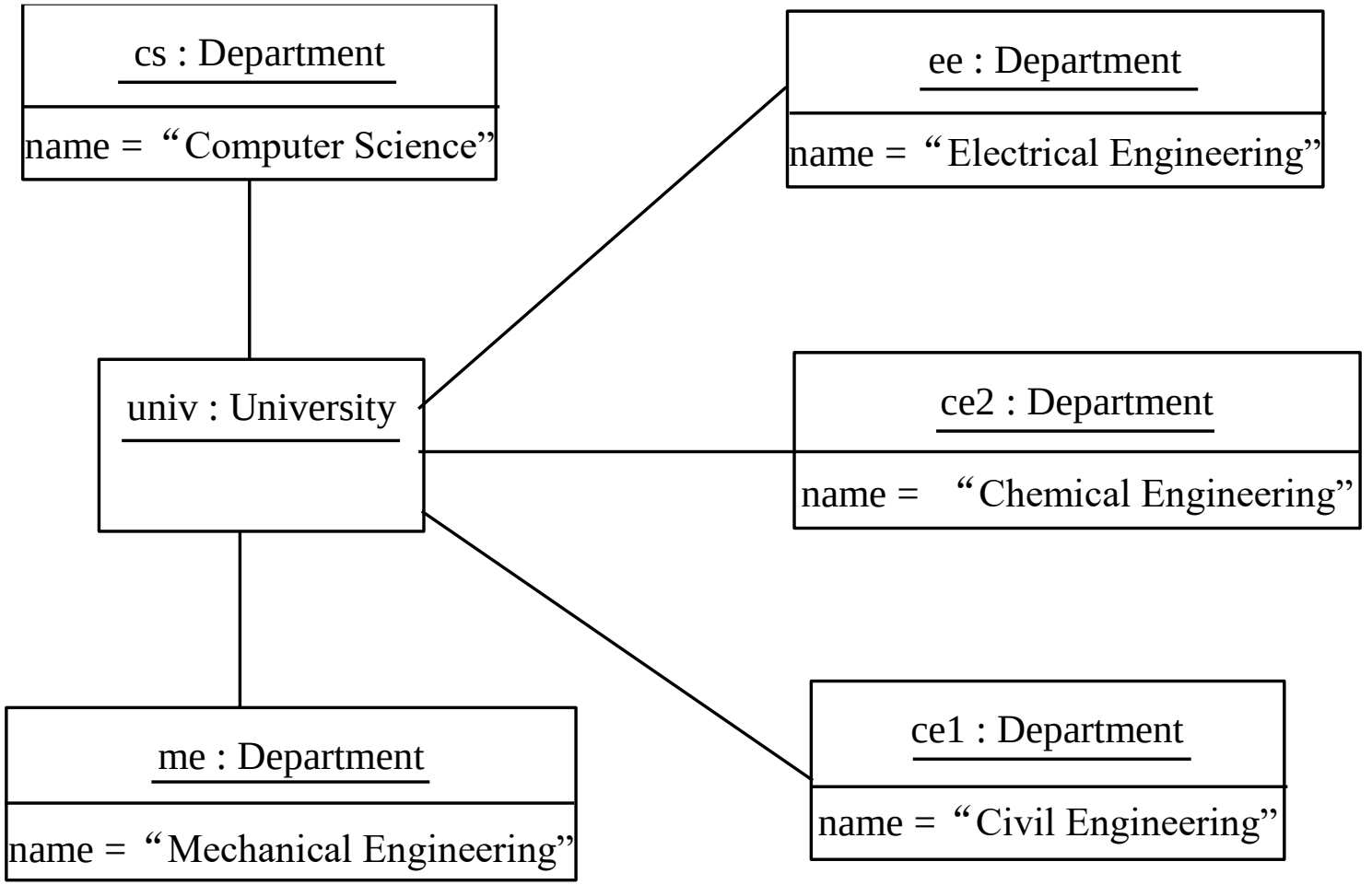


对象图

- 对象图（Object Diagrams）描述了某一瞬间对象集及对象间的关系。
- 为处在时域空间某一点的系统建模，描绘了系统的对象、对象的状态及对象间的关系。
- 对象图主要用来为对象结构建模。
- 对象图中通常含有：
  - ✓对象。
  - ✓连接。
  - ✓像其他的图一样，对象图中还可以有注解、约束、包或子系统。



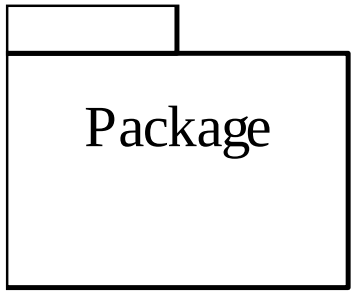
对象图的应用


cs : Department
name = “Computer Science”
ee : Department
name = “Electrical Engineering”
univ : University
ce2 : Department
name = “Chemical Engineering”
me : Department
name = “Mechanical Engineering”
ce1 : Department
name = “Civil Engineering”

7-3 包图
Package Diagram

包

- 包是一个用来将模型单元分组的通用机制。
- 包可以用在任何一个UML图中，但一般多用于用例图和类图，它就象文件夹一样，可以将模型元素分组隐藏，从而简化UML图，使得UML图更易理解。



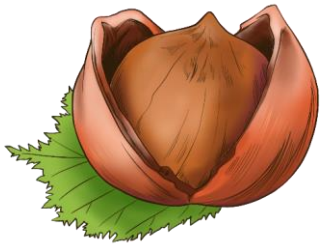
包图

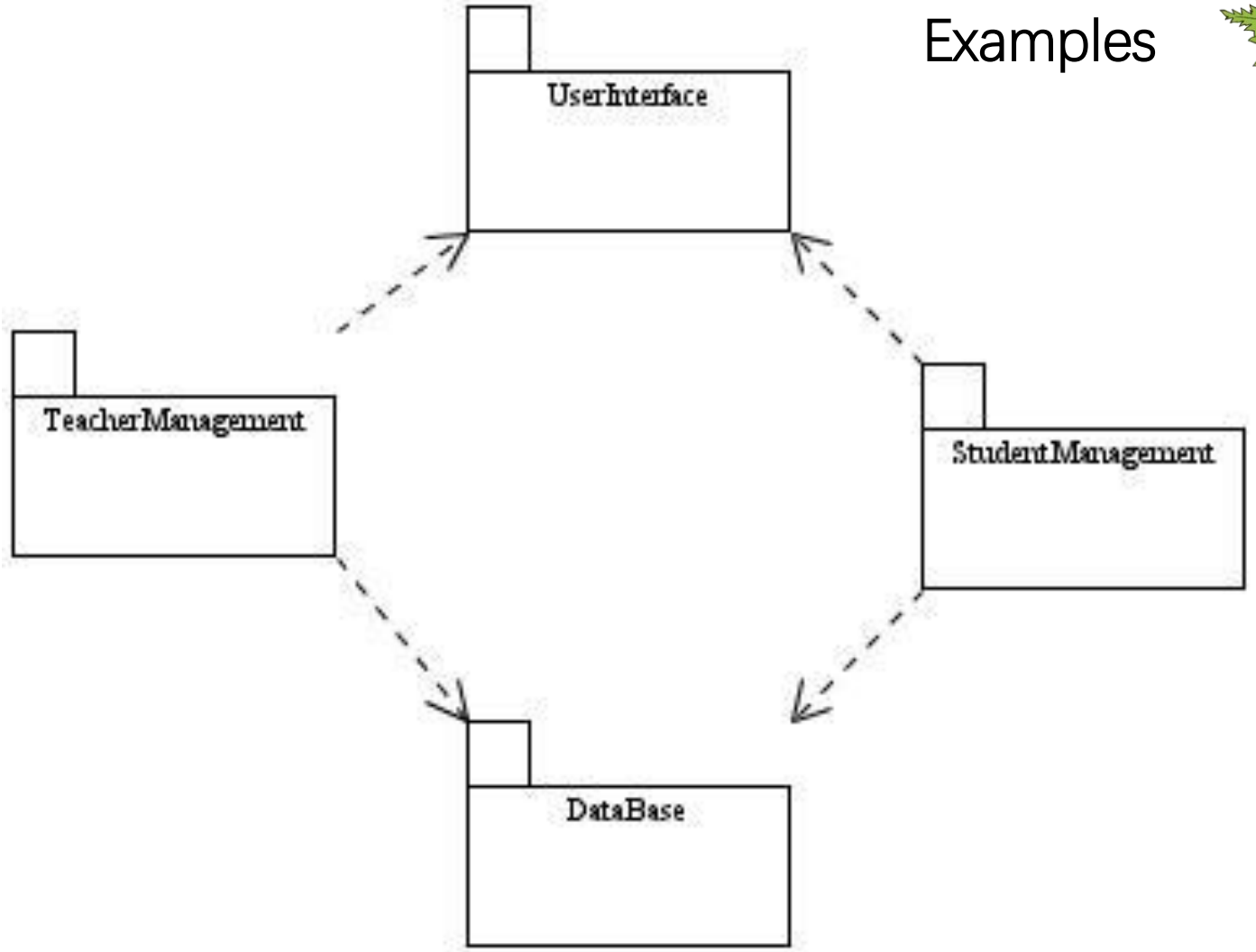
- 包图描述了包及包间的关系。包用来对建模元素进行分组，简化UML图从而使得UML图更易于理解。
- 在用包对类进行分组时，有3个经验法则可以遵循。
  - ✓ 将具有继承关系的类分到一个包里。
  - ✓ 将具有组合关系的类分到一个包里。
  - ✓ 将协作较多的类分到一个包里。类之间的协作多可以从顺序图或通信图中看出。

包图

举个例子

Examples

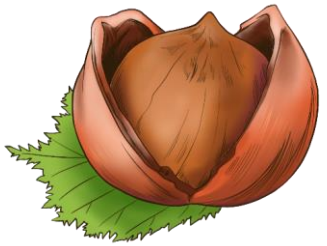


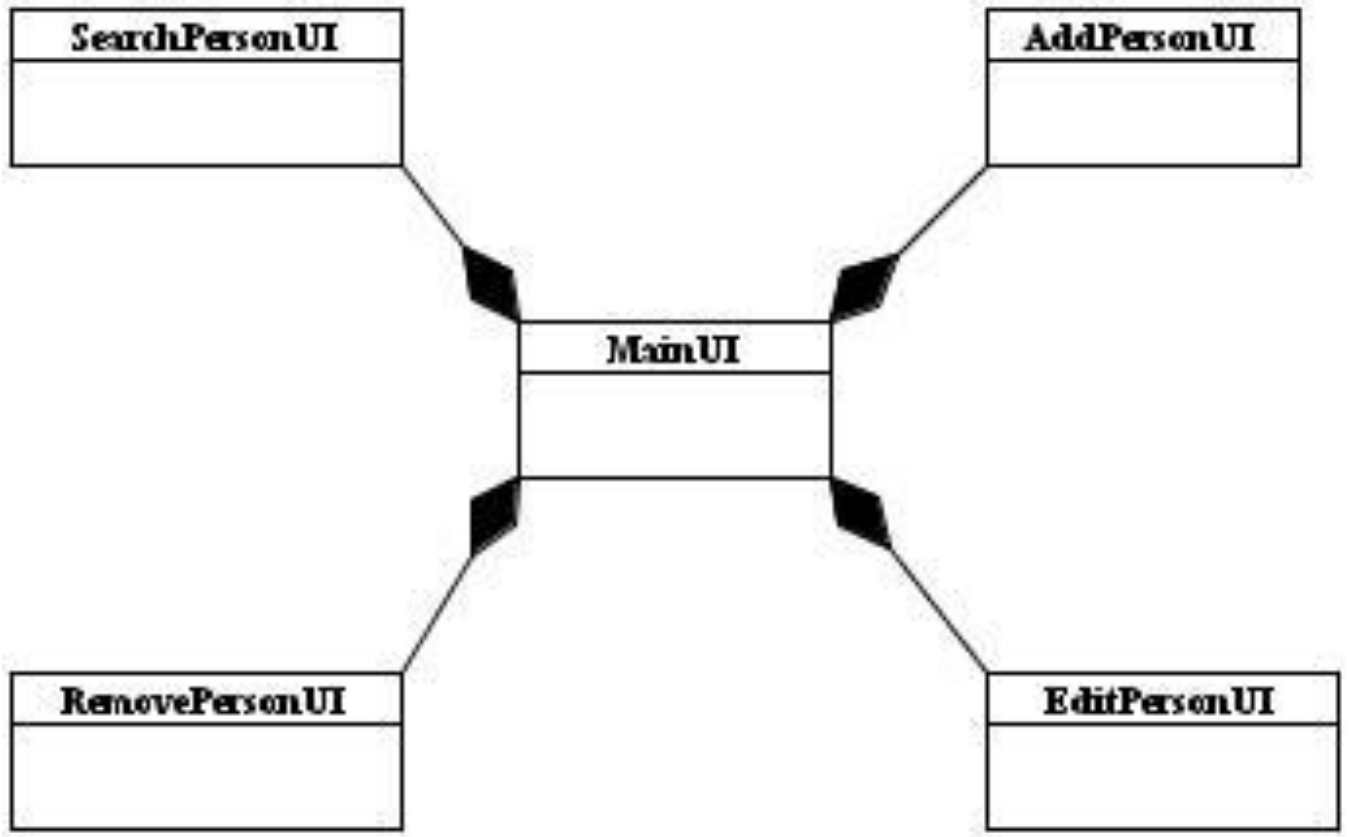


包图

举个例子

Examples





第11章 组件图与部署图

11-1 组件图
Component Diagram

组件

- 组件代表了一个接口定义良好的软件模块。
- 组件是系统的一个物理的、可替代的部分，它遵循接口定义，并为接口提供了实现。
- 组件的特点如下：
  - ✓组件是物理的。
  - ✓组件是可替代的。
  - ✓组件是系统的一部分。
- 组件的图形符号

组件与类

- 组件与类的区别:
  - ✓类代表了逻辑的抽象，而组件是物理的、可以存在于现实世界中的。也就是说，组件可以在节点上存在，而类不能。
  - ✓组件代表了其他逻辑单元的物理封装，与类的抽象存在于不同的层次上。
  - ✓类本身有属性和操作，但是，组件的操作通常只能通过接口来访问。

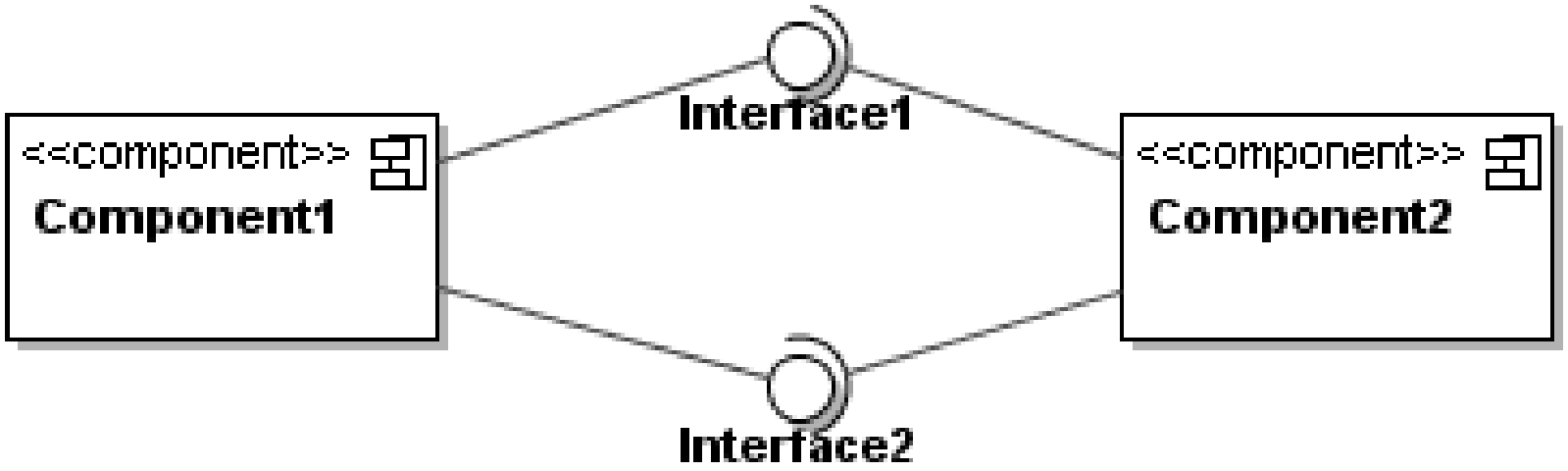
组件与接口

- 接口是操作的集合，定义了类或组件的服务。接口通常被用作粘合剂将组件连接在一起。
- 被一个组件实现的接口被称为该组件的输出接口（Export/provided Interface），也就是说，组件将该接口作为服务窗口向其他组件开放。一个组件可以有多个输出接口。“棒棒糖”
- 被一个组件使用的接口被称做该组件的引入接口（Import/required Interface）。“插座”

组件图

- 组件图描述了组件及组件间的关系，表示了组件之间的组织和依赖关系。
- 组件图是用来为面向对象系统的物理实现建模的两种图之一。
- 组件图包含下列元素：
  - 组件。
  - 接口。
  - 依赖关系、类属关系、关联关系和实现关系。
  - 如同其他的图，组件图中也可以有注释、约束、包或子系统。

组件图


Interface1
<<component>>
Component1
<<component>>
Component2
Interface2

11-2 组件图的应用

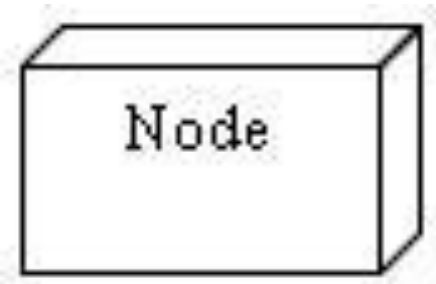
组件图的应用

- 组件图的应用
  - 为源代码建模
  - 为可执行版本建模
  - 为数据库建模
  - 为自适应系统建模

11-3 部署图
Deployment Diagram

节点

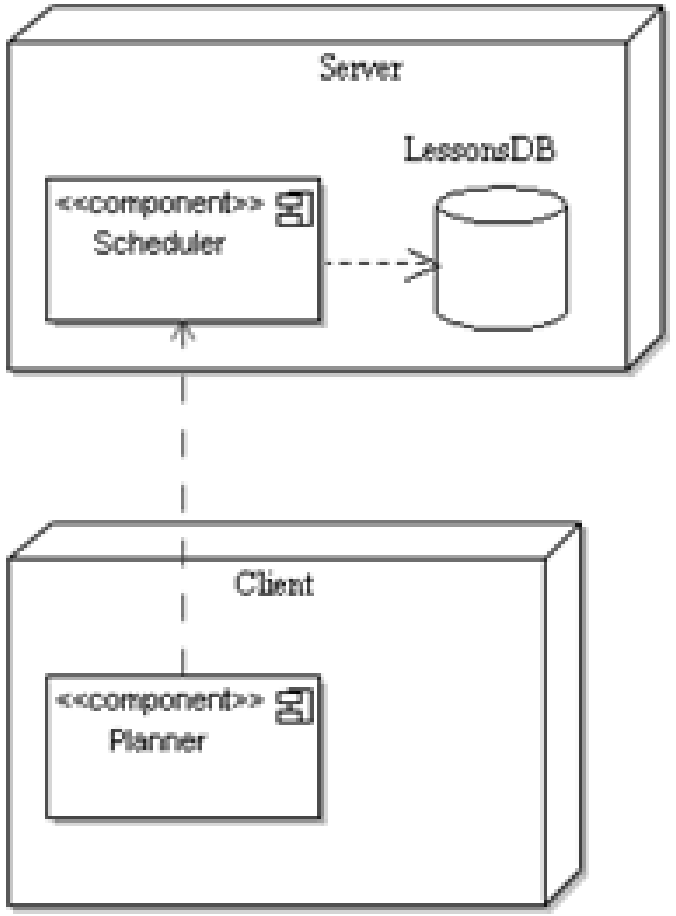
- 节点是运行时存在的物理单元，它代表了具有内存以及处理能力的计算资源。
- 节点与组件之间有许多重要的不同之处：
  - ✓组件参加系统的运行；节点是运行组件的硬件。
  - ✓组件代表了其他逻辑组件的物理封装；节点代表了组件的物理分布。
- 节点的UML符号



部署图

- 部署图描述了节点和运行其上的组件的配置。
- 部署图描述了运行系统的硬件拓扑，它为系统中物理节点、节点之间关系的静态方面建立了可视化的模型，并规定了构造的细节。
- 部署图含有：
  - ✓节点。
  - ✓依赖、关联关系。
  - ✓像其他的图一样，部署图中可以有注释、约束、包或子系统。

部署图

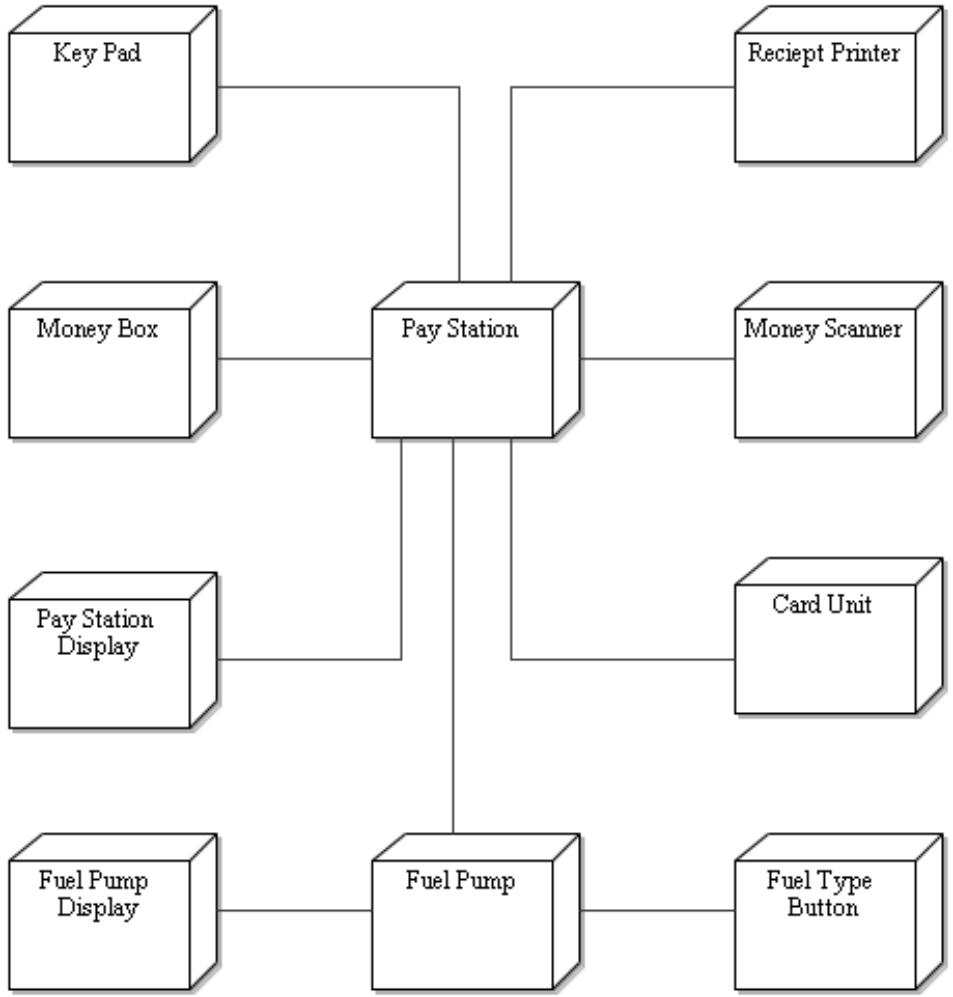

Server
LessonsDB
«component»
Scheduler
Client
«component»
Planner

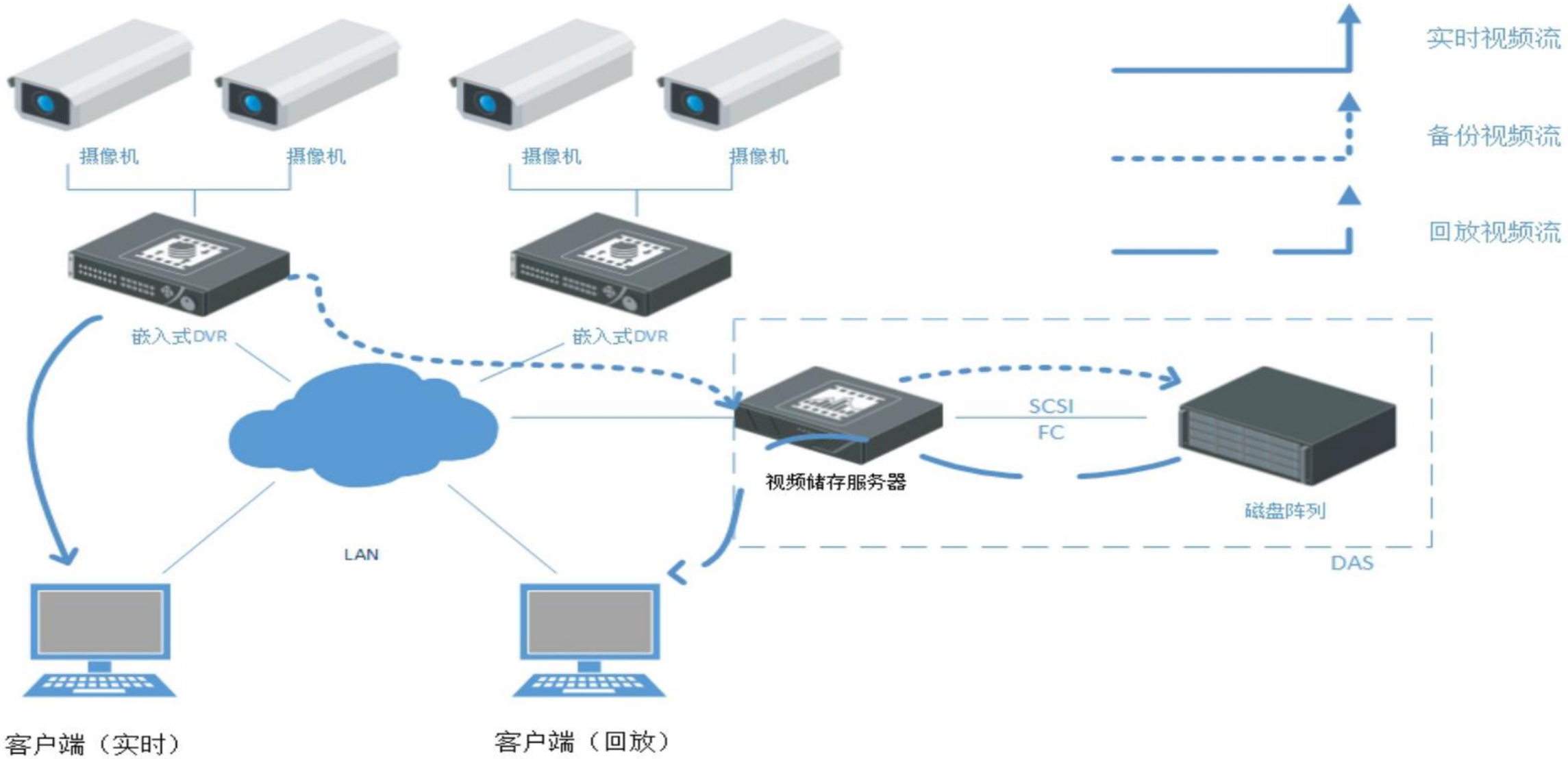
11-4 部署图的应用

部署图的应用

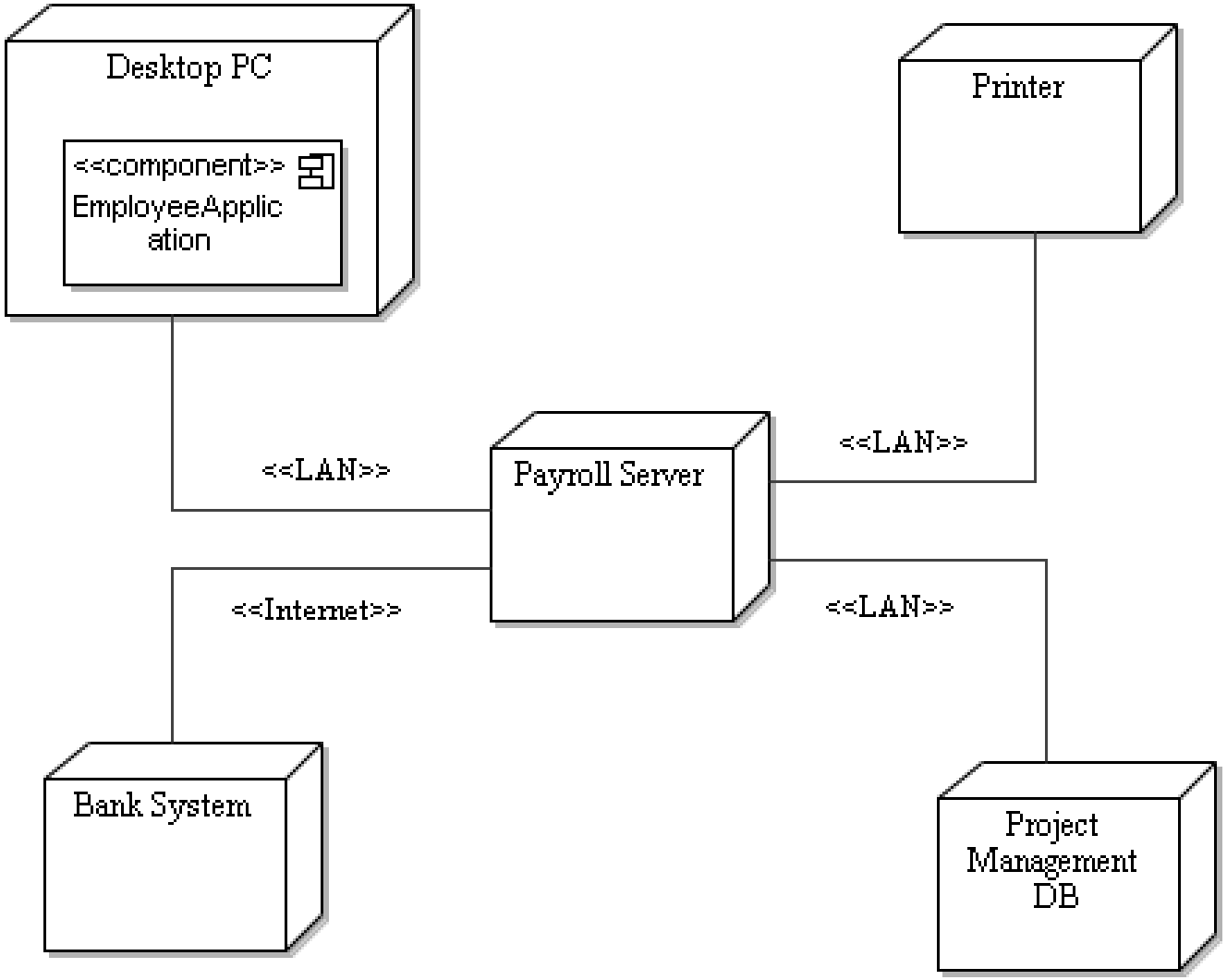
- 部署图的应用
  - **为嵌入式系统建模**
  - **为客户/服务器系统建模**
  - **为完全的分布式系统建模**

为嵌入式系统建模

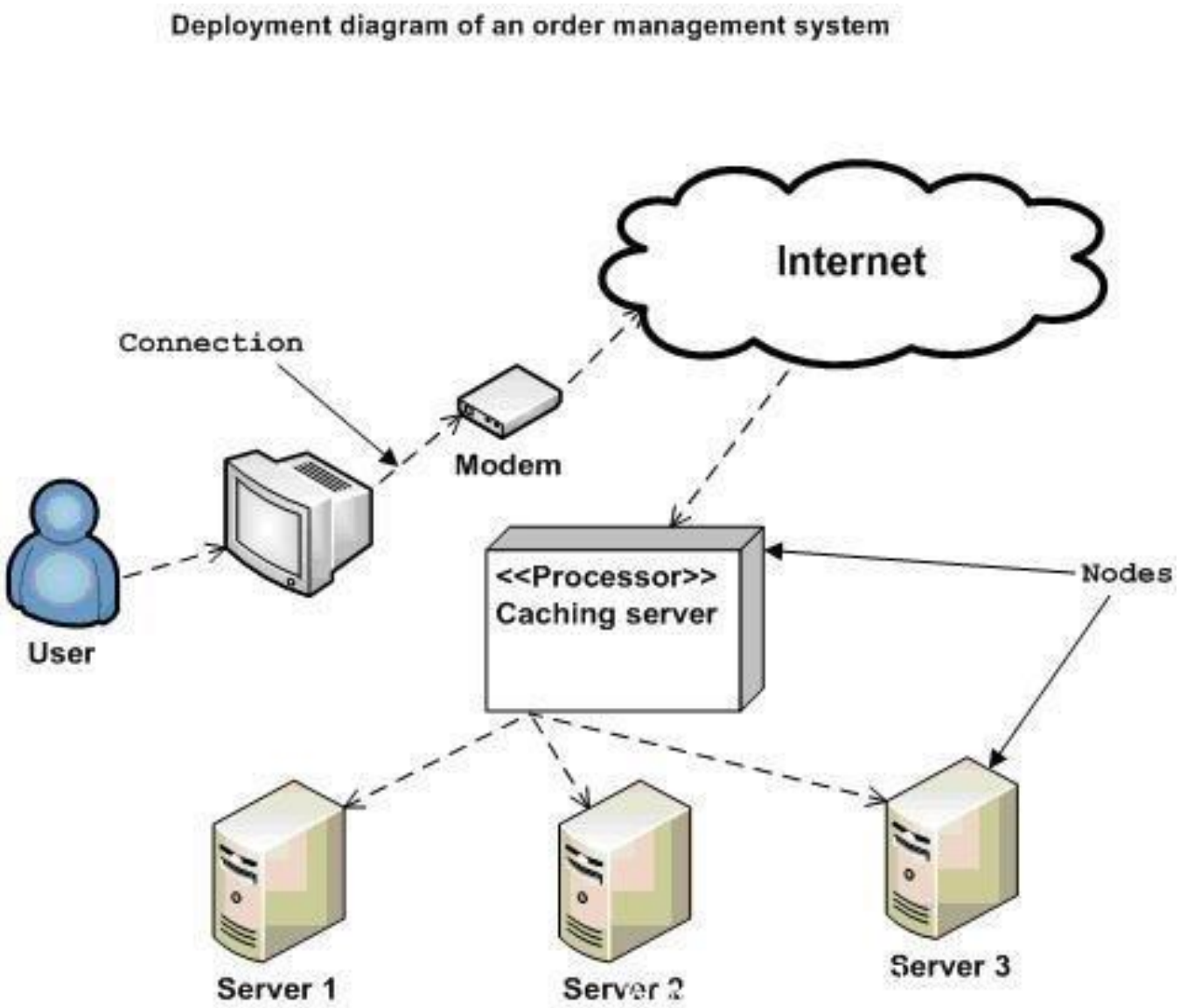

Key Pad
Reciept Printer
Money Box
Pay Station
Money Scanner
Pay Station Display
Card Unit
Fuel Pump Display
Fuel Pump
Fuel Type Button


实时视频流
备份视频流
回放视频流
摄像机
摄像机
摄像机
摄像机
嵌入式DVR
嵌入式DVR
SCSI
FC
视频储存服务器
磁盘阵列
DAS
LAN
客户端（实时）
客户端（回放）

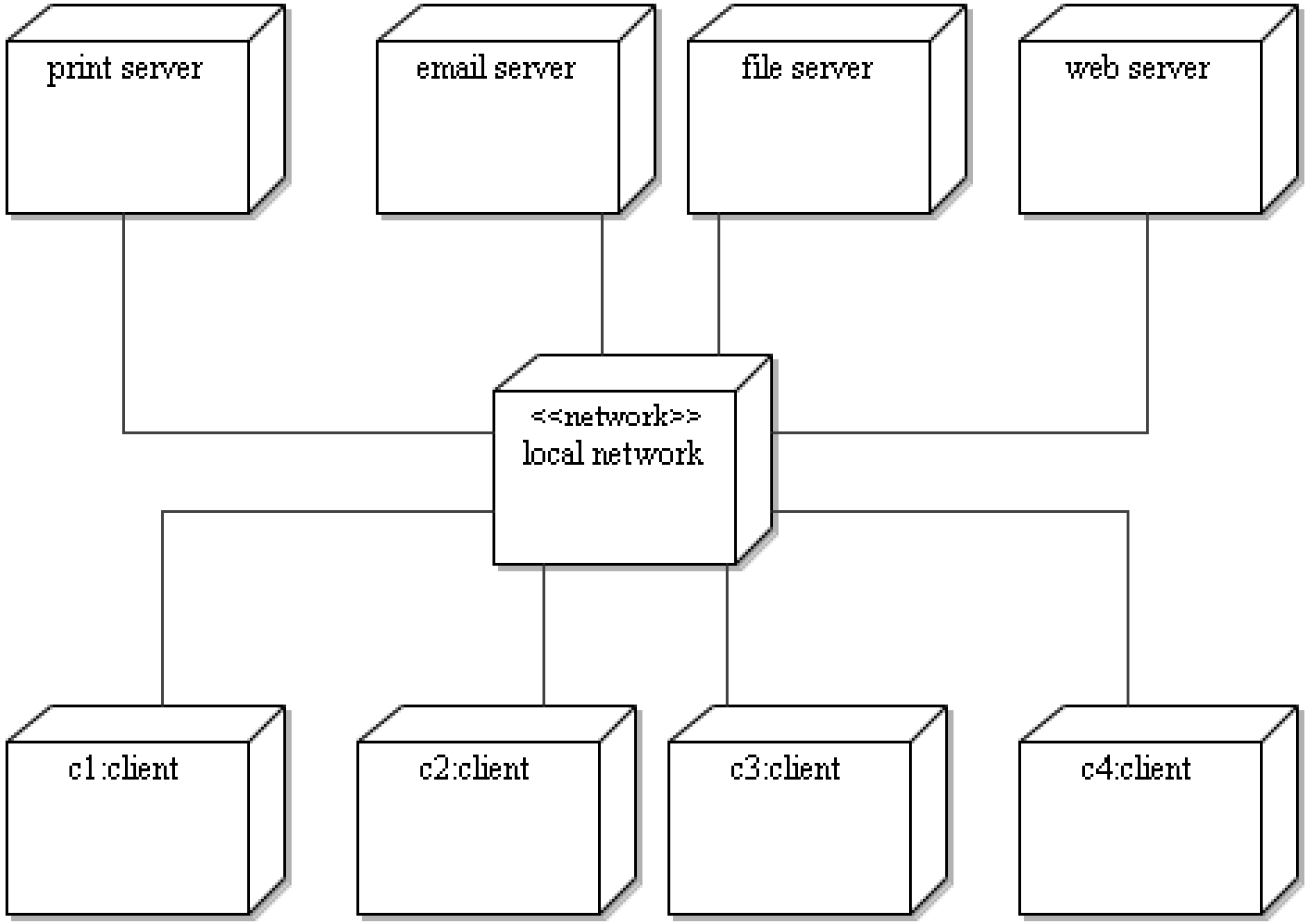
为客户/服务器系统建模


Desktop PC
<<component>>
EmployeeApplic
ation
Printer
<<LAN>>
Payroll Server
<<LAN>>
<<Internet>>
<<LAN>>
Bank System
Project
Management
DB

为客户/服务器系统建模


Deployment diagram of an order management system
Internet
Connection
Modem
User
<<Processor>>
Caching server
Nodes
Server 1
Server 2
Server 3

为完全的分布式系统建模



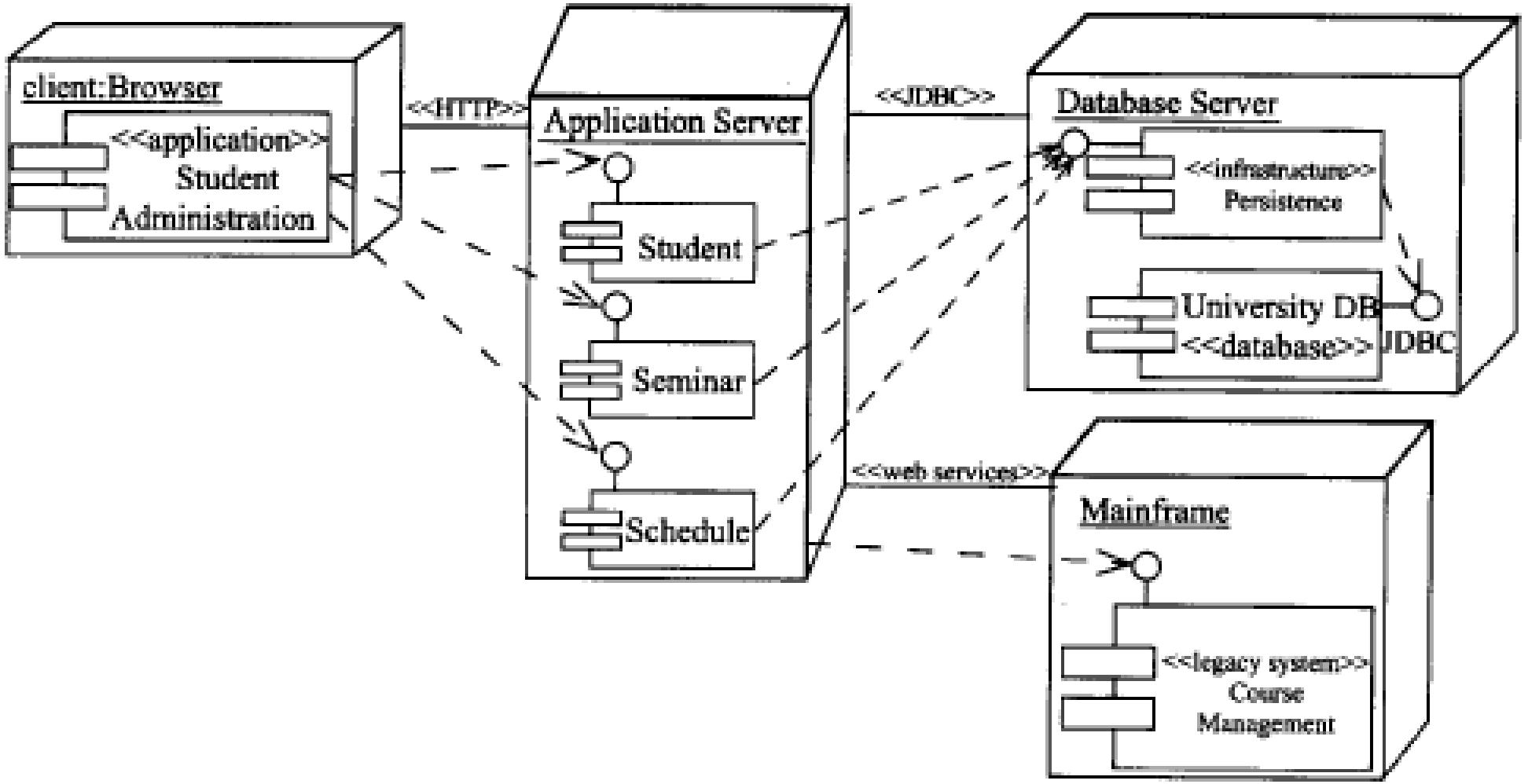

client:Browser
<<application>>
Student
Administration
<<HTTP>>
Application Server
Student
Seminar
Schedule
<<JDBC>>
Database Server
<<infrastructure>>
Persistence
University DB
<<database>>
JDBC
<<web services>>
Mainframe
<<legacy system>>
Course
Management

图 2-13　部署图示例