

# 使用UML开展面向对象的分析

## Chapter 5



# 基于用例的面向对象分析

- ❖ 定义核心业务用例
- ❖ 获取关键业务对象
- ❖ 分析核心业务用例
- ❖ 组织业务对象

# 基于用例的面向对象分析

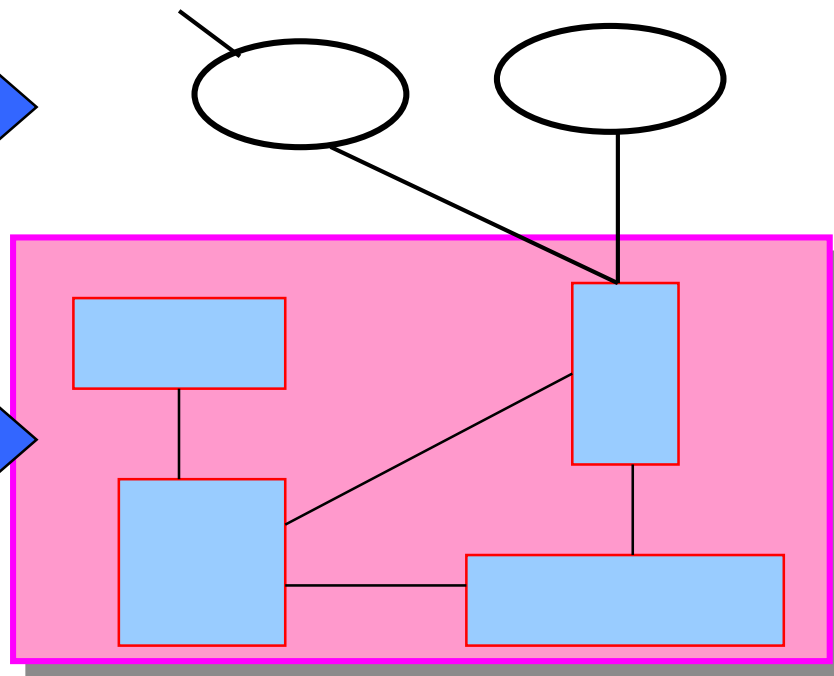
- ❖ 定义核心业务用例
- ❖ 获取关键业务对象
- ❖ 分析核心业务用例
- ❖ 组织业务对象

# 从用例开始分析

用例：外观



分析/设计：内部机理



# 从用例开始分析迭代

- ❖ 迭代开发是现代软件开发的主流，而迭代的基础就是用例
- ❖ 从用例开始分析基本思路
  - ◆ 用例分级：根据风险、重要性以及项目组的能力确定用例以及用例相关路径的优先级
- ❖ 早期迭代关注的重点（架构）
  - ◆ 核心业务的主要部分
  - ◆ 系统架构有重要影响
  - ◆ 影响系统性能等其他关键非功能需求的部分
  - ◆ 存在高风险的部分，如新技术、新产品

# 回顾：用例分级原则

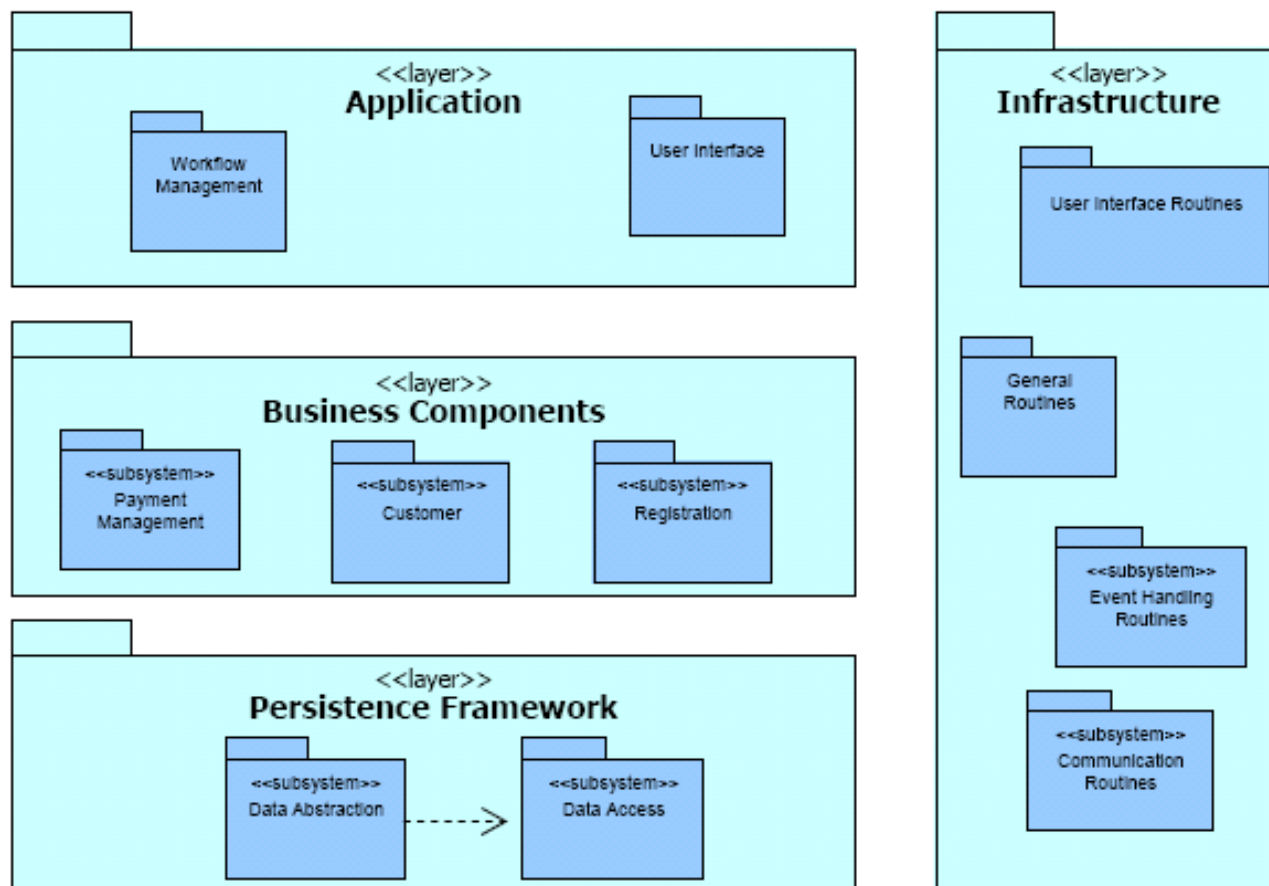
## ❖ 用例分级的一个基本原则

- ◆ 高级别用例是那些对系统核心架构影响最大的用例

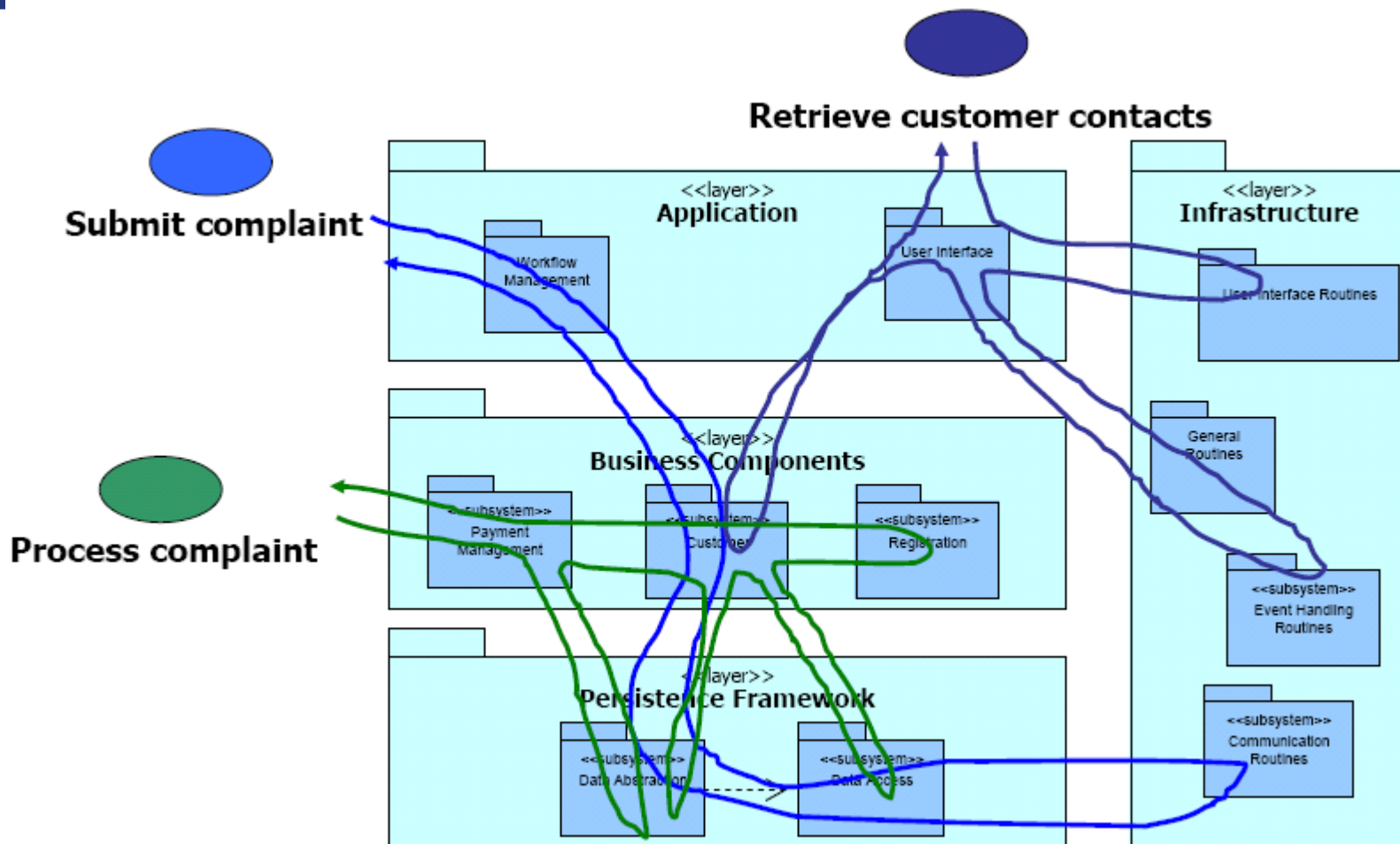
## ❖ 提高用例级别的特性：

- ◆ (1) 对架构设计有重要影响的用例，如在领域层中增加多个类的用例或者需要持久化的用例
- ◆ (2) 体现系统核心业务流程的用例
- ◆ (3) 存在开发风险的用例
- ◆ (4) 涉及新技术或需要创新的用例
- ◆ (5) 能够尽快投入使用并带来直接经济效益的用例

# 利用早期迭代建立软件架构



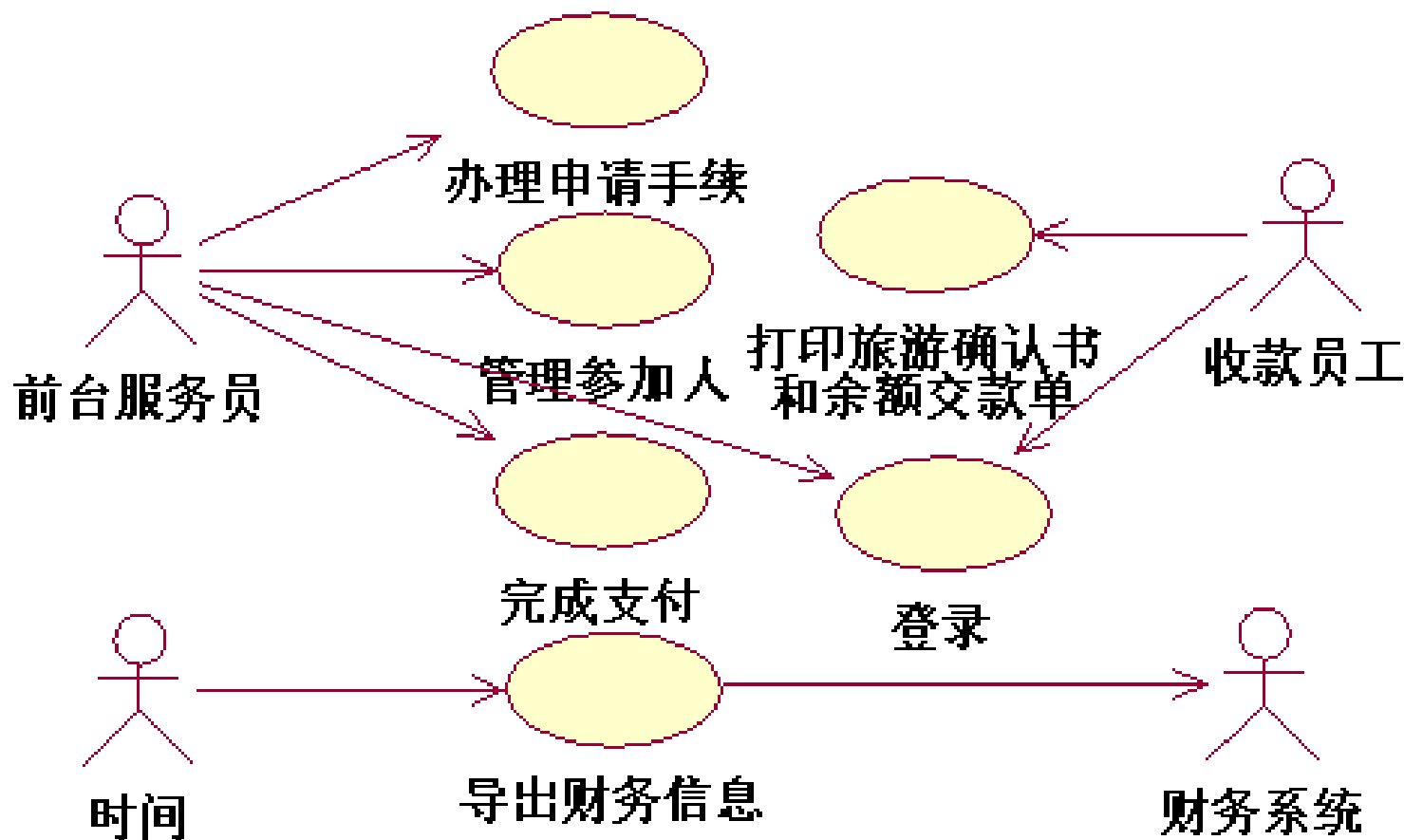
# 定义良好的迭代周期



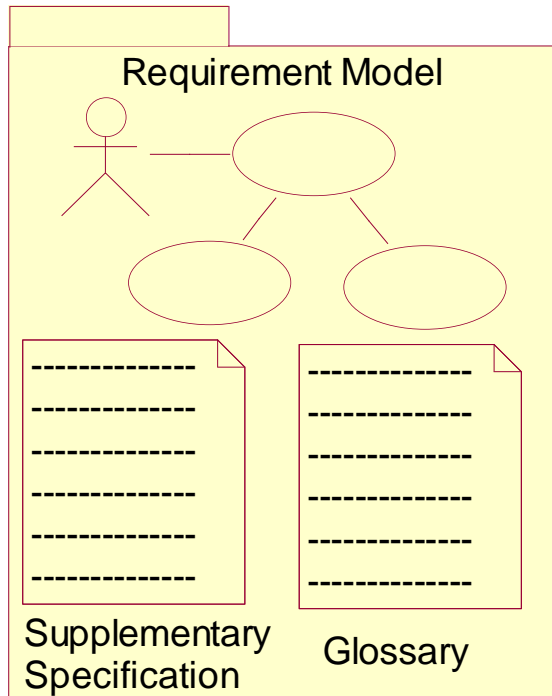
通过用例建模获取系统60%-80%的需求后，从中找出5%-10%重要的用例，来定义系统备选架构



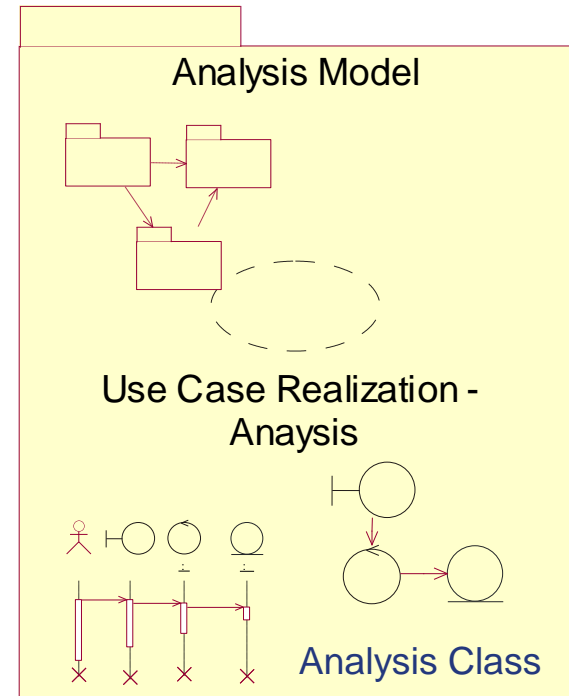
# 实例-旅游申请系统迭代1



# 从需求到分析



Analysis workflow

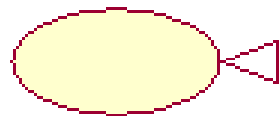


# 协作

- ❖ 协作 (Collaboration) 是设计(分析)模型中系统用例的表达式
  - ◆ 早期建模工具可以在用例上使用构造型 <<use-case realization>> (用例实现)
  - ◆ 将用例模型中的用例和设计(分析)模型中的类和关系连接在一起
  - ◆ 说明了每个用例必须用那些类来实现
- ❖ 用例实现提供了从分析和设计到需求的可跟踪性

# 创建协作

Use-Case Model



Use Case

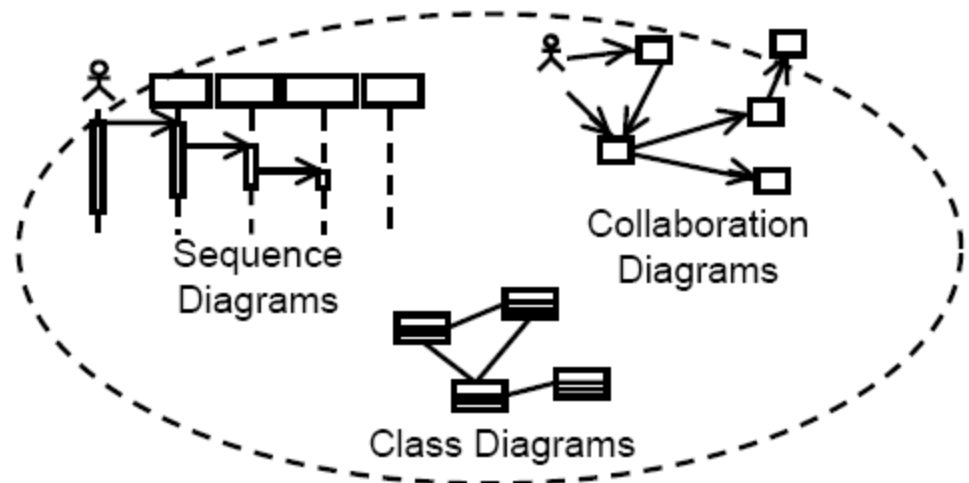
Analysis (Design) Model



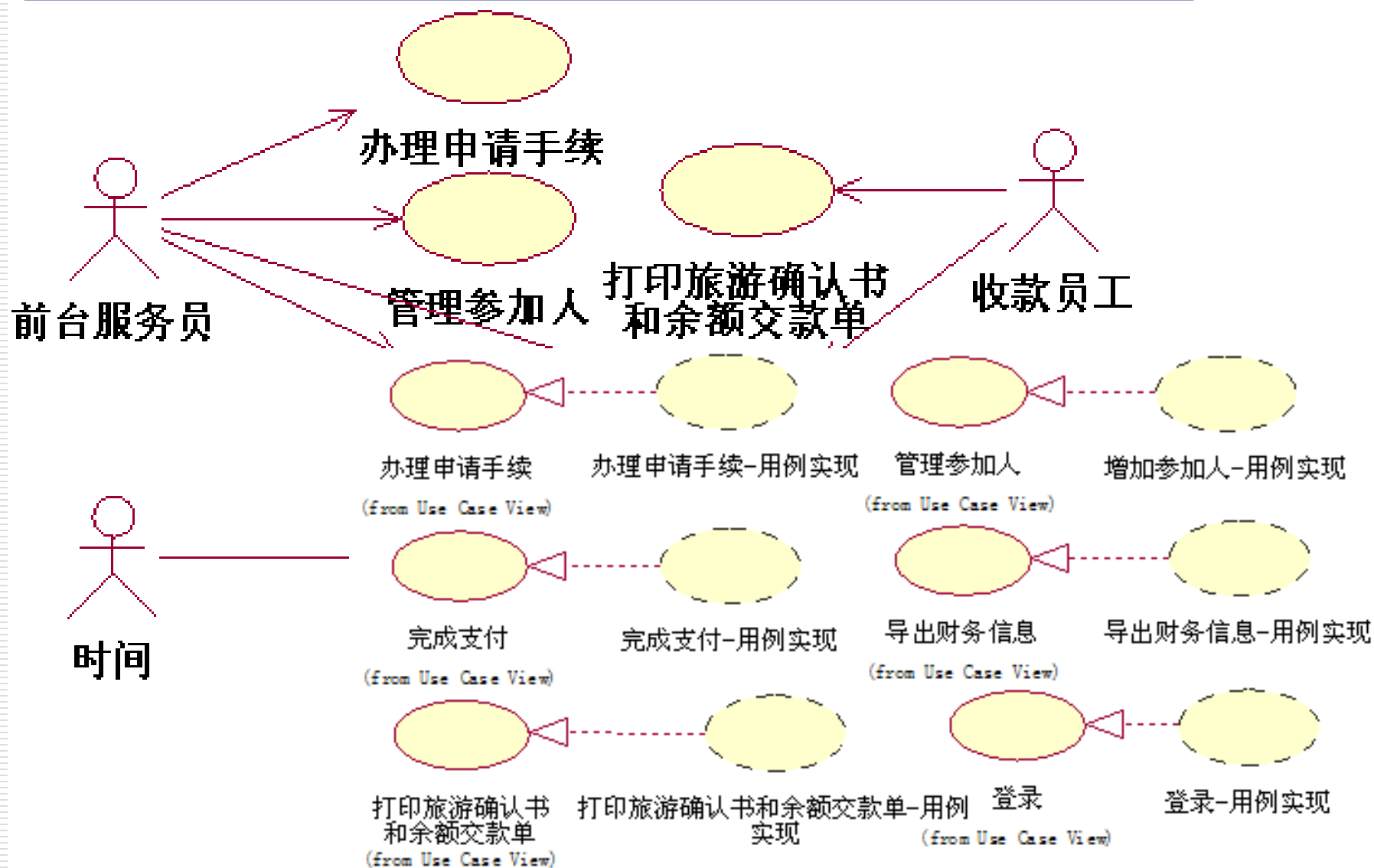
Use-Case Realization  
Collaboration



Use Case



# 实例-旅游申请系统迭代1



# 基于用例的面向对象分析

- ❖ 定义核心业务用例
- ❖ 获取关键业务对象
- ❖ 分析核心业务用例
- ❖ 组织业务对象

# 业务对象（领域对象）

## ❖ 业务对象（Business Object）

- ◆ The business object is **a key concept** what must be expressed in the system from the business /requirement
- ◆ 来自于业务（需求），在系统中必须揭示的核心概念（体现了系统所必须处理的业务数据结构和关系）
- ◆ 别名：实体类（Entity Class）、领域对象（Domain Object）、关键抽象（Key Abstraction）

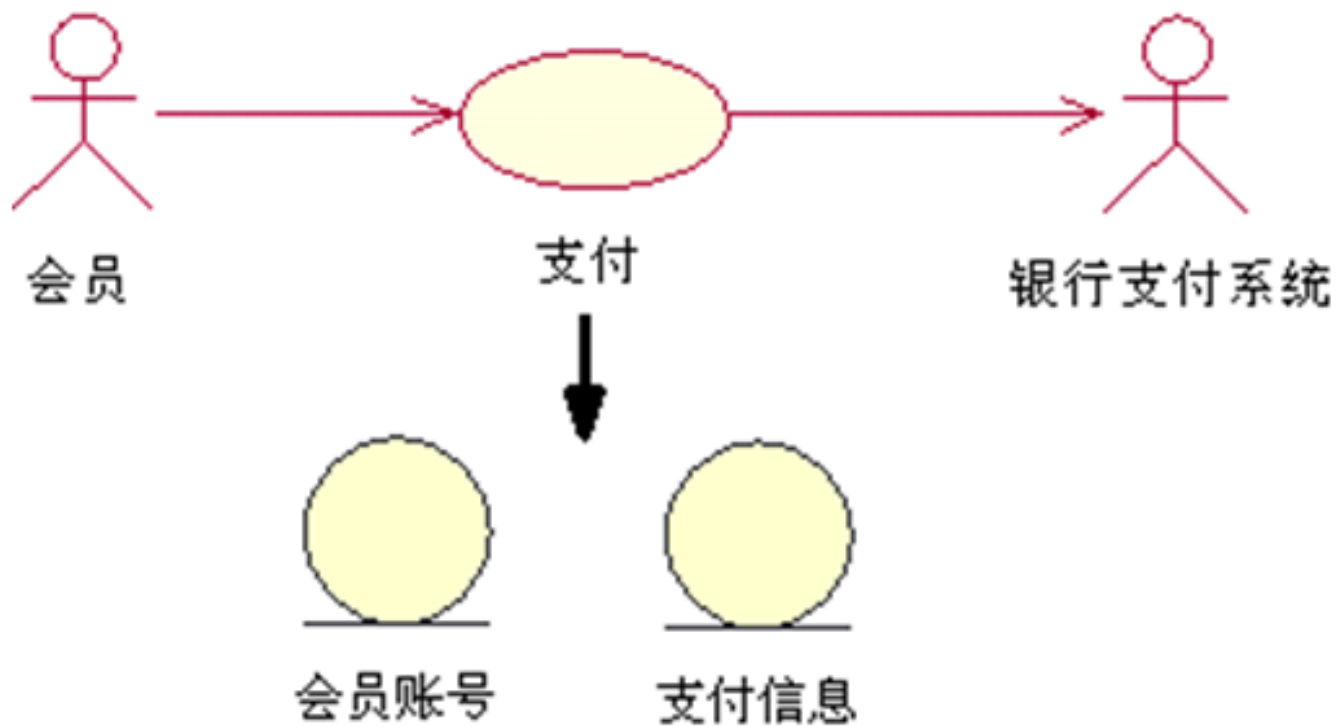
# 指南：名词筛选法识别业务对象

## ❖ 名词筛选法识别业务对象的基本思路：

- ◆ 将用例事件流作为输入，找出名词或名词性短语，形成了业务对象的初始候选列表
- ◆ 合并那些含义相同的名词
- ◆ 删除那些系统不需要处理的名词
- ◆ 删除作为参与者的名词
- ◆ 删除与实现相关的名词
- ◆ 删除那些作为其他业务对象属性的名词
- ◆ 对剩余的名词，综合考虑它在当前用例以及整个系统中的含义、作用以及职责，并基于此确定合适的名字，作为初始业务对象存在



# 示例：从用例中寻找业务对象



# 标注用例中的名字

<b>DESCRIPTION:</b>	This use case describes the <b>event</b> of a member submitting a new order for <b>SoundStage products</b> via the <b>world wide web</b> . The member selects the <b>items</b> they wish to purchase. Once they have completed their <b>shopping</b> , the member's <b>demographic information</b> as well as their <b>account standing</b> will be validated. Once the products are verified as being in <b>stock</b> , a <b>packing order</b> is sent to the distribution center for them to prepare the <b>shipment</b> . For any product not in stock, a <b>back order</b> is created. On completion, the member will be sent an <b>order confirmation</b> .	
<b>PRE-CONDITION:</b>	The <b>individual</b> submitting the order must be an active club member. The member must login in to the system (provide <b>identification</b> ) to enter an order.	
<b>TRIGGER:</b>	This use case is initiated when the member selects the <b>option</b> to enter a new order.	
<b>TYPICAL COURSE OF EVENTS:</b>	<b>Actor Action</b>	<b>System Response</b>
	<b>Step 1:</b> The member requests the option to enter a new order.	<b>Step 2:</b> The <b>system</b> responds by displaying the <b>catalogue</b> of the SoundStage products.
	<b>Step 3:</b> The Member browses the <b>available items</b> and selects the ones they wish to <b>purchase</b> along with the <b>quantity</b> .	<b>Step 4:</b> Once the member has completed their <b>selections</b> the system retrieves from <b>file</b> and presents the member's <b>demographic information</b> ( <b>shipping</b> and <b>billing addresses</b> ).
	<b>Step 5:</b> The member verifies demographic information (shipping and billing addresses). If no changes are necessary they respond accordingly (to continue).	<b>Step 6:</b> For each <b>product ordered</b> , the system verifies the <b>product availability</b> and determines an <b>expected ship date</b> , determines the <b>price to be charged</b> to the member, and determines the <b>cost of the total order</b> . If an item is not immediately available it indicates that the product is <b>backordered</b> or that it has not been released for shipping (for <b>pre-orders</b> ). If an item is no longer available that is indicated also. The system then displays a <b>summary of the order</b> to the member for verification.
	<b>Step 7:</b> The member verifies the order. If no changes are necessary they respond accordingly (to continue).	<b>Step 8:</b> The system checks the <b>status</b> of the <b>member's account</b> . If satisfactory, the system prompts the member to select the desired <b>payment option</b> (to be billed later or pay immediately with a <b>credit card</b> ).

# 候选对象清单

Accounts receivable	Member Account Standing
Actions	Member's Account Status
Active Member	New Order
Available Items	New Promotions
Back Order	Option
Back Ordered	Order Activity
Billing Addresses	Order Confirmation
Catalog	Order Total Cost
Club Member	Ordered Products
Company Performance	Packing Order
Credit Card	Payment Option
Credit Card Expiration Date	Preorders
Credit Card Number	Price to Be Charged
Credits	Problem
Customer Satisfaction	Procurement
Daily report	Product Availability
Demographic Information	Product Ordered
Distribution Center	Promotion
E-Mail	Purchase
E-Mail Addresses	Quantity
Event	Reason
Expected Ship Date	Requirement
External Receiver	Requirements Use Case
External Server	Sales Activity
File	Search Criteria
Hold Status	Selections
Identification	Shipment
In Stock	Shipping Address
Individual	Shopping
Inventory	SoundStage Products
Items	Summary of the Order
Main Page	System
Management	Warehouse
Marketing	World Wide Web
Member	

# 整理候选对象

Potential Object		Reason
New Order	✓	MEMBER ORDER
New Promotions	✓	PROMOTION
Option	✗	Potential Interface item to be addressed in object-oriented design
Order Activity	✗	Potential Interface item to be addressed in object-oriented design (report)
Order Confirmation	✗	Potential Interface item to be addressed in object-oriented design
Order Total Cost	✗	Attribute of MEMBER ORDER
Ordered Products	✓	MEMBER ORDERED PRODUCT
Packing Order	✗	Potential Interface item to be addressed in object-oriented design
Payment Option	✗	Attribute of MEMBER ORDER
Preorders	✓	Type of MEMBER ORDER
Price to Be Charged	✗	Attribute of MEMBER ORDERED PRODUCT
Problem	✗	Needs better focus – probably will be a comments attribute in MEMBER ORDER
Procurement	✗	Not relevant for current project
Product Availability	✗	Attribute of PRODUCT
Product Ordered	✗	Synonym of MEMBER ORDERED PRODUCT
Promotion	✓	PROMOTION
Purchase	✗	Synonym of MEMBER ORDER
Quantity	✗	Attribute of MEMBER ORDERED PRODUCT
Reason	✗	Needs better focus – probably will be a comments attribute in MEMBER ORDER
Requirement	✗	Not relevant for current project
Requirements Use Case	✗	Not relevant for current project
Sales Activity	✗	Potential Interface item to be addressed in object-oriented design (report)
Search Criteria	✗	Potential Interface item to be addressed in object-oriented design
Selections	✗	Synonym of MEMBER ORDERED PRODUCT
Shipment	✗	Not relevant for current project – responsibility of shipping and receiving
Shipping Address	✓	Type of ADDRESS
Shopping	✗	Potential Interface item to be addressed in object-oriented design
SoundStage Products	✗	Synonym of DISTRIBUTION CENTER
Summary of the Order	✗	Potential Interface item to be addressed in object-oriented design
System	✗	Not relevant for current project
Warehouse	✗	Synonym of DISTRIBUTION CENTER
World Wide Web	✗	Potential Interface item to be addressed in object-oriented design

# 建议对象列表

## Proposed Object List

ACTIVE MEMBER  
BILLING ADDRESS  
CLUB MEMBER  
CREDIT CARD ACCOUNT  
DISTRIBUTION CENTER  
EMAIL ADDRESS  
MEMBER  
MEMBER ORDER  
PROMOTION  
MEMBER ORDERED PRODUCT  
PRE-ORDER  
PROMOTION  
SHIPPING ADDRESS

*-PLUS-*

AGREEMENT  
AUDIO TITLE  
FORMER MEMBER  
GAME TITLE  
INACTIVE MEMBER  
MERCHANDISE  
RETURN  
TITLE  
TRANSACTION  
VIDEO TITLE

# 实例：名词筛选法找业务对象

## 基本事件流

1. 该用例起始于旅客需要办理申请手续
2. 前台服务员录入要申请的旅游团旅行路线代码和出发日期
3. 系统查询要申请的旅游团信息 (A-1)
4. 系统显示查询到的旅游团和相关路线信息 (D-1) (A-2、A-3)
5. 前台服务员录入本次申请信息 (D-2)
6. 系统计算并显示旅行费用的总额和申请订金金额
7. 申请责任人缴纳订金，前台服务员录入订金信息，提交本次申请信息
8. 系统保存该申请信息 (A-4)，用例结束

# 细化用例文档

- ❖ 需求阶段用例文档是从用户角度看待用户问题，侧重描述交互的1、4步
- ❖ 分析阶段则需要从系统角度看待用户问题，重点关注交互的2、3步：即系统如何响应用户请求；此时可以对需求阶段用例文档中系统的处理流程进一步细化
  - ◆ 获取理解系统必要的内部行为所需的其他信息：从“黑盒”到“白盒”的过渡
  - ◆ 有些情况下，可能找到一些不正确或不易理解的需求；此时，原始的用例事件流也需要进行更新

# 示例：细化用例文档



- 系统计算并显示旅行费用的总额和申请订金金额



- 系统根据旅游团价格和参加人情况计算费用总额以及订金金额， .....



# 实例：名词筛选法找业务对象（续）

## 基本事件流

1. 该用例起始于旅客需要办理申请手续
2. 前台服务员录入要申请的旅游团旅行路线代码和出发日期
3. 系统查询要申请的旅游团信息（A-1）
4. 系统显示查询到的旅游团和相关路线信息（D-1）（A-2、A-3）
5. 前台服务员录入本次申请信息（D-2）
6. 系统计算并显示旅行费用的总额和申请订金金额
7. 申请责任人缴纳订金，前台服务员录入订金信息，提交本次申请信息
8. 系统保存该申请信息（A-4），用例结束

# 更可行的方案

- ❖ 在实际应用中，依赖于类似项目的经验和对业务及系统的理解（或领域专家意见），来**从全局获取系统关键抽象概念**，作为初始的业务对象（实体类），再结合每个用例行为，辅以名词筛选法补充完善业务对象
- ❖ 此外，还有其他业务对象的来源
  - ◆ 系统原始需求书/问题描述
  - ◆ 该领域相关文献、专家意见或个人知识
  - ◆ 过去的类似系统

# 业务对象定义准则

- ❖ 综合考虑该业务对象在系统中的职责来定义业务对象
  - ◆ 系统是否需要处理该业务对象
  - ◆ 系统处理该业务对象时需要关注哪些行为
  - ◆ 业务对象应覆盖用例所约定的所有场景
- ❖ 使用该领域中最经常使用的名称为业务对象命名
  - ◆ 用户、开发方都认可的名称

# 为业务对象添加属性




- ❖ 属性(Attribute)是类的已命名属性, 用来存储对象的数据信息, 是没有职责的原子事物
  - ◆ 属性名是一个名词, 清楚地表达了属性保留的信息
  - ◆ 可以利用文字详细说明属性中将要存储的相关信息
  - ◆ 属性类型应来自业务领域, 与编程语言无关
- ❖ 从以下几个方面来定义属性:
  - ◆ 识别业务对象的过程中, 也可同时发现类的属性, 包括: 接在所有格后面的名词或形容词 (即某某的属性)、不能成为类的名词以及字段列表中所描述的数据需求
  - ◆ 作为一般业务常识, 是否有从类职责范围考虑所应包括的属性
  - ◆ 该业务领域的专家意见以及过去的类似系统

# 实例：为分析类添加属性

<<entity>>

路线






(from Entity Layer)

 代码  
 名称 : 字符串  
 天数 : 整数

<<entity>>

旅游团






(from Entity Layer)

 出发日期 : 日期  
 截止日期 : 日期  
 可申请人数 : 整数  
 大人价格 : 费用  
 小孩价格 : 费用

<<entity>>

申请





(from Entity Layer)

 申请编号  
 大人人数 : 整数  
 小孩人数 : 整数  
 申请状态  
 申请日期 : 日期

<<entity>>

支付明细








(from Entity Layer)

 金额 : 费用  
 截止日期 : 日期  
 支付日期 : 日期  
 支付状态

<<entity>>

参加人

(from Entity Layer)

 姓名 : 字符串  
 性别  
 出生日期 : 日期  
 电话号码 : 字符串  
 联系地址 : 字符串  
 邮政编码 : 字符串  
 email : 字符串

# 基于用例的面向对象分析

- ❖ 定义核心业务用例
- ❖ 获取关键业务对象
- ❖ 分析核心业务用例
- ❖ 组织业务对象

# 如何验证业务对象满足需求

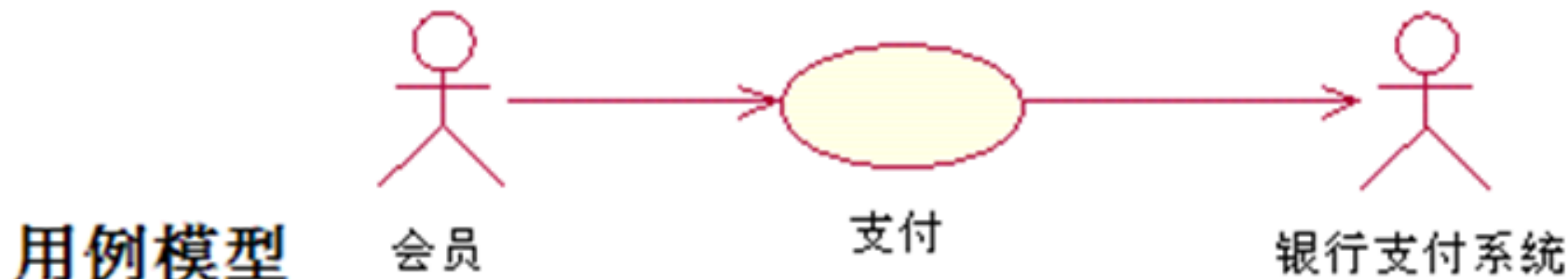
- ❖ 以用例为单位，分析用例行为是否可由业务对象满足
- ❖ 使用UML交互模型，围绕系统核心业务用例，开展用例行为分析
  - ◆ 顺序图 (Sequence Diagram)
  - ◆ 通信图 (Communication Diagram)
- ❖ 具体工作：
  - ◆ 将自然语言描述的用例场景，转换为UML顺序图

# 边界和控制对象

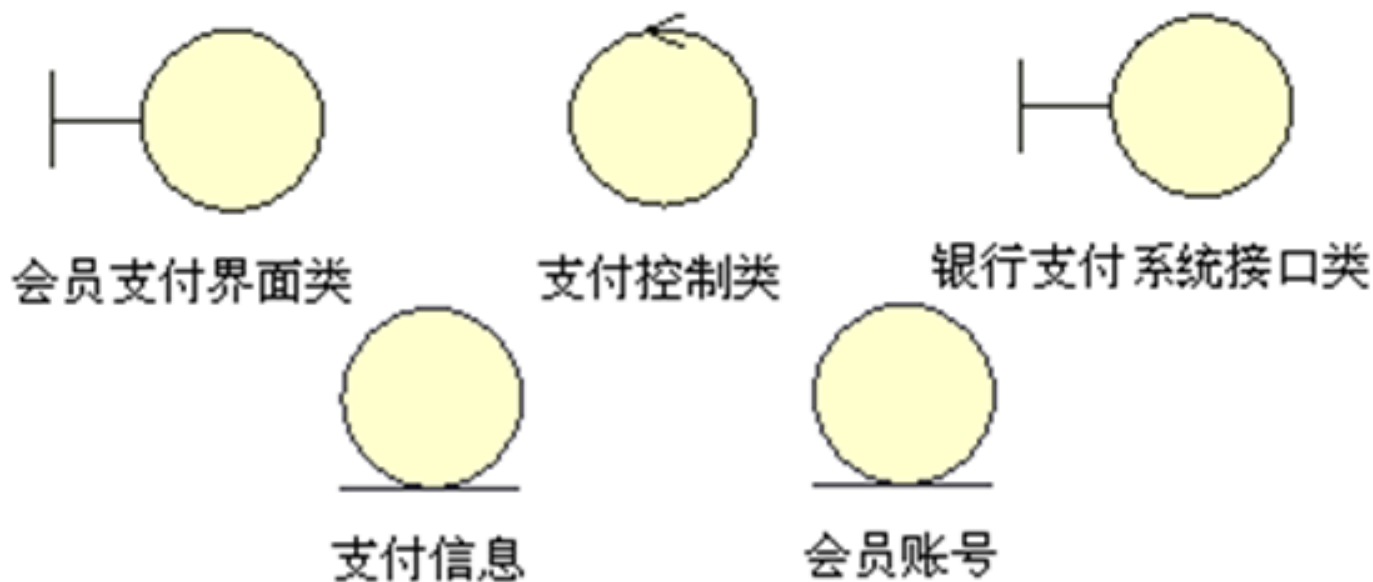
- ❖ 为了分析系统与用户交互的过程，处理基本的业务对象，还需要定义两类对象
  - ◆ 现阶段对象和类的概念混用，没有明确界定
- ❖ 边界对象（Boundary Object）
  - ◆ 表示参与者与用户的交互接口，包括用户界面和系统接口
  - ◆ 为每一对参与者和用户的交互定义一个边界对象
- ❖ 控制对象（Control Object）
  - ◆ 表示用例行为的控制流程（协调器）
  - ◆ 为每一个用例定义一个控制对象



# 示例：添加边界对象和控制对象



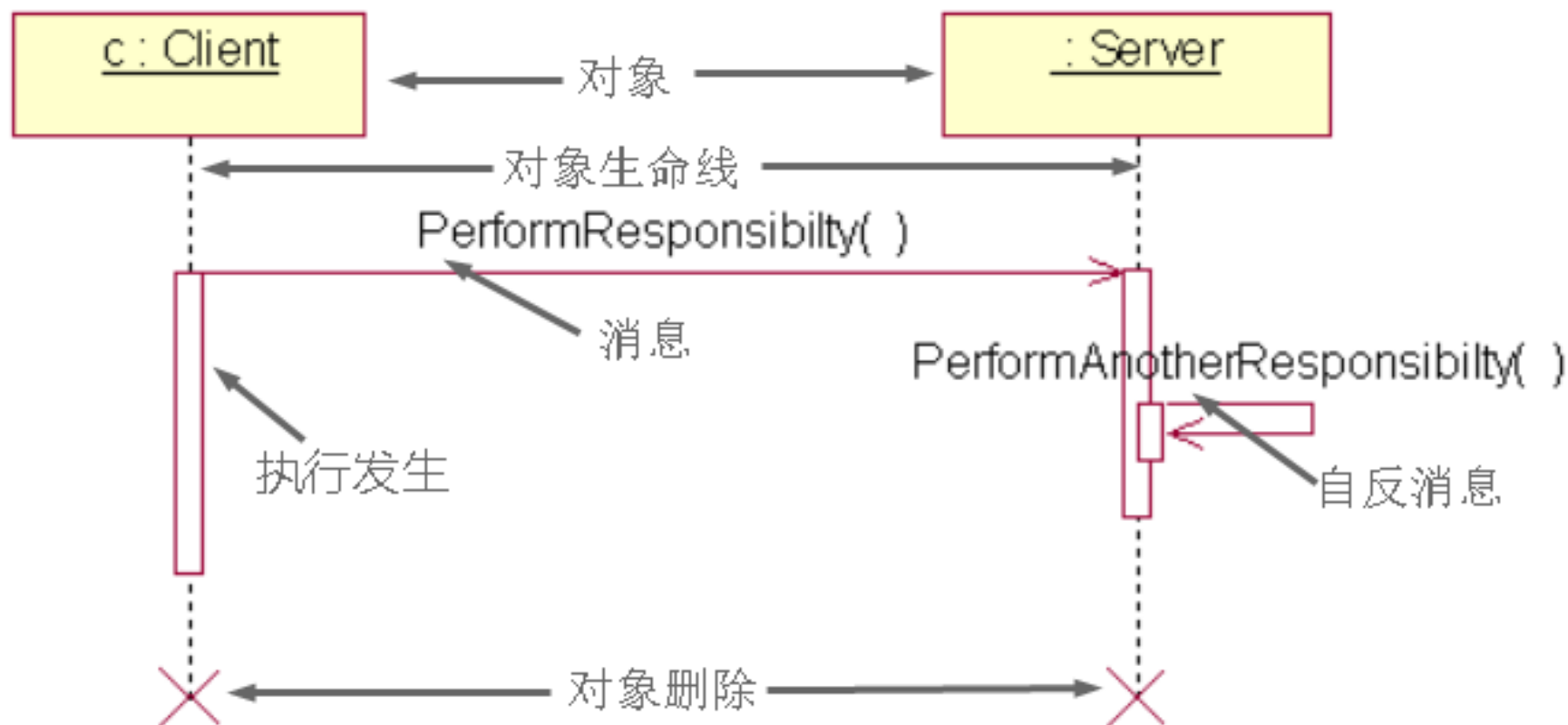
## 分析模型



# 顺序图Sequence Diagram

- ❖ 顺序图是一种交互图，描述对象之间的**动态交互**关系，着重体现对象间消息传递的时间顺序
  - ◆ 对象(Object)：对象、对象的生命线、对象的执行发生和对象的删除
  - ◆ 消息(Message)：简单消息、同步消息、异步消息、返回消息

# 顺序图剖析



# 利用顺序图分析用例行为

- ❖ 按照边界-控制-实体（业务对象）的方式绘制顺序图，并以控制类将控制逻辑隐藏起来
- ❖ 对象之间的消息传递以白话的方式进行说明，在信息长度不够时，可以加上UML的注释来做说明
- ❖ 分析阶段顺序图中的对象来自于前面初始化对象集合，在分析过程中可能发现新的业务对象，补充到业务对象集合中

# 如何进行消息分配

## ❖ 以不同的分析类作为分配标准

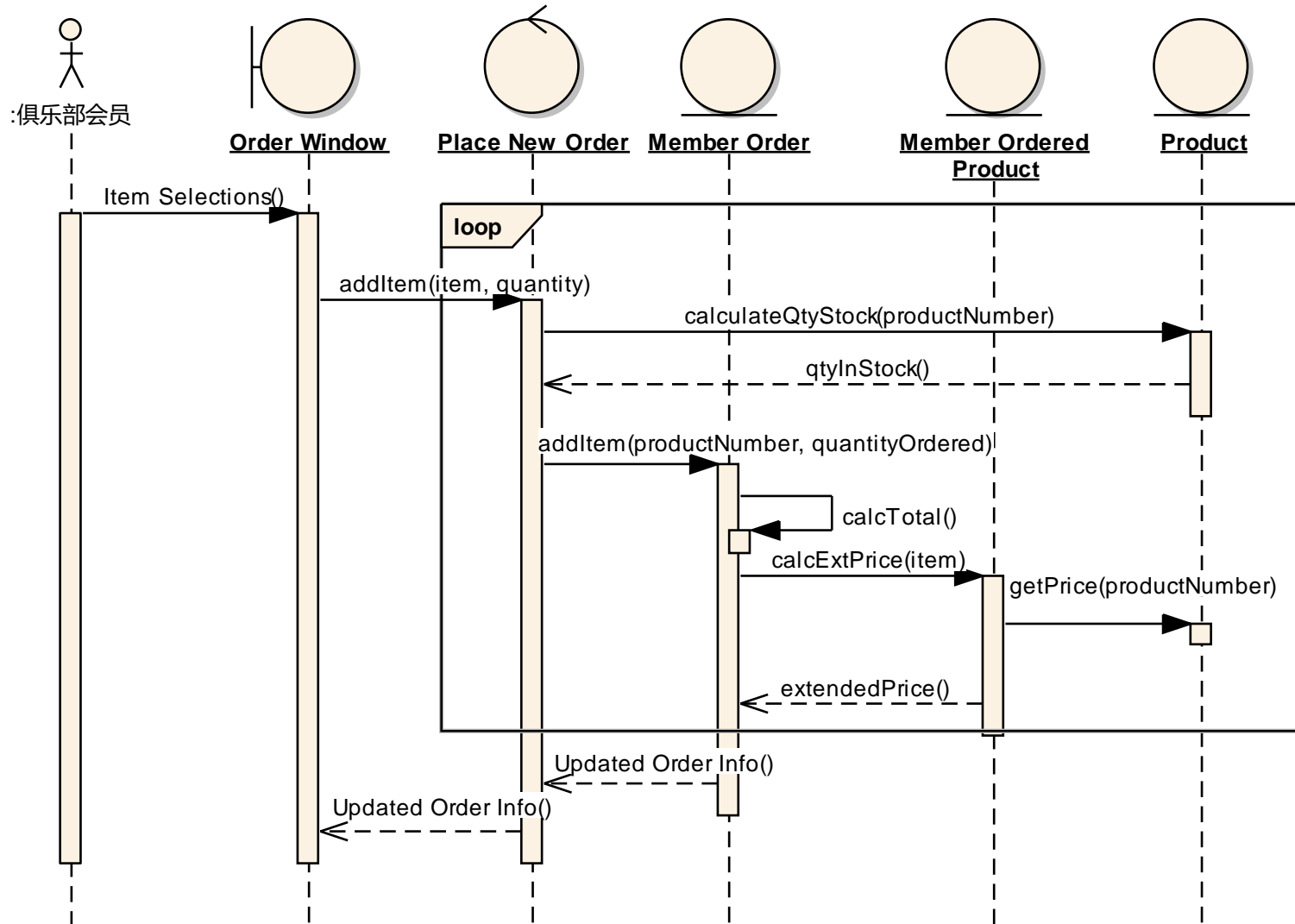
- ◆ 边界类：承担与参与者进行通信的职责
- ◆ 控制类：承担协调用例参与者与数据操作之间交互的职责
- ◆ 实体类（业务对象）：承担对被封装的内部数据进行操作的职责

# 如何进行消息分配（续）

- ❖ 专家模式：将职责分配给具有当前职责所需要的数据的类
  - ◆ 如果一个类有这个数据，就将职责分配给这个类
  - ◆ 如果多个类有这个数据
    - 将职责分配给其中的一个类，并对其它类增加一个关系
    - 将职责放在控制类中，并对需要该职责的类增加关系
    - 创建一个新类，将职责分配给该类，并对需要该职责的类增加关系

# 下订单的顺序图（基本场景）

sd Place New Order - Basic Scena...

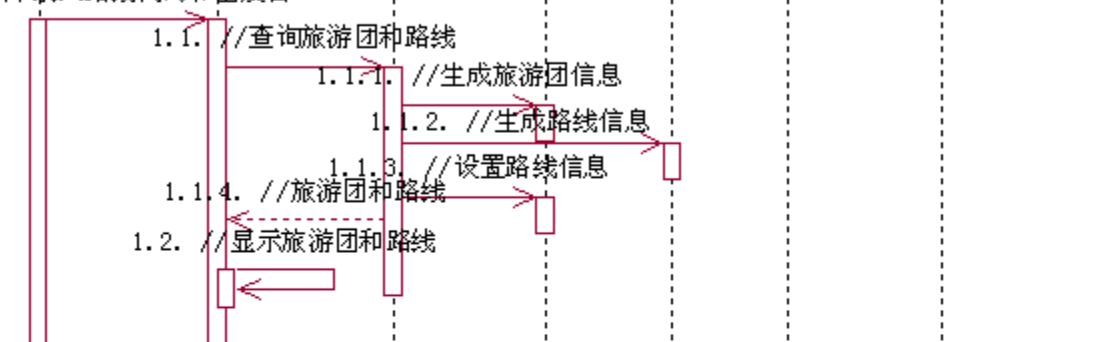


# 实例：进行职责分配

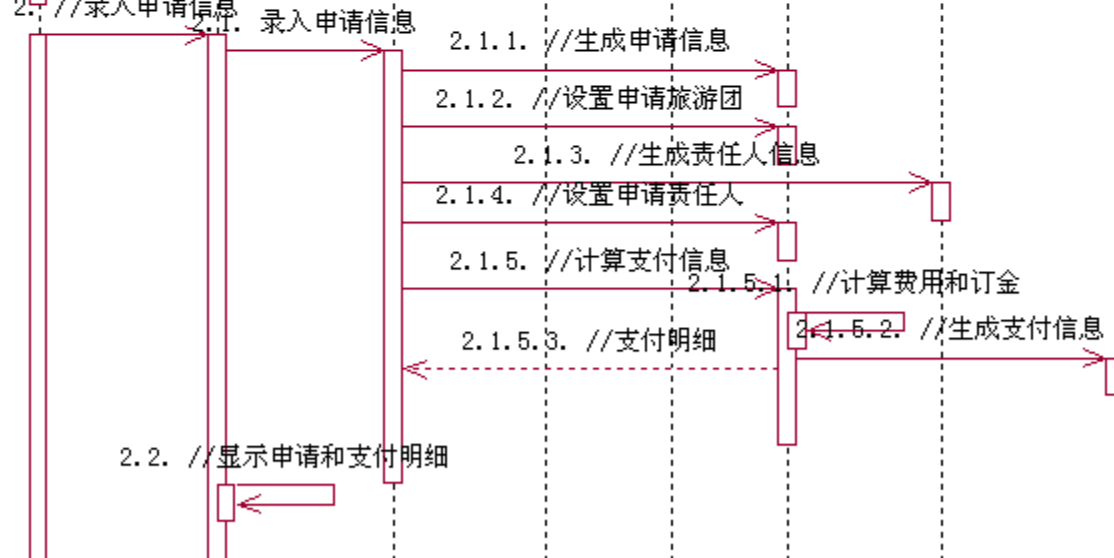


: 前台服务员 : 申请界面类 : 申请控制类 : 旅游团 : 路线 : 申请 责任人 : 参加人 : 支付明细

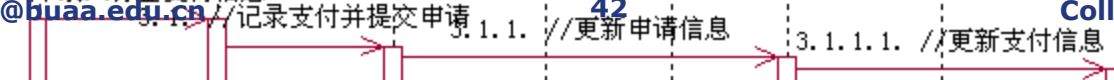
1. //录入路线代码和出发日



2. //录入申请信息



3. //录入订金支付信息





# 顺序图中的交互片段

- ❖ 顺序图主要用于描述顺序的执行流程，对于简单分支或循环，可直接描述
  - ◆ 执行的条件用[ ]括起来描述，表示条件为真时才执行，否则不执行
  - ◆ 循环条件要在条件前加上 “\*” 来描述，表示条件为真时重复执行
  - ◆ 其他约束用{ }括起来
- ❖ UML 2为顺序图提供了交互片段来描述分支、循环、并发等各种非顺序的情况

# 常见的交互片段

## ❖ 可选(opt)

- ◆ 该片段只有在守卫条件成立时才执行

## ❖ 选择(alt)

- ◆ 用水平虚线分割成几个分区。每个分区都有守卫条件，当守卫条件为真时执行

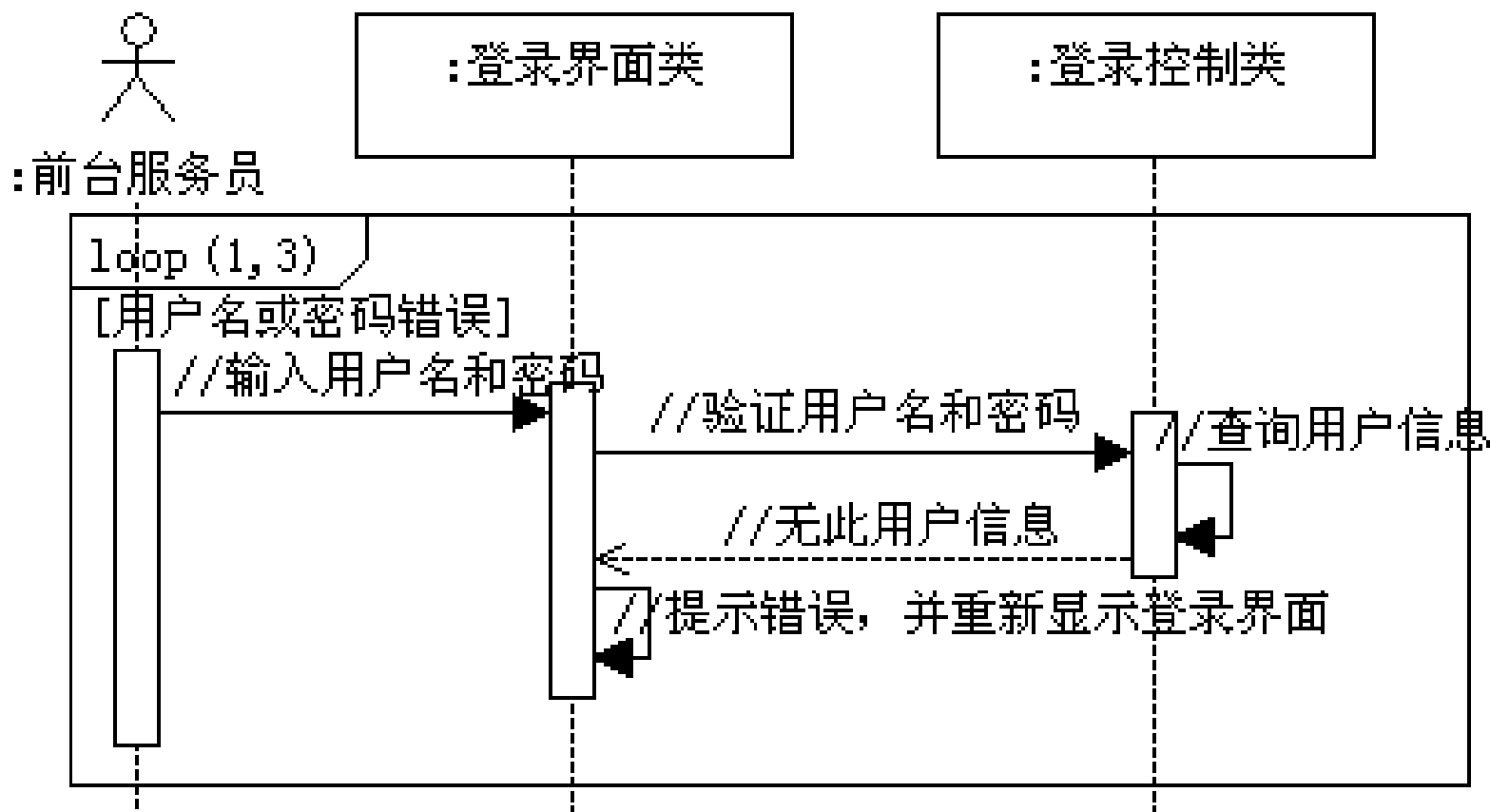
## ❖ 循环(loop)

- ◆ 在守卫条件为真的情况下循环执行

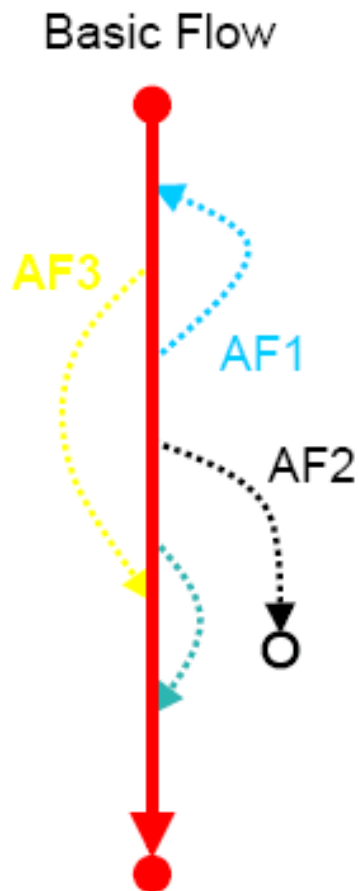
## ❖ 并行(par)

- ◆ 几个分区要并行（或并发）执行

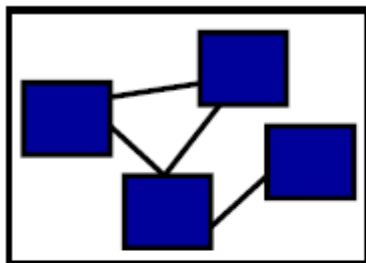
# 使用循环交互片段



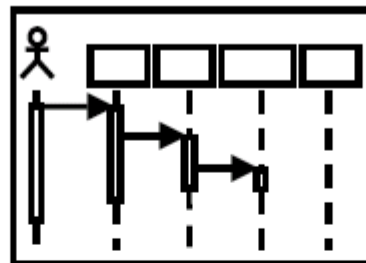
# 通过多个交互图分析用例



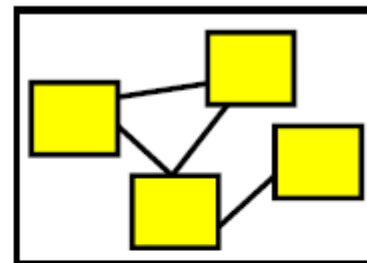
Alternate Flow 1



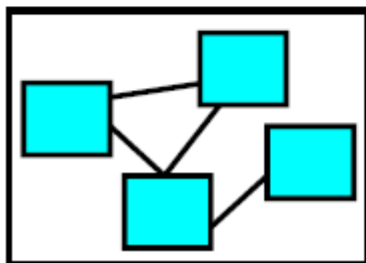
Alternate Flow 2



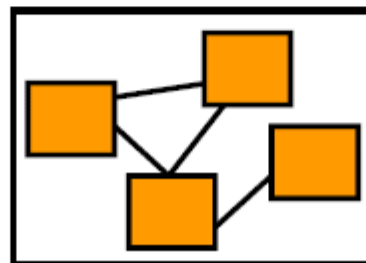
Alternate Flow 3



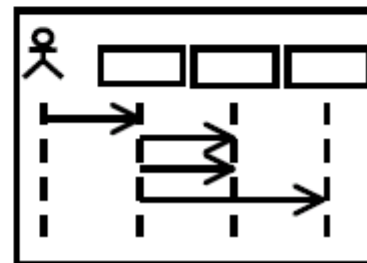
Alternate Flow 4



Alternate Flow 5



Alternate Flow n

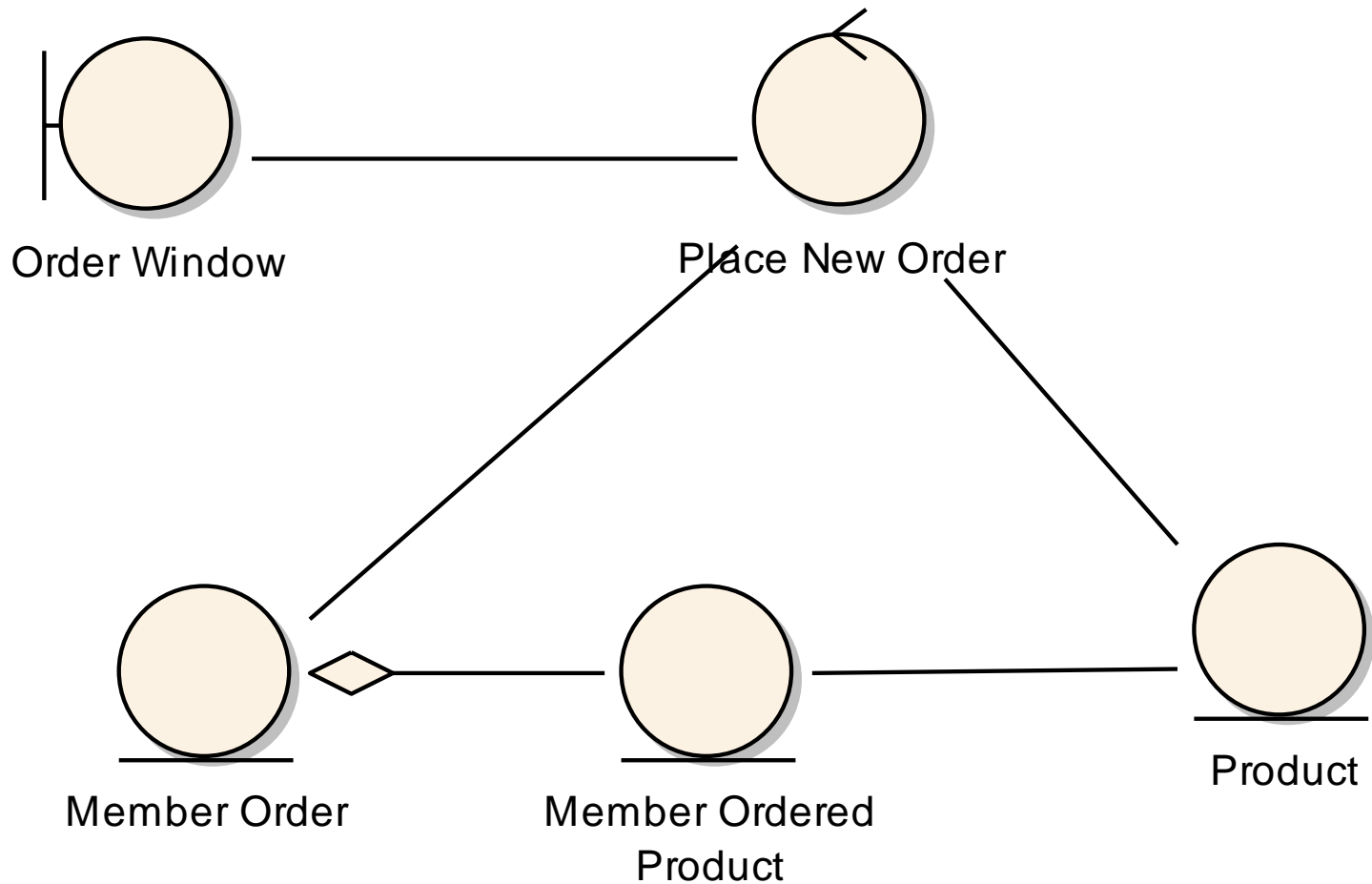


# 用例分析类图

- ❖ 对于每个用例行为分析可能都存在若干张交互图进行描述，而这些交互图中会使用到各种分析类的对象
- ❖ 对于每一个用例分析，需要绘制与之相关的类图，即VOPC图
  - ◆ 参与类类图(VOPC, View Of Participating Classes Class Diagram)
  - ◆ 类图中的元素来自于交互图中的对象
  - ◆ 类图中的关系来自于交互图中的消息(和业务对象模型)，分析阶段主要使用关联关系，也可根据业务模型引入泛化、聚合等关系

# 下订单类图 (VOPC)

class 下新订单



# 为各类分析对象添加行为

- ❖ 类的行为是要求某个类的对象所要履行的职责契约，在设计中将演化为类的操作(一个或多个)
- ❖ 获取类的职责
  - ◆ 从交互图中的消息
  - ◆ 从非功能需求中
- ❖ 分析阶段表示类的职责
  - ◆ “分析”操作，约定分析操作前加“//”
  - ◆ 文本描述

# 实例：利用分析操作表示职责

<<boundary>>

申请界面类

(from Boundary Layer)

- ◆//录入路线代码和出发日()
- ◆//显示旅游团和路线()
- ◆//录入申请信息()
- ◆//显示申请和支付明细()
- ◆//录入订金支付信息()

<<control>>

申请控制类

(from Control Layer)

- ◆//查询旅游团和路线()
- ◆//录入申请信息()
- ◆//记录支付并提交申请()

<<entity>>

申请

(from Entity Layer)

- ◆//设置申请旅游团()
- ◆//设置申请责任人()
- ◆//计算支付信息()
- ◆//计算费用和订金()
- ◆//更新申请信息()

<<entity>>

旅游团

(from Entity Layer)

- ◆//设置路线信息()

<<entity>>

支付明细

(from Entity Layer)

- ◆//更新支付信息()



# 利用文本方式描述职责

- ❖ 可以直接利用文字描述单个类的职责
- ❖ 传统的面向对象方法提供了一种“CRC卡”的技术可以更好地描述类的职责
- ❖ CRC卡由三部分组成，即类(Class)、职责(Responsibility)、协作(Collaborator)

类名	
职责1	职责1的协作
职责2	职责2的协作
...	...

# 基于用例的面向对象分析

- ❖ 定义核心业务用例
- ❖ 获取关键业务对象
- ❖ 分析核心业务用例
- ❖ 组织业务对象

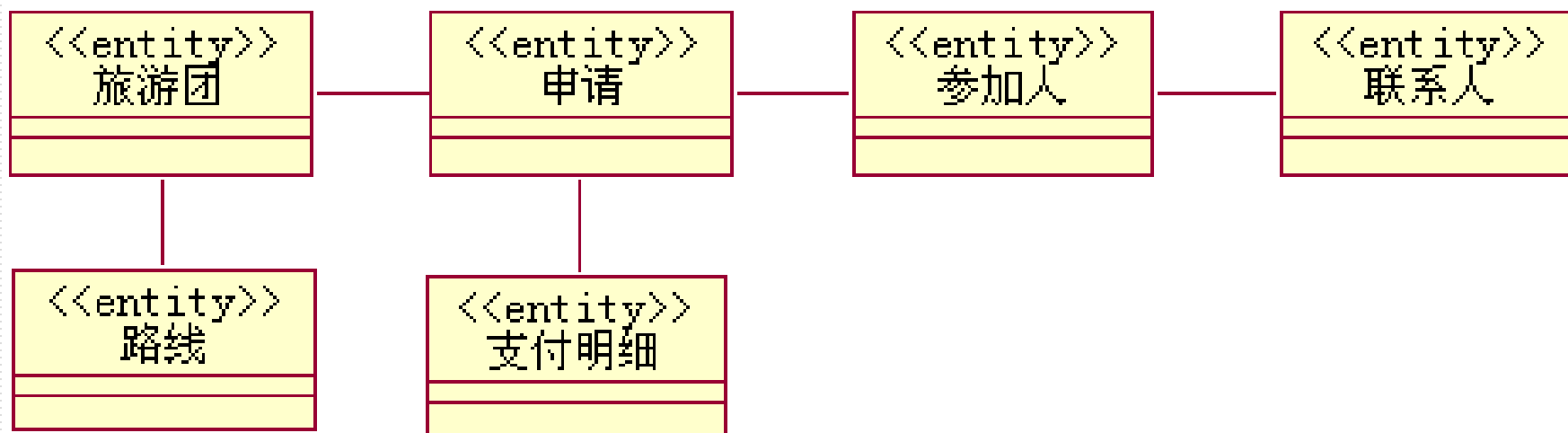
# 组织业务对象

- ❖ 对象不能孤立地存在，它们之间需要频繁地通过消息进行交互从而执行有用的工作，并达到用例的目标
- ❖ 为此，相应的类之间也应该存在特定的关系来支持这种交互过程，通过类关系组织业务对象
  - ◆ 1 关联关系：协作关系
  - ◆ 2 聚合关系与组合关系：整体-部分
  - ◆ 3 泛化关系：抽象-具体

# 1 关联关系

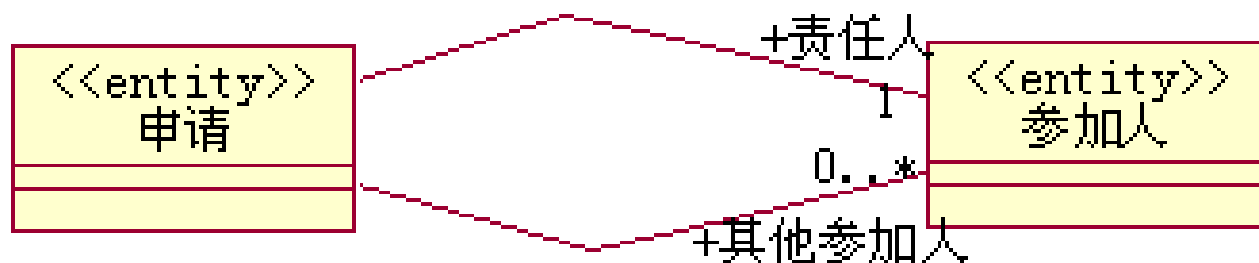
- ❖ 关联是类之间的一种结构化关系，是类之间的语义联系
  - ◆ 表明类的对象之间存在着链接
  - ◆ 对象是类的实例，而链接是关联的实例
- ❖ 识别关联的基本思路
  - ◆ 从交互模型中发现对象之间的链接，从而在相应的类上建立关联关系：如VOPC图中关联关系
  - ◆ 从业务领域出发，分析领域中所存在的业务对象之间的语义联系，为那些存在语义联系的类之间建立关联关系：如业务对象之间的各种语义联系

# 实例：业务对象之间的关联关系



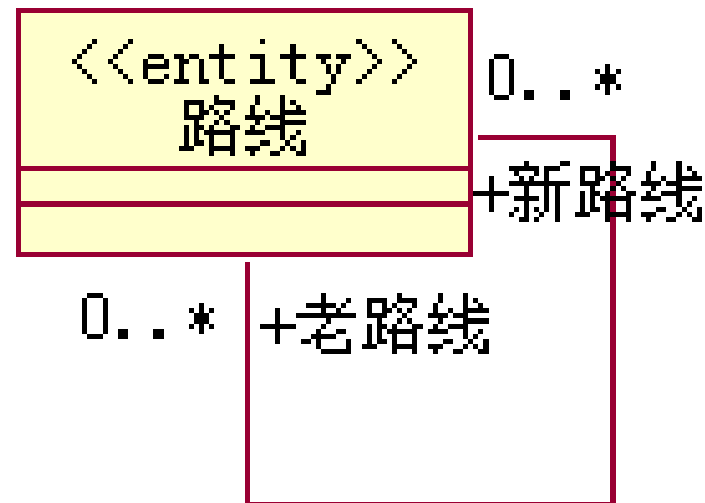
# 细化关联关系

- ❖ 关联具有：名称、端点和角色名、多重性
  - ◆ 名称：动词短语
  - ◆ 端点和端点名
  - ◆ 多重性表达式：\*, 1..\*, 1-40, 5, 3,5,8, ...



# 自反关联

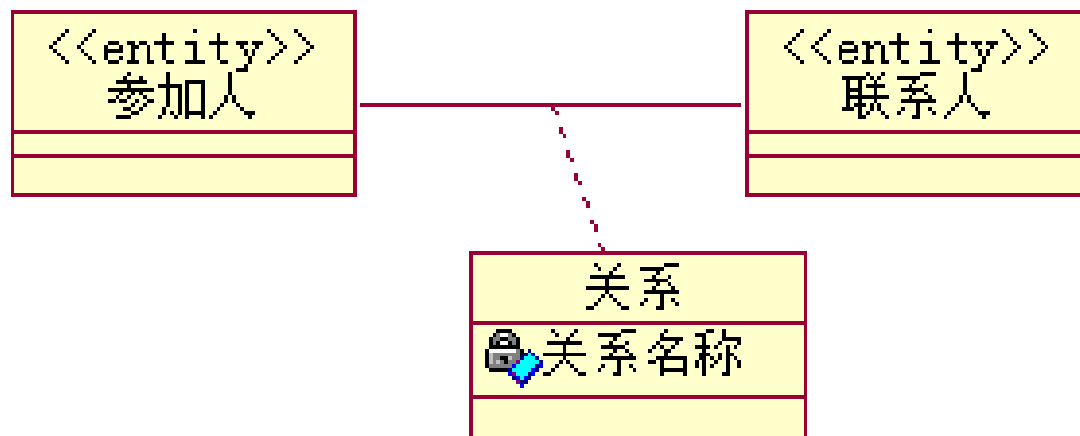
- ❖ 自反关联是指一个类自身之间存在关联，它表明同一个类的不同对象之间存在链接



# 关联类

## ❖ 关联类(Association Class)

- ◆ 是一种被附加到关联上的类，用来描述该关联自身所拥有的属性和行为
- ◆ 当某些属于关联自身的特征信息无法被附加到关联两端的类时，就需要为该关联定义关联类





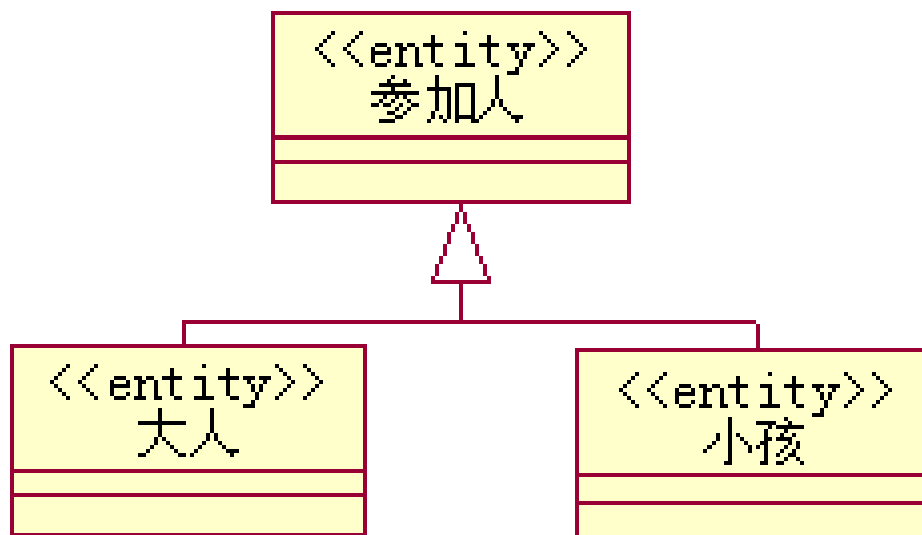
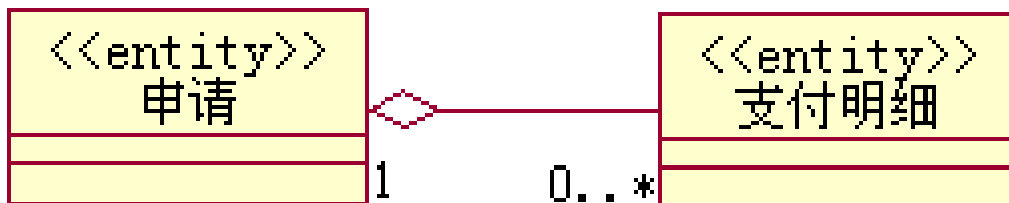
## 2 聚合关系与组合关系

- ❖ 聚合(Aggregation)关系是一种特殊的关联关系
  - ◆ 除了拥有关联关系所有的基本特征之外
  - ◆ 两个关联的类还分别代表“整体”和“部分”，意味着整体包含部分
- ❖ 可以在已有的关联关系基础上，通过分析两个关联的类之间是否存在如何语义来识别聚合关系
  - ◆ A（整体）由B（部分）构成
  - ◆ B（部分）是A（整体）的一部分
- ❖ 组合（Composition）关系是聚合的一种形式，具有很强的归属关系和一致的生存期
  - ◆ 部分不能脱离整体而存在

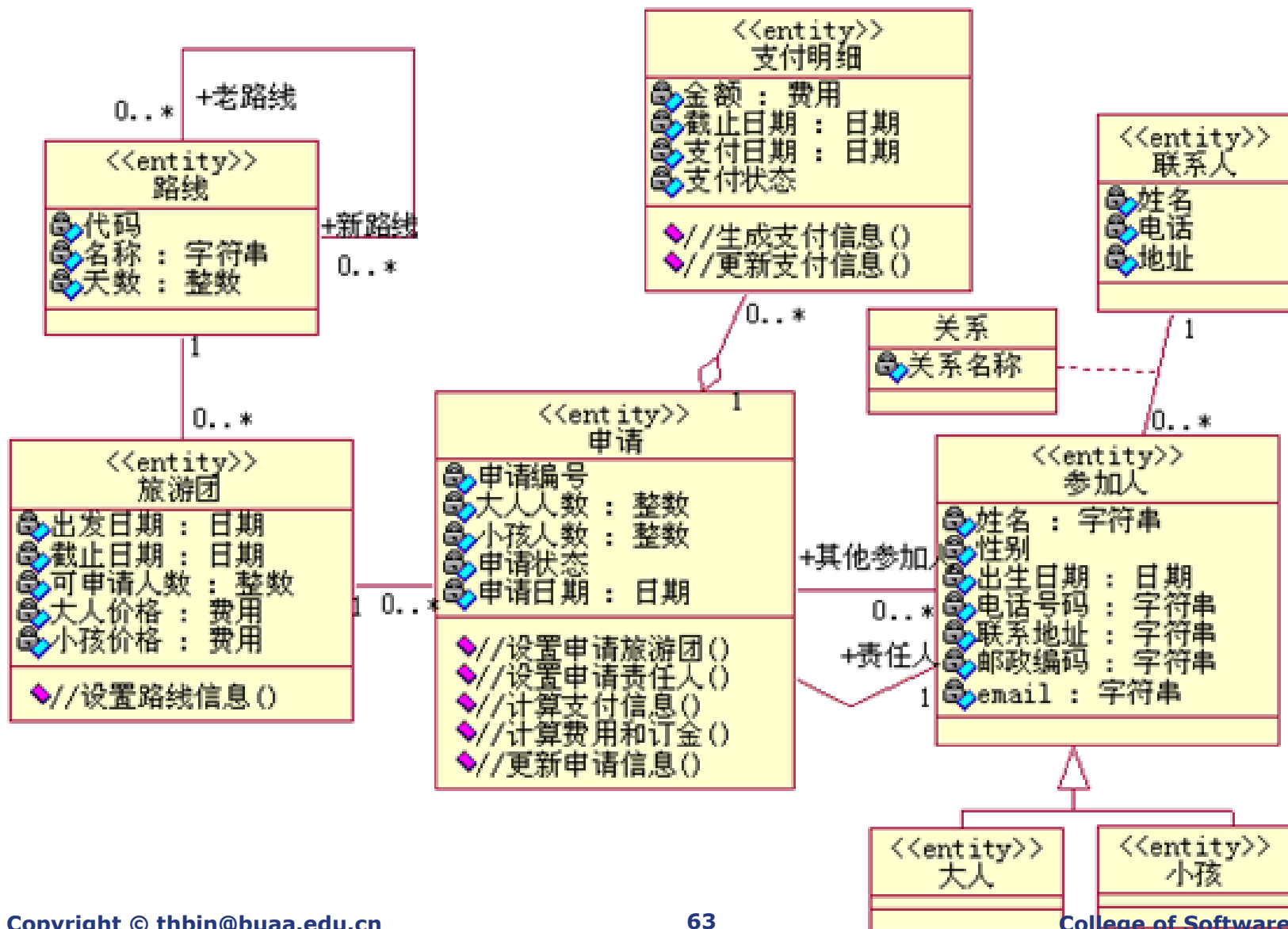
### 3 泛化关系

- ❖ 泛化是指类间的结构关系、亲子关系
  - ◆ 子类继承父类所具有的属性、操作和关联
- ❖ 分析阶段的泛化关系主要来自与业务对象模型，针对业务对象，结合业务领域的需求，从两个方面来提取泛化关系：
  - ◆ 是否有类似的结构和行为的类，从而可以抽取出通用的结构和行为构成父类
  - ◆ 单个业务对象是否存在一些不同类别的结构和行为，从而可以将这些不同类别的结构抽取出来构成不同的子类

# 实例：业务对象间的聚合和泛化



# 实例：旅游申请系统业务对象类图



# 作业

## ❖ 作业3：分析模型

### ◆ 进度安排

- 10月6日之前：寻找系统中的业务对象，建业系统分析模型：包括由业务对象构成的类模型和关键用例的交互模型
- 10月6日：讲解类模型和交互模型分析思路
- 10月6日之后：根据讨论结果，完成分析模型

### ◆ 提交

- DDL：2022年10月8日晚上10:00
- 交付物：完成系统核心业务类类图和关键用例交互模型，将全部分析模型整合在一个word文档中提交
  - 类图中至少包括类名、属性和类之间的关系，其中：类名和属性可以采用中文，可以考虑列一张类清单的表格，针对每个类的作用进行简要的说明；针对重点类的重点属性进行必要的解释
  - 关键用例行为分析的交互模型（如顺序图，建议每个同学至少一个）
  - 如果觉得有必要，可以首先定义一个简化的用例模型，作为分析的输入