



第三部分 面向对象的分析

Part III Object-Oriented Analysis



主要内容

- 系统分析方法;
- OOA的主要步骤;
- UML



为什么需要OOA

- 软件系统变得越来越庞大和复杂，并且容易变化。

系统很多代码是和用户界面相关的，而面向对象方法是处理这种用户界面的系统一种优于传统面向过程的方法；



系统分析

- 分析：
 - 就是研究问题空间；
 - 产生一个外部可观察的行为的规格说明；
 - 即：需求说明——系统需要做什么来满足用户，而不是实现这个系统。



系统分析方法

- 功能分解方法
- 数据流方法
- 信息模型方法
- 面向对象的方法



功能分解方法

- 功能分解：用一些步骤和子步骤划分功能。
 - 功能分解 =
 功能
 + 子功能
 + 功能界面
 - OOA在定义对象的服务时，也会使用功能分解。



数据流方法

- 数据流： 也称作“结构化分析”
 - 数据流方法 = 数据（和控制）流
+ 数据（和控制）转移 + 数据（和控制）存储
+ 终结符，小说明，数据字典
 - 这个流描述的是处理步骤（用过程抽象），而不是数据流及其逐步细化。
 - 数据流方法强调功能，容易出现应变能力弱的问题。



信息模型方法

- 信息模型工具：实体关系图；
信息模型化 = 对象 + 属性 + 关系
+ 超类/子类 + 联合对象
- 信息模型不是完整的方法，遗漏了
 - 服务：对每个对象的处理需求和属性封装在一起
 - 继承：表示属性和服务的共性
 - 消息：对象之间一个较窄的定义明确的界面
 - 结构：分类结构和组装结构



面向对象的方法

- 面向对象 =
对象（属性及其服务的封装） + 分类
+ 继承 + 通过消息的通讯



面向对象的系统分析

- 运用面向对象方法，对问题域和系统任务进行分析和理解；
- 对其中的事物和它们之间的关系产生正确的认识；
- 找出描述问题域和系统任务所需的类和对象；
- 定义类和对象的属性和行为，以及它们之间的结构关系（包括静态和动态两个方面，类是静态的，对象是动态的）



OOA

- OOA是在信息模型（实体关系图和语义数据模型）和面向对象程序设计语言两者的最好概念上建立的。
 - 来自信息模型：属性、关系、结构和代表问题空间事物的若干实例的对象表示。
 - 来自面向对象程序设计语言：属性和服务的封装，分类，继承。



提出OOA的原因

- 能在概括客观事物的三个基本方法的框架上定义和交流系统需求。
 - 三个基本方法：对象及其属性；分类结构；组装结构；
- 首先着眼于问题空间的理解
- 将属性及专用于那些属性的服务时为一个固有的整体。



提出OOA的原因（续）

- 使用自包含分块（对象之间依赖性最小）进行分析和说明。
- 通过显式表示共性得益。
- 提供一个支持分析（做什么）和设计（怎么做）的工具。
- 能适应系列化的系统和系统的调整。



OOA的主要步骤

- 标识对象;
- 标识结构;
- 定义主题;
- 定义属性;
- 定义服务;



五个层次

- 一旦模型建立，就可在五个主要层次上进行表示和复审。
 - 主题层
 - 对象层
 - 结构层
 - 属性层
 - 服务层

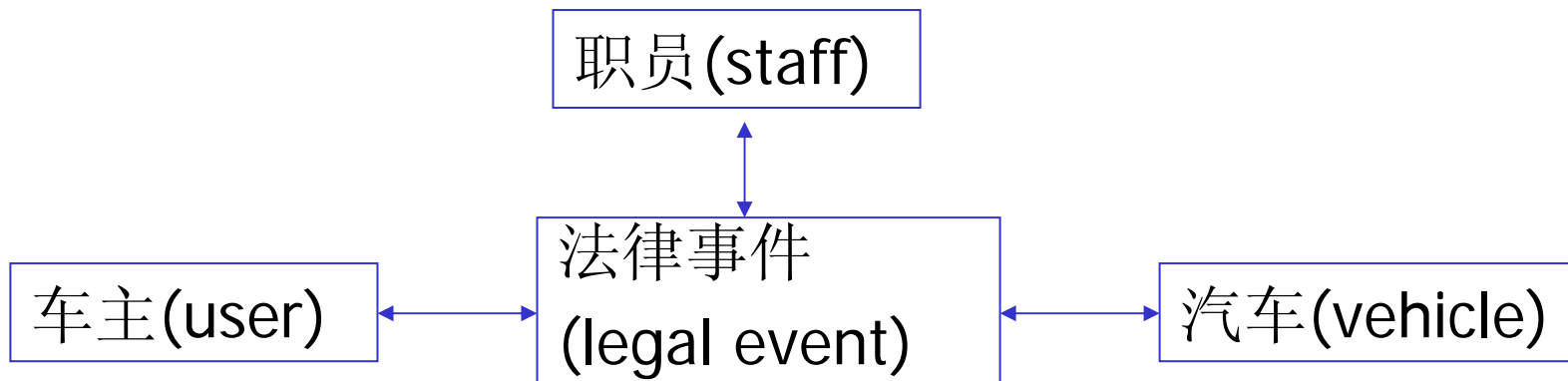


主题层

- 主题：

- 控制读者在同一时间所能考虑的模型规模的机制。

例如：在汽车发照注册系统中，可看到的主题





对象层

- 对象是数据及其操作的抽象，反映了系统保存相关的信息和与外界交互的能力。
前例包含6个对象：
汽车，发照，注册，车主，职员，组织机构。

组织(organization)

职员(clerk)

车主(owner)

汽车(vehicle)

发照(title)

注册(registration)



结构层

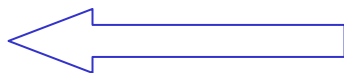
- 分类结构描述类属成员的构成，反映通用性和特殊性。
- 组装结构表示聚合，反映整体和组成部分。

在前例中



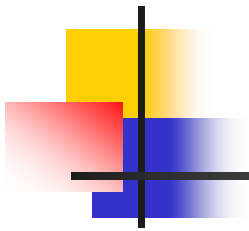
组织(organization)

职员(clerk)



整体

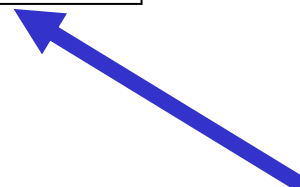
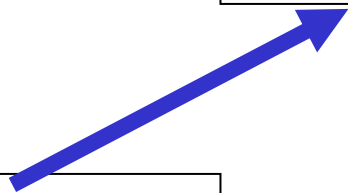
部分



法律事件
(legal event)

发照(title)

注册(registration)





汽车(vehicle)

客车(passenger)

卡车(truck)

摩托(motor)

拖车(trailer)



属性层

- 一个属性就是一个数据成员
- 属性用于描述对象特征。



组织(organization)

姓名(name)

管理者(manager)

地址(address)

电话(telephone)



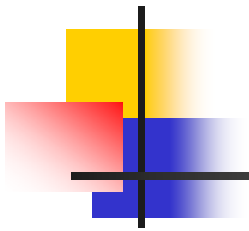
职员(clerk)

用户名(username)

授权(authorization)

开始日(begindate)

结束日(enddate)



法律事件
(legal event)

日期时间
(datetime)

发照(title)

号码(number)

所有权证明

(OwnershipEvidence)

费用(fee)

注册(registration)

起始日
(datetimeStart)

结束日
(datetimeEnd)

费用(fee)



汽车(vehicle)

号码(number)

年份(year)

样式(model)

毛重(gross)

乘客数(passengers)

柴油机(diesel)

色彩(color)

价格(cost)

里程(mileage)



服务层

- 服务是收到消息之后所执行的处理。
- 对外的接口
- 成员函数



OOA步骤一：标识对象

- 对象：数据和其上操作的抽象，反映一个系统为现实世界的事物保存信息与其发生相互作用的能力。
 - 如何寻找对象：结构，设备、事件、扮演的角色，组织等等
 - 对象的属性：需要什么样的信息，提供什么样的服务。



例子：实时空运系统

- 该系统提供一些自动支持一帮助全体职员提供高空运乘客和货物的速度。在这个系统中存在哪些对象呢？

任务（Mission） 航班（Flight）

货物（Cargo） 乘客（Passenger）

飞机（Aircraft） 雷达（Radar）



例子：实时空运系统（续）

Mission(code name, number, description)

Flight (number, origin, destination, date, time)

Passenger (name,rank ,number,origin,destination,date,
time)

Cargo (weight,dimensions,number, date , time , origin,
destination)

Aircraft (model, status,capacity,location)

Radar (frequency, pulse interval ,location, serch space)



OOA步骤二：标识结构

- 在问题空间里，发现分类结构和组装结构
 - 分类：找出继承关系——从一般到特殊
 - 组装：找出整体和部分的关系。



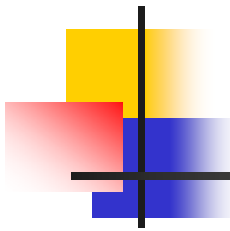
OOA步骤三：标识主题

- 主题提供了在一个时间内所能考虑和理解模型的多少部分的机制。
- 主题同时也给出了OOA模型中各图的概观。
- 类似数据库里的实体关系图。



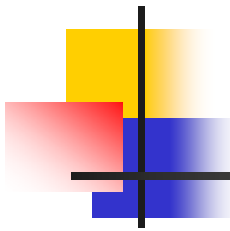
OOA步骤四： 定义属性

- 属性是描述对象的数据单元。
- 如何定义属性：
 - 标识属性；
 - 属性定位；
 - 标识和定义实例连接；
 - 修订对象；
 - 说明属性和实例连接的约束；



定义属性（一）：标识属性

- 分析问题空间中的对象，将现实中的对象与描述该对象的属性联系起来。
- 在原子概念层表示属性。
 - 原子概念是一个单独的数据单元。



定义属性（二）：属性定位

- 利用分类结构中的继承机制确定属性的位置。
 - 通用（共同）的属性放在结构的高层；
 - 特殊的属性放在底层。



定义属性（三）： 标识和定义实例连接

- 实例连接就是一个实例与另一个实例的映射关系。
- 分四步完成：
 - 添加实例连接线：
 - 反映问题空间中对象之间的连接
 - 定义多重性
 - 即：对象之间是一对多还是一对一的关系
 - 定义参与性
 - 对象之间的存在关系是强制性的还是任意性的
 - 检查特殊情况
 - 多对多的连接复杂的情况



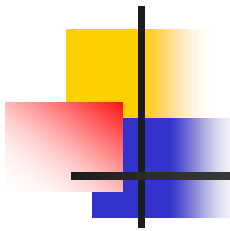
定义属性（四）：修订对象

- 带有独特值的属性意味着另一个分类结构。
- 单属性的对象意味着可采用更高层次的抽象。
- 属性值的冗余也预示着一个新的对象。



定义属性（五）： 说明属性和实例连接的约束

- 通过名字，描述，数据结构等说明属性。
- 通过实例之间映射的限制来说明实例连接约束。



定义属性——实例

- 在前文中提到的实时空运系统主题层



属性层

Mission
CodeName
Number
Description

Flight
Number
Origin
Destination

Aircraft
CallNumber
Status
Model
Capacity
Location

Radar
Frequency
PulseInterval
Location
SearchSpace

Shipment Item
Number
Origin NowAt
Destination

Passenger
Name
Rank

Cargo Item
Weight
Dimensions
Description



OOA步骤五： 定义服务

- 面向对象分析的一个主要目标：
 - 提供系统处理和流程需求的详细描述。
 - 首先讨论对象、结构、属性、实例连接
 - 然后考虑服务、消息连接。



什么是服务

■ 服务

- 就是收到一条消息之后所执行的处理。
- 定义服务的中心问题是为每个对象和分类结构定义所需要的行为。
- 针对三种行为的分类：
 - 直接动因→状态-事件-响应
 - 进化史→对象生命历程
 - 功能→最基本的服务



如何定义服务

- 定义服务的策略：
 - 标识服务（基本策略）—— 将服务名放入OOA模型图中；
 - 标识服务（辅助策略）—— 在OOA模型图中放入附加的服务名；
 - 标识消息连接 —— 建立实例间需要的处理。
 - 详细说明服务 —— 产生处理需求。



标识服务——基本策略

- 考虑三种服务
 - Occur: 实例的增加、修改、删除、选择
 - Calculate: 为某个实例或代表另一个实例计算结果。
 - Monitor: 执行对外界系统、设备或用户的运行监控。



标识服务——辅助策略

- 对象生命历程描绘一个对象或分类结构以后发生的事件。
- 分几步：
 - 定义基本的对象生命历程序列；
 - 检查每一步的演变；
 - 增加基本序列；
 - 增加服务；



状态-事件-响应

- 定义主要的系统状态；
- 对每一个状态，列出外部事件和所需要的响应；
- 扩充服务和消息连接。



标识消息连接

- 消息连接结合了事件响应和数据流两个方面；
 - 每一个消息连接既表示发出一条消息，也表示接收到一个响应。
 - 消息连接将一个实例对应于另一个实例。
 - 发送者发出一条消息给接收者去完成一些处理。
 - 在发送者的服务说明中给出所需要的处理名。
 - 在接收者的服务说明中定义这个处理。



详细说明服务

- 服务的规格说明分以下几个部分：
 - 集中于所需要的外部可见的行为。
 - 使用一个模板。
 - 增加图示以简化服务说明。
 - 增加支持的表。
 - 建立服务的文字叙述
 - 汇集所有文档



详细说明服务（一）

集中于所需外部可见的行为

- 为每个实例说明所需求的外部可见的行为。
 - 每次提问“这个需求是不是外部可见的”
 - 可以用动词时态来表示：
 - shall: 表示必须的可观测的需求;
 - will: 表示有待于将来确定的;
 - ing: 表现其他情况;



详细说明服务（二）

使用一个模板

- 使用模板建立需求框架表

- specification <对象名>

 - descriptionAttribute <...>

 - // 由“occur”服务(增加、修改、删除、选择)操纵其值的属性。

 - definitionData <...>

 - // 适用于多个对象或分类结构实例的属性。

 - alwaysDerivableAttribute <...>

 - // 随时可导出的属性(因而不需保存其数据)

 - occasionallyDerivableAttribute <...>

 - // 只是偶尔可导出的属性(因而需要保存其数据)



详细说明服务（二） 使用一个模板（续）

```
externalSystemInput <...>
// 来自设备或外界系统的输入
externalSystemOutput <...>
// 向设备或外界系统的输出。
instanceConectionConstraint
// 实例连接约束
stateEventResponse
// 用于标识服务的状态-事件-响应。
objectLifeHistory
// 用于标识服务的对象生命历程模式
service
//服务的规格说明：需求框架表，每个框架说明一个需求。
```



详细说明服务（三）

增加图示以简化服务说明

- 大多数服务说明结构比较简洁，但是对于少数服务，增加一些引导读者浏览说明的辅助信息是有益的。



详细说明服务（四）

增加支持的表

- 对象间的依赖关系在模型中用消息连接刻画。
- 为了概括交互作用，尤其对实时系统的分析和规格说明，应使用：
 - 服务及可适用的状态表
 - 关键运行路径分析表
 - 时间和规模分析表



详细说明服务（五）

建立服务的文字叙述

- 用框架表说明服务使得文本说明简洁扼要。
- 另一方面，管理人员、用户可能需要用更广泛的文字叙述建立系统需求文档。
- 同样，文字叙述文档也要简明扼要。



详细说明服务（六） 汇集所有文档

- 将所有文档汇集在一起，包括：
 - OOA图
 - 主题、对象、结构、属性和服务层；
 - OOA库
 - 每个服务或分类结构为一项
 - 支持表
 - 服务及可适用状态表，关键运行路径表，时间和规模表。



转到面向对象的设计

- 从分析到设计
 - 从OOA到OOD是一个累进的模型扩充过程。
 - OOA的各层模型化了“问题空间”
 - 作为OOA各层扩充的OOD，则模型化了一个特定的“实现空间”。
 - 这种扩充主要从增加属性和服务开始。



OOD(Object-Oriented Design)

- OOA是独立于程序设计语言的。
- 初步的OOD在很大程度上也是独立于程序设计语言的。
- 详细OOD则依赖于语言，并可以利用语言加以实现。



作为分析员

- OOA独立于程序设计语言

- 基本原则:

- 过程抽象、数据抽象、封装、继承、用消息通讯、普遍的组织方法（对象和属性、类和成员、整体和部分）、功能分类（基本函数、状态-事件-响应、对象生命周期）

- 模型表示及建立策略

- 对象、分类结构、组装结构、属性、实例连接、消息连接、服务



UML

(Unified Modeling Language)
