

# OOA

## 纲要

- OOA概述
- 标示实体—领域模型
- 职责分配的一般性原则
- 分析类和用例实现
- 架构分析

## 纲要

- **OOA概述**
- 标示实体—领域模型
- 职责分配的一般性原则
- 分析类和用例实现
- 架构分析

## 面向对象方法与其它

- 传统上，面向对象方法与**用例驱动**的分析方法深度融合；
- 面向对象分析一般以UML作为建模语言；
- RUP开发过程一般采用面向对象分析方法。

## 用例图的基本作用

- 描述系统的基本内涵及边界；
- 获取系统的需求；
- 评估和验证系统的基本架构；
- 驱动系统的实现和产生测试用例。

## 确定actor的方法

- 谁会提供、使用和删除信息？
- 谁会使用系统提供的功能？
- 谁会对某种需求感兴趣？
- 系统被什么部门使用？
- 谁将支持和维护系统？
- 该系统需要什么外部资源？
- 系统需要与别的系统进行交互吗？

**备注：**actor是角色，而不是具体的人，硬件设备等！  
**实例**

## 确定actor的方法

- 谁会提供、使用和删除信息？
- 谁会使用系统提供的功能？
- 谁会对某种需求感兴趣？
- 系统被什么部门使用？
- 谁将支持和维护系统？
- 该系统需要什么外部资源？
- 系统需要与别的系统进行交互吗？

数据、  
功能、  
基本角  
色和辅  
助角色  
等

备注：actor是角色，而不是具体的人，硬件设备等！  
实例

7

© 苏州大学计算机科学与技术学院

## 如何发现用例？

- 对每个行动者角色，系统需要提供什么支持？（参与行动者的何种任务？）
- 对于系统发生的某些事情，需要通知行动者吗？
- 对于外部出现的突然变化，行动者需要通知系统吗？
- 系统中什么消息被产生和改变？

实例：选课系统

8

© 苏州大学计算机科学与技术学院

## 用例描述

- 用例描述为外部行为者与系统的交互序列：
  - ✓ 系统可以作为整体出现，但在顺序图中常表现为某一对象集合；
  - ✓ 一般，用例描述采用业务语言进行，尽量避免技术语言，例如连接数据库；
  - ✓ 一般，用例描述中应避免出现实现要素，例如按钮，文本框等；
  - ✓ 用例描述应体现交互性。

9

© 苏州大学计算机科学与技术学院

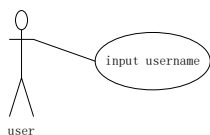
## 用例之间的关系

- Include
- Extend
- generalization

10

© 苏州大学计算机科学与技术学院

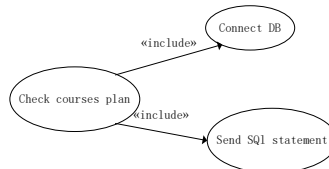
## 在用例分析中容易出现的问题(1/6)



11

© 苏州大学计算机科学与技术学院

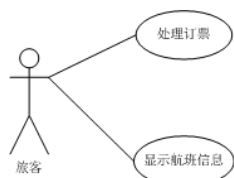
## 在用例分析中容易出现的问题(2/6)



12

© 苏州大学计算机科学与技术学院

## 在用例分析中容易出现的问题(3/6)

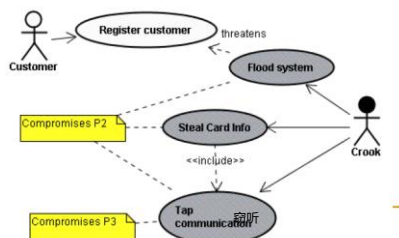


15

© 苏州大学计算机科学与技术学院

## 在用例分析中容易出现的问题(4/6)

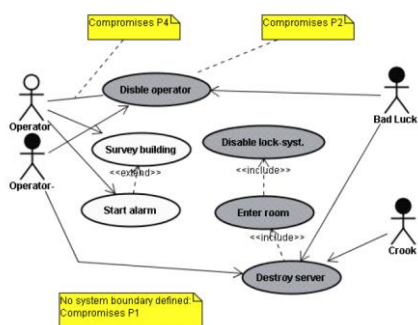
- Compromise P1: 没有定义清晰的系统边界
- Compromise P2: 与用户无关的需求 (非正常的功能性需求)
- Compromise P3: 不是系统边界内功能
- Compromise P4: 不正确的通信



14

© 苏州大学计算机科学与技术学院

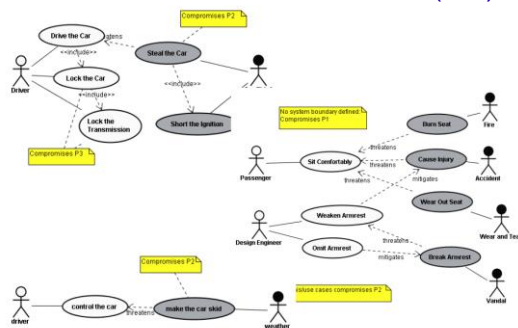
## 在用例分析中容易出现的问题(5/6)



15

学院与科技学院

## 在用例分析中容易出现的问题(6/6)



© 2004 Blackwell Publishing Ltd *Journal of Internal Medicine* 255: 103–110

## 什么是“分析”？

- 在软件工程中，分析指把用户需求转变为系统描述（规约）的过程；
- 系统规约，即系统的逻辑结构，指从开发者视角而言的系统模型；
- 基于过程（功能）的分析重点在**复杂功能的分解**；
- 面向对象分析重点在**对象和对象之间关系的识别**。

12

© 苏州大学计算机科学与技术学院

## 什么是“分析”？（续）

- 分析的目标是**理解和认识问题**
  - ✓ 开始建立目标系统的可视化模型，这种模型**独立于实现及别的相关的技术方面的要素**
- 分析的**基本任务**是把功能需求转换为软件概念
  - ✓ 得到构成系统的原始的对象集合，**重点关注行为**。
  - ✓ 所得到的分析模型可以快速度，无缝地进化为设计模型

18

© 苏州大学计算机科学与技术学院

## 分析与设计

### 分析:

- 理解问题
- 理想设计
- 行为
- 系统结构
- 功能需求
- 小模型

### 设计:

- 理解解决方案
- 行为和属性
- 性能
- Close to real code
- 对象生命周期
- 非功能需求
- 大模型

19

© 苏州大学计算机科学与技术学院

## 不同的对象类模型

- 领域对象模型
- 分析类模型
- 设计类模型
- 实现类

© 苏州大学计算机科学与技术学院

## OOA的三种方法

- 概念模型(Larman):从概念目录列表或名词性短语找出问题领域中的候选概念,形成类的集合,添加重要属性和关联关系;
  - 产生轻量级的类图
- CRC卡片(Beck, Cuningham)
  - 索引卡片
- 具有衍型的分析模型
  - Boundaries, entities和控制
- Grady Booch:《面向对象分析与设计》;
- Larman:《UML 和模式应用》

21

© 苏州大学计算机科学与技术学院

## CRC卡片

以下内容来自百度百科:

- CRC卡是一个标准索引卡集合,每一张卡片表示一个类;
- 在CRC建模中,用户、设计者、开发人员都有参与,完成对整个面向对象工程的设计。

CRC Card Example	
<b>LibraryMember</b>	
<b>Responsibilities</b>	<b>Collaborators</b>
Maintain data about copies currently borrowed	
Meet requests to borrow and return copies	Copy
<b>Copy</b>	
<b>Responsibilities</b>	<b>Collaborators</b>
Maintain data about a particular copy of a book	
Inform corresponding Book when borrowed and returned	Book
<b>Book</b>	
<b>Responsibilities</b>	<b>Collaborators</b>
Maintain data about one book	
Know whether there are borrowable copies	
<b>Questions:</b> Do these three classes conform to our notion of a good design? What is their level of cohesion and coupling?	

技术学院

## OOA的基本过程

- 识别对象
  - CRC, 概念, 衍型 (stereotypes) 等
- 组织对象
- 对象分类
- 识别对象之间的关系
- 定义对象行为
- 定义对象内部属性

23

© 苏州大学计算机科学与技术学院