

《需求分析与系统设计》

Requirements Analysis and System Design

任课教师： 范 国 祥

电 话： 0451-86418876-811(O)

13199561265(Mobile)

邮 箱： fgx@hit.edu.cn

哈工大计算学部
国家示范性软件学院
软件工程教研室

2023. 09



本章主要内容

1. 敏捷开发中的“用户故事” (User Story)
2. 面向对象方法中的“用例” (Use Case)
3. 用例建模的基本过程
4. 用例模型的提交物
5. 活动图 & 泳道图



说明

- 在敏捷开发中，需求表述为一组“**用户故事（User Story）**”
- 在传统的OO分析方法中，需求被表述为一组“**用例（Use Case）**”
- 二者都是对需求的描述形式
- 区别在于大小不同、具体形式不同
- 对初学者来说，可以将二者看作等价模型，在实践中汇集二者的优点



本章主要内容

1. 敏捷开发中的“用户故事” (User Story)
2. 面向对象方法中的“用例” (Use Case)
3. 用例建模的基本过程
4. 用例模型的提交物
5. 活动图 & 泳道图



何谓“用户故事”？

- 用户故事：
 - 对软件用户（或所有者）有价值的功能性的简明的书面描述
- 从用户的角度来描述用户渴望得到的软件功能：
 - 角色(谁要使用这个功能)、目标/活动(需要完成什么样的功能)、商业价值(为什么需要这个功能，这个功能带来什么样的价值)
- 三个组成部分：
 - 卡片（Card）：用户故事一般在小卡片上写着故事的简短描述，工作量估算等
 - 交谈（Conversation）：用户故事背后的细节来源于和客户或者产品负责人的交流沟通
 - 确认（Confirmation）：通过验收测试确认用户故事被正确完成



用户故事的描述

As a **[user role]** I want to **[goal]** so I can **[reason]**

作为一个<角色>, 我想要<活动>, 以便于<商业价值>

As who, I want
what so that why.

Who (user role) -- 角色: 谁要使用这个功能?

What (goal) -- 功能: 需要完成什么样的功能?

Why (reason) -- 价值: 为什么需要这个功能, 功能带来什么样的价值?

例如:

① As a **registered user** I want to **log in** so I can **access subscriber-only content**.

② 作为一个“**网站管理员**”, 我想要“**统计每天有多少人访问了我的网站**”, 以便于“**赞助商了解我的网站会给他们带来什么收益**”。



用户故事卡的正面：Conversation

#0001 **USER LOGIN** Fibonacci Size # 3

As a [registered user], I want to [log in], so I can [access subscriber content].

For new features, annotated wireframe. For bugs, steps to reproduce with screenshot. For non-functional stories, explain scope/standards.

User Login

Username:

Password:

Remember me ☐

Login

[Forgot password?](#)

[message]

Store cookie if ticked and login successful.

Display message here if not successful. (see confirmation scenarios over)

User's email address. Validate format.

Authenticate against SRS using new web service.

Go to forgotten password page.



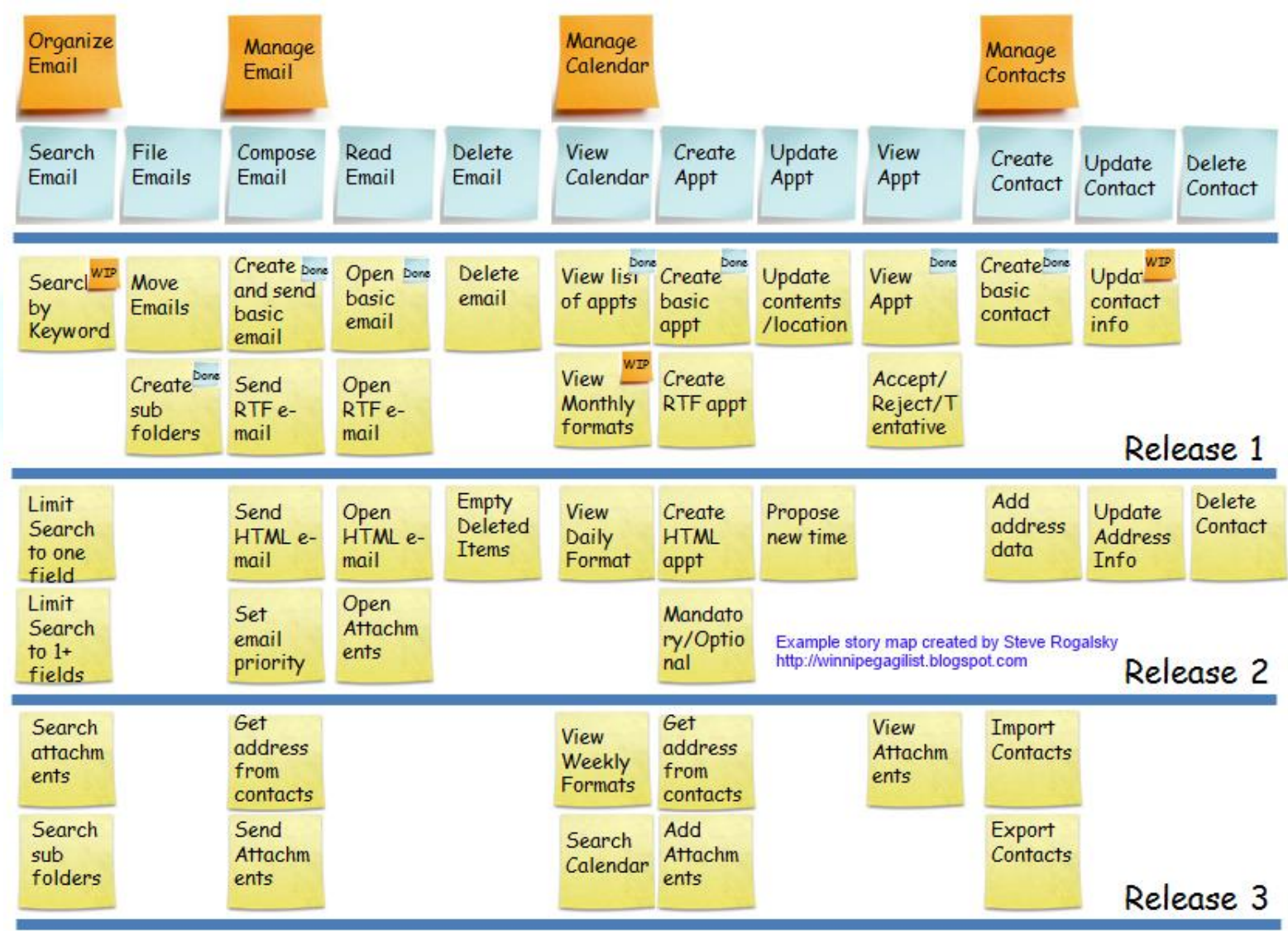
用户故事卡的反面：Confirmation

Confirmation

1. Success – valid user logged in and referred to home page.
 - a. 'Remember me' ticked – store cookie / automatic login next time.
 - b. 'Remember me' not ticked – force login next time.
2. Failure – display message:
 - a) "Email address in wrong format"
 - b) "Unrecognised user name, please try again"
 - c) "Incorrect password, please try again"
 - d) "Service unavailable, please try again"
 - e) Account has expired – refer to account renewal sales page.



用户故事板



Example story map created by Steve Rogalsky
<http://winnipegagilist.blogspot.com>



好的用户故事应具备的特征：INVEST

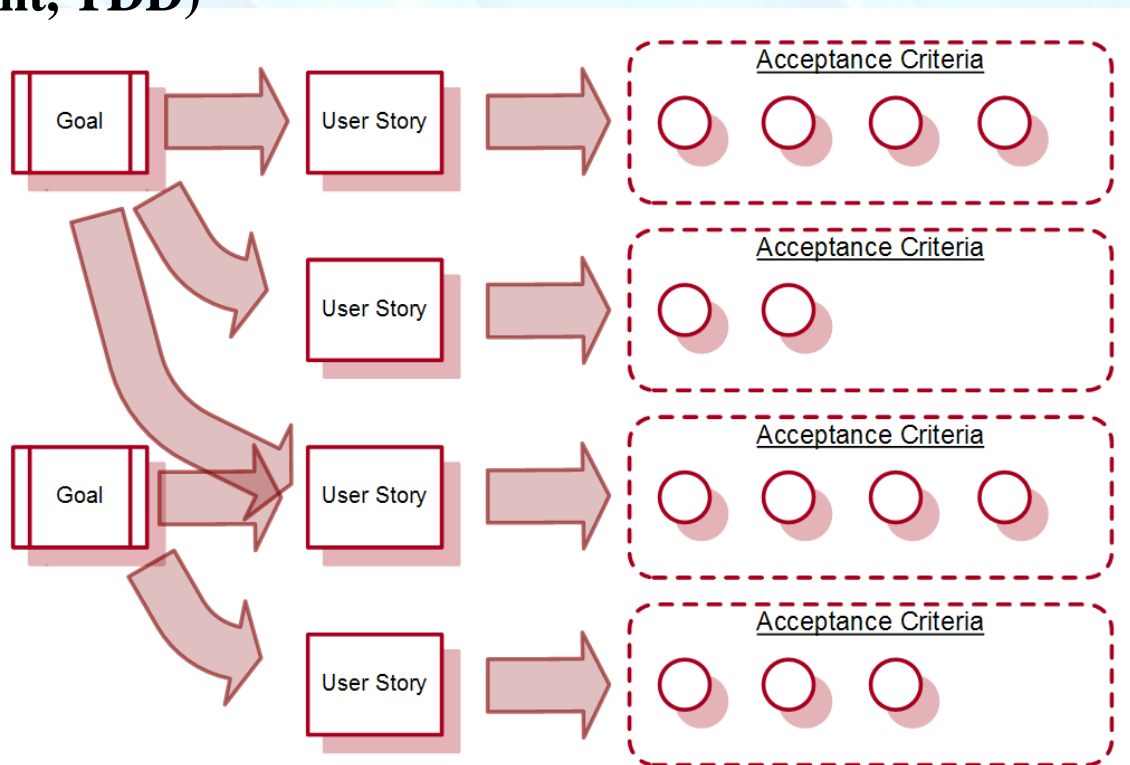
- **I**ndependent – 尽可能独立
- **N**egotiable – 可讨论的：它不是一个合同，没有详细的规约，后续开发阶段可以不断协商和改进
- **V**aluable – 对用户/客户有价值的，以用户可理解的语言书写，是系统的“特性”而非“开发任务”
- **E**stimatable – 其工作量可以估计
- **S**mall – 小，而不是大
- **T**estable – 可测试的、可验证的





用户故事支持验收测试*

- 在每个用户故事背后，列出未来用户测试时可能使用的“测试用例”，作为该故事是否已被完整实现的基本标准
- 对应于敏捷开发的一个基本思想：在写代码之前先写测试(Test-Driven Development, TDD)





用户故事支持敏捷迭代计划*

- 针对识别出的每一个故事，使用Story Point估算其工作量
 - 故事点：一个达到共识的基本时间单位，例如1天
 - 使用预定的值：1/2、1、2、3、5、8、13、20、40、80
- 团队成员分别估计(而不是由项目经理一人决定)，差异较大时面对面讨论，发现分歧，形成共识

形成估算清单

Story	To Do		In Process	To Verify	Done
As a user, I... 8 points	Code the... 9	Test the... 8	Code the... DC 4	Test the... SC 6	Code the... D
	Code the... 2	Code the... 8	Test the... SC 8		Test the... SC 8
	Test the... 8	Test the... 4			Test the... SC 6
As a user, I... 5 points	Code the... 8	Test the... 8	Code the... DC 8		Test the... SC
	Code the... 4	Code the... 6			Test the... SC 6



用户故事支持敏捷迭代计划*

- 对用户故事排列优先级
- 安排责任人
- 汇聚为迭代计划并发布
- 开发过程中监控进度





课外阅读

- **Mick Cohn. User Stories Applied: for Agile Software Development**
(《**用户故事与敏捷方法**》，清华大学出版社，2010年4月，ISBN 9787302223405)

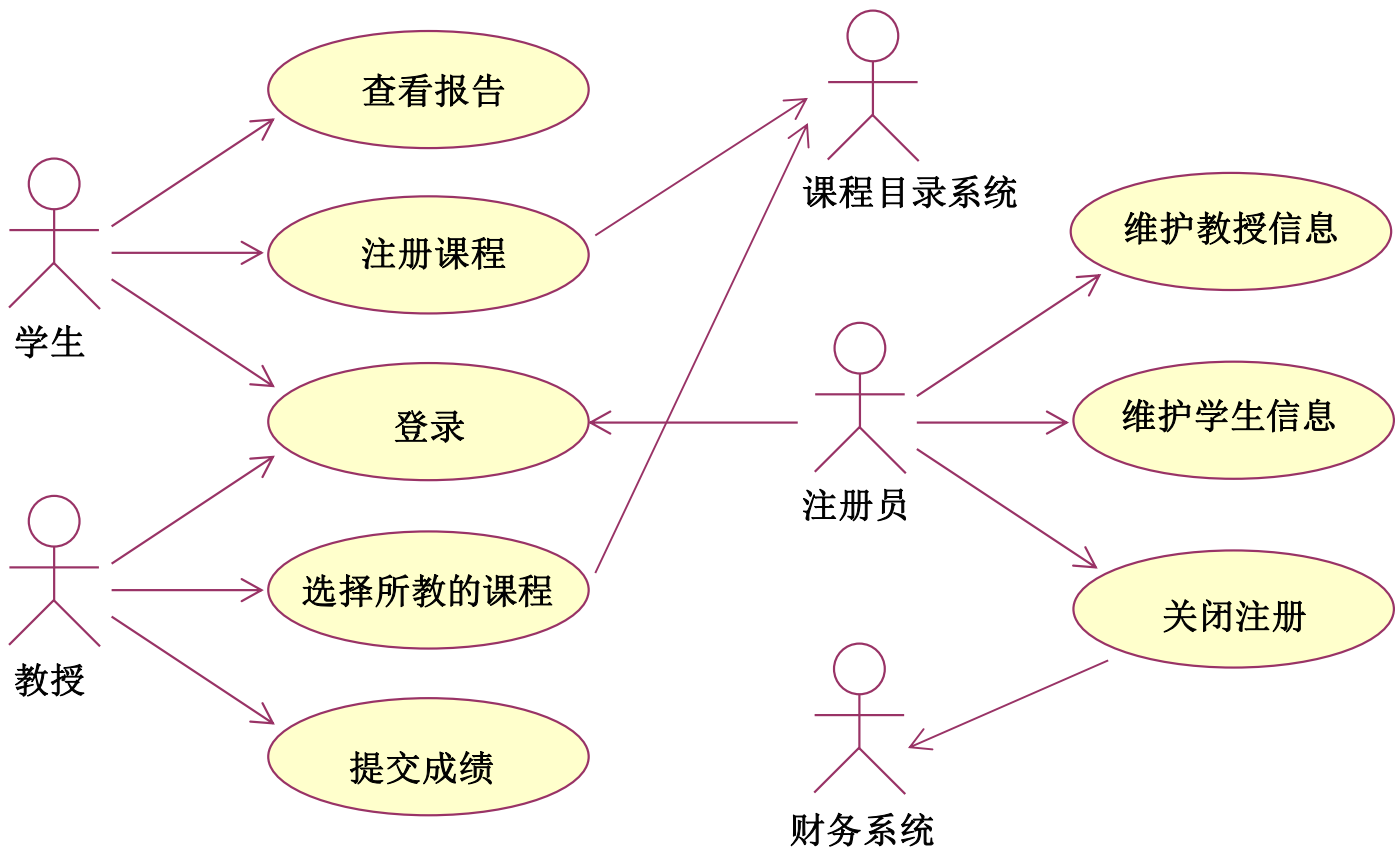


本章主要内容

1. 敏捷开发中的“用户故事” (User Story)
2. 面向对象方法中的“用例” (Use Case)
3. 用例建模的基本过程
4. 用例模型的提交物
5. 活动图 & 泳道图



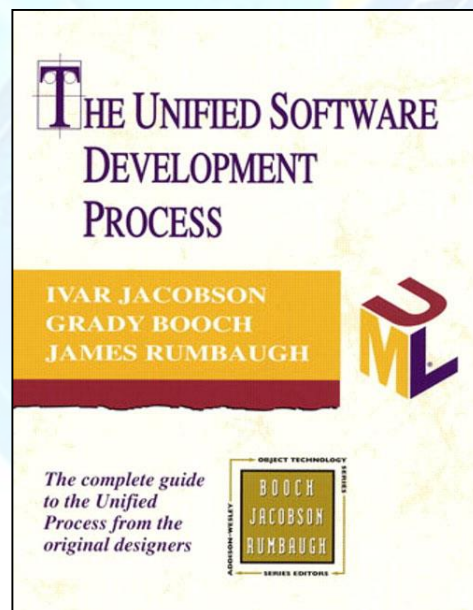
需求分析技术：用例





什么是用例(Use Case)?

- 用例(Use Case): 表示系统所提供的服务或可执行的某种行为
 - 定义了系统是如何被参与者所使用的, 描述了参与者为了使用系统所提供的某一完整功能而与系统之间发生的一段“对话”
 - 用例的概念在1986年由Ivar Jacobson正式提出之后被广泛接受, 迅速发展, 已成为OO、UML、RUP的标准规范和方法





什么是用例(Use Case)?

- 用例：站在用户角度定义软件系统的外部特性
- 四大特征：
 - 行为序列(sequences of actions)：一个用例由一组可产生某些特定结果的行为构成，这些行为是不可再分解的(接收用户输入、执行、产生结果)
 - 系统执行(system performs)：系统为外部角色提供服务
 - 可观测到的、有价值的结果(observable result of value)：用例必须对用户产生价值
 - 特定的角色(particular actor)：某人、某台设备、某外部系统、等等，能够触发某些行为



用例方法的基本思想

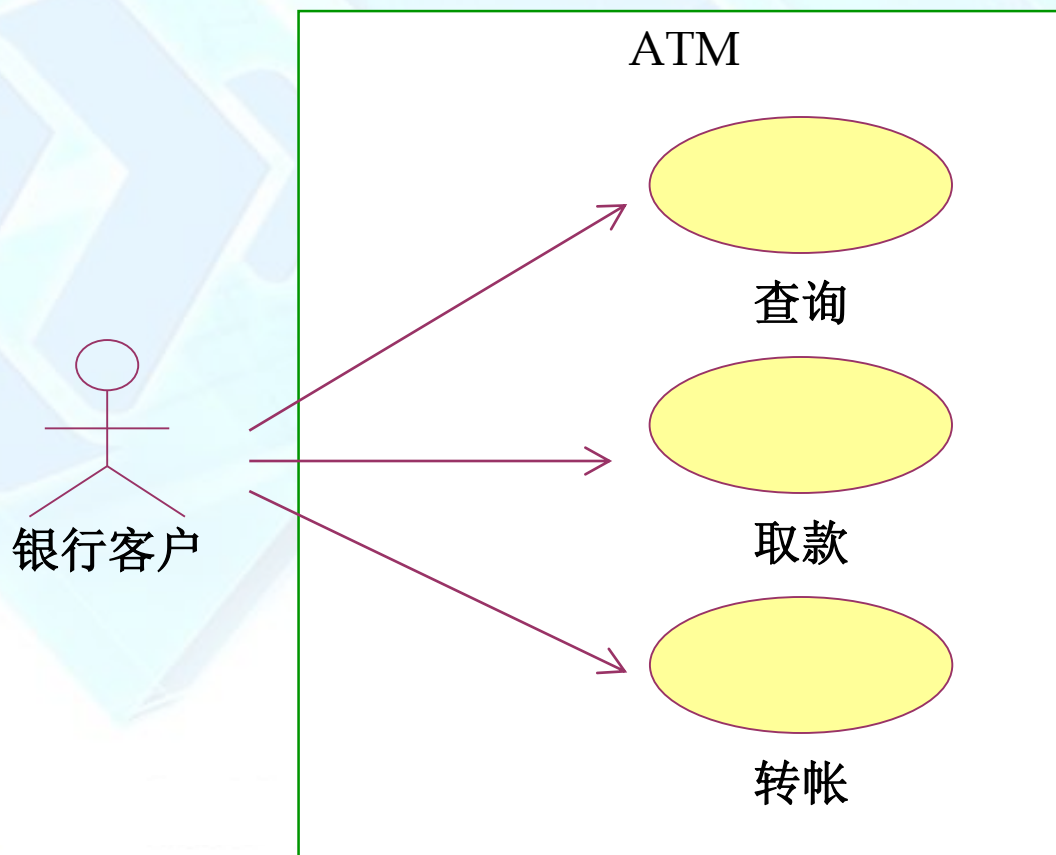
- 用例方法的基本思想：从用户的角度来看，他们并不想了解系统的内部结构和设计，他们所关心的是系统所能提供的服务，也就是被开发出来的系统将是如何被使用的
- 用例模型主要由以下模型元素构成：
 - **参与者(Actor)**：存在于被定义系统外部并与该系统发生交互的人或其他系统，代表系统的使用者或使用环境
 - **用例(Use Case)**：表示系统所提供的服务或可执行的某种行为
 - **通讯关联(Communication Association)**：用于表示参与者和用例之间的对应关系，它表示参与者使用了系统中的哪些服务(用例)、系统所提供的服务(用例)是被哪些参与者所使用的





示例：ATM系统的用例

- 参与者：银行客户
- 用例：银行客户使用自动提款机来进行银行帐户的查询、提款和转帐交易





关于“通讯关联”的几点说明

- 通讯关联表示的是参与者和用例之间的关系：
 - 箭头表示在这一关系中哪一方是对话的主动发起者，箭头所指方是对话的被动接受者
 - 如果不想强调对话中的主动与被动关系，可以使用不带箭头的关联实线
 - 通讯关联不表示在参与者和用例之间的信息流，并且信息流向是双向的，它与通讯关联箭头所指的方向没有关系





用例的内部剖析

- 用例 \neq 椭圆 + 名字！

Name of the Use Case (用例的名字)

Description (描述)

Actor(s) (参与者)

Flow of events(事件流)

Basic flow(常规流)

Event 1 (事件)

Event 2

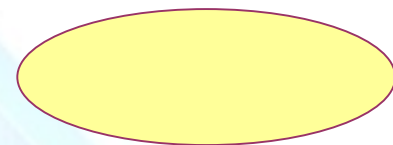
.....

Alternate flow(备选流)

Pre-conditions (前置条件)

Post-conditions (后置条件)

.....



Use case



用例方法的优点

- 系统被看作是一个黑箱，并不关心系统内部是如何完成它所提供的功能的
- 首先描述了被定义系统有哪些外部使用者(抽象为Actor)、这些使用者与被定义系统发生交互
- 针对每一参与者，又描述了系统为这些参与者提供了什么样的服务(抽象成为Use Case)、或者说系统是如何被这些参与者使用的
- 用例模型容易构建、也容易阅读
- 完全站在用户的角度上，从系统外部来描述功能
- 帮助系统的最终用户参与到需求分析过程中来，其需求更容易表达出来



本章主要内容

1. 敏捷开发中的“用户故事” (User Story)
2. 面向对象方法中的“用例” (Use Case)
3. 用例建模的基本过程
4. 用例模型的提交物
5. 活动图 & 泳道图



用例建模的基本过程

- Step 1: 识别并描述参与者(actor);
- Step 2: 识别用例(use case), 并给出简要描述;
- Step 3: 识别参与者与用例之间的通讯关联(Association);
- Step 4: 给出每一个用例的详细描述
- Step 5: 细化用例模型



Step 1: 识别并描述参与者

■ 通过以下问题来识别Actor:

- 谁使用这个系统的功能?
- 谁从该系统获得信息?
- 谁向该系统提供信息?
- 该系统需要访问(读写)那些外部硬件设备?
- 谁来负责维护和管理这个系统以保证其正常运行?
- 该系统需要与其他系统进行交互吗?





识别并描述参与者

- 例1：对一个图书馆管理系统来说，有哪些参与者？

- 普通读者
- 图书管理员



- 例2：对ATM系统来说，有哪些参与者？

- 银行客户
- ATM维护人员





特殊的参与者：系统时钟

- 有时候需要在系统内部定时的执行一些操作，如检测系统资源使用情况、定期生成统计报表等等
- 但这些操作并不是由外部的人或系统触发的
- 对于这种情况，可以抽象出一个系统时钟或定时器参与者，利用该参与者来触发这一类定时操作
- 从逻辑上，这一参与者应该被理解成是系统外部的，由它来触发系统所提供的用例对话





Step 2: 识别用例(use case)

- 找到参与者之后，据此来确定系统的用例，主要是看各参与者需要系统提供什么样的服务，或者说参与者是如何使用系统的
- 寻找用例可以从以下问题入手(针对每一个参与者):
 - 参与者使用该系统执行什么任务?
 - 参与者是否会在系统中创建、修改、删除、访问、存储数据? 如果是的话，参与者又是如何来完成这些操作的?
 - 参与者是否会将外部的某些事件通知给该系统?
 - 系统是否会将内部的某些事件通知该参与者?



Use case



识别用例

■ 例1：对图书馆管理系统来说，有哪些用例？

— 图书管理员

- 登录
- 管理读者信息
- 管理图书信息
- 登记借书
- 登记还书

— 普通读者：

- 登录
- 预订图书
- 取消预订
- 查询浏览图书信息

■ 例2：对ATM系统来说，有哪些参与者？

— 银行客户

- 查询
- 取款
- 转帐

— ATM维护人员

- 维护系统

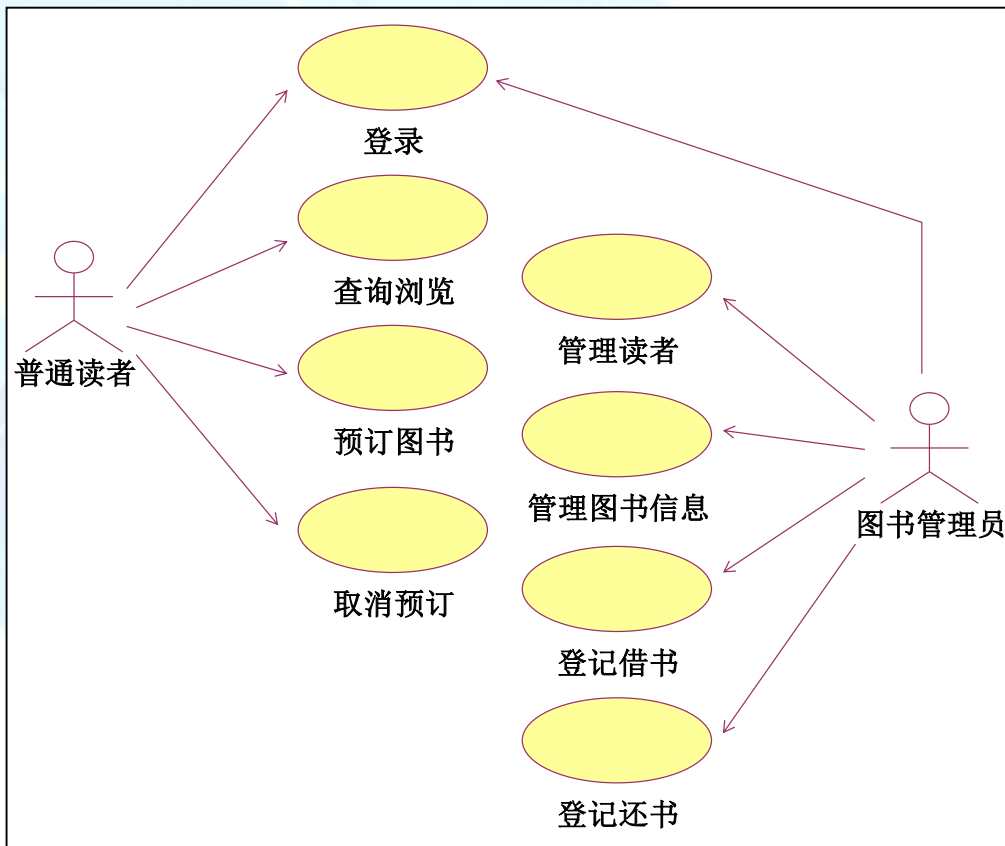
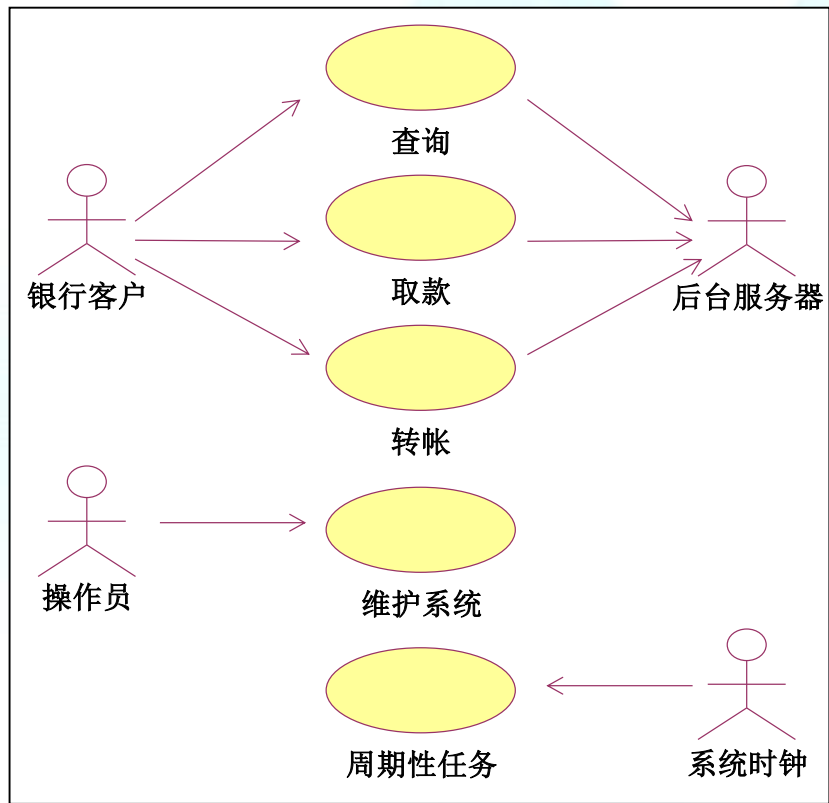


识别用例的几点注意事项

- 用例必须是由某一个actor触发而产生的活动，即每个用例至少应该涉及一个actor
- 如果存在与actor不进行交互的用例，需要将其并入其他用例，或者是检查该用例相对应的参与者是否被遗漏
- 反之，每个参与者也必须至少涉及到一个用例，如果发现有不与任何用例相关联的参与者存在：
 - 仔细考虑该参与者是如何与系统发生对话的
 - 由参与者确定一个新的用例
 - 该参与者是一个多余的模型元素，应该将其删除



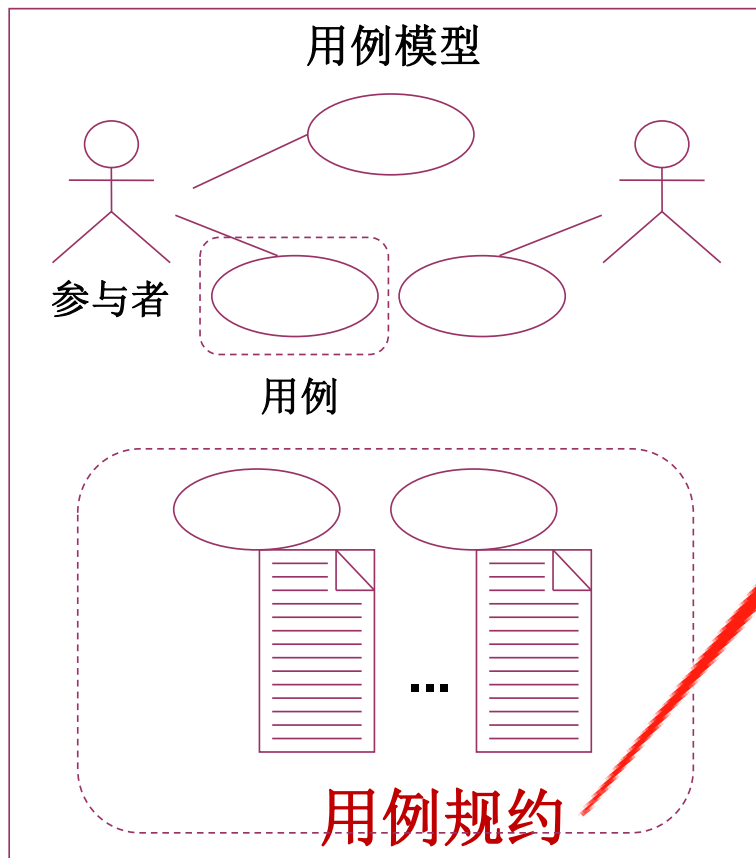
Step 3: 识别参与者与用例之间的通讯关联





Step 4: 给出用例的详细描述

- 单纯的用例图并不能描述完整的信息，需要用文字描述不能反映在图形上的信息



Name of the Use Case (用例的名字)

Description (描述)

Actor(s) (参与者)

Flow of events(事件流)

Basic flow(常规流)

Event 1 (事件)

Event 2

.....

Alternate flow(备选流)

Pre-conditions (前置条件)

Post-conditions (后置条件)

.....



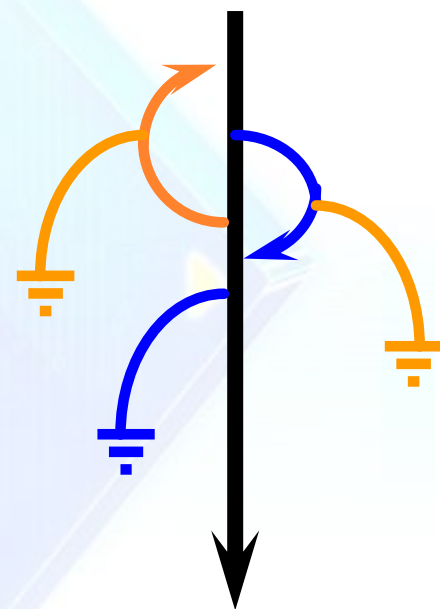
事件流

■ 用例的事件流：

- 说明用例如何启动，即哪些参与者在何种情况下启动用例？
- 说明参与者与用例之间的信息处理过程
- 说明用例在不同条件下可以选择执行的多种方案
- 说明用例在什么情况下才能被视作完成

■ 分为常规流和备选流两类：

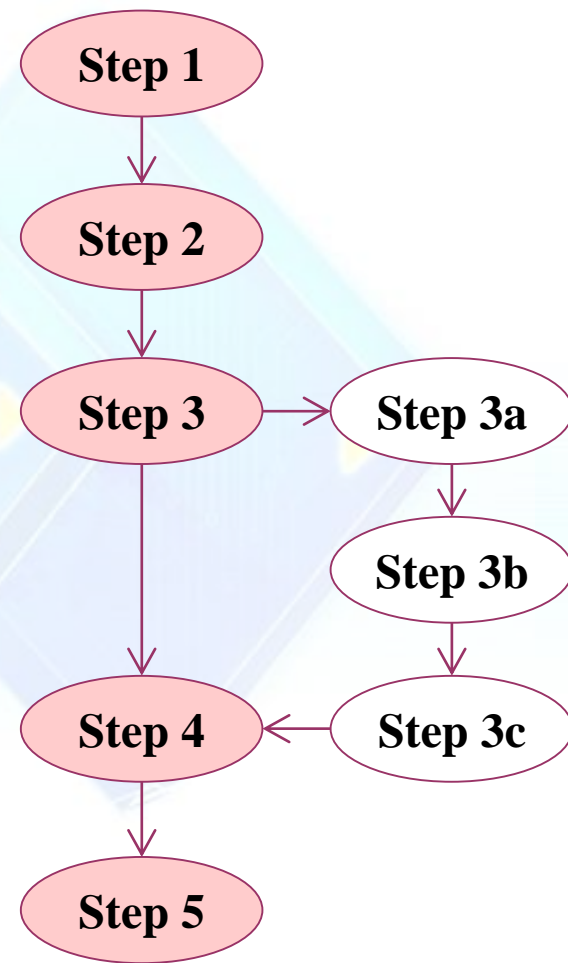
- **常规流**：描述该用例最正常的一种场景，系统执行一系列活动步骤来响应参与者提出的服务请求
- **备选流**：负责描述用例执行过程中异常的或偶尔发生的一些情况





常规事件流

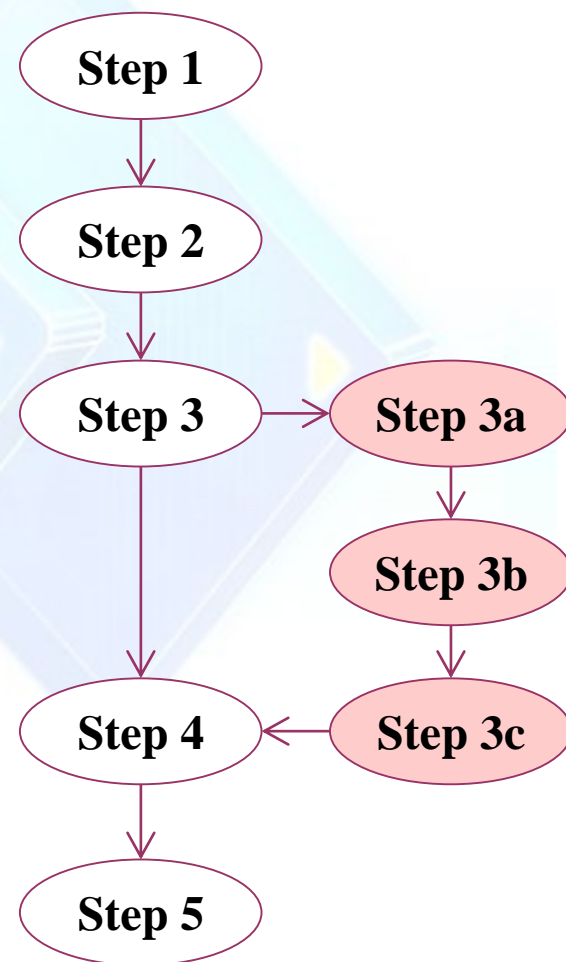
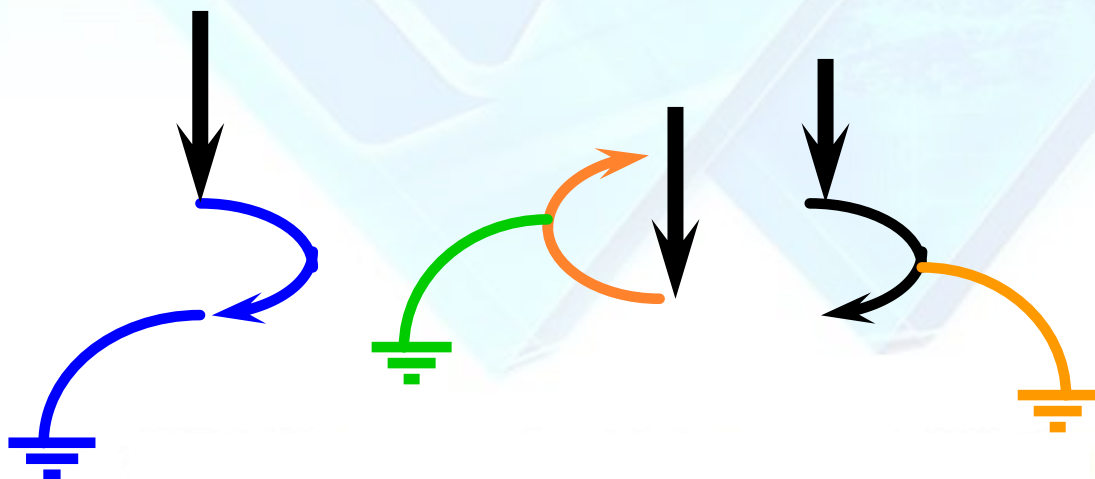
- 每一个步骤都需要用数字编号以清楚地标明步骤的先后顺序
- 用一句简短的标题来概括每一步骤的主要内容
- 对每一步骤，从正反两个方面来描述
 - 参与者向系统提交了什么信息
 - 对此系统有什么样的响应





备选事件流

- 备选流的描述格式可以与基本流的格式一致，也需要编号并以标题概述其内容
 - 起点：该备选流从事件流的哪一步开始
 - 条件：在什么条件下会触发该备选流
 - 动作：系统在该备选流下会采取哪些动作
 - 恢复：该备选流结束之后，该用例应如何继续执行





[案例] 用例描述

用例：登记借书

1. 目标：

本用例允许图书管理员登记普通读者的借书记录

2 事件流：

2.1 常规流程

当读者希望借书、图书管理员准备登记有关的借书记录时，本用例开始执行

- (1) 系统要求管理员输入读者的注册号和所借图书号；
- (2) 图书管理员输入信息后，系统产生一个唯一的借书记录号；
- (3) 系统显示新生成的借书记录；
- (4) 图书管理员确认后，系统增加一个新的借书记录

2.2 备选流程

(1) 读者没有注册：

在主流程中，如果系统没有读者的注册信息，系统将显示错误信息，用例结束

(2) 所借图书不存在

在主流程中，如果所借图书已被借出或系统中无该图书，系统将显示错误信息，用例结束

3 前提条件：用例开始前，图书管理员必须在系统登录成功

4 后置条件：如果用例执行成功，该读者的借书记录被更新，否则，系统状态不变



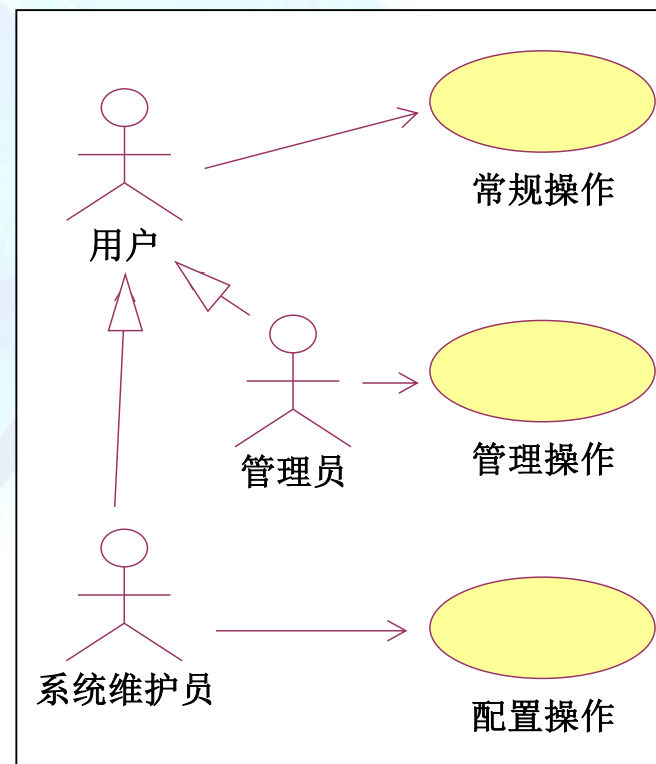
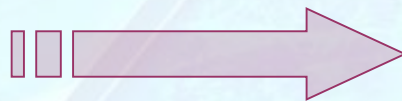
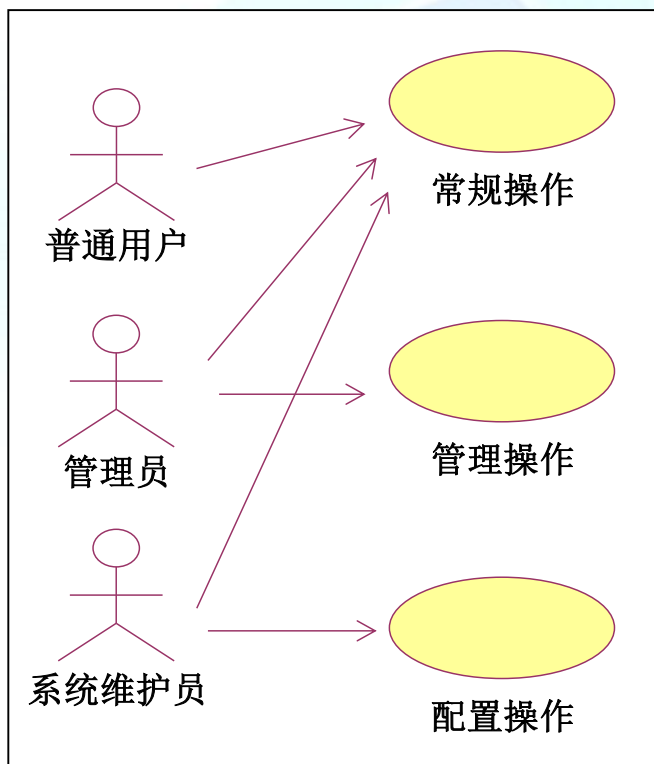
Step 5: 细化用例模型

- 在一般的用例图中，只需表述参与者和用例之间的通讯关联
- 除此之外，还可以描述：
 - 参与者与参与者之间的泛化(generalization)
 - 用例和用例之间的包含(include)
 - 用例和用例之间的扩展(extend)
 - 用例和用例之间的泛化(generalization)关系
- 利用这些关系来调整已有的用例模型，把一些公共的信息抽取出来复用，使得用例模型更易于维护



参与者之间的关系

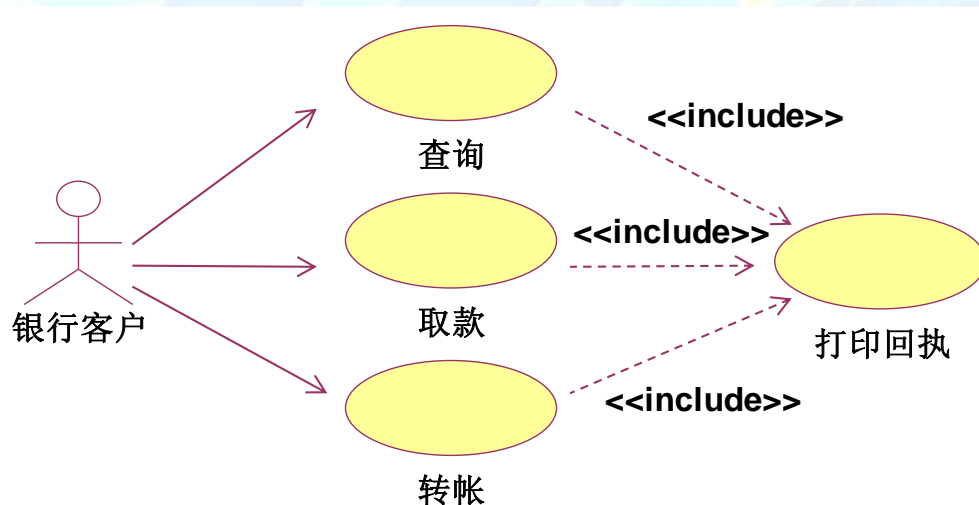
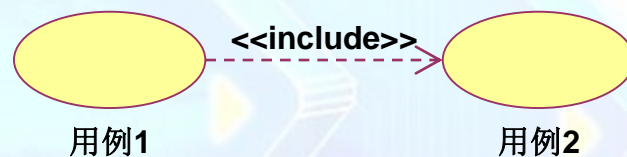
- 参与者之间也可以有泛化(Generalization)关系





用例之间的关系：包含(include)

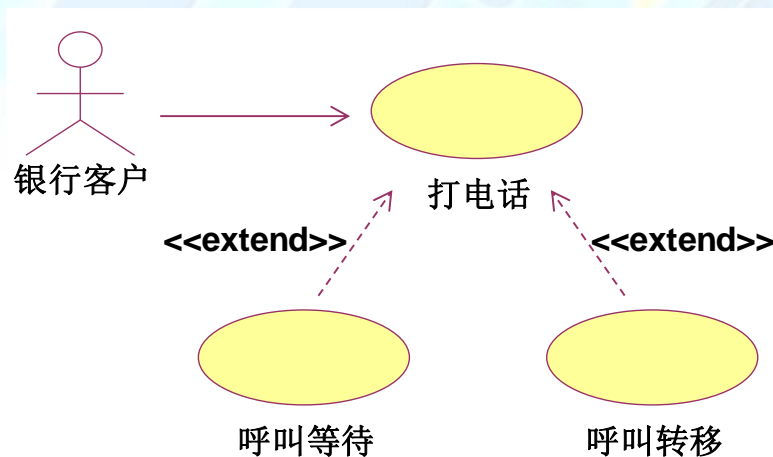
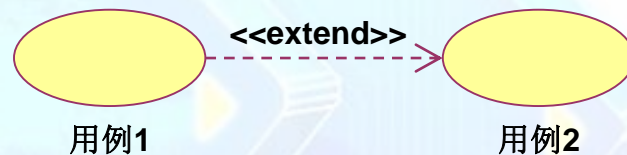
- “包含关系”是通过在关联关系上加入<<include>>标记来表示
- 语义：用例1会用到用例2，用例2的事件流将被插入到用例1的事件流中





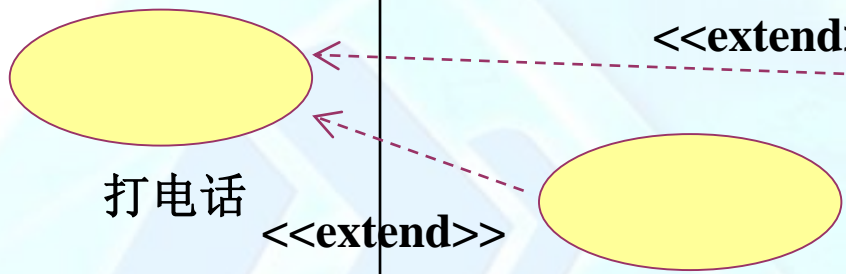
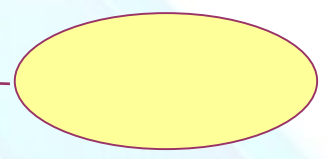
用例之间的关系：扩展(extend)

- “扩展关系”是通过在关联关系上加入<<extend>>标记来表示
- 语义：用例2在某些特定情况下会用到用例1，此时，用例1的事件流将被插入到用例2的事件流中





用例之间的关系：扩展(extend)

 <p>打电话</p> <p><<extend>></p> <p>呼叫等待</p>	 <p>呼叫转移</p>	
常规流： 1 拨号 2 建立通话链路 3 通话 4 挂机	常规流： 1 如果应答方正忙，用铃声提示应答方并保持拨号呼叫	常规流： 1 如果应答方无应答，进行呼叫转移

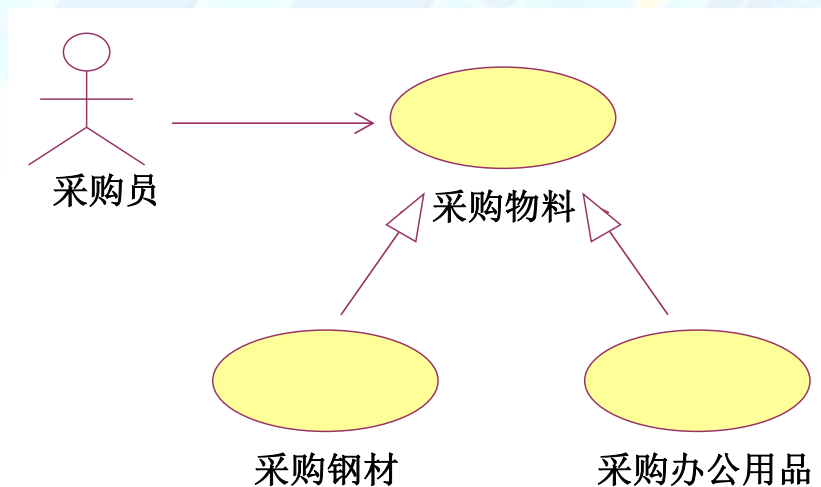
实际上相当于第一个用例的“备选流”

实际上相当于第一个用例的“备选流”



用例之间的关系：泛化(generalization)

- 当多个用例共同拥有一种类似的结构和行为的时候，可将它们的共性抽象成为父用例，其他的用例作为泛化关系中的子用例
- 子用例继承了父用例所有的结构、行为和关系





“用例的关系”

- 参与者与参与者之间的泛化(generalization)
- 用例和用例之间的包含(include)
- 用例和用例之间的扩展(extend)
- 用例和用例之间的泛化(generalization)关系
- 为什么要引入上述关系？有什么优越性？



需注意问题1：用例的粒度

- 用例识别的标准：actor与系统之间的一次独立交互
 - 如果多次交互总是同时发生且不会单独发生，可合并为一个用例
- 用例也可称为user story，将来使用时可看作一个独立存在的功能体
- 例如“查询/浏览宝贝或者店铺”，这个用例过大，拆分为：
 - 按类别浏览宝贝
 - 按关键字查询宝贝
 - 浏览宝贝的详细信息（商品特性、评价信息、已售出信息、etc）
 - 查询店铺
 - 浏览店铺的详细信息
- 例如“卖家对发布宝贝进行管理”：什么是“管理”？
 - 上架宝贝、下架宝贝、暂停销售
 - 修改宝贝描述、修改当前库存、设定价格等



需注意问题2：用例是actor与系统的交互

- 用例是actor与系统的交互：
 - actor对系统发出的请求
 - 系统对actor请求的响应
- actor与actor在现实当中的交互不应包含在use case中
 - 例如“讨价还价”：
并非系统提供的交互，而是二者在外部系统（旺旺）内完成的
 - 例如“处理纠纷”和“处理投诉”：
需要细化为淘宝客服与买家、卖家在淘宝系统中的操作/交互；若完全是人工操作，无需在系统中出现，只需要管理处理的结果即可



需注意问题3: actor与外部系统的区分

- 买家
 - 卖家
 - 淘宝平台管理员（小二）
 - 物流系统
 - 支付宝系统
 - 前三者是user，后二者是external system
-
- “淘宝平台”就是你所面对的系统，既不是user也不是外部系统
 - 系统自己做的事情，不是单独的用例：系统的行为受到actor的触发(可能是系统时钟)



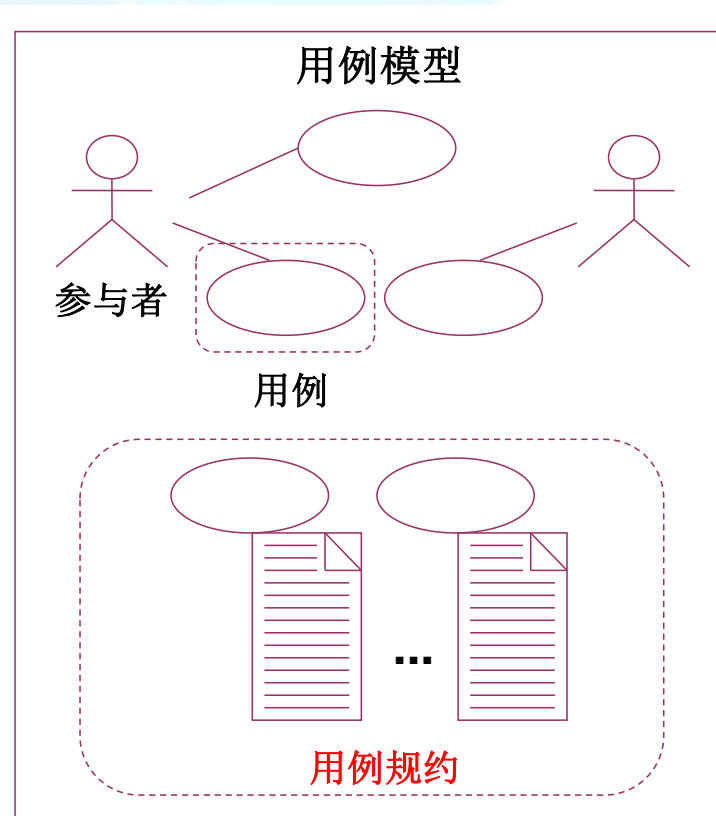
本章主要内容

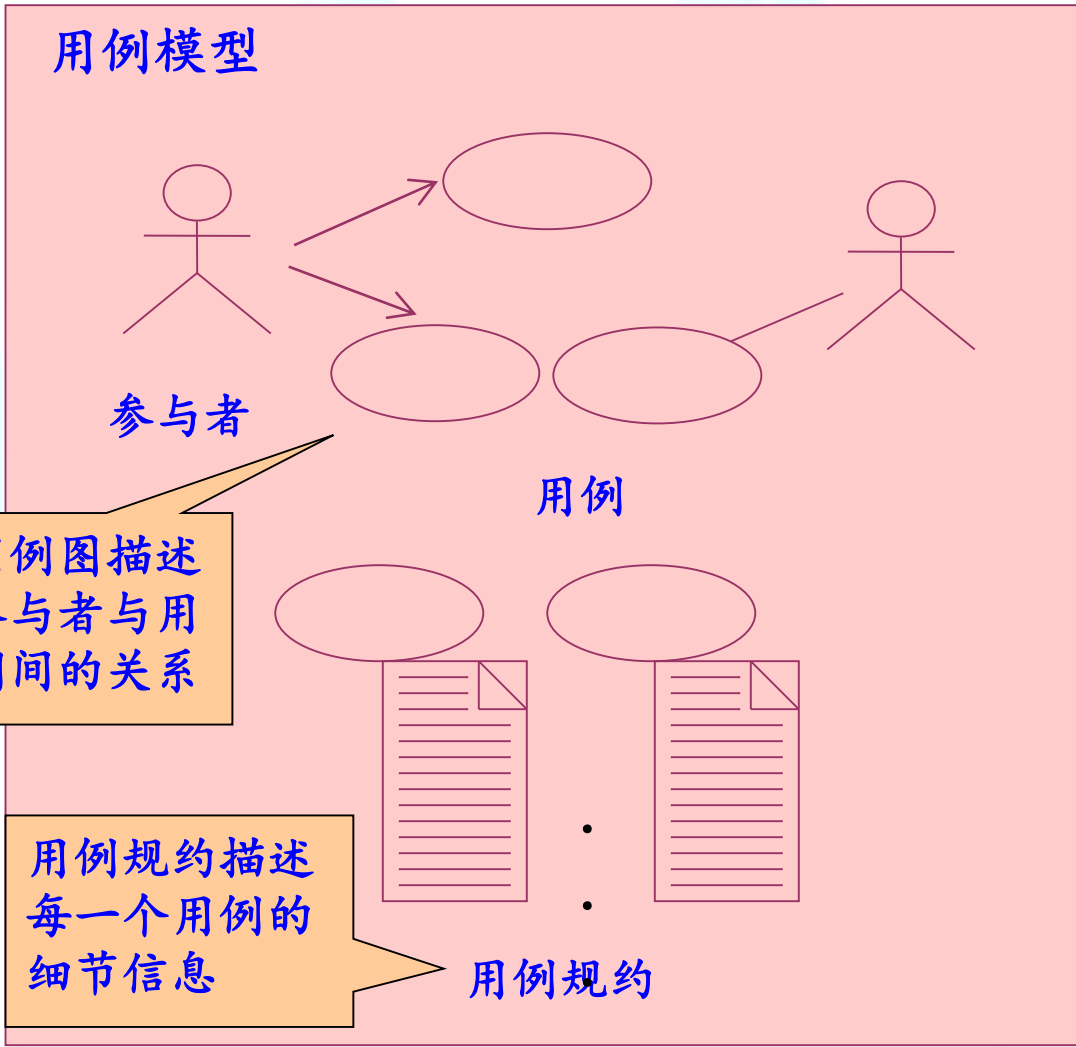
1. 敏捷开发中的“用户故事” (User Story)
2. 面向对象方法中的“用例” (Use Case)
3. 用例建模的基本过程
4. 用例模型的提交物
5. 活动图 & 泳道图



用例模型的提交物

- 1 用例模型
- 2 每个用例的详细描述
- 3 术语表：所用到的术语说明
- 4 补充规约：非功能性需求的说明





用例模型

参与者

用例

用例图描述参与者与用例间的关系

用例规约描述每一个用例的细节信息

用例规约

记录一些系统需求相关的术语

术语表

补充规约

记录一些全局性的功能需求、非功能性需求和设计约束等



本章主要内容

1. 敏捷开发中的“用户故事” (User Story)
2. 面向对象方法中的“用例” (Use Case)
3. 用例建模的基本过程
4. 用例模型的提交物
5. 活动图 & 泳道图

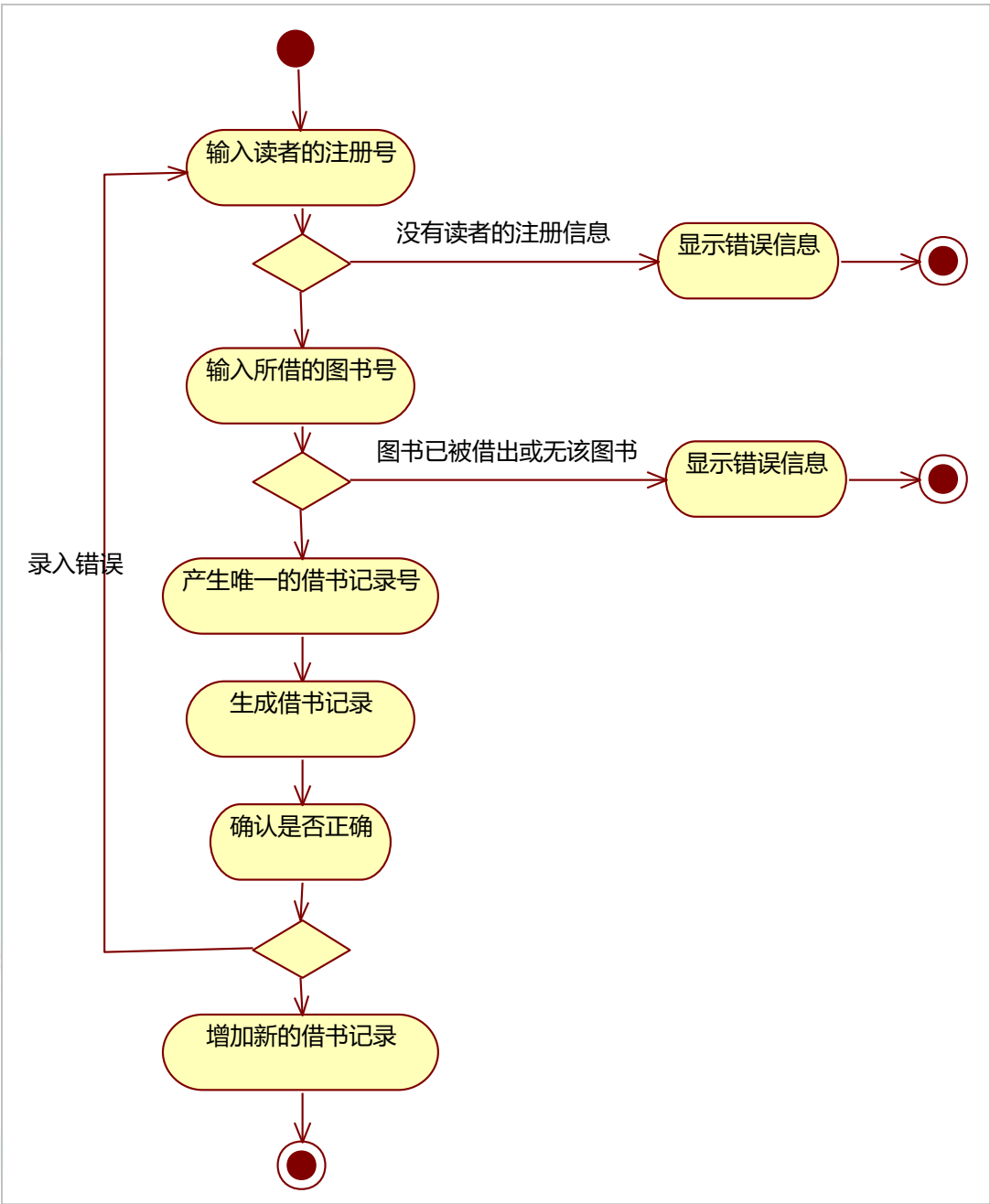


活动图 & 泳道图：对用例描述的图形化补充

- UML活动图(Activity Diagram)提供一种可视化的流程图方式，对use case的事件流进行直观展示，以便于读者更好的理解
- 同时，UML活动图也可以用来描述多个用例之间所形成的大粒度流程
- 两种形式：
 - 传统的活动图：只涉及一个参与者
 - 泳道图(swim-lane diagram)：侧重于描述多个参与者的活动之间的交互关系
- 活动图的基本要素：
 - 起始点、结束点
 - 活动；决策点
 - 活动之间的时序连接；活动之间的并发点
 - 泳道



UML活动图





UML泳道图

