用例图

基于用例的建模过程

- 获取原始需求
- 识别参与者
- 识别用例
- 识别用例之间的关系
- 描述脚本
- 构建用例图
- 进行用例描述

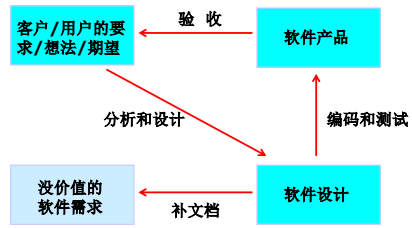
获取原始需求：石头问题

- 我要一块石头…
- 差不多，但我要小一点的…
- 很好，不过我要蓝色的…
- 啊，没有那么小…
- 咳，还是原来那个好了…

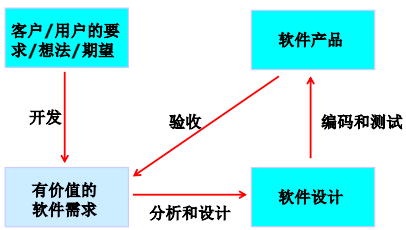
难捕获，易变！

小一点的蓝色大理石

获取原始需求：如此脆弱



获取原始需求：也需要开发



获取原始需求：技巧

技巧	描述
实地观察	直接观察个人工作的情况，以发现现存的实践方式和问题
访谈	从个人处收集特定信息
特定群体调查	对一组人员进行调查，以便了解工作态度和共同看法
问卷调查	收集详细数据和统计意义上比较重要的数据
用户指导	让最终用户告诉你，他们是如何操作系统的
原型制作	模拟一个无法直接测试的系统
统计版本	使用具有统计功能的应用程序来记录用户完成任务的方式

获取原始需求

目标：构建一个棋牌馆管理系统
问题描述：
客户通过Internet预订座位，检查座位详情，如果没有空闲的座位或满意的座位，可以选择进入等候队列。
总台服务员在客户到棋牌馆时，根据客户的预订信息，安排客户座位。
当客户要离开棋牌馆时，客户到总台服务员办理结账，可以采用两种方式，一种是现金结账，另一种是银行卡结账，而银行卡结账将通过与银联POS系统交互来完成。

填空题

- **需求分析的基本任务是_____。**

- **确定用户需要软件做什么**

简答题

- **简单描述软件需求所包含的三个层次的内容**

包含三个层次：业务需求、用户需求和功能需求。（2分）
业务需求反映了组织机构或客户对系统、产品高层次的目标要求，它们在项目视图与范围文档中予以说明。（1分）
用户需求描述了用户使用产品必须要完成的任务和具备的功能，这在使用实例文档或方案脚本说明中予以说明。（2分）

简答题

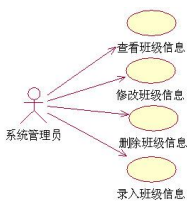
- **形成需求规格说明书的主要目的是什么?**

需求分析的主要目的为：
1) 为了方便需求分析小组共同讨论软件需求。(2分)
2) 作为下一步软件设计的基础（1分）
3) 作为软件测试的根据。（2分）

什么叫用例图

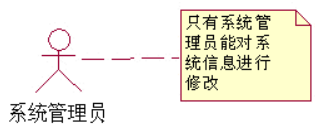
1. 用例图的含义

- 由参与者（Actor）、用例（Use Case）以及它们之间的关系构成的用于描述系统功能的动态视图称为用例图。要在用例图上显示某个用例，可绘制一个椭圆，然后将用例的名称放在椭圆的中心或椭圆下面的中间位置。
- 要在用例图上绘制一个参与者（表示一个系统用户），可绘制一个人形符号。参与者和用例之间的关系使用带箭头或者不带箭头的线段来描述，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。



什么叫用例图

- **在用例建模中，为了更加清楚的描述用例或者参与者，会使用到注释。**



什么叫用例图

2. 用例图的作用

- 用例图是需求分析中的产物，主要作用是描述参与者和用例之间的关系，帮助开发人员可视化的了解系统的功能。借助于用例图，系统用户、系统分析人员、系统设计人员、领域专家能够以可视化的方式对问题进行探讨，减少了大量交流上的障碍，便于对问题达成共识。
- 用例图可视化地表达了系统的需求，具有直观、规范等优点，克服了纯文字性说明的不足。
- 用例方法是完全从外部来定义系统功能，它把需求和设计完全的分离开来。我们不用关心系统内部是如何完成各种功能的，系统对于我们来说就是一个黑箱子。

用例图的构成要素

1. 参与者

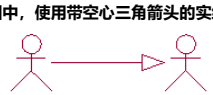
- 参与者（Actor）是指存在于系统外部并直接与系统进行交互的人、系统、子系统或类的外部实体的抽象。
- 每个参与者可以参与一个或多个用例，每个用例也可以有一个或多个参与者。
- 在用例图中使用一个人形图标来表示参与者，参与者的名字写在人形图标下面。

参与者名称

用例图的构成要素

2. 参与者间的关系

- 由于参与者实质上也是类，所以它拥有与类相同的关系描述，即参与者与参与者之间主要是泛化关系（或称为“继承”关系）。
- 泛化关系的含义是把某些参与者的共同行为提取出来表示成通用行为，并描述成超类。泛化关系表示的是参与者之间的一般/特殊关系，在UML图中，使用带空心三角箭头的实线表示泛化关系。

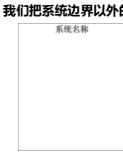


特殊参与者　　超类参与者

用例图的构成要素

3. 系统边界

- 在项目开发过程中，边界是一个非常重要的概念。这里说的系统边界是指系统与系统之间的界限。通常我们所说的系统可以认为是由一系列的相互作用的元素形成的具有特定功能的有机整体。
- 系统同时又是相对的，一个系统本身又可以是另一个更大系统的组成部分，因此，系统与系统之间需要使用系统边界进行区分开来。我们把系统边界以外的同系统相关联的其他部分，称之为系统环境。



识别参与者(actor)

对于一个大系统，难以列出所有用例的清单。此时，应先列出所有的参与者，然后在对每个参与者列出他所需的所有用例。即提供了一种获取用例的系统化过程。

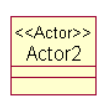
“参与者”（活动者、执行者）是指在系统之外，透过系统边界与系统进行有意义交互的任何事物。

识别参与者

UML中的Actor实际上是一个版型化的类，可以有三种表示形式

Actor1

Icon形式



Label形式

Decoration形式

识别参与者：参与者要点

- 系统外
  - 参与者代表在系统边界之外的真实事物，并不是系统的成分
- 系统边界
  - 参与者透过系统边界直接与系统交互，参与者的确定代表系统边界的确定
- 有意义交互
- 任何事物
  - 人、外部系统、外部因素等

识别参与者：参与者要点

- 参与者指在系统中所扮演的角色。即在确定参与者时，应主要考虑他的角色，而不是这个角色的实例。
  - 某些组织中可能有很多营销人员，但他们均起着同一种作用，扮演着相同的角色。
  - 一个用户也可以扮演多种角色：一个高级营销人员既可以是贸易经理，也可以是普通的营销人员。
- 一个参与者可以执行多个用例。
- 一个用例也可以由多个参与者使用。

识别参与者：任何事物

- 参与者不仅可以由人承担，还可以是其它系统、硬件设备、甚至是时钟
  1）其它系统：当系统需要与其它系统交互时，如ATM柜员机系统中，银行后台系统就是一个参与者；
  2）硬件设备：如果系统需要与硬件设备交互时，如在开发IC卡门禁系统时，IC卡读写器就是一个参与者；
  3）时钟：当系统需要定时触发时，时钟就是参与者

思考：识别参与者？

- 寻呼台系统：用户如果预定了天气预报，系统每天定时给他发天气消息；如果当天气温高于35度，还要提醒用户注意防暑；

在这个叙述里，谁是寻呼台系统的Actor？
用户？气温？时间？

时间作为参与者，一种习惯用法，用于激活那些系统定期的、自动执行的用例

识别参与者：参与者与系统边界

系统边界的确定就是要确定我们要开发的系统和外部环境之间的界限，也就是要区分系统本身和它的外部环境。

- 某企业要求开发一个企业信息管理系统，并与原来已有的库存系统相连接
- 某企业要求开发一个企业信息管理系统，并把原来已有的库存管理系统加以改造，成为企业信息管理系统的一部分

思考：系统边界？

一个银行系统，它的系统边界如何确定呢？

- 银行系统的外部活动者有储户、前台出纳员、银行管理员，这些都不属于银行系统本身，他们是此系统的外部环境；
- 银行系统要打印交易凭条，打印机对于系统来说是外部环境；
- 银行系统可能与客户的工作单位的工资发放系统有交互，那么客户工作单位的工资发放系统也是外部环境。
- 而对于银行系统来说，使用此系统的银行的建筑格局、人员构成、所处地域等就是此系统的内部环境。

识别参与者：确定系统边界的作用

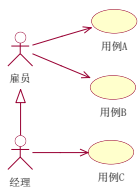
- 系统边界一确定，我们就已经知道有哪些外部对象在与系统进行交互，于是我们就可以在系统中为该对象设计相应的接口，从而实现这些交互。
- 如果这些外部环境改变了，我们可能要重新设计我们的接口。但不在系统边界上的因素我们就不用考虑。

识别参与者：技巧

- 谁使用系统的主要功能
- 谁改变系统的数据
- 谁从系统获取信息
- 谁需要系统的支持以完成日常工作任务
- 谁负责日常维护、管理并保证系统正常运行
- 系统需要应付（处理）那些硬设备
- 系统需要和那些外部系统交互
- 谁（或什么）对系统运行产生的结果（值）感兴趣
- 时间、气温等内部外部条件
- ……

识别参与者：参与者的泛化

- 参与者的泛化表示一个一般性的参与者（父参与者）与另一个更为特殊的参与者（子参与者）之间的联系。
- 子参与者继承了父参与者的行为和含义，还可以增加自己特有的行为和含义，子参与者可以出现在父参与者能出现的任何位置上。
  - 如系统中经理可以参加雇员的所有用例



识别参与者：棋牌馆管理系统

目标：构建一个棋牌馆管理系统
问题描述：
客户通过Internet预订座位，检查座位详情，如果没有空闲的座位或满意的座位，可以选择进入等候队列。
总台服务员在客户到棋牌馆时，根据客户的预订信息，安排客户座位。
当客户要离开棋牌馆时，客户到总台服务员办理结账，可以采用两种方式，一种是现金结账，另一种是银行卡结账，而银行卡结账将通过与银联POS系统交互来完成。

识别用例（use case）

分析典型用例是开发者准确迅速地了解用户要求的最常用也是最有效的方法，是用户和开发者一起深入剖析系统功能需求的起点。

“用例”是Ivar Jacobson于20世纪60~70年代在爱立信公司开发AKE、AXE系列时发明的。

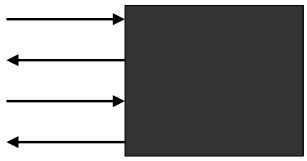
“Object-oriented software engineering: a use case driven approach”

- 用例实例是在系统中执行的一系列动作，这些动作将生成特定参与者可见的价值结果。
- 一个用例定义一组用例实例，用例实例也就是常说的“使用场景”，就是用户使用系统的一个实际的、特定的场景

识别用例：用例要点

- 可观测→用例止于系统边界
- 价值结果→用例是有意义的目标
- 系统执行→结果值由系统生成
- 由参与者观测→业务语言、用户观点
- 一组用例实例→用例的粒度

用例要点：用例止于系统边界

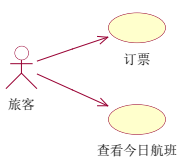


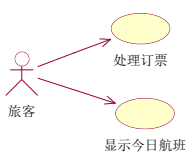
在系统外部描述与系统功能的交互，而不考虑系统内部对该功能的实现方式。

用例要点：业务语言而非技术语言

- **用户词汇，而不是技术词汇**
  - **如：发票，商品，洗衣机**
  - **而不是：记录，字段，COM，C++等**

用例要点：用户观点而非系统观点





用户观点 **系统观点**

用例 VS. 功能

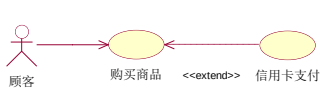
- **呼叫某人**
- **接听电话**
- **发送短信**
- **记住电话号码**
- **……**

- **传输/接收**
- **电源/基站**
- **输入输出（显示、键盘）**
- **电话簿管理**
- **……**

用户观点 **系统观点**

用例的命名

- **执行者视角：**
  - **（状语）动词+（定语 ）宾语**



用例的获取

找出用例的最简单途径是对参与者提问，然后从答案中获取用例:

- **参与者的主要任务是什么?**
- **参与者需要了解系统的什么信息? 需要修改系统的什么信息?**
- **参与者是否需要把系统外部的变化通知系统?**
- **参与者是否希望系统把异常情况通知自己?**

用例粒度

用例的粒度指的是用例所包含的系统服务或功能单元的多少。用例的粒度越大，用例包含的功能越多，反之则包含的功能越少。

如果用例的粒度很小，得到的用例数就会太多。反之，如果用例的粒度很大，那么得到的用例数就会很少。

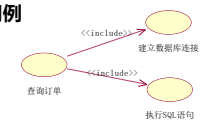
如果用例数目过多会造成用例模型过大和引入设计困难大大提高。如果用例数目过少会造成用例的粒度太大，不便于进一步的充分分析。

用例粒度

最常犯错误：粒度过细，陷入功能分解

- **把步骤当作用例**

- **把系统活动当作用例**



识别用例：棋牌馆管理系统

目标：构建一个棋牌馆管理系统
问题描述：
客户通过Internet预订座位，检查座位详情，如果没有空闲的座位或满意的座位，可以选择进入等候队列。
总台服务员在客户到棋牌馆时，根据客户的预订信息，安排客户座位。
当客户要离开棋牌馆时，客户到总台服务员办理结账，可以采用两种方式，一种是现金结账，另一种是银行卡结账，而银行卡结账将通过与银联POS系统交互来完成。

用例之间的关系

Generalization

泛化关系中，子用例继承父用例的行为和含义，子用例也可以增加新的行为和含义或覆盖父用例中的行为和含义

<<include>> **Include**

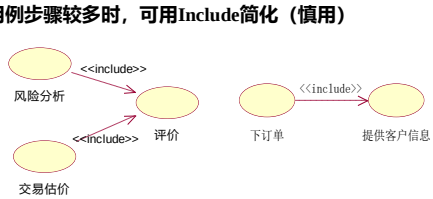
一个用例（称作基本用例）的行为包含了另一个用例（称作包含用例）的行为

<<extend>> **Extend**

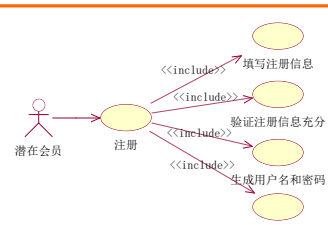
扩展关系比泛化关系用更多的规则限制，基础用例提供扩展点，扩展用例只能在这些扩展点上增加新的行为。

包含关系

- **某些步骤在多个用例重复出现，且单独形成价值**
- **被包含的用例不是孤立存在的，它仅作为某些包含它的更大的基用例的一部分出现**
- **用例步骤较多时，可用Include简化（慎用）**



包含关系的误用



- **包含关系使用不当容易诱使人们进行功能分解，从而导致对用例的误用**

扩展关系

- 常规动作放在一个基本的用例中，将非常规动作放在它的扩展用例中。
- 基本用例是可以独立于扩展用例存在的，只是在特定的条件下，它的行为可以被另一个用例的行为所扩展 。
- 扩展用例通过引用扩展点（extension point）建立与基用例的联系，扩展点指明了在基本用例中的扩展位置

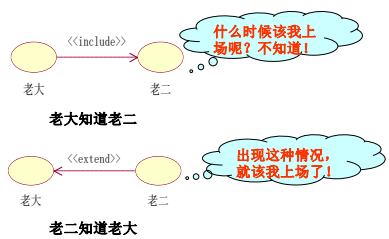
扩展VS包含

- 包含：由用例A连向用例B，表示用例A中使用了用例B中的行为或功能
  - 一个基本用例执行时，一定会执行包含用例的部分。
- 扩展：由用例B连向用例A，表示用例A描述了一项基本需求，而用例B则描述了该基本需求的特殊情况，即一种扩展
  - 扩展用例的目的是在不改变某个已存在（或假定存在）的用例的前提下为之增添新行为。
  - 一个基本用例执行时，可以执行、也可以不执行扩展部分。

扩展VS包含

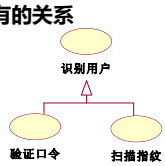
- 扩展和包含用例本质上其实非常相似，都表示从基本用例中抽取一些行为放到一个单独的用例中。
  - 扩展和包含用例都与基本用例相联。在基用例的执行过程中，可能在某种条件下基本用例的执行被中断，转而执行扩展或包含用例（附加用例）。当附加用例执行完毕，控制将返回到基用例原来被中断的那个位置恢复执行。
- 它们的主要区别在于用例实例中断基本用例、执行附加用例的方式
  - 包含用例一定会执行，扩展用例只有在特殊情况下才能执行。

扩展VS包含

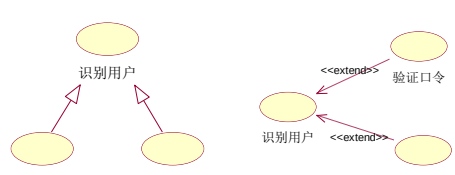


泛化关系

- 同一业务目的不同技术实现
  - 一个用例可以特化另一个更普通用例（更普通用例泛化特殊用例）
  - 用例间的泛化关系表明子用例包含父用例中定义的所有属性、行为序列和扩展点，并且参与父用例中所有的关系

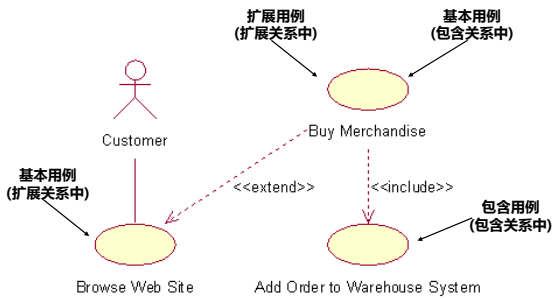


扩展VS泛化



采用不同关系，文档结构不同

用例之间的关系：网上购物的部分用例



用例之间的关系：几种关系的符号

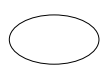
关系类型	说明	表示符号
关联	actor与use case之间	
泛化	actor之间或use case之间	
包含	use case之间	<<include>>
扩展	use case之间	<<extend>>

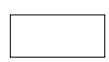
用例之间的关系：用例关系的应用

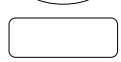
- **当描述一般行为的某种变化时，采用泛化关系。**
- **当描述一般行为的某种变异且希望通过基用例中的扩展点来加以控制时，则应采用扩展关系。**
- **当两个或更多的用例中出现重复描述而又想避免这种重复时，采用包含关系。**

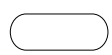
选择题

- **下面哪个图形代表用例（ ）**

A. 

B. 

C. 

D. 

- **选A**

选择题

- **用例与用例之间的关系可以是（ ）**
  - **A. 包含关系**
  - **B. 泛化关系**
  - **C. 关联关系**
  - **D. 继承关系**

- **选A、B**

选择题

- **执行者（Actor）与用例之间的关系是（ ）**
  - **A. 包含关系**
  - **B. 泛化关系**
  - **C. 关联关系**
  - **D. 扩展关系**

- **选C**

选择题

- **用例（Use-case）用来描述系统在事件做出响应时所采取的行动。用例之间是具有相关性的。在一个“订单输入子系统”中，创建新订单和更新订单都需要检查用户帐号是否正确。那么，用例“创建新订单”、“更新订单”与用例“检查用户帐号”之间是（ ）关系**

A．包含（include）

B．扩展（extend）

C．分类（classification）

D．聚集（aggregation）

- **选A**

55

填空题

- **用例图两个最核心的元素是_____与用例。**

- **参与者**

56

简答题

- **简述用例模型的组成元素以及建模步骤**

组成元素有：参与者，用例，通信关联，用例规约 （每个0.5）
建立用例模型步骤：
1) 根据系统边界确定参与者（1分）
2) 根据每个参与者确定与之相关联的用例（1分）
3) 对每个用例写出用例规约文档，调整用例模型（优化参与者与参与者之间的关系，用例与用例之间的关系）（1分）

57

简答题

- **为什么说UML是用例图驱动的开发方法，试论述UML用例图的用途。**

因为用例代表系统中各个相关人员之间就系统的行为所达成的契约，一个软件的开发就是从分析这些工作开始的。软件的开发过程分为需求分析、设计、实现、测试等阶段，用例把所有这些都捆绑到一起，用例分析的结果也为预测系统的开发时间和预算提供依据，保证项目的顺利进行，从各个方面综合起来讲软件开发是用例驱动的。

58

简答题

- **什么是用例图？用例图有什么作用？**

定义：由参与者（Actor）、用例（Use Case）以及它们之间的关系构成的用于描述系统功能的图成为用例图。（2分）
作用：用例图是从软件需求分析到最终实现的第一步，它显示了系统的用户和用户希望提供的功能，有利于用户和软件开发人员之间的沟通（1分）。用例图可视化的表达了系统的需求，具有直观、规范等优点，克服了纯文字性说明的不足（1分）。用例方法是完全从外部来定义系统的，它把需求和设计完全分离开来（1分），使用户不用关心系统内部是如何完成各种功能的。

59

习题

在一个销售系统中，存在两类用户：客户和销售人员。存在三个用例：网上购物、销售统计、发货。其中客户只和“网上购物”交互，而销售人员则和“销售统计”、“发货”用例交互。请根据这个描述画出用例图。

60

习题

画出简化的文本编辑程序的用例图，该编辑程序的主要功能有，建立文件、打开文件、插入文本、修改文本和保存文件。

脚本

脚本(scenario)在UML中指贯穿用例的一条单一路径, 用来显示用例中的某种特殊情况.

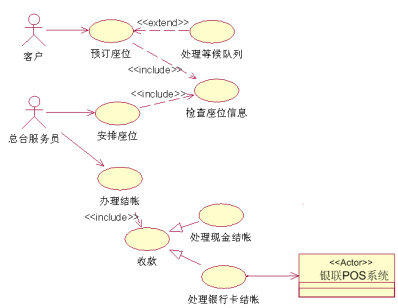
其它译名: 情景、场景、情节、剧本.

每个用例有一系列脚本, 包括一个主要脚本, 以及几个次要脚本. 相对于主要脚本, 次要脚本描述了执行路径中的异常或可选择的情况.

例：在“订货”用例中包括几个相关脚本：

- **订货顺利进行的脚本;**
- **相关货源不足时的脚本;**
- **购货者的信用卡被拒绝时的脚本;**
- **……**

构建用例图



用例描述

用例描述是指对一个用例的功能进行的文字描述, 是参与者与系统交互动作序列的说明.

用例描述才是用例的主要部分, 是后续的交互图分析和类图分析必不可少的部分.

用例采用自然语言描述参与者与系统的交互行为,要易于理解. 其读者是开发人员、用户、项目经理、测试人员等.

用例描述：用例描述的内容

- **用例的目标**
- **用例是怎么启动的**
- **参与者与用例之间的消息如何传送**
- **用例中除了主路径外, 其它路径是什么**
- **用例结束后系统的状态**
- **其它需要描述的内容**

描述用例时的原则是尽可能写得“充分”，而不是形式化、完整或漂亮.

用例描述：用例的描述格式

描述项	说明
用例名称	表明用户的意图或用例的用途
标识符[可选]	惟一标识符, 便于引用该用例
用例描述	概述用例的几句话
参与者	与此用例相关的参与者
优先级	一个有序的排列, 1代表优先级最高
状态[可选]	用例状态, 可以是: 进行中, 等待审查, 通过审查, 未通过审查
前置条件	一个条件列表, 这些条件必须在访问用例前得到满足
后置条件	一个条件列表, 这些条件必须在用例完成之后得到满足
基本操作流程	描述用例中各项工作都顺利进行时用例的工作方式
可选操作流程	描述变异工作方式、出现异常或发生错误的情况下的路径

用例的描述格式(续表)

描述项	说明
被泛化的用例	此用例所泛化的用例列表
被包含的用例	此用例所包含的用例列表
被扩展的用例	此用例所扩展的用例列表
修改历史记录[可选]	关于用例的修改时间、修改原因、修改人的详细信息
问题[可选]	与此用例的开发有关的问题列表
决策[可选]	关键决策的列表, 将这些决策信息记录下来以便维护时使用
频率[可选]	参与者访问此用例的频率, 如: 每日一次/每月一次等

67

用例描述：描述用例时易出现的错误

- 只描述系统的行为, 没有描述参与者的行为
- 只描述参与者的行为, 没有描述系统的行为
- 在用例描述中就设定了对用户界面的设计的要求
- 描述过于冗长

68

ATM系统“取款”用例的两个错误描述：

Use case: Withdraw cash
Actor: customer
主事件流：
(1) 储户插入ATM卡,并输入密码
(2) 储户按“取款”按钮,并输入取款数目
(3) 储户取走现金/ATM卡/收据
(4) 储户离开

只描述了actor的行为

Use case: Withdraw cash
Actor: customer
主事件流：
- ATM系统获得ATM卡和密码
- 设置交易类型为“取款”
- ATM系统获得取款金额
- 输出现金、收据和ATM卡
- 系统复位

只描述了System的行为

69

ATM系统“取款”用例的正确描述

Use case: Withdraw cash
Actor: customer
主事件流：
- 储户通过读卡机插入ATM卡
- ATM系统从卡上读取银行ID、账号、加密密码,并通过主银行系统验证银行ID和账号
- 储户输入密码,ATM系统根据加密密码对输入密码进行验证
- 储户按“取款”按钮,并输入取款数目,该数目应该为$5的倍数
- ATM系统通知主银行系统,传递账号和金额,并接收返回的确认信息和账户余额
- ATM系统输出现金、ATM卡和收据
- ATM系统记录交易到日志文件

70

用例描述：前置、后置条件-1

- 前置条件约束在用例开始前系统的状态
  - 把它们看做是看门人，它阻止参与者触发该用例直到满足所有条件
  - 说明在用例触发之前什么必须为真
- 后置条件约束用例执行后系统的状态
  - 用例执行后什么必须为真
  - 对于有多个操作流的用例，则应该有多个后置条件

71

用例描述：前置、后置条件-2

- 某些用例依赖于其他用例
  - 一个用例在离开系统时，可能是另一个用例的前置条件（例如：“登录”和“管理系统”）
- 有助于识别漏掉的用例
  - 如果一个用例的前置条件不能有执行其他用例满足，可能意味着丢失了用例（例如：“管理订单”却没有“登录”用例）

72

用例描述示例

1.用例名称：处理银行卡结帐
2.标识符：
3.用例描述：客户来到付款处，总台服务员记录客户离开信息并接受付款，付款完成后，客户离开。
4.参与者：总台服务员，POS系统
5.前置条件：客户退出旗牌桌。
6.后置条件：无
7.基本操作流
8.可选操作流
4.非功能需求
7.扩展点：无
8.优先级：最高（满意度5，不满意度5）

用例描述示例

基本操作流
1.系统显示客户的消费总额。
2.总台服务员接收客户的银行卡。
3.客户输入密码，POS系统对密码进行验证。
4.POS系统返回确认消息。
5.系统打印付款收据
6.总台服务员将银行卡和打印付款收据交给客户
7.系统记录本次交易
8.客户离开

可选操作流
第2步：如果输入的密码不正确，系统显示出错信息
第7步：客户没有足够的现金，则系统显示出错信息，付款不成功。

用例描述：操作流描述要点

- 只书写“可观测”的（说人话）
- 使用主动语句
- 句子必须以参与者或系统作为主语
- 不要涉及界面细节
- 分支和循环

使用Rose创建用例的步骤说明

1. 需求分析

- “学生信息管理系统”部分功能性需求包括以下内容：
(1) 系统管理员登录后可以对班级的基本信息进行增加、删除、修改、查询等操作。学校领导登录后可以对班级基本信息进行查询操作。
(2) 教师登录后可以对学生的考试成绩进行录入、删除、修改、查询等操作。学生登录后可以对考试成绩进行查询操作。
(3) 学生登录后可以了解所有选修课程的具体信息，可以根据自己的需要选择不同课程。系统管理员登录后可以增加、修改、查询、删除选修课程。
(4) 系统管理员可以对账号进行创建、设置、查看、删除等操作。

使用Rose创建用例的步骤说明

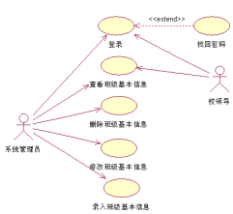
2. 识别参与者

- 对于一个学校来说，最重要的就是教育学生成才，所以我们首先要考虑到的参与者就是学生。
- 要给学生上课，必然就需要教师。教师负责教育学生、并且在日常管理中可以查询学生的基本信息、查询学生的考试成绩。
- 作为一个学校，除了教师和学生，还有不可或缺的就是校领导。为了便于校领导掌握学校的基本情况，加强对学校的管理导.
- 不管什么系统，基本都会有比较专业的人员来负责管理系统，本系统也不例外，系统管理员除了负责维护系统的日常运行，还要进行录入学生基本信息、维护选课信息等工作。

使用Rose创建用例的步骤说明

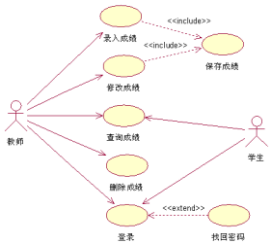
3. 构建用例模型

- 系统管理员直接参与的用例为登录、找回密码、查看班级基本信息、删除班级基本信息、修改班级基本信息和录入班级基本信息。校领导直接参与用例登录、找回密码和查看班级基本信息。当登录过程中出现忘记密码的情况，就需要使用找回密码的功能来找回密码，而在正常情况下用户不到找回密码这个功能所以用例找回密码”和用例登录之间是扩展关系。



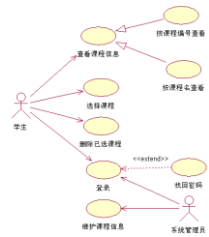
使用Rose创建用例的步骤说明

- 教师参与用例录入成绩、修改成绩、保存成绩、查询成绩、删除成绩和登录。学生参与用例登录和查询成绩。因为修改成绩和录入成绩的时候都要保存成绩，所以将保存成绩抽象出来作为单独的一个用例。用例录入成绩、修改成绩和用例保存成绩之间是包含关系，用例找回密码和用例登录之间是扩展关系。



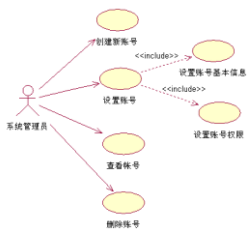
使用Rose创建用例的步骤说明

- 学生作为参与者直接参与用例查看课程信息、按课程编号查看、按课程名查看、选择课程、删除已选课程、登录和找回密码。系统管理员参与用例登录、找回密码和维护课程信息。其中查看课程信息有两种方式，一种是按照课程名查看，另一种是按照课程编号查看，所以查看课程信息是父用例，而按照课程名查看和按照课程编号查看是子用例，他们之间的关系是泛化关系。用例找回密码和用例登录之间是扩展关系。



使用Rose创建用例的步骤说明

- 系统管理员参与用例创建新账号、设置账号、设置账号基本信息、设置账号权限、查看账号和删除账号。在设置帐号时，主要分为设置账号的基本信息和设置账号的权限，为了便于修改和维护，将这两个功能分别抽象为两个用例。所以用例设置账号基本信息、设置账号权限和用例设置账号之间是包含关系。



练习题

- 网络的普及带给了人们更多的学习途径，随之用来管理远程网络教学的“远程网络教学系统”也诞生了。
- “远程网络教学系统”的功能需求包括：

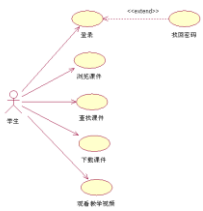
(1)学生登录网站后，可以浏览课件、查找课件、下载课件、观看教学视频。

(2)教师登录网站后，可以上传课件、上传教学视频、发布教学心得、查看教学心得、修改教学心得。

(3)系统管理员负责对网站页面的维护，审核不法课件和不法教学信息，批准用户注册。

练习题

(1) 学生需要登录“远程网络教学系统”后才能正常使用该系统所有功能。如果忘记密码，可以通过“找回密码”功能找回密码。登录后学生可以浏览课件、查找课件、下载课件、观看教学视频，请画出学生参与者的用例图。



练习题

(2) 教师登录“远程网络教学系统”后可以上传课件、上传教学视频课件、发布教学心得、修改教学心得。如果忘记密码，可以通过“找回密码”功能找回密码。请画出教师参与者的用例图。