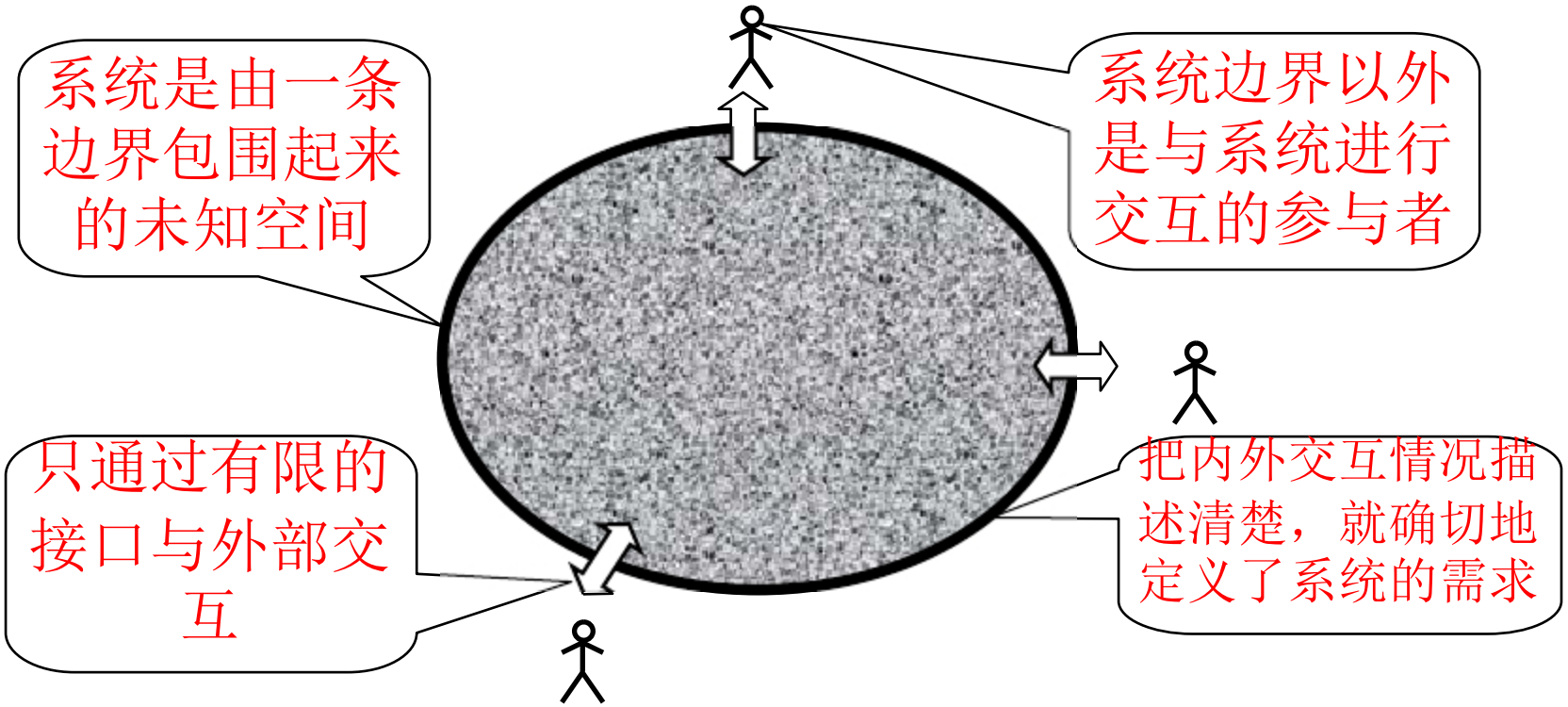
第5章　用况图

本章的主要概念—系统边界、参与者、用况、包含、扩展、泛化

问题的提出：在系统尚未存在时，如何描绘用户需要一个什么样的系统？如何规范地定义用户需求？

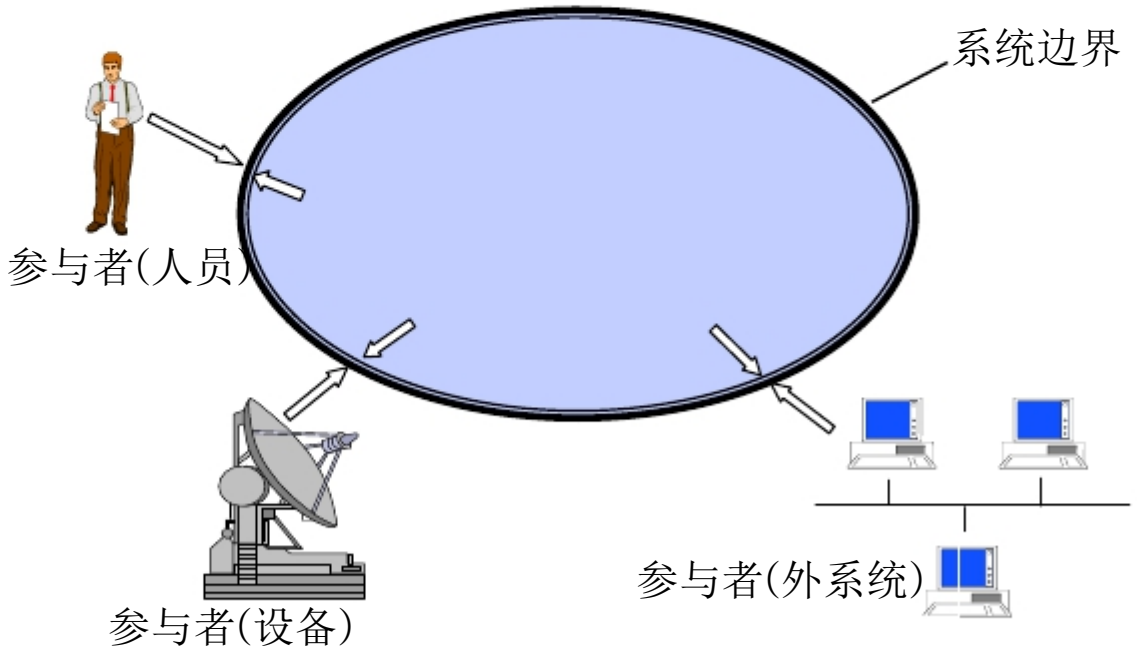
考虑问题的思路：把系统看作一个黑箱，看它对外部的客观世界发挥什么作用，描述它外部可见的行为。



5.1 系统边界

系统边界：一个系统所包含的所有系统成分与系统以外各种事物的分界线。

参与者：在系统边界以外，与系统进行交互的事物——人员、设备、外系统



现实世界中的事物与系统的关系包括如下几种情况：

■某些事物位于系统边界内，作为系统成分。如超市中的商品，抽象为“超市商品销售系统”内的“商品”对象。

■ 某些事物位于系统边界外，作为参与者。

■某些事物可能既有一个对象作为其抽象描述，而本身（作为现实世界中的事物）又是在系统边界以外与系统进行交互的参与者。如超市中的收款员，他本身是现实中的人，作为参与者；在系统边界内，又有一个相应的“收款员”对象来模拟其行为或管理其信息，作为系统成分。

■某些事物即使属于问题域，也与系统责任没有什么关系。如超市中的保安员，在现实中与超市有关系，但与所开发的系统超市商品管理系统无关系。这样的事物既不位于系统边界内，也不作为系统的参与者。

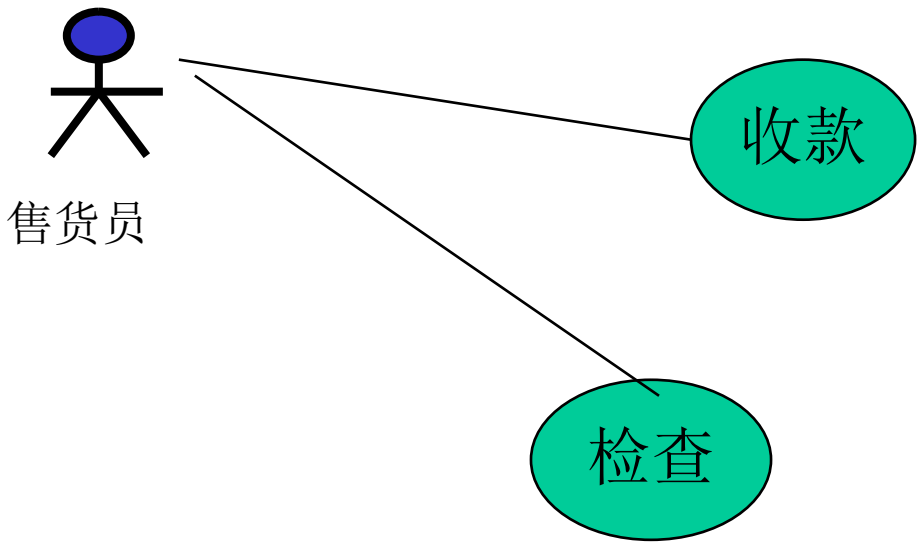
认识清楚上述事物之间的关系，也就划分出了系统边界。

5．2 参与者

简言之，参与者是在系统之外的与系统进行交互的任何事物。

5．2．1 概念与表示法

- 一个参与者定义了用况的使用者在与这些用况交互时所扮演的一组功能高内聚的角色。



- 参与者可以向系统发出请求
- 参与者回复系统的响应

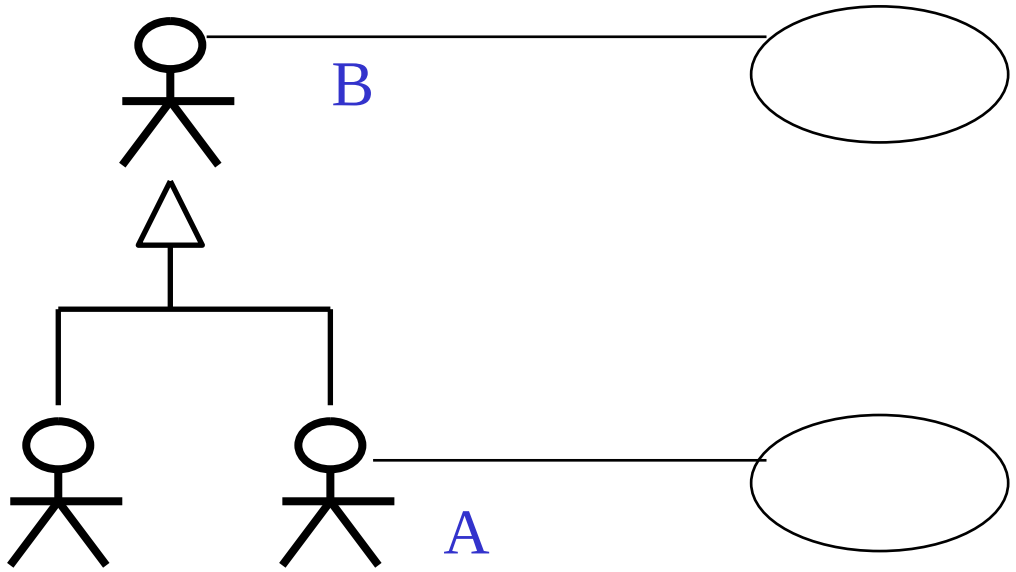
参与者和系统之间可能存在着复杂的对话

所有的参与者与系统间的请求/响应确定了系统边界

一个参与者的一个实例，要以一种特定的方式与系统进行交互。

尽管在模型中使用参与者，但参与者实际上并不是系统的一部分。它们存在于系统之外。

如果一组参与者具有共同的性质，可以把这些性质抽取出来放在另一个参与者中，它们再从中继承，把这种关系称为参与者之间的泛化关系。



从参与者**A**到参与者**B**之间的泛化关系是指，**A**的实例除了拥有自己的与系统的交互外，还拥有**B**的实例与系统进行的交互。

5．2．2 识别参与者

■ 用户

从直接使用系统的人员中发现参与者。

这里强调的是直接使用，而不是间接的。

特定的人，在系统中可扮演不同的角色。

———例如，添加数据、统计某项数据及产生汇总表的那个人就扮演了三种不同的角色，反映为三种不同的参与者。

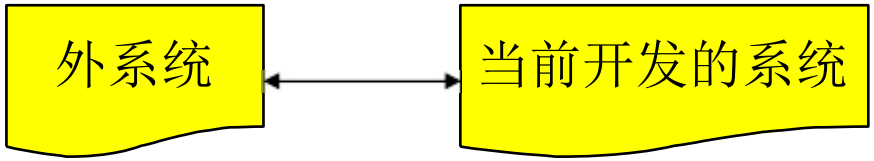
———例如，用户角色的类别可为：目标终端用户、管理员、经理或顾客。

■ 外部系统

所有与系统交互的外部应用系统都是参与者。

从系统边界的角度，应该把与当前所开发的软件系统一起运行以完成特定任务的其它应用系统，看作是它的外部的应用。

相对于当前在正在开发的系统而言，外部应用系统可以是其他子系统、上级系统、下级系统或任何与它进行协作的系统，但对它的开发并不是当前系统的开发小组的责任。



■ 设备

识别所有与系统交互的设备。

这样的设备与系统相连，向系统提供外界信息，或在系统的控制下运行。如外部传感器（输入信息）和受控马达（输出信息）。

通常，不包括监视器、键盘、鼠标和其它的标准的用户接口类型设备。

总结：如何发现参与者？

人员——

- 系统的直接使用者
- 直接为系统服务的人员

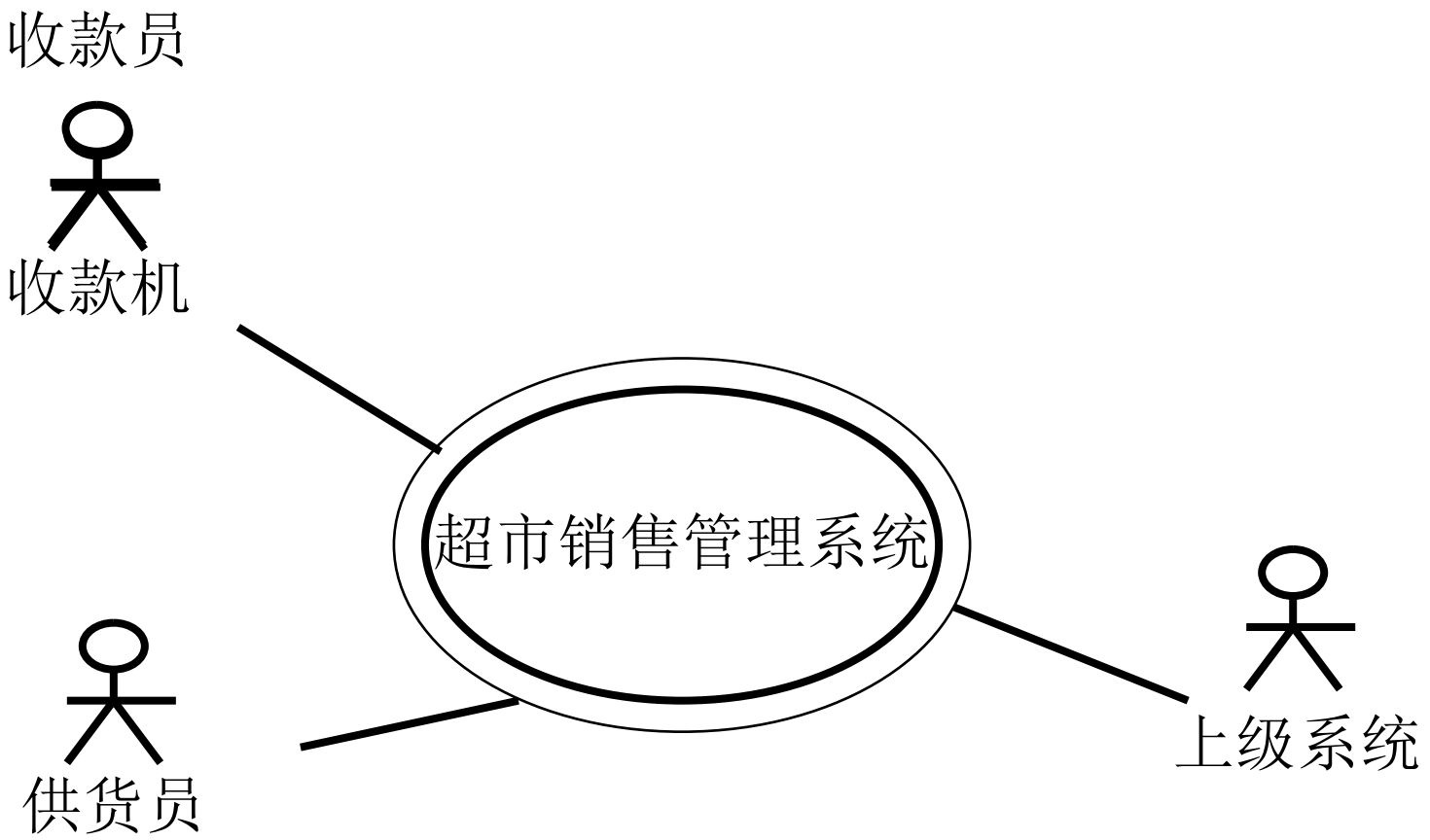
设备——

- 与系统直接相联的设备
  - 为系统提供信息
  - 在系统控制下运行
- 不与系统相联的设备 ×
- 计算机设备 ×

外系统——

- 上级系统
- 子系统
- 其它系统

参与者：收款员、供货员、导购员、经理、保安、顾客、
收款机、上级系统？



参与者：收款员、供货员、导购员、经理、保安、顾客、收款机、上级系统？

5．3用况

1 定义及表示法

用况是对参与者使用系统的一项功能时所进行的交互过程的一个文字描述序列。

用况示例 1

表示形式：

名称（参与者，功能）
行为描述
控制语句
括号或标号

例

```
收款员收款
输入开始本次收款的命令;
                作好收款准备，应收款总
                数置为0，输出提示信息;
for    顾客选购的每种商品    do
   输入商品编号;
   if    此种商品多于一件    then
      输入商品数量
   end  if;
                检索商品名称及单价;
                货架商品数减去售出数;
                if    货架商品数低于下限then
                   通知供货员请求上货
                end  if;
                计算本种商品总价并打印编号、
                名称、数量、单价、总价;
                总价累加到应收款总数;
end  for;
                打印应收款总数;
输入顾客交来的款数;
                计算应找回的款数，
                打印以上两个数目，
                应收款数计入帐册。
```

几点说明：

（1）一个用况描述参与者对一项系统功能的使用情况。而且只有当外部的参与者与该系统进行交互时，该功能才发挥作用。

（2）用况中描述的行为实际上是系统级的。在用况内所描述的交互中的动作应该是详细的，准则是对用况的理解不产生歧义即可;若描述得过于综合，则不易认识清楚系统的功能。

（3）陈述参与者和系统在交互过程中双方所做的事。而且描述彼此为对方直接地做什么事，不描述怎么做，内部细节不要在其中描述。

（4）用况既表达了系统的功能需求，也表达了系统的功能划分。

（5）描述应力求准确、清晰，允许概括，但不要把双方的行为混在一起。

（6）系统执行该动作序列来为参与者产生一个可观察的结果值。

（7）用况描述的是一个参与者所使用的一项系统功能，该项功能应该相对完整。这就要求一个用况描述的功能，即不能过大以至于包含过多的内容，也不能过小以至于仅包含完成一项功能的若干步骤。

（**8**）用况描述中的一个步骤应该描述且仅描述参与者或系统要完成的一件事情。

用况示例 2

在ATM 系统中，描述一个用况“验证用户”：

基本流：在系统提示顾客输入PIN编号时，用况开始。顾客通过按键输入PIN号。顾客按“输入”按钮确认登录。系统校验这个PIN 号是否有效。如果有效，系统承认这次登录，该用况结束。

可选流：顾客可以在任何时间通过按“取消”按钮取消一个事务，这样，该用况重新开始。顾客的账户未发生改变。

可选流：顾客可以在确认之前的任何时刻清除PIN号，并重新输入一个新的PIN号。

可选流：如果顾客输入了一个无效的PIN 号，用况重新开始。如果连续3 次无效，系统将取消整个事务，并在60 秒内阻止该顾客与ATM 进行交易。

用况的一个场景是指用况执行过程中的一个特定的实例或执行路径。

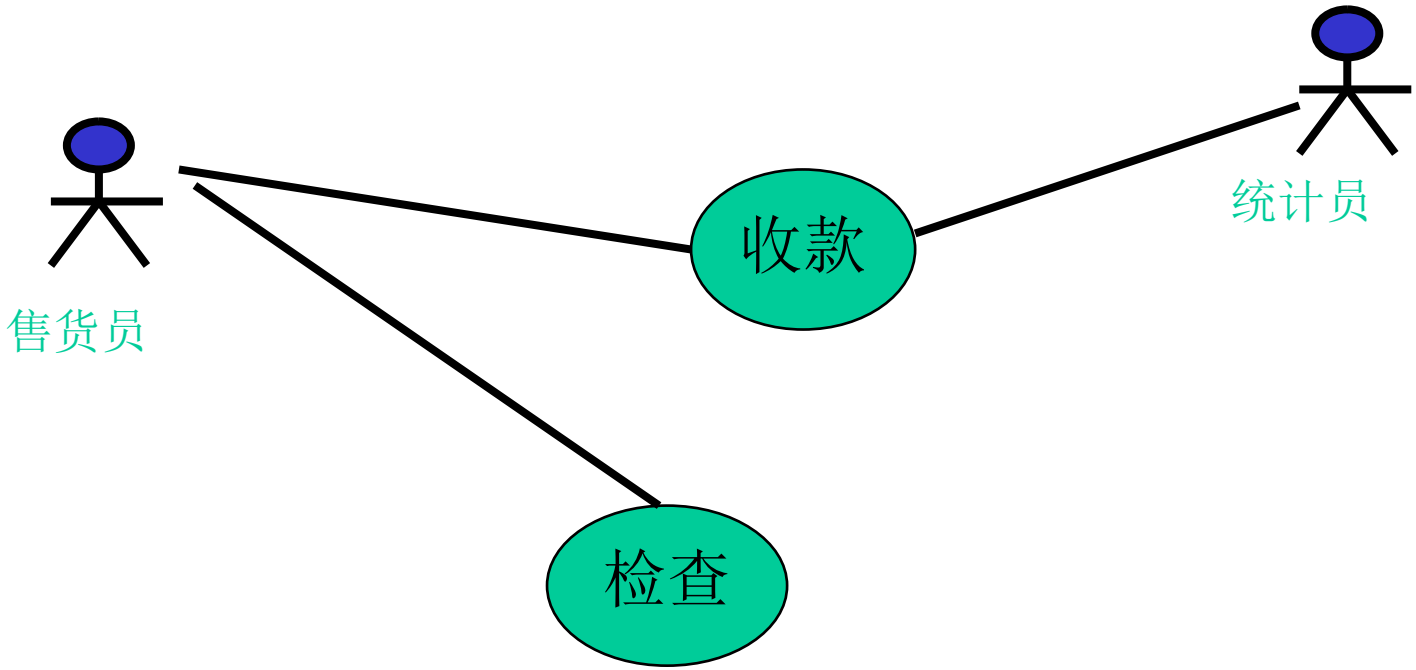
2 用况与参与者之间的关系

用况与参与者间的关联是参与者在用况中的参与（也就是参与者实例与用况实例之间的相互通信）。

若没有进行特殊的说明，任何一方都可发送和接收消息。即交互是双向的，参与者能够产生对系统的请求，或系统要求参与者采取某些动作。

这是参与者和用况之间的唯一关系。

把参与者和用况之间的关联表示成参与者和用况之间的一条实线。



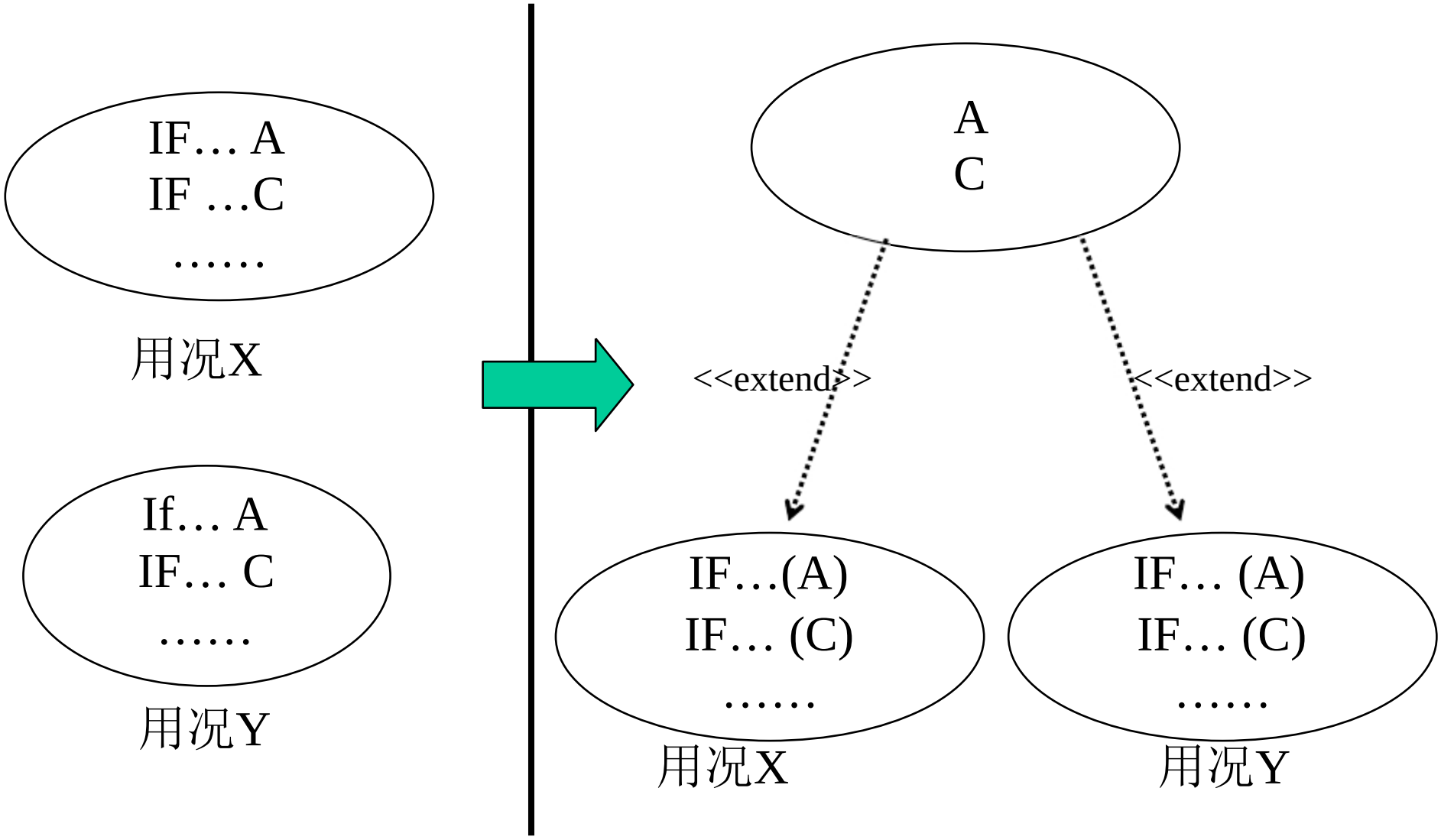
一个用况可能要与系统的一个或几个参与者交互。

3 用况之间的关系

在下述情况下，就需要考虑产生新的用况，并在用况间建立关系：

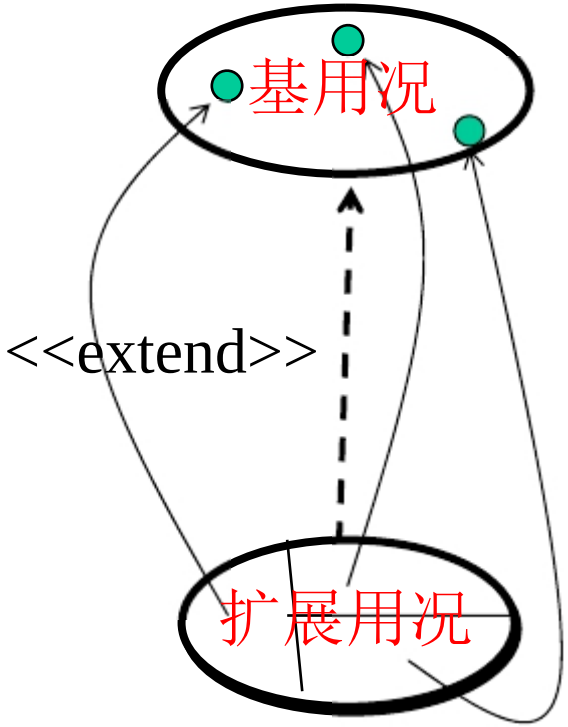
- 在一个用况中存在着几处重复使用的动作序列；
- 在几个用况中存在着重复使用的动作序列；
- 一个用况中的主要动作序列或分支动作序列过于冗长或复杂，而且分离它们有助于管理和理解。

1）扩展


IF… A
IF …C
……
用况X
If… A
IF… C
……
用况Y
A
C
<<extend>>
<<extend>>
IF…(A)
IF… (C)
……
用况X
IF… (A)
IF… (C)
……
用况Y

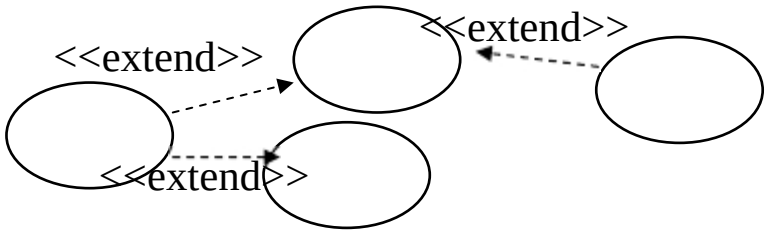
从基用况到扩展用况的扩展关系表明：按基用况中指定的扩展条件，把扩展用况的行为插入到由基用况中的扩展点定义的位置。

通过敞开的虚线箭头表示用况之间的扩展关系，该箭头从提供扩展的用况指向基用况。这个箭头用关键字**<<extend>>**标记。可以在这个关键字附近表示这个关系的条件。

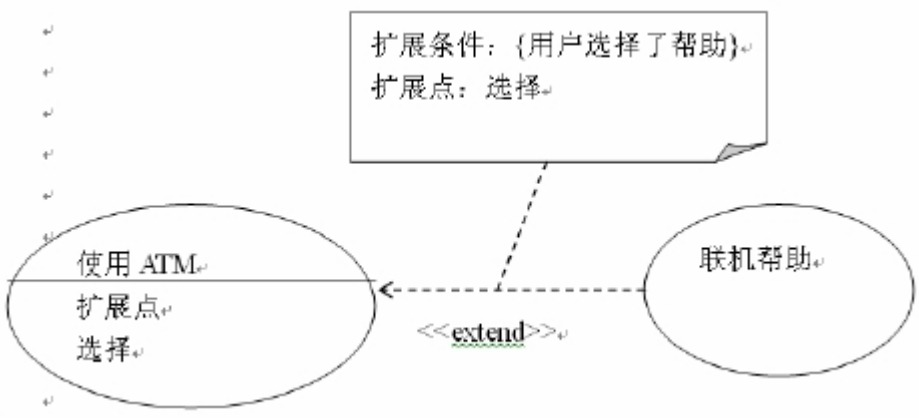


基用况是可单独存在的，但是在一定的条件下，它的行为可以被另一个用况的行为扩展。在扩展用况中可定义一组行为增量，在其中定义的行为离开基用况单独是没意义的。

一个扩展用况可以扩展多个用况，一个用况也可以被多个用况扩展，甚至一个扩展用况自身也可以被其他扩展用况来扩展。



一个扩展点是基用况中的一个位置，在这样的位置上，如果扩展条件为真，就在其中插入扩展用况中描述的全部动作序列或其中的一部分，并予以执行。执行完后，基用况继续执行扩展点下面的行为。如果扩展条件为假，扩展不会发生。



基用况　　扩展用况

可以把扩展点列在用况中的一个题头为“扩展点”的分栏中。以一种适当的方式给出扩展点的位置描述，通常采用普通的文本。

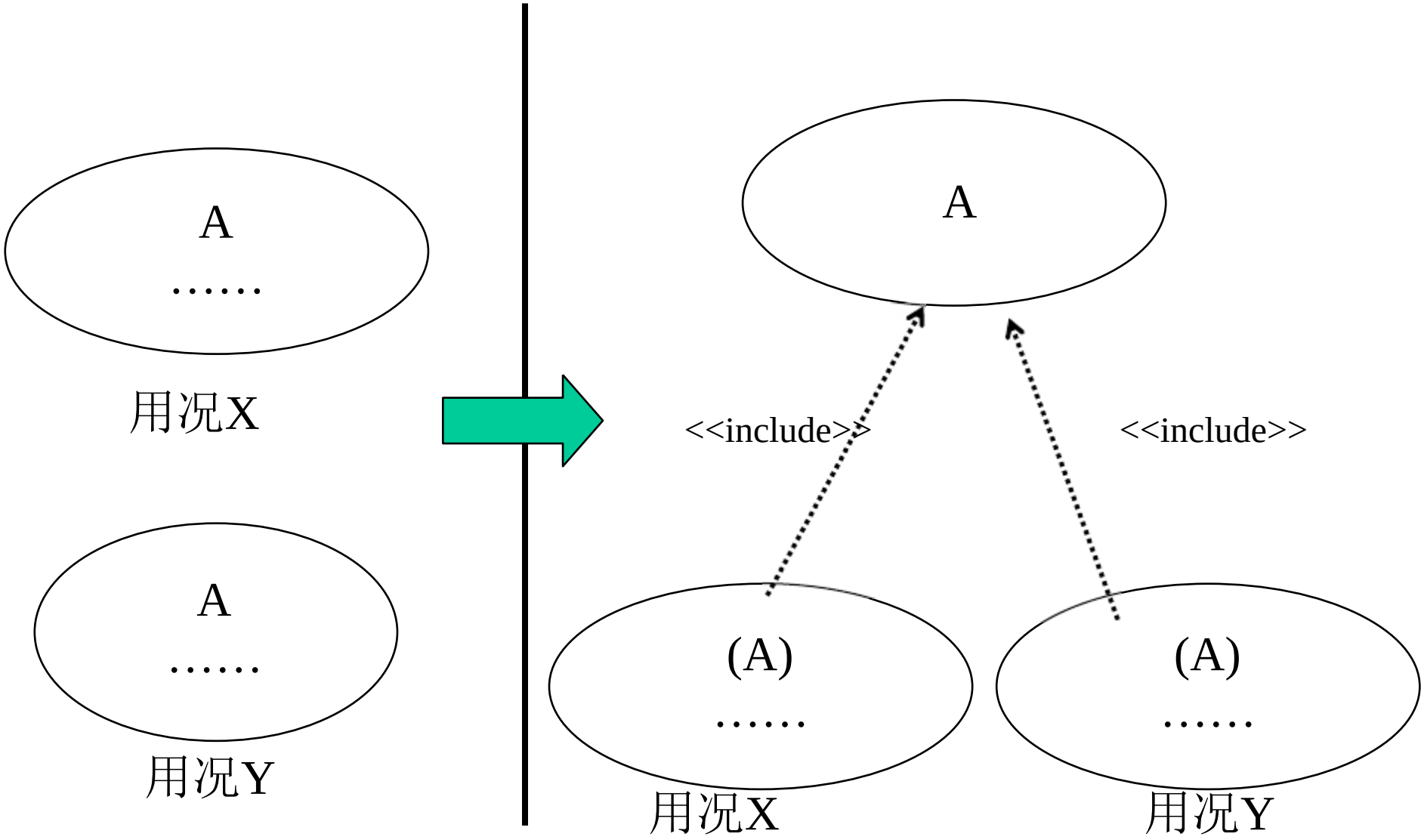
适用之处：

（1）异常情况

（2）正常的变形描述时，而且希望采用更多的控制方式时，采用扩展。即在一个变化点上可附加多个变体的场合下使用。

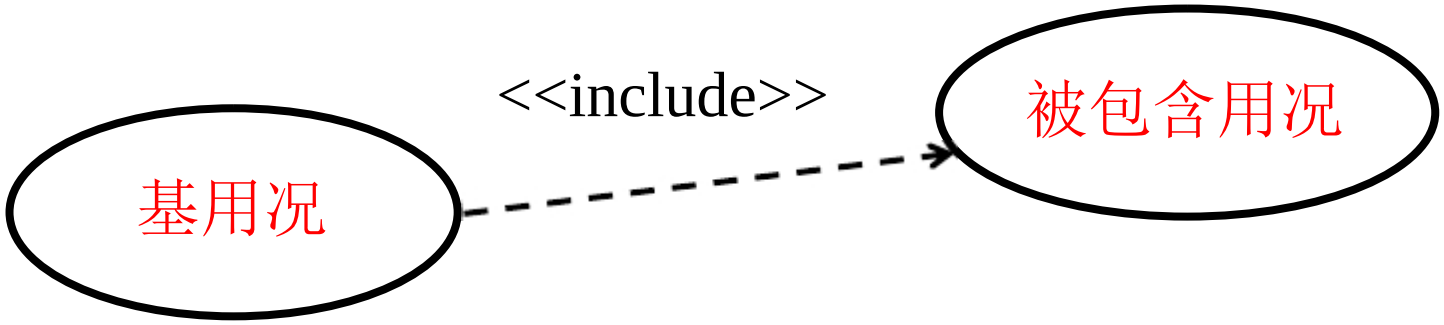
（3）用扩展关系来区分所实现系统的可配置部分。

2）包含



从基用况到被包含用况的包含关系表明：基用况在它内部说明的某一（些）位置上显式地使用供应者用况的行为的结果。

通过一个敞开的虚线箭头表示用况之间的包含关系，该箭头从基用况指向被包含的用况。这个箭头用关键字**<<include>>**标记。

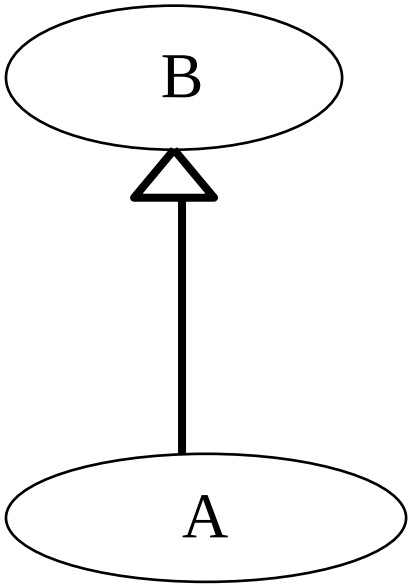


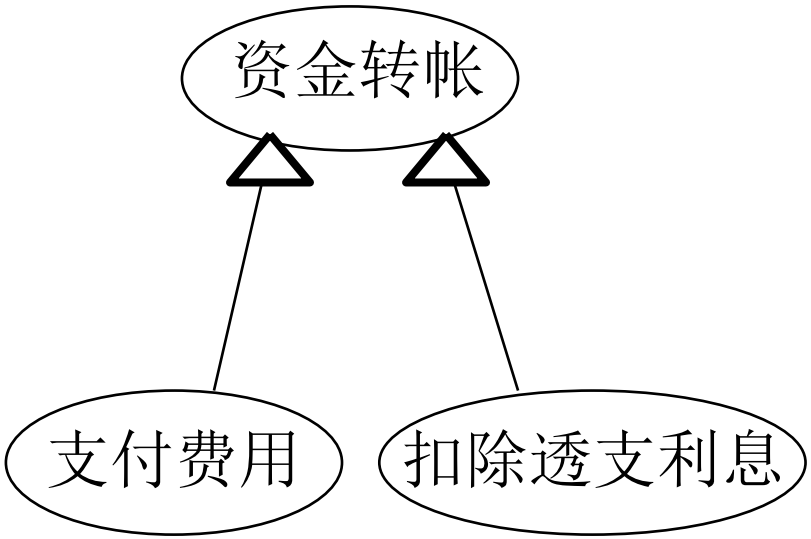
包含关系使得可在一个用况中局部化多个用况中共同的活动序列。这样，可以避免多次描述同一事件流；当这个共同的序列发生变化时，这样就显现出优势，即只需要在一个地方进行改动。

3）泛化

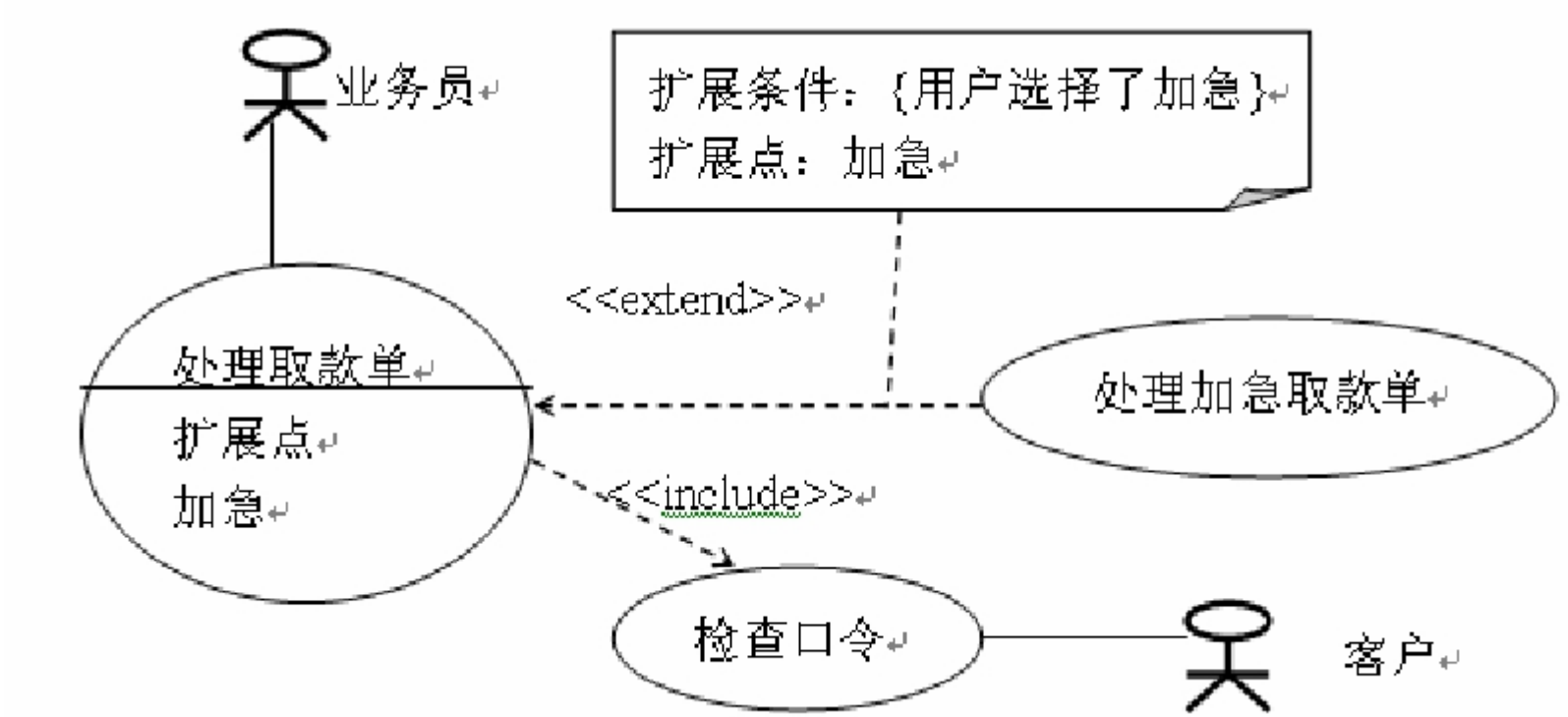
用况之间的泛化关系就像类之间的泛化关系一样，子用况继承父用况的行为和含义。

用一个指向父用况的带有封闭的空心箭头的实线来表示用况之间的泛化关系。





也可以使用包含



被包含的用况可以 单独存在

处理取款单

业务员输入处理取款命令

Include 检查口令

收集客户的取款单上的信息；

加急（扩展点）{如果客户选择了加急} extend 处理加急取款单

……

4 捕获用况

1）利用参与者捕获用况

对所有的参与者（把自己作为参与者），提问下列问题：

- 每个参与者的主要任务是什么

该参与者是否在交互过程中读写系统，它们是怎样使用系统的服务来完成它们的任务以达到目的？

为了达到某种目的，参与者参加什么活动？

4 捕获用况

1）利用参与者捕获用况

例

收款员收款

输入开始本次收款的命令；
作好收款准备，应收款总数置为0，输出提示信息；

for 顾客选购的每种商品 do
  输入商品编号；
  if 此种商品多于一件 then
    输入商品数量
  end if；
  检索商品名称及单价；
  货架商品数减去售出数；
  if 货架商品数低于下限then
    通知供货员请求上货
  end if；
  计算本种商品总价并打印编号、名称、数量、单价、总价；
  总价累加到应收款总数；
end for；

打印应收款总数；
输入顾客交来的款数；
计算应找回的款数，
打印以上两个数目，
应收款数计入帐册。

2）从系统功能角度捕获用况

（1）以穷举的方式检查用户对系统的功能需求是否能在各个用况中体现出来。

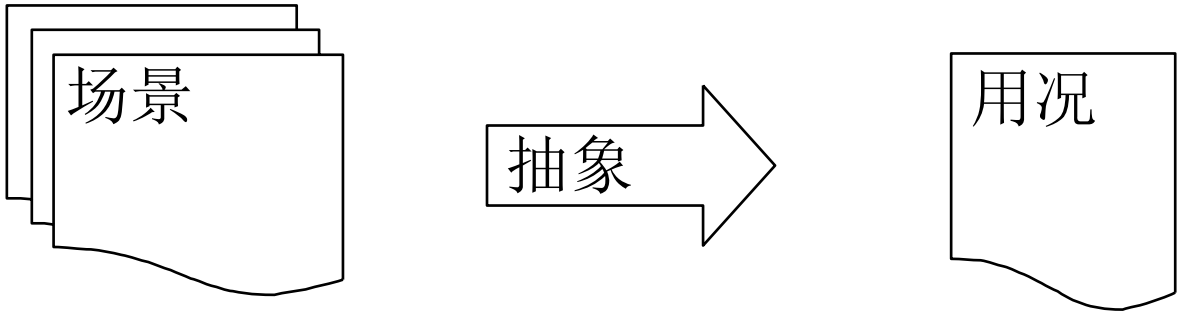
（2）以穷举的方式考虑每一个参与者与系统的交互情况，看看每个参与者要求系统提供什么功能，以及参与者的每一项输入信息将要求系统作出什么反映，进行什么处理。

（3）考虑对例外情况的处理。针对用况描述的基本流，要详尽地考虑各种其他的情况

（4） 一个用况描述一项功能，这项功能不能过大。例如，把一个企业信息管理系统粗略地分为生产管理、供销管理、财务管理和人事管理等几大方面的功能，就显得粒度太大了，应该再进行细化。

（5）一个用况应该是一个完整的任务，通常应该在一个相对短的时间段内完成。如果一个用况的各部分被分配在不同的时间段，尤其被不同的参与者执行，最好还是将各部分作为单独的用况对待。

3）使用场景技术



如果不能顺利地确定一个用况的描述，可“角色扮演”技术。

5．4 用况图

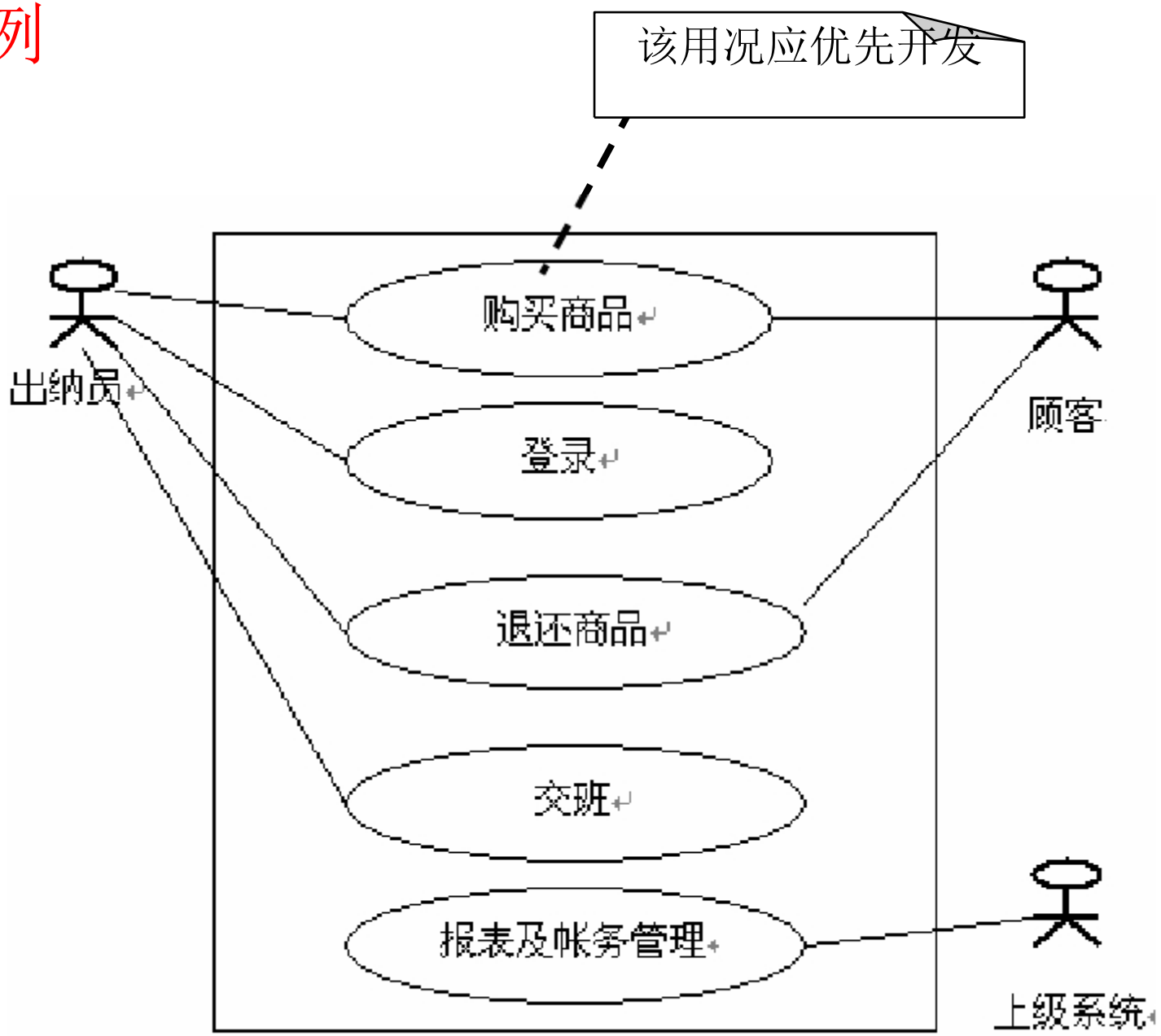
用况图呈现了一些参与者和一些用况，以及它们之间的关系。

在图形上，用况图是一幅由一组参与者、一组用况以及这些元素之间的关系组成的图。这些关系是参与者和用况之间的关联、参与者之间的泛化，以及用况之间的泛化、扩展和包含。

可以选择把一些用况用一个矩形围起来，用来表示系统边界。

用况图可以包含注解和约束。

示例



用况图的益处：

- 为开发者提供一种认识和理解系统的方法
- 为领域专家、最终用户和开发者提供一种相互交流的手段。
- 易于对需求规范化

——用户给出的需求材料常常是不规范或不够准确的。故要较为规范地定义用况，以全面和比较准确地用用况中表达用户的功能需求。

- 有利于进行**OOA**

——有助于发现主动对象

- 对系统测试来说，产生测试用例。
- 有助于设计人机界面
- ……

5.5 审查

参与者

- 确定系统环境中的所有与系统有关的角色，并都归入了相应的参与者。
- 每个参与者都至少和一个用况关联；
- 若一个参与者是另一个参与者的一部分，或扮演了类似的角色，考虑在它们之间使用泛化关系；

用况

- 每个用况都至少和一个参与者相关；
- 若两个用况有相同或相似的序列，可能需要合并它们，或抽取出一个新用况，在它们之间使用包含、扩展或泛化关系。
- 若用况过于复杂，为了易于理解，考虑进行分解；若一个用况中有完全不同的事件流，应该把它分解成不同的用况

例题

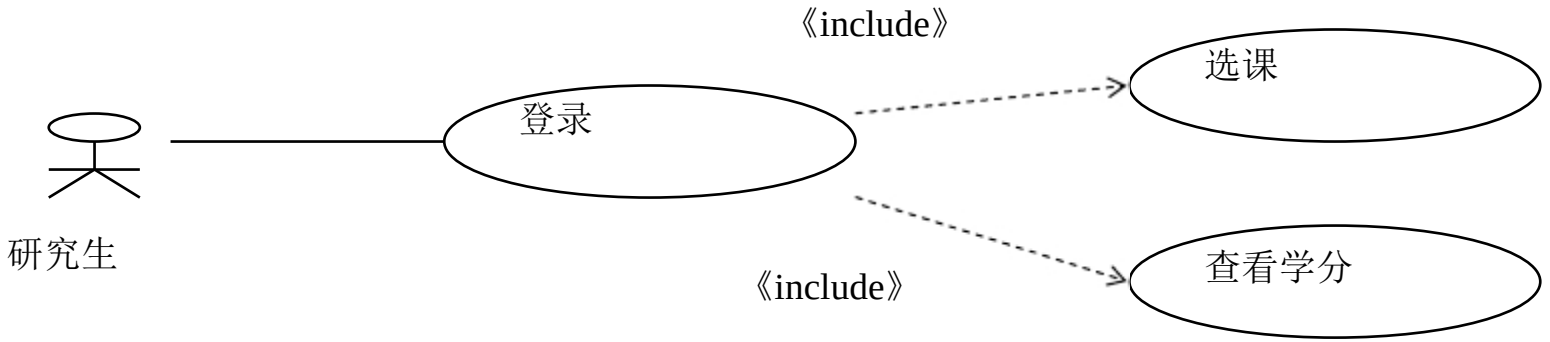
很多软件系统在一开始都需要登录，若用户登录成功，则可进入系统。

如下以一个研究生学籍管理系统为例，描述四种登录方法。

为了简化起见，假设此处仅描述登录、选课和查看学分这3项功能。

方案一：

由于选课和查看学分都需要登录，故专门设立一个“登录”用况。若登录成功，则可以进行选课，也可以进行查看学分。



如下为对用况“登录”的描述：

研究生启动系统；
　　系统提示研究生输入研究生证号和密码；
研究生输入研究生证号和密码；
　　系统进行验证，给出验证信息；
若通过，若该生选择选课
　　系统执行用况“选课”；
若通过，若该生选择查看学分
　　系统执行用况“查看学分”；

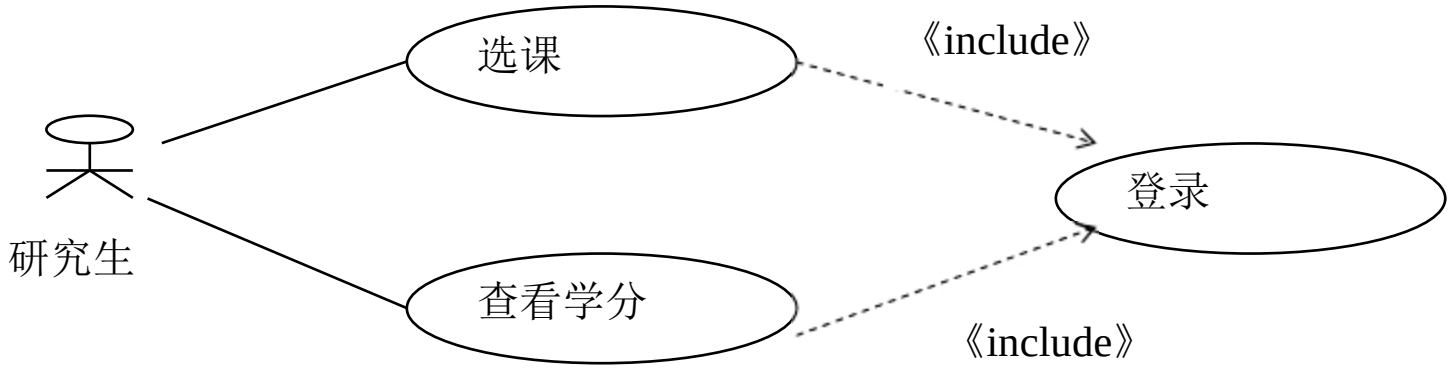
该方法的缺点是，

（1）必须要了解系统的所有其它模块，才能描述清楚“登录”用况。向系统增加新用况时，也要修改登录用况。从维护的角度看，有时会忘记对“登录”用况进行修改。

（2） “登录”用况的功能不单一。

方案二：

让所有的相关用况都包含登录用况。



如下为对用况“选课”的描述：

研究生启动系统，调用用况“登录”
若通过，系统执行用况“选课”的其余部分；

如下为对用况“查看学分”的描述：

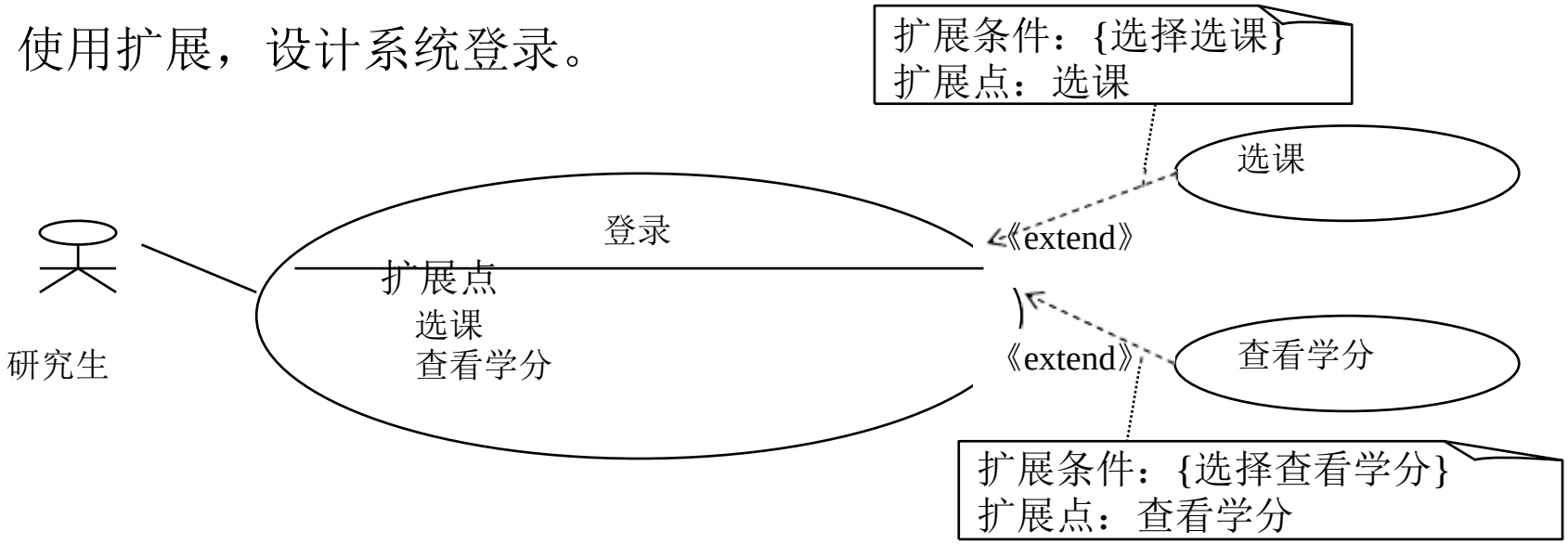
研究生启动系统，调用用况“登录”
若通过，系统执行用况“查看学分”的其余部分；

这个方法中的“登录”用况仅描述有关登录的信息。

其缺点为，对研究生要进行多次验证。——研究生执行系统的每项功能都要先登录。

方案三：

使用扩展，设计系统登录。



如下为对用况“登录”的描述

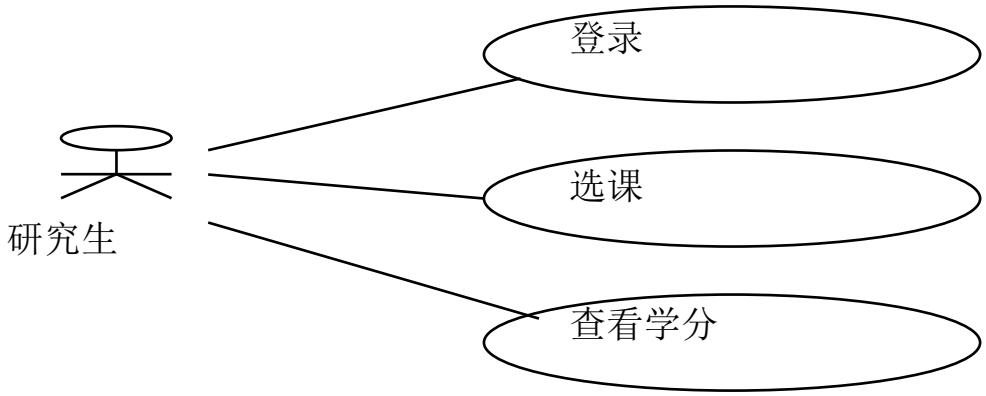
研究生启动系统；
　　系统提示研究生输入研究生证号和密码；
研究生输入研究生证号和密码；
　　系统进行验证，给出验证信息；
若通过，若该生选择选课
　　系统在扩展点”选课”处执行用况“选课”；
若通过，若该生选择查看学分
　　系统在扩展点”查看学分”处执行用况“查看学分”；

该方法与方法一相比，对“登录”用况的描述要清楚一些。在增加新用况时，仅在登录用况中添加扩展点即可。

缺点：“登陆”用况的功能仍然不单一。

方案四：

登录用况完全独立于其它用况



如下为对用况“登录”的描述：

研究生启动系统；
　　系统提示研究生输入研究生证号和密码；
研究生输入研究生证号和密码；
　　系统进行验证，给出验证信息；

如下为对用况“选课”的开始部分的描述

若研究生通过了登录且选择了选课，
　　系统开始执行用况“选课”；
……

使用该方法，必须要在“选课”用况和“查看学分”用况中指定前置条件：只有在登录成功后才能执行自己。

习题

1、用况之间包含关系、扩展关系与泛化关系有相同之处吗？

2、论述用况图在面向对象方法中的地位。

3、通常自动售货机会按用户的要求进行自动售货。供货员会巡查向其内供货，取款员会定时取款。请建立用况图，并描述各个用况。