# 第四章 用例和用例图

## 4.1 用例和用例图的概念

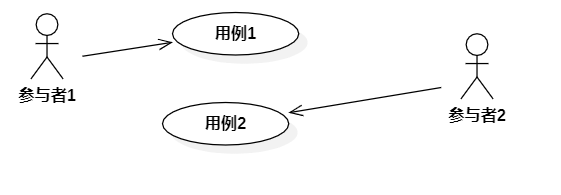
用例模型的基本组成部分有用例、角色（或参与者）和系统。

用例用于描述系统的功能，就是从用户的角度来说，系统具体应包含哪些功能，帮助分析人员理解系统的行为，它是对系统功能的宏观的、整体的描述。

参与者是指那些与系统进行交互的外部实体，通常它是系统的一个用户，但它也可以是其他系统或硬件设备。总之凡是需要与系统进行交互的任何实体都可以称作参与者。

用例图（Use Case Diagram）是显示一组用例、参与者以及它们之间的关系的一种图。

用例图的主要作用有：（1）用来描述将要开发系统的功能需求和系统的使用场景。（2）作为设计和开发过程的基础，促进各阶段开发工作的进展.（3）用于验证与确认系统需求。

用例图的组成元素有：（1）参与者（Actor）：也称为角色，它代表系统的用户。（2）系统边界（System Scope）：它确定系统的范围，（3）用例（Use Case）：它代表用户提供的服务。（4）关联（Association）：它表示参与者和用例间的关系。

### 4.1.1 参与者

参与者（Actor）也可以称为角色，是系统外部的一个人或物，它以某种方式参与了系统的执行过程。参与者不是特制人，是指系统以外的，在使用系统或与系统交互中所扮演的角色。

参与者的作用有：(1)建立系统的外部用户模型。（2）对系统边界之外的对象进行描述。

### 4.1.2 用例

需求获取（Requirement Elicitation）是需求分析阶段的主体部分，其主要的工作就是要建立待开发系统的模型，而用例就是用于建立这种模型的最好方法。

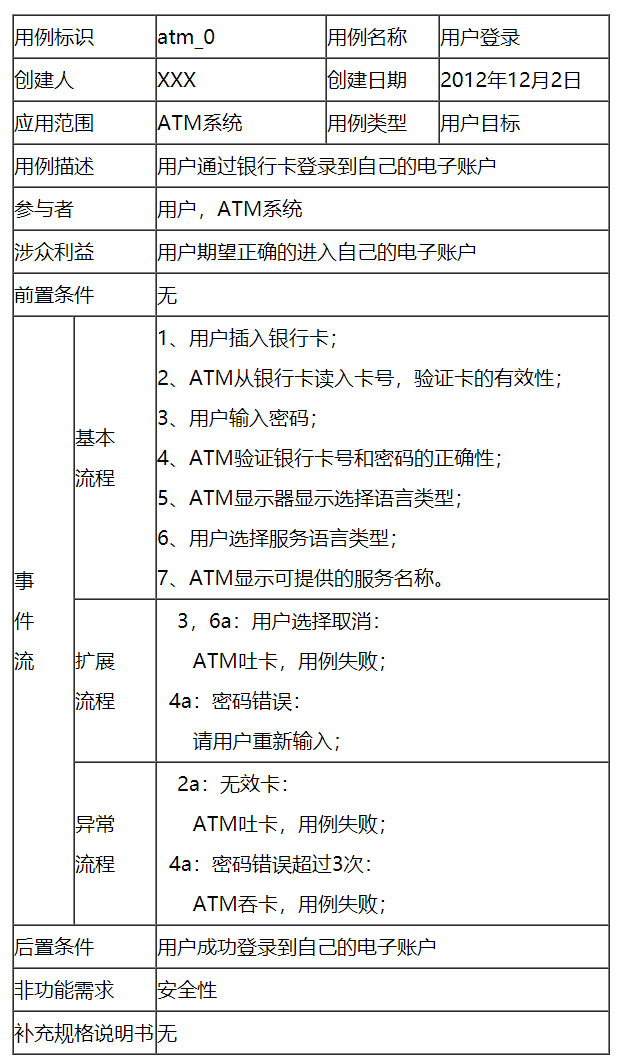
用例是代表系统中的各个项目相关人员之间根据系统的行为所达成的契约。用例描述了在不同条件下，针对某一项目相关人员的请求，系统对其做出的响应。用例指的是对一组动作的描述，系统通过执行这些动作将对用例的参与者产生能够看到的结果，用来描述参与者可以感受到的系统服务或功能。

使用用例进行系统的需求分析的特点：（1）用例是从系统的使用角度描述系统中的信息。即在系统的外部所能看到的系统的功能，而不是考虑系统内部对该功能的具体实现方式。（2）用例描述了用户提出的一些可见需求，对应一个具体的用户目标。使用用例可以促进与用户的沟通，正确地理解需求，同时也可以用来划分系统与外部实体的界限，是面向对象分析与设计的起点，是类、对象、操作的来源。（3）用例通常由某个参与者来执行。（4）用例把执行的结果反馈给参与者。（5）用例在功能上具有完整性，即它从参与者接收输入，产生的结果最终再输出给参与者。  
4.1.3 用例描述

从软件开发的角度，用例就是需求的文字性描述，主要是说明系统如何工作的功能性或行为性需求。用例图只是简单地用图形的方式描述了一下系统。实际上，用例是文本形式，不是图形。用例是作为人与人之间，尤其是没有受过专门培训的人员之间互相交流的一种手段。因此对于每个用例，还需要有详细的说明，编写用例描述让用户对这个系统有一个更加详细的了解。

用例描述一般包括：用例编号、用户概述（说明）、前置（前提）条件、基本事件流、其他事件流、异常事件流、后置（事后）条件。

用户描述示例：



## 4.2 用例之间的可视化表示

### 4.2.1 包含关系

包含关系指的是两个用例之间的关系，其中一个用例称为基本用例（Base Use Case）的行为包含另一个用例，称为包含用例（Inclusion Use Case）的行为。基本用例会用到包含用例，表示基本用例中重用包含用例中的步骤。在UML图中，使用带虚线箭头表示，并在线上标有<<include>>。

在包含关系中，箭头方向是从基本用例到包含用例，即基本用例是依赖于包含用例的。

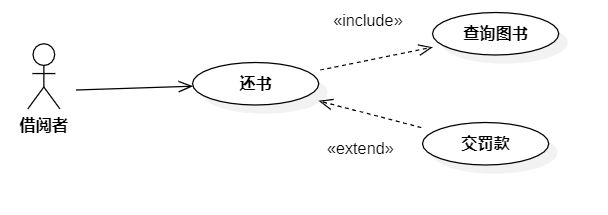


### 4.2.2 扩展关系

扩展（extend）关系是对基本用例的扩展，基本用例是一个完整的用例，即使没有子用例的参与，也可以完成一个完整的功能。在扩展关系中，对于扩展用例（Extension Use Case）有更多的规则限制，即基本用例必须声明若干“扩展点“（Extension Point），而扩展用例只能在这些扩展点上增加新的行为和定义。

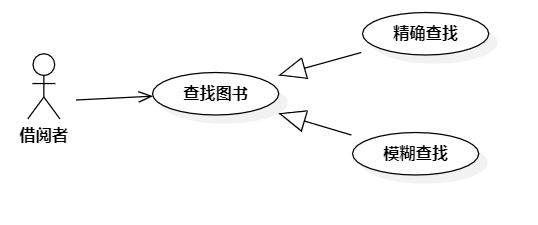
在以下几种情况，可以使用扩展用例：（1）表明用例的某一部分是可选的系统行为。（2）表面只在特定条件（如例外条件）下才执行的分支。（3）表明可能只有一组行为，其中一个或多个可以在基本用例中的扩展点处插入。所插入的行为和插入的顺序取决于在执行基本用例时与主角进行的交互。

在UML图中，使用带虚线箭头表示，并在线上标有<<extend>>。



### 4.2.3 泛化关系

泛化关系指的是一般与特殊的关系。当多个用例共同拥有一个类似的结构和行为的时候，可以将它们的共性抽象为父用例，其他的用例作为泛化关系中的子用例。在泛化关系中，子用例是父用例的一种特殊形式，子用例继承了父用例所有的结构、行为和关系。



### 4.2.4 分组关系

在一些用例图中，用例的数目可能很多，这时就需要把这些用例组织起来。最直接的方法就是把相关的用例放在一个包中组织起来。一组用例可以放在一个文件夹中。

## 4.3 用例图建模技术及应用

创建用例图模型主要包括以下三部分内容：

（1）识别出系统中的角色和用例。

这项任务通常是由系统分析员，通过与用户进行沟通来完成的。通过与用户之间的交流，提出问题，了解他们的业务需求。对于这些业务需求，需要向用户提出一些问题以得到所需要的答案。这些需求和得到的答案将成为创建用例图的基础信息。

（2）区分用例之间的先后次序。

某些用例必须在其他用例之前完成，因为它们之间需要相互依赖。

（3）创建用例图模型结构。

将已确定并细化的角色和用例放入用例图中，此时，再借助包含、扩展和泛化的关系给出用例之间的结构模型。

## 问题：

1. 判断题：参与者是指那些与系统进行交互的外部实体，通常它是系统的一个用户，但不能是其他系统或硬件设备。

答案：参与者也可以是其他系统或硬件设备。总之凡是需要与系统进行交互的任何实体都可以称作参与者。

2.简述题：包含关系和扩展关系的区别。

答案：扩展是从基本用例的基础上扩展出新的功能（子用例），子用例不影响基本用例，基本用例本身是完整的，没有子用例的参与也可以完成自己的功能，而包含关系是提取出来的用例是基本用例的一部分基本用例和子用例必须一起使用才完整。二者的关键在于离开子用例，基本用例是否可以完成一个完整的功能。