

# Object oriented Analysis &Design

## 面向对象分析与设计



### 交互图

Presented by: Xiaohong Chen  
[xhchen@sei.ecnu.edu.cn](mailto:xhchen@sei.ecnu.edu.cn)

2020/10/20



# Interaction Diagram 交互图 (掌握内容)

- 交互 interaction
  - 对象、角色(role)、参与者 actor
  - 消息 message
- 交互图
  - 顺序图 Sequence diagram
  - 协作图 (合作图) Communication diagram
- 交互图到类图的转换

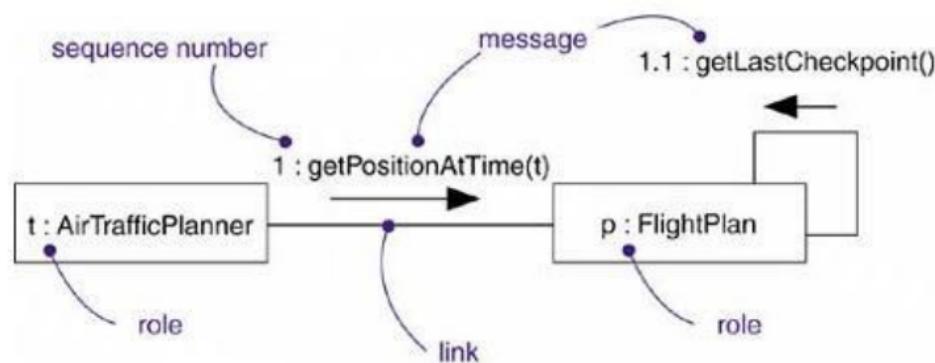
# 1 引言

- 在任何有意义的系统中，对象都不是孤立存在的，它们之间通过传递消息进行交互。
- 使用**交互**建模软件系统中对象之间的消息的传递，用以描述对象之间的交互行为。
- **交互**是为达某一目的而在一组对象之间进行消息交换的行为。
- 交互可以对软件系统为实现某一任务而必须实施的动态行为进行建模。
- 交互所包含的UML建模元素包括
  - 对象或角色
  - 角色、参与者
  - 消息
- 在UML中，使用**交互图**建模对象之间的交互。

## 2. 交互的基本概念-- 消息

- 对象间的互相合作与交流表现为一个对象以某种方式启动另一个对象的活动，通过发送消息实现对象相互之间的交互。

Figure 16-1. Messages, Links, and Sequencing



## 2.1 交互的基本概念-- 消息

- 消息所能采取的形式：
  - 调用 (call) : 启动某个对象的操作
    - 操作是对象的类所能提供的服务的实现
    - 对象也可以给自己发送消息
  - 返回(return): 操作向调用者返回一个值
  - 发送(send): 向一个对象发送一个信号。
    - 同步消息
    - 异步消息
  - 创建(create): 此消息的发送导致目标对象被创建。
  - 销毁(destroy): 此消息的发送导致目标对象被销毁
    -

## 2.1 交互的基本概念-- 消息

### 消息的表示

在UML里，消息用箭头表示，从发送消息的对象指向接收消息的对象

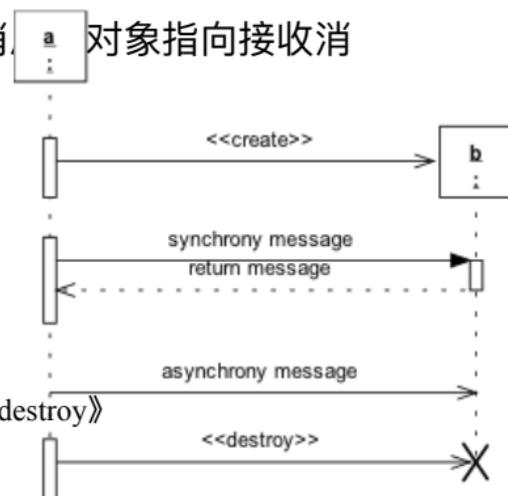
同步消息：实心箭头

异步消息：枝状箭头

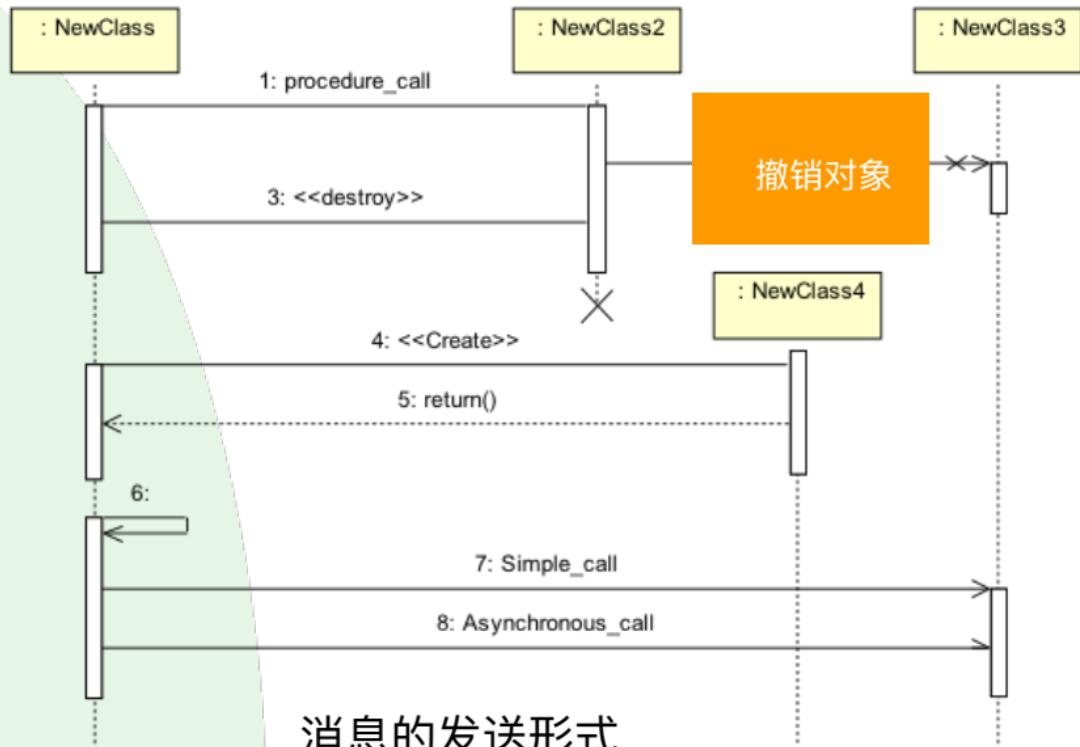
在消息的各种形式中  
创建和销毁消息

用消息的构造型来表示《create》, 《destroy》

返回消息，用带虚线的箭头表示



## 2.1 交互的基本概念-- 消息



## 2.1 交互的基本概念-- 消息

### 消息的表示

消息可以有名字

它列在消息的箭头的  
直线上

如果对象的实现类已  
经确定，则此名字可以  
标记为实现类的某一操  
作的定义

例如，C/C++语言里的  
函数定义等

消息的发送是有顺序的  
此顺序由它在顺序图  
垂直方向上的位置决定  
垂直方向靠近顺序图  
的顶端的消息先执行  
靠近顺序图底部的消  
息后执行

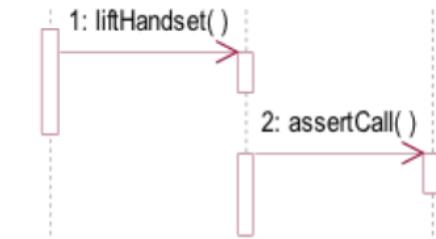
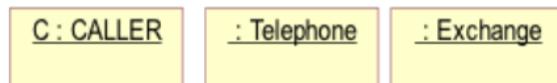


图 消息、消息名和消息顺序号 (单调顺序号)

## 2.1 | 交互的基本概念-- 消息

### 消息的顺序号

此顺序号可前缀于消息的名字前面  
它们之间用冒号分隔

顺序号分为两种：

**单调顺序号**(flat sequence)

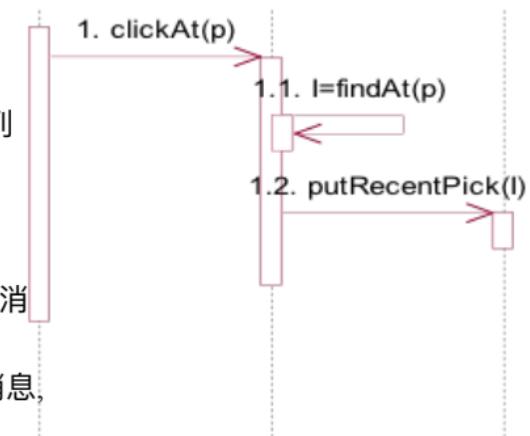
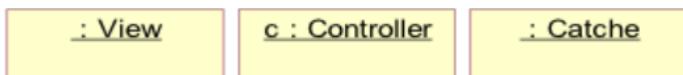
单调顺序号严格按照消息的发送顺序排列  
如:1,2,3,...,等等

**过程顺序号**(procedural sequence)

过程顺序号是嵌入式的

当一个消息启动了另一个消息顺序时, 此消息顺序内的各消息就可以重新开始编号。

如：消息1发送后, 启动了其后的一系列消息,  
则这些消息就可以编号为1.1, 1.2, 1.3, ...



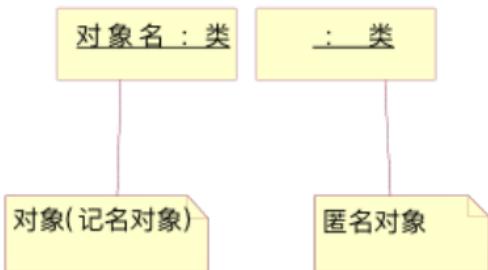


## 2.2 交互的基本概念—语境

- 交互通常发生在一定的语境、场景中：
- 例如：C/S系统中，Client对象和Server对象之间有交互
- 在操作的实现中可以发现对象之间的交互
- 操作的参数、局部/全局变量
- 在构件、节点或用例的表示中发现交互

## 2.3 交互的基本概念– 对象和角色

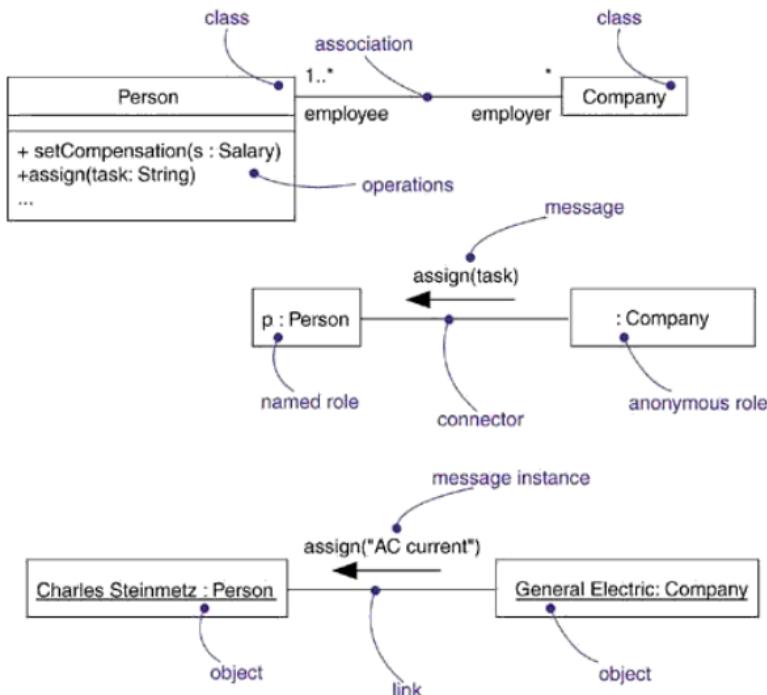
- 参与交互的**对象**既可以是具体的事物，又可以是角色
  - 学生张力、学生王海（具体的事物）
  - 学生甲、路人乙（角色）
- 影视剧中的角色：
  - 肯定需要一个人来演
  - 但不一定是某个特定的演员来演



## 2.4 | 交互的基本概念 - 链

- 链是关联的实例(a link is an instance of an association )

Figure 16-2. Associations, Links, and Connectors





### 3. 交互图

- 在UML中使用交互图建模系统中的交互行为：
  - 交互图描述了一系列的对象之间传递的消息
- 交互图分为两种：
  - 顺序图 Sequence diagram
  - 协作图 collaboration diagram
- 它们在语义上是等价的
  - 顺序图：强调消息的时间顺序
  - 协作图：强调接收和发送消息的对象的结构组织



## 3.1 顺序图

### 顺序图的定义：

顺序图是交互图的一种，它强调的是消息发送在时间上的先后顺序。

### 顺序图的构成

参加交互的各角色（对象、参与者）在顺序图的顶端沿水平方向排列

#### 角色之间传递的消息

用箭头表示，水平放置，按照消息发送的先后顺序沿垂直方向排列，很直观。

每个对象的底部中心都有一个垂直虚线，这条虚线被称为**对象生命线**（object lifeline）

对象生命线代表一个对象在某个时间段内的存在

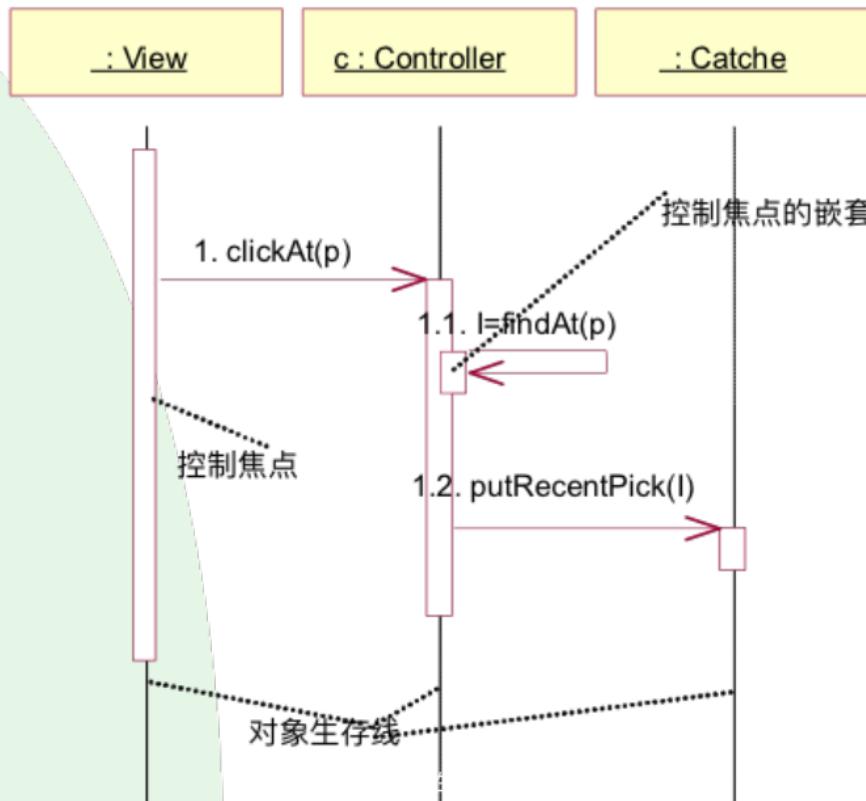
如果在顺序图上某一对象收到了创建消息或销毁消息，则此对象的生存期始于它收到创建消息的时刻，终止于收到销毁消息的时刻。

## 3.1 顺序图

- ### 控制焦点

- 在UML里，由消息引发的动作的执行过程被描述为控制焦点
- 定义：
  - 控制焦点代表一个对象直接地或通过一个子过程间接地执行一个动作的那段时间。
- 图形化表示：它由位于对象生命线上的一个窄长方形表示
  - 控制焦点长方形的顶端代表动作的开始时刻
  - 底端代表动作的结束时刻
  - 控制焦点可以理解为是C语言中一对花括弧（“{}”）内的内容

## 3.1 | 顺序图





## 3.1 顺序图

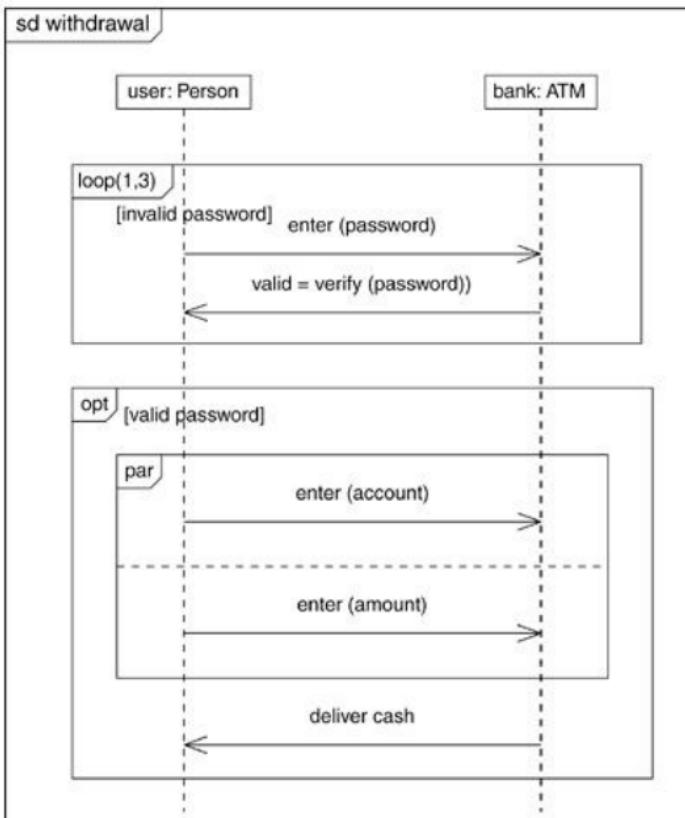
### • 动作(Action)

- 当一个对象收到了消息后，此对象把消息当做执行某种动作的命令。
- 在UML里，一个动作(Action)被定义为
- 动作是执行一系列可执行的原子计算 (atomic computation)，并可导致系统状态的变化或返回某个值。
  - 典型的例子：执行某个函数
  - 在动作的执行过程中，会导致其它一系列的消息的顺序发送
    - 典型的例子是：函数调用其它函数

# 3.1 | 顺序图中的结构化控制

Figure 19-3. Structured Control Operators

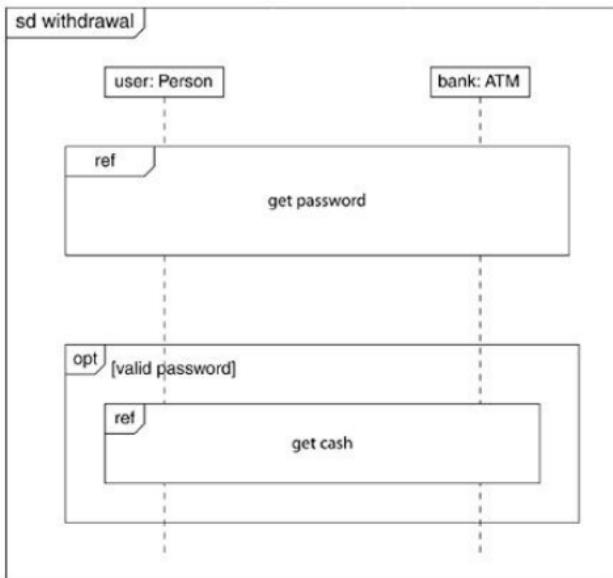
- 顺序图中的结构化控制
  - 可选执行 (标签: opt)
  - 条件执行 (标签: alt)
  - 并行执行 (标签: par)
  - 循环 (迭代) 执行 (标签: loop)



# 3.1 顺序图

Ref (reference):  
表示引用其他交互

Figure 19-4. Nested activity diagram

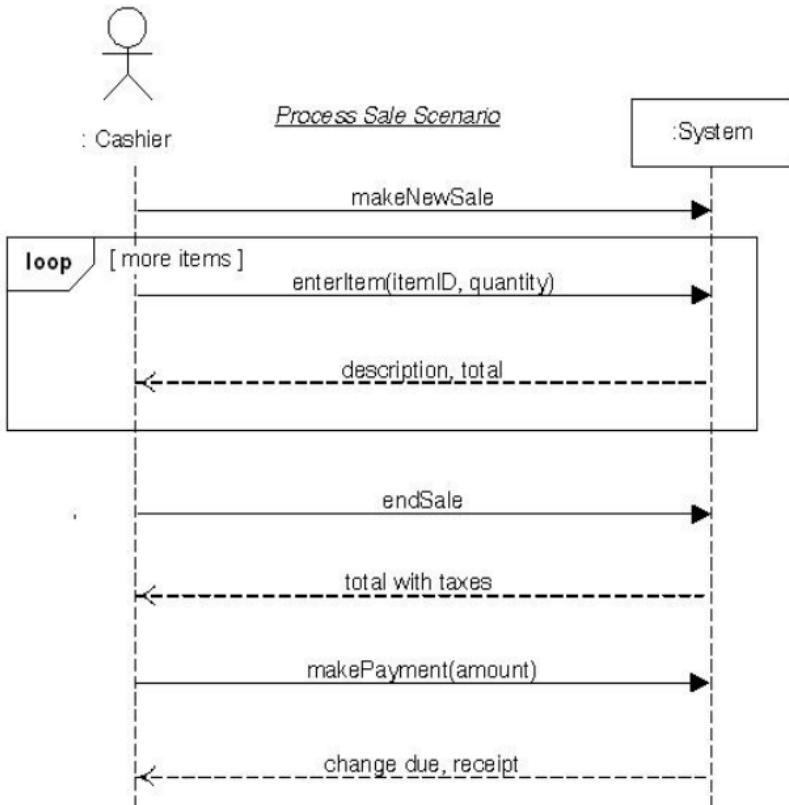


# SD 案例（POS系统）用例描述和用例图

## 用例：处理销售

- 1、顾客携带所购商品来到POS机付款处进行购买交易
- 2、收银员开始一次新的销售交易
- 3、收银员输入商品ID
- 4、系统逐条记录出售的商品条目，并显示该商品的描述、价格和累计额。价格通过一组价格规则来计算
- 5、收银员重复步骤3~4，直到结束
- 6、系统显示总额
- 7、收银员告知顾客总额并提请付款。
- 8、顾客支付，系统处理支付。

° ° °  
 (此处仅有主事件流)



# 练习1：大象放到冰箱里的过程



图片来源：视觉中国 www.vcg.com



作者 www.qingting.com



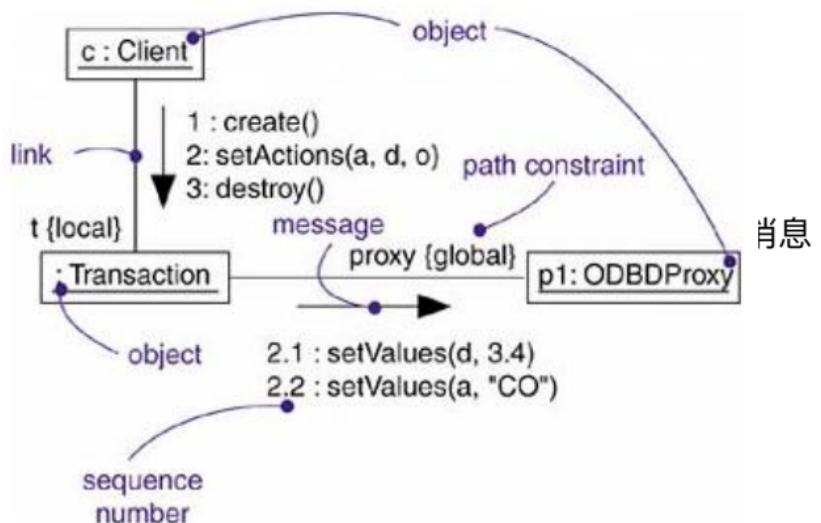
## 小结：顺序图的作用

- 1. 建模系统中对象之间的交互行为
- 2. 根据顺序图上建模的交互动作，定义类图中具体某个类的操作
- 3. 描述USE CASE的事件流
- . . .

## 3.2 协作图

- 协作图是交互图的另一种表现形式。
- 它在语义上和顺序图是等价的

Figure 19-5. Communication Diagram



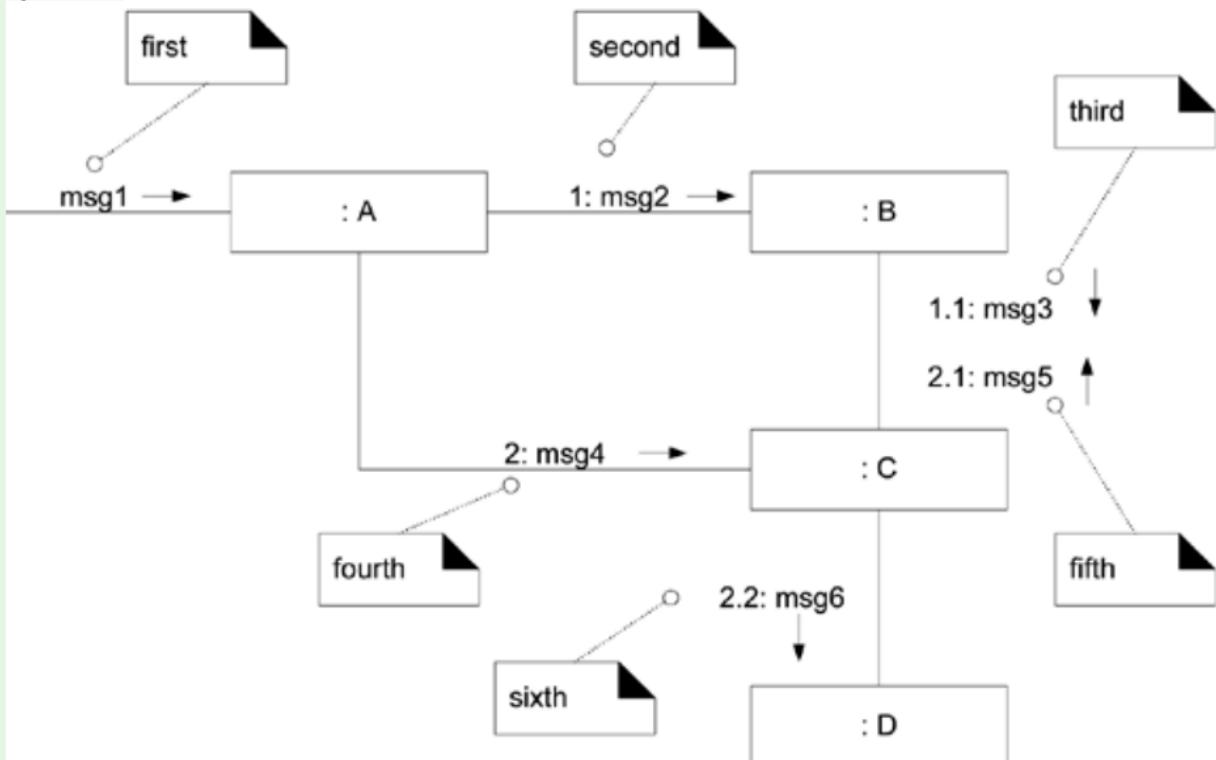


## 3.2 协作图

### • 链接的表示

- 在协作图上，链接用对象之间相连的直线来表示
- 链接可以有名字，它标在表示链接的直线上
- 如果有消息借助此链接关系传递，则把消息的图符沿直线方向绘制，消息的箭头指向接受消息的对象
- 由于仅从图符的绘制上无法在协作图上读出消息发送的顺序
  - 所以，通常在消息上保留对应的顺序图的消息顺序号

## 3.2 协作图 (复杂一点)





### 3.3 顺序图和协作图的区别

- 顺序图和协作图都来自UML的元模型中相同的信息，所以二者在语义上是等价的，可以相互转化。
- 协作图显示对象之间是如何被连接的
  - 但顺序图没有
- 顺序图显示消息的返回、有对象生命线、控制焦点，而协作图没有
- 协作图的特征：
  - 有路径
  - 在消息前加前缀，表示消息的时间顺序

## 4. 建模指南

- 交互图是连接系统边界和系统内部的重要桥梁
- 当一个待建造的系统的需求分析由用例图描述清楚之后
- 使用交互图描述系统的动态行为以及为实现此动态行为系统应具备的合理的结构
- **绘制顺序图时，需要考虑的因素有：**
  - 软件系统的边界
  - 系统参与者和系统的交互
  - 系统为实现这交互内部应设置的对象及其职责



## 4. 建模指南

- 在设置对象时
  - 应考虑软件结构的合理性、软件部件的可重用性、可维护性、可移植性。
  - 在顺序图上，用对象之间的消息定义各对象之间为实现系统的功能而进行的交互。
- 在描述消息顺序时
  - 使用控制焦点来突出为实现特定的动作所需的消息子顺序
  - 动作的嵌套通过控制焦点的嵌套来描述
  - 嵌套的消息顺序，使用过程顺序号来标识
  - 可将顺序图转换为协作图
  - 以进一步考察软件的组织结构，为下一步设计类图作好准备。

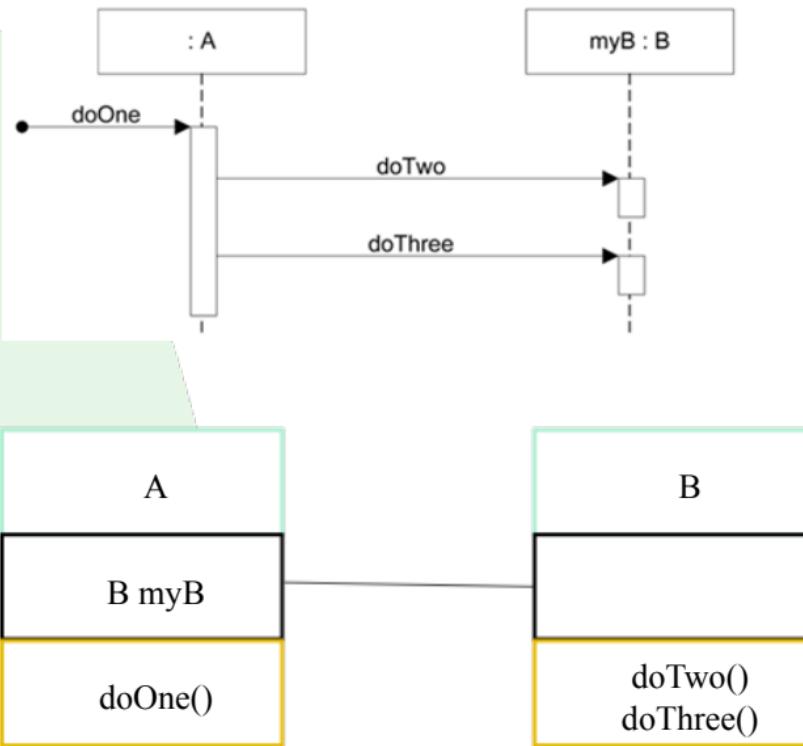
## 4. 建模指南

- 绘制交互图时，应注意图的组织
  - 对于复杂的问题，用多张交互图分别描述
  - 例如，一个用例的多个场景分别表示的多个事件流程，可以用不同的交互图描述
  - 要充分利用UML的模型包的机制，标注的机制使问题的描述有合理明晰的结构。
- 绘制顺序图时
  - 要突出问题的重点，省略对描述问题无关紧要的细节问题
  - 应有节制地在顺序图上描述复杂的分支循环结构
  - 无关紧要的分支循环可留到程序设计时解决
  - 重要而复杂的分支循环，可用活动图来描述

## 4. 建模指南

- 分析和考察类的职责的平衡分布，可以在交互图上进行。
- 软件系统只有和外部直接交互，才能够为其用户提供使用价值
- 因此在设计和建造一个软件系统时，需要对软件系统之外的非软件事物
  - 如用户、外部设备等，进行分析和建模
  - 例如：建造一个信息管理系统
    - 需要分析用户群的结构、分工、职责和业务流程
- 类和对象也是描述这类非软件事物的手段
  - 可以用特定的类来代表一类特定的非软件事物
  - 用交互图描述它们的业务流程
  - 用属性和操作定义它们的职责
  - 可以用UML的变体机制将非软件事物与软件事物相区分

## 5. SD到类图的映射



# 练习：现实中的掷色子游戏的过程

- 一个人拿着两个色子，掷下，如果总和为7，则人赢了，否则输了。