UML2.0 第6章

## 顺序图

**1.定义与作用**

顺序图是强调消息时间顺序的交互图，它描述了对象之间传送消息的时间顺序，用于表示用例中的行为顺序。顺序图显示不同业务对象如何交互，对于交流当前业务如何进行很有用。因此，一个用于描述业务的顺序图能被当作一个需求文件使用，为实现一个未来系统传递需求。

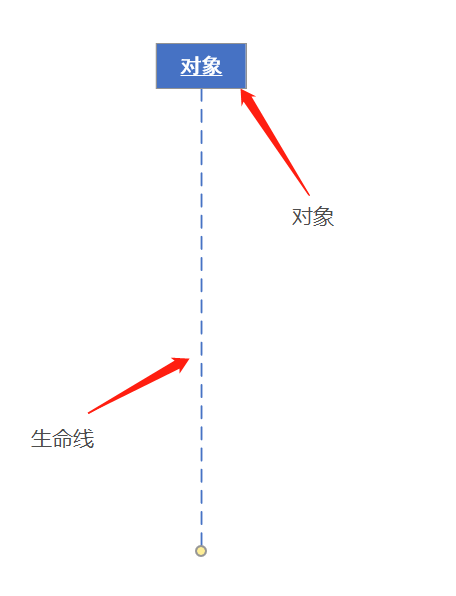
顺序图一般用于确认和丰富一个使用情境的逻辑，主要用途之一就是把用例图、类图表达的需求，转化为进一步、更加正是层次的精确表达。相对于用例图、类图的静态表示，顺序图描述的是一种动态关系

**2.基本元素**

顺序图将交互关系表现为一个二维图，纵向是时间轴，时间沿竖线向下延伸。横向轴代表了在协作中各独立对象的类元角色，类元角色的活动用生命线表示。

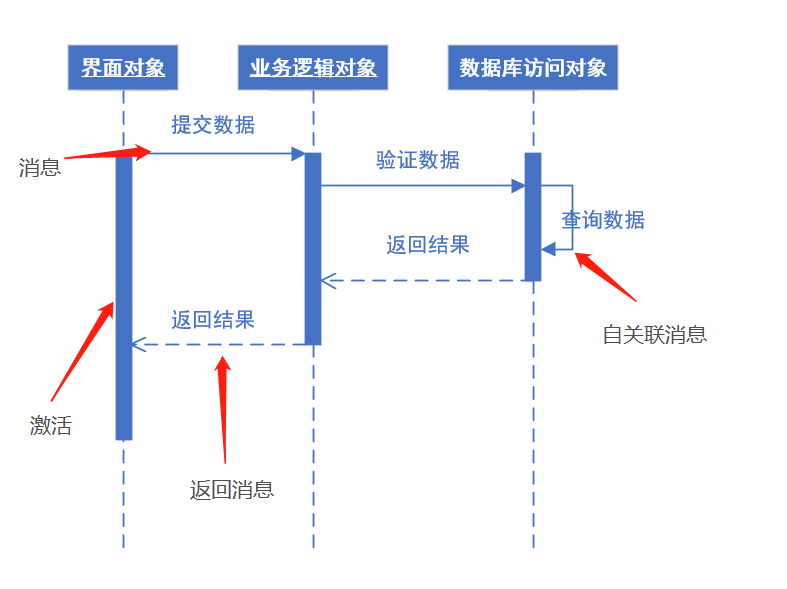
(1).生命线用一条纵向虚线表示。对象在生命线上包含两种状态：激活状态和休眠状态

(2).在UML中，对象表示为一个矩形，其中对象名称标有下划线。

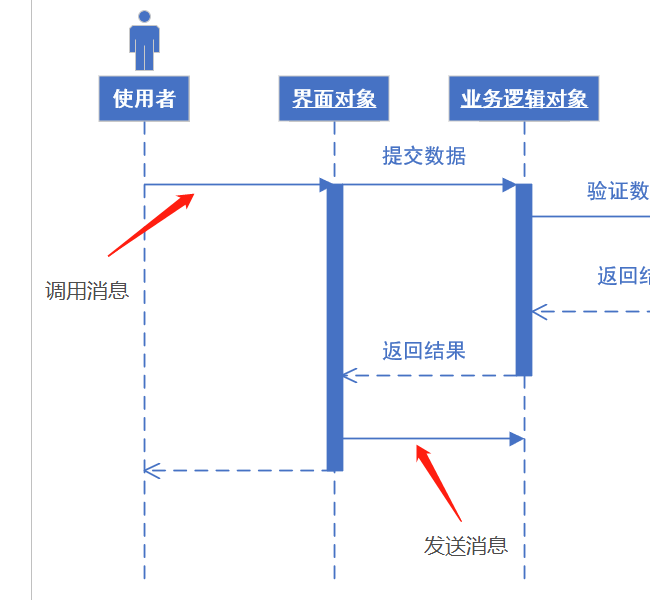


(3)激活是过程的执行，包括等待过程执行的时间。在顺序图中激活部分替换生命线，使用长条的矩形表示。

(4)消息是对象之间的通信，是两个对象之间的单路通信，是从发送者到接收者（两者可以是同一个对象）之间的控制信息流。消息在顺序图中由有标记的箭头表示，箭头从一个对象的生命线指向另一个对象的生命线，消息按时间顺序在图中从上到下排列。其中实现表示发送出去的消息，虚线表示接受到的消息。消息可以激发某个操作、唤起某个信号或导致目标对象的创建或撤销，一个对象到另一个对象的消息用跨越对象生命线的消息线表示

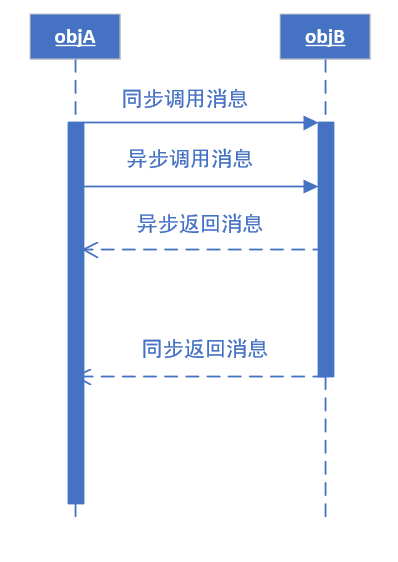


(5)在顺序图中，对象安排在X轴。启动交互的对象放在最左边，随后放入消息的对象放在启动交互对象的右边。交互中对象发送和接收的消息沿着Y轴以时间增加的次序放置。在顺序图中，有的消息对应于激活，表示它将会激活一个对象，这种消息称为调用消息(Call Message)；如果消息没有对应激活框，表示它不是一个调用消息，不会引发其他对象的活动，这种消息称为发送消息(Send Message)。

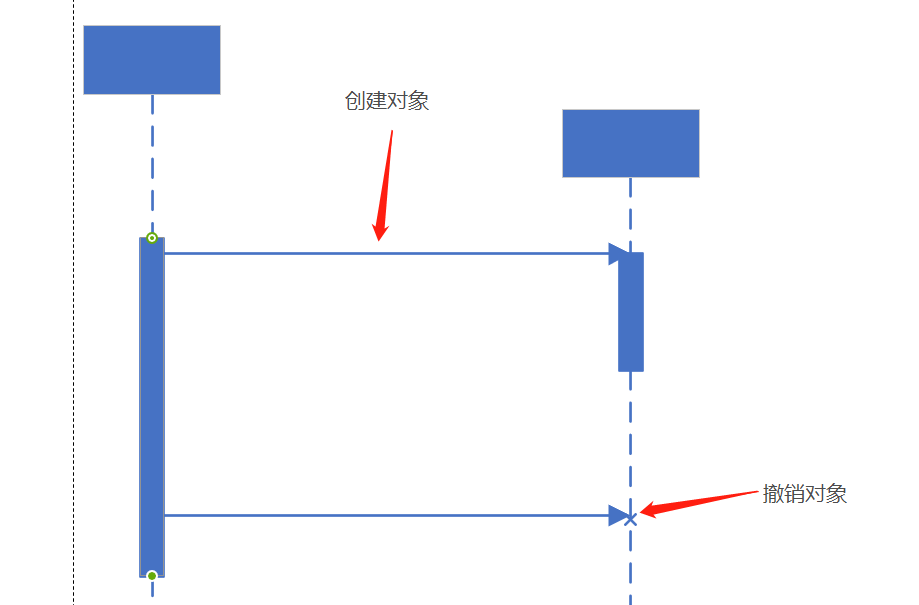


(6) 同步消息：发送消息的对象要等到接收消息的对象执行完所有操作后，发送消息的对象才能继续执行自己的操作。

异步消息：发送消息的对象发送消息后，不用等待接收对象是否执行，继续执行自己的操作。



(7)创建对象：一个对象可以通过发送消息来创建另一个对象；撤销对象：当一个对象被删除，通过“x”来表示



**3.约束**

(1)在UML1.X中，会在消息名前加入约束条件，并放于“[]”中。约束条件用于描述代码中if语句结构。使用“\*[]”描述循环

而在UML2.0中，这种约束被称为组合片段或约束片段

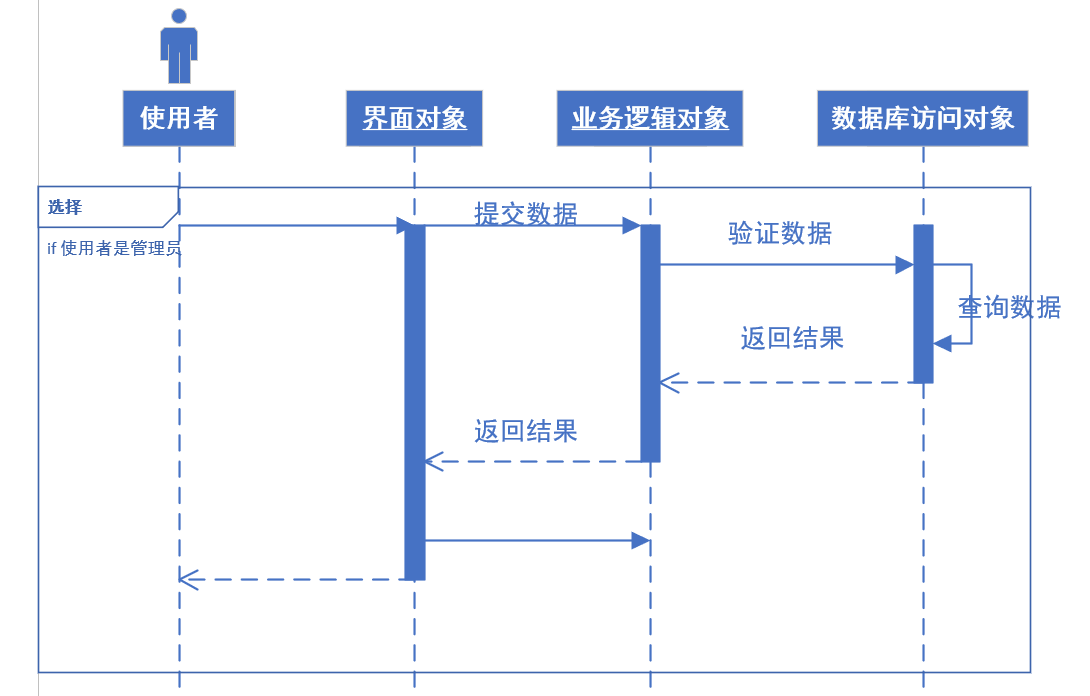
(2)约束片段

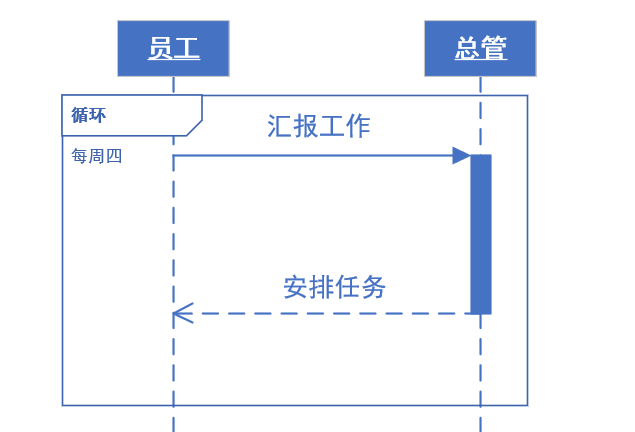
一个复杂的顺序图可以划分为几个小块，每一个小块称为一个交互片段。每个交互片段由一个大方框包围，其名称显示在方框左上角的间隔区内，表示该顺序图的信息。常用操作符如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 操作符 | 缩写 | 名称 | 操作域 | 说明 |
| Alternatives | alt | 抉择 | 多个 | 包含一个片段列表，这些片段包含备选消息序列。在任何场合下只发生一个序列。可以在每个片段中设置一个临界来指示该片段可以运行的条件。else的临界指示其他任何临界都不为True时应运行的片段。如果所有临界都为False并且没有else，则不执行任何片段。 |
| Option | opt | 选项 | 1个 | 包含一个可能发生或可能不发生的序列，可以在临界中指定序列发生的条件。 |
| Break | break | 中断 | 1个 | 如何执行此片段，则放弃序列的其他部分。可以使用临界来指定发生中断的条件 |
| Parallel | par | 并行 | 多个 | 多个操作域的行为并行，操作域以任意顺序交替执行 |
| Week Sequencing | seq | 弱顺序 | 多个 | 有限制的并行。同一条生命线的不同操作域按顺序执行，不同生命线的操作域以任意顺序交替执行 |
| Strict Sequencing | strict | 强顺序 | 多个 | 有两个或更多操作数片段。这些片段必须按给定顺序发生。 |
| Negative | neg | 否定 | 1个 | 不可能发生的消息系列，无操作 |
| Critical Region | critical | 关键 | 多个 | 临界区，用在Par或Seq片段中。指示此片段中的消息不得与其他消息交错。 |
| Ignore | ignore | 忽略 | 多个 | 消息可以在任何地方出现，但会被忽略，往往与其他片段组合在一起 |
| Consider | consider | 考虑 | 多个 | 域ignore相反，不可忽略的消息，往往和其他片段组合使用 |
| Assertion | assertion | 断言 | 多个 | 断言，说明有效的序列。操作数片段指定唯一有效的序列。通常用在Consider或Ignore片段中。 |
| Reference | ref | 引用 | 1个 | 引用组合片段 |
| Loop | loop | 循环 | 1个 | 片段重复一定次数。可以在临界中指示片段重复的条件。Loop组合片段具有"Min"和“Max”属性，它们指示片段可以重复的最小和最大次数。默认值是无限制。 |

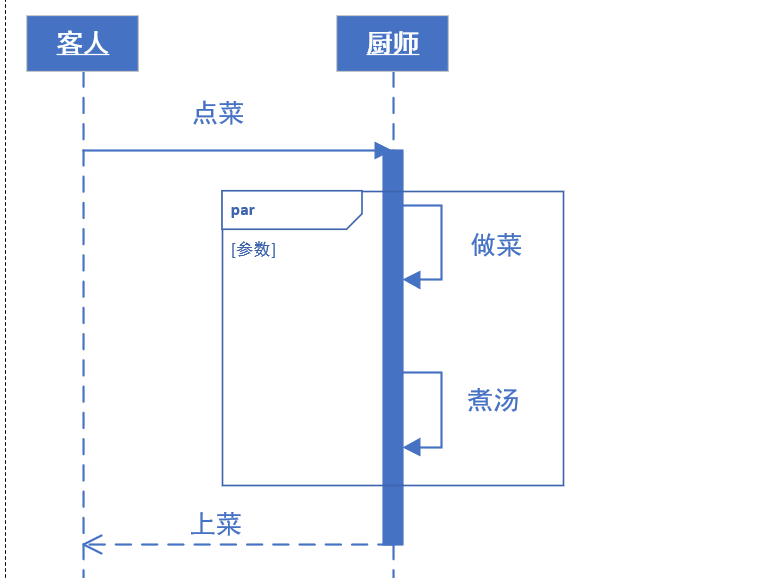
其中较为典型的如

选项(opt)

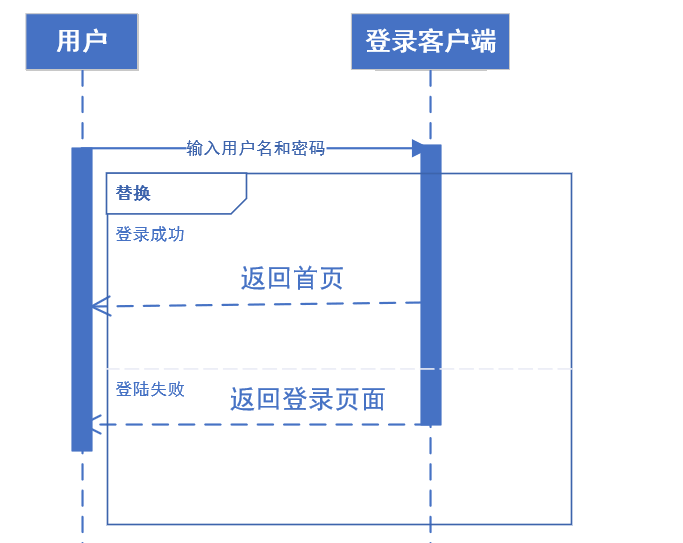
  
循环（Loop）



并行(Par)



替换(Alt)



Q:顺序图用来解决什么问题

A:用于表示程序执行时各个对象的交互过程（这些对象会按调用时间顺序排序）；

# 通信图

**1.定义与作用**

通信图也称为合作图。在UML1.0中被称为协作图，在UML2.0中则不再使用协作图的说法，而是明确定义为“通信图”。通信图是一种交互图，强调的是发送和接受消息的对象之间的组织结构。通信图使描述复杂的程序逻辑或多个平行事务变得容易，强调一组对象的上下文关系。

通信图特别适合用来描述少量对象的简单交互。随着对象和消息数量的增多，理解通信图将越来越困难。因此要用通信图描述一个系统，好的思路是采用模块化的方式，用通信图来描述一个小的模块，再将几个小模块抽象成一个大模块，再用通信图来表示几个大模块的交互。

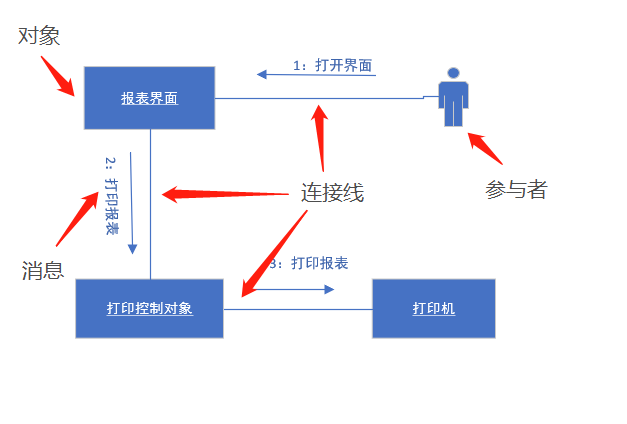
通信图的作用：

1. 通过描绘对象之间消息的传递情况来反映具体的使用语境的逻辑表达。
2. 显示对象及其交互关系的空间组织结构
3. 表现一个类操作的实现，因为通信图中包含类操作中用到的参数、局部变量和返回值。

**2.元素**

(1)通信图由以下基本元素组成：参与者(Actor)、对象(Object)、连接(Link，也称为链)、消息(Message)和控制点条件(Condition)。

(2)在UML中，使用实线表示两个对象之间的连接；通信图中的消息，由在连接上方的带有标记的箭头表示，同时可以用数字注明消息的次序。



（3）控制点条件：用来根据消息表达式的计算结果来限制消息的发送。以“[条件表达式]”格式表示。

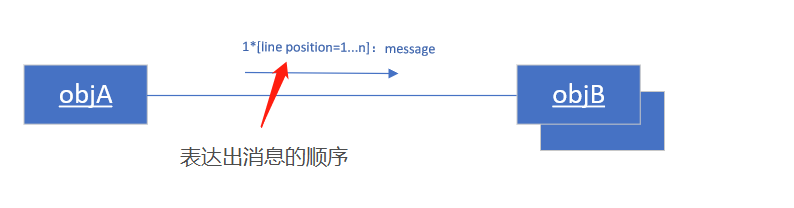
a.表示逻辑判断的控制点条件



b．表示发送给多个对象同时发送一个消息。



c．按顺序发送对象



(4)消息可能要求某个对象进行计算并返回结果的值。



(5)构造型

构造型可以在现有的UML元素的基础上创建新的元素

构造型的概念再使用UML建模工具的时候很有用。建模工具的一个重要特点是具备“字典”的功能，能够跟踪在模型中创建的所有的元素，包括类、用例、构件等。

# 三、通讯图与顺序图的比较

通信图与顺序图的异同点:

通信图与顺序图的共同点主要有如下3点：

1）主要元素相同。两种图中的主要元素都是对象与消息，且都支持所有的消息类型。

2）表达语义相同。两种图都是对系统中的交互建模，描述了系统中某个用例或操作的执行过程，二者的语义是等价的。

3）对象责任相同。两种图中的对象都担任了发送者与接收者的角色并承担了发送与接收消息的责任。通过对象之间消息的传递来实现系统的功能。

两种图之间的不同点也有如下3点：

1）通信图偏重于将对象的交互映射到连接它们的链上，这有助于验证类图中对应的类之间关联关系的正确性或建立新的关联关系的必要性。然而顺序图偏重描述交互中消息传递的逻辑顺序。因此通信图更适用于展示系统中的对象结构，而顺序图则擅长表现交互中消息的顺序。

2）顺序图可以显式地表现出对象创建与撤销的过程，而在通信图中，只能通过消息的描述隐式地表现这一点。

3）顺序图还可以表示对象的激活情况，而对于通信图来说，由于缺少表示时间的信息，除了对消息进行解释，无法清晰地表示对象的激活情况。

Q:请说出庭训徒和顺序图异同点各一个