第12章 通信图(协作图)
(communication diagram)

学习目标

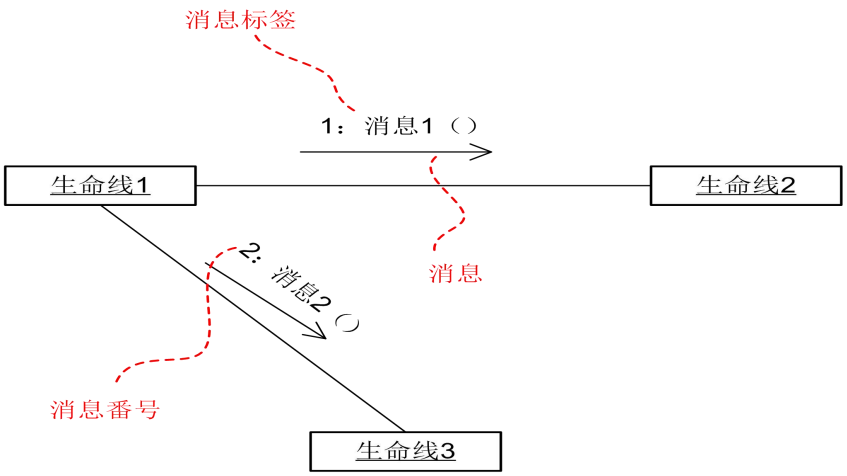
- 学习完本章节，要求达到以下状态：
  - 能够说明通信图的表示方法和使用方法
  - 能够读懂通信图并理解其中的含义
  - 能够说明顺序图和通信图的异同点
  - 能够说明类图和交互图的一致性

协作(Collaboration)图

- **协作图**包含一组对象和链(link)，用于描述系统的行为是如何由系统的成分协作实现的。
- A **collaboration diagram** is a diagram that shows interactions organized around roles—that is, slots for instances and their links within a **collaboration**.

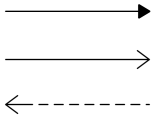
通信图的概要

- 通信图同顺序图一样，都属于交互图中的一种。
- 顺序图是按照时间顺序来配置消息描述生命线之间的消息交互，通信图是着眼于生命线之间的链接来描述生命线之间的消息交互。
- 顺序图和通信图可以互相转换。
- 通信图由以下元素构成：
  - 生命线
    - 通信图中的生命线同顺序图中的生命线基本一致，唯一不同的是通信图中生命线矩形框下面不带虚线。
  - 消息
    - 生命线之间用实线进行连接，实线上面标明消息方向和消息标签。



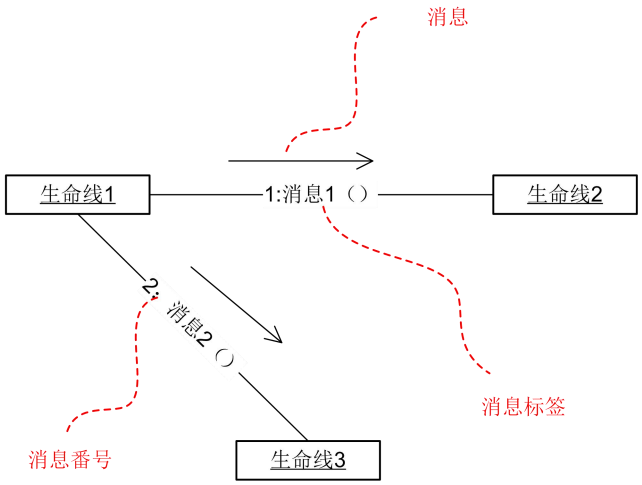
消息

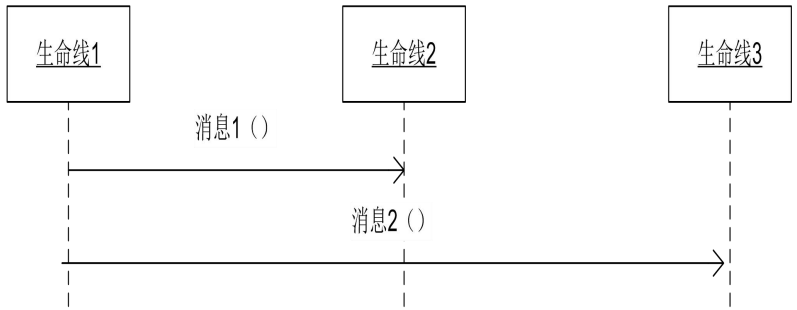
- 消息种类：
  - 同步消息
  - 异步消息
  - 返回消息



- 顺序图中消息是按照时间顺序从上到下进行排列的，因此表征消息发送的先后顺序的消息番号可以省略。

  通信图中的消息没有按照时间先后顺序进行组织，需要注明消息番号。





消息的标签格式

◆ 通信图中的消息标签格式:

前置项/ 消息番号 [监护条件] *[循环条件] : 返回值 := 消息名(参数1, 参数2.。。)

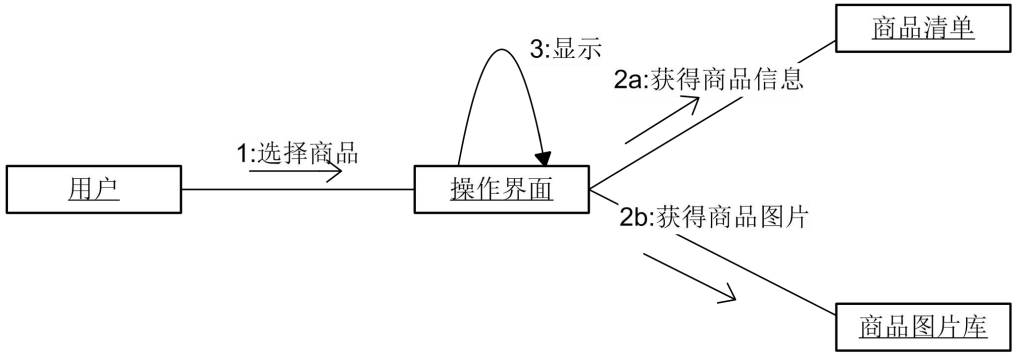
- ■ 前置项
  - 用来指定发送该消息之前必须要发送完成的消息。
  - 被指定为前置项消息序号之后加“/”来表示。
  - 当有多个前置项时, 之间用逗号进行分隔。
- ■ 消息番号
  - 表明消息执行的先后顺序。
- ■ 监护条件
  - 表示发送该消息必须具备的前提条件。
- ■ 循环条件
  - 表示重复发送该消息的次数条件。

例: 1b,1c/ 2 [ID号码≠0] *[i=1..n]: 商品信息 : = 取得商品信息(商品ID)

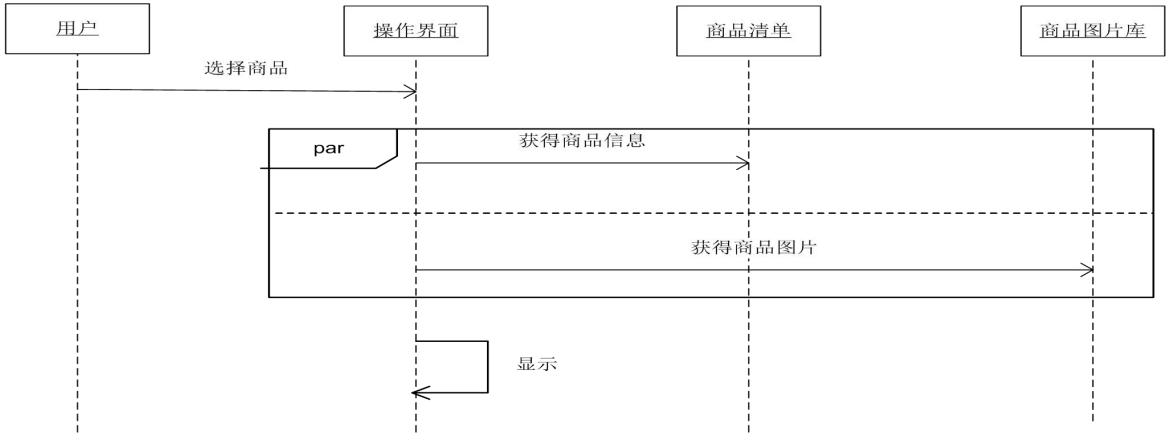
在消息1b和消息1c发送完毕前提下, 当商品ID号码不为零时重复发送消息番号为2的消息, 消息名称为“取得商品信息”, 消息参数为“商品ID”,消息处理的返回值返回给变量“商品信息”。

并行处理的表示方法

- 多条消息的同时并行发送称为并行处理。
- 在顺序图中，用组合片段“par”来表示消息的并行发送。
- 在通信图中，为了表示并行发送的消息，相同数字的消息番号后面跟上不同字母来加以区分。



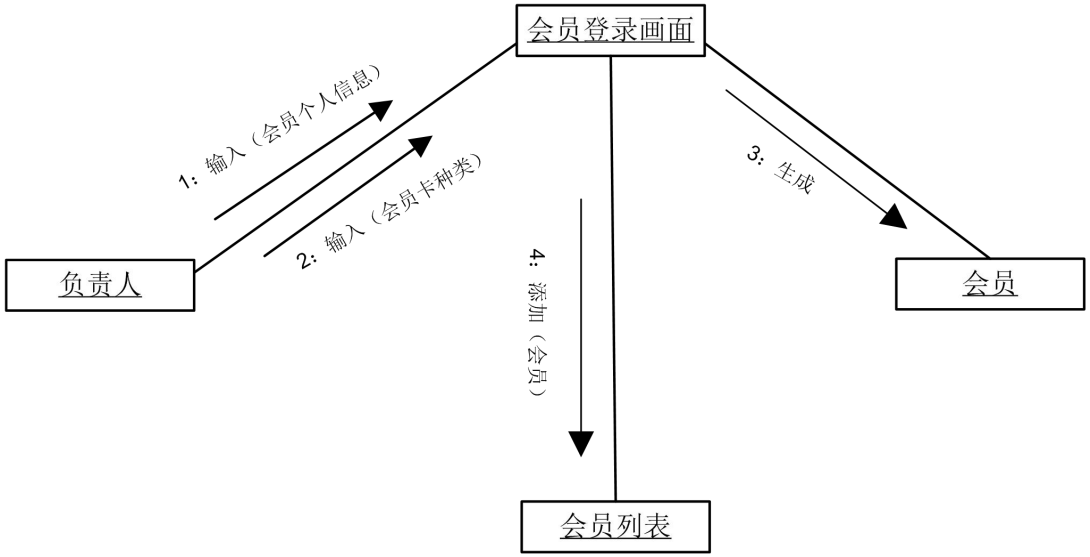
消息“获得商品信息”和“获得商品图片”是同时并行发送的消息。



例题:画通信图

- 把教材P112页所描述的脚本用通信图进行表示。

题解：



类图和交互图的一致性

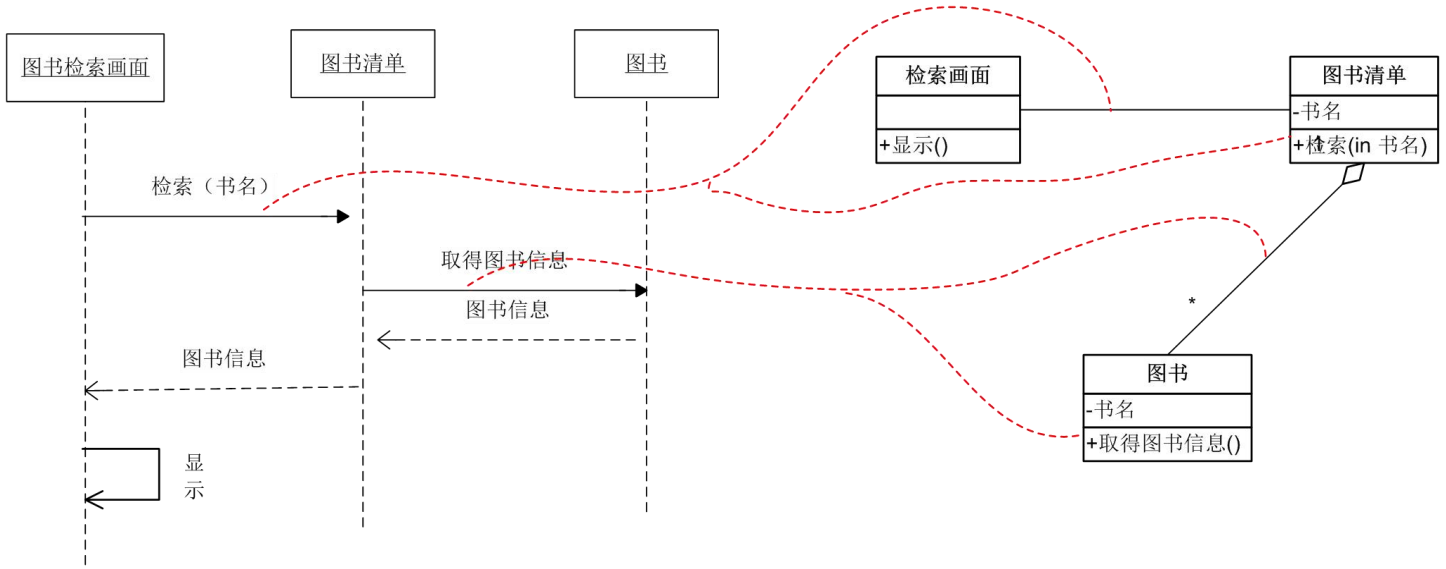
◆ 类图表示的是系统的静态结构，交互图表示的是系统的动态行为。两者之间要保持一致性。

◆ 保持类图和交互图之间的一致性观点有：

■ 交互图中的生命线对应于类图中适当的类。

■ 交互图中有消息交互的两个生命线，在类图中对应的两个类之间应该有一定的关系。

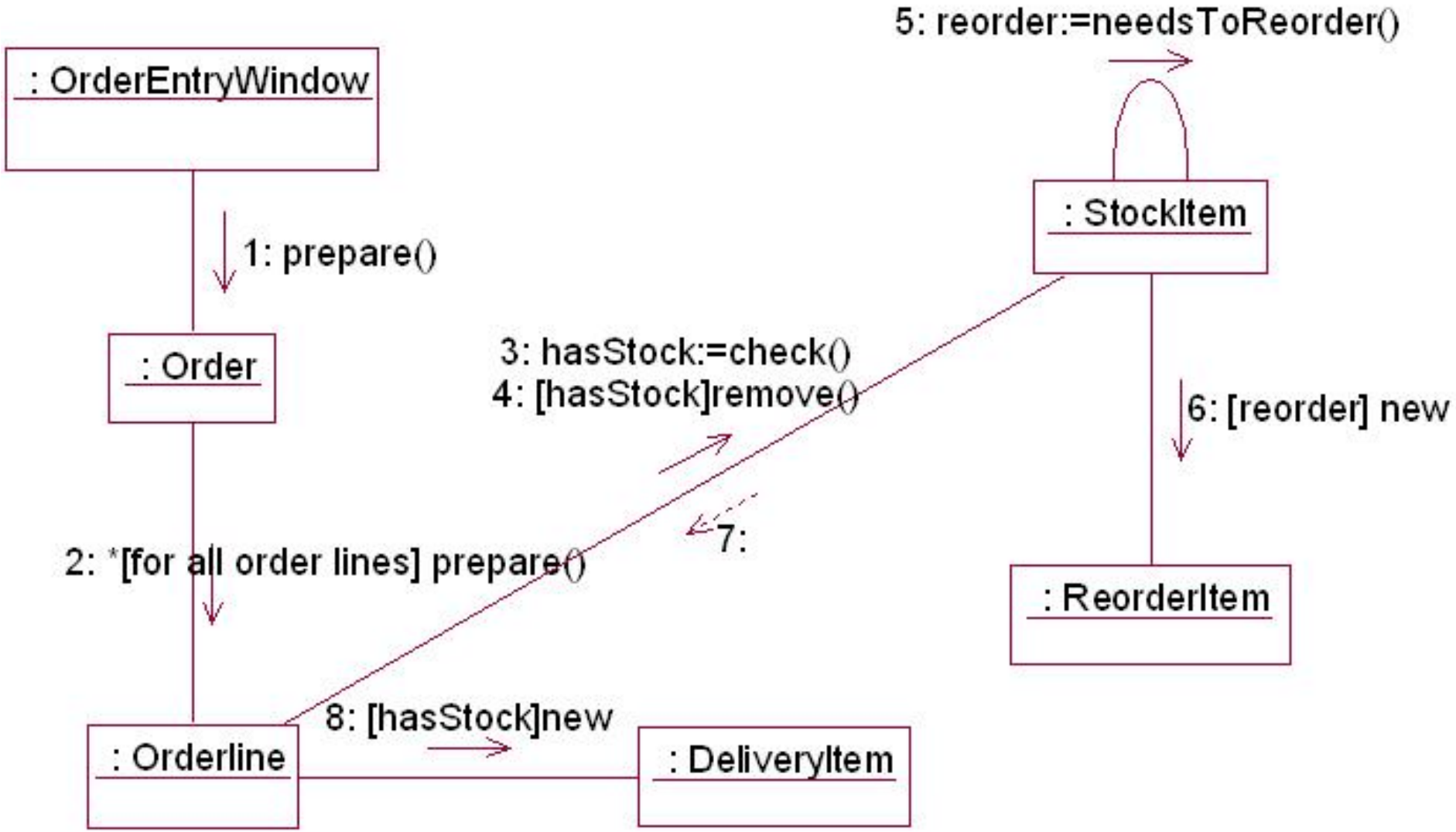
■ 交互图中的消息，在类图中接受该消息的生命线所对应的类中应该有某个操作与之相对应。



建立collaboration图的步骤

1. 确定交互过程的上下文(context)；
2. 识别参与交互过程的对象；
3. 如果需要，为每个对象设置初始特性；
4. 确定对象之间的链(link)，以及沿着链的消息；
5. 从引发这个交互过程的初始消息开始，将随后的每个消息附到相应的链上；
6. 如果需要表示消息的嵌套，则用Dewey十进制表示法；
7. 如果需要说明时间约束，则在消息旁边加上约束说明；
8. 如果需要，可以为每个消息附上前置条件和后置条件。

Collaboration图的例子：由顺序图转换来的协作图



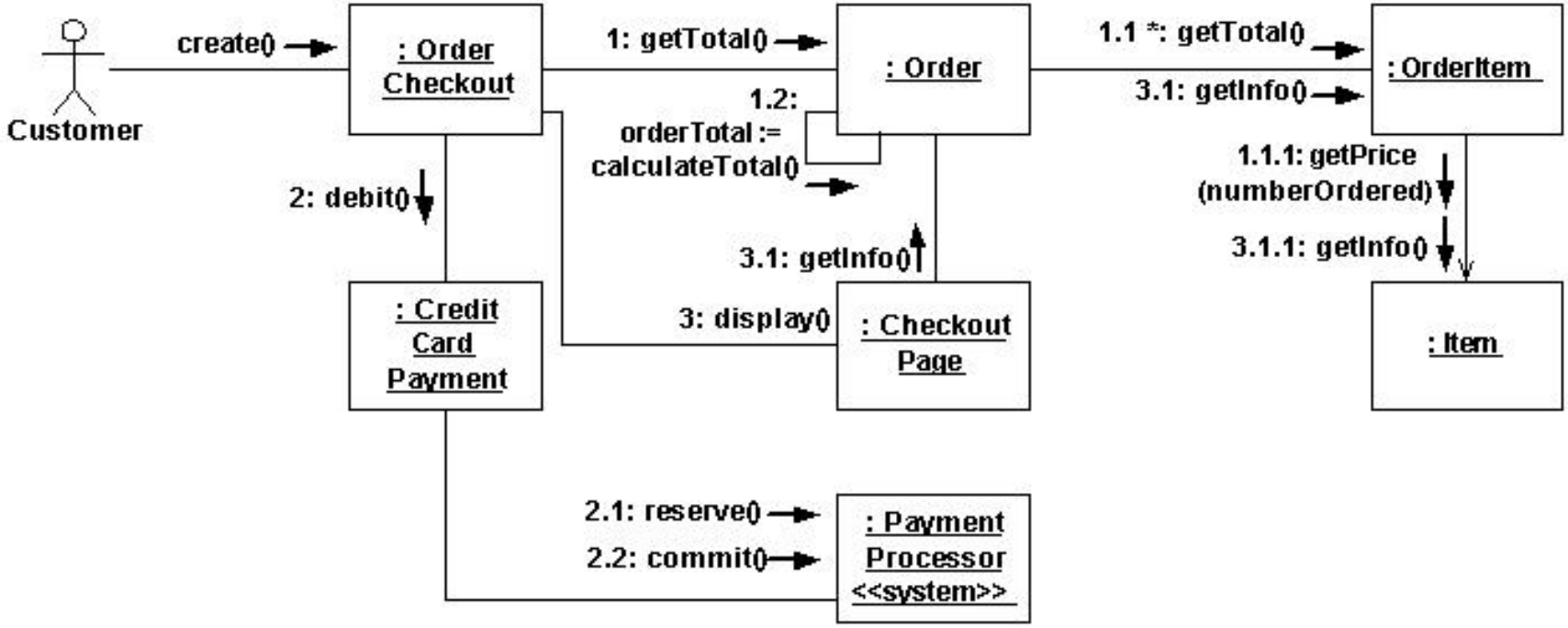
协作图建模风格

类似于顺序图：

- 把注意力集中于关键的交互。
- 对于参数，优先考虑使用参数名而不是参数类型。
- 不要对明显的返回值建模。
- 可以把返回值建模为方法调用的一部分。

其他的建模风格：

- 建模风格5：为每个消息画出箭头。
  - 便于可视化地确定流向一个给定对象的消息数量，以此判断该对象所涉及的潜在耦合情况，这通常是在对设计进行重构时要考虑的一个重要因素。
  - 例子：



- 建模风格6：注明导航性(Navigability)时要慎重。
  - 可以在协作图中对导航性建模，但这不是很常见，因为这太容易和消息流混淆。
  - 用UML类图来描绘导航性更好。
  - 例子：Slide 49

顺序图和协作图的比较

- 顺序图强调消息的时间顺序，协作图强调参加交互的对象的组织，两者可以相互转换。
- 顺序图不同于协作图的两个特征：
  - 顺序图有对象生命线
  - 顺序图有控制焦点
- 协作图不同于顺序图的两个特征：
  - 协作图有路径
  - 协作图必须有消息顺序号

- 和协作图相比，顺序图：
  - Take more space
  - Easier to follow algorithms
  - Easier to depict lifetimes
  - Easier to show multithreading in one object
  - More difficult to visualizing multiple concurrent flows of control
  - Do not show association links
- 顺序图可以表示某些协作图无法表示的信息；同样，协作图也可以表示某些顺序图无法表示的信息。

总结

- 顺序图和通信图之间的比较：
  - 顺序图是按照时间顺序从上到下排列消息来描述系统中生命线之间的消息交互情况。
    通信图是着眼于生命线之间的链接来描述生命线之间的消息交互情况。
  - 顺序图中的消息可以省略消息番号。
    通信图中的消息必须明确消息番号来表示消息发送的先后顺序。
  - 顺序图中，用组合片段“par”来描述消息的并行处理。
    通信图中，用“数字+字母”的消息番号定义方式来描述消息的并行处理。
- 描述系统静态结构的类图和描述系统动态行为的交互图之间要保持一致性：
  - 交互图中的生命线对应于类图中的类。
  - 交互图中有消息交互的生命线，对应在类图中的两个类之间应该有一定的关联。
  - 交互图中的消息对应于类图中接受该消息生命线所对应的类的某个操作。