第12章：交互图

引例：餐厅吃饭

- 思考下去餐厅吃饭（从进入餐厅到离开餐厅），顾客和服务员之间发生了什么事情？

1.我进入餐厅

2.服务员领我去座位

3.服务员给我菜单

4. 我选择菜式

5. 我向服务员指示点菜

。。。。。。此处省略N字

引例：餐厅吃饭

- 使用交互图（顺序图）可以做什么？

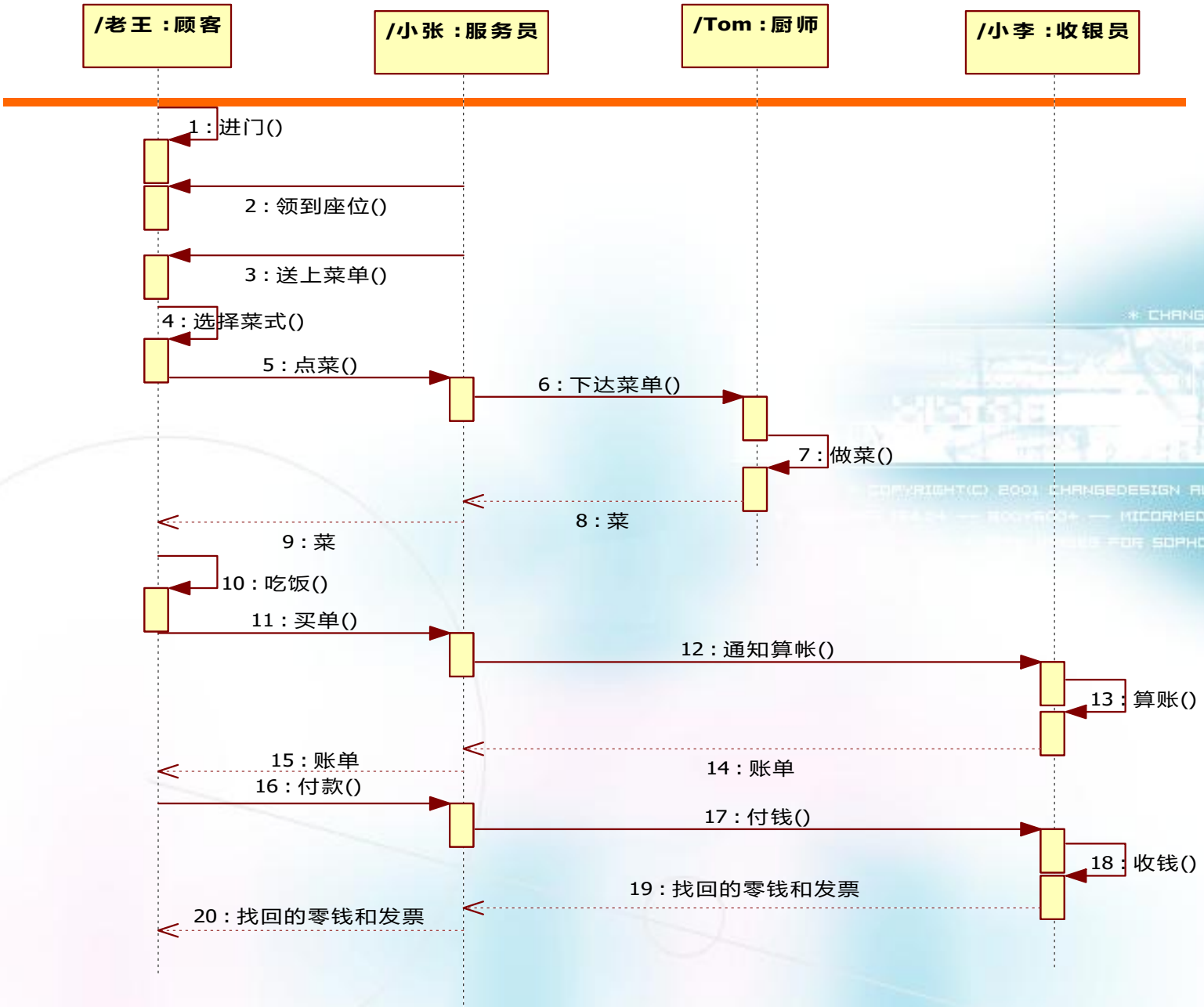
（1）顺序图以图形方式将复杂的交互按照时间顺序分解

（2）任何复杂的交互，其实都可以分解为自己与自己，自己与别人，别人与别人的多个简单交互

（3）从上到下，从左到右阅读交互图

引例：餐厅吃饭

- 假如要开发一个餐厅管理系统，考虑除了顾客和服务员，餐厅里还有谁？
- 厨师！！
- 收银员！！
- 请完善顺序图


/老王 :顾客
/小张 :服务员
/Tom :厨师
/小李 :收银员
1 : 进门()
2 : 领到座位()
3 : 送上菜单()
4 : 选择菜式()
5 : 点菜()
6 : 下达菜单()
7 : 做菜()
8 : 菜
9 : 菜
10 : 吃饭()
11 : 买单()
12 : 通知算帐()
13 : 算账()
14 : 账单
15 : 账单
16 : 付款()
17 : 付钱()
18 : 收钱()
19 : 找回的零钱和发票
20 : 找回的零钱和发票

本章内容

- 交互与交互图
- 如何阅读交互图
- 如何绘制交互图
- 交互图应用说明

本章内容

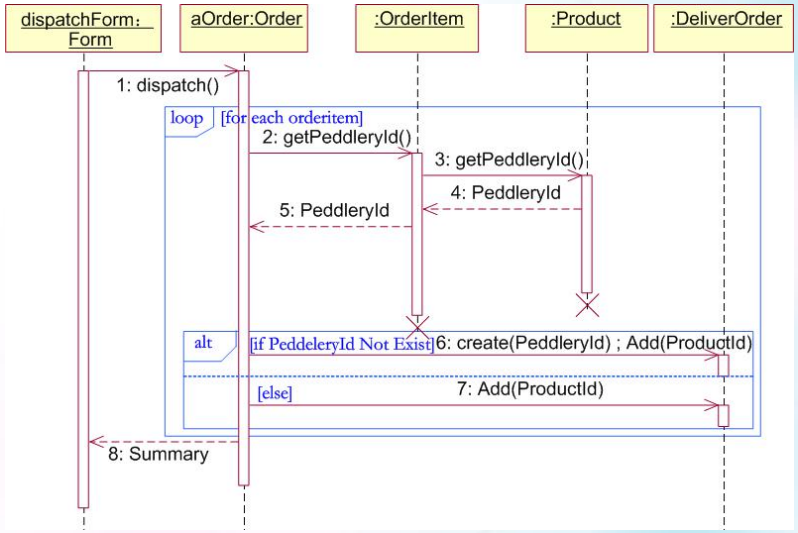
- 交互与交互图
- 如何阅读交互图
- 如何绘制交互图
- 交互图应用说明

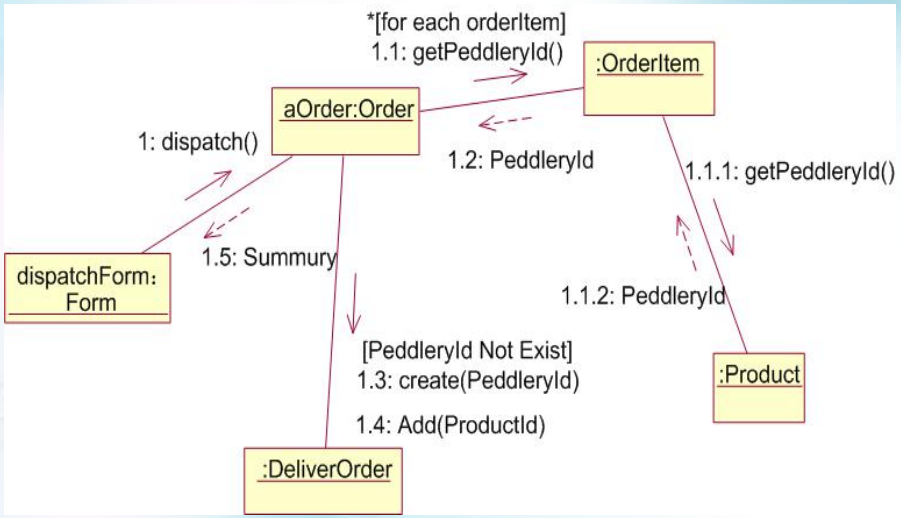
交互的概念

- 一次交互就是指在特定语境中，为了实现某一个目标，而在一组对象之间进行交换的一组消息所表示的行为

UML中的交互图

- 顺序图：顺序图是一种强调消息时间顺序的交互图，为读者提供了控制流随着时间推移的清晰的可视化轨迹
- 通信图：UML 2.0中的通信图实际上就是UML 1中的协作图，它强调的是参加交互的对象的组织，为读者提供了在协作对象结构组织的语境中观察控制流的一个清晰的可视化轨迹





UML流程分析三剑客

- **顺序图**

（1）强调对象之间的交互，信息传递很明确

（2）强调按时间顺序分别发生了什么事情

（3）不适合表达复杂的特殊流程（如循环，分支等）

- **活动图**

（1）强调每一泳道的角色做了什么事情，这些事情的先后顺序

（2）适合表示各种特殊流程，比如并发，分支等

- **状态机图**

（1）围绕某一对象/事物展开

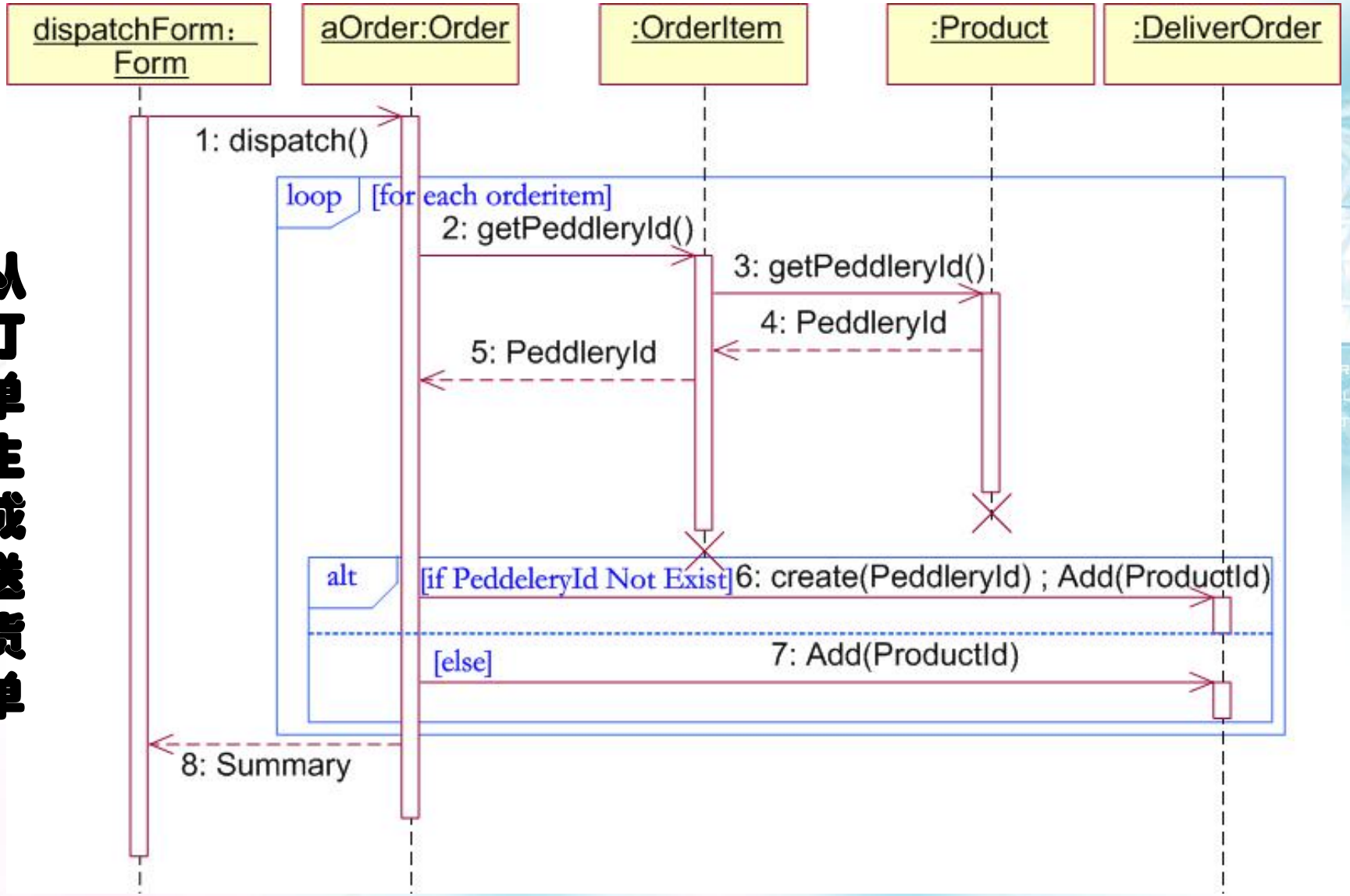
（2）该对象/事物有多种状态，状态会因为发生了一些事件而变化

本章内容

- 交互与交互图
- 如何阅读交互图
- 如何绘制交互图
- 交互图应用说明

阅读顺序图

从订单生成送货单

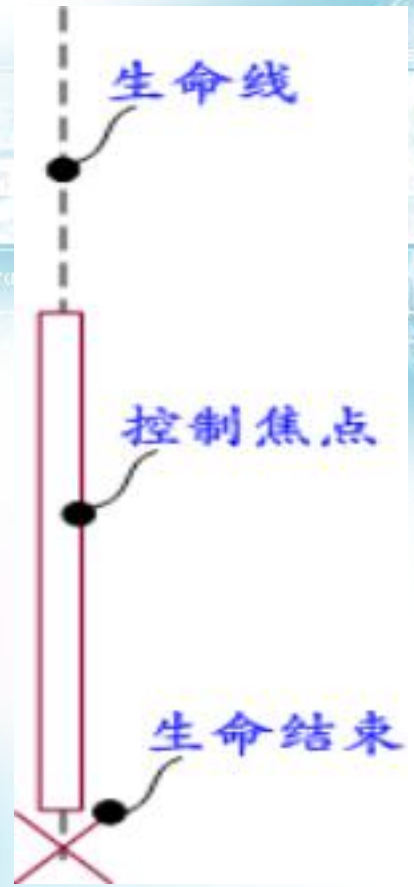


顺序图的主要元素

- 对象与参与者：最顶上一排矩形框。参与交互的对象既可以是具体的对象，又可以是匿名对象。

- 生命线：每个对象都有自己的生命线，
  对象生命线是一条垂直的虚线，
  用来表示一个对象在一段时间内存在。

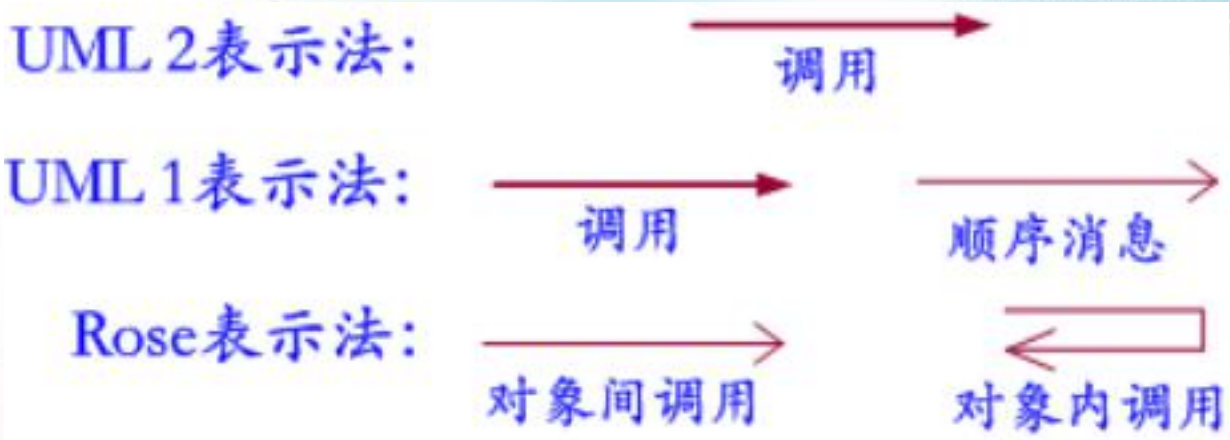
- 控制焦点：表示一个对象执行一个动作所经历的时间段。



顺序图的主要元素 call send

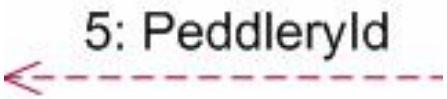
- 消息：用来描述对象之间所进行的通信。

- 消息分为五种：调用、返回、发送、创建和销毁
  - 调用：表示调用某个对象一个操作

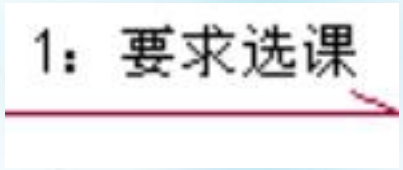


顺序图的主要元素

- 消息分为五种：调用、返回、发送、创建和销毁
  - 返回：表示被调用对象向调用者返回一个值。



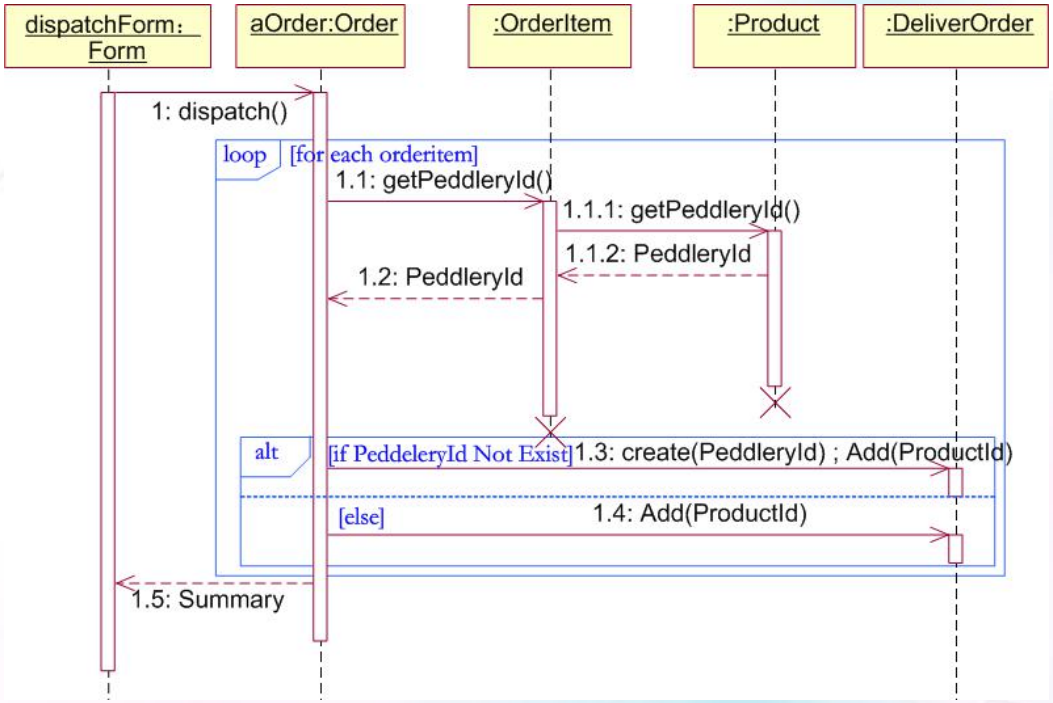
  - 发送：向对象发送一个信号，实现实例间通信的异步激发。

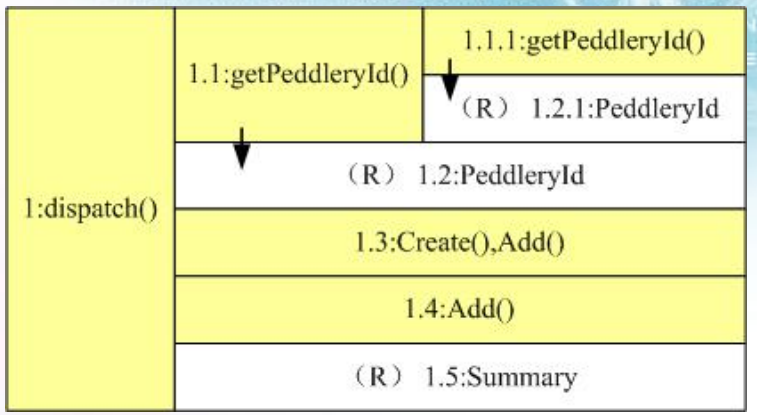


  - 创建和销毁：创建和销毁一个对象。

顺序图的主要元素

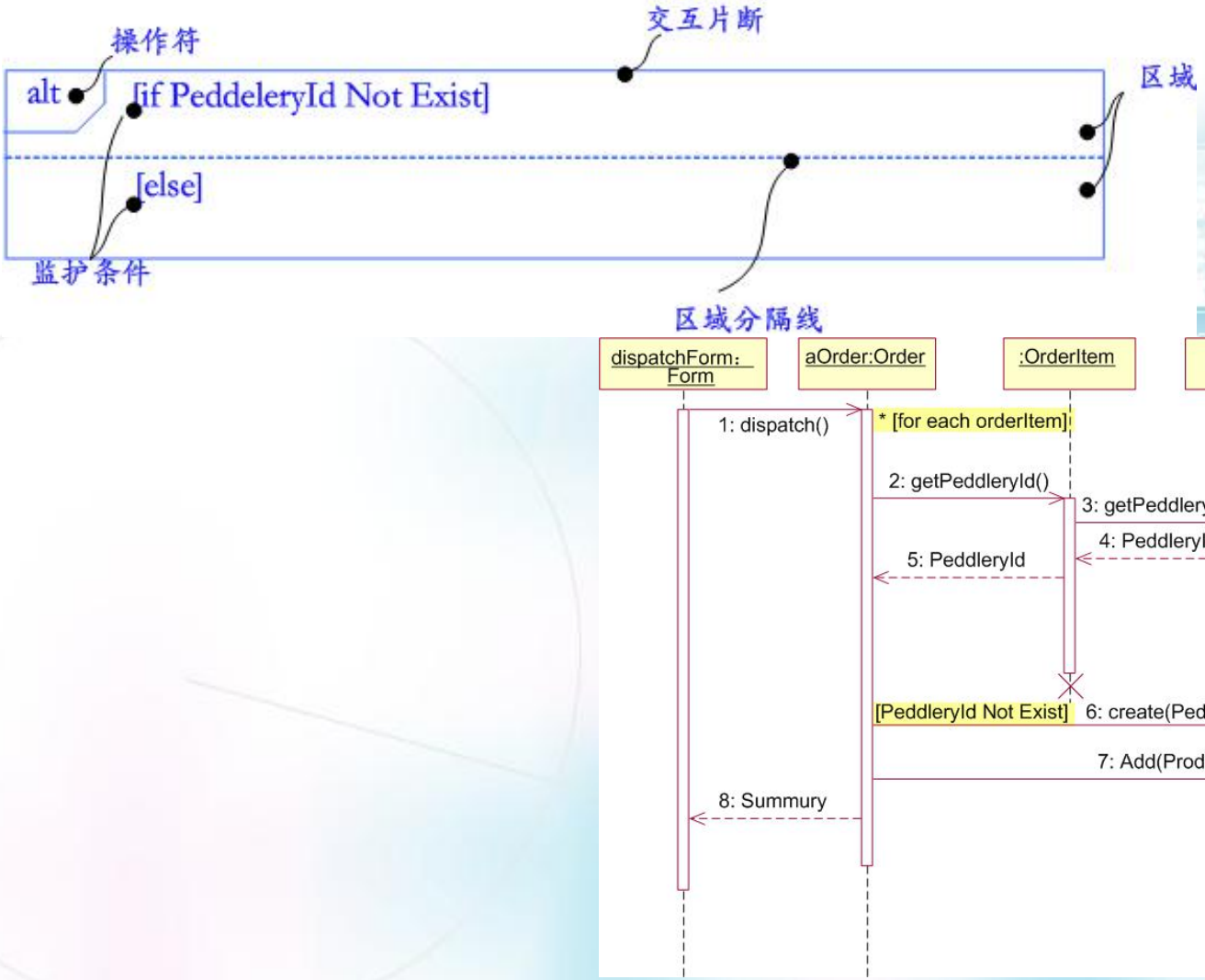
- 顺序编号：整个消息的传递过程形成一个完整的序列，通过在每个消息的前面加上一个用冒号隔开的顺序号来表示其顺序。除顺序编号外，还可以采用嵌套方案。





顺序图的主要元素

- 循环与分支



读图小结

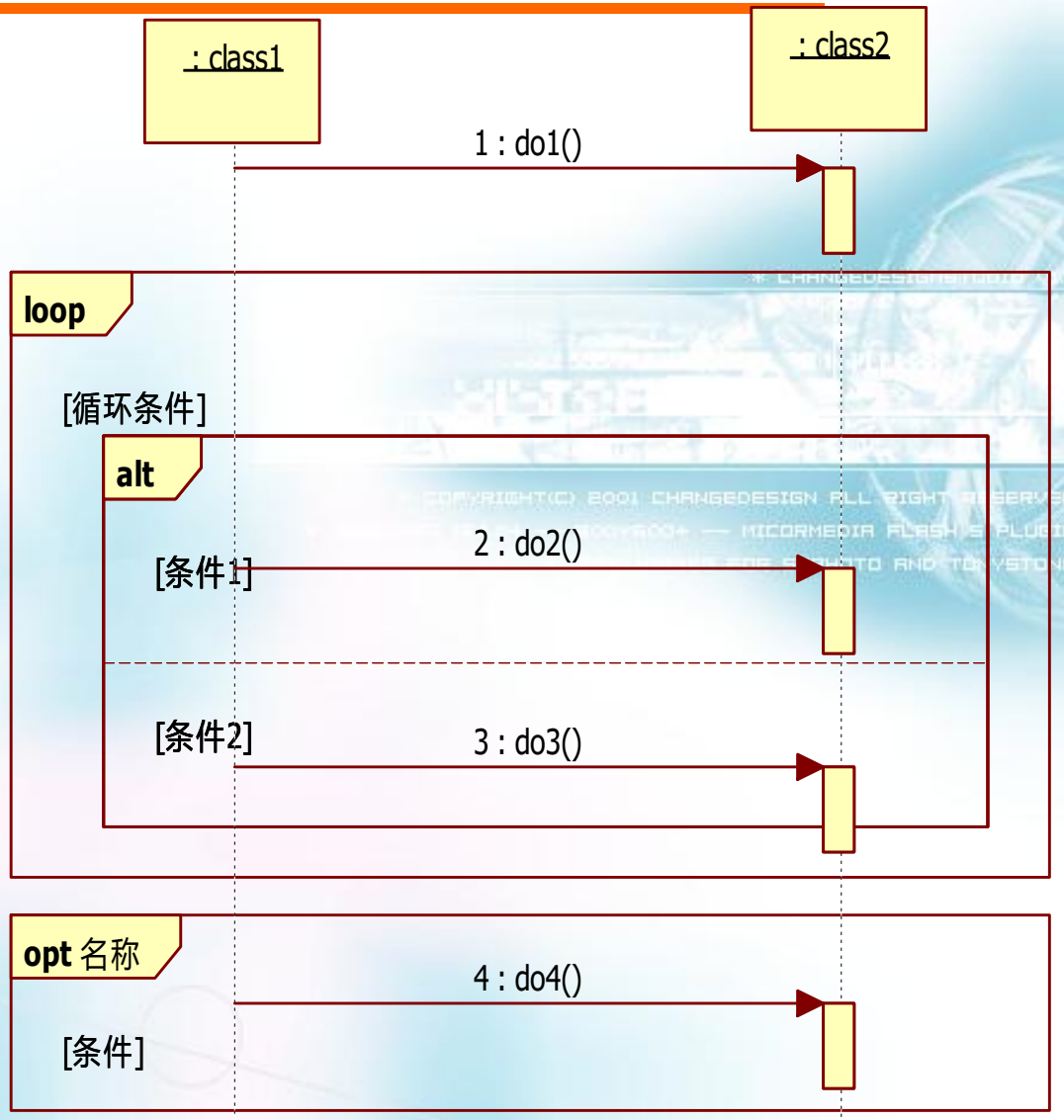
- 在dispatchForm（分发窗体）中，对于某个已支付的Order进行分发时，就会调用该订单（一个Order类的实例对象aOrder）的dispatch()方法
- dispatch()方法将逐个调用该Order对应的所有OrderItem对象的getPeddleryId()方法还获取供应商ID（PeddleryId），而OrderItem对象则是通过其所对应的Product对象来的getPeddleryId()方法来获取供应商ID

读图小结

- 当Order的实例对象aOrder得到返回的PeddleryId后，根据该值判断是否已经有相对应的DeliverOrder对象，如果没有就创建它（调用create(PeddleryId)），然后再将对应的Product添加到这个DeliverOrder对象中。否则就直接添加到相应的DeliverOrder对象中

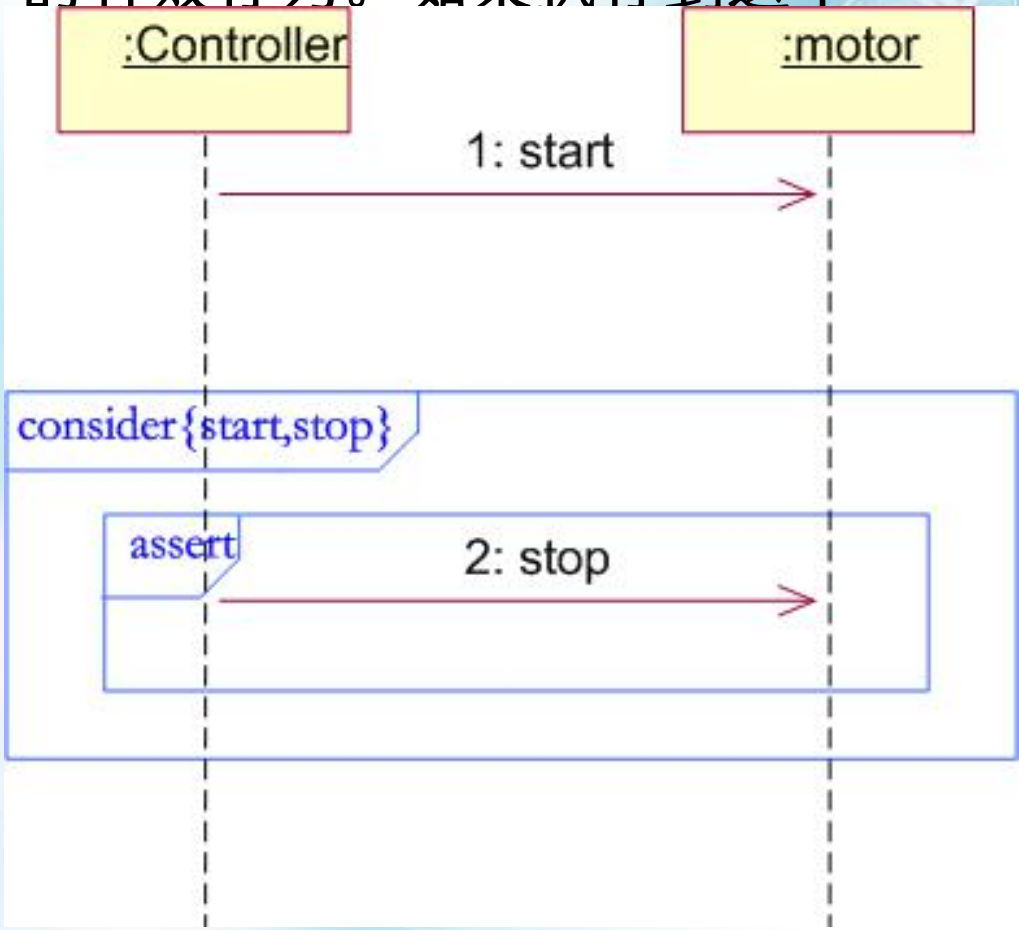
交互片断操作符loop，alt，opt

- Loop：循环，在满足循环条件的前提下，不断重复做某些事。
- Alt：alternative的缩写，条件分支，根据不同条件选择不同分支。（if..else...或case语句 ）
- Opt：optional的缩写，可选分支，满足一定条件执行该分支，否则跳过。（if语句）



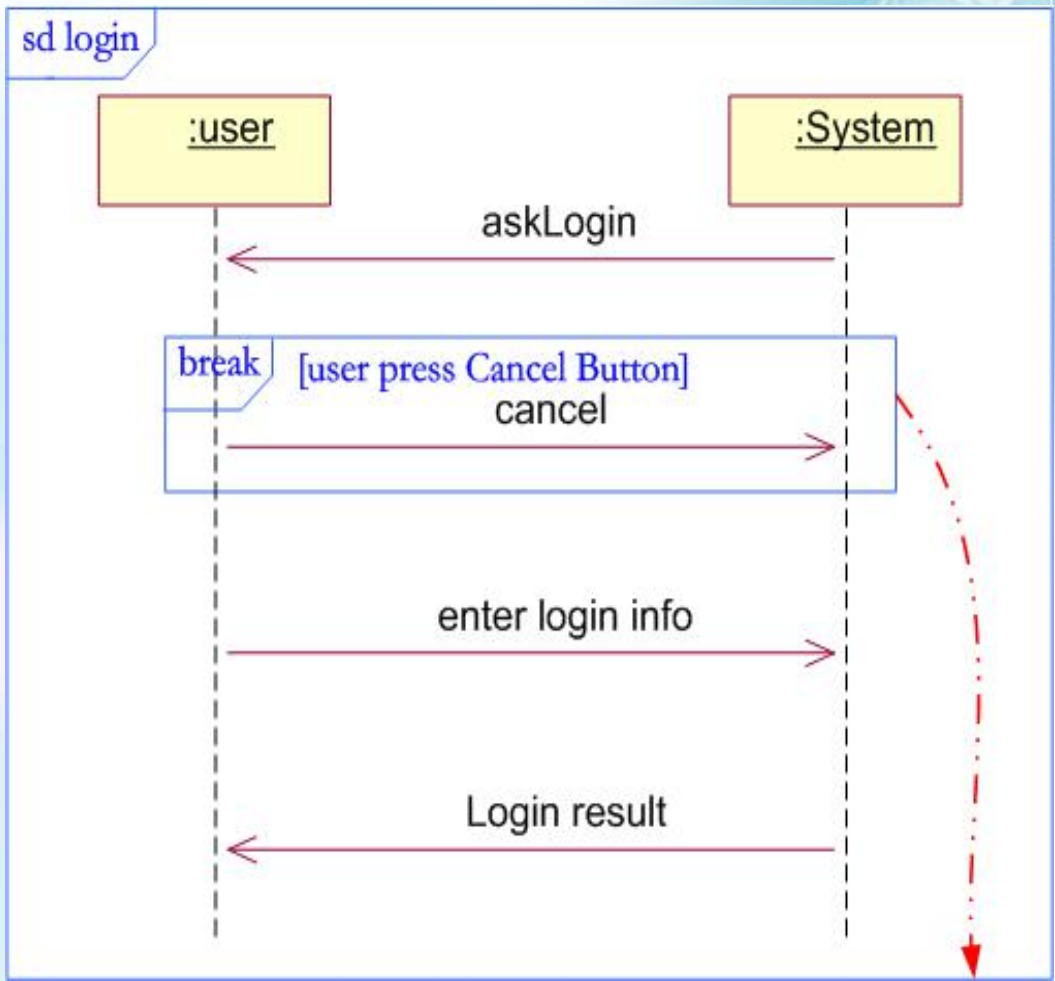
交互片断操作符assert

- 交互片断操作符**assert**是用来表示内容所描述的行为是执行过程中那个时刻唯一的有效行为。如果执行到这个片断的前面，则说明该片断就一定会发生。它通常和**ignore**或**consider**一起使用，以断言某种特定种类的消息行为



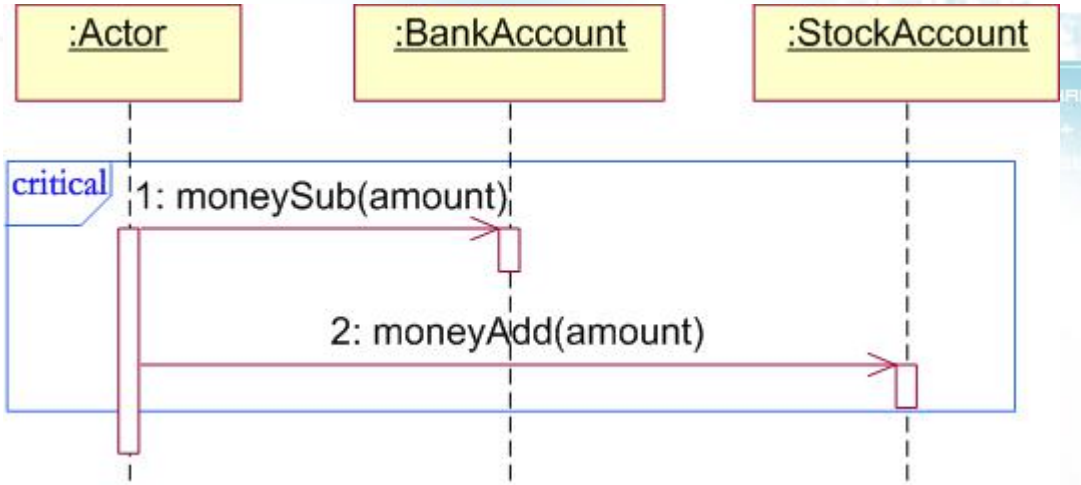
交互片断操作符break

- 交互片断操作符**break**和循环语句的**break**有点类似，通常**break**用来定义一个含有监护条件的子片断。如果监护条件为“真”则执行子片断，而且不执行包含子片断的图中其它交互将不会执行；如果监护条件为“假”，那么执行将正常地继续进行



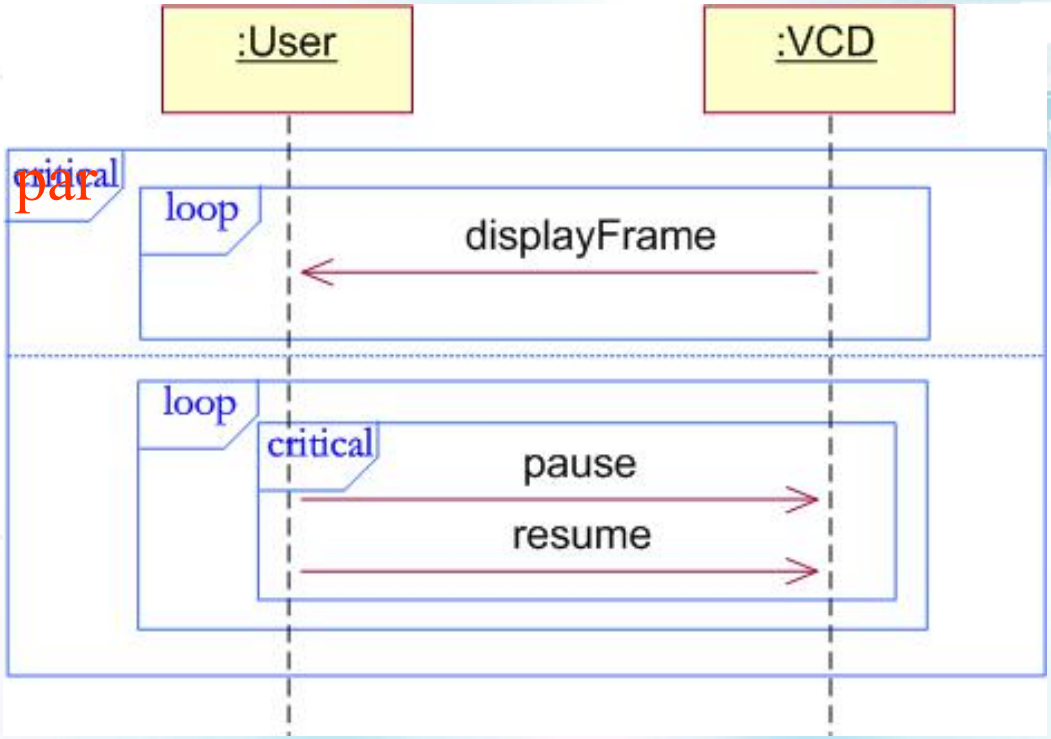
交互片断操作符critical

- 表示该子片断是“临界区域”，在临界区域中生命线上的事件序列不能够和其它区域中的任何其他事件交错。通常用来表示一个原子性的连续操作，例如事务性操作



交互片断操作符par

- 用来表示“并行”的，也就是用来表示两个或多个并发执行的子片断，并行子片断中单个元素的执行次序可以以任何可能的顺序相互操作

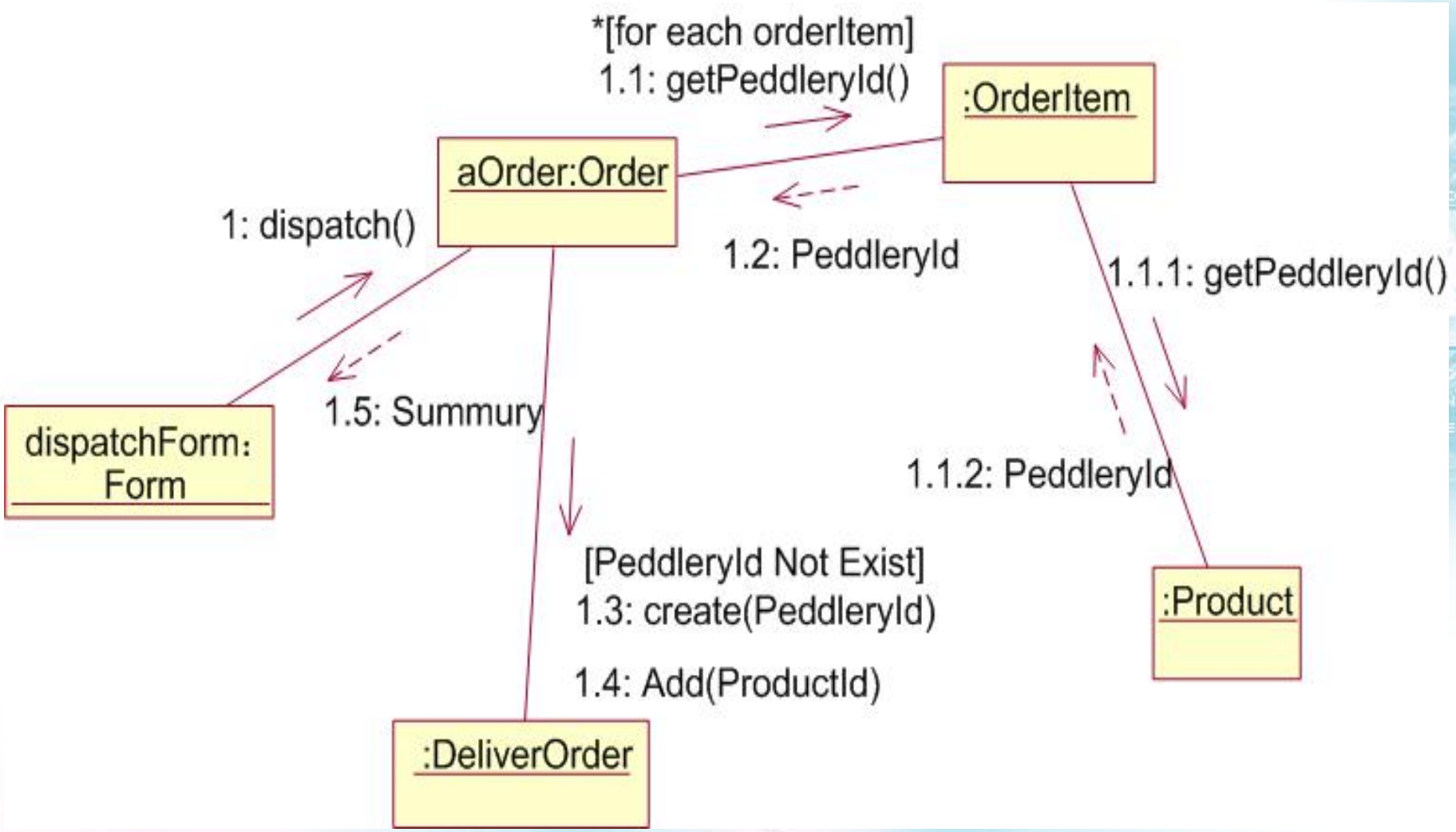


交互片断操作符ref

- 在一个交互图中，我们可以引用其它的交互图，其表示的方法是用一个矩形，加上**ref**操作符，并写明引用的交互图名称即可

图	表示法	图	表示法
类图	class	对象图	object
包图	package	用例图	use case
顺序图	sd	通信图	comm
定时图	timing	活动图	activity
交互概观图	intover	状态机图	statemachine
构件图	component	部署图	deployment

阅读通信图

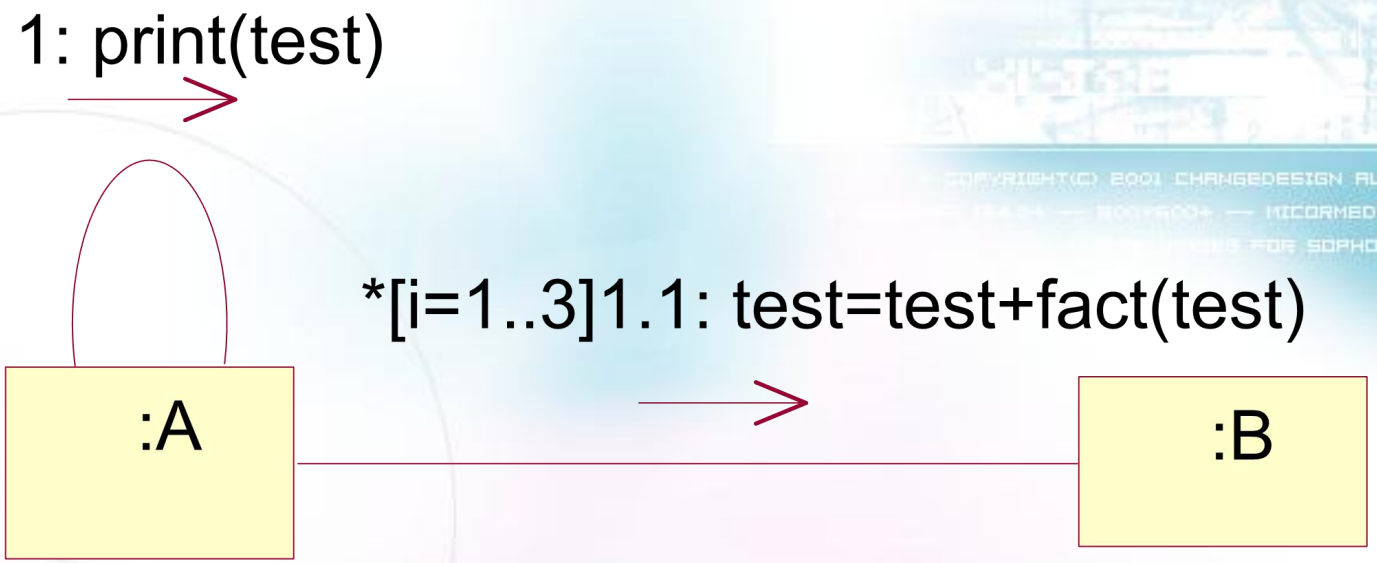

*[for each orderItem]
1.1: getPeddleryId()
:OrderItem
aOrder:Order
1: dispatch()
1.2: PeddleryId
1.1.1: getPeddleryId()
dispatchForm:
Form
1.5: Summury
1.1.2: PeddleryId
[PeddleryId Not Exist]
1.3: create(PeddleryId)
:Product
1.4: Add(ProductId)
:DeliverOrder

通信图主要元素

- 链：Object Link，是用来表示对象之间的语义连接，一般而言，链是关联的一个实例。
- 消息编号：，一种是无层次编号，它简单直观；另一种是嵌套的编号，它更易于表示消息的包含关系。
- 迭代：即循环。用一个迭代符号*，和迭代表达式来表示。
- 监护条件：通常是用来表示分支的，也就是表示“如果条件为true，才发送消息”，以[条件表达式]形式表示。

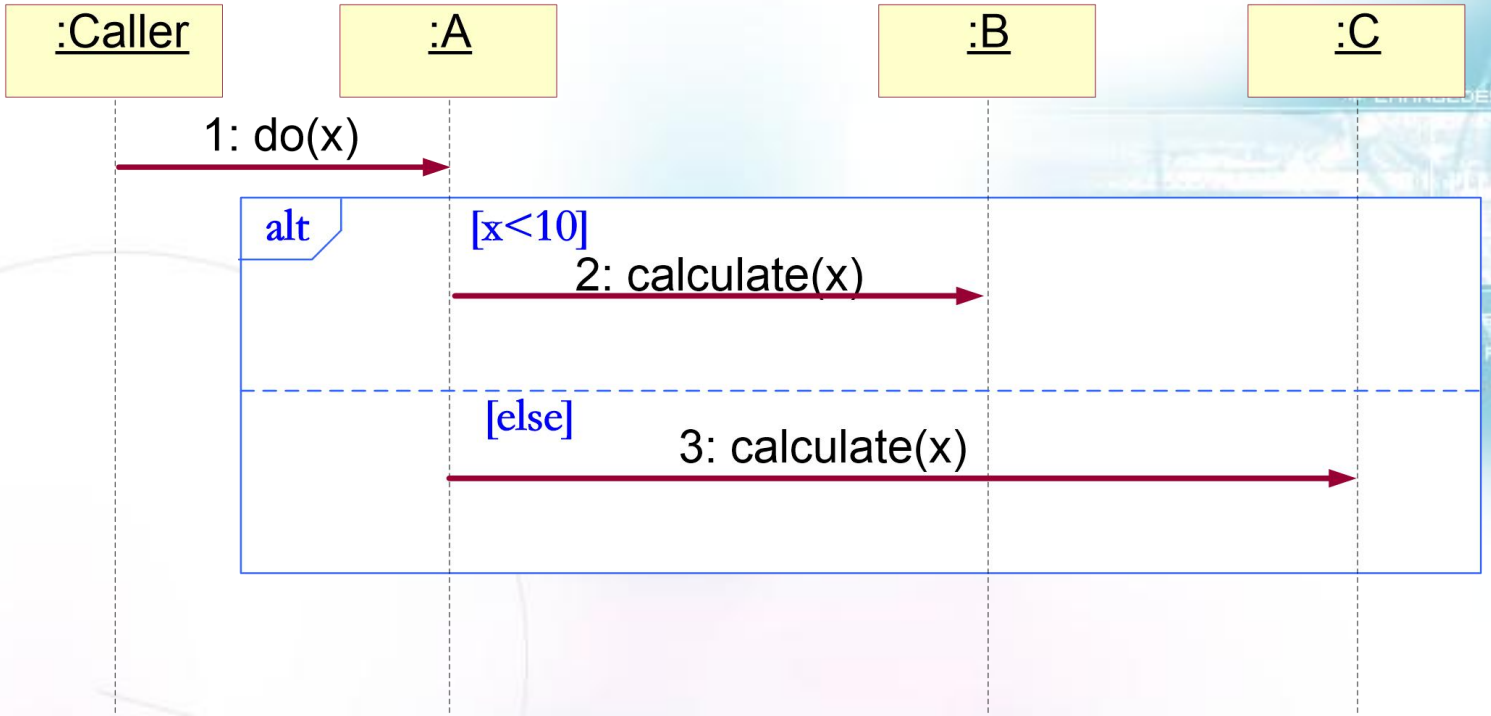
练一练（一）

- 假设有一系统的通信图局部如图所示，print方法的功能是将传入参数的值打印在屏幕上；fact方法是用来计算阶乘的，test的初值为1。那么将打印出什么？



练一练（二）

- 请说明以下顺序图的含义（使用伪代码演示）：



本章内容

- 交互与交互图
- 如何阅读交互图
- 如何绘制交互图
- 交互图应用说明

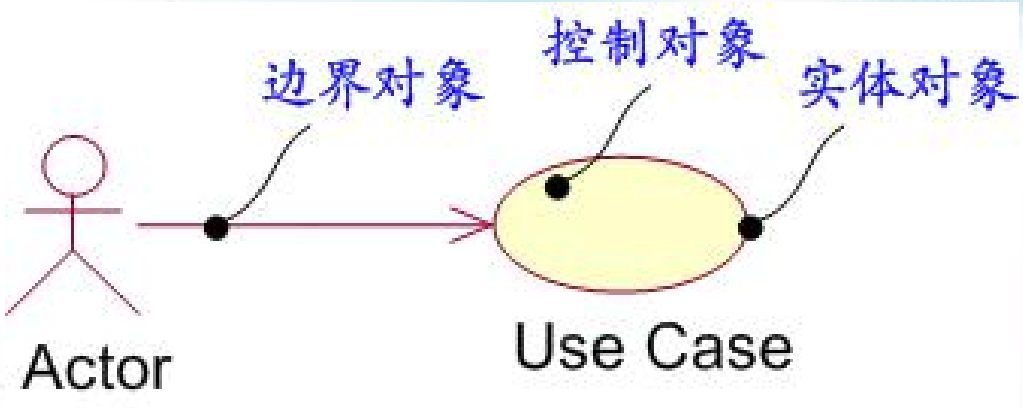
准备工作

- 根据自己的喜好和实际的表现需要来选择顺序图或通信图。顺序图强调时间的顺序，通信图强调对象间结构和职责。这两种图是同构的，可以互相转换，并不产生信息的丢失。

- 先确定参与交互的对象、对象之间的关系（通信图），然后确定对象间的消息交互流程（用同步调用、异步消息、返回消息表示），并利用交互片断（顺序图）或迭代标记及监护条件来表示循环和分支结构。

使用方法：鲁棒分析

- **Robustness分析不是UML模型的一部分，它是一个强大的草图工具，是介于分析和设计之间的一种有效工具。**
- **在Robustness分析中，将应用边界类、控制类和实体类。**
- **从一个用例中抽取三类对象的方法：**



鲁棒分析—1、从事件流开始

- 以个人图书管理系统的“新增书籍”用例为例

3.1 基本事件流

1. 图书管理员向系统发出“新增书籍信息”请求；
2. 系统要求图书管理员选择要新增的书籍是计算机类还是非计算机类；
3. 图书管理员做出选择后，显示相应界面，让图书管理员输入信息，并自动根据书号规则生成书号；
4. 图书管理员输入书籍的相关信息，包括：书名、作者、出版社、ISBN号、开本、页数、定价、是否有 CDROM；
5. 系统确认输入的信息中书名未有重名；
6. 系统将所输入的信息存储建档。

3.2 扩展事件流

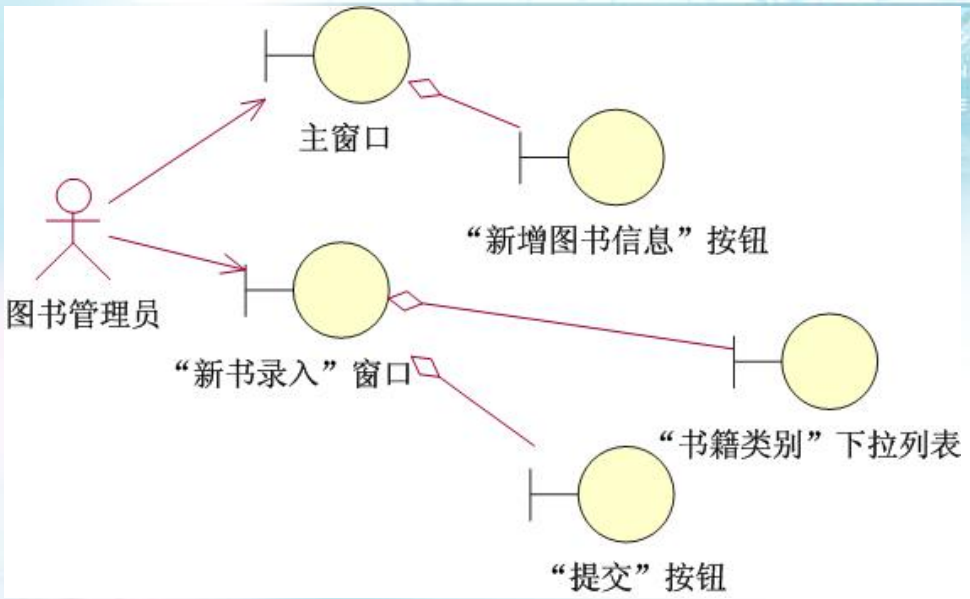
5a）如果输入的书名有重名现象，则显示出重名的书籍，并要求图书管理员选择修改书名或取消输入；

5a1）图书管理员选择取消输入，则结束用例，不做存储建档工作；

5a2）图书管理员选择修改书名后，转到 5）

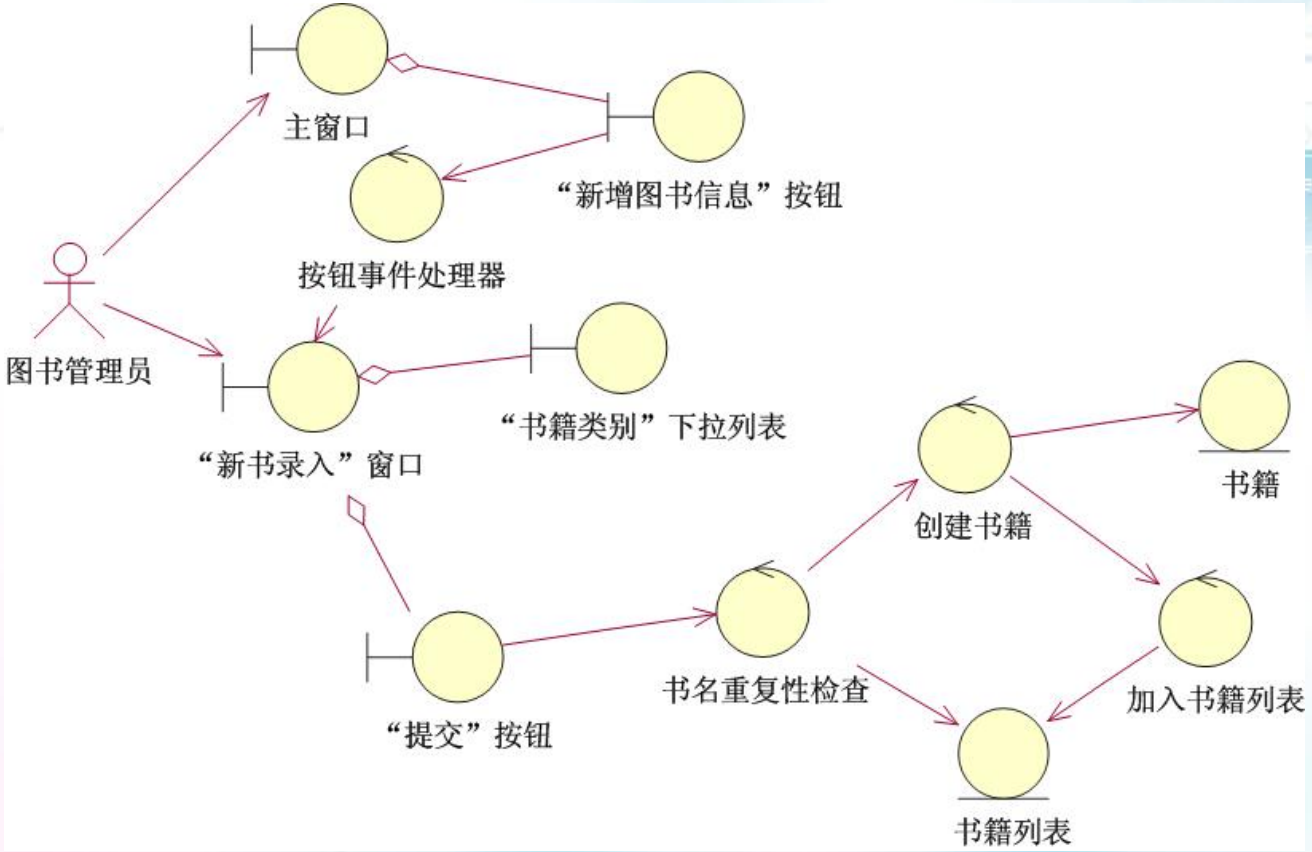
鲁棒分析—2、寻找边界对象

- 图书管理员向系统发出“新增书籍信息”请求——主窗口、“新增书籍信息”按钮
- 系统要求图书管理员选择要新增的书籍是计算机类还是非计算机类——书籍类别列表框。
- 图书管理员做出选择后，显示相应界面，让图书管理员输入信息，并自动根据书号规则生成书号——“新书信息录入”窗口及辅助的“提交”按钮



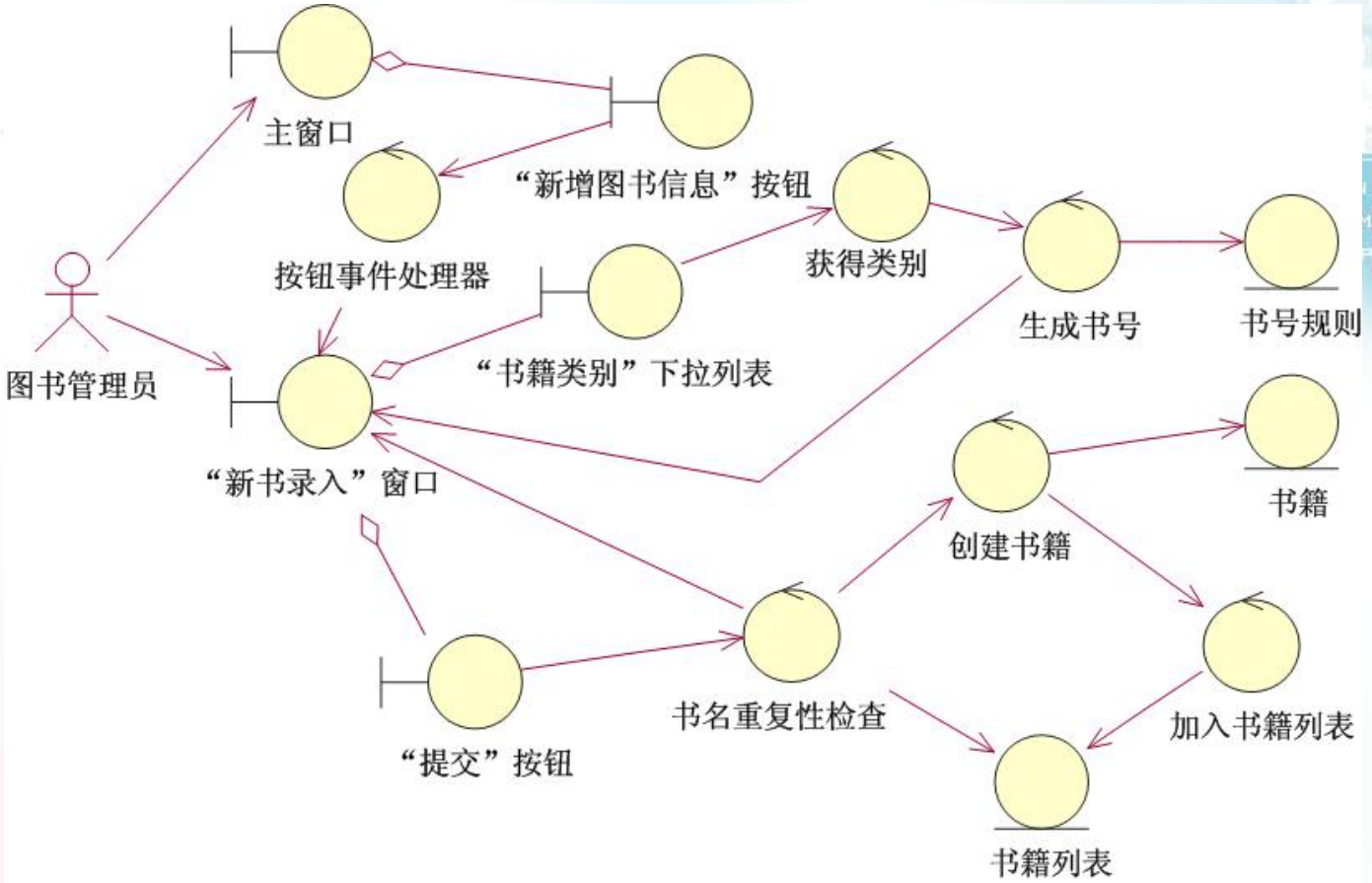
鲁棒分析—3、寻找控制对象和实体对象

- 根据事件流中的步骤5，以及扩展路径的描述，就可以在原图上增加相应的控制对象，得到更进一步的Robustness分析图。

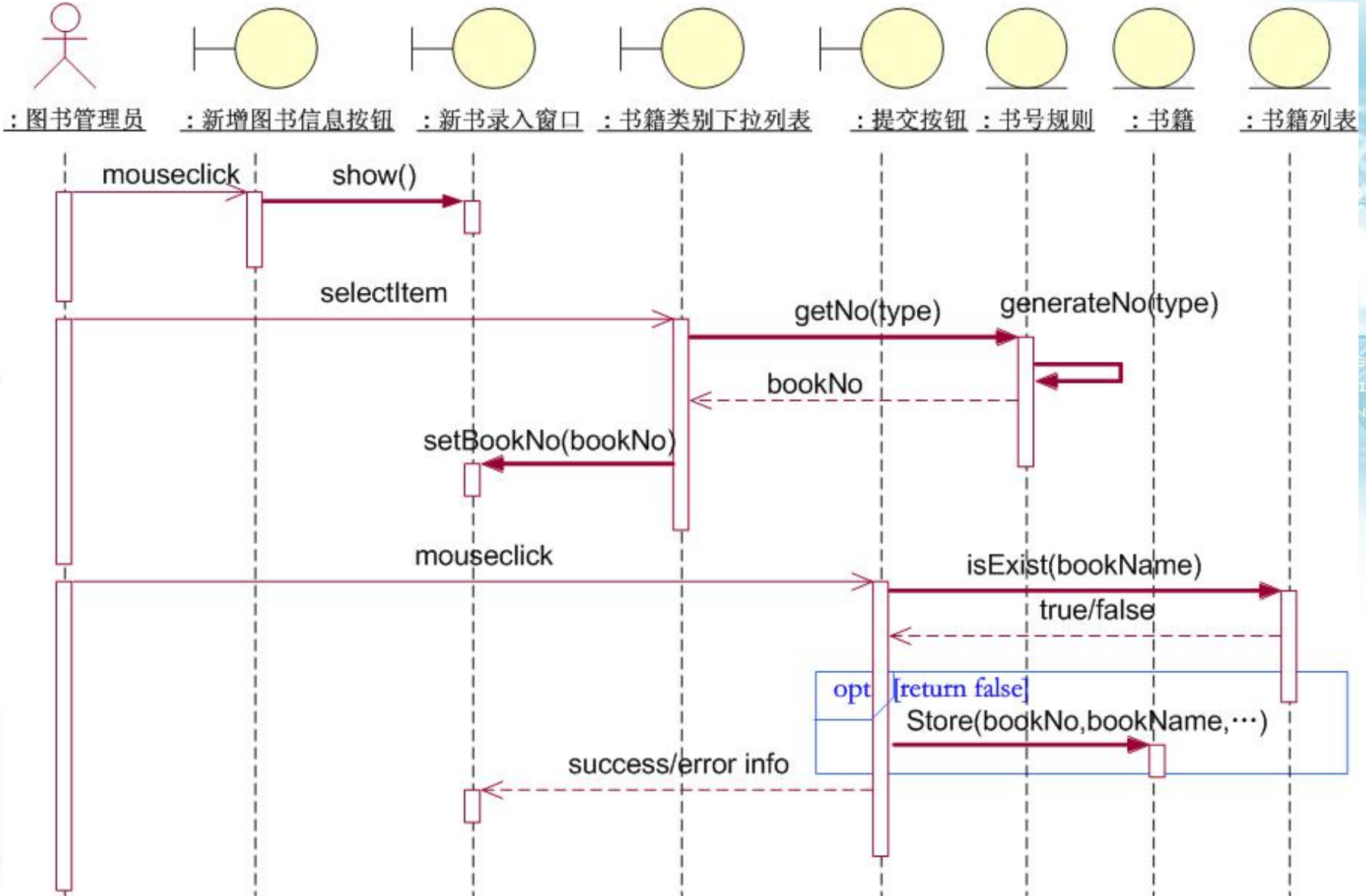


鲁棒分析—3、寻找控制对象和实体对象

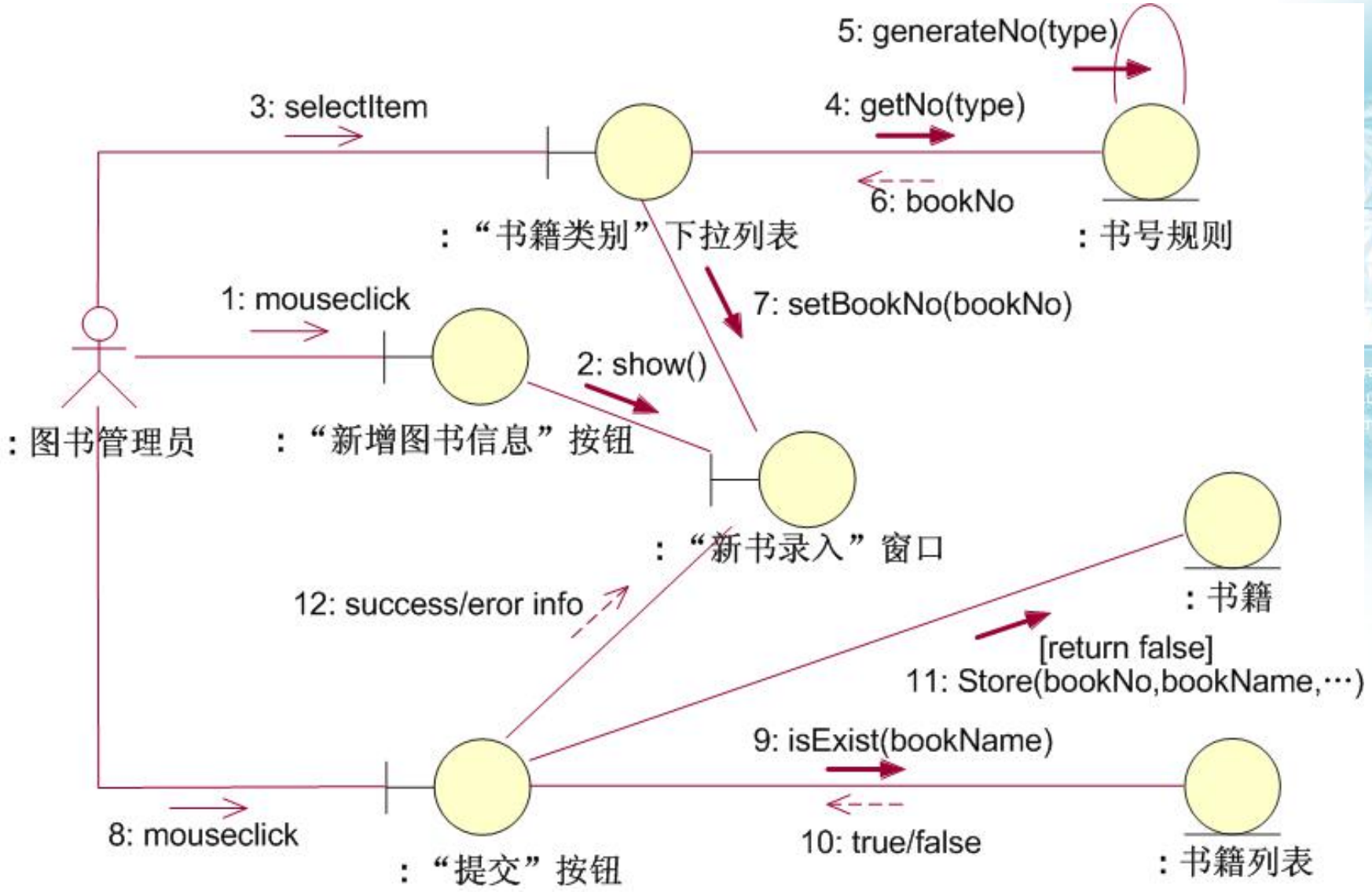
- 新添两个逻辑：一是基本事件流中的步骤2、3要求根据用户选择的类别，自动获得书号；二是当书名重复性检查没有通过（有重名），则应返回要求其重输。



构建交互模型


：图书管理员
：新增图书信息按钮
：新书录入窗口
：书籍类别下拉列表
：提交按钮
：书号规则
：书籍
：书籍列表
mouseclick
show()
selectItem
getNo(type)
generateNo(type)
bookNo
setBookNo(bookNo)
mouseclick
isExist(bookName)
true/false
opt
[return false]
Store(bookNo,bookName,…)
success/error info

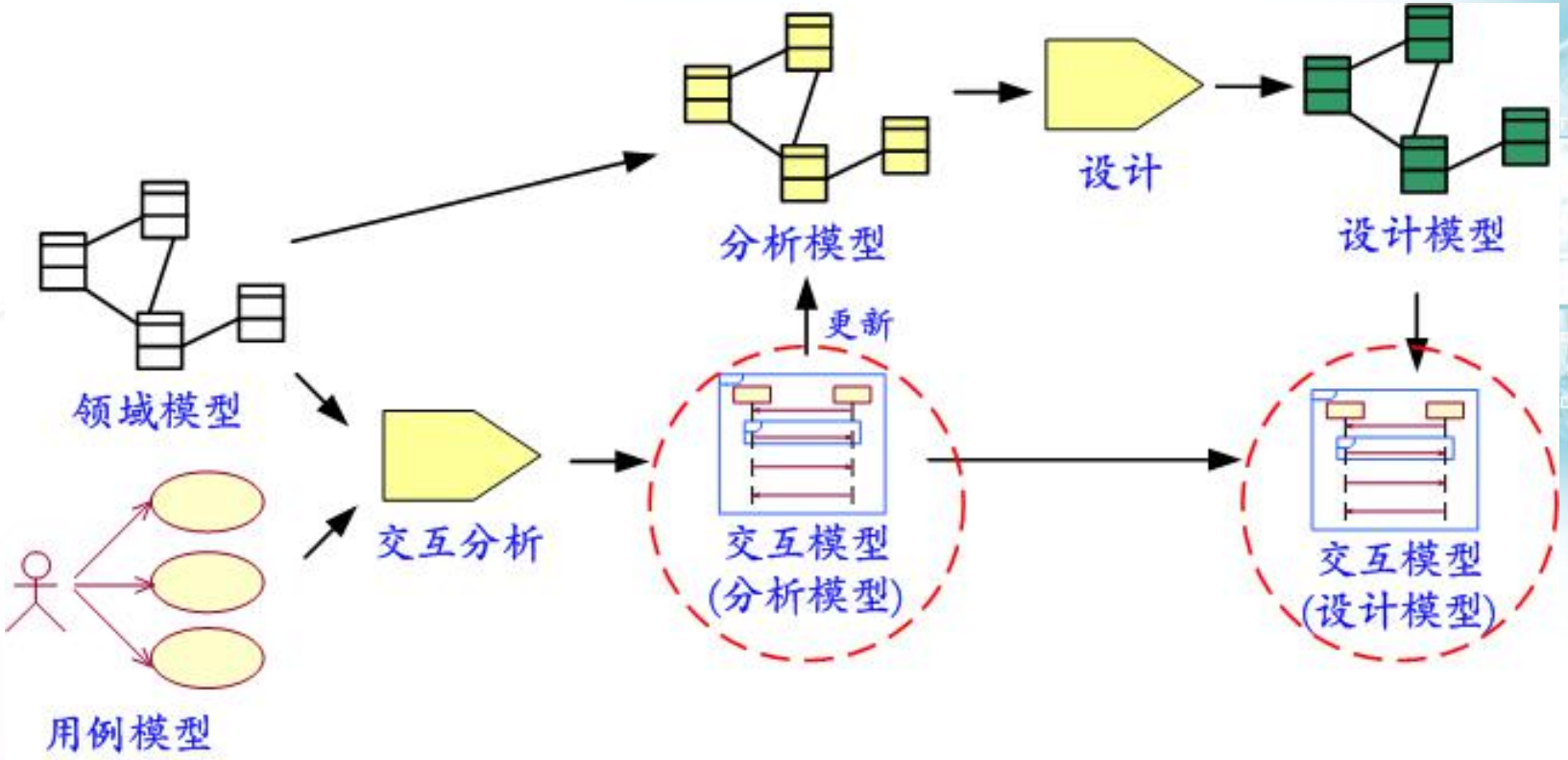
转换成通信图


5: generateNo(type)
3: selectItem
4: getNo(type)
6: bookNo
：“书籍类别”下拉列表
：书号规则
1: mouseclick
7: setBookNo(bookNo)
2: show()
：图书管理员
：“新增图书信息”按钮
：“新书录入”窗口
：书籍
12: success/eror info
[return false]
11: Store(bookNo,bookName,…)
9: isExist(bookName)
8: mouseclick
10: true/false
：“提交”按钮
：书籍列表

本章内容

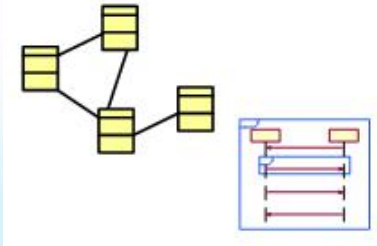
- 交互与交互图
- 如何阅读交互图
- 如何绘制交互图
- 交互图应用说明

交互模型的类型与演变



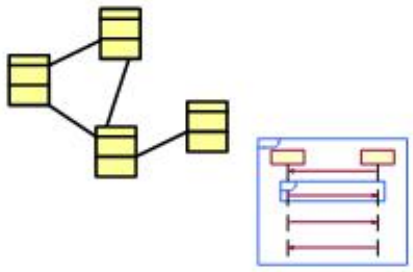
分析阶段的交互模型

- 工作方法：针对用例图中的每个用例，并结合领域模型中的类，寻找分析类，并通过Robustness分析来理清业务逻辑流程，再用交互模型将其确定下来
- 注意：主要关注于区分出边界对象、实体对象和控制对象，暂时不要考虑其具体的实现类
- 说明：对于较复杂的用例，可以按上述的流程逐渐地进行分析、设计、实施；但对于比较简单的用例而言，也是可以直接从用例描述中导出设计阶段交互模型



分析阶段的交互模型之后

- 引入基础类：包括基础框架、程序库等
- 质量评审：
  - -- 低耦合：耦合性是指两个类之间的连接强度小
  - -- 高内聚：内聚性是指一个类的属性与方法高度集成
  - -- 效率：解决方案的执行效率是否满足系统的需求
  - -- 完整性：是指在任何环境下都可以重复使用
  - -- 简单性：类越简单，出错的可能性越小，系统的灵活性和可维护性也越好
- 优化类设计：设计模式与重构



设计阶段的交互模型 & 交互建模要点

- 在分析模型的基础上引入基础类、优化类设计之后，必然会获得新的类模型（设计模型），因此就可能需要基于新引入的“设计类”来更新交互模型，以获得与实际代码相吻合的模型

- 给出一个能表达其目的的名称；通过修改元素的布局，尽量避免交叉线的存在；可以通过注解和颜色作为可视化提示，以突出图形中的重要特性；尽量少用分支，对于分支很多的场景，可以考虑用活动图来补充。

本章内容回顾

- 首先介绍了交互的概念，并延伸出UML中的两种交互图
- 以为“从订单生成送货单”场景绘制的顺序图为例，介绍了对象与角色、生命线与控制焦点、消息、顺序编号、循环与分支、交互片断操作符等基本概念
- 以等价的通信图为例，介绍了通信图的基本概念
- 演示了如何采用Robustness分析法，从一个用例的事件流描述中导出相应的交互模型
- 讨论了交互模型的实际应用