



北京航空航天大学
BEIHANG UNIVERSITY

信息系统分析与设计

用例实现 Use Case Realization

信息系统系 刘冠男



>>> 用例实现 Use Case Realizations?

- 类之间如何交互以实现系统的功能
- 核心元素

Element	Purpose
Analysis class diagrams	Show the analysis classes that interact to realize the use case
Interaction diagrams	Show collaborations and interactions between specific instances that realize the use case – they are “snapshots” of the running system
Special requirements	The process of use case realization may well uncover new requirements specific to the use case – these must be captured
Use case refinement	

What we have?

Are they enough?

How they function?

Make a refinement!

>> UML 四种类型图



- **Analysis Class Diagram**
- **Activity Diagram**
- **Interaction Diagram**
 - Collaboration Diagram
 - Sequence Diagram

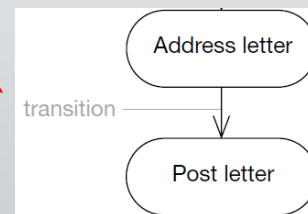
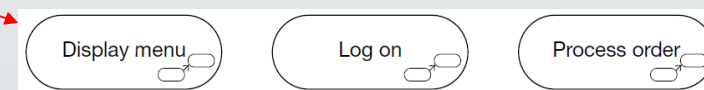
>> 动作图 Activity Diagram



- 描述了实现用例所要执行的各项活动的顺序安排，展现从一个活动到另一个活动的控制流程。
- Activity diagrams are “OO flowcharts”.

- **Elements:**

- Start, stop states
- Action states
- Subactivity states
- Transitions



Activity 活动

- 活动是构成活动图的核心元素，表示在用例工作流程中执行的某个动作或步骤。
- 在活动图中，一个活动结束后立即进入下一个活动。



初态



终态

活动名

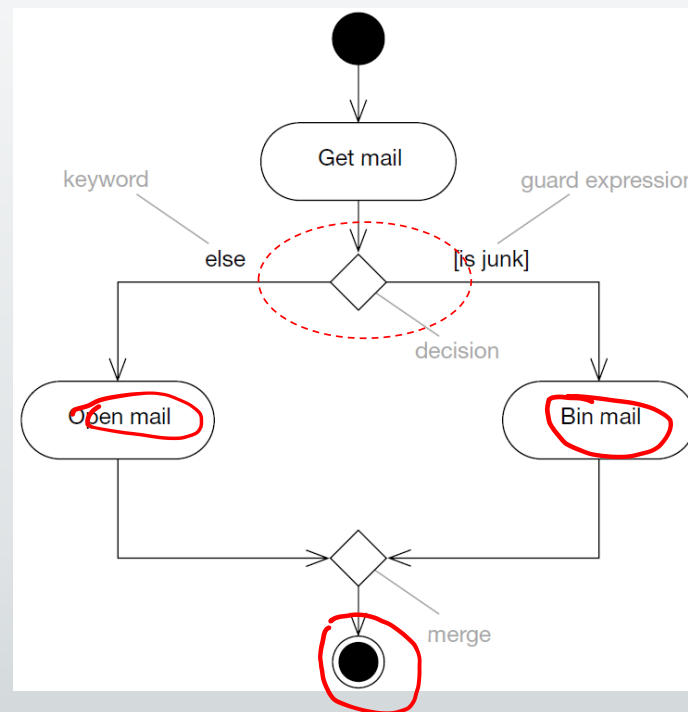
活动

转移

- 转移表示活动之间的跳转行为，它由活动的完成来触发。



>>> Activity Diagram: Decisions

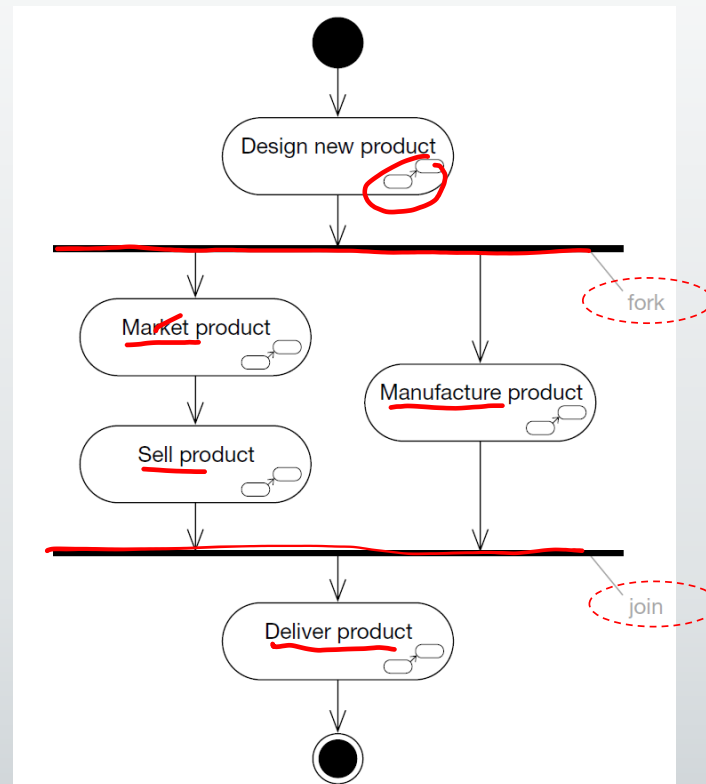


Decisions allow you to model decision points.

>>> Activity Diagram: Forks and Joins

Forks split a path into two or more concurrent flows.

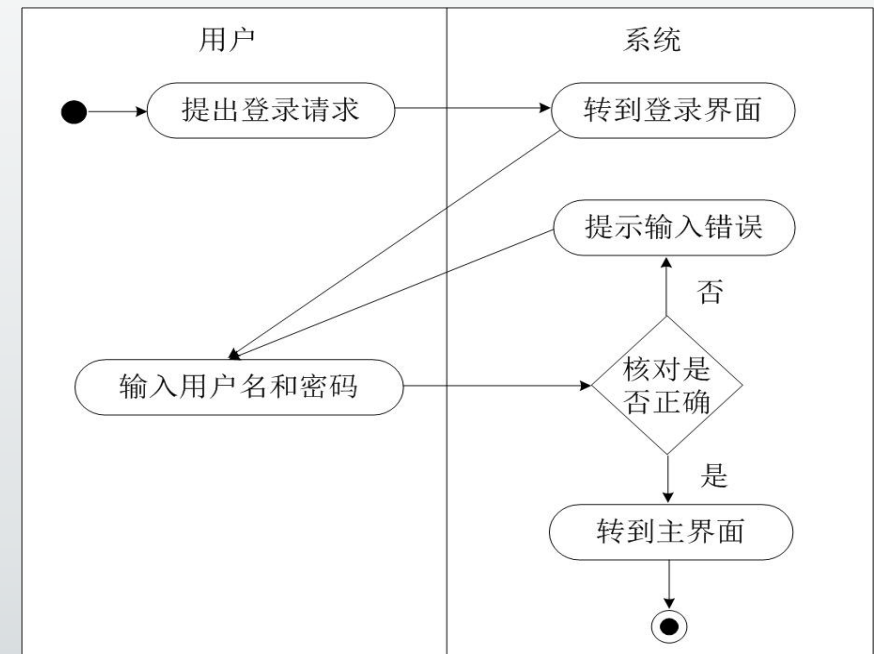
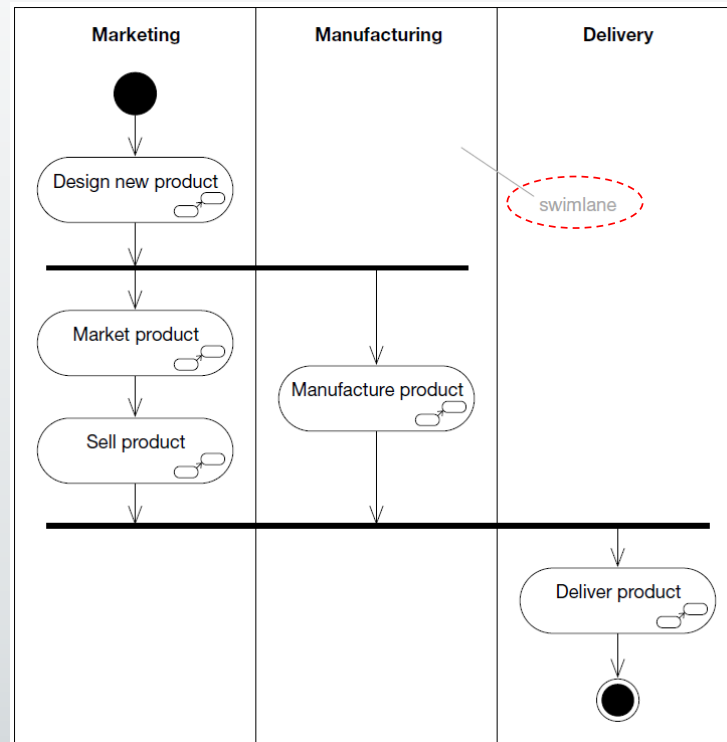
Joins synchronize two or more concurrent flows.



>>> Activity Diagram: Swimlanes



Swimlanes allow us to partition activity diagrams.



>> 练习：取款用例的活动图



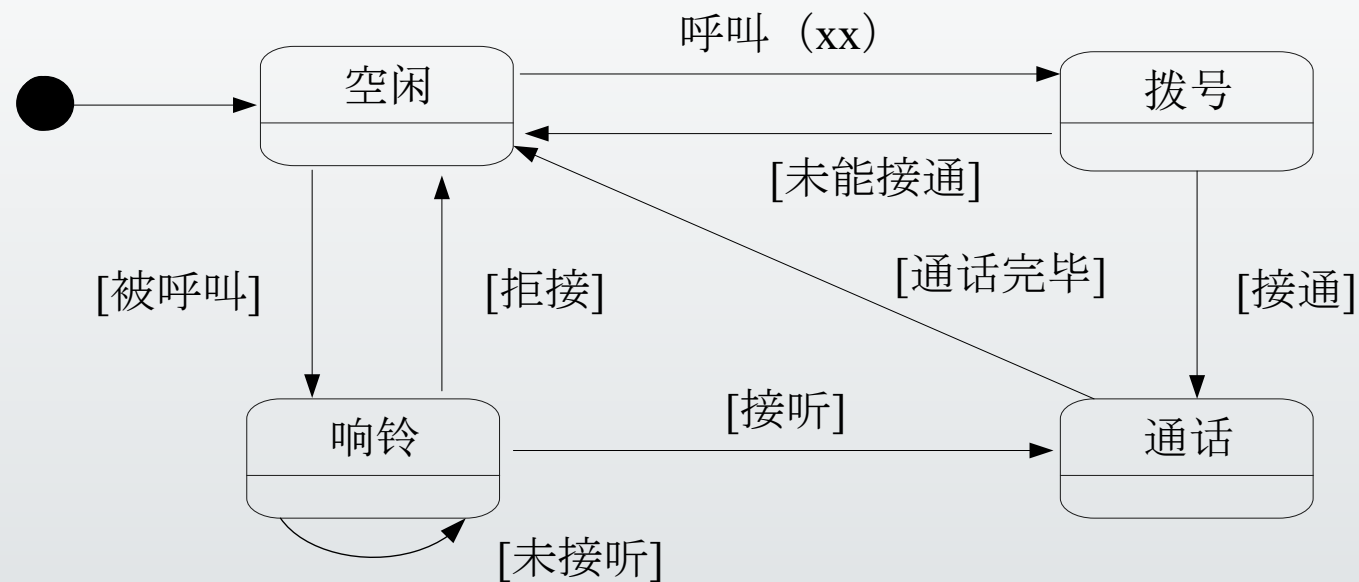
>> UML状态图



- 状态图 (Statechart Diagram) 用于描述一个特定对象在其生存期间基于事件反应的动态行为，显示该对象是如何根据当前所处状态对不同事件做出反应的。
- 通常只有对于一些具有复杂行为或处于不同状态对应不同处理的对象，才有必要用状态图描述它的状态转移过程。
- 对于画了状态图的对象，其相应实体类所对应的表中要加上状态这一属性，其属性值为状态图中的各状态。



>> 状态图示例

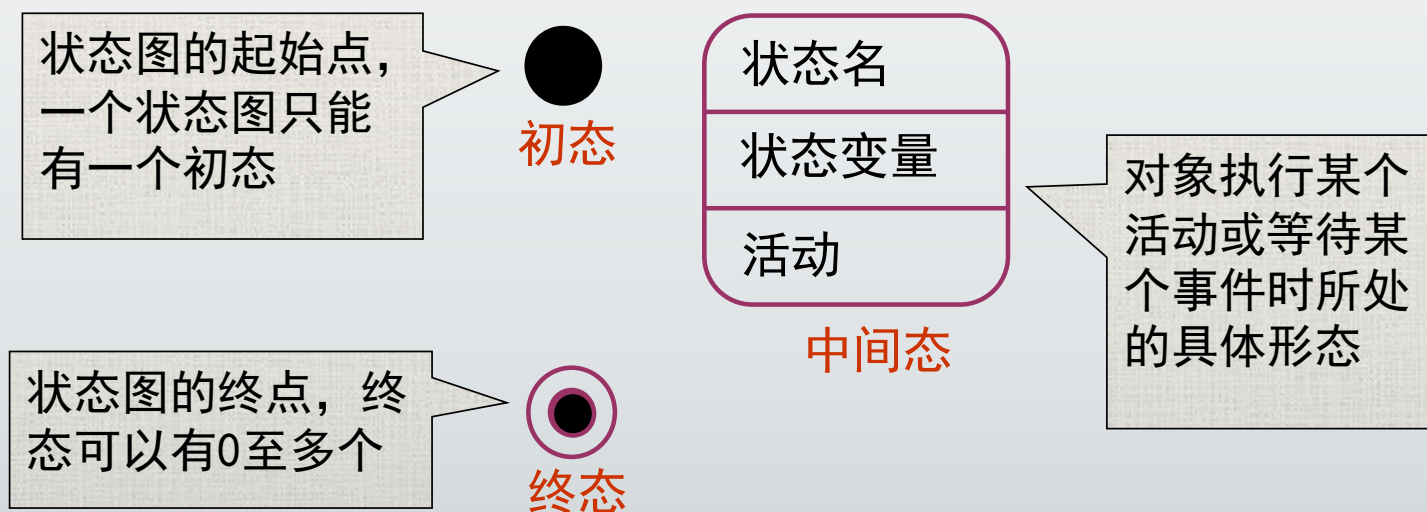


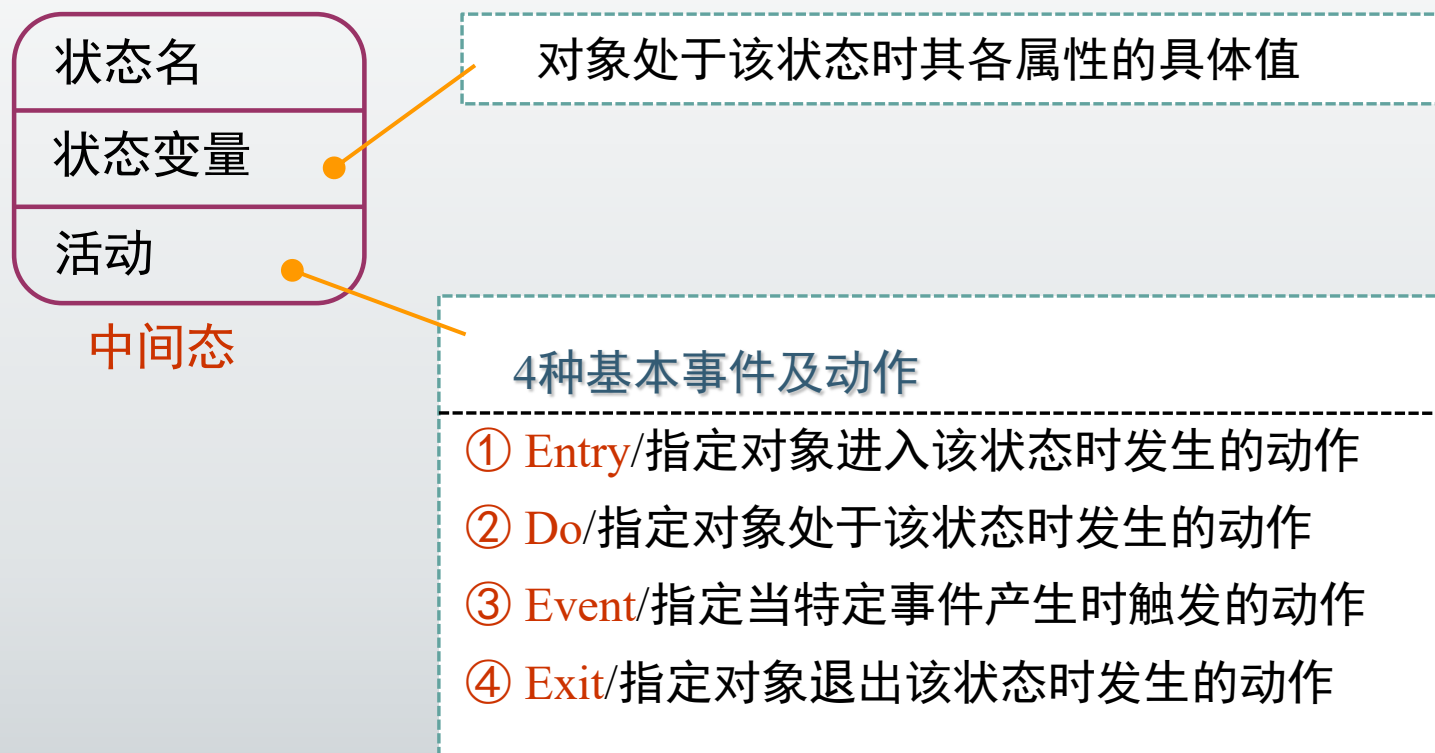
- 在UML中，状态图由状态、状态间的转移、引起状态转移的事件组成。

>>> 状态



- 状态是对象执行了一系列活动的结果，所有对象都具有状态，当某个事件发生后，对象的状态将发生变化。
- 在状态图中，对象的状态主要分为三种：





Lighting

Entry/turn on

Do/compute degree

Event Power off/power supply

Exit/turn off

电灯对象之点亮状态

通话

Entry/通话连接

Do/计时并计费

Event 断电/挂断并提示

Exit/关闭通话

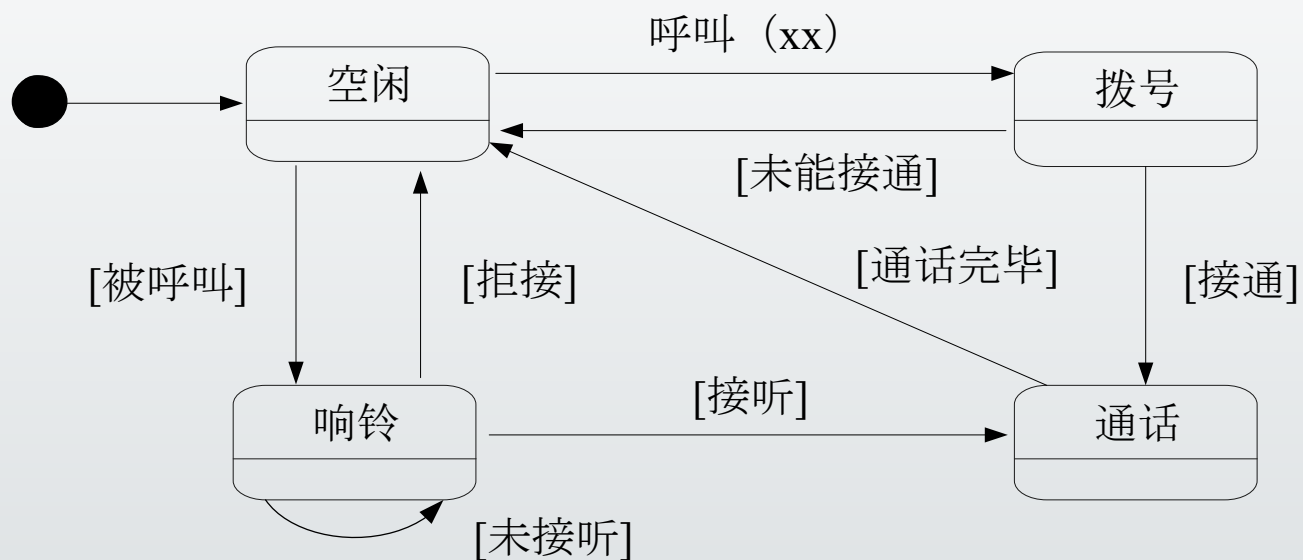
手机对象之通话状态

>>> 状态转移

- 一个对象状态的变迁称为状态的转移
- 状态转移的条件：
 - 事件触发状态转换
 - 该状态的内部活动执行完毕自动触发转移
- 给定的状态只能产生一个转移



>>> 状态转移



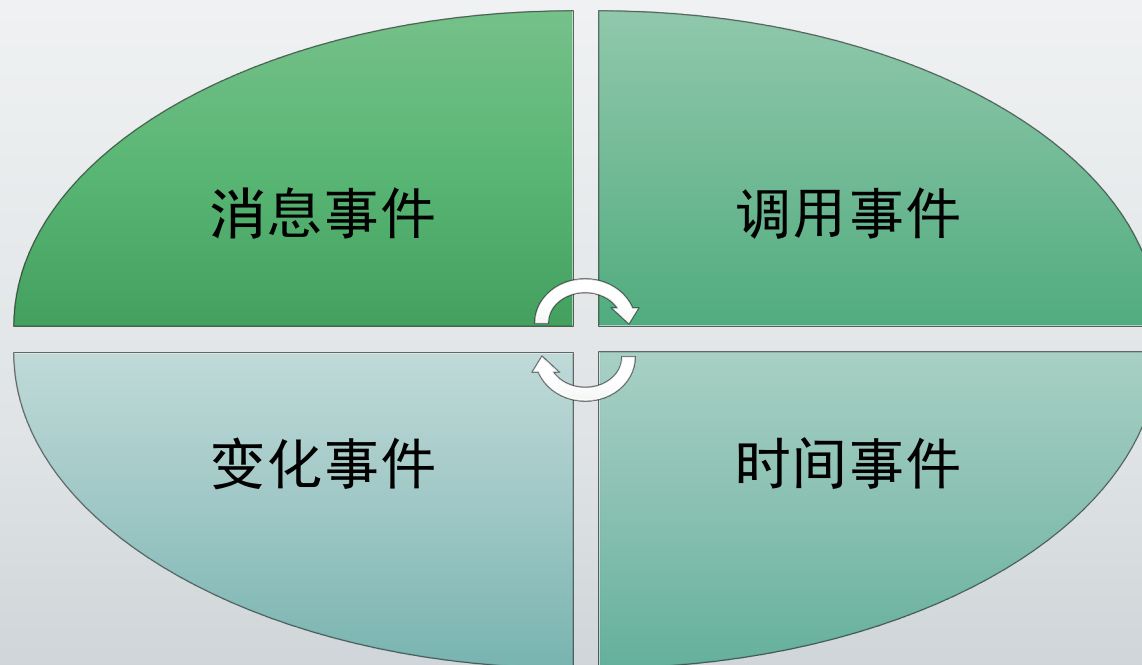
P. “手机对象” 状态转移图



>> 事件



- 事件是触发状态转移的条件或操作。



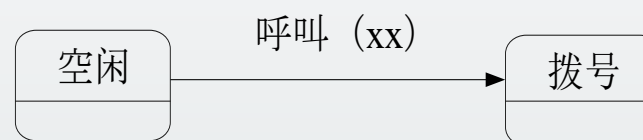
事件

- 消息事件：由外界传递的简单信号或消息，对象收到后发生状态转移。消息事件的格式为：[消息或信号]。
- 调用事件：外界传递的要求对象调用执行某个操作并发生状态转移的请求。调用事件的格式为：事件名（参数列表）。
- 时间事件：根据某时间表达式的满足情况决定对象状态转移的事件。时间事件的格式为：[时间表达式]。
- 变化事件：根据某特定条件的满足情况决定对象状态转移的事件。变化事件的格式为：[when（条件表达式）]。

>> 事件示例



(a) 消息事件—电灯对象



(b) 调用事件—手机对象

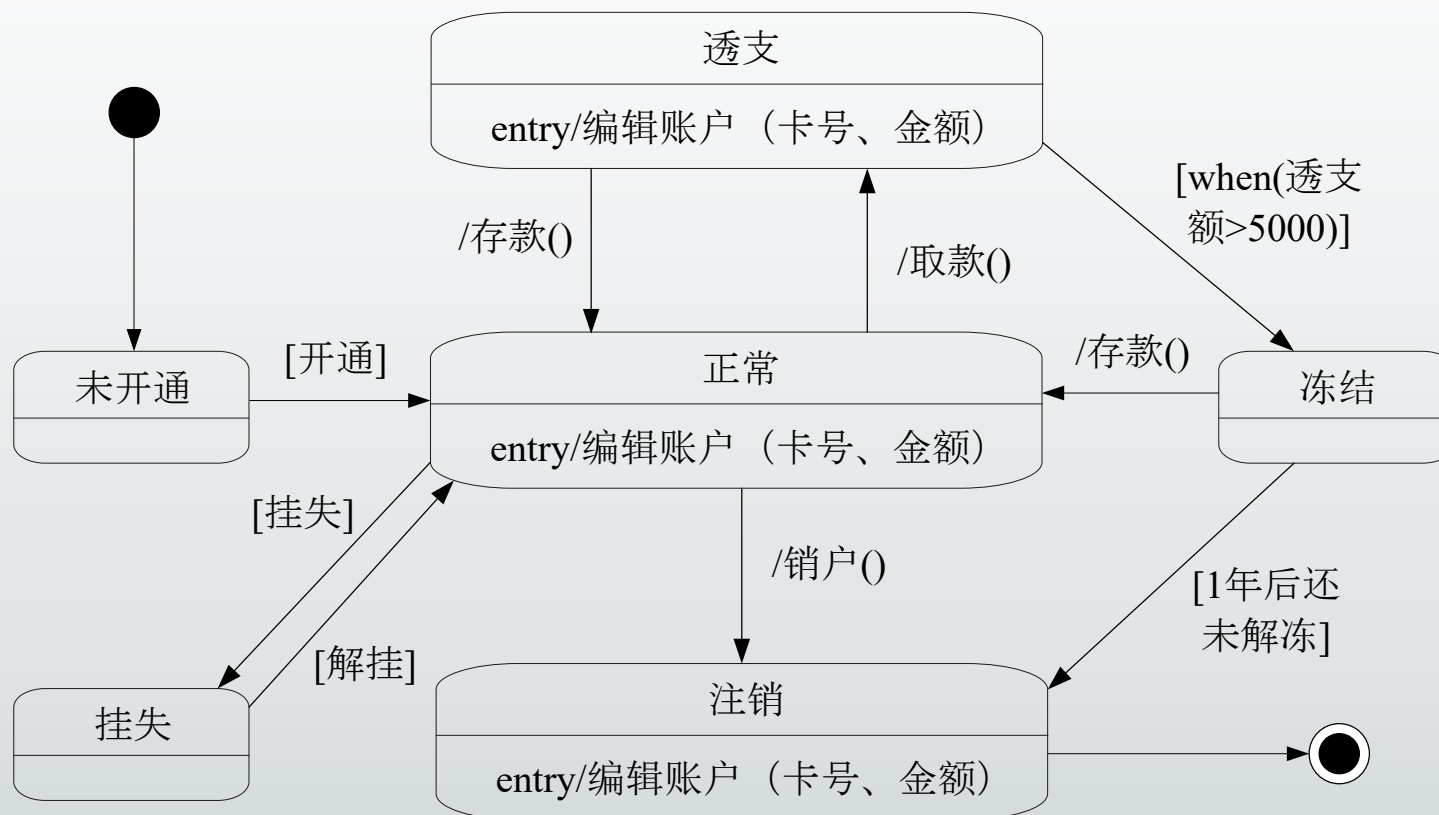


(c) 时间事件—电脑对象



(d) 变化事件—保险丝对象

>>> 状态图示例



P. “信用卡账户” 对象状态图

>> 交互图 Interaction Diagrams



- **Purpose**

- Model collaborations and interactions between objects that **realize** a use case, or part of a use case.

- **Collaboration diagram:**

- Emphasize the **structural relationships** between objects and are very useful for creating a quick sketch of an object collaboration.

- **Sequence diagram:**

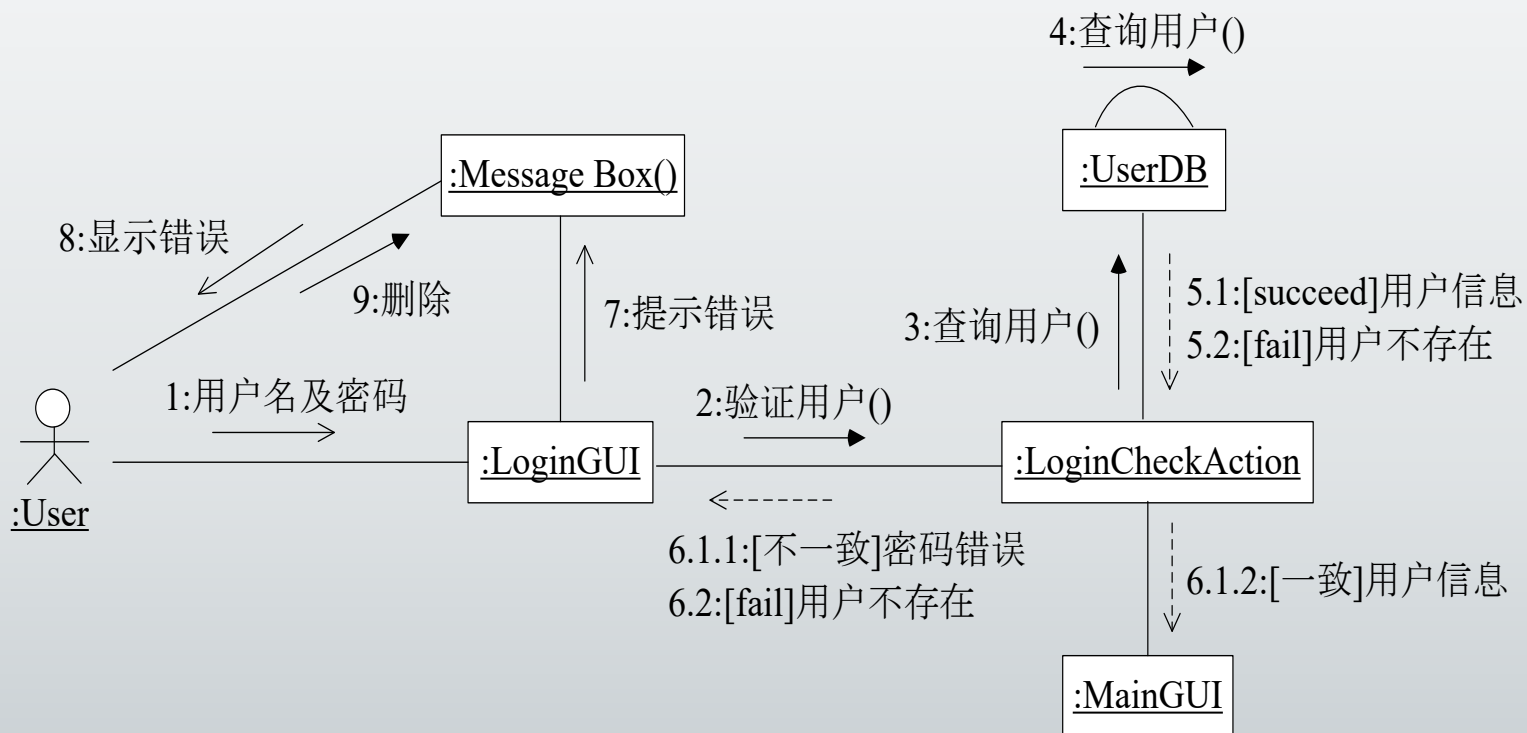
- Emphasize the **time-ordered sequence** of message sends between objects.

Collaboration and sequence diagrams provide different views of the same underlying object interaction.

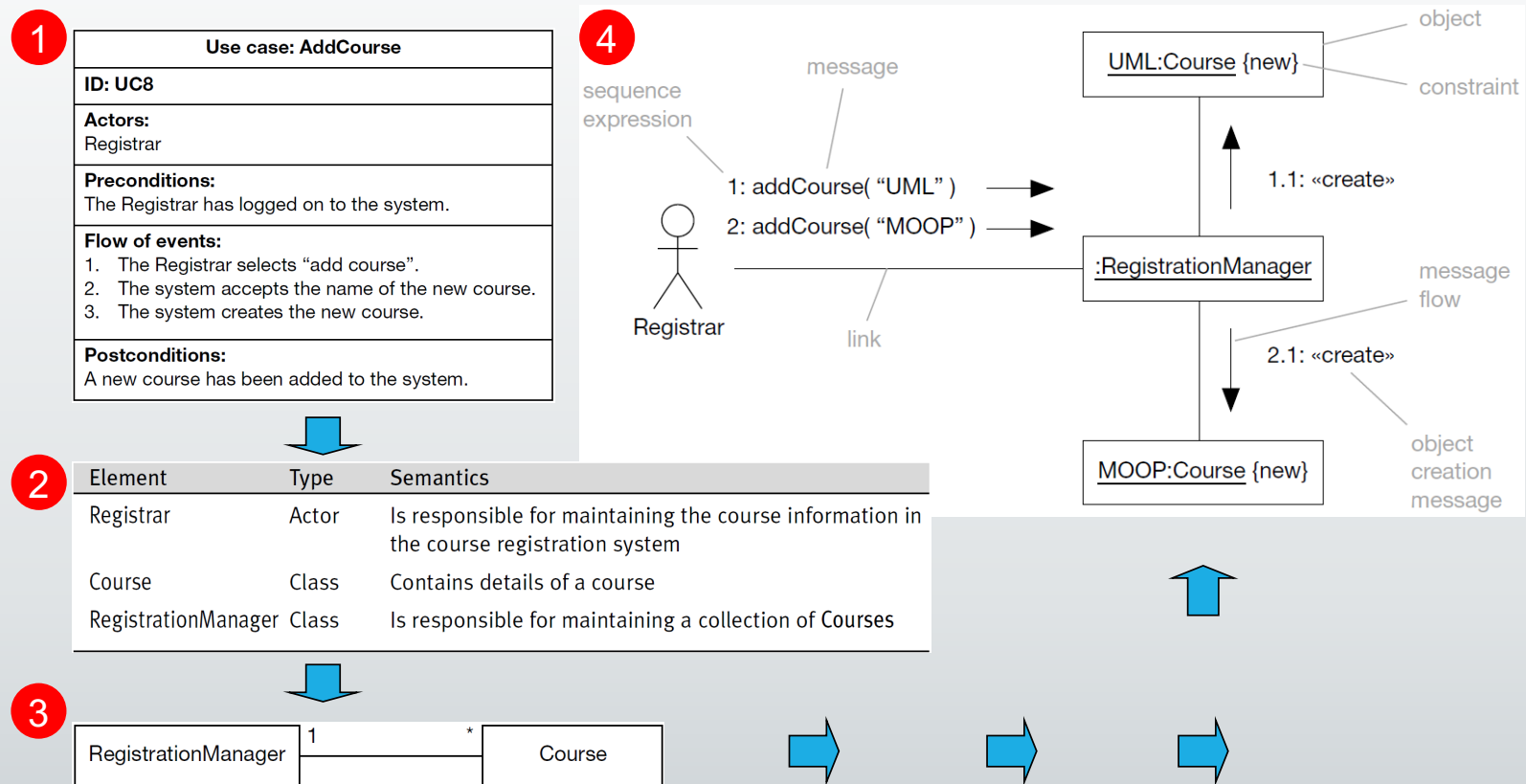
>> 协作图 Collaboration Diagram



- 协作图 (Communication Diagram) 描述了用例相关的多个对象及其之间的动态合作关系，与顺序图一样，通常也用于解释用例的实现过程；
- 协作图强调对象间的合作关系。



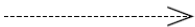


>> 协作图 Collaboration Diagram



>>> Collaboration Diagram: Elements

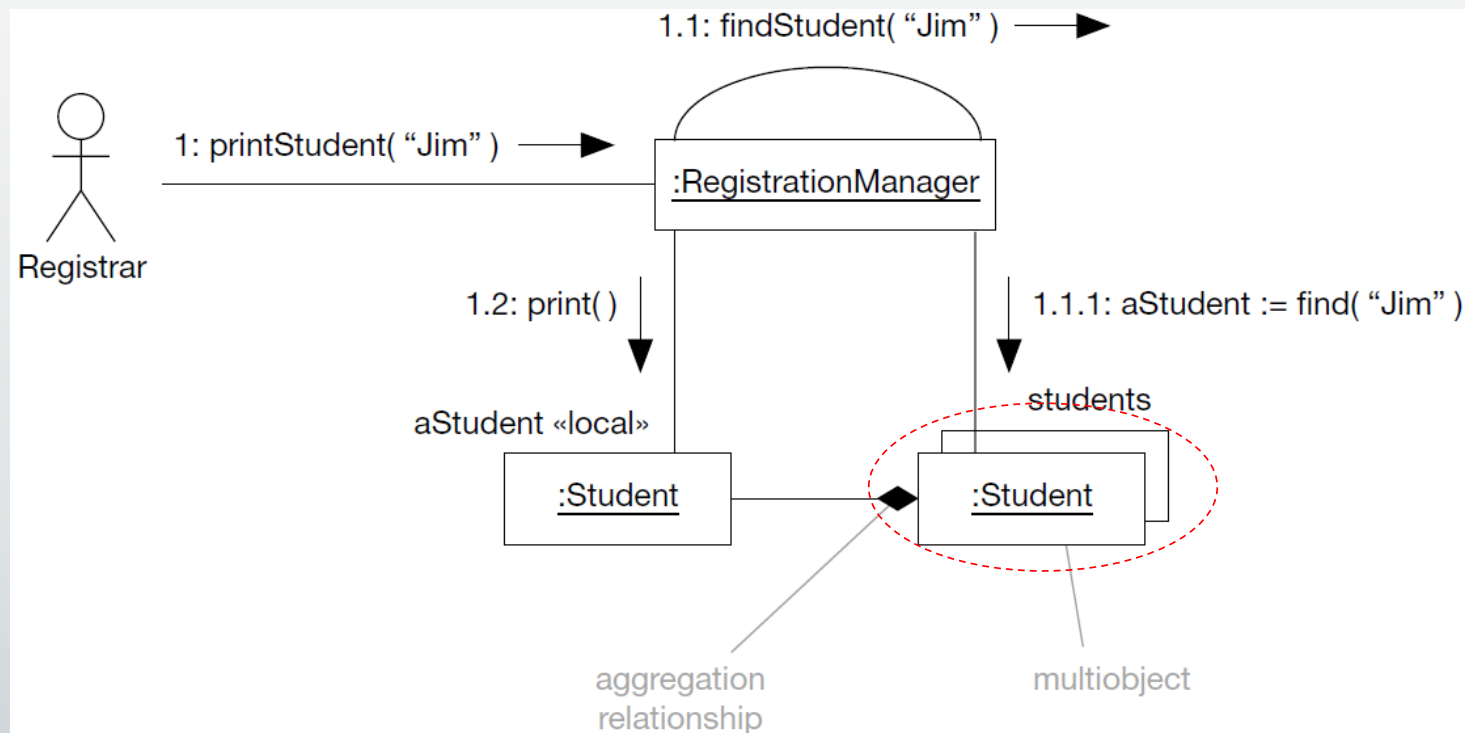


Message flow	Semantics
	Procedure call – the sender waits until the receiver has finished This is the most common option
	Asynchronous communication – the sender carries on as soon as the message has been sent; it does not wait for the receiver This is often used when there is concurrency
	Return from a procedure call – the return is always implicit in a procedure call, but it may be explicitly shown using this arrow

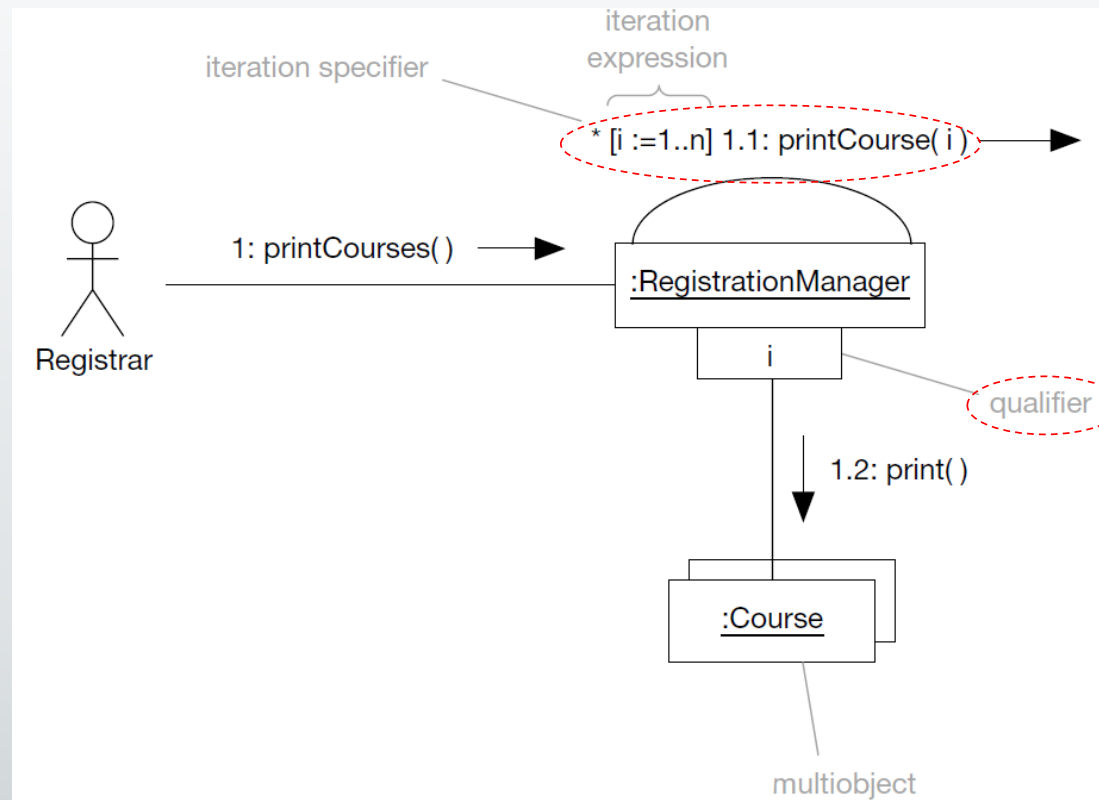
Constraint	Semantics
{new}	The instance or link is created in the interaction
{destroyed}	The instance or link is destroyed in the interaction
{transient}	The instance or link is created and then destroyed in the interaction – this is equivalent to using {new} and {destroyed} together, but it is clearer

>> Collaboration Diagram: Multiobjects

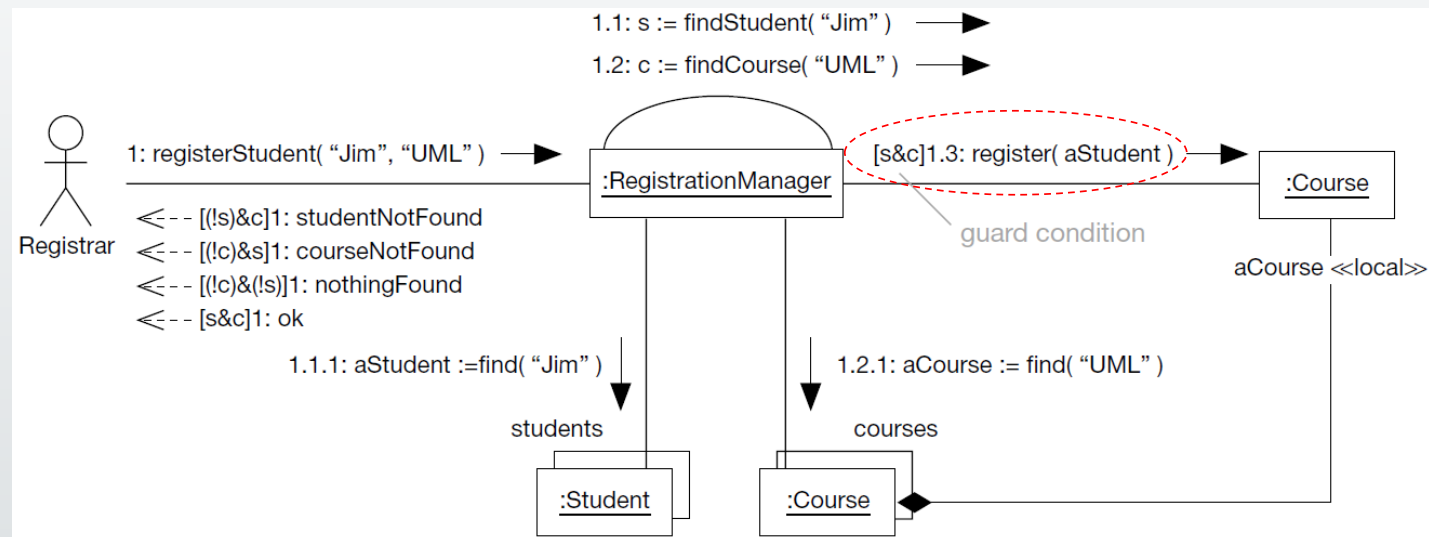
- Multiobjects represent sets of objects



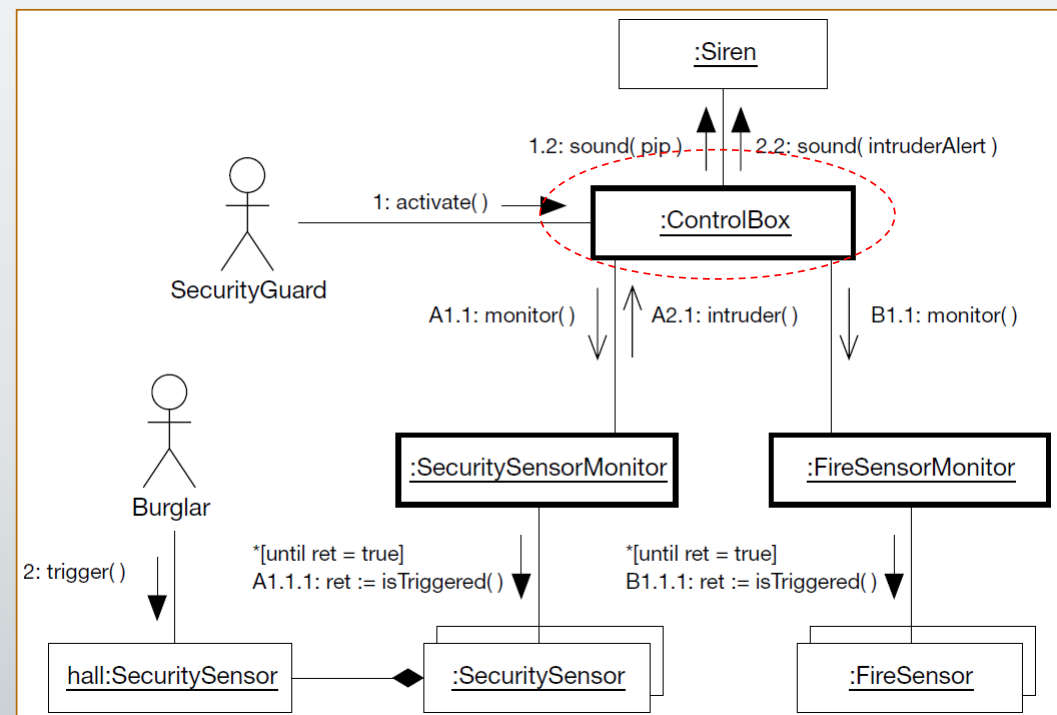
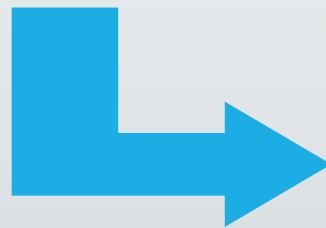
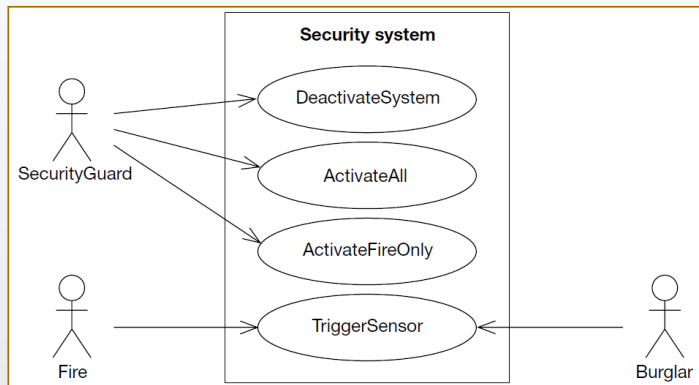
>>> Collaboration Diagram: Iteration



>>> Collaboration Diagram: Branching



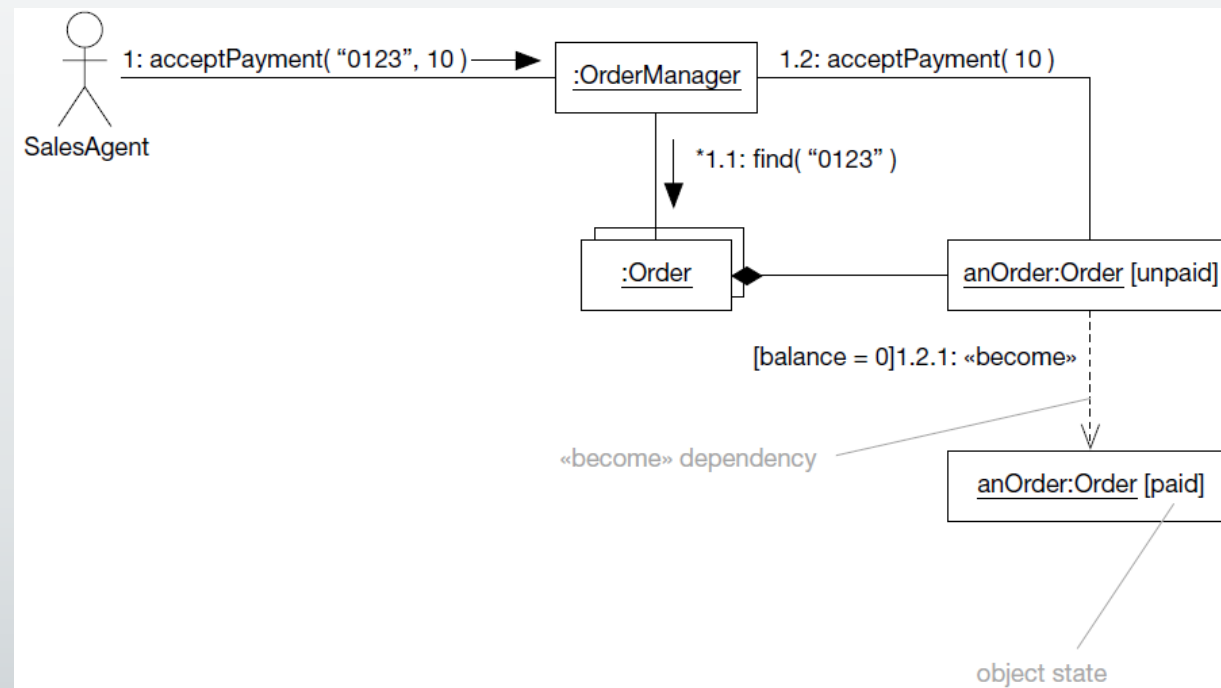
>>> Collaboration Diagram: Concurrency



>> Collaboration Diagram: Object State



- Show how object state changes as the object Interaction progresses.



>> 顺序图



- 顺序图 (Sequence Diagram) 描述了用例相关的多个对象及其之间的**动态交互关系**，通常用于解释系统**用例的实现过程**。
- 顺序图强调对象间交互的**时间**和**顺序**。

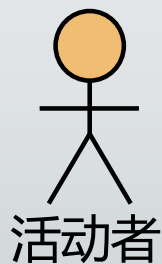


活动者

- 活动者是指用例的执行者，使用执行者的人形符号来表示。

对象

- 在顺序图中，对象用一个矩形框表示，它们代表用例中参与交互的对象。
- 对象使用标准的UML格式 “name: ClassName” 来标记。

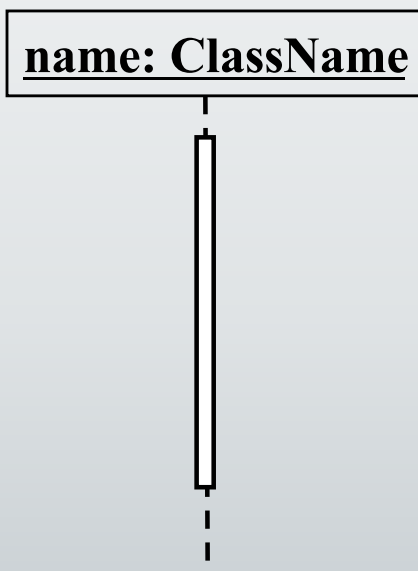


name: ClassName

百货大楼: 建筑

生命线

- 生命线表示对象存在的时间；
- 在顺序图中生命线用从对象图标向下延伸的一条虚线表示。

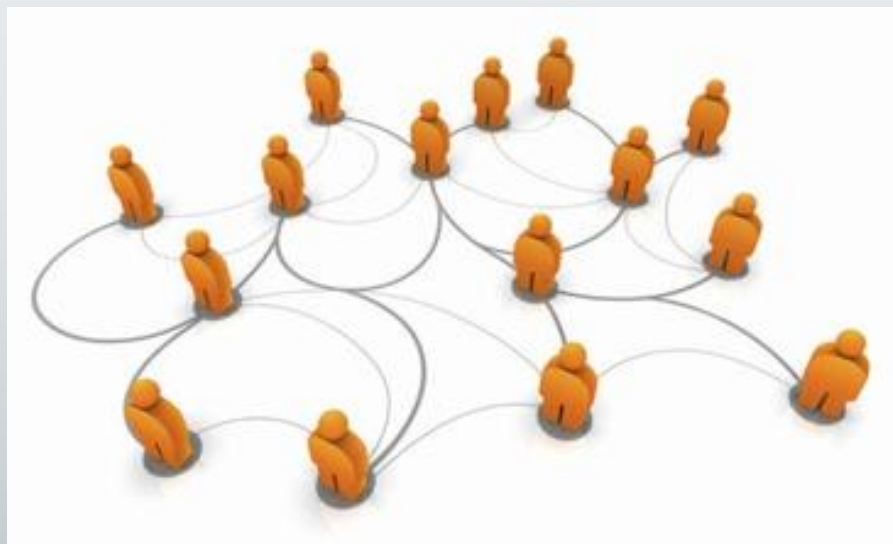


激活

- 激活表示对象执行相应操作的时间段，它是用替换生命线的双道线表示；
- 处于激活期的对象能够响应或发送消息，执行动作或活动；
- 不在激活期的对象处于休眠状态，需要等待新的消息来激活它。

消息

- 在面向对象方法中，对象间的交互是通过传递消息来完成的。消息是对象间的一种通信方式，UML中定义的消息包括以下4种：
 - 简单消息
 - 调用消息
 - 返回消息
 - 异步消息



消息—简单消息

- 表示简单的控制流，用于描述控制流如何在对象间进行传递，而不考虑通信的细节。

简单消息



用户名、密码



消息—调用消息

- 传递了要求接收对象执行某种操作或调用某个方法的请求。发送对象发出消息后必须等待消息返回，只有处理消息的操作执行完毕后，发送对象才可以继续执行下一步操作；
- 调用消息的格式为：操作（参数列表）。

调用消息



验证（用户名、密码）



消息—返回消息

- 返回消息是调用消息中的操作完成后，返回给调用消息发送对象的回应消息；
- 调用消息一般都对应一个返回消息。

返回消息



登录成功



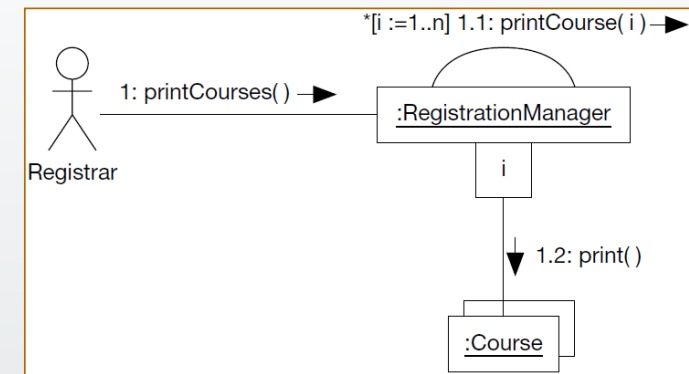
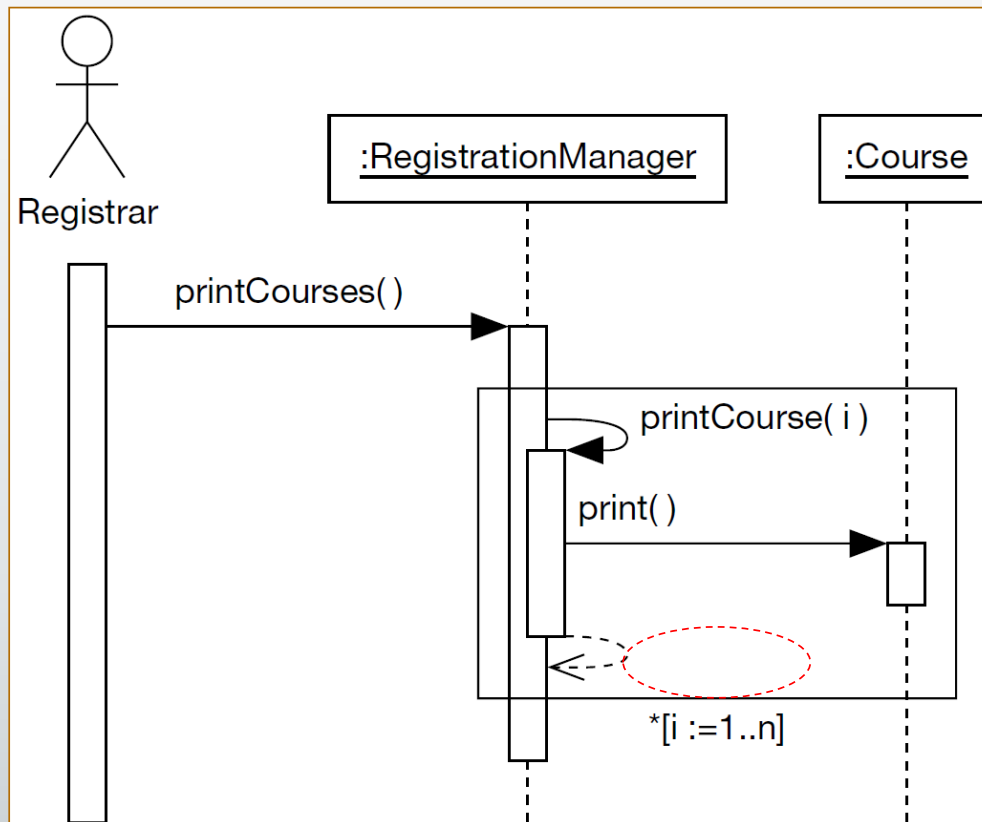
消息—异步消息

- 是一种不需等待返回消息的特殊调用消息。异步消息主要用于描述实时系统中的**并发行为**，发送对象发出消息后可立刻进行下一步操作。

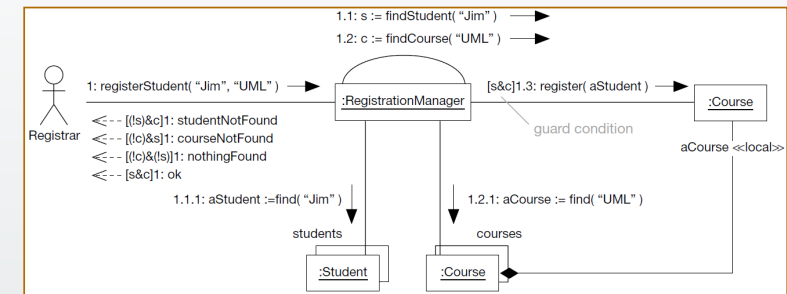
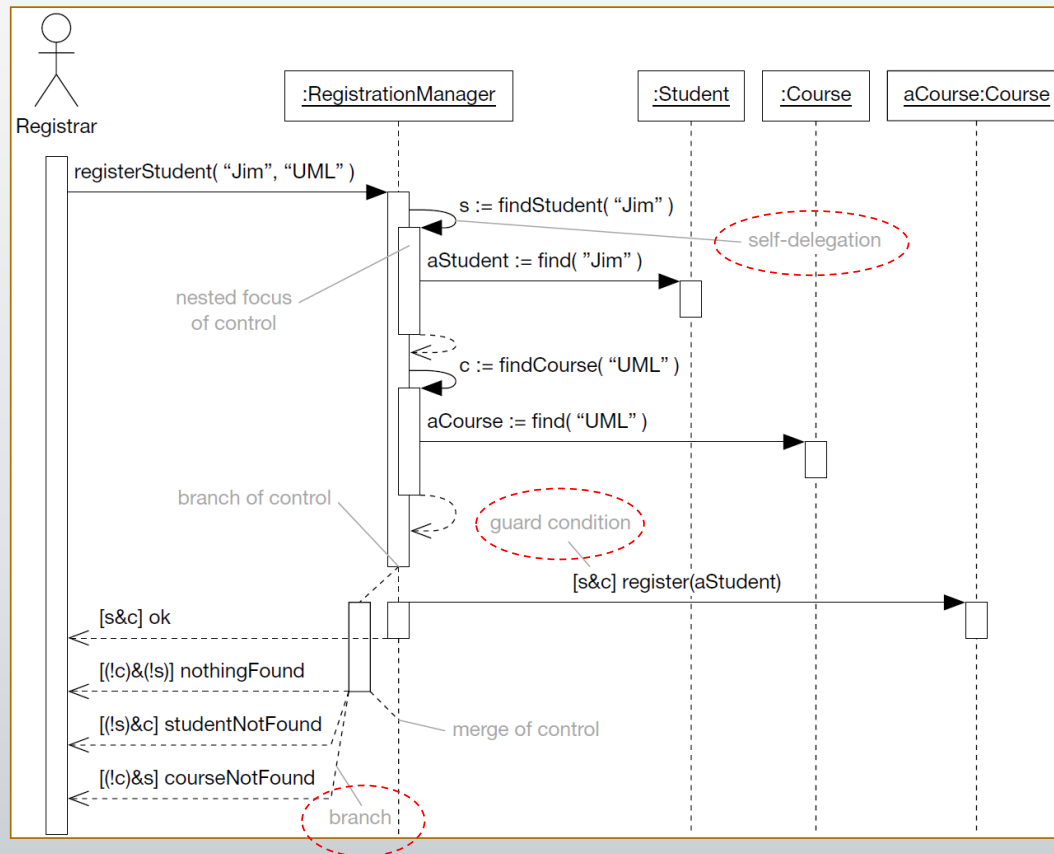
异步消息



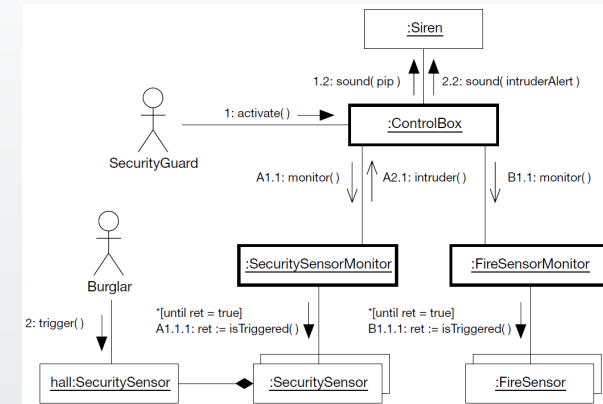
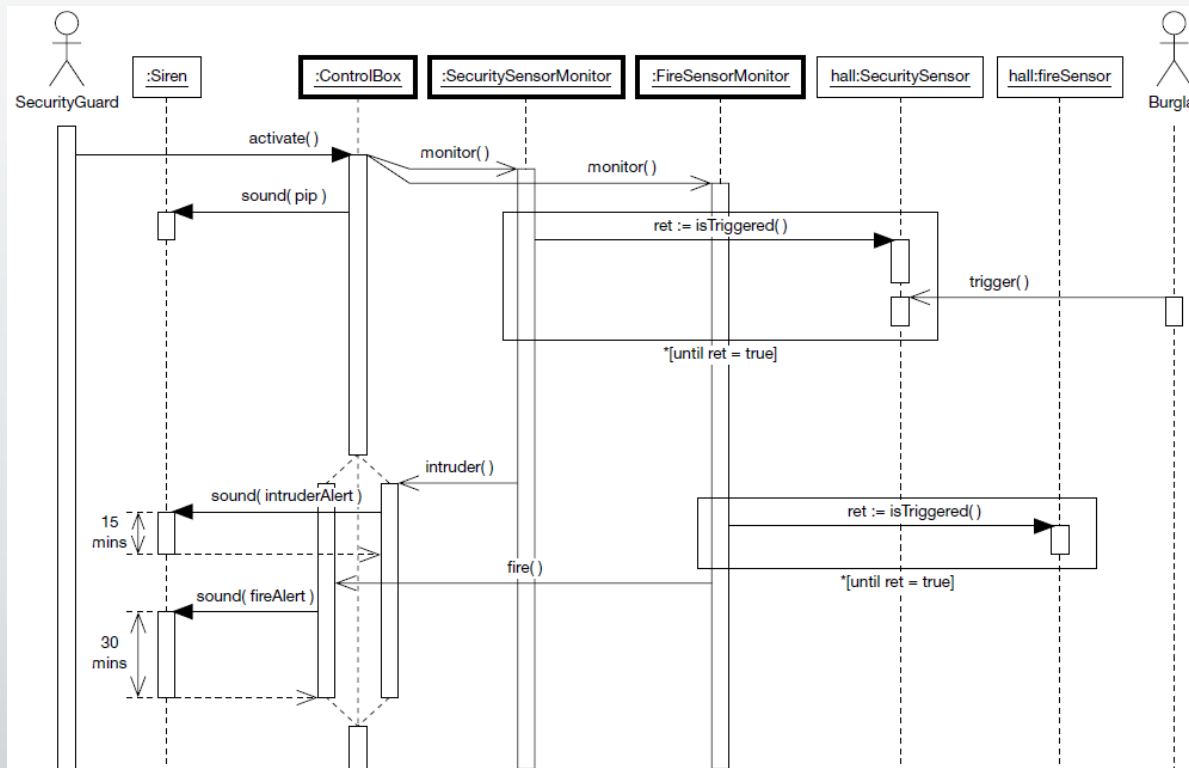
>> Sequence Diagram: Iteration



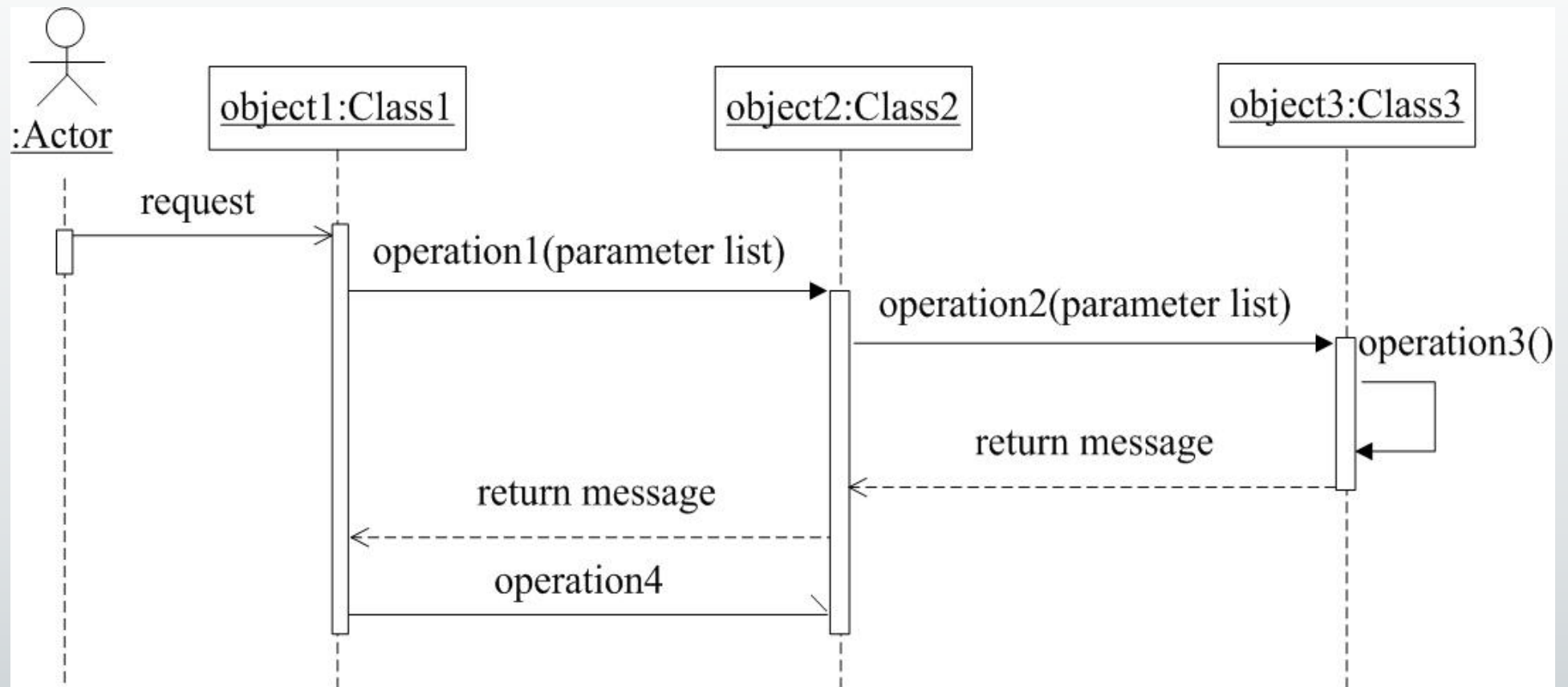
>>> Sequence Diagram: Branching



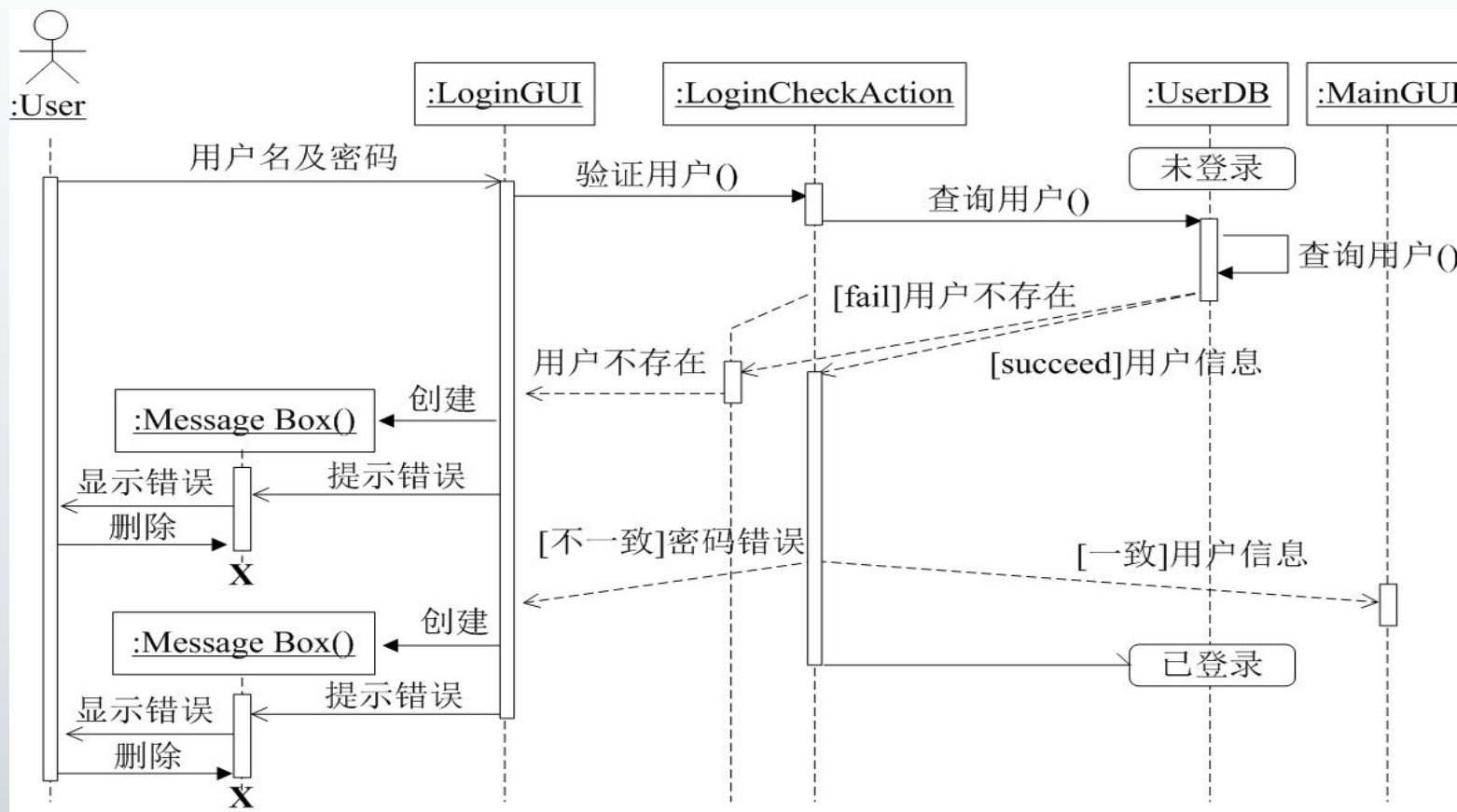
>>> Sequence Diagram: Concurrency



>> 序列图画法



>> 序列图示例



P. “用户登录”用例顺序图

序列图

强调消息发送的时间和顺序

对象具有生命线和激活

对象之间无连接线

通过时间轴表示消息传递顺序

协作图

强调对象间的合作关系

对象无生命线和激活

对象之间存在连接线

通过序列号表示消息传递顺序

>>> In-class practice



- 考虑之前Catherine's Catering这一案例，结合拟设计的订单管理系统，进行分析建模
 - 绘制这一订单系统的用例图
 - 绘制“用户订餐—就餐—买单”全过程的活动图
 - 详细描述“订单处理”这一用例，并绘制顺序图

>> 系统分析小结

- 需求定义与需求建模
- 数据流图与数据字典
- 面向对象分析：类、分析类、包图
- 用例建模
 - 用例图
- 用例实现
 - 活动图
 - 状态图
 - 顺序图/协作图
- How is your group project?



刘冠男

liugn@buaa.edu.cn

