Novel Introduction of UML Models

Yang YI, Ph.D

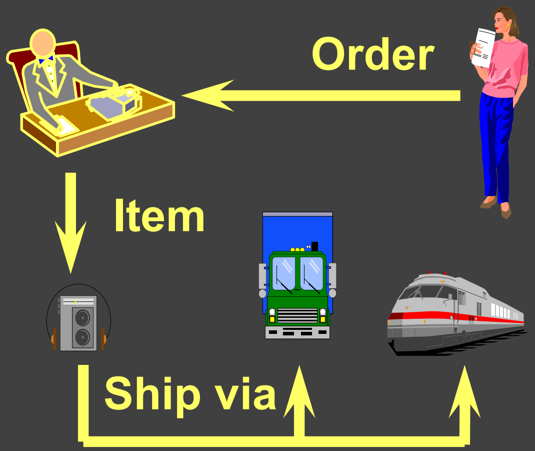
issyy@mail.sysu.edu.cn

chart：13902295111

Where Are We?

- What is visual modeling?
- What is the UML?
- UML diagrams
- Extending UML notation

What Is Visual Modeling?



Order

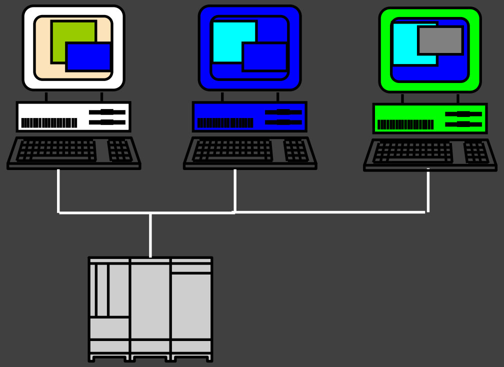
Item

Ship via

Business Process

“Modeling captures essential parts of the system.”
Dr. James Rumbaugh

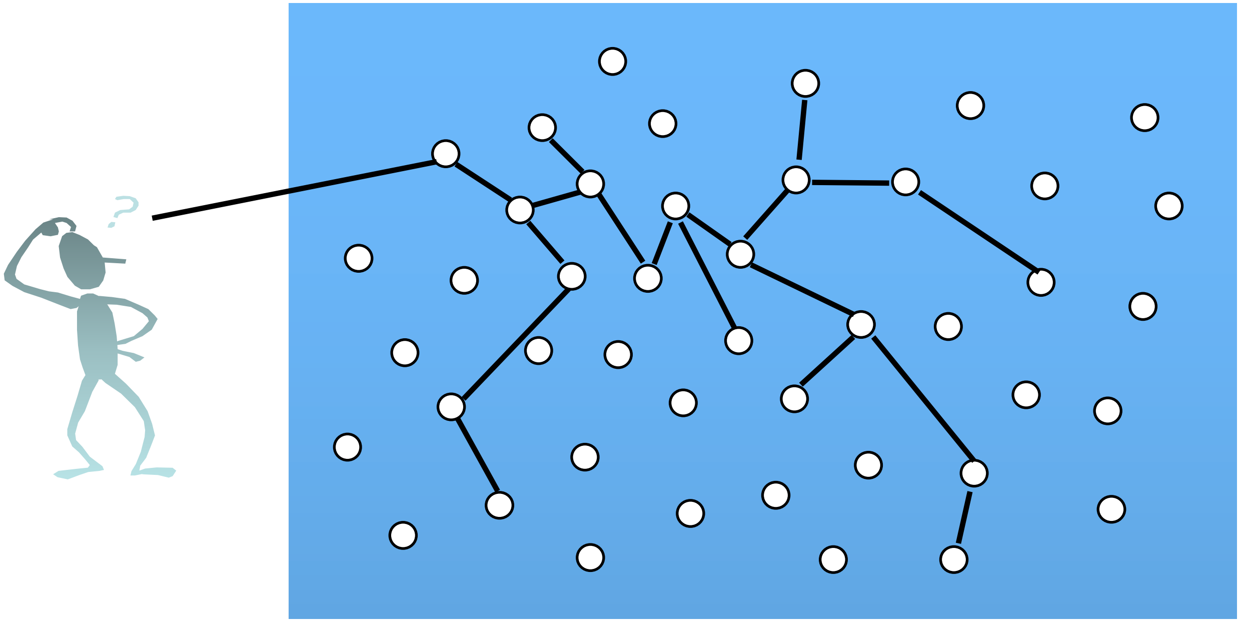
Visual Modeling is modeling using standard graphical notations



Computer System

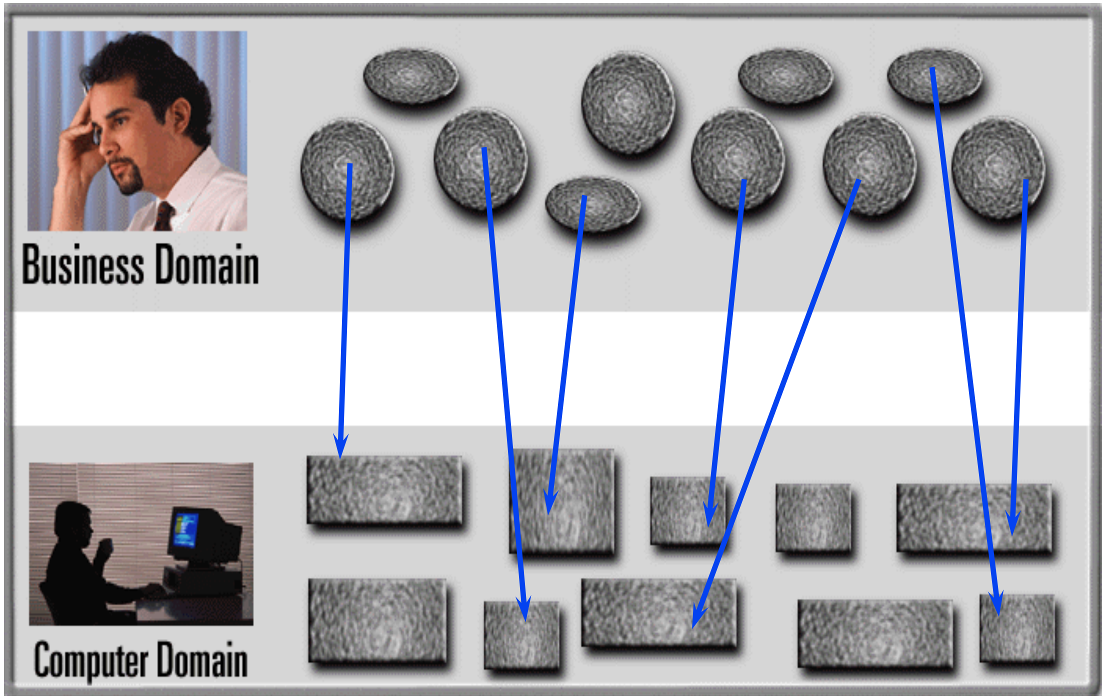
Visual Modeling Captures Business Processes

Use-case analysis is a technique to capture business processes from a user’s perspective.



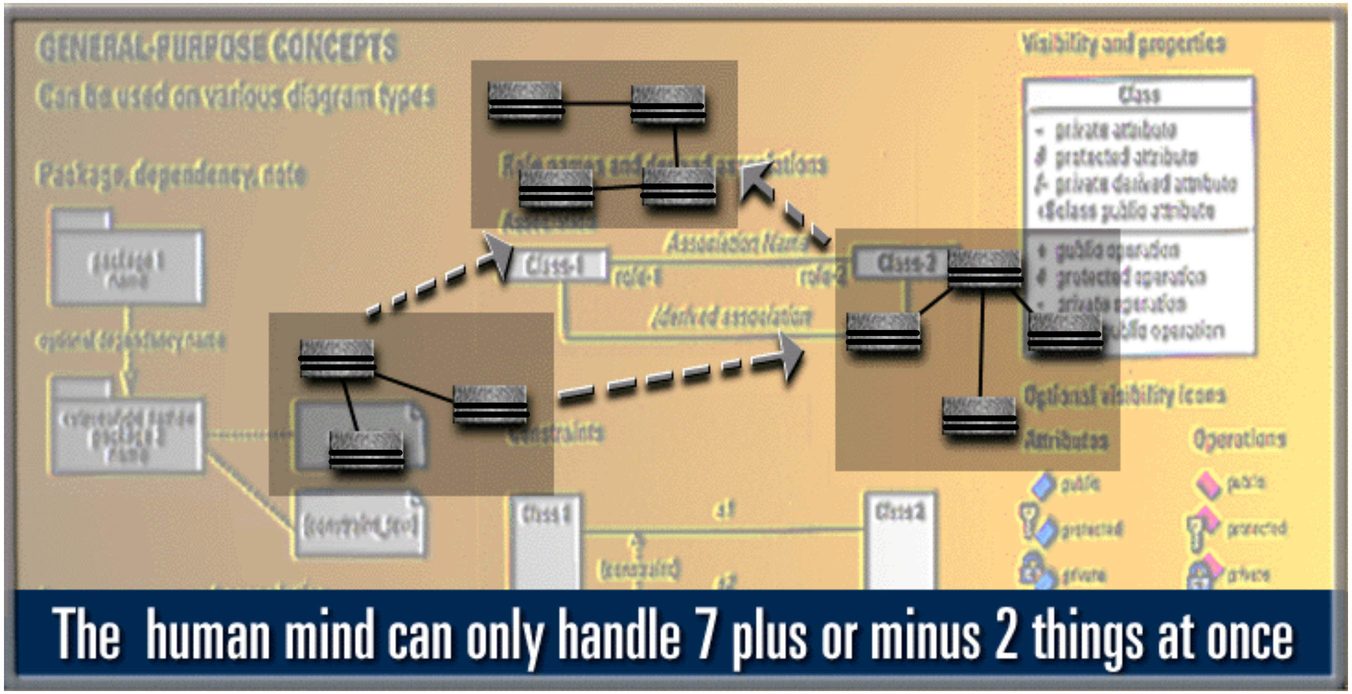
Visual Modeling Is a Communication Tool

Use visual modeling to capture business objects and logic.



Use visual modeling to analyze and design your application.

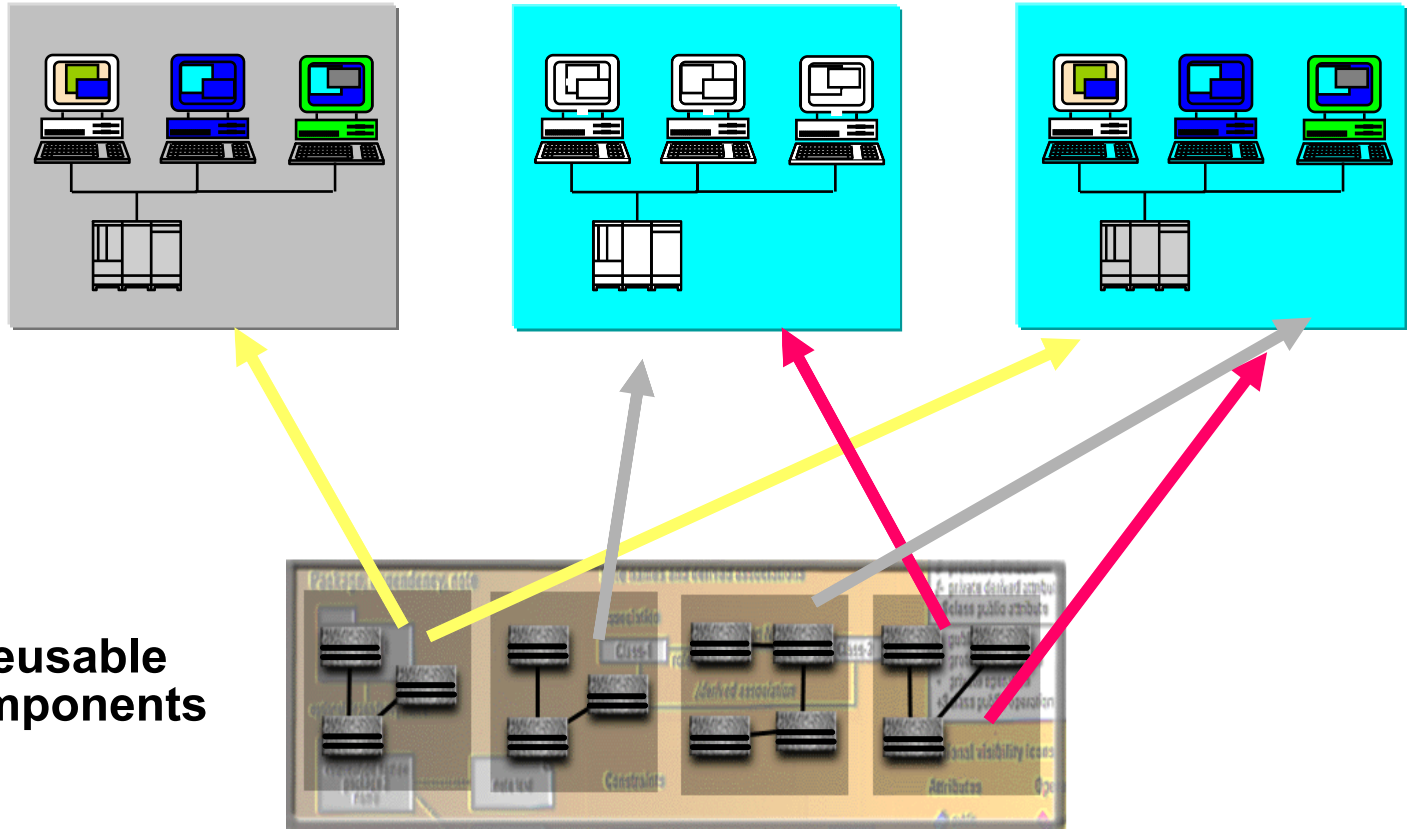
Visual Modeling Manages Complexity



The human mind can only handle 7 plus or minus 2 things at once

Visual Modeling Promotes Reuse

Multiple Systems



Reusable Components

Where Are We?

- What is visual modeling?
- What is the UML?
- UML diagrams
- Extending UML notation

What Is the Unified Modeling Language?

- The UML is the standard language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system.
- The UML combines the best from
  - Data modeling
  - Business modeling
  - Object modeling
  - Component modeling

History of the UML

Planned major revision (2001)

Planned minor revision (2000)

Public
Feedback

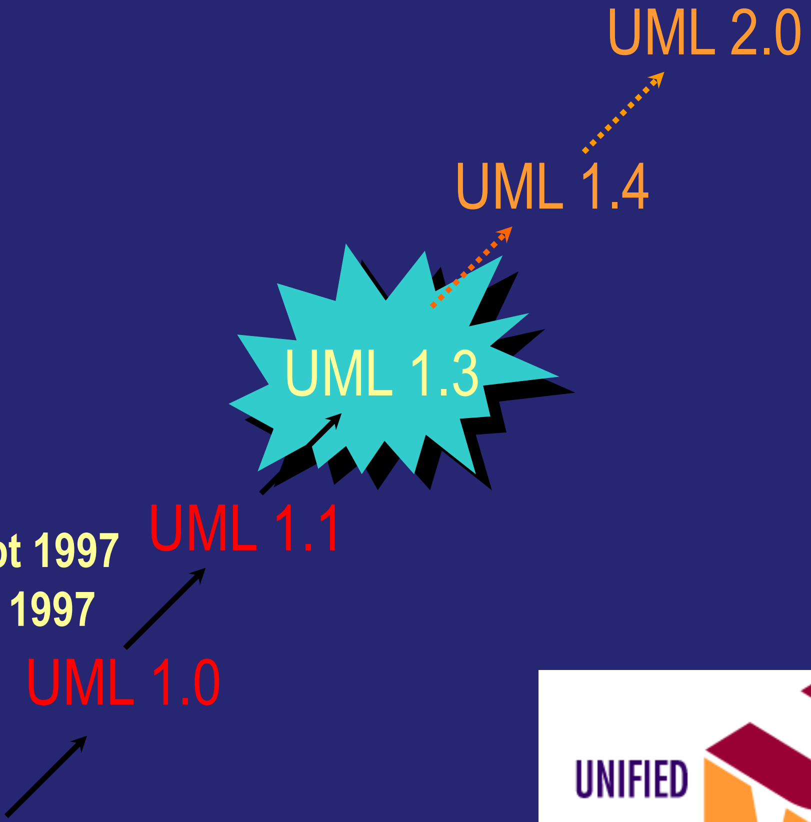
Current minor revision 1999

OMG Acceptance, Nov 1997
Final submission to OMG, Sept 1997
First submission to OMG, Jan 1997

UML partners

Web - June 1996

OOPSLA 95



UML 2.0

UML 1.4

UML 1.3

UML 1.1

UML 1.0

UNIFIED
MODELING
LANGUAGE

UML Concepts

- The UML may be used to visually model
  - The interaction of your application with the outside world.
  - The behavior of your application.
  - The structure of your system.
  - The architecture of your enterprise.
  - The components in your system.

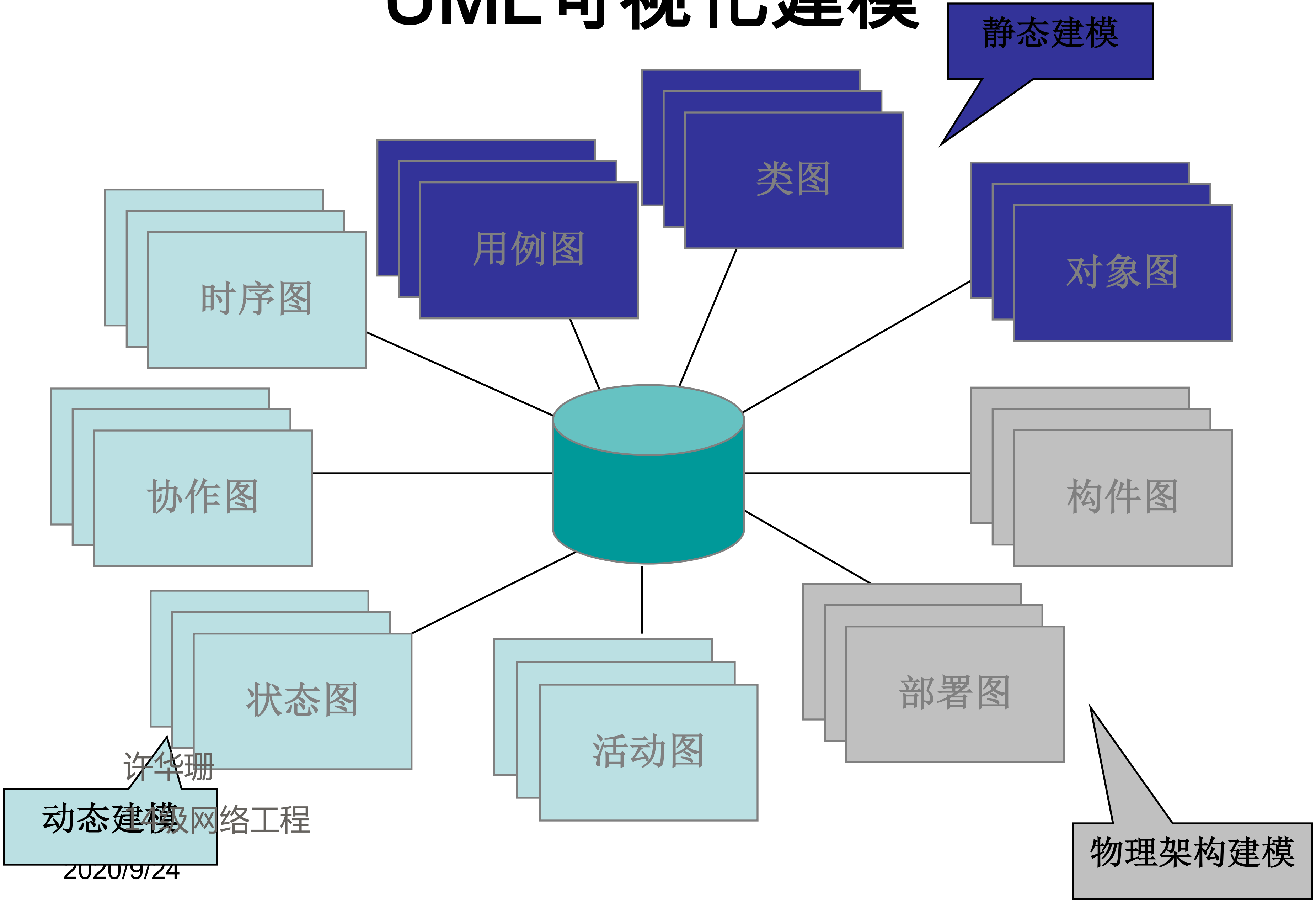
Where Are We?

- What is visual modeling?
- What is the UML?
- UML diagrams
- Extending UML notation

Agenda

- Use case model: use case diag. & activity diag.
- Interaction diagrams：sequence diag. & collaboration diag.
- Class model: class diag
- UML other diagrams: statechart diag., deployment diag., component diag., package.

UML可视化建模



作业

1. 何为UML，其有什么用途？
2. 简述UML产生的历史过程
3. UML一共有几个图？其名称分别是什么？其用途分别是什么？

Essentials of Visual Modeling with UML

Use-Case Modeling

Agenda

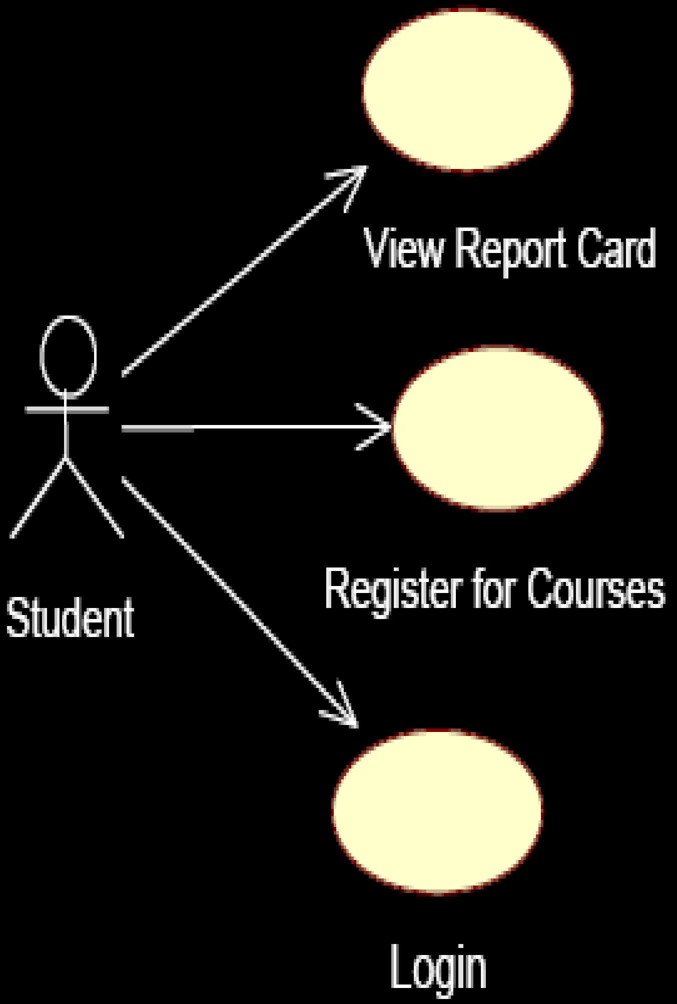
- Use case model: use case diag. & activity diag.
- Interaction diagrams：sequence diag. & collaboration diag.
- UML other diagrams: statechart diag., deployment diag., component diag., package diag.

Objectives

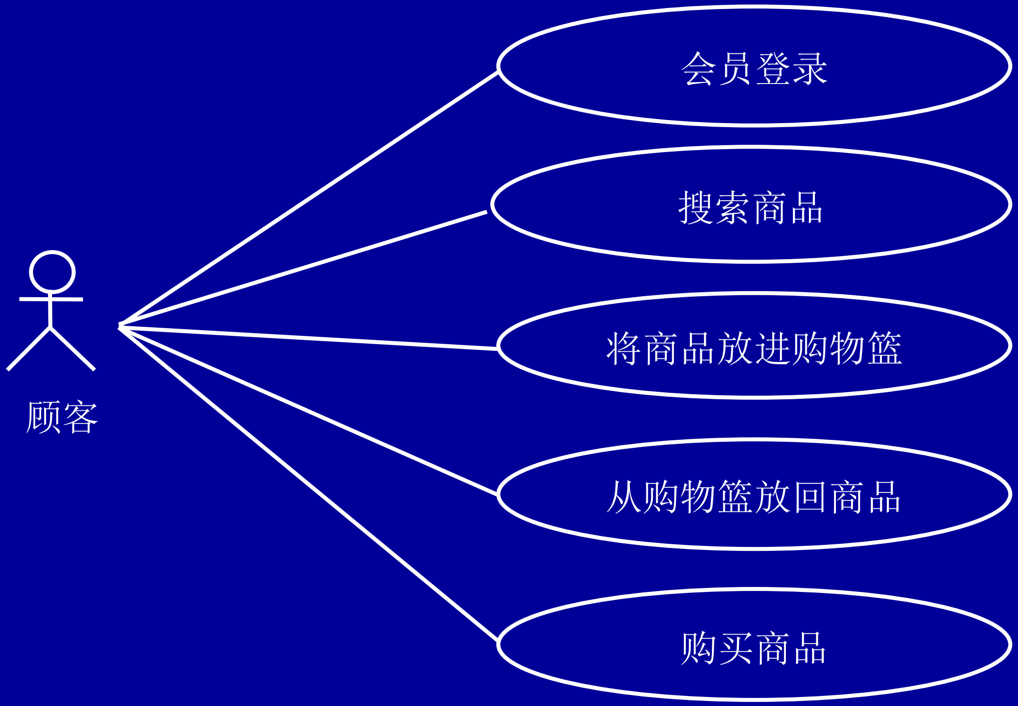
- Describe system behavior and show how to capture it in a model.
- Demonstrate how to read and interpret:
  - A use-case diagram
  - An activity diagram

What Is a Use-Case Model?

- A model that describes a system's functional requirements in terms of use cases.
- A model of the system's intended functions (use cases) and its environment (actors).



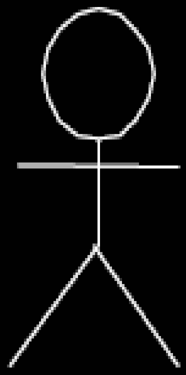
用例图实例



网上购物系统用例

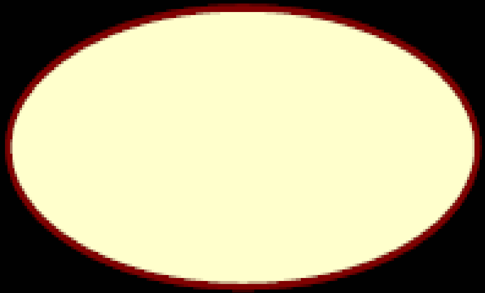
Major Concepts in Use-Case Modeling

- An actor represents anything that interacts with the system.



Actor

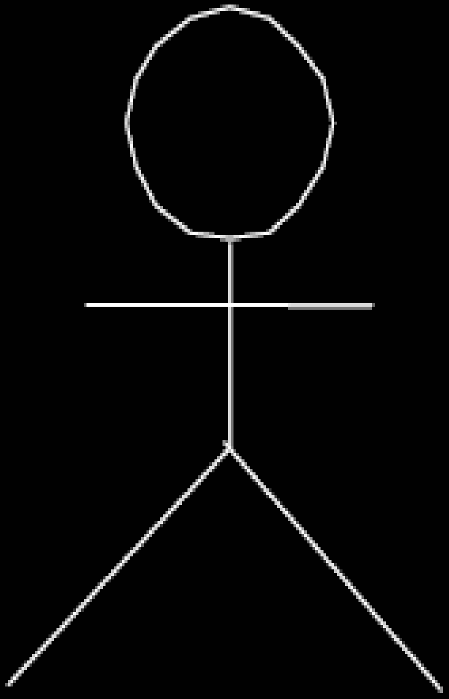
- A use case describes a sequence of events, performed by the system, that yields an observable result of value to a particular actor.



Use Case

What Is an Actor?

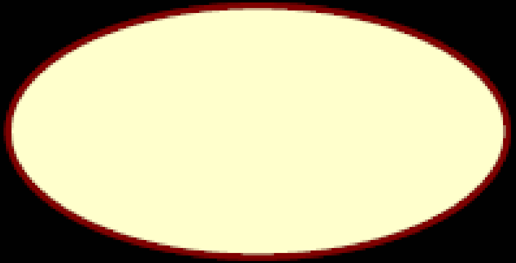
- Actors represent roles a user of the system can play.
- They can represent a human, a machine, or another system.
- They can actively interchange information with the system.
- They can be a giver of information.
- They can be a passive recipient of information.
- Actors are not part of the system.
  - Actors are EXTERNAL.



Actor

What Is a Use Case?

- Defines a set of use-case instances, where each instance is a sequence of actions a system performs that yields an observable result of value to a particular actor.
  - A use case models a dialogue between one or more actors and the system
  - A use case describes the actions the system takes to deliver something of value to the actor

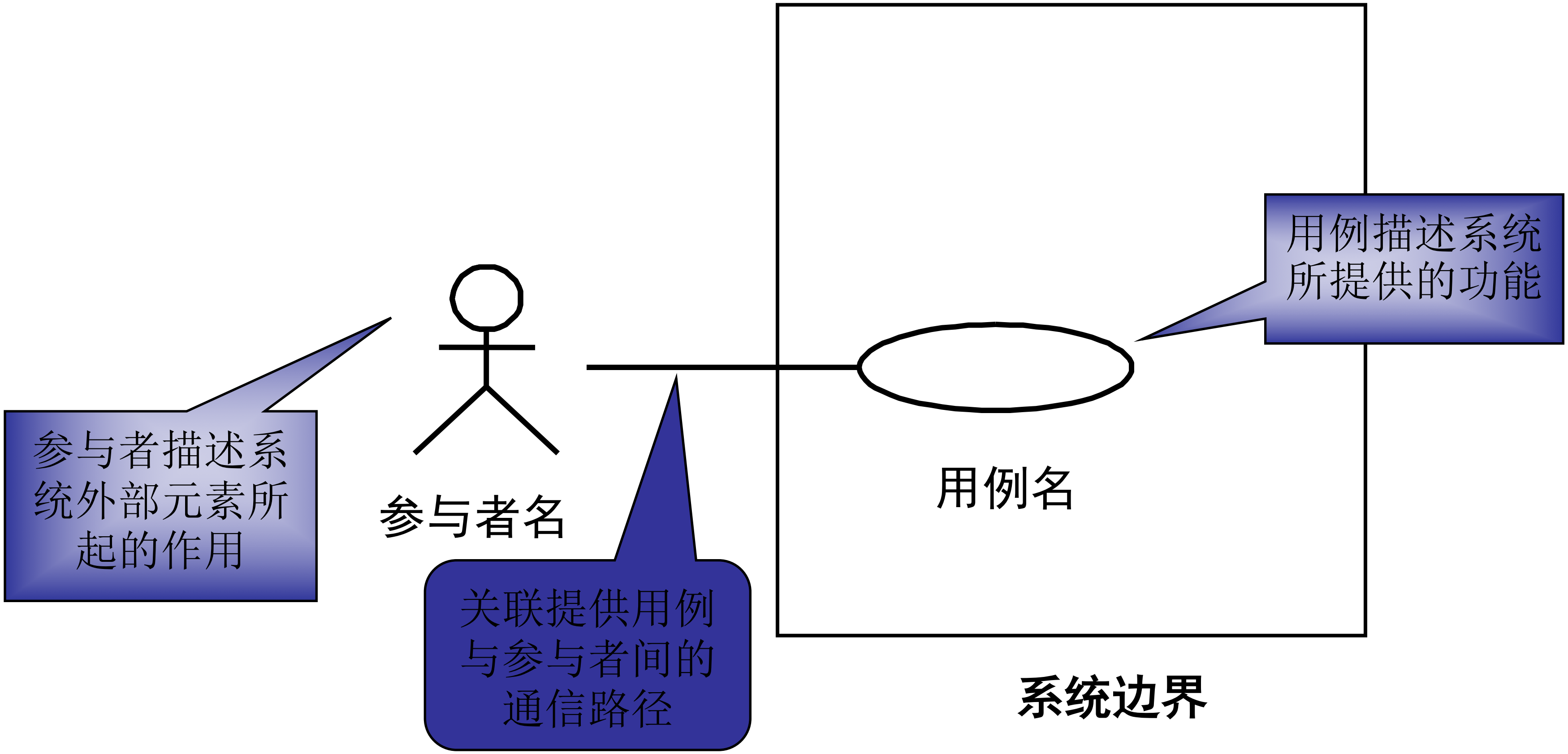


Use Case

用例图

- 从系统的外部用户的观点看系统应具有的功能
- 用例图主要用于对系统，子系统或类的行为进行建模
- 它只说明系统实现什么功能，而不必说明如何实现

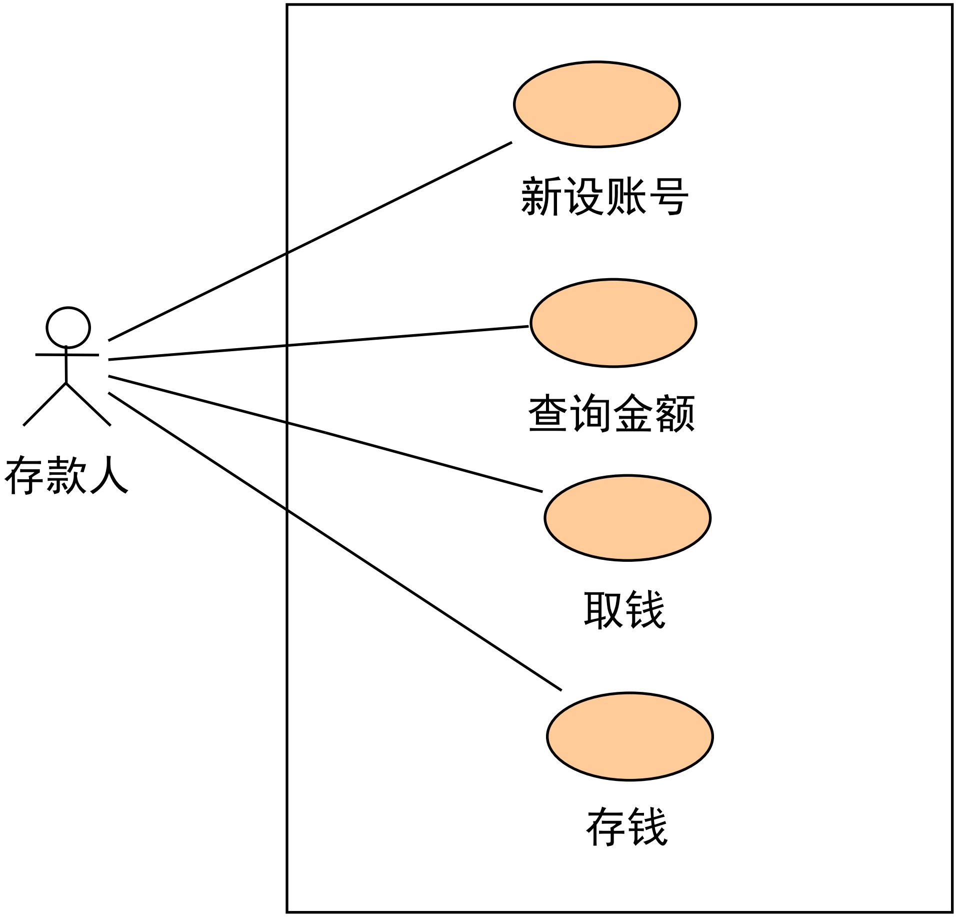
用例图的模型元素


用例描述系统所提供的功能
参与者描述系统外部元素所起的作用
参与者名
用例名
关联提供用例与参与者间的通信路径
系统边界

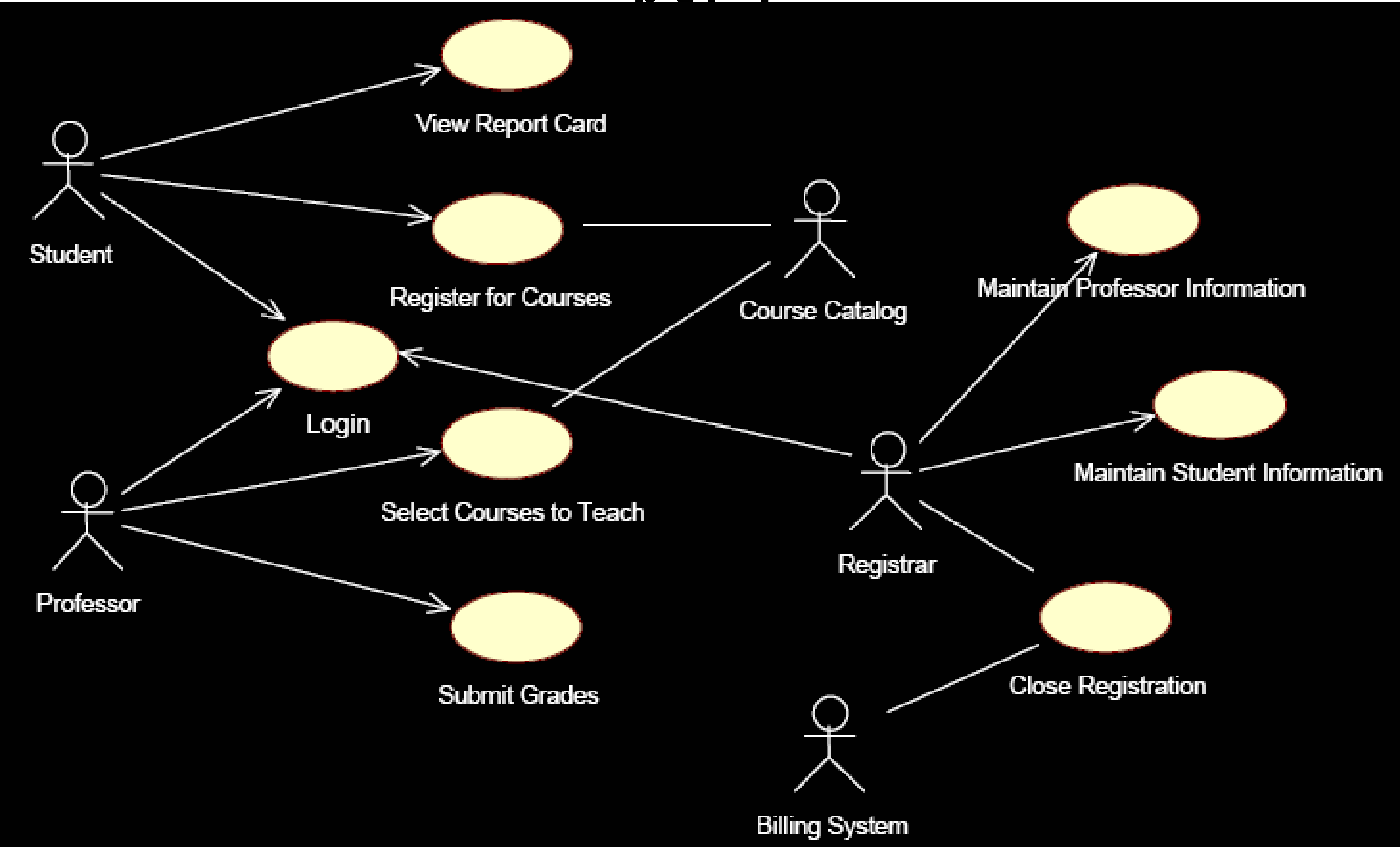
例题：试画用例图

- 需要描述的内容
  - 存款人新设一个账号
  - 存款人查询剩余金额
  - 存款人取钱
  - 存款人存钱

例题：试画用例图（题解）



Exercise:设计一个课程管理系统用例图



Exercise:设计一个课程管理系统用例图

- 管理员：维护教师、学生信息、关闭注册
- 学生：登陆、查看课表、注册课程
- 教师：登陆、开课报告、提交成绩
- 其它系统信息：关闭关闭注册系统

What Is an Activity Diagram?

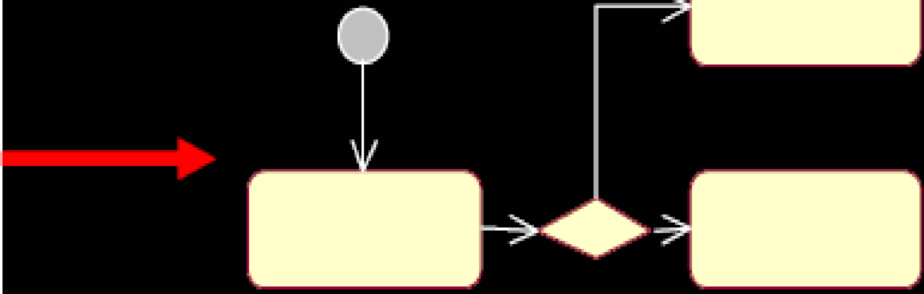
- An activity diagram in the use-case model can be used to capture the activities in a use case.
- It is essentially a flow chart, showing flow of control from activity to activity.

Flow of Events

This use case starts when the Registrar requests that the system close registration.

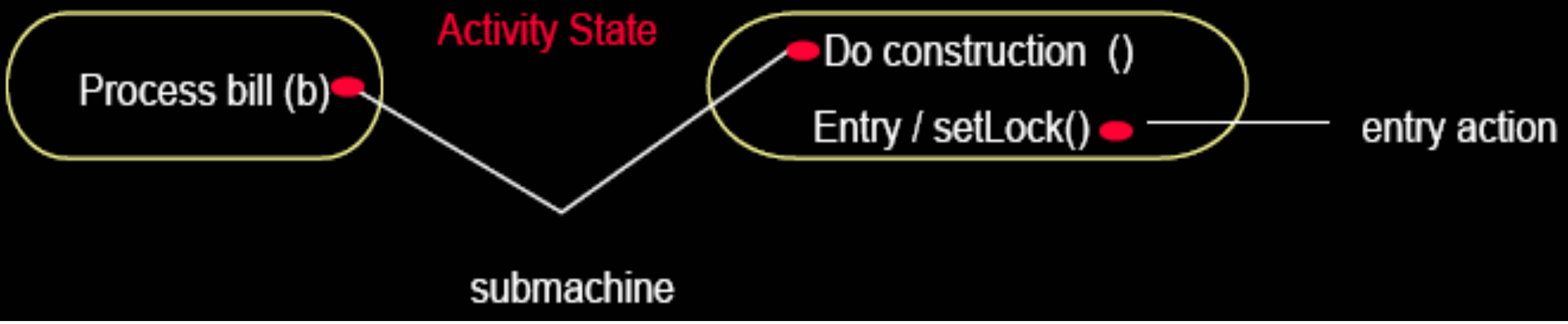
1. The system checks to see if registration is in progress. If it is, then a message is displayed to the Registrar and the use case terminates. The Close Registration processing cannot be performed if registration is in progress.

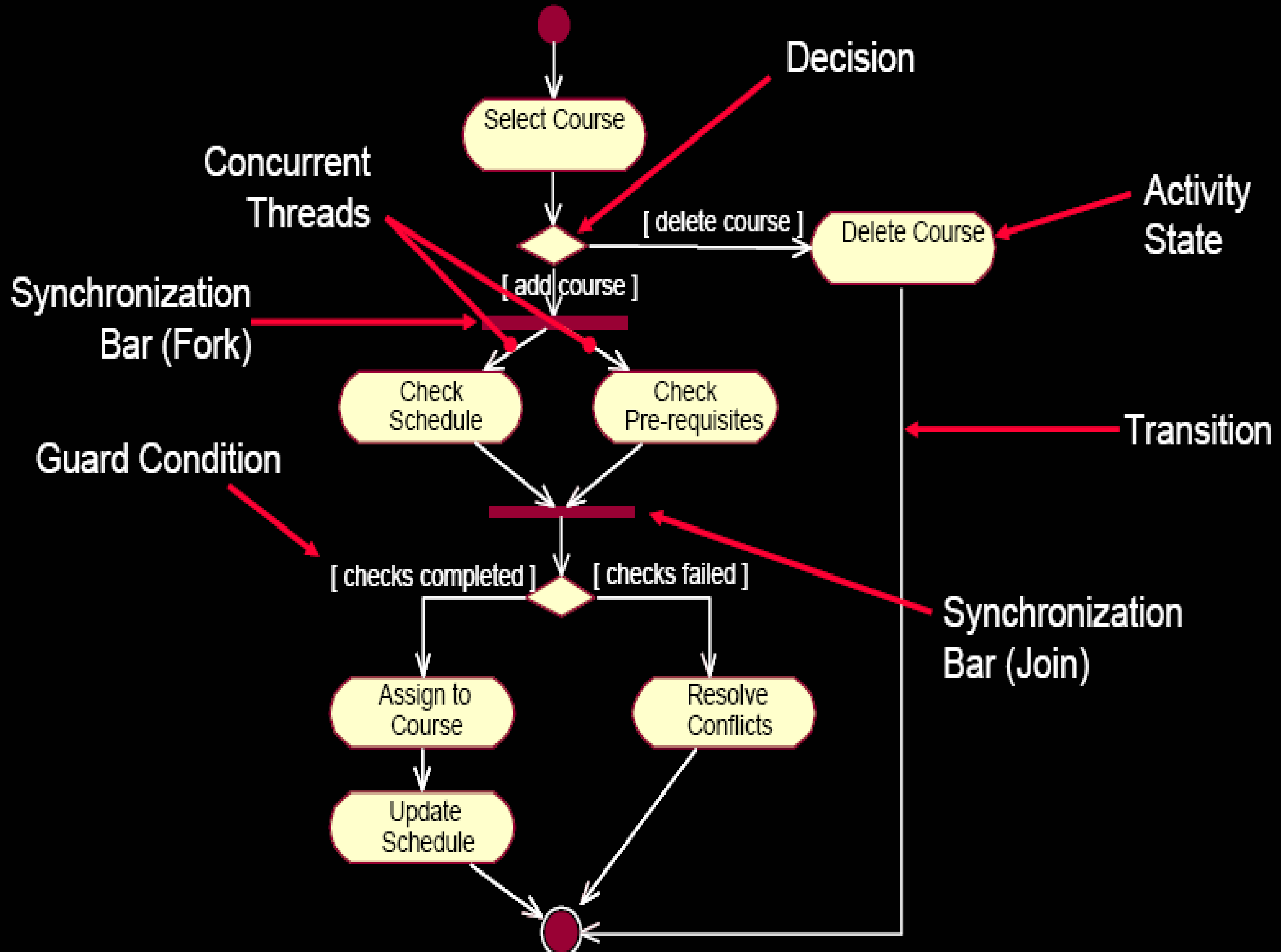
2. For each course offering, the system checks if a professor has signed up to teach the course offering and at least three students have registered. If so, the system commits the course offering for each schedule that contains it.



What Is an Activity State?

- The performance of an activity or step within the workflow.
- An activity is an operation that takes time to complete. It:
  - May have additional parts, such as entry and exit actions
  - Can have submachine specifications

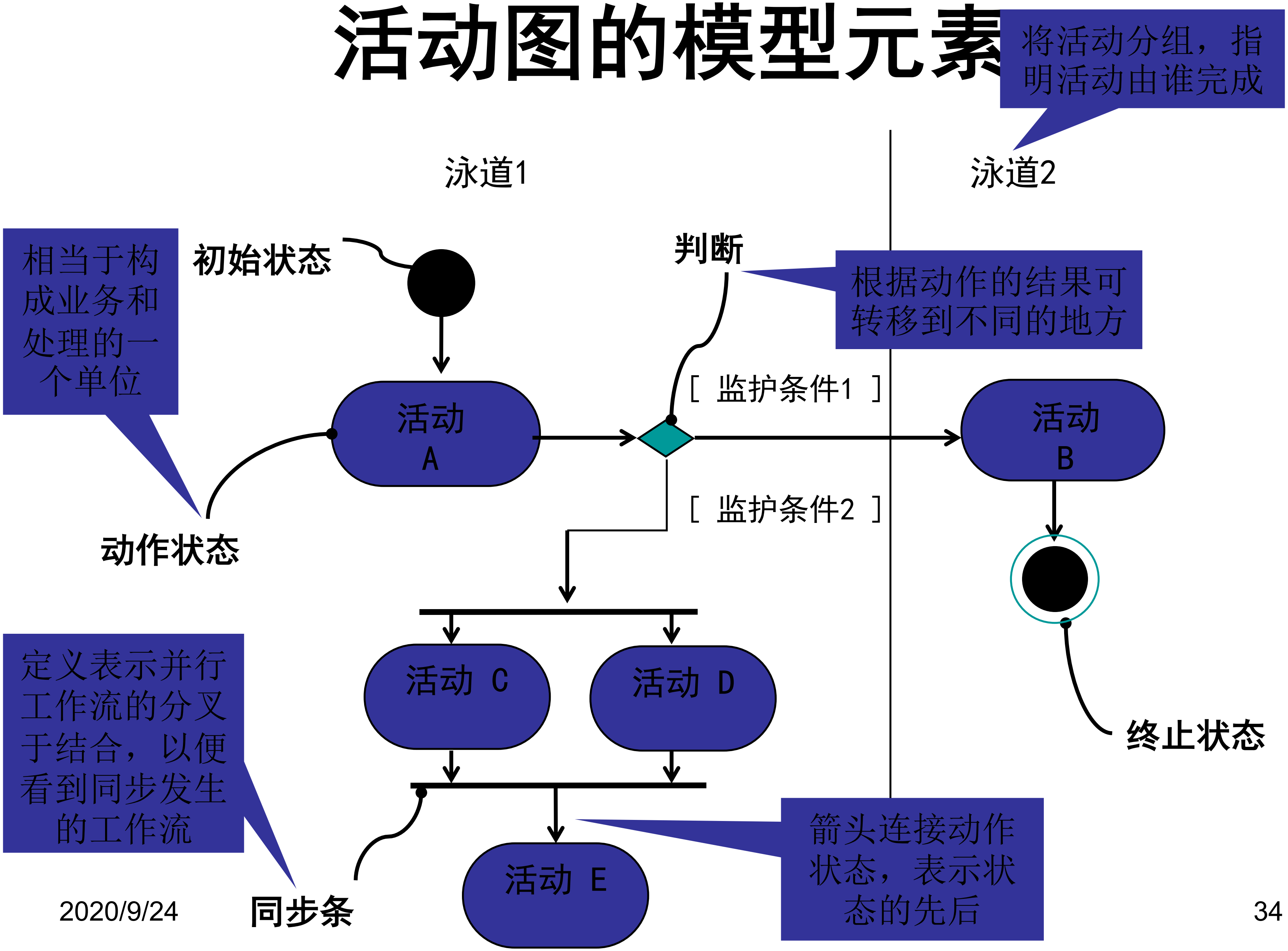



Decision
Select Course
Concurrent Threads
[delete course]
Delete Course
Activity State
Synchronization Bar (Fork)
[add course]
Check Schedule
Check Pre-requisites
Transition
Guard Condition
[checks completed]
[checks failed]
Synchronization Bar (Join)
Assign to Course
Resolve Conflicts
Update Schedule

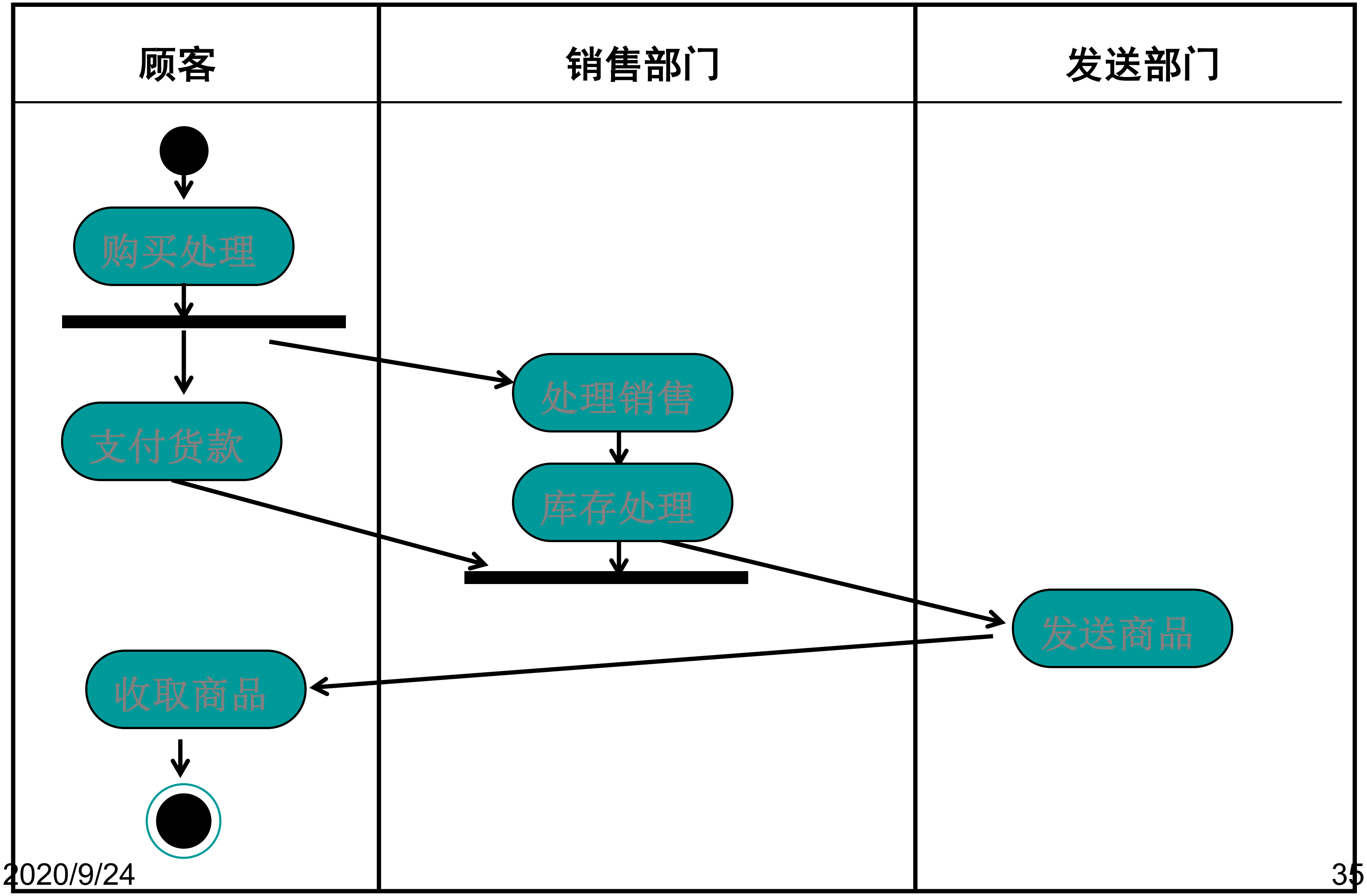
活动图

- 提供了对工作流进行建模的途径
- 活动图中的活动
  - 表示执行工作流中一组的动作
  - 一旦结束，控制流将自动转移到下一个活动，或通过转换进入下一个状态

活动图的模型元素


将活动分组，指明活动由谁完成
泳道1
泳道2
相当于构成业务和处理的一个单位
初始状态
判断
根据动作的结果可转移到不同的地方
[监护条件1]
活动 A
活动 B
[监护条件2]
动作状态
活动 C
活动 D
终止状态
定义表示并行工作流的分叉于结合，以便看到同步发生的工作流
活动 E
箭头连接动作状态，表示状态的先后
同步条

活动图示例



作业

- What is system behavior?
- What is a use-case model? What are its benefits?
- What is an actor? A use case?
- What is an activity diagram?

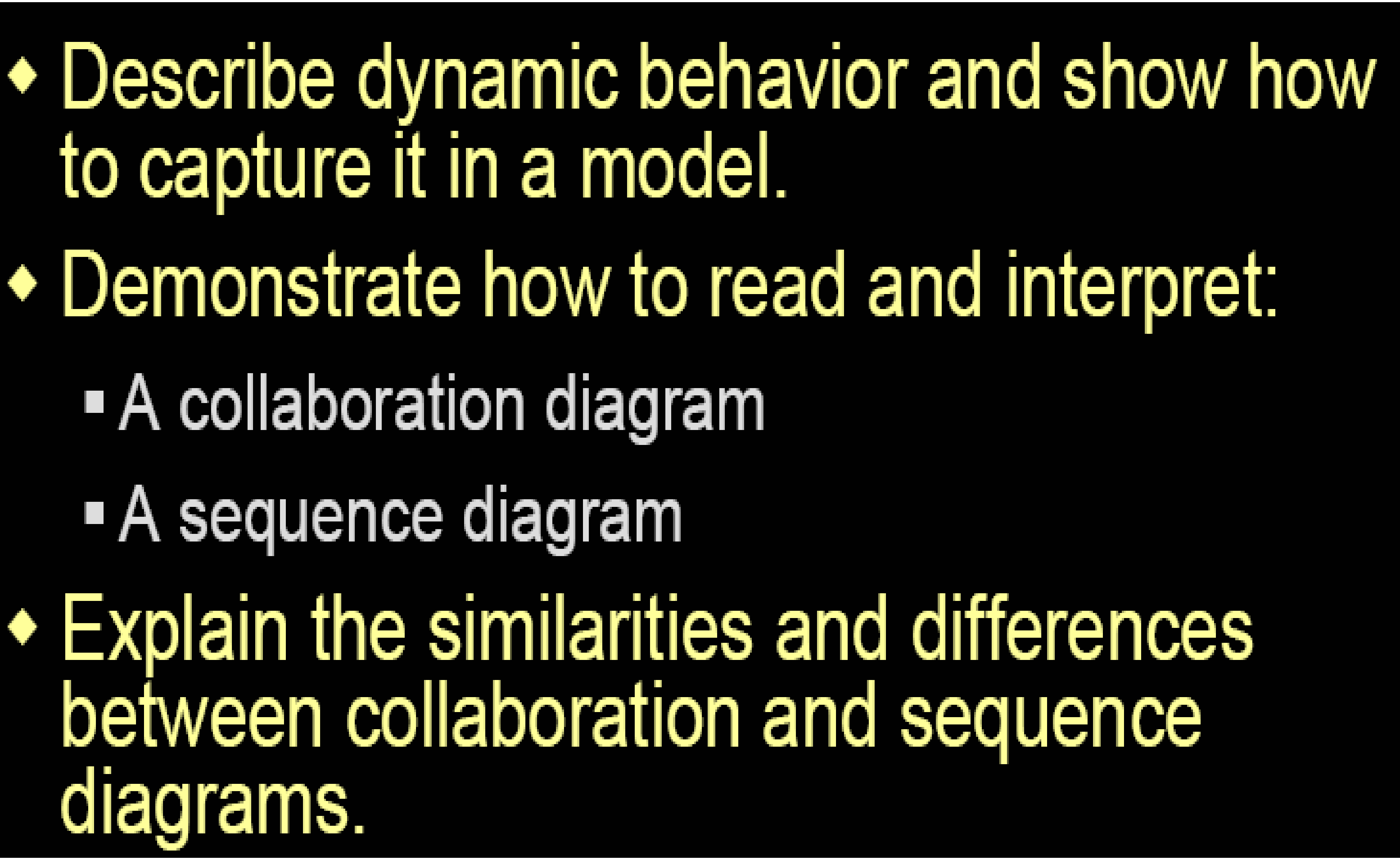
Essentials of Visual Modeling with UML

Interaction Diagrams

agenda

- Use case model: use case diag. & activity diag.
- Interaction diagrams：sequence diag. & collaboration diag.
- Class model: class diag
- UML other diagrams: statechart diag., deployment diag., component diag., package diag.

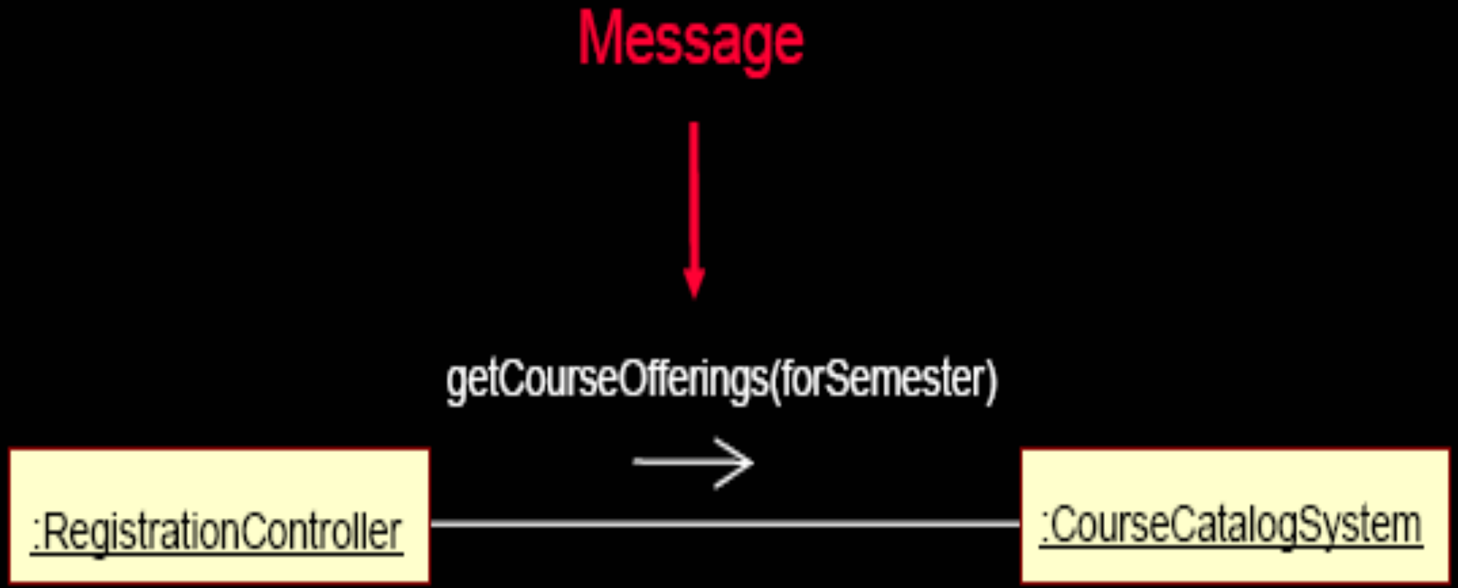
Objectives



- Describe dynamic behavior and show how to capture it in a model.
- Demonstrate how to read and interpret:
  - A collaboration diagram
  - A sequence diagram
- Explain the similarities and differences between collaboration and sequence diagrams.

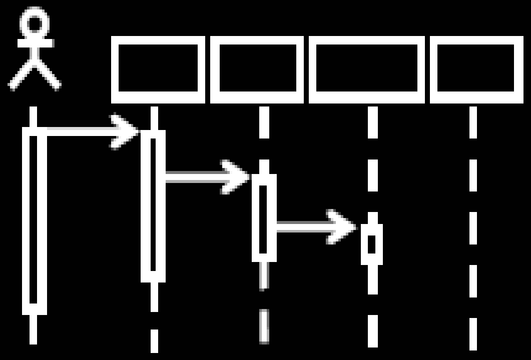
Objects Interact with Messages

- A message shows how one object asks another object to perform an operation.

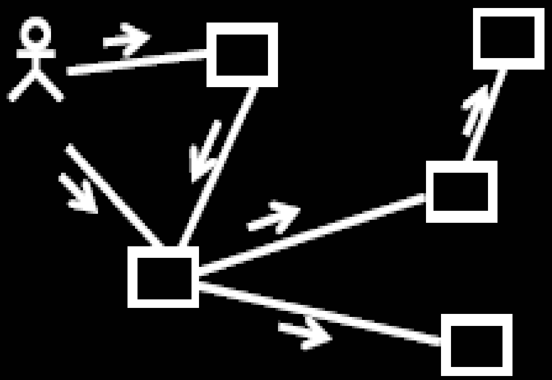


What Is an Interaction Diagram?

- An interaction diagram shows an interaction that consists of a set of objects and their relationships, including the messages that may be dispatched among them.
- It models the dynamic aspects of a system.



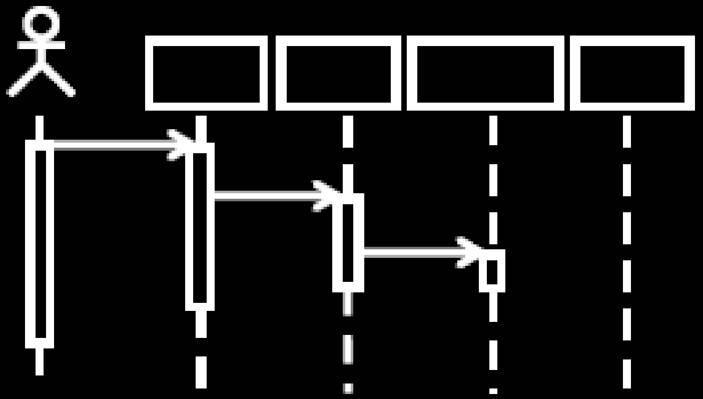
Sequence Diagrams



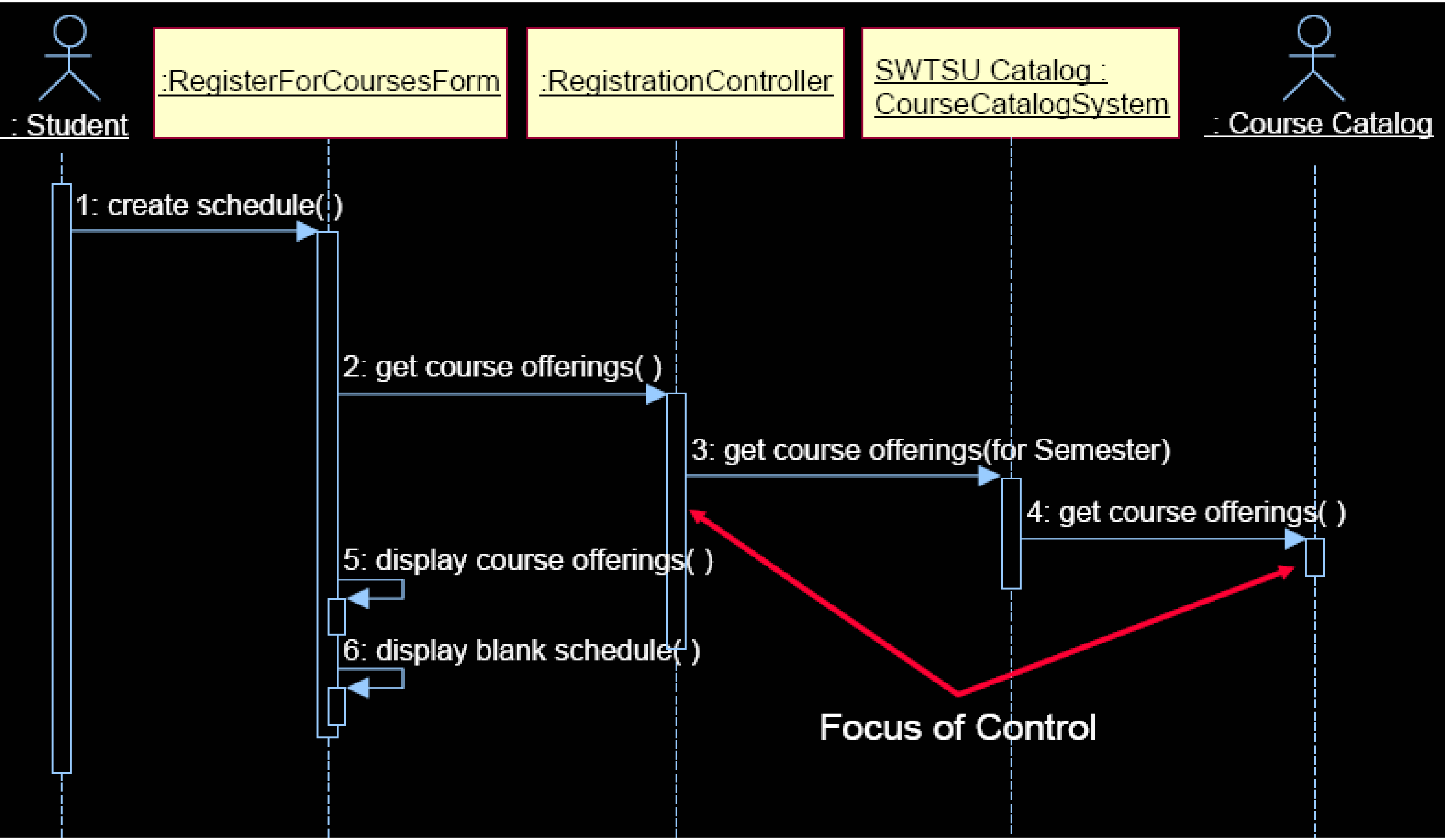
Collaboration Diagrams

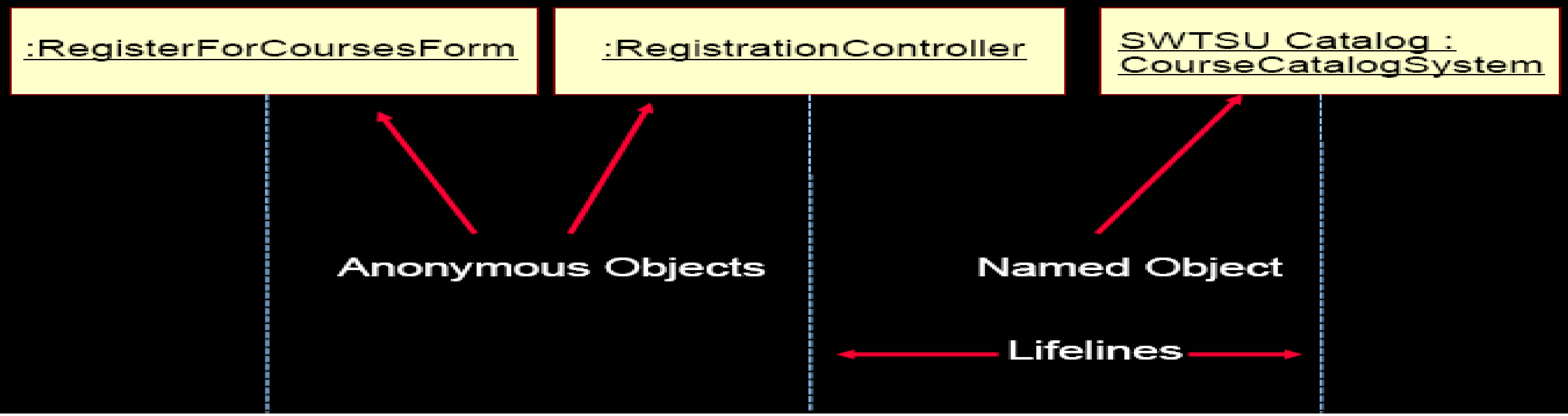
What Is a Sequence Diagram?

- A sequence diagram is an interaction diagram that emphasizes the time orderin of messages.
- The diagram shows:
  - The objects participating in the interaction.
  - The sequence of messages exchanged.



Sequence Diagrams


: Student
:RegisterForCoursesForm
:RegistrationController
SWTSU Catalog : CourseCatalogSystem
: Course Catalog
1: create schedule( )
2: get course offerings( )
3: get course offerings(for Semester)
4: get course offerings( )
5: display course offerings( )
6: display blank schedule( )
Focus of Control



:RegisterForCoursesForm

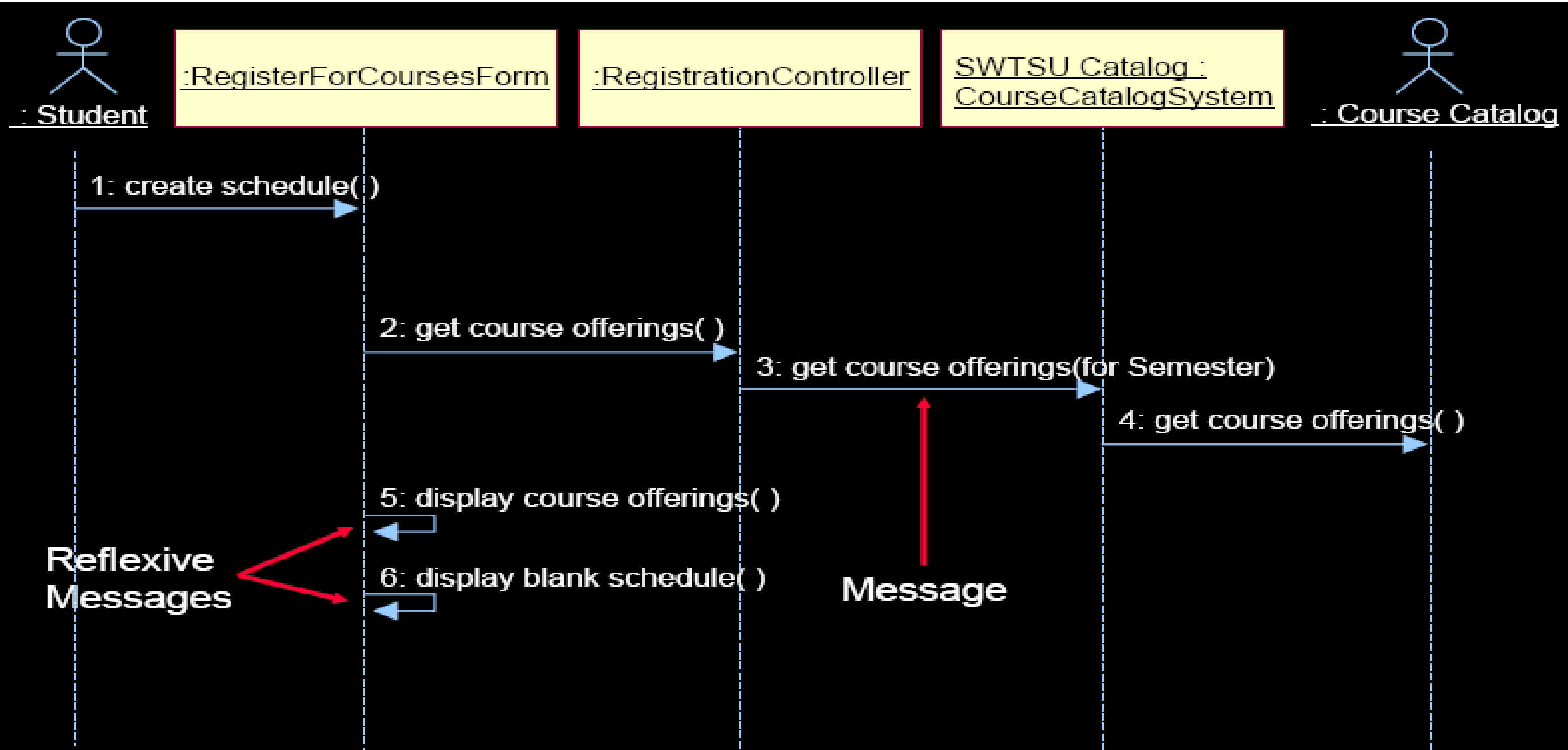
:RegistrationController

SWTSU Catalog : CourseCatalogSystem

Anonymous Objects

Named Object

Lifelines



: Student

:RegisterForCoursesForm

:RegistrationController

SWTSU Catalog : CourseCatalogSystem

: Course Catalog

1: create schedule( )

2: get course offerings( )

3: get course offerings(for Semester)

4: get course offerings( )

5: display course offerings( )

6: display blank schedule( )

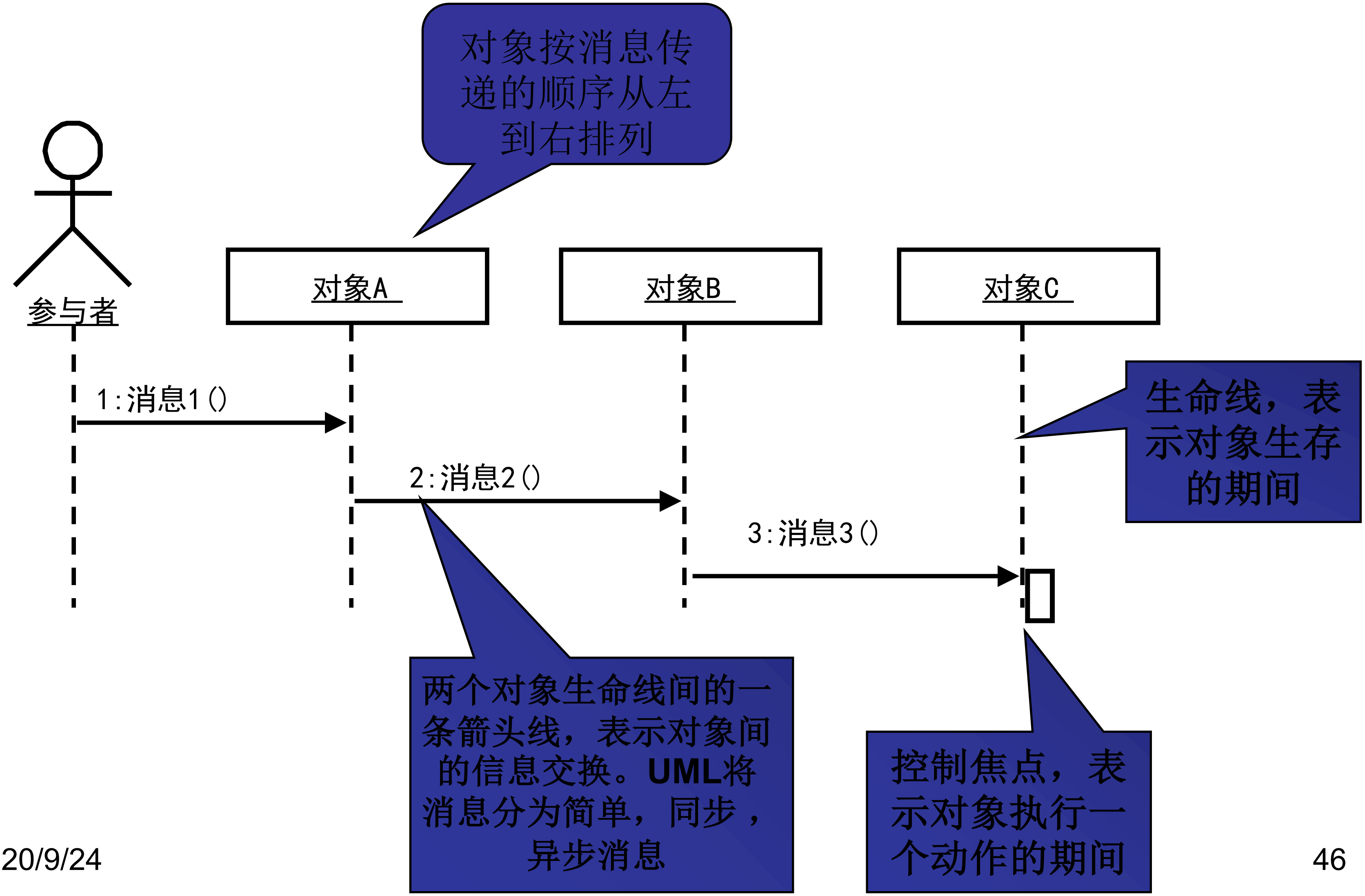
Reflexive Messages

Message

时序图

- 时序图描述了在时间上对象交互的安排
- 图形展现了
  - 多个交互对象
  - 信息交流的序列
- 时序图包含
  - 对象
  - 对象的生命线
  - 按顺序对象间的信息交流
  - 控制焦点（可选的）

时序图的模型元素



对象按消息传递的顺序从左到右排列

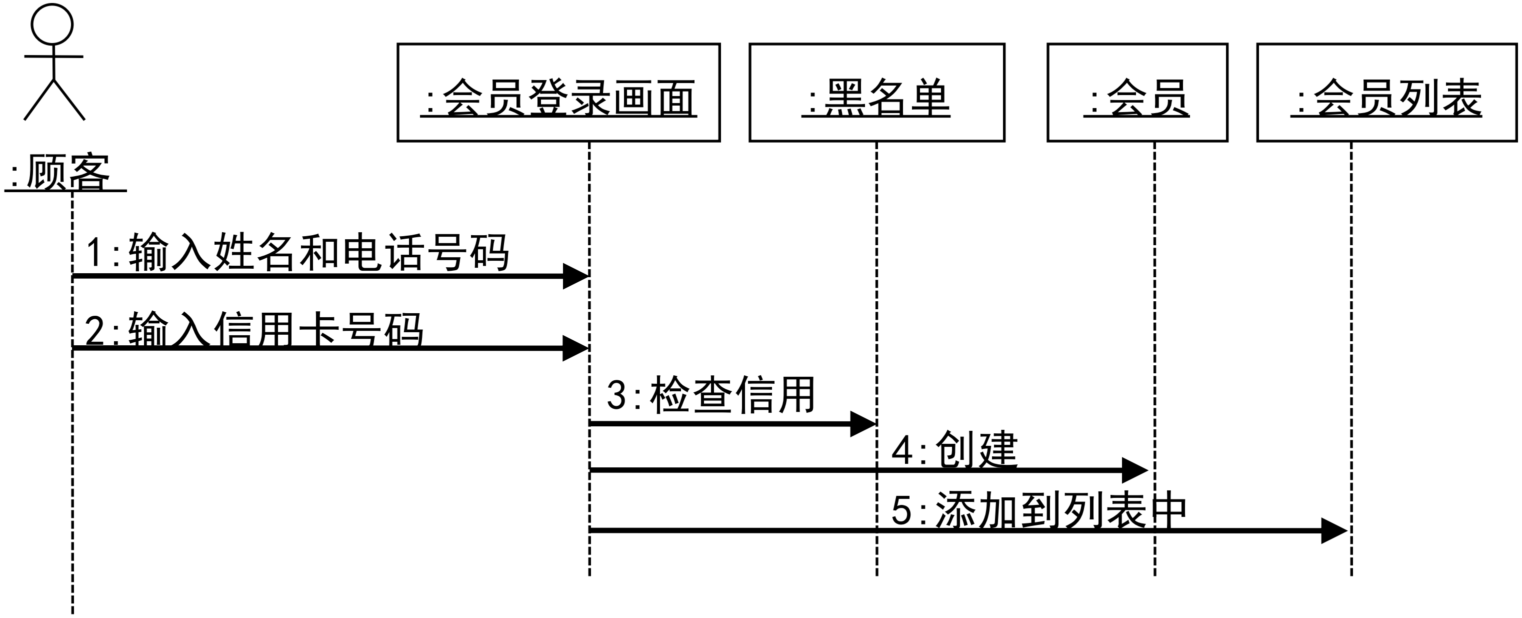
生命线，表示对象生存的期间

两个对象生命线间的一条箭头线，表示对象间的信息交换。UML将消息分为简单，同步，异步消息

控制焦点，表示对象执行一个动作的期间

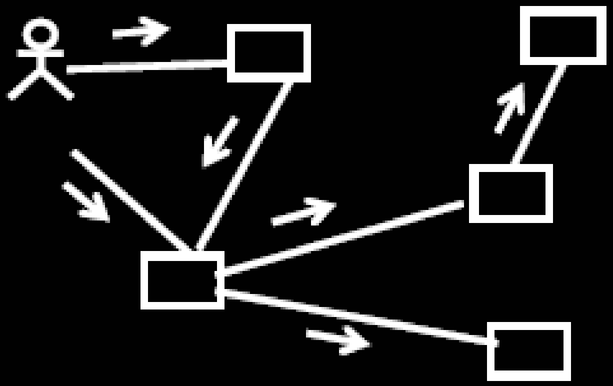
时序图示例

会员登录的时序图

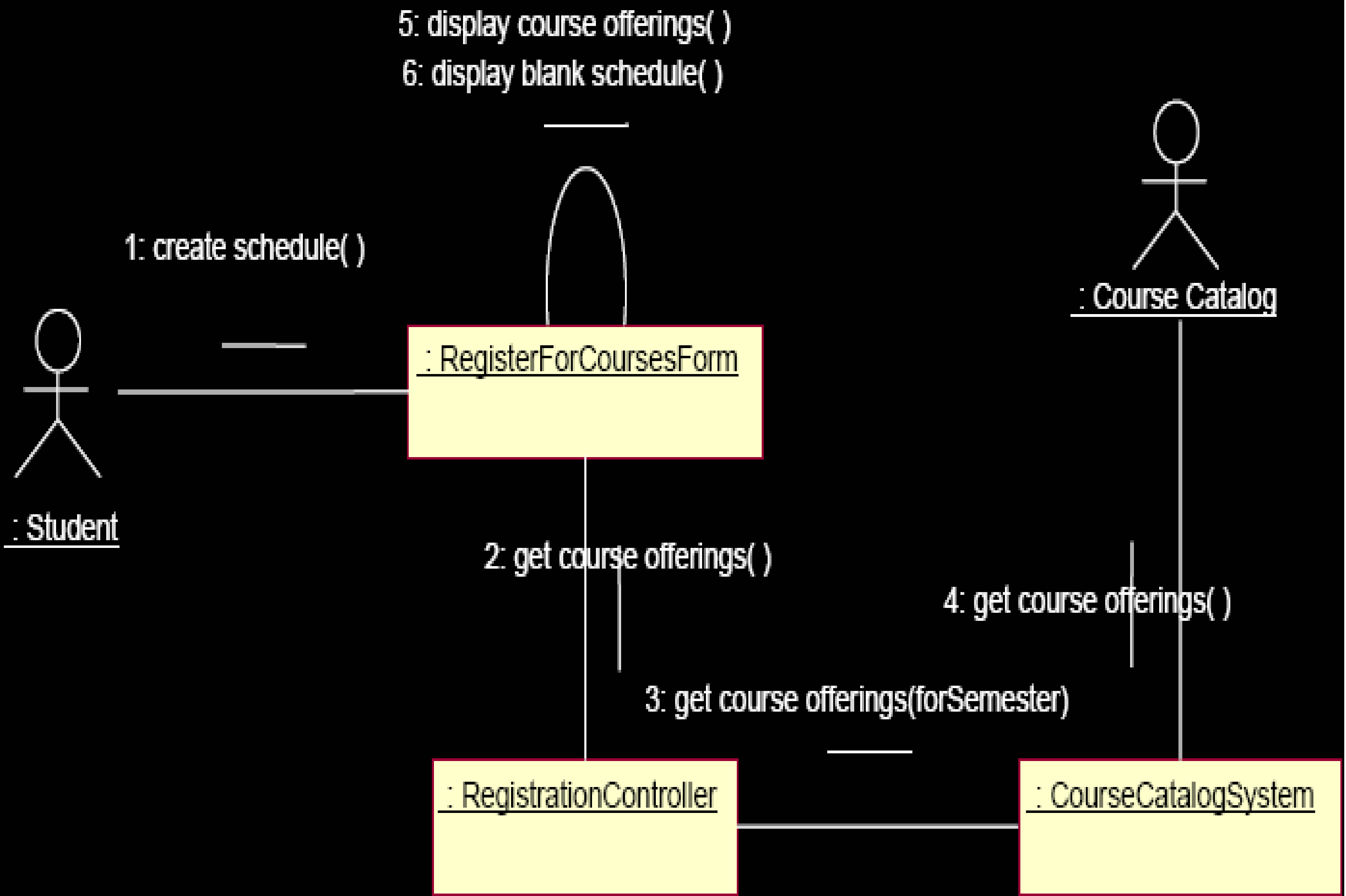


What Is a Collaboration Diagram?

- A collaboration diagram emphasizes the organization of the objects that participate in an interaction.
- The collaboration diagram shows:
  - The objects participating in the interaction.
  - Links between the objects.
  - Messages passed between the objects.



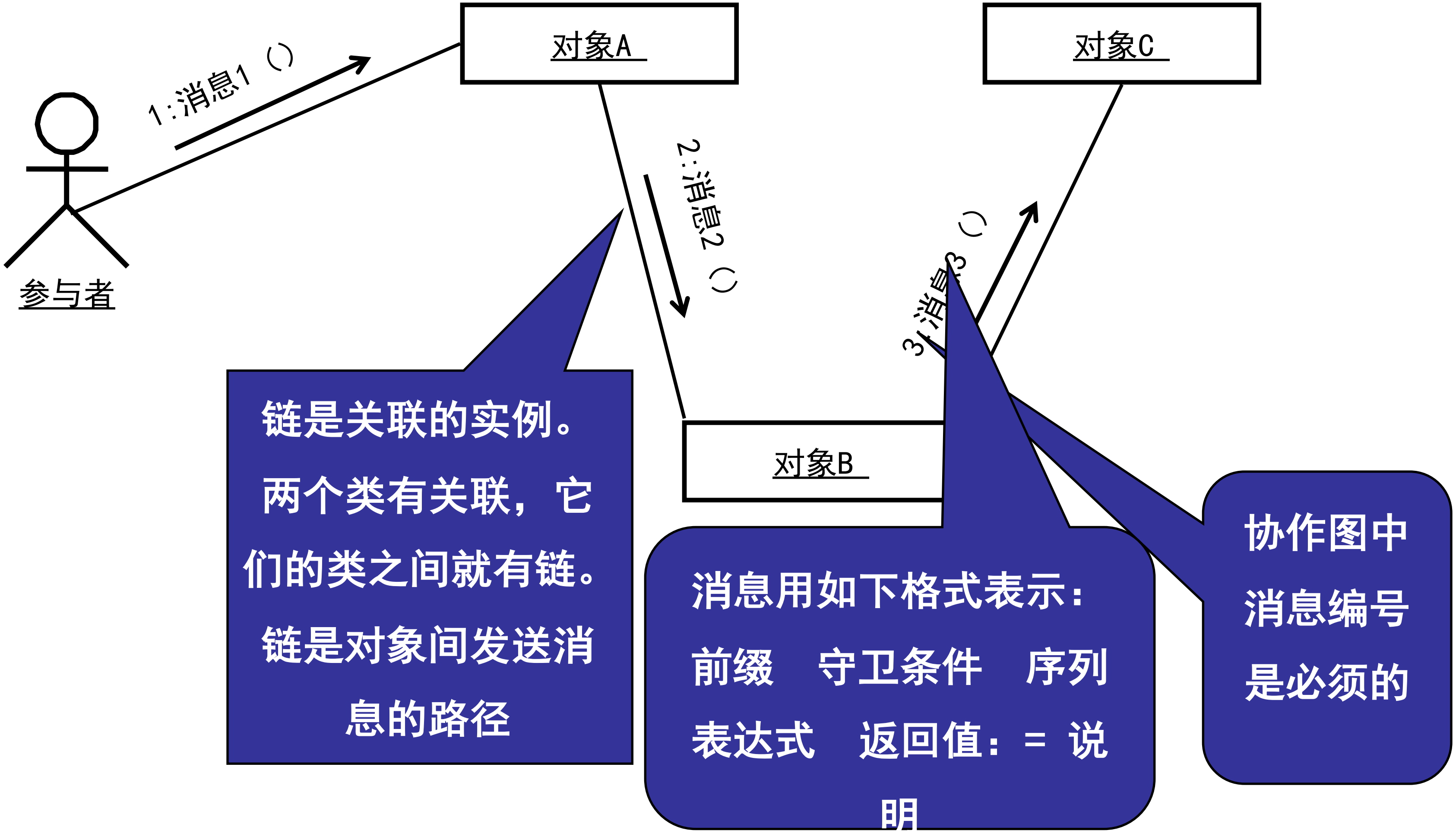
Collaboration Diagrams


5: display course offerings( )
6: display blank schedule( )
1: create schedule( )
: Course Catalog
: RegisterForCoursesForm
: Student
2: get course offerings( )
4: get course offerings( )
3: get course offerings(forSemester)
: RegistrationController
: CourseCatalogSystem

协作图

- 协作图是强调发送和接收消息的对象间的结构组织的交互图。在图形上，协作图是顶点和弧的结合
- 协作图包含
  - 对象
  - 链
  - 消息

协作图的模型元素



链是关联的实例。两个类有关联，它们的类之间就有链。链是对象间发送消息的路径

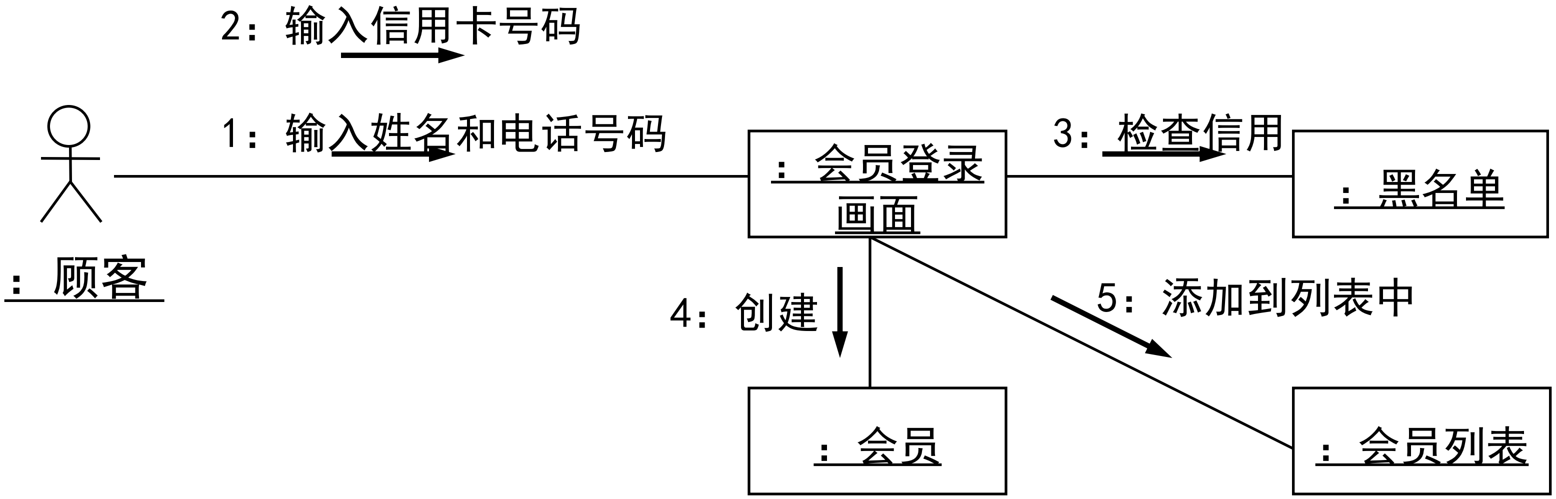
消息用如下格式表示：前缀　守卫条件　序列表达式　返回值：= 说明

协作图中消息编号是必须的

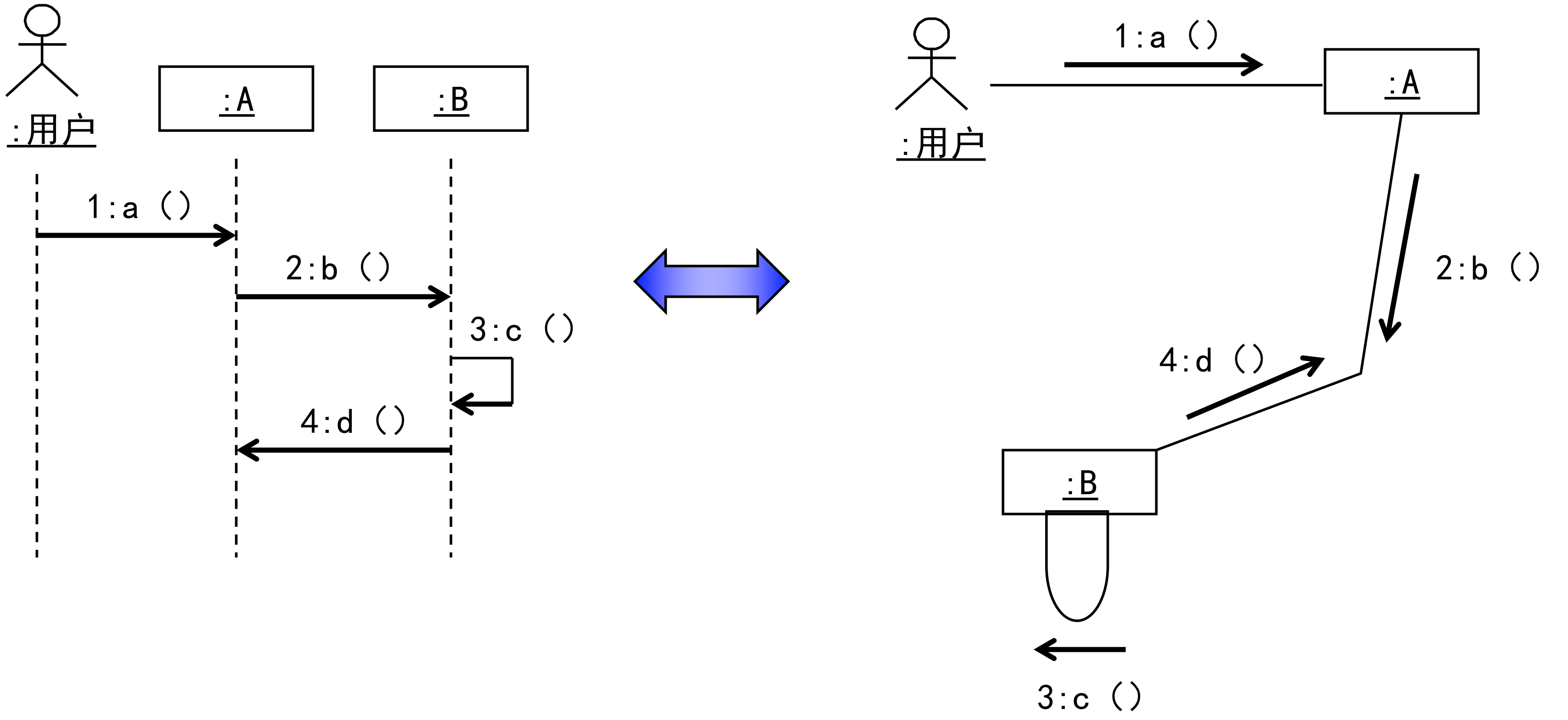
2020/9/24

协作图示例

会员登录的协作图



协作图和时序图是等价的



Discussion

- What is the purpose of an interaction diagram?
- What is a sequence diagram? A collaboration diagram?
- What are the similarities between sequence and collaboration diagrams?
- What are the differences between sequence and collaboration diagrams?

Sequence and Collaboration Diagram Differences

- Collaboration diagrams
  - Show relationships in addition to interactions
  - Better for visualizing patterns of collaboration
  - Better for visualizing all of the effects on a given object
  - Easier to use for brainstorming sessions
- Sequence diagrams
  - Show the explicit sequence of messages
  - Show focus of control
  - Better for visualizing overall flow
  - Better for real-time specifications and for complex scenarios

Essentials of Visual Modeling with UML

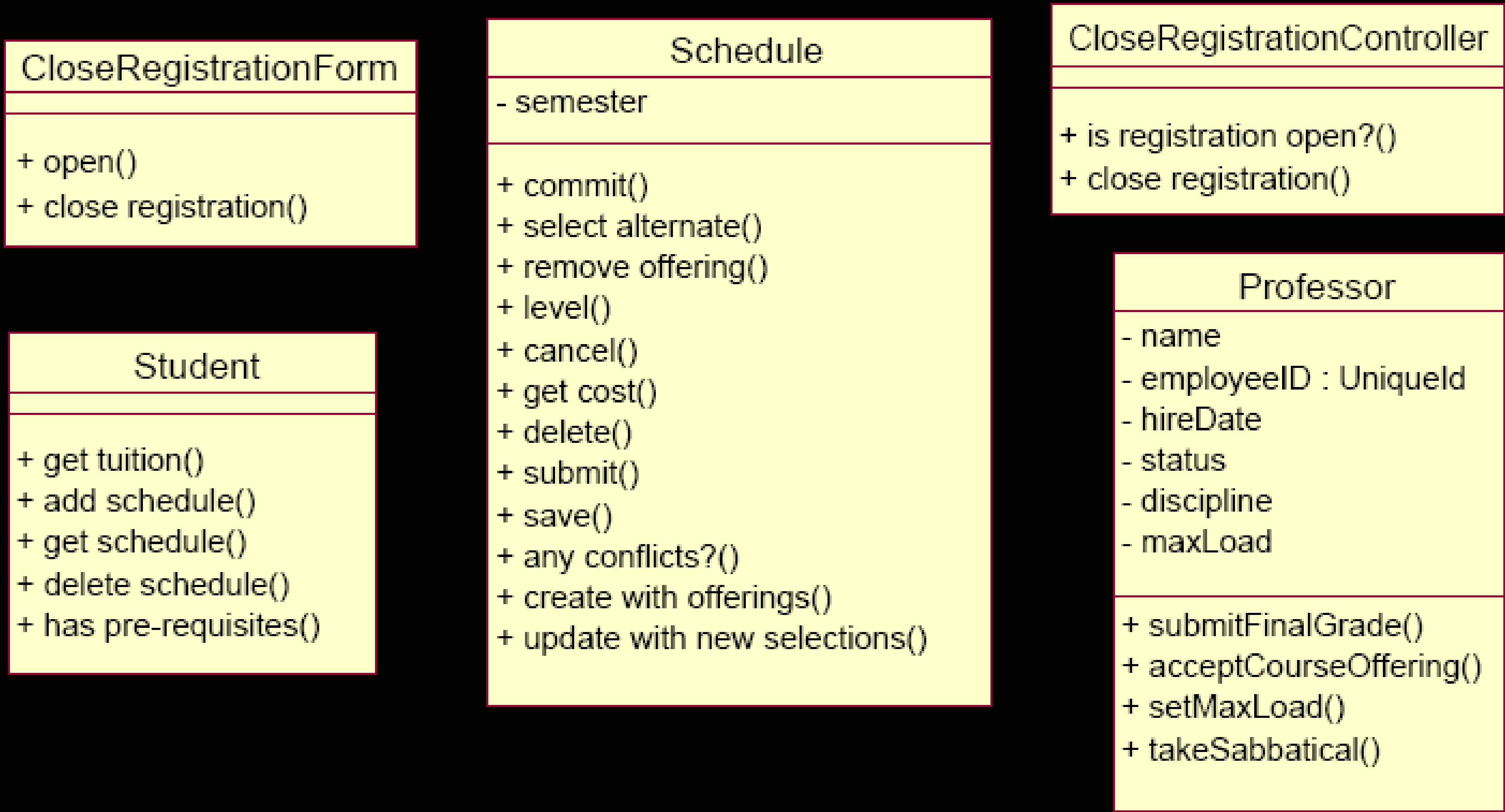
Class Diagrams

Agenda

- Use case model: use case diag. & activity diag.
- Interaction diagrams：sequence diag. & collaboration diag.
- Class models: class diag.
- UML other diagrams: statechart diag., deployment diag., component diag., package diag.

What Is a Class Diagram?

- Static view of a system



CloseRegistrationForm

+ open()
+ close registration()

Student

+ get tuition()
+ add schedule()
+ get schedule()
+ delete schedule()
+ has pre-requisites()

Schedule

- semester

+ commit()
+ select alternate()
+ remove offering()
+ level()
+ cancel()
+ get cost()
+ delete()
+ submit()
+ save()
+ any conflicts?()
+ create with offerings()
+ update with new selections()

CloseRegistrationController

+ is registration open?()
+ close registration()

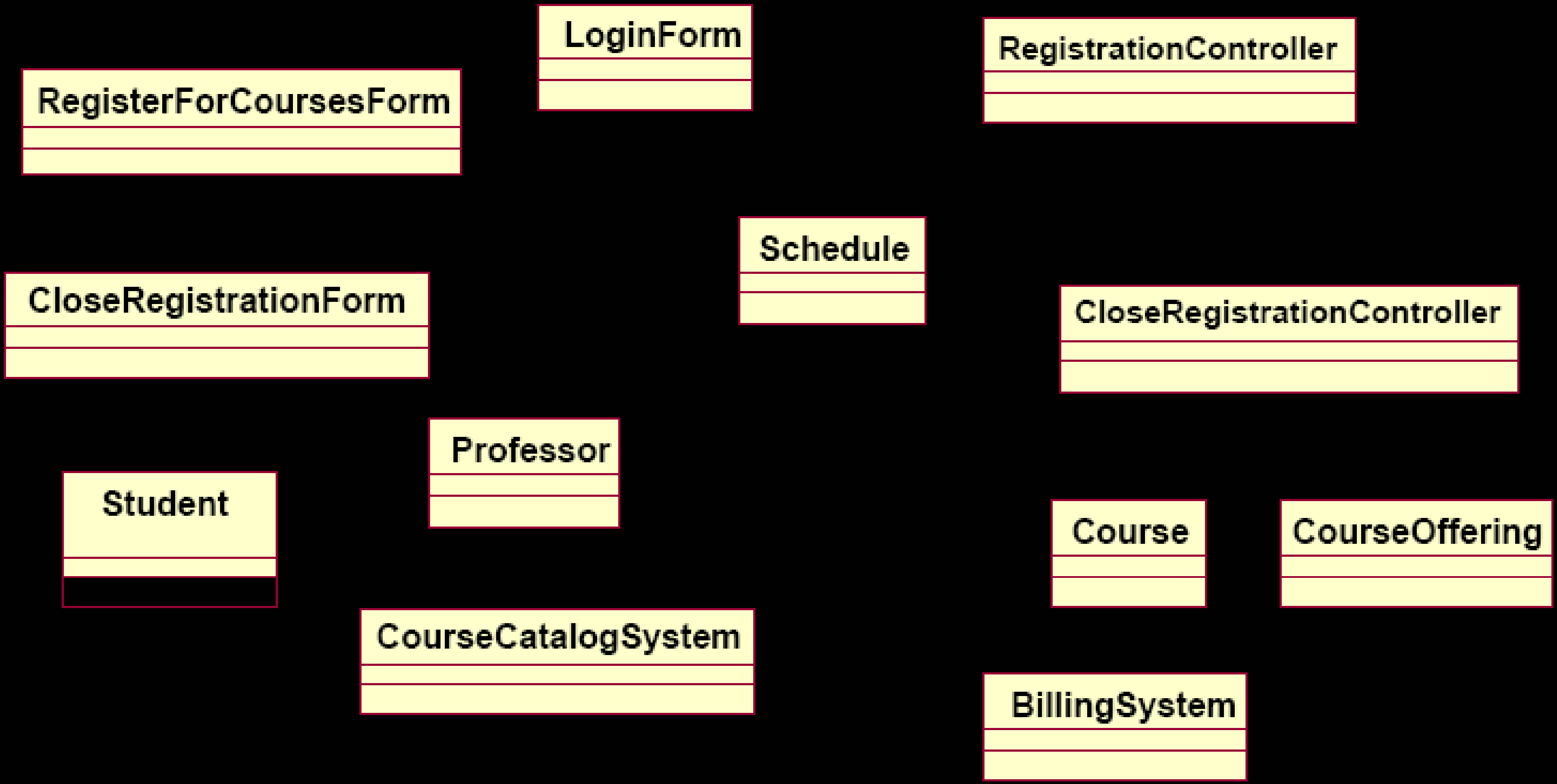
Professor

- name
- employeeID : UniqueId
- hireDate
- status
- discipline
- maxLoad

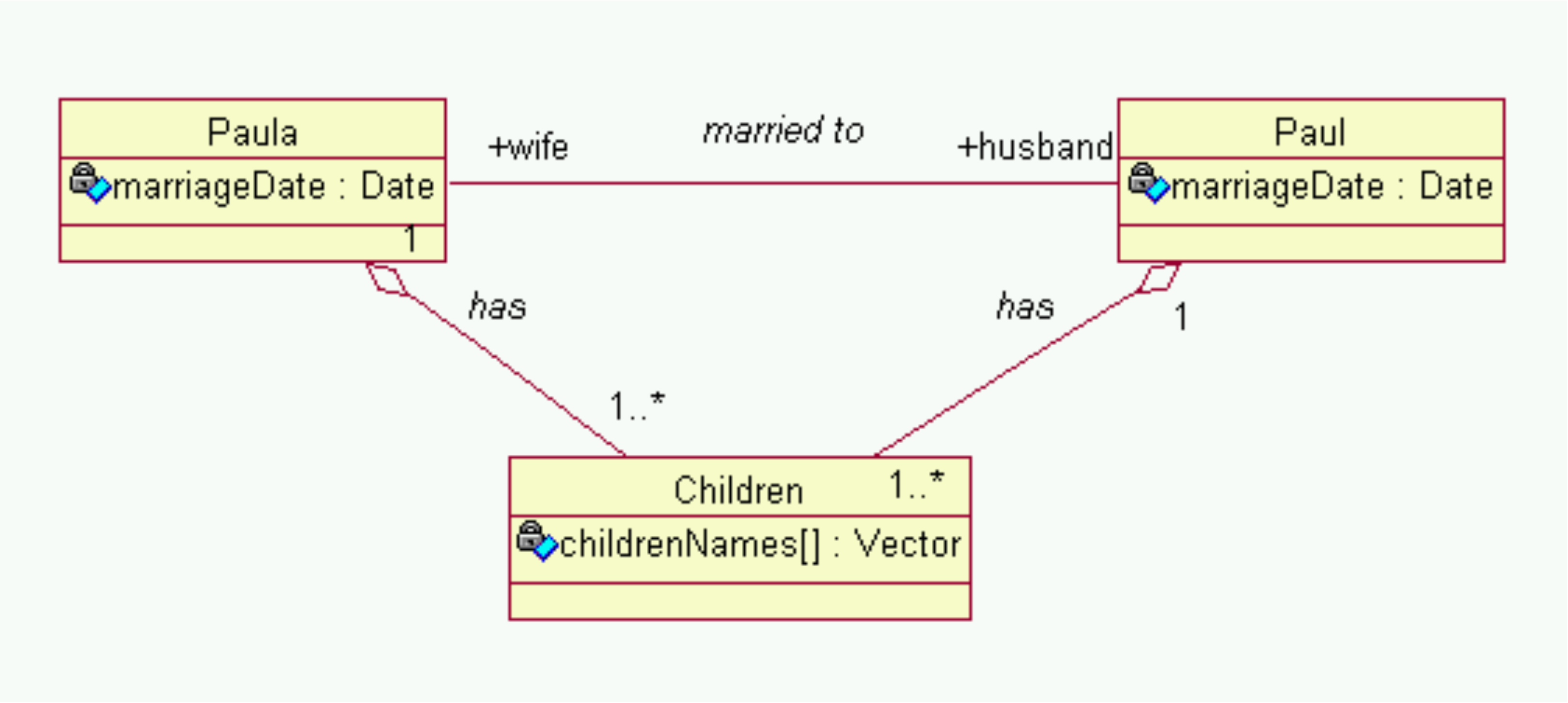
+ submitFinalGrade()
+ acceptCourseOffering()
+ setMaxLoad()
+ takeSabbatical()

Example: Class Diagram

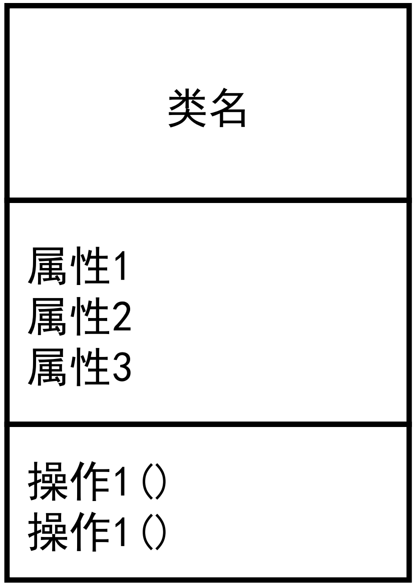
- Is there a better way to organize class diagrams?



类图


Paula
marriageDate : Date
+wife
married to
+husband
Paul
marriageDate : Date
1
has
has
1
1..*
Children
1..*
childrenNames[] : Vector

类的表示法



类名

属性1
属性2
属性3

操作1()
操作1()

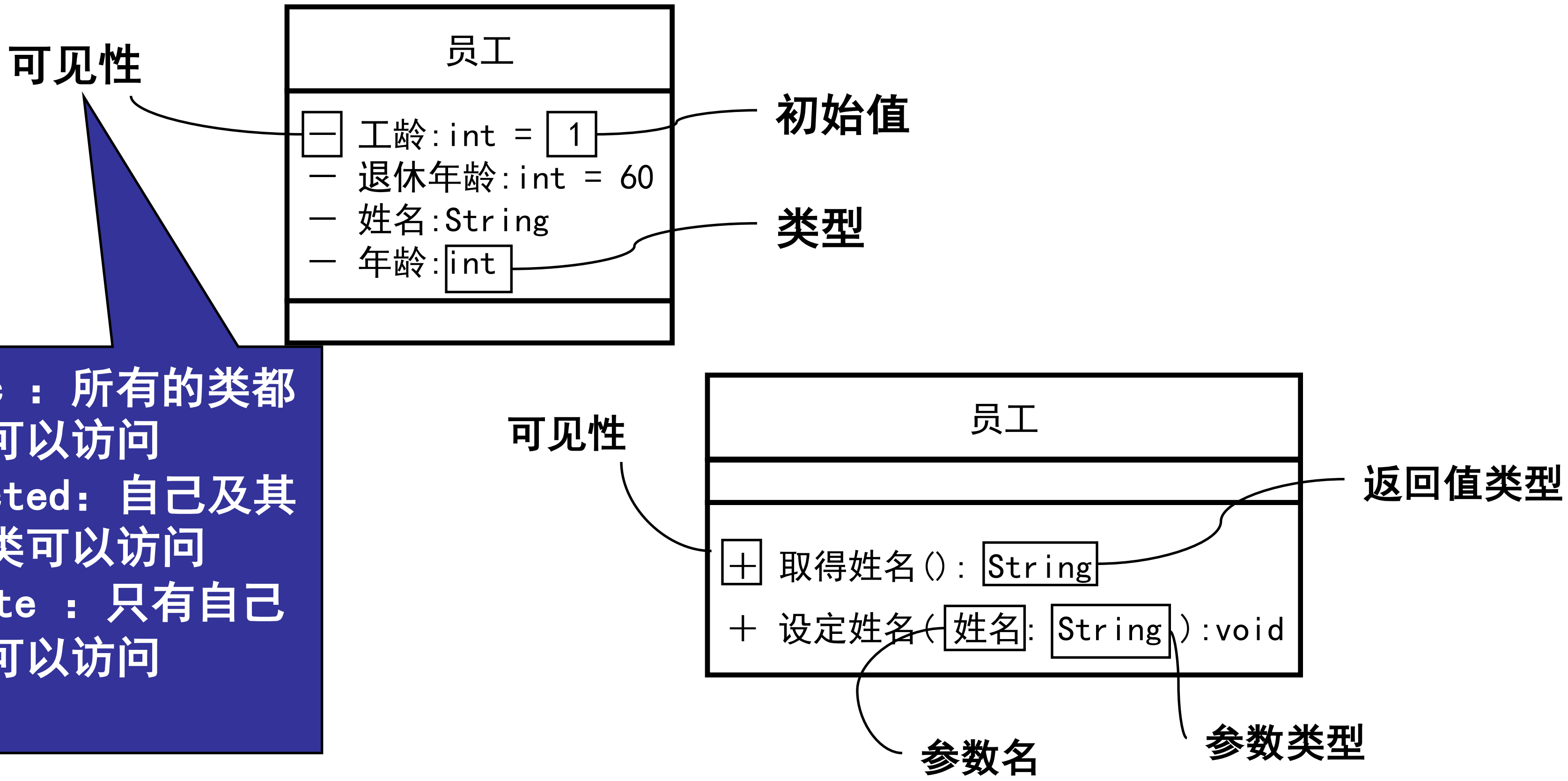
类属性的语法为：

可见性 属性名：类型=缺省值{约束特性}

类操作的语法为：

可见性　操作名（参数表）：返回类型{约束特性}

类的表示实例



可见性

员工

－ 工龄:int = 1

－ 退休年龄:int = 60

－ 姓名:String

－ 年龄:int

初始值

类型

+public ：所有的类都可以访问

#protected：自己及其子类可以访问

-private ：只有自己可以访问

可见性

员工

＋ 取得姓名():String

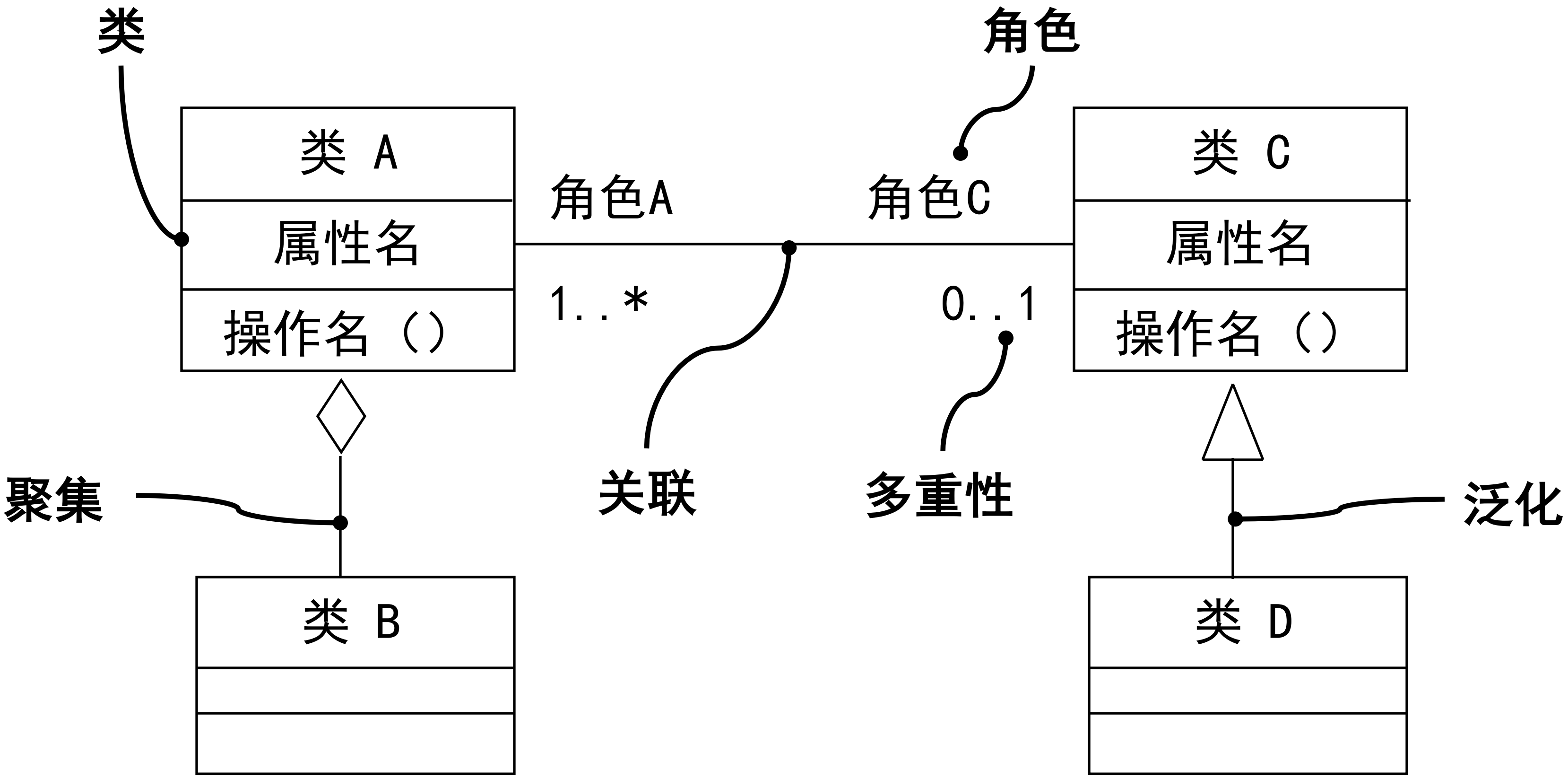
＋ 设定姓名(姓名:String):void

返回值类型

参数名

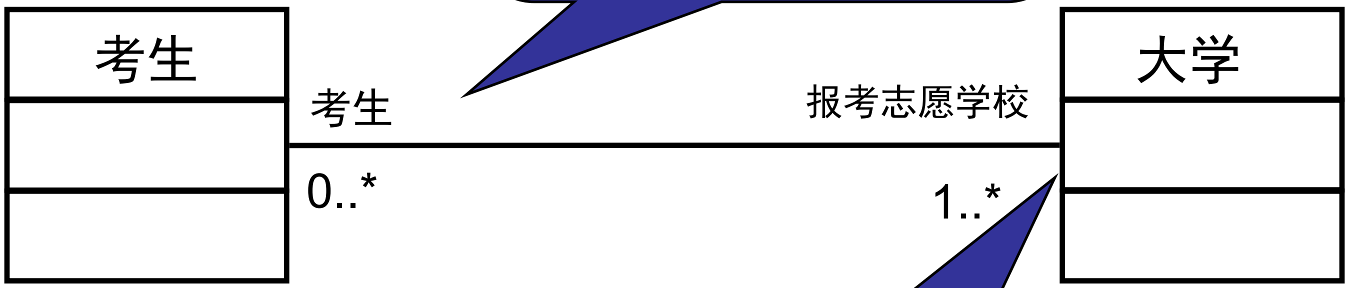
参数类型

类图的模型元素



角色和多重性

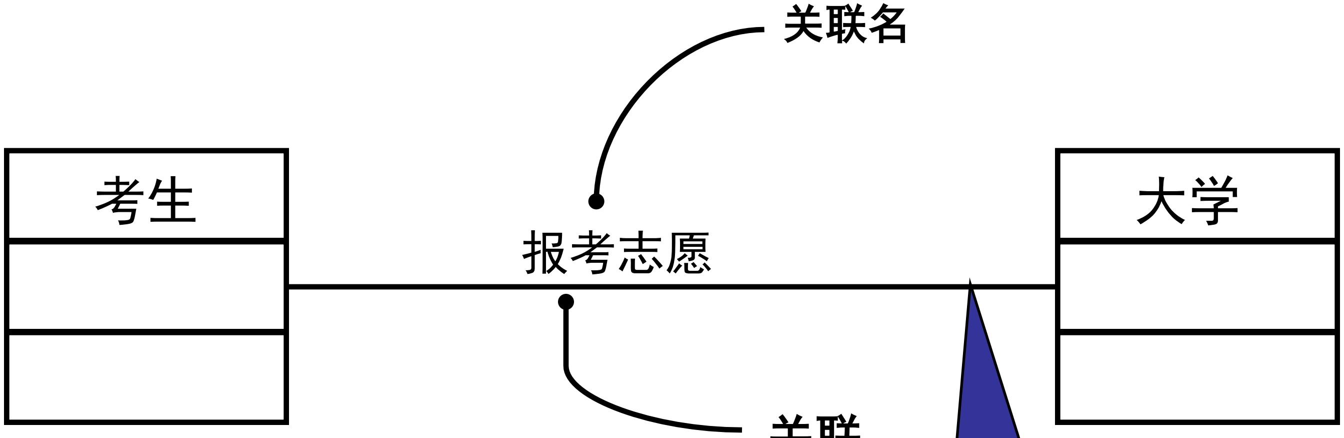
角色名，表示一个类对于对方类来说自己扮演的角色或所处的位置



多重性，表示相关联的类之间，对象数目的相应关系

关联

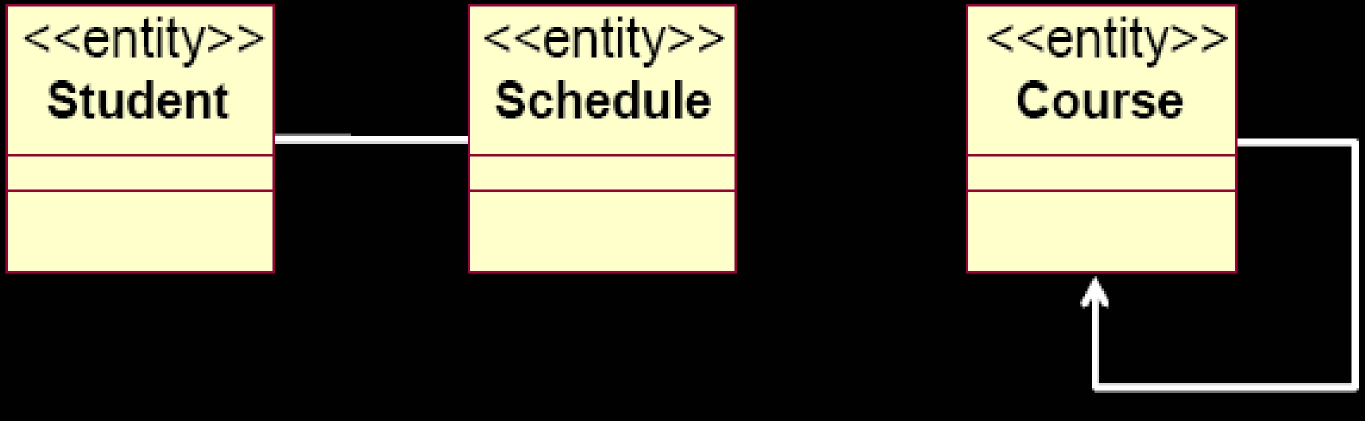
- 使用关联表示对象之间具有永久关系



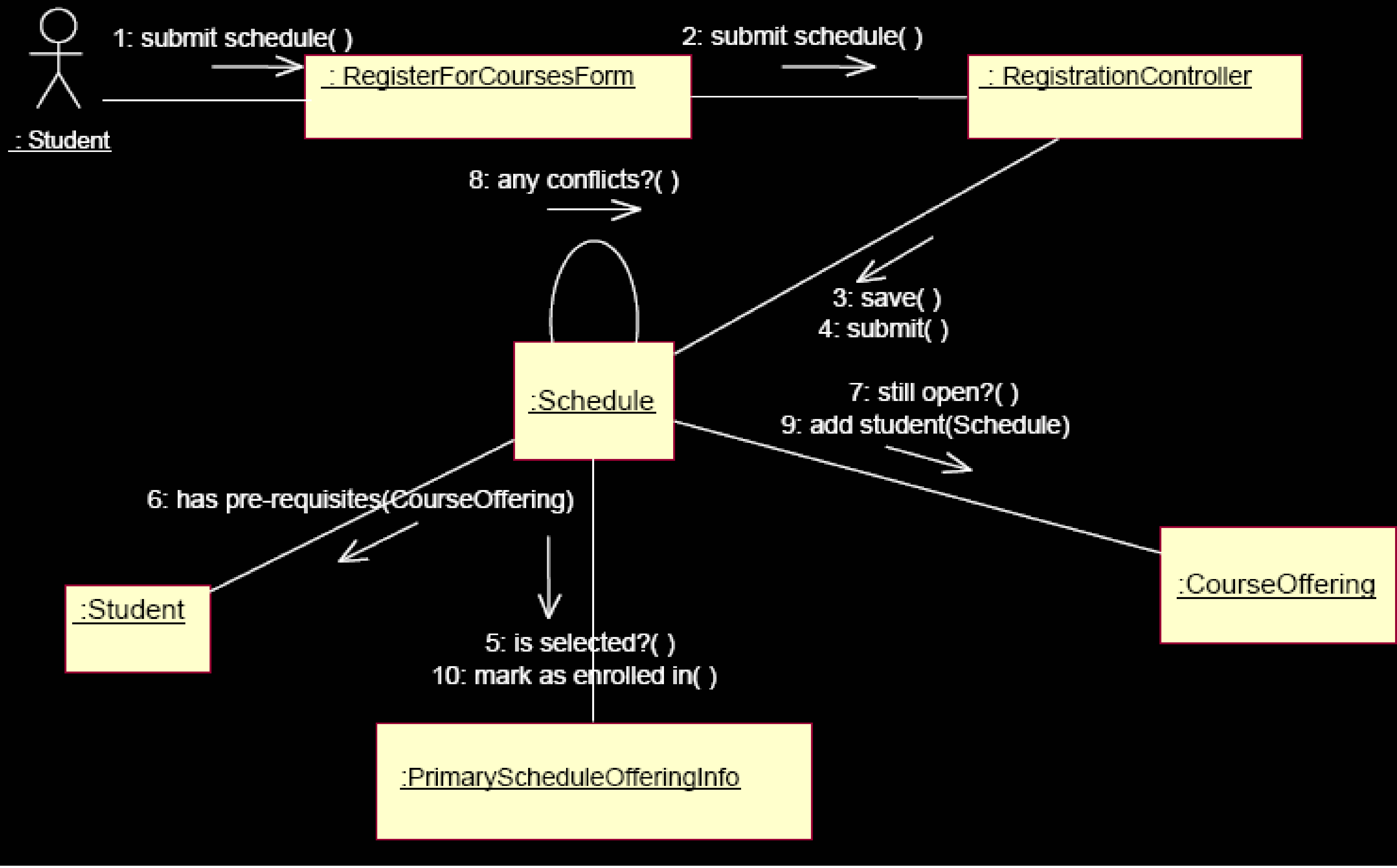
可以用箭头表示关联的方向。没有箭头表示双向关联

What Is an Association?

- The semantic relationship between two or more classifiers that specifies connections among their instances.
- A structural relationship specifying that objects of one thing are connected to objects of another thing.

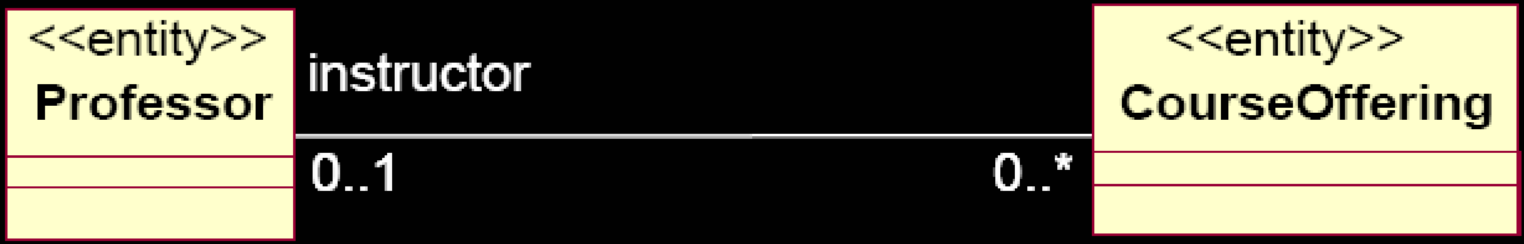


Example: What Associations Can You Find?



What Is Multiplicity?

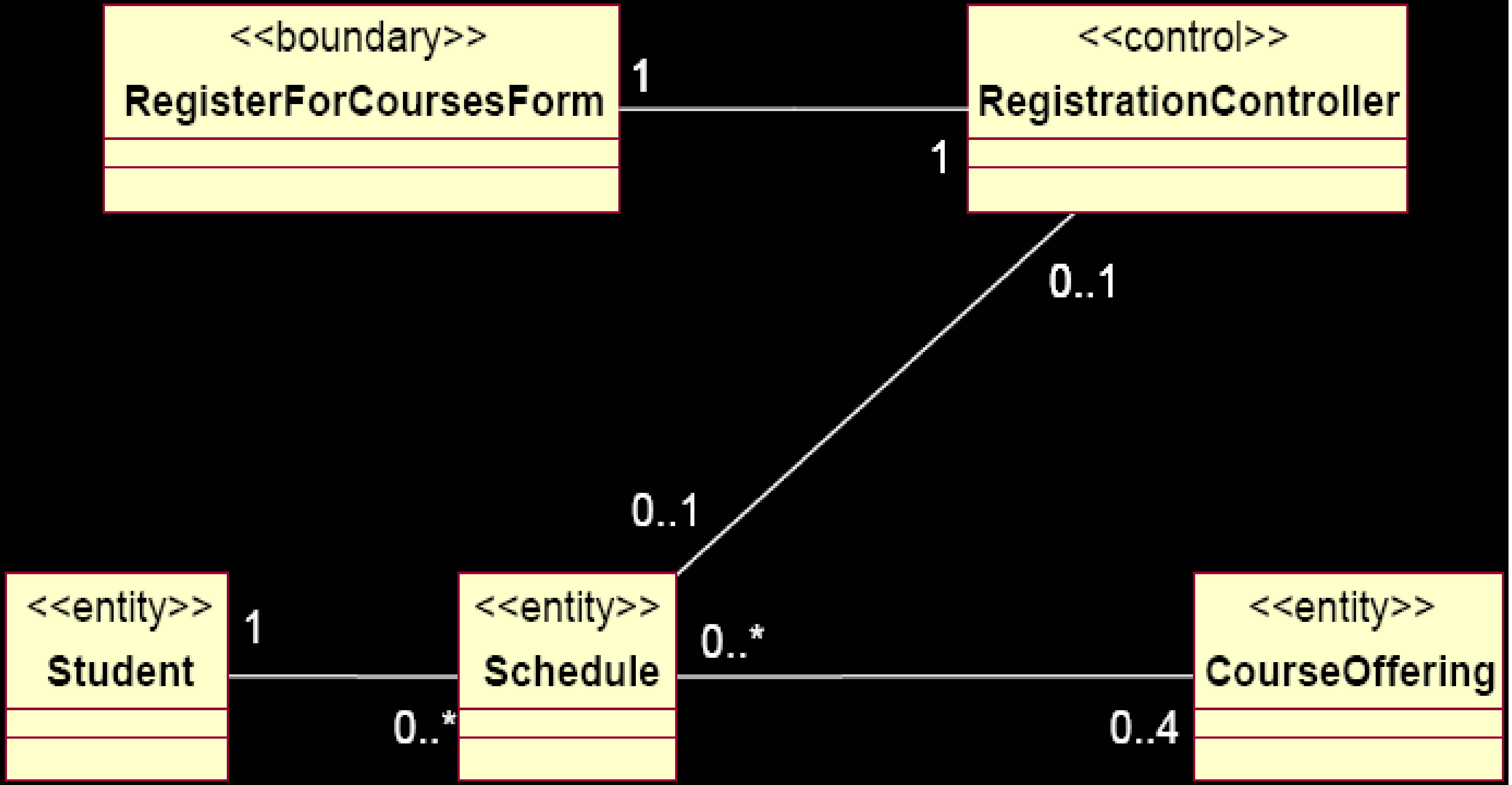
- Multiplicity is the number of instances one class relates to ONE instance of another class.
- For each association, there are two multiplicity decisions to make, one for each end of the association.
  - For each instance of Professor, many Course Offerings may be taught.
  - For each instance of Course Offering, there may be either one or zero Professor as the instructor.



Multiplicity Indicators

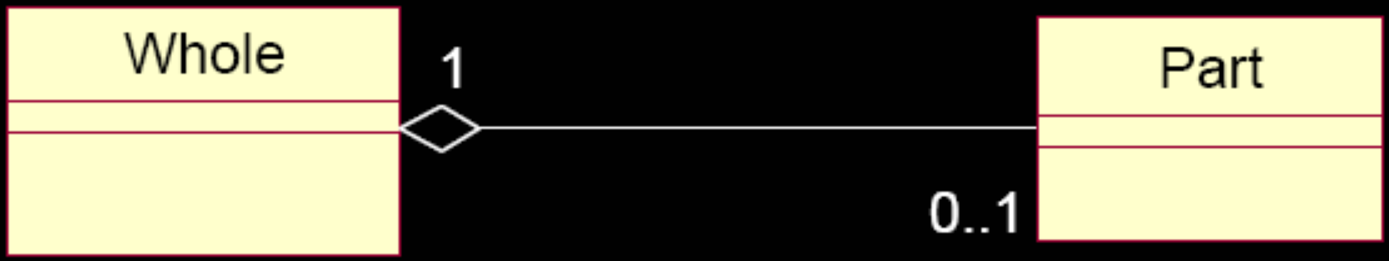
Unspecified	
Exactly One	1
Zero or More	0..*
Zero or More	*
One or More	1..*
Zero or One (optional scalar role)	0..1
Specified Range	2..4
Multiple, Disjoint Ranges	2, 4..6

Example: Multiplicity

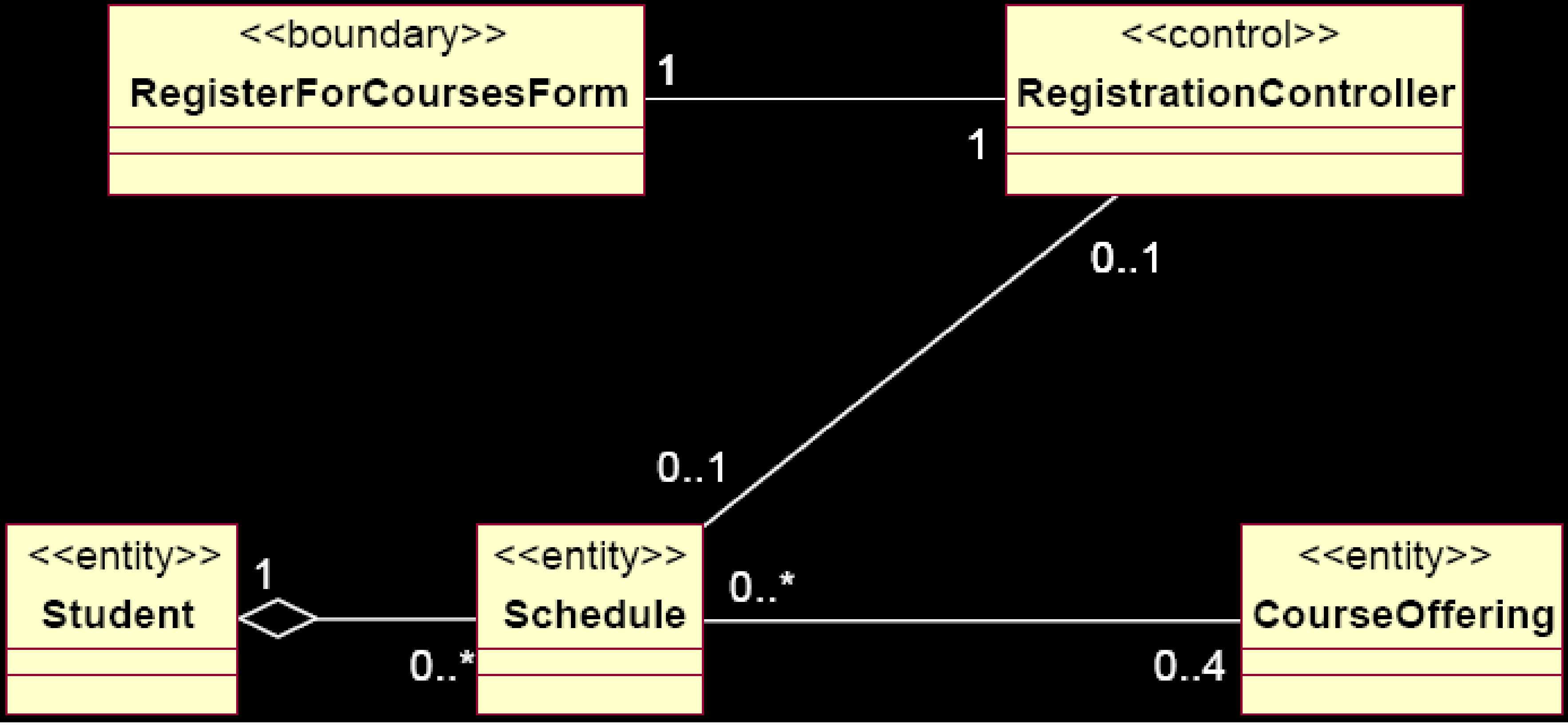


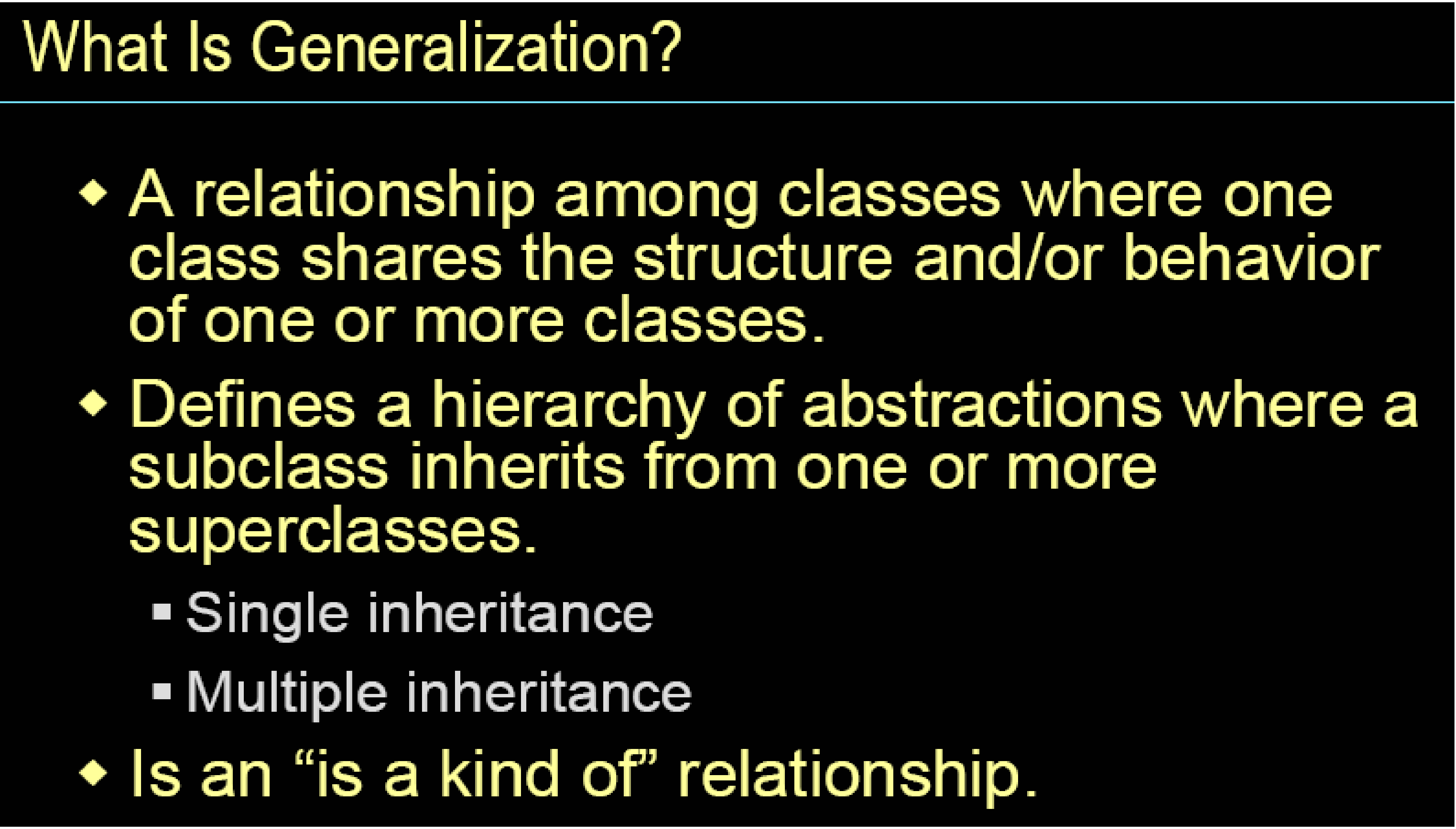
What Is an Aggregation?

- A special form of association that models a whole-part relationship between the aggregate (the whole) and its parts.
  - An aggregation is an "is a part-of" relationship.
- Multiplicity is represented like other associations.



Example: Aggregation



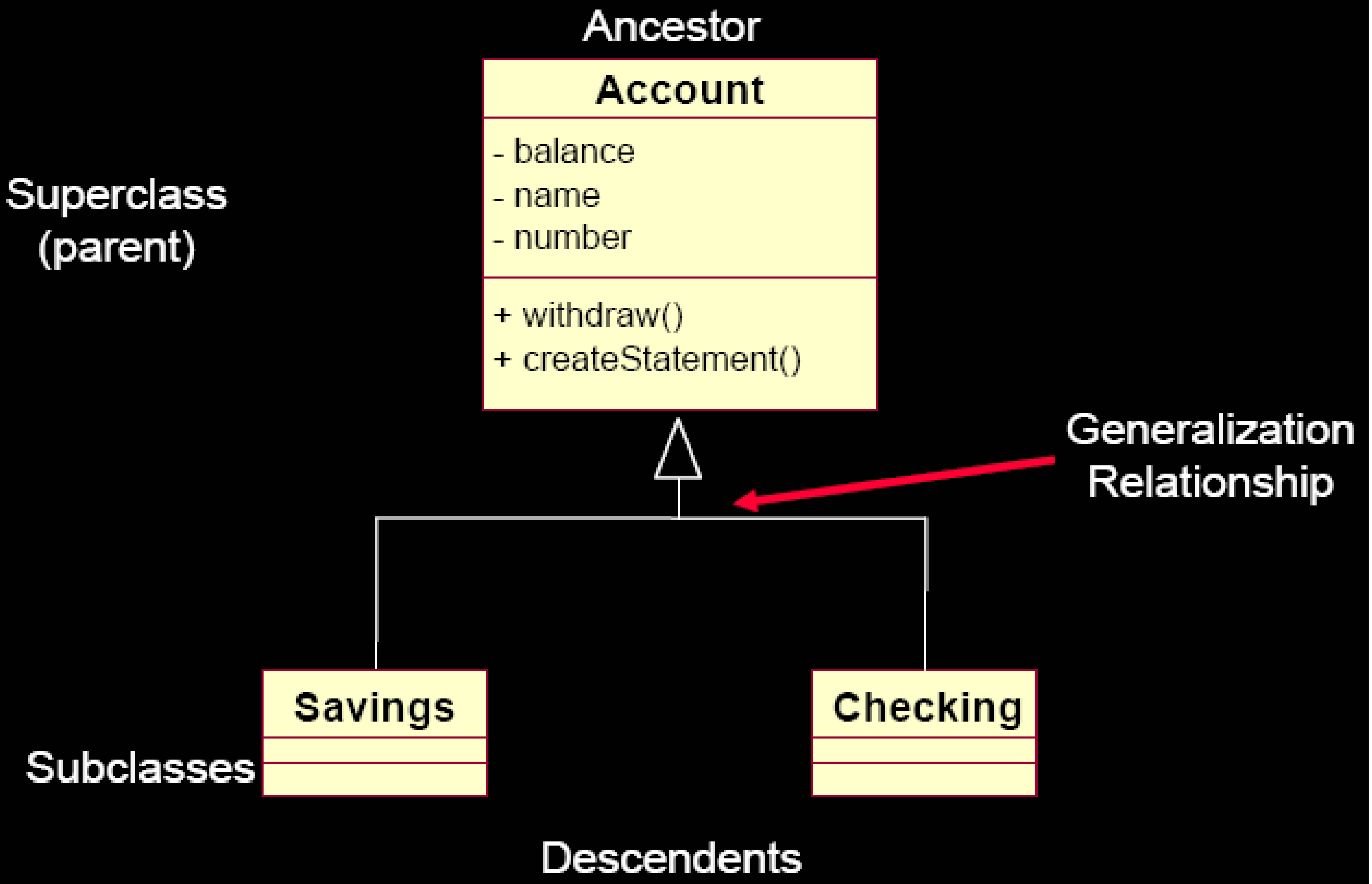


What Is Generalization?

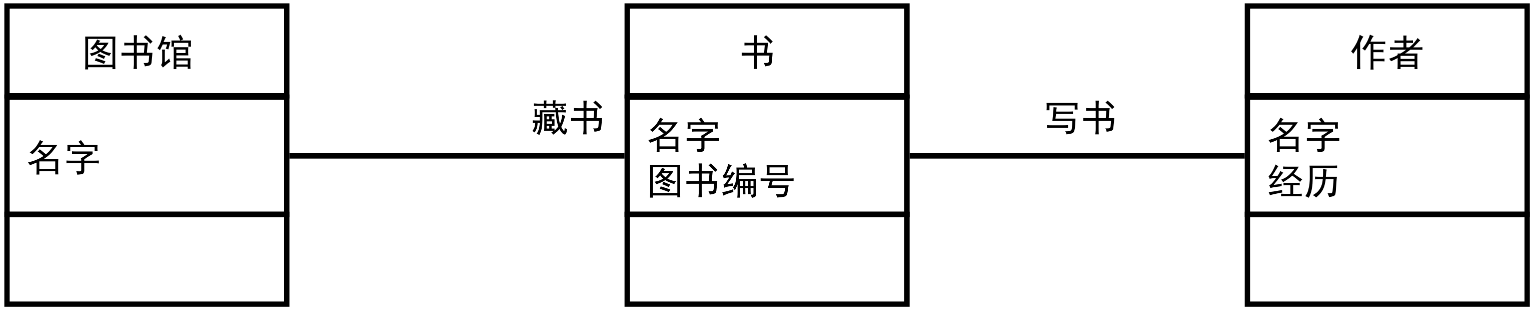
- A relationship among classes where one class shares the structure and/or behavior of one or more classes.
- Defines a hierarchy of abstractions where a subclass inherits from one or more superclasses.
  - Single inheritance
  - Multiple inheritance
- Is an “is a kind of” relationship.

Example: Single Inheritance

- One class inherits from another.

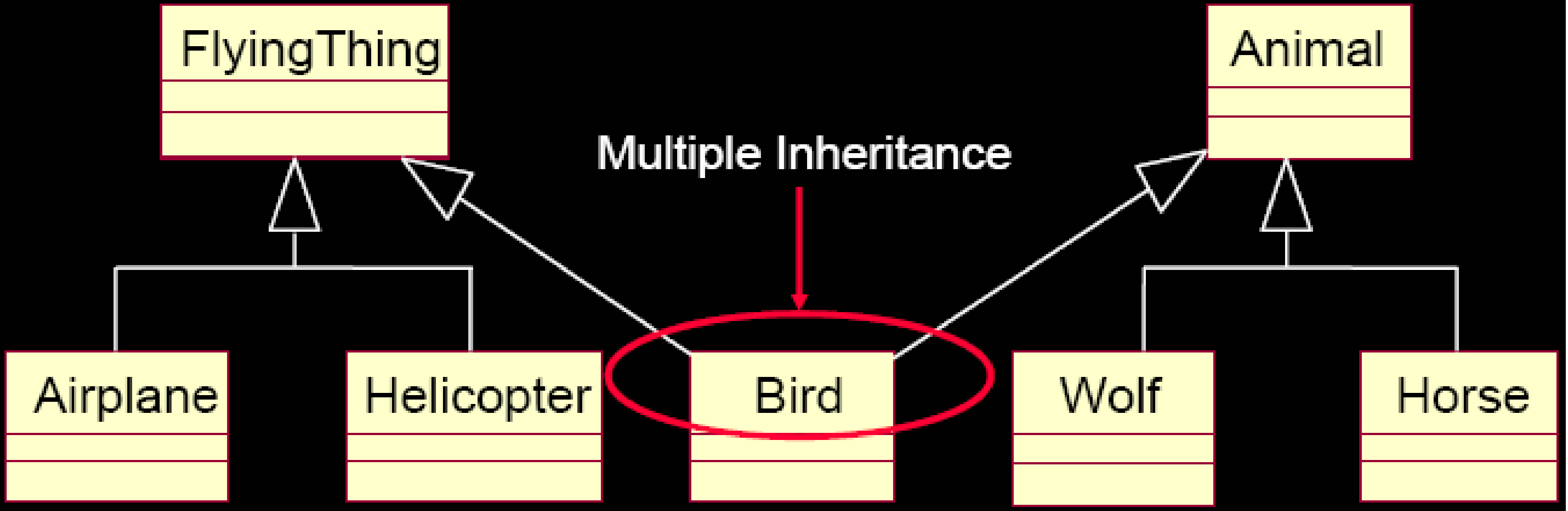


例题：试画类图（题解）



Example: Multiple Inheritance

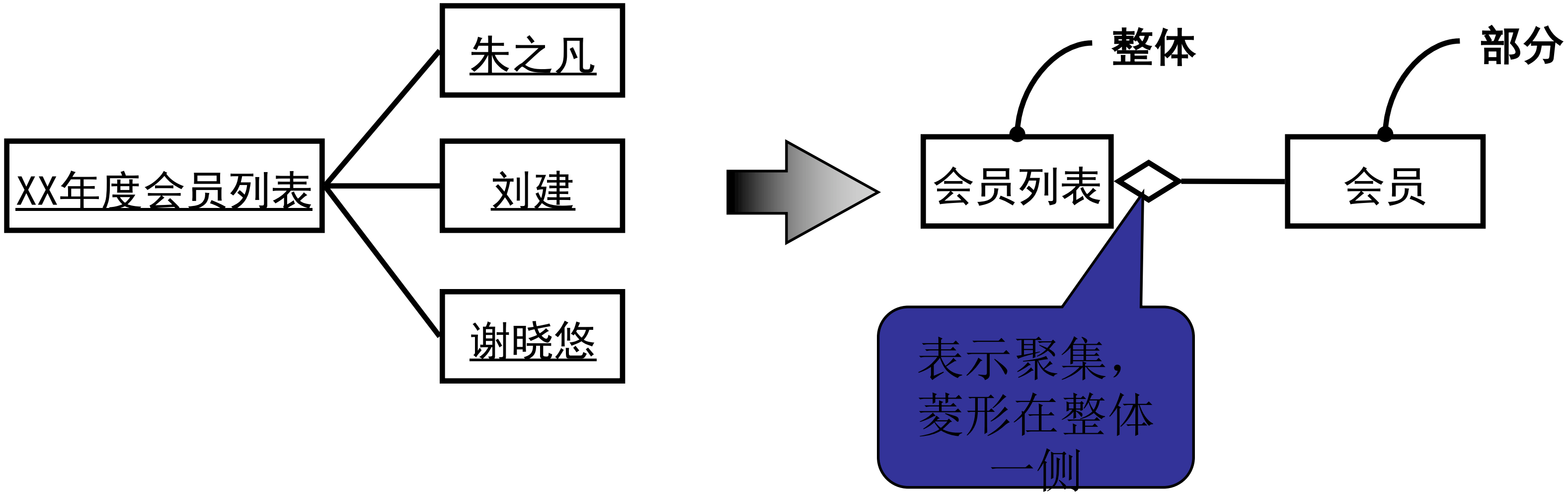
- A class can inherit from several other classes.



Use multiple inheritance only when needed and always with caution!

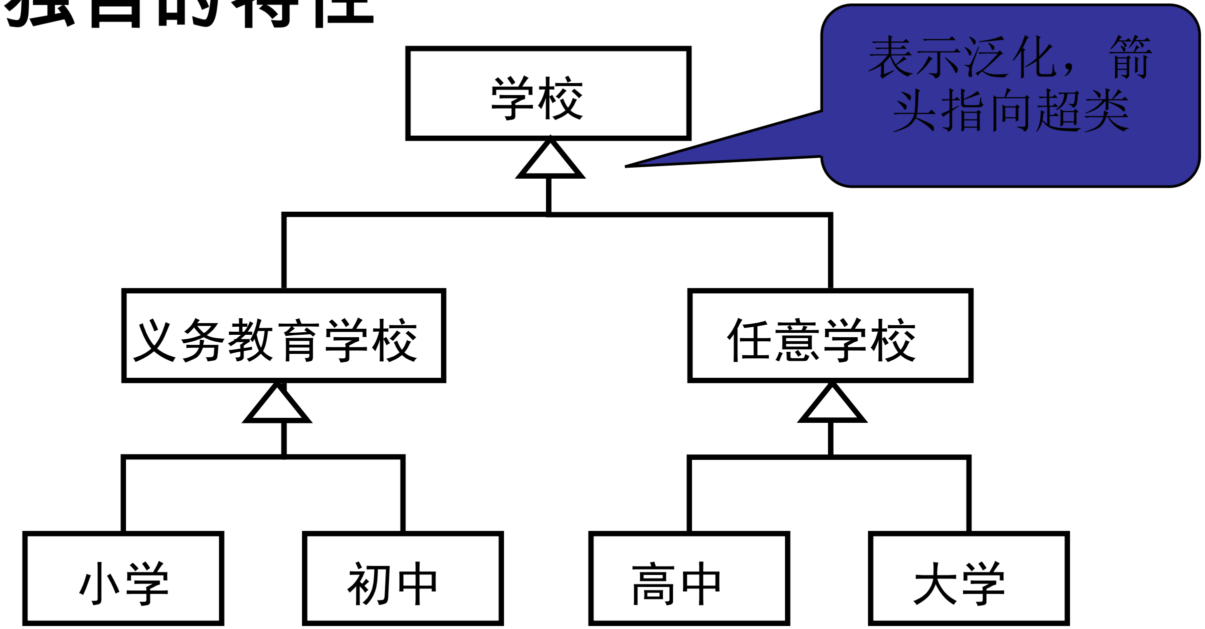
聚集

- 当某一个类成为另一类的一部分时，可使用聚集的关系
  - 表示类之间的关系是整体与部分的关系



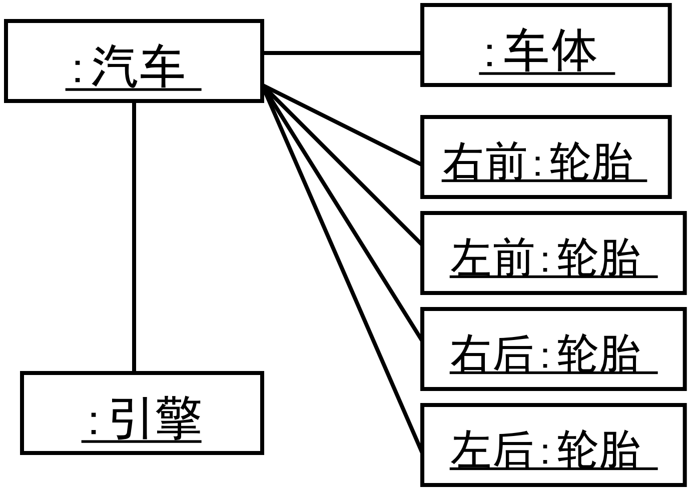
泛化

- UML将继承描述为泛化关系
- 子类继承超类的所有特性（属性、操作、关系），并具有独自的特性



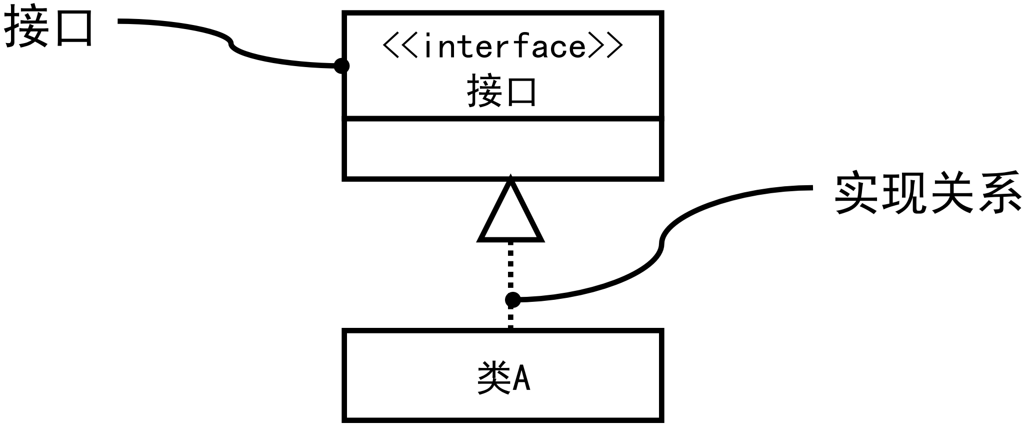
对象图

- 对象图
  - 类图的一种变形
  - 在对象名下面要加下划线
  - 所使用的符号与类图基本相同



接口

- 只具有操作的功能，不具有属性，关联，操作的实现
- 和类一样用四角形来表示实例，使用名为《interface》的构造型
- 用实现关系（带空心白色三角的虚线）符号来连接实现接口的元素（类，构件等）



作业(2017.03.07)

1. What does a class diagram represent?
2. What benefits do packages provide to the model?(此题不必写了，答：包是一种成组的机制，其作用相当于文件夹，当很多制品，譬如类、对象、构件、包，需要组织和管理时，可以利用包（package）进行分组、对制品进行抽象化表达、构建架构等)
3. How to define association, aggregation, and generalization
4. How to find and describe associations?
5. What is multiplicity? What information does multiplicity provide to the modeler?
6. What is difference between sequence diagram vs collaboration diagram?(本次不做）
7. 画一个类图，其中包括若干类及对象，练习如何描述关联、继承、聚合、多重性等关系，并自己在UML编辑软件（Rational Rose ）上练习
8. 预习 时序图、协作图，熟悉图形元素，分析两种图相同和不同点

Essentials of Visual Modeling with UML

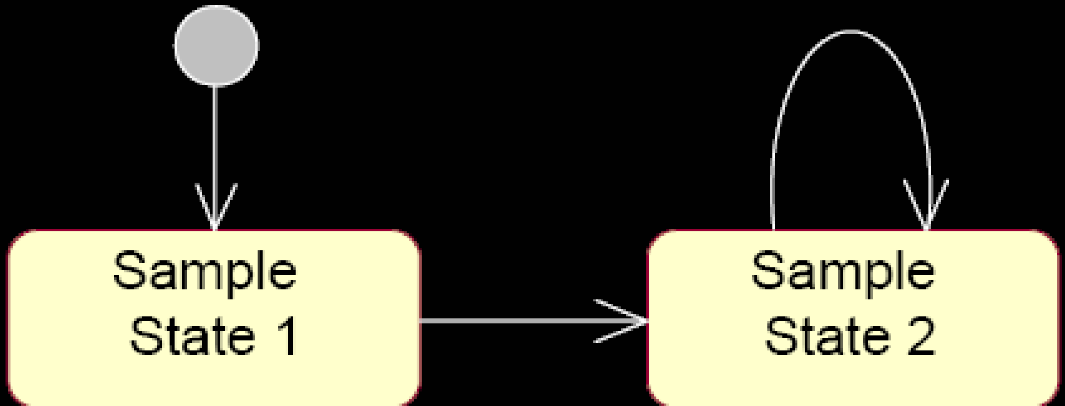
UML Other Diagrams

agenda

- Use case model: use case diag. & activity diag.
- Interaction diagrams：sequence diag. & collaboration diag.
- Class model: class diag.
- UML other diagrams: statechart diag., deployment diag., component diag., package diag.

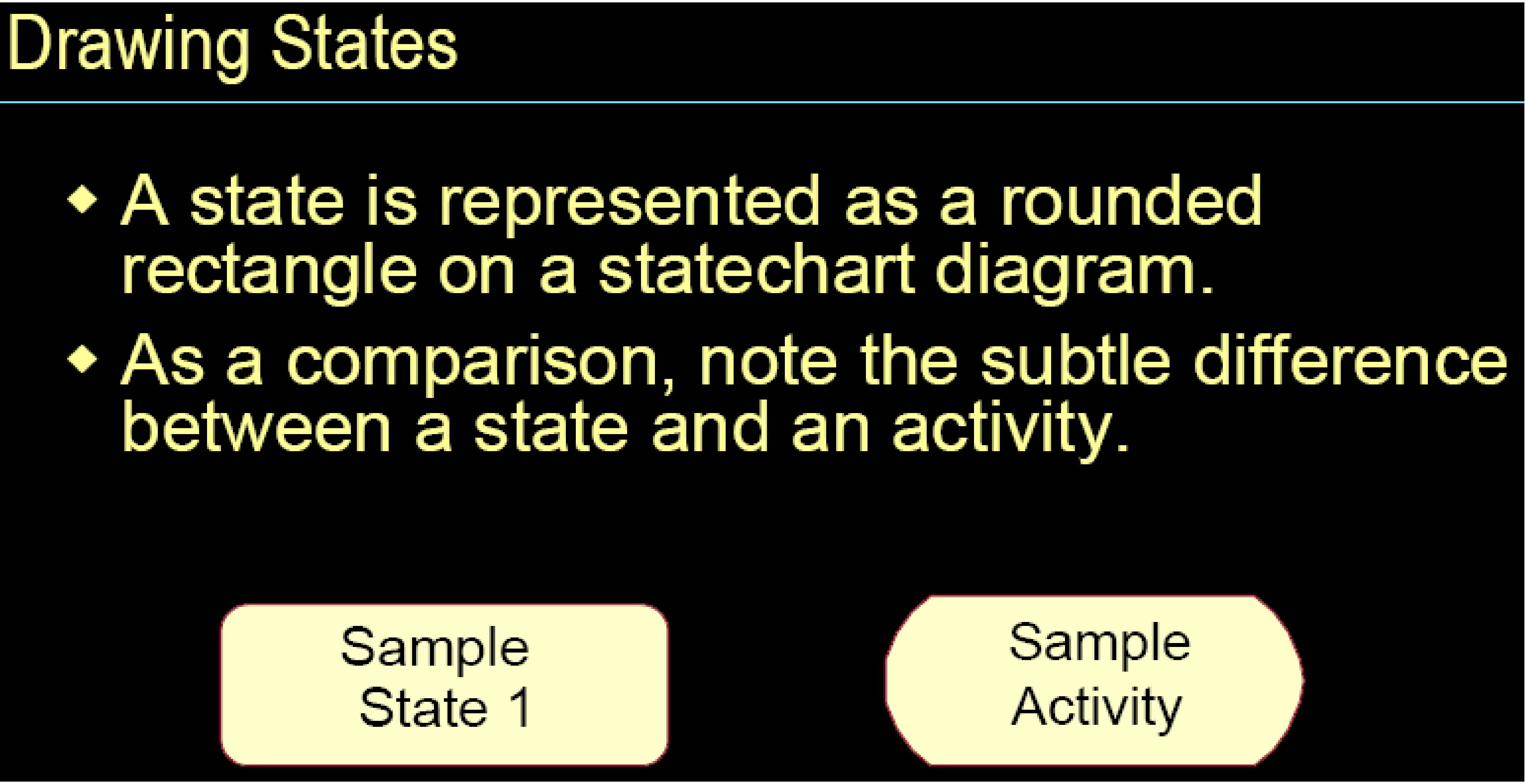
Statechart Diagrams

- A statechart diagram shows a state machine.
- It specifies the sequence of states that an object can be in:
  - The events and conditions that cause the object to reach those states
  - The actions that take place when those states are reached



Sample State 1

Sample State 2

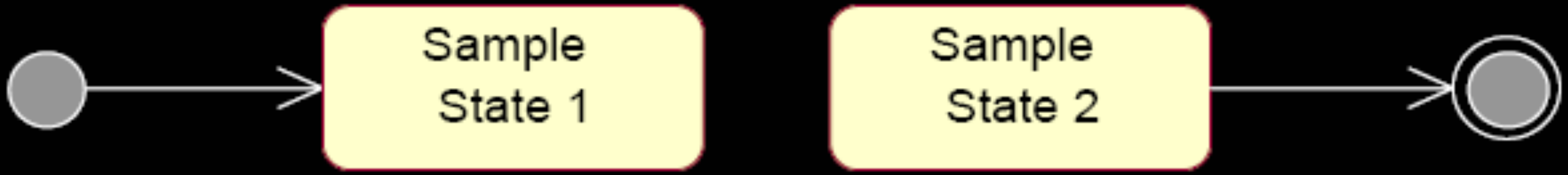


Drawing States

- A state is represented as a rounded rectangle on a statechart diagram.
- As a comparison, note the subtle difference between a state and an activity.

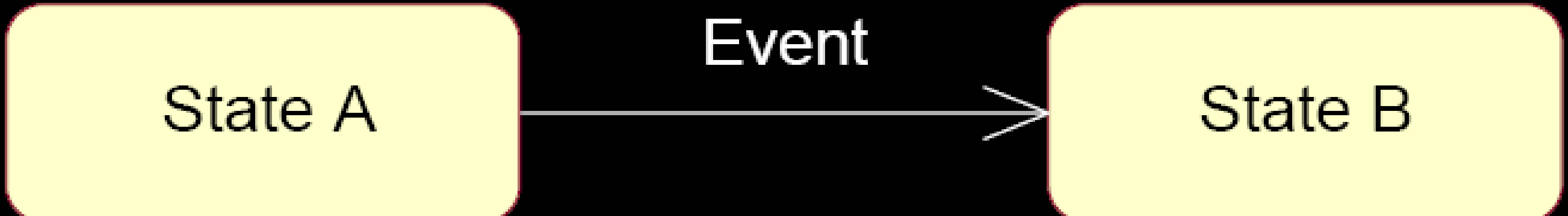
Special States

- The initial state is the state entered when an object is created.
  - An initial state is mandatory.
  - Only one initial state is permitted.
  - The initial state is represented as a solid circle.
- A final state indicates the end of life for an object.
  - A final state is optional.
  - A final state is indicated by a bull's eye.
  - More than one final state may exist.

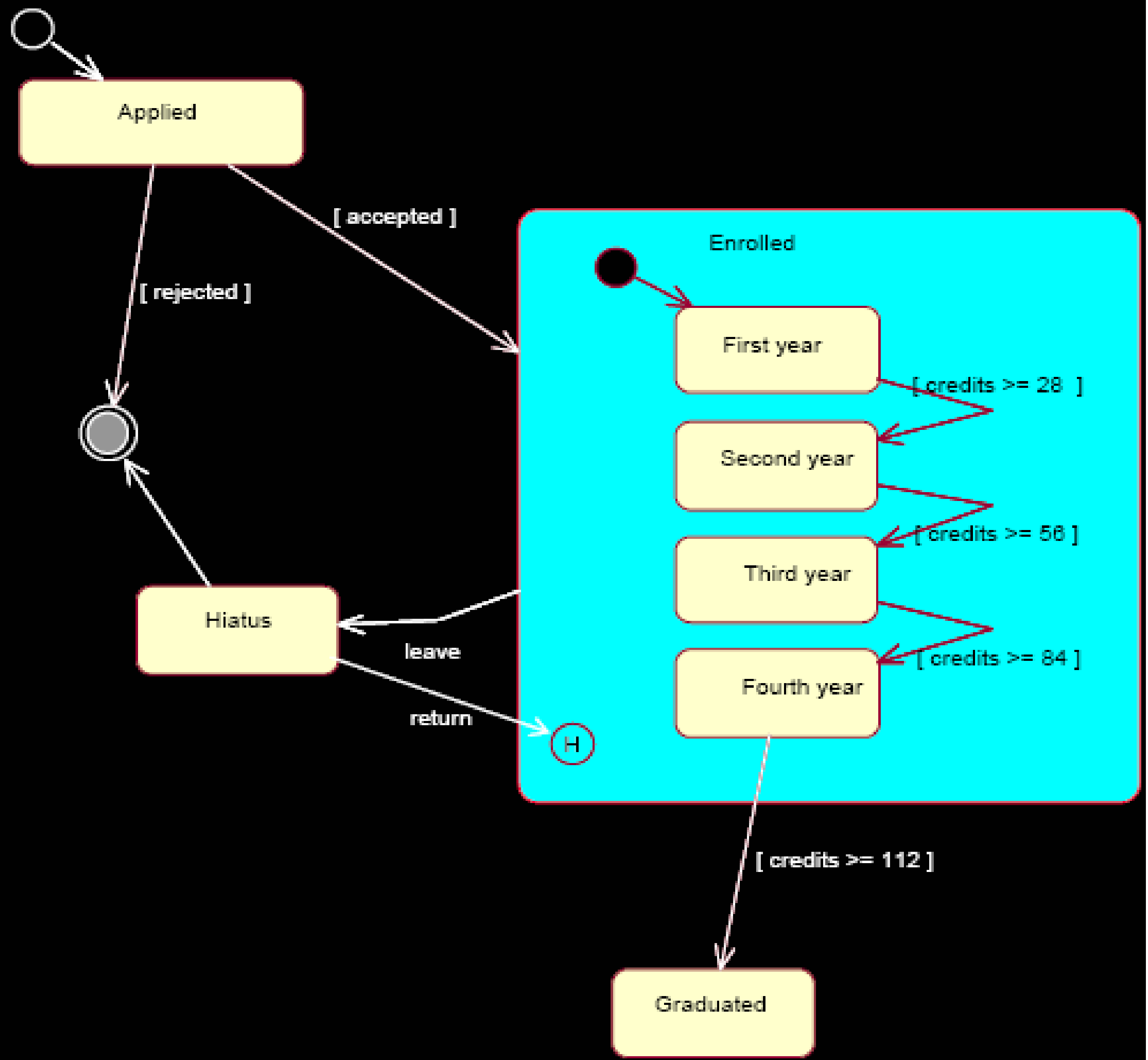


What Are Events?

- An event is the specification of a significant occurrence that has a location in time and space.
  - An event is an occurrence of a stimulus that can trigger a state transition.
  - Example:
    - Adding a student to a course
    - Creating a new course



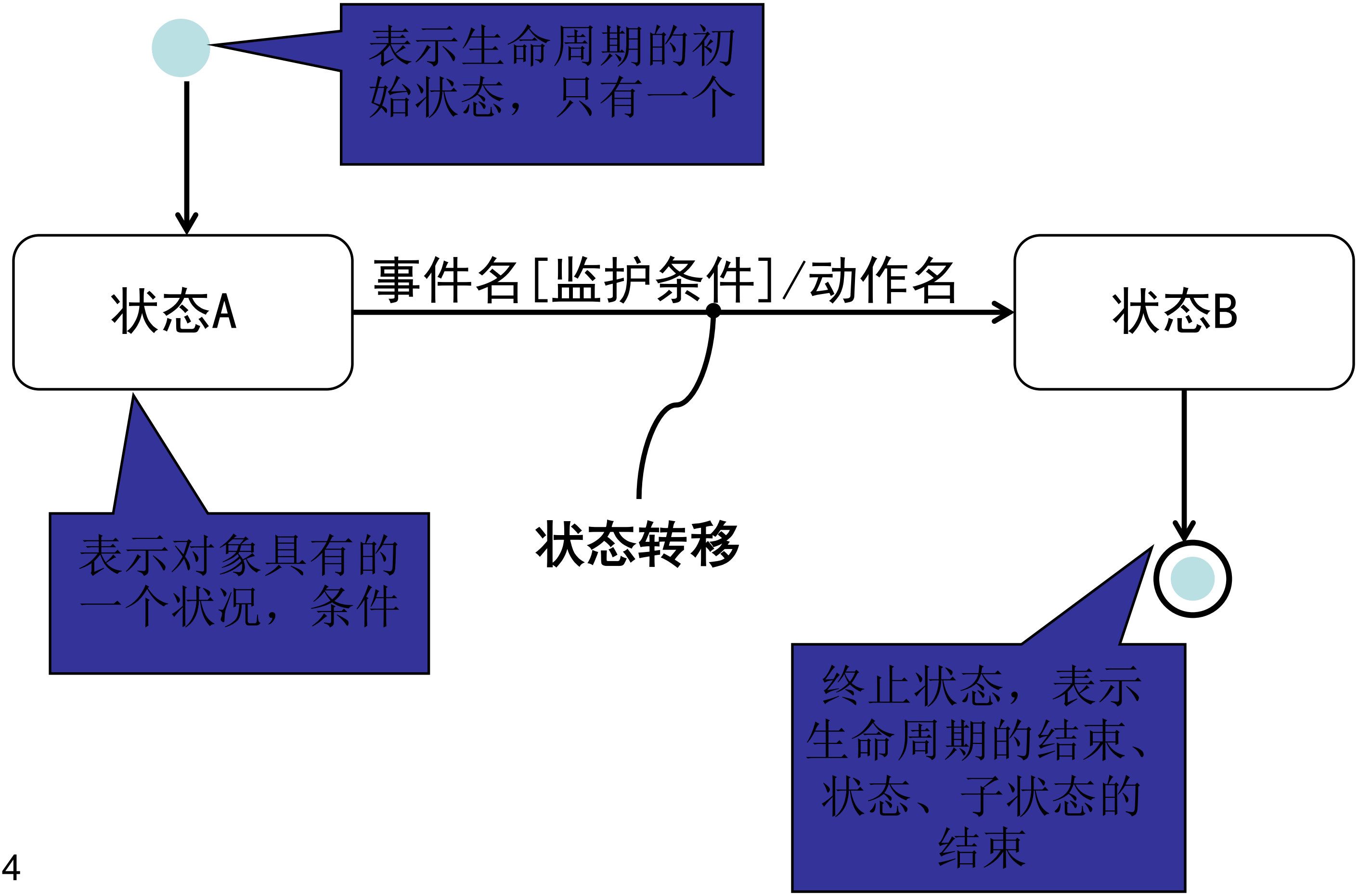
Example: Statechart



状态图

- 作用
  - 显示一个对象从创建到消亡的整个生命周期
- 状态图主要显示内容
  - 对象在生命周期所经历的状态序列
  - 诱发对象从一个状态变为另一个状态的事件
  - 状态改变所导致的动作

状态图的模型元素



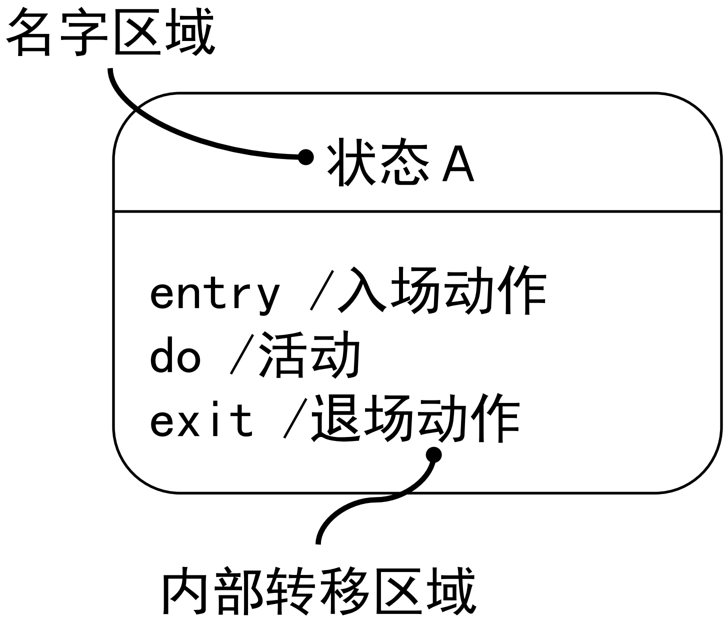
表示生命周期的初始状态，只有一个

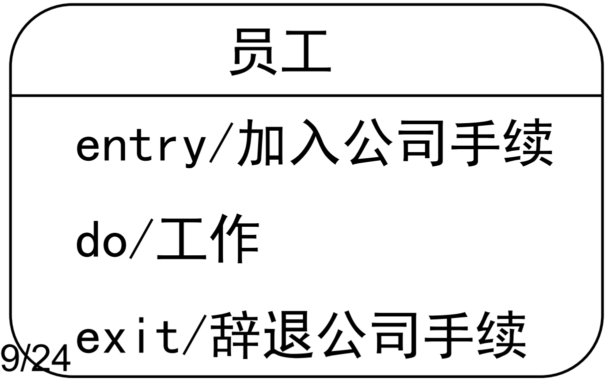
表示对象具有的一个状况，条件

状态转移

终止状态，表示生命周期的结束、状态、子状态的结束

状态可分成2个区域

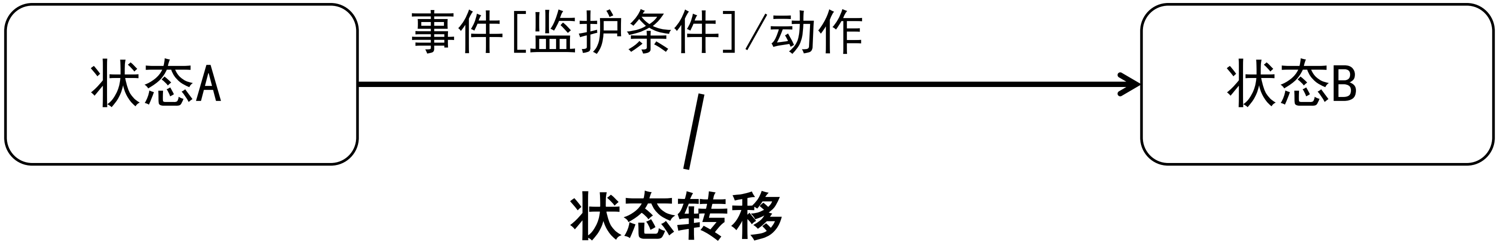




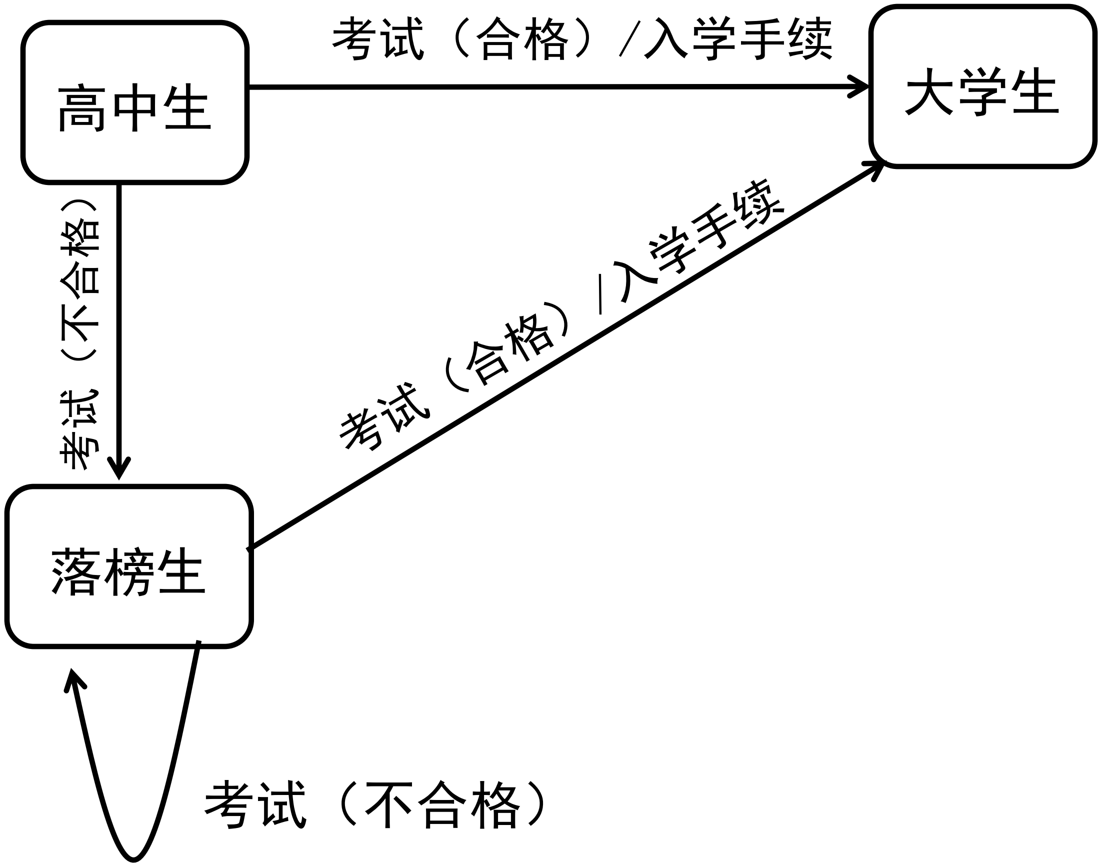
- 名字区域
  - 指定状态名字
- 内部转移区域
  - 入场动作
    - 指进入该状态时所执行的动作
  - 活动
    - 指从入场到退场的时间内，或到动作结束时所要持续进行的动作
  - 退场动作
    - 指从一个状态退出时所执行的动作

状态转移

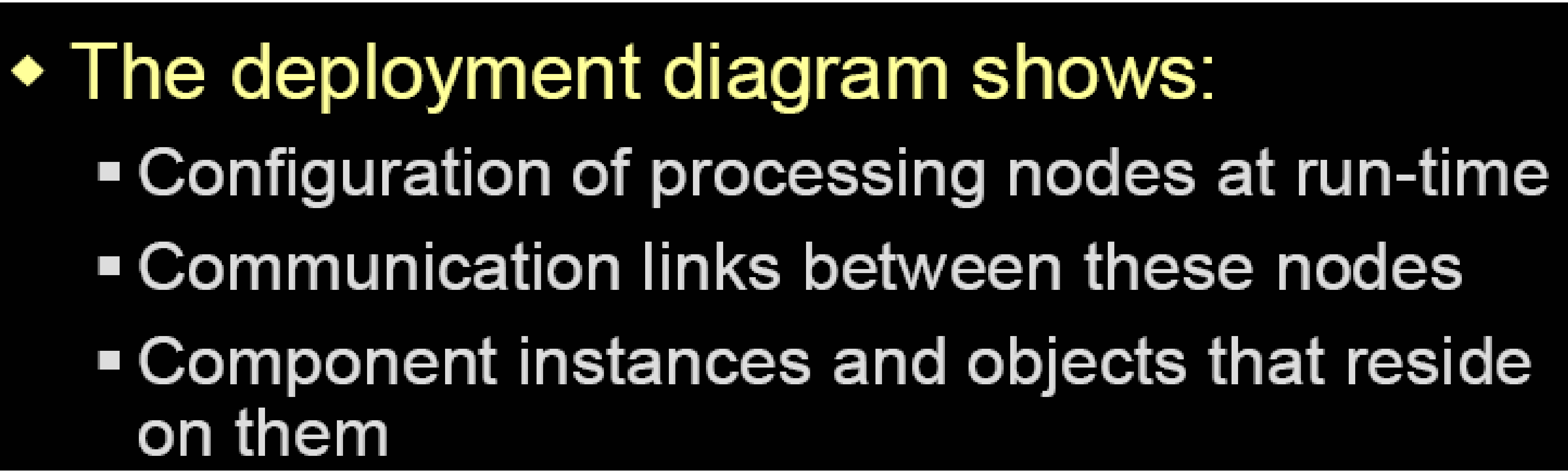
- 事件
  - 给对象带来某种影响的触发情况
- 监护条件
  - 只有该条件成立时，才发生相应的状态转移
- 动作
  - 发生转移时所执行的动作
- 以下情况会引起转移
  - 事件触发/满足监护条件



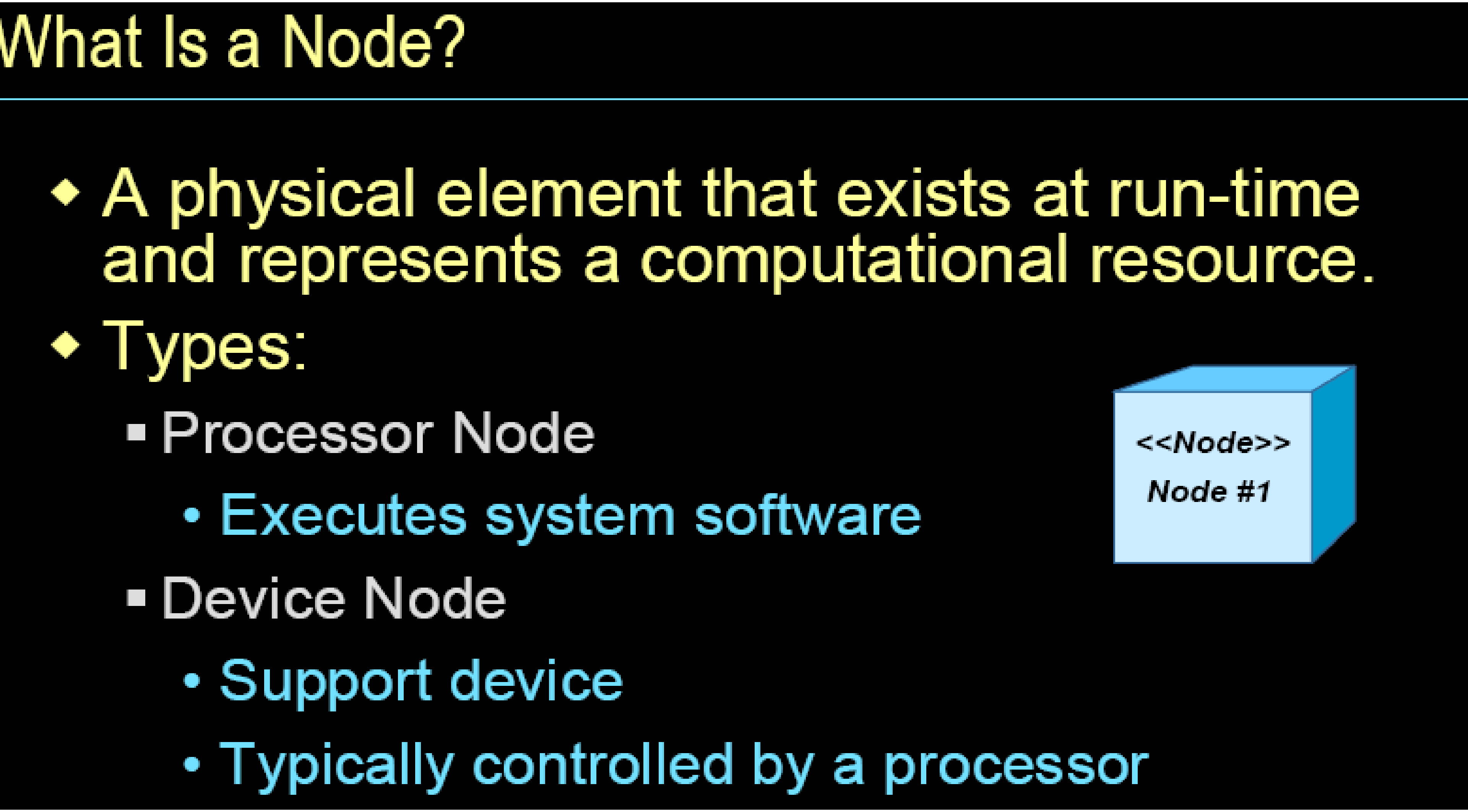
状态图示例



Deployment Diagram



- The deployment diagram shows:
  - Configuration of processing nodes at run-time
  - Communication links between these nodes
  - Component instances and objects that reside on them

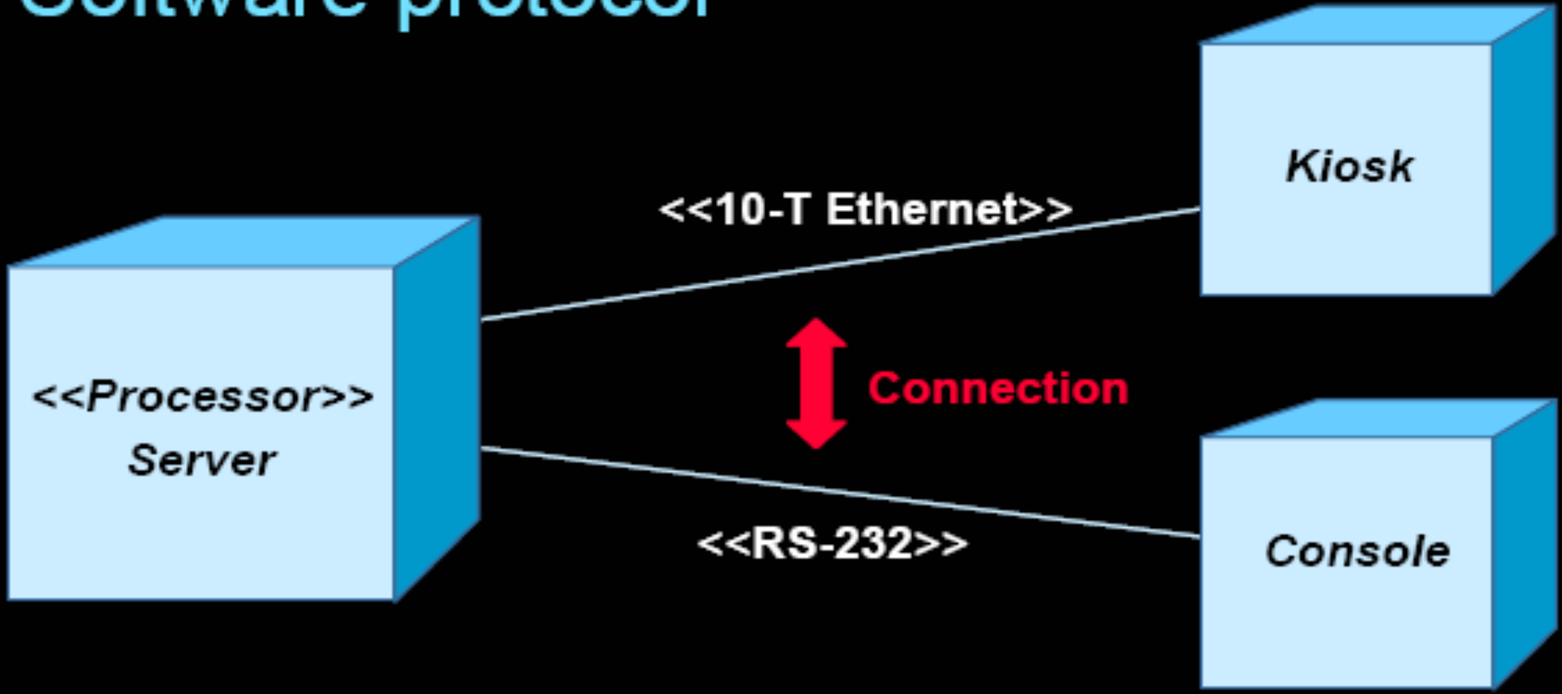


What Is a Node?

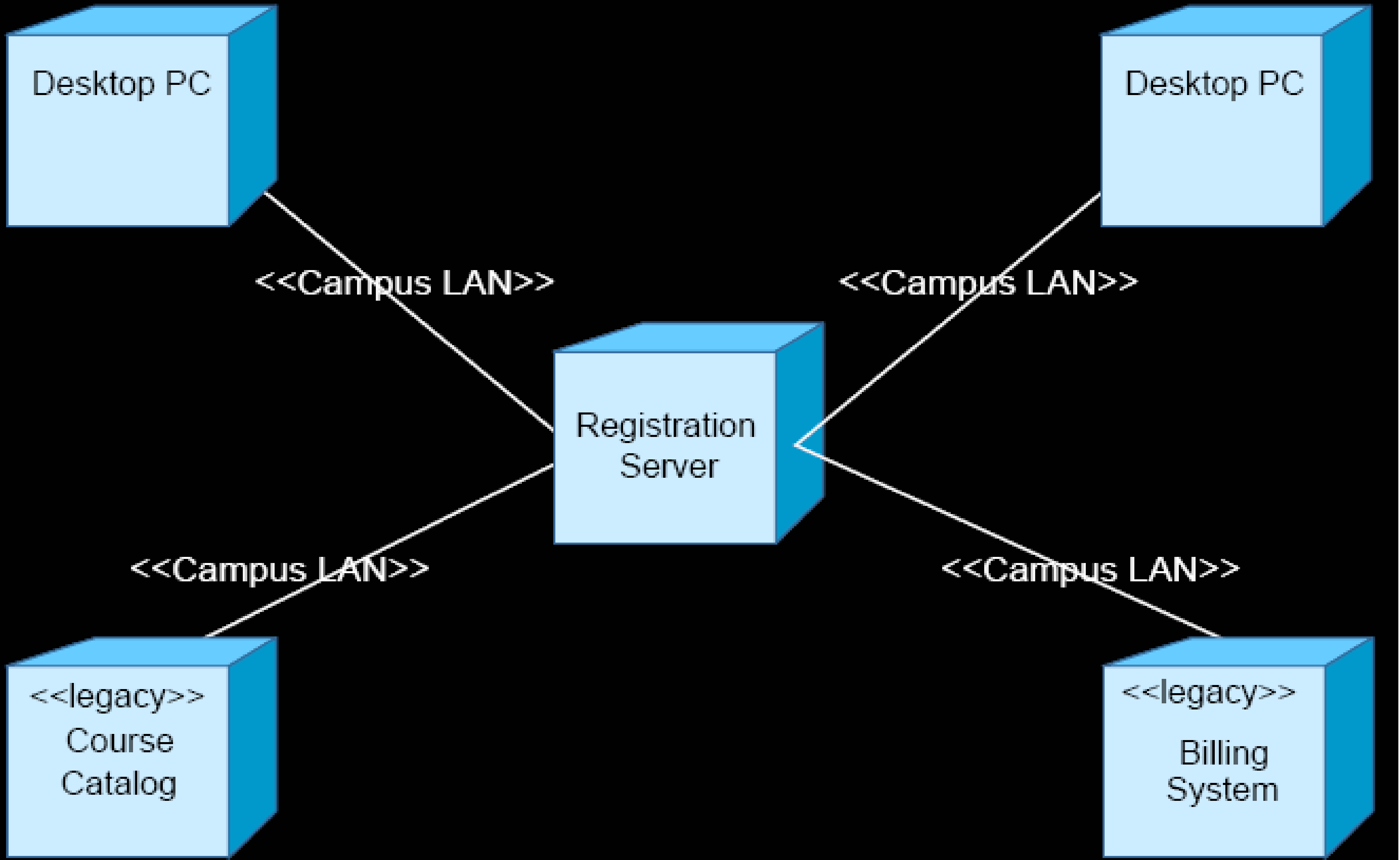
- A physical element that exists at run-time and represents a computational resource.
- Types:
  - Processor Node
    - Executes system software
  - Device Node
    - Support device
    - Typically controlled by a processor

What Is a Connection?

- A connection represents a:
  - Communication mechanism
    - Physical medium
    - Software protocol



Example: Deployment Diagram

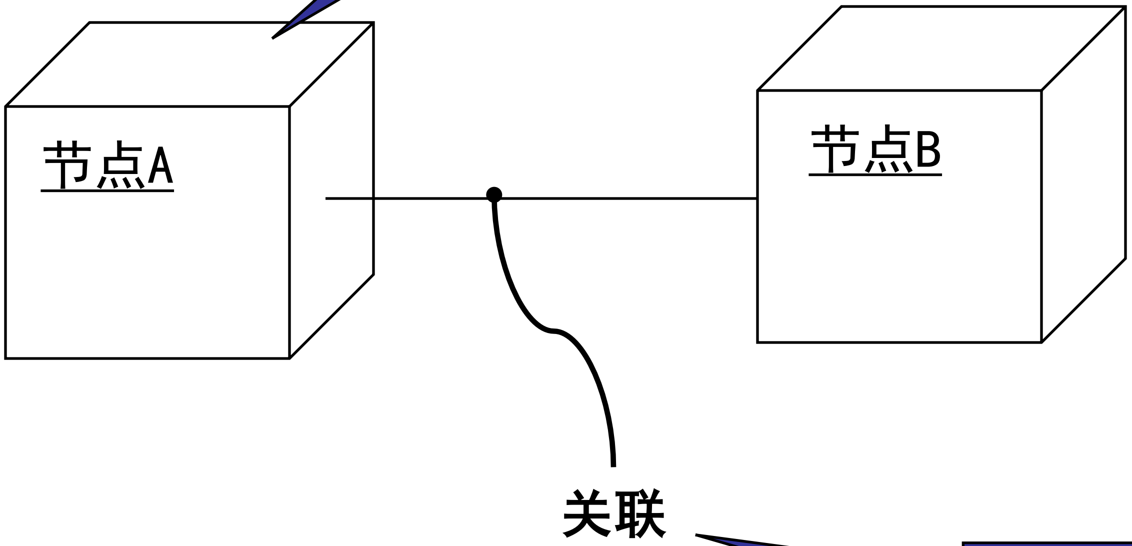


部署图

- 描述执行时的系统结构（硬件、软件）
  - 执行环境中的硬件结构和连接关系
  - 对硬件（节点）部署软件（构件）

部署图的模型元素

执行演算的资源可以是构件，进程，对象



节点A

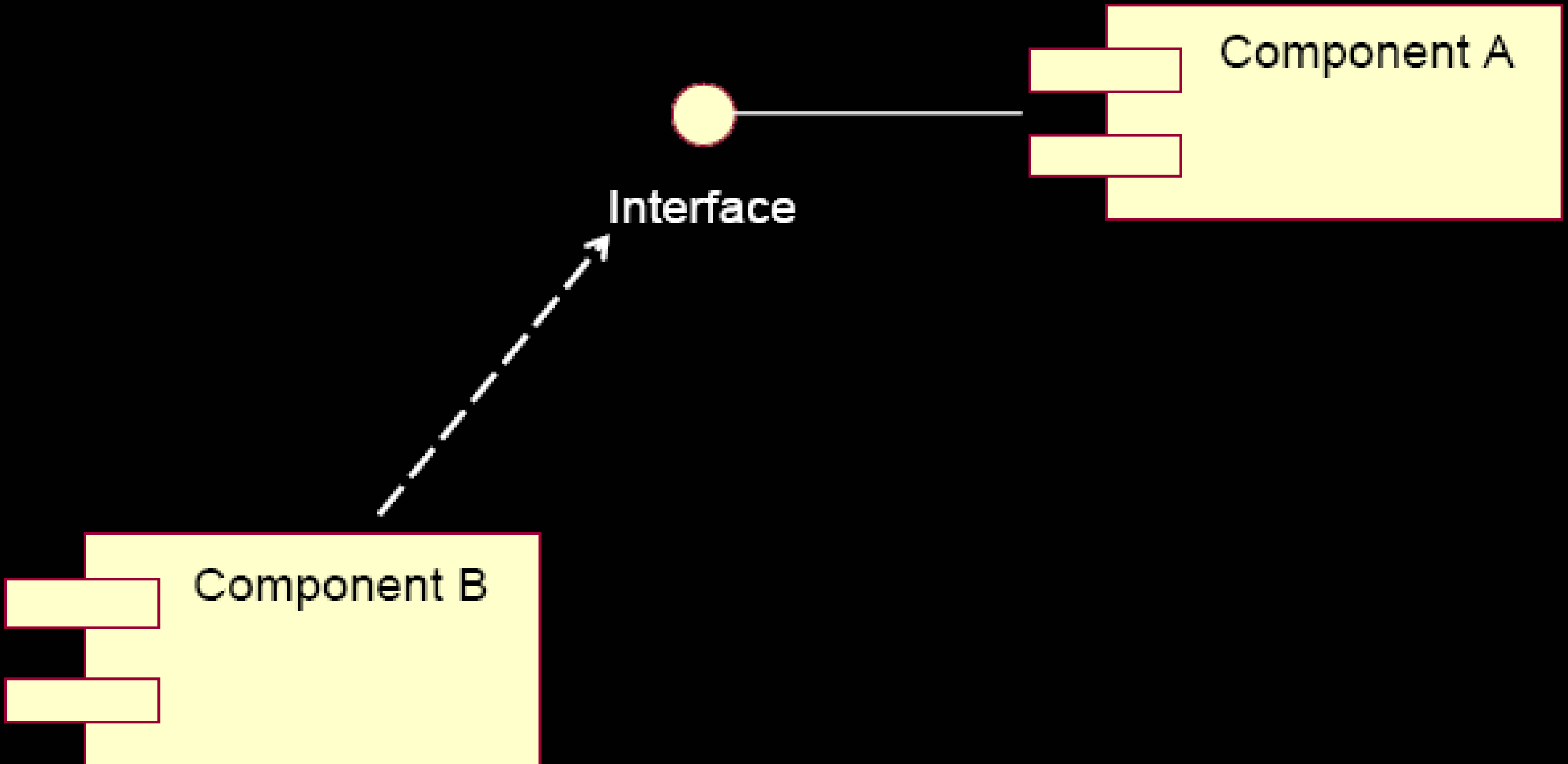
节点B

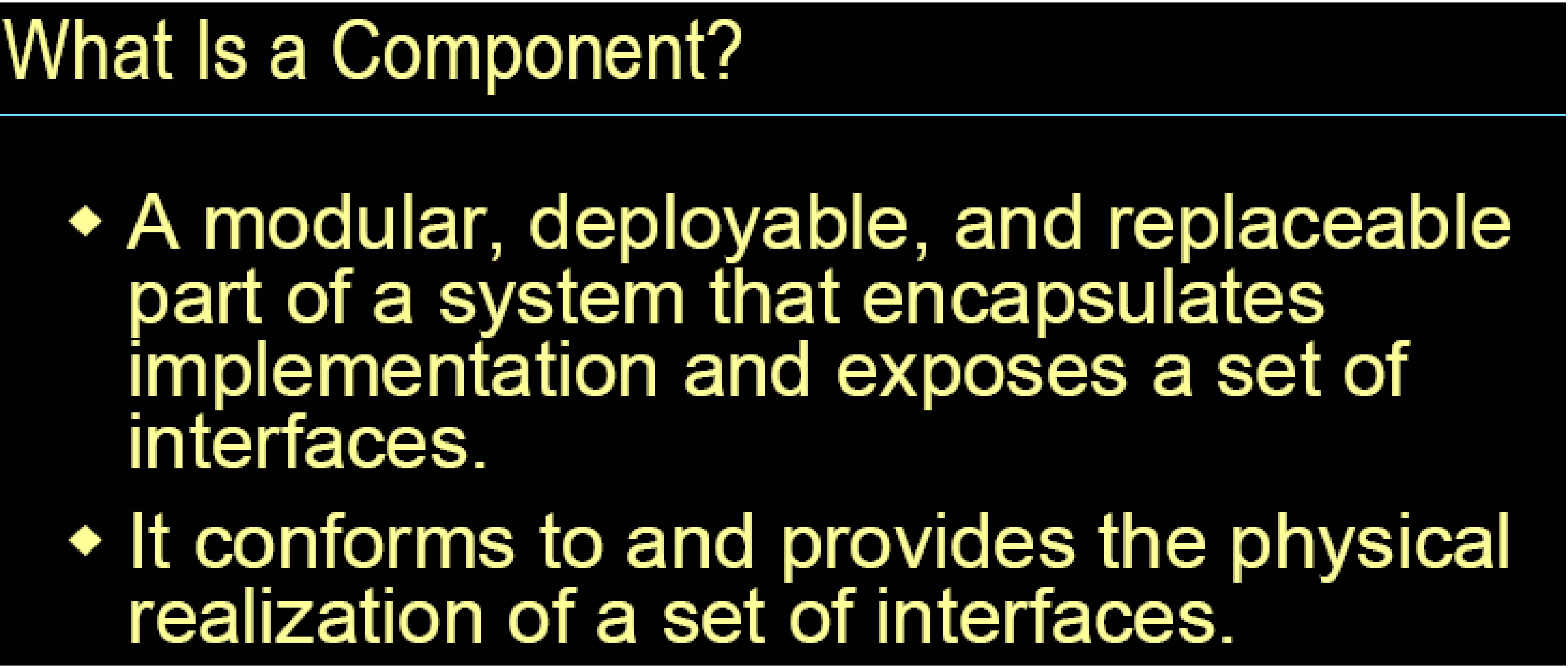
关联

表示节点之间有交流（有通信路径）

Component Diagram

- A diagram that shows the organization of and the dependencies among a set of components.





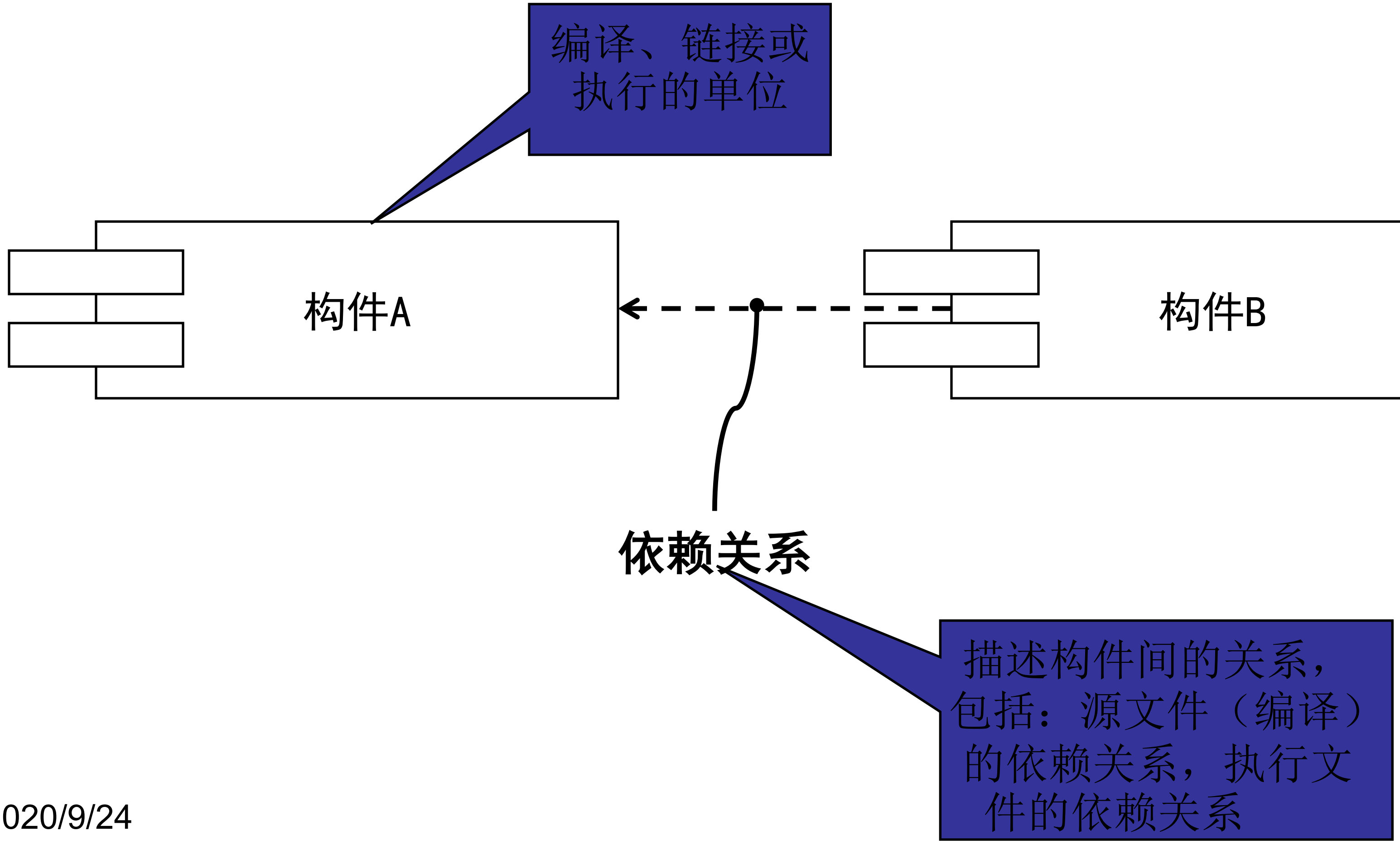
What Is a Component?

- A modular, deployable, and replaceable part of a system that encapsulates implementation and exposes a set of interfaces.
- It conforms to and provides the physical realization of a set of interfaces.

构件图

- 提供当前模型的物理视图，对系统的静态实现视图进行建模
- 从组织内容看，构件图显示软件构件的组织以及构件间的依赖关系
  - 源代码构件
  - 二进制代码构件
  - 可执行构件
- 构件图中，构件间的调用表示为构件间的依赖关系

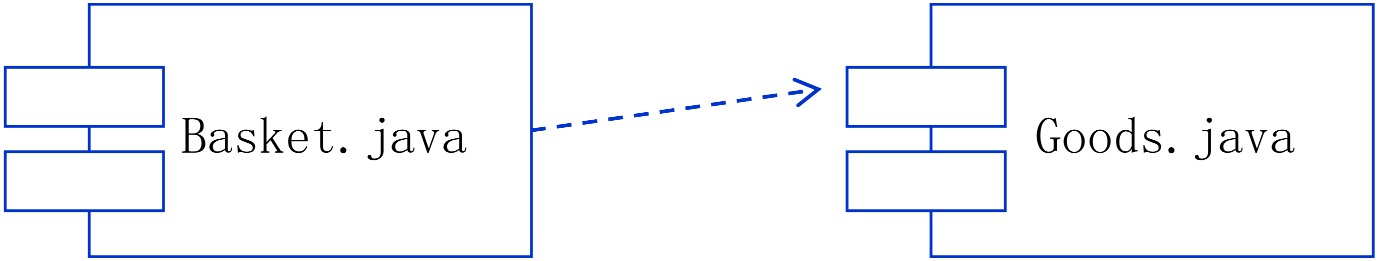
构件图的模型元素

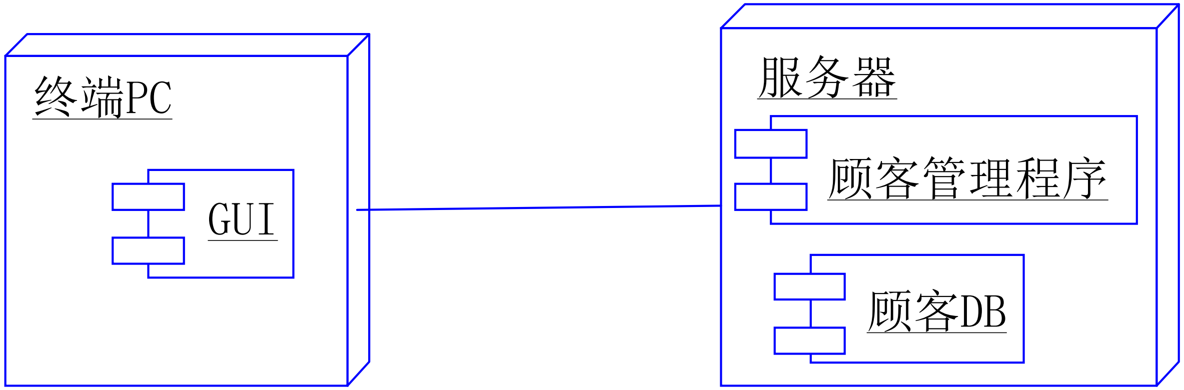


编译、链接或执行的单位

描述构件间的关系，包括：源文件（编译）的依赖关系，执行文件的依赖关系

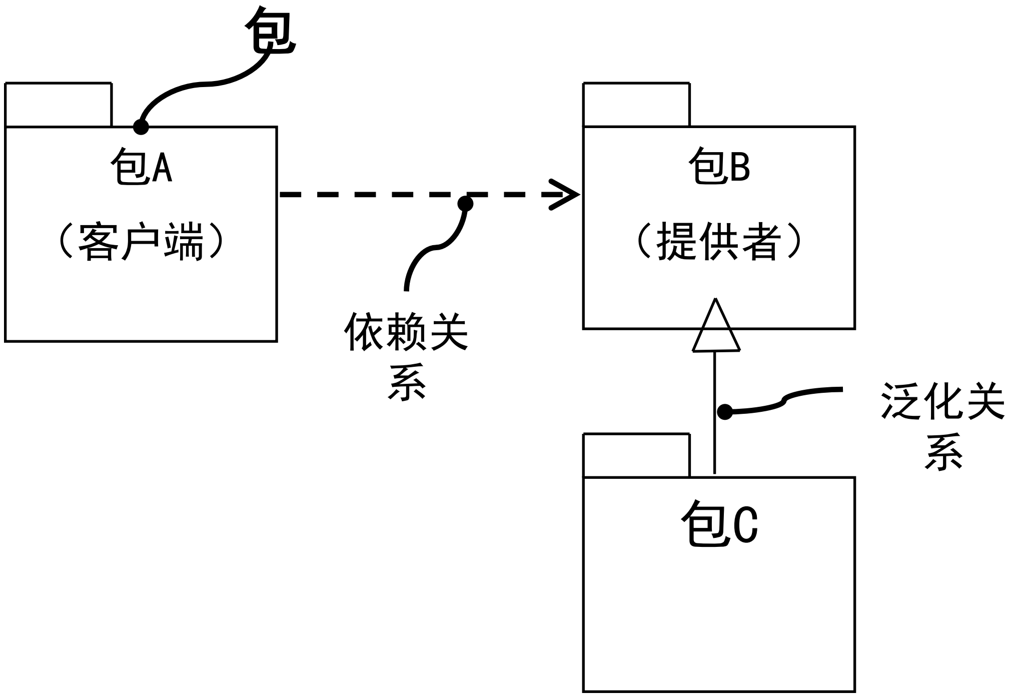
构件图和部署图示例





包

- 包是基于模型元素的含义或作用将模型元素分组的一种机制
- 目的:通过分组，可提高模型的维持性



Review

- Define state. How do you determine the classes with significant state?
- What is a statechart diagram? Describe the different parts of the diagram.
- How do statechart diagrams map to the rest of the world?
- What is the purpose of a deployment diagram?
- What is a component diagram?

作业 (4th week)

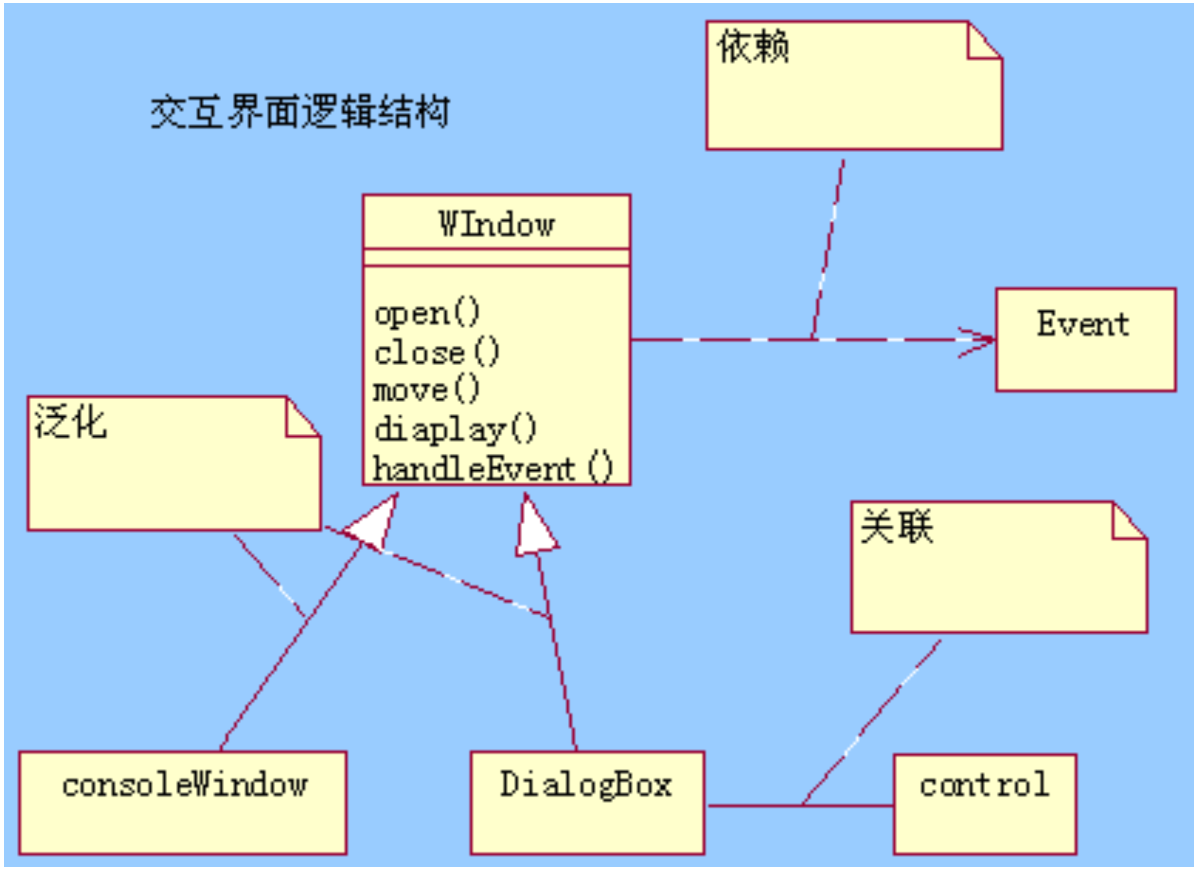
1. What is the state in statechart?
2. How do determine the classes with significant state?
3. What is the use of statechart? Describe the difference parts of the diagram.
4. What is the purpose of deployment diagram?
5. Please present the concept of Component, and what is the purpose of component diagram?
6. (练习作业，不必提交）请每位同学练习画deployment diagram，component diagram，and state chart.

什么是UML

- UML
  - Unified Modeling Language
  - 面向对象软件工程使用的统一建模语言
  - 一种图形化了的语言，主要用图形方式表示
  - 一种开放的标准
- 主要创始人
  - Jim Rumbaugh
  - Ivar Jacobson
  - Grady Booch
- 发展历程
  - 1997年确立的OMG(Object Management Group)标准
  - 2003年发行的UML1.5最新版
  - 2004年预定发行UML2.0版

UML的特点

- 统一标准
- 面向对象
- 可视化
- 表达能力强

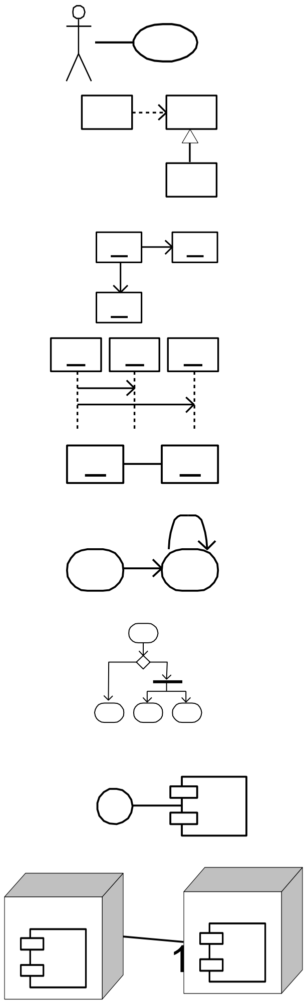


UML的应用

- UML在软件开发中的应用
  - 视化（visualize）
  - 说明(specify)
  - 建造(construct)
  - 建档(document)
- UML是一个通用的标准建模语言
  - 静态结构建模
  - 动态行为建模
  - 体系架构建模
- UML是一种建模语言
  - 不是一种方法，它独立于过程
  - 可遵循任何类型的建模过程

UML的图

分类(使用地方)	名称		描述内容
捕获需求内容 （用于需求分析）	用例图		描述系统提供的功能以及和外部元素的关联
捕获逻辑结构 (用于分析，设计)	类图		描述类的定义以及类间的关系
	对象图		描述某一时刻的对象状态
捕获行为 (用于分析，设计)	交互图	时序图	按时间顺序描述对象间的交互动作
		协作图	以连接关系描述对象间的交互动作
	状态图		描述一个对象的状态转换
	活动图		描述工作流和事件流以及算法
捕获实现执行环境 （用于设计）	构件图		描述源代码和模块的结构以及文件间的依赖关系
	部署图		描述系统运行的执行环境和执行模块的分配



HomeWork

- Draw with Rational Rose
  - Class diagram
  - Sequence diagram
  - Collaboration diagram
  - Use case diagram
  - Activity diagram

HomeWork

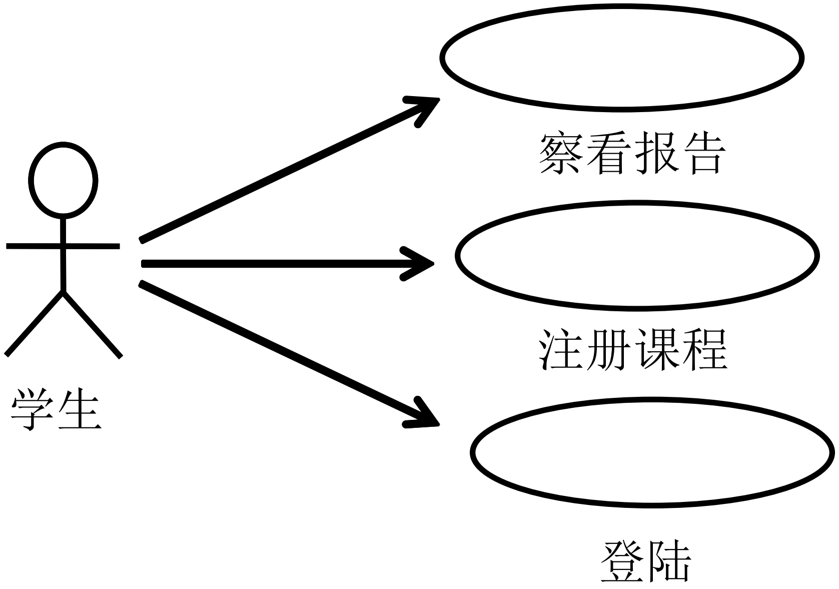
- Draw with Rational Rose
  - Class diagram
  - Sequence diagram
  - Collaboration diagram
  - Use case diagram

例题：试画用例图

- 想描述的内容如下：

在课程注册系统中，学生使用该系统可以进行登陆系统，注册课程和查看报告的操作。

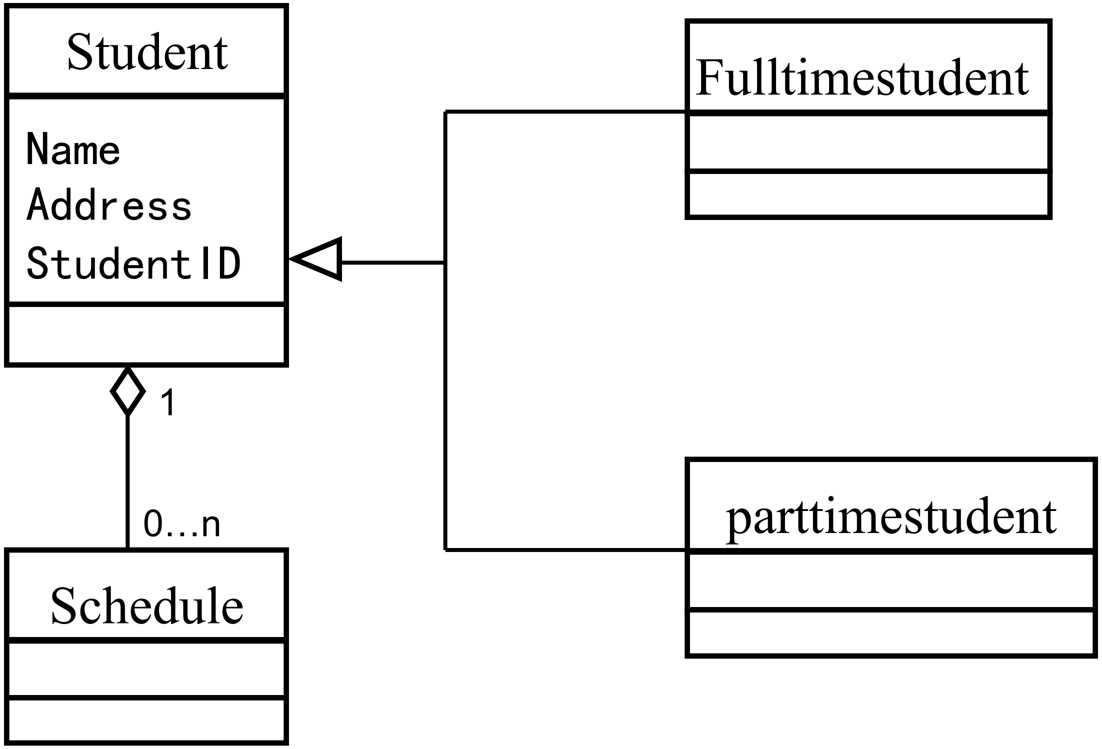
例题：试画用例图（题解）



例题：试画类图

- 想描述的内容
  - 有学生，课程表这两个类
  - 学生又分为fulltime和parttime两类
  - 一个学生包含有0到多张课程表；一张课程表只属于一个学生
  - 学生有属性姓名，地址和学号

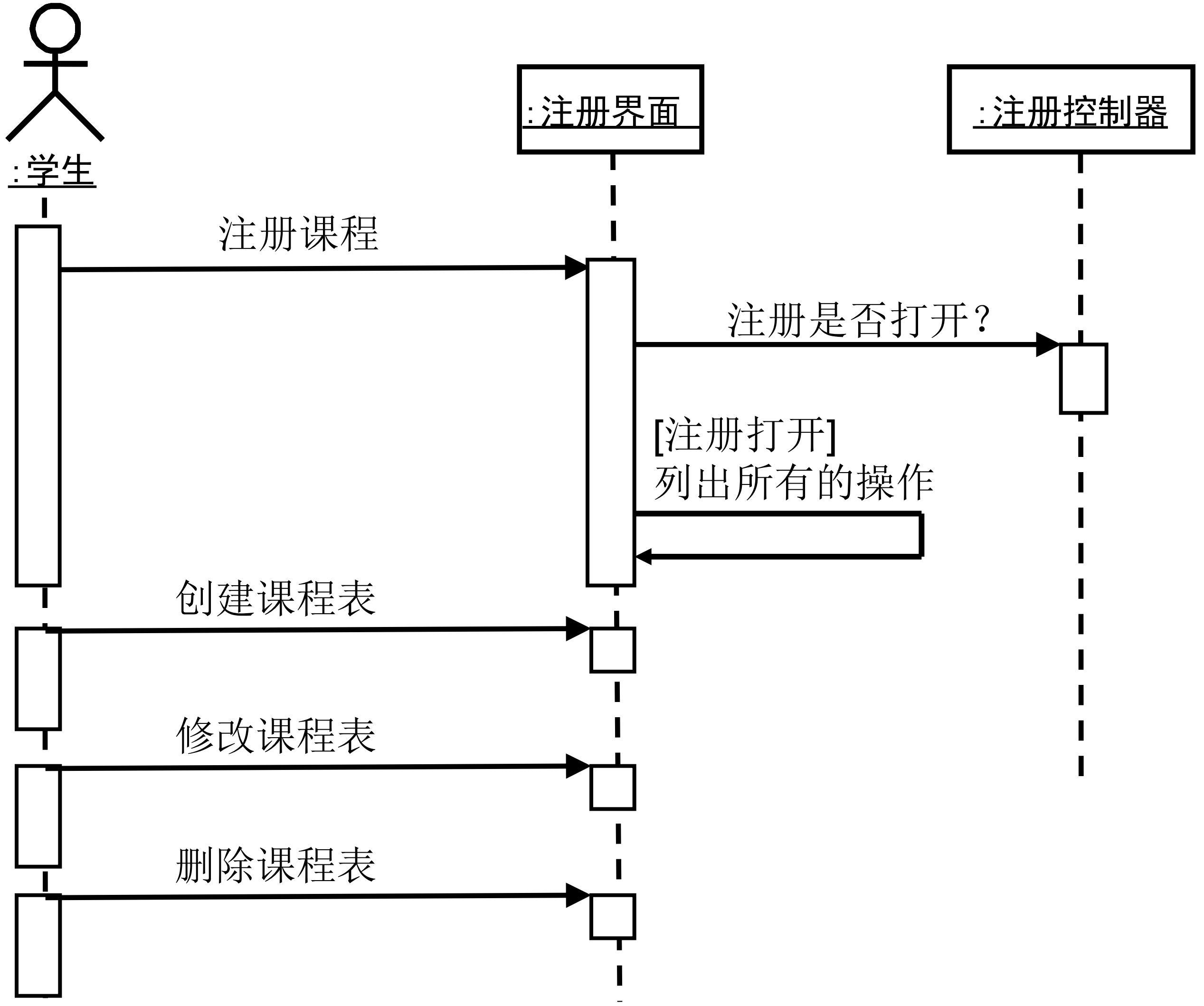
例题：试画类图（题解）


Student
Name
Address
StudentID
Fulltimestudent
1
0…n
Schedule
parttimestudent

例题：试画时序图

- 试画注册课程的时序图
- 学生首先发出注册课程的请求，注册控制器判断注册是否打开。如果打开则列出学生可以进行的操作，包括对课程表的创建，删除和修改。

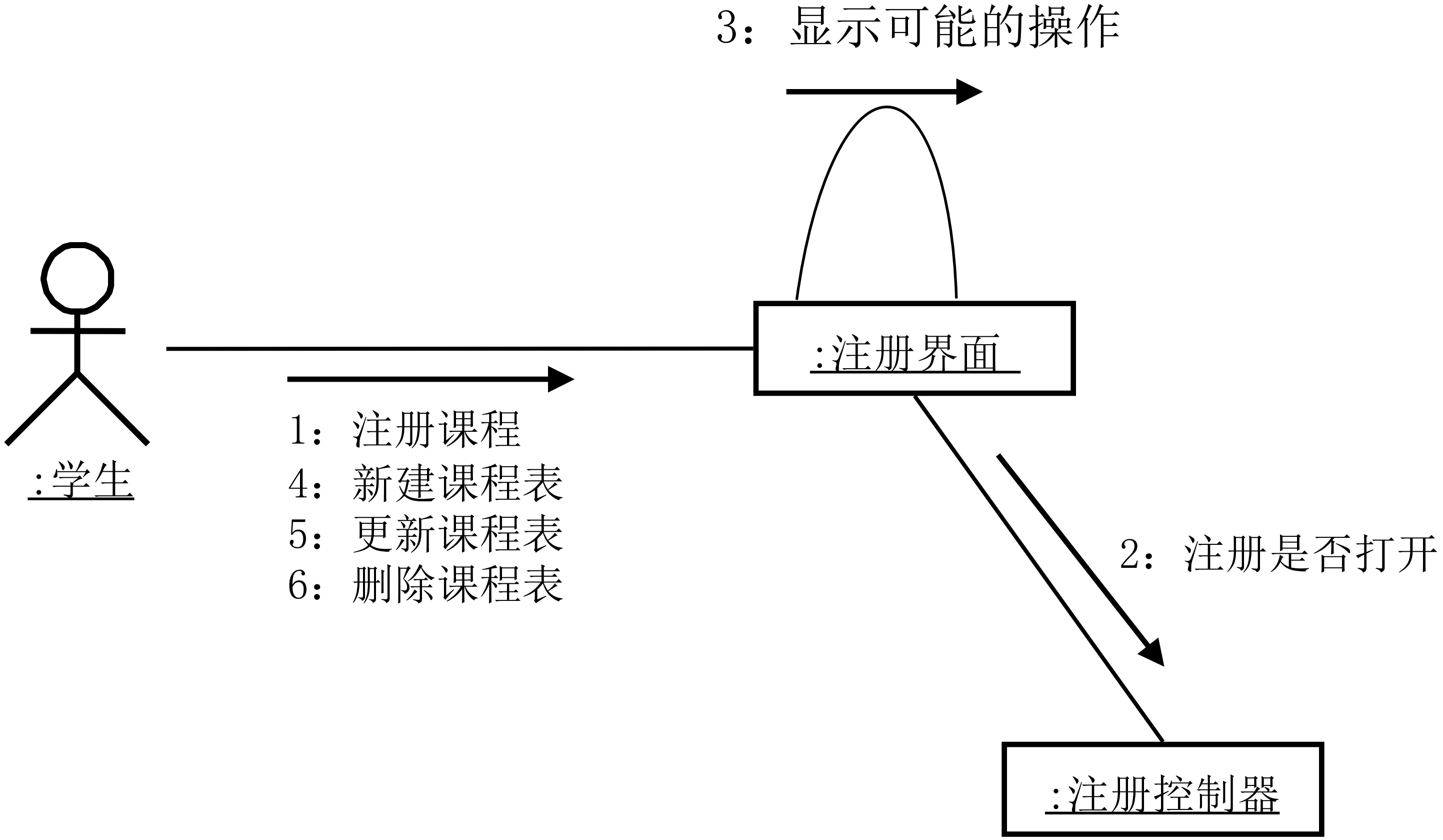
例题：试画时序图（题解）



例题：试画协作图

- 要求同上例，画出注册课程的协作图

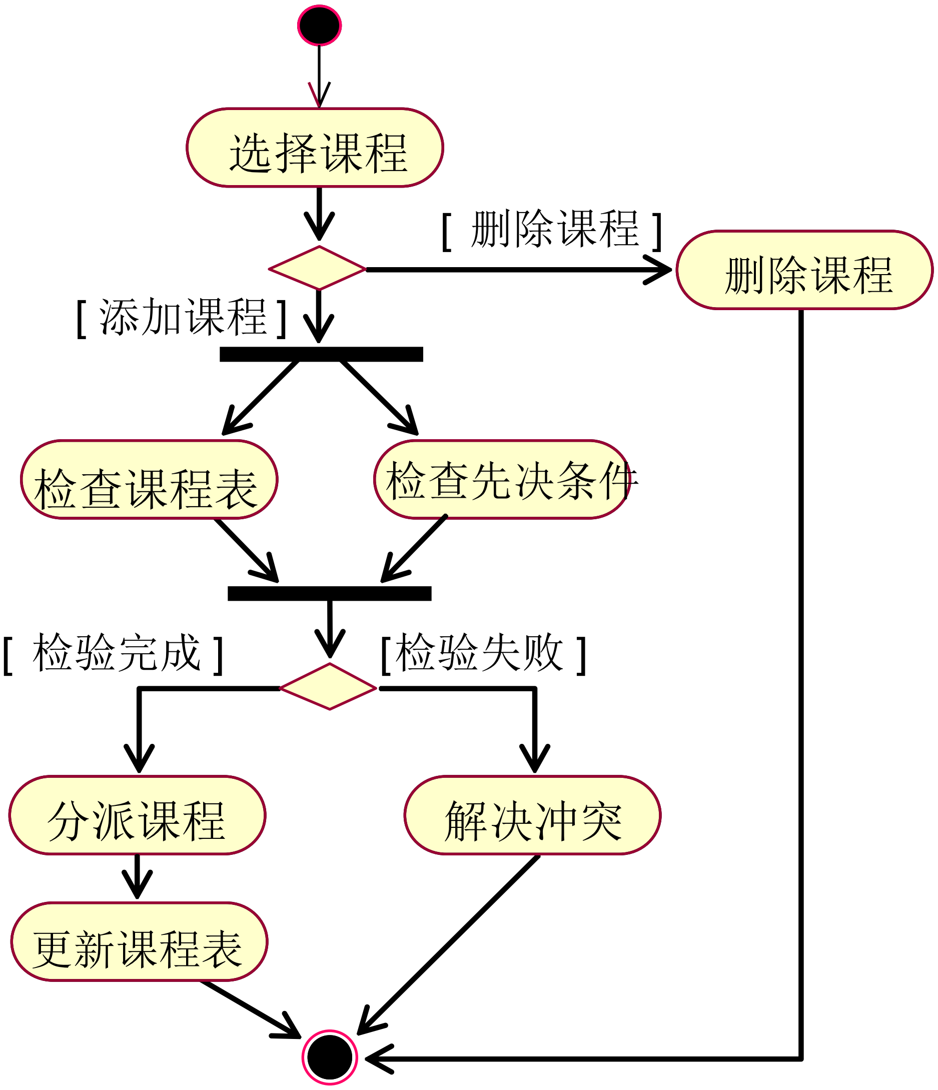
例题：试画协作图（题解）



例题：试画活动图

- 修改课程表的活动图
- 首先学生选择课程，并选定是添加还是删除该课程。若为删除则直接结束；若为添加，则需要检查课程表和先决条件，满足则分配课程并更新课程表；不满足需要解决冲突。

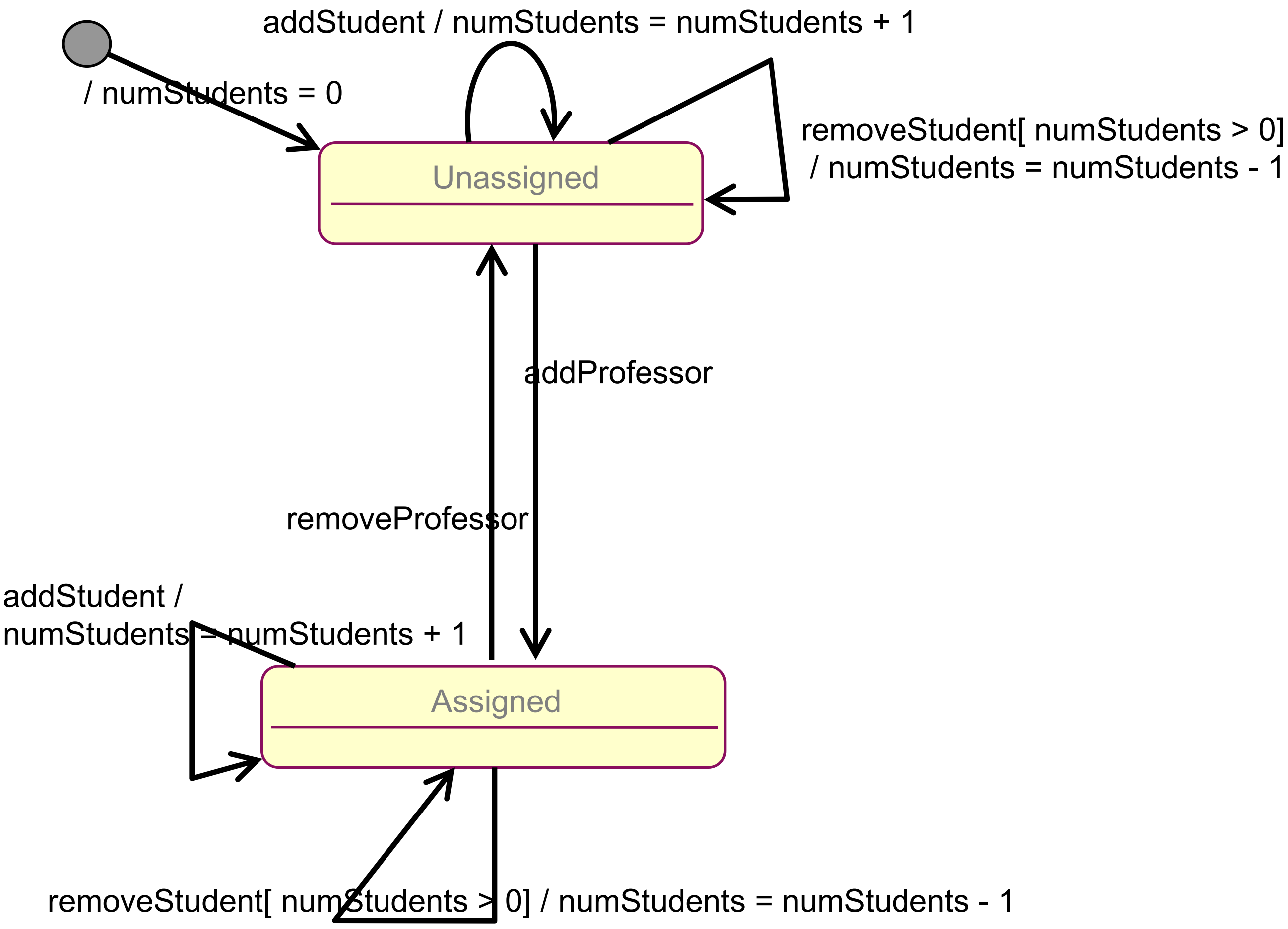
例题：试画活动图（题解）



例题：试画状态图

- 试画课程提供类的状态图
- 课程提供类的状态有分配和未分配两种，取决于是否添加了教授。两种状态都有添加学生和移除学生的事件。

例题：试画状态图（题解）


addStudent / numStudents = numStudents + 1
/ numStudents = 0
removeStudent[numStudents > 0]
/ numStudents = numStudents - 1
Unassigned
addProfessor
removeProfessor
addStudent /
numStudents = numStudents + 1
Assigned
removeStudent[numStudents > 0] / numStudents = numStudents - 1

例题：试画类图

- 想描述的内容
  - 有图书馆、书、作者这几个类
  - 图书馆、书、作者有各自的名字
  - 书上有图书编号
  - 作者有自己的经历
  - 从图书馆的角度来看，书具有藏书的作用
  - 作者写书

homework

- Based on the problem statement you prepared, provide following:
- Usecase diag., sequence diag., activity diag., class diag., and so on.